# RÁDIO COGNITIVO PARA O SISTEMA BRASILEIRO DE COLETA DE DADOS

**RELATÓRIO DO PROJETO DE INICIAÇÃO CIENTÍFICA**

**(PIBIC/CNPq/INPE)**

Ivan Pedro Varella Albuquerque (UNP, Bolsista PIBIC/CNPq)

E-mail: ivanvarella@gmail.com

Alexandre Guirland Nowosad (INPE/CRN, Orientador)

E-mail: agnowosad@crn.inpe.br

**Junho de 2009**

## RESUMO

Este trabalho tem objetivo dar continuidade ao desenvolvimento do demodulador de BPSK com recuperação de portadoras para o Sistema Brasileiro de Coleta de Dados (SBCD). O SBCD é um sistema que coleta dados ambientais espalhados pelo nosso território e os retransmite via satélite para as estações terrenas de Cuiabá e Alcântara, em que são processados e enviados à instalação Cachoeira Paulista do Instituto Nacional de Pesquisas Espaciais. Esses dados são usados em pesquisas e assessoria à Administração Pública em geral. O SBCD tem por escopo funcionar em tempo real e com robustez. Em função deste objetivo, optou-se pelo uso da tecnologia denominada “Rádio Definido em Software” neste trabalho. O demodulador tem três estágios, o identificador de portadoras, o *phase locked loop* e o detector binário.

Foi realizado o desenvolvimento do programa de Radio Cognitivo (rádio "inteligente") que simula a identificação das frequencias portadoras de sinais de Plataforma de Coleta de Dados (PCD) do SBCD, ou seja, o estágio de identificação de portadoras. O programa contem 4 módulos. O primeiro gera sinal contendo diversos canais de PCD, o segundo calcula a Transformada de Fourier de Tempo Discreto ("Discrete Fourier Transform - DFT"), o terceiro calcula a Densidade Espectral de Energia ("ESD" ou simplesmente "Spectrum") e o quarto identifica os máximos do "Spectrum", que cujas frequencias são as portadoras desejadas. O programa foi desenvolvido utilizando linguagem C++ através do compilador Dev-C++ e posteriormente o MatLab e o Scilab para visualização dos sinais criados e depois manipulados. O programa foi desenvolvido de forma modulada em C/C++ facilitando manter funcionando em tempo real e modificações futuras, possibilitando a utilização real nos sistemas do INPE. Para o desenvolvimento desse projeto foi necessário estudo de Processamento Digitais de Sinais e outros assuntos relacionados.



Palavras chaves: Rádio cognitivo, Processamento digitais de sinais, C++, MatLab, Scilab, DFT, Fourier.

# COGNITIVE RADIO SYSTEM FOR THE BRAZILIAN COLLECTION OF DATA

## ABSTRACT

This work has the objective to continue development of the BPSK demodulator with carrier recovery of the Brazilian System of Data Collection (SBCD). The SBCD is a system that collects environmental data throughout our territory and satellite relays to ground stations in Cuiaba and Alcantara, which are processed and sent to the installation of Cachoeira Paulista Instituto Nacional de Pesquisas Espaciais. These data are used in research and advice to government in general. The SBCD aims at work in real time and robustness. In light of this objective, the option is the use of technology called "in Software Defined Radio" in this work. The detector has three stages, the identification of carriers, the phase locked loop detector and binary.

Was the development of the program of Cognitive Radio (radio "smart") to simulate the identification of the carrier frequencies of signals from the Data Collection Platform (UCP) of SBCD, ie the stage of identifying carriers. The program contains 4 modules. The first generates signal containing several channels of PCD, the second computes the Fourier Transform of Discrete-Time (Discrete Fourier Transform - DFT), the third calculates the Power Spectral Density ( "ESD" or simply "Spectrum") and fourth identifies the maximum of the "Spectrum", which the carriers whose frequencies are desired. The language program was developed using C + + compiler through the Dev-C + + and then the MatLab and Scilab for visualization of signals created and then manipulated. The program was developed in modular form in C / C + + easier to maintain functioning in real time and future modifications, allowing the use of real systems INPE. For the development of this project was to study Digital Processing of Signals and other related issues.

Keywords: cognitive radio, digital signal processing, C + +, MatLab, Scilab, DFT, Fourier.

**SUMÁRIO**

## LISTA DE FIGURAS

## 1. Introdução

O projeto em questão tem como objetivo criar um programa que identifica as portadoras de um sinal. Futuramente este programa poderá ser utilizado no recebimento de sinais dos satélites do projeto ARGOS, utilizados pelo Instituto Nacional de Pesquisas Espaciais – Centro regional de Natal (INPE-CRN).

O Centro Regional do INPE, com sede em Natal, foi estabelecido por volta de 1970, quando o INPE chamava-se Comissão Nacional de Pesquisas Espaciais (CNAE) e assinou um convênio com o Governo do Estado e Universidade Federal do RN, visando estabelecer um núcleo de apoio aos lançamentos de foguetes e balões operados na Barreira do Inferno.

## 2. O sistema brasileiro de coleta de dados

Esse tópico irá abordar o sistema brasileiro de coloca de dados e tem como base o relatório elaborado por Ramon Augusto Sousa Lins.

A partir do lançamento do primeiro satélite brasileiro o SCD1, foi iniciado a operação do Sistema Brasileiro de Coleta de Dados ambientais. O SBCD foi desenvolvido no intuito de automatizar o processo de aquisição de dados ambientais e do desenvolvimento de tecnologia nacional. Na figura 1 abaixo é mostrado o sistema visto de um modo geral.

Figura 1 - Sistema brasileiro de coleta de dados ambientais

O Sistema atual é composto pelas PCDs, MTRs (Míni Transmissores Remotos) e pelos Satélites SCD1, SCD2 e CBERS (Satélite Sino-Brasileiro de Recursos Terrestres). As PCDs e MTRs coletam, através de vários sensores acoplados, dados ambientais e realizam a geolocalização de embarcações, pessoas e animais transmitindo os dados recolhidos em forma de mensagem na faixa de freqüência UHF2 aos satélites do sistema. Estes satélites, através dos seus *transponders*, recebem a mensagem e as retransmitem na faixa de banda-S3 para as Estações Terrenas de Recepção de Sinais de Satélite. Essas estações demodulam as mensagens, recuperam os dados e as informações transmitidas e as envia para o Centro de Missão, em Cachoeira Paulista

As mensagens enviadas pelas PCDs e MTRs aos satélites obedecem a um padrão, estas são enviadas como mensagens de 360 ms a 920 ms de duração, e transmitidas a intervalos regulares de 40 s a 220 s. Os dados são codificados em *Biphase_L* em 400 bps e modulados em fase ± 60o (com portadora residual) nas freqüências de 401,62 MHz (faixa MECB4) ou 401,65

MHz (faixa ARGOS). Na tabela é mostrado o formato da mensagem:

| TRANSMISSÃO COMPLETA T3 | | | | |
|---|---|---|---|---|
| Portadora pura T1 | Portadora Modulada T2 | | | |
| | Preâmbulo | Tamanho da mensagem | Identificação da Plataforma (ID) | Dados dos sensores |
| T1 = 160 ms ± 2,5 ms | 24 bits (FFFE2F) | 4 bits | 20 bits | N x 32 bits (1≤ N ≤8) |

Figura 2 - Formato e estrutura da mensagem

## 3. Transformada Discreta de Fourier (TDF)

A Transformada Discreta de Fourier (TDF) é calculada usando *Fast Fourier Transform* (*FFT*). Este é um algoritmo eficiente para computar a transformada discreta de Fourier e sua inversa. A FFT é utilizada em diversas aplicações. Como exemplos têm o seu uso no processamento digital de imagens e também no processamento digital de sinais, que será nosso caso. A TDF é dada pela equação:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}$$

## 4. Ondas Portadoras

Ondas portadoras são caracterizadas por três variáveis: amplitude, freqüência e fase. Ela é a responsável por transportar o sinal desejado, a partir de algum tipo de modulação, modificando a onda portadora de acordo com o sinal desejado.

## 5. Funcionamento do *software* de identificação das portadoras

O *software* foi feito em linguagem C++, a partir do compilador DEV-C++, e os resultados verificados no MatLab e Scilab. O software foi separado em funções para simplicidade das alterações. Algumas dessas funções não serão descritas neste trabalho por não serem necessárias para seu entendimento, porém todo o código fonte pode ser verificado no anexo 1. Veja a seguir funcionamento do *software*.

### 5.1. Loop Principal

O objetivo do loop principal é a chamada das funções, que juntas serão capazes de realizar a identificação das portadoras utilizadas para envio de informações pelo satélite. Essa estrutura foi utilizada a fim de obter uma maior facilidade na alteração do programa, veja figura 3.

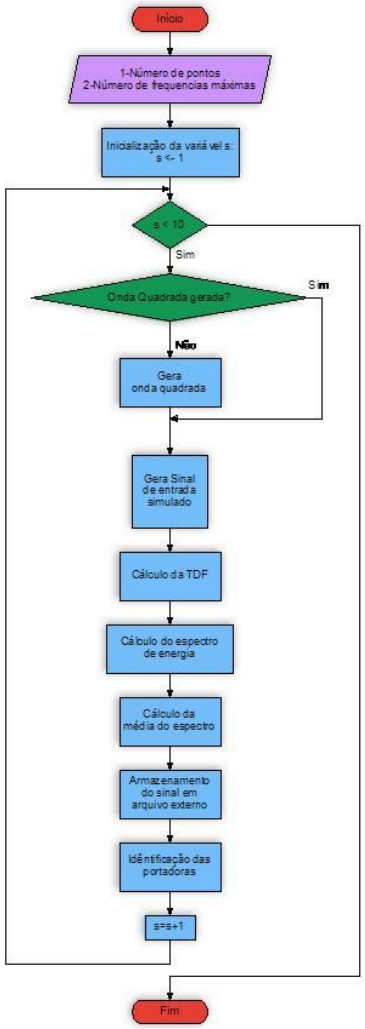


Figura 3 - Loop Principal

### 5.2. Função: Gera onda quadrada

Ela auxilia a função que gera o sinal de entrada simulado (sinalin), ou seja, dá a característica de um sinal senoidal ao sinal de entrada.

x[m] = 2.5*cos(O4*m+(0.1***wquad[m]**)) + 2.75*cos(O3*m+(0.1***wquad[m]**)) + 3.0*cos(O2*m+(0.1***wquad[m]**)) + 4.0*cos(O1*m+(0.1***wquad[m]**));

(Parte do código onde são utilizados os valores da onda quadrada)

A onda quadrada gerada pode ser visualizada na figura 5.

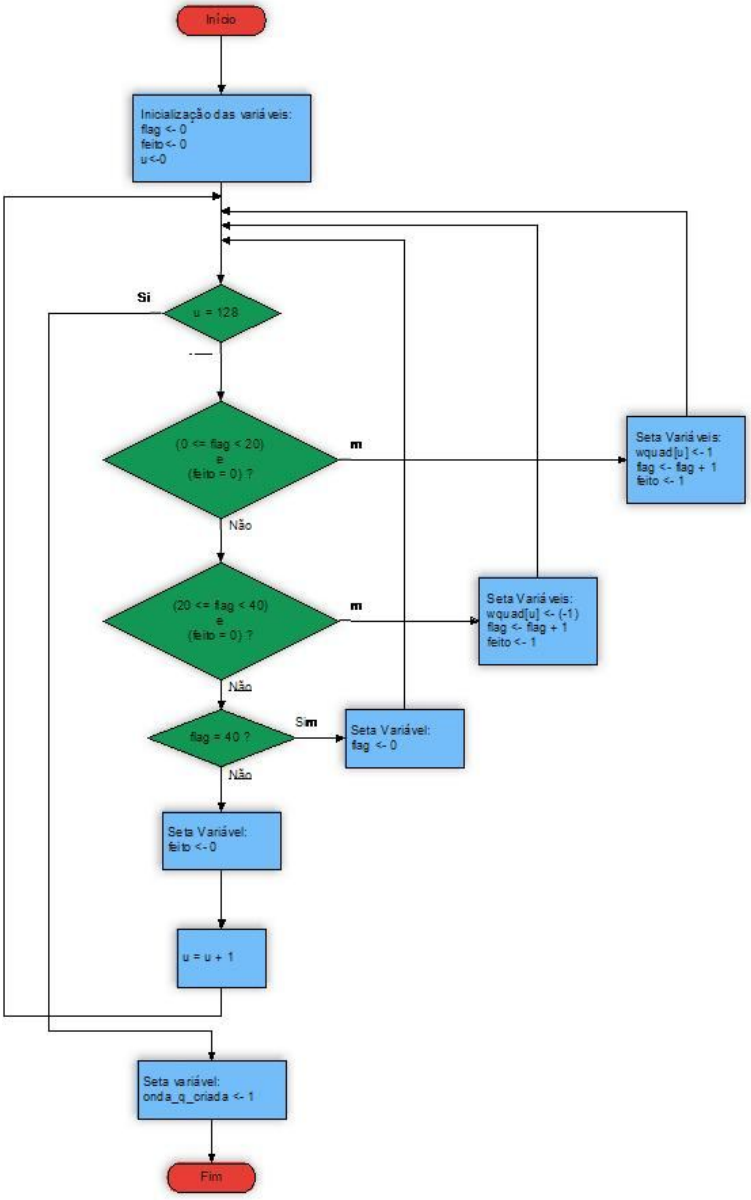


Figura 4 - Gera onda Quadrada

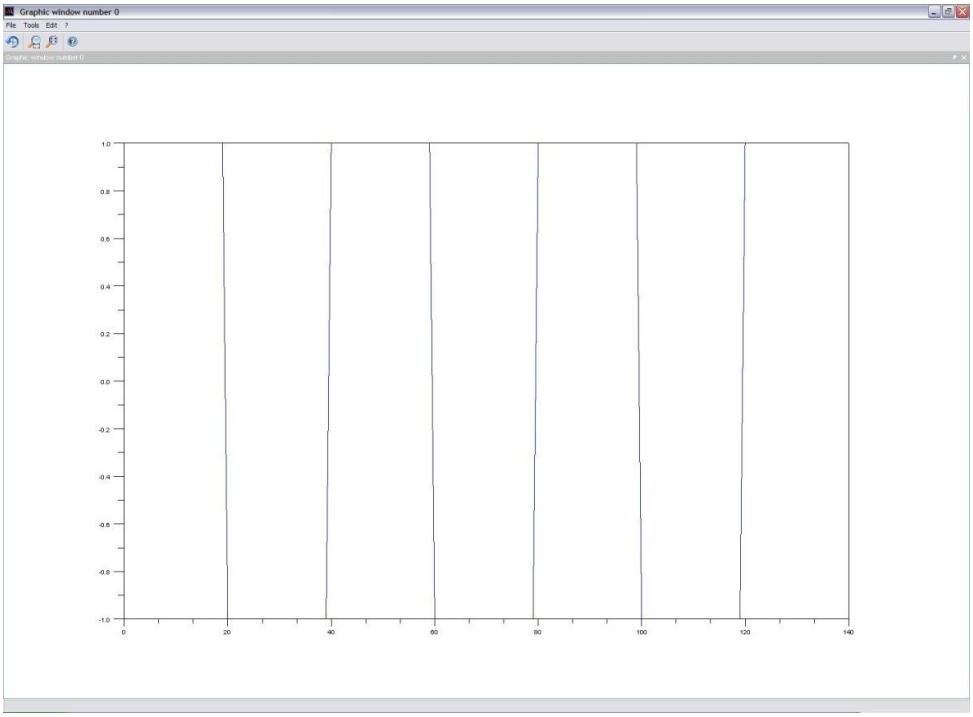


Figura 5 - Onda Quadrada Simulada

### 5.3. Função: Gera sinal de entrada simulado

No futuro esta função deve ser modificada fazendo com que a geração de sinal simulado, que foi criado com o objetivo de testar o programa, obtenha o sinal real e assim seja possível a utilização do programa. Para isso bastar modificar essa função para adquirir o sinal real ao invés da simulação do sinal.

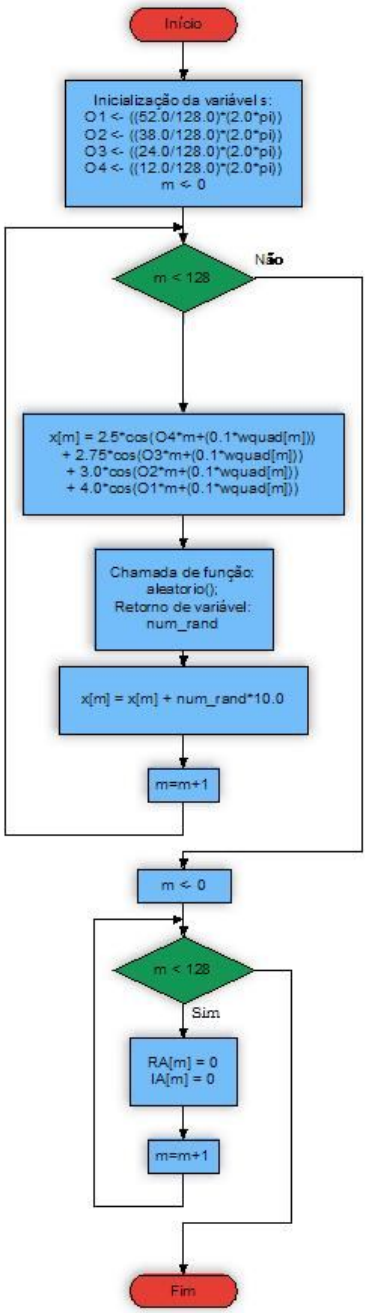


Figura 6 - Gera Sinal de entrada simulado

### 5.4. Função: Cálculo da transformada discreta de Fourier (TDF)

Nesta função é possível obter uma janela do sinal, então se deve verificar qual a quantidade de valores será mais apropriado, modificando as propriedades dos vetores a serem utilizados. No programa usou-se a transformada discreta de Fourier para obter o espectro de freqüências do sinal de entrada, armazenando a parte real dela em um vetor e a imaginária em outro.

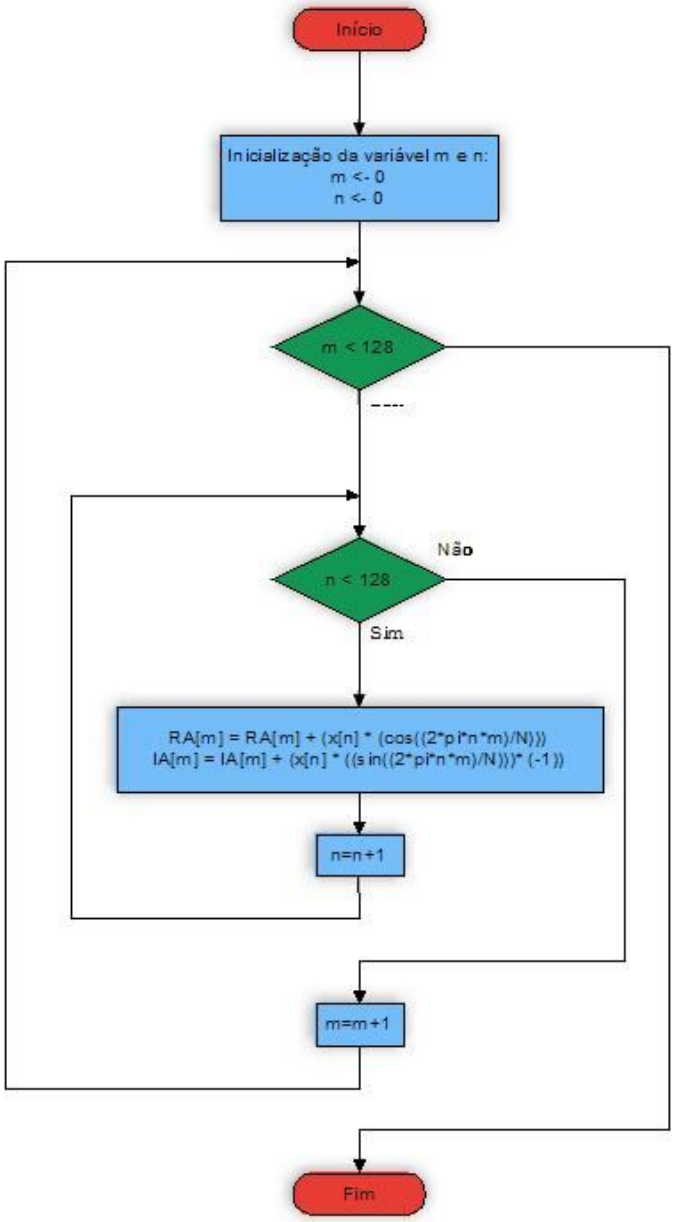


Figura 7 - Cálculo da Transformada Discreta de Fourier (TDF)

### 5.5. Função: Cálculo do espectro de energia

Essa parte do programa é responsável por separar em um vetor somente a metade do sinal obtido pela TDF, já que a partir da metade do sinal até o fim, ele se repete de forma simétrica, não sendo necessário seu uso.

O espectro de energia é o módulo da transformada de Fourier elevado ao quadrado, como é verificado na figura 8.

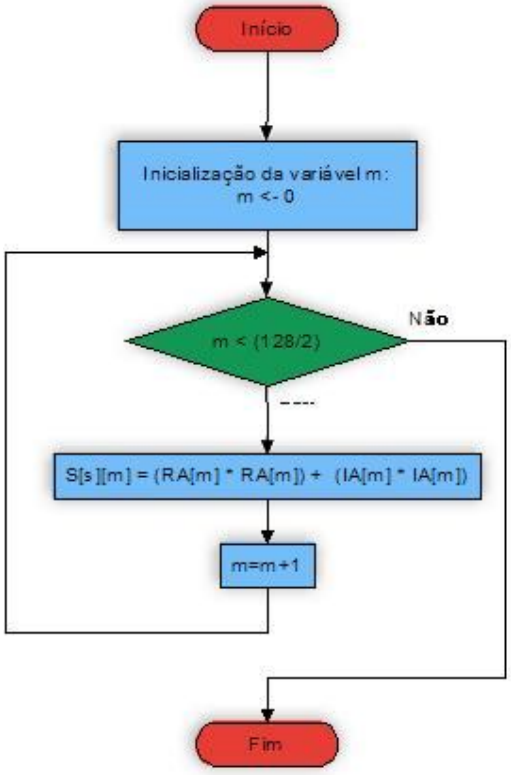


Figura 8 - Cálculo do espectro de energia

### 5.6. Função: Cálculo da média do espectro

Como sabemos, na prática, adicionado ao sinal teremos ruídos (por exemplo numérico) e distorções, então para possibilitar a identificação da portadora utilizaremos essa função para minimizar esses ruídos e distorções, fazendo com que o sinal recebido se sobressaia.

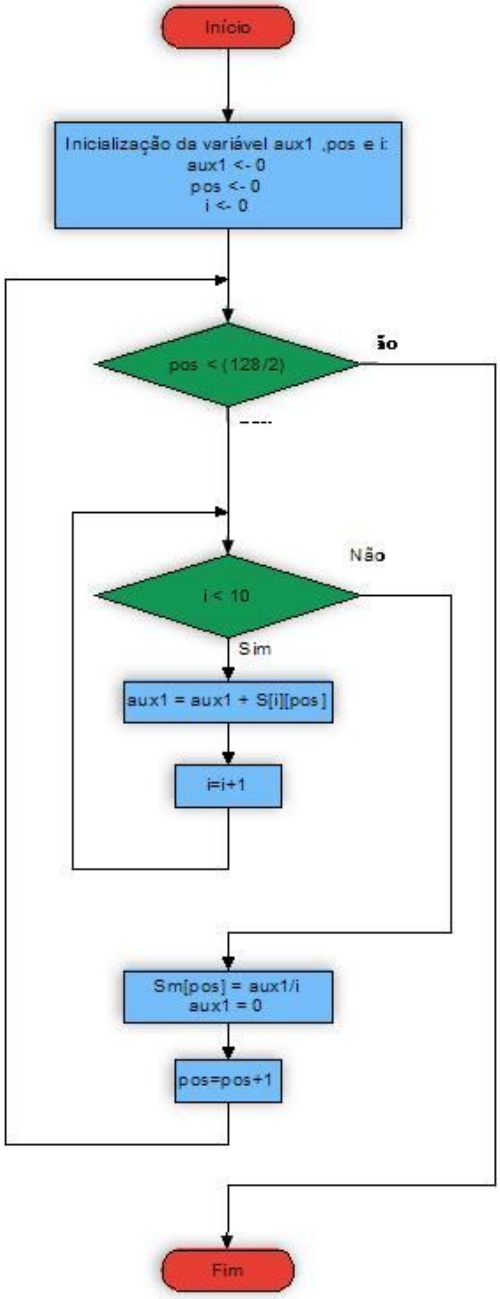


Figura 9 - Cálculo da média do espectro

### 5.7. Função: Identificação das portadoras

Após a obtenção do sinal e seu tratamento, é feita a identificação das portadoras presentes no sinal recebido. As freqüências das portadoras são os locais dos máximos do espectro de energia.

Mostram-se as portadoras do sinal, com suas freqüências digitais e amplitude separadas em vetores diferentes. O número máximo de portadoras identificáveis é escolhido previamente, por isso deve-se configurar as características dos vetores e da função.

Na figura 11 pode-se verificar as portadoras identificadas em um dos testes realizados.

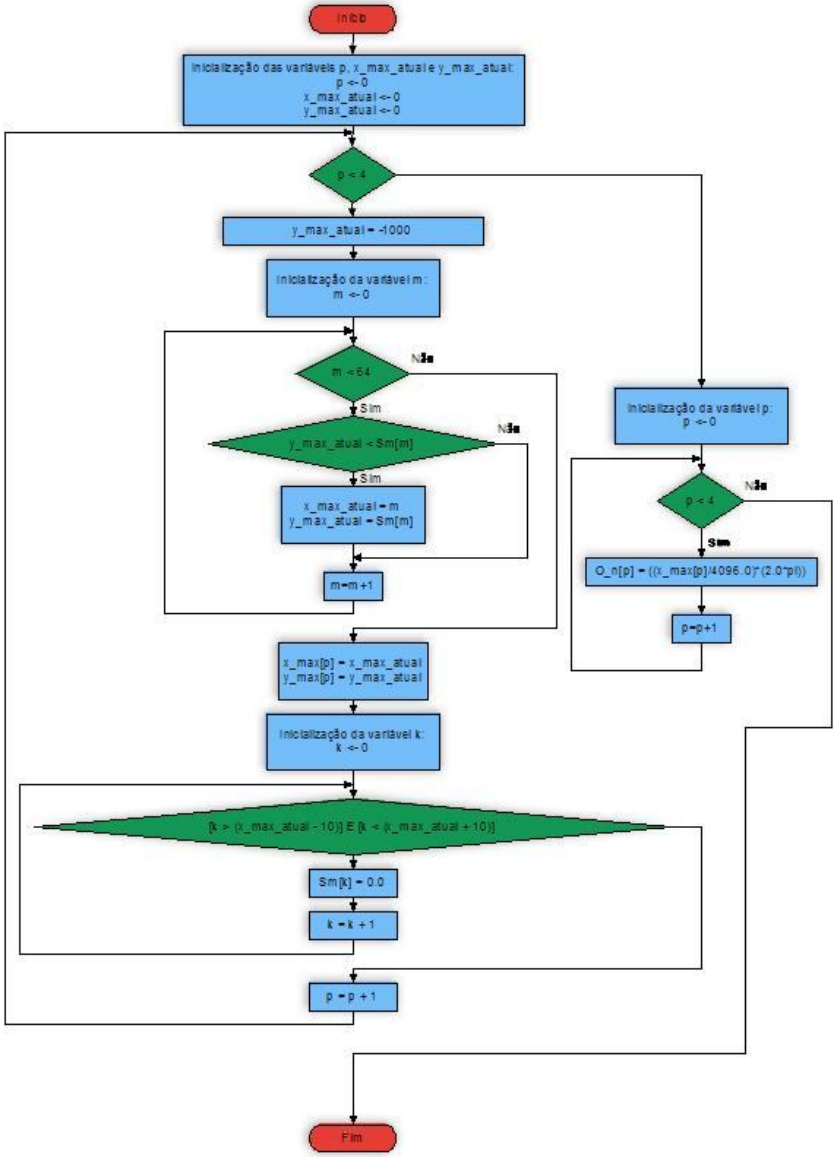


Figura 10 - Identificação das portadoras

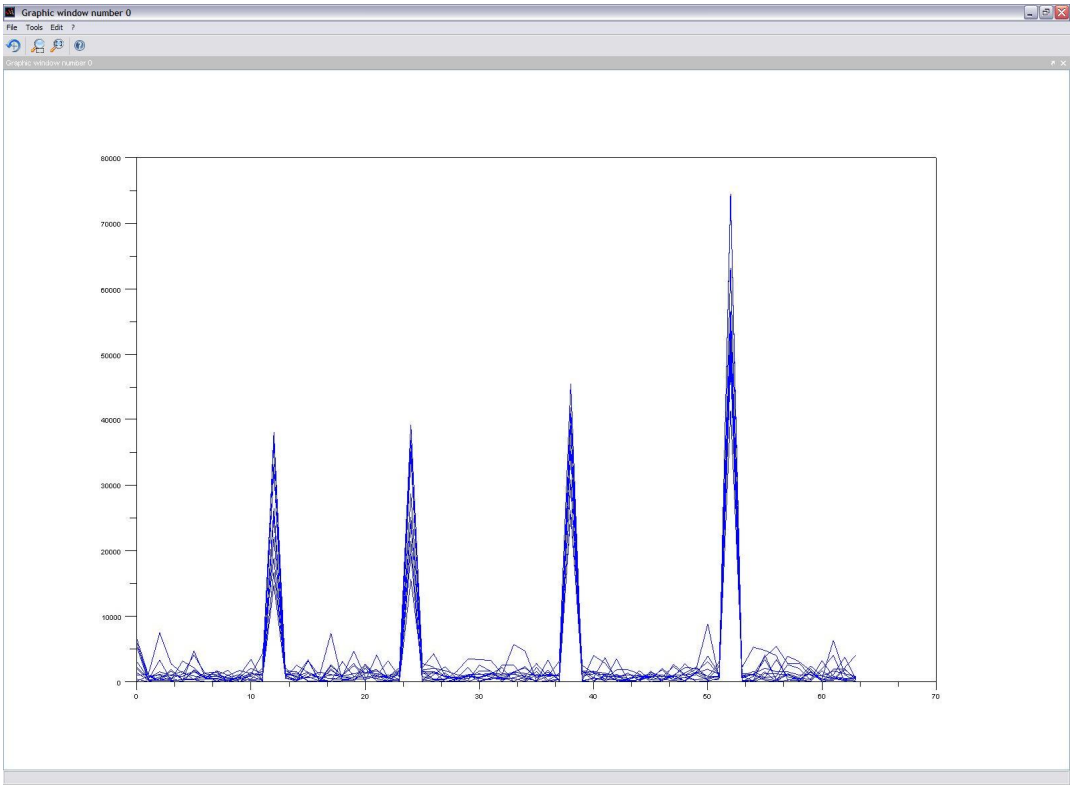


Figura 11 - Portadoras identificadas

## 6. Conclusão

O objetivo deste trabalho foi o de desenvolver programa em C++ de identificação de portadoras para uso no SBCD. O programa foi escrito nesta linguagem para facilitar o uso em tempo real. O programa também é organizado em módulos (ou subrotinas) para ser facilmente atualizado ou corrigido. A identificação de portadoras moduladas por *BPSK* foi realizada calculando-se os máximos do espectro de energia de Fourier.

O sistema construído foi testado usando sinais digitais simulados adicionados a ruídos aleatórios e distorções, incluindo o erro de representação numérica, caracterizado pela discretização do sinal e por perdas ocorridas na conversão do sinal de analógico para digital e vice-versa. Mesmo levando em conta esses fatores, foi possível a identificação das portadoras, como é objetivado neste trabalho.

Porém nos sistemas reais, é necessário que essa identificação seja feita em tempo real ou próximo, o que não foi possível. Duas soluções são propostas neste trabalho:

1- Utilização da transformada rápida de Fourier, por ser mais eficiente que a transforma discreta de Fourier, daria ao programa maior velocidade de processamento.
   A transformada rápida de Fourier pode ser expressa da seguinte forma:

$$F(u) = \sum_{n=0}^{N/2-1} f(2n)\, W_{N/2}^{-kn} + W_{N}^{-k} \sum_{n=0}^{N/2-1} f(2n+1)\, W_{N/2}^{-kn}$$

2- Implementação do programa com utilização de Threads, que consiste em dividir um processo em duas ou mais tarefas que podem ser processadas simultaneamente, fazendo com que o programa seja processado mais rápido.

Essas duas soluções podem ainda ser implementadas simultaneamente, visando se aproximar ainda mais da execução do programa em tempo real.

## 7. Referências

[1] Centre National d'Etudes Sapatiales, CNES. ARGOS PLATFORM TRANSMITTER TERMINALS: General Specifications and certification. Versão2: Julho 1988. 32p. [5] Lathi, B.P., Modern Digital and Analog Communication Systems, Ed. Holt, Rinehart and Winston, New York 1973. 708p.

[2] Hayes, Monson H., Teoria e problemas de processamento digital de sinais, Ed. Bookman, 2006.

[3] S. K. Mitra, Digital Signal Processing: A Computer Based Approach", Ed. MacGraw-Hill, 2006.

[4] Lathi, B.P., Modern Digital and Analog Communication Systems, Ed. Holt, Rinehart and Winston, New York 1973. 708p.

[5] Oppenhem, Alan V., Willsky. Alan S., Young, Ian T., Signals and Systems, Ed. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1983. 796p.

[6] Rae, JCP. Detector de Sinais para os Satélites do Sistema Brasileiro de Coleta de dados usando Análise Espectral Digital. Instituto Tecnológico de Aeronáutica, São José dos Campos. 2005. 123p.

## Anexo 1 – Código fonte do *software* realizado em linguagem C/C++.

```
#include <iostream>

#include <cstdlib>

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <conio.h>

#include <time.h>

#include <fstream>

#define pi 3.14159


using namespace std;


float x[128];

double RA[128], IA[128], S[10][64], Sm[64], x_max[4], y_max[4],
num_rand;

double O_n[4];

int N,n,m, onda_q_criada;

int tam, i, n_max, wquad[128], s;

long seed=0;



float sinalin(int);

void sinalout();

void calc_s();

void arquivo();

void max();

void quadrada();

void arquivo_wquad();

void aleatorio();
```

```
void media_sinal();

int main(int argc, char *argv[]) // Fluxograma OK
{

    onda_q_criada = 0;
    //cout << "Digite o numero de pontos: ";
    //cin >> N;
    //cout << "Digite o numero de frequencias maximas: ";
    //cin >> n_max;
    cout << "Pressione uma tecla para continuar!";
    cout << endl;
    getch();
    N = 128;
    n_max = 4;
    for (s = 0; s < 10; s++){
        if (onda_q_criada != 1){
           quadrada();    // Fluxograma OK
           arquivo_wquad();
        }
        sinalin(N);        // Fluxograma OK
        sinalout();        // Fluxograma OK
        calc_s();
    }
    media_sinal();
    arquivo();
    max();
    cout << endl;

    int p;
```

```
    for (p = 0; p < n_max; p++){

        cout << "x_max[" << p << "]: " << x_max[p] << " | y_max[" << p
<< "]: " << y_max[p] << endl;

    }


    cout << "\n\n";

    system("PAUSE");

    return EXIT_SUCCESS;

}


//Geração do sinal de entrada simulado

float sinalin(int N){


    double O1,O2,O3,O4;


    tam = N;


    //Cossenóide

    O1 = ((52.0/128.0)*(2.0*pi));

    O2 = ((38.0/128.0)*(2.0*pi));

    O3 = ((24.0/128.0)*(2.0*pi));

    O4 = ((12.0/128.0)*(2.0*pi));


    for (m=0;m < N ;m++){


        x[m] = 2.5*cos(O4*m+(0.1*wquad[m])) +
2.75*cos(O3*m+(0.1*wquad[m])) + 3.0*cos(O2*m+(0.1*wquad[m])) +
4.0*cos(O1*m+(0.1*wquad[m]));

        aleatorio();

        //cout << "numero aleatorio: " << num_rand << endl;
```

```
        x[m] = x[m] + num_rand*10.0;

    }

    /*//Sinal de entrada - Impulso
    for (m=0;m < N ;m++){

        if (m == 0){
           x[m]= 1;
        }
        else{
           x[m] = 0;
        }
    }*/

    /*Inicialização dos vetores auxiliares para calcular o sinal*/
    for (m=0;m < N;m++){
        RA[m] = 0;
        IA[m] = 0;
    }

}

//Calculo da DFT
void sinalout(){

    /*Calculo DFT*/
    for (m=0;m < N;m++){
        for (n=0;n < N;n++){
            RA[m] = RA[m] + (x[n] * (cos((2*pi*n*m)/N)));
```

```
            IA[m] = IA[m] + (x[n] * ((sin((2*pi*n*m)/N)))* (-1));
        }
    }

}

//Calculo do Espectro de energia
void calc_s(){

    /*Calculo do S*/
    for (m=0;m < (N/2);m++){
        S[s][m] = (RA[m] * RA[m]) +  (IA[m] * IA[m]);
    }

}

void arquivo(){

    /*Impressão do sinal no arquivo*/

    ofstream saida("Sinal.txt");      //Abre o arquivo

for (s = 0; s < 10;s++){

    if (s != 0){
       saida<<endl;                  //Pula linha
    }

    //Impressão dos pontos no eixo X
    saida<<"x"<<s<<"=[";
```

```
    for (i = 0; i < tam/2;i++){
        if (i < (tam/2)-1){
           saida<< i <<";";
        }
        else
        {
            saida<< i ;
        }
    }
    saida<<"];"<<endl;

    //Envia os dados para o arquivo:
    saida<<"y"<<s<<"=[";
    //Joga os valores do sinal
    for (i = 0; i < tam/2;i++){
        if (i < (tam/2)-1){
           //saida<<x[i]<<";";
           saida<<S[s][i]<<";";
        }
        else
        {
            //saida<<x[i];
            saida<<S[s][i];
        }
    }
    saida<<"];"<<endl;

    //Comando plot
    saida<<"plot(x"<<s<<",y"<<s<<")";
} //For 10 sinais
```

```
//Média dos Espectros:
saida<<endl;
//Eixo X
saida<<"xm=[";
for (i = 0; i < tam/2;i++){
    if (i < (tam/2)-1){
       saida<< i <<";";
    }
    else
    {
        saida<< i ;
    }
}
saida<<"];"<<endl;
//Eixo Y
saida<<"ym=[";
//Joga os valores do sinal
for (i = 0; i < tam/2;i++){
    if (i < (tam/2)-1){
       saida<<Sm[i]<<";";
    }
    else
    {
        saida<<Sm[i];
    }
}
saida<<"];"<<endl;
saida<<"plot(xm,ym)";
```

```
    //Fechamento do arquivo - arquivo já gerado..
    saida.close();
}

//Identificação das portadoras
void max(){

     int p,k;
     double x_max_atual, y_max_atual;

     //n_max = 4;

     for (p = 0; p < n_max; p++){  //Percorre todo o sinal o numero de
pontos a serem encontrados

         y_max_atual = -1000;

         //N = 128

         for (m=0;m < (N/2);m++){  //Percorre todo o sinal

             if (y_max_atual < Sm[m]){  //Pega o maximo
                x_max_atual = m;
                y_max_atual = Sm[m];
             }
        }

        // Armazena os valores de máximo
        x_max[p] = x_max_atual;
        y_max[p] = y_max_atual;
```

```
        for (k = (int(x_max_atual - 10));k < (int(x_max_atual +
10));k++){ // Percorre o sinal na regiao (-10 ate + 10) do max
encontrado, apagando essa regiao
            Sm[k] = 0.0;
        }


    }


    // Calculando os Omegas Estimados:
    for (p = 0; p < n_max; p++){
        O_n[p] = ((x_max[p]/4096.0)*(2.0*pi));
    }


}


void quadrada(){
    int u, flag, teste_flag[128], feito;


    flag = 0;
    feito = 0;


    for (u = 0; u < 128; u++){


        if ( (flag >= 0) && (flag < 20) && (feito == 0) ){
           wquad[u] = 1;
           flag++;
           feito = 1;
        }


        if ( (flag >= 20) && (flag < 40) && (feito == 0) ){
           wquad[u] = -1;
```

```
            flag++;
            feito = 1;
          }

          if (flag == 40){
            flag = 0;
          }

          feito = 0;

      }
      onda_q_criada = 1;
}

void arquivo_wquad(){
      //Gerar aquivo para teste de grafico....
      /*Impressão do sinal no arquivo*/

     ofstream saida("Sinal_wquad.txt");      //Abre o arquivo

     //Impressão dos pontos no eixo Y
     saida<<"x=[";
     for (i = 0; i < 128;i++){
         if (i < 127){
            saida<< i <<";";
         }
         else
         {
             saida<< i ;
         }
```

```
    }
    saida<<"];"<<endl;

    //Envia os dados para o arquivo:
    saida<<"y=[";
    //Joga os valores do sinal
    for (i = 0; i < 128;i++){
        if (i < 127){
           //saida<<x[i]<<";";
           saida<<wquad[i]<<";";
        }
        else
        {
            //saida<<x[i];
            saida<<wquad[i];
        }
    }
    saida<<"];"<<endl;

    //Comando plot
    saida<<"plot(x,y)";

    //Fechamento do arquivo - arquivo já gerado..
    saida.close();
}

void aleatorio(){
  int num_int;
  num_int = rand() % 101;
  num_rand = (num_int-50.0)/100.0;
```

```
}


void media_sinal(){
     int pos;
     double aux1;


     aux1 = 0;
     for (pos=0;pos < (N/2);pos++){
         for (i = 0; i < 10;i++){
             aux1 = aux1 + S[i][pos];
         }
         Sm[pos] = aux1/i;
         aux1 = 0;
     }
}
```

**Anexo 2 – Código criado pelo *software* para geração do sinal da onda quadrada.**

```
x=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25
;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48;
49;50;51;52;53;54;55;56;57;58;59;60;61;62;63;64;65;66;67;68;69;70;71;7
2;73;74;75;76;77;78;79;80;81;82;83;84;85;86;87;88;89;90;91;92;93;94;95
;96;97;98;99;100;101;102;103;104;105;106;107;108;109;110;111;112;113;1
14;115;116;117;118;119;120;121;122;123;124;125;126;127];

y=[1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;-1;-1;-1;-1;-1;-1;-1;-1;-
1;-1;-1;-1;-1;-1;-1;-1;-1;-1;-1;-
1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;-1;-1;-1;-1;-1;-1;-1;-1;-
1;-1;-1;-1;-1;-1;-1;-1;-1;-1;-1;-
1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;-1;-1;-1;-1;-1;-1;-1;-1;-
1;-1;-1;-1;-1;-1;-1;-1;-1;-1;-1;1;1;1;1;1;1;1];

plot(x,y)
```

Esse código foi utilizado no Matlab e Scilab para vizualização.

**Anexo 3 - Código criado pelo *software* para geração do gráfico da identificação das portadoras.**

Esses códigos foram utilizados no Matlab e Scilab para vizualização, separadamente ou não.

```
x0=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];

y0=[1427.17;238.331;772.665;1085.83;1613.54;360.862;469.883;483.667;25
6.949;548.707;231.985;624.227;14819.3;998.106;1383.91;171.434;29.5523;
2014.79;368.084;265.04;2480.95;904.94;631.32;170;21432.4;301.628;1616.
56;776.743;745.704;944.672;1274.92;760.431;517.838;532.072;121.469;222
2.05;264.509;1355.9;40881.5;1693.21;206.434;877.472;876.063;253.43;323
.491;878.415;871.891;793.541;2784.4;1105.55;245.824;421.588;53522.3;89
1.459;172.703;4114.4;208.135;2642.4;2067.35;327.75;1996.28;4064.64;59.
4805;828.571];

plot(x0,y0)

x1=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];

y1=[5987.34;324.821;537.379;367.913;3189.52;2185.27;550.3;1681.42;34.0
233;384.44;1018.38;1624.06;33118.7;713.129;53.6069;3293.33;65.0026;247
6.12;888.491;4699.68;815.806;520.544;144.989;308.712;15619.8;1802.57;2
124.23;1734.5;1132.42;836.879;569.239;335.92;2610.14;2580.02;548.641;2
22.624;1802.72;266.615;35963.9;2551.12;1366.83;1046.91;243.97;74.9464;
375.218;1283.94;375.381;254.446;86.2172;1500.92;374.821;2415.3;51349.3
;57.3566;874.643;3426.55;443.854;1264.99;689.24;2478.88;915.358;541.00
4;206.639;638.537];

plot(x1,y1)

x2=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];

y2=[5932.41;865.649;278.037;1897.8;395.892;363.487;1014.29;136.32;340.
685;783.663;2100.64;1230.84;32418;706.31;2561.94;1465.1;765.863;524.34
9;479.546;404.538;1286.12;1718.42;793.036;1827.5;36851;354.217;141.768
;1418.66;1.5815;1160.76;123.581;481.743;670.819;109.693;824.673;114.06
;3382.65;21.6663;30825.8;1768.96;649.392;3708.61;188.933;331.221;19.16
;1672.81;10.5509;2723.72;1120.77;83.1452;159.207;584.373;74287.4;1258.
79;1175.8;3817.28;5418.91;2822.97;2828.07;1064.14;369.914;219.939;1905
.89;472.048];

plot(x2,y2)
```

```
x3=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];

y3=[6564.59;1078.39;247.128;312.166;151.05;4727.7;708.857;1701.43;1021
.67;1772.26;1284.31;979.514;16752.7;1726.7;1532.98;46.6593;241.601;182
.543;1011.67;2478.14;1432.09;1930.75;243.768;393.083;34488.8;1526.45;1
538.45;65.6943;467.866;2253.31;171.908;629.689;347.846;1548.46;2171.53
;697.599;1539.59;267.934;26113.5;740.918;1597.3;1361.59;868.118;1079.2
1;944.384;663.717;1878.95;223.701;1417.36;2271.68;408.193;973.501;5549
6.7;303.983;154.893;1291.85;1249.32;1092.82;399.057;1599.66;88.8479;40
8.258;2205.69;4134.49];

plot(x3,y3)

x4=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];

y4=[1077.29;1169.87;7494.07;2778.31;1311.51;925.884;681.308;1006.88;18
01.18;42.2796;1545.47;315.825;25276.3;986.14;316.961;1289.42;551.388;6
8.0423;225.603;264.504;2484.53;566.126;1080.4;390.578;19518;103.264;11
15.38;2335.3;631.087;236.43;2585.51;1653.21;370.376;661.686;1134.87;17
.0967;195.281;468.229;39041.6;1740.38;1513.03;67.441;1242.98;34.8268;4
23.652;1518.2;810.881;2355;1009;2010.02;8883.28;472.832;74467.5;60.922
8;297.723;1570.59;79.2281;1249.95;85.0274;2093.62;234.989;1136.82;3780
.3;367.018];

plot(x4,y4)

x5=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];

y5=[5332.23;246.433;285.732;601.257;6.52994;1873.41;750.312;859.106;25
9.996;714.018;856.105;239.297;22082.9;1290.42;1013.3;3367.39;174.798;2
645.61;852.483;414.946;436.018;894.883;3252.17;782.317;24225.6;2184.43
;4321.96;983.709;515.823;886.141;781.107;1240.4;2207.29;1.711;51.4147;
1166.79;887.466;3196.66;45544;117.625;90.8596;1072.64;333.599;104.473;
1731.59;329.993;2076.59;628.417;234.703;1615.72;3101.25;1115.21;49778.
1;924.877;1082.45;266.076;74.3621;271.306;616.314;413.274;3246.49;853.
75;388.976;904.435];

plot(x5,y5)

x6=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];

y6=[429.407;64.7581;558.034;1654.15;323.768;342.824;74.1574;125.488;10
44.43;53.0147;459.043;2090.38;36500.1;548.83;340.379;3194.34;1033.1;74
20.3;22.8603;31.3369;2109.86;176.549;92.8473;1468.2;35033.2;1881.37;31
6.064;326.294;1577.81;3541.53;3409.31;3293.44;1240.52;1183.22;844.863;
2810.12;888.881;991.822;30352.6;222.158;535.955;234.171;3521.08;356.33
8;1166.57;356.302;497.263;931.614;194.68;1589.28;868.128;857.887;41357
```

```
.9;2227.62;5305.48;4829.65;4030.94;324.928;210.737;1480.39;506.34;6339
.57;1132.32;78.1233];


plot(x6,y6)

x7=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];

y7=[2220.5;635.334;1190.3;593.636;1205.83;4117.36;1229.55;1112.51;650.
324;1150.84;359.062;34.6095;38042.5;1777.27;687.596;693.817;744.497;10
38.56;1116.11;2856.76;170.403;4102.64;470.994;1696.17;35112.9;862.95;3
28.812;68.2374;176.655;82.2604;660.763;1131.45;688.531;1397.32;318.513
;1611.23;7.64743;390.696;41819.7;170.013;4050.73;2887.26;551.588;1379.
97;236.584;798.851;253.2;896.776;650.431;343.258;630.934;827.613;48379
.3;949.161;1609.03;705.625;3387.29;798.978;265.707;134.207;1679.29;599
.701;361.599;148.057];

plot(x7,y7)

x8=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];

y8=[161.853;890.779;1061.79;145.423;391.626;490.531;1390.65;1490.82;49
.6383;11.1027;1164.97;4325.36;18946.7;957.84;229.326;863.657;1522.99;1
141.96;798.113;675.487;2728.14;853.349;789.376;2196.59;39200.5;2807.54
;2275.5;750.054;375.767;830.295;640.068;1407.82;1165.17;5720.06;4716.0
2;172.305;920.915;1140.77;34940.9;138.886;644.945;438.252;329.366;431.
83;832.72;1081.36;1104.1;364.911;690.821;1132.57;3979.17;1209.39;63171
.1;605.128;1207.65;498.312;203.484;1120.45;518.096;20.2506;5.68939;747
.418;591.198;608.788];

plot(x8,y8)

x9=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];

y9=[2077.34;374.292;3353.18;209.505;358.617;1569.23;579.346;576.905;27
.5264;1017.73;3486.67;123.659;26130.2;2200.9;306.715;829.21;145.487;14
6.075;3132.97;836.606;694.642;1123.9;98.5796;1633.4;25082.5;1433.84;45
4.297;503.152;766.623;511.707;1712.42;1640.22;99.1162;1458.08;2362.6;6
92.276;1187.92;956.614;25591.5;1284.4;1670.17;421.701;1844.52;1888.18;
1240.69;80.4645;219.773;1049.19;2201.77;2085.53;821.713;2937.67;55071.
2;85.6563;34.7175;680.74;863.712;3946.11;3109.91;1718.33;227.426;1936.
81;2003.44;220.35];

plot(x9,y9)

xm=[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;2
5;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41;42;43;44;45;46;47;48
;49;50;51;52;53;54;55;56;57;58;59;60;61;62;63];
```

```
ym=[3121.01;588.865;1577.83;964.598;894.788;1695.66;744.866;917.455;54
8.642;647.805;1250.66;1158.78;26408.7;1190.56;842.671;1521.44;527.427;
1765.83;889.593;1292.7;1463.86;1279.21;759.748;1086.66;28656.5;1325.83
;1423.3;896.235;639.134;1128.4;1192.88;1257.43;991.764;1519.23;1309.46
;972.616;1107.76;905.691;35107.5;1042.77;1232.56;1211.61;1000.02;593.4
42;729.406;866.404;809.858;1022.13;1039.01;1373.77;1947.25;1181.54;566
88.1;736.496;1191.51;2120.11;1595.92;1553.49;1078.95;1133.05;927.063;1
684.79;1263.55;840.042];

plot(xm,ym)
```