MATH 521 - Numerical Analysis of Differential Equations

Christoph Ortner, 01/2024

Assignment 1 : One Dimension

Name:

Student ID:

Q1: Implementation of Model Problem

Recall our first boundary value problem that we studied in class,

$$\begin{aligned} -u'' &= f, \quad x \in (0,1), \\ u(0) &= u(1) = 0. \end{aligned}$$

We reformulated this in the weak form: find $u \in H^1_0(0,1)$ such that

$$\int_0^1 u'v'dx = \int_0^1 fvdx \qquad \forall v \in H^1_0(0,1).$$

We then defined the finite element method as follows:

- Specify the nodes for a mesh: $0 = x_0 < x_1 < \cdots < x_N = 1$
- Specify the space $V_h = \{u_h : \text{cts, p.w. affine w.r.t. } (x_j)_j\}$
- Fine $u_h \in V_h$ such that

$$\int_0^1 u_h'v_h'dx = \int_0^1 fv_hdx \qquad \forall v_h \in H^1_0(0,1).$$

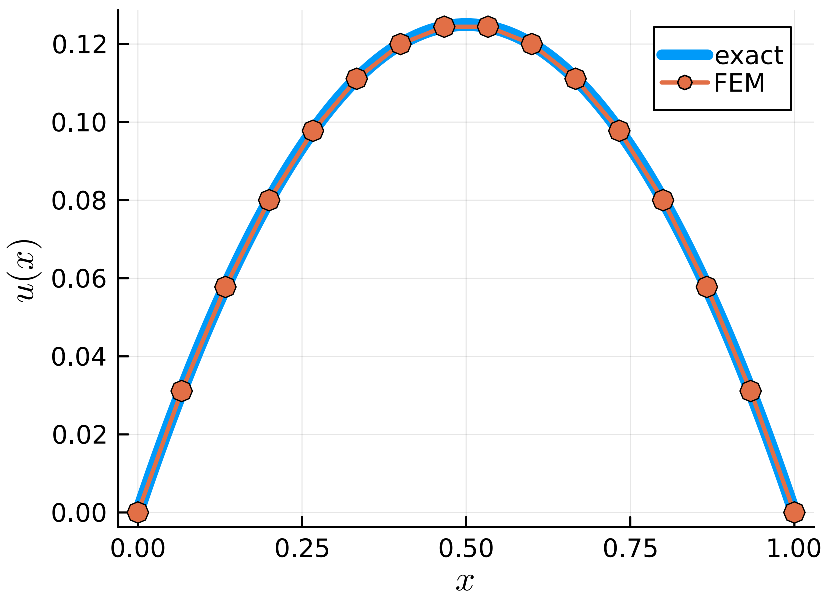
Your task: Implement this numerical scheme, using mid-point quadrature (as in class) solve it with $f(x) = 1$, plot both the exact solution and the finite element solution (for $N = 15$).

In [1]:
```
using Plots, LaTeXStrings
```

In [2]:
```julia
# outline of the implementation

function assemble_system(X, f)
    # input
    #   X : list of grid points, e.g. as Vector{Float64}
    #   f : function to evaluate f(x)

    N = length(X) - 1    # number of elements
    A = zeros(N+1, N+1)  # should be sparse, but let's not worry
    F = zeros(N+1)

    for j = 1:N
        # compute the contributions to F and A from the element (xⱼ₋₁, xⱼ)
        # and write them into A, F
        ξⱼ = 0.5 * (X[j]+X[j+1])
        hⱼ = X[j+1] - X[j]
        # assemble the forcing term
        # ψⱼ(ξⱼ) = ψⱼ₋₁(ξⱼ) = 0.5
        f̄ⱼ = hⱼ * f(ξⱼ)
        F[j] += f̄ⱼ * 0.5
        F[j+1] += f̄ⱼ * 0.5
        # assemble the stiffness matrix
        # ψⱼ' = 1/hⱼ, ψⱼ₋₁' = -1/hⱼ constant in the element
        A[j, j] += 1/hⱼ
        A[j, j+1] -= 1/hⱼ
        A[j+1, j] -= 1/hⱼ
        A[j+1, j+1] += 1/hⱼ
    end

    return A, F
end

# My suggestion is that `assemble_system` returns
# A and F ignoring the boundary condition i.e. for the full
# N+1 DOFs. We can then reduce those to the required size
# for solving only for the free DOFs. (Think about why this works!)

N = 15
X = range(0, 1, length = N+1)
f = x -> 1.0
A, F = assemble_system(X, f)
U = zeros(N+1)
U[2:N] = A[2:N, 2:N] \ F[2:N];
```

In [3]:
```
# the postprocessing and visualization should be done in a separate cell
# from the computation.

xp = range(0, 1, 100)
u = xp -> 0.5 * xp .* (1-xp)

plot(; xlabel = L"x", ylabel = L"u(x)", size = (400, 300))
plot!(xp, u, lw=6, label = "exact")
plot!(X, U, lw=2, m=:o, ms=5, label = "FEM")
```

Out[3]:



Q2-pre

To solve the following question you will need a little extra piece of information that I hinted at in class but didn't really work out completely: in one dimension, point evaluation is a continuous / bounded operation in the typical Sobolev spaces we encounter. Concretely, the following is true: let $\hat{x} \in (0,1)$ and let $v \in C^1([0,1])$ then

$$\big|v(\hat{x}) - v(0)\big| \leq C \|v'\|_{L^2(0,1)}$$

for some suitable constant $C > 0$. Prove this statement.

$$
\begin{aligned}
\left|v(\hat{x}) - v(0)\right| &= \left|\int_0^{\hat{x}} v'(x)dx\right| \\
&\leq \int_0^{\hat{x}} |v'(x)|dx \\
&\leq \sqrt{\hat{x}}\left(\int_0^{\hat{x}} \left|v'(x)\right|^2 dx\right)^{1/2} \\
&\leq \sqrt{\hat{x}}\|v'\|_{L^2(0,1)}.
\end{aligned}
$$

Q2: Neumann Boundary Condition

Consider the boundary value problem

$$
\begin{aligned}
-u'' &= f, \quad x \in (0,1), \\
u(0) &= 0, \\
u'(1) &= g.
\end{aligned}
$$

where a, f are continuous in $[0,1]$, $a(x) > 0$, $g \in \mathbb{R}$.

(1) Derive the weak form. Prove that it has a unique solution.

HINT: the correct function space this time is not $H_0^1(0,1)$. Remember from class how we chose the test function!

(2) Formulate the corresponding finite element method. Prove that it has a unique solution.

(3) Prove that the FEM solution is the best approximation in a natural norm that you should specify.

Solution (Q2.1)

$$\int_0^1 u'v'dx = \int fvdx + gv(1) \qquad \forall v \in H_N^1(0,1),$$

where the space $H_N^1(0,1)$ is defined as follows:

$$H_N^1(0,1) = \{u \in H^1(0,1) : u(0) = 0\} = \mathrm{clos}\{u \in C^1([0,1]) : u(0) = 0\}$$

with the closure taken in the norm $|v|_1 := \|v'\|_{L^1(0,1)}$.

By construction, H_N^1 is a Hilbert space under the inner product $(u,v)_{H_N^1} = (u',v')_{L^2}$. To show that

Solution (Q2.2)

Let $0 = x_0 < x_1 < \cdots < x_N$,

$$V_h = \{v_h \in C([0,1]) : v_h(0) = 0, \mathrm{p.w.aff.\ w.r.t.\ } (x_j)\}.$$

FEM : Find $u_h \in V_h$ such that

$$\int_0^1 u_h'v_h'dx = \int fv_hdx + gv_h(1) \qquad \forall v \in V_h.$$

Solution (Q2.3)

The proofs from class can be followed verbatim, there is no change to be made. For ease of notation, we define $a(u, v) = \int_0^1 u'v'dx, \ell(v) = \int_0^1 fvdx$ and $|u|_a := |u|_1 = \sqrt{a(u, u)}$. Step 1 : Galerkin orthogonality. Since $V_h \subset H^1_N$,

$$\begin{aligned} &a(u, v_h) = \ell(v_h) = a(u_h, v_h) \qquad \forall v_h \in V_h \\ &\Rightarrow \quad a(u - u_h, v_h) = 0. \end{aligned}$$

Step 2 : Cea's lemma

$$\begin{aligned} |u - u_h|_a^2 &= a(u - u_h, u - u_h) \\ &= a(u - u_h, u - w_h) \qquad \forall w_h \in V_h \\ &\leq |u - u_h|_a |u - w_h|_a \qquad \text{(Cauchy\_Schwarz Ineq)} \end{aligned}$$

Dividing by $|u - u_h|_a$ we obtain

$$|u - u_h|_a \leq |u - w_h|_a \qquad \forall w_h \in V_h.$$

(Note if $u = u_h$ then division by zero is not a problem since the LHS = 0 and hence the inequality still holds.)

Q3: Implementation of Q2

Implement the method you defined in Q2. Copy-paste your code from Q1 and adapt it.

HINT: only a single line needs to be added to the assemble, then the solution script that enforces the boundary condition needs to be adapted suitably.

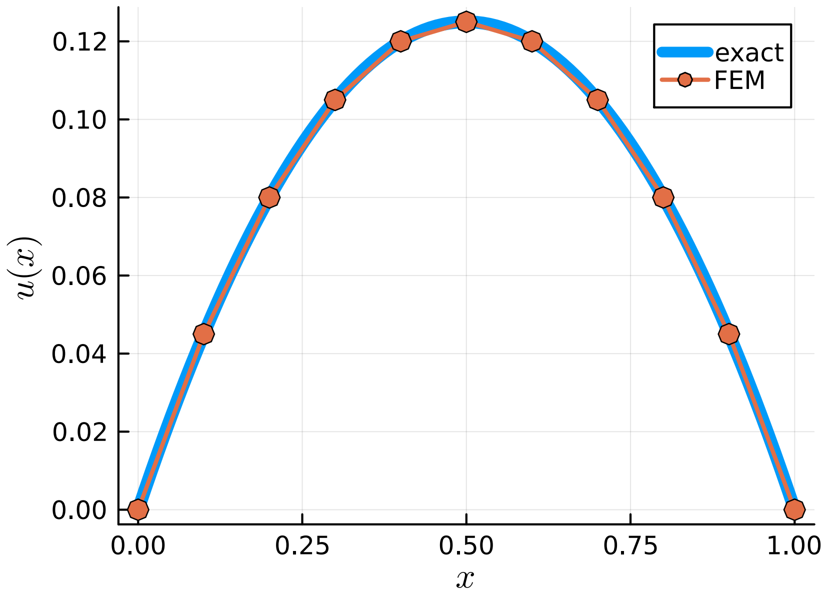
Use it to solve the BVP from Q2 with $f = 1$ and $g = -1/2$ and $N = 10$. Plot the exact solution and the FEM solution.

In [4]:
```
# outline of the implementation

function assemble_system_neumann(X, f, g)
    # input
    #   X : list of grid points, e.g. as Vector{Float64}
    #   f : function to evaluate f(x)
    #   g : traction value (number)

    N = length(X) - 1     # number of elements
    A = zeros(N+1, N+1)  # should be sparse, but let's not worry
```

```
    F = zeros(N+1)

    for j = 1:N
        # compute the contributions to F and A from the element (xⱼ₋₁, xⱼ)
        # and write them into A, F
        ξⱼ = 0.5 * (X[j]+X[j+1])
        hⱼ = X[j+1] - X[j]
        # assemble the forcing term
        # ψⱼ(ξⱼ) = ψⱼ₋₁(ξⱼ) = 0.5
        f̄ⱼ = hⱼ * f(ξⱼ)
        F[j] += f̄ⱼ * 0.5
        F[j+1] += f̄ⱼ * 0.5
        # assemble the stiffness matrix
        # ψⱼ' = 1/hⱼ, ψⱼ₋₁' = -1/hⱼ constant in the element
        A[j, j] += 1/hⱼ
        A[j, j+1] -= 1/hⱼ
        A[j+1, j] -= 1/hⱼ
        A[j+1, j+1] += 1/hⱼ
    end

    # for j = N we still need to deal with the traction term
    #  g v_h(1) which becomes g ψⱼ(1). This is non-zero ONLY
    # for j = N.
    F[N+1] += g

    return A, F
end

# My suggestion is that `assemble_system` returns
# A and F ignoring the boundary condition. We can
# use this as follows:

N = 10
X = range(0, 1, length = N+1)
f = x -> 1.0
g = -0.5
A, F = assemble_system_neumann(X, f, g)
U = zeros(N+1)
U[2:N+1] = A[2:N+1, 2:N+1] \ F[2:N+1];
# (why is this correct?!?!?)
```

In [5]:
```julia
# the postprocessing and visualization should be done in a separate cell
# from the computation.

xp = range(0, 1, 100)
u = xp -> 0.5 * xp * (1-xp)

plot(; xlabel = L"x", ylabel = L"u(x)", size=(400, 300))
plot!(xp, u, lw=6, label = "exact")
plot!(X, U, lw=2, m=:o, ms=5, label = "FEM")
```

Out[5]:



NOTE TO STUDENTS: it was an accident that the solution is the same as for the Dirichlet problem. I chose poor parameters. Of course this is not normally the case. Try it with some other parameters