21º COLÓQUIO BRASILEIRO DE MATEMÁTICA

EQUAÇÕES DIFERENCIAIS EM MODELAGEM MATEMÁTICA COMPUTACIONAL

ANDRÉ NACHBIN ESTEBAN TABAK

IMPA 21 - 25 JULHO, 1997

impa

ANDRÉ NACHBIN **(IMPA/RJ)**
ESTEBAN TABAK **(Courant Institute of Mathematics Sciences - New York/USA)**

ISBN 85-244-0127-3

Conselho Nacional de Desenvolvimento Científico e Tecnológico

INSTITUTO DE MATEMÁTICA PURA E APLICADA

Estrada Dona Castorina, 110 - Jardim Botânico

22460-320 - Rio de Janeiro, RJ, Brasil

Prefácio

Este mini-curso foi preparado com o intuito de introduzir o aluno de graduação à Modelagem Matemática e Computacional. O enfoque principal é dado a questões de modelagem (tanto física, quanto matemática e numérica). O uso do computador é feito como um "laboratório matemático". Hoje em dia, percebe-se uma dificuldade generalizada, universal (i.e., nas melhores universidades, tanto nacionais como no exterior), do aluno em conseguir transformar um problema apresentado em palavras (uma estória contendo fatos a serem levados em conta) em um problema matemático (com expressões). Essa dificuldade justamente se faz na passagem do modelo físico para o modelo matemático. Na introdução, fazemos uma analogia com o processo de tradução entre duas linguagens: no caso, a do português e a da matemática. Exercícios dessa passagem são realizados no capítulo de aplicações. Os modelos matemáticos usados neste livro serão equações diferenciais ordinárias e parciais (edo e edp). Conceitos básicos serão introduzidos e depois verificados através de experimentos computacionais. Não será esperado do aluno nenhum conhecimento prévio de edo's ou de edp's. Na verdade, este mini-curso tem a visão moderna segundo a qual, se pode/deve aprender e motivar a teoria recorrendo-se a situações práticas, e também fazendo uso do computador.

Parte dessas notas surgiram a partir de dois cursos de verão no IMPA e de um mini-curso no XIX CNMAC em Goiânia de 1996 (ministrados por A. Nachbin).

Gostaríamos de agradecer ao Prof. Martin Tygel e ao Prof. Marco Antônio Raupp pelo convite para ministrar esse mini-curso. Também gostaríamos de agradecer à gentileza de Beata Gundelach, Mônica Souza e Prof. Dan Marchesin pela revisão do texto.

Índice

Capítulo 1

Introdução

1.1 Visão geral do assunto

1.1.1 O problema e o tradutor simultâneo

Os problemas em Matemática Aplicada, em particular os ligados à modelagem, são em geral apresentados em palavras, sem fórmulas. Isto ocorre principalmente quando o problema é originado na indústria. Muitas vezes a estória do problema é contada por um especialista de outra área. Com palavras, essa pessoa nos conta o que está querendo investigar, que tipos de respostas busca, que idéias tem para encontrá-las, e também de extrema importância, quais são os **ingredientes relevantes** do problema. Isto significa fazer/escolher as hipóteses físicas a serem levadas em conta. Vale observar que quando falamos da física do problema, nos referimos também aos ingredientes químicos, ou biológicos, etc, que podem, ou não, ser levados em conta. Por exemplo: em Dinâmica dos Fluidos, podemos desprezar efeitos térmicos (de troca de calor) durante o escoamento de um fluido em um canal aberto. O mesmo pode aplicar-se a efeitos viscosos ... O matemático aplicado que se interessa por modelagem deve se acostumar a fazer o papel de um tradutor simultâneo, realizando a tradução do problema em palavras (**em português**) para o problema na linguagem matemática (**"em matematiquês"** $\Leftrightarrow$ expressão em questão) e vice-versa. Por exemplo, quando dizemos, em português,

que *o crescimento da população de insetos é proporcional à população dos mesmos*, estamos dizendo, em matematiquês, que $dN/dt = \lambda N$, onde $\lambda > 0$ é a constante de proporcionalidade. Se a população $N(t)$ dobrar, o crescimento da população é duas vezes mais rápido.

Na verdade, qualquer matemático deve fazer o exercício de tradução simultânea, tradução esta que depende do contexto. A Matemática é uma linguagem que nos permite expressar e operar com grandezas. Esta linguagem é universal, no sentido que quando escrevemos $f(x) = x$, qualquer pessoa "matematizada" do planeta sabe o que isso quer dizer. A leitura mais precisa vai depender do contexto. Podemos traduzir $f(x) = x$ como *pontos que não vão a lugar nenhum (pontos fixos)*, ou como *uma reta inclinada a 45 graus passando pela origem* ou até como *uma máquina xerox de números*, entre outros. A escolha depende do contexto. A expressão $dN/dt = \lambda N$ pode muito bem ser traduzida como *a velocidade de uma partícula aumenta linearmente conforme ela se afasta da origem.* Nesse caso, $N(t)$ representa a posição de uma partícula. Esse exercício de tradução é muito saudável, pois conforme vamos sentindo mais facilidade no processo de tradução (em geral), estamos melhorando a nossa intuição quanto ao problema.

A partir das escolhas quanto ao modelo físico, obtemos um modelo matemático, que na maioria das vezes é uma equação diferencial. Nesta monografia, vamos estudar métodos para a solução numérica de equações diferenciais. No processo de escolha de um método, vamos recair em um modelo numérico, ou seja, uma versão discreta da equação diferencial. Se o método adotado for o de Diferenças Finitas, obteremos como modelo numérico uma equação de diferenças.

1.1.2 O modelo matemático

O que significa *o modelo matemático*? Vamos usar um exemplo de Dinâmica dos Fluidos. No enunciado do problema, inicialmente fazemos hipóteses sobre a física

considerada. Isso já foi comentado acima. Por exemplo, quando estudamos ondas do mar (também chamadas de ondas de gravidade, devido ao efeito da gravidade na oscilação da superfície livre) podemos supor que a água é um fluido incompressível e invíscido (ao contrário de um óleo, que é pegajoso, i.e. viscoso). Isto simplifica as leis da Física envolvidas e conseqüentemente também o modelo matemático adotado (que neste caso são equações diferenciais parciais (edp's)).

O "pulo-do-gato", que exige experiência, é saber como simplificar um modelo o suficiente para que possamos resolvê-lo com um esforço razoavelmente pequeno, mas ao mesmo tempo deixá-lo "rico" o bastante para que fenômenos de interesse possam ser simulados através do modelo final. Note que um modelo "pobre" é aquele que não serve para simular nada além do trivial. É nesse ponto que se torna importante a colaboração com especialistas de outras áreas (por exemplo, dependendo do problema, um físico ou químico ou biólogo etc ...) para interagir na questão da modelagem e saber o que devemos "buscar" e qual a nossa meta final.

Mesmo os modelos mais simples em Dinâmica dos Fluidos são complexos a ponto de tornar-se praticamente impossível a obtenção de soluções analíticas. Por *solução analítica* entende-se uma expressão que descreve a solução do problema. Uma técnica analítica pode nos levar a uma solução exata ou aproximada. Esta última se verifica quando usamos, por exemplo, análise assintótica ou expansão em série (se a truncarmos).

Nos casos mais complexos, a única saída é usar o computador para encontrar uma solução aproximada. Usamos acima a palavra *modelo* tanto no sentido de *modelo físico* como de *modelo matemático*. O primeiro está relacionado com as hipóteses adotadas para a física do problema. O segundo com o objeto matemático que usamos para estudar o modelo físico. O objeto matemático pode ser uma equação diferencial, um sistema de equações não-lineares, uma equação integral,

e assim por diante. Por exemplo, podemos formular o problema de Dirichlet (na teoria do potencial) usando a equação de Laplace (edp) ou usando uma equação integral de contorno. Trta-se de dois modelos matemáticos equivalentes, utilizados na resolução do mesmo probelma. Em geral o modelo matemático será atacado (resolvido) usando-se um método numérico (i.e., um modelo numérico). Vamos discutir questões relativas a modelos numéricos mais adiante. Antes disso, vamos entender o que significa *validar* o modelo físico, o modelo matemático e o modelo numérico.

1.1.3 Validação

Essa é uma das fases mais importantes de um projeto em Matemática Computacional! Existem, em geral, três modelos a serem validados: o modelo físico, o modelo matemático e o modelo numérico. É nessa ordem que fomos construindo o nosso projeto através dos modelos. No entanto, na fase de validação, procedemos na ordem contrária. É muito importante a validação (testagem) do modelo numérico, para se ter mais confiança quanto ao código implementado. A fase de validação, do ponto de vista físico e matemático, é usada para verificar se o modelo, ao menos, representa fenômenos já conhecidos, para então partirmos para novas descobertas.

Na validação numérica devemos escolher, inicialmente, problemas bem simples, que servem apenas para depurar o programa e ver se ele executa corretamente até o final. A seguir, passamos para testes computacionais um pouco mais complexos, mas para os quais conhecemos os resultados. Isso testará não só a parte numérica, mas também o modelo matemático. Em outras palavras, queremos ver se resultados teóricos, relativos à parte matemática, são observados numericamente. Isto aumentará a nossa confiança com respeito à capacidade "exploratória" do código escrito. Afinal de contas, para o cientista computacional o código é como uma

sonda que vai navegando por um espaço desconhecido e que está equipada para enviar informações interessantes sobre esse espaço jamais navegado. Essas informações interessantes são as grandezas que decidimos plotar e visualizar através de um pacote gráfico. A partir delas, vamos adquirir uma intuição mais apurada desse novo "território explorado" e possivelmente fazer descobertas de cunho teórico ou prático.

Existem duas fontes de erros (provenientes das aproximações adotadas) em um projeto de Matemática Computacional:

(1): Erros na física do problema. Esses erros estão ligados às hipóteses simplificadoras. Por exemplo, que a água é um fluido incompressível.

(2): Erros no método de solução numérica do problema.

(2a): Erro de arredondamento.

(2b): Erro de discretização e/ou truncamento.

Vale ressaltar que o avanço tecnológico dos computadores tem permitido o uso de modelos cada vez mais sofisticados, o que permite simular uma gama maior de fenômenos. Os modelos mais recentes são bem mais complexos e exigem um esforço computacional maior. Um exemplo típico é o caso da meteorologia e da previsão numérica do tempo. Com o uso de supercomputadores foi possível se "colocar mais física" nos modelos (por exemplo levar em conta o efeito de montanhas ou de vegetação, etc ...) e ainda assim resolver a previsão do tempo para 24 horas em aproximadamente 6 horas de computação. Há alguns anos, o mesmo modelo levaria mais de 24 horas para ser executado nos computadores de então. Saber hoje a previsão do tempo de ontem não faz sentido, a não ser do ponto de vista acadêmico, ou seja, para validar o modelo em questão.

1.2 Introdução à solução numérica de equações diferenciais

Façamos uma breve introdução à solução numérica de equações diferenciais ordinárias (edo's). Um Problema de Valor Inicial (PVI) é escrito da seguinte maneira:

$$y' \equiv \frac{dy}{dx} = f(y, x), \quad y(x_0) = y_0.$$

De agora em diante, vamos usar tanto x como t para a variável independente. Vamos estudar métodos baseados em séries de Taylor (Diferenças Finitas) para resolver PVI's no computador. Consideremos inicialmente o método mais simples de todos: o Método de Euler.

Vamos definir a notação a ser adotada: $Y(x)$ é a solução exata de edo, enquanto que y_n denota a solução aproximada em $x = x_n = n\Delta x$. Nosso objetivo é calcular valores discretos (aproximações) da solução $Y(x)$ até o ponto $x_N = b$, usando N passos de tamanho Δx. O método é derivado a partir da série de Taylor:

$$Y(x_{n+1}) = Y(x_n) + \Delta x Y'(x_n) + O(\Delta x^2),$$

onde fazemos uso da edo para substituir

$$Y'(x_n) \text{ por } f(Y(x_n), x_n).$$

Até esse ponto nenhuma aproximação foi feita. O termo "ordem delta x ao quadrado" ($O(\Delta x^2)$) é uma abreviação para o erro cometido ao aproximarmos $Y(x)$ pelo polinômio de Taylor de grau 1. Nesse caso $O(\Delta x^2) = 0,5\ Y''(\xi)(x_{n+1} - x_n)^2$, onde $x_n < \xi < x_{n+1}$. Ignorando os termos de ordem mais alta (i.e., $O(\Delta x^2)$) chegamos à equação de diferenças conhecida como o método de Euler explícito ou progressivo (a série de Taylor centrada em x_n estima o valor de Y no ponto seguinte x_{n+1})

$$\boxed{y_{n+1} = y_n + \Delta x f(y_n, x_n)}$$

Note que é a solução aproximada y (e não mais a exata Y) que satisfaz esta equação de diferenças. A solução aproximada não é uma função e sim o vetor $[y_0 y_1 ... y_n ... y_N]^T$. A nossa meta é obter $y_n \approx Y(x_n)$, com um erro razoável.

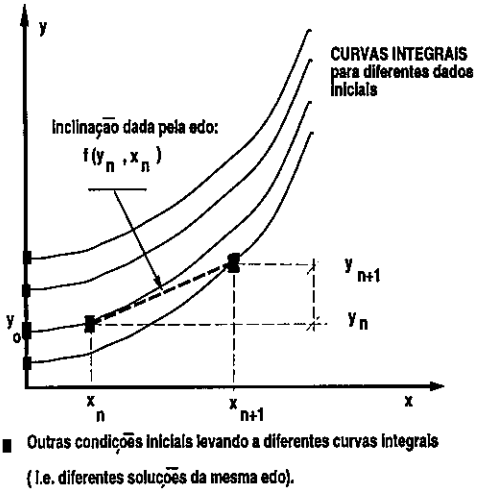

Figura 1: Interpretação geométrica do método de Euler.

Do ponto de vista geométrico, está acontecendo o seguinte: temos a reta $y = b + \Delta x(dy/dx)$ escrita na forma

$$y_{n+1} = y_n + \Delta x f(y_n, x_n)$$

ou ainda

$$\frac{y_{n+1} - y_n}{\Delta x} = f(y_n, x_n).$$

A solução numérica, pelo método de Euler, vai mudando de uma curva integral para outra ao longo de segmentos de reta. No entanto, para Δx's suficientemente pequenos, a solução numérica (interpretada graficamente como uma poligonal) estará próxima da solução exata.

O método de Euler também poderia ter sido formulado através da versão integral da edo:

$$Y(x_{n+1}) = Y(x_n) + \int_{x_n}^{x_{n+1}} f(Y(x), x)\, dx,$$

e usando

$$\int_{x_n}^{x_{n+1}} f(Y(x), x)\, dx \approx \Delta x f(Y(x_n), x_n)$$

(a regra do retângulo, pela esquerda) temos o esquema numérico desejado.

O Método de Euler é um método de primeira ordem ou de ordem 1 (ordem do erro global $= O(\Delta x)$). Em outras palavras, se dividirmos o passo Δx por 2, o erro global é reduzido aproximadamente à metade. É fácil verificar que o método de Euler é de primeira ordem. Note que o erro de truncamento local é $O(\Delta x^2)$. Esse é o termo desprezado na série de Taylor dada acima. Suponhamos que o intervalo de integração (no qual resolveremos a edo) vá de x_0 a x_N, contendo assim N passos de tamanho Δx. O erro local será acumulado N-vezes. Consequentemente, o erro global é dado por $N\ O(\Delta x^2) = (x_N - x_0)/\Delta x\ \ O(\Delta x^2) = O(\Delta x)$.

Vamos agora procurar dar ao menos uma idéia de como escolher o passo Δx (ou Δt). Vejamos um exemplo que ilustra a questão da instabilidade numérica de soluções de edo's.

Exemplo 1: Seja o PVI (usemos como variável independente t)

$$\frac{dy}{dt} = -y, \quad y(0) = y_0,$$

onde a solução exata é dada por

$$Y(t) = y_0 e^{-t}.$$

Substituindo no método de Euler (com $h = \Delta t$) temos

$$y_{n+1} = y_n + hf(y_n, t_n) = y_n - hy_n = (1-h)y_n$$

ou ainda

$$\boxed{y_{n+1} = (1-h)^{n+1} y_0}.$$

Suponhamos que $h = 3$. Isto nos dá

$$y_{n+1} = (1-3)y_n = -2y_n,$$

e conseqüentemente

$$\begin{aligned} y_1 &= -2y_0 \\ y_2 &= 4y_0 \\ y_3 &= -8y_0 \\ y_4 &= 16y_0 \\ &\cdots \end{aligned} \tag{1.1}$$

Obtivemos uma solução oscilatória que **cresce indefinidamente!** Como a solução exata decai, isto é um sinal de instabilidade numérica. É de extrema importância sabermos os valores de h que não geram instabilidades. Sempre podemos usar h's muito pequenos. Mas isso pode ser impraticável devido ao custo computacional!

Tendo em mente o problema de estabilidade numérica, e consequentemente o intervalo de "h"-permissíveis (que nos levará à noção de região de estabilidade absoluta), vejamos o exemplo seguinte:

Exemplo 2: Vamos definir o "problema-teste": $y' \equiv dy/dt = \lambda y$, $y(0) = y_0$ e λ complexo com parte real negativa. A solução desse problema é simples: $Y(t) = y_0 \exp(\lambda t)$. Escrevendo $\lambda = -\alpha + i\ \theta$ temos que $Y(t) = y_0 \exp(-\alpha t)(\cos(\theta t) + i\ \sin(\theta t))$. A solução exata decai no tempo com taxa α. Logo, qualquer que seja o método numérico que usemos, ele deverá produzir uma solução que não cresça no tempo. Caso seja observado crescimento, saberemos que se trata de uma instabilidade numérica, ou seja, espúria. Dentro dessa filosofia, o problema-teste deve ser encarado como um campo de provas para o método numérico estudado. Da mesma maneira que um carro é testado inicialmente em uma pista oval, simples, sem outros carros. Esse tipo de teste é feito para se ter uma idéia inicial quanto à estabilidade do carro nas curvas, etc ... Note que o problema-teste é propositadamente construído de maneira que sua solução exata decaia exponencialmente. Por isso fazemos a exigência de que λ tenha parte real negativa. Dentro desse contexto,

se a solução numérica crescer, claramente é o método numérico que está (artificialmente) produzindo esse crescimento. Entendendo o mecanismo por trás desse crescimento artificial, podemos achar condições de estabilidade, e assim evitar que o método "derrape".

Usemos o critério do problema-teste para o método de Euler progressivo. É evidente que $y_n = (1 + \lambda h)^n y_0$. Chamemos $(1 + \lambda h)$ de fator de amplificação para o método de Euler. Aqui abusamos um pouco do português e dizemos que taxa de amplificação menor do que 1 é taxa de decaimento. Sabemos que $Y(t) \to 0$ quando $t \to \infty$ e por isso y_n deve decair no intervalo estudado. Mas isso *só* será verdade se

$$\boxed{|1 + h\lambda| < 1}.$$

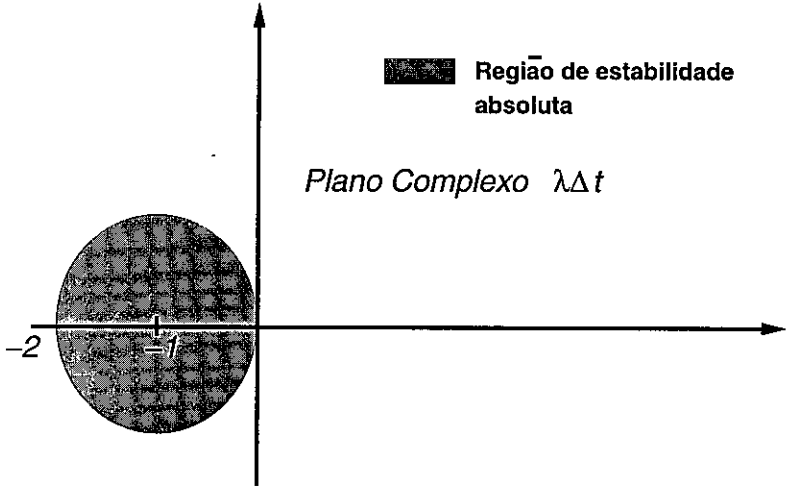

Figura 2: Região de estabilidade absoluta para o método de Euler progressivo (explícito).

Ao definirmos o número complexo $z \equiv h\lambda$, fica natural a definição da *região de estabilidade absoluta* $\mathcal{A}$ como sendo o disco unitário (aberto) centrado em $z_0 = -1$! Em outras palavras quando $z = h\lambda$ estiver na região $|z-(-1)| < 1$ (o disco unitário;

ver figura 2) o método de Euler progressivo será estável.

Podemos fazer uma aplicação do problema-teste para sistemas da forma $y' = Ay$, onde A é uma matriz $m \times m$, constante, com m autovalores distintos tendo parte real negativa. Suponhamos que a matriz A possa ser diagonalizada na forma $A = MDM^{-1}$, onde D é a matriz diagonal com os autovalores e M a matriz que contém os autovetores, correspondentes a cada autovalor, organizados por colunas. Substituindo A escrevemos

$$\frac{d(M^{-1}y)}{dt} = D\ (M^{-1}y)$$

para obter o sistema totalmente desacoplado

$$\frac{dw}{dt} = D\ w,$$

com $w = M^{-1}y$. Temos agora m problemas-teste! Devemos escolher o passo no tempo Δt de maneira a satisfazer a condição de estabilidade para o autovalor de maior valor absoluto. Uma vez conhecido o espectro (autovalores $\lambda_1, \lambda_2, ..., \lambda_m$) da matriz A, que denotamos por $\sigma(A)$ (supondo parte real$(\sigma(A)) < 0$), podemos calcular o espaçamento a ser adotado.

Queremos frisar que a noção de região de estabilidade absoluta está associada ao problema-teste. Métodos diferentes irão gerar regiões distintas. Isto pode ser constatado nos exemplos a seguir. Ao escolhermos um método para solução numérica do problema-teste, a região de estabilidade absoluta indicará o passo máximo de integração $h_{\max}$. Para sistemas, do tipo $y' = Ay$ dado acima, a condição de estabilidade é $|h\lambda_i + 1| < 1$, para $i = 1, ..., m$.

Exemplo 3: Analisemos o "problema-teste" $y' = \lambda y$, $y(0) = y_0$, usando o método de Euler implícito (também chamado de retroativo, pois expandimos a série de Taylor com um passo negativo):

$$Y(t_n) = Y(t_{n+1}) - \Delta x Y'(t_{n+1}) + O(\Delta t^2),$$

Isto nos leva ao método de Euler retroativo:

$$\boxed{y_{n+1} = y_n + \Delta t f(y_{n+1}, t_{n+1})}.$$

Note que neste método não podemos calcular diretamente o valor de y_{n+1}, pois ele aparece também como argumento do campo vetorial f. Por isso, este método é do tipo implícito. Será necessário usar um algoritmo, do tipo método de Newton, para achar zeros de uma função. Mas deixemos essa questão de lado e voltemos ao nosso "campo de provas".

Se o aplicarmos no problema-teste obtemos

$$y_{n+1} = y_n + \lambda h y_{n+1}.$$

Podemos reescrever esta relação como

$$y_{n+1} = \frac{1}{1 - h\lambda}\, y_n = \left(\frac{1}{1 - h\lambda}\right)^{n+1} y_0.$$

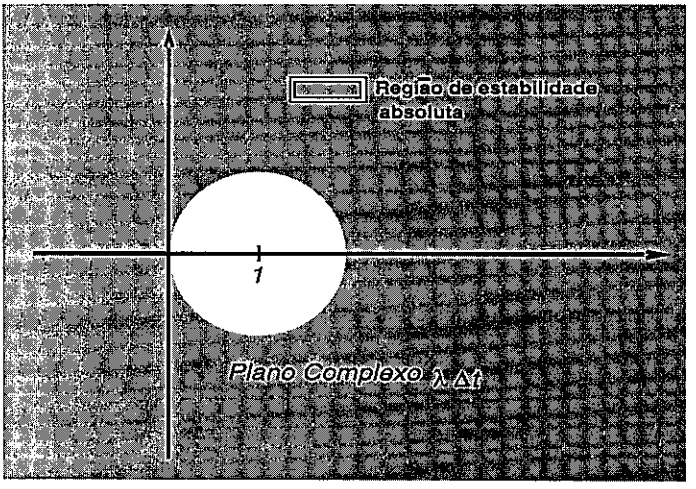

Figura 3: Região de estabilidade absoluta para o método de Euler retroativo (implícito).

A condição de estabilidade é dada por $|h\lambda - 1| > 1$, que é a região externa ao disco unitário centrado em $z = 1$. Isto significa que o esquema numérico será

estável para qualquer valor de λ com parte real negativa. O método implícito de Euler (retroativo) é *incondicionalmente estável.* Veja a região de estabilidade na figura 3.

Exemplo 4: Estudemos o "problema-teste" $y' = \lambda y$, $y(0) = y_0$, usando a regra do trapézio:

$$\boxed{y(t_{n+1}) = y(t_n) + 0,5\ \Delta t\,(f(y(t_{n+1}), t_{n+1}) + f(y(t_n), t_n))}.$$

Substituindo no problema-teste, obtemos

$$y_{n+1} = y_n + \frac{\lambda h}{2}(y_{n+1} + y_n).$$

Podemos reescrever esta relação como

$$y_{n+1} = \left(\frac{1 + 0,5h\lambda}{1 - 0,5h\lambda}\right)^{n+1} y_0.$$

Fazendo uso de análise complexa (teoria de mapeamento conforme), é fácil mostrar que a região de estabilidade absoluta corresponde ao semi-plano esquerdo (ver figura 4).

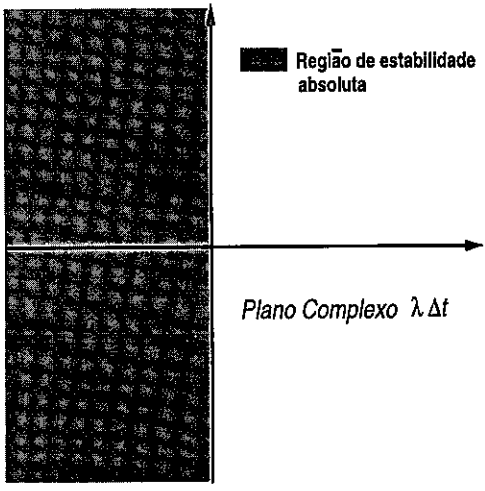

Figura 4: Região de estabilidade absoluta para a regra do trapézio.

Exemplo 5: Vamos usar um problema específico para verificar computacionalmente as propriedades inferidas pelas regiões de estabilidade absoluta. Seja o seguinte sistema de edo's

$$\frac{d}{dt}\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} \varepsilon & \delta \\ -\delta & \varepsilon \end{bmatrix}\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} \tag{1.2}$$

É fácil verificar que os autovalores da matriz do sistema acima são os números complexos (conjugados) $\lambda^{\pm} = \varepsilon \pm i\delta$. Note que o ε nos dá a taxa de crescimento (ou decaimento, quando negativo) da solução, enquanto δ é a frequência das oscilações. Brincando com os valores atribuídos a ε e δ podemos passear por todo o plano complexo (λh) e ver o que acontece quando estamos dentro ou fora da região de estabilidade absoluta do método adotado.

Lembre-se que no problema-teste, por definição, temos autovalores com a parte real negativa (aqui, $\varepsilon < 0$). O que faremos agora é explorar situações mais gerais do que as permitidas pelo problema-teste. A solução geral (exata) do problema proposto é da forma

$$\begin{bmatrix} Y_1(t) \\ Y_2(t) \end{bmatrix} = \begin{bmatrix} y_1^0/(m_{11}+m_{12})\,(m_{11}\exp(\lambda^+ t) + m_{12}\exp(\lambda^- t)) \\ y_2^0/(m_{11}+m_{12})\,(m_{21}\exp(\lambda^+ t) + m_{22}\exp(\lambda^- t)) \end{bmatrix},$$

onde os m_{ij} são os coeficientes da matriz M, com os autovetores organizados em colunas. Os dois autovalores estão presentes nas duas componentes do vetor-solução e por isso é *importantíssimo* satisfazer a condição de estabilidade absoluta para todos os autovalores. Façamos a escolha $(\varepsilon = 0)$ que nos fornece uma solução que apenas oscila no tempo, já que $\exp(it) = \cos(t) + i\ \sin(t)$. Em outras palavras, no nosso passeio pelo plano complexo $h\lambda$, estamos situados sobre o eixo imaginário (ver a figura 6).

O método de Euler explícito nos dá resultados instáveis, quaisquer que sejam os h escolhidos. Correto? Não há como *entrarmos* na região de estabilidade absoluta, qualquer que seja o valor de h que escolhamos. Veja o exemplo da figura 6 onde,

ao refinarmos a discretização (passando de Δt para $\Delta t/2$), o quadrado baixa ao longo do eixo imaginário, mas mesmo assim não entra na região de estabilidade absoluta.

Verifiquemos o que foi dito acima usando o programa EuE.m com os seguintes dados: $\varepsilon = 0$, $\delta = 1$, $y_1^0 = y_2^0 = 1$, com o tempo inicial $t_i = 0$, tempo final $t_f = 30$ e $nt = 300$ passos no tempo. O resultado é dado na figura 5.

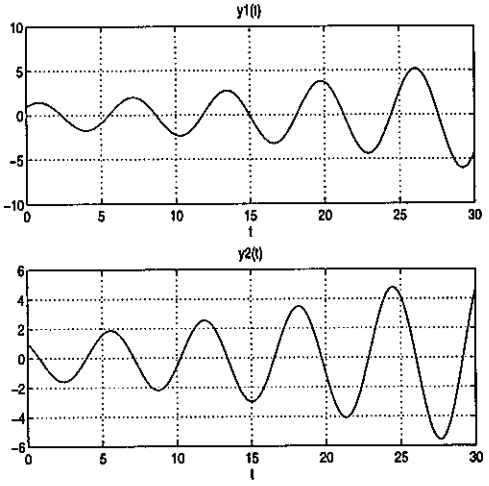

Figura 5: Método de Euler explícito com $y_1^0 = y_2^0 = 1$, com o tempo inicial $t_i = 0$, tempo final $t_f = 50$ $nt = 500$ passos no tempo.
Os autovalores da matriz do sistema são $\lambda = \pm i$.

Por outro lado, com o método de Euler implícito o eixo imaginário está dentro da região cinza (figura 4), região esta onde a solução não cresce pois o fator de amplificação é menor do que 1. Façamos um experimento com o programa EuI.m (Euler implícito; veja o apêndice), usando os dados $y_1^0 = y_2^0 = 1$, com o tempo inicial $t_i = 0$, tempo final $t_f = 50$ e $nt = 500$ passos no tempo. O resultado é dado na figura 7.

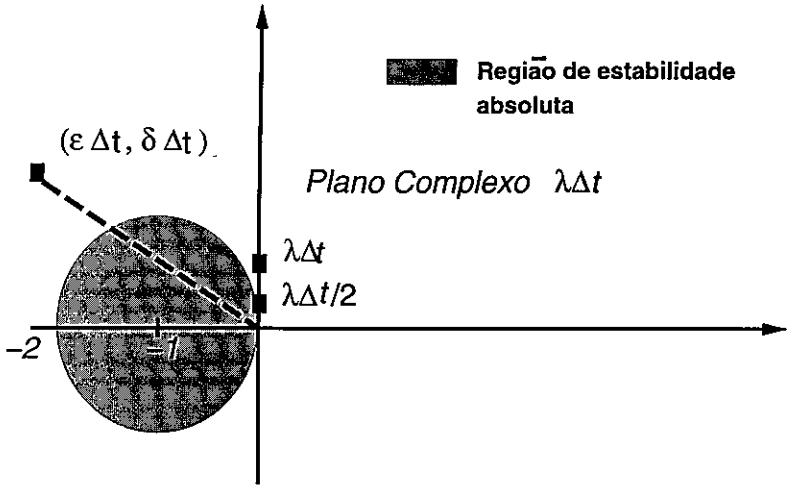

Figura 6: Região de estabilidade absoluta para o método de Euler explícito.

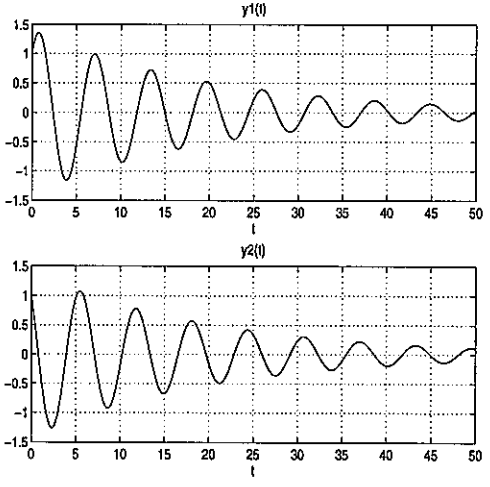

Figura 7: Método de Euler implícito com $y_1^0 = y_2^0 = 1$, com o tempo inicial $t_i = 0$, tempo final $t_f = 50$ $nt = 500$ passos no tempo.

Os autovalores da matriz do sistema são $\lambda = \pm i$.

Algo inesperado está acontecendo! A solução está decaindo, quando o esperado era que apenas oscilasse. Este *mecanismo dissipativo* é de cunho puramente

numérico. Ele não está presente no modelo original $dy/dt = Ay$ que escolhemos! Por isso dizemos que é um *fenômeno espúrio.* Em analogia com sistemas massa–mola, dizemos que temos um mecanismo de amortecimento numérico. Há situações em dinâmica dos fluidos computacional, análogas a essa, onde o mecanismo de decaimento numérico é chamado de viscosidade numérica. Esse nome se deve ao fato de o modelo físico não ter viscosidade, mas o modelo numérico se comportar como se esse ingrediente físico estivesse presente.

A explicação para a presença desse mecanismo foi dada logo acima: o fator de amplificação para o método de Euler implícito, neste exemplo, é menor do que 1. Amplificação menor do que 1 é na verdade abuso de linguagem! Por isso temos em vez de amplificação, decaimento da solução. Convença-se de que não temos como fazer o fator de amplificação (do Euler retroativo) deixar de ser menor do que 1, no caso de autovalores puramente imaginários.

Para enfatizar ainda mais esta questão de decaimento espúrio, vamos escolher um outro exemplo onde a solução exata cresce exponencialmente. Para tal, considere o sistema com taxa de crescimento $\varepsilon = 1$ e frequência $\delta = 10$. Os autovalores são $\lambda = 1 \pm i\ 10$. Novamente usemos o programa EuI.m com os dados $y_1^0 = y_2^0 = 1$, $t_i = 0$, $t_f = 5$ e $nt = 50$. O resultado é mostrado na figura 8.

De novo obtivemos uma solução numérica que apresenta um mecanismo de decaimento espúrio. Nesse caso, no entanto, podemos "acertar os ponteiros". Em outras palavras, podemos diminuir o passo no tempo h de maneira a entrarmos dentro do círculo branco da figura 9, e com isso fazer com que a solução numérica cresça, conforme o esperado. Note que o disco branco representa a região onde o fator de amplificação é maior do que 1. Nesse caso, o crescimento é desejável, já que os autovalores têm parte real positiva. Faça, por conta própria, o exercício de ir entrando no disco branco, usando valores cada vez menores para o h.

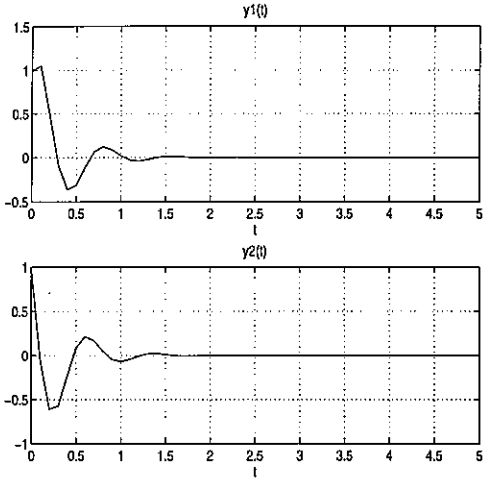

Figura 8: Método de Euler implícito com $y_1^0 = y_2^0 = 1$, com o tempo inicial $t_i = 0$, tempo final $t_f = 5$ e $nt = 50$ passos no tempo. Os autovalores da matriz do sistema são $\lambda = 1 \pm i\ 10$.

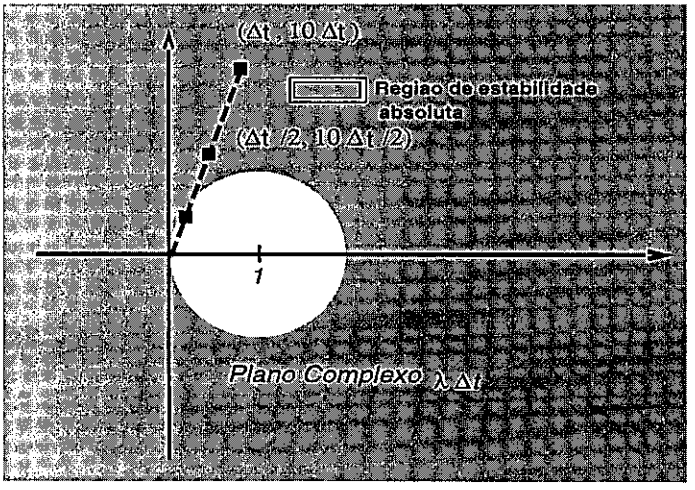

Figura 9: Região de estabilidade absoluta para o método de Euler implícito.

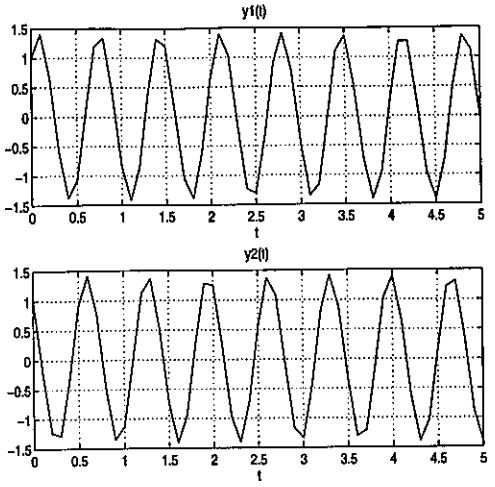

Figura 10: Regra do Trapézio (implícito) com $y_1^0 = y_2^0 = 1$, com o tempo inicial $t_i = 0$, tempo final $t_f = 5$, $nt = 50$ passos no tempo.

Os autovalores da matriz do sistema são $\lambda = \pm i\ 10$.

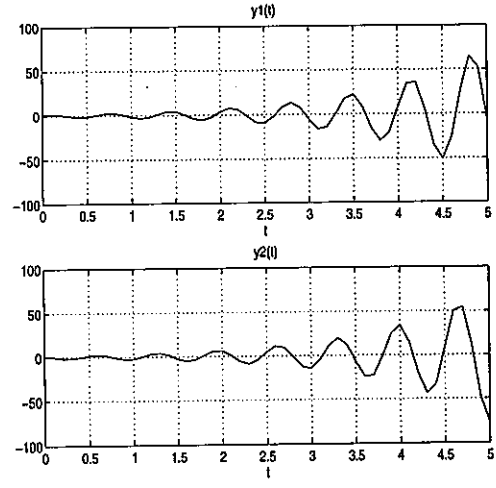

Figura 11: Regra do Trapézio (implícito) com $y_1^0 = y_2^0 = 1$, com o tempo inicial $t_i = 0$, tempo final $t_f = 5$ e $nt = 50$ passos no tempo.

Os autovalores da matriz do sistema são $\lambda = 1 \pm i\ 10$.

Vejamos agora o que acontece com a regra do trapézio. O que devemos esperar? Como a região de estabilidade absoluta é o semi-plano esquerdo, sabemos que a solução numérica vai decair quando os autovalores tiverem parte real negativa e vai crescer quando a parte real for positiva. Vamos verificar usando o programa Trap.m! Usemos os dois exemplos já utilizados para testar o Euler implícito. Todos os dados são idênticos. Os resultados são apresentados nas figuras 10 e 11.

Não há nenhuma indicação da presença de fenômenos espúrios.

Agora deixemos nossa curiosidade matemática bem solta e vamos entender como cada método estudado acima modela a exponencial. Em que sentido estamos fazendo essa pergunta? Vamos por partes ... O problema em questão é a edo $dy/dt = \lambda y$, onde λ é um número complexo qualquer. A solução exata é $Y(t) = y_0 \exp(\lambda t)$. Podemos escrever que $Y(n\Delta t) = y_0\,[\exp(\lambda \Delta t)]^n$, de onde concluímos que o fator de amplificação de cada método é, na verdade, uma aproximação para $\exp(\lambda \Delta t)$. Foi isso que chamamos de "método modelando a exponencial". Temos os três casos seguintes:

método de Euler explícito:

$$\exp(\lambda \Delta t) \approx 1 + \Delta t \lambda$$

método de Euler implícito:

$$\exp(\lambda \Delta t) \approx \frac{1}{1 - \Delta t \lambda}$$

método da regra do trapézio:

$$\exp(\lambda \Delta t) \approx \frac{1 + 0,5 \Delta t \lambda}{1 - 0,5 \Delta t \lambda}$$

No método de Euler explícito, a exponencial é aproximada por um polinômio linear de Taylor, ou seja, pela série de Taylor truncada no segundo termo. No método de

Euler implícito e no método da regra do trapézio, são usados elementos da aproximação de Padé. Façamos uma breve introdução a esta teoria. Os detalhes podem ser encontrados em diversos livros de métodos numéricos e teoria da aproximação. Como referência sugerimos o Burden & Faires.

A teoria da aproximação de Padé diz respeito a aproximações que usam funções racionais da forma

$$r(x) = \frac{p(x)}{q(x)},$$

onde $p(x)$ é um polinômio de grau n, e $q(x)$ um polinômio de grau m. Este tipo de aproximação é vantajoso, por exemplo com respeito à polinomial, quando queremos aproximar funções com descontinuidades infinitas localizadas fora do intervalo de interesse. Chamemos a nossa função de interesse de $f(x)$. A meta é parecida com a do polinômio de Taylor: queremos que todas as derivadas, até uma determinada ordem, coincidam com as mesmas derivadas de $f(x)$ em um determinado ponto. Por simplicidade, vamos escolher esse ponto como sendo a origem $x = 0$. As condições a serem impostas são:

$$f^{(k)}(0) = r^{(k)}(0), \quad k = 0, 1, ..., N, \tag{1.3}$$

onde (k) indica a ordem da derivada. Logo, a função $(f(x) - r(x))$ tem uma raiz (i.e. um zero) de ordem $N + 1$ em $x = 0$. Isso é o mesmo que dizer que a função $h(x) \equiv (f(x)q(x) - p(x))$ tem uma raiz de ordem $N + 1$ em $x = 0$. Usando a série de Taylor para $f(x)$ em torno do zero, e escrevendo os dois polinômios por extenso, obtemos

$$f(x)q(x) - p(x) = \sum_{i=0}^{\infty} a_i x^i \sum_{j=0}^{m} q_j x^j - \sum_{i=0}^{n} p_i x^i. \tag{1.4}$$

Neste ponto estamos supondo que conhecemos os coeficientes a_i da função $f(x)$. Agora necessitamos calcular os coeficientes q_j e p_i de forma a satisfazer a condição (1.3). Satisfazer (1.3) é sinônimo de poder reescrever (1.4) na forma

$$h(x) = x^{N+1} \sum_{i=0}^{\infty} \alpha_i x^i,$$

indicando explicitamente a existência de uma raiz de ordem $N+1$ em $x=0$. Concluímos então que a série de Taylor da função $h(x)$ tem os primeiros $N+1$ coeficientes identicamente nulos. Essa é a condição que nos fornece os coeficientes dos polinômios $p(x)$ e $q(x)$, como veremos abaixo nos dois exemplos provenientes da solução numérica de edo's.

Façamos a aproximação de Padé para e^x, com $n=0$ e $m=1$. O polinômio $q(x)$ é linear e $p(x)$ é uma constante. Precisamos calcular um total de 2 coeficientes: p_0 e q_1. O coeficiente q_0 é sempre tomado igual a 1. Dessa maneira $p(0)=f(0)$. Precisamos de 2 condições, ou seja, $h(x)$ deve ter um zero de ordem 2. O primeiro termo não-nulo da sua série de Taylor tem que ser o x^2. Com isso escrevemos

$$h(x)=(1+x+\frac{x^2}{2}+...)(1+q_1x)-p_0,$$

de onde concluímos que

$$1-p_0=0$$

e que

$$1+q_1=0.$$

O termo truncado é $O(x^2)$. A aproximação de Padé nos dá

$$e^x\approx r(x)=\frac{1}{1-x}.$$

Substituindo $x=\lambda\Delta t$ obtemos a expressão do método de Euler retroativo.

Seja agora $m=n=1$. Temos que

$$h(x)=(1+x+\frac{x^2}{2}+\frac{x^3}{6}+...)(1+q_1x)-(p_0+p_1x).$$

A condição da raiz ser de ordem 3 nos leva a

$$(1-p_0)+x(1+q_1-p_1)+x^2(1/2+q_1)=0.$$

Calculamos os coeficientes de $r(x)$ a partir da identidade acima para obter

$$e^x\approx r(x)=\frac{1+x/2}{1-x/2}.$$

Note que a discretização para a regra do trapézio é de ordem mais alta. Na aproximação de Padé truncamos a partir do termo x^3.

Acabamos de ver acima como um problema tão simples quanto o problema-teste nos permite ver de forma cristalina várias propriedades de um método numérico. Vimos questões de estabilidade. Generalizando para casos onde λ é um número complexo qualquer, observamos mecanismos espúrios de amortecimento e também a forma com que o método em questão captura a solução exata, que tem a forma de uma exponencial. O uso de um problema simples para testar e validar um método numérico está dentro do melhor espírito de matemática computacional. A partir desse ponto podemos ir, gradativamente, iniciando nossa jornada para explorar um problema mais difícil e interessante.

Exemplo 6: Consideremos os dois sistemas abaixo (Lambert).

$$\frac{d}{dt}\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ 1 & -2 \end{bmatrix}\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} + \begin{bmatrix} 2\sin(t) \\ 2(\cos(t) - \sin(t)) \end{bmatrix}, \tag{1.5}$$

com

$$\begin{bmatrix} y_1(0) \\ y_2(0) \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

e

$$\frac{d}{dt}\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ 998 & -999 \end{bmatrix}\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} + \begin{bmatrix} 2\sin(t) \\ 998(\cos(t) - \sin(t)) \end{bmatrix}, \tag{1.6}$$

com

$$\begin{bmatrix} y_1(0) \\ y_2(0) \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}.$$

Os dois sistemas tem a mesma solução:

$$\begin{bmatrix} Y_1(t) \\ Y_2(t) \end{bmatrix} = 2e^{-t}\begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix}. \tag{1.7}$$

No entanto resolver numericamente o segundo sistema dá muito mais "dor de cabeça" do que o primeiro! O leitor consegue identificar porque? Vamos entender isso de forma bem clara.

Cada sistema deve ser resolvido usando a troca de variáveis $w = M^{-1}y$ indicada anteriormente. Vejamos um exemplo em detalhe. Os autovalores e autovetores da matriz

$$A = \begin{bmatrix} -2 & 1 \\ 1 & -2 \end{bmatrix}$$

são usados para se resolver o sistema homogênio, ou seja, sem o termo mais da direita que só depende do tempo e independe de y. Resolvendo a equação caracterítisca $det(A - \lambda I) = 0$ obtemos $\lambda_1 = -1$ e $\lambda_2 = -3$. Os respectivos autovetores (que são soluções do sistema $Ax = \lambda x$) compõem as colunas da matriz M, da troca de variáveis:

$$M = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

A sua inversa é dada por

$$M^{-1} = \begin{bmatrix} 1/2 & 1/2 \\ 1/2 & -1/2 \end{bmatrix}.$$

Efetuando-se a troca de variáveis, temos

$$\frac{d}{dt}\begin{bmatrix} w_1(t) \\ w_2(t) \end{bmatrix} = \begin{bmatrix} -w_1(t) & 0 \\ 0 & -3w_2(t) \end{bmatrix} + \begin{bmatrix} 1/2 & 1/2 \\ 1/2 & -1/2 \end{bmatrix}\begin{bmatrix} 2\sin(t) \\ 2(\cos(t) - \sin(t)) \end{bmatrix}. \tag{1.8}$$

A solução homogênea $w^h(t)$ é trivialmente obtida. Temos $w_i^h(t) = \exp(\lambda_i t)$ com $i = 1, 2$. A solução geral é uma combinação linear com a solução particular.

Para resolver o sistema não-homogênio usamos a técnica de "coeficientes a determinar" para cada uma das equações desacopladas. Isto consiste em dizer que cada solução particular $w_{1,2}^p(t)$ é da forma $w_{1,2}^p(t) = A\sin(t) + B\cos(t)$, onde A e B são os coeficientes a determinar. O valor desses coeficientes são obtidos facilmente após substituirmos $w_{1,2}^p$ nas suas respectivas equações. Ao final, desfazemos a troca de variáveis antes de impor as condições iniciais.

A solução geral do primeiro sistema é

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \alpha_1 e^{-t}\begin{bmatrix} 1 \\ 1 \end{bmatrix} + \alpha_2 e^{-3t}\begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix}, \tag{1.9}$$

enquanto que a do segundo sistema é dada por

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \beta_1 e^{-t} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \beta_2 e^{-1000t} \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix}. \qquad (1.10)$$

As contantes α_1, α_2, β_1 e β_2 são escolhidas de maneira a satisfazer as condições iniciais. Resulta que $\alpha_1 = 2$, $\alpha_2 = 0$, $\beta_1 = 2$ e $\beta_2 = 0$. Para estas condições iniciais ocorre uma coincidência: as duas segundas constantes, em cada sistema, são zero. Os segundos autovalores e autovetores não participam da dinâmica neste caso específico! Do ponto de vista de modelagem dizemos que as condições iniciais eram tais que apenas uma escala de tempo foi acionada: a escala do decaimento com taxa igual a 1. As outras taxas de decaimento (que podemos pensar como uma segunda escala de tempo do problema) não estão presentes nestas soluções. Por exemplo, na solução geral do segundo sistema, temos uma escala de tempo lenta e outra rápida. A rápida é a associada ao autovalor –1000. Mesmo quando presente na solução, esta componente irá decair rapidamente e efetivamente sumir da solução. Após um pequeno intervalo de tempo a sua contribuição para a solução será desprezível!

No entanto essa escala de tempo rápida cria dificuldades para a solução numérica do segundo sistema. Agora, a nossa experiência prévia com o problema-teste será muito útil!!! Sabemos que temos de escolher um Δt de maneira que, tanto $-\Delta t$ como $-1000\Delta t$ estejam dentro da região de estabilidade absoluta, do método numérico adotado. A notícia desagradável é que temos que escolher um Δt bem pequeno (para acomodar o autovalor –1000), mesmo sabendo que esta componente da solução geral não faz parte da dinâmica em questão. O método ficará muito lento, já que avançaremos no tempo usando um passo bem pequeno. Em situações onde temos escalas discrepantes de decaimento no tempo, chamamos a equação de rígida. Ela é rígida no sentido de (em geral) nos *forçar* a usar um passo bem pequeno, simplesmente por questões de estabilidade e não por causa de precisão.

Vamos verificar esses fatos através de exemplos computacionais. Consideremos os métodos de Euler progressivo e o retroativo. O desempenho do método implícito certamente será melhor. Por que? Adapte os programas EuE.m e EuI.m para os exemplos acima. Considere um sistema na forma

$$\frac{d}{dt}\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} + \begin{bmatrix} g_1(t) \\ g_2(t) \end{bmatrix}.$$

Para adaptar os programas indicados acima, vale lembrar que o algoritmo para o Euler explícito deve ser escrito a partir da expressão matricial

$$\vec{y}^{\,(n+1)} = (I + \Delta t A)\vec{y}^{\,(n)} + \Delta t \vec{g}^{\,(n)}.$$

Para o método de Euler implícito temos

$$\vec{y}^{\,(n+1)} = (I - \Delta t A)^{-1}(\vec{y}^{\,(n)} + \Delta t \vec{g}^{\,(n+1)}).$$

Implemente os novos programas e compare com os resultados apresentados nas figuras 12, 13 e 14.

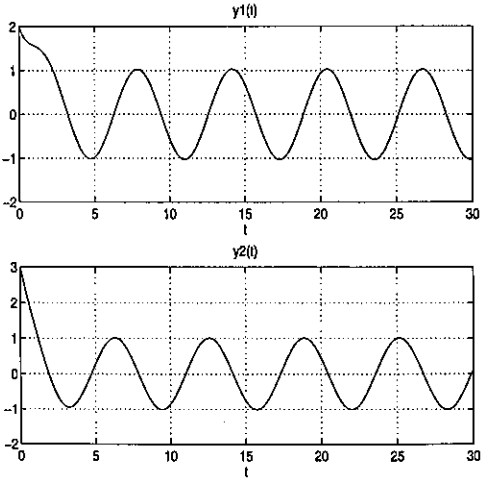

Figura 12: Solução da equação (1.5) via o método de Euler explícito com $t_0 = 0$ e $\Delta t = 0,1$.

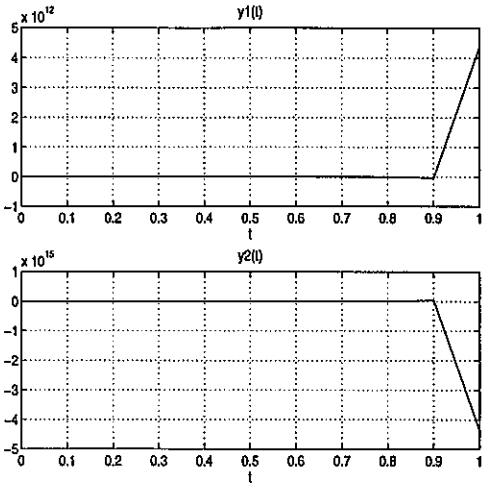

Figura 13: Solução da equação (1.6) via o método de Euler explícito com $t_0 = 0$ e $\Delta t = 0,1$.

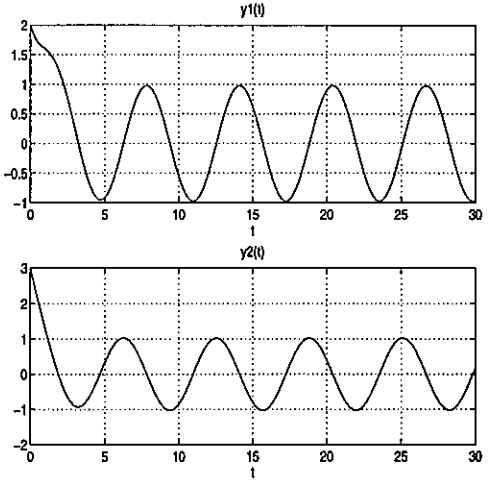

Figura 14: Solução da equação (1.6) via o método de Euler implícito com $t_0 = 0$ e $\Delta t = 0,1$.

Um ótimo exercício é ir diminuindo o Δt, no método explícito, até desaparecer a instabilidade numérica.

Capítulo 2

Modelagem e aplicações

2.1 Modelo para a dinâmica de populações

Vamos começar exercitando o processo de tradução **português** $\rightarrow$ **matematiquês**. Este modelo matemático é usado para simular diversos tipos de populações, variando de micro-organismos a animais competindo no contexto presa–predador. Como exemplo ilustrativo do processo de modelagem, vamos começar com o caso mais simples possível. Seja $N(t)$ a população de uma dada espécie em um dado instante do tempo t. O modelo mais simples que podemos imaginar é aquele no qual o crescimento (ou decaimento) da população é proporcional ao número de membros da mesma. Matematicamente isso se escreve na forma da equação diferencial ordinária (edo)

$$\frac{dN}{dt}(t) = rN(t), \quad N(0) = N_0, \tag{2.1}$$

onde dN/dt é a variação da população com respeito ao tempo, a constante N_0 é a população inicial e r é a taxa de variação. Se $r > 0$ (taxa de natalidade) a população cresce, enquanto que se $r < 0$ (taxa de mortalidade) a população decresce. No entanto, este modelo tem um problema sério. No caso de crescimento ($r > 0$) a população pode crescer indefinidamente. Note que o crescimento é exponencial. Como é que sabemos que o crescimento é exponencial e não algébrico (ou seja, quadrático ou cúbico ...)? Esse modelo está longe de ser realista pois

sabemos que, após um certo período de tempo, a população será tão grande que surgirão problemas de limitação de alimentos, espaço e assim por diante.

Temos então a equação logística

$$\frac{dN}{dt}(t) = f(N)N(t), \quad N(0) = N_0, \tag{2.2}$$

onde a taxa de variação agora depende da população e é dada por $f(N)$. Vamos construir uma função $f(N)$ que tenha as propriedades desejadas, ou seja:

(a): A população cresce exponencialmente enquanto ela for pequena.

(b): O crescimento vai desacelerando conforme a população vai crescendo.

(c): Observamos mortalidade quando a população for muito grande. Matematicamente escrevemos isso na forma $f(N) < 0$ quando $N \gg 1$.

A pessoa encarregada de escrever um modelo propõe a seguinte função:

$$f(N) = r\left(1 - \frac{N}{M}\right).$$

A constante M indica o ponto a partir do qual observaremos a mortalidade de indivíduos, pois quando $N > M$ a taxa de variação passa a ser negativa. Ver item (c) acima. Note que quando $N/M \ll 1$ (que significa "muito pequeno") o termo em parênteses é aproximadamente igual a 1 e a taxa de crescimento é aproximadamente igual a r, de acordo com o que foi "encomendado" no item (a) acima. Usando Cálculo Diferencial é fácil ver que quando a população estiver no intervalo $[M/2, M]$, a taxa de variação ainda é positiva, mas começa a decair. Portanto satisfazemos o item (b). Podemos ver tudo isso através do gráfico dN/dt versus N apresentado na figura 15.

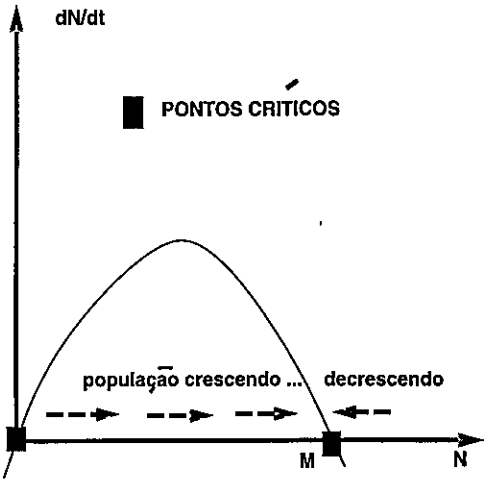

Figura 15: Gráfico dN/dt versus $N(t)$.

Um modelo um pouco mais complexo é dado a seguir:

$$\frac{dN}{dt}(t) = \left[-r\left(1-\frac{N}{M_1}\right)\left(1-\frac{N}{M_2}\right)\right]N(t), \quad N(0) = N_0. \tag{2.3}$$

Fazendo o gráfico N versus dN/dt, podemos entender o que este modelo representa. A taxa de variação é negativa quando $0 < N < M_1$ ou $N > M_2$, e positiva para $M_1 < N < M_2$. Os pontos $N = 0$, $N = M_1$ e $N = M_2$ são chamados de pontos críticos, ou estacionários, pois a população não varia (i.e., $dN/dt = 0$) se começar com esses valores. Os pontos críticos $N = 0$ e $N = M_2$ são pontos estáveis pois, se começarmos com valores nas suas respectivas vizinhanças, a população irá gradativamente se aproximando do valor crítico. O ponto crítico $N = M_1$ é instável pois, se tivermos uma população inicial na sua vizinhança ela irá gradativamente se afastando do mesmo. Convença-se desses fatos estudando o gráfico $N \times dN/dt$. Podemos concluir que esse modelo simples (com aplicação em epidemiologia; hoje em dia os modelos epidemiológicos são muito mais sofisticados) contém dois regimes importantes: abaixo do valor M_1 a população não tem força suficiente e é levada à extinção; a partir de M_1, ela se sente fortalecida e cresce gradativamente, tendendo

para o valor estacionário M_2.

Finalmente vamos estudar um modelo onde duas populações competem para sobreviver. Seja $N_1(t)$ a população das presas e $N_2(t)$ a população dos predadores. Vamos supor que as presas tem um suprimento abundante de alimento, enquanto que o predador se alimenta das presas. Vamos também supor que um biólogo nos forneceu as seguintes hipóteses: (todas as constantes $k_i, i = 1, 2, 3, 4$ dadas abaixo são tidas como positivas)

Presa:

Taxa de natalidade: é proporcional à população naquele instante: $+k_1N_1(t)$.

Taxa de mortalidade: depende da população das presas assim como da dos predadores. Por simplicidade adotamos a expressão $-k_2N_1(t)N_2(t)$. Note que a mortalidade aumenta quando quaisquer das populações cresçe.

Predador:

Taxa de natalidade: é proporcional à população de predadores assim como à das presas: $+k_3N_1(t)N_2(t)$.

Taxa de mortalidade: é proporcional à população de predadores unicamente: $-k_4N_2(t)$.

Colocando essa informação na forma de edo's temos o seguinte sistema

$$\begin{cases} dN_1/dt = k_1N_1 - k_2N_1N_2 \\ dN_2/dt = k_3N_1N_2 - k_4N_2, \end{cases}$$

com dados iniciais $N_1(0) = N_1^0$ e $N_2(0) = N_2^0$. Esse sistema é conhecido como o modelo de Lotka-Volterra.

Neste problema temos dois pontos críticos: $(0, 0)$ e $(k_4/k_3, k_1/k_2)$. Esses são os valores de N_1 e N_2, respectivamente, tais que o lado direito do sistema de edo's

seja zero, ou seja, as populações não variam no tempo. Por isso são chamados de pontos críticos, ou também de pontos estacionários. Denotemos esses pontos por (N_1^*, N_2^*).

É muito útil estudar o comportamento da solução na vizinhança dos pontos críticos. No ponto crítico a solução não varia ao longo do tempo. Na vizinhança dos mesmos, a sua dinâmica deve ser relativamente simples. Vamos traduzir esse fato em linguagem matemática. Se estamos supondo que as populações estão na vizinhança dos pontos críticos, matematicamente isso deve ser expresso na forma

$$N_1(t) = N_1^* + \varepsilon \tilde{N}_1(t) \tag{2.4}$$

e

$$N_2(t) = N_2^* + \varepsilon \tilde{N}_2(t), \tag{2.5}$$

onde ε é um número pequeno. As funções $\tilde{N}_{1,2}(t)$ representam as variações das populações da presa e predador, em torno do ponto crítico (N_1^*, N_2^*). Quanto menor for o ε mais perto do ponto crítico estamos. O que precisamos saber agora é como se comportam essas funções $\tilde{N}_{1,2}$, que ainda não conhecemos. Para obter uma resposta, façamos a substituição nas equações diferenciais:

$$\varepsilon \frac{d\tilde{N}_1}{dt} = k_1 N_1^* + \varepsilon k_1 \tilde{N}_1 - k_2 \left(N_1^* N_2^* + \varepsilon (N_1^* \tilde{N}_2 + \tilde{N}_1 N_2^*) + \varepsilon^2 \tilde{N}_1 \tilde{N}_2 \right)$$

$$\varepsilon \frac{d\tilde{N}_2}{dt} = k_3 \left(N_1^* N_2^* + \varepsilon (N_1^* \tilde{N}_2 + \tilde{N}_1 N_2^*) + \varepsilon^2 \tilde{N}_1 \tilde{N}_2 \right) - k_4 N_2^* - \varepsilon k_4 \tilde{N}_2.$$

Alguns termos desaparecem, levando-se em conta a definição de ponto crítico. Temos que

$$k_1 N_1^* - k_2 N_1^* N_2^* = 0$$

e

$$k_3 N_1^* N_2^* - k_4 N_2^* = 0.$$

A seguir, fazemos uma aproximação que equivale a linearizar o sistema de equações, tornando-o mais simples. Por hipótese, sabemos que o valor de ε^2 é muito menor

do que o de ε. Sendo assim, consideraremos os termos multiplicados por ε^2 desprezíveis diante dos outros termos. Eliminamos os termos multiplicados por ε^2, para obter o sistema linear de equações diferenciais ordinárias dadas abaixo:

$$\frac{d\tilde{N}_1}{dt} = k_1\tilde{N}_1 - k_2(N_1^*\tilde{N}_2 + \tilde{N}_1N_2^*)$$

$$\frac{d\tilde{N}_2}{dt} = k_3(N_1^*\tilde{N}_2 + \tilde{N}_1N_2^*) + -k_4\tilde{N}_2.$$

O sistema linear é re-escrito de maneira a identificarmos, claramente, a matriz "A":

$$\frac{d}{dt}\begin{bmatrix} \tilde{N}_1(t) \\ \tilde{N}_2(t) \end{bmatrix} = \begin{bmatrix} (k_1 - k_2N_2^*) & -k_2N_1^* \\ k_3N_2^* & (k_3N_1^* - k_4) \end{bmatrix}\begin{bmatrix} \tilde{N}_1(t) \\ \tilde{N}_2(t) \end{bmatrix}$$

Este é o sistema $O(\varepsilon)$, ou seja, o sistema obtido ao coletarmos os termos que estão multiplicados por ε.

Na vizinhança do primeiro ponto crítico (0,0) as taxas de variação, de cada população, são respectivamente $\lambda_1 = k_1$ e $\lambda_2 = -k_4$. Estes são os autovalores da matriz A, com $(N_1^*, N_2^*) = (0,0)$. Isto já era de se esperar. Porquê? As populações são muito pequenas ($N_{1,2}(t) = \varepsilon\tilde{N}_{1,2}(t)$) e a interação entre elas é desprezível. A população das presas (por terem alimentos abundantes) cresce de acordo com a sua taxa de natalidade, enquanto que a de predadores (por não terem alimentos o suficiente – i.e. presas) decai de acordo com a sua taxa de mortalidade. Verifique que a frase acima é uma tradução em português, do que está expresso pelos autovalores. É como se uma população não percebesse a existência da outra. Note que o sistema linear, para este ponto crítico, já está desacoplado. Não precisamos nem resolver a equação característica $det(A - \lambda I$.

No entanto, na vizinhança do outro ponto crítico $(k_4/k_3, k_1/k_2)$,

$$A = \begin{bmatrix} 0 & -k_2k_4/k_3 \\ k_1k_3/k_2 & 0 \end{bmatrix}$$

e as taxas de variação são $\lambda_{1,2} = \pm\sqrt{-1}\sqrt{k_1k_4}$. Lembrando da fórmula de Euler ($\exp(\sqrt{-1}\theta) = \cos(\theta) + \sqrt{-1}\sin(\theta)$), concluímos que na vizinhança deste ponto

crítico as populações oscilam. Note que o sistema linear mantém as duas populações acopladas. A interação entre as duas populações gera o movimento oscilatório das mesmas.

Vamos verificar esses fatos numericamente. Usaremos os dois exemplos para testar e validar os códigos escritos. Usando uma rotina do MATLAB (ode23) para resolução de edo's, obtemos os resultados apresentados nas figuras a seguir. Essa rotina escolhe o passo no tempo automaticamente. Ela é *auto-adaptativa*. O leitor interessado é fortemente recomendado a adaptar o programa dado (runpop.m) para os casos (mais simples) do método de Euler, explícito e implícito. Uma vez feita a adaptação, o leitor deverá usar os seus conhecimentos do problema-teste, olhar para as equações de $O(\varepsilon)$ obtidas acima, e escolher os passos no tempo em função dos autovalores em cada caso.

Em todos os experimentos, usamos $k_1 = 3$, $k_2 = 0,002$, $k_3 = 0,0006$ e $k_4 = 0,5$ (Burden e Faires). Para esses valores, o segundo ponto crítico passa a ser $(833,33...,1500)$. No primeiro experimento, começamos com dados iniciais na vizinhança do primeiro ponto crítico. Fica claro (figura 16) o comportamento descrito acima: a população de presas cresce exponencialmente enquanto que a de predadores decai exponencialmente. Não medimos as taxas de variação, mas devem ser aproximadamente k_1 e $-k_4$, respectivamente.

No segundo experimento (figura 17), vemos as duas populações oscilando, pois começamos perto do segundo ponto crítico. No terceiro experimento (figura 18) escolhemos dados iniciais sem propriedades específicas.

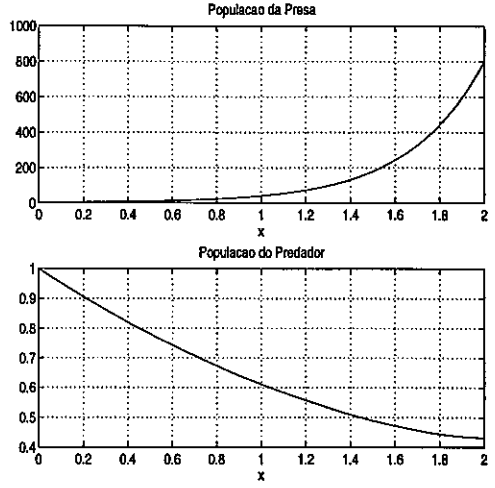

Figura 16: Solução começando na vizinhança do primeiro ponto crítico $(0, 0)$.

Usamos $(N_1^0 = 2, N_2^0 = 1)$.

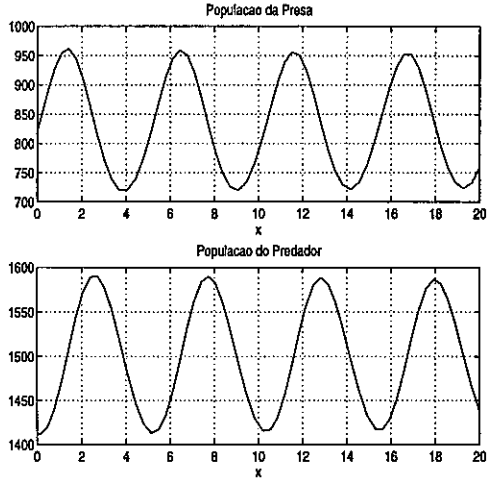

Figura 17: Solução começando na vizinhança do segundo ponto crítico.

Usamos $(N_1^0 = 820, N_2^0 = 1412)$.

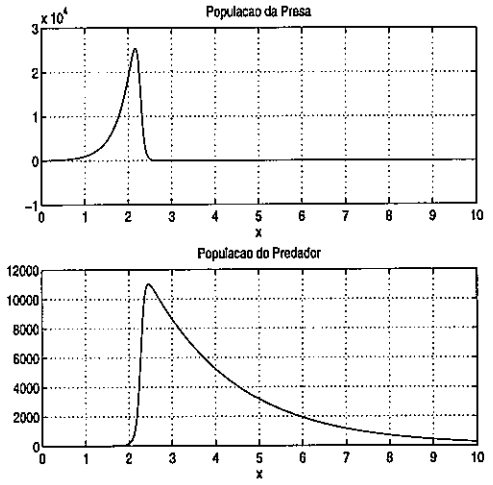

Figura 18: Solução com dados iniciais $N_1^0 = 50$ e $N_2^0 = 10$.

Vemos que as duas populações passam por um pico, após o qual ambas entram (efetivamente) em extinção. O que ocorre é que a população de predadores cresce o suficiente para matar todas as presas. Em seguida, sem alimentos, a própria população de predadores entra em extinção. Note que acima dissemos **efetivamente em extinção.** Isso se deve ao fato das populações de presas e predadores serem funções periódicas que chegam bem perto do valor zero, permanecendo um tempo na vizinhança do zero, para mais adiante repetir o mesmo ciclo de crescimento e decaimento. Convença-se desse fato repetindo o experimento anterior, só que agora com um intervalo de tempo igual a 30. Matematicamente vemos isso através do retrato de fase do sistema de Lotka-Volterra, dado acima (Hirsh e Smale, Murray).

2.2 Por que as expectativas de vida das cigarras são números primos?

2.2.1 Cigarras periódicas

Você provavelmente já escutou cigarras cantando, numa preguiçosa tarde de verão no campo. No entanto, talvez você não tenha se dado conta de que as cigarras passam a maior parte de suas vidas debaixo da terra, saindo por apenas duas semanas. Nesse período elas cantam, se reproduzem e morrem. As duas espécies de cigarras que vivem na parte leste e sul dos Estados Unidos são surpreendentes. Essas cigarras passam, respectivamente, dezessete e treze anos debaixo da terra! De repente, elas aparecem todas de uma vez, em poucos dias, em áreas de algumas centenas de quilômetros de raio. Elas colocam os seus ovos em árvores; esses ovos caem na superfície da terra e novas cigarras nascem na forma de ninfas. As ninfas penetram no solo e se apegam às raízes. Elas vão se desenvolvendo lentamente ao longo dos próximos treze ou dezessete anos (treze no sul dos E.U. e dezessete no leste), até poucos dias antes de saírem para repetir o ciclo descrito acima. De modo que existem os "anos das cigarras", eventos raros que se repetem apenas de tantos em tantos anos, como uma surpresa da natureza, nos quais de repente aparece uma miríade de buracos no solo e cigarras por todos os lados cantando e acasalando.

O comportamento estranho dessas cigarras periódicas requer uma explicação. Por que elas esperam por tanto tempo? E por que elas saem todas ao mesmo tempo? Uma explicação plausível é dada em Jay Gould, e diz respeito a um mecanismo chamado de *saciação do predador* (*"predator satiation"*). A idéia é a seguinte: quando as cigarras saem do solo, os predadores – em geral pássaros – se alimentam delas. Se as cigarras aparecessem com frequência, os pássaros iriam prosperar tanto que eventualmente levariam as cigarras à extinção. Como

as cigarras só aparecem na superfície após longos intervalos de tempo, elas se garantem desta forma que os pássaros não possam depender delas para viver. Na pior das hipóteses, vai haver um único banquete para pássaros de três em três (ou quatro em quatro) gerações. De mais a mais, as cigarras saem do solo aos milhões, e mesmo com os pássaros fazendo a festa, vão sobrar suficientes cigarras para perpetuar a espécie; daí o nome *saciação do predador.*

Esse mecanismo também explica a necessidade de uma sincronização precisa: se algumas cigarras sairem muito cedo, ou muito tarde, elas serão presas fáceis para os pássaros que estarão "esperando". Saindo todas juntas, elas diminuem bastante a probabilidade de serem todas devoradas.

Para o leigo, parece que chegamos ao fim de uma estória sobre "o estranho estilo de vida das cigarras". No entanto, uma pessoa "matematicamente atenta" vai ficar encucada: ela provavelmente vai notar que as expectativas de vida dessas cigarras são números primos! Será apenas uma coincidência? Ou existirá uma razão para que números primos tenham preferência, sobre os não-primos, na história da evolução das cigarras? Este será um dos pontos chave da nossa investigação.

Na verdade, esta é uma questão de debate entre biólogos: muitos pensam que 13 e 17 anos apareceram ao acaso ([11][12]). No entanto, Stephen Jay Gould fornece uma bonita explicação de por que números primos foram selecionados ([6]). O argumento é o seguinte:

> Os predadores em potencial vivem em ciclos de 2 a 5 anos. Esses ciclos não originaram por causa das cigarras! Afinal, os predadores tem vários picos (populacionais) durante os anos de não-emergência das cigarras. Por exemplo, se as cigarras viessem à superfície a cada 15 anos, elas poderiam ser abatidas por um pico dos predadores. Como o seu ciclo é um primo muito grande, as chances de uma coincidência são mínimas. Ciclos de treze ou dezessete anos muito raramente vão coincidir com

ciclos de poucos anos.

Esse argumento encantador tem problemas. De fato, poderíamos argumentar que caso o ciclo das cigarras fosse de 15 anos, e o dos predadores de 5 anos, as cigarras poderiam levar vantagem. Se os picos não coincidirem, elas nunca veriam o "florescer" dos predadores. A pista para esse contra-argumento é a de que os ciclos dos predadores nada tem a ver com a disponibilidade de cigarras. Os pássaros poderiam ir adaptando-se lentamente aos picos das cigarras e eventualmente conseguir a sincronização de picos.

O evento das cigarras é raro e afeta muito pouco os pássaros. Como pôde esse evento evoluir e adaptar-se a circunstâncias que ocorrem uma vez, por duas semanas, de três em três gerações de pássaros? A resposta é dada através de um conceito matemático: ressonância. Este é um mecanismo que permite pequenas perturbações se acumularem ao longo do tempo, para ao final produzirem um efeito significativo.

2.2.2 Ressonância

O fenômeno de resonância mais comum, acontece em balanços de um "playground". Mesmo quando damos pequeninos empurrões a intervalos regulares, intervalos esses que são múltiplos do período do balanço, o balanço vai alcançando alturas cada vez maiores. Analogamente, cada aparição das cigarras funciona como um pequeno "empurrão" na população dos predadores, os pássaros, que nestas ocasiões se deparam com um suprimento ilimitado de comida. Se esses pequenos "empurrões" acontecerem regularmente, a um múltiplo da média da expectativa de vida de uma das espécies de pássaros, esta espécie irá prosperar, o que eventualmente conduzirá à extinção das cigarras. Assim, as cigarras mais adequadas ao processo de sobrevivência são aquelas que vem à superfície a um número primo de anos, já que primos não são múltiplos de nenhum número (a expectativa de vida dos

predadores).

Como funciona a matemática do mecanismo de ressonância? Vamos primeiro ao exemplo do balanço. Podemos modelar o balanço com a seguinte (simples) equação diferencial ordinária:

$$\frac{d^2x}{dt^2} + \frac{g}{L}x = 0\,; \tag{2.6}$$

onde x representa o deslocamento angular do balanço, a partir da sua posição de equilíbrio vertical, g é a aceleração da gravidade (aproximadamente $10m/s^2$), e L é o comprimento das cordas que seguram o balanço. Esse é o modelo linear do pêndulo, ou seja, para pequenos deslocamentos. A solução da equação (2.6) tem uma forma muito simples:

$$x(t) = x_h(t) = A\cos(\omega t - \alpha)\,, \tag{2.7}$$

onde ω, a *frequência natural do balanço*, é dada por

$$\omega = \sqrt{\frac{g}{L}}\,, \tag{2.8}$$

com A e α constantes arbitrárias, determinadas a partir das condições iniciais em $t = 0$. Denotamos por x_h a "solução homogênea", significando que o lado direito da equação é zero, ou seja, temos um balanço não-forçado (sem empurrões). As soluções em (2.7) representam oscilações periódicas de período $T = 2\pi/\omega$ com amplitudes A. O balanço vai oscilar para sempre, já que no nosso modelo não temos fricção.

Como podemos modelar um balanço forçado, isto é, um balanço que está sendo empurrado? A equação mais simples descrevendo um balanço forçado é a seguinte variação da (2.6):

$$\frac{d^2x}{dt^2} + \frac{g}{L}x = f(t)\,, \tag{2.9}$$

onde $f(t)$ é o forçante, que depende do tempo. Agora temos uma equação não-homogênea, já que esta contém um termo que não depende de x nem de suas

derivadas. Consideremos um forçante periódico, já que queremos modelar uma pessoa que empurra o balanço em intervalos regulares (da mesma maneira que as cigarras "empurram" os pássaros a cada 17 anos). Um exemplo simples é $f(t) = B\cos(\lambda t)$, onde λ é a frequência do forçante. Para este forçante (e por isso começamos com este exemplo) a equação (2.9) tem uma solução fechada (sem usar expansão em série) muito simples:

$$x(t) = \frac{B}{\omega^2 - \lambda^2}\cos(\lambda t) + x_h(t). \tag{2.10}$$

Vemos que esta solução tem duas componentes: uma homogênea, x_h, que oscila indefinidamente (como no caso homogêneo), e outra não-homogênea, que oscila de acordo com a frequência do forçante. Note que a parte não-homogênea será muito grande quando a frequência λ do forçante for muito próxima de ω, a frequência natural do balanço. Na verdade, a solução (2.10) deixa de fazer sentido quando essas duas frequências são exatamente as mesmas. É fácil verificar que nesse caso a solução é dada por

$$x(t) = \frac{B}{\omega}t\sin(\omega t) + x_h(t). \tag{2.11}$$

Essa solução oscila com a frequência natural do balanço, mas com uma amplitude que cresce indefinidamente. Após um certo intervalo de tempo, a amplitude é tão grande que o modelo linear do pêndulo (2.9) deixa de fazer sentido. Lembre-se que esse modelo foi construído em cima da hipótese de pequenos deslocamentos angulares. O crescimento indefinido da amplitude é uma manifestação do fenômeno de ressonância.

Do ponto de vista físico, a interpretação do modelo forçado é que estamos empurrando o balanço uma vez por oscilação (i.e. por período). Isso vai dando mais e mais energia ao movimento do balanço. Devemos fazer esse ponto ficar mais claro. No caso $f(x) = \cos(\omega t)$ o forçante está sempre presente, como se tivéssemos um forçante pendurado no balanço. Esse forçante pode ser usado para

modelar uma criança balançando sozinha. Com as suas próprias pernas a criança faz o mecanismo do forçante pendurado no balanço. A criança dá um "chute" com os dois pés quando o balanço vai para a frente e dá uma rápida "encolhida" nas pernas ("chute com os dois calcanhares") quando o balanço vai para trás. Abaixo mostraremos como modelar uma pessoa parada, dando empurrões no balanço.

O que aconteceria se empurrássemos o balanço a cada duas oscilações? À primeira vista pode parecer que com $\lambda = \omega/2$ nada de especial irá acontecer à solução (2.10). No entanto isto não parece estar fisicamente correto. E de fato não está. Os nossos empurrões são funções periódicas, mas não são senos nem cossenos. Veja o exemplo dado em (2.12) abaixo. Sabemos, no entanto, que qualquer função periódica, com um certo grau de regularidade, pode ser decomposta em uma soma de senos e cossenos, ou seja, em uma série de Fourier:

$$f(t) = \sum_{k=0}^{\infty} [a_k \cos(\lambda k t) + b_k \sin(\lambda k t)],$$

onde os coeficientes a_k e b_k são facilmente calculados através de fórmulas envolvendo integrais de $f(t)$ sobre o seu período. A razão para fazermos esta decomposição de $f(t)$ é muito simples. Sabemos achar a solução para forçantes na forma de senos ou cossenos. Fizemos isto acima! No caso do $f(t)$ genérico, dado acima pela sua série de Fourier, usamos o princípio da superposição (já que a equação é linear). Achamos a solução para um termo genérico (do seno e do cosseno) da série de Fourier e somamo-las ao final. Note então que se a frequência é $\lambda = \omega/2$, um termo de frequência ω vai aparecer na série de Fourier quando $k = 2$. A solução que corresponde a este termo vai crescer linearmente no tempo, e veremos ressonância de novo. Este mesmo raciocínio se aplica quando a frequência dos empurrões no balanço for qualquer múltiplo do período natural do balanço.

A figura 19 mostra soluções numéricas para a equação (2.9) com $g/L = 1$ (o

que implica em $\omega = 1$ e período 2π) e com o forçante

$$f(t) = \begin{cases} 1 & \text{para } 0 < t < \frac{T}{8} \\ 0 & \text{para } \frac{T}{8} < t < T \\ f(t-T) & \text{para } t > T \end{cases} \tag{2.12}$$

correspondendo a empurrões de duração $T/8$ e período T. Faça o gráfico desse forçante $f(t)$. Podemos ver que, quando T se aproxima de um múltiplo do período natural 2π, as oscilações do balanço ficam cada vez maiores. Quando T é exatamente um múltiplo de 2π, a amplitude das oscilações cresce linearmente no tempo. O pacote MATLAB, usado para realizar os experimentos computacionais, está descrito no apêndice. Recomenda-se ao leitor fazer uma série de experimentos por conta própria!

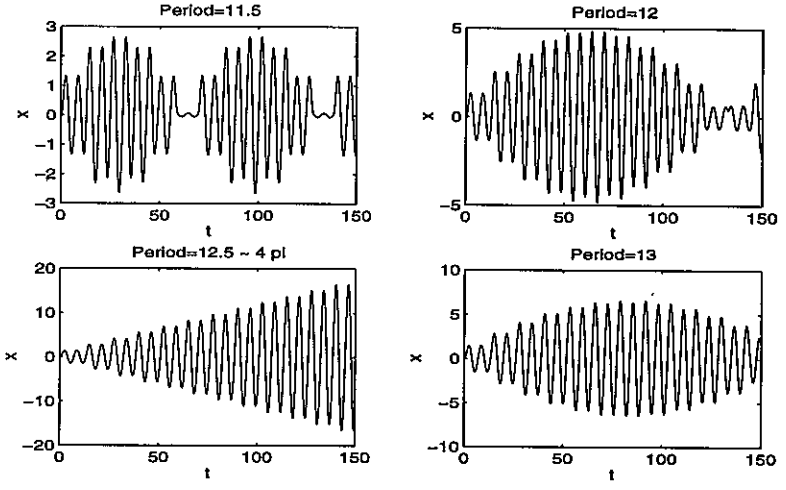

Figura 19: Oscilador com diferentes forçantes:

$\lambda = 11,5, \quad 12, \quad 12,5$ e $\quad 13$.

2.2.3 Um modelo para as cigarras

De maneira a aplicar o conceito de ressonância no exemplo das cigarras e pássaros, precisamos fazer alguns ajustes em (2.9). Antes de mais nada: nem as cigarras, nem os pássaros, tem frequências naturais ω ajustáveis por parâmetros. Suas

expectativas de vida são dadas por números inteiros, em anos. Vale observar que o mecanismo pelo qual organismos vivos se ajustam aos ciclos das quatros estações pode em si envolver ressonâncias. As expectativas de vida inteiras indicam que devemos usar um modelo discreto no tempo (em vez do contínuo (2.9)), no qual as populações de cigarras e pássaros são avaliadas uma vez por ano.

Seguindo essa linha de raciocínio, vamos desenvolver um modelo computacional para cigarras e pássaros, adaptado a partir de um modelo de Hoppensteadt e Keller, que trata apenas de cigarras e que não inclui efeitos de ressonância. O nosso modelo vai incluir diversas espécies de cigarras e pássaros, cada qual com uma expectativa de vida diferente. Fazemos isso para poder verificar se, de fato, as cigarras com expectativa de vida igual a um número primo têm maiores chances de sobrevivência. Este é o nosso **alvo principal**, no presente estudo. Para rastrearmos a evolução de cada geração de cigarras, a cada ano, temos que registrar a idade de cada cigarra e pássaro. Com essa finalidade introduzimos as variáveis

nc	Número de espécies de cigarras
$x(j,k)$	População de cigarras da espécie k com idade j
$nx(k)$	Expectativa de vida das cigarras da espécie k

para as cigarras e

np	Número de espécies de predadores
$y(j,k)$	População de predadores da espécie k com idade j
$ny(k)$	Expectativa de vida dos predadores da espécie k

para os pássaros. Vamos monitorar a evolução de $x(j,k)$ e $y(j,k)$ ao longo dos anos. Para tal, precisamos decidir como faremos a atualização dessas populações. Para $1 < j \leq nx(k)$, $x^{n+1}(j,k)$, o número de cigarras da espécie k no ano $n+1$, será dado por $x^n(j-1,k)$, i.e. o número de cigarras que eram um ano mais novas no ano anterior, multiplicado pelo coeficiente $ax(k)$ que mede a taxa anual de sobrevivência das cigarras (da espécie k) sob a terra. Simbolicamente temos

$$x^{n+1}(j+1,k) = ax(k) * x^n(j,k). \tag{2.13}$$

Obviamente uma expressão semelhante pode ser escrita para os pássaros. No entanto, neste caso, a taxa anual de sobrevivência depende, mesmo que fracamente, do número de cigarras adultas que saem do solo a cada ano. Levando isto em conta, para os pássaros temos a expressão

$$y^{n+1}(j+1,k) = ay(k) * y^n(j,k), \tag{2.14}$$

onde

$$ay(k) = ay0(k) + ay1(k) * xn\,,$$

xn representando o número total de cigarras adultas que emergiram neste ano.

O que fazemos com respeito às cigarras recém-nascidas, i.e. $x^{n+1}(1,k)$? Essa população depende do número de cigarras adultas que saem do solo ($x^n(nx(k),k)$), assim como de outros fatores. Para começar, temos que alguns pássaros podem comer algumas cigarras antes destas terem tempo de colocar seus ovos. Para representar isto matematicamente, devemos multiplicar $x^n(nx(k),k)$ pela probabilidade de uma cigarra da espécie k ser comida. A probabilidade desse evento é proporcional ao número de pássaros (e pode ser ponderada pela idade do pássaro, que afeta o apetite), e inversamente proporcional ao total de cigarras, de todas as espécies, que podem ser comidas naquele momento. Podemos traduzir isto, matematicamente, através da expressão

$$x^{n+1}(1,k) = (rx(k) * ax(k) - xy) * x^n(nx(k),k)\,,$$

onde $rx(k)$ é o número de crias por cigarra, enquanto que xy é a proporção de cigarras comidas, da espécie k. Sua expressão é

$$xy = dxy * yt/xn\,,$$

onde dxy é uma constante, yt é o número total de pássaros, ponderado pela idade, e xn é o número total de cigarras adultas.

Esse modelo discreto está repleto de ingredientes. Sugerimos ao leitor acompanhar a leitura desse texto, com a leitura do programa apresentado no apêndice. O leitor irá notar que certos detalhes da modelagem, refrentes aos diversos ingredientes, só aparecerão no programa. Tentamos com isso evitar que o texto se torne um emaranhado de pequenos detalhes.

Continuando com a modelagem ... Outro fator: o subsolo não é uma fonte infinita de nutrientes. Isto significa que nem todas as ninfas irão sobreviver. Como colocamos essa informação no nosso modelo? Para modelar este aspecto, introduzimos o número K, representando a capacidade do solo de comportar ninfas. Após contar todas as ninfas dos anos anteriores, digamos K_t, só podemos adicionar $K - K_t$ novas ninfas. Se nascerem mais do que $K - K_t$, o excesso não terá chances de sobreviver e alcançar a maturidade. A quantia $x^{n+1}(1,k)$, calculada acima, será ainda mais reduzida.

Podemos, de forma semelhante, colocar restrições no número de pássaros nascidos; mas evitaremos colocar qualquer tipo de predador de pássaros, de maneira a não complicar demais o nosso modelo.

Os detalhes desse modelo estão explicados nos comentários do programa MATLAB. No final do programa encontram-se as equações para a evolução da população de cigarras e para a população de pássaros. A maior dificuldade (isto é quase uma arte) é calibrar os parâmetros corretamente, de maneira a não gerar uma situação onde, por exemplo, nenhuma cigarra sobreviverá (que ocorreria quando a população de pássaros crescesse rapidamente). Devemos também ter cuidado para não gerar um sistema tendencioso, no qual uma espécie de cigarras ou de pássaros seria privilegiada. Afinal de contas, nós queremos verificar o efeito de ressonância entre as duas populações, a longo prazo.

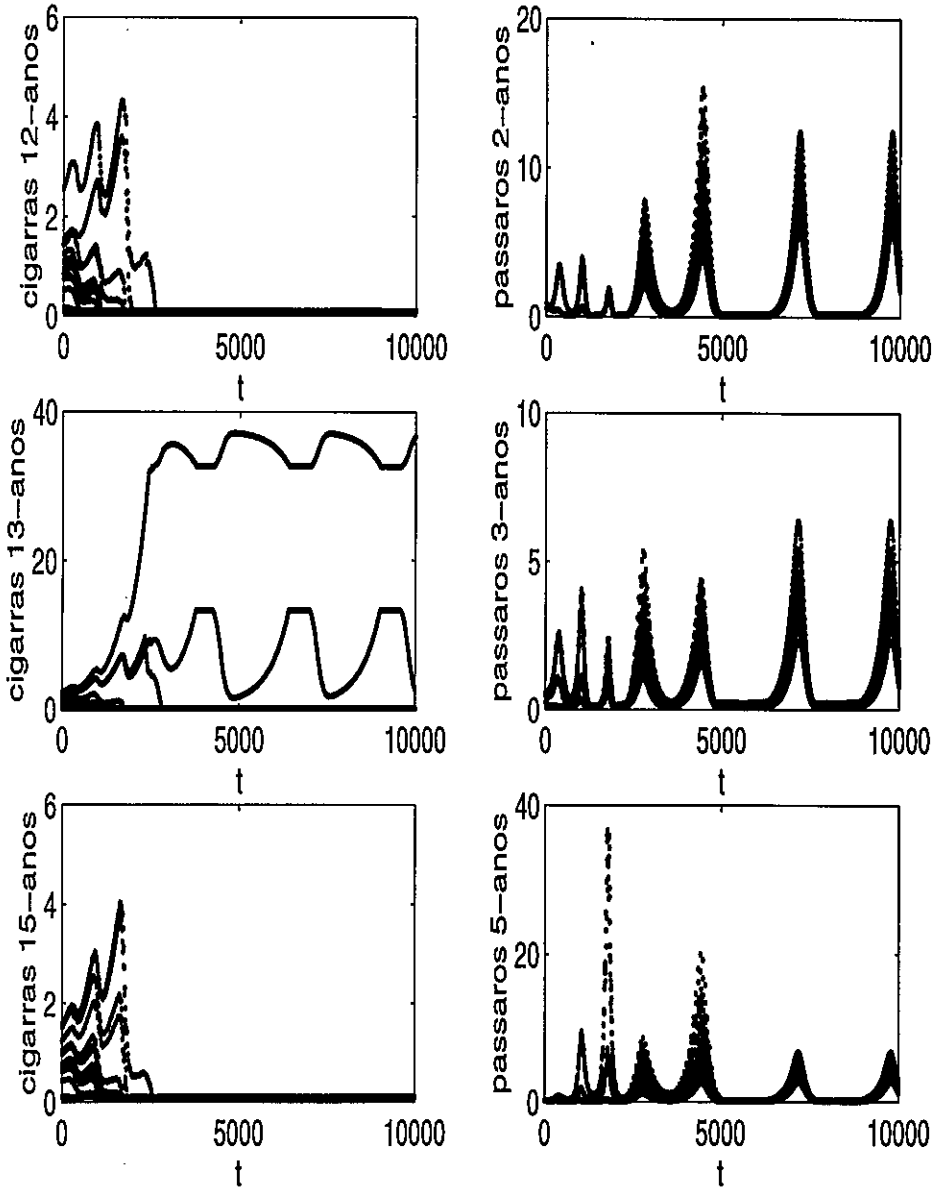

Figura 20: Resultados para o primeiro experimento.

Cada ponto representa o número de indivíduos nascidos naquele ano.

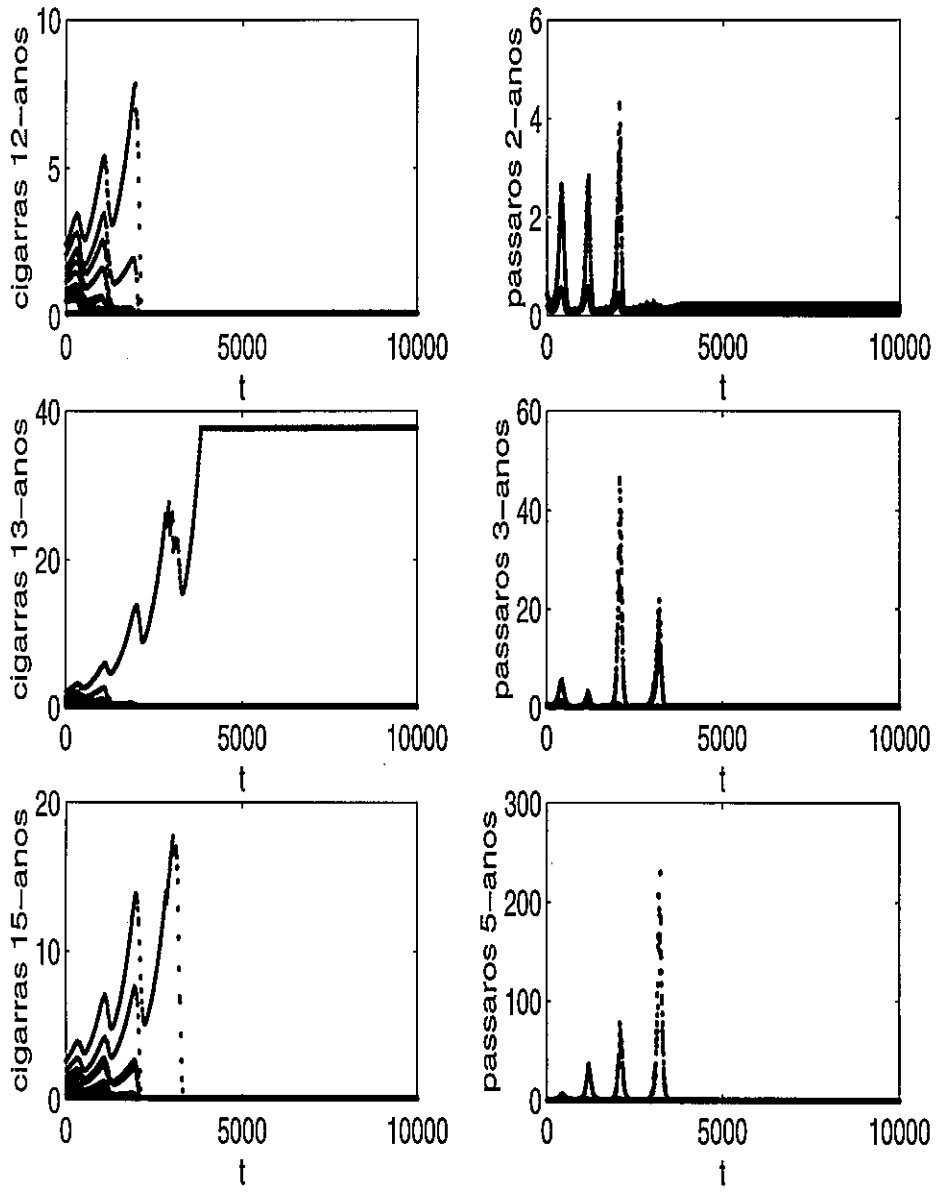

Figura 21: Resultados para o segundo experimento.

Cada ponto representa o número de indivíduos nascidos naquele ano.

As figuras 20 e 21 mostram dois experimentos computacionais com o modelo, nos quais dados iniciais aleatórios foram usados. Fizemos uma sequência de 3 experimentos. No primeiro, as condições iniciais são tais que todas as cigarras vão à extinção. Mostramos o segundo e terceiro experimentos pois nestes dois apenas as cigarras "anos–primos" sobrevivem. Escolhemos três espécies de cigarras, com expectativas de vida iguais a 12, 13 e 15 anos respectivamente. Também escolhemos três espécies de pássaros com expectativas de vida iguais a 2, 3 e 5 anos. A nossa meta é ver se realmente as cigarras "13–anos" são as mais aptas a sobreviverem.

Os gráficos representam o número de recém-nascidos de cada espécie de cigarras e pássaros em função do ano. As curvas que se formam (por exemplo para as cigarras "13–anos") são gerações, ou seja, cigarras da mesma família nascendo a cada 13 anos. As cigarras de outra geração "13–anos" também nascem a cada 13 anos, mas em anos diferentes da outra geração.

Vemos uma extinção em massa com várias gerações de cigarras morrendo. Ao final, vemos que no máximo uma ou duas gerações de uma, ou duas, espécies sobrevivem. Essas espécies que sobrevivem, quase sempre, são as que têm expectativas de vida iguais a números primos. Isso confirma a nossa teoria de ressonâncias. Mais do que isso, quando uma espécie de cigarras vai à extinção, quase sempre este fato está associado a uma enorme população de pássaros com expectativa de vida igual a um divisor da expectativa de vida das cigarras extintas. Após a extinção há um decaimento brutal da população dos pássaros, voltando a um nível normal.

Nos experimentos apresentados nas figuras 20 e 21, apenas cigarras "13–anos" sobrevivem: na figura 20 apenas duas gerações. No caso da figura 21 apenas uma geração "13–anos" sobrevive, refletindo a realidade no sul dos Estados Unidos.

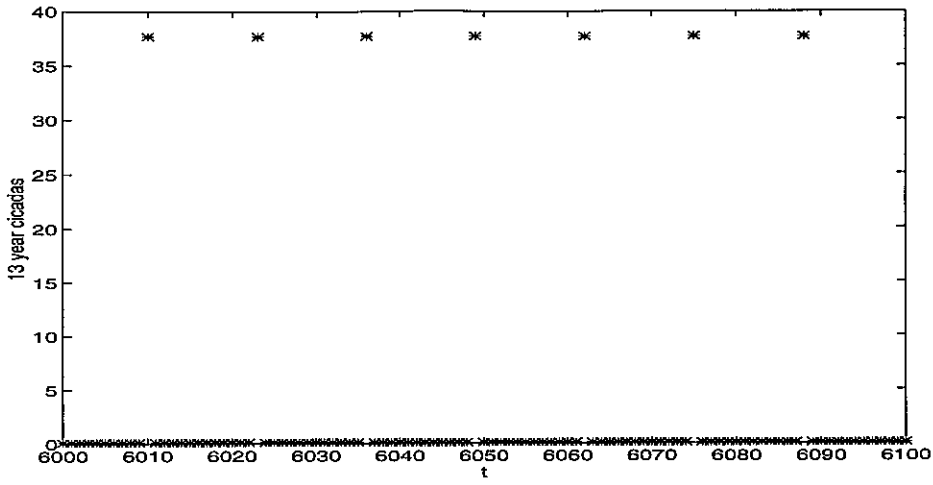

Figura 22: Detalhe do segundo experimento.

A figura 22 mostra detalhes desses últimos resultados, com apenas uma geração emergindo a cada 13 anos. Novamente encorajamos o leitor a fazer modificações nos programas MATLAB, para realizar experimentos próprios. O que acontece se começarmos com apenas uma espécie de cigarras "não–primas" (i.e. a expectativa de vida não é um número primo)? Ela vai à extinção? E se começarmos com uma população bem grande? E depois, quais são so efeitos se mudarmos os parâmetros no programa? Mais ainda ... podemos fazer algum tratamento das hipóteses para ficarem mais realistas? Existem várias perguntas e uma série de experimentos numéricos a serem realizados, na tentativa de respondê-las.

2.3 Modelo para a dispersão de poluentes no ar

Nesta seção apresentaremos um modelo simplificado para estudar a dispersão de poluentes no ar. Consideremos a situação indicada na figura 23 (c.f. Friedman).

A chaminé de uma fábrica solta fumaça com um produto tóxico, com uma concentração inicial igual a α. Uma pessoa interessada em comprar uma casa a

uma distância d da fábrica nos consulta sobre a condição do ar na vizinhança da mesma. Vamos então ver o que necessitamos levar em conta, neste estudo inicial, para fazer uma estimativa da qualidade do ar.

Vamos levar em conta a pior situação possível, na qual o vento sopra na direção da casa com a velocidade máxima (U) até hoje registrada na área. O mecanismo de **transporte**, no qual a nuvem é carregada (transportada) pelo vento sem mudar de forma, é chamado de **advecção**. Devemos levar em conta também o mecanismo de **difusão**, no qual a nuvem vai se espalhando ao mesmo tempo em que a concentração do poluente vai baixando. Esse fenômeno é semelhante à situação em que pingamos uma gota de tinta num enorme recipiente com água e a tinta vai se espalhando, fazendo com que a concentração da tinta (em analogia ao poluente) vá baixando. Após um intervalo de tempo razoável nós nem percebemos que há tinta misturada à água no recipiente.

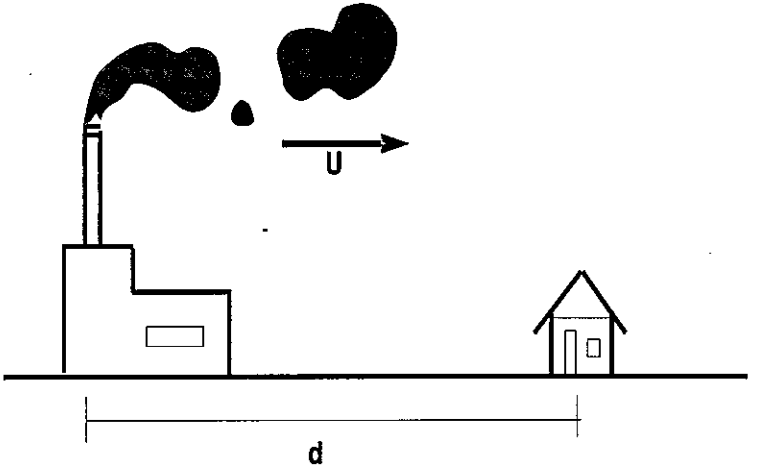

Figura 23: Diagrama ilustrando o problema de dispersão de poluentes

Na análise acima acabamos de escolher o modelo físico, ou seja, decidimos que tipos de mecanismos levaremos em conta no nosso estudo. Passemos agora para

a representação matemática dos mecanismos de transporte e difusão. Obteremos o respectivo modelo matemático, a partir do qual calcularemos o nível final do poluente (i.e., a sua concentração $c = c(x,t)$) nos arredores da casa.

Comecemos pela equação de advecção

$$\frac{\partial c}{\partial t} + U\,\frac{\partial c}{\partial x} = 0, \tag{2.15}$$

com a condição inicial

$$c(x,0) = c_0(x). \tag{2.16}$$

A condição inicial pode ser visualizada na figura 24.

Pode-se mostrar que a solução dessa equação é da forma

$$c(x,t) = c_0(\xi), \quad \xi(x,t) = x - Ut, \tag{2.17}$$

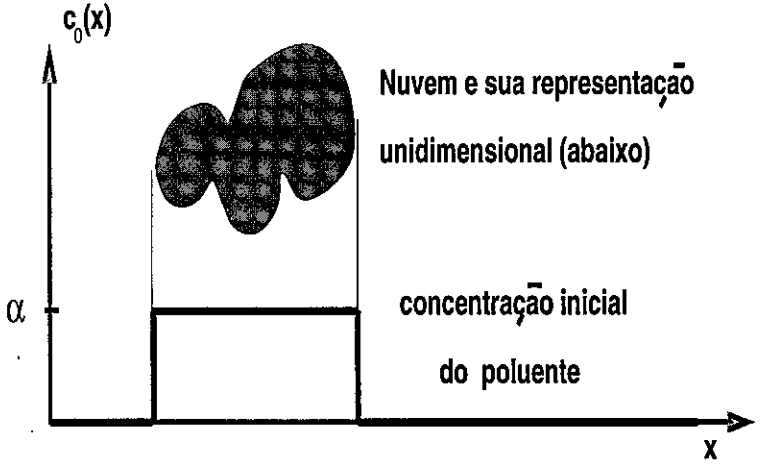

Figura 24: Diagrama ilustrando como modelar uma nuvem de poluente.

que significa que o perfil inicial c_0 da concentração de poluente viaja (i.e., é transladada de acordo com ξ) para a direita com velocidade U. Faça vários gráficos de $c(x,t)$, para diferentes valores do tempo t e verifique esse fato. Verifique também que $c_0(\xi)$ satisfaz a equação de advecção e a condição inicial. Lembre-se de usar a

regra da cadeia! Se a velocidade do vento for variável ($U = U(x,t)$ é uma função conhecida), a equação de advecção deve ser escrita na forma

$$\frac{\partial c}{\partial t} + \frac{\partial Uc}{\partial x} = 0. \tag{2.18}$$

O modelo matemático para a difusão é o mesmo da equação do calor:

$$\frac{\partial c}{\partial t} = \kappa \frac{\partial^2 c}{\partial x^2} \tag{2.19}$$

com a condição inicial

$$c(x,0) = c_0(x), \tag{2.20}$$

onde κ é a constante de difusão que depende do meio em questão. Para entender o fenômeno de difusão consideremos o caso em que a temperatura nos extremos $x = 0$ e $x = L$ é mantida a zero graus: $c(0,t) = c(L,t) = 0$. Seja a distribuição inicial de temperatura $c_0 = \sin(2\pi x/L)$. Devido à forma da condição inicial, vamos supor que a solução pode ser escrita como $c(x,t) = a(t)\sin(2\pi x/L)$. Substituindo esta expressão na equação da difusão, temos uma equação diferencial ordinária para a amplitude $a(t)$ da distribuição de temperatura:

$$\frac{da}{dt} = -\frac{\kappa 4\pi^2}{L} a.$$

Integrando os dois lados com respeito ao tempo, temos que

$$a(t) = \exp(-\frac{4\pi^2\kappa}{L}t)$$

e conseqüentemente

$$c(x,t) = \exp(-\frac{4\pi^2\kappa}{L}t)\sin(\frac{2\pi x}{L}). \tag{2.21}$$

O mecanismo de difusão faz com que a temperatura (ou concentração de poluente) decaia exponencialmente ao longo do tempo.

Para o projeto da estimativa da qualidade do ar temos que combinar os dois mecanismos. Obtemos a equação de advecção-difusão:

$$\frac{\partial c}{\partial t} + U\frac{\partial c}{\partial x} = \kappa \frac{\partial^2 c}{\partial x^2} \tag{2.22}$$

com a condição inicial

$$c(x,0) = c_0(x). \tag{2.23}$$

Note que se fizermos $U = 0$, "desligamos" o transporte, enquanto que se fizermos $\kappa = 0$, "desligamos" a difusão. Com esses dois parâmetros controlamos os dois mecanismos considerados no modelo fisíco.

Usando duas expansões em série de Taylor fazemos as aproximações necessárias para formular um modelo numérico para a resolução da equação diferencial parcial (edp) dada acima. Comecemos com a parte da advecção. Usando série de Taylor no tempo, temos que

$$\frac{\partial c}{\partial t}(j\Delta x, n\Delta t) \approx \frac{c(j\Delta x,(n+1)\Delta t) - c(j\Delta x, n\Delta t)}{\Delta t} = \frac{c_j^{n+1} - c_j^n}{\Delta t},$$

enquanto que no espaço

$$\frac{\partial c}{\partial x}(j\Delta x, n\Delta t) \approx \frac{c(j\Delta x, n\Delta t) - c((j-1)\Delta x, n\Delta t)}{\Delta x} = \frac{c_j^n - c_{j-1}^n}{\Delta x}.$$

Substituindo estas expressões na equação de transporte obtemos

$$c_j^{n+1} = c_j^n - \frac{U\Delta t}{\Delta x}\left(c_j^n - c_{j-1}^n\right), \tag{2.24}$$

que é a equação de diferenças aproximando a edp de transporte original. O método se chama *Método de Diferenças Finitas*. Note que o método é progressivo no tempo (em t) e retroativo no espaço (em x). Isto está relacionado ao fato de que uma das séries de Taylor foi expandida com um incremento positivo (Δt) e a outra com um incremento negativo ($-\Delta x$). Verifique isso por conta própria para ter certeza de que os conceitos e idéias estão bem entendidas. Note também que o esquema numérico é explícito. Isto significa que podemos obter **diretamente** os valores da solução no próximo estágio no tempo (aqui indexado por $(n+1)$). Em um método implícito, vários valores da solução no estágio $(n+1)$ estão acoplados e por isso temos que resolver um sistema de equações algébricas lineares. Isto ficará mais

claro mais adiante. Na figura 25 mostramos como o método numérico propaga (transporta) os valores da solução ao longo do tempo. Em outras palavras, o diagrama mostra como se faz a "ligação" entre os nós vizinhos da malha (grade) no plano espaço-tempo. Essa malha é representada pelos índices $j = 1, 2, ..., J$ e $n = 1, 2, ..., N$. A molécula do método é um diagrama indicando quais nós vizinhos se comunicam diretamente. Isto é obtido diretamente a partir dos índices usados na equação (2.24). Um bom exercício é calcular a velocidade numérica do método. Para tal coloque uma condição inicial que seja (0 0 0 0 1 0 0 0 ...) nos quadradinhos no eixo dos x. Agora usando o método numérico calcule quanto tempo levará para a solução assumir um valor diferente de 0 no ponto $4\Delta x$ à direita de onde o 1 começou.

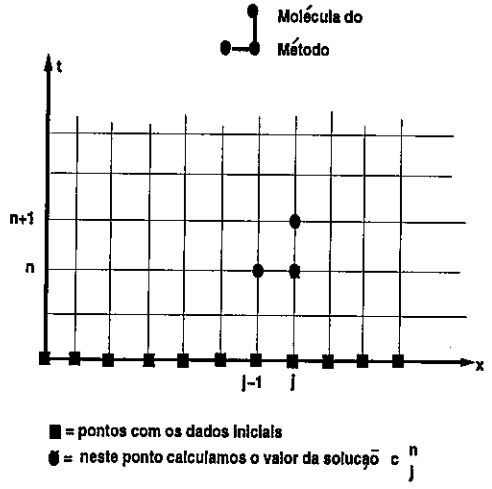

Figura 25: Diagrama com a malha para o problema discreto.

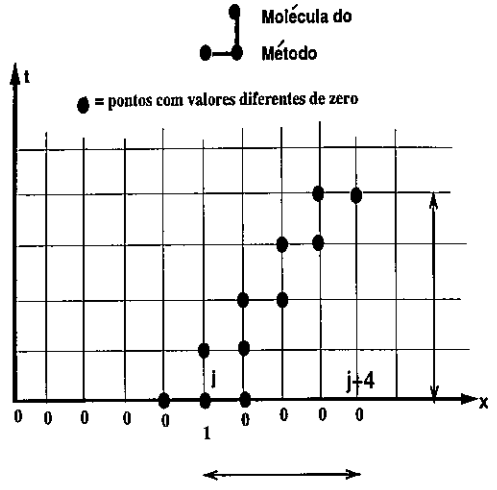

Figura 26: Diagrama com a malha para o problema discreto.

Na verdade, não é necessário fazer contas, bastando usar a molécula do método. A divisão de $4\Delta x$ pelo número de Δt's gastos, nos dá a velocidade numérica de propagação. Veja a figura 26 e repita esse exercício. Usando o método, faça as contas dos valores da solução ao longo da diagonal saindo do ponto $c_j^0 = 1$. Os valores ao longo dessa diagonal são dados por c_{j+n}^n. Faça esse cálculo usando valores genéricos de Δt e Δx; examine a influência desses valores na maneira como o número 1 propaga ao longo do tempo.

Defina o parâmetro $\sigma = U\Delta t/\Delta x$. Existe um teorema afirmando que o método será estável se $0 < \sigma \leq 1$ e instável se $\sigma > 1$. A condição de estabilidade acima é conhecida como condição CFL (em homenagem a Courant, Friedrichs e Lewy; c.f. Ames). Note que esta condição estabelece que a velocidade numérica tem que ser maior ou igual à velocidade de advecção U.

Agora vamos incluir difusão no nosso modelo numérico. Usando 2 séries de Taylor (uma para frente e outra para trás – verifique como) podemos escrever que

$$\frac{\partial^2 c}{\partial x^2}(j\Delta x, n\Delta t) \approx \frac{1}{\Delta x^2}\left(c_{j+1}^n - 2c_j^n + c_{j-1}^n\right). \tag{2.25}$$

Usando esta aproximação na equação de advecção-difusão obtemos

$$c_j^{n+1} = c_j^n - \frac{U\Delta t}{\Delta x}\left(c_j^n - c_{j-1}^n\right) + \kappa\frac{\Delta t}{\Delta x^2}\left(c_{j+1}^n - 2c_j^n + c_{j-1}^n\right). \qquad (2.26)$$

Façamos alguns experimentos numéricos para testar e nos sentirmos mais confiantes com respeito ao código escrito. No primeiro experimento desligamos a difusão (colocando $\kappa = 0$) e testamos apenas a parte de transporte. Usamos como dados $U = 1$ e $\Delta t = \Delta x = 0,1$. A condição inicial, calculada internamente no programa, é um pulso quadrado, cujos dados de entrada são os seus pontos inicial e final. Estamos supondo que a concentração inicial do poluente é $\alpha = 1,0$. Neste exemplo, adotamos para pontos extremos da nuvem $x = 1$ e $x = 2$, respectivamente. Note que o espaçamento espacial e temporal é tal que não violamos a condição CFL. O resultado pode ser visto na figura 27. Na verdade, estamos usando a relação ótima, no sentido em que a velocidade numérica é igual à velocidade de transporte U. A figura 27 representa uma nuvem viajando para a direita com velocidade constante igual a um, e sem mudar de forma. A configuração inicial foi mantida.

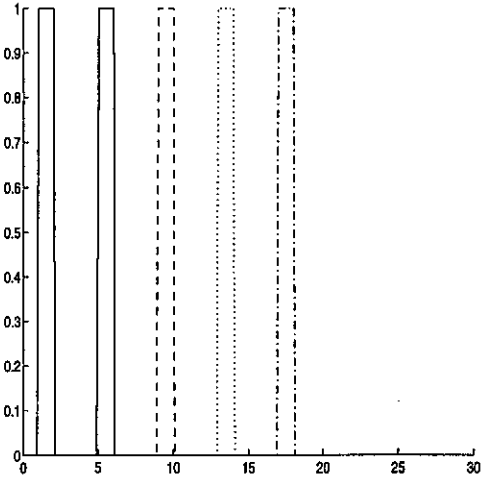

Figura 27: Resultado para o problema de advecção pura.

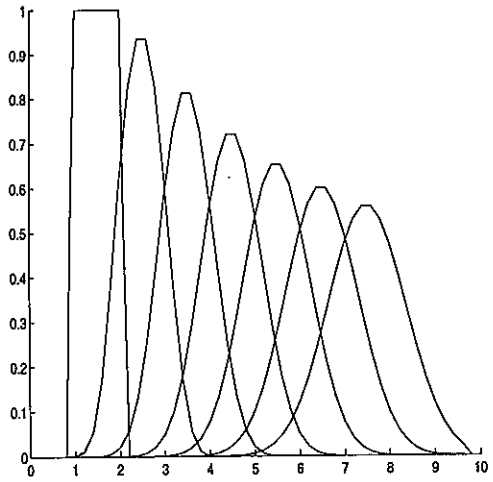

Figura 28: Resultado para o problema de advecção pura.

Observamos o efeito de difusão numérica.

Façamos agora um teste no qual a velocidade numérica é duas vezes maior do que U, para percebermos (graficamente) que neste caso ocorre um mecanismo (espúrio) de difusão numérica (figura 28). Faça outros experimentos diminuindo os espaçamentos de tal forma que a velocidade numérica não mude, e continue maior do que U. O mecanismo de difusão numérica deve diminuir. A explicação é dada em cursos mais avançados de análise numérica, usando o que se chama de *equação modificada do método numérico.* No entanto, o experimento acima ilustra o que chamamos de um *fenômeno numérico espúrio*, pois este fenômeno não é legítimo do modelo estudado. Lembre-se que nós desligamos a difusão e, no entanto, ela ainda aparece quando $\Delta x/\Delta t > U$. Por isso, a difusão que vemos é de cunho numérico.

Na figura 29 apresentamos um exemplo no qual temos um pouco de difusão: o valor adotado é $\kappa = 0,1$. Os outros dados são idênticos aos do experimento anterior. O que vemos agora é um exemplo de instabilidade numérica.

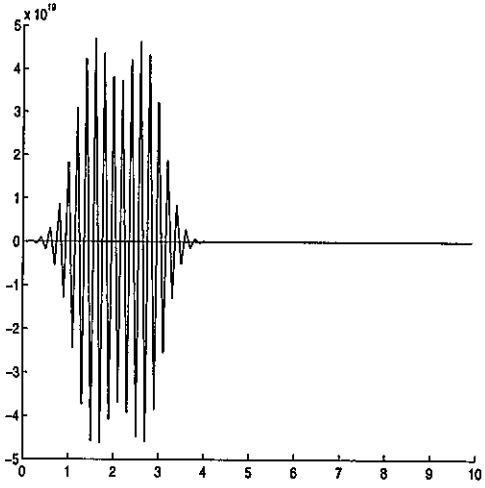

Figura 29: Resultado para o problema de advecção-difusão.
Vemos o efeito de instabilidade numérica.

O pulso quadrado foi completamente degradado pelo crescimento exponencial de componentes do erro de arredondamento. Note que no intervalo de tempo considerado, o crescimento levou alguns pontos da solução a ficarem da ordem de 10^{19}. No jargão do matemático computacional, dizemos que a solução numérica "explodiu"!

Para melhorar a solução numérica, adote um espaçamento temporal bem pequeno, a fim de eliminar o mecanismo de instabilidade. No caso em que a condição inicial é da forma $c_0(x) = \sin(kx)$, a condição de estabilidade para a difusão (i.e., $U = 0$) é dada por

$$\Delta t \leq \frac{1}{2\kappa}\left(\frac{\Delta x^2}{1-\cos(k\Delta x)}\right),$$

onde k é um número inteiro entre 0 e J/2. Note que a difusão faz com que o passo no tempo seja da ordem do quadrado do passo espacial Δx. Isso restringe bastante o passo no tempo e o algoritmo fica caro, pois temos que avançar no tempo de forma bastante lenta (i.e., com passos curtos).

A alternativa é usar um método implícito do tipo Crank-Nicolson que, pode-se mostrar, é incondicionalmente estável. Em outras palavras, temos que escolher o passo no tempo usando apenas um critério de precisão, e não um de estabilidade. O método implícito é dado por:

$$c_j^{n+1} = c_j^n - \frac{U\Delta t}{\Delta x}\left(c_j^n - c_{j-1}^n\right) + \kappa\frac{\Delta t}{\Delta x^2}\left[\frac{1}{2}\left(c_{j+1}^{n+1} - 2c_j^{n+1} + c_{j-1}^{n+1}\right) + \frac{1}{2}\left(c_{j+1}^n - 2c_j^n + c_{j-1}^n\right)\right]. \tag{2.27}$$

Note que este método é implícito e que sua molécula contém três átomos no nível de tempo $(n+1)$ e três átomos no nível n. Esse método é análogo à regra do trapézio em edo's. A cada passo de tempo temos que resolver um sistema de equações da forma

$$A\vec{c}^{\,(n+1)} = B\vec{c}^{\,(n)},$$

onde temos o vetor solução

$$\vec{c}^{\,(n+1)} = \left[c_2^{n+1} c_3^{n+1} ... c_{J-1}^{n+1}\right]^T;$$

as matrizes A e B são obtidas dos coeficientes do esquema implícito dado acima, e por isso dependem de Δx e Δt. Note que estamos levando em conta as condições de contorno homogêneas $c_1^n = c_J^n = 0$ para todo n. Escreva estas matrizes a partir da equação (2.27), colocando do lado esquerdo da equação os termos no tempo t_{n+1} e do lado direito os termos no tempo t_n. O lado direito do sistema é conhecido, supondo que já conhecemos a solução no estágio-n no tempo. Logo temos um sistema do tipo $Ax = b$, com $b = B\vec{c}^{\,(n)}$ e $J-2$ equações.

Dentro da metodologia de depuração e validação de um código, iniciamos os experimentos com o método implícito testando **exatamente** a parte nova do esquema numérico. Para isso desligamos o transporte, fazendo $U = 0$. Assim, testamos apenas a parte de difusão e, conseqüentemente, o método de Crank-Nicolson. Isto pode ser visto na figura 30, onde usamos $\kappa = 0,3$. Note que a concentração vai

baixando gradativamente mas a nuvem não se move. Note também que a nuvem vai se espalhando.

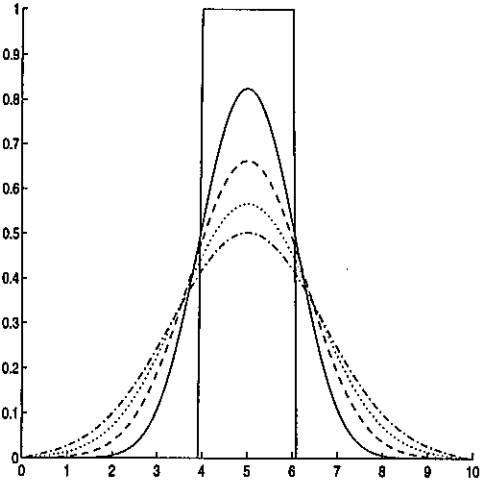

Figura 30: Resultado para o problema de difusão pura.

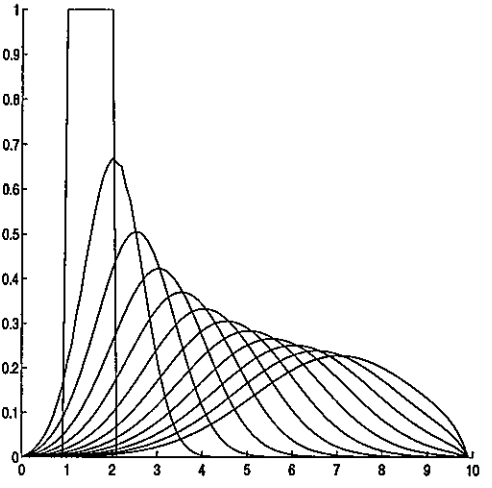

Figura 31: Resultado para o problema de advecção-difusão.

O problema acima é grande demais para caber na memória do MATLAB, versão para estudante. O tamanho máximo de um vetor é de 1024 ou então uma matriz de

32 por 32. O exemplo a seguir também será muito grande para a versão-estudante. Ele pode ser executado em um computador pessoal (PC), desde que este contenha a versão-profissional do MATLAB. Para testar os programa *advim.m* o usuário da versão-estudante deve tentar um problema de menor porte como, por exemplo, $U = 1$, $\kappa = 0,05$, 30 pontos na direção x e um comprimento total de 3.0 unidades de distância, tempo total de 1.5 unidades com $\Delta t = 0,05$ e a nuvem começando em $x = 0$ e terminando em $x = 1$.

Na figura 31 ligamos o transporte (vento) e apresentamos a evolução da solução numérica no caso em que $U = 1$, $\kappa = 0,3$, $\Delta x = 0,1$, $\Delta t = 0,05$ e o pulso quadrado inicial está situado entre $x = 1$ e $x = 2$. Conforme a onda (neste caso a nuvem, que representa uma onda de poluente) propaga para a direita vemos o efeito da difusão. A concentração máxima baixa de 1 para aproximadamente 0,25, e a nuvem ocupa uma área muito maior. Note que a concentração de poluente baixou em 75%! Agora seria o momento de consultar um especialista em engenharia ambiental para saber se essa queda de concentração do dito poluente está num nível satisfatório ou não. Na figura 32 mostramos uma

representação tridimensional da evolução da nuvem. Os eixos no plano horizontal são posição (x) e tempo (t) e o eixo vertical é concentração ($c(x,t)$). Esse gráfico foi gerado após a execução dos arquivo-m (*advim.m*; ver apêndice) com o comando MATLAB ">> mesh(d,[120,30])< R >".

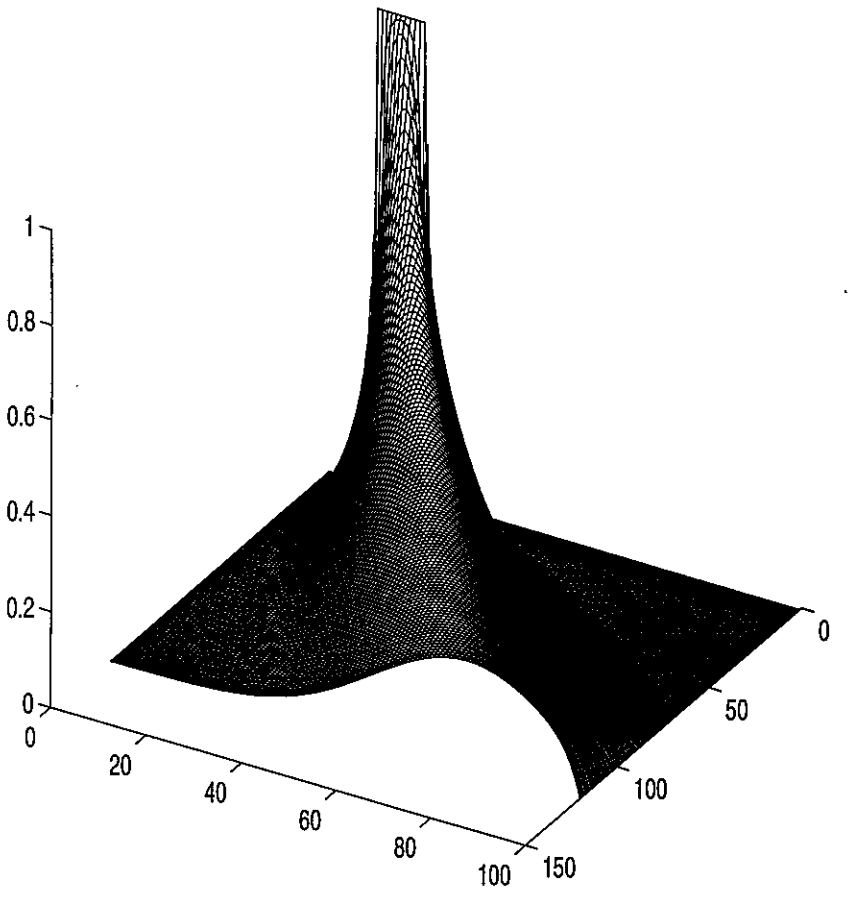

Figura 32: Reprsentação tridimensional da solução do problema de advecção-difusão. O eixo espacial x vai da esquerda para a direita enquanto que o eixo do tempo vai de dentro para fora do plano do papel.

Estamos em um ótimo ponto para discutir questões relativas à eficiência de um código. O método implícito apresentado acima (Crank-Nicolson) gera duas matrizes J–2xJ–2, consumindo assim muita memória. A versão estudantil do MATLAB comporta matrizes ("arrays") de no máximo 1024 coeficientes. Isto significa que podemos ter uma matriz de no máximo 32x32. Essa é uma ótima motivação

para utilizar o algoritmo de Thomas. Esse algoritmo leva em conta a estrutura do sistema de equações resultante do método de Crank-Nicolson e economiza bastante memória, como veremos a seguir.

Considere o método de Crank-Nicolson na forma

$$c_j^{n+1} = c_j^n - \alpha\left(c_j^n - c_{j-1}^n\right) + \beta\left(c_{j+1}^{n+1} - 2c_j^{n+1} + c_{j-1}^{n+1} + c_{j+1}^n - 2c_j^n + c_{j-1}^n\right),$$

com

$$\alpha = \frac{U\Delta t}{\Delta x}$$
$$\beta = \kappa\frac{\Delta t}{2\Delta x^2}.$$

Re-agrupamos de maneira a facilitar a notação matricial:

$$-\gamma c_{j-1}^{n+1} + c_j^{n+1} - \gamma c_{j+1}^{n+1} = \frac{(\alpha+\beta)}{1+2\beta}c_{j-1}^n + \frac{1-(\alpha+2\beta)}{1+2\beta}c_j^n + \frac{\beta}{1+2\beta}c_{j+1}^n. \quad (2.28)$$

Observe que dividimos toda a equação por $(1+2\beta)$. O novo parâmetro é definido como $\gamma = \beta/(1+2\beta)$. A cada passo de tempo, temos que resolver o sistema

$$\begin{bmatrix} 1 & -\gamma & 0 & 0 & ... & 0 \\ -\gamma & 1 & -\gamma & 0 & ... & 0 \\ 0 & -\gamma & 1 & -\gamma & ... & 0 \\ ... & ... & ... & ... & ... & ... \\ 0 & ... & ... & 0 & -\gamma & 1 \end{bmatrix} \begin{bmatrix} c_2^{n+1} \\ c_3^{n+1} \\ c_4^{n+1} \\ ... \\ c_{J-1}^{n+1} \end{bmatrix} = \begin{bmatrix} b_2^n \\ b_3^n \\ b_4^n \\ ... \\ b_{J-1}^n \end{bmatrix}. \quad (2.29)$$

O vetor independente $\vec{b}^{(n)}$ é dado pela parte à direita na equação (2.28). Este vetor é conhecido e deve ser atualizado a cada passo no tempo.

O algoritmo de Thomas consiste em fazer a eliminação de Gauss analiticamente. Como a matriz do sistema é tridiagonal e constante no tempo, podemos executar o processo de eliminação de uma vez por todas. O primeiro passo resulta em

$$\begin{bmatrix} 1 & \tilde{a}_2 & 0 & 0 & ... & 0 \\ 0 & 1 & \tilde{a}_3 & 0 & ... & 0 \\ 0 & -\gamma & 1 & -\gamma & ... & 0 \\ ... & ... & ... & ... & ... & ... \\ 0 & ... & ... & 0 & -\gamma & 1 \end{bmatrix} \begin{bmatrix} c_2^{n+1} \\ c_3^{n+1} \\ c_4^{n+1} \\ ... \\ c_{J-1}^{n+1} \end{bmatrix} = \begin{bmatrix} b_2^n \\ \tilde{b}_3^n \\ b_4^n \\ ... \\ b_{J-1}^n \end{bmatrix}.$$

onde temos os novos coeficientes $\tilde{a}_2 = -\gamma$, $\tilde{a}_3 \equiv -\gamma/(1+\gamma\tilde{a}_2)$ e $\tilde{b}_3^n \equiv (b_3^n+\gamma b_2^n)/(1+\gamma\tilde{a}_2)$.

O segundo passo, no processo de eliminação, é feito com a terceira linha do sistema:

$$\begin{bmatrix} 1 & \tilde{a}_2 & 0 & ... & 0 & \\ 0 & 1 & \tilde{a}_3 & 0 & ... & 0 \\ 0 & 0 & 1 & \tilde{a}_4 & ... & 0 \\ ... & ... & ... & ... & ... & ... \\ 0 & ... & ... & 0 & -\gamma & 1 \end{bmatrix} \begin{bmatrix} c_2^{n+1} \\ c_3^{n+1} \\ c_4^{n+1} \\ ... \\ c_{J-1}^{n+1} \end{bmatrix} = \begin{bmatrix} b_2^n \\ \tilde{b}_3^n \\ \tilde{b}_4^n \\ ... \\ b_{J-1}^n \end{bmatrix}.$$

onde $\tilde{a}_4 \equiv -\gamma/(1+\tilde{a}_3\gamma)$ e $\tilde{b}_4^n \equiv (b_4^n + \gamma b_3^n)/(1+\tilde{a}_3\gamma)$.

É fácil generalizar o processo acima e calcular um termo genérico da supra-diagonal (diagonal acima da principal)

$$\tilde{a}_2 = -\gamma$$

e

$$\tilde{a}_j = \frac{-\gamma}{1+\tilde{a}_{j-1}\gamma}, \quad j = 3, 4, ..., J-1,$$

assim como os termos do vetor independente

$$\tilde{b}_2^n = b_2^n$$

e

$$\tilde{b}_j^n = \frac{b_j^n + \gamma\tilde{b}_{j-1}^n}{1+\tilde{a}_{j-1}\gamma}.$$

A solução do sistema é imediata. Por retro-substituição obtemos

$$\vec{c}^{\,(n+1)} = \begin{bmatrix} c_2^{n+1} \\ c_3^{n+1} \\ ... \\ c_{J-2}^{n+1} \\ c_{J-1}^{n+1} \end{bmatrix} = \begin{bmatrix} \tilde{b}_2^n - \tilde{a}_2 c_3^{n+1} \\ \tilde{b}_3^n - \tilde{a}_3 c_4^{n+1} \\ ... \\ \tilde{b}_{J-2}^n - \tilde{a}_{J-2} c_{J-1}^{n+1} \\ \tilde{b}_{J-1}^n \end{bmatrix}.$$

Ao programar o algoritmo de Thomas, não precisamos montar qualquer matriz. Basta conhecer as fórmulas recursivas para os coeficientes $\tilde{a}_j$ e $\tilde{b}_j^n$ e a solução, a cada passo no tempo, é dada pelo vetor acima. Note que apenas os coeficientes $\tilde{b}_j^n$ precisam ser atualizados.

O programa crank.m usa o algoritmo de Thomas para resolver um sistema da forma mostrada em (2.29). O exemplo criado (dentro do crank.m) é tal que a solução é $[1234...N]^T$. Fica como exercício para o leitor colocar a estrutura do crank.m dentro do programa advim.m.

2.4 Modelo para enchentes em rios

2.4.1 Um modelo cinemático

Nesta seção estudaremos um modelo simples para a simulação de enchentes em rios. Vamos verificar que a estrutura matemática desses modelos tem muito em comum com o modelo de advecção e difusão de poluentes estudado na seção anterior. De fato, os dois modelos podem ser combinados para estudar a difusão de poluentes em rios.

Considere um rio semelhante ao apresentado no mapa da figura 33.

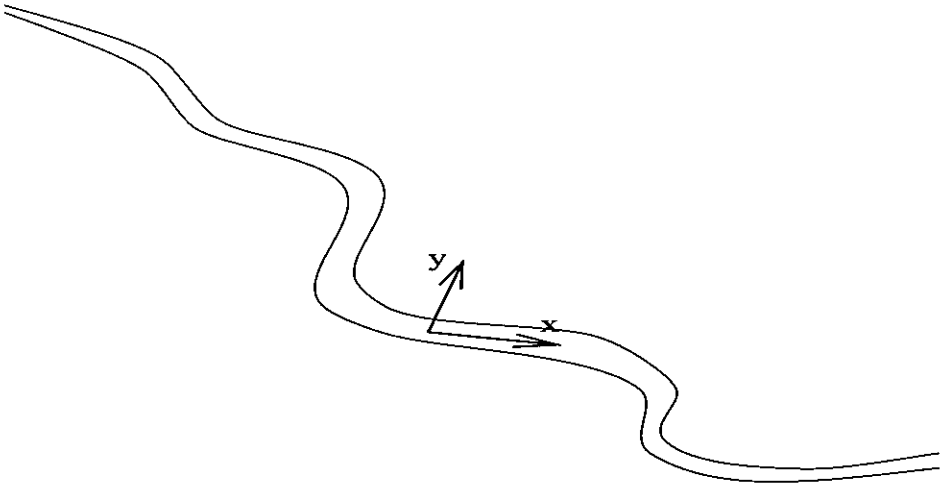

Figura 33: Vista superior do rio.

Um fato decorre simplesmente de olharmos para o mapa. Esse fato é tão óbvio que é capaz de nem nos darmos conta dele: *o rio é muito mais longo do*

que largo. Esse fato tem uma consequência muito importante no nosso processo de modelagem. Podemos modelar a dinâmica de ondas em rios sem considerar detalhes do escoamento em cada seção transversal.

Em outras palavras, podemos construir um modelo **unidimensional** que leva em conta apenas variações do escoamento médio na direção longitudinal, indicada pela variável x no mapa, desprezando as variações na direção transversal y. O que queremos dizer por escoamento médio ficará mais claro um pouco mais adiante. No entanto para dar uma breve indicação do que queremos fazer, imaginemos que em um corte transversal do rio a velocidade da água varia muito pouco. Essa é uma das hipóteses para o modelo unidimensional. Neste contexto, as velocidades em todos os pontos dessa seção não diferem muito da média em si. Então nada melhor do que trabalhar com apenas uma. Estaremos economizando esforço, pois manipularemos apenas uma grandeza em vez de várias grandezas muito parecidas, isto é, a velocidade em cada ponto desta seção. Note que as suposições feitas acima só são válidas porque estamos considerando ondas muito longas, como as provenientes de enchentes originadas a montante (na região mais alta do rio, ou seja, quando caminhamos em direção à nascente) e as geradas – digamos – por chuvas extremamente fortes. Essas ondas irão propagar em direção a cidades populosas a jusante (região mais baixa do rio, quando caminhamos na direção à foz), podendo causar graves problemas às cidades ribeirinhas. Se nós quiséssemos estudar fenômenos mais localizados, como por exemplo a erosão diferencial causada pela curva do rio, a nossa aproximação unidimensional não seria mais válida.

Sendo assim, vamos estudar grandezas médias que são funções de x, a coordenada ao longo do rio, e do tempo t. As duas grandezas médias de interesse são a altura da superfície da água $h(x,t)$ e a velocidade média do escoamento $U(x,t)$ através de uma seção transversal do rio naquele ponto x. Vamos definir duas grandezas úteis, intimamente relacionadas com as duas acima. Seja

a seção transversal de água $S(x,t)$ definida como a área molhada ao cortarmos o rio transversalmente por um plano, e o escoamento total através de uma seção transversal, por unidade de tempo, denotado por $Q(x,t)$ (figura 34). Em outras palavras, Q é a vazão, a taxa de variação do volume no tempo (por exemplo dado em m^3/s). A altura da superfície livre é determinada de forma única pela função $S(x,t)$ em cada seção, já que uma é função crescente da outra.

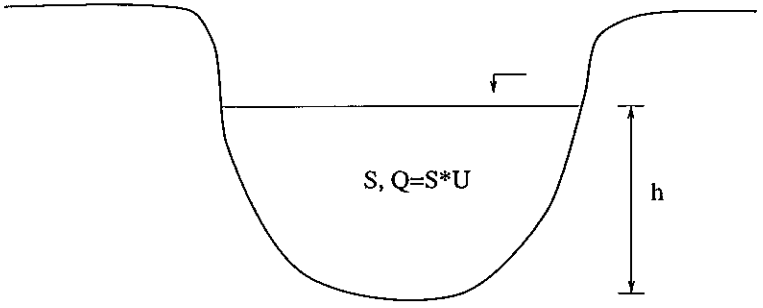

Figura 34: Seção transversal do rio.

A velocidade média U também pode ser determinada a partir de Q e S já que

$$U(x,t) = \frac{Q(x,t)}{S(x,t)}.$$

Adiante veremos que Q e S são grandezas mais adequadas ao estudo de ondas em rios do que as grandezas h e U, definidas inicialmente. Por isso vamos trabalhar com elas.

A nossa ferramenta principal de análise é a equação de conservação de massa. Consideremos um segmento do rio situado entre as seções indicadas por x_1 e x_2. Sabemos que qualquer aumento no volume V de água entre dois instantes t_1 e t_2 deve ter sido produzido pela diferença do volume total de água que entrou

pela seção em x_1 e saiu no ponto x_2 durante esse intervalo de tempo $(t_2 - t_1)$. Estamos ignorando pequenas chuvas locais, o efeito de evaporação e infiltração de água pelo solo; incorporar esses efeitos no modelo é relativamente simples. No entanto terminaríamos com um modelo mais complexo que dificultaria a análise que queremos fazer a seguir. Em termo matemáticos, o princípio da conservação de massa se escreve

$$V(t_2) - V(t_1) = \int_{t_1}^{t_2} [Q(x_1, t) - Q(x_2, t)]\,\mathrm{dt}. \tag{2.30}$$

Lembre que o volume é a integral da área da seção transversal (i.e. $S(x,t)$) entre os pontos x_1 e x_2:

$$V(t) = \int_{x_1}^{x_2} S(x,t)\mathrm{dx},$$

de onde temos em vez de (2.30),

$$\int_{x_1}^{x_2} [S(x,t_2) - S(x,t_1)]\,\mathrm{dx} + \int_{t_1}^{t_2} [\mathrm{Q}(\mathrm{x}_2, \mathrm{t}) - \mathrm{Q}(\mathrm{x}_1, \mathrm{t})]\,\mathrm{dt} = 0. \tag{2.31}$$

Do teorema fundamental do cálculo temos que para qualquer função diferenciável $F(s)$

$$F(s_2) - F(s_1) = \int_{s_1}^{s_2} \frac{\mathrm{dF}}{\mathrm{ds}}\mathrm{ds}.$$

Usemos esse teorema para escrever

$$S(x,t_2) - S(x,t_1) = \int_{t_1}^{t_2} S_t(x,t)\mathrm{dt}.$$

Podemos então transformar a equação (2.31) na forma

$$\int_{x_1}^{x_2}\int_{t_1}^{t_2} [S_t(x,t) + Q_x(x,t)]\,\mathrm{dtdx} = 0. \tag{2.32}$$

O que vem a seguir é um argumento que aparece a todo momento em matemática. Derivamos a expressão (2.32) válida para um segmento x_1-x_2 qualquer do rio. No entanto não há nada de particular, de especial, com respeito a esses pontos do rio. Por isso a expressão (2.32) tem que ser válida para quaisquer valores de

x_1 e x_2 assim como para quaisquer intervalos no tempo $[t_1, t_2]$. Note que estamos integrando sobre o retângulo no plano espaço–tempo dado por $[x_1, x_2] \times [t_1, t_2]$. Mas o valor da integral é sempre zero, independentemente do retângulo que escolhemos. Só há uma saída: o integrando tem que ser zero. Matemáticos mais rigorosos diriam que o integrando é zero em quase toda parte; mas isso é um formalismo a ser estudado com rigor em cursos mais avançados de Análise. Podemos então escrever a equação

$$S_t(x,t) + Q_x(x,t) = 0 \tag{2.33}$$

que é válida em cada seção x do rio, e em qualquer instante de tempo t. Essa equação diferencial parcial é o elemento chave no nosso modelo. Traduzindo-a para o português, ela nos diz como as variações do escoamento médio de água (i.e. Q_x) determinam (influenciam) a evolução temporal da altura de água $h(x,t)$, refletida pela variação temporal da área da seção transversal (molhada) $S(x,t)$. Essa tradução fica mais fácil quando escrevemos a equação na forma

$$S_t(x,t) = -Q_x(x,t).$$

Fica claro que se a vazão diminui ao descermos ao longo do rio (na direção x), então Q_x é negativa e S_t é positiva. Isto significa que a altura de água está aumentando. Lógico! Nesta seção do rio está entrando mais água do que saindo!

Até este ponto do processo de modelagem temos apenas uma equação (2.33) e duas variáveis dependentes S e Q, a determinar. Precisamos de uma outra relação entre S e Q. A maneira mais simples de *fechar* o sistema de equações (ou seja, de obtermos 2 equações e 2 incógnitas) é usando uma relação que os engenheiros hidráulicos tem usado através dos anos. Eles medem, nas estações localizadas ao longo do rio, a altura da água e as velocidades médias em vários instantes do tempo, ao longo de muitos anos. A conclusão é a de que valores medidos caem, aproximadamente, ao longo de uma curva conhecida como a *Lei Hidrológica* da

seção. Um caso típico é mostrado na figura 35.

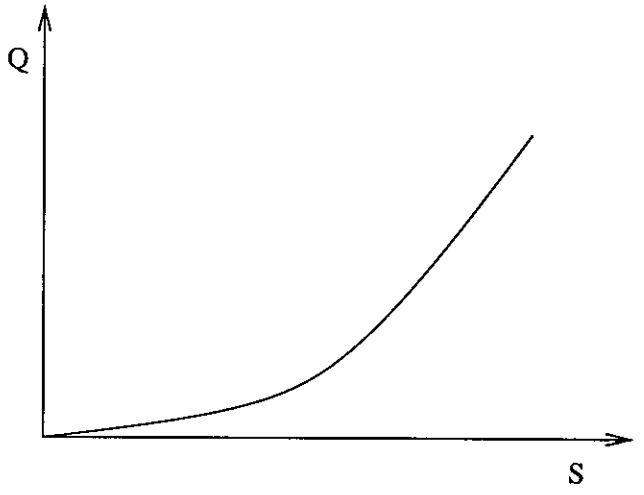

Figura 35: Lei hidrológica utilizada.

Em vez de plotar U como função de h, prefirimos usar $Q(S)$ que é muito mais útil para o nosso modelo. Dada essa curva, a equação (2.33) tem a forma

$$S_t(x,t) + Q_x(S(x,t)) = S_t(x,t) + Q'(S)S_x(x,t) = 0. \tag{2.34}$$

Esta é uma equação apenas para S. O valor do escoamento médio é obtido pela relação empírica $Q = Q(S)$, que é uma função conhecida. Como um exemplo–protótipo, considere a lei hidrológica dada pela função

$$Q = \frac{S^2}{2}.$$

Neste caso obtemos (2.34) a equação de Burgers inviscida a partir de (2.34):

$$S_t(x,t) + \frac{1}{2}\left(S^2(x,t)\right)_x = S_t(x,t) + SS_x(x,t) = 0. \tag{2.35}$$

A análise desta equação é muito semelhante à da equação de advecção da seção anterior. Note que S tem um papel análogo ao da velocidade de propagação U do problema de advecção. Convença-se de que pontos com valores constantes

de S se movem para a direita com velocidade igual a esse valor de S. A velocidade de transporte S (que faz o papel da velocidade do vento U no problema com poluentes) depende da solução. Mecanismos *ativos* de transporte, onde a velocidade de propagação depende da solução, são chamados de *mecanismos de convecção*. Temos um *mecanismo de advecção* quando o transporte é *passivo*, como no caso do poluente (o vento não quer nem saber do valor da concentração $c(x,t)$ do poluente). Devido à nossa escolha da lei hidrológica acima a velocidade de propagação depende linearmente do valor da solução. Sendo assim, pontos com valores altos de S se movem para a direita com velocidade mais alta do que pontos com valores mais baixos de S. Uma onda de enchente inicialmente suave, como por exemplo a da figura 36, vai se deformando até eventualmente quebrar. Quando isto acontece, a altura da água é dada por uma função descontínua.

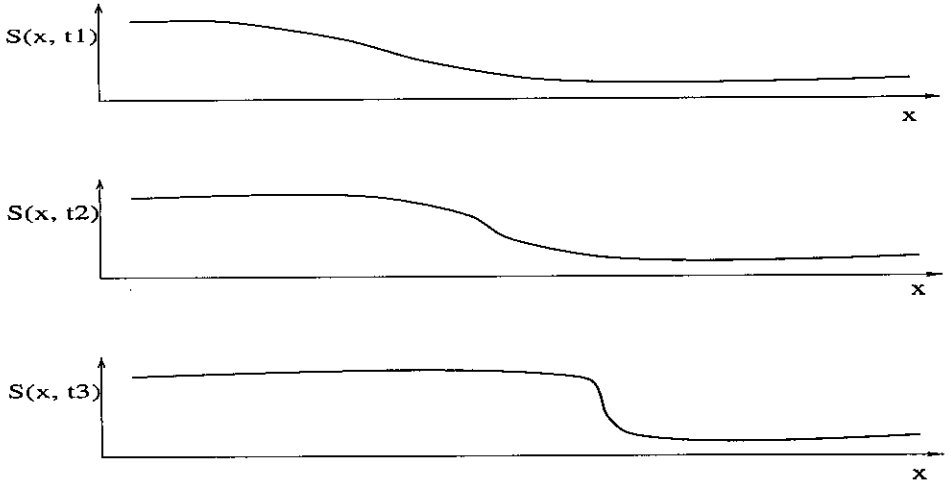

Figura 36: Situações típicas para a onda de enchente.

É como se naquele ponto tivéssemos uma parede de água propagando para a direita. Isto não é apenas um artefato do nosso modelo: muitos rios apresentam ondas de enchente aproximadamente descontínuas. Dizemos aproximadamente,

pois a inclinação é tão grande que podemos representá-la por um salto na função $h(x,t)$. É comum observar em riachos, de regiões montanhosas, a formação quase instantânea de ondas de enchente, muitas vezes com consequências trágicas para pessoas acampando perto das margens. Essas pessoas não tem o menor "aviso-prévio" de que essa parede de água vem se aproximando. De repente o nível muda!

2.4.2 Modelos numéricos

Vamos tentar resolver a equação (2.35) com o mesmo método numérico que usamos para a equação de advecção no exemplo dos poluentes. Neste contexto, o esquema numérico é colocado na forma

$$S_j^{n+1} = S_j^n - S_j^n \frac{\Delta t}{\Delta x}\left(S_j^n - S_{j-1}^n\right). \tag{2.36}$$

Essa equação de diferenças foi obtida a partir de edp avaliada no ponto x_j e no instante t_n, para os quais fizemos as aproximações para S_x e S_t. Da mesma maneira que fizemos anteriormente, precisamos fazer $\sigma = S_j^n \Delta t/\Delta x$ menor do que um, para todos os valores de j e de n. Isso significa que a velocidade numérica será maior do que a velocidade da onda do problema. Se violarmos essa condição o método será instável (experimente, por conta própria, valores grandes de Δt no programa dado abaixo e observe o que acontece!). De maneira a completar esse modelo necessitamos especificar os dados iniciais e de contorno. Consideremos um rio de comprimento $L = 5$, medido em centenas de quilômetros. Por simplicidade, vamos tomar como dado inicial

$$S(x,0) = 0,3,$$

medido em centenas de metros quadrados. Isto significa que a superfície livre é inicialmente plana e sem ondas. A montante ($x = 0$) o valor de S depende da quantidade Q de chuvas ou degelo proveniente das montanhas mais altas. Consideremos

uma situação plausível para gerar uma onda de enchente. Por exemplo, vamos supor que ocorre uma chuva torrencial por uma semana, seguida da situação média (normal) de $S = 0,3$. A função típica, modelando essa configuração a montante é expressa na forma

$$S(0,t) = \begin{cases} 0,3 & \text{para } t < 0 \\ 0,3 + 0,1\left(1 - \cos\left(\frac{2\pi}{7}t\right)\right) & \text{para } 0 < t < 7 \\ 0,3 & \text{para } t > 7 \end{cases}$$

O programa Rio1.m (apresentado no apêndice) usa o algoritmo (2.36) para calcular a evolução dessa onda de enchente. Experimente o programa! O leitor vai descobrir que são necessários muitos pontos (aproximadamente 1000) para obter resultados aparentemente satisfatórios. A explicação é simples: o rio é muito longo, as ondas são relativamente curtas, e o nosso método não é muito preciso. Na verdade, a precisão do nosso método é de primeira ordem. Note que o fato da onda ser curta significa que a superfície, em um determinado ponto, varia bruscamente. Para representar essa variação brusca, precisamos de muitos pontos nessa região. Mas como a onda está viajando rio abaixo, precisamos ser capazes de representar a variação brusca da superfície, onde quer que ela esteja. Não há outra alternativa a não ser termos muitos pontos ao longo de todo o rio! Métodos de segunda ordem, usados na prática com mais frequência, precisam de menos pontos exatamente por serem mais precisos. Mas vamos continuar trabalhando com o método de primeira ordem que, por ser mais simples, nos permite um entendimento mais fácil das questões de modelagem numérica envolvidas. Afinal, é isso que buscamos no momento, e não um código comercial de aplicação industrial!

Nos seus experimentos numéricos, o leitor deve ser capaz de ver a onda de enchente quebrar e observar uma parede quase vertical propagando rio abaixo. Matematicamente, essas descontinuidades são chamadas de "ondas de choque" ou, nesse contexto, de "salto hidráulico". O leitor deve fazer experimentos numéricos

aumentando o número de pontos e diminuindo o passo no tempo Δt, até achar que a solução numérica é satisfatória, dentro do que o seu senso crítico considerar aceitável. Isso se chama análise de resolução numérica. É uma técnica experimental, na qual vamos refinando gradativamente a malha (no espaço e no tempo) até que a solução numérica não mude praticamente nada com respeito ao experimento anterior (nessa hierarquia de experimentos). No entanto, tem uma coisa que não está bem com respeito à solução numérica: o código está "vazando" água! Para podermos visualizar isto, o programa produz um gráfico do volume total de água do rio como uma função do tempo. O volume total é, por definição, a integral de S ao longo do rio. O cálculo aproximado do volume é feito somando-se todos os S_j's vezes o espaçamento Δx. Durante a primeira semana o volume aumenta, o que é esperado devido às chuvas torrenciais. No entanto, ao término destas o volume deveria se manter constante enquanto a onda de enchente percorresse o rio inteiro, até a sua saída a jusante, provavelmente onde se encontra o delta do rio. Afinal de contas, a vazão no rio a montante é $Q(0,3)$, que deve ser exatamente a mesma quantidade saindo a jusante. No entanto, pelos resultados numéricos observamos que o volume total começa a baixar em torno do final da primeira semana. De fato, conforme veremos abaixo, o programa começa a "derramar" uma quantidade significativa de água assim que o salto hidráulico se forma. Veja a figura 37.

Ficamos então com o seguinte fato, aparentemente contraditório: como pode um modelo, baseado no princípio de conservação, perder água? Afinal de contas a equação (2.33) é obtida diretamente da equação (2.31), pela qual há conservação de água. A equação de Burgers (2.35) é uma instância particular de (2.33) e o esquema numérico (2.36) é uma discretização bem fundada de (2.35).

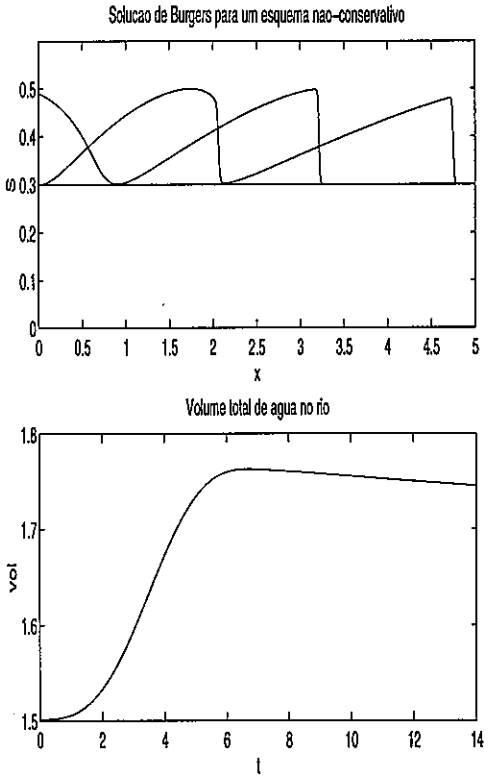

Figura 37: Solução numérica nos tempos $t = 3, 7, 10e14$.

Foram usados 1000 pontos no espaço e $\Delta t = 0,005$.

Volume em $10^7 m^3$. Tempo em dias.

Essa sensação de confusão pode ser frustrante, intolerável, mas é daí que surgem fatos interessantes em análise numérica e computação científica. A resposta para este quebra-cabeça está ligada à presença do salto hidráulico. A nossa equação diferencial só faz sentido se a solução for *suave* o suficiente para ter algumas derivadas bem definidas. Esse não é o caso quando temos o salto hidráulico. Uma vez que uma descontinuidade se forma, precisamos dar um sentido mais geral à noção de derivada, para aplicarmos às que aparecem na equação diferen-

cial. Isto pode ser feito matematicamente, usando-se o conceito de *solução fraca*. Este conceito é um dos pontos centrais de uma belíssima área de Análise. Muitos matemáticos famosíssimos trabalharam nessa área, e em especial, no esforço de se dar um sentido mais amplo, mais geral, à noção de derivada quando, por exemplo, uma função não é suave. Neste caso, o conceito clássico de *solução forte* dá lugar à noção de *solução fraca* de uma equação diferencial. A grosso modo, soluções fracas são soluções não-suaves que satisfazem a equação diferencial na média, ou seja, de acordo com uma média ponderada muito especial. Mas não vamos nos preocupar com soluções fracas! Ou melhor, nós autores vamos, mas sem dizer que estamos... O que realmente queremos é que a água seja conservada e vamos encontrar uma maneira de conseguir isso numericamente. Neste processo vamos aprender algo importante. Para isso, olhemos para a equação de Burgers (2.35). Nós escrevemos esta equação de duas maneiras, uma envolvendo $Q_x = (S^2/2)_x$, e outra (para soluções suaves) com a derivada efetuada explicitamente (SS_x). Foi esta segunda versão que nós discretizamos em (2.36). Se tivéssemos escolhido a primeira versão dada acima, o esquema numérico seria escrito na forma

$$S_j^{n+1} = S_j^n - \frac{\Delta t}{2\Delta x}\left(\left(S_j^n\right)^2 - \left(S_{j-1}^n\right)^2\right). \tag{2.37}$$

Esse esquema conserva água! Para verificar esse fato basta somar (2.37) para todos os $j's$ entre dois valores, digamos j_1 e j_2. Muitos dos termos à direita se cancelam (este é um exemplo de uma soma "telescópica"; você já deve ter visto antes ...), e obtemos a identidade

$$\sum_{j=j_1}^{j=j_2} S_j^{n+1}\Delta x - \sum_{j=j_1}^{j=j_2} S_j^n \Delta x = \frac{1}{2}\left(S_{j_1-1}^n\right)^2 \Delta t - \frac{1}{2}\left(S_{j_2}^n\right)^2 \Delta t. \tag{2.38}$$

Esta expressão pode ser re-escrita na forma

$$\sum_{j=j_1}^{j=j_2} S_j^{n+1}\Delta x = \sum_{j=j_1}^{j=j_2} S_j^n \Delta x + \Delta t\left(Q_{j_1-1}^n - Q_{j_2}^n\right). \tag{2.39}$$

Traduzindo em palavras, vemos que a variação do volume de água dentro de um intervalo $[x_{j1}, x_{j2}]$ qualquer, após um lapso de tempo Δt, é igual à diferença entre a vazão (fluxo) a montante e a vazão a jusante, vezes Δt. Quando as vazões forem iguais, por exemplo depois que parou de chover (depois da primeira semana e aproximadamente antes da terceira semana, de acordo com o nosso experimento), o volume total de água, nesse segmento do rio, não varia ao longo do tempo. O esquema (2.37) é dito estar em *forma conservativa.* O novo algoritmo está implementado no programa Rio2.m. Experimente, brinque com esse novo código. Verifique que ele não vaza, ou seja, que agora a água é conservada! Use os mesmos dados do caso não-conservativo. Os detalhes da análise apresentada acima, em especial questões ligadas à área de *Leis de Conservação*, são temas encontrados com frequência em cursos de pós-graduação em matemática aplicada (Equações Diferenciais Parciais, Análise Numérica, entre outros ...).

Agora que temos uma solução que, além de parecer correta, também não perde água, podemos tentar entender o que deu errado no caso anterior. Execute os programas Rio1.m e Rio2.m com os mesmos dados iniciais e compare as soluções numéricas no tempo, digamos, $t = 14$, no final da segunda semana. Você irá observar que as soluções são muito semelhantes. De fato, elas são indistinguíveis a menos dos saltos hidráulicos. O salto hidráulico, pelo programa Rio2.m, propaga mais rápido do que pelo Rio1.m. *O único efeito em usar o método na forma conservativa foi consertar a posição do salto hidráulico.* Mas isso faz toda a diferença do mundo: um salto propagando mais rápido tem mais água atrás dele; é essa água que estamos perdendo no primeiro experimento com Rio1.m. Coloque os gráficos resultantes dos experimentos com Rio1.m e com Rio2.m um sobre o outro, e será fácil perceber o que estamos descrevendo acima. Do ponto de vista prático a previsão da posição do salto é a coisa mais importante do modelo, pois é isso que nos diz quando a onda de enchente atingirá cada cidade ribeirinha. Nós queremos,

devemos, fazer a melhor previsão possível e por isso devemos escolher esquemas numéricos que estão na forma conservativa.

Se pararmos para pensar um pouco, vamos notar que o que acabamos de fazer nos diz algo sobre o modelo analítico (2.35) que nós não sabíamos antes. Sabíamos que, enquanto a solução era suave, esta propagaria para a direita com velocidade, em cada ponto, igual a $S(x,t)$. O que aconteceu quando a descontinuidade foi criada? A que velocidade deve a descontinuidade propagar? Agora sabemos: *a velocidade do salto é determinada pela lei de conservação.* Se nós movermos a descontinuidade (salto) com a velocidade errada, água será criada ou perdida pelo modelo numérico. Existe uma expressão analítica para esta afirmação, obtida através da teoria de soluções fracas, que implica em um conceito de extrema importância na teoria de Leis de Conservação. Esse conceito se aplica não somente a rios, mas também a escoamentos em tubulações, dinâmica de gases e escoamento no tráfego, onde carros são idealizados como partículas ao longo de um duto unidimensional (via de mão única sem ultrapassagem) (Whitham [13]).

2.4.3 Um exercício de escoamento no tráfego

Uma pequena modificação em nossa análise e código nos permite estudar escoamentos no tráfego. Devemos reinterpretar a área molhada $S(x,t)$ como a densidade de carros, ou seja, como o número de carros por quilômetro de estrada. Também precisamos redefinir a vazão $Q(x,t)$ como o fluxo de carros passando pelo ponto x por unidade de tempo. É fácil nos convencermos de que S e Q estão relacionados pela equação

$$Q(x,t) = U(x,t)\, S(x,t)\,, \tag{2.40}$$

onde $U(x,t)$ é a velocidade média dos carros na posição x no tempo t. É intuitivo que essa velocidade média é uma função decrescente de S. Em outras palavras, os motoristas tendem a dirigir mais devagar quando há mais tráfego (mais carros na

pista). Podemos modelar isto através da simples relação

$$U(S) = \begin{cases} U_{\max} & \text{para } S < S_1 \\ U_{max}\left(1 - \frac{S-S_1}{S_2-S_1}\right) & \text{para } S_1 < S < S_2 \\ 0 & \text{para } S > S_2\,, \end{cases}$$

onde $U_{\max}$ é a velocidade média de carros quando não há outros carros ou todos os carros são muito rápidos, S_1 é a densidade de carros a partir da qual a velocidade média começa a diminuir, e S_2 é a densidade para a qual os carros não se movem mais (engarrafamento total). Substituindo essa expressão em (2.40), nós obtemos uma relação entre Q e S, semelhante à lei hidrológica para os rios, mas com uma diferença de concavidade. Substituindo em (2.33), obtemos uma equação diferencial para a evolução de S.

Sugerimos que o leitor faça as modificações no programa River2.m para que ele possa ser usado no modelo de tráfego. Você irá notar que o programa só funciona quando $Q'(S)$ é positivo, i.e. para $S < S_2/2$ se $S_1 < S_2/2$, e para $S < S_1$ no outro caso. A razão para isto é que $Q'(S)$ é a velocidade de propagação de informação, e o algoritmo pressupõe que a velocidade é positiva, já que a aproximação de S_x é feita usando uma esquema retroativo no espaço. Se Q' for negativo, basta mudar o esquema para um esquema progressivo no espaço. Se Q' trocar de sinal em algumas partes, sendo ora positivo, ora negativo, o esquema numérico deve detectar isso e usar o esquema correspondente (retroativo ou progressivo) em cada posição onde isso aconteça. Fazendo isso, o leitor pode tentar usar dados correspondendo a um sinal de trânsito passando de vermelho para verde:

$$S(x,0) = \begin{cases} S_2 & \text{para } -L < x < 0 \\ 0 & \text{para } x > 0 \end{cases} \qquad S(-L,t) = S_2$$

e uma condição de contorno à direita correspondendo a um sinal vermelho:

$$S(L,t) = S_2.$$

Os resultados obtidos com o prgrama correspondem à sua experiéncia com sinais de trânsito?

Apêndice A

Arquivos-m de comandos do pacote MATLAB ("m-files")

O pacote MATLAB (que quer dizer Matrix Laboratory) nos permite escrever programas usando uma linguagem muito fácil (para quem já utiliza qualquer outra). Detalhes de programação são dados no manual do estudante (Student Edition of MATLAB). Mas o que apresentaremos abaixo já serve como uma iniciação ao uso do pacote e seus comandos. Para obter informações via internet consulte o *site* **http://www.mathworks.com**.

O MATLAB interpreta os programas como um supercomando (ou macro – um conglomerado de comandos gerando um novo e mais sofisticado comando). Isto funciona da seguinte maneira. Digitamos o comando *matlab* no diretório (em UNIX ou DOS) onde temos os nosso programas, ou seja, onde queremos trabalhar. O MATLAB imprime um texto de boas-vindas na tela e nos coloca em um ambiente próprio indicado por "≫". O MATLAB está esperando comandos! Suponha que tenhamos um programa, de comandos MATLAB, chamado de *prog1.m* . Esse arquivo foi construído usando o editor de nossa preferência. Para o MATLAB saber que esse arquivo contém um supercomando (ou programa) devemos **sempre** colocar o sufixo "m". Por isso MATLAB os chama de arquivos-m ("m-files"). Para rodar o programa basta digitar "≫ prog1" e dar um *return* (que indicaremos por

$< R >$). O MATLAB executa os comandos e cria os respectivos gráficos, quando esses são chamados pelo arquivo-m em questão. Antes de rodar um programa é bom limpar a parte gráfica com o comando $\gg clg < R >$. Vale observar que o MATLAB interpreta como comentário **tudo** que está escrito à direita do sinal % .

Neste apêndice apresentaremos **todos** os arquivos-m necessários para resolver os problemas enunciados no capítulo de aplicações.

A.1 Solução numérica de equações diferenciais ordinárias

A seguir apresentamos três arquivos-m para os métodos de Euler explícito (EuE.m), Euler implícito (EuI.m) e a regra do trapézio (Trap.m). O usuário precisa apenas digitar (por exemplo) "$\gg$ EuE$< R >$". Algumas perguntas aparecerão na tela, por exemplo pedindo informações. A cada pergunta, o usuário digita a resposta e dá um $< R >$.

A.1.1 Arquivo para o método de Euler explícito

```
%=======================================================
%     Esse arquivo deve ser guardado com o nome de EuE.m
%
% programa para executar e plotar os resultados do problema
% teste usando o metodo de Euler explicito
%=======================================================
clear
%
% leitura dos dados iniciais para um sistema de edo 2x2
num_y(1,1) = input('Digite o dado inicial y1_0 =  ');
num_y(1,2) = input('Digite o dado inicial y2_0 =  ');
% Intervalo de tempo usado [ti,tf]
ti = input('Tempo inicial =   ');
tf = input('Tempo final =   ');
nt = input('numero de passos no tempo  =  ');
dt=(tf-ti)/nt;
% A matriz A (constante) do sistema de equacoes y' = A y
A(1,1) = input('coeficiente A11 da matriz =  ');
A(1,2) = input('coeficiente A12 da matriz =  ');
A(2,1) = input('coeficiente A21 da matriz =  ');
A(2,2) = input('coeficiente A22 da matriz =  ');
eig(A)
```

```
B1= eye(2) + dt * A;

    x(1)=ti;
for i=2:nt+1,
    num_y(i,1:2) = num_y(i-1,1:2) * B1';
    x(i)=ti+(i-1)*dt;
end

subplot(211), plot(x,num_y(:,1)),...
title('y1(t)'),xlabel('t'),grid,...
subplot(212),plot(x,num_y(:,2)),...
title('y2(t)'),xlabel('t'),grid
```

A.1.2 Arquivo para o método de Euler implícito

```
%=========================================================
%     Esse arquivo deve ser guardado com o nome de EuI.m
%
% programa para executar e plotar os resultados do problema
% teste usando o metodo de Euler implicito
%=========================================================
clear
%
% leitura dos dados iniciais para um sistema de edo 2x2
num_y(1,1) = input('Digite o dado inicial y1_0 =  ');
num_y(1,2) = input('Digite o dado inicial y2_0 =  ');
% Intervalo de tempo usado [ti,tf]
ti = input('Tempo inicial =   ');
tf = input('Tempo final =   ');
nt = input('numero de passos no tempo  =  ');
dt=(tf-ti)/nt;
% A matriz A (constante) do sistema de equacoes y' = A y
A(1,1) = input('coeficiente A11 da matriz =  ');
A(1,2) = input('coeficiente A12 da matriz =  ');
A(2,1) = input('coeficiente A21 da matriz =  ');
A(2,2) = input('coeficiente A22 da matriz =  ');
eig(A)
B = eye(2) - dt * A;
B1 = inv(B)

    x(1)=ti;
for i=2:nt+1,
    num_y(i,1:2) = num_y(i-1,1:2) * B1';
    x(i)=ti+(i-1)*dt;
end

subplot(211), plot(x,num_y(:,1)),...
title('y1(t)'),xlabel('t'),grid,...
subplot(212),plot(x,num_y(:,2)),...
title('y2(t)'),xlabel('t'),grid
```

A.1.3 Arquivo para a regra do trapézio

```
%=======================================================
%     Esse arquivo deve ser guardado com o nome de Trap.m
%
% programa para executar e plotar os resultados do problema
% teste
%=======================================================
clear
%
% leitura dos dados iniciais para um sistema de edo 2x2
num_y(1,1) = input('Digite o dado inicial y1_0 =  ');
num_y(1,2) = input('Digite o dado inicial y2_0 =  ');
% Intervalo de tempo usado [ti,tf]
ti = input('Tempo inicial =   ');
tf = input('Tempo final =   ');
nt = input('numero de passos no tempo  =  ');
dt=(tf-ti)/nt;
% A matriz A (constante) do sistema de equacoes y' = A y
A(1,1) = input('coeficiente A11 da matriz =  ');
A(1,2) = input('coeficiente A12 da matriz =  ');
A(2,1) = input('coeficiente A21 da matriz =  ');
A(2,2) = input('coeficiente A22 da matriz =  ');
eig(A)
B = eye(2) - 0.5* dt * A;
C = eye(2) + 0.5* dt * A;
B1 = inv(B)*C;

    x(1)=ti;
for i=2:nt+1,
    num_y(i,1:2) = num_y(i-1,1:2) * B1';
    x(i)=ti+(i-1)*dt;
end

subplot(211), plot(x,num_y(:,1)),...
title('y1(t)'),xlabel('t'),grid,...
subplot(212),plot(x,num_y(:,2)),...
title('y2(t)'),xlabel('t'),grid
```

A.2 Dinâmica de populações

O primeiro arquivo-m (chamado de *runpop.m*) contém os comandos para executar o esquema numérico e a geração de gráficos no problema de dinâmica de populações. Este comando (*runpop*) automaticamente aciona outro supercomando (o arquivo-m *pop.m*) que contém o campo vetorial do sistema de edo's apresentado no capítulo 2. O usuário precisa apenas digitar "$\gg$ runpop$<$ R $>$". Algumas perguntas

aparecerão na tela, por exemplo pedindo os dados iniciais. Para cada pergunta o usuário digita a resposta e dá um $< R >$.

A.2.1 Arquivo para calcular e plotar os resultados

```
%=======================================================
%     Esse arquivo deve ser guardado com o nome de runpop.m
%
% programa para executar e plotar os resultados do problema
%                        predador-presa
%=======================================================
%
% leitura dos dados iniciais para um sistema de edo 2x2
initial(1) = input('Digite a populacao inicial da presa');
initial(2) = input('Digite a populacao inicial do predador ');
% Intervalo de tempo usado [ti,tf]
ti = input('Tempo inicial ');
tf = input('Tempo final ');
% Constantes referentes a presa e ao predador
k1 = 3
k2 = 0.002
k3 = 0.0006
k4 = 0.5
% ode23 = rotina do MATLAB para resolucao de edo's
[x, num_y] = ode23('pop',ti,tf,initial);
subplot(211), plot(x,num_y(:,1)),...
title('Populacao da Presa'),xlabel('x'),grid,...
subplot(212),plot(x,num_y(:,2)),...
title('Populacao do Predador'),xlabel('x'),grid
```

A.2.2 Arquivo com o campo vetorial da respectiva edo

```
%=============================================================
%      Esse arquivo deve ser guardado com o nome de pop.m
%
% campo vetorial para o problema predador-presa
%=============================================================
function xf = pop (t,x)
global k1 k2 k3 k4
xf(1)= k1*x(1)-k2*x(1)*x(2);
xf(2)= k3*x(1)*x(2)-k4*x(2);
```

A.3 Arquivos para o modelo das cigarras

O programa abaixo resolve o problema (edo) para um oscilador linear forçado:

A.3.1 Programa para o oscilador forçado

```
% Esse programa resolve a equacao do oscilador forcado

%  X_tt + X = f(t), X(0)=X0, X_t(0)=Y0

% usando o metodo do previsor-corretor. Primeiro a equacao
% e reescrita na forma de um sistema de primeira ordem:

%  X_t = Y
%  Y_t = -X + forca(t).

% Em um instante t, os valores de X(t) e Y(t), chamados de X0 e Y0,
% sao usados para computar a primeira estimativa das derivadas no tempo
% X0_t e Y0_t. Com isso um chute inicial para X(t+dt) e Y(t+dt),
% aqui chamados de X1 e Y!, e obtido atraves de

%  X1 = X0 + dt*X0_t
%  Y1 = Y0 + dt*Y0_t.

% De posse de X1, Y1 e t+dt, uma nova estimativa para as derivadas
% X1_t and Y1_t e calculada. Finalmente, a solucao aproximada, no tempo
% t+dt, e obtida atraves de

%  X(t+dt) = X(t) + dt/2 * (X0_t + X1_t)
%  Y(t+dt) = Y(t) + dt/2 * (Y0_t + Y1_t)

% t0, tf, dt, t, u (variavel grafica):

t0=0; tf=150; dt=0.1; t=[t0:dt:tf]; u=zeros(size(t)); nt=size(t,2);

% Dados inciais:

X0=0; Y0=0;

% Periodo do forcante:

P = 13.0;

u(1)=X0;

% Loop principal (no tempo):

for j=1:nt-1,

  X0t=Y0;
  Y0t=-X0+forca(t(j),P);

  X1=X0+dt*X0t;
  Y1=Y0+dt*Y0t;

  X1t=Y1;
  Y1t=-X1+forca(t(j+1),P);
```

```
  X0=X0+0.5*dt*(X0t+X1t);
  Y0=Y0+0.5*dt*(Y0t+Y1t);

  u(j+1)=X0;

end

% Grafico dos resultados:

plot(t,u); xlabel('t'); ylabel('x');
```

O arquivo-m abaixo contém o programa para calcular o forçante em questão:

```
function [f] = forca(tt,P)

% Forcante periodico com periodo P.

ttt=tt;
while ttt > P,
  ttt=ttt-P;
end

f=(ttt<(P/8));
```

A.3.2 Programa para o modelo das cigarras e pássaros

```
% Cigarras e predadores (passaros)
% Porque as cigarras emergem todas de uma vez?
% Porque a intervalos primos, em numero de anos?

% nc      Numero de especies de cigarras
% x(j,k) Populacao de cigarras da especie k com idade j.
% nx(k)  Expectativa de vida das cigarras de cada especie k.

% np      Numero de especies de predadores.
% y(j,k) Populacao de predadores da especie k com idade j.
% ny(k)  Expectativa de vida dos predadores da especie k.

% Inicializacao:

clear nx ny x y x2 y2 xp yp
clear ax rx dxy ay0 ay1 ry0 ry1

nc=3;
nx=[12 13 15];

x=zeros(max(nx),nc);
```

```
np=3;
ny=[2 3 5];

y=zeros(max(ny),np);

% Parametros:

% K       Capacidade do solo em conter ninfas.
% ax      Taxa de sobrevivencia anual das cigarras abaixo da superficie.
% rx      Numero de crias por cigarra.
% dxy     Numero de cigarras comidas pelos predadores.

% ay=ay0+ay1*xn     Taxa de sobrevivencia anual dos predadores.
% ry=ry0+ry1*xn     Numero de crias  por predador.
%             xn:   Numero total de cigarras adultas.

K=sum(nx);
Nxmin=min(nx);
Nymin=min(ny);
Nymax=max(ny);

% cr Quanto a mais os predadores comem antes de se reproduzirem.

cr=4;
cr=cr-1;

% Valor aproximado da esperanca de xn e yt:

xnexp=(K*0.9^Nxmin)/Nxmin;
ytexp=(Nymin+cr)*np/Nymin;

% Inicializacao como populacoes aleatorias de cigarras e passaros,
% com parametros escolhidos de maneira a (aproximadamente) nao favorecer
% nenhuma especie.

for k=1:nc,
  xmin(k)=0.1;
  for j=1:nx(k),
    x(j,k)=Nxmin*rand/nx(k);
  end
  x(1,k)=max(x(1,k),xmin(k));
  ax(k)=1-0.1*Nxmin/nx(k);
  dxy(k)=0.01;
  rx(k)=ax(k)^(-nx(k))+dxy(k)*nx(k)*ytexp/(Nxmin*xnexp);;
end

for k=1:np
  ymin(k)=0.1;
  for j=1:ny(k),
    y(j,k)=Nymin*rand/ny(k);
  end
  y(1,k)=max(y(1,k),ymin(k));
  ay0(k)=0.75;
```

```
  ay1(k)=0.025;
  ry1(k)=0.3*ny(k)/(Nymax*xnexp);
  ry0(k)=(ay0(k)+ay1(k)*xnexp)^(-ny(k))-ry1(k)*xnexp;
end

eps=0.00000001; % Um valor pequeno, para evitar divisao por zero.

% Ultimo ano considerado:

tf=10000;

xp=zeros(1:tf,1:nc); % Variaveis para fazer graficos
yp=zeros(1:tf,1:np);

% Ciclo (loop) principal;

for t=1:tf,

%   yt: Numero total de passaros, ponderado pelo quanto eles comem.

    yt=sum(sum(y));
    for k=1:np,
      yt=yt+cr*y(ny(k),k);
    end

%   xn: numero total de cigarras adultas.

    xn=0;
    for k=1:nc,
      xn=xn+x(nx(k),k);
    end

% Atualizacao das populacoes de cigarras

  for k=1:nc,

%   xy * x(nx(k),k) mede a quantidade de cigarras da
%                   especie k que foram comidas pelos passaros.
%                   Esse numero e proporcional a yt, o numero total
%                   de passaros, e a x(nx(k),k)/xn,  a proporcao de
%                   cigarras adultas da especies k com respeito a
%                   populacao total de cigarras adultas.

    xy=dxy(k)*yt/(xn+eps);

    x2(2:nx(k),k)=ax(k)*x(1:nx(k)-1,k);

    x2(1,k)=max((rx(k)*ax(k)-xy)*x(nx(k),k),xmin(k));

  end

% Numero de novas  ninfas que cabem sob a terra
% Kt e o numero total de cigarras; dessa apenas
```

```
% K podem ficar.

  Kt=sum(sum(x2));

  if(Kt > K)
    K1=sum(x2(1,:));
    Kavail=K-(Kt-K1);
    prop=Kavail/K1;
    x2(1,:)=prop*x2(1,:);
  end

% Atualizacao da populacao de predadores;
% A sua taxa de sobrevivencia ay depende do
% numero total de cigarras adultas xn.

  for k=1:np,

    ay=min(ay0(k)+ay1(k)*xn,1);
    ry=ry0(k)+ry1(k)*xn;

    y2(2:ny(k),k)=ay*y(1:ny(k)-1,k);
    y2(1,k)=max(ry*ay*y(ny(k),k),ymin(k));

  end

  x=x2;
  y=y2;

% Armazenamento:

  xp(t,1:nc)=x(1,1:nc);
  yp(t,1:np)=y(1,1:np);

end

subplot(321)
plot(xp(:,1),'.'); xlabel('t'); ylabel('cigarras 12-anos');
subplot(323)
plot(xp(:,2),'.'); xlabel('t'); ylabel('cigarras 13-anos');
subplot(325)
plot(xp(:,3),'.'); xlabel('t'); ylabel('cigarras 15-anos');

subplot(322)
plot(yp(:,1),'.'); xlabel('t'); ylabel('passaros 2-anos');
subplot(324)
plot(yp(:,2),'.'); xlabel('t'); ylabel('passaros 3-anos');
subplot(326)
plot(yp(:,3),'.'); xlabel('t'); ylabel('passaros 5-anos');
```

A.4 Dispersão de poluentes

Os próximos 2 arquivos-m (chamados de *advex.m* e *advim.m*) contém os comandos para executar os esquemas numéricos (explícito e implícito, respectivamente) e a geração de gráficos no problema de dispersão de poluentes. O usuário precisa apenas digitar ">> advex< R >", para o esquema explícito e "≫ advim< R >" para o esquema implícito. Algumas perguntas aparecerão na tela, pedindo os coeficientes de advecção (velocidade U), o coeficiente de difusão κ, o número de pontos J para discretizar o intervalo de interesse (espacial), o final desse intervalo (que supomos começar em $x = 0$), o final do intervalo no tempo (que supomos começar em $t = 0$), o espaçamento no tempo (Δt), a posição do começo e do fim da nuvem de poluente (e.g. $x = 1$ e $x = 2$; basta digitar o número) e finalmente de quantos em quantos intervalos Δt devemos imprimir a solução.

A.4.1 Método explícito para a equação de advecção-difusão

```
%=====================================================================
% Programa para o MatLab resolver a equacao de adveccao-difusao usando
% um metodo explicito
%=====================================================================

clg
clear c;
clear xj;
U=input('Entre com a velocidade ');
K=input('Entre com a constante de difusao ');

N=input('Entre com o numero de pontos em x ');
xf=input('Entre com final do intervalo ');
tf=input('Entre com o final do intervalo de tempo ');
dt=input('Entre com o intervalo de tempo dt ');
a=input('Entre com a posicao inicial da nuvem de poluente ');
b=input('Entre com a posicao final da nuvem de poluente ');
iprint=input('Voce quer imprimir a solucao de quantos em quantos dt's? ');

dx=xf/N;        % espacamento em x
icount = 0;     % parametro auxiliar no controle de impressao dos resultados
icol = 0;       % parametro auxiliar para contar colunas na impressao

T=floor(tf/dt); % numero de passos no tempo usados
TT=floor(T/iprint) - 1; % numero de vezes a solucao sera impressa
for ii=1:N;
```

```
    c1(ii)=0;  % para dimensionar o vetor c1 = solucao do problema em t_n
    c2(ii)=0.;  % para dimensionar o vetor c2 = solucao do problema em t_n+1
end
%d=zeros(N,TT);  % para dimensionar a matriz auxiliar d = contem resultados

% construcao dos dados iniciais (onda de poluente retangular)
%-----------------------------------------------------------------
for j=1:N,
xj(j)=dx*(j-1);
if xj(j)>=a & xj(j)<= b
   c1(j)=1.;
end
end
hold
plot(xj,c1)

% condicao de contorno
%-----------------------------------------------------------------
c1(1)=0;
c1(N)=0;
c2(1)=0;
c2(N)=0;


% evolucao no tempo
%=================================================================
for i= 1:T-1,


for j=2:N-1,

c2(j)=c1(j)-(U*dt/dx)*(c1(j)-c1(j-1)) + (K*dt/(dx)^2)*(c1(j+1)-2*c1(j)+c1(j-1));

        end
%       controle de impressao dos resultados
%       ------------------------------------
        icount = icount + 1;
        if icount == iprint
           plot(xj,c2)
           icount = 0;
        end
        c1=c2;
end
%=================================================================

% impressao dos resultados via a matriz auxiliar d
%-----------------------------------------------------------------
%plot(xj,d)
```

A.4.2 Método implícito para a equação de advecção-difusão

```
%=====================================================================
%       Esse arquivo deve ser guardado com o nome de advim.m
```

```
%
% Programa para o MatLab resolver a equacao de adveccao-difusão usando
% um metodo implicito
%=========================================================================
clg;
clear c;
clear xj;
U=input('Entre com a velocidade ');
K=input('Entre com a constante de difusao ');

N=input('Entre com o numero de pontos em x ');
xf=input('Entre com final do intervalo ');
tf=input('Entre com o final do intervalo de tempo ');
dt=input('Entre com o intervalo de tempo dt ');
a=input('Entre com a posicao inicial da nuvem de poluente ');
b=input('Entre com a posicao final da nuvem de poluente ');
iprint=input('Voce quer imprimir a solucao de quantos em quantos dt's? ');

dx=xf/N;          % espacamento em x
icount = 0;       % parametro auxiliar no controle de impressao dos resultados
icol = 0;         % parametro auxiliar para contar colunas na impressao

T=floor(tf/dt); % numero de passos no tempo usados
TT=floor(T/iprint) - 1; % numero de vezes a solucao sera impressa
c=zeros(N,TT);  % para dimensionar a matriz c = solucao do problema
d=zeros(N,TT);  % para dimensionar a matriz auxiliar d = contem resultados

% construcao dos dados iniciais (onda de poluente retangular)
%-------------------------------------------------------------------
for j=1:N,
xj(j)=dx*(j-1);
if xj(j)>=a & xj(j)<= b
   c(j,1)=1.;
end
end
hold
plot(xj,c(:,1))

% condicao de contorno
%-------------------------------------------------------------------
for i=1:T;
c(1,i)=0;
c(N,i)=0;
end


% evolucao no tempo
%=================================================================
alpha=K*dt/(dx*dx);
% o comando abaixo deve ser digitado na mesma linha
A=diag((-alpha/2)*ones(N-1,1),-1)+ diag((1+alpha)*ones((N),1),0)
+diag((-alpha/2)*ones(N-1,1),1);
```

```
A(1,1)=1;
A(N,N)=1;
A(2,1)=0;
A(N-1,N)=0;
A(1,2)=0;
A(N,N-1)=0;

% o comando abaixo deve ser digitado na mesma linha
B=diag((alpha/2+U*dt/dx)*ones(N-1,1),-1)+diag((1-alpha-U*dt/dx)*
ones((N),1),0)+diag((alpha/2)*ones(N-1,1),1);

B(1,1)=1;
B(N,N)=1;
B(2,1)=0;
B(N-1,N)=0;
B(1,2)=0;
B(N,N-1)=0;

A1=inv(A)*B;

for i= 1:T-1,
        c(:,i+1)=A1*c(:,i);

%       controle de impressao dos resultados
%       -----------------------------------
        icount = icount + 1;
        if icount == iprint
           icol = icol + 1;
           d(:,icol)=c(:,i+1);
        icount = 0;
end
end
%================================================================

% impressao dos resultados via a matriz auxiliar d
%-------------------------------------------------
plot(xj,d)
```

A seguir apresentamos o arquivo-m com o algoritmo de Thomas.

A.4.3 Arquivo para o algoritmo de Thomas

```
clear
n=input('numero de equacoes  ');
gamma=input('parametro fora da diagonal  ');
A=eye(n);
%
for i=2:n,
    A(i,i-1)=-gamma;
    A(i-1,i)=-gamma;
end
```

```
aux=ones(n,1);
for i=1:n,
    aux(i)=i*aux(i);
end
b=A*aux;
%
x=A\b
%
% Versao eficiente: algoritmo de Thomas (Ames pag. 62)
%
a=-gamma*ones(n,1);
for i=2:n-1
    a(i)=-gamma/(1+gamma*a(i-1));
    b(i)=(b(i)+gamma*b(i-1))/(1+gamma*a(i-1));
end
    b(n)=(b(n)+gamma*b(n-1))/(1+gamma*a(n-1));
%
% Solucao do sistema por retro-substituicao
%
x(n)=b(n);
for i=n-1:1:1,
    x(i)=b(i)-a(i)*x(i+1);
end
x
```

A.5 Arquivos para o problema da enchente no rio

A seguir apresentamos os dois programas usados nos experimentos para simular a propagação de uma onda de enchente em um rio.

A.5.1 Método não-conservativo

```
% Rio1.m

% Esse programa resolve a edp S_t + (S^2/2)_x = 0, com
% uma diferenciacao retroativa (Euler) no espaco e outra
% progressiva no tempo

clear Ssum ts

L = 5;  % Comprimento do Rio.

N = input('Numero de pontos ao longo do Rio? ');

dt = input('dt? ');

tt = input('Tempos a serem plotados? Escreva entre colchetes, ex: [3 4 8] ');
```

```
nt=size(tt,2)+1;

tt = [0 tt];  % Aqui estamos incluindo t=0 como o primeiro tempo plotado.

% Inicializacao:

dx = L/(N-1);

x = [0:dx:L]; % Grade espacial

S = 0.3*ones(size(x)) ; % Valores iniciais de S;
%                         um valor 0.3 uniforme nesse caso.

subplot(211)

%  Plot para t= 0

plot(x,S)
axis([0 L 0 0.6]);
hold on

c=dt/dx;

jt=1;

for it = 1:nt-1

% Loop entre os plots;

   for t = tt(it):dt:tt(it+1)

       S(2:N)=S(2:N)-c*S(2:N).*(S(2:N)-S(1:N-1));

       S(1) = 0.3 + 0.1*(t<7)*(1-cos(2*pi*t/7)); % dados a montante

       ts(jt)=t;
       Ssum(jt)=dx*sum(S);  % Volume total de agua no rio.
       jt=jt+1;

   end

%  Grafico para t=tt(it+1)

   plot(x,S)

end

xlabel('x')
ylabel('S')

title('Solucao de Burgers para um esquema nao-conservativo')

hold off
```

```
subplot(212)
plot(ts,Ssum)
xlabel('t')
ylabel('vol')
title('Volume total de agua no rio')
```

A.5.2 Método conservativo

```
% Rio2.m

% Esse programa resolve a edp S_t + (S^2/2)_x = 0 com um esquema de
% Euler retroativo e conservativo no espaco, e progressivo no tempo.

clear Ssum ts

L = 5;  % Comprimento do rio..

N = input('Numero de pontos ao longo do rio? ');

dt = input('dt? ');

tt = input('Tempos a serem plotados? Escreva entre colchetes, ex: [3 4 8] ');

nt=size(tt,2)+1;

tt = [0 tt];  %  Aqui estamos incluindo t=0 como o primeiro tempo plotado.


% Inicializacao:

dx = L/(N-1);

x = [0:dx:L]; % Grade espacial.

S = 0.3*ones(size(x)) ; % Valores iniciais de S;
%                         um valor 0.3 uniforme nesse caso.


subplot(211)

%  Plot para t= 0

plot(x,S)
axis([0 L 0 0.6]);
hold on

c=0.5*dt/dx;

jt=1;

for it = 1:nt-1
```

```
% Loop entre os plots:

   for t = tt(it):dt:tt(it+1)

       S(2:N)=S(2:N)-c*(S(2:N).^2-S(1:N-1).^2);

       S(1) = 0.3 + 0.1*(t<7)*(1-cos(2*pi*t/7)); % Upstream data.

       ts(jt)=t;
       Ssum(jt)=dx*sum(S);  % Total volume of water in the river.
       jt=jt+1;

   end

%  Graficos para t=tt(it+1)

   plot(x,S)

end

xlabel('x')
ylabel('S')

title('Solucao de Burgers para um esquema conservativo')

hold off

subplot(212)
plot(ts,Ssum)
xlabel('t')
ylabel('vol')
title('Volume total de agua no rio')
```

Bibliografia

[1] Ames, W.F. (1992), *Numerical Methods for Partial Differential Equations*, Academic Press.

[2] Burden, R.L. and Faires, J.D. (1993), *Numerical Analysis*, 5th. ed., PWS-Kent Publishing Company.

[3] Etter, D. (1993), *Engineering Problem Solving with MATLAB*, Prentice Hall.

[4] Friedman, A., and Littman, W. (1994), *Industrial Mathematics, A Course in Solving Real-World Problems*, SIAM.

[5] Golub, G. and Ortega, J. (1992), *Scientific Computing and Differential Equations (An Introduction to Numerical Methods)*, Academis Press.

[6] Gould, S. Jay, (1977), Of bamboos, cicadas and the economy of Adam Smith, in *Ever Since Darwin.*

[7] Hirsh, M.W. and Smale, S. (1974), *Differential Equations, Dynamical Systems and Linear Algebra,* Academic Press.

[8] Hoppensteadt, F.C. and Keller, J.N. (1976), Synchronization of periodical cicada emergences, *Science* **194**, 335-337.

[9] Lambert, J.D. (1991), *Numerical Methods for Ordinary Differential Systems,* John Wiley.

[10] *The Student Edition of MATLAB* (1995), version 4 for Microsoft Windows, the (book/disk package) 1/e, Prentice Hall.

[11] May, R.M., (1979), Periodic cicadas, *Nature* **277**, 347-349.

[12] Murray, J.D. (1989) *Mathematical Biology*, Springer-Verlag, 100-106.

[13] Whitham, G.B., (1974) *Linear and Nonlinear Waves*, Wiley-Interscience.

Impresso na Gráfica do

impa

pelo Sistema Xerox / 5390