Journal of the Brazilian Society of Mechanical Sciences and Engineering (2021) 43:198
https://doi.org/10.1007/s40430-021-02908-7

TECHNICAL PAPER

Spacecraft real-time thermal simulation using artificial neural networks

J. D. Reis Junior[1] · A. M. Ambrosio[1] · F. L. de Sousa[1] · D. F. Silva[1]

Received: 23 July 2020 / Accepted: 22 February 2021

Abstract
Spacecraft Operational Simulators are mainly used for training satellite operators, to test the ground control system, and the evaluation of operational and onboard procedures before their execution in the real satellite. To achieve these objectives, all the internal models of the Operational Simulator must provide information in real time. Traditionally, the thermal simulation in these simulators is accomplished through interpolation on a set of pre-calculated scenarios or by the integration of a very simplified mathematical model. Both approaches, however, have limitations in both fidelity and runtime. In order to overcome these limitations, in this work it is proposed to build the thermal model of a Spacecraft Operational Simulator using artificial neural networks. This approach was applied to the Amazonia-1, a medium size satellite currently being developed at the Brazilian National Institute for Space Research. The obtained results show increased fidelity and an extremely short execution time, evidencing the potential of the approach to simulate satellite thermal behavior in Operational Simulators.

Keywords Real-time simulation · Artificial neural networks · Machine learning · Thermal control · Space engineering

Abbreviations

ACE	Attitude control electronics
ANN	Artificial neural network
EPC	Electronic power controller
GD	Gradient descent algorithm
GPS	Global positioning system
GYRO EM	Gyro sensor electronic module
GYRO ICU	Gyro sensor inertial control unit
IR	Earth's infrared radiation
LNA	Low noise amplifier
MLP	Multi-layer perceptron
OBDH	On-board data handling
PCDU	Power conditioning and distribution unit
QPSK-TX	Quadrature phase shift keying
RTU	Remote terminal unit
SADA	Solar array drive assembly
SADE	Solar array drive electronics
SDC	Subsystem Distribution controller
SINDA	Systems improved numerical differencing analyzer
SPE	Signal processing electronics
SSR	Solid state recorder
TCE	Thruster control electronics
TT&C	Telemetry, tracking and command
TWT	Travelling wave tube

1 Introduction

With the advancement of computer processing capabilities, computational modeling and simulation have been progressively used in the development process of complex systems. In particular for space systems, simulations contribute to reduce costs and time required for development, since they allow the evaluation of functionality and performance without the need to construct physical models and to test scenarios that would not be possible in the laboratory, increasing the reliability of the system [1–6]. Simulations are used along all the satellite development phases, from the system conception, through the establishment of mission performance requirements, functional engineering processes, functional validation, software validation, assembly, integration and test (AIT), ground system test, to the final phases for training, operations and maintenance [7, 8]. In

Technical Editor by Francis H. R. Franca.

✉ J. D. Reis Junior
daniel.reis@inpe.br

[1] National Institute for Space Research, Sao Jose dos Campos, SP 17227-010, Brazil

Published online: 12 March 2021

the early phases, the models are simpler and more general because the details of the system have not yet been defined. While in the final stages, as the system is already completely defined, the models are much more complex and have a high fidelity to the real system [9].

In this work, the focus is on the simulators used for training, operations, and maintenance called here simply Operational Simulators. These simulators are mainly used in the pre-launch phase for ground system validation and operator training, and also during the operation for prior evaluation of operational and onboard procedures before performing them with the actual system and also for analysis of contingency situations. The Operational Simulator allows executing complete scenarios so realistically that it is virtually impossible to tell the difference between a simulated scenario and the actual operation of the satellite [10, 11].

In order to meet these requirements, the Operational Simulator must provide information in real time, i.e. the information must be available to the operator at the same time as it would be if the actual satellite were in contact with the satellite control system. This implies that all internal models of the simulator must run in a short enough time so that the response of the system as a whole is given in real time, having also high dependability requirements [12].

In general terms, the simulation of a spacecraft thermal behavior with high fidelity has a high computational cost (consequently requiring an extensive time to be processed), since the computational methods necessary to determine the temperature distribution require the resolution of a large number of differential equations.

For the satellite thermal design, usually, commercial software packages are used, in which the satellite is discretized in a set of nodes. This set can reach thousands of nodes on an average satellite like the Amazonia-1 [13]. The discretized set of equations is solved using mainly the finite difference and finite element methods and must be calculated each time the thermal state of the satellite is required. This type of simulation, applied to a complete satellite, in different operating scenarios and environmental conditions, presents a high computational cost. For example, the simulation of each scenario of Amazonia-1 satellite running on a personal computer, using the SINDA/FLUINT software [14], took up to 10 h, which makes its direct use impracticable in an Operational Simulator.

In the literature, there are very few works on the construction of thermal model for operational simulators [15, 16]. According to Perpiñán [15], there are basically two alternatives to perform this task: interpolation over predefined scenarios or integration of a simplified thermal model.

Both the interpolation and integration approaches present limitations for their use in a real-time simulator. Generally, the interpolation is performed between scenarios in which the thermal behavior of the satellite was predetermined. Despite producing good results for these scenarios, the interpolation may produce results with unpredictable errors for the scenarios that were not previously calculated, because the temperature varies in a nonlinear way and depends on the combination of many system and environmental parameters. On the other hand, in the second approach, the satellite thermal model is simplified so that it is possible to integrate the heat transfer equations in real time. The disadvantage of this approach is the loss of fidelity in comparison with the actual satellite for all scenarios.

Recently, Manon et al. [16] proposed a hybrid approach in which the external heat fluxes on the satellite and the internal radiation exchange factors are pre-calculated for many discretized directions and then used to build a coefficient table. During the simulation, the total flux received by each node is determined by interpolation in the coefficient table using information from satellite attitude and orbital position, as well as the power of the source and the physical properties of the node. The temperatures of each node are then calculated by integration of the heat equations. The authors applied this method to a CubeSat and obtained promising results.

In this work it is proposed the use of machine learning, particularly, artificial neural networks (ANN) of the recurrent type, for the simulation of the thermal behavior of satellites in real time.

The recurrent ANNs are universal approximators of nonlinear dynamic systems [17]. The choice of neural networks is due to their ability to map complex relationships between large input and output data sets, as well as to interpolate or generalize, that is, to provide reasonable outputs for inputs not presented in the training [18, 19].

In recent years, neural networks have been successfully applied in many aerospace engineering problems [20–23] and also in general thermal problems [24–26]. In all these works, neural networks have proven to be good alternatives (sometimes even better) to the traditional approaches.

During the research, different architectures and training algorithms were experimented, including the Levenberg–Marquardt [38, 39] and an adaptation of the generalized extreme optimization algorithm [40] to artificial neural networks. All algorithms provided similar results, but the gradient descent (GD) algorithm provided the smallest generalization errors and with shorter training times.

In addition, the complexity of the simulated source were gradually increased, first an aluminum sphere, then a CubeSat [27–29], and finally a real medium-sized satellite. In this work it is presented the final ANN architecture and the results obtained with its application for the Amazonia-1 satellite [30], which is currently being developed at the National Institute for Space Research (INPE).

This machine learning approach has shown the advantage of presenting high fidelity for the trained scenarios, small

mean generalization error for the untrained ones and very low computational cost.

The paper is organized as follows, Sect. 2 describes the Amazonia-1 satellite, Sect. 3 discusses artificial neural networks, Sect. 4 presents the approach used to simulate the satellite thermal behavior using RNN, Sect. 5 shows and discusses the results, and Sect. 6 concludes the paper.

2 Amazonia-1 satellite

The Amazonia-1 is a remote sensing satellite, whose main objective is to monitor Brazilian natural resources, especially the Amazon rainforest and agricultural areas [30]. The generated data will also be useful to attend other related applications, such as monitoring of coastal region, water reservoirs, natural and cultivated forests, environmental disasters, among others. The satellite, which will act in synergy with existing environmental programs, is being completely designed, integrated and tested in Brazil. In Fig. 1 is shown the satellite's physical model used for the thermal balance tests.

Some features of Amazonia-1 are: dimensions $2.20 \times 0.95 \times 0.95$ m; about 650 kg; Sun-synchronous orbit; 752.4 km of altitude; 98.405º of inclination (polar); 3 axes of stabilization; 4 years of mission lifetime.

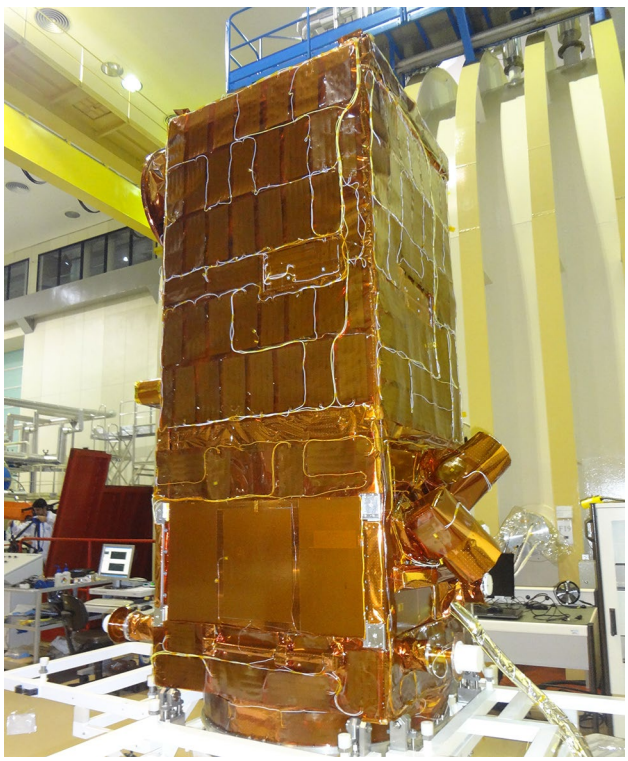

Fig. 1 Thermal balance physical model of the Amazonia-1 satellite

The Amazonia-1 is a Sun-synchronous (polar) orbiting satellite with a 5-day revisit rate. It has a wide-view optical imager, a camera with 3 visible frequency bands and 1 near infrared band, capable of observing a swath of approximately 850 km with 60 m of resolution.

In addition to the objectives associated with providing data for monitoring the environment, the Mission has an important technological goal: the validation of the Multi-Mission Platform (PMM) as a system, which will be used for the first time in the Amazonia-1 satellite.

The PMM, developed at INPE with financial support from the Brazilian Space Agency (AEB), is a generic platform for LEO satellites in the 500 kg class. It provides pointing, power generation, data management and service telecommunication for the payload, and is capable of being used in different types of missions.

Figure 2 shows an exploded view of the Amazonia-1 satellite, with the bus (PMM) on the left and the payload module on the right.

The Amazonia-1 thermal mathematical model was built using ThermalDesktop/Radcad—SINDA/FLUINT thermal software package [14], which runs under an AutoCad 2005 platform. The whole model is composed by 11,806 nodes, including diffusion and arithmetic nodes (used to represent components with very low thermal capacitances), and is shown in Fig. 3.

3 Artificial neural networks

In its most general form, an artificial neural network (ANN) is a virtual machine designed to model the way the biological brain acquires knowledge through experience [18]. To achieve this goal, artificial neural networks employ a vast interconnection of simple computational cells or processing units, called neurons.

An artificial neuron is the fundamental information processing unit for the operation of a neural network. Figure 4 illustrates the model of an artificial neuron in a diagrammatic way. In this diagram, three basic elements of the neuronal model can be identified: a set of synapses or connection links, each one characterized by its own intensity, called synaptic weight (w); a fixed input, called bias; a summation junction, which performs a linear combination of the input signals (x) with the synaptic weights; and an activation function (f) that restricts the amplitude of the output of the neuron to a finite value (s).

Figure 5 shows a general architecture of a multi-layered network, called multi-layer perceptron (MLP) [18, 31]. MLP networks have one or more inner layers of neurons, called hidden layers. The circles in the figure represent the artificial neurons.

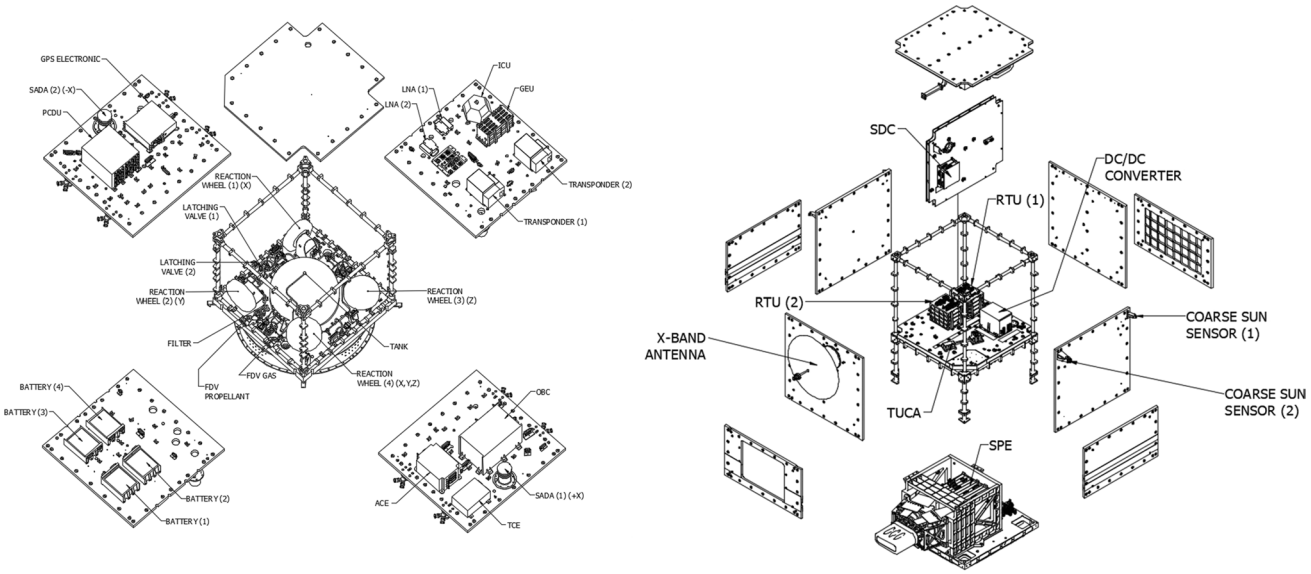

Fig. 2 Exploded view of the Amazonia-1 satellite

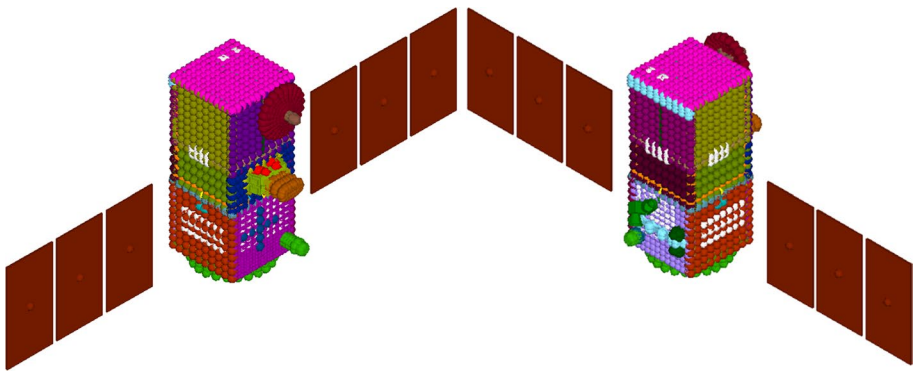

Fig. 3 Amazonia-1 thermal mathematical model

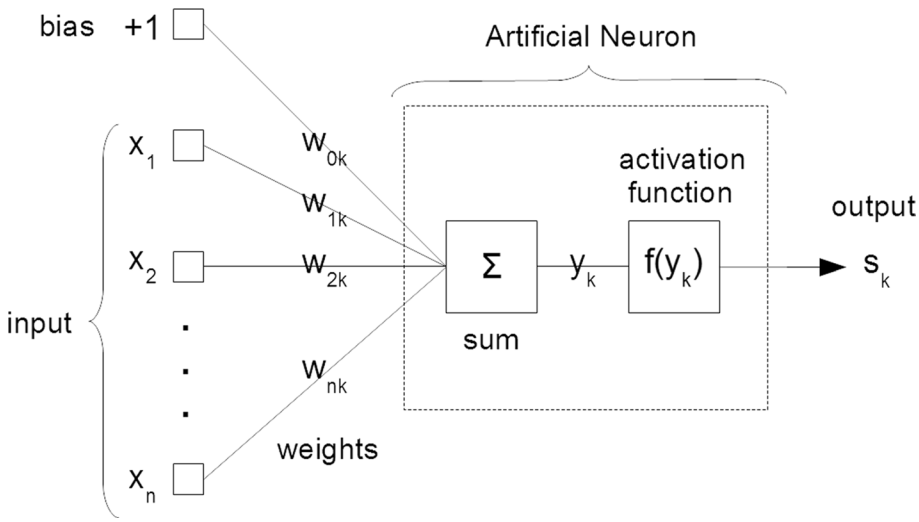

Fig. 4 Artificial neuron model (adapted from [18])

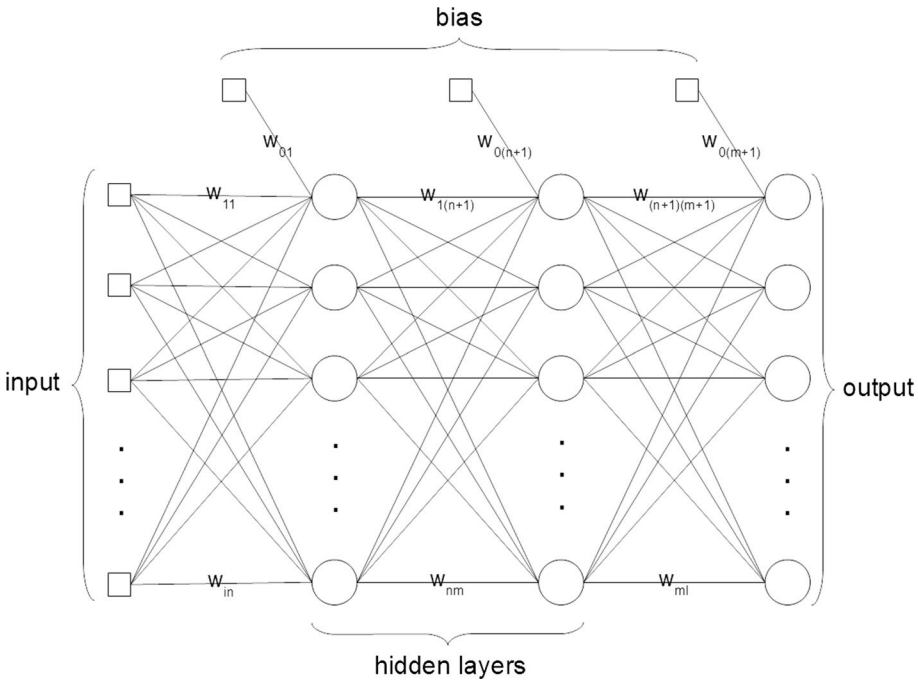

Fig. 5 Multi-layer perceptron neural network (adapted from [18])

Knowledge is acquired by the network through a training process, in which a series of data, also called training samples, are presented to the network. In a simplified way, the training process of a neural network consists of inserting into the network an input value that is transmitted by the neurons until the production of an output value. The output value is then compared to the desired response and the associated error is calculated. The error, in turn, is used to update the weights of the network, so that the output is closer to the desired value. This procedure is repeated until the total average error is less than a specified tolerance value.

This process, known as "learning", allows the network to not only reproduce the acquired knowledge but also generalize, that is, produce reasonable outputs for inputs not provided during training.

The generalization capability is essential for the application of neural networks, since even if it presents a low training error, it may not respond well to data not presented in training phase. For this reason, the minimization of the generalization error is as important as the training error.

In order to avoid overfitting, i.e. when the network is so adapted to the training data that it loses its ability to generalize (low training error but high generalization error), a procedure called cross-validation or simply validation can be used. In this procedure, the training set is partitioned in two subsets: training (or estimation) and validation. The training subset is used as input data to train the network, and the validation subset is used to test the network during training to avoid overfitting. In this way, the validation procedure can be used as a stopping criterion when it reaches a minimum. However, the validation error can have several local minima, depending on the complexity of the problem. So it is recommended to continue the training for a longer period and then select the parameters of the point where the minimum in the validation occurred.

According to Yang [24] ANNs are very useful, because firstly ANNs accurately recognize the inherent relationship between any input and output set without a physical model, and yet the results account for all the physics that relates output to input. Secondly, the approach is inherently fault-tolerant because of the large number of processing units that perform massive parallel data processing. Finally, their learning ability also allows adaptation to changes in the parameters.

4 Satellite thermal simulation using ANNs

For the simulation of satellite thermal behavior using ANNs, it is first necessary to have available data for training the ANN. They can come from experiments, or from numerical simulation, provided that the satellite numerical thermal model has been validated using experimental data. In fact, a validated detailed numerical thermal model of the satellite is more appropriate, since it can generate as many training scenarios for the ANN as desired, or necessary, what would be very costly, or even impractical, if experimental data were to be used.

Next, the ANN´s architecture is defined, along with the input, output, and internal parameters. The input data consists of the internal and external thermal loads acting on the satellite, while the output data is the temperature of chosen

points of interest, for example equipment, over the satellite. Note that both the heat loads as well as the temperatures are functions of time, since in orbit the satellite is subjected to varying thermal conditions, resulting in a time variable temperature distribution.

The following step is the ANN training, in which the weights of the network are updated in an iterative process to reduce the error in the output in comparison with the expected values.

After ANN training, the results are analyzed, and its generalization capability is verified comparing to the data not presented in training. If it is not capable of properly predict the expected results, the ANN architecture or internal parameters are redefined, until a suitable arrangement for the ANN is found. Figure 6 summarizes this process in a block diagram, and Sects. 4.1 to 4.5 details the steps for ANN training.

4.1 Input and output data

The input and output data used for training the ANN were generated by simulating the transient thermal behavior of the Amazonia-1 satellite, for a set of orbit scenarios. ThermalDesktop—SINDA/FLUINT (SINDA) satellite thermal analysis software was used for the simulations. The satellite model is the same used for the thermal design and analysis of Amazonia-1 satellite. This model was adjusted and validated by a Thermal Balance Test (TBT) at INPE´s Test and Integration Laboratory (LIT) [13].

The thermal model used with SINDA has a total of 11,806 nodes, since it is a high precision model in which the temperatures are obtained by calculating the thermal conduction between each node and its neighbors, and the radiation emitted and received by each node. In order to train the ANN only 50 nodes, in which the temperature is relevant in the Operational Simulator, were chosen. The nodes' selection criteria consisted of choosing at least one node per piece of equipment that needs its temperature to be monitored according to its thermal control requirements.

A total of 12 scenarios were generated in SINDA, two of them being the extreme Hot and Cold scenarios, and 10 intermediate scenarios, nominated from *A* to *J*, with parameters randomly defined within the nominal ranges for each variable. The Hot scenario is an operational scenario with all the equipment working at full power and with the space environment parameters at their maximum values. In the Cold scenario, in turn, the pieces of equipment are in standby mode and the environmental parameters are set to their minimum values.

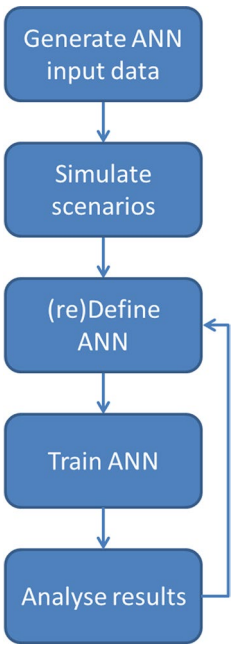

Fig. 6 ANN modeling process

4.2 ANN architecture

There are different ANN architectures and methodologies in the literature such as multi-layer perceptron networks, radial basis functions networks, support vector machines, convolutional neural networks, and deep learning neural networks. Among the existing architectures, MLP networks are the most used for function approximation and identification of dynamic systems [18, 19, 32, 41].

MLP networks have three distinctive features: a nonlinear activation function, one or more layers of hidden neurons and a high degree of connectivity. The combination of these characteristics gives these networks a significant computational power, allowing them to learn through a training process and then generalize from the acquired knowledge.

Recurrent neural networks were initially proposed for modeling time series [33–35]. The structure of the network is similar to that of a canonical MLP, with the distinction that non-feedforward connections among neurons are allowed and associated with a time delay. Through these connections the model can retain information about the past, enabling it to discover temporal correlations between events that are far away from each other in the data [36]. Additionally, it has been proved that the recurrent networks are universal approximators of nonlinear dynamic systems [17]. For this reason, in this work it was chosen to use recurrent MLP networks, and to optimize their parameters for the particular problem of satellite thermal control simulation.

During the research, it was experimented with 2 and 3 hidden layers, but better results (smaller generalization errors) were obtained with just one hidden layer, besides the input and the output layers. The ANN used has 95 input parameters in total, 1 hidden layer with 20 neurons, and 50 output neurons, as illustrated in Fig. 7. Tests with 10 to 70 neurons in the hidden layer were performed. It was observed

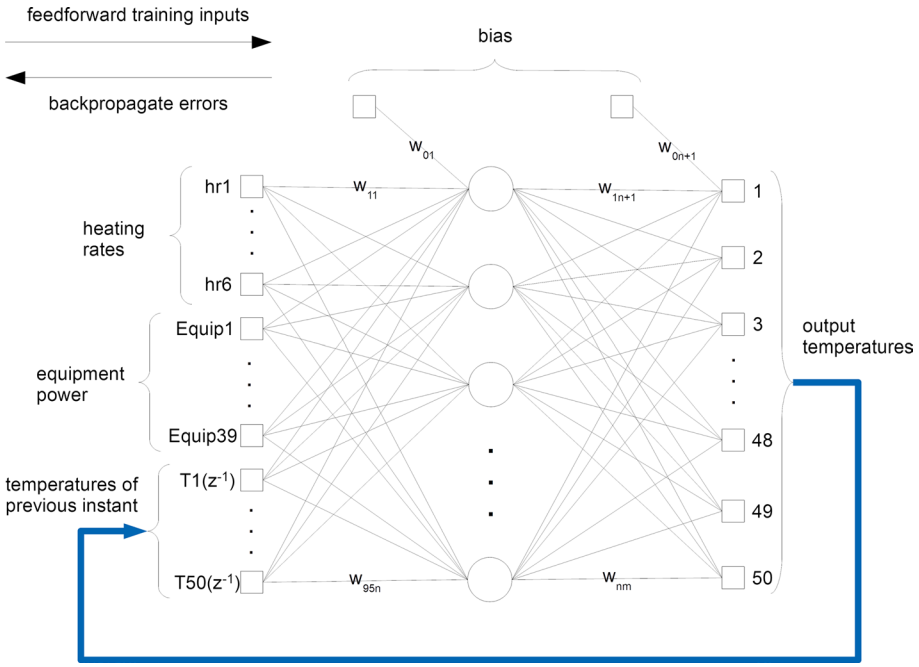

Fig. 7 ANN architecture

that with 20 neurons the ANN presented greater stability in the results and, at the same time, a lower generalization error. The variables hr1 to hr6 represent the heating rates on the 6 external surfaces of the satellite. These heating rates were calculated using SINDA, based on the values of the environmental variables (Solar, Albedo, and Earth's infrared radiations) and the satellite's attitude at each orbit position. The variables Equip1 to Equip39 refer to the power of the internal equipment selected for training and the variables $T1(z^{-1})$ to $T50(z^{-1})$ are the temperatures generated at the exit of the network for the immediately previous instant with $\Delta t = 60$ s (recurrent inputs). At the first iteration of each scenario, the temperatures of the first point in orbit (t_0) are used as the temperatures of the previous instant and the training starts with data from the second point in orbit (t_1).

4.3 ANN training

In order to train the recurrent MLP architecture, it was used the gradient descent (GD) algorithm, defined by RUMELHART et al. [33] for ANN training.

Initially, the training was performed with only the Hot and Cold scenarios. Then, scenarios *A* through *E* were gradually added to the training data set in independent runs. The training set for each scenario had 101 patterns (vectors of data), generated with SINDA in intervals of 60 s, referring to a complete orbit (6000 s) plus one point, to ensure continuity at the boundary. Thus, the initial training set (HotCold) had 202 patterns and the final training set, with the inclusion of 5 additional scenarios (HotColdABCDE) had 707 patterns.

During the training process, each line (pattern) of each scenario of the training set is presented to the network. To avoid overfitting, each scenario is presented in a random order. When the last scenario is presented, this constitutes one epoch of training.

The training process consisted of the following general steps:

1. Initialize the weights of the ANN.
2. Read the initial temperatures of the 50 nodes ($T1(z^{-1})$ to $T50(z^{-1})$) of the first point in orbit and skip to the next training pattern.
3. Insert on the input layer the heating rates (hr1 to hr6), the power of the equipment (Equip1 to Equip39), and the temperatures of the 50 nodes of the previous instant.
4. Propagate the data through the network to obtain the temperatures on the output layer.
5. Compare the temperatures with the data generated with SINDA. If the error is greater than a pre-defined value (tolerance), backpropagate the error to update the network weights (GD algorithm).
6. If the training pattern is less than or equal 101, return to step 3 (loop until complete one orbit).
7. If there is another scenario to be trained, return to step 2 (loop until all scenarios are trained).
8. At each 1000 epochs, suspend the training process and perform the validation procedure with scenarios F and G (see Sect. 4.4).
9. If there is at least one pattern with error greater than the tolerance, restart the training set to the first scenario and return to step 2 (this counts as one epoch). Otherwise,

if the network has converged or if the number of epochs reaches 10^6, finish the training.

This training process is illustrated in Fig. 8.

The order in which the scenarios were presented to the ANN was randomized at each epoch. For each training set, 10 independent runs with different random initialization of weights were performed, aiming to reduce, on average, the probability of the GD algorithm to be stuck in local minima.

The ANN internal parameters used for training were: tolerance $\tau = 10^{-5}$, learning rate $\eta = 0.01$, and momentum constant $\alpha = 0.5$. The learning rate is a parameter used in the GD algorithm to adjust the size of the changes to the weights of the network. The smaller the learning rate, the smoother will be the trajectory in weight space, at the cost of a slower rate of learning. On the other hand, the larger the learning rate, the faster will be the rate of learning, but if it is too large the network may become unstable. To avoid this instability, a momentum term can be included in the weight correction equation [18]:

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n—1) + \eta \delta_j(n) y_i(n) \quad (1)$$

where $w_{ji}(n)$ is the synaptic weight connecting neuron i to neuron j; n is the number of the iteration; α is the momentum constant; η is the learning rate; $\delta_j(n)$ is the local gradient; and $y_i(n)$ is an input signal of neuron j.

4.4 ANN validation

The validation procedure consists of, at each predefined interval of epochs (in this case 1000 epochs), to verify the error of the temperature values generated by the ANN for a set of data not applied in training. This procedure is used to avoid the so-called overfitting, which is the excessive adaptation of the network parameters to the training set, consequently harming its generalization capability [18].

The complete set of data was separated into three groups: training, validation, and test. Scenarios Hot, Cold, *A*, *B*, *C*, *D*, and *E* were used for training. Scenarios *F* and *G* were used for validation. Finally, the scenarios *F*, *G*, *H*, *I*, and *J* were selected to test the generalization capability of the ANN. Scenarios F and G were also included in the test as

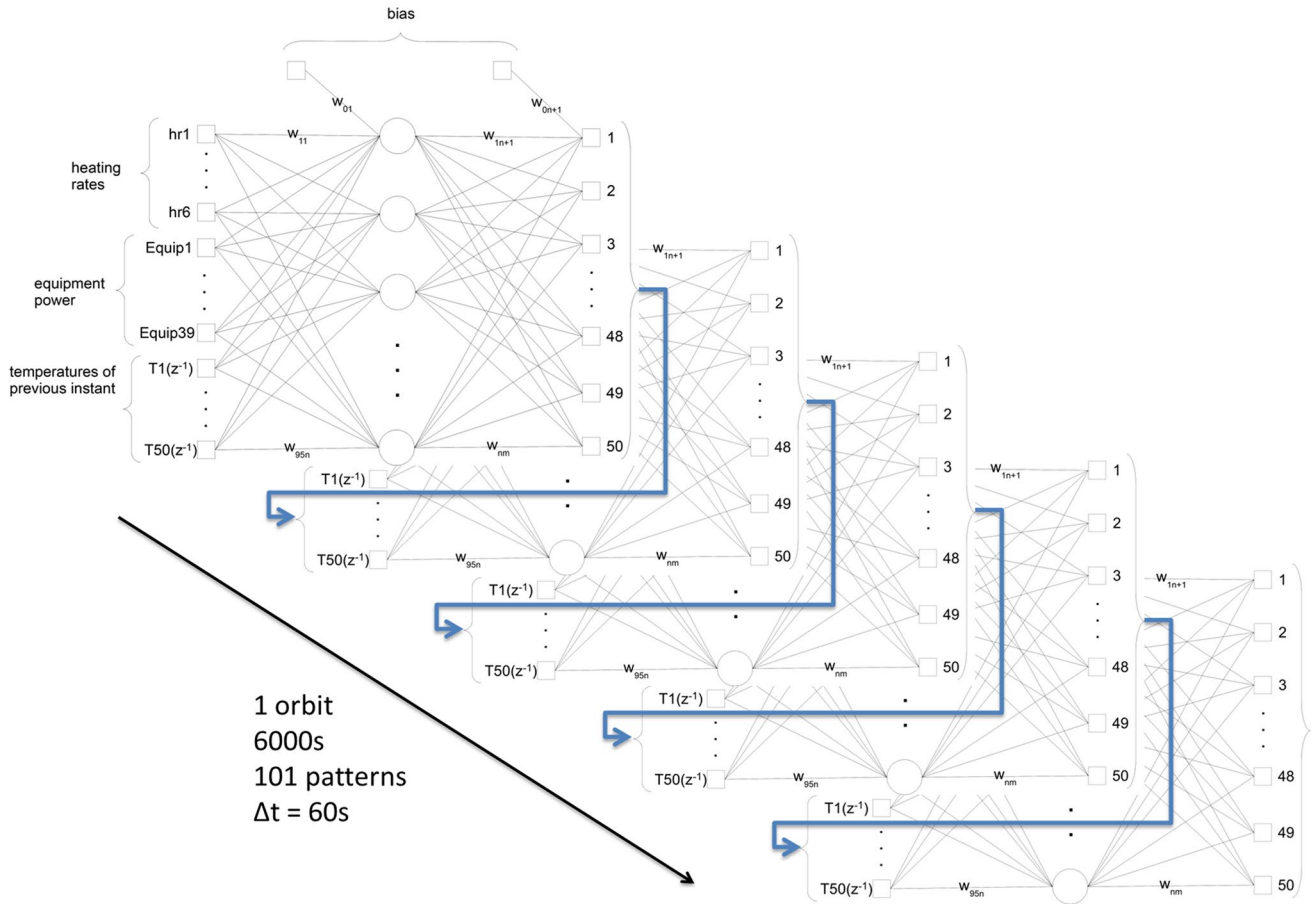

Fig. 8 Training process

they were not part of the scenarios used as input during training.

In this way, at each 1000 epochs the training process was suspended to validate the ANN, comparing the output temperatures with the expected temperatures (generated with SINDA) of the F and G scenarios. If this validation error was lower than the previous ones, the weight values were stored. At the end of training, the weights of the network that presented the lowest validation error were selected as the best weight set and then used to generate temperature curves for testing.

4.5 Additional details

The testing procedure consisted of using the trained ANN to generate temperature curves for each scenario (Hot, Cold, *A*, *B*, *C*, *D*, *E*, *F*, *G*, *H*, *I*, and *J*) and then compare with the data generated with SINDA, in order to test its capability to reproduce the training data and to generalize for data not used for training (random scenarios). Figure 9 illustrates how the network was used for testing after training.

To use the network after training, data defining the scenario and satellite's current state (heating rates, equipment power, and node temperatures), for a given point in orbit, is entered into the input layer and propagated to the output layer to generate the temperatures. In the next cycle (next point in orbit), temperatures at the exit of the previous cycle are used as input and the other variables are updated according to the satellite position in orbit and the duty cycle of the equipment.

For testing, the initial temperatures were defined as the average values between their lowest (cold case) and highest (hot case) values. First, it was cycled through the network for 10 orbits to reach the equilibrium. Then, the ANN was used to generate temperature curves for 1 orbit. Finally, these curves were compared with the data generated by SINDA.

The implementation of neural networks in this paper was carried out using C programming language and most of the training was performed on a workstation with an 8-core Intel Xenon processor. The main reason to use this language is because INPE's Operational Simulator was developed using it. So, it would be easier for their developers to understand it, especially if they aren't experts on neural networks. Besides

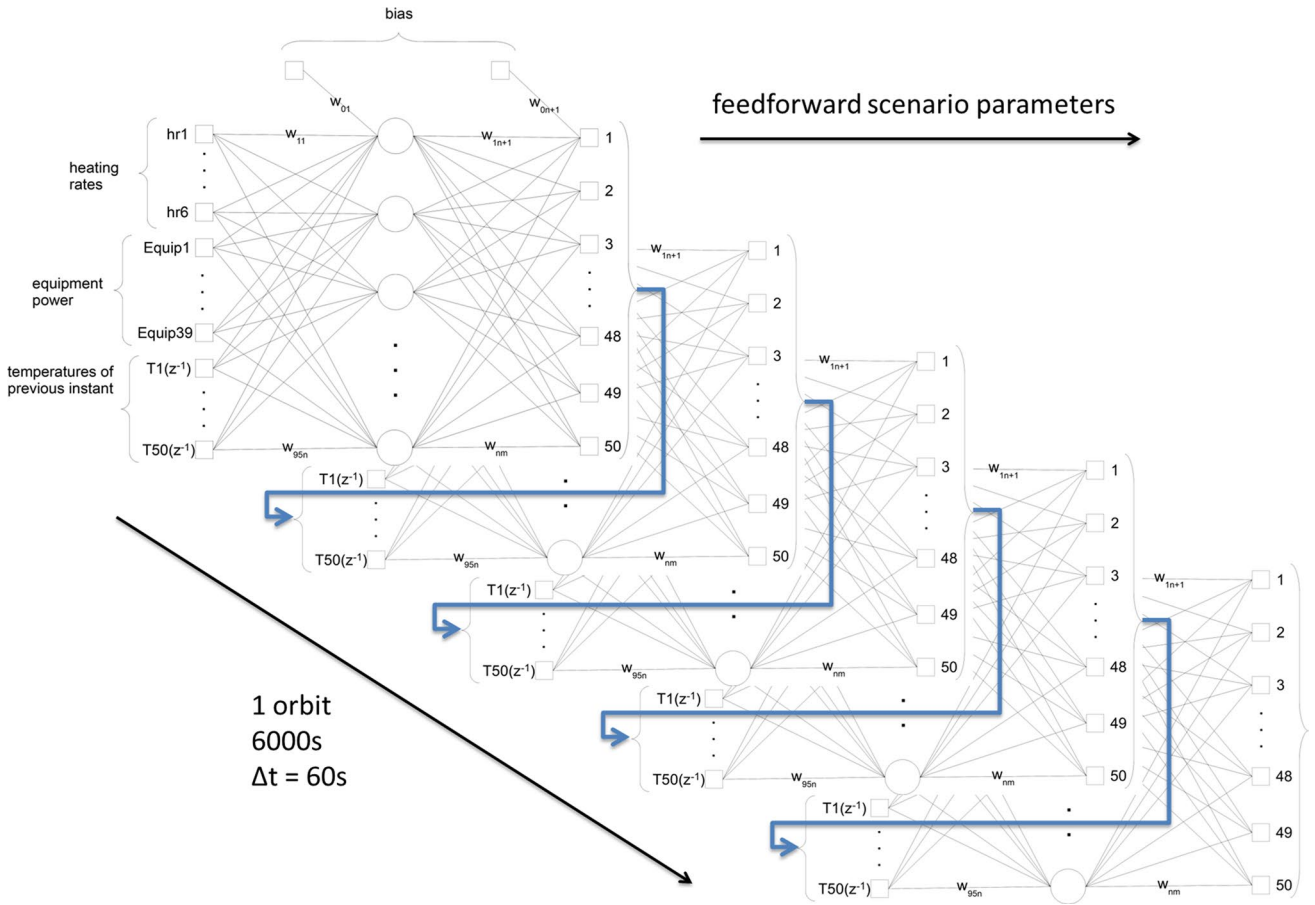

Fig. 9 Use of the ANN after training

that, C/C++ usually has performance advantages over other languages (like Python, Java, etc.). In addition, it was opted to develop the algorithms from the ground up and not use any off-the-shelf machine learning framework, as these add an additional layer of opacity, that is, lack of full knowledge of its internal processing, besides the already known opacity of neural networks [37].

Usually, in the design of a satellite thermal control subsystem, a difference of up to 5ºC between numerically calculated and experimentally obtained temperatures over the satellite is considered acceptable for validating the satellite´s numerical thermal model. This value was used in the present work as a limiting error to the difference between the temperatures calculated using SINDA and the ones predicted by the ANN.

Since the numerical thermal model has already been adjusted to the Amazonia-1′s Thermal Balance Test, the difference between it and the real satellite is very low. Moreover, after the satellite launch, the ANN can be re-trained with the telemetry data obtained from the in-orbit satellite, in order to improve its accuracy.

5 Results and discussion

This section shows the results of applying ANN to simulate the thermal behavior of the Amazonia-1 satellite.

5.1 Amazonia-1 thermal simulation with ANN

A sample of the values comprising the scenarios used for training, validation, and testing are shown in the columns of Table 1 (D, E, G, I, and J scenarios, as well as some equipment, are not shown due to space constraints).The top three rows are the external radiation sources (used to calculate the heating rates on SINDA) and the remaining rows are the equipment power. The duty cycles of the equipment are described in Table 1 notes (just below the table).

The parameters of the Hot and Cold scenarios cover just the maximum and minimum/standby operational values. These scenarios teach to the ANN the range (maximum and minimum) in which the temperatures can vary, whereas the parameters of the *A* to *H* scenarios are chosen randomly in order to teach to the ANN how to generalize in unpredictable situations. During the research, different approaches were used to guide the choice of the parameters' values, but choosing them randomly provided better generalization results.

Although some parameters on Table 1 are constant for each scenario, the heating rates (also an input parameter) on the satellite's external surfaces vary throughout the orbit due to the variation of the surface exposure to the Sun and the satellite's attitude.

The following 3 figures show a comparison between the temperature curves generated by SINDA tool and the ANN, for different sets of scenarios. In this way the influence of increasing the number of random scenarios on the ANN training can be analyzed. The idea is that, as the number of scenarios increases, the ANN will receive more information about the satellite's thermal behavior and will be able to not only reproduce the trained scenarios, but also generalize to scenarios not presented during training.

Each figure has 4 graphs that shows the results for 2 scenarios presented in the training (Hot and Cold) and 2 scenarios not presented (F and H), to analyze the ANN capability to reproduce and generalize, respectively.

The temperature curves refer to 3 pieces of equipment: TCE (Thruster Control Electronics), Gyro ICU (Gyro Inertial Control Unit) and SPE (Signal Processing Electronics). These pieces of equipment were selected because they have temperature difference high enough to allow their visualization independently in the same graph.

Figure 10 shows results for the ANN training with data of just the Hot and Cold scenarios (HotCold set).

As expected, the temperatures generated by the ANN (in blue) for the trained cases Hot and Cold present a good match with those simulated in SINDA tool (in red). However, for the untrained cases F and H, the difference between the curves is very significant (> 10 ºC). This big difference occurs because the network has acquired information from only the two most extreme scenarios (Hot and Cold) and is therefore unable to produce satisfactory results for other untrained scenarios.

Figure 11 shows the results for training the ANN with the Hot, Cold, *A*, and *B* scenarios (HotColdAB set). The curves produced by the ANN for the Hot and Cold cases continue to present a small error in comparison with those generated by the SINDA, although there are small differences in some regions. The curves for the F and H scenarios, in turn, remain distant and have some localized oscillations (noise).

The results for the training with the Hot, Cold, *A*, *B*, *C*, *D*, and *E* scenarios (HotColdABCDE set) are shown in Fig. 12. In this case, despite the small error, the curves for the trained scenarios (Hot and Cold) present some more evident differences, especially for the Cold scenario. The increased error for these trained cases, as new scenarios are included, is due to the fact that the neural network must adapt to all training data at the same time. In other words, the greater the number of scenarios presented during training, the greater the difficulty of the network adapting to each one individually.

Concerning the untrained scenarios (F and H), it is observed that the difference between the curves generated by the ANN and the SINDA was drastically reduced. The localized oscillations also decreased their intensity, but they still remain in some regions. This noise is mainly due to, in most scenarios, the equipment starts in operating mode and

Table 1 Sample of data set used for training, validation, and generalization analysis

Heat source	Scenarios						
	Hot	Cold	*A*	*B*	*C*	*F*	*H*
Solar	1418	1326	1357	1367	1355	1365	1375
Albedo	0.42	0.34	0.383	0.377	0.397	0.406	0.39
IR	233	208	209.2	221	226	215.6	221.9
ACE	22.0	12.0	13.8	14.4	21.7	18.1	18.4
Battery 1	2.8	2.3	2.61	2.8	2.54	2.59	2.76
GPS receiver	11.7	9.3	11.44	10.79	9.5	10.58	10.24
GYRO EM	30.0	21.6	26.2	22.3	29.8	29.7	29.6
GYRO ICU	0.15	0.08	0.086	0.08	0.119	0.132	0.148
LNA	0.65	0.65	0.65	0.65	0.65	0.65	0.65
Magnetotorq. 1	2.7	0.0	2.08	1.86	0.06	0.77	0.59
OBDH	32.5	32.5	32.5	32.5	32.5	32.5	32.5
PCDU*	57.0	6.3	19.6	56.0	53.3	45.8	21.8
R. wheel 1**	161.6	6.7	119.7	69.6	48.3	101.8	55.5
SADA 1	6.5	5.1	5.81	5.73	6.24	5.96	6.5
SADE	17.5	10.95	16.87	15.44	13.91	15.5	13.61
Star sensor 1	13.5	0.225	6.79	7.95	11.22	0.29	9.19
TCE	0.8	0.0	0.8	0.8	0.8	0.8	0.8
TT&C 2***	21.0	6.5	19.1	0.0	8.7	20.9	0.0
DC/DC	10.4	6.42	8.05	6.49	8.11	6.8	6.61
EPC 1***	9.0	0.1	8.1	0.0	8.2	5.2	2.0
QPSK-TX***	12.0	0.0	11.3	7.0	4.1	11.4	3.2
RTU 1	5.7	5.3	5.5	5.5	5.5	5.5	5.5
SDC***	5.3	0.0	4.5	2.4	2.8	0.1	2.9
SPE****	57.3	6.7	48.1	30.0	34.7	12.5	55.0
SSR	10.8	10.08	10.8	10.8	10.8	10.8	10.8
TWT 1***	45.0	0.0	0.0	18.7	0.0	7.2	37.2

The Solar and IR radiations are measured in W/m2, the albedo is a fraction the Solar radiation, and the power of all equipment is measured in Watts (W)

Nominal values: TT&C2 = 21.0 W; EPC1 = 9.0 W; QPSK-TX = 12.0 W; SDC = 5.3 W; TWT1 = 45.0 W

Standby values: TT&C2 = 6.5 W; EPC1 = QPSK-TX = SDC = TWT1 = 0.0 W

*starts at displayed value and decreases to 15.8 W at t = 3980 s

**starts at displayed value, decays to 26.8 W at t = 180 s, returns to the value shown at t = 720 s, and decays again to 26.8 W in 900 s

***start at their nominal value (see below) and decay to standby at $t = 1200$ s

****starts at displayed value and decreases to 6.7 W at t = 1500 s

in specific points it turns to standby mode. This makes it more difficult for the network to reproduce the temperature curves smoothly, but it is much more realistic than keeping the equipment turned on (or in standby) throughout the whole orbit.

The results for the HotColdA, HotColdABC, and HotColdABCD data sets are not shown to avoid excessive repetitions.

The temperature errors calculated as the difference of the temperatures obtained with the ANN and with SINDA for the F, G, H, I, and J scenarios, not used as training input, are called generalization errors. Table 2 contains the mean generalization errors of the temperatures generated by the ANN trained with the HotColdABCDE data set, for the 10 independent weight initialization runs. The values were calculated as the average for the full orbit and for the 39 equipment. The Studentized Range is the difference between the largest and the smallest E_{mg} divided by the standard deviation.

In order to analyze the evolution of the generalization error with the inclusion of the additional scenarios (*A*, *B*, *C*, *D*, and *E*), in Table 3 it is shown the values of the mean generalization errors (E_{mg}) for each data set. The values were calculated as the average of the 10 independent weight initialization runs. In other words, there were 6 tables, similar to Table 2, one for each data set (HotCold, HotColdA,

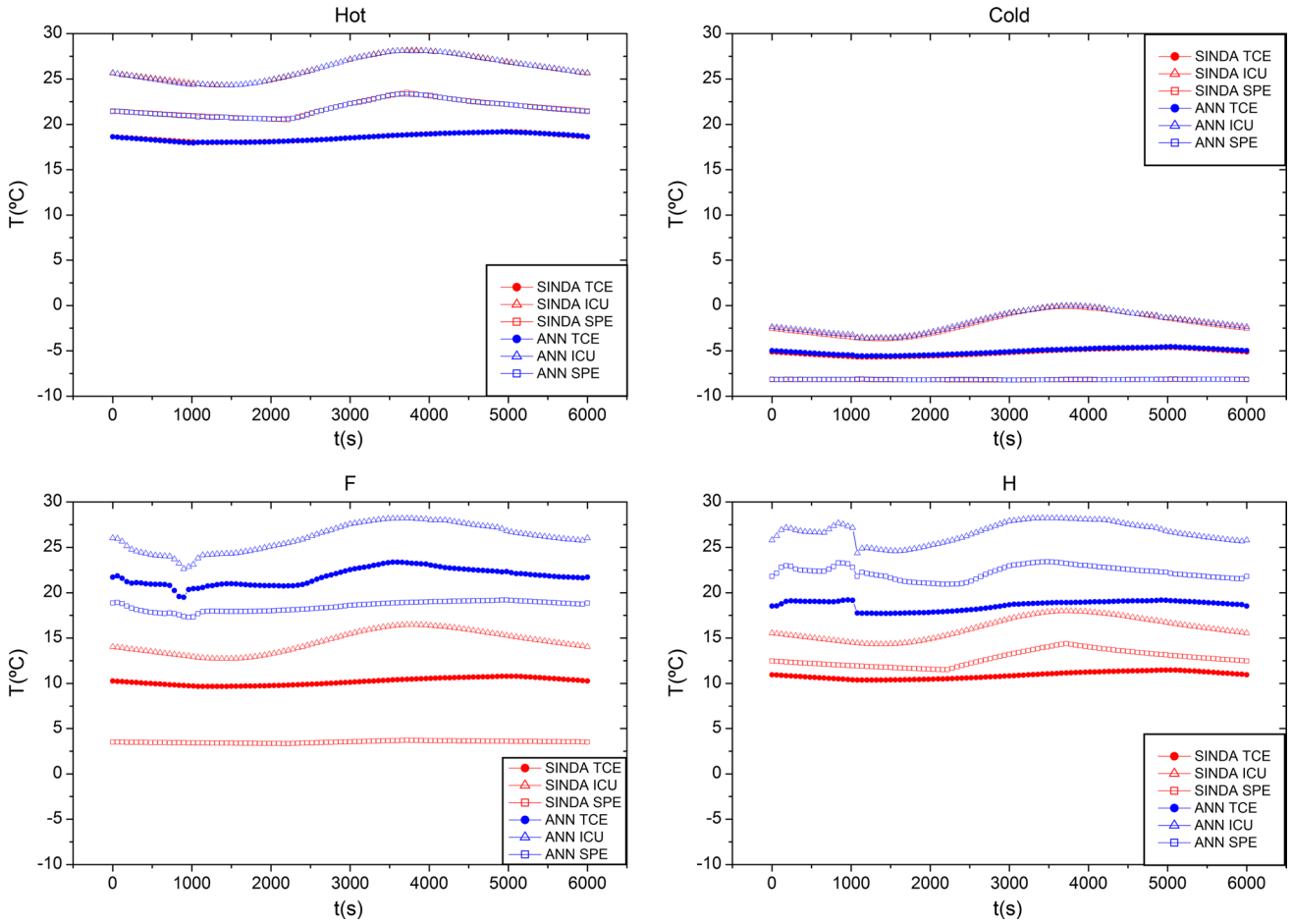

Fig. 10 Comparison of the temperature curves generated by SINDA and the ANN trained with the HotCold set (TCE, ICU, SPE)

HotColdAB, HotColdABC, HotColdABCD, and HotColdABCDE), and the last row of each of those tables was used to build Table 3.

Figure 13 shows the evolution of the mean generalization error as a function of the number of random scenarios added in training. The graph was constructed with the data shown in the last column of Table 3. The solid lines are only guides to the eye, and the dashed line represents the limit of 5ºC, referred to the requirement that the generalization error should be less than this temperature value.

Despite the abrupt decline with the inclusion of the first random scenario, E_{mg} was not reduced below 5ºC. This occurred only after the addition of the third random scenario, and the E_{mg} decayed to less than 4ºC after the inclusion of the fifth random scenario.

As mentioned before, this result refers to the mean generalization error obtained over 10 independent weight initializations. This means that in some training runs it was possible to get even lower errors. As in a real application one would use the weights obtained with just one independent run. In Table 4 it is shown the best (lowest) mean generalization errors, obtained for the best training run for the HotColdABCDE set (Run 4), as well as the maximum errors (E_{max}) observed for this specific run.

The overall mean of the best E_{mg} (Run 4) obtained for the untrained scenarios was 2.3ºC, well below the requirement of 5ºC, with a standard deviation of less than 1.0ºC. This means that the temperatures generated by ANN are very close to the expected values (generated by SINDA).

Besides that, some larger localized errors were observed, being 18.0ºC the greatest error observed for one of the nodes in scenario H. In order to further analyze these maximum errors, in Table 5 it is shown the maximum generalization temperature errors for the equipment shown in Table 1 (except the LNA because its temperature is not part of the 50 network output nodes), on the best run (Run 4) for scenarios F, G, H, I, and J.

As can be seen in Table 5, the maximum generalization errors for some nodes are higher than 10ºC, reaching the limit of 18.0ºC. Although these values are not desirable, in most cases the maximum errors are below or near 5ºC. This higher error occurred due to the presence of the observed

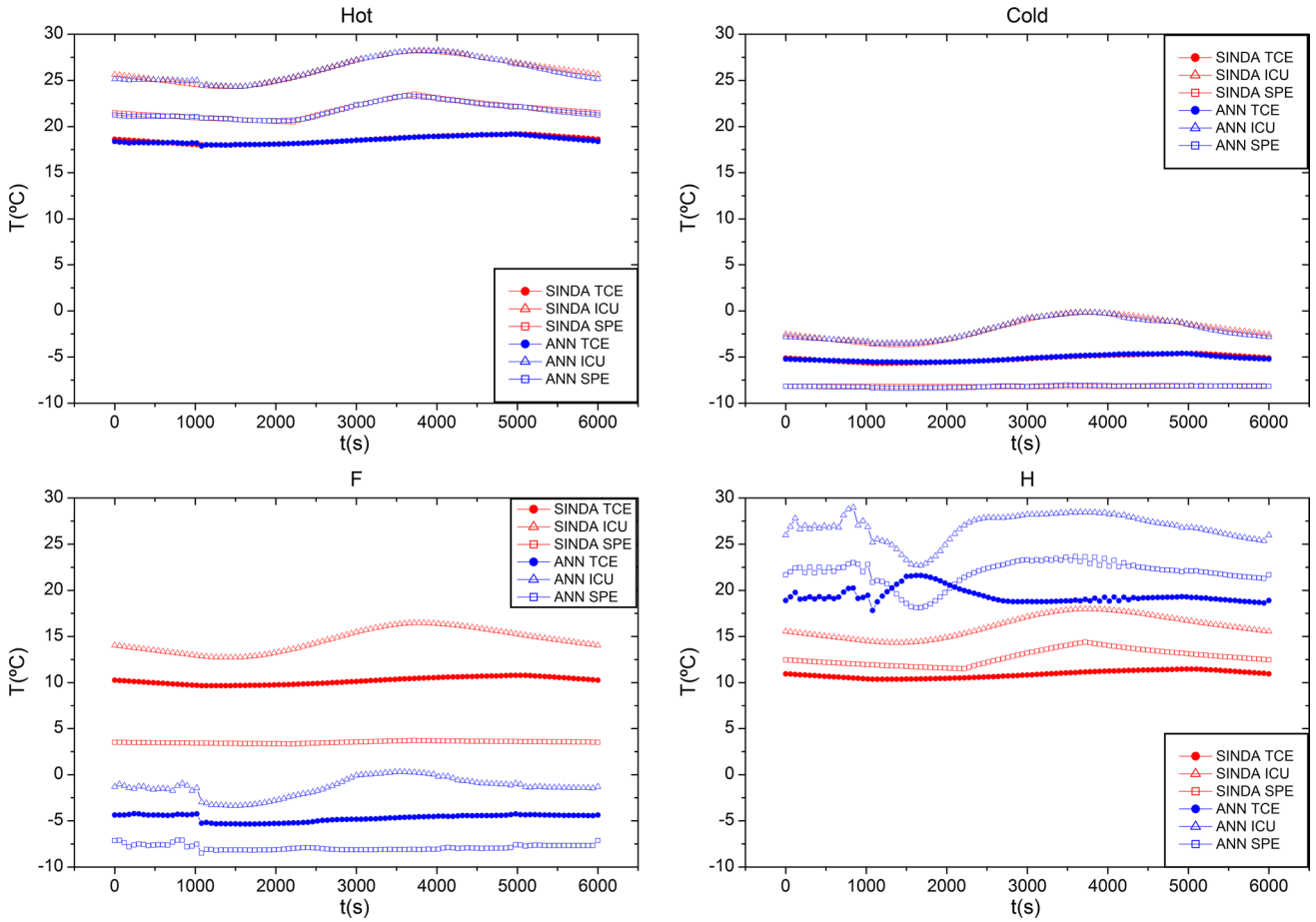

Fig. 11 Comparison of the temperature curves generated by SINDA and the ANN trained with the HotColdAB set (TCE, ICU, SPE)

noise in the curves generated by the ANN, which is related to the complexity of the information that the neural network needs to acquire, considering the thermal behavior of 50 nodes for several different scenarios, associated with the change in the equipment operation modes during orbit. Noting that these results refer to scenarios not used as input during training, in the conclusion chapter, possible approaches to improve these results are discussed.

5.2 Execution time evaluation

As pointed out previously, the satellite thermal model in the Operational Simulator must provide the temperature variation in a time interval short enough for the Simulator to generate the telemetries and to execute as a whole in real time. For this reason, run-time tests were performed on different personal computers to verify the processing time required for the ANN to generate 5 orbits for the Amazonia-1 satellite (the choice of 5 orbits was to obtain a minimum significant average). The results of these tests are shown in Table 6.

Although only the processor name is listed, other factors may also influence the running time such as the number of simultaneous running programs, hard disk and memory speed, and operating system.

The values shown in the table refer to the average time for the ANN to generate 5 orbits (equivalent to 500 points), calculated from several independent runs. This means that the ANN can provide the temperatures of 50 different points in the satellite for 5 orbits at around 0.1 s.

Since the satellite's thermal behavior changes very slowly, the temperatures can be updated at a time interval like every 10 s on the Operational Simulator. This gives more than enough time for the ANN to generate the temperatures.

By comparison, the time required for SINDA to generate the same 5 orbits for a specific scenario of the Amazonia-1 satellite can take up to 10 h on a personal computer.

6 Conclusions

In this paper, it is shown how artificial neural networks can be used to simulate with good accuracy the thermal behavior of a satellite in orbit. The use of a trained ANN makes possible the estimation of temperatures at predefined satellite

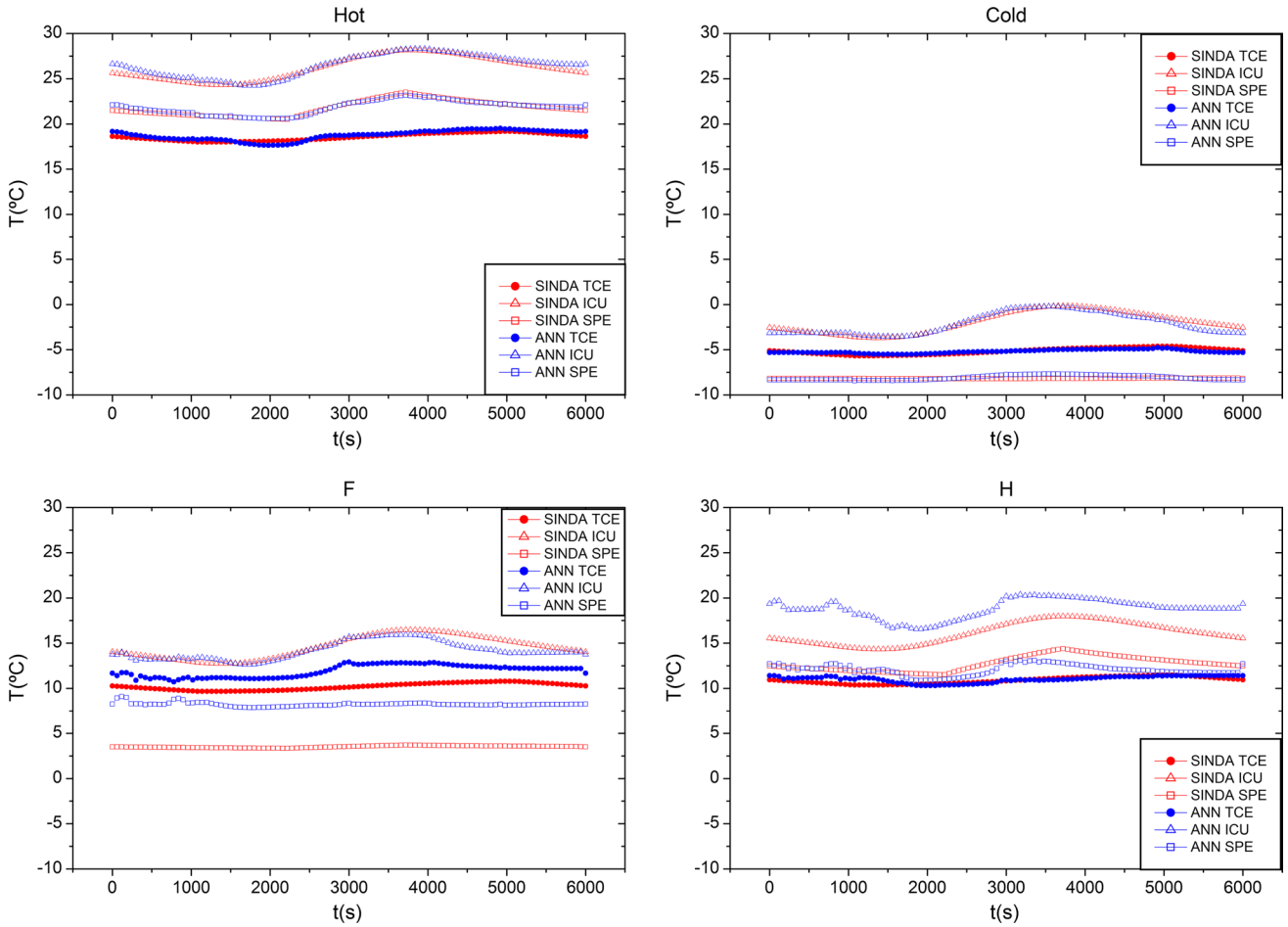

Fig. 12 Comparison of the temperature curves generated by SINDA and the ANN trained with the HotColdABCDE set (TCE, ICU, SPE)

Table 2 Mean generalization errors of the 10 independent runs for the HotColdABCDE data set

Run No	E_{mg} (ºC)					Total mean	Standard deviation	Studentized range
	F	*G*	*H*	*I*	*J*			
Run 1	13.9	3.3	3.3	2.6	12.8	7.2	5.7	2.0
Run 2	10.9	1.9	3.0	2.4	7.3	5.1	3.9	2.3
Run 3	3.8	1.6	4.9	2.5	1.8	2.9	1.4	2.4
Run 4	3.5	1.8	1.7	2.4	2.0	2.3	0.7	2.5
Run 5	4.3	2.2	8.6	2.8	1.9	4.0	2.8	2.4
Run 6	3.5	2.0	2.2	3.0	1.9	2.5	0.7	2.3
Run 7	4.5	2.0	2.9	2.2	1.5	2.6	1.2	2.6
Run 8	4.5	1.9	2.7	3.2	1.7	2.8	1.1	2.5
Run 9	2.9	1.5	14.6	2.8	15.1	7.4	6.8	1.9
Run 10	4.1	2.1	2.1	2.9	1.9	2.6	0.9	2.4
Total mean	5.6	2.0	4.5	2.7	4.8	3.9	1.5	2.4

Table 3 Evolution of the mean generalization errors with the inclusion of additional scenarios

Data set	E_{mg} (ºC)					Total mean	Standard deviation	Studentized range
	F	*G*	*H*	*I*	*J*			
HotCold	11.8	10.1	9.4	7.3	8.2	9.4	1.7	2.6
HotColdA	8.3	4.1	5.4	4.8	5.0	5.5	1.6	2.6
HotColdAB	5.4	8.6	10.4	6.5	7.8	7.8	1.9	2.6
HotColdABC	5.5	4.7	2.2	9.4	2.4	4.9	2.9	2.5
HotColdABCD	4.5	4.2	5.4	5.3	4.7	4.8	0.5	2.3
HotColdABCDE	5.6	2.0	4.5	2.7	4.8	3.9	1.5	2.4

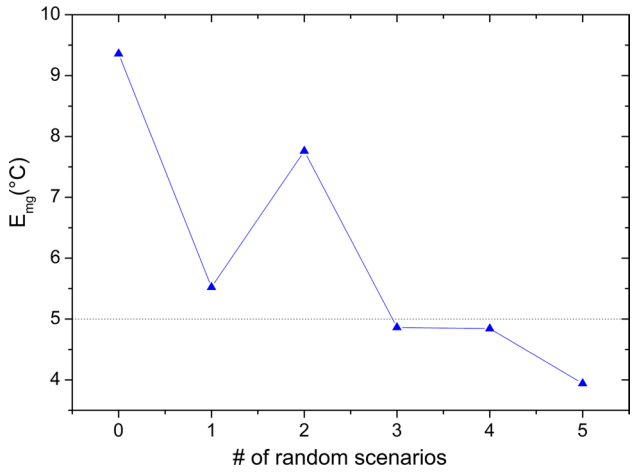

Fig. 13 Mean generalization error as a function of the number of random scenarios added in training

Table 4 Best mean generalization errors (E_{mg}), standard deviation, and maximum generalization errors (E_{max}) for the best training run (Run 4)

Scenario	Best E_{mg} (ºC)	Standard deviation	E_{max} (ºC)
F	3.5	0.5	11.9
G	1.8	0.4	8.9
H	1.7	0.8	18.0
I	2.4	0.7	8.1
J	2.0	0.4	10.2
Overall mean	2.3		

internal locations, fast enough to be used in a real-time simulator.

As a case study, the proposed solution was applied to the estimation of on orbit temperatures of the Earth observation Amazonia-1 satellite, currently being developed in Brazil.

The results obtained with the ANN show low mean generalization errors, which represents a good accuracy in comparison with the data simulated with SINDA, although some larger localized errors were observed.

Table 5 Maximum generalization errors per equipment for the best run

Equipment	*F* E_{max} (ºC)	*G* E_{max} (ºC)	*H* E_{max} (ºC)	*I* E_{ax} (ºC)	*J* E_{max} (ºC)
ACE	4.2	1.6	0.8	3.3	4.0
Battery 1	1.8	1.2	2.4	2.1	3.7
GPS receiver	3.2	2.0	1.4	1.9	2.3
GYRO EM	2.6	1.4	4.0	4.1	2.3
GYRO ICU	1.4	1.0	4.9	3.7	1.9
Magnetotorq. 1	1.5	0.8	1.7	2.1	2.1
OBDH	2.1	1.2	1.2	1.6	2.4
PCDU	8.7	4.5	1.9	5.5	5.1
R. wheel 1	8.1	4.8	18.0	8.1	6.7
SADA 1	2.5	1.2	1.1	1.7	1.4
SADE	1.8	0.4	1.9	0.5	1.7
Star sensor 1	2.2	1.3	5.6	4.6	1.6
TCE	2.7	1.1	0.8	2.2	2.7
TT&C 2	1.4	2.2	8.2	4.9	2.7
DC/DC	7.8	3.0	0.4	4.3	1.1
EPC 1	11.3	5.2	3.3	4.5	2.2
QPSK-TX	7.4	2.8	1.5	3.5	1.0
RTU 1	6.4	3.2	1.4	3.3	0.9
SDC	7.5	2.9	1.3	3.4	1.6
SPE	9.4	7.0	1.6	3.5	1.0
SSR	6.7	2.5	1.4	3.4	1.2
TWT 1	11.9	8.9	16.1	6.7	10.2

Table 6 Mean time to generate 5 orbits

#	Processor	Mean time (s)
1	Intel Core i5 M 460	0,063
2	Intel Core i5 750	0,115
3	Intel Core i5-4590S	0,084
Overall mean		0,088

In addition, the results showed that the processing time required for ANN to generate the thermal behavior of the Amazonia-1 satellite is about 5 orders of magnitude shorter than that required for SINDA.

Besides that, it must be said that the computational cost for obtaining the trained ANN may be high, in the case of the present work on the order of 100 h. Nevertheless, this is worth the effort since it is done just once, and after that the trained ANN can be used to simulate very quickly the satellite thermal behavior.

As future work, in order to reduce the observed maximum errors, the authors intend to explore the following strategies: (i) reduce the time interval from 60 s down to 1 s, consequently increasing the number of training patterns for each scenario; (ii) increase the number of scenarios presented in training; and (iii) use data from the two or three previous instants at the input of the ANN, to acquire long term information.

Authors' contributions J.D. Reis Junior, A.M. Ambrosio, and F.L. de Sousa contributed to the study conception and design. D.F. Silva performed thermal simulations and provided the input data. The custom code and ANN design and training were performed by J.D. Reis Junior. The analysis of the results was performed by all authors. The first draft of the manuscript was written by J.D. Reis Junior and all authors commented and revised previous versions of the manuscript. All authors read and approved the final manuscript.

Funding The research received no direct funding.

Code availability For the artificial neural networks computations it was used custom code written in C language. The training data was generated with ThermalDesktop—SINDA/FLUINT thermal software package version 4.8, running with AutoCad 2005.

Compliance with ethical standards

Conflicts of interest There are no conflicts of interest associated with this research. Also, there's no financial/personal interest or belief that could affect its objectivity.

Availability of data and material This research is part of a Ph.D. thesis, conducted at the Brazilian National Institute for Space Research (INPE). The results were not published or submitted to any publication other than the Ph.D. thesis. All data used in the research were generated by the authors. Some figures were adapted from other sources by the authors (references are provided in the captions), while others were produced by the authors or are property of INPE.

References

1. Smahat A, Mankour A, Slimane S, Roubache R, Bendine K, Guelailia A (2020) Numerical investigation of debris impact on spacecraft structure at hyper-high velocity. J Braz Soc Mech Sci Eng 42:117. https://doi.org/10.1007/s40430-020-2196-7
2. Hendricks R, Eickhoff J (2005) The significant role of simulation in satellite development and verification. Aerosp Sci Technol 9:273–283. https://doi.org/10.1016/j.ast.2004.12.006
3. Eickhoff J (2009) Simulating spacecraft systems. Springer, Berlin. https://doi.org/10.1007/978-3-642-01276-1
4. Bodin P, Nylund M, Battelino M (2012) SATSIM - A real-time multi-satellite simulator for test and validation in formation flying projects. Acta Astronaut 74:29–39. https://doi.org/10.1016/j.actaastro.2011.11.015
5. Irvine M, Fritzen P, Ellsiepen P (2013) Trends in european space simulation: standards, architectures and tools across the mission lifecycle. In: RAST 2013—proceedngs 6th international conference on recent advances in space technologies, pp 112: 1163–1167. https://doi.org/https://doi.org/10.1109/RAST.2013.6581179
6. Chagas RAJ, de Sousa FL, Louro AC, dos Santos WG (2019) Modeling and design of a multidisciplinary simulator of the concept of operations for space mission pre-phase A studies. Concurr Eng Res Appl 27:28–39. https://doi.org/10.1177/1063293X18804006
7. European Cooperation for Space Standardization (2010) ECSS-E-TM-10-21A: System modeling and simulation. Noordwijk, The Netherlands
8. Rainey LB, Davis PK (2004) Space Modeling and simulation roles and applications throughout the system life cycle. The Aerospace Press, El Segundo
9. Tominaga J, Cerqueira C, Kono J, Ambrosio A (2012) Specifying Satellite Behavior for an Operational Simulator. In: Proceedings of the workshop on simulation and egse for space programmes
10. Reggestad V, Pecchioli M, Merri M (2011) Virtual reality for real operations: Developing and using operational simulators. Eur Sp Agency Bull 148:42–51
11. Reggestad V, Pantoquilho M, Werner D, Antoniou P (2012) Increasing Performance of ESA Operational Spacecraft Simulators. In: Proceenings of the Workshop on simulation and EGSEE facilities for space programmes
12. Rotondi Azevedo D, Ambrosio AM, Vieira M (2013) HLA middleware robustness and scalability evaluation in the context of satellite simulators. Proceedings of the IEEE Pacific rim international symposium on dependable computing—PRDC, pp 312–317. https://doi.org/10.1109/PRDC.2013.53
13. da Silva DF, Muraoka I, Garcia EC (2014) Thermal control design conception of the Amazonia-1 satellite. J Aerosp Technol Manag 6:169–176. https://doi.org/10.5028/jatm.v6i2.320
14. C&R TECHNOLOGIES (2013) ThermalDesktop: CAD-based thermal engineering tool suite with SINDA/FLUINT thermal/fluid solver. Version 4.8. https://crtech.com/
15. Perpiñán MAC (1994) The Modelling of the thermal subsystem in spacecraft real time simulators. Proc Work Sim Eur Sp Prog 3:69–78
16. Manon F, Baroukh J, Mas G, Pasquier H, Toussaint F, Courtes H, Deschamps S (2018) A novel concept for thermal simulation in spacecraft simulators. In: 15th International Conference on SpaceOps conference, pp 1–14. https://doi.org/https://doi.org/10.2514/6.2018-2419
17. Lo ZP, Yu Y, Bavarian B (1993) Analysis of the convergence properties of topology preserving neural networks. IEEE Trans Neural Networks 4:207–220. https://doi.org/10.1109/72.207609
18. Haykin S (2009) Neural networks and learning machines. Pearson education 1:1–2.
19. Hagan MT, Demuth HB, Beale MH, de Jesús O (2014) Neural network design. 2 edn, eBook
20. Martinez-Heras JA, Donati A (2004) Artificial neural networks in support of spacecraft thermal behaviour modelling. IEEE Aerosp Conf Proc 2:1269–1274. https://doi.org/10.1109/AERO.2004.1367724

21. El-Madany HT, Fahmy FH, El-Rahman NMA, Dorrah HT (2011) Spacecraft power system controller based on neural network. Acta Astronaut 69:650–657. https://doi.org/10.1016/j.actaastro.2011.05.028
22. Pérez D, Wohlberg B, Lovell TA, Shoemaker M, Bevilacqua R (2014) Orbit-centered atmospheric density prediction using artificial neural networks. Acta Astronaut 98:9–23. https://doi.org/10.1016/j.actaastro.2014.01.007
23. Bu X, Wu X, He G, Huang J (2016) Novel adaptive neural control design for a constrained flexible air-breathing hypersonic vehicle based on actuator compensation. Acta Astronaut 120:75–86. https://doi.org/10.1016/j.actaastro.2015.12.004
24. Yang KT (2008) Artificial neural networks (ANNs): a new paradigm for thermal science and engineering. J Heat Transfer 130:093001. https://doi.org/10.1115/1.2944238
25. Tahan M, Muhammad M, Karim ZAA (2017) A multi-nets ANN model for real-time performance-based automatic fault diagnosis of industrial gas turbine engines. J Braz Soc Mech Sci Eng 39:2865–2876. https://doi.org/10.1007/s40430-017-0742-8
26. Mohanraj M, Jayaraj S, Muraleedharan C (2015) Applications of artificial neural networks for thermal analysis of heat exchangers—a review. Int J Therm Sci 90:150–172. https://doi.org/10.1016/j.ijthermalsci.2014.11.030
27. Reis Junior JD, Ambrosio AM, de Sousa FL (2015) Towards spacecraft real-time thermal simulation with artificial neural networks. In: Proceedings of the 23rd ABCM international congress of mechanical engineering
28. Reis Junior JD, Ambrosio AM, de Sousa FL (2016) Real-Time Cubesat Thermal Simulation using Artificial Neural Networks. Proc of the 4th international Conferenceon high performance compilation intermediate with science
29. Reis Junior JD (2018) Simulação do comportamento térmico de satélites em tempo real com uso de redes neurais artificiais. Doctoral thesis, National Institute for Space Research (INPE). http://urlib.net/rep/8JMKD3MGP3W34R/3SFU6RH
30. National Institute for Space Research (2020) Satélite Amazonia-1 e Plataforma Multimissão (PMM). http://www3.inpe.br/amazonia-1. Accessed 17 July 2020
31. Minsky M, Papert S (1969) Perceptrons. MIT Press, Cambridge
32. Ojha VK, Abraham A, Snášel V (2017) Metaheuristic design of feedforward neural networks: a review of two decades of research. Eng Appl Artif Intell 60:97–116. https://doi.org/10.1016/j.engappai.2017.01.013
33. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323:533–536. https://doi.org/10.1038/323533a0
34. Werbos PJ (1988) Generalization of backpropagation with application to a recurrent gas market model. Neural Netw 1:339–356. https://doi.org/10.1016/0893-6080(88)90007-X
35. Elman JL (1990) Finding structure in time. Cogn Sci 14:179–211. https://doi.org/10.1016/0364-0213(90)90002-E
36. Pascanu R, Mikolov T, Bengio Y (2013) On the Difficulty of Training Recurrent Neural Networks. In: Proceedings of the 30th international conference on machine learning, vol 28, pp III–1310--III–1318. http://dl.acm.org/citation.cfm?id=3042817.3043083
37. Burrell J (2016) How the machine 'thinks': Understanding opacity in machine learning algorithms. Big Data Soc 3:1–12. https://doi.org/10.1177/2053951715622512
38. Levenberg KA (1944) Method for the solution of certain nonlinear problems in least squares. Q Appl Math 2:164–168
39. Marquardt DW (1963) An algorithm for least-squares estimation of nonlinear parameters. J Soc Ind Appl Math 11:431–441
40. De Sousa FL, Ramos FM (2002) Function optimization using extremal dynamics. Proceedings of the 4th international conference of inverse problem in engineering
41. Chojaczyk AA, Teixeira AP, Neves LC, Cardoso JB, Guedes Soares C (2015) Review and application of artificial neural networks models in reliability analysis of steel structures. Struct Saf 52:78–89. https://doi.org/10.1016/j.strusafe.2014.09.002