DEFIN - COMPUTATIONAL APPLICATION OF FINITE DIFFERENCE METHOD IN HYDRAULICS ENGINEERING

Marcos Rogério Szeliga
Civil Engineering Department – State University of Ponta Grossa
Av. Carlos Cavalcanti, 4748.
84030-900 – Ponta Grossa – PR – Brazil.
marcosrs@uepg.br

Abstract: The extraction of water from a groundwater aquifer through a well causes the lowering of hydraulic head around the point of extraction. One of the tasks in the project of use of this water resource is to estimate the lowering the water table and the extent of influence on the aquifer caused by extraction. Based on this activity a computer program for applying the Finite Difference Method (FDM) was formulated. With this program results using different algorithms for comparison and verification of the efficiency of each algorithm were obtained. The computer program DIFIN in VBA language - Visual Basic for Applications - is a tool for visualization of results and comparison of methods for solving differential equations applied. In this program is used the MDF with numerical solution by the iterative algorithms Jacobi, Gauss and SOR. Through graphical results is possible to compare the methods, verify the numerical solutions, analytical solutions, the numerical convergence and conclude about the feasibility of the MDF and algorithms in solving differential equations using numerical methods. Using the graphical interface is provided an analysis of these results and is possible to conclude on the efficiency of the methods, as well as on the ability to aid in the understanding the application providing a didactic tool for learning in numerical methods applied in engineering.

Keywords: Finite Difference Method, Iterative Methods, Hydraulic Engineering.

1. INTRODUCTION

The Finite Difference Method consists on obtaining approximate solution of a partial differential equation in discrete points in the domain. The technique uses the discretization of the domain and the replacement of the derivatives present in the differential equation by approaches involving finite increments. In practice the derivatives are replaced by the incremental ratio that the problem was discretized. When the domain has more than one variable, the technique is applied to each variable separately (CUMINATO *et al*., 1999).

The DIFIN computational program is a visualization tool of results and comparison of methods for the solution of partial differential equations applied to extraction of water from an aquifer through a water-table well.

This simulation program is the resolution of a hydrogeological model using 3 different numerical methods by introducing physical parameters that characterize the aquifer, such as conductivity (K) of the medium, the thickness of the aquifer (e), its

transmissivity (T) and other parameters used in definition of numerical methods. The methods used are Jacobi, Gauss-Seidel and SOR - Successive Over Relaxation.

The methods are recursive and differ by data source and by parameterization, achieving solutions with a different number of iterations, however, tend to provide identical results.

Jacobi method uses data from the previous iteration to update the calculations. Gauss-Seidel uses data from the previous and from the current iteration for to recalculate the values. SOR method - based on the Gauss-Seidel method introduces a relaxation parameter (α) to accelerate the iterative process.

The problem refers to obtaining a two dimensional contour graph of pressure in the aquifer in the region affected by the extraction of water flow-rate Q at a given point. With this result it can be noticed the effects of water extraction in lowering level of groundwater and its extension.

The language VBA - Visual Basic Aplicattions - using a GUI - Graphical User Interface - provided the right conditions for a friendly and intuitive interface to offer an overview of the application. (JELEN *et al*., 2008)

The program also allows viewing the solving of the problem in a mesh discretization in top view and in profile, beyond the possibilities of comparison between the three methods.

The numerical results can be compared with each other, and also between analytical results through the graphical interface available to the user.

2. FINITE DIFFERENCE METHOD - FDM

The numerical finite difference approximations are based on the Taylor series expansion of a function h. The expansion allows the estimation of the function h in h(x), knowing the value of h for h_0. Assuming that h is continuous on the interval [a, b] of interest and that it has continuous derivatives up to order n in this interval, Taylor's Theorem allows to write, for every point $x \in [a, b]$:

$$h(x)=h(x_0)+\Delta x \left.\frac{\partial h}{\partial x}\right|_{x_0} + \frac{(\Delta x)^2}{2!}\left.\frac{\partial^2 h}{\partial x^2}\right|_{x_0} + \frac{(\Delta x)^3}{3!}\left.\frac{\partial^3 h}{\partial x^3}\right|_{x_0} + \ldots + R_n \qquad (01)$$

where $\Delta x = x - xo$ and R_n is the rest (FORTUNA, 2000).

Whereas Figure 1, which shows a few points of an one-dimensional mesh. The points are evenly spaced of $\Delta x = x_i - x_{i-1}$.

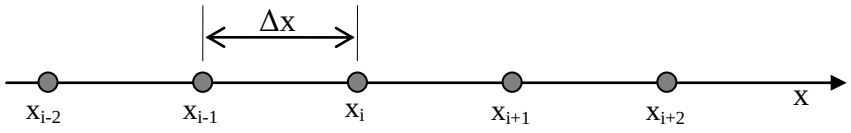

Figure 1 - One-dimensional mesh of finite differences.

Expanding $h(x_i + \Delta x)$ in Taylor series around the point x_i, we have:

$$h(x_i + \Delta x) = h(x_i) + \Delta x \frac{\partial h}{\partial x}\Big|_i + \frac{(\Delta x)^2}{2!}\frac{\partial^2 h}{\partial x^2}\Big|_i + \frac{(\Delta x)^3}{3!}\frac{\partial^3 h}{\partial x^3}\Big|_i + \ldots + R_n \qquad (02)$$

Isolating the first derivative:

$$\frac{\partial h}{\partial x}\Big|_i = \frac{h(x_i + \Delta x) - h(x_i)}{\Delta x} - \left[\frac{(\Delta x)^2}{2!}\frac{\partial^2 h}{\partial x^2}\Big|_i + \frac{(\Delta x)^3}{3!}\frac{\partial^3 h}{\partial x^3}\Big|_i + \ldots + R_n\right] \qquad (03)$$

or

$$\frac{\partial h}{\partial x}\Big|_i = \frac{h(x_i + \Delta x) - h(x_i)}{\Delta x} - \text{LTE} \qquad (05)$$

where LTE is the Local Truncation Error. Simplifying the notation, writing $h_{i\pm k}$ for $h(x_i \pm x\Delta x)$, the equation 05, suppressing the LTE, becomes:

$$\frac{\partial h}{\partial x}\Big|_i = \frac{h_{i+1} - h_i}{\Delta x} \qquad (06)$$

Expression 06 is a finite difference equation that represents a first-order approximation for the first derivative of h, using forward differences (FORTUNA, 2000).

2.1. FDM applied to a field of groundwater extraction

Equation 07 - Darcy's Law - describes the phenomenon related to fluid flow in a porous medium:

$$Q = -KA\frac{\partial h}{\partial x} \qquad (07)$$

where K is the hydraulic conductivity, A is the area of the transverse section, h is the hydraulic head and x is the flow direction.

$$\frac{\partial h}{\partial x} = -\frac{KA}{Q} = \text{constant} \rightarrow \frac{\partial^2 h}{\partial x^2} = 0 \qquad (08)$$

which is the one-dimensional Laplace equation for a homogeneous medium and steady-state flow. (MARGOLIN *et al*., 1998).

Discretizing the problem domain under study and considering equation 08 is obtained, by finite differences, the expression:

$$\frac{\partial^2 h}{\partial x^2} = \frac{h_{i+1} - 2h_i + h_{i-1}}{\Delta x^2} = 0$$

$$h_i = \frac{h_{i+1} + h_{i-1}}{2} \qquad (09)$$

For a two-dimensional problem, Laplace's equation assumes the following:

$$\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2} = 0 \qquad (10)$$

Discretizing:

$$\frac{h_{i+1,j}-2h_{i,j}+h_{i-1,j}}{\Delta x^2}+\frac{h_{i,j+1}-2h_{i,j}+h_{i,j-1}}{\Delta y^2}=0$$

if $\Delta x = \Delta y$:

$$h_{i,j}=\frac{h_{i+1,j}+h_{i-1,j}+h_{i,j+1}+h_{i,j-1}}{4} \qquad (11)$$

The expression 11 is known as Equation five-point-star. Its geometric configuration related to neighboring nodes in the finite difference mesh is shown in Figure 2.

For the specific problem of extraction of water through a water well, Laplace's equation is parameterized with the values of flow-rate extraction (Q - m^3/s) and hydraulic transmissivity (T- m^2/s). The value of the aquifer thickness (b) multiplied by the hydraulic conductivity (K) provides the value of the hydraulic transmissivity.

$$\frac{\partial^2 h}{\partial x^2}+\frac{\partial^2 h}{\partial y^2}-\frac{Q}{T}=0 \qquad (12)$$

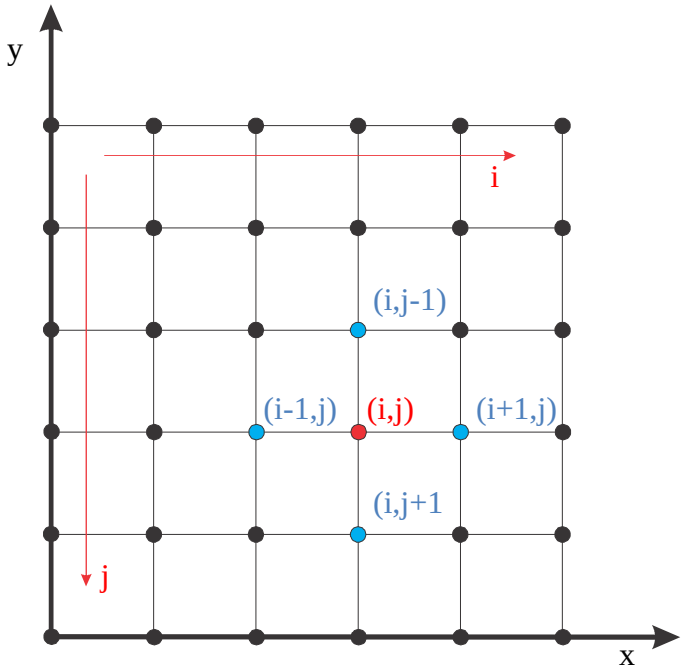

Figure 2 - Two-dimensional mesh of finite differences.

2.2. Iterative numerical methods

Jacobi method uses data from the previous iteration to update the calculations. Equation 13 is adapted for use in the Jacobi iterative method where the index v is the number of the iteration.

$$h_{i,j}^{v}=\frac{h_{i+1,j}^{v-1}+h_{i-1,j}^{v-1}+h_{i,j+1}^{v-1}+h_{i,j-1}^{v-1}}{4} \qquad (13)$$

Gauss-Seidel through the equation 14 uses data from the previous and from the current iteration for to recalculate the values in each iteration.

$$h_{i,j}^{v}=\frac{h_{i+1,j}^{v-1}+h_{i-1,j}^{v}+h_{i,j+1}^{v-1}+h_{i,j-1}^{v}}{4} \qquad (14)$$

SOR method - based on the Gauss-Seidel method introduces a relaxation parameter (α) to accelerate the iterative process.

$$h_{i,j}^{v}=(1-\alpha)h_{i,j}^{v-1}+\alpha\frac{h_{i+1,j}^{v-1}+h_{i-1,j}^{v}+h_{i,j+1}^{v-1}+h_{i,j-1}^{v}}{4} \qquad (15)$$

These iterative methods use as stopping criterion a numerical precision user-defined or a limit number of iterations.

An analytical solution to this problem can be obtained by the expression 16:

$$h_R=\frac{Q\log\frac{R}{R_\infty}}{2\pi T}+R_\infty \qquad (16)$$

where R is the radius of the point of calculation with respect to the well and R_∞ is the radius of the border not affected by the action of the extraction flow. (ICHIRO *et al.*, 2003)

3. DIFIN PROGRAM

The computer program DIFIN in VBA language, developed for application of numerical methods shows the main screen reproduced in Figure 3 where the user must enter the values of physical and numerical parameters like cell spaces of the mesh, hydraulic head, flow-rate of extraction, conductivity, thickness of the aquifer, values of equipotential lines and numerical precision as criteria of stop of iterations.

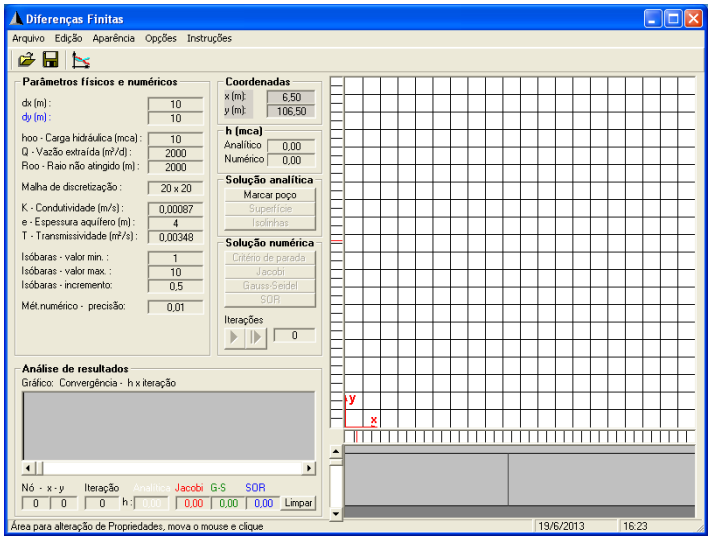

Figure 3 - Main screen of the DIFIN program.

To change the values the user must click on the fields, so a box will appear for entering values as shown in Figure 4.

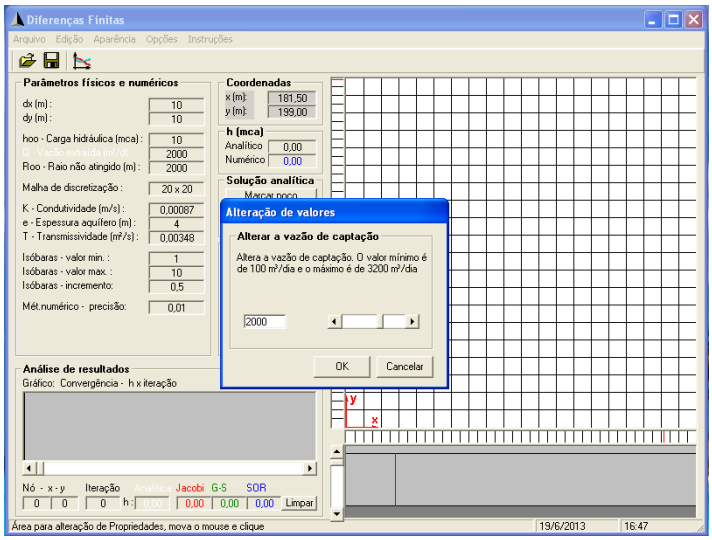

Figure 4 - Box for change values.

The next step is to locate the well catchment in the coordinate system of the finite difference mesh pressing the button "Marcar poço" and clicking on a node in the mesh as shown by Figure 5.

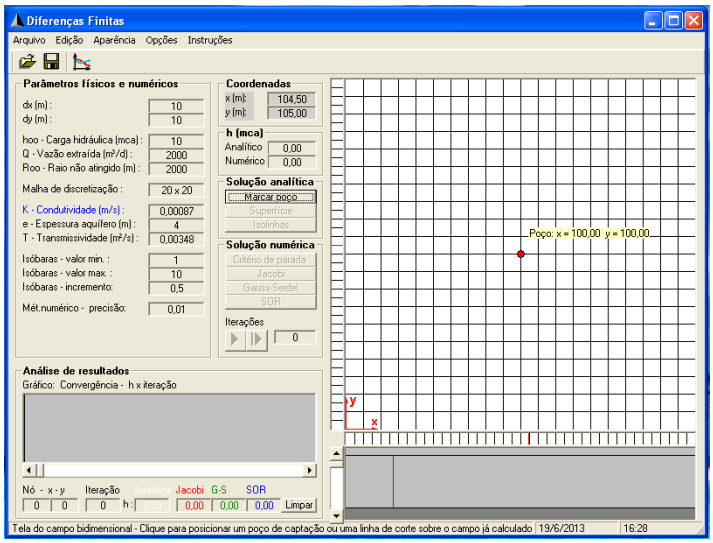

Figure 5 - Positioning the well on the coordinate system.

Pressing the "Superfície" button we obtain the analytical solution for the lowering the water table. In the window a map of hydraulic head values will be displayed in grayscale as well as in the numerical displays "Coordenadas" and "h (mca)" values according to the cursor position, as can be noticed in the reproduction of Figure 6

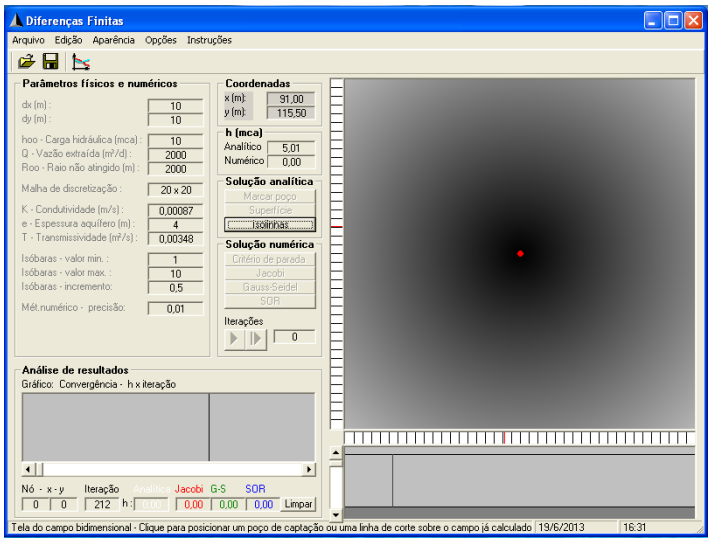

Figure 6 - Colormap grayscale - Hydraulic head.

The solution in colormap grayscale can be changed to map of equipotential lines through the "Isolinhas" button showing a result as shown in Figure 7.

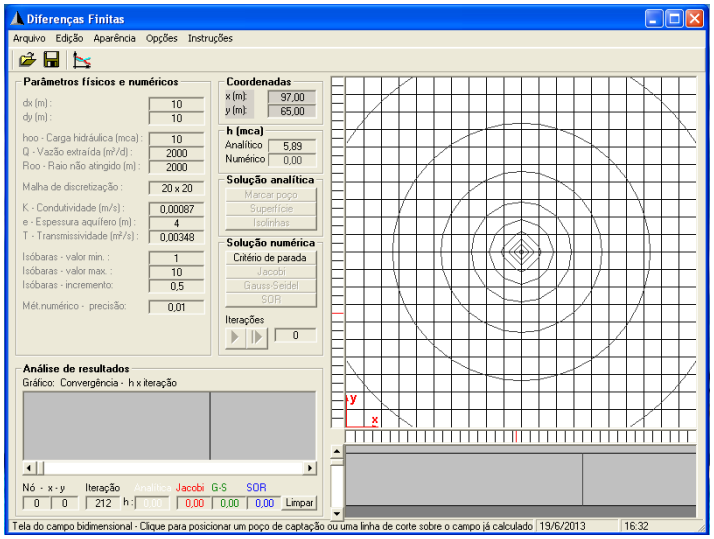

Figure 7 - Equipotential lines - Hydraulic head.

To start the numerical solutions the user must define a stopping criterion for the iterative processes. Clicking on the "Critério de parada" button a window of options will appear. In this window the user sets the operating parameters of the iterative methods according to Figure 8.

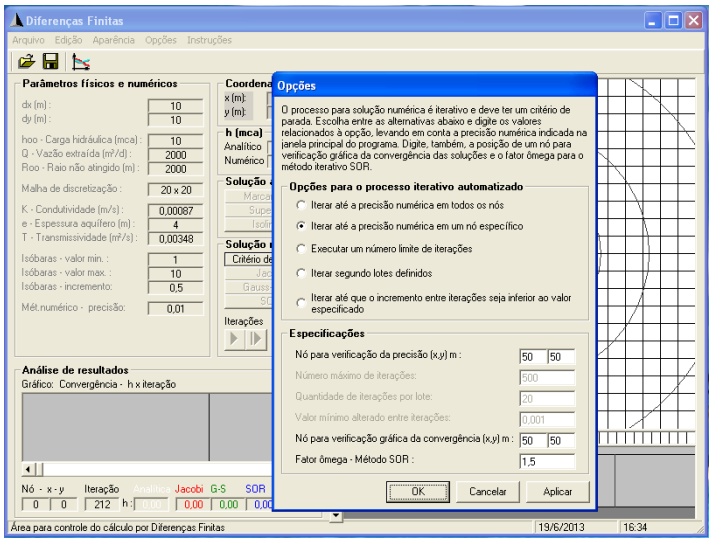

Figure 8 - Operating parameters of the iterative methods.

Clicking on "Jacobi" button will start the numerical procedure and the equipotential lines are drawn on the results window and hydraulic profile in the lower window and the convergence graph in the left display. Numerical values can be seen in displays "Coordenadas" and "h (mca)" by positioning the cursor over the windows. The result can be observed through the reproduced screen in Figure 9.

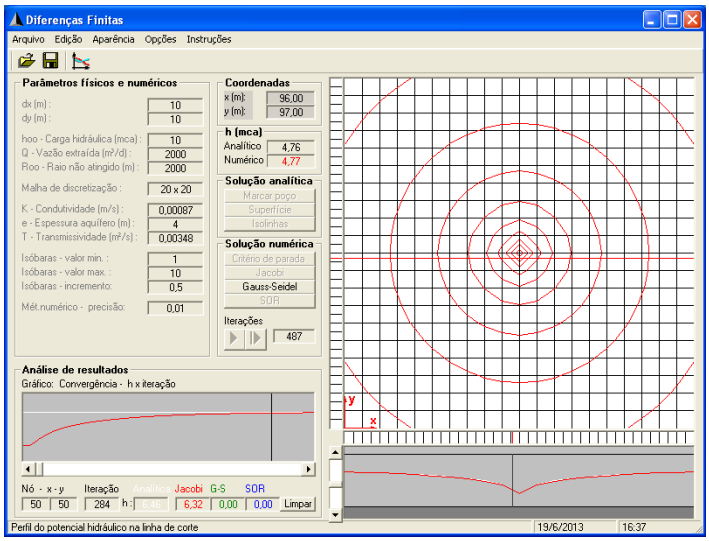

Figure 9 - Result for the Jacobi method.

After the execution of the Jacobi procedure, the button "Gauss-Seidel" will be available to actuation and will provide the execution of the subroutine Gauss. The source code of the subroutine Gauss is reproduced in Table 1. This code in VBA language presents the variables listed in Table 2.

```
Sub Gauss()
If px = 0 And py = 0 Then
   For Y = 0 To grid - 1
   For X = 0 To grid - 1
      If X = 0 And Y = 0 Then
         potnum(X, Y) = (2 * potnum(X + 1, Y) + 2 * potnum(X, Y + 1) - CDbl(Label3(3).Caption) / (CDbl(Label3(8).Caption) *
86400)) / 4
      ElseIf X = 0 And Y = grid Then
         potnum(X, Y) = (2 * potnum(X + 1, Y) + 2 * potnum(X, Y - 1)) / 4
      ElseIf X = 0 And Y > 0 And Y < grid Then
         potnum(X, Y) = (2 * potnum(X + 1, Y) + potnum(X, Y - 1) + potnum(X, Y + 1)) / 4
      ElseIf X = grid And Y = 0 Then
         potnum(X, Y) = (2 * potnum(X - 1, Y) + 2 * potnum(X, Y + 1)) / 4
      ElseIf X = grid And Y = grid Then
         potnum(X, Y) = (2 * potnum(X - 1, Y) + 2 * potnum(X, Y - 1)) / 4
      ElseIf X = grid And Y > 0 And Y < grid Then
         potnum(X, Y) = (2 * potnum(X - 1, Y) + potnum(X, Y - 1) + potnum(X, Y + 1)) / 4
      ElseIf X > 0 And X < grid And Y = 0 Then
         potnum(X, Y) = (potnum(X + 1, Y) + potnum(X - 1, Y) + 2 * potnum(X, Y + 1)) / 4
      ElseIf X > 0 And X < grid And Y = grid Then
         potnum(X, Y) = (potnum(X + 1, Y) + potnum(X - 1, Y) + 2 * potnum(X, Y - 1)) / 4
      Else
         potnum(X, Y) = (potnum(X + 1, Y) + potnum(X - 1, Y) + potnum(X, Y + 1) + potnum(X, Y - 1)) / 4
      End If
   Next X
   Next Y
Else
   For Y = 1 To grid - 1
   For X = 1 To grid - 1
      If X = pxx And Y = pyy Then
         potnum(X, Y) = (potnum(X + 1, Y) + potnum(X - 1, Y) + potnum(X, Y + 1) + potnum(X, Y - 1) - CDbl(Label3(3).Caption)
/ (CDbl(Label3(8).Caption) * 86400)) / 4
      Else
         potnum(X, Y) = (potnum(X + 1, Y) + potnum(X - 1, Y) + potnum(X, Y + 1) + potnum(X, Y - 1)) / 4
      End If
   Next X
   Next Y
End If
alteração1 = potnum(xnó / CDbl(Label3(0).Caption), ynó / CDbl(Label3(1).Caption)) - anterior1
anterior1 = potnum(xnó / CDbl(Label3(0).Caption), ynó / CDbl(Label3(1).Caption))
alteração2 = potnum(xcnó / CDbl(Label3(0).Caption), ycnó / CDbl(Label3(1).Caption)) - anterior2
anterior2 = potnum(xcnó / CDbl(Label3(0).Caption), ycnó / CDbl(Label3(1).Caption))
isolinhas
nint = nint + 1
Picture6.Line -(nint, 10 / CDbl(Label3(2).Caption) * 77 / 13 * potnum(xcnó / CDbl(Label3(0).Caption), ycnó /
CDbl(Label3(1).Caption)) + 231 / 13), RGB(CR, CG, CB)
If nint > 332 Then
   Picture6.Left = 333 - nint
   HScroll1.Value = -(Picture6.Left)
End If
Picture6.Refresh
Label7.Caption = nint
Label7.Refresh
End Sub
```

Table 1 - Source code - Subroutine for the Gauss method.

Variable	Content
px – py	Coordinates of well of groundwater
Potnum	Hydraulic head - numeric solution
Grid	Maximum coordinate in the mesh
Nint	Number of iterations

Table 2 - Variables in the Gauss method.

Figures 10, 11, 12 and 13 show the evolution of the application of the Gauss method in the problem. In the lower graphs, the white lines correspond to the analytical solution. It may be noted the successive approximation of the solution with the development of the iterative process. There is also a faster result with the Gauss algorithm (green lines) with relation to Jacobi algorithm (red lines).

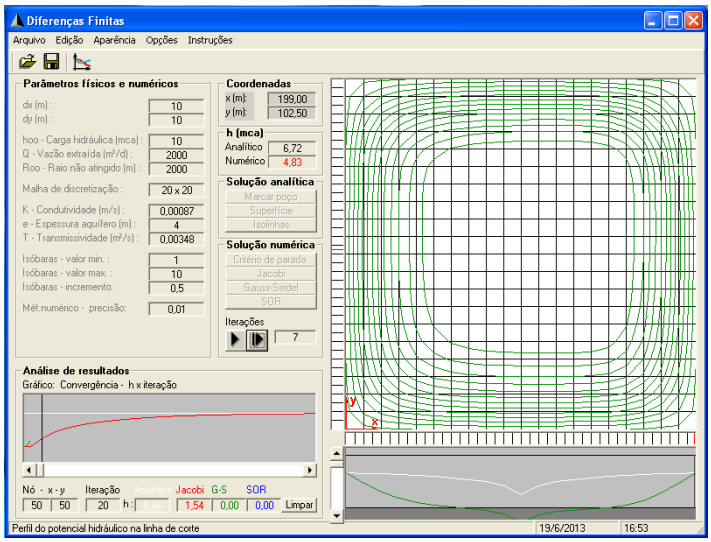

Figure 10 - Gauss method - Iteration 4.

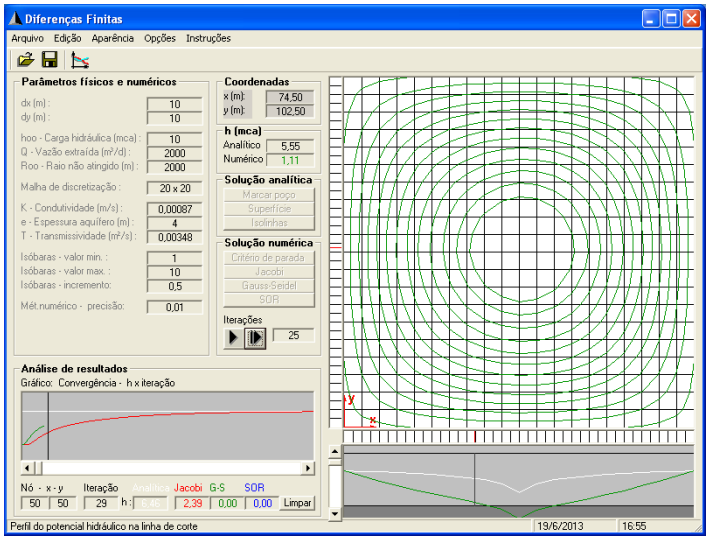

Figure 11 - Gauss method - Iteration 20.

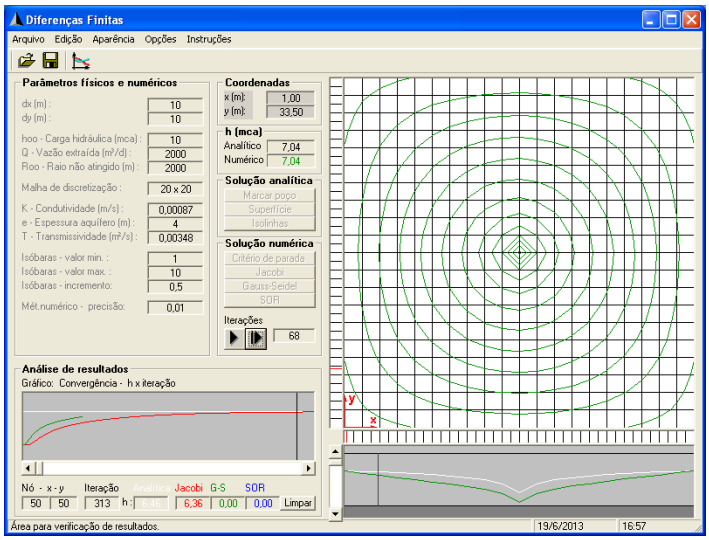

Figure 12 - Gauss method - Iteration 40.

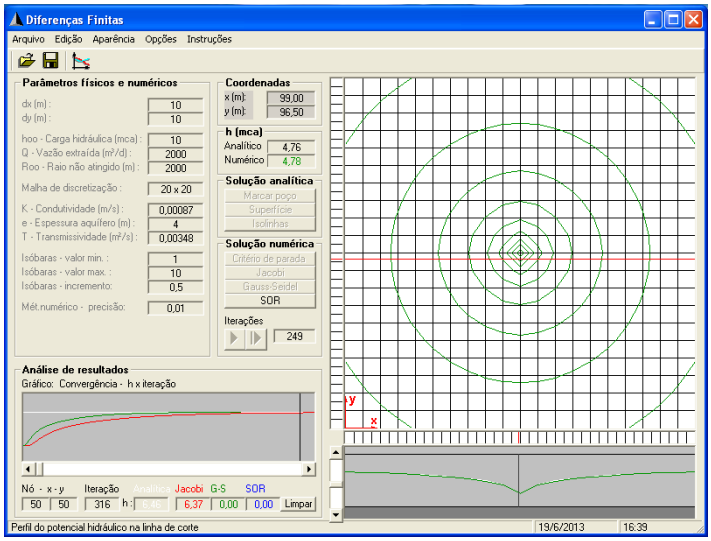

Figure 13 - Gauss method - Iteration 230.

The fastest result is obtained with the SOR algorithm, in this application 84 iterations were needed to reach the final solution. From Figure 14 we can see the result on the window contours map, the vertical profile in the bottom window and in the graph of analysis of results we can verify the convergence to the solution for each iteration with blue lines for the SOR algorithm, green lines for the Gauss algorithm and red lines for the Jacobi algorithm.

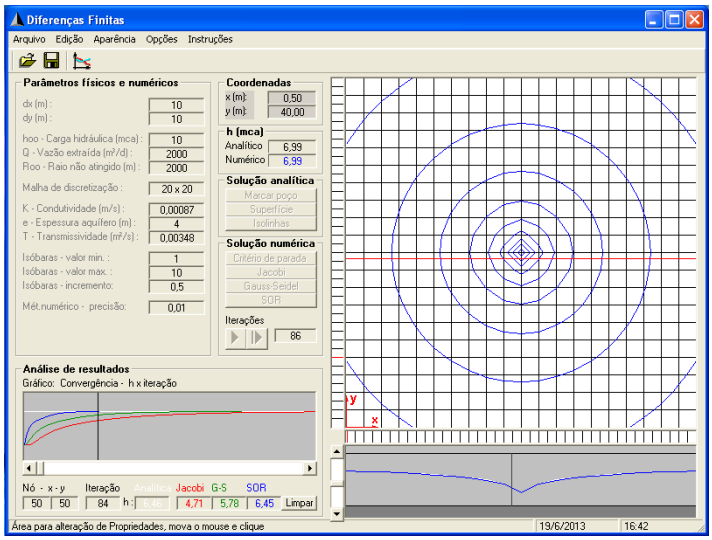

Figure 13 - Gauss method - Result after 84 iterations.

4. CONCLUSIONS

The DIFIN program consists in a basic tool for implementation and comparison of iterative methods for numerical solution of differential equations using Finite Differences. With the intention to demonstrate the applicability and efficiency of the Finite Difference Method in solving differential equations was used a known model and also known analytical solution for comparison purposes . The Finite Difference Method achieves satisfactory approximations to the exact solution (analytical solution) and the algorithms used have adequate convergence. Comparing the algorithms, is possible realize the efficiency of the SOR method in fast attainment of the solution, however is need to evaluate the coefficient of relaxation to avoid the lack of convergence and numerical dispersion . The computational tool provides a wide range of parameters and optimal visualization of results and is also an excellent teaching tool.

REFERENCES

CUMINATO, J.A., MENEGUETE M. “Discretização de Equações Diferenciais Parciais: Técnicas de Diferenças Finitas”, ICMC/USP, p203. 1999

FORTUNA, A. O. Técnicas Computacionais para Dinâmica dos Fluidos – Conceitos Básicos e Aplicações, São Paulo: Editora USP, 426 p. 2000.

ICHIRO, K. MAKATO, N. MITSURU, K. Groundwater Engineering:Recent Advances. Proceedings of the International Symposium on Groundwater Problems Related to Geo-environment, Okayama, Japan, 28-30 May 2003

JELEN, B. SYRSTAD, T. VBA e Macros para Microsoft Office Excel. 1ª Ed. Rio de Janeiro: Prentice Hall. 2008.

MARGOLIN, G. BERKOWITZ, B.; SCHER, H. Structure, flow, and generalized conductivity scaling in fractured networks - Water Resource Research v34, n9, pp.2103-2121. 1998.