

MIT 18.335, Fall 2006: Homework 5

Due November 28

A Linear Elasticity Model Problem

In problems 1-7 below you will study inverse iteration and classical iterative methods for a linear elasticity problem. Some utility functions in MATLAB are available on the course web pages, for creating the matrices and visualizing the solutions. You can essentially use these functions as black-boxes, only understanding what they do. Here is what you need to know:

- `K=assemble(n)` Creates a matrix K (the “stiffness” matrix for the problem). We will study the eigenvalue problem $Ku = \lambda u$ for the problem size $n=20$.
- `[K,f]=mkmodel(n)` Creates a matrix K and a vector f , implementing a typical boundary-value model problem. We will study the linear system of equations $Ku = f$.
- `qdplot(u)` Plots the displacement field u
- `qdanim(u)` Animates the displacement field u (only makes sense for eigenfunctions)

Please hand in short and concise MATLAB codes. You do not have to hand in code for plotting or setting up calculations, or any displacement plots with `qdplot`.

Inverse Iteration and Rayleigh Quotient Iteration for Eigenvalues

1. Write MATLAB code that performs inverse iteration on a matrix K with shift μ . Initialize $v^{(0)}$ with `randn`, and compute and store all the Rayleigh quotients $\lambda^{(k)}$. Iterate until the difference between $\lambda^{(k+1)}$ and $\lambda^{(k)}$ is less than 10^{-14} .

You can create the shifted matrix by `K-mu*speye(N,N)`, where `N=size(K,1)`, and you can solve the linear systems using the MATLAB `\` command. You might also want to plot the eigenfunctions in each iteration using `qdplot(v); drawnow;`.

2. Run the example in the help-text for `assemble`. The true eigenvalues are on the diagonal of D , or `d=diag(D)`. Based on the true eigenvalues choose four shifts μ according to below. For each shift, run the code you wrote in 1 and plot the error $e^{(k)} = \lambda^{(k)} - \lambda$ versus the iteration number k using `semilogy` (you can plot all four curves in the same graph). For the true eigenvalues λ you can use the value in `d` or the last iterate $\lambda^{(k)}$. Also make a simple estimate of the convergence rates (the factor C in $e^{(k+1)} = Ce^{(k)}$), and compare with the expected convergence rate (which you need the true eigenvalues in `d` for). Choose shifts that are:
 - a) Slightly below a distinct eigenvalue (but not too close!)
 - b) As in a) but above the eigenvalue. The Rayleigh quotient should behave as in a), but why does the eigenfunction flip back and forth?
 - c) Almost right between two distinct eigenvalues (about 10% closer to one of them)
 - d) Close to an eigenvalue with multiplicity > 1 . Does it seem to have trouble finding the eigenvalue? You do not have to explain why.
3. Change your code so it performs Rayleigh quotient iteration instead of inverse iteration (a very small change!). Run the four shifts again and plot the errors versus the iteration number. Can you observe the cubic convergence? No calculations required, just try to see if the number of digits are tripled at some iteration.

Classical Iterative Methods for Linear Systems

4. Write MATLAB code that implements the Jacobi method, the Gauss-Seidel method, and the SOR method for a matrix K . Initialize $u^{(0)}$ with `zeros`, and compute and store all residual norms $\|r^{(k)}\| = \|f - Ku\|$. Iterate until the relative residual $\|r^{(k)}\|/\|r^{(0)}\|$ is less than 0.01 (or greater than 100, in case it does not converge), but not for more than 400 iterations.

You can create the splitting into diagonal and triangular parts by

```
D = triu(tril(K));
U = -triu(K,1);
L = -tril(K,-1);
```

Again solve the linear systems using the MATLAB `\` command, and you might want to plot the approximate solutions every now and then using `qdplot(u); drawnow;`.

5. Create a typical problem by `[K,f]=mkmodel(20);`. Since K is symmetric positive definite we know that Gauss-Seidel and SOR will converge, but it is not clear about Jacobi. Compute the spectral radius of the Jacobi iteration matrix $D^{-1}(L+U)$ by iterating with power iteration:

```
u=randn(size(f));
for i=1:1000, u=D\((L+U)*u); u=u/norm(u); end
rho=abs(u'*(D\((L+U)*u)));
```

Based on this, will the Jacobi method converge? Compute the spectral radius for Gauss-Seidel and SOR with $\omega = 2 - 2/n = 1.9$ in a similar way.

6. Solve $Ku = f$ using each of the three methods you wrote in 4 and plot the residuals $\|r^{(k)}\|$ versus the iteration number k using `semilogy` (you can plot all three curves in the same graph). Make a simple estimate of the convergence rates (the factor C in $r^{(k+1)} = Cr^{(k)}$), and compare with the expected convergence rates (from the spectral radii in 5).
7. For the model Poisson problem, we saw that the convergence rate scaled like $1 - \text{const}/n^2$ for Jacobi and Gauss-Seidel, but like $1 - \text{const}/n$ for SOR with optimal ω . To see if we have a similar dependence in our elasticity problem, estimate the convergence rates in Gauss-Seidel and SOR for $n = 10$ (that is, create a new system `[K,f]=mkmodel(10);` and run your solvers again). Use the heuristic parameter $\omega = 2 - 2/n = 1.8$ for SOR. Approximately which exponent α do you obtain in $1 - \text{const}/n^\alpha$ based on your two convergence rates for $n = 10$ and $n = 20$?

Problems from the textbook

8. Trefethen/Bau 35.3 (b)-(c) **Note:** You do not have to read about GMRES for this problem, just use the lecture slides on classical iterative methods.
9. Trefethen/Bau 38.2
10. Trefethen/Bau 38.3
11. Trefethen/Bau 38.4