# Assignment #3

## Due Monday, 10 February 2025, at the start of class

Please read sections 2.4–2.16 from the textbook.[1]

**Problem P12.** **(a)** Set up a finite difference method for the general linear, second-order, Dirichlet, two-point boundary value problem:

$$u''(x) + p(x)\,u'(x) + q(x)\,u(x) = f(x), \quad u(a) = \alpha, \quad u(b) = \beta. \tag{1}$$

Here $p(x), q(x), f(x)$ are all arbitrary functions. Use approximation (1.3), namely the centered finite difference $D_0$, for the $u'$ term, and the usual centered formula $D^2$ for the $u''$ term. State the $A$ and $F$, which appear in your linear system $AU = F$, assuming $m$ unknowns $U_1, \dots, U_m$, on an equally-spaced grid with $m$ interior points.

**(b)** Implement this method. A recommended approach is to modify an existing, tested solver, for example:

`bueler.github.io/nade/assets/codes/S25/second.m`

The recommended Matlab form of your new solver is a function with signature something like

```
function [x, U] = generalbvp(p, q, f, a, b, alpha, beta, m)
```

Turn in a listing of the code itself, but please do part **(c)** to test and debug it before posting your final version.

**(c)** Use the method of manufactured solutions to debug and test your code, especially to verify that you are handling $p(x)$ and $q(x)$ correctly. To avoid making things *too* complicated, fix the simple values $a = 0, b = 1, \alpha = 0, \beta = 0$ for all testing. Generate a first exact solution in the case where $q(x) = 0$ but $p(x)$ is non-zero and non-constant. Then generate a second exact solution in the case where $p(x) = 0$ but $q(x)$ is non-zero and non-constant. (*In these exact solutions you will* choose $u(x)$ *as a function which satisfies the boundary conditions, and* choose $p(x)$ *or* $q(x)$*, whichever is non-zero, and then you will differentiate and simplify to get* $f(x)$ *from these assumptions. You will need to make simple choices of* $u(x), p(x), q(x)$*, but not* too *simple.*) Now compute error norms for each of the two cases, i.e. $\|E^h\|_2 = \|U^h - \hat{U}^h\|_2$, and produce a convergence plot showing the expected $O(h^2)$ convergence rate.

**Problem P13.** *Sometimes these Dirichlet boundary value problems are ill-posed!*

Consider the following linear BVP with Dirichlet boundary conditions:

$$u''(x) + u(x) = 0 \quad \text{for } a < x < b, \qquad u(a) = \alpha, \qquad u(b) = \beta. \tag{2}$$

[1] R. J. LeVeque, *Finite Difference Methods for Ordinary and Partial Diff. Eqns.*, SIAM Press 2007

**(a)** Determine the exact solution to BVP (2) when $a = 0, b = 1, \alpha = 2, \beta = 3$. (*Observe that there is exactly one solution for these parameters!*) Test your code from **P12**, using this exact solution, and demonstrate convergence at the expected $O(h^2)$ rate.

**(b)** Instead let $a = 0$ and $b = \pi$ in BVP (2). For what values of $\alpha$ and $\beta$ does (2) have any solutions? First identify a case where there is no solution. Second, identify a case where there are infinitely-many solutions, and sketch that family of solutions.

**Problem P14.** Section 2.13 addresses the problem

$$u''(x) = f(x), \qquad u'(0) = \sigma_0, \qquad u'(1) = \sigma_1, \tag{3}$$

which has Neumann conditions at both ends. In the steady heat equation interpretation of BVP (3), the values $\sigma_0$ and $\sigma_1$ are heat fluxes. (*The Dirichlet condition values $\alpha, \beta$ in BVPs* (1) *and* (2) *are temperatures.*)

**(a)** Apply the "second approach" from Section 2.12 to BVP (3). Show that you get linear system (2.58) as shown in the text.

**(b)** Generate $A$, the matrix from (2.58), in Matlab/etc; choose $m = 10$ here for example. (*Modify existing code to do this assembly? In that case show me your code.*) Enter the constant vector $e = [1, 1, \dots, 1]^\top$, and confirm numerically that $e$ is in the null space of $A$.

**(c)** The text in Section 2.13 explains a condition that $f(x)$ and $\sigma_0$ and $\sigma_1$ must satisfy so that there *is* a solution to BVP (3). If that condition is satisfied then there are actually infinitely many solutions. Choose such a case and see what happens numerically, that is, when you solve linear system (2.58) in Matlab/etc. Explain.