M3/4/5N9 – Project 3
Due 12 Jan 2018

Reminder: High level MATLAB functions, such as 'backslash' are not permitted to be used unless explicitly instructed to do so.

I. Schrodinger equation as an eigenvalue problem

In quantum mechanics, the quantization of energy is linked to the spectrum of the linear operator appearing in the time-independent Schrodinger equation. In one-dimension, the time-independent (non-dimensional) Schrodinger equation is

$$\left[-\frac{d^2}{dx^2} + V(x)\right]\Psi = E\Psi, \tag{1}$$

where the wavefunction, Ψ, is the eigenvector and the energy, E, is the eigenvalue. The function $V(x)$ is the external potential. To find numerically the eigenvalue-eigenvector pairs on a periodic domain, we introduce a grid of equispaced points $x_j = (j-1)\Delta x$ for $j = 1, \ldots, N$ with $\Delta x = 2\pi/N$ at which we seek approximate values of the wave function Ψ. The differential operator is replaced by the $N \times N$ matrix

$$D = \frac{1}{\Delta x^2}\begin{bmatrix} -2 & 1 & & & 1 \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ 1 & & & 1 & -2 \end{bmatrix}. \tag{2}$$

With this discretization, the Schrodinger equation becomes

$$\left[-D + V_N\right]\Psi_N = E\Psi_N \tag{3}$$

where Ψ_N is the $N \times 1$ vector of the values of the wave function at the grid points, V_N is the $N \times N$ diagonal matrix with non-zero entries $V_{N,ii} = V(x_i)$. Note that $-D + V_N$ is a symmetric matrix. Using the permutation matrix

$$P = [\boldsymbol{e}_1\ \boldsymbol{e}_N\ \boldsymbol{e}_2\ \boldsymbol{e}_{N-1}\ \boldsymbol{e}_3 \ldots], \tag{4}$$

we consider the matrix $L = P^T(-D + V_N)P$ which transforms $-D + V_N$ into a pentadiagonal matrix.

In this problem, we will find the wave functions and energy levels associated with the Gaussian potential well $V(x) = -V_0\exp(-\gamma(x-\pi)^2)$. The matrix L is provided by the code `Schrodinger.m` with $N = 1024$, $V_0 = 200$, and $\gamma = 10$. For a matrix with structure such as that of L, the preferred approach to performing reduction to a tridiagonal matrix and QR decomposition is to use Givens rotations to take advantage of the many zeros already present in the matrix. More on Givens rotations and how they can be used for QR decomposition can be found in the attached section from Ch. 5 of Golub and Van Loan.

1. Explain why it is valid to use the permuted problem to find the eigenvalues.
2. Transform the pentadiagonal matrix, L, into a tridiagonal matrix, T. You will receive some marks for using the general Householder-based methods described in lecture. For full marks, use Givens rotations to obtain T more efficiently. In your report, describe the algorithm you have implemented and examine how the operation count scales with N. Confirm your analysis by timing your code.

3. Implement the QR algorithm with shifts to find all the eigenvalues of T and consequently $-D+V_N$. Again, you will only receive some marks for implementing the general algorithm that does not take advantage of the fact that T is tridiagonal. For full marks, use Givens rotations to perform the QR decomposition of the tridiagonal matrix more efficiently. In your report, describe the algorithm you have implemented and examine how the operation count scales with N. Confirm your analysis by timing your code. Plot the eigenvalues, i.e. the energy levels E_n, against their index. Provide the values of the lowest 10 energy levels in a table in your report. Also, discuss in your report whether you think the initial reduction to a tridiagonal matrix is necessary for this problem.

4. Perform inverse iteration to obtain the eigenvectors, i.e. wave functions, Ψ_n that have $E_n < 0$, as well as a case where $E_n > 0$. For full marks, your implementation of inverse iteration should take advantage of matrix structure when solving linear systems. Discuss your implementation and how the number of operations for your solver scales with N. Plot each $|\Psi_n|^2$ against x. Discuss the qualitative differences in the wave functions as E_n increases.

II. Portfolio optimization with conjugate gradients

With our discussions of least-squares and the conjugate gradient method, we saw how many linear algebra computations are directly related to optimization problems. Linear algebra computations are also an essential component to solving the linear systems that appear when applying iterative methods to solve more general optimization problems. This problem explores this application in the context of financial mathematics.

In a paper from 1952, the economist Harry Markowitz presented an optimization problem that aims to minimize investment risk while maintaining a prescribed level of return. A variant of his original optimization problem is

$$\text{minimize } \gamma x^T \Sigma x - (1-\gamma)\bar{p}^T x \tag{5}$$

$$\text{subject to } \mathbf{1}^T x = 1 \tag{6}$$

$$x_i \geq 0, i = 1, \ldots, N \tag{7}$$

where x is the portfolio vector with each entry x_i describing the percentage of the portfolio invested in investment i and $\mathbf{1}$ is the $N \times 1$ vector with each entry equal to 1. The vector $\bar{p}$ is the vector whose entries are the mean price of each investment, and the symmetric positive definite matrix Σ is the covariance matrix of the investment prices. The parameter $0 \leq \gamma \leq 1$ controls the relative importance of risk, $x^T \Sigma x$, to expected return, $\bar{p}^T x$, in the cost function.

To handle the constraint that each entry of x is positive, we consider the modified Markowitz optimization problem,

$$\text{minimize } \gamma x^T \Sigma x - (1-\gamma)\bar{p}^T x - t\sum_{i=1}^{N} \log(x_i) \tag{8}$$

$$\text{subject to } \mathbf{1}^T x = 1 \tag{9}$$

where the parameter $t > 0$ controls the accuracy of the approximate optimization problem. As $t \to 0$, we recover the original problem. The minimizer to this problem can be found iteratively using Newton's method, where the initial guess is required to satisfy $\mathbf{1}^T x_0 = 1$ and at each subsequent iteration indexed by n (and not to be confused with the entry index, i), one solves the linear system

$$\begin{bmatrix} M & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \Delta x_n \\ \lambda \end{bmatrix} = A \begin{bmatrix} \Delta x_n \\ \lambda \end{bmatrix} = \begin{bmatrix} g_n \\ 0 \end{bmatrix} \tag{10}$$

where $M = 2\Sigma + \Phi(x_n)$, $\Phi(x_n)$ is the diagonal matrix whose non-zero entries are $\Phi_{ii}(x_n) = t/(x_{n,i}^2)$, and $g_n = -2\Sigma x_n + (1-\gamma)\bar{p} + \phi_n$ with ϕ_n being the vector whose entries are given by $\phi_{n,i} = -t/(x_{n,i})$. The approximate minimizer is then updated $x_{n+1} = x_n + \alpha \Delta x_n$, where α is the step size. Performing this iteration while letting t go to zero gives the optimum, $x^\star$, to the original problem.

1. The code `PortfolioOpt.m` implements Newton's method as described above using the mean price and covariance data (`PortfolioData.mat`) for a portfolio of all 225 stocks of the Nikkei 225. The solver for the linear system is missing. Implement LDL^T factorization (see Ch. 4 in Golub and Van Loan) to solve the linear system and describe why this algorithm is an appropriate choice. Vary γ from 0.1 to 0.9 with steps of $\Delta\gamma = 0.1$. Provide a table of the risk and return. For each value of γ provide the two largest entries of $x^\star$.

2. For the equality constraint, find a matrix Z, such that $\mathbf{1}^T Z = 0$, i.e. the columns of Z span the null space of $\mathbf{1}^T$. Provide evidence in your report that $\mathbf{1}^T Z = 0$ is satisfied.

3. The constraints can be removed from the problem by considering $\Delta x_n = Z\Delta\tilde{x}_n$ and solving $\tilde{M}\Delta\tilde{x}_n = \tilde{g}_n$, where $\tilde{M} = Z^T M Z$ and $\tilde{g}_n = Z^T g_n$. Explain why $\tilde{M}$ is positive semidefinite. Modify `PortfolioOpt.m` and implement the conjugate gradient (CG) method to solve this linear system and explain why it can be used even though $\tilde{M}$ is not full rank. Comment on the performance of CG relative to the direct LDL^T factorization. In which cases does CG perform poorly? Provide evidence of your arguments by timing your implementation and providing iteration counts for different values of γ. Discuss your results and observations.

4. Determine a suitable preconditioner and implement preconditioned CG. Demonstrate that the number of iterations has decreased as a result of preconditioning. Remember, there is a cost associated with solving the linear system involving the preconditioner and this should be kept to a minimum. Discuss your preconditioner choice and improvements in performance in your report.