SME0306 - 2019

Gustavo C. Buscaglia

ICMC - Sala 4-219, Ramal 738176, gustavo.buscaglia@gmail.com

http://www.lcad.icmc.usp.br/~buscaglia/teaching/sme0306_aeronautica2019/index.html

---

Métodos Numéricos e Computacionais II - Engenharia Aeronáutica

Ementa (Júpiter): Métodos numéricos para a determinação de zeros de funções: método da bisecção e Newton para equações em uma variável. Método de Newton em várias variáveis. Introdução à otimização. Método do Gradiente. Aplicação do método de Newton. Método dos mínimos quadrados. Interpolação. Transformada de Fourier.Diferenciação e integração numérica. Sistemas de equações diferenciais ordinárias: Métodos numéricos de Taylor e Runge-Kutta, métodos implícitos, previsor-corretor. Problemas de valor contorno: método de shooting, resolução por transformada rápida de Fourier. Aplicação da linguagem de programação (SCILAB ou OCTAVE) na solução de problemas de cálculo numérico.

Programa (estimado):

1. Introdução e Octave. 29/07, 30/07, 05/08, 06/08
2. Solução numérica de equações não lineares. 12/08, 13/08, 19/08
3. Introdução à otimização. 20/08, 26/08
4. **Prova 1**. 27/08
5. Semana da Pátria. 02/09, 03/09
6. Interpolação. 09/09, 10/09, 16/09, 17/09, 23/09
7. **Prova 2**. 24/09
8. Mínimos quadrados. 01/10, 07/10, 08/10
9. **Prova 3**. 14/10
10. Diferenciação e integração numérica. 15/10, 21/10, 22/10
11. Feriado. 28/10
12. **Prova 4**. 29/10
13. Revisão, projetos. 4/11, 5/11, 11/11, 12/11
14. Solução numérica de EDOs. Fourier. 18/11, 19/11, 25/11, 26/11
15. **Prova 5**. 02/12
16. **Prova sub**. 03/12

Programa realizado:

1. Introdução e Octave. 29/07, 30/07, 05/08, 06/08
2. Solução numérica de equações não lineares. 12/08, 13/08, 19/08
3. Introdução à otimização. 20/08, 26/08
4. **Prova 1**. 30/08
5. Semana da Pátria. 02/09, 03/09
6. Interpolação. 09/09, 10/09, 16/09, 17/09
7. **Prova 2**. 23/09
8. Mínimos quadrados. 01/10, 08/10, 14/10, 15/10
9. **Prova 3**. 21/10
10. Projeto. 22/10
11. Feriado. 28/10
12. **Prova 4, apresentação de projetos**. 29/10
13. Diferenciação e integração numérica. 4/11, 5/11, 12/11
14. Solução numérica de EDOs. Fourier. 18/11, 19/11, 25/11, 26/11
15. **Prova 5**. 02/12
16. **Prova sub**. 03/12

Mecanismos de avaliação:

- **5** Provas escritas **objetivas**. Necessário trazer calculadora.
- Uma prova **substitutiva** na última semana, com o mesmo sistema.
- A média de provas se calcula tirando a média das provas do semestre com a nota da sub (se a média do semestre for maior que a nota da sub, fica a média do semestre). Para passar, a média de provas obtida dessa maneira deve superar 4.9.
- **Bonus sub:** Aqueles alunos cuja média do semestre seja superior a 4.9 são incentivados a fazer a prova sub com um bonus de 1 ponto na média final, apenas sob a condição de tirar 5 ou mais na sub.
- **Bonus supervivência:** Os alunos cuja média de provas seja superior a 4.9 obterão um bonus por terem sobrevivido e não precisarem ser ”recuperados”. Para esse bonus não é condição tirar 5 ou mais na sub (nem sequer precisa fazé-la). O valor do bonus será de 1 ponto.

Requerimento de tempo: Essa disciplina deverá requerir apenas o tempo das aulas e **duas horas adicionais** por semana, em média. Se ao longo do semestre acharem que está levando mais do que isto, **avisem ao professor**.

Bibliografia:
A. Quarteroni e F. Saleri, *Cálculo científico com MATLAB e Octave*.

Um questionário inicial: Responder em grupos de 2 ou 3. Entregar as respostas ou mandar por e-mail.

1. Considerando os temas da ementa (ou do programa), qual lhes geram maior curiosidade ou interesse? Porquê?

2. Utilizando seus conhecimentos prévios e/ou googleando um pouco,

   (a) podem descrever a utilidade desses temas (ou de algum/ns deles) para o resto de seu curso?

   (b) podem descrever atividades de sua vida post-universidade (seja o que for que vocês pensem fazer das suas vidas) nas quais prevêm usar algum/ns desses temas?

3. Gostariam de desenvolver algum projeto nessa disciplina (dentro do tempo previsto para ela, ver "Requerimento de tempo" na página anterior)? Podem imaginar algum tema de projeto que, se for possível, lhes interessaria? Não se preocupem com a dificuldade, se exagerarem o professor adaptará ou, simplesmente, dirá que não dá.

4. Comentários adicionais (assunto libre: disciplina, ementa, programa, texto, ...):

1 Revisão de Octave

Leitura: Capítulo 1 do texto (“O que não se pode ignorar”).
Leitura adicional: Slides `intro_matlab.pdf`, do Prof. Afonso Paiva Neto.

Software: Instalar Octave ou Matlab.

Exercícios:

1. Leia uma matriz de um arquivo de texto usando o comando `load`. Como exemplo, leia o arquivo `people.txt` disponível no site. A matriz corresponde a uma imagem, que pode visualizar com o comando `imshow`.

   ```
   matriz=load("people.txt");
   imshow(matriz,[0 256])
   ```

   Veja os valores da matriz. Veja o que acontece se os limites são variados. Qual pixel da imagem corresponde ao elemento `matriz(1,1)`? Explique como descubriu isto.

2. Veja (usando o comando `size`) que a imagem de people.txt tem 625 linhas e 500 colunas. Crie outra imagem de 625×500 com os seguintes comandos

```
i=1:625;
j=1:500;
ij=i'*j;
```

Visualize a matriz `ij`. Execute e explique o resultado do seguinte código

```
ij=ij/(500*625);
imshow(ij);
imshow(matriz.*ij,[0 256])
```

3. Explique o resultado das seguintes operações

```
matriz=load("people.txt");
matriz=matriz/256;
imshow(matriz)
matriz=matriz*1.3;
imshow(matriz)
matriz=matriz*1.3;
imshow(matriz)
imshow(matriz,[0 1.69]);
```

4. Programar funções com as seguintes especificações:

   (a) `function B = suaviz1(A)`

   A matriz *B* deve ter, em cada pixel (célula), a média dos valores *das células correspondentes da matriz A e das células adjacentes em horizontal e em vertical*.

   Tente vetorizar a função anterior.

   (b) `function B = suaviz2(A)`

   A matriz *B* deve ter, em cada pixel (célula), a média dos valores *das células correspondentes da matriz A e das células adjacentes em horizontal, em vertical ou diagonalmente*.

   Tente vetorizar a função anterior.

5. Veja o efeito sobre a imagem people.txt de utilizar repetidamente as funções de suavização 1 e 2 anteriores.

6. Leia o arquivo flor.txt disponível no site. `A=load("flor.txt")` Visualize usando `imshow(A,[0 256])`. Escreva uma função

```
function B = fundopreto(A)
```

tal que o comando `imshow(B,[0 256])` mostre a mesma imagem da flor, mas com o fundo totalmente preto.

7. Explique os seguintes comandos

```
x=(0:800)/800;
y=(0:800)/800;
[xx yy]=meshgrid(x,y);
imshow(xx)
imshow(yy)
imshow((xx.^2+yy.^2)<.25)
```

8. Usando o item anterior, faça um código que desenhe a parábola $y = 4(x - 0.5)^2$ em branco sobre fondo preto, com linha de espessura 2 pixels. Tome cuidado para que o eixo y aponte para cima. Confira se a espessura é uniforme e entenda porquê.

9. Faça um código que desenhe uma elipse com o eixo maior a 30 degraus da horizontal.

10. Leia a imagem em `sky.txt` numa matriz `A`.

    (a) Descreva e explique os resultados dos seguintes comandos

```
A = load("sky.txt");
size(A)
imshow(A/256)
imshow(A>50)
imshow(A>100)
imshow(A>150)
```

(b) Calcule numéricamente o número de estrelas que aparecem quando se executa `imshow(A>limiar)`, como função do valor da variável `limiar`.

(c) Escreva um código cujo resultado seja um arquivo com a lista de coordenadas $x - y$ das estrelas da imagem com intensidade > 190, considerando que a imagem como um todo corresponde ao quadrado $(0, 1) \times (0, 1)$.

2 Solução numérica de equações não lineares

2.1 O problema

1. **O problema exato:** Seja f uma função de $\Omega \subset \mathbb{R}^m$ em $\mathbb{R}^n$. Determinar $x^* \in \Omega$ tal que $f(x^*) = 0_m$, onde 0_m é o vetor nulo de dimensão m.

   - Existe solução? É única?
   - Existe solução se $m = n$? É única?
   - É possível achar a solução em tempo finito?
   - Exemplos. Barra articulada.

2. **O problema aproximado:** Seja $f(x)$ o resultado de uma série de operações matemáticas em ponto flutuante a partir de certos dados de entrada, também em ponto flutuante, x. Determinar $\tilde{x}$ (vetor de ponto flutuante) tal que

   - **Teste do resíduo:** $\|f(\tilde{x})\|$ seja menor que uma tolerância ϵ.
   - **Erro em x:** $\|\tilde{x} - x^*\|$ seja menor que ϵ.

   Como estimar o erro em x? Absoluto ($\|x^n - x^{n+1}\|$), relativo ($\|x^n - x^{n+1}\|/\|x^n\|$).

2.2 Iterações de ponto fixo

- É frequente resolver sistemas não lineares com iterações do tipo $x^{(n+1)} = g(x^{(n)})$, parando quando $x^{(n+1)} \simeq x^{(n)}$, i.e., quando $g(x^{(n)}) \simeq x^{(n)}$. Certamente deve acontecer

$$g(x) = x \;\Leftrightarrow\; f(x) = 0\ .$$

- **Teorema (contração):** Seja $X \subseteq E$ um subconjunto fechado de um espaço vetorial completo E, e seja $g : X \to X$ uma função tal que $\|g(x) - g(y)\| \leq m\,\|x - y\|$ para todo x e y, com $m < 1$. Então g tem um único ponto fixo x^* em X e para qualquer x^0 a sequência $x^{(k+1)} = g(x^{(k)})$ que começa em $x^{(0)}$ converge para x^*.

  Prova: É simples provar que $\{x^{(k)}\}$ é uma sequência de Cauchy, porque

$$\|x^{(k)} - x^{(k+1)}\| = \|g(x^{(k-1)}) - g(x^{(k)})\| \leq m\,\|x^{(k-1)} - x^{(k)}\| \leq \ldots \leq m^k\,\|x^{(0)} - x^{(1)}\|$$

  e assim,

$$\|x^{(k)} - x^{(k+m)}\| \leq \|x^{(k)} - x^{(k+1)}\| + \|x^{(k+1)} - x^{(k+2)}\| + \ldots \leq \left(m^k + m^{k+1} + \ldots + m^{k+n-1}\right)\,\|x^{(0)} - x^{(1)}\|$$

$$\leq \frac{m^k}{1-m}\,\|x^{(0)} - x^{(1)}\| \longrightarrow_{k\to+\infty} 0\ .$$

  Então a sequência converge para certo x^*, e $g(x^*) = g(\lim x^{(k)}) = \lim g(x^{(k)}) = \lim x^{(k+1)} = x^*$. Notar que g por ser contração é automáticamente contínua. O fato de ser contração estrita ($m < 1$) garante que o ponto fixo é único.

- **Teorema:** Se $g \in C^1(X)$ e $\|Dg(x)\| \leq m$ para todo $x \in X$, então $\|g(x) - g(y)\| \leq m\,\|x - y\|$, $\forall\, x, y \in X$.

2.3 Algoritmo geral

a Avaliar f em $x^{(k)}$: $\quad f^{(k)} = f(x^{(k)})$

b Calcular a direção $d^{(k)}$ do passo resolvendo

$$B^{(k)} d^{(k)} = -f^{(k)} \ . \tag{1}$$

c Calcular o avanço α_k do passo (procura linear).

d $x^{(k+1)} = x^{(k)} + \alpha_k \, d^{(k)}$.

e Avaliar a medida de erro, terminar se erro < tolerância.

f $k \leftarrow k+1$ e voltar a **a**.

- Resumindo, $x^{(k+1)} = x^{(k)} - \alpha_k [B^{(k)}]^{-1} f(x^{(k)})$.
- Se $\alpha_k = \alpha(x^{(k)})$ e $B^{(k)} = B(x^{(k)})$, trata-se de um método de ponto fixo para a função

$$g(x) = x - C(x) f(x) \ , \text{ i.e., } g_i(x) = x_i - \sum_k C_{ik}(x) f_k(x) \ , \tag{2}$$

  com $C = \alpha B^{-1}$. Ela cumpre $f(x^*) = 0 \ \Rightarrow \ x^* = g(x^*)$. Sua derivada é

$$Dg = I - DC\, f - C\, Df \ , \text{ i.e., } Dg_{ij}(x) = \delta_{ij} - \sum_k \frac{\partial C_{ik}}{\partial x_j}(x) f_k(x) - \sum_k C_{ik}(x) \frac{\partial f_k}{\partial x_j}(x) \tag{3}$$

- Por ser $f(x^*) = 0$, $Dg(x^*) = I - C(x^*)\, Df(x^*)$.

1. O **método de Richardson** corresponde a $\alpha_k = \beta$ (constante) e $B^{(k)} = I$, i.e., $x^{(k+1)} = g(x^{(k)}) = x^{(k)} - \beta f(x^{(k)})$. Sob **hipóteses adequadas**, prove que existe β tal que as iterações de ponto fixo convergem para x^*.

2. **(a)** Provar que, se $g \in C^1(X)$ e $Dg(x^*) = 0$, então para todo $m > 0$ existe um $\delta > 0$ tal que $\|g(x) - x^*\| \leq m\|x - x^*\|$ para todo x tal que $\|x - x^*\| < \delta$.
   (b) Utilizar isto para provar que, se $f \in C^1(X)$, $C \in C^1(X)$, $Df(x^*)$ tem inversa, e $C(x^*) = Df(x^*)^{-1}$, então para todo $m > 0$ existe $\delta > 0$ tal que se $\|x - x^*\| < \delta$ então $\|g(x) - x^*\| \leq m\|x - x^*\|$.

   Dessa maneira, se $C(x^*) = Df(x^*)^{-1}$ o método de ponto fixo converge **superlinearmente** a x^*.

   O **método de Newton** é definido por $\alpha_k = 1$ e $B^{(k)} = Df(x^{(k)})$. É um método con convergência superlinear se $Df(x^*)$ é não singular.

2.4 Exemplo

Seja uma barra articulada em 2D. O primeiro segmento, de comprimento L, pode rotacionar em torno da origem (ângulo θ). O segundo, de comprimento $\ell < L$, pode rotacionar (ângulo ϕ) relativo ao primeiro. Dessa maneira, a posição do extremo é

$$\begin{aligned} X_1 &= h_1(\theta, \phi) = L\cos\theta + \ell\cos(\theta + \phi) \ , \\ X_2 &= h_2(\theta, \phi) = L\sin\theta + \ell\sin(\theta + \phi) \ . \end{aligned}$$

Os pontos alcançáveis são $x \in U = \{x \mid L - \ell \leq \|x\| \leq L + \ell\}$.
Seja a barra na sua posição de descanso O (corresponde a $\theta = 0$, $\phi = \pi/2$), e seja a um ponto tal que o segmento $\overline{Oa}$ esteja contido em U.
O objetivo é escrever um código em Octave que, utilizando o método de Newton, calcule um movimento $(\theta(t), \phi(t))$ que leve a ponta da barra de O a a por uma linha reta a velocidad constante (ou aproximadamente constante).

Vejamos que $O = (L, \ell)$, e definamos n pontos

$$X^{(i)} = O + \frac{i}{n}(a - O) \ ,$$

de tal maneira que $X^{(n)} = a$, e o tempo desejado de chegada a $X^{(i)}$ é iT/n (T é o tempo total). Nossa estratégia é ir de a a $X^{(1)}$, de ali a $X^{(2)}$, etc. Cada passo suficientemente pequeno para o método de Newton convergir, sendo que usamos sempre como chute inicial os valores $(\theta^{(i)}, \phi^{(i)})$ correspondentes ao último $X^{(i)}$ calculado.

Vejamos então como calcular $q^* = (\theta^{(i+1)}, \phi^{(i+1)})$. Notemos que q^* cumpre $f(q^*) = 0$, onde

$$f(q) = X^{(i+1)} - h(q) \ , \text{ i.e., } f(\theta,\phi) = \begin{pmatrix} X_1^{(i+1)} - L\cos\theta - \ell\cos(\theta+\phi) \\ X_2^{(i+1)} - L\sin\theta - \ell\sin(\theta+\phi) \end{pmatrix} .$$

Assim,

$$Df(q) = \begin{pmatrix} L\sin\theta + \ell\sin(\theta+\phi) & \ell\sin(\theta+\phi) \\ -L\cos\theta - \ell\cos(\theta+\phi) & -\ell\cos(\theta+\phi) \end{pmatrix} .$$

```
function hq = h(q)
global Xnew L ell
hq = [L*cos(q(1))+ell*cos(q(1)+q(2));
      L*sin(q(1))+ell*sin(q(1)+q(2))];
end

function f = fun(q)
global Xnew L ell
f=Xnew-h(q);
end

function df = der(q)
global Xnew L ell
df=[L*sin(q(1))+ell*sin(q(1)+q(2)),ell*sin(q(1)+q(2));
  -L*cos(q(1))-ell*cos(q(1)+q(2)),-ell*cos(q(1)+q(2))]
end
```

Terminar um código que calcule o desejado.

2.4.1 Octave

A **função** `fsolve` resolve $f(x) = 0$ por métodos iterativos, começando do ponto x^0 escolhido pelo usuário.

```
Function File: fsolve (FCN, X0, OPTIONS)
[X, FVEC, INFO, OUTPUT, FJAC] = fsolve (FCN, ...)
--------------------Exemplo-------------------------
function y = f (x)
  y(1,1) = -2*x(1)^2 + 3*x(1)*x(2)   + 4*sin(x(2)) - 6;
  y(2,1) =  3*x(1)^2 - 2*x(1)*x(2)^2 + 3*cos(x(1)) + 4;
endfunction
[x, fval, info] = fsolve (@f, [1; 2])
x = 0.57983 2.54621 ** fval = -5.7184e-10 5.5460e-10
info = 1
```

Ela pode utilizar a Jacobiana programada pelo usuário ou aproximações dela por métodos de diferenças finitas e/ou secantes. Isto último torna o cálculo mais lento. Ver mais detalhes fazendo `help fsolve`.

2.5 Exercícios práticos

O método de Newton é implementado em Octave como

```
k=1; x=x0; err=1e+10;
while ((k<=kmax) && (err > tol))
 f  = ..............;
 B  = ..............;
 dx = -B\f;
 err = ..............;
 x = x + dx;
 k = k + 1;
endwhile
```

onde `f` e `B` correspondem à função cujo zero se quer determinar e à sua matriz Jacobiana, respectivamente. Em `err` deve ser programada a medida de erro.

1. Que deve ser programado nas partes faltantes do código acima para que seja resolvida a equação

$$f(t) = \sin(\omega_1 t + \phi_1) - \sin(\omega_2 t + \phi_2) \ ,$$

   determinando-se o tempo t^* no qual esses dois sinais senoidais coincidem?

   Resposta: Considere o seguinte código (arquivo exemplo.m no site):

```
m1=10; phi1=pi/3; om2=13; phi2=-pi/5; x0=0;
k=1; x=x0; err=1e+10; tol=1e-10; kmax=10;
while ((k<=kmax) && (err > tol))
 f=sin(om1*x+phi1)-sin(om2*x+phi2);
```

```
 B=cos(om1*x+phi1)*om1-cos(om2*x+phi2)*om2;
 dx = -B\f;
 err = norm(dx);
 x=x+dx;  k=k+1;
endwhile
```

Ele se corresponde com o problema solicitado, tendo-se implementado o critério de parada conhecido como “erro absoluto”.

2. Como deve ser modificado o código da resposta anterior para implementar os critérios de parada de “erro relativo” e de “teste do resíduo”?

3. Escrever um código baseado no método de Newton para calcular a raiz n-éssima de um número positivo a, utilizando apenas somas, subtrações, multiplicações, divisões e potências inteiras.

   Resposta: Se a solução $z = a^{1/n}$, então $z^n = a$ e portanto z é zero da função

   $$f(x) = x^n - a \ ,$$

   cuja derivada é $f'(x) = n\,x^{n-1}$. Tanto f quanto f' podem ser calculadas sem calcular potências fracionárias. Assim, o método de Newton satisfaz o pedido. O resto fazer por conta...

4. Escrever um código baseado no método de Newton que, dados tres pontos `a`, `b` e `c` no plano (como vetores coluna), calcule o centro `z` da circunferência que passa por eles.

5. Considere duas curvas senoidais no plano:

   $$\begin{aligned} C_1 &= \{(x_1, x_2) \in \mathbb{R}^2 \,|\, x_2 = \sin(\omega_1 x_1 + \phi_1)\} \ , \\ C_2 &= \{(x_1, x_2) \in \mathbb{R}^2 \,|\, x_1 = \sin(\omega_2 x_2 + \phi_2)\} \ . \end{aligned}$$

Faça um código que determine pelo método de Newton um ponto $z \in \mathbb{R}^2$ pertencente a ambas curvas.

Resposta: É possível utilizar o código básico do Lembrete, como mostrado no arquivo exemplo2.m no site, sendo

```
f = [x(2)-sin(om1*x(1)+phi1);
     x(1)-sin(om2*x(2)+phi2)];
B = [ -cos(om1*x(1)+phi1)*om1, 1;
      1, -cos(om2*x(2)+phi2)*om2 ];
```

6. Escreva um código baseado no método de Newton que, dadas as posições tridimensionais `a`, `b` e `c` (como vetores coluna) de três emissores, e dadas as distâncias `da`, `db` e `dc` de uma posição incógnita `z` a cada um deles, determine a localização de `z`.

7. Desenhe uma imagem mostrando o número de iterações necessárias para resolver o sistema não linear

$$x^2 + y^2 = 1 \tag{4}$$
$$x^2 - y = 0 \tag{5}$$

sendo a solução procurada $(x^*, y^*) = (c, c^2)$ com $c = \sqrt{(-1+\sqrt{5})/2}$. Haverá um número de iterações para cada condição inicial (x^0, y^0), o que permite criar uma imagem de 100×100 pixeis restringindo (x^0, y^0) ao quadrado unitário.

2.6 Um mínimo sobre otimização

2.6.1 O problema

Seja X um conjunto não vazio, e seja $F : X \to \mathbb{R}$.

Problema MAX:

$$\max_{x \in X} F(x)$$

Uma solução x^* do problema, i.e., $x^* = \max_{x \in X} F(x)$, por definição satisfaz $x^* \in X$ e $F(x^*) \geq F(x)$ para todo $x \in X$. Quando é único, escrevemos

$$x^* = \max_{x \in X} F(x) \ .$$

Problema MIN: Análogo. $x^* = \min_{x \in X} F(x)$.

Teorema: Seja (X, d) um espaço métrico não vazio. Se X é compacto e $F : X \to \mathbb{R}$ é continua, então tanto MAX como MIN tem no mínimo uma solução.

Porque na verdade trabalhamos em ponto flutuante, nos interessa também o seguinte resultado.

Teorema: Seja (X, d) um espaço métrico não vazio, seja $F : X \to \mathbb{R}$ contínua, e seja $C \subseteq X$ denso em X. Então, se x^* é solução de MAX, então para qualquer $\epsilon > 0$ existe $c \in C$ tal que $F(c) > F(x^*) - \epsilon$.

Prova: Um exercício fácil de epsilons e deltas.

Def: Seja $X \subseteq \mathbb{R}^n$, $F : X \to \mathbb{R}$, e $x^* \in X$.

- x^* é um **máximo local** se e só se existe $\epsilon > 0$ tal que $F(x^*) \geq F(x)$, $\forall\, x \in B_\epsilon(x^*) \cap X$.
- x^* é um **máximo local interior** se e só se existe $\epsilon > 0$ tal que $B_\epsilon(x^*) \subseteq X$ e $F(x^*) \geq F(x)$, $\forall\, x \in X$.
- x^* é um **máximo global** se e só se $F(x^*) \geq F(x)$, $\forall\, x \in X$.

As definições para mínimo são análogas.

Exemplo:

- Seja $F(x) = -x^2$. Se $X = (a < 0, b > 0)$ então $x^* = 0$ é um máximo local interior, e global. Se $X = (a > 0, b > a)$ não há solução. Se $X = [a > 0, b > a]$ então $x^* = a$ é um máximo global, não interior.
- Seja $X = \{x \in \mathbb{R}^2 \,|\, Ax - b \leq 0\}$, onde $A \in \mathbb{R}^{m\times 2}$e $b \in \mathbb{R}^m$ é surgem de **restrições lineares** tais como

$$\left\{\begin{array}{rcl} x_1 & \geq & 0 \\ x_2 & \geq & 0 \\ x_1 + x_2 & \leq & 10 \\ x_1 + 4x_2 & \leq & 8 \\ -x_1 + x_2 & \geq & 5 \\ 5x_1 - x_2 & \geq & 10 \end{array}\right\} \quad \Leftrightarrow \quad A = \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \\ 1 & 4 \\ 1 & -1 \\ -5 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 0 \\ 0 \\ 10 \\ 8 \\ -5 \\ -10 \end{pmatrix} \tag{6}$$

Desenhe o conjunto X e conclua que ele é **compacto**. Ele de fato é um **polígono convexo**.

Otimização linear: Uma função afim qualquer $F(x) = a_0 + a_1x_1 + a_2x_2 = a_0 + c^Tx$ é **contínua** e sempre tem um **máximo global** em X. Não é difícil provar que **um dos vértices do polígono é sempre máximo global**. Porém, havendo m restrições, quais são os vértices do polígono e qual é o custo computacional de determiná-los em $\mathbb{R}^n$? É necessário analisar quantos casos? Combinatório de m tomados de a n? $C_m^n = m!/[(m-n)!n!]$? No caso, 15?

Teorema: Seja $F : X \to \mathbb{R}$. Se x^* é um máximo (ou mínimo) local interior de F e F é diferenciável em x^*, então

$$\nabla F(x^*) = 0 \tag{7}$$

Prova: Se $g^* = \nabla F(x^*) \neq 0$, então $g^* \cdot g^* > 0$, o que implica que a derivada direcional d de F em x^* na direção g^* é $\lim_{t \to 0} \frac{F(x^* + tg^*) - F(x^*)}{t} > 0$. Seja $x^n = x^* + g^*/n$. Então,

$$0 < d = \lim_n \frac{F(x^n) - F(x^*)}{(1/n)}$$

Para tudo n a partir de um valor suficientemente grande aconteceria que $F(x^n) > F(x^*)$. Então x^* não é um máximo local.

A condição $\nabla F(x^*)$ é necessária mas não suficiente para x^* ser um máximo ou mínimo local interior.

2.6.2 Otimização numérica

- Os **problemas lineares** tem métodos próprios (SIMPLEX, etc.). Normalmente são vistos em disciplinas de “otimização linear”, ou “programação matemática”, dentre outros títulos.
- Os **problemas lineares** sempre tem **restrições**.
- A rotina **glpk** de Octave

```
[xopt, fmin, errnum, extra] = glpk (c, A, b, lb, ub, ctype, vartype, sense, param)
```

resolve (não exaustivo)

$$\min c'*x$$

sujeito a

$$A*x = b, \quad x >= lb, \quad x <= ub$$

- Os **problemas não lineares** se dividem em **com** ou **sem** restrições.
- Desconsiderando as restrições, e se $F \in C^r(X)$, $r \geq 1$, a função $f(x) = \pm\nabla F(x)$ satisfaz a equação

$$f(x) = 0 \ .$$

- Se F é **quadrática**, i.e., se $\exists\, B$ matriz quadrada (simétrica), e r vetor coluna, ambos em $\mathbb{R}^n$, tais que

$$F(x) = \frac{1}{2} x^T B x + x^T r + F(0) \ , \tag{8}$$

e se ainda B é definida positiva (todos os autovalores positivos), então existe um único $x^* \in \mathbb{R}^n$ que é mínimo local e global interior. Ele pode ser calculado resolvendo

$$f(x^*) = \nabla F(x^*) = B x^* + r = 0 \ ,$$

e portanto o problema se reduz à **algebra linear**.

- Problemas quadráticos são fáceis de resolver e analisar. A **matriz Hessiana** de F, i.e.,

$$H(x) = D^2 F(x) \ , \quad \leftrightarrow \quad H_{ij}(x) = \frac{\partial^2 F}{\partial x_i \, \partial x_j}(x) \ , \tag{9}$$

é a matriz B. Se H é definida positiva (resp. negativa) haverá um mínimo (resp. máximo) em x^*.

- O custo de resolver um **problema de otimização quadrático sem restrições** é, essencialmente, o de **resolver um sistema linear**.

- Para problemas não quadráticos, o **algoritmo geral** é essencialmente o mesmo que para resolver sistemas não lineares, tomando $f = \nabla F$ (quando for um problema MIN, se for MAX alguns sinais mudam, melhor trocar F por $-F$ e minimizar).

a Avaliar f em $x^{(k)}$: $\quad f^{(k)} = \nabla F(x^{(k)})$

b Calcular a direção $d^{(k)}$ do passo resolvendo

$$B^{(k)} d^{(k)} = -f^{(k)} \ . \tag{10}$$

c Calcular o avanço α_k do passo (procura linear).

d $x^{(k+1)} = x^{(k)} + \alpha_k \, d^{(k)}$.

e Avaliar a medida de erro, terminar se erro < tolerância.

f $k \leftarrow k + 1$ e voltar a **a**.

- O **método de gradiente** (steepest descent) corresponde a tomar $B^{(k)} = I$ e calcular α realizando uma minimização de F ao longo da linha $x^{(k)} + t \, d^{(k)}$, $t \in \mathbb{R}$.

  Notar que $d^{(k)} = -\left(B^{(k)}\right)^{-1} \nabla F(x^{(k)}) = -\nabla F(x^{(k)})$.

- O **método de Newton** corresponde a tomar, para o caso MIN,

$$B^{(k)} = D^2F(x^{(k)}) \ , \qquad \alpha = 1 \ .$$

Dessa maneira, o método iterativo não é outra coisa que a resolução de uma **sequência de problemas de otimização quadrática**, onde a cada passo a função minimizada é

$$G^k(x) = F(x^{(k)}) + (x - x^{(k)})^T \nabla F(x^{(k)}) + \frac{1}{2}(x - x^{(k)})^T D^2F(x^{(k)}) (x - x^{(k)}) \ , \tag{11}$$

isto é, **o desenvolvimento de Taylor a ordem 2 da função F no ponto $x^{(k)}$**.

- O método de Newton não distingue entre máximos, mínimos, ou outros pontos críticos ($\nabla F = 0$).
- Se cumprem as mesmas propriedades de convergência que para resolver sistemas não lineares, substituindo f por ∇F e Df por D^2F. Se a matriz Hessiana é singular no mínimo ($\det D^2F(x^*) = 0$) a convergência piora ou não acontece.

2.6.3 Exercícios

1. Seja $\ell(t) = F(x^k + t\,d^k)$, onde F é de classe C^2, sendo D^2F uniformemente definida positiva (todos os autovalores de $D^2F(x)$, para todo x, são $\geq \mu > 0$. Dizer se as seguintes afirmações são verdadeiras ou falsas:

   (a) $\ell'(0) = \nabla F(x^k) \cdot d^k = (g^k)^T d^k$.

   (b) $\ell''(0) = (d^k)^T D^2F(x^k) d^k$.

   (c) Se $d^{(k)}$ é calculada pelo método do gradiente, então $\ell'(0) \leq 0$ e $\ell'(0) = 0$ se e só se $x^k = x^*$.

   (d) Se $d^{(k)}$ é calculada pelo método de Newton, então $\ell'(0) \leq 0$ e $\ell'(0) = 0$ se e só se $x^k = x^*$.

   (e) Se t^* minimiza $\ell(t)$, e $x^{k+1} = x^k + t^*\,d^k$, então $\nabla F(x^{k+1})$ é ortogonal a d^k.

   (f) No método do gradiente, se α_k se calcula minimizando exatamente a função $\ell(t)$ (isto é, se $\alpha_k = t^*$), então $d^k \cdot d^{k+1} = 0$.

   (g) Se a função F é quadrática ($= (1/2)x^T Bx + x^T r + F(0)$), então $t^* = -(d^k)^T g^k / [(d^k)^T B d^k]$. Se d^k provem do método do gradiente ou de Newton, t^* é sempre não negativo.

   (h) Se F é quadrática o método de Newton converge em uma iteração. O método do gradiente não.

2. Escrever um código baseado no método de Newton que calcule o valor x^* que minimize a distância (euclidiana usual) entre o gráfico da função $f(x) = -\cos x$ e o ponto $(x, y) = (0, 3)$. Utilize critério de parada de "erro absoluto".

3. Escrever um código baseado no método de Newton que minimize a função

$$\varphi(x, y) = 4x^2 - xy + 9y^2 + \sin(xy)\ .$$

   Utilize critério de parada de "teste do resíduo".

4. Sejam duas funções $K(x)$ e $L(x)$, definidas em $X \subseteq \mathbb{R}$ e suaves, e defina as curvas $C_K = \{(x, y) \in \mathbb{R}^2 \,|\, y = K(x)\}$ e $C_L = \{(x, y) \in \mathbb{R}^2 \,|\, y = L(x)\}$. Escrever um código baseado no método de Newton que permita calcular dois pontos $(x_K, y_K) \in C_K$ e $(x_L, y_L) \in C_L$ que minimizem a distância euclidiana entre as curvas.

   Programar e resolver em particular o caso $K(x) = e^x$, $L(x) = \ln x$.

5. Considere em $\mathbb{R}^3$ o elipsoide $\mathcal{E} = \{x_1^2 + 2x_2^2 + 3x_3^2 = 1\}$ e o paraboloide $\mathcal{P} = \{x_1^2 + x_2^2 - x_3 + 10 = 0\}$. Como determinar os pontos mais próximos $e \in \mathcal{E}$ e $p \in \mathcal{P}$? Isto é,

$$\|e - p\| \leq \|x - y\| \,, \quad \forall\, x \in \mathcal{E} \,, \quad \forall\, y \in \mathcal{P} \,.$$

2.6.4 Octave

A função `fminunc`

```
[X, FVAL, INFO, OUTPUT, GRAD, HESS] = fminunc (FCN, X0,OPTIONS)
```

resolve

$$\min_x \varphi(x)$$

sem restrições.

Exercício: Resolva com a função fminunc alguns dos exercícios anteriores. Aqui mostramos como exemplo a função de Rosenbrock. Observe o uso de optimset.

```
function [obj grad]=rosenbrock(x)
 obj=(1-x(1))^2+100*(x(2)-x(1)^2)^2;
 grad(1)=-2*(1-x(1))+200*(x(2)-x(1)^2)*(-2*x(1));
 grad(2)=200*(x(2)-x(1)^2);
end

function obj=ros(x,y)
 obj=(1-x).^2+100*(y-x.^2).^2;
end

>> x=linspace(-2,2,101); y=linspace(-1,3,81); [xx yy]=meshgrid(x,y);
>> zz=ros(xx,yy);
>> contourf(xx,yy,zz) ## primeiro plotamos
>> surf(xx,yy,zz) ## plot
```

Agora minimizamos sem restrições:

```
>> op2=optimset("fminunc")
>> [X, FVAL, INFO, OUTPUT] = fminunc (@rosenbrock,[-1.1 1.1],op2)
X =
   0.999999992158549   0.999999983908220
FVAL =    7.82064833251550e-17
INFO =  1
OUTPUT =
  scalar structure containing the fields:
    iterations =  48
    successful =  41
    funcCount =  215
>> op2=optimset(op2,"GradObj","on")
>> [X, FVAL, INFO, OUTPUT] = fminunc (@rosenbrock,[-1.1 1.1],op2)
X =
   0.999999999465288   0.999999998526949
FVAL =    1.65773586694611e-17
INFO =  1
OUTPUT =
  scalar structure containing the fields:
    iterations =  48
    successful =  41
    funcCount =  89
```

A função sqp (Sequential Quadratic Programming)

```
[X,OBJ,INFO,ITER,NF,LAMBDA]=sqp(X0,PHI,G,H,LB,UB,...
                                MAXITER,TOL)
```

resolve

$$\min_x \varphi(x)$$

sujeito a

$$G(x) = 0\,, \quad H(x) \geq 0\,, \quad \mathrm{LB} \leq x \leq \mathrm{UB}$$

Exercício: Resolva com a função sqp alguns dos exercícios anteriores.

```
>> x=0:0.01:2*pi;
>> f=@(x) x.^2 + (3+cos(x)).^2;    ### Exo. 2. func. anonima
## ou f={(@(x) x.^2 + (3+cos(x)).^2),(@(x) 2*x+2*(3+cos(x)).*(-sin(x)))};
>> plot(x,f(x))
>> [xx,obj,info,iter,nf,lambda]=sqp([2],f)
xx =  2.1179
obj =  10.635
info =  101  ### The algorithm terminated normally.
iter =  5
nf =  7
lambda = [](0x1)
```

Agora resolvamos o exercício 3 com funções não anônimas:

```
function [obj,grad]=exe3min(x)
 obj=4*x(1)^2-x(1)*x(2)+9*x(2)^2+sin(x(1)*x(2));
 grad(1)=8*x(1)-x(2)+cos(x(1)*x(2))*x(2);
 grad(2)=-x(1)+18*x(2)+cos(x(1)*x(2))*x(1);
end

>> [a b]=exe3min([1;1])
a =  12.841
b =
    7.5403   17.5403
>> [xx,obj,info,iter,nf,lambda]=sqp([1;1],@exe3min) ## notar o @
xx =
  -7.5323e-09
  -7.5726e-09
obj =    7.4303e-16
info =  101
iter =  8
nf =  13
lambda = [](0x1)
```

Exercício: Encontre um caso em que fminunc ou sqp funcionem errado, explicando porquê o funcionamento deve ser considerado errado e qual a causa do mal funcionamento. Hint: Experimente por exemplo com $\varphi(x, y) = |x - 2| + 100|y - 1|$.

Resumo:

- **Conceitos envolvidos:** Problema exato, problema aproximado, método iterativo, critérios de parada, ponto inicial. Método da bisecção. Algoritmo geral para resolução de sistemas não lineares. Método de Newton. O teorema de convergência de iterações de ponto fixo. Convergência linear e quadrática. Otimização. Mínimos e máximos. Otimização sem restrições. Algoritmo geral. Método do gradiente. Método de Newton. Busca linear. Octave: funções fsolve, glpk, fminunc, sqp. Ler páginas 39 a 56 do livro de Quarteroni & Saleri.
- **Habilidades almejadas:** Entender os diversos passos para a resolução numérica de um problema matemático algébrico ou de otimização. Ser capaz de obter soluções a problemas relativamente simples utilizando MATLAB ou Octave.

3 Interpolação

3.1 O problema algébrico da interpolação

- Considere conhecidos:

  1. **Um espaço vetorial V de funções, de dimensão finita n.**
     Exemplo: $\mathbb{P}_{n-1}$, os polinômios de grau $\leq n-1$.
     Exemplo: O espaço gerado pelas funções $\phi_1, \ldots, \phi_n$, sendo estas linearmente independentes.
  2. **Um conjunto $\{L_i\}$, $i = 1, \ldots, m$ composto por m funcionais lineares (elementos do dual V').**
     Exemplo: $L_i f = f(x_i)$, os valores da função em m pontos distintos.
     Exemplo: $L_i f = f^{(i)}(a)$, os valores das $m-1$ primeiras derivadas num ponto fixo a.
  3. **Um conjunto $\{w_i\}$, $i = 1, \ldots, m$ composto por m números (reais ou complexos).**

- **Determine um elemento $p \in V$ satisfazendo**

$$L_i p = w_i \,, \qquad \forall\, i = 1, \ldots, m \,. \tag{12}$$

Teorema: O problema algébrico acima tem solução para quaisquer $w_1, \ldots, w_m$ se e só se $m = n$ e o conjunto $\{L_i\}$ é linearmente independente. Prova-se também que nesse caso a solução é única.

- O problema ter ou não solução depende, então, do **espaço** V e do **conjunto de funcionais** $\{L_i\}$.
- Se o conjunto $\{L_i\}$ é l.i., mas $m < n$, haverá infinitos p satisfazendo (12). Em particular, haverá um elemento $z \in V$ não nulo, tal que $L_i z = 0$ para todo $i = 1, \dots, m$.
- Se o conjunto $\{L_i\}$ não é l.i., digamos, se existem $\{\alpha_k\}$ não todos zero tais que $\ell = \sum_{k=1}^{m} \alpha_k L_k = 0$, então só haverá solução se $\sum_{k=1}^{m} \alpha_k w_k = 0$.
- **Teorema**:Seja $L = \{L_i\}_{i=1}^{n}$ um conjunto de n funcionais lineares de V' (de dimensão n). As seguintes afirmações são equivalentes:
  - L é linearmente independente.
  - O único elemento $f \in V$ que cumpre $L_i f = 0$ para todo i é o elemento nulo $f = 0$.
  - Para qualquer base $\{\phi_j\}_{j=1}^{n}$ se cumpre $\det \Phi \neq 0$, onde $\Phi_{ij} = L_i(\phi_j)$.
- Em outras palavras, dado um espaço V de dimensão n (de funções, em particular $\mathbb{P}_{n-1}(\mathbb{R})$), e um conjunto L de n restrições lineares (como valor em um ponto, valor da integral, valor da derivada em outro ponto, etc.), quase sempre haverá uma e só uma função p de V que cumprirá os n valores $L_i p = w_i$. Nesse caso, dizemos que a interpolação (L, V) está **bem definida**.

- **Interpolação polinomial de Lagrange:** $L = \{L_i f = f(x_i)\}$ com os pontos $x_1, \ldots, x_n$ distintos (reais ou complexos), e $V = \mathbb{P}_{n-1}$.

  Teorema: A interpolação (L, V) está bem definida.

  Como já discutido, apenas precisamos provar que o único polinômio de grau $n-1$ que é zero em n pontos distintos é o polinômio identicamente nulo.

  Consequência: Existe um e só um polinômio de grau $\leq n-1$ que toma n valores $\{w_i\}$ dados em n pontos distintos. Existem infinitos de grau maior que $n-1$, e em geral nenhum de grau estritamente menor que $n-1$, salvo exceções (valores w_i especiais).

- Outros exemplos de interpolações bem definidas:

  - **Interpolação de Taylor:**

$$L = \{L_i = f^{(i)}(a)\}_{i=0}^{n-1}\,, \qquad V = \mathbb{P}_{n-1}\,. \tag{13}$$

    Exemplo: O polinômio de grau n cujas derivadas de ordem $j = 0, \ldots, n-1$ tomam os valores $w_0, \ldots, w_{n-1}$ é

$$p_w(x) = w_0 + w_1\, x + \frac{w_2}{2}\, x^2 + \ldots + \frac{w_{n-1}}{(n-1)!}\, x^{n-1}\,.$$

– **Interpolação trigonométrica:** V é o **espaço gerado pelas funções** 1**,** $\cos x$**,** ...**,** $\cos nx$**,** $\sin x$**,** ...**,** $\sin nx$ (espaço de polinômios trigonométricos de grau $\leq n$, $\mathbb{T}_n$, de dimensão $2n+1$), e L é **o conjunto de funcionais** $L_1 f = f(x_1)$**,** ...**,** $L_{2n+1} f = f(x_{2n+1})$**, correspondentes a avaliar em** $2n+1$ **pontos distintos em** $[-\pi, \pi)$**, é linearmente independente**.

– **Interpolação trigonométrica complexa:** V é o **espaço gerado pelas combinações lineares complexas das funções** 1, $\exp(Ix)$, $\exp(I2x)$,...,$\exp(I(n-1)x)$. e L é o valor de f em n pontos distintos em $[0, 2\pi)$.

Em outras palabras, escrevendo

$$p(x) = \sum_{j=0}^{n-1} z_j \exp(Ijx),$$

e tomando $x_i = i\,2\pi/n$, resulta para $i = 0, \ldots, n-1$,

$$\sum_j \exp(Ijx_i) z_j = \sum_j \exp[Iji(2\pi/n)] z_j = \sum_j \theta^{ij} z_j = w_i$$

onde $\theta = \exp(I2\pi/n)$, e então $\theta^{ij} = \theta^{\ \mathrm{mod}\,(ij,n)}$.

$$\theta^{ij} = \begin{pmatrix} 1 & 1 & 1 & 1 & \ldots & 1 \\ 1 & \theta & \theta^2 & \theta^3 & \ldots & \theta^{n-1} \\ 1 & \theta^2 & \theta^4 & \theta^6 & \ldots & \theta^{2n-2} \\ & & \ldots & & & \end{pmatrix}$$

– **Interpolação de Hermite:** $V = \mathbb{P}_{n-1}(\mathbb{R})$, com $n = km$, $k, m \in \mathbb{N}$, e $L_1 f = f(x_1)$, $L_2 f = f'(x_1)$, ..., $L_k f = f^{(k-1)}(x_1)$, $L_{k+1} f = f(x_2)$, ..., $L_n f = f^{(k-1)}(x_m)$. Isto é, os valores das $k-1$ primeiras derivadas em m pontos diferentes.

– **Interpolação afim em d dimensões:** V é $\mathbb{P}_1(\mathbb{R}^d)$ (polinômios de grau ≤ 1 em d variáveis). Os funcionais $L_i f = f(x_i)$ correspondem a avaliar a f em $d+1$ pontos distintos **não alinhados** (vértices de um simplex de volume não zero).

– **Interpolação linear por partes (contínua):**Sejam $x_1 < x_2 < ... < x_n$ pontos em $\mathbb{R}$. Seja

$$V = \{ f \in C([x_1, x_n]) \mid f \text{ restrita a } [x_i, x_{i+1}] \text{ é um polinômio de grau } \leq 1 \} . \tag{14}$$

Seja $L = \{L_i = f(x_i)\}$. Ver que (L, V) é uma interpolação bem definida.

3.2 Cálculo da interpolada

- Para o cálculo da interpolada p, que assumimos bem definida, utilizamos uma base de V. Seja ϕ_1, ..., ϕ_n a base (um conjunto qualquer de n elementos l.i.). Assim, determinar p corresponde a determinar a_1, ..., a_n (em $\mathbb{R}$ ou $\mathbb{C}$), tais que

$$p(x) = \sum_{j=1}^{n} a_j \phi_j(x) \ , \qquad \forall\, x \ . \tag{15}$$

- Substituindo em $L_i p = w_i$,

$$\sum_j L_i(\phi_j)\, a_j = w_i \ , \qquad i = 1, \ldots, n \ .$$

  Em términos matriciais, $M\, a = w$, onde $M_{ij} = L_i(\phi_j)$.

- Assim, para calcular $p(x)$, sendo x um ponto qualquer (do domínio das funções de V), os passos são:

  1. Construa M e w a partir dos dados.
  2. Resolva $Ma = w$ para obter o vetor a. $\det M \neq 0$ sempre!
  3. Calcule $p(x) = \sum_{j=1}^{n} a_j\, \phi_j(x)$.

- **A matriz M depende da escolha da base, e portanto o vetor de coeficientes a também, mas p não**. Isto é, $p(x)$ é independente da base escolhida, para todo x.

- A **base** pode ser escolhida pelo usuário. Um motivo acostuma ser a **facilidade** de resolver $Ma = w$.

- Existe sempre uma base privilegiada (canônica) associada ao conjunto L, definida por $L_i(\varphi_j) = \delta_{ij}$. Nessa base M é a matriz identidade.

- Para a interpolada de Lagrange, i.e., $V = \mathbb{P}_{n-1}$, $L_i f = f(x_i)$, a base canônica é

$$\varphi_j(x) = \frac{\Pi_{i \neq j}(x - x_i)}{\Pi_{k \neq j}(x_j - x_k)} \ , \tag{16}$$

  os chamados **polinômios de Lagrange** associados a $\{x_1, \ldots, x_n\}$.

- Exercício: Calcular a base canônica associada à interpolação de Hermite, considerando $n = 4$ ($k = m = 2$).

- A base escolhida pelo método de Newton de cálculo da interpolada é

$$\varphi_1(x) = 1 \ , \quad \varphi_2(x) = x - x_1 \ , \quad \varphi_3(x) = (x - x_1)(x - x_2) \ , \quad \text{etc.} \tag{17}$$

  Nessa base, a matriz M da interpolação de Lagrange é triangular inferior. Por isto, é possível calcular $a_1 = f(x_1)$ sem conhecer $a_2, \ldots$, e sem sequer saber quanto vale n nem $x_2, x_3, \ldots$. Isto permite ir adicionando pontos um após o outro sem refazer os cálculos dos $a_i's$ prévios.

3.3 Usos da interpolação: Exercícios

A interpolação é de uso frequente na modelagem, na análise numérica, na estatística, dentre muitos outros campos de atuação. Os seguintes são exercícios que exemplificam esses usos.

1. Sejam $\{x_1, \ldots, x_m\} \subset \mathbb{R}$ pontos conhecidos, $x_1 < x_2 < \ldots < x_m$, e $\{y_1, \ldots, y_m\} \subset \mathbb{R}$ valores conhecidos de uma função.

   (a) A partir dos valores y_m, y_{m-1} e y_{m-2}, calcule uma estimativa da derivada $y'(x)$ avaliada em x_m utilizando interpolação quadrática. Uma vez calculada, particularize-a para o caso em que os x_i estão equi-espaçados a distância h. Considerando a função $y(x) = \sin(x)$ e $m = 3$, com $x_m = 0$, compare a estimativa com o valor exato $y'(x_m) = 1$ para vários valores de h (0.1,0.01,0.001). A diferença entre o valor exato e o estimado tende a zero com que potência de h?

   (b) A partir dos valores y_m, y_{m-1} e y_{m-2}, calcule uma estimativa da derivada segunda $y''(x)$ avaliada em x_{m-1} utilizando interpolação quadrática. Particularize para o caso em que os x_i estão equi-espaçados a distância h.

   (c) A partir dos valores y_m, y_{m-1}, y_{m-2} e y_{m-3}, calcule uma estimativa da derivada segunda $y''(x)$ avaliada em x_m utilizando interpolação cúbica. Particularize para o caso em que os x_i estão equi-espaçados a distância h.

2. A integral da interpolada aproxima a integral da função.

   Escrever um código octave que estime as integrais

$$I_1 = \int_{x_1}^{x_m} y(x)\, dx$$

$$I_2 = \int_{x_1}^{x_m} (y'(x))^2\, dx$$

$$I_3 = \int_{x_1}^{x_m} (y''(x))^2 \, dx$$

utilizando a interpolada de Hermite cúbica por partes que coincide com y em valor e derivada ($y(x_i) = y_i$, $y'(x_i) = d_i$, y_i e d_i são dados) nos pontos $x_1, \ldots, x_m$.

3. Seja $m = 3$. Estime a integral

$$I = \int_{x_1}^{x_m} y(x) \, dx$$

integrando a interpolante quadrática de $y(x)$. Especificamente, calcule A_1, A_2 e A_3 tais que

$$I = \int_{x_1}^{x_3} y(x) \, dx = A_1 y(x_1) + A_2 y(x_2) + A_3 y(x_3)$$

sempre que $y \in \mathbb{P}_2(\mathbb{R})$. Uma vez calculado o caso geral, particularizar ao caso de pontos equiespaçados a distância h. Escrever um código octave que, dados x_1, x_2 e x_3, calcule A_1, A_2 e A_3.

4. Repetir o exercício anterior com $m = 4$ e interpolantes cúbicas.

5. Considerar uma viga ocupando o intervalo $0 \leq x \leq L$. A deflecção vertical da viga é uma função $y(x)$ que será interpolada por uma função quadrática a partir dos valores $y(0) = y'(0) = 0$, $y_1 = y(x = L)$.

   Escrever um código em Octave que, a partir de y_1 estime

   (a) Energia de flexão:
   $E_B(y_1, y_2) \simeq K \int_0^L (y''(x))^2 \, dx$

   (b) Energia potencial gravitatória:
   $E_G(y_1, y_2) \simeq \int_0^L y(x) \, dx$

6. A partir do resultado do exercício anterior, escrever um código que calcule y_1^* que minimize a energia total $E = E_B + E_G$. Isto permite calcular a deformação da viga quando submetida ao peso próprio.

7. A equação de um circuito RLC passivo é dada, em termos da corrente $I(t)$, por

$$\frac{d^2}{dt^2} I(t) + 2\,\alpha\,\frac{d}{dt} I(t) + \omega_0^2\, I(t) = 0\ ,$$

onde α é chamada de frequência de Neper e ω_0 é a frequência natural.

Foram medidos valores I_1, I_2, ..., I_m, a tempos $t_1 < t_2 < \ldots < t_m$. Se pede dar uma estimativa de α e de ω_0 a partir desses valores, utilizando interpolantes quadráticas entre ternas de pontos consecutivos.

8. A equação de um pêndulo é

$$\frac{d^2\theta}{dt^2} + \frac{g}{\ell}\,\sin\theta = 0\ .$$

Considere tempos $t_{m-2} = t_m - 2\,h$, $t_{m-1} = t_m - h$ e t_m, com valores angulares correspondentes θ_{m-2}, θ_{m-1} e θ_m.

Qual deve ser o valor de θ_m, como função de θ_{m-1} e θ_{m-2}, de tal maneira que a equação diferencial seja cumprida **ao tempo** $\tau = t_{m-1}$ **pela interpolada quadrática de** $\theta(t)$.

Programar a regra obtida $(\theta_{m-2}, \theta_{m-1}) \mapsto \theta_m$ para calcular a evolução de um pêndulo com condições iniciais $\theta_1 = \theta_2 = 0.95\,\pi$.

9. Repetir o exercício anterior para $\tau = t_m$.

3.4 Teoremas de aproximação

Nessa seção consideramos, ademais da interpolada (L, V) um espaço maior W (que contém V), no qual as formas lineares L_i estão bem definidas (i.e., são contínuas). Assim, a cada $f \in W$ se associa $\mathcal{I}f \in V$ tal que

$$L_i(\mathcal{I}f) = L_i f, \qquad \forall i = 1, \dots, n.$$

A função $\mathcal{I}f$ é a **interpolada** de f. O operador $\mathcal{I} : W \to V$ cumpre $\mathcal{I}f = f$ para todo $f \in V$. E surge a pergunta: "Quanto vale o **erro de interpolação** $|f(x) - \mathcal{I}f(x)|$? Ele pode ser reduzido a valores arbitrariamente pequenos? Como?"

Teorema (Weierstrass): Seja $f \in C([a, b])$. Então, para todo $\epsilon > 0$ existe um polinômio p tal que

$$\|f - p\|_\infty \leq \epsilon$$

Corolário: Seja $f \in C^1([a, b])$. Para todo $\epsilon > 0$ existe p tal que $\|f - p\|_{C^1([a,b])} < \epsilon$.

Esse resultado não vale apenas para polinômios, como mostra o seguinte teorema.

Teorema (Stone-Weierstrass): Seja $D \subset \mathbb{R}^d$ um compacto. Seja $\mathcal{S}$ um subespaço de $C(D)$ com as seguintes propriedades:

a) $\mathcal{S}$ contem as funções constantes.

b) $u, v \in \mathcal{S} \quad \Rightarrow \quad uv \in \mathcal{S}$.

c) Para cada par de pontos $x, y \in D$, $x \neq y$, existe $v \in \mathcal{S}$ tal que $v(x) \neq v(y)$.

Então, $\mathcal{S}$ é denso em $C(D)$, isto é, para todo $v \in C(D)$ existe $\{v_n\} \subset \mathcal{S}$ tal que

$$\|v - v_n\|_\infty \to 0 \quad \text{quando} \quad n \to \infty \ .$$

Corolários:

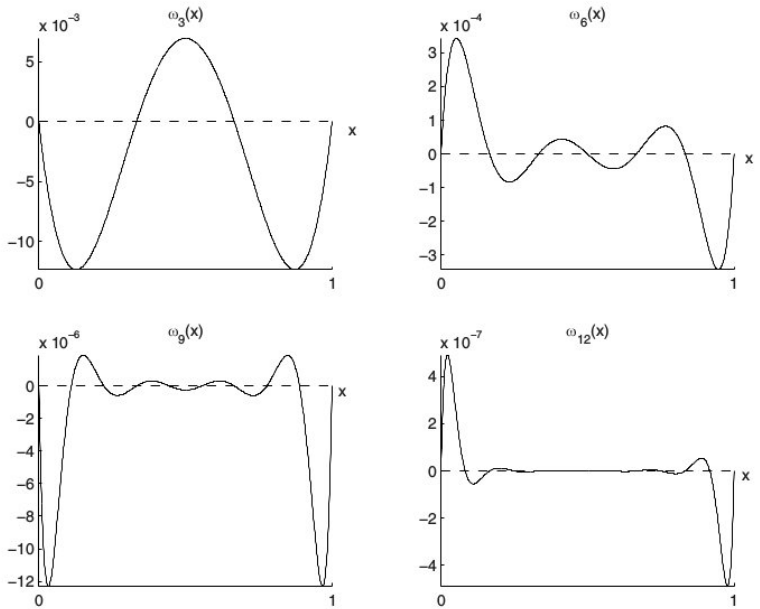
- Seja $D \subset \mathbb{R}^d$ um compacto. Os polinômios são densos em $C(D)$ e portanto também em $L^2(D)$.
- O conjunto de polinômios trigonométricos é denso em $C_p(-\pi, \pi)$ (funções 2π-periódicas contínuas em $\mathbb{R}$).

- Notar que esses resultados teóricos indicam que a aproximação é possível com erro $< \epsilon$, para qualquer $\epsilon > 0$, mas não indicam como calcular o polinômio aproximante.
- Uma possibilidade: Será que a **interpolada de Lagrange** da função f, quando o grau do polinômio cresce (e portanto também o número de pontos de interpolação), tende à função f (i.e., o erro tende a zero)?

Teorema: Seja $f \in C^n([a,b])$, e suponha que $f^{(n+1)}(x)$ existe $\forall\, x \in (a,b)$. Se $a \le x_1 < \ldots < x_{n+1} \le b$, seja $p_n \in \mathbb{P}_n$ a interpolada de Lagrange de f. Então

$$f(x) - p_n(x) = \underbrace{\frac{\omega_n(x)}{(n+1)!}\, f^{(n+1)}(\xi)}_{R_n(x)}\,, \qquad \text{onde} \quad \omega_n(x) = (x-x_1)(x-x_2)\ldots(x-x_{n+1})\,, \tag{18}$$

e ξ é $> \min(x_1, x)$ e $< \max(x_{n+1}, x)$.



Polinômios $\omega_n(x)$ em $[0,1]$ para pontos equiespaçados.

Convergência, fenômeno de Runge:

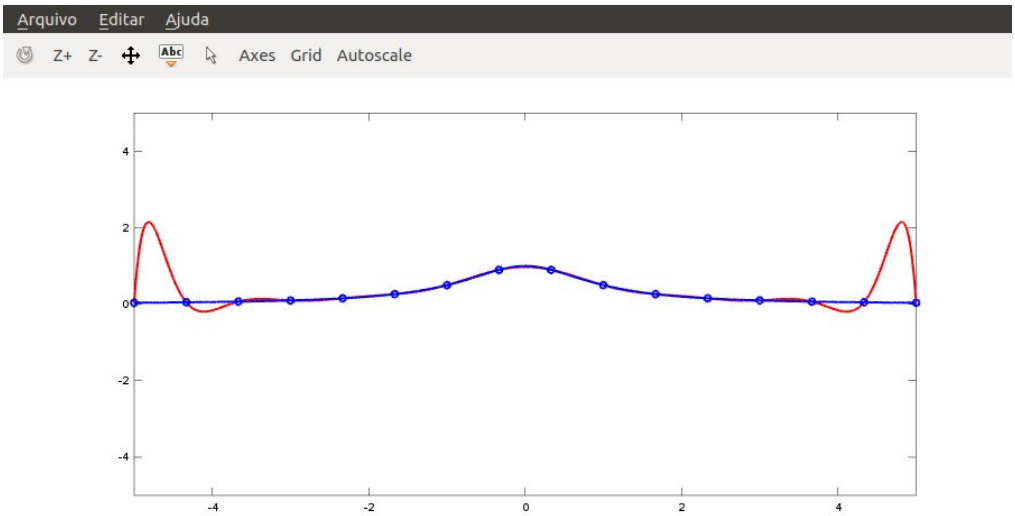
- Seja $f \in C^0([a,b])$. Dado um $\epsilon > 0$ sabemos, pelo teorema de Weierstrass, que existe um polinômio p_n tal que $|f(x) - p_n(x)| < \epsilon$ para todo $x \in [a,b]$.
- Também sabemos que, dado um número n qualquer, existe um único polinômio q_n de grau n que coincide com f em $n+1$ pontos (de um conjunto enumerável pre-estabelecido).
- Porém, **se os pontos estão equi-espaçados**, não é possível garantir que $q_n(x) \to f(x)$ quando $n \to \infty$.
- No arquivo `runge.m` está programada a seguinte função:

```
function mm=runge(n)
i=1:n+1; a=-5; b=5;
f=@(x) 1./(1+x.^2);
## equispaced
x=a+(i-1)*(b-a)/n;
p=polyfit(x,f(x),n);
m=5000; j=1:m+1;
xp=a+(j-1)*(b-a)/m;
yp=polyval(p,xp);
plot(xp,yp,"-r",x,f(x),"ob",xp,f(xp),"-b")
axis([-5 5 -5 5])
mm=norm(f(xp)-yp,inf);
end
```

- A função plota $f(x) = 1/(1+x^2)$ no intervalo $[-5, 5]$, e seu polinômio interpolador q_n a partir de $n+1$ nós equi-espaçados. A aparência de $f(x)$ é inocente. Os valores obtidos são os seguintes

n	3	6	9	12	15	18	21	24
$\|f - q_n\|_\infty$	0.707	0.617	0.300	3.663	2.107	29.190	17.602	257.21

  Os polinômios q_n não convergem a f.



- Existem duas técnicas fundamentais para evitar o fenômeno de Runge:
  - Usar pontos **não equiespaçados** (e.g., pontos de Chebyshev),
  - Usar interpolantes **polinomiais por partes** (e.g., splines).

Pontos de Chebyshev, splines

- Notar que, se $f \in C^{n+1}([a,b])$, então

$$\|f - p_n\|_\infty \leq \frac{1}{(n+1)!} \|f^{(n+1)}\|_\infty \|\omega_n\|_\infty \ . \tag{19}$$

- Faz sentido ver qual é a distribuição de pontos $x_1, \ldots, x_{n+1}$ que minimiza $\|\omega_n\|_\infty \Longrightarrow$ **Pontos de Chebyshev**.

$$x_i = \frac{a+b}{2} - \frac{b-a}{2} \cos\left(\frac{2i-1}{2n+2}\pi\right) \ , \qquad i = 1, \ldots, n+1. \tag{20}$$

- Com apenas **Lipschitz-continuidade**, as interpolantes de Chebyshev convergem a f.

  Definição: Uma função se diz **Lipschitz-contínua**, de módulo L, em um domínio D (normado) se $|f(x) - f(y)| \leq L\|x - y\|$ para todo $x, y \in D$.

- Se $f \in C^s([a,b])$, então $\|f - p_n\|_\infty \leq C\, n^{-s}$ quando $n \to \infty$.

- Faber em 1942 provou que nenhum esquema de interpolação pode convergir para toda função apenas contínua.

- A função **spline** de Octave implementa a spline cúbica natural.
- Para a função de Runge se utiliza

```
function mm=runge(n)
i=1:n+1; a=-5; b=5;
f=@(x) 1./(1+x.^2);
## equispaced
x=a+(i-1)*(b-a)/n;
m=5000; j=1:m+1;
xp=a+(j-1)*(b-a)/m;
ys=spline(x,f(x),xp);
plot(xp,ys,"-r","linewidth", 2,x,f(x),"ob","linewidth", 2,xp,f(xp),"-b","linewidth", 2)
axis([-5 5 -5 5])
mm=norm(f(xp)-ys,inf);
end
```

- O resultado das diversas variantes é ($n+1$ = número de pontos)
  - Usando interpolação polinomial global (n = grau do polinômio):

n	3	6	9	12	15	18	21	24	
$\|f-q_n\|_\infty$	0.707	0.617	0.300	3.663	2.107	29.190	17.602	257.21	equi-espaçados
$\|f-q_n\|_\infty$	0.750	0.264	0.269	0.069	0.083	0.022	0.025	0.007	Chebyshev

  - Usando splines (polinômios cúbicos por partes):

n	3	6	9	12	15	18	21	24	
$\|f-q_n\|_\infty$	0.707	0.132	0.142	6.9E-3	3.1E-2	3.7E-3	8.1E-3	1.9E-3	equi-espaçados
$\|f-q_n\|_\infty$	0.750	0.216	0.285	4.7E-2	0.102	6.2E-3	3.9E-2	3.5E-3	chebyshev

- Exercício: Modificar o código `runge.m` e reproduzir as tabelas acima.
- O código `census.m` do site mostra outras funções de interpolação.

- Exercício: Considere o intervalo $[0, 1]$, e nele insira N pontos distintos arbitrários $X_1, \ldots, X_N$. Chamemos $\mathcal{X}$ o conjunto desses pontos. A partir deles, defina a interpolada $\mathcal{I}f$ no ponto x de uma função contínua f como o valor em x da função quadrática que coincide com f **nos 3 pontos de $\mathcal{X}$ mais próximos de** x.
  - Escreva um código Octave que plote a função e a interpolada quando $f(x) = \cos^7(3x)$ e $\mathcal{X} = \{0, 0.1, 0.3, 0.35, 0, 5, 0.75, 0.8, 0.95\}$.
  - Qual é o espaço V **imagem** de $\mathcal{I}$?
  - Pode plotar uma **base** de V?

- Exercício: Escreva uma função de Octave que, dados n pontos no plano, $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, plote a "spline cúbica natural que passa por eles". Os pontos estão em qualquer posição, não distribuidos como o gráfico de uma função.
  - Veja a solução proposta no capítulo de interpolação do livro de Moler. Ela se baseia na criação de duas splines naturais para $x(t)$ e $y(t)$. Para isto, é necessário acrescentar um dado: os valores $t_1, t_2, \ldots, t_n$ do parâmetro da curva em cada um dos pontos. Qual é o valor implementado por Moler para t_i?
  - Analize o efeito da parametrização. Se são escolhidos outros valores de t_i's, muda a curva construida? Poderia explicar essa mudança?
  - Discuta sobre a continuidade da **curva plotada** (não a continuidade das **funções** $x(t)$ e $y(t)$, que yá sabemos são C^2). Ela é C^0? E C^1? E C^2? Pode mostrar isto numericamente? Pode provar isto teoricamente?

4 Melhor aproximação e mínimos quadrados

4.1 Exemplos

Tomados principalmente de Davis.

- **Aproximação de uma função:** Se é definida uma distância entre funções, é possível procurar num conjunto S de funções aquela que está mais próxima de um dado f.

  O resultado depende da função distância. Para aproximar $y = x^4$ em $[0, 1]$ por uma reta $y = p(x)$, por exemplo

  - Se se minimiza $\int_0^1 (x^4 - p(x))^2 \, dx$, se obtém $p(x) = \frac{4}{5}x - \frac{1}{5}$.
  - Se se minimiza $\int_0^1 (x^4 - p(x))^2 \, dx + \int_0^1 (d/dx(x^4 - p(x)))^2 \, dx$ se obtém $p(x) = \frac{54}{55}x - \frac{1}{5}$.
  - Se se minimiza $\max_{0 \le x \le 1} |x^4 - p(x)|$ se obtém $p(x) = x - 0.236\ldots$.

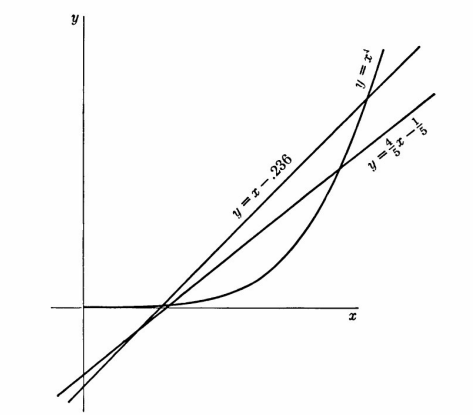


Figure 7.1.1 Least Square and Best Uniform Linear Approximations to x^4 on $[0, 1]$.

- **Aproximação de um conjunto de dados:** Dadas n medições (x_k, y_k), procurar uma função p em S que minimiza a distância entre as medições $\{(x_k, y_k)\}$ e $(x, p(x))$. A distância pode ser definida como
  - $d = \left(\sum_k |y_k - p(x_k)|^p\right)^{1/p}$,
  - $d = \max_k |y_k - p(x_k)|$.
- **Melhor aproximação por combinações lineares:** Sejam $z_1, \ldots, z_n$ um conjunto de elementos linearmente independentes de um espaço vetorial S. Seja y um elemento adicional. Aproximar y por $a_1 z_1 + \ldots + a_n z_n$ minimizando a distância de y a S.
- **Melhor solução de sistemas sobredeterminados:** Sejam os números A_{ij}, y_i, conhecidos para $1 \leq i \leq m$, $1 \leq j \leq n$, $m > n$. A "melhor solução" x^* pode ser definida como aquela que minimiza

$$\min_{x=(x_j)} \max_{1 \leq i \leq m} |y_i - (A_{i1}x_1 + \ldots + A_{in}x_n)| \ .$$

- Ler capítulo 3 de Quarteroni-Saleri (pags. 71 a 100).

4.2 Distâncias, normas

- Uma **distância** em um conjunto X é uma função de $X \times X$ em $\mathbb{R}$ satisfazendo **não negatividade** ($d(x,y) \geq 0$), **simetria** ($d(x,y) = d(y,x)$), **desigualdade triangular** ($d(x,y) \leq d(x,z) + d(z,y)$) e **positividade** ($d(x,y) = 0 \Leftrightarrow x = y$).

- Uma **norma** em um espaço vetorial X é uma função de X em $\mathbb{R}$ satisfazendo **não negatividade** ($\|x\| \geq 0$), **desigualdade triangular** ($\|x+y\| \leq \|x\| + \|y\|$), **homogeneidade** ($\|\alpha x\| = |\alpha| \, \|x\|$) e **positividade** ($\|x\| = 0 \Leftrightarrow x = 0$).

- **Teorema:** Se $\|\cdot\|$ é uma norma em X, então $d(x,y) = \|x-y\|$ é uma distância em X.

- Exemplos de normas:

  - Em $\mathbb{R}^n$,
  $$\|x\| = (|x_1|^p + \ldots + |x_n|^p)^{\frac{1}{p}} \quad , \quad \text{ou} \quad \|x\| = \max_j \{|x_1|, \ldots, |x_n|\} \; .$$

  - Seja $B([a,b])$ o espaço das funções limitadas em $[a,b]$ (ou qualquer domínio $D \subset \mathbb{R}^d$). Equipado com a norma $\|f\| = \sup_{x \in D} |f(x)|$ é um espaço vetorial normado.

  - $C([a,b])$ (ou qualquer compacto), com $\|f\| = \max_{x \in [a,b]} |f(x)|$. Essa é a **norma da convergência uniforme**.

  - $C([a,b])$, com $\|f\| = \left(\int_a^b w(x) \, f(x)^2 \, dx \right)^{1/2}$, onde $w(x) > 0$, $\forall \, x$ (**norma da média quadrática com pesso**).

  - Dados m pontos, dos quais no mínimo $n+1$ diferentes, em $\mathbb{R}$, a função
  $$\|f\| = \left(\sum_{i=1}^{m} |f(x_i)|^p \right)^{\frac{1}{p}}$$

é uma norma em $\mathbb{P}_k$, $k \leq n$. Sejam $(x_1, y_1), \ldots, (x_m, y_m)$, m medições em $\mathbb{R}$. Seja $V = \mathbb{P}_k$, $k \leq n$. Então $\|f\|^2 = \sum_{i=1}^{m} (f(x_i))^2$ é uma norma em P_k. Esse é o problema do **ajuste polinomial de dados por mínimos quadrados**.

- **Definição:** Um espaço vetorial normado X é chamado de **estritamente convexo** se $\|x\| \leq r$ e $\|y\| \leq r$ implicam que $\|x + y\| < 2r$ sempre que $x \neq y$.

- Em $\mathbb{R}^2$, sendo

$$\|x\|_p = \begin{cases} \left(|x_1|^p + |x_2|^p\right)^{\frac{1}{p}} & \text{se } p \in \mathbb{R}\,, \\ \max\{|x_1|, |x_2|\} & \text{se } p = +\infty\,, \end{cases}$$

  com $1 < p < +\infty$, a norma é estritamente convexa.

4.3 Existência, unicidade

Teorema: Seja K um subespaço de dimensão finita de um **espaço normado** X. Então para todo $x \in X$ **existe** um elemento $x^* \in K$ tal que

$$\|x - x^*\| = \inf_{y \in K} \|x - y\| .$$

Equivalentemente, dados n elementos linearmente independentes $z_1, \ldots, z_n$ de X, **existem** $a_1^*, \ldots, a_n^*$ em $\mathbb{R}$ (ou $\mathbb{C}$, se for um espaço complexo) tais que

$$\|x - a_1^* z_1 - \ldots - a_n^* z_n\| = \inf_{a_1, \ldots, a_n} \|x - a_1 z_1 - \ldots - a_n z_n\| .$$

Se ainda o espaço é **estritamente convexo**, então o problema de melhor aproximação acima tem **solução única**.

- Exercício: Seja $X = \mathbb{P}_1$, com a norma $\|f\| = |f(0)| + |f(1)|$. Qual a função constante que melhor aproxima a função $x \in \mathbb{P}_1$? *Dica: $\|\cdot\|$ é uma norma? A solução é única? Identifique o conjunto solução.*
- Exercício: Repetir o exercício anterior com a norma $\|f\|^2 = f(0)^2 + f(1)^2$. Qual a função constante que melhor aproxima a função $x \in \mathbb{P}_1$? A função $\|\cdot\|$ é uma norma estritamente convexa? A solução é única?
- Exercício: Seja $Y = C([0,1])$ e $\|f\|^2 = \int_0^1 f(x)^2\, dx$. Qual a função constante que melhor aproxima a função $e^x \in Y$? Considere o espaço $X = \mathbb{P}_0 + \{\alpha e^x, \alpha \in \mathbb{R}\}$ (combinações lineares de constantes e múltiplos de e^x).

4.4 Mínimos quadrados

- O cálculo da melhor aproximação se simplifica muito se a **distância provém de um produto escalar**.
- **Definição:** Um **produto escalar** em um espaço vetorial X é uma função de $X \times X$ em $\mathbb{R}$ (pode ser adaptado a $\mathbb{C}$) satisfazendo **bilinearidade**, **simetria**, e **positividade** ($(x,x) \geq 0$, e $(x,x) = 0 \Leftrightarrow x = 0$).
- Exemplos:
  - $\mathbb{R}^n$, $(x,y) = \sum_i x_i y_i$. (produto ℓ_2)
  - $C([a,b])$, $(f,g) = \int_a^b f(x)g(x)\,dx$. (produto L^2)
  - $X = \mathbb{R}^n$. Então, uma função bilinear $(\cdot,\cdot)$ é um produto escalar se e só se existe uma **matriz simétrica definida positiva** G tal que

    $$(x,y) = \sum_{ij} G_{ij} x_i y_j \qquad \forall x, y \in \mathbb{R}^n.$$

    Se G existe, ela é única.
  - O conjunto de produtos escalares diferentes num X de dimensão n é isomorfo (embora não sejam espaços vetoriais, ☺) ao conjunto de matrizes definidas positivas.
- **Teorema:** Se $(\cdot,\cdot)$ é um produto escalar em X, então $\|x\| = (x,x)^{1/2}$ é uma **norma estritamente convexa** em X, e $d(x,y) = \|x - y\|$ é uma **distância** em X. Assim, dado um subespaço S de dimensão finita de X, o problema

  $$\inf_{z \in S} d(z,x)\,, \qquad \text{ou} \quad \inf_{z \in S} \|z - x\|\,,$$

  tem uma solução única $p^* \in S$.

- As distâncias provenientes de produto escalar facilitam muito o cálculo da melhor aproximação.

- **Teorema (exercício: provar):** Seja X um espaço com produto escalar, f um elemento de X, e $\varphi_1, \ldots, \varphi_n$ elementos l.i. de X. Então:

  1. O problema
  $$\min_{(a_1,\ldots,a_n)} \|f - a_1\varphi_1 - \ldots - a_n\varphi_n\|$$
  tem uma única solução $(a_1^*, \ldots, a_n^*)$.
  2. Esses coeficientes ótimos são solução do sistema linear (**equações normais**)
  $$Ma^* = b\ , \quad \text{sendo}\ \ M_{ij} = (\varphi_i, \varphi_j)\ , \quad \text{e}\ \ b_i = (\varphi_i, f)\ , \qquad i, j = 1, \ldots, n\ . \tag{21}$$
  3. O sistema acima sempre tem solução porque M é simétrica definida positiva (**matriz de Gram** de $\varphi_1, \ldots, \varphi_n$).
  4. A solução obtida $g = a_1^*\varphi_1 + \ldots + a_n^*\varphi_n$ é a **melhor aproximação** de f em $K = \text{span}\{\varphi_1, \ldots, \varphi_n\}$ (o espaço de combinações lineares das $\{\varphi_i\}$).
  5. O **erro de aproximação** $f - g$ é ortogonal a K, i.e.,
  $$(f - g, z) = 0\ , \qquad \forall\, z \in K\ , \tag{22}$$
  o que qualifica g como a **projeção ortogonal** de f sobre K.
  6. Definindo as funcionais lineares $L_i f = (f, \varphi_i)$, o espaço K equipado com $\{L_i\}$ pode ser visto como um sistema de interpolação. A "interpolada associada" seria g, definida por
  $$L_i g = w_i\ , \quad \forall\, i = 1, \ldots, n\ , \qquad \text{onde} \quad w_i = (f, \varphi_i) = L_i f\ .$$

- **Corolário (sistemas lineares sobredeterminados):** Dado o sistema linear $Ax = b$, com $x \in \mathbb{R}^n$ e $b \in \mathbb{R}^m$, $m \geq n$, a solução x^* de mínimos quadrados, i.e., que minimiza $\|Ax - b\|$ proveniente do produto escalar em $\mathbb{R}^m$ dado por $(x, y) = \sum_{i=1}^{m} x_i y_i = x^T y$, satisfaz $A^T A x^* = A^T b$. Por outro lado, se o produto escalar é $(x, y) = \sum_{ij} G_{ij} x_i y_j = x^T G y$, então o sistema de equações normais é

$$A^T G A x^* = A^T G b \, .$$

  Em ambos casos, Ax^* é a projeção ortogonal de b sobre $\mathrm{Im}(A)$. Isto é,

$$(Ax^* - b, Az) = 0 \, , \qquad \forall \, z \in \mathbb{R}^n \, .$$

- Dado o sistema linear $Ax = b$, a solução de mínimos quadrados correspondente ao produto escalar $(x, y) = \sum_i x_i y_i$ é obtida por Octave fazendo `x=A\b`.

- Dado um conjunto de dados f, tais como uma função ou um conjunto de valores, a melhor aproximação deles, g, desde um espaço de dimensão finita K, é calculada como segue quando a distância desejada provém do produto escalar $(\cdot,\cdot)$:
  - Escolha uma base de K, seja ela $\{\varphi_1, \ldots, \varphi_n\}$.
  - Calcule todos os produtos escalares $M_{ij} = (\varphi_i, \varphi_j)$ e $b_i = (\varphi_i, f)$.
  - Resolva o sistema linear $M\,a^* = b$.
  - $g = a_1^*\varphi_1 + \ldots + a_n^*\varphi_n$.
- Exercício: Sejam $a = x_1 < x_2 \ldots < x_n = b$ pontos ordenados em $\mathbb{R}$, e seja K o espaço P_1-contínuo (polinômios $\mathbb{P}_1$ por partes, já discutido anteriormente). Fazer um código de Octave que calcule a melhor aproximação desde K à função

$$f(x) = \begin{cases} \frac{1}{d-c} & \text{se } x \in [c,d]\,, \\ 0 & \text{se não,} \end{cases}$$

onde $a \le c < d \le b$ e o produto escalar é

$$(h,g) = \int_a^b h(x)\,g(x)\,dx\,, \qquad \forall\, f, g \in L^2([a,b])\,.$$

Fazendo $d - c$ pequeno se obtém a representação em K do delta de Dirac.

- **O ajuste de dados por mínimos quadrados**
  - Sejam $(x_1, y_1), \ldots, (x_m, y_m)$ medições, das quais algumas acontecem no mesmo x mas no mínimo $n+1$ dos x's são diferentes.
  - Seja K o espaço onde se deseja achar uma aproximação, de dimensão $n+1$. Por exemplo, $\mathbb{P}_n$.
  - Qualquer que seja K, seja $\phi_1, \ldots, \phi_{n+1}$ uma base.
  - A distância do vetor de dados y a $p \in K$ é dada por

$$\|y - p\| = \sqrt{\sum_{i=1}^{m} W_i [y_i - p(x_i)]^2} \ ,$$

    onde $W_i > 0$ podem ser pesos assignados a cada medição, e os produtos escalares por

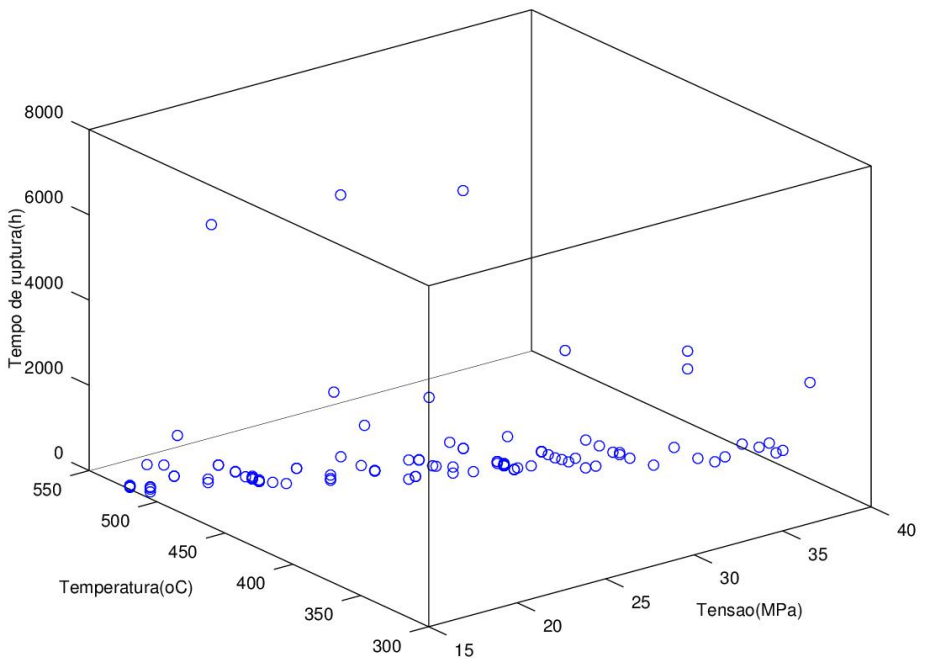
$$M_{k\ell} = (\phi_k, \phi_\ell) = \sum_i W_i \phi_k(x_i) \phi_\ell(x_i), \qquad b_k = (\phi_k, y) = \sum_i W_i \phi_k(x_i)\, y_i \ .$$

  - Só resta resolver $Mc = b$ e escrever a melhor aproximação como

$$p^*(x) = \sum_{k=1}^{n+1} c_k \phi_k(x) \ .$$

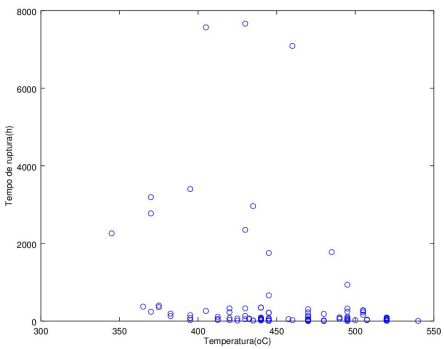
Miniprojeto

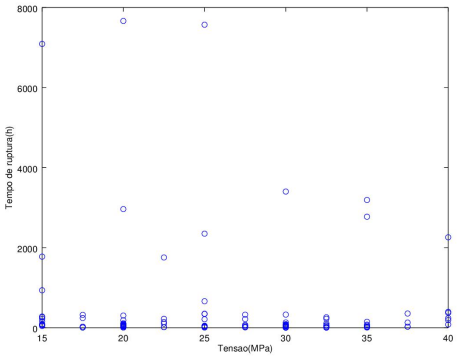
Uma empresa que vende peças metálicas que operam a altas temperaturas e tensões precisa fazer testes de resistência do seu material antes de vendê-lo. Dessa forma considere que os dados sobre os experimentos realizados pela empresa foram fornecidos por um arquivo de texto, que contém a tensão σ (MPa) aplicada ao material, a temperatura T(ºC) do experimento e t_R, o tempo de ruptura (em horas).

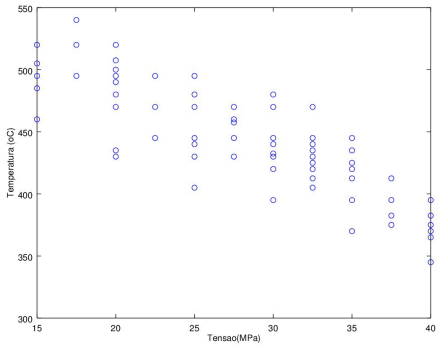


Procurando conseguir ajustar esses dados a alguma função, podemos verificar visualizando as variáveis

duas a duas que temos sempre uma nuvem de dados.



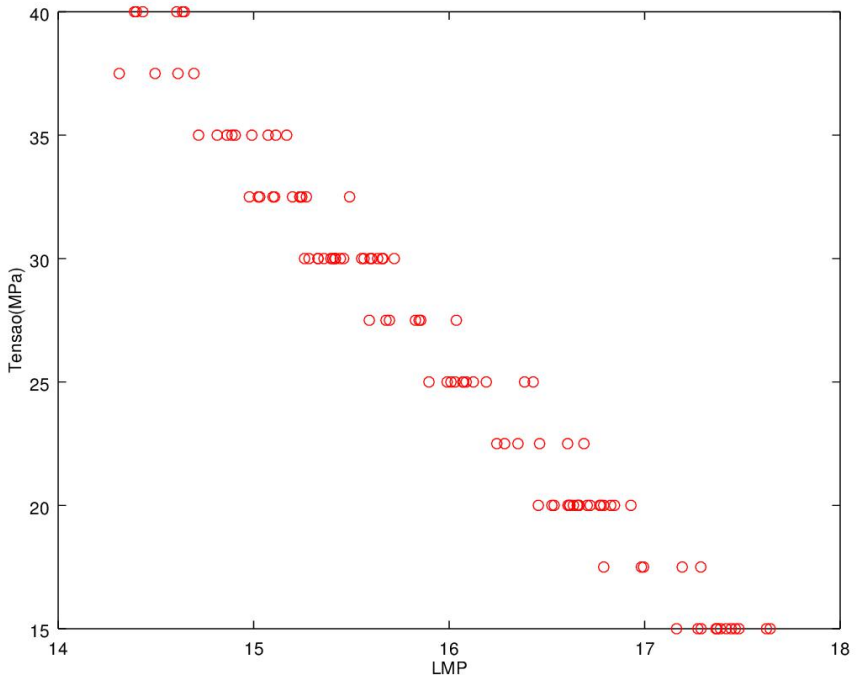




Para tentar relacionar melhor os parâmetros fazendo uma redução de dimensão do problema, introduz-se o parâmetro de Larson-Miller (LMP) que é calculado segundo dados do experimento pela equação:

$$LMP = \frac{(273 + T) \cdot (20 + log(t_R))}{1000},$$

em que $T(C)$ é a temperatura aplicada e t_R é o tempo de ruptura (em horas).


40
35
30
25
20
15
Tensao(MPa)
14
15
16
17
18
LMP

Proposta de trabalho:

1. Dado um conjunto de valores referentes a testes realizados por essa empresa `DadosFicticios.txt`, sabemos que estes devem ser ajustados com a equação de *Spera*:

$$LMP = a_1 + a_2 \log_{10}(\sigma) + a_3\sigma + a_4\sigma^2, \tag{23}$$

   em que os coeficientes a_1, a_2, a_3, a_4 são desconhecidos. Como melhor determinar estes coeficientes a partir dos dados fornecidos?

2. Agora, se um comprador potencial desta empresa especifica qual a temperatura e o tempo de ruptura que ele espera que o determinado material que ele irá comprar deverá suportar, somos capazes de dizer qual deve ser a tensão de ruptura deste material sob tais condições? Isto é, fornecendo o parâmetro de Larson-Miller na função (23) estimada pelo item anterior, como poderíamos estimar um valor de tensão máxima que o material pode suportar?

3. Programe uma função em Octave que dado qualquer valor para LMP, calcule a tensão máxima segundo a sua resposta ao item anterior.

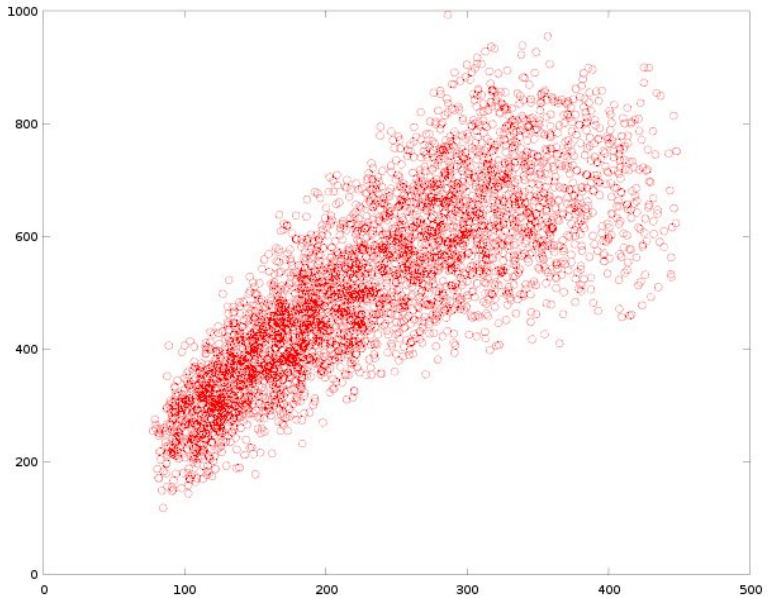
Exemplo de ajuste de dados:

Muitos problemas de engenharia começam com grandes matrizes de dados.

#	Terr m^2	Constr m^2	Ano	suites	quartos	banh	Plantas	Piscina m	Vagas	Seg24h 1=sim	Preço kR$
1	202	140	1998	0	3	3	2	0	2	0	423
2	250	137	2011	1	2	4	1	0	3	1	611
3	156	102	2001	1	1	3	1	0	1	1	354
4	353	182	2004	3	0	4	1	12	3	0	712
5	198	145	1983	2	1	5	1	0	2	0	387
6	376	251	2007	3	1	5	2	14	3	1	971
7	242	165	2015	1	3	4	2	12	3	1	765
8	177	133	1976	0	3	3	1	0	1	1	313
9	298	223	1997	3	0	5	1	0	2	1	789
10	422	351	2004	3	2	7	2	18	3	1	1310
11	202	140	1998	0	3	3	2	0	2	0	423
12	250	137	2011	1	2	4	1	0	3	1	611
13	156	102	2001	1	1	3	1	0	1	1	354
14	353	182	2004	3	0	4	1	12	3	0	712
15	198	145	1983	2	1	5	1	0	2	0	387
16	376	251	2007	3	1	5	2	14	3	1	971
17	242	165	2015	1	3	4	2	12	3	1	765
18	177	133	1976	0	3	3	1	0	1	1	313
19	298	223	1997	3	0	5	1	0	2	1	789
20	422	351	2004	3	2	7	2	18	3	1	1310
21	202	140	1998	0	3	3	2	0	2	0	423
22	250	137	2011	1	2	4	1	0	3	1	611
23	156	102	2001	1	1	3	1	0	1	1	354
24	353	182	2004	3	0	4	1	12	3	0	712
25	198	145	1983	2	1	5	1	0	2	0	387
26	376	251	2007	3	1	5	2	14	3	1	971
27	242	165	2015	1	3	4	2	12	3	1	765
28	177	133	1976	0	3	3	1	0	1	1	313
29	298	223	1997	3	0	5	1	0	2	1	789
30	422	351	2004	3	2	7	2	18	3	1	1310
31	202	140	1998	0	3	3	2	0	2	0	423
32	250	137	2011	1	2	4	1	0	3	1	611
33	156	102	2001	1	1	3	1	0	1	1	354
34	353	182	2004	3	0	4	1	12	3	0	712
35	198	145	1983	2	1	5	1	0	2	0	387
36	376	251	2007	3	1	5	2	14	3	1	971

- Poderia se procurar uma relação entre superfície construída e preço, por exemplo. Graficando as colunas uma como função da outra,

```
plot(Dados(:,3),Dados(:,12))
```



Preço vs. Superfície construída

- O gráfico mostra uma certa tendência, mas não fornece uma fórmula para realizar uma estimativa rápida.

  É popular procurar fórmulas lineares (afins), do tipo

  $$\text{preço} \simeq \text{constante}_1 + \text{constante}_2 \times \text{supconstr}$$

  Isto é, procurar k_1 e k_2 tais que

  $$\begin{array}{rcl} k_1 + S_1\,k_2 & = & P_1 \\ k_1 + S_2\,k_2 & = & P_2 \\ \cdots & & \cdots \\ k_1 + S_m\,k_2 & = & P_m \end{array} \Leftrightarrow \begin{pmatrix} 1 & S_1 \\ 1 & S_2 \\ \cdots & \cdots \\ 1 & S_m \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \end{pmatrix} = \begin{pmatrix} P_1 \\ P_2 \\ \cdots \\ P_m \end{pmatrix}$$

  Eis aqui um **sistema sobredeterminado**. Como discutido na teoria, podemos achar $k = (k_1, k_2)$ resolvendo as **equações normais**:

```
A=[ones(m,1),Dados(1:m,3)];
b=Dados(1:m,12);
k=(A'*A)\(A'*b); ## k=A\b seria equivalente!
```

  e, se queremos o valor estimado para uma superfície `sup`, ele simplesmente é igual a `k(1)`k(2)*sup+.

- O gráfico mostra também que a tendência dos dados não é exatamente uma reta, se observa uma concavidade para abaixo... Para modelar a concavidade é popular ajustar uma quadrática,

$$\text{preço} \simeq k_1 + k_2 \times \text{supconstr} + k_3 \times \text{supconstr}^2$$

$$\begin{array}{c} k_1 + S_1 k_2 + S_1^2 k_3 = P_1 \\ k_1 + S_2 k_2 + S_2^2 k_3 = P_2 \\ \cdots \\ k_1 + S_m k_2 + S_m^2 k_3 = P_m \end{array} \Leftrightarrow \begin{pmatrix} 1 & S_1 & S_1^2 \\ 1 & S_2 & S_2^2 \\ \cdots & \cdots & \cdots \\ 1 & S_m & S_m^2 \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} = \begin{pmatrix} P_1 \\ P_2 \\ \cdots \\ P_m \end{pmatrix}$$

Eis aqui mais um **sistema sobredeterminado** que montamos e resolvemos fazendo

```
A=[ones(m,1),Dados(1:m,3),Dados(1:m,3).^2];
b=Dados(1:m,12);
k=(A'*A)\(A'*b); ## k=A\b seria equivalente!
```

Assim,

```
preço = k(1)+k(2)*a+k(3)*a.^2
```

seria a melhor aproximação no valor `a`.

- Podemos suspeitar que é necessário incluir na estimativa a **superfície do terreno**:

$$\text{preço} \simeq k_1 + k_2 \times \text{supconstr} + k_3 \times \text{supterr}$$

$$\begin{pmatrix} 1 & S_1 & T_1 \\ 1 & S_2 & T_2 \\ \dots & \dots & \dots \\ 1 & S_m & T_m \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} = \begin{pmatrix} P_1 \\ P_2 \\ \dots \\ P_m \end{pmatrix}$$

Então o ajuste seria

```
A=[ones(m,1),Dados(1:m,3),Dados(1:m,2)];
b=Dados(1:m,12);
k=(A'*A)\(A'*b); ## k=A\b seria equivalente!
```

e o preço para superfície construida `a` e superfície do terreno `b` seria

```
preço = k(1)+k(2)*a+k(3)*b
```

Exercícios:

1. Determinar o polinômio linear $p^*(x, y)$ tal que

$$\int_0^1 \int_0^1 (f(x,y) - p^*(x,y))^2 \, dx \, dy \leq$$

$$\leq \int_0^1 \int_0^1 (f(x,y) - p(x,y))^2 \, dx \, dy$$

   para todo $p(x, y)$ linear. Considere $f(x, y) = e^{-2x-y}$. Plote o resultado.

2. Seja $D \in \mathbb{R}^{m \times N}$ uma matriz de dados ($m \gg 1$). A quantidade representada numa coluna i qualquer será chamada de d_i (e.g., superfície, preço, etc.). Se deseja ajustar a coluna `k` (e.g., preço de uma casa) como um polinômio de segundo grau da coluna `j` (e.g., área coberta da casa) e calcular o valor do polinômio ajusta quando d_j é igual a $a = 150\ m^2$.

   Diga quais dos seguintes códigos estão corretos:

   - ```
     A=[ones(m,1),D(:,j),D(:,j).*D(:,j)];
     z=[1,a,a^2]';
     b=D(:,k);
     x=A\b;
     val=z'*x;
     ```
   - ```
     A=[D(:,j).*D(:,j),D(:,j),ones(m,1)];
     z=[a^2,a,1];
     b=D(:,k);
     x=A\b;
     val=z*x;
     ```

- ```
  A=[ones(m,1),D(:,j),D(:,j).*D(:,j)];
  b=D(:,k);
  x=A\b;
  val=x(1)+x(2)*a+x(3)*a*a;
  ```
- ```
  A=[D(:,j).*D(:,j),D(:,j),ones(m,1)];
  b=D(:,k);
  x=A\b;
  val=x(1)+x(2)*a+x(3)*a*a;
  ```
- ```
  A=[ones(m,1),D(:,j),D(:,j).*D(:,j)];
  b=D(:,k);
  x=A\b;
  val=x'*a;
  ```
- ```
  A=[ones(m,1),D(:,j),D(:,j).*D(:,j)];
  b=D(:,k);
  x=A\b;
  val=1+a*x+a^2*x^2;
  ```
- ```
  A=[1,D(:,j),D(:,j)*D(:,j)];
  b=D(:,k);
  x=A\b;
  val=x(1)+x(2)*a+x(3)*a*a;
  ```
  ```

5 Integração e diferenciação numéricas

- **Problema:** Seja f uma função de $I \subset \mathbb{R}$ (intervalo) em $\mathbb{R}$. Sejam $x_1, x_2, \ldots$ pontos em I. Em cada ponto, se dispõe de um valor, $y_1, y_2, \ldots$ (reais, por simplicidade) que foi obtido **avaliando** a função f nos pontos x correspondentes. O **processo de avaliação** (ou **medição**) comporta um erro ϵ, que é independente medição a medição e com uma certa pdf, por exemplo $N(0, \sigma)$. Assim,

$$y_i = f(x_i) + \epsilon_i \ .$$

  O que se deseja é uma fórmula $\tilde{L}(x, y)$ que permita estimar, ou aproximar, funcionais lineares $L(f)$ tais como

  - Filtragem: O valor em um certo $a \in I$,

$$L(f) = f(a) \ .$$

  - Integração: Com $a, b \in I$,

$$L(f) = \int_a^b f(x)\, dx \ .$$

  - Diferenciação: Com $a \in I$,

$$L(f) = f'(a) \ .$$

- Um caso exemplo de **filtragem**: Se dispõe de (x_i, y_i), ordenados $(x_1 < x_2 < ...)$ e equiespaçados $(\delta x = 1)$. Qual seria melhor estimativa de $f(0)$?
  - $\tilde{L}_1 = y_0$
  - $\tilde{L}_2 = (y_{-1} + y_0 + y_1)/3$
  - $\tilde{L}_3 = (y_{-2} + y_{-1} + y_0 + y_1 + y_2)/5$

  Certamente a resposta deve depender da função f e do ruido ϵ, já que:
  - Se $\epsilon = 0$, a fórmula L_1 é exata.
  - Se $\epsilon \neq 0$, por exemplo $\epsilon_i \sim N(0, \sigma)$, e f é **constante**,

$$f(0) - y_0 = \epsilon_0 \sim N(0, \sigma) \;\Rightarrow\; \langle (f(0) - L_1)^2 \rangle = \sigma^2 \,.$$

$$f(0) - L_2 = \frac{\epsilon_{-1} + \epsilon_0 + \epsilon_1}{3} \sim \frac{N(0, \sqrt{3}\sigma)}{3} = N(0, \sigma/\sqrt{3}) \;\Rightarrow\; \langle (f(0) - L_2)^2 \rangle = \frac{\sigma^2}{3} \,.$$

$$f(0) - L_3 = \frac{\epsilon_{-2} + \epsilon_{-1} + \epsilon_0 + \epsilon_1 + \epsilon_2}{5} \sim \frac{N(0, \sqrt{5}\sigma)}{5} = N(0, \sigma/\sqrt{5}) \;\Rightarrow\; \langle (f(0) - L_3)^2 \rangle = \frac{\sigma^2}{5} \,.$$

  Nesse caso $\tilde{L}_3$ é a mais precisa.

Texto básico: Capítulo 4 do livro de Quarteroni e Saleri (em português), páginas 101 a 122. Ler!

5.1 Integração

- A **integração numérica** consiste na aproximação de integrais, da forma

$$L(x,y) = \int_a^b f(s)\ ds \simeq \tilde{L}(x,y) = \sum_i w_i\, y_i\ . \tag{24}$$

- A combinação dos **pontos de quadratura** $\{x_i\}$ e dos **pesos de quadratura** conforma uma **regra de quadratura**.
- As principais propriedades se estudam considerando **ruido zero** ($\epsilon = 0$).
- Assim, $\tilde{L}(f) = \sum_i w_i\, f(x_i)$.
- Veja que $\tilde{L} : V \to \mathbb{R}$, assim como $L(f) = \int f$, são lineares em f. O espaço V depende da aplicação.
- Muitas **regras de quadratura** surgem da seguinte "receita":
  1. Se escolhe o conjunto de pontos: $\{x_i\}_{i=1}^n$. Por exemplo porque as medições são feitas a intervalos regulares.
  2. Se escolhe um espaço de **aproximação** K de dimensão finita m.
  3. Se **interpola** ou **aproxima** os dados $\{y_i\}$ com um certo $p \in K$.

     Por exemplo, se $K = \mathbb{P}_{m-1}$ e $m = n$, é possível utilizar a **interpolada polinomial de Lagrange**, i.e., o único $p \in K$ tal que

$$p(x_i) = y_i\ , \qquad \forall\ i = 1, \dots, n\ .$$

  4. Se **integra** el interpolante/aproximante p, i.e.,

$$\tilde{L}(x,y) = \int_a^b p(s)\ ds\ . \tag{25}$$

- No caso descrito acima, isto é, quando $K = \mathbb{P}_{n-1}$ (sendo n pontos dados) e a interpolante/aproximante é a interpolada de Lagrange, as fórmulas resultantes se chamam **Fórmulas de Newton-Cotes**.

- Tomando a **base canônica** de K, denotada por $\{\varphi_j\}_{j=1}^{m}$, que satisfaz

$$\varphi_j(x_i) = \delta_{ij} \ ,$$

(os **polinômios de Lagrange**, no caso), resulta que

$$\tilde{L}(x,y) = \int_a^b p(s)\ ds \ = \int_a^b \sum_i y_i \phi_i(s)\ ds = \sum_i \underbrace{\left(\int_a^b \phi_i(s)\, ds \right)}_{w_i} y_i \ .$$

Portanto, a **receita** acima proposta entrega uma **regra de quadratura**, de acordo com a nossa definição.

- **Por construção**, as regras de quadratura obtidas com a receita anterior **integram exatamente toda** $f \in K$.

- **Exercício: Newton-Cotes**

  Seja a seguinte regra de quadratura:

  - Divida (a, b) em $n-1$ segmentos iguais, e sejam $\{x_i = a + (i-1)(b-a)/(n-1), i = 1, \ldots, n\}$ os pontos de quadratura.
  - Tome $K = \mathbb{P}_{n-1}$ e calcule w_i. Realize o cálculo explícito para $n = 2$, 3 e 4.

- **Exercício: Regra dos Trapézios**

  Repita o exercício anterior tomando K como o espaço linear por partes P_1 contínuo discutido anteriormente.

- **Exercício: Regra de Simpson**

  Seja a seguinte regra de quadratura:

  - Divida (a, b) em k segmentos iguais, e sejam $\{x_i = a + (i-1)(b-a)/(n-1), i = 1, \ldots, n\}$ os pontos de quadratura, com $n = 2k + 1$. Assim, os pontos 1 e 3 são extremos do primeiro subintervalo, os pontos 3 a 5 do segundo, etc.
  - Tome $K = P_2$ contínuo, o conjunto das funções quadráticas por partes em cada subintervalo, contínuas.
  - Calcule $\{w_i\}$ explícitamente para $k = 1$, 2 e 3, e generalize para todo k.

- **Exercício: Regra do Ponto Médio**

  Identifique os pontos de quadratura e o espaço de aproximação K que correspondem à regra do ponto médio.

Todos os casos anteriores se generalizam facilmente a casos com pontos não equi-espaçados.

- **Exercício:**

  A concentração de carbono em um tubo de aço é dada por uma função $C(r)$, sendo $a \le r \le b$ (raios interno e externo). A massa total de carbono no tubo de comprimento L é, então,

$$M = 2\pi L \int_a^b C(r)\, r\, dr \ .$$

  - Calcule uma **regra de quadratura de um ponto** (r_1, w_1) que integre exatamente **todas as funções lineares**, isto é, todas as $C(r) = \alpha + \beta\, r$, com $\alpha, \beta \in \mathbb{R}$.
  - Para que valores tende r_1 quando $(b-a) \ll a$? E quando $b \gg a$? É razoável?
  - Calcule uma regra de quadratura cujos pontos sejam $r_1 = a$ e $r_2 = b$, que integre exatamente todas as funções lineares.

- **Exercício (facultativo, mas o resultado é importante!):**

Mudança de variável em integração numérica: Seja $\{(\hat{x}_i, \hat{w}_i)\}$ uma regra de quadratura que integra exatamente todos os polinômios de $\mathbb{P}_k$ no intervalo $\hat{I} = [-1, 1]$. Provar que então $\{(x_i, w_i)\}$, onde

$$x_i = a + \frac{\hat{x}_i + 1}{2}\,(b-a)\ , \qquad w_i = \frac{b-a}{2}\,\hat{w}_i\ ,$$

é uma regra de quadratura em $I = (a, b)$ que integra exatamente $\mathbb{P}_k$.

- **Exercício (integração numérica 2D):**
  - Determinar uma **regra de quadratura de 1 ponto** $\{r_1 = (x_1, y_1), w_1\}$ para a integral

$$I(f) = \int_{\square} f(x, y)\, dx\, dy$$

    onde $\square$ é o retângulo $(0, 1) \times (0, 2)$. Deseja-se uma regra que integre exatamente polinômios $\mathbb{P}_1$, i.e., $p(x, y) = a + bx + cy$.
  - Determinar **um segundo ponto de quadratura** $r_2 = (x_2, y_2)$ em $\square$, de tal maneira que, com pesos adequados $\tilde{w}_1$ e $\tilde{w}_2$, a regra de quadratura de dois pontos $\{r_i, \tilde{w}_i\}$ integre exatamente todos os polinômios da forma $p(x, y) = a + bx + cy + dxy$.

5.2 Diferenciação numérica

- A **diferenciação numérica** consiste na aproximação de derivadas, num certo ponto a, a partir de valores (x_i, y_i) em pontos x_i vizinhos de a. A linearidade da operação de diferenciação leva a fórmulas da forma
$$L(f) = f'(a) \simeq \tilde{L}(x, y) = \sum_i w_i \, y_i \ . \tag{26}$$
- As principais propriedades se estudam considerando **ruido zero** ($\epsilon = 0$).
- Assim, $\tilde{L}(f) = \sum_i w_i \, f(x_i)$.
- Muitas **regras de diferenciação numérica** surgem da seguinte "receita":
  1. Se escolhe o conjunto de pontos: $\{x_i\}_{i=1}^n$. Por exemplo porque as medições são feitas a intervalos regulares.
  2. Se escolhe um espaço de **aproximação** K de dimensão finita m.
  3. Se **interpola** ou **aproxima** os dados $\{y_i\}$ com um certo $p \in K$.

     Por exemplo, se $K = \mathbb{P}_{m-1}$ e $m = n$, é possível utilizar a **interpolada polinomial de Lagrange**, i.e., o único $p \in K$ tal que
     $$p(x_i) = y_i \ , \qquad \forall \ i = 1, \dots, n \ .$$
  4. Se **deriva** el interpolante/aproximante p, i.e.,
     $$\tilde{L}(x, y) = p'(s) \ . \tag{27}$$
- No caso da interpolada polinomial de Lagrange, por exemplo, sabemos que $p(x) = \sum_i y_i \, \ell_i(x)$, e portanto,
$$\tilde{L}(x, y) = \sum_i \underbrace{\ell_i'(a)}_{w_i} \ y_i \ .$$

- **Exemplo: Diferenças centradas.** Sejam os pontos equiespaçados, a distância h, e consideremos querer aproximar $f'(x_j)$ (para um dado j fixo a partir dos dados (x_{j-1}, y_{j-1}), (x_j, y_j) e (x_{j+1}, y_{j+1}). Tendo três pontos, podemos utilizar $K = \mathbb{P}_2$.

  A construção de $p(x)$ é fácil, já que é o único polinômio quadrático que vale y_{j-1} em $x_j - \delta x$, y_j em x_j e y_{j+1} em $x_j + \delta x$ (verificar):

$$p(x) = y_j + \frac{y_{j+1} - y_{j-1}}{2h}(x - x_j) + \frac{y_{j+1} - 2y_j + y_{j-1}}{2h^2}(x - x_j)^2 \ .$$

  Então,

$$\tilde{L}(x, y) = p'(x_j) = \frac{y_{j+1} - y_{j-1}}{2h} \ .$$

  O **erro de consistência** dessa fórmula, considerando $h \to 0$, é

$$f'(x_j) - \tilde{L}(x, y) = f'(x_j) - \frac{f(x_{j+1}) - f(x_{j-1})}{2h}$$

$$= f'(x_j) - \frac{f(x_j) + f'(x_j)h + f''(x_j)\frac{h^2}{2} + f'''(\xi_1)\frac{h^3}{6} - \left[f(x_j) - f'(x_j)h + f''(x_j)\frac{h^2}{2} - f'''(\xi_2)\frac{h^3}{6}\right]}{2h}$$

$$= -\frac{f'''(\xi_1) + f'''(\xi_2)}{12}h^2 = O(h^2) \ .$$

  Considerando o ruido,

$$f'(x_j) - \tilde{L}(x, y) = -\frac{f'''(\xi_1) + f'''(\xi_2)}{12}h^2 + \frac{\epsilon_1 - \epsilon_2}{2h}$$

$$= -\frac{f'''(\xi)}{6}h^2 + \frac{\epsilon}{\sqrt{2}h}$$

  porque $\langle \epsilon_1 - \epsilon_2, \epsilon_1 - \epsilon_2 \rangle = \langle \epsilon_1, \epsilon_1 \rangle + \langle \epsilon_2, \epsilon_2 \rangle = 2\sigma^2$.

- **Exemplo: Diferenças unilaterais.**

  Limitando os pontos a x_{j-1} e x_j é possível interpolar um polinômio de $\mathbb{P}_1$.

$$p(x) = y_j + \frac{y_j - y_{j-1}}{h}(x - x_j) \ .$$

  De onde

$$\tilde{L}(x, y) = p'(x_j) = \frac{y_j - y_{j-1}}{h} \ .$$

  Assim

$$f'(x_j) - \tilde{L}(x, y) = f'(x_j) - \frac{f(x_j) - [f(x_j) - f'(x_j)h + f''(\xi)h^2/2]}{h} = \frac{f''(\xi)}{2} \ h \ = O(h) \ ,$$

  assim, trata-se de uma fórmula de primeira ordem.

  Considerando o ruído chegamos a

$$f'(x_j) - \tilde{L}(x, y) = \ C \ h \ f''(\xi) + \ \gamma \ \frac{\epsilon}{h} \ ,$$

  com $C = 1/2$ e $\gamma = \sqrt{2}$.

- **Teorema: O erro de consistência** de uma fórmula $\tilde{L}(x,y)$ que pretende aproximar $L(f) = f'(x_j + a\,h)$ (f função suave), sendo que cada avaliação y_i apresenta ruído ϵ_i não correlacionado (i.e., $y_i = f(x_i) + \epsilon_i$, com $\langle \epsilon_i \rangle = 0$, $\langle \epsilon_i \epsilon_j \rangle = \sigma^2\,\delta_{ij}$) e que a fórmula é exata para polinômios de grau k (i.e., quando $y_i = p(x_i)$, $p \in \mathbb{P}_k$) é

$$f'(x_j + a\,h) - \tilde{L}(x,y) = \; C\; h^k\; f^{(k+1)}(\xi) \;+\; \gamma\,\frac{\epsilon}{h}$$

Dessa maneira, escrevendo a fórmula como

$$f'(x_j + a\,h) \simeq \tilde{L}(x,y) = \frac{\ldots + w_{-3}\,y_{j-3} + w_{-2}\,y_{j-2} + w_{-1}\,y_{j-1} + w_0\,y_j + w_1\,y_{j+1} + w_2\,y_{j+2} + \ldots}{h} = \frac{1}{h}\sum_{i=-n_-}^{n_+} w_i y_{j+i}\,,$$

os valores de $w_i \in \mathbb{R}$ estão totalmente determinados pelos números de pontos "à ezquerda" (n_-) e "à direita" (n_+) e pelo grau do polinômio derivado exatamente, k. Para cada fórmula, pode-se calcular γ como $\gamma = \sqrt{\sum_i w_i^2}$ e C como $p'(x_j + a\,h)$, onde p é o polinômio de grau $\leq k$ tal que $p(x_{j+i}) = \frac{(x_{j+i} - x_j - a\,h)^{k+1}}{(k+1)!}$.

- **Uma fábrica de fórmulas:**

  Fazendo a mudança de variável $\hat{x} = (x - x_j - a\,h)/h$ teremos os nós $\hat{x}_i$ a distância 1, equiespaçados. Notar que

  $$\frac{d^\ell f}{dx^\ell}(x_j + a\,h) = \frac{1}{h^\ell}\,\frac{d^\ell f}{d\hat{x}^\ell}(0)\ .$$

  Ajustaremos um polinômio $p(\hat{x})$ a partir dos pares de valores $(\hat{x}_i, y_{j-n_-+i-1})$. Devemos decidir o grau k do polinômio e $n_\pm$ tais que os nós da fórmula sejam $j-n_-, j-n_-+1, \ldots, j, j+1, \ldots, j+n_+$. O número de pontos será $n = n_- + n_+ + 1$. Com apenas esses dados podemos calcular a matriz de ajuste (notar que `x` é vetor coluna):

```
np=0; nm=2; k=2; a=0; ##dados
n=nm+np+1;
x=([-nm:1:np]-a)';
M=ones(n,1);
for i=1:k
  M=[M,x.^i];
endfor
```

  No caso acima temos 3 pontos, i.e., $j-2$, $j-1$ e j, e queremos aproximar $f'(x_j)$ (porque $a = 0$). O ajuste do polinômio $\hat{p}$ é da forma

  $$\hat{p}(\hat{x}) = c_1 + c_2\,\hat{x} + c_3\,\hat{x}^2 \quad \in \mathbb{P}_2\ ,$$

  o que faz que $\hat{p}(0) = c_1$, $\hat{p}'(0) = c_2$ e $\hat{p}''(0) = 2\,c_3$.

  Assim, para qualquer vetor (coluna) $y(1:n)$ temos que o polinômio $\hat{p}$ que satisfaz $\hat{p}(\hat{x}_i) = y_i$ tem por coeficientes o resultado de `c=M\y`. Ainda, $c(1) = \hat{p}(0)$, $c(2) = \hat{p}'(0)$, etc.

Agora consideremos o resultado de fazer

```
 >> c(:,1)=M\[1;zeros(n-1,1)]
c =
   0.00000
   0.50000
   0.50000
```

Isto quer dizer que $p_1(\hat{x}) = 0 \times 1 + 0.5\ \hat{x} + 0.5\ \hat{x}^2$ é o polinômio que corresponde aos dados $y = (1, 0, \dots, 0)^T$. Em particular, a esses dados corresponde

$$p_1(x_j) = \hat{p}_1(0) = c(1,1) = 0, \quad p_1'(x_j) = \frac{\hat{p}_1'(0)}{h} = \frac{c(2,1)}{h} = \frac{1}{2h}, \quad p_1''(x_j) = \frac{p_1''(0)}{h^2} = \frac{2\,c(3,1)}{h^2} = \frac{1}{h^2}\ .$$

Se substituirmos o vetor $(1, 0, \dots, 0)^T$ por $e^i = (0, \dots, 0, 1, 0, \dots)^T$, com o 1 na linha i, construiremos o polinômio

$$\hat{p}_i(\hat{x}) = c(1,i) + c(2,i)\,\hat{x} + c(3,i)\,\hat{x}^2$$

que ajusta o dado e^i. O interessante é que toda a matriz c pode ser calculada fazendo, simplesmente,

```
>> c=M\eye(n)
c =
   0.00000    0.00000    1.00000
   0.50000   -2.00000    1.50000
   0.50000   -1.00000    0.50000
```

o que quer dizer que

$$p(x_j) = 0\,y_{j-2} + 0\,y_{j-1} + 1\,y_j\ , \quad p'(x_j) = \frac{0.5\,y_{j-2} - 2\,y_{j-1} + 1.5\,y_j}{h}\ , \quad p''(x_j) = \frac{2\ (0.5\,y_{j-2} - 1\,y_{j-1} + 0.5\,y_j)}{h^2}\ .$$

Notar que, para todo $0 \leq \ell \leq k$

$$p^{(\ell)}(x_j + a\,h) = \frac{\ell!}{h^\ell} \sum_{i=1}^{n} c(\ell+1, i) y_{j-n_-+i-1} \ .$$

Não deixar de notar que para obter os coeficientes deve-se multiplicar a linha $\ell + 1$ da matriz c por $\ell!/h^\ell$.

- **Resumo:** O código

```
np=3; nm=3; k=6; a=0; ##dados
n=nm+np+1;
x=[-nm:1:np]-a;
M=ones(n,1);
for i=1:k
  M=[M,x'.^i];
endfor
c=M\eye(n)
c(2,:)
gamma=norm(c(2,:))
## erro de consistencia
pol=x.^(k+1)./factorial(k+1);
k+1
cerr=-c(2,:)*pol'
pol=x.^(k+2)./factorial(k+2);
k+2
cerr=-c(2,:)*pol'
```

calcula a matriz c de coeficientes, dos quais a linha 2 são os coeficientes utilizados pela derivada primeira em $x_j + a\,h$. As constantes γ e C da fórmula do erro de consistência estão nas linhas seguintes.

O resultado é

```
c =
 0.00000  0.00000 -0.00000  1.00000  0.00000 -0.00000 -0.00000
-0.01667  0.15000 -0.75000 -0.00000  0.75000 -0.15000  0.01667
 0.00556 -0.07500  0.75000 -1.36111  0.75000 -0.07500  0.00556
 0.02083 -0.16667  0.27083  0.00000 -0.27083  0.16667 -0.02083
-0.00694  0.08333 -0.27083  0.38889 -0.27083  0.08333 -0.00694
-0.00417  0.01667 -0.02083 -0.00000  0.02083 -0.01667  0.00417
 0.00139 -0.00833  0.02083 -0.02778  0.02083 -0.00833  0.00139
gamma =  1.0819
cerr =  -0.0071429
```

indicando as seguintes fórmulas centradas de 7 pontos:

$$
\begin{aligned}
f(x_j) &\simeq y_j \\
f'(x_j) &\simeq D_h^1 f = -\frac{1}{60h}y_{j-3} + \frac{3}{20h}y_{j-2} - \frac{3}{4h}y_{j-1} + \frac{3}{4h}y_{j+1} - \frac{3}{20h}y_{j+2} + \frac{1}{60h}y_{j+3} \\
f''(x_j) &\simeq D_h^2 f = 2\,\frac{0.00556y_{j-3} - 0.075y_{j-2} + 0.75y_{j-1} - 1.36111y_j + 0.75y_{j+1} - 0.075y_{j+2} + 0.00556y_{j+3}}{h} \\
\ldots &\ \ldots\ \ldots
\end{aligned}
$$

e em particular que o erro de consistência satisfaz

$$f'(x_j) - D_h^1 f = -0.0071429 \times h^6 \times f^{(7)}(\xi) + 1.0819 \times \frac{\epsilon}{h}\ .$$

- **Exemplo: Fórmulas no ponto médio.**

  Colocando $n_- = 0$, $n_+ = 1$, $a = 0.5$ e $k = 1$ obtemos

```
c =
   0.50000   0.50000
  -1.00000   1.00000
gamma =  1.4142
ans =  2
cerr = 0
ans =  3
cerr =  -0.041667
```

  isto é, as fórmulas

$$f(x_j + h/2) \simeq 0.5\, y_j + 0.5\, y_{j+1}\ , \quad f'(x_j + h/2) \simeq \frac{y_{j+1} - y_j}{h}\ .$$

  e sabemos que o erro de aproximação da derivada é de $-0.041667h^2 f^{(3)}(\xi) + 1.4142\sigma/h$ e não $O(h) + 1.4142\sigma/h$, visto que $C = 0$ para o primeiro caso. Algumas vezes uma fórmula calculada para ser exata para $\mathbb{P}_k$ o é também para $\mathbb{P}_{k+1}$ automaticamente.

- **Exercício:** Colocando $n_- = 1$, $n_+ = 2$, $a = 0.5$ e $k = 3$ obtemos

```
c =
  -0.062500   0.562500   0.562500  -0.062500
   0.041667  -1.125000   1.125000  -0.041667
   0.250000  -0.250000  -0.250000   0.250000
  -0.166667   0.500000  -0.500000   0.166667
gamma =  1.5921
ans =  4
cerr =  0
ans =  5
cerr = 0.0046875
```

e assim as fórmulas

$$
\begin{aligned}
f(x_j + h/2) &\simeq -0.0625 y_{j-1} + 0.5625 y_j + 0.5625 y_{j+1} - 0.0625 y_{j+2} \\
f'(x_j + h/2) &= \frac{0.041667 y_{j-1} - 1.125 y_j + 1.125 y_{j+1} - 0.041667 y_{j+2}}{h} + 0.0046875\, h^5\, f^{(6)}(\xi) + 1.5921\, \frac{\epsilon}{h} \\
f''(x_j + h/2) &\simeq 2\, \frac{0.25 y_{j-1} - 0.25 y_j - 0.25 y_{j+1} + 0.25 y_{j+2}}{h^2} \\
&\ldots \quad \ldots \quad \ldots
\end{aligned}
$$

Verificar as fórmulas acima, e calcular os erros da primeira e da terceira.

- **Exercício (derivadas com segunda ordem de convergência com pontos equiespaçados):** Quais são as fórmulas com segunda ordem de convergência e **o mais compactas possíveis** (i.e., envolvendo vizinhos o mais próximos possíveis) para aproximar
  - a primeira derivada $f'(x_j)$,
  - a segunda derivada $f''(x_j)$,
  - a terceira derivada $f'''(x_j)$,
  - a quarta derivada $f^{(4)}(x_j)$,

  sendo que os pontos $\{x_i\}$ estão equiespaçados a distância h e que se considera conhecidos $y_i = f(x_i)$ para todo i.

6 Solução numérica de equações diferenciais ordinárias

6.1 Introdução

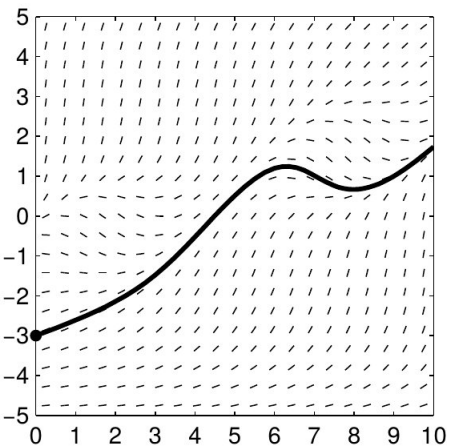
- Dada uma função $y(t)$, diferenciável em $I = (t_0, t_f)$, dizemos que y satisfaz a **EDO**

$$y' = f(t, y) \tag{28}$$

  em I se, para cada $t \in I$, $y'(t) = f(t, y(t))$.

- A função f define um "campo de inclinações" em qualquer retângulo $(t_0, t_f) \times (a, b)$.

FIGURE 5.1. The slopefield for $f(t,y) = [(y+1-t/3)^2 - \sin t]e^{-.15(y-1)^2}$ and the solution to $y'(t) = f\big(t, y(t)\big)$, $y(0) = -3$.



Tomado de Arnold, D.

- A EDO define-se componente a componente para $y(t) \in \mathbb{R}^n$.
- O **Problema de Valor Inicial** (PVI) é: Determinar $y : (t_0, t_f) \to \mathbb{R}$ tal que

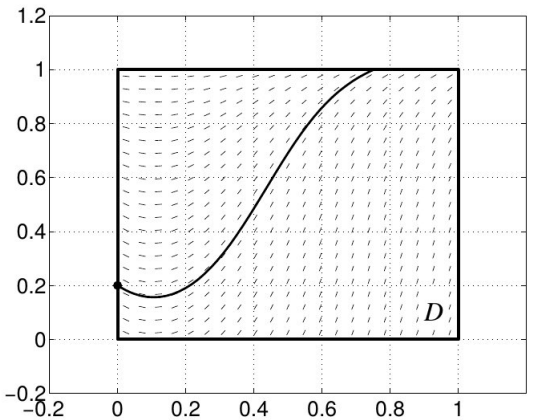
$$y'(t) = f(t, y(t)) \ , \qquad y(t_0) = z \in \mathbb{R}^n \ , \tag{29}$$

  onde $f : (t_0, t_f) \times B \to \mathbb{R}^n$ e $z \in B$.
- **Teorema (Picard):** Se f é continua e satisfaz a condição de Lipschitz

$$\|f(t, y) - f(t, x)\| \leq L \|y - x\| \ , \qquad \forall\, x, y \in B \ ,$$

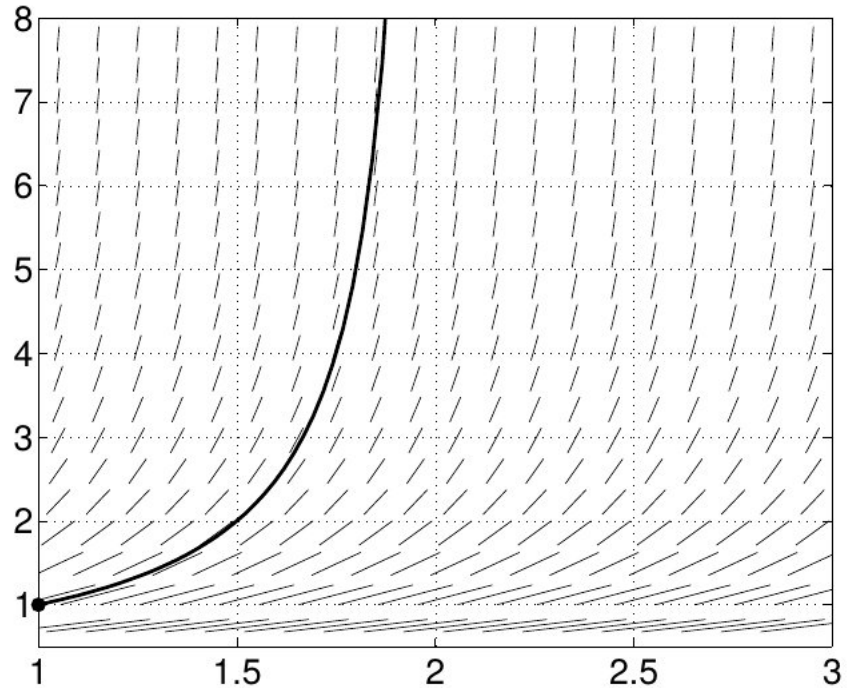
  para algum $L > 0$, então o PVI tem solução única em (t_0, t_*).

FIGURE 5.2. The solution to $y' = f(t, y)$, $y(0) = .2$. In this example the domain D of f is the unit square, and the solution curve leaves D at $t = .75$, so there is no solution to the initial value problem defined on the whole interval $I = [0, 1]$.



Tomado de Arnold, D.

FIGURE 5.3. The solution to $y' = y^2$, $y(1) = 1$ cannot be continued to $t \geq 2$.



- Um PVI é também uma equação integral,

$$y(t) = (Ty)(t) = z + \int_{t_0}^{t} f(s, y(s))\, ds\ ,$$

onde ressaltamos que a solução é um ponto fixo do operador T no conjunto de funções de $C^0(\bar{I})$ que satisfazem $y(t_0) = z$.

- **Teorema:** Nas condições do Teorema de Picard, duas soluções y e w da EDO com diferentes condições iniciais satisfazem

$$\|y(t) - w(t)\| \leq \|y(t_0) - w(t_0)\| \exp L(t - t_0)\ . \tag{30}$$

Assim, há **estabilidade** em intervalos de integração finitos.

- A estimativa anterior pode ser super-pessimista. Quando $\|y(t) - w(t)\| \to 0$ para $t \to \infty$ se diz que a solução y é assintóticamente estável.

- Uma EDO autônoma $y' = f(y)$ é um campo vetorial em B. Pode ser visto como um **campo de velocidade** independente do tempo. As soluções são **trajetórias**:

$$\varphi(t)\ , \qquad \varphi'(t) = f(\varphi(t))\ , \qquad \varphi(t_0) = z\ .$$

- Equações diferenciais com derivadas de ordem > 1 transformam-se a primeira ordem acrescentando incógnitas. EDOs não autônomas transformam-se em autônomas acrescentando $y_1(t)$, com $y_1(0) = t_0$ e $y_1'(t) = 1$.

Dinâmica de populações: Equações de Lotka-Volterra.

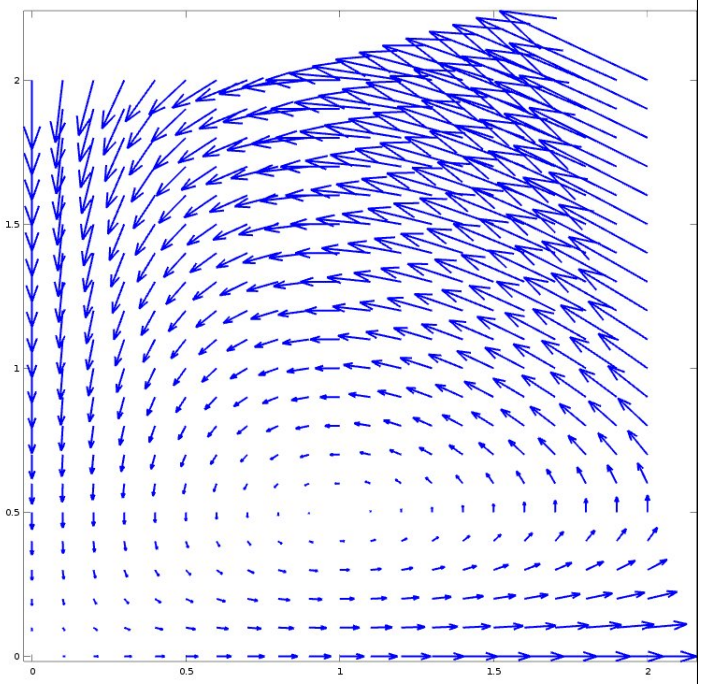
$$\frac{dy_1}{dt} = \alpha y_1 - \beta y_1 y_2$$
$$\frac{dy_2}{dt} = \delta y_1 y_2 - \gamma y_2$$

- Equação autônoma.

$$f(y) = (\alpha y_1 - \beta y_1 y_2, \delta y_1 y_2 - \gamma y_2)^T$$

- Equilíbrios: $(0,0)$ e $(\gamma/\delta, \alpha/\beta)$.
- Caso $\alpha = 2/3$, $\beta = 4/3$, $\gamma = \delta = 1$.

```
u=@(x,y) 2/3*x-4/3*x.*y;
v=@(x,y) x.*y-y;
range=linspace(0,2,21);
[X, Y]=meshgrid(range,range);
h=quiver(X,Y,u(X,Y),v(X,Y),4);
axis tight
set (h,"linewidth",2)
```



Constantes de movimento: Seja $K = y_2^{\alpha} e^{-\beta y_2} y_1^{\gamma} e^{-\delta y_1}$, então $dK/dt = 0$.

$$\frac{dK}{dt} = \left[\left(\frac{\gamma}{y_1} - \delta \right) (\alpha y_1 - \beta y_1 y_2) + \left(\frac{\alpha}{y_2} - \beta \right) (\delta y_1 y_2 - \gamma y_2) \right] K = 0$$

Pêndulo simples: Comprimento ℓ, massa m, gravidade g.

$$\frac{d^2\theta}{dt^2} = -\omega^2 \sin\theta\ , \quad \theta(0) = a,\ \theta'(0) = v\ .$$

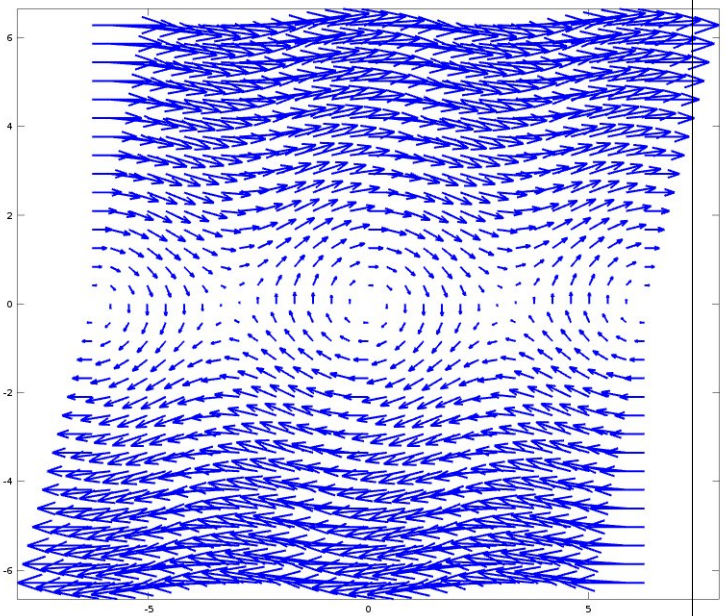
- Transformamos à forma $y' = f(t, y)$:

$$\begin{aligned} \frac{dy_1}{dt} &= y_2 \\ \frac{dy_2}{dt} &= -\omega^2 \sin y_1 \end{aligned}$$

  com condição inicial $y(0) = (a, v)^T$.

- Equação autônoma.
- Equilíbrios: $(0, 0)$ e $(\pi, 0)$.

```
om=1;
u=@(x,y) y;
v=@(x,y) -om*om*sin(x);
range=linspace(-2*pi,2*pi,31);
[X, Y]=meshgrid(range,range);
h=quiver(X,Y,u(X,Y),v(X,Y),3);
axis tight
set (h,"linewidth",2)
```



Pêndulo duplo: Comprimentos ℓ_1, ℓ_2, etc.

- EDO autônoma. O seguinte código calcula f:

```
function f = fpend2(q,t)
global g l1 l2 m1 m2
A(1,1) = (m1+m2)*l1;           A(1,2) = m2*l2*cos(q(1)-q(2));
A(2,1) = m2*l1*cos(q(1)-q(2)); A(2,2) = m2*l2;
F(1,1) = -m2*l2*sin(q(1)-q(2))*v(2)^2-(m1+m2)*g*sin(q(1));
F(2,1) =  m2*l1*sin(q(1)-q(2))*v(1)^2-m2*g*sin(q(2));
f(1:2) = q(3:4);  f(3:4) = A\F;
```

- O seguinte código resolve com `lsode`. Ver `pend2.m` no site.

```
global g l1 l2 m1 m2
dt = 0.1;  N = 2000; T=[0:dt:N*dt];
g = 1; l1 = 1; m1 = 1; l2 = 0.3; m2 = 1.1;
[q, istate, MSG]=lsode("fpend2",[2*pi/3 0 0 0],T);
%% Recovering cartesian coordinates
x1 =  l1*sin(q(:,1)); y1 = -l1*cos(q(:,1));
x2 = x1 + l2*sin(q(:,2)); y2 = y1 - l2*cos(q(:,2));
```

- Sistema simples que desenvolve comportamento caótico.

```
http://web.mit.edu/jorloff/www/chaosTalk/double-pendulum/double-pendulum-en.html
https://www.youtube.com/watch?v=d0Z8wLLPNE0
https://www.youtube.com/watch?v=_A4ahQKYjVQ
```

- **Exercício:** Resolva o sistema de Lotka-Volterra com a função `lsode`.
- **Exercício:** Modifique o código `pend2.m` para que calcule e grafique a energia cinética e potencial do pêndulo. Elas são

$$T = \frac{1}{2} m_1 (\dot{x}_1^2 + \dot{y}_1^2) + \frac{1}{2} m_2 (\dot{x}_2^2 + \dot{y}_2^2) \ , \quad V = g \left(m_1 y_1 + m_2 y_2 \right) \ .$$

- **Exercício:** O oscilador de van der Pol é um exemplo clássico de sistema auto-excitado,

$$x''(t) = -x(t) + a \left(x(t)^2 - 1 \right) x'(t) + b \cos \omega t \ , \qquad x(0) = x_0, \ x'(0) = v_0 \ , \tag{31}$$

  onde o caso $b \neq 0$ corresponde ao sistema forçado, e $a, b \in \mathbb{R}$.

  - Transforme em sistema de EDOs, havendo duas maneiras clássicas:

$$\begin{cases} x' &= a \left(x - \frac{1}{3} x^3 - w \right) , \\ w' &= \frac{1}{a} x \end{cases} \quad , \text{ e } \quad \begin{cases} x' &= w , \\ w' &= a \left(1 - x^2 \right) w - x \end{cases}$$

  - Desenhe o campo vetorial que corresponde ao sistema autônomo ($b = 0$).
  - Escreva um código que resolva numéricamente o oscilador vdP com e sem termo forçante.
  - Investigue numericamente o seguinte **Teorema:** O oscilador vdP tem exatamente um ciclo limite e todas as trajetórias se aproximam a ele. Determine numericamente ciclos limites para diferentes valores de $a > 0$, indo desde $a \ll 1$ até $a \gg 1$.

6.2 Métodos numéricos

Um **método numérico** para "resolver" o PVI

$$y'(t) = f(t, y(t)) \ , \quad y(t_0) = y^0 \ ,$$

é uma **receita** para gerar uma **sequência** de vetores Y^0, Y^1, ..., que **aproxima** a sequência $y(t_0)$, $y(t_1)$, ..., para uma **sequência de tempos** $t_0, t_1, \ldots$.
Atenção: (1) Os supra-índices não são expoentes. (2) Quando não houver risco de confusão, também usaremos $y^1, y^2, \ldots$ para denotar a solução numérica.

Algumas definições:

- **Método geral de um passo:**

$$Y^{n+1} = Y^n + \delta t \, \Phi(t_n, \delta t, Y^n, Y^{n+1}) \tag{32}$$

- Exemplo: Método θ.

$$\Phi(t_n, \delta t, Y^n, Y^{n+1}) = (1-\theta) f(t_n, Y^n) + \theta f(t_n + \delta t, Y^{n+1})$$

  Resulta assim que

$$\lim_{\delta t \to 0} \frac{Y^{n+1} - Y^n}{\delta t} = \lim_{\delta t \to 0} \left[(1-\theta) f(t_n, Y^n) + \theta f(t_n + \delta t, Y^{n+1}) \right] = f(t_n, Y^n) \ .$$

- Se a sequência Y^n vai aproximar a solução exata, então $\lim_{\delta t \to 0} \Phi(t_n, \delta t, Y^n, Y^{n+1}) = f(t_n, Y^n)$.

- Se Φ depende de Y^{n+1} se diz que o método é **implícito**, se não, que é **explícito**. Nos métodos **implícitos**, em geral, é necessário resolver **um sistema não linear de equações** para calcular Y^{n+1}.

- O **erro de truncamento** é definido como

$$T_n = \frac{y(t_n + \delta t) - y(t_n)}{\delta t} - \Phi(t_n, \delta t, y(t_n), y(t_n + \delta t)) \ , \tag{33}$$

  onde y é **solução exata** da EDO. Um método é **consistente** se $\lim_{\delta t \to 0} T_n = 0$.

- Um método se diz **consistente de ordem** p se $\| T_n \| = O(\delta t^p)$.

- O **erro** (o verdadeiro!) é

$$e_n = y(t_n) - Y^n \ . \tag{34}$$

- Exercício: Provar que o método θ é consistente de ordem 1 se $\theta \neq 1/2$, e consistente de ordem 2 se $\theta = 1/2$.

  Quando $\theta = 0$, o método é chamado de **Euler explícito**.
  Quando $\theta = 1$, é chamado de **Euler implícito**.
  Quando $\theta = 1/2$ é chamado de **regra trapezoidal**.

- Exercício: Programar o método de Euler implícito para o oscilador de van der Pol.

Teorema: Seja um método de um passo definido por uma função Φ que

1. (**condição de Lipschitz**) para algum $L > 0$, satisfaz

$$\|\Phi(t, \delta t, v, w) - \Phi(t, \delta t, \bar{v}, \bar{w})\| \leq L \left(\|v - \bar{v}\| + \|w - \bar{w}\|\right) \tag{35}$$

   para todo v, $\bar{v}$, w e $\bar{w}$ em B (domínio da $f(t, \cdot)$), e que

2. **é consistente de ordem** p (i.e. $\|T_n\| = O(\delta t^p)$).

Então, **o método é convergente de ordem** p, isto é, $\|e_n\| = O(h^p)$ e, em particular,

$$Y^n \overset{\delta t \to 0,\, n\delta t \to t}{\longrightarrow} y(t) \,. \tag{36}$$

- Se Φ é C^1 no dois últimos argumentos, ela é Lipschitz automaticamente. **Consequência:** Se f é Lipschitz, o método θ é convergente de ordem 1 ($\theta \neq 1/2$) ou 2 ($\theta = 1/2$).
- Atenção: Um método que é **convergente** ainda pode ser **instável para** δt **muito grande**. Isto tem como consequência a "explosão" da sequência Y^n (floating overflow). A escolha do passo de tempo é portanto fundamental, como será discutido daqui a pouco.

- **Demonstração:** Seja $e_0 = y(t_0) - Y^0$, que pode ser zero ou ter algum erro. Calculemos e_1,

$$\|e_1\| = \|y(t_1) - Y^1\| = \| \left[y(t_0) + \delta t\, \Phi(t_0, \delta t, y(t_0), y(t_1)) + \delta t\, T_0\right] - \left[Y^0 + \delta t\, \Phi(t_0, \delta t, Y^0, Y^1)\right] \|$$

$$\leq \|y(t_0) - Y^0\| + \delta t\, \|T_0\| + \delta t\, L\, (\|y(t_0) - Y^0\| + \|y(t_1) - Y^1\|)$$

o que implica que

$$\|e_1\| \leq \frac{1 + \delta t\, L}{1 - \delta t\, L} \|e_0\| + \frac{\delta t}{1 - \delta t\, L}\, \|T_0\| \ .$$

Agora vamos supor δt independente de n por simplicidade. Com o mesmo argumento anterior, provamos que

$$\|e_{n+1}\| \leq \underbrace{\frac{1 + \delta t\, L}{1 - \delta t\, L}}_{A} \|e_n\| + \underbrace{\frac{\delta t}{1 - \delta t\, L}}_{B}\, \|T_n\| \ . \tag{37}$$

Assim, definindo $T = \max_n \|T_n\|$,

$$\begin{aligned}
\|e_1\| &\leq A\|e_0\| + BT \ , \\
\|e_2\| &\leq A\|e_1\| + BT \leq A(A\|e_0\| + BT) + BT = A^2\|e_0\| + (1 + A)BT \\
\|e_3\| &\leq \ldots \quad \leq A^3\|e_0\| + (1 + A + A^2)BT \\
\ldots &\quad \ldots \\
\|e_n\| &\leq A^n\|e_0\| + \frac{A^n - 1}{A - 1}\, BT
\end{aligned}$$

Utilizando agora a desigualdade $(1 + x)/(1 - x) \leq e^{3x}$ para todo $x < 1/2$ (graficar em Octave ou gnuplot para verificar), chegamos a (quando $\delta t < 1/(2L)$)

$$\|e_n\| \leq e^{3L(t_n - t_0)}\|e_0\| + \frac{e^{3L(t_n - t_0)} - 1}{2L}\, T \ .\square$$

- Existem também **métodos multi-passo**, que permitem ordem elevada com **menos avaliações de** f. Contras: **Condições iniciais** mais complexas, restrições de **estabilidade**.
- Exercício: Por exemplo, o método

$$Y^{n+1} = Y^{n-1} + 2\,\delta t\, f(t_n, Y^n)$$

parece bem conveniente, já que é consistente de ordem 2, explícito, e requer apenas uma avaliação de f.

Porém, **prove** que esse método **diverge** a partir de qualquer condição inicial ($\neq 0$) quando aplicado ao problema $y' = -\lambda y$, **para todo** δt.

Dica: Reescrever o método como $Y^{n+1} = Z^n + 2\delta t\, f(t_n, Y^n)$, $Z^{n+1} = Y^n$ e chamar $W^n = (Y^n, Z^n)$. Quando $f(t, y) = -\lambda y$ resulta o método discreto $W^{n+1} = A(\lambda, \delta t)\, W^n$, onde $A(\lambda, \delta t)$ é uma matriz 2×2. Mostrar que essa matriz sempre tem autovalores com módulo maior que 1.

- **Teorema:** Dado um método multi-passo arbitrário, para ser **convergente** ele deve ser **consistente** e **estável**.
  - A **consistência** é verificada pelo desenvolvimento de Taylor.
  - A **estabilidade** é estudada considerando o problema linear escalar $y' = -\lambda y$. Se, sempre que $\lambda > 0$, temos $Y^n \to 0$ para todo δt, dizemos que o método é **incondicionalmente estável**. Se isto só acontece quando $\delta t \leq \delta t_{\max}$, dizemos que a estabilidade é **condicional**.
  - Os **métodos explícitos**, como regra geral, são condicionalmente estáveis, com $\delta t_{\max} \simeq 2/r$, sendo r o módulo do maior autovalor da matriz Jacobiana $J = \partial f/\partial y$ (i.e., $J_{ij} = \partial f_i/\partial y_j$).
- Exercício: Considere o **método de Euler explícito** para o problema $y' = -\lambda y$, com $\lambda > 0$. Mostre que $Y^n \to 0$ sempre que $\delta t < 2/\lambda$. Mostre que o **método de Euler implícito** é incondicionalmente estável.

6.3 Métodos de Runge-Kutta explícitos

- São **métodos de um passo** (isto permite trocar δt ao longo do cálculo), **explícitos** (não precisa iterar a cada passo de tempo), **condicionalmente estáveis** ($\delta t < \delta t_{\max}$) e, dependendo de qual é usado, de **alta ordem de convergência**.

- A expressão geral é (método de s estágios)

$$Y^{n+1} = Y^n + \Delta t \, (b_1 \, k_1 + b_2 \, k_2 + ... + b_s \, k_s)$$

onde

$$\begin{aligned} k_1 &= f(t_n, Y^n) \\ k_2 &= f(t_n + c_2 \, \Delta t, Y^n + \Delta t \, (a_{21} \, k_1)) \\ k_3 &= f(t_n + c_3 \, \Delta t, Y^n + \Delta t \, (a_{31} \, k_1 + a_{32} \, k_2)) \\ \ldots &\quad \ldots \end{aligned}$$

- **Tabela de Butcher**

0					
c_2	a_{21}				
c_3	a_{31}	a_{32}			
...		...			
	b_1	b_2	...	...	b_s

- **Programação:**

$$Y^{n+1} = Y^n + \Delta t \, (b_1 \, k_1 + b_2 \, k_2 + \ldots + b_s \, k_s)$$

onde

$$\begin{aligned} k_1 &= f(t_n, Y^n) \\ k_2 &= f(t_n + c_2 \, \Delta t, Y^n + \Delta t \, (a_{21} \, k_1)) \\ k_3 &= f(t_n + c_3 \, \Delta t, Y^n + \Delta t \, (a_{31} \, k_1 + a_{32} \, k_2)) \\ \ldots &\quad \ldots \end{aligned}$$

```
K(:,1)=f(y(:,n),time(n)); ## ydot=f(y,t), notar inversão de ordem
for m=2:nstage
 tt=time(n)+c(m)*dt;
 yy=y(:,n)+dt*K(:,1:m-1)*a(m,1:m-1)';
 K(:,m)=f(yy,tt);
endfor
ynew=y(:,n)+dt*K(:,1:nstage)*b';
time(n+1)=time(n)+dt;
dtime(n+1)=dt;
y(:,n+1)=ynew;
```

- **Runge-Kutta de ordem 2**

$$
\begin{aligned}
y^{n+1} &= y^n + \Delta t \left(\frac{1}{2} k_1 + \frac{1}{2} k_2 \right) \\
k_1 &= f(t_n, y^n) \\
k_2 &= f(t_n + \Delta t, y^n + \Delta t\, k_1)
\end{aligned}
$$

Tabela de Butcher

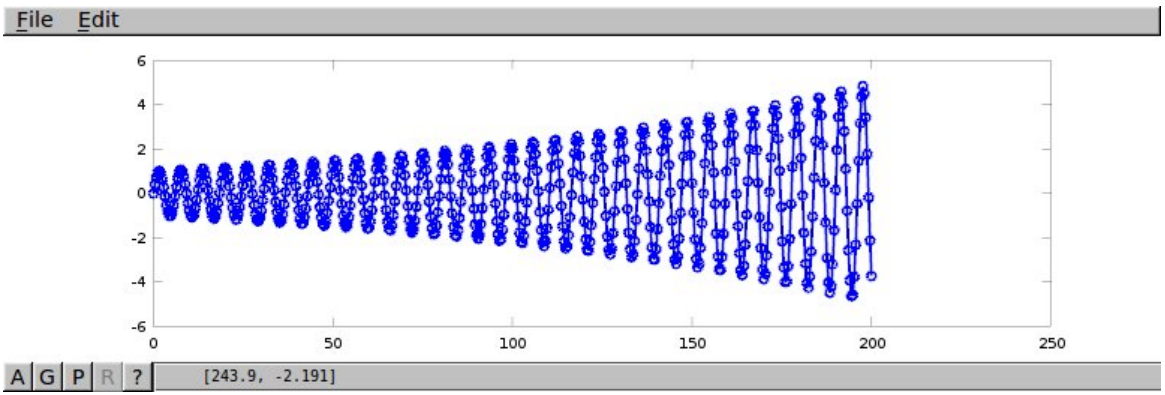
0	0	0
1	1	0
	$\frac{1}{2}$	$\frac{1}{2}$

```
function [y time]=rk2(f,y0,t0,dt,nt)
time(1)=t0; y(:,1)=y0;
nstage=2;a=[0 0;1 0];c=[0; 1];b=[0.5 0.5];
for n=1:nt
 K(:,1)=feval(f,y(:,n),time(n));
 for m=2:nstage
  tt=time(n)+c(m)*dt;
  yy=y(:,n)+dt*K(:,1:m-1)*a(m,1:m-1)';
  K(:,m)=feval(f,yy,tt);
 endfor
 y(:,n+1)=y(:,n)+dt*K(:,1:nstage)*b';
 time(n+1)=time(n)+dt;
endfor
```

```
function f = foscil(y,t)
ome=1.;
f(1)=y(2);
f(2)=-ome*ome*y(1);
end
```

- Resolvemos $x' = v,\ v' = -x$ com $x(0) = 0,\ v(0) = 1$. A solução exata é $x(t) = \sin t$, $v(t) = \cos t$.
- Matriz jacobiana $= \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$, então $J_f = 1$ (autovalores $\pm i$), limite de estabilidade $\Delta t < 2$.

```
> [y time]=rk2("foscil",[0;1],0,0.4,500);
> plot(time,y(1,:),"-o","linewidth",2)
```



- A pouca precisão leva a comportamento errado.

- **Runge-Kutta de ordem 4:** Tabela de Butcher:

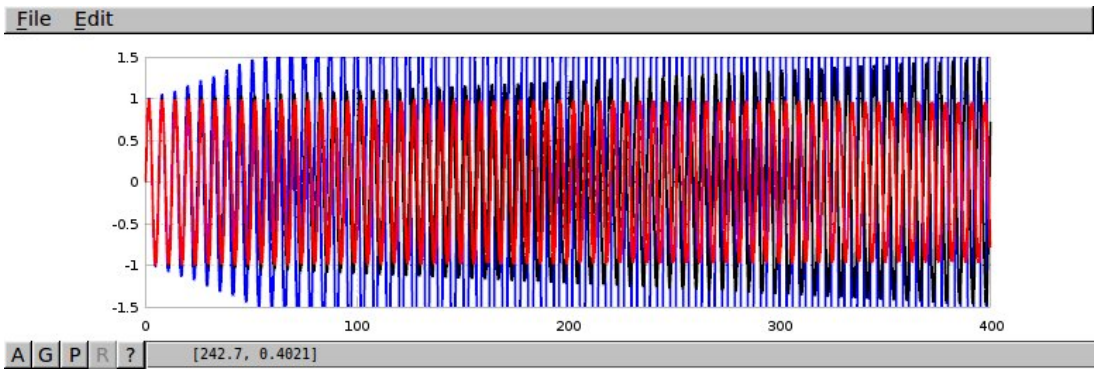
$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ \hline & \frac{1}{6} & \frac{2}{6} & \frac{2}{6} & \frac{1}{6} \end{array}$$

Mesmo código do slide anterior, com

```
nstage=4;
a=[0 0 0 0;1/2 0 0 0;0 1/2 0 0;0 0 1 0];
c=[0; 1/2; 1/2; 1];
b=[1/6 2/6 2/6 1/6];
```

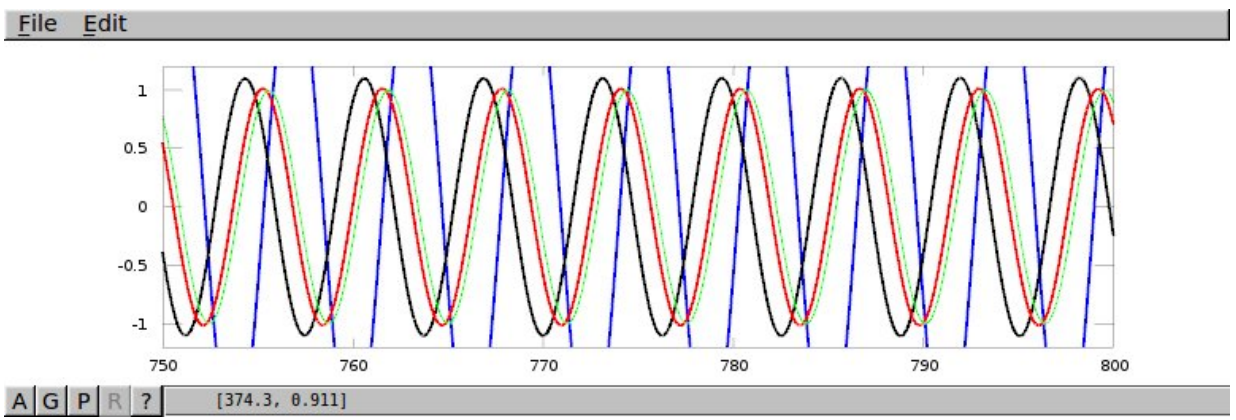
- Comparamos RK2 com $\Delta t = 0.4$ e 0.2, e RK4 com $\Delta t = 0.4$.

```
>[y1 time1]=rk2("foscil",[0;1],0,0.4,1000);
>[y2 time2]=rk2("foscil",[0;1],0,0.2,2000);
>[y4 time4]=rk4("foscil",[0;1],0,0.4,1000);
> plot(time1,y1(1,:),"-b","linewidth",2,...
time2,y2(1,:),"-k","linewidth",2,...
time4,y4(1,:),"-r","linewidth",2)
> axis([0 400 -1.5 1.5])
```



- Qual é o Δt tal que a $t = 800$ a amplitude continua sendo $\simeq 1$ com RK2?

```
>[y1 time1]=rk2("foscil",[0;1],0,0.2,4000);
>[y2 time2]=rk2("foscil",[0;1],0,0.1,8000);
>[y4 time4]=rk2("foscil",[0;1],0,0.05,16000);
>plot(time1,y1(1,:),"-b","linewidth",2,time2,y2(1,:),"-k",...
"linewidth",2,time4,y4(1,:),"-r","linewidth",2,...
time1,sin(time1),"-g","linewidth",1)
> axis([750 800 -1.2 1.2])
```



- **Miniprojeto:** A equação unidimensional de um foguete é

$$m\frac{dv}{dt} = -c\frac{dm}{dt} - mg - \frac{1}{2}\rho C_D A|v|v,$$

onde $m(t)$ é a massa total, c a velocidade de expulsão dos gases, g a gravidade, ρ a densidade do ar na altura z do foguete, C_D o coeficiente de arrasto e A a área transversal.

Consideramos $\rho = \rho_0 \exp(-z/H)$, com $\rho_0 = 1.2$ kg/m^3 e $H = 8000$ metros, também $c = 3000$ m/s, $A = 1$ m^2, $C_D = 0.2$. A massa inicial do foguete é de 1100 kg, dos quais 1000 kg são combustível que é queimado a taxa dm/dt constante em um tempo T (i.e., dm/dt =1000 kg/T).

Calcular utilizando o método RK4 (e disponibilizar o código):

  - A altura máxima que o foguete alcança, para vários valores de T. Estime o erro numérico dos valores obtidos realizando um estudo de sensibilidade ao δt.
  - Existe um valor ótimo para T?
  - Plote a altura como função do tempo para vários T.
  - São importantes os efeitos do arrasto? Qual seria a altura máxima se $C_D = 0$?
  - É importante considerar a variação da densidade do ar com a altura? Qual seria a altura máxima se $H \to +\infty$?
  - Qual o custo (em massa de combustível) de um kg adicional de carga? Em outras palavras, quanto mais combustível deveriamos adicionar para que o foguete chegue até a mesma altura sendo que a carga é de 101 kg e não de 100 kg?

Ajuste automático de passo

Ideia geral: Em cada passo de tempo,

1. Calcular Y^{n+1} com **dois** métodos de diferente ordem.
   Exemplo: RK2$\rightarrow Y^{n+1}$, RK4$\rightarrow Z^{n+1}$.
2. Estimar o erro como a diferença desses resultados.

$$e = \left\| Y^{n+1} - Z^{n+1} \right\|$$

3. Se (erro $>$ tolerância) reduzir δt e voltar a 1.
4. Predizer um δt adequado para próximos passos. Vamos supor que o esquema de menor ordem é de ordem p, **o "passo ideal" δt_* daria erro igual à tolerância**.

$$e \simeq C\,\delta t^{p+1}, \quad \epsilon \simeq C\,\delta t_*^{p+1} \quad \Rightarrow \quad \delta t_* \simeq \delta t \left(\frac{\epsilon}{e}\right)^{\frac{1}{p+1}}$$

$$\delta t \longleftarrow \max\left(0.5\delta t, \min\left(2\delta t, 0.9\delta t_*\right)\right)$$

6.4 Métodos BDF para problemas “rígidos”

- Os problemas “rígidos”, ou *stiff* em inglês, são problemas nos quais se deseja calcular uma solução suave (ou “lenta”) mas a EDO governante faz que qualquer perturbação dessa solução exiba transientes de variação bem rápida.
- Um bom exemplo, tomado de LeVeque, é

$$u'(t) = \lambda\,(u(t) - \cos t) - \sin t\ , \qquad u(0) = 1\ , \tag{38}$$

cuja solução exata é $u(t) = \cos t$ para todo λ. Se $u(0) = \eta \neq 1$ e $\lambda < 0$, a solução exata é

$$u(t) = e^{\lambda t}(\eta - 1) + \cos t\ ,$$

que se aproxima exponencialmente rápido do atrator $\cos t$.

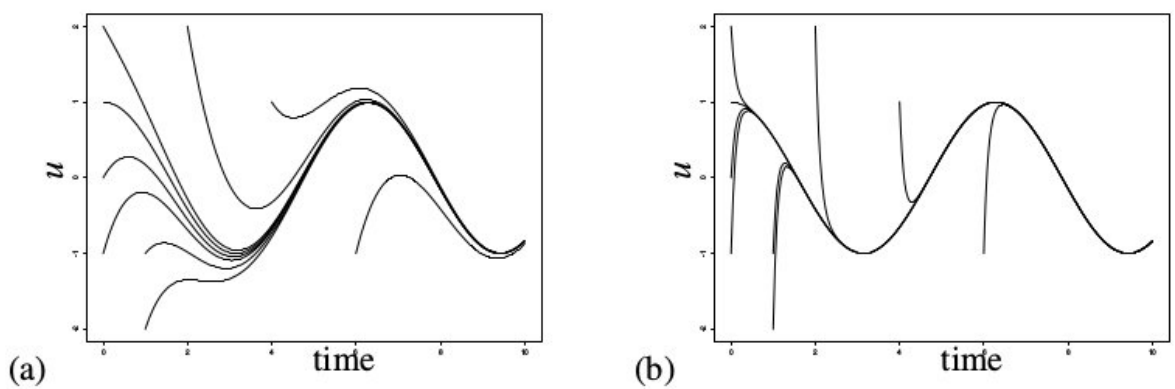


Figure 8.1. *Solution curves for the ODE* (8.1) *for various initial values.* (a) *With* $\lambda = -1$. (b) *With* $\lambda = -10$ *and the same set of initial values.*

- Sendo que $f(t,u) = \lambda(u - \cos t) - \sin t$, a Jacobiana é $\partial_u f = \lambda$. Métodos explícitos terão uma restrição no passo de $\delta t < 2/|\lambda|$. Isto é: A solução que se deseja calcular pode muito bem ser **interpolada** com $\delta t \simeq 2\pi/N$, com $N = 10, 100, \ldots$ dependendo da precisão desejada, mas a **estabilidade numérica** nos obriga a calcular com δt muito menor se $\lambda \ll 0$.

- **Exercício:** Programar os métodos de Euler implícito, Euler explícito e trapezoidal para o problema acima. Calcular com $\lambda = -10$ e diversos passos de tempo ($\delta t = 2\pi/20, 2\pi/40, 2\pi/80, \ldots$). Como se comportam os métodos?

- Para esses problemas se recomenda usar métodos **implícitos**, tais como os métodos BDF abaixo (caso δt constante):

$$\frac{1}{\delta t}\left(Y^n - Y^{n-1}\right) - f(t_n, Y^n) = 0 \qquad \text{Euler implícito} \tag{39}$$

$$\frac{1}{\delta t}\left(1.5Y^n - 2Y^{n-1} + 0.5Y^{n-2}\right) - f(t_n, Y^n) = 0 \qquad \text{BDF de dois passos} \tag{40}$$

$$\frac{1}{\delta t}\left(\frac{11}{6}Y^n - 3Y^{n-1} + \frac{3}{2}Y^{n-2} - \frac{1}{3}Y^{n-3}\right) - f(t_n, Y^n) = 0 \qquad \text{BDF de três passos} \tag{41}$$