MATH 2130 Midterm Test 2 — Solutions

Prof. A. Willms, Dept. of Mathematics and Statistics, University of Guelph

Feb. 27, 2019

SECTION A.
ANSWER QUESTIONS 1–8 IN THE SPACE PROVIDED.

4 pts. 1. Consider the problem of solving $f(x) = 0$, where x and $f(x)$ are both vectors in $\mathbb{R}^m$. Suppose you had an iterative method for this problem that was coded in MATLAB so that at the end of each iteration you had an old estimate, `xold`, and a new estimate, `x`, to the solution, as well as the value of f at both these estimates: `fxold` and `fx`, and a count, `K`, of the number of iterations already performed. Write the `while` statement you would use in MATLAB so that iterations continued until the the absolute distance between the two estimates is below a specified tolerance level `TOLX`, and so that the number of iterations does not exceed a specified maximum `MAXITER`.

Solution:

```
while norm(x - xold) >= TOLX && K < MAXITER
```

9 pts. 2. Consider the system of equations $Ax = b$ where $A = \begin{bmatrix} 4 & -1 & 0 \\ 1 & 3 & -1 \\ -2 & 1 & -3 \end{bmatrix}$ and $b = \begin{bmatrix} 2 \\ -1 \\ 6 \end{bmatrix}$. Using an initial estimate of the solution as $x^{(0)} = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix}$, find the next estimate, $x^{(1)}$, using

(a) the Jacobi algorithm

(b) the Gauss-Seidel algorithm

Solution:
For both algorithms the update equations are

$$x_1 = \frac{2 + x_2}{4}, \qquad x_2 = \frac{-1 - x_1 + x_3}{3}, \qquad x_3 = \frac{6 + 2x_1 - x_2}{-3},$$

where for Jacobi the old guess is used in the right hand sides, and for Gauss-Seidel, the most up-to-date values are used.

(a) One iteration of Jacobi gives

n	x_1	x_2	x_3
0	1	-1	2
1	$\frac{2-1}{4} = \frac{1}{4}$	$\frac{-1-1+2}{3} = 0$	$\frac{6+2-(-1)}{-3} = -3$

so $x^{(1)} = \begin{bmatrix} \frac{1}{4} \\ 0 \\ -3 \end{bmatrix}$.

(b) One iteration of Gauss-Seidel gives

n	x_1	x_2	x_3
0	1	-1	2
1	$\frac{2-1}{4} = \frac{1}{4}$	$\frac{-1-(1/4)+2}{3} = \frac{1}{4}$	$\frac{6+2(1/4)-(1/4)}{-3} = -\frac{25}{12}$

so $x^{(1)} = \begin{bmatrix} \frac{1}{4} \\ \frac{1}{4} \\ -\frac{25}{12} \end{bmatrix}$.

10 pts. 3. Write MATLAB code that implements the back substitution algorithm to solve $Ux = c$ where U is an upper triangular matrix. The first line of your function is given below. (You do not need to do any input checking; you can assume the input is valid. However, you should check for the case where the system is unsolvable and exit in that case with a warning.)

```
function x = backsubs(U,c)
```

Solution:

```
function x = backsubs(U,c)

n = size(U,1);
x = zeros(n,1);
for i=n:-1:1
    temp = c(i) - U(i,i+1:n)*x(i+1:n);
    if U(i,i) ~= 0.0
        x(i) = temp/U(i,i);
    else
        if temp ~= 0.0
            error('System has no solution');
        else
            x(i) = 0.0;  % x(i) is arbitrary, set to zero.
        end
    end
end
end
```

2 pts. 4. For what kinds of systems $Ax = b$ may it be faster to solve using either Jacobi or Gauss-Seidel rather than Gaussian elimination with partial pivoting?

Solution:
If the matrix A is sparse, that is, has lots of zeros, and if your implementation of Jacobi or Gauss-Seidel takes advantage of this, then it may be faster to solve such systems using these algorithms rather than Gaussian elimination with partial pivoting.

3 pts. 5. State and define the condition under which the Jacobi and Gauss-Seidel methods are guaranteed to converge to the solution of $Ax = b$.

Solution:
If A is strictly diagonally dominant, then both the Jacobi and Gauss-Seidel methods are guaranteed to converge. An $m \times m$ matrix A is strictly diagonally dominant if

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|, \qquad \text{for } i = 1, 2, \ldots, m.$$

8 pts. 6. Below is MATLAB code intended to implement Gaussian elimination with partial pivoting to solve $Ax = b$. The code that is written is completely correct. However, there are four areas that you need to complete (marked with numerals on the far left). In each of these I have given you the left side of the needed line; you need only fill in the right hand side of that one line. However, if you wish to code it in another way, possibly using more lines, you may strike out my code and write your own code in these areas.

```
function [x]=GEPP(A,b)
% GEPP - Solve Ax=b by Gaussian elimination with partial pivotting
% [x] = GEPP(A,b) will solve Ax=b by Gaussian elimination with partial pivoting.
%  INPUT:
%    A : square matrix size nxn
%    b : column vector size nx1
%  OUTPUT:
%    x : solution to Ax=b
n = size(A,1);
if n ~= size(A,2)
    error('A must be square')
end
```

```
    if any(size(b) ~= [n,1])
        error('b must be nx1')
    end
    A = [A,b];      % augment A with b
    for j = 1:n-1  % Loop through each column (except last).
        % Find max entry (and location) in column j of A (rows j to n).
        [m,k] = max(abs(A(j:n,j)));
        k = k+j-1;
        % Swap rows of [A|b] if necessary.
        if k ~= j

1           A([k,j],j:n+1) =

        end
        % check for singularity. If A(j,j) == 0 then exit with an error.
        if A(j,j) ~= 0
            % Determine the multipliers for rows j+1 to n.

2           m =


            % Update the (j+1)st through nth rows of [A|b].

3           A(j+1:n,j+1:n+1) =


        else
            error('A is singular')
        end
    end
    % Call backsubs to solve the now upper triangular system.

4    x =

    end
```

Solution:
The four lines of code are

```
A([k,j],j:n+1) = A([j,k],j:n+1);

m = A(j+1:n,j)/A(j,j);

A(j+1:n,j+1:n+1) = A(j+1:n,j+1:n+1) - m*A(j,j+1:n+1);

x = backsubs(A(:,1:n),A(:,n+1));
```

10 pts. 7. Newton's method can be used to solve the system of nonlinear equations $f(x) = 0$ (where f and x are vectors in $\mathbb{R}^m$). Use two terms of a Taylor series to develop the system of linear equations that must be solved in each iteration of Newton's method. Be sure to define all symbols in your answer.

Solution:
Let $x^{(n)}$ be an estimate of the solution of $f(x) = 0$. Take a Taylor series for $f(x)$ centred at $x^{(n)}$:

$$f(x) = f(x^{(n)}) + Df(x^{(n)}) \left(x - x^{(n)}\right) + \cdots,$$

where Df is the Jacobian matrix, that is, the matrix

$$Df = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_m} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_m} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_m} \end{bmatrix}.$$

Newton's method replaces the function $f(x)$ with these first two terms of the Taylor series and finds the next estimate $x^{(n+1)}$ as the place where this approximation of the function is zero:

$$0 = f(x^{(n)}) + Df(x^{(n)}) \left(x^{(n+1)} - x^{(n)} \right).$$

Setting $s_{n+1} = x^{(n+1)} - x^{(n)}$, the linear system of equations that is needed to be solved in each iteration of Newton's method is:

$$Df(x^{(n)}) s_{n+1} = -f(x^{(n)}).$$

5 pts. 8. Broyden's method for solving $f(x) = 0$, where $f : \mathbb{R}^m \to \mathbb{R}^m$, uses an approximation, A_n, to the Jacobian $Df(x^{(n)})$. Each iteration, the approximation is updated by the formula

$$A_n = A_{n-1} + \frac{\left[f(x^{(n)}) - f(x^{(n-1)}) - A_{n-1} s_n \right] s_n^T}{\|s_n\|^2}, \tag{1}$$

where $s_n = x^{(n)} - x^{(n-1)}$.

The Shermann-Morrison formula for any invertible matrix A and nonzero vectors u and v is

$$(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1} u v^T A^{-1}}{1 + v^T A^{-1} u}, \qquad \text{provided } v^T A^{-1} u \neq -1.$$

(a) If one wishes to use the Shermann-Morrison formula for Broyden's method, identify the vectors u and v that one would need to use to compute A_n^{-1}.

(b) What is the primary advantage of using the Shermann-Morrison formula when implementing Broyden's method?

Solution:

(a) The vectors are

$$u = \frac{f(x^{(n)}) - f(x^{(n-1)}) - A_{n-1} s_n}{\|s_n\|^2}, \qquad v = s_n.$$

(b) The primary advantage is speed. The Shermann-Morrison update to the estimate for the inverse of the Jacobian can be computed in $O(n^2)$ operations and the new estimate $x^{(n+1)}$ can be computed from it in another $O(n^2)$ operations. Thus in total, the new estimate in each iteration using the Shermann-Morrison formula is obtained in $O(n^2)$ operations. In contrast, without the Shermann-Morrison formula, Broyden's method would need to solve a linear system each iteration, which would cost $O(n^3)$ operations.

SECTION B.

ANSWER THESE QUESTIONS USING THE BUBBLE SHEET. You may tear off this page.

2 pts. 1. What advantage does Successive Over Relaxation (SOR) have over the Gauss-Seidel method? (Indicate the most correct answer.)

(a) The matrix does not need to be diagonally dominant.

(b) SOR works sometimes when Gauss-Seidel does not.

(c) SOR sometimes converges faster than Gauss-Seidel.

(d) SOR is less work per iteration than Gauss-Seidel.

(e) SOR can work on non-sparse matrices.

Solution:
The correct answer is (c).

2 pts. 2. Suppose $A = L+D+U$ where L is strictly lower triangular, D is diagonal, and U is strictly upper triangular. The Jacobi method (J) and the Gauss-Siedel method (G-S) for solving $Ax = b$ can be written as $x^{(n+1)} = Cx^{(n)} + d$, where

(a) for J: $C = -D(L + U)$ and $d = Db$; and for G-S: $C = -(D + L)U$ and $d = (D + U)b$.

(b) for J: $C = -D^{-1}U$ and $d = D^{-1}Ub$; and for G-S: $C = -(D + L + U)^{-1}U$ and $d = (D + L + U)^{-1}b$.

(c) for J: $C = -D^{-1}(L + U)$ and $d = D^{-1}b$; and for G-S: $C = -(D + U)^{-1}L$ and $d = (D + U)^{-1}b$.

(d) for J: $C = -(D + L)U$ and $d = (D + L)b$; and for G-S: $C = -(D + L)^{-1}U$ and $d = (D + L)^{-1}b$.

(e) for J: $C = -D^{-1}(L + U)$ and $d = D^{-1}b$; and for G-S: $C = -(D + L)^{-1}U$ and $d = (D + L)^{-1}b$.

Solution:
The correct answer is (e).

2 pts. 3. Gaussian elimination (GE) is usually implemented with partial pivoting because (indicate the most correct answer)

(a) it makes the algorithm $O(n^3)$.

(b) it can sometimes find solutions when GE alone cannot.

(c) it reduces round-off error.

(d) it is faster.

(e) it works better on sparse matrices.

Solution:
The correct answer is (c).

2 pts. 4. The Guass-Thomas method is used to solve

(a) $Ax = b$ by an iterative procedure that terminates when $\left\|x^{(n+1)} - x^{(n)}\right\| < TOLX$.

(b) $f(x) = 0$ when f and x are vectors in $\mathbb{R}^m$.

(c) $Ax = b$ when A is tridiagonal.

(d) $Ux = c$ when U is upper triangular.

(e) $Ax = b$ when A is sparse.

Solution:
The correct answer is (c).

DID YOU ANSWER THE QUESTIONS IN SECTION B USING THE BUBBLE SHEET? IF NOT, YOU WILL GET ZERO FOR THEM.