MATH 4093/5093 Homework 3 Due Fri, 2/18/2011

Problem 1. Consider the function $f(x) = \frac{1}{\sqrt{4-x}}$ (well defined for any $x < 4$). Let P_2 be its second-degree Taylor polynomial (where f is expanded around $x_0 = 0$), and R_2 be the corresponding remainder, so that $f(x) = P_2(x) + R_2(x)$ (see page 25 of the book).

(a) Write explicitly the polynomial $P_2(x)$.

(b) Write explicitly $R_2(x)$. Remember that, in the notations of the book, $\xi(x)$ is an unknown number between x_0 and x.

(c) If we want to compute the value $P_2(x)$ instead of $f(x)$ for some particular value of x, we need to know how large the (absolute) error, $|f(x) - P_2(x)| = |R_2(x)|$, can be. Use the expression for $R_2(x)$ to compute the rigorous upper bound on the error in replacing $f(0.9) = \frac{1}{\sqrt{3.1}}$ by $P_2(0.9)$. In such a computation, you will have to take the worst possible case, i.e., the upper bound on the error is

$$\max_{\xi} |R_n| = \max_{\xi} \frac{\left|f^{(n+1)}(\xi)\right|}{(n+1)!} |x - x_0|^{n+1} ,$$

where ξ is allowed to be anywhere between x_0 and x. The value of x in this computation is fixed (it is equal to 0.9), and the maximum is taken only over ξ.

(d) Compute the exact value $f(0.9)$, the approximate value $P_2(0.9)$, and the numerical value of the true error, $|f(0.9) - P_2(0.9)|$. Compare it with the theoretical upper bound on the error found in part (c). Discuss your findings.

(e) Compute the value of the relative error in replacing $f(0.9)$ by $P_2(0.9)$.

(f) Now assume that we want to replace the exact value $f(x)$ with the approximate value $P_2(x)$, where x is allowed to be in the interval $[0, 0.9]$. Find the rigorous upper bound on $|f(x) - P_2(x)|$ if x is allowed to vary anywhere in the interval $[0, 0.9]$.

Remark: Here you have to find $\max\limits_{x \in [0,0.9]} |R_2(x)|$ – to find the exact upper bound, in principle you would have to differentiate $\xi(x)$ with respect to x. However, since you do not know $\xi(x)$, treat x and ξ as independent variables and take the worst possible case: $\max\limits_{x \in [0,0.9]} \max\limits_{\xi \in [0,0.9]} |R_2(x)|$. Clearly, your rigorous upper bound may be loose, but that's the best you can do.

Problem 2. Gauss-Jordan elimination is an alternative to the Gaussian elimination followed by back substitution. The Gauss-Jordan elimination replaces the elements both above and below the pivot with zeros by using ERO_3 until the coefficient part of the augmented matrix becomes diagonal, and then makes this diagonal matrix equal to the unit matrix by using

ERO_2. After this procedure the solution $\mathbf{x}$ of the system $A\mathbf{x} = \mathbf{b}$ is equal to the vector that is in the right-hand side of the augmented matrix. This is illustrated in the example below, whose exact solution is $\mathbf{x} = (1\ \ 2\ \ 3)^T$; the pivot in each pass is written in angular brackets $\langle\ \rangle$. We called the last step in Gauss-Jordan elimination "normalization".

$$\left(\begin{array}{ccc|c} \langle 2\rangle & 3 & -2 & 2 \\ 3 & 5 & 0 & 13 \\ 1 & 2 & 1 & 8 \end{array}\right) \xrightarrow{\text{pass 1}} \left(\begin{array}{ccc|c} 2 & 3 & -2 & 2 \\ 0 & \langle\frac{1}{2}\rangle & 3 & 10 \\ 0 & \frac{1}{2} & 2 & 7 \end{array}\right) \xrightarrow{\text{pass 2}} \left(\begin{array}{ccc|c} 2 & 0 & -20 & -58 \\ 0 & \frac{1}{2} & 3 & 10 \\ 0 & 0 & \langle -1\rangle & -3 \end{array}\right)$$

$$\xrightarrow{\text{pass 3}} \left(\begin{array}{ccc|c} 2 & 0 & 0 & 2 \\ 0 & \frac{1}{2} & 0 & 1 \\ 0 & 0 & -1 & -3 \end{array}\right) \xrightarrow{\text{normalization}} \left(\begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{array}\right).$$

Gauss-Jordan elimination can be used for inverting an $n \times n$-matrix A as follows. Write the $n \times 2n$ "augmented matrix" $(A|I)$, where I is the $n \times n$ unit matrix. Then perform elementary row operations ERO_2 and ERO_3 (but *not* ERO_1!) on the rows of this augmented matrix by following the Gauss-Jordan strategy until the matrix becomes of the form $(I|B)$. Then the matrix B is the inverse of A. Here is an example of such an inversion:

$$\left(\begin{array}{ccc|ccc} \langle 2\rangle & 3 & -2 & 1 & 0 & 0 \\ 3 & 5 & 0 & 0 & 1 & 0 \\ 1 & 2 & 1 & 0 & 0 & 1 \end{array}\right) \xrightarrow{\text{pass 1}} \left(\begin{array}{ccc|ccc} 2 & 3 & -2 & 1 & 0 & 0 \\ 0 & \langle\frac{1}{2}\rangle & 3 & -\frac{3}{2} & 1 & 0 \\ 0 & \frac{1}{2} & 2 & -\frac{1}{2} & 0 & 1 \end{array}\right) \xrightarrow{\text{pass 2}} \left(\begin{array}{ccc|ccc} 2 & 0 & -20 & 10 & -6 & 0 \\ 0 & \frac{1}{2} & 3 & -\frac{3}{2} & 1 & 0 \\ 0 & 0 & \langle -1\rangle & 1 & -1 & 1 \end{array}\right)$$

$$\xrightarrow{\text{pass 3}} \left(\begin{array}{ccc|ccc} 2 & 0 & 0 & -10 & 14 & -20 \\ 0 & \frac{1}{2} & 0 & \frac{3}{2} & -2 & 3 \\ 0 & 0 & -1 & 1 & -1 & 1 \end{array}\right) \xrightarrow{\text{normaliz.}} \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & -5 & 7 & -10 \\ 0 & 1 & 0 & 3 & -4 & 6 \\ 0 & 0 & 1 & -1 & 1 & -1 \end{array}\right).$$

In this problem you will count the number of operations needed to invert a matrix by Gauss-Jordan elimination as explained above. Always assume that nothing goes wrong (i.e., assume that A is invertible and that in all divisions the denominator is not zero). Also, keep in mind that the rows of the augmented matrix are never interchanged (i.e., ERO_1 is forbidden).

(a) Show that if one naively applies Gauss-Jordan elimination without taking into account the structure of the identity matrix in the initial form $(A|I)$ of the augmented matrix, then the computation of A^{-1} requires $3n^3 - 2n^2$ arithmetic operations. By an operation we mean addition, subtraction, multiplication, or division. The computation of the element $m = -a_{\text{row,pass}}/a_{\text{pass,pass}}$ as in the pseudocode on page 155 of the book is counted as one operation. Setting an element to be equal to zero (as in the line $a_{\text{row,pass}} = 0$ of the pseudocode on page 155) is not counted as an operation.

Please write your calculations legibly! An unclear calculation will be considered wrong!

Hint: On page 156 of the book the number of operations in the Gauss-Jordan elimination to solve the linear system $A\mathbf{x} = \mathbf{b}$ is counted, and the result is

$$\begin{aligned}
&\sum_{p=1}^{n-1} \sum_{r=p+1}^{n} \left(1 + \sum_{c=p+1}^{n+1} 2\right) + \sum_{p=2}^{n} \sum_{r=1}^{p-1} \left(1 + \sum_{c=p+1}^{n+1} 2\right) + n \\
&\quad = \sum_{p=1}^{n-1} \sum_{r=p+1}^{n} (3 + 2n - 2p) + \sum_{p=2}^{n} \sum_{r=1}^{p-1} (3 + 2n - 2p) + n \\
&\quad = \sum_{p=1}^{n-1} (n-p)\,(3 + 2n - 2p) + \sum_{p=2}^{n} (p-1)\,(3 + 2n - 2p) + n \\
&\quad = \sum_{p=1}^{n-1} \left[2p^2 - (4n+3)p + n(2n+3)\right] + \sum_{p=2}^{n} \left[-2p^2 + (2n+5)p - (2n+3)\right] + n \\
&\quad = n^3 + n^2 - n \ ,
\end{aligned}$$

where in the last part the following facts were used: $\sum_{p=1}^{k} p = \frac{1}{2}k(k+1)$, $\sum_{p=1}^{k} p^2 = \frac{1}{6}k(k+1)(2k+1)$. This calculation should be very helpful for you – the modifications needed to solve part (a) are minor. But you have to be *very careful* when writing the very first expression (the one with the triple sums).

(b) **Only for the students taking the class as MATH 5093!**

Show that if one takes into account the structure of the identity matrix in $(A|I)$ (e.g., does not multiply by 1 and does add/subtract 0), then the computation of A^{-1} by using Gauss-Jordan elimination can be reduced to $2n^3 - 2n^2 + n$ operations.

Problem 3. The MATLAB code `nikola_petrov_gauss_elim.m` (available at the class web-site) solves the linear system $A\mathbf{x} = \mathbf{b}$ by Gaussian elimination and back substitution, without any pivoting. To see how it works, run it as follows:

```
A = [ 3 2 -4 ; -4 5 -1 ; 2 -3 5 ]
b = [ -5 ; 3 ; 11 ]
x = nikola_petrov_gauss_elim(A,b)
```

This will compute the solution $\mathbf{x} = (1\ \ 2\ \ 3)^T$ of the linear system, and the accuracy will be excellent. Try also

```
A = [ 3 2 -4 ; 2 -3 5 ; -4 5 -1 ]
b = [ -5 ; 11 ; 3 ]
x = nikola_petrov_gauss_elim(A,b)
```

which is the first system, but with the equations are written in a different order (in this case, it is clear from the output that the program does not do interchange the rows of the augmented matrix since pivoting is not used).

In the case

```
A = [ 1 2 3 ; 4 5 6 ; 7 8 9 ]
b = [ -5 ; 3 ; 11 ]
x = nikola_petrov_gauss_elim(A,b)
```

think about what went wrong (hint: what is $\det A$?). This is just for you to see how the code works, there is no need to attach anything in your homework about it.

Below is a brief explanation how the code `nikola_petrov_gauss_elim.m` works.

Line 1 is the standard first line for a MATLAB function, and lines 3–17 are the comments about the function (what it does, how to call the function, what are the inputs and outputs, etc.). To display the results more clearly, on line 19 we request that MATLAB prints the results in "`long e`" format.

On line 20, the program finds the number of rows and columns of the coefficient matrix A, and on lines 21–24 checks if A is a square matrix (if not, the program outputs an error message and exits). Let A be an $n \times n$ matrix; in the notations of the MATLAB code, this means that $\texttt{nrow} = \texttt{ncol} = n$. Similarly, on lines 25–29 we check if $\mathbf{b}$ is a column vector of the appropriate size (i.e., a matrix of size $n \times 1$); to do this, we compute the variable `nb` to be the row vector $(n, 1)$ (on line 25), and then check if n is equal to the size of A and if $\mathbf{b}$ is a row vector (in MATLAB the vertical bar, `|`, is the logical OR operation).

Gaussian elimination starts on line 35. On line 36 we check if the element a_{ii} is equal to 0 – if it is, we look for a nonzero element below it, i.e., among $a_{i+1,i}$, $a_{i+2,i}$, …, $a_{n,i}$; the array of these elements of A is written in MATLAB as `A(i+1:nrow,i)` (the colon operator gives the range in which the indices can vary – in this case the first index varies from `i+1` to `nrow`, while the second index is always `i`). This search for a nonzero element below a_{ii} is done with the MATLAB function `find`, which locates all nonzero elements of an array and returns the indices of those elements in a vector; if the array contains no nonzero elements, then `find` returns an empty array (for more information on the MATLAB commands `find` and `isempty`, look them up in the index of Overman's *MATLAB Overview*). If a_{ii} and all elements below it are zero, this means that the matrix A is singular (why?), and the program outputs an error message and stops. If the first nonzero element below a_{ii} is $a_{t,i}$, we swap rows i and t of the augmented matrix (the swapping is done on lines 42–44). After we have performed the swapping (if needed), we make all elements below a_{ii} equal to zero in the standard way (lines 46–51). On line 52, the matrix A is printed on the screen, just to monitor how the code works (*you have to leave this line in your codes, so that the grader can see how your codes work!*).

On line 63 we use the command `zeros` to define the vector `x` to be the column vector of length `nrow` with zero entries; at the end this vector will contain the solution of the linear system. On lines 64–67 we perform the back substitution in the most straightforward way. The command `.*` is not the standard matrix multiplication, but an elementwise multiplication. If `C` and `D` are matrices of the same dimension, then `E = C .* D` has elements `E(i,j)=C(i,j)D(i,j)` (no summation!); similarly, `F = C ./ D` has elements `F(i,j)=C(i,j)/D(i,j)` (again, no summation). The prime means the transposed matrix (on line 66, `x(i+1:nrow)` is a column vector of length `nrow-i`, while `x(i+1:nrow)'` is a row vector of the same length); `A(i,i+1:nrow)` is a row vector of length `nrow-i`, so that the elementwise multiplication `A(i, i+1:nrow) .* x(i+1:nrow)'` is well-defined.

(a) Run the following commands:

```
A = [ 1 2 3 ; 2 4-1e-12 7 ; 1/5 -3/7 2/5 ]
xexact = [ 1 ; 2 ; 3 ]
b = A * xexact
x = nikola_petrov_gauss_elim(A,b)
```

```
norm( x - xexact, 1 )
norm( x - xexact, inf )
norm( x - xexact )
```

Here you are solving the linear system $A\mathbf{x} = \mathbf{b}$, where $A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 4-10^{-12} & 7 \\ \frac{1}{5} & -\frac{3}{7} & \frac{2}{5} \end{pmatrix}$, $\mathbf{b} = A\mathbf{x}_{\text{exact}}$, and $\mathbf{x}_{\text{exact}} = (1\ \ 2\ \ 3)^T$. After that you are comparing the exact solution $\mathbf{x}_{\text{exact}}$ with the approximate solution $\mathbf{x}$ obtained without using pivoting. The MATLAB command `norm` computes the norm of a vector: if $\mathbf{y} = (y_1\ \ y_2\ \ \ldots\ \ y_n)^T$, then

$$\begin{array}{rcl} \texttt{norm(y,1)} & \text{returns} & \|\mathbf{y}\|_1 := \sum_{i=1}^{n} |y_i| \ ; \\ \texttt{norm(y,inf)} & \text{returns} & \|\mathbf{y}\|_\infty := \max_{1\le i\le n} |y_i| \ ; \\ \texttt{norm(y,2)} & \text{returns} & \|\mathbf{y}\|_2 := \sqrt{\sum_{i=1}^{n} |y_i|^2} \ ; \\ \texttt{norm(y)} & \text{returns} & \text{the same as } \texttt{norm(y,2)} \ . \end{array}$$

Write down the approximate solution $\mathbf{x}$ (with all digits MATLAB gives you), as well as the 1-, infinity-, and 2-norms of the difference $\mathbf{x} - \mathbf{x}_{\text{exact}}$ (when you write the norms of the difference, you may write only 3-4 digits, but do not forget the power of 10).

(b) Write a MATLAB code called `yourfirstname_yourfamilyname_gauss_elim_1.m` that performs Gaussian elimination with partial pivoting, followed by back substitution. The partial pivoting in your code must be done without using a row vector, i.e., by directly swapping the rows of the augmented matrix, if needed (as in Example 3.4 on page 162 of the book). You do this by appropriately modifying the part of the code `nikola_petrov_gauss_elim.m` that performs Gaussian elimination. It will be enough to replace lines 36–45 of `nikola_petrov_gauss_elim.m` with appropriate new lines. Leave the printing of the matrix A on line 52 of `nikola_petrov_gauss_elim.m` for the grader to see how your code works. Attach a printout of your MATLAB code to your homework, and put your code in the Dropbox of D2L (`learn.ou.edu`). Run it as in part (a) and write your results. What are the norms of the error in this case?

(c) Write a MATLAB code `yourfirstname_yourfamilyname_gauss_elim_2.m` that performs Gaussian elimination with partial pivoting followed by back substitution. The partial pivoting in your new code must be done by using a row vector, as explained on pages 163–164 of the book. Attach a printout of your MATLAB code to your homework, and put your code in the Dropbox of D2L. Run it as in part (a) and write your results. What are the norms of the error in this case?