# Scilab Textbook Companion for Introductory Methods Of Numerical Analysis by S. S. Sastry[1]

Created by
Rohit Kumar Singh
B Tech
Others
NIT, JAMSHEDPUR
College Teacher
NA
Cross-Checked by
Lavitha Pereira

May 24, 2016

[1]Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website http://scilab.in

# Book Description

**Title:** Introductory Methods Of Numerical Analysis

**Author:** S. S. Sastry

**Publisher:** Phi Learning

**Edition:** 5

**Year:** 2012

**ISBN:** 9788120345928

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

# Errors in numerical calculation

**Scilab code Exa 1.1** rounding off

```
//example 1.1
//rounding off
//page 7
clc;clear;close;
a1=1.6583;
a2=30.0567;
a3=0.859378;
a4=3.14159;
printf('\nthe numbers after rounding to 4 significant figures are given below\n')
printf('    %f         %.4g\n',a1,a1);
printf('    %f         %.4g\n',a2,a2);
printf('    %f         %.4g\n',a3,a3);
printf('    %f         %.4g\n',a4,a4);
```

**Scilab code Exa 1.2** percentage accuracy

```
//example 1.2
```

```
//percentage accuracy
//page 9
clc;clear;close;
x=0.51;// the number given
n=2;//correcting upto 2 decimal places
dx=((10^-n)/2)
p_a=(dx/x)*100;//percentage accuracy
printf('the percentage accuracy of %f after
   correcting to two decimal places is %f',x,p_a);
```

**Scilab code Exa 1.3** absolute and relative errors

```
//example 1.3
//absolute and relative errors
//page 9
clc;clear;close;
X=22/7;//approximate value of pi
T_X=3.1415926;// true value of pi
A_E=T_X-X;//absolute error
R_E=A_E/T_X;//relative error
printf('Absolute Error = %0.7f \n Relative Error =
   %0.7f',A_E,R_E);
```

**Scilab code Exa 1.4** best approximation

```
//example 1.4
//best approximation
//page 10
clc;clear;close;
A_X=1/3;//the actual number
X1=0.30;
X2=0.33;
X3=0.34;
```

```
A_E1=abs(A_X-X1);
A_E2=abs(A_X-X2);
A_E3=abs(A_X-X3);
if(A_E1<A_E2)
if(A_E1<A_E3)
    B_A=X1;
end
end
if(A_E2<A_E1)
if(A_E2<A_E3)
    B_A=X2;
end
end
if(A_E3<A_E2)
if(A_E3<A_E1)
    B_A=X3;
end
end
printf('the best approximation of 1/3 is %.2g ',B_A)
   ;
```

**Scilab code Exa 1.5** relative error

```
//relative error
//example 1.5
//page 10
clc;clear;close;
n=8.6;// the corrected number
N=1;//the no is rounded to one decimal places
E_A=(10^-N)/2;
E_R=E_A/n;
printf('the relative error of the number is:%0.4f',
   E_R);
```

**Scilab code Exa 1.6** absolute error and relative error

```
//example 1.6
//absolute error and relative error
//page 10
clc;clear;close;
s=sqrt(3)+sqrt(5)+sqrt(7);//the sum square root of
    3,5,7
n=4;
Ea=3*((10^-n)/2);//absolute error
R_E=Ea/s;
printf('the sum of square roots  is %0.4g \n',s );
printf('the absolute error is %f \n',Ea);
printf('the relative error is %f ',R_E);
```

**Scilab code Exa 1.7** absolute error

```
//absolute error
//example 1.7
//page 10
clc;clear;close;
n=[0.1532 15.45 0.0000354 305.1 8.12 143.3 0.0212
    0.643 0.173];//original numbbers
//rounding all numbers to 2 decimal places
n=[305.1 143.3 0.15 15.45 0.00 8.12 0.02 0.64];
sum=0;
l=length(n);
for i=1:l
    sum=sum+n(i);
end
E_A=2*(10^-1)/2+7*(10^-2)/2;
printf('the absolute error is:%0.2f',E_A);
```

**Scilab code Exa 1.8** difference in 3 significant figures

```
//difference in 3 significant figures
//example 1.8
//page 11
clc;clear;close;
X1=sqrt(6.37);
X2=sqrt(6.36);
d=X1-X2;//difference between two numbers
printf('the differencecorrected to 3 significant
   figures is %0.3g',d);
```

**Scilab code Exa 1.10** relative error

```
//relative error
//example 1.10
//page 12
clc;clear;close;
a=6.54;b=48.64;c=13.5
da=0.01;db=0.02;dc=0.03;
s=(a^2*sqrt(b))/c^3;
disp(s,'s=');
r_err=2*(da/a)+(db/b)/2+3*(dc/c);
printf(' the relative error is :%f',r_err);
```

**Scilab code Exa 1.11** relative error

```
//relative error
//example 1.11
```

```
//page 13
clc;clear;close;
x=1;y=1;z=1;
u=(5*x*y^3)/z^3;
dx=0.001;dy=0.001;dz=0.001;
u_max=((5*y^2)/z^3)*dx+((10*x*y)/z^3)*dy+((15*x*y^2)
   /z^4)*dz;
r_err=u_max/u;
printf(' the relative error is :%f',r_err);
```

**Scilab code Exa 1.12** taylor series

```
//taylor series
//example 1.12
//page 12
clc;clear;close;
deff('y=f(x)','y=x^3+5*x-10');
deff('y=f1(x)','y=3*x^2-6*x+5')//first derivative
deff('y=f2(x)','y=6*x-6')//second derivative
deff('y=f3(x)','y=6')//third derivative
D=[f(0) f1(0) f2(0) f3(0)]
S1=0;
h=1;
for i=1:4
    S1=S1+h^(i-1)*D(i)/factorial(i-1);
end
printf('the third order taylors series approximation
    of f(1) is :%d',S1);
```

**Scilab code Exa 1.13** taylor series

```
//taylor series
//example 1.13
```

```
//page 16
clc;clear;close;
deff('y=f(x)','y=sin(x)');
deff('y=f1(x)','y=cos(x)');
deff('y=f2(x)','y=-sin(x)');
deff('y=f3(x)','y=-cos(x)');
deff('y=f4(x)','y=sin(x)');
deff('y=f5(x)','y=cos(x)');
deff('y=f6(x)','y=-sin(x)');
deff('y=f7(x)','y=-cos(x)');
D=[f(%pi/6) f1(%pi/6) f2(%pi/6) f3(%pi/6) f4(%pi/6) f5(%pi/6) f6(%pi/6) f7(%pi/6)];
S1=0;
h=%pi/6;
printf('order of approximation  computed value of sin(pi/3)   absolute eror\n\n');
for j=1:8
for i=1:j
    S1=S1+h^(i-1)*D(i)/factorial(i-1);
end
printf('%d                              %0.9f                    %0.9f\n',j,S1,abs(sin(%pi/3)-S1));
S1=0;
end
```

**Scilab code Exa 1.14** maclaurins expansion

```
//maclaurins expansion
//example 1.14
//page 18
clc;clear;close;
x=1;
n=8;//correct to 8 decimal places
for i=1:50
```

```
    if x/factorial(i)<(10^-8/2) then
        c=i;
        break;

    end
end
printf('no of terms needed to correct to 8 decimal
   places is :%d',c)
```

**Scilab code Exa 1.15** series approximation

```
//series apprixamation
//example 1.15
//page 18
clc;clear;close;
x=1/11;
S1=0;
for i=1:2:5
        S1=S1+(x^i/(i))
    end
printf('value of log(1.2) is : %0.8f\n\n',2*S1)
c=0;
for i=1:50
    if (1/11)^i/i<(2*10^-7) then
        c=i;
        break;
       end
end
printf('min no of terms needed to get value wuth
   same accuracy  is :%d',c)
```

# Chapter 2

# Solution of Algebraic and Transcendental Equation

**Scilab code Exa 2.1** bisection method

```
//example 2.1
//bisection method
//page 24
clc;clear;close;
deff('y=f(x)','y=x^3-x-1');
x1=1,x2=2;//f(1) is negative and f(2) is positive
d=0.0001;//for accuracy of root
c=1;
printf('Succesive approximations \t    x1\t    \tx2\t
      \tm\t    \tf(m)\n');
while abs(x1-x2)>d
    m=(x1+x2)/2;
printf('                              \t%f\t%f\t%f\t%f\n
    ',x1,x2,m,f(m));
    if f(m)*f(x1)>0
        x1=m;
    else
        x2=m;
end
```

```
c=c+1;// to count number of iterations
end
printf('the solution of equation after %i iteration
    is %g',c,m)
```

**Scilab code Exa 2.2** bisection method

```
//example 2.2
//bisection method
//page 25
clc;clear;close;
deff('y=f(x)','y=x^3-2*x-5');
x1=2,x2=3;//f(2) is negative and f(3) is positive
d=0.0001;//for accuracy of root
c=1;
printf('Succesive approximations \t    x1\t    \tx2\t
      \tm\t    \tf(m)\n');
while abs(x1-x2)>d
    m=(x1+x2)/2;
printf('                             \t%f\t%f\t%f\t%f\n
    ',x1,x2,m,f(m));
    if f(m)*f(x1)>0
        x1=m;
    else
        x2=m;
end
c=c+1;// to count number of iterations
end
printf('the solution of equation after %i iteration
    is %0.4g',c,m)
```

**Scilab code Exa 2.3** bisection method

```
//example 2.3
//bisection method
//page 26
clc;clear;close;
deff('y=f(x)','y=x^3+x^2+x+7');
x1=-3,x2=-2;//f(-3) is negative and f(-2) is
    positive
d=0.0001;//for accuracy of root
c=1;
printf('Succesive approximations \t    x1\t    \tx2\t
      \tm\t    \tf(m)\n');
while abs(x1-x2)>d
    m=(x1+x2)/2;
printf('                              \t%f\t%f\t%f\t%f\n
    ',x1,x2,m,f(m));
    if f(m)*f(x1)>0
        x1=m;
    else
        x2=m;
end
c=c+1;// to count number of iterations
end
printf('the solution of equation after %i iteration
    is %0.4g',c,m)
```

**Scilab code Exa 2.4** bisection method

```
//example 2.4
//bisection method
//page 26
clc;clear;close;
deff('y=f(x)','y=x*exp(x)-1');
x1=0,x2=1;//f(0) is negative and f(1) is positive
d=0.0005;//maximun tolerance value
c=1;
```

```
printf('Succesive approximations \t    x1\t    \tx2\t    \tm\t    \ttol\t     \tf(m)\n');
while abs((x2-x1)/x2)>d
    m=(x1+x2)/2;//tolerance value for each iteration
   tol=((x2-x1)/x2)*100;
printf('                              \t%f\t%f\t%f\t%f\t%f\n',x1,x2,m,tol,f(m));
    if f(m)*f(x1)>0
        x1=m;
    else
        x2=m;
end
c=c+1;// to count number of iterations
end
printf('the solution of equation after %i iteration is %0.4g',c,m)
```

---

**Scilab code Exa 2.5** bisection method

```
//example 2.5
//bisection method
//page 27
clc;clear;close;
deff('y=f(x)','y=4*exp(-x)*sin(x)-1');
x1=0,x2=0.5;//f(0) is negative and f(1) is positive
d=0.0001;//for accuracy of root
c=1;
printf('Succesive approximations \t    x1\t    \tx2\t    \tm\t    \t     \tf(m)\n');
while abs(x2-x1)>d
    m=(x1+x2)/2;
printf('                              \t%f\t%f\t%f\t%f\n',x1,x2,m,f(m));
    if f(m)*f(x1)>0
        x1=m;
```

```
    else
        x2=m;
end
c=c+1;// to count number of iterations
end
printf('the solution of equation after %i iteration is %0.3g',c,m)
```

**Scilab code Exa 2.6** false position method

```
//example 2.6
//false position method
//page 28
clc;clear;close
deff('y=f(x)','y=x^3-2*x-5');
a=2,b=3;//f(2) is negative and f(3)is positive
d=0.00001;
 printf('succesive iterations    \ta\t    b\t    f(a)\t    f(b)\t\  x1\n');
for i=1:25
    x1=b*f(a)/(f(a)-f(b))+a*f(b)/(f(b)-f(a));
    if(f(a)*f(x1))>0
        b=x1;
    else
        a=x1;
    end
    if abs(f(x1))<d
        break
    end
    printf('                              \t%f   %f   %f   %f   %f\n',a,b,f(a),f(b),x1);
end
printf('the root of the equation is  %f',x1);
```

**Scilab code Exa 2.7** false position method

```
//example 2.7
//false position method
//page 29
clc;clear;close
deff('y=f(x)','y=x^2.2-69');
a=5,b=6;//f(5) is negative and f(6)is positive
d=0.00001;
 printf('succesive iterations    \ta\t    b\t    f(a)\t    f(b)\t\  x1\n');
for i=1:25
    x1=b*f(a)/(f(a)-f(b))+a*f(b)/(f(b)-f(a));
    if(f(a)*f(x1))>0
        b=x1;
    else
        a=x1;
    end
    if abs(f(x1))<d
        break
    end
    printf('                          \t%f  %f  %f  %f  %f\n',a,b,f(a),f(b),x1);
end
printf('the root of the equation is  %f',x1);
```

**Scilab code Exa 2.8** false position method

```
//example 2.8
//false position method
//page 29
clc;clear;close
```

```
deff('y=f(x)','y=2*x-log10(x)-7');
a=3,b=4;//f(3) is negative and f(4)is positive
d=0.00001;
 printf('succesive iterations     \ta\t     b\t     f(a)\t     f(b)\t\   x1\n');
for i=1:25
    x1=b*f(a)/(f(a)-f(b))+a*f(b)/(f(b)-f(a));
    if(f(a)*f(x1))>0
        b=x1;
    else
        a=x1;
    end
    if abs(f(x1))<d
        break
    end
    printf('                          \t%f   %f   %f   %f   %f\n',a,b,f(a),f(b),x1);
end
printf('the root of the equation is  %0.4g',x1);
```

**Scilab code Exa 2.9** false position method

```
//example 2.9
//false position method
//page 30
clc;clear;close
deff('y=f(x)','y=4*exp(-x)*sin(x)-1');
a=0,b=0.5;//f(0) is negative and f(0.5)is positive
d=0.00001;
 printf('succesive iterations     \ta\t     b\t     f(a)\t     f(b)\t\   x1\n');
for i=1:25
    x1=b*f(a)/(f(a)-f(b))+a*f(b)/(f(b)-f(a));
    if(f(a)*f(x1))>0
        b=x1;
```

```
    else
        a=x1;
    end
    if abs(f(x1))<d
        break
    end
    printf('                          \t%f  %f  %f  %f  %f\n',a,b,f(a),f(b),x1);
end
printf('the root of the equation is  %f',x1);
```

**Scilab code Exa 2.10** iteration method

```
//example 2.10
//iteration method
//page 33
clc;clear,close;
deff('x=f(x)','x=1/(sqrt(x+1))');
x1=0.75,x2=0;
n=1;
d=0.0001;// accuracy opto 10^-4
c=0;// to count no of iterations
printf('successive iterations \t\x1\tf(x1)\n')
while abs(x1-x2)>d
printf('                          \t%f    %f\n',x1,f(x1))
x2=x1;
x1=f(x1);
c=c+1;
end
printf(' the root of the eqaution after %i iteration is %0.4g',c,x1)
```

**Scilab code Exa 2.11** iteration method

```
//example 2.11
//iteration method
//page34
clc;clear,close;
deff('x=f(x)','x=(cos(x)+3)/2');
x1=1.5;// as roots lies between 3/2 and pi/2
x2=0;
d=0.0001;// accuracy opto 10^-4
c=0;// to count no of iterations
printf('successive iterations \t\x1\tf(x1)\n')
while abs(x2-x1)>d
printf('                              \t%f     %f\n',x1,f(x1))
x2=x1;
x1=f(x1);
c=c+1;
end
printf(' the root of the eqation after %i iteration is %0.4g',c,x1)
```

**Scilab code Exa 2.12** iteration method

```
//example 2.12
//iteration method
//page 35
clc;clear,close;
deff('x=f(x)','x=exp(-x)');
x1=1.5;// as roots lies between 0 and 1
x2=0;
d=0.0001;// accuracy opto 10^-4
c=0;// to count no of iterations
printf('successive iterations \t\x1\t  f(x1)\n')
while abs(x2-x1)>d
```

```
printf('                             \t%f     %f\n',x1,f(x1)
   )
x2=x1;
x1=f(x1);
c=c+1;
end
printf(' the root of the eqaution after %i iteration
     is %0.4g',c,x1)
```

**Scilab code Exa 2.13** iteration method

```
//example 2.12
//iteration method
//page 35
clc;clear,close;
deff('x=f(x)','x=1+(sin(x)/10)');
x1=1;// as roots lies between 1 and pi evident from
   graph
x2=0;
d=0.0001;// accuracy opto 10^-4
c=0;// to count no of iterations
printf('successive iterations \t\x1\tf(x1)\n')
while abs(x2-x1)>d
printf('                             \t%f  %f\n',x1,f(x1))
x2=x1;
x1=f(x1);
c=c+1;
end
printf(' the root of the eqaution after %i iteration
     is %0.4g',c,x1)
```

**Scilab code Exa 2.14** aitkens process

```
//example 2.14
//aitken's process
//page 36
clc,clear,close
deff('x=f(x)','x=(3+cos(x))/2');
x0=1.5;
y=0;
e=0.0001;
c=0;
printf('successive iterations      \tx0\t       x1\t    x2\t      x3\t    y\n')
for i=1:10
    x1=f(x0),x2=f(x1),x3=f(x2);
    y=x3-((x3-x2)^2)/(x3-2*x2+x1);
    d=y-x0;
    x0=y;
    if abs(f(x0))<e then
        break;
    end
    c=c+1;
printf('                          \t%f    %f    %f    %f    %f\n',x0,x1,x2,x3,y)
end
printf('the root of the equation after %i iteration is %f',c,y);
```

**Scilab code Exa 2.15** newton raphson method

```
//example 2.15
//newton-raphson method
//page 39
clc;clear;close
deff('y=f(x)','y=x^3-2*x-5');
deff('y1=f1(x)','y1=3*x^2-2');// first derivative of the function
```

```
x0=2;// initial value
d=0.0001;
c=0;n=1;
printf('successive iterations \tx0\t         f(x0)\t
           f1(x0)\n');
while n==1
    x2=x0;
    x1=x0-(f(x0)/f1(x0));
    x0=x1;
printf('                          \t%f\t%f\t%f\n',x2,f(
   x1),f1(x1))
c=c+1;
if abs(f(x0))<d then
break;
end
end
printf('the root of %i iteration is:%f',c,x0);
```

**Scilab code Exa 2.16** newton raphson method

```
//example 2.16
//newton-raphson method
//page 40
clc;clear;close
deff('y=f(x)','y=x*sin(x)+cos(x)');
deff('y1=f1(x)','y1=x*cos(x)');// first derivation of
    the function
x0=%pi;// initial value
d=0.0001;
c=0;n=1;
printf('successive iterations \tx0\t         f(x0)\t
           f1(x0)\n');
while n==1
    x2=x0
    x1=x0-(f(x0)/f1(x0));
```

```
    x0=x1;
printf('                              \t%f\t%f\t%f\n',x2,f(x1),f1(x1))
c=c+1;
if abs(f(x0))<d then
break;
end
end
printf('the root of %i iteration is:%f',c,x0);
```

**Scilab code Exa 2.17** newton raphson method

```
//example 2.17
//newton-raphson method
//page 40
clc;clear;close
deff('y=f(x)','y=x*exp(x)-1');
deff('y1=f1(x)','y1=exp(x)+x*exp(x)');//first derivative of the function
x0=0;// initial value
d=0.0001;
c=0;n=1
printf('successive iterations \tx0\t           f(x0)\t           f1(x0)\n');
while n==1
    x2=x0;
    x1=x0-(f(x0)/f1(x0));
    x0=x1;
printf('                              \t%f\t%f\t%f\n',x2,f(x1),f1(x1))
c=c+1;
if abs(f(x0))<d then
break;
end
end
```

```
printf('the root of %i iteration is:%f',c,x0);
```

**Scilab code Exa 2.18** newton raphson method

```
//example 2.18
//newton-raphson method
//page 41
clc;clear;close
deff('y=f(x)','y=sin(x)-x/2');
deff('y1=f1(x)','y1=cos(x)-1/2');
x0=%pi/2;// initial value
d=0.0001;
c=0;n=1;
printf('successive iterations \tx0\t          f(x0)\t f1(x0)\n');
while n==1
    x2=x0;
    x1=x0-(f(x0)/f1(x0));
    x0=x1;

printf('                              \t%f\t%f\t%f\n',x2,f(x1),f1(x1))
c=c+1;
if abs(f(x0))<d then
    break;
end
end
printf('the root of %i iteration is:%0.4g',c,x0);
```

**Scilab code Exa 2.19** newton raphson method

```
//example 2.19
//newton-raphson method
```

```
//page 41
clc;clear;close
deff('y=f(x)','y=4*exp(-x)*sin(x)-1');
deff('y1=f1(x)','y1=cos(x)*4*exp(-x)-4*exp(-x)*sin(x)');
x0=0.2;// initial value
d=0.0001;
c=0;n=1;
printf('successive iterations \tx0\t       f(x0)\t       f1(x0)\n');
while n==1
    x2=x0;
    x1=x0-(f(x0)/f1(x0));
    x0=x1;
printf('                       \t%f\t%f\t%f\n',x2,f(x1),f1(x1))
c=c+1;
if abs(f(x0))<d then
break;
end
end
printf('the root of %i iteration is:%0.3g',c,x0);
```

**Scilab code Exa 2.20** newton raphson method

```
//example 2.20
//generalized newton-raphson method
//page 42
clc;clear;close;
deff('y=f(x)','y=x^3-x^2-x+1');
deff('y1=f1(x)','y1=3*x^2-2*x-1');
deff('y2=f2(x)','y2=6*x-2');
x0=0.8;// initial value to finf double root
n=1;
printf('successive iterations   \tx0\t    x1\t
```

```
          x2\n')
while n==1
x1=x0-(f(x0)/f1(x0));
x2=x0-(f1(x0)/f2(x0));
if abs(x1-x2)<0.000000001 then
    x0=(x1+x2)/2;
    break;
else
    x0=(x1+x2)/2;
end
printf('                              %f\t%f\t%f\n',x0,
   x1,x2);
end
printf('\n \nthe double root is at: %f',x0 );
```

**Scilab code Exa 2.21** ramanujans method

```
//ramanujan's method
//example 2.21
//page 45
clc;clear;close;
deff('y=f(x)','1-((13/12)*x-(3/8)*x^2+(1/24)*x^3)');
a1=13/12,a2=-3/8,a3=1/24;
b1=1;
b2=a1;
b3=a1*b2+a2*b1;
b4=a1*b3+a2*b2+a3*b1;
b5=a1*b4+a2*b3+a3*b2;
b6=a1*b5+a2*b4+a3*b3;
b7=a1*b6+a2*b5+a3*b4;
b8=a1*b7+a2*b6+a3*b5;
b9=a1*b8+a2*b7+a3*b6;
printf('\n\n%f',b1/b2);
printf('\n%f',b2/b3);
printf('\n%f',b3/b4);
```

```
printf('\n%f',b4/b5);
printf('\n%f',b5/b6);
printf('\n%f',b6/b7);
printf('\n%f',b7/b8);
printf('\n%f',b8/b9);
printf('\n it appears as if the roots are converging
    at 2')
```

**Scilab code Exa 2.22** ramanujans method

```
//ramanujan's method
//example 2.22
//page 46
clc;clear;close;
deff('y=f(x)','x+x^2+x^3/2+x^4/6+x^5/24');
a1=1,a2=1,a3=1/2,a4=1/6,a5=1/24;
b1=1;
b2=a2;
b3=a1*b2+a2*b1;
b4=a1*b3+a2*b2+a3*b1;
b5=a1*b4+a2*b3+a3*b2;
b6=a1*b5+a2*b4+a3*b3;
printf('\n%f',b1/b2);
printf('\n%f',b2/b3);
printf('\n%f',b3/b4);
printf('\n%f',b4/b5);
printf('\n%f',b5/b6);
printf('\n it appears as if the roots are converging
    at around %f',b5/b6);
```

**Scilab code Exa 2.23** ramanujans method

```
//ramanujan's method
```

```
//example 2.23
//page 47
clc;clear;close;
deff('y=f(x)','1-2*((3/2)*x+(1/4)*x^2-(1/48)*x^4+x^6/1440-x^8/80640)');
a1=3/2,a2=1/4,a3=0,a4=1/48,a5=0,a6=1/1440,a7=0,a8=-1/80640;
b1=1;
b2=a1;
b3=a1*b2+a2*b1;
b4=a1*b3+a2*b2+a3*b1;
b5=a1*b4+a2*b3+a3*b2;
b6=a1*b5+a2*b4+a3*b3;
b7=a1*b6+a2*b5+a3*b4;
b8=a1*b7+a2*b6+a3*b5;
b9=a1*b8+a2*b7+a3*b6;
printf('\n%f',b1/b2);
printf('\n%f',b2/b3);
printf('\n%f',b3/b4);
printf('\n%f',b4/b5);
printf('\n%f',b5/b6);
printf('\n%f',b6/b7);
printf('\n%f',b7/b8);
printf('\n it appears as if the roots are converging at around %f',b7/b8)
```

**Scilab code Exa 2.24** ramanujans method

```
//ramanujan's method
//example 2.23
//page 47
clc;clear;close;
deff('y=f(x)','1-(x-x^2/factorial(2)^2+x^3/factorial(3)^2-x^4/factorial(4)^2)');
a1=1,a2=-1/(factorial(2)^2),a3=1/(factorial(3)^2),a4
```

```
=-1/(factorial(4)^2),a5=-1/(factorial(5)^2),a6
   =1/(factorial(6)^2);
b1=1;
b2=a1;
b3=a1*b2+a2*b1;
b4=a1*b3+a2*b2+a3*b1;
b5=a1*b4+a2*b3+a3*b2;
printf('\n\n%f',b1/b2);
printf('\n%f',b2/b3);
printf('\n%f',b3/b4);
printf('\n%f',b4/b5);
printf('\n it appears as if the roots are converging
    at around %f',b4/b5);
```

**Scilab code Exa 2.25** secant method

```
//example 2.25
//secant method
//page 49
clc;clear;close;
deff('y=f(x)','y=x^3-2*x-5');
x1=2,x2=3// initial values
n=1;
c=0;
printf('successive iterations     \tx1              \tx2\
   t        x3\t          f(x3)\n')
while n==1
    x3=(x1*f(x2)-x2*f(x1))/(f(x2)-f(x1));
printf('                              \t%f\t%f\t%f\t%f\n
   ',x1,x2,x3,f(x3));
if f(x3)*f(x1)>0 then
x2=x3;
else
x1=x3;
end
```

```
if abs(f(x3))<0.000001 then
    break;
end
c=c+1;
end
printf('the root of the equation after %i iteration
   is: %f',c,x3 )
```

**Scilab code Exa 2.26** secant method

```
//example 2.26
//secant method
//page 50
clc;clear;close;
deff('y=f(x)','y=x*exp(x)-1');
x1=0,x2=1// initial values
n=1;
c=0;
printf('successive iterations     \tx1          \tx2\
   t      x3\t         f(x3)\n')
while n==1
    x3=(x1*f(x2)-x2*f(x1))/(f(x2)-f(x1));
printf('                          \t%f\t%f\t%f\t%f\n
   ',x1,x2,x3,f(x3));
if f(x3)*f(x1)>0 then
x2=x3;
else
x1=x3;
end
if abs(f(x3))<0.0001 then
    break;
end
c=c+1;
end
printf('the root of the equation after %i iteration
```

```
is: %0.4g',c,x3 )
```

---

**Scilab code Exa 2.27** mulllers method

```
// example 2.27
//mulller's method
//page 52
clc;clear;close;
deff ('y=f(x)','y=x^3-x-1');
x0=0,x1=1,x2=2;// initial values
n=1;c=0;
printf(' successive iterations      \tx0\t      x1\t      x2\t      f(x0)\t      f(x1)\t   f(x2)\n')
while n==1
    c=c+1;
y0=f(x0),y1=f(x1),y2=f(x2);
h2=x2-x1,h1=x1-x0;
d2=f(x2)-f(x1),d1=f(x1)-f(x0);
printf('                              \t%f\t   %f\t   %f\t   %f\t   %f\t   %f\n',x0,x1,x2,f(x0),f(x1),f(x2));
A=(d2/h2-d1/h1)/(h1+h2);
B=d2/h2+A*h2;;
S=sqrt(B^2-4*A*f(x2));
x3=x2-(2*f(x2))/(B+S);
E=abs((x3-x2)/x2)*100;
if E<0.003 then
    break;
else
    if c==1 then
  x2=x3;
end
if c==2 then
    x1=x2;
    x2=x3;
```

```
end
if c==3 then
    x0=x1;
    x1=x2;
    x2=x3;
 end
 if c==3 then
     c=0;
 end
end
end
printf('the required root is : %0.4f',x3)
```

**Scilab code Exa 2.28** graeffes method

```
//graeffe's method
//example 2.28
//page 55
clc;clear;close;
deff('y=f(x)','y=x^3-6*x^2+11*x-6');
x=poly(0,'x');
g=f(-x);
printf('the equation is:\n')
disp(g(x)*f(x));
A=[1 14 49 36];//coefficients of the above equation
printf('%0.4g\n',sqrt(A(4)/A(3)));
printf('%0.4g\n',sqrt(A(3)/A(2)));
printf('%0.4g\n',sqrt(A(2)/A(1)));
printf('the equation is:\n')
disp(g*(-1*g));
B=[1 98 1393 1296];
printf('%0.4g\n',(B(4)/B(3))^(1/4));
printf('%0.4g\n',(B(3)/B(2))^(1/4));
printf('%0.4g\n',(B(2)/B(1))^(1/4));
printf('It is apparent from the outputs that the
```

```
roots converge at 1 2 3')
```

**Scilab code Exa 2.29** quadratic factor by lins bairsttow method

```
//quadratic factor by lin's--bairsttow method
//example 2.29
//page 57
clc;clear;close;
deff('y=f(x)','y=x^3-x-1');
a=[-1 -1 0 1];
r1=1;s1=1;
b4=a(4);
deff('b3=f3(r)','b3=a(3)-r*a(4)');
deff('b2=f2(r,s)','b2=a(2)-r*a(3)+r^2*a(4)-s*a(4)');
deff('b1=f1(r,s)','b1=a(1)-s*a(3)+s*r*a(4)');
A=[1,1;2,-1];
C=[0;1];
X=A^-1*C;
dr=X(1,1);ds=X(2,1);
r2=r1+dr;s2=s1+ds;
//second pproximation
r1=r2;s1=s2;
b11=f1(r2,s2);
b22=f2(r2,s2);
h=0.001;
dr_b1=(f1(r1+h,s1)-f1(r1,s1))/h;
ds_b1=(f1(r1,s1+h)-f1(r1,s1))/h;
dr_b2=(f2(r1+h,s1)-f2(r1,s1))/h;
ds_b2=(f2(r1,s1+h)-f2(r1,s1))/h;
A=[dr_b1,ds_b1;dr_b2,ds_b2];
C=[-f1(r1,s1);-f2(r1,s2)];
X=A^-1*C;
r2=r1+X(1,1);
s2=s1+X(2,1);
printf(' roots correct to 3 decimal places are : %0
```

```
.3f       %0.3f',r2,s2);
```

**Scilab code Exa 2.31** method of iteration

```
//method of iteration
//example 2.31
//page 62
clc;clear;close;
deff('x=f(x,y)','(3*y*x^2+7)/10');
deff('y=g(x,y)','(y^2+4)/5');
h=0.0001;
x0=0.5;y0=0.5;
f1_dx=(f(x0+h,y0)-f(x0,y0))/h;
f1_dy=(f(x0,y0+h)-f(x0,y0))/h;
g1_dx=(g(x0+h,y0)-g(x0,y0))/h;
g1_dy=(g(x0+h,y0)-g(x0,y0))/h;
if f1_dx+f1_dy<1 & g1_dx+g1_dy<1  then
    printf('coditions for convergence is satisfied\n
        \n')
end
printf(' X\t         Y\t\n\n');
for i=1:10
    X=(3*y0*x0^2+7)/10;
    Y=(y0^2+4)/5;
    printf('%f\t       %f\t\n',X,Y);
    x0=X;y0=Y;
end
printf('\n\n CONVERGENCE AT (1  1) IS OBVIOUS');
```

**Scilab code Exa 2.32** newton raphson method

```
//newton raphson method
//example 2.32
```

```
//page 65
clc;clear;close;
deff('y=f(x,y)','y=3*y*x^2-10*x+7');
deff('x=g(y)','x=y^2-5*y+4');
hh=0.0001;
x0=0.5,y0=0.5;//initial values
f0=f(x0,y0);
g0=g(y0);
df_dx=(f(x0+hh,y0)-f(x0,y0))/hh;
df_dy=(f(x0,y0+hh)-f(x0,y0))/hh;
dg_dx=(g(y0)-g(y0))/hh;
dg_dy=(g(y0+hh)-g(y0))/hh;
D1=determ([df_dx,df_dy;dg_dx,dg_dy]);
h=determ([-f0,df_dy;-g0,dg_dy])/D1;
k=determ([df_dx,-f0;dg_dx,-g0])/D1;
x1=x0+h;
y1=y0+k;
f0=f(x1,y1);
g0=g(y1);
df_dx=(f(x1+hh,y1)-f(x1,y1))/hh;
df_dy=(f(x1,y1+hh)-f(x1,y1))/hh;
dg_dx=(g(y1)-g(y1))/hh;
dg_dy=(g(y1+hh)-g(y1))/hh;
D2=determ([df_dx,df_dy;dg_dx,dg_dy]);
h=determ([-f0,df_dy;-g0,dg_dy])/D2;
k=determ([df_dx,-f0;dg_dx,-g0])/D2;
x2=x1+h;
y2=y1+k;
printf(' the roots of the equation are x2=%f and y2=
   %f ',x2,y2);
```

**Scilab code Exa 2.33** newton raphson method

```
//newton raphson method
//example 2.33
```

```
//page 66
clc;clear;close;
deff('y=f(x,y)','y=x^2+y^2-1');
deff('x=g(x,y)','x=y-x^2');
hh=0.0001;
x0=0.7071,y0=0.7071;//initial values
f0=f(x0,y0);
g0=g(x0,y0);
df_dx=(f(x0+hh,y0)-f(x0,y0))/hh;
df_dy=(f(x0,y0+hh)-f(x0,y0))/hh;
dg_dx=(g(x0+hh,y0)-g(x0,y0))/hh;
dg_dy=(g(x0,y0+hh)-g(x0,y0))/hh;
D1=determ([df_dx,df_dy;dg_dx,dg_dy]);
h=determ([-f0,df_dy;-g0,dg_dy])/D1;
k=determ([df_dx,-f0;dg_dx,-g0])/D1;
x1=x0+h;
y1=y0+k;
f0=f(x1,y1);
g0=g(x1,y1);
df_dx=(f(x1+hh,y1)-f(x1,y1))/hh;
df_dy=(f(x1,y1+hh)-f(x1,y1))/hh;
dg_dx=(g(x1+hh,y1)-g(x1,y1))/hh;
dg_dy=(g(x1,y1+hh)-g(x1,y1))/hh;
D2=determ([df_dx,df_dy;dg_dx,dg_dy]);
h=determ([-f0,df_dy;-g0,dg_dy])/D2;
k=determ([df_dx,-f0;dg_dx,-g0])/D2;
x2=x1+h;
y2=y1+k;
printf(' the roots of the equation are x2=%f and y2=
   %f ',x2,y2);
```

**Scilab code Exa 2.34** newton raphson method

```
//newton raphson method
//example 2.33
```

```
//page 66
clc;clear;close;
deff('y=f(x,y)','y=sin(x)-y+0.9793');
deff('x=g(x,y)','x=cos(y)-x+0.6703');
hh=0.0001;
x0=0.5,y0=1.5;//initial values
f0=f(x0,y0);
g0=g(x0,y0);
df_dx=(f(x0+hh,y0)-f(x0,y0))/hh;
df_dy=(f(x0,y0+hh)-f(x0,y0))/hh;
dg_dx=(g(x0+hh,y0)-g(x0,y0))/hh;
dg_dy=(g(x0,y0+hh)-g(x0,y0))/hh;
D1=determ([df_dx,df_dy;dg_dx,dg_dy]);
h=determ([-f0,df_dy;-g0,dg_dy])/D1;
k=determ([df_dx,-f0;dg_dx,-g0])/D1;
x1=x0+h;
y1=y0+k;
f0=f(x1,y1);
g0=g(x1,y1);
df_dx=(f(x1+hh,y1)-f(x1,y1))/hh;
df_dy=(f(x1,y1+hh)-f(x1,y1))/hh;
dg_dx=(g(x1+hh,y1)-g(x1,y1))/hh;
dg_dy=(g(x1,y1+hh)-g(x1,y1))/hh;
D2=determ([df_dx,df_dy;dg_dx,dg_dy]);
h=determ([-f0,df_dy;-g0,dg_dy])/D2;
k=determ([df_dx,-f0;dg_dx,-g0])/D2;
x2=x1+h;
y2=y1+k;
printf('the roots of the equation are x2=%0.4f and
    y2=%0.4f ',x2,y2);
```

# Chapter 3

# interpolation

**Scilab code Exa 3.4** interpolation

```
//example 3.4
//interpolation
//page 86
clc;clear;close;
x=[1 3 5 7];
y=[24 120 336 720];
h=2//interval between values of x
c=1;
for i=1:3
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:2
    d2(c)=d1(i+1)-d1(i);
    c=c+1
end
c=1;
for i=1:1
    d3(c)=d2(i+1)-d2(i);
    c=c+1;
```

```
end

d=[d1(1) d2(1) d3(1)];
x0=8;//value at 8;
pp=1;
y_x=y(1);
p=(x0-1)/2;
for i=1:3
    pp=1;
    for j=1:i
    pp=pp*(p-(j-1))
    end
y_x=y_x+(pp*d(i))/factorial(i);
end
printf('value of function at %f is :%f',x0,y_x);
```

**Scilab code Exa 3.6** interpolation

```
//example 3.6
//interpolation
//page 87
clc;clear;close;
x=[15 20 25 30 35 40];
y=[0.2588190 0.3420201 0.4226183 0.5 0.5735764
   0.6427876];
h=5//interval between values of x
c=1;
for i=1:5
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:4
    d2(c)=d1(i+1)-d1(i);
    c=c+1
```

```
end
c=1;
for i=1:3
    d3(c)=d2(i+1)-d2(i);
    c=c+1;
end
c=1;
for i=1:2
    d4(c)=d3(i+1)-d3(i);
    c=c+1;
end
c=1;
for i=1:1
    d5(c)=d4(i+1)-d4(i);
    c=c+1;
end
c=1;
d=[d1(5) d2(4) d3(3) d4(2) d5(1)];
x0=38;//value at 38 degree
pp=1;
y_x=y(6);
p=(x0-x(6))/h;
for i=1:5
    pp=1;
    for j=1:i
    pp=pp*(p+(j-1))
end
y_x=y_x+((pp*d(i))/factorial(i));
end
printf('value of function at %i is :%f',x0,y_x);
```

**Scilab code Exa 3.7** interpolation

```
//example 3.7
//interpolation
```

```
//page 89
clc;clear;close;
x=[0 1 2 4];
y=[1 3 9 81];
//equation is y(5)−4∗y(4)+6∗y(2)−4∗y(2)+y(1)
y3=(y(4)+6*y(3)-4*y(2)+y(1))/4;
printf(' the value of missing term of table is :%d',
    y3);
```

**Scilab code Exa 3.8** interpolation

```
//example 3.8
//interpolation
//page 89
clc;clear;close;
x=[0.10 0.15 0.20 0.25 0.30];
y=[0.1003 0.1511 0.2027 0.2553 0.3093];
h=0.05//interval between values of x
c=1;
for i=1:4
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:3
    d2(c)=d1(i+1)-d1(i);
    c=c+1
end
c=1;
for i=1:2
    d3(c)=d2(i+1)-d2(i);
    c=c+1;
end
c=1;
for i=1:1
```

```
    d4(c)=d3(i+1)-d3(i);
    c=c+1;
end

d=[d1(1) d2(1) d3(1) d4(1)];
x0=0.12;//value at 0.12;
pp=1;
y_x=y(1);
p=(x0-x(1))/h;
for i=1:4
    pp=1;
    for j=1:i
    pp=pp*(p-(j-1))
    end
y_x=y_x+(pp*d(i))/factorial(i);
end
printf('value of function at %f is :%0.4g\n \n',x0,
   y_x);
d=[d1(4) d2(3) d3(2) d4(1)];
x0=0.26;//value at 0.26;
pp=1;
y_x=y(5);
p=(x0-x(5))/h;
for i=1:4
    pp=1;
    for j=1:i
    pp=pp*(p-(j-1))
    end
y_x=y_x+(pp*d(i))/factorial(i);
end
printf('value of function at %f is :%0.4g\n \n',x0,
   y_x);
d=[d1(4) d2(3) d3(2) d4(1)];
x0=0.40;//value at 0.40;
pp=1;
y_x=y(5);
p=(x0-x(5))/h;
for i=1:4
```

```
    pp=1;
    for j=1:i
    pp=pp*(p+(j-1))
    end
y_x=y_x+(pp*d(i))/factorial(i);
end
printf('value of function at %f is :%0.4g\n \n',x0,y_x);
d=[d1(4) d2(3) d3(2) d4(1)];
x0=0.50;//value at 0.50;
pp=1;
y_x=y(5);
p=(x0-x(5))/h;
printf('value of function at %f is :%0.5g\n \n',x0,y_x);
```

**Scilab code Exa 3.9** Gauss forward formula

```
//example 3.9
//Gauss' forward formula
//page 3.9
clc;clear;close;
x=[1.0 1.05 1.10 1.15 1.20 1.25 1.30];
y=[2.7183 2.8577 3.0042 3.1582 3.3201 3.4903 3.66693];
h=0.05//interval between values of x
c=1;
for i=1:6
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:5
    d2(c)=d1(i+1)-d1(i);
    c=c+1
```

```
end
c=1;
for i=1:4
    d3(c)=d2(i+1)-d2(i);
    c=c+1;
end
c=1;
for i=1:3
    d4(c)=d3(i+1)-d3(i);
    c=c+1;
end
c=1;
for i=1:2
    d5(c)=d4(i+1)-d4(i);
    c=c+1;
end
c=1;
for i=1:1
    d6(c)=d5(i+1)-d5(i);
    c=c+1;
end
d=[d1(4) d2(3) d3(3) d4(2) d5(1) d6(1)];
x0=1.17;//value at 1.17;
pp=1;
y_x=y(4);
p=(x0-x(4))/h;
for i=1:6
    pp=1;
    for j=1:i
    pp=pp*(p-(j-1))
    end
y_x=y_x+(pp*d(i))/factorial(i);
end
printf('value of function at %f is :%0.4g\n \n',x0,
   y_x);
```

**Scilab code Exa 3.10** practical interpolation

```
//practical interpolation
//example 3.10
//page 97
clc;clear;close;
x=[0.61 0.62 0.63 0.64 0.65 0.66 0.67];
y=[1.840431 1.858928 1.877610 1.896481 1.915541
   1.934792 1.954237];
h=0.01//interval between values of x
c=1;
for i=1:6
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:5
    d2(c)=d1(i+1)-d1(i);
    c=c+1
end
c=1;
for i=1:4
    d3(c)=d2(i+1)-d2(i);
    c=c+1;
end
c=1;
for i=1:3
    d4(c)=d3(i+1)-d3(i);
    c=c+1;
end
d=[d1(1) d2(1) d3(1) d4(1)];
x0=0.644;
p=(x0-x(4))/h;
y_x=y(4);
```

```
y_x=y_x+p*(d1(3)+d1(4))/2+p^2*(d2(2))/2;//stirling formula
printf(' the value at %f by stirling formula is : %f\n\n',x0,y_x);
y_x=y(4);
y_x=y_x+p*d1(4)+p*(p-1)*(d2(3)+d2(4))/2;
printf(' the value at %f by bessels formula is : %f\n\n',x0,y_x);
y_x=y(4);
q=1-p;
y_x=q*y(4)+q*(q^2-1)*d2(3)/2+p*y(5)+p*(q^2-1)*d2(4)/2;
printf(' the value at %f by everrets formula is : %f\n\n',x0,y_x);
```

**Scilab code Exa 3.11** practical interpolation

```
//practical interpolation
//example 3.11
//page 99
clc;clear;close;
x=[0.61 0.62 0.63 0.64 0.65 0.66 0.67];
y=[1.840431 1.858928 1.877610 1.896481 1.915541 1.934792 1.954237];
h=0.01//interval between values of x
c=1;
for i=1:6
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:5
    d2(c)=d1(i+1)-d1(i);
    c=c+1
end
```

```
c=1;
for i=1:4
    d3(c)=d2(i+1)-d2(i);
    c=c+1;
end
c=1;
for i=1:3
    d4(c)=d3(i+1)-d3(i);
    c=c+1;
end
d=[d1(1) d2(1) d3(1) d4(1)];
x0=0.638;
p=(x0-x(4))/h;
y_x=y(4);
y_x=y_x+p*(d1(3)+d1(4))/2+p^2*(d2(2))/2;//stirling formula
printf(' the value at %f by stirling formula is : %f\n\n',x0,y_x);
y_x=y(3);
p=(x0-x(3))/h;
y_x=y_x+p*d1(3)+p*(p-1)*(d2(2)/2);
printf(' the value at %f by bessels formula is : %f\n\n',x0,y_x);
```

**Scilab code Exa 3.12** practical interpolation

```
//practical interpolation
//example 3.12
//page 99
clc;clear;close;
x=[1.72 1.73 1.74 1.75 1.76 1.77 1.78];
y=[0.1790661479 0.1772844100 0.1755204006 0.1737739435 0.1720448638 0.1703329888 0.1686381473];
h=0.01//interval between values of x
```

```
c=1;
for i=1:6
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:5
    d2(c)=d1(i+1)-d1(i);
    c=c+1
end
c=1;
for i=1:4
    d3(c)=d2(i+1)-d2(i);
    c=c+1;
end
c=1;
for i=1:3
    d4(c)=d3(i+1)-d3(i);
    c=c+1;
end
x0=1.7475;
y_x=y(3);
p=(x0-x(3))/h;
y_x=y_x+p*d1(3)+p*(p-1)*((d2(2)+d2(3))/2)/2;
printf('the value at %f by bessels formula is : %0
    .10f\n\n',x0,y_x);
y_x=y(4);
q=1-p;
y_x=q*y(3)+q*(q^2-1)*d2(2)/6+p*y(4)+p*(p^2-1)*d2(2)
    /6;
printf('the value at %f by everrets formula is : %0
    .10f\n\n',x0,y_x);
```

**Scilab code Exa 3.13** lagranges interpolation formula

```
//example 3.13
//lagrange's interpolation formula
//page 104
clc;clear;close;
x=[300 304 305 307];
y=[2.4771 2.4829 2.4843 2.4871];
x0=301;
log_301=0;
poly(0,'x');
for i=1:4
    p=y(i);
    for j=1:4
        if i~=j then
            p=p*((x0-x(j) )/( x(i)-x(j)))
        end
    end
    log_301=log_301+p;
    end
disp(log_301,'log_301=');
```

**Scilab code Exa 3.14** lagranges interpolation formula

```
//example 3.14
//lagrange's interpolation formula
//page 105
clc;clear;close;
y=[4 12 19];
x=[1 3 4];
y_x=7;
Y_X=0;
poly(0,'y');
for i=1:3
    p=x(i);
    for j=1:3
        if i~=j then
```

```
            p=p*((y_x-y(j) )/( y(i)-y(j)))
        end
    end
    Y_X=Y_X+p;
    end
disp(Y_X,'Y_X=');
```

**Scilab code Exa 3.15** lagranges interpolation formula

```
//example 3.15
//lagrange's interpolation formula
//page 105
clc;clear;close;
x=[2 2.5 3.0];
y=[0.69315 0.91629 1.09861];
deff('y=l0(x)','y=(x-2.5)*(x-3.0)/(-0.5)*(-1.0)')
x=poly(0,'x');
disp(l0(x),'l0(x)=');
deff('y=l1(x)','y=((x-2.0)*(x-3.0))/((0.5)*(-0.5))')
x=poly(0,'x');
disp(l1(x),'l1(x)=');
deff('y=l2(x)','y=((x-2.0)*(x-2.5))/((1.0)*(0.5))')
x=poly(0,'x');
disp(l2(x),'l2(x)=');
f_x=l0(2.7)*y(1)+l1(2.7)*y(2)+l2(2.7)*y(3);
printf(' the calculated value is %f:',f_x);
printf('\n\n the error occured in the value is %0.9f
   ',abs(f_x-log(2.7)))
```

**Scilab code Exa 3.16** lagranges interpolation formula

```
//example 3.16
//lagrange's interpolation formula
```

```
//page 104
clc;clear;close;
x=[0 %pi/4 %pi/2];
y=[0 0.70711 1.0];
x0=%pi/6;
sin_x0=0;
poly(0,'x');
for i=1:3
    p=y(i);
    for j=1:3
        if j~=i then
            p=p*((x0-x(j) )/( x(i)-x(j)))
        end
    end
    sin_x0=sin_x0+p;
    end
disp(sin_x0,'sin_x0=');
```

**Scilab code Exa 3.17** lagranges interpolation

```
//lagrange's interpolation
//example 3.17
//page 106
clc;clear;close;
x=[0  3 4];
y=[-12 12 24];
//1 appears to be one the roots the polynomial
for i=1:3
    r_x(i)=y(i)/(x(i)-1);
end
deff('y=l0(x)','y=((x-3)*(x-4))/((-3)*(-4))')
x=poly(0,'x');
disp(l0(x),'l0(x)=');
deff('y=l1(x)','y=((x-0)*(x-4))/((3)*(-1))')
x=poly(0,'x');
```

```
disp(l1(x),'l1(x)=');
deff('y=l2(x)','y=((x-0)*(x-3))/((4)*(1))')
x=poly(0,'x');
disp(l2(x),'l2(x)=');
disp(l0(x)*r_x(1)+l1(x)*r_x(2)+l2(x)*r_x(3),'f_(x)='
    );
disp((x-1)*(l0(x)*r_x(1)+l1(x)*r_x(2)+l2(x)*r_x(3))
    ','the required polynimial is :')
```

---

**Scilab code Exa 3.18** error in lagranges interpolation formula

```
//error in lagrange's interpolation formula
//example 3.18
//page 107
clc;clear;close;
x=[2 2.5 3.0];
y=[0.69315 0.91629 1.09861];
deff('y=l0(x)','y=(x-2.5)*(x-3.0)/(-0.5)*(-1.0)')
x=poly(0,'x');
disp(l0(x),'l0(x)=');
deff('y=l1(x)','y=((x-2.0)*(x-3.0))/((0.5)*(-0.5))')
x=poly(0,'x');
disp(l1(x),'l1(x)=');
deff('y=l2(x)','y=((x-2.0)*(x-2.5))/((1.0)*(0.5))')
x=poly(0,'x');
disp(l2(x),'l2(x)=');
f_x=l0(2.7)*y(1)+l1(2.7)*y(2)+l2(2.7)*y(3);
printf(' the calculated value is %f:',f_x);
err=abs(f_x-log(2.7));
deff('y=R_n(x)','y=(((x-2)*(x-2.5)*(x-3))/6)');
est_err=abs(R_n(2.7)*(2/8))
if est_err>err then
    printf('\n\n the error agrees with the actual
        error')
end
```

**Scilab code Exa 3.19** error in lagranges interpolation formula

```
//error in lagrenge's interpolation
//example 3.19
//page 107
clc;clear;close;
x=[0 %pi/4 %pi/2];
y=[0 0.70711 1.0];
deff('y=l0(x)','y=((x-0)*(x-%pi/2))/((%pi/4)*(-%pi/4))')
x=poly(0,'x');
disp(l0(x),'l0(x)=');
deff('y=l1(x)','y=((x-0)*(x-%pi/4))/((%pi/2)*(%pi/4))')
x=poly(0,'x');
disp(l1(x),'l1(x)=');
f_x=l0(%pi/6)*y(2)+l1(%pi/6)*y(3);
err=abs(f_x-sin(%pi/6));
deff('y=f(x)','y=((x-0)*(x-%pi/4)*(x-%pi/2))/6');
if abs(f(%pi/6))>err then
    printf('\n\n the error agrees with the actual error')
end
```

**Scilab code Exa 3.21** hermites interpolation formula

```
//hermite's interpolation formula
//exammple 3.21
//page 110
clc;clear;close;
x=[2.0 2.5 3.0]
```

```
y=[0.69315 0.91629 1.09861]
deff('y=f(x)','y=log(x)')
h=0.0001;
for i=1:3
    y1(i)=(f(x(i)+h)-f(x(i)))/h;
end
deff('y=l0(x)','y=(x-2.5)*(x-3.0)/(-0.5)*(-1.0)')
a=poly(0,'x');
disp(l0(a),'l0(x)=');
deff('y=l1(x)','y=((x-2.0)*(x-3.0))/((0.5)*(-0.5))')
a=poly(0,'x');
disp(l1(a),'l1(x)=');
deff('y=l2(x)','y=((x-2.0)*(x-2.5))/((1.0)*(0.5))')
a=poly(0,'x');
disp(l2(a),'l2(x)=');
dl0=(l0(x(1)+h)-l0(x(1)))/h;
dl1=(l1(x(2)+h)-l1(x(2)))/h;
dl2=(l2(x(3)+h)-l2(x(3)))/h;
x0=2.7;
u0=[1-2*(x0-x(1))*dl0]*(l0(x0))^2;
u1=[1-2*(x0-x(2))*dl1]*(l1(x0))^2;
u2=[1-2*(x0-x(3))*dl2]*(l2(x0))^2;
v0=(x0-x(1))*l0(x0)^2;
v1=(x0-x(2))*l1(x0)^2;
v2=(x0-x(3))*l2(x0)^2;
H=u0*y(1)+u1*y(2)+u2*y(3)+v0*y1(1)+v1*y1(2)+v2*y1(3);
printf(' the approximate value of ln(%0.2f) is %0.6f:',x0,H);
```

**Scilab code Exa 3.22** newtons general interpolation formula

```
//newton's general interpolation formula
//example 3.22
//page 114
```

```
clc;clear;close;
x=[300 304 305 307];
y=[2.4771 2.4829 2.4843 2.4871];
for i=1:3
    d1(i)=(y(i+1)-y(i))/(x(i+1)-x(i));
end
for i=1:2
    d2(i)=(d1(i+1)-d1(i))/(x(i+2)-x(i));
end
x0=301;
log301=y(1)+(x0-x(1))*d1(1)+(x0-x(2))*d2(1);
printf(' valure of log(%d) is :%0.4f',x0,log301);
```

**Scilab code Exa 3.23** newtons divided formula

```
//example 3.22
//newton's divided formula
//page 114
clc;clear;close
x=[-1 0 3 6 7];
y=[3 -6 39 822 1611];
for i=1:4
    d1(i)=(y(i+1)-y(i))/(x(i+1)-x(i));
end
for i=1:3
    d2(i)=(d1(i+1)-d1(i))/(x(i+2)-x(i));
end
for i=1:2
    d3(i)=(d2(i+1)-d2(i))/(x(i+3)-x(i));
end
for i=1:1
    d4(i)=(d3(i+1)-d3(i))/(x(i+4)-x(i));
end
X=poly(0,'X')
f_x=y(1)+(X-x(1))*(d1(1))+(X-x(2))*(X-x(1))*d2(1)+(X
```

```
-x(1))*(X-x(2))*(X-x(3))*d3(1)+(X-x(1))*(X-x(2))
*(X-x(3))*(X-x(4))*d4(1)
disp(f_x,'the polynomial equation is =')
```

**Scilab code Exa 3.24** interpolation by iteration

```
//interpolation by iteration
//example 3.24
//page 116
clc;clear;close;
x=[300 304 305 307];
y=[2.4771 2.4829 2.4843 2.4871];
x0=301;
for i=1:3
    d=determ([y(i),(x(i)-x0);y(i+1),(x(i+1)-x0)])
    d1(i)=d/(x(i+1)-x(i));
end
for i=1:2
    d=determ([d1(i),(x(i+1)-x0);d1(i+1),(x(i+2)-x0)
       ])
    d2(i)=d/(x(i+2)-x(i+1));
end
for i=1:1
    d=determ([d2(i),(x(i+2)-x0);d2(i+1),(x(i+3)-x0)
       ])
    d3(i)=d/(x(i+3)-x(i+2));
end
printf(' the value of log(%d) is : %f',x0,d3(1))
```

**Scilab code Exa 3.25** inverse intrpolation

```
//inverse intrpolation
//example 3.25
```

```
//page 118
clc;clear;close;
x=[2 3 4 5];
y=[8 27 64 125];
for i=1:3
    d1(i)=y(i+1)-y(i);
end
for i=1:2
    d2(i)=d1(i+1)-d1(i);
end
for i=1:1
    d3(i)=d2(i+1)-d2(i);
end
yu=10;//square rooot of 10
y0=y(1);
d=[d1(1) d2(1) d3(1)];
u1=(yu-y0)/d1(1);
u2=((yu-y0-u1*(u1-1)*d2(1)/2)/d1(1));
u3=(yu-y0-u2*(u2-1)*d2(1)/2-u2*(u2-1)*(u2-2)*d3(1)/6)/d1(1);
u4=(yu-y0-u3*(u3-1)*d2(1)/2-u3*(u3-1)*(u3-2)*d3(1)/6)/d1(1);
u5=(yu-y0-u4*(u4-1)*d2(1)/2-u4*(u4-1)*(u4-2)*d3(1)/6)/d1(1);
printf(' %f \n %f \n %f \n %f \n %f \n ',u1,u2,u3,u4,u5);
printf(' the approximate square root of %d is: %0.3f',yu,x(1)+u5)
```

**Scilab code Exa 3.26** double interpolation

```
//double interpolation
//example 3.26
//page 119
clc;clear;close;
```

```
y=[0 1 2 3 4];
x=[0 1 4 9 16;2 3 6 11 18;6 7 10 15 22;12 13 16 21
   28;18 19 22 27 34];
printf(' y\t\n');
for i=1:5
    printf('\n%d',y(i));
end
printf('\n\n
   ---------------------------------------------------------------x
   -----------------------------------------------------------\n');
printf('0\t       1\t      2\t      3\t     4\t\n');
printf('
   ---------------------------------------------------------------------------------------------------------\
   n');
for i=1:5
    for j=1:5
printf('%d\t',x(i,j));
end
printf('\n');
end
//for x=2.5;
for i=1:5
    z(i)=(x(i,3)+x(i,4))/2;
end
//y=1.5;
Z=(z(2)+z(3))/2;
printf(' the interpolated value when x=2.5 and y=1.5
    is : %f',Z);
```

# Chapter 4

# least squares and fourier transform

**Scilab code Exa 4.1** least square curve fitting procedure

```
//example 4.1
//least square curve fitting procedure
//page 128
clc;clear;close;
x=[1 2 3 4 5];
y=[0.6 2.4 3.5 4.8 5.7];
for i=1:5
    x_2(i)=x(i)^2;
    x_y(i)=x(i)*y(i);
end
S_x=0,S_y=0,S_x2=0,S_xy=0,S1=0,S2=0;
for i=1:5
    S_x=S_x+x(i);
    S_y=S_y+y(i);
    S_x2=S_x2+x_2(i);
    S_xy=S_xy+x_y(i);
end
a1=(5*S_xy-S_x*S_y)/(5*S_x2-S_x^2);
a0=S_y/5-a1*S_x/5;
```

```
printf('x\t        y\t          x^2\t        x*y\t             (y-avg(S_y))\t(y-a0-a1x)^2\n\n');
for i=1:5
printf('%d\t      %0.2f\t      %d\t          %0.2f\t      %0.2f\t                              %.4f\t\n',x(i),y(i),x_2(i),x_y(i),(y(i)-S_y/5)^2,(y(i)-a0-a1*x(i))^2);
S1=S1+(y(i)-S_y/5)^2;
S2=S2+(y(i)-a0-a1*x(i))^2;
end
printf('------------------------------------------------------------------------------------------------n\n');
printf('%d\t        %0.2f\t        %d\t          %0.2f\t      %0.2f\t                          %0.4f\t\n\n',S_x,S_y,S_x2,S_xy,S1,S2);
cc=sqrt((S1-S2)/S1);//correlation coefficient
printf('the correlation coefficient is:%0.4f',cc);
```

**Scilab code Exa 4.2** least square curve fitting procedure

```
//example 4.2
//least square curve fitting procedure
//page 129
clc;clear;close;
x=[0 2 5 7];
y=[-1 5 12 20];
for i=1:4
    x_2(i)=x(i)^2;
    xy(i)=x(i)*y(i);
end
printf('x\t        y\t        x^2\t        xy\t   \n\n');
S_x=0,S_y=0,S_x2=0,S_xy=0;
for i=1:4
printf('%d\t      %d\t      %d\t           %d\t\n',x(i),y(i)
```

```
,x_2(i),xy(i));
S_x=S_x+x(i);
S_y=S_y+y(i);
S_x2=S_x2+x_2(i);
S_xy=S_xy+xy(i);
end
printf('%d\t      %d\t     %d\t       %d\t\n',S_x,S_y,S_x2,S_xy);
A=[4,S_x;S_x,S_x2];
B=[S_y;S_xy];
C=(A)^-1*B;
printf('Best straight line fit Y=%.4f+x(%.4f)',C(1,1),C(2,1));
```

**Scilab code Exa 4.3** least square curve fitting procedure

```
//example 4.3
//least square curve fitting procedure
//page 130
clc;clear;close;
x=[0 1 2 4 6];
y=[0 1 3 2 8];
z=[2 4 3 16 8];
for i=1:5
    x2(i)=x(i)^2;
    y2(i)=y(i)^2;
    z2(i)=z(i)^2;
    xy(i)=x(i)*y(i);
    zx(i)=z(i)*x(i);
    yz(i)=y(i)*z(i);
end
S_x=0,S_y=0,S_z=0,S_x2=0,S_y2=0,S_z2=0,S_xy=0,S_zx=0,S_yz=0;
for i=1:5
    S_x=S_x+x(i);
```

```
    S_y=S_y+y(i);
    S_z=S_z+z(i);
    S_x2=S_x2+x2(i);
    S_y2=S_y2+y2(i);
    S_z2=S_z2+z2(i);
    S_xy=S_xy+xy(i);
    S_zx=S_zx+zx(i);
    S_yz=S_yz+yz(i);
end
printf('x\t         y\t         z\t         x^2\t          xy\t          zx\t        y^2\t       yz\n\n');
for i=1:5
    printf('%d\t      %d\t      %d\t      %d\t      %d\t      %d\t      %d\t       %d\n',x(i),y(i),z(i),x2(i),xy(i),zx(i),y2(i),yz(i));
    end
printf('-------------------------------------------------------------------------------\n\n');
printf('%d\t      %d\t       %d\t       %d\t      %d\t      %d\t       %d\t      %d\n\n',S_x,S_y,S_z,S_x2,S_xy,S_zx,S_y2,S_yz);
A=[5,13,14;13,57,63;14,63,78];
B=[33;122;109];
C=A^-1*B;
printf('solution of above equation is:a=%d b=%d c=%d',C(1,1),C(2,1),C(3,1));
```

**Scilab code Exa 4.4** linearization of non linear law

```
//example 4.4
//linearization of non-linear law
//page 131
clc;clear;close;
x=[1 3 5 7 9];
```

```
y=[2.473 6.722 18.274 49.673 135.026];
for i=1:5
    Y(i)=log(y(i));
    x2(i)=x(i)^2;
    xy(i)=x(i)*Y(i);
end
S_x=0,S_y=0,S_x2=0,S_xy=0;
printf('X\t    Y=lny\t       X^2\t      XY\n\n');
for i=1:5
    printf('%d\t     %0.3f\t      %d\t    %0.3f\n',x(i),
       Y(i),x2(i),xy(i));
    S_x=S_x+x(i);
    S_y=S_y+Y(i);
    S_x2=S_x2+x2(i);
    S_xy=S_xy+xy(i);
end
printf('
   -------------------------------------------------------------------------------------------
   n\n')
printf('%d\t    %0.3f\t      %d\t      %0.3f\t\n\n',S_x,
   S_y,S_x2,S_xy);
A1=((S_x/5)*S_xy-S_x*S_y)/((S_x/5)*S_x2-S_x^2);
A0=(S_y/5)-A1*(S_x/5);
a=exp(A0);
printf('y=%0.3fexp(%0.2fx)',a,A1);
```

**Scilab code Exa 4.5** linearization of non linear law

```
//example 4.5
//linearization of non-linear law
//page 131
clc;clear;close;
x=[3 5 8 12];
y=[7.148 10.231 13.509 16.434];
for i=1:4
```

```
    X(i)=1/x(i);
    Y(i)=1/y(i);
    X2(i)=X(i)^2;
    XY(i)=X(i)*Y(i);
end
S_X=0,S_Y=0,S_X2=0,S_XY=0;
printf('X\t    Y\t    X^2\t    XY\t\n\n');
for i=1:4
printf('%0.3f\t    %0.3f\t    %0.3f\t    %0.3f\t\n',X(
    i),Y(i),X2(i),XY(i));
S_X=S_X+X(i);
S_Y=S_Y+Y(i);
S_X2=S_X2+X2(i);
S_XY=S_XY+XY(i);
end
printf('
    ------------------------------------------------------------------------
    n\n');
printf('%0.3f\t    %0.3f\t    %0.3f\t    %0.3f\n\n',
    S_X,S_Y,S_X2,S_XY);
A1=(4*S_XY-S_X*S_Y)/(4*S_X2-S_X^2);
Avg_X=S_X/4;
Avg_Y=S_Y/4;
A0=Avg_Y-A1*Avg_X;
printf('y=x/(%f+%f*x)',A1,A0);
```

**Scilab code Exa 4.6** curve fitting by polynomial

```
//example 4.6
//curve fitting by polynomial
//page 134
clc;clear;close;
x=[0 1 2];
y=[1 6 17];
for i=1:3
```

```
    x2(i)=x(i)^2;
    x3(i)=x(i)^3;
    x4(i)=x(i)^4;
    xy(i)=x(i)*y(i);
    x2y(i)=x2(i)*y(i);
end
printf('x\t    y\t    x^2\t    x^3\t    x^4\t    x*y\t    x^2*y\t\n\n');
S_x=0,S_y=0,S_x2=0,S_x3=0,S_x4=0,S_xy=0,S_x2y=0;
for i=1:3
    printf('%d\t    %d\t    %d\t    %d\t    %d\t    %d\t    %d\n',x(i),y(i),x2(i),x3(i),x4(i),xy(i),x2y(i));
    S_x=S_x+x(i);
    S_y=S_y+y(i);
    S_x2=S_x2+x2(i);
    S_x3=S_x3+x3(i);
    S_x4=S_x4+x4(i);
    S_xy=S_xy+xy(i);
    S_x2y=S_x2y+x2y(i);
end
printf('-------------------------------------------------------------------------------------\n\n');
printf('%d\t    %d\t    %d\t    %d\t    %d\t    %d\t    %d\n',S_x,S_y,S_x2,S_x3,S_x4,S_xy,S_x2y);
A=[3,S_x,S_x2;S_x,S_x2,S_x3;S_x2,S_x3,S_x4];
B=[S_y;S_xy;S_x2y];
C=A^-1*B;
printf('a=%d   b=%d   c=%d \n\n',C(1,1),C(2,1),C(3,1));
printf('exact polynomial :%d + %d*x +%d*x^2',C(1,1),C(2,1),C(3,1))
```

**Scilab code Exa 4.7** curve fitting by polynomial

```
//example 4.7
//curve fitting by polynomial
//page 134
clc;clear;close;
x=[1 3 4 6];
y=[0.63 2.05 4.08 10.78];
for i=1:4
    x2(i)=x(i)^2;
    x3(i)=x(i)^3;
    x4(i)=x(i)^4;
    xy(i)=x(i)*y(i);
    x2y(i)=x2(i)*y(i);
end
printf('x\t    y\t    x^2\t    x^3\t    x^4\t    x*y\t
    x^2*y\t\n\n');
S_x=0,S_y=0,S_x2=0,S_x3=0,S_x4=0,S_xy=0,S_x2y=0;
for i=1:4
    printf('%d\t    %0.3f\t    %d\t    %d\t    %d\t    %0
        .3f\t    %d\n',x(i),y(i),x2(i),x3(i),x4(i),xy(
        i),x2y(i));
    S_x=S_x+x(i);
    S_y=S_y+y(i);
    S_x2=S_x2+x2(i);
    S_x3=S_x3+x3(i);
    S_x4=S_x4+x4(i);
    S_xy=S_xy+xy(i);
    S_x2y=S_x2y+x2y(i);
end
printf('
    ------------------------------------------------------------------------------------------------
    n\n');
printf('%d\t    %0.3f\t    %d\t    %d\t    %d\t    %0.3f\
    t    %0.3f\n',S_x,S_y,S_x2,S_x3,S_x4,S_xy,S_x2y);
A=[4,S_x,S_x2;S_x,S_x2,S_x3;S_x2,S_x3,S_x4];
B=[S_y;S_xy;S_x2y];
C=A^-1*B;
printf('a=%0.2f   b=%0.2f   c=%0.2f  \n\n',C(1,1),C
    (2,1),C(3,1));
```

```
printf('exact polynomial :%0.2f + %0.2f*x +%0.2f*x^2',C(1,1),C(2,1),C(3,1))
```

---

**Scilab code Exa 4.8** curve fitting by sum of exponentials

```
//curve fitting by sum of exponentials
//example 4.8
//page 137
clc;clear;close;
x=[1 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1. 1.9];
y=[1.54 1.67 1.81 1.97 2.15 2.35 2.58 2.83 3.11];
s1=y(1)+y(5)-2*y(3);
h=x(2)-x(1);
I1=0;
for i=1:3
    if i==1|i==3 then
       I1=I1+y(i)

  elseif (modulo(i,2))==0 then
           I1=I1+4*y(i)

   elseif (modulo(i,2))~=0 then
          I1=I1+2*y(i)
           end
    end
    I1=(I1*h)/3

I2=0;
for i=3:5
    if i==3|i==5 then
       I2=I2+y(i)

  elseif (modulo(i,2))==0 then
           I2=I2+4*y(i)

```

```
   elseif (modulo(i,2))~=0 then
          I2=I2+2*y(i)
           end
    end
    I2=(I2*h)/3;
    for i=1:5
        y1(i)=(1.0-x(i))*y(i);
    end
    for i=5:9
        y2(i)=(1.4-x(i))*y(i);
    end
I3=0;
for i=1:3
    if i==1|i==3 then
       I3=I3+y1(i)
 elseif (modulo(i,2))==0 then
           I3=I3+4*y1(i)
       elseif (modulo(i,2))~=0 then
          I3=I3+2*y1(i)
           end
    end
    I3=(I3*h)/3
I4=0;
for i=3:5
    if i==3|i==5 then
       I4=I4+y2(i)

  elseif (modulo(i,2))==0 then
           I4=I4+4*y2(i)

   elseif (modulo(i,2))~=0 then
          I4=I4+2*y2(i)
           end
    end
    I4=(I4*h)/3;
    s2=y(5)+y(9)-2*y(7);
I5=0;
for i=5:7
```

```
    if i==5|i==7 then
       I5=I5+y(i)
 elseif (modulo(i,2))==0 then
           I5=I5+4*y(i)

   elseif (modulo(i,2))~=0 then
          I5=I5+2*y(i)
           end
    end
    I5=(I5*h)/3;
I6=0;
for i=7:9
    if i==7|i==9 then
       I6=I6+y(i)

  elseif (modulo(i,2))==0 then
           I6=I6+4*y(i)

   elseif (modulo(i,2))~=0 then
          I6=I6+2*y(i)
           end
    end
I6=(I6*h)/3;
I7=0;
for i=5:7
    if i==5|i==7 then
       I7=I7+y2(i)

  elseif (modulo(i,2))==0 then
           I7=I7+4*y2(i)

   elseif (modulo(i,2))~=0 then
          I7=I7+2*y2(i)
           end
    end
    I7=(I7*h)/3;
I8=0;
for i=7:9
```

```
    if i==7|i==9 then
       I8=I8+y2(i)

  elseif (modulo(i,2))==0 then
           I8=I8+4*y2(i)

   elseif (modulo(i,2))~=0 then
          I8=I8+2*y2(i)
           end
    end
    I8=(I8*h)/3;
A=[1.81 2.180;2.88 3.104];
C=[2.10;3.00];
Z=A^-1*C
X=poly(0,'X');
y=X^2-Z(1,1)*X-Z(2,1);
R=roots(y)
printf(' the unknown value of equation is %1.0f   %1
   .0f',R(1,1),R(2,1));
```

**Scilab code Exa 4.9** linear weighted least approx

```
//linear weighted least approx
//example 4.9
//page 139
clc;clear;close;
x=[0 2 5 7];
y=[-1 5 12 20];
w=10;//given weight 10;
W=[1 1 10 1];
for i=1:4
    Wx(i)=W(i)*x(i);
    Wx2(i)=W(i)*x(i)^2;
    Wx3(i)=W(i)*x(i)^3;
    Wy(i)=W(i)*y(i);
```

```
    Wxy(i)=W(i)*x(i)*y(i);
   end
S_x=0,S_y=0,S_W=0,S_Wx=0,S_Wx2=0,S_Wy=0,S_Wxy=0;
for i=1:4
    S_x=S_x+x(i)
    S_y=S_y+y(i)
    S_W=S_W+W(i)
    S_Wx=S_Wx+Wx(i)
    S_Wx2=S_Wx2+Wx2(i)
    S_Wy=S_Wy+Wy(i)
    S_Wxy=S_Wxy+Wxy(i)
end
A=[S_W,S_Wx;S_Wx,S_Wx2];
C=[S_Wy;S_Wxy];
printf('x\t   y\t   W\t   Wx\t   Wx^2\t   Wy\t   Wxy
   \t\n\n');
for i=1:4
    printf('%d\t   %d\t   %d\t    %d\t   %d\t   %d\t
          %d\t\n',x(i),y(i),W(i),Wx(i),Wx2(i),Wy(i)
        ,Wxy(i))
end
printf('
   ------------------------------------------------------------------------------------------
   n\n');
printf('%d\t   %d\t   %d\t   %d\t   %d\t   %d\t
   %d\t\n',S_x,S_y,S_W,S_Wx,S_Wx2,S_Wy,S_Wxy);
X=A^-1*C;
printf('\n\nthe equation is y=%f+%fx',X(1,1),X(2,1))
```

**Scilab code Exa 4.10** linear weighted least approx

```
//linear weighted least approx
//example 4.10
//page 139
clc;clear;close;
```

```
x=[0 2 5 7];
y=[-1 5 12 20];
w=100;//given weight 100;
W=[1 1 100 1];
for i=1:4
    Wx(i)=W(i)*x(i);
    Wx2(i)=W(i)*x(i)^2;
    Wx3(i)=W(i)*x(i)^3;
    Wy(i)=W(i)*y(i);
    Wxy(i)=W(i)*x(i)*y(i);
   end
S_x=0,S_y=0,S_W=0,S_Wx=0,S_Wx2=0,S_Wy=0,S_Wxy=0;
for i=1:4
    S_x=S_x+x(i)
    S_y=S_y+y(i)
    S_W=S_W+W(i)
    S_Wx=S_Wx+Wx(i)
    S_Wx2=S_Wx2+Wx2(i)
    S_Wy=S_Wy+Wy(i)
    S_Wxy=S_Wxy+Wxy(i)
end
A=[S_W,S_Wx;S_Wx,S_Wx2];
C=[S_Wy;S_Wxy];
printf('x\t    y\t    W\t    Wx\t    Wx^2\t    Wy\t    Wxy\t\n\n');
for i=1:4
    printf('%d\t    %d\t    %d\t     %d\t    %d\t    %d\t        %d\t\n',x(i),y(i),W(i),Wx(i),Wx2(i),Wy(i),Wxy(i))
end
printf('--------------------------------------------------------------------------------------------\n\n');
printf('%d\t    %d\t    %d\t    %d\t    %d\t    %d\t    %d\t\n',S_x,S_y,S_W,S_Wx,S_Wx2,S_Wy,S_Wxy);
X=A^-1*C;
printf('\n\nthe equation is y=%f+%fx',X(1,1),X(2,1))
printf('\n\nthe value of y(5) is %f',X(1,1)+X(2,1)
```

```
*5)
```

**Scilab code Exa 4.11** least square for quadratic equations

```
//least square for quadratic equations
//example 4.11
//page 141
clc;clear;close;
I1=integrate('1','x',0,%pi/2);
I2=integrate('x','x',0,%pi/2);
I3=integrate('x^2','x',0,%pi/2);
I4=integrate('x^3','x',0,%pi/2);
I5=integrate('x^4','x',0,%pi/2);
I6=integrate('sin(x)','x',0,%pi/2);
I7=integrate('x*sin(x)','x',0,%pi/2);
I8=integrate('x^2*sin(x)','x',0,%pi/2);
printf('the equations are:\n\n');
A=[I1,I2,I3;I2,I3,I4;I3,I4,I5];
C=[I6;I7;I8];
X=A^-1*C;
printf(' the quadratic equation is of the form %f+
    %fx+%fx^2',X(1,1),X(2,1),X(3,1));
//value of sin pi/4
y=X(1,1)+X(2,1)*%pi/4+X(3,1)*(%pi/4)^2
printf( '\n\nsin(pi/4)=%0.9f',y)
printf('\n\nerror in the preecing solution %0.9f',
    abs(y-sin(%pi/4)))
```

**Scilab code Exa 4.20** cooley Tukey method

```
//cooley-Tukey method
//example 4.20
//page 168
```

```
clc;clear;close;
f=[1,2,3,4,4,3,2,1];
F1(1,1)=f(1)+f(5);
F1(1,2)=f(1)-f(5);
F1(2,1)=f(3)+f(7);
F1(2,2)=f(3)-f(7);
F1(3,1)=f(2)+f(6);
F1(3,2)=f(2)-f(6);
F1(4,1)=f(4)+f(8);
F1(4,2)=f(4)-f(8);
printf('the solutions after first key equation\n\n')
disp(F1);
F2(1,1)=F1(1,1)+F1(2,1);
F2(1,2)=F1(1,1)+F1(2,1);
F2(2,1)=F1(1,2)+%i*F1(3,2);
F2(2,2)=F1(3,2)-%i*F1(4,2);
F2(3,1)=F1(1,1)-F1(2,1);
F2(3,2)=F1(1,1)-F1(2,1);
F2(4,1)=F1(1,2)+%i*F1(2,2);
F2(4,2)=F1(3,2)-%i*F1(1,2);
printf('the solutions after second key equation\n\n')
disp(F2);

W=[1,(1-%i)/sqrt(2),-%i,-(1+%i)/sqrt(2),-1,-(1-%i)/sqrt(2),%i,(1+%i)/sqrt(2)];
F3(1)=F2(1,1)+F2(1,2);
F3(2)=F2(2,1)+W(2)*F2(2,2);
F3(3)=F2(3,1)+F2(3,2);
F3(4)=F2(4,1)+W(4)*F2(4,2);
F3(5)=F2(3,1)+F2(3,2);
F3(6)=F2(2,1)+W(6)*F2(2,2);
F3(7)=F2(3,1)+F2(3,2);
F3(8)=F2(4,1)+W(8)*F2(4,2);
printf('the solutions after third key equation\n\n')
disp(F3);
```

# Chapter 5

# spline functions

**Scilab code Exa 5.1** linear splines

```
//linear splines
//example 5.1
//page 182
clc;clear;close;
X=[1 2 3];
y=[-8 -1 18];
m1=(y(2)-y(1))/(X(2)-X(1));
deff('y1=s1(x)','y1=y(1)+(x-X(1))*m1');
m2=(y(3)-y(2))/(X(3)-X(2));
deff('y2=s2(x)','y2=y(2)+(x-X(2))*m2');
a=poly(0,'x');
disp(s1(a));
disp(s2(a));
printf(' the value of function at 2.5 is %0.2f: ',s2
   (2.5));
```

**Scilab code Exa 5.2** quadratic splines

```
//quadratic splines
//example 5.2
//page 18
clc;clear;close;
X=[1 2 3];
y=[-8 -1 18];
h=X(2)-X(1);
m1=(y(2)-y(1))/(X(2)-X(1));
m2=(2*(y(2)-y(1)))/h-m1;
m3=(2*(y(3)-y(2)))/h-m2;
deff('y2=s2(x)','y2=(-(X(3)-x)^2*m1)/2+((x-X(2))^2*m3)/2+y(2)+m2/2');
a=poly(0,'x');
disp(s2(a));
printf('the value of function is 2.5: %0.2f',s2(2.5));
x=2.0;
h=0.01;
deff('y21=s21(x,h)','y21=(s2(x+h)-s2(x))/h');
d1=s21(x,h);
printf('\n\nthe first derivative at 2.0 :%0.2f',d1);
```

**Scilab code Exa 5.3** cubic splines

```
//cubic splines
//example 5.3
//page 188
clc;clear;close;
X=[1 2 3];
y=[-8 -1 18];
M1=0,M2=8,M3=0;
h=1;
deff('y=s1(x)','y=3*(x-1)^3-8*(2-x)-4*(x-1)')
deff('y=s2(x)','y=3*(3-x)^3+22*x-48');
h=0.0001;n=2.0;
```

```
D=(s2(n+h)-s2(n))/h;
a=poly(0,'x');
disp(s1(a),' s1(x)=');
disp(s2(a),'s2(x)=');
disp(s2(2.5),'y(2.5)=');
disp(D,'y1(2.0)=');
```

**Scilab code Exa 5.4** cubic splines

```
//cubic spline
//example 5.4
//page 189
clc;clear;close;
x=[0 %pi/2 %pi]
y=[0 1 0]
h=x(2)-x(1)
M0=0;M2=0;
M1=((6*(y(1)-2*y(2)+y(3))/h^2)-M0-M2)/4;
X=%pi/6;
s1=(((x(2)-X)^3)*(M0/6)+((X-x(1))^3)*M1/6+(y(1)-(h^2)*M0/6)*(x(2)-X)+(y(2)-(h^2)*M1/6)*(X-x(1)))/h;
x=[0 %pi/4 %pi/2 3*%pi/4 %pi];
y=[0 1.414 1 1.414];
M0=0,M4=0;
A=[4 1 0;1 4 1;0 1 4];//calculating value of M1 M2 M3 by matrix method
C=[-4.029;-5.699;-4.029];
B=A^-1*C
printf('M0=%f\t    M1=%f\t    M2=%f\t    M3=%f\t    M4=%f\t\n\n',M0,B(1,1),B(2,1),B(3,1),M4);
h=%pi/4;
X=%pi/6;
s1=[-0.12408*X^3+0.7836*X]/h;
printf(' the value of sin(pi/6) is:%f',s1)
```

**Scilab code Exa 5.5** cubic splines

```
//cubic spline
//example 5.5
//page 191
clc;clear;close;
x=[1 2 3];
y=[6 18 42];
m0=40;
m1=(3*(y(3)-y(1))-m0)/4;
X=poly(0,'X');
s1=m0*((x(2)-X)^2)*(X-x(1))-m1*((X-x(1))^2)*(x(2)-X)
    +y(1)*((x(2)-X)^2)*[2*(X-x(1))+1]+y(2)*((X-x(1))
    ^2)*[2*(x(2)-X)+1];
disp(s1,'s1=');
```

**Scilab code Exa 5.7** surface fitting by cubic spline

```
//surface fitting by cubic spline
//example 5.7
//page 195
clc;clear;close;
z=[1 2 9;2 3 10;9 10 17];
deff('y=L0(x)','y=x^3/4-5*x/4+1');
deff('y=L1(x)','y=-x^3/2+3*x/2');
deff('y=L2(x)','y=x^3/4-x/4');
x=0.5;y=0.5;
S=0;
S=S+L0(x)*(L0(x)*z(1,1)+L1(x)*z(1,2)+L2(x)*z(1,3));
S=S+L1(x)*(L0(x)*z(2,1)+L1(x)*z(2,2)+L2(x)*z(2,3));
S=S+L2(x)*(L0(x)*z(3,1)+L1(x)*z(3,2)+L2(x)*z(3,3));
printf('approximated value of z(0.5 0.5)=%f\n\n',S);
```

```
printf(' error in the approximated value : %f',(abs
    (1.25-S)/1.25)*100)
```

**Scilab code Exa 5.8** cubic B splines

```
//cubic B-splines
//example 5.8
//page 200
clc;clear;close;
k=[0 1 2 3 4];
pi=[0 0 4 -6 24];
x=1;
S=0;
for i=3:5
    S=S+((k(i)-x)^3)/(pi(i));
end
printf( ' the cubic splines for x=1 is %f\n\n',S);
S=0;
x=2;
for i=4:5
    S=S+((k(i)-x)^3)/(pi(i));
end
printf( ' the cubic splines for x=2 is  %f\n\n',S);
```

**Scilab code Exa 5.9** cubic B spline

```
//cubic B-spline
//example 5.8
//page 201
clc;clear;close;
k=[0 1 2 3 4];
x=1;//for x=1
s11=0;s13=0;s14=0;
```

```
s24=0;
s12=1/(k(3)-k(2));
s22=((x-k(1))*s11+(k(3)-x)*s12)/(k(3)-k(1));
s23=((x-k(2))*s11+(k(4)-x)*s13)/(k(4)-k(2));
s33=((x-k(1))*s22+(k(4)-x)*s23)/(k(4)-k(1));
s34=((x-k(2))*s23+(k(5)-x)*s24)/(k(5)-k(2));
s44=((x-k(1))*s33+(k(5)-x)*s34)/(k(5)-k(1));
printf('s11=%f\t    s22=%f\t    s23=%f\t    s33=%f\t    s34=%f\t    s44=%f\n\n',s11,s22,s23,s33,s34,s44);
x=2;//for x=2;
s11=0;s12=0,s14=0;s22=0;
s13=1/(k(4)-k(3));
s23=((x-k(2))*s12+(k(4)-x)*s13)/(k(4)-k(2));
s24=((x-k(3))*s13+(k(5)-x)*s14)/(k(3)-k(1));
s33=((x-k(1))*s22+(k(4)-x)*s23)/(k(4)-k(1));
s34=((x-k(2))*s23+(k(5)-x)*s24)/(k(5)-k(2));
s44=((x-k(1))*s33+(k(5)-x)*s34)/(k(5)-k(1));
printf('s13=%f\t    s23=%f\t    s24=%f\t    s33=%f\t    s34=%f\t    s44=%f\n\n',s13,s23,s24,s33,s34,s44);
```

# Chapter 6

# Numerical Diffrentiation and Integration

**Scilab code Exa 6.1** numerical diffrentiation by newtons difference formula

```
//example 6.1
//numerical diffrentiation by newton's difference
   formula
//page 210
clc;clear;close
x=[1.0 1.2 1.4 1.6 1.8 2.0 2.2];
y=[2.7183 3.3201 4.0552 4.9530 6.0496 7.3891
   9.0250];
c=1;
for i=1:6
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:5
    d2(c)=d1(i+1)-d1(i);
    c=c+1;
end
```

```
c=1;
for i=1:4
    d3(c)=d2(i+1)-d2(i);
    c=c+1;
end
c=1;
for i=1:3
    d4(c)=d3(i+1)-d3(i);
    c=c+1;
end
c=1;
for i=1:2
    d5(c)=d4(i+1)-d4(i);
    c=c+1;
end
c=1;
for i=1:1
    d6(c)=d5(i+1)-d5(i);
    c=c+1;
end
x0=1.2//first and second derivative at 1.2
h=0.2;
f1=((d1(2)-d2(2)/2+d3(2)/3-d4(2)/4+d5(2)/5)/h);
printf('the first derivative of fuction at 1.2 is:%f
   \n',f1);
f2=(d2(2)-d3(2)+(11*d4(2))/12-(5*d5(2))/6)/h^2;
printf('the second derivative of fuction at 1.2 is:
   %f\n',f2);
```

**Scilab code Exa 6.2** numerical diffrentiation by newtons difference formula

```
//example 6.2
//numerical diffrentiation by newton's difference
   formula
```

```
//page 211
clc;clear;close
x=[1.0 1.2 1.4 1.6 1.8 2.0 2.2];
y=[2.7183 3.3201 4.0552 4.9530 6.0496 7.3891
   9.0250];
c=1;
for i=1:6
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:5
    d2(c)=d1(i+1)-d1(i);
    c=c+1;
end
c=1;
for i=1:4
    d3(c)=d2(i+1)-d2(i);
    c=c+1;
end
c=1;
for i=1:3
    d4(c)=d3(i+1)-d3(i);
    c=c+1;
end
c=1;
for i=1:2
    d5(c)=d4(i+1)-d4(i);
    c=c+1;
end
c=1;
for i=1:1
    d6(c)=d5(i+1)-d5(i);
    c=c+1;
end
x0=2.2//first and second derivative at 2.2
h=0.2;
f1=((d1(6)+d2(5)/2+d3(4)/3+d4(3)/4+d5(2)/5)/h);
```

```
printf('the first derivative of fuction at 1.2 is:%f\n',f1);
f2=(d2(5)+d3(4)+(11*d4(3))/12+(5*d5(2))/6)/h^2;
printf('the second derivative of fuction at 1.2 is:%f\n',f2);
x1=2.0;// first derivative also at 2.0
f1=((d1(5)+d2(4)/2+d3(3)/3+d4(2)/4+d5(1)/5+d6(1)/6)/h);
printf('\n \nthe first derivative of function at 1.2 is:%f\n',f1);
```

**Scilab code Exa 6.3** numerical diffrentiation by newtons difference formula

```
//example 6.3
//numerical diffrentiation by newton's difference formula
//page 211
clc;clear;close
x=[1.0 1.2 1.4 1.6 1.8 2.0 2.2];
y=[2.7183 3.3201 4.0552 4.9530 6.0496 7.3891 9.0250];
c=1;
for i=1:6
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:5
    d2(c)=d1(i+1)-d1(i);
    c=c+1;
end
c=1;
for i=1:4
    d3(c)=d2(i+1)-d2(i);
```

```
    c=c+1;
end
c=1;
for i=1:3
    d4(c)=d3(i+1)-d3(i);
    c=c+1;
end
c=1;
for i=1:2
    d5(c)=d4(i+1)-d4(i);
    c=c+1;
end
c=1;
for i=1:1
    d6(c)=d5(i+1)-d5(i);
    c=c+1;
end
x0=1.6//first and second derivative at 1.6
h=0.2;
f1=(((d1(3)+d1(4))/2-(d3(2)+d3(3))/4+(d5(1)+d5(2))
   /60))/h
printf('the first derivative of function at 1.6 is:
   %f\n',f1);
f2=((d2(3)-d4(2)/12)+d6(1)/90)/(h^2);
printf('the second derivative of function at 1.6 is:
   %f\n',f2);
```

**Scilab code Exa 6.4** estimation of errors

```
//example 6.4
//estimation of errors
//page 213
clc;clear;close
x=[1.0 1.2 1.4 1.6 1.8 2.0 2.2];
y=[2.7183 3.3201 4.0552 4.9530 6.0496 7.3891
```

```
    9.0250];
c=1;
for i=1:6
    d1(c)=y(i+1)-y(i);
    c=c+1;
end
c=1;
for i=1:5
    d2(c)=d1(i+1)-d1(i);
    c=c+1;
end
c=1;
for i=1:4
    d3(c)=d2(i+1)-d2(i);
    c=c+1;
end
c=1;
for i=1:3
    d4(c)=d3(i+1)-d3(i);
    c=c+1;
end
c=1;
for i=1:2
    d5(c)=d4(i+1)-d4(i);
    c=c+1;
end
c=1;
for i=1:1
    d6(c)=d5(i+1)-d5(i);
    c=c+1;
end
x0=1.6//first and second derivative at 1.6
h=0.2;
f1=((d1(2)-d2(2)/2+d3(2)/3-d4(2)/4+d5(2)/5)/h);
printf('the first derivative of fuction at 1.2 is:%f\n',f1);
f2=(d2(2)-d3(2)+(11*d4(2))/12-(5*d5(2))/6)/h^2;
printf('the second derivative of fuction at 1.2 is:
```

```
%f\n',f2);
T_error1=((d3(2)+d3(3))/2)/(6*h);//truncation error
e=0.00005;//corrected to 4D values
R_error1=(3*e)/(2*h);
T_error1=T_error1+R_error1;//total error
f11=(d1(3)+d1(4))/(2*h);//using stirling formula first derivative
f22=d2(3)/(h*h);//second derivative
T_error2=d4(2)/(12*h*h);
R_error2=(4*e)/(h*h);
T_error2=T_error2+R_error2;
printf('total error in first derivative is %0.4g:\n',T_error1);
printf('total error in second derivative is %0.4g:',T_error2);
```

**Scilab code Exa 6.5** cubic spline method

```
//cubic spline method
//example 6.5
//page 214
clc;clear;close;
x=[0 %pi/2 %pi];
y=[0 1 0];
M0=0,M2=0;
h=%pi/2;
M1=(6*(y(1)-2*y(2)+y(3))/(h^2)-M0-M2)/4;
deff('y=s1(x)','y=2*((-2*3*x^2)/(%pi^2)+3/2)/%pi');
S1=s1(%pi/4);
disp(S1,'S1(pi/4)=');
deff('y=s2(x)','y=(-24*x)/(%pi^3)');
S2=s2(%pi/4);
disp(S2,'S2(pi/4)=');
```

**Scilab code Exa 6.6** derivative by cubic spline method

```
//derivative by cubic spline method
//example 6.6
//page 216
clc;clear;close;
x=[-2 -1 2 3];
y=[-12 -8 3 5]
deff('y=f(x)','y=x^3/15-3*x^2/20+241*x/60-3.9');
deff('y=s2(x)','y=[((2-x)^3/6)*(14/55)+(x+1)
    ^3/6*(-74/55)]/3+[-8-21/55]*(2-x)/3+[3-(9/6)
    *(-74/55)]*(x+1)/3');
h=0.0001;
x0=1.0;
y1=(s2(x0+h)-s2(x0))/h;
printf(' the value y1(%0.2f) is : %f',x0,y1);
```

**Scilab code Exa 6.7** maximun and minimun of functions

```
//maximun and minimun of functions
//example 6.7
//page 218
clc;clear;close;
x=[1.2 1.3 1.4 1.5 1.6];
y=[0.9320 0.9636 0.9855 0.9975 0.9996];
for i=1:4
    d1(i)=y(i+1)-y(i);
end
for i=1:3
    d2(i)=d1(i+1)-d1(i);
end
p=(-d1(1)*2/d2(1)+1)/2;
```

```
disp(p,'p=');
h=0.1;
x0=1.2;
X=x0+p*h;
printf(' the value of X correct to 2 decimal places
    is : %0.2f',X);
Y=y(5)-0.2*d1(4)+(-0.2)*(-0.2+1)*d2(3)/2;
disp(Y,'the value Y=');
```

**Scilab code Exa 6.8** trapezoidal method for integration

```
//example 6.8
//trapezoidal method for integration
//page 226
clc;clear;close;
x=[7.47 7.48 7.49 7.0 7.51 7.52];
f_x=[1.93 1.95 1.98 2.01 2.03 2.06];
h=x(2)-x(1);
l=length(x);
area=0;
for i=1:l
    if i==1|i==l then
        area=area+f_x(i)
    else
        area=area+2*f_x(i)
    end
end
area=area*(h/2);
printf('area bounded by the curve is %f',area);
```

**Scilab code Exa 6.9** simpson 1by3 method for integration

```
//example 6.8
```

```
//simpson 1/3rd  method for integration
//page 226
clc;clear;close;
x=[0.00 0.25 0.50 0.75 1.00];
y=[1.000 0.9896 0.9589 0.9089 0.8415];
y=y^2;
h=x(2)-x(1);
l=length(x);
area=0;
for i=1:l
    if i==1|i==l then
       area=area+y(i)

  elseif (modulo(i,2))==0 then
           area=area+4*y(i)

   elseif (modulo(i,2))~=0 then
          area=area+2*y(i)
           end
    end
area=area*(h*%pi)/3;
printf('area bounded by the curve is %f',area);
```

**Scilab code Exa 6.10** integration by trapezoidal and simpsons method

```
//example 6.10
//integration by trapezoidal and simpson's method
//page 228
clc;clear;close
deff('y=f(x)','y=1/(1+x)');
h=0.5;
x=0:h:1;
l=length(x);
for i=1:l
    y(i)=f(x(i));
```

```
end
area=0;//trapezoidal method
for i=1:l
    if i==1|i==l then
       area=area+y(i)
   else
       area=area+2*y(i)
    end
end
area=area*(h/2);
printf('area bounded by the curve by trapezoidal method with h=%f is %f\n \n',h,area);
area=0;//simpson 1/3rd rule
for i=1:l
    if i==1|i==l then
       area=area+y(i)

  elseif (modulo(i,2))==0 then
           area=area+4*y(i)

   elseif (modulo(i,2))~=0 then
          area=area+2*y(i)
           end
    end
area=(area*h)/3;
printf('area bounded by the curve by simpson 1/3rd method with h=%f is %f\n \n',h,area);
h=0.25;
x=0:h:1;
l=length(x);
for i=1:l
    y(i)=f(x(i));
end
area=0;//trapezoidal method
for i=1:l
    if i==1|i==l then
       area=area+y(i)
   else
```

```
        area=area+2*y(i)
    end
end
area=area*(h/2);
printf('area bounded by the curve by trapezoidal method with h=%f is %f\n \n',h,area);
area=0;//simpson 1/3rd rule
for i=1:l
    if i==1|i==l then
        area=area+y(i)

  elseif (modulo(i,2))==0 then
            area=area+4*y(i)

   elseif (modulo(i,2))~=0 then
           area=area+2*y(i)
            end
    end
area=(area*h)/3;
printf('area bounded by the curve by simpson 1/3rd method with h=%f is %f\n \n',h,area);
h=0.125;
x=0:h:1;
l=length(x);
for i=1:l
    y(i)=f(x(i));
end
area=0;//trapezoidal method
for i=1:l
    if i==1|i==l then
        area=area+y(i)
   else
        area=area+2*y(i)
    end
end
area=area*(h/2);
printf('area bounded by the curve by trapezoidal method with h=%f is %f\n \n',h,area);
```

```
area=0;//simpson 1/3rd rule
for i=1:l
    if i==1|i==l then
       area=area+y(i)

  elseif (modulo(i,2))==0 then
           area=area+4*y(i)

   elseif (modulo(i,2))~=0 then
          area=area+2*y(i)
           end
    end
area=(area*h)/3;
printf('area bounded by the curve by simpson 1/3rd
   method with h=%f is %f\n \n',h,area);
```

**Scilab code Exa 6.11** rommbergs method

```
//example 6.11
//rommberg's method
//page 229
clc;clear;close;
deff('y=f(x)','y=1/(1+x)');
k=1;
h=0.5;
x=0:h:1;
l=length(x);
for i=1:l
    y(i)=f(x(i));
end
area=0;//trapezoidal method
for i=1:l
    if i==1|i==l then
       area=area+y(i)
   else
```

```
        area=area+2*y(i)
    end
end
area=area*(h/2);
I(k)=area;
k=k+1;
h=0.25;
x=0:h:1;
l=length(x);
for i=1:l
    y(i)=f(x(i));
end
area=0;//trapezoidal method
for i=1:l
    if i==1|i==l then
        area=area+y(i)
   else
        area=area+2*y(i)
    end
end
area=area*(h/2);
I(k)=area;
k=k+1;
h=0.125;
x=0:h:1;
l=length(x);
for i=1:l
    y(i)=f(x(i));
end
area=0;//trapezoidal method
for i=1:l
    if i==1|i==l then
        area=area+y(i)
   else
        area=area+2*y(i)
    end
end
area=area*(h/2);
```

```
I(k)=area;
printf('results obtained with h=0.5 0.25 0.125 is %f %f %f\n \n',I(1),I(2),I(3));
for i=1:2
    I1(i)=I(i+1)+(I(i+1)-I(i))/3
end
for i=1:1
    T2(i)=I1(i+1)+(I1(i+1)-I1(i))/3
end
printf('the area is %f',T2(1))
```

**Scilab code Exa 6.12** Trapezoidal and Simpsons rule

```
//example 6.12
//Trapezoidal and Simpson's rule
//page 230
clc;clear;close;
deff('y=f(x)','y=sqrt(1-x^2)');
 k=10:10:50
 for i=1:length(k)
     T_area(i)=0,S_area(i)=0;
   h=1/k(i);
    x=0:h:1
    l=length(x);
    for j=1:l
        y(j)=f(x(j));
    end
    for j=1:l
    if j==1|j==l then
       T_area(i)=T_area(i)+y(j)
   else
       T_area(i)=T_area(i)+2*y(j)
    end

end
```

```
T_area(i)=T_area(i)*(h/2);
for j=1:l
    if j==1|j==l then
       S_area(i)=S_area(i)+y(j)

  elseif (modulo(j,2))==0 then
           S_area(i)=S_area(i)+4*y(j)

   elseif (modulo(i,2))~=0 then
          S_area(i)=S_area(i)+2*y(j)
           end
    end
S_area(i)=S_area(i)*(h)/3;
end
printf(' no of subintervals      Trapezoidal Rule      Simpsons Rule\t \n \n')
for i=1:length(k)
printf(' %0.9g                          %0.9g                  %0.9g             \n ',k(i),T_area(i),S_area(i));

end
```

**Scilab code Exa 6.13** area using cubic spline method

```
//area using cubic spline method
//example 6.2
//page 230
clc;clear;close;
x=[0 0.5 1.0];
y=[0 1.0 0.0]
h=0.5;
M0=0,M2=0;
M1=(6*(y(3)-2*y(2)+y(1))/h^2-M0-M2)/4;
M=[M0 M1 M2];
```

```
I=0;
for i=1:2
    I=I+(h*(y(i)+y(i+1)))/2-((h^3)*(M(i)+M(i+1))/24);
end
printf('the value of the integrand is : %f',I);
```

**Scilab code Exa 6.15** eulers maclaurin formula

```
//euler's maclaurin formula
//example 6.15
//page 233
clc;clear;close;
y=[0 1 0];
h=%pi/4;
I=h*(y(1)+2*y(2)+y(3))/2+(h^2)/12+(h^4)/720;
printf('the value of integrand with h=%f is : %f\n\n',h,I)
h=%pi/8;
y=[0 sin(%pi/8) sin(%pi*2/8) sin(%pi*3/8) sin(%pi*4/8)]
I=h*(y(1)+2*y(2)+2*y(3)+2*y(4)+y(5))/2+(h^2)/2+(h^2)/12+(h^4)/720;
printf('the value of integrand with h=%f is : %f',h,I)
```

**Scilab code Exa 6.17** error estimate in evaluation of the integral

```
// example 6.17
// error estimate in evaluation of the integral
// page 236
clc;clear;close;
deff('z=f(a,b)','z=cos(a)+4*cos((a+b)/2)+cos(b)')
```

```
a=0,b=%pi/2,c=%pi/4;
I(1)=(f(a,b)*((b-a)/2)/3)
I(2)=(f(a,c)*((c-a)/2)/3)
I(3)=(f(c,b)*((b-c)/2)/3)
Area=I(2)+I(3);
Error_estimate=((I(1)-I(2)-I(3))/15);
Actual_area=integrate('cos(x)','x',0,%pi/2);
Actual_error=abs(Actual_area-Area);
printf('the calculated area obtained is:%f\n',Area)
printf('the actual area obtained is:%f\n',
    Actual_area)
printf('the actual error obtained is:%f\n',
    Actual_error)
```

**Scilab code Exa 6.18** error estimate in evaluation of the integral

```
// example 6.18
// error estimate in evaluation of the integral
// page 237
clc;clear;close;
deff('z=f(a,b)','z=8+4*sin(a)+4*(8+4*sin((a+b)/2))
    +8+4*sin(b)')
a=0,b=%pi/2,c=%pi/4;
I(1)=(f(a,b)*((b-a)/2)/3)
I(2)=(f(a,c)*((c-a)/2)/3)
I(3)=(f(c,b)*((b-c)/2)/3)
Area=I(2)+I(3);
Error_estimate=((I(1)-I(2)-I(3))/15);
Actual_area=integrate('8+4*cos(x)','x',0,%pi/2);
Actual_error=abs(Actual_area-Area);
printf('the calculated area obtained is:%f\n',Area)
printf('the actual area obtained is:%f\n',
    Actual_area)
printf('the actual error obtained is:%f\n',
    Actual_error)
```

**Scilab code Exa 6.19** gauss formula

```
//gauss' formula
//example 6.19
//page 242
clc;clear;close;
u=[-0.86113 -0.33998 0.33998 0.86113];
W=[0.34785 0.65214 0.65214 0.34785];
I=0;
for i=1:4
    I=I+(u(i)+1)*W(i);
end
I=I/4;
printf('the value of integrand is : %0.5f',I);
```

**Scilab code Exa 6.20** double integration

```
//example 6.20
//double integration
//page 247
clc;clear;close;
deff('z=f(x,y)','z=exp(x+y)');
h0=0.5,k0=0.5;
h=[0 0.5 1];,k=[0 0.5 1];
for i=1:3
    for j=1:3
      x(i,j)=f(h(i),k(j));
    end
end
T_area=h0*k0*(x(1,1)+4*x(1,2)+4*x(3,2)+6*x(1,3)+x
   (3,3))/4//trapezoidal method
```

```
printf('the integration value by trapezoidal method
    is %f\n ',T_area);
S_area=h0*k0*((x(1,1)+x(1,3)+x(3,1)+x(3,3)+4*(x(1,2)
   +x(3,2)+x(2,3)+x(2,1))+16*x(2,2)))/9
printf('the integration value by Simpson method  is
   %f',S_area);
```

# Chapter 7

# Numerical linear algebra

**Scilab code Exa 7.1** inverse of matrix

```
//example 7.1
//inverse of matrix
//page 256
clc;clear;close;
A=[1,2,3;0,1,2;0,0,1];
A_1=1/A//inverse of matrix
for i=1:3
    for j=1:3
        printf('%d  ',A_1(i,j))
    end
    printf('\n')
end
```

---

**Scilab code Exa 7.2** Factorize by triangulation method

```
//example 7.2
//Factorize by triangulation method
//page 259
```

```
clc;clear;close;
A=[2,3,1;1,2,3;3,1,2];
L(1,2)=0,L(1,3)=0,L(2,3)=0;
U(2,1)=0,U(3,1)=0,U(3,2)=0;
for i=1:3
    L(i,i)=1;
end
for i=1:3
    U(1,i)=A(1,i);
end
L(2,1)=1/U(1,1);
for i=2:3
    U(2,i)=A(2,i)-U(1,i)*L(2,1);
end
L(3,1)=A(3,1)/U(1,1);
L(3,2)=(A(3,2)-U(1,2)*L(3,1))/U(2,2);
U(3,3)=A(3,3)-U(1,3)*L(3,1)-U(2,3)*L(3,2);
printf('The Matrix A in Triangle form\n \n')
printf('Matrix L\n');
for i=1:3
    for j=1:3
        printf('%.2f   ',L(i,j));
    end
    printf('\n');
end
printf('\n \n');
printf('Matrix U\n');
for i=1:3
    for j=1:3
        printf('%.2f   ',U(i,j));
    end
    printf('\n');
end
```

**Scilab code Exa 7.3** Vector Norms

```
//example 7.3
//Vector Norms
//page 262
clc;clear;close;
A=[1,2,3;4,5,6;7,8,9];
s=0;
for i=1:3
    for j=1:3
        s=s+A(j,i);
    end
    C(i)=s;
    s=0;
end
printf('||A||1=%d\n',max(C));
for i=1:3
    for j=1:3
        s=s+(A(i,j)*A(i,j))
    end
end
printf('||A||e=%.3f\n',sqrt(s));
s=0;
for i=1:3
    for j=1:3
        s=s+A(i,j);
    end
    C(i)=s;
    s=0;
end
printf('||A||~=%d\n',max(C));
```

**Scilab code Exa 7.6** Gauss Jordan

```
//example 7.4
//Gauss Jordan
//page 266
```

```
clc;clear;close;
A=[2,1,1,10;3,2,3,18;1,4,9,16];//augmented matrix
for i=1:3
    j=i
    while(A(i,i)==0&j<=3)

for k=1:4
    B(1,k)=A(j+1,k)
    A(j+1,k)=A(i,k)
    A(i,k)=B(1,k)
end
disp(A);
j=j+1;
end
disp(A);
for k=4:-1:i
    A(i,k)=A(i,k)/A(i,i)
end
disp(A)
for k=1:3
    if(k~=i) then
        l=A(k,i)/A(i,i)
        for m=i:4
            A(k,m)=A(k,m)-l*A(i,m)
        end
    end
end
disp(A)
end
for i=1:3
    printf('\nx(%i )=%g\n',i,A(i,4))
end
```

**Scilab code Exa 7.7** modern gauss jordan method

```
//modern gauss jordan method
//example 7.7
//page 269
clc;clear;close;
A=[2 1 1;3 2 3;1 4 9];
I=eye(3,3);
I1=[1;0;0];
I2=[0;1;0];
I3=[0;0;1];
A1=A^-1*I1;
A2=A^-1*I2;
A3=A^-1*I3;
for i=1:3
    AI(i,1)=A1(i,1)
end
for i=1:3
    AI(i,2)=A2(i,1)
end
for i=1:3
    AI(i,3)=A3(i,1)
end
printf('the inverse of the matrix\n')
for i=1:3
    for j=1:3
        printf('%0.2g      ',AI(i,j))
    end
    printf('\n');
    end
```

**Scilab code Exa 7.8** LU decomposition method

```
//LU decomposition method
//example 7.8
//page 273
clc;clear;close;
```

```
A=[2,3,1;1,2,3;3,1,2];
B=[9;6;8]
L(1,2)=0,L(1,3)=0,L(2,3)=0;
U(2,1)=0,U(3,1)=0,U(3,2)=0;
for i=1:3
    L(i,i)=1;
end
for i=1:3
    U(1,i)=A(1,i);
end
L(2,1)=1/U(1,1);
for i=2:3
    U(2,i)=A(2,i)-U(1,i)*L(2,1);
end
L(3,1)=A(3,1)/U(1,1);
L(3,2)=(A(3,2)-U(1,2)*L(3,1))/U(2,2);
U(3,3)=A(3,3)-U(1,3)*L(3,1)-U(2,3)*L(3,2);
printf('The Matrix A in Triangle form\n \n')
printf('Matrix L\n');
for i=1:3
    for j=1:3
        printf('%.2f   ',L(i,j));
    end
    printf('\n');
end
printf('\n \n');
printf('Matrix U\n');
for i=1:3
    for j=1:3
        printf('%.2f   ',U(i,j));
    end
    printf('\n');
end
Y=L^-1*B;
X=U^-1*Y;
printf(' the values of x=%f,y=%f,z=%f',X(1,1),X(2,1)
    ,X(3,1));
```

**Scilab code Exa 7.9** ill conditioned linear systems

```
//ill conditioned linear systems
//example 7.9
//page 276
clc;clear;close;
A=[2 1;2 1.01];
B=[2;2.01];
X=A^-1*B;
A_e=0;
for i=1:2
    for j=1:2
        A_e=A_e+A(i,j)^2;
    end
end
A_e=sqrt(A_e);
inv_A=A^-1;
invA_e=0;
for i=1:2
    for j=1:2
        invA_e=invA_e+inv_A(i,j)^2;
    end
end
invA_e=sqrt(invA_e);
C=A_e*invA_e
de_A=determ(A);
for i=1:2
    s=0;
    for j=1:2
        s=s+A(i,j)^2
    end
    s=sqrt(s);
    k=de_A/s;
end
```

```
if k<1 then
    printf(' the fuction is ill conditioned ')
end
```

**Scilab code Exa 7.10** ill condiioned linear systems

```
//ill condiioned linear systems
//example 7.10
//page 277
clc;clear;close;
A=[1/2 1/3 1/4;1/5 1/6 1/7;1/8 1/9 1/10]//hilbert's
    matrix
de_A=det(A);
if de_A<1 then
    printf(' A is ill-conditioned ')
end
```

**Scilab code Exa 7.11** ill conditioned linear systems

```
//ill conditioned linear system
//example 7.11
//page 277
clc;clear;close;
A=[25 24 10;66 78 37;92 -73 -80];
de_A=det(A);
for i=1:3
    s=0;
    for j=1:3
        s=s+A(i,j)^2
    end
    s=sqrt(s);
    k=de_A/s;
end
```

```
if k<1 then
    printf('the fuction is ill conditioned')
end
```

**Scilab code Exa 7.12** ill conditioned system

```
//ill-conditioned system
//example 7.12
//page 278
clc;clear;close;
//the original equations are 2x+y=2    2x+1.01y=2.01
A1=[2 1;2 1.01];
C1=[2;2.01];
x1=1;y1=1//approximate values
A2=[2 1;2 1.01];
C2=[3;3.01];
C=C1-C2;
X=A1^-1*C;
x=X(1,1)+x1;
y=X(2,1)+y1;
printf('the exact solution is X=%f \t  Y=%f',x,y);
```

**Scilab code Exa 7.14** solution of equations by iteration method

```
//solution of equations by iteration method
//example 7.14
//page 282
//jacobi's method
clc;clear;close;
C=[3.333;1.5;1.4];
X=[3.333;1.5;1.4];
B=[0 -0.1667 -0.1667;-0.25 0 0.25;-0.2 0.2 0];
for i=1:10
```

```
    X1=C+B*X;
    printf('X%d',i);
    for k=1:3
        for l=1:1
            printf(' %f ',X1(k,l))
        end
        printf('\n');
    end
    X=X1;
end
printf(' the solution of the equation is converging at 3  1  1\n\n');
//gauss-seidel method
C=[3.333;1.5;1.4];
X=[3.333;1.5;1.4];
B=[0 -0.1667 -0.1667;-0.25 0 0.25;-0.2 0.2 0];
X1=C+B*X;
x=X1(1,1);y=X1(2,1);z=X1(3,1);
for i=1:5
    x=3.333-0.1667*y-0.1667*z
    y=1.5-0.25*x+0.25*z
    z=1.4-0.2*x+0.2*y
    printf(' the value after %d iteration is : %f\t %f\t %f\t\n\n',i,x,y,z)
end
printf(' again we conclude that roots converges at 3  1  1')
```

**Scilab code Exa 7.15** eigenvalues and eigenvectors

```
//eigenvalues and eigenvectors
//example 7.15
//page 285
clc;clear;close
A=[5 0 1;0 -2 0;1 0 5];
```

```
x=poly(0,'x');
for i=1:3
    A(i,i)=A(i,i)-x;
end
d=determ(A);
X=roots(d);
printf(' the eigen values are \n\n')
disp(X);
X1=[0;1;0]
X2=[1/sqrt(2);0;-1/sqrt(2)];
X3=[1/sqrt(2);0;1/sqrt(2)];
//after computation the eigen vectors
printf('the eigen vectors for value %0.2g is',X(3));
disp(X1);
printf('the eigen vectors for value %0.2g is',X(2));
disp(X2);
printf('the eigen vectors for value %0.2g is',X(1));
disp(X3);
```

**Scilab code Exa 7.16** largest eigenvalue and eigenvectors

```
//largest eigenvalue and eigenvectors
//example 7.16
//page 286
clc;clear;close;
A=[1 6 1;1 2 0;0 0 3];
I=[1;0;0];//initial eigen vector
X0=A*I
disp(X0,'X0=')
X1=A*X0;
disp(X1,'X1=')
X2=A*X1;
disp(X2,'X2=')
X3=X2/3;
disp(X3,'X3=')
```

```
X4=A*X3;
X5=X4/4;
disp(X5,'X5=');
X6=A*X5;
X7=X6/(4*4);
disp(X7,'X7=');
printf('as it can be seen that highest eigen value
    is 4 \n\n the eigen vector is %d %d %d',X7(1),X7
    (2),X7(3));
```

**Scilab code Exa 7.17** housrholders method

```
//housrholder's method
//example 7.17
//page 290
clc;clear;close;
A=[1 3 4;3 2 -1;4 -1 1];
S=sqrt(A(1,2)^2+A(1,3)^2);
v2=sqrt([1+(A(1,2)/S)]/2)
v3=A(1,3)/(2*v2*S)
V=[0 v2 v3];
P1=[1 0 0;0 1-2*v2^2 -2*v2*v3;0 -2*v2*v3 1-2*v3^2];
A1=P1*A*P1;
printf(' the reduced matrix is \n\n');
for i=1:3
    for j=1:3
        printf('%0.2f    ',A1(i,j));
    end
    printf('\n');
end
```

**Scilab code Exa 7.18** single value decommposition

```
//single value decommposition
//example 7.18
//page 292
clc;clear;close;
A=[1 2;1 1;1 3];
A1=A'*A;
x=poly(0,'x');
A1(1,1)=A1(1,1)-x;
A1(2,2)=A1(2,2)-x;
de_A1=det(A1);
C=roots(de_A1);
printf( ' eigen values are %0.2f  %0.2f\n\n',C(1),C(2));
X1=[0.4033;0.9166];
X2=[0.9166;-0.4033];
Y1=(A*X1)/sqrt(C(1));
Y2=(A*X2)/sqrt(C(2));
printf(' singular decomposition of A is given by \n\n');
D1=[Y1(1) Y2(1);Y1(2) Y2(2);Y1(3) Y2(3)];
D2=[sqrt(C(1)) 0;0,sqrt(C(2))];
D3=[X1(1) X2(1);X1(2) X2(2)];
for i=1:3
    for j=1:2
        printf('%0.4f    ',D1(i,j))
    end
    printf('\n')
end
printf('\n\n')
for i=1:2
    for j=1:2
        printf('%0.4f    ',D2(i,j))
    end
    printf('\n')
end
printf('\n\n');
for i=1:2
    for j=1:2
```

```
            printf('%0.4f    ',D3(i,j))
        end
        printf('\n')
end
```

# Chapter 8

# Numerical Solution of ordinary diffrential equation

**Scilab code Exa 8.1** taylors method

```
//example 8.1
//taylor's method
//page 304
clc;clear;close;
f=1;//value of function at 0
deff('z=f1(x)','z=x-f^2');
deff('z=f2(x)','z=1-2*f*f1(x)');
deff('z=f3(x)','z=-2*f*f2(x)-2*f2(x)^2');
deff('z=f4(x)','z=-2*f*f3(x)-6*f1(x)*f2(x)');
deff('z=f5(x)','z=-2*f*f4(x)-8*f1(x)*f3(x)-6*f2(x)^2');
h=0.1;//value at 0.1
k=f;
        for j=1:5
            if j==1  then
                k=k+h*f1(0);
            elseif j==2 then
                k=k+(h^j)*f2(0)/factorial(j)
            elseif j ==3
```

```
                k=k+(h^j)*f3(0)/factorial(j)
            elseif j ==4
                k=k+(h^j)*f4(0)/factorial(j)
            elseif j==5
                k=k+(h^j)*f5(0)/factorial(j)

end
end
printf('the value of the function at %.2f is :%0.4f'
   ,h,k)
```

**Scilab code Exa 8.2** taylors method

```
//taylor's method
//example 8.2
//page 304
clc;clear;close;
f=1;//value of function at 0
f1=0;//value of first derivatie at 0
deff('y=f2(x)','y=x*f1+f')
deff('y=f3(x)','y=x*f2(x)+2*f1');
deff('y=f4(x)','y=x*f3(x)+3*f2(x)');
deff('y=f5(x)','y=x*f4(x)+4*f3(x)');
deff('y=f6(x)','y=x*f5(x)+5*f4(x)');
h=0.1;//value at 0.1
k=f;
        for j=1:6
            if j==1  then
                k=k+h*f1;
            elseif j==2 then
                k=k+(h^j)*f2(0)/factorial(j)
            elseif j ==3
                k=k+(h^j)*f3(0)/factorial(j)
            elseif j ==4
                k=k+(h^j)*f4(0)/factorial(j)
```

```
            elseif j==5
                k=k+(h^j)*f5(0)/factorial(j)
                else
k=k+(h^j)*f6(0)/factorial (j)
end
end
printf('the value of the function at %.2f is :%0.7f'
   ,h,k)
```

**Scilab code Exa 8.3** picards method

```
//example 8.3
//picard's method
//page 306
clc;clear;close;
deff('z=f(x,y)','z=x+y^2')
y(1)=1;
for i=1:2
    y(i+1)=y(1)+integrate('f(x,y(i))','x',0,i /10);
    printf ( ' \n y (%g) = %g\n ' ,i/10 ,y(i +1) );
    end
```

**Scilab code Exa 8.4** picards method

```
//example 8.4
//picard's method
//page 306
clc;clear;close;
deff('z=f(x,y)','z=x^2/(y^2+1)')
y(1)=0;//value at 0
c=0.25;
for i=1:3
    y(i+1)=y(1)+integrate('f(x,y(i))','x',0,c);
```

```
    printf ( ' \n y(%0.2 f ) = %g\n',c ,y(i +1) );
    c=c*2;
    end
```

**Scilab code Exa 8.5** eulers method

```
//example 8.5
//euler 's method
//page 308
clc;clear;close;
deff('z=f(y)','z=-y')
y(1)=1;//value at 0
h=0.01;c=0.01;
for i=1:4
    y(i+1)=y(i)+h*f(y(i))
    printf ('\ny(%g)=%g\n',c,y(i+1));
    c=c+0.01;
end
```

**Scilab code Exa 8.6** error estimates in eulers

```
//example 8.6
//error estimates in euler 's
//page 308
clc;clear;close;
deff('z=f(y)','z=-y')
y(1)=1;//value at 0
h=0.01;c=0.01;
for i=1:4
    y(i+1)=y(i)+h*f(y(i))
    printf ('\ny(%g)=%g\n',c,y(i+1));
    c=c+0.01;
end
```

```
for i=1:4
    L(i)=abs(-(1/2)*h^2*y(i+1));
    printf('L(%d) =%f\n\n',i,L(i))
end
e(1)=0;
for i=1:4
    e(i+1)=abs(y(2)*e(i)+L(1));
    printf('e(%d)=%f\n\n',i,e(i))
end
Actual_value=exp(-0.04);
Estimated_value=y(5);
err=abs(Actual_value-Estimated_value);
if err<e(5) then
    disp(' VERIFIED');
end
```

**Scilab code Exa 8.7** modified eulers method

```
//example 8.7
//modified euler's method
//page 310
clc;clear;close;
h=0.05;
f=1;
deff('z=f1(x,y)','z=x^2+y');
x=0:0.05:0.1
y1=0;
y1(1)=f+h*f1(x(1),f);
y1(2)=f+h*(f1(x(1),f)+f1(x(2),y1(1)))/2;
y1(3)=f+h*(f1(x(1),f)+f1(x(3),y1(2)))/2;
y2(1)=y1(2)+h*f1(x(2),y1(2));
y2(2)=y1(2)+h*(f1(x(2),y1(2))+f1(x(3),y2(1)))/2;
y2(3)=y1(2)+h*(f1(x(2),y1(2))+f1(x(3),y2(2)))/2;
printf(' y1(0)\t y1(1)\t y1(2)\t y2(0)\t y2(1)\t y3
    (2)\n\n' );
```

```
printf(' %f\t      %f\t      %f\t      %f\t       %f\t      %f
   \n',y1(1),y1(2),y1(3),y2(1),y2(2),y2(3))
printf('\n\n the value of y at %0.2f is : %0.4f',x
   (3),y2(3));
```

**Scilab code Exa 8.8** runge kutta formula

```
//example 8.8
//runge-kutta formula
//page 313
clc;clear;close;
deff('y=f(x,y)','y=y-x')
y=2;x=0;h=0.1;
K1=h*f(x,y);
K2=h*f(x+h,y+K1);
y1=y+( K1+K2)/2
printf ('\n y(0.1) by second order runge kutta
   method:%0.4f',y1);
y=y1;x=0.1;h=0.1;
K1=h*f(x,y);
K2=h*f(x+h,y+K1);
y1=y+( K1+K2)/2
printf ('\n y(0.2) by second order runge kutta
   method:%0.4f',y1);
y=2,x=0,h=0.1;
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
y1=y+(K1+2*K2+2*K3+K4)/6;
printf ('\n y(0.1) by fourth order runge kutta
   method:%0.4f',y1);
y=y1,x=0.1,h=0.1;
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
```

```
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
y1=y+(K1+2*K2+2*K3+K4)/6;
printf ('\n y(0.1) by fourth order runge kutta
    method:%0.4f',y1);y=2,x=0,h=0.1;
```

**Scilab code Exa 8.9** runge kutta formula

```
//example 8.9
//runge kutta method
//page 315
clc;clear;close;
deff('y=f(x,y)','y=1+y^2');
y=0,x=0,h=0.2;
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
y1=y+(K1+2*K2+2*K3+K4)/6;
printf ('\n y(0.2) by fourth order runge kutta
    method:%0.4f',y1);
y=y1,x=0.2,h=0.2;
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
y1=y+(K1+2*K2+2*K3+K4)/6;
printf ('\n y(0.4) by fourth order runge kutta
    method:%0.4f',y1);
y=2,x=0,h=0.1;
y=y1,x=0.4,h=0.2;
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
```

```
y1=y+(K1+2*K2+2*K3+K4)/6;
printf ('\n y(0.6) by fourth order runge kutta
   method:%0.4f',y1);
```

**Scilab code Exa 8.10** initial value problems

```
//example 8.10
//initial value problems
//page 315
clc;clear;close;
deff('y=f1(x,y)','y=3*x+y/2');
y(1)=1;
h=0.1;c=0;
for i=1:2
    y(i+1)=y(i)+h*f1(c,y(i))
    printf ('\ny(%g)=%g\n',c,y(i))
    c=c+0.1;
end
```

**Scilab code Exa 8.11** adams moulton method

```
//example 8.11
//adam's moulton method
//page  316
clc;clear;close;
deff('y=f(x,y)','y=1+y^2');
y=0,x=0,h=0.2,f1(1)=0;
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
y1=y+(K1+2*K2+2*K3+K4)/6;
f1(1)=y1;
```

```
printf ('\n y(0.2) by fourth order runge kutta method:%0.4f',y1);
y=y1,x=0.2,h=0.2;
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
y1=y+(K1+2*K2+2*K3+K4)/6;
f1(2)=y1;
printf ('\n y(0.4) by fourth order runge kutta method:%0.4f',y1);
y=2,x=0,h=0.1;
y=y1,x=0.4,h=0.2;
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
y1=y+(K1+2*K2+2*K3+K4)/6;
f1(3)=y1;
printf ('\n y(0.6) by fourth order runge kutta method:%0.4f',y1);
y_p=y1+h*(55*(1+f1(3)^2)-59*(1+f1(2)^2)+37*(1+f1(1)^2)-9)/24;
y_c=y1+h*(9*(1+(y_p-1)^2)+19*(1+f1(3)^2)-5*(1+f1(2)^2)+(1+f1(1)^2))/24;
printf(' \nthe predicted value is:%0.4f:\n ',y_p);
printf(' the computed value is:%0.4f:',y_c);
```

**Scilab code Exa 8.12** milnes method

```
//example 8.12
//milne's method
//page 320
clc;clear;close;
deff('y=f(x,y)','y=1+y^2');
```

```
y=0,x=0,h=0.2,f1(1)=0;
printf('x                    y                    y1=1+y^2\n\n')
Y1(1)=1+y^2;
printf('%0.4f          %0.4f          %0.4f\n',x,y,(1+y^2));
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
y1=y+(K1+2*K2+2*K3+K4)/6;
f1(1)=y1;
Y1(2)=1+y1^2;
printf('%0.4f          %0.4f          %0.4f\n',x+h,y1,(1+y1^2));
y=y1,x=0.2,h=0.2;
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
y1=y+(K1+2*K2+2*K3+K4)/6;
f1(2)=y1;
Y1(3)=1+y1^2
printf('%0.4f          %0.4f          %0.4f\n',x+h,y1,(1+y1^2));
y=y1,x=0.4,h=0.2;
K1=h*f(x,y);
K2=h*f(x+h/2,y+K1/2);
K3=h*f(x+h/2,y+K2/2);
K4=h*f(x+h,y+K3);
y1=y+(K1+2*K2+2*K3+K4)/6;
f1(3)=y1;
Y1(4)=1+y1^2;
printf('%0.4f          %0.4f          %0.4f\n',x+h,y1,(1+y1^2));
Y_4=4*h*(2*Y1(2)-Y1(3)+2*Y1(4))/3;
printf('y(0.8)=%f\n',Y_4);
Y=1+Y_4^2;
```

```
Y_4=f1(2)+h*(Y1(3)+4*Y1(4)+Y)/3;//more correct value
printf('y(0.8)=%f\n',Y_4);
```

**Scilab code Exa 8.13** milnes method

```
//example 8.13
//milne's method
//page 320
clc;clear;close;
deff('y=f1(x,y)','y=x^2+y^2-2');
x=[-0.1 0 0.1 0.2 ];
y=[1.0900 1.0 0.8900 0.7605];
h=0.1;
for i=1:4
    Y1(i)=f1(x(i),y(i));
end
printf('    x                    y                    y1=x^2+
    y^2-2      \n\n');
for i=1:4
printf('  %0.2f              %f              %f
    \n',x(i),y(i),Y1(i));
end
Y_3=y(1)+(4*h/3)*(2*Y1(2)-Y1(3)+2*Y1(4));
printf('y(0.3)=%f\n',Y_3)
Y1_3=f1(0.3,Y_3);
Y_3=y(3)+h*(Y1(3)+4*Y1(4)+Y1_3)/3;//corrected value
printf('corrected y(0.3)=%f',Y_3)
```

**Scilab code Exa 8.14** initial value problems

```
//example 8.14
//initial-value problem
//page 322
```

```
clc;clear;close;
deff('y=f(x)','y=13*exp(x/2)-6*x-12');
s1=1.691358;s3=3.430879;
printf('the erorr in the computed values are %0.7g %0.7g',abs(f(0.5)-s1),abs(f(1)-s3))
```

**Scilab code Exa 8.15** boundary value problem using finite difference method

```
//boundary value problem using finite difference method
//example 8.15
//page 328
clc;clear;close;
deff('y=f(x)','y=cos(x)+((1-cos(1))/sin(1))*sin(x)-1');
h1=1/2;
Y=f(0.5);
y0=0,y2=0;
y1=4*(1/4+y0+y2)/7
printf('computed value with h=%f of y(0.5) is %f\n',h1,y1)
printf('error in the result with actual value %f\n',abs(Y-y1) )
h2=1/4;
y0=0,y4=0;
//solving the approximated diffrential equation
A=[-31/16 1 0;1 -31/16 1;0 1 -31/16];
X=[-1/16;-1/16;-1/16];
C=A^-1*X;
printf('computed value with h=%f of y(0.5) is %f\n',h2,C(2,1))
printf('error in the result with actual value %f\n',abs(Y-C(2,1)))
```

**Scilab code Exa 8.16** boundary value problem using finite difference method

```
//boundary value problem using finite difference
    method
//example 8.16
//page 329
clc;clear;close;
deff('y=f(x)','y=sinh(x)')
y0=0//y(0)=0;
y4=3.62686//y(2)=3.62686
h1=0.5;
Y=f(0.5)
//arranging and calculating the values
A=[-9 4 0;4 -9 4;0 4 -9];
C=[0;0;-14.50744];
X=A^-1*C
printf('computed value with h=%f of y(0.5) is %f\n',
    h1,X(1,1))
printf('error in the result with actual value %f\n',
    abs(Y-X(1,1)) )
h2=1.0;
y0=0//y(0)=0;
y2=3.62686//y(2)=3.62686
y1=(y0+y2)/3;
Y=(4*X(2,1)-y1)/3;
printf('with better approximation error is reduced
    to %f',abs(Y-f(1.0)));
```

**Scilab code Exa 8.17** cubic spline method

```
//cubic spline method
//example 8.17
```

```
//page 330
clc;clear;close;
deff('y=f(x)','y=cos(x)+((1-cos(1))/sin(1))*sin(x)-1');
y=[f(0) f(0.5) f(1)];
h=1/2;
Y=f(0.5);
y0=0,y2=0;
M0=-1;M1=-22/25;M2=-1;
s1_0=47/88;s1_1=-47/88;s1_05=0;
printf('the cubic spline value is %f',Y)
```

**Scilab code Exa 8.18** cubic spline method

```
//cubic spline method
//example 8.18
//page 331
clc;clear;close;
//after arranging and forming equation
A=[10 -1 0 24;0 16 -1 -32;1 20 0 16;0 1 26 -24];
C=[36;-12;24;-9];
X=A^-1*C;
printf(' Y1=%f\n\n',X(4,1));
printf(' the error in the solution is :%f',abs((2/3)-X(4,1)))
```

**Scilab code Exa 8.19** boundary value problem by cubic spline method

```
//boundary value problem by cubisc spline nethod
//example 8.18
//page 331
clc;clear;close;
h=1/2;
```

```
//arranging in two subintervals we get
A=[10 -1 0 24;0 16 -1 -32;1 20 0 16;0 1 26 -24];
C=[36;-12;24;-9];
X=A^-1*C
printf('the computed value of y(1.5) is %f ',X(4,1))
    ;
```

# Chapter 9

# Numerical Solution of Partial Diffrential Equation

**Scilab code Exa 9.1** standard five point formula

```
//standard five point formula
//example 9.1
//page 350
clc;clear;close;
u2=5;u3=1;
for i=1:3
    u1=(u2+u3+6)/4;
    u2=u1/2+5/2;
    u3=u1/2+1/2;
    printf(' the values are u1=%d\t u2=%d\t u3=%d\t\
       n\n',u1,u2,u3);
end
```

---

**Scilab code Exa 9.2** solution of laplace equation by jacobi method gauss seidel method and SOR method

```
//solution of laplace equation by jacobi method, gauss-seidel method and SOR method
//example 9.2
//page 351
clc;clear;close;
u1=0.25;u2=0.25;u3=0.5;u4=0.5;//initial values
printf('jacobis iteration process\n\n')
printf('u1\t     u2\t     u3\t     u4\t \n\n')
printf('%f\t     %f\t     %f\t     %f\t  \n',u1,u2,u3,u4)
for i=1:7
    u11=(0+u2+0+u4)/4
    u22=(u1+0+0+u3)/4;
    u33=(1+u2+0+u4)/4;
    u44=(1+0+u3+u1)/4;
    u1=u11;u2=u22;u3=u33;u4=u44;
printf('%f\t     %f\t     %f\t     %f\t  \n',u11,u22,u33,u44)
end
printf(' gauss seidel process\n\n');
u1=0.25;u2=0.3125;u3=0.5625;u4=0.46875;//initial values
printf('u1\t     u2\t     u3\t     u4\t \n\n')
printf('%f\t     %f\t     %f\t     %f\t  \n',u1,u2,u3,u4)
for i=1:4
    u1=(0+u2+0+u4)/4
    u2=(u1+0+0+u3)/4;
    u3=(1+u2+0+u4)/4;
    u4=(1+0+u3+u1)/4;
    printf('%f\t     %f\t     %f\t     %f\t  \n',u1,u2,u3,u4)
end
printf('u1\t     u2\t     u3\t     u4\t \n\n')
printf('%f\t     %f\t     %f\t     %f\t  \n',u1,u2,u3,u4)
```

**Scilab code Exa 9.4** poisson equation

```
//poisson equation
//exaample 9.4
//page 354
clc;clear;close;
u2=0;u4=0;
printf('  u1\t     u2\t     u3\t     u4\t\n\n');
for i=1:6
    u1=u2/2+30;
    u2=(u1+u4+150)/4;
    u4=u2/2+45;
    printf(' %0.2f\t    %0.2f\t    %0.2f\t    %0.2f\n',
       u1,u2,u2,u4);
end
printf(' from last two iterates we conclude u1=67
   u2=75    u3=75    u4=83\n')
```

---

**Scilab code Exa 9.6** bender schmidt formula

```
//bender-schmidt formula
//example 9.6
//page 362
clc;clear;close;
deff('y=f(x)','y=4*x-x^2');
u=[f(0) f(1) f(2) f(3) f(4)];
u11=(u(1)+u(3))/2;
u12=(u(2)+u(4))/2;
u13=(u(3)+u(5))/2;
printf(' u11=%0.2f\t   u12=%0.2f\t   u13=%0.2f\t \n',
   u11,u12,u13)
u21=(u(1)+u12)/2;
```

```
u22=(u11+u13)/2;
u23=(u12+0)/2;
printf(' u21=%0.2f\t   u22=%0.2f\t   u23=%0.2f\t  \n',u21,u22,u23)
u31=(u(1)+u22)/2;
u32=(u21+u23)/2;
u33=(u22+u(1))/2;
printf(' u31=%0.2f\t   u32=%0.2f\t   u33=%0.2f\t  \n',u31,u32,u33)
u41=(u(1)+u32)/2;
u42=(u31+u33)/2;
u43=(u32+u(1))/2;
printf(' u41=%0.2f\t   u42=%0.2f\t   u43=%0.2f\t  \n',u41,u42,u43)
u51=(u(1)+u42)/2;
u52=(u41+u43)/2;
u53=(u42+u(1))/2;
printf(' u51=%0.2f\t   u52=%0.2f\t   u53=%0.2f\t  \n',u51,u52,u53)
```

**Scilab code Exa 9.7** bender schimdts formula and crank nicolson formula

```
//bender-schimdt's formula and crank-nicolson formula
//example 9.7
//page 363
//bender -schimdt's formula
clc;clear;close;
deff('y=f(x,t)','y=exp(-%pi^2*t)*sin(%pi*x)');
u=[f(0,0) f(0.2,0) f(0.4,0) f(0.6,0) f(0.8,0) f(1,0)];
u11=u(3)/2;u12=(u(2)+u(4))/2;u13=u12;u14=u11;
printf(' u11=%f\t   u12=%f\t   u13=%f\t   u14=%f\n\n',u11,u12,u13,u14)
u21=u12/2;u22=(u12+u14)/2;u23=u22;u24=u21;
```

```
printf(' u21=%f\t  u22=%f\t  u23=%f\t  u24=%f\n\n',
   u21,u22,u23,u24)
printf(' the error in the solution is: %f\n\n',abs(
   u22-f(0.6,0.04)))
//crank-nicolson formula
//by putting i=1,2,3,4 we obtain four equation
A=[4 -1 0 0 ;-1 4 -1 0;0 -1 4 -1;0 0 -1 4];
C=[0.9510;1.5388;1.5388;0.9510];
X=A^-1*C;
printf( ' u11=%f\t  u21=%f\t  u31=%f\t  u41=%f\t\n\n
   ',X(1,1),X(2,1),X(3,1),X(4,1))
printf(' the error in the solution is: %f\n\n',abs(X
   (2,1)-f(0.6,0.04)))
```

**Scilab code Exa 9.8** heat equation using crank nicolson method

```
//heat equation using crank-nicolson method
//example 9.8
//page 364
clc;clear;close;
U=0.01878;
//h=1/2;l=1/8,i=1;
u01=0;u21=1/8;
u11=(u21+u01)/6;
printf(' u11=%f\n\n',u11);
printf('error is %f\n\n',abs(u11-U));
//h=1/4,l=1/8,i=1,2,3
A=[-3 -1 0;1 -3 1;0 1 -3];
C=[0;0;-1/8];
X=A^-1*C;
printf(' u12=%f\n\n',X(2,1));
printf('error is %f\n\n',abs(X(2,1)-U));
```