

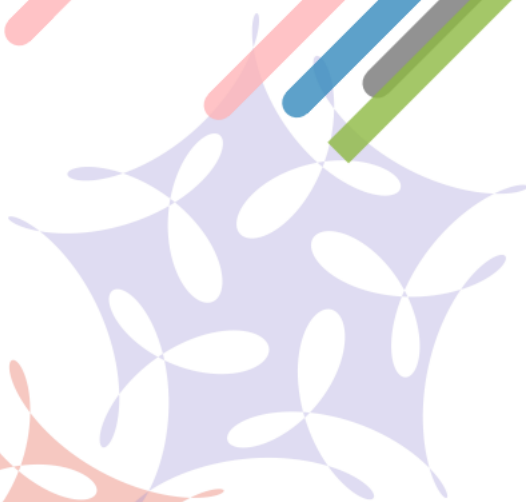
Tut III of MA2213

Qian Lilong

Department of Mathematics
National University of Singapore

October 8, 2018

qian.lilong@nus.edu



1 Solving Linear system

1.1 Question 1

A matrix $A = (a_{ij})_{n \times n}$ is called a lower triangular matrix if $a_{ij} = 0$ for all $i < j$. Prove the following statements.

- (a) The product of two $n \times n$ lower triangular matrices is lower triangular.
- (b) The product of two $n \times n$ upper triangular matrices is upper triangular.
- (c) The inverse of a nonsingular $n \times n$ lower (respectively upper) triangular matrix is lower (respectively upper) triangular. (Hint: Consider solving for columns of the matrix B in $AB = I$ by applying the “forward substitution” procedure (cf. Question 1).)

Proof. (a) We can solve this problem by mathematical induction.

First, when $n = 1$, this is true.

Next, assume $n = k$ holds. We consider $n = k + 1$. Suppose

$$A = \begin{pmatrix} a_{11} & 0 \\ h & A_{22} \end{pmatrix}, B = \begin{pmatrix} b_{11} & 0 \\ f & B_{22} \end{pmatrix}, \quad (1)$$

where A_{22}, B_{22} are two lower triangular matrices of dimension $(k-1) \otimes (k-1)$ and h, f are two $(k-1)$ -dimension column vectors. Then

$$AB = \begin{pmatrix} a_{11}b_{11} & 0 \\ b_{11}h + A_{22}f & A_{22}B_{22} \end{pmatrix}. \quad (2)$$

Since A_{22} and B_{22} are both $(k-1)$ -dimension lower triangular matrix, by the assumption, their product remains a lower triangular matrix.

Since both the base case and the inductive step have been performed, the conclusion holds for any natural n .

- (b) Suppose A, B are two upper triangular matrices. Then A^\top, B^\top are two lower triangular matrices. By the conclusion of (a), we have

$$AB = (B^\top A^\top)^\top \quad (3)$$

is an upper triangular matrix.

- (c) We can also solve this by mathematical induction.

First, consider the case when $n = 1$. The inverse of a scalar is also a scalar. The conclusion holds.

Next, assume conclusion holds for k . We consider the case $k+1$.

Suppose A is a upper triangular with the block representation

$$A = \begin{pmatrix} a_{11} & f \\ 0 & A_{22} \end{pmatrix}, \quad (4)$$

where A_{22} is an upper triangular matrix of dimension $(k-1) \times (k-1)$, f is a $(k-1)$ -dimension row vector.

Suppose $B = A^{-1}$. Write B in the block representation accordingly:

$$B = \begin{pmatrix} b_{11} & h \\ g & B_{22} \end{pmatrix}, \quad (5)$$

where g, f^\top are two $(k-1)$ -dimension column vector and B_{22} is a $(k-1) \times (k-1)$ matrix.

Since $AB = I$, we have

$$\begin{aligned} a_{11}b_{11} &= 1, \\ a_{11}h + fB_{22} &= 0, \\ A_{22}g &= 0, \\ A_{22}B_{22} &= I_{k-1}. \end{aligned} \quad (6)$$

Since A is invertible, we have $g = 0$. And by the last in equality in eq. (6) and the assumption, B_{22} is an upper triangular matrix. Therefore, B is upper triangular.

By the mathematical induction, the proof completes. ■

1.2 Question 2

Prove that

(a) The LU decomposition of an invertible matrix A , i.e., $A = LU$ where L has 1's on its diagonal, is unique.

(b) If A is nonsingular and symmetric ($A = A^\top$) and $A = LU$ where L has 1's on its diagonal, then $U = DLT$, with D the diagonal matrix having the same diagonal entries as U .

Proof. (a) Suppose there are two LU decomposition

$$A = L_1U_1 = L_2U_2, \quad (7)$$

where L_1, L_2 are two lower triangular matrices with diagonal entries being 1. U_1, U_2 are two upper triangular matrices.

Since A is invertible, L_i, U_i are all invertible. We have

$$L_2^{-1}L_1 = U_2U_1^{-1}. \quad (8)$$

Note that L_2^{-1} is lower triangular and U_1^{-1} is upper triangular by the results in question 1. We have $L_2^{-1}L_1$ is lower triangular and $U_2U_1^{-1}$ is upper triangular. Hence $L_2^{-1}L_1$ is both lower and upper triangular, which must be diagonal.

Compute the diagonal entries of $L_2^{-1}L_1$, which is an identity matrix. Therefore, $L_2^{-1}L_1 = I$, i.e., $L_1 = L_2$.

(b) Suppose $A = LU$, where L is lower triangular with diagonal being 1 and U is upper triangular. By the symmetry of A , we have

$$LU = U^\top L^\top = U_1^\top D^{-\top} L^\top, \quad (9)$$

where

$$U = \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & u_{nn} \end{pmatrix}, D = \begin{pmatrix} u_{11} & 0 & \cdots & 0 \\ 0 & u_{22} & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & u_{nn} \end{pmatrix}, U_1 = D^{-1}U. \quad (10)$$

Note that A is invertible, hence $U_{ii} \neq 0$. Forward, D is invertible and diagonal.

Now U_1^\top is a lower diagonal matrix with diagonal entries being 1, by the uniqueness of LU decomposition in (a), we have

$$U = D^\top L. \quad (11)$$

■

1.3 Question 3

Solve the linear equation $Ax = b$ in four-digit rounding floating point arithmetic, where

$$A = \begin{pmatrix} 0.00211 & 0.08204 \\ 0.337 & 12.84 \end{pmatrix}, b = \begin{pmatrix} 0.04313 \\ 6.757 \end{pmatrix} \quad (12)$$

Proof. (a) Gaussian elimination; m_{21} is 159.7. The new A is:

$$\begin{pmatrix} 0.00211 & 0.08204 \\ 0 & -0.26 \end{pmatrix} \quad (13)$$

The new b is:

$$\begin{pmatrix} 0.04313 \\ -0.131 \end{pmatrix} \quad (14)$$

By back forward substitution :

$$x = \begin{pmatrix} 0.8531 \\ 0.5038 \end{pmatrix} \quad (15)$$

(b) Gaussian elimination with partial pivoting

Change 1-row and 2-th row. New A after row exchange

$$\begin{pmatrix} 0.337 & 12.84 \\ 0.00211 & 0.08204 \end{pmatrix}$$

New b after row exchange

$$\begin{pmatrix} 6.757 \\ 0.04313 \end{pmatrix}$$

m(21) is 0.006261 The new A is:

$$\begin{pmatrix} 0.337 & 12.84 \\ 0 & 0.00165 \end{pmatrix} \quad (16)$$

The new b is:

$$\begin{pmatrix} 6.757 \\ 0.00082 \end{pmatrix}$$

By the backward substitution,

$$x = (1.1160.497)$$

(c) Gaussian elimination with scaled partial pivoting

Gaussian Elimination with scaled partial pivoting Since

$$0.002110/0.08204 < 0.3370/12.84; \quad (17)$$

so, row exchange is needed. The result is same as that in part (b)

Multiply the first equation by 1000 and do (b) and (c) for the resulting system

(b') The result is same as (b)

(c') The result is same as (c);

■

1.4 Question 4

Use Gaussian elimination with partial pivoting and three-digit chopping arithmetic to solve the following linear system and compare the approximation to the actual solution:

$$Ax = b \quad (18)$$

, where

$$A = \begin{pmatrix} 3.03 & -12.1 & 14 \\ -3.03 & 12.1 & -7 \\ 6.11 & -14.2 & 21 \end{pmatrix}, b = \begin{pmatrix} -119 \\ 120 \\ -139 \end{pmatrix}. \quad (19)$$

Proof. The exact solution can be solved:

$$x = \begin{pmatrix} 0 \\ 10 \\ \frac{1}{7} \end{pmatrix}. \quad (20)$$

The results for computing by GS partial pivoting are:

1	change 1-row and 3-th row			
2	New A after row exchange			
3	6.11	-14.2		21
4	-3.03	12.1		-7
5	3.03	-12.1		14
6				

```

7 New b after row exchange
8   -139
9    120
10   -119
11
12 m(21) is -0.495
13 m(31) is 0.495
14 The new A is:
15     6.11      -14.2       21
16     0        -0.581     -0.597
17     0         0.581      0.37
18
19 The new b is:      -139
20     -0.568
21      0.569
22
23   change 2-row and 3-th row
24 New A after row exchange
25     6.11      -14.2       21
26     0         0.581      0.37
27     0        -0.581     -0.597
28
29 New b after row exchange
30     -139
31      0.569
32     -0.568
33
34 m(32) is -1
35 The new A is:
36     6.11      -14.2       21
37     0         0.581      0.37
38     0          0       -0.226
39
40 The new b is:      -139
41      0.569
42      0.1
43
44 x(3) is -0.442 new b is -0.138
45 new b is 0.732
46 new b is 0.1
47 x(2) is 0.125 new b is 0.389
48 new b is 0.732
49 new b is 0.1
50 x(1) is 0.636 new b is 0.389
51 new b is 0.732
52 new b is 0.1
53
54 x =
55
56     0.636
57     0.125
58    -0.442
59

```



1.5 Question 5

Solve the following linear system of equations by Gaussian elimination with partial pivoting with four-digit arithmetic with rounding

$$Ax = b \quad (21)$$

, where

$$A = \begin{pmatrix} 0.003 & 59.14 & 0 & 0 & 0 \\ 5.291 & -6.13 & 0 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0 \\ 0 & 0 & 0 & 0.00211 & 0.08204 \\ 0 & 0 & 0 & 0.337 & 12.84 \end{pmatrix}, b = \begin{pmatrix} 59.17 \\ 46.78 \\ 1 \\ 0.04313 \\ 6.757 \end{pmatrix} \quad (22)$$

The results for solving this linear equation are:

```

1  change 1-row and 2-th row
2  New A after row exchange
3      5.291      -6.13      0      0      0
4      0.003      59.14      0      0      0
5      0      0      0.6      0      0
6      0      0      0      0.00211      0.08204
7      0      0      0      0.337      12.84
8
9  New b after row exchange
10     46.78
11     59.17
12     1
13     0.04313
14     6.757
15
16 m(21) is 0.000567
17 m(31) is 0
18 m(41) is 0
19 m(51) is 0
20 The new A is:
21     5.291      -6.13      0      0      0
22     0      59.14      0      0      0
23     0      0      0.6      0      0
24     0      0      0      0.00211      0.08204
25     0      0      0      0.337      12.84
26
27 The new b is:      46.78
28     59.14
29     1
30     0.04313
31     6.757
32
33 change 2-row and 2-th row
34 New A after row exchange
35     5.291      -6.13      0      0      0
36     0      59.14      0      0      0
37     0      0      0.6      0      0
38     0      0      0      0.00211      0.08204
39     0      0      0      0.337      12.84
40
41 New b after row exchange
42     46.78
43     59.14
44     1
45     0.04313
46     6.757
47
48 m(32) is 0
49 m(42) is 0
50 m(52) is 0
51 The new A is:
52     5.291      -6.13      0      0      0
53     0      59.14      0      0      0
54     0      0      0.6      0      0
55     0      0      0      0.00211      0.08204

```

```

56         0         0         0         0.337         12.84
57
58 The new b is:         46.78
59         59.14
60         1
61         0.04313
62         6.757
63
64 change 3-row and 3-th row
65 New A after row exchange
66         5.291         -6.13         0         0         0
67         0         59.14         0         0         0
68         0         0         0.6         0         0
69         0         0         0         0.00211         0.08204
70         0         0         0         0.337         12.84
71
72 New b after row exchange
73         46.78
74         59.14
75         1
76         0.04313
77         6.757
78
79 m(43) is 0
80 m(53) is 0
81 The new A is:
82         5.291         -6.13         0         0         0
83         0         59.14         0         0         0
84         0         0         0.6         0         0
85         0         0         0         0.00211         0.08204
86         0         0         0         0.337         12.84
87
88 The new b is:         46.78
89         59.14
90         1
91         0.04313
92         6.757
93
94 change 4-row and 5-th row
95 New A after row exchange
96         5.291         -6.13         0         0         0
97         0         59.14         0         0         0
98         0         0         0.6         0         0
99         0         0         0         0.337         12.84
100        0         0         0         0.00211         0.08204
101
102 New b after row exchange
103         46.78
104         59.14
105         1
106         6.757
107         0.04313
108
109 m(54) is 0.006261
110 The new A is:
111         5.291         -6.13         0         0         0
112         0         59.14         0         0         0
113         0         0         0.6         0         0
114         0         0         0         0.337         12.84
115         0         0         0         0         0.00165
116
117 The new b is:         46.78
118         59.14
119         1
120         6.757
121         0.00082
122
123 x(5) is 0.497
124 x(4) is 1.116

```



```

125 x(3) is 1.667
126 x(2) is 1
127 x(1) is 10
128
129 x =
130
131      10
132      1
133      1.667
134      1.116
135      0.497
136
137

```

1.6 Question 6

Proof. After one step Gauss elimination with partial pivoting we have

$$\left[\begin{array}{ccc|c} 6 & \alpha & 10 & 5 \\ 2 & 1 & 3 & 1 \\ 4 & 6 & 8 & 5 \end{array} \right] \Rightarrow \left[\begin{array}{ccc|c} 6 & \alpha & 10 & 5 \\ 0 & 1 - \frac{\alpha}{3} & -\frac{1}{3} & -\frac{2}{3} \\ 0 & 6 - \frac{2}{3}\alpha & \frac{4}{3} & \frac{5}{3} \end{array} \right].$$

- If $\alpha = 6$ The augmented matrix is

$$\left[\begin{array}{ccc|c} 6 & \alpha & 10 & 5 \\ 0 & -1 & -\frac{1}{3} & -\frac{2}{3} \\ 0 & 2 & \frac{4}{3} & \frac{5}{3} \end{array} \right].$$

Since $|-1| < 2$, we need to exchange the second and last row in the next step.

- If $\alpha = 9$

The augmented matrix is

$$\left[\begin{array}{ccc|c} 6 & \alpha & 10 & 5 \\ 0 & -2 & -\frac{1}{3} & -\frac{2}{3} \\ 0 & 0 & \frac{4}{3} & \frac{5}{3} \end{array} \right].$$

Since $|-2| > 0$, we need not to exchange the second and last row in the next step.

- If $\alpha = -3$

The augmented matrix is

$$\left[\begin{array}{ccc|c} 6 & \alpha & 10 & 5 \\ 0 & 2 & -\frac{1}{3} & -\frac{2}{3} \\ 0 & 8 & \frac{4}{3} & \frac{5}{3} \end{array} \right].$$

Since $2 < 8$, we need to exchange the second and last row in the next step.

it is clear that only for $\alpha = 9$ there will be no row interchange required when solving this system using Gaussian Elimination with partial pivoting. ■

2 The MATLAB code for performing the Gauss elimination

2.1 Gauss elimination with rounding or chopping arithmetic

The source file is

```
1      function x      = GS_round(A,b,digit,type)
2  %% function to perform the Gauss elimination with rounding or chopping
   arithmetics
3  input: Ax = b
4  %      digit, how many digits to keep when using rounding or chopping
5  %      type, be 'round' or 'chop', which is a string
6  if ~exist('digit','var')
7      digit      = 4;
8  end
9  if ~exist('type','var')
10     type      = 'round';    % using rounding arithmetic
11 end
12 n      = length(A); % dimension of A
13 global d t
14 d      = digit;
15 t      = type;
16 %% perform GS elimination
17 for i      = 1:n-1
18     for j      = i+1:n
19         mji      = tdiv(A(j,i),A(i,i));
20         fprintf('m(%d%d) is %g\n',j,i,mji);
21                 % -mji row i + row j
22
23         for k      = i+1:n
24             A(j,k) = tsub(A(j,k),    tmulti(mji,A(i,k)));
25
26         end
27         b(j)      = tsub(b(j),tmulti(mji,b(i)));
28     end
29     A(i+1:n,i)    =0;
30     fprintf('The new A is:\n');
31     disp(A);
32     fprintf('The new b is:');
33     disp(b);
34 end
35 %% call backward substitution
36 x      = backsub(A,b);
37 end
38
39
40 function y      = tadd(x,y)    %
41 x      = tround(x);
42 y      = tround(y);
43 y      = tround(x+y);
44 end
45
46 function y      = tmulti(x,y)
47 x      = tround(x);
48 y      = tround(y);
49 y      = tround(x*y);
50 end
51 function y      = tdiv(x,y)
52 x      = tround(x);
53 y      = tround(y);
54 y      = tround(x/y);
55 end
56
57 function y      = tsub(x,y)
58 x      = tround(x);
```

```

59 y          = tround(y);
60 y          = tround(x-y);
61 end
62
63 function y   = tround(x)
64 global d t
65 switch t
66     case 'round'
67         y   = round(x,d,'significant');
68     otherwise
69         ex   = ceil(log10(abs(x)));
70         x    = x/(10^ex);
71         y    = fix(x*(10^d))/(10^d);
72         x    = x*(10^ex);
73 end
74
75 end
76
77 function x    = backsub(A,b)
78 %% A is a upper triangular matrix
79 n = length(A);
80 x = zeros(n,1);
81 for i = n:-1:1
82     x(i) = tround(b(i)/A(i,i));
83     fprintf('x(%d) is %g ',i,x(i));
84     for j = 1:i-1
85         b(j) = tround(b(j) - tmulti(A(j,i),x(i)));
86     end
87     fprintf('new b is %g\n',b);
88
89 end
90 end

```

2.2 Gauss elimination with partial pivoting

The source file is

```

1  function x = GS_partial(A,b,digit,type)
2  %% function to perform the Gauss elimination with partial
3  %   pivoting and rounding or chopping arithmetics
4  %   input: Ax = b
5  %   digit, how many digits to keep when using rounding or chopping
6  %   type, be 'round' or 'chop', which is a string
7  if ~exist('digit','var') % the default value for digit
8      digit = 4;
9  end
10 if ~exist('type','var') % by default, using rounding arithmetic
11     type = 'round';      % using rounding arithmetic
12 end
13 n = length(A);          % dimension of A
14 global d t
15 d = digit;
16 t = type;
17 %% perform GS elimination
18 for i = 1:n-1
19     %% partial pivoting
20     % largest value of A(i,j);
21     v = A(i,i:end);
22     v = abs(v);
23     [~,idx] = max(v);
24     idx = idx + i-1;
25     fprintf(' change %d-row and %d-th row\n',i,idx);
26     v = A(i,:);
27     A(i,:) = A(idx,:);
28     A(idx,:) = v;

```

```

29     fprintf('New A after row exchange\n');
30     disp(A);
31                                     % exchange b also
32     v          = b(i);
33     b(i)       = b(idx);
34     b(idx)     = v;
35     fprintf('New b after row exchange\n');
36     disp(b);
37     for j      = i+1:n
38         mji    = tdiv(A(j,i),A(i,i));
39         fprintf('m(%d%d) is %g\n',j,i,mji);
40                                     % -mji row i + row j
41
42         for k  = i+1:n
43             A(j,k) = tsub(A(j,k), tmulti(mji,A(i,k)));
44
45         end
46         b(j)     = tsub(b(j),tmulti(mji,b(i)));
47     end
48     A(i+1:n,i)  =0;
49     fprintf('The new A is:\n');
50     disp(A);
51     fprintf('The new b is:');
52     disp(b);
53 end
54 %% call backward substitution
55 x      = backsub(A,b);
56 end
57
58
59 function y      = tadd(x,y) %
60 x              = tround(x);
61 y              = tround(y);
62 y              = tround(x+y);
63 end
64
65 function y      = tmulti(x,y)
66 x              = tround(x);
67 y              = tround(y);
68 y              = tround(x*y);
69 end
70 function y      = tdiv(x,y)
71 x              = tround(x);
72 y              = tround(y);
73 y              = tround(x/y);
74 end
75
76 function y      = tsub(x,y)
77 x              = tround(x);
78 y              = tround(y);
79 y              = tround(x-y);
80 end
81
82 function y      = tround(x)
83 global d t
84 switch t
85     case 'round'
86         y      = round(x,d,'significant');
87     otherwise
88         ex     = ceil(log10(abs(x)));
89         x      = x/(10^ex);
90         y      = fix(x*(10^d))/(10^d);
91         x      = x*(10^ex);
92 end
93
94 end
95
96 function x      = backsub(A,b)
97 %% A is a upper triangular matrix

```

```

98 n           = length(A);
99 x           = zeros(n,1);
100 for i       = n:-1:1
101     x(i)     = tround(b(i)/A(i,i));
102     fprintf('x(%d) is %g ',i,x(i));
103     for j    = 1:i-1
104         b(j)  = tround(b(j) - tmulti(A(j,i),x(i)));
105     end
106     fprintf('new b is %g\n',b);
107 end
108 end
109 end

```

2.3 Gauss elimination with scaled partial pivoting

```

1  function x      = GS_scaled(A,b,digit,type)
2  %% function to perform the Gauss elimination with scaled partial
3  %% pivoting and rounding or chopping arithmetics
4  % input: Ax     = b
5  % digit, how many digits to keep when using rounding or chopping
6  % type, be 'round' or 'chop', which is a string
7  if ~exist('digit','var')
8      digit      = 4;
9  end
10 if ~exist('type','var')
11     type        = 'round'; % using rounding arithmetic
12 end
13 n              = length(A); % dimension of A
14 global d t
15 d              = digit;
16 t              = type;
17 %% perform GS elimination
18 for i          = 1:n-1
19     %% partial pivoting
20                                     % largest value of A(i,j);
21                                     % largest value of each row
22     v           = zeros(n-i+1,1);
23     for j       = i:n
24         v(j-i+1) = max(abs(A(j,i:n)));
25     end
26     for j       = i:n
27         v(j-i+1) = abs(A(j,i))/v(j-i+1);
28     end
29     fprintf('The relative pivoting is:\n');
30     disp(v);
31     [~,idx]     = max(v);
32     idx         = idx + i-1;
33
34     fprintf('  change %d-row and %d-th row\n',i,idx);
35     v           = A(i,:);
36     A(i,:)      = A(idx,:);
37     A(idx,:)    = v;
38     fprintf('New A after row exchange\n');
39     disp(A);
40                                     % exchange b also
41     v           = b(i);
42     b(i)        = b(idx);
43     b(idx)      = v;
44     fprintf('New b after row exchange\n');
45     disp(b);
46     for j       = i+1:n
47         mji     = tdiv(A(j,i),A(i,i));
48         fprintf('m(%d%d) is %g\n',j,i,mji);
49                                     % -mji row i + row j

```

```

50
51     for k      = i+1:n
52         A(j,k) = tsub(A(j,k), tmulti(mji,A(i,k)));
53
54     end
55     b(j)      = tsub(b(j),tmulti(mji,b(i)));
56 end
57 A(i+1:n,i)   =0;
58 fprintf('The new A is:\n');
59 disp(A);
60 fprintf('The new b is:\n');
61 disp(b);
62 end
63 %% call backward substitution
64 x      = backsub(A,b);
65 end
66
67
68 function y      = tadd(x,y) %
69 x              = tround(x);
70 y              = tround(y);
71 y              = tround(x+y);
72 end
73
74 function y      = tmulti(x,y)
75 x              = tround(x);
76 y              = tround(y);
77 y              = tround(x*y);
78 end
79 function y      = tdiv(x,y)
80 x              = tround(x);
81 y              = tround(y);
82 y              = tround(x/y);
83 end
84
85 function y      = tsub(x,y)
86 x              = tround(x);
87 y              = tround(y);
88 y              = tround(x-y);
89 end
90
91 function y      = tround(x)
92 global d t
93 switch t
94     case 'round'
95         y      = round(x,d,'significant');
96     otherwise
97         ex     = ceil(log10(abs(x)));
98         x      = x/(10^ex);
99         y      = fix(x*(10^d))/(10^d);
100        x      = x*(10^ex);
101 end
102
103 end
104
105 function x      = backsub(A,b)
106 %% A is a upper triangular matrix
107 n      = length(A);
108 x      = zeros(n,1);
109 for i   =n:-1:1
110     x(i) = tround(b(i)/A(i,i));
111     fprintf('x(%d) is %g ',i,x(i));
112     for j = 1:i-1
113         b(j) = tround(b(j) - tmulti(A(j,i),x(i)));
114     end
115     fprintf('new b is %g\n',b);
116
117 end
118 end

```

2.4 Usage of there functions

First, given value of digits, and choosing rounding or chopping. Then input your matrix A, and vector b.

```
1  digit = 4;  
2  type = 'round';  
3  A = [3.03,-12.1,14;  
4      -3.03,12.1,-7;  
5      6.11 -14.2 21];  
6  b = [-119;120;-139];  
7  x = GS_partial(A,b,digit,type)
```