MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2014/04.14.18.23-PUD

INTRODUÇÃO AO NCAR COMMAND LANGUAGE (NCL)

José Guilherme Martins dos Santos

URL do documento original:
<http://urlib.net/8JMKD3MGP5W34M/3G5LGP5>

INPE
São José dos Campos
2018

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GBDIR)
Serviço de Informação e Documentação (SESID)
CEP 12.227-010
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/7348
E-mail: pubtc@inpe.br

COMISSÃO DO CONSELHO DE EDITORAÇÃO E PRESERVAÇÃO DA PRODUÇÃO INTELECTUAL DO INPE (DE/DIR-544):
Presidente:
Dr. Marley Cavalcante de Lima Moscati - Centro de Previsão de Tempo e Estudos Climáticos (CGCPT)
Membros:
Dra. Carina Barros Mello - Coordenação de Laboratórios Associados (COCTE)
Dr. Alisson Dal Lago - Coordenação-Geral de Ciências Espaciais e Atmosféricas (CGCEA)
Dr. Evandro Albiach Branco - Centro de Ciência do Sistema Terrestre (COCST)
Dr. Evandro Marconi Rocco - Coordenação-Geral de Engenharia e Tecnologia Espacial (CGETE)
Dr. Hermann Johann Heinrich Kux - Coordenação-Geral de Observação da Terra (CGOBT)
Dra. Ieda Del Arco Sanches - Conselho de Pós-Graduação - (CPG)
Silvia Castro Marcelino - Serviço de Informação e Documentação (SESID)
BIBLIOTECA DIGITAL:
Dr. Gerald Jean Francis Banon
Clayton Martins Pereira - Serviço de Informação e Documentação (SESID)
REVISÃO E NORMALIZAÇÃO DOCUMENTÁRIA:
Simone Angélica Del Ducca Barbedo - Serviço de Informação e Documentação (SESID)
André Luis Dias Fernandes - Serviço de Informação e Documentação (SESID)
EDITORAÇÃO ELETRÔNICA:
Marcelo de Castro Pazos - Serviço de Informação e Documentação (SESID)
Murilo Luiz Silva Gino - Serviço de Informação e Documentação (SESID)

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2014/04.14.18.23-PUD

INTRODUÇÃO AO NCAR COMMAND LANGUAGE (NCL)

José Guilherme Martins dos Santos

URL do documento original:
<http://urlib.net/8JMKD3MGP5W34M/3G5LGP5>

INPE
São José dos Campos
2018

Esta obra foi licenciada sob uma Licença Creative Commons Atribuição-NãoComercial 3.0 Não Adaptada.

This work is licensed under a Creative Commons Attribution-NonCommercial 3.0 Unported License.

Sumário

1 Introdução **9**

2 Informações sobre os arquivos NetCDF **11**

3 Instalação **14**

4 Visão geral **18**

4.1 Palavras-chave reservadas 19
4.2 Scripts prontos e dados disponibilizados 19
4.3 Executando o NCL 19
4.3.1 Modo interativo 20
4.3.2 Modo batch 21
4.4 Extensões disponíveis para saída gráfica 21
4.5 Tipos de dados e precisão numérica 23
4.6 Expressões numéricas 25
4.7 Variáveis 25
4.8 Loops 25
4.9 Condicionais 26
4.10 Parando a execução do loop 26
4.11 Dimensão e indexação 27
4.12 Redução de dimensão 28
4.13 Símbolos especiais do NCL 29
4.13.1 Símbolo ; 29
4.13.2 Símbolo @ 30
4.13.3 Símbolo ! 30
4.13.4 Simbolo & 31
4.13.5 Simbolo {...} 31
4.13.6 Simbolo $...$ 31
4.13.7 Símbolo [/.../] 32
4.13.8 Símbolo (/.../) 32
4.13.9 Símbolo : 32
4.13.10 Símbolo | 33
4.13.11 Símbolo \ 33
4.13.12 Símbolo -> 33
4.13.13 Símbolo = ... 33
4.13.14 Símbolo := 33
4.13.15 Símbolo [:] 34
4.14 Sobre _FillValue 35
4.15 Arquivos suportados 37
4.16 Imprimindo o conteúdo de um arquivo 38
4.17 Sobre os recursos utilizados no NCL 39

5 Visualizando as linhas de grade **41**

6 Explorando o site do NCL e obtendo ajuda **42**

7 Inserindo acentos em português **43**

8 Superescrito e subescrito **45**

9 Cores, fontes, estilos de linha, marcadores e projeções **46**
9.1 Tipos de cores 46
9.2 Tipos de fonte 47
9.3 Estilos de linha 48
9.4 Padrão de preenchimento 49
9.5 Tipos de marcadores 50
9.6 Tipos de projeções 51

10 Manipulando arquivo texto **52**
10.1 Lendo um arquivo texto com o asciiread 52
10.2 Retornando o número de linhas e colunas de um arquivo texto 56
10.3 Salvando arquivo texto 57
10.3.1 Usando a função write_table 57
10.3.2 Outras possibilidades de escrita no formato texto 58
10.4 Manipulando strings 58
10.5 Funcão str_fields_count 58
10.6 Funções str_get_cols e str_get_field 59
10.7 Função strs_sub_str 62

11 Salvando e manipulando arquivo NetCDF **63**
11.1 Salvando arquivo no formato NetCDF 63
11.2 Método simples 63
11.3 Método avançado 64
11.4 Editando o arquivo NetCDF 65

12 Convertendo dados grib para NetCDF **67**

13 Informações diversas usando as funções get **69**
13.1 Função get_cpu_time 69
13.2 Função get_file_suffix 69
13.3 Função get_unique_values 69
13.4 Função getfiledimsizes 70
13.5 Função getfilevaratts 70
13.6 Função getfilevarnames 71
13.7 Função getfilevartypes 72
13.8 Função get1Dindex 72
13.9 Função get1Dindex_Collapse 73
13.10 Função getvardims 74

14 Funções do NCL **74**
14.1 Função addfile 74
14.2 Função addfiles 75
14.3 Função all 77

14.4 Função any . . . 77
14.5 Função avg . . . 78
14.6 Função para anomalia . . . 78
14.6.1 Anomalia diária . . . 78
14.6.2 Anomalia mensal . . . 79
14.7 Função calculate_daily_values . . . 82
14.8 Função calculate_monthly_values . . . 83
14.9 Função calculate_segment_values . . . 85
14.10 Funcão ceil . . . 87
14.11 Função para climatologia . . . 87
14.11.1 Climatologia diária usando a função clmDayTLL . . . 87
14.11.2 Climatologia diária usando a função clmDayTLLL . . . 89
14.11.3 Climatologia mensal usando a função clmMonLLLT . . . 90
14.12 Função conform . . . 92
14.13 Função copy_VarCoords . . . 95
14.14 Função day_of_year . . . 97
14.15 Função days_in_month . . . 98
14.16 Função dewtemp_trh . . . 98
14.17 Função dim_acumrun_n . . . 99
14.18 Função dim_avg_n_Wrap . . . 100
14.19 Funções dim_min_n_Wrap e dim_max_n_Wrap . . . 101
14.20 Função dim_rmsd_n_Wrap . . . 102
14.21 Função dim_rmvmean_n_Wrap . . . 103
14.22 Função dim_spi_n . . . 104
14.23 Função dim_stddev_n_Wrap . . . 106
14.24 Função dim_sum_n_Wrap . . . 106
14.25 Função dimsizes . . . 106
14.26 Função fabs . . . 107
14.27 Função fileexists . . . 107
14.28 Função fspan . . . 108
14.29 Funcão ind . . . 108
14.30 Função ind_nearest_coord . . . 109
14.31 Funções lógicas de checagem . . . 109
14.31.1 Função iscoord . . . 109
14.31.2 Função isdim . . . 110
14.31.3 Função isfloat . . . 111
14.32 Função latGlobeF . . . 111
14.33 Função latGlobeFo . . . 112
14.34 Função lonFlip . . . 114
14.35 Função lonGlobeF . . . 115
14.36 Função lonGlobeFo . . . 116
14.37 Função max . . . 117
14.38 Função maxind . . . 117
14.39 Função min . . . 117
14.40 Função minind . . . 117
14.41 Função mixhum_ptd . . . 118

14.42 Função mixhum_ptrh . . . 118
14.43 Função mod . . . 119
14.44 Função month_to_annual . . . 119
14.45 Função month_to_annual_weighted . . . 120
14.46 Função month_to_season . . . 121
14.47 Função month_to_seasonN . . . 121
14.48 Função ndtooned . . . 122
14.49 Função new . . . 122
14.50 Função num . . . 124
14.51 Função omega_to_w . . . 124
14.52 Função onedtond . . . 126
14.53 Função pot_temp . . . 128
14.54 Função pot_vort_isobaric . . . 129
14.55 Função prcwater_dp . . . 130
14.56 Função printMinMax . . . 131
14.57 Função printVarSummary . . . 131
14.58 Função qsort . . . 132
14.59 Função relhum . . . 133
14.60 Função rmAnnCycle1D . . . 134
14.61 Função rmMonAnnCycTLL . . . 134
14.62 Função round . . . 135
14.63 Função runave_n_Wrap . . . 135
14.64 Função smthClmDayTLL . . . 136
14.65 Função stdatmus_p2tdz . . . 137
14.66 Função stdatmus_z2tdp . . . 138
14.67 Função stdMonTLL . . . 139
14.68 Função system . . . 139
14.69 Função systemfunc . . . 140
14.70 Função uv2dvF_Wrap . . . 140
14.71 Função uv2sfvpF . . . 140
14.72 Função uv2vrdvF . . . 142
14.73 Função vr2uvF_Wrap . . . 143
14.74 Função where . . . 144
14.75 Função wgt_areaave_Wrap . . . 145
14.76 Função wind_direction . . . 146

15 Criando máscaras 147
15.1 Mascarando o continente . . . 147
15.2 Mascarando valores . . . 149

16 Conversão entre diferentes tipos de dados 150
16.1 Funções para coerção entre dados . . . 152
16.1.1 Função floattochar . . . 152
16.1.2 Função floattoint . . . 152
16.1.3 Função integertochar . . . 153
16.2 Funções para conversão entre dados . . . 153

16.2.1 Função dble2flt 153
16.2.2 Função flt2dble 153
16.2.3 Função flt2string 154
16.2.4 Função int2flt 154
16.2.5 Função numeric2int 155
16.2.6 Função short2flt 155
16.2.7 Função todouble 156
16.2.8 Função tofloat 156
16.2.9 Função toint 156
16.2.10 Função tostring 156
16.2.11 Função totype 157

17 Interpolação e regrid **157**
17.1 Qual é o melhor método para interpolação/regrid? 157
17.2 Interpolação 159
17.2.1 Função int2p_Wrap 159
17.2.2 Função poisson_grid_fill 161
17.3 Regrid 162
17.3.1 Função f2fosh_Wrap 163
17.3.2 Função f2foshv_Wrap 163
17.3.3 Função f2fsh_Wrap 165
17.3.4 Função f2fshv_Wrap 167
17.3.5 Função fo2fsh_Wrap 170
17.3.6 Função fo2fshv_Wrap 170
17.3.7 Função g2fsh_Wrap 172
17.3.8 Função g2fshv_Wrap 175
17.3.9 Função linint2_Wrap 177
17.3.10 Função rcm2rgrid 180

18 Criando painéis **183**

19 Interpretação de erros **197**

20 Técnicas de plot **202**
20.1 Posicionamento da figura na página 202
20.2 Explicação sobre o recurso frame 205
20.3 Gerando animações 208
20.4 Adicionando anotações ao plot ou ao frame 210
20.5 Anexando um mapa a outro mapa 211
20.6 Inserindo uma legenda no gráfico de barras 213
20.7 Anexando um gráfico de linha ao mapa 215
20.8 Alterando o tamanho das fontes do gráfico 217
20.9 Personalização dos rótulos da legenda 224
20.10 Polígonos 229
20.10.1 Desenhando uma caixa sobre o mapa 230
20.10.2 Desenhando marcadores sobre um mapa 231
20.11 Adicionando e formatando texto 233

20.11.1 Utilizando as coordenadas da figura 233
20.11.2 Formatando a posição do texto da figura 234
20.11.3 Utilizando as coordenadas NDC 236
20.11.4 Adicionando quebra de linha no texto do gráfico 237
20.12 Personalizando os marcadores de escala (tickmarks) 238
20.13 Personalizando os títulos da figura 242

21 Geração de gráficos 249
21.1 O que preciso saber para criar um gráfico? 251
21.1.1 Tipos de gráficos de linha 252
21.1.2 Tipos de gráficos de barras 253
21.1.3 Tipos de gráficos espaciais 253
21.2 Gráficos de linha 254
21.2.1 Gráficos de linha com um eixo y 254
21.2.2 Gráfico de linha com três eixos y 261
21.2.3 Gráfico de linha com dois eixos x e y 263
21.2.4 Gráfico de linha com um eixo x e dois eixos y 265
21.2.5 Gráfico de linha com tendência linear 267
21.2.6 Gráfico de linha com desvio padrão 270
21.3 Gráficos de barra 272
21.3.1 Gráfico barra simples 272
21.3.2 Gráfico com duas barras 274
21.3.3 Gráfico de barras com anomalia 277
21.4 Gráficos espaciais 279
21.4.1 Campos escalares 279
21.4.2 Campos vetoriais 283
21.4.3 Campos polares 288
21.4.4 Campos de linhas de corrente 295
21.5 Seção vertical 299
21.5.1 Pressão/altura versus latitude 299
21.5.2 Pressão/altura versus longitude 305
21.5.3 Pressão/altura versus tempo 311
21.5.4 Longitude versus tempo (Hovmöeller) 317
21.6 Usando shapefile 318
21.6.1 Usando shapefile para mascarar dados 318
21.6.2 Extrair a série temporal de uma área usando shapefile 321
21.7 Radiossondagem 323
21.7.1 Plotando uma radiossondagem 323
21.7.2 Plotando duas radiossondagens 326
21.8 Meteogramas 329
21.9 Gráfico de pizza 334
21.10 Diagrama de Taylor 337
21.11 Rosa dos ventos 340
21.12 Boxplot 344
21.13 Sobreposição de campos 348

22 Estatística com o NCL **353**
22.1 Correlação 353
22.1.1 Correlação de Pearson entre dois campos multidimensionais . 353
22.1.2 Correlação de Pearson entre uma série temporal e um dado multidimensional com teste de significânia 356
22.2 Empirical Orthogonal Functions (EOF) 358
22.2.1 EOF utilizando ponderação 361
22.2.2 EOF sem ponderação 366
22.3 Lanczos Filter Weights 370
22.3.1 Filtragem em uma dimensão (tempo) 370
22.4 Função Distribuição de Probabilidade (PDF) 378
22.5 Análise Espectral (Spectral Analysis) 384

23 Links interessantes **390**

24 Âpendice **391**
24.1 Âpendice 1 - NCL Reference Cards 391

Agradecimentos

Esse documento é o resultado de muitos anos de pesquisa, dúvidas obtidas da internet, conversas com amigos e colegas de trabalho, como também das dificuldades encontradas que me fizeram seguir adiante neste projeto ambicioso em que muitas vezes tive vontade de desistir. Toda essa situação fez com que eu quase parasse de trabalhar com essa ferramenta fantástica, porém com a persistência eu segui adiante, e hoje, posso concretizar mais um projeto em minha em vida.

A ideia de documentar o conhecimento adquirido surgiu da necessidade de se ter algum documento em língua portuguesa, uma vez que, o NCAR Command Language (NCL) foi todo desenvolvido em língua inglesa, e para quem não tem conhecimento básico em inglês, esse documento veio em boa hora para ajudar aqueles que desejam utilizar uma ferramenta poderosa com suas diversas funcionalidades. A literatura portuguesa é escassa no que diz respeito a livros técnicos de programação voltado para Meteorologia, por isso, o motivo de tornar esse conhecimento público.

Agradeço imensamente a todas as oportunidades que tive em ministrar o curso de NCL, pois essa primeira experiência em transferir o conhecimento adquirido foi muito importante para saber se eu estava no caminho correto. Lembro-me perfeitamento que em 2013 a convite do Cláudio Santos e Silva e da Neusa Lemes fui convidado para ministrar o meu primeiro curso na Universidade Federal do Rio Grande do Norte (UFRN) no Programa de Pós-Graduação em Ciências Climáticas (PPGCC) que me projetou para as próximas experiências como docente. Agradeço também ao Silvio Nilo do Centro de Previsão de Tempo e Estudos Climáticos (CPTEC) do Instituto Nacional de Pesquisas Espaciais (INPE) por me proporcionar a oportunidade de ensinar aos alunos como também aos funcionários dessa instituição. Agradeço também a Sociedade Brasileira de Meteorologia (SBMET) por me convidar a ministrar o curso no VI Simpósio Internacionai de Climatologia que ajudou a divulgar ainda mais a ferramenta.

Com o NCL tive a oportunidade de capacitar mais de 200 pessoas e a avaliação do curso sempre foi excelente e isso somente foi possível graças ao empenho de todos e pelo enorme interesse pela ferramenta.

Não poderia deixar de agradecer a Mary Haley e Dennis Shea por tornar possível minha visita ao *National Center for Atmospheric Research* (NCAR), pelo grande incentivo e pelas dicas importantíssimas para a melhoria desse documento.

Espero que todos gostem dessa publicação porque ela foi escrita com muita dedicação pensando em todos os tipos de usuários. São mostrados vários exemplos comentados com um número considerável de ilustrações para facilitar a absorção da informação.

Assim, espero que esse documento possa auxiliar os interessados em NCL a processar e visualizar os seus dados porque pela minha experiência, é uma das melhores ferramentas que conheci e recomendo o seu uso.

O meu muito obrigado à todos pelas oportunidades em transferir o meu conhecimento, e hoje, posso dizer que deixo uma contribuição para a Meteorologia brasileira.

1 Introdução

Dentre as ferramentas de trabalho para manipular e visualizar dados atmosféricos, o *NCAR Command Language* ou simplesmente NCL tem se mostrado uma ferramenta altamente poderosa com suas diversas funções, scripts e uma variedade de opções disponíveis no site para manipular e visualizar dados dos mais variados formatos. Ele é um software grátis que facilita sua ampla divulgação no meio científico. Além disso, há uma excelente lista de discussão onde os usuários estão sempre dispostos a ajudar.

O NCL foi desenvolvido pelo *Computational & Information Systems Laboratory*, pertencente ao *National Center for Atmospheric Research* (NCAR) financiado pelo *National Science Foundation*. É uma linguagem interpretada livre com o objetivo de processar dados científicos como também sua visualização. Esta disponível para os sistemas operacionais Linux/UNIX, MacOS e Windows (Cygwin). Há mais de 600 funções e procedimentos para manipulação de dados que incluem subrotinas para o cálculo de:

- *Empirical Ortogonal Functions* (EOF), médias, desvio padrão, seno, cosseno, máximo, mínimo;
- Personalização de linhas, marcadores, símbolos de tempo, isolinhas, dentre outros;
- Interpolação, aproximação e regrid de dados 1D, 2D e 3D dentre outras possibilidades.

Outra característica é o suporte a chamadas externas de programas em linguagem C e Fortran.

Todas essas funções e procedimentos mostram o quanto essa ferramente é robusta e completa.

O objetivo desse documento consiste em fornecer os passos iniciais para aqueles que desejam utilizar o NCL. Desse modo, serão abordados exemplos práticos onde os scripts serão comentados com o máximo de detalhamento. E ao mesmo tempo, serão fornecidas informações adicionais aos usuários para a criação e personalização deles.

Atualmente, observa-se um número cada vez maior de usuários ao redor do globo que estão aderindo ao NCL. No Brasil, esse número vem crescendo, e espera-se que dentro de poucos anos, essa ferramenta torne-se conhecida e utilizada por grande parte da comunidade científica, particularmente, a ciências atmosféricas.

O livro está organizado da seguinte forma:

- No capítulo 2 é apresentado uma visão geral sobre arquivo NetCDF.
- No capítulo 3 é mostrado como instalar o NCL no sistema operacional Linux Ubuntu 64 bits.

- No capítulo 4 é mostrado uma visão geral do NCL, como por exemplo, como executá-lo, expressões numéricas, loops, símbolos especiais, arquivos suportados e recursos utilizados.
- No capítulo 5 é mostrado como utilizar as linhas de grade que são úteis para inserir por exemplo, textos na figura.
- No capítulo 6 é mostrado como explorar o site do NCL.
- No capítulo 7 é mostrado como acentuar palavras em português.
- No capítulo 8 é mostrado como utilizar superescrito e subescrito.
- No capítulo 9 é mostrado como utilizar tabela de cores, fontes, estilos de linha, marcadores e projeções.
- No capítulo 10 é mostrado como manipular arquivo texto.
- No capítulo 11 é mostrado como salvar e manipular arquivo NetCDF.
- No capítulo 12 é mostrado como converter dados grib para NetCDF.
- No capítulo 13 são mostradas aplicações da função get.
- No capítulo 14 é mostrado como utilizar as funções no NCL.
- No capítulo 15 é mostrado como crias máscaras.
- No capítulo 16 é mostrado como realizar a conversão entre diferentes tipos de dados.
- No capítulo 17 é mostrado como realizar interpolação e regridding.
- No capítulo 18 é mostrado como criar paíneis.
- No capítulo 19 é mostrado como interpretar erros.
- No capítulo 20 é mostrado como aplicar diferentes técnicas de plot para deixar o gráfico com aparência profissional.
- No capítulo 21 é mostrado como gerar diferentes tipos de gráficos.
- No capítulo 22 é mostrado alguns exemplos da aplicação de estatística.
- No capítulo 23 são mostrados alguns links importantes.

2 Informações sobre os arquivos NetCDF

O formato de dado utilizado neste livro é o NetCDF (Network Common Data Form) e por isso, é importante conhecer um pouco sobre a sua estrutura porque será muito útil na hora escrever os scripts.

Há diferentes tipos de convenções para os dados NetCDF, são elas: COARDS (*Cooperative Ocean/Atmosphere Research Data Service*) e CF (*Climate and Forecast Metadata Convention*). Informações adicionais podem ser obtidas em:

COARDS: http://ferret.wrc.noaa.gov/noaa_coop/coop_cdf_profile.html

CF: http://www.cgd.ucar.edu/cms/eaton/netcdf/CF-20010629.htm

As convenções são importantes porque elas tornam a comparação entre os dados mais fáceis e facilitam a sua visualização (ncview, GrADS, dentre outras). A convenção COARDS foi criada para grade rectilinear (latitude e longitude possuem apenas uma dimensão), enquanto que a convenção CF é uma extensão da COARDS e tem suporte para grades mais complexas (curvilinear [latitude e longitude possuem duas dimensões] e não-estruturada) e vários tipos de calendários (no_leap, 350_days, 365_days e etc). A maioria dos modelos climáticos (CMIP3, CMIP5, CESM, IPCC, etc) utilizam essa convenção.

Ao instalar a biblioteca NetCDF há inúmeros programas que podem ser utilizados para manipular e visualizar o conteúdo de um arquivo NetCDF. Eles são úteis para mostrar o conteúdo de um arquivo NetCDF. Para visualizar as informações sobre esse formato, basta digitar no seu terminal Linux: ncdump -h seu_arquivo.nc.

Exemplo: Visualizando o contéudo do arquivo olr.jan2000.dez2009.nc:

```
ncdump -h olr.jan2000.dez2009.nc | cat -n
```

O resultado do comando acima será:

```
1   netcdf olr.jan2000.dez2009 {
2   dimensions:
3           lat = 73 ;
4           lon = 144 ;
5           time = UNLIMITED ; // (120 currently)
6   variables:
7           float lat(lat) ;
8                   lat:units = "degrees_north" ;
9                   lat:actual_range = 90.f, -90.f ;
10                  lat:long_name = "Latitude" ;
11                  lat:standard_name = "latitude" ;
12                  lat:axis = "Y" ;
13          float lon(lon) ;
14                  lon:units = "degrees_east" ;
15                  lon:long_name = "Longitude" ;
16                  lon:actual_range = 0.f, 357.5f ;
17                  lon:standard_name = "longitude" ;
18                  lon:axis = "X" ;
19          short olr(time, lat, lon) ;
20                  olr:long_name = "OLR monthly means" ;
21                  olr:unpacked_valid_range = 0.f, 500.f ;
22                  olr:actual_range = 84.04f, 330.16f ;
23                  olr:units = "W/m^2" ;
24                  olr:add_offset = 327.65f ;
25                  olr:scale_factor = 0.01f ;
26                  olr:missing_value = 32766s ;
27                  olr:var_desc = "Outgoing Longwave Radiation" ;
28                  olr:precision = 2s ;
29                  olr:dataset = "NOAA Interpolated OLR" ;
30                  olr:level_desc = "Other" ;
31                  olr:statistic = "Mean" ;
32                  olr:parent_stat = "Individual Obs" ;
33                  olr:valid_range = -32765s, 17235s ;
34          double time(time) ;
35                  time:units = "hours since 1800-01-01 00:00:0.0" ;
36                  time:long_name = "Time" ;
37                  time:actual_range = 1753152., 1840080. ;
38                  time:delta_t = "0000-01-00 00:00:00" ;
39                  time:avg_period = "0000-01-00 00:00:00" ;
40                  time:standard_name = "time" ;
41                  time:axis = "T" ;
42
43  // global attributes:
44                  :title = "Monthly means of OLR from interpolated OLR dataset" ;
45                  :history = "Tue Mar 15 16:29:33 2016: ncks -O -d lat,-90.000000,90.000000 -d
    lon,0.000000,360.000000 -d time,307,426 /Datasets/interp_OLR/olr.mon.mean.nc /Public/www/
    X179.98.198.208.74.16.29.32.nc\n",
46                          "Created from daily OLR files obtained at NCEP and further processed.
    Stored in netCDF in 1996. Last update 10/2003" ;
47                  :description = "Data is interpolated in time and space from NOAA twice-daily
    values and averaged to once daily (from which means are calculated)" ;
48                  :platform = "Observation" ;
49                  :Conventions = "CF-1.2" ;
50                  :References = "http://www.esrl.noaa.gov/psd/data/gridded/data.interp_OLR.html
51                  :references = "http://www.esrl.noaa.gov/psd/data/gridded/data.interp_OLR.html
52                  :Citation = "Liebmann and Smith: June 2006: Description of a Complete
    (Interpolated) Outgoing Longwave Radiation Dataset. Bulletin of the American Meteorological
    Society, 77, 1275-1277" ;
53                  :NCO = "4.2.6" ;
54  }
```

Dimensões da variável

Nome da dimensão

Tipo do dado

Nome da variável

Atributos da variável

Atributos globais

Esse dado possui uma grade rectilinear porque a dimensão latitude e longitude possui apenas uma dimensão. Por exemplo, na linha 7 há as seguintes informações: float lat(lat). O nome (lat) entre parênteses define o tipo de grade da variável, e como possui apenas uma dimensão, isto é, apenas lat, define-se então como grade rectilinear. Outros exemplos de grade rectilinear são: modelos gaussianos e reanálises.

Os modelos com grade curvilinear são: WRF, POP, GODAS, RegCM e NARR que apresentam a latitiude e a longitude com duas dimensões.

Por exemplo, o modelo WRF apresenta **XLAT**(Time, south_north, west_east) e **XLONG**(Time, south_north, west_east).

Nas linhas 3 a 5 são mostrados os nomes e o tamanho das dimensões do arquivo. Nota-se que as dimensões lat, lon e time possuem 73 pontos, 144 pontos e 120 tempos, respectivamente. No caso do tempo, o *UNLIMITED* quer dizer que há possibilidade de aumentar o número de tempos dessa dimensão.

A partir das linhas 6 até a 41 são os nomes das variáveis do arquivo com o seu respectivo tipo (float, short, double, etc).

O nome entre parênteses é o nome da dimensão da variável (lat(lat) e lon(lon)).

A variável lat (sem parênteses) na linha 7 é do tipo float e das linhas 8 a 12 são os atributos dessa variável.

Na linha 13 é mostrada a variável lon que é do tipo float, e das linhas 14 a 18 são os seus atributos.

Na linha 19 é mostrada a variável olr que é do tipo short e é função da dimensão time, lat e lon, e das linhas 20 a 33 são os seus atributos.

E finalmente, a variável time que é do tipo double, e os seus atributos estão entre as linhas 35 e 41.

Todos os atributos podem ser alterados e as informações do tipo da variável são importantes porque em alguns casos é necessário fazer a descompactação do dado, por exemplo, do tipo short para float usando a função short2flt. Por isso, antes de tudo, é muito importante conhecer o conteúdo do arquivo e a sua estrutura para depois iniciar a importação dos dados e o seu processamento.

Há também os atributos globais (linhas 43 a 53) que fornecem informações variadas sobre o contato do responsável pelo dado, fonte dos dados, algum tipo de conversão, tipo de convenção utilizada, data de criação do dado, dentre outras possibilidades.

Uma regra muito importante antes de começar a utilizar o NCL é ter o conhecimento sobre a estrutura do dado, isto é, saber o que tem dentro do seu arquivo.

A maioria dos erros ocorrem por falta de informações sobre o arquivo a ser manipulado. Por isso, é muito importante conhecer primeiro o conteúdo do arquivo para depois iniciar o processamento.

Uma ferramente muito útil é o **ncl_filedump** que é nativo do NCL. Use-o para explorar o arquivo.

Exemplo de uso: **ncl_filedump seu_arquivo.nc**

3 Instalação

O NCL versão 6.4.0 utilizado nesse livro foi instalado no sistema operacional Linux Ubuntu 64 bits.

Tenha sempre a última versão do NCL instalada para aproveitar as últimas implementações feitas pelos desenvolvedores. Normalmente, são fornecidas novas funcionalidades, tabelas de cores, novos comandos.

É possível saber quais as novas funcionalidades da versão mais recente do NCL, para isso, basta clicar no menu *What's New* e depois em *In Latest Release*, o link está disponível em: http://www.ncl.ucar.edu.

Para realizar o download do NCL clique no link abaixo:

https://www.earthsystemgrid.org

No final da página, e na seção *Analysis & Visualization Software* clique em *NCL: NCAR Command Language* (Figura 7).

Figura 1: Site para realizar o download do NCL.

O usuário será direcionado para uma nova página que contém as versões a serem instaladas. Clique na versão mais recente do NCL, o formato utilizado é **NCL Version x.x.x**, em que *x.x.x* representa a versão do NCL. Por exemplo, *NCL Version 6.4.0*. Ao clicar na versão de interesse uma nova página será exibida, selecione *NCL Version 6.4.0 precompiled binaries (OPeNDAP-enabled)* que é a versão pré-compilada com todas as bibliotecas necessárias para manipular e visualizar dados. Clique em **Download Options** para visualizar a lista de arquivos disponíveis para download.

Qual versão utilizar dentre vários arquivos disponíveis?

Isso dependerá da arquitetura (32 ou 64 bits) do sistema operacional Linux. Para saber isso, digite o comando *uname -m* no terminal do Linux (o resultado será i686 ou x86_64). Se o sistema for 32 bits aparecerá o resultado *i686*, caso seja 64 bits, a informação será *x86_64*.

Além disso, o usuário precisar saber qual é a versão do gcc instalada, e para saber isso, basta digitar no seu terminal Linux o comando *gcc -v*. Será mostrada a versão

conforme a linha abaixo. Obtida a versão do gcc do computador e sabendo qual a arquitetura do seu sistema será selecionada a versão do NCL a ser instalada.

Resultado do comando *gcc -v*: gcc version 5.4.0 20160609

Resultado do comando *uname -m*: x86_64

Com base nas informações do gcc e da arquitetura, será selecionado o arquivo *ncl_ncarg-6.4.0-Debian8.6_64bit_gnu492.tar.gz*. Veja que este arquivo é para arquitetura 64 bits (x86_64) e para a versão do gcc mais próxima da versão do gcc do computador a ser instalado (versão 5.4.0). Quem usa Ubuntu selecione o nome do arquivo que contém Debian.

Após realizar o download do arquivo acima, descompacte-o com o comando abaixo. Apenas lembrando que o arquivo será salvo no diretório Download:

tar -zxvf ncl_ncarg-6.4.0-Debian8.6_64bit_gnu492.tar.gz

Serão criados os diretórios: *bin*, *lib* e *include*

Crie uma pasta chamada *ncarg* e mova os diretórios *bin*, *lib* e *include* para *ncarg*. Em seguida, mova a pasta *ncarg* para */usr/local* com o comando *sudo mv ncarg /usr/local*.

Acesse o endereço:

http://www.ncl.ucar.edu/Document/Graphics/hluresfile

E salve o arquivo no seu diretório *home*. Esse arquivo não deve conter nenhuma extensão, apenas o nome *hluresfile* porque ele é um arquivo de configuração do NCL responsável por alterar o tamanho da janela gráfica, tipo de fonte usada, cor, dentre outras opções. Fica a critério do usuário explorar suas características. Acesse o link abaixo para mais informações.

http://www.ncl.ucar.edu/Document/Graphics/hlures.shtml.

Com o arquivo *hluresfile* no seu *home*, proceda da seguinte forma:

mv hluresfile .hluresfile

Com isso, ele passará a ser um arquivo oculto do sistema. Para ver se ele está oculto, digite no seu *home* o comando *ls -a*, e você verá que o arquivo *.hluresfile* tem um ponto na frente dele que é uma característica dos arquivos ocultos do sistema.

O próximo passo será editar o arquivo *.bashrc* que também é um arquivo oculto do sistema porque tem um ponto na frente dele. Para visualizá-lo basta digitar *ls -a* no seu *home* e *adicione as linhas abaixo em vermelho caso não tenha a variável PATH no seu .bashrc*:

export NCARG_ROOT=/usr/local/ncarg

PATH=/usr/bin:/bin:/usr/local/bin:$PATH:/usr/bin/X11:/lib:/usr/lib:$NCARG_ROOT/bin

export PATH=$NCARG_ROOT/bin:$PATH

export LD_LIBRARY_PATH=/usr/local/lib:/usr/lib:/lib

Não esqueça de atualizar o seu *.bashrc* digitando no seu *home*:

source .bashrc

Agora, deve-se digitar ncl no terminal do Linux para saber se o programa foi instalado corretamente. Se surgir algum erro como nos exemplos abaixo, digite a solução para resolver. O que está em negrito são os erros e em itálico são as soluções:

ncl: error while loading shared libraries: libgfortran.so.3: cannot open shared object file: No such file or directory

Digite no seu terminal:

sudo apt update
sudo apt install libgfortran3 libgomp1

ou

ncl: error while loading shared libraries: libssh2.so.1: cannot open shared object file: No such file or directory

Digite no seu terminal: *sudo apt-get install libssh2-1*

ou

ncl: error while loading shared libraries: libssl.so.0.9.8: cannot open shared object file: No such file or directory

Digite no seu terminal: *sudo apt-get install libssh2-1*

ou

ncl: error while loading shared libraries: libcrypto.so.0.9.8: cannot open shared object file: No such file or directory

Digite no seu terminal: *sudo apt-get install libssl0.9.8*

ou

ncl: error while loading shared libraries: librtmp.so.0: cannot open shared object file: No such file or directory

O NCL está pedindo a biblioteca librtmp.so.0, mas no Linux só há a biblioteca librtmp.so.1, e a solução é criar um link simbólico para librtmp.so.0. Para localizar a biblioteca librtmp.so.1, basta digitar no seu terminal **locate "librtmp.so.1"** e será mostrado o local onde ela está instalada.

Resultado do comando locate: /usr/lib/x86_64-linux-gnu/librtmp.so.1

Basta criar o link simbólico. Ir para /usr/lib e digitar:

sudo ln -s /usr/lib/x86_64-linux-gnu/librtmp.so.1 librtmp.so.0

Com isso, será criado o link simbólico da biblioteca (librtmp.so.0) que o NCL necessita.

Ao instalar essas bibliotecas, digite novamente ncl e caso apareça o erro abaixo:

bash: /usr/local/ncarg/bin/ng4ex: /bin/csh: interpretador inválido: Arquivo ou diretório não encontrado

Digite no seu terminal Linux: *sudo apt-get install csh*

Os passsos acima resolvem os erros.

Para testar se o NCL foi instalado corretamente digite no terminal *ncl* e se aparecer as linhas abaixo, o ncl foi instalado corretamente.

Copyright (C) 1995-2017 - All Rights Reserved
University Corporation for Atmospheric Research
NCAR Command Language Version 6.4.0
The use of this software is governed by a License Agreement.
See http://www.ncl.ucar.edu/ for more details.
ncl 0>

Para encerrar a sessão digite *quit* ou *exit*.

Outra forma de testar se o NCL foi instalado corretamente é digitar diretamente no terminal do Linux e não com a sessão do ncl ativa a linha de comando abaixo:

ng4ex gsun01n

Tem que aparecer uma sequência de imagens, isso é um script teste para saber se o NCL foi instalado corretamente. Apenas clique sobre as imagens que elas vão mudando. Se isso acontecer, o NCL está instalado.

Para obter mais opções de execução do NCL, digite no terminal o comando:

ncl -h

Serão mostradas as opções abaixo que alteram o comportamento do NCL.

```
Usage: ncl -fhnopxQV <args> <file.ncl>
         -f: use new file structure and NetCDF4 features when possible
         -h: print this message and exit
         -n: don't enumerate values in print()
         -o: retain former behavior for certain backwards-incompatible changes
         -p: don't page output from the system() command
         -x: echo NCL commands
         -Q: turn off echo of NCL version and copyright info
         -V: print NCL version and exit
```

4 Visão geral

É aconselhável que o usuário tenha sempre acesso à internet para tirar dúvidas no site oficial do NCL (http://www.ncl.ucar.edu) ou os tutoriais salvos no seu computador que estão disponíveis em: http://www.ncl.ucar.edu/Document/Manuals.

O objetivo desse tópico consiste em fornecer uma visão geral sobre o NCL. Dúvidas mais específicas podem ser obtidas em: http://www.ncl.ucar.edu.

O NCL é uma linguagem de programação interpretada com a função de acessar, analisar e visualizar dados ambientais. Uma excelente dica é acessar o site abaixo para ter uma ideia da quantidade de informações disponibilizadas.

http://www.ncl.ucar.edu/Document/Manuals/Ref_Manual

Ao entrar no site a primeira visão é a da imagem abaixo.

Figura 2: Site do NCL.

Na parte superior dessa página há oito links importantes que irão auxiliar no processamento e geração das figuras. São eles:

- **NCL**: que contém informações gerais sobre a ferramenta.
- **Example**: como o próprio nome sugere, são todos os exemplos disponíveis na página.
- **Functions**: são as funções para as mais variadas rotinas (anomalia, climatologia, EOF, wavelets, dentre outras).
- **Resources**: são os recursos, ou seja, a parte de personalização gráfica.
- **Popular links**: estão disponíveis as tabelas de cores, marcadores, fontes, projeções, dentre outras informações.

- **What's new**: notícias sobre a ferramenta.
- **Support**: estão disponíveis os tutoriais, fórum do NCL, workshops e etc.
- **External**: informações sobre ferramentas externas como o CDO, NetCDF e outros projetos.

Não deixe de acessar as correções e novas implementações da versão mais recente do NCL disponível no link abaixo.

http://www.ncl.ucar.edu/current_release.shtml

4.1 Palavras-chave reservadas

São palavras que não devem ser utilizadas para criação de nome de variáveis ou funções. Essas palavras são case sensitive, isto é, há diferença entre letras maiúsculas e minúsculas. Por exemplo, as palavras **dado** e **Dado** são interpretadas de forma distinta pelo NCL. As palavras-chave devem ser escritas da seguinte forma:

begin, break, byte, character, continue, create, defaultapp, do, double, else, end, enumeric, external, False, file, float, function, getvalues, graphic, if, integer, int64, list, load, local, logical, long, _Missing, Missing, new, noparent, numeric, procedure, quit, Quit, QUIT, record, return, setvalues, short, string, then, True, undef, while and all built-in function and procedure names.

Uma lista com todas as palavras reservadas pode ser visualizada em:

http://www.ncl.ucar.edu/Document/Manuals/Ref_Manual/NclKeywords.shtml

4.2 Scripts prontos e dados disponibilizados

É disponibilizado um diretório chamado NCL que contém quatro subdiretórios (auxiliares, dados, figuras e script) com todos os scripts mostrados nesse documento assim como os dados necessários para executá-los. Não é necessário realizar qualquer alteração neles, bastando simplesmente executá-los.

Link para os dados e scripts: https://goo.gl/Qr2wAU

Não esqueça de adicionar no topo de cada script as bibliotecas do NCL como mostrado na página 252.

O nome das bibliotecas estão no diretório NCL/auxiliares e possui o nome de todas_bibliotecas.txt. Basta copiar os nomes e colar no topo do script que se deseja executar.

4.3 Executando o NCL

O NCL pode ser executado de duas formas: modo interativo ou modo batch (por meio da criação de um script).

4.3.1 Modo interativo

No modo interativo, o usário digita diretamente os comandos no terminal do Linux, por exemplo, ao digitar ncl no terminal aparecerá a seguinte mensagem:

```
 Copyright (C) 1995-2017 - All Rights Reserved
 University Corporation for Atmospheric Research
 NCAR Command Language Version 6.4.0
 The use of this software is governed by a License Agreement.
 See http://www.ncl.ucar.edu/ for more details.
ncl 0>
```

Digitando comandos (modo interativo) no ambiente NCL:

```
ncl 0> x=ispan(1,5,1)
ncl 1> print(x)


Variable: x
Type: integer
Total Size: 20 bytes
            5 values
Number of Dimensions: 1
Dimensions and sizes:   [5]
Coordinates:
(0)     1
(1)     2
(2)     3
(3)     4
(4)     5
ncl 2>
```

Para sair do ncl basta digitar quit ou exit.

4.3.2 Modo batch

No modo batch é necessário criar um script porque não é interessante ficar digitando vários comando no terminial. Por isso, crie um arquivo texto com o seu editor de sua preferência chamado teste.ncl e execute-o digitando *ncl teste.ncl* no seu terminal Linux.

```
1   begin
2
3   i = ispan(1,5,1)
4
5   print(i)
6
7   end
```

O resultado do script teste.ncl será:

```
 Copyright (C) 1995-2017 - All Rights Reserved
 University Corporation for Atmospheric Research
 NCAR Command Language Version 6.4.0
 The use of this software is governed by a License Agreemen
 See http://www.ncl.ucar.edu/ for more details.


Variable: i
Type: integer
Total Size: 20 bytes
            5 values
Number of Dimensions: 1
Dimensions and sizes:   [5]
Coordinates:
(0)     1
(1)     2
(2)     3
(3)     4
(4)     5
```

4.4 Extensões disponíveis para saída gráfica

Há sete tipos de saídas gráficas disponíveis, são elas:

- **ps**
- **eps**
- **epsi**

- **pdf**
- **X11**
- **png**
- **svg**

Importante: Caso sejam geradas muitas figuras para impressão, recomenda-se o uso do formado pdf que ocupa menos espaço em disco do que os formatos ps, eps e epsi. Todos esses formatos citados são ideais para impressão. No caso de figuras a serem inseridas em uma página web, o ideal é utilizar o formato png.

4.5 Tipos de dados e precisão numérica

Os tipos de dados são:

Numéricos: double (64 bits), float (32 bits), long (32 ou 64 bits), integer (32 bits), short (16 bits), byte (8 bit), números complexos não são suportados.

Não-numéricos: string, character, graphic, file, logical, list.

Mais informações sobre a precisão numérica podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Manuals/Ref_Manual/NclDataTypes.shtml

Um resumo sobre a precisão numérica é visualizado na Tabela 1.

Tabela 1: Tipos de dados numéricos.

Tipo	**categoria numérica**	**Tipo de sistema**	**Tipo de tamanho**	**Mínimo valor**	**Máximo valor**	**_FillValue padrão**	**Sufixo literal**
double	numeric	todos	64 bits	+/- 2.22507e-308	+/- 8.98846e+307	9.969209968386869e+36	d ou D
int64	enumeric	todos	64 bits	-9223372036854775808	9223372036854775807	-9223372036854775806	q
uint64	enumeric	todos	64 bits	0	18446744073709551615	18446744073709551614	Q
long	numeric	64 bits	64 bits	-9223372036854775808	9223372036854775807	-2147483647	l
ulong	enumeric	64 bits	64 bits	0	18446744073709551615	4294967295	L
float	numeric	todos	32 bits	+/- 1.175494e-38	+/- 1.701411e+38	9.96921e+36	nada
long	numeric	32 bits	32 bits	-2147483648	2147483647	-2147483647	l
ulong	enumeric	32 bits	32 bits	0	4294967295	4294967295	L
integer	numeric	todos	32 bits	-2147483648	2147483647	-2147483647	i (opcional)
uint	enumeric	todos	32 bits	0	4294967295	4294967295	I
short	numeric	todos	16 bits	-32768	32767	-32767	h
ushort	enumeric	todos	16 bits	0	65535	65535	H
byte	numeric	todos	8 bits	-128	127	-127	b
ubyte	enumeric	todos	8 bits	0	255	255	B

4.6 Expressões numéricas

Operadores algébricos: http://www.ncl.ucar.edu/Document/Language/algebraic.shtml

+	soma (também pode ser usado para concatenação)
-	subtração
*	multiplicação
∧	exponenciação
%	módulo
#	multiplicação de matrizes
>,<	maior que, menor que

Operadores lógicos: http://www.ncl.ucar.edu/Document/Language/logical.shtml

.lt.	menor que
.le.	menor ou igual a
.gt.	maior ou igual a
.ne.	diferente de
.eq.	igual a
.and.	e
.or.	ou
.xor.	ou exclusivo
.not.	não

4.7 Variáveis

O nome das variáveis deve começar com caracter alfanumérico e pode conter mistura de números e letras. O símbolo underscore "_" é permido na criação de variáveis.

O link abaixo mostra o guia de referência para variáveis:

http://www.ncl.ucar.edu/Document/Manuals/Ref_Manual/NclVariables.shtml

4.8 Loops

Há dois tipos de loops: o **do while** e o **do**.

Loop Do:

do n = início,fim,incremento_opcional
    [sentença(s)]
end do *; observe que há espaço*

Loop do while:

do while (expressão_escalar_lógica)
    [sentença(s)]
end do

4.9 Condicionais

Como em outras linguaguens de programação utilizam-se duas estruturas de **if**, são elas: **if-then** e **if-then-else**.

if (expressão_escalar_lógica) **then**
    [sentença(s)]
end if

if (expressão_escalar_lógica) **then**
    [sentença(s)]
else
    [sentença(s)]
end if

Outra possibilidade:

```
if (expressão_escalar_lógica) then
        [A_sentença(s)]
    else if (expressão_escalar_lógica) then
        [B_sentença(s)]
    else if (expressão_escalar_lógica) then
        [C_sentença(s)]
    else
        [D_sentença(s)]          ; C & D
    end if                       ; B
    end if                       ; A
end if
```

4.10 Parando a execução do loop

Há duas formas de parar um loop, por meio do break ou continue.

Exemplo1: Uso do break.

```
x = 1

do while(x.lt.5)
    print("x: " + x)
    x=x+1
    break
end do
```

O resultado será x: 1.

Exemplo2: Uso do continue.

```
y = 1
z = 2

do while(y.lt.5)
    print("y: " + y)
    y=y+1
    continue ; Aqui o loop será interrompido
    print("zA: " + z)
    z=z+1
    print("zD: " + z)
end do
    print("Executando a próxima instrução")
```

4.11 Dimensão e indexação

Há duas formas de indexar (no sentido de fixar dimensões) arranjos[1] em NCL, são elas: a forma *padrão* e a *coordenada*.

O índice em NCL inicia em *0* e terminam em *N-1*.

As dimensões mais a direita variam mais rapidamente, enquanto que as dimensões mais a esquerda, variam mais lentamente.

Os arranjos possuem a seguinte forma:

início:fim:incremento

Exemplo1: Supondo que a variável temperatura (T) seja tridimensional, isto é, tempo, latitude e longitude. A forma padrão de selecionar determinada dimensão é feita de acordo com o exemplo abaixo:

```
T ou T(:,:,:)      ; seleciona toda a matriz de dados
T(0,:,::5)         ; primeiro tempo (0), todas as latitudes (:) e a cada 5 graus de longitude (::5)
T(0,::-1,4:40)     ; primeiro tempo (0), inverte a latitude (::-1) e seleciona os índices 4 a 40 de longitude (4:40)
T(:1,45,10:20)     ; seleciona os dois primeiros tempos, latitude fixa no índice 45 e longitude variando de 10-20
```

Exemplo2: Outra forma de selecionar o dado é por meio da indexação por coordenadas (ideal para nível vertical, latitude e longitude). Nesse caso, deve-se utilizar o símbolo {...}.

X = T(20,{200},{-20:20},{90:290:2})

Nesse caso, foi selecionado o tempo 21 do arquivo, nível vertical de 200hPa, latitude entre 20°S e 20°N e longitude de 90 a 290 (formato 0-360°) variando de 2 em 2 graus.

Importante: No NCL a latitude varia de sul para norte e a longitude de oeste para leste.

[1] A palavra arrays ou arranjos é um conjuntos de elementos que podem ser de uma dimensão (vetores), de duas dimensões (matrizes) ou multidimensionais

Exemplo3: Supondo o vetor x do tipo integer com 10 posições.

x = (/1,2,3,4,5,6,7,8,9,10/) ou x = ispan(1,10,1)

A primeira linha representa os dez valores do vetor x, iniciando em 1 e terminando em 10. A segunda linha é o índice correspondente a cada posição do vetor x.

1	2	3	4	5	6	7	8	9	10
0	1	2	3	4	5	6	7	8	9

y1 = x ; Resultado: 1 2 3 4 5 6 7 8 9 10

y2 = x(6) ; Resultado: 7 porque no NCL o índice começa em zero e não em 1

y3 = x(3:8) ; Resultado: 4 5 6 7 8 9

y4 = x(0:9:2) ; Resultado: 1 3 5 7 9. Primeiro índice até o último índice a cada intervalo de 2 valores

y5 = x(:4) ; Resultado: 1 2 3 4 5. Isso é o mesmo que x(0:4)

y6 = x(3:8:-1) ; Resultado: 9 8 7 6 5 4

y7 = x(8:3) ; Resultado: 9 8 7 6 5 4. Nota-se a ausência do -1

y8 = x(::-1) ; Resultado: 10 9 8 7 6 5 4 3 2 1

y9 = x(9:0) ; Resultado: 10 9 8 7 6 5 4 3 2 1. Similar ao exemplo anterior

4.12 Redução de dimensão

Ao fixar a dimensão de uma variável ocorre sua redução. Supondo que T seja do tipo T(nt,nz,ny,nx). Onde nt é o número de tempos, nz é o número de níveis verticais, ny é o número pontos de latitude e nx é o número de pontos de longitude:

```
T1 = T(5,:,12,:)   ; apenas nz e nx variam enquanto nt (5) e ny (12) estão fixos.
T2 = T(:,:,:,0)    ; apenas nt, nz e ny variam enquanto nx (0) está fixo.
```

No primeiro caso (T1) houve redução de 4 para 2 dimensões porque nt e ny estão fixos. No segundo caso (T2) houve redução de 4 para 3 dimensões porque nx está fixo.

Uma particularidade sobre redução de dimensão. Caso seja feita a redução de dimensão ocorre a perda de informações da dimensão que foi reduzida, mas pode-se forçar que isso não ocorra da seguinte forma:

Exemplo1: Abrindo o arquivo precip.mon.mean.nc que contém a variável precip.

```
f=addfile("precip.mon.mean.nc","r")
ppt1=f->precip(0,:,:) ; Fixando o primeiro tempo do arquivo.
printVarSummary(ppt1)
```

Resultado de um pequeno trecho do printVarSummary(ppt1):

Dimensions and sizes: [lat |72] x [lon |144]

Nota-se que houve a redução da dimensão, isto é, como foi definido o tempo 0, essa dimensão tempo passa a não existir mais na variável ppt1, restando apenas as dimensões lat e lon.

Para evitar a perda da dimensão tempo, no lugar de:

ppt1=f->precip(**0**,:,:)

Faça:

ppt2=f->precip(**0:0**,:,:)

printVarSummary(ppt2)

Resultado de um pequeno trecho do printVarSummary(ppt2):

Dimensions and sizes: **[time |1]** x [lat |72] x [lon |144]

Com isso, a dimensão time continua na variável ppt2 sem ocorrer sua perda.

4.13 Símbolos especiais do NCL

Os símbolos abaixo possuem as mais variadas aplicações, por exemplo, desde abertura de arquivos, criação de lista, criação ou alteração de atributos, dimensões ou coordenadas, dentre outras possibilidades. A seguir, cada um desses símbolos será comentado.

;	início de um comentário;
@	cria/referencia atributos;
!	cria/referencia dimensões;
&	cria/referencia uma variável de coordenada;
{...}	usado para subindexação de coordenada;
$...$	encerra strings ao importar/exportar variáveis via addfile;
[...]	subindexa variáveis do tipo lista;
(/.../)	constrói um vetor ou matriz;
:	usado na sintaxe de vetores ou matrizes;
|	usado como separador para dimensões (reordenamento de dimensões);
\	indica que a declaração continua na próxima linha;
->	usado para a entrada/saída de dados.
=	usado para atribuição;
:=	usado para reatribuição;
[:]	todos os elementos de uma lista.

4.13.1 Símbolo ;

O ponto e vírgula (;) é utilizado para comentar trechos do script, isto é, tudo que estiver à direita deste símbolo será considerado como comentário.

Exemplo:

f = addfile ("uv300.nc","r") ; abertura do arquivo uv300.nc

O trecho em vermelho é um comentário porque está à direita do símbolo ";".

A partir da versão 6.4.0 é possível utilizar a estrutura abaixo para comentar o script:

/;

Comentário a ser adicionado no script.

;/

O que estiver entre os símbolos /; e ;/ é considerado como comentário e não será executado pelo programa.

4.13.2 Símbolo @

Para criar ou referenciar atributos utiliza-se o símbolo "@". Os valores assumidos podem ser numéricos ou textos.

Pequeno trecho de um script.

```
f = addfile("uv300.nc","r")
u = f->U
printVarSummary(u)

; Alterando o nome dos atributos usando o "@"

u@units = "m/s"
u@long_name = "vento zonal"
printVarSummary(u)
```

Os atributos units e long_name da variável "u" foram alterados com o símbolo "@".

4.13.3 Símbolo !

As variáveis contidas nos arquivos possuem um nome para cada dimensão. O usuário pode renomear ou criar um novo nome para a dimensão. O símbolo "!" é responsável por realizar essa tarefa. A ordem das dimensão será sempre da esquerda para a direita, com a dimensão mais a esquerda iniciando em zero. Os valores assumidos são strings.

Assumindo a seguinte variável x que possui quatro dimensões na ordem abaixo:

x(time,lev,lat,lon)

Para renomear as dimensões, utiliza-se o exemplo abaixo:

```
x!0 = "time"
x!1 = "level"
x!2 = "latitude"
x!3 = "longitude"
```

Exemplo: Pequeno trecho de um script. Foram criadas três dimensões (time, lat e lon).

```
prec_interp!0 = "time"
```

prec_interp!1 = “lat”
prec_interp!2 = ”lon”

Em que prec_interp é a variável que o usuário está manipulando.

4.13.4 Simbolo &

Para criar uma variável de coordenada (latitude e longitude, por exemplo), utiliza-se o símbolo “&”. Para criar essa variável, primeiro é necessário nomear as dimensões com o símbolo “!”. Os valores assumidos são numéricos.

Exemplo: Pequeno trecho de um script.

prec!0 = “lat”
prec!1 = “lon”
prec&lat = fspan(-49.875,49.875,72) ; são os valores da coordenada de latitude.
prec&lon = fspan(-178.75,178.75,144) ; são os valores da coordenada de longitude.
prec&lat@units = “degrees_north” ; é a unidade da dimensão latitude.
prec&lon@units = “degrees_east” ; é a unidade da dimensão longitude.

Essa tarefa criou os valores da coordenada de latitude e longitude. Isso é importante na hora de referenciar o dado no mapa. As duas últimas linhas criam as unidades para latitude e longitude (tem que ter essas duas linhas), respectivamente.

A função fspan cria uma matriz de números de ponto flutuante uniformemente espaçados. Mais informações podem ser obtidas no link abaixo:

http://www.ncl.ucar.edu/Document/Functions/Built-in/fspan.shtml

4.13.5 Simbolo {...}

O símbolo “{...}” é utilizado para subindexação de coordenada. Este símbolo é ideal para definir nível vertical, latitude e longitude.

Exemplo: A variável “u” possui as seguintes dimensões u(nt,nz,ny,nx).

a = addfile(“temp1000hpa.mensal.nc”,“r”)

u = a->air(:,{1000},{-10},{300})

Foram selecionados todos os tempos (:), nível vertical de 1000hPa, latitude 10°S e longitude 300, ou seja, uma série temporal.

4.13.6 Simbolo $...$

O símbolo “$...$” encerra strings ao importar/exportar variáveis via addfile.

f = addfile(“uv.nc”,“r”)

varn = (/“uwnd”,“vwnd”/)

```
do i = 0,1
   uv = f->$varn(i)$
end do
```

As variáveis são passadas como string com o símbolo "$...$".

Esse símbolo pode ser usado quando o nome da variável contém caracteres não-alfanuméricos, como por exemplo, espaços no nome da variável, sinais de ±.

Exemplo: Abertura de um arquivo que contém o nome da variável "ice cream+oreo-cookies...yummy!".

```
x = f->$"ice cream+oreo-cookies...yummy!"$
```

4.13.7 Símbolo [/.../]

O símbolo "[...]" é utilizado para subindexar variáveis do tipo lista que pode conter um conjunto heterogêneo de variáveis. Especificamente, as variáveis dentro de uma lista podem ser de diferentes tipos, tamanhos e formas.

Exemplo:

```
i = (/ (/1,2,3/), (/4,5,6/), (/7,8,9/) /)   ; Um arranjo 2D do tipo inteiro.
x = 5.0                                     ; Um escalar do tipo float.
d = (/100000.d, 283457.23d/)                ; Um arranjo double 1D.
s = "abcde"                                 ; String.
c = stringtochar("abcde")                   ; Caracter.
vl = [/i, x, d, c, s/]                      ; Criação da lista via [/.../].
```

4.13.8 Símbolo (/.../)

O símbolo "(/.../)" cria um vetor ou matriz de dados.

Exemplo:

```
res@xyMarkers = (/16,16,16/)
res@xyLineColors = (/"red","black","blue"/)
res@xyLineThicknesses = (/2,4,6/)
```

A primeira linha define o tipo de marcador para cada curva, neste caso, todos possuem o mesmo tipo de marcador. Na segunda, são definidas as cores para cada curva e na terceira linha, define-se a espessura da linha de cada curva.

4.13.9 Símbolo :

O símbolo ":" é utilizado na sintaxe de vetores ou matrizes.

Exemplo:

```
plot = gsn_csm_xy(wks,x,y(0:11,{0},{300}),res)
```

Foram selecionados os primeiro 12 tempos (0:11), latitude 0 e longitude 300.

4.13.10 Símbolo |

O símbolo "|" é utilizado para reordenar dimensões.

Tem-se a variável x com as seguintes dimensões *x(nt,ny,nx)*, em que nt é o tempo, ny é a latitude e nx e a longitude. Supondo que deseja-se realizar um cálculo, só que a dimensão tempo (nt) precisa ser a última dimensão. A nova variável precisará ter o seguinte formato: a(ny,nx,nt). Como proceder?

Basta seguir o exemplo abaixo:

a = x(ny:|,nx:|,nt:|)

Agora, a variável a apresenta a dimensão tempo (nt) como a última dimensão.

4.13.11 Símbolo \

O símbolo "\" indica que a declaração continua na próxima linha.

Exemplo: Quebrando a linha para continuar na linha seguinte.

```
res@tmXBLabels = (/"J","F","M","A","M","J","J", \
                   "A","S","O","N","D"/)
```

4.13.12 Símbolo ->

O símbolo "->" é usado para a entrada/saída de dados de arquivos em um dos formatos suportados.

Exemplo: Abertura de um arquivo.

f = addfile("temp.nc","r") ; abertura do arquivo
t = f->air ; importação da variável "air" do arquivo "temp.nc" (f).

4.13.13 Símbolo =

O símbolo "=" é usado para atribuição.

Exemplo:

tc = tk-273.15

4.13.14 Símbolo :=

O símbolo ":=" é usado para reatribuição.

Exemplo: Supondo o vetor abaixo com os seguintes valores:

k = (/1,3,4,9/) ; um vetor 1D do tipo inteiro.

k = (/17.5,21.4/) ; vetor 1D do tipo real com tamanho e tipo diferente do vetor acima.

Antes da versão 6.1.1 era necessário apagar o primeiro vetor "k" para poder atribuir os novos valores, isto é,

delete(k)

E depois é feita a atribuição com os novos valores, como no exemplo abaixo:

k = (/17.5,21.4/)

Na versão 6.1.2 não será mais necessário realizar essa tarefa, basta utilizar o símbolo ":=".

k := (/17.5,21.4/)

O NCL não permite usar este símbolo para a mesma variável. Por exemplo:

x := x(::4,:,:)

4.13.15 Símbolo [:]

O símbolo "[:]" é usado para acessar todos os elementos da lista.

Exemplo: Há 12 arquivos mensais sendo que cada um possui apenas um tempo de precipitação (mes01.nc, mes02.nc, ..., mes12.nc). O objetivo é ler todos os arquivos como se fosse um arquivo com todos os 12 tempos.

O script abaixo realiza essa tarefa.

```
1   ; Nome do script: cap4_ex01.ncl
2
3   begin
4
5   ; Lista todos os arquivos (mes.01.nc, mes.02.nc, ..., mes.12.nc).
6   all_files = systemfunc("ls ../../dados/mes.*.nc")
7
8   ; Note para o "s" na função addfiles, isto é, lê vários arquivos.
9   fall = addfiles(all_files, "r")
10
11  ; Concatena os arquivos que é o padrão.
12  ListSetType(fall, "cat")
13
14  ; Uso da sintaxe [:]. O precip é o nome da variável do arquivo.
15  ppt = fall[:]->precip
16
17  printVarSummary(ppt)
18
19  end
```

Ao unir todos os 12 arquivos, a dimensão tempo (time) passa a ter o valor 12, isto é, para cada tempo está um arquivo associado. Agora, basta realizar as devidas operações para manipular a variável.

```
Variable: ppt
Type: float
Total Size: 48 bytes
            12 values
Number of Dimensions: 3
Dimensions and sizes:   [time | 12] x [lat | 1] x [lon | 1]
Coordinates:
            time: [78162..78496]
            lat: [   0..   0]
            lon: [   0..   0]
Number Of Attributes: 12
  long_name :   Average Monthly Rate of Precipitation
  units :       mm/day
  _FillValue :  -9.96921e+36
  missing_value :       -9.96921e+36
  precision :   32767
  least_significant_digit :     2
  var_desc :    Precipitation
  dataset :     GPCP Version 2.2 Combined Precipitation Dataset
  level_desc :  Surface
  statistic :   Mean
  parent_stat : Mean
  actual_range :        (  0, 9.985388e+29 )
```

A variável passa a ter 12 tempos.

Importante: Para usar a opção "join" em vez de "cat", é necessário que todos os arquivos tenham as mesmas dimensões. O join diferente do cat serve para adicionar uma nova dimensão a variável.

Mais informações podem ser obtidas no link abaixo:

http://www.ncl.ucar.edu/Document/Functions/Built-in/ListSetType.shtml

4.14 Sobre _FillValue

O _FillValue significa valor ausente, isto é, ao ler uma determinada variável e ao tentar gerar o gráfico, os valores _FillValue não serão mostrados. É uma convenção dos arquivos NetCDF que o NCL também adota. Para definir um valor indefinido, utiliza-se o símbolo "@".

Importante: Caso o dado somente possua o atributo missing_value (que pode ser visto com o ncl_filedump ou ncdump) e não possua _FillValue, o NCL vai gerar a figura considerando esses valores porque ele não entende o missing_value. Por isso, é importante criar o _FillValue como será mostrado nos exemplos a seguir.

Para mais informações, acessar:

https://www.ncl.ucar.edu/Document/Language/fillval.shtml

Exemplo1:

x@_FillValue = -999

Exemplo2:

x@_FillValue = default_fillvalue("double") ; Define o _FillValue como double.

Exemplo3:

x = (/1.0, 2.0, 3.0, 4.0, 5.0/) ; Definindo uma variável x com 5 valores do tipo float.

print(avg(x)) ; Imprime o valor médio de x que é 3

x@_FillValue = 5. ; Definindo o valor ausente como 5.

print(avg(x)) ; Ao calcular a média, o valor 5 não entrará no cálculo, o que resulta em 2.5, que corresponde a média entre 1.0, 2.0, 3.0 e 4.0.

A Tabela 2 mostra os valores para o atributo _FillValue.

Tabela 2: Valores ausentes para cada tipo de dado

Tipo	**Valor do _FillValue**	**_FillValue antes da versão 6.0.0**
double	9.969209968386869e+36	-9999.0
int64	-9223372036854775806	-999999
uint64	18446744073709551614	0
float	9.96921e+36	-999.0
long	-2147483647	-9999
ulong	4294967295	0
integer	-2147483647	-999
uint	4294967295	0
short	-32767	-99
ushort	65535	0
byte	-127	0xff
ubyte	255	0
logical	Missing	Missing
string	"missing"	"missing"
character	0x00	0x00
graphic	-1	-1
file	-1	-1
list	-1	-1

4.15 Arquivos suportados

O NCL é capaz de ler e manipular dados nos seguintes formatos: NetCDF, HDF4, HDF4-EOS, HDF5, GRIB1, GRIB2, shapefile, CCM History Tape e ascii (texto). Informações adicionais sobre os formatos suportados podem ser encontradas no link abaixo.

http://www.ncl.ucar.edu/Document/Language/supported.shtml

Para leitura de dados, o NCL é capaz de importar os seguintes formatos:

- NetCDF3 e NetCDF4
- HDF4-SDS, HDF4-EOS HDF5, HDF5-EOS
- GRIB1 e GRIB2
- CCM History Tape
- shapefiles
- Binário
- ASCII

No caso de exportar (escrever) os seguintes dados são suportados:

- NetCDF3 e NetCDF4
- HDF4 e HDF5
- GRIB1 e GRIB2
- Binário
- ASCII

A abertura de um arquivo é feita usando a função addfile. Veja o exemplo abaixo.

f = addfile("nome_arquivo.ext",status)

Onde:

f é um nome dado pelo usuário para identificar o arquivo aberto.

nome_arquivo.ext é o nome do arquivo com sua extensão (.nc, .cdf, .hdf, .hdfeos, .grb ou .grib1 e .ccm).

O **status** pode ser:

- **"r"** apenas para leitura de arquivos (há suporte para todos os arquivos citados acima),
- **"c"** criação de arquivos e

- **"w"** para leitura e edição de arquivos.

Exemplo de abertura de um arquivo para somente leitura ("r"):

a = addfile("temperatura.nc", "r")

A letra *a* receberá todo o conteúdo do arquivo aberto *temperatura.nc* que será lido apenas como leitura (*"r"*).

E supondo que a variável desse arquivo tenha o nome de *temp*. Para importar essa variável utiliza-se o símbolo ->.

Exemplo:

t=a->temp

A partir de agora, qualquer cálculo deve ser feito com a variável *t* que contém todas as informações da variável *temp*. Essa será a variável de referência.

4.16 Imprimindo o conteúdo de um arquivo

Após a abertura de um arquivo, é necessário saber qual o seu conteúdo. Para tal, utiliza-se o comando print. Esse comando é similar ao "*ncdump -h*" da biblioteca NetCDF. Dessa forma, o usuário será capaz de saber o nome da variável do arquivo, dimensões (tempo, nível vertical, latitude e longitude). Há também informações sobre o tipo de dado da variável, isto é, float, integer, character ou double. É possível extrair informações importantes com esse comando.

Exemplo de uso do comando print:

Digite ncl no seu terminal Linux, em seguida abra o seu arquivo com o comando addfile.

a = addfile("precip.mon.mean.nc", "r")

Simplesmente, digite o comando abaixo:

print(a)

Serão mostradas várias informações sobre o arquivo aberto.

Não deixe de usar também o printVarSummary para mostrar informações sobre a variável do arquivo. A diferença é que este comando fornece informações somente sobre a variável.

Sobre o printVarSummary:

http://www.ncl.ucar.edu/Document/Functions/Built-in/printVarSummary.shtml

Outra forma de explorar o arquivo pode ser feita usando o *ncl_filedump*, um comando similar ao ncdump da biblioteca NetCDF.

Para utilizá-lo, basta digitar diretamente no terminal do Linux o comando abaixo:

ncl_filedump arquivo.nc

Exemplo:

Caso você tenha um arquivo NetCDF e deseja visualizar o contéudo do mesmo, proceda da seguinte forma digitando no seu terminal Linux sem o NCL aberto:

ncl_filedump precip.mon.mean.nc

4.17 Sobre os recursos utilizados no NCL

Os recursos utilizados no NCL servem para modificar as características da figura, como por exemplo, tamanho, cores das letras, personalização dos eixos x e y, formatação dos números, dentre outras possibilidades.

As duas primeiras letras do recurso definem o tipo de recurso utilizado. A exceção para esse caso são os recursos que aprensentam o nome gsn que é um recurso genérico aplicado para os mais diferentes tipos de gráficos.

A utilização de alguns recursos está condicionada a habilitação de alguns pré-requisitos ou a definição de alguns valores, por esse motivo, tenha cuidado ao utilizar esses recursos. Uma dica interessante é acessar o link abaixo e estudar cada um dos recursos disponíveis. A Tabela 3 mostra um resumo dos recursos.

http://www.ncl.ucar.edu/Document/Graphics/Resources/index.shtml

O recursos gsn estão no link abaixo:

http://www.ncl.ucar.edu/Document/Graphics/Resources/gsn.shtml

Tabela 3: Tipos de recursos disponíveis

am	annotation manager	**pm**	plot manager	**vf**	vector field
cn	contour	**pr**	primitive	**vc**	vectors
ca	coordinate arrays	**sf**	scalar fields	**vp**	viewport
gs	graphical style	**ti**	title	**wk**	workstation
lb	labelbar	**tm**	tickmarks	**ws**	workspace
lg	legend	**tx**	text	**xy**	xy plot
mp	map	**tr**	transformation		

A ideia deste tópico é mostrar alguns recursos e como utilizá-los porque no momento de criar as figuras dependendo do que seja feito, o usuário saberá onde buscar o recurso adequado para sua finalidade.

Abaixo é mostrado um pequeno trecho de um script como exemplo da utilização dos recursos.

```
res                       = True
res@gsnLeftString         = "Texto do lado esquerdo"
res@gsnRightString        = "Texto do lado direito"
res@gsnStringFontHeightF  = 0.03
res@cnFillOn              = True
res@tiMainString          = "titulo principal da figura"
res@tmXBLabelFontHeightF  = 0.02
res@tmYLLabelFontHeightF  = 0.02
```

Explicação das informações acima.

- Linha 1: quer dizer que o usuário utilizará os recursos para formatar o gráfico. Caso contrário, modificar para False. O nome *res* vem de resources, mas este nome pode ser alterado sem problemas desde que as demais linhas também sejam alteradas.

- Linha 2: gsnLeftString: É um recurso que adiciona um título no canto superior esquerdo do gráfico. Tudo que for referente a texto deve estar entre aspas duplas.

- Linha 3: gsnRightString: É um recurso que adiciona um título no canto superior direito do gráfico. Tudo que for referente a texto deve estar entre aspas duplas.

- Linha 4: gsnStringFontHeightF: Modifica o tamanho da fonte do título do gráfico. A letra F no final desse recurso significa Float.

- Linha 5: cnFillOn: É um recurso que habilita o preechimento do gráfico (shaded). Lembrando que as duas primeiras letras definem o tipo de recurso, ou seja, o *cn* quer dizer que está sendo formatado um gráfico do tipo contorno (contour).

- Linha 6: tiMainString: Escreve o título principal na parte superior central do gráfico. Tudo que for referente a texto deve estar entre aspas duplas.

- Linha 7: tmXBLabelFontHeightF: Responsável por formatar o lado inferior (B) (bottom) do eixo x (X) do gráfico (tickmark, *tm*). Lembrando que F no final do recurso quer dizer um valor do tipo Float. Se fosse o eixo superior, basta trocar B por T (top).

- Linha 8: tmYLLabelFontHeightF: Responsável por formatar o lado esquerdo (L) (left) do eixo y (Y) do gráfico (tickmark, *tm*). Lembrando que F no final do recurso quer dizer um valor do tipo Float. Se fosse o eixo direito, basta trocar L por R (right).

Isso foi apenas um pequeno exemplo da utilização de recursos em NCL. Cada usuário terá uma demanda distinta, logo cabe cada um pesquisar o que é melhor para suas necessidades. É impossível mostrar todos os recurso, mas recomenda-se a leitura do link abaixo:

http://www.ncl.ucar.edu/

Na parte superior da página há o item *Resources*. Leia e estude os scripts prontos que estão disponíveis no link abaixo. Estudando e fazendo os exercícios será possível entender melhor a utilização dos recursos.

http://www.ncl.ucar.edu/Document/Graphics/Resources/list_alpha_res.shtml

5 Visualizando as linhas de grade

Esse procedimento é útil para escrever textos e marcadores na figura utilizando as coordenadas da página.

Exemplo:

wks = gsn_open_wks("pdf","tmp")

drawNDCGrid(wks) ; desenha as linhas de grade na página

O resultado é mostrado abaixo. Com essas informações é possível obter as coordenadas da página.

6 Explorando o site do NCL e obtendo ajuda

No momento da criação de um script é bem provável que erros surgirão no momento de executá-lo.

Alguns passos são importantes na tentativa de encontrar erros. São eles: reveja seu arquivo no que diz respeito a abertura do arquivos, conversão de formatos, se o caminho está certo, falta de vírgula, pontos, parênteses. Após checar esses passos, tente entender o erro que é mostrado na tela, pois a mensagem diz qual é o problema e onde ele está.

Caso não tenha conseguido resolver o problema, envie um e-mail para a lista de discussão do NCL em inglês para solicitar ajuda.

Faça parte da lista de discussão acessando a página abaixo:

http://www.ncl.ucar.edu/Support/email_lists.shtml

Após sua inscrição, basta enviar suas dúvidas para o e-mail **ncl-talk@ucar.edu**.

Algumas dicas ao enviar e-mail: diga a versão do NCL utilizada, seja o mais breve possível ao relatar o seu problema e tudo deve ser escrito em inglês.

Os links recomendados para quem está começando a utilizar o NCL são:

http://www.ncl.ucar.edu/Training

http://www.ncl.ucar.edu/Training/Workshops

http://www.ncl.ucar.edu/Training/Workshops/Scripts

Nesses links há scripts prontos e explicação detalhada de cada um deles com os resultados e a possibilidade de realizar o download dos mesmos.

Outros links interessantes são mostrados abaixo:

http://www.ncl.ucar.edu/Document/Functions/list_alpha.shtml

http://www.ncl.ucar.edu/Document/Functions

http://www.ncl.ucar.edu/Document/Functions/list_type.shtml

Nesses links estão disponíveis todas as funções para manipulação e cálculo diversos (climatologia, EOF, anomalia, correlação, dentre outros).

E para finalizar, no link abaixo estão todos os exemplos para manipulação de dados e a geração dos mais variados tipos de gráficos.

http://www.ncl.ucar.edu/Applications

Com essas dicas, o usuário será capaz de processar dados e gerar suas figuras. É necessário explorar o site para buscar o que se deseja porque cada usuário terá uma necessidade distinta.

7 Inserindo acentos em português

Para inserir acentos nos scripts para figuras em português, o Mateus da Silva Teixeira e coloboradores desenvolveram um esquema de acentuação que encontra-se no Âpendice A (página 83) disponível no link abaixo:

http://www.ncl.ucar.edu/Document/Manuals/ncl_ptBR.pdf.

Copie o arquivo acentos.ncl que esta no diretório NCL/auxiliares para:

/usr/local/ncarg/lib/ncarg/nclscripts/contrib/

E no topo do seu script basta carregar esse arquivo da seguinte forma:

load "$NCARG_ROOT/lib/ncarg/nclscripts/contrib/acentos.ncl"

Com isso, basta apenas saber quais as letras deseja-se acentuar e inserir diretamente no script.

No fim deste capítulo estão os códigos necessários para acentuar as letras em português.

Exemplo1:

Acentuando as letras *ç* e *ã* da palavra *precipitação*.

res@gsnCenterString = "Precipita"+cedil+atilde+"o"

O que está entre aspas duplas representam as strings. Lembrando que o símbolo "+" é usado para concatenar (juntar) palavras.

Exemplo2:

Acentuando a expressão "Média da Precipitação".

res@tiYAxisString = "M"+eacute+"dia da Precipita"+cedil+atilde+"o"

Nos próximos capítulos serão mostrados mais exemplos de como acentuar letras.

```
Agrave = "A~H-15V6F35~A~FV-6H3~" ; À
agrave = "a~H-13V2F35~A~FV-2H3~" ; à
Aacute = "A~H-15V6F35~B~FV-6H3~" ; Á
aacute = "a~H-13V2F35~B~FV-2H3~" ; á
Acirc  = "A~H-15V6F35~C~FV-6H3~" ; Â
acirc  = "a~H-13V2F35~C~FV-2H3~" ; â
Atilde = "A~H-15V6F35~D~FV-6H3~" ; Ã
atilde = "a~H-13V2F35~D~FV-2H3~" ; ã
Auml   = "A~H-15V6F35~H~FV-6H3~" ; Ä
auml   = "a~H-13V2F35~H~FV-2H3~" ; ä

Egrave = "E~H-15V6F35~A~FV-6H3~" ; È
egrave = "e~H-13V2F35~A~FV-2H3~" ; è
Eacute = "E~H-15V6F35~B~FV-6H3~" ; É
eacute = "e~H-13V2F35~B~FV-2H3~" ; é
Ecirc  = "E~H-15V6F35~C~FV-6H3~" ; Ê
ecirc  = "e~H-13V2F35~C~FV-2H3~" ; ê
Euml   = "E~H-15V6F35~H~FV-6H3~" ; Ë
euml   = "e~H-13V2F35~H~FV-2H3~" ; ë

Igrave = "I~H-10V6F35~A~FV-6H3~" ; Ì
igrave = "i~H-10V2F35~A~FV-2H3~" ; ì
Iacute = "I~H-08V6F35~B~FV-6H3~" ; Í
iacute = "i~H-08V2F35~B~FV-2~"   ; í
Icirc  = "I~H-09V6F35~C~FV-6H3~" ; Î
icirc  = "i~H-09V2F35~C~FV-2H3~" ; î
Iuml   = "I~H-09V6F35~H~FV-6H3~" ; Ï
iuml   = "i~H-09V2F35~H~FV-2H3~" ; ï

Ograve = "O~H-15V6F35~A~FV-6H3~" ; Ò
ograve = "o~H-13V2F35~A~FV-2H3~" ; ò
Oacute = "O~H-15V6F35~B~FV-6H3~" ; Ó
oacute = "o~H-13V2F35~B~FV-2H3~" ; ó
Ocirc  = "O~H-16V6F35~C~FV-6H3~" ; Ô
ocirc  = "o~H-14V2F35~C~FV-2H3~" ; ô
Otilde = "O~H-15V6F35~D~FV-6H3~" ; Õ
otilde = "o~H-13V2F35~D~FV-2H3~" ; õ
Ouml   = "O~H-16V6F35~H~FV-6H3~" ; Ä
ouml   = "o~H-14V2F35~H~FV-2H3~" ; ä

Ugrave = "U~H-15V6F35~A~FV-6H3~" ; Ù
ugrave = "u~H-13V2F35~A~FV-2H3~" ; ù
Uacute = "U~H-13V6F35~B~FV-6H3~" ; Ú
uacute = "u~H-13V2F35~B~FV-2H3~" ; ú
Ucirc  = "U~H-15V6F35~C~FV-6H3~" ; Û
ucirc  = "u~H-13V2F35~C~FV-2H3~" ; û
Uuml   = "U~H-15V6F35~H~FV-6H3~" ; Ü
uuml   = "u~H-13V2F35~H~FV-2H3~" ; ü

Cedil = "C~H-15F35~K~FH2~" ; Ç
cedil = "c~H-13F35~K~FH2~" ; ç

Ntilde = "N~H-15V6F35~D~FV-6H3~" ; Ñ
ntilde = "n~H-13V2F35~D~FV-2H3~" ; ñ
```

8 Superescrito e subescrito

No NCL há várias formas de manipular strings. Uma das possibilidades está no link abaixo.

http://www.ncl.ucar.edu/Applications/fcodes.shtml

Os mais usados são: *"∼S∼"* (superescrito) e *"∼B∼"* (subescrito).

O *∼N∼* quer dizer para voltar ao modo normal, porque se deixarmos apenas o *∼S∼* ou o *∼B∼*, tudo que vier após ficará no modo superescrito ou subescrito, respectivamente.

Alguns exemplos:

velocidade: m s∼S∼-1∼N∼ ⇒ m s^{-1}

radiação: W m∼S∼-2∼N∼ ⇒ W m^{-2}

temperatura: ∼S∼o∼N∼C ⇒ ^{o}C

alt. geop. em 200 hpa - Jan: zgeo∼B∼200hpa∼N∼ - Jan ⇒ $zgeo_{200hpa}$ - Jan

Neste último exemplo, foi adicionado o símbolo *∼N∼*, sem ele o "- Jan" ficaria no modo superescrito.

Caso não haja mais texto/símbolo após o subescrito ou superescrito, não há necessidade de acrescentar o *∼N∼* como nos dois primeiros exemplos acima.

9 Cores, fontes, estilos de linha, marcadores e projeções

9.1 Tipos de cores

Há um número considerável de cores disponíveis no NCL. Além das tabelas, o usuário tem a liberdade de criar suas próprias cores.

O link abaixo mostra o site para as tabelas de cores disponíveis. Explore essa página para visualizar as diversas possibilidades.

http://www.ncl.ucar.edu/Document/Graphics/color_table_gallery.shtml

Há também a seleção pelo nome da cor. Basta acessar no link abaixo e selecionar a tabela desejada para mostrar os nomes das cores.

http://www.ncl.ucar.edu/Document/Graphics/named_colors.shtml

9.2 Tipos de fonte

Há diversas opções para personalizar o tipo de fonte. O link abaixo mostra algumas possibilidades. Basta selecionar a tabela de fonte desejada no link abaixo:

http://www.ncl.ucar.edu/Document/Graphics/font_tables.shtml

9.3 Estilos de linha

Os estilos de linha podem ser obtidos no link abaixo.

http://www.ncl.ucar.edu/Document/Graphics/Images/dashpatterns.png

9.4 Padrão de preenchimento

Os padrões de preenchimento podem ser obtidos no link abaixo.

http://www.ncl.ucar.edu/Document/Graphics/Images/fillpatterns.png

9.5 Tipos de marcadores

O link abaixo mostra os tipos de marcadores disponíveis.

http://www.ncl.ucar.edu/Document/Graphics/Images/markers.png

9.6 Tipos de projeções

Os tipos de projeções suportados podem ser visualizados no link abaixo.

http://www.ncl.ucar.edu/Document/Graphics/map_projections.shtml

10 Manipulando arquivo texto

O objetivo deste capítulo consiste em auxiliar na manipulação de arquivo texto.

Para informações sobre manipulação de arquivos CSV (comma-separated values) acesse os links abaixo:

http://www.ncl.ucar.edu/Applications/write_csv.shtml

http://www.ncl.ucar.edu/Applications/read_csv.shtml

10.1 Lendo um arquivo texto com o asciiread

Se o dado é do tipo numérico e há caracteres não-numéricos, eles serão ignorados e os números serão lidos na ordem que aparecem no arquivo de entrada. Tenha cuidado ao declarar o número certo de elementos a serem lidos.

Caso seu arquivo possua "nan", ele será lido como um valor numérico. Por essa, é melhor tratar esses valores para depois manipular o dado.

A separação entre as colunas pode conter TAB, espaço, vírgula ou um caracter alfanumérico.

Para dados do tipo string, eles serão lidos como uma string. Para os dados do tipo caracter, o arquivo será lido byte por byte.

Se a opção "-1" é passada no parâmetro dimensões, todos os valores do arquivo serão lidos como uma variável unidimensional. O tamanho da dimensão desta variável será igual ao número de elementos do arquivo.

O NCL até lê valores númericos com vírgula, só que a leitura é feita de forma distinta, onde a parte inteira é lida separadamente da parte decimal. O ideal é que o seu arquivo tenho ponto no lugar de vírgula.

Para mais informações:

http://www.ncl.ucar.edu/Document/Functions/Built-in/asciiread.shtml

Supondo o arquivo "temp.txt" abaixo com 4 linhas e 3 colunas, isto é, temperatura mínima (coluna da esquerda), média (coluna central) e máxima (coluna da direita), respectivamente. Para abrir esse arquivo será utilizado o asciiread.

22.1	26.6	30.9
22.2	24.5	30.5
22.4	25.5	30.4
21.8	26.7	30.8

Abertura do arquivo com o asciiread.

```
txt1 = asciiread("temp.txt",(/4,3/),"float")
print(txt1)
```

Onde: temp.txt é o arquivo aberto, (/4,3/) corresponde a dimensão (4 linhas por 3 colunas) do arquivo e "float" é o tipo da variável. Nesse caso, foram lidos 4x3 = 12 elementos do arquivo, e a leitura é feita linha a linha.

O resultado será:

```
Variable: txt1
Type: float
Total Size: 48 bytes
            12 values
Number of Dimensions: 2
Dimensions and sizes: [4] x [3]
Coordinates:
Number Of Attributes: 1
_FillValue : 9.96921e+36
(0,0) 22.1
(0,1) 26.6
(0,2) 30.9
(1,0) 22.2
(1,1) 24.5
(1,2) 30.5
(2,0) 22.4
(2,1) 25.5
(2,2) 30.4
(3,0) 21.8
(3,1) 26.7
(3,2) 30.8
```

Também é possível realizar a abertura de apenas alguns elementos do arquivo.

```
txt2 = asciiread("temp.txt",(/2,4/),"float")
print(txt2)
```

Serão lidos apenas 8 (2x4=8) elementos do arquivo temp.txt. Lembrando que a leitura será feita linha a linha.

O resultado será:

```
Variable: txt2
Type: float
Total Size: 32 bytes
            8 values
Number of Dimensions: 2
Dimensions and sizes: [2] x [4]
Coordinates:
Number Of Attributes: 1
_FillValue : 9.96921e+36
(0,0) 22.1
(0,1) 26.6
```

(0,2) 30.9
(0,3) 22.2
(1,0) 24.5
(1,1) 30.5
(1,2) 22.4
(1,3) 25.5

Outra possibilidade de abertura do arquivo pode ser feita com a opção "-1". Lembrando que neste caso o arquivo será lido e armazenado em uma variável unidimensional.

```
txt3 = asciiread("temp.txt",-1,"float")
print(txt3)
```

O resultado do print(txt3) será:

```
Variable: txt3
Type: float
Total Size: 48 bytes
            12 values
Number of Dimensions: 1
Dimensions and sizes: [12]
Coordinates:
Number Of Attributes: 1
_FillValue : 9.96921e+36
(0) 22.1
(1) 26.6
(2) 30.9
(3) 22.2
(4) 24.5
(5) 30.5
(6) 22.4
(7) 25.5
(8) 30.4
(9) 21.8
(10) 26.7
(11) 30.8
```

Lendo arquivos com cabeçalho que apresentam caracteres alfanuméricos e númericos.

O arquivo abaixo (tempN.txt) contém valores numéricos e não-numéricos. A primeira linha contém col1, col2 e col3. Lembrando que o asciiread somente lê valores numéricos, mas neste caso serão lidos os valores 1, 2 e 3 de col1, col2 e col3 porque eles são numéricos.

col1	col2	col3
Tmin	Tmed	Tmax
22.1	26.6	30.9
22.2	24.5	30.5
22.4	25.5	30.4
21.8	26.7	30.8

A leitura é feita com a opção "-1".

```
txt4 = asciiread("tempN.txt",-1,"float")
print(txt4)
```

O resultado será:

```
Variable: txt4
Type: float
Total Size: 60 bytes
            15 values
Number of Dimensions: 1
Dimensions and sizes: [15]
Coordinates:
Number Of Attributes: 1
_FillValue : 9.96921e+36
(0) 1 ⟹ 1 de col1
(1) 2 ⟹ 2 de col2
(2) 3 ⟹ 3 de col3
(3) 22.1 ⟹ a partir daqui começam os dados, no índice 3
(4) 26.6
(5) 30.9
(6) 22.2
(7) 24.5
(8) 30.5
(9) 22.4
(10) 25.5
(11) 30.4
(12) 21.8
(13) 26.7
(14) 30.8
```

Agora, deseja-se apenas os valores de interesse, isto é, do índice 3 ao 14, como resolver isso?

```
txt4 = asciiread("tempN.txt",-1,"float")
txt5 = onedtond(txt4(3:), (/4,3/))
print(txt5)
```

A função onedtond converte um vetor para uma matriz multidimensional.

O índice onde começa o dado precisa ser conhecido, no caso anterior, o dado começa no índice 3.

Lembrando que serão lidos 12 elementos, por isso, (/4,3/).

O resultado será:

```
Variable: txt5
Type: float
Total Size: 48 bytes
            12 values
Number of Dimensions: 2
Dimensions and sizes: [4] x [3]
Coordinates:
Number Of Attributes: 1
_FillValue : 9.96921e+36
(0,0) 22.1
(0,1) 26.6
(0,2) 30.9
(1,0) 22.2
(1,1) 24.5
(1,2) 30.5
(2,0) 22.4
(2,1) 25.5
(2,2) 30.4
(3,0) 21.8
(3,1) 26.7
(3,2) 30.8
```

10.2 Retornando o número de linhas e colunas de um arquivo texto

As funções numAsciiRow e numAsciiCol retornam o número de linhas e de colunas de um arquivo, respectivamente.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/numAsciiCol.shtml

http://www.ncl.ucar.edu/Document/Functions/Contributed/numAsciiRow.shtml

Supondo o arquivo com o nome tmp.txt que contém os seguintes valores:

22.1	26.6	30.9
22.2	24.5	30.5
22.4	25.5	30.4
21.8	26.7	30.8

nlin = numAsciiRow("tmp.txt") ; tmp.txt é o arquivo aberto

ncol = numAsciiCol("tmp.txt")
print("num. de linhas " + nlin + " num. de colunas " + ncol)

O resultado será:

(0) num. de linhas 4 num. de colunas 3

10.3 Salvando arquivo texto

Este tópico consiste em mostrar as diferentes formas de salvar um arquivo no formato texto.

As opções utilizadas podem ser "w" para escrita e "a" para append (adiciona mais informações).

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/write_ascii.shtml

10.3.1 Usando a função write_table

A função write_table escreve todos os elementos de uma lista em um arquivo texto.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/write_table.shtml

Supondo o arquivo temp.txt abaixo:

22.1	26.6	30.9
22.2	24.5	30.5
22.4	25.5	30.4
21.8	26.7	30.8

Abertura do arquivo temp.txt e manipulação do mesmo.

txt = asciiread("temp.txt",(/4,3/),"float")
tmin = txt(:,0)+273.15 ; leitura de todas as linhas (:) da primeira coluna (0)
tmed = txt(:,1)+273.15 ; leitura de todas as linhas (:) da segunda coluna (1)
tmax = txt(:,2)+273.15 ; leitura de todas as linhas (:) da terceira coluna (2)

Escrita das informações no arquivo tempKelvin.txt.

write_table("tempKelvin.txt","w",[/tmin,tmed,tmax/], "%4.2f %4.2f %4.2f")

Salva no arquivo tempKelvin.txt as temperaturas (tmin,tmed,tmax) com os formatos passado no último argumento ("%4.2f %4.2f %4.2f").

A opçao "w" escreve no arquivo de saída. Caso fosse necessário adicionar mais informações, utiliza-se a opção "a".

Para adicionar mais espaços entre as colunas de temperatura, basta inserir mais espaços entre os formatos ("%4.2f %4.2f %4.2f").

O resultado do arquivo tempKelvin.txt será:

295.25	299.75	304.05
295.35	297.65	303.65
295.55	298.65	303.55
294.95	299.85	303.95

Supondo que deseja-se adicionar um cabeçalho ao arquivo tempKelvin.txt. Como fazer isso? Utiliza-se a opção "w" e "a" da seguinte forma:

write_table("tempKelvin.txt","w",[/"TminK TmedK TmaxK"/], "%s %s %s") ; escreve no arquivo com a opção "w"

write_table("tempKelvin.txt","a",[/tmin,tmed,tmax/], "%4.2f %4.2f %4.2f") ; adiciona mais informações com a opção "a"

10.3.2 Outras possibilidades de escrita no formato texto

Outra possibilidade é o uso da função asciiwrite. A desvantangem desta função está no fato de que ela é limitada para escrita de valores, ou escreve-se apenas valores numéricos ou strings. Ela deve ser utilizada para escrita de pequenos arquivos. O usuário não tem controle sobre a formatação do dado que pode ser feita usando a função sprintf.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/asciiwrite.shtml

Há também a função write_matrix que escreve a saída dos dados para uma saída padrão ou para um arquivo.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/write_matrix.shtml

10.4 Manipulando strings

10.5 Funcão str_fields_count

Conta o número de campos separados por um delimitador em um arranjo de strings.

Os delimitadores podem ser TAB, espaços, dois pontos, ponto e vírgula, vírgulas, etc. Também pode ser uma combinação de delimitadores.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/str_fields_count.shtml

Exemplo1:

a = "This;is.a:test:of=the/string\tokenizer-function."
delim = ":;=-"

fa = str_fields_count(a, delim)
print(fa)

O resultado será 8. Observe que o delimitador "." não está na lista de delimitadores.

O resultado será: (1) This, (2) is.a, (3) test, (4) of, (5) the, (6) string, (7) tokenizer e (8) function

Exemplo2:

b = "I am a string."
fb = str_fields_count(b, " ")
print(fb) ; Resuldo: fb = 4

Exemplo3:

b = "I am a string."
fc = str_fields_count(b, ",")
print(fc) ; Resultado: fc = 1

Quando um delimitador não está na lista, será retornado o valor 1.

10.6 Funções str_get_cols e str_get_field

Retorna uma matriz de substrings dado um índice para a string de interesse.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/str_get_cols.shtml

http://www.ncl.ucar.edu/Document/Functions/Built-in/str_get_field.shtml

Supondo o arquivo abaixo (temp.txt). A primeira coluna corresponde a data no formato AAAAMMDD. Esse arquivo possui 4 campos, nessa ordem, coluna 1 (campo 1), coluna 2 (campo 2), coluna 3 (campo 3) e coluna 4 (campo 4), contando da esquerda para a direita.

20140101	22.1	26.6	30.9
20140102	22.2	24.5	30.5
20140103	22.4	25.5	30.4
20140104	21.8	26.7	30.8

A função str_fields_count conta o número de campos separados por um ou mais espaços. Nesse caso, será selecionado apenas o primeiro campo (arq(0)).

arq = asciiread("temp.txt",-1,"string")
ncampos = str_fields_count(arq(0)," ")
print(ncampos)

Resultado:

Variable: ncampos

Type: integer
Total Size: 4 bytes
1 values
Number of Dimensions: 1
Dimensions and sizes: [1]
Coordinates:
(0) 4

Selecionando o campo de interesse. Nessa caso, foi selecionado o primeiro campo (exemplo abaixo) do arquivo aberto, isto é, a coluna das datas.

substrings = str_get_field(arq,1,“ ”)
print(substrings)

Resultado:

Variable: substrings
Type: string
Total Size: 32 bytes
4 values
Number of Dimensions: 1
Dimensions and sizes: [4]
Coordinates:
(0) 20140101
(1) 20140102
(2) 20140103
(3) 20140104

A partir da coluna das datas, é selecionado apenas o ano com a função str_get_cols. O valor 0 e 3 corresponde a posição do ano da variável substrings, isto é:

2	0	1	4	0	1	0	1
2	0	1	4	0	1	0	2
2	0	1	4	0	1	0	3
2	0	1	4	0	1	0	4
0	1	2	3	4	5	6	7

O que está em vermelho corresponde a posição da string.

ano = str_get_cols(substrings,0,3)
print(ano)

Resultado:

Variable: ano
Type: string
Total Size: 32 bytes
4 values
Number of Dimensions: 1
Dimensions and sizes: [4]

Coordinates:
(0) 2014
(1) 2014
(2) 2014
(3) 2014

O mesmo racicíonio do ano é feito para obter a string mês.

2	0	1	4	0	1	0	1
2	0	1	4	0	1	0	2
2	0	1	4	0	1	0	3
2	0	1	4	0	1	0	4
0	1	2	3	4	5	6	7

O que está em vermelho corresponde a posição da string.

mes = str_get_cols(substrings,4,5)

print(mes)

Resultado:

Variable: mes
Type: string
Total Size: 32 bytes
4 values
Number of Dimensions: 1
Dimensions and sizes: [4]
Coordinates:
(0) 01
(1) 01
(2) 01
(3) 01

O mesmo racicíonio do ano é feito para obter a string dia.

2	0	1	4	0	1	0	1
2	0	1	4	0	1	0	2
2	0	1	4	0	1	0	3
2	0	1	4	0	1	0	4
0	1	2	3	4	5	6	7

O que está em vermelho corresponde a posição da string.

dia = str_get_cols(substrings,6,7)
print(dia)
Resultado:

Variable: dia

Type: string
Total Size: 32 bytes
4 values
Number of Dimensions: 1
Dimensions and sizes: [4]
Coordinates:
(0) 01
(1) 02
(2) 03
(3) 04

10.7 Função strs_sub_str

Substitui uma string por outra.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/str_sub_str.shtml

aob = "apple_orange_banana"
o = "orange"
s = "strawberry"
aob = str_sub_str(aob,o,s)
print(aob)

A string orange na variável "aob" será substituída por strawberry.

O resultado será:

(0) apple_strawberry_banana

11 Salvando e manipulando arquivo NetCDF

11.1 Salvando arquivo no formato NetCDF

Há algumas possibilidades para criação de um arquivo NetCDF.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/write_netcdf.shtml

Importante: A opção "c" cria um novo arquivo caso ele não exista e o usuário tem que ter permissão de escrita no diretório. Caso o arquivo exista, será retornado erro, por isso a importância de remover o arquivo antes de executar o script.

A opção "w" é utilizada quando o arquivo já existe e o usuário deseja alterar algo dentro dele.

11.2 Método simples

Link para o método simples:

http://www.ncl.ucar.edu/Applications/method_1.shtml

Exemplo1: Uso do método simples para cria um arquivo NetCDF.

```
; Nome do script: cap11_ex01.ncl

begin

f = addfile("../../dados/tair.2011.2012.nc","r") ; Abertura do arquivo.

t = short2flt(f->air) ; Descompactando o dado (short) para o tipo float.

tc = t-273.15 ; Converte de Celsius para Kelvin.

copy_VarCoords(t,tc) ; Copia as coordenadas da variável original "t" para
                     ; a nova variável "tc". A variável "tc" não possui
                     ; nenhuma coordenada associada, por
                     ; isso utilizou-se essa função.

system("rm -f ../../dados/temp.celsius.nc") ; Remove o arquivo caso ele exista.

; Arquivo NetCDF a ser criado ("c") no seu computador.
nc = addfile("../../dados/temp.celsius.nc" ,"c")

filedimdef(nc,"time",-1,True) ; Possibilidade de aumentar o número de tempos
                              ; da dimensão time.

nc->tar = tc ; "tar" é o nome da nova variável do arquivo netCDF que será
             ; criado e "tar" receberá o conteúdo da variável "tc" que foi
             ; calculada na linha 7.
end
```

11.3 Método avançado

Exemplo2: Uso do método avançado para criar um arquivo NetCDF.

Informações adicionais podem ser acessadas em:

http://www.ncl.ucar.edu/Applications/method_2.shtml

```
; Nome do script: cap11_ex02.ncl

begin

f = addfile("../../dados/tair.2011.2012.nc","r") ; Abertura do arquivo.

T  = short2flt(f->air) ; Descompactando o dado (short) para o tipo float.

Tc = T-273.15 ; Converte de Celsius para Kelvin.

x = dimsizes(T) ; (0)   24  => Número de tempos              x(0).
                ; (1)   12  => Número de níveis verticais    x(1).
                ; (2)   73  => Número de pontos de latitude  x(2).
                ; (3)   144 => Número de pontos de longitude x(3).
nlat = x(2)
nlon = x(3)
nlev = x(1)

time  = f->time  ; Importando as variáveis
level = f->level ; para serem utilizadas
lat   = f->lat   ; para criar as coordenadas
lon   = f->lon   ; e dimensões.

system("rm -f ../../dados/tc.MA.nc") ; Remove o arquivo caso ele exista.

; Arquivo NetCDF a ser criado ("c") no seu computador.
nc = addfile("../../dados/tc.MA.nc" ,"c")

setfileoption(nc,"DefineMode",True)

; Criação de atributos globais.
fAtt               = True
fAtt@title         = "Criação eficiente de um arquivo netCDF"
fAtt@source_file   = "tair.2011.2012.nc"
fAtt@Conventions   = "Nenhum convenção adotada"
fAtt@contact       = "Meu e-mail para contato"
fAtt@creation_date = systemfunc ("date")
fileattdef(nc,fAtt)     ; Copia os atributos globais para o arquivo
                        ; que será gerado (tc.MA.nc).

; Predefinição das coordenadas das variáveis. A opção "-1" é usada
; para a dimensão time para ser definida como UNLIMITED (possibilidade
; de aumentar o número de tempos).
dimNames = (/"time", "lat", "lon", "level"/)
dimSizes = (/ -1   ,  nlat,  nlon, nlev /)
dimUnlim = (/ True , False, False, False/)
filedimdef(nc,dimNames,dimSizes,dimUnlim)

; Predefinição dos nomes das variáveis, tipos e dimensões do arquivo
; a ser gerado.
filevardef(nc,"time" ,typeof(time) ,getvardims(time))
filevardef(nc,"level",typeof(level),getvardims(level))
```

```
filevardef(nc,"lat"  ,typeof(lat)  ,getvardims(lat))
filevardef(nc,"lon"  ,typeof(lon)  ,getvardims(lon))
filevardef(nc,"tc"   ,typeof(T)    ,getvardims(T))

filevarattdef(nc,"tc"  ,T)      ; Copia os atributos de T para "Tc".
filevarattdef(nc,"time",time)   ; Copia os atributos de time para "time".
filevarattdef(nc,"level",level); Copia os atributos de lev para "level".
filevarattdef(nc,"lat" ,lat)    ; Copia os atributos de lat para "lat".
filevarattdef(nc,"lon" ,lon)    ; Copia os atributos de lon para "lon".

; Escrita das variáveis no novo arquivo tc.MA.nc. O símbolo (/.../)
; significa copiar somente os valores da variável que estão entre (/.../)
; para as novas variáveis, uma vez que, elas foram predefinidas ao
; longo do programa.

nc->time  = (/time/)
nc->level = (/level/)
nc->lat   = (/lat/)
nc->lon   = (/lon/)
nc->tc    = (/Tc/)

end
```

11.4 Editando o arquivo NetCDF

A função addfile pode ser utilizada para ler, criar ou editar um arquivo NetCDF. Para editar o arquivo deve-se usar a opção "w" como argumento da função addfile. Neste caso, não é possível mudar a dimensionalidade de qualquer variável existente, mas pode-se alterar os seus valores.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/change_netCDF.shtml

Exemplo:

```
1   ; Nome do script: cap11_ex03.ncl
2
3   begin
4
5   ; As informações abaixo são do comando ==> ncdump prec.2005.nc
6   ;
7   ;netcdf prec.2005 {
8   ;dimensions:
9   ;       lon = 1 ;
10  ;       lat = 1 ;
11  ;       time = UNLIMITED ; // (12 currently)
12  ;variables:
13  ;       double lon(lon) ;
14  ;               lon:standard_name = "longitude" ;
15  ;               lon:long_name = "longitude" ;
16  ;               lon:units = "degrees_east" ;
17  ;               lon:axis = "X" ;
18  ;       double lat(lat) ;
19  ;               lat:standard_name = "latitude" ;
20  ;               lat:long_name = "latitude" ;
21  ;               lat:units = "degrees_north" ;
```

```
22  ;                lat:axis = "Y" ;
23  ;        double time(time) ;
24  ;                time:standard_name = "time" ;
25  ;                time:long_name = "time" ;
26  ;                time:units = "days since 1900-1-1 00:00:00" ;
27  ;                time:calendar = "standard" ;
28  ;                time:axis = "T" ;
29  ;        double pre(time, lat, lon) ;
30  ;                pre:long_name = "precipitation" ;
31  ;                pre:units = "mm" ;
32  ;                pre:_FillValue = 9.96920996838687e+36 ;
33  ;                pre:missing_value = 9.96920996838687e+36 ;
34  ;                pre:correlation_decay_distance = 450. ;
35  ;data:
36  ;
37  ; lon = 0 ;
38  ;
39  ; lat = 0 ;
40  ;
41  ; time = 38366, 38396, 38425, 38456, 38486, 38517, 38547, 38578, 38609,
42  ;        38639, 38670, 38700 ;
43  ;
44  ; pre =
45  ;  5.31968154222574,
46  ;  5.59480643916749,
47  ;  6.4032231783253,
48  ;  5.18781546763753,
49  ;  4.98682642410234,
50  ;  3.40967712475132,
51  ;  2.50282044695336,
52  ;  2.63375088184085,
53  ;  2.59883242095348,
54  ;  3.97419849336847,
55  ;  4.22262434923237,
56  ;  5.93353982877502 ;
57  ;}
58
59  f = addfile("prec.2005.nc","w") ; Abertura do arquivo para edição ("w").
60
61  lat_n = -10 ; Alterando o valor da latitude que antes era "0".
62  lon_n = 300 ; Alterando o valor da longitude que antes era "0".
63
64  ; São os novos valores alterados.
65  prec_n = (/6.2,6.4,7.0,4.0,3.5,2.0,8.3,3.2,5.5,1.8,0.4,3.6/)
66
67  ; O nome "pre" é o nome da variável existente no arquivo que receberá
68  ; o novo conteúdo da variável "prec_n".
69
70  f->pre = (/prec_n/) ; O símbolo (/.../) apenas salva os valores da
71  f->lat = (/lat_n/)  ; variável.
72  f->lon = (/lon_n/)
73
74  end
```

12 Convertendo dados grib para NetCDF

É possível converter dados no formato grib para NetCDF por meio do programa ncl_convert2nc.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Tools/ncl_convert2nc.shtml

http://www.ncl.ucar.edu/Applications/gribselvar.shtml

http://www.ncl.ucar.edu/Applications/griball.shtml

Exemplo1: Esse é o método mais recomendado.

ncl_convert2nc GPOSNMC19701201001971010100P.fct.TQ0062L028.grb

Será gerado o arquivo GPOSNMC19701201001971010100P.fct.TQ0062L028.nc com a extensão nc.

Exemplo2: Caso o usuário necessite de um maior controle sobre a geração do arquivo.

```
; Nome do script: cap12_ex01.ncl

begin

f = addfile("../../dados/GPOSNMC19701201001971010100P.fct.TQ0062L028.grb","r")

file_out = "../../dados/saida.nc" ; Nome do arquivo NetCDF a ser criado.

names = getfilevarnames(f) ; Obtém uma lista de todas as variáveis
                           ; do arquivo aberto.

print(names) ; Imprime uma lista com o nome das variáveis.

; Cria o nome do arquivo NetCDF.

system("rm -f "+file_out) ; Remove qualquer arquivo existente.
                          ; Sem essa linha gera erro.

ncdf_out = addfile(file_out,"c") ; Cria o NetCDF.

; Faz o loop de todas as variáveis e salva no arquivo NetCDF.

do i = 0, dimsizes(names)-1
   ncdf_out->$names(i)$ = f->$names(i)$
end do
```

Exemplo3: Selecionando variáveis específicas do arquivo.

```
; Nome do script: cap12_ex02.ncl

begin

f = addfile("../../dados/GPOSNMC19701201001971010100P.fct.TQ0062L028.grb",

names = getfilevarnames(f) ; Obtém uma lista de todas as variáveis
                           ; do arquivo aberto.

print(names) ; Imprime o nome das variáveis.

file_out = "../../dados/saida.nc" ; Nome do arquivo NetCDF a ser criado.

; Variáveis específicas para salvar no NetCDF.

namesVar = (/"ZGEO_GDS4_ISBL","UVEL_GDS4_ISBL","VVEL_GDS4_ISBL"/)

; Troca dos nomes de "ZGEO_GDS4_ISBL" para "zgeo", "UVEL_GDS4_ISBL" para
; "uvel" e "VVEL_GDS4_ISBL" para "vvel".

novo_nome = (/"zgeo","uvel","vvel"/)

; Cria o nome do arquivo NetCDF.

system("rm -f "+file_out) ; Remove qualquer arquivo existente.

ncdf_out = addfile(file_out,"c") ; Cria o NetCDF.

; Faz o loop de todas as variáveis e salva no arquivo NetCDF.

do i = 0, dimsizes(namesVar)-1
   ncdf_out->$novo_nome(i)$ = f->$namesVar(i)$
end do

end
```

13 Informações diversas usando as funções get

13.1 Função get_cpu_time

Retorna o tempo de CPU usado pelo NCL. O resultado é dado em segundos.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/get_cpu_time.shtml

Exemplo: Pequeno trecho de um script.

begin

begTime = get_cpu_time()

Corpo do seu script

print("Tempo total: " + (get_cpu_time() - begTime) + " segundos")

end

13.2 Função get_file_suffix

Extrai o sufixo associado com o nome do arquivo. As opções disponíveis são 0 para extrair apenas o sufixo mais a direita do arquivo e 1 para extrair todos os sufixos do arquivo.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/get_file_suffix.shtml

Exemplo1:

s1 = "GPOSNMC19701201001971010100P.fct.TQ0062L028.grb"
sfx1 = get_file_suffix(s1,0)

print(""+sfx1)

O resultado será: **.grb**

Exemplo2:

s2 = "GPOSNMC19701201001971010100P.fct.TQ0062L028.grb.tar.gz"

sfx2 = get_file_suffix(s2,1)

print(""+sfx2)

O resultado será: **.fct.TQ0062L028.grb.tar.gz**

13.3 Função get_unique_values

Retorna os valores do arranjo que não se repetem. O dado de entrada pode ser um arranjo de qualquer dimensão e a saída será um escalar ou um arranjo de uma dimensão contendo os valores que não se repetem.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/get_unique_values.shtml

Exemplo:

a = (/1,2,3,4,5,5,4,3,2,1,1,2,3,4,5/)

au = get_unique_values(a) ; Resultado: (/1,2,3,4,5/)

ac = count_unique_values(a) ; Conta o total de valores. Resultado: 5

13.4 Função getfiledimsizes

Retorna uma lista que contém o tamanho das dimensões do arquivo.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/getfiledimsizes.shtml

Exemplo:

f = addfile("GPCP.jan.1979.nc","r")

dSizes = getfiledimsizes(f)

print (dSizes)

O resultado do comando getfiledimsizes será:

```
Variable: dSizes
Type: integer
Total Size: 12 bytes
            3 values
Number of Dimensions: 1
Dimensions and sizes: [3]
Coordinates:
(0) 144
(1) 72
(2) 1
```

O resultado mostrou que há 144 pontos de longitude, 72 pontos de latitude e 1 tempo.

13.5 Função getfilevaratts

Retorna todos os nomes dos atributos do arquivo associado com a variável do arquivo corrente.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/getfilevaratts.shtml

Exemplo:

f = addfile ("GPCP.jan.1979.nc","r")
pptAtts = getfilevaratts(f,"precip") ; precip é o nome da variável do arquivo
print (pptAtts)

O resultado do comando acima será:

Variable: pptAtts
Type: string
Total Size: 96 bytes
12 values
Number of Dimensions: 1
Dimensions and sizes: [12]
Coordinates:
(0) long_name
(1) units
(2) _FillValue
(3) missing_value
(4) actual_range
(5) precision
(6) least_significant_digit
(7) var_desc
(8) dataset
(9) level_desc
(10) statistic
(11) parent_stat

13.6 Função getfilevarnames

Retorna o nome das variáveis do arquivo.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/getfilevarnames.shtml

Exemplo:

f = addfile ("GPCP.jan.1979.nc","r")
vNames = getfilevarnames(f)
print (vNames)

O resultado será:

Variable: vNames
Type: string
Total Size: 32 bytes
4 values
Number of Dimensions: 1
Dimensions and sizes: [4]
Coordinates:
(0) time

(1) lon
(2) lat
(3) precip

13.7 Função getfilevartypes

Retorna o tipo (float, double, integer, short) de dado das variáveis do arquivo.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/getfilevartypes.shtml

Exemplo:

f = addfile ("GPCP.jan.1979.nc","r")
varNames = getfilevarnames(f) ; guarda o nome de todas as variáveis
varTypes = getfilevartypes(f,varNames) ; realiza a classificação de cada variável
print("Tipo de " + varNames + " eh " + varTypes)

O resultado será:

(0) Tipo de time eh double
(1) Tipo de lon eh float
(2) Tipo de lat eh float
(3) Tipo de precip eh float

13.8 Função get1Dindex

Retorna os índices correspondentes as datas especificadas pelo usuário.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/get1Dindex.shtml

Exemplo1: Serão criadas duas variáveis, uma chamadada "ano" com os anos de 2000 a 2010, e outra, "ano_interesse" com os anos específicos de interesse. A variável "i" retornará os indíces dos anos especificados pelo usuário. A linha inferior da tabela abaixo representa o índice da cada ano resultante da função ispan.

2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010
0	1	2	3	4	5	6	7	8	9	10

ano = ispan(2000,2010,1) ; Cria um vetor com 11 posições de 2000 a 2010.

ano_interesse = (/2002,2005,2006,2008/) ; Lista com os anos de interesse.

i = get1Dindex(ano,ano_interesse) ; Retorna os índices dos anos de interesse.

O resultado serão os índices 2, 5, 6 e 8 que estão destacados em vermelho na tabela acima.

13.9 Função get1Dindex_Collapse

Retorna uma matriz de índices subscritos que não coincidem com os valores contidos em uma lista de usuários especificada.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/get1Dindex_Collapse.shtml

Exemplo:

```
; Nome do script: cap13_ex01.ncl

begin

year            = ispan(2000,2015,1) ; Cria um vetor de 2000 a 2015
                                     ; com incremento 1.
year@_FillValue = -999               ; Defini o valor indefinido.

year_exc  = (/2001,2005,2007,2010,2014/) ; Anos que serão excluídos.
                                         ; A ideia é obter o índice
                                         ; dos anos que estão fora
                                         ; desta lista.

i = get1Dindex_Collapse(year,year_exc) ; Obtenção dos índices
                                       ; sem os anos da variável
                                       ; year_exc.

print( " i = " + i + "   " + year(i) ) ; Mostra na tela o resultado.

end
```

O resultado será:

```
(0)       i = 0   2000
(1)       i = 2   2002
(2)       i = 3   2003
(3)       i = 4   2004
(4)       i = 6   2006
(5)       i = 8   2008
(6)       i = 9   2009
(7)       i = 11   2011
(8)       i = 12   2012
(9)       i = 13   2013
(10)      i = 15   2015
```

Os índices que foram excluídos com base na variável year_exc foram: 1, 5, 7, 10 e 14 que correspondem aos anos de 2001, 2005, 2007, 2010 e 2014, respectivamente. Observe que eles não aparecem no resultado.

A função get1Dindex_Exclude é similar a função get1Dindex_Collapse.

13.10 Função getvardims

Retorna o nome das dimensões do arquivo.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/getvardims.shtml

Exemplo:

```
f = addfile("GPCP.jan.1979.nc","r")
dNames = getvardims(f)
print(dNames)
```

O resultado será:

```
Variable: dNames
Type: string
Total Size: 24 bytes
            3 values
Number of Dimensions: 1
Dimensions and sizes: [3]
Coordinates:
Number Of Attributes: 1
_FillValue : missing
(0) time
(1) lon
(2) lat
```

14 Funções do NCL

Todas as funções podem ser encontradas no link abaixo:

http://www.ncl.ucar.edu/Document/Functions/list_alpha.shtml

ou por categoria:

http://www.ncl.ucar.edu/Document/Functions

14.1 Função addfile

Abre um arquivo de dados que é (ou deve ser) gravado em um formato de arquivo suportado.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/addfile.shtml

Opções de uso:

1. r = para leitura de um arquivo
2. c = para criação de um arquivo

3. w = para edição de um arquivo

Exemplo1: Abertura de um arquivo para leitura (r):

a = addfile("ex01B1_uv300.hs.nc","r")

print(a)

Exemplo2: Criação (c) de um arquivo:

a = addfile("ex01B1_uv300.hs.nc","r") ; abertura do arquivo.

system("rm -f novo.nc") ; remove o arquivo caso ele exista. Tem que ter esse "rm" para não retornar erro na criação do novo arquivo.

b = addfile("novo.nc","c") ; cria o arquivo novo.nc.

Exemplo3: Apenas para edição (w) de um arquivo já existente:

a = addfile("foo.nc","w")

t = a->TEMP ; lê a variável em graus Celsius.

t = t + 273.15 ; altera os valores da variável t.

t@units = "degK" ; atualiza o atributo units.

a->TEMP = t ; sobrescreve a variável TEMP do arquivo foo.nc.

14.2 Função addfiles

Utilizado para ler vários arquivos de uma vez. Nota-se que há um "s" diferente do addfile.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/addfiles.shtml

Exemplo1:

files = systemfunc("ls *.nc") ; lista todos os arquivos ".nc" do diretório corrente
f = addfiles(files,"r") ; apenas para leitura (r)
T_agg = f[:]->T(:,0:5,:,:) ; para referenciar uma variável, utiliza-se o símbolo [:]

Pode-se utilizar apenas um subconjunto dos dados listados. Basta substituir o símbolo ":" pelo subconjunto de dados de interesse.

T_agg = f[0:10:2]->T(:,0:5,:,:)

Algumas particularidades desta função incluem o uso das opções cat e join no momento de manipular o dado.

Exemplo2: Uso da opção cat. Supondo que há 5 arquivos sendo que cada um possui 12 tempos. Ao concatenar esses arquivos, a dimensão tempo passará a ter 60 tempos (5 arquivos x 12 tempos cada = 60 tempos) como está destacado em vermelho no resultado do printVarSummary(T).

fils = systemfunc ("ls /model/annual*.nc") ; lista todos os arquivos annual*.nc
f = addfiles (fils,"r") ; abre o arquivo para leitura
ListSetType (f,"cat") ; concatena os dados (opção padrão)
T = f[:]->T ; realiza a leitura da variável T de todos os arquivos
printVarSummary(T)

O resultado do printVarSummary(T) será:

```
Variable: T
Type: float
Total Size: 5529600 bytes
            1382400 values
Number of Dimensions: 4
Dimensions and sizes: [time |60] x [lev 5] x [lat |48] x [lon |96]
Coordinates:
            time: [2349..4143]
            lev: [850..200]
            lat: [-87.15909..87.15909]
            lon: [ 0..356.25]
Number Of Attributes: 2
            units : K
            long_name : temperature
```

O opção join é útil para adicionar uma nova dimensão ao dado.

Exemplo3: Supondo que os arquivos "XXX.nc" possuem as dimensões level, latitude e longitude. Ao utilizar a opção join será adicionada uma dimensão extra chamada ncl_join.

diri = "/fs/cgd/data0/casguest/CLASS/" ; diretório onde estão os dados
fils = systemfunc ("ls "+diri+"XXX*.nc") ; lista os arquivos
f = addfiles (fils,"r")
ListSetType (f,"join")
T = f[:]->T ; leitura da variável T de todos os arquivos
printVarSummary(T)

O resultado do printVarSummary(T) será:

```
Variable: T
Type: float
Total Size: 460800 bytes
            115200 values
Number of Dimensions: 5
Dimensions and sizes: [ncl_join |5] x [lev |5] x [lat |48] x [lon |96]
Coordinates:
            lev: [850000..250.]
            lat: [-87.15909..87.15909]
            lon: [ 0..356.25]
Number Of Attributes: 2
```

units : K
long_name : temperature

14.3 Função all

Retorna True se todos os elementos de entrada são avaliados como True.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/all.shtml

Exemplo:

x = new(5,float,-999) ; cria uma variável vazia x com 5 posições do tipo float e com valor ausente -999.
x = (/1.,2.,-999,4.,5./) ; "preenchimento" dos valores da variável x

print(all(x.ge.5)) ; Resposta: False. A explicação seria, todos os valores de x (1.,2.,-999,4.,5.) maior ou igual a 5. Logo, não há, e por isso, o valor False.

print(all(x.gt.0.and.x.lt.6)) ; Resposta: True. A explicação seria, todos os valores de x (1.,2.,-999,4.,5.) maior do que 0 e x menor do que 6, por isso, o valor True

14.4 Função any

Retorna True se qualquer um dos valores de entrada são avaliados como True.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/any.shtml

Exemplo1:

x = new(5,float,-999) ; cria uma variável x vazia com 5 posições do tipo float e com valor ausente -999.

x = (/1.,2.,-999,4.,5./) ; "preenchimento" dos valores da variável x

print(any(x.ge.4)) ; Resposta: True

print(any(x.lt.0)) ; Resposta: False

print(any(x.gt.2.and.x.lt.4)) ; Resposta: False

Exemplo2: Verifica se há dados ausentes no arquivo.

```
if(any(ismissing(x))) then
  print("x possui um ou mais valores ausentes, saindo do programa.")
  return
end if
```

14.5 Função avg

Calcula a média de uma variável independente da dimensão. Não considera no cálculo os valores ausentes.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/avg.shtml

Para saber o número de dados utilizado no cálculo da média, utilize o seguinte comando:

n = num(.not.ismissing(x))

Exemplo:

a = (/ (/1,2,3/), (/4,5,6/), (/7,-999,9/)/) ; um arranjo do tipo 3x3.

a@_FillValue = -999 ; definindo o valor ausente -999.

media = avg(a) ; calcula a média de a.

n = num(.not.ismissing(a)) ; conta quantos valores foram utilizados no cálculo.

print(media) ; imprime o valor 4.625 que é a média de a.

print(n) ; imprime o número total de valores utilizados na média, isto é, 8 porque não entrou no cálculo o -999 que é um dado ausente definido acima como _FillValue.

14.6 Função para anomalia

Neste item serão mostrados alguns exemplos para o cálculo de anomalia.

Atenção deve ser dada para a ordem das dimensões. Caso seja necessário realizar a mudança de ordem delas, utilize o símbolo "|" (página 33).

14.6.1 Anomalia diária

Calcula a anomalia diária usando a função calcDayAnomTLL em que as dimensões do dado de entrada devem estar na seguinte ordem: time, latitude e longitude. Essas informações podem ser vistas com o ncl_filedump.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/calcDayAnomTLL.shtml

O script abaixo calcula a anomalia diária de precipitação desde 01/01/1997 até 31/12/2001.

```
; Nome do script: cap14_ex01.ncl

begin

fili = "../../dados/GPCP_1DD_v1.2_19970101-20013112.nc"
f    = addfile (fili,"r")
;*************************************************************
; Leitura do tempo e criação dele no formato yyyyddd
;*************************************************************
time    = f->time
TIME    = cd_calendar(time, 0) ; tipo float
year    = toint(TIME(:,0))
month   = toint(TIME(:,1))
day     = toint(TIME(:,2))
ddd     = day_of_year(year,month,day)
yyyyddd = year*1000 + ddd      ; É necessário como argumento
                               ; para o cálculo da anomalia.
;*************************************************************
; Leitura do dado
;*************************************************************
ppt     = f->PREC
printVarSummary(ppt)
;*************************************************************
; Calcula a climatologia diária bruta e a suavizada.
;*************************************************************
pptClmDay  = clmDayTLL(ppt,yyyyddd) ; Climatologia.
printVarSummary(pptClmDay)
;*************************************************************
; Calcula a climatologia suavizada usando 2 harmônicos.
;*************************************************************
pptClmDay_sm = smthClmDayTLL(pptClmDay,2)
printVarSummary(pptClmDay_sm)
;*************************************************************
; Calcula a anomalia diária usando a climatologia bruta e a suavizada
;*************************************************************
pptAnom   = calcDayAnomTLL(ppt,yyyyddd,pptClmDay) ; Anomalia bruta.
printVarSummary(pptAnom)
printMinMax(pptAnom,0)

; Anomalia suavizada.
pptAnom_sm = calcDayAnomTLL(ppt,yyyyddd,pptClmDay_sm)

pptAnom_sm@long_name = "Anomalia da climatologia diária suavizada"
printVarSummary(pptAnom_sm)
printMinMax(pptAnom_sm,0)

end
```

14.6.2 Anomalia mensal

Calcula a anomalia mensal subtraindo da média de longo período. As dimensões do dado devem estar no formato (time,latitude,longitude) e a dimensão tempo deve ser múltiplo de 12, isto é, todos os meses completos. Note o detalhe do nome das funções clmMonTLL e calcMonAnomTLL. Observe que há TLL que quer dizer time, latitude e longitude.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/calcMonAnomTLL.shtml

Exemplo: Calcula a anomalia mensal de 1979 a 2012.

```
; Nome do script: cap14_ex02.ncl

begin

;*************************************************************
; Abertura do arquivo
;*************************************************************
f   = addfile("../../dados/gpcp.mensal.1979.2012.nc","r")
ppt = f->precip ; float precip(time, lat, lon)
;*************************************************************
; Calcula a climatologia
;*************************************************************
clima = clmMonTLL(ppt)
;*************************************************************
; Calcula a anomalia mensal usando a climatologia
;*************************************************************
pptAnom = calcMonAnomTLL(ppt,clima)

printVarSummary(pptAnom)

end
```

Caso a ordem das dimensões do seu arquivo seja diferente deste exemplo, isto é, time, latitude e longitude, há outras funções para esse cálculo que podem ser utilizadas. Por exemplo:

- calcMonAnomLLT: neste caso a ordem das dimensões é latitude, longitude e time, por isso o LLT.
- calcMonAnomLLLT: neste caso a ordem das dimensões é level, latitude, longitude e time, por isso o LLLT. Muito útil para arquivos que apresentam nível vertical.
- calcMonAnomTLLL: neste caso a ordem das dimensões é time, level, latitude e longitude, por isso o TLLL. Muito útil para arquivos que apresentam nível vertical.

Se mesmo assim, a ordem das dimensões do seu dado não estiver em alguma dessas possibilidades, utilize o símbolo "|" para reordenar as dimensões. Veja no capítulo visão geral (página 33) o símbolo "|" que ensina como reordenar as dimensões.

Supondo que o dado esteja com os meses incompletos, como no exemplo abaixo em que os meses começam em maio de 1979 e terminam em abril de 2012. A função para o cálculo de anomalia exige que os meses sejam múltiplos de 12. Logo, com esse dado seria impossível realizar esse cálculo, mas ao manipular a dimensão tempo para selecionar o período de interesse, esse problema é contornado.

```
; Nome do script: cap14_ex03.ncl

begin

;*************************************************************
; Abertura do arquivo. Lembrando que neste dado
; (maio1979 a abr2012), os meses do primeiro e do último
; ano não estão completos, por isso, foi selecionado um perído
; em que todos os meses estão completos, isto é, de janeiro
; de 1980 a dezembro de 2011. Poderia ser qualque período desde
; que os meses não estejam faltando.
;*************************************************************
f   = addfile("../../dados/gpcp.mensal.051979.042012.nc","r")
;*************************************************************

time   = f->time
YYYYMM = cd_calendar(time,-1) ; Converte uma data do calendário
                              ; juliano/gregoriano
                              ; para o formato AAAAMM (opção -1)
                              ; da variável time.
anoi   = 198001               ; Data inicial no formato AAAAMM.
anof   = 201112               ; Data final no formato AAAAMM.
istr   = ind(YYYYMM.eq.anoi ) ; Índice que será utilizado
iend   = ind(YYYYMM.eq.anof ) ; para selecionar o período de interesse.
ppt = f->precip(istr:iend,:,:); Escolha do período de interesse
                              ; com base nas datas (linhas 21 e 22).
;*************************************************************
; Calcula a climatologia no período de 198001 a 201112
;*************************************************************
clima = clmMonTLL(ppt)
;*************************************************************
; Calcula a anomalia mensal usando a climatologia
;*************************************************************
pptAnom = calcMonAnomTLL(ppt,clima)
;*************************************************************

printVarSummary(pptAnom)

end
```

14.7 Função calculate_daily_values

Função que calcula os valores diários (média, soma, mínimo e máximo) de valores temporais de alta frequência.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/calculate_daily_values.shtml

No exemplo abaixo o arquivo possui 360 tempos em intervalos de seis hora. A variável utilizada é a temperatura do ar em vários níveis verticais para os meses de janeiro, fevereiro e março de 2009.

```
1   ; Nome do script: cap14_ex04.ncl
2
3   begin
4
5   f = addfile("../../dados/010120090000UTCa310320091800UTC.nc","r")
6
7   t = short2flt(f->air)
8
9   ; Informações da variável t:
10  ; [time | 360] x [level | 12] x [lat | 73] x [lon | 144]
11  ; short air ( time, level, lat, lon )
12  ; Como o dado é do tipo short (visto com o ncl_filedump),
13  ; será necessário descompactá-lo para o tipo float
14  ; com a função short2flt.
15
16  ; Calcula a média diária. O valor "0" refere-se a dimensão para
17  ; realizar a estatística, ou seja, a dimensão tempo.
18
19  xdayAvg = calculate_daily_values(t,"avg",0,False)
20
21  printVarSummary(xdayAvg) ; Imprime um resumo da variável xdayAvg.
22
23  end
```

O resultado do printVarSummary(xdayAvg) será:

O que está em vermelho ([time |90]) representa a conversão dos dados horários em diários. Os meses utilizados foram janeiro (31 dias), fevereiro (28 dias) e março(31 dias) o que resulta em 90 tempos ou dias.

Variable: xdayAvg
Type: float
Total Size: 45411840 bytes
11352960 values
Number of Dimensions: 4
Dimensions and sizes: [time |90] x [level |12] x [lat |73] x [lon |144]
Coordinates:
time: [1832064..1834200]
level: [1000..100]
lat: [90..-90]
lon: [0..357.5]

Outra possibilidade de uso desta função é informando um número mínimo de valores

a serem utilizados no cálculo da estatística. No exemplo abaixo, foi utilizado o valor 4 (opt@nval_crit = 4) como o mínimo de valores para efetuar esse cálculo.

```
; Nome do script: cap14_ex05.ncl

begin

f = addfile("../../dados/010120090000UTCa310320091800UTC.nc","r")

t = short2flt(f->air)

; Informações da variável t:
; [time | 360] x [level | 12] x [lat | 73] x [lon | 144]
; short air ( time, level, lat, lon )
; Como o dado é do tipo short (visto com o ncl_filedump),
; será necessário descompactá-lo para o tipo float
; com a função short2flt.

opt           = True
opt@nval_crit = 4     ; Requer pelo menos 4 valores para
                      ; calcular a média.

; Calcula a média mensal. O valor "0" refere-se a dimensão para
; realizar a estatística, ou seja, a dimensão tempo.

xdayAvg = calculate_daily_values(t,"avg",0,opt)

printVarSummary(xdayAvg)

end
```

O resultado do printVarSummary(xdayAvg) é o mesmo do exemplo anterior.

14.8 Função calculate_monthly_values

Função que calcula valores mensais (média, soma, mínimo e máximo) de valores temporais de alta frequência. Esta função suporta até quatro dimensões.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/calculate_monthly_values.shtml

Exemplo: O arquivo abaixo é um dado a cada 6 horas de temperatura do ar em vários níveis verticais para os meses de janeiro a março de 2009. Os argumentos válidos para calcular a estatística são: avg ou ave, sum, min e max. No exemplo abaixo, a variável tem as seguintes dimensões: tempo, nível, latitude e longitude ou os índices 0 1 2 3, respectivamente. A estatística será realizada na dimensão tempo, por isso o valor "0" como um dos argumetos da função. Como resultado, será gerado uma variável com 3 tempos (janeiro, fevereiro e março) porque os dados eram a cada seis horas e foram convertidos para a resolução mensal.

```
; Nome do script: cap14_ex06.ncl

begin

f = addfile("../../dados/010120090000UTCa310320091800UTC.nc","r")

t = short2flt(f->air)

; Informações da variável t:
; [time | 360] x [level | 12] x [lat | 73] x [lon | 144]
; short air ( time, level, lat, lon )
; Como o dado é do tipo short (visto com o ncl_filedump),
; será necessário descompactá-lo para o tipo float
; com a função short2flt.

; Calcula a média mensal. O valor "0" refere-se a dimensão
; para realizar a estatística, ou seja, a dimensão tempo.
xmonAvg = calculate_monthly_values(t,"avg",0,False)

; Calcula a soma.
xmonSum = calculate_monthly_values(t,"sum",0,False)

; Calcula o mínimo.
xmonMin = calculate_monthly_values(t,"min",0,False)

; Calcula a máximo.
xmonMax = calculate_monthly_values(t,"max",0,False)

printVarSummary(xmonAvg)

end
```

O resultado do printVarSummary(xmonAvg) será:

O que está vermelho ([time |3]) são os tempos que foram convertidos para a resolução mensal. Como os dados são para os meses de janeiro, fevereiro e março, o time é igual a 3.

Variable: xmonAvg
Type: float
Total Size: 1513728 bytes
378432 values
Number of Dimensions: 4
Dimensions and sizes: [time |3] x [level |12] x [lat |73] x [lon |144]
Coordinates:
time: [1832064..1833480]
level: [1000..100]
lat: [90..-90]
lon: [0..357.5]

Outra possibilidade de uso desta função é informando o número mínimo de valores a serem utilizados no cálculo da estatística. No exemplo abaixo, foi utilizado o valor 4 (opt@nval_crit = 4) como o mínimo de valores para efetuar esse cálculo.

```
; Nome do script: cap14_ex07.ncl

begin

f = addfile("../../dados/010120090000UTCa310320091800UTC.nc","r")

t = short2flt(f->air)

; Informações da variável t:
; [time | 360] x [level | 12] x [lat | 73] x [lon | 144]
; short air ( time, level, lat, lon )
; Como o dado é do tipo short (visto com o ncl_filedump),
; será necessário descompactá-lo para o tipo float
; com a função short2flt.

opt           = True
opt@nval_crit = 4     ; Requer pelo menos 4 valores para
                      ;calcular a média.

; Calcula a média mensal. O valor "0" refere-se a dimensão
; para realizar a estatística, ou seja, a dimensão tempo.

xmonAvg = calculate_monthly_values(t,"avg",0,opt)

printVarSummary(xmonAvg)

end
```

O resultado do printVarSummary(xmonAvg) é o mesmo do exemplo anterior.

14.9 Função calculate_segment_values

Esta função calcula valores segmentados, como por exemplo, pêntadas (5 dias), semanais (7 dias) de valores em alta resolução temporal. Essa função suporta até 4 dimensões e funciona somente para calendário gregoriano e caso não haja nenhum calendário associado, será retornado erro. Os argumentos válidos para realizar as estatísticas são: avg ou ave, sum, min e max. O cálculo normalmente é feito na dimensão tempo.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/calculate_segment_values.shtml

Ao habilitar a opção opt para True é possível determinar o número mínimo de tempos (opt@nval_crit) a ser utilizado, como também o comprimento da estatística (opt@segment_length) que se deseja aplicar. Para obter valores semanais, o padrão é 7 e para pêntadas é 5.

Exemplo: O arquivo abaixo corresponde a 365 dias de precipitação para o ano de 2005. Por padrão, será utilizado o valor 7 para realizar a estatística.

```
; Nome do script: cap14_ex08.ncl

begin

f = addfile("../../dados/ppt.2005.nc","r")

ppt = f->PREC ; [time | 365] x [lat | 180] x [lon | 360]

; Calcula a média a cada 7 dias (padrão). O valor "0"
; refere-se a dimensão para realizar a estatística, ou seja,
; a dimensão tempo.
; Como os dados possuem 365 dias / 7 = 52 semanas.

xSegAvg = calculate_segment_values(ppt,"avg",0,False)
xSegSum = calculate_segment_values(ppt,"sum",0,False)
xSegMin = calculate_segment_values(ppt,"min",0,False)
xSegMax = calculate_segment_values(ppt,"max",0,False)

printVarSummary(xSegAvg) ; Imprime um resumo da variável xSegAvg.

end
```

O resultado do printVarSummary(xSegAvg) será:

O que está em vermelho é o resultado da estatística, isto é, 365 dias/7=52 semanas.

Variable: xSegAvg
Type: float
Total Size: 13478400 bytes
3369600 values
Number of Dimensions: 3
Dimensions and sizes: [time |52] x [lat |180] x [lon |360]

Ao invés de utilizar o valor padrão que é 7, será utilizado o valor 5, mas para isso deve-se habilitar a opção opt com as seguintes configurações: opt@nval_crit = 4 e opt@segment_length = 5 como no script abaixo.

```
; Nome do script: cap14_ex09.ncl

begin

f = addfile("../../dados/ppt.2000.nc","r")
ppt = f->PREC ; [time | 366] x [lat | 180] x [lon | 360]

opt                 = True
opt@nval_crit       = 4    ; Requer pelo menos 4 valores para realizar a
                           ; estatística.
opt@segment_length = 5    ; Média a cada 5 dias. O valor padrão é 1.

; Calcula a média a cada 5 dias. O valor "0" refere-se a dimensão para
; realizar a estatística, ou seja, a dimensão tempo.
; Como os dados possuem 366 dias / 5 = 73 valores de tempo.

xSegAvg = calculate_segment_values(ppt,"avg",0,opt)

printVarSummary(xSegAvg) ; imprime um resumo da variável xSegAvg.

end
```

O resultado do printVarSummary(xSegAvg) será 366 dias/5=73 valores de tempo.

Variable: xSegAvg
Type: float
Total Size: 18921600 bytes
4730400 values
Number of Dimensions: 3
Dimensions and sizes: [time |73] x [lat |180] x [lon |360]

14.10 Funcão ceil

Calcula o menor valor inteiro maior ou igual à entrada. Valores ausentes são ignorados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/ceil.shtml

Exemplo:

f = 6.4
ceil_f = ceil(f)
print(ceil_f) ; Resposta: 7

14.11 Função para climatologia

Neste item serão mostrados alguns exemplos para o cálculo de climatologia.

Atenção deve ser dada para a ordem das dimensões. Caso seja necessário realizar a mudança de ordem das dimensões, utilize o símbolo "|" (página 33).

14.11.1 Climatologia diária usando a função clmDayTLL

Para calcular a climatologia diária utiliza-se a função clmDayTLL em que o dado de entrada deve ser tridimensional, isto é, time, latitude e longitude, por isso o TLL. Será retornada uma série temporal onde a dimensão mais a esquerda refere-se ao dia sequencial do ano (366 dias).

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/clmDayTLL.shtml

Exemplo: Calculando a climatologia diária no período de 01/01/1979 a 31/12/2001.

```
; Nome do script: cap14_ex10.ncl

begin

f = addfile("../../dados/GPCP_1DD_v1.2_19970101-20013112.nc","r")

;*************************************************************
; Leitura do tempo e criação dele no formato yyyyddd.
; Será utilizado na função para calcular a climatologia.
;*************************************************************
time    = f->time
TIME    = cd_calendar(time, 0) ; Tipo float.
year    = toint(TIME(:,0))     ; Guarda o ano.
month   = toint(TIME(:,1))     ; Guarda o mês.
day     = toint(TIME(:,2))     ; Guarda o dia.
ddd     = day_of_year(year,month,day) ; Retorna o dia do ano no
                                      ; formato 365 ou 366 dias.
yyyyddd = year*1000 + ddd             ; É necessário como argumento
                                      ; para o cálculo da climatologia.
;*************************************************************
; Leitura do dado
;*************************************************************
ppt = f->PREC
;*************************************************************
; Calcula a climatologia
;*************************************************************
pptClmDay = clmDayTLL(ppt,yyyyddd) ; Calcula a climatologia. Será
                                   ; retornada uma variável com 366 dias
printVarSummary(pptClmDay)

end
```

O resultado do printVarSummary(pptClmDay) será:

O que está em vermelho ([year_day |366]) é o resultado do cálculo da climatologia diária.

```
Variable: pptClmDay
Type: float
Total Size: 6148800 bytes
            1537200 values
Number of Dimensions: 3
Dimensions and sizes:   [year_day | 366] x [lat | 70] x [lon | 60]
Coordinates:
            year_day: [1..366]
            lat: [9.5..-59.5]
            lon: [270.5..329.5]
Number Of Attributes: 4
  long_name :   Daily Climatology: GPCP: daily precipitation
  units :       mm/day
  information : Raw daily averages across all years
  smoothing :   None
```

14.11.2 Climatologia diária usando a função clmDayTLLL

O raciocínio é similar ao item anterior com a diferença que neste caso, o dado de entrada possui quatro dimensões, isto é, as dimensões devem estar na seguinte ordem: time, level, latitude e longitude, por isso o TLLL.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/clmDayTLLL.shtml

Exemplo: Calculando a climatologia diária no período de 01/01/1970 a 31/12/1983.

```
1   ; Nome do script: cap14_ex11.ncl
2
3   begin
4
5   f = addfile("../../dados/tar.01jan1970.31dez1983.nc","r")
6
7   ;************************************************************
8   ; Leitura do tempo e criação dele no formato yyyyddd.
9   ; Será utilizado na função para calcular a climatologia.
10  ;************************************************************
11  time    = f->time
12  TIME    = cd_calendar(time, 0) ; Tipo float.
13  year    = toint(TIME(:,0))     ; Guarda o ano.
14  month   = toint(TIME(:,1))     ; Guarda o mês.
15  day     = toint(TIME(:,2))     ; Guarda o dia.
16  ddd     = day_of_year(year,month,day) ; Retorna o dia do ano
17                                        ; no formato 365 ou 366 dias.
18  yyyyddd = year*1000 + ddd             ; É necessário como argumento
19                                        ; para o cálculo da climatologi
20  ;************************************************************
21  ; Leitura do dado.
22  ;************************************************************
23  t = short2flt(f->air) ; Descompacta o dado para o tipo float porque
24                        ; ele é do tipo short.
25  ;************************************************************
26  ; Calcula a climatologia
27  ;************************************************************
28  tClmDay = clmDayTLLL(t,yyyyddd) ; Climatologia. Será retornada uma
29                                  ; variável com 366 dias.
30  printVarSummary(tClmDay)
31
32  end
```

O resultado do printVarSummary(tClmDay) será:

```
Variable: tClmDay
Type: float
Total Size: 46168704 bytes
            11542176 values
Number of Dimensions: 4
Dimensions and sizes:   [year_day | 366] x [level | 3] x [lat | 73] x [lon | 144]
Coordinates:
            year_day: [1..366]
            level: [1000..850]
            lat: [90..-90]
            lon: [ 0..357.5]
Number Of Attributes: 4
  long_name :   Daily Climatology: Daily Air Temperature on Pressure Levels
  units :       degK
  information : Raw daily averages across all years
  smoothing :   None
```

14.11.3 Climatologia mensal usando a função clmMonLLLT

Calcula a climatologia mensal de dados mensais. A ordem das dimensões do dado de entrada são: level, latitude, longitude e time, por isso o LLLT. Caso as dimensões não estejam nessa ordem, utilize o símbolo "|" para reordenar as dimensões. O valor retornado será do mesmo tamanho e tipo do dado de entrada, com exceção de que a dimensão mais a direita será de tamanho 12 (meses) por se tratar da climatologia mensal.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/clmMonLLLT.shtml

Exemplo: Calculando a climatologia mensal no período de jan/1979 a dez/1988.

```
; Nome do script: cap14_ex12.ncl

begin

f = addfile("../../dados/tar.mensal.1979.1988.nc","r")

;*************************************************************
; Leitura do dado
;*************************************************************

t = short2flt(f->air)

; Descompacta o dado para o tipo float porque ele
; é short. Essa informação foi vista com o ncl_filedump.
; A variável apresenta a seguinte disposição das dimensões
; [time | 120] x [level | 3] x [lat | 73] x [lon | 144].
; Para usar a função de climatologia, as dimensões precisam
; estar na seguinte ordem:
; [level | 3] x [lat | 73] x [lon | 144] x [time | 120]
; em que a dimensão time tem que ser a última porque a
; função será aplicada nessa dimensão. Para isso, será
; necessário reordenar as dimensões.

; Com o uso do operador "|:" foi possível reordenar as dimensões.
; A nova ordem será:
; [level | 3] x [lat | 73] x [lon | 144] x [time | 120]

t_ord = t(level|:,lat|:,lon|:,time|:)

;*************************************************************
; Calcula a climatologia
;*************************************************************
tClm = clmMonLLLT(t_ord) ; Climatologia.
                         ; Será retornado uma variável
                         ; com 12 tempos (meses).

printVarSummary(tClm)

end
```

O que está em vermelho ([month |12]) é o resultado da função.

```
Variable: tClm
Type: float
Total Size: 1513728 bytes
            378432 values
Number of Dimensions: 4
Dimensions and sizes:   [level | 3] x [lat | 73] x [lon | 144] x [month | 12]
Coordinates:
            level: [1000..850]
            lat: [90..-90]
            lon: [ 0..357.5]
            month: [0..11]
Number Of Attributes: 22
  _FillValue :  -32767
  valid_range : ( 137.5, 362.55 )
  missing_value_original :      32766
  _FillValue_original : -32767
  cell_methods :        time: mean (monthly from 6-hourly values)
  standard_name :       air_temperature
  parent_stat : Other
  statistic :   Individual Obs
  level_desc :  Pressure Levels
  dataset :     NCEP/DOE AMIP-II Reanalysis (Reanalysis-2) Monthly Averages
  var_desc :    Air temperature
  GRIB_name :   TMP
  GRIB_id :     11
  least_significant_digit :     1
  precision :   2
  units :       degK
  actual_range :        ( 226.18, 315.12 )
  unpacked_valid_range :        ( 137.5, 362.5 )
  long_name :   Monthly Air Temperature on Pressure Levels
  missing_value :       -32767
  time_op_ncl : Climatology: 10 years
  info :        function clmMonLLLT: contributed.ncl
```

Uma vez que o dado apresenta as dimensões time, level, latitude e longitude nessa ordem, seria mais fácil usar uma função que tenha o seguinte padrão TLLL (time, level, latitude, longitude) em vez de reordenar as dimensões.

Outras possibilidades para o cálculo da climatologia mensal:

- Função clmMonTLL: as dimensões precisam estar nessa ordem: time, latitude, longitude.
- Função clmMonTLLL: as dimensões precisam estar nessa ordem: time, level, latitude, longitude.

14.12 Função conform

Expande uma matriz ou um escalar de forma que se adapte ao formato de uma dada variável. Para que duas variáveis sejam conformes elas precisam ter as mesmas dimensões.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/conform.shtml

Exemplo: A variável temperatura do ar possui as seguintes dimensões time, level, latitude e longitude e a variável pressão à superfície apenas time, latitude e longitude. Deseja-se que a pressão tenha as mesmas dimensões da temperatura. Para isso, utiliza-se a função conform. As únicas dimensões que aparecem nas duas variáveis são: time, latitude e longitude, ou seja, os índices 0, 2 e 3 que serão utilizados na função conform (linha 22 do script abaixo).

Tabela 4: Dimensões das variáveis.

Temperatura	time	level	latitude	longitude
Pressão	time	-	latitude	longitude
	0	1	2	3

```
1   ; Nome do script: cap14_ex13.ncl
2
3   begin
4
5   f1 = addfile("../../dados/psfc.mensal.1979.1988.nc","r")
6   f2 = addfile("../../dados/tar.mensal.1979.1988.nc","r")
7
8   psfc = short2flt(f1->pres)
9   tar  = short2flt(f2->air)
10
11  ; Variáveis do tipo short precisam ser descompactadas.
12  ; para float. Informação vista com o ncl_filedump.
13  ; short air ( time, level, lat, lon )
14  ; short pres ( time, lat, lon )
15
16  printVarSummary(psfc) ; [time | 120] x [lat | 73] x [lon | 144]
17  printVarSummary(tar)  ; [time | 120] x [level | 3] x [lat | 73] x [lon | 144]
18
19  ; 0 (time), 2 (lat) e 3 (lon) são as dimensões de "tar" que correspondem
20  ; a "psfc"
21
22  psfc_conform = conform(tar,psfc,(/0,2,3/))
23
24  printVarSummary(psfc_conform)
25
26  ; A variável "psfc_conform" não possui informações sobre as
27  ; coordenadas e nome das dimensões.
28  ; Dimensions and sizes: [120] x [3] x [73] x [144]
29
30  copy_VarCoords(tar,psfc_conform)
31  ; Copiando as coordenadas e as dimensões de "tar" para "psfc_conform"
32  ; [time | 120] x [level | 3] x [lat | 73] x [lon | 144]
33
34  printVarSummary(psfc_conform)
35  ; verificando se as coordenadas e as dimensões foram
36  ; copiadas para "psfc_conform"
37
38  end
```

Abaixo é mostrado o resultado de um pequeno trecho do printVarSummary(psfc) da linha 16. O dado foi descompactado para o formato float que antes era do tipo short. Essa variável apresenta as seguintes dimensões: time, lat e lon.

```
Variable: psfc
Type: float
Total Size: 5045760 bytes
            1261440 values
Number of Dimensions: 3
Dimensions and sizes:   [time | 120] x [lat | 73] x [lon | 144]
Coordinates:
            time: [1569072..1656000]
            lat: [90..-90]
            lon: [ 0..357.5]
```

Abaixo é mostrado o resultado de um pequeno trecho do printVarSummary(tar) da linha 17. O dado foi descompactado para o formato float que antes era do tipo short. Essa variável apresenta as seguintes dimensões: time, level, lat e lon.

```
Variable: tar
Type: float
Total Size: 15137280 bytes
            3784320 values
Number of Dimensions: 4
Dimensions and sizes:   [time | 120] x [level | 3] x [lat | 73] x [lon | 144]
Coordinates:
            time: [1569072..1656000]
            level: [1000..850]
            lat: [90..-90]
            lon: [ 0..357.5]
```

Ao realizar o uso da função conform, foram perdidas as informações das coordenadas e das dimensões. Como já foi mencionado, o NCL não será capaz de gerar corretamente a figura. Aqui, cabe o uso da função copy_VarCoords e apenas lembrando que ao copiar as informações de uma variável para outra, obrigatoriamente a variável fonte precisa ter o mesmo número de dimensões da variável destino que é o caso.

Resultado do printVarSummary(psfc_conform) da linha 24.

```
Variable: psfc_conform
Type: float
Total Size: 15137280 bytes
            3784320 values
Number of Dimensions: 4
Dimensions and sizes:   [120] x [3] x [73] x [144]
Coordinates:
Number Of Attributes: 1
  _FillValue :  32766
```

Após o uso da função copy_VarCoords a variável psfc_conform passa a ter os mesmos nomes das coordenadas e dimensões da variável tar.

Resultado do printVarSummary(psfc_conform) da linha 34.

```
Variable: psfc_conform
Type: float
Total Size: 15137280 bytes
            3784320 values
Number of Dimensions: 4
Dimensions and sizes:   [time | 120] x [level | 3] x [lat | 73] x [lon | 144]
Coordinates:
            time: [1569072..1656000]
            level: [1000..850]
            lat: [90..-90]
            lon: [ 0..357.5]
Number Of Attributes: 1
   FillValue :  32766
```

14.13 Função copy_VarCoords

Esta função copia todos os nomes das dimensões e coordenadas de uma variável para outra. Sem essas informações no arquivo será impossível gerar figuras. Pode-se criar manualmente essas informações, porém é um atividade mais trabalhosa do que copiar as informações de uma variável para outra.

Ao usar esta função é obrigatório que as duas variáveis possuam o mesmo número de dimensões e tamanhos. Por exemplo, o usuário está manipulando a variável vento que é 4D (tempo, nível, lat, lon) e deseja copiar as informações das coordenadas e das dimensões para a variável precipitação que é 3D (tempo, lat, lon). Isso vai gerar erro porque o número de dimensões não é igual.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/copy_VarCoords.shtml

Exemplo:

```
1   ; Nome do script: cap14_ex14.ncl
2
3   begin
4
5   f = addfile("../../dados/prec.1986.1989.nc","r")
6
7   ppt = f->precip ; [time | 48] x [lat | 28] x [lon | 24]
8
9   printVarSummary(ppt)
10
11  pptmes = ppt * 30 ; Conversão de mm/dia para mm/mês.
12                    ; Ao multiplicar por 30 as informações
13                    ; de dimensões e coordenadas foram perdidas.
14                    ; Basta usar a função copy_VarCoords para
15                    ; copiar o nome das dimensões e coordenadas
16                    ; da variável "ppt" para "pptmes".
17                    ; Sem as informações das dimensões e das
18                    ; coordenadas será impossível gerar a figura.
19
20  printVarSummary(pptmes) ; Observe que não há o nome das
21                          ; dimensões e nem das coordenadas.
22
```

```
23   copy_VarCoords(ppt,pptmes) ; Copia as dimensões e coordenadas
24                              ; da variável ppt para pptmes.
25
26   printVarSummary(pptmes) ; Note que a variável pptmes agora
27                           ; apresenta as informações das
28                           ; dimensões e das coordenadas.
29   end
```

Resultado do primeiro printVarSummary(ppt) da linha 9:

Note a linha Dimensions and sizes que tem o nome das dimensões time, lat e lon. O NCL precisa desses nomes para gerar a figura sem erros.

```
Variable: ppt
Type: float
Total Size: 129024 bytes
            32256 values
Number of Dimensions: 3
Dimensions and sizes:   [time | 48] x [lat | 28] x [lon | 24]
Coordinates:
            time: [67935..69365]
            lat: [8.75..-58.75]
            lon: [271.25..328.75]
```

Resultado do segundo printVarSummary(pptmes) da linha 20:

Ao multiplicar por 30 houve a perda das informações das dimensões. Nota-se que na linha Dimensions and sizes não há o nome das dimensões, como foi visto nas informações acima. Sem isso, o NCL não será capaz de gerar corretamente a figura.

```
Variable: pptmes
Type: float
Total Size: 129024 bytes
            32256 values
Number of Dimensions: 3
Dimensions and sizes:   [48] x [28] x [24]
Coordinates:
Number Of Attributes: 1
  _FillValue :  -9.96921e+36
```

Ao usar a função copy_VarCoords(ppt,pptmes) ocorre a copia das informações das coordenadas e dimensões de ppt para pptmes.

O resultado do printVarSummary(pptmes) da linha 26 após o uso da função mostra que a variável pptmes apresenta as informações de coordenadas e de dimensões.

```
Variable: pptmes
Type: float
Total Size: 129024 bytes
            32256 values
Number of Dimensions: 3
Dimensions and sizes:   [time | 48] x [lat | 28] x [lon | 24]
Coordinates:
            time: [67935..69365]
            lat: [8.75..-58.75]
            lon: [271.25..328.75]
```

Há outras rotinas para manipulação de metadados:

http://www.ncl.ucar.edu/Document/Functions/metadata.shtml

- Função copy_VarCoords_1: copia todos os nomes das dimensões e das coordenadas de uma variável para outra, com exceção da dimensão mais a direita.
- Função copy_VarCoords_2: copia todos os nomes das dimensões e das coordenadas de uma variável para outra, com exceção das duas dimensões mais a direita.

Informações sobre o nome das dimensões e variáveis coordenadas podem ser obtidas nos links abaixo:

http://www.ncl.ucar.edu/Document/Language/named.shtml

http://www.ncl.ucar.edu/Document/Language/cv.shtml

14.14 Função day_of_year

Calcula o dia do ano dado o calendário Gregoriano mês, dia e ano. Não confundir com o dia Juliano. Os argumentos necessários são: ano (tem que ser positivo), mês (valores entre 1 e 12) e dia (valores entre 1 e 31). Os argumentos podem ser um arranjo multidimensional ou um escalar.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/day_of_year.shtml

Exemplo1:

doy = day_of_year((/1900,1990,1996/),(/3,3,3/),(/1,1,1/)) ; doy = (/60,60,61/)

Exemplo2: Este exemplo é similar ao Exemplo1 só que utiliza diferentes calendários.

year = (/1900,1990,1996/)

month = (/3,3,3/)

day = (/1,1,1/)

; Sem calendário

doy = day_of_year(year,month,day) ; doy = (/60,60,61/)

year@calendar = "standard"
doy = day_of_year(year,month,day) ; doy = (/60,60,61/)

year@calendar = "julian"
doy = day_of_year(year,month,day) ; doy = (/61,60,61/)

year@calendar = "noleap" ; inclui também "365"
doy = day_of_year(year,month,day) ; doy = (/61,60,60/)

year@calendar = "allleap" ; inclui também "366"
doy = day_of_year(year,month,day) ; doy = (/61,61,61/)

year@calendar = "360"
doy = day_of_year(year,month,day) ; doy = (/61,61,61/)

14.15 Função days_in_month

Calcula o número de dias no mês dado o mês e o ano. Os argumentos ano e mês são arranjos multidimensionais do tipo inteiro em que o ano deve ser um valor positivo e o mês deve ser um valor entre 1 e 12.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/days_in_month.shtml

Exemplo1:

dim = days_in_month((/1996,1997,1/),(/2,2,1/)) ; dim = (/29,28,31/)

Examplo2: Similar ao Exemplo1 só que usando diferentes tipos de calendários.

year = (/1996,1997,1/)
month = (/2,2,1/)

year@calendar = "standard"
dim = days_in_month(year,month) ; dim = (/29,28,31/)

year@calendar = "noleap"
dim = days_in_month(year,month) ; dim = (/28,28,31/)

year@calendar = "allleap"
dim = days_in_month(year,month) ; dim = (/29,29,31/)

year@calendar = "360_day"
dim = days_in_month(year,month) ; dim = (/30,30,30/)

14.16 Função dewtemp_trh

Calcula a temperatura do ponto de orvalho dado a temperatura em Kelvin e a umidade relativa do ar em porcentagem. As duas variáveis devem ter o mesmo número de dimensões. Os valores ausente são ignorado. O cálculo é feito com base na equação de John Dutton's "Ceaseless Wind" (página 273-274).

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/dewtemp_trh.shtml

Exemplo:

```
; Nome do script: cap14_ex15.ncl

begin

f1 = addfile("../../dados/ur.mensal.1979.1988.nc","r")
f2 = addfile("../../dados/tar.mensal.1979.1988.nc","r")

; Variáveis do tipo short precisam ser descompactadas
; para float. Informação vista com o ncl_filedump.

ur = short2flt(f1->rhum) ; short rhum ( time, level, lat, lon )
tk = short2flt(f2->air)  ; short air ( time, level, lat, lon )

td = dewtemp_trh(tk,ur) ; As informações das coordenadas e dimensões
                        ; de "td" foram perdidas. Basta usar o
                        ; copy_VarCoords para resolver esse problema.

printVarSummary(td)
; Dimensions and sizes: [120] x [3] x [73] x [144]

copy_VarCoords(ur,td)   ; Poderia usar no lugar de "ur" o "tk" porque o
                        ; número de dimensões é o mesmo.

; Dimensions and sizes: [time | 120] x [level | 3] x [lat | 73] x [lon | 144]

printVarSummary(td)

end
```

14.17 Função dim_acumrun_n

Calcula a soma acumuluda de um comprimento específico.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/dim_acumrun_n.shtml

Exemplo1: O "x" na função é o arranjo numérico de qualquer dimensionalidade, o valor 3 é o comprimento da soma, isto é, a soma será feita a cada 3 valores. O valor 0 (em vermelho) quer dizer que se encontrar algum valor indefinido ele será utilizado como indefinido no comprimento utilizado e será retornado o valor indefinido. Quando o valor for 1 (em vermelho) o cálculo continuará a ser feito com os valores disponíveis. Quando não ocorrer nenhum valor indefinido a opção 0 e 1 tem o mesmo efeito. O último argumeto 0 é a dimensão que se deseja realizar a soma.

x = (/1,2,3,4,5,7,-23/)
x@_FillValue = -999
x_ars = dim_acumrun_n(x,3,0,0) ; (/-999,-999,6,9,12,16,-11/)
x_ars = dim_acumrun_n(x,3,1,0) ; (/-999,-999,6,9,12,16,-11/)

Explicação: São somados os valores 1+2+3=6, 2+3+4=9, 3+4+5=12, 4+5+7=16 e 5+7-23=-11

Exemplo2:

x = (/1,2,-999,4,5,7,-999, 2, -9, 5/)
x@_FillValue = -999
x_ars = dim_acumrun_n(x,3,0,0) ; (/-999,-999,-999,-999,-999,16,-999,-999,-999,-2/)
x_ars = dim_acumrun_n(x,3,1,0) ; (/-999,-999,3,6,9,16,12,9,-7,-2/)

Explicação: No primeiro caso, a soma será 1+2+(-999)=-999 por causa da opção 0 (em vermelho). Quando esta opção é selecionada, basta aparecer um valor indefinido no cálculo e o resultado será indefinido. Continuando com os demais cálculos, 2+(-999)+4=-999, (-999)+4+5=-999, 4+5+7=16, 5+7+(-999)=-999, 7+(-999)+2=-999, (-999)+2+(-9)=-999, 2+(-9)+5=-2. O resultado será: -999,-999,-999,16,-999,-999,-999,-2.

No segundo caso, a opção 1 (em vermelho) foi selecionada, isto quer dizer que ao aparecer o valor indefinido o cálculo continuará como se não houvesse esse valor, 1+2+(-999)=3, 2+(-999)+4=6, (-999)+4+5=9, 4+5+7=16, 5+7+(-999)=12, 7+(-999)+2=9, (-999)+2+(-9)=-7, 2+(-9)+5=-2. Perceba a diferença quando se usa a opção 0 e 1 nesses dois casos. O resultado será: 3,6,9,16,12,9,-7,-2.

14.18 Função dim_avg_n_Wrap

Calcula a média de uma determinada dimensão de uma variável e retém os metadados (Wrap). O parâmetro de entrada é uma variável numérica de qualquer dimensionalidade. Outro parâmetro a ser utilizado é o índice da dimensão que se deseja realizar a média. Os valores ausentes são ignorados no cálculo.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/dim_avg_n_Wrap.shtml

No exemplo abaixo, será feito o cálculo zonal, isto é, na longitude, e como resultado será retornada uma variável apenas com duas dimensões (time e lat).

```
1   ; Nome do script: cap14_ex16.ncl
2
3   begin
4
5   f = addfile("../../dados/prec.1986.1989.nc","r")
6   ppt = f->precip ; float precip ( time, lat, lon ) ou em termos de
7                   ; índices das dimensões 0 1 2.
8
9   ; Calculando média zonal da precipitação. O cálculo será feito na
10  ; dimensão longitude ou no índice 2.
11
12  ppt_zonal = dim_avg_n_Wrap(ppt,2)
13  ; Dimensions and sizes: [time | 48] x [lat | 28]
14
15  ; As informações das coordenadas e dimensões não foram perdidas por
16  ; causa do "_Wrap" na função que as reteve.
17
18  printVarSummary(ppt_zonal)
19
20  end
```

O resultado do printVarSummary(ppt_zonal) da linha 18 terá apenas duas dimensões ([time |48] x [lat |28]), uma vez que, a média foi feita na dimensão longitude.

```
Variable: ppt_zonal
Type: float
Total Size: 5376 bytes
            1344 values
Number of Dimensions: 2
Dimensions and sizes:   [time | 48] x [lat | 28]
Coordinates:
            time: [67935..69365]
            lat: [8.75..-58.75]
```

Caso a média fosse feita na dimensão tempo, basta substituir na linha 12, o valor 2 por 0 que corresponde a dimensão tempo.

O comando seria: ppt_tempo = dim_avg_n_Wrap(ppt,0)

É sempre aconselhável selecionar as funções que possuem "_Wrap" para evitar a criação de coordenadas e dimensões de forma manual porque elas armazenam as informações dos metadados.

14.19 Funções dim_min_n_Wrap e dim_max_n_Wrap

Calcula o mínimo e o máximo de determinada dimensão de uma variável e retém os metadados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/dim_min_n_Wrap.shtml

http://www.ncl.ucar.edu/Document/Functions/Contributed/dim_max_n_Wrap.shtml

Exemplo:

```
; Nome do script: cap14_ex17.ncl

begin

f = addfile("../../dados/prec.1986.1989.nc","r")

ppt = f->precip ; float precip ( time, lat, lon ) ou em termos de
                ; índices das dimensões 0 1 2.

; Calculando o mínimo e máximo valor da precipitação. O cálculo
; será feito na dimensão tempo ou no índice 0.

ppt_min = dim_min_n_Wrap(ppt,0)

ppt_max = dim_max_n_Wrap(ppt,0)

; Como o cálculo foi feito na dimensão tempo, restam apenas
; duas dimensões (lat e lon).

printVarSummary(ppt_min) ; Dimensions and sizes: [lat | 28] x [lon | 24]
printVarSummary(ppt_max) ; Dimensions and sizes: [lat | 28] x [lon | 24]

end
```

14.20 Função dim_rmsd_n_Wrap

Calcula a diferença da raiz quadrada média entre as dimensões de duas variáveis em todas as outras dimensões. Os metados são retidos. Os valores ausentes são ignorados no cálculo.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/dim_rmsd_n_Wrap.shtml

Exemplo:

```
; Nome do script: cap14_ex18.ncl

begin

f1 = addfile("../../dados/gpcp.nc","r")
f2 = addfile("../../dados/cmip5.CCSM4.nc","r")

ppt_obs = f1->pc1 ; float pc1(time, lat, lon).
ppt_mod = f2->pc1 ; float pc1(time, lat, lon).
                  ; Índices das dimensões: 0 1 2.
                  ; Os dois arquivos são do tipo
                  ; [time | 27] x [lat | 1] x [lon | 1] que
                  ; representa a série temporal de precipitação
                  ; observada e simulada.

; Calculando a diferença da raiz quadrada média da precipitação.
; O cálculo será feito na dimensão tempo ou no índice 0.

rmsdTime = dim_rmsd_n_Wrap(ppt_obs,ppt_mod,0)

; Como o cálculo foi feito na dimensão tempo, restam apenas
; duas dimensões (lat e lon). Como foi usado a função que tem
; "_Wrap", não há necessidade de criar as dimensões e coordenadas

printVarSummary(rmsdTime) ; Dimensions and sizes: [lat | 1] x [lon | 1]
                          ; Tem apenas um ponto de lat/lon porque
                          ; é uma série temporal

print("RMSD = " + rmsdTime) ; RMSD = 1.73723

end
```

14.21 Função dim_rmvmean_n_Wrap

Calcula e remove a média de uma dada dimensão e retém os metadados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/dim_rmvmean_n_Wrap.shtml

Exemplo:

```
; Nome do script: cap14_ex19.ncl

begin

f = addfile("../../dados/olr.jan2000.dez2009.nc","r")

olr = short2flt(f->olr) ; [time | 120] x [lat | 73] x [lon | 144]
                        ; short olr ( time, lat, lon )

olr_desv = dim_rmvmean_n_Wrap(olr,0) ; Calcula o desvio em relação
                                     ; a média para a dimensão
                                     ; tempo (0) e retém as coordenadas
                                     ; e dimensões.

; Dimensions and sizes: [time | 120] x [lat | 73] x [lon | 144]

printVarSummary(olr_desv)

end
```

14.22 Função dim_spi_n

Calcula o índice de precipitação padronizado (standardized precipitation index, SPI) ajustando uma distribuição gama ou uma distribuição Pearson tipo III para valores mensais de precipitação. O tamanho da variável deve ser divisível por 12 (os meses devem ser completos e sem falhas). É recomendável pelo menos 30 anos de dados mensais de precipitação.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/dim_spi_n.shtml

Os valores mensais mais comuns para o cálculo do SPI são 3, 6, 12, 24 e 36. Ao usar o valor 3 para o SPI, os dois primeiros tempos são indefinidos. Para 6, os 5 primeiros tempos são indefinidos, para 12, os 11 primeiros tempos são indefinidos. O mesmo raciocínio se aplica aos valores 24 e 36.

Exemplo:

```
; Nome do script: cap14_ex20.ncl

begin

f = addfile("../../dados/gpcp.mensal.1979.2012.nc","r")

ppt = f->precip ; float precip ( time, lat, lon )
                ; [time | 408] x [lat | 72] x [lon | 144]
                ; Dado mensal de precipitação.

spi = dim_spi_n(ppt,12,False,0) ; São necessários 4 argumentos:
                                ; 1) a variável precipitação (ppt)
                                ; 2) o valor para calcular o SPI (12).
                                ; Lembrando que neste caso os 11
                                ; primeiros tempos serão indefinidos.
                                ; O False usa a distribuição gamma
                                ; e "0" define qual a dimensão será
                                ; aplicada a função, neste caso, será a
                                ; dimensão tempo (0).
```

```
20
21  printVarSummary(spi) ; Ao calcular o spi as informações das
22                       ; coordenadas e dimensões foram perdidas.
23                       ; Dimensions and sizes: [408] x [72] x [144]
24
25  copy_VarCoords(ppt,spi) ; Por isso, é necessário copiar as
26                          ; coordenadas e dimensões de "ppt" para "spi",
27                          ; uma vez que, possuem o mesmo número de dimensões
28
29  printVarSummary(spi)
30  ; O spi possui as coordenadas e dimensões corretas.
31  ; Dimensions and sizes: [time | 408] x [lat | 72] x [lon | 144]
32
33  end
```

Outra opção para calcular o spi é usar a distribuição de Pearson III (opt@spi_type = 3) como é mostrado na linha 12 do script abaixo. Lembrando, como foi usado o valor 12, os 11 primeiros tempos serão indefinidos.

```
1   ; Nome do script: cap14_ex21.ncl
2
3   begin
4
5   f = addfile("../../dados/gpcp.mensal.1979.2012.nc","r")
6
7   ppt = f->precip ; float precip ( time, lat, lon )
8                   ; [time | 408] x [lat | 72] x [lon | 144]
9                   ; Dado mensal de precipitação.
10
11  opt          = True
12  opt@spi_type = 3 ; Calcula o spi usando a distribuição de Pearson III.
13
14  spi = dim_spi_n(ppt,12,opt,0)   ; São necessários 4 argumentos:
15                                  ; 1) a variável precipitação (ppt).
16                                  ; 2) o valor para calcular o SPI (12).
17                                  ; Lembrando que neste caso os 11 primeiros
18                                  ; tempos serão indefinidos.
19                                  ; O opt usa a distribuição de Pearson III.
20                                  ; E "0" define qual a dimensão será
21                                  ; aplicada a função, neste caso, será a
22                                  ; dimensão tempo (0).
23
24  printVarSummary(spi) ; Ao calcular o spi as informações das coordenadas
25                       ; e dimensões foram perdidas.
26                       ; Dimensions and sizes: [408] x [72] x [144]
27
28  copy_VarCoords(ppt,spi) ; Por isso, é necessário copiar as coordenadas e
29                          ; dimensões de "ppt" para "spi"
30
31  printVarSummary(spi)
32  ; O spi possui as coordenadas e dimensões corretas
33  ; Dimensions and sizes: [time | 408] x [lat | 72] x [lon | 144]
34
35  end
```

14.23 Função dim_stddev_n_Wrap

Calcula o desvio padrão em uma dada dimensão.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/dim_stddev_n_Wrap.shtml

```
; Nome do script: cap14_ex22.ncl

begin

f = addfile("../../dados/gpcp.mensal.1979.2012.nc","r")

ppt = f->precip ; float precip ( time, lat, lon )
                ; [time | 408] x [lat | 72] x [lon | 144]
                ; Dado mensal de precipitação.

xStdTime = dim_stddev_n_Wrap(ppt,0) ; Calcula o desvio padrão
                                    ; na dimensão tempo (0).

; Dimensions and sizes: [lat | 72] x [lon | 144]
printVarSummary(xStdTime)

end
```

14.24 Função dim_sum_n_Wrap

Calcula a soma aritmética em uma dada dimensão.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/dim_stddev_n_Wrap.shtml

```
; Nome do script: cap14_ex23.ncl

begin

f = addfile("../../dados/gpcp.mensal.1979.2012.nc","r")

ppt = f->precip ; float precip ( time, lat, lon )
                ; [time | 408] x [lat | 72] x [lon | 144]
                ; Dado mensal de precipitação.

xSumTime = dim_sum_n_Wrap(ppt,0) ; Calcula a soma
                                 ; na dimensão tempo (0).

; Dimensions and sizes: [lat | 72] x [lon | 144]
printVarSummary(xSumTime)

end
```

14.25 Função dimsizes

Retorna o tamanho das dimensões da variável.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/dimsizes.shtml

```
; Nome do script: cap14_ex24.ncl

begin

f = addfile("../../dados/gpcp.mensal.1979.2012.nc","r")

ppt = f->precip ; float precip ( time, lat, lon )
                ; [time | 408] x [lat | 72] x [lon | 144]
                ; Dado mensal de precipitação.

info_ppt = dimsizes(ppt) ; Informações sobre o tamanho das
                         ; dimensões da variável info_ppt.

print(info_ppt) ; Serão retornadas as seguintes informações:
                ; (0) 408 => número de tempos.
                ; (1) 72  => número de pontos de lat.
                ; (2) 144 => número de pontos de lon.
end
```

14.26 Função fabs

Calcula o valor absoluto. Valores ausentes são ignorados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/fabs.shtml

Exemplo:

iarray = (/-3.5, -2.2, -1.0, 0.0, 1.2, 2.4, 3.9/)
jarray = fabs(iarray)
print(jarray) ; Resposta: (/3.5, 2.2, 1, 0, 1.2, 2.4, 3.9/)

14.27 Função fileexists

Verifica a existência do arquivo no diretório.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/fileexists.shtml

```
; Nome do script: cap14_ex25.ncl

begin

dir      = "../../dados/" ; Diretório corrente, onde está o dado.
filename = "gpcp.nc"      ; Nome do arquivo.

if ( fileexists ( dir+filename ) ) then
    print("O ARQUIVO ==> " + filename + " <== EXISTE")
else
    print("O ARQUIVO ==> " + filename + " <== NAO EXISTE")
end if

end
```

O resultado será:

(0) O ARQUIVO ==> gpcp.nc <== EXISTE

14.28 Função fspan

Cria uma matriz de números de ponto flutuante uniformemente espaçados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/fspan.shtml

Exemplo1:

x = fspan(0, 100, 11) ; x = (/0., 10., ... , 90., 100./) (11 valores)

Exemplo2:

mlon = 128
dlon = 360. / mlon
lon = fspan (0, (mlon - 1) * dlon, mlon) ; lon = (/0, 2.8125, ... , 357.1875/)

14.29 Funcão ind

Retorna o índice onde o valor de entrada é True.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/ind.shtml

Exemplo:

```
; Nome do script: cap14_ex26.ncl

begin

; Exemplo1:

; Define um vetor qualquer.
a            = (/1,2,3,4,5,5,4,3,2,1,1,2,3,4,5/)
a@_FillValue = 5 ; Define o valor indefinido igual a 5.

; Retorna os índices onde os valores são indefinidos.
b = ( ind( ismissing(a) ) )

print(b) ; Motra na tela os índices.
         ; Resposta: 4, 5 e 14.
         ; Lembrando que no NCL o índice
         ; inicia em zero e não em 1.

; Exemplo2:

c = ind( a .eq. 4 ) ; Qual é o índice no vetor "a" que
                    ; é igual a 4?

print(c) ; Resposta: 3, 6 e 13.

end
```

14.30 Função ind_nearest_coord

Determina os índices de locais mais próximos para uma matriz de coordenadas.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/ind_nearest_coord.shtml

```
; Nome do script: cap14_ex27.ncl

begin

f = addfile("../../dados/tar.mensal.1979.1988.nc","r")

lat = f->lat ; float lat(lat). Variável que está no arquivo.
lon = f->lon ; float lon(lon). Variável que está no arquivo.
             ; A variável "lon" esta no formato 0-360.
             ; O nome foi visto com o ncl_filedump.

rlat = (/-1.5  , 45.0/)  ; Qual a lat/lon que está mais próxima
rlon = (/307.0 , 271.0/) ; destes pontos ( (-1.5,307) e (45,271) )?

i_rlat = ind_nearest_coord(rlat,lat,0) ; Calcula o índice da latitude
                                       ; mais próximo dos pontos
                                       ; -1.5 e 45.0.

i_rlon = ind_nearest_coord(rlon,lon,0) ; Calcula o índice da longitude
                                       ; mais próximo dos pontos
                                       ; 300 e 270.

print("rlat = " + rlat + " i_rlat = " + i_rlat + \
      " lat = " + lat(i_rlat) + " lon = " + lon(i_rlon))

; Resposta:
; (0)   rlat = -1.5 i_rlat = 37 lat = -2.5 lon = 307.5
; (1)   rlat = 45 i_rlat = 18 lat = 45 lon = 270

end
```

14.31 Funções lógicas de checagem

14.31.1 Função iscoord

Retorna True se o nome da coordenada de uma determinada variável está contida em uma variável.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/iscoord.shtml

```
; Nome do script: cap14_ex28.ncl

begin

f = addfile("../../dados/tar.mensal.1979.1988.nc","r")

t = f->air ; O nome das dimensões da variável
           ; "air" são: time, level, lat, lon.

; O "t" é a variável que foi importada do arquivo "f".
; A função checará se "latitude" é o nome de uma coordenada
; da variável "t". Caso seja verdade, retornará como
; resultado o valor True, caso contrário, False. O mesmo raciocínio
; é aplicado para as demais dimensões.

check = iscoord(t,(/"latitude","lon","time","nivel"/))

print(check) ; A resposta será:
             ; (0) False
             ; (1) True
             ; (2) True
             ; (3) False
end
```

14.31.2 Função isdim

Retorna True se os nomes das dimensões pertencem a variável.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/isdim.shtml

Exemplo:

```
; Nome do script: cap14_ex29.ncl

begin

f = addfile("../../dados/tar.mensal.1979.1988.nc","r")

t = f->air ; O nome das dimensões da variável
           ; "air" são time, level, lat, lon

; O "t" é a variável que foi importada do arquivo "f".
; A função checará se "latitude" é o nome de uma dimensão
; da variável "t". Caso seja verdade, retornará como
; resultado True, caso contrário, False. O mesmo raciocínio
; é aplicado para as demais dimensões.

check = isdim(t,(/"latitude","lon","time","nivel"/))

print(check) ; A resposta será:
             ; (0) False
             ; (1) True
             ; (2) True
             ; (3) False
end
```

14.31.3 Função isfloat

Retorna True se a variável é do tipo float.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/isfloat.shtml

Exemplo:

```
; Nome do script: cap14_ex30.ncl

begin

f = addfile("../../dados/tar.mensal.1979.1988.nc","r")

t = f->air ; O nome das dimensões da variável
           ; "air" são time, level, lat e lon

; O "t" é a variável que foi importada do arquivo "f".
; A função checará se a variável "t" é do tipo float.
; Caso seja verdade, retornará como resultado True,
; caso contrário, False.

check = isfloat(t)

print(check) ; A resposta será:
             ; (0) False
end
```

Todas essas funções podem ser acessadas no link abaixo:

http://www.ncl.ucar.edu/Document/Functions/list_alpha.shtml#I

Algumas possibilidades que podem ser testadas:

- Função isinteger: Retorna True se a variável for do tipo inteiro.
- Função isleapyear: Determina se um dado ano é bissexto.
- Função isnumeric: Retorna True se a variável é do tipo numérica.
- Função ispan: Cria uma matriz de valores inteiros igualmente espaçados.
- Função isshort: Retorna True se a variável for do tipo short.
- Função isstring: Retorna True se a variável for do tipo string.

14.32 Função latGlobeF

Cria valores de latitude e metadados associados para uma grade global fixa.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/latGlobeF.shtml

Exemplo: Cria um vetor de latitudes variando do Polo Sul para o Polo Norte. Com o exemplo abaixo serão criados "73" pontos de latitude, a dimensão "lat", long_name (atributo) "latitude" e a unidade "degrees_north".

```
nlat = 73
lat = latGlobeF(nlat,"lat","latitude","degrees_north")
print(lat)
```

Resultado:

```
Variable: lat
Type: float
Total Size: 292 bytes
            73 values
Number of Dimensions: 1
Dimensions and sizes: [lat |73]
Coordinates:
            lat: [-90..90]
Number Of Attributes: 2
      long_name : latitude
      units : degrees_north
(0) -90
(1) -87.5
(2) -85
(3) -82.5
(4) -80
.
.
.
(68) 80
(69) 82.5
(70) 85
(71) 87.5
(72) 90
```

Nota1: Caso seja necessário inverter os valores de latitude, isto é, de 90°N para 90°S, utilize o subescrito padrão (::-1) para fazer essa inversão.

```
lat = lat(::-1) ; 90 to -90
```

Nota2: Caso seja necessário transformar a dimensão lat para dupla precisão, basta fazer:

```
nlat@double = True
```

14.33 Função latGlobeFo

Gera os valores de latitude e metadados associados para uma grade global fixa deslocada. A diferença em relação a função latGlobeF é que neste caso, os valores de latitude são deslocados em relação ao polo. Enquanto que na função latGlobeF os valores de latitude são considerados até ±90, na função latGlobeFo os valores vão até ±88.75.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/latGlobeFo.shtml

Exemplo: Cria um vetor de latitudes deslocado em relação aos polos. Com o exemplo abaixo serão criados "73" pontos de latitude, a dimensão "lat", long_name (atributo) "latitude" e a unidade "degrees_north".

```
nlat = 73
lat = latGlobeFo(nlat,"lat","latitude","degrees_north")
print(lat)
```

Resultado:

```
Variable: lat
Type: float
Total Size: 288 bytes
            72 values
Number of Dimensions: 1
Dimensions and sizes: [lat |72]
Coordinates:
            lat: [-88.75..88.75]
Number Of Attributes: 2
      long_name : latitude
      units : degrees_north
(0) -88.75
(1) -86.25
(2) -83.75
(3) -81.25
(4) -78.75
.
.
.
(67) 78.75
(68) 81.25
(69) 83.75
(70) 86.25
(71) 88.75
```

Nota1: Caso seja necessário inverter os valores de latitude, isto é, de 90^oN to 90^oS, utilize o subescrito padrão (::-1) para fazer essa inversão.

```
lat = lat(::-1) ; 90 to -90
```

Nota2: Caso seja necessário transformar a dimensão para dupla precisão, basta fazer:

```
nlat@double = True
```

14.34 Função lonFlip

Reordenar uma matriz de coordenada longitude (somente para grade rectilinear). A dimensão mais a direita deve ser a longitude em um domínio global, se for regional, retornará erro. O tamanho da dimensão de longitude deve ser um número par. Essa função simplesmente converte as coordenadas de longitude para o formato -180° a +180°.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/lonFlip.shtml

Exemplo: A variável psfc (pressão à superfície) apresenta os valores de longitude de 0 a 357.5. Para reordenar esses valores para o formato -180° a +180° será utilizada a função lonFlip. Lembrando que essa função somente deve ser usada em um dado global e para uma grade rectilinear.

Variable: psfc
Type: short
Total Size: 2522880 bytes
1261440 values
Number of Dimensions: 3
Dimensions and sizes: [time |120] x [lat |73] x [lon |144]
Coordinates:
time: [1569072..1656000]
lat: [90..-90]
lon: [0..357.5]

Basta utilizar o comando abaixo:

psfc = lonFlip(psfc) ; atualiza a variável psfc

ou

psfc_R = lonFlip(psfc) ; cria uma nova variável

A coordenda da nova variável psfc_R foi alterada (em vermelho) para o formato -180 a 177.5. O resultado abaixo foi visto com o printVarSummary(psfc_R).

Variable: psfc_R
Type: short
Total Size: 2522880 bytes
1261440 values
Number of Dimensions: 3
Dimensions and sizes: [time |120] x [lat |73] x [lon |144]
Coordinates:
time: [1569072..1656000]
lat: [90..-90]
lon: [-180..177.5]

14.35 Função lonGlobeF

Gera os valores de longitude e metadados associados para uma grade global fixa.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/lonGlobeF.shtml

Exemplo: Cria um vetor de longitudes iniciando no Meridiano de Greenwich. Com o exemplo abaixo serão criados 144 pontos de longitude, a dimensão lon, long_name longitude e a unidade degrees_east.

```
mlon = 144
lon = lonGlobeF(mlon,"lon","longitude","degrees_east")
print(lon)
```

Resultado:

```
Variable: lon
Type: float
Total Size: 576 bytes
            144 values
Number of Dimensions: 1
Dimensions and sizes: [lon |144]
Coordinates:
            lon: [ 0..357.5]
Number Of Attributes: 2
      long_name : longitude
      units : degrees_east
(0) 0
(1) 2.5
(2) 5
(3) 7.5
(4) 10
.
.
.
(140) 350
(141) 352.5
(142) 355
(143) 357.5
```

Nota1: Caso seja necessário transformar a dimensão para dupla precisão, basta fazer:

```
mlon@double = True
```

14.36 Função lonGlobeFo

Gera os valores de longitude e metadados associados para uma grade global fixa deslocada.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/lonGlobeFo.shtml

Exemplo: Cria um vetor de longitudes iniciando no Meridiano de Greenwich. Com o exemplo abaixo serão criados 144 pontos de longitude, a dimensão lon, long_name longitude e a unidade degrees_east.

```
mlon = 144
lon = lonGlobeFo(mlon,"lon","longitude","degrees_east")
print(lon)
```

Resultado:

```
Variable: lon
Type: float
Total Size: 576 bytes
            144 values
Number of Dimensions: 1
Dimensions and sizes: [lon |144]
Coordinates:
            lon: [1.25..358.75]
Number Of Attributes: 2
      long_name : longitude
      units : degrees_east
(0) 1.25
(1) 3.75
(2) 6.25
(3) 8.75
(4) 11.25
.
.
.
(139) 348.75
(140) 351.25
(141) 353.75
(142) 356.25
(143) 358.75
```

Nota1: Caso seja necessário transformar a dimensão para dupla precisão, basta fazer:

```
mlon@double = True
```

14.37 Função max

Calcula o valor máximo de um arranjo de qualquer dimensão.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/max.shtml

Exemplo:

f = (/2.1, 3.2, 4.3, 5.4, 6.5, 7.6, 8.7, 9.8/)
max_f = max(f)
print(max_f) ; Resposta: 9.8

14.38 Função maxind

Retorna o índice da primeira ocorrência do máximo valor.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/maxind.shtml

Exemplo:

x = (/3.,2.,5.,1.,5.,2.,5.,1.,3.,2./)
i = maxind(x) ; Resposta: 2.

14.39 Função min

Calcula o valor mínimo de um arranjo de qualquer dimensão.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/min.shtml

Exemplo:

f = (/2.1, 3.2, 4.3, 5.4, 6.5, 7.6, 8.7, 9.8/)
min_f = min(f)
print(min_f) ; Resposta: 2.1

14.40 Função minind

Retorna o índice da primeira ocorrência do mínimo valor.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/minind.shtml

Exemplo:

x = (/3.,2.,5.,1.,5.,2.,5.,1.,3.,2./)
i = minind(x) ; Resposta: 3.

14.41 Função mixhum_ptd

Calcula a razão de mistura ou umidade específica dada a pressão (em Pa) e a temperatura do ponto de orvalho (em Kelvin). Para gerar a razão de mistura, utiliza-se a opção 1 (kg/kg) e 2 para gerar a umidade específica (kg/kg). Caso seja -1 ou -2 gera os valores em g/kg.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/mixhum_ptd.shtml

Exemplo1:

p = 100000 ; Pa
tdk= 6.4 + 273.15 ; K
w = mixhum_ptd(p,tdk,1) ; ==> 0.006061566 (kg/kg)

Examplo2:

Caso a variável pressão (p) esteja em hPa, ela precisa ser convertida para Pa.

w = mixhum_ptd(p*100.,tdk,-1) ; ==> 6.06 (g/kg)

14.42 Função mixhum_ptrh

Calcula a razão de mistura ou umidade específica dada a pressão (em hPa), temperatura (em Kelvin) e umidade relativa (%). Para gerar a razão de mistura, utiliza-se a opção 1 (kg/kg) e 2 para gerar a umidade específica (kg/kg). Caso seja -1 ou -2 gera os valores em g/kg.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/mixhum_ptrh.shtml

Exemplo1:

p = 1000. ; hPa
tk = 18. + 273.15 ; K
rh = 46.5 ; %
q = mixhum_ptrh(p,tk,rh,1) ; q = 0.006018462 (kg/kg)
q = mixhum_ptrh(p,tk,rh,-1) ; q = 6.018462 (g/kg)
q = mixhum_ptrh (p,tk,rh,2) ; q = 0.005982457 (kg/kg)
q = mixhum_ptrh (p,tk,rh,-2) ; q = 5.982456 (g/kg)

Examplo2:
Caso a variável pressão (p) esteja em Pa, ela precisa ser convertida para hPa, para isso, basta multiplicar por 0.01.

p = 100000. ; Pa
w = mixhum_ptrh (p*0.01,t,rh,-1)

14.43 Função mod

Calcula o resto da divisão. O ideal é que os valores sejam do mesmo tipo, por exemplo, inteiro ou float.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/mod.shtml

Exemplo:

a = mod(17,3) = 2 ; tipo = integer

b = mod(17.5,5.5) = 1 ; tipo = float

14.44 Função month_to_annual

Converte valores mensais para anuais. A dimensão tempo deve ser a dimensão mais a esquerda e deve ser divisível por 12. A opção 0 calcula a soma não ponderada de 12 valores e a opção 1 divide pela soma não ponderada por 12 para obter o valor médio anual.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/month_to_annual.shtml

Por exemplo, para precipitação é aconselhável a opção 0 porque realiza a soma dos 12 valores e retorna o seu resultado. Para temperatura, recomenda-se a opção 1 que será retornada a média anual. Todos os meses devem estar presentes para ser realizado o cálculo, do contrário será retornado _FillValue para esse ano.

Exemplo1: A variável Umidade Relativa (ur) possui 120 meses (10 anos) com todos os meses. A conversão de dados mensais para média anual é feita de seguinte forma:

ur_media_anual = month_to_annual(ur,1) ; 1 para fazer média anual

Após o uso da função, os 120 meses foram convertidos para 10 anos (120 meses/12=10 anos).

Resultado do printVarSummary(ur_media_anual):

Dimensions and sizes: [year |10] x [level |3] x [lat |73] x [lon |144]

Exemplo2: Calculando a soma anual de precipitação que apresenta 48 meses completos.

ppt_soma_anual=month_to_annual(ppt,0) ; 0 para fazer a soma anual

Após o uso da função, os 48 meses foram convertidos para 4 anos (48 meses/12=4 anos).

Resultado do printVarSummary(ppt_soma_anual):

Dimensions and sizes: [year |4] x [lat |28] x [lon |24]

14.45 Função month_to_annual_weighted

Calcula os valores anuais ponderados pelo número de dias do mês. É necessário o argumento tempo no formato yyyymm (exemplo, 200001), yyyy com 4 digítos e mm com 2 digítos. A dimensão tempo deve ser a dimensão mais a direita e deve ser divisível por 12. As seguintes opções estão disponíveis: 0 retorna a soma ponderada, 1 retorna a média anual ponderada e 2 retorna a soma ponderada dividido por 12 para obter o valor médio mensal.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/month_to_annual_weighted.shtml

Para a variável precipitação utiliza-se a opção 0 e para a temperatura a opção 1.

Os pesos são os números de dias adequados para um mês. O mês de fevereiro de ano bissexto utiliza 29 dias.

Se o intervalo de tempo não é divisível por 12, então os meses "extras" são ignoradas. Por exemplo, se yyyymm está compreendido entre 199101 e 199603, serão calculadas as médias anuais ponderadas de 1991 a 1995 porque 1996 não está com os meses completos.

Exemplo:

```
1   ; Nome do script: cap14_ex31.ncl
2
3   begin
4
5   f = addfile("../../dados/prec.1986.1989.nc","r")
6
7   ;*********************************************************
8   ; Leitura do tempo e criação dele no formato yyyymm
9   ;*********************************************************
10  time   = f->time   ; time:units = "days since 1990-1-1 00:00:00"
11  TIME   = cd_calendar(time,0) ; Tipo float.
12  year   = toint(TIME(:,0))
13  month  = toint(TIME(:,1))
14  yyyymm = year*100 + month ; É necessário como argumento para
15                            ; o cálculo.
16  ;*********************************************************
17
18  ppt = f->precip
19
20  ; Soma anual, por isso a opção 0 na função abaixo.
21
22  prc_total_anual = month_to_annual_weighted(yyyymm,ppt,0)
23
24  printVarSummary(prc_total_anual)
25
26  end
```

14.46 Função month_to_season

Calcula a média sazonal de três meses (DJF, JFM, FMA, MAM, AMJ, MJJ, JJA, JAS, ASO, SON, OND, NDJ). A dimensão tempo deve ser divisível por 12 e assume-se que os dados são mensais e que o primeiro registro seja janeiro. O argumento trimestral é do tipo string, por exemplo, "DJF" ou "JJA".

A primeira (DJF=JF) e a última (NDJ=ND) média representam a média dos dois meses.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/month_to_season.shtml

Exemplo1: A variável ppt(time,lat,lon) tem 48 meses. Para calcular a média trimestral JJA, basta fazer:

xJJA = month_to_season(ppt,"JJA")

Exemplo2: Cálculo para o trimestre "DJF".

xDJF = month_to_season(ppt,"DJF")

14.47 Função month_to_seasonN

Calcula a média sazonal de três meses dado uma lista definida pelo usuário (DJF, JFM, FMA, MAM, AMJ, MJJ, JJA, JAS, ASO, SON, OND, NDJ). A dimensão tempo deve ser divisível por 12 e assume-se que os dados são mensais e que o primeiro registro seja janeiro. O argumento trimestral é um arranjo do tipo string, por exemplo, (/"DJF","JJA","SON"/).

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/month_to_seasonN.shtml

A primeira (DJF=JF) e a última (NDJ=ND) média representam a média dos dois meses.

Exemplo: A variável ppt(time,lat,lon) possui 48 meses e para realizar a média trimestral de MAM e SON, basta fazer:

ppt_med = month_to_seasonN(ppt,(/"MAM","SON"/))

Será criada uma nova dimensão na variável ppt_med chamada de season (em vermelho) que corresponde as estações selecionadas, isto é, duas estações em que o índice 0 representa a estação "MAM" e o índice 1, "SON". Como o dado possui 48 meses e foram feitas as médias para MAM e SON, logo o resultado será de 4 tempos para cada trimestre, ou seja, média de MAM_1986, MAM_1987, MAM_1988 e MAM_1989. O mesmo raciocínio se aplica ao trimestre SON.

O resultado do printVarSummary(ppt_med) será:

Variable: ppt_med
Type: float

Total Size: 21504 bytes
5376 values
Number of Dimensions: 4
Dimensions and sizes: [season |2] x [time |4] x [lat |28] x [lon |24]
Coordinates:
season: [MAM..SON]
time: [68025..69121]
lat: [8.75..-58.75]
lon: [271.25..328.75]

14.48 Função ndtooned

Converte um arranjo multidimensional para um arranjo unidimensional.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/ndtooned.shtml

Exemplo:

a = (/(/1,2,3/),(/4,5,6/),(/7,8,9/)/) ; arranjo 3x3
b = ndtooned(a)
print(b) ; Resposta: (/1,2,3,4,5,6,7,8,9/)

14.49 Função new

Cria uma variável vazia.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/new.shtml

Sintaxe: x = new(tamaho das dimensões, tipo do dado, parâmetro)

Em que: tamanho das dimensões é um valor inteiro ou double (que permite criar variáveis com mais de 2Gb em uma arquitetura 64bits). O tipo do dado é uma string informando se o dado é float, integer, string e esse nome pode ser com ou sem aspas duplas. O parâmetro que é opcional pode assumir um valor ausente (_FillValue) ou uma string "No_FillValue" para indicar que nenhum atributo No_FillValue será criado. Caso não seja definido esse argumento que é opcional, o valor ausente padrão para o tipo da variável será usado.

Exemplo1: Criando uma variável vazia com 2 linhas e 3 colunas (6 elementos no total) do tipo integer.

x = new((/2,3/),"integer")

```
ncl 0> x = new((/2,3/),"integer")
ncl 1> printVarSummary(x)

Variable: x
Type: integer
Total Size: 24 bytes
            6 values
Number of Dimensions: 2
Dimensions and sizes:   [2] x [3]
Coordinates:
Number Of Attributes: 1
  _FillValue :  -2147483647
ncl 2>
ncl 3> print(x)


Variable: x
Type: integer
Total Size: 24 bytes
            6 values
Number of Dimensions: 2
Dimensions and sizes:   [2] x [3]
Coordinates:
Number Of Attributes: 1
  _FillValue :  -2147483647
(0,0)   -2147483647
(0,1)   -2147483647
(0,2)   -2147483647
(1,0)   -2147483647
(1,1)   -2147483647
(1,2)   -2147483647
```

Exemplo2: Criando uma variável vazia com 5 valores do tipo float.

```
ncl 0> y = new(5,"float")
ncl 1> printVarSummary(y)

Variable: y
Type: float
Total Size: 20 bytes
            5 values
Number of Dimensions: 1
Dimensions and sizes:   [5]
Coordinates:
Number Of Attributes: 1
  _FillValue :  9.96921e+36
ncl 2>
ncl 3> print(y)


Variable: y
Type: float
Total Size: 20 bytes
            5 values
Number of Dimensions: 1
Dimensions and sizes:   [5]
Coordinates:
Number Of Attributes: 1
  _FillValue :  9.96921e+36
(0)     9.96921e+36
(1)     9.96921e+36
(2)     9.96921e+36
(3)     9.96921e+36
(4)     9.96921e+36
```

14.50 Função num

Conta o número de valores verdadeiros (True) na variável de entrada e ignora valores ausentes.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/num.shtml

Exemplo1:

a = (/1,2,3,4,5/)

print(num(a.gt.3)) ; Imprime na tela quantos valores são maiores que 3.

O resultado será: 2

14.51 Função omega_to_w

Converte velocidade vertical omega (Pa/s) em w (m/s). Para essa função são necessárias três variáveis, isto é, omega (Pa/s), pressão (Pa) e temperatura (K). Todas as 3 variáveis devem ter as mesmas coordenadas e dimensões.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/omega_to_w.shtml

Exemplo:

```
; Nome do script: cap14_ex32.ncl

begin

f = addfile("../../dados/omega.pressao.tar.R1.nc","r")

omega   = f->omega ; (Pa/s) => float omega(time, level, lat, lon)
pressao = f->pres  ; (Pa)   => float pres(time, lat, lon)
tar     = f->air   ; (K)    => float air(time, level, lat, lon)

; Lembrando que as variáveis devem ter o mesmo número de dimensões
; e a mesma forma. A variável pressão não apresenta as mesmas
; dimensões das demais variáveis (omega e tar). Para resolver isso,
; será utilizada a função conform para uniformizar todas as dimensões.
; As dimensões da variável "omega" que coincidem com as dimensões da
; variável "pressao" são: 0 = tempo, 2 = latitude e 3 = longitude.

; A variável "pressao_N" possui as mesmas coordenadas e dimensões de
; "omega". Houve perda das informações de coordenadas e dimensões.
pressao_N = conform(omega,pressao,(/0,2,3/))

printVarSummary(pressao_N)

; Dimensions and sizes: [5] x [12] x [73] x [144]

; Copia as coordenadas e dimensões da variável "omega" para "pressao_N"
copy_VarCoords(omega,pressao_N)

printVarSummary(pressao_N)

; Dimensions and sizes: [time | 5] x [level | 12] x [lat | 73] x [lon |

w = omega_to_w(omega,pressao_N,tar) ; Cálculo de w (m/s)

printVarSummary(w)

; Dimensions and sizes: [time | 5] x [level | 12] x [lat | 73] x [lon |

end
```

Utilize a função w_to_omega para converter de m/s para Pa/s.

http://www.ncl.ucar.edu/Document/Functions/Contributed/w_to_omega.shtml

14.52 Função onedtond

Converte um arranjo unidimensional para multidimensional.

Se o produto de saída das dimensões for menor do que o número de elementos do arranjo de entrada, uma mensagem de aviso será mostrada e o arranjo de saída é preenchido com alguns valores do dado de entrada. Caso o tamanho das dimensões de saída seja maior do que o dado de entrada, então o dado de entrada é repetido várias vezes até preencher o dado de saída.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/onedtond.shtml

Exemplo1: Caso onde a entrada e a saída possuem o mesmo número de dimensões.

a = (/1,2,3,4,5,6,7,8/) ; Um vetor qualquer.
a0 = onedtond(a,(/2,4/)) ; Transforma de uma dimensão para duas dimensões.
print(a0) ; Mostra na tela o resultado.

Resultado do print(a0)

```
Variable: a0
Type: integer
Total Size: 32 bytes
            8 values
Number of Dimensions: 2
Dimensions and sizes: [2] x [4]
Coordinates:
(0,0) 1
(0,1) 2
(0,2) 3
(0,3) 4
(1,0) 5
(1,1) 6
(1,2) 7
(1,3) 8
```

Exemplo2: Existem duas vezes o número de elementos tanto na saída quanto na entrada. Neste caso, os valores são copiados duas vezes.

a1 = onedtond(a,(/2,8/))
print(a1)

Resultado do print(a1)

```
Variable: a1
Type: integer
Total Size: 64 bytes
            16 values
Number of Dimensions: 2
Dimensions and sizes: [2] x [8]
```

Coordinates:
(0,0) 1
(0,1) 2
(0,2) 3
(0,3) 4
(0,4) 5
(0,5) 6
(0,6) 7
(0,7) 8
(1,0) 1
(1,1) 2
(1,2) 3
(1,3) 4
(1,4) 5
(1,5) 6
(1,6) 7
(1,7) 8

Exemplo3: Caso em que há metade do número de elementos tanto na saída quanto na entrada. Apenas os primeiros quatro valores da entrada são copiados para a saída, gerando uma mensagem de aviso (warning).

a2 = onedtond(a,(/2,2/)) ; warning:onedtond : output dimension sizes have fewer elements than input, some data not copied
print(a2)

Resultado do print(a2)

Variable: a2
Type: integer
Total Size: 16 bytes
4 values
Number of Dimensions: 2
Dimensions and sizes: [2] x [2]
Coordinates:
(0,0) 1
(0,1) 2
(1,0) 3
(1,1) 4

Exemplo4: Caso em que há mais elementos na saída do que na entrada.

a3 = onedtond(a,(/2,7/)) ; warning:onedtond : output dimension sizes not even multiples of input, check output
print(a3)

Resultado do print(a3)

Variable: a3
Type: integer

Total Size: 56 bytes
14 values
Number of Dimensions: 2
Dimensions and sizes: [2] x [7]
Coordinates:
(0,0) 1
(0,1) 2
(0,2) 3
(0,3) 4
(0,4) 5
(0,5) 6
(0,6) 7
(1,0) 8
(1,1) 1
(1,2) 2
(1,3) 3
(1,4) 4
(1,5) 5
(1,6) 6

14.53 Função pot_temp

Calcula a temperatura potencial dada a pressão (hPa) e a temperatura (K).

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/pot_temp.shtml

Exemplo:

```
; Nome do script: cap14_ex33.ncl

begin

f = addfile("../../dados/tar.pres.uwnd.vwnd.nc","r")

                              ; Índices    0      1     2    3
tar     = short2flt(f->air)   ; short air(time, level, lat, lon),
                              ; por isso, o uso da função short2flt.
z       = f->level            ; Níveis de pressão em hPa. A variável
                              ; "level" pertence ao arquivo aberto.

                                 ; Lembrando que z = hPa e tar = K.
pt1  = pot_temp(z,tar,1,False)   ; Cálculo da temperatura potencial.
                                 ; O valor 1 refe-se a dimensão de
                                 ; "tar" que corresponde a "z".

printVarSummary(pt1)

end
```

14.54 Função pot_vort_isobaric

Calcula a vorticidade potencial em níveis isobáricos em uma grade fixa (regular) dado os níveis de pressão (Pa) em apenas uma dimensão, componentes zonal e meridional do vento (m/s) e temperatura (K). As variáveis do vento e de temperatura devem estar dispostas no sentido sul para norte. O usuário pode definir o tipo de grade como saída, por exemplo, gridType = 0 significa uma grade gaussiana e gridType=1 significa uma grade regular ou fixa. A variável opt pode ser 0 que retorna a vorticidade potencial ou 1 que retorna uma lista de variáveis contendo a vorticidade potencial, a estabilidade estática e a temperatura potencial.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/pot_vort_isobaric.shtml

Exemplo:

```
; Nome do script: cap14_ex34.ncl

begin

f = addfile("../../dados/tar.pres.uwnd.vwnd.nc","r")

u    = short2flt(f->uwnd) ; (m/s). short air(time, level, lat, lon),
v    = short2flt(f->vwnd) ; (m/s). short air(time, level, lat, lon),
t    = short2flt(f->air)  ; (K). short air(time, level, lat, lon),
                          ; por isso, o uso da função short2flt.

lev  = f->level           ; Níveis de pressão em hPa. A variável
                          ; "level" pertence ao arquivo aberto.
lev  = lev*100            ; Converte os níveis verticais de
                          ; hPa para Pa.

lat  = f->lat             ; As coordenadas de latitude. A variável
                          ; "lat" pertence ao arquivo aberto.

gridType = 0    ; 0 = grade gaussiana e 1 = grade fixa (regular).
opt      = 0    ; 0 = calcula a vorticidade potencial.
                ; 1 = retorna uma lista de variáveis:
                ; vorticidade potencial, estabilidade estática e
                ; temperatura potencial.

; Caso o dado não esteja na disposição sul->norte, será retornada
; a seguinte mensagem:
; (0)   pot_vort_isobaric: data must be in S-N order.
; Para resolver isso, basta inverter a latitude utilizando o operador ":".
; Por exemplo, "::-1", com isso, o dado passa a ser orientado de sul->norte.

; Calcula a PV e inverte as coordenadas de sul->norte.

PV = pot_vort_isobaric(lev,u(:,:,::-1,:),v(:,:,::-1,:),t(:,:,::-1,:),\
                       lat(::-1),gridType,opt)

printVarSummary(PV)

end
```

Como saber se o dado está de norte para sul ou de sul para norte? Com o comando printVarSummary é possível verificar isso.

Exemplo:

printVarSummary(u)

Coordinates:
time: [1569072..1570488]
level: [1000..100]
lat: [90..-90] ; dado de norte para sul (90 a -90)
lon: [0..357.5]

Para inverter a latitude, basta usar o "::-1" no dimensão latitude.

14.55 Função prcwater_dp

Calcula a coluna de água precipitável da dimensão mais a direita, isto é, a dimensão nível vertical. Caso ela não seja a dimensão mais a direita, será necessário fazer a reordenação das dimensões para satisfazer essa condição. São necessários dois argumentos para utilizar essa função, a umidade específica em kg/kg e os níveis de pressão em Pa. A unidade da coluna de água precipitável será em kg/m^2.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/prcwater_dp.shtml

Exemplo:

```
; Nome do script: cap14_ex35.ncl

begin

f = addfile("../../dados/SA.umidade.q.psfc.R1.2010.nc","r")

q    = f->shum  ; g/kg => float shum ( time, level, lat, lon )
z    = f->level ; hPa  => float level ( level )
psfc = f->pres  ; hPa  => float pres(time, lat, lon)
q    = q/1000.0 ; g/kg->kg/kg
ptop = 300      ; Até onde será integrado (hPa).

dp = dpres_plevel_Wrap(z,psfc,ptop,0) ; Calculo da espessura da camada
                                      ; em hPa.
dp = dp*100 ; hPa->Pa. dp será utilizado para o cálculo de "pw".


dp_N = dp(time|:,lat|:,lon|:,level|:) ; Reordenamento das dimemsões.
                                      ; A dimensão level tem que ser a
                                      ; dimensão mais a direita.

q_N  = q(time|:,lat|:,lon|:,level|:)  ; A dimensão "level" precisar
                                      ; estar mais a direita. Por isso,
                                      ; foi feita a reordenação da
                                      ; dimensões.

; "q_N" possui agora a seguinte ordem das dimensões:
; Dimensions and sizes: [time | 12] x [lat | 73] x [lon | 144] x [level | 8]
```

```
pw = prcwater_dp(q_N,dp_N) ; pw (kg/m2)
                           ; Dimensions and sizes: [12] x [73] x [144]
                           ; A variável "pw" não tem as coordenadas.
                           ; Será criada uma variável auxiliar para
                           ; resolver isso.

q_aux = q(:,0,:,:) ; Artifício para copiar as coordenadas de "q_aux" para
                   ; "pw". Ao fixar qualquer nível vertical, ocorre a
                   ; redução de dimensões. Neste caso, restaram apenas as
                   ; dimensões tempo, lat e lon que serão copiadas para
                   ; a variável "pw" que contém apenas 3 dimensões.

copy_VarCoords(q_aux,pw) ; Copia as coordenadas e dimensões de "q_aux"
                         ; para "pw".

printVarSummary(pw)
; Dimensions and sizes: [time | 12] x [lat | 73] x [lon | 144]

end
```

14.56 Função printMinMax

Imprime os valores mínimo e máximo de uma variável.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/printMinMax.shtml

Exemplo:

f=addfile("precip.mon.mean.nc","r")
ppt=f->precip
printMinMax(ppt,False)
; Average Monthly Rate of Precipitation: min=0 max=47.076

14.57 Função printVarSummary

Imprime um resumo das informações de uma variável.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/printVarSummary.shtml

Exemplo:

f=addfile("precip.mon.mean.nc","r")
ppt=f->precip
printVarSummary(ppt)

Resultado do printVarSummary(ppt)

Variable: ppt
Type: float
Total Size: 18330624 bytes
4582656 values
Number of Dimensions: 3

Dimensions and sizes: [time |442] x [lat |72] x [lon |144]
Coordinates:
time: [65378..78800]
lat: [88.75..-88.75]
lon: [1.25..358.75]
Number Of Attributes: 15
long_name : Average Monthly Rate of Precipitation
valid_range : (0, 100)
units : mm/day
add_offset : 0
scale_factor : 1
missing_value : -9.96921e+36
precision : 32767
least_significant_digit : 2
var_desc : Precipitation
dataset : GPCP Version 2.2 Combined Precipitation Dataset
level_desc : Surface
statistic : Mean
parent_stat : Mean
actual_range : (0, 9.985388e+29)
_FillValue : -9.96921e+36

Caso seja necessário imprimir as informações sobre uma variável em um arquivo, não utilize printVarSummary, uma vez que irá copiar toda a variável na memória. Em vez disso, use o printFileVarSummary.

Exemplo:

printFileVarSummary(f,"precip")

Em que "f" faz referência a unidade do arquivo aberto e "precip" é o nome da variável do arquivo.

14.58 Função qsort

Ordena um arranjo em ordem ascendente.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/qsort.shtml

Exemplo:

```
x = (/4.3, 0.9, 5.2, 7.7, 2.3, -4.7, -9.1/)
qsort(x)
print(x)
```

Resultado do print(x)

Variable: x
Type: float

Total Size: 28 bytes
7 values
Number of Dimensions: 1
Dimensions and sizes: [7]
Coordinates:
(0) -9.1
(1) -4.7
(2) 0.9
(3) 2.3
(4) 4.3
(5) 5.2
(6) 7.7

14.59 Função relhum

Calcula a umidade relativa do ar (%) dado a temperatura (K), razão de mistura (kg/kg) e pressão (Pa).

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/relhum.shtml

Exemplo:

```
; Nome do script: cap14_ex36.ncl

begin

f = addfile("../../dados/shum.air.rhum.pres.nc","r")

t    = f->air   ; (C)     float air(time, level, lat, lon)
q    = f->shum  ; (g/kg)  float shum(time, level, lat, lon)
rh   = f->rhum  ; (%)     float rhum(time, level, lat, lon)
psfc = f->pres  ; (hPa)   float pres(time, lat, lon)

tk = t+273.15 ; Tc->Tk
q  = q*0.001  ; g/kg->kg/kg
p  = psfc*100 ; hPa->Pa. Será utilizado no cálculo de rh.

; Todas as variáveis precisam ter o mesmo número de dimensões. A pressão
; apresenta apenas 3 (time,lat,lon) enquanto as demais possuem
; 4 (time,level,lat,lon). Por isso, o uso da função conform para
; deixar a pressão com o mesmo número de dimensões das demais variáveis.

; p1 = pressão à superfície.
; p1 será usado para calcular "q" que utiliza pressão em hPa.
p1 = conform(tk,psfc,(/0,2,3/)) ; As dimensões de "tk" que são
                                ; iguais a "p1" são 0 = time,
                                ; 2 = lat e 3 = lon

; p2 = pressão à superfície.
; p2 será usado para calcular "rh" que utiliza pressão em Pa.
p2 = conform(tk,p,(/0,2,3/)) ; As dimensões de "tk" que são
                             ; iguais a "p2" são 0 = time,
                             ; 2 = lat e 3 = lon.

```

```
; Cálculo da razão de mistura que será utilizado para obtenção de "rh".
q = mixhum_ptrh(p1,tk,rh,1) ; q (kg/kg), por isso a opção 1.

; Cálculo da umidade relativa (%).
rh = relhum(tk,q,p2)

rh = rh < 100 ; truque do NCL para fazer com que os valores maiores que
              ; 100% sejam limitados a 100%.

printVarSummary(rh)

end
```

14.60 Função rmAnnCycle1D

Remove o ciclo anual de uma série temporal unidimensional. A dimensão tempo tem que ser múltiplo de 12.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/rmAnnCycle1D.shtml

Exemplo:

f=addfile("SA.prec.1979.2014.nc","r")
ppt=f->precip(:,0,0) ; float precip (time, lat, lon). O dado tem que ser uma série temporal.
rmppt=rmAnnCycle1D(ppt)
printVarSummary(rmppt)

14.61 Função rmMonAnnCycTLL

Remove o ciclo anual de dados mensais e subtrai a média de longo período de cada mês. As dimensões devem estar na seguinte ordem: time,lat,lon, por isso, o TLL. A dimensão tempo deve ser múltiplo de 12.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/rmMonAnnCycTLL.shtml

Exemplo:

f=addfile("precip.mon.1979.2014.nc","r")
ppt=f->precip ; float precip (time, lat, lon) = TLL
rmppt=rmMonAnnCycTLL(ppt) ; Remove o ciclo anual
printVarSummary(rmppt) ; mostra na tela o resultado

Outras possibilidades para remoção do ciclo anual para dados mensais são listadas abaixo. Lembrando que os meses devem ser múltiplos de 12:

- Função rmMonAnnCycLLLT: O dado deve possuir as dimensões nessa ordem: level,lat,lon,time, por isso, o LLLT.

- Função rmMonAnnCycLLT: O dado deve possuir as dimensões nessa ordem: lat,lon,time, por isso, o LLT.

14.62 Função round

Arredonda uma variável float ou double para o número inteiro mais próximo. Os argumentos podem ser um arranjo de um ou mais valores de qualquer dimensionalidade e opt pode assumir as seguintes opções:

opt=0: retorna um valor do mesmo tipo do dado de entrada
opt=1: retorna um valor do tipo float
opt=2: retorna um valor do tipo double
opt=3: retorna um valor do tipo inteiro

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/round.shtml

Exemplo:

```
x = 5.7
q = round(x,0)   ; ===> q=6 (tipo float)
r = round(x,1)   ; ===> r=6 (tipo float)
d = round(x,2)   ; ===> d=6 (tipo double)
i = round(x,3)   ; ===> i=6 (tipo inteiro)
```

14.63 Função runave_n_Wrap

Realiza a suavização do dado. São quatro os argumetos necessários, são eles: a variável a ser feita a média, o comprimento da média (valor inteiro), a opção que pode ser menor (utiliza a condição ciclíca), igual (define os pontos inicias e finais como indefinidos) ou maior que zero (utiliza condições reflectivas (simétricas)). Normalmente, utiliza-se o valor zero e a dimensão a ser feita a suavização deve ser a dimensão mais a direita.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/runave.shtml

Exemplo:

```
f=addfile("SA.prec.1979.1980.nc","r")
ppt=f->precip
rppt=runave_n_Wrap(ppt,3,0,0) ; suavização da variável ppt, suavização de 3 três valores com opção 0 na dimensão tempo (0).
print(rppt)
```

Resultado do comando print(rppt). Os valores em vermelho são os indefinidos conforme a opção zero que fixa os valores iniciais e finais com essa característica.

```
Variable: rppt
Type: float
Total Size: 96 bytes
            24 values
Number of Dimensions: 3
Dimensions and sizes:   [time | 24] x [lat | 1] x [lon | 1]
Coordinates:
            time: [65378..66078]
            lat: [ 0.. 0]
            lon: [ 0.. 0]
Number Of Attributes: 13
  _FillValue :  -9.96921e+36
  long_name :   Average Monthly Rate of Precipitation
  units :       mm/day
  precision :   32767
  least_significant_digit :     2
  var_desc :    Precipitation
  dataset :     GPCP Version 2.2 Combined Precipitation Dataset
  level_desc :  Surface
  statistic :   Mean
  parent_stat : Mean
  actual_range :        (  0, 9.985388e+29 )
  missing_value :       -9.96921e+36
  runave_op_ncl :       runave_n: nave=3
(0,0,0) -9.96921e+36
(1,0,0) 3.99178
(2,0,0) 3.981366
(3,0,0) 3.897237
(4,0,0) 3.532421
(5,0,0) 3.21502
(6,0,0) 2.94638
(7,0,0) 2.964706
(8,0,0) 3.069009
(9,0,0) 3.345414
(10,0,0)        3.549695
(11,0,0)        3.819283
(12,0,0)        4.064079
(13,0,0)        4.157448
(14,0,0)        4.13774
(15,0,0)        3.889276
(16,0,0)        3.720415
(17,0,0)        3.382187
(18,0,0)        3.035251
(19,0,0)        2.832716
(20,0,0)        2.896988
(21,0,0)        3.216803
(22,0,0)        3.539978
(23,0,0)        -9.96921e+36
```

14.64 Função smthClmDayTLL

Calcula o ciclo anual diário suavizado de uma arranjo com as seguintes dimensões: time,lat,lon, por isso, o TLL. Os argumentos necessários são: a climatologia diária no formato (366,lat,lon) e o número de harmônicos para construir o ciclo anual médio suavizado. Normalmente, o número de harmônicos varia entre 1 e 3, sendo 2

o número mais comum que significa usar somente os harmônicos anual e semi-anual.

O valor retornado será uma série onde a dimensão mais a esquerda refere-se ao dia sequencial do ano.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/smthClmDayTLL.shtml

Exemplo:

```
; Nome do script: cap14_ex37.ncl

begin

f = addfile("../../dados/olr.01jan2000a31dez2004.nc","r")
;*************************************************************
; Leitura do tempo e criação dele no formato yyyyddd.
;*************************************************************
time    = f->time
TIME    = cd_calendar(time, 0) ; tipo float
year    = toint(TIME(:,0))
month   = toint(TIME(:,1))
day     = toint(TIME(:,2))
ddd     = day_of_year(year,month,day)
yyyyddd = year*1000 + ddd      ; É necessário como argumento
                               ; para o cálculo da climatologia.
;*************************************************************
; Leitura do dado
;*************************************************************
olr = short2flt(f->olr) ; short olr ( time, lat, lon )
;*************************************************************
; Calcula a climatologia diária em cada ponto de grade.
;*************************************************************
olrClmDay  = clmDayTLL(olr,yyyyddd)
;*************************************************************
; Calcula a climatologia suavizada usando 2 harmônicos.
;*************************************************************
olrClmDay_sm = smthClmDayTLL(olrClmDay,2)

printVarSummary(olrClmDay_sm)

; O resultado do comando printVarSummary(olrClmDay_sm) será:
; Dimensions and sizes: [year_day | 366] x [lat | 73] x [lon | 144]

end
```

Outra possibilidade:

- Função smthClmDayTLLL: as dimensões do dado devem estar na seguinte ordem: time,level,lat,lon.

14.65 Função stdatmus_p2tdz

Calcula a temperatura (^{o}C), densidade (kg/m^3) e altura (m) baseado na atmosfera padrão dos Estados Unidos de 1976. O parâmetro de entrada é a pressão em hPa (mb).

Nota: Valores não válidos acima de 84852 metros.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/stdatmus_p2tdz.shtml

Exemplo:

p = 1001.29 ; hPa
tdz = stdatmus_p2tdz(p)
print(tdz)

Resultado do comando print(tdz)

Variable: tdz
Type: float
Total Size: 12 bytes
3 values
Number of Dimensions: 1
Dimensions and sizes: [3]
Coordinates:
(0) 14.34976 ; temperatura (^{o}C)
(1) 1.213278 ; densidade (kg/m^3)
(2) 100.037 ; altura (m)

14.66 Função stdatmus_z2tdp

Calcula a temperatura (^{o}C), densidade (kg/m^3) e altura (m) baseado na atmosfera padrão dos Estados Unidos de 1976. O parâmetro de entrada é a altura (m).

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/stdatmus_z2tdp.shtml

z = 100 ; metros
tdp = stdatmus_z2tdp(z)
print(tdp)

Resultado do comando print(tdp):

Variable: tdp
Type: float
Total Size: 12 bytes
3 values
Number of Dimensions: 1
Dimensions and sizes: [3]
Coordinates:
(0) 14.35 ; temperatura (^{o}C)
(1) 1.213282 ; densidade (kg/m^3)
(2) 1001.294 ; pressão (hPa)

14.67 Função stdMonTLL

Calcula o desvio padrão de médias mensais. A dimensão tempo deve ser múltiplo de 12. O valor retornado será do mesmo tipo do dado de entrada sendo que a dimensão mais a esquerda terá comprimento 12.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/stdatmus_z2tdp.shtml

http://www.ncl.ucar.edu/Document/Functions/Built-in/stdatmus_z2tdp.shtml

http://www.ncl.ucar.edu/Document/Functions/Contributed/stdMonTLL.shtml

Exemplo:

```
f=addfile("precip.mon.1979.2014.nc","r")
ppt=f->precip
std_ppt=stdMonTLL(ppt)
printVarSummary(std_ppt)
```

Resultado do printVarSummary(std_ppt):

```
Variable: std_ppt
Type: float
Total Size: 497664 bytes
            124416 values
Number of Dimensions: 3
Dimensions and sizes: [month |12] x [lat |72] x [lon |144]
Coordinates:
            month: [0..11]
            lat: [88.75..-88.75]
            lon: [1.25..358.75]
```

Outras possibilidades:

- Função stdMonLLLT: as dimensões da variável devem estar na seguinte ordem: level,lat,lon,time, por isso, o LLLT.
- Função stdMonLLT: as dimensões da variável devem estar na seguinte ordem: lat,lon,time, por isso, o LLT.
- Função stdMonTLLL: as dimensões da variável devem estar na seguinte ordem: time,level,lat,lon, por isso, o TLLL.

14.68 Função system

Executa um comando shell.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/system.shtml

Exemplo:

system("date")

O resultado será:

Seg Mar 21 22:15:33 BRT 2016

Acesse o site acima para verificar as diversas possibilidades de uso desta função.

14.69 Função systemfunc

Executa um comando shell e retorna a saída.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/systemfunc.shtml

Exemplo:

TimeDate = systemfunc("date")

print(TimeDate)

Resultado do comando print(TimeDate):

(0) Seg Mar 21 22:18:17 BRT 2016

14.70 Função uv2dvF_Wrap

Calcula a divergência usando harmônicos esféricos dado as componentes zonal e meridional do vento em uma grade fixa (regular). Esta função não permite valores indefinidos, e caso seja encontrado algum, para este determinado ponto será considerado indefinido.

Se o dado não for global ou apresentar valores ausentes, utilize a função uv2dv_cfd.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/uv2dvF_Wrap.shtml

f=addfile("tar.pres.uwnd.vwnd.nc","r")
u=short2flt(f->uwnd) ; Dimensions and sizes: [time |3] x [level |12] x [lat |73] x [lon |144]
v=short2flt(f->vwnd) ; Dimensions and sizes: [time |3] x [level |12] x [lat |73] x [lon |144]
dv=uv2dvF_Wrap(u,v) ; Calcula a divergência (1/s)
printVarSummary(dv) ; Imprime na tela as informações de dv

Para uma grade gaussiana, utilize a função uv2dvG_Wrap.

14.71 Função uv2sfvpF

Calcula a função de corrente e a velocidade potencial via harmônicos esféricos dado as componentes zonal e meridional do vento em uma grade fixa (regular). O dado tem que estar em uma latitude ascendente e ser um dado global. Esta função não aceita valores indefinidos e caso seja encontrado algum em um determinado ponto, este será considerado indefinido.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/uv2sfvpF-1.shtml

Exemplo:

```
; Nome do script: cap14_ex38.ncl

begin

f = addfile("../../dados/tar.pres.uwnd.vwnd.nc","r")

u    = short2flt(f->uwnd) ; short vwnd ( time, level, lat, lon )
v    = short2flt(f->vwnd) ; short vwnd ( time, level, lat, lon )
                          ; A latitude está em ordem ascendente:
                          ; lat: [90..-90] e o dado é global.

tempo = f->time  ; Será utilizado para criar a coordenada de tempo
                 ; da variável sfpv.
nivel = f->level ; Será utilizado para criar a coordenada de nível
                 ; vertical da variável sfpv.

sfvp = uv2sfvpF(u,v) ; Calcula a função de corrente e a velocidade
                     ; potencial.

; A disposição das dimensões da variável sfvp são:
; Dimensions and sizes: [2] x [3] x [12] x [73] x [144].
; A nova dimensão [2] quer dizer que o primeiro valor
; é a função de corrente e o segundo valor é a velocidade
; potencial. [3], [12], [73] e [144] são as dimensões:
; tempo, nível vertical, lat e lon, respectivamente.
; Essa variável não possui o nome das dimensões e
; coordenadas, por isso a necessidade de criação delas.

; Nome das dimensões para a variável sfvp.
sfvp!0 = "var"   ; Os nomes selecionados
sfvp!1 = "time"  ; são escolhidos pelo
sfvp!2 = "level" ; usuário. Agora, há 5
sfvp!3 = "lat"   ; dimensões. O ideal é manter
sfvp!4 = "lon"   ; o mesmo nome das dimensões da variável original.

; Criação das coordendas da variável sfvp.
sfvp&time  = tempo
sfvp&level = nivel
sfvp&lat   = latGlobeF(73,"lat","latitude","degrees_north")
sfvp&lon   = lonGlobeF(144,"lon","longitude","degrees_east")

printVarSummary(sfvp)

; Dimensions and sizes: [var | 2] x [time | 3] x [level | 12] x
;                       [lat | 73] x [lon | 144]

end
```

Outra possibilidade de cálculo:

- Função uv2sfvpG: Calcula a função de corrente e velocidade potencial para uma grade gaussiana.

14.72 Função uv2vrdvF

Calcula a vorticidade (1/s) e a divergência (1/s) via harmônicos esféricos dado as componentes zonal e meridional do vento em uma grade fixa (regular). Os valores de latitude devem estar na ordem ascendente e o dado deve ser global. A variável retornada terá uma dimensão extra que corresponde as variáveis vorticidade no índice zero e a divergência no índice 1. Esta função não trabalha com valores ausentes. Caso seja encontrado algum ponto indefinido, para este ponto em particular será considerado como indefinido.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/uv2vrdvF-1.shtml

Exemplo:

```
; Nome do script: cap14_ex39.ncl

begin

f = addfile("../../dados/tar.pres.uwnd.vwnd.nc","r")

u    = short2flt(f->uwnd) ; short vwnd ( time, level, lat, lon )
v    = short2flt(f->vwnd) ; short vwnd ( time, level, lat, lon )
                          ; A latitude está em ordem ascendente:
                          ; lat: [90..-90] e o dado é global.

tempo = f->time  ; Será utilizado para criar a coordenada de tempo
                 ; da variável vrdvx.
nivel = f->level ; Será utilizado para criar a coordenada de nível
                 ; vertical da variável vrdvx.

; Calcula a vorticidade e a divergência. A
; disposição das dimensões da variável vrdvx são:
; Dimensions and sizes: [2] x [3] x [12] x [73] x [144].
; A nova dimensão [2] quer dizer que o primeiro índice
; é a vorticidade e o segundo índice é a divergência.
; [3], [12], [73] e [144] são as dimensões:
; tempo, nível vertical, lat e lon, respectivamente.
; Essa variável não possui o nome das dimensões e
; coordenadas, por isso a necessidade de criação delas.

vrdvx = uv2vrdvF(u,v)

; Nome das dimensões para a variável vrdvx.
vrdvx!0 = "var"   ; Os nomes selecionados
vrdvx!1 = "time"  ; são escolhidos pelo
vrdvx!2 = "level" ; usuário. Agora, há 5
vrdvx!3 = "lat"   ; dimensões. O ideal é manter
vrdvx!4 = "lon"   ; o mesmo nome das dimensões das variáveis originais.
```

```
35
36  ; Criação das coordendas da variável vrdvx.
37  vrdvx&time  = tempo
38  vrdvx&level = nivel
39  vrdvx&lat   = latGlobeF(73,"lat","latitude","degrees_north")
40  vrdvx&lon   = lonGlobeF(144,"lon","longitude","degrees_east")
41
42  printVarSummary(vrdvx)
43
44  ; O resultado do comando acima mostra que a variável está com suas
45  ; dimensões e coordenadas:
46  ; Dimensions and sizes: [var | 2] x [time | 3] x [level | 12] x
47  ;                       [lat | 73] x [lon | 144]
48
49  end
```

Outras possibilidades:

- Função uv2vrdvG: Calcula a vorticidade e a divergência em uma grade gaussiana.
- Função uv2vrF_Wrap: Calcula somente a vorticidade em uma grade fixa e retém os metadados.
- Função uv2vrG_Wrap: Calcula somente a vorticidade em uma grade gaussiana e retém os metadados.

14.73 Função vr2uvF_Wrap

Calcula o rotacional das componentes zonal e meridional em m/s via harmônicos esféricos dado a vorticidade relativa (1/s) em uma grade fixa (regular). Os valores de latitude devem estar na ordem ascendente e o dado deve ser global. A variável retornada terá uma dimensão extra que corresponde ao rotacional da componente zonal no índice zero e o rotacional da componente meridional no índice 1. Esta função não trabalha com valores ausentes. Caso seja encontrado algum ponto indefinido, para este ponto em particular será considerado como indefinido.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/vr2uvF_Wrap.shtml

Exemplo:

```
; Nome do script: cap14_ex40.ncl

begin

f = addfile("../../dados/tar.pres.uwnd.vwnd.nc","r")

u    = short2flt(f->uwnd) ; short vwnd ( time, level, lat, lon )
v    = short2flt(f->vwnd) ; short vwnd ( time, level, lat, lon )
                          ; A latitude está em ordem ascendente:
                          ; lat: [90..-90] e o dado é global.

vrt = uv2vrF_Wrap(u, v)  ; Calcula a vorticidade (1/s).
uvr = vr2uvF_Wrap(vrt)   ; Calcula o rotacional das componentes
                         ; do vento (m/s).

printVarSummary(uvr)

; Resultado do comando printVarSummary(uvr):
; Dimensions and sizes: [component | 2] x [time | 3] x [level | 12] x
;                       [lat | 73] x [lon | 144]
; Onde: [component | 2] quer dizer que o índice 0 é o rotacional
; da componente zonal e 1 é o rotacional da componente meridional.

end
```

Para uma grade gaussiana, utilizar a função vr2uvG_Wrap.

14.74 Função where

Avalia uma expressão lógica e atribui um resultado ou um cálculo quando a condição for verdadeira, caso a condição não seja satisfeita um resultado ou um cálculo também é atribuído.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/where.shtml

Exemplo1:

xsqrt = where(x.gt.0,sqrt(x),x@_FillValue)

Onde x for maior que 0, retorna a raiz quadrada de x, e caso contrário, retorna o valor indefinido (x@_FillValue).

Exemplo2: Supondo que o dado é referente a uma máscara com valores 1 indicando continente e 0 indicando superfície líquida. Para realizar dois cálculos distintos sobre outra variável chamada "a" independente se é continente ou não, basta seguir o exemplo abaixo:

x = where(oro.eq.1,a+273.15,1.8*a+32)

Onde oro for igual a 1, soma 273.15 a variável "a" e caso contrário, multiplica "a" por 1.8 e depois soma 32.

14.75 Função wgt_areaave_Wrap

Calcula a média da área de uma quantidade utilizando pesos. Os metadados são retidos dispensando a criação das coordenadas.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/wgt_areaave_Wrap.shtml

Sintaxe: x = wgt_areaave_Wrap(q,wgty,wgtx,opt)

Em que:

- **q:** É um arranjo de duas ou mais dimensões contendo o dado. As dimensões mais a direita são latitude (lat) e longitude (lon).
- **wgty:** É um escalar (tipicamente 1.0) ou um arranjo de uma dimensão do tamanho da lat contendo os pesos.
- **wgtx:** É um escalar (tipicamente 1.0) ou um arranjo de uma dimensão do tamanho da lon contendo os pesos.
- **opt:** Se opt=0 a área média é calculada sem dados faltantes. Se opt=1 então qualquer ponto de q que contenha dado faltante, a média da área não será calculada.

O resultado será um escalar caso a variável utilizada tenha duas dimensões. Quando a variável tiver 4 dimensões, nessa ordem: time, lev, lat e lon, serão retornadas duas dimensões, isto é, time e lev porque a média foi feita nas dimensões lat e lon.

Exemplo1: É feita a média na área da variável precipitação utilizando a média sem ponderação (linha 17), onde é utilizado o valor 1.0 tanto para a latitude quanto para a longitude, e com podenderação (linha 20) utilizando como pesos os valores da latitude (clat).

```
1   ; Nome do script: cap14_ex41.ncl
2
3   begin
4
5   ;                 442   72  144
6   ; float precip ( time, lat, lon )
7
8   f = addfile("../../dados/precip.mon.mean.nc","r")
9
10  lat = f->lat                ; Importação da variável lat.
11  ppt = f->precip             ; Importação da variável precip.
12
13  rad  = 4.0*atan(1.0)/180.0  ; Para radianos => 0.01745329
14  clat = cos(lat*rad)         ; Pesos que serão utilizados.
15
16  ; Média sem ponderação.
17  pptma1 = wgt_areaave_Wrap(ppt,1.0,1.0,0)
18
19  ; Média com ponderação na latitude.
20  pptma2 = wgt_areaave_Wrap(ppt,clat,1.0,0)
21
22  ; Imprime na tela apenas o primeiro tempo (0) => 2.223029
```

```
print( pptma1(0) )

; Imprime na tela apenas o primeiro tempo (0) => 2.703792
print( pptma2(0) )

; Resultado obtido com o GrADS: 2.70379
; Resultado obtido com o CDO: 2.70388

end
```

14.76 Função wind_direction

Calcula a direção do vento meteorológico dado as componentes zonal e meridional do vento. Os argumentos necessários são: componentes zonal e meridional do vento e um terceiro argumento indicando como será retornado o vento calmo, isto é, se a opção for zero, para o vento calmo será retornado o valor zero (direção) e se a opção for 1, será retornado para o vento calmo o valor indefinido.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/wind_direction.shtml

Exemplo:

u = (/ 10, 0, 0, -10, 10, 10, -10, -10, 0/)
v = (/ 0, 10, -10, 0, 10, -10, 10, -10, 0/)
wdir = wind_direction(u,v,0)
print(u+“ ”+v+“ ”+wdir)

Abaixo está o resultado do print com a opção 0, isto é, quando o vento for calmo a direção retornada será zero:

u	v	dir
10	0	270
0	10	180
0	-10	0
-10	0	90
10	10	225
10	-10	315
-10	10	135
-10	-10	45
0	0	0

Quando a opção for 1, o resultado para o vento calmo será indefinido:

```
wdir = wind_direction(u,v,1)
print(u+" "+v+" "+wdir)
```

u	v	dir
10	0	270
0	10	180
0	-10	0
-10	0	90
10	10	225
10	-10	315
-10	10	135
-10	-10	45
0	0	1e+20

15 Criando máscaras

Todos os exemplos para criação de máscaras podem ser encontrados no link abaixo:

http://www.ncl.ucar.edu/Applications/mask.shtml

15.1 Mascarando o continente

Exemplo:

```
; Nome do script: cap15_ex01.ncl

begin

f = addfile("../../dados/prec.2014.nc","r") ; Arquivo de precipitação.

; Arquivo que contém as máscaras. Vem junto com a instalação do NCL.
g = addfile("/usr/local/ncarg/lib/ncarg/data/cdf/landsea.nc","r")

ppt   = f->precip ; lat: [88.75..-88.75]. Essa variável está de norte
                  ; para sul e o mapa sairá invertido. Para resolver
                  ; isso, basta inverter a dimensão latitude da seguinte
                  ; forma "::-1".

; lat: [-88.75..88.75]. Latitude de sul para norte.
ppt   = ppt(:,::-1,:)

; Importação da variável de máscaras do arquivo landsea.nc.
maskv = g->LSMASK

; Opções para mascarar:
; 0 = oceano, 1 = continente, 2 = lagos, 3 = pequenas ilhas e
; 4 = camada de  gelo.

; Cria a máscara.
; Dimensions and sizes: [lat | 72] x [lon | 144]
mascara  = landsea_mask(maskv,ppt&lat,ppt&lon)

; Mascara o dado, isto é, apenas o continente (1).
; Dimensions and sizes: [12] x [72] x [144]
so_oceano = mask(ppt,mascara.eq.1,False)

; A variável so_oceano não tem informações das dimensões e
; coordenadas e ao plotar esse gráfico o resultado estará errado.
; Como essa variável possui as mesmas dimensões de ppt, é possível
; copiar as coordenadas e dimensões de ppt para so_oceano.

copy_VarCoords(ppt,so_oceano)

wks = gsn_open_wks("pdf","../../figuras/cap13/cap13_ex43")

res                  = True     ; Habilita personalização da figura.
res@cnFillOn         = True     ; Habilita gráfico preenchido.
res@cnLinesOn        = False    ; Desabilita as isolinhas.
res@mpLandFillColor  = "white"  ; O mapa com fundo branco.
res@tiMainString     = "Somente oceano" ; Título da figura.

plot = gsn_csm_contour_map(wks,so_oceano(3,:,:),res)

end
```

O resultado será:

15.2 Mascarando valores

Exemplo:

```
; Nome do script: cap15_ex02.ncl

begin

f = addfile("../../dados/prec.2014.nc","r") ; Arquivo de precipitação.

ppt   = f->precip ; lat: [88.75..-88.75] Essa variável está de norte
                  ; para sul e o mapa sairá invertido. Para resolver
                  ; isso, basta inverter a dimensão latitude
                  ; da seguinte forma "::-1".
; Dimensions and sizes: [time | 12] x [lat | 72] x [lon | 144]

ppt   = ppt(:,::-1,:) ; lat: [-88.75..88.75]. Agora a latitude está
                      ; de sul para norte.

; Mascara o dado, isto é, mostra apenas os valores de precipitação
; maiores que 2. Os valores menores que 2 são indefinidos.
; Dimensions and sizes: [12] x [72] x [144]
ppt_mask = mask(ppt,(ppt.gt.2),True)

; A variável ppt_mask não tem informações das dimensões e
; coordenadas e ao plotar esse gráfico o resultado estará errado.
; Como essa variável possui as mesmas dimensões de ppt, é possível
; copiar as coordenadas e dimensões de ppt ppt_mask.

; Dimensions and sizes: [time | 12] x [lat | 72] x [lon | 144]
copy_VarCoords(ppt,ppt_mask)

wks = gsn_open_wks("pdf","../../figuras/cap15/cap15_ex02")

res                  = True             ; Habilita personalização.
res@cnFillOn         = True             ; Habilita gráfico preenchido.
res@cnLinesOn        = False            ; Desabilita as isolinhas.
res@mpLandFillColor  = "white"          ; O mapa com fundo branco.
res@tiMainString     = "Mascarando valores menores que 2 mm/dia"

```

```
plot = gsn_csm_contour_map(wks,ppt_mask(3,:,:),res)

end
```

O resultado será:

16 Conversão entre diferentes tipos de dados

Muitas vezes torna-se necessário realizar a mudança de um tipo de dado para outro, ou até mesmo realizar a coerção (deixar os dados com o mesmo tipo entre eles).

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/type_convert.shtml

Entende-se por coerção a conversão implícita de dados de um tipo (float, integer, double, string, etc) para outro tipo de dado. Isto pode ocorrer durante a atribuição de uma variável de um tipo para uma variável de um tipo diferente, ou quando argumentos que não são do tipo declarado são passados para uma função ou um procedimento, ou quando dois valores de tipos diferentes são operandos do mesmo operador. Operandos devem ser de tipos compatíveis para efetuar a operação desejada.

Exemplo: Multiplicação de um número do tipo float (5.0) por um valor inteiro (2). O valor inteiro é convertido automaticamente para o valor do tipo float e o resultado é um valor do tipo float. Isso ocorre porque qualquer valor inteiro possível está dentro do intervalo de valores do tipo float.

$y = 5.0 \times 2$; $y = 10$.

A Tabela 5 indica quais as conversões podem ocorrer automaticamente. O "sim" indica que o tipo especificado na extremidade esquerda da tabela pode ser convertido automaticamente para o tipo da parte superior da tabela.

Tabela 5: Possibilidades de coerção.

de / para	**float**	**double**	**byte**	**ubyte**	**short**	**ushort**	**integer**	**uint**	**long**	**ulong**	**int64**	**uint64**	**character**	**string**	**logical**
float	-	sim	não	não	não	não	não	não	não	não	não	não	não	sim	sim
double	não	-	não	não	não	não	não	não	não	não	não	não	não	sim	sim
byte	sim	sim	-	sim	sim	sim	sim	sim	sim	sim	sim	sim	sim	sim	sim
ubyte	sim	sim	sim	-	sim	sim	sim	sim	sim	sim	sim	sim	sim	sim	sim
short	sim	sim	não	não	-	sim	sim	sim	sim	sim	sim	sim	não	sim	sim
ushort	sim	sim	não	não	sim	-	sim	sim	sim	sim	sim	sim	não	sim	sim
integer	sim	sim	não	não	não	não	-	sim	sim	sim	sim	sim	não	sim	sim
uint	sim	sim	não	não	não	não	sim	-	sim	sim	sim	sim	não	sim	sim
long	sim	sim	não	não	não	não	não	não	-	sim	sim	sim	não	sim	sim
ulong	sim	sim	não	não	não	não	não	não	sim	-	sim	sim	não	sim	sim
int64	sim	sim	não	não	não	não	não	não	não	não	-	sim	não	sim	sim
uint64	sim	sim	não	não	não	não	não	não	não	não	sim	-	não	sim	sim
character	não	não	não	não	não	não	não	não	não	não	não	não	-	sim	sim
string	não	não	não	não	não	não	não	não	não	não	não	não	não	-	não
logical	não	não	não	não	não	não	não	não	não	não	não	não	não	sim	-

A maioria das funções que realizam conversão entre os diferentes tipos de dados não copiam os seus metadados. A exceção disso são as funções que utilizam a biblioteca contributed:

- byte2flt
- flt2dble
- short2flt
- dble2flt

16.1 Funções para coerção entre dados

Os diferentes tipos de coerção podem ser visto no link abaixo:

http://www.ncl.ucar.edu/Document/Manuals/Ref_Manual/NclDataTypes.shtml#Coercion

16.1.1 Função floattochar

Realiza a coerção de valores do tipo float para o tipo character e qualquer valor fora do limite de valores do tipo character (0 a 255) será retornado valor ausente. Esta função é a mesma que floattocharacter.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/floattochar.shtml

Exemplo:

a = (/ 65, 97, 33.34, -1, 150 /)
print(floattocharacter(a)) ; Resultado: A, a, !, 0x00, 0x96

16.1.2 Função floattoint

Realiza a coerção de valores do tipo float para o tipo integer. O tipo float é truncado na sua parte decimal e qualquer valor fora do limite de valores do tipo integer (-2147483648 a 2147483648) será retornado valor ausente. Esta função é a mesma que floattointeger.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/floattoint.shtml

Exemplo1:

-1.9 é truncado para -1
-1.1 é truncado para -1
2.1 é truncado para 2
2.9 é truncado para 2

Exemplo2:

a = (/ -1.123456712345, -1.91234567891234, 2.823456712345, 22. /)

print(floattoint(a)) ; Resultado: -1, -1, 2, 22

16.1.3 Função integertochar

Realiza a coerção de valores do tipo integer para o tipo character e qualquer valor fora do limite de valores do tipo character (0 a 255) será retornado valor ausente. Esta função é a mesma que integertocharacter e inttochar.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/integertochar.shtml

Exemplo:

a = (/ 65, 97, 33.34, -1, 150 /)
print(integertochar(a)) ; Resultado: A, a, !, 0x00, 0x96

16.2 Funções para conversão entre dados

16.2.1 Função dble2flt

Converte valores do tipo double para valores do tipo float e preserva os metadados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/dble2flt.shtml

Exemplo: Supondo que a variável time é do tipo double e seja necessário convertê-la para o tipo float.

f = addfile("precip.mon.mean.nc","r")
t = dble2flt(f->time)
printVarSummary(t)

Parte do resultado do printVarSummary(t):

Variable: t
Type: float
Total Size: 1768 bytes
442 values
Number of Dimensions: 1

16.2.2 Função flt2dble

Converte valores do tipo float para o tipo double e preserva os metadados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/flt2dble.shtml

Exemplo: Supondo que a variável precip é do tipo float e seja necessário convertê-la para o tipo double.

f = addfile("precip.mon.mean.nc","r")

```
ppt = flt2dble(f->precip)
printVarSummary(ppt)
```

Parte do resultado do printVarSummary(ppt):

```
Variable: ppt
Type: double
Total Size: 36661248 bytes
            4582656 values
Number of Dimensions: 3
```

16.2.3 Função flt2string

Converte valores do tipo float para o tipo string.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/flt2string.shtml

Exemplo:

```
x = fspan(12, 23.5, 10)
x_str = flt2string(x)
print(x_str)
```

Resultado do print(x_str):

```
Variable: x_str
Type: string
Total Size: 80 bytes
            10 values
Number of Dimensions: 1
Dimensions and sizes: [10]
Coordinates:
Number Of Attributes: 0
(0)     12
(1)     13.2778
(2)     14.5556
(3)     15.8333
(4)     17.1111
(5)     18.3889
(6)     19.6667
(7)     20.9444
(8)     22.2222
(9)     23.5
```

16.2.4 Função int2flt

Converte valores do tipo integer para o tipo float.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/int2flt.shtml

Exemplo:

```
i = ispan(1,10,1)
iFlt = int2flt( i )
print( iFlt )
```

Resultado do print(iFlt):

```
Variable: iFlt
Type: float
Total Size: 40 bytes
            10 values
Number of Dimensions: 1
Dimensions and sizes: [10]
```

16.2.5 Função numeric2int

Converte valores de qualquer tipo numérico para o tipo integer. São necessários dois argumentos para essa função, o primeiro será o valor a ser convertido e o segundo possui dois valores, isto é, 0 para truncar e 1 para arredondar o valor.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/numeric2int.shtml

Se o valor for do tipo integer, o valor se mantém inalterado. Se o valor for do tipo float ou double, os valores retornados serão truncados ou arredondados.

Exemplo1:

```
x = 7.2
ix = numeric2int(x,0) ; a opção 0 realiza o truncamento do valor
print(ix) ; Resultado = 7
```

Exemplo2:

```
y = 9.7
iy = numeric2int(y,1) ; a opção 1 realiza o arredondamento do valor
print(iy) ; Resultado = 10
```

16.2.6 Função short2flt

Converte valores do tipo short para o tipo float e preserva os metadados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/short2flt.shtml

```
f = addfile("precip.mon.mean.nc","r")
ppt = short2flt(f->precip)
printVarSummary(ppt)
```

Parte do resultado do printVarSummary(ppt):

Variable: ppt
Type: float
Total Size: 36661248 bytes
4582656 values

16.2.7 Função todouble

Converte valores de qualquer tipo numérico ou string para o tipo double.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/todouble.shtml

Exemplo:

a = (/ 65.2, 97.5, 33.8/)
print(todouble(a)) ; Resultado = 65.19999694824219, 97.5, 33.79999923706055

16.2.8 Função tofloat

Converte valores de qualquer tipo numérico ou string para o tipo float.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/tofloat.shtml

Exemplo:

a = (/ 65., 97., 32.8/)
print(tofloat(a)) ; Resultado = 65, 97, 32.8

16.2.9 Função toint

Converte valores de qualquer tipo numérico ou string para o tipo integer. Os valores são truncados. Essa função é igual a tointeger.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/toint.shtml

Exemplo:

a = (/ 65.8, 97.3, 32.8/)
print(toint(a)) ; Resultado = 65, 97, 32

16.2.10 Função tostring

Converte valores de qualquer tipo numérico para o tipo string.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/tostring.shtml

Exemplo:

a = (/ 65.8, 97.3, 32.8/)
print(tostring(a)) ; Resultado = 65, 97, 32. Será considerado como string.

16.2.11 Função totype

Converte valores de qualquer tipo numérico ou string para qualquer tipo numérico.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/totype.shtml

Exemplo1:

a = (/ 65.8, 97.3, 32.8/)
print(totype(a,"integer")) ; Resultado = 65, 97, 32

Exemplo2:

a = (/ 65.8, 97.3, 32.8/)
print(totype(a,"double")) ; Resultado = 65.80000305175781, 97.30000305175781, 32.79999923706055

17 Interpolação e regrid

17.1 Qual é o melhor método para interpolação/regrid?

A seguir é apresentado um fluxograma sobre o melhor método utilizado para interpolar ou realizar regrid do dado.

Fluxograma para selecionar o melhor método de interpolação/regridding

Fonte: http://www.ncl.ucar.edu/Document/Functions/Images/OLD_NCL_RegriddingDecisionTree.pdf

17.2 Interpolação

O link abaixo apresenta todas as funções para interpolação. Serão mostrados alguns exemplos.

http://www.ncl.ucar.edu/Document/Functions/interp.shtml

17.2.1 Função int2p_Wrap

Interpola os níveis de pressão para um conjunto diferente de novos níveis de pressão e mantém os metadados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/int2p_Wrap.shtml

São necessários 4 argumentos. São eles:

Sintaxe: xo = int2p_Wrap (pi,xi,po,linlog)

1. **pin:** Um arranjo de qualquer dimensionalidade contendo os níveis de pressão. Se o dado de entrada for multidimensional, a dimensão nível vertical deve ser a dimensão mais a direita e os valores devem aumentar ou diminuir monotonicamente.

2. **xin:** Um arranjo de qualquer dimensionalidade contendo o dado a ser interpolado. Deve conter a mesma forma e tamanho de pin.

3. **pout:** Um arranjo de qualquer dimensionalidade contendo os níveis de pressão a serem interpolados com os valores aumentando ou diminuindo monotonicamente.

4. **linlog:** Um escalar indicando o tipo de interpolação. Se o valor for 1, realiza interpolação linear e se for 2, realiza interpolação log. Caso o valor seja negativo, então é feita uma extrapolação dos níveis fora dos valores de pin.

Exemplo:

```
1   ; Nome do script: cap17_ex01.ncl
2
3   begin
4
5   f = addfile("../../dados/uwnd.vwnd.nc","r")
6
7   xiu = short2flt(f->uwnd)
8   xiv = short2flt(f->vwnd)
9
10  printVarSummary(xiu)
11  ; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
12
13  ; Para usar essa função com dados de mais de uma dimensão, é necessário
14  ; que a dimensão level seja a dimensão mais a direita, por isso, é feita
15  ; a reordenação delas.
16
17  ; Reordenação das dimensões.
18
19  ; Dimensão level sendo a dimensão mais a direita.
20  xiu_n = xiu(time|:,lat|:,lon|:,level|:)
21  xiv_n = xiv(time|:,lat|:,lon|:,level|:)
22
23  ; É preciso importar os níveis verticais do arquivo pois eles serão
24  ; utilizados na função. A variável level existe no arquivo.
25
26  ; São os níveis verticais do arquivo: 1000, 925, 850, 700, 600, 500,
27  ; 400, 300
28
29  pi = f->level
30
31  ; É preciso também os novos níveis verticais para realizar a interpolação.
32  ; O dado será interpolado para os níveis verticais abaixo:
33
34  po = (/1000.,950.,925.,850.,800.,750.,700.,650.,600.,550.,500., \
35          400.,300./)
36
37  ; Tipo de interpolação a ser feito. Se o valor for igual a 1, realiza
38  ; interpolação linear e se for diferente de 1, realiza interpolação log.
39  ; Caso o valor seja negativo, ocorre a extrapolação para além dos níveis
40  ; selecionados. Utilize a extrapolação com cuidado.
41
42  linlog = 2 ; Tipo de interpolação.
43
44  ; Interpola u para os novos níveis verticais.
45  xou = int2p_Wrap (pi,xiu_n,po,linlog)
46  ; Interpola v para os novos níveis verticais.
47  xov = int2p_Wrap (pi,xiv_n,po,linlog)
48
49  printVarSummary(xou)
50
51  ; Nota: Na linha 55 são os valores originais (8 níveis verticais)
52  ; da variável e na linha 56 são os valores gerados pela interpolação
53  ; (13 níveis verticais). Foram criados mais 5 níveis verticais.
54
55  ; 1000,      925, 850,           700,      600,      500, 400, 300
56  ; 1000, 950, 925, 850, 800, 750, 700, 650, 600, 550, 500, 400, 300
57
58  end
```

17.2.2 Função poisson_grid_fill

Substitui todos os valores ausentes (_FillValue) em uma grade com valores derivados da equação de Poisson utilizando o método de relaxamento. Os locais onde há valores não-ausentes são utilizados como condições de borda para realizar o preenchimento, e não são alterados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/poisson_grid_fill.shtml

São necessários 7 parâmetros para utilizar essa função. São eles:

Sintaxe: poisson_grid_fill(x,is_cyclic,guess,nscan,epsx,relc,opt)

1. **x:** Um arranjo de duas ou mais dimensões. As duas dimensões mais a direita são usadas na interpolação. Os valoes ausentes devem ser fixados apropriadamente.

2. **is_cyclic:** É uma opção que indica se a dimensão mais a direita de x é ciclíca. Defina como True se x deveria ser tratado como ciclíco ou False, para o caso contrário.

3. **guess_type:** Especifica o tipo de grade inicial. guess_type = 0 significa usar 0.0 como condição inicial; guess_type = 1 significa que a média zonal (média na direção x) será usada. É recomendável que seja utilizado guess_type = 1.

4. **nscan:** O número máximo de iterações a ser usada na relaxação.

5. **epsx:** É a tolerânia para o fim da relaxação antes do limite de nscan.

6. **relc:** relaxação constante: Usualmente 0.45≤relc≤0.6.

7. **opt:** Atualmente não tem uso, mas é definido como 0.

Exemplo:

```
; Nome do script: cap17_ex02.ncl

begin

f = addfile("../../dados/zg_hadgem2_200501.nc","r")

zgeo = f->zg ; float zg ( time, plev, lat, lon )

; [time | 1] x [plev | 17] x [lat | 144] x [lon | 192]
printVarSummary(zgeo)

is_cyclic = True    ; Dado global.
guess     = 1       ; Usa média zonal.
nscan     = 1000    ; Número máximo de iterações a ser usado.
epsx      = 1.0e-2  ; Variável dependente.
relc      = 0.6     ; Coeficiente de relaxação.
opt       = 0       ; Não usado.

```

```
; Preenchimento dos dados ausentes.
poisson_grid_fill(zgeo,is_cyclic,guess,nscan,epsx,relc,opt)

; [time | 1] x [plev | 17] x [lat | 144] x [lon | 192]
printVarSummary(zgeo)

; Em seguida, basta realizar o plot utilizando a variável zgeo.

end
```

O resultado será:

17.3 Regrid

O link abaixo mostra todas as funções para regrid. Serão mostrados apenas alguns exemplos.

http://www.ncl.ucar.edu/Document/Functions/regrid.shtml.

O regrid consiste na interpolação de uma estrututa espacial bem definida (dado em alta resolução) para outra, por exemplo, horizontal ou vertical. A aplicação do regrid de uma baixa para uma alta resolução não fornece resultados mais precisos. O que normalmente se utiliza é a interpolação de uma grade em alta resolução espacial para uma grade de baixa resolução.

17.3.1 Função f2fosh_Wrap

Interpola uma quantidade escalar de uma grade fixa (incluindo os pontos dos polos) para uma grade fixa deslocada utilizando harmônicos esféricos, além disso, os metadados são retidos.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/f2fosh_Wrap.shtml

O argumento de entrada para a função é um arranjo de duas ou mais dimensões em que as duas dimensões mais a direita são a latitude e a longitude. Os valores devem estar na ordem ascendente da latitude (latitude de sul para norte). Caso seja encontrado algum valor ausente em algum ponto, será retornado valor ausente apenas para este ponto. Essa função deve ser utilizada em dados globais e não em dados regionais. A variável retornada terá as mesmas dimensões do dado de entrada, com exceção da latitude que terá um ponto a menos. Por exemplo, um dado com ny = 73 e nx = 144, ao utilizar essa função, o ny será de 72 e o nx permanecerá o mesmo.

Exemplo:

```
1   ; Nome do script: cap17_ex03.ncl
2
3   begin
4
5   f = addfile("../../dados/olr.jan2000.dez2009.nc","r")
6
7   olr = short2flt(f->olr)
8
9   printVarSummary(olr) ; [time | 120] x [lat | 73] x [lon | 144
10                       ; time: [1753152..1840080]
11                       ; lat: [90..-90] => latitude de norte pa
12                       ; lon: [ 0..357.5]
13
14  ; A latitude deve ser de sul para norte ("::-1").
15  olrR = f2fosh_Wrap(olr(:,::-1,:))
16
17  printVarSummary(olrR) ; [time | 120] x [lat | 72] x [lon | 14
18                        ; time: [1753152..1840080]
19                        ; lat: [-88.75..88.75] => latitude de s
20                        ; para norte.
21                        ; lon: [1.25..358.75]
22
23  end
```

17.3.2 Função f2foshv_Wrap

Interpola uma variável vetorial em uma grade fixa (incluindo os polos) para uma grade fixa deslocada utilizando harmônicos esféricos, além disso, os metadados são retidos.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/f2foshv_Wrap.shtml

Essa função deve ser utilizada em dados globais e não em dados regionais.

São necessários quatro argumentos para utilizar essa função:

Sintaxe: z = f2foshv_Wrap(ureg,vreg,uoff,voff)

Onde:

1. **ureg e vreg:** É um arranjo vetorial com duas ou mais dimensões cuja as duas dimensões mais a direita representam a latitude e a longitude. Os valores devem estar na ordem ascendente da latitude, ou seja, de sul para norte.

2. **uoff e voff:** É um arranjo vetorial com duas ou mais dimensões cuja as duas dimensões mais a direita representam a latitude e a longitude. Os valores devem estar na ordem ascendente da latitude, ou seja, de sul para norte. Esses arranjos possuem o mesmo tamanho de ureg e vreg, porém a dimensão latitude possui um ponto a menos. É necessário pré-alocar memória para armazenar as variáveis a serem interpoladas.

A variável retornada terá as mesmas dimensões do dado de entrada, com exceção da latitude que terá um ponto a menos. Por exemplo, um dado com ny = 73 e nx = 144, ao utilizar essa função, o ny será de 72 e o nx permanecerá o mesmo.

Exemplo:

```
1   ; Nome do script: cap17_ex04.ncl
2
3   begin
4
5   f = addfile("../../dados/uwnd.vwnd.nc","r")
6
7   u = short2flt(f->uwnd)
8   v = short2flt(f->vwnd)
9
10  printVarSummary(u) ; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
11                     ; time: [1569072..1570488]
12                     ; level: [1000..100]
13                     ; lat: [90..-90] => latitude de norte para sul.
14                     ; lon: [ 0..357.5]
15
16  printVarSummary(v) ; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
17                     ; time: [1569072..1570488]
18                     ; level: [1000..100]
19                     ; lat: [90..-90] => latitude de norte para sul.
20                     ; lon: [ 0..357.5]
21
22  nt = 3   ; Número de tempos do arquivo.
23  nz = 12  ; Número de níveis verticais do arquivo.
24  ny = 72  ; Número de pontos de latitude do arquivo.
25           ; Lembrando que a latitude terá um ponto a menos do
26           ; dado original que tem 73 pontos.
27  nx = 144 ; Número de pontos de longitude do arquivo.
28
29  ; Cria uma nova variável para alocar a variável interpolada.
30  uN = new((/nt,nz,ny,nx/),typeof(u))
31  vN = new((/nt,nz,ny,nx/),typeof(v))
32
33  ; Não é possível utilizar a função copy_VarCoords porque o número de pontos
34  ; de latitude da variável de entrada não é o mesmo da variável de saída. As
35  ; coordenadas serão criadas na manualmente.
36
```

```
37  ; Copia apenas as duas dimensões mais a esquerda, isto é, time e level.
38  copy_VarCoords_2(u,uN)
39  copy_VarCoords_2(v,vN)
40
41  printVarSummary(uN) ; [time | 3] x [level | 12] x [72] x [144]
42  printVarSummary(vN) ; [time | 3] x [level | 12] x [72] x [144]
43
44  ; Criação manual das coordendas.
45  uN!2         = "lat"
46  uN!3         = "lon"
47  uN&lat       = fspan(-88.75,88.75,ny)
48  uN&lon       = u&lon
49  uN&lat@units = "degrees_north"
50  uN&lon@units = "degrees_east"
51
52  ; Criação manual das coordendas.
53  vN!2         = "lat"
54  vN!3         = "lon"
55  vN&lat       = fspan(-88.75,88.75,ny)
56  vN&lon       = v&lon
57  vN&lat@units = "degrees_north"
58  vN&lon@units = "degrees_east"
59
60  ; A latitude deve ser de sul para norte ("::-1").
61  f2foshv_Wrap(u(:,:,::-1,:),v(:,:,::-1,:),uN,vN)
62
63  printVarSummary(uN) ; [time | 3] x [level | 12] x [lat | 72] x [lon | 144]
64                      ; time: [1569072..1570488]
65                      ; level: [1000..100]
66                      ; lat: [-88.75..88.75]
67                      ; lon: [1.25..358.75]
68
69  printVarSummary(vN) ; [time | 3] x [level | 12] x [lat | 72] x [lon | 144]
70                      ; time: [1569072..1570488]
71                      ; level: [1000..100]
72                      ; lat: [-88.75..88.75]
73                      ; lon: [1.25..358.75]
74
75  end
```

17.3.3 Função f2fsh_Wrap

Interpola uma quantidade escalar de uma grade fixa para outra grade fixa, além disso, os metadados são retidos.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/f2fsh_Wrap.shtml

O argumento de entrada para a função é um arranjo de duas ou mais dimensões em que as duas dimensões mais a direita são a latitude e a longitude. Os valores devem estar na ordem ascendente da latitude (latitude de sul para norte). O outro argumeto, é a resolução que se deseja realizar a interpolação nas dimensões mais a direita. Caso seja encontrado algum valor ausente em algum ponto, será retornado valor ausente apenas para este ponto.

A variável retornada com essa função possui as mesmas dimensões do dado de entrada, exceto que as coordenadas de latitude e de longitude serão substuídas pelos valores do segundo argumento da função, isto é, o número de pontos na direção y e

x, respectivamente.

Essa função deve ser utilizada em dados globais e não em dados regionais.

Exemplo: Interpolando um dado com resolução espacial de 2.5^{o}lat x 2.5^{o}lon (ny=73 e nx=144) para um dado com resolução de 1.0^{o}lat x 1.0^{o}lon.

```
; Nome do script: cap17_ex05.ncl

begin

f = addfile("../../dados/olr.jan2000.dez2009.nc","r")

olr = short2flt(f->olr)

printVarSummary(olr) ; [time | 120] x [lat | 73] x [lon | 144]
                     ; time: [1753152..1840080]
                     ; lat: [90..-90] => latitude de norte para sul.
                     ; lon: [ 0..357.5]

; A latitude deve ser de sul para norte ("::-1").
; 181 pontos na direção y (latitude) e 360 pontos na direção x (longitude).
olrR = f2fsh_Wrap(olr(:,::-1,:),(/181,360/))

printVarSummary(olrR) ; [time | 120] x [lat | 181] x [lon | 360]
                      ; time: [1753152..1840080]
                      ; lat: [-90..90] => latitude de sul para norte.
                      ; lon: [ 0..359]

end
```

O resultado será:

17.3.4 Função f2fshv_Wrap

Interpola uma variável vetorial em uma grade fixa para outra grade fixa utilizando harmônicos esféricos, além disso, os metadados são retidos.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/f2fshv_Wrap.shtml

Essa função deve ser utilizada em dados globais e não em dados regionais.

São necessários quatro argumentos para utilizar essa função:

Sintaxe: z = f2fshv_Wrap(ua,va,ub,vb)

Onde:

1. **ua e va:** É um arranjo vetorial com duas ou mais dimensões cuja as duas dimensões mais a direita representam a latitude e a longitude. Os valores devem estar na ordem ascendente da latitude, ou seja, de sul para norte.

2. **ub e vb:** É um arranjo vetorial com duas ou mais dimensões cuja as duas dimensões mais a direita representam a latitude e a longitude. Os valores devem

estar na ordem ascendente da latitude, ou seja, de sul para norte. É necessário pré-alocar memória para armazenar as variáveis a serem interpoladas.

Exemplo: Interpolando um dado com resolução espacial de 2.5^{o}lat x 2.5^{o}lon (ny=73 e nx=144) para um dado com resolução de 1.0^{o}lat x 1.0^{o}lon (ny=181 e nx=360).

```
1   ; Nome do script: cap17_ex06.ncl
2
3   begin
4
5   f = addfile("../../dados/uwnd.vwnd.nc","r")
6
7   u = short2flt(f->uwnd)
8   v = short2flt(f->vwnd)
9
10  printVarSummary(u) ; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
11                     ; time: [1569072..1570488]
12                     ; level: [1000..100]
13                     ; lat: [90..-90] => latitude de norte para sul.
14                     ; lon: [ 0..357.5]
15
16  printVarSummary(v) ; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
17                     ; time: [1569072..1570488]
18                     ; level: [1000..100]
19                     ; lat: [90..-90] => latitude de norte para sul.
20                     ; lon: [ 0..357.5]
21
22  nt = 3   ; Número de tempos do arquivo.
23  nz = 12  ; Número de níveis verticais do arquivo.
24  ny = 181 ; Número de pontos de latitude do arquivo.
25           ; Lembrando que a latitude terá um ponto a
26           ; menos do dado original que têm 73 pontos.
27  nx = 360 ; Número de pontos de longitude do arquivo.
28
29  ; Cria uma nova variável para alocar a variável interpolada.
30  uN = new((/nt,nz,ny,nx/),typeof(u))
31  vN = new((/nt,nz,ny,nx/),typeof(v))
32
33  ; Não é possível utilizar a função copy_VarCoords porque o número
34  ; de pontos de latitude da variável de entrada não é o mesmo da
35  ; variável de saída. As coordenadas serão criadas manualmente.
36
37  copy_VarCoords_2(u,uN)
38  copy_VarCoords_2(v,vN)
39
40  printVarSummary(uN) ; [time | 3] x [level | 12] x [72] x [144]
41  printVarSummary(vN) ; [time | 3] x [level | 12] x [72] x [144]
42
43  ; Criação manual das coordendas.
44  uN!2         = "lat"
45  uN!3         = "lon"
46  uN&lat       = fspan(-90.,90.,ny)
47  uN&lon       = fspan(0.,359.,nx)
48  uN&lat@units = "degrees_north"
49  uN&lon@units = "degrees_east"
50
51  ; Criação manual das coordendas.
52  vN!2         = "lat"
```

```
vN!3          = "lon"
vN&lat        = fspan(-90.,90.,ny)
vN&lon        = fspan(0.,359.,nx)
vN&lat@units = "degrees_north"
vN&lon@units = "degrees_east"

; A latitude deve ser de sul para norte ("::-1").
f2fshv_Wrap(u(:,:,::-1,:),v(:,:,::-1,:),uN,vN)

printVarSummary(uN) ; [time | 3] x [level | 12] x [lat | 181] x [lon | 360]
                    ; time: [1569072..1570488]
                    ; level: [1000..100]
                    ; lat: [-90..90]
                    ; lon: [ 0..359]

printVarSummary(vN) ; [time | 3] x [level | 12] x [lat | 181] x [lon | 360]
                    ; time: [1569072..1570488]
                    ; level: [1000..100]
                    ; lat: [-90..90]
                    ; lon: [ 0..359]

end
```

O resultado será:

17.3.5 Função fo2fsh_Wrap

Interpola uma quantidade escalar de uma grade fixa deslocada para outra grade fixa, além disso, os metadados são retidos.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/fo2fsh_Wrap.shtml

O argumento de entrada para a função é um arranjo de duas ou mais dimensões em que as duas dimensões mais a direita são a latitude e a longitude. Os valores devem estar na ordem ascendente da latitude (latitude de sul para norte). Caso seja encontrado algum valor ausente em algum ponto, será retornado valor ausente apenas para este ponto.

Essa função deve ser utilizada em dados globais e não em dados regionais.

A variável retornada terá as mesmas dimensões do dado de entrada, com exceção da latitude que terá um ponto a mais. Por exemplo, um dado com ny = 72 e nx = 144, ao utilizar essa função, o ny será de 73 e o nx permanecerá o mesmo.

Exemplo:

```
1   ; Nome do script: cap17_ex07.ncl
2
3   begin
4
5   f = addfile("../../dados/precip.mon.mean.nc","r")
6
7   ppt = f->precip
8
9   printVarSummary(ppt)    ; [time | 442] x [lat | 72] x [lon | 144]
10                          ; time: [65378..78800]
11                          ; lat: [88.75..-88.75]
12                          ; lon: [1.25..358.75]
13
14  pptR = fo2fsh_Wrap(ppt) ; Realiza a interpolação.
15
16  printVarSummary(pptR)   ; [time | 442] x [lat | 73] x [lon | 144]
17                          ; time: [65378..78800]
18                          ; lat: [-90..90]   => Nova latitude interpolada
19                          ; lon: [ 0..357.5] => Nova longitude interpolada
20  end
```

17.3.6 Função fo2fshv_Wrap

Interpola uma variável vetorial em uma grade fixa deslocada para uma grade fixa utilizando harmônicos esféricos, além disso, os metadados são retidos.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/fo2fshv_Wrap.shtml

Essa função deve ser utilizada em dados globais e não em dados regionais.

São necessários quatro argumentos para utilizar essa função:

Sintaxe: z = f2foshv_Wrap(ureg,vreg,uoff,voff)

Onde:

1. **ureg e vreg:** É um arranjo vetorial com duas ou mais dimensões cuja as duas dimensões mais a direita representam a latitude e a longitude. Os valores devem estar na ordem ascendente da latitude, ou seja, de sul para norte.

2. **uoff e voff:** É um arranjo vetorial com duas ou mais dimensões cuja as duas dimensões mais a direita representam a latitude e a longitude. Os valores devem estar na ordem ascendente da latitude, ou seja, de sul para norte. Esses arranjos possuem o mesmo tamanho de ureg e vreg, porém a dimensão latitude possui um ponto a mais. É necessário pré-alocar memória para armazenar as variáveis a serem interpoladas.

A variável retornada terá as mesmas dimensões do dado de entrada, com exceção da latitude que terá um ponto a mais. Por exemplo, um dado com ny = 72 e nx = 144, ao utilizar essa função, o ny será de 73 e o nx permanecerá o mesmo.

Exemplo: Interpolando um dado de uma grade fixa com ny=72 e nx=144 para uma nova grade com ny=73 e nx=144 (o nx permanece inalterado).

```
1   ; Nome do script: cap17_ex08.ncl
2
3   begin
4
5   f = addfile("../../dados/uwnd.vwnd.R1.nc","r")
6
7   u = short2flt(f->uwnd)
8   v = short2flt(f->vwnd)
9
10  printVarSummary(u)
11
12  printVarSummary(v)
13
14  nt = 3   ; Número de tempos do arquivo.
15  nz = 12  ; Número de níveis verticais do arquivo.
16  ny = 73  ; Número de pontos de latitude do arquivo.
17           ; Lembrando que a latitude terá um ponto a mais
18           ; em relação ao dado original que têm 72 pontos.
19  nx = 144 ; Número de pontos de longitude do arquivo.
20
21  ; Cria uma nova variável para alocar a variável interpolada.
22  uN = new((/nt,nz,ny,nx/),typeof(u))
23  vN = new((/nt,nz,ny,nx/),typeof(v))
24
25  ; Não é possível utilizar a função copy_VarCoords porque o número
26  ; de pontos de latitude da variável de entrada não é o mesmo da
27  ; variável de saída.
28  ; As coordenadas serão criadas na manualmente.
29  ; Copia apenas as duas dimensões mais a esquerda, isto é, time e level.
30  copy_VarCoords_2(u,uN)
31  copy_VarCoords_2(v,vN)
```

```
32
33  printVarSummary(uN) ; [time | 3] x [level | 12] x [73] x [144]
34  printVarSummary(vN) ; [time | 3] x [level | 12] x [73] x [144]
35
36  ; Criação manual das coordendas.
37  uN!2         = "lat"
38  uN!3         = "lon"
39  uN&lat       = fspan(-90.,90.,ny)
40  uN&lon       = fspan(0.,357.5,nx)
41  uN&lat@units = "degrees_north"
42  uN&lon@units = "degrees_east"
43
44  ; Criação manual das coordendas.
45  vN!2         = "lat"
46  vN!3         = "lon"
47  vN&lat       = fspan(-90.,90.,ny)
48  vN&lon       = fspan(0.,357.5,nx)
49  vN&lat@units = "degrees_north"
50  vN&lon@units = "degrees_east"
51
52  ; A latitude deve ser de sul para norte ("::-1").
53  fo2fshv_Wrap(u(:,:,::-1,:),v(:,:,::-1,:),uN,vN)
54
55  printVarSummary(uN) ; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
56                      ; time: [1569072..1570488]
57                      ; level: [1000..100]
58                      ; lat: [-90..90] => Latitude com mais um ponto (73)
59                      ; lon: [ 0..357.5]
60
61  printVarSummary(vN) ; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
62                      ; time: [1569072..1570488]
63                      ; level: [1000..100]
64                      ; lat: [-90..90] => Latitude com mais um ponto (73)
65                      ; lon: [ 0..357.5]
66
67  end
```

17.3.7 Função g2fsh_Wrap

Interpola de uma grade gaussiana para uma grade fixa, além disso, os metadados são retidos.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/g2fsh_Wrap.shtml

As rotinas de regrid baseadas em harmônicos esféricos são rápidas e de alta precisão. No entanto, eles não são apropriadas para certas aplicações, em particular, para interpolação de variáveis delimitadas.

Por exemplo, a precipitação é delimitada por zero e umidade relativa do ar é limitado entre 0 e 100. O uso dos interpoladores harmônicos esféricos pode produzir valores que excedam os limites da variável. Isto é devido ao spectral ringing (Fenômeno de Gibbs). Geralmente, os valores excedem os limites por algumas pequenas quantidades.

Isso pode ser resolvido de duas maneiras:

1) Para garantir que os resultados não excedam os limites da variável, o usuário deve usar a função bilinear regrid linint2 (linint2_Wrap).

2) Utilize em conjunto as funções de regrid baseado em harmônicos esféricos e os operadores de recorte do NCL “>” e “<”. Por exemplo, considere uma variável “x”, que é delimitada por 0 e 100:

x = x < 100 ; qualquer valor maior que 100 será fixado em 100
x = x > 0 ; qualquer valor menor que 0 será fixado em 0

Apesar de toda essa explicação, o usuário deve estar ciente das limitações que esta função pode ter. Por exemplo, se a variável a ser interpolada for a precipitação (delimitada por zero) então, os valores em regiões de deserto que originalmente possuem valor zero, com o uso desta função, podem surgir valores diferentes de zero mesmo com o uso do operador > 0.

Exemplo: Interpolando o dado de um MCGA que possui grade gaussiana para uma grade regular com dimensões de 72 x 144 pontos (latxlon).

```
1   ; Nome do script: cap17_ex09.ncl
2
3   begin
4
5   ; Este script interpola os dados de uma grade gaussiana para uma grade
6   ; fixa e retém os metadados. Caso seja encontrado algum valor ausente,
7   ; para este determinado ponto não será feita a interpolação e será
8   ; retornado o valor ausente.
9
10  ; Esta função é utilizada apenas para dados globais e não deve ser
11  ; utilizada para dados regionais.
12
13  ; São necessários dois parâmetros para essa função:
14
15  ; 1) O dado de entrada precisa ser uma variável com 2 ou mais dimensões
16  ; em que as dimensões mais a direita são latitude e longitude.
17  ; Os valores estar devem na ordem ascendente de latitude.
18
19  ; 2) Um arranjo indicando a grade de saída das dimensões mais a direita,
20  ; onde a primeira informação é a latitude e a segunda, a longitude.
21
22  a = addfile("../../dados/MCGA.1985.nc","r")
23
24  prec = a->prec
25
26  printVarSummary(prec)
27
28  ; Resultadod do comando printVarSummary(prec):
29  ;
30  ;Variable: prec
31  ;Type: float
32  ;Total Size: 73728 bytes
33  ;            18432 values
34  ;Number of Dimensions: 3
35  ;Dimensions and sizes:  [time | 1] x [lat | 96] x [lon | 192]
36  ;Coordinates:
37  ;            time: [   0..   0]
38  ;            lat: [88.57217..-88.57217] => latitude de norte para sul
39  ;            lon: [   0..358.125]
40
```

```
; Interpola de uma grade gaussiana para uma grade fixa com as seguintes
; dimensões 72 lat x 144 lon.
; (:,::-1,:) => Inversão da latitude como requisito da função que
; foi explicado acima.

prec_interp = g2fsh_Wrap(prec(:,::-1,:),(/72,144/))

printVarSummary(prec_interp)

; Resultado do comando printVarSummary(prec_interp)
;
;Variable: prec_interp
;Type: float
;Total Size: 41472 bytes
;            10368 values
;Number of Dimensions: 3
;Dimensions and sizes:  [time | 1] x [lat | 72] x [lon | 144]
;Coordinates:
;            time: [   0..   0]
;            lat: [-90..90]      => Latitude de sul para norte
;            lon: [ 0..357.5]

end
```

O resultado será:

17.3.8 Função g2fshv_Wrap

Interpola uma variável vetorial em uma grade gaussiana para uma grade fixa utilizando harmônicos esféricos, além disso, os metadados são retidos.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/g2fshv_Wrap.shtml

Essa função deve ser utilizada em dados globais e não em dados regionais.

São necessários quatro argumentos para utilizar essa função:

Sintaxe: z = f2foshv_Wrap(ua,va,ub,vb)

Onde:

1. **ua e va:** É um arranjo vetorial com duas ou mais dimensões cuja as duas dimensões mais a direita representam a latitude e a longitude. Os valores devem estar na ordem ascendente da latitude, ou seja, de sul para norte.

2. **ub e vb:** É um arranjo vetorial com duas ou mais dimensões cuja as duas dimensões mais a direita representam a latitude e a longitude. Os valores devem estar na ordem ascendente da latitude, ou seja, de sul para norte. É necessário pré-alocar memória para armazenar as variáveis a serem interpoladas.

Exemplo: Interpolando de uma grade gaussiana para uma grade regular com resolução de 2.5^o lat x 2.5^o lon.

```
1   ; Nome do script: cap17_ex10.ncl
2
3   begin
4
5   f = addfile("../../dados/MCGA.1985.nc","r")
6
7   u = f->uvel
8   v = f->vvel
9
10  printVarSummary(u)
11
12  printVarSummary(v)
13
14  nt = 1   ; Número de tempos do arquivo.
15  nz = 18  ; Número de níveis verticais do arquivo.
16  ny = 73  ; Número de pontos de latitude a ser interpolado.
17  nx = 144 ; Número de pontos de longitude a ser interpolado.
18
19  ; Cria uma nova variável para alocar a variável interpolada.
20  ; 73x144 corresponde a uma grade de 2.5° lat x 2.5° lon.
21  ; Essa escolha é feita de acordo com a necessidade do usuário.
22  uN = new((/nt,nz,ny,nx/),typeof(u))
23  vN = new((/nt,nz,ny,nx/),typeof(v))
24
25  ; Não é possível utilizar a função copy_VarCoords porque o número de
26  ; pontos de latitude da variável de entrada não é o mesmo da variável
27  ; de saída. As coordenadas serão criadas na manualmente.
28
```

```
29  ; Copia apenas as duas dimensões mais a esquerda (time e level).
30  copy_VarCoords_2(u,uN)
31  ; Copia apenas as duas dimensões mais a esquerda (time e level).
32  copy_VarCoords_2(v,vN)
33
34  printVarSummary(uN) ; [time | 1] x [lev | 18] x [73] x [144]
35  printVarSummary(vN) ; [time | 1] x [lev | 18] x [73] x [144]
36
37  ; Criação manual das coordendas de latitude.
38  uN!2         = "lat"
39  uN!3         = "lon"
40  uN&lat       = fspan(-90.,90.,ny)
41  uN&lon       = fspan(0.,357.5,nx)
42  uN&lat@units = "degrees_north"
43  uN&lon@units = "degrees_east"
44
45  ; Criação manual das coordendas de latitude.
46  vN!2         = "lat"
47  vN!3         = "lon"
48  vN&lat       = fspan(-90.,90.,ny)
49  vN&lon       = fspan(0.,357.5,nx)
50  vN&lat@units = "degrees_north"
51  vN&lon@units = "degrees_east"
52
53  ; A latitude deve ser de sul para norte ("::-1").
54  g2fshv_Wrap(u(:,:,::-1,:),v(:,:,::-1,:),uN,vN)
55
56  printVarSummary(uN) ; [time | 1] x [lev | 18] x [lat | 73] x [lon | 144]
57                      ; time: [   0..   0]
58                      ; lev: [1000..   3]
59                      ; lat: [-90..90] => Latitude com espaçamento de 2.5°.
60                      ; lon: [ 0..357.5]
61
62  printVarSummary(vN) ; [time | 1] x [lev | 18] x [lat | 73] x [lon | 144]
63                      ; time: [   0..   0]
64                      ; lev: [1000..   3]
65                      ; lat: [-90..90] => Latitude com espaçamento de 2.5°.
66                      ; lon: [ 0..357.5]
67
68  end
```

O resultado será:

17.3.9 Função linint2_Wrap

Interpola de uma grade rectilinear para outro grade rectilinear usando interpolação bilinear, além disso, os metadados são retidos.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Contributed/linint2_Wrap.shtml

São necessários 7 parâmetros para utilizar essa função. A interpolação é feita inicialmente na direção x e depois na y.

Sintaxe: fo = linint2_Wrap (xi,yi,fi, False ou True, xo,yo, 0)

Onde: fo: é a variável a ser interpolada.

1. **xi:** É um arranjo que especifica a coordenada x do dado de entrada. Essa informação é do tipo 1D que aumenta monotonicamente e pode ser igualmente ou não espaçado. Para dados georeferenciados xi é geralmente o arranjo de longitude.

2. **yi:** É um arranjo que especifica a coordenada y do dado de entrada. Essa informação é do tipo 1D que aumenta monotonicamente e pode ser igualmente

ou não espaçado. Para dados georeferenciados yi é geralmente o arranjo de latitude.

3. **fi:** É um arranjo de duas ou mais dimensões, ou seja, é a variável que será interpolada. As dimensões mais a direita devem ser a latitude e a longitude. Se o dado apresenta valores ausentes, é recomendável fixar o atributo fi@_FillValue.

4. Uma opção que indica se a dimensão mais a direita é cíclica ou não. Caso o dado seja global, essa opção deve ser definida como **True**. Por exemplo, se as informações de longitude variam de -179.75 a 179.75 ou 0.5 to 359.5, então essa opção deve ser definida como True.

5. **xo:** É um arranjo unidimensional que especifica as coordenadas x do dado a ser interpolado. Deve aumentar monotonicamente, mas pode ter valores espaçados de forma irregular. Para dados georeferenciados essa informação corresponde ao arranjo longitude.

6. **yo:** É um arranjo unidimensional que especifica as coordenadas y do dado a ser interpolado. Deve aumentar monotonicamente, mas pode ter valores espaçados de forma irregular. Para dados georeferenciados essa informação corresponde ao arranjo de latitude.

7. **foOption:** Atualmente, não tem nenhum uso, e por isso, é definido como zero.

Uma observação é feita no uso desta função. Na linha 46 do exemplo abaixo, o nome das dimensões está em maiúsculo e isso não é um erro e serve para evitar conflito no nome da variável de entrada e na variável interpolada, uma vez, que elas possuem coordenadas diferentes. Isso é resolvido renomeando as dimensões com o símbolo "!". Caso essa renomeação não ocorra, não será possível gerar o gráfico.

Exemplo:

```
1   ; Nome do script: cap17_ex11.ncl
2
3   begin
4
5   a    = addfile("../../dados/trmm.2009.mensal.nc","r")
6   prec = a->r
7
8   ; Como o dado apresenta valores ausentes, é necessário definir
9   ; o atributo para valores ausentes.
10
11  prec@_FillValue = getFillValue(prec) ; Defino o valor ausente do dado.
12                                       ; A função getFillValue guarda
13                                       ; o valor ausente
14                                       ; da variável prec (-9999).
15
16  printVarSummary(prec) ; lat: [-49.875..49.875]
17                        ; lon: [-179.875..179.875]
18                        ; Resolução espacial de 0.25° lat x 0.25° lon.
19
```

```
20  ; Criação das coordenadas de latitude e de longitudes da nova váriavel
21  ; interpolada. A função fspan cria um vetor de valores do tipo float.
22  ; Serão criados 144 valores variando de -178.75 a 178.75. O valor 144
23  ; é o número de pontos do GPCP. O mesmo raciocínio é aplicado para a
24  ; latitude, isto é, serão criados 72 valores de latitude variando
25  ; de -49.875 a 49.875.
26
27  new_lat = fspan(-49.875,49.875,72)
28  new_lon = fspan(-178.75,178.75,144)
29
30  ; Realiza a interpolação bilinear. A variável prec_interp é a variável
31  ; que será interpolada.
32  ; prec&lon: Acessa os valores das coordenadas de longitude. O nome lon é
33  ; a dimensão do arquivo aberto. As dimensões são time, lat e lon. Caso o
34  ; arquivo aberto tenha o nome longitude, altere o nome para
35  ; prec&longitude.
36  ; prec&lat: Acessa os valores das coordenadas de latitude. O mesmo
37  ; raciocínio da longitude se aplica para a latitude.
38  ; prec: É a variável do arquivo que foi aberto.
39  ; True: Porque o dado é global.
40  ; new_lon: São as coordenadas de longitude criadas acima.
41  ; new_lat: São as coordenadas de latitude criadas acima.
42  ; 0: Não tem nenhum uso.
43
44  prec_interp = linint2_Wrap(prec&lon,prec&lat,prec,True,new_lon,new_lat,0)
45
46  printVarSummary(prec_interp) ; [time | 12] x [LAT | 72] x [LON | 144]
47                               ;    0            1             2
48
49  ; O resultado do comando printVarSummary(prec_interp):
50
51  ;Coordinates:
52  ; time: [96432..104448]
53  ; LAT: [-49.875..49.875] => O NCL não gosta do nome LAT em maiúsculo.
54  ;                           Tem que ser lat.
55  ; LON: [-178.75..178.75] => O NCL não gosta do nome LON em maiúsculo.
56  ;                           Tem que ser lon.
57
58  ; Renomeando as dimensões de LON para lon e LAT para lat e atribuindo
59  ; uma unidade para cada uma delas.
60
61  ; A posição 0 é a dimensão tempo.
62  prec_interp!1         = "lat"
63  prec_interp!2         = "lon"
64  prec_interp&lat@units = "degrees_north"
65  prec_interp&lon@units = "degrees_east"
66
67  printVarSummary(prec_interp)
68
69  end
```

O resultado será:

17.3.10 Função rcm2rgrid

Interpola de uma grade curvilinear (por exemplo, RCM, WRF, NARR) para uma grade rectilinear. Valores ausentes são permitidos, mas são ignorados.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/rcm2rgrid.shtml

São necessários 6 parâmetros para utilizar essa função:

Sintaxe: xgrd = rcm2rgrid(lat2d,lon2d,fi,lat,lon,0)

1. **lat2d:** Um arranjo de duas dimensões referente a latitude do dado de entrada (fi). A latitude deve ser de sul para norte.

2. **lon2d:** Um arranjo de duas dimensões referente a longitude do dado de entrada (fi). A longitude deve ser de oeste para leste.

3. **fi:** Um arranjo multidimensional a ser interpolado. As duas dimensões mais a direita (latitude e longitude) são as dimensões a serem interpoladas.

4. **lat:** Um arranjo de uma dimensão que especifica as coordenadas de latitude da grade regular e deve aumentar monotonicamente.

5. **lon:** Um arranjo de uma dimensão que especifica as coordenadas de longitude da grade regular e deve aumentar monotonicamente.

6. **option:** Reservado para uso futuro. Atualmente definido como zero.

Exemplo: Interpolando o dado do modelo WRF para uma grade regular.

```
1   ; Nome do script: cap17_ex12.ncl
2
3   begin
4
5   f = addfile("../../dados/wrfout_d01_2010-03-15_00_00_00","r")
6
7   lat2d  = f->XLAT(0,:,:) ; Importa a variável XLAT. Essa variável será
8                           ; utilizada na interpolação.
9
10  lon2d  = f->XLONG(0,:,:) ; Importa a variável XLON. Essa variável será
11                           ; utilizada na interpolação.
12
13  t2 = wrf_user_getvar(f,"T2",-1) ; float T2(Time, south_north, west_east).
14                                  ; O valor -1 quer dizer para ler todos
15                                  ; os tempos.
16
17  ; float XLAT(Time, south_north, west_east).
18  ; O valor 0 quer dizer para ler apenas o primeiro tempo.
19  t2@lat2d = wrf_user_getvar(f,"XLAT",0)
20
21  ; float XLONG(Time, south_north, west_east).
22  ;O valor 0 quer dizer para ler apenas o primeiro tempo.
23  t2@lon2d = wrf_user_getvar(f,"XLONG",0)
24
25  ; (0)   TEMP at 2 M: min=272.116   max=308.423
26  printMinMax(t2,0) ; Mostra o mínimo e máximo valor da variável t2.
27
28  ; (0)   LATITUDE, SOUTH IS NEGATIVE: min=-19.3722   max=19.3722
29  printMinMax(lat2d,0) ; Mostra o mínimo e máximo valor da variável lat2d.
30
31  ; 0)    LONGITUDE, WEST IS NEGATIVE: min=-88.9592   max=-21.0408
32  printMinMax(lon2d,0) ; Mostra o mínimo e máximo valor da variável lon2d.
33
34  ; Seleção de um domínio global para interpolar o dado.
35  ; Resolução espacial de 2.5° (lat x lon).
36  ; 180°/2.5° = 73 pontos de latitude. 90° graus do HN e 90° do HS = 180°
37  ; 360°/2.5° = 144 pontos de longitude. 360° quer dizer todas as longitudes
38
39  ; Criação das coordenadas para interpolar o dado. Mesmo sendo um dado
40  ; regional, a interpolação é feita em um domínio global e na hora de gerar
41  ; a figura, basta aplicar um zoom na área de interesse. Os dados somente
42  ; serão interpolados onde tiver dados. Essa informação será utilizada como
43  ; nova grade da variável.
44
45  ; Cria um vetor do tipo float de -90 a 90 com 73 valores.
46  lat = fspan(-90,90,73)
47  ; Cria um vetor do tipo float de -178.75 a 178.75 com 144 valores.
48  lon = fspan(-178.75,178.75,144)
49
50  ; Interpola de uma grade curvilinear para uma grade rectilinear de 2.5°x2.
51
52  t2_rect = rcm2rgrid(lat2d,lon2d,t2,lat,lon,0)
53
54  printVarSummary(t2_rect)
55
56  ; Criação das dimensões e coordendas da variável t2_rect.
57  t2_rect!0         = "time"
58  t2_rect!1         = "lat"
59  t2_rect!2         = "lon"
60  t2_rect&lat       = lat
```

```
t2_rect&lon       = lon
t2_rect&lat@units = "degrees_north"
t2_rect&lon@units = "degrees_east"

printVarSummary(t2_rect)

end
```

O resultado será:

18 Criando painéis

O objetivo deste tópico consiste em criar painéis e como personalizá-los. Informações mais detalhadas podem ser obtidas no link abaixo.

http://www.ncl.ucar.edu/Applications/panel.shtml.

Serão criados os scripts e em seguida serão mostrados os resultados na forma de figura. Com o avanço das informações serão adicionadas novas linhas aos scripts.

Abaixo é apresentado um script para criação de um painel com dimensão 1x4 (1 linha e 4 colunas). As linhas serão comentadas para o melhor entendimento.

- Linhas 5 a 8: Abertura dos arquivos com a função addfile apenas para leitura ("r").
- Lnhas 10 a 13: Declaração dos nomes das variáveis.
- Linha 15: Formato (pdf) e o nome do arquivo de saída (cap18_ex01).
- Linhas 17 a 24: Configuração dos recursos para personalizar o gráfico (linha 17, res = True). O recurso gsnDraw diz para não avançar o frame e o gsnFrame não cria novas páginas. Esses comandos são importantes na criação de painéis. O recurso gsnAddCyclic é utilizado porque o dado não é global, é apenas regional. Sem esse recurso o NCL mostrará um aviso.
- Linhas 26 a 29: Correspondem à geração das figuras.
- Linha 32: Esta é a parte responsável pela criação do painel, por meio da função gsn_panel. A disposição das figuras é dada pelo argumento (/1,4/) e como não é feita nenhuma personalização, é declarado False no último argumento da função.

```
; Nome do script: cap18_ex01.ncl

begin

f1 = addfile("../../dados/psfc.nc","r")
f2 = addfile("../../dados/shum.nc","r")
f3 = addfile("../../dados/uwnd.nc","r")
f4 = addfile("../../dados/vwnd.nc","r")

psfc = f1->pres(0,:,:)
qesp = short2flt(f2->shum(0,0,:,:))
uwnd = short2flt(f3->uwnd(0,0,:,:))
vwnd = short2flt(f4->vwnd(0,0,:,:))

wks = gsn_open_wks("pdf","../../figuras/cap18/cap18_ex01")

res                = True  ; Habilita a personalização do gráfico.
res@gsnDraw        = False ; Não desenha a figura.
res@gsnFrame       = False ; Não avança o frame.
res@mpMinLatF      = -60.0 ; Define a
res@mpMaxLatF      =  15.0 ; área de
res@mpMinLonF      = -90.0 ; interesse.
res@mpMaxLonF      = -30.0 ;
res@gsnAddCyclic   = False ; Não adiciona ponto cíclico.

plot1 = gsn_csm_contour_map_ce(wks,psfc,res)
plot2 = gsn_csm_contour_map_ce(wks,qesp,res)
plot3 = gsn_csm_contour_map_ce(wks,uwnd,res)
plot4 = gsn_csm_contour_map_ce(wks,vwnd,res)

; Cria o painel no formato 1 linha com 4 colunas.
gsn_panel(wks,(/plot1,plot2,plot3,plot4/),(/1,4/),False)

end
```

O resultado será:

O mesmo script da figura acima é feito, alterando apenas a disposição das figuras que antes tinham dimensões de 1x4 na função gsn_panel, e agora foi alterado para 4x1.

```
; Nome do script: cap18_ex02.ncl

begin

f1 = addfile("../../dados/psfc.nc","r")
f2 = addfile("../../dados/shum.nc","r")
f3 = addfile("../../dados/uwnd.nc","r")
f4 = addfile("../../dados/vwnd.nc","r")

psfc = f1->pres(0,:,:)
qesp = short2flt(f2->shum(0,0,:,:))
uwnd = short2flt(f3->uwnd(0,0,:,:))
vwnd = short2flt(f4->vwnd(0,0,:,:))

wks = gsn_open_wks("pdf","../../figuras/cap18/cap18_ex02")

; Criando uma variável do tipo graphic para armazenar as figuras.
plot = new(4,graphic)

res                 = True
res@gsnDraw         = False
res@gsnFrame        = False
res@mpMinLatF       = -60.0
res@mpMaxLatF       =  15.0
res@mpMinLonF       = -90.0
res@mpMaxLonF       = -30.0
res@gsnAddCyclic    = False
res@mpFillOn        = False

plot(0) = gsn_csm_contour_map_ce(wks,psfc,res)
plot(1) = gsn_csm_contour_map_ce(wks,qesp,res)
plot(2) = gsn_csm_contour_map_ce(wks,uwnd,res)
plot(3) = gsn_csm_contour_map_ce(wks,vwnd,res)

gsn_panel(wks,plot,(/4,1/),False)

end
```

O resultado será:

Monthly Mean of Surface Pressure
mbars
0
30S
60S
80W
60W
40W
CONTOUR FROM 640 TO 1080 BY 40
Monthly Mean of Specific Humidity
grams/kg
0
30S
60S
80W
60W
40W
CONTOUR FROM 4 TO 22 BY 2
Monthly mean u wind
m/s
0
30S
60S
80W
60W
40W
CONTOUR FROM -8 TO 6 BY 2
Monthly mean v wind
m/s
0
30S
60S
80W
60W
40W
CONTOUR FROM -7 TO 8 BY 1

O script abaixo é o mesmo da figura anterior, alterando apenas a disposição das figuras que antes era de 4x1 na função gsn_panel e agora é de 2x2 (4 figuras, 2 na parte superior e 2 na parte inferior).

```
; Nome do script: cap18_ex03.ncl

begin

f1 = addfile("../../dados/psfc.nc","r")
f2 = addfile("../../dados/shum.nc","r")
f3 = addfile("../../dados/uwnd.nc","r")
f4 = addfile("../../dados/vwnd.nc","r")

psfc = f1->pres(0,:,:)
qesp = short2flt(f2->shum(0,0,:,:))
uwnd = short2flt(f3->uwnd(0,0,:,:))
vwnd = short2flt(f4->vwnd(0,0,:,:))

wks = gsn_open_wks("pdf","../../figuras/cap18/cap18_ex03")

plot = new(4,graphic)

res                 = True
res@gsnDraw         = False
res@gsnFrame        = False
res@mpMinLatF       = -60.0
res@mpMaxLatF       =  15.0
res@mpMinLonF       = -90.0
res@mpMaxLonF       = -30.0
res@gsnAddCyclic    = False
res@mpFillOn        = False

plot(0) = gsn_csm_contour_map_ce(wks,psfc,res)
plot(1) = gsn_csm_contour_map_ce(wks,qesp,res)
plot(2) = gsn_csm_contour_map_ce(wks,uwnd,res)
plot(3) = gsn_csm_contour_map_ce(wks,vwnd,res)

gsn_panel(wks,plot,(/2,2/),False)

end
```

O resultado será:

0
30S
60S
80W
60W
40W
960
CONTOUR FROM 640 TO 1080 BY 40
16
CONTOUR FROM 4 TO 22 BY 2
Monthly mean u wind
m/s
CONTOUR FROM -8 TO 6 BY 2
Monthly mean v wind
m/s
CONTOUR FROM -7 TO 8 BY 1

O script abaixo mostra a criação de painéis complexos. Nota-se a criação de um recurso para personalização dos painéis (linha 34, pres = True). Com isso habilitado, é possível personalizá-los. Para a criação de painéis complexos utiliza-se o recurso gsnPanelRowSpec (linha 35) na parte do script referente a personalização dos painéis conforme a função gsn_panel.

```
1   ; Nome do script: cap18_ex04.ncl
2
3   begin
4
5   f1 = addfile("../../dados/psfc.nc","r")
6   f2 = addfile("../../dados/shum.nc","r")
7   f3 = addfile("../../dados/uwnd.nc","r")
8   f4 = addfile("../../dados/vwnd.nc","r")
9
10  psfc = f1->pres(0,:,:)
11  qesp = short2flt(f2->shum(0,0,:,:))
12  uwnd = short2flt(f3->uwnd(0,0,:,:))
13  vwnd = short2flt(f4->vwnd(0,0,:,:))
14
15  wks = gsn_open_wks("pdf","../../figuras/cap18/cap18_ex04")
16
17  plot = new(4,graphic)
18
19  res                 = True
20  res@gsnDraw         = False
21  res@gsnFrame        = False
22  res@mpMinLatF       = -60.0
23  res@mpMaxLatF       =  15.0
24  res@mpMinLonF       = -90.0
25  res@mpMaxLonF       = -30.0
26  res@gsnAddCyclic    = False
27  res@mpFillOn        = False
28
29  plot(0) = gsn_csm_contour_map_ce(wks,psfc,res)
30  plot(1) = gsn_csm_contour_map_ce(wks,qesp,res)
31  plot(2) = gsn_csm_contour_map_ce(wks,uwnd,res)
32  plot(3) = gsn_csm_contour_map_ce(wks,vwnd,res)
33
34  pres                  = True
35  pres@gsnPanelRowSpec = True
36
37  gsn_panel(wks,plot,(/1,2,1/),pres)
38
39  end
```

O resultado será:

Monthly Mean of Surface Pressure
millibars
0
30S
60S
80W
60W
40W
CONTOUR FROM 640 TO 1080 BY 40
Monthly Mean of Specific Humidity
grams/kg
CONTOUR FROM 4 TO 22 BY 2
Monthly mean u wind
m/s
CONTOUR FROM -8 TO 6 BY 2
Monthly mean v wind
m/s
CONTOUR FROM -7 TO 8 BY 1

Nesse script foi alterada a disposição das figuras que antes era (/1,2,1/) na função gsn_panel para (/2,1,1/).

```
; Nome do script: cap18_ex05.ncl

begin

f1 = addfile("../../dados/psfc.nc","r")
f2 = addfile("../../dados/shum.nc","r")
f3 = addfile("../../dados/uwnd.nc","r")
f4 = addfile("../../dados/vwnd.nc","r")

psfc = f1->pres(0,:,:)
qesp = short2flt(f2->shum(0,0,:,:))
uwnd = short2flt(f3->uwnd(0,0,:,:))
vwnd = short2flt(f4->vwnd(0,0,:,:))

wks = gsn_open_wks("pdf","../../figuras/cap18/cap18_ex05")

plot = new(4,graphic)

res                 = True
res@gsnDraw         = False
res@gsnFrame        = False
res@mpMinLatF       = -60.0
res@mpMaxLatF       =  15.0
res@mpMinLonF       = -90.0
res@mpMaxLonF       = -30.0
res@gsnAddCyclic    = False
res@mpFillOn        = False

plot(0) = gsn_csm_contour_map_ce(wks,psfc,res)
plot(1) = gsn_csm_contour_map_ce(wks,qesp,res)
plot(2) = gsn_csm_contour_map_ce(wks,uwnd,res)
plot(3) = gsn_csm_contour_map_ce(wks,vwnd,res)

pres                  = True
pres@gsnPanelRowSpec  = True

gsn_panel(wks,plot,(/2,1,1/),pres)

end
```

O resultado será:

Monthly Mean of Surface Pressure
millibars
0
30S
60S
80W
60W
40W
CONTOUR FROM 640 TO 1080 BY 40
Monthly Mean of Specific Humidity
grams/kg
CONTOUR FROM 4 TO 22 BY 2
Monthly mean u wind
m/s
CONTOUR FROM -8 TO 6 BY 2
Monthly mean v wind
m/s
CONTOUR FROM -7 TO 8 BY 1

No script abaixo foi inserido um título (linha 37, txString) principal para os painéis como também a identificação das figuras (linha 36, gsnPanelFigureStrings) por meio das letras a, b, c e d. Outra informação inserida foi um comando para aumentar o espaçamento entre as figuras (linha 38, espaçamento na direção y, gsnPanelYWhiteSpacePercent e linha 39, espaçamento na direção x, gsnPanelXWhiteSpacePercent).

```
1   ; Nome do script: cap18_ex06.ncl
2
3   begin
4
5   f1 = addfile("../../dados/psfc.nc","r")
6   f2 = addfile("../../dados/shum.nc","r")
7   f3 = addfile("../../dados/uwnd.nc","r")
8   f4 = addfile("../../dados/vwnd.nc","r")
9
10  psfc = f1->pres(0,:,:)
11  qesp = short2flt(f2->shum(0,0,:,:))
12  uwnd = short2flt(f3->uwnd(0,0,:,:))
13  vwnd = short2flt(f4->vwnd(0,0,:,:))
14
15  wks = gsn_open_wks("pdf","../../figuras/cap18/cap18_ex06")
16
17  plot = new(4,graphic)
18
19  res               = True
20  res@gsnDraw       = False
21  res@gsnFrame      = False
22  res@mpMinLatF     = -60.0
23  res@mpMaxLatF     =  15.0
24  res@mpMinLonF     = -90.0
25  res@mpMaxLonF     = -30.0
26  res@gsnAddCyclic  = False
27  res@cnInfoLabelOn = False
28  res@mpFillOn      = False
29
30  plot(0) = gsn_csm_contour_map_ce(wks,psfc,res)
31  plot(1) = gsn_csm_contour_map_ce(wks,qesp,res)
32  plot(2) = gsn_csm_contour_map_ce(wks,uwnd,res)
33  plot(3) = gsn_csm_contour_map_ce(wks,vwnd,res)
34
35  pres                            = True
36  pres@gsnPanelFigureStrings      = (/"a)","b)","c)","d)"/)
37  pres@txString                   = "Titulo principal do painel"
38  pres@gsnPanelYWhiteSpacePercent = 10
39  pres@gsnPanelXWhiteSpacePercent = 10
40
41  gsn_panel(wks,plot,(/2,2/),pres)
42
43  end
```

O resultado será:

Titulo principal do painel
Monthly Mean of Surface Pressure
millibars
Monthly Mean of Specific Humidity
grams/kg
Monthly mean u wind
m/s
Monthly mean v wind
m/s
a)
b)
c)
d)
0
30S
60S
80W
60W
40W

O gráfico foi alterado para o tipo shaded (linha 24, cnFillOn) pois o objetivo é criar apenas uma escala de cores comum para todos os gráfico. Nas linhas 27 a 30 defini-se a escala de valores dos contornos para as duas variáveis. Também foi inserido um recurso para alterar a posição das letras que identificam as figuras (linha 39, amJust) como também foi removida a borda dessas letras (linha 40, gsnPanelFigureStrings-PerimOn).

```
1   ; Nome do script: cap18_ex07.ncl
2
3   begin
4
5   f1 = addfile("../../dados/uwnd.nc","r")
6   f2 = addfile("../../dados/vwnd.nc","r")
7
8   uwnd = short2flt(f1->uwnd(0,0,:,:))
9   vwnd = short2flt(f2->vwnd(0,0,:,:))
10
11  wks = gsn_open_wks("pdf","../../figuras/cap18/cap18_ex07")
12
13  plot = new(2,graphic)
14
15  res                       = True
16  res@gsnDraw               = False
17  res@gsnFrame              = False
18  res@mpMinLatF             = -60.0
19  res@mpMaxLatF             =  15.0
20  res@mpMinLonF             = -90.0
21  res@mpMaxLonF             = -30.0
22  res@gsnAddCyclic          = False
23  res@cnInfoLabelOn         = False
24  res@cnFillOn              = True
25  res@lbLabelBarOn          = False
26  res@mpFillOn              = False
27  res@cnLevelSelectionMode  = "ManualLevels"
28  res@cnMinLevelValF        = -8.0
29  res@cnMaxLevelValF        = 8.0
30  res@cnLevelSpacingF       = 1.0
31
32  plot(0) = gsn_csm_contour_map_ce(wks,uwnd,res)
33  plot(1) = gsn_csm_contour_map_ce(wks,vwnd,res)
34
35  pres                                = True
36  pres@gsnPanelFigureStrings          = (/"a)","b)"/)
37  pres@txString                       = "Titulo principal do painel"
38  pres@gsnPanelLabelBar               = True
39  pres@amJust                         = "TopRight"
40  pres@gsnPanelFigureStringsPerimOn   = False
41
42  gsn_panel(wks,plot,(/1,2/),pres)
43
44  end
```

O resultado será:

Titulo principal do painel
Monthly mean u wind
m/s
a)
Monthly mean v wind
m/s
b)
0
30S
60S
80W
60W
40W
-8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8

19 Interpretação de erros

Este capítulo mostrará como interpretar erros. É muito comum retornar algum tipo de mensagem de erro ao escrever um script. Para isso, basta interpretar a mensagem pois são fornecidas informações como: a linha onde está o erro, qual o tipo de erro e em algumas situações, o próprio NCL sugere a solução para o problema.

O link abaixo fornece diferentes soluções para os mais variados erros:

http://www.ncl.ucar.edu/Document/Language/error_messages.shtml

Serão mostrados apenas alguns exemplos de forma que o usuário entenda como é feita a interpretação do erro. Caso não seja possível resolver o problema, envie um e-mail para a lista de discussão do NCL, mas antes de tudo, tente entender o erro.

As soluções para os erros mais comuns são encontradas em:

http://www.ncl.ucar.edu/Document/Language/error_messages.shtml

https://www.ncl.ucar.edu/Document/Graphics/error_msg.shtml

Para inspecionar o seu script com o objetivo de identificar erros, faça uso do print e printVaSummary.

Conforme novas linhas são adicionadas ao script, utilize o print ou printVaSummary para verificar se tudo está correto.

Exemplo1: Erro na abertura de um arquivo. Ao abrir um arquivo são necessários dois argumentos, o nome do arquivo e como ele será lido, essas duas informações devem estar entre aspas duplas, neste caso será lido como leitura ("r").

```
1   ; Nome do script: cap19_ex01.ncl
2
3   begin
4
5   f = addfile("../../dados/prec.2005.nc",)
6
7   ppt = f->pre
8
9   wks = gsn_open_wks("pdf","../../figuras/cap19/cap19_ex02")
10
11  plot=gsn_csm_y(wks,ppt,False)
12
13  end
```

Ao executar o script acima, será retornado o seguinte erro:

```
fatal:syntax error: line 5 in file cap19_ex01.ncl
near )
f = addfile("../../dados/prec.2005.nc",)
----------------------------------------^

fatal:Syntax Error in block, block not executed
fatal:error at line 13 in file cap19_ex01.ncl
```

Como foi mencionado, são necessários dois argumentos para abrir o arquivo, e nesse

caso, está faltando como ele será lido, isto é, o "r".

Solução: f = addfile("prec.2005.nc","r")

Exemplo2: Utilizando o mesmo arquivo do Exemplo1. Nesse caso, foi utilizada uma variável chamada "res" que não foi declarada anteriormente. Neste caso, é a variável para personalização da figura (res), e possui o valor lógico True ou False.

```
1   ; Nome do script: cap19_ex02.ncl
2
3   begin
4
5   f = addfile("../../dados/prec.2005.nc","r")
6
7   ppt = f->pre
8
9   wks = gsn_open_wks("pdf","../../figuras/cap19/cap19_ex02")
10
11  plot=gsn_csm_y(wks,ppt,res)
12
13  end
```

Ao executar o script será retornada a mensagem de erro abaixo:

```
fatal:Variable (res) is undefined
fatal:["Execute.c":8575]:Execute: Error occurred at
or near line 11 in file cap19_ex02.ncl
```

O que aconteceu? A variável "res" é um argumento da função gsn_csm_y que precisa ser declarada antes como True ou False. Por isso, o erro.

Solução: Declarar **res = True** ou **res = False** depois da linha 9

Exemplo3: Utilizando o mesmo arquivo do Exemplo1. O gráfico a ser gerado é do tipo linha e o dado de entrada é uma série temporal e possui 3 dimensões (time, lat e lon).

```
1   ; Nome do script: cap19_ex03.ncl
2
3   begin
4
5   f = addfile("../../dados/prec.2005.nc","r")
6
7   ppt = f->pre
8
9   wks = gsn_open_wks("pdf","../../figuras/cap19/cap19_ex03")
10
11  res = True
12
13  plot=gsn_csm_y(wks,ppt,res)
14
15  end
```

Ao executar o script será retornada a mensagem de erro abaixo:

```
(0)     Error: gsn_csm_y: The input Y array must either be 1-dimensional,
 or 2-dimensional, where the leftmost dimension represents the number of
curves and the rightmost dimension the number of points in each curve.
```

O que aconteceu? A variável ppt na linha 5 possui 3 dimensões, isto é, 12 tempos, 1 ponto de latitude e 1 ponto de longitude, logo é preciso fixar as dimensões latitude e longitude de forma que apenas a dimensão time varie.

Solução: plot=gsn_csm_y(wks,ppt(:,0,0),res)

Exemplo4: Utilizando o mesmo arquivo do Exemplo1. O gráfico a ser gerado é do tipo linha e o dado de entrada é uma série temporal e possui 3 dimensões (time, lat e lon). Foi declarada a criação do eixo x como sendo um vetor com 15 posições.

```
1   ; Nome do script: cap19_ex04.ncl
2
3   begin
4
5   f = addfile("../../dados/prec.2005.nc","r")
6
7   ppt = f->pre ; Importação da variável.
8
9   wks = gsn_open_wks("pdf","../../figuras/cap19/cap19_ex04")
10
11  x =ispan(1,15,1) ; Definindo o eixo x do gráfico.
12                   ; Vetor com 15 posições.
13
14  res = True
15
16  plot = gsn_csm_xy(wks,x,ppt(:,0,0),res)
17
18  end
```

Ao executar o script será retornada a mensagem de erro abaixo:

```
(0)     gsn_csm_xy: Fatal: X and Y must have the same
 dimensions sizes, or one must be one-dimensional and
 both have the same rightmost dimension.
```

O que aconteceu? A variável ppt na linha 5 possui 3 dimensões, isto é, 12 tempos, 1 ponto de latitude e 1 ponto de longitude, logo é preciso fixar as dimensões latitude e longitude de forma que apenas a dimensão time varie. Além disso, foi criada uma nova variável x que contém 15 valores, e aí está o problema porque a variável ppt só tem 12 valores, e por isso, a mensagem de erro.

Solução:

x =ispan(1,12,1) ; Definindo o eixo x do gráfico. Vetor com 12 posições na linha 11 do script. Com isso, o erro sumirá.

Exemplo5: O dado utilizado possui as seguintes dimensões: time = 442, lat = 72 e lon = 144. É um dado espacial e para plotar um dado espacial, isto é, um mapa lat/lon é necessário fixar a dimensão tempo e variar as outras dimensões.

```
1   ; Nome do script: cap19_ex05.ncl
2
3   begin ; Início do script.
4
5   f = addfile("../../dados/precip.mon.mean.nc","r")
6
7   ppt = f->precip
8
9   wks = gsn_open_wks("pdf","../../figuras/cap19_ex05")
10
11  plot = gsn_csm_contour_map_ce(wks,ppt,False)
12
13  end
```

Ao executar o script será retornada a mensagem de erro abaixo. O erro ocorre na linha 11 do script como mostrado na mensagem abaixo:

```
(0)     gsn_csm_contour_map_ce: Fatal: the input data array
 must be 1D or 2D
fatal:Illegal right-hand side type for assignment
fatal:["Execute.c":8575]:Execute: Error occurred at or near
 line 11 in file cap19_ex05.ncl
```

O que aconteceu? Para gerar uma figura espacial, neste caso onde há as dimensões time, lat e lon é necessário fixar a dimensão time e variar as demais dimensões, caso contrário, será retornada a mensagem de erro acima. Para resolver esse erro, defini-se o tempo 2 (seria o terceiro tempo da variável, lembrando que no NCL o primeiro índice começa em 0 e não em 1) e lê-se todas os pontos de latitude e longitude da variável.

Solução:

plot = gsn_csm_contour_map_ce(wks,ppt(2,:,:),False)

Exemplo6: O dado utilizado possui as seguintes dimensões: time = 442, lat = 72 e lon = 144. É um dado espacial e para plotar um dado espacial, isto é, um mapa lat/lon é necessário fixar a dimensão tempo e variar as outras dimensões.

```
1   ; Nome do script: cap19_ex06.ncl
2
3   begin
4
5   f = addfile("../../dados/precip.mon.mean.nc","r")
6
7   ppt = f->precip
8
9   pptN = ppt * 30 ; mm/dia -> mm/mes
10
11  printVarSummary(pptN)
12
13  wks = gsn_open_wks("pdf","../../figuras/cap19_ex06")
14
15  plot = gsn_csm_contour_map_ce(wks,pptN(6,:,:),False)
16
17  end
```

Ao executar o script será retornada a mensagem de erro abaixo.

```
  _FillValue :  -9.96921e+36
(0)     check_for_y_lat_coord: Warning: Data either does not contain a v
alid latitude coordinate array or doesn't contain one at all.
(0)     A valid latitude coordinate array should have a 'units' attribut
e equal to one of the following values:
(0)         'degrees_north' 'degrees-north' 'degree_north' 'degrees nort
h' 'degrees_N' 'Degrees_north' 'degree_N' 'degreeN' 'degreesN' 'deg nort
h'
(0)     check_for_lon_coord: Warning: Data either does not contain a val
id longitude coordinate array or doesn't contain one at all.
(0)     A valid longitude coordinate array should have a 'units' attribu
te equal to one of the following values:
(0)         'degrees_east' 'degrees-east' 'degree_east' 'degrees east' '
degrees_E' 'Degrees_east' 'degree_E' 'degreeE' 'degreesE' 'deg east'
```

O que aconteceu? Como o dado foi multiplicado por 30 e os valores atribuídos a uma nova variável pptN, houve a perda das informações das coordenadas e dos nomes das dimensões como pode ser verificado no resultado do printVaSummary(pptN) acima. A solução é criar as coordenadas e os nomes das dimensões latitude e longitude na variável pptN.

Solução:

```
; Nome do script: cap19_ex07.ncl

begin

f = addfile("../../dados/precip.mon.mean.nc","r")

ppt = f->precip

pptN = ppt * 30 ; mm/dia -> mm/mes

printVarSummary(pptN)

; Criação das coordenadas das dimensões latitude e longitude.
pptN!1          = "lat"
pptN!2          = "lon"
pptN&lat        = latGlobeF(72,"lat","latitude","degrees_north")
pptN&lon        = lonGlobeF(144,"lon","longitude","degrees_east")

wks = gsn_open_wks("pdf","../../figuras/cap19/cap19_ex07")

plot = gsn_csm_contour_map_ce(wks,pptN(6,:,:),False)

end
```

20 Técnicas de plot

20.1 Posicionamento da figura na página

O NCL gera figuras em unidade quadrada que varia de 0 a 1. A caixa que contém a figura (sem os rótulos e os tickmarks fora dela) é chamada de viewport (janela de visualização).

As dimensões da figura são alteradas com os recursos vpWidthF e vpHeightF e o seu posicionamento fica no canto superior esquerdo da janela gráfica e para alterar sua posição, utiliza-se o vpXF e vpYF. Outras partes da figura serão redimensionadas automaticamente.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Graphics/Resources/vp.shtml

Exemplo: O script abaixo gera uma figura sem personalização. O objetivo é mostrar o uso dos recursos para posicionamento (vpXF e vpYF), altura (vpHeightF) e largura (vpWidthF) da figura.

```
1   ; Nome do script: cap20_ex01.ncl
2
3   begin
4
5   f = addfile("../../dados/ppt.clima.nc","r")
6
7   ppt = f->precip
8
9   wks = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex01")
10
11  drawNDCGrid(wks) ; Desenha as linhas de grade para selecionar
12                   ; o local  da janela gráfica para mostrar
13                   ; a figura.
14
15  plot = gsn_csm_y(wks,ppt(:,0,0),False)
16
17  end
```

Ao executar o script, o resultado será mostrado abaixo sem as setas e as cores em azul, laranja e vermelho que foram inseridas para facilitar o entendimento dos recursos:

Explicação das informações do gráfico:

- A contagem da unidade na direção x é feita da esquerda para a direita. No y é feita de baixo para cima. Os valores variam de 0.0 a 1.0.
- Por padrão, a figura é gerada no ponto x=0.2 e y=0.8 como é demarcado na figura. O gráfico é gerado apartir deste ponto que pode ter a sua posição alterada por meio dos recursos vpXF e vpYF.
- A altura (vpHeightF) representada pela seta azul corresponde a 0.6 porque ao olhar no eixo y, o gráfico começa em 0.2 e termina em 0.8 o que resulta em 0.6.
- A largura (vpWidthF) representada pela seta laranja corresponde também a 0.6 porque ao olhar no eixo x, o gráfico começa em 0.2 e termina em 0.8 o que resulta em 0.6.
- A caixa vermelha que envolve o gráfico é o viewport (janela de visualização).

No exemplo abaixo foi selecionada outra posição para gerar a figura, e além disso, alterou-se a sua largura e tamanho. Neste caso, a nova posição corresponde aos valores x=0.2 e y=0.6.

```
; Nome do script: cap20_ex02.ncl

begin

f = addfile("../../dados/ppt.clima.nc","r")

ppt = f->precip

wks = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex02")

drawNDCGrid(wks) ; Desenha as linhas de grade para selecionar o local
                 ; da janela gráfica para mostrar a figura.

res           = True
res@vpWidthF  = 0.7   ; Largura da figura.
res@vpHeightF = 0.4   ; Altura da figura.
res@vpXF      = 0.2   ; Posição x onde será desenhada a figura.
res@vpYF      = 0.6   ; Posição y onde será desenhada a figura.

plot = gsn_csm_y(wks,ppt(:,0,0),res)

end
```

Ao executar o script, o resultado será a figura abaixo:

20.2 Explicação sobre o recurso frame

Por padrão, as funções para geração de figura criam o plot e depois avança o frame. Entende-se o frame como uma página, por exemplo, de um livro. Caso seja necessário continuar gerando novas figuras no mesmo frame ou página como é o caso do script abaixo que gera um arquivo com duas figuras. Caso não seja interessante ter todas as figuras no mesmo arquivo, então é necessário desabilitar o avanço do frame com o recurso especial "gsnFrame". Exemplo de uso: res@gsnFrame = False.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/frame.shtml

Exemplo: Serão geradas duas figuras em páginas diferentes.

```
1   ; Nome do script: cap20_ex03.ncl
2
3   begin
4
5   f = addfile("../../dados/ppt.clima.nc","r")
6   g = addfile("../../dados/olr.clima.nc","r")
7
8   ppt = f->precip
9   olr = short2flt(g->olr)
10
11  printVarSummary(ppt) ; Imprime um resumo sobre a variável ppt.
12  printVarSummary(olr) ; Imprime um resumo sobre a variável olr.
13
14  wks = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex03")
15
16  plot = gsn_csm_y(wks,ppt(:,0,0),False) ; Plot da precipitação.
17
18  plot = gsn_csm_y(wks,olr(:,0,0),False) ; Plot da OLR.
19
20  end
```

O resultado é mostrado abaixo. Lembrando que será gerado um arquivo com duas figuras.

Average Monthly Rate of Precipitation
0
2
4
6
8
10
0
2
4
6
8
10
12

OLR monthly means
200
210
220
230
240
250
260
270
0
2
4
6
8
10
12

Com o uso do recurso gsnFrame (exemplo abaixo), as duas figuras foram colocadas uma ao lado da outra por meio do recurso vpXF (res@vpXF).

```
; Nome do script: cap20_ex04.ncl

begin

f = addfile("../../dados/ppt.clima.nc","r")
g = addfile("../../dados/olr.clima.nc","r")

ppt = f->precip
olr = short2flt(g->olr)

printVarSummary(ppt) ; Imprime um resumo sobre a variável ppt.
printVarSummary(olr) ; Imprime um resumo sobre a variável olr.

wks = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex04")

res           = True
res@gsnFrame  = False ; Não avança o frame.
res@vpWidthF  = 0.4   ; Largura da figura.
res@vpHeightF = 0.4   ; Altura da figura.
res@vpXF      = 0.08  ; Posição x onde será desenhada a primeira
                      ; figura.

plot = gsn_csm_y(wks,ppt(:,0,0),res) ; Plot da precipitação.

res@vpXF      = 0.57  ; Posição x onde será desenhada a segunda
                      ; figura.

plot = gsn_csm_y(wks,olr(:,0,0),res)

frame(wks)

end
```

Ao executar o script, o resultado será:

20.3 Gerando animações

Em alguns casos deseja-se geras animações de algumas variáveis, seja para alguma apresentação ou apenas para fins didáticos. O NCL por meio de outros programas, por exemplo, o imagemagick é capaz de gerar animações. O link abaixo mostra alguns exemplos.

http://www.ncl.ucar.edu/Applications/animate.shtml

Para converter uma animação para um arquivo mpeg, é necessário instalar o programa adequado para isso, por exemplo, o ffmpeg. Basta digitar o comando abaixo no seu terminal para instalar:

sudo apt-get install ffmpeg

Para gerar um vídeo, o procedimento é similar aos arquivos png com a diferença que o arquivo com as animações terá a extensão mpg.

convert -delay 60 tmp*.png video.mpg

Exemplo: Criando animação com dado mensal de precipitação do GPCP. No exemplo abaixo serão criadas 24 figuras no formato png que correspondem aos meses de janeiro de 1980 a dezembro de 1981, e posteriormente, é gerada a animação (anima.gif) com o comando convert.

```
; Nome do script: cap20_ex05.ncl

begin

; jan1979 a out2015.
; [time | 442] x [lat | 72] x [lon | 144]

f = addfile("../../dados/precip.mon.mean.nc","r")

yrStrt = 198001              ; Data inicial => AAAAMM.
yrLast = 198112              ; Data final   => AAAAMM.
TIME   = f->time             ; Importação da variável time do arquivo f.
YYYY   = cd_calendar(TIME,-1); Tempo no formato AAAAMM (opção -1).
ti     = ind(yrStrt.eq.YYYY) ; Seleciona o índice do primeiro tempo.
tf     = ind(yrLast.eq.YYYY) ; Seleciona o índice do último tempo.

ppt    = f->precip ; Importação da variável de interesse do arquivo f.

; Remove os arquivos tmp.*.png e anima.gif.
system("rm -f ../../figuras/cap20/tmp.*.png "+ \ +"
             "../../figuras/cap20/anima.gif")

res                       = True
res@cnFillOn              = True
res@cnFillMode            = "RasterFill"
res@cnLinesOn             = False
res@gsnLeftString         = ""
res@gsnRightString        = ""
res@gsnCenterString       = ""
res@cnLinesOn             = False
res@cnLineLabelsOn        = False
res@cnLevelSelectionMode = "ExplicitLevels"
res@cnLevels              = ispan(2,20,2)
res@cnFillColors          = ispan(2,18,1)
res@mpCenterLonF          = 180.0

do t = ti,tf
   print(" " + YYYY(t))
   wks = gsn_open_wks("png","../../figuras/cap20/tmp."+YYYY(t))
   gsn_define_colormap(wks,"prcp_1")
   res@tiMainString          = "Precipita"+cedil+atilde+"o "+YYYY(t)
   plot = gsn_csm_contour_map_ce(wks,ppt(t,:,:),res)
end do

print("")
print("Criando a animacao")
print("")

; 30 é o tempo entre um frame e outro. Quanto menor o valor, mais rápida
; será a transição entre as figuras. O comando system chama os comandos
; do UNIX para serem executados.
; O convert é um comando do programa ImageMagick. Nesse caso, junta
; todos os ".png" em um único arquivo gif.

system("convert -delay 30 ../../figuras/cap20/*.png "+ \ +"
                        "../../figuras/cap20/anima.gif")
system("rm -f ../../figuras/cap20/tmp.*.png")

end
```

20.4 Adicionando anotações ao plot ou ao frame

No NCL é possível adicionar textos, marcadores, polígonos, linhas, legendas e até outras figuras na figura corrente.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/annotate.shtml

As funções utilizadas são:

- gsn_add_annotation
- gsn_add_polygon
- gsn_add_polyline
- gsn_add_polymarker
- gsn_add_text

As funções para adicionar texto (string ou caracteres) e legendas de forma que é possível anexá-las a outra figura são:

- gsn_create_labelbar
- gsn_create_legend
- gsn_create_text

A função gsn_add_annotation pode ser usada para anexar qualquer objeto como uma anotação a outro objeto (base). Quando você desenhar o objeto base, em seguida, a anotação será adicionada. Se você redimensionar o objeto base, a anotação será redimensionada também.

Há três principais recursos que permitem deteminar onde anexar a anotação:

- amJust
- amOrthogonalPosF
- amParallelPosF

o amJust indica o canto que será usado para posicionar a anotação e baseia-se em outros dois recursos. Há nove possibildiades de posicionamento conforme a Tabela 6:

Tabela 6: Opções para amJust.

"TopLeft"	"CenterLeft"	"BottomLeft"
"TopCenter"	"CenterCenter"	"BottomCenter"
"TopRight"	"CenterRight"	"BottomRight"

A Tabela 7 mostra os valores para utilização dos recursos amParallelPosF e amOrthogonalPosF. Esses valores deslocam na direção x e y a posição do objeto.

Tabela 7: Valores para os recursos amOrthogonalPosF e amParallelPosF.

0.0/ 0.0	Posição no centro do plot
0.5/ 0.5	Posição na parte inferior direita do plot
0.5/-0.5	Posição na parte superior direita do plot
-0.5/-0.5	Posição na parte superior esquerda do plot
-0.5/ 0.5	Posição na parte inferior esquerda do plot

20.5 Anexando um mapa a outro mapa

```
; Nome do script: cap20_ex06.ncl

begin

wks  = gsn_open_wks("pdf", "../../figuras/cap20/cap20_ex06")

; Geração do primeiro mapa (mapa sobre a América do Sul).

res                       = True        ; Habilita os recursos do mapa.
res@gsnMaximize           = True        ; A figura ocupa toda a página.
res@gsnFrame              = False       ; Não avança o frame.
res@gsnDraw               = False       ; Não desenha.
res@mpFillOn              = False       ; Mapa sem preenchimento.
res@mpOutlineBoundarySets = "National"  ; Mostra divisão dos países.
res@mpDataSetName         = "Earth..4"  ; Para a mostrar a divisão
                                        ; dos estados
res@mpDataBaseVersion     = "MediumRes" ; brasileiros, são necessárias
res@mpOutlineSpecifiers   = (/"Brazil:states"/); estas três linhas.
res@mpMinLatF             = -60.               ; Latitude sul.
res@mpMaxLatF             = 20.                ; Latitude norte.
res@mpMinLonF             = -100.              ; Longitude oeste.
res@mpMaxLonF             = -20.               ; Longitude leste.
res@gsnMajorLonSpacing    = 10    ; Espaçamento do eixo x da longitude.
res@gsnMajorLatSpacing    = 10    ; Espaçamento do eixo y da latitude.
res@tmXBMinorOn           = False ; Desabilita os traços secundários dos
res@tmYLMinorOn           = False ; eixos x e y.

base_map = gsn_csm_map_ce(wks,res) ; Cria somente o mapa.

; Geração do segundo mapa (mapa global).

mpres                       = True   ; Habilita os recursos para o mapa.
mpres@gsnFrame              = False  ; Não desenha o mapa.
mpres@gsnDraw               = False  ; Não avança o frame.
mpres@pmTickMarkDisplayMode = "Always" ; Habilita os tickmark
                                       ; (tracinhos) do mapa.
mpres@tmXBLabelFontHeightF  = 0.01   ; Tamanho da fonte do eixo x.
mpres@tmYLLabelFontHeightF  = 0.01   ; Tamanho da fonte do eixo y.
mpres@mpOutlineOn           = True   ; Desenha as linhas no entorno
                                     ; do mapa.
mpres@vpHeightF             = 0.3    ; Torna o segundo mapa
mpres@vpWidthF              = 0.3    ; um pouco menor.

map2 = gsn_csm_map_ce(wks,mpres)
```

```
46  ; Anexa um mapa sobre o outro mapa.
47
48  amres                  = True       ; Habilita os recursos para anotação.
49  amres@amParallelPosF   = 0.105      ; Desloca a figura na direção x.
50  amres@amOrthogonalPosF = 0.44       ; Desloca a figura na direção y.
51  amres@amJust           = "BottomLeft" ; Posicionamento da figura
52                                        ; (parte inferior esquerda).
53
54  ; Anexa o mapa a outro mapa. O map2 é anexado ao base_map.
55  map_anno = gsn_add_annotation(base_map,map2,amres)
56
57  draw(base_map)
58  frame(wks)
59
60  end
```

O resultado será:

20.6 Inserindo uma legenda no gráfico de barras

```
; Nome do script: cap20_ex07.ncl

begin

; GPCP    ACCESS1-0 CCSM4   HADGEM2-CC HADGEM2-ES => primeira linha
; 1804.4  1779.7    1893.3  1739.0     1898.5     => segunda linha

f = asciiread("../../dados/pr.obs.model.txt",(/2,5/),"float")

y = f(1,:)        ; Lê a segunda linha (1) e todos os seus valores (:).
x = (/1,2,3,4,5/) ; Valores para o eixo x.

labels = (/"GPCP","ACCESS1-0","CCSM4","HADGEM2-CC","HADGEM2-ES"/)

nboxes = dimsizes(labels) ; Número de legendas.

cores  = (/"firebrick","red","orange","green","navy"/)

wks = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex07")

res                       = True  ; Habilita personalização do gráfico.       …
res@gsnDraw               = False ; Não gera a figura.
res@gsnFrame              = False ; Não avançao o frame.
res@gsnXYBarChart         = True  ; Habilita gráfico de barras.
res@gsnXYBarChartBarWidth = 0.75  ; Espessura do gráfico de barras.
res@gsnXYBarChartColors   = cores ; Cores das barras.
res@tmXBOn                = False ; Desabilita o tickmarks do eixo x
                                  ; inferior.
res@tmXTOn                = False
res@trYMinF               = 0     ; Mínimo valor do eixo y.
res@trYMaxF               = 2500  ; Máximo valor do eixo y.
res@tmYLMode              = "Manual"
res@tmYLTickStartF        = res@trYMinF ; Para personalizar o eixo y
                                        ; esquerdo (YL)
res@tmYLTickEndF          = res@trYMaxF ; do gráfico de forma manual.
res@tmYLTickSpacingF      = 500         ; Espaçamento do eixo y.
res@tmYLMinorOn           = False
res@trXMinF               = 0           ; Mínimo valor do eixo x.
res@trXMaxF               = 6           ; Máximo valor do eixo x.
res@tiMainString          = "Precipita"+cedil+atilde+ \
                            "o anual na Amaz"+ocirc+"nia"

plot = gsn_csm_xy (wks,x,y,res) ; Geração do gráfico de barras.

; Criação da legenda.

lbres                   = True
lbres@lbAutoManage      = True        ; Necessário para controlar o
                                      ; tamanho das legendas.
lbres@vpWidthF          = 0.2         ; Largura da legenda.
lbres@vpHeightF         = 0.15        ; Altura da legenda.
lbres@lbBoxMajorExtentF = 0.75        ; Espaçamento entre os nomes na
                                      ; legenda.
lbres@lbFillColors      = cores       ; Cores de cada barra da legenda.
lbres@lbMonoFillPattern = True        ; Preechimento das barras da
                                      ; legenda.
lbres@lbLabelFontHeightF = 0.08       ; Tamanho da fonte da legenda.
lbres@lbLabelJust       = "CenterLeft"; Posição da legenda.
```

```
59   lbres@lbPerimOn          = False       ; Não desenha borda em torno
60                                          ; da legenda.
61
62   lbid = gsn_create_labelbar(wks,nboxes,labels,lbres) ; Gera a legenda.
63
64   ; Criação das duas figuras em uma só.
65
66   amres                   = True
67   amres@amJust            = "TopRight" ; Local onde ficará a legenda.
68   amres@amParallelPosF    =  -0.17     ; Posição em x em relação a figura.
69   amres@amOrthogonalPosF  = -0.5       ; Posição em y em relação a figura.
70
71   annoid = gsn_add_annotation(plot,lbid,amres) ; Anexa a legenda (lbid) a
72                                                ; figura de barras (plot).
73   draw(plot)
74   frame(wks)
75
76   end
```

O resultado será:

20.7 Anexando um gráfico de linha ao mapa

```
; Nome do script: cap20_ex08.ncl

begin

f = addfile("../../dados/ppt.clima.nc","r")

ppt = f->precip

printVarSummary(ppt) ; [time | 12] x [lat | 1] x [lon | 1]

wks = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex08")

; Geração do primeiro mapa.
res                         = True
res@gsnFrame                = False      ; Não avança o frame.
res@gsnDraw                 = False      ; Não desenha.
res@mpFillOn                = False      ; Mapa sem preenchimento.
res@mpOutlineBoundarySets   = "National" ; Mostra divisão dos países.
res@mpDataSetName           = "Earth..4" ; Para mostrar a divisão dos
res@mpDataBaseVersion       = "MediumRes"; estados brasileiros são
res@mpOutlineSpecifiers     = (/"Brazil:states"/); necessárias estas linhas.
res@mpMinLatF               = -30.     ; Latitude sul.
res@mpMaxLatF               = 10.      ; Latitude norte.
res@mpMinLonF               = -80.     ; Longitude oeste.
res@mpMaxLonF               = -30.     ; Longitude leste.
res@gsnMajorLonSpacing      = 5        ; Espaçamento do eixo x
                                       ; da longitude.
res@gsnMajorLatSpacing      = 5        ; Espaçamento do eixo y
                                       ; da latitude.
res@tmXBMinorOn             = False    ; Desabilita os traços
                                       ; secundários dos
res@tmYLMinorOn             = False    ; eixos x e y.
res@mpGeophysicalLineColor  = "gray"   ; Cor da linha do contorno do
                                       ; continente.
res@mpNationalLineColor     = "gray"   ; Cor da linha do contorno dos
                                       ; países.
res@mpProvincialLineColor   = "gray"   ; Cor da linha do contorno da
                                       ; divisão dos estados brasileiros.
res@tiMainString            = "Precipita"+cedil+atilde+"o: m"+eacute+ \
                              "dia na "+aacute+"rea"

base_map = gsn_csm_map_ce(wks,res) ; Plot apenas do mapa.

; Geração do gráfico de linha de precipitação.
mpres                   = True
mpres@gsnFrame          = False          ; Não desenha o mapa.
mpres@gsnDraw           = False          ; Não avança o frame.
mpres@vpHeightF         = 0.21           ; Torna o segundo mapa
mpres@vpWidthF          = 0.21           ; um pouco menor.
mpres@xyMarkLineModes   = "MarkLines"    ; Linha com marcadores.
mpres@xyMarkers         =  16            ; Tipo de marcador.
mpres@xyMarkerColor     = "red"          ; Cor do marcador.
mpres@xyLineColors      = "red"          ; Cor da linha.
mpres@xyMarkerSizeF     = 0.01           ; Tamanho do marcador.
mpres@trYMaxF           = 10.0           ; Máximo valor do eixo y.
mpres@trYMinF           = 0.0            ; Mínimo valor do eixo y.
mpres@trXMinF           = 1.0            ; Mínimo valor do eixo x.
mpres@tmXBMode          = "Explicit"     ; Formata o eixo x do meu jeito.
```

```
mpres@tmXBValues        = ispan(1,12,1) ; Valores do eixo x.
mpres@tmXBLabels        = (/"J","F","M","A","M","J","J","A","S","O", \
                            "N","D"/) ; Rótulos do eixo x.
mpres@tiYAxisString       = "Precipita"+cedil+atilde+ \
                            "o (mm dia ~S~-1~N~)"  ; Título.
mpres@tmYLMinorOn         = False      ; Desabilita o minortick
                                       ; secundário da eixo y.
mpres@tmXBLabelFontHeightF = 0.012     ; Tamanho da fonte do eixo x
                                       ; inferior (XB).
mpres@tmYLLabelFontHeightF = 0.012     ; Tamanho da fonte do eixo y
                                       ; esquerdo (YL).
mpres@tiYAxisFontHeightF   = 0.012     ; Tamanho da fonte do título do
                                       ; eixo y.
mpres@tmXBLabelJust       = "CenterCenter" ; Alinhamento dos rótulos
                                           ; do eixo x.

plot = gsn_csm_xy(wks,ispan(1,12,1),ppt(:,0,0),mpres)

; Anexa o gráfico de linha ao mapa.
amres                   = True          ; Habilita os recursos para
                                        ; anotação.
amres@amParallelPosF    = -0.30         ; Desloca a figura na direção x.
amres@amOrthogonalPosF  = 0.25          ; Desloca a figura na direção y.
amres@amJust            = "BottomLeft"  ; Posicionamento da figura
                                        ; (parte inferior esquerda).

; Anexa o gráfico de linha ao mapa. O plot é anexado ao base_map.
map_anno = gsn_add_annotation(base_map,plot,amres)

draw(base_map)
frame(wks)

end
```

O resultado será:

20.8 Alterando o tamanho das fontes do gráfico

No momento de criar os gráficos em algumas situações o tamanho da fonte não é o desejado, mas há alguns recursos específicos para isso.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/font.shtml

```
; Nome do script: cap20_ex09.ncl

begin

f = addfile("../../dados/prec.2014.nc","r")

ppt  = f->precip

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex09")

plot = new(2,graphic) ; Variável necessária para criar o painel com
                      ; duas figuras.

res                = True           ; Habilita personalização do gráfico.
res@cnFillOn       = True           ; Gráfico preenchido.
res@cnFillPalette = "precip2_17lev" ; Define o mapa de cores.
res@lbOrientation  = "Vertical"     ; Orientação da legenda da figura.
res@gsnDraw        = False          ; Não desenha.
res@gsnFrame       = False          ; Não avança o frame.

plot(0) = gsn_csm_contour_map_ce(wks,ppt(0,:,:),res)

; Altera o tamanho da fonte de todos os títulos da segunda figura. Caso
; seja necessário alterar individualmente o tamanho da fonte dos títulos,
; basta usar os recursos: gsnLeftStringFontHeightF,
; gsnRightStringFontHeightF e gsnCenterStringFontHeightF.

res@gsnStringFontHeightF = 0.025

; Plot da segunda figura com o tamanho dos títulos alterado para 0.025.

plot(1) = gsn_csm_contour_map(wks,ppt(0,:,:),res)

; Criação do painel.
resP                            = True
resP@gsnPanelYWhiteSpacePercent = 10   ; Espaçamento vertical entre
                                       ; as figuras.

gsn_panel(wks,plot,(/2,1/),resP)

end
```

O resultado será:

Alternado o tamanho das fontes dos eixos x e y.

```
1   ; Nome do script: cap20_ex10.ncl
2
3   begin
4
5   f = addfile("../../dados/prec.2014.nc","r")
6
7   ppt  = f->precip
8
9   wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex10")
10
11  plot = new(2,graphic) ; Variável necessária para criar o painel
12                        ; com duas figuras.
13
14  res                 = True            ; Habilita personalização do gráfico.
15  res@cnFillOn        = True            ; Gráfico preenchido.
16  res@cnFillPalette   = "precip2_17lev" ; Define o mapa de cores.
17  res@lbOrientation   = "Vertical"      ; Orientação da legenda da figura.
18  res@gsnDraw         = False           ; Não desenha.
19  res@gsnFrame        = False           ; Não avança o frame.
20
21  ; Plot da primeira figura com o tamanho original da fonte.
22
23  plot(0) = gsn_csm_contour_map_ce(wks,ppt(0,:,:),res)
```

```
24
25  res@tmXBLabelFontHeightF = 0.02  ; Tamanho da fonte do eixo y inferior.
26  res@tmYLLabelFontHeightF = 0.02  ; Tamanho da fonte do eixo y esquerdo.
27  res@lbLabelStride        = 2     ; Intervalo dos rótulos da legenda da
28                                   ; figura.
29  res@tmXBTickSpacingF     = 60.   ; Espaçamento dos rótulos do eixo x
30                                   ; inferior (XB).
31  res@gsnStringFontHeightF = 0.025 ; Altera o tamanho da fonte de todos
32                                   ; os títulos da figura simultaneamente
33                                   ; da segunda figura. Caso seja
34                                   ; necessário alterar individualmente o
35                                   ; tamanho da fonte dos títulos, basta
36                                   ; fazer uso dos recursos:
37                                   ; gsnLeftStringFontHeightF,
38                                   ; gsnRightStringFontHeightF e
39                                   ; gsnCenterStringFontHeightF.
40
41  ; Plot da segunda figura com o tamanho dos títulos alterado para 0.025.
42
43  plot(1) = gsn_csm_contour_map(wks,ppt(0,:,:),res)
44
45  ; Criação do painel.
46  resP                            = True
47  resP@gsnPanelYWhiteSpacePercent = 10   ; Espaçamento vertical entre
48                                         ; as figuras.
49
50  gsn_panel(wks,plot,(/2,1/),resP)
51
52  end
```

O resultado será:

Alterando o tamanho da fonte da legenda.

```
; Nome do script: cap20_ex11.ncl

begin

f = addfile("../../dados/prec.2014.nc","r")

ppt  = f->precip

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex11")

plot = new(2,graphic) ; Variável necessária para criar o
                      ; painel com duas figuras.

res                = True            ; Habilita personalização do gráfico.
res@cnFillOn       = True            ; Gráfico preenchido.
res@cnFillPalette  = "precip2_17lev" ; Define o mapa de cores.
res@lbOrientation  = "Vertical"      ; Orientação da legenda da figura.
res@gsnDraw        = False           ; Não desenha.
res@gsnFrame       = False           ; Não avança o frame.

plot(0) = gsn_csm_contour_map_ce(wks,ppt(0,:,:),res) ; Plot da primeira
                                                     ; figura.

res@lbLabelFontHeightF = 0.02 ; Altera o tamanho da fonte da legenda
                              ; de cores.

plot(1) = gsn_csm_contour_map(wks,ppt(0,:,:),res)  ; Plot da segunda
                                                   ; figura.

; Criação do painel.
resP                            = True
resP@gsnPanelYWhiteSpacePercent = 10   ; Espaçamento vertical entre
                                       ; as figuras.

gsn_panel(wks,plot,(/2,1/),resP)

end
```

O resultado será:

Alterando o tamanho da fonte do título da figura.

```
; Nome do script: cap20_ex12.ncl

begin

f = addfile("../../dados/prec.2014.nc","r")

ppt  = f->precip

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex12")

plot = new(2,graphic) ; Variável necessária para criar o painel com
            ; duas figuras.

res                 = True            ; Habilita personalização do gráfico.
res@cnFillOn        = True            ; Gráfico preenchido.
res@cnFillPalette   = "precip2_17lev" ; Define o mapa de cores.
res@lbOrientation   = "Vertical"      ; Orientação da legenda da figura.
res@gsnDraw         = False           ; Não desenha.
res@gsnFrame        = False           ; Não avança o frame.

plot(0) = gsn_csm_contour_map_ce(wks,ppt(0,:,:),res) ; Plot da primeira
                                                     ; figura.

res@tiMainString      = "Precipita"+cedil+atilde+"o" ; Título da figura.
res@tiMainFontHeightF = 0.03 ; Tamanho da fonte do título principal
                             ; da figura.
```

```
27
28  plot(1) = gsn_csm_contour_map(wks,ppt(0,:,:),res) ; Plot da segunda
29                                                    ; figura.
30
31  ; Criação do painel.
32  resP                              = True
33  resP@gsnPanelYWhiteSpacePercent = 10    ; Espaçamento vertical entre
34                                          ; as figuras.
35
36  gsn_panel(wks,plot,(/2,1/),resP)
37
38  end
```

O resultado será:

Alternado o tamanho dos rótulos dos contornos:

```
; Nome do script: cap20_ex13.ncl

begin

f = addfile("../../dados/prec.2014.nc","r")

ppt  = f->precip

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex13")

plot = new(2,graphic) ; Variável necessária para criar o painel com
            ; duas figuras.

res                 = True       ; Habilita personalização do gráfico.
res@cnFillPalette   = "wgne15"   ; Define o mapa de cores.
res@lbOrientation   = "Vertical" ; Orientação da legenda da figura.
res@gsnDraw         = False      ; Não desenha.
res@gsnFrame        = False      ; Não avança o frame.
res@mpMinLonF       = -90.0      ; Aplica
res@mpMaxLonF       = -20.0      ; um zoom
res@mpMinLatF       = -30.0      ; na áres de
res@mpMaxLatF       =  20.0      ; interesse.

plot(0) = gsn_csm_contour_map_ce(wks,ppt(0,:,:),res) ; Plot da primeira
                                                     ; figura.

res@cnLineLabelFontHeightF   = 0.03  ; Tamanho da fonte dos rótulos
                                     ; dos contornos.
res@cnInfoLabelFontHeightF   = 0.03  ; Tamanho da fonte das informações
                                     ; dos rótulos dos contornos.

plot(1) = gsn_csm_contour_map(wks,ppt(0,:,:),res) ; Plot da segunda
                                                  ; figura.

; Criação do painel.
resP                            = True
resP@gsnPanelYWhiteSpacePercent = 10   ; Espaçamento vertical entre
                                       ; as figuras.

gsn_panel(wks,plot,(/2,1/),resP)

end
```

O resultado será:

20.9 Personalização dos rótulos da legenda

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/labelbar.shtml

http://www.ncl.ucar.edu/Document/Graphics/Resources/lb.shtml

Exemplo:

```
; Nome do script: cap20_ex14.ncl

begin

f = addfile("../../dados/prec.2014.nc","r")

ppt  = f->precip

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex14")

res                 = True            ; Habilita personalização do gráfico.
res@cnFillOn        = True            ; Gráfico preenchido.
res@cnFillPalette   = "precip2_17lev" ; Define o mapa de cores.

; Para desabilitar a legenda de cores, basta definir o seguinte recurso:
; res@lbLabelBarOn = False

plot = gsn_csm_contour_map_ce(wks,ppt(0,:,:),res) ; Plot da figura.

end
```

O resultado será:

Legenda de cores na vertical:

```
; Nome do script: cap20_ex15.ncl

begin

f = addfile("../../dados/prec.2014.nc","r")

ppt  = f->precip

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex15")

res                 = True            ; Habilita personalização do gráfico.
res@cnFillOn        = True            ; Gráfico preenchido.
res@lbOrientation   = "vertical"      ; Legenda de cores na vertical.
                                      ; O padrão é horizontal.
res@cnFillPalette   = "precip2_17lev" ; Define o mapa de cores.

plot = gsn_csm_contour_map_ce(wks,ppt(0,:,:),res)

end
```

O resultado será:

Adicionando título na legenda de cores:

```
; Nome do script: cap20_ex16.ncl

begin

f = addfile("../../dados/prec.2014.nc","r")

ppt  = f->precip

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex16")

res                          = True
res@cnFillOn                 = True        ; Gráfico preenchido.
res@lbLabelAngleF            = 90          ; Inclinação em graus dos
                                           ; nomes na legenda.
res@lbLabelFontHeightF       = 0.025       ; Tamanho dos valores da
                                           ; legenda.
res@lbTitleOn                = True        ; Habilita um título para a
                                           ; legenda.
res@lbTitleString            = "(mm/dia)"  ; Título da legenda.
res@lbTitleFontHeightF       = 0.025       ; Tamanho da fonte do título
                                           ; da legenda.
res@pmLabelBarOrthogonalPosF = 0.15        ; Deslocamento na direção y
                                           ; da barra de legenda.
res@cnFillPalette  = "precip2_17lev"       ; Define o mapa de cores.

plot = gsn_csm_contour_map_ce(wks,ppt(0,:,:),res) ; Plot da figura.

end
```

O resultado será:

Formatando a legenda de cores:

```
; Nome do script: cap20_ex17.ncl

begin

f = addfile("../../dados/prec.2014.nc","r")

ppt  = f->precip

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex17")

res                          = True
res@cnFillOn                 = True       ; Gráfico preenchido.
res@cnFillPalette            = "precip2_17lev" ; Define o mapa de cores.
res@lbLabelFontHeightF       = 0.015      ; Tamanho dos valores da
                                          ; legenda.
res@lbTitleOn                = True       ; Habilita um título para
                                          ; a legenda.
res@lbTitleString            = "(mm/dia)" ; Título da legenda.
res@lbTitleFontHeightF       = 0.025      ; Tamanho da fonte do título
                                          ; da legenda.
res@pmLabelBarOrthogonalPosF = 0.15       ; Deslocamento na direção y
                                          ; da barra de legenda.
res@lbTitlePosition          = "Right"    ; Posição do título da legenda.
res@lbTitleDirection         = "Across"   ; Direção do título da legenda.
res@lbLabelFont              = "Helvetica-Bold" ; Tipo de fonte a ser
                                                ; utilizado.
res@lbLabelPosition          = "Center"         ; Posição dos rótulo da
                                                ; legenda.
res@lbLabelAlignment         = "BoxCenters"     ; Orientação dos rótulos.
                               ; Os valores abaixo correspondem aos
                               ; rótulos que vão aparecer na legenda,
                               ; ou seja, os próprios valores da variável.
res@lbLabelStrings           =  (/"1","2","3","4","5","6","7","8", \
                                  "9","10","11","12"/)

plot = gsn_csm_contour_map_ce(wks,ppt(0,:,:),res)

end
```

O resultado será:

20.10 Polígonos

Esté tópico mostrará como adicionar marcadores, linhas, áreas prenchidas e texto em uma figura.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/polyg.shtml

Para desenhar primitivas em um mapa deve-se usar o espaço ocupado pela figura. Em um mapa, por exemplo, utilizam-se os valores de latitutde e de longitude para posicionar o que se deseja inserir. As rotinas são mostradas na Tabela 8:

Tabela 8: Rotinas para inserir polígonos.

gsn_add_polyline	gsn_polyline
gsn_add_polymarker	gsn_polymarker
gsn_add_polygon	gsn_polygon
gsn_add_text	gsn_text

A diferença entre as versões "add" dessas rotinas (que são funções) e as outras rotinas (que são procedimentos) é que as funções "add" anexam os marcadores, polígonos e textos ao mapa existente e o usuário não verá o mapa até realmente desenhar o gráfico. Se o mapa é redimensionado, a primitiva também será redimensionada.

Para desenhar primitivas em unidades NDC deve-se utilizar valores entre 0.0 e 1.0 para definir a localização do que se pretende inserir. A localização (0.0,0.0) representa o canto inferior esquerdo e (1.0,1.0) representa o canto superior direito. Os procedimentos para desenhar as primitvas são:

- gsn_polyline_ndc
- gsn_polymarker_ndc

- gsn_polygon_ndc
- gsn_text_ndc

20.10.1 Desenhando uma caixa sobre o mapa

```
; Nome do script: cap20_ex18.ncl

begin

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex18")

res                        = True
res@gsnFrame               = False                ; Não avança o frame.
res@mpMinLatF              = -60.                 ; Latitude sul.
res@mpMaxLatF              = 20.                  ; Latitude norte.
res@mpMinLonF              = -90.                 ; Longitude leste.
res@mpMaxLonF              = -30.                 ; Longitude oeste.
res@mpOutlineBoundarySets  = "National"           ; Divisão dos países.
res@mpDataSetName          = "Earth..4"           ;
res@mpDataBaseVersion      = "MediumRes"          ;
res@mpOutlineSpecifiers    = (/"Brazil:states"/)  ; Divisão do estados
                                                  ; brasileiros.
res@mpFillOn               = False                ; Mapa com fundo transparente.

plot = gsn_csm_map_ce(wks,res) ; Gera somente o mapa.

; Caixa sobre a Amazônia.

; Valores de latitude para desenhar a caixa.
boxlat = (/-13.7,-13.7,  2.0,  2.0,-13.7 /)

; Valores de longitude para desenhar a caixa.
boxlon = (/290.7,312.0,312.0,290.7,290.7 /)

; Desenha a caixa.
lnres                   = True  ; Habilita a personalização.
lnres@gsLineColor       = "red" ; Cor da linha da caixa.
lnres@gsLineThicknessF  = 5.0   ; Espessura da linha da caixa.

caixa = gsn_add_polyline(wks,plot,boxlon,boxlat,lnres) ; Desenha a caixa.

; Adiciona um texto dentro da caixa.
txres               = True  ; Habilita a personalização.
txres@txFontHeightF = 0.025 ; Tamanho do texto que está dentro da caixa.
txres@txFontColor   = "red" ; Cor do texto que está dentro da caixa.

midlon = 0.5 * (max(boxlon)+min(boxlon)) ; Define o centro da caixa
midlat = 0.5 * (max(boxlat)+min(boxlat)) ; para inserir o texto AMZ.

; Adiciona o texto AMZ a caixa.
texto = gsn_add_text(wks,plot,"AMZ",midlon,midlat,txres)

draw(plot)
frame(wks)

end
```

O resultado será:

20.10.2 Desenhando marcadores sobre um mapa

```
1   ; Nome do script: cap20_ex19.ncl
2
3   begin
4
5   nlinhas  = 336 ; Número de linhas do arquivo.
6   ncolunas = 2   ; Número de colunas do arquivo.
7
8   ; Abertura do arquivo lat_lon.txt.
9   data = asciiread("../../dados/lat_lon.txt",(/nlinhas,ncolunas/),"float")
10  lat  = data(:,0) ; Primeira coluna do arquivo => valores de latitude.
11  lon  = data(:,1) ; Segunda coluna do arquivo  => valores de longitude.
12
13  wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex19")
14
15  res                        = True
16  res@gsnFrame               = False              ; Não avança o frame.
17  res@mpMinLatF              = -60.               ; Latitude sul.
18  res@mpMaxLatF              = 20.                ; Latitude norte.
19  res@mpMinLonF              = -90.               ; Longitude leste.
20  res@mpMaxLonF              = -30.               ; Longitude oeste.
21  res@mpOutlineBoundarySets  = "National"         ; Divisao dos países.
22  res@mpDataSetName          = "Earth..4"         ;
23  res@mpDataBaseVersion      = "MediumRes"        ;
24  res@mpOutlineSpecifiers    = (/"Brazil:states"/) ; Divisão do estados
25                                                  ; brasileiros.
```

```
26  res@mpFillOn              = False       ; Mapa com fundo transparente.
27  res@gsnMajorLonSpacing    = 10          ; Espaçamento da longitude.
28  res@gsnMajorLatSpacing    = 10          ; Espaçamento da latitude.
29
30  map = gsn_csm_map(wks,res)              ; Gera o mapa.
31
32  ; Gera os  marcadores.
33  mkres                = True  ; Habilita a personalização dos marcadores.
34  mkres@gsMarkerIndex = 2     ; Tipo de marcador. 2 = Cruz.
35  mkres@gsMarkerColor = "Blue"; Cor do marcador.
36
37  do k = 0,nlinhas-1
38     gsn_polymarker(wks,map,lon(k),lat(k),mkres)
39  end do
40
41  ; Desenha no mapa no canto superior direito o marcador (+). Como se
42  ; fosse uma legenda.
43  gsres                     = True   ; Habilita personalização.
44  gsres@gsMarkerIndex       = 2      ; Tipo de marcador.
45  gsres@gsMarkerColor       = "Blue" ; Cor do marcador.
46  gsres@gsMarkerThicknessF = 3.0    ; Espessura do marcador.
47  gsres@gsMarkerSizeF       = 0.015  ; Tamanho do marcador.
48
49  ; Adiciona o marcador na posição x=0.63 e y=0.82.
50  gsn_polymarker_ndc(wks,0.63,0.82,gsres)
51
52  ; Adiciona o nome "AMZ LEGAL" no canto superior direito.
53  txres                = True       ; Habilita personalização.
54  txres@txFontHeightF = 0.015      ; Tamanho da fonte.
55  txres@txFontColor   = "Blue"     ; Cor da fonte.
56
57  ; Desenha o nome "AMZ LEGAL" no canto superior direito  na posição
58  ; x=0.70 e y=0.82.
59  gsn_text_ndc(wks,"AMZ LEGAL",0.70,0.82,txres)
60
61  frame(wks)
62
63  end
```

O resultado será:

20.11 Adicionando e formatando texto

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/text.shtml

20.11.1 Utilizando as coordenadas da figura

```
1   ; Nome do script: cap20_ex20.ncl
2
3   begin
4
5   f = addfile("../../dados/prec.2005.nc","r") ; Abertura do arquivo.
6
7   ppt = f->pre ; Importação da variável.
8
9   wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex20")
10
11  res                 = True   ; Habilita personalização.
12  res@gsnFrame        = False  ; Não avança o frame.
13  res@gsnDraw         = False  ; Não desenha a figura.
14
15  plot=gsn_csm_y(wks,ppt(:,0,0),res) ; Gera o gráfico de linha.
16
17  txres               = True   ; Habilita a personalização do texto.
18  txres@txFontHeightF = 0.03   ; Tamanho da fonte do texto a
19                               ; ser inserido no gráfico.
20
21  ; O texto será inserido na posição x=4.0 e y=8.0. Ou seja, no
22  ; valor do eixo x=4 e eixo y=8.
23  ; O texto ficará centralizado na posição x=4.0. É possível
24  ; alinhar o texto para obter o que se deseja.
25
```

```
; Adicionando o texto.
dum = gsn_add_text(wks,plot,"Adicionando texto",4.0,8.0,txres)

draw(plot)
frame(wks) ; Avança o frame.

end
```

O resultado será:

20.11.2 Formatando a posição do texto da figura

```
; Nome do script: cap20_ex21.ncl

begin

f = addfile("../../dados/prec.2005.nc","r") ; Abertura do arquivo.

ppt = f->pre ; Importação da variável.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex21")

res                 = True    ; Habilita personalização.
res@gsnFrame        = False   ; Não avança o frame.
res@gsnDraw         = False   ; Não desenha a figura.

plot=gsn_csm_y(wks,ppt(:,0,0),res) ; Gera o gráfico de linha.

txres               = True          : Habilita a personalizacão do texto.
```

```
18   txres@txFontHeightF = 0.03          ; Tamanho da fonte do texto a ser
19                                       ; inserido no gráfico.
20   txres@txJust        = "CenterLeft" ; Alinhamento do texto no
21                                       ; centro/esquerda.
22                                       ; O texto será escrito após o ponto
23                                       ; (x,y) definido pelo usuário por
24                                       ; meio da função gsn_add_text.
25
26   ; O texto será inserido na posição x=4.0 e y=8.0. Ou seja, no valor do
27   ; eixo x=4 e eixo y=8.
28   ; O texto ficará centralizado na posição x=4.0. É possível alinhar o
29   ; texto para obter o que se deseja.
30
31   dum = gsn_add_text(wks,plot,"Adicionando texto",4.0,8.0,txres)
32
33   draw(plot)
34   frame(wks)
35
36   end
```

O resultado será:

20.11.3 Utilizando as coordenadas NDC

```
; Nome do script: cap20_ex22.ncl

begin

f = addfile("../../dados/prec.2005.nc","r") ; Abertura do arquivo.

ppt = f->pre ; Importação da variável.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex22")

drawNDCGrid(wks) ; Desenha as linhas de grade para adicionar o
                 ; texto na posição correta.

res                 = True   ; Habilita personalização.
res@gsnFrame        = False  ; Não avança o frame.

plot=gsn_csm_y(wks,ppt(:,0,0),res) ; Gera o gráfico de linha.

txres               = True   ; Habilita a personalização do texto.
txres@txFontHeightF = 0.03   ; Tamanho da fonte.

; O texto será inserido na posição x=0.2 e y=0.9 que representa
; as coordenadas NDC. O texto ficará centralizado na posição x=0.2.
; É possível alinhar o texto para obter o que se deseja.

gsn_text_ndc(wks,"Adicionando texto",0.2,0.9,txres)

frame(wks)

end
```

O resultado será:

20.11.4 Adicionando quebra de linha no texto do gráfico

```
; Nome do script: cap20_ex23.ncl

begin

f = addfile("../../dados/prec.2005.nc","r")

ppt = f->pre ; Importação da variável.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex23")

res                  = True   ; Habilita personalização.

; Basta usar ~C~ para adicionar quebra de linha. Para alinhar
; o texto deve-se inserir espaços até conseguir o efeito que se deseja.

res@tiMainString     = "Adicionando quebra de ~C~         linha no texto"

plot=gsn_csm_y(wks,ppt(:,0,0),res) ; Gera o gráfico de linha.

end
```

O resultado será:

20.12 Personalizando os marcadores de escala (tickmarks)

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/tickmarks.shtml

```
; Nome do script: cap20_ex24.ncl

begin

f = addfile("../../dados/SA.CRU.prec.anual.1979.2013.nc","r")

ppt = f->pre ; Importação da variável.

years  = ispan(1979,2013,1) ; Cria os valores do eixo x.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex24")

plot = new(2,graphic) ; Arranjo gráfico para a criação
                      ; do painel com as duas figuras.

res            = True         ; Habilita personalização.
res@gsnDraw    = False        ; Não gera a figura.
res@gsnFrame   = False        ; Não avança o frame.
res@vpWidthF   = 0.8          ; Largura da figura.
res@vpHeightF  = 0.3          ; Altura da figura.
res@trYMinF    = 4.0          ; Mínimo valor do eixo y.
res@trYMaxF    = 5.0          ; Máximo valor do eixo y.
res@trXMinF    = min(years)   ; Mínimo valor do eixo x.
res@trXMaxF    = max(years)   ; Máximo valor do eixo x.
res@tiMainString = "Define o minimo e maximo valor dos eixos x e y"

plot(0)=gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico de linha.

res@tiMainString    = "Define o eixo x inferior de forma manual"
res@tmXBMode        = "Manual"    ; Define o eixo x inferior (XB)
                                  ; de forma manual.
res@tmXBTickStartF  = res@trXMinF ; O valor que começará o tickmark
                                  ; do eixo x inferior.

plot(1)=gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico de linha.

; Criação dos paineis.
resP                            = True
resP@gsnPanelYWhiteSpacePercent = 10    ; Espaçamento vertical entre
                                        ; as figuras.

gsn_panel(wks,plot,(/2,1/),resP)        ; Gera o painel.

end
```

O resultado será:

Definindo o eixo x de forma manual e explícita:

```
; Nome do script: cap20_ex25.ncl

begin

f = addfile("../../dados/SA.CRU.prec.anual.1979.2013.nc","r")

ppt = f->pre ; Importação da variável.

years  = ispan(1979,2013,1) ; Cria os valores do eixo x.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex25")

plot = new(2,graphic) ; Arranjo gráfico para a criação do painel com as
                      ; duas figuras.

res                 = True                ; Habilita personalização.
res@gsnDraw         = False               ; Não gera a figura.
res@gsnFrame        = False               ; Não avança o frame.
res@vpWidthF        = 0.8                 ; Largura da figura.
res@vpHeightF       = 0.3                 ; Altura da figura.
res@trYMinF         = 4.0                 ; Mínimo valor do eixo y.
res@trYMaxF         = 5.0                 ; Máximo valor do eixo y.
res@trXMinF         = min(years)          ; Mínimo valor do eixo x.
res@trXMaxF         = max(years)          ; Máximo valor do eixo x.
res@tmXBMode        = "Manual"            ; Eixo x inferior (XB) de
                                          ; forma manual.
res@tmXBTickStartF  = res@trXMinF         ; Onde começam os valores do
```

```
                                                ; eixo x.
res@tmXBTickEndF     = res@trXMaxF              ; Onde terminam os valores
                                                ; do eixo x.
res@tmXBTickSpacingF= 5                         ; Espaçamento entre os
                                                ; rótulos.
res@gsnLeftString    = "tmXBMode = Manual"      ; Título da figura no lado
                                                ; esquerdo.

plot(0)=gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico de linha.

res@tmXBMode         = "Explicit"                ; Eixo x inferior (XB)
                                                 ; de forma explícita.
res@tmXBValues       = (/1980,1990,2000,2010/)   ; Valores que vão
                                                 ; aparecer no eixo x.
res@tmXBLabels       = res@tmXBValues            ; Rótulos do eixo x
                                                 ; inferior.
res@tmXBMinorValues  = ispan(1978,2014,1)        ; Intervalo do minortick
                                                 ; secundário.
res@gsnLeftString    = "tmXBMode = Explicit"     ; Título da figura no
                                                 ; lado esquerdo.

plot(1)=gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico de linha.

; Criação dos paineis.
resP                              = True
resP@gsnPanelYWhiteSpacePercent = 10     ; Espaçamento vertical entre
                                         ; as figuras.

gsn_panel(wks,plot,(/2,1/),resP)         ; Gera o painel.

end
```

O resultado será:

Personalizando o eixo x e as barras do gráfico:

```
; Nome do script: cap20_ex26.ncl

begin

; Abertura do arquivo de anomalia de precipitação para o ano de
; 2013 (jan a dez).

f = addfile("../../dados/SA.CRU.anom.prec.2013.nc","r")

ppt = f->pre ; Importação da variável.

years  = ispan(1,12,1) ; Cria os valores do eixo x.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex26")

res                        = True          ; Habilita personalização.
res@vpWidthF               = 0.8           ; Largura da figura.
res@vpHeightF              = 0.3           ; Altura da figura.
res@trYMinF                = -1.0          ; Mínimo valor do eixo y.
res@trYMaxF                = 1.0           ; Máximo valor do eixo y.
res@trXMinF                = min(years)-1  ; Mínimo valor do eixo x.
res@trXMaxF                = max(years)+1  ; Máximo valor do eixo x.
res@gsnXYBarChart          = True          ; Habilita gráfico de barras.
res@gsnYRefLine            = 0.            ; Linha de referência no
                                           ; valor zero.
res@gsnAboveYRefLineColor  = "Blue"        ; Cor azul para valores negativos.
res@gsnBelowYRefLineColor  = "Red"         ; Cor vermelho para valores positivos.
res@trYReverse             = True          ; Inverte o eixo y.
res@tmXBLabelAngleF        = 45            ; Ângulo dos rótulos do eixo x
                                           ; inferior (XB).
res@tmXBLabelDeltaF        = 1.5           ; Distância dos rótulos do
                                           ; eixo x do seu eixo.
res@tmXBLabelFontColor     = "orange"      ; Cor dos rótulos do eixo x.

plot=gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico de linha.

end
```

O resultado será:

20.13 Personalizando os títulos da figura

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/title.shtml

```
; Nome do script: cap20_ex27.ncl

begin

f = addfile("../../dados/SA.CRU.anom.prec.2013.nc","r")

ppt = f->pre ; Importação da variável.

years  = ispan(1,12,1) ; Cria os valores do eixo x.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex27")

plot = new(2,graphic) ; Arranjo gráfico para a criação do painel
                      ; com as duas figuras.

res                       = True       ; Habilita personalização.
res@gsnDraw               = False      ; Não gera a figura.
res@gsnFrame              = False      ; Não avança o frame.
res@vpWidthF              = 0.8        ; Largura da figura.
res@vpHeightF             = 0.3        ; Altura da figura.
res@trYMinF               = -1.0       ; Mínimo valor do eixo y.
res@trYMaxF               = 1.0        ; Máximo valor do eixo y.
res@trXMinF               = min(years) ; Mínimo valor do eixo x.
res@trXMaxF               = max(years) ; Máximo valor do eixo x.
res@gsnXYBarChart         = True       ; Habilita gráfico de barras.
res@gsnYRefLine           = 0.         ; Linha de referência no
                                       ; valor zero.
res@gsnAboveYRefLineColor = "Red"      ; Cor vermelho para valores positivos.
res@gsnBelowYRefLineColor = "Blue"     ; Cor azul para valores negativos.
res@tiMainString          = "Titulo principal1"
res@tiXAxisString         = "Titulo do eixo x1" ; Título do eixo x.
res@tiYAxisString         = "Titulo do eixo y1" ; Título do eixo y.

plot(0) = gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico.

res@tiMainString         = "Titulo principal2" ; Título principal.
res@tiMainJust           = "CenterLeft"        ; Alinhamento do texto.
res@tiMainPosition       = "Left"              ; Posição do texto.

res@tiXAxisString        = "Titulo do eixo x2" ; Título do eixo x.
res@tiXAxisJust          = "CenterRight"       ; Alinhamento do texto.
res@tiXAxisPosition      = "Right"             ; Posição do texto.

res@tiYAxisString        = "Titulo do eixo y2" ; Título do eixo y.
res@tiYAxisJust          = "CenterLeft"        ; Alinhamento do texto.
res@tiYAxisSide          = "Right"             ; Posição do texto.
res@tiYAxisPosition      = "Bottom"          ; Título na parte inferior
                                             ; do eixo direito.

plot(1) = gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico.

; Criação dos paineis.
resP                           = True
```

```
resP@gsnPanelYWhiteSpacePercent = 10     ; Espaçamento vertical entre
                                         ; as figuras.

gsn_panel(wks,plot,(/2,1/),resP)         ; Gera o painel.

end
```

O resultado será:

Adicionando títulos:

```
; Nome do script: cap20_ex28.ncl

begin

f = addfile("../../dados/SA.CRU.anom.prec.2013.nc","r")

ppt = f->pre ; Importação da variável.

years  = ispan(1,12,1) ; Cria os valores do eixo x.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex28")

res                        = True       ; Habilita personalização.
res@vpWidthF               = 0.8        ; Largura da figura.
res@vpHeightF              = 0.3        ; Altura da figura.
res@trYMinF                = -1.0       ; Mínimo valor do eixo y.
res@trYMaxF                = 1.0        ; Máximo valor do eixo y.
```

```
18  res@trXMinF                = min(years) ; Mínimo valor do eixo x.
19  res@trXMaxF                = max(years) ; Máximo valor do eixo x.
20  res@gsnXYBarChart          = True       ; Habilita gráfico de barras.
21  res@gsnYRefLine            = 0.         ; Linha de referência no
22                                          ; valor zero.
23  res@gsnAboveYRefLineColor = "Red"       ; Cor vermelho para valores positivos.
24  res@gsnBelowYRefLineColor = "Blue"      ; Cor azul para valores negativos.
25  res@gsnLeftString          = "Esquerdo" ; Adiciona texto do lado esquerdo.
26  res@gsnCenterString        = "Centro"   ; Adiciona texto no centro.
27  res@gsnRightString         = "Direito"  ; Adiciona texto do lado direito.
28
29  plot = gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico.
30
31  end
```

O resultado será:

Adicionando o título principal e nos eixos x e y:

```
1   ; Nome do script: cap20_ex29.ncl
2
3   begin
4
5   f = addfile("../../dados/SA.CRU.anom.prec.2013.nc","r")
6
7   ppt = f->pre ; Importação da variável.
8
9   years  = ispan(1,12,1) ; Cria os valores do eixo x.
10
11  wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex29")
12
13  res                        = True       ; Habilita personalização.
14  res@vpWidthF               = 0.8        ; Largura da figura.
15  res@vpHeightF              = 0.3        ; Altura da figura.
16  res@trYMinF                = -1.0       ; Mínimo valor do eixo y.
17  res@trYMaxF                = 1.0        ; Máximo valor do eixo y.
18  res@trXMinF                = min(years) ; Mínimo valor do eixo x.
19  res@trXMaxF                = max(years) ; Máximo valor do eixo x.
20  res@gsnXYBarChart          = True       ; Habilita gráfico de barras.
21  res@gsnYRefLine            = 0.         ; Linha de referência no valor
22                                          ; zero.
23  res@gsnAboveYRefLineColor = "Red"       ; Cor vermelho para valores positivos.
24  res@gsnBelowYRefLineColor = "Blue"      ; Cor azul para valores negativos.
25  res@tiMainString           = "Titulo principal"; Título principal.
```

```
res@tiXAxisString          = "Eixo x"          ; Título do eixo x.
res@tiYAxisString          = "Eixo y"          ; Título do eixo y.
res@tiMainFontHeightF      = 0.05              ; Tamanho da fonte do
                                               ; título principal.
res@tiXAxisFontHeightF     = 0.03 ; Tamanho da fonte do título do eixo x.
res@tiYAxisFontHeightF     = 0.03 ; Tamanho da fonte do título do eixo y.

plot = gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico.

end
```

O resultado será:

Aumentando a fonte dos título esquerdo, central e direito:

```
; Nome do script: cap20_ex30.ncl

begin

f = addfile("../../dados/SA.CRU.anom.prec.2013.nc","r")

ppt = f->pre ; Importação da variável.

years  = ispan(1,12,1) ; Cria os valores do eixo x.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex30")

res                        = True        ; Habilita personalização.
res@vpWidthF               = 0.8         ; Largura da figura.
res@vpHeightF              = 0.3         ; Altura da figura.
res@trYMinF                = -1.0        ; Mínimo valor do eixo y.
res@trYMaxF                = 1.0         ; Máximo valor do eixo y.
res@trXMinF                = min(years)  ; Mínimo valor do eixo x.
res@trXMaxF                = max(years)  ; Máximo valor do eixo x.
res@gsnXYBarChart          = True        ; Habilita gráfico de barras.
res@gsnYRefLine            = 0.          ; Linha de referência no valor
                                         ; zero.
res@gsnAboveYRefLineColor = "Red"        ; Cor vermelho para valores positivos.
res@gsnBelowYRefLineColor = "Blue"       ; Cor azul para valores negativos.
```

```
res@gsnLeftString          = "Esquerdo" ; Adiciona texto do lado esquerdo
                                        ; da figura.
res@gsnCenterString        = "Centro"   ; Adiciona texto no centro da
                                        ; figura.
res@gsnRightString         = "Direito"  ; Adiciona texto do lado direito
                                        ; da figura.
res@txFontHeightF          = 0.04       ; Afeta ao mesmo tempo o tamanho
                                        ; de todos os
                                        ; títulos (esquerdo, central
                                        ; e direito).

plot = gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico.

end
```

O resultado será:

Alterar o tamanho da fonte dos títulos do lado esquerdo, central e direito:

```
; Nome do script: cap20_ex31.ncl

begin

f = addfile("../../dados/SA.CRU.anom.prec.2013.nc","r")

ppt = f->pre ; Importação da variável.

years  = ispan(1,12,1) ; Cria os valores do eixo x.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex31")

res                            = True       ; Habilita personalização.
res@vpWidthF                   = 0.8        ; Largura da figura.
res@vpHeightF                  = 0.3        ; Altura da figura.
res@trYMinF                    = -1.0       ; Mínimo valor do eixo y.
res@trYMaxF                    = 1.0        ; Máximo valor do eixo y.
res@trXMinF                    = min(years) ; Mínimo valor do eixo x.
res@trXMaxF                    = max(years) ; Máximo valor do eixo x.
res@gsnXYBarChart              = True       ; Habilita gráfico de barras.
res@gsnYRefLine                = 0.         ; Linha de referência no
                                            ; valor zero.
res@gsnAboveYRefLineColor      = "Red"      ; Cor vermelho para valores
positivos.
```

```
res@gsnBelowYRefLineColor       = "Blue"     ; Cor azul para valores negativos.
res@gsnLeftString               = "Esquerdo" ; Adiciona texto do lado
                                             ; esquerdo.
res@gsnCenterString             = "Centro"   ; Adiciona texto no centro.
res@gsnRightString              = "Direito"  ; Adiciona texto do lado
                                             ; direito.
res@gsnLeftStringFontHeightF    = 0.02       ; Tamanho da fonte do título
                                             ; do lado esquerdo.
res@gsnCenterStringFontHeightF  = 0.025      ; Tamanho da fonte do título
                                             ; central.
res@gsnRightStringFontHeightF   = 0.03       ; Tamanho da fonte do título
                                             ; do lado direito.

plot = gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico.

end
```

O resultado será:

Personalizando o título central da figura:

```
; Nome do script: cap20_ex32.ncl

begin

f = addfile("../../dados/SA.CRU.anom.prec.2013.nc","r")

ppt = f->pre ; Importação da variável.

years  = ispan(1,12,1) ; Cria os valores do eixo x.

wks  = gsn_open_wks("pdf","../../figuras/cap20/cap20_ex32")

plot = new(2,graphic) ; Arranjo gráfico para a criação do painel com
                      ; as duas figuras.

res                       = True       ; Habilita personalização.
res@gsnDraw               = False      ; Não gera a figura.
res@gsnFrame              = False      ; Não avança o frame.
res@vpWidthF              = 0.8        ; Largura da figura.
res@vpHeightF             = 0.3        ; Altura da figura.
res@trYMinF               = -1.0       ; Mínimo valor do eixo y.
res@trYMaxF               = 1.0        ; Máximo valor do eixo y.
```

```
res@trXMinF                = min(years) ; Mínimo valor do eixo x.
res@trXMaxF                = max(years) ; Máximo valor do eixo x.
res@gsnXYBarChart          = True       ; Habilita gráfico de barras.
res@gsnYRefLine            = 0.         ; Linha de referência no
                                        ; valor zero.
res@gsnAboveYRefLineColor = "Red"       ; Cor vermelho para valores positivos.
res@gsnBelowYRefLineColor = "Blue"      ; Cor azul para valores negativos.
res@gsnLeftString          = "Esquerdo" ; Adiciona texto do lado
                                        ; esquerdo da figura.
res@gsnCenterString        = "Centro"   ; Adiciona texto no centro
                                        ; da figura.
res@gsnRightString         = "Direito"  ; Adiciona texto do lado direito
                                        ; da figura.
res@gsnStringFontHeightF   = 0.03       ; Altera o tamanho da fonte
                                        ; dos títulos esquerdo, central
                                        ; e direito simultaneamente.
res@tiYAxisOn              = False      ; Desliga o título do eixo y.

plot(0) = gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico.

; Desloca a posição do texto da parte central na direção y. Caso fosse
; necessário alterar o lado direito ou esquerdo, basta alterar o nome
; para Right ou Left, respectivamente.

res@gsnCenterStringOrthogonalPosF = 0.08

plot(1) = gsn_csm_xy(wks,years,ppt(:,0,0),res) ; Gera o gráfico.

panres                             = True
panres@gsnPanelYWhiteSpacePercent = 10.0

gsn_panel(wks,plot,(/2,1/),panres)

end
```

O resultado será:

21 Geração de gráficos

O objetivo deste capítulo consiste em disponibilizar scripts prontos para que sejam alterados.

O site abaixo é uma excelente dica:

http://www.ncl.ucar.edu/Document/Manuals/Getting_Started

Cada usuário terá uma necessidade diferente na hora de gerar o seu gráfico, isto é, gráfico de linha, barra, um gráfico com duas linhas, com dois eixos, dentre outras possibilidades. Os scripts serão comentandatos para facilicar o entedimento.

As informações abaixo fornecem algumas dicas sobre os elementos que compõem um gráfico.

O comando abaixo:

ncl_filedump prec.2014.nc

mostra algumas informações do arquivo NetCDF que serão inseridas automaticamente no gráfico por meio das funções que utilizam "_csm" no nome. Caso as informações das dimensões lat e lon tenham o atributo long_name, elas serão inseridas no gráfico. Nesse caso, será inserido o nome (long_name) longitude (tiYAxisString, Figura 3) no eixo x e o nome (long_name) latitude (tiXAxisString, Figura 3) no eixo y conforme as informações abaixo destacadas em azul.

No caso da variável a ser visualizada (precip), o seu atributo long_name com o nome Average Monthly Rate of Precipitation será adicionado ao gráfico no lado esquerdo (gsnLeftString) como mostra a Figura 21.2.1. O mesmo ocorrerá para a sua unidade que tem o atributo units igual a mm/day que será adicionada no lado direito da figura (gsnRightString). Por isso, é recomendável utilizar o "_csm" por essas facilidades.

```
variables:
   float lon ( lon )
      standard_name :         longitude
      long_name :     longitude   tiYAxisString
      units :         degrees_east
      axis : X

   float lat ( lat )
      standard_name :         latitude
      long_name :     latitude    tiXAxisString
      units :         degrees_north
      axis : Y

   double time ( time )
      standard_name :         time
      long_name :     Time
      units :         days since 1800-1-1 00:00:00
      calendar :      standard
      axis : T

   float precip ( time, lat, lon )                gsnLeftString
      long_name :     Average Monthly Rate of Precipitation
      units :         mm/day   gsnRightString
      _FillValue :    -9.96921e+36
      missing_value :         -9.96921e+36
      precision :     32767
      least_significant_digit :       2
      var_desc :      Precipitation
      dataset :       GPCP Version 2.2 Combined Precipitation
      level_desc :    Surface
      statistic :     Mean
      parent_stat :   Mean
      actual_range : (  0, 9.985388e+29 )
```

A Figura 3 mostra alguns elementos que compõem um gráfico. As informações servem para gráficos 1D e 2D. Todos esses elementos podem ser personalizados via recursos.

Figura 3: Elementos que compõem um gráfico.

21.1 O que preciso saber para criar um gráfico?

É aconselhavável conhecer a estrutura do dado, se há necessidade de descompactação, se o dado está completo ou não para ser utilizado em alguma função, se há alguma limitação. Esse passo inicial é útil para evitar problemas posteriores, por isso, conhecer bem o dado é fundamental. O NCL possui uma ferramenta chamada ncl_filedump similar ao ncdump da biblioteca NetCDF, muito útil para explorar o conteúdo de um arquivo. Tendo esse conhecimento básico é possível avançar para a criação das figuras.

Inicialmente, o usuário precisar saber qual gráfico será gerado, por exemplo:

- gráfico de linha,
- gráfico de barra,
- gráfico de isolinhas (pressão, temperatura, precipitação e etc),

- gráfico de dispersão,
- gráfico de seção vertical,
- dentre outras possibilidades.

Para tal, foram selecionados dois links específicos para geração de figuras. Nesses links estão todas as possibilidades gráficas.

1. http://www.ncl.ucar.edu/Applications/list_ptypes.shtml
2. http://www.ncl.ucar.edu/Applications

Uma excelente dica é acrescentar no topo do seu script as bibliotecas abaixo. Elas servem para personalização dos gráficos, uso de funções, dentre outras possibilidades. Não deixe de explorar o seu conteúdo.

Os exemplos a seguir não possuem essas bibliotecas, mas entende-se que elas são previamente lidas.

A partir da versão 6.4.0 os scripts destacados em vermelho são lidos automaticamente.

```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_csm.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/contributed.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/shea_util.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/contrib/time_axis_labels.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/skewt_func.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/contrib/acentos.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/wind_rose.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/shapefile_utils.ncl"
```

Nas versões mais recentes do NCL a maiorias dos procedimentos e funções são lidas automaticamente, mas há algumas funções que ainda necessitam que essas bibliotecas sejam carregadas, por isso, adicione essas linhas no topo do script.

21.1.1 Tipos de gráficos de linha

Links para gráficos de linha:

- http://www.ncl.ucar.edu/Applications/generic_xy.shtml
- http://www.ncl.ucar.edu/Applications/xy.shtml

Para o gráfico de linha há algumas possibilidades de criação, são elas:

- Função: **gsn_csm_xy** ⇒ Ao usar essa função é necessário fornecer informações dos eixos x e y.
- Função: **gsn_xy** ⇒ A diferença em relação ao anterior é que há poucas possibilidades de formatação do gráfico. Observe que não tem o csm.

- Função: **gsn_csm_xy2** ⇒ Gráfico de linha com dois eixos y.
- Função: **gsn_csm_xy3** ⇒ Gráfico de linha com três eixos y.
- Função: **gsn_csm_x2y2** ⇒ Gráfico de linha com dois eixos x e dois eixos y.
- Função: **gsn_csm_x2y** ⇒ Gráfico de linha com dois eixos x.
- Função: **gsn_y** ⇒ Gráfico de linha com eixo y.

Exemplo de uso: Criação de um gráfico de linha.

plot = gsn_csm_xy(wks,ispan(1,12,1),u(0:11,{1000},{-2},{300}),False)

Nesse exemplo, os argumentos dos eixos x e y são necessários. Onde: x = ispan(1,12,1) e y = u(0:11,{1000},{-2},{300}).

A variável u possui quatro dimensões, isto é, tempo (0:11), nível vertical ({1000}), latitude ({-2}) e longitude ({300}).

Para facilitar a criação dos gráficos, utilize sempre as funções que apresentam csm no nome, pois elas possuem maiores possibilidades de personalização do gráfico.

21.1.2 Tipos de gráficos de barras

O gráfico de barras utiliza algumas opções que são empregadas no gráfico de linha, e para sua criação, basta definir gsnXYBarChart para True.

Exemplo: res@gsnXYBarChart = True

21.1.3 Tipos de gráficos espaciais

- Função: **gsn_csm_contour_map_ce** ⇒ Gera mapa de isolinha.
- Função: **gsn_csm_streamline_map_ce** ⇒ Gera mapa de linhas de corrente (as componentes zonal e meridional do vento são necessárias).
- Função: **gsn_csm_streamline_contour_map_ce** ⇒ Gera mapa de linhas de corrente e um outro campo escalar (sobreposição de campos).
- Função: **gsn_csm_vector_map_ce** ⇒ Gera mapa de vetores do vento.
- Função: **gsn_csm_vector_scalar_map_ce** ⇒ Gera mapa de vetor do vento e um outro campo escalar (sobreposição de campos).

O **CE** quer dizer Cylindrical Equidistant (CE) Projections.

Exemplo: A variável t é do tipo: t(time,level,latitude,longitude).

plot = gsn_csm_contour_map_ce(wks,t(0,{1000},:,:),False)

21.2 Gráficos de linha

Dicas: Links para todos os recursos utilizados na personalização dos gráficos.

http://www.ncl.ucar.edu/Document/Graphics/Resources/gsn.shtml

http://www.ncl.ucar.edu/Document/Graphics/Resources/list_alpha_res.shtml

21.2.1 Gráficos de linha com um eixo y

Os gráficos de linha podem ser visualizados nos links abaixo. Com esses links o usuário será capaz de criar seus próprios scripts e gerar suas figuras.

http://www.ncl.ucar.edu/Applications/xy.shtml

http://www.ncl.ucar.edu/Applications/generic_xy.shtml

Para gerar um gráfico de linha são necessárias algumas informações, como por exemplo, o título do gráfico, 0 título para os eixos x e y, a cor da linha, se terá marcadores ou não, a espessura da linha, o estilo de linha (contínua ou tracejada) e a legenda.

Com o script abaixo será criada uma série temporal de temperatura em Kelvin. O arquivo corresponde a dois anos de dados mensais, ou seja, apresenta 24 tempos, porém serão mostrados apenas os 12 primeiros.

Execute o script cap21_ex01.ncl digitando no seu terminal Linux:

ncl cap21_ex01.ncl

```
1   ; Nome do script: cap21_ex01.ncl
2
3   begin
4
5   f = addfile("../../dados/tair.2011.2012.nc","r")
6   t = short2flt(f->air)
7   wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex01")
8   plot = gsn_csm_xy(wks,ispan(1,12,1),t(0:11,{1000},{-2},{300}),False)
9
10  end
```

Descrição do script:

Um script é composto por *begin* e *end* (linhas 3 e 10, respectivamente) e as instruções e comandos ficam entre elas. Nas versões mais recentes do NCL, não é mais necessário inserir o *begin* e o *and*, isso ficará a critério de cada inserir ou não.

A linha 5 corresponde a abertura do arquivo para somente leitura "r".

A linha 6 é a atribuição da variável do arquivo. Foi usada a função short2flt para descompactar o dado. Isso foi feito porque o dado é do tipo short.

Na linha 7 está a extensão (pdf) e o diretório onde será criada a figura com o nome cap21_ex01.

A linha 8 é a criação do gráfico (plot) por meio da função gsn_csm_xy. Foram selecionados os doze primeiros meses (0:11). Lembrando que no NCL o primeiro

índice inicia em 0 e não em 1.

A função ispan (linha 8) cria um vetor de valores inteiros com 12 posições.

Lembrando que serão mostrados os 12 primeiros tempos (0:11), primeiro nível vertical que corresponde a 1000hPa, latitude fixa em 2^oS e longitude fixa em 300. A palavra False (linha 8) indica que não está sendo aplicada nenhuma formatação ao gráfico.

O resultado será:

Observe que o gráfico não é o que se deseja, pois ele apresenta uma formatação bruta. Contudo, pode-se formatar a figura habilitando o recurso para personalização que assume o valor True. Na linha 8 do script acima, nota-se que há um False, o que significa que o gráfico não apresenta qualquer tipo de formatação.

Agora, altera-se essa variável para True para atribuir a formatação ao gráfico.

Inicialmente, serão atribuídos os títulos a figura usando as funções tiMainString, tiXAxisString e tiYAxisString. Essas funções são responsáveis por incluir o título principal na figura e os títulos para os eixos x e y, respectivamente.

O script abaixo adicionará os títulos. Note que será usado o mesmo script com a diferença que serão acrescentadas informações para formatar o gráfico.

```
; Nome do script: cap21_ex02.ncl

begin

f = addfile("../../dados/tair.2011.2012.nc","r")

u = short2flt(f->air)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex02")

res                = True
res@tiMainString   = "Temperatura do Ar"
res@tiXAxisString  = "M"+ecirc+"s"
res@tiYAxisString  = "Temperatura (K)"

plot = gsn_csm_xy(wks,ispan(1,12,1),u(0:11,{1000},{-2},{300}),res)

end
```

Ao executar esse script será gerada a figura abaixo.

Aplicando formatação aos eixos x e y do gráfico.

```
1   ; Nome do script: cap21_ex03.ncl
2
3   begin
4
5   f = addfile("../../dados/tair.2011.2012.nc","r")
6
7   u = short2flt(f->air)
8
9   u = u-273.15 ; Converte de Kelvin para Celsius.
10
11  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex03")
12
13  res               = True
14  res@tiMainString  = "Temperatura do ar em 2~S~o~N~S e 300~S~o~N~W"
15  res@tiXAxisString = "M"+ecirc+"s"
16  res@tiYAxisString = "Temperatura (~S~o~N~C)"
17  res@trXMinF       = 1.0
18  res@tmXBMode      = "Explicit"
19  res@tmXBValues    = (/1,2,3,4,5,6,7,8,9,10,11,12/)
20  res@tmXBLabels    = (/"J","F","M","A","M","J","J","A","S","O","N","D"/)
21  res@trYMinF       =  24.0
22  res@trYMaxF       =  27.0
23
24  plot = gsn_csm_xy(wks,ispan(1,12,1),u(0:11,{1000},{-2},{300}),res)
25
26  end
```

Execute este script no seu terminal Linux.

Na linha 18 o eixo x é formatado de forma explícita. Há outras formas de formatação disponíveis em:

http://www.ncl.ucar.edu/Document/Graphics/Resources/tm.shtml#tmXBMode.

Na linha 19, utiliza-se apenas 12 meses, isso corresponde a posição no eixo x para cada mês e cada posição recebe uma string (linha 20).

Nas linhas 21 e 22 são fixados os valores mínimo e máximo, respectivamente.

O resultado do script será a figura abaixo.

Após adicionar os títulos ao gráfico e formatar os eixos x e y, falta personalizar a linha e adicionar uma legenda.

```
; Nome do script: cap21_ex04.ncl

begin

f = addfile("../../dados/tair.2011.2012.nc","r")

u = short2flt(f->air)

u = u-273.15 ; Converte de Kelvin para Celsius.

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex04")

res                          = True
res@tiMainString             = "Temperatura do ar em 2~S~o~N~S" + \ +
                               "e 300~S~o~N~W"
res@tiXAxisString            = "M"+ecirc+"s"
res@tiYAxisString            = "Temperatura (~S~o~N~C)"
res@trXMinF                  = 1.0
res@tmXBMode                 = "Explicit"
res@tmXBValues               = (/1,2,3,4,5,6,7,8,9,10,11,12/)
res@tmXBLabels               = (/"J","F","M","A","M","J","J","A", \
                                 "S","O","N","D"/)
res@trYMinF                  =  24.0
res@trYMaxF                  =  27.0
res@xyLineThicknesses        = 5.0
res@xyLineColors             = "red"
```

```
27  res@xyMarkLineMode          = "MarkLines"
28  res@xyMarkers               = 16
29  res@xyMarkerColors          = "red"
30  res@xyDashPattern           = 15
31  res@xyMarkerSizeF           = 0.02
32  res@pmLegendDisplayMode     = "Always"
33  res@pmLegendSide            = "Top"
34  res@pmLegendParallelPosF    = 0.22
35  res@pmLegendOrthogonalPosF  = -0.3
36  res@pmLegendWidthF          = 0.15
37  res@pmLegendHeightF         = 0.18
38  res@lgPerimOn               = False
39  res@lgLabelFontHeightF      = 0.02
40  res@xyExplicitLegendLabels  = "Temperatura"
41
42  plot = gsn_csm_xy(wks,ispan(1,12,1),u(0:11,{1000},{-2},{300}),res)
43
44  end
```

Execute este script no seu terminal Linux.

Nesse novo script foram adicionadas novas formatações referente à linha do gráfico.

- Linha 25: é a espessura da linha.
- Linha 26: representa a cor da linha.
- Linha 27: o usário escolhe se deseja personalizar a linha, isto é, somente linha (Lines), linha com marcadores (MarkeLines) ou apenas os marcadores (Markers).
- Linha 28: seleciona o tipo de marcador, uma vez que foi escolhido Marklines.
- Linha 29: é a cor do marcador selecionado.
- Linha 30: é o estilo de linha usado.
- Linha 31: é o tamanho do marcador.
- Linha 32: mostra a legenda.
- Linha 33: localização da legenda.
- Linha 34: é o deslocamento do legenda para direita (maiores valores) ou esquerda (menores valores).
- Linha 35: desloca para cima (maiores valores) ou para baixo (menores valores).
- Linhas 36 e 37: referem-se a largura e altura da legenda, respectivamente.
- Linha 38: habilita (True) ou não (False) o contorno em volta da legenda.
- Linha 39: é o tamanho da fonte da legenda.
- Linha 40: é o nome que aparecerá na legenda.

O resultado do script será:

Finalmente, foi gerado o primeiro gráfico de linha com algumas formatações. É possível aplicar mais formatações ao gráfico. O usuário deve pesquisar outras opções de acordo com o seu interesse.

Os links abaixo possuem outras possibilidades de formatação para gráficos de linha.

http://www.ncl.ucar.edu/Applications/xy.shtml.

http://www.ncl.ucar.edu/Applications/generic_xy.shtml.

21.2.2 Gráfico de linha com três eixos y

```
; Nome do script: cap21_ex05.ncl

begin

; Abertura do arquivo txt com 12 linhas e 6 colunas do tipo float.
f = asciiread("../../dados/barcarena.txt",(/12,6/),"float")


prec = f(:,0)  ; Importação da precipitação (mm/mês).
u    = f(:,4)  ; Importação da componente zonal do vento (m/s).
v    = f(:,5)  ; Importação da componente meridional do vento (m/s).
vel  = wind_speed(u,v) ; Calcula a velocidade do vento (m/s) dado u e v.

; Criação de uma variável que contém as 3 colunas de temperatura
; (mínima, média e máxima em Celsius).

; temp é nome da variável que vai armazenar as 3 colunas de temperatura.
; As dimensões são: 3 colunas de temperatura com 12 o valores mensais
; cada uma.
temp = new((/3,12/),float)
temp(0,:) = f(:,1) ; A segunda coluna [f(:,1)] do arquivo f é igual a
                   ; temperatura mínima.
temp(1,:) = f(:,2) ; A terceira coluna [f(:,2] do arquivo f é igual a
                   ; temperatura média.
temp(2,:) = f(:,3) ; A quarta coluna [f(:,3)] do arquivo f é igual a
                   ; temperatura máxima.

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex05")

; Personalização do eixo direito do gráfico (temp. mín, med e máx).
resD1                        = True ; Habilita para formatação.
resD1@xyDashPattern          = 0    ; Tipo de linha do gráfico.
resD1@xyMarkLineMode         = "MarkLines"  ; Gráfico com marcadores.
resD1@xyMarkers              = (/16,16,16/) ; Tipo de marcador para cada
                                            ; curva.
resD1@xyMarkerColors         = (/"blue","black","red"/)
resD1@xyLineColors           = (/"blue","black","red"/)
resD1@xyLineThicknesses      = (/2,2,2/)
resD1@xyLineColors           = (/"blue","black","red"/)
resD1@xyExplicitLegendLabels = (/"Tm"+iacute+"n","Tm"+eacute+"d", \
                                 "Tm"+aacute+"x"/)
resD1@trYMaxF                = 34.0    ; Valor máximo do eixo y.
resD1@trYMinF                = 20.0    ; Valor mínimo do eixo y.
resD1@pmLegendDisplayMode    = "Always"; Habilita a legenda.
resD1@pmLegendWidthF         = 0.1     ; Largura da legenda.
resD1@pmLegendHeightF        = 0.1     ; Altura da legenda.
resD1@pmLegendOrthogonalPosF = -1.07   ; Deslocamento da legenda
                                       ; para cima ou para baixo.
resD1@pmLegendParallelPosF   =  0.12   ; Deslocamento da legenda
                                       ; para direita ou esquerda.
resD1@lgPerimOn              = False   ; Sem linhas em torno da legenda.
resD1@lgLabelFontHeightF     = 0.02    ; Tamanho da fonte da legenda.
resD1@tmYRMinorOn            =  False  ; Desabilita os traços
                                       ; secundários do eixo (Y) da
                                       ; direita (R).
resD1@tiYAxisOffsetXF        = -0.013  ; Deslocamento do título do eixo
                                       ; y para direita ou esquerda.
resD1@tiMainString           = "Climatologia para Barcarena/PA"
resD1@tiYAxisString          = "Temperatura (~S~o~N~C)"

```

```
61   ; Personalização do eixo direito do gráfico (velocidade do vento).
62   resD2                          = True
63   resD2@xyDashPattern            = 0
64   resD2@xyMarkLineMode           = "MarkLines"
65   resD2@xyMarkers                = 16
66   resD2@xyMarkerColors           = "green"
67   resD2@xyLineColors             = "green"
68   resD2@tiYAxisString            = "Velocidade do vento (m s ~S~-1~N~)"
69   resD2@xyLineThicknesses        = 2
70   resD2@trYMaxF                  = 3.0
71   resD2@trYMinF                  = 0.0
72   resD2@xyLineColors             = "green"
73   resD2@tmYRMinorOn              =  False
74   resD2@pmLegendDisplayMode      = "NoCreate" ; Não cria legenda.
75   resD2@tiYAxisFontColor         = "green"
76   resD2@tmYRLabelFontColor       = "green"
77
78   ; Personalização do eixo esquerdo do gráfico (precipitação).
79   resE                           = True
80   resE@gsnXYBarChart             = True       ; Habilita gráfico de barras.
81   resE@gsnXYBarChartColors       = "orange"   ; Cor da barra.
82   resE@trYMaxF                   = 600.0
83   resE@trYMinF                   = 0.0
84   resE@tmYLMode                  = "Manual"   ; Formatação do eixo y esquerdo.
85   resE@tmYLTickStartF            = resE@trYMinF
86   resE@tmYLTickSpacingF          = 50
87   resE@tmYLTickEndF              = resE@trYMaxF
88   resE@tmXBMode                  = "Explicit" ; Formatação do eixo x do
89                                               ; meu jeito.
90   ; Valores necessários para o eixo x. São os valores dos meses.
91   resE@tmXBValues                = (/1,2,3,4,5,6,7,8,9,10,11,12/)
92   ; Cada valor acima recebe uma string.
93   resE@tmXBLabels                = (/"J","F","M","A","M","J","J","A","S",\
94                                     "O","N","D"/)
95   resE@tiYAxisString             = "Precipita"+cedil+atilde+ \
96                                    "o (mm m"+ecirc+"s~S~-1~N~)"
97   resE@gsnXYBarChartBarWidth     = 0.75
98   resE@tmYLMinorOn               = False
99   resE@gsnDraw                   = False
100  resE@gsnFrame                  = False
101  resE@vpWidthF                  = 0.55
102  resE@vpHeightF                 = 0.45
103  resE@tiYAxisFontColor          = "orange"
104  resE@tmYLLabelFontColor        = "orange"
105
106  ; Plota os dados. A variável prec será personalizada com a opção resE,
107  ; temp com a opção resD1 e vel com a opção resD2.
108  ; Essa ordem é muito importante.
109
110  plot = gsn_csm_xy3(wks,ispan(1,12,1),prec,temp,vel,resE,resD1,resD2)
111
112  draw(plot)
113
114  end
```

O resultado será:

21.2.3 Gráfico de linha com dois eixos x e y

```
; Nome do script: cap21_ex06.ncl

begin

; double pre(time, lat, lon)
f1 = addfile("../../dados/SA.CRU.prec.1979.2013.nc","r")
; float precip(time, lat, lon)
f2 = addfile("../../dados/SA.GPCP.prec.1979.2013.nc","r")

; f1 =  Dimensions and sizes: [time | 420] x [lat | 1] x [lon | 1]
; f2 =  Dimensions and sizes: [time | 420] x [lat | 1] x [lon | 1]

; O dado possui time,lat,lon apesar de ser um dado pontual, é necessário
; fixar a lat e a lon para que ocorra a redução de dimensão de 3 para
; apenas 1, no caso, a dimensão time. Por isso, é importante conhecer
; o dado antes de realizar qualquer tarefa.

;*********************************************************************
; Leitura da dimensão tempo. Será utilizada para criar o eixo x
;*********************************************************************
time     = f1->time
TIME     = cd_calendar(time, 0) ; Tipo float.
year     = toint(TIME(:,0))     ; Guarda o ano.
month    = toint(TIME(:,1))     ; Guarda o mês.
day      = toint(TIME(:,2))     ; Guarda o dia.
ddd      = day_of_year(year,month,day) ; Retorna o dia do ano no formato
                                       ; 365 ou 366 dias.
YYYYMM   = year*100 + month
anoi     = 197901               ; Data inicial no formato AAAAMM.
anof     = 198012               ; Data final no formato AAAAMM.
datai    = ind(YYYYMM.eq.anoi)  ; Índice que será utilizado no tempo
dataf    = ind(YYYYMM.eq.anof)  ; para selecionar o período de interesse.

x1 = ispan(datai,dataf,1)       ; Valores do eixo x1.
x2 = ispan(datai,dataf,1)       ; Valores do eixo x2.

y1 = f1->pre(datai:dataf,0,0)    ; Seleciona o tempo de interesse para y1
y2 = f2->precip(datai:dataf,0,0); e y2. Houve redução de dimensão de 3
                                 ; para 1, pois o objetivo é gerar uma
                                 ; série temporal.

```

```
printVarSummary(y1) ; Dimensions and sizes: [time | 12]
printVarSummary(y2) ; Dimensions and sizes: [time | 12]

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex06")

; Recursos para o eixo x (eixo inferior) e y (eixo da esquerda).
resEB                     = True
resEB@tmXBLabelAngleF     = -90           ; Inclinação dos rótulos do
                                          ; eixo x.
resEB@tmXBLabelJust       = "CenterRight" ; Alinhamento dos rótulos do
                                          ; eixo x.
resEB@tmXBLabelStride     = 2             ; Ajusta quantos valores serão
                                          ; mostrados no eixo x.
resEB@trXMinF             = datai         ; Mínimo valor do eixo x.
resEB@trXMaxF             = dataf         ; Máximo valor do eixo x.
resEB@xyMarkLineModes     = "MarkLines"   ; Linha com marcadores.
resEB@xyMarkers           =  16           ; Tipo de marcador.
resEB@xyMarkerColor       = "red"         ; Cor do marcador.
resEB@xyLineColors        = "red"         ; Cor da linha.
resEB@xyMarkerSizeF       = 0.02          ; Tamanho do marcador.
resEB@trYMaxF             = 7.0           ; Máximo valor do eixo y.
resEB@trYMinF             = 0.0           ; Mínimo valor do eixo y.
resEB@tmXBMode            = "Explicit"    ; Formata o eixo x do meu jeito.
resEB@tmXBValues          = ispan(datai,dataf,1)
resEB@tmXBLabels          = YYYYMM(datai:dataf)
resEB@tiYAxisString       = "Precipita"+cedil+atilde+ \
                            "o CRU (mm dia ~S~-1~N~)"
resEB@tiYAxisFontColor    = "red"         ; Cor da fonte do eixo y.
resEB@tiYAxisFontColor    = "red"         ; Cor da fonte do eixo y.
resEB@tmYLLabelFontColor  = "red"         ; Cor dos valores do eixo y.
resEB@tmXBLabelFontColor  = "red"         ; Cor dos valores do eixo x.
resEB@tiMainString        = "Compara"+cedil+atilde+ \
                            "o entre a PREC do CRU e do GPCP"

; Recursos para o eixo x (eixo superior) e y (eixo da direita).
resDT                      = True
resDT@tmXTLabelAngleF      = -270
resDT@tmXTLabelJust        = "CenterRight"
resDT@tmXTLabelStride      = 2
resDT@trXMinF              = datai
resDT@trXMaxF              = dataf
resDT@xyMarkLineModes      = "MarkLines"
resDT@xyMarkers            =  12
resDT@xyMarkerColor        = "blue"
resDT@xyLineColors         = "blue"
resDT@xyMarkerSizeF        = 0.02
resDT@trYMaxF              = 7.0
resDT@trYMinF              = 0.0
resDT@tmXTMode             = "Explicit"
resDT@tmXTValues           = ispan(datai,dataf,1)
resDT@tmXTLabels           = YYYYMM(datai:dataf)
resDT@tiYAxisString        = "Precipita"+cedil+atilde+ \
                             "o GPCP (mm dia ~S~-1~N~)"
resDT@tiYAxisFontColor     = "blue"
resDT@tmYRLabelFontColor   = "blue"
resDT@tmXTLabelFontColor   = "blue"

; O eixo x1 (x2) está associado com o y1 (y2).

plot = gsn_csm_x2y2(wks,x1,x2,y1,y2,resEB,resDT)

end
```

O resultado será:

21.2.4 Gráfico de linha com um eixo x e dois eixos y

```
; Nome do script: cap21_ex07.ncl

begin

f1 = addfile("../../dados/SA.olr.2000.2004.nc","r")
f2 = addfile("../../dados/SA.GPCP.prec.2000.2004.nc","r")

; f1 =  Dimensions and sizes: [time | 60] x [lat | 1] x [lon | 1]
; f2 =  Dimensions and sizes: [time | 60] x [lat | 1] x [lon | 1]

; O dado possui time,lat,lon apesar de ser um dado pontual,
; é necessário fixar a lat e a lon para que ocorra a redução de dimensão
; de 3 para apenas 1, no caso, a dimensão time.

;*********************************************************************
; Leitura da dimensão tempo. Será utilizada para criar o eixo x
;*********************************************************************
time    = f1->time
TIME    = cd_calendar(time, 0) ; Tipo float.
year    = toint(TIME(:,0))     ; Guarda o ano.
month   = toint(TIME(:,1))     ; Guarda o mês.
day     = toint(TIME(:,2))     ; Guarda o dia.
ddd     = day_of_year(year,month,day) ; Retorna o dia do ano no formato
                                      ; 365 ou 366 dias.
```

```
YYYYMM  = year*100 + month
anoi    = 200001                ; Data inicial no formato AAAAMM.
anof    = 200012                ; Data final no formato AAAAMM.
datai   = ind(YYYYMM.eq.anoi)   ; Índice que será utilizado na
                                ; dimensão tempo para selecionar o
dataf   = ind(YYYYMM.eq.anof)   ; período de interesse.

x  = ispan(datai,dataf,1)       ; Valores do eixo x.

y1 = f1->olr(datai:dataf,0,0)    ; Seleciona o tempo de interesse para y1
y2 = f2->precip(datai:dataf,0,0); e y2. Houve redução de dimensão de 3
                                  ; para 1, pois o objetivo é gerar uma
                                  ; série temporal.

printVarSummary(y1) ; Dimensions and sizes: [time | 12]
printVarSummary(y2) ; Dimensions and sizes: [time | 12]

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex07")

; Recursos para o eixo x e y (eixo da esquerda).
resEB                   = True
resEB@tmXBLabelAngleF   = -90   ; Inclinação dos rótulos do eixo x.
resEB@tmXBLabelJust     = "CenterRight"; Alinhamento dos rótulos do
                                       ; eixo x.
resEB@tmXBLabelStride   = 1            ; Ajusta quantos valores serão
                                       ; mostrados no eixo x.
resEB@trXMinF           = datai        ; Mínimo valor do eixo x.
resEB@trXMaxF           = dataf        ; Máximo valor do eixo x.
resEB@xyMarkLineModes   = "MarkLines"  ; Linha com marcadores.
resEB@xyMarkers         =  16          ; Tipo de marcador.
resEB@xyMarkerColor     = "red"        ; Cor do marcador.
resEB@xyLineColors      = "red"        ; Cor da linha.
resEB@xyMarkerSizeF     = 0.015        ; Tamanho do marcador.
resEB@trYMaxF           = 260.0        ; Máximo valor do eixo y.
resEB@trYMinF           = 220.0        ; Mínimo valor do eixo y.
resEB@tmXBMode          = "Explicit"   ; Formata o eixo x do meu jeito.
resEB@tmXBValues        = ispan(datai,dataf,1)
resEB@tmXBLabels        = YYYYMM(datai:dataf:1)
resEB@tiYAxisString     = "Radia"+cedil+atilde+ \
                          "o de Onda Longa (W m~S~-2~N~)"
resEB@tiYAxisFontColor  = "red"         ; Cor da fonte do eixo y.
resEB@tmYLLabelFontColor = "red"        ; Cor dos valores do eixo y.
resEB@tiMainString      = "ROL (NOAA) x PREC (GPCP)"

; Recursos para o eixo x e y (eixo da direita).
resDT                     = True
resDT@xyMarkLineModes     = "MarkLines"
resDT@xyMarkers           =  12
resDT@xyMarkerColor       = "blue"
resDT@xyLineColors        = "blue"
resDT@xyMarkerSizeF       = 0.015
resDT@trYMaxF             = 5.0
resDT@trYMinF             = 0.0
resDT@tiYAxisString       = "Precipita"+cedil+atilde+ \
                            "o GPCP (mm dia ~S~-1~N~)"
resDT@tiYAxisFontColor    = "blue"
resDT@tmYRLabelFontColor  = "blue"

; O eixo x é o mesmo para as duas curvas (y1 e y2).
plot = gsn_csm_xy2(wks,x,y1,y2,resEB,resDT)

end
```

O resultado será:

21.2.5 Gráfico de linha com tendência linear

```
; Nome do script: cap21_ex08.ncl

begin

arquivo = "co2.1980.2005.RCP85.txt" ; Nome do arquivo.

nlin = numAsciiRow("../../dados/" + arquivo)
ncol = numAsciiCol("../../dados/" + arquivo)

f = asciiread("../../dados/" + arquivo,(/nlin,ncol/),"float")

x  = toint(f(:,0)) ; x é um vetor float de anos que foi convertido para
                   ; inteiro (toint).
y  = f(:,1)        ; y (CO2) é um vetor float.

y@_FillValue = -999 ; Defino o valor indefinido caso haja no arquivo.

nanos = max(x)-min(x) + 1 ; Total de anos.

rc = regline(x,y) ; Calcula a regressão linear.
r  = escorc(x,y)  ; Correlação de Pearson.
r2 = r*r          ; Coeficiente de determinção.

printVarSummary(rc)

```

```
26  ; Parte do resultado do comando printVarSummary(rc):
27  ;
28  ;Number Of Attributes: 7
29  ;  _FillValue :   -999
30  ;  yintercept :   -3161.886
31  ;  yave :   366.5367
32  ;  xave :   1997.5
33  ;  nptxy :  36
34  ;  rstd :   0.02959174
35  ;  tval :   59.693
36
37  ; O valor 2 abaixo significa quantas curvas serão geradas.
38
39  pltarry      = new ((/2,nanos/),typeof(y),y@_FillValue)
40  pltarry(0,:) = y  ; A primeira curva (0) é o valor de CO2.
41  pltarry(1,:) = rc*(x-rc@xave) + rc@yave ; A segunda curva (1)
42                                          ; é a reta de regressão.
43
44  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex08")
45
46  res                     = True
47  res@gsnDraw             = False
48  res@gsnFrame            = False
49  res@tiMainString        = "RCP8.5 CO~B~2~N~ (ppm): "+ min(x) + \
50                            " - " + max(x)
51  res@gsnMaximize         = True              ; A figura ocupa todo o
52                                              ; espaço da tela.
53  res@xyLineColors        = (/"blue","red"/) ; Cores das linhas.
54  res@xyMarkerSizeF       = 0.008             ; Tamanho do marcador.
55  res@xyDashPatterns      = (/0,0/)           ; Padrão da linha de cada
56                                              ; curva.
57  res@xyLineThicknesses   = (/1,3/)           ; Espessura de cada curva.
58  res@xyMarkLineModes     = (/"Markers","Lines"/) ; Estilo de linha de
59                                              ; cada curva.
60  res@trXMinF             = min(x)            ; Mínimo valor do eixo x.
61  res@trXMaxF             = max(x)            ; Máximo valor do eixo x.
62  res@trYMinF             = 300.0             ; Mínimo valor do eixo y.
63  res@trYMaxF             = 420.0             ; Máximo valor do eixo y.
64  res@tiXAxisString       = "Anos"            ; Título do eixo x.
65  res@tiYAxisString       = "CO~B~2~N~ (ppm)"; Título do eixo y.
66  res@tmXBLabelAngleF     = -90               ; Rotaciona em -90 graus os
67                                              ; rótulos de x.
68  res@xyMarkers           =   (/16,0/)        ; Tipo de marcador para cada
69                                              ; curva.
```

```
70   res@xyMarkerColors      = (/"blue","red"/) ; Cor do marcador para cada
71                                              ; curva.
72   res@tmXBLabelJust       = "CenterRight"    ; Alinhamento dos rótulos
73                                              ; do eixo x.
74   res@tmXBLabelStride     = 2                ; Ajusta quantos valores
75                                              ; serão mostrados no eixo x.
76   res@tmXBMode            = "Explicit"       ; Eixo x do meu jeito.
77   res@tmXBValues          = ispan(min(x),max(x),1) ; Valores do eixo x.
78   res@tmXBLabels          = ispan(min(x),max(x),1) ; Rótulo do eixos.
79
80   plot  = gsn_csm_xy(wks,x,pltarry,res) ; Plot da figura.
81
82   ; Inserindo a equação y = mx + b na posição x,y do gráfico.
83   ; Onde m = rc | b = yave.
84   ; Inseri também o R2.
85
86   txres1                        = True
87   txres1@txFontHeightF          = 0.025          ; Tamanho do texto.
88   txres1@txFontColor            = "red"          ; Cor do texto.
89   txres1@txBackgroundFillColor  = "transparent"  ; Fundo do texto. Pode
90                                                  ; usar "transparent".
91   txres1@txJust                 = "CenterLeft"   ; Alinhamento do texto.
92
93   ; "1981" é o valor do eixo x e o "412" valor do eixo y. Foi feita a
94   ; quebra de linha com o símbolo "\".
95   dum1 = gsn_add_text(wks,plot,"y = "+sprintf("%4.3f",rc)+"x + "+ \
96                                       sprintf("%4.3f",rc@yave),1981, \
97                                       412,txres1)
98
99   ; "1981" é o valor do eixo x e o "404" valor do eixo y.
100  dum2 = gsn_add_text(wks,plot,"R~S~2~N~ = "+sprintf("%4.3f",r2),1981, \
101                                            404,txres1)
102
103  ; Insere o marcador círculo azul na posição x,y do gráfico.
104
105  pmres               = True
106  pmres@gsMarkerColor = "blue" ; Cor do marcador.
107  pmres@gsMarkerIndex = 16     ; Tipo do marcador.
108  pmres@gsMarkerSizeF = 0.02   ; Tamanho do marcador.
109
110  dum3 = gsn_add_polymarker(wks,plot,1981.5,396,pmres)
111
112  ; Insere o texto CO2 azul na posição x,y do gráfico.
113
114  txres2                        = True
115  txres2@txFontHeightF          = 0.025          ; Tamanho do texto.
116  txres2@txFontColor            = "blue"         ; Cor do texto.
117  txres2@txBackgroundFillColor  = "transparent"  ; Fundo do texto. Pode
118                                                  ; usar "transparent".
119  txres2@txJust                 = "CenterLeft"   ; Alinhamento do texto.
120
121  ; "1982.5" é o valor do eixo x e o "395.5" valor do eixo y.
122  dum4 = gsn_add_text(wks,plot,"CO~B~2",1982.5,395.5,txres2)
123
124  draw(plot)
125
126  end
```

O resultado será:

21.2.6 Gráfico de linha com desvio padrão

```
; Nome do script: cap21_ex09.ncl

begin

f = addfile("../../dados/precip.mon.1979.2014.nc","r")

; Inversão da latitude de sul para norte.
ppt   = f->precip(:,::-1,:)

nlat   = dimsizes(ppt&lat) ; Extrai o número de pontos de latitude (72).

; Calcula a média zonal no primeiro tempo (0), com isso, a dimesão
; lon "some" restando apenas a dimensão time e lat.
ppt_mz = zonalAve(ppt(0,:,:))

printVarSummary(ppt_mz) ; Dimensions and sizes: [lat | 72]

sigma  = 2

xp     = new( (/sigma*nlat/), float ) ; Cria um vetor com 144 posições.
yp     = new( (/sigma*nlat/), float ) ; Cria um vetor com 144 posições.

do k = 0, nlat-1
   dx                 = sqrt(ppt_mz(k))
   yp(k)              = ppt_mz(k) + dx
```

```
26      xp(k)               = ppt_mz&lat(k)
27      xp(sigma*nlat-1-k) = ppt_mz&lat(k)
28      yp(sigma*nlat-1-k) = ppt_mz(k) - dx
29   end do
30
31   wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex09")
32
33   res                  = True
34   res@gsnDraw          = False                   ; Não desenha a figura.
35   res@gsnFrame         = False                   ; Gera apenas o frame.
36   res@xyLineThicknessF = 4.0                     ; Espessura da linha.
37   res@trYMinF          = -2.0                    ; Mínimo valor do eixo y.
38   res@trYMaxF          = 10.0                    ; Máximo valor do eixo y.
39   res@tiMainString     = "Prec zonal - Jan/1979"
40   res@tiXAxisString    = "Latitude"
41   res@tiYAxisString    = "Prec zonal (mm dia~S~-1 ~N~)"
42   res@tmYLLabelStride  = 1                ; Intervalo com que aparecem
43                                           ; os números no eixo y.
44   res@tmYLMode         = "Manual"         ; Formata o eixo y do meu jeito.
45   res@tmYLTickStartF   = res@trYMinF      ; Mínimo valor do eixo y.
46   res@tmYLTickSpacingF = 1                ; Incremento do eixo y.
47   res@tmYLTickEndF     = res@trYMaxF      ; Máximo valor do eixo y.
48   res@tmYLMinorOn      = False            ; Desabilita os traços
49                                           ; secundários do eixo y.
50
51   plot  = gsn_csm_xy (wks,ppt_mz&lat,ppt_mz,res)
52
53   ; Gera a curva preenchida.
54
55   gsres                = True
56   gsres@tfPolyDrawOrder = "Predraw"    ; Desenha primeiro o desvio padrão
57                                        ; e depois a linha.
58   gsres@gsFillColor     = "SlateBlue"  ; Cor selecionada.
59   gsres@gsFillOpacityF  = 0.4          ; Opacidade do preenchimento
60                                        ; 0 (menos opaco) a 1 (mais opaco).
61
62   dummy = gsn_add_polygon (wks,plot,xp,yp,gsres)
63
64   draw(plot)
65   frame(wks)
66
67   end
```

O resultado será:

21.3 Gráficos de barra

21.3.1 Gráfico barra simples

Para gerar gráficos de barra podem ser usadas algumas funções e opções do gráfico de linha. A diferença está em habilitar o recurso gsnXYBarChart para True.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/bar.shtml

Exemplo: Criação de um gráfico de barra.

```
1   ; Nome do script: cap21_ex10.ncl
2
3   begin
4
5   f = addfile ("../../dados/tair.2011.2012.nc","r")
6
7   u = short2flt(f->air)
8   u = u - 273.15
9
10  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex10")
11
12  res               = True
13  res@gsnXYBarChart = True ; Habilita gráfico de barras.
14
15  plot = gsn_csm_xy(wks,ispan(1,12,1),u(0:11,{1000},{-2},{300}),res)
16
17  end
```

O resultado será:

Esse gráfico necessita de formatação e o exemplo abaixo mostrará algumas possibilidades para realizar essa tarefa.

```
; Nome do script: cap21_ex11.ncl

begin

f = addfile ("../../dados/tair.2011.2012.nc","r")

u = short2flt(f->air)
u = u - 273.15

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex11")

res                        = True
res@gsnXYBarChart          = True     ; Habilita o gráfico de barras.
res@trYMaxF                = 28.      ; Máximo valor de y.
res@trYMinF                = 22.      ; Mínimo valor de y.
res@gsnXYBarChartColors    = "navy"   ; Cor da barra.
res@gsnXYBarChartBarWidth  = 0.60     ; Espaçamento entre as barras.
res@tmXBMode               = "Explicit"
res@tmXBValues             = (/1,2,3,4,5,6,7,8,9,10,11,12/)
res@tmXBLabels             = (/"J","F","M","A","M","J","J","A","S",\
                               "O","N","D"/)
res@tiMainString           = "Temperatura do ar em Lat: 2~S~o~N~S e " +\
                             "Lon: 300~S~o~"
res@tiXAxisString          = "M"+ecirc+"s"
res@tiYAxisString          = "Temperatura (~S~o~N~C)"

plot = gsn_csm_xy(wks,ispan(1,12,1),u(0:11,{1000},{-2},{300}),res)

end
```

O resultado será:

21.3.2 Gráfico com duas barras

```
; Nome do script: cap21_ex12.ncl

begin

; float precip(time, lat, lon)
f1 = addfile("../../dados/SA.GPCP.prec.1979.2013.nc","r")
; float precip(time, lat, lon)
f2 = addfile("../../dados/SA.CRU.prec.1979.2013.nc","r")

; f1 =  Dimensions and sizes: [time | 420] x [lat | 1] x [lon | 1]
; f2 =  Dimensions and sizes: [time | 420] x [lat | 1] x [lon | 1]

; O dado possui time,lat,lon apesar de ser um dado pontual, é necessário
; fixar a lat e a lon para que ocorra a redução de dimensão de 3 para
; apenas 1, no caso, a dimensão time. Por isso, é importante conhecer
; o dado antes de realizar qualquer tarefa.

;*********************************************************************
; Leitura da dimensão tempo para criar os rótulos do eixo x.
;*********************************************************************
time    = f1->time
TIME    = cd_calendar(time, 0) ; Tipo float.
year    = toint(TIME(:,0))     ; Guarda o ano.
month   = toint(TIME(:,1))     ; Guarda o mês.
day     = toint(TIME(:,2))     ; Guarda o dia.
```

```
ddd     = day_of_year(year,month,day) ; Retorna o dia do ano no formato
                                      ; 365 ou 366 dias.
YYYYMM  = year*100 + month
anoi    = 197901                ; Data inicial no formato AAAAMM.
anof    = 198012                ; Data final no formato AAAAMM.
datai   = ind(YYYYMM.eq.anoi)   ; Índice que será utilizado na dimensão
                                ; tempo para selecionar o período de
dataf   = ind(YYYYMM.eq.anof)   ; interesse.
nt      = (dataf-datai)+1       ; Número total de tempos.
dt      = 0.15                  ; Incremento para "desenhar" as barras.

x1 = fspan(datai-dt,dataf-dt,nt) ; Valores de x para a primeira curva.
x2 = fspan(datai+dt,dataf+dt,nt) ; Valores de x para a segunda curva.

y1 = f1->precip(datai:dataf,0,0) ; Tempo de interesse para y1.
y2 = f2->pre(datai:dataf,0,0)    ; Houve redução de dimensão de 3 para
                                 ; 1, pois o objetivo é gerar uma série
                                 ; temporal.

printVarSummary(y1)              ; Dimensions and sizes: [time | 12]
printVarSummary(y2)              ; Dimensions and sizes: [time | 12]

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex12")

; Recursos para os eixos x e y.
res                       = True
res@gsnXYBarChart         = True
res@tmXBLabelAngleF       = -90           ; Inclinação dos rótulos do
                                          ; eixo x.
res@tmXBLabelJust         = "CenterRight" ; Alinhamento dos rótulos do
                                          ; eixo x.
res@tmXBLabelStride       = 1             ; Ajusta quantos valores serão
                                          ; mostrados no eixo x.
res@trXMinF               = datai-1       ; Mínimo valor do eixo x.
res@trXMaxF               = dataf+1       ; Máximo valor do eixo x.
res@trYMaxF               = 7.0           ; Máximo valor do eixo y.
res@trYMinF               = 0.0           ; Mínimo valor do eixo y.
res@tmXBMode              = "Explicit"    ; Formata o eixo x do meu jeito
res@tmXBValues            = ispan(datai,dataf,2)
res@tmXBLabels            = YYYYMM(datai:dataf:2)
res@tiYAxisString         = "Precipita"+cedil+atilde+"o (mm/dia)"
res@tiMainString          = "Precipita"+cedil+atilde+"o CRU x GPCP"
res@gsnXYBarChartBarWidth = 0.22  ; Largura das barras.
res@gsnDraw               = True
```

```
70   res@gsnFrame                = False
71   res@tmYLMinorOn             = False ; Desliga os traços secundários do
72                                       ; eixo y.
73   res@tmYLLabelFontHeightF  = 0.02  ; tamanho do texto do eixo y.
74   res@tmXBLabelFontHeightF  = 0.02  ; tamanho do texto do eixo x.
75   res@vpXF                    = 0.125 ; Onde desenhar o gráfico em
76   res@vpYF                    = 0.75  ; coordenadas NDC.
77   res@vpHeightF               = 0.43  ; Altura da figura.
78   res@vpWidthF                = 0.7   ; Largura da figura.
79
80   res@gsnXYBarChartColors    = "blue" ; Cor da primeira barra.
81   plot1 = gsn_csm_xy(wks,x1,y1,res)  ; Plot da primeira barra.
82
83   res@gsnXYBarChartColors    = "red"  ; Cor da segunda barra.
84   plot2 = gsn_csm_xy(wks,x2,y2,res)  ; Plot da segunda barra.
85
86   ; Parte responsável pela formatação da legenda.
87   lbres                      = True
88   lbres@vpWidthF             = 0.25              ; Largura da legenda.
89   lbres@vpHeightF            = 0.08              ; Altura da legenda.
90   lbres@lbBoxMajorExtentF    = 0.8               ; Espaço entre as legendas.
91   lbres@lbFillColors         = (/"red","blue"/) ; Cores da legenda.
92   lbres@lbMonoFillPattern    = True              ; Habilita o preenchimento
93                                                 ; da legenda.
94   lbres@lbLabelFontHeightF = 0.02               ; Tamanho da fonte da legenda
95   lbres@lbLabelJust          = "CenterLeft"      ; Justificativa da legenda.
96   lbres@lbPerimOn            = False             ; Não desenha borda em torno
97                                                 ; legenda.
98   labels                     = (/"CRU","GPCP"/) ; Rótulos que vão aparecer
99                                                 ; na legenda.
100
101  ; drawNDCGrid(wks) ; Mostra as linhas de grade das coordenadas NDC.
102                     ; Útil para desenhar textos e acrescentar legendas.
103
104  ; Desenha a legenda na posição x (0.1), y (0.75) do gráfico.
105  ; 2 quer dizer quantos textos vão aparecer.
106  ; O labels é o texto que vai aparecer.
107
108  gsn_labelbar_ndc(wks,2,labels,0.1,0.75,lbres)
109
110  end
```

O resultado será:

21.3.3 Gráfico de barras com anomalia

```
; Nome do script: cap21_ex13.ncl

begin

f = addfile("../../dados/anom.SA.GPCP.prec.1979.2014.nc","r")

time    = f->time            ; time:units = "days since 1990-1-1 00:00:00"
TIME    = cd_calendar(time, 0); Tipo float.
year    = toint(TIME(:,0))    ; Guarda o ano.
month   = toint(TIME(:,1))    ; Guarda o mês.
day     = toint(TIME(:,2))    ; Guarda o dia.
ddd     = day_of_year(year,month,day) ; Retorna o dia do ano no formato
                                      ; 365 ou 366 dias.
YYYYMM  = year*100 + month
anoi    = 200001            ; Data inicial no formato AAAAMM. Altere aqui.
anof    = 200312             ; Data final no formato AAAAMM. Altere aqui.
datai   = ind(YYYYMM.eq.anoi); Índice que será utilizado na dimensão tempo
dataf   = ind(YYYYMM.eq.anof); para selecionar o período de interesse.
nt      = (dataf-datai)+1    ; Número total de tempos que será utilizado.
dt      = 0.15               ; Incremento para "desenhar" as barras.

x = fspan(datai-dt,dataf-dt,nt) ; Valores de x para a primeira curva.

y = f->precip(datai:dataf,0,0); Seleciona o tempo de interesse para y.
                             ; Houve redução de dimensão de 3 para 1,
```

```
                                ; pois o objetivo é gerar uma série temporal.

printVarSummary(y)              ; Dimensions and sizes: [time | 12]

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex13")

; Recursos para os eixos x e y.
res                        = True
res@gsnXYBarChart          = True          ; Habilita gráfico de barra.
res@tmXBLabelAngleF        = -90           ; Inclinação do eixo x.
res@tmXBLabelJust          = "CenterRight" ; Alinhamento do eixo x.
res@tmXBLabelStride        = 2             ; Ajusta quantos valores serão
                                           ; mostrados no eixo x.
res@trXMinF                = datai-1       ; Mínimo valor do eixo x.
res@trXMaxF                = dataf+1       ; Máximo valor do eixo x.
res@trYMaxF                = 1.0           ; Máximo valor do eixo y.
res@trYMinF                = -1.0          ; Mínimo valor do eixo y.
res@tmYLMode               = "Manual"      ; Formato o eixo y do meu jeito.
res@tmYLTickStartF         = res@trYMinF
res@tmYLTickSpacingF       = 0.5
res@tmYLTickEndF           = res@trYMaxF
res@tmXBMode               = "Explicit"    ; Formata o eixo x do meu jeito.
res@tmXBValues             = ispan(datai,dataf,1)
res@tmXBLabels             = YYYYMM(datai:dataf:1)
res@tiYAxisString          = "Anomalia de precipita"\
                             +cedil+atilde+"o (mm dia ~S~-1~N~)"
res@tiMainString           = "Anomalia mensal do GPCP"
res@gsnXYBarChartBarWidth = 0.5  ; Largura das barras.
res@tmYLMinorOn            = False; Desliga os traços secundários do y.
res@tmYLLabelFontHeightF   = 0.02 ; tamanho do texto do eixo y.
res@tmXBLabelFontHeightF   = 0.02 ; tamanho do texto do eixo x.
res@vpXF                   = 0.145; Onde desenhar o gráfico em
res@vpYF                   = 0.75 ; coordenadas NDC.
res@vpHeightF              = 0.43 ; Altura da figura.
res@vpWidthF               = 0.83 ; Largura da figura.
res@gsnYRefLine            = 0.   ; Linha de referência no valor zero.
res@gsnAboveYRefLineColor = "red"; Valor positivo preenche com vermelho.
res@gsnBelowYRefLineColor = "blue"; Valor negativo preenche com azul.

plot = gsn_csm_xy(wks,x,y,res)

end
```

O resultado será:

Vários exemplos de gráficos de barras podem ser encontrados no link abaixo. Navegue pela página e escolha a melhor opção.

http://www.ncl.ucar.edu/Applications/bar.shtml

21.4 Gráficos espaciais

21.4.1 Campos escalares

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/cylineq.shtml

Exemplo:

```
; Nome do script: cap21_ex14.ncl

begin

f = addfile ("../../dados/tair.2011.2012.nc","r")

t = f->air ; O dado é do tipo short. Visto com ncl_filedump.

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex14")

; Plota o primeiro tempo (0), primeiro nível vertical (1000hPa),
; todas as latitudes (:) e todas longitudes (:), uma vez que, o
; dado possui 4 dimensões.

plot = gsn_csm_contour_map_ce(wks,t(0,{1000},:,:),False)

end
```

O resultado será:

Os valores de temperatura (K) estão diferentes, isso acontece porque a variável air está compactada (linha 7), para descompactar, utiliza-se a função short2flt. O script abaixo realiza esse tarefa.

```
1   ; Nome do script: cap21_ex15.ncl
2
3   begin
4
5   f = addfile ("../../dados/tair.2011.2012.nc","r")
6
7   t = short2flt(f->air) ; Descompactando o dado com a função
8                         ; short2flt.
9
10  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex15")
11
12  ; Plota o primeiro tempo (0), primeiro nível vertical (1000hPa),
13  ; todas as latitudes (:) e todas as longitudes (:)
14
15  plot = gsn_csm_contour_map_ce(wks,t(0,{1000},:,:),False)
16
17  end
```

O resultado será:

Os valores estão na ordem de grandeza da temperatura (K).

No script abaixo serão incluídos os títulos como também serão alterados os intervalos dos contornos.

```
; Nome do script: cap21_ex16.ncl

begin

f = addfile ("../../dados/tair.2011.2012.nc","r")

t = short2flt(f->air)
t = t - 273.15

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex16")

res                       = True
res@tiMainString          = "Temperatura - 1000hPa" ; Título principal
                                                    ; da figura.
res@gsnLeftString         = "Fonte: NCEP/R2" ; Título do lado esquerdo.
res@gsnCenterString       = "Janeiro"        ; Título central.
res@gsnRightString        = "~S~o~N~C"       ; Título do lado direito.
res@cnLevelSelectionMode = "ManualLevels"   ; Fixa os valores manualmente.
res@cnMinLevelValF        = -40.0            ; Fixa o mínimo valor do
                                             ; contorno.
res@cnMaxLevelValF        = 40.0             ; Fixa o máximo valor do
                                             ; contorno.
res@cnLevelSpacingF       = 10.0             ; Fixa o espaçamento do
                                             ; contorno.
res@cnInfoLabelOn         = False            ; Desabilita as informações
                                             ; do contorno.

; Plota o primeiro tempo (0), primeiro nível vertical (1000hPa),
; todas as latitudes (:) e todas as longitudes (:)

plot = gsn_csm_contour_map_ce(wks,t(0,{1000},:,:),res)

end
```

O resultado será:

Além do gráfico de contorno, às vezes, deseja-se visualizar o gráfico de forma preen-

chida ou shaded. O script abaixo é um exemplo de como realizar essa tarefa.

```
; Nome do script: cap21_ex17.ncl

begin

f = addfile ("../../dados/tair.2011.2012.nc","r")

t = short2flt(f->air)
t = t - 273.15

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex17")

res                 = True
res@tiMainString    = "Temperatura em 1000hPa"
res@gsnLeftString   = "Fonte: NCEP/R2"
res@gsnCenterString = "Janeiro"
res@gsnRightString  = "~S~o~N~C"
res@cnLinesOn       = False          ; Desabilita o gráfico de contorno.
res@cnFillOn        = True           ; Gráfico com preenchimento.
res@lbOrientation   = "Horizontal"   ; Orientação da legenda (Horizontal
                                     ; ou Vertical).

; Plota o primeiro tempo (0), primeiro nível vertical (1000 hPa),
; todas as latitudes (:) e longitudes (:)

plot = gsn_csm_contour_map_ce(wks,t(0,{1000},:,:),res)

end
```

O resultado será:

21.4.2 Campos vetoriais

Outro exemplo é a geração de mapas de vetor do vento. O exemplo abaixo mostra uma figura sobre a América do Sul no nível de 200hPa para o mês de dezembro.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/vector.shtml

```
; Nome do script: cap21_ex18.ncl

begin

a = addfile("../../dados/u.2011.2012.nc","r")
b = addfile("../../dados/v.2011.2012.nc","r")

u = short2flt(a->uwnd)
v = short2flt(b->vwnd)

vel = sqrt(u^2+v^2)

copy_VarCoords(u,vel)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex18")

res                          = True
res@tiMainString             = "Vetor velocidade do vento em 200hPa"
res@gsnLeftString            = "NCEP/R2"
res@gsnCenterString          = "Dezembro"
res@gsnRightString           = "m s~S~-1"
res@vpXF                     = 0.1  ; Posição (vpX,vpY) onde será
res@vpYF                     = 0.85 ; desenhada figura.
res@gsnMajorLonSpacing       = 10      ; Variação da escala do eixo x.
res@gsnMajorLatSpacing       = 10      ; Variação da escala do eixo y.
res@vcRefMagnitudeF          = 15.0    ; Magnitude de vetor.
res@vcRefLengthF             = 0.050   ; Comprimento do vetor.
res@vcGlyphStyle             = "CurlyVector"; Efeito de curvatura nos
                                            ; vetores.
res@vcMinDistanceF           = 0.01    ; Espessura dos vetores.
res@vcRefAnnoOrthogonalPosF  = -1.0    ; Deslocamento da legenda da
                                       ; velocidade.
res@mpMinLonF                = -90.0   ; Longitude oeste.
res@mpMaxLonF                = -30.0   ; Longitude leste.
res@mpMinLatF                = -60.0   ; Latitude sul.
res@mpMaxLatF                =  20.0      ; Latitude norte.
res@mpOutlineBoundarySets    = "National" ; Mostra a divisão dos países.
res@mpDataSetName            = "Earth..4"
res@mpDataBaseVersion        = "MediumRes"
res@mpOutlineOn              = True
res@mpOutlineSpecifiers      = (/"Brazil:states"/) ; Divisão dos estados
                                                   ; brasileiros.
res@mpFillOn                 = False       ; Mapa com fundo branco.
res@lbOrientation            = "Vertical"  ; Orientação da legenda.

plot=gsn_csm_vector_scalar_map_ce(wks,u(11,{200},:,:),v(11,{200},:,:),\
                                  vel(11,{200},:,:), res)

end
```

O resultado será:

Abrindo um arquivo global e plotando os vetores.

```
; Nome do script: cap21_ex19.ncl

begin

; short uwnd ( time, level, lat, lon )
; short vwnd ( time, level, lat, lon )
f = addfile("../../dados/uwnd.vwnd.nc","r")

; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
u = short2flt(f->uwnd)
v = short2flt(f->vwnd)

wks  = gsn_open_wks("pdf","ex01_vetor_p2")

; Como o dado é 4D (time,level,lat,lon) é necessário que ele seja do
; tipo 2D (lat,lon) para gerar a figura. Por isso, a redução de dimensão
; de 4 para 2, isto é, lat e lon mediante a fixação do tempo (0) e
; nível vertical em 1000hPa (detalhe para o uso do símbolo "{}").

; O gráfico será gerado sem nenhuma formação, por isso, o False.

plot = gsn_csm_vector_map_ce(wks,u(0,{1000},:,:),v(0,{1000},:,:),False)

end
```

O resultado será:

Aplicando zoom e personalizando o gráfico:

```
; Nome do script: cap21_ex20.ncl

begin

; short uwnd ( time, level, lat, lon )
; short vwnd ( time, level, lat, lon )
f = addfile("../../dados/uwnd.vwnd.nc","r")

; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
u = short2flt(f->uwnd)
; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
v = short2flt(f->vwnd)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex20")

res                          = True
res@vcRefMagnitudeF          = 4.0        ; Define o vetor de referência.
res@vcRefLengthF             = 0.045      ; Define o tamanho do vetor de
                                          ; referência.
res@vcRefAnnoOrthogonalPosF  = -1.0       ; Deslocamento o vetor de
                                          ; referência.
res@vcGlyphStyle             = "CurlyVector" ; Aplica efeito de curvatura
                                             ; aos vetores.
res@vcLineArrowColor         = "blue"        ; Muda a cor do vetor de
                                             ; referência.
res@vcLineArrowThicknessF    = 2.0           ; Espessura dos vetores.
res@mpMinLonF                = -110.0     ; Aplica
res@mpMaxLonF                = -20.0      ; um zoom
res@mpMinLatF                = -60.0      ; na áres de
res@mpMaxLatF                =  20.0      ; interesse.
res@gsnLeftString            = "NCEP"     ; Título do lado esquerdo. Caso
                                          ; não queira plotar nada,
                                          ; basta colocar "".
res@gsnCenterString          = "Vetor do vento" ; Título central.
res@gsnRightString           = "m/s"      ; Título do lado direito.
res@gsnMajorLonSpacing       = 10         ; Espaçamento do eixo x.
```

```
res@gsnMajorLatSpacing        = 10        ; Espaçamento do eixo y.
res@tmXBMinorOn               = False ; Desabilita os traços secundários
res@tmYLMinorOn               = False ; dos eixos x e y.

; Como o dado é 4D (time,level,lat,lon) é necessário que ele seja do
; tipo 2D (lat,lon) para gerar a figura. Por isso, a redução de dimensão
; de 4 para 2, isto é, lat e lon mediante a fixação do tempo (0) e
; nível vertical em 1000hPa (detalhe para o uso do símbolo "{}").

plot = gsn_csm_vector_map_ce(wks,u(0,{1000},:,:),v(0,{1000},:,:),res)

end
```

O resultado será:

Gerando os vetores e inserindo uma informação de uma variável escalar nos vetores:

```
; Nome do script: cap21_ex21.ncl

begin

; short uwnd ( time, level, lat, lon )
; short vwnd ( time, level, lat, lon )
f = addfile("../../dados/uwnd.vwnd.nc","r")

; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
u = short2flt(f->uwnd)
v = short2flt(f->vwnd)
vel = wind_speed(u,v) ; Calcula a velocidade do vento.

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex21")

res                           = True
res@vcRefMagnitudeF           = 4.0       ; Define o vetor de referência.
```

```
18  res@vcRefLengthF            = 0.045     ; Define o tamanho do vetor
19                                          ; de referência.
20  res@vcRefAnnoOrthogonalPosF = -1.0      ; Deslocamento o vetor de
21                                          ; referência.
22  res@vcGlyphStyle            = "CurlyVector" ; Aplica efeito de curvatura
23                                              ; aos vetores.
24  res@vcLineArrowThicknessF   = 2.0           ; Espessura dos vetores.
25  res@mpMinLonF               = -110.0    ; Aplica
26  res@mpMaxLonF               = -20.0     ; um zoom
27  res@mpMinLatF               = -60.0     ; na área de
28  res@mpMaxLatF               =  20.0     ; interesse.
29  res@gsnLeftString           = "NCEP"    ; Título do lado esquerdo.
30                                          ; Caso não queira plotar nada,
31                                          ; basta colocar "".
32  res@gsnCenterString         = "Vetor do vento" ; Título central.
33  res@gsnRightString          = "m/s"     ; Título do lado direito.
34  res@gsnMajorLonSpacing      = 10        ; Espaçamento do eixo x da
35                                          ; longitude.
36  res@gsnMajorLatSpacing      = 10        ; Espaçamento do eixo y da
37                                          ; latitude.
38  res@tmXBMinorOn             = False     ; Desabilita os traços
39  res@tmYLMinorOn             = False     ; secundários dos eixos x e y.
40  res@mpFillOn                = False       ; Mapa sem preenchimento.
41  res@mpOutlineBoundarySets   = "National"  ; Mostra divisão dos países.
42  res@mpDataSetName           = "Earth..4"  ; Para a mostrar a divisão
43  res@mpDataBaseVersion       = "MediumRes" ; dos estados brasileiros, são
44  res@mpOutlineSpecifiers     = (/"Brazil:states"/) ; necessárias estas
45                                                    ; três linhas.
46  res@vcMinDistanceF          = 0.01        ; Densidade de vetores.
47
48  ; Como o dado é 4D (time,level,lat,lon) é necessário que ele seja do
49  ; tipo 2D (lat,lon) para gerar a figura. Por isso, a redução de dimensão
50  ; de 4 para 2, isto é, lat e lon mediante a fixação do tempo (0) e
51  ; nível vertical em 1000hPa (detalhe para o uso do símbolo "{}").
52
53  plot = gsn_csm_vector_scalar_map_ce(wks,u(0,{1000},:,:),\
54                                      v(0,{1000},:,:),\
55                                    vel(0,{1000},:,:),res)
56
57  end
```

O resultado será:

21.4.3 Campos polares

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/polar.shtml

```
; Nome do script: cap21_ex22.ncl

begin

f = addfile("../../dados/hgt.jan.mar.1979.nc","r")

; short hgt ( time, level, lat, lon ) = 3, 1, 144, 73
hgt = short2flt(f->hgt)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex22")

res                 = True
res@gsnPolar        = "SH"  ; Hemisfério a ser visualizado.
res@cnFillOn        = True  ; Gráfico do tipo shaded.
res@mpFillOn        = False ; Mapa com fundo branco.
res@cnLinesOn       = False ; Desabilita as linhas de contorno.
res@cnFillMode      = "RasterFill" ; Não interpola o dado (grfill do
                                   ; GrADS).
res@tiMainString    = "Altura geopotencial em 500hPa" + \
                      " - Fevereiro/1979"
res@gsnCenterString = ""
res@gsnLeftString   = ""
res@gsnRightString  = ""

plot = gsn_csm_contour_map_polar(wks,hgt(1,0,:,:),res)

end
```

O resultado será:

Defindo as cores das linhas (valores positivos e negativos).

```
; Nome do script: cap21_ex23.ncl

begin

f = addfile("../../dados/anom.hgt.jan1979.dez2000.nc","r")

; short hgt ( time, level, lat, lon ) = 3, 1, 144, 73
hgt = short2flt(f->hgt)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex23")

res                                 = True
res@gsnPolar                        = "NH"  ; Hemisfério a ser visualizado.
res@mpFillOn                        = False ; Mapa com fundo branco.
res@tiMainString                    = "Anomalia de altura geopotencial - " +\
                                      "Fev/1979"
res@gsnCenterString                 = ""
res@gsnLeftString                   = ""
res@gsnRightString                  = ""
res@mpGridLatSpacingF               = 10.0  ; Espaçamento da latitude.
res@mpGridLonSpacingF               = 30.0  ; Espaçamento da longitude.
res@gsnDraw                         = False
```

```
res@gsnFrame                      = False
res@gsnContourZeroLineThicknessF = 4.0 ; Espessura do contorno zero.
                                       ; Para eliminar o contorno zero,
                                       ; basta colocar o valor zero.
res@mpCenterLonF                  = -180  ; Ponto de vista do mapa.
res@gsnPolarLabelDistance         = 1.08  ; Distância das longitudes.
res@gsnPolarLabelFontHeightF      = 0.020 ; Altera o tamanho da fonte
                                          ; dos rótulos de longitude.
res@cnLineLabelPlacementMode      = "constant" ; Rótulo dos contornos.
res@cnInfoLabelOn                 = False ; Desabilita as informações
                                          ; de contorno.

plot = gsn_csm_contour_map_polar(wks,hgt(1,0,:,:),res)

; Define cores para os valores negativos (blue), zero (black)
; e positivos (red).
plot = ColorNegDashZeroPosContour(plot,"blue","black","red")

draw(plot)
frame(wks)

end
```

O resultado será:

Plotando valores positivos e negativos e padronizando a escala.

```
; Nome do script: cap21_ex24.ncl

begin

f = addfile("../../dados/anom.hgt.jan1979.dez2000.nc","r")

; short hgt ( time, level, lat, lon ) = 3, 1, 144, 73
hgt = short2flt(f->hgt)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex24")

gsn_define_colormap(wks,"BlueDarkRed18") ; Defini a tabela de cores.

res                                = True
res@gsnPolar                       = "NH"  ; Hemisfério a ser visualizado.
res@mpFillOn                       = False ; Mapa com fundo branco.
res@tiMainString                   = "Anomalia de altura geopotencial - " +\
                                     "Fev/1979"
res@gsnCenterString                = ""
res@gsnLeftString                  = ""
res@gsnRightString                 = ""
res@mpGridLatSpacingF              = 30.0  ; Espaçamento da latitude.
res@mpGridLonSpacingF              = 90.0  ; Espaçamento da longitude.
res@gsnContourZeroLineThicknessF = 4.0   ; Espessura do contorno zero.
                                         ; Para eliminar o contorno zero
                                         ; basta colocar o valor zero.
res@mpCenterLonF                   = -90   ; Ponto de vista do mapa.
res@gsnPolarLabelDistance          = 1.08  ; Distância das longitudes.
res@gsnPolarLabelFontHeightF       = 0.020 ; Altera o tamanho da fonte dos
                                           ; rótulos de longitude.
res@cnFillOn                       = True  ; Gráfico preenchido (shaded).
res@cnLinesOn                      = False ; Desabilita as linhas de
                                           ; contorno.
res@cnFillOpacityF                 = 0.7   ; Opacidade do preenchimento.
                                           ; 0 (mais opaco) e
                                           ; 1 (menos opaco).
res@cnLevelSelectionMode           = "ExplicitLevels"
res@cnLevels                       = ispan(-200,200,50)
res@cnFillColors                   = (/3,4,5,6,0,0,14,15,16,17/)

plot = gsn_csm_contour_map_polar(wks,hgt(1,0,:,:),res)

end
```

O resultado será:

Plotando campo vetorial:

```
; Nome do script: cap21_ex25.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")

; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
u   = short2flt(f->uwnd)
v   = short2flt(f->vwnd)
vel = wind_speed(u,v)    ; Calcula a velocidade do vento.

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex25")

res                          = True
res@gsnPolar                 = "NH"  ; Hemisfério a ser visualizado.
res@mpFillOn                 = False ; Mapa com fundo branco.
res@tiMainString             = "Vetor do vento e velocidade - Fev/1979"
res@gsnCenterString          = ""
res@gsnLeftString            = ""
res@gsnRightString           = ""
res@mpGridLatSpacingF        = 30.0  ; Espaçamento da latitude.
res@mpGridLonSpacingF        = 90.0  ; Espaçamento da longitude.
```

```
res@mpCenterLonF               = -90    ; Ponto de vista do mapa.
res@gsnPolarLabelDistance      = 1.08   ; Distância das longitudes.
res@gsnPolarLabelFontHeightF   = 0.020  ; Altera o tamanho da fonte
                                        ; dos rótulos de longitude.
res@vcRefMagnitudeF            = 10.0   ; Define o vetor de referência.
res@vcRefLengthF               = 0.050  ; Tamanho do vetor de referência.
res@vcMinDistanceF             = 0.02   ; Densidade de vetores.
res@vcGlyphStyle               = "CurlyVector" ; Efeito de curvatura.
res@vcRefAnnoArrowLineColor    = "black" ; Cor do vetor de referência.
res@vcRefAnnoArrowUseVecColor  = False

plot = gsn_csm_vector_scalar_map_polar(wks,u(1,0,:,:),v(1,0,:,:),\
                                       vel(1,0,:,:),res)

end
```

O resultado será:

Plotando campo vetorial e escalar no mesmo gráfico:

```
; Nome do script: cap21_ex26.ncl

begin

; short uwnd ( time, level, lat, lon )
f = addfile("../../dados/uwnd.vwnd.nc","r")

; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
u   = short2flt(f->uwnd)
v   = short2flt(f->vwnd)
vel = wind_speed(u,v)    ; Calcula a velocidade do vento.

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex26")

res                          = True
res@gsnPolar                 = "NH"  ; Hemisfério a ser visualizado.
res@gsnScalarContour         = True  ; Vetores sobre um campo escalar.
res@cnFillOn                 = True  ; Habilita gráfico shaded.
res@cnLevelSpacingF          = 2.0   ; Intervalo do espaçamento.
res@cnLinesOn                = False ; Sem linhas de contorno.
res@mpFillOn                 = False ; Mapa com fundo branco.
res@tiMainString             = "Vetor do vento e velocidade - Fev/1979"
res@gsnCenterString          = ""
res@gsnLeftString            = ""
res@gsnRightString           = ""
res@mpGridLatSpacingF        = 30.0  ; Espaçamento da latitude.
res@mpGridLonSpacingF        = 90.0  ; Espaçamento da longitude.
res@mpCenterLonF             = -90   ; Ponto de vista do mapa.
res@gsnPolarLabelDistance    = 1.06  ; Distância das longitudes.
res@gsnPolarLabelFontHeightF = 0.020 ; Altera o tamanho da fonte
                                     ; dos rótulos de longitude.
res@vcRefMagnitudeF          = 10.0  ; Define o vetor de referência.
res@vcRefLengthF             = 0.050 ; Define o tamanho do vetor
                                     ; de referência.
res@vcMinDistanceF           = 0.02  ; Densidade de vetores.
res@vcGlyphStyle             = "CurlyVector" ; Aplica efeito de
                                             ; curvatura.
res@vcRefAnnoArrowLineColor  = "black"       ; Cor do vetor de
                                             ; referência.
res@vcRefAnnoArrowUseVecColor= False
res@mpGeophysicalLineThicknessF = 4.0   ; Espessura da linha dos países.
res@mpGeophysicalLineColor      = "red" ; Cor da linha dos países.

plot = gsn_csm_vector_scalar_map_polar(wks,u(1,0,:,:),v(1,0,:,:),\
                                       vel(1,0,:,:),res)

end
```

O resultado será:

21.4.4 Campos de linhas de corrente

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/stream.shtml

http://www.ncl.ucar.edu/Applications/generic_stream.shtml

Exemplo:

```
1   ; Nome do script: cap21_ex27.ncl
2
3   begin
4
5   f = addfile("../../dados/uwnd.vwnd.nc","r")
6
7    ; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
8   u = short2flt(f->uwnd)
9   v = short2flt(f->vwnd)
10
11  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex27")
12
13  ; Como o dado é 4D (time,level,lat,lon) é necessário que ele seja do
14  ; tipo 2D (lat,lon) para gerar a figura. Por isso, a redução de
15  ; dimensão de 4 para 2, isto é, lat e lon mediante a fixação do
16  ; tempo (0) e nível vertical em 850hPa (detalhe para o uso do símbolo
17  ; "{}").
```

```
18
19  ; O gráfico será gerado sem nenhuma formatação, por isso, o False.
20
21  plot = gsn_csm_streamline_map_ce(wks,u(0,{850},:,:), \
22                                   v(0,{850},:,:),False)
23
24  end
```

O resultado será:

Aplicando zoom e personalizando o gráfico:

```
1   ; Nome do script: cap21_ex28.ncl
2
3   begin
4
5   f = addfile("../../dados/uwnd.vwnd.nc","r")
6
7    ; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
8   u = short2flt(f->uwnd)
9   v = short2flt(f->vwnd)
10
11  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex28")
12
13  res                     = True
14  res@stArrowLengthF      = 0.008     ; Tamanho das flechas dos vetores.
15  res@stMinArrowSpacingF  = 0.001     ; Espaçamento dos vetores
16                                      ; da mesma linha.
17  res@stArrowStride       = 1         ; Número de vetores em cada linha,
18                                      ; quanto menor o valor, mais vetores.
19  res@stLineThicknessF    = 1.5       ; Espessura da linha.
20  res@stLineColor         = "red"     ; Cor da linha.
21  res@mpMinLonF           = -110.0    ; Aplica
22  res@mpMaxLonF           = -20.0     ; um zoom
23  res@mpMinLatF           = -60.0     ; na área de
24  res@mpMaxLatF           =  20.0     ; interesse.
25  res@gsnLeftString       = "NCEP"    ; Título do lado esquerdo.
26                                      ; Caso não queira plotar nada,
27                                      ; basta colocar "".
```

```
res@gsnCenterString     = "Linhas de corrente" ; Título central.
res@gsnRightString      = "m/s"      ; Título do lado direito.
res@gsnMajorLonSpacing = 10         ; Espaçamento do eixo x da longitude.
res@gsnMajorLatSpacing = 10         ; Espaçamento do eixo y da latitude.
res@tmXBMinorOn         = False      ; Desabilita os traços secundários
res@tmYLMinorOn         = False      ; dos eixos x e y.

; Como o dado é 4D (time,level,lat,lon) é necessário que ele seja do
; tipo 2D (lat,lon) para gerar a figura. Por isso, a redução de dimensão
; de 4 para 2, isto é, lat e lon mediante a fixação do tempo (0) e
; nível vertical em 850hPa (detalhe para o uso do símbolo "{}").

plot = gsn_csm_streamline_map_ce(wks,u(0,{850},:,:),v(0,{850},:,:),res)

end
```

O resultado será:

Delimitando os estados brasileiros e personalizando o gráfico.

```
; Nome do script: cap21_ex29.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")

 ; [time | 3] x [level | 12] x [lat | 73] x [lon | 144]
u   = short2flt(f->uwnd)
v   = short2flt(f->vwnd)
vel = sqrt(u^2+v^2)

copy_VarCoords(u,vel)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex29")

res                       = True
res@stArrowLengthF        = 0.008  ; Tamanho das flechas dos vetores.
res@stMinArrowSpacingF    = 0.001  ; Espaçamento dos vetores da mesma
                                   ; linha.
res@stArrowStride         = 1      ; Número de vetores em cada linha,
                                   ; quanto menor o valor, mais vetores.
res@stLineThicknessF      = 1.5    ; Espessura da linha.
res@stLineColor           = "red"  ; Cor da linha.
res@mpMinLonF             = -110.0 ; Aplica
res@mpMaxLonF             = -10.0  ; um zoom
res@mpMinLatF             = -60.0  ; na área de
res@mpMaxLatF             =  20.0  ; interesse.
res@gsnLeftString         = ""     ; Título do lado esquerdo. Caso não
                                   ; queira plotar nada, basta colocar "".
res@gsnCenterString       = "Linhas de corrente e velocidade (colorido)"
res@gsnRightString        = ""     ; Título do lado direito.
res@gsnMajorLonSpacing    = 10     ; Espaçamento do eixo x da longitude.
res@gsnMajorLatSpacing    = 10     ; Espaçamento do eixo y da latitude.
res@tmXBMinorOn           = False  ; Desabilita os traços secundários dos
res@tmYLMinorOn           = False  ; eixos x e y.
res@mpFillOn              = False ; Mapa sem preenchimento (fundo branco).
res@mpOutlineBoundarySets = "National" ; Mostra divisão dos países.
res@mpDataSetName         = "Earth..4" ; Para a mostrar a divisão dos
res@mpDataBaseVersion     = "MediumRes"; estados brasileiros, são
res@mpOutlineSpecifiers   = (/"Brazil:states"/) ; necessárias estas três
                                        ; linhas.
res@stLevelSpacingF       = 10.0        ; Espaçamento entre as linhas das
                                        ;linhas de corrente.

; Como o dado é 4D (time,level,lat,lon) é necessário que ele seja do
; tipo 2D (lat,lon) para gerar a figura. Por isso, a redução de dimensão
; de 4 para 2, isto é, lat e lon mediante a fixação do tempo (0) e
; nível vertical em 850hPa (detalhe para o uso do símbolo "{}").

nivel = 200 ; Nível vertical de interesse.

plot = gsn_csm_streamline_scalar_map(wks,u(0,{nivel},:,:),\
                                     v(0,{nivel},:,:),\
                                     vel(0,{nivel},:,:),res)

end
```

O resultado será:

21.5 Seção vertical

21.5.1 Pressão/altura versus latitude

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/height_lat.shtml

Exemplo:

```
; Nome do script: cap21_ex30.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")

; short uwnd ( time, level, lat, lon ) = 3, 12, 73, 144
u = short2flt(f->uwnd)

uz = zonalAve(u) ; Realiza a média zonal (longitude).

printVarSummary(uz)
; Dimensions and sizes: [time | 3] x [level | 12] x [lat | 73]

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex30")

res                                = True
res@tiMainString                   = "M"+eacute+"dia zonal"
res@cnLevelSpacingF                = 5.0 ; Espaçamento entre
                                         ; as isolinhas.
res@gsnContourZeroLineThicknessF = 5.   ; Espessura da linha
                                         ; do contorno zero.
res@gsnContourNegLineDashPattern = 1    ; Fixa as isolinhas negativas
                                         ; com o padrão 1
                                         ; (linhas tracejadas).

plot = gsn_csm_pres_hgt(wks,uz(2,:,:),res)

end
```

O resultado será:

Definindo padrão de preenchimento:

```
; Nome do script: cap21_ex31.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")

; short uwnd ( time, level, lat, lon ) = 3, 12, 73, 144
u = short2flt(f->uwnd)

uz = zonalAve(u) ; Realiza a média zonal (longitude).

printVarSummary(uz)
; Dimensions and sizes: [time | 3] x [level | 12] x [lat | 73]

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex31")

res                                = True
res@tiMainString                   = "M"+eacute+"dia zonal"
res@cnLevelSpacingF                = 5.0    ; Espaçamento entre
                                            ; as isolinhas.
res@gsnContourZeroLineThicknessF = 5.0    ; Espessura da linha do
                                            ; contorno zero.
res@gsnContourNegLineDashPattern = 1      ; Fixa as isolinhas negativas
                                            ; com o padrão 1
                                            ; (linhas tracejadas).
res@gsnDraw                        = False ; Não cria a figura.
res@gsnFrame                       = False ; Não avança o frame.

plot = gsn_csm_pres_hgt(wks,uz(0,:,:),res)

; Personalização do padrão de preenchimento.
```

```
32
33   opt                   = True
34   opt@gsnShadeFillType = "pattern"  ; Pode ser "color" ou "pattern".
35   opt@gsnShadeLow      = 17         ; Tipo de preechimento.
36
37   plot = gsn_contour_shade(plot,0,0,opt) ; Preenche os valores menores
38                                          ; que zero.
39
40   draw(plot)
41   frame (wks)
42
43   end
```

O resultado será:

Personalizando o padrão de preenchimento.

```
1   ; Nome do script: cap21_ex32.ncl
2
3   begin
4
5   f = addfile("../../dados/uwnd.vwnd.nc","r")
6
7   ; short uwnd ( time, level, lat, lon ) = 3, 12, 73, 144
8   u = short2flt(f->uwnd)
9
10  uz = zonalAve(u) ; Realiza a média zonal (longitude).
11
12  printVarSummary(uz)
13   ; Dimensions and sizes: [time | 3] x [level | 12] x [lat | 73]
14
15  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex32")
16
17  res                                = True
18  res@tiMainString                   = "M"+eacute+"dia zonal"
19  res@cnLevelSpacingF                = 5.0   ; Espaçamento entre
20                                             ; as isolinhas.
21  res@gsnContourZeroLineThicknessF = 5.0   ; Espessura da linha
22                                             ; do contorno zero.
23  res@gsnContourNegLineDashPattern = 1     ; Fixa as isolinhas
24                                             ; Negativas com o padrão 1
25                                             ; (linhas tracejadas).
26  res@gsnDraw                        = False ; Não cria a figura.
27  res@gsnFrame                       = False ; Não avança o frame.
28
29  plot = gsn_csm_pres_hgt(wks,uz(0,:,:),res)
30
31  ; Personalização do padrão de preenchimento.
32
33  opt                  = True
34  opt@gsnShadeFillType = "pattern" ; Pode ser "color" ou "pattern".
35  opt@gsnShadeLow      = 17        ; Tipo de preechimento para
36                                   ; os menores valores.
37  opt@gsnShadeHigh     = 3         ; Tipo de preechimento para
38                                   ; os maiores valores.
39
40  ; Preenche os valores menores que zero com o padrão
41  ; 17 e os valores maiores que 20 com padrão 3.
42
43  plot = gsn_contour_shade(plot,0,20,opt)
44
45  draw(plot)
46  frame (wks)
47
48  end
```

O resultado será:

Gráfico preenchido:

```
; Nome do script: cap21_ex33.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")

; short uwnd ( time, level, lat, lon ) = 3, 12, 73, 144
u = short2flt(f->uwnd)

uz = zonalAve(u) ; Realiza a média zonal (longitude).

; Dimensions and sizes: [time | 3] x [level | 12] x [lat | 73]
printVarSummary(uz)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex33")

res                              = True
res@tiMainString                 = "M"+eacute+"dia zonal"
res@gsnContourZeroLineThicknessF = 5.0 ; Espessura da linha do
                                       ; contorno zero.
res@gsnContourNegLineDashPattern = 1   ; Fixa as isolinhas negativas
                                       ; com o padrão 1
                                       ; (linhas tracejadas).
res@cnLevelSelectionMode         = "ManualLevels" ; Fixa os níveis
                                                  ; de forma manual.
res@cnLevelSpacingF              = 5.0      ; Espaçamento dos valores.
res@cnMinLevelValF               = -50.     ; Mínimo valor.
res@cnMaxLevelValF               =  50.     ; Máximo valor.
res@cnLineLabelsOn               = True     ; Habilita as linhas
                                            ; de contorno.
res@cnFillOn                     = True     ; Habilita o preechimento
                                            ; do gráfico.
res@cnFillPalette                = "BlWhRe" ; Escolha da tabela de cores.
```

```
res@tiYAxisString                    = "Press"+atilde+"o (hPa)"
res@cnLineLabelPlacementMode         = "constant" ; Defini o tipo de
                                                  ; rótulo dos contornos.
res@gsnLeftString                    = ""
res@gsnCenterString                  = ""
res@gsnRightString                   = ""
res@tmYRMode                         = "Automatic"; Desabilita o eixo
                                                  ; y da direita (altura).

plot = gsn_csm_pres_hgt(wks,uz(0,:,:),res)

end
```

O resultado será:

21.5.2 Pressão/altura versus longitude

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/height_long.shtml

Exemplo:

```
; Nome do script: cap21_ex34.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")

; short uwnd ( time, level, lat, lon ) = 3, 12, 73, 144
u = short2flt(f->uwnd)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex34")

; Plot da figura pressão (y) versus longitude (x).
; Latitude escolhida "0". Com o símbolo "{}" pode-se
; utilizar a coordenada de latitude.

plot = gsn_csm_pres_hgt(wks,u(0,:,{0},:),False)

end
```

O resultado será:

Definindo o estilo de linha:

```
; Nome do script: cap21_ex35.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")

; short uwnd ( time, level, lat, lon ) = 3, 12, 73, 144
u = short2flt(f->uwnd)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex35")

res                                = True
res@tiMainString                   = "Vento zonal"
res@cnLevelSpacingF                = 5.0 ; Espaçamento entre
                                         ; as isolinhas.
res@gsnContourZeroLineThicknessF = 5.   ; Espessura da linha
                                         ; do contorno zero.
res@gsnContourNegLineDashPattern = 1    ; Fixa as isolinhas
                                         ; negativas com o padrão 1
                                         ; (linhas tracejadas).

plot = gsn_csm_pres_hgt(wks,u(0,:,{0},:),res)

end
```

O resultado será:

Definindo o tipo de preenchimento.

```
; Nome do script: cap21_ex36.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")

u = short2flt(f->uwnd)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex36")

res                                = True
res@tiMainString                   = "Vento zonal"
res@cnLevelSpacingF                = 5.0 ; Espaçamento entre as isolinhas.
res@gsnContourZeroLineThicknessF = 5.0 ; Espessura da linha do
                                       ; contorno zero.
res@gsnContourNegLineDashPattern = 1   ; Fixa as isolinhas
                                       ; negativas com o padrão 1
                                       ; (linhas tracejadas).
res@gsnDraw                        = False ; Não cria a figura.
res@gsnFrame                       = False ; Não avança o frame.

plot = gsn_csm_pres_hgt(wks,u(0,:,{0},:),res)

; Personalização do padrão de preenchimento.

opt                  = True
opt@gsnShadeFillType = "pattern"  ; Pode ser "color" ou "pattern".
opt@gsnShadeLow      = 17         ; Tipo de preechimento.

; Preenche os valores menores que zero.
plot = gsn_contour_shade(plot,0,0,opt)

draw(plot)
frame (wks)

end
```

O resultado será:

Personalizando o tipo de preenchimento.

```
; Nome do script: cap21_ex37.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")

u = short2flt(f->uwnd)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex37")

res                                = True
res@tiMainString                   = "Vento zonal"
res@cnLevelSpacingF                = 5.0   ; Espaçamento entre as
                                           ; isolinhas.
res@gsnContourZeroLineThicknessF = 5.0   ; Espessura da linha
                                           ; do contorno zero.
res@gsnContourNegLineDashPattern = 1     ; Fixa as isolinhas
                                           ; negativas com o
                                           ; padrão 1.
res@gsnDraw                        = False ; Não cria a figura.
res@gsnFrame                       = False ; Não avança o frame.
res@tmYRTickSpacingF               = 1     ; Altera o espaçamento
                                           ; do eixo y de 1 em 1.

plot = gsn_csm_pres_hgt(wks,u(0,:,{0},:),res)

; Personalização do padrão de preenchimento.

opt                   = True
opt@gsnShadeFillType = "pattern" ; Pode ser "color" ou "pattern".
```

```
opt@gsnShadeLow      = 17        ; Tipo de preechimento para os
                                 ; menores valores.
opt@gsnShadeHigh     = 3         ; Tipo de preechimento para os
                                 ; maiores valores.

; Preenche os valores menores que zero com o padrão 17,
; e os valores acima de 15 com o padrão 3.

plot = gsn_contour_shade(plot,0,15,opt)

draw(plot)
frame (wks)

end
```

O resultado será:

Gráfico preenchido:

```
; Nome do script: cap21_ex38.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")

u = short2flt(f->uwnd)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex38")

res                                = True
res@tiMainString                   = "Vento zonal"
res@gsnContourZeroLineThicknessF = 5.0   ; Espessura da linha
                                         ; do contorno zero.
res@gsnContourNegLineDashPattern = 1     ; Fixa as isolinhas
                                         ; negativas com o
                                         ; padrão 1
                                         ; (linhas tracejadas).
res@cnLevelSelectionMode           = "ManualLevels" ; Fixa os níveis
                                                    ; de forma manual.
res@cnLevelSpacingF                = 5.0      ; Espaçamento dos valores.
res@cnMinLevelValF                 = -50.     ; Mínimo valor.
res@cnMaxLevelValF                 =  50.     ; Máximo valor.
res@cnLineLabelsOn                 = True     ; Habilita as linhas
                                              ; de contorno.
res@cnFillOn                       = True     ; Habilita o preechimento
                                              ; do gráfico.
res@cnFillPalette                  = "BlWhRe" ; Escolha da tabela de
                                              ; cores.
res@tiYAxisString                  = "Press"+atilde+"o (hPa)" ; Título
                                                              ; do eixo y.
res@cnLineLabelPlacementMode       = "constant" ; Define o tipo de rótulo
                                                ; dos contornos.
res@gsnLeftString                  = ""
res@gsnCenterString                = ""
res@gsnRightString                 = ""
res@tmYRMode                       = "Automatic" ; Desabilita o eixo y
                                                 ; da direita (altura).

plot = gsn_csm_pres_hgt(wks,u(0,:,{0},:),res)

end
```

O resultado será:

21.5.3 Pressão/altura versus tempo

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/height_time.shtml

Exemplo:

```
; Nome do script: cap21_ex39.ncl

begin

f = addfile("../../dados/tar.mensal.1979.1988.nc","r")

 ; short uwnd ( time, level, lat, lon ) = 3, 12, 73, 144
t = short2flt(f->air)

; A dimensão nível vertical deve ser a primeira dimensão,
; por isso, foi feita a reordenação das dimensões.

tN = t(level|:,time|:,lat|:,lon|:)

tN = tN-273.15 ; Tk->Tc

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex39")

; Plot da figura pressão (y) versus tempo (x).
; Lat/Lon escolhida "0"/"300". Com o símbolo "{}"
; pode-se utilizar a coordenada de latitude/longitude.

plot = gsn_csm_pres_hgt(wks,tN(:,:,{0},{300}),False)

end
```

O resultado será:

Personalizando o eixo do tempo:

```
; Nome do script: cap21_ex40.ncl

begin

f = addfile("../../dados/uwnd.mensal.2000.2005.nc","r")

timeUnits = f->time@units

ti = cd_inv_calendar(2000,01,01,00,0,0,timeUnits,0)
tf = cd_inv_calendar(2001,12,01,00,0,0,timeUnits,0)

timeUnits = "days since 1800-01-01 00:00:00"

 ; short uwnd ( time, level, lat, lon ) = 3, 12, 73, 144
u = short2flt(f->uwnd({ti:tf},:,:,:))

; uN teve a ordem das dimensões alterada de time, level, lat, lon para
; level, time, lat, lon.

; A dimensão nível vertical deve ser a primeira dimensão,
; para gerar esse tipo de gráfico em que a pressão está no
; eixo y e o tempo no eixo x, por isso, foi feita a
; reordenação das dimensões.

uN = u(level|:,time|:,lat|:,lon|:)

```

```
uN&time = ut_convert( uN&time, timeUnits )

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex40")

res                   = True
res@tmXBLabelAngleF   = -90    ; Inclinação dos rótulos
                               ; do eixo x.
res@tmXBLabelJust     = "CenterRight" ; Alinhamento dos
                                      ; rótulos do eixo x.

; Personaliza o eixo x (dimensão tempo).
resTick                 = True
resTick@ttmFormat       = "%c%Y"   ; %c é o mês no formato
                                   ; Mmm (Jan) e %Y é o ano no
                                   ; formato AAAA (2000).
resTick@ttmAxis         = "XB"     ; Qual eixo formatar?
                                   ; XB = eixo "X" e "B" = bottom.
resTick@ttmMajorStride = 2         ; Intervalo dos rótulos do eixo x.

; Fixa recursos necessários para padronizar a dimensão tempo.

time_axis_labels(uN&time,res,resTick)

; Plot da figura pressão (y) versus tempo (x).
; Lat/lon escolhida "0"/"300". Com o símbolo "{}"
; pode-se utilizar a coordenada de latitude e longitude.

plot = gsn_csm_pres_hgt(wks,uN(:,:,{0},{300}),res)

end
```

O resultado será:

Personalizando o padrão de preenchimento:

```
1   ; Nome do script: cap21_ex41.ncl
2
3   begin
4
5   f = addfile("../../dados/uwnd.mensal.2000.2005.nc","r")
6
7   timeUnits = f->time@units
8
9   ti = cd_inv_calendar(2000,01,01,00,0,0,timeUnits,0)
10  tf = cd_inv_calendar(2001,12,01,00,0,0,timeUnits,0)
11
12  timeUnits = "days since 1800-01-01 00:00:00"
13
14  u = short2flt(f->uwnd({ti:tf},:,:,:))
15
16  ; A dimensão nível vertical deve ser a primeira dimensão,
17  ; para gerar esse tipo de gráfico.
18
19  uN = u(level|:,time|:,lat|:,lon|:)
20
21  uN&time = ut_convert( uN&time, timeUnits )
22
23  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex41")
24
25  res                              = True
26  res@tmXBLabelAngleF              = -90
27  res@tmXBLabelJust                = "CenterRight"
28  res@gsnDraw                      = False
29  res@gsnFrame                     = False
30  res@gsnContourZeroLineThicknessF = 5.0
31  res@gsnContourNegLineDashPattern = 1     ; Fixa as isolinhas
32                                           ; negativas com o padrão 1
33
34  ; Personaliza o eixo x (dimensão tempo).
35  resTick                = True
36  resTick@ttmFormat      = "%c%Y"  ; %c é o mês no formato
37                                   ; Mmm (Jan) e %Y é o ano no
38                                   ; formato AAAA (2000).
39  resTick@ttmAxis        = "XB"    ; XB = eixo "X" e "B" = bottom.
40  resTick@ttmMajorStride = 2       ; Intervalo dos rótulos do eixo x.
41
42  time_axis_labels(uN&time,res,resTick)
43
44  plot = gsn_csm_pres_hgt(wks,uN(:,:,{0},{300}),res)
45
46  plot = ShadeLtGtContour(plot,-10.0,3,5.0,17)
47
48  draw (plot)
49  frame(wks)
50
51  end
```

O resultado será:

Gráfico preenchido e personalização da figura:

```
; Nome do script: cap21_ex42.ncl

begin

f = addfile("../../dados/uwnd.mensal.2000.2005.nc","r")

timeUnits = f->time@units

; Tempo inicial no formato AAAA,MM,DD,HH,MM,SS,unidade do tempo e
; 0 (não tem nenhum efeito).

ti = cd_inv_calendar(2000,01,01,00,00,00,timeUnits,0)
tf = cd_inv_calendar(2001,12,01,00,00,00,timeUnits,0)

u = short2flt(f->uwnd({ti:tf},:,:,:))

; uN teve a ordem das dimensões alterada de time, level, lat, lon
; para level, time, lat, lon.

uN = u(level|:,time|:,lat|:,lon|:)

uN&time = ut_convert( uN&time, timeUnits )

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex42")

```

```
gsn_define_colormap(wks,"BlWhRe")

res                                = True
res@tmXBLabelAngleF                = -90
res@tmXBLabelJust                  = "CenterRight"
res@cnFillOn                       = True
res@cnLinesOn                      = False
res@gsnSpreadColors                = True
res@cnLevelSelectionMode           = "ManualLevels"
res@cnMinLevelValF                 = -90.
res@cnMaxLevelValF                 =  90.
res@cnLevelSpacingF                =  5
res@tmYRTickSpacingF               = 2
res@pmLabelBarOrthogonalPosF       = -0.063
res@gsnLeftString                  = ""
res@gsnCenterString                = ""
res@gsnRightString                 = ""
res@tiMainString                   = "Vento zonal (m s~S~-1~N~)"
res@tiYAxisString                  = "Press"+atilde+"o (hPa)"
res@tmYRMode                       = "Automatic"

; Personaliza o eixo x (dimensão tempo).
resTick                 = True
resTick@ttmFormat       = "%c%Y"  ; %c é o mês no formato
                                  ; Mmm (Jan) e %Y é o ano no
                                  ; formato AAAA (2000).
resTick@ttmAxis         = "XB"    ; XB = eixo "X" e "B" = bottom.
resTick@ttmMajorStride = 2        ; Intervalo dos rótulos do eixo x.

time_axis_labels(uN&time,res,resTick)

plot = gsn_csm_pres_hgt(wks,uN(:,:,{0},{300}),res)

end
```

O resultado será:

21.5.4 Longitude versus tempo (Hovmöeller)

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/time_labels.shtml

```
; Nome do script: cap21_ex43.ncl

begin

f = addfile("../../dados/olr.01jan2000a31dez2004.nc","r")

timeUnits = f->time@units
ti        = cd_inv_calendar(2000,11,01,00,0,0,timeUnits,0)
tf        = cd_inv_calendar(2000,12,31,00,0,0,timeUnits,0)

latS = 2.5  ; Latitude de interesse
latN = 15.0 ; para fazer a média.

olr  = short2flt(f->olr({ti:tf},{latS:latN},:))

; Média na dimensão latitude (1) entre 2.5N e 15N.
; Com isso, restam apenas as dimensões [time | 61] x [lon | 144]
olrM = dim_avg_n_Wrap(olr,1)

timeUnits = "days since 1800-01-01 00:00:00"
olr&time  = ut_convert(olrM&time,timeUnits )

cmap = read_colormap_file("BlueDarkRed18")

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex43")

res                      = True
res@gsnMaximize          = True
res@cnFillOn             = True  ; Habilita o preechimento.
res@cnLineLabelsOn       = False ; Sem rótulos nas isolinhas.
res@cnInfoLabelOn        = False ; Sem informações da variável.
res@cnLinesOn            = False ; Não mostra as isolinhas.
res@cnFillPalette        = cmap(9::-1,:) ; Seleciona um subconjunto
                                         ; da escala de cores e
                                         ; inverte as cores.
res@cnLevelSelectionMode = "ExplicitLevels"
res@cnLevels             = (/160,180,200,220,240/) ; Valores que desejo.
res@trYReverse           = True                    ; Inverte o eixo y.
res@gsnLeftString        = ""
res@gsnCenterString      = ""
res@gsnRightString       = "(W m~S~-2~N~)"
res@tiMainString         = "ROL : M"+eacute+"dia entre " + \
                           latS + "N e " + latN+"N"
res@tiYAxisString        = "Dias"

; Fixa recursos especiais para o eixo do tempo.
resTick                = True
resTick@ttmFormat      = "%d%c%Y" ; dMmmAAAA => Exemplo: 1Jan2000
resTick@ttmAxis        = "YL"     ; Formata o eixo y da esquerda.
resTick@ttmMajorStride = 5        ; Espaçamento das datas no eixo y.
```

```
time_axis_labels(olrM&time,res,resTick)

plot = gsn_csm_hov(wks,olrM(:,{120:320}),res)

end
```

O resultado será:

21.6 Usando shapefile

21.6.1 Usando shapefile para mascarar dados

O NCL somente lê arquivo shapefile (.shp), mas não é capaz de modificá-lo.

Informações adicionais podem ser encontradas no link abaixo. Será necessário realizar o download do arquivo shapefile_utils.ncl (encontra-se no link abaixo) e salvá-lo em:

$NCARG_ROOT/lib/ncarg/nclscripts/csm

http://www.ncl.ucar.edu/Applications/shapefiles.shtml

Será necessário carregar a biblioteca abaixo no topo do seu script. Sem essa biblioteca o script não será executado. Essa biblioteca contém algumas rotinas para mostrar informações sobre o shapefile.

load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/shapefile_utils.ncl"

A linha 17 utiliza a função shapefile_mask_data que serve para mascarar o dado utilizando o shapefile.

A linha 43 (gsn_add_shapefile_polylines) é a função que cria o contorno do shapefile nos dados mascarados.

```
; Nome do script: cap21_ex44.ncl

begin

a = addfile("../../dados/precip.mon.mean.nc","r")

dSizes = getfiledimsizes(a)

p = a->precip(0,:,:) ; [time | 442] x [lat | 72] x [lon | 144]

p&lon = p&lon-360    ; O dado precisa estar no formato -180 a 180
                     ; por causa do shapefile que trabalha apenas
                     ; neste formato de longitude. Por isso, a
                     ; conversão para o formato -180 to 180.

; Importação do shapefile.
data_mask = shapefile_mask_data(p,\
            "../../dados/bacia_amaz/amazlm_1608.shp",True)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex44")

res                       = True
res@gsnAddCyclic          = False
res@mpMinLonF             = -90.0
res@mpMaxLonF             = -30.0
res@mpMinLatF             = -40.0
res@mpMaxLatF             = 10.0
res@gsnDraw               = False
res@gsnFrame              = False
res@mpDataBaseVersion     = "MediumRes"
res@mpFillOn              = False
res@mpOutlineBoundarySets = "National"

; Personalização da linha do shapefile.
lnres                     = True
lnres@gsLineColor         = "blue"
lnres@gsLineThicknessF    = 2.0

; Gera o mapa de precipitação mascarado com o uso do shapefile.
map_mask = gsn_csm_contour_map(wks,data_mask,res)

; Adiciona a linha do contorno do shapefile ao gráfico.
line_mask = gsn_add_shapefile_polylines(wks,map_mask,\
            "../../dados/bacia_amaz/amazlm_1608.shp",lnres)

draw(map_mask)
frame(wks)

end
```

O resultado será:

21.6.2 Extrair a série temporal de uma área usando shapefile

Às vezes é de interesse extrair os valores de um determinado domínio, normalmente utiliza-se um quadrilátero para realizar esta tarefa, e muitas das vezes são incluídas áreas indesejáveis. Com o uso do shapefile pode-se extrair exatamente a área desejada.

O script abaixo mostra como realizar essa tarefa.

Na linha 7, a função getfilevarnames retorna o nome das variáveis do arquivo aberto.

Na linha 14 é retida a variável de interesse (precip), por isso o valor 2 porque na sequência é mostrado o seguinte resultado:

(0) lon
(1) lat
(2) precip
(3) time

Na linha 16 é feita a importação da variável "time" que está no arquivo aberto.

As linhas 18 e 19 são as datas do arquivo que serão utilizadas nas linhas 20 e 21 que retorna o índice dessas datas.

Na linha 25 é feita a conversão da variável longitude para o formato -180 a 180.

Na linha 46 é a função que escreve o resultado das variáveis data e med_area no arquivo prec.bacia.amaz.txt com formatação ("%6i %4.2f").

```
; Nome do script: cap21_ex45.ncl

begin

a = addfile("../../dados/precip.mon.mean.nc","r")

vNames   = getfilevarnames(a)
print(vNames)         ; Verificar a posição do nome da variável
                      ; e trocar aqui pelo valor inteiro.
                      ; (0)     lat
                      ; (1)     lon
                      ; (2)     precip => Usar este índice "2"
                      ; (3)     time
nome_var = vNames(2)  ; Nome da variável.

time   = a->time
YYYYMM = cd_calendar(time,-1)
anoi   = 197901                  ; Data inicial no formato AAAAMM.
anof   = 197912                  ; Data final no formato AAAAMM.
istr   = ind( YYYYMM .eq. anoi ) ; Índice que será utilizado
iend   = ind( YYYYMM .eq. anof ) ; para selecionar o período de
                                 ; interesse.

p      = a->$nome_var$ ; Importação da variável de interesse.
p&lon  = p&lon-360     ; O dado precisa estar no formato -180 a 180
                       ; por causa do shapefile que trabalha apenas
                       ; neste formato.

lat  = a->lat               ; Importação da variável lat.
rad  = 4.0*atan(1.0)/180.0  ; Para radianos => 0.01745329
clat = cos(lat*rad)         ; Pesos que serão utilizados no
                            ; cálculo da média.

fout = "../../dados/prec.bacia.amaz.txt" ; Nome do arquivo de saída que
                                         ; terá duas colunas
                                         ; no formato ==> AAAAMM VALOR

system("/bin/rm -f " + fout) ; remove o arquivo caso ele exista
                             ; para evitar qualquer erro.

do i = istr,iend
   data      = YYYYMM(i)
   data_mask = shapefile_mask_data(p(i,:,:),\
               "../../dados/bacia_amaz/amazlm_1608.shp",True)
   med_area  = wgt_areaave_Wrap(data_mask,clat,1.0,1)
   write_table(fout,"a",[/data,med_area/],"%6i %4.2f")
   print(data + " " + med_area)
end do

end
```

21.7 Radiossondagem

21.7.1 Plotando uma radiossondagem

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/skewt.shtml

```
1   ; Nome do script: cap21_ex46.ncl
2
3   begin
4
5   ; Link para os dados de radiossondagem:
6   ; http://weather.uwyo.edu/upperair/sounding.html
7
8   ; As informações no topo do skewt são:
9   ; Cape  -  Convective Available Potential Energy [J]
10  ; Pwat  -  Precipitable Water [cm]
11  ; Shox  -  Showalter Index (stability)
12  ; Plcl  -  Pressure of the lifting condensation level [hPa]
13  ; Tlcl  -  Temperature at the lifting condensation level [C]
14
15  ; Para alterar o nome das palavras "Temperature (C)" ou
16  ; "Temperature (F)" para "Temperatura (°C)" ou "Temperatura (°F)",
17  ; altere as linhas 314 e 316. E para alterar o nome "Height (Km)" para
18  ; "Altura (Km)" altere a linha 370 da função skewt_func.ncl. Basta
19  ; copiar e colar o comando abaixo no seu terminal para editar o arquivo.
20  ; gedit "$NCARG_ROOT/lib/ncarg/nclscripts/csm/skewt_func.ncl"
21
22  ; Nome do arquivo de entrada.
23  nome_arq   = "../../dados/radiossondagem_belem_12Z_27mar2016.txt"
24  ; Nome do arquivo temporário.
25  arq_tmp    = "../../dados/tmp.txt"
26  ; Altere o nome do local aqui.
27  nome_local = "Bel"+eacute+"m/PA - 27/03/2016 - 1200Z"
28
29  ; o trecho abaixo gera um arquivo temporário chamado tmp.txt porque esta
30  ; função lê os dados de 1000hPa até 100hPa mesmo que o arquivo possua mai
31  ; níveis verticais.
32
33  ; A função "system" chama um comando do Unix para ser executado.
34
35  ; Checa (fileexists) se o arquivo existe no diretório. Caso não exista,
36  ; sai do script.
37
38  if ( fileexists(nome_arq) ) then
39     ; Seleciona apenas as linhas que contém os níveis verticais de
40     ; 1000-100hPa.
41     system("sed -n '/1000.0/,/100.0/p' " + nome_arq + " > " + arq_tmp + "")
42     nlin = numAsciiRow(arq_tmp) ; Número de linhas do arquivo.
43     ncol = numAsciiCol(arq_tmp) ; Número de colunas do arquivo.
44     dado = asciiread (arq_tmp,(/nlin,ncol/),"float") ; Leitura do arquivo.
45  else
46     print("")
47     print(" O arquivo não existe! Saindo do script! ")
48     print("")
49     exit
50  end if
51
52  ; Leitura das variáveis do arquivo.
53  p      = dado (:,0)     ; Pressão [mb/hPa].
```

```
z      = dado (:,1)     ; Altura [m].
tc     = dado (:,2)     ; Temperatura [C].
tdc    = dado (:,3)     ; Ponto de orvalho [C].
wdir   = dado (:,6)     ; Direção do vento em graus.
wspdk  = dado (:,7)     ; Velocidade do vento [knots].

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex46")

; Opções do Skew-t.
skewtOpts                   = True
skewtOpts@DrawIsotherm      = True       ; Linhas das isotermas.
skewtOpts@DrawIsobar        = True       ; Linhas das isobáras.
skewtOpts@DrawMixRatio      = True       ; Linhas da razão de mistura.
skewtOpts@DrawDryAdiabat    = True       ; Linhas da adibática seca.
skewtOpts@DrawMoistAdiabat  = True       ; Linhas da adibática úmida.
skewtOpts@DrawWind          = True       ; Desabilita a velocidade do
                                         ; vento.
skewtOpts@DrawStandardAtm   = True       ; Atmosfera padrão.
skewtOpts@DrawColLine       = True       ; Colore ou não as linhas do
                                         ; skew-t.
skewtOpts@DrawColAreaFill   = True       ; Preenchimento do fundo do
                                         ; skew-t.
skewtOpts@DrawFahrenheit    = False      ; Eixo x em Celsius (°C).
skewtOpts@DrawHeightScale   = True       ; Habilita a escala de altura.
skewtOpts@DrawHeightScaleFt = False      ; True = Feet | False = Km
skewtOpts@tiMainString      = nome_local ; Título da radiossondagem.

; Personalização do dado.
dataOpts                        = True
dataOpts@colTemperature         = "red"    ; Cor da linha da
                                           ; temperatura (T).
dataOpts@colDewPt               = "blue"   ; Cor da linha da temperatura
                                           ; do ponto de orvalho (Td).
dataOpts@colCape                = "black"  ; Cor da linha do CAPE.
dataOpts@colWindP               = "blue"   ; Cor da barbela do vento.
dataOpts@linePatternTemperature = 0        ; Estilo de linha da
                                           ; temperatura.
dataOpts@linePatternDewPt       = 0        ; Estilo de linha da
                                           ; temperatura do
                                           ; ponto de orvalho.
dataOpts@linePatternCape        = 2        ; Estilo de linha para o CAPE
dataOpts@xpWind                 = 45       ; Deslocamento na direção "x"
                                           ; da barbela do vento. O
                                           ; padrão é 45.
dataOpts@WspdWdir               = True     ; A partir da direção e
                                           ; velocidade, desenha a
                                           ; barbela do vento.
dataOpts@hemisphere             = "SH"     ; Hemisfério da radiossonda.
dataOpts@Wthin                  = 1        ; Espaçamento da barbela do
                                           ; vento.

skewt_bkgd = skewT_BackGround (wks,skewtOpts)
skewt_data = skewT_PlotData   (wks,skewt_bkgd,p,tc,tdc,z,wspdk, \
                               wdir,dataOpts)

draw (skewt_bkgd)
draw (skewt_data)

system("rm -f " + arq_tmp + "") ; Remove o arquivo temporário.

end
```

O resultado será:

Belém/PA - 27/03/2016 - 1200Z
Plcl=971 Tlcl[C]=23 Shox=0 Pwat[cm]=6 Cape[J]= 2404
P (hPa)
Temperature (C)
Height (Km)

21.7.2 Plotando duas radiossondagens

```
; Nome do script: cap21_ex47.ncl

begin

; Radiossondagem 1. Nome do arquivo 1.
nome_arq1   = "../../dados/radiossondagem_belem_12Z_27mar2016.txt"
arq_tmp1    = "../../dados/tmp1.txt"     ; Nome do arquivo temporário 1.
nome_local1 = "Bel"+eacute+"m/PA (preto)"; Altere o nome do local aqui.

; Radiossondagem 2
nome_arq2   = "../../dados/radiossondagem_porto_alegre_12Z_27mar2016.txt"
arq_tmp2    = "../../dados/tmp2.txt"
nome_local2 = "Porto Alegre/RS (vermelho)"

data        = "27/03/2016 - 1200Z" ; Considerando que a data é a mesma
                                   ; para as duas radiossondagens.

; o trecho abaixo gera um arquivo temporário chamado tmp*.txt porque
; esta função lê dados de 1000hPa até 100hPa mesmo que o arquivo possua
; mais níveis verticais.

; A função "system" chama um comando do Unix para ser executado.

; Checa (fileexists) se o arquivo existe no diretório. Caso não exista,
; sai do script.

if ( fileexists(nome_arq1) .and. fileexists(nome_arq2) ) then

   ; Seleciona apenas as linhas que contém os níveis verticais de
   ; 1000-100hPa.
   system("sed -n '/1000.0/,/100.0/p' " + nome_arq1 + " > " + \
          arq_tmp1 + " ")
   system("sed -n '/1000.0/,/100.0/p' " + nome_arq2 + " > " + \
          arq_tmp2 + " ")

   nlin1 = numAsciiRow(arq_tmp1) ; Número de linhas do arquivo 1.
   ncol1 = numAsciiCol(arq_tmp1) ; Número de colunas do arquivo 1.
   dado1 = asciiread (arq_tmp1,(/nlin1,ncol1/),"float")

   nlin2 = numAsciiRow(arq_tmp2)
   ncol2 = numAsciiCol(arq_tmp2)
   dado2 = asciiread (arq_tmp2,(/nlin2,ncol2/),"float")
else
   print("")
   print(" O arquivo não existe! Saindo do script! ")
   print("")
   exit
end if

; Leitura das variáveis do radiossondagem 1.
p1    = dado1(:,0)    ; Pressão [mb / hPa].
z1    = dado1(:,1)    ; Altura [m].
tc1   = dado1(:,2)    ; Temperatura [C].
tdc1  = dado1(:,3)    ; Ponto de orvalho [C].
wdir1 = dado1(:,6)    ; Direção do vento em graus.
wspd1 = dado1(:,7)    ; Velocidade do vento [knot].

; Leitura das variáveis do radiossondagem 2.
p2    = dado2(:,0)    ; Pressão [mb / hPa].
z2    = dado2(:,1)    ; Geopotential [gpm].
```

```
tc2   = dado2(:,2)    ; Temperatura [C].
tdc2  = dado2(:,3)    ; Ponto de orvalho [C].
wdir2 = dado2(:,6)    ; Direção do vento em graus.
wspd2 = dado2(:,7)    ; Velocidade do vento [knot].

p1@_FillValue    = -999 ; Definindo valores ausentes (-999)
z1@_FillValue    = -999 ; caso a radiossondagem possua. Não esqueça
tc1@_FillValue   = -999 ; de preencher no seu dado com os valores -999.
tdc1@_FillValue  = -999
wdir1@_FillValue = -999
wspd1@_FillValue = -999

p2@_FillValue    = -999
z2@_FillValue    = -999
tc2@_FillValue   = -999
tdc2@_FillValue  = -999
wdir2@_FillValue = -999
wspd2@_FillValue = -999

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex47")

skewtOpts                    = True
skewtOpts@DrawColAreaFill    = True    ; Preenchimento do fundo
                                       ; (cor verde) do skew-t.
skewtOpts@tiMainString       = nome_local1 + " e " + nome_local2 + \
                               "~C~        " + data
skewtOpts@DrawFahrenheit     = False   ; Eixo x em Celsius (C).
skewtOpts@DrawHeightScale    = True    ; Habilita a escala de altura,
                                       ; lado direito.
skewtOpts@DrawHeightScaleFt = False    ; True = Feet | False = Km.

dataOpts                    = True     ;
dataOpts@colTemperature     = "black"  ; Cor da linha da temperatura (T).
dataOpts@colDewPt           = "black"  ; Cor da linha da temperatura
                                       ; do ponto de orvalho (Td).
dataOpts@colWindP           = "black"  ; Cor da barbela do vento.
dataOpts@PrintZ             = False    ; Não mostra a altura do lado
                                       ; esquerdo.
dataOpts@ThermoInfo         = False    ; Não mostra as infomações
                                       ; termodinâmicas.
                                       ; no topo do diagrama.
dataOpts@xpWind             = 45       ; Deslocamento na direção "x" da
                                       ; barbela do vento.
                                       ; O padrão é 45.
dataOpts@WspdWdir           = True     ; A partir da direção e
                                       ; velocidade, desenha
                                       ; a barbela do vento.
dataOpts@hemisphere         = "SH"     ; Hemisfério usado: "NH" ou "SH".
dataOpts@Wthin              = 2        ; Espaçamento da barbela do vento.

dataOpts@linePatternTemperature = 0    ; Estilo de linha para a
                                       ; temperatura.
dataOpts@linePatternDewPt       = 1    ; Estilo de linha para a
                                       ; temperatura do
                                       ; ponto de orvalho.

skewt_bkgd = skewT_BackGround (wks, skewtOpts)

draw (skewt_bkgd)

; Gera a primeira radiossondagem - Belém
```

```
skewt_data = skewT_PlotData(wks,skewt_bkgd,p1,tc1,tdc1,z1,wspd1, \
                            wdir1,dataOpts)

draw (skewt_data)

dataOpts@Wthin              = 2     ; Espaçamento da barbela do vento.
dataOpts@xpWind             = 42    ; Localização do eixo do vento.
dataOpts@colTemperature     = "red"
dataOpts@colDewPt           = "red"
dataOpts@colWindP           = "red"

; Gera a segunda radiossondagem - Porto Alegre
skewt_data = skewT_PlotData(wks,skewt_bkgd,p2,tc2,tdc2,z2,wspd2, \
                            wdir2,dataOpts)

draw (skewt_data)
frame(wks)

system("rm -f " + arq_tmp1 + "") ; Remove o arquivo temporário.
system("rm -f " + arq_tmp2 + "")

end
```

O resultado será:

21.8 Meteogramas

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/meteo.shtml

```
; Nome do script: cap21_ex48.ncl

begin

; Variáveis necessárias:
; Componente u [u] em níveis verticais (m/s).
; Componente v [v] em níveis verticais (m/s).
; Temperatura do ar [air] em níveis verticais (°C).
; Umidade relativa do ar [rhum] em níveis verticais (%).
; Precipitação (mm/hora).
; Temperatura à superfície (°C).

; As dimensões de u, v, air e rhum precisam ter a seguinte disposição
; level e time e não time e level, sem isso, vai dar erro. Por isso, é
; feita a reordenação das dimensões.

f = addfile("../../dados/meteograma.uwnd.vwnd.air.pres.prec.t2m.nc","r")

;;;
```

```
diai       = str_get_cols(tostring(datai),6,7)
mes        = str_get_cols(tostring(datai),4,5)
ano        = str_get_cols(tostring(datai),0,3)
horai      = str_get_cols(tostring(datai),8,9)
dlocal     = diai+"/"+mes+"/"+ano+" 00"+horai+"z"
nome_data  = nlocal+" - "+dlocal

if (mes.eq."01") then mesc = "Jan" end if
if (mes.eq."02") then mesc = "Fev" end if
if (mes.eq."03") then mesc = "Mar" end if
if (mes.eq."04") then mesc = "Abr" end if
if (mes.eq."05") then mesc = "Mai" end if
if (mes.eq."06") then mesc = "Jun" end if
if (mes.eq."07") then mesc = "Jul" end if
if (mes.eq."08") then mesc = "Ago" end if
if (mes.eq."09") then mesc = "Set" end if
if (mes.eq."10") then mesc = "Out" end if
if (mes.eq."11") then mesc = "Nov" end if
if (mes.eq."12") then mesc = "Dez" end if

; short air(time, level, lat, lon)
;            20 ,   12 , 73 , 144
; Como foram fixadas as dimensões lat e lon, houve a redução de 4 para
; 2 dimensões, isto é, time e level.
tempisobar  = short2flt(f->$t$(ti:tf,{z1:z2},{lat},{lon}))

; Reordenando de time e level para level e time e convertendo de Kelvin
; para Celsius.
tempisobarN = tempisobar(level|:,time|:)-273.15

levels = f->level(0:6)  ; Dimensions and sizes: [level | 7]

taus = ispan(0,nt,dt) ; Horário da observação. 0,6,12,18,24,30,36,...,114

; short (time, level, lat, lon) = 20,12,73,144
rh = short2flt(f->$ur$(ti:tf,{z1:z2},{lat},{lon}))
rhN = rh(level|:,time|:) ; Reordenando as dimensões para level e time.

; short (time, level, lat, lon) = 20,12,73,144
ugrid = short2flt(f->$u$(ti:tf,{z1:z2},{lat},{lon}))
ugridN = ugrid(level|:,time|:) ; Reordenando as dimensões para level
                               ; e time.

; short (time, level, lat, lon) = 20,12,73,144
vgrid = short2flt(f->$v$(ti:tf,{z1:z2},{lat},{lon}))
vgridN = vgrid(level|:,time|:) ; Reordenando as dimensões para level
                               ; e time.

; double pre(time, lat, lon) = 20,73,144
rain = f->$ppt$(ti:tf,{lat},{lon})

; double tmp(time, lat, lon) = 20,73,144
tempht = f->$tsfc$(ti:tf,{lat},{lon})

; Suavização da temperatura e umidade relativa. Opcional.
smothtemp = smth9(tempisobarN,0.50,-0.25,False)
smothrh   = smth9(rhN,0.50,-0.25,False)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex48")
```

```
116
117   ; Define novas cores.
118   colors = (/                                                          \
119               (/255,255,255/),   (/  0,  0,  0/),   (/255,255,255/),  \
120               (/255,255,255/),   (/255,255,255/),   (/240,255,240/),  \
121               (/220,255,220/),   (/190,255,190/),   (/120,255,120/),  \
122               (/ 80,255, 80/),   (/ 50,200, 50/),   (/ 20,150, 20/),  \
123               (/255,  0,  0/)                                         \
124               /) / 255.0
125
126   gsn_define_colormap(wks,colors) ; Cria um mapa de cores com as cores
127                                   ; acima.
128
129   ; Habilita a personalização das variáveis.
130   rh_res      = True
131   temp_res    = True
132   uv_res      = True
133   rain_res    = True
134   tempsfc_res = True
135
136   ; Recursos para rh_res, temp_res e uv_res.
137   rh_res@trYReverse    = True     ; Inverte os valores do eixo y.
138   rh_res@gsnDraw       = False    ;
139   rh_res@gsnFrame      = False    ;
140   rh_res@vpXF          = 0.15     ; Localização em x da figura.
141   rh_res@vpYF          = 0.90     ; Localização em y da figura.
142   rh_res@vpWidthF      = 0.7      ; Largura.
143   rh_res@vpHeightF     = 0.40     ; Altura.
144
145   temp_res             = rh_res   ; Copia até aqui os recursos de rh_res
146   uv_res               = rh_res   ; para temp_res e uv_res.
147
148   ; Recursos para a umidade relativa.
149   rh_res@gsnSpreadColors    =  True     ; Utiliza todas as cores.
150   rh_res@gsnSpreadColorEnd = -2        ; Salva a última cor.
151   rh_res@cnFillOn           = True      ; Gráfico preenchido.
152   rh_res@tiMainString       = nome_data ; Título da figura.
153   rh_res@sfXArray           = taus      ; Valores do eixo x.
154   rh_res@sfYArray           = levels    ; Valores do eixo y.
155   rh_res@tiYAxisString      = "Press"+atilde+"o (hPa)" ; Título do eixo y.
156   rh_res@cnInfoLabelOn      = False  ; Desabilita as informações do
157                                      ; contorno.
158
159   ; Personalização do eixo y esquerdo do gráfico de seção vertical.
160   rh_res@tmYLMode           = "Explicit" ; Defino o eixo y esquerdo (L) do
161                                          ; meu jeito.
162   rh_res@tmYLValues         = levels     ; Valores do eixo y.
163   rh_res@tmYLLabels         = levels     ; Rótulos para cada valor do
164                                          ; eixo y.
165
166   ; Personalização do eixo x inferior do gráfico de seção vertical.
167   rh_res@tmXBMode           = "Explicit" ; Defino o eixo x inferior (B) do
168                                          ; meu jeito.
169   rh_res@tmXBValues         = taus       ; Valores do eixo x.
170                               ; Rótulos do eixo x.
171   rh_res@tmXBLabels         = (/tointeger(diai)+mesc  ,"06z","12z","18z",\
172                                 tointeger(diai)+1+mesc,"06z","12z","18z",\
173                                 tointeger(diai)+2+mesc,"06z","12z","18z",\
174                                 tointeger(diai)+3+mesc,"06z","12z","18z",\
175                                 tointeger(diai)+4+mesc,"06z","12z","18z"/)
176   rh_res@tmXBLabelAngleF      = 90. ; Gira os rótulos do eixo x inferior.
```

```
177   rh_res@tmXBLabelJust        = "CenterCenter" ; Alinhamento dos rótulos
178                                                ; do eixo x inferior (B).
179   rh_res@tmXBLabelFontHeightF = 0.016          ; Tamanho da fonte do eixo
180                                                ; x inferior (B).
181
182   ; Recursos para a temperatura do ar.
183   temp_res@sfXArray           = taus            ; Valores do eixo x.
184   temp_res@sfYArray           = levels          ; Valores do eixo y.
185   temp_res@cnLineThicknessF   = 4.0             ; Espessura das linhas de
186                                                 ; contorno.
187   temp_res@cnLineColor        = "Red"           ; Cor da linha.
188   temp_res@cnInfoLabelOn      = False           ; Desabilita as
189                                                 ; informações do contorno.
190   temp_res@gsnContourZeroLineThicknessF = 6.0   ; Espessura do contorno
191                                                 ; de valor zero de
192                                                 ; temperatura.
193
194   ; Recursos para a velocidade do vento.
195   uv_res@vfXArray             = taus       ; Valores do eixo x.
196   uv_res@vfYArray             = levels     ; Valores do eixo y.
197   uv_res@vcRefAnnoOn          = False      ; Desabilita o vetor
198                                            ; de referência.
199   uv_res@vcRefLengthF         = 0.040      ; Define o comprimento do vetor.
200   uv_res@vcGlyphStyle         = "WindBarb" ; Habilita barbela do vento.
201   uv_res@vcMapDirection       = False      ; Necessário para dissociar o
202                                            ; plano de coordendas a
203                                            ; a partir da barbela do vento.
204
205   ; Recursos para a precipitação (gráfico de linha).
206   rain_res@vpXF              = 0.15          ; Posição x para desenhar
207                                              ; a figura.
208   rain_res@vpYF              = 0.40          ; Posição y para desenhar
209                                              ; a figura.
210   rain_res@vpWidthF          = 0.70          ; Largura da figura.
211   rain_res@vpHeightF         = 0.10          ; Altura da figura.
212   rain_res@trYMinF           = 0.0           ; Mínimo valor do eixo y.
213   rain_res@trYMaxF           = max(rain)+1.0 ; Máximo valor do eixo y.
214   rain_res@trXMinF           = min(taus)     ; Mínimo valor do eixo x.
215   rain_res@trXMaxF           = max(taus)     ; Máximo valor do eixo x.
216   rain_res@tiXAxisString     = ""            ; Título do eixo x.
217   rain_res@tiYAxisString     = "Prec total "+dt+"h" ; Título do eixo y.
218   rain_res@tmXBMode          = "Explicit"     ; Defino o eixo x inferior
219                                                ; (B) do meu jeito.
220   rain_res@tmXBValues        = taus           ; Valores para o eixo x
221                                                ; inferior (B).
222                                ; Rótulos do eixo x.
223   rain_res@tmXBLabels        = (/tointeger(diai)+mesc,"06z","12z","18z",\
224                                  tointeger(diai)+1+mesc,"06z","12z","18z",\
225                                  tointeger(diai)+2+mesc,"06z","12z","18z",\
226                                  tointeger(diai)+3+mesc,"06z","12z","18z",\
227                                  tointeger(diai)+4+mesc,"06z","12z","18z"/)
228   rain_res@tmXTOn            = False  ; Desabilita os traços do eixo x
229                                       ; superior (T).
230   rain_res@gsnDraw           = False
231   rain_res@gsnFrame          = False
232   rain_res@gsnYRefLine       = 0.0   ; Cria linha de referência
233                                      ; no valor 0.0.
234   rain_res@gsnAboveYRefLineColor = "orange" ; Cor acima da linha de
235                                              ; referência.
236   rain_res@gsnXYBarChart     = True           ; Habilita gráfico de barras.
237
238
```

```
239  ; Recursos para a temperatura à superfície (gráfico de linha).
240  tempsfc_res@vpXF          = 0.15   ; Posição x para desenhar a figura.
241  tempsfc_res@vpYF          = 0.22   ; Posição y para desenhar a figura.
242  tempsfc_res@vpWidthF      = 0.70   ; Largura da figura.
243  tempsfc_res@vpHeightF     = 0.10   ; Altura da figura.
244  tempsfc_res@trXMaxF       = max(taus)       ; Máximo valor do eixo x.
245  tempsfc_res@trYMaxF       = max(tempht)+.5  ; Máximo valor do eixo y.
246  tempsfc_res@trYMinF       = min(tempht)-.5  ; Mínimo valor do eixo y.
247  tempsfc_res@tiYAxisString = "Temp a 2m"     ; Título do eixo y da figura.
248  tempsfc_res@tmXBMode      = "Explicit" ; Defino o eixo x inferior (B)
249                                          ; do meu jeito.
250  tempsfc_res@tmXBValues    = taus       ; Valores do eixo x inferior (B).
251  tempsfc_res@tmXBLabels    = (/tointeger(diai)+mesc,"06z","12z","18z",\
252                               tointeger(diai)+1+mesc,"06z","12z","18z",\
253                               tointeger(diai)+2+mesc,"06z","12z","18z",\
254                               tointeger(diai)+3+mesc,"06z","12z","18z",\
255                               tointeger(diai)+4+mesc,"06z","12z","18z"/)
256  tempsfc_res@tmXTOn        = False    ; Desabilita os traços do eixo x
257                                       ; superior (T).
258  tempsfc_res@xyLineThicknesses = 2    ; Espessura da linha.
259  tempsfc_res@xyLineColor    =  "red"  ; Define cor da linha.
260  tempsfc_res@gsnDraw         = False
261  tempsfc_res@gsnFrame        = False
262
263  ; Gerando as figuras.
264  rhfill    = gsn_contour(wks,smothrh,rh_res)
265  templine  = gsn_contour(wks,smothtemp,temp_res)
266  templine  = ColorNegDashZeroPosContour(templine,"blue","black","red")
267  windlayer = gsn_vector(wks,ugridN,vgridN,uv_res)
268  rainhist  = gsn_csm_xy(wks,taus,rain,rain_res)
269  temptmsz  = gsn_csm_xy(wks,taus,tempht,tempsfc_res)
270
271  ; Sobreposição dos campos para gerar a figura final.
272  overlay(rhfill,templine)   ; Sobreposição do campo templine sobre o
273                             ; rhfill.
274  overlay(rhfill,windlayer)  ; Sobreposição do campo windlayer sobre os
275                             ; dois campos.
276  draw(rhfill)               ; Gera a sobreposição.
277  draw(rainhist)             ; Gera o gráfico de barras.
278  draw(temptmsz)             ; Gera o gráfico de linha.
279  frame(wks)                 ; E finalmente, avança o frame para juntar
280                             ; todas as páginas.
281
282  end
```

O resultado será:

21.9 Gráfico de pizza

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/pie_chart.shtml

```
; Nome do script: cap21_ex49.ncl

begin

; Média espacial da precipitação mensal sobre
; a América do Sul.
f = addfile("../../dados/SA.GPCP.prec.1979.2013.nc","r")

; float precip ( time, lat, lon ) = 420, 1, 1
ppt       = f->precip

; Calcula a Climatologia. [month | 12] x [lat | 1] x [lon | 1]
ppt_clima = clmMonTLL(ppt)

; Cálculo sazonal.
djf = ppt_clima(11,:,:) + ppt_clima(0,:,:) + ppt_clima(1,:,:)
mam = ppt_clima(2,:,:)  + ppt_clima(3,:,:) + ppt_clima(4,:,:)
jja = ppt_clima(5,:,:)  + ppt_clima(6,:,:) + ppt_clima(7,:,:)
son = ppt_clima(8,:,:)  + ppt_clima(9,:,:) + ppt_clima(10,:,:)

; Soma total.
soma = dim_sum_n_Wrap(ppt_clima,0)

; Cálculo sazonal em porcentagem (%).
djf_pct   = (djf(0,0) * 100.0)/soma(0,0)
```

```
26  mam_pct   = (mam(0,0) * 100.0)/soma(0,0)
27  jja_pct   = (jja(0,0) * 100.0)/soma(0,0)
28  son_pct   = (son(0,0) * 100.0)/soma(0,0)
29
30  ; Armazena os valores em um arranjo de 4 linhas e 1 coluna.
31  estacao      = new((/4,1/),"float")
32  estacao(0,:) = djf_pct
33  estacao(1,:) = mam_pct
34  estacao(2,:) = jja_pct
35  estacao(3,:) = son_pct
36
37  ; Meses que vão aparecer na legenda.
38  mes = (/"DJF","MAM","JJA","SON"/)
39
40  ; Cor para cada uma das estações.
41  cor = (/"red","green","orange","yellow"/)
42
43  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex49")
44
45  pcRes             = True
46  pcRes@gsnMaximize = True ; A figura ocupa toda a tela.
47
48  pcRes@tiMainString  = "Precipita"+cedil+atilde+ \
49                        "o sazonal (%) sobre a Am"+eacute+"rica do Sul"
50
51  plot = pie_chart(wks,estacao(:,0),mes,cor,pcRes)
52
53  end
```

O resultado será:

Outra possibilidade de gráfico de pizza:

```
; Nome do script: cap21_ex50.ncl

begin

; Média espacial da precipitação mensal sobre
; a América do Sul.
f = addfile("../../dados/SA.GPCP.prec.1979.2013.nc","r")

ppt       = f->precip

ppt_clima = clmMonTLL(ppt)

; Cálculo sazonal.
djf       = ppt_clima(11,:,:) + ppt_clima(0,:,:) + ppt_clima(1,:,:)
mam       = ppt_clima(2,:,:)  + ppt_clima(3,:,:) + ppt_clima(4,:,:)
jja       = ppt_clima(5,:,:)  + ppt_clima(6,:,:) + ppt_clima(7,:,:)
son       = ppt_clima(8,:,:)  + ppt_clima(9,:,:) + ppt_clima(10,:,:)

; Soma total.
soma      = dim_sum_n_Wrap(ppt_clima,0)

; Cálculo sazonal em porcentagem (%).
djf_pct   = (djf(0,0) * 100.0)/soma(0,0)
mam_pct   = (mam(0,0) * 100.0)/soma(0,0)
jja_pct   = (jja(0,0) * 100.0)/soma(0,0)
son_pct   = (son(0,0) * 100.0)/soma(0,0)

; Armazenamento dos valores em um arranjo.
estacao      = new((/4,1/),"float")
estacao(0,:) = djf_pct
estacao(1,:) = mam_pct
estacao(2,:) = jja_pct
estacao(3,:) = son_pct

; Meses que vão aparecer na legenda.
mes = (/"DJF","MAM","JJA","SON"/)

; Cor para cada uma das estações.
cor = (/"red","green","orange","yellow"/)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex50")

pcRes             = True
pcRes@pcLabelType = "block" ; Legenda separada da figura.
pcRes@gsnMaximize = True    ; A figura ocupa toda a tela.

pcRes@tiMainString  = "Precipita"+cedil+atilde+ \
                      "o sazonal (%) sobre a Am"+eacute+"rica do Sul"

plot = pie_chart(wks,estacao(:,0),mes,cor,pcRes)

end
```

O resultado será:

21.10 Diagrama de Taylor

Deve-se carregar a biblioteca taylor_diagram_enoble.ncl. Agradecimentos a Arielle Bassanelli por otimizar o código.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/taylor.shtml

```
; Nome do script: cap21_ex51.ncl

load "$NCARG_ROOT/lib/ncarg/nclscripts/contrib/acentos.ncl"
load "../../dados/CMIP5/taylor_diagram_enoble.ncl"

begin

observado = (/"GPCP"/) ; Nome do dado observado.

modelos   = (/"ACCESS1-0","BCC-CSM1.1","CANESM2","CCSM4",   \
              "CNRM-CM5","CSIRO-MK3-6-0","FGOALS-G2","GFDL-CM3", \
              "HADGEM2-CC","HADGEM2-ES"/) ; Nome dos modelos.

; Nome das variáveis de cada modelo.
nome_var  = (/"pr","pr","pr","pr","pr","pr","pr","pr","pr","pr","pr"/)

; Usado no nome dos arquivos => prec.djf.ACCESS1-0.nc e também no nome
; das legendas (DJF, MAM, JJA e SON).
estacao   = (/"djf","mam","jja","son"/)

; Criação de novas variáveis.
pptm = new((/dimsizes(modelos),78,1,1/),float)
ppto = new((/dimsizes(observado),78,1,1/),float)
std  = new((/dimsizes(modelos),1,1,1/),float)
r    = new((/dimsizes(modelos),1,1,1/),float)
stdo = new((/dimsizes(estacao)/),float)
rms  = new((/dimsizes(modelos),1,1,1/),float)
```

```
28  bias = new((/dimsizes(modelos),1,1,1/),float)
29  rmsN = new((/dimsizes(modelos),1,1,1/),float)
30
31  ; O comando system acessa comandos dos Shell e neste caso, os arquivos
32  ; estão sendo removidos caso eles existam.
33
34  system("rm -f tmp_????.txt")
35  system("rm -f l2c_????.txt")
36  system("rm -f plot_????.txt")
37  system("rm -f plot_???.txt")
38
39  do k = 0,dimsizes(estacao)-1
40     do i = 0,dimsizes(modelos)-1
41             ; Abertura do aquivo observado.
42        f = addfile("../../dados/CMIP5/netcdf/prec."+estacao(k)+"."+modelos
    +".nc","r")
43           ; Abertura dos aquivos dos modelos.
44        g = addfile("../../dados/CMIP5/netcdf/prec."+estacao(k)+"."+observa
    +".nc","r")
45                        ; Importação da variável observada.
46        pptm(i,:,0,0) = f->pr(:,0,0)
47                        ; Importação das variáveis dos modelos.
48        ppto(0,:,0,0) = g->pr(:,0,0)
49                        ; Cálculo do desvio padrão observado.
50        stdo(k)       = stddev(ppto(0,:,0,0))
51                        ; Cálculo do desvio padrão dos modelos divido
52                        ; pela observação.
53        std(i,:,0,0)  = stddev(( pptm(i,:,0,0))/stdo(k))
54                        ; Cálculo da correlação de Pearson.
55        r(i,:,0,0)    = escorc(ppto(0,:,0,0),pptm(i,:,0,0))
56                        ; Cálculo do RMSE.
57        rms(i,:,0,0)  = dim_rmsd(ppto(0,:,0,0),pptm(i,:,0,0))
58                        ; Cálculo do bias.
59        bias(i,:,0,0) = dim_avg_n(ppto(0,:,0,0)-pptm(i,:,0,0),0)
60                        ; Cálculo do RMSE normalizado.
61  ;      rmsN(i,:,0,0) = sqrt((rms(i,:,0,0)^2 - bias(i,:,0,0)^2))
62        rmsN(i,:,0,0) = rms(i,:,0,0)/( max(pptm(i,:,0,0)))-min(pptm(i,:,0,0
63     end do
64        ; Mostra os valores mínimo e máximo.
65        print("Min: "+min(rmsN) + " Max: " + max(rmsN))
66        ; Escreve no arquivo os valores do desvio padrão.
67        write_table("tmp_std"+k+".txt","a",[/std/],"%f")
68        ; Escreve no arquivo os valores da correlação.
69        write_table("tmp_cor"+k+".txt","a",[/r/],"%f")
70
71        ; Transforma coluna em linha.
72        system("cat tmp_std"+k+".txt | xargs > l2c_std"+k+".txt")
73        ; Salva no arquivo texto os valores do desvio padrão.
74        system("cat l2c_std"+k+".txt >> plot_std.txt ")
75
76        ; Transforma coluna em linha.
77        system("cat tmp_cor"+k+".txt | xargs > l2c_cor"+k+".txt")
78        ; Salva no arquivo texto os valores da correlação.
79        system("cat l2c_cor"+k+".txt >> plot_cor.txt ")
80  end do
81
82  ; Cria as variáveis que serão utilizadas para gerar o plot do diagrama
83  ; de Taylor. As variáveis precisam ter duas dimensões em que a primeira
84  ; dimensão é o experimento e a segunda são os valores do desvio padrão
85  ; e da correlação.
```

```
 86
 87   desvio = new ((/dimsizes(estacao),dimsizes(modelos)/),float)
 88   corr   = new ((/dimsizes(estacao),dimsizes(modelos)/),float)
 89
 90   ; Os arquivos texto de desvio padrão e correlação foram gerados acima
 91   ; e agora serão lidos e estão prontos para serem utilizados na função
 92   ; do plot do diagrama de Taylor.
 93
 94   desvio = asciiread("plot_std.txt",(/dimsizes(estacao), \
 95            dimsizes(modelos)/),"float")
 96   corr   = asciiread("plot_cor.txt",(/dimsizes(estacao), \
 97            dimsizes(modelos)/),"float")
 98
 99   wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex51")
100
101   res                = True
102   res@Markers        = (/16,16,16,16/)    ; Tipo de marcadador para cada
103                                           ; experimento.
104   res@Colors         = (/"red","green","blue","orange"/) ; Cores para cada
105                                                           ; experimento.
106   res@varLabels      = modelos                            ; Nome dos modelos.
107   res@centerDiffRMS = True                   ; Habilitas as linhas do RMSE.
108   res@stnRad         = (/0.25,0.5,0.75,1.25,1.50/); Linhas dos desvio padrão
109   res@ccRays         = (/0.2,0.4,0.6,0.8,0.9/)    ; Linhas da correlação.
110   res@caseLabels     = str_upper(estacao); Converte para maiúsculo os
111                                           ; nomes da variável estacao.
112                                           ; Essa linha refere-se aos rótulos
113                                           ; da legenda.
114   res@caseLabelsFontHeightF = 0.10    ; Tamanho do texto da legenda
115                                       ; do diagrama.
116   res@varLabelsFontHeightF  = 0.015   ; Tamanho da fonte do nome dos
117                                       ; modelos.
118   res@varLabelsYloc         = 1.7     ; Desloca para cima ou para baixo
119                                       ; o nome dos modelos.
120   res@gsMarkerSizeF         = 0.015   ; Tamanho do marcador da legenda.
121   res@tiMainFontHeightF     = 0.025   ; Tamanho da fonte do título da
122                                       ; figura
123   res@tiMainString          = "Precipita"+cedil+atilde+ \
124                               "o dos modelos do CMIP5: 1979 - 2005"
125
126   plot = taylor_diagram(wks,desvio,corr,res) ; Geração da figura.
127
128   ; Remove os arquivos desnecessários.
129   system("rm -f tmp_????.txt")
130   system("rm -f l2c_????.txt")
131   system("rm -f plot_????.txt")
132   system("rm -f plot_???.txt")
133
134   end
```

O resultado será:

21.11 Rosa dos ventos

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/rose.shtml

```
; Nome do script: cap21_ex52.ncl

begin

; As informações que aparecem no topo da rosa
; dos ventos significam:
; SpdAve = velocidade média do vento.
; SpdStd = desvio padrão médio do vento.
; DirAve = direção média do vento.
; No Calm Reports = Sem vento calmo reportado.
; Nwnd = número de observações do dado.
; O número na ponta de cada direção representa sua velocidade média.

f = addfile("../../dados/uv.10m.jan1979.dez2014.nc","r")

; short uwnd(time, level, lat, lon) = 432, 1, 1, 1
u = short2flt(f->uwnd(:,0,0,0))
; short vwnd(time, level, lat, lon) = 432, 1, 1, 1
v = short2flt(f->vwnd(:,0,0,0))

dirv = wind_direction(u,v,0) ; Calcula a direção do vento em graus.
velv = wind_speed(u,v)       ; Calcula a velocidade do vento (m/s).

numPetals = 8     ; Número de direções da rosa dos ventos:
                  ; N,NE,E,SE,S,SW,W,NW.
                  ; Esse é o valor padrão.
circFr    = 10.0 ; Frequência com que os círculos aparecem na figura.
```

```
wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex52")

res                 = True
res@tiMainString    = "Rosa dos ventos"

plot = WindRoseBasic(wks,velv,dirv,numPetals,circFr,res)

end
```

O resultado será:

Inserindo informações sobre a velocidade máxima:

```
; Nome do script: cap21_ex53.ncl

begin

; As informações que aparecem no topo da rosa
; dos ventos significam:
; SpdAve = velocidade média do vento.
; SpdStd = desvio padrão médio do vento.
; DirAve = direção média do vento.
; No Calm Reports = Sem vento calmo reportado.
; Nwnd = número de observações do dado.
; O número na ponta de cada direção representa sua velocidade média.

f = addfile("../../dados/uv.10m.jan1979.dez2014.nc","r")

; short uwnd(time, level, lat, lon) = 432, 1, 1, 1
u = short2flt(f->uwnd(:,0,0,0))
; short vwnd(time, level, lat, lon) = 432, 1, 1, 1
v = short2flt(f->vwnd(:,0,0,0))

dirv = wind_direction(u,v,0) ; Calcula a direção do vento em graus.
velv = wind_speed(u,v)       ; Calcula a velocidade do vento (m/s).
```

```
23
24  numPetals = 8              ; 8 é o número de direções
25                             ; (N,NE,E,SE,S,SW,W,NW) da rosa
26                             ; dos ventos. Esse é o valor padrão.
27  circFr    = 10.0           ; Frequência com que os círculos aparecem
28                             ; na figura.
29  spdBounds = (/2.,3.,4.,5./); Limites de velocidade para a linha mais
30                             ; espessa.
31
32  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex53")
33
34  res               = True
35  res@tiMainString  = "Rosa dos ventos"
36
37  plot = WindRoseThickLine(wks,velv,dirv,numPetals,circFr,spdBounds,res)
38
39  end
```

O resultado será:

Colorindo os intervalos de velocidade:

```
; Nome do script: cap21_ex54.ncl

begin

f = addfile("../../dados/uv.10m.jan1979.dez2014.nc","r")

u = short2flt(f->uwnd(:,0,0,0))
v = short2flt(f->vwnd(:,0,0,0))

dirv = wind_direction(u,v,0) ; Calcula a direção do vento em graus.
velv = wind_speed(u,v)       ; Calcula a velocidade do vento (m/s).

numPetals = 8

circFr    = 10.0  ; Frequência com que os círculos aparecem na figura.
spdBounds = (/2.,3.,4.,5./)*1.0 ; Limites de velocidade para
                                ; a linha mais espessa.

colorBounds = (/"blue","green","yellow","red"/)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex54")

res              = True
res@tiMainString = "Rosa dos ventos - Jan/1979 a dez/2014"
res@gsnFrame     = False

plot = WindRoseColor(wks,velv,dirv,numPetals,circFr,spdBounds, \
                     colorBounds,res)

; Criação da legenda da velocidade.

lbres                    = True
lbres@lbAutoManage       = False ; Necessário para controlar o tamanho.
lbres@vpWidthF           = 0.05  ; Largura da legenda.
lbres@vpHeightF          = 0.15  ; Altura da legenda.
lbres@vpXF               = 0.78  ; Posição da legenda na direção x.
lbres@vpYF               = 0.58  ; Posição da legenda na direção y.
lbres@lbBoxMajorExtentF  = 0.80  ; Adiciona espaço entre a legenda.
lbres@lbFillColors       = colorBounds ; Cores das caixas da legenda.
lbres@lbMonoFillPattern  = True  ; Caixas com preenchimento sólido.
lbres@lbLabelFontHeightF = 0.015 ; Tamanho da fonte.
lbres@lbPerimOn          = False ; Sem linhas ao redor da legenda.
lbres@lbTitleString      ="Velocidade do vento (m/s)"
lbres@lbTitleFontHeightF = 0.01  ; Tamanho do título da legenda.

; Cria os rótulos da legenda.

nbar   = dimsizes(spdBounds)
labels = new(nbar,string)
labels(0) = 0 + "-" + spdBounds(0)

do i=1,nbar-1
   labels(i) = spdBounds(i-1) + "-" +spdBounds(i)
end do

lbid = gsn_create_labelbar(wks,nbar,labels,lbres) ; Adiciona a legenda.

draw(lbid)
frame(wks)

end
```

O resultado será:

21.12 Boxplot

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/box.shtml

```
; Nome do script: cap21_ex55.ncl

begin

; O arquivo pr.djf.cx2.txt possui 4 colunas na ordem abaixo com
; valores de precipitação (mm/dia) para o nordeste da Amazônia
; para os meses de DJF (1979 a 2005).
; Col1   Col2      Col3      Col4
; GPCP ACCESS1-0 BCC-CSM1.1 CANESM2

atxt     = "../../dados/pr.djf.cx2.txt"  ; Nome do arquivo no formato txt.
nlinhas  = numAsciiRow(atxt) ; Número de linhas do arquivo.
ncolunas = numAsciiCol(atxt) ; Número de colunas do arquivo.

f = asciiread(atxt,(/nlinhas,ncolunas/),"float")

opt           = True ; Habilita personalização da estatística.
opt@PrintStat = True ; Mostra na tela o resultado da estátistica
                     ; da função stat_dispersion.

obs  = stat_dispersion(f(:,0),opt)
mod1 = stat_dispersion(f(:,1),opt)
mod2 = stat_dispersion(f(:,2),opt)
mod3 = stat_dispersion(f(:,3),opt)

; Os valores (2), (7), (8), (11) e (14) abaixo correspondem ao
; valor mínimo, primeiro quartil, mediana, terceiro quartil e
; o valor máximo da série da função stat_dispersion.

```

```
yval      = new((/4,5/),"float",-999.)
yval(0,0) = obs(2)   ; Tamanho da haste inferior.
yval(0,1) = obs(7)   ; Primeiro quartil.
yval(0,2) = obs(8)   ; Mediana.
yval(0,3) = obs(11)  ; Terceiro quartil.
yval(0,4) = obs(14)  ; Tamanho da haste superior.

yval(1,0) = mod1(2)  ; Tamanho da haste inferior.
yval(1,1) = mod1(7)  ; Primeiro quartil.
yval(1,2) = mod1(8)  ; Mediana.
yval(1,3) = mod1(11) ; Terceiro quartil.
yval(1,4) = mod1(14) ; Tamanho da haste superior.

yval(2,0) = mod2(2)  ; Tamanho da haste inferior.
yval(2,1) = mod2(7)  ; Primeiro quartil.
yval(2,2) = mod2(8)  ; Mediana.
yval(2,3) = mod2(11) ; Terceiro quartil.
yval(2,4) = mod2(14) ; Tamanho da haste superior.

yval(3,0) = mod3(2)  ; Tamanho da haste inferior.
yval(3,1) = mod3(7)  ; Primeiro quartil.
yval(3,2) = mod3(8)  ; Mediana.
yval(3,3) = mod3(11) ; Terceiro quartil.
yval(3,4) = mod3(14) ; Tamanho da haste superior.

x = ispan(1,ncolunas,1) ; Valores para o eixo x.

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex55")

res                 = True
res@tmXBLabelAngleF = 45.   ; Inclinação do texto.
res@tmXBLabels      = (/"GPCP","ACCESS1-0","BCC-CSM1.1","CANESM2"/)
res@tiMainString    = "Box plot default"

plot = boxplot(wks,x,yval,False,res,False)

draw(wks)
frame(wks)

end
```

O resultado será:

Personalizando o boxplot:

```
; Nome do script: cap21_ex56.ncl

begin

; O arquivo pr.djf.cx2.txt possui 4 colunas na ordem a
; baixo com valores de precipitação (mm/dia) para o
; nordeste da Amazônia para os meses de DJF (1979 a 2005).
; Col1   Col2      Col3      Col4
; GPCP ACCESS1-0 BCC-CSM1.1 CANESM2

atxt     = "../../dados/pr.djf.cx2.txt"  ; Nome do arquivo no formato txt.
nlinhas  = numAsciiRow(atxt) ; Número de linhas do arquivo.
ncolunas = numAsciiCol(atxt) ; Número de colunas do arquivo.

f = asciiread(atxt,(/nlinhas,ncolunas/),"float")

modelos = (/"GPCP","ACCESS1-0","BCC-CSM1.1","CANESM2"/)

opt           = True ; Habilita personalização
                     ; da estatística.
opt@PrintStat = True ; Mostra na tela o resultado
                     ; da estátistica da função stat_dispersion.

y = new((/dimsizes(modelos),nlinhas,ncolunas+1/),float)

x = ispan(1,ncolunas,1) ; Valores para o eixo x.

do j = 0,dimsizes(modelos)-1
   g = stat_dispersion(f(:,j),opt)
```

```
    y(j,:,0) = g(2)  ; Tamanho da haste inferior.
    y(j,:,1) = g(7)  ; Primeiro quartil.
    y(j,:,2) = g(8)  ; Mediana.
    y(j,:,3) = g(11) ; Terceiro quartil.
    y(j,:,4) = g(14) ; Tamanho da haste superior.
  end do

  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex56")

  res                  = True          ; Personalização do boxplot.
  res@tmXBLabels       = modelos       ; Rótulos do eixo x.
  res@tiMainString     = "Precipita"+cedil+atilde+ \
                         "o sobre o nordeste da Amaz"+ocirc+"nia"
  res@trYMinF          = 0.0          ; Mínimo valor do eixo y.
  res@trYMaxF          = 12.0         ; Máximo valor do eixo y.
  res@tmYLMode         = "Manual"     ; Eixo y esquerdo (YL) de forma
                                      ; manual.
  res@tmYLTickStartF   = res@trYMinF  ; Onde começam os valores do eixo y.
  res@tmYLTickEndF     = res@trYMaxF  ; Onde terminam os valores do eixo y.
  res@tmYLTickSpacingF= 1             ; Espaçamento entre os rótulos.
  res@tmYLMinorOn      = False        ; Desliga o minortick do eixo y
                                      ; esquerdo.
  res@tmYRBorderOn     = True         ; Habilita a borda do eixo y direito.
  res@tmXTBorderOn     = True         ; Habilita a borda do eixo x superior.
  res@tmXBLabelFontHeightF = 0.014    ; Tamanho da fonte do eixo x inferior.
  res@tiYAxisString    = "Precipita"+cedil+atilde+"o (mm/dia)"

  llres                    = True    ; Personalização das linhas do boxplot.
  llres@gsLineThicknessF   = 2.5     ; Espessura da linha.

  opti           = True ; Habilita personalização da caixa.
  opti@boxWidth  = 0.35 ; Largura da caixa do boxplot.
  opti@boxColors = (/"blue","red","green","orange"/) ; Cores de cada caixa.

  plot = boxplot(wks,x,y(:,dimsizes(modelos),:),opti,res,llres)

  draw(wks)
  frame(wks)

  end
```

O resultado será:

21.13 Sobreposição de campos

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/overlay.shtml

```
; Nome do script: cap21_ex57.ncl

begin

f = addfile("../../dados/uwnd.vwnd.nc","r")
g = addfile("../../dados/hgt.jan.mar.1979.nc","r")

 ; short uwnd ( time, level, lat, lon ) = 3, 12, 73, 144
u    = short2flt(f->uwnd)
 ; short vwnd ( time, level, lat, lon ) = 3, 12, 73, 144
v    = short2flt(f->vwnd)
wspd = wind_speed(u,v)
; short hgt ( time, level, lat, lon )  = 3, 12, 73, 144
geo  = short2flt(g->hgt)

wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex57")

; Personalização da variável vento.
resv                        = True
resv@gsnDraw                = False  ; Não gera a figura.
resv@gsnFrame               = False  ; Não avança o frame.
resv@vcRefLengthF           = 0.025  ; Tamanho do vetor de referência.
resv@vcRefMagnitudeF        = 20.0   ; Velocidade do vetor de
                                     ; referência.
resv@gsnLeftString          = ""     ; Não adiciona título no lado
                                     ; esquerdo da figura.
resv@gsnCenterString        = ""     ; Não adiciona título no centro
                                     ;  da figura.
resv@gsnRightString         = ""     ; Não adiciona título no lado
```

```
30                                          ; direito da figura.
31  resv@lbOrientation           = "vertical"     ; Orientação da legenda.
32  resv@vcRefAnnoOrthogonalPosF = -1.0           ; Deslocamento do vetor de
33                                                ; referência.
34  resv@vcGlyphStyle            = "CurlyVector"  ; Aplica efeito de
35                                                ; curvatura aos vetores.
36  resv@vcLineArrowThicknessF   = 2.0            ; Espessura dos vetores.
37  resv@lbTitleString           = "(m/s)"        ; Título da legenda.
38  resv@lbTitlePosition         = "Top"          ; Posição do título da
39                                                ; legenda.
40  resv@lbTitleFontHeightF      = 0.025          ; Tamanho da fonte da
41                                                ; legenda.
42  resv@vcLevelSelectionMode    = "ManualLevels" ; Fixa de forma manual os
43                                                ; contornos.
44  resv@vcMinLevelValF          = 10.0           ; Fixa o mínimo valor do
45                                                ; contorno.
46  resv@vcMaxLevelValF          = 40.0           ; Fixa o máximo valor do
47                                                ; contorno.
48  resv@vcLevelSpacingF         = 5.0            ; Fixa o intervalo dos
49                                                ; contornos.
50
51  ; Personalização da variável geopotencial.
52  resc                       = True
53  resc@gsnDraw               = False            ; Não gera a figura.
54  resc@gsnFrame              = False            ; Não avança o frame.
55  resc@cnFillOn              = True             ; Habilita preenchimento.
56  resc@cnLinesOn             = False            ; Desabilita as linhas de
57                                                ; contorno.
58  resc@mpMaxLatF             = 20.0             ; Latitude sul.
59  resc@mpMinLatF             = -60.0            ; Latitude norte.
60  resc@mpMinLonF             = -100.0           ; Longitude oeste.
61  resc@mpMaxLonF             = -10.0            ; Longitude leste.
62  resc@gsnLeftString         = ""               ; Não adiciona título no
63                                                ; lado esquerdo da figura.
64  resc@gsnCenterString       = ""               ; Não adiciona título no
65                                                ; centro da figura.
66  resc@gsnRightString        = ""               ; Não adiciona título no
67                                                ; lado direito da figura.
68  resc@tiMainString          = "Alt. geopot. (shaded) e vento em 200hPa"
69  resc@gsnMajorLonSpacing    = 10               ; Espaçamento do eixo x
70                                                ; da longitude.
71  resc@gsnMajorLatSpacing    = 10               ; Espaçamento do eixo y
72                                                ; da latitude.
73  resc@tmXBMinorOn           = False            ; Desabilita os traços
74                                                ; secundários do eixo x.
75  resc@tmYLMinorOn           = False            ; Desabilita os traços
76                                                ; secundários do eixo y.
77  resc@lbTitleString         = "(m)"            ; Título da legenda.
78  resc@lbTitlePosition       = "Bottom"         ; Posição do título da
79                                                ; legenda.
80  resc@lbTitleFontHeightF    = 0.02             ; Tamanho da fonte do
81                                                ; título da legenda.
82  resc@cnLevelSelectionMode = "ManualLevels"    ; Fixa de forma manual
83                                                ; os contornos.
84  resc@cnMinLevelValF        = 5000.0           ; Fixa o mínimo valor
85                                                ; do contorno.
86  resc@cnMaxLevelValF        = 5800.0           ; Fixa o máximo valor
87                                                ; do contorno.
88  resc@cnLevelSpacingF       = 100.0            ; Fixa o intervalo dos
89                                                ; contornos.
90  resc@cnFillPalette         = "prcp_1"         ; Definição da tabela
91                                                ; de cores.
```

```
 92
 93   ; Cria o plot dos vetores e velocidade (m/s).
 94   plot1  = gsn_csm_contour_map(wks,geo(0,{500},:,:),resc)
 95   ; Cria o plot da altura geopotencial (m).
 96   plot2 = gsn_csm_vector_scalar(wks,u(0,{200},:,:),v(0,{200},:,:), \
 97                                 wspd(0,{200},:,:),resv)
 98
 99   ; Essas linhas são necessárias para fazer a sobreposição dos campos.
100   overlay(plot1,plot2) ; Sobreposição do plot2 (altura geopotencial)
101                        ; sobre o plot1 (vento).
102   draw(plot1)          ; Gera o plot do vento.
103   frame(wks)           ; Avança o frame.
104
105   end
```

O resultado será:

Outra possibilidade, plotando ROL e precipitação:

```
1   ; Nome do script: cap21_ex58.ncl
2
3   begin
4
5   ; float precip(time, lat, lon) - jan1979 a out2015
6   f = addfile("../../dados/precip.mon.mean.nc","r")
7   ; short olr(time, lat, lon)    - jan2000 a dez2009
8   g = addfile("../../dados/olr.jan2000.dez2009.nc","r")
9
10  data = 200001 ; Data de interesse.
11
12  yrStrt1 = data
13  TIME1   = f->time                ; Importa a variável time do arquivo f.
14  YYYYMM1 = cd_calendar(TIME1,-1)  ; Data no formato YYYYMM (-1).
15  ti1     = ind(YYYYMM1.eq.yrStrt1); Encontra o índice correspondente a
16                                   ; yrStrt1.
17
18  yrStrt2 = data
19  TIME2   = g->time                ; Importa a variável time do arquivo g.
20  YYYYMM2 = cd_calendar(TIME2,-1)  ; Data no formato YYYYMM (-1).
21  ti2     = ind(YYYYMM2.eq.yrStrt2); Encontra o índice correspondente a
22                                   ; yrStrt1.
23
24  ppt = f->precip(ti1,::-1,:)      ; Inverte a latitude porque estava de
25                                   ; norte para sul.
26                                   ; Agora => lat: [-88.75..88.75] => Sul
27                                   ; para norte
28  rol = short2flt(g->olr(ti2,::-1,:)) ; Inverte a latitude porque estava
29                                      ; de norte para sul.
30                                      ; Agora => lat: [-90..90] => Sul
31                                      ; para norte
32
33  wks = gsn_open_wks("pdf","../../figuras/cap21/cap21_ex58")
34
35  gsn_define_colormap(wks,"precip2_17lev") ; Define a tabela de cores para
36                                           ; gráfico.
37
38  res1                    = True
39  res1@gsnDraw            = False          ; Não gera a figura.
40  res1@gsnFrame           = False          ; Não avança o frame.
41  res1@cnFillOn           = True           ; Habilita preenchimento.
42  res1@cnLinesOn          = False          ; Desabilita as linhas de
43                                           ; contorno.
44  res1@mpMaxLatF          = 20.0           ; Latitude sul.
45  res1@mpMinLatF          = -60.0          ; Latitude norte.
46  res1@mpMinLonF          = -100.0         ; Longitude oeste.
47  res1@mpMaxLonF          = -10.0          ; Longitude leste.
48  res1@gsnLeftString      = ""             ; Não adiciona título no lado
49                                           ; esquerdo da figura.
50  res1@gsnCenterString    = ""             ; Não adiciona título no centro
51                                           ; da figura.
52  res1@gsnRightString     = ""             ; Não adiciona título no lado
53                                           ; direito da figura.
54  res1@gsnMajorLonSpacing = 10             ; Espaçamento do eixo x da
55                                           ; longitude.
56  res1@gsnMajorLatSpacing = 10             ; Espaçamento do eixo y da
```

```
                                          ; latitude.
res1@tmXBMinorOn        = False           ; Desabilita os traços
                                          ; secundários do eixo x.
res1@tmYLMinorOn        = False           ; Desabilita os traços
                                          ; secundários do eixo y.
res1@tiMainString       = "Precipita"+cedil+atilde+ \
                          "o (shaded) e ROL (contorno)"

res2                          = True
res2@gsnDraw                  = False       ; Não gera a figura.
res2@gsnFrame                 = False       ; Não avança o frame.
res2@cnLineLabelPlacementMode = "constant"  ; Define como os rótulos
                                            ; serão mostrados.
res2@cnInfoLabelOn            = False       ; Não mostra as informações
                                            ; do contorno.
res2@gsnLeftString            = ""          ; Não adiciona título no
                                            ; lado esquerdo da figura.
res2@gsnCenterString          = ""          ; Não adiciona título no
                                            ; centro da figura.
res2@gsnRightString           = ""          ; Não adiciona título no
                                            ; lado direito da figura.

plot1 = gsn_csm_contour_map(wks,ppt,res1) ; Cria o plot da
                                          ; precipitação (mm/dia).
plot2 = gsn_csm_contour(wks,rol,res2)     ; Cria o plot da radiação
                                          ; de onlda longa (W/m^2).

; Essas linhas são necessárias para fazer a sobreposição dos campos.
overlay(plot1,plot2) ; Sobreposição do plot2 (ROL) sobre o plot1
                     ; (precipitação).
draw(plot1)          ; Gera o plot da precipitação.
frame(wks)           ; Avança o frame.

end
```

O resultado será:

22 Estatística com o NCL

22.1 Correlação

22.1.1 Correlação de Pearson entre dois campos multidimensionais

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/escorc.shtml

```
; Nome do script: cap22_ex01.ncl

begin

; O objetivo desse script é exercitar o cálculo da correlação entre duas
; variáveis (ppt e rol). Elas possuem números de tempos e resolução
; espacial diferentes.
; Utilizou-se a função f2fosh_Wrap para uniformizar o dado de rol
; removendo um ponto de latitude para ficar igual ao dado de
; precipitação. A função cd_calendar e ind são utilizadas para selecionar
; os tempos de interesse. Para utilizar a função f2fosh_Wrap, a
; latitude deve estar orientada de sul para norte.

; short olr(time, lat, lon)
; O dado vai de jan2000 a dez2009.
; lat: [90..-90] => norte para sul
; lon: [ 0..357.5]

f = addfile("../../dados/olr.jan2000.dez2009.nc","r")

; float precip(time, lat, lon)
; O dado vai de jan1979 a dez2015.
; lat: [88.75..-88.75] => norte para sul.
; lon: [1.25..358.75]

g = addfile("../../dados/precip.mon.mean.nc","r")

; O NCL lê os índices correspondentes a uma determinada data e não a
; data em si. Nesse caso, foram criadas duas variáveis TIME1 e TIME2
; porque a data de interesse possui índice diferente no arquivo
; "f" e "g". Por exemplo, no arquivo "f" a data 200001 corresponde ao
; índice 0, enquanto que no arquivo "g" essa data corresponde ao
; índice 252.

yrStrt1 = 200001 ; Tempo inicial no formato AAAAMM.
yrLast1 = 200912 ; Tempo final no formato AAAAMM.

TIME1   = f->time ; Exportação da variável time do arquivo "f".
YYYYMM1 = cd_calendar(TIME1,-1)  ; Tempo no formato AAAAMM (-1).
; Índice que corresponde ao tempo yrStrt1.
ti1 = ind(YYYYMM1.eq.yrStrt1)
; Índice que corresponde ao tempo yrLast1.
tf1 = ind(YYYYMM1.eq.yrLast1)

TIME2   = g->time ; Exportação da variável time do arquivo "g".
YYYYMM2 = cd_calendar(TIME2,-1)  ; Tempo no formato AAAAMM (-1)
; Índice que corresponde ao tempo yrStrt1.
ti2 = ind(YYYYMM2.eq.yrStrt1)
```

```
; Indice que corresponde ao tempo yrLast1.
tf2 = ind(YYYYMM2.eq.yrLast1)

; Latitude tem que ser de sul para norte (::-1).
rol     = short2flt(f->olr(ti1:tf1,::-1,:))
ppt     = g->precip(ti2:tf2,::-1,:)

; Cria média temporal para DJF.
ppt_DJF = month_to_season(ppt,"DJF")
; Cria média temporal para DJF.
rol_DJF = month_to_season(rol,"DJF")

; Realiza o regridding para deixar o dado com a mesma
; resolução espacial de ppt.
x = f2fosh_Wrap(rol_DJF)
y = ppt_DJF

printVarSummary(x)  ; time: [1753152..1832064]
                    ; lat: [-88.75..88.75]
                    ; lon: [1.25..358.75]

printVarSummary(y)  ; time: [65378..68665]
                    ; lat: [-88.75..88.75]
                    ; lon: [1.25..358.75]

; Calcula a correlação de Pearson na dimensão tempo (0) para as duas
; variáveis.

r = escorc_n(x,y,0,0)

printVarSummary(r) ; [72] x [144]

; Varíavel auxiliar para copiar as dimensões e coordenadas para r. Como
; foi fixado o primeiro tempo (poderia ser outro qualquer), com isso,
; ocorre a redução de dimensão de 3 para 2, dessa forma é possível copiar
; as dimensões de var_aux para r com o copy_VarCoords, uma vez que r
; possui apenas as dimensões latitude e longitude.

var_aux = ppt(0,:,:)

; Copia as coordenadas e dimensões de var_aux para r.
copy_VarCoords(var_aux,r)

printVarSummary(r)   ; [lat | 72] x [lon | 144]

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex01")

; Definição da escala de cores.
gsn_define_colormap(wks,"CBR_coldhot")

res                     = True
res@mpMaxLatF           = 20.0   ; Latitude sul.
res@mpMinLatF           = -60.0  ; Latitude norte.
res@mpMinLonF           = -100.0 ; Longitude oeste.
res@mpMaxLonF           = -10.0 ; Longitude leste.
res@cnFillOn            = True  ; Habilita preenchimento.
res@cnLinesOn           = False ; Desabilita as linhas de contorno.
res@tiMainString        = "Correla"+cedil+atilde+"o" +\
                          " entre prec e ROL"
```

```
res@cnFillMode            = "RasterFill" ; (grfill do GrADS).
; Escala de cores do meu jeito.
res@cnLevelSelectionMode = "ExplicitLevels"
; Valores que vão aparecer na barra de cores.
res@cnLevels              = (/-0.8,-0.6,-0.4,-0.2,0,0.2, \
                                 0.4,0.6,0.8/)
; Para cada valor está associada uma cor da escala de cores CBR_coldhot.
res@cnFillColors          = (/2,3,4,5,0,0,9,10,11,12/)
res@gsnMajorLonSpacing    = 10 ; Espaçamento do eixo x.
res@gsnMajorLatSpacing    = 10 ; Espaçamento do eixo y.
res@tmXBMinorOn           = False ; Desabilita os traços
                                  ; secundários
res@tmYLMinorOn           = False ; dos eixos x e y.

plot = gsn_csm_contour_map_ce(wks,r,res)

end
```

O resultado será:

22.1.2 Correlação de Pearson entre uma série temporal e um dado multidimensional com teste de significânia

```
; Nome do script: cap22_ex02.ncl

begin

; Os dados são referentes aos meses de DJF, são 90 meses.
; asst é a anomalia de SST e aprec é a anomalia de precipitação.

f = addfile("../../dados/anom.prec.dez1980.fev2010.nc","r")
g = addfile("../../dados/anom.sst.dez1980.fev2010.nc","r")

aprec = f->precip ; Dado espacial.

; [time | 90] x [lat | 72] x [lon | 144]
; time: [66078..76732]
; lat: [88.75..-88.75]
; lon: [1.25..358.75]

asst  = short2flt(g->anom) ; Série temporal.

; [time | 90] x [zlev | 1] x [lat | 1] x [lon | 1]
; time: [46355..57009]
; zlev: [ 0.. 0]
; lat: [   0..   0]
; lon: [   0..   0]

printVarSummary(aprec)
printVarSummary(asst)

; nt = 90, ny = 72, nx = 144. O que nos interessa é o número de tempos.
dim = dimsizes(aprec)

nt  = dim(0) ; Número de tempos do arquivo

x = aprec(:,::-1,:) ; Inverte a latitude do dado de precipitação.

y = asst(:,0,0,0) ; asst precisar ser uma série temporal.
                  ; Apenas a dimensão tempo precisar
                  ; variar, enquanto as demais são fixas.

r = escorc_n(x,y,0,0) ; Calcula a correlação de Pearson
                      ; entre uma série temporal (asst)
                      ; e um dado bidimensional (aprec).

printVarSummary(r)  ; [72] x [144]

var_aux = x(0,:,:)  ; Variável auxiliar para copiar as
                    ; dimensões e coordenadas para a
                    ; variável r.

copy_VarCoords(var_aux,r) ; Copia as dimensões e as
                          ; coordenadas de var_aux para r.

printVarSummary(r)   ; [lat | 72] x [lon | 144]
                     ; lat: [-88.75..88.75]
                     ; lon: [1.25..358.75]

prob = rtest(r,nt,0) ; Calcula a estatística
                     ; t = r * sqrt( (Nr-2) / (1-r^2) )
                     ; r = correlação e nt é o tamanho da
                     ; amostra (número de tempos) e 0 não
```

```
                        ; tem uso.

printVarSummary(prob)

copy_VarCoords(var_aux,prob) ; Copia as dimensões e as coordenadas de
                             ; var_aux para prob.

printVarSummary(prob)

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex02")

; Definição do mapa de cores a ser usado.
gsn_define_colormap(wks,"temp_diff_18lev")

res1                       = True
res1@cnFillOn              = True    ; Habilita preenchimento.
res1@cnLinesOn             = False   ; Desabilita as linhas de contorno.
res1@tiMainString          = "Correla"+cedil+atilde+  \
                             "o entre ASST~B~NINO3.4~N~ e APREC"
res1@gsnCenterString       = "Regi"+atilde+ \
                             "o pontilhada: 95% de " +\+
                             "signific"+acirc+"ncia."
res1@cnFillMode            = "RasterFill" ; (grfill do GrADS).
; Escala de cores do meu jeito.
res1@cnLevelSelectionMode = "ExplicitLevels"
; Valores que vão aparecer na barra de cores.
res1@cnLevels              = ispan(-9,9,1)/10.
; res1@cnFillColors => Para cada valor da correlação está
; associado uma cor da escala de cores da tabela de cores temp_diff_18lev
res1@cnFillColors          = (/2,3,4,5,6,7,8,9,10,0,0,12,13, \
                              14,15,16,17,18,19,20/)
res1@gsnMajorLonSpacing    = 30 ; Espaçamento do eixo x.
res1@gsnMajorLatSpacing    = 15 ; Espaçamento do eixo y.
res1@tmXBMinorOn           = False ; Desabilita os traços secundários
res1@tmYLMinorOn           = False ; dos eixos x e y.
res1@gsnFrame              = False ; Não avança o frame.
res1@gsnDraw               = False ; Não desenha.
res1@mpCenterLonF          = -180  ; Centraliza o mapa em -180.

; Cria o plot da correlação.
plot1 = gsn_csm_contour_map(wks,r,res1)

alfa = 0.207 ; Valor tabelado para 95% de significância e n = 90 = nt.

res2                        = True
res2@gsnFrame               = False ; Não avança o frame.
res2@gsnDraw                = False ; Não desenha.
; Defino a escala dos contornos do meu jeito.
res2@cnLevelSelectionMode   = "ExplicitLevels"
res2@cnInfoLabelOn          = False ; Desabilita as informações de
                                    ; contorno.
res2@cnLinesOn              = False ; Desabilita as linhas de contorno.
res2@cnLineLabelsOn         = False ; Desabilita os rótulos dos contornos.
res2@cnFillScaleF           = 0.7   ; Densidade do pontilhado
                                    ; no mapa. Quanto menor o valor,
                                    ; maior será a densidade de pontos.

; Cria o mapa da probabilidade.
plot2   = gsn_csm_contour(wks,prob,res2)
; Preenche todas as regiões menores do que alfa com
; o padrão 17 (pontilhado).
```

```
plot2   = ShadeLtContour(plot2,alfa,17)

overlay (plot1, plot2) ; Sobrepõe os campos, plot2 sobre o plot1.

draw(wks)

end
```

O resultado será:

22.2 Empirical Orthogonal Functions (EOF)

Para calcular EOF ou Principal Component Analysis (PCA) é necessário um arranjo em que o número de observações seja a dimensão mais a direita e geralmente essa dimensão é o tempo. O calculo é normalmente feito com anomalias, mas essa não é uma exigência. Os valores ausentes são ignorados ao calcular a matriz de correlação ou de variância. Os valores calculados são normalizados tal que a soma dos quadrados de cada padrão da EOF é igual a um.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Document/Functions/Built-in/eofunc.shtml

Para utilizar a função eofunc, são necessários três argumentos:

Sintaxe: eof = eofunc(data,neval,optEOF)

Em que:

1. **data**: É um arranjo multidimensional em que a dimensão mais a direita é o tempo.

2. **neval**: Um escalar que determina o número de autovalores (eigenvalues) e os autovetores (eigenvectors). O neval é usualmente menor do que ou igual ao número mínimo de observações ou número de variáveis.

3. **optEOF**: Uma variável lógica para os argumentos opcionais que podem ser avaliados. Pode assumir duas possibilidades, optEOF@jopt=1 (usa a matriz de correlação para calcular a EOF) ou optEOF@jopt=0 (usa a matriz de covariância para calcular a EOF). O valor padrão utilizado para calcular a EOF é zero.

O valor retornado é um arranjo contendo as EOF's normalizadas. O arranjo retornado tem o mesmo tamanho do dado de entrada sendo que a dimensão mais a direita (tempo) é removida e uma dimensão mais a esquerda é criada com o mesmo tamanho de neval.

A variável retornada tem os seguintes atributos:

1. **eval**: Um arranjo de uma dimensão do tamanho de neval que contém os eigenvalues.

2. **pcvar**: Um arranjo unidimensional do tipo float do tamanho de neval igual a porcentagem da variância associada com cada eigenvalue.

3. **pcrit**: O mesmo valor e tipo de optEOF@pcrit caso o usuário mude o valor padrão (50%).

4. **matrix**: Uma string especificando o tipo de matriz usada que pode ser correlation ou covariance. A matriz de covariância (covariance) é o padrão.

5. **method**: Uma string especificando se o dado de entrada foi ou não transposto com o objetivo de calcular os eigenvalues e os eigenvectors. Pode assumir dois valores, isto é, "transpose" ou "no transpose".

6. **eval_transpose**: Este atributo somente será retornado se method=transpose. O eva_transpose terá os eigenvalues da matriz de covariância transposta.

Para acessar esses atributos utiliza-se o símbolo @. Basta seguir o procedimento abaixo após calcular a EOF.

eof = eofunc_Wrap(x,neof,optEOF) ; Calcula a EOF.

Desejamos saber qual a matriz utilizada, basta inserir no script a linha abaixo:

print("==> " + eof@matrix) ; Resposta: ==> covariance

Apenas lembrando que o nome eof é o nome da variável que contém o cálculo da EOF.

A série temporal da amplitude associada com cada eigenvalue da EOF é calculada utilizando a função eofunc_ts. Essa função calcula a série temporal das amplitudes associadas a cada valor do eigenvalue da EOF.

Para mais informações, acessar o link abaixo:

http://www.ncl.ucar.edu/Document/Functions/Built-in/eofunc_ts.shtml

Para utilizar a função eofunc_ts, são necessários três argumentos:

Sintaxe: eof_ts = eofunc_ts(data,evec,optETS)

Em que:

1. **data**: É um arranjo multidimensional em que a dimensão mais a direita é o tempo.

2. **evec**: Um arranjo multidimensional contendo as EOF's que foram calculadas com a função eofunc.

3. **optETS**: Uma variável lógica para os argumentos opcionais que podem ser avaliados. Pode assumir dois valores, 0 (padrão) ou 1 que utiliza a matriz de dados padronizada para calcular a série temporal. O padrão é utilizar data e evec.

A variável retornada será um arranjo bidimensional que contém o mesmo tamanho dos eigenvalues escolhidos e o mesmo número de tempos do dado de entrada.

Ao calcular a série temporal é gerado o atributo ts_mean que é um arranjo de mesmo tamanho e tipo de evec que contém a média removida dos dados como parte do cálculo. Para acessar esse atributo utilize o símbolo @. Por exemplo:

print("==> " + eof_ts@ts_mean)

22.2.1 EOF utilizando ponderação

```
; Nome do script: cap22_ex03.ncl

begin

f = addfile("../../dados/slp.jan1979.dez2003.nc","r")

latS   =  25.0  ; Latitude sul.
latN   =  80.0  ; Latitude norte.
lonW   = -70.0  ; Longitude oeste.
lonE   =  40.0  ; Longitude leste.

yrStrt = 1979   ; Ano inicial.
yrLast = 2003   ; Ano final.

season = "DJF"  ; Escolha da estação a ser feita a EOF.

neof   = 3      ; Número de EOF's a serem calculadas.

optEOF      = True ; Personaliza a EOF.
optEOF@jopt = 0    ; 0 = Usa a matrix de covariância para calcular a EOF.
                   ; Esse é o padrão.
                   ; 1 = Usa a matrix de correlação para calcular a EOF.

optETS = False     ; Não personaliza a série temporal da EOF.

TIME = f->time     ; Importação da variável time do arquivo f.
YYYY = cd_calendar(TIME,-1)/100 ; Tempo no formato AAAAMM (opção -1).
                                ; A divisão por 100 mostra apenas os
                                ; anos no formato AAAA.
iYYYY = ind(YYYY.ge.yrStrt.and.YYYY.le.yrLast) ; Seleciona os índices
                                               ; entre 1979 e 2003.

slp = f->slp(iYYYY,:,:) ; Seleciona a variável slp apenas no período
                        ; de 1979 a 2003.

printVarSummary(slp) ; [time | 300] x [lat | 73] x [lon | 144]
                     ; time: [1569072..1787472]
                     ; lat: [90..-90]
                     ; lon: [ 0..357.5]

slp = lonFlip(slp)   ; Longitude no formato -180 a 180. Isso facilita
                     ; Definer a longitude.

printVarSummary(slp) ; lon: [-180..177.5]

SLP  = month_to_season(slp,season) ; Extrai apenas a média de DJF.
nyrs = dimsizes(SLP&time)          ; Número de anos.

printVarSummary(SLP) ; [time | 25] x [lat | 73] x [lon | 144] => 25 DJF's

rad = 4.0*atan(1.0)/180.0  ; Conversão de graus para radianos.
clat = f->lat              ; Importa a variável lat do arquivo f.
clat = sqrt(cos(rad*clat)) ; Cria os valores que serão utilizados na
                           ; ponderação.

```

```
wSLP           = SLP ; Copia os metadados de SLP para wSLP.
wSLP           = SLP*conform(SLP,clat,1) ; O valor 1 corresponde a
                                         ; dimensão de clat que
                                         ; equivale a SLP.
wSLP@long_name = "Wgt: "+wSLP@long_name  ; Atributo long_name foi
                                         ; adicionado a wSLsP.

; Reordena as dimensões de wSLP para x(lat,lon,time) porque antes era
; x(time,lat,lon).
x = wSLP({lat|latS:latN},{lon|lonW:lonE},time|:)

eof    = eofunc_Wrap(x,neof,optEOF)    ; Calcula a EOF.
eof_ts = eofunc_ts_Wrap(x,eof,optETS)  ; Calcula a série temporal da EOF.

print("==> " +eof@matrix)

printVarSummary(eof)
; Dimensions and sizes: [evn | 3] x [lat | 23] x [lon | 45]

printVarSummary(eof_ts)
; Dimensions and sizes: [evn | 3] x [time | 25]

; Normalização da série temporal.
dimx   = dimsizes(x) ; (0) 23
                     ; (1) 45
                     ; (2) 25
mln    = dimx(1)     ; Número de pontos de longitude (45 pontos).
sumWgt = mln*sum(clat({lat|latS:latN}))  ; 45 * 17.1868
eof_ts = eof_ts/sumWgt                   ; Normaliza a série temporal.
yyyymm = cd_calendar(eof_ts&time,-2)/100 ; Extrai apenas o ano e o mês
                                         ; da série temporal no
                                         ; formato AAAAMM.

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex03a")

gsn_define_colormap(wks,"BlWhRe")        ; Escolha do mapa de cores.

plot = new(neof,graphic) ; Cria um arranjo gráfico para criar o painel
                         ; com as 3 figuras.

; Personalização da EOF.
res                      = True
res@gsnDraw              = False  ; Não desenha a figura.
res@gsnFrame             = False  ; Não avança o plot.
res@gsnAddCyclic         = False  ; Quando o dado for regional.
res@mpMinLatF            = latS   ; Aplica zoom ao mapa.
res@mpMaxLatF            = latN
res@mpMinLonF            = lonW
res@mpMaxLonF            = lonE
res@cnFillOn             = True   ; Habilita o preenchimento do mapa.
res@cnLinesOn            = False  ; Desabilita as linhas de contorno.
res@lbLabelBarOn         = False  ; Desabilita a barra de cores para
                                  ; cada figura.
symMinMaxPlt(eof,16,False,res)    ; Define o valor mínimo e máximo.
                                  ; 16 representa o número de
                                  ; contornos desejados. False quer
                                  ; dizer que os valores estão dentro
                                  ; do valor máx/min. Útil para criar
                                  ; escalas de valores para a barra
```

```
                                    ; de cores.

; Personalização do painel de figuras.
resP                    = True  ; Habilita personalização dos paineis.
resP@gsnMaximize        = True  ; A figura ocupa todo o espaço da página.
resP@gsnPanelLabelBar   = True  ; Barra de cores comum para todos os
                                ; gráficos.

yStrt                   = yyyymm(0)/100      ; Extrai o primeiro ano.
yLast                   = yyyymm(nyrs-1)/100 ; Extrai o último ano.

resP@txString           = "SLP: "+season+": "+yStrt+"-"+yLast ; Título
                           ; da figura.

do n=0,neof-1
   res@gsnLeftString  = "EOF "+(n+1)  ; Título do lado esquerdo da
                                      ; figura.
   ; Título do lado direito da figura.
   res@gsnRightString = sprintf("%5.1f", eof@pcvar(n)) +"%"
   ; Gera o plot das 3 EOF's.
   plot(n)=gsn_csm_contour_map_ce(wks,eof(n,:,:),res)
end do

; Cria o painel com as 3 figuras (3 linhas e 1 coluna).
gsn_panel(wks,plot,(/neof,1/),resP)

; Geração da figura da série temporal da EOF.

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex03b")

; Extrai apenas o ano, sem a divisão por 100, o valor sai como AAAAMM.
year = yyyymm/100

rts                       = True   ; Habilita a personalização da série
                                   ; temporal.
rts@gsnDraw               = False  ; Não desenha a figura.
rts@gsnFrame              = False  ; Não avança o frame.
rts@vpHeightF             = 0.40   ; Altura da figura.
rts@vpWidthF              = 0.85   ; Largura da figura.
rts@vpXF                  = 0.10   ; Posição na página onde
rts@vpYF                  = 0.75   ; ficará a figura.
rts@tiYAxisString         = "hPa"  ; Título do eixo y.
rts@gsnYRefLine           = 0.0    ; Linha de referência em zero.
rts@gsnXYBarChart         = True   ; Habilita gráfico de barras.
rts@gsnAboveYRefLineColor = "red"  ; Cor "red" acima da linha de
                                   ; referência.
rts@gsnBelowYRefLineColor = "blue" ; Cor "blue" abaixo da linha de
                                   ; referência.
rts@trYMaxF               = 0.15   ; Máximo valor do eixo y.
rts@trYMinF               = -0.15  ; Mínimo valor do eixo y.
rts@tmYLMinorOn           = False  ; Desliga os traços secundários do
                                   ; eixo y.
rts@tmXBMode              = "Explicit"   ; Formata o eixo x do meu jeito.
rts@tmXBValues            = ispan(yStrt,yLast,1) ; Valores do eixo x.
rts@tmXBLabels            = year   ; Rótulos do eixo x inferior.
rts@tmXBLabelAngleF       = -90    ; Inclinação dos rótulos do eixo x.
rts@tmXBLabelJust         = "CenterRight" ; Alinhamento dos rótulos
                                          ; do eixo x.
rts@tmXBLabelStride       = 2  ; Intervalo de valores que serão
                               ; mostrados no eixo x.
```

```
175
176   ; Personalização do painel para as 3 figuras.
177   rtsP              = True   ; Habilita personalização.
178   rtsP@gsnMaximize = True   ; A figura ocupa todo o espaço na página.
179   rtsP@txString     = "SLP: "+season+": "+yStrt+"-"+yLast ; Título da figura
180
181   ; Criação individual das figuras.
182   do n=0,neof-1
183      rts@gsnLeftString  = "EOF "+(n+1) ; Título no lado esquerdo da
184                                        ; figura.
185      ; Título no lado direito da figura.
186      rts@gsnRightString = sprintf("%5.1f", eof@pcvar(n)) +"%"
187      plot(n) = gsn_csm_xy (wks,year,eof_ts(n,:),rts) ; Geração das figuras.
188   end do
189
190   ; Cria o painel com as 3 figuras (3 linhas e 1 coluna).
191   gsn_panel(wks,plot,(/neof,1/),rtsP)
192
193   end
```

O resultado será:

SLP: DJF: 1979-2003
EOF 1
53.3%
EOF 2
16.5%
EOF 3
11.6%
hPa
0.10
0.00
-0.10
1979
1981
1983
1985
1987
1989
1991
1993
1995
1997
1999
2001
2003

22.2.2 EOF sem ponderação

```
; Nome do script: cap22_ex04.ncl

begin

f = addfile("../../dados/sst.anual.1984.2012.nc","r") ; Dado anual de SST

neof = 3 ; Número de EOF's a serem feitas.

yrStrt = 1984   ; Ano inicial.
yrLast = 2012   ; Ano final.
TIME   = f->time                  ; Importação da variável time do
                                  ; arquivo f.
YYYY   = cd_calendar(TIME,-1)/100 ; Tempo no formato AAAAMM (opção -1).
                                  ; A divisão por 100 mostra apenas
                                  ; os anos no formato AAAA.
iYYYY = ind(YYYY.ge.yrStrt.and.YYYY.le.yrLast) ; Seleciona os índices
                                               ; entre 1984 e 2012.

x = short2flt(f->sst(iYYYY,0,:,:)) ; Importa a variável sst do arquivo f.

printVarSummary(x) ; [time | 29] x [lat | 31] x [lon | 86]

X = x(lat|:,lon|:,time|:) ; Reordena a variável de forma que a dimensão
                          ; time seja a dimensão mais a direita.

optEof = True
eof    = eofunc_Wrap(X,neof,optEof)
eof_ts = eofunc_ts_Wrap(X,eof,False)

printVarSummary(eof)    ; [evn | 3] x [lat | 31] x [lon | 86]
printVarSummary(eof_ts) ; [evn | 3] x [time | 29]

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex04a")

gsn_define_colormap(wks,"posneg_1")       ; Define a tabela de cores a
                                          ; ser utilizada.

plot = new(neof,graphic)                  ; Cria um arranjo gráfico para
                                          ; as 3 figuras.

res                              = True ; Habilita os recursos para a
                                        ; figura.
res@gsnDraw                      = False; Não desenha a figura.
res@gsnFrame                     = False; Não avança o plot.
res@gsnAddCyclic                 = False; Quando o dado for regional.
res@mpMinLatF                    = -30. ; Aplica zoom ao mapa.
res@mpMaxLatF                    = 30.  ;
res@mpMaxLonF                    = -70. ;
res@mpMinLonF                    = -180.;
res@gsnMajorLonSpacing           = 10   ; Espaçamento da longitude.
res@gsnMajorLatSpacing           = 10   ; Espaçamento da latitude.
res@gsnContourZeroLineThicknessF = 5.  ; Espessura do contorno zero.
res@cnFillOn                     = True ; Habilita gráfico preenchido.
res@lbLabelBarOn                 = False; Desabilita a legenda individual
res@cnLineLabelsOn               = False; Desabilita os rótulos dos
                                        ; contornos.
symMinMaxPlt(eof,16,False,res)          ; Define o valor mínimo e máximo.
                                        ; 16 representa o número de
```

```
                                        ; contornos desejados.
                                        ; False quer dizer que os
                                        ; valores estão dentro
                                        ; do valor máx/min. Útil para
                                        ; criar escalas de valores
                                        ; para a barra de corres.

; Recursos para a criação do painel de figuras.
resP                      = True        ; Habilita os recursos para o
                                        ; painel.
resP@gsnMaximize          = True        ; A figura ocupa toda a página.
resP@gsnPanelLabelBar     = True        ; Barra de cores comum para
                                        ; todas as figuras.
resP@lbLabelAutoStride    = True        ; Intervalo dos valores na
                                        ; escala de cores.
resP@txString             = "SST: "+yrStrt+"-"+yrLast ; Título da figura.

do n=0,neof-1
   res@gsnLeftString  = "EOF "+(n+1) ; Título do lado esquerdo da figura.
   ; Título do lado direito da figura.
   res@gsnRightString = sprintf("%5.1f", eof@pcvar(n)) +"%"
   plot(n) = gsn_csm_contour_map_ce(wks,eof(n,:,:),res) ; Cria o plot.
end do

gsn_panel(wks,plot,(/neof,1/),resP) ; Gera o painel com as 3 figuras.

; Plot da série temporal da EOF.

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex04b")

eof_ts@long_name = "Amplitude" ; Adiciona o atributo lon_name a variável.

rts                        = True   ; Habilita a personalização da série
                                    ; temporal.
rts@gsnDraw                = False  ; Não desenha a figura.
rts@gsnFrame               = False  ; Não avança o frame.
rts@vpHeightF              = 0.40   ; Altura da figura.
rts@vpWidthF               = 0.85   ; Largura da figura.
rts@vpXF                   = 0.10   ; Posição na página onde
rts@vpYF                   = 0.75   ; ficará a figura.
rts@gsnYRefLine            = 0.0    ; Linha de referência em zero.
rts@gsnAboveYRefLineColor = "red"   ; Cor "red" acima da linha de
                                    ; referência.
rts@gsnBelowYRefLineColor = "blue"  ; Cor "blue" abaixo da linha de
                                    ; referência.
rts@trYMaxF                = 30.    ; Máximo valor do eixo y.
rts@trYMinF                = -30.   ; Mínimo valor do eixo y.
rts@tmYLMinorOn            = False  ; Desliga os traços secundários do
                                    ; eixo y.
rts@tmXBMode               = "Explicit"  ; Formata o eixo x do meu jeito.
rts@tmXBValues             = ispan(yrStrt,yrLast,1) ; Valores do eixo x.
rts@tmXBLabels             = ispan(yrStrt,yrLast,1) ; Rótulos do eixo x
                                                    ; inferior.
rts@tmXBLabelAngleF        = -90  ; Inclinação dos rótulos do eixo x.
rts@tmXBLabelJust          = "CenterRight" ; Alinhamento dos rótulos
                                           ; do eixo x.
rts@tmXBLabelStride        = 2 ; Intervalo de valores que serão mostrados
                               ; no eixo x.
; panel plot only resources
rtsP                      = True ; Habilita personalização do painel.
rtsP@gsnMaximize          = True ; A figura ocupa todo o espaço na página.
rtsP@txString             = "SST: "+yrStrt+"-"+yrLast ; Título da figura.

```

```
do n=0,neof-1
   ; Título do lado esquerdo da figura.
   rts@gsnLeftString  = "EOF "+(n+1)
   ; Título do lado direito da figura.
   rts@gsnRightString = sprintf("%5.1f", eof@pcvar(n)) +"%"
   plot(n) = gsn_csm_xy (wks,ispan(yrStrt,yrLast,1),eof_ts(n,:),rts)
end do

gsn_panel(wks,plot,(/neof,1/),rtsP) ; Cria o painel com as 3 figuras.

end
```

O resultado será:

SST: 1984-2012
EOF 1
49.1%
EOF 2
17.9%
EOF 3
9.6%
Amplitude
30
20
10
0
-10
-20
-30
1984
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004
2006
2008
2010
2012

22.3 Lanczos Filter Weights

22.3.1 Filtragem em uma dimensão (tempo)

A função filwgts_lanczos calcula os pesos dos filtros unidimensionais.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/filter.shtml

Sintaxe: wgt = filwgts_lanczos(nwt,ihp,fca,fcb,nsigma)

Para utilizar essa função são necessários 5 parâmetros:

1. **nwt:** Uma escalar indicando o total de pesos a serem utilizados (deve ser um número ímpar; nwt$\geq$3. Quanto maior o número de pesos, melhor será o filtro, porém haverá uma perda maior de dados no início e no fim da série.

2. **ihp:** Um escalar indicando o tipo de filtro: 0 = passa-baixa, 1 = passa-alta e 2 = passa-banda.

3. **fca:** Um escalar indicando a freqüência de corte do filtro passa-alta ou passa-baixa (0.0<fca<0.5).

4. **fcb.:** Um escalar usado apenas quando o filtro passa-banda é desejado. É a segunda frequência de corte (fca<fcb<0.5). Caso o filtro seja passa-baixa ou passa-alta o valor "-999." é utilizado.

5. **nsigma:** Um escalar indicando a potência do factor sigma (nsigma$\geq$0). Nota: nsigma = 1.0 é o mais utilizado.

Os pesos são utilizados como parâmetros de entrada para a função wgt_runave que serve para suavizar a série. Normalmente, essa função é aplicada na dimensão tempo (time) cujo índice é 0.

A função de resposta, frequência e amplitude são retornados como atributos dos pesos devolvidos (wt). Especificamente os atributos wt@freq e wt@resp são matrizes unidimensionais de comprimento (2$\times$nwt+3) e são do mesmo tipo de wt. Esses atriubutos podem ser representados graficamente.

Exemplo1: Aplicação do filtro passa-baixa na TSM da região do NINO3.4 (5°S-5°N e 170°W-120°W). A frequência de corte (fca=1/24=0.042) é de 2 anos (2x12=24 meses).

```
; Nome do script: cap22_ex05.ncl

begin

; Anomalia mensal de SST na região do NINO3.4.

f = addfile ("../../dados/sst.anom.nino3.4.nc" , "r")

vNames = getfilevarnames(f) ; Obtém a lista com os nomes das variáveis
                            ; do arquivo f.

print (vNames) ; (0)    lon
               ; (1)    lat
               ; (2)    zlev
               ; (3)    time
               ; (4)    sst

time   = f->$vNames(3)$       ; Importação da variável "time" do arquivo.
YYYYMM = cd_calendar(time,-1) ; Converte uma data do calendário
                              ; juliano/gregoriano
                              ; para o formato AAAAMM (opção -1)
                              ; da variável time.
anoi   = 196601               ; Data inicial no formato AAAAMM.
anof   = 201512               ; Data final no formato AAAAMM.
istr   = ind(YYYYMM.eq.anoi ) ; Índice que será utilizado
iend   = ind(YYYYMM.eq.anof ) ; para selecionar o período de interesse.

; Importação da variável sst do arquivo f.
; short sst ( time, zlev, lat, lon )
;             600   1     1    1
; É uma série temporal de anomalia de sst. Por isso, fixou-se o
; zlev, lat e lon.
ssta = short2flt(f->$vNames(4)$(istr:iend,0,0,0))

ntim = dimsizes(ssta) ; Número de tempos da variável.

; O uso da função filwgts_lanczos retorna os valores da frequência
; (freq) e amplitude (resp) do filtro. Eles podem ser acessados via
; símbolo "@". Exemplo: freq = wgtq@fre ou amp = wgtq@resp.

ihp   = 0      ; 0 = filtro passa-baixa, 1 = passa-alta e 2 = passa-banda
sigma = 1.0    ; Fator sigma de Lanczos.
nWgt  = 49     ; Perda de 24 meses no início e fim da série.
fca   = 1./24. ; 2 anos => 1.0/24.0 = 0.042 é o valor do fca.
wgtq  = filwgts_lanczos(nWgt,ihp,fca,-999.,sigma)

printVarSummary(wgtq)  ; Number Of Attributes: 2
                       ; resp : <ARRAY of 101 elements>
                       ; freq : <ARRAY of 101 elements>
resposta   = wgtq@resp ; Será utilizada para gerar a figura
frequencia = wgtq@freq ; Frequência (x) versus Resposta de frequência (y)

wgtqs = wgt_runave(ssta,wgtq,0)  ; 2 anos. wgtqs => suavização da série
                                 ; pela função wgt_runave que cálcula a
                                 ; média móvel.

plot = new(3,"graphic") ; Cria uma variável do tipo graphic para
                        ; armazenar as 3 figuras (painel de figuras).

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex05")
```

```
61
62  res1                  = True ; Personalização da figura.
63  res1@gsnDraw          = False; Não desenha.
64  res1@gsnFrame         = False; Não avança o frame.
65  res1@vpHeightF        = 0.4  ; Altura da figura.
66  res1@vpWidthF         = 0.8  ; Largura da figura.
67  res1@trYMinF          = -3.0 ; Mínimo valor do eixo y.
68  res1@trYMaxF          =  3.0 ; Máximo valor do eixo y.
69  res1@trXMinF          = istr ; Mínimo valor do eixo x.
70  res1@trXMaxF          = iend ; Máximo valor do eixo x.
71  res1@vpXF             = 0.1  ; Posição na página onde será criada a figura
72  res1@gsnYRefLine      = 0.0  ; Cria linha de referência no valor 0.
73  res1@gsnCenterString = "Anomalia de SST no NINO3.4" ; Título da figura.
74  res1@tmXBMode         = "Explicit"   ; Formata o eixo x do meu jeito.
75  res1@tmXBValues       = ispan(istr,iend,12)      ; Valores do eixo x.
76  res1@tmXBLabels       = YYYYMM(istr:iend:12)/100 ; Rótulos que vão aparece
77                                                   ; no eixo x.
78  res1@tmXBLabelAngleF = 90                        ; Rotaciona os rótulos
79                                                   ; do eixo x.
80  res1@tmXBLabelJust    = "CenterRight"            ; Posicionamento dos
81                                                   ; rótulos do eixo x.
82  res1@tiYAxisString    = "Anomalia de SST (~S~o~N~C)"
83  res1@tmXBLabelStride = 3  ; Mostra os rótulos do eixo x a cada 3 valores.
84  res1@gsnAboveYRefLineColor = "red"  ; Cor vermelho para os valores
85                                      ; acima de 0.
86  res1@gsnBelowYRefLineColor = "blue" ; Cor azul para os valores abaixo
87                                      ; de 0.
88  res1@tmXTOn                 = False  ; Desabilita o minortick do eixo x
89                                      ; superior (XT).
90
91  ; Cria a primeira figura (superior).
92  plot(0) = gsn_csm_xy(wks,ispan(istr,iend,1),ssta,res1)
93
94  res1@xyMonoDashPattern = True ; Defino o tipo de linha. O padrão é sólido
95  res1@xyLineThicknessF  = 2    ; Espessura da linha.
96  res1@gsnCenterString   = "Filtro passa-baixa: 2 anos" ; Título da gráfico
97
98  ; Cria a figura do meio.
99  plot(1) = gsn_csm_xy(wks,ispan(istr,iend,1),wgtqs,res1)
100
101  res2                   = True  ; Habilita personalização da figura.
102  res2@trXMinF           = 0.0   ; Mínimo valor do eixo x.
103  res2@trXMaxF           = 0.12  ; Máximo valor do eixo x.
104  res2@trYMinF           = -0.2  ; Mínimo valor do eixo y.
105  res2@trYMaxF           = 1.2   ; Máximo valor do eixo y.
106  res2@vpHeightF         = 0.4   ; Altura da figura.
107  res2@vpWidthF          = 0.8   ; Largura da figura.
108  res2@gsnDraw           = False ; Não desenha.
109  res2@gsnFrame          = False ; Não avança o frame.
110  res2@tiXAxisString     = "Frequ"+ecirc+"ncia" ; Título do eixo x.
111  res2@tiYAxisString     = "Resposta de frequ"+ecirc+"ncia"
112  res2@tmXBMode          = "Manual"     ; Formata o eixo x do meu jeito.
113  res2@tmXBTickStartF    = res2@trXMinF ; Mínimo valor do eixo x.
114  res2@tmXBTickSpacingF = 0.01          ; Incremento do eixo x.
115  res2@tmXBTickEndF      = res2@trXMaxF ; Máximo valor do eixo x.
116  res2@tmXBMinorOn       = False        ; Desabilita os traços secundários d
117                                        ; eixo x.
118  res2@tmYLMode          = "Manual"     ; Formata o eixo y do meu jeito.
119  res2@tmYLTickStartF    = res2@trYMinF ; Mínimo valor do eixo y.
120  res2@tmYLTickSpacingF = 0.2           ; Incremento do eixo y.
121  res2@tmYLTickEndF      = res2@trYMaxF ; Máximo valor do eixo y.
122  res2@tmYLMinorOn       = False        ; Desabilita os traços secundários
123                                        ; do eixo y.
```

```
res2@xyLineThicknessF = 4            ; Espessura da linha.
res2@xyLineColors     = "red"        ; Cor de cada linha.
res2@tmXBFormat       = "0@;*.2f"    ; Formata os números do eixo x
                                     ; inferior (XB).

; Frequência (x) versus Resposta de frequência (y).
plot(2) = gsn_csm_xy(wks,frequencia,resposta,res2)

; Desenha as linhas horizontal e vertical (cor azul no gráfico).
lnres                   = True   ; Habilita a personalização.
lnres@gsLineColor       = "blue" ; Cor da linha da caixa.
lnres@gsLineThicknessF  = 2.0    ; Espessura da linha da caixa.
lnres@gsLineDashPattern = 2      ; Estilo de linha.

x = (/0.0,fca,fca,res2@trXMaxF/); Coordenadas onde desenhar as linhas
y = (/1.0,1.0,0.0,0.0/)       ; horizontal e vertical no gráfico inferior

caixa = gsn_add_polyline(wks,plot(2),x,y,lnres) ; Desenha as linhas.

; Cria o painel com as 3 figuras.
resP               = True
resP@gsnMaximize   = True

gsn_panel(wks,plot,(/3,1/),resP) ; Painel com 3 linhas e 1 coluna.

end
```

O resultado será:

Anomalia de SST no NINO3.4
Anomalia de SST (°C)
3.0
2.0
1.0
0.0
-1.0
-2.0
-3.0
1966
1969
1972
1975
1978
1981
1984
1987
1990
1993
1996
1999
2002
2005
2008
2011
2014
Filtro passa-baixa: 2 anos
Anomalia de SST (°C)
Resposta de frequência
1.2
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
0.00 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.10 0.11 0.12
Frequência

Exemplo2: Aplicando o filtro passa-banda nas frequências de corte (fca=1/72=0.014 e fcb=1/36=0.028) de 6 anos (6x12=72 meses) e 3 anos (3x12=36 meses).

```
; Nome do script: cap22_ex06.ncl

begin

; Anomalia mensal de SST na região do NINO3.4.

f = addfile ("../../dados/sst.anom.nino3.4.nc" , "r")

vNames = getfilevarnames(f) ; Obtém a lista com os nomes das variáveis
                            ; do arquivo f.

print (vNames) ; (0)    lon
               ; (1)    lat
               ; (2)    zlev
               ; (3)    time
               ; (4)    sst

time   = f->$vNames(3)$       ; Importação da variável "time" do arquivo.
YYYYMM = cd_calendar(time,-1) ; Converte uma data do calendário
                              ; juliano/gregoriano
                              ; para o formato AAAAMM (opção -1) da
                              ; variável time.
anoi   = 196601               ; Data inicial no formato AAAAMM.
anof   = 201512               ; Data final no formato AAAAMM.
istr   = ind(YYYYMM.eq.anoi ) ; Índice que será utilizado
iend   = ind(YYYYMM.eq.anof ) ; para selecionar o período de interesse.

; Importação da variável sst do arquivo f.
; short sst ( time, zlev, lat, lon )
;              600   1     1    1
; É uma série temporal de anomalia de sst. Por isso, fixou-se o
; zlev, lat e lon.
ssta = short2flt(f->$vNames(4)$(istr:iend,0,0,0))

ntim = dimsizes(ssta) ; Número de tempos da variável.

; O uso da função filwgts_lanczos retorna os valores da
; frequência (freq) e amplitude (resp) do filtro.
; Exemplo: freq = wgtq@fre ou amp = wgtq@resp.

ihp   = 2      ; 0 = filtro passa-baixa, 1 = passa-alta e
               ; 2 = passa-banda.
sigma = 1.0    ; Fator sigma de Lanczos.
nWgt  = 201    ; Perda de 100 meses no início e fim da série.
fca   = 1./72. ; 6 anos (6x12meses=72) => 1.0/72.0 = 0.014 é o
               ; valor do fca.
fcb   = 1./36. ; 3 anos (3x12meses=36) => 1.0/36.0 = 0.028 é o
               ; valor do fcb.

wgt   = filwgts_lanczos(nWgt,ihp,fca,fcb,sigma)

printVarSummary(wgt)  ; Number Of Attributes: 2
                      ; resp :  <ARRAY of 405 elements>
                      ; freq :  <ARRAY of 405 elements>
resposta   = wgt@resp ; Será utilizada para gerar a figura.
frequencia = wgt@freq ; Frequência (x) versus Resposta de frequência (y).

wgts = wgt_runave(ssta,wgt,0)  ; wgts => suavização da série pela função
                               ; wgt_runave que cálcula a média móvel.

```

```
plot = new(3,"graphic")

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex06")

res1                  = True  ; Personalização da figura.
res1@gsnDraw          = False ; Não desenha.
res1@gsnFrame         = False ; Não avança o frame.
res1@vpHeightF        = 0.4   ; Altura da figura.
res1@vpWidthF         = 0.8   ; Largura da figura.
res1@trYMinF          = -3.0  ; Mínimo valor do eixo y.
res1@trYMaxF          =  3.0  ; Máximo valor do eixo y.
res1@trXMinF          = istr  ; Mínimo valor do eixo x.
res1@trXMaxF          = iend  ; Máximo valor do eixo x.
res1@vpXF             = 0.1   ; Posição na página onde será
                              ; criada a figura.
res1@gsnYRefLine      = 0.0   ; Cria linha de referência no valor 0.
res1@gsnCenterString = "Anomalia de SST no NINO3.4" ; Título da figura.
res1@tmXBMode         = "Explicit" ; Formata o eixo x do meu jeito.
res1@tmXBValues       = ispan(istr,iend,12)      ; Valores do eixo x.
res1@tmXBLabels       = YYYYMM(istr:iend:12)/100 ; Rótulos que vão
                                                 ; aparecer no eixo x.
res1@tmXBLabelAngleF = 90 ; Rotaciona os rótulos do eixo x.
res1@tmXBLabelJust   = "CenterRight"; Posicionamento dos rótulos do eixo x
res1@tiYAxisString   = "Anomalia de SST (~S~o~N~C)"
res1@tmXBLabelStride = 3  ; Mostra os rótulos do eixo x a cada 3 valores.
res1@gsnAboveYRefLineColor = "red"
res1@gsnBelowYRefLineColor = "blue"
res1@tmXTOn                = False

; Cria a primeira figura (superior).
plot(0) = gsn_csm_xy(wks,ispan(istr,iend,1),ssta,res1)

res1@xyMonoDashPattern = True ; Defino o tipo de linha. O padrão é sólido
res1@xyLineThicknessF  = 2    ; Espessura da linha.
res1@gsnCenterString   = "Filtro passa-baixa: 2 anos" ; Título da gráfico

; Cria a figura do meio.
plot(1) = gsn_csm_xy(wks,ispan(istr,iend,1),wgts,res1)

res2                   = True ; Habilita personalização da figura.
res2@trXMinF           = 0.0  ; Mínimo valor do eixo x.
res2@trXMaxF           = 0.08 ; Máximo valor do eixo x.
res2@trYMinF           = -0.2 ; Mínimo valor do eixo y.
res2@trYMaxF           = 1.2  ; Máximo valor do eixo y.
res2@vpHeightF         = 0.4  ; Altura da figura.
res2@vpWidthF          = 0.8  ; Largura da figura.
res2@gsnDraw           = False ; Não desenha.
res2@gsnFrame          = False ; Não avança o frame.
res2@tiXAxisString     = "Frequ"+ecirc+"ncia" ; Título do eixo x.
res2@tiYAxisString     = "Resposta de frequ"+ecirc+"ncia"
res2@tmXBMode          = "Manual"       ; Formata o eixo x do meu jeito.
res2@tmXBTickStartF    = res2@trXMinF   ; Mínimo valor do eixo x.
res2@tmXBTickSpacingF  = 0.01           ; Incremento do eixo x.
res2@tmXBTickEndF      = res2@trXMaxF   ; Máximo valor do eixo x.
res2@tmXBMinorOn       = False          ; Desabilita os traços secundários
                                        ; do eixo x.
res2@tmYLMode          = "Manual"       ; Formata o eixo y do meu jeito.
res2@tmYLTickStartF    = res2@trYMinF   ; Mínimo valor do eixo y.
res2@tmYLTickSpacingF  = 0.2            ; Incremento do eixo y.
res2@tmYLTickEndF      = res2@trYMaxF   ; Máximo valor do eixo y.
res2@tmYLMinorOn       = False; Desabilita os traços secundários do eixo y
res2@xyLineThicknessF  = 4              ; Espessura da linha.
res2@xyLineColors      = "red"          ; Cor de cada linha.
```

```
res2@tmXBFormat       = "0@;*.1f"     ; Formata os números do eixo x
                                      ; inferior (XB).

; Frequência (x) versus Resposta de frequência (y).
plot(2) = gsn_csm_xy(wks,frequencia,resposta,res2)

; Desenha as linhas horizontal e vertical (cor azul no gráfico).
lnres                   = True   ; Habilita a personalização.
lnres@gsLineColor       = "blue" ; Cor da linha da caixa.
lnres@gsLineThicknessF  = 2.0    ; Espessura da linha da caixa.
lnres@gsLineDashPattern = 2      ; Estilo de linha.

; Coordenadas onde desenhar as linhas horizontal e vertical
; no gráfico inferior.
x = (/0.0,fca,fca,fcb,fcb,res2@trXMaxF/)
y = (/0.0,0,1.0,1.0,0,0/)

caixa = gsn_add_polyline(wks,plot(2),x,y,lnres) ; Desenha as linhas.

; Cria o painel com as 3 figuras.
resP               = True
resP@gsnMaximize   = True

gsn_panel(wks,plot,(/3,1/),resP) ; Painel com 3 linhas e 1 coluna.

end
```

O resultado será:

22.4 Função Distribuição de Probabilidade (PDF)

Gera uma distribuição densidade de probabilidade univariada (Probability Distribution Function, PDF). A unidade da pdf é dada em porcentagem (%)

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/pdf.shtml

Sintaxe: x = pdf(x,nbin,opt)

Para utilizar essa função são necessários 3 parâmetros:

1. **x:** Um arranjo de qualquer dimensionalidade.
2. **nbin:** Número de caixas a serem utilizadas (bin). O valor 0 resultará em 25 bins, que é o valor padrão. Caso seja necessário um número diferente de bins, este valor deve ser maior do que 2.

3. **opt:** Os atributos podem estar associados com essa variável e eles irão alterar o comportamento padrão da função pdfx. Normalmente definido como False. Para mais detalhes, acessa o site sobre a função pdfx.

Os valores retornados com a função pdfx são:

- **nbins:** Número de bins utilizados.
- **bin_spacing:** Espaçamento dos bins.
- **bin_bound_min:** O valor mínimo do bin.
- **bin_bound_max:** O valor máximo do bin.
- **bin_center:** Uma matriz unidimensional do tamanho dos bins contendo o ponto central de cada bin. Para gerar a figura da PDF, utiliza-se esse valor no eixo x.
- **bin_bounds:** Uma matriz unidimensional de tamanho (nbins+1), contendo os limites de cada bin.

Para acessar esses valores utiliza-se o símbolo "@". Por exemplo:

ap = pdfx(appt,0,False) ; Calculo da PDF.

x = ap@bin_center ; Atribui os valores de ap@bin_center a variável x.

Exemplo1: Calculando a PDF utilizando dados de anomalia de precipitação sobre a América do Sul (AS). O dado refere-se a média da área (série temporal).

```
; Nome do script: cap22_ex07.ncl

begin

f = addfile("../../dados/anom.SA.GPCP.prec.1979.2014.nc","r")

appt = f->precip(:,0,0) ; float precip(time, lat, lon)
                        ;               432   1    1

ap  = pdfx(appt,0,False) ; Cálculo da PDF.

printVarSummary(ap)

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex07")

res                     = True      ; Habilita personalização do gráfico.
res@xyLineThicknessF    = 2         ; Espessura da linha.
res@tiYAxisString       = "PDF (%)" ; Título do eixo y.
res@gsnCenterString     = "Anomalia de precipita"+cedil+atilde+ \
                          "o sobre a AS - 1979/2014"
res@gsnXYBarChart       = True       ; Habilita gráfico de barras.
res@gsnXYBarChartColors = "gray"     ; Cor da barra.
res@trYMaxF             = 16.0       ; Máximo valor do eixo y.
res@trYMinF             = 0.0        ; Mínimo valor do eixo y.
res@trXMaxF             = 0.8        ; Máximo valor do eixo x.
res@trXMinF             = -0.8       ; Mínimo valor do eixo x.
res@tmYLMode            = "Manual"   ; Personalização do eixo y do meu
                                     ; jeito.
res@tmYLTickStartF      = res@trYMinF
res@tmYLTickSpacingF    = 2
res@tmYLTickEndF        = res@trYMaxF
res@tmXBMode            = "Manual"   ; Personalização do eixo x do meu
                                     ; jeito.
res@tmXBTickStartF      = res@trXMinF
res@tmXBTickSpacingF    = 0.2
res@tmXBTickEndF        = res@trXMaxF
res@tmYLMinorOn         = False      ; Desabilita o minortick do eixo y
                                     ; esquerdo (YL).
res@tmXBMinorOn         = False      ; Desabilita o minortick do eixo x
                                     ; inferior (XB).
res@tmXBFormat          = "0@;*.2f"  ; Formata os números do eixo x
                                     ; inferior (XB).

plot = gsn_csm_xy(wks,ap@bin_center,ap,res)

end
```

O resultado será:

Exemplo2: Mesmo cálculo realizado no Exemplo1 com a diferença que o parâmetro opt da função pdfx foi alterado para True.

```
; Nome do script: cap22_ex08.ncl

begin

f = addfile("../../dados/anom.SA.GPCP.prec.1979.2014.nc","r")

appt = f->precip(:,0,0) ; float precip(time, lat, lon)
                        ;                432   1    1

opt         = True   ; Habilita a personalização da função pdfx.
opt@bin_min = -0.1   ; Mínimo valor do bin.
opt@bin_max =  0.1   ; Máximo valor do bin.

ap  = pdfx(appt,50,True) ; Cálculo da PDF. Serão mostrados 50 bins.

printVarSummary(ap)

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex08")

res                     = True       ; Habilita personalização do gráfico.
res@xyLineThicknessF    = 2          ; Espessura da linha.
res@tiYAxisString       = "PDF (%)"  ; Título do eixo y.
res@gsnCenterString     = "Anomalia de precipita"+cedil+atilde+ \
                          "o sobre a AS - 1979/2014"
res@gsnXYBarChart       = True       ; Habilita gráfico de barras.
res@gsnXYBarChartColors = "gray"     ; Cor da barra.
res@trYMaxF             = 10.0       ; Máximo valor do eixo y.
res@trYMinF             = 0.0        ; Mínimo valor do eixo y.
res@trXMaxF             = 0.8        ; Máximo valor do eixo x.
res@trXMinF             = -0.8       ; Mínimo valor do eixo x.
res@tmYLMode            = "Manual"   ; Personalização do eixo y do meu
                                     ; jeito.
res@tmYLTickStartF      = res@trYMinF
res@tmYLTickSpacingF    = 2
```

```
res@tmYLTickEndF          = res@trYMaxF
res@tmXBMode              = "Manual"    ; Personalização do eixo x do meu
                                        ; jeito.
res@tmXBTickStartF        = res@trXMinF
res@tmXBTickSpacingF      = 0.2
res@tmXBTickEndF          = res@trXMaxF
res@tmYLMinorOn           = False       ; Desabilita o minortick do eixo y
                                        ; esquerdo (YL).
res@tmXBMinorOn           = False       ; Desabilita o minortick do eixo x
                                        ; inferior (XB).
res@tmXBFormat            = "0@;*.2f"   ; Formata os números do eixo x
                                        ; inferior (XB).

plot = gsn_csm_xy(wks,ap@bin_center,ap,res)

end
```

O resultado será:

Exemplo3: Mesmo cálculo realizado no Exemplo1 com a diferença que o parâmetro opt da função pdfx foi alterado para True e valores razoáveis serão utilizados para gerar o bin.

```
; Nome do script: cap22_ex09.ncl

begin

f = addfile("../../dados/anom.SA.GPCP.prec.1979.2014.nc","r")

appt = f->precip(:,0,0) ; float precip(time, lat, lon)
                        ;               432   1    1

opt          = True  ; Habilita a personalização da função pdfx.
opt@bin_nice = True  ; Valores razoáveis do limite bin e espaçamento
                     ; serão calculados.

ap  = pdfx(appt,35,True) ; Cálculo da PDF. Serão mostrados 50 bins.

printVarSummary(ap)

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex09")

res                     = True      ; Habilita personalização do gráfico.
res@xyLineThicknessF    = 2         ; Espessura da linha.
res@tiYAxisString       = "PDF (%)" ; Título do eixo y.
res@gsnCenterString     = "Anomalia de precipita"+cedil+atilde+ \
                          "o sobre a AS - 1979/2014"
res@gsnXYBarChart       = True       ; Habilita gráfico de barras.
res@gsnXYBarChartColors = "gray"     ; Cor da barra.
res@trYMaxF             = 12.0       ; Máximo valor do eixo y.
res@trYMinF             = 0.0        ; Mínimo valor do eixo y.
res@trXMaxF             = 0.8        ; Máximo valor do eixo x.
res@trXMinF             = -0.8       ; Mínimo valor do eixo x.
res@tmYLMode            = "Manual"   ; Personalização do eixo y do meu
                                     ; jeito.
res@tmYLTickStartF      = res@trYMinF
res@tmYLTickSpacingF    = 2
res@tmYLTickEndF        = res@trYMaxF
res@tmXBMode            = "Manual"   ; Personalização do eixo x do meu
                                     ; jeito.
res@tmXBTickStartF      = res@trXMinF
res@tmXBTickSpacingF    = 0.2
res@tmXBTickEndF        = res@trXMaxF
res@tmYLMinorOn         = False      ; Desabilita o minortick do eixo y
                                     ; esquerdo (YL).
res@tmXBMinorOn         = False      ; Desabilita o minortick do eixo x
                                     ; inferior (XB).
res@tmXBFormat          = "0@;*.2f"  ; Formata os números do eixo x
                                     ; inferior (XB).

plot = gsn_csm_xy(wks,ap@bin_center,ap,res)

end
```

O resultado será:

22.5 Análise Espectral (Spectral Analysis)

Calcula o espectro de uma série.

Informações adicionais podem ser encontradas em:

http://www.ncl.ucar.edu/Applications/spec.shtml

Sintaxe: x = specx_anal(x,iopt,jave,pct)

São necessários 4 parâmetros:

1. **x:** Um arranjo unidimensional. Valores ausentes não são permitidos.
2. **iopt:** iopt = 0, remove a média da série e iopt=1 remove a média e a tendência linear.
3. **jave:** Um escalar representando a suavização a ser realizada na estimativa do periodograma. Este valor deve ser um número ímpar ($\geq$3). Caso contrário, a rotina irá forçar para o próximo número ímpar.
4. **pct:** Representa o percentual da série a ser afunilada (0.0$\leq$pct$\leq$1.0). Se pct=0.0, nenhum afunilamento será feito. Se pct=1.0, a série inteira é afetada. Um valor de 0.10 é o mais comum. Esse afunilamento deve sempre ser feito.

O resultado fornece alguns atributos:

- **spcx**: Um arranjo de uma dimensão de comprimento N/2.

- **frq**: Um arranjo de uma dimensão de comprimento N/2 que representa a frequência (ciclos/tempo).
- **bw**: Um escalar que representa a banda espectral.
- **xavei**: Um escalar que representa a média da série de entrada.
- **xvari**: Um escalar que representa a variância da série de entrada.
- **xvaro**: Um escalar que representa a variância da série após removida a tendência.
- **xlag1**: Um escalar que representa a auto-correlação da série de entrada após removida a tendência.
- **xslope**: Um escalar que representa a inclinação dos mínimos quadrados por intervalo de tempo da tendência linear (se iopt=1) da série x.

A escolha de jave altera a largura da banda espectral, o carácter do espectro e o número de graus de liberdade. Menor valor de jave produz maior resolução no domínio da frequência (menor largura de banda). O espectro vai verificar as irregularidades e o número de graus de liberdade será menor.

A função specx_ci calcula o espectro teórico de Markov e os intervalos de confiança inferior e superior.

Sintaxe do espectro de Markov: splt = specx_ci(sdof,lowval,highval)

São necessários 3 parâmetros:

1. **sdof:** O arranjo de graus de liberdade retornado da função specx_anal or specxy_anal.
2. **lowval:** O limite de confiança inferior (0.0<lowval<1.0). Um valor típico utilizado é de 0.05.
3. **highval:** O limite de confiança superior (0.0<HiVal<1.0). Um valor típico utilizado é de 0.95.

Uma matriz bidimensional com dimensões 4xN, em que N é o tamanho de sdof@spcx. O resultado terá 4 curvas:

1. **split(0,:):** É o espectro de entrada.
2. **split(1,:):** É o "ruído vermelho" de Markov.
3. **split(2:):** Intervalo de confiança inferior.
4. **split(3:):** Intervalo de confiança superior.

Exemplo1: Cálculo do espectro utilizando a anomalia de precipitação.

```
; Nome do script: cap22_ex10.ncl

begin

; Anomalia mensal de precipitação sobre a América do Sul (1979 a 2014).
in   = addfile("../../dados/anom.SA.GPCP.prec.1979.2014.nc","r")
appt = in->precip(:,0,0)

; detrending opt: 0=>remove a média e 1=>remove a média e a tendência.
d = 0

; Suavização do periodograma. Deve ser ímpar e valor mínimo de 3.
sm = 7

; Porcentagem de suaviação: (0.0 <= pct <= 1.0) sendo o valor 0.10 o
; mais comum.
pct = 0.10

spec = specx_anal(appt,d,sm,pct) ; Calcula o espectro.

printVarSummary(spec)

; Resultado do printVarSummary(spec)

;Number Of Attributes: 8
;  xslope :      0
;  xlag1 :      0.2093593
;  xvaro :      0.04039465
;  xvari :      0.04039465
;  xavei :      -4.850638e-11
;  bw : 0.01436011
;  frq :        <ARRAY of 216 elements>
;  spcx :       <ARRAY of 216 elements>

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex10")

res                = True
res@tiMainString   = "Precipita"+cedil+atilde+"o - 1979 a 2014"
res@tiXAxisString  = "Frequ"+ecirc+"ncia (ciclos/m"+ecirc+"s)"
res@tiYAxisString  = "Vari"+acirc+"ncia"

plot=gsn_csm_xy(wks,spec@frq,spec@spcx,res)

end
```

O resultado será:

Exemplo2: Mesmo do Exemplo1 só que é calculado o ruído vermelho (linha verde) e os intervalos de confiança (linhas tracejadas vermelha e azul).

```
; Nome do script: cap22_ex11.ncl

begin

; Anomalia mensal de precipitação sobre a América do Sul (1979 a 2014).
in   = addfile("../../dados/anom.SA.GPCP.prec.1979.2014.nc","r")
appt = in->precip(:,0,0)

; detrending opt: 0=>remove a média e 1=>remove a média e a tendência.
d = 0

; Suavização do periodograma. Deve ser ímpar e valor mínimo de 3.
sm = 21

; Porcentagem de suaviação: (0.0 <= pct <= 1.0) sendo o valor 0.10 o
; mais comum.
pct = 0.10

sdof = specx_anal(appt,d,sm,pct) ; Calcula o espectro.

printVarSummary(sdof)

; Resultado do printVarSummary(spec)

;Number Of Attributes: 8
;  xslope :      0
;  xlag1 :      0.2093593
;  xvaro :      0.04039465
```

```
;  xvari :      0.04039465
;  xavei :      -4.850638e-11
;  bw : 0.01436011
;  frq :        <ARRAY of 216 elements>
;  spcx :       <ARRAY of 216 elements>

splt = specx_ci(sdof,0.05,0.95) ; Calcula o intervalo de confiança do
                                ; ruído vermelho (red noise).

; Graus de liberdade retornado da função specx_anal.
; Limite inferior do intervalo de confiança (0.0 < lowval < 1.). O valor
; típico é de 0.05.
; Limite superior do intervalo de confiança (0.0 < hival < 1.). O valor
; típico é de 0.95.

printVarSummary(splt)

; O splt resulta em 4 curvas. Essas curvas são utilizadas para gerar o
; gráfico. Nota-se na figura que há 4 curvas na ordem abaixo.

; splt(0,:) - input spectrum
; splt(1,:) - Markov "Red Noise" spectrum
; splt(2,:) - lower confidence bound for Markov
; splt(3,:) - upper confidence bound for Markov

wks = gsn_open_wks("pdf","../../figuras/cap22/cap22_ex11")

res                        = True  ; Habilita personalização do gráfico.
res@tiMainString           = "Precipita"+cedil+atilde+"o - 1979 a 2014"
res@tiXAxisString          = "Frequ"+ecirc+"ncia (ciclos/m"+ecirc+"s)"
res@tiYAxisString          = "Vari"+acirc+"ncia"
res@xyLineThicknesses      = (/2.,1.,1.,1./)  ; Espessura de cada linha.
res@xyDashPatterns         = (/0,0,1,1/)      ; Tipo de linha.
res@xyLineColors           = (/"foreground","green","blue","red"/)
res@pmLegendDisplayMode    = "Always"    ; Habilita a legenda.
res@pmLegendSide           = "Top"       ; Localização da legenda.
res@pmLegendParallelPosF   = 0.68        ; Move a legenda na direção x.
res@pmLegendOrthogonalPosF = -0.38       ; Move a legenda na direção y.
res@pmLegendWidthF         = 0.1         ; Largura da legenda.
res@pmLegendHeightF        = 0.15        ; Altura da legenda.
res@lgPerimOn              = False       ; Desabilita a caixa em torno
                                         ; da legenda.
res@lgLabelFontHeightF     = 0.024       ; Tamanho da fonte da legenda.
res@lgItemOrder            = (/3,2,1,0/) ; Reordena os rótulos da legenda
res@trYMaxF                = 0.22        ; Máximo valor do eixo y.
res@trYMinF                = 0.0         ; Mínimo valor do eixo y.
res@trXMaxF                = 0.5         ; Máximo valor do eixo x.
res@trXMinF                = 0.0         ; Mínimo valor do eixo x.
res@tmYLMode               = "Manual"    ; Personalização do eixo y do
                                         ; meu jeito.
res@tmYLTickStartF         = res@trYMinF
res@tmYLTickSpacingF       = 0.02
res@tmYLTickEndF           = res@trYMaxF
res@tmXBMode               = "Manual"    ; Personalização do eixo x do
                                         ;meu jeito.
res@tmXBTickStartF         = res@trXMinF
res@tmXBTickSpacingF       = 0.1
res@tmXBTickEndF           = res@trXMaxF
res@tmYLMinorOn            = False      ; Desabilita o minortick do eixo
                                        ; y esquerdo (YL).
```

```
res@tmXBMinorOn            = False      ; Desabilita o minortick do eixo
                                        ; x inferior (XB).
res@tmYLFormat             = "0@;*.2f"  ; Formata os números do eixo y
                                        ; esquerdo (YL).
res@xyExplicitLegendLabels = (/"Espectro",                        \
                               "Ru"+iacute+"do vermelho",         \
                               "5% de signific"+acirc+"ncia",     \
                               "95% de signific"+acirc+"ncia"/)

plot=gsn_csm_xy(wks,sdof@frq,splt,res) ; Gera o gráfico com as 4 curvas.

end
```

O resultado será:

23 Links interessantes

- Cursos: http://www.ncl.ucar.edu/Training
- Exemplos: http://www.ncl.ucar.edu/Applications
- Manuais: http://www.ncl.ucar.edu/Document/Manuals
- Novidades: http://www.ncl.ucar.edu/announcements.shtml
- Site do NCL: http://www.ncl.ucar.edu
- Variados: http://www.ncl.ucar.edu/Support
- Webinars: http://www.ncl.ucar.edu/Training/Webinars
- Dicas variadas: http://www.ncl.ucar.edu/Applications/concepts_list.shtml

24 Âpendice

24.1 Âpendice 1 - NCL Reference Cards

NCL quick reference card

NCL version 6.4.0 — Karin Meier-Fleischer, DKRZ
February 27, 2017 — Mary Haley, NCAR

Syntax characters

=	assignment syntax
:=	reassignment operator
;	starts a comment
/;…;/	starts a block comment
@	create or reference an attribute
!	create or reference a named dimension
&	create or reference a coordinate variable
$…$	enclose strings when importing or exporting variables via *addfile*
{…}	subscript arrays using coordinate values
[…]	subscripts variables of type *list*
(/…/)	array constructor
[/…/]	list constructor
:	array syntax delimiter
|	separator for named dimensions
****	continuation character for wrapping long lines
::	separator when calling external codes
->	used to im/export variables from/to supported file formats

Expressions

Algebraic operators

+	Addition, string concatenation
-	Subtraction / Negation
*****	Multiplication
/	Division
%	Modulus (integers only)
>	Greater than
<	Less than
^	Exponentiation
#	Matrix multiplication

Logical operators

.lt.	Less than
.le.	Less than or equal
.eq.	Equal
.ne.	Not equal
.ge.	Greater than or equal
.gt.	Greater than
.and.	AND
.or.	OR
.xor.	Exclusive OR
.not.	NOT

Data types

Numeric

double	64 bit
float	32 bit
long	32 bit or 64 bit; signed +/-
integer	32 bit; signed +/-
short	16 bit; signed +/-
byte	8 bit; signed +/-
complex	NOT supported

Enumeric

int64	64 bit; signed +/-
uint64	64 bit; unsigned
uint	32 bit; unsigned
ulong	32 bit or 64 bit; unsigned
ushort	16 bit; unsigned
ubyte	8 bit; unsigned

Non-numeric

string
character
graphic
file
logical
list

Variables

Assign a variable

```
x = 1                          ; integer
y = 2.6                        ; float
d = 20.d                       ; double
str = "This is a string"       ; string
res = True                     ; logical (True/False)
a = (/1,2,3,4/)                ; integer array
b = (/2,7.0,4./)               ; float array
c = (/1.,2,3.,4.0/) * 1d5      ; double array
d = (/"red","green","blue"/)   ; string array
e = (/True,False,False,True/)  ; logical array
f = (/(/1,2/),(/3,6/),(/4,2/)/) ; 2D array (3 x 2)
```

Arrays

The leftmost dimension (dim) of a multi-dim array varies slowest and the rightmost dim varies fastest (row major).

```
a = (/4,2,1,3/)        ; 4 elements; index 0-3
b = (/0,1,1,0/)        ; 4 elements; index 0-3

c = a + b          →   c = (/4,3,2,3/)
c = a - b          →   c = (/4,1,0,3/)
c = a * b          →   c = (/0,2,1,0/)
c = a/(b+0.1)      →   c = (/40,1.8182, 0.909090,30/)
```

To create a new array

```
n = new(4,integer)        → integer array of size 4
q = new((/2,3,5/),float)  → float array of size 2x3x5
l = new(100,float,1e20)   → float array with
                            _FillValue=1e20
cities = new(20,string)   → string array of size 20
```

Standard subscripting of arrays

The indices used in standard subscripting are integers and the general form of a standard subscript is:

```
m:n:i          ; range m to n in strides of i
```

```
 a = (/1,2,3,4,5,6/)
a1 = a(3)       ; a1 is 4
a2 = a(0:2)     ; a2 contains 1,2,3
a3 = a(0:4:2)   ; a3 contains 1,3,5
a4 = a(1:4:-1)  ; a4 contains 5,4,3,2
a5 = a(:3)      ; a5 contains 1,2,3,4
a6 = a(5:3)     ; a6 contains 6,5,4
a7 = a(::-1)    ; reverse a 6,5,4,3,2,1
```

Named dimensions

The dimensions of an array are numbered from 0 to *n*-1. To attach a name to an array dimension, use the **!** character.

```
varNew!0 = "time"
varNew!1 = "lev"
varNew!2 = "lat"
varNew!3 = "lon"
```

Named subscripting

Named dimensions allow you to reorder and subscript arrays.

```
pres(lat,lon)       ; lat=21, lon=40

pres_new1 = pres(lon|:, lat|:)  ; reorder (reshape)
pres_new2 = pres(lon|19:38, lat|0:9)
            ; define an new array pres_new2(20,10)
            ; with pres_new2(lon,lat)
```

Coordinate variables

A coordinate variable is a one-dimensional variable with the same name as a dimension, which provides coordinate values for that dimension. It must be strictly monotonic (values increasing or decreasing, not mixed).

```
lat_pts       = (/30.,40.,50.,60.,/)    ; size 4
lon_pts       = (/ 0.,15, 30, 45, 60/)  ; size 5
lat_pts@units = "degrees_north"  ; set units attribute
lon_pts@units = "degrees_east"   ; set units attribute
grid          = new((/4,5/),float)   ; define 2D array
grid!0        = "lat"        ; name left dimension
grid!1        = "lon"        ; name right dimension
grid&lat      = lat_pts      ; assign values to named
                             ; dimension "lat"
grid&lon      = lon_pts      ; assign values to named
                             ; dimension "lon"
```

Coordinate subscripting

For coordinate subscripting, all of the rules for standard subscripting apply except for curly brackets { }, which are used to distinguish coordinate subscripts from standard subscripts.

```
m     = (/-5.0,10.0,15.0,20.0,25.0,30.0/)
m!0   = "lat"          ; name dimension 0
m&lat = m              ; associate the array
mw    = m({-5. : 25. : 2})  ; contains -5.0,-15.0,25.0
```

Use coordinate subscripting to select a subregion in a global grid.

```
var(96,192)          ; 96 lat and 192 lon elements

var_region = var({20:60},{0:70})
```

→ Returns an array containing latitudes nearest to the values between 20 and 60 degrees inclusive, and longitudes nearest to the values between 0 and 70 degrees inclusive.

Statements

If-statement

```
if(scalar_logical_expression) then
    [statement(s)]
else
    [statement(s)]
end if
```

There is no "else if" statement; use a trick to get the same effect. Combine the "if" and "else" on one line, and end with an "end if" for each "if" statement:

```
if(scalar_logical_expression_A) then
   [statement(s)]
else if(scalar_logical_expression_B) then
   [statement(s)]
else if(scalar_logical_expression_C) then
   [statement(s)]
else
   [statement(s)]
end if  ; expression C (includes the "else")
end if  ; expression B
end if  ; expression A
```

Loops

Loops are useful but may not be efficient; they should be used minimally. Use array arithmetic and/or built-in functions if available.

```
do n=start,end[,stride]
    [statement(s)]
end do  ; the stride is not optional if end < start
```

Loop while a logical expression is True:

```
do while(scalar_logical_expression)
    [statement(s)]
end do
```

Use “continue” to skip to next loop iteration; “break” to exit a loop.

Assignment/Reassignment

Assign a variable:

```
var = "This is a string"      ; type string
```

Reassign the variable with a different type and shape:

```
var := (/1,2,3,4/)            ; type integer
```

Metadata and attributes

Metadata is the information associated with a variable or file that describes the data. The metadata of a variable can be attributes like *units*, *_FillValue*, and for a file it can be *creation_date* and *history*.

```
var@units      = "degK"
var@long_name  = "Near Surface Temperature"
var@_FillValue = -99999

title = var@long_name
```

Get the attributes of a variable "slp" of a file "file_name.nc":

```
fin       = addfile("file_name.nc","r")
file_atts = getfilevaratts(fin,"slp")
```

To verify whether an attribute of a variable exists, use *isatt*:

```
if(isatt(slp,"units")) then
   print(slp@units)
end if
```

Print

Print procedures echoing to stdout (standard out).

1. Prints all the values of a variable or expression
   `print(variable_or_expression or file)`
2. Prints summary of a variable's information (commonly used)
   `printVarSummary(data_variable)`
3. Formatted print of all elements from a list
   `print_table(list)`
4. Prints the minimum and maximum value of a variable
   `printMinMax(data_variable,0)`
5. Prints a summary of a file variable's information
   `printFileVarSummary(file,varname)`

Free memory

Use the **delete** procedure to free memory. It can be used to delete a single variable or a variable list.

```
delete(var)
delete([/var1,var2,var3/])
```

User-defined functions and procedures

Generally, functions return values; procedures perform tasks. They must have a **begin** and an **end** statement.

Procedures:

```
undef("procedure_name")
procedure procedure_name(declaration_list)
local local_variables  ; optional, but recommended
begin
   statements
end
```

Functions:

```
undef("function_name")
function function_name(declaration_list)
local local_variables  ; optional, but recommended
begin
   statements
   return(return_variable)
end
```

Functions can return multiple variables contained within a variable of type list:

```
undef("ret_mulvar")
function ret_mulvar(val1,val2)
local ni,nj     ; optional, but recommended
begin
   ni = val1 + val2
   nj = val1 * val2
   return([/ni,nj/])      ; return value list
end

  comp  = ret_mulvar(5,2) ; call function
  v_add = comp[0]        ; retrieve 1st list element
  v_mul = comp[1]        ; retrieve 2nd list element
```

Important built-in functions and procedures

all / any	Returns True if all/any of the values of its input evaluate as True
cd_calendar	Converts a mixed Julian/Gregorian date to a UT-referenced date
conform	Conforms an array to the shape of another
dimsizes	Returns dimension sizes of input variable
exit	Forces an NCL script to exit immediately
ind	Returns indices where the input is True
ismissing	Returns True for every element of the input that contains a missing value
num	Counts the number of True values in input
systemfunc	Executes shell command and returns output
typeof	Returns type of input variable
where	Performs array assignments based on a conditional array