Uma introdução Teórica e Numérica em alguns problemas clássicos de EDP's
(PARTE 3)

Prof.: Pitágoras Pinheiro de Carvalho - **(UESPI)**

(Escola de Verão 2021 - UFPB)

26 de fevereiro de 2021

Programa de Pós Graduação em
MATEMÁTICA

Sumário

Alguns comandos para plots:

Comandos úteis:

- plot(u) (plota a solução)
- Th (agrega malha no plot)
- wait=true (Necessita clicar enter para atualizar a figura)
- wait=false (Atualização automatica da figura)
- value=true (Agrega uma escala de autovalores na lateral)
- fill=true (Plota a figura colorida)
- fill=false (Plota somente as curvas dos autovalores)

  Assim:

- plot(u, Th, wait=false, value=true, fill=true);
  Plota a figura com malha, atualiza automaticamente os plots, agrega uma escala de autovalores na lateral e gera a figura em cores.
- plot(u, Th, dim=3, wait=false, value=true, fill=true);
  Plota a figura do item anterior em 3D.

Nosso problema (**Onda 2D**):

- Vamos agora considerar a equação da onda, em $\Omega = (0,\ 1)^2$:

$$\begin{cases} \dfrac{\partial^2 u}{\partial t^2} - c^2 \Delta u = f & \text{em} \quad \Omega \times (0,3), \\ u(x,y,\underbrace{0}_{tempo}) = u_t(x,y,\underbrace{0}_{tempo}) = 0 & \text{em} \quad \Omega, \\ \dfrac{\partial u}{\partial \eta} = g & \text{sobre} \quad \partial\Omega. \end{cases} \qquad \textbf{(3)}$$

onde temos: $\underbrace{f = x * y * t,}_{\text{termo força}}$ $\underbrace{g = 1}_{\text{condição de fronteira}}$, $\underbrace{t = m \cdot \tau}_{\text{tempo}}$ e $\underbrace{\tau = 0.5}_{\text{dt}}$.

- A derivada parcial da solução na variável temporal é dada por:

$$\frac{\partial^2 u}{\partial t^2}(x,y,t) = \lim_{\tau \to 0} \frac{u(x,y,t+\tau) + 2u(x,y,t) - u(x,y,t-\tau)}{\tau^2}.$$

- Como $t = m\tau$, logo: $t + \tau = (m+1)\tau$ e $t - \tau = (m-1)\tau$.

Discretização (**Diferenças Finitas em t**):

- Isto indica que $u^m(x,y) = u(x,y,m\tau)$, deverá satisfazer (aproximadamente):

$$\frac{\partial^2 u}{\partial t^2}(x,y,m\tau) \simeq \frac{u^{m+1}(x,y) - 2u^m(x,y) + u^{m-1}(x,y)}{\tau^2}.$$

Assim, $(\forall m = 0, ..., T/\tau)$ **a discretização da equação da onda no tempo é dada por**:

$$\begin{cases} \dfrac{u^{m+1} - 2u^m + u^m}{\tau^2} - c^2 \Delta u^{m+1} = f^{m+1} & \text{em } \Omega \times (0,3), \\ u^0(x,y) = u_0(x,y) \quad e \quad u^1(x,y) = u_1(x,y) & \text{em } \Omega, \\ \dfrac{\partial u^{m+1}}{\partial \eta} = g & \text{sobre } \partial\Omega. \end{cases} \qquad \textbf{(4)}$$

O problema e o programa:

- O problema:

$$\int_{\Omega}\{u^{m+1}v+\tau^2c^2(\nabla u^{m+1}\cdot\nabla v)\}-\int_{\Omega}\{2u^m v-u^{m-1}v+\tau^2 f^{m+1}v\}=0\ ,$$

$$\text{com}\quad \left.\frac{\partial u^{m+1}}{\partial\eta}\right|_{\partial\Omega}=g.$$

- O programa:

problem **dWave** (u, v) = int2d(Th)(u*v + τ^2*c^2*(dx(u)*dx(v) + dy(u)*dy(v)))
- int2d(Th)(2uu*v - uu1*v + τ^2*f*v)
+ int1d(Th)(g*v);

O domínio parametrizado:

```
load "medit"
verbosity=0;
//——- NUMERICAL VALUES TO PROVIDE———
int C0, C1, C2, C3; // the labels

//———-DESCRIPTION OF THE BOUNDARIES—————–
border C01(t=0.,1.){ x=-0.5 + 10.*t; y=-4.; label=C0;}
border C02(t=0.,1.){ x=9.5; y=-4. + 8.*t; label=C1;}
border C03(t=0.,1.){ x=9.5 -10.*t; y=4.; label=C2;}
border C04(t=0.,1.){x=-0.5; y=4.-8.*t; label=C3;}

//——PLOT BOUNDARIES:———
plot ( C01(40) + C02(40)+ C03(40)+ C04(40), wait=true);

// ———-MESH————–
mesh Th=buildmesh(C01(40) + C02(40)+ C03(40)+ C04(40));

//—— PLOT MESH———
plot(Th);
```

PLOT:

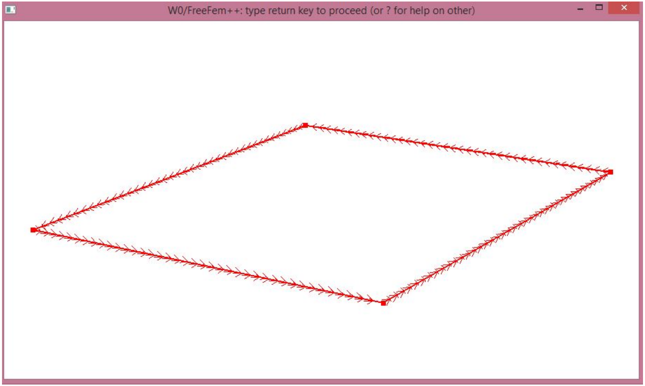


Figura: PARAMETRIZADA

PLOT:

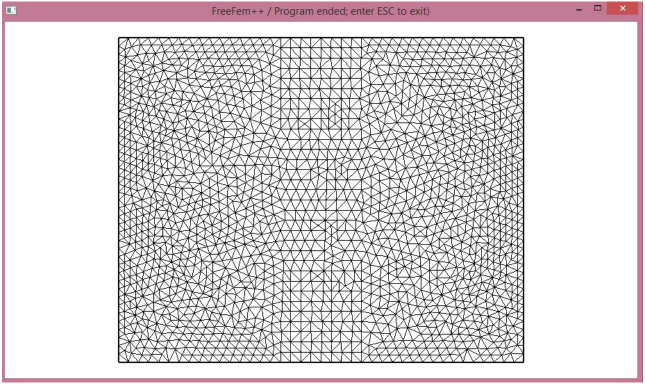


Figura: PARAMETRIZADA E MALHADA

MALHA UNIFORME:

```
mesh Th = square(20, 20);
plot(Th);
```

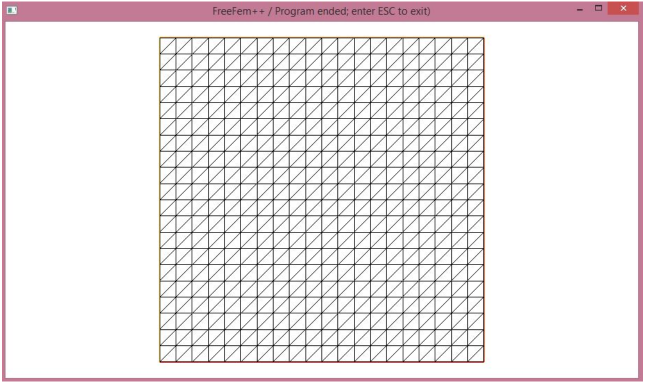


Figura: PARAMETRIZADA E MALHADA

Nosso terceiro programa (**Onda 2D**):

// Parametros (dt e μ):
real dt = 0.5, dt2=dt*dt;
real mu = 0.5, mu2=mu*mu; // mu=c, mu2=c^2.

// Malha (Uniforme no quadrado unitário):
mesh Th = square(20, 20);

// Espaço de Funções:
fespace Vh(Th, P1);
Vh u, v, uu, uu1, f, ff, g;

// Problema:
problem **dWave** (u, v) = int2d(Th)(u*v + dt2*mu2*(dx(u)*dx(v) + dy(u)*dy(v)))
- int2d(Th)(dt2*ff*v)
+ int1d(Th)(g*v);

Nosso programa (**Onda 2D**):

real t = 0; // **Tempo inicial:**
uu = 0, uu1=0. ; // **Condições Iniciais:**

for (int m = 0; m < = 10/dt; m++) // **Atualizações:**
{
$t = t + dt$; // **Atualização do Tempo (** $t_{m+1} = t_m + dt$**):**
$f = x * y * t$; // **Termo de força:**
$g = 1$; // **Condição na Fronteira:**
$ff = (2. * uu - uu1) + dt2 * f$; // **O "lado direito" da equação:**
$uu = u$; // **Atualização da condição inicial:**

$uu1 = uu$; // **Atualização da condição inicial:**

dWave; // **Solicitação do sistema dWave:**

plot(u, value=true, fill=true, wait=true); // **Plotar a solução** u**:**
cout ≪ "t="≪ t ≪ - L2 - Error ="≪ sqrt(int2d(Th)($u * u$)) ≪ endl;
}

$\Uparrow$

// **Atualização de** $\|u\|_{L^2}$ **em cada atualização de tempo t.**

Estado final **u**:

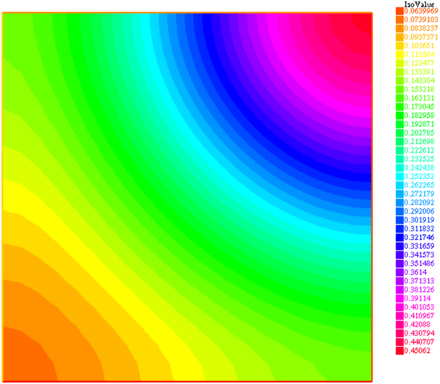


Figura: Estado final para $g\Big|_{\partial\Omega} = 0$

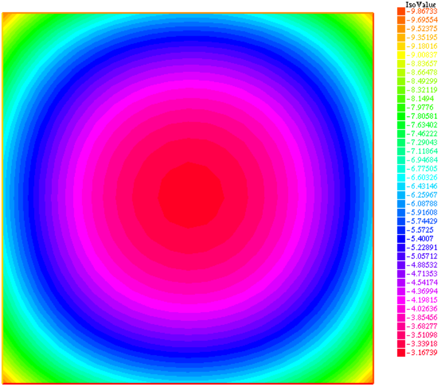


Figura: Estado final para $g\Big|_{\partial\Omega} = 1$

Condições mistas: (Dirichlet + Neumann)

- Para casos em que é necessário especificar mais de uma condição de fronteira, tem-se:

$$
\left\{
\begin{array}{ll}
\dfrac{u^{m+1} - 2u^m + u^m}{\tau^2} - c^2 \Delta u^{m+1} = f^{m+1} & \text{em} \quad \Omega \times (0,3), \\
u^0(x,y) = u_0(x,y) \quad e \quad u^1(x,y) = u_1(x,y) & \text{em } \Omega, \\
u^{m+1} = h & \text{sobre} \quad \partial\Omega_D, \\
\dfrac{\partial u^{m+1}}{\partial \eta} = g & \text{sobre} \quad \partial\Omega_N
\end{array}
\right.
\qquad \textbf{(D+N)}
$$

sendo $\partial\Omega = \partial\Omega_D + \partial\Omega_N$.

- Assim, na programação tem-se:

problem **dWave** (u, v) = int2d(Th)(u*v + dt2*mu2*(dx(u)*dx(v) + dy(u)*dy(v)))
- int2d(Th)(dt2*ff*v)
+ on($\partial\Omega_D$, u=h);
+ int1d(Th,$\partial\Omega_N$)(g*v);

O programa:

- Basta substituir no programa da Onda o problem **dWave** , por

problem **dWave** (u, v) = int2d(Th)(u*v + τ^2*c^2*(dx(u)*dx(v) + dy(u)*dy(v)))
- int2d(Th)(2uu*v - uu1*v + τ^2*f*v)
+ int1d(Th,3,4)(g*v) + on(1,2,u=1);

Estado final **u**:

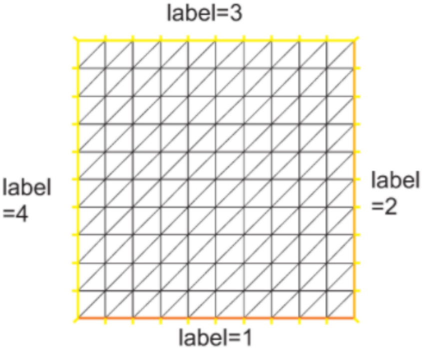


Figura: O domínio

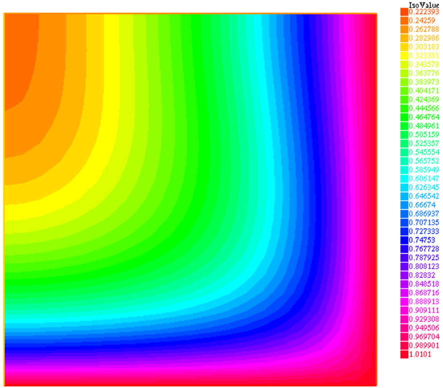


Figura: Em 1 e 2 Dirichlet (com u=1), 3 e 4 Neumann (com g=0).

Um pouco mais aplicado (Domínios Externos)...

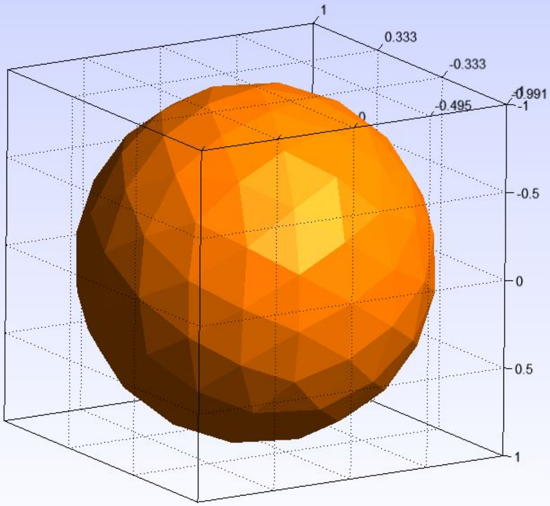


Figura: GMSH: salvar .msh ou .mesh

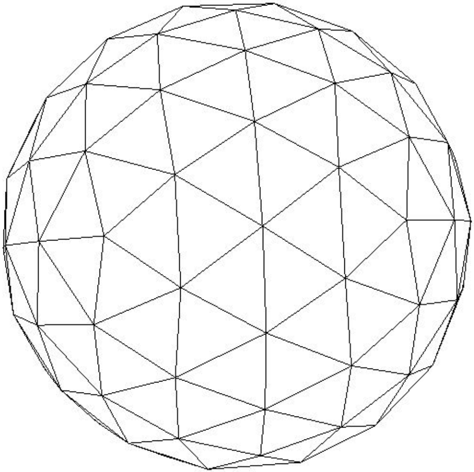
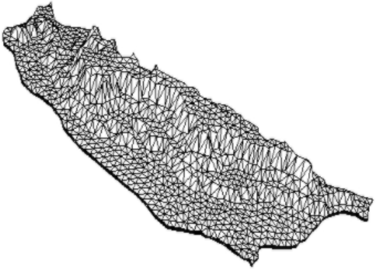


Figura: FREEFEM++



Taiwan 3D mesh

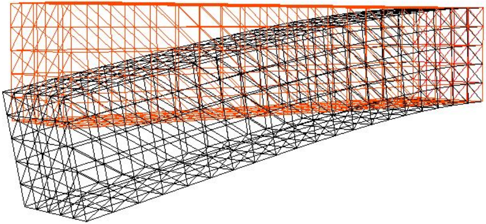


Figura: FREEFEM++

SimVascular

Gmsh

FREEFEM

Figura: SimVascular: AORTA-FEMORAL (Imagem DICON)

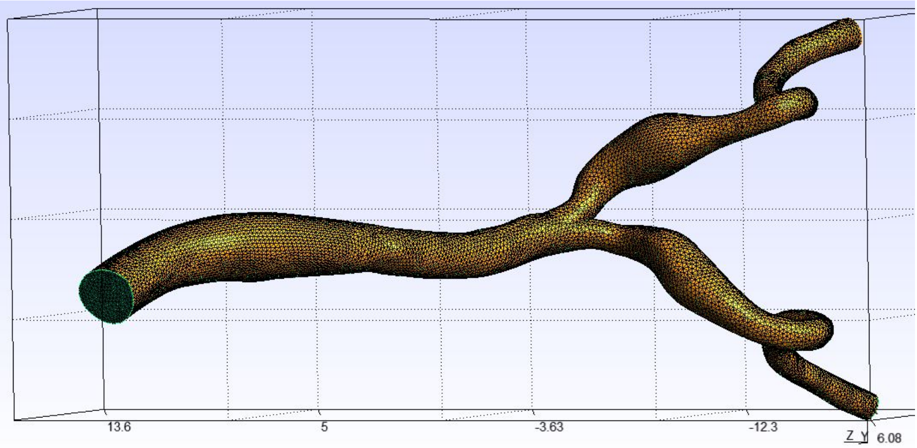


Figura: GMSH

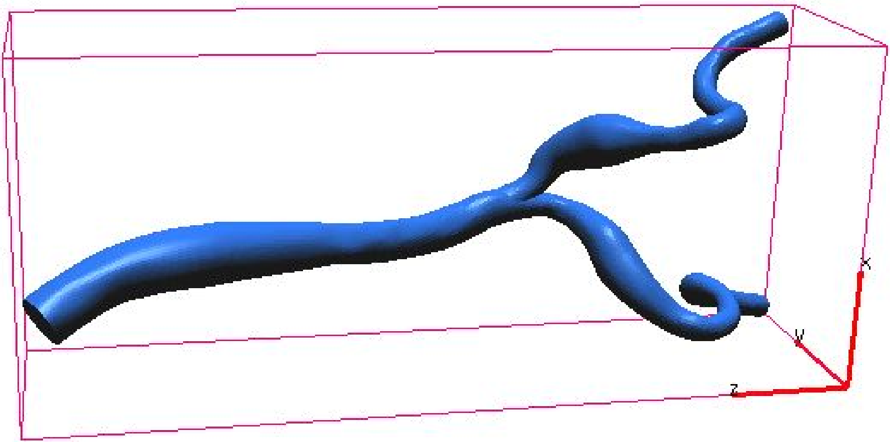


Figura: Freefem++

Referências Bibliográficas

F. Hecht. New development in freefem++. J. Numer. Math., 20(3-4):251–265, 2012.

F. Hecht.
https://www.ljll.math.upmc.fr/hecht/ftp/ff++/2019-Kenitra/Cours-Kenitra-2019.pdf

Obrigado pela presença na parte 3!