Mario L. Gutierrez Abed
364009832
mlg3843@rit.edu

Problem Set 2
Numerical Analysis II

04-17-2021

Problem 1. Derive the variational form (or weak form) of the BVP

$$-\frac{d}{dx}\left(k(x)\frac{du}{dx}\right) + p(x)u = f(x), \quad 0 < x < 1, \tag{1a}$$

$$u(0) = 0, \tag{1b}$$

$$u(1) = 0, \tag{1c}$$

where p and k satisfy $p(x) > 0$ and $k(x) > 0$ for $x \in [0, 1]$. What is the bilinear form for this BVP? Write a Matlab code using piecewise linear finite elements to solve the above problem. Construct one example. Compare the numerical solution with the exact solution for $n = 20$.

Proof. To simplify the notation we drop the explicit dependence on x and write primes for the derivatives; thus our task is to find the weak form of

$$-(ku')' + pu = f. \tag{2}$$

We will make use of test functions from the space

$$H_0^1 = \left\{v \in L^2([0,1]) \mid v' \in L^2([0,1]), v(0) = 0, v(1) = 0\right\}.$$

Multipliying Eq. (2) by $v \in H_0^1$ and integrating, we get

$$-\int_0^1 (ku')' v \, dx + \int_0^1 puv \, dx = \int_0^1 fv \, dx. \tag{3}$$

We notice that an application of the product rule yields

$$ku'v\Big|_0^1 = \int_0^1 (ku'v)' \, dx = \int_0^1 (ku')' v \, dx + \int_0^1 ku'v' \, dx.$$

Substituting back into Eq. (3), we get

$$\int_0^1 ku'v' \, dx - ku'v\Big|_0^1 + \int_0^1 puv \, dx = \int_0^1 fv \, dx.$$

We then notice that the term

$$ku'v\Big|_0^1 = k(1)u'(1)v(1) - k(0)u'(0)v(0)$$

vanishes because v also vanishes at the endpoints. Thus we conclude that the weak form of the BVP (2) is given by

$$\boxed{\int_0^1 ku'v' \, dx + \int_0^1 puv \, dx = \int_0^1 fv \, dx} \tag{4}$$

Whence the bilinear form $a(\cdot,\cdot)$ associated with this system is

$$a(u,v) = \int_{[0,1]} (ku'v' + puv).$$

Our job now is to find a suitable solution $u \in H_0^1$ that satisfies (4) for all $v \in H_0^1$. In fact, the space H_0^1 is too large to be of practical use; in the Galerkin approach a suitable finite subspace is used instead. For our purposes, the subspace containing all continuous, piecewise linear functions will suffice. Let $I = [0, 1]$ and let the vector space of linear functions on I be denoted by $P_1(I)$:

$$P_1(I) = \{v \mid v(x) = \alpha_0 + \alpha_1 x;\ x \in I;\ \alpha_0, \alpha_1 \in \mathbb{R}\}.$$

We shall make use of $n+1$ nodes $\{x_i\}_{i=0}^n$ and partition I in the usual way

$$0 = x_0 < x_1 < \cdots < x_{n-1} < x_n = 1,$$

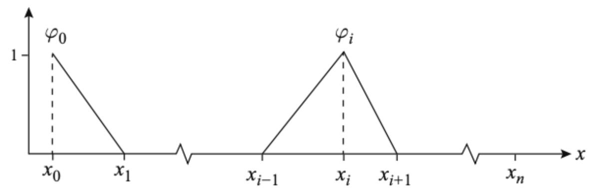
so there are n subintervals $I_i = [x_{i-1}, x_i]$, with $i = 1, \ldots, n$, each of length $h_i = x_i - x_{i-1}$.[1] Then the subspace we shall work with is

$$V_n = \left\{ v \mid v \in C^0(I);\ v|_{I_i} \in P(I_i); v(0) = v(1) = 0 \right\},$$

where $C_0(I)$ denotes as usual the space of all continuous functions on I. Hence, as we alluded to earlier, our work space V_n is the space containing all continuous, piecewise linear functions on the interval I. Moreover, since we need to fulfill the boundary criteria from the original strong-form BVP, we are also imposing the vanishing property at the endpoints in V_n.

Our next order of business is to introduce the basis of hat functions $\{\varphi_j\}_{j=0}^n$ for V_n, which satisfies

$$\varphi_j(x_i) = \begin{cases} 1 & \text{if } i = j; \\ 0 & \text{otherwise.} \end{cases}$$



Referring to the figure, we can easily deduce an explicit expression for the hats:

$$\varphi_i(x) = \begin{cases} \frac{x - x_{i-1}}{h_i} & \text{if } x \in I_i; \\ \frac{x_{i+1} - x}{h_{i+1}} & \text{if } x \in I_{i+1}; \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Next we approximate the solution u by a continuous piecewise linear function ${}^{(n)}u$, so that ${}^{(n)}u, v \in V_n$. Then, since $\{\varphi_j\}_{j=0}^n$ is a basis for V_n, we can do the following two things: i) replace the v's with φ's in Eq. (4), since it suffices to see what happens at the basis only; ii) we consider the ansatz

$${}^{(n)}u = \sum_{i=0}^{n} U_i \varphi_i. \tag{6}$$

We may then rewrite Eq. (4) as

$$\int_I k\,{}^{(n)}u' \varphi_i' \, dx + \int_I p\,{}^{(n)}u \varphi_i \, dx = \int_I f \varphi_i \, dx \qquad i = 0, \ldots, n. \tag{7}$$

However, note that, even though this expression is valid for all $i \in \{0, \ldots, n\}$, due to the vanishing boundary conditions we only get nonvanishing terms for $i \in \{1, \ldots, n-1\}$; whence from now on we shall focus only in i in this range. Plugging in the ansatz (6) on the LHS, we get

$$\begin{aligned} \int_I k \left(\sum_{j=1}^{n-1} U_j \varphi_j' \right) \varphi_i' \, dx + \int_I p \left(\sum_{j=1}^{n-1} U_j \varphi_j \right) \varphi_i \, dx &= \sum_{j=1}^{n-1} U_j \int_I k \varphi_j' \varphi_i' \, dx + \sum_{j=1}^{n-1} U_j \int_I p \varphi_j \varphi_i \, dx \\ &= \sum_{j=1}^{n-1} U_j \int_I \left(k \varphi_j' \varphi_i' + p \varphi_j \varphi_i \right) \, dx \qquad i = 1, \ldots, n-1. \end{aligned}$$

Hence we have a system of the form

$$\boxed{(\boldsymbol{K} + \boldsymbol{M}) \boldsymbol{U} = \boldsymbol{F}} \tag{8}$$

where

$$K_{ij} = \int_I k \varphi_j' \varphi_i' \, dx \qquad \text{(Stiffness Matrix)}$$

$$M_{ij} = \int_I p \varphi_j \varphi_i \, dx \qquad \text{(Mass Matrix)}$$

$$F_i = \int_I f \varphi_i \, dx. \qquad \text{(Load Vector)}$$

[1] For future reusability of the code and to ensure flexibility, we will not assume that the partition is uniform; i.e., there will be no h such that $h = h_i \ \forall i$.

Let us now write out the nonvanishing components of these arrays. Note that, since for $|i-j|>1$ *the hats (and their derivatives) lack common support, both* $\mathbf{K}$ *and* $\mathbf{M}$ *will be tridiagonal. In all cases we shall use Simpson's quadrature; we recall that Simpson's method applied to an interval* $I=[x_{i-1},x_i]$ *takes the form*

$$\int_I f \approx \frac{h_i}{6}\left[f(x_{i-1})+4f(m_i)+f(x_i)\right],$$

where x_m *is the midpoint* $m_i=\frac{1}{2}(x_i+x_{i-1})$ *and* $h_i=x_i-x_{i-1}$*. We start with the mass matrix* $\mathbf{M}$*; its diagonal entries are given by*

$$\begin{aligned}
M_{ii} &= \int_I p\varphi_i^2\,dx \\
&= \int_{x_{i-1}}^{x_i} p\varphi_i^2\,dx + \int_{x_i}^{x_{i+1}} p\varphi_i^2\,dx \\
&= \frac{h_i}{6}\left[p(x_{i-1})\cdot 0 + 4\cdot p(m_i)\cdot\left(\frac{1}{2}\right)^2 + p(x_i)\cdot 1\right] + \frac{h_{i+1}}{6}\left[p(x_i)\cdot 1 + 4\cdot p(m_{i+1})\cdot\left(\frac{1}{2}\right)^2 + p(x_{i+1})\cdot 0\right] \\
&= \frac{h_i}{6}\left[p(m_i)+p(x_i)\right] + \frac{h_{i+1}}{6}\left[p(x_i)+p(m_{i+1})\right].
\end{aligned}$$

Similarly, for the subdiagonal entries,

$$\begin{aligned}
M_{i+1,i} &= \int_I p\varphi_i\varphi_{i+1}\,dx \\
&= \int_{x_i}^{x_{i+1}} p\varphi_i\varphi_{i+1}\,dx \\
&= \frac{h_{i+1}}{6}\left[p(x_i)\cdot 0 + 4\cdot p(m_{i+1})\cdot\left(\frac{1}{2}\right)^2 + p(x_{i+1})\cdot 0\right] \\
&= \frac{h_{i+1}\cdot p(m_{i+1})}{6}.
\end{aligned}$$

By symmetry, the superdiagonal entries are identical to the subdiagonal ones; i.e., $M_{i,i+1}=M_{i+1,i}$*. The following Matlab routine will assemble* $\mathbf{M}$*:*

```matlab
function M = MassMatD0(x,p)
    %input mesh vector x and function p to  MassMatD0
    %output Mass Matrix M

    n = length(x)-1;        %number of subintervals
    M = zeros(n-1, n-1);    %allocate mass matrix
    %No need for half-hats due to vanishing BCs; otherwise M would have dim (n+1)x(n+1)

    for i = 1:n-1
        h_minus = x(i+1) - x(i);                %h_i        (index offset)
        xmid = (x(i+1) + x(i))/2;               %m_i        (index offset)
        h_plus = x(i+2) - x(i+1);               %h_{i+1}    (index offset)
        xmid_plus = (x(i+2) + x(i+1))/2;        %m_{i+1}    (index offset)

        M(i,i) = (h_minus/6) * ( p(xmid) + p(x(i+1)) )
                + (h_plus/6) * ( p(xmid_plus) + p(x(i+1)) ) ;

        if i ~= n-1
            M(i+1,i)  =  ( h_plus * p(xmid_plus) )/6;
            M(i,i+1) = M(i+1,i) ;
        end
    end

end
```

Similarly, we now build the stiffness matrix. Before we start, however, we neeed to know the derivatives of the hat functions. A quick glance at Eq. (5) reveals that

$$\varphi_i'(x)=\begin{cases}\frac{1}{h_i} & \text{if } x\in I_i;\\ -\frac{1}{h_{i+1}} & \text{if } x\in I_{i+1};\\ 0 & \text{otherwise.}\end{cases} \tag{9}$$

The diagonal entries of $\mathbf{K}$ *are then given by*

$$
\begin{aligned}
K_{ii} &= \int_I k \left(\varphi_i'\right)^2 dx \\
&= \int_{x_{i-1}}^{x_i} k \left(\varphi_i'\right)^2 dx + \int_{x_i}^{x_{i+1}} k \left(\varphi_i'\right)^2 dx \\
&= \frac{h_i}{6}\left[k(x_{i-1}) \cdot \left(\frac{1}{h_i}\right)^2 + 4 \cdot k(m_i) \cdot \left(\frac{1}{h_i}\right)^2 + k(x_i) \cdot \left(\frac{1}{h_i}\right)^2\right] \\
&+ \frac{h_{i+1}}{6}\left[k(x_i) \cdot \left(-\frac{1}{h_{i+1}}\right)^2 + 4 \cdot k(m_{i+1}) \cdot \left(-\frac{1}{h_{i+1}}\right)^2 + k(x_{i+1}) \cdot \left(-\frac{1}{h_{i+1}}\right)^2\right] \\
&= \frac{1}{6h_i}\left[k(x_{i-1}) + 4k(m_i) + k(x_i)\right] + \frac{1}{6h_{i+1}}\left[k(x_i) + 4k(m_{i+1}) + k(x_{i+1})\right].
\end{aligned}
$$

Similarly, for the subdiagonal entries,

$$
\begin{aligned}
K_{i+1,i} &= \int_I k\varphi_i'\varphi_{i+1}' \, dx \\
&= \int_{x_i}^{x_{i+1}} k\varphi_i'\varphi_{i+1}' \, dx \\
&= \frac{h_{i+1}}{6}\left[k(x_i) \cdot \left(-\frac{1}{h_{i+1}}\right)\left(\frac{1}{h_{i+1}}\right) + 4 \cdot k(m_{i+1}) \cdot \left(-\frac{1}{h_{i+1}}\right)\left(\frac{1}{h_{i+1}}\right) + k(x_{i+1}) \cdot \left(-\frac{1}{h_{i+1}}\right)\left(\frac{1}{h_{i+1}}\right)\right] \\
&= -\frac{1}{6h_{i+1}}\left[k(x_i) + 4k(m_{i+1}) + k(x_{i+1})\right].
\end{aligned}
$$

Again, by symmetry, $K_{i+1,i} = K_{i,i+1}$. *The following Matlab routine assembles* $\mathbf{K}$*:*

```
function K = StiffMatD0(x, k)
    %input mesh vector x and function k to  StiffMatD0
    %output Stiffness Matrix K

    n = length(x)-1;          %number of subintervals
    K = zeros(n-1, n-1);      %allocate stiffness matrix
    %No need for half-hats due to vanishing BCs; otherwise M would have dim (n+1)x(n+1)

    for i = 1:n-1
        h_minus = x(i+1) - x(i);                 %h_i       (index offset)
        xmid = (x(i+1) + x(i))/2;                %m_i       (index offset)
        h_plus = x(i+2) - x(i+1);                %h_{i+1}   (index offset)
        xmid_plus = (x(i+2) + x(i+1))/2;         %m_{i+1}   (index offset)

        K(i,i) = (1/(6*h_minus)) * ( k(x(i)) + 4*k(xmid) + k(x(i+1)) )
            + (1/(6*h_plus)) * ( k(x(i+1)) + 4*k(xmid_plus) + k(x(i+2)) ) ;

        if i ~= n-1
            K(i+1,i)  =  - (1/(6*h_plus)) * ( k(x(i+1)) + 4*k(xmid_plus) + k(x(i+2)) );
            K(i,i+1) = K(i+1,i);
        end
    end

end
```

We are down to the final component that needs to be calculated; the load vector $\mathbf{F}$*:*

$$
\begin{aligned}
F_i &= \int_I f\varphi_i \, dx \\
&= \int_{x_{i-1}}^{x_i} f\varphi_i \, dx + \int_{x_i}^{x_{i+1}} f\varphi_i \, dx \\
&= \frac{h_i}{6}\left[f(x_{i-1})\varphi_i(x_{i-1}) + 4f(m_i)\varphi_i(m_i) + f(x_i)\varphi_i(x_i)\right] \\
&+ \frac{h_{i+1}}{6}\left[f(x_i)\varphi_i(x_i) + 4f(m_{i+1})\varphi_i(m_{i+1}) + f(x_{i+1})\varphi_i(x_{i+1})\right] \\
&= \frac{h_i}{6}\left[f(x_{i-1}) \cdot 0 + 4f(m_i) \cdot \left(\frac{1}{2}\right) + f(x_i) \cdot 1\right] \\
&+ \frac{h_{i+1}}{6}\left[f(x_i) \cdot 1 + 4f(m_{i+1}) \cdot \left(\frac{1}{2}\right) + f(x_{i+1}) \cdot 0\right]
\end{aligned}
$$

$$= \frac{h_i}{6}\left[2f(m_i) + f(x_i)\right] + \frac{h_{i+1}}{6}\left[f(x_i) + 2f(m_{i+1})\right].$$

The following Matlab routine assembles $\mathbf{F}$:

```
function F = LoadVecD0(x, f)
    %input mesh vector x and function f to  LoadVecD0
    %output Load Vector F

    n = length(x)-1;          %number of subintervals
    F = zeros(n-1, 1);       %allocate load vector
    %No need for half-hats due to vanishing BCs; otherwise F would have dim n+1

    for i = 1:n-1
        h_minus = x(i+1) - x(i);                  %h_i         (index offset)
        xmid = (x(i+1) + x(i))/2;                 %m_i         (index offset)
        h_plus = x(i+2) - x(i+1);                 %h_{i+1}     (index offset)
        xmid_plus = (x(i+2) + x(i+1))/2;          %m_{i+1}     (index offset)

        F(i) = (h_minus/6) * ( f(x(i+1)) + 2*f(xmid) )
             + (h_plus/6) * ( f(x(i+1)) + 2*f(xmid_plus) ) ;

    end

end
```

We now construct an example to test our code. Consider the ansatz

$$u(x) = -x^2 + x.$$

The function u *certainly satisfies the vanishing Dirichlet boundary conditions. We then choose the following functions* p *and* k:

$$p(x) = 5xe^x;$$
$$k(x) = 1 + x.$$

Then, for the given u, p, *and* k, *the BVP (1) becomes*

$$-\frac{d}{dx}\left([1+x]\,\frac{d}{dx}\left(x - x^2\right)\right) + 5xe^x\left(x - x^2\right) = 5x^2e^x - 5x^3e^x + 4x + 1.$$

Thus we ended up with

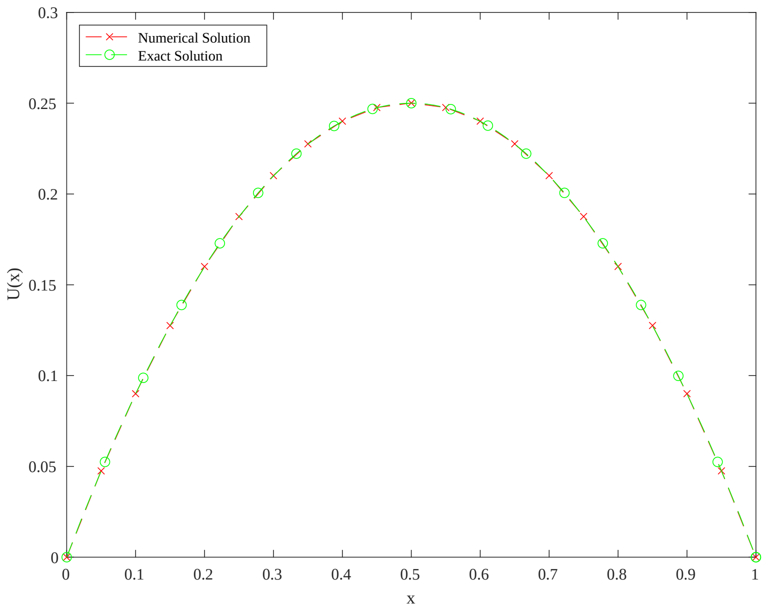
$$f(x) = 5x^2e^x\,(1 - x) + 4x + 1.$$

The following Matlab code solves the system (8) for the proposed example. Even though the code is flexible to handle nonuniform spacing, in this example we use a uniform grid:

```
%------------------------------------------------------------------
% FEM code to solve the BVP (-ku')' + pu = f  w/ vanishing Dirichlet BCs
%------------------------------------------------------------------

%Interval endpoints and number of subintervals
a = 0;
b = 1;
n = 20;

x = linspace(a,b, n+1);    %uniform mesh

%functions to be called
k_funct  = @(x) 1+x;
p_funct  = @(x) 5 .*x .* exp(x);
f_funct  = @(x) (5 .* (x.^2) .* exp(x)) .* (1-x) + 4 .*x + 1;

M = MassMatD0(x, p_funct);       %call mass matrix
K = StiffMatD0(x, k_funct);      %call stiffness matrix
F = LoadVecD0(x, f_funct);       %call load vector

U = (M+K)\F;              %Solve (M+K)U = F
U_full = [0; U; 0];       %extend solution to include BCs
```

```
%Plot results:
plot(x, U_full, "r--x")
hold on
funct = @(x) x - x.^2;      %closed-form solution
fplot(funct, [0,1], "g--o")
ylabel('U(x)')
xlabel('x')
legend("Numerical Solution", "Exact Solution", 'Location','northwest')
exportgraphics(gcf,'FEM_Prob1.pdf')
close
```

The following plot shows an excellent fit between the numerical and the exact solution. All the work was worth it!



□

Problem 2. *Write the expression $\nabla \cdot (\kappa \nabla u)$ explicitly in terms of partial derivatives and show that*

$$\nabla \cdot (\kappa \nabla u) = \kappa \Delta u + \nabla \kappa \cdot \nabla u.$$

Proof. *We work over $\mathbb{R}^n$ and use the notation $\partial_k := \partial / \partial x_k$. Then, expanding the LHS, we have*

$$\begin{aligned}
\nabla \cdot (\kappa \nabla u) &= \begin{bmatrix} \partial_1 \\ \vdots \\ \partial_n \end{bmatrix} \cdot \kappa \begin{bmatrix} \partial_1 u \\ \vdots \\ \partial_n u \end{bmatrix} \\
&= \partial_1 (\kappa \partial_1 u) + \cdots + \partial_n (\kappa \partial_n u) \\
&= \kappa \partial_1^2 u + \partial_1 \kappa \partial_1 u + \cdots + \kappa \partial_n^2 u + \partial_n \kappa \partial_n u \\
&= \kappa \left(\partial_1^2 + \cdots + \partial_n^2 \right) u + \begin{bmatrix} \partial_1 \kappa \\ \vdots \\ \partial_n \kappa \end{bmatrix} \cdot \begin{bmatrix} \partial_1 u \\ \vdots \\ \partial_n u \end{bmatrix} \\
&= \kappa \Delta u + \nabla \kappa \cdot \nabla u.
\end{aligned}$$

□

Problem 3. Show that the following two systems are equivalent when μ and λ are constants:

- *System 1:*

$$
\begin{aligned}
-\nabla \cdot \sigma &= f \quad \text{in} \quad \Omega, \\
\sigma &= 2\mu\epsilon + \lambda tr(\epsilon) I \\
\epsilon &= \frac{1}{2}(\nabla u + \nabla u^\top).
\end{aligned}
$$

- *System 2:*

$$
\begin{aligned}
-(2\mu + \lambda)\frac{\partial^2 u_1}{\partial x^2} - \mu\frac{\partial^2 u_1}{\partial y^2} - (\mu + \lambda)\frac{\partial^2 u_2}{\partial y \partial x} &= f_1. \\
-(\mu + \lambda)\frac{\partial^2 u_1}{\partial y \partial x} - \mu\frac{\partial^2 u_2}{\partial x^2} - (2\mu + \lambda)\frac{\partial^2 u_2}{\partial y^2} &= f_2.
\end{aligned}
$$

Proof. *The divergence operator $\nabla \cdot ()$ is a rank-lowering operation on tensors. In particular, when applied to the matrix σ (rank-2) we end up with a vector (rank-1). The latter vector, explicitly, has components that are the divergences of the rows of the original matrix σ. Thus,*

$$
-\nabla \cdot \sigma = -\begin{bmatrix} \partial_x \\ \partial_y \end{bmatrix} \cdot \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix} = -\begin{bmatrix} \partial_x \sigma_{11} + \partial_y \sigma_{12} \\ \partial_x \sigma_{21} + \partial_y \sigma_{22} \end{bmatrix}. \tag{10}
$$

In order to expand this expression and show its equivalence to the LHS of System 2, we need to write the matrix σ explicitly; the first order of business then is to write ϵ explicitly in matrix form. To accomplish the latter we first note that, since now $u : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$, the gradients are Jacobians:

$$
\nabla u = \begin{bmatrix} \partial_x u_1 & \partial_y u_1 \\ \partial_x u_2 & \partial_y u_2 \end{bmatrix}, \qquad \nabla u^\top = \begin{bmatrix} \partial_x u_1 & \partial_x u_2 \\ \partial_y u_1 & \partial_y u_2 \end{bmatrix}. \tag{11}
$$

Thus,

$$
\begin{aligned}
\epsilon &= \frac{1}{2}\left(\begin{bmatrix} \partial_x u_1 & \partial_y u_1 \\ \partial_x u_2 & \partial_y u_2 \end{bmatrix} + \begin{bmatrix} \partial_x u_1 & \partial_x u_2 \\ \partial_y u_1 & \partial_y u_2 \end{bmatrix}\right) \\
&= \begin{bmatrix} \partial_x u_1 & \frac{1}{2}\left(\partial_x u_2 + \partial_y u_1\right) \\ \frac{1}{2}\left(\partial_y u_1 + \partial_x u_2\right) & \partial_y u_2 \end{bmatrix},
\end{aligned}
$$

and

$$
tr(\epsilon) = \epsilon_{11} + \epsilon_{22} = \partial_x u_1 + \partial_y u_2.
$$

Hence, for constant λ and μ, we get

$$
\begin{aligned}
\sigma &= 2\mu\epsilon + \lambda tr(\epsilon) I \\
&= 2\mu \begin{bmatrix} \partial_x u_1 & \frac{1}{2}\left(\partial_x u_2 + \partial_y u_1\right) \\ \frac{1}{2}\left(\partial_y u_1 + \partial_x u_2\right) & \partial_y u_2 \end{bmatrix} + \begin{bmatrix} \lambda\left(\partial_x u_1 + \partial_y u_2\right) & 0 \\ 0 & \lambda\left(\partial_x u_1 + \partial_y u_2\right) \end{bmatrix} \\
&= \begin{bmatrix} \partial_x u_1 (2\mu + \lambda) + \lambda \partial_y u_2 & \mu\left(\partial_x u_2 + \partial_y u_1\right) \\ \mu\left(\partial_y u_1 + \partial_x u_2\right) & \partial_y u_2 (2\mu + \lambda) + \lambda \partial_x u_1 \end{bmatrix}.
\end{aligned}
$$

We now substitute into Eq. (10), one row at a time:

$$
\begin{aligned}
-\partial_x \sigma_{11} - \partial_y \sigma_{12} &= -(2\mu + \lambda)\, \partial_x^2 u_1 - \lambda \partial_{yx} u_2 - \mu \partial_{xy} u_2 - \mu \partial_y^2 u_1 \\
&= -(2\mu + \lambda)\, \partial_x^2 u_1 - (\mu + \lambda)\, \partial_{xy} u_2 - \mu \partial_y^2 u_1
\end{aligned}
$$

$$
\begin{aligned}
-\partial_x \sigma_{21} - \partial_y \sigma_{22} &= -\mu \partial_{yx} u_1 - \mu \partial_x^2 u_2 - (2\mu + \lambda)\, \partial_y^2 u_2 - \lambda \partial_{xy} u_1 \\
&= -(\mu + \lambda)\, \partial_{xy} u_1 - \mu \partial_x^2 u_2 - (2\mu + \lambda)\, \partial_y^2 u_2.
\end{aligned}
$$

(In these calculations we used the commutativity of the mixed partials; $\partial_{xy} = \partial_{yx}$.) Hence, since $f = [f_1\ f_2]^\top$, we conclude that the two systems are identical. □

Problem 4. Let Ω be the unit square: $\Omega = (0,1) \times (0,1)$. Verify that

$$-\int_\Omega v\Delta u = \int_\Omega \nabla v \cdot \nabla u - \int_{\partial\Omega} v\frac{\partial u}{\partial n}, \tag{12}$$

for

$$u(x,y) = 1 + xy^2, \quad v(x,y) = x + xy.$$

Proof. *We start by tackling the LHS; first note that*

$$\Delta u = \partial_x^2 u + \partial_y^2 u = 0 + 2x = 2x.$$

Then,

$$\begin{aligned}
-\int_\Omega v\Delta u &= -\int_0^1\int_0^1 (x+xy)(2x)\,dx\,dy \\
&= -\int_0^1\int_0^1 2x^2(1+y)\,dx\,dy \\
&= -\int_0^1 \frac{2}{3}x^3\Big|_0^1 (1+y)\,dy \\
&= -\frac{2}{3}\int_0^1 (1+y)\,dy \\
&= -\frac{2}{3}\left(y + \frac{1}{2}y^2\right)\Big|_0^1 \\
&= -1.
\end{aligned}$$

Now, on to the first term on the RHS:

$$\begin{aligned}
\int_\Omega \nabla v \cdot \nabla u &= \int_0^1\int_0^1 \begin{bmatrix}\partial_x v\\ \partial_y v\end{bmatrix} \cdot \begin{bmatrix}\partial_x u\\ \partial_y u\end{bmatrix} dx\,dy \\
&= \int_0^1\int_0^1 \begin{bmatrix}1+y\\ x\end{bmatrix} \cdot \begin{bmatrix}y^2\\ 2xy\end{bmatrix} dx\,dy \\
&= \int_0^1\int_0^1 \left(y^3 + y^2 + 2x^2 y\right) dx\,dy \\
&= \int_0^1\int_0^1 \left(y^3 + y^2 + y\,\frac{2}{3}x^3\Big|_0^1\right) dy \\
&= \left(\frac{1}{4}y^4 + \frac{1}{3}y^3 + \frac{2}{3}\frac{1}{2}y^2\right)\Big|_0^1 \\
&= \frac{11}{12}.
\end{aligned}$$

For the last term on the RHS of Eq. (12) we must choose an orientation for the boundary $\partial\Omega$; let us choose the "right-handed" orientation (i.e., counterclockwise). We also use the definition of the normal derivative:

$$\frac{\partial u}{\partial n} := \nabla u \cdot n,$$

where n is the (outward-pointing) unit normal vector. Then,

$$\int_{\partial\Omega} v\frac{\partial u}{\partial n} = \int_0^1 v\nabla u\cdot n\,dx\Big|_{y=0} + \int_0^1 v\nabla u\cdot n\,dy\Big|_{x=1} + \int_1^0 v\nabla u\cdot n\,dx\Big|_{y=1} + \int_1^0 v\nabla u\cdot n\,dy\Big|_{x=0}$$

$$
\begin{aligned}
&= \int_0^1 (x+xy)\begin{bmatrix} y^2 \\ 2xy \end{bmatrix} \cdot \begin{bmatrix} 0 \\ -1 \end{bmatrix} dx\Big|_{y=0} + \int_0^1 (x+xy)\begin{bmatrix} y^2 \\ 2xy \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} dy\Big|_{x=1} \\
&+ \int_1^0 (x+xy)\begin{bmatrix} y^2 \\ 2xy \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} dx\Big|_{y=1} + \int_1^0 (x+xy)\begin{bmatrix} y^2 \\ 2xy \end{bmatrix} \cdot \begin{bmatrix} -1 \\ 0 \end{bmatrix} dy\Big|_{x=0} \\
&= \cancelto{0}{\int_0^1 (x+xy)(-2xy)\, dx}\Big|_{y=0} + \int_0^1 (x+xy)\, y^2\, dy\Big|_{x=1} \\
&+ \int_1^0 (x+xy)\,(2xy)\, dx\Big|_{y=1} + \cancelto{0}{\int_1^0 (x+xy)\left(-y^2\right) dy}\Big|_{x=0} \\
&= \int_0^1 \left(y^3+y^2\right) dy + \int_1^0 4x^2\, dx \\
&= \frac{1}{4} + \frac{1}{3} + \frac{4}{3} = \frac{23}{12}.
\end{aligned}
$$

Hence the RHS of Eq. (12) is

$$\frac{11}{12} - \frac{23}{12} = -1,$$

which proves the validity of Eq. (12). □

Problem 5. Let $\sigma : R^2 \longrightarrow R^{2\times 2}$ be smooth. Use the ordinary divergence theorem to show that

$$\int_\Omega \nabla \cdot \sigma = \int_{\partial\Omega} \sigma n. \tag{13}$$

Proof. We recall from Eq. (10) that

$$\nabla \cdot \sigma = \begin{bmatrix} \partial_x \sigma_{11} + \partial_y \sigma_{12} \\ \partial_x \sigma_{21} + \partial_y \sigma_{22} \end{bmatrix}.$$

On the other hand,

$$\sigma n = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix} \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} = \begin{bmatrix} \sigma_{11} n_1 + \sigma_{12} n_2 \\ \sigma_{21} n_1 + \sigma_{22} n_2 \end{bmatrix}.$$

Now, since the integral of a vector-valued function is computed by taking the integral of each component of the function, Eq. (13) yields the following system:

$$\int_\Omega \left(\partial_x \sigma_{11} + \partial_y \sigma_{12}\right) = \int_{\partial\Omega} \left(\sigma_{11} n_1 + \sigma_{12} n_2\right) \tag{14a}$$

$$\int_\Omega \left(\partial_x \sigma_{21} + \partial_y \sigma_{22}\right) = \int_{\partial\Omega} \left(\sigma_{21} n_1 + \sigma_{22} n_2\right). \tag{14b}$$

Then, considering the vectors

$$\sigma_{(1)} = \begin{bmatrix} \sigma_{11} \\ \sigma_{12} \end{bmatrix} \qquad \sigma_{(2)} = \begin{bmatrix} \sigma_{21} \\ \sigma_{22} \end{bmatrix},$$

the above system becomes

$$\int_\Omega \nabla \cdot \sigma_{(1)} = \int_{\partial\Omega} \sigma_{(1)} \cdot n \tag{14c}$$

$$\int_\Omega \nabla \cdot \sigma_{(2)} = \int_{\partial\Omega} \sigma_{(2)} \cdot n. \tag{14d}$$

Both of these integral equations hold by the ordinary Divergence Theorem, thereby demonstrating the validity of Eq. (13). □

Problem 6. Let $\sigma : R^2 \longrightarrow R^{2\times 2}$ and $v : R^2 \longrightarrow R^2$ be smooth. Show that

$$\nabla \cdot (\sigma v) = (\nabla \cdot \sigma^\top) \cdot v + \sigma \cdot \nabla v^\top. \tag{15}$$

Proof. We first expand the LHS:

$$\begin{aligned}
\nabla \cdot (\sigma v) &= \begin{bmatrix} \partial_x \\ \partial_y \end{bmatrix} \cdot \left(\begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \right) \\
&= \begin{bmatrix} \partial_x \\ \partial_y \end{bmatrix} \cdot \begin{bmatrix} \sigma_{11} v_1 + \sigma_{12} v_2 \\ \sigma_{21} v_1 + \sigma_{22} v_2 \end{bmatrix} \\
&= \partial_x \left(\sigma_{11} v_1 + \sigma_{12} v_2 \right) + \partial_y \left(\sigma_{21} v_1 + \sigma_{22} v_2 \right) \\
&= v_1 \left(\partial_x \sigma_{11} + \partial_y \sigma_{21} \right) + v_2 \left(\partial_x \sigma_{21} + \partial_y \sigma_{22} \right) + \sigma_{11} \partial_x v_1 + \sigma_{12} \partial_x v_2 + \sigma_{21} \partial_y v_1 + \sigma_{22} \partial_y v_2.
\end{aligned}$$

Now on to the first term on the RHS:

$$\begin{aligned}
(\nabla \cdot \sigma^\top) \cdot v &= \left(\begin{bmatrix} \partial_x \\ \partial_y \end{bmatrix} \cdot \begin{bmatrix} \sigma_{11} & \sigma_{21} \\ \sigma_{12} & \sigma_{22} \end{bmatrix} \right) \cdot \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \\
&= \begin{bmatrix} \partial_x \sigma_{11} + \partial_y \sigma_{21} \\ \partial_x \sigma_{12} + \partial_y \sigma_{22} \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \\
&= v_1 \left(\partial_x \sigma_{11} + \partial_y \sigma_{21} \right) + v_2 \left(\partial_x \sigma_{21} + \partial_y \sigma_{22} \right).
\end{aligned}$$

On the second term of the RHS the dot product we use is the real Frobenius inner product, which is defined by

$$A \cdot B := A \otimes_F B = \sum_{i,j} a_{ij} b_{ij}.$$

Hence,

$$\begin{aligned}
\sigma \cdot \nabla v^\top &= \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix} \cdot \begin{bmatrix} \partial_x v_1 & \partial_x v_2 \\ \partial_y v_1 & \partial_y v_2 \end{bmatrix} \\
&= \sigma_{11} \partial_x v_1 + \sigma_{12} \partial_x v_2 + \sigma_{21} \partial_y v_1 + \sigma_{22} \partial_y v_2.
\end{aligned}$$

Looking at the color-coded results, we see that the equality (15) does hold. □

Problem 7. Derive the weak form of the following BVP with inhomogeneous boundary conditions:

$$\begin{aligned}
-\nabla \cdot \sigma &= f \quad \text{in} \quad \Omega, \\
\sigma &= 2\mu\epsilon + \lambda tr(\epsilon) I \\
\epsilon &= \frac{1}{2} (\nabla u + \nabla u^\top) \\
u &= g \quad \text{in} \quad \Gamma_1 \\
\sigma n &= h \quad \text{in} \quad \Gamma_2.
\end{aligned}$$

Solution. Consider some test function v. We showed in Problem 6 that

$$\nabla \cdot (\sigma v) = (\nabla \cdot \sigma^\top) \cdot v + \sigma \cdot \nabla v^\top.$$

When σ is symmetric (which is indeed true in our case, since ϵ is symmetric), the above expression becomes

$$\nabla \cdot (\sigma v) = (\nabla \cdot \sigma) \cdot v + \sigma \cdot \nabla v. \tag{16}$$

This holds because, when σ is symmetric,

$$\begin{aligned}\sigma \cdot \nabla v &= \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix} \cdot \begin{bmatrix} \partial_x v_1 & \partial_y v_1 \\ \partial_x v_2 & \partial_y v_2 \end{bmatrix} \\ &= \sigma_{11}\partial_x v_1 + \sigma_{12}\partial_y v_1 + \sigma_{21}\partial_x v_2 + \sigma_{22}\partial_y v_2 \\ &= \sigma_{11}\partial_x v_1 + \sigma_{12}\partial_x v_2 + \sigma_{21}\partial_y v_1 + \sigma_{22}\partial_y v_2 \\ &= \sigma \cdot \nabla v^\top.\end{aligned}$$

But then

$$\sigma \cdot \nabla v = \sigma \cdot \nabla v^\top = \sigma \cdot \epsilon_v,$$

where

$$\epsilon_v := \frac{1}{2}\left(\nabla v + \nabla v^\top\right).$$

By the Divergence Theorem, we have

$$\int_\Omega \nabla \cdot (\sigma v) = \int_{\partial\Omega} (\sigma v) \cdot n$$

and, moreoever, since σ is symmetric,

$$(\sigma v) \cdot n = v \cdot (\sigma n).$$

Thus, combining these results with Eq. (16), we get

$$\int_{\partial\Omega} v \cdot (\sigma n) = \int_\Omega (\nabla \cdot \sigma) \cdot v + \int_\Omega \sigma \cdot \epsilon_v.$$

Hence, going back to our original BVP, if we multiply through by a test function v and integrate over Ω, we have

$$\begin{aligned} -\int_\Omega (\nabla \cdot \sigma) \cdot v &= \int_\Omega f \cdot v \\ \int_\Omega \sigma \cdot \epsilon_v - \int_{\partial\Omega} v \cdot (\sigma n) &= \int_\Omega f \cdot v. \end{aligned}$$

Lastly, since

$$\partial\Omega = \Gamma_1 \amalg \Gamma_2,$$

taking into account the imposed boundary conditions we end up with the weak form of the BVP:

$$\boxed{\int_\Omega \sigma \cdot \epsilon_v - \int_{\Gamma_1} v \cdot (\sigma_g n) - \int_{\Gamma_2} v \cdot h = \int_\Omega f \cdot v} \tag{17}$$

where

$$\begin{aligned}\sigma_g &= 2\mu\epsilon_g + \lambda\, tr(\epsilon_g) I; \\ \epsilon_g &= \frac{1}{2}\left(\nabla g + \nabla g^\top\right).\end{aligned}$$

□

Problem 8. Solve the following heat equation by using the FEM:

$$\begin{aligned} \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} &= x(1-x)\cos t, \quad 0 < x < 1, \quad t > 0, \\ u(0,x) &= 1, \quad 0 < x < 1, \\ u(t,0) &= 0, \quad t > 0, \\ u(t,1) &= 0, \quad t > 0. \end{aligned}$$

Use S_3 as the approximating subspace. Explicitly compute the mass matrix M, the stiffness matrix K, and the load vector $F(t)$. Explicitly set up the system of ODEs and solve it.

Solution. We use the "dot" notation for time-derivatives and "prime" notation for spatial derivatives; moreover we let $I = [0, 1]$. Then, multiplying through by some test function v and integrating, we have

$$\int_I \dot{u}v - \int_I u''v = \int_I fv, \tag{18}$$

with

$$f = f(t,x) := x(1-x)\cos t.$$

Now, from a straightforward application of the product rule,

$$\cancelto{0}{u'v\Big|_0^1} = \int_I (u'v)' = \int_I u''v + \int_I u'v'.$$

Thus, plugging back into Eq. (18), we get

$$\boxed{\int_I \dot{u}v + \int_I u'v' = \int_I fv} \tag{19}$$

This is the weak form of the original Heat Equation. We then recall the ansatz (6); since we are now using S_3 as the approximating subspace, we will only be using three hat-functions. Moreoever, the coefficients U_i now depend on time. Thus we have

$$^{(3)}u(t,x) = \sum_{i=1}^{3} U_i(t)\varphi_i(x).$$

Plugging this into our weak form (19) and substituting φ_i's for v's, we have

$$\int_I \left(\sum_{j=1}^{3} U_j\varphi_j\right)^{\cdot} \varphi_i + \int_I \left(\sum_{j=1}^{3} U_j\varphi_j\right)' \varphi_i' = \int_I f\varphi_i$$

$$\sum_{j=1}^{3} \dot{U}_j \int_I \varphi_j\varphi_i + \sum_{j=1}^{3} U_j \int_I \varphi_j'\varphi_i' = \int_I f\varphi_i, \qquad \text{for } i = 1, 2, 3.$$

This last expression is of the form

$$\boldsymbol{M}\dot{\boldsymbol{U}} + \boldsymbol{K}\boldsymbol{U} = \boldsymbol{F}, \tag{20}$$

where $\boldsymbol{M}$ and $\boldsymbol{K}$ are, respectively, the mass and stiffness matrices we defined before in Problem 1, except that now $p(x) = k(x) \equiv 1$. We also note that this time the load vector, $\boldsymbol{F}$, does depend on time. Using uniform grid-spacing $h \equiv 1/(3+1) = 1/4$ and plugging back into the expressions we derived on Problem 1, the system takes the form

$$\underbrace{\begin{bmatrix} 1/6 & 1/24 & 0 \\ 1/24 & 1/6 & 1/24 \\ 0 & 1/24 & 1/6 \end{bmatrix}}_{\boldsymbol{M}} \underbrace{\begin{bmatrix} \dot{U}_1 \\ \dot{U}_2 \\ \dot{U}_3 \end{bmatrix}}_{\dot{\boldsymbol{U}}} + \underbrace{\begin{bmatrix} 8 & -4 & 0 \\ -4 & 8 & -4 \\ 0 & -4 & 8 \end{bmatrix}}_{\boldsymbol{K}} \underbrace{\begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix}}_{\boldsymbol{U}} = \underbrace{\cos t \begin{bmatrix} 17/384 \\ 23/384 \\ 17/384 \end{bmatrix}}_{\boldsymbol{F}}.$$

Hence we have reduced a PDE problem to a simple ODE problem, which we can now solve using any of the ODE methods we have previously studied. Since we are only using three hat functions to approximate a solution over the entire interval $[0, 1]$, we cannot realitistically expect to get a very smooth solution. Thus, since accuracy is not much of a concern for this exercise, we don't need some highly accurate method like RK4. We shall instead implement Backward Euler, so that Eq. (20) is rewritten as

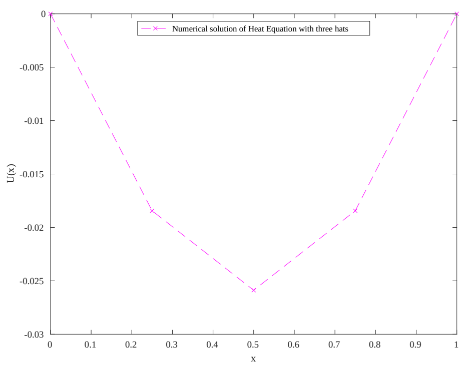
$$(\boldsymbol{M} + \Delta t\boldsymbol{K})\,\boldsymbol{U}^{n+1} = \boldsymbol{M}\boldsymbol{U}^{n} + \Delta t\boldsymbol{F}^{n+1}, \tag{21}$$

where, per usual notation, the superscripts denote the time step; i.e., $\boldsymbol{U}^n = \boldsymbol{U}(t_0 + n\Delta t)$. The following Matlab script implements the BackWard Euler method:

```
%Backward Euler solution to the Heat Eq using only three hat functions
%(result expected to look rough!)

m = 3;
h = 1/4;
dt = h^2/2;      %time-stepping (can be larger since we're using an implicit scheme)

%Generate arrays:
M = zeros(m);
K = zeros(m);
U_0 = ones(m,1);     %initial conditions
rhs = zeros(m,1);    %initialize rhs of Eq
```

```
for i = 1:m
    M(i,i) = 1/6;
    K(i,i) = 8;
    if i ~= m
        M(i,i+1) = 1/24;
        M(i+1,i) = M(i,i+1);
        K(i,i+1) = -4;
        K(i+1,i) = K(i,i+1);
    end
end

Mat = M + dt*K;

f_vec = [17/384; 23/384; 17/384];
f = @(t) cos(t);

%-----------------------------------------
%         BACKWARD EULER CODE
%-----------------------------------------
it_max = 500;       %max number of iterations allowed
tol = 1e-5;         %tolerance allowed
it = 0;

for n = 1 : it_max
    it = it +1;
    rhs = M * U_0 + dt * f((n+1)*dt) * f_vec;
    U = Mat\rhs;

    if norm(U - U_0) <= tol
        disp(['It took ', num2str(it), ' iterations for the solution to converge.'])
        break
    elseif  it == it_max
      disp('No convergence; max number of iterations reached.')
    end

    U_0 = U;  %update U_0 value for next iteration
end
%---------------------------------------------
%       END OF BACKWARD EULER CODE
%---------------------------------------------

%extend solution to include boundaries
U = [0; U; 0];

x = linspace(0,1,m+2);

%Plot results:
plot(x,U, "r--x")
ylabel('U(x)')
xlabel('x')
legend("Numerical solution of Heat Equation with three hats", 'Location','north')
exportgraphics(gcf,'BE_Heateq_S_3.pdf')
close
```

The code reaches the desired tolerance after 103 iterations and outputs the following plot:



As expected, the solution is not very smooth-looking, since we are only using three φ_i's over the entire intreval $[0, 1]$. However, it does showcase the power of using FEM for the space discretization, since had we used only three interior points for a Finite Differences implementation, the results would look a lot worse! □