Partial Solution
Your Homework Assignment
Finite Differences and Finite Elements:
Getting to Know You

Dianne P. O'Leary

First, a thank you to those who followed the submission instructions.

How to submit your homework: Please submit your work on paper. (Email submission is acceptable only in an emergency.) Put your work in this order:

- A listing of the output produced by your experiments.
- A discussion for Problem 6.
- A listing of each of your Matlab functions. (Remember to document them.)

Grading:

- Problem 1: 0 points (just an exercise worth doing).
- Clear output: 5 points.
- Discussion in Problem 6: 15 points.
- Design and correctness of Matlab programs: 20 points.
- Documentation: 10 points.

Second, I really believe that it is important that a program's output be clear and easy to understand **without additional formatting and annotation**. Scientific computing in the workplace is a team sport, and your program's documentation and output are your means of communicating with colleagues.

Finally, here are the answers to the problems you did plus one or two more.

In this homework, we explore the nuts-and-bolts of finite difference and finite element approximations to a simple problem:

$$-(a(x)u'(x))' + c(x)u(x) = f(x) \text{ for } x \in (0,1)$$

with the functions a, c, and f given and $u(0) = u(1) = 0$.

We will assume that $a(x) \geq a_0$, where a_0 is a positive number, and $c(x) \geq 0$ for $x \in [0,1]$.

In the finite difference approach, we approximate each derivative of u by a finite difference:

$$\begin{aligned} u'(x) &= \frac{u(x) - u(x-h)}{h} + O(h), \\ u''(x) &= \frac{u(x-h) - 2u(x) + u(x+h)}{h^2} + O(h^2). \end{aligned}$$

Problem 1. Let $M = 6$, $a(x) = 1$ and $c(x) = 0$, and write the 4 finite difference equations for u at $x = .2, .4, .6$, and $.8$.

Answer:

$$\frac{1}{h^2}\begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}$$

where $h = 1/5$, $u_j \approx u(jh)$, and $f_j = f(jh)$.

Problem 2. The Matlab function `finitediff1.m`, found on the website, implements the finite difference method for our equation. The inputs are the parameter M and the functions `a`, `c`, and `f` that define the equation. Each of these functions takes a vector of points as input and returns a vector of function values. (The function `a` also returns a second vector of values of a'.) The outputs of `finitediff1.m` are a vector `ucomp` of computed estimates of u at the mesh points `xmesh`, along with the matrix `A` and the right-hand side `g` from which `ucomp` was computed, so that `A ucomp = g`. Add documentation to the function `finitediff1.m` so that a user could easily use it, understand the method, and modify the function if necessary.

Answer: Documentation is posted on the website for the program `finitediff2.m` of Problem 3, which is very similar to `finitediff1.m` but more useful. Please remember that if you use a code like `finitediff1.m` and fail to include the name of the code's author, or at least a reference to the website from which you obtained it, it is plagairism. Similarly, your implementation of `finitediff2.m` should probably include a statement like, "Derived from finitediff1.m by Dianne O'Leary."

Problem 3. Define a central difference approximation to the first derivative by

$$u'(x) \approx \frac{u(x+h) - u(x-h)}{2h}\,.$$

a. Use the Taylor series expansions

$$\begin{aligned} u(x+h) &= u(x) + hu'(x) + \frac{h^2}{2}u''(x) + \frac{h^3}{6}u'''(\xi_1), \\ u(x-h) &= u(x) - hu'(x) + \frac{h^2}{2}u''(x) - \frac{h^3}{6}u'''(\xi_2), \end{aligned}$$

where ξ_1 is some point between x and $x+h$, and ξ_2 is some point between x and $x-h$, to show that the difference between $u'(x)$ and our approximation is $O(h^2)$ if u has a continuous third derivative.

b. Modify the function of Problem 2 to produce a function `finitediff2.m` that uses this approximation in place of the first order approximation.

Answer:

a. From the given equations, we obtain

$$u(x+h) - u(x-h) = 2hu'(x) + \frac{h^3}{6}u'''(\xi_1) + \frac{h^3}{6}u'''(\xi_2),$$

so, rearranging and using the Mean Value Theorem from calculus we obtain

$$u'(x) - \frac{u(x+h) - u(x-h)}{2h} = -\frac{h^2}{3}u'''(\xi)$$

where ξ is some point in the interval $[x-h, x+h]$.

Our other finite difference approximations could be derived in a similar way.

b. See `finitediff2.m` on the website.

In using piecewise linear finite elements, we can express our approximate solution u_h as

$$u_h(x) = \sum_{j=1}^{M-2} u_j \phi_j(x)$$

for some coefficients u_j, where

$$\phi_j(x) = \begin{cases} \frac{x-x_{j-1}}{x_j-x_{j-1}} & x \in [x_{j-1}, x_j] \\ \frac{x-x_{j+1}}{x_j-x_{j+1}} & x \in [x_j, x_{j+1}] \\ 0 & \text{otherwise} \end{cases}$$

Define

$$\begin{aligned} a(u,v) &= \int_0^1 (a(x)u'(x)v'(x) + c(x)u(x)v(x))dx, \\ (f,v) &= \int_0^1 f(x)v(x)dx. \end{aligned}$$

Problem 4.
a. Since the functions ϕ_j form a basis for S_h, any function $v_h \in S_h$ can be written as

$$v_h(x) = \sum_{j=1}^{M-2} v_j \phi_j(x)$$

for some coefficients v_j. Show that if

$$a(u_h, \phi_j) = (f, \phi_j)$$

for $j = 1, \ldots, M-2$, then

$$a(u_h, v_h) = (f, v_h).$$

b. Putting the unknowns u_j in a vector $\mathbf{u}$ we can write the resulting system of equations as $\boldsymbol{A}\mathbf{u} = \mathbf{g}$ where the (j,k) entry in $\boldsymbol{A}$ is $a(\phi_j, \phi_k)$ and the jth entry in $\mathbf{g}$ is (f, ϕ_j). Write this system of equations for $M = 6$, $a(x) = 1$, $c(x) = 0$, and compare with your solution to Problem 1.

Answer: First notice that if α and β are constants and v and z are functions of x, then

$$a(u, \alpha v + \beta z) = \alpha a(u,v) + \beta a(u,z),$$

since we can compute the integral of a sum as the sum of the integrals and then move the constants outside the integrals. Therefore,

$$a(u_h, v_h) \quad = \quad a(u_h, \sum_{j=1}^{M-2} v_j \phi_j)$$

$$
\begin{aligned}
&= \sum_{j=1}^{M-2} v_j a(u_h, \phi_j) \\
&= \sum_{j=1}^{M-2} v_j (f, \phi_j) \\
&= (f, \sum_{j=1}^{M-2} v_j \phi_j) \\
&= (f, v_h).
\end{aligned}
$$

b. We compute

$$
\begin{aligned}
a(\phi_j, \phi_j) &= \int_0^1 (\phi_j'(x))^2 dx \\
&= \int_{(j-1)h}^{(j+1)h} (\phi_j'(x))^2 dx \\
&= 2 \int_{(j-1)h}^{jh} \frac{1}{h^2} dx \\
&= \frac{2}{h}
\end{aligned}
$$

and

$$
\begin{aligned}
a(\phi_j, \phi_{j+1}) &= \int_0^1 \phi_j'(x)\phi_{j+1}'(x) dx \\
&= \int_{jh}^{(j+1)h} \phi_j'(x)\phi_{j+1}'(x) dx \\
&= \int_{jh}^{(j+1)h} \frac{(-1)}{h} \frac{1}{h} dx \\
&= -\frac{1}{h}.
\end{aligned}
$$

So our system becomes

$$
\frac{1}{h} \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}
$$

where u_j is the coefficient of ϕ_j in the representation of u_h and

$$
f_j = \int_0^1 f(x)\phi_j(x) dx = \int_{(j-1)h}^{(j+1)h} f(x)\phi_j(x) dx,
$$

which is h times a weighted average of f over the jth interval. The only difference between the finite difference system and this system is that we have replaced point samples of f by average values. Note that if $a(x)$ is not constant, then the systems look even more different.

Problem 5. Write a function `fe_linear.m` that has the same inputs and outputs as `finitediff1.m` but computes the finite element approximation to the solution using piecewise linear elements. Remember to store $\boldsymbol{A}$ as a sparse matrix.

Answer: See the code posted on the website.

Problem 6. Write a function `fe_quadratic.m` that has the same inputs and outputs as `finitediff1.m` but computes the finite element approximation to the solution using piecewise quadratic elements.

Answer: See the code posted on the website.

Problem 7. Use your four algorithms to solve 7 problems. Compute three approximations for each algorithm and each problem, with the number of unknowns in the problem chosen to be 9, 99, and 999. For each approximation, print $\|\mathbf{u}_{computed} - \mathbf{u}_{true}\|_\infty$ where $\mathbf{u}_{true}$ is the vector of true values at the points jz, where $z = 1/10$, $1/100$, or $1/1000$, respectively.
Discuss the results:

- How easy is it to program each of the four methods? Estimate how much work Matlab does to form and solve the linear systems. (The work to solve the tridiagonal systems should be about $5M$ multiplications, and the work to solve the 5-diagonal systems should be about $11M$ multiplications, so you just need to estimate the work in forming each system.)
- For each problem, note the observed convergence rate r: if the error drops by a factor of 10^r when M is increased by a factor of 10, then the observed convergence rate is r.
- Explain any deviations from the theoretical convergence rate: $r = 1$ and $r = 2$ for the two finite difference implementations, and $r = 2$ and $r = 3$ for the finite element implementations.

Answer: Here are the results:

```
                         PROBLEM 1
Using coefficient functions a(1) and c(1) with true solution u(1)
         Infinity norm of the error at the mesh points
   for various methods and numbers of interior mesh points M
                    M =            9           99          999
1st order finite difference   2.1541e-03   2.1662e-05   2.1662e-07
2nd order finite difference   2.1541e-03   2.1662e-05   2.1662e-07
     Linear finite elements   1.3389e-13   1.4544e-14   1.4033e-13
  Quadratic finite elements   3.1004e-05   3.5682e-09   3.6271e-13

                         PROBLEM 2
Using coefficient functions a(1) and c(2) with true solution u(1)
         Infinity norm of the error at the mesh points
   for various methods and numbers of interior mesh points M
                    M =            9           99          999
```

```
1st order finite difference   1.7931e-03   1.8008e-05   1.8009e-07
2nd order finite difference   1.7931e-03   1.8008e-05   1.8009e-07
     Linear finite elements   6.1283e-04   6.1378e-06   6.1368e-08
  Quadratic finite elements   2.7279e-05   3.5164e-09   1.7416e-12

                        PROBLEM 3
Using coefficient functions a(1) and c(3) with true solution u(1)
         Infinity norm of the error at the mesh points
   for various methods and numbers of interior mesh points M
                    M =            9           99          999
1st order finite difference   1.9405e-03   1.9529e-05   1.9530e-07
2nd order finite difference   1.9405e-03   1.9529e-05   1.9530e-07
     Linear finite elements   4.3912e-04   4.3908e-06   4.3906e-08
  Quadratic finite elements   2.8745e-05   3.5282e-09   3.6134e-13

                        PROBLEM 4
Using coefficient functions a(2) and c(1) with true solution u(1)
         Infinity norm of the error at the mesh points
   for various methods and numbers of interior mesh points M
                    M =            9           99          999
1st order finite difference   1.5788e-02   1.8705e-03   1.8979e-04
2nd order finite difference   3.8465e-03   3.8751e-05   3.8752e-07
     Linear finite elements   1.3904e-03   1.3930e-05   1.3930e-07
  Quadratic finite elements   1.6287e-04   1.9539e-08   1.9897e-12

                        PROBLEM 5
Using coefficient functions a(3) and c(1) with true solution u(1)
         Infinity norm of the error at the mesh points
   for various methods and numbers of interior mesh points M
                    M =            9           99          999
1st order finite difference   1.1858e-02   1.4780e-03   1.5065e-04
2nd order finite difference   3.6018e-03   3.6454e-05   3.6467e-07
     Linear finite elements   8.3148e-04   8.2486e-06   1.2200e-06
  Quadratic finite elements   1.0981e-04   1.6801e-06   2.5858e-06

                        PROBLEM 6
Using coefficient functions a(1) and c(1) with true solution u(2)
         Infinity norm of the error at the mesh points
   for various methods and numbers of interior mesh points M
                    M =            9           99          999
1st order finite difference   8.9200e-02   9.5538e-02   9.6120e-02
2nd order finite difference   8.9200e-02   9.5538e-02   9.6120e-02
     Linear finite elements   8.6564e-02   9.5219e-02   9.6086e-02
  Quadratic finite elements   8.6570e-02   9.5224e-02   9.6088e-02

                        PROBLEM 7
Using coefficient functions a(1) and c(1) with true solution u(3)
         Infinity norm of the error at the mesh points
   for various methods and numbers of interior mesh points M
                    M =            9           99          999
1st order finite difference   1.5702e-01   1.6571e-01   1.6632e-01
2nd order finite difference   1.5702e-01   1.6571e-01   1.6632e-01
     Linear finite elements   1.4974e-01   1.6472e-01   1.6622e-01
  Quadratic finite elements   1.4975e-01   1.6472e-01   1.6622e-01
```

Discussion:

Clearly, the finite difference methods are easier to program and therefore are almost always used when x is a single variable. Finite elements become useful, though, when x has 2 or more components and the shape of the domain is nontrivial.

The bulk of the work in these methods is in function evaluations. We need $O(M)$ evaluations of a, c, and f in order to form each matrix. For finite differences, the constant is close to 1, but `quad` uses many function evaluations per call (on the order of 10), making formation of the finite element matrices about 10 times as expensive.

The experimental rate of convergence should be calculated as the $\log_{10}$ of the successive errors (since we increase the number of mesh points by a factor of 10 each time). There are several departures from the expected rate of convergence:

- `finitediff1` is expected to have a linear convergence rate ($r = 1$), but has $r = 2$ for the first three problems because $a' = 0$ and the approximation is the same as that in `finitediff2`.
- The quadratic finite element approximation has $r = 4$ on Test Problems 1-4, better than the $r = 3$ we might expect. This is called *superconvergence* and happens because we only measured the error at the mesh points, whereas the $r = 3$ result was for the average value of the error over the entire interval.
- Linear finite elements give almost an exact answer to Test Problem 1 at the mesh points (but not between the mesh points). This occurs because our finite element equations demand that

  $$a(u_h, \phi_j) = (u_h', \phi_j') = [-u_h(x_{j-1}) + 2u_h(x_j) - u_h(x_{j+1})]/h = (f, \phi_j),$$

  and our true solution also satisfies this relation.
- In Test Problem 5, the coefficient function a has a discontinuous derivative at $x = 1/3$. The matrix entries computed by the numerical integration routine are not very accurate, so the finite element methods appear to have slow convergence. This can be fixed by extra calls to `quad` so that it never tries to integrate across the discontinuity.
- The “solution” to Test Problem 6 has a discontinuous derivative, and the “solution” to Test Problem 7 is discontinuous. None of our methods compute good approximations, although all of them return a reasonable answer (See Figure 1) that could be mistaken for what we are looking for. The finite difference approximations lose accuracy because their error term depends on u''. The finite element equations were derived from the weak formulation of our problem, and when we used integration by parts, we left off the boundary term that we would have gotten at $x = 2/3$, so our equations are wrong. This is a case of, “Be careful what you ask for.”
- The entries in the finite element matrices are only approximations to the true values, due to inaccuracy in estimation of the integrals. This means that as the mesh size is decreased, we need to reduce the tolerance that we send to `quad` in order to keep the matrix accurate enough.

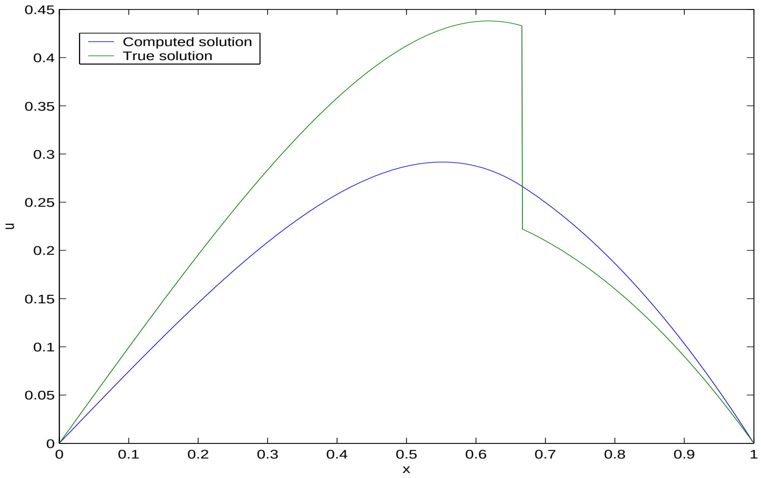

Figure 1: The solution to the 7th test problem

- The theoretical convergence rate only holds down to the round-off level of the machine, so if we took even finer meshes (much larger M), we would fail to see the expected rate.