Método de Elementos Finitos

Pedro H A Konzen

24 de agosto de 2018

Licença

Prefácio

Nestas notas de aula são abordados temas introdutórios sobre o método de elementos finitos para a simulação de equações diferenciais. Como ferramenta computacional de apoio didático, faço uso de códigos em python com suporte da biblioteca FEniCS.

Agradeço aos(às) estudantes e colegas que assiduamente ou esporadicamente contribuem com correções, sugestões e críticas em prol do desenvolvimento deste material didático.

Pedro H A Konzen

Sumário

Capítulo 1

Método de elementos finitos em 1D

1.1 Fundamentos preliminares

Seja dado um intervalo $I = [x_0, x_1] \subset \mathbb{R}$, $x_0 \neq x_1$. O espaço vetorial das funções lineares em I é definido por

$$P_1(I) := \{v : \ v(x) = c_0 + c_1 x, \, x \in I, \, c_0, c_1 \in \mathbb{R}\}. \tag{1.1}$$

Observamos que dado $v \in P_1(I)$, temos que v é unicamente determinada pelos valores $\alpha_0 = v(x_0)$ e $\alpha_1 = v(x_1)$. Como consequência, existe exatamente uma única função $v \in P_1(I)$ para quaisquer dados valores α_0 e α_1. Desta observação, introduzimos a chamada base nodal $\{\varphi_0, \varphi_1\}$ para $P_1(I)$, definida por

$$\varphi_j(x_i) = \left\{ \begin{array}{ll} 1 & , i = j, \\ 0 & , i \neq j \end{array} \right. , \tag{1.2}$$

com $i,j = 0,1$. Com esta base, toda função $v \in P_1(I)$ pode ser escrita como uma combinação linear das funções φ_0 e φ_1 com coeficientes α_0 e α_1 (**graus de liberdade**), i.e.

$$v(x) = \alpha_0 \varphi_0(x) + \alpha_1 \varphi_1(x_1). \tag{1.3}$$

Além disso, observamos que

$$\varphi_0(x) = \frac{x_1 - x}{x_1 - x_0}, \quad \varphi_1(x) = \frac{x - x_0}{x_1 - x_0}. \tag{1.4}$$

Uma extensão do espaço $P_1(I)$ é o espaço das funções lineares por partes. Dado $I = [l_0, l_1]$, $l_0 \neq l_1$, consideremos uma partição (**malha**) de I com $n+1$ pontos $\mathcal{I} = \{l_0 = x_0, x_1, \ldots, x_n = l_1\}$ e, portanto, com n subintervalos $I_i = [x_{i-1}, x_i]$ de comprimento $h_i = x_i - x_{i-1}$, $i = 1, 2, \ldots, n$. Na malha $\mathcal{I}$ definimos o seguinte espaço das funções lineares por partes

$$V_h := \{v :\ v \in C^0(\mathcal{I}),\, v|_{I_i} \in P_1(I_i),\, i = 1{,}2, \ldots, n\}. \tag{1.5}$$

Observamos que toda função $v \in V_h$ é unicamente determinadas por seus valores nodais $\{\alpha_i = v(x_i)\}_{i=0}^n$. Reciprocamente, todo conjunto de valores nodas $\{\alpha_i\}_{i=0}^n$ determina unicamente uma função $v \in V_h$. Desta observação, temos que os valores nodais determinam os graus de liberdade com a base nodal $\{\varphi_j\}_{j=0}^n$ para V_h definida por

$$\varphi_j(x_i) = \left\{\begin{array}{ll} 1 & , i = j, \\ 0 & , i \neq j \end{array}\right. , \tag{1.6}$$

com $i{,}j = 0{,}1, \ldots, n$. Podemos verificar que

$$\varphi_i(x) = \left\{\begin{array}{ll} (x - x_{i-1})/h_i & , x \in I_i, \\ (x_{i+1} - x)/h_{i+1} & , x \in I_{i+1}, \\ 0 & , \text{noutros casos} \end{array}\right. \tag{1.7}$$

veja, Figura 1.1. É notável que $\phi_i(x)$ tem suporte compacto $I_i \cup I_{i+1}$.

1.1.1 Interpolação

A interpolação é uma das técnicas de aproximação de funções. Dada uma função contínua f em $I = [x_0, x_1]$, definimos o **operador de interpolação linear** $\pi : C^0(I) \to P_1(I)$ por

$$\pi f(x) = f(x_0)\varphi_0(x) + f(x_1)\varphi_1(x). \tag{1.8}$$

Observamos que πf é igual a f nos nodos x_0 e x_1.

Exemplo 1.1.1. A Figura 1.2 ilustra a interpolação da função $f(x) = 3\,\mathrm{sen}(2\pi x)$ no espaço $P_1([1/4, 3/4])$. Neste caso

$$\pi f(x) = f\left(\frac{1}{4}\right)\frac{3/4 - x}{1/2} + f\left(\frac{3}{4}\right)\frac{x - 1/4}{1/2}. \tag{1.9}$$

Com o FENiCS, podemos computar a função interpolada πf com o seguinte código:

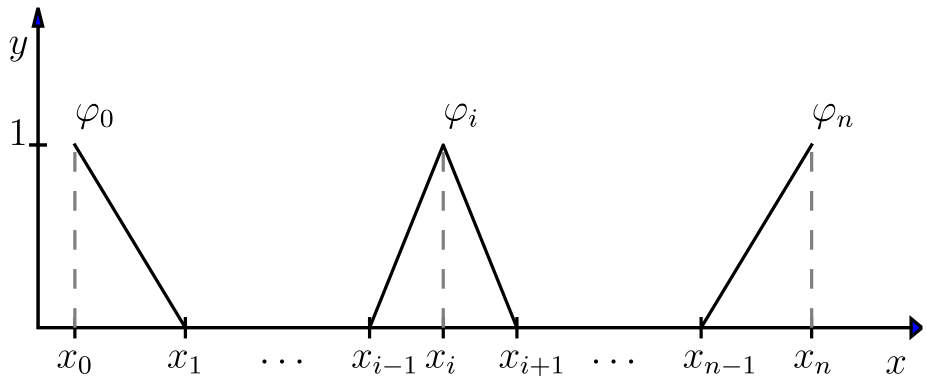

Figura 1.1: Base nodal para o espaço das funções lineares por parte.

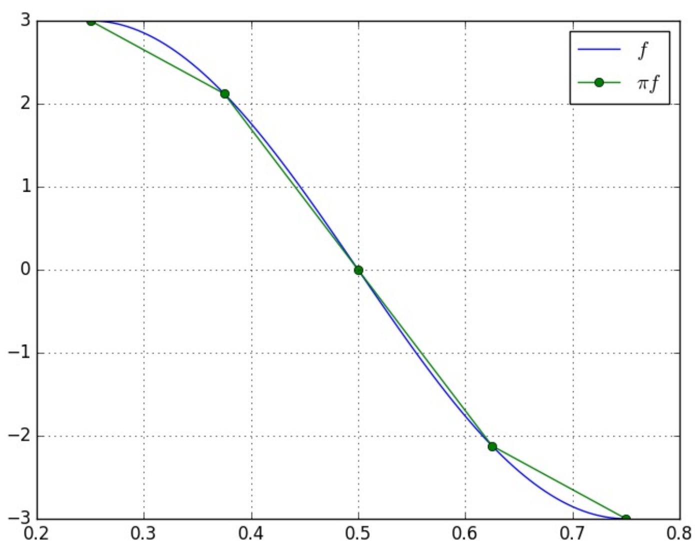

Figura 1.2: Interpolação linear de $f(x) = 3\,\mathrm{sen}(2\pi x)$ no espaço $P_1([1/4, 3/4])$.

```
from __future__ import print_function, division
from fenics import *
import numpy as np
```

```
import matplotlib.pyplot as plt

# malha
mesh = IntervalMesh(4,0.25,0.75)

# espaco
V = FunctionSpace(mesh, 'P', 1)

# funcao
f = Expression('3*sin(2*pi*x[0])',
                   degree=10)

# interpolacao
pif = interpolate(f,V)

# grafico
xx = IntervalMesh(100,0.25,0.75)
plot(f,mesh=xx,label="$f$")
plot(pif,mesh=mesh,
     marker='o',label="$\pi f$")
plt.legend(numpoints=1)
plt.grid('on')
plt.show()
```

Agora, vamos buscar medir o erro de interpolação, i.e. $f - \pi f$. Para tanto, podemos usar a norma L^2 definida por

$$\|v\|_{L^2(I)} = \left(\int_i v^2\,dx\right)^{1/2}. \tag{1.10}$$

Lembramos que valem as desigualdades triangular

$$\|v + w\|_{L^2(I)} \leq \|v\|_{L^2(I)} + \|w\|_{L^2(I)} \tag{1.11}$$

e a de Cauchy-Schwarz

$$\int_I vw\,dx \leq \|v\|_{L^2(I)}\|w\|_{L^2(I)}, \tag{1.12}$$

para qualquer funções $v,w \in L^2(I)$.

Proposição 1.1.1. (Erro da interpolação linear) O interpolador πf satisfaz as estimativas

$$\|f - \pi f\|_{L^2(I)} \leq Ch^2 \|f''\|_{L^2(I)}, \tag{1.13}$$

$$\|(f - \pi f)'\|_{L^2(I)} \leq Ch \|f''\|_{L^2(I)}, \tag{1.14}$$

onde C é uma constante e $h = x_1 - x_0$.

Demonstração. Denotemos o erro de interpolação por $e = f - \pi f$. Do teorema fundamental do cálculo, temos

$$e(y) = e(x_0) + \int_{x_0}^{y} e'(x)\, dx, \tag{1.15}$$

onde $e(x_0) = f(x_0) - \pi f(x_0) = 0$. Daí, usando a desigualdade de Cauchy-Schwarz (1.12), temos

$$\begin{aligned} e(y) &= \int_{x_0}^{y} e'\, dx && (1.16)\\ &\leq \int_{x_0}^{y} |e'|\, dx && (1.17)\\ &\leq \int_I 1 \cdot |e'|\, dx && (1.18)\\ &\leq \left(\int_I 1^2\, dx\right)^{1/2} \left(\int_I e'^2\, dx\right)^{1/2} && (1.19)\\ &= h^{1/2} \left(\int_I e'^2\, dx\right)^{1/2}, && (1.20) \end{aligned}$$

donde

$$e(y)^2 \leq h \int_I e'^2\, dx = h\|e'\|^2_{L^2(I)}. \tag{1.21}$$

Então, integrando em I obtemos

$$\|e\|^2_{L^2(I)} = \int_I e^2(y)\, dy \leq \int_I h\|e'\|^2_{L^2(I)}\, dy = h^2\|e'\|^2_{L^2(I)}, \tag{1.22}$$

ou seja,

$$\|e\|_{L^2(I)} \leq h\|e'\|_{L^2(I)}. \tag{1.23}$$

Agora, observando que $e(x_0) = e(x_1) = 0$, o teorema de Rolle garante a existência de um ponto $\tilde{x} \in I$ tal que $e'(\tilde{x}) = 0$, donde do teorema fundamental do cálculo e da desigualdade de Cauchy-Schwarz, segue

$$e'(y) = e'(\tilde{x}) + \int_{\tilde{x}}^{y} e''\,dx \tag{1.24}$$
$$= \int_{\tilde{x}}^{y} e''\,dx \tag{1.25}$$
$$\leq \int_I 1 \cdot |e''|\,dx \tag{1.26}$$
$$\leq h^{1/2} \left(\int_I e''^2 \right)^{1/2}. \tag{1.27}$$

Então, integrando em I, obtemos

$$\|e'\|^2_{L^2(I)} \leq h^2 \|e''\|^2_{L^2(I)}, \tag{1.28}$$

a qual, observando que $e'' = f''$, equivale a segunda estimativa procurada, i.e.

$$\|(f - \pi f)'\|_{L^2(I)} \leq Ch\|f''\|_{L^2(I)}. \tag{1.29}$$

Por fim, de (1.23) e de (1.23), obtemos a primeira estimativa desejada

$$\|f - \pi f\|_{L^2(I)} \leq Ch^2 \|f''\|_{L^2(I)}. \tag{1.30}$$

□

Exemplo 1.1.2. A Figura 1.3 mostra a evolução do erro na norma L^2 da interpolação de $f(x) = 3\,\text{sen}(2\pi x)$ no espaço $P_1([0, h)]$ para $h = 10^{-5}, 10^{-4}, \cdots, 10^{-1}$. Com o FENiCS, podemos computar os erros de interpolação com o seguinte código:

```
from __future__ import print_function, division
from fenics import *
import numpy as np
import matplotlib.pyplot as plt

# funcao
f = Expression('3*sin(2*pi*x[0])',
               degree=10)

n=5
for k in np.arange(1,n+1):
```

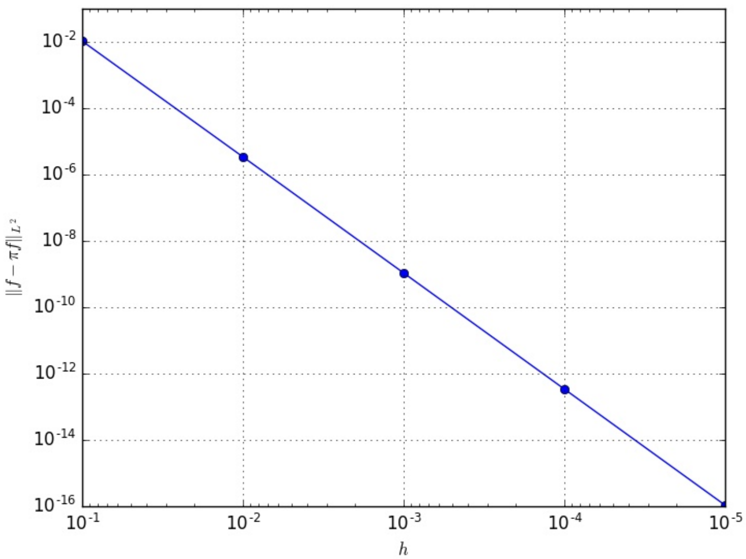

Figura 1.3: Erro de interpolação de $f(x) = 3\,\mathrm{sen}(2\pi x)$ no espaço $P_1([0,h])$.

```
h = 10**-k
mesh = IntervalMesh(1,0,h)
V = FunctionSpace(mesh, 'P', 1)
pif = interpolate(f,V)
e = errornorm(f,pif,'L2')
print("%d %1.0E %1.1E" % (k,h,e))
```

Vamos, agora, generalizar o resultado da Proposição 1.1.1 para a interpolação no espaço V_h das funções lineares por parte. Dada uma função contínua f em $I = [l_0, l_1]$, definimos o operador interpolador $\pi : C^0(I) \to V_h$ na malha $\mathcal{I}$ de I por

$$\pi f(x) = \sum_{i=0}^{n} f(x_i)\varphi_i(x). \tag{1.31}$$

Exemplo 1.1.3. A Figura 1.4 ilustra a interpolação da função $f(x) = 3\,\mathrm{sen}(2\pi x)$ no espaço V_h das funções lineares por partes em uma malha uniforme do intervalo $I = [1/4, 3/4]$ com $n = 4$ subintervalos (5 pontos).

Com o FENiCS, podemos computar a função interpolada πf com o seguinte código:

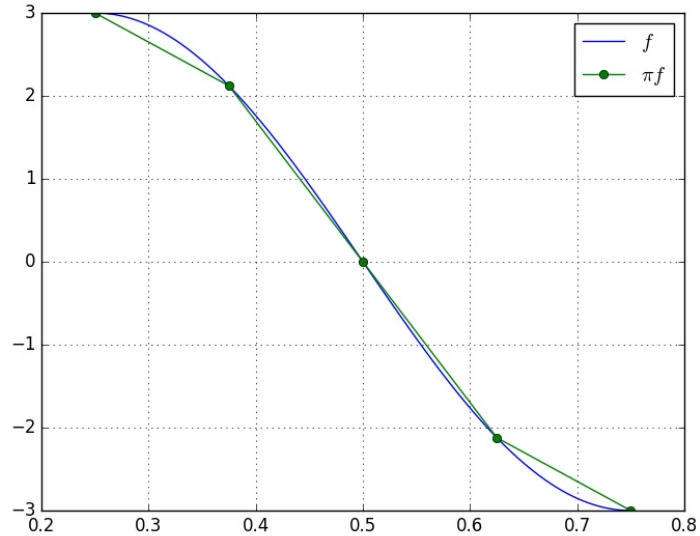

Figura 1.4: Interpolação linear de $f(x) = 3\,\text{sen}(2\pi x)$ no espaço V_h das funções lineares por partes sobre uma malha com 5 pontos.

```
from __future__ import print_function, division
from fenics import *
import numpy as np
import matplotlib.pyplot as plt

# malha
mesh = IntervalMesh(4,0.25,0.75)

# espaco
V = FunctionSpace(mesh, 'P', 1)

# funcao
f = Expression('3*sin(2*pi*x[0])',
                   degree=10)

# interpolacao
pif = interpolate(f,V)
```

```
# grafico
xx = IntervalMesh(100,0.25,0.75)
plot(f,mesh=xx,label="$f$")
plot(pif,mesh=mesh,
     marker='o',label="$\pi f$")
plt.legend(numpoints=1)
plt.grid('on')
plt.show()
```

O seguinte resultado fornece uma estimativa do erro de interpolação em relação ao tamanho h_i de cada elemento da malha.

Proposição 1.1.2. *O interpolador πf satisfaz as estimativas*

$$\|f - \pi f\|_{L^2(I)}^2 \leq C \sum_{i=1}^{n} h_i^4 \|f''\|_{L^2(I)}^2, \tag{1.32}$$

$$\|(f - \pi f)'\|_{L^2(I)}^2 \leq C \sum_{i=1}^{n} h_i^2 \|f''\|_{L^2(I)}^2. \tag{1.33}$$

$$\tag{1.34}$$

Demonstração. Ambas desigualdades seguem da desigualdade triangular e da Proposição 1.1.1. Por exemplo, para a primeira desigualdade, temos

$$\|f - \pi f\|_{L^2(I)}^2 \leq \sum_{i=1}^{n} \|f - \pi f\|_{L^2(I_i)}^2 \tag{1.35}$$

$$\leq \sum_{i=1}^{n} C h_i^4 \|f''\|_{L^2(I_i)}^2. \tag{1.36}$$

□

1.1.2 Projeção L^2

Dada uma função $f \in L^2(I)$, definimos o **operador de projeção** L^2 $P_h : L^2(I) \to V_h$ por

$$\int_I (f - P_h f) v \, dx = 0, \quad \forall v \in V_h. \tag{1.37}$$

Como V_h é um espaço de dimensão finita, a condição (1.38) é equivalente a

$$\int_I (f - P_h f) \varphi_i \, dx = 0, \quad i = 0, 1, \cdots, n, \tag{1.38}$$

onde φ_i é a i-ésima função base de V_h. Além disso, como $P_h f \in V_h$, temos

$$P_h f = \sum_{j=0}^{n} \xi_j \varphi_j, \tag{1.39}$$

onde ξ_j, $j = 0, 1, \dots, n$, são $n+1$ incógnitas a determinar. Logo,

$$\begin{align}
\int_I (f - P_h f)\varphi_i \, dx = 0 &\Leftrightarrow \int_I f\varphi_i \, dx = \int_I P_h f \varphi_i \, dx \tag{1.40}\\
&\Leftrightarrow \int_I f\varphi_i \, dx = \int_I \left(\sum_{j=0}^{n} \xi_j \varphi_j\right) \varphi_i \, dx \tag{1.41}\\
&\Leftrightarrow \sum_{j=0}^{n} \xi_j \int_I \varphi_j \varphi_i \, dx = \int_I f\varphi_i \, dx, \tag{1.42}
\end{align}$$

para $i = 0, 1, \dots, n$.
Observemos, agora, que (1.42) consiste em um sistema de $n+1$ equações lineares para as $n+1$ incógnitas ξ_j, $j = 0, 1, \dots, n$. Este, por sua vez, pode ser escrito na seguinte forma matricial

$$M\xi = b, \tag{1.43}$$

onde $M = [m_{i,j}]_{i,j=0}^{n+1}$ é chamada de matriz de massa

$$m_{i,j} = \int_I \varphi_j \varphi_i \, dx \tag{1.44}$$

e $b = (b_0, b_1, \dots, b_n)$ é chamado de vetor de carregamento

$$b_i = \int_I f\varphi_i \, dx. \tag{1.45}$$

Ou seja, a projeção L^2 de f no espaço V_h é

$$P_h f = \sum_{j=0}^{n} \xi_j \varphi_j, \tag{1.46}$$

onde $\xi = (\xi_0, \xi_1, \dots, \xi_n)$ é solução do sistema (1.43).

Exemplo 1.1.4. A Figura 1.5 ilustra a projeção L^2 da função $f(x) = 3\,\mathrm{sen}(2\pi x)$ no espaço V_h das funções lineares por partes em uma malha uniforme do intervalo $I = [1/4, 3/4]$ com $n = 4$ subintervalos (5 pontos).
Com o FENiCS, podemos computar $P_h f$ com o seguinte código:

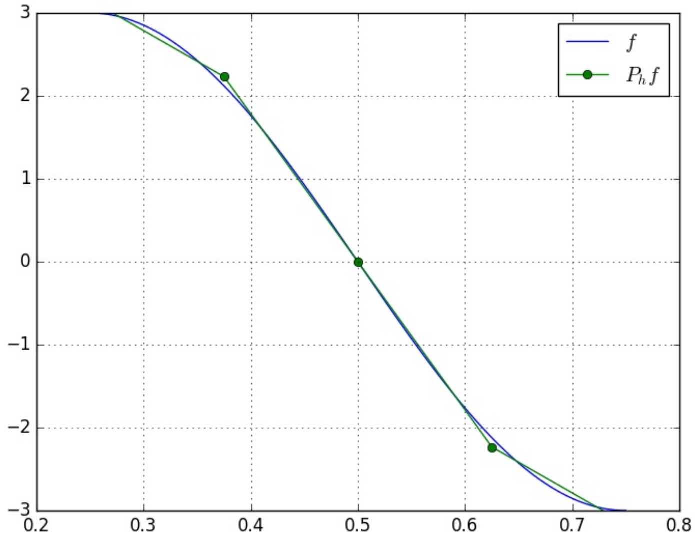

Figura 1.5: Projeção L^2 de $f(x) = 3\,\mathrm{sen}(2\pi x)$ no espaço V_h das funções lineares por partes sobre uma malha com 5 pontos.

```
from __future__ import print_function, division
from fenics import *
import numpy as np
import matplotlib.pyplot as plt

# malha
mesh = IntervalMesh(4,0.25,0.75)

# espaco
V = FunctionSpace(mesh, 'P', 1)

# funcao
f = Expression('3*sin(2*pi*x[0])',
                   degree=10)

# projecao
Phf = project(f, V)
```

```
# grafico
xx = IntervalMesh(100,0.25,0.75)
plot(f,mesh=xx,label="$f$")
plot(Phf,mesh=mesh,
     marker='o',label="$P_h f$")
plt.legend(numpoints=1)
plt.grid('on')
plt.show()
```

O próximo teorema mostra que P_hf é a função que melhor aproxima f dentre todas as funções do espaço V_h.

Teorema 1.1.1. *(*A melhor aproximação.) A projeção L^2 satisfaz

$$\|f - P_hf\|_{L^2(I)} \leq \|f - v\|_{L^2(I)}, \quad \forall v \in V_h. \tag{1.47}$$

Demonstração. Dado $v \in V_h$, temos

$$\begin{aligned}
\|f - P_hf\|_{L^2(I)}^2 &= \int_I |f - P_hf|^2\,dx &(1.48)\\
&= \int_I (f - P_hf)(f - v + v - P_hf)\,dx &(1.49)\\
&= \int_I (f - P_hf)(f - v)\,dx + \int_I (f - P_hf)(v - P_hf)\,dx &(1.50)\\
&= \int_I (f - P_hf)(f - v)\,dx &(1.51)\\
&\leq \int_I \|f - P_hf\|_{L^2(I)}\|f - v\|_{L^2(I)}, &(1.52)
\end{aligned}$$

donde segue o resultado. □

O próximo teorema fornece uma estimativa *a-priori* do erro $\|f - P_hf\|_{L^2(I)}$ em relação ao tamanho da malha.

Teorema 1.1.2. *A projeção L^2 satisfaz*

$$\|f - P_hf\|_{L^2(I)} \leq C\sum_{i=1}^{n} h_i^4\|f''\|_{L^2(I_i)}^2. \tag{1.53}$$

Demonstração. Tomando a interpolação $\pi f \in V_h$, temos do Teorema da melhor aproximação (Teorema **??**) e da estimativa do erro de interpolação (Proposição 1.1.2) que

$$\begin{align}
\|f - P_h f\|_{L^2(I)}^2 &\leq \|f - \pi f\|_{L^2(I)}^2 \tag{1.54}\\
&\leq C \sum_{i=1}^{n} h_i^4 \|f''\|_{L^2(I_i)}^2. \tag{1.55}
\end{align}$$

□

Em construção ...

1.1.3 Exercícios

E 1.1.1. Faça um código para verificar a segunda estimativa da Proposição 1.1.1 no caso da interpolação da função $f(x) = 3\,\mathrm{sen}(2\pi x)$ no espaço P_1 das funções lineares.

E 1.1.2. Faça um código para verificar as estimativas da Proposição 1.1.2 no caso da interpolação da função $f(x) = 3\,\mathrm{sen}(2\pi x)$ no espaço V_h das funções lineares por partes.

Em construção ...

Resposta dos Exercícios

Referências Bibliográficas

[1] Hans Petter Langtangen and Anders Logg. *Solving PDEs in Python*. Springer, 2017.

[2] M.G. Larson and F. Bengson. *The Finite Element Method: Theory, Implementation, and Applications*. Springer, 2013.

Índice Remissivo