

Mecânica de Fluidos Computacional I

Prof. Gustavo Carlos Buscaglia

Laboratório de Matemática Aplicada e Computação Científica (LMACC)
Departamento de Matemática Aplicada e Estatística
Instituto de Ciências Matemáticas e de Computação (ICMC)
USP – São Carlos

2017

Mecânica dos Fluidos Computacional

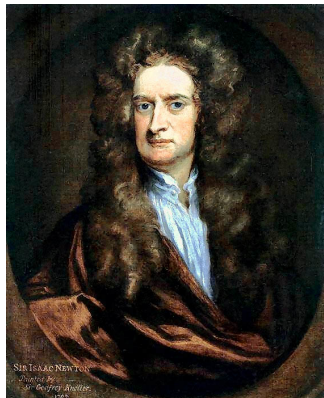
- A Mecânica dos Fluidos é a ciência que estuda o comportamento dos fluidos.
- Este estudo é feito de três formas:
 - **Experimental:** Fenômenos físicos estudados em ambientes controlados.
 - **Teórico:** Obtenção de soluções simplificadas às equações de modelo.
 - **Numérico:** Utilizar o auxílio do computador.
- Neste curso estudaremos a utilização do computador na resolução de vários problemas de mecânica dos fluidos.

Breve histórico

- Desde os primórdios de nossa civilização, o ser humano se interessa pelo movimento dos fluidos (ventos, rios, clima, etc.)
- **Arquimedes** (287-212 a.C.): planejamento de aquedutos, canais, casas de banho, etc.
- **Leonardo da Vinci** (1452-1519): observou e reportou vários fenômenos, reconhecendo sua forma e estrutura, reportando-os na forma de desenhos e esquemas.



Breve histórico

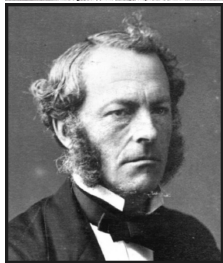


- **Isaac Newton** (1643-1727): Muitas contribuições à mecânica dos fluidos
- Sua segunda lei: $F = m \cdot a$
- Viscosidade: A tensão é proporcional à taxa de deformação.

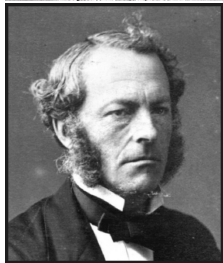


- **Daniel Bernoulli** (1700-1782): Equação de Bernoulli.
- **Leonhard Euler** (1707-1783): Equações de Euler para escoamento invíscido, conservação de quantidade de movimento, conservação de massa, potencial de velocidade.

Breve histórico



- **Claude Louis Marie Henry Navier** (1785-1836)
- **Gabriel Stokes** (1819-1903)
- Introduziram transporte viscoso às equações de Euler, resultando nas equações de Navier-Stokes para escoamentos incompressíveis



- **Claude Louis Marie Henry Navier** (1785-1836)
- **Gabriel Stokes** (1819-1903)
- Introduziram transporte viscoso às equações de Euler, resultando nas equações de Navier-Stokes para escoamentos incompressíveis

$$\begin{aligned}\frac{\partial(\rho u)}{\partial t} + \nabla \cdot (\rho u \otimes u) = \\ - \nabla p + \nabla \cdot [\mu(\nabla u + \nabla u^T)] + \rho g \\ \nabla \cdot u = 0\end{aligned}$$

Breve histórico

- **Lewis Fry Richardson** (1881-1953): desenvolveu o primeiro método numérico para previsão do tempo (escoamento atmosférico)
- Sua tentativa de calcular a previsão do tempo para um período de 8 horas lhe tomou 6 semanas de cálculos, e foi um fracasso.
- Forecast-factory



Mecânica dos fluidos computacional

- Soluções numéricas das equações de Navier-Stokes demandam muitos cálculos.
(em 1953, **M. Kawaguti** calculou a solução de um escoamento em torno de um cilindro, levou 18 meses trabalhando 20 horas por semana).
- A evolução da computação beneficia diretamente a área.
- Hoje, com o advento dos supercomputadores, é possível resolver escoamentos complexos com precisão em tempo factível.

Objetivo da disciplina

- Mostrar como, utilizando a modelagem matemática e o cálculo numérico, é possível resolver problemas de mecânica de fluidos cuja resolução analítica é impossível.

- Cada capítulo do curso será resolvido um problema.
- A modelagem física e matemática será desenvolvida pelo professor.
- O professor sugerirá um ou vários tratamentos numéricos, e os explicará detalhadamente.
- Os estudantes, em grupos de 2, implementarão um programa para cada problema.
- Os programas serão testados e comparados.
- Um estudante escolhido aleatoriamente de cada grupo realizará uma apresentação de quinze minutos. Os slides serão considerados como relatório do grupo.
- A nota final será calculada a partir das notas obtidas em cada trabalho, sendo que todos os capítulos devem ser aprovados.

- 1 Cálculo de forças e torques em hidrostática. Dinâmica de corpos rígidos flutuantes e seu cálculo numérico.
- 2 Aproximação numérica de interfaces com tensão superficial. Minimização da energia e aproximação variacional.
- 3 Modelagem numérica de redes hidráulicas. Origem e tratamento das não-linearidades.
- 4 Resolução numérica das equações de Navier-Stokes incompressíveis. Convergência em malha a uma solução manufaturada.

Técnicas numéricas envolvidas em cada capítulo

- 1 Parametrização de formas. Interpolação. Integração numérica. EDOs numéricas.
- 2 EDOs numéricas. Minimização de funções.
- 3 Grafos e sua representação computacional. Resolução de sistemas de equações não lineares.
- 4 Diferenças finitas. Volumes finitos. Aproximação numérica de EDPs. Cálculo experimental de ordem de convergência.

Tecnologias relacionadas com cada capítulo

- 1 Engenharia civil. Forças em represas, em prédios, etc. Engenharia naval. Estabilidade de estruturas flutuantes, navios, etc.
- 2 Indústria química. Pintura por imersão, por deposição de sprays. Impressão de jato de tinta. Indústria do petróleo. Separação de misturas. Misturas bifásicas.
- 3 Engenharia hidráulica. Distribuição urbana de água.
- 4 Microfluídica. Lab on a chip. Incorporando turbulência numérica (que não veremos): Meteorologia, Oceanografia, Indústria automotiva, etc.

Duração estimada de cada capítulo

- 1 Cálculo de forças e torques em hidrostática. Dinâmica de corpos rígidos flutuantes e seu cálculo numérico. $\Rightarrow$ **3 semanas**
- 2 Aproximação numérica de interfaces com tensão superficial. Minimização da energia e aproximação variacional. $\Rightarrow$ **4 semanas**
- 3 Modelagem numérica de redes hidráulicas. Origem e tratamento das não-linearidades. $\Rightarrow$ **3 semanas**
- 4 Resolução numéricas das equações de Navier-Stokes incompressíveis. Convergência em malha a uma solução manufaturada. $\Rightarrow$ **4 semanas**

- Essa disciplina é direcionada a alunos do BMAcc, com menos formação em física e fluidos que alunos de engenharia ou física. Em cada capítulo será feita uma revisão rigorosa mas rápida dos conceitos de mecânica que sejam necessários.
- Se espera que os estudantes tenham familiaridade com o cálculo numérico. Os conceitos usados serão definidos mas não fundamentados teoricamente.
- É necessário saber programar. As aulas conterão exemplos de implementação em Octave.
- Seria conveniente ter acesso a uma notebook por grupo.
- Cada capítulo envolverá 2 ou 3 aulas de trabalho em sala, consultando ao professor e/ou ao PAE.
- Haverá sessões de monitoria em laboratório. Definir dia e horário.

Simulação numérica de corpos em flutuação

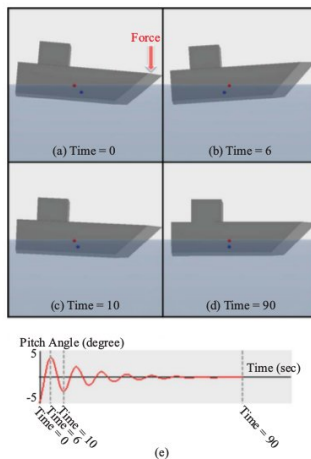


Fig. 10. (a)-(d) Snapshots of the pitch motion of a ship, (e) the magnitude of the pitch motion at each time step.

1

¹Ueng, J. Marine Sci. and Technol., 2013

Exemplos

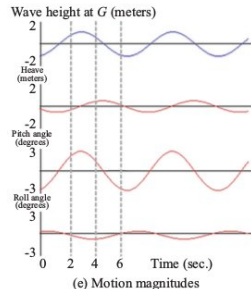
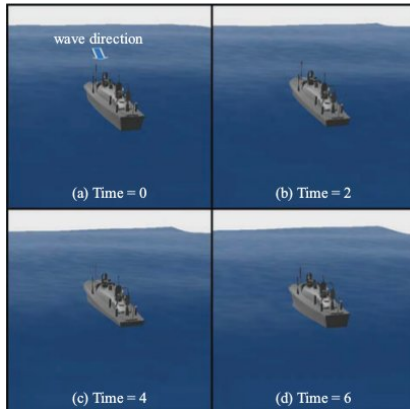
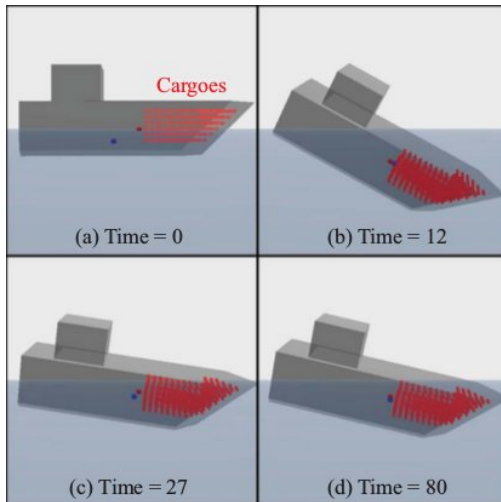


Fig. 11. (a)-(d) Snapshots of the wave-induced motions, (e) the wave height at G and the magnitudes motion at each time step.

Exemplos



■ Estruturas



*Fig. 4 Nordhordland Floating Bridge,
Norway*



*Fig. 5 West India Quay Footbridge,
United Kingdom*

2

²Watanabe et al, CORE Rep. 2004-02

Exemplos



Fig. 9. Shirashima Floating Oil Storage Base, Japan (Photo courtesy of Shirashima Oil Storage Co Ltd)



Fig. 10 Kamigoto Floating Oil Storage Base, Nagasaki Prefecture, Japan



*Fig. 13 Concept Design of a Clean Energy Plant
by Floating Structure Association of Japan*



Fig. 15 Mega-Float in Tokyo Bay, Japan (Photo courtesy of SRCJ)



Fig. 16 Proposed Floating Runway at Tokyo International Airport (Haneda)

Exemplos

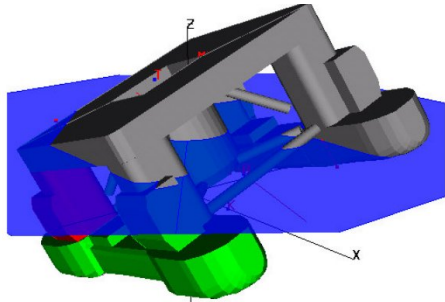


Figure 26- Sstab evaluation of flooding using measured drafts.

3

³Coelho et al, 8th Conf. Stab. Ships and Ocean Vesicles, 2003

Como flutuam os corpos?

■ Equilíbrio dinâmico entre:

- Peso próprio
- Forças exercidas pelo líquido
- Forças exercidas pela atmosfera (vento)
- Forças externas (âncoras,...)

$$M \frac{d^2 \vec{c}}{dt^2} = \sum \vec{F}, \quad \frac{d\vec{L}}{dt} = \sum \vec{T}$$

onde $\vec{c}$ é a posição do centro de massa, $\vec{F}$ força, $\vec{L}$ momento angular e $\vec{T}$ torque (ambos respeito de $\vec{c}$).

- O corpo *deforma* pelas forças aplicadas.
- Os fluidos *respondem* à presença do corpo.

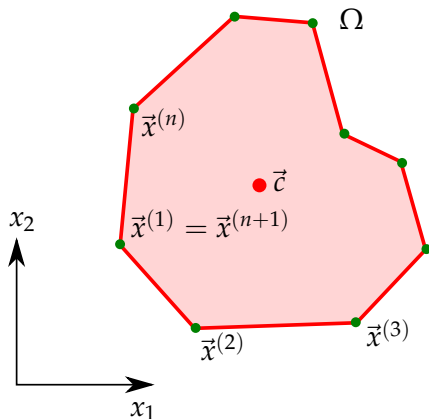
Descrição geométrica do corpo

- Existem diversas maneiras: Primitivas, interpolatórias com/sem malha, etc.
- Consideraremos a seguinte:
 - A superfície $\mathcal{S}$ do corpo é a união disjunta de um conjunto de patches, cada um deles sendo a imagem de um símplice M por uma transformação $\vec{\phi} : M \rightarrow \mathbb{R}^d$.

$$\mathcal{S} = \cup_K \vec{\phi}_K(M)$$

- A geometria de cada patch é definida por um conjunto de nós.

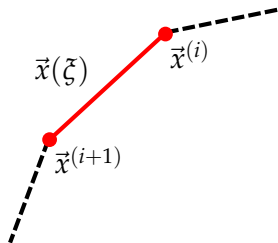
Representação linear por partes em 2D



Lista de pontos:

$x_1^{(1)}$	$x_2^{(1)}$
$x_1^{(2)}$	$x_2^{(2)}$
$\vdots$	$\vdots$
$x_1^{(n)}$	$x_2^{(n)}$
$x_1^{(n+1)}$	$x_2^{(n+1)}$

Representação linear por partes em 2D

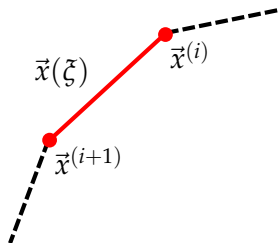


$$\vec{x}(\xi) = \vec{\varphi}_{K=i}(\xi) = \frac{1-\xi}{2} \vec{x}^{(i)} + \frac{1+\xi}{2} \vec{x}^{(i+1)},$$

$$\xi \in M = [-1, 1]$$

A imagem de $\vec{\varphi}_i$ é o segmento reto de $\vec{x}^{(i)}$ a $\vec{x}^{(i+1)}$.

Representação linear por partes em 2D



- Cálculo do Jacobiano (comprimento de arco):

$$ds = |d\vec{x}| = \left\| \frac{d\vec{x}}{d\xi} \right\| d\xi = \left\| \frac{\vec{x}^{(i+1)} - \vec{x}^{(i)}}{2} \right\| d\xi$$

- Cálculo da normal (anti-horária):

$$\vec{d} \stackrel{\text{def}}{=} \vec{x}^{(i+1)} - \vec{x}^{(i)}, \quad \vec{n} = \frac{(d_2, -d_1)}{\|\vec{d}\|}$$

Ambos constantes em cada patch.

Seja f uma função definida em $\mathbb{R}^d$, então

$$\begin{aligned}\int_{\mathcal{S}} f(\vec{x}) \, d\mathcal{S} &= \sum_K \int_{\vec{\varphi}_K(M)} f(\vec{x}) \, d\mathcal{S} \\ &= \sum_K \int_M f(\underbrace{\vec{\varphi}_K(\vec{\xi})}_{\vec{x}(\vec{\xi})}) \underbrace{\frac{d\mathcal{S}}{dM}}_{\text{Jacobiano de } \vec{\varphi}_K} \, dM.\end{aligned}$$

- A integral de curva/superfície transformou-se numa soma de integrais sobre um único símplice M .
- Em 1D, certamente escolhemos $M = [-1, 1]$.

- Dados três pontos arbitrários em 3D, com coordenadas $\vec{x}^{(1)}$, $\vec{x}^{(2)}$ e $\vec{x}^{(3)}$, determinar uma transformação $\vec{\varphi} : M \rightarrow \mathbb{R}^3$ que transforme o triângulo unitário M (com vértices $(0,0)$, $(1,0)$ e $(0,1)$) no triângulo plano definido pelos três pontos.
- Quanto vale o Jacobiano dessa transformação?

Integração em $\mathcal{S}$, caso 1D

Nos cálculos, vão aparecer **muitas** integrais da forma

$$\int_{\mathcal{S}} f(\vec{x}) \, ds = \sum_K \int_{-1}^1 \underbrace{f(\vec{\phi}_K(\xi))}_{g(\xi)} \frac{ds}{d\xi} \, d\xi .$$

Quadratura de Gauss-Legendre

Existe uma quadratura

$$\int_{-1}^1 g(\xi) d\xi \simeq \sum_{k=1}^N A_k g(\xi_k)$$

que integra exatamente polinômios de grau $2N - 1$. Se esperamos que a função a integrar seja aproximável por um polinômio desse grau, escolhemos:

$N = 2$	$N = 3$
$\xi_1 = -1/\sqrt{3} \quad A_1 = 1$	$\xi_1 = -\sqrt{3/5} \quad A_1 = 5/9$
$\xi_2 = 1/\sqrt{3} \quad A_2 = 1$	$\xi_2 = 0 \quad A_2 = 8/9$
	$\xi_3 = \sqrt{3/5} \quad A_3 = 5/9$

Integração em $\mathcal{S}$, numérica

$$\begin{aligned}\int_{\mathcal{S}} f(\vec{x}) \, ds &= \sum_K \int_{-1}^1 \underbrace{f(\vec{\varphi}_K(\xi))}_{g(\xi)} \frac{ds}{d\xi} \, d\xi . \\ &\simeq \sum_K \sum_{k=1}^N A_k \underbrace{f(\vec{\varphi}_K(\xi_k))}_{f(\vec{x}_k)} \frac{ds}{d\xi}\end{aligned}$$

A seguir, alguns exemplos de cálculos de integrais sobre superfícies “numéricas”.

Cálculo do volume (2D) encerrado em $\mathcal{S}$

$$\begin{aligned} V &= \int_{\Omega} 1 \, dx \, dy \\ &= \int_{\Omega} \frac{1}{2} \operatorname{div} \vec{x} \, dx \, dy \\ &= \oint_{\mathcal{S}} \frac{1}{2} \vec{x} \cdot \vec{n} \, ds \\ &= \sum_K \sum_{k=1}^N A_k \frac{1}{2} \vec{x}_k \cdot \vec{n}_k \frac{ds}{d\zeta}(\zeta_k) \end{aligned}$$

Indeed, if the geometrical interpolation is $\mathbb{P}_1$, then for each K we have that $\vec{x}(\zeta)$ is $\mathbb{P}_1$ while $\vec{n}$ and $ds/d\zeta$ are constant in each segment (in fact, $ds/d\zeta = \ell_K/2$).

The integrand is thus $\mathbb{P}_1$, so that $N = 1$ suffices to compute the volume to roundoff error.

Cálculo do centro de massa de um polígono homogêneo

$$\begin{aligned}
 \vec{c} &= \frac{1}{3V} \begin{bmatrix} \oint_{\partial\Omega} (\vec{x} \cdot \vec{n}) x_1 ds \\ \oint_{\partial\Omega} (\vec{x} \cdot \vec{n}) x_2 ds \end{bmatrix} \\
 &= \frac{1}{3V} \begin{bmatrix} \sum_{i=1}^n \int_{-1}^1 (\vec{x}(\xi) \cdot \vec{n}^{(i)}) x_1(\xi) \frac{\ell_i}{2} d\xi \\ \sum_{i=1}^n \int_{-1}^1 (\vec{x}(\xi) \cdot \vec{n}^{(i)}) x_2(\xi) \frac{\ell_i}{2} d\xi \end{bmatrix} \\
 &= \frac{1}{3V} \begin{bmatrix} \sum_{i=1}^n \sum_{k=1}^N A_k \vec{x}_k \cdot \vec{n}^{(i)} x_{1k} \frac{\ell_i}{2} \\ \sum_{i=1}^n \sum_{k=1}^N A_k \vec{x}_k \cdot \vec{n}^{(i)} x_{2k} \frac{\ell_i}{2} \end{bmatrix}
 \end{aligned}$$

Cálculo do empuxo hidrostático

- A força de pressão por unidade de área que um meio fluido faz sobre um corpo que ocupa um domínio Ω é dada por

$$\vec{f}_p = -p, \vec{n}.$$

A força total é obtida por integração em S ,

$$\vec{F}_p = \int_S -p \vec{n} dS.$$

- Se o fluido está imóvel (estático), a única força é a de pressão. Se não, devem ser consideradas também as forças *viscosas*.

- Considerando a superfície livre do fluido num instante t como sendo

$$x_2 = Z(t, x_1) , \quad (x_3 = Z(t, x_1, x_2) \text{ em 3D}),$$

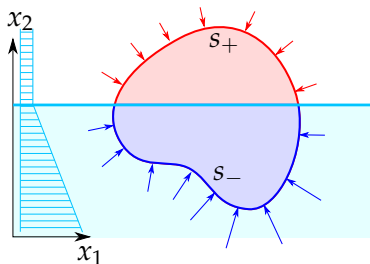
a aproximação de pressão hidrostática é

$$p(t, x_1, x_2) = p_{atm}(t, x_1) + \rho_L g \max(0, Z(t, x_1) - x_2)$$

- Em geral se toma p_{atm} independente de $\vec{x}$.
- Caso estático: $Z(t, x_1) = \text{constante}$.
- Enchimento/esvaziamento quaseestático: $Z(t, x_1) = h(t)$.
- Onda de celeridade v :

$$Z(t, x_1) = h(x - vt) , \quad \text{e.g.;} \quad Z(t, x_1) = h_0 + a \sin(k(x_1 - vt)) .$$

Caso estático



$$\vec{F}_A = \int_{S_+} -p_{atm} \vec{n} \, ds = -p_{atm} \int_{S_+} \vec{n} \, ds$$

$$\begin{aligned} \vec{F}_L &= \int_{S_-} -p(\vec{x}) \vec{n} \, ds \\ &= \int_{S_-} -[p_{atm} + \rho_L g(z - x_2)] \vec{n} \, ds \\ &= -p_{atm} \int_{S_-} \vec{n} \, ds - \rho_L g \int_{S_-} (z - x_2) \vec{n} \, ds \end{aligned}$$

$$\begin{aligned} \vec{F} &= \vec{F}_A + \vec{F}_L = -p_{atm} \int_{S_+} \vec{n} \, ds - p_{atm} \int_{S_-} \vec{n} \, ds - \rho_L g \int_{S_-} (z - x_2) \vec{n} \, ds \\ &= -p_{atm} \oint_{S_+ \cup S_-} \vec{n} \, ds - \rho_L g \int_{S_-} (z - x_2) \vec{n} \, ds \end{aligned}$$

0

- Como p_{atm} não tem influência no balanço de forças, pode-se definir $p_{atm} = 0$.
- *Princípio de Arquimedes*: No caso estático o empuxo é igual ao peso do líquido deslocado, e de sentido contrário.
- Se Z ou p_{atm} dependem de x_1 , o empuxo tem componente horizontal.

- Programe uma função que, a partir da matriz de coordenadas coor de um polígono e da função $Z(t, x_1)$, calcule o empuxo do líquido sobre o corpo.
- Grafique as componentes horizontal e vertical do empuxo como função do tempo.
- Considere $Z(t, x_1) = t$, e outra função que represente uma onda que passa pela posição do corpo.

A partir dos conceitos desenvolvidos, uma formulação razoável seria:

- Seja x_j^i a componente j do vetor de coordenadas do nó i , com $i = 1, \dots, N$. Isto poderia ser uma matriz `coor(1:n, 1:2)` em que cada linha é o vetor (linha) posição de um nó. Por exemplo,

```
n=4; coor=[0 0;1 0;1 2;0 2];  
coor=[coor;[coor(1,:)]];  
constroi o retângulo  $(0,1) \times (0,2)$ 
```

- A regra de quadratura poderia ser

```
N=3; xi=[-sqrt(3/5),0,sqrt(3/5)];A=[5/9,8/9,5/9];
```

- Pesos interpolatórios $(1 - \xi)/2$ e $(1 + \xi)/2$:

```
pint=[(1-xi)/2;(1+xi)/2];
```

- Imagens dos pontos de quadratura no segmento ("patch") i

```
for k=1:N  
    x(1,k)=pint(1,k)*coor(i,1)+pint(2,k)*coor(i+1,1);  
    x(2,k)=pint(1,k)*coor(i,2)+pint(2,k)*coor(i+1,2);  
endfor
```

- Os Jacobianos e vetores normais:

```
d=[coor(2:n+1,:)-coor(1:n,:)];ll=norm(d,"rows");jac=ll/2;  
  
normal=[d(:,2)./ll,-d(:,1)./ll]
```

■ Integral da função $f = 1$ (perímetro):

```
perim=0;
for i=1:n
    perim=perim+A*ones(N,1)*jac(i);
endfor
```

■ Integral de função arbitrária:

```
f=@(x) x(1)*x(2);
res=0;
for i=1:n
    for k=1:N
        x=pint(1,k)*coor(i,:)'+pint(2,k)*coor(i+1,:);
        res=res+A(k)*f(x)*jac(i);
    endfor
endfor
```

■ Integral da normal (= 0):

```
res=[0;0];  
for i=1:n  
    for k=1:N  
        res=res+A(k)*normal(i,:)'*jac(i);  
    endfor  
endfor
```

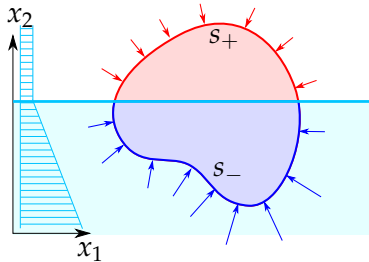
■ Integral de $\frac{1}{2}\vec{x} \cdot \vec{n}$ (=volume):

```
res=0;  
for i=1:n  
    for k=1:N  
        x=pint(1,k)*coor(i,:)'+pint(2,k)*coor(i+1,:)';  
        res=res+A(k)*1/2*normal(i,1:2)*x*jac(i);  
    endfor  
endfor
```

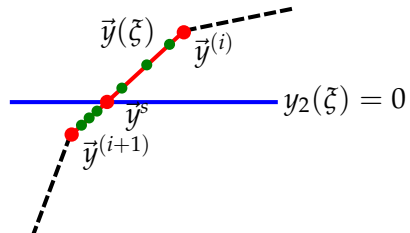
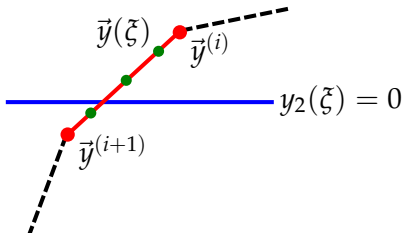
■ Empuxo:

```
Z=@(t,x1) 1.5;  
p=@(t,x) max(0,Z(t,x(1))-x(2));  
t=0;  
res=[0;0];  
for i=1:n  
    for k=1:N  
        x=pint(1,k)*coor(i,:)'+pint(2,k)*coor(i+1,:);  
        res=res+A(k)*normal(i,1:2)'*(-p(t,x))*jac(i);  
    endfor  
endfor
```

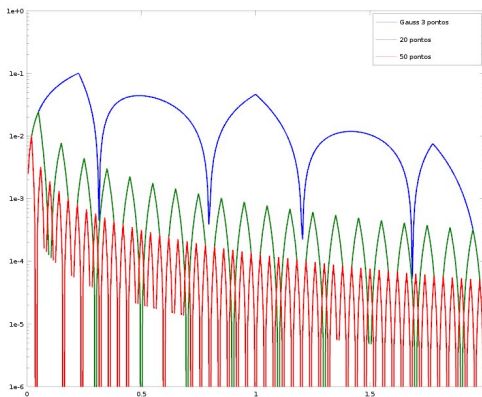
Erros de integração



- A função $p(t, \vec{x})$ **não é um polinômio** ao longo das arestas cortadas pela superfície livre. Isto, no caso de corpos com arestas muito compridas, requer uma integração especial.



- Consideremos o quadrilátero definido pelos pontos $(1,0)$, $(2,0)$, $(3,2)$ e $(0,2)$. Seu empuxo (Arquimedes) é, em função da altura da água, $E(z) = \rho g z (1 + z/2)$.
- Na figura se compara o **erro relativo** de usar quadratura de Gauss de 3 pontos, com o de quadraturas de 20 e 50 pontos.



Cálculo do torque de pressão

- O **torque** é a quantidade que governa as rotações, assim como a **força** governa as translações.
- É uma quantidade **vetorial** (pseudo), mas em 2D é escalar.

$$\vec{T}_P = \int_S (\vec{x} - \vec{c}) \times (-p(\vec{x}) \vec{n}) dS$$

- O torque é medido **respeito de um centro de rotação**, nesse caso $\vec{c}$.
- Quando o centro de rotação coincide com o centro de massa, a dinâmica rotacional é dada por $I \omega'(t) = T$.

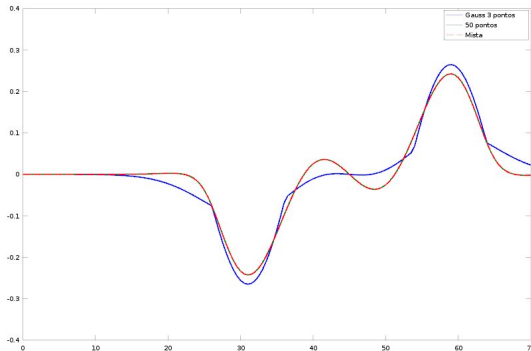
■ Torque:

```
Z=@(t,x1) 1.+exp(-(x1-0.1*t+3)^2/0.16);  
p=@(t,x) max(0,Z(t,x(1))-x(2));  
t=0; res=0;  
for i=1:n  
    for k=1:N  
        x=pint(1,k)*coor(i,:)'+pint(2,k)*coor(i+1,:);  
        ff=(x(1)-cg(1))*normal(i,2)-(x(2)-cg(2))*normal(i,1);  
        res=res+A(k)*ff*(-p(t,x))*jac(i);  
    endfor  
endfor
```

Erro de integração

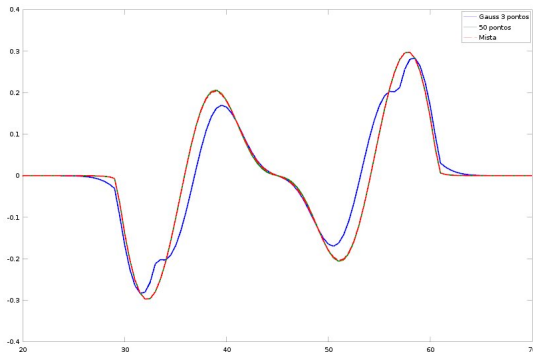
- Consideremos o quadrilátero definido pelos pontos $(1,0)$, $(2,0)$, $(3,2)$ e $(0,2)$. Calculamos o torque respeito de $\vec{c}$ quando passa a onda

$$Z(t, x_1) = 1 + \exp \left(- (x_1 + 3 - 0.1 t)^2 \right) .$$



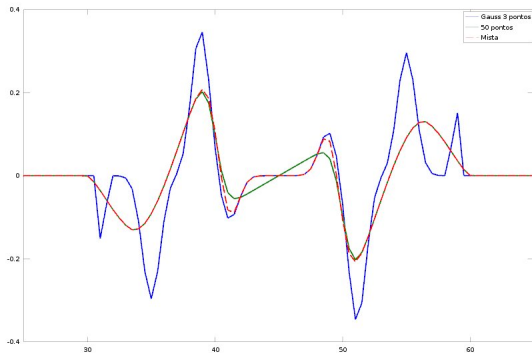
Onda mais ingreme,

$$Z(t, x_1) = 1 + \exp \left(- \frac{(x_1 + 3 - 0.1 t)^2}{0.4^2} \right) .$$



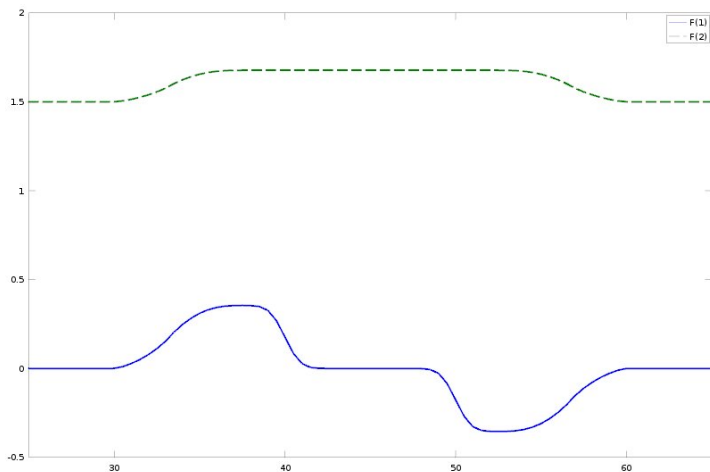
Onda ainda mais ingreme,

$$Z(t, x_1) = 1 + \exp \left(- \frac{(x_1 + 3 - 0.1 t)^2}{0.1^2} \right) .$$



O método misto também dá resultado errado, porque a integração de 3 pontos não consegue capturar a onda.

Complementemos vendo o empuxo para essa última onda.



Definição (Corpo rígido)

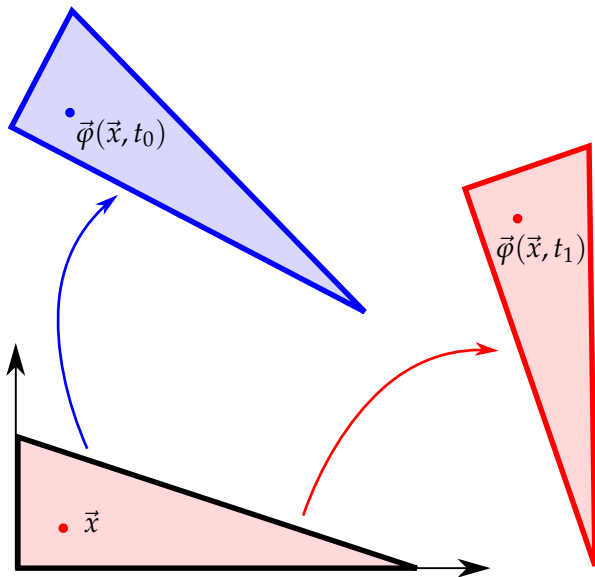
Um corpo Ω é dito rígido se, quando submetido a um movimento, as distâncias entre seus pontos não variam.

Para conseguir descrever o movimento de corpos rígidos, usaremos uma transformação

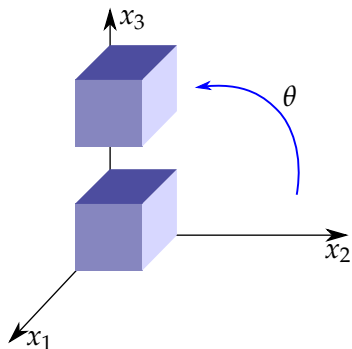
$$\vec{\varphi} : \mathbb{R}^2 \times \mathbb{R} \rightarrow \mathbb{R}^2$$
$$\vec{x} \mapsto \vec{\varphi}(\vec{x}, t)$$

onde $\vec{x}$ é tomado em uma configuração de referência de Ω e $\vec{\varphi}(\vec{x}, t)$ é a posição ocupada no tempo t pela partícula do corpo que na configuração de referência está no ponto $\vec{x}$.

Corpo rígido



Exemplo



Rotação em torno do eixo x_1 de um cubo deslocado em $x_3 = 2$, por um ângulo $\theta(t)$. A configuração de referência do cubo permanece na origem.

$$\vec{\phi}(\vec{x}, t) =$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta(t) & -\sin \theta(t) \\ 0 & \sin \theta(t) & \cos \theta(t) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 + 2 \end{bmatrix}$$

Teorema (Representação do movimento)

Uma transformação $\vec{x} \mapsto \vec{\psi}(\vec{x})$ satisfaz

$$\|\vec{x} - \vec{y}\| = \|\vec{\psi}(\vec{x}) - \vec{\psi}(\vec{y})\|$$

se e somente se existe uma matriz ortogonal $Q \in \mathbb{R}^{d \times d}$ e um vetor $\vec{b} \in \mathbb{R}^d$ tais que

$$\vec{\psi}(\vec{x}) = Q \cdot \vec{x} + \vec{b}$$

Teorema (Representação do movimento)

Uma transformação $\vec{x} \mapsto \vec{\psi}(\vec{x})$ satisfaz

$$\|\vec{x} - \vec{y}\| = \|\vec{\psi}(\vec{x}) - \vec{\psi}(\vec{y})\|$$

se e somente se existe uma matriz ortogonal $Q \in \mathbb{R}^{d \times d}$ e um vetor $\vec{b} \in \mathbb{R}^d$ tais que

$$\vec{\psi}(\vec{x}) = Q \cdot \vec{x} + \vec{b}$$

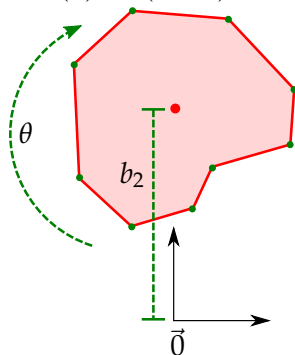
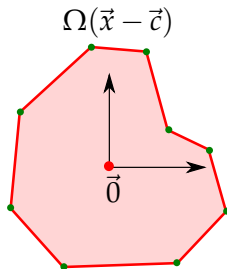
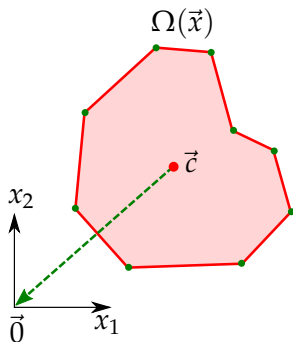
Definição (Movimento rígido)

São transformações da forma

$$\vec{\varphi}(\vec{x}, t) = Q(t) \cdot \vec{x} + \vec{b}(t)$$

Movimentos rígidos

$$\vec{\varphi}(\theta, b_2) = Q(\theta) \cdot \Omega(\vec{x} - \vec{c}) + \vec{b}$$



Transformação rígida que rota o corpo um ângulo θ e leva o centro $\vec{c}$ à posição $\vec{b} = (0, b_2)$.

Transformação das coordenadas dos nós

Lembremos que

$$\vec{\varphi}(\vec{x}) = Q_{\theta} (\vec{x} - \vec{c}) + \vec{b}$$

onde $\vec{x}$ pertence à configuração de referência (coor0).

```
n=4; coor0=[1 0;2 0;3 2;0 2];  
coor0=[coor0;[coor0(1,:)]];  
cg=[1.5;0.5];  
theta=60*(2*pi/360);b=[0;0];  
qq=[cos(theta) -sin(theta);sin(theta) cos(theta)];  
for i=1:n+1  
    coor(i,:)=(qq*(coor0(i,:)'-cg)+b)';  
endfor
```

Assim, podemos estudar empuxo e torque em diversas posições do corpo.

Projeto: Parte I - Estática

Consideramos um corpo bidimensional homogêneo cuja geometria é dada por um vetor de coordenadas $\text{coord}(n, 2)$ e cuja densidade é ρ (a densidade da água é 1, assim como a gravidade). O centro de massa está sempre em $x_1 = 0$, mas o corpo pode rotacionar e movimentar verticalmente. A superfície livre é horizontal ($x_2 = 0$).

- 1 Programe um código que, para cada valor de ρ , construa um gráfico do torque T como função do ângulo θ . A posição vertical para cada ângulo deve ser tal que o empuxo equilibre ao peso.
- 2 Identifique assim as posições de equilíbrio para cada ρ e analise sua estabilidade.

Dinâmica de uma partícula pontual

- Partícula de massa m , posição $\vec{x}$, força aplicada total $\vec{F}$.
- Momento linear: $\vec{p} = m \vec{v} = m \frac{d\vec{x}}{dt}$.
- Segunda lei de Newton: $m \frac{d^2\vec{x}}{dt^2} = \frac{d\vec{p}}{dt} = \vec{F}$.
- Energia cinética: $K = \frac{m}{2} \|\vec{v}\|^2$.

Momentos angulares

- Momento de uma força (torque) em $\vec{x}$ respeito de $\vec{0}$:

$$\vec{T} = \vec{x} \times \vec{F}$$

- Momento respeito de um eixo pela origem de direção $\check{e}$: $T_{\check{e}} = \check{e} \cdot \vec{T}$.
- **Momento angular respeito de $\vec{0}$:**

$$\vec{L} = \vec{x} \times \vec{p} = \vec{x} \times (m\vec{v})$$

- Conservação do momento angular:

$$\frac{d\vec{L}}{dt} = \underbrace{\frac{d\vec{x}}{dt} \times \vec{p}}_{=0} + \vec{x} \times \frac{d\vec{p}}{dt} = \vec{x} \times \vec{F} = \vec{T}$$

que é simplesmente uma re-escrita de $F = ma$.

Várias partículas

- Partícula k , massa m_k , posição $\vec{x}^k$, momento $\vec{p}^k = m_k \vec{v}^k$.
- Força (externa + inter-partículas):

$$\vec{F}^k = \vec{F}_e^k + \sum_{j \neq k} \vec{F}^{j \rightarrow k}$$

- Força total, momento linear total:

$$\vec{F} = \sum_k \left(\vec{F}_e^k + \sum_{j \neq k} \vec{F}^{j \rightarrow k} \right) = \sum_k \vec{F}_e^k, \quad \vec{p} = \sum_k \vec{p}^k$$

- Conservação do momento linear (segunda lei para várias partículas):

$$\frac{d\vec{p}}{dt} = \sum_k \frac{d\vec{p}^k}{dt} = \sum_k \left(\vec{F}_e^k + \sum_{j \neq k} \vec{F}^{j \rightarrow k} \right) = \vec{F}$$

Centro de massa

■ Massa total: $M = \sum_k m_k$.

■ Centro de massa:

$$\vec{c} = \frac{1}{M} \sum_k m_k \vec{x}^k \quad \rightarrow \quad \frac{1}{M} \int_{\Omega} \rho \vec{x} d\Omega$$

■ Então,

$$\vec{p} = M \frac{d\vec{c}}{dt}, \quad M \frac{d^2\vec{c}}{dt^2} = \vec{F}$$

o centro de massa se comporta como uma partícula pontual.

Momento angular de várias partículas

- Momento angular total, torque total:

$$\vec{L} = \sum_k \vec{L}^k = \sum_k \vec{x}^k \times \vec{p}^k, \quad \vec{T} = \sum_k \vec{x}^k \times \vec{F}_e^k$$

- Conservação do momento angular:

$$\begin{aligned} \frac{d\vec{L}}{dt} &= \sum_k \frac{d\vec{L}^k}{dt} = \sum_k \vec{x}^k \times \left(\vec{F}_e^k + \sum_{j \neq k} \vec{F}^{j \rightarrow k} \right) = \\ &= \sum_k \vec{x}^k \times \vec{F}_e^k + \underbrace{\sum_k \left(\vec{x}^k \times \sum_{j \neq k} \vec{F}^{j \rightarrow k} \right)}_{=0!} = \vec{T} \end{aligned}$$

- Essa equação é agora **independente da conservação do momento linear**.

Equações dinâmicas de corpo rígido

- Um conjunto de partículas deve satisfazer as 6 EDOs (3 em 2D)

$$\begin{aligned}\frac{d\vec{p}}{dt} &= \vec{F} \\ \frac{d\vec{L}}{dt} &= \vec{T}\end{aligned}$$

- Se o conjunto é rígido, ele tem coincidentemente 6 graus de liberdade apenas (3 em 2D).
- **Felizmente**, as 6 EDOs **determinam totalmente** os 6 graus de liberdade! As equações dinâmicas são um sistema fechado.
- Apenas devemos trabalhar um pouco mais para levar à forma final...

Momento angular de corpo rígido

■ Movimento rígido:

$$\vec{x}^k(t) = \vec{\varphi}(\vec{X}^k, t) = Q(t)\vec{X}^k + \vec{b}(t), \quad \vec{X}^k = Q(t)^T(\vec{x}^k(t) - \vec{b}(t))$$

■ Velocidade:

$$\vec{v}^k(t) = \frac{dQ}{dt}(t)\vec{X}^k = \frac{dQ}{dt}(t) Q(t)^T(\vec{x}^k(t) - \vec{b}(t))$$

■ $Q(t)Q(t)^T = \mathbb{I} \Rightarrow \frac{dQ}{dt} Q^T$ é antissimétrica.

$$0 = \frac{d}{dt}(Q Q^T) = \frac{dQ}{dt} Q^T + Q \frac{dQ^T}{dt} = \frac{dQ}{dt} Q^T + \left(\frac{dQ}{dt} Q^T \right)^T$$

- Produto com matrizes antissimétricas é equivalente a produto vetorial por (pseudo)vetor.

$$A(\vec{\omega}) \vec{z} = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{vmatrix} \check{\mathbf{i}} & \check{\mathbf{j}} & \check{\mathbf{k}} \\ \omega_1 & \omega_2 & \omega_3 \\ z_1 & z_2 & z_3 \end{vmatrix} = \vec{\omega} \times \vec{z}$$

- No caso 2D, simplesmente tomar $\vec{\omega} = \omega \check{\mathbf{k}}$, e $A = \begin{pmatrix} 0 & -\omega \\ \omega & 0 \end{pmatrix}$
- $\vec{v}^k(t) = \vec{\omega}(t) \times (\vec{x}^k(t) - \vec{b}(t)) = \vec{\omega}(t) \times \vec{x}^k(t) + \vec{V}(t)$
Onde $\vec{V}$ é a velocidade da partícula atualmente em $\vec{0}$.
- Notar que, sendo $A(\vec{\omega}) = \frac{dQ}{dt} Q^T$,

$$\frac{dQ}{dt} = A(\vec{\omega}) Q$$

Em 2D isto é simplesmente $\frac{d\theta}{dt} = \omega$.

- Substituindo,

$$\vec{L} = \sum_k \vec{x}^k \times m_k \vec{V} + \sum_k m_k \vec{x}^k \times (\vec{\omega} \times \vec{x}^k)$$

o primeiro termo é zero se tomamos momentos respeito de $\vec{c}(t)$.

- Utilizando $\vec{a} \times (\vec{b} \times \vec{c}) = (\vec{a} \cdot \vec{c}) \vec{b} - (\vec{a} \cdot \vec{b}) \vec{c}$,

$$\vec{L} = \underbrace{\left[\sum_k m_k \left(\|\vec{x}^k\|^2 \mathbb{I} - \vec{x}^k (\vec{x}^k)^T \right) \right]}_{\mathcal{J}} \vec{\omega} \rightarrow \left[\int_{\Omega} \rho (\|\vec{x}\|^2 \mathbb{I} - \vec{x} \vec{x}^T) d\Omega \right] \vec{\omega}$$

- $\mathcal{J}$ é o **tensor (matriz) de inércia angular**.

$$\vec{T} = \frac{d\vec{L}}{dt} = \frac{d}{dt} (\mathcal{J} \vec{\omega}) = \mathcal{J} \frac{d\vec{\omega}}{dt} + \underbrace{\vec{\omega} \times (\mathcal{J} \vec{\omega})}_{0 \text{ em 2D}} \quad \left(\frac{d\mathcal{J}}{dt} = 0 \text{ em 2D} \right)$$

Equações finais em 2D

$$\frac{d}{dt} \begin{pmatrix} \vec{c} \\ \theta \\ \vec{p} \\ L \end{pmatrix} = \begin{pmatrix} \vec{V} \\ \omega \\ \vec{F} \\ T \end{pmatrix} = \begin{pmatrix} \frac{1}{M} \vec{p} \\ \frac{1}{\mathcal{J}} L \\ \vec{F} \\ T \end{pmatrix}$$

- $\mathcal{J} = \int_{\Omega^0} \rho(x_1^2 + x_2^2) d\Omega^0$, pre-calculado, independente do tempo.
- Forças totais = Peso + Empuxo + “Amortecimento” ($= -\beta \vec{V}$, onde β poderia ser matriz diagonal)
- Torque total = Torque líquido + “Amortecimento” ($= -\gamma \omega$)

Projeto: Parte II - Dinâmica

Consideramos o mesmo corpo da Parte I. A posição inicial é dada por $\vec{c}(0)$ e $\theta(0)$, a velocidade inicial é nula. Pede-se um código que resolva a dinâmica com a superfície dada por uma função $Z(x_1, t)$ arbitrária.

- 1 Calculamos resposta a perturbações nos equilíbrios calculados na parte I, com $Z(x_1, t) = 0$. Ajustar β e γ para que o sistema seja levemente amortecido.
- 2 Calculamos resposta a uma onda cuja frequência seja menor/parecida/maior que as frequências de oscilação do sistema (ponto anterior).

Equações finais em 3D

- No caso 2D a orientação era um ângulo $\theta(t)$. Em 3D ela é dada pela matriz $Q(t)$.
- No caso 2D o tensor $\mathcal{J}$ é um número constante. Em 3D é uma matriz **que varia com a orientação**. Se $\mathcal{J}_0$ é o momento de inércia na posição de referência,

$$\mathcal{J}(t) = Q(t) \mathcal{J}_0 Q(t)^T, \quad \mathcal{J}^{-1} = Q \mathcal{J}_0^{-1} Q^T$$

$$\frac{d}{dt} \begin{pmatrix} \vec{c} \\ Q \\ \vec{p} \\ \vec{L} \end{pmatrix} = \begin{pmatrix} \vec{V} \\ A(\vec{\omega}) Q \\ \vec{F} \\ \vec{T} \end{pmatrix} = \begin{pmatrix} \frac{1}{M} \vec{p} \\ A(Q \mathcal{J}_0^{-1} Q^T \vec{L}) Q \\ \vec{F} \\ \vec{T} \end{pmatrix}$$

- Os graus de liberdade de rotação são 3, **porém estamos representando as rotações com uma matriz ortogonal (9 incógnitas!)**.
- O sistema depende fortemente de $Q(t)$ ser ortogonal para todo t .
- **Exercício:** Prove que, se Q satisfaz $dQ/dt = A(t)Q(t)$ com $A(t)$ antissimétrica, e $Q(0)$ é ortogonal, então $Q(t)$ é ortogonal $\forall t$.
- **Quando se utiliza resolução numérica, a matriz Q perde ortogonalidade por erro de aproximação!**
- **Exercício:** Provar que a aproximação de Euler $Q_{n+1} = Q_n + \Delta t A_n Q_n$ faz que Q_{n+1} não seja ortogonal, de fato

$$Q_{n+1}Q_{n+1}^T = \mathbb{I} - (\Delta t A_n)^2.$$

- Ambas dificuldades acima levam a preferir os **quatérnions**.

- Quatérnions são uma melhor representação das rotações do que matrizes.
- Um quatérnion é um par escalar-vetor $q = [s, \vec{v}]$ cuja multiplicação é definida como

$$[s, \vec{v}][r, \vec{u}] = [s r - \vec{v} \cdot \vec{u}, s \vec{u} + r \vec{v} + \vec{v} \times \vec{u}]$$

- Conjugação: $q^* = [s, -\vec{v}]$. Também: $(p q)^* = q^* p^*$.
- Norma: $\|q\|^2 = q q^* = s^2 + \|\vec{v}\|^2$.
- Inversa: $q^{-1} = q^* / \|q\|^2$.

- Uma rotação de ângulo θ em torno de uma direção unitária $\vec{u}$ é representada pelo quatérnion unitário

$$q_{(\theta, \vec{u})} = [\cos(\frac{\theta}{2}), \sin(\frac{\theta}{2}) \vec{u}] , \quad \|q\| = \cos^2(\frac{\theta}{2}) + \sin^2(\frac{\theta}{2}) \|\vec{u}\|^2 = 1$$

- Rotação de vetores: $x = [0, \vec{x}] \Rightarrow q x q^* = [0, Q_{\theta, \vec{u}} \vec{x}]$
- Composição de rotações:

$$[0, \vec{y}] = [0, Q(q)Q(p)\vec{x}] \Rightarrow [0, \vec{y}] = q p [0, \vec{x}] p^* q^*$$

Isto é, $Q(qp) = Q(q)Q(p)$.

- A matriz associada a um quatérnion unitário $q = [s, \vec{v}]$ é

$$Q(q) = \begin{pmatrix} 1 - 2v_2^2 - 2v_3^2 & 2v_1v_2 - 2sv_3 & 2v_1v_3 + 2sv_2 \\ 2v_1v_2 + 2sv_3 & 1 - 2v_1^2 - 2v_3^2 & 2v_2v_3 - 2sv_1 \\ 2v_1v_3 - 2sv_2 & 2v_2v_3 + 2sv_1 & 1 - 2v_1^2 - 2v_2^2 \end{pmatrix}$$

- Em termos de quatérnions, a equação $Q'(t) = A(\vec{\omega}(t)) Q(t)$ se reescreve

$$\frac{dq}{dt} = \frac{1}{2} [0, \vec{\omega}(t)] q(t)$$

Equações 3D com quatérnions

$$\frac{d}{dt} \begin{pmatrix} \vec{c} \\ q \\ \vec{p} \\ \vec{L} \end{pmatrix} = \begin{pmatrix} \vec{V} \\ \frac{1}{2} [0, \vec{\omega}] q \\ \vec{F} \\ \vec{T} \end{pmatrix} = \begin{pmatrix} \frac{1}{M} \vec{p} \\ \frac{1}{2} [0, Q(q) \mathcal{J}_0^{-1} Q(q)^T \vec{L}] q \\ \vec{F} \\ \vec{T} \end{pmatrix}$$

- Representam a mesma física com menos incógnitas (13 em vez de 18).
- Sempre que q se mantenha unitário, não há problema de $Q(q)$ deixar de ser matriz ortogonal (conveniência numérica).

O sistema é assim transformado em

$$y'(t) = f(t, y(t)) , \quad \text{onde } y = (\vec{c}, q, \vec{p}, \vec{L})^T$$

Algoritmo para avaliar $f(t, y)$

- 1 Normalizar q .
- 2 Calcular $\vec{V} = \vec{p}/M$.
- 3 De q , calcular $Q(q)$ (fórmula matriz associada).
- 4 Calcular $\mathcal{J}^{-1} = Q(q)\mathcal{J}_0^{-1}Q(q)^T$. Notar que $\mathcal{J}_0^{-1}$ foi pre-calculado no começo.
- 5 Calcular $\vec{\omega} = \mathcal{J}^{-1}\vec{L}$.
- 6 Calcular $\vec{F}$ e $\vec{T}$, funções de t e y .
- 7 $f(t, y) = (\vec{V}, \frac{1}{2}[0, \vec{\omega}]q, \vec{F}, \vec{T})^T$.

- D. Baraff. An introduction to physically based modeling. SIGGRAPH'97 Course Notes.
- M. Mason. Mechanics of Manipulation, 2010. Course Notes.

Equações finais da dinâmica 2D $\Rightarrow \boxed{\frac{dy}{dt}(t) = f(t, y(t))}$ onde

$$y = \begin{pmatrix} c_1 \\ c_2 \\ \theta \\ p_1 \\ p_2 \\ L \end{pmatrix}, \quad f(t, y) = \begin{pmatrix} p_1/M \\ p_2/M \\ L/\mathcal{J} \\ F_1^h(t, \vec{X}(\vec{c}, \theta)) - \beta_1 p_1/M \\ F_2^h(t, \vec{X}(\vec{c}, \theta)) - \beta_2 p_2/M - Mg \\ T^h(t, \vec{X}(\vec{c}, \theta)) - \gamma L/\mathcal{J} \end{pmatrix},$$

(F_1^h, F_2^h) o empuxo, $\vec{X}(\vec{c}, \theta)$ as posições dos vértices do polígono ($\vec{X} = \vec{c} + Q(\theta)(\vec{X}^{ref} - \vec{c}^{ref})$), T^h o torque hidrostático.

Solução numérica de $y' = f(t, y)$

- Seja $Y(t)$ a solução exata de $Y'(t) = f(t, Y(t))$, $Y(0) = Y^0$.
- Métodos numéricos são receitas para calcular um conjunto de vetores $y^0, y^1, \dots$, que aproximam sistematicamente $Y(0), Y(t_1), Y(t_2), \dots$, para uma sequência de tempos $t_1, t_2, \dots$.
- **Exemplo mais simples:** Método de Euler com passo fixo definido pelo usuário.

$$y^0 = Y^0, \quad t_k = k \Delta t, \quad y^{k+1} = y^k + \Delta t f(t_k, y^k)$$

É um método explícito porque não requer inverter f , como é na versão implícita $y^{k+1} = y^k + \Delta t f(t_{k+1}, y^{k+1})$.

- Nos métodos explícitos, o custo computacional está na avaliação da função f . Para o método de Euler: 1 avaliação por passo.
- Esses métodos se tornam instáveis se $\Delta t > c/J_f$, com $c \simeq 2$ e J_f o maior autovalor em módulo da matriz Jacobiana de f .
- Por J_f depender de y o limite de estabilidade pode apenas ser *adivinhado*.
- Métodos Runge-Kutta constituem bons compromissos entre custo (número de avaliações de f) e precisão ($\|y^k - Y(t_k)\|$).
- A **precisão** é usualmente avaliada pela **ordem do método**:
 $\|y^k - Y(t_k)\| \leq C \Delta t^p$.

Métodos de Runge-Kutta e Tabela de Butcher

$$y^{n+1} = y^n + \Delta t (b_1 k_1 + b_2 k_2 + \dots + b_s k_s)$$

onde

$$k_1 = f(t_n, y^n)$$

$$k_2 = f(t_n + c_2 \Delta t, y^n + \Delta t (a_{21} k_1))$$

$$k_3 = f(t_n + c_3 \Delta t, y^n + \Delta t (a_{31} k_1 + a_{32} k_2))$$

...

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

$$y^{n+1} = y^n + \Delta t (b_1 k_1 + b_2 k_2 + \dots + b_s k_s)$$

onde

$$k_1 = f(t_n, y^n)$$

$$k_2 = f(t_n + c_2 \Delta t, y^n + \Delta t (a_{21} k_1))$$

$$k_3 = f(t_n + c_3 \Delta t, y^n + \Delta t (a_{31} k_1 + a_{32} k_2))$$

$$\dots \quad \dots$$

```
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 ordem 2

$$y^{n+1} = y^n + \Delta t \left(\frac{1}{2} k_1 + \frac{1}{2} k_2 \right)$$

$$k_1 = f(t_n, y^n)$$

$$k_2 = f(t_n + \Delta t, y^n + \Delta t k_1)$$

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.;
```

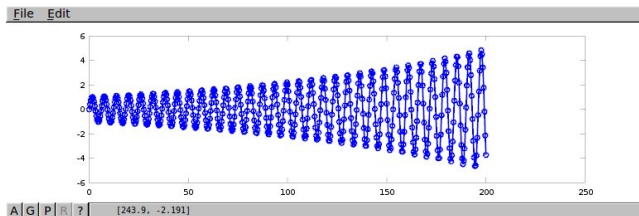
```
    f(1)=y(2);
```

```
    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:

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
	$\frac{1}{6}$	$\frac{2}{6}$	$\frac{2}{6}$	$\frac{1}{6}$

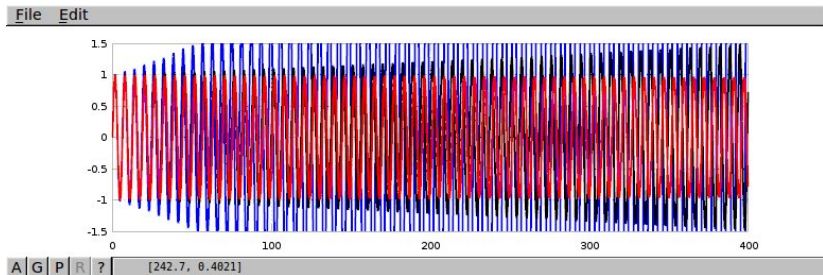
■ Mesmo código do slide anterior, com

```
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];
```

Comparação

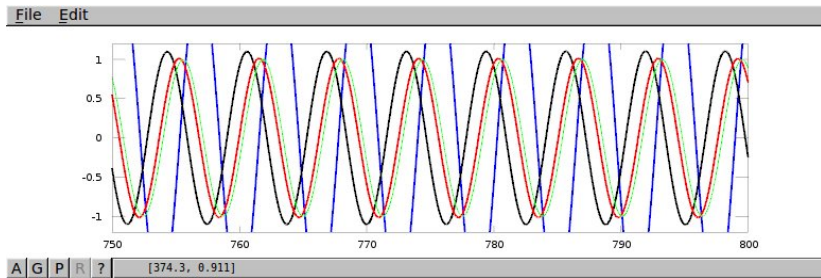
Comparamos RK2 com $\Delta t = 0.4$ e 0.2 , e RK4 com $\Delta t = 0.4$.

```
>[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])
```



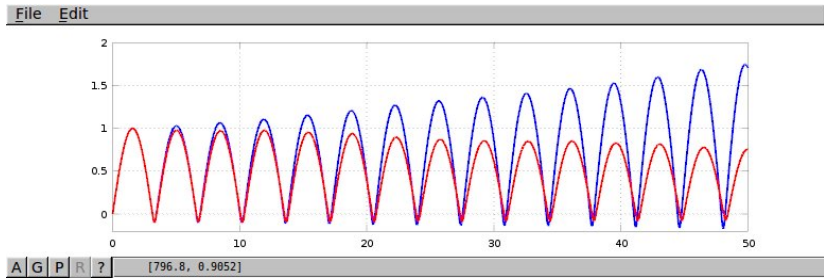
Qual é o Δt tal que a $t = 800$ a amplitude continua sendo $\simeq 1$ com RK2?

```
>[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])
```

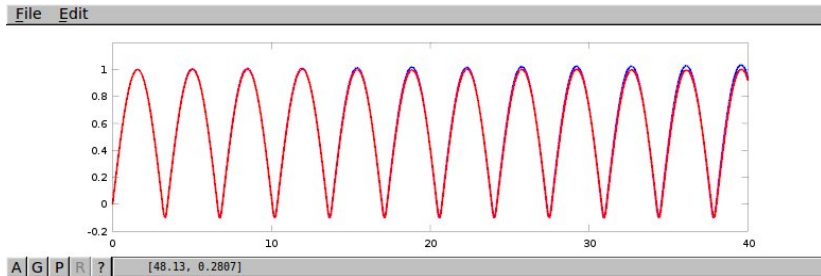


Massa-mola com parede

```
function f = foscilw(y,t)
ome=1.;f(1)=y(2);
if (y(1)>0) f(2)=-ome*ome*y(1);
else f(2)=-100*ome*ome*y(1);
endif
end
>[y1 time1]=rk2("foscilw",[0;1],0,0.1,500);
>[y2 time2]=rk4("foscilw",[0;1],0,0.2,250);
>plot(time1,y1(1,:),"-b","linewidth",2,time2,y2(1,:),"-r",...
"linewidth",2)
> axis([0 50 -.2 2])
```



```
>[y1 time1]=rk2("foscilw",[0;1],0,0.02,2000);  
>[y2 time2]=rk4("foscilw",[0;1],0,0.04,1000);  
>plot(time1,y1(1,:),"-b","linewidth",2,time2,y2(1,:),"-r",...  
"linewidth",2)  
> axis([0 40 -.2 1.2])
```



Ajuste automático de passo

Ideia geral: Em cada passo de tempo,

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

$$e = \|y^{n+1} - z^{n+1}\|$$

- Se (erro > tolerância) reduzir Δt e voltar a 1.
- 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 a tolerância.

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

$$\Delta t \leftarrow \max(0.5\Delta t, \min(2\Delta t, 0.9\Delta t_*))$$

Embedded Runge-Kutta methods (ERKM)

- Para ajustar automaticamente Δt são precisos **2 métodos** de diferente ordem.
- Em geral, esses dois métodos requerem avaliações **em pontos diferentes**.
- ERKM consegue construir os dois métodos **maximizando o número de pontos comuns**.
- São **pares otimizados de métodos RK**. Os mais utilizados são os de **Fehlberg**, de **Dormand-Prince**, entre outros. Matlab e Octave utilizam ERKM.

Método de Dormand-Prince

0	0	0	0	0	0	0	0
$\frac{1}{5}$	$\frac{1}{5}$	0	0	0	0	0	0
$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$	0	0	0	0	0
$\frac{4}{5}$	$\frac{44}{45}$	$-\frac{56}{15}$	$\frac{32}{9}$	0	0	0	0
$\frac{8}{9}$	$\frac{19372}{6561}$	$-\frac{5360}{2187}$	$\frac{64448}{6561}$	$-\frac{212}{729}$	0	0	0
1	$\frac{9017}{3168}$	$-\frac{355}{33}$	$\frac{46732}{5247}$	$\frac{49}{176}$	$-\frac{5103}{18656}$	0	0
1	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	0
	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	0
	$\frac{5179}{57600}$	0	$\frac{7571}{16695}$	$\frac{393}{640}$	$-\frac{92097}{339200}$	$\frac{187}{2100}$	$\frac{1}{40}$

Há 2 vetores b , o primeiro é de ordem 4, o segundo de ordem 5.

Utilizando lsode (Octave)

- A função `lsode` implementa as melhoras discutidas.
- Se especifica a função e os tempos em que se deseja a solução, Δt é automático.

```
T=[0:0.2:40];  
[X, istate, MSG]=lsode("foscilw",[0 1],T);  
plot(T,X,"linewidth",2)
```

