Math 5601 Final Project

Jacob Hauck

December 4, 2023

Consider the following second-order ODE with Dirichlet boundary conditions:

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(c(x)\frac{\mathrm{d}u(x)}{\mathrm{d}x}\right) = f(x), \qquad a \leq x \leq b, \tag{1}$$

$$u(a) = g_a, \quad u(b) = g_b. \tag{2}$$

Problem 1.

Consider the second-order ODE (1). Multiplying by $v \in H^1([a,b])$ and integrating by parts gives

$$\int_a^b fv = c(b)u'(b)v(b) - c(a)u'(a)v(a) - \int_a^b cu'v. \tag{3}$$

(a) Suppose we have the boundary conditions

$$u'(a) = p_a, \qquad u(b) = g_b. \tag{4}$$

Equation (3) still holds, and we can impose the condition $v(b) = 0$ because we already know that $u(b) = p_b$. Since $u'(a) = p_a$, equation (3) becomes

$$\int_a^b fv = -c(a)p_a v(a) - \int_a^b cu'v' \tag{5}$$

for all $v \in H^1([a,b])$ such that $v(b) = 0$, which is our weak formulation of (1) with the given boundary conditions.

(b) Suppose we have the boundary conditions

$$u'(a) = p_a, \qquad u'(b) + q_b u(b) = p_b. \tag{6}$$

Equation (3) still holds. Since $u'(b) = p_b - q_b u(b)$, and $u'(a) = p_a$, we get

$$\int_a^b fv = c(b)(p_b - q_b u(b))v(b) - c(a)p_a v(a) - \int_a^b cu'v' \tag{7}$$

for all $v \in H^1([a,b])$, which is our weak formulation of (1) with the given boundary conditions.

(c) Suppose we have the boundary conditions

$$u'(a) = p_a, \qquad u'(b) = p_b. \tag{8}$$

Equation (3) still holds. Since $u'(a) = p_a$, and $u'(b) = p_b$, we get

$$\int_a^b fv = c(b)p_b v(b) - c(a)p_a v(a) - \int_a^b cu'v' \tag{9}$$

for all $v \in H^1([a,b])$, which is our weak formulation of (1) with the given boundary conditions.

We note that solutions of this formulation are not unique. Indeed, if $u \in H^1([a,b])$ satisfies (9) for all $v \in H^1([a,b])$, then so does $u + \alpha$, where $\alpha \in \mathbf{R}$ is any real number because $(u+\alpha)' = u'$ regardless of what α is, and the weak formulation depends only on u'.

Problem 2.

Consider the Poisson equation

$$\nabla \cdot (c\nabla u) = f \text{ in } D. \tag{10}$$

Using integration by parts, we have

$$\int_D fv = \int_D \nabla \cdot (c\nabla u)v = \int_{\partial D} cv\nabla u \cdot n \, \mathrm{d}S - \int_D c\nabla u \cdot \nabla v, \tag{11}$$

where $\mathrm{d}S$ is the surface measure on ∂D, and $v \in H^1\left(\overline{D}\right)$.

(a) Suppose that we have the boundary condition

$$u = g \text{ on } \partial D. \tag{12}$$

Equation (11) still holds. Since we know the value of u on ∂D, we can set $v = 0$ on ∂D. Then we get

$$\int_D fv = -\int_D c\nabla u \cdot \nabla v \tag{13}$$

for all $v \in H^1\left(\overline{D}\right)$ such that $v = 0$ on ∂D, which is our weak formulation of (10) with the given boundary condition.

(b) Suppose that we have the boundary condition

$$\nabla u \cdot n + qu = p \text{ on } \partial D, \tag{14}$$

where n is the outward unit normal vector to ∂D, and p and q are functions on ∂D. Equation (11) still holds. Since $\nabla u \cdot n = p - qu$ on ∂D, it follows that

$$\int_D fv = \int_{\partial D} cv(p - qu) \, \mathrm{d}S - \int_D c\nabla u \cdot \nabla v \tag{15}$$

for all $v \in H^1\left(\overline{D}\right)$, which is our weak formulation of (10) with the given boundary condition.

Problem 3.

If $u \in C^2[a,b]$, then

$$\|u - I_h u\|_\infty \leq \frac{1}{8}h^2\|u''\|_\infty, \tag{16}$$

$$\|(u - I_h u)'\|_\infty \leq \frac{1}{2}h\|u''\|_\infty. \tag{17}$$

Proof. Consider the interval $[x_i, x_{i+1}]$, where $1 \leq i \leq N$. Restricted to this interval, $I_h u$ is the degree-1 Lagrange polynomial interpolation of u on with nodes x_i and x_{i+1}. By the error formula for Lagrange polynomial approximation in the slides,

$$u(x) - I_h u(x) = \frac{f''(\xi(x))(x - x_i)(x - x_{i+1})}{2} \tag{18}$$

for some $\xi(x) \in [x_i, x_{i+1}]$. Then

$$|u(x) - I_h u(x)| \leq \|f''\|_\infty \cdot \frac{1}{2}(x - x_i)(x_{i+1} - x). \tag{19}$$

The function $g(x) = (x - x_i)(x_{i+1} - x)$ is a downward-opening parabola, so it achieves maximum halfway between its roots x_i and x_{i+1}. Therefore,

$$|u(x) - I_h u(x)| \leq \|f''\|_\infty \cdot \frac{\left(\frac{x_i + x_{i+1}}{2} - x_i\right)\left(x_{i+1} - \frac{x_i + x_{i+1}}{2}\right)}{2} \tag{20}$$

$$= \|f''\|_\infty \frac{(x_{i+1} - x_i)^2}{8} = \frac{h^2}{8}\|f''\|_\infty. \tag{21}$$

Since this holds for all $x \in [x_i, x_{i+1}]$ and all $1 \leq i \leq N$, it holds for all $x \in [a, b]$. Therefore, the inequality (16) follows.

Let $1 \leq i \leq N$, and let $x \in (x_i, x_{i+1})$. By Taylor's Theorem,

$$u(x_i) = u(x) + (x_i - x)u'(x) + \frac{1}{2}(x_i - x)^2 u''(\xi(x_i)) \tag{22}$$

$$u(x_{i+1}) = u(x) + (x_{i+1} - x)u'(x) + \frac{1}{2}(x_{i+1} - x)^2 u''(\xi(x_{i+1})) \tag{23}$$

for some $\xi(x_i), \xi(x_{i+1}) \in [x_i, x_{i+1}]$. Then

$$u(x_{i+1}) - u(x_i) = (x_{i+1} - x_i)u'(x) + \frac{1}{2}(x_{i+1} - x)^2 u''(\xi(x_{i+1})) - \frac{1}{2}(x_i - x)^2 u''(\xi(x_i)). \tag{24}$$

Since $I_h u(x) = \frac{u(x_{i+1}) - u(x_i)}{x_{i+1} - x_i}(x - x_i) + u(x_i)$ for $x \in (x_i, x_{i+1})$, it follows that $(I_h u')(x) = \frac{u(x_{i+1}) - u(x_i)}{x_{i+1} - x_i}$ for $x \in (x_i, x_{i+1})$. Thus,

$$(u - I_h u)'(x) = u'(x) - (I_h u)'(x) = u'(x) - \frac{u(x_{i+1} - u(x_i))}{x_{i+1} - x_i} \tag{25}$$

$$= \frac{(x_i - x)^2}{2(x_{i+1} - x_i)} u''(\xi(x_i)) - \frac{(x_{i+1} - x)^2}{2(x_{i+1} - x_i)} u''(\xi(x_{i+1})) \tag{26}$$

if $x \in (x_i, x_{i+1})$. Taking absolute values on both sides gives

$$|(u - I_h u)'(x)| \leq \frac{1}{2(x_{i+1} - x_i)} \left[(x_i - x)^2 |u''(\xi(x_i))| + (x_{i+1} - x)^2 |u''(\xi(x_{i+1}))|\right] \tag{27}$$

$$\leq \frac{1}{2h}\left[(x_i - x)^2 + (x_{i+1} - x)^2\right] \|u''\|_\infty \tag{28}$$

$$= \frac{1}{2h} g(x) \|u''\|_\infty, \tag{29}$$

where $g(x) = (x_i - x)^2 + (x_{i+1} - x)^2$. We note that $g'(x) = 4x - 2(x_{i+1} + x_i)$, so g achieves a maximum on $[x_i, x_{i+1}]$ when $g'(x) = 0$, that is, when $x = \frac{x_{i+1} + x_i}{2}$, or else when $x \in \{x_i, x_{i+1}\}$, by the Extreme Value Theorem. If $x \in \{x_i, x_{i+1}\}$, then $g(x) = h^2$, and if $x = \frac{x_i + x_{i+1}}{2}$, then $g(x) = \frac{h^2}{2}$. Therefore, the maximum of g on $[x_i, x_{i+1}]$ is h^2, and

$$|(u - I_h u)'(x)| \leq \frac{h}{2}\|u''\|_\infty \tag{30}$$

if $x \in (x_i, x_{i+1})$. Since i was arbitrary, this inequality holds for all $x \in [a, b]$ except at the nodes $\{x_i\}$ where $I_h u$ is potentially not differentiable. The L^∞ norm $\|\cdot\|_\infty$ does not depend on the value of a function at finitely many points, so it follows that

$$\|(u - I_h u)'\|_\infty \leq \frac{1}{2} h \|f''\|_\infty, \tag{31}$$

as desired. □

Problem 4.

Consider the weak formulation of

$$\nabla \cdot (c\nabla u) = f \text{ in } D, \qquad u = g \text{ on } \partial D \tag{32}$$

derived in problem 2 (a):

$$\int_D fv = -\int_D c\nabla u \cdot \nabla v \tag{33}$$

for all $v \in H^1\left(\overline{D}\right)$ such that $v = 0$ on ∂D. Suppose that we have basis functions $\{\phi_i\}_{i=1}^{N+1}$ for a finite element space U_h on $\overline{D}$. To approximate a solution of the weak formulation, we approximate H^1 by U_h. Thus, we want to find $u \in U_h$ such that (33) holds for all $v \in U_h$.

By the linearity of the problem and the fact that $U_h = \operatorname{span}\{\phi_i\}$, this is equivalent to (33) being true for $v = \phi_i$, for $i = 1, \ldots, N+1$. Since we want $u \in U_h$, there exist coefficients u_j such that

$$u = \sum_{j=1}^{N+1} u_j \phi_j. \tag{34}$$

Hence, we need

$$\int_D f\phi_i = -\int_D c\nabla\left(\sum_{j=1}^{N+1} u_j\phi_j\right) \cdot \nabla\phi_i \tag{35}$$

for all $i = 1, \cdots, N+1$. Using the linearity of ∇ and rearranging terms, this is equivalent to

$$\sum_{j=1}^{N+1} u_j \left[-\int_D c\nabla\phi_j \cdot \nabla\phi_i\right] = \int_D f\phi_i \tag{36}$$

for all $i = 1, \ldots, N+1$. If we set

$$A_{ij} = -\int_D c\nabla\phi_j \cdot \nabla\phi_i, \qquad b_i = \int_D f\phi_i, \qquad X_j = u_j, \tag{37}$$

then this is equivalent to the linear system $AX = b$.

Problem 5.

Let A be a nonsingular, lower-triangular matrix; that is, $i < j$ implies that $A_{ij} = 0$. Then A^{-1} is also lower-triangular.

Proof. We use induction on the size of the matrix. All 1×1 matrices are trivially lower-triangular, so the base case holds. Now suppose that the claim is true for all matrices of size $n \times n$, where $n \geq 1$.

Let A be a nonsingular, $(n+1) \times (n+1)$, lower-triangular matrix. Then every entry but the last entry of the last column of A is zero by the lower-triangular condition. That is, we can write A in block matrix form as

$$A = \begin{bmatrix} B & 0 \\ c & d \end{bmatrix}, \tag{38}$$

where B is a $n \times n$ matrix, c is a $1 \times n$ row vector, and d is a scalar. Since $A_{ij} = B_{ij}$ if $i, j \leq n$, it follows that B is also lower-triangular. Furthermore, B must be nonsingular.

Indeed, suppose for the sake of contradiction that B is singular. Then its rows $\{B_1, \cdots, B_n\}$ are linearly dependent. That is, there exist $\alpha_1, \ldots, \alpha_n$ not all zero such that

$$\alpha_1 B_1 + \cdots + \alpha_n B_n = 0. \tag{39}$$

Let $\{A_1, \cdots, A_n, A_{n+1}\}$ denote the rows of A. Then $A_i = \begin{bmatrix} B_i & 0 \end{bmatrix}$ for $1 \leq i \leq n$. Hence,

$$\alpha_1 A_1 + \cdots + \alpha_n A_n = 0 \tag{40}$$

as well. This implies that the rows of A are linearly dependent, which contradicts the nonsingularity of A.

Therefore, B is a nonsingular, $n \times n$, lower-triangular matrix, and the induction hypothesis implies that B^{-1} is lower-triangular.

In addition, $d \neq 0$ because $d = 0$ implies that $\det(A) = 0$ upon expansion by cofactors on the last column of A, which contradicts the nonsingularity of A.

We now observe that

$$A \begin{bmatrix} B^{-1} & 0 \\ -cB^{-1}d^{-1} & d^{-1} \end{bmatrix} = \begin{bmatrix} B & 0 \\ c & d \end{bmatrix} \begin{bmatrix} B^{-1} & 0 \\ -cB^{-1}d^{-1} & d^{-1} \end{bmatrix} = \begin{bmatrix} I_{n \times n} & 0 \\ cB^{-1} - cB^{-1}d^{-1}d & 1 \end{bmatrix} = I_{(n+1)\times(n+1)}, \tag{41}$$

so

$$A^{-1} = \begin{bmatrix} B^{-1} & 0 \\ -cB^{-1}d^{-1} & d^{-1} \end{bmatrix}. \tag{42}$$

Then A^{-1} is lower-triangular because B^{-1} is lower triangular. Hence, the inverse of any nonsingular, lower-triangular matrix is also lower-triangular by induction. □

Problem 6.

Let

$$A = \begin{bmatrix} \kappa & \lambda \\ \lambda & \mu \end{bmatrix} \tag{43}$$

be a positive definite matrix. Then the Jacobi method for $Ax = b$ converges.

Proof. We recall from the slides that the Jacobi method is the iteration

$$x^{(k+1)} = -D^{-1}Nx^{(k)} + D^{-1}b, \tag{44}$$

where D is the diagonal of A, and N is the off-diagonal of A. This iteration converges if and only if $\rho(-D^{-1}N) < 1$. In this case,

$$-D^{-1}N = -\begin{bmatrix} 0 & \frac{\lambda}{\mu} \\ \frac{\lambda}{\kappa} & 0 \end{bmatrix}, \tag{45}$$

so any eigenvalue ρ of $-D^{-1}N$ satisfies $\rho^2 - \frac{\lambda^2}{\kappa\mu} = 0$. Therefore $|\rho| < 1$ if and only if $\lambda^2 < \kappa\mu$, or $\kappa\mu - \lambda^2 > 0$. Since $\kappa\mu - \lambda^2 = \det(A)$, and the positive definiteness of A implies that $\det(A) > 0$, it follows that $\rho(-D^{-1}N) < 1$, and the Jacobi method converges. □

Problem 7.

(a)

(b)

Input: A symmetric, positive-definite, $n \times n$ matrix A
Input: A symmetric, positive-definite, $n \times n$ matrix M that is easy to invert (the preconditioner)
Input: A vector b of length n
Input: Initial guess $x^{(0)}$ for the solution of $Ax = b$
Input: Residual tolerance $\varepsilon > 0$
Output: Approximate solution x of $Ax = b$
`// Initialization`
$r^{(0)} \leftarrow b - Ax^{(0)}$;
$d^{(0)} \leftarrow M^{-1}r^{(0)}$;
$k \leftarrow 0$;
`// Iteration`
while $\left\|r^{(k)}\right\| \geq \varepsilon$ **do**
  `// Update` $x^{(k)}$
  $\alpha^{(k)} \leftarrow \frac{\left(r^{(k)}\right)^T M^{-1} r^{(k)}}{\left(d^{(k)}\right)^T A d^{(k)}}$;
  $x^{(k+1)} \leftarrow x^{(k)} + \alpha^{(k)} d^{(k)}$;
  `// Update search direction and residual`
  $r^{(k+1)} \leftarrow r^{(k)} - \alpha^{(k)} A d^{(k)}$;
  $\beta^{(k+1)} \leftarrow \frac{\left(r^{(k+1)}\right)^T M^{-1} r^{(k+1)}}{\left(r^{(k)}\right)^T M^{-1} r^{(k)}}$;
  $d^{(k+1)} \leftarrow M^{-1} r^{(k+1)} + \beta^{(k+1)} d^{(k)}$;
end

Problem 8.

(a) The main implementation of the finite element method for our second-order BVP is found in `fem_dirichlet_1d.m`, and copied below. In order to avoid code duplication, this function takes the various data determining the equation as parameters, namely, a, b, g_a, g_b, f, and c, as well as the number of points in the mesh N. Most importantly, the method used for numerical integration and the solving of a linear system are passed as parameters as well.

```
function [v_x, v_u] = fem_dirichlet_1d(a, b, ga, gb, c, f, n, integrator, solver)

% Initialization
h = (b - a) / n;

v_x = linspace(a, b, n + 1);

m_A = sparse(n + 1, n + 1);
v_b = zeros(n + 1, 1);

% Compute integral of c(x) over each element
v_c_int = zeros(n, 1);
for j = 1:n
    v_c_int(j) = integrator(c, v_x(j), v_x(j + 1)) / h^2;
end

% Assemble stiffness matrix (no BC)
for j = 1:n
```

```
    m_A(j + 1, j) = v_c_int(j);
    m_A(j, j + 1) = v_c_int(j);
end

for j = 2:n
    m_A(j, j) = -v_c_int(j - 1) - v_c_int(j);
end

m_A(1, 1) = -v_c_int(1);
m_A(n + 1, n + 1) = -v_c_int(n);

% Assemble load vector (no BC)
for j = 2:n
    v_b(j) = integrator(@(x) f(x) .* (x - v_x(j - 1)) / h, v_x(j - 1), v_x(j));
    v_b(j) = v_b(j) + integrator(@(x) f(x) .* (v_x(j + 1) - x) / h, v_x(j), v_x(j + 1));
end

v_b(1) = integrator(@(x) f(x) .* (x - v_x(1)) / h, v_x(1), v_x(2));
v_b(n + 1) = integrator(@(x) f(x) .* (v_x(n + 1) - x) / h, v_x(n), v_x(n + 1));

% Enforce Dirichlet BC
m_A(1, :) = 0;
m_A(1, 1) = 1;
m_A(n + 1, :) = 0;
m_A(n + 1, n + 1) = 1;

v_b(1) = ga;
v_b(n + 1) = gb;

% Solve
v_u = solver(m_A, v_b)';
```

To apply 4-point Gaussian quadrature, we reuse `quad.m` from Homework 6 (renamed to `myquad.m`). Note that the Guass-Legendre quadrature nodes and weights on $[-1, 1]$ are given by

$$x_1 = ... \tag{46}$$

We implement the use of these nodes and weights in `gauss4_integrator.m`, copied below. Using the MATLAB \ operator in the `solver` parameter is trivial, so we don't need a custom function for it.

```
function result = gauss4_integrator(f, a, b)

nodes = [
    sqrt(3/7 - 2/7 * sqrt(6/5)), ...
    -sqrt(3/7 - 2/7 * sqrt(6/5)), ...
    sqrt(3/7 + 2/7 * sqrt(6/5)), ...
    sqrt(3/7 + 2/7 * sqrt(6/5))
];

weights = [
    (18 + sqrt(30)) / 36, ...
    (18 + sqrt(30)) / 36, ...
    (18 - sqrt(30)) / 36, ...
    (18 - sqrt(30)) / 36
];
```

```
local_nodes = nodes * ((b - a) / 2) + (a + b) / 2;
local_weights = weights * ((b - a) / 2);

result = myquad(f, local_nodes, local_weights);
```

(b) The input and output in the command window of MATLAB required to run the code in part (a) with the desired step sizes can be found in p8_output.txt. The errors at $x = 2$ and $x = 3$ are reproduced in

h	Error at $x = 2$	Error at $x = 3$
$\frac{1}{4}$	0.146906	0.004765
$\frac{1}{8}$	0.073208	0.003406
$\frac{1}{16}$	0.036484	0.003254
$\frac{1}{32}$	0.018204	0.002028
$\frac{1}{64}$	0.009091	0.001116
$\frac{1}{128}$	0.004543	0.000584