Scilab Textbook Companion for
Linear Algebra And Its Applications
by G. Strang[1]

Created by
Sri Harsha Chillara
B.Tech (pursuing)
Electrical Engineering
NIT, Suratkal
College Teacher
NA
Cross-Checked by
Sonanya Tatikola, IIT Bombay

August 10, 2013

Book Description

Title: Linear Algebra And Its Applications

Author: G. Strang

Publisher: Cengage Learning

Edition: 4

Year: 2011

ISBN: 81-3150-172-8

Scilab numbering policy used in this document and the relation to the above book.

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

AP Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

Contents

List of Scilab Codes

Chapter 1

Matrix Notation and Matrix Multiplication

Scilab code Exa 1.3.1 Breakdown of elimination

```
1  clear;
2  close;
3  clc;
4  a =[1 1 1;2 2 5;4 6 8]
5  disp('x=[u;v;w]')
6  disp('R2=R2−R1,R3=R3−4∗R1')
7  a(2,:)=a(2,:)-2*a(1,:);
8  a(3,:)=a(3,:)-4*a(1,:);
9  disp(a);
10 disp('R2<−>R3')
11 b=a(2,:);
12 a(2,:)=a(3,:);
13 a(3,:)=b;
14 disp(a);
15 disp('The system is now triangular and the equations
        can be solved by Back substitution');
16 //end
```

7

Scilab code Exa 1.3.2 Breakdown of elimination

```
1  clear ;
2  close ;
3  clc ;
4  a =[1 1 1;2 2 5;4 4 8];
5  disp(a,'a=');
6  disp('x=[u;v;w]');
7  disp('R2=R2-2*R1,R3=R3-4*R1');
8  a(2,:)=a(2,:)-2*a(1,:);
9  a(3,:)=a(3,:)-4*a(1,:);
10 disp(a);
11 disp('No exchange of equations can avoid zero in the
        second pivot positon ,therefore the equations
      are unsolvable');
12 //end
```

Scilab code Exa 1.4.1 Multiplication of Two Matrices

```
1  clear ;
2  close ;
3  clc ;
4  A=[2 3;4 0];
5  disp(A,'A=');
6  B=[1 2 0;5 -1 0];
7  disp(B,'B');
8  disp(A*B,'AB=')
9  //end
```

Scilab code Exa 1.4.2 Multiplication with Row exchange matrix

```
1  clear ;
2  close ;
3  clc ;
4  A =[2 3;7 8];
5  disp (A , 'A=') ;
6  P =[0 1;1 0];
7  disp (P , 'P( Row exchange matrix )=')
8  disp (P*A , 'PA=')
9  // end
```

Scilab code Exa 1.4.3 Multiplication with Identity Matrix

```
1   // page 24
2   clear ;
3   close ;
4   clc ;
5   A =[1 2;3 4];
6   disp (A , 'A=') ;
7   I = eye (2 ,2) ;
8   disp (I , 'I=') ;
9   disp (I*A , 'IA=')
10  // end
```

Scilab code Exa 1.4.4 Marix multiplication not commutative

```
1  // page 25
2  clear ;
3  close ;
4  clc ;
5  E = eye (3 ,3) ;
6  E (2 ,:) =E (2 ,:) -2*E (1 ,:) ;
```

```
7  disp(E,'E=');
8  F=eye(3,3);
9  F(3,:)=F(3,:)+F(1,:);
10 disp(F,'F=');
11 disp(E*F,'EF=')
12 disp(F*E,'FE=')
13 disp('Here,EF=FE,so this shows that these two
        matrices commute')
14 //end
```

Scilab code Exa 1.4.5 Order of Elimination

```
1  //page 25
2  clear;
3  close;
4  clc;
5  E=eye(3,3);
6  E(2,:)=E(2,:)-2*E(1,:);
7  disp(E,'E')
8  F=eye(3,3);
9  F(3,:)=F(3,:)+F(1,:);
10 disp(F,'F=');
11 G=eye(3,3);
12 G(3,:)=G(3,:)+G(2,:);
13 disp(G,'G')
14 disp(G*E,'GE=')
15 disp(E*G,'EG=')
16 disp('Here EG is not equal to GE,Therefore these two
        matrices do not commute and shows that most
        matrices do not commute.')
17 disp(G*F*E,'GFE=')
18 disp(E*F*G,'EFG=')
19 disp('The product GFE is the true order of elimation
        .It is the matrix that takes the original A to
        the upper triangular U.')
```

```
20  //end
```

Scilab code Exa 1.5.1 Triangular factorization

```
 1  //page 34
 2  clear;
 3  close;
 4  clc;
 5  A=[1 2;3 8];
 6  disp(A,'A=')
 7  [L,U]=lu(A);
 8  disp(L,'L=');
 9  disp(U,'U=')
10  disp('LU=')
11  disp(L*U)
12  disp('This shows that LU=A')
13  //end
```

Scilab code Exa 1.5.2 To check LU equals to A

```
 1  //page 34
 2  clear;
 3  close;
 4  clc;
 5  A=[0 2;3 4];
 6  disp(A,'A=')
 7  disp('Here this cannot be factored into A=LU,(Needs
        a row exchange)');
 8  //end
```

Scilab code Exa 1.5.3 To check LU equals to A

```
1  // page 34
2  clear ;
3  close ;
4  clc ;
5  disp ( 'Given Matrix : ')
6  A =[1 1 1;1 2 2;1 2 3];
7  disp (A , 'A=') ;
8  [L , U ]= lu (A) ;
9  disp (L , 'L=') ;
10 disp (U , 'U=') ;
11 disp (L * U , 'LU=') ;
12 disp ( 'Here LU=A, from A to U there are subtraction of
       rows . Frow U to A there are additions of rows ') ;
13 // end
```

Scilab code Exa 1.5.4 If U equals to I then L equals to A

```
1  // page 34
2  clear ;
3  close ;
4  clc ;
5  a = rand (1) ;
6  b = rand (1) ;
7  c = rand (1) ;
8  L =[1 0 0;a 1 0;b c 1];
9  disp (L , 'L=') ;
10 U = eye (3 ,3) ;
11 disp (U , 'U=') ;
12 E =[1 0 0; -a 1 0;0 0 1];
13 disp (E , 'E=') ;
14 F =[1 0 0;0 1 0; -b 0 1];
15 disp (F , 'F=') ;
16 G =[1 0 0;0 1 0;0 -c 1];
```

```
17  disp (G, 'G=') ;
18  disp ('A=inv (E) * inv (F) * inv (G) * U')
19  A = inv (E) * inv (F) * inv (G) * U ;
20  disp (A, 'A=') ;
21  disp ('When U is  identity matrix then L is same as A'
        ) ;
22  //end
```

Scilab code Exa 1.5.5 Spilting A to L and U

```
1  //page 39
2  clear ;
3  close ;
4  clc ;
5  A =[1 -1 0 0 ;-1 2 -1 0;0 -1 2 -1;0 0 -1 2];
6  disp (A, 'A=') ;
7  [L ,U]= lu (A) ;
8  disp (U, 'U=') ;
9  disp (L, 'L=') ;
10 disp ('This shows how a matrix A with 3 diagnols has
       factors L and U with two diagnols . ')
11 //end
```

Scilab code Exa 1.5.6 Solving for X using L and U

```
1  //page 36
2  clear ;
3  close ;
4  clc ;
5  a =[1 -1 0 0;-1 2 -1 0;0 -1 2 -1;0 0 -1 2];
6  disp (a, 'a=')
7  b =[1;1;1;1]
8  disp (b, 'b=')
```

```
 9  disp('Given Equation ,ax=b')
10  [L,U]=lu(a);
11  disp(U,'U=');
12  disp(L,'L=');
13  disp('Augmented Matrix of L and b=');
14  A=[L b];
15  disp(A)
16  c=zeros(4,1);
17  n=4;
18  //Algorithm Finding the value of c
19  c(1)=A(1,n+1)/A(1,1);
20  for i=2:n;
21      sumk=0;
22      for k=1:n-1
23          sumk=sumk+A(i,k)*c(k);
24      end
25      c(i)=(A(i,n+1)-sumk)/A(i,i)
26  end
27  disp(c,'c=')
28  x=zeros(4,1);
29  disp('Augmented matrix of U and c=')
30  B=[U c];
31  disp(B)
32  //Algorithm for finding value of x
33  x(n)=B(n,n+1)/B(n,n);
34  for i=n-1:-1:1;
35      sumk=0;
36      for k=i+1:n
37          sumk=sumk+B(i,k)*x(k);
38      end
39      x(i)=(B(i,n+1)-sumk)/B(i,i);
40  end
41  disp(x,'x=')
42  //end
```

14

Scilab code Exa 1.5.7 Elimination in a nutshell

```
1  // page 39
2  clear ;
3  close ;
4  clc ;
5  A =[1 1 1;1 1 3;2 5 8];
6  disp (A, 'A=');
7  [L,U,P]= lu (A);
8  disp (L, 'L=');
9  disp (U, 'U=');
10  disp (P, 'P=');
11  disp (P*A, 'PA=')
12  disp (L*U, 'LU=')
13  disp ('This shows that PA is the same as LU')
14  // end
```

Scilab code Exa 1.6.1 Gauss Jordon method

```
1  // page 47
2  clear ;
3  close ;
4  clc ;
5  disp ('Given matrix : ')
6  A =[2 1 1;4 -6 0; -2 7 2];
7  disp (A);
8  [n,m]= size (A);
9  disp ('Augmented matrix : ')
10  a =[A eye (n,m)];
11  disp (a)
12  disp ('R2=R2-2*R1,R3=R3-(-2)*R1');
13  a (2,:)= a (2,:) -2*a (1,:);
14  a (3,:)= a (3,:) -(-1)*a (1,:);
15  disp (a)
16  disp ('R3=R3-(-1)*R2');
```

```
17  a(3,:)=a(3,:)-(-1)*a(2,:);
18  disp(a,'a=')
19  disp(a,'[U inv(L)] :')
20  disp('R2=R2-(-2)*R3,R1=R1-R3')
21  a(2,:)=a(2,:)-(-2)*a(3,:);
22  a(1,:)=a(1,:)-a(3,:);
23  disp(a)
24  disp('R1=R1-(-1/8)*R2)')
25  a(1,:)=a(1,:)-(-1/8)*a(2,:);
26  disp(a)
27  a(1,:)=a(1,:)/a(1,1);
28  a(2,:)=a(2,:)/a(2,2);
29  disp('[I inv(A)]:')
30  a(3,:)=a(3,:)/a(3,3);
31  disp(a);
32  disp('inv(A):')
33  a(:,4:6);
34  disp(a(:,4:6))
```

Scilab code Exa 1.6.2 Symmetric products

```
1  //Caption :Symmetric Products
2  //Example:1.6.2-To Find the product of transpose(R)
       and R.
3  //page 51
4  clear;
5  close;
6  clc;
7  R=[1 2];
8  disp(R,'R=');
9  Rt=R';
10 disp(Rt,'Transpose of the given matrix is :')
11 disp(R*Rt,'The product of R & transpose(R)is :')
12 disp(Rt*R,'The product of transpose(R)& R is :' )
13 disp('Rt*R and R*Rt are not likely to be equal even
```

16

```
     if m==n.')
14  //end
```

Chapter 2

Vector Spaces

Scilab code Exa 2.1.1 Vector Spaces and subspaces

```scilab
1  //page 70
2  clear;
3  close;
4  clc;
5  disp('Consider all vectors in R^2 whose components
       are positive or zero')
6  disp('The subset is first Quadrant of x-y plane,the
       co-ordinates satisfy x>=0 and y>=0.It is not a
       subspace.')
7  v=[1,1];
8  disp(v,'If the Vector=');
9  disp('Taking a scalar,c=-1')
10 c=-1;  //scalar
11 disp(c*v,'c*v=')
12 disp('It lies in third Quadrant instead of first,
       Hence violating the rule(ii).')
13 //end
```

Scilab code Exa 2.1.2 Vector Spaces and subspaces

```
1  //page 71
2  clear;
3  close;
4  clc;
5  disp('Take vector space of 3X3 matrices')
6  disp('One possible subspace is the set of lower
       triangular matrices,Another is set of symmetric
       matrices')
7  disp('A+B,cA are both lower triangular if A and B
       are lower triangular ,and are symmetric if A and
       B are symmetric and Zero matrix is in both
       subspaces')
```

Scilab code Exa 2.3.1 Linear Independence

```
1  //page 92
2  clear;
3  close;
4  clc;
5  disp('For linear independence ,C1V1+C2V2+......CkVk=0
       ')
6  disp('If we choose V1=zero vector ,then the set is
       linearly dependent.We may choose C1=3 and all
       other Ci=0;this is a non−trival solution that
       produces zero.')
7  //end
```

Scilab code Exa 2.3.2 Linear Independence

```
1  //page 92
2  clear;
3  close;
4  clc;
```

```
 5  A =[1 3 3 2;2 6 9 5;-1 -3 3 0];
 6  disp('Given matrix:')
 7  disp(A)
 8  B=A;
 9  disp('C2->C2-3*C1')
10  A(:,2)=A(:,2)-3*A(:,1);
11  disp(A)
12  disp('Here,C2=3*C1,Therefore the columns are
        linearly dependent.')
13  disp('R3->R3-2*R2+5*R1')
14  B(3,:)=B(3,:)-2*B(2,:)+5*B(1,:);
15  disp(B)
16  disp('Here R3=R3-2*R2+5*R1,therefore the rows are
        linearly dependent.')
17  //end
```

Scilab code Exa 2.3.3 Linear Independence

```
1  clear;
2  close;
3  clc;
4  A =[3 4 2;0 1 5;0 0 2];
5  disp(A,'A=');
6  disp('The columns of the triangular matrix are
        linearly independent,it has no zeros on the
        diagonal');
7  //end
```

Scilab code Exa 2.3.4 Linear Independence

```
1  //page 93
2  clear;
3  close;
```

```
4  clc;
5  disp('The columns of the nxn identity matrix are
       independent.')
6  n=input('Enter n:');
7  I=eye(n,n);
8  disp(I,'I=');
9  //end
```

Scilab code Exa 2.3.5 Linear Independence

```
1   //page 93
2   clear;
3   close;
4   clc;
5   disp('Three columns in R2 cannot be independent.')
6   A=[1 2 1;1 2 3];
7   disp(A,'Given matrix:')
8   [L,U]=lu(A);
9   disp(U,'U=');
10  disp('If c3 is 1 ,then back-substitution Uc=0 gives
        c2=-1,c1=1,With these three weights,the first
        column minus the second plus the third equals
        zero ,therefore linearly dependent.')
```

Scilab code Exa 2.3.6 Linear Independence

```
1  //page 93
2  clear;
3  close;
4  clc;
5  disp('The vectors w1=(1,0,0),w2=(0,1,0),w3=(-2,0,0)
        span a plane (x-y plane) in R3. The first two
```

```
      vectors  also  span  this  plane ,  whereas  w1  and  w3
         span  only  a  line . ') ;
6  // end
```

Scilab code Exa 2.3.7 Linear Independence

```
1  // page  93
2  clear ;
3  close ;
4  clc ;
5  disp ( 'The  column  space  of  A  is  excatly  the  space
         that  is  spanned  by  its  columns . The  row  space  is
         spanned  by  the  rows . The  definition  is  made  to
         order . Multiplying  A  by  any  x  gives  a  combination
         of  columns ;  it  is  a  vector  Ax  in  the  column  space
         .  The  coordinate  vectors  e_1 , .... e_n  coming  from
         the  identity  matrix  span  Rn .  Every  vector  b=( b_1
         ..... , b_n )  is  a  combination  of  those  columns . In
         this  example  the  weights  are  the  components  b_i
         themselves : b=b_1e_1 + .... +  b_ne_n . But  the  columns
         of  other  matrices  also  span  R_ . ')
6  // end
```

Scilab code Exa 2.3.8 Basis for a vector space

```
1  // page  93
2  clear ;
3  close ;
4  clc ;
5  disp ( 'Here , the  vector  v1  by  itself  is  linearly
         independent  ,  but  it  fails  to  span  R2 . The  three
         vectors  v1 , v2 , v3  certainly  span  R2 ,  but  are  not
         independent .  Any  two  of  these  vectors  say  v1  and
```

v2 have both properties −they span and they are
independent.So they form a basis.(A vector space
does not have a unique basis)')

```scilab
6  //end
```

Scilab code Exa 2.3.9 Basis for a vector space

```scilab
1  //page 96
2  clear;
3  close;
4  clc;
5  disp('These four columns span the column space U,but
        they are not independent.')
6  U=[1 3 3 2;0 0 3 1;0 0 0 0];
7  disp(U,'U=');
8  disp('The columns that contains pivots (here 1st & 3
      rd) are a basis for the column space. These
      columns are independent, and it is easy to see
      that they span the space.In fact,the column space
       of U is just the x−y plane withinn R3. C(U) is
      not the same as the column space C(A) before
      elimination−but the number of independent columns
       did not change.')
```

Scilab code Exa 2.4.1 The four fundamental subspaces

```scilab
1  //page 107
2  clear;
3  close;
4  clc;
5  A=[1 2;3 6];
6  disp(A,'A=');
7  [m,n]=size(A);
```

```scilab
 8  disp(m,'m=');
 9  disp(n,'n=');
10  [v,pivot]=rref(A);
11  r=length(pivot);
12  disp(r,'rank=')
13  cs=A(:,pivot);
14  disp(cs,'Column space=');
15  ns=kernel(A);
16  disp(ns,'Null space=');
17  rs=v(1:r,:)';
18  disp(rs,'Row space=')
19  lns=kernel(A');
20  disp(lns,'Left null sapce=');
```

Scilab code Exa 2.4.2 Inverse of a mxn matrix

```scilab
 1  //page 108
 2  clear;
 3  close;
 4  clc;
 5  A=[4 0 0;0 5 0];
 6  disp(A,'A=');
 7  [m,n]=size(A);
 8  disp(m,'m=');
 9  disp(n,'n=')
10  r=rank(A);
11  disp(r,'rank=');
12  disp('since m=r=2 ,there exists a right inverse .');
13  C=A'*inv(A*A');
14  disp(C,'Best right inverse=')
15  //end
```

Scilab code Exa 2.5.1 Networks and discrete applied mathematics

```scilab
1  //page 121
2  clear;
3  close;
4  clc;
5  disp('Applying current law A''y=f at nodes 1,2,3:')
6  A=[-1 1 0;0 -1 1; -1 0 1;0 0 -1;-1 0 0];
7  disp(A','A''=');
8  C=diag(rand(5,1)); //Taking some values for the
       resistances.
9  b=zeros(5,1);
10 b(3,1)=rand(1);//Taking some value of the battery.
11 f=zeros(3,1);
12 f(2,1)=rand(1);//Taking some value of the current
       source.
13 B=[b;f];
14 disp('The other equation is inv(C)y+Ax=b.The block
       form of the two equations is:')
15 C=[inv(C) A;A' zeros(3,3)];
16 disp(C);
17 X=['y1';'y2';'y3';'y4';'y5';'x1';'x2';'x3'];
18 disp(X,'X=')
19 X=C\B;
20 disp(X,'X=');
21 //end
```

Chapter 3

Orthogonality

Scilab code Exa 3.1.1 Orthogonal vectors

```
1  // page 143
2  clear;
3  close;
4  clc;
5  x1 =[2;2; -1];
6  disp (x1, 'x1=');
7  x2 =[ -1;2;2];
8  disp (x2, 'x2=');
9  disp (x1'*x2, 'x1''*x2=');
10 disp ('Therefore , X1 is  orthogonal  to  x2 . Both  have
        length  of  sqrt (14) .')
```

Scilab code Exa 3.1.2 Orthogonal vectors

```
1  // page 144
2  clear;
3  close;
4  clc;
```

```
5  disp('Suppose V is a plane spanned by v1=(1,0,0,0)
       and v2=(1,1,0,0).If W is the line spanned by w
       =(0,0,4,5),then w is orthogonal to both v''s.The
       line W will be orthogonal to the whole plane V.')
```

Scilab code Exa 3.1.3 Orthogonal vectors

```
1  //page 145
2  clear;
3  close;
4  clc;
5  A=[1 3;2 6;3 9];
6  disp(A,'A=');
7  ns=kernel(A);
8  disp(ns,'Null space=');
9  disp(A(1,:)*ns,'A(1,:)*ns=');
10 disp(A(2,:)*ns,'A(2,:)*ns=');
11 disp(A(3,:)*ns,'A(3,:)*ns=');
12 disp('This shows that the null space of A is
       orthogonal to the row space.');
13 //end
```

Scilab code Exa 3.2.1 Projections onto a line

```
1  //page 155
2  clear;
3  close;
4  clc;
5  b=[1;2;3];
6  disp(b,'b=');
7  a=[1;1;1];
8  disp(a,'a=')
9  x=(a'*b)/(a'*a)
```

```
10  disp(x*a,'Projection p of b onto the line through a
        is x^*a=');
11  disp((a'*b)/(sqrt(a'*a)*sqrt(b'*b)),'cos(thetha)=');
12  //end
```

Scilab code Exa 3.2.2 Projections onto a line

```
1  //page 156
2  clear;
3  close;
4  clc;
5  a=[1;1;1];
6  disp(a,'a=');
7  P=(a*a')/(a'*a);
8  disp(P,'Matrix that projects onto a line through a
        =(1,1,1) is ');
9  //end
```

Scilab code Exa 3.2.3 Projections onto a line

```
1  //page 156
2  clear;
3  close;
4  clc;
5  thetha=45; //Taking some value for thetha
6  a=[cos(thetha);sin(thetha)];
7  disp(a,'a=');
8  P=(a*a')/(a'*a);
9  disp(P,'Projection of line onto the thetha−direction
        (thetha taken as 45) in the x−y plane passing
        through a is ');
10  //end
```

Scilab code Exa 3.3.1 Projection matrices

```
1  // page 165
2  clear;
3  close;
4  clc;
5  A=rand(4,4);
6  disp(A,'A=');
7  P=A*inv(A'*A)*A';
8  disp('P=A*inv(A''*A)*A');
9  disp(P,'Projection of a  invertible 4x4 matrix on to
        the whole space is:');
10 disp('Its identity matrix.')
11 // end
```

Scilab code Exa 3.3.2 Least squares fitting of data

```
1  // page 166
2  clear;
3  close;
4  clc;
5  disp('b=C+Dt');
6  disp('Ax=b');
7  A=[1 -1;1 1;1 2];
8  disp(A,'A=');
9  b=[1;1;3];
10 disp(b,'b=');
11 disp('If Ax=b could be solved then they would be no
        errors, they can''t be solved because the points
        are not on a line.Therefore they are solved by
        least squares.')
12 disp('so,A''Ax^=A''b');
```

```
13  x=zeros(1,2);
14  x=(A'*A)\(A'*b);
15  disp(x(1,1),'C^ =');
16  disp(x(2,1),'D^=');
17  disp('The best line is 9/7+4/7t')
18  //end
```

Scilab code Exa 3.4.1 Orthogonal matrices

```
1  //page 175
2  clear;
3  close;
4  clc;
5  thetha=45;//Taking some value for thetha.
6  Q=[cos(thetha) -sin(thetha);sin(thetha) cos(thetha)
      ];
7  disp(Q,'Q=');
8  disp(Q','Q''=inv(Q)=');
9  disp('Q rotates every vector through an angle thetha
      , and Q'' rotates it back through -thetha.The
      columns are clearly orthogonal and they are
      orthonormal because sin^2(theta)+cos^2(thetha)=1.
      ');
10  //end
```

Scilab code Exa 3.4.2 Orthogonal matrices

```
1  //page 175
2  clear;
3  close;
4  clc;
5  disp('Any permutation matrix is an orthogonal matrix
      .The columns are certainly unit vectors and
```

```
          certainly  orthogonal−because  the  1  appears  in  a
          differnt  place  in  each  column ')
 6  P =[0  1  0;0  0  1;1  0  0];
 7  disp (P, 'P=');
 8  disp (P', 'inv (P)=P''=');
 9  disp (P'*P, 'And,P''*P=');
10  //end
```

Scilab code Exa 3.4.3 Projection onto a plane

```
 1  //page  175
 2  clear ;
 3  close ;
 4  clc ;
 5  disp ('If  we  project  b=(x,y,z)  onto  the  x−y  plane
          then  its  projection  is  p=(x,y,0) ,and  is  the  sum
          of  projection  onto  x−  any  y−axes . ')
 6  b=rand (3 ,1) ;
 7  q1 =[1;0;0];
 8  disp (q1 , 'q1=');
 9  q2 =[0;1;0];
10  disp (q2 , 'q2=');
11  P=q1*q1'+q2*q2';
12  disp (P, 'Overall  projection  matrix ,P=');
13  disp ('and ,P[x;y;z]=[x;y;0] ')
14  disp ('Projection  onto  a  plane=sum  of  projections
          onto  orthonormal  q1  and  q2 . ')
15  //end
```

Scilab code Exa 3.4.4 Least squares fitting of data

```
 1  //page  166
 2  clear ;
```

```
3  close ;
4  clc ;
5  disp ( 'y=C+Dt ') ;
6  disp ( 'Ax=b ') ;
7  A =[1  -3;1  0;1  3];
8  disp (A , 'A=') ;
9  y = rand      (3 ,1) ;
10  disp (y , 'y=') ;
11  disp ( 'the  columns  of  A  are  orthogonal , so ')
12  x = zeros (1 ,2) ;
13  disp (([1  1  1]*y) /(A (: ,1) '*A (: ,1)) , 'C^ =') ;
14  disp (([-3  0  3]*y) /(A (: ,2) '*A (: ,2)) , 'D^ =')
15  disp ( 'C^  gives  the  besy  fit  ny  horizontal  line ,
         whereas  D^t  is  the  best  fit  by  a  straight  line
         through  the  origin . ')
16  // end
```
_____

Scilab code Exa 3.4.5 Gram Schmidt process

```
1  // page  166
2  clear ;
3  close ;
4  clc ;
5  A =[1  0  1;1  0  0;2  1  0]; // independent  vectors  stored
        in  columns  of  A
6  disp (A , 'A=') ;
7  [m , n ]= size (A) ;
8  for  k =1: n
9      V (: , k)=A (: , k) ;
10      for  j =1: k-1
11          R (j , k)=V (: , j) '*A (: , k) ;
12          V (: , k)=V (: , k)-R (j , k)*V (: , j) ;
13      end
14      R (k , k)= norm (V (: , k)) ;
15      V (: , k)=V (: , k) /R (k , k) ;
```

```
16  end
17  disp(V,'Q=')
```

Chapter 4

Determinants

Scilab code Exa 4.3.1 Determinant of a matrix is the product of its pivots

```
1  clear ;
2  close ;
3  clc ;
4  n=input ('Enter the value of n:');
5  for i=1
6       for j=i;
7            a(i,j)=2;
8            a(i,j+1)=-1;
9       end
10 end
11 for i=2:n-1
12      for j=i
13           a(i,j-1)=-1;
14           a(i,j)=2;
15           a(i,j+1)=-1;
16      end
17 end
18 for i=n
19      for j=i
20           a(i,j-1)=-1;
21           a(i,j)=2;
```

```
22        end
23  end
24  disp(a,'a=');
25  [L,D,U]=lu(a)
26  determinant =1;
27  for i=1:n
28          determinant=determinant*D(i,i);
29  end
30  disp(determinant,'Determinant=')
31  //end
```

Scilab code Exa 4.3.2 Calculation of determinant of a matrix by using cofactors

```
1  clear;
2  close;
3  clc;
4  disp('For a 3*3 matrix:')
5  disp('det A=a11(a22a33−a23a32)+a12(a23a31−a21a33)+
       a13(a21a32−a22a31)')
6  //end
```

Scilab code Exa 4.3.3 Calculation of determinant of a matrix by using cofactors

```
1  //page 214
2  clear;
3  close;
4  clc;
5  A=[2 -1 0 0;-1 2 -1 0;0 -1 2 -1;0 0 -1 2];
6  disp(A,'A=');
7  [m,n]=size(A)
8  a=A(1,:);
```

```
 9   c=[];
10   for l=1:4
11       B=A([1:0,2:4],[1:l-1,l+1:4]);
12       c1l=(-1)^(1+l)*det(B);
13       c=[c;c1l];
14   end
15   d=a*c;
16   disp(d)
```

Scilab code Exa 4.4.1 Inverse of a sum matrix is a difference matrix

```
 1   //282
 2   clear;
 3   close;
 4   clc;
 5   A=[1 1 1;0 1 1;0 0 1];
 6   disp(A,'A=')
 7   n=size(A,1); d=1:n-1;
 8   B=zeros(n); AA=[A,A;A,A]';
 9   for j=1:n
10      for k=1:n
11          B(j,k)=det(AA(j+d,k+d));
12      end
13   end
14   disp(B,'Adjoint  of  A:');
15   disp(B/det(A),'inv(A):');
16   //end
```

Scilab code Exa 4.4.2 Cramers rule

```
 1   //page 222
 2   clear;
 3   close;
```

```
 4  clc;
 5  //x1+3x2=0
 6  //2x1+4x2=6
 7  A=[1 3;2 4];
 8  b=[0;6];
 9  X1=[0 3;6 4];
10  X2=[1 0;2 6];
11  disp(det(X1)/det(A),'x1=');
12  disp(det(X2)/det(A),'x2=');
13  //end
```

Chapter 5

Eigenvalues and Eigenvectors

Scilab code Exa 5.1.1 Eigenvalues and eigenvectors

```
1  //page 238
2  clear;
3  close;
4  clc;
5  A=[3 0;0 2];
6  eig=spec(A);
7  [V,Val]=spec(A);
8  disp(eig,'Eigen values:')
9  x1=V(:,1);
10 x2=V(:,2);
11 disp(x1,x2,'Eigen vectors:');
12 //end
```

Scilab code Exa 5.1.2 Eigenvalues and eigenvectors

```
1  //page 238
2  clear;
3  close;
```

```
 4 clc;
 5 disp('The eigen values of a projection matrix are 1
      or 0.')
 6 P=[1/2 1/2;1/2 1/2];
 7 eig=spec(P);
 8 [V,Val]=spec(P);
 9 disp(eig,'Eigen values:')
10 x1=V(:,1);
11 x2=V(:,2);
12 disp(x1,x2,'Eigen vectors:');
13 //end
```

Scilab code Exa 5.2.1 Diagonalization

```
 1 //page 238
 2 clear;
 3 close;
 4 clc;
 5 A=[1/2 1/2;1/2 1/2];
 6 [V,Val]=spec(A);
 7 disp(Val,'Eigenvalue matrix:');
 8 disp(V,'S=');
 9 disp(A*V,'AS=S*eigenvaluematrix')
10 disp('Therefore inv(S)*A*S=eigenvalue matrix')
11 //end
```

Scilab code Exa 5.2.2 Diagonalization

```
 1 //page 238
 2 clear;
 3 close;
 4 clc;
```

```
5 disp('The eigenvalues themselves are not so clear
      for a rotation.')
6 disp('90 degree rotation')
7 K=[0 -1;1 0];
8 disp(K,'K=')
9 eig=spec(K);
10 [V,Val]=spec(K);
11 disp(eig,'Eigen values:')
12 x1=V(:,1);
13 x2=V(:,2);
14 disp(x1,x2,'Eigen vectors:');
15 //end
```

Scilab code Exa 5.2.3 Powers and Products

```
1 //page 249
2 clear;
3 close;
4 clc;
5 disp('K is rotation through 90 degree,then K^2 is
      rotation through 180 degree and inv(k is rotation
       through −90 degree)')
6 K=[0 -1;1 0];
7 disp(K,'K=')
8 disp(K*K,'K^2=')
9 disp(K*K*K,'K^3=')
10 disp(K^4,'K^4=')
11 [V,D]=spec(K);
12 disp('K^4 is a complete rotation through 360 degree.
      ')
13 disp(D,'Eigen value matrix,D of K:');
14 disp(D^4,'and also D^4=')
15 //end
```

Scilab code Exa 5.3.1 Difference equations

```
1  //page 249
2  clear;
3  close;
4  clc;
5  A=[0 4;0 1/2];
6  disp(A,'A=');
7  eig=spec(A);
8  disp(eig,'Eigen values:')
9  [v,D]=spec(A);
10 u0=[v(:,1)];//Taking u0 as the 1st eigen Vector.
11 for k=0:5
12     disp(k,'k=');
13     u=A*u0;
14     disp(u,'U(k+1)(K from 0 to 5)')
15     u0=u;
16 end
17 u0=[v(:,2)];//Taking u0 as the 2nd eigen vector.
18 for k=0:5
19     disp(k,'k=');
20     u=A*u0;
21     disp(u,'U(k+1)=')
22     u0=u;
23 end
```

Scilab code Exa 5.5.1 Complex matrices

```
1  //page282
2  clear;
3  close;
4  clc;
```

```
 5  i = sqrt (-1) ;
 6  x =3+4* i ;
 7  disp (x , 'x=') ;
 8  x_ = conj (x) ;
 9  disp ( x*x_ , 'xx_=') ;
10  r = sqrt ( x*x_ ) ;
11  disp (r , 'r=')
12  // end
```

Scilab code Exa 5.5.2 Inner product of a complex matrix

```
 1  // 282
 2  clear ;
 3  close ;
 4  clc ;
 5  i = sqrt (-1) ;
 6  x =[1  i ] ';
 7  y =[2+1* i  2 -4* i ] ';
 8  disp (x '*x , 'Length  of  x  squared : ') ;
 9  disp (y '*y , 'Length  of  y  squared : ') ;
10  // end
```

Chapter 6

Positive Definite Matrices

Scilab code Exa 6.1.1 Definite versus indefinite

```
1  //313
2  clear;
3  close;
4  clc;
5  disp('f(x,y)=x^2-10*x*y+y^2');
6  a=1;
7  c=1;
8  deff('[f]=f(x,y)','f=x^2-10*x*y+y^2');
9  disp(f(1,1),'f(1,1)=');
10 disp('The conditions a>0 and c>0 ensure that f(x,y)
       is positive on the x and y axes. But this
       function is negative on the line x=y,because b
       =-10 overwhelms a and c. ');
11 //end
```

Scilab code Exa 6.1.3 Maxima Minima And Saddle points

```
1  //315
```

```
 2  clear ;
 3  close ;
 4  clc ;
 5  disp ( ' f ( x , y )=2*x^2+4*x*y+y^2 ') ;
 6  A =[2  2;2  1];
 7  a =1;
 8  c =1;
 9  b =2;
10  disp ( a * c , 'ac= ') ;
11  disp ( b^2 , 'b^2= ') ;
12  disp ( 'Saddle  point , as  ac<b^2 ') ;
```

Scilab code Exa 6.1.4 Maxima Minima And Saddle points

```
 1  // 315
 2  clear ;
 3  close ;
 4  clc ;
 5  disp ( ' f ( x , y )=2*x^2+4*x*y+y^2 ') ;
 6  A =[2  2;2  1];
 7  a =0;
 8  c =0;
 9  b =1;
10  disp ( a * c , 'ac= ') ;
11  disp ( b^2 , 'b^2= ') ;
12  disp ( 'Saddle  point , as  ac<b^2 ') ;
```

Scilab code Exa 6.2.2 Maxima Minima And Saddle points

```
 1  // 313
 2  clear ;
 3  close ;
 4  clc ;
```

```
 5  disp ( ' f ( x , y )=xˆ2+4∗x∗y+yˆ2 ' ) ;
 6  a =1;
 7  c =1;
 8  deff ( ' [ f ]= f ( x , y ) ' , ' f=xˆ2+4∗x∗y+yˆ2 ' ) ;
 9  disp ( f ( 0 , 0 ) , ' f ( 0 , 0 )= ' )
10  disp ( ' Here  2b=4  it  still  does  not  ensure  a  minimum
        , the  sign  of  b  is  of  no  importance . Neither  F  nor
        f  has  a  minimum  at ( 0 , 0 )  because  f ( 1 ,−1)=−1. ' )
11  //end
```

Scilab code Exa 6.3.1 Singular value decomposition

```
 1  //332
 2  clear ;
 3  close ;
 4  clc ;
 5  A =[ -1  2  2] ' ;
 6  disp ( A , ' A= ' ) ;
 7  [ U  diagnol  V ]= svd ( A ) ;
 8  disp ( U , ' U= ' ) ;
 9  disp ( diagnol , ' diagnol= ' ) ;
10  disp ( V ' , ' V ' '= ' ) ;
11  disp ( U∗ diagnol ∗V ' , ' A=U∗ diagnol ∗V ' ' ' )
12  //end
```

Scilab code Exa 6.3.2 Singular value decomposition

```
 1  //332
 2  clear ;
 3  close ;
 4  clc ;
 5  A =[ -1  1  0;0  -1  1] ;
 6  disp ( A , ' A= ' ) ;
```

```
7  [U diagnl V]=svd(A);
8  disp(U,'U=');
9  disp(diagnl,'Diagonal=');
10 disp(V','V''=');
11 disp(U*diagnl*V','A=U*diagonal*V''=')
12 //end
```

Scilab code Exa 6.3.3 Polar decomposition

```
1  //332
2  clear;
3  close;
4  clc;
5  A=[1 -2;3 -1];
6  disp(A,'A=');
7  [U S V]=svd(A);
8  Q=U*V';
9  S=V*S*V';
10 disp(Q,'Q=');
11 disp(S,'S=');
12 disp(Q*S,'A=SQ=')
13 //end
```

Scilab code Exa 6.3.4 Reverse polar decomposition

```
1  //332
2  clear;
3  close;
4  clc;
5  A=[1 -2;3 -1];
6  disp(A,'A=');
7  [U diag1 V]=svd(A);
8  Q=U*V';
```

```
 9  S =[2  1;1  3];
10  disp(Q,'Q=');
11  disp(S,'S=')
12  disp(S'*Q,'A=S''Q=')
13  //end
```

Chapter 7

Computations with Matrices

Scilab code Exa 7.4.1 Jacobi Method

```
1  //page 238
2  clear ;
3  close ;
4  clc ;
5  A =[2 -1; -1  2];
6  S =[2  0;0  2];
7  T =[0  1;1  0];
8  p=inv (S)*T;
9  b=[2 2]';
10 x=zeros (2 ,1);
11 disp(x,'intial  v & w:')
12 x_1=zeros (1 ,2);
13 for k=0:25
14     x_1=p*x+inv(S)*b;
15     x=x_1;
16     disp(k,'k=')
17     disp(x_1,'v(k+1) & w(k+1)=');
18 end
```

Scilab code Exa 7.4.2 Gauss Seidel method

```
1  // page  238
2  clear ;
3  close ;
4  clc ;
5  A =[2  -1; -1  2];
6  S =[2  0; -1  2];
7  T =[0  1;0  0];
8  b= rand (2 ,1) ;
9  p= inv (S)*T;
10  x= zeros (2 ,1) ;
11  disp (x, 'intial  v & w: ')
12  x_1= zeros (1 ,2) ;
13  for  k =0:25
14      x_1 = p*x+ inv (S)*b;
15      x =x_1 ;
16      disp (k, 'k= ')
17      disp (x_1 , 'v( k+1)  &  w( k+1)= ');
18  end
```

Chapter 8

Linear Programming and Game Theory

Scilab code Exa 8.2.2 Minimize cx subject to x greater than or equal to zero and Ax equals to b

```
1  // page 238
2  clear ;
3  close ;
4  clc ;
5  A =[1 0 1 6 2;0 1 1 0 3];
6  b =[8 9] ';
7  c =[0 0 7 -1 -3] ';
8  lb =[0 0 0 0 0] '
9  ub =[];
10 [x , lagr , f]= linpro (c ,A ,b ,lb ,ub );
11 disp (x , 'New  corner : ');
12 disp (f , 'Minimum  cost : ');
13 // end
```