18.336 Pset 4 Solutions

Due Thursday, 20 April 2006.

Problem 1: Galerkin warmup

For a basis $b_n(x)$, the matrix A is defined as $A_{mn} = (b_m, Pb_n)$. Therefore, for any $\mathbf{c}$, the product $\mathbf{c} \cdot A\mathbf{c} = \sum_{m,n}(c_m b_m, Pc_n b_n) = (u, Pu)$ where $u(x) = \sum_n c_n b_n(x)$. If $\mathbf{c} \neq 0$ then $u \neq 0$ because we always require our basis functions to be linearly independent, and thus $(u, Pu) > 0$ by assumption, and therefore $\mathbf{c} \cdot A\mathbf{c} > 0$ and A is positive-definite. Q.E.D.

Problem 2: Galerkin FEM

You will implement a Galerkin finite-element method, with piecewise linear elements, for the Schrodinger eigen-equation in 1d:

$$[-\frac{d^2}{dx^2} + V(x)]\psi(x) = E\psi(x)$$

with a given potential $V(x)$ and for an unknown eigenvalue E and eigenfunction $\psi(x)$. As usual, we'll use periodic boundary conditions $\psi(x+2) = \psi(x)$ and only solve for $\psi(x)$ in $x \in [-1, 1]$.

As in class, we will approximate $\psi(x)$ by its value c_n at N points x_n ($n = 1, 2, \ldots, N$) and linearly interpolate in between.

Our basis functions are, as in class, the "tent" functions $b_n(x)$, which $= 1$ at x_n and $= 0$ at other $x_{m \neq n}$, linearly interpolated in between x_n and $x_{n\pm1}$. The periodic boundary conditions simply mean that $c_{N+1} = c_1$ for $x_{N+1} = x_1 + 2$. We define $\Delta x_n = x_{n+1} - x_n$ as in class.

(a) By definition, we have $B_{mn} = (b_m, b_n)$ and $A_{mn} = (b_m, [-\frac{d^2}{dx^2} + V(x)]b_n)$. However, we'd like to make this a bit more explicit and do some of the integrals analytically. To start with, let's evaluate B:

$$B_{nn} = \int b_n^2 dx = (\Delta x_{n-1} + \Delta x_n)/3,$$

since in each Δx interval the integral looks like $\int_0^{\Delta x} (x/\Delta x)^2 dx$. The off-diagonal elements are only non-zero for $B_{n,n\pm1}$, and come from an integral of the form $\int_0^{\Delta x} x(\Delta x - x)/\Delta x^2 dx$, giving

$$B_{n,n+1} = \Delta x_n/6,$$

$$B_{n,n-1} = \Delta x_{n-1}/6.$$

A_{mn} can be split into two pieces: $A_{mn} = K_{mn} + V_{mn}$ where K is the "kinetic" energy from the $-\frac{d^2}{dx^2}$ and V is the potential energy from the $V(x)$. K_{mn} we evaluated in class for Poisson's equation (with opposite sign), and is:

$$K_{mn} = \left\{ \begin{array}{c} 0 \quad m \neq n, n \pm 1 \\ \frac{1}{\Delta x_{m-1}} + \frac{1}{\Delta x_m} \quad m = n \\ -\frac{1}{\Delta x_m} \quad m = n - 1 \\ -\frac{1}{\Delta x_{m-1}} \quad m = n + 1 \end{array} \right. .$$

Finally, the V_{mn} terms will involve integrals over $V(x)$, which will have to be performed numerically in general:

$$V_{mn} = \int b_m(x) V(x) b_n(x) dx$$

which we can make more explicit by breaking up the integral into Δx chunks. Let

$$P_n = \int_0^{\Delta x_n} V(x + x_n) \left(\frac{x}{\Delta x_n}\right)^2 dx,$$

$$M_n = \int_0^{\Delta x_n} V(x + x_n) \left(\frac{\Delta x_n - x}{\Delta x_n}\right)^2 dx,$$

$$C_n = \int_0^{\Delta x_n} V(x + x_n) \frac{x(\Delta x_n - x)}{\Delta x_n^2} dx.$$

Then:

$$V_{mn} = \left\{ \begin{array}{c} 0 \quad m \neq n, n \pm 1 \\ P_{m-1} + M_m \quad m = n \\ C_m \quad m = n - 1 \\ C_{m-1} \quad m = n + 1 \end{array} \right. .$$

Note that these integrals are properly set up for Gaussian quadrature if $V(x)$ is smooth, since we have broken them up into pieces with smooth integrands.

One more note: because of the periodic boundary conditions, we need to set $A_{1,N} = A_{1,0}$and $A_{N,1} = A_{N,N+1}$.

(b) See the Matlab files posted on the web site. Note that in addition to `schrodinger_galerkin`, I also implemented a `schrodinger_galerkin_sparse` function that does the same thing but using Matlab's `spdiags` function to exploit the

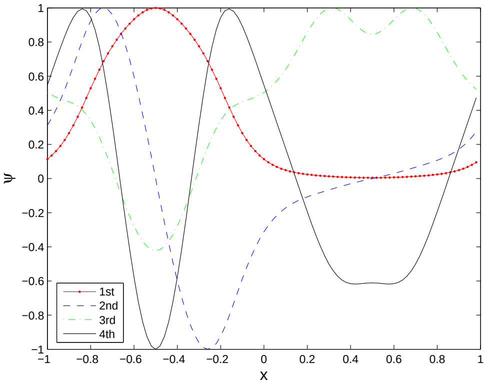

Figure 1: Wavefunctions $\psi(x)$ for four lowest eigenvalues, computed with a uniform grid for $N = 100$. (Note that the normalization of ψ is arbitrary.)

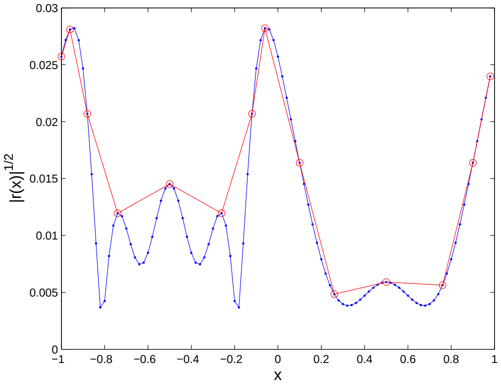

Figure 2: Residual function $\sqrt{|r(x)|}$ (blue dots), and a simplified interpolated envelope (red circles/lines) that we'll use for $\rho(x) \sim 1/\sqrt{|r(x)|}$.

sparsity of the matrix and the `eigs` function to use an iterative method to just find a few eigenvalues (it uses the Arnoldi algorithm I believe). This way I can calculate the solution for much larger N values if need be.

(c) Here we computed the solutions for $N = 100$:

(i) For uniform spacing, my lowest four E were -35.1923, -3.7363, 24.1334, and 42.6923. The corresponding eigenfunctions are shown in figure. 1.

(ii) In order to choose $\rho(x)$ in a reasonable way (*not* necessarily the best way), I tried to mimic what a "real" adaptive-grid solver would likely do: I looked at the *residual* of the solution and changed the grid density to try to equalize the residual everywhere. In particular, I calculated the residual for the $N = 100$ uniform grid as $\mathbf{r} = A\mathbf{c}_0 - EB\mathbf{c}_0$ where $\mathbf{c}_0$ is the "exact" solution on the same grid (interpolated from $N = 8000$)[1]—in particular, we'll use the residual for the lowest band since that's what we want to compute below. Now, if we believe that this scheme is quadratically accurate, then in order to cut the residual by 4 we should double the resolution. Thus, if we suppose that we want to equalize the residual everywhere, this would argue for using $\rho(x) \sim 1/\sqrt{|r(x)|}$ where $r(x)$ is the residual function, suitably interpolated from $\mathbf{r}$. The residual $\mathbf{r}$ is plotted in figure 2; since it is kind of "bumpy" we'll just use an $r(x)$ formed by interpolating an upper-bound envelope of $\mathbf{r}$, as shown by the straight red lines in the figure. Using this $\rho(x)$ and $N = 100$ (via the makegrid function supplied on the web site), I get eigenvalues E of -35.1944, -3.7364, 24.1668, and 42.7061. The $\psi(x)$ plots are indistinguishable and are therefore not shown.

(d) The error ΔE_N for $N = 32, 64, 128, 256, 512$ is plotted in figure 3. Both are clearly quadratic, and the $\rho(x)$ helps slightly (the error is reduced by about 1.4). Darn. Fitting the last two points, it seems that $\Delta E_N \approx 42N^{-1.9993}$ for $\rho(x)$. For reference, the "exact" value of E (as computed by a uniform grid for $N = 8000$) should be -35.1999808. A table of our data follows:

[1] A more realistic adaptive-grid code would normally estimate the residual by comparing with the solution at a *coarser* resolution.

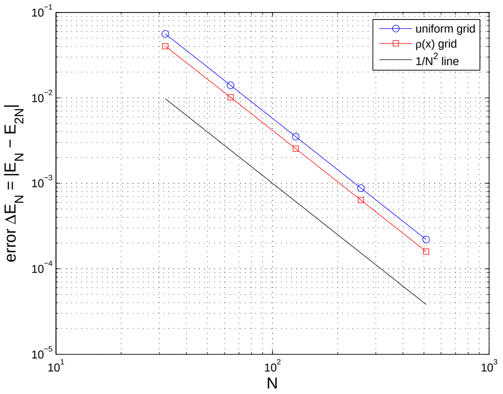

Figure 3: Error ΔE_N vs. N for a uniform grid and the $\rho(x)$ grid we selected, along with a quadratic $1/N^2$ line for comparison. Both are quadratic, and our $\rho(x)$ helps slightly but not much.

N	E_N	ΔE_N
32	-35.146211	0.04023
64	-35.186436	0.010157
128	-35.196592	0.0254
256	-35.199134	0.000636
512	-35.199770	0.000159

Problem 3: Orthogonal polynomials

Let x_n ($n = 1, \ldots, N$) be the roots of the orthogonal polynomial $p_N(x)$, which we showed must lie in (a, b). Suppose that the k-th root is repeated, with multiplicity $M > 1$; from this, we will prove a contradiction similar to the proof in class. In particular, form the new polynomial $S(x) = p_N(x)/(x - x_k)^L$, where $L = M$ if M is even and $L = M - 1$ if M is odd. Clearly, $S(x)$ has smaller degree than $p_N(x)$ since $L > 0$, and therefore we have zero inner product $(S, p_N) = 0$

However, it must also be the case that $S(x)$ has the same sign as $p_N(x)$ everywhere. If M is even, then $p_N(x)$ does not change sign at x_k (which is an extremum), and neither does $S(x)$ since $L = M$ (we removed the x_k root entirely). If M is odd, then $p_N(x)$ changes sign at x_k and so does $S(x)$ (since $L = M - 1$ and thus $S(x)$ contains x_k with multiplicity 1). All the other sign changes are the same since the other roots were unchanged. Therefore, we must have a positive inner product $(S, p_N) > 0$, which contradicts the orthogonality above. Q.E.D.