Tridiagonal matrices

In this lab you will study Gaussian elimination for tridiagonal matrices. Send me by email by next Monday a listing of your debugged routine `trisolve`.

A tridiagonal matrix is one for which $a_{ij} = 0$ unless $j = i-1$, i, or $i+1$. Thus a 5×5 tridiagonal matrix has the form

$$A = \begin{pmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{pmatrix}.$$

Tridiagonal matrices come up very often in applications. For all but very small tridiagonal matrices, it is very inefficient to store the entire matrix, including all the zero entries. E.g., for a 100×100 tridiagonal matrix, how many zero and nonzero entries are there: zero____________ ; nonzero____________ . Instead it is better to store an $n \times n$ tridiagonal matrix as three vectors: a vector l of length $n-1$ giving the entries below the diagonal, a vector d of length n giving the entries on the diagonal, and a vector u of length $n-1$ giving the entries above the diagonal. For the 5×5 case the storage looks like this:

$$A = \begin{pmatrix} d_1 & u_1 & 0 & 0 & 0 \\ l_1 & d_2 & u_2 & 0 & 0 \\ 0 & l_2 & d_3 & u_3 & 0 \\ 0 & 0 & l_3 & d_4 & u_4 \\ 0 & 0 & 0 & l_4 & d_5 \end{pmatrix}.$$

More important even than the storage is the number of operations needed to solve a tridiagonal linear system with Gaussian elimination. If we use Gaussian elimination in the usual way, we need $2n^3/3 + O(n^2)$ flops. However an efficient routine which takes into account all the zero elements can be *much* faster: it costs only $O(n)$ operations.

1. Write a matlab function `function x = trisolve(l,d,u,b)` which solves the matrix equation $Ax = b$. The inputs should be an $n-1$ vector `l`, an n-vector `d`, and an $n-1$ vector

`u`, which together specify the matrix A, and an n vector `b`, which gives the right hand side. The output should be the n-vector `x` giving the solution. The algorithm should be Gaussian elimination in which `l` gets overwritten with the multipliers and back substitution for the resulting triangular system.

Hints: To get n, use `n=length(d)`. To loop from 1 to $n-1$ use `for i=1:n-1`; ... ; `end`. To loop backward from $n-1$ to 1 use `for i=n-1:-1:1`.

2. To debug your program, use some simple data. Let `l=1:4`, `d=8*ones(1,5)`, `u=-1:-1:-4`, `b=[1 -1 1 -1 1]`. Solve $Ax = b$ using your function. Now create a regular matrix in Matlab with the command `A=diag(l,-1)+diag(d,0)+diag(u,1)` and solve $Ax = b$ with `x=A\b'`. The answers should be the same. If not, fix your program!

The tridiagonal system

$$\begin{pmatrix} -2 & 1 & 0 & \dots & 0 & 0 \\ 1 & -2 & 1 & \dots & 0 & 0 \\ 0 & 1 & -2 & & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & & -2 & 1 \\ 0 & 0 & 0 & & 1 & -2 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{n-1} \\ z_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{n-1} \\ b_n \end{pmatrix},$$

describes the following application. A taut elastic band of length 1 is pinned down at the ends $x = 0$ and $x = 1$. At the n points $x_1 = 1/(n+1)$, $x_2 = 2/(n+1)$, ... $x_n = n/(n+1)$ a weight of b_i is suspended. This causes the elastic band to deflect, with the deflection of the ith point proportional to z_i.

3. Take $n = 10$ and use **`trisolve`** to find the deflections z_i for a constant weight b_i (`b=ones(1,n)`). Plot the deflection. (A good plotting command is `plot(linspace(0,1,n+2),[0;z(:);0])`. This shows the two end points where the deflection is 0.) Do the same thing for $n = 50$ and $n = 100$. What shape does the band appear to take? ____________ (When n is large, this is a model for a cable deflecting under its own weight.)

4. Use `flops(0)` before `trisolve` and then `flops` afterwards to find out how many flops `trisolve` needs when $n = 100$. ____________ The answer should be under 1,500. If not, your routine is not efficient and you should fix it.

5. How many flops if you store the matrix as a regular square matrix and use Matlab's \ operation to solve the system? ____________

6. How many flops using `trisolve` if $n = 1000$. Please do *not* attempt $n = 1000$ using Matlab's \ operation!

7. Now let's study a more interesting set of weights. Put unit weights on the left half of the band, but none on the right half (`b=[ones(1,n/2),zeros(1,n/2)]`) and use `trisolve` to determine the resulting deflection. Try various values of n. For what x value is the deflection greatest? ____________ If you attach a length of light but strong fishing line to an equal length of heavy cable, and hang the whole thing from two poles, where will the lowest point be? ____________

Further optional problems:

8. Investigate some other weightings.

9. The matrix from the deflection problem is symmetric. If we simply multiply by -1 it is positive definite as well. Write a `trichol` routine to do Cholesky decomposition for tridiagonal matrices.

10. Write an efficient routine to solve pentadiagonal matrices, i.e., matrices with $a_{ij} = 0$ unless $j = i-2, i-1, i, i+1$, or $i+2$.