A Matlab code to fit periodic data

Ruth da Silva Brum[1]
Jairo Valões de Alencar Ramalho[2]
Luiz Alberto Oliveira Rocha[1]
Liércio André Isoldi[3]
Elizaldo Domingues dos Santos[3]

Abstract: *This paper presents a computer method to find the best sine-based function, in the sense of least squares, to fit periodic data. Even though the least squares method is not a novelty, there is a void in the literature about its use to find trigonometric functions, particularly when it gives rise to nonlinear systems, as it is done in this article. The respective code, implemented in the Matlab programming language, is detailed and analyzed exploring experimental data from the air and soil temperatures measured along the year in earth air heat exchangers (EAHE) built in the facilities of a case study house in the Brazilian state of Rio Grande do Sul. The fitting curves have been employed by the authors in different works to define boundary conditions to study new computer models of EAHE.*

Keywords: *Earth air heat exchanger. Least squares. Trigonometric curve fitting.*

Resumo: Este artigo apresenta um método computacional para encontrar uma função do tipo senoidal que melhor se adequa, no sentido dos mínimos quadrados, a um conjunto de dados periódicos. Embora o método dos mínimos quadrados não seja uma novidade, foi percebida uma lacuna na literatura relativa ao seu uso para a obtenção de funções trigonométricas, particularmente envolvendo sistemas não lineares, como é feito neste trabalho. O código proposto, escrito na linguagem de programação do Matlab, é detalhado e avaliado por meio da exploração de dados empíricos e numéricos das temperaturas do ar e do solo medidos em trocadores de calor solo-ar (TCSA) instalados em uma casa experimental no interior do estado brasileiro do Rio Grande do Sul. As curvas ajustadas estão sendo empregadas pelos autores em diferentes pesquisas na definição de condições de contorno para novos modelos computacionais de TCSA.

Palavras-chave: Ajuste de funções trigonométricas. Mínimos quadrados. Trocadores de calor solo-ar.

1 Introduction

To face the increasing energy consumption, it is necessary to consider not only alternative/renewable energy sources and devices, but also to review the building construction models. As a remarkable example, in the summer of 2014, there was a change in the electrical energy demand peak in Brazil. It occurred at noon, instead of late afternoon, due to a heavy use of air conditioners during a heat wave [1].

In this context, the Earth Air Heat Exchangers (EAHE) stand out as a natural solution to improve air conditioning at low energy cost. In general, EAHE are composed by a system of fans and buried ducts. The fans blow the air inside the ducts, where it exchanges heat with the soil and returns at a milder temperature [2, 3, 4].

[1] Programa de Pós-Graduação em Engenharia Mecânica, UFRGS, Porto Alegre (RS), Brasil.
{ruth.silva.brum@ufpel.edu.br, luizrocha@mecanica.ufrgs.br}
[2] Departamento de Matemática e Estatística, UFPel, Capão do Leão (RS), Brasil.
{jairo.ramalho@ufpel.edu.br}
[3] Programa de Pós-Graduação em Modelagem Computacional, Furg, Rio Grande (RS), Brasil.
{liercioisoldi,elizaldosantos@furg.br}

http://dx.doi.org/10.5335/rbca.2015.4618

All this happens because the soil acts as a seasonal thermal reservoir, storing heat during summer and releasing it to the atmosphere along the winter. Hence, in the winter, the soil is warmer than the external air, what allows heating the air inside buried ducts. On the contrary, in the summer, the soil is colder than the environment air, reducing the air temperature inside the ducts [2].

On the other hand, to develop new computer models for EAHE it is necessary to know experimentally the air and soil temperatures of a given place. In particular, such states as the Rio Grande do Sul (RS) in Brazil, where the four seasons are well defined, it is important to measure this data for at least one year, as it is done in [2].

After plotting the results in a graph, it is noticeable that the annual temperatures in RS are periodic following the pattern of a translated sine (or cosine) curve in the interval $I = [0, 2\pi]$. Thus, the year begins in summer when the environment air temperatures reach their peak. Between March and June, the temperatures drop slowly (which characterizes the fall) to reach a minimum between June and September (in the winter). From September to December the temperatures go rising again (in the spring), and the year ends as it started with high summer temperatures. The soil temperatures follow an analogous behavior to the air ones, although there is a lag between them which depends on the soil depth.

Given that, it is very natural to look for the “best” sine-based function which represents (models) the air or soil temperatures along the year. To find it, this work employed the experimental data from [2], where the temperatures were measured daily, hour by hour. However, choosing to adopt the term “best” in the sense of least squares, the authors found a gap in classical literature [5-10] concerning the expected trigonometric fitting.

To be more specific, there is a lack of texts exploiting the problem of finding a real continuous function $f(x) = a\ \mathrm{sen}(bx+c)+d$ (with a, b, c, and d being real constants) to fit, by the least squares method, a set of n ordered pairs $U=\{(x_i\ , y_i),\ i = 1, 2, \ldots, n\}$. Despite using a trigonometric function, instead of a high order polynomial, brings the drawback of solving a non-linear system of equations to find the coefficients a, b, c, and d, this research found the Newton method very effective to circumvent this matter in all the simulated cases.

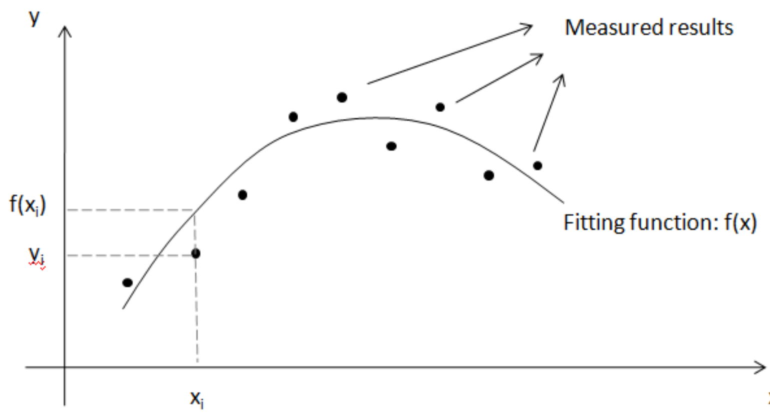
In general, this paper addresses two objectives: fill a void in the literature about trigonometric fitting; illustrate its application based on a computer code which can help the study and simulation of EAHE.

2 Methodology

2.1 Least Squares Method

Considering a set of ordered pairs $\{(x_i, y_i),\ i = 1, 2, \ldots, n\}$, relative to the temperatures values y_i measured at n regular time intervals x_i, one can find a representative continuous function f to fit this data, as in Figure 1, by minimizing the sum of the squared differences between y_i and $f(x_i)$.

Figure 1: An example of a fitting function



From the discussions presented in [2, 3], to solve the differential equations that model the temperature inside EAHE, it is very convenient to use sine based functions in the form f(x) =asen(bx+c)+d, where a, b, c and d are real constants, to represent boundary conditions. It is natural to use these functions because they are periodic and their pattern is similar to the analytical solutions for the equations in simplified forms [3].

Therefore, by the least squares method, the constants are chosen minimizing the error function:

$$\text{Error} = \text{Error}\,(a, b, c, d) = \sum_{i=1}^{n} [f(x_i) - y_i]^2 = \sum_{i=1}^{n} [a\,\text{sen}(bx_i + c) + d - y_i]^2. \tag{1}$$

As it is demonstrated in [11], the minimum value of this function occurs in the critical point (a, b, c, d) where all partial derivatives are zero. Computing these derivatives, one finds:

$$\text{Error}_a = 2\sum_{i=1}^{n} [a\,\text{sen}(bx_i + c) + d - y_i]\,\text{sen}(bx_i + c), \tag{2}$$

$$\text{Error}_b = 2a\sum_{i=1}^{n} [a\,\text{sen}(bx_i + c) + d - y_i]\cos(bx_i + c)x_i, \tag{3}$$

$$\text{Error}_c = 2a\sum_{i=1}^{n} [a\,\text{sen}(bx_i + c) + d - y_i]\cos(bx_i + c), \tag{4}$$

$$\text{Error}_d = 2\sum_{i=1}^{n} [a\,\text{sen}(bx_i + c) + d - y_i]. \tag{5}$$

Therefore, assuming a ≠ 0, to discover the minimum value of the error function, it is necessary to find a, b, c, and d satisfying the non-linear system of equations:

$$\sum_{i=1}^{n} [a\,\text{sen}(bx_i + c) + d - y_i]\,\text{sen}(bx_i + c) = 0, \tag{6}$$

$$\sum_{i=1}^{n} [a\,\text{sen}(bx_i + c) + d - y_i]\cos(bx_i + c)x_i = 0, \tag{7}$$

$$\sum_{i=1}^{n} [a\,\text{sen}(bx_i + c) + d - y_i]\cos(bx_i + c) = 0, \tag{8}$$

$$\sum_{i=1}^{n} [a\,\text{sen}(bx_i + c) + d - y_i] = 0. \tag{9}$$

To solve these equations in this work, it was adopted the Newton method, which is reviewed next.

2.2 Newton Method

Given a differentiable vector field

$$\mathbf{F}(a, b, c, d) = \begin{pmatrix} u(a,b,c,d) \\ v(a,b,c,d) \\ w(a,b,c,d) \\ y(a,b,c,d) \end{pmatrix}, \tag{10}$$

it is possible [12, 13] to approximate **F**, on the neighborhood of a point $\xi_0 = (a_0, b_0, c_0, d_0)$, by the linear vector field:

$$\mathbf{L}(a,b,c,d) = \begin{pmatrix} u(\xi_0)+u_a(\xi_0)(a-a_0)+u_b(\xi_0)(b-b_0)+u_c(\xi_0)(c-c_0)+u_d(\xi_0)(d-d_0) \\ v(\xi_0)+v_a(\xi_0)(a-a_0)+v_b(\xi_0)(b-b_0)+v_c(\xi_0)(c-c_0)+v_d(\xi_0)(d-d_0) \\ w(\xi_0)+w_a(\xi_0)(a-a_0)+w_b(\xi_0)(b-b_0)+w_c(\xi_0)(c-c_0)+w_d(\xi_0)(d-d_0) \\ y(\xi_0)+y_a(\xi_0)(a-a_0)+y_b(\xi_0)(b-b_0)+y_c(\xi_0)(c-c_0)+y_d(\xi_0)(d-d_0) \end{pmatrix}. \tag{11}$$

Therefore, one can approximate the solution of **F**(a, b, c, d) = **0** (which composes a non-linear system of equations) by solving alternatively **L**(a,b,c,d) = **0**, which is equivalent to solve the linear system:

$$u(\xi_0) + u_a(\xi_0)(a\text{-}a_0) + u_b(\xi_0)(b\text{-}b_0) + u_c(\xi_0)(c\text{-}c_0) + u_d(\xi_0)(d\text{-}d_0) = 0, \tag{12}$$

$$v(\xi_0) + v_a(\xi_0)(a\text{-}a_0) + v_b(\xi_0)(b\text{-}b_0) + v_c(\xi_0)(c\text{-}c_0) + v_d(\xi_0)(d\text{-}d_0) = 0, \tag{13}$$

$$w(\xi_0) + w_a(\xi_0)(a\text{-}a_0) + w_b(\xi_0)(b\text{-}b_0) + w_c(\xi_0)(c\text{-}c_0) + w_d(\xi_0)(d\text{-}d_0) = 0, \tag{14}$$

$$y(\xi_0) + y_a(\xi_0)(a\text{-}a_0) + y_b(\xi_0)(b\text{-}b_0) + y_c(\xi_0)(c\text{-}c_0) + y_d(\xi_0)(d\text{-}d_0)=0. \tag{15}$$

In matrix form, the equations (12)-(15), can be written as:

$$\begin{pmatrix} u_a(\xi_0) & u_b(\xi_0) & u_c(\xi_0) & u_d(\xi_0) \\ v_a(\xi_0) & v_b(\xi_0) & v_c(\xi_0) & v_d(\xi_0) \\ w_a(\xi_0) & w_b(\xi_0) & w_c(\xi_0) & w_d(\xi_0) \\ y_a(\xi_0) & y_b(\xi_0) & y_c(\xi_0) & y_d(\xi_0) \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} u_a(\xi_0) & u_b(\xi_0) & u_c(\xi_0) & u_d(\xi_0) \\ v_a(\xi_0) & v_b(\xi_0) & v_c(\xi_0) & v_d(\xi_0) \\ w_a(\xi_0) & w_b(\xi_0) & w_c(\xi_0) & w_d(\xi_0) \\ y_a(\xi_0) & y_b(\xi_0) & y_c(\xi_0) & y_d(\xi_0) \end{pmatrix} \begin{pmatrix} a_0 \\ b_0 \\ c_0 \\ d_0 \end{pmatrix} - \begin{pmatrix} u(\xi_0) \\ v(\xi_0) \\ w(\xi_0) \\ y(\xi_0) \end{pmatrix} \tag{16}$$

The matrix of partial derivatives, which is repeated in both sides of equation (16), is known as the Jacobian matrix, J, of the vector field **F**. Thus, the equation (16) can be written simply as:

$$J(\xi_0)\,\xi= J(\xi_0)\,\xi_0 - \mathbf{F}(\xi_0), \tag{17}$$

where ξ = (a, b, c, d), and, as before, ξ_0= (a_0, b_0, c_0, d_0).

Following these formulations, an approximation to a root, x_r, of a vector field **F** can be found iteratively by choosing a suitable initial approximation ξ_0 and computing the successive approximations:

$$J(\xi_k)\,\xi_{k+1}= J(\xi_k)\,\xi_k - \mathbf{F}(\xi_k), \tag{18}$$

for k = 0, 1, 2, …, etc. These resolutions should continue until reaching a maximum number of iterations or a convergence is achieved, that is, the difference in a chosen norm between successive solutions ξ_k and ξ_{k+1} is under a specified tolerance. For this work, it was adopted the infinity norm [6].

This presented technique is called the Newton Method [5, 6]. In fact, to reduce computational costs, the matrix-vector multiplications in equation (18) can be avoided by introducing the variable t = $\xi_{k+1} - \xi_k$. In this way, the algorithm for the Newton method is divided in two parts:

1. Find t by solving equation $J(\xi_k)\,t = -\mathbf{F}(\xi_k)$,
2. Compute $\xi_{k+1} = \xi_k + t$.

Under certain conditions the Newton method can converge "very fast" or not converge at all. Particularly, the convergence is very dependent on the initial approximation, which must be chosen carefully. Following the guidelines proposed in Section 3, for all the cases simulated in this research, the Newton method converged in less than five iterations.

Before proceeding, it is important to present the 4 by 4 Jacobian matrix, J, relative to the set of equations (6)-(9), which were obtained from the least squares methods. The elements, J_{ij}, corresponding to the i-th row and j-th column of J, are given by:

$$J_{11} = \sum_{i=1}^{n} \sin^2(bx_i+c), \tag{19}$$

$$J_{12}= \sum_{i=1}^{n} x_i \cos(bx_i+c)\,[2a \sin(bx_i+c)+d-y_i], \tag{20}$$

$$J_{13}= \sum_{i=1}^{n} \cos(bx_i+c)\,[2a \sin(bx_i+c)+d-y_i], \tag{21}$$

$$J_{14}= \sum_{i=1}^{n} \sin(bx_i+c), \tag{22}$$

$$J_{21}=\sum_{i=1}^{n} x_i \sin(bx_i+c)\cos(bx_i+c), \quad (23)$$

$$J_{22}=\sum_{i=1}^{n} x_i^2 \{a\cos^2(bx_i+c)-[a\,\mathrm{sen}(bx_i+c)+d-y_i]\,\mathrm{sen}(bx_i+c)\}, \quad (24)$$

$$J_{23}=\sum_{i=1}^{n} x_i \{a\cos^2(bx_i+c)-[a\,\mathrm{sen}(bx_i+c)+d-y_i]\,\mathrm{sen}(bx_i+c)\}, \quad (25)$$

$$J_{24}=\sum_{i=1}^{n} x_i \cos(bx_i+c), \quad (26)$$

$$J_{31}=\sum_{i=1}^{n} \sin(bx_i+c)\cos(bx_i+c), \quad (27)$$

$$J_{32}=J_{23}, \quad (28)$$

$$J_{33}=\sum_{i=1}^{n} \{a\cos^2(bx_i+c)-[a\,\mathrm{sen}(bx_i+c)+d-y_i]\,\mathrm{sen}(bx_i+c)\}, \quad (29)$$

$$J_{34}=\sum_{i=1}^{n} \cos(bx_i+c), \quad (30)$$

$$J_{41}=J_{14}, \quad (31)$$

$$J_{42}=\sum_{i=1}^{n} a\,x_i \cos(bx_i+c), \quad (32)$$

$$J_{43}=\sum_{i=1}^{n} a\cos(bx_i+c), \quad (33)$$

$$J_{44}=n. \quad (34)$$

2.3 Commented Matlab Code

To show how this methodology was implemented computationally, this subsection presents a corresponding code written in the Matlab (R2008a) programming language [14]. For those not used to Matlab, it is worth to read the following points.

- Anything written to the right of the symbol `%` is taken as a commentary.
- A very long command statement can be continued to the next lines using ellipsis (...).
- The function **`sum()`**, usually written with the syntax **`sum(x)`**, returns the sum of the elements of an array **`x`**. It is useful to compute summations without the need of a loop.
- The function **`max()`**, written with the syntax **`max(x)`**, returns the largest element of an array **`x`**.
- It is possible to make operations for all the elements of an array at once. For instance, if **`x`** and **`y`** are two n×1 arrays, and **`k`** is a constant, the following statements: `abs(x)`, **`k*x`**, `x.^2`, `x.*y`, return n×1 arrays with, respectively, the absolute value of all elements of **`x`**, all the elements of **`x`** multiplied by **`k`**, all the squares of **`x`**, an element by element product of **`x`** and **`y`**.
- A linear system `Ax=b`, where `A` and `b` are, respectively, n×n and n×1, arrays, can be solved simply by the command statement `x=A\b`;
- As usual, Matlab counts with a conditional **`if-else`** construction and also a loop **`while`**. Even though the later has a syntax **`while`**`(condition) do commands` **`end`**, the loop terminates not only when the specified condition holds false, but also when a **`break`** statement is executed.

After this preamble, the commented code is given below.

```
% .................INPUT DATA........................................
%
% Initial approximation for the sine coefficients
a=7.5; b=0.0185; c=0.8316; d=20.5;
% Tolerance and maximum number of iterations
tol = 1e-5; Nmax = 10;
% Time intervals (x) and Temperature values (y)
      % Number of time intervals
n=1000;
      % Command to specify the path to the data files folder
addpath('\files_folder');
      % Reading n lines of data from a file
```

```
[x,y]=textread('file_with_time_and_temperature_data.txt','%f %f',n);
%
%............COMPUTING THE SINE COEFFICIENTS (a, b, c, d)...........
%
cont=1; % Counter for the number of iterations
%
while(cont < Nmax)
       % Computing the Jacobian Matrix
       jac=[sum(sin(b*x+c).^2),sum(x.*cos(b*x+c).*(2*a*sin(b*x+c)+d-y)),...
       sum(cos(b*x+c).*(2*a*sin(b*x+c)+d-y)),sum(sin(b*x+c));...
       % end of the matrix first line
       sum(x.*sin(b*x+c).*cos(b*x+c)),...
       sum(x.^2.*(a*(cos(b*x+c)).^2-(a*sin(b*x+c)+d-y).*sin(b*x+c))),...
       sum(x.*(a*(cos(b*x+c)).^2-(a*sin(b*x+c)+d-y).*sin(b*x+c))),...
       sum(x.*cos(b*x+c));...
       % end of the matrix second line
       sum(sin(b*x+c).*cos(b*x+c)),...
       sum(x.*(a*(cos(b*x+c)).^2-(a*sin(b*x+c)+d-y).*sin(b*x+c))),...
       sum((a*(cos(b*x+c)).^2-(a*sin(b*x+c)+d-y).*sin(b*x+c))),...
       sum(cos(b*x+c));...
       % end of the matrix third line
       sum(sin(b*x+c)),sum(a*cos(b*x+c).*x),sum(a*cos(b*x+c)),n];
       % end of the matrix fourth line

       % Computing the Vector Field F
       F=[sum((a*sin(b*x+c)+d-y).*sin(b*x+c));...
       sum((a*sin(b*x+c)+d-y).*cos(b*x+c).*x);...
       sum((a*sin(b*x+c)+d-y).*cos(b*x+c));sum((a*sin(b*x+c)+d-y))];

       % Solving the Linear System jac t =-F
       t =jac\(-F);

       % Check for the convergence in the infinity norm
       if (max( abs(t) )< tol )
              break; % terminates the while loop
       %
       else
              cont=cont+1;% advance the counter
              %
              a=a+t(1); b=b+t(2); c=c+t(3); d=d+t(4); %updates the solution
       end
  end
```

3 Results and Discussion

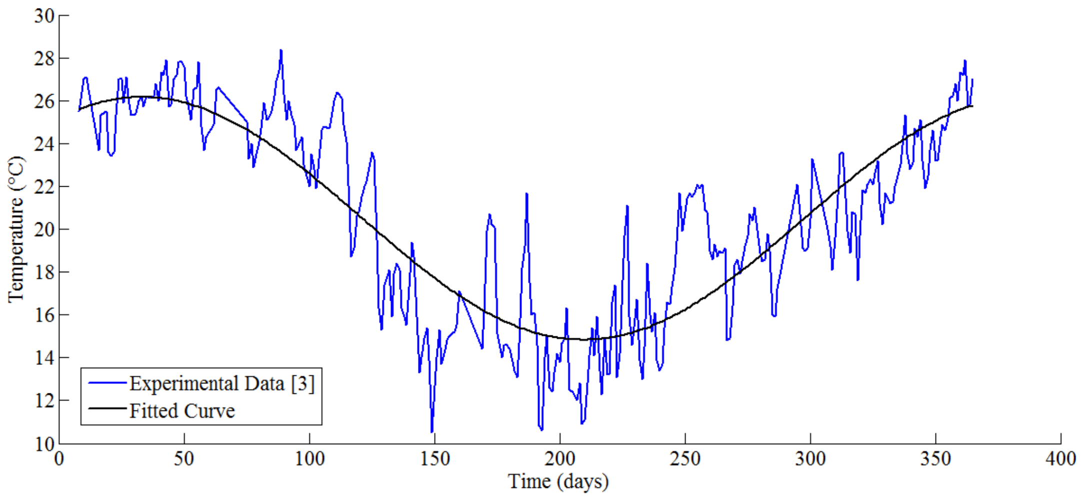
This work used empirical data from the research done in [2], which provided the temperatures measured along the year on the soil surface and at the air inlet of an EAHE duct build in the experimental house, called Casa Ventura, located in the city of Viamão (RS)

Even though the temperatures were measured hour by hour, this paper considered daily averages of those data to find a sine based function f(x) = asen(bx+c)+d corresponding to the 365 days of a year, using the methodology outlined in the previous section.

The Figure 2 shows a graph displaying the experimental data and the fitted curve representing the air temperature at the inlet of a duct along the year. It can be noticed that the sine function is well suited, and even natural, to describe the temperature evolution, reaching its maximum value in the summer and minimum in the winter. It also follows the gradual drop and rise of the temperatures, respectively, in the fall and in the spring.

Besides, it is worth to say that there is a high correlation coefficient, of 0.8693, between the fitted curve and the experimental data.

Figure 2: Air temperature at the inlet of an EAHE duct



As it was mentioned in the methodology, the Newton method converged fast, that is, in less than five iterations for all cases tested. However, it should be pointed that this depends on the quality of the initial approximation (a_0, b_0, c_0, d_0) which can be fairly estimated from a previous analysis of the experimental data.

In this regard, computing an average of the temperatures along the summer, A_s, and along the winter, A_w, it is possible to estimate the amplitude term by $a_0 = (A_s-A_w)/2$. Certainly, there are other valid ways to estimate the same term. For instance, taking a look at the Figure 2, one can note the peaks of temperature during summer, in average, are around 28°C. In the same way, the average minimum temperatures during winter are close to 13°C. Therefore, it is reasonable to assume an amplitude $a_0 \approx (28-13)/2 = 7.5$.

As for the term d_0, it represents the annual average temperatures. Thus, it can be calculated taking the mean value of the experimental data. A simple way to do it in Matlab is using the function `mean()`, whose usual syntax is `mean(A)`, which computes the average or mean value of a given array `A`. It is worth to mention that one estimate can also be deduced from Figure 2. The value of d_0 is approximately an average between the maximum summer temperatures and the minimum winter ones, in other words, $d_0 \approx (28+13)/2 = 20.5$.

Finally, to estimate the terms b_0 and c_0, it was adopted the following procedure. For the external air, it is known empirically that the peak of summer temperatures occurs, usually, at the beginning of February (about forty days after the year start). On the other hand, the minimum winter temperatures occur in July (about two hundred and ten days after the year start). Since the sine function achieves its maximum value at the point $\pi/2$ and its minimum at $3\pi/2$, it is sufficient to make a correspondence among, respectively, the days x = 40 and x = 210, and the points $\pi/2$ and $3\pi/2$. Thus, b_0 and c_0 are found solving the linear system:

$$40\, b_0 + c_0 = \pi/2, \tag{35}$$

$$210\, b_0 + c_0 = 3\pi/2, \tag{36}$$

whose solution is $b_0 \approx 0.0185$ and $c_0 \approx 0.8316$.

All these values are presented in the Matlab code above. In fact, running the program, it was obtained the following function to represent the temperature at the air inlet of the EAHE duct:

$$T(x) = 20.4967 + 5.6634 \sin(0.0178x + 0.9787). \quad (37)$$

Therefore, except for the amplitude term, all the initial approximations were fairly precise.

On the other hand, one should highlight that the least squares and Newton methods are robust enough to compute the coefficients even when the initial approximations are rather rough. For instance, using the same previous estimates, it was obtained the following function to represent the temperature at the air inlet of other EAHE duct:

$$T(x) = 21.7895 - 5.9614 \sin(-0.0184x + 5.3991), \quad (38)$$

which is very different than the one in (37).

The Figure 3 displays a graph comparing the fitting curve with the experimental data for the temperature at this second inlet. As before, it is possible to see that the sine curve also matches the behavior of the temperature along the year. Besides, for this case, the correlation between the fitted curve and the experimental data is 0.8788, which is higher than the previous result.

This methodology was also employed to obtain the temperatures in the soil for different depths. These results were obtained from numerical data available using the EAHE model validated in [15]. In the Figure 4, there is a graph comparing the temperature curves, at the depth of 4 m, for the numerical model and the fitted one. In this case, the correlation between them is 0.9976. In fact, the curves fitted from the least squares method were used by the authors to construct new EAHE models presented in the articles [3], [4] and [15].

Figure 3: Air temperature at the inlet of a second EAHE duct

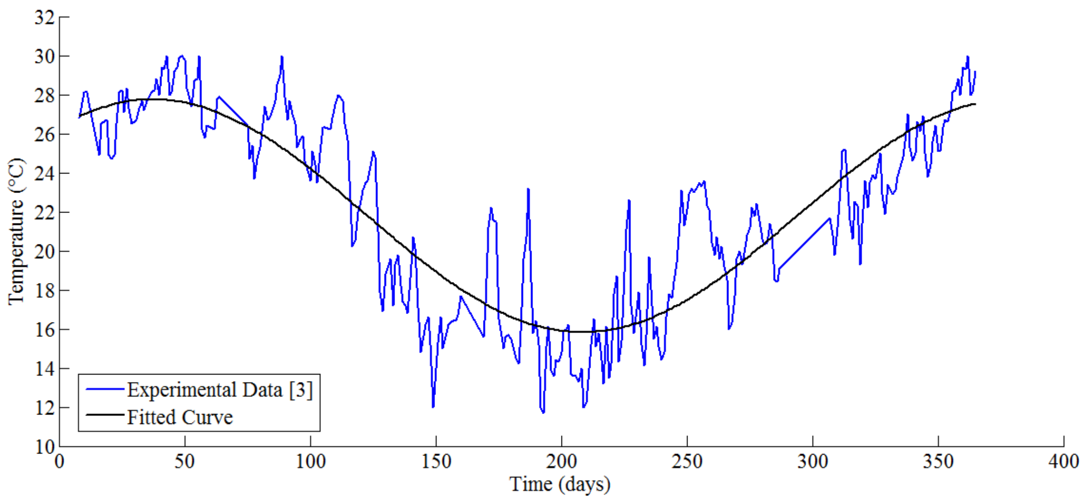
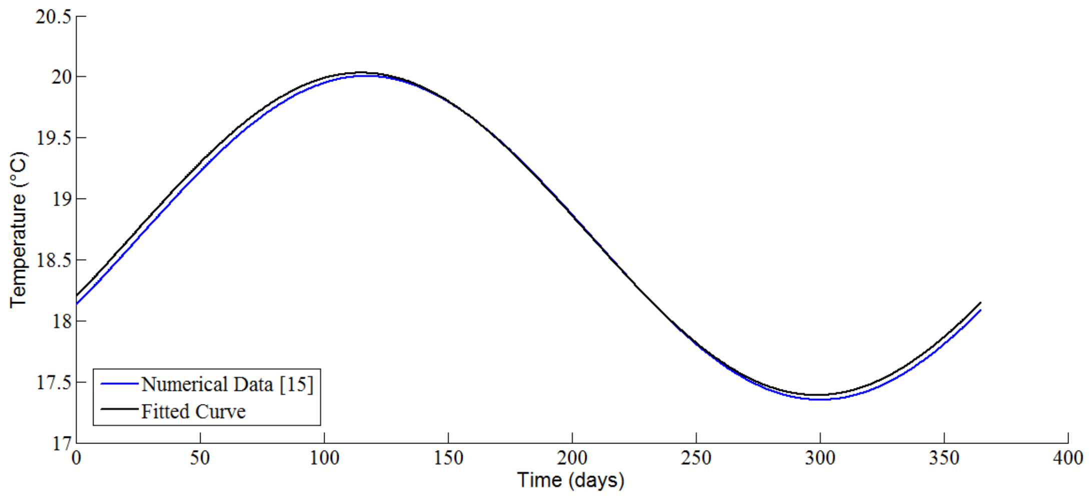


Figure 4: Soil temperature at the depth of 4m



Before concluding this section, it may be worth to note that all simulations were run in a computer with the following specifications: 64-bit processor, Intel Core i5, 2.53 GHz, and 3GB of RAM.

4 Conclusions

This paper aimed to present a computer methodology to find fitting curves which can help in the development of new EAHE models. For this purpose, it was developed a computer code following a methodology based on the least squares and Newton methods. Such code was written in the Matlab programming language.

For this research, it was used several empirical data relative to the temperature of air and soil measured in the facilities of an experimental house in the Brazilian state of Rio Grande do Sul. As it was verified, the method allowed finding a sine-based function which naturally modeled the phenomena.

Even though the proposed methodology is relatively more complex than using polynomial fitting, due to the need of solving non-linear systems of equations, it was observed a fast convergence in the solutions. This was the result of a previous analysis to find suitable initial approximations. In fact, even facing poor initial approximations, the method was robust enough to converge for highly correlated results.

Finally, this work also helps to fill a void in the literature about the use of the least squares methods with trigonometric functions.

Acknowledgments

Ruth da Silva Brum thanks the CNPq for her doctoral scholarship. The authors Luiz Alberto Oliveira Rocha, Liércio André Isoldi and Elizaldo Domingues dos Santos thank CNPq for their research grants.

References

[1] VETORAZZO, L. *Alta dos serviços e calor mudam o pico de energia para meio da tarde*. 2014. Disponível em: <http://www1.folha.uol.com.br/mercado/2014/01/1403798-alta-dos-servicos-e-calor-mudam-o-pico-de-energia-para-meio-da-tarde.shtml>. Access in: 19 dec. 2014.

[2] VAZ, J. *Estudo Experimental e Numérico sobre o Uso do Solo como Reservatório de Energia para o Aquecimento e Resfriamento de Ambientes Edificado*. Tese (Doutorado em Engenharia Civil) – Programa de Pós-Graduação em Engenharia Civil da Universidade Federal do Rio Grande do Sul, 2011.

[3] BRUM, R. S.; *et al*. Development of simplified numerical model for evaluation of the influence of soil-air heat exchanger installation depth over its thermal potential. *International Journal of Advanced Renewable*

Energy Research, v. 1, p. 505–514, 2012. ISSN 2251-9408. Available in: <http://www.grsci-techpress.org/index.php/IJARER/article/view/62>.

[4] BRUM, R. S.; *et al*. A new computational modeling to predict the behavior of earth-air heat exchangers. *Energy and Buildings*, Elsevier, v. 64, p. 395–402, 2013. ISSN 0378-7788. Available in: < http://www.sciencedirect.com/science/article/pii/S0378778813003186>.

[5] GERALD, C. F. *Applied Numerical Analysis*. Menlo Park, CA, USA: Addison-Wesley Publishing Company, 1984.

[6] BURDEN, R. L.; FAIRES, J. D. *Análise numérica*. São Paulo, SP, Brasil: Pioneira Thomsom Learning, 2003.

[7] FRANCO, N. B. *Cálculo Numérico*. São Paulo, SP, Brasil: Pearson Prentice Hall, 2006.

[8] ARENALES, S; DAREZZO, A. *Cálculo Numérico, Aprendizagem com Apoio de Software*. São Paulo, SP, Brasil: Pioneira Thomsom Learning, 2008.

[9] MATHEWS, J. H. *Numerical Methods for Mathematics, Science and Engineering.* Upper Saddle River, NJ, USA: Prentice Hall, Inc., 1992.

[10] RUGIERO, M. A. G.; LOPES, V. L. R. *Cálculo numérico, aspectos teóricos e computacionais.* São Paulo, SP, Brasil: Makron Books, 1996.

[11] BRUM, R. S. *Modelagem Computacional de Trocadores de Calor Solo-Ar*. Dissertação (Mestrado em Modelagem Computacional) – Programa de Pós-Graduação em Modelagem Computacional da Universidade Federal do Rio Grande, 2013.

[12] STEWART, J. *Cálculo Volume II.* São Paulo, SP, Brasil: Pioneira Thomsom Learning, 2006.

[13] BARTLE, R. G. *The elements of real analysis.* New York, USA: John Wiley & Sons, 1976.

[14] CHAPMAN, S. J. *Programação em Matlab para engenheiros.* São Paulo, SP, Brasil: Cengage Learning, 2011.

[15] BRUM, R. S.; *et al*. Two-Dimensional Computational Modeling Of The Soil Thermal Behavior Due To The Incidence Of Solar Radiation. *EngenhariaTérmica (Thermal Engineering)*, DEMEC – UFPR, v. 12, n. 2, p. 63–68, 2013. ISSN 1676-1790. Available in: < http://demec.ufpr.br/reterm/ed_ant/21/>.