Gonzalo Bonilla Moreno (5850177)
David Plazas (5695767)
Programme: MSc in Applied Mathematics

**WI4205 Applied Finite Elements**
Assignment 1.3

---

# 1 Verification of the 1D Finite Element Method

A. Computing the error in the $\|\cdot\|_0$ and $\|\cdot\|_1$ can be done by computing the error in each element in the following way

$$\|u_h - u_e\|_0 = \sqrt{\int_\Omega (u_h - u_e)^2 dx} = \sqrt{\sum_\ell \int_{\Omega_\ell} (u_h - u_e)^2 dx}, \tag{1}$$

$$\|u_h - u_e\|_1 = \sqrt{\int_\Omega (u_h - u_e)^2 + \int_\Omega (u_h' - u_e')^2 dx} = \sqrt{\sum_\ell \left(\int_{\Omega_\ell} (u_h - u_e)^2 dx + \int_{\Omega_\ell} (u_h' - u_e')^2 dx\right)}. \tag{2}$$

We first present the algorithm that extracts the values of a basis function on a particular element. For a given element $\Omega_\ell$, it returns the evaluation points on this element, as well as the value of the basis function and derivatives for a given vector of weighting coefficients. This process is presented in Algorithm 1, and it performs similar steps to those of a FE problem assembly.

**Algorithm 1** `eval_func`

**Require:** `current_element` ▷ Index of current element
**Require:** `coefs` ▷ The coefficients of the reference basis functions.
**Require:** `element` $= [\omega_1, \omega_2]$
**Require:** `param_map` = (`map`, `map_derivatives`, `imap_derivatives`)
**Require:** `space` = ($n$, `supported_bases`, `extraction_coefficients`)
**Require:** `ref_data` = (`deg`, `reference_element`, `evaluation_points`, `quadrature_weights`, `reference_basis`, `reference_basis_derivatives`)

Construct $X \in \mathbb{R}^q$ where $X_i = \texttt{map}\,(\texttt{evaluation\_points}_i, x_1, x_2)$, for $i = 1, \ldots, q$
Initialize $N \leftarrow \mathbf{0} \in \mathbb{R}^q$
Initialize $dN \leftarrow \mathbf{0} \in \mathbb{R}^q$
$\ell \leftarrow$ `current_element`
$i' \leftarrow 0$
**for** $i \in \{\texttt{supported\_bases}_{rk} : r = \ell\}$ **do** ▷ Non-zero basis functions ($\ell$-th row)
  Construct $\mathbf{e} \in \mathbb{R}^{\texttt{deg}+1}$ where $\mathbf{e}_k = \texttt{extraction\_coefficients}_{\ell,i,k}$ for $k = 1, \ldots, \texttt{deg}+1$
  $N \leftarrow N + \texttt{coeffs}_{i'} \mathbf{e}^T \texttt{reference\_basis}$ ▷ Basis function on each evaluation point
  $dN \leftarrow dN + \texttt{coefs}_{i'} \texttt{imap\_derivatives}_\ell \left(\mathbf{e}^T \texttt{reference\_basis\_derivatives}\right)$
  $i' \leftarrow i' + 1$

**Ensure:** $X \in \mathbb{R}^q$ ▷ The evaluation points on the current element.
**Ensure:** $N \in \mathbb{R}^q$ ▷ The value of the basis function on $X$.
**Ensure:** $dN \in \mathbb{R}^q$ ▷ The value of the derivative of $N$ on $X$.

Based on this procedure, the pseudocode for the estimation of the $H^0$-norm ($L^2$-norm) is presented in Algorithm 2.

**Algorithm 2** `h0_norm`

**Require:** $u_e : \Omega \to \mathbb{R}$ ▷ The exact solution
**Require:** $\mathbf{u} \in \mathbb{R}^n$
**Require:** $\texttt{mesh} = (m, \texttt{elements})$
**Require:** $\texttt{param\_map} = (\texttt{map}, \texttt{map\_derivatives}, \texttt{imap\_derivatives})$
**Require:** $\texttt{space} = (n, \texttt{supported\_bases}, \texttt{extraction\_coefficients})$
**Require:** $\texttt{ref\_data} = (\texttt{deg}, \texttt{reference\_element}, \texttt{evaluation\_points}, \texttt{quadrature\_weights}, \texttt{reference\_basis}, \texttt{reference\_basis\_derivatives})$

$q \leftarrow \dim(\texttt{evaluation\_points})$
norm $\leftarrow 0$
**for** $\ell \in \{1, \dots, m\}$ **do**
  $\texttt{element} \leftarrow (\texttt{elements}_{1,\ell}, \texttt{elements}_{2,\ell})$
  $X, N, \_ \leftarrow \texttt{eval\_func}\,(\ell, \mathbf{u}, \texttt{element}, \texttt{param\_map}, \texttt{space}, \texttt{ref\_data})$
  Construct $\mathbf{u}^e \in \mathbb{R}^q$ where $\mathbf{u}^e_i = u_e(X_i)$
  Construct $y \in \mathbb{R}^q$ where $y_i = (N_i - \mathbf{u}^e_i)^2$
  $\texttt{norm} \leftarrow \texttt{norm} + \texttt{map\_derivatives}_\ell \left(y^T \texttt{quadrature\_weights}\right)$
$\texttt{norm} \leftarrow \sqrt{\texttt{norm}}$
**Ensure:** `norm`

Moreover, for the $H^1$-norm, the pseudo-code is presented in Algorithm 3.

**Algorithm 3** `h1_norm`

**Require:** $u_e : \bar{\Omega} \to \mathbb{R}$ ▷ The exact solution
**Require:** $du_e : \Omega \to \mathbb{R}$ ▷ The derivative of the exact solution
**Require:** $\mathbf{u} \in \mathbb{R}^n$
**Require:** $\texttt{mesh} = (m, \texttt{elements})$
**Require:** $\texttt{param\_map} = (\texttt{map}, \texttt{map\_derivatives}, \texttt{imap\_derivatives})$
**Require:** $\texttt{space} = (n, \texttt{supported\_bases}, \texttt{extraction\_coefficients})$
**Require:** $\texttt{ref\_data} = (\texttt{deg}, \texttt{reference\_element}, \texttt{evaluation\_points}, \texttt{quadrature\_weights}, \texttt{reference\_basis}, \texttt{reference\_basis\_derivatives})$

$q \leftarrow \dim(\texttt{evaluation\_points})$
norm $\leftarrow 0$
**for** $\ell \in \{1, \dots, m\}$ **do**
  $\texttt{element} \leftarrow (\texttt{elements}_{1,\ell}, \texttt{elements}_{2,\ell})$
  $X, N, dN \leftarrow \texttt{eval\_func}\,(\ell, \mathbf{u}, \texttt{element}, \texttt{param\_map}, \texttt{space}, \texttt{ref\_data})$
  Construct $\mathbf{u}^e \in \mathbb{R}^q$ where $\mathbf{u}^e_i = u_e(X_i)$
  Construct $d\mathbf{u}^e \in \mathbb{R}^q$ where $d\mathbf{u}^e_i = du_e(X_i)$
  Construct $y \in \mathbb{R}^q$ where $y_i = (N_i - \mathbf{u}^e_i)^2 + (dN_i - d\mathbf{u}^e_i)^2$
  $\texttt{norm} \leftarrow \texttt{norm} + \texttt{map\_derivatives}_\ell \left(y^T \texttt{quadrature\_weights}\right)$
$\texttt{norm} \leftarrow \sqrt{\texttt{norm}}$
**Ensure:** `norm`

B. We choose our function to be

$$u_e(x) = \pi x^2 - ex + 1. \tag{3}$$

The function evaluated in the endpoints of the interval is $u_e(0) = 1$ and $u_e(1) = \pi - e + 1$ and its second derivative is $u_e''(x) = 2\pi$, thus, the parameters of the problem are chosen as

$$f(x) = -2\pi, \qquad g_0 = 1, \qquad g_1 = \pi - e + 1. \tag{4}$$

The plots of the FEM approximation are presented in Fig. 1. Note how they show that the computed solution coincides with the exact solution. Moreover, the error in the $L^2$-norm and $H^1$-norm is of the order of $10^{-16}$ so we conclude both functions are equal up to rounding errors.

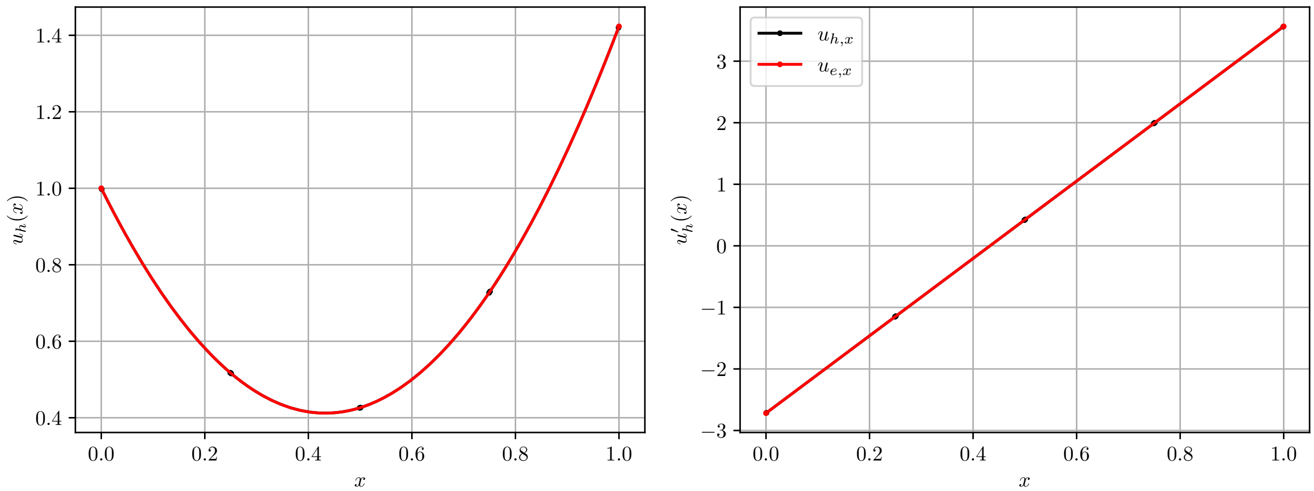



Figure 1: Plot of the FEM and exact solution and their derivatives.

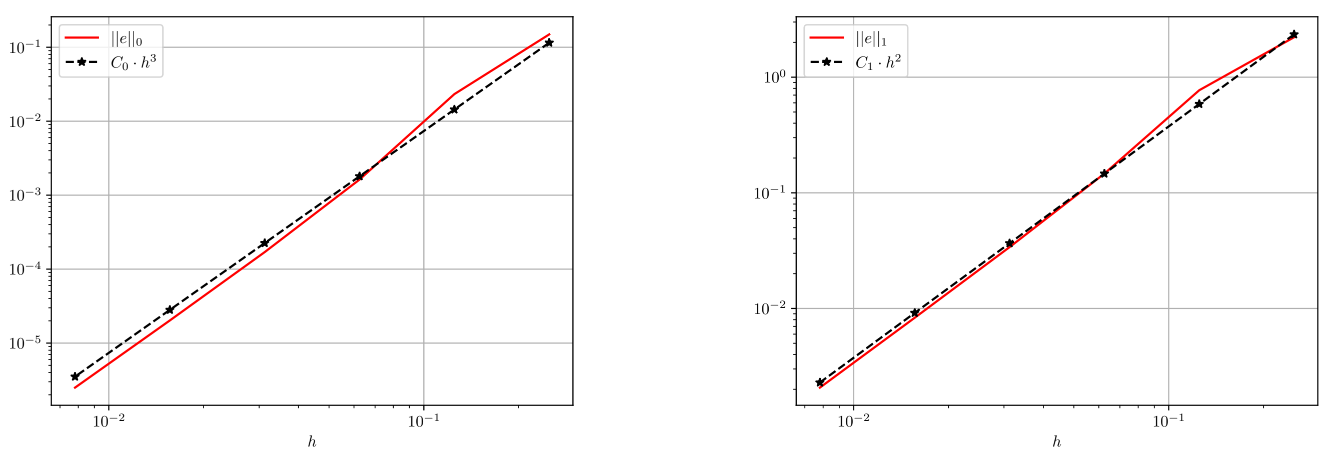
C. We choose our function to be

$$u_e(x) = e^{-6x^2} \sin(4\pi x), \tag{5}$$

The function evaluated in the endpoints of the interval is $u_e(0) = 0$ and $u_e(1) = 0$ and its second derivative is $u_e''(x) = -4e^{-6x^2}\left((-36x^2 + 4\pi^2 + 3)\sin(4\pi x) + 24\pi x \cos(4\pi x)\right)$. The parameters of the problem are chosen as

$$f(x) = 4e^{-6x^2}\left((-36x^2 + 4\pi^2 + 3)\sin(4\pi x) + 24\pi x \cos(4\pi x)\right), \qquad g_0 = 0, \qquad g_1 = 0. \tag{6}$$

We have a first-order problem ($s = 1$) and the exact solution is $C^\infty$. Following the results from the lectures, the convergence order should be 2 for the $H^1$ norm and 3 for the $L^2$ norm.




(a) $L^2$-norm.

(b) $H^1$ norm

Figure 2: Convergence of the error for the two norms (red) and expected convergence rate (black).

In Figs. 2, the logarithmic plots show the error for the two norms and $C_j \cdot h^r$, where $r$ is the expected convergence order and $C_j$ is a constant. Since the error is approximately parallel to the lines $C_j \cdot h^r$ it means that the error behaves in the way $K \cdot h^r$, i.e. the convergence error is precisely the expected one.

## 2 A 1D Mixed Finite Element Method

D. We first multiply the first equation of the strong formulation by a test function $\tau \in \mathcal{W}_\sigma$. By integration, we obtain

$$\int_\Omega \sigma\tau dx + \int_\Omega u_{,x}\tau dx = 0, \qquad \forall \tau \in \mathcal{W}_\sigma, \tag{7}$$

and integrating by parts yields

$$\int_\Omega \sigma\tau dx + (u\tau)\Big|_0^1 - \int_\Omega u\tau_{,x} dx = 0, \qquad \forall \tau \in \mathcal{W}_\sigma, \tag{8}$$

and using the boundary conditions of $u$

$$(\tau, \sigma) - (\tau_{,x}, u) = 0, \qquad \forall \tau \in \mathcal{W}_\sigma. \tag{9}$$

In this case, the boundary conditions are not enforced in the test functions to arrive at the weak form. Therefore, the boundary conditions are enforced in a weak way and do not need to be accounted for in the functional spaces.

E. For the forcing $f(x) = \pi^2 \cos(\pi(x - 0.5))$ we solve the problem

$$\begin{cases} -u_{,xx} = \pi^2 \cos(\pi(x - 0.5)), \\ u(0) = u(1) = 0, \end{cases} \tag{10}$$

Integrating the forcing two times yields

$$\iint -\pi^2 \cos(\pi(x - 0.5)) dx = \cos(\pi(x - 0.5)) + C_1 x + C_2, \tag{11}$$

and using the boundary conditions we get that

$$u(x) = \cos(\pi(x - 0.5)), \tag{12}$$

so

$$\sigma(x) = -u(x)_{,x} = \pi \sin(\pi(x - 0.5)). \tag{13}$$

F. In a general setting, if we assume that we have a system of $R$ differential equations with $R$ variables. We denote by superscripts the indexing of such variables and equations. In this way, let $B^{r,r'}(\cdot,\cdot)$ be the bilinear form of equation $r$ w.r.t. the variable $r'$, and let $L^r(\cdot)$ be the linear form for equation $r$, thus, the weak form of the system will be in the form

$$\sum_{r'=1}^{R} B^{r,r'}(u_h^r, w_h^r) = L^r(w_h^r), \; r = 1, \ldots, R, \tag{14}$$

and where $w_h^r$ is a test function. Therefore, we can assemble each combination of bilinear-linear forms using our implementation for the previous assignment and construct the linear system of equations using the resulting block matrices. The pseudo-code for a generalized implementation of the FEM for a mixed problem is presented in Algorithm 4.

**Algorithm 4** `assemble_mixed_problem`

**Require:** $\texttt{mesh} = (m, \texttt{elements})$

**Require:** $\texttt{spaces} = \left\{ (n^r, \texttt{supported\_bases}^r, \texttt{extraction\_coefficients}^r) \right\}_{r=1,\dots,R}$
▷ The set of FE spaces for each equation ($R$ equations in total)

**Require:** $\texttt{ref\_datas} = \big\{(\texttt{deg}^r, \texttt{reference\_element}^r, \texttt{evaluation\_points}^r, \texttt{quadrature\_weights}^r, \texttt{reference\_basis}^r, \texttt{reference\_basis\_derivatives}^r)\big\}_{r=1,\dots,R}$
▷ The set of reference data for each equation

**Require:** $\texttt{param\_maps} = \big\{(\texttt{map}^r, \texttt{map\_derivatives}^r, \texttt{imap\_derivatives}^r)\big\}_{r=1,\dots,R}$
▷ The set of parametric maps for each equation

**Require:** $\texttt{problem\_B\_mat} = \big\{\texttt{problem\_B}^{r,r'}(\cdot,\cdot,\cdot,\cdot,\cdot)\big\}_{r,r'=1,\dots,R}$
▷ The matrix-like set of left-hand sides

**Require:** $\texttt{problem\_Ls} = \big\{\texttt{problem\_L}^r(\cdot,\cdot,\cdot)\big\}_{r=1,\dots,R}$ ▷ The set of right-hand sides

$N \leftarrow \sum_{r=1}^{R} n^r$
$A \leftarrow \mathbf{0} \in \mathbb{R}^{N\times N}$
$b \leftarrow \mathbf{0} \in \mathbb{R}^{N}$
**for** $\ell \in \{1,\dots,m\}$ **do** ▷ Iterate over all elements
  $x_1 \leftarrow \texttt{elements}_{1,\ell}$
  $x_2 \leftarrow \texttt{elements}_{2,\ell}$
  $t \leftarrow 0$
  **for** $r \in \{1,\dots,R\}$ **do** ▷ Iterate over equations
    $q \leftarrow \dim(\texttt{evaluation\_points}^r)$
    Construct $X \in \mathbb{R}^q$ where $X_i = \texttt{map}^r(\texttt{evaluation\_points}^r_i, x_1, x_2)$, for $i = 1,\dots,q$
    $i' \leftarrow 0$
    **for** $i \in \{\texttt{supported\_bases}^r_{zk} : z = \ell\}$ **do**
      Let $\mathbf{e} \in \mathbb{R}^{\texttt{deg}^r+1}$ where $\mathbf{e}_k = \texttt{extraction\_coefficients}^r_{\ell,i',k}$, $k = 1,\dots,\texttt{deg}^r+1$
      $N_i \leftarrow \mathbf{e}^T\texttt{reference\_basis}^r$
      $dN_i \leftarrow \texttt{imap\_derivatives}^r_\ell\left(\mathbf{e}^T\texttt{reference\_basis\_derivatives}^r\right)$
      $L \leftarrow \texttt{problem\_L}^r(X, N_i, dN_i)$
      $b_{i+t} \leftarrow b_{i+t} + \texttt{map\_derivatives}^r_\ell\left(L^T\texttt{quadrature\_weights}^r\right)$
      $t' \leftarrow 0$
      **for** $r' \in \{1,\dots,R\}$ **do** ▷ Iterate over equations again
        $j' \leftarrow 0$
        **for** $j \in \left\{\texttt{supported\_bases}^{r'}_{zk} : z = \ell\right\}$ **do**
          Let $\tilde{\mathbf{e}} \in \mathbb{R}^{\texttt{deg}^{r'}+1}$ where $\tilde{\mathbf{e}}_k = \texttt{extraction\_coefficients}^{r'}_{\ell,j',k}$, $k = 1,\dots,\texttt{deg}^{r'}+1$
          $N_j \leftarrow \tilde{\mathbf{e}}^T\texttt{reference\_basis}^{r'}$
          $dN_j \leftarrow \texttt{imap\_derivatives}^{r'}_\ell\left(\tilde{\mathbf{e}}^T\texttt{reference\_basis\_derivatives}^{r'}\right)$
          $B \leftarrow \texttt{problem\_B}^{r,r'}(X, N_i, dN_i, N_j, dN_j)$
          $A_{i+t,j+t'} \leftarrow A_{i+t,j+t'} + \texttt{map\_derivatives}^{r'}_\ell\left(B^T\texttt{quadrature\_weights}^{r'}\right)$
          $j' \leftarrow j' + 1$
        $t' \leftarrow t' + n^{r'}$ ▷ Advance to the next column-block
      $i' \leftarrow i' + 1$
    $t \leftarrow t + n^r$ ▷ Advance to the next row-block

**Ensure:** $A \in \mathbb{R}^N$

**Ensure:** $b \in \mathbb{R}^N$

G. – Choice 1.

$$\begin{aligned} \mathcal{S}_{u,h}: &\quad p_u = 0, \quad k_u = -1, \\ \mathcal{S}_{\sigma,h}: &\quad p_\sigma = 1, \quad k_\sigma = 0. \end{aligned} \tag{15}$$

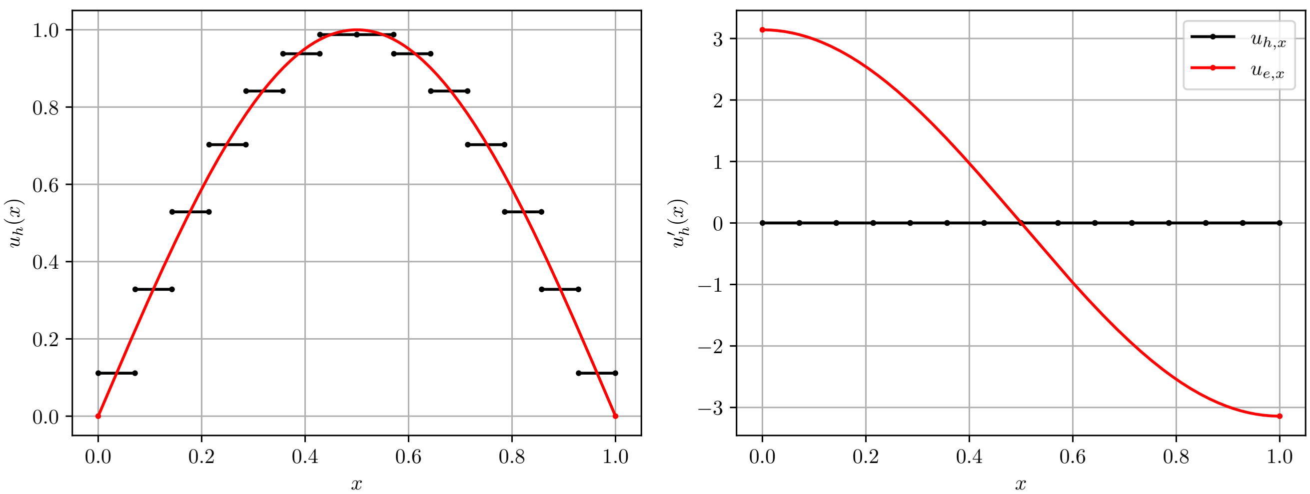



Figure 3: Plot of the solution $u$ and $u_{,x}$.

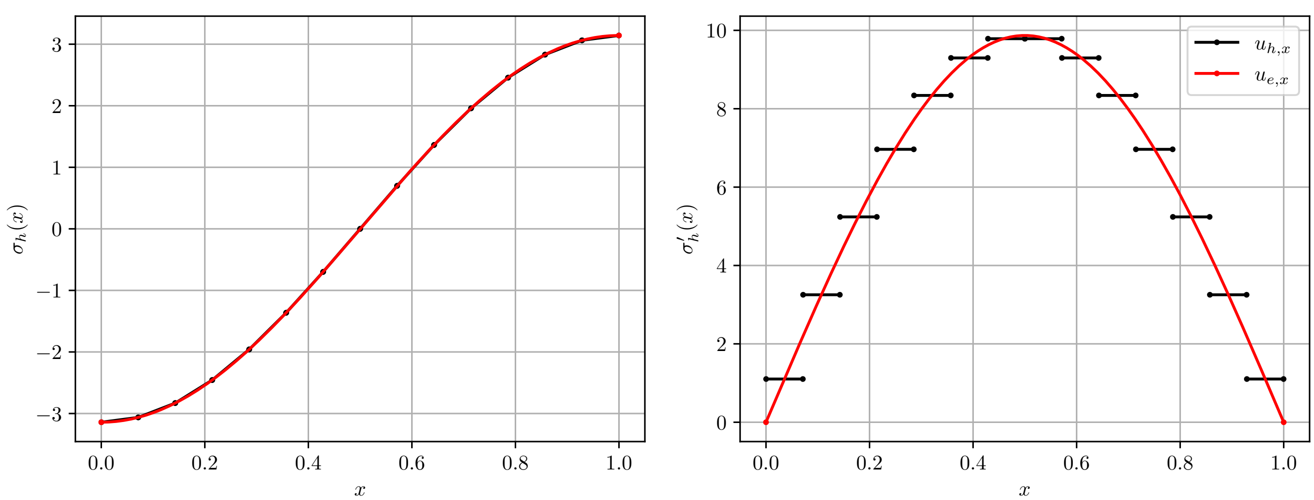



Figure 4: Plot of the solution $\sigma$ and $\sigma_{,x}$.

Figures 3 and 4 show that the first choice of finite element discretizations leads to a stable accurate solution.

– Choice 2.

$$\begin{aligned} \mathcal{S}_{u,h}: &\quad p_u = 1, \quad k_u = 0, \\ \mathcal{S}_{\sigma,h}: &\quad p_\sigma = 1, \quad k_\sigma = 0. \end{aligned} \tag{16}$$

This choice of finite element discretizations leads to a singular matrix.

– Choice 3.

$$\begin{aligned} \mathcal{S}_{u,h}: &\quad p_u = 0, \quad k_u = -1, \\ \mathcal{S}_{\sigma,h}: &\quad p_\sigma = 2, \quad k_\sigma = 0. \end{aligned} \tag{17}$$

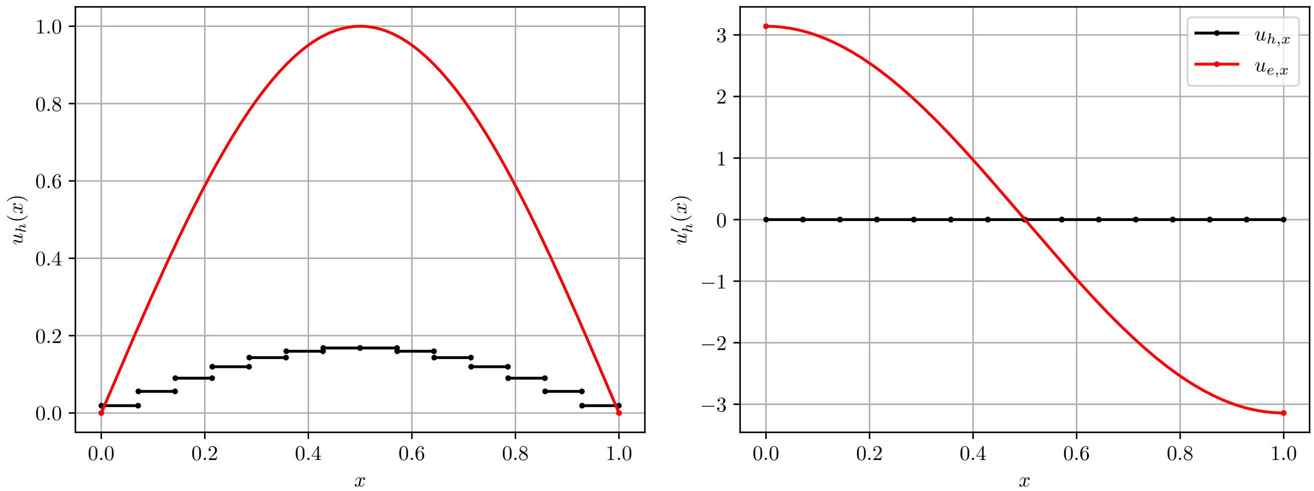



Figure 5: Plot of the solution $u$ and $u_{,x}$.

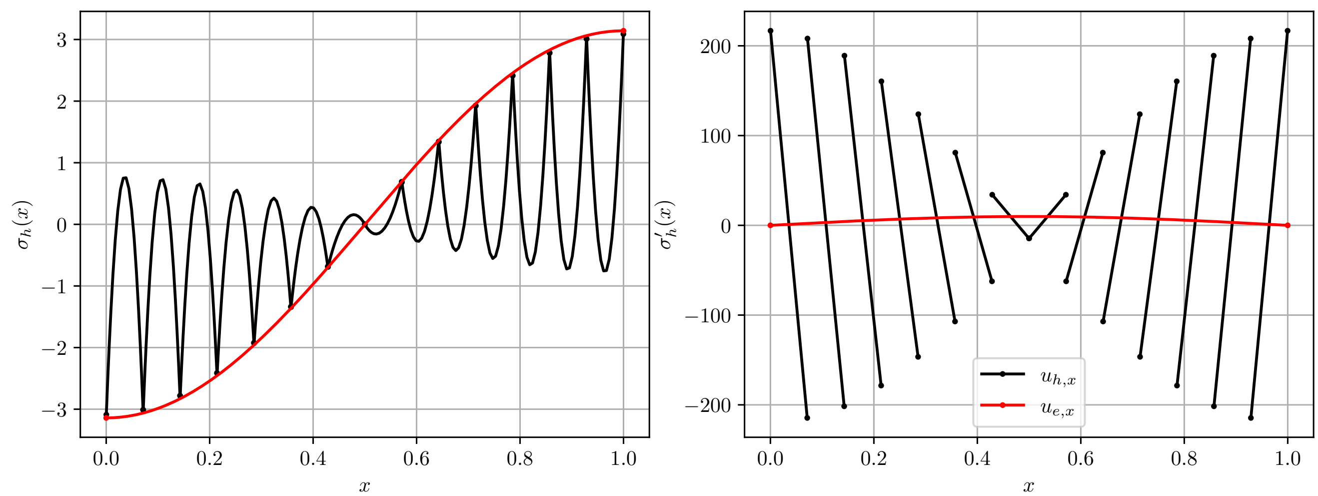



Figure 6: Plot of the solution $\sigma$ and $\sigma_{,x}$.

Figures 5 and 6 show that the last choice of finite element discretizations leads to an unstable inaccurate solution.

H. The finite-dimensional spaces would correspond to conforming finite element methods if $\mathcal{S}_{u,h} = \mathcal{F}(p_u, k_u; \mathcal{T}_h)$ and $\mathcal{S}_{\sigma,h} = \mathcal{F}(p_\sigma, k_\sigma; \mathcal{T}_h)$ are subspaces of $\mathcal{S}_u = L^2(\Omega)$ and $\mathcal{S}_\sigma = H^1(\Omega)$ respectively and non-conforming finite element methods otherwise. For the function to be in $H^1(\Omega)$, it must be at least continuous. It is clear that a function with jumps can still be in $L^2$, but its derivative at a jumping point would be given in terms of Dirac deltas, and it is well known that this operator is not in $L^2$. Given how the finite element spaces are constructed, this condition reduces to $k_u \geq -1$ and $k_\sigma \geq 0$. Both conditions are always satisfied so all choices correspond to conforming finite element methods.