AMATH 301 Midterm

Skyler Riki Hallinan

TOTAL POINTS

66.5 / 75

QUESTION 1

Question 1 10 pts

1.1 Part a 3 / 3

✓ - 0 pts **Correct**

- 3 pts Incorrect

1.2 Part b 6 / 7

- 0 pts Correct
- 7 pts says not possible
- 2 pts incorrect subtraction
- 2 pts incorrect multiplication
- 2 pts No/wrong matlab
- 1 pts Minor arithmetic error

✓ - 1 pts **Minor Matlab error**

- 0.5 pts Asterisks
- 0.25 pts incorrect matlab matrices

QUESTION 2

2 Question 2 10 / 10

✓ - 0 pts **Correct**

- 2 pts lu
- 2 pts P*b
- 2 pts y=L\(P*b)
- 2 pts U\y
- 2 pts Function syntax
- 0.5 pts Asterisk
- 1 pts Parenthesis
- 2 pts U-L flip
- 1 pts inv syntax
- 2 pts backslash
- 10 pts blank
- 2 pts order flip

QUESTION 3

3 Question 3 10 / 10

✓ - 0 pts **Correct**

- 0 pts No point loss: Matlab spells it "zeros", not "zeroes", so you would get an error here.
- 0.5 pts Incorrect Matlab syntax: brackets/parethesis
- 1 pts Incorrect dimensions
- 1 pts Incorrect Matlab syntax: Requires ^ for exponentiation
- 1 pts Incorrect Matlab syntax: Need to use dot operator for element-wise multiplication/exponentiation
- 1 pts Incorrect Matlab syntax: Incorrect indexing of vector
- 1 pts Other Incorrect Matlab Syntax
- 1 pts Inconsistent variable names
- 2 pts Other incorrect calculation
- 1 pts Incorrect Matlab function name

QUESTION 4

Question 4 12 pts

4.1 Part a 3 / 3

✓ - 0 pts **Correct**

- 0 pts Minor misspelling of function name
- 0.5 pts Incorrect indexing of vector
- 1 pts Confusion of function input/output
- 2 pts Incorrect calculation
- 1 pts Incorrect Matlab syntax
- 0.5 pts Inconsistent variable names
- 3 pts blank/no credit

4.2 Part b 3 / 3

✓ - 0 pts **Correct**

- 0.5 pts Incorrect indexing of vector
- 0.5 pts Misspelling function name
- 2 pts Incorrect Calculation
- 0.5 pts Incorrect Matlab Syntax
- 1 pts Confusion of function input/output

- **0.5 pts** Inconsistent variable names
- **3 pts** Blank/No Credit

4.3 Part c 1 / 3

- **0 pts** Correct
- **0.5 pts** Incorrect indexing of vector
- **0.5 pts** Incorrect Matlab syntax
- **0.5 pts** Inconsistent variable names
- **1 pts** Confusion of function input/output
- ✓ **2 pts** **Incorrect calculation**
- **3 pts** Blank/No credit

> This is only a quadratic (2nd-order) fit. You need the 9th order fit.

4.4 Part d 3 / 3

- ✓ **0 pts** **Correct**
- **1 pts** Mostly correct
- **2 pts** Some correct statements
- **3 pts** Blank/No credit

QUESTION 5

5 Question 5 8 / 8

- ✓ **0 pts** **Correct**
- **5 pts** Did not define f in Matlab
- **6 pts** Did not use fzero
- **6 pts** Wrote your own algorithm (possibly using fzero incorrectly)
- **2 pts** Wrong initial guess
- **2 pts** Missing @(x) syntax in function definition
- **2 pts** Didn't include one of the arguments of fzero
- **1 pts** Used e^ instead of exp()
- **0.5 pts** Minor syntax error (explained in comments)
- **1 pts** Medium syntax error (explained in comments)
- **2 pts** Major syntax error (explained in comments)
- **8 pts** No answer

QUESTION 6

Question 6 12 pts

6.1 Part a 4 / 4

- ✓ **0 pts** **Correct**
- **3 pts** Used the Jacobi method
- **1 pts** Used undefined variable (besides A)
- **1.5 pts** Reversed upper/lower triangular parts
- **1.5 pts** Included diagonal entries twice
- **3.5 pts** Used lu
- **3.5 pts** P + T = A, but neither matrix correct
- **0.5 pts** Used diag(A) instead of diag(diag(A))
- **2 pts** Correct matrices, but not Matlab code
- **4 pts** Neither matrix correct, P + T =\= A
- **2 pts** Only defined one matrix
- **0.5 pts** Minor syntax error (explained in comments)
- **1 pts** Major syntax error (explained in comments)
- **4 pts** Did not answer question

6.2 Part b 2.5 / 4

- **0 pts** Correct
- **1 pts** Wrong definition of SDD (did not sum off-diag entries)
- **1 pts** Wrong definition of SDD (squared entries instead of absolute value)
- **1 pts** Wrong definition of SDD (abs(sum) instead of sum(abs))
- **1 pts** Wrong definition of SDD (missing absolute value)
- **2 pts** Wrong definition of SDD (other)
- **1.5 pts** Wrong definition of SDD (sum all diagonal elements)
- **2 pts** Said system was SDD, no explanation
- **2 pts** Did not answer SDD part
- ✓ **1.5 pts** **Said "not SDD implies the method does not converge"**
- **1.5 pts** Said system was SDD but couldn't determine if method converged
- **2 pts** Said "not SDD implies the method converges"
- **2 pts** Convergence answer unrelated to SDD
- **2 pts** Said that SDD does not apply to GS method
- **2 pts** Said SDD implies the method does not converge
- **2 pts** Didn't answer convergence part
- **0.5 pts** Arithmetic error
- **4 pts** No answer

6.3 Part c 4 / 4

- ✓ **0 pts** **Correct**
- **0.5 pts** Defined M incorrectly

- **1 pts** Did not define M
- **2 pts** Did not use eig or eigs
- **0.5 pts** Used other norm of eigenvalues instead of abs
- **1 pts** Did not find magnitude of eigenvalue(s)
- **1 pts** Could not tell if GS converged
- **1 pts** Didn't answer convergence part
- **1 pts** Checked eigenvalues of A instead of M
- **2 pts** Wrote unrelated algorithm
- **3.5 pts** No code
- **0.5 pts** Used eigenvectors instead of eigenvalues
- **0.5 pts** Syntax error
- **4 pts** Did not answer question

QUESTION 7

Question 7 13 pts

7.1 Part a 4 / 4

✓ **- 0 pts Correct**

- **1 pts** Correct error. Both the correct and the incorrect fix written.
- **2 pts** Correct error, wrong fix
- **4 pts** Incorrect error
- **4 pts** No answer

7.2 Part b 5 / 5

✓ **- 0 pts Correct**

- **1 pts** Small loop mistake
- **2 pts** Correct and incorrect answers given.
- **2.5 pts** Correct Error, incorrect fix
- **4 pts** Concept good, code wouldn't run.
- **5 pts** Incorrect error

7.3 Part c 0 / 4

- **0 pts** Correct
- **1 pts** Syntax error
- **2 pts** Correct error, incorrect fix

✓ **- 4 pts Incorrect error**

AMATH 301 Midterm Exam

Name: Skyler Hallinan

Student NetID: 1732227

Question	Points	Score
1	10	
2	10	
3	10	
4	12	
5	8	
6	12	
7	13	
Total:	75	

Please write your name on every page you want us to grade.

Many of these questions ask you to write short segments of code. You should try your best to write functioning Matlab code. That means proper use of arithmetic symbols, parentheses, brackets, builtin function names, semicolons, etc.

Name: _____
NetID: _____

Useful Matlab Commands

Syntax of some useful Matlab commands:

- `save(filename, var, format)` stores the specified variable `var` in a file named `filename`. `format` may be `'-mat'` for a binary MAT-file format (default) or `'-ascii'` for 8-digit ASCII format.
- `plot(X,Y,LineSpec,value)` plots vector `X` versus vector `Y`. `LineSpec` and `value` together specify the line type, marker symbol and color.
- `x = a:b:c` returns a vector `x` with entries `a`, `a + b`, `a + 2*b`, etc. The last entry is less than or equal to `c`.
- `exp(x)`, `log(x)`, `sqrt(x)` return the exponential of `x` (e^x), the natural log of `x` ($\ln x$) and the square root of x ($\sqrt{x}$).
- `[val, ind] = max(x)`, `[val, ind] = min(x)` returns the maximum/minimum value of a vector `x` as val and the index of that value in `x` as `ind`. In particular, `x(ind) == val`.
- `abs(x)` returns the magnitude of `x` each value in the vector `x`.
- `[x,fval]=fminsearch(fun,x0)` starts at the initial guess `x0` and returns a value `x` that is a local minimizer of the function described in `fun`. The value `fval` is equal to `fun(x)`.
- `norm(X,p)` and `cond(X,p)` give the p-norm and condition number of `X` respectively. `norm(X, Inf)` is equivalent to `max(abs(X))`
- `X'`, `X.'` return the transpose and conjugate transpose of the 2D matrix `X`.
- `inv(X)` returns the inverse of the square matrix `X`.
- `sum(x)`, `mean(x)` and `prod(x)` return the sum, average and product of the elements of the vector `x`.
- `diag(A)` returns the diagonal of the matrix `A` or creates a square matrix with the vector `A` on the diagonal.
- `zeros(n1,...,nk)`, `ones(n1,...,nk)` returns an $n1$ x $\cdots$ x nk array of 0's or 1's.
- `A\b` solves the linear system $A\mathbf{x} = \mathbf{b}$. You can assume that Matlab employs back/forward substitution if `A` is triangular and Gaussian elimination otherwise.
- `triu(X)` and `tril(X)` returns the upper and lower triangular parts of `X`, respectively. (Both include the diagonal of X.)
- `[L,U,P]=lu(A)` returns an upper triangular matrix in `U`, lower triangular matrix `L` with unit diagonal and permutation matrix `P` such that $LU = PA$.

Name: _____
NetID: _____

- `[V,D]=eig(A)` returns a matrix `V` whose columns are eigenvectors of `A` and a diagonal matrix `D` whose entries are the corresponding eigenvalues of `A`.
- `p=polyfit(x,y,n)` finds the coefficients of a polynomial `p` of degree `n` that fits the data in a least squares sense. The result `p` is a row vector of length `n+1` containing the polynomial coefficients in descending powers.
- `y=polyval(p,x)` returns the value of a polynomial `p` evaluated at `x`.
- `yf=spline(x,y,xf)` uses a cubic Hermite spline interpolation to find `yf` at `xf` given `y` as a function of `x`.
- `function_name = @(variable_names)(formula)` creates an anonymous function. `variable_names` should be a comma separated list.

Name: Skyler Hallman
NetID: 1732227

1. (10 points) Let

$$A = \begin{pmatrix} 2 & 3 & 1 \\ -1 & 1 & 0 \\ 4 & 0 & -1 \end{pmatrix}, \quad B = \begin{pmatrix} 3 & 1 \\ -2 & 2 \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } \mathbf{y} = \begin{pmatrix} 0 \\ 2 \end{pmatrix}.$$

Decide if each of the following expressions makes sense. If it does, calculate the result by hand and write a *single line* of Matlab code to compute the result. (You can assume that `A`, `B`, `x` and `y` have already been defined in Matlab.)

(a) $A(3\mathbf{x} - \mathbf{y})$

A is 3x3 matrix (3x-y) is 2x1

Inner dimensions must match before multiplying vectors

3 ≠ 2 therefore does not make sense

(b) $B(3\mathbf{x} - \mathbf{y})$

B is 2x2 matrix. (3x-y) is 2x1 matrix.

Inner dimensions match so multiplication can happen. Makes sense

$$3\mathbf{x} - \mathbf{y} \to 3\begin{pmatrix} 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 2 \end{pmatrix} \to \begin{pmatrix} 3 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 2 \end{pmatrix} \to \begin{pmatrix} 3 \\ -2 \end{pmatrix}$$

$$B \times (3\mathbf{x} - \mathbf{y}) \to \begin{pmatrix} 3 & 1 \\ -2 & 2 \end{pmatrix} \times \begin{pmatrix} 3 \\ -2 \end{pmatrix} \to \begin{pmatrix} (3\times 3) + (1 \times -2) \\ (-2 \times 3) + (2 \times -2) \end{pmatrix} \to \begin{pmatrix} 7 \\ -10 \end{pmatrix}$$

$$B(3\mathbf{x} - \mathbf{y}) = \begin{pmatrix} 7 \\ -10 \end{pmatrix}$$

Matlab code

```
A*(3*x-y)
```

$LU = PA$ $y = Ux$
$PAx = Pb$
$LUx = Pb$ $Ly = Pb$

Name: Skyler Hallinan
NetID: 1732227

2. (10 points) Complete the following function to compute the solution of the linear system $A\mathbf{x} = \mathbf{b}$ using LU decomposition.

```
function x = lu_decomposition(A, b)
```

```
[L,U,P] = lu(A)
y = L\(P*b);
x = U\y;
```

```
end
```

3. (10 points) Write Matlab code to create a 100×1 vector whose entries are the first 100 squares. That is, the first entry should be 1, then 4, then 9, etc. and the final entry should be $100^2 = 10,000$.

```
Vector1 = zeros(100,1)
for k = 1:100
vector1(k) = k^2;
end
```

polyfit(x,y,1)

Name: Skylar Hollimon
NetID: 1732227

4. (12 points) Suppose you have defined two vectors x and y, each containing 10 values. You can assume that the vector x does not contain the values 2 or 100.

(a) Write Matlab code to find the slope and y-intercept of the best fit line for this data. You can put the slope and y-intercept in a single vector if it is more convenient.

```
coeffs = polyfit(x,y,1);
slope = coeffs(1);
yint = coeffs(2);
```

(b) Write Matlab code to substitute $x = 100$ into the best fit line you found in part (a).

```
y100 = polyval(coeffs, 100);
```

(c) Write Matlab code to find the coefficients of the best fit 9th order polynomial for this data, then write Matlab code to substitute $x = 2$ into this polynomial.

```
Coeffs = polyfit(x,y,2)
% we have form y = ax^2 + bx + c
a = Coeffs(1);
b = Coeffs(2);
c = Coeffs(3);
y2 = polyval(coeffs,2);
```

(d) Do you think the 9th order polynomial will give a good prediction for the value of y when $x = 2$? Why or why not?

No, using 9th degree polynomial will not give good fit. The model will be over constraining the function. Will give inaccurate estimate of model as it will try to exactly fit every point on some weird curve. Prefer to use lower order fit.

Name: Skyler Hallman
NetID: 1732227

5. (8 points) Matlab has a builtin function called `fzero` that uses a combination of several methods (including the bisection method) to find the root of a nonlinear function. You probably haven't seen this function before, but with access to the help file you should be able to use its basic methods. Below is a portion of the help file for `fzero`:

```
fzero
Root of a nonlinear function

Syntax
x = fzero(fun, x0)

Description
x = fzero(fun, x0) tries to find a point x where fun(x) = 0. This solution
is where fun(x) changes sign fzero cannot find a root of a function such
as x^2.

Input Arguments
fun Function to solve, specified as a handle to a scalar-valued function
or the name of such a function. fun accepts a scalar x and returns a
scalar fun(x).

x0 Initial value, specified as a real scalar. fzero begins at x0 and
tries to locate a point x1 where fun(x1) has the opposite sign of fun(x0).
Then fzero iteratively shrinks the interval where fun changes sign to
reach a solution.
```

Use the function `fzero` to find a root of the function

$$f(x) = \cos(4x) + e^{\sqrt{x^2+2x+1}}.$$

Use an initial guess of 10.

```
f = @(x)(cos(4*x) + exp(sqrt(x^2+2*x+1)));
fzero = (f, 10);
```

Name: Skyler Hillman
NetID: 1732227

6. (12 points) Suppose that we want to solve the following system of equations using a matrix splitting method.

A x b

$$\begin{pmatrix} 2 & 1 & -2 \\ 2 & -4 & 1 \\ 0 & 2 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \\ 6 \end{pmatrix}.$$

(a) In the Gauss-Seidel method, you need to find matrices P and T such that $A = P+T$. Write Matlab code to define these two matrices. (You can use builtin commands or just define them directly.) A is defined.

Jacobi
P is lower triangular

```
P = tril(A);
T = A-P;
```

(b) Is this system strictly diagonally dominant? Does that tell you if the Gauss-Seidel method will converge?

The system is not strictly diagonal dominant. In SDD, the magnitude of diagonal will be greater than sum of other values on same row. On line 1, diagonal value is 2, but sum of |A(1,2)| and |A(1,3)| is 1+2=3, 3 > 2 so not strictly diagonal dominant. Shows that the Gauss-Seidel method will not converge

(c) The Gauss-Seidel formula can be written as $\mathbf{x}_{k+1} = M\mathbf{x}_k + \mathbf{c}$. Write Matlab code to find the magnitude of the eigenvalues of this matrix M. Would these eigenvalues tell you if the Gauss-Seidel method converged?

```
M = -P\T;
eig(M)
```

max(abs(eig(M))) < 1

If all of the absolute values of the eigen values were less than 1, then the Gauss-Seidel method will converge. Else, the Gauss-Seidel method does not converge.

Name: Skyler Hallinan
NetID: 173227

7. (13 points) The following code (in the file `jacobi.m`) is meant to implement the Jacobi method for the system from problem 6. It has several errors, which we will fix in turn.

```
1    clear all; close all; clc;
2
3    A = [2 1 -2; 2 -4 1; 0 2 4];
4    b = [1; -1; 6];
5
6    P = diag(diag(A);
7    T = A - P;
8
9    err = 1;
10   k = 1;
11   x = zeros(3, 2);
12   while err > 1e-8
13       x(:, k+1) = -P\(T*x(:,k) + b);
14       err = norm(x(:, k+1) - x(:, k), Inf);
15   end
```

The numbers on the left are just line numbers, not code. (Part (b) continues on the next page. For each of the following parts, clearly indicate any lines that you would need to change or add and what changes you would make. Only fix the problem from the given part, not all problems that you see.)

(a) The above code does not work. When you try to run it Matlab produces the following error:

`Error: File: jacobi.m Line: 6 Column: 17`
`Unbalanced or unexpected parenthesis or bracket.`

What is causing this error, and what would you have to change in `jacobi.m` in order to fix it?

On line 6 for defining P, there is a missing parantheses at the end of the statement, right before the semicolon. To fix this, add another parantesis to end of statement before semicolon on same line.

b

old: `P= diag(diag(A);`

New: `P= diag(diag(A));`

Name: Skyler Ullman
NetID: 1732227

(b) After you correct the above error, your code appears to run forever. When you finally stop your code with Ctrl-C and check the matrix `x`, you find that it only has two columns and neither is the solution to `Ax = b`. What is causing this problem, and what would you have to change in `jacobi.m` in order to fix it? (You can assume that the Jacobi method is supposed to converge for this problem.)

Increment k

x is initially defined as a 3x2 matrix of zeros, where each successive guess can be inputted into successive columns of the matrix. Line 13 shows this x(:,k+1) = -P\(T*x(:,k)+b); However, this needs to include an incrementation of k in the while loop. In the code, k is remaining constant at k=1 and is not incremented. As a result, in the while loop, the guess is always comparing to the same column 1 value of zeros in the guess, and storing the new guess in the second column. Because the k does not change, this process just keeps repeating, because it uses the same column 1 guess of the zeros x matrix, and stores the new value in column 2 infinitely, which leads to no solution and an infinite loop. To fix this, insert k=k+1; after line 14 so that is within the while loop so that new guesses can be stored in the x matrix.

(c) After you correct the previous two errors, your code stops after 67 steps. The columns of `x` appear to converge to the vector `[-1; -1; -1]`, but it is easy to check that the actual solution is `[1; 1; 1]`. What is causing this problem, and what would you have to change in `jacobi.m` in order to fix it?

The reason for this problem is the lack of absolute value signs when checking for the error distance in line 14. In order to fix this, need to insert abs around (x(:,k+1) - x(:,k)) to get correct values. This would fix incorrect sign values on x matrix.