da tese defendida por Adriano Lima Ribeiro e aprovada pela comissão julgadora em 24/06/1997

ORIENTADOR

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA

Desenvolvimento de Programa Computacional para Interpretação de Testes de Bombeamento de Aqüífero

Autor : **Adriano Lima Ribeiro**
Orientador: **Antônio Cláudio de França Corrêa**

01/97

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
DEPARTAMENTO DE ENGENHARIA DE PETRÓLEO

Desenvolvimento de Programa Computacional para Interpretação de Testes de Bombeamento de Aqüífero

Autor : **Adriano Lima Ribeiro**
Orientador: **Antônio Cláudio de França Corrêa**

Curso: Engenharia Mecânica.
Área de concentração: Estudos de Reservatórios

Dissertação de mestrado apresentada à comissão de Pós Graduação da Faculdade de Engenharia Mecânica, como requisito para obtenção do título de Mestre em Engenharia Mecânica.

Campinas, 1997
S.P. - Brasil

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

R354d

Ribeiro, Adriano Lima
Desenvolvimento de programa computacional para interpretação de testes de bombeamento de aqüífero / Adriano Lima Ribeiro.--Campinas, SP: [s.n.], 1997.

Orientador: Antônio Cláudio de França Corrêa
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Mecânica.

1. Águas subterrâneas - Escoamento. 2. Aqüíferos. 3. Hidrogeologia. 4. Poços. 5. Reservatórios subterrâneos. I. Corrêa, Antônio Cláudio de França. II. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. III. Título.

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
DEPARTAMENTO DE ENGENHARIA DE PETRÓLEO

DISSERTAÇÃO DE MESTRADO

Desenvolvimento de Programa Computacional para Interpretação de Testes de Bombeamento de Aqüífero

Autor : **Adriano Lima Ribeiro**
Orientador: **Antônio Cláudio de França Corrêa**

Prof. Dr. Antônio Cláudio de França Corrêa, Presidente
UNICAMP

Prof. Dr.Álvaro Marcello Marco Peres
PETROBRÁS

Prof. Dr.Denis José Schiozer
UNICAMP

Campinas, 24 de Junho de 1997

Agradecimentos

Ao Professor da UNICAMP Antônio Cláudio de França Corrêa, pela orientação fornecida.

Ao Professor Ernesto da Silva Pitombeira, da Universidade Federal do Ceará, também pela orientação fornecida.

Ao geólogo Fernando Antônio Carneiro Feitosa, da Fundação Cearense de Meteorologia e Recursos Hídricos - FUNCEME, pela orientação e fornecimento de softwares demonstrativos.

À PETROBRÁS pela bolsa de estudo concedida.

Aos colegas de trabalho da Companhia de Gestão dos Recursos Hídricos do Ceará, pela compreensão, incentivo e orientação com base em suas experiências. Especialmente a José Carlos de Araújo Filho e a Assis de Sousa Filho.

A todos os colegas, professores e funcionários do Departamento de Engenharia de Petróleo e Centro de Estudos de Petróleo da UNICAMP, pelo estímulo e amizade.

Resumo

O presente trabalho versa sobre o desenvolvimento de um software destinado a auxiliar na interpretação de testes de bombeamento realizados em aqüíferos. Para isto ele utiliza modelos cujas incógnitas correspondem a parâmetros do aqüífero que se deseja conhecer. Os modelos empregados são: Theis (1935), Hantush e Jacob (1955), Boulton (1963) e Neuman (1972). O "software", fruto de um trabalho de mestrado, é uma ferramenta numérica e gráfica de apoio, que procura ajustar a curva da função do modelo a pontos de rebaixamento versus tempo, obtidos do teste realizado em campo. O método numérico de ajuste empregado foi o de Levemberg-Marquardt associado a uma busca restrita de parâmetros. Os resultados obtidos com a aplicação desta ferramenta a casos analisados na literatura especializada têm se mostrados satisfatórios.

Abstract

The present work is about the development of a software destined to help in the interpretation of pumping tests realized in aquifers. To make this it utilizes models of which unkown parameters correspond to aquifer parameters that someone want to know. The models employed are: Theis (1935), Hantush and Jacob (1955), Boulton (1963) and Neuman (1972). The software, result of a master course, is a numerical and graphical support tool, that tries to fit the model function curve to points of drawdown versus time obtained from tests realized in the field. The numerical fit method employed was the Levemberg-Marquardt method associated with a restricted search of parameters. The results obtained with the application of this tool to cases analised in the specialized literature have manifested satisfatories.

Índice

1 Introdução

Apesar da grande importância dada à água superficial, a água subterrânea pode ser solução econômica e tecnicamente mais viável em muitos dos casos de abastecimento. No contexto da região Nordeste, onde predomina o clima semi-árido, este recurso poderia ser muito melhor explorado, especialmente nas localidades onde ocorre aqüíferos sedimentares, de maneira a minimizar em muito o problema da falta de água. Para que esta exploração se dê de forma racional, é importante conhecer e modelar o fluxo das águas subterrâneas. Observa-se, então, que este processo de fluxo em meios porosos guarda estreita relação com o conteúdo de disciplinas sobre reservatórios, do curso de mestrado em engenharia de petróleo, para o qual foi desenvolvida esta dissertação.

O conhecimento de características médias de um aqüífero sedimentar, como a transmissividade e o coeficiente de armazenabilidade, é de importância fundamental para uma explotação racional do mesmo, sem mesnosprezar o conhecimento geológico, que deve mesmo preceder a avaliação das características (parâmetros hidrogeológicos) mencionadas. Neste sentido o teste de bombeamento é de grande valia, visto conseguir inferir, mediante a análise dos seus resultados, valores médios que representam uma grande extensão superficial do aqüífero. Esta análise normalmente é realizada buscando-se um ajuste entre um gráfico dos pontos de rebaixamento em um poço de observação versus tempo, e um gráfico das curvas tipo que representam uma determinada modelagem matemática do aqüífero. Diversos programas computacionais foram desenvolvidos como ferramentas gráficas para agilizar e dar mais precisão a esta operação, incluindo técnicas numérias para ajuste de parâmetros de modelos e delimitação de faixas de erro. O valores obtidos auxiliam na determinação da viabilidade econômica, na vazão e distância entre os poços de explotação.

Esta dissertação tem justamente como objetivo apresentar um programa computacional, desenvolvido como trabalho do curso de mestrado mencionado, com a finalidade acima descrita. Este programa foi denominado por ITBA, para abreviar a expressão "interpretador de teste de bombeamento em aqüífero". A idéia nasceu de prosseguir com o desenvolvimento de um programa já existente, denominado por AQUATEST, devido aos autores Milton de Oliveira Santos Júnior, quanto a parte de cálculo numérico, escrita em FORTRAN, e Carlos Eduardo Pereira, quanto a parte interativa e gráfica, que foi desenvolvida em linguagem C para ambiente DOS. O AQUATEST foi desenvolvido com recursos do Centro de Estudos de Petróleo (CEPETRO) da UNICAMP. O programa ITBA conservou-se do programa AQUATEST apenas a ferramenta de desenvolvimento em linguagem C, pois o código do programa AQUATEST foi aos poucos completamente substituído.

Como o AQUATEST possuía implementado apenas o modelo de Theis, que será comentado sucintamente no Capítulo 2, o plano inicial consistia em introduzir outros modelos de aqüífero. Porém, ao longo do desenvolvimento e da análise de outros programas já existentes, incluindo o trabalho de SUGAHARA (1996), foram surgindo novas idéias, ou objetivos a serem alcançados. Um deles consistiu em substituir uma concepção do programa original (AQUATEST), em que o conjunto dos pontos de dados do teste de bombeamento possuia mobilidade na tela, permitindo um ajuste com o gráfico de uma curva tipo do modelo, gráfico este que permanecia fixo, pela concepção de se fixar os pontos em um gráfico e se ter uma curva de modelo flexível (ou ajustável). Isto simplifica a realização da parte do programa responsável pelos gráficos e interação com o usuário. Também possibilita uma flexibilidade durante a fase de ajuste visual, a de se poder trabalhar com modelos envolvendo mais do que três parâmetros ajustáveis. Esta situação, no caso do ajuste dos dados a curvas tipo, oferece algumas restrições e dificuldades. Observa-se, ainda, que a concepção escolhida permite estudar com mais precisão o comportamento da curva do modelo, podendo-se fazer extrapolações para os dados do teste.

Enfim, procurou-se desenvolver um programa que permitisse ao usuário entrar com o nome de um arquivo contendo as coordenadas dos pontos do teste de bombeamento, selecionar um modelo dentre as opções fornecidas (que são atualmente em número de quatro), definir os limites do gráfico (que é bilogarítmico), ou deixar que sejam definidos automaticamente, que permita entrar diretamente com valores para os parâmetros dos modelos, e que dentre estes, os parâmetros ajustáveis possam ser incrementados ou decrementados de forma prática de um certo percentual, também definido pelo usuário. Além disso, o programa desenvolvido é capaz de realizar um ajuste automático de parâmetros, a partir de uma aproximação inicial, pelo método numérico de Levenberg-Marquardt, o qual está descrito em PRESS et al. (1989). Condições de convergência relativas a este ajuste podem ser fornecidas pelo usuário, assim como o ajuste também pode ser realizado iteração por iteração. O usuário também pode estabelecer limites (faixa) para variação de cada parâmetro ajustável, aprimorando o método de Levenberg-Marquardt. É possível ainda estabelecer o desvio padrão de cada ponto do teste, permitindo, deste modo, controlar a influência de cada um no resultado do ajuste. A seleção dos pontos para definição do desvio padrão é feita de forma prática, tendo-se uma visualização permanente dos pontos com desvios padrões modificados. Além do mais, dentro das limitações impostas pela ferramenta de desenvolvimento, que opera em ambiente DOS, procurou-se fazer com que o programa apresentasse uma interface gráfica agradável, a qual inclui uma barra de comandos que indica quais as teclas que estão ativas naquele momento, cada qual associada a uma palavra que lembre sua função.

A estrutura deste texto ficou da seguinte forma: O Capítulo 2 descreve sobre a fundamentação numérica do software desenvolvido, e comenta sucintamente sobre os modelos com que ele trabalha. O Capítulo 3 é um manual de utilização do software, por meio do qual pode-se conhecer mais minuciosamente os recursos que ele oferece, especialmente na sua parte interativa e gráfica. O Capítulo 4 analisa os resultados da aplicação do programa a casos relatados na literatura. Também são mostradas aplicações a dados sintéticos com o intuito de se fazer uma validação do programa. O Capítulo 5 faz uma avaliação de todo o trabalho e dá sugestões para sua continuidade. O Anexo A apresenta a listagem dos arquivos fontes que formam o programa ITBA. O Anexo B traz a listagem de alguns arquivos demonstrativos, com

dados de rebaixamento versus tempo, que foram utilizados em aplicações do Capítulo 5. O Apêndice faz uma análise e tece comentários sobre outros programas similares existentes, e que influenciaram na concepção do programa ITBA, objeto deste trabalho.

Note-se bem que o problema a ser resolvido no curso de mestrado em pauta, assunto desta dissertação, foi o desenvolvimento do programa com a finalidade e as características acima descrita, e não propriamente a análise de alguma aplicação particular do programa, tornando este apenas uma etapa do processo. Na verdade, se este fosse o caso, poderia se ter recorrido a algum programa similar já existente.

2 Fundamentação Numérica

As funções (ou procedimentos) de cálculo numérico estão no arquivo num.c listado no Anexo A. Boa parte delas foram extraídos de PRESS et al. (1989): *bessj0, bessi0, bessi1, besskO, besskl, gaussj, jacobi, mrqcof e mrqmin.*

A função *mrqmin* é responsável pelo ajuste dos parâmetros dos modelos pelo método de Levenberg-Marquardt. Ela possui a flexibilidade de se poder especificar o desvio padrão da ordenada de cada ponto de dado, como também de se especificar aleatoriamente quais os parâmetros a serem ajustados e quais devem permanecer fixos. Cada chamada realiza uma iteração do método e permite que se acompanhe a evolução de χ^2 (chi-quadrado, parâmetro relacionado a qualidade do ajuste, quanto menor melhor), da matriz de curvatura (mais precisamente uma aproximação), do fator λ (que define a participação do procedimento de se deslocar na direção da maior declividade descendente), e da inversa da matriz de curvatura modificada por λ, denominada por $[\alpha']^{-1}$. Assim cada iteração do ajuste automático é realizada por uma chamada a função *mrqmin*, onde cada chamada promove um ajuste simultâneo de todos os parâmetros ajustáveis, podendo-se opcionalmente fixar uma quantidade qualquer deles em valores pré-fixados. O critério de parada das iterações está relacionado ao valores do χ^2 e de λ gerados em cada iteração. Se χ^2 se tornar menor que um limite máximo estabelecido bem menor que 1, ou a variação de χ^2 relativa a iteração anterior se tornar menor que um percentual bem pequeno estabelecido, e se além de satisfazer a uma das condições anteriores λ tiver decrescido nas duas últimas iterações, as iterações podem parar. Uma última iteração pode ainda ser realizada, colocando λ igual a 0, para que *mrqmin* forneça a matriz de covariância dos parâmetros ajustados, agora sem a influência de λ e com uma precisão satisfatória. A função *mrqmin* é auxiliada pelas funções *mrqcof* e *gaussj*, esta última responsável pelo cálculo das

raízes e da inversa da matriz dos coeficientes de um sistema de equações lineares. A função *mrqcof* calcula a matriz de curvatura e o gradiente de χ^2.

Foi posto como comentário em *mrqmin* um código alternativo, que substitui o cálculo original do vetor de deslocamento para correção dos parâmetros, e de $[\alpha']^{-1}$. Tal código emprega a função *jacobi* descartando a função *gaussj*, onde jacobi é responsável pelo cálculo dos autovalores e autovetores normalizados de uma matriz simétrica. Esta matriz é definida no código alternativo como $[\alpha']$ escalonada. As equações 9 a 12 de ROSA & HORNE (1983) apresentam como foi realizado este escalonamento, que é uma forma de aperfeiçoar os aspectos numéricos da solução do sistema de equações linearizadas. A solução deste sistema foi realizada tomando-se uma decomposição espectral de $[\alpha']$, conforme pode ser visto nas equações 16 a 18 de ORELLANA & CORRÊA (1992). A vantagem desta solução reside em poder se eliminar autovalores próximos de zero, que significam insensibilidade do modelo matemático às correspondentes direções de busca. Entretanto, como tal alternativa de código não foi ainda suficientemente testada, optou-se por manter ativo no programa o código original de PRESS et al. (1989).

Foi realizada ainda a inclusão de algumas linhas em *mrqmin*, as quais foram mantidas ativas, com finalidade de fazer com que a busca da solução do vetor de parâmetros ajustáveis, denominado por **a**, se processasse sempre dentro de faixas estabelecidas para cada parâmetro. Assim, se calcula o maior valor, limitado a 1, para um fator positivo, denominado por f, o qual multiplicará o vetor de deslocamento original, para a atualização de **a**, de forma a manter este vetor dentro dos limites estabelecidos. Na realidade, quando f é menor do que 1, o mesmo é multiplicado por uma constante próxima de 1 (igual a 0,99999), que evita problemas relacionados a erros de truncamento do código inserido. Esta inclusão é fundamental porque determinados parâmetros dos modelos, se postos negativos ou nulos, geraria condições de erro. Além disto, a impotância desta inclusão reside em que a busca pode ser mais eficiente, quando se conhece melhor a situação que se está modelando, de forma a se poder restringir mais o intervalo de variação de cada parâmetro ajustável.

As funções *bessj0*, *bessi0*, *bessi1*, *besskO* e *besskl* reproduzem respectivamente as função de Bessel J_0, e as funções de Bessel modificadas I_0, I_1, K_0 e K_1. Estas funções serão utilizadas pelas funções dos modelos discutidas a seguir.

Para finalizar a discussão sobre as funções *bessj0, bessi0, bessi1, besskO, besskl, gaussj, jacobi, mrqcof e mrqmin,* estraídas de PRESS et al. (1989), informa-se que outras modificações foram introduzidas nos códogos originais. Assim, a linguagem foi convertida de Pascal para C, e as funções (*procedures* do Pascal) que possuíam mensagens de erro foram convertidas para funções inteiras, que retornam um código de erro. Estas funções são *gaussj*, *jacobi* e *mrqmin.* O código zero significa ausência de erro. Naturalmente *mrqmin*, que chama *gaussj* ou *jacobi*, passa a conter agora alguma manipulação do código de erro gerado pelas funções chamadas.

De um modo geral as funções dos modelos geram seus resultados a partir das equações dos modelos e suas derivadas no espaço de Laplace, pois uma propriedade nos diz que a transformada de Laplace da derivada de uma função com relação a um parâmetro, que não o da transformação, é igual a derivada da transformada com relação ao mesmo parâmetro. Veremos adiante como estas equações foram obtidas. Por enquanto, observaremos como foi realizada a inversão para o espaço real. Esta operação foi feita empregando-se o algoritmo de STEHFEST (1970), cuja equação pode ser vista a seguir:

$$F_a = \frac{\ln 2}{t} \sum_{i=1}^{N} V_i f\left(\frac{\ln 2}{t} i\right). \qquad (1)$$

F_a é o valor aproximado de F(t), cuja transformada de Laplace é f(w). Os coeficientes V_i são dados por:

$$V_i = (-1)^{N/2+i} \sum_{K=INT[(i+1)/2]}^{MIN(i,N/2)} \frac{K^{N/2+1}(2K)!}{(N/2-K)!(K!)^2(i-K)!(2K-i)!}, \qquad (2)$$

onde N é um inteiro par relacionado à precisão da máquina, ou seja, ao número de dígitos da representação em ponto flutuante. Alguns testes levaram a definir N igual a 10 no programa, no qual todas as variáveis reais foram definidas com dupla precisão. INT é a função que retorna a parte inteira do argumento, enquanto que MIN retorna o menor dos dois argumentos. Os valores de V_i e de outros parâmetros são calculados na função *inivar* (inicialização de variáveis) do programa.

Observa-se ainda, que qualquer dos modelos descritos a seguir supõe um sistema de unidades qualquer, desde que derivado das mesmas unidades básicas de comprimento e tempo elevadas a expoentes variados.

Iniciaremos pelo modelo de Theis, que é implementado por uma função de mesmo nome no programa. Este modelo é descrito minuciosamente por WALTON (1970, p. 129 a 133). Ele considera um aqüífero homogêneo e isotrópico limitado por dois planos horizontais, mas com extensão radial infinita, e confinado, isto é, está sempre saturado e não é possível o fluxo de água pelas fronteiras do aqüífero. Este modelo também supõe equilíbrio estático após o qual inicia-se um bombeamento a vazão constante, por um poço vertical, de diâmetro desprezível, que atravessa todo o aqüífero (fluxo horizontal). Resumindo, este modelo é destinado a fluxo radial não permanente em aqüífero confinado, que é bombeado a vazão constante por um poço que o penetra completamente. Abaixo podemos ver a equação diferencial do modelo e suas condições inicial e de contorno:

$$\frac{\partial^2 s}{\partial r^2} + \frac{1}{r}\frac{\partial s}{\partial r} = \frac{S}{T}\frac{\partial s}{\partial t} \tag{3}$$

$$s(r,0) = 0 \tag{4}$$

$$s(\infty, t) = 0 \tag{5}$$

$$\lim_{r \to 0}\left(r \frac{\partial s}{\partial r} \right) = -\frac{Q}{2\pi T} \tag{6}$$

Nestas equações s é o rebaixamento da superfície potenciométrica no tempo t, a uma distância r do poço de bombeamento, o qual é tratado como se possuísse um diâmetro desprezível. S é o coeficiente de armazenabilidade e T é a transmissividade. O bombeamento tem início no tempo t igual a zero e sua vazão vale Q.

Aplicando-se a transformada de Laplace à equação diferencial, e trabalhando-se com as condições inicial e de contorno, chega-se ao seguinte resultado:

$$\bar{s} = \frac{Q}{2\pi T} \frac{K_0\left(r\sqrt{\frac{S}{T} w} \right)}{w}, \tag{7}$$

onde a barra denota que a variável correspondente está no campo de Laplace. Então, temos:

$$\frac{\partial \bar{s}}{\partial T} = \frac{Q}{2\pi T^2} \frac{\frac{1}{2} K_1\left(r\sqrt{\frac{S}{T} w} \right) r\sqrt{\frac{S}{T} w} - K_0\left(r\sqrt{\frac{S}{T} w} \right)}{w} \tag{8}$$

$$\frac{\partial \bar{s}}{\partial S} = -\frac{Q}{4\pi TS} \frac{K_1\left(r\sqrt{\frac{S}{T} w} \right) r\sqrt{\frac{S}{T} w}}{w} \tag{9}$$

O modelo seguinte, modelo de Hantush, é descrito minuciosamente por WALTON (1970), e implementado em nosso programa por uma função com o mesmo nome do modelo. Este guarda muita similaridade com o modelo anterior, distinguindo-se, porém, por ser o aqüífero limitado acima por um aquitard (formação de permeabilidade reduzida) saturado, homogêneo, de espessura constante, e cuja permeabilidade é bem menor que a do aqüífero. Logo acima do aquitard existe um outro aqüífero, que alimenta por meio do aquitard o aqüífero inferior, quando este é bombeado. Supõe-se neste modelo que não haja variação no conteúdo de água do aquitard, sendo toda contribuição para o aqüífero inferior provinda do superior, o qual também se supõe possuir carga hidráulica sempre constante. Diz-se que o aqüífero inferior é semi-confinado porque o mesmo está sempre saturado, e porque pode haver um pequeno fluxo através de sua fronteira natural. Resumindo, este modelo é destinado a fluxo radial não permanente em aqüífero confinado com vazamento (semi-confinado), que é bombeado a vazão constante por um poço que o penetra completamente, não ocorrendo liberação de água do aquitard. A seguir podemos ver a equação diferencial do modelo, que está sujeita às mesmas condições inicial e de contorno do modelo anterior, visto que estes modelos possuem uma única dimensão espacial (r).

$$\frac{\partial^2 s}{\partial r^2} + \frac{1}{r}\frac{\partial s}{\partial r} - \frac{s}{B^2} = \frac{S}{T}\frac{\partial s}{\partial t} \tag{10}$$

$B^2=T/(P'/m')$, onde P' é a permeabilidade do aquitard e m' a sua espessura.

Resolvendo a equação diferencial no espaço de Laplace obtemos:

$$\bar{s} = \frac{Q}{2\pi T}\frac{K_0\left(\sqrt{r^2\frac{S}{T}w + \left(\frac{r}{B}\right)^2}\right)}{w} \tag{11}$$

Derivando-se esta equação em relação aos parâmetros ajustáveis T, S e r/B temos:

$$\frac{\partial \bar{s}}{\partial T} = \frac{Q}{2\pi T^2} \frac{\frac{1}{2} K_1\left(\sqrt{r^2 \frac{S}{T} w + \left(\frac{r}{B}\right)^2}\right) r^2 \frac{S}{T} w \Big/ \sqrt{r^2 \frac{S}{T} w + \left(\frac{r}{B}\right)^2} - K_0\left(\sqrt{r^2 \frac{S}{T} w + \left(\frac{r}{B}\right)^2}\right)}{w} \quad (12)$$

$$\frac{\partial \bar{s}}{\partial S} = -\frac{Q}{4\pi S T} \frac{K_1\left(\sqrt{r^2 \frac{S}{T} w + \left(\frac{r}{B}\right)^2}\right) r^2 \frac{S}{T} w \Big/ \sqrt{r^2 \frac{S}{T} w + \left(\frac{r}{B}\right)^2}}{w} \quad (13)$$

$$\frac{\partial \bar{s}}{\partial (r/B)} = -\frac{Q(r/B)}{2\pi T} \frac{K_1\left(\sqrt{r^2 \frac{S}{T} w + \left(\frac{r}{B}\right)^2}\right) \Big/ \sqrt{r^2 \frac{S}{T} w + \left(\frac{r}{B}\right)^2}}{w} \quad (14)$$

Passando ao próximo modelo, o modelo de Boulton, este é implementado por uma função de mesmo nome que do modelo no programa ITBA, e se acha descrito em WALTON (1970). Este modelo subentende um aqüífero de extensão horizontal infinita, limitado abaixo por um plano horizontal impermeável, possuindo inicialmente uma espessura saturada constante, a qual está em contato com o ar atmosférico do solo (aqüífero livre), ou seja, esta superfície está submetida a pressão atmosférica. O aqüífero, considerado homogêneo e isotrópico, a partir do instante inicial é bombeado a uma vazão constante, por um poço vertical de diâmetro desprezível, aberto ao longo de toda a extensão saturada do aqüífero. O rebaixamento da superfície livre, promovido pelo bombeamento, também é considerada pequena pelo modelo, quando comparada a espessura inicialmente saturada. Resumindo, este modelo subentende fluxo em aqüífero livre e isotrópico, com poço penetrando completamente o aqüífero, o qual é bombeado a vazão constante. A seguir temos a equação diferencial, que também está sujeita às mesmas condições inicial e de contorno dos modelos anteriores.

$$T\left(\frac{\partial^2 s}{\partial r^2} + \frac{1}{r}\frac{\partial s}{\partial r}\right) = S\frac{\partial s}{\partial t} + D_i S_y \int_0^t \frac{\partial s}{\partial \tau} e^{-D_i(t-\tau)} d\tau \quad (15)$$

D_i é o inverso do índice de retardo, um parâmetro empírico, e S_y é o coeficiente de armazenabilidade associado à drenagem gravitacional dos interstícios.

Solucionando-se a equação diferencial anterior no espaço de Laplace obtemos:

$$\bar{s} = \frac{Q}{2\pi T} \frac{K_0\left(\sqrt{r^2 \frac{S}{T} w + \frac{1}{\frac{\sigma T}{r^2 S w} + \frac{1}{(r/D_t)^2}}}\right)}{w}, \tag{16}$$

onde $\sigma = \frac{S}{S_y}$ e $D_t = \sqrt{\frac{T}{D_i S_y}}$.

Derivando-se a equação anterior com respeito aos parâmetros ajustáveis T, S, r/D_t e σ temos:

$$\frac{\partial \bar{s}}{\partial T} = \frac{Q}{2\pi T^2} \frac{1}{w} \left\{ \frac{1}{2} \frac{K_1\left(\sqrt{r^2 \frac{S}{T} w + \frac{1}{\frac{\sigma T}{r^2 S w} + \frac{1}{(r/D_t)^2}}}\right)\left(r^2 \frac{S}{T} w + \left(\frac{1}{\frac{\sigma T}{r^2 S w} + \frac{1}{(r/D_t)^2}}\right)^2 \frac{\sigma T}{r^2 S w}\right)}{\sqrt{r^2 \frac{S}{T} w + \frac{1}{\frac{\sigma T}{r^2 S w} + \frac{1}{(r/D_t)^2}}}} - \right.$$

$$\left. K_0\left(\sqrt{r^2 \frac{S}{T} w + \frac{1}{\frac{\sigma T}{r^2 S w} + \frac{1}{(r/D_t)^2}}}\right) \right\} \tag{17}$$

$$\frac{\partial \bar{s}}{\partial S} = -\frac{Q}{4\pi ST} \frac{K_1\left(\sqrt{r^2\frac{S}{T}w + \frac{1}{\frac{\sigma T}{r^2 Sw} + \frac{1}{(r/D_t)^2}}}\right)\left(r^2\frac{S}{T}w + \left(\frac{1}{\frac{\sigma T}{r^2 Sw} + \frac{1}{(r/D_t)^2}}\right)^2 \frac{\sigma T}{r^2 Sw}\right)}{w\sqrt{r^2\frac{S}{T}w + \frac{1}{\frac{\sigma T}{r^2 Sw} + \frac{1}{(r/D_t)^2}}}} \quad (18)$$

$$\frac{\partial \bar{s}}{\partial (r/D_t)} = -\frac{Q}{2\pi T (r/D_t)^3} \frac{K_1\left(\sqrt{r^2\frac{S}{T}w + \frac{1}{\frac{\sigma T}{r^2 Sw} + \frac{1}{(r/D_t)^2}}}\right)\left(\frac{1}{\frac{\sigma T}{r^2 Sw} + \frac{1}{(r/D_t)^2}}\right)^2}{w\sqrt{r^2\frac{S}{T}w + \frac{1}{\frac{\sigma T}{r^2 Sw} + \frac{1}{(r/D_t)^2}}}} \quad (19)$$

$$\frac{\partial \bar{s}}{\partial \sigma} = \frac{Q}{4\pi T} \frac{K_1\left(\sqrt{r^2\frac{S}{T}w + \frac{1}{\frac{\sigma T}{r^2 Sw} + \frac{1}{(r/D_t)^2}}}\right)\left(\frac{1}{\frac{\sigma T}{r^2 Sw} + \frac{1}{(r/D_t)^2}}\right)^2}{r^2\frac{S}{T}w^2\sqrt{r^2\frac{S}{T}w + \frac{1}{\frac{\sigma T}{r^2 Sw} + \frac{1}{(r/D_t)^2}}}} \quad (20)$$

O quarto modelo implementado pelo prgrama ITBA é modelo de NEUMAN (1972). Este modelo é similar ao anterior, porém considera o aqüífero anisotrópico, com as componentes horizontal e vertical da permeabilidade constantes. Atualmente este modelo é mais bem aceito que o de Boulton, pois não recorre a nenhum parâmetro empírico, como o D_i do modelo de Boulton, para modelar um teste de bombeamento em um aqüífero livre. De qualquer forma, o modelo de Boulton pode ser ainda ser muito utilizado no programa ITBA: Devido a solução numérica do modelo de Neuman ser bem mais lenta que a dos modelos anteriores, pode-se

recorrer ao modelo de Boulton para um ajuste inicial dos parâmetros que este tem em comum com o modelo em questão, o que facilita a fase de ajuste do modelo de Neuman. A seguir observamos a equação diferencial e as condições inicial e de contorno que descrevem o modelo.

$$\frac{\partial^2 s}{\partial r^2} + \frac{1}{r}\frac{\partial s}{\partial r} + K_D \frac{\partial^2 s}{\partial z^2} = \frac{1}{\alpha_s}\frac{\partial s}{\partial t}, \tag{21}$$

onde $0 < z < b$

$$s(r, z, 0) = 0 \tag{22}$$

$$s(\infty, z, t) = 0 \tag{23}$$

$$\frac{\partial s}{\partial z}(r, 0, t) = 0 \tag{24}$$

$$\frac{\partial s}{\partial z}(r, b, t) = -\frac{1}{\alpha_y}\frac{\partial s}{\partial t}(r, b, t) \tag{25}$$

$$\lim_{r \to 0} \int_0^b r \frac{\partial s}{\partial r} dz = -\frac{Q}{2\pi K_r} \tag{26}$$

A equação 25 supõe que o rebaixamento é pequeno quando comparado a espessura b do aqüífero inicialmente saturado. A última equação supõe, como nos modelos anteriores, que o diâmetro do poço de bombeamento é desprezível. $K_D = K_z/K_r$, onde K_z é a permeabilidade vertical e K_r é a permeabilidade horizontal. z é a altura em relação ao fundo do aqüífero. Como temos componente vertical no fluxo, s varia ao longo da direção vertical em um mesmo instante, e corresponde, então, ao decréscimo do nível potenciométrico a partir do instante inicial. $\alpha_s = K_r/S_s$, onde S_s é a armazenabilidade específica ($S = S_s b$). E $\alpha_y = K_z/S_y$.

Seguindo-se os procedimentos descritos no Apêndice A de NEUMAN (1972) chegamos a seguinte solução para o modelo, onde s^* é o rebaixamento s submetido à transformada de Laplace seguida pela transformada de Hankel.

$$s^* = \frac{Q}{2\pi T K_D p \eta^2}\left(1 - \frac{p\cosh(\eta z)}{\alpha_y \eta \operatorname{senh}(\eta b) + p\cosh(\eta b)}\right) \tag{27}$$

$\eta^2 = \left(a^2 / K_D\right) + (p / \alpha_s K_D)$, a é o parâmetro da transformada de Hankel e p representa a variável da transformada de Laplace.

Fazendo-se a consideração de que o poço de observação é perfurado ao longo de toda a extensão vertical do aqüífero, segundo NEUMAN (1972, eq. 19) o rebaixamento observado é dado por

$$s(r,t) = \frac{1}{b}\int_0^b s(r,z,t)dz \tag{28}$$

Como $s(r,z,t) = L^{-1}\left[H^{-1}[s^*]\right]$, onde L^{-1} denota a inversão da transformada de Laplace e H^{-1} a inversão da transformada de Hankel, é possível realizar permutas das transformadas e da integral da equação acima, de forma que

$$s(r,t) = H^{-1}\left[L^{-1}\left[\frac{1}{b}\int_0^b s^* dz\right]\right] \tag{29}$$

Redefinindo-se s^* como a integral multiplicada por 1/b que aparece na equação anterior, e definindo-se $\beta = K_D r^2 / b^2$, como sugerido em NEUMAN (1973), após a solução da integral e algumas manipulações algébricas obtemos o seguinte resultado:

$$s^* = \frac{Q}{2\pi\left(a^2T + pS\right)}\left(\frac{1}{p} - \frac{1}{\sigma\left(a^2T + pS\right)/S + \left\{pr\sqrt{\left(a^2T + pS\right)/\beta T}\Big/\tanh\left[r\sqrt{\left(a^2T + pS\right)/\beta T}\right]\right\}}\right) \quad (30)$$

Devido a derivada de s em relação a um parâmetro ajustável, que não inclue t e r, ser igual às transformadas inversas de Laplace e de Hankel da derivada de s^* com respeito ao mesmo parâmetro, foram deduzidas as seguintes equações:

$$\frac{\partial s^*}{\partial T} = \frac{Q}{2\pi\left(a^2T + pS\right)}\left\{\left(\frac{1}{\sigma\left(a^2T + pS\right)/S + \left\{pr\sqrt{\left(a^2T + pS\right)/\beta T}\Big/\tanh\left[r\sqrt{\left(a^2T + pS\right)/\beta T}\right]\right\}}\right)^2 \cdot \right.$$

$$\left[\frac{\sigma a^2}{S} - \frac{p^2r^2S}{2\beta T^2}\left(\frac{1}{r\sqrt{\left(a^2T + pS\right)/\beta T}\tanh\left[r\sqrt{\left(a^2T + pS\right)/\beta T}\right]} + 1 - \frac{1}{\tanh^2\left[r\sqrt{\left(a^2T + pS\right)/\beta T}\right]}\right)\right]$$

$$\left. - \frac{a^2}{a^2T + pS}\left(\frac{1}{p} - \frac{1}{\sigma\left(a^2T + pS\right)/S + \left\{pr\sqrt{\left(a^2T + pS\right)/\beta T}\Big/\tanh\left[r\sqrt{\left(a^2T + pS\right)/\beta T}\right]\right\}}\right)\right\} \quad (31)$$

$$\frac{\partial s^*}{\partial S} = \frac{Q}{2\pi\left(a^2T + pS\right)}\left\{\left(\frac{1}{\sigma\left(a^2T + pS\right)/S + \left\{pr\sqrt{\left(a^2T + pS\right)/\beta T}\Big/\tanh\left[r\sqrt{\left(a^2T + pS\right)/\beta T}\right]\right\}}\right)^2 \cdot \right.$$

$$\left[-\frac{\sigma a^2 T}{S^2}+\frac{p^2 r^2}{2\beta T}\left(\frac{1}{r\sqrt{\left(a^2T+pS\right)/\beta T}\tanh\left[r\sqrt{\left(a^2T+pS\right)/\beta T}\right]}+1-\frac{1}{\tanh^2\left[r\sqrt{\left(a^2T+pS\right)/\beta T}\right]}\right)\right]$$

$$-\frac{p}{a^2T+pS}\left(\frac{1}{p}-\frac{1}{\sigma\left(a^2T+pS\right)/S+\left\{pr\sqrt{\left(a^2T+pS\right)/\beta T}\Big/\tanh\left[r\sqrt{\left(a^2T+pS\right)/\beta T}\right]\right\}}\right)\Bigg\} \quad (32)$$

$$\frac{\partial s^*}{\partial\beta}=-\frac{Qr^2p}{4\pi T\beta^2}\left(\frac{1}{\sigma\left(a^2T+pS\right)/S+\left\{pr\sqrt{\left(a^2T+pS\right)/\beta T}\Big/\tanh\left[r\sqrt{\left(a^2T+pS\right)/\beta T}\right]\right\}}\right)^2 \cdot$$

$$\left(\frac{1}{r\sqrt{\left(a^2T+pS\right)/\beta T}\tanh\left[r\sqrt{\left(a^2T+pS\right)/\beta T}\right]}+1-\frac{1}{\tanh^2\left[r\sqrt{\left(a^2T+pS\right)/\beta T}\right]}\right) \quad (33)$$

$$\frac{\partial s^*}{\partial\sigma}=\frac{Q}{2\pi S}\left(\frac{1}{\sigma\left(a^2T+pS\right)/S+\left\{pr\sqrt{\left(a^2T+pS\right)/\beta T}\Big/\tanh\left[r\sqrt{\left(a^2T+pS\right)/\beta T}\right]\right\}}\right)^2 \quad (34)$$

No programa a transformada inversa de Laplace de s^*, e de suas derivadas em relação aos parâmetros ajustáveis, é implementada pela função *HNeuman*. A implementação do modelo é complementada por outra função denominada de *Neuman*, que calcula s e suas derivadas, onde realiza a transformada inversa de Hankel sobre os resultados da função *HNeuman*. Esta transformada inversa é dada pela equação que segue, tirada do Apêndice C de NEUMAN (1972).

$$H^{-1}\left[F(a)\right]=f(r)=\int_0^\infty aJ_0(ra)F(a)da \quad (34)$$

A solução numérica da equação anterior foi encontrada subdividindo-se o intervalo de integração em sub-intervalos finitos, cujos pontos de divisão são os zeros de $J_0(ra)$. O programa realiza o somatório das integrais, limitando-se a um máximo número de termos correspondente aos duzentos primeiros sub-intervalos, até que $|\bar{s}_N - \bar{s}_{N-1}| < 10^{-2}|\bar{s}_N|$, onde vale uma consideração similar para as derivadas parciais. $\bar{s}_N = 0{,}5(s_N + s_{N-1})$ e s_N é o somatório das N primeiras integrais. A precisão 10^{-2} foi escolhida por ter sido o valor empregado em NEUMAN (1972), conforme lá está descrito no Apêndice D. Em cada sub-intervalo empregou-se como método de integração a quadratura de Gauss-Legendre com dez pontos. Para tanto foi feito uma adaptação do procedimento qgaus apresentado em PRESS et al. (1989). Vale salientar que os duzentos primeiros zeros de J_0 são calculados na função *inivar*, que serve para inicialização de variáveis, como as do algoritmo de Stehfest. Esta função por sua vez recorre a função *secantes*, a qual emprega uma metodologia de mesmo nome, descrita por Cunha (1993), para calcular zeros de funções. A função *secantes* chama, naturalmente, a função *bessj0*.

Para finalizar a discussão sobre a parte numérica do programa, resta ainda dizer, que a função *funcs* foi criada simplesmente para compatibilizar a chamada das funções dos modelos pela função *mrqcof*, obtida a partir de PRESS et al. (1989).

3 Interação com o Usuário

Neste Capítulo veremos como utilizar o programa, conhecendo detalhadamente os seus recursos. Também, serão fornecidas algumas informações sobre o arquivo tela.c, que contém o programa principal (função main) e é responsável pela interação com o usuário. Estas informações devem ser do interesse de qualquer um que deseje alterar o programa, consistindo basicamente de informações sucintas sobre as funções que integram o arquivo. Vale lembrar que os arquivos fontes do programa são somente dois, ambos listados no Anexo A, tendo sido um deles já discutido no Capítulo anterior, ficando então o tela.c para este Capítulo.

3.1 Manual de utilização

Ao executarmos o programa a tela aparece dividida em três regiões, ou janelas: A maior, mais acima, é para o gráfico. A seguinte, logo abaixo, contém o nome ou valor de diversas variáveis passíveis de serem editadas pelo usuário. Finalmente, em baixo, encontramos uma barra indicativa dos comandos ativos em dado instante, servindo também, em alguns casos, para dar alguma mensagem. A figura 1, a seguir, dá uma idéia da tela que está sendo comentada, porém sem uma reprodução das cores, ocorrendo algo similar com o restantes das figuras referentes ao ITBA, que apareceram até o final deste trabalho.

Os três comandos que aparecem inicialmente são denominados pelas palavras linha, edita e sai. O comando linha, acionado pelas setas verticais, seleciona a linha evidenciada da janela de variáveis, que corresponde a uma variável. A janela de variáveis "rola", ao se tentar ultrapassar o seu limite inferior, para que se possa selecionar outras variáveis não visíveis

inicialmente.Também, ocorrem mudanças na barra de comandos ao se selecionar uma linha diferente.

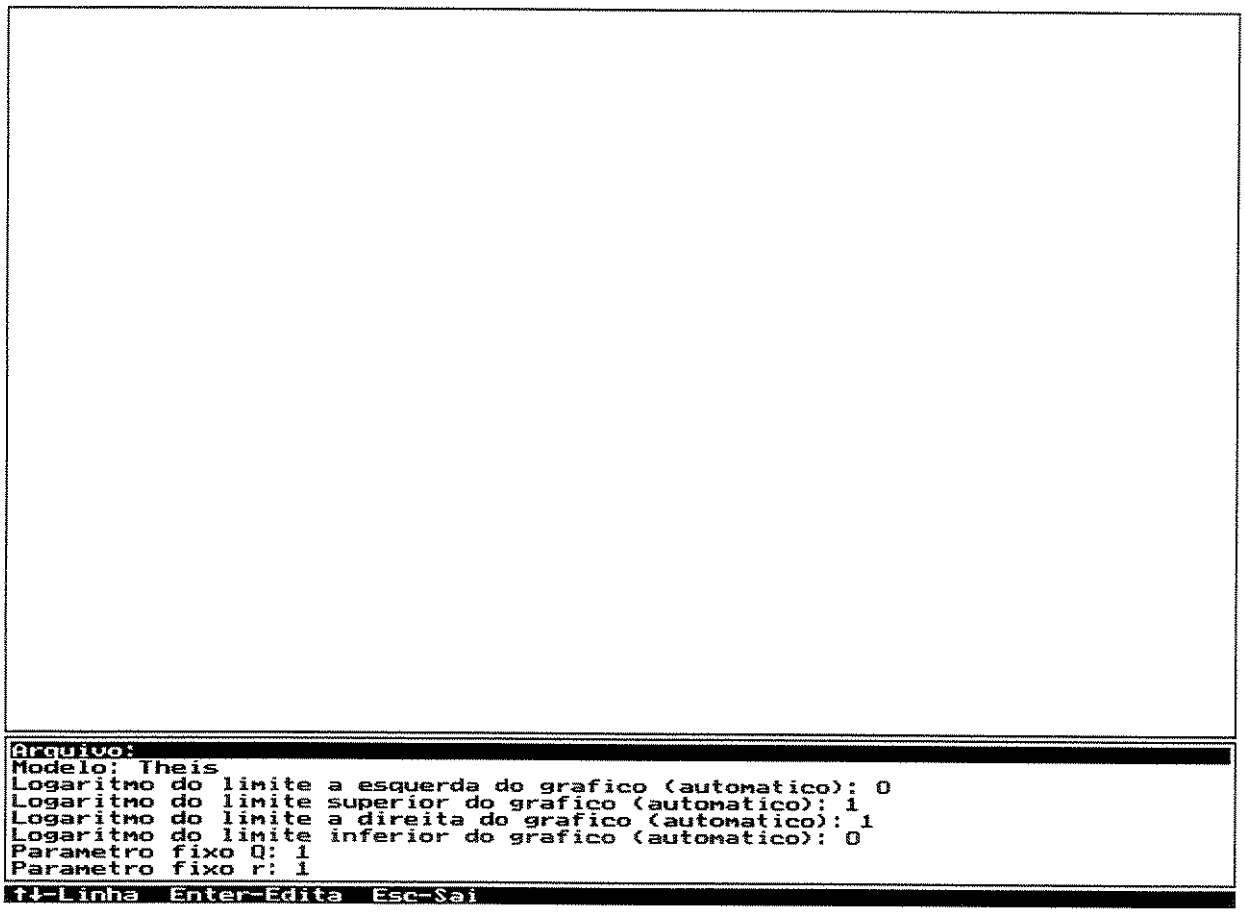

Figura 1

O comando edita, acionado pela tecla enter, permite alterar ou definir um nome ou valor para a variável selecionada. Por exemplo, ao selecionarmos a variável arquivo, que inicialmente está em branco, e pressionarmos a tecla enter, aparece uma janela de edição no canto superior esquerdo do vídeo. Podemos digitar, então, o nome de um arquivo que contenha os valores das coordenadas dos pontos de um teste de bombeamento. Este arquivo deve possuir duas colunas, a primeira para o tempo, e a segunda para o rebaixamento. O programa não aceita valores nulos, visto o gráfico dos pontos ser logarítmico. Podemos, se quisermos, digitar um dos seguintes nomes de arquivos demonstrativos, que acompanham o pacote e estão listados no Anexo B: theis.dmt, hantush.dmt, boulton.dmt e neuman.dmt. A origem destes arquivos é comentada no

capítulo seguinte, onde cada um deles é analisado segundo o modelo que lhe deu o nome. A figura 2 esclarece melhor os procedimentos descritos ao longo deste parágrafo até aqui. Ao finalizarmos a digitação devemos pressionar enter para confirmar o nome escolhido. Se este nome e o respectivo arquivo fizerem sentido, o programa plota os pontos no gráfico e desenha a curva de um modelo pré-definido, que ao iniciarmos o programa é Theis. A figura 3 mostra o que acontece. Se, porém, a curva não aparecer, é porque ela está além dos limites do gráfico. Para desistirmos da operação de edição podemos pressionar a tecla Esc. A tecla "Back Space", que corresponde a uma seta para a esquerda, apaga o último caracter do nome durante a edição. Estes últimos comandos podem ser visualizados na barra correspondente que aparece na figura 2.

O comando sair é para deixar o programa. Ao pressionarmos a tecla Esc, que aciona o comando, uma mensagem é mostrada na barra, solicitando que se confirme a operação mediante o aperto da tecla enter. Se qualquer outra tecla for pressionada a operação é interrompida.

Para escolhermos o modelo com o qual iremos trabalhar, devemos selecionar a variável modelo. Ao fazermos isto aparece na barra de comandos números associados a nomes de modelos. Basta pressionarmos o número correspondente ao modelo desejado para que este seja assumido pelo programa, o qual gerará a curva correspondente.

As quatro linhas que seguem, abaixo da variável modelo, correspondem aos valores dos logaritmos dos limites do gráfico. Editando estas variáveis podemos alterar os limites do gráfico, o que faz desaparecer a palavra automático, se estiver presente, do respectivo limite modificado. Para definirmos, novamente, de forma automática, um limite selecionado, basta pressionarmos a tecla A. A definição automática corresponde aquela cujos logaritmos dos limites são valores inteiros, englobando todos os pontos do teste, e que apresenta a melhor definição, ou ampliação.

A seguir temos os parâmetros fixos Q e r, que estão presentes em todos os modelos. O primeiro corresponde a vazão do poço de bombeamento, e o segundo a distância entre os poços de bombeamento e observação. Estes parâmetros, como todas as outras variáveis numéricas,

possuem valores pré-definidos ao ligarmos o programa. No caso destes parâmetros, e também dos parâmetros ajustáveis, a serem vistos logo mais, o valor pré-definido é 1. Mas podemos alterar o conteúdos destas variáveis por meio da edição. Com respeito as unidades, qualquer variável dimensional neste programa subentende um sistema de unidades coerente, ou seja, um sistema formado a partir das mesmas unidades básicas de comprimento e tempo.

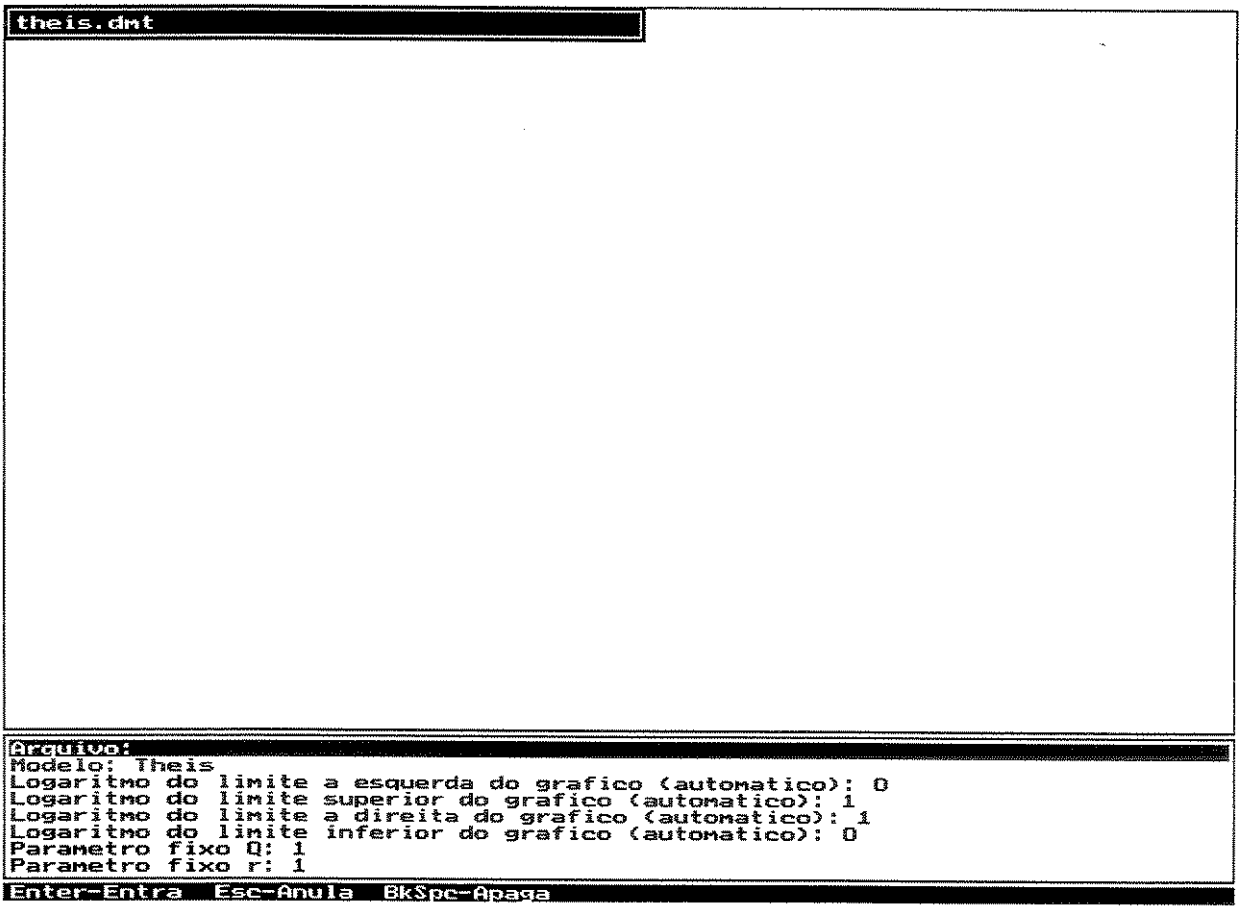

Figura 2

Abaixo dos parâmetros fixos temos os parâmetros ajustáveis. No total de seis, T, S, r/B, r/Dt, beta e sigma, alguns não são utilizados pelo modelo escolhido, estando estes indicados pela palavra inoperante, o que pode ser observado na figura 4. Mesmo assim, qualquer um deles pode ser modificado pelo comando de edição. Outra forma de alterarmos qualquer dos parâmetros ajustáveis consiste em incrementarmos ou decrementarmos o seu valor de um percentual pré-definido. Isto é feito utilizando-se as teclas + e -, a primeira para incrementar e a segunda para

decrementar. Quanto ao percentual de variação, este é inicialmente definido como sendo igual a 10 % para qualquer parâmetro selecionado, mas veremos mais adiante como é possível alterar este valor. Da mesma forma como ocorre ao modificarmos o valor de um parâmetro fixo, ao fazermos isto com um parâmetro ajustável operante, a curva do modelo se modifica de acordo.

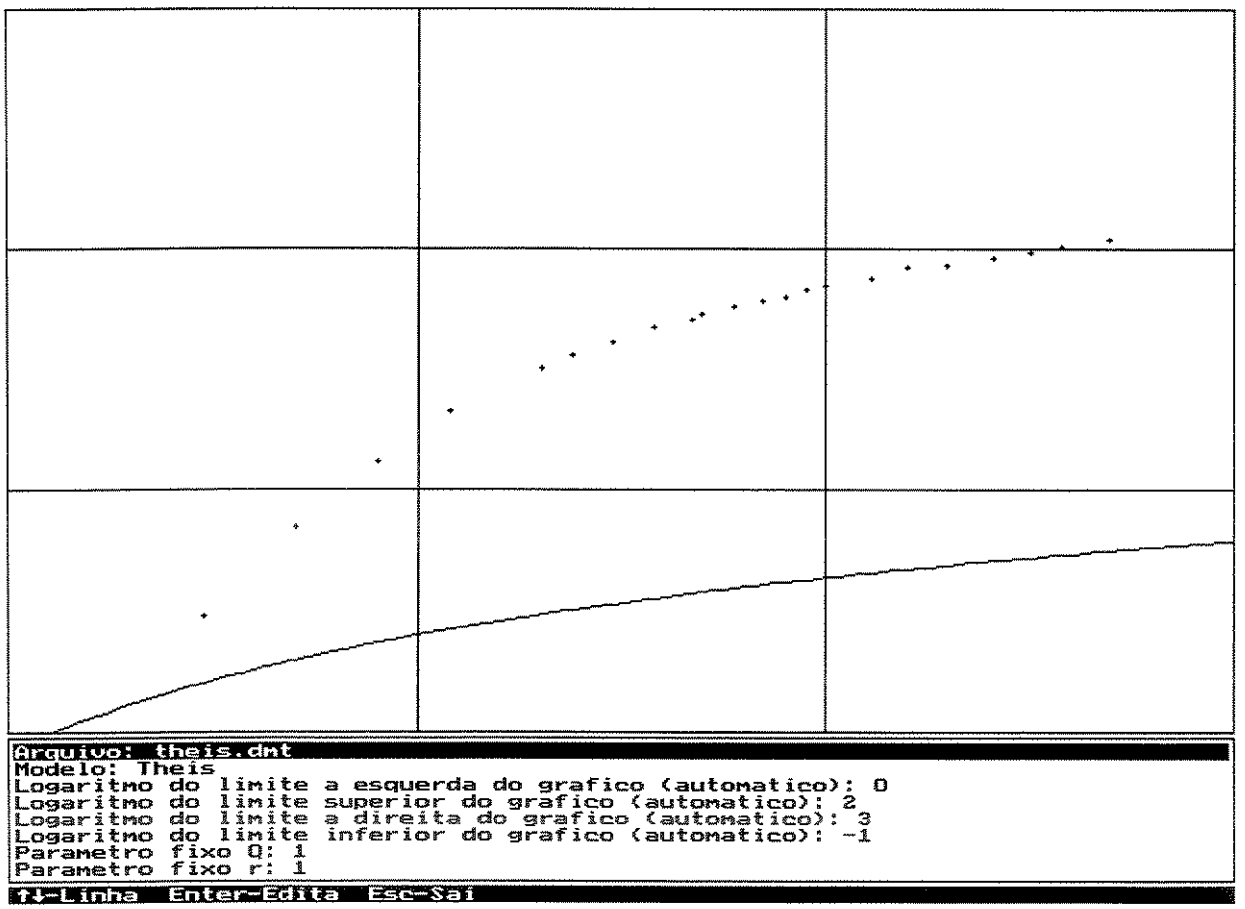

Figura 3

Há também o comando designado pela palavra fixa, que é acionado pela tecla F, e que permuta qualquer parâmetro ajustável operante entre os estados fixado e não fixado. Quando um parâmetro está fixado aparece o nome correspondente. Um parâmetro fixado mantém o seu valor inalterado durante o ajuste numérico do modelo. Este ajuste é feito a partir do comando ajusta, acionado pela tecla A. Ao apertarmos esta tecla a barra de comandos se modifica, e também aparece mais uma janela, abaixo do gráfico e acima da janela de variáveis. Na nova janela visualizamos outras variáveis, que não podem ser modificadas diretamente, por meio de edição.

As variáveis desta janela são: Iteração, Lambda, Chi-quadrado, e uma matriz denominada por [C]. A figura 5 ilustra este ponto. A variável Iteração é zerada toda vez que se abre a janela de ajuste. Ela é incrementada a cada chamada da função de ajuste mrqmin, comentada no capítulo anterior, indicando o número de iterações realizadas. Lambda (λ) é um parâmetro de chamada da função mrqmin, e também já foi comentado no capítulo anterior. Sempre que abrimos a janela de ajuste ele é feito igual a -1, um número negativo, que informa a mrqmin que se está dando início a um novo processo de ajuste. Ao final de cada iteração, deflagrada pala tecla I (iteração), ou de um conjunto delas, no intuito de se atingir certa condição de convergência, o que é acionado pela tecla A (automático), os parâmetros ajustáveis operantes e não fixados se modificam de acordo, da mesma forma a curva do modelo, assim como a variável lambda passa a assumir um valor numérico significativo, a ser adotado na próxima iteração. Aparece, então, mais alguns comandos na barra, os quais serão comentados adiante, e que podem ser visualizados na barra de comandos da figura 6. Também será visto posteriormente quais as condições que controlam as iterações do ajuste automático, e como alterá-las. Por enquanto, resta dizer, que a variável Chi-quadrado corresponde ao valor desta grandeza, mencionada Capítulo 3, ao final de cada iteração, enquanto que a matriz [C] é a inversa da matriz de curvatura modificada por λ, cujo número de elementos é igual ao quadrado do número de parâmetros ajustáveis operantes e não fixados. Como Chi-quadrado e [C] correspondem a última iteração realizada, eles podem estar fora de seu significado próprio ao abrirmos a janela de ajuste.

Falemos agora dos outros comandos que aparecem na barra de comandos durante a fase de ajuste. São eles denominados por Novo, Covariância, Refaz e Retorna, sendo acionados respectivamente pelas teclas N, C, R e Esc. O primeiro comando zera a variável Iteração e faz igual a -1 a variável Lambda, dando início assim, a um novo processo de ajuste. Equivale, portanto, a abrir a janela de ajuste. O segundo comando zera a variável Lambda e recalcula a matriz [C], que nesta condição equivale a matriz de covariância dos parâmetros ajustáveis e não fixados. A organização dos elementos desta matriz mantém correspondência com a ordem em que aparecem na janela de variáveis os parâmetros mencionados. Ao se executar este comando não é mais possível prosseguir com as iterações, as quais eram executadas a partir dos comandos das teclas I ou A, a não ser que se recorra ao comando Novo, ou ao comando Refaz, que será

visto a seguir. Deste modo, normalmente, o comando da tecla C é utilizado apenas ao final do processo de ajuste, e isto quando se deseja conhecer a matriz de covariância. O comando Refaz, como o nome sugere, refaz as condições existentes antes da execução do último comando dado, seja ele qual for. Ele poderia assim ser utilizado, quando durante o processo de ajuste surge uma mensagem de erro na barra de comando, ou quando o resultado dos comandos Iteração ou Automático parecer visualmente pior, para refazer as condições anteriores. Finalmente, o comando Retorna fecha a janela de ajuste, retornando ao antigo conjunto de comandos, que permite a edição de variáveis.

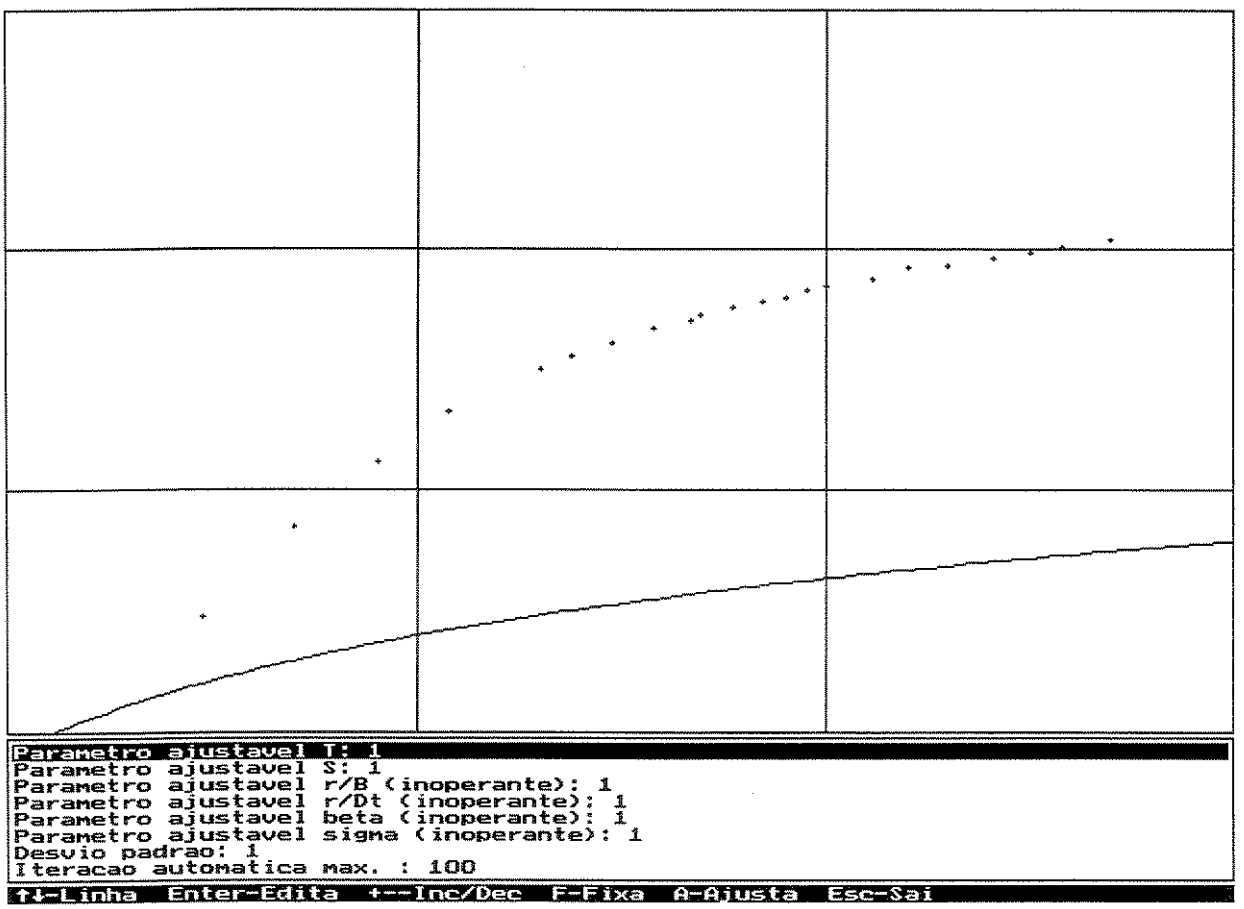

Figura 4

Prosseguindo, na janela de edição de variáveis, encontramos abaixo dos parâmetros ajustáveis uma variável denominada desvio padrão. Na realidade, ao selecionarmos esta linha, visualizamos o desvio padrão do ponto evidenciado com a cor vermelha. Sempre que o

programa lê um arquivo de dados ele atribui o valor 1 para o desvio padrão de cada um dos pontos. Podemos, então, alterar este valor para o ponto evidenciado, por meio de edição, enquantos as setas horizontais permitem que se selecione o ponto em questão. Quanto menor o desvio padrão, maior será a importância dada ao respectivo ponto no processo de ajuste. Observemos, também, que a cor de um ponto não evidenciado está relacionada ao seu desvio padrão. Se este for igual a 1 a cor é ciano claro, se for menor do que 1 a cor é branca, e se for maior do que 1 a cor é cinza escuro.

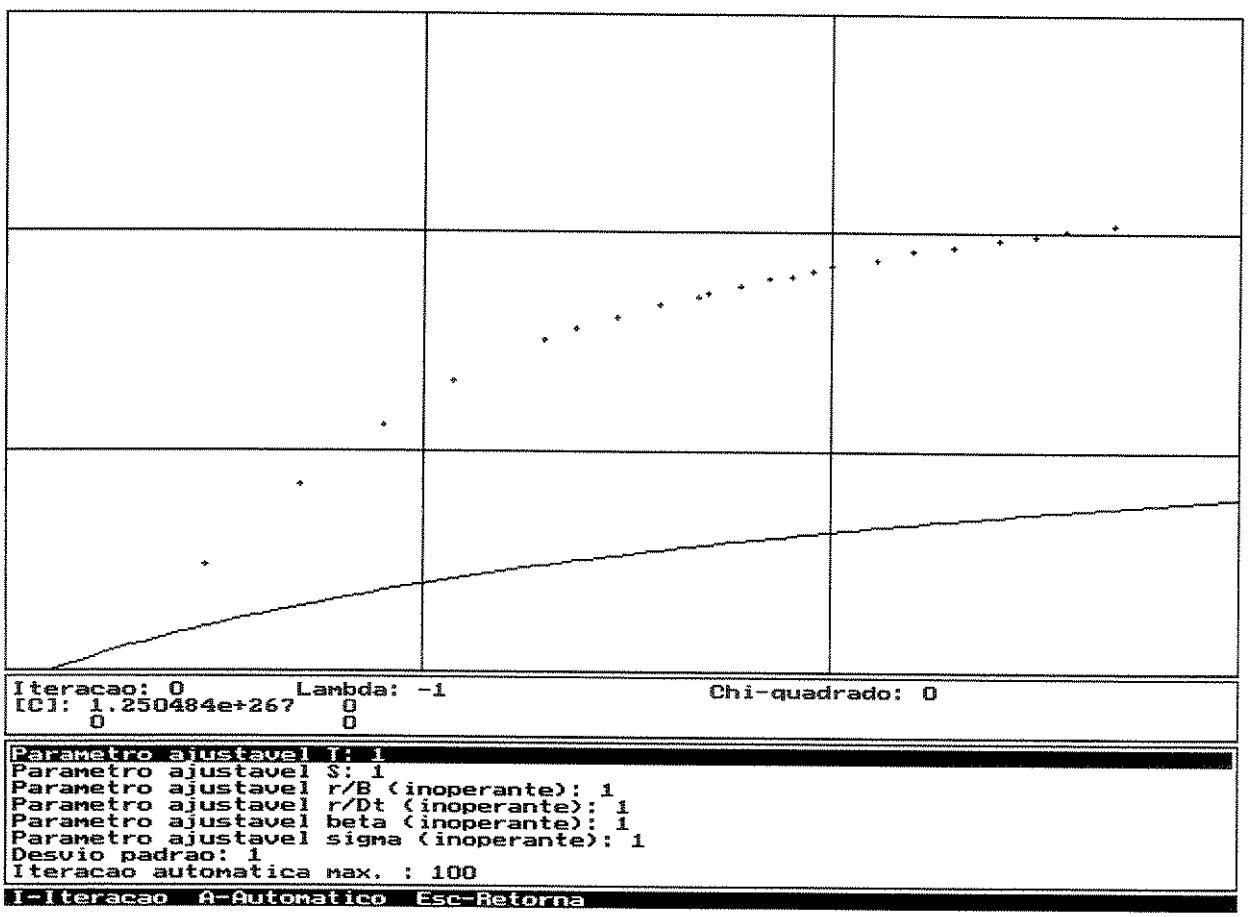

Figura 5

Continuando, as três linhas que seguem abaixo do desvio padrão dizem respeito ao ajuste automático, comentado anteriormente. Estas linhas na figura 7 aparecem no início da janela de variáveis. A primeira destas linhas limita o número máximo de iterações, e seu valor ao ligarmos o programa é igual a 100. Enquanto este limite não for ultrapassado, as iterações prosseguem até

que o chi-quadrado seja menor ou igual ao valor indicado na segunda das três linhas em análise, ou que a variação relativa do chi-quadrado, nas duas últimas iterações, seja menor ou igual ao valor indicado na última das três linhas. Cabe aqui uma observação: As condições expressas pelas duas últimas linhas só conseguem interromper a seqüência de iterações se simultaneamente, o valor de λ não houver aumentado nas duas últimas iterações. Finalmente, os valores pré-definidos das segunda e terceira linhas, ao ligarmos o programa, são, respectivamente, 1×10^{-6} e 0,001.

Finalmente, abaixo das três linhas que controlam o processo de ajuste automático, encontramos três grupos de seis linhas, onde o primeiro é responsável pela variação percentual dos incrementos e decrementos, associado às teclas + e -, de cada um dos parâmetros ajustáveis. O segundo e terceiro grupos dizem respeito, respectivamente, aos valores máximos e mínimos, que estabelecem faixas de variação para cada parâmetro ajustável, sendo isto necessário devido a característica de busca restrita, comentada no capítulo anterior, que o programa efetua durante a fase de ajuste. Os valores pré-definidos nestes grupos ao ligarmos o programa são: 10, para todas linhas do primeiro grupo, 1×10^{100} para todas as linhas do segundo, e 1×10^{-100} para todas as linhas do terceiro grupo.

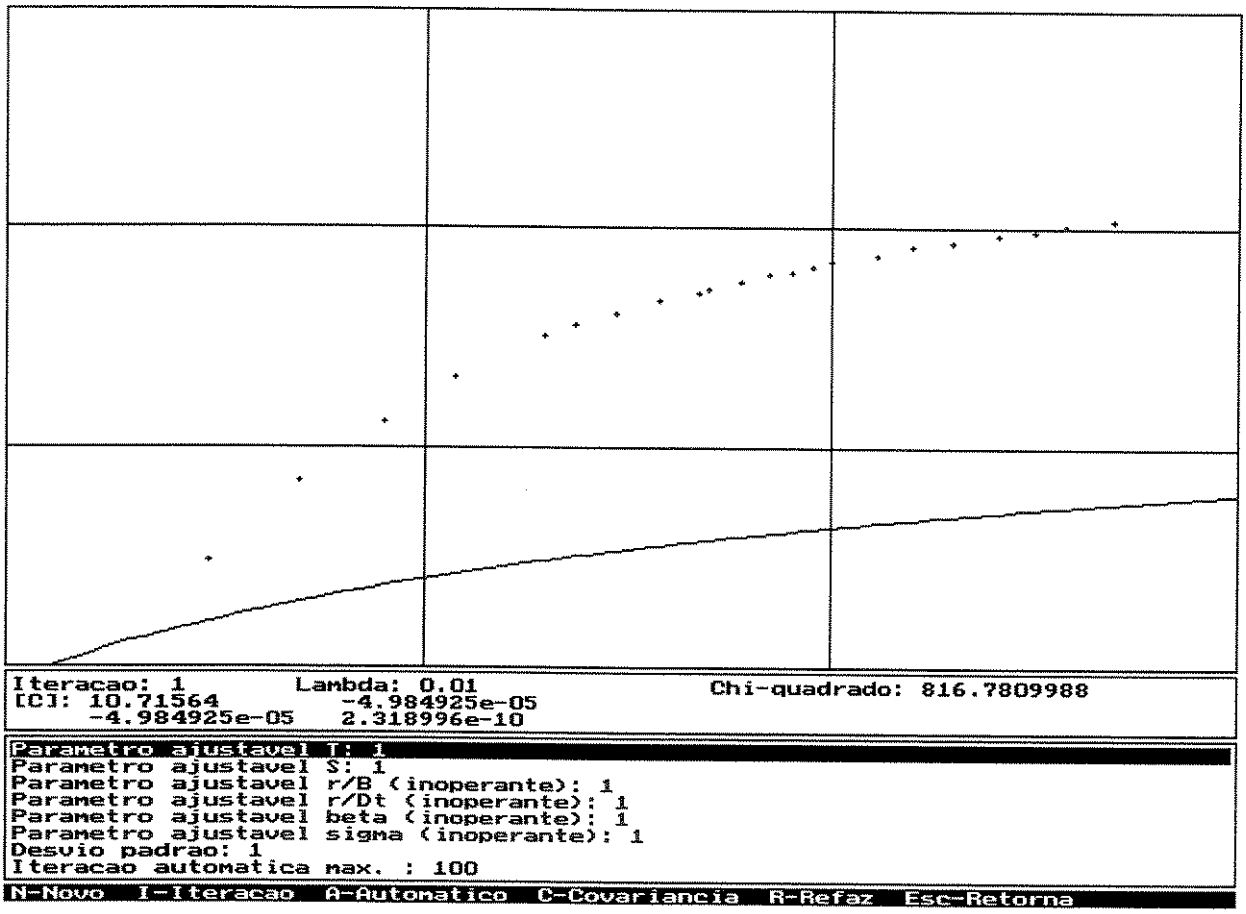

Figura 6

Resta ainda dizer, que após a edição de qualquer variável, o novo valor é verificado, e somente é aceito se for coerente, logicamente e com os outros valores estabelecidos.

3.2 Funções do arquivo tela.c

O arquivo tela tela.c, listado no Anexo A.2, possui quatro funções: grafico, arquivo, edição e main, que é o programa ou função principal. Somente main chama as três primeiras funções. A função main chama ainda as funções inivar e mrqmin do arquivo num.c, listado no Anexo A.1, e comentado no capítulo anterior; e a função gráfico chama a função funcs, também do arquivo num.c.

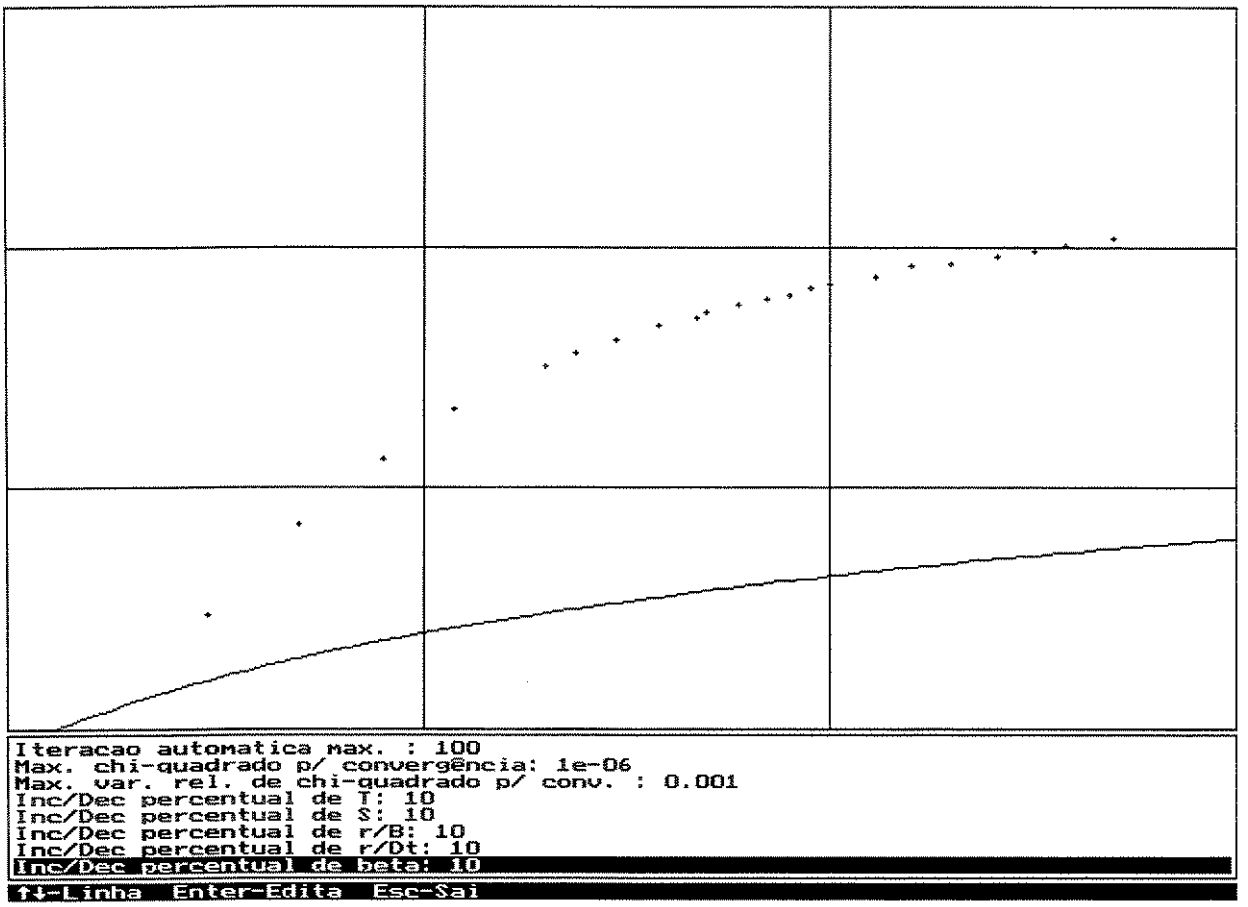

Figura 7

A função gráfico é responsável pelo desenho do gráfico, que ocupa a parte superior da tela, e possui os seguintes parâmetros: xmax, ymax, loglim, nomarq, xxpto, yypto, npto, a, sig e ptsig. Os limites do gráfico à esquerda e acima são sempre os limites da tela. Os limites à direita e abaixo são dados, respectivamente, pelas coordenadas de tela xmax e ymax. loglim é um vetor com os logaritmos dos limites do gráfico na dimensão do problema em análise, ou seja, nas unidades dos pontos do teste de bombeamento. nomarq é o nome do arquivo com os pontos do teste de bombeamento. xxpto, yypto são vetores que contém, respectivamente, as abcissas e as ordenadas dos pontos do teste. npto é a quantidade destes pontos. a é o vetor dos parâmetros, fixos e ajustáveis, dos modelos. Porém, o elemento de índice zero, simplesmente é para informar, quando seu valor é nulo, que houve alguma mudança em algum dos outros elementos deste vetor. sig é o vetor que contém o desvio padrão dos pontos, é utilizado pela função gráfico para estabelecer as cores dos pontos. ptsig é o número do ponto, de cor vermelho claro, cujo

desvio padrão é mostrado na janela de variáveis. Quando for igual a zero significa que o respectivo ponto não deve ser evidenciado.

A função gráfico é otimizada em termo de velocidade de execução, de forma que diversos dos parâmetros de entrada, como é o caso do parâmetro nomarq, são comparados com valores correspondentes da chamada anterior, os quais são armazenados em variáveis estáticas, para que somente sejam refeitos os cálculos necessários. Isto é útil quando se opera com um modelo mais lento, como é o caso do modelo de Neuman.

Faremos agora um comentário sucinto das variáveis internas da função gráfico, basicamente só descrevendo os significados de delas. Comecemos pelas do tipo inteiro estáticas, que são xmaxpas, ymaxpas, xpto, ypto, modpas, nmod, xmod e ymod. xmaxpas e ymaxpas são as coordenadas de tela dos limites à direita e à esquerda do gráfico, quando da última chamada da função. xpto e ypto são vetores com as coordenadas de tela dos pontos do teste. modpas é o número do modelo vigente na chamada anterior de grafico. Este número é confrontado com a variável global Gmod, que especifica o número do modelo atual. Gmod é a única variável global utilizada pela função gráfico. Nenhuma outra função do arquivo tela.c, a exceção de main, faz uso de variáveis globais. Prosseguindo, nmod+1 é o número de pontos do curva do modelo, que serão interligados por segmentos de reta. xmod e ymod são as coordenadas de tela destes pontos. Passemos, agora, às variáveis estáticas de ponto flutuante com dupla precisão, que são: loglimpas, cclx, ccly, xgrdmin, ygrdmin e yymod. loglimpas é um vetor que contém os logaritmos dos limites do gráfico nas unidades do teste de bombeamento, e que foram empregados na última chamada da função gráfico. cclx e ccly são os comprimentos dos ciclos logarítmicos horizontal e vertical na unidade de tela. xgrdmin e ygrdmin são as coordenadas de tela das linhas mais à esquerda e mais acima da grade logarítmica do gráfico. yymod é um vetor contendo as ordenadas dos pontos da curva do modelo, na unidade do teste de bombeamento, a serem interligados. Estas ordenadas são geradas chamando-se a função funcs, do arquivo num.c. A variável estática restante é nomarqpas, que é o nome do arquivo passado, ou seja, aquele da última chamada de grafico. x, Dx e y são variáveis auxiliares de dupla precisão; i também é

auxiliar, mas inteira; enquanto que h, v e arq são variáveis inteiras indicativas de condições lógicas, como alteração dos limites horizontais, dos limites verticais, ou do arquivo de dados.

Na função grafico todos os parâmetros eram de entrada, e nenhum era modificado pela função; como também ela era do tipo "void", ou seja, apesar do nome função, ela não retornava nenhum valor, constituindo-se mais propriamente em um procedimento. O mesmo ocorre com a função main, que aliás nem parâmetro de entrada ou saída possui. Mas analisemos agora a função arquivo. Ela é responsável pela abertura e leitura dos pontos do arquivo de dados. Seus parâmetros são: nome, x, y, sig, lim e n. Somente o primeiro é parâmetro de entrada, sendo os restantes de saída. Ela também retorna um valor: zero se tudo correu bem; um se o parâmetro nome, que é o nome do arquivo de dados, especifica o mesmo arquivo já lido anteriormente, o que dispensa a repetição desta tarefa; dois se não consegue abrir o arquivo; e três quando não consegue ler nenhum ponto no arquivo. Quanto aos parâmetros de saída, x, y, sig e lim são vetores, que retornam, respectivamente, as abcissas, as ordenadas, e os desvios padrões (todos feitos iguais a 1) dos pontos, e os limites automáticos do gráfico, comentados na seção anterior. O parâmetro de saída n é o número de pontos lidos. O máximo número de pontos que o programa consegue ler de um arquivo é 199. Quanto as variáveis internas da função arquivo, estas são poucas, ficando claro seus significados no código.

A função edicao é responsável pela edição de variáveis. É ela que gera a barra de comandos específica da edição, a janela de edição, e faz todo o controle dos caracteres digitados e apagados, como também do cursor. Ela retorna o valor zero quando a edição é finalizada pela tecla enter, que confirma a intenção de se adotar o valor ou nome expresso na janela. Quando a edição termina com a tecla Esc, que cancela a operação, a função edicao retorna o valor 1. Seus parâmetros são somente três: alfanum, num e tipo. tipo especifica se a edição vai trabalhar com um nome ou um número. Para o primeiro caso, tipo deve ser igual a zero, para o segundo, ele deve ser igual a 1. Quando estamos editando um nome, alfanum deve conter o nome a ser editado, para que o mesmo apareça na janela no início da edição. Ao final desta, alfanum conterá o novo nome, fruto desta operação. Quando estamos editando um número algo similar é feito

com o parâmetro num, o qual aponta para uma variável numérica. Mesmo assim, neste caso, o parâmetro alfanum ainda é utilizado, como "buffer", o que promove alterações em seu conteúdo.

Quanto às variáveis internas da função edicao, que são n, cursor, estado, x, y, nmax, ymax, tecla e t, faremos um comentário sucinto, como foi feito com a função gráfico. n é o número de caracteres da expressão que está sendo editada. cursor igual a um significa que o cursor está aceso, ou seja, visível na tela. cursor e estado são variáveis indicativas de estado. cursor igual a zero significa que o cursor está apagado. estado igual a um significa que a expressão ainda não sofreu nenhuma modificação, estando com o fundo escuro, e que se for pressionada qualquer tecla que possa ser impressa, toda a expressão será apagada, sendo substituída pelo respectivo caracter. Se, porém, enquanto estado for igual a um, for pressionada uma tecla que não possa ser impressa, a expressão inicial não é apagada, para que possa ser modificada a partir de como ela está. estado igual a zero significa, portanto, que já foi pressionada alguma tecla após a chamada de edicao. x e y definem a posição onde será impresso o próximo caracter. nmax é o número máximo de caracteres aceitável na expressão, que está sendo editada. ymax é o valor do limite inferior da tela. E t contém informação sobre o tempo, fornecida pelo relógio da máquina, informação esta que controla a pulsação do cursor.

Finalmente, a função main é responsável pelo controle geral do programa. Seu código é extenso, e sua principal função é realizar a janela de variáveis, controlando cada uma das ações disponíveis que aparecem na barra de comandos. É esta função que verifica a coerência dos resultados da edição de cada variável, aceitando ou rejeitando tais resultados, que no último caso significa permanecer com os conteúdos já existentes antes da edição.

Comentemos agora o significado das variáveis da função main. Gamin, Gamax e Gmod são as três únicas variáveis globais empregadas por main. As duas primeiras são vetores que contém os limites mínimos e máximos dos parâmetros dos modelos, limites estes que são utilizados no ajuste, devido a característica de busca restrita de parâmetros. Passemos, então, as demais variáveis, que são internas à função main. x, y e sig são, respectivamente, vetores que contém as abcissas, ordenadas e desvios padrões dos pontos do teste de bombeamento. a, a2 e

percent são vetores. Os dois primeiros contém todos os parâmetros dos modelos, porém em circunstâncias diferentes, e o último contém os percentuais de variação dos incrementos e decrementos, associados às teclas + e -. alpha, alpha2, covar e covar2 são matrizes, as duas primeiras relativas à matriz de curvatura, e as duas últimas relativas à inversa da matriz de curvatura modificada por λ (ver Capítulo 3). lista é um vetor que agrupa os parâmetros a serem ajustados, e aqueles que devem permanecer inalterados, referenciando-os por meio de números indicativos. Ele pode ser compreendido melhor estudando-se a função mrqmin em PRESS et al. (1989). fix e oper são vetores indicativos dos parâmetros fixados e dos parâmetros operantes. loglimgrg e loglimautgrf são vetores que contém os logaritmos dos limites do gráfico. O segundo contém os limites automáticos, e o primeiros contém os limites atuantes, que podem ser iguais aos primeiros. aux é uma variável auxiliar. alamda, alamda2 e alamda3 são todas variáveis referentes ao parâmetro λ, da mesma forma como chisq, chisq2 e chisq3 são todas referentes ao valor de chi-quadrado. chisqabs e chisqrel contêm limites superiores para chi-quadrado, onde o primeiro é absoluto e o segundo relativo, os quais são empregados no ajuste automático. graphdriver e graphmode são parâmetros de funções da linguagem responsáveis pela inicialização do modo gráfico. xmax e ymax são os limites da tela à direita e abaixo. ygrf e ygrfmax são limites inferiores do gráfico, o qual é gerado pela função de mesmo nome. c, cc e ccc contém caracteres lidos do teclado. i e j são variáveis normalmente utilizadas como contadores. mfit é o número de parâmetros ajustáveis. linha1max e linha2max são, respectivamente, o número total de linhas menos um da janela de variáveis, e número de linhas visíveis menos um da mesma janela. linha1 e linha2 servem para referenciar uma linha selecionada. mma é o número total de parâmetros dos modelos. nca é uma variável sem significado, criada apenas para se manter o padrão dos parâmetros de chamada da função mrqmin, devido ao intuito de se modificar o mínimo possível a função original, que foi obtida de um pacote relativo a publicação de PRESS et al. (1989). iter e iter2 referem-se a números de iterações, enquanto que itermax refere-se ao número máximo das iterações permitidas no ajuste automático. erro é uma variável indicativa da existência de erro. ndata é o número de pontos do teste de bombeamento. ptsig é o número do ponto do teste cujo desvio padrão é apresentado na janela de variáveis. automat é um vetor que diz quais limites do gráfico são automáticos. buffer é um "buffer" normalmente utilizado na chamada da função edicao. nomarq é o nome do arquivo

dos pontos do teste de bombeamento. nommod contém os nomes dos modelos. sethor é uma "string" contendo as setas horizontais. tipolim contém os nomes dos tipos de limites do gráfico. setver é semelhante à sethor, porém contendo as setas verticais. nomparfix contém os nomes dos parâmetros fixos. E nomparaju contém os nomes dos parâmetros ajustáveis.

4 Aplicação a Casos da Literatura e a Dados Sintéticos, e Avaliação dos Resultados

Com respeito aos modelos de Theis, Hantush e Boulton, o programa foi testado aplicando-o aos dados relativos a estes modelos que aparecem na seção "Case Histories of Aquifer Test Analysis" do Capítulo 4 de WALTON (1970). Esta seção mostra os dados ajustados a curvas tipo, e a determinação correspondente dos parâmetros dos aqüíferos. No caso do modelo de Neuman a verificação do programa foi feita empregando-se os dados e o resultado do ajuste apresentados por KRUSEMAN & RIDDER (1990, exemplo 5.1).

Como de um modo geral a concordância não foi tão boa entre os resultados dos ajustes realizados pelo ITBA, e resultados fornecidos pela literatura, a última seção deste capítulo apresenta uma validação em termos de ajuste dos modelos a dados sintéticos, realizados pelo programa.

4.1 Aplicação para o modelo de Theis

A aplicação do modelo de Theis foi selecionada do caso intitulado *Grydley, Illinois*, da Seção 4.11 de WALTON (1970). Este caso trata de um teste de aqüífero realizado em 2 de julho de 1953. Foram utilizados um grupo de poços da vila de Gridley, McLean County, Illinois. Os perfis dos poços correspondem aproximadamente a uma zona arenosa e cascalhenta, com cerca de 20 ft de espessura, onde os poços eram abertos, sobreposta por uma espessa (cerca de 250 ft) zona argilosa. O bombeamento em um dos poços foi realizado a uma vazão constante de 220 galões por minuto (gpm), por cerca de 8 horas. O rebaixamento do nível da água foi registrado

ao longo do tempo em um poço de observação distante 824 ft do primeiro. Esta distância corresponde ao parâmetro fixo r. O Anexo B.1 lista o arquivo que serve de entrada para o ITBA, que contém na primeira coluna os tempos de observação em minutos, a partir do início do bombeamento, e na segunda os rebaixamentos respectivos no poço de observação, cuja unidade é o pé. Como o programa trabalha com um sistema de unidades derivadas de uma unidade básica de tempo, e outra de comprimento, é conveniente que se converta a vazão de bombeamento para o sistema de unidades formado a partir do pé e do minuto, resultando em 29,409732 ft^3/min, correspondendo ao parâmetro fixo Q. Em resumo,

$$Q = 220 \text{ gpm} = 29{,}409732 \text{ ft}^3/\text{min}$$

$$r = 824 \text{ ft}$$

Os resultados da aplicação do modelo aos dados, obtidos pelo programa, que procura a minimização do Chi-quadrado, conforme está descrito no Capítulo 3, são os que seguem abaixo. A figura 8 apresenta o ajuste final obtido pelo programa, onde aparecem o Chi-quadrado e a matriz de correlação.

$$T = 0{,}919847 \text{ ft}^2/\text{min}$$

$$S = 2{,}09485 \times 10^{-5}$$

onde T é a transmissividade média do aqüífero, e S o seu coeficiente de armazenabilidade médio, ambos parâmetros ajustáveis do modelo.

Os resultados do ajuste apresentado na literatura para o caso mencionado foram:

$$T = 10.100 \text{ gpd/ft} = 0{,}9376298 \text{ ft}^2/\text{min}$$

$$S = 0{,}00002$$

A figura 9 apresenta a qualidade do ajuste para estes valores, e foi obtida fixando-se os parâmetros ajustáveis com os valores acima, procurando-se a seguir realizar uma iteração de ajuste, para com isto efetuar a avaliação de qui-quadrado (χ^2). Este é um parâmetro que mede a qualidade do ajuste, conforme foi mencionado no Capítulo 3.

Os erros percentuais tomados em relação aos resultados apresentados por Walton são: -1,897 % para a transmissividade e 4,742 % para o coeficiente de armazenabilidade. Os resultados do programa podem ser considerados satisfatórios uma vez que o ajuste apresentado na literatura parece ter sido realizado visualmente, o valor de S relativo ao mesmo ajuste apresenta um único algarismo significativo, e o qui-quadrado do ajuste do ITBA é menor.

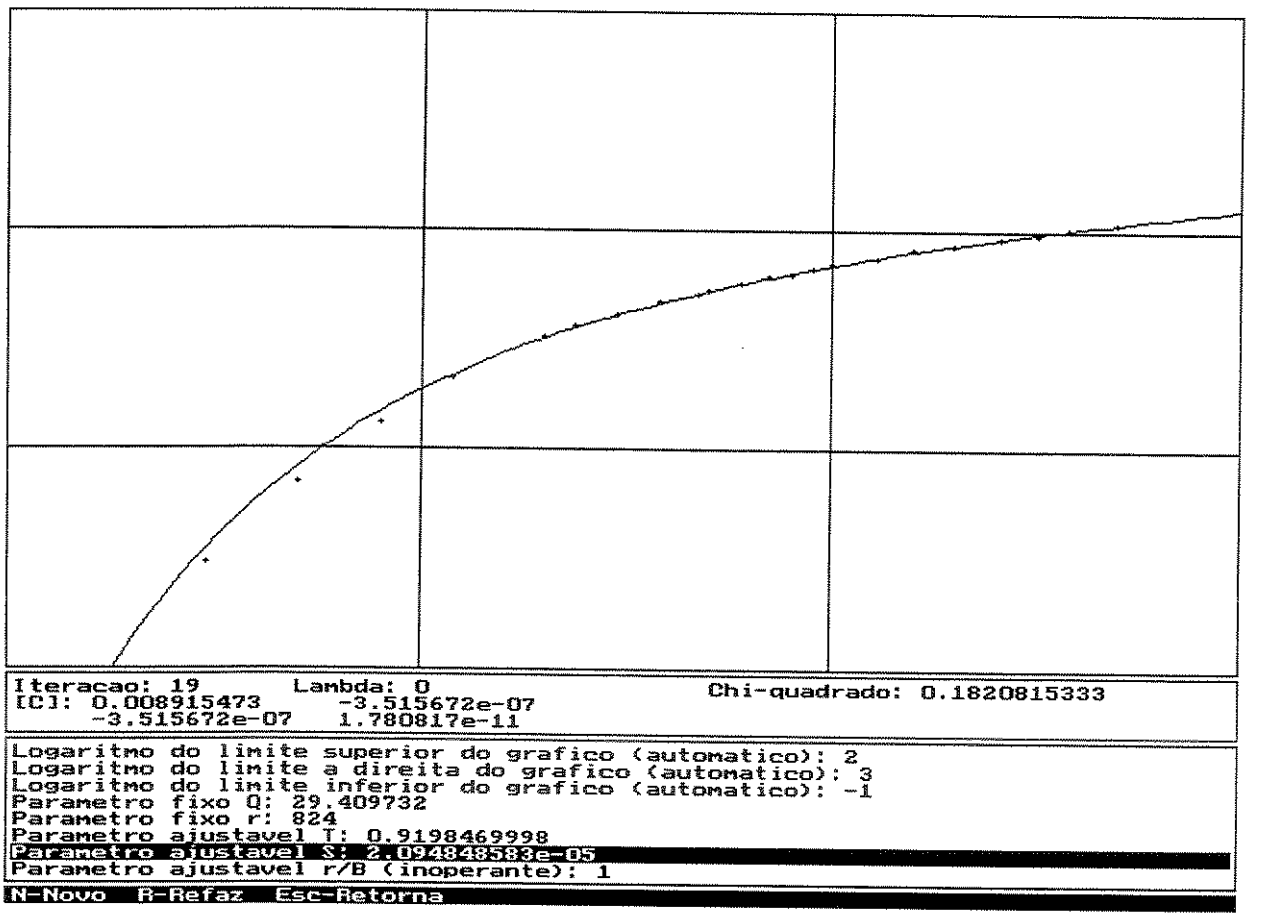

Figura 8

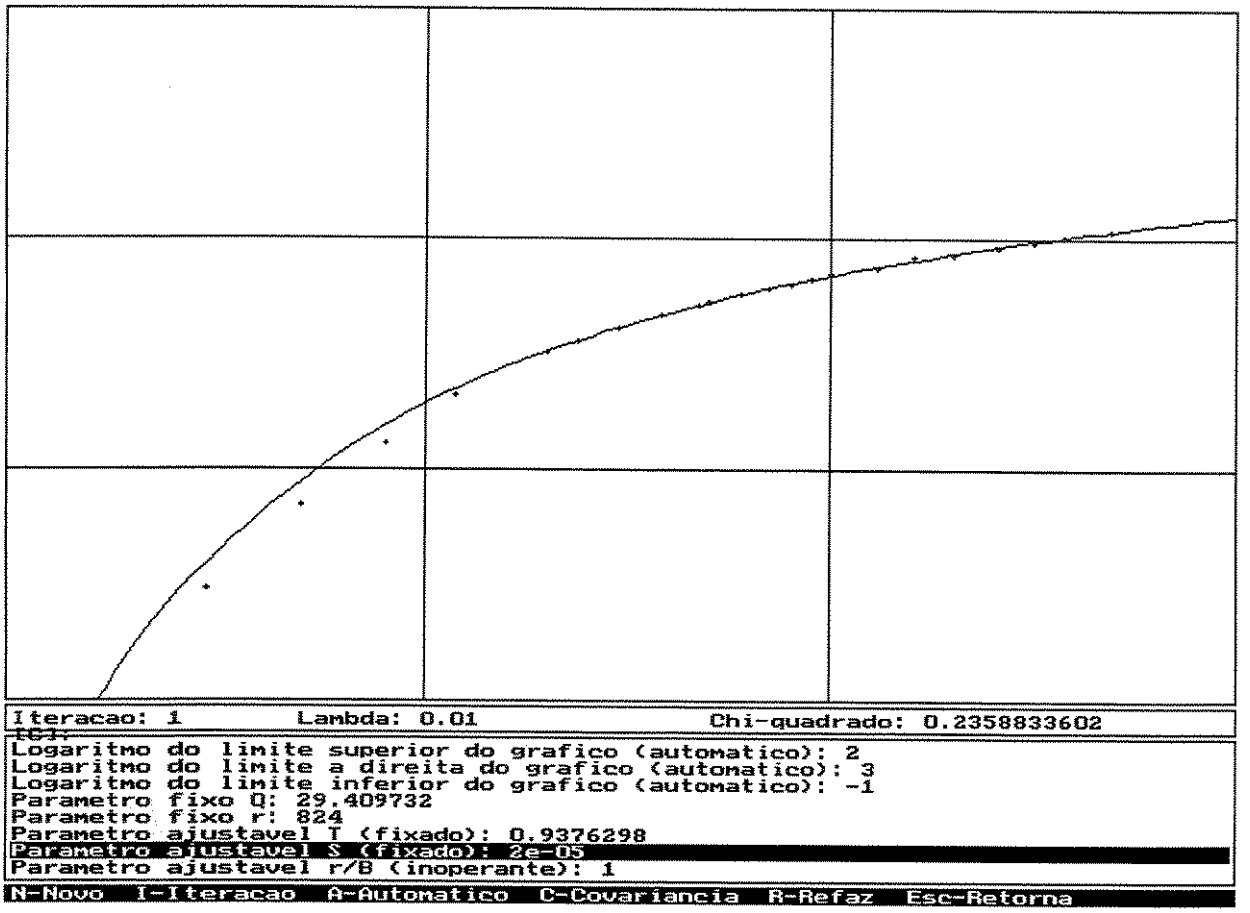

Figura 9

4.2 - Aplicação para o modelo de Hantush

O segundo modelo do programa, Hantush, foi aplicado a dados do quarto caso da Seção 4.11 de WALTON (1970), caso intitulado *Dieterich, Illinois*. O teste de bombeamento deste caso foi conduzido em um grupo de poços localizados em Effingham County, a cerca de 1 milha a sudoeste dos limites da vila de Dieterich, Illinois. O teste foi realizado nos dias 2 e 3 de julho de 1951. Os perfis dos poços de bombeamento e observação selecionados se compõem basicamente de uma zona arenosa, ou arenosa e cascalhenta, de cerca de 7 a 9 pés de espessura, onde os poços estão abertos para produção, sobreposta por uma zona argilosa de cerca de 8 a 12 pés de espessura, ainda sobreposta por outra zona areno-argilosa de cerca de 12 pés, onde ocorre o nível freático, tendo finalmente acima, uma fina camada de solo. Os dados de rebaixamento

versus tempo, referentes ao poço de observação analisado, compõem o arquivo hantush.dmt, que está listado no Anexo B.2. A organização deste arquivo e as unidades são similares às do item anterior. A vazão Q (constante) a que foi submetido o poço de bombeamento , e a distância r deste ao poço de observação, estão explicitadas a seguir:

$$Q = 25 \text{ gpm} = 3{,}342015 \text{ ft}^3/\text{min}$$

$$r = 96 \text{ ft}$$

Os resultados obtidos da aplicação do programa aos dados, e que podem ser visualizados na figura 10, são:

$$T = 0{,}172305 \text{ ft}^2/\text{min}$$

$$S = 0{,}000165768$$

$$r/B = 0{,}147126$$

T e S possuem o mesmo significado da seção anterior, r/B é um parâmetro adimensional que define uma curva tipo, que é utilizada quando se trabalha com um método de análise por ajuste visual, como é o caso em WALTON (1970). B é dado pela expressão $\sqrt{T/(P'/m')}$, que possui dimensão de comprimento, onde P' e m' são respectivamente a permeabilidade e a espessura do aquitard, ou seja, a camada semi-permeável que separa o aqüífero inferior do superior.

Os resultados apresentados por Walton para este caso são:

$$T = 1510 \text{ gpd/ft} = 0{,}1401789 \text{ ft}^2/\text{min}$$

$$S = 0{,}0002$$

$$r/B = 0{,}22$$

Observa-se na figura 11 a qualidade do ajuste para os valores acima.

Os erros percentuais dos resultados do programa, com respeito aos resultados apresentados por Walton, são: 22,92 % para a transmissividade do aqüífero, -17,12 % para o

coeficiente de armazenabilidade, e -33,12 % para r/B. A concordância entre os resultados neste caso não é tão boa.

O texto de Walton sugere que o ajuste entre os dados e uma das curvas tipo foi realizado visualmente. Verificou-se também que não é muito clara a seleção de uma curva tipo que melhor se adeqüe aos dados a partir de um ajuste visual. No intuito de avaliar a qualidade dos resultados do programa, foi fixado provisoriamente o parâmetro r/B, tornando o ajuste numérico no modelo de Hantush independente do parâmetro da curva tipo. Este parâmetro foi igualado, então, ao resultado apresentado por Walton, ou seja, r/B = 0,22. Para que fique mais claro, o que se está querendo verificar é se as divergêngias encontradas anteriormente se devem basicamente a uma escolha diferente do parâmetro que identifica uma curva tipo, uma vez que esta escolha é difícil de ser feita visualmente. Nestas condições o ajuste numérico conduziu aos seguintes resultados, que estão associados à figura 12:

$$T = 0{,}141914 \text{ ft}^2/\text{min}$$

$$S = 0{,}000202343$$

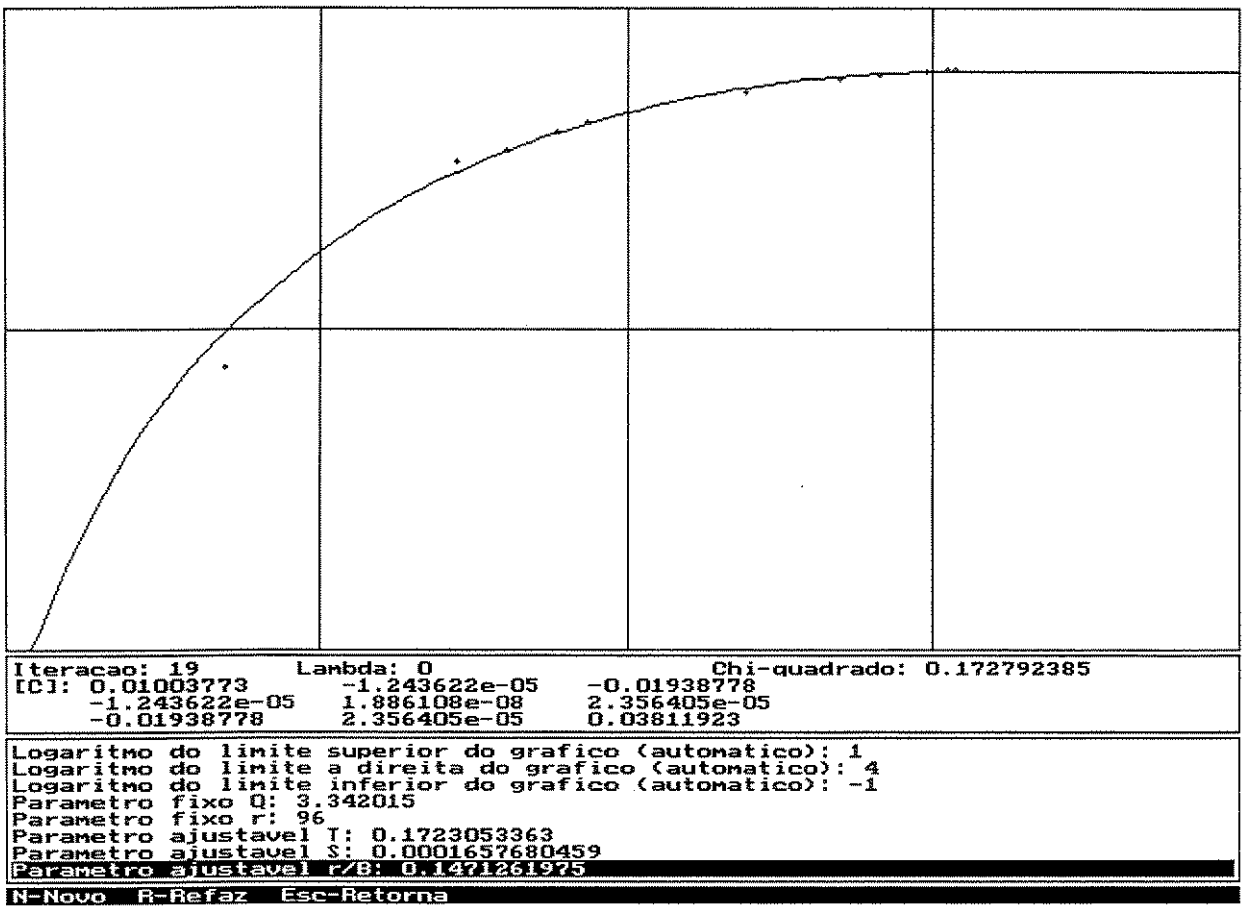

Figura 10

Os erros percentuais associados a estes resultados são: 1,238 % para a transmissividade e 1,172 % para o coeficiente de armazenabilidade do aqüífero. A concordância entre os resultados agora é muito mais forte. Esta análise sugere que as discrepâncias observadas inicialmente se deviam a um ajuste deficiente, entre os dados e as curvas tipo do modelo, no caso apresentado por Walton, em especial na seleção da curva tipo. Assim sendo, os resultados iniciais, fornecidos pelo programa ITBA, parecem ser os mais confiáveis, conforme atestam os valores de qui-quadrado, observados nas figuras.

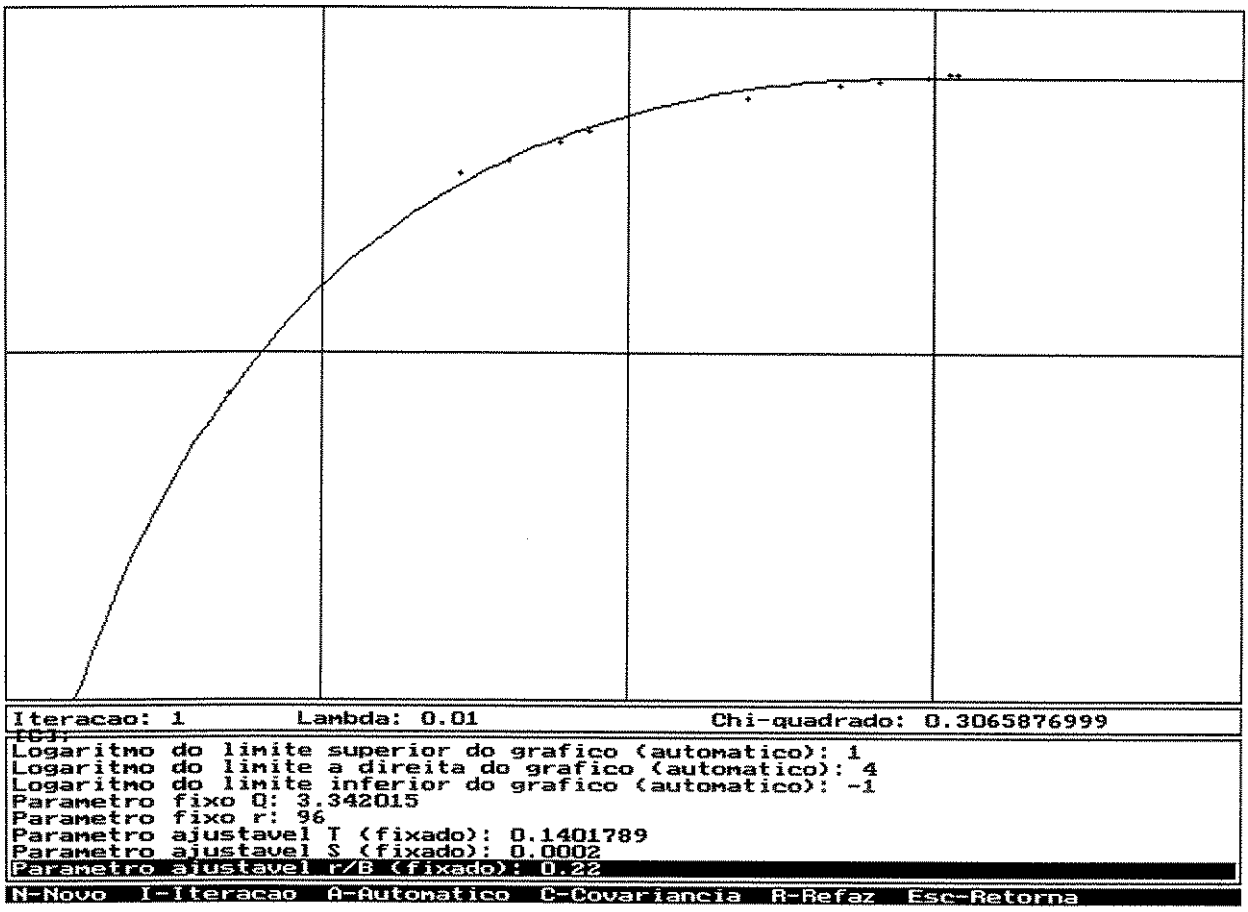
Iteracao: 1 Lambda: 0.01 Chi-quadrado: 0.3065876999
[CJ]:
Logaritmo do limite superior do grafico (automatico): 1
Logaritmo do limite a direita do grafico (automatico): 4
Logaritmo do limite inferior do grafico (automatico): -1
Parametro fixo Q: 3.342015
Parametro fixo r: 96
Parametro ajustavel T (fixado): 0.1401789
Parametro ajustavel S (fixado): 0.0002
Parametro ajustavel r/B (fixado): 0.22
N-Novo I-Iteracao A-Automatico C-Covariancia R-Refaz Esc-Retorna

Figura 11

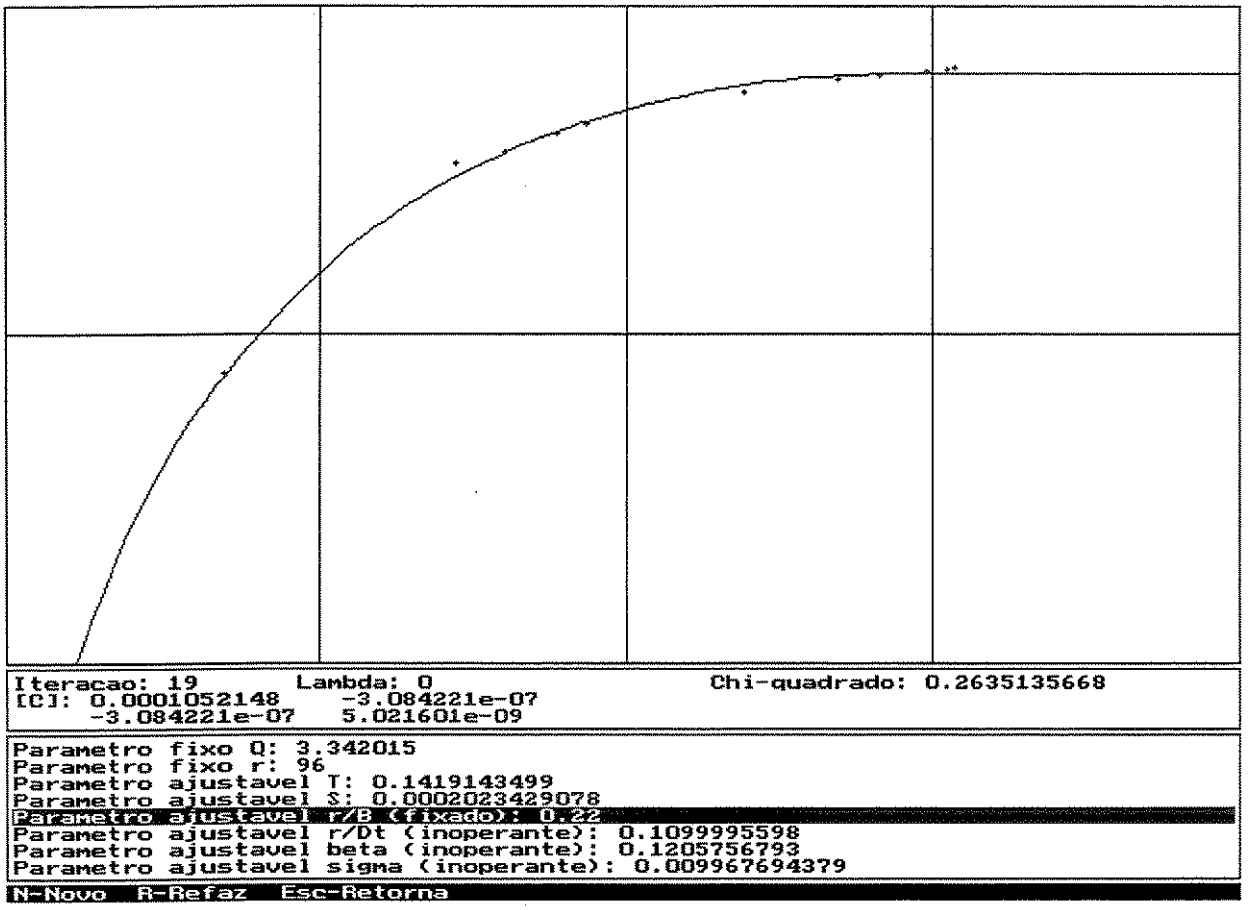

Figura 12

4.3 - Aplicação para o modelo de Boulton

O terceiro modelo do programa, modelo de Boulton, foi verificado utilizando-se o sexto caso, *Lawrenceville, Illinois*, apresentado na 4.11 de WALTON (1970). Este teste foi realizado em 17 e 18 de maio de 1950, em um grupo de poços localizados a cerca de 4 milhas a sudeste dos limites da cidade de Lawrenceville, Illinois. Observa-se nos perfis apresentados tratar-se de um aqüífero livre, com pelo menos cerca de 100 ft de profundidade, formado por diversas zonas sobrepostas, variáveis quanto a composição granulométrica. Os dados de rebaixamento versus tempo do teste foram obtidos da leitura de um gráfico apresentado na figura 4.28 da referência mencionada, e não mais a partir de uma tabela, que daria valores mais precisos. Os dados lidos estão no arquivo boulton.dmt, listado no Anexo B.3, e que possui a mesma organização e

unidades dos arquivos correspondentes das seções anteriores. A vazão Q no poço de bombeamento, mantida constante, e a distância r deste ao poço de observação escolhido para análise, estão a seguir:

$$Q = 1.000 \text{ gpm} = 133,6806 \text{ ft}^3/\text{min}$$

$$r = 200 \text{ ft}$$

Os resultados fornecidos pelo ajuste numérico do programa, que podem ser visualizados na figura 13, são:

$$T = 30,2993 \text{ ft}^2/\text{min}$$

$$S = 0,00251202$$

$$r/Dt = 0,523694$$

$$\sigma = 0,196001$$

T é a transmissividade, S é o coeficiente de armazenabilidade, r/Dt é um parâmetro adimensional que define um par curvas tipo do modelo, utilizadas quando se opera com ajuste visual, como Walton dá a entender que foi realizado na análise que ele apresenta. Dt é definido por

$$Dt = \sqrt{\frac{T}{D_i S_y}},$$

onde D_i é um parâmetro empírico conhecido como o inverso do índice de retardo, e S_y é a produção específica. σ é igual a S/S_y.

Quanto aos os resultados fornecidos por Walton para este caso, cuja qualidade do ajuste respectivo pode ser apreciada na figura 14, são os seguintes:

$$T = 272.500 \text{ gpd/ft (média entre 279.000 e 266.000)} = 25,297197 \text{ ft}^2/\text{min}$$

$$S = 2,33 \times 10^{-3}$$

$$r/Dt = 0,7$$

$$\sigma = 0,0735015$$

Os erros percentuais associados aos resultados do programa, com relação aos resultados de Walton, são: 19,77 % para T, 7,811 % para S, -25,19 % para r/Dt, e 166,7 % para σ. Como se observa os resultados não concordam muito.

Do mesmo modo como foi feito com o modelo de Hantush, tentou-se uma fixação provisória do parâmetro r/Dt no programa, de tal forma que o modelo, que possui os parâmetros ajustáveis T, S, r/Dt e σ, passasse a ser função só dos dois primeiros e do último, ficando r/Dt igual ao resultado dado por Walton, ou seja, 0,7. Isto produziu os resultados abaixo, conforme pode ser visto na figura 15.

$$T = 27{,}8130 \text{ ft}^2/\text{min}$$

$$S = 0{,}00208933$$

$$\sigma = 0{,}124907$$

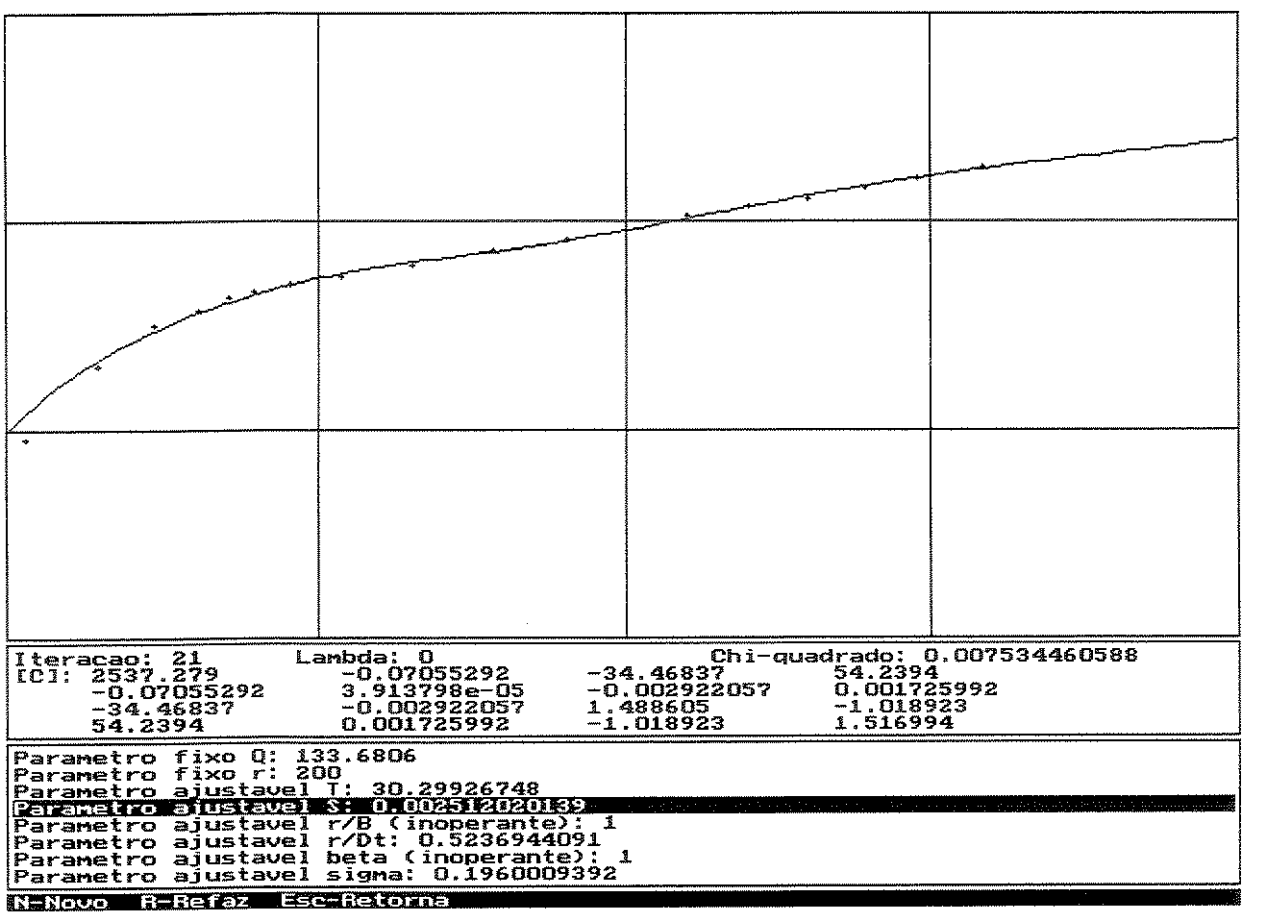

Figura 13

Como se observa a concordância destes resultados com aqueles fornecidos por Walton ainda não é boa. Na realidade, visualmente, a concordância dos pontos com a curva do modelo parece até um pouco pior que aquela dada pelo programa ITBA, o que pode ser observado comparando-se as figuras 13 e 15. Ao se entrar no programa com os valores fornecidos por Walton a concordância piora, ainda, um pouco mais, coforme pode ser percebido na figura 14. Há ainda outras razões para se crer que são os resultados fornecidos pelo ITBA os melhores: Quanto a metodologia de ajuste, enquanto o ITBA utilizou a técnica numérica de minimização dos quadrados dos resíduos, os resultados fornecidos por Walton foram provavelmente frutos de um ajuste visual, o qual oferece dificuldade em se definir com precisão o valor do parâmetro do par de curvas tipo. Além do mais, uma das condições para que o método gráfico de ajuste, empregado por Walton, dê resultados precisos, conforme ele mesmo colocou na página 224 de seu trabalho, eqüivale a se ter um valor para σ menor que 0,01, o que de fato não ocorreu. Recorda-se ainda uma outra razão para a imprecisão dos resultados: os valores expressos no arquivo de dados boulton.dmt, do rebaixamento versus tempo, foram obtidos não a partir de uma tabela, mas a partir da leitura de um gráfico com pouca precisão apresentado por Walton.

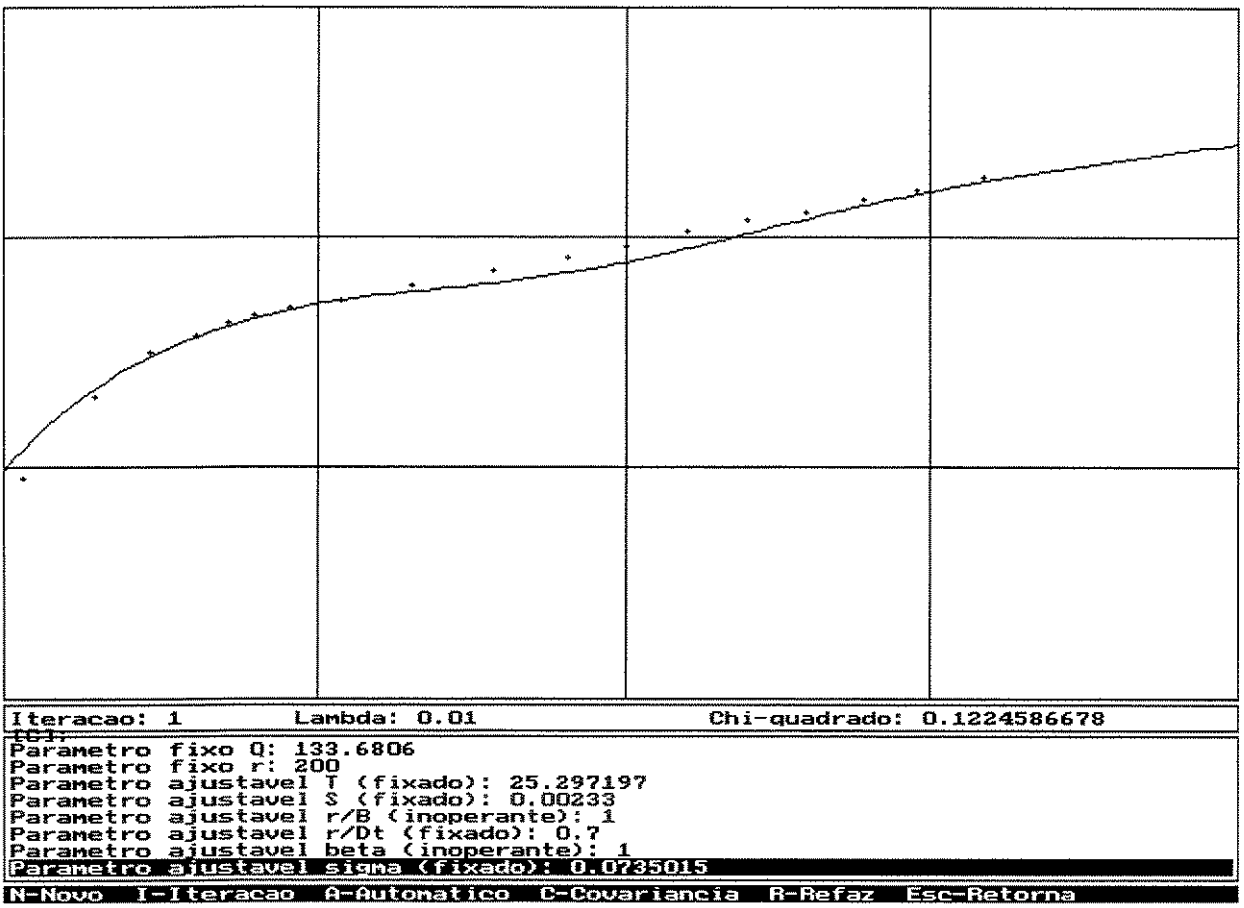
Iteracao: 1 Lambda: 0.01 Chi-quadrado: 0.1224586678
[C]:
Parametro fixo Q: 133.6806
Parametro fixo r: 200
Parametro ajustavel T (fixado): 25.297197
Parametro ajustavel S (fixado): 0.00233
Parametro ajustavel r/B (inoperante): 1
Parametro ajustavel r/Dt (fixado): 0.7
Parametro ajustavel beta (inoperante): 1
Parametro ajustavel sigma (fixado): 0.0735015
N-Novo I-Iteracao A-Automatico C-Covariancia R-Refaz Esc-Retorna

Figura 14

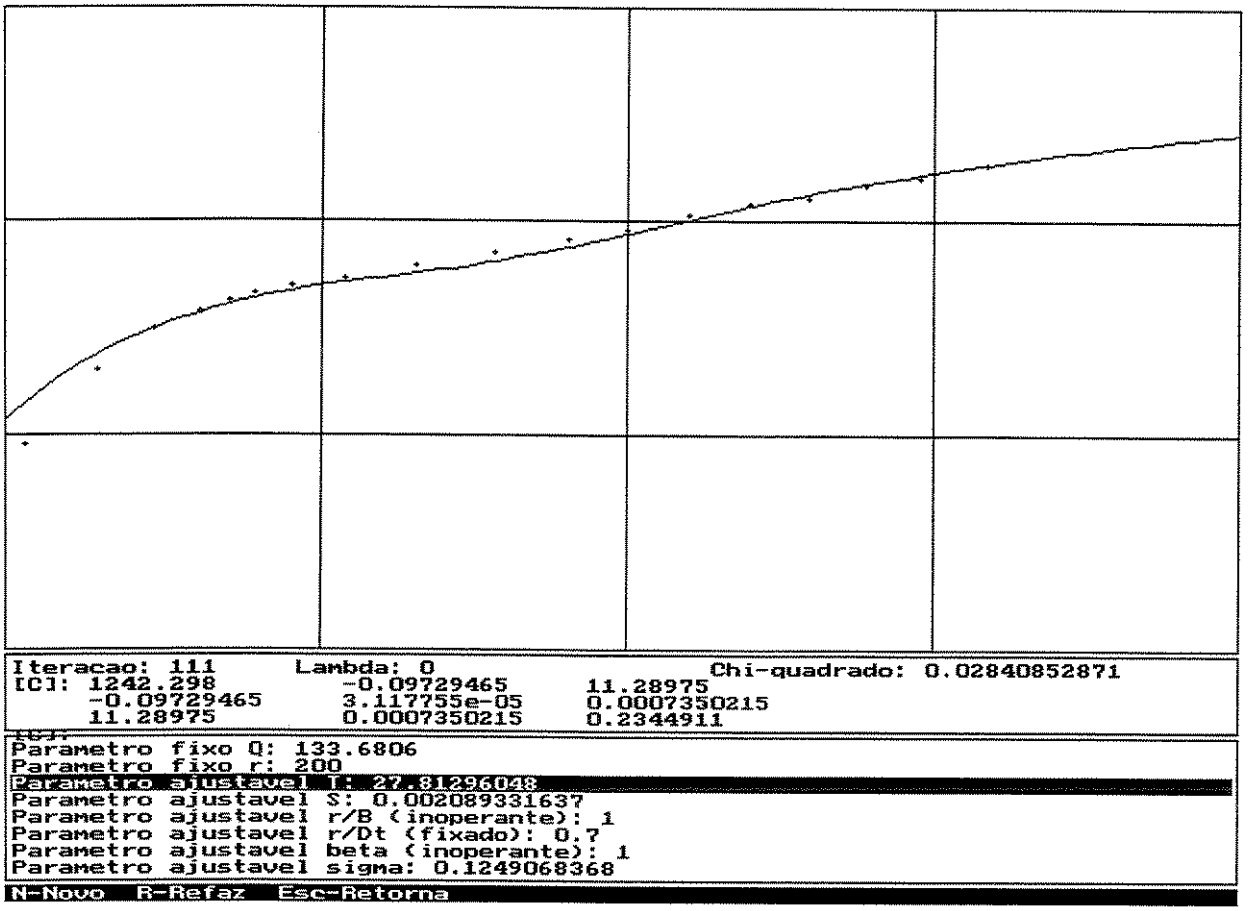

Figura 15

4.4 Aplicação para o modelo de Neuman

Os dados para verificação do modelo de Neuman foram obtidos de KRUSEMAN & RIDDER (1990), em um caso por eles mencionado no exemplo 5.1. Um perfil litoestatigráfico dos dados de perfuração mostra um aqüífero livre, composto por diversas zonas de granulometrias variadas, com uma espessura total da ordem de 21 m, sobre uma base impermeável de argila. O poço de bombeamento era aberto de uma profundidade de 10 m até 21 m. A observaçào do rebaixamento foi realizada em um piezômetro aberto em torno da profundidade de 16 m. Os valores de rebaixamento versus tempo encontram-se no arquivo neuman.dmt, o qual está listado no Anexo B.4, e cujas unidades respectivamente são o metro e o

minuto. Nesta seção, estas são as unidades básicas, de onde derivam todas as outras. A vazão Q mantida constante no poço de bombeamento, e a distância r deste ao piezômetro, estão a seguir:

$$Q = 36{,}37\ m^3/h = 0{,}606167\ m^3/min$$

$$r = 90\ m$$

Os valores dos parâmetros ajustáveis do modelo devido ao ajuste realizado pelo programa, ajuste este mostrado na figura 16, são:

$$T = 0{,}928815\ m^2/min$$

$$S = 0{,}000560198$$

$$\beta = 0{,}0127802$$

$$\sigma = 0{,}0497633$$

T, S e σ possuem o mesmo sisnificado da seção anterior. β, definido por $K_z r^2/K_r b^2$, onde K_z é a permeabilidade vertical, K_r é apermeabilidade horizontal, e b é a espessura saturada inicial do aqüífero, especifica um par de curvas tipo do modelo, utilizadas quando se procede o ajuste do modelo de forma visual.

Os respectivos valores obtidos por meio do ajuste com curvas tipo, e fornecidos por Kruseman e Ridder, são:

$$T = 1.531{,}5\ m^2/d\ (\text{média entre } 1.447 \text{ e } 1.616) = 1{,}06354\ m^2/min$$

$$S = 5{,}2 \times 10^{-4}$$

$$\beta = 0{,}01$$

$$\sigma = 0{,}106122$$

A qualidade do ajuste para estes valores pode ser observada na figura 17.

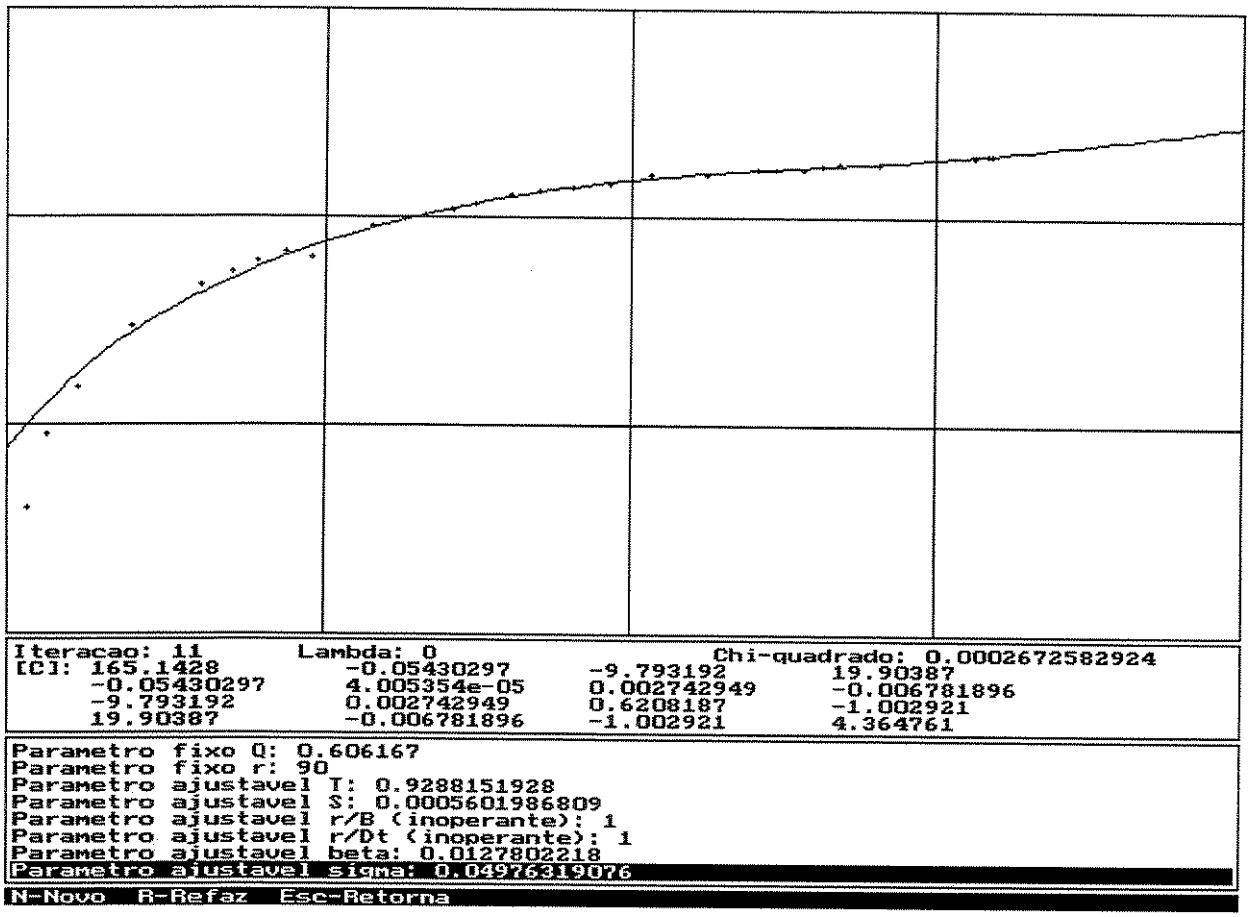

Figura 16

Os correspondentes erros percentuais, referenciados ao resultado de Kruseman e Ridder, são: -12,67 % para a transmissividade, 7,7 % para o coeficiente de armazenabilidade, 28,80 % para β, e -53,31 % para σ. Apesar destas discordâncias, que de fato é muito significativa para σ, observa-se que os resultados do programa estão associados a um chi-quadrado de 0,000267, o qual é menor que o chi-quadrado de 0,000928, que o programa indica quando se fixa os valores dos parâmetros ajustáveis, igualando-os ao resultado de Kruseman e Ridder. Observa-se, entretanto, que visualmente é difícil perceber qual dos dois resultados fornece o melhor ajuste entre a curva do modelo e os pontos do teste de bombeamento.

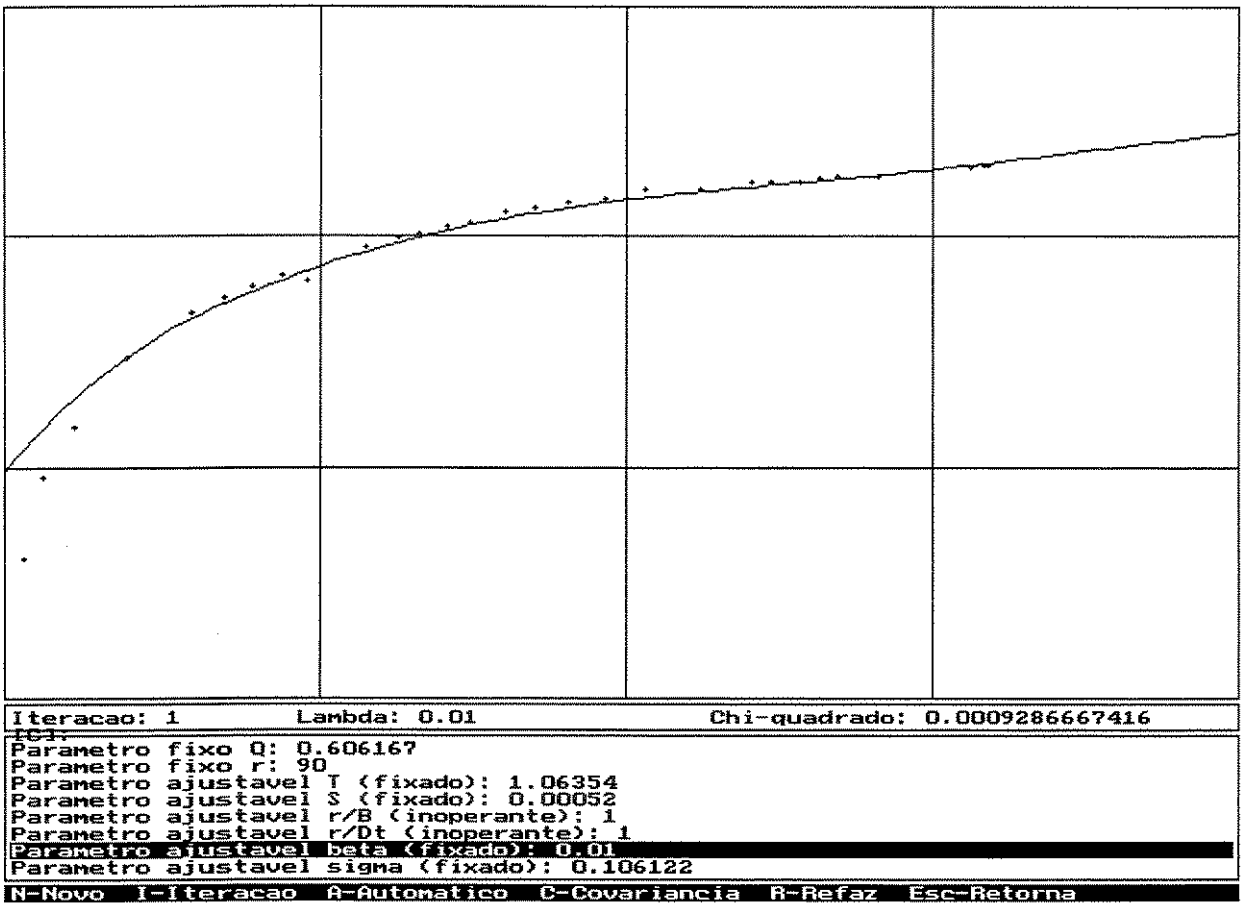

Figura 17

4.5 Testes utilizando dados sintéticos

Como, nos exemplos acima, as concordâncias entre os resultados fornecidos pelo programa ITBA, e aqueles da literatura, de um modo geral não foram tão boas, mais uma verificação foi realizada para cada modelo, utilizado-se dados sintéticos acrescentados de um pequeno ruído randômico. Estes dados foram gerados para um intervalo de tempo variando entre 1 e 1.000.000, sempre com o seguinte conjunto de valores de parâmetros, qualquer que seja o modelo: Q = 10, r = 100, T = 1, S = 0,001, r/B = 0,1, r/Dt = 0,11, β = 0,12 e σ = 0,01. Quanto ao ruído adicionado aos dados, este tinha como módulo máximo, 0.001 % do valor gerado. As figuras 18 a 21 apresentam os resultados dos ajustes dos modelos aos dados, fornecidos pelo ITBA, na seguinte ordem de modelos: Theis, Hantush, Boulton e Neuman.

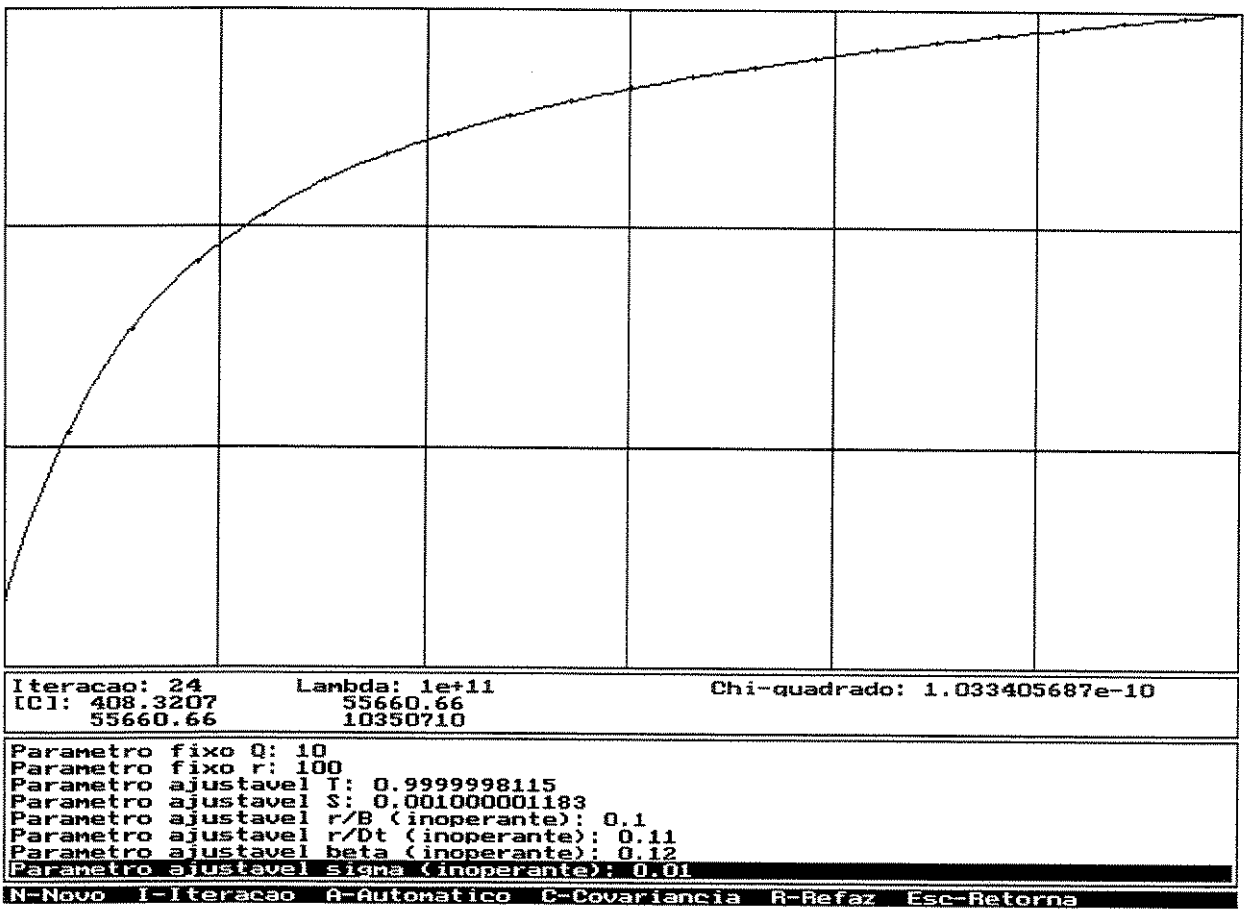
Iteracao: 24 Lambda: 1e+11 Chi-quadrado: 1.033405687e-10
[C]: 408.3207 55660.66
55660.66 10350710
Parametro fixo Q: 10
Parametro fixo r: 100
Parametro ajustavel T: 0.9999998115
Parametro ajustavel S: 0.001000001183
Parametro ajustavel r/B (inoperante): 0.1
Parametro ajustavel r/Dt (inoperante): 0.11
Parametro ajustavel beta (inoperante): 0.12
Parametro ajustavel sigma (inoperante): 0.01
N-Novo I-Iteracao A-Automatico C-Covariancia R-Refaz Esc-Retorna

Figura 18

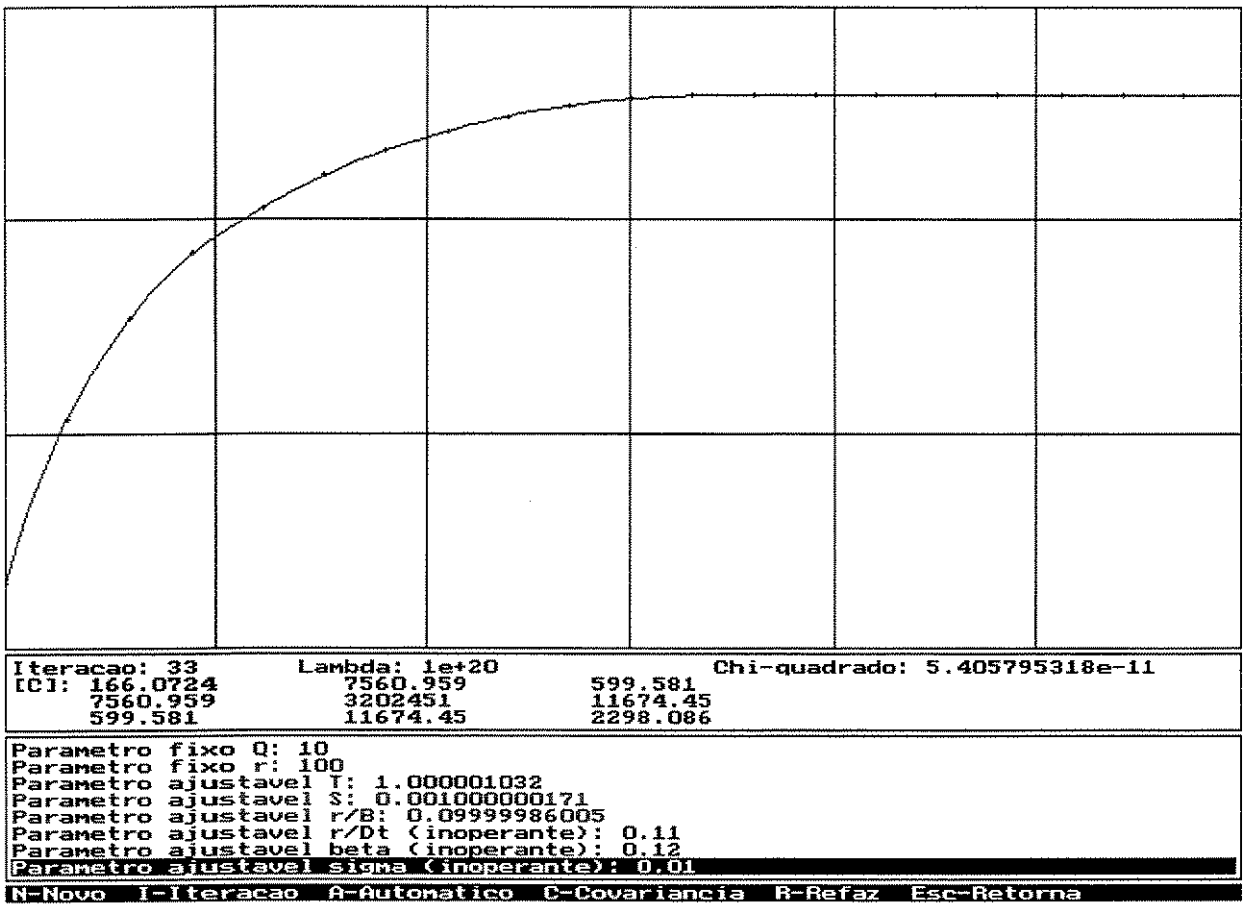
Iteracao: 33 Lambda: 1e+20 Chi-quadrado: 5.405795318e-11
[C]: 166.0724 7560.959 599.581
7560.959 3202451 11674.45
599.581 11674.45 2298.086
Parametro fixo Q: 10
Parametro fixo r: 100
Parametro ajustavel T: 1.000001032
Parametro ajustavel S: 0.001000000171
Parametro ajustavel r/B: 0.09999986005
Parametro ajustavel r/Dt (inoperante): 0.11
Parametro ajustavel beta (inoperante): 0.12
Parametro ajustavel sigma (inoperante): 0.01
N-Novo I-Iteracao A-Automatico C-Covariancia R-Refaz Esc-Retorna

Figura 19

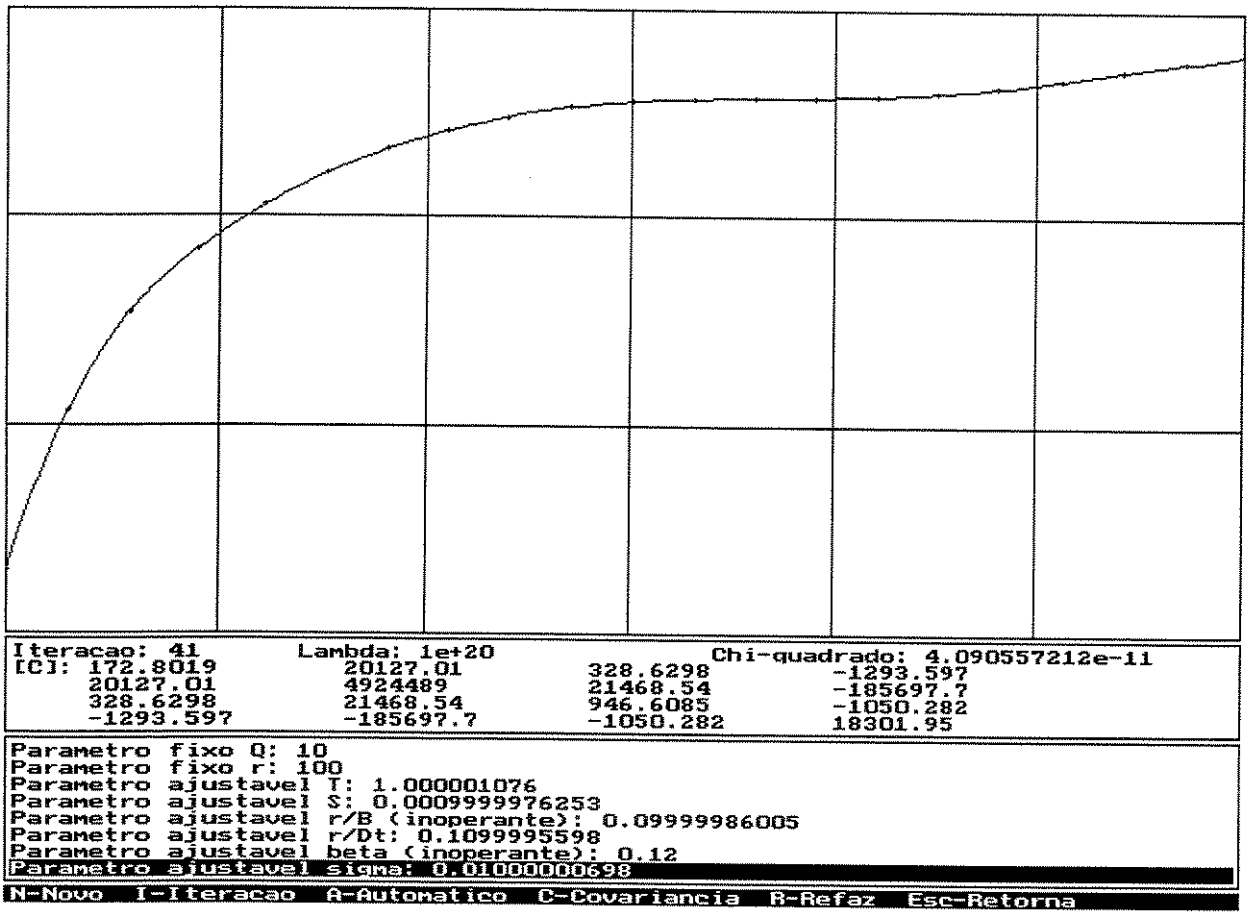
Iteracao: 41 Lambda: 1e+20 Chi-quadrado: 4.090557212e-11
[C]: 172.8019 20127.01 328.6298 -1293.597
20127.01 4924489 21468.54 -185697.7
328.6298 21468.54 946.6085 -1050.282
-1293.597 -185697.7 -1050.282 18301.95
Parametro fixo Q: 10
Parametro fixo r: 100
Parametro ajustavel T: 1.000001076
Parametro ajustavel S: 0.0009999976253
Parametro ajustavel r/B (inoperante): 0.09999986005
Parametro ajustavel r/Dt: 0.1099995598
Parametro ajustavel beta (inoperante): 0.12
Parametro ajustavel sigma: 0.01000000698
N-Novo I-Iteracao A-Automatico C-Covariancia R-Refaz Esc-Retorna

Figura 20

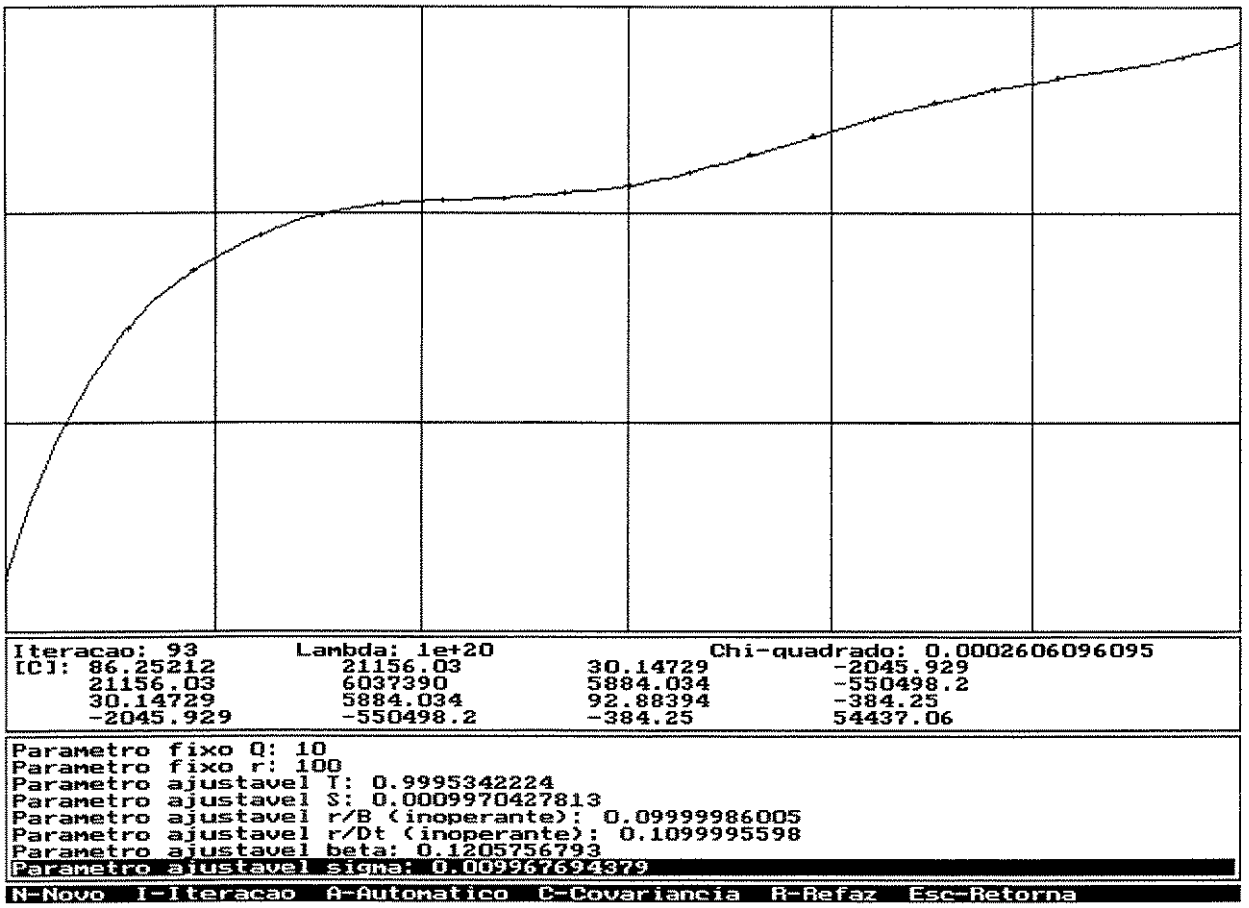

Figura 21

Observando-se as figuras acima, verifica-se que para os três primeiros modelos a concordância é muito boa, o mesmo não podendo-se dizer com respeito ao modelo de Neuman, ao observar-se o qui-quadrado e os valores dos parâmetros ajustados. Foi realizada mais uma tentativa de ajuste para este modelo, eliminando-se o ruído randômico, porém os resultados permaneceram o mesmo da figura 21. Também, até o momento, ainda não se tem uma explicação para este fato.

Vale salientar que nas figuras 18 a 21 não se teve a preocupação de calcular a matriz de covariância, estando os valores de seus elementos fora de seu significado próprio.

5 Conclusões e Recomendações

Conforme a análise feita no capítulo precedente, o programa tem apresentado resultados satisfatórios. Ele se destina a facilitar o trabalho de manipulação dos dados de um teste de bombeamento, após sofridos alguns possíveis ajustes e correções necessários, como seria o caso de um rebaixamento significativo com respeito a espessura inicialmente saturada, na tentativa de a eles ajustar a curva de um determinado modelo. Devido ao uso de técnica de regressão não linear, apresentada no Capítulo 3, a precisão dos resultados é superior a de um ajuste puramente visual. Isto pode ser compreendido quando se considera a concepção estatística embutida no método dos quadrados mínimos, conforme está descrita no Capítulo 14 de PRESS et al. (1989).

Contudo, para o bom uso desta ferramenta, como seria para qualquer técnica de determinação de parâmetros por meio do ajuste de dados a curvas tipo, é importante que se conheça bem os modelos implementados, suas considerações e limitações. É bom que se tenha, também, algumas informações geológicas (como o perfil dos poços) e locais, para que se possa planejar bem o teste de bombeamento, ou selecionar o modelo apropriado para o ajuste.

Ressalte-se, ainda, que o programa não foi exaustivamente testado, apesar de que os testes já efetuados sugiram sua confiabilidade. Assim, reforça-se a necessidade de uma análise crítica dos resultados.

Na realidade, já existem diversos programas deste gênero no mercado, inclusive com mais recursos e melhor performance, como pôde ser visto no Capítulo 2. Como se observa, o mérito deste programa consiste em ser mais um software nacional, desenvolvido de forma

independente, podendo ser aprimorado em outros trabalhos. Neste sentido, o programa poderia ser aperfeiçoado com a inclusão de novos modelos, com a melhoria da técnica de regressão não linear (ver técnicas robustas em PRESS et al. [1989, seções 14.6 e 10.4]), com as delimitações das faixas de erros dos resultados (ver ROSA & HORNE [1983, Apêndice A] e PRESS et al. [1989, Seção 14.5]), com a inclusão de técnicas que permitam um tratamento ou correção preliminar dos dados, como no caso exemplificado no início e comentado por WALTON (1970, p. 224), além de se substituir a ferramenta de desenvolvimento, no caso o Turbo C, versão 2.0, por uma mais moderna, como o Visual C++ ou Delphi. Outras sugestões para o aperfeiçoamento incluem: flexibilidade de se definir a escala (linear ou logarítmica) de cada eixo do gráfico, possibilidade de se trabalhar com modelos cujo ajuste dos pontos se faz com uma linha reta, elaboração uma tela de apresentação, especificação de unidades e parâmetros fixos no arquivo de dados, que deve poder ser editado. Pode-se também experimentar outra técnica de busca restrita de parâmetros, como aquela apresentada por ROSA & HORNE (1983).

Ainda uma outra sugestão para o aperfeiçoamento do programa, cujos resultados não são certos, mas que teriam de ser estudados, consiste em se modelar a partir de parâmetros adimensionais; pois, como se pode observar com alguns modelos, ao se expandir muito os limites do gráfico, a curva começa a apresentar distorções. Acredita-se que este problema se deve a impresisões oriundas do algoritmo de STEHFEST (1970), que foi utilizado para inversão da transformada de Laplace, as quais estão associadas a limitação da representação em ponto flutuante, quanto ao número de dígitos significativos.

Durante este trabalho procurou-se, também, em uma forma de ajuste numérico das funções dos modelos, que diferentemente da minimização da soma dos quadrados dos resíduos, se assemelhasse mais ao ajuste visual, que normalmente se faz entre o gráfico dos pontos do teste de bombeamento e o gráfico da(s) curva(s) tipo de um modelo escolhido. Conforme SUGAHARA (1996) observou, ocorrem divergências entre os resultados das duas formas de ajuste. Porém, um estudo da concepção estatística embutida no método dos quadrados mínimos,

conforme descrito no Capítulo 14 de PRESS et al. (1989), sugere que este método fornece resultados mais realísticos. Assim, após esta descoberta, abandonou-se a idéia em questão.

Finalmente, informa-se que durante o desenvolvimento deste trabalho, buscou-se ainda a implementação de um quinto modelo, relativo a poços parcialmente penetrantes em aqüíferos anisotrópicos livres (STRELTSOVA (1974)). Porém, devido a dificuldades em se produzir solução numérica com precisão satisfatória, de uma integral da equação do modelo, e ao prazo para finalização do curso, esta busca não obteve sucesso. As técnicas de integração tentadas foram: Romberg (Seção 4.3 de PRESS et al[1989]), quadraturas de Gauss-Legendre e Gauss-Laguerre (Seção 4.5), Runge-Kutta com controle do tamanho de passo adaptativo (Seção 15.2), e Bulirsch-Stoer (Seção 15.4). Outras técnicas, inclusive mais elementares, destinadas a integrais impróprias, são apresentadas no Capítulo 4 de Press. Nenhuma delas foi ainda experimentada pelo autor deste trabalho, podendo ser a chave da solução para alguém que se proponha a resolver o problema.

Anexo A

Este anexo traz as listagens dor arquivos que compõe o programa: num.c e tela.c.

A.1 num.c

```
#include <math.h>
#include <stdio.h>
#include <conio.h>

typedef double glndata[200 /*1..ndata*/], glmma[9 /*1..mma*/],
              glncabynca[9 /*1..nca*/][9 /*1..nca*/],
              glnalbynal[9 /*1..nalp*/][9 /*1..nalp*/],
              glnpbynp[9 /*1..np*/][9 /*1..np*/],
              glnpbymp[9 /*1..np*/][2 /*1..mp*/],
              glnpnp[9 /*1..np*/][9 /*1..np*/];
typedef int gllista[9 /*1..mma*/], glnp[9 /*1..np*/];

glmma glbeta, Gamax, Gamin;
int Gmod, GNStehf;
double glochisq, Gln2, Gpi, GVStehf[20], GzeroJ0[200];

double bessj0(double x) {
  double ax, xx, z, y, ans, ans1, ans2;
  if (fabs(x) < 8.0) {
    y = x*x;
    ans1 = 57568490574.0+y*(-13362590354.0+y*(651619640.7
         +y*(-11214424.18+y*(77392.33017+y*(-184.9052456)))));
    ans2 = 57568490411.0+y*(1029532985.0+y*(9494680.718
         +y*(59272.64853+y*(267.8532712+y*1.0))));
    return (ans1/ans2);
  } else {
    ax = fabs(x);
    z = 8.0/ax;
    y = z*z;
    xx = ax-0.785398164;
```

```
    ans1 = 1.0+y*(-0.1098628627e-2+y*(0.2734510407e-4
          +y*(-0.2073370639e-5+y*0.2093887211e-6)));
    ans2 = -0.1562499995e-1+y*(0.1430488765e-3
          +y*(-0.6911147651e-5+y*(0.7621095161e-6-y*0.934945152e-7)));
    ans = sqrt(0.636619772/ax)*(cos(xx)*ans1-z*sin(xx)*ans2);
    return (ans);
  }
}

double bessi0(double x) {
  double ax, y, ans;
  if (fabs(x) < 3.75) {
    y = x/3.75;
    y *= y;
    ans = 1.0+y*(3.5156229+y*(3.0899424+y*(1.2067492+y*(0.2659732+
         y*(0.360768e-1+y*0.45813e-2)))));
  } else {
    ax = fabs(x);
    y = 3.75/ax;
    ans = (exp(ax)/sqrt(ax))*(0.39894228+y*(0.1328592e-1+y*(0.225319e-2+
         y*(-0.157565e-2+y*(0.916281e-2+y*(-0.2057706e-1+y*(0.2635537e-1+
         y*(-0.1647633e-1+y*0.392377e-2)))))));
  }
  return (ans);
}

double bessi1(double x) {
  double ax, y, ans;
  if (fabs(x) < 3.75) {
    y = x/3.75;
    y *= y;
    ans = x*(0.5+y*(0.87890594+y*(0.51498869+y*(0.15084934+y*(0.2658733e-1+
         y*(0.301532e-2+y*0.32411e-3))))));
  } else {
    ax = fabs(x);
    y = 3.75/ax;
    ans = 0.2282967e-1+y*(-0.2895312e-1+y*(0.1787654e-1-y*0.420059e-2));
    ans = 0.39894228+y*(-0.3988024e-1+y*(-0.362018e-2+y*(0.163801e-2+
         y*(-0.1031555e-1+y*ans))));
    ans = (exp(ax)/sqrt(ax))*ans;
  }
  return (ans);
}

double bessk0(double x) {
  double y, ans;
  if (x <= 2.0) {
    y = x*x/4.0;
    ans = (-log(x/2.0)*bessi0(x))+(-0.57721566+y*(0.42278420+y*(0.23069756+
         y*(0.3488590e-1+y*(0.262698e-2+y*(0.10750e-3+y*0.74e-5))))));
  } else {
```

```
    y = (2.0/x);
    ans = (exp(-x)/sqrt(x))*(1.25331414+y*(-0.7832358e-1+y*(0.2189568e-1+
        y*(-0.1062446e-1+y*(0.587872e-2+y*(-0.251540e-2+y*0.53208e-3))))));
  }
  return (ans);
}

double besskl(double x) {
  double y, ans;
  if (x <= 2.0) {
    y = x*x/4.0;
    ans = (log(x/2.0)*bessil(x))+(1.0/x)*(1.0+y*(0.15443144+y*(-0.67278579+
        y*(-0.18156897+y*(-0.1919402e-1+y*(-0.110404e-2+y*(-0.4686e-4)))))));
  } else {
    y = 2.0/x;
    ans = (exp(-x)/sqrt(x))*(1.25331414+y*(0.23498619+y*(-0.3655620e-1+
        y*(0.1504268e-1+y*(-0.780353e-2+y*(0.325614e-2+y*(-0.68245e-3)))))));
  }
  return (ans);
}

double secantes(double (*f)(), double x0, double x1) {
  double x;
  do {
    x = (x0*(*f)(x1)-x1*(*f)(x0))/((*f)(x1)-(*f)(x0));
    x0 = x1;
    x1 = x;
  } while (fabs(x1-x0) > 1e-6*x1+1e-20);
  return (x1);
}

void inivar(void) {
  double fatorial[20], h;
  int i, K;
  GNStehf = 10;
  Gln2 = log(2.0);
  Gpi = acos(-1.0);
  fatorial[0] = 1.0;
  for (i = 1; i <= GNStehf; i++)
    fatorial[i] = i*fatorial[i-1];
  for (i = 1; i <= GNStehf; i++) {
    GVStehf[i] = 0.0;
    for (K = (i+1)/2; K <= (i < GNStehf/2 ? i : GNStehf/2); K++)
      GVStehf[i] += pow(K, GNStehf/2+1)*fatorial[2*K]/(fatorial[GNStehf/2-K]*
                    fatorial[K]*fatorial[K]*fatorial[i-K]*fatorial[2*K-i]);
    GVStehf[i] *= pow(-1.0, GNStehf/2+i);
  }
  GzeroJ0[0] = 0.0;
  GzeroJ0[1] = secantes(bessj0, 2.3, 2.5);
  h = 3.1;
  for (i = 2; i < 200; i++) {
```

```
    GzeroJ0[i] = secantes(bessj0, GzeroJ0[i-1]+0.9*h, GzeroJ0[i-1]+1.1*h);
    h = GzeroJ0[i]-GzeroJ0[i-1];
  }
}

void Theis(double t, glmma a, double *s, glmma dsda) {
  double aux[6], soma[3], w;
  int i;
  aux[0] = Gln2/t;
  aux[1] = aux[0]*a[7]/(2.0*Gpi*a[1]);
  aux[2] = a[8]*sqrt(a[2]/a[1]);
  soma[0] = 0.0;
  if (dsda != NULL) {
    soma[1] = 0.0;
    soma[2] = 0.0;
  }
  for (i = 1; i <= GNStehf; i++) {
    w = aux[0]*i;
    aux[3] = aux[2]*sqrt(w);
    aux[4] = GVStehf[i]*bessk0(aux[3])/w;
    soma[0] += aux[4];
    if (dsda != NULL) {
      aux[5] = GVStehf[i]*bessk1(aux[3])*aux[3]/(2.0*w);
      soma[1] += aux[5]-aux[4];
      soma[2] -= aux[5];
    }
  }
  *s = aux[1]*soma[0];
  if (dsda != NULL) {
    dsda[1] = aux[1]*soma[1]/a[1];
    dsda[2] = aux[1]*soma[2]/a[2];
  }
}

void Hantush(double t, glmma a, double *s, glmma dsda) {
  double aux[8], soma[4], w;
  int i;
  aux[0] = Gln2/t;
  aux[1] = aux[0]*a[7]/(2.0*Gpi*a[1]);
  aux[2] = a[8]*a[8]*a[2]/a[1];
  aux[3] = a[3]*a[3];
  soma[0] = 0.0;
  if (dsda != NULL) {
    soma[1] = 0.0;
    soma[2] = 0.0;
    soma[3] = 0.0;
  }
  for (i = 1; i <= GNStehf; i++) {
    w = aux[0]*i;
    aux[4] = sqrt(aux[2]*w+aux[3]);
    aux[5] = GVStehf[i]*bessk0(aux[4])/w;
```

```
    soma[0] += aux[5];
    if (dsda != NULL) {
      aux[6] = GVStehf[i]*bessk1(aux[4])/aux[4];
      aux[7] = 0.5*aux[2]*aux[6];
      soma[1] += aux[7]-aux[5];
      soma[2] -= aux[7];
      soma[3] -= aux[6]/w;
    }
  }
  *s = aux[1]*soma[0];
  if (dsda != NULL) {
    dsda[1] = aux[1]*soma[1]/a[1];
    dsda[2] = aux[1]*soma[2]/a[2];
    dsda[3] = aux[1]*soma[3]*a[3];
  }
}

void Boulton(double t, glmma a, double *s, glmma dsda) {
  double aux[13], soma[5], w;
  int i;
  aux[0] = Gln2/t;
  aux[1] = aux[0]*a[7]/(2.0*Gpi*a[1]);
  aux[2] = a[8]*a[8]*a[2]/a[1];
  aux[4] = a[4]*a[4];
  soma[0] = 0.0;
  if (dsda != NULL) {
    soma[1] = 0.0;
    soma[2] = 0.0;
    soma[3] = 0.0;
    soma[4] = 0.0;
  }
  for (i = 1; i <= GNStehf; i++) {
    w = aux[0]*i;
    aux[3] = aux[2]*w;
    aux[5] = a[6]+aux[3]/aux[4];
    aux[6] = 1.0+1.0/aux[5];
    aux[7] = sqrt(aux[3]*aux[6]);
    aux[8] = GVStehf[i]*bessk0(aux[7])/w;
    soma[0] += aux[8];
    if (dsda != NULL) {
      aux[9] = -GVStehf[i]*bessk1(aux[7])/(2.0*aux[7]*w);
      aux[10] = -aux[3]/(aux[5]*aux[5]);
      aux[11] = aux[9]*aux[10];
      aux[12] = aux[9]*(aux[6]+aux[10]/aux[4])*aux[3];
      soma[1] -= aux[8]+aux[12];
      soma[2] += aux[12];
      soma[3] -= aux[11]*aux[3];
      soma[4] += aux[11];
    }
  }
  *s = aux[1]*soma[0];
```

```
  if (dsda != NULL) {
    dsda[1] = aux[1]*soma[1]/a[1];
    dsda[2] = aux[1]*soma[2]/a[2];
    dsda[4] = aux[1]*soma[3]*2.0/(aux[4]*a[4]);
    dsda[6] = aux[1]*soma[4];
  }
}

void HNeuman(double aa, double t, glmma a, double soma[], int n) {
  double aux[14], w;
  int i;
  aux[0] = Gln2/t;
  aux[1] = aa*aa;
  for (i = 0; i <= n; i++)
    soma[i] = 0.0;
  for (i = 1; i <= GNStehf; i++) {
    w = aux[0]*i;
    aux[2] = (aux[1]*a[1]+w*a[2]);
    aux[3] = a[8]*sqrt(aux[2]/(a[5]*a[1]));
    aux[4] = tanh(aux[3]);
    aux[5] = 1.0/(a[6]*aux[2]/a[2]+w*aux[3]/aux[4]);
    aux[6] = GVStehf[i]*(1/w-aux[5])/aux[2];
    soma[0] += aux[6];
    if (n == 0)
      continue;
    aux[7] = GVStehf[i]*aux[5]*aux[5];
    aux[8] = w*a[8]*a[8]/(2.0*a[5]*a[1])*
           (1.0+1.0/(aux[3]*aux[4])-1.0/(aux[4]*aux[4]));
    aux[9] = aux[7]/aux[2];
    aux[10] = a[6]*aux[1]/a[2];
    aux[11] = w*aux[8];
    aux[12] = a[2]/a[1];
    aux[13] = aux[6]/aux[2];
    soma[1] += aux[9]*(aux[10]-aux[11]*aux[12])-aux[13]*aux[1];
    soma[2] += aux[9]*(aux[11]-aux[10]*a[1]/a[2])-aux[13]*w;
    soma[3] -= aux[7]*aux[8];
    soma[4] += aux[7];
  }
  if (n != 0) {
    soma[4] /= a[2];
    soma[3] /= a[5];
  }
  aux[0] *= a[7]/(2.0*Gpi);
  for (i = 0; i <= n; i++)
    soma[i] *= aa*bessj0(a[8]*aa)*aux[0];
}

void Neuman(double t, glmma a, double *s, glmma dsda) {
  static double x[6] = {0.0, 0.1488743389, 0.4333953941, 0.6794095682,
                        0.8650633666, 0.97390652},
              w[6] = {0.0, 0.2955242247, 0.2692667193, 0.2190863625,
```

```
                    0.1494513491, 0.06667134};
double aa, bb, sN[5], sNbarra[5], sN_1[5], sN_1barra[5], ss[5], xr, xm, dx,
      s1[5], s2[5];
int i, imax, n, cont, j;
if (dsda == NULL)
  imax = 0;
else
  imax = 4;
for (i = 0; i <= imax; i++) {
  sN[i] = 0.0;
  sNbarra[i] = 0.0;
}
bb = 0.0;
for (n = 1; n < 200; n++) {
  aa = bb;
  bb = GzeroJ0[n]/a[8];
  /*As 13 linhas que seguem foram adaptadas de qgaus()*/
  xm = 0.5*(bb+aa);
  xr = 0.5*(bb-aa);
  for (i = 0; i <= imax; i++)
    ss[i] = 0.0;
  for (j = 1; j <= 5; j++) {
    dx = xr*x[j];
    HNeuman(xm+dx, t, a, s1, imax);
    HNeuman(xm-dx, t, a, s2, imax);
    for (i = 0; i <= imax; i++)
     ss[i] += w[j]*(s1[i]+s2[i]);
  }
  for (i = 0; i <= imax; i++)
    ss[i] *= xr;
  /*Fim da adaptacao*/
  cont = 0;
  for (i = 0; i <= imax; i++) {
    sN_1[i] = sN[i];
    sN[i] += ss[i];
    sN_1barra[i] = sNbarra[i];
    sNbarra[i] = (sN[i]+sN_1[i])/2.0;
    if (fabs(sNbarra[i]-sN_1barra[i]) < 1.0e-2*fabs(sNbarra[i])+1.0e-20)
     cont++;
  }
  if (cont > imax && n >= 3) {
    break;
  }
}
*s = sNbarra[0];
if (dsda != NULL) {
  dsda[1] = sNbarra[1];
  dsda[2] = sNbarra[2];
  dsda[5] = sNbarra[3];
  dsda[6] = sNbarra[4];
}
```

```
}

void funcs(double xx, glmma a, double *yfit, glmma dyda, int mma) {
  switch (Gmod) {
    case 1: Theis(xx, a, yfit, dyda); break;
    case 2: Hantush(xx, a, yfit, dyda); break;
    case 3: Boulton(xx, a, yfit, dyda); break;
    case 4: Neuman(xx, a, yfit, dyda); break;
  }
  mma++;
}

int /*originalmente void*/ gaussj(glnpbynp a, int n, int np, glnpbymp b, int m,
                                  int mp) {
  double big, dum, pivinv;
  int i, icol, irow, j, k, l, ll;
  glnp indxc, indxr, ipiv;
  for (j = 1; j <= n; j++)
    ipiv[j] = 0;
  for (i = 1; i <= n; i++) {
    big = 0.0;
    for (j = 1; j <= n; j++)
      if (ipiv[j] != 1)
       for (k = 1; k <= n; k++)
         if (ipiv[k] == 0)
           if (fabs(a[j][k]) >= big) {
             big = fabs(a[j][k]);
             irow = j;
             icol = k;
           }
         else if (ipiv[k] > 1)
           return (1); /*pause 1 in GAUSSJ - singular matrix*/
    ipiv[icol]++;
    if (irow != icol) {
      for (l = 1; l<= n; l++) {
       dum = a[irow][l];
       a[irow][l] = a[icol][l];
       a[icol][l] = dum;
       }
      for (l = 1; l <= m; l++) {
       dum = b[irow][l];
       b[irow][l] = b[icol][l];
       b[icol][l] = dum;
       }
    }
    indxr[i] = irow;
    indxc[i] = icol;
    if (a[icol][icol] == 0.0)
      return (2); /*pause 2 in GAUSSJ - singular matrix*/
    pivinv = 1.0/a[icol][icol];
    a[icol][icol] = 1.0;
```

```
    for (l = 1; l <= n; l++)
      a[icol][l] *= pivinv;
    for (l = 1; l <= m; l++)
      b[icol][l] *= pivinv;
    for (ll = 1; ll <= n; ll++)
      if (ll != icol) {
       dum = a[ll][icol];
       a[ll][icol] = 0.0;
       for (l = 1; l <= n; l++)
         a[ll][l] -= a[icol][l]*dum;
       for (l = 1; l <= m; l++)
         b[ll][l] -= b[icol][l]*dum;
       }
  }
  for (l = n; l >= 1; l--)
    if (indxr[l] != indxc[l])
      for (k = 1; k <= n; k++) {
       dum = a[k][indxr[l]];
       a[k][indxr[l]] = a[k][indxc[l]];
       a[k][indxc[l]] = dum;
       }
  mp = mp; /*linha acrescentada*/
  np = np; /*linha acrescentada*/
  return (0); /*linha acrescentada*/
}

int /*originalmente void*/ jacobi(glnpnp a, int n, glmma /*glnp[1..np]*/ d,
                              glnpnp v, int *nrot) {
  int j, iq, ip, i;
  double tresh, theta, tau, t, sm, s, h, g, c, b[101 /*1..nmax*/],
      z[101 /*1..nmax*/];
  for (ip = 1; ip <= n; ip++) {
    for (iq = 1; iq <= n; iq++)
      v[ip][iq] = 0.0;
    v[ip][ip] = 1.0;
  }
  for (ip = 1; ip <= n; ip++) {
    b[ip] = a[ip][ip];
    d[ip] = b[ip];
    z[ip] = 0.0;
  }
  *nrot = 0;
  for (i = 1; i <= 50; i++) {
    sm = 0.0;
    for (ip = 1; ip <= n-1; ip++)
      for (iq = ip+1; iq <= n; iq++)
       sm += fabs(a[ip][iq]);
    if (sm == 0.0)
      return (0); /*linha modificada*/
    if (i < 4)
      tresh = 0.2*sm/(n*n);
```

```
else
  tresh = 0.0;
for (ip = 1; ip <= n-1; ip++)
  for (iq = ip+1; iq <= n; iq++) {
   g = 100.0*fabs(a[ip][iq]);
   if (i > 4 && fabs(d[ip])+g == fabs(d[ip]) &&
      fabs(d[iq])+g == fabs(d[iq]))
     a[ip][iq] = 0.0;
   else if (fabs(a[ip][iq]) > tresh) {
     h = d[iq]-d[ip];
     if (fabs(h)+g == fabs(h))
       t = a[ip][iq]/h;
     else {
       theta = 0.5*h/a[ip][iq];
       t = 1.0/(fabs(theta)+sqrt(1.0+theta*theta));
       if (theta < 0.0)
         t = -t;
     }
     c = 1.0/sqrt(1+t*t);
     s = t*c;
     tau = s/(1.0+c);
     h = t*a[ip][iq];
     z[ip] = z[ip]-h;
     z[iq] = z[iq]+h;
     d[ip] = d[ip]-h;
     d[iq] = d[iq]+h;
     a[ip][iq] = 0.0;
     for (j = 1; j <= ip-1; j++) {
       g = a[j][ip];
       h = a[j][iq];
       a[j][ip] = g-s*(h+g*tau);
       a[j][iq] = h+s*(g-h*tau);
     }
     for (j = ip+1; j <= iq-1; j++) {
       g = a[ip][j];
       h = a[j][iq];
       a[ip][j] = g-s*(h+g*tau);
       a[j][iq] = h+s*(g-h*tau);
     }
     for (j = iq+1; j <= n; j++) {
       g = a[ip][j];
       h = a[iq][j];
       a[ip][j] = g-s*(h+g*tau);
       a[iq][j] = h+s*(g-h*tau);
     }
     for (j = 1; j <= n; j++) {
       g = v[j][ip];
       h = v[j][iq];
       v[j][ip] = g-s*(h+g*tau);
       v[j][iq] = h+s*(g-h*tau);
     }
```

```
        *nrot++;
      }
      }
  }
  for (ip = 1; ip <= n; ip++) {
    b[ip] += z[ip];
    d[ip] = b[ip];
    z[ip] = 0.0;
  }
  return (1); /*pause in routine JACOBI*/
            /*50 iterations should not happen*/
}

void mrqcof(glndata x, glndata y, glndata sig, int ndata, glmma a, int mma,
     gllista lista, int mfit, glnalbynal alpha, glmma beta, int nalp,
     double *chisq) {
  int k, j, i;
  double ymod, wt, sig2i, dy;
  glmma dyda;
  for (j = 1; j <= mfit; j++) {
    for (k = 1; k <= j; k++)
      alpha[j][k] = 0.0;
    beta[j] = 0.0;
  }
  *chisq = 0.0;
  for (i = 1; i <= ndata; i++) {
    funcs(x[i], a, &ymod, dyda, mma);
    sig2i = 1.0/(sig[i]*sig[i]);
    dy = y[i]-ymod;
    for( j = 1; j <= mfit; j++) {
      wt = dyda[lista[j]]*sig2i;
      for (k = 1; k <= j; k++)
       alpha[j][k] += wt*dyda[lista[k]];
      beta[j] += dy*wt;
    }
    *chisq += dy*dy*sig2i;
  }
  for (j = 2; j <= mfit; j++)
    for (k = 1; k <= j-1; k++)
      alpha[k][j] = alpha[j][k];
  nalp = nalp; /*linha acrescentada*/
}

int /*originalmente void*/ mrqmin(glndata x, glndata y, glndata sig, int ndata,
                              glmma a, int mma, gllista lista, int mfit,
                              glncabynca covar, glncabynca alpha, int nca,
                              double *chisq, double *alamda) {
  int k, kk, j, ihit;
  static glmma atry; /*originalmente sem static*/
  glmma da /*, d*/;
  glnpbymp oneda;
```

```
/*glncabynca v;*/
double f, aux; /*linha incluida*/
if (*alamda < 0.0) {
  kk = mfit+1;
  for (j = 1; j <= mma; j++) {
    ihit = 0;
    for (k = 1; k <= mfit; k++)
     if (lista[k] == j)
       ihit++;
    if (ihit == 0) {
     lista[kk] = j;
     kk++;
    } else if (ihit > 1) {
     return (1); /*pause 1 in routine MRQMIN*/
                 /*Improper permutation in LISTA*/
    }
  }
  if (kk != mma+1) {
    return (2); /*pause 2 in routine MRQMIN*/
              /*Improper permutation in LISTA*/
  }
  *alamda = 0.001;
  mrqcof(x, y, sig, ndata, a, mma, lista, mfit, alpha, glbeta, nca, chisq);
  glochisq = *chisq;
  for (j = 1; j <= mma; j++)
    atry[j] = a[j];
}
for (j = 1; j <= mfit; j++) {
  for (k = 1; k <= mfit; k++)
    covar[j][k] = alpha[j][k];
  covar[j][j] = alpha[j][j]*(1.0+(*alamda));
  oneda[j][1] = glbeta[j];
}
j = gaussj(covar, mfit, nca, oneda, 1, 1);
if (j > 0)
  return (3+j);
for (j = 1; j <= mfit; j++)
  da[j] = oneda[j][1];
/*Alternativa que substitui as 11 linhas acima
for (j = mfit; j >= 1; j--) {
  atry[j] = sqrt(alpha[j][j]);
  for (k = mfit; k > j; k--)
    covar[j][k] = alpha[j][k]/(atry[j]*atry[k]);
  covar[j][j] = 1.0+(*alamda);
  oneda[j][1] = glbeta[j]/atry[j];
}
if (jacobi(covar, mfit, d, v, &kk) != 0)
  return (3);
for (j = 1; j <= mfit; j++) {
  da[j] = 0.0;
  for (k = 1; k <= mfit; k++) {
```

```
    if (j <= k) {
     covar[j][k] = 0.0;
     for (kk = 1; kk <= mfit; kk++) {
       if (*alamda == 0)
         covar[j][k] += v[j][kk]*v[k][kk]/d[kk];
       else if (d[kk] > 1.0e-5)
         covar[j][k] += v[j][kk]*v[k][kk]/d[kk];
     }
     } else
     covar[j][k] = covar[k][j];
     da[j] += covar[j][k]*oneda[k][1];
  }
  da[j] /= atry[j];
}
for (j = 1; j <= mfit; j++)
  for (k = 1; k <= mfit; k++)
    covar[j][k] /= atry[j]*atry[k];
Fim da alternativa*/
if (*alamda == 0.0) {
  /*covsrt(covar, nca, mma, lista, mfit);*/
  return (0); /*linha modificada*/
}
/*Inicio de inclusao*/
f = 1.0;
for (j = 1; j <= mfit; j++) {
  aux = a[lista[j]]+f*da[j];
  if (aux > Gamax[lista[j]])
    f = (Gamax[lista[j]]-a[lista[j]])/da[j];
  else if (aux < Gamin[lista[j]])
    f = (Gamin[lista[j]]-a[lista[j]])/da[j];
}
if (f < 1.0)
  f *= 0.99999;
/*Fim de inclusao*/
for (j = 1; j <= mfit; j++)
  atry[lista[j]] = a[lista[j]]+f*da[j]; /*sem f no original*/
mrqcof(x, y, sig, ndata, atry, mma, lista, mfit, covar, da, nca, chisq);
if (*chisq < glochisq) {
  *alamda = 0.1*(*alamda);
  glochisq = *chisq;
  for (j = 1; j <= mfit; j++) {
    for (k = 1; k <= mfit; k++)
     alpha[j][k] = covar[j][k];
    glbeta[j] = f*da[j]; /*sem f no original*/
    a[lista[j]] = atry[lista[j]];
  }
} else {
  *alamda = 10.0*(*alamda);
  *chisq = glochisq;
}
return (0); /*linha incluida*/
```

```
}
```

A.2 - tela.c

```
#include <ctype.h>
#include <graphics.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include "num.c"

glmma glbeta2;
double glochisq2;

void grafico(int xmax, int ymax, double loglim[], char nomarq[],
     double xxpto[], double yypto[], int npto, double a[], double sig[],
     int ptsig) {
  static int xmaxpas, ymaxpas, xpto[200], ypto[200], modpas, nmod, xmod[300],
          ymod[300];
  static double loglimpas[4], cclx, ccly, xgrdmin, ygrdmin, yymod[300];
  static char nomarqpas[40] = "";
  double x, Dx, y;
  int i, h, v, arq;
  h = 0;
  v = 0;
  arq = 0;
  if (loglim[0] != loglimpas[0] || loglim[2] != loglimpas[2] ||
     xmax != xmaxpas) {
    cclx = xmax/(loglim[2]-loglim[0]);
    xgrdmin = (ceil(loglim[0])-loglim[0])*cclx;
    if (xgrdmin == 0.0)
      xgrdmin = cclx;
    loglimpas[0] = loglim[0];
    loglimpas[2] = loglim[2];
    xmaxpas = xmax;
    h = 1;
  }
  if (loglim[1] != loglimpas[1] || loglim[3] != loglimpas[3] ||
     ymax != ymaxpas) {
    ccly = ymax/(loglim[1]-loglim[3]);
    ygrdmin = (loglim[1]-floor(loglim[1]))*ccly;
    if (ygrdmin == 0.0)
      ygrdmin = ccly;
    loglimpas[1] = loglim[1];
    loglimpas[3] = loglim[3];
    ymaxpas = ymax;
    v = 1;
  }
```

```
if (strcmp(nomarq, nomarqpas) != 0) {
  arq = 1;
  strcpy(nomarqpas, nomarq);
}
if (h || arq || Gmod != modpas || a[0] == 0.0) {
  a[0] = 1.0;
  modpas = Gmod;
  for (i = 1; i <= npto; i++)
    xpto[i] = (log10(xxpto[i])-loglim[0])*cclx;
  i = 1;
  while (xxpto[i] <= loglim[0] && i <= npto)
    i++;
  nmod = -1;
  Dx = xmax/100.0;
  for (x = 0.0; x < xmax+Dx; x += Dx) {
    if (x > xpto[i] && i <= npto) {
     x = xpto[i];
     i++;
     }
    nmod++;
    xmod[nmod] = x;
    funcs(pow(10.0, x/cclx+loglim[0]), a, &yymod[nmod], NULL, 0);
    if (yymod[nmod] <= 0.0)
     nmod--;
    else
     ymod[nmod] = (loglim[1]-log10(yymod[nmod]))*ccly;
  }
}
if (v || arq)
  for (i = 1; i <= npto; i++)
    ypto[i] = (loglim[1]-log10(yypto[i]))*ccly;
if (v)
  for (i = 0; i <= nmod; i++)
    ymod[i] = (loglim[1]-log10(yymod[i]))*ccly;
setviewport(0, 0, xmax, ymax, 1);
clearviewport();
setcolor(LIGHTGRAY);
rectangle(0, 0, xmax, ymax);
for (x = xgrdmin; x < xmax; x += cclx)
  line(x, 0, x, ymax);
for (y = ygrdmin; y < ymax; y += ccly)
  line(0, y, xmax, y);
for (i = 1; i <= nmod; i++)
  line(xmod[i-1], ymod[i-1], xmod[i], ymod[i]);
setcolor(LIGHTCYAN);
for (i = 1; i <= npto; i++) {
  if (sig[i] == 1.0)
    setcolor(LIGHTCYAN);
  else
    if (sig[i] < 1.0)
     setcolor(WHITE);
```

```
      else
       setcolor(DARKGRAY);
    if (i == ptsig)
      setcolor(LIGHTRED);
    pieslice(xpto[i], ypto[i], 0, 360, 1);
  }
  setviewport(0, 0, xmax, getmaxy(), 0);
}

int arquivo(char nome[], double x[], double y[], double sig[],
  double lim[], int *n) {
  double xaux, yaux, xmin, xmax, ymin, ymax, naux;
  FILE *arq;
  int i;
  static char nomepas[40] = "";
  if (strcmp(nome, nomepas) == 0)
    return (1);
  arq = fopen(nome , "r");
  if (arq == NULL)
    return (2);
  naux = 0;
  while (!feof(arq) && naux < 199)
    if (fscanf(arq, "%lg %lg", &xaux, &yaux) == 2) {
      x[naux+1] = xaux;
      y[naux+1] = yaux;
      naux++;
    } else
      break;
  fclose(arq);
  if (naux == 0)
    return (3);
  strcpy(nomepas, nome);
  *n = naux;
  for (i = 1; i <= naux; i++)
    sig[i] = 1.0;
  xmin = x[1];
  xmax = x[1];
  ymin = y[1];
  ymax = y[1];
  for (i = 2; i <= naux; i++) {
    if (x[i] < xmin)
      xmin = x[i];
    else if (x[i] > xmax)
      xmax = x[i];
    if (y[i] < ymin)
      ymin = y[i];
    else if (y[i] > ymax)
      ymax = y[i];
  }
  lim[0] = floor(log10(xmin));
  lim[1] = ceil(log10(ymax));
```

```
  lim[2] = ceil(log10(xmax));
  lim[3] = floor(log10(ymin));
  return (0);
}

int edicao(char alfanum[], double *num, int tipo) {
  size_t n;
  int cursor, estado, x = 6, y = 6, nmax = 39, ymax;
  char tecla[2] = {0, 0};
  time_t t;
  setfillstyle(SOLID_FILL, LIGHTGRAY);
  ymax = getmaxy();
  bar(0, ymax-8, getmaxx(), ymax);
  moveto(4, ymax-7);
  setcolor(RED);
  outtext("Enter");
  setcolor(DARKGRAY);
  outtext("-Entra  ");
  setcolor(RED);
  outtext("Esc");
  setcolor(DARKGRAY);
  outtext("-Anula  ");
  setcolor(RED);
  outtext("BkSpc");
  setcolor(DARKGRAY);
  outtext("-Apaga  ");
  bar(0, 0, 330, 18);
  rectangle(2, 2, 328, 16);
  if (tipo == 1)
    gcvt(*num, 10, alfanum);
  n = strlen(alfanum);
  if (n > 0) {
    setfillstyle(SOLID_FILL, DARKGRAY);
    bar(x, y, x+n*8-1, y+7);
    setcolor(WHITE);
    outtextxy(x, y, alfanum);
  }
  setfillstyle(SOLID_FILL, LIGHTGRAY);
  setcolor(DARKGRAY);
  cursor = 0;
  estado = 1;
  t = clock();
  x += 8*n;
  while (1) {
    if (kbhit() != 0) {
      tecla[0] = getch();
      bar(x+1, y+7, x+6, y+7);
      if (estado == 1) {
       bar(x-8*n, y, x-1, y+7);
       if (isprint(tecla[0])) {
         x -= 8*n;
```

```
          n = 0;
        }
        else
          outtextxy(x-8*n, y, alfanum);
        estado = 0;
       }
       if (isprint(tecla[0]) && n < nmax) {
        alfanum[n] = tecla[0];
        n++;
        outtextxy(x, y, tecla);
        x += 8;
       } else if (tecla[0] == '\b' && n > 0) {
        n--;
        x -= 8;
        bar(x, y, x+7, y+7);
       } else if (tecla[0] == '\r') {
        alfanum[n] = 0;
        if (tipo == 1)
          sscanf(alfanum, "%lg", num);
        return (0);
       } else if (tecla[0] == '\x1B') {
         alfanum[n] = 0;
        return (1);
       } else if (tecla[0] == 0)
        getch();
     }
     if ((clock()-t)/CLK_TCK > 0.15) {
       if(cursor == 0) {
        line(x+1, y+7, x+6, y+7);
        cursor = 1;
       } else {
        bar(x+1, y+7, x+6, y+7);
        cursor = 0;
       }
       t = clock();
     }
   }
}

void main(void) {
  glndata x, y, sig;
  glmma a, a2, percent;
  glncabynca alpha, alpha2, covar, covar2;
  gllista lista;
  static gllista fix, oper;
  double loglimgrf[4], aux, alamda3, chisq2, chisq3;
  static double alamda = -1.0, alamda2 = -1.0, chisq, chisqabs = 1.0e-6,
              chisqrel = 1.0e-3, loglimautgrf[4] = {0.0, 1.0, 1.0, 0.0};
  int graphdriver, graphmode, xmax, ymax, ygrf, ygrfmax, c, cc, ccc, i, j,
      mfit;
  static int linha1max = 28, linha2max = 7, linha1, linha2, mma = 8, nca, iter,
```

```
     iter2, itermax = 100, erro, ndata, ptsig, automat[4] = {1, 1, 1, 1};
char buffer[40];
static char nomarq[40] = "",
           nommod[5][8] = {"", "Theis", "Hantush", "Boulton", "Neuman"},
           sethor[3] = {26, 27, 0},
           tipolim[4][11] = {"a esquerda", "superior", "a direita",
                        "inferior"},
           setver[3] = {24, 25, 0},
           nomparfix[2][4] = {"Q: ", "r: "},
           nomparaju[6][6] = {"T", "S", "r/B", "r/Dt", "beta", "sigma"};
inivar();
for (i = 1; i <= mma; i++) {
  a[i] = 1.0;
  Gamin[i] = 1.0e-100;
  Gamax[i] = 1.0e100;
  percent[i] = 10.0;
}
Gmod = 1;
oper[1] = 1;
oper[2] = 1;
oper[7] = 1;
oper[8] = 1;
fix[7] = 1;
fix[8] = 1;
for (i = 0; i < 4; i++)
  if (automat[i] == 1)
    loglimgrf[i] = loglimautgrf[i];
detectgraph(&graphdriver, &graphmode);
initgraph(&graphdriver, &graphmode, "");
setbkcolor(BLUE);
xmax = getmaxx();
ymax = getmaxy();
ygrfmax = ymax-28-9*linha2max;
grafico(xmax, ygrfmax, loglimgrf, nomarq, x, y, ndata, a, sig, 0);
setcolor(LIGHTGRAY);
rectangle(0, ygrfmax+3, xmax, ymax-11);
do {
  setfillstyle(SOLID_FILL, BLUE);
  bar(3, ygrfmax+6, xmax-3, ymax-14);
  setfillstyle(SOLID_FILL, LIGHTGRAY);
  bar(3, ygrfmax+6+9*linha2, xmax-3, ygrfmax+14+9*linha2);
  for (i = 0; i <= linha2max; i++) {
    if (linha2 == i)
     setcolor(WHITE);
    else
     setcolor(LIGHTGRAY);
    moveto(4, ygrfmax+i*9+7);
    if (linha1+i == 0) {
     outtext("Arquivo: ");
     outtext(nomarq);
    }
```

```
if (linha1+i == 1) {
 outtext("Modelo: ");
 outtext(nommod[Gmod]);
}
if (linha1+i >= 2 && linha1+i <= 5) {
 outtext("Logaritmo do limite ");
 outtext(tipolim[linha1+i-2]);
 outtext(" do grafico");
 if (automat[linha1+i-2] == 1)
   outtext(" (automatico)");
 outtext(": ");
 outtext(gcvt(loglimgrf[linha1+i-2], 10, buffer));
}
if (linha1+i == 6 || linha1+i == 7) {
 outtext("Parametro fixo ");
 outtext(nomparfix[linha1+i-6]);
 outtext(gcvt(a[linha1+i+1], 10, buffer));
}
if (linha1+i >= 8 && linha1+i <= 13) {
 outtext("Parametro ajustavel ");
 outtext(nomparaju[linha1+i-8]);
 if (oper[linha1+i-7] == 0)
   outtext(" (inoperante)");
 else if (fix[linha1+i-7])
   outtext(" (fixado)");
 outtext(": ");
 outtext(gcvt(a[linha1+i-7], 10, buffer));
}
if (linha1+i == 14) {
 outtext("Desvio padrao: ");
 if (ptsig > 0)
   outtext(gcvt(sig[ptsig], 10, buffer));
}
if (linha1+i == 15) {
 outtext("Iteracao automatica max. : ");
 outtext(itoa(itermax, buffer, 10));
}
if (linha1+i == 16) {
 outtext("Max. chi-quadrado p/ converg^ncia: ");
 outtext(gcvt(chisqabs, 10, buffer));
}
if (linha1+i == 17) {
 outtext("Max. var. rel. de chi-quadrado p/ conv. : ");
 outtext(gcvt(chisqrel, 10, buffer));
}
if (linha1+i >= 18 && linha1+i <= 23) {
 outtext("Inc/Dec percentual de ");
 outtext(nomparaju[linha1+i-18]);
 outtext(": ");
 outtext(gcvt(percent[linha1+i-17], 10, buffer));
}
```

```
  if (linha1+i >= 24 && linha1+i <= 29) {
   outtext("Maximo ");
   outtext(nomparaju[linha1+i-24]);
   outtext(": ");
   outtext(gcvt(Gamax[linha1+i-23], 10, buffer));
   }
  if (linha1+i >= 30 && linha1+i <= 35) {
   outtext("Minimo ");
   outtext(nomparaju[linha1+i-30]);
   outtext(": ");
   outtext(gcvt(Gamin[linha1+i-29], 10, buffer));
   }
}
setfillstyle(SOLID_FILL, LIGHTGRAY);
bar(0, ymax-8, xmax, ymax);
moveto(4, ymax-7);
setcolor(RED);
outtext(setver);
setcolor(DARKGRAY);
outtext("-Linha  ");
if (linha1+linha2 == 14) {
  setcolor(RED);
  outtext(sethor);
  setcolor(DARKGRAY);
  outtext("-Seleciona  ");
}
if (linha1+linha2 != 1) {
  setcolor(RED);
  outtext("Enter");
  setcolor(DARKGRAY);
  outtext("-Edita  ");
}
if (linha1+linha2 == 1) {
  setcolor(RED);
  outtext("1        2          3          4         ");
  setcolor(DARKGRAY);
  moverel(-328, 0);
  outtext(" -Theis   -Hantush   -Boulton   -Neuman  ");
}
if (linha1+linha2 > 1 && linha1+linha2 < 6) {
  setcolor(RED);
  outtext("A");
  setcolor(DARKGRAY);
  outtext("-Automatico  ");
}
if (linha1+linha2 > 7 && linha1+linha2 < 14) {
  setcolor(RED);
  outtext("+-");
  setcolor(DARKGRAY);
  outtext("-Inc/Dec  ");
  setcolor(RED);
```

```
    outtext("F");
    setcolor(DARKGRAY);
    outtext("-Fixa  ");
    setcolor(RED);
    outtext("A");
    setcolor(DARKGRAY);
    outtext("-Ajusta  ");
  }
  setcolor(RED);
  outtext("Esc");
  setcolor(DARKGRAY);
  outtext("-Sai  ");
  c = getch();
  if (c == 0) {
    c = getch();
    if (c == 72) {
     if (linha2 > 0)
       linha2--;
     else if (linha1 > 0)
       linha1--;
     if (linha1+linha2 == 13)
       grafico(xmax, ygrfmax, loglimgrf, nomarq, x, y, ndata, a, sig, 0);
    }
    if (c == 80) {
     if (linha2 < linha2max)
       linha2++;
     else if (linha1 < linha1max)
       linha1++;
     if (linha1+linha2 == 15)
       grafico(xmax, ygrfmax, loglimgrf, nomarq, x, y, ndata, a, sig, 0);
    }
    if (linha1+linha2 == 14) {
     if (c == 77)
       if (ptsig < ndata)
         ptsig++;
     if (c == 75)
       if (ptsig > 1)
         ptsig--;
      grafico(xmax, ygrfmax, loglimgrf, nomarq, x, y, ndata, a, sig, ptsig);
    }
  } else {
    if (c == '\r') {
     if (linha1+linha2 == 0) {
       strcpy(buffer, nomarq);
       if (edicao(buffer, &aux, 0) == 0)
         if (arquivo(buffer, x, y, sig, loglimautgrf, &ndata) == 0) {
           strcpy(nomarq, buffer);
           for (i = 0; i < 4; i++)
            if (automat[i] == 1)
              loglimgrf[i] = loglimautgrf[i];
           ptsig = (ndata+1)/2;
```

```
    }
}
if (linha1+linha2 == 2) {
  aux = loglimgrf[0];
  if (edicao(buffer, &aux, 1) == 0 && aux < loglimgrf[2]) {
    loglimgrf[0] = aux;
    automat[0] = 0;
  }
}
if (linha1+linha2 == 3) {
  aux = loglimgrf[1];
  if (edicao(buffer, &aux, 1) == 0 && aux > loglimgrf[3]) {
    loglimgrf[1] = aux;
    automat[1] = 0;
  }
}
if (linha1+linha2 == 4) {
  aux = loglimgrf[2];
  if (edicao(buffer, &aux, 1) == 0 && aux > loglimgrf[0]) {
    loglimgrf[2] = aux;
    automat[2] = 0;
  }
}
if (linha1+linha2 == 5) {
  aux = loglimgrf[3];
  if (edicao(buffer, &aux, 1) == 0 && aux < loglimgrf[1]) {
    loglimgrf[3] = aux;
    automat[3] = 0;
  }
}
if (linha1+linha2 == 6 || linha1+linha2 == 7) {
  aux = a[linha1+linha2+1];
  if (edicao(buffer, &aux, 1) == 0 && aux > 0.0) {
    a[linha1+linha2+1] = aux;
    a[0] = 0.0;
  }
}
if (linha1+linha2 >= 8 && linha1+linha2 <= 13) {
  aux = a[linha1+linha2-7];
  if (edicao(buffer, &aux, 1) == 0) {
    if (aux > Gamax[linha1+linha2-7])
      a[linha1+linha2-7] = Gamax[linha1+linha2-7];
    else if (aux < Gamin[linha1+linha2-7])
      a[linha1+linha2-7] = Gamin[linha1+linha2-7];
    else
      a[linha1+linha2-7] = aux;
    a[0] = 0.0;
  }
}
if (linha1+linha2 == 14) {
  aux = sig[ptsig];
```

```
    if (edicao(buffer, &aux, 1) == 0)
      if (aux > 0.0)
        sig[ptsig] = aux;
  }
  if (linha1+linha2 == 15) {
    aux = itermax;
    if (edicao(buffer, &aux, 1) == 0)
      if (aux >= 1.0)
        itermax = aux;
  }
  if (linha1+linha2 == 16) {
    aux = chisqabs;
    if (edicao(buffer, &aux, 1) == 0)
      if (aux >= 0.0)
        chisqabs = aux;
  }
  if (linha1+linha2 == 17) {
    aux = chisqrel;
    if (edicao(buffer, &aux, 1) == 0)
      if (aux >= 0.0)
        chisqrel = aux;
  }
  if (linha1+linha2 >= 18 && linha1+linha2 <= 23) {
    aux = percent[linha1+linha2-17];
    if (edicao(buffer, &aux, 1) == 0)
      if (aux >= 0.0)
        percent[linha1+linha2-17] = aux;
  }
  if (linha1+linha2 >= 24 && linha1+linha2 <= 29) {
    aux = Gamax[linha1+linha2-23];
    if (edicao(buffer, &aux, 1) == 0)
      if (aux > 0.0 && aux >= Gamin[linha1+linha2-23])
        Gamax[linha1+linha2-23] = aux;
  }
  if (linha1+linha2 >= 30 && linha1+linha2 <= 35) {
    aux = Gamin[linha1+linha2-29];
    if (edicao(buffer, &aux, 1) == 0)
      if (aux > 0.0 && aux <= Gamax[linha1+linha2-29])
        Gamin[linha1+linha2-29] = aux;
  }
}
if (c >= '1' && c <= '4' && linha1+linha2 == 1) {
Gmod = c-'0';
for (i = 3; i <= 6; i++)
  oper[i] = 0;
switch (Gmod) {
  case 2: oper[3] = 1; break;
  case 3: oper[4] = 1; oper[6] = 1; break;
  case 4: oper[5] = 1; oper[6] = 1; break;
}
}
```

```
if (c == 'F' || c == 'f')
 if (linha1+linha2 >= 8 && linha1+linha2 <= 13)
   if (oper[linha1+linha2-7] == 1)
     fix[linha1+linha2-7] = 1-fix[linha1+linha2-7];
if (c == 'A' || c == 'a') {
 if (linha1+linha2 >= 2 && linha1+linha2 <= 5) {
   automat[linha1+linha2-2] = 1;
   loglimgrf[linha1+linha2-2] = loglimautgrf[linha1+linha2-2];
 }
 if (linha1+linha2 >= 8 && linha1+linha2 <= 13) {
   iter = 0;
   alamda = -1.0;
   mfit = 0;
   for (i = 1; i <= mma; i++)
     if (oper[i] == 1 && fix[i] == 0) {
       mfit++;
       lista[mfit] = i;
     }
   j = mfit;
   for (i = 1; i <= mma; i++)
     if (oper[i] == 0 || fix[i] == 1) {
       j++;
       lista[j] = i;
     }
   ygrf = ygrfmax-17-9*mfit;
   while (1) {
     grafico(xmax, ygrf, loglimgrf, nomarq, x, y, ndata, a, sig, 0);
     setfillstyle(SOLID_FILL, BLUE);
     bar(0, ygrf+1, xmax, ygrfmax);
     setcolor(LIGHTGRAY);
     rectangle(0, ygrf+3, xmax, ygrfmax);
     moveto(4, ygrf+7);
     outtext("Iteracao: ");
     outtext(itoa(iter, buffer, 10));
     moveto(148, ygrf+7);
     outtext("Lambda: ");
     outtext(gcvt(alamda, 10, buffer));
     moveto(364, ygrf+7);
     outtext("Chi-quadrado: ");
     outtext(gcvt(chisq, 10, buffer));
     outtextxy(4, ygrf+16, "[C]:");
     for (i = 1; i <= mfit; i++)
       for (j = 1; j <= mfit; j++)
        outtextxy(128*j-84, ygrf+9*i+7, gcvt(covar[i][j], 7, buffer));
     for (i = 0; i <= linha2max; i++)
        if (linha1+i >= 8 && linha1+i <= 13) {
       if (linha2 == i) {
         setfillstyle(SOLID_FILL, LIGHTGRAY);
         setcolor(WHITE);
       } else {
         setfillstyle(SOLID_FILL, BLUE);
```

```
      setcolor(LIGHTGRAY);
    }
    bar(3, ygrfmax+6+9*i, xmax-3, ygrfmax+14+9*i);
    moveto(4, ygrfmax+i*9+7);
    outtext("Parametro ajustavel ");
    outtext(nomparaju[linha1+i-8]);
    if (oper[linha1+i-7] == 0)
      outtext(" (inoperante)");
    else if (fix[linha1+i-7])
      outtext(" (fixado)");
    outtext(": ");
    outtext(gcvt(a[linha1+i-7], 10, buffer));
   }
if (erro > 0) {
  erro = 0;
  setfillstyle(SOLID_FILL, RED);
  bar(0, ymax-8, xmax, ymax);
  moveto(4, ymax-7);
  setcolor(WHITE);
  outtext("Erro no calculo de [C]. ");
  outtext("Pressione ");
  setcolor(YELLOW);
  outtext("Esc");
    do {
   ccc = getch();
   if (ccc == 0)
     getch();
  } while (ccc != '\x1B');
}
  setfillstyle(SOLID_FILL, LIGHTGRAY);
bar(0, ymax-8, xmax, ymax);
moveto(4, ymax-7);
if (iter != 0) {
  setcolor(RED);
  outtext("N");
  setcolor(DARKGRAY);
  outtext("-Novo  ");
}
if (alamda != 0.0 && fabs(alamda) > 1.0e-300 &&
   fabs(alamda) < 1.0e300) {
  setcolor(RED);
  outtext("I");
  setcolor(DARKGRAY);
  outtext("-Iteracao  ");
}
if (alamda != 0.0 && fabs(alamda) > 1.0e-300 &&
   fabs(alamda) < 1.0e300) {
  setcolor(RED);
  outtext("A");
  setcolor(DARKGRAY);
  outtext("-Automatico  ");
```

```
}
 if (iter >= 1 && alamda != 0.0) {
  setcolor(RED);
  outtext("C");
  setcolor(DARKGRAY);
  outtext("-Covariancia  ");
}
if (iter != iter2 || alamda != alamda2) {
  setcolor(RED);
  outtext("R");
  setcolor(DARKGRAY);
  outtext("-Refaz  ");
}
setcolor(RED);
outtext("Esc");
setcolor(DARKGRAY);
outtext("-Retorna  ");
cc = getch();
if (((cc == 'N' || cc == 'n') && iter != 0) ||
   ((cc == 'C' || cc == 'c') && iter >= 1 && alamda != 0.0) ||
   ((cc == 'I' || cc == 'i' || cc == 'A' || cc == 'a') &&
    alamda != 0.0 && fabs(alamda) > 1.0e-300 &&
    fabs(alamda) < 1.0e300)) {
  iter2 = iter;
  alamda2 = alamda;
  chisq2 = chisq;
  glochisq2 = glochisq;
  for(i = 1; i <= mfit; i++) {
   a2[lista[i]] = a[lista[i]];
   glbeta2[i] = glbeta[i];
   for (j = 1; j <= mfit; j++) {
     covar2[i][j] = covar[i][j];
     alpha2[i][j] = alpha[i][j];
   }
  }
  a[0] = 0.0;
}
if ((cc == 'N' || cc == 'n') && iter != 0) {
  iter = 0;
  alamda = -1.0;
}
if ((cc == 'I' || cc == 'i') && alamda != 0.0 &&
   fabs(alamda) > 1.0e-300 && fabs(alamda) < 1.0e300) {
  erro = mrqmin(x, y, sig, ndata, a, mma, lista, mfit, covar,
              alpha, nca,     &chisq, &alamda);
  iter++;
  a[0] = 0.0;
}
if ((cc == 'A' || cc == 'a') && alamda != 0.0 &&
   fabs(alamda) > 1.0e-300 && fabs(alamda) < 1.0e300) {
  do {
```

```
        alamda3 = alamda;
        chisq3 = chisq;
        erro = mrqmin(x, y, sig, ndata, a, mma, lista, mfit, covar,
                      alpha, nca, &chisq, &alamda);
        iter++;
        } while (erro == 0 && iter < itermax && (alamda > alamda3 ||
              (chisq > chisqabs &&
              fabs(chisq-chisq3) > chisqrel*fabs(chisq3))));
        a[0] = 0.0;
      }
       if ((cc == 'C' || cc == 'c') && iter >= 1 && alamda != 0.0) {
        alamda = 0.0;
        erro = mrqmin(x, y, sig, ndata, a, mma, lista, mfit, covar,
                   alpha, nca,     &chisq, &alamda);
        a[0] = 0.0;
      }
      if ((cc == 'R' || cc == 'r') && (iter != iter2 ||
         alamda != alamda2)) {
        iter = iter2;
        alamda = alamda2;
        chisq = chisq2;
        glochisq = glochisq2;
        for(i = 1; i <= mfit; i++) {
         a[lista[i]] = a2[lista[i]];
         glbeta[i] = glbeta2[i];
         for (j = 1; j <= mfit; j++) {
           covar[i][j] = covar2[i][j];
           alpha[i][j] = alpha2[i][j];
         }
        }
        a[0] = 0.0;
      }
      if (cc == '\x1B')
        break;
      if (cc == 0)
        getch();
    }
  }
}
if (c == '+' && linha1+linha2 >= 8 && linha1+linha2 <= 13) {
 aux = a[linha1+linha2-7]*(1.0+percent[linha1+linha2-7]/100.0);
 if (aux > Gamax[linha1+linha2-7])
   a[linha1+linha2-7] = Gamax[linha1+linha2-7];
 else
   a[linha1+linha2-7] = aux;
 a[0] = 0;
}
if (c == '-' && linha1+linha2 >= 8 && linha1+linha2 <= 13) {
 aux = a[linha1+linha2-7]*(1.0-percent[linha1+linha2-7]/100.0);
 if (aux < Gamin[linha1+linha2-7])
   a[linha1+linha2-7] = Gamin[linha1+linha2-7];
```

```
      else
        a[linha1+linha2-7] = aux;
      a[0] = 0;
     }
     if (linha1+linha2 != 14)
      grafico(xmax, ygrfmax, loglimgrf, nomarq, x, y, ndata, a, sig, 0);
     else
       grafico(xmax, ygrfmax, loglimgrf, nomarq, x, y, ndata, a, sig, ptsig);
     if (c == '\x1B') {
      setfillstyle(SOLID_FILL, RED);
      bar(0, ymax-8, xmax, ymax);
      moveto(4, ymax-7);
      setcolor(LIGHTGRAY);
      outtext("Pressione ");
      setcolor(YELLOW);
      outtext("Enter ");
      setcolor(LIGHTGRAY);
      outtext("para confirmar.");
      cc = getch();
      if (cc == '\r')
        break;
      if (cc == 0)
        getch();
     }
   }
} while (1);
closegraph();
```

Anexo B

Este anexo contém as listagens de arquivos demonstrativos, com pontos de rebaixamento versus tempo, para os modelos. O primeiro valor de cada linha relaciona-se com o tempo e o segundo com o rebaixamento no poço de observação. Os arquivos são: theis.dmt, hantush.dmt, boulton.dmt e neuman.dmt.

B.1 - theis.dmt

```
  3  0.3
  5  0.7
  8  1.3
 12  2.1
 20  3.2
 24  3.6
 30  4.1
 38  4.7
 47  5.1
 50  5.3
 60  5.7
 70  6.1
 80  6.3
 90  6.7
100  7.0
130  7.5
160  8.3
200  8.5
260  9.2
320  9.7
380 10.2
500 10.9
```

B.2 - hantush.dmt

```
5 0.76
28 3.30
41 3.59
60 4.08
75 4.39
244 5.47
493 5.96
669 6.11
958 6.27
1129 6.40
1185 6.42
```

B.3 - boulton.dmt

```
1.166 0.08933
1.995 0.2027
2.974 0.315
4.074 0.3752
5.089 0.4269
6.072 0.4606
8.004 0.497
11.79 0.5363
20.13 0.6149
36.86 0.7158
64.91 0.8207
100 0.9128
158.5 1.063
253.1 1.192
393.1 1.287
603 1.466
913.4 1.607
1512 1.83
```

B.4 - neuman.dmt

```
1.17 0.004
1.34 0.009
1.7 0.015
2.5 0.030
4.0 0.047
5.0 0.054
```

6.0 0.061
7.5 0.068
9 0.064
14 0.090
18 0.098
21 0.103
26 0.110
31 0.115
41 0.128
51 0.133
65 0.141
85 0.146
115 0.161
175 0.161
260 0.172
300 0.173
370 0.173
430 0.179
485 0.183
665 0.182
1340 0.200
1490 0.203
1520 0.204

Nomenclatura

Símbolos Alfabéticos

a, parâmetro da transformada de Hankel;

a, vetor dos parâmetros ajustáveis dos modelos;

b, espessura saturada inicial de um aqüífero, L;

$B = \sqrt{T/(P'/m')}$, L;

D_i, inverso do indice de retardo, T^{-1};

$D_t = \sqrt{\frac{T}{D_i S_y}}$, L;

F(t), função representativa de um modelo de aqüífero;

f(w), transformada de Laplace de F(t);

F_a, valor dado pelo algoritmo de Stehfest para a função F(t);

H^{-1}, operador da inversão da transformada de Hankel;

I_0, função modificada de Bessel de ordem zero e primeira espécie;

I_1, função modificada de Bessel de ordem um e primeira espécie;

INT, função que retorna a parte inteira do argumento;

J_0, função de Bessel de ordem zero e primeira espécie;

K_0, função modificada de Bessel de ordem zero e segunda espécie;

K_1, função modificada de Bessel de ordem um e segunda espécie;

K_r, permeabilidade horizontal, LT^{-1};

K_z, permeabilidade vertical, LT^{-1};

K_D, permeabilidade adimensional, igual a K_z/K_r;

L^{-1}, operador da inversão da transformada de Laplace;

m', espessura de um aquitard, L;

MIN, função que retorna o menor de dois argumentos;

N, parâmetro do algoritmo de Stehfest associado à precisão da máquina;

p, parâmetro da transformada de Laplace (o mesmo que w);

P', permeabilidade de um aquitard, LT^{-1};
Q, vazão de bombeamento, L^3T^{-1};
r, distância radial do poço de bombeamento, L;
s, rebaixamento, L;
$\bar{s}$, transformada de Laplace de s;
s*, transformada de Hankel de $\bar{s}$;
S, coeficiente de armazenabilidade, igual a $S_s b$;
S_s, armazenabilidade específica (elástica), L^{-1};
S_y, produção específica;
t, tempo de bombeamento;
T, transmissividade, igual a $K_r b$, L^2T^{-1};
z, distância vertical acima do fundo do aqüífero;
V_i, coeficiente do algoritmo de Stehfest;
w, parâmetro da transformada de Laplace (o mesmo que p).

Símbolos Gregos

$[\alpha]$, matriz de curvatura modificada por λ;
$[\alpha]^{-1}$, inversa da matriz de curvatura modificada por λ;
$\alpha_s = K_r/S_s$, L^2T^{-1};
$\alpha_y = K_z/S_y$, LT^{-1};
$\beta = K_d r^2/b^2$;
χ^2, parâmetro estatístico do ajuste do modelo (chi-quadrado);
$\eta = \sqrt{\left(a^2/K_D\right)+\left(p/\alpha_s K_D\right)}$;
λ, parâmetro do algoritmo de Levemberg-Marquardt;
$\sigma = S/S_y$.

Bibliografia

Referências Bibliográficas

CUNHA, C.: Métodos numéricos para as engenharias e ciências aplicadas, Campinas - SP, Editora da UNICAMP, 1993.

DOGRU, A. N., DIXON, T. N. & EDGAR, T. F.: "Confidence limits on the parameters and predictions of slightly compressible, single-phase reservoirs", Soc. Pet. Eng. J., p. 42-56, feb 1977.

FETTER, C. W.: Applied Hydrogeology, 3. ed., New York, Macmillan Publishing Company, 1994.

KRUSEMAN, G. P. & RIDDER, N. A. de: Analysis and evaluation of pumping test data, International Institute for Land Reclamation and Improvement/ILRI, Wageningen, The Netherlands, 1990.

NEUMAN, S. P.: "Theory of flow in unconfined aquifers considering delayed response of the water table", Water Resourse Research, Vol. 8, No. 4, p. 1031-1045, 1972.

NEUMAN, S. P.: "Supplementary Comments on 'Theory of flow in unconfined aquifers considering delayed response of the water table'", Water Resourse Research, Vol. 9, No. 4, p. 1102-1103, 1973.

ORELLANA, E. N. & CORRÊA, A. C. F.: "Automation of pressure transient test in laboratory using deconvolution", SPE paper 23702, 1992.

PRESS, W. H., FLANNERY, B. P., TEUKOLSKY, S. A. & VETTERLING, W. T.: Numerical recipes: the art of scientific computing, New York, Cambridge Univ. Press, 1989.

ROSA, A. J. & HORNE, R. N.: "Automated type-curve matching in well test analysis using Laplace space determination of parameters gradients", SPE paper 12131, 1983.

STEHFEST, H.: "Algorithm 368 numerical inversion of Laplace transforms" D - 5, Communications of TH ACM. Vol. 13, No. 1, p.47 - 49, jan 1970.

SUGAHARA, L. A. N. & CORRÊA, A. C.: "Análise automatizada de testes de bombeamento", Campinas - SP, dissertação de mestrado - Universidade Estadual de Campinas, 1996.

WALTON, W. C.: Groundwater resource evaluation, Mc Graw-Hill Book Company, 1970.

Obras Consultadas

ABRAMOWITZ, M. & STEGUN, I. A.: Handbook of mathematical functions, Washington D. C. , National Bureau of Standards, 1965.

BARD, Y.: Nonlinear parameter estimation, New York, Academic Press, inc. Ltd., 1974.

BASTOS, A. M., LIMA FILHO, A. S., NERY, F., MANNHEIMER, P. H. & FARIAS, A. S.: Linguagem C: programação e aplicações / Módulo Consultoria e Informática, 3. ed., Rio de Janeiro, LTC - Livros Técnicos e Científicos Editora Ltda., 1989.

FREEZE, R. A. & CHERRY, J. A.: Groundwater, Prentice-Hall Inc., 1979.

GILL, P., MURRAY, W. & WRIGHT, M.: Practical optimization, Academic Press, 1981.

JONHSON, E. E.: Água subterrâneas e poços tubulares, Tradução de "Groundwater & Wells", Jonhson Division - Universal Oil Products Company, editado pela CETESB.

NEUMAN, S. P.: "Analysis of pumping test data from anisotropic unconfined aquifers considering delayed gravity response", Water Resourse Research, Vol. 11, No. 2, p. 329-342, 1975.

TODD, D. K.: Hidrologia de águas subterrâneas, Ed. Edgard Blücher Ltda., 1959.

WYLIE, C. R. & BARRETT, L. C.: Advanced engineering mathematics, McGraw - Hill Book Co - Singapore manufacture and export, 1985.

Apêndice

Neste apêndice, são analisados quatro programas desenvolvidos para a análise de testes em aqüíferos. São eles: THCVFIT versão 1.0, AQTESOLV versão 1.10, AQUIX 1-2-3 versão 1.5, e AQUATEST. Estes programas foram "mentores" de idéias utilizadas no ITBA.

1 THCVFIT versão 1.0

Este programa é bem elementar quando comparado com os demais apresentados a seguir. Foi desenvolvido em 1987 por Paul Van der Heijde, do International Ground Water Modeling Center - Holcomb Research Institute - Butler University - Indianapolis - Indiana - USA. O programa utiliza um único modelo para representar o relacionamento entre o rebaixamento em um poço de observação e o tempo. Trata-se do modelo de Theis, apresentado por WALTON (1970), que subentende um aqüífero confinado, homogêneo e isotrópico, bombeado por um outro poço a uma vazão constante, entre outras condições.

O THCVFIT gera um gráfico contendo uma curva representativa de modelo, e pontos relativos as medições do poço de observação. Os dados que localizam estes pontos podem ser fornecidos via teclado, ou por meio de um arquivo externo com uma tabela. Pode-se optar entre dois conjuntos de unidades de trabalho: o de unidades métricas, e o de unidades inglesas. É possível fazer movimento de translação com o conjunto de pontos de modo que se localizem mais próximos da curva do modelo. Esta movimentação é feita com a ajuda das setas do teclado, que a partir de um ponto de dados central geram uma linha, cuja extremidade indica a posição para onde se deslocará o conjunto dos pontos ao se pressionar a tecla F8. Correspondendo ao

ajuste realizado o programa calcula e apresenta os valores de dois parâmetros do modelo: a transmissividade (T), e o coeficiente de armazenabilidade (S) do aqüífero.

Como características adicionais do programa podemos citar: foi desenvolvido em BASIC, opera em ambiente DOS, possui definição e interface gráfica pobres, é simples de aprender, não realiza ajuste automático dos parâmetros do modelo (regressão não linear), não edita os dados de entrada já fornecidos, e a interação com o usuário é deficiente no sentido de facilitar a tarefa do mesmo.

2 AQTESOLV versão 1.10

O programa AQTESOLV versão 1.10, da Geraghty & Miller, Inc. - Reston - Virginia é o que possui maior número de modelos entre os apresentados neste apêndice. Para testes em aqüíferos confinados dispõe dos métodos padrões de Theis e Cooper - Jacob, e Papadopulos - Cooper (poço de bombeamento ou observação) para poços de grande diâmetro. Para testes em aqüíferos semi-confinados dispõe dos métodos padrões de Hantush, com ou sem armazenagem nos aquitards, e Moench (poço de bombeamento ou observação) para poços de grande diâmetro. Para testes em aqüíferos não confinados dispõe do método de Neuman, ou dos métodos de Theis e Cooper - Jacob com correção de Jacob para redução na espessura saturada. Para testes de recuperação dispõe do método de Theis para aqüíferos confinados. Para testes com piezômetros (slug tests) dispõe dos métodos de Bouwer e Rice para aqüíferos não confinados, ou do método de Cooper, Bredehoeft, e Papadopulos para aqüíferos confinados. Para testes em aqüíferos fraturados dispõe do método de Moench para poços de bombeamento e observação. Além do mais, o programa permite trabalhar com fronteiras de recarga ou barreira utilizando poço imagem, e trabalhar com penetração parcial utilizando a correção de Hantush para piezômetros ou poços de observação.

O programa analisado era demonstrativo e acompanhava a publicação FETTER (1994). Por ser demonstrativo, o pacote analisado implementava somente o método de Theis para aqüíferos confinados, o método de Hantush, sem armazenamento no aquitard, para aqüífero semi-confinado, e o método de Bouwer e Rice para testes com piezômetros (slug tests). Também, ele não permitia trabalhar com fronteiras ou com penetração parcial.

No AQTESOLV, a determinação dos parâmetros dos modelos pode ser feita automaticamente. O programa utiliza a técnica de mínimos quadrados não lineares de Marquardt. Neste processo podemos especificar o número máximo de iterações, os valores iniciais dos parâmetros, os limites superiores e inferiores dos intervalos onde os valores dos parâmetros devem permanecer, e tornar constantes os parâmetros que quisermos. Ao final, podemos visualizar diversos resultados: os parâmetros estimados juntamente com o erro padrão de cada um, a matriz de correlação, estatísticas dos resíduos (número de observações do teste, número de parâmetros do modelo, graus de liberdade, média dos resíduos, variância dos resíduos, e erro padrão dos resíduos), além do gráfico dos pontos do teste juntamente com a curva do modelo ajustado.

O programa também possibilita que o ajuste do modelo seja realizado pelo usuário de forma visual, ou gráfica. Assim, há teclas que permitem o incremento ou decremento de determinado percentual (ajustável entre 1 e 10 %) dos parâmetros do modelo, ou a entrada via teclado de novos valores. Sempre que um parâmetro é modificado a curva do modelo se altera, enquanto os pontos do teste permanecem fixos. É possível, ainda, dependendo do modelo, efetuar movimentos de translação de sua curva no gráfico, ou retraçar, a partir de dois pontos que podem ser localizados no gráfico, a reta que representa o modelo, sendo estas operações realizadas com o uso das setas do teclado. O passo de deslocamento nestas operações com as setas pode ser ajustado. A cada alteração feita na posição da curva ou reta do modelo os parâmetros são imediatamente recalculados.

Como recursos e características adicionais do programa temos que: as medições observadas nos poços podem ser cadastradas pelo teclado ou serem lidas de um arquivo, e o conjunto de dados que está sendo utilizado pode ser modificado (edição).

3 AQUIX 1-2-3 versão 1.5

Será visto agora o programa AQUIX 1-2-3 versão 1.5, de janeiro de 1991, da Interpex Limited - Golden - Colorado - USA. Este programa é voltado para a análise de teste de poços, mas a empresa também possui outro, o SLUGIX, projetado para interpretar testes com piezômetros (slug test).

O AQUIX 1-2-3 opera com quatro modelos: Theis (1935) para aqüífero confinado, Hantush (1960) para aqüífero confinado com infiltração ou vazamento (semi-confinado), Hantush (1964) para aqüífero confinado com infiltração ou vazamento com anisotropia entre a direção vertical e o plano horizontal, e Neuman (1975) para aqüífero não confinado com o mesmo tipo de anisotropia anterior.

Para cada um dos modelos mencionados o programa permite especificar, na entrada de dados, o tipo de teste (rebaixamento constante, vazão constante ou vazão variável), o tipo de aqüífero (confinado ou não confinado), o tipo de poço (bombeamento, observação ou piezômetro), e se o teste é normal ou de recuperação. A opção teste de recuperação não é possível quando se seleciona teste de vazão variável. Com a continuação do processo de entrada de dados aparece outra tela, a qual solicita informações gerais sobre o teste, o poço e o aqüífero, conforme as especificações dadas na tela anterior. No caso de teste de recuperação é pedido nesta etapa a vazão e o tempo do bombeamento. Na mesma tela o programa solicita ainda as profundidades que especificam o início e o fim da abertura de filtro dos poços: bombeamento, ou bombeamento e observação, conforme o caso. Com isto poderia se fazer uma conjectura: a de

que o programa possa introduzir em qualquer dos quatro modelos correções devido a penetração parcial.

Prosseguindo com a entrada de dados uma última tela é apresentada. Esta tela é para os pontos do teste de bombeamento. Ela possui normalmente duas colunas: uma para o tempo e a outra para o rebaixamento, no caso de vazão constante, ou para tempo e vazão no caso de rebaixamento constante. No caso de vazão variável teremos três colunas: tempo, rebaixamento e vazão. As unidades dos dados de entrada são selecionadas com muita flexibilidade. Assim, diversas grandezas, como rebaixamento, raio do poço, transmissividade, vazão ou tempo, dispõe cada qual de um conjunto próprio de unidades para seleção. Esta seleção de unidades é feita logo na primeira tela dos procedimentos de entrada de dados.

A apresentação gráfica da curva do modelo e dos pontos do teste é de muito boa qualidade, podendo ser impressa. Diversas características desta apresentação podem ser configuradas pelo usuário. Deste modo ele pode selecionar escala linear ou logarítmica para os eixos x e y, a direção (sentido) do eixo y, e o escalonamento dos eixos, se automático ou personalizado.

O programa perfaz a inversão (estimativa dos parâmetros) automática do modelo selecionado iteração por iteração, ou efetuando um número determinável de iterações. A cada etapa ele apresenta o erro, os novos valores dos parâmetros, e a nova curva do modelo. Se desejado alguns parâmetros do modelo selecionado podem ser mantidos fixos durante a análise. Também é possível mascarar (e desmascarar) com praticidade pontos do teste, de forma que estes não influenciem na análise. Ao final da inversão, para uma margem de erro ajustável, o programa determina a faixa de variação de cada parâmetro. O programa, pode apresentar, também, alguns conjuntos de valores equivalentes dos parâmetros ajustados, com seus respectivos erros, e a matriz de resolução.

É possível, também, fazer um ajuste visual dos parâmetros do modelo. Neste caso os valores dos parâmetros são alterados entrando-se com novos valores, o que faz a curva do modelo ser modificada de acordo.

Uma característica adicional do programa é que os comandos podem ser acionados de duas formas: por meio de menus, ou por meio de linhas de comando.

Finalmente, um esclarecimento sobre os testes de vazão variável: No gráfico para interpretação do teste, as curvas dos modelos e os pontos retratam rebaixamento versus tempo, sem apresentar o comportamento do parâmetro vazão de bombeamento.

4 AQUATEST

Os comentários das seções anteriores foram elaborados unicamente a partir da análise dos respectivos programas computacionais. O conteúdo desta seção, ao contrário, foi desenvolvido unicamente a partir da dissertação de mestrado, SUGAHARA (1996), que deu origem ao programa em questão. Não se trata do mesmo programa AQUATEST comentado no Capítulo 1, e que serviu de partida para o programa ITBA. Na realidade, o programa AQUATEST do Capítulo 1 serviu também de base para o desenvolvimento do programa AQUATEST desta seção.

Este programa foi desenvolvido em ambiente Windows, com a ferramenta Visual BASIC, e opera basicamente com três modelos de aqüíferos: Theis, Hantush (com aquitard incompressível) e Boulton. Este último destinado a aqüífero livre. Com estes modelos é possível analisar testes de bombeamento a vazão constante, incluindo, se desejado, o período de recuperação após o fechamento do poço. Neste caso, aos pontos do período de recuperação é ajustado um Spline cúbico, que servirá para gerar pontos, utilizando-se o princípio da

superposição, que extrapolem a tendência do período de bombeamento, prosseguindo-se a análise como um teste de bombeamento simples.

O programa também possibilita, a partir da análise de dados de dois períodos de produção consecutivos (com vazões constantes, distintas e maiores que zero), a realização de teste de desempenho de poço, e geração da curva característica. Aqui considera-se que o reservatório tenha um comportamento semelhante ao modelo de Theis, porém com o poço de bombeamento, que se confunde com o de observação, possuindo um raio finito e perda de carga localizada, onde menospreza-se o efeito da estocagem.

Em qualquer tipo de teste o programa permite que se realize um ajuste visual dos parâmetros, que servirá como aproximação inicial para um ajuste automático a ser realizado em seguida. Este ajuste automático emprega o método de Gauss-Newton para o modelo de Theis, e o de Levemberg-Marquardt para os demais. Com a realização do ajuste automático o programa mostra os valores ajustados e as faixas de variação de cada parâmetro, onde ele se encontra com uma probabilidade igual a 95 %. Esta análise estatística foi implementada segundo o método de DOGRU, DIXON & EDGARD (1977). Além do mais, o programa realiza busca restrita dos parâmetros, que melhora os aspectos de convergência, onde em cada iteração, se algum dos parâmetros atingir um valor fora de um intervalo significativo, a ele é atribuído a média aritmética do valor da iteração anterior e o limite extrapolado.

Como características adicionais, a dissertação de Sugahara não deixa claro, mas sugere que o programa por ele desenvolvido também é capaz de editar o arquivo de dados, e selecionar um sistema de unidades de trabalho.