Polinômios de Bernstein para o método de Galerkin

Lucas B. Andrade
Orientadores:
Sonia Maria Gomes
Philippe Remy Devloo

5 de Fevereiro de 2018

1 Introdução

O método de Galerkin foi desenvolvido para a solução computacional de problemas de valores de contorno com um problema modelo.
Encontrar u $\in C^2[0,1]$ que satifaça

$$-(\alpha(x)u'(x))' + \sigma(x)u(x) = f(x) \qquad x \in (0,1) \tag{1}$$

$$u(0) = u(1) = 0.$$

Faremos uma breve introdução ao problema e seguiremos com a formulação variacional que dará origem ao método.

2 Formulação Variacional

Para a formulação variacional definem-se dois espaços vetoriais.

$$H^1 := \{v : [0,1] \to \mathbb{R};\ v \in C([0,1]) \text{ e } v' \in C^0([0,1])\}.$$

$$H_0^1 := \{v \in H^1;\ v(0) = v(1) = 0\}.$$

Em que $C^0([0,1])$ é o espaço de funções contínuas por partes em $[0,1]$.

Inicialmente multiplicamos a equação (1) por uma função $v \in H_0^1$:

$$-(\alpha(x)u'(x))'v(x) + \sigma(x)u(x)v(x) = f(x)v(x).$$

e, em seguida, integramos a igualdade no domínio [0,1]

$$\int_0^1 -(\alpha(x)u'(x))'v(x)dx + \int_0^1 \sigma(x)u(x)v(x)dx = \int_0^1 f(x)v(x).dx$$

Integrando

$$\int_0^1 -(\alpha(x)u'(x))'v(x)dx.$$

por partes obtém-se

$$\int_0^1 \alpha(x)u'(x)v'(x)dx + [\alpha(x)u'(x)v(x)]_0^1.$$

Como $v \in H_0^1$ o segundo termo acima se anula, e obtemos a formulação variacional do problema: encontrar $u \in H_0^1$ tal que

$$\int_0^1 \alpha(x)u'(x)v'(x)dx + \int_0^1 \sigma(x)u(x)v(x)dx = \int_0^1 f(x)v(x)dx, \qquad \forall\ v \in H_0^1. \tag{2}$$

Podemos simplificar a escrita, e carregar menos notação definindo

$$A(u,v) = \int_0^1 \Big(\alpha(x)u'(x)v'(x) + \sigma(x)u(x)v(x)\Big)dx, \text{ e.} \tag{3}$$

$$b(v) = \int_0^1 f(x)v(x)dx. \tag{4}$$

2.1 Discretização

Vamos considerar agora um espaço $V_0^h \subset H_0^1$ com $\dim V_0^h = n < \infty$ desse modo teremos uma base $\{\varphi_i\}_{i=1,\dots,n}$ para V_0^h, ou seja, se $v \in V_0^h$ podemos escrever $v(x) = \sum_{i=1}^{n} c_i\varphi_i(x)$, o que nos permitirá trabalhar computacionalmente. Temos agora, uma discretização para o problema variacional, denominada Método de Galerkin.

Devemos agora encontrar $u_h \in V_0^h$ que satisfaça

$$A(u_h, v) = b(v), \qquad \forall\ v \in V_0^h. \tag{5}$$

Particularmente, como devemos ter (5) para todo elemento de V_0^h, podemos, simplesmente, verificar se essa equação vale para todo elemento da base de V_0^h. ou seja

$$A(u_h, \varphi_j) = b(\varphi_j) \qquad j = 1, \dots, n.$$

E como $u_h \in V_0^h$ podemos escrever $u_h(x) = \sum_{i=1}^{n} c_i\varphi_i(x)$ e ficamos com

$$\sum_{i=1}^{n} c_i A(\varphi_i, \varphi_j) = b(\varphi_j), \qquad j = 1, \dots, n \tag{6}$$

Note que o problema (6) é um sistema linear $n \times n$, que podemos escrever como sendo $\mathbf{Ac} = \mathbf{b}$, em que

$$\mathbf{A} = \{d_{i,j}\},\ d_{i,j} = A(\varphi_i, \varphi_j).$$
$$\mathbf{c} = [c_1, c_2, \dots, c_n]^t.$$
$$\mathbf{b} = [b(\varphi_1), b(\varphi_2), \dots, b(\varphi_n)]^t.$$

Podemos também escrever $\mathbf{A}$ como sendo a soma de duas matrizes $\mathbf{A} = \mathbf{M} + \mathbf{S}$, em que

$$\mathbf{M} = \{m_{i,j}\},\ m_{i,j} = \int_a^b \sigma(x)\varphi_i(x)\varphi_j(x)dx.$$

é a matriz de massa (**M**ass) e

$$\mathbf{S} = s_{i,j},\ s_{i,j} = \int_a^b \alpha(x)\varphi_i'(x)\varphi_j'(x)dx$$

é a matriz de rigidez (**S**tiffness).

3 Espaços de Polinômios por Partes

Antes de dar continuidade ao método, introduziremos espaços de polinômios por partes, que são fáceis de trabalhar computacionalmente, e que utilizaremos como espaço para aproximação da solução.

Considere inicialmente a partição de $[a,b]$: $\Pi_N := \{\pi_i \in (x_i, x_{i+1});\ 0 < i < N-1\}$, em que $a = x_0 < x_1 < x_2 < \cdots < x_{N-1} < x_N = b$. Com base nessa partição, definimos o espaço:

$$L_n(\Pi_N) := \{p\ ;\ p|_{\pi_i} \in \mathbb{P}^n\}.$$

em que $\mathbb{P}^n$ representa o espaço dos polinômios de grau menor ou igual a n. Note que $\dim L_n(\Pi_N) = nN + 1$.

3.1 Lagrange Linear por Partes (Splines Lineares)

Considere o caso $n = 1$. Como $\dim L_1(\Pi_N) = N + 1$ podemos definir uma base finita para este espaço, ou seja: $L_1(\Pi_N) := span\{\varphi_i\}_{i=0,1,2,\ldots,N}$ em que

$$\varphi_i := \begin{cases} \dfrac{(x - x_{i-1})}{(x_i - x_{i-1})} & , \text{ se } x \in [x_{i-1}, x_i] \\ \dfrac{(x_{i+1} - x}{(x_{i+1} - x_i} & , \text{ se } x \in [x_i, xi + 1] \\ 0 & , \text{ caso contrário} \end{cases} \qquad \text{i=1,2,...,N-1} \tag{7}$$

$$\varphi_0 = \begin{cases} \dfrac{(x_0 - x)}{(x_1 - x_0)} & , \text{ se } x \in [x_0, x_1] \\ 0 & , \text{ caso contrário} \end{cases} \tag{8}$$

$$\varphi_n = \begin{cases} \dfrac{(x - x_N)}{(x_N - x_{N-1})} & , \text{ se } x \in [x_{N-1}, x_N] \\ 0 & , \text{ caso contrário} \end{cases}. \tag{9}$$

Para uma visualização gráfica destas funções base ver [3].

3.2 Polinômios de Bernstein

Além disso introduziremos uma base para o espaço $\mathbb{P}^n$ e chamaremos o espaço gerado por elas de $\mathbb{B}^n = span\{b_{i,n}\}_{i=0,1,2,\ldots,n}$, em que

$$b_{i,n} = \frac{1}{(b-a)^n}\binom{n}{i}(x-a)^i(b-x)^{n-i}. \tag{10}$$

$\binom{n}{i} = \frac{n!}{i!(n-i)!}$,
Esta base tem propridades interessantes para este tipo de trabalho, como

- P_1) Multiplicação:
  $b_{i,n}(x)b_{j,m}(x) = \dfrac{\binom{n}{i}\binom{m}{j}}{\binom{n+m}{i+j}} b_{i+j,n+m}$
- P_2) Derivada:
  $\dfrac{db_{i,n}(x)}{dx} = \frac{n}{b-a}\big(b_{i-1,n-1}(x) - b_{i,n-1}(x)\big)$
- P_3) Integral:
  $\int_a^b b_{i,n}(x)dx = \dfrac{b-a}{n+1}$
- P_4) Recorrência:
  1. $b_{0,n}(x) = (b-x)^n$, $b_{i,n}(x) = \frac{x-a}{b-x}\frac{n-i}{i+1}\frac{b_{i-1,n}}{(b-a)}$ ou
  2. $b_{n,n}(x) = (x-a)^n$, $b_{i,n}(x) = \frac{b-x}{x-a}\frac{i+1}{n-i}b_{i+1,n}(b-a)$

Um estudo melhor sobre esta base de polinômios em algumas aplicações numéricas encontra-se em [2].

3.2.1 Polinômios de Bernstein por Partes

Podemos também construir uma base do espaço $L_n(\Pi_N)$ a partir dos polinômios de Bernstein, para isso iremos primeiramente definir bases locais

$$b_{i,n}^k(x) := \begin{cases} b_{i,n}(x) & , \text{ se } x \in [x_k, x_{k+1}] \\ 0 & , \text{ c.c} \end{cases}.$$

Em que $b_{i,n}$ é definido de acordo com o intervalo que ele representa $b_{i,n}^k = \frac{1}{(x_{k+1}-x_k)^n}\binom{n}{i}(x - x_k)^i(x_{k+1} - x)^{n-i}$. Para as bases globais temos para $0 \leq k \leq N-1$

$$\phi^k(x) = b^k_{0,n}(x) + b^{k-1}_{n,n}(x)$$
$$\varphi^k_i = b^k_{i,n}(x), \quad 0 < i < n$$

Em que os termos que possuem índice k fora do domínio de variação são simplesmente ignorados (e.g. $b^N_{0,n} = b^{-1}_{n,n} = 0$)
Verifica-se que $L_n(\Pi_N) = \text{span}\{\phi^k_n(x), \varphi^k_{i,n}, 0 \leq k \leq N-1, 0 < i < n\}$

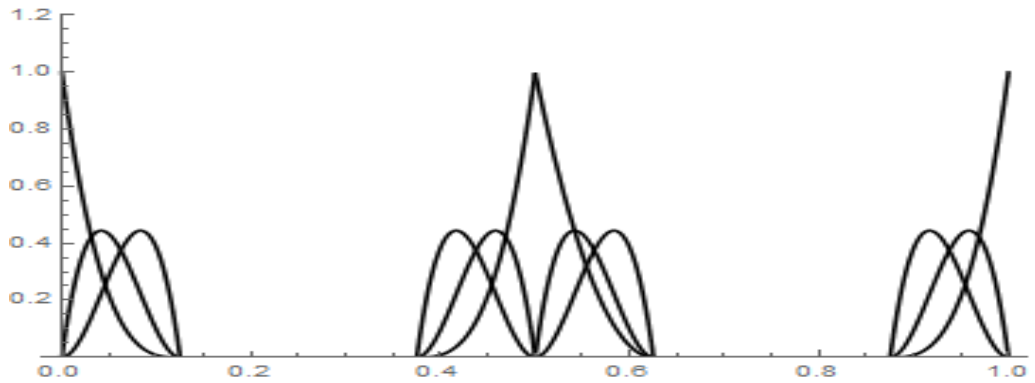


Figura 1: Gráfico com as representações das bases globais de Bernstein para o caso $n = 3$

4 O Método de Galerkin

Vamos agora encontrar uma solução para o problema variacional discreto (6). Para isso precisamos definir o nosso espaço V^h_0. Faremos isso utilizando o espaço de Splines Lineares e posteriormente utilizando os polinômios de Bernstein.

4.1 Splines Lineares

Primeiramente para a implementação do método de Galerkin utilizaremos o espaço $L^0_1(\Pi_N) := \{p \in L_1(\Pi_N)\ ;\ p(a) = p(b) = 0\}$. Note que $L_1(\Pi_N) \subset V$, então podemos ter $V^h_0 = L^0_1(\Pi_N)$.

Para os elementos da base de $L^0_1(\Pi_N)$ basta tomar $\{\varphi_i\}_{i=1,2,\ldots,N-1}$, já que estes são os únicos elementos da base de $L_1(\Pi_N)$ que não se anulam nas extremidades. Utilizando pontos de malha igualmente espaçados ($h = (x_{i+1} - x_i),\ i = 0, 1, 2, \ldots, N-1$) vamos então calcular os coeficientes da equação (6), considerando coeficientes constantes $\alpha, \sigma \in \mathbb{R}^+$.
Para os elementos que distam mais de um ponto na malha, uma das bases se anula completamente, portanto teremos uma matriz tridiagonal, logo basta apenas calcular os fatores da diagonal principal, e da diagonal superior, já que sabemos que esta matriz deve ser simétrica. Para a matriz de massa teremos:

$$\int_{x_{i-1}}^{x_{i+1}} (\varphi_i(x))^2 dx = \frac{2h}{3}$$

$$\int_{x_{i-2}}^{x_{i+1}} \varphi_{i-1}(x)\varphi_i(x) = \int_{x_{i-1}}^{x_{i+2}} \varphi_i(x)\varphi_{i+1}(x) = \int_{x_i}^{x_{i+1}} \varphi_i(x)\varphi_{i+1}(x) = \frac{h}{6}$$

E para a matriz de rigidez:

$$\int_{x_{i-1}}^{x_{i+1}} (\varphi'_i(x))^2 dx = \frac{2}{h}$$

$$\int_{x_{i-2}}^{x_{i+1}} \varphi'_{i-1}(x)\varphi'_i(x) = \int_{x_{i-1}}^{x_{i+2}} \varphi'_i(x)\varphi'_{i+1}(x) = \int_{x_i}^{x_{i+1}} \varphi'_i(x)\varphi'_{i+1}(x) = -\frac{1}{h}$$

Então ficamos com as matrizes:

$$M = \sigma h \begin{bmatrix} \frac{2}{3} & \frac{1}{6} & & 0 \\ \frac{1}{6} & \ddots & \ddots & \\ & \ddots & \ddots & \frac{1}{6} \\ 0 & & \frac{1}{6} & \frac{2}{3} \end{bmatrix} \tag{11}$$

$$S = \frac{\alpha}{h} \begin{bmatrix} 2 & -1 & & 0 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{bmatrix} \tag{12}$$

Assim podemos facilmente construir o sistema linear que dará a solução de Galerkin, a qual deve convergir para a solução do problema variacional, veremos resultados sobre a convergência na Seção 5.

4.2 Polinômios de Bernstein

Seja $V_0^h = \{\varphi \in L_n(\Pi_N);\ \varphi(a) = \varphi(b) = 0\}$. Analogamente ao exemplo anterior, retiramos os dois elementos extremos da base para satisfazer as condições de contorno.

Vamos então calcular os elementos das matrizes da equação (6). Assume-se que $\alpha, \sigma \in \mathbb{R}^+$ são constantes. Para a matriz de massa pelas propriedades P_1 e P_2, temos:

$$\int_a^b b_{i,n}(x) b_{j,n}(x) dx = \frac{\binom{n}{i}\binom{n}{j}}{\binom{2n}{i+j}(2n+1)}$$

Para a matriz de rigidez, utilizando as propriedades P_1, P_2 e P_3, temos

$$\int_a^b b'_{i,n}(x) b'_{j,n}(x) dx = \frac{n^2}{(2n-1)(b-a)} \left(\frac{\binom{n-1}{i-1}\binom{n-1}{j-1}}{\binom{2n-1}{i+j-2}} + \frac{\binom{n-1}{i}\binom{n-1}{j}}{\binom{2n-2}{i+j}} - \frac{\binom{n-1}{i-1}\binom{n-1}{j} + \binom{n-1}{i}\binom{n-1}{j-1}}{\binom{2n-2}{i+j-1}} \right)$$

Em que os termos que posuem coeficientes binomiais indeterminados (e.g. $\binom{n}{-1}$, $\binom{n}{n+1}$) devem ser ignorados da soma.

Para o caso de coeficientes variáveis $\alpha, \sigma \in C([0,1])$, e para calcular o vetor de carga, podemos utilizar o método de integração de Gauss-Jacobi, com função peso $(1-x)^l(x+1)^q$. Assim temos:

$$\int_{-1}^{1} (1-x)^l (x+1)^q f(x) dx \approx \sum_{i=1}^{k} \psi_i f(\xi_i) \tag{13}$$

Em que ξ_i e ψ_i são os pontos e pesos de integração respectivamente. Nota-se que

$$\int_{-1}^{1} (1-x)^{n-q} (x+1)^q f(x) dx = \frac{1}{\binom{n}{q}} \int_{-1}^{1} b_{q,n}(x) f(x) dx \tag{14}$$

Em que $b_{i,n}$ é definido no intervalo de integração. Para calcular os elementos das matrizes e o vetor de carga definimos os **momentos** $\mu_i(f)$ de uma função:

$$\mu_i(f) = \int_a^b b_{i,n}(x) f(x) dx \tag{15}$$

Então implementamos os seguintes algoritmos, com base nos trabalhos em [1]:

Algorithm 1 Moment($F,\ n$)

Entrada: F: vetor com os valores da função nos pontos de quadratura n: grau do polinômio de Berntein.
Saída: Vetor com os momentos de f para cada índice da base

$q = \text{Length}(F)$ // Número de pontos de integração
$F_{out} = $ Array com $n+1$ entradas;
For $i = 1$ **to** q **do**
  $\xi = \xi_i^{(0,0)}$; // $\xi_i^{(0,0)}$ é a abscissa para quadratura com q pontos
  $\omega = \omega_i^{(0,0)}$;
  $s = 1 - \xi$;
  $r = \xi/s$; // coeficiente de recorrência de $P4$-1 (Seção 3.2)
  $w = \omega * s^n$;
  For $\alpha = 0$ **to** n **do**
    // Aqui temos $w == \omega_i^{(0,0)} b_{\alpha,n}(\xi_i^{(0,0)})$
    $F_{out}[\alpha] += w * F[i]$;
    $w* = r * \frac{n-\alpha}{1+\alpha}$; // Trata do do passo de recorrência
  end For
end For
return F_{out};

Este algoritmo utiliza da propriedade $P4$-1 (Seção 3.2), para calcular os momentos $\mu_i(f)$. Temos inicialmente no algoritmo $w == \omega_i^{(0,0)} b_{0,n}(\xi_i^{(0,0)})$ de onde partimos para o passo de recorrência de $P4$-1 com $\frac{\xi}{1-\xi} = r$. Cada coeficiente i da quadratura é somado na posição α do vetor F_{out} que terá em sua entrada α o valor de $\mu_\alpha(f)$.

Algorithm 2 MomentB($f,\ n$), uma versão alternativa ao algortimo acima, que utiliza a relação da equação (14)

Entrada: f: função n: grau do polinômio de Berntein.
Saída: Vetor com os momentos de f para cada índice da base

$q = n + 1$;
$F_{out} = \text{ConstantArray}(0, n+1)$;
For $\alpha = 0$ **to** n **do**
  For $i = 1$ **to** q **do**
    $\xi = \xi_i^{(n-\alpha,\alpha)}$; // $\xi_i^{(n-\alpha,\alpha)}$ é a abscissa para quadratura com q pontos
    $\omega = \omega_i^{(n-\alpha,\alpha)}$;
    $F_{out}[\alpha] += \omega * f(\xi)$;
  end For
  $F_{out} /= \binom{n}{i}$;
end For
return F_{out};

Ambos algoritmos de cálculo dos momentos $\mu_i(f)$ tem complexidade $\mathcal{O}(nq)$. O primeiro apresenta instabilidade para as últimas entradas do vetor de momentos, podendo ser feita uma adaptação que utiliza a segunda forma recursiva da propriedade P4 junto com a primeira (já utilizada para o algoritmo) para calcular esses valores.

Algorithm 3 MatMassa($\alpha,\ n$)

Entrada: α: função da matriz de massa, n: grau do polinômio de Bernstein.
Saída: Matriz de massa do problema considerando a função $\alpha(x)$.

```
// F_out : Matriz n + 1 X n + 1
F = Array com α(ξ_i^(0,0))   i = 1, ..., q
F = Moment(F, 2n);
For i = 0 to n do
  For j = 0 = 0 to n do
    F_out[i][j] = C(i+j, i) / C(2n, n) * C(2n-i-j, n-i) * F[i + j];
  end For
end For
return F_out;
```

Algorithm 4 MatRigidez($\sigma,\ n$)

Entrada: σ: função da matriz de rigidez, n: grau do polinômio de Bernstein.
Saída: Matriz de rigidez do problema considerando a função $\sigma(x)$.

```
// F_out : Matriz n + 1 X n + 1
// μ: vetor tridimensional de tamanho (2, 2, 2n − 1)
∇λ = {1, −1};
F = Array com σ(ξ_i^(0,0))   i = 1, ..., q;
F = Moment(F, 2n − 2);
For k = 0 to 1 do
  For l = 0 to 1 do
    For i = 0 to 2n − 2 do
      μ[k][l][i] = ∇λ[k]∇λ[l] * F[i];
    end For
  end For
end For
For i = 0 to n − 1 do
  For j = 0 to n − 1 do
    For k = 0 to 1 do
      For l = 0 to 1 do
        F_out[i − (k − 1)][j − (l − 1)] = C(i+j, i) / C(2n, n) * C(2n-i-j, n-i) * F[i + j];
      end For
    end For
  end For
end For
return F_out;
```

Ambos algoritmos de cálculo da matriz de massa e de rigidez tem complexidade $\mathcal{O}(n^2)$. Como o esperado em [1].

5 Estimativa de Erro

Primeiramente utilizaremos os seguintes resultados demonstrados em [3]:

Proposição 1 *A forma $A(\cdot,\cdot)$, definida em (3), é uma forma bilinear coerciva, ou seja, se $u \in V_0$, existe $c_1 > 0$ tal que*

$$A(u,u) \geq c_1 \|u\|^2_{H^1}.$$

Em que

$$\|f\|_{H^1} = \sqrt{\int_a^b (f(x)^2 + f'(x)^2)dx}. \tag{16}$$

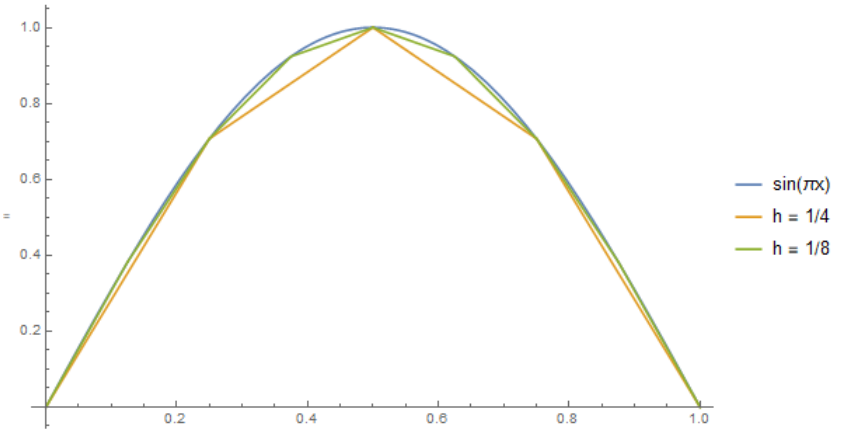


Figura 2: Gráfico de $u(x)$ e as aproximações no caso linear (Lagrange) para dois valores de h.

Proposição 2 *A forma $b(\cdot)$, definida em (4), é uma forma linear contínua, ou seja, se $u \in V_0$, existe $c_2 > 0$ tal que:*

$$b(u) \leq \|u\|^2_{H^1}.$$

Este resultado nos permite aplicar o teorema de Lax-Milgram, que implica em uma solução única para o problema variacional discreto (5). O Lema de Céa garante que as soluções u do problema variacional exato (2) u_h do problema variacional discreto (5) satisfazem a propriedade de melhor aproximação:

$$\|u - u_h\|_{H^1} = \inf \|u - v\|_{H^1}, \forall v \in V_h.$$

Desta forma, se

$$E_h = \|u - u_h\|_{H^1}.$$

e $\mathcal{P}u$ o interpolante de u (em $L_n(\Pi_N)$), então, como $L_n(\Pi_N) \subset V_h$, pelo lema de Céa

$$E_h \leq \|u - \mathcal{P}u\|_{H^1}.$$

Pela fórmula de erro de interpolação do Teorema 6.8 de [4], existe $C \in \mathbb{R}^+$ tal que

$$E_h \leq Ch^n \|\frac{d^{n+1}u}{dx}\|_{L^2}. \tag{17}$$

E para a norma L^2, utilizando o método de Aubin-Nitsche, mencionado em [4], chegamos em

$$\|u - u_h\|_{L^2} \leq Ch^{n+1} \|\frac{d^{n+1}u}{dx}\|_{L^2} \tag{18}$$

6 Alguns Resultados

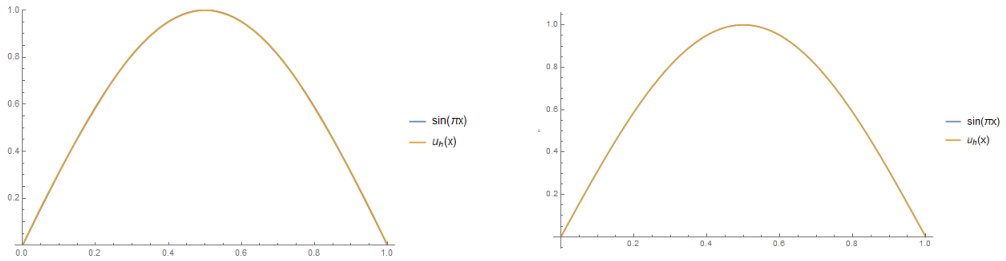
Foi implementado o método utilizando as bases de Lagrange (Splines Lineares) e as bases dos polinômios de Bernstein para grau $n = 2$ e $n = 3$. Todos foram implementados utilizando o Mathematica 11.0 para o problema:

$$\begin{cases} u''(x) = \pi^2 \sin(\pi x) \\ u(0) = u(1) = 0 \end{cases} \tag{19}$$

Com solução exata $u(x) = \sin(\pi x)$.

Observe nas figuras de 2 à 5 os resultados obtidos da implementação para aproximar a solução da equação (19) acima.

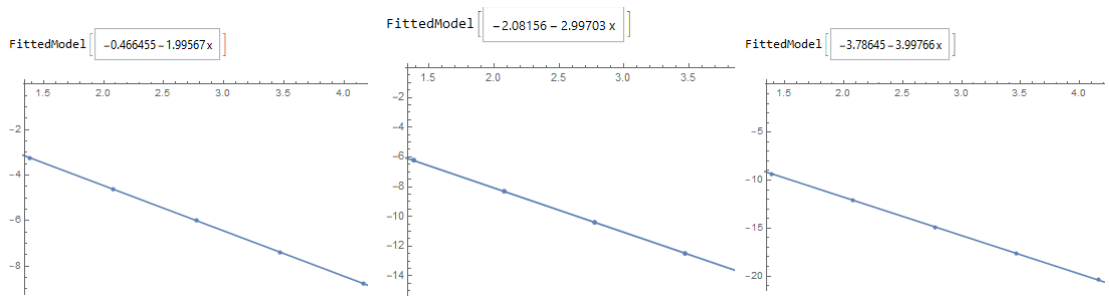
Observe na figura 5 e na tabela 1 que os resultados obtidos correspondem com a teoria em (17).



(a) Bernstein de grau 2

(b) Bernstein de grau 3

Figura 3: Gráficos das aproximações de Galerkin utlizando as bases Bernstein, para $h = 1/4$. A solução aproximada acaba cobrindo a solução exata no gráfico.

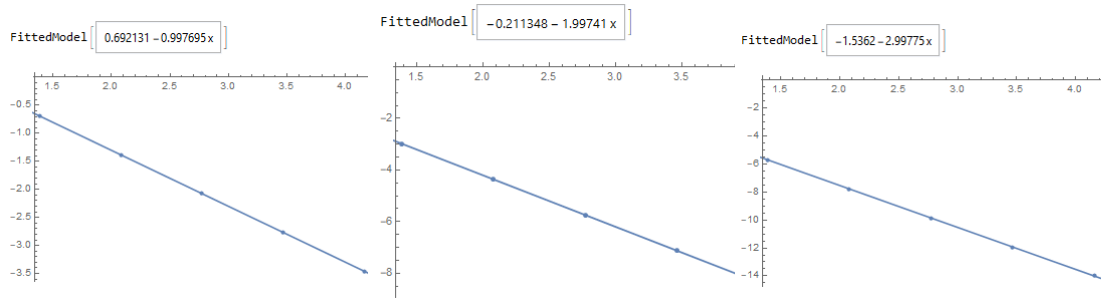


(a) Lagrange Linear.

(b) Bernstein de grau 2.

(c) Bernstein de grau 3.

Figura 4: Gráficos de $\log(\|u_h - u\|_{L^2})$ em função de $\log(h^{-1})$ com as respectivas regressões lineares para os casos implementados ($\|f\|_{L^2} = \left(\int_a^b f(x)^2 dx\right)^{\frac{1}{2}}$).



(a) Lagrange Linear.

(b) Bernstein de grau 2.

(c) Bernstein de grau 3.

Figura 5: Gráficos de $\log(\|u_h - u\|_V)$ em função de $\log(h^{-1})$ com as respectivas regressões lineares para os casos implementados.

Tabela 1: Tabela de convergência para cada método, nas normas do máximo L^2 e V

Método/Norma	$\|\cdot\|_{L^2}$	$\|\cdot\|_\infty$	$\|\cdot\|_V$
Lagrange	2	1.99	1
Bernstein (2)	3	2.98	2
Bernstein (3)	4	3.98	3

Referências

[1] Mark Ainsworth, Gaelle Andriamaro, and Oleg Davydov. Bernstein-bézier finite elements of arbitrary order and optimal assmbly procedures. *SIAM J. Sci. Comput.*, 33(6):3087–3109, 2011.

[2] Lucas B. Andrade. Estudo de aproximação por polinômios de bernstein. 2017.

[3] Petronio Pulino; Marcio R. Fernandes. Resolução de equações diferenciais via método dos elementos finitos. 2002.

[4] J. T. Oden; J. N. Reddy. *And Introduction to the mathematical theory of finite elements.* 1976.