Lupascu Felicia

Homework 2

Ex.1

Problem 12

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

The factor of $\pi/2$ guarantees that $\lim_{x\to\infty} erf(x) = 1$

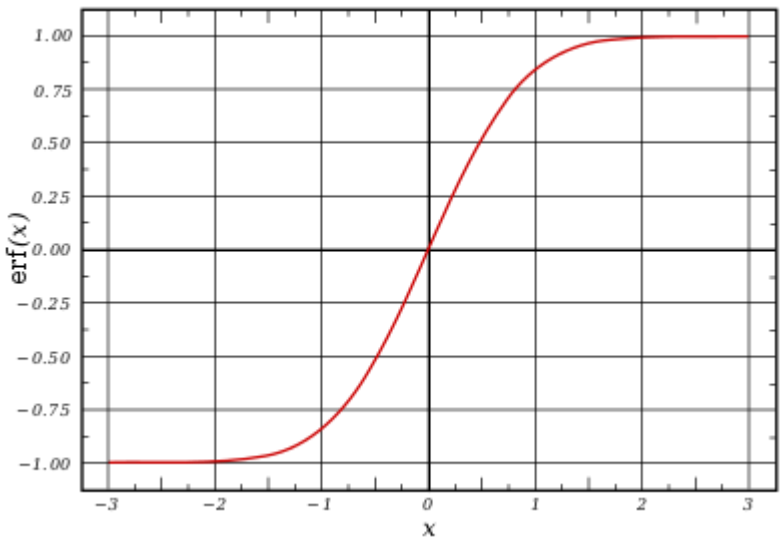
We can integrate power series $\Rightarrow e^{-t^2} = 1 - t^2 + \frac{t^4}{2!} - \frac{t^6}{3!} + \dots \quad (R = \infty)$

we can now find a power series expansion for the error function

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt = \frac{2}{\sqrt{\pi}} \int_0^x \left(1 - t^2 + \frac{t^4}{2!} - \frac{t^6}{3!} + \dots\right) dt =$$

$$= \frac{2}{\sqrt{\pi}} \left[t - \frac{t^3}{3} + \frac{t^5}{5 \cdot 2!} - \frac{t^7}{7 \cdot 3!} + \dots\right]_0^x = \frac{2}{\sqrt{\pi}} \left(x - \frac{x^3}{3} + \frac{x^5}{5 \cdot 2!} - \frac{x^7}{7 \cdot 3!} + \dots\right)$$

We have to find such n, so the difference between I_n and $erf(x)$ should be less than 10^{-5}



Lupascu Felicia

Ex.2

main.py

```
memoized = {0: 1, 1: 1}

def fib(n):
  if (n in memoized):
    return(memoized[n])
  else:
    memoized[n] = fib(n-1) + fib(n-2)
    return(memoized[n])

for i in range(40):
  print(i+1,fib(i))
```

Program that generates Fibonacci numbers

main.py

```
memoized = {0: 1, 1: 1}

def fib(n):
  if (n in memoized):
    return(memoized[n])
  else:
    memoized[n] = fib(n-1) + fib(n-2)
    print("  Golden ratio=",fib(n-1)/fib(n-2))
    return(memoized[n])

for i in range(40):
  print(i+1,fib(i))
```

The ratio of successive Fibonacci numbers approaches the golden ratio. Here's a program to explore this connection.

```
1 1
2 1
3 2
4 3
5 5
6 8
7 13
8 21
9 34
10 55
11 89
12 144
13 233
14 377
15 610
16 987
17 1597
18 2584
19 4181
20 6765
21 10946
22 17711
23 28657
24 46368
25 75025
26 121393
27 196418
28 317811
29 514229
30 832040
31 1346269
32 2178309
33 3524578
34 5702887
35 9227465
36 14930352
37 24157817
38 39088169
39 63245986
40 102334155
```

```
1 1
2 1
    Golden ratio= 1.0
3 2
    Golden ratio= 2.0
4 3
    Golden ratio= 1.5
5 5
    Golden ratio= 1.6666666666666667
6 8
Golden ratio= 1.6
7 13
    Golden ratio= 1.625
8 21
    Golden ratio= 1.6153846153846154
9 34
    Golden ratio= 1.619047619047619
10 55
    Golden ratio= 1.6176470588235294
11 89
    Golden ratio= 1.6181818181818182
12 144
    Golden ratio= 1.6179775280898876
13 233
    Golden ratio= 1.6180555555555556
14 377
    Golden ratio= 1.6180257510729614
15 610
    Golden ratio= 1.6180371352785146
16 987
    Golden ratio= 1.618032786885246
17 1597
    Golden ratio= 1.618034447821682
18 2584
    Golden ratio= 1.6180338134001253
19 4181
    Golden ratio= 1.618034055727554
20 6765
    Golden ratio= 1.6180339631667064
21 10946
    Golden ratio= 1.6180339985218033
22 17711
    Golden ratio= 1.618033985017358
```

```
23 28657
    Golden ratio= 1.6180339901755971
24 46368
    Golden ratio= 1.618033988205325
25 75025
    Golden ratio= 1.618033988957902
26 121393
    Golden ratio= 1.6180339886704431
27 196418
    Golden ratio= 1.6180339887802426
28 317811
    Golden ratio= 1.618033988738303
29 514229
    Golden ratio= 1.6180339887543225
30 832040
    Golden ratio= 1.6180339887482036
31 1346269
    Golden ratio= 1.6180339887505408
32 2178309
    Golden ratio= 1.6180339887496482
33 3524578
    Golden ratio= 1.618033988749989
34 5702887
    Golden ratio= 1.618033988749859
35 9227465
    Golden ratio= 1.6180339887499087
36 14930352
    Golden ratio= 1.6180339887498896
37 24157817
    Golden ratio= 1.618033988749897
38 39088169
    Golden ratio= 1.618033988749894
39 63245986
    Golden ratio= 1.6180339887498951
40 102334155
```

Thus we have found that the ratio of successive terms of a Fibonacci sequence a_{n+1}/a_n, which is equal to b_n/a_n, converges to the Golden Ratio.

Lupascu Felicia

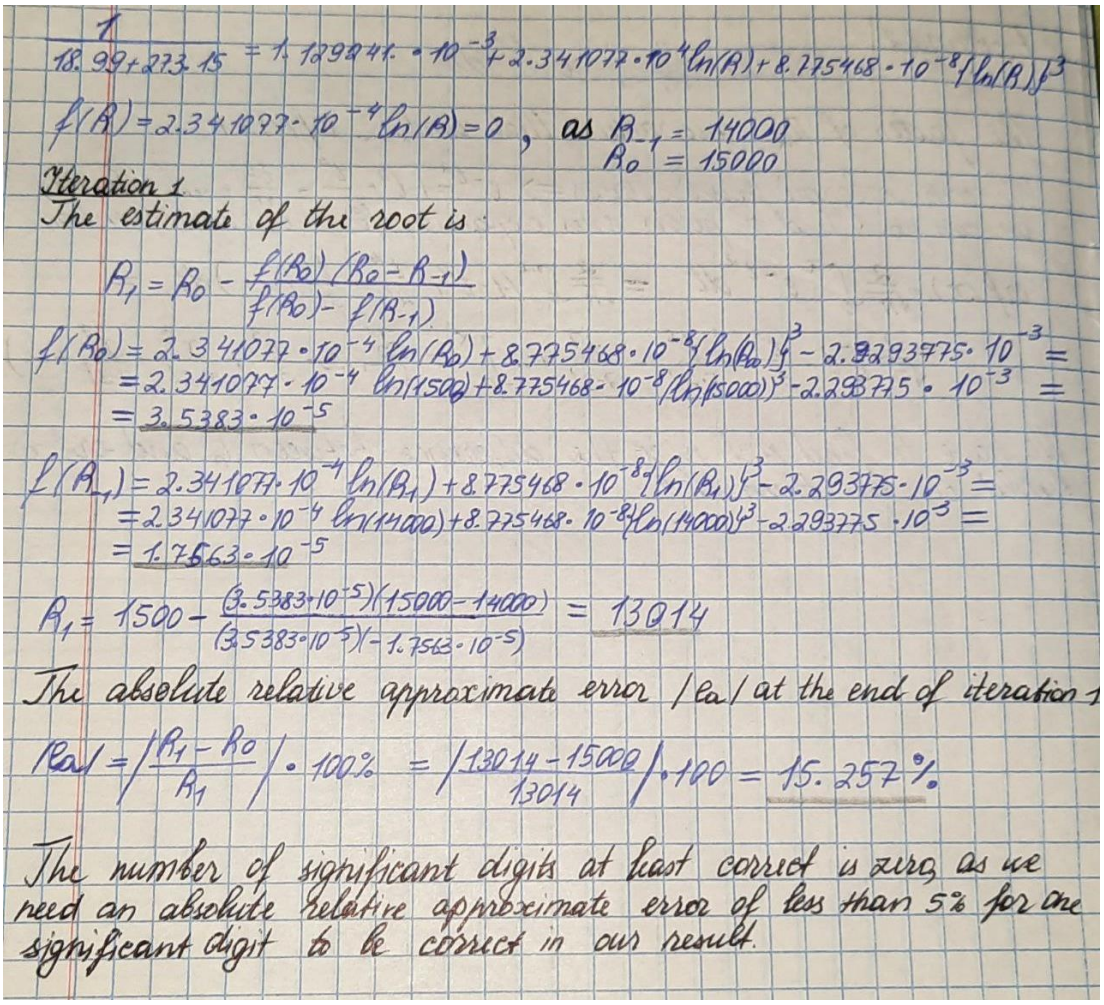
Ex.3

the resistance R of the thermistor and the temperature T is given by

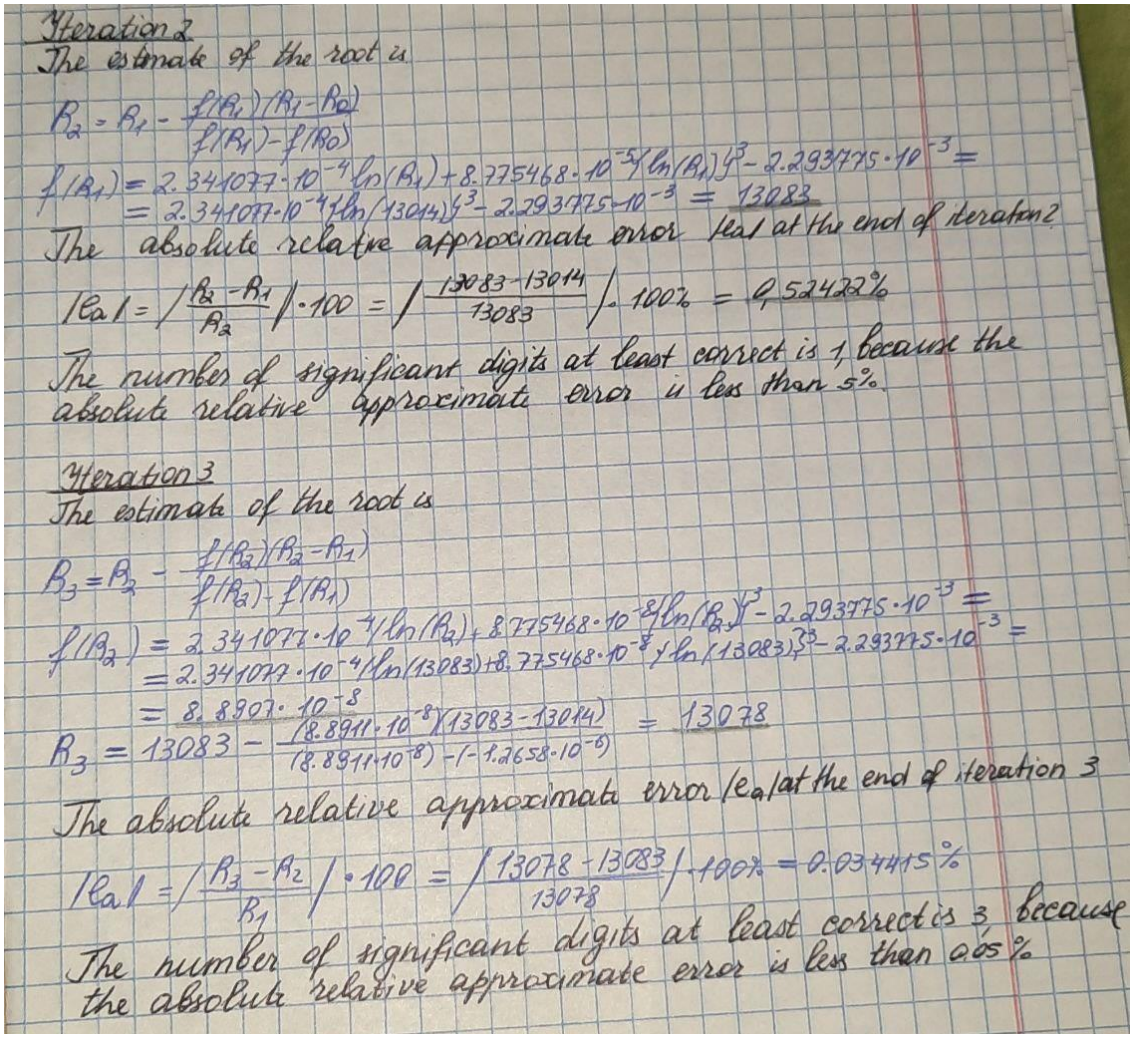
$$\frac{1}{T} = 1.129241 \times 10^{-3} + 2.341077 \times 10^{-4} \log R + 8.775468 \times *10^{-8} \Big(\log R \Big)^3$$

$$\frac{1}{19.01 + 273.15} = 1.129241 \times 10^{-3} + 2.341077 \times 10^{-4} \log R + 8.775468 \times *10^{-8} \Big(\log R \Big)^3 \quad (1)$$

$$\frac{1}{18.99 + 273.15} = 1.129241 \times 10^{-3} + 2.341077 \times 10^{-4} \log R + 8.775468 \times *10^{-8} \Big(\log R \Big)^3 \quad (2)$$



Lupascu Felicia

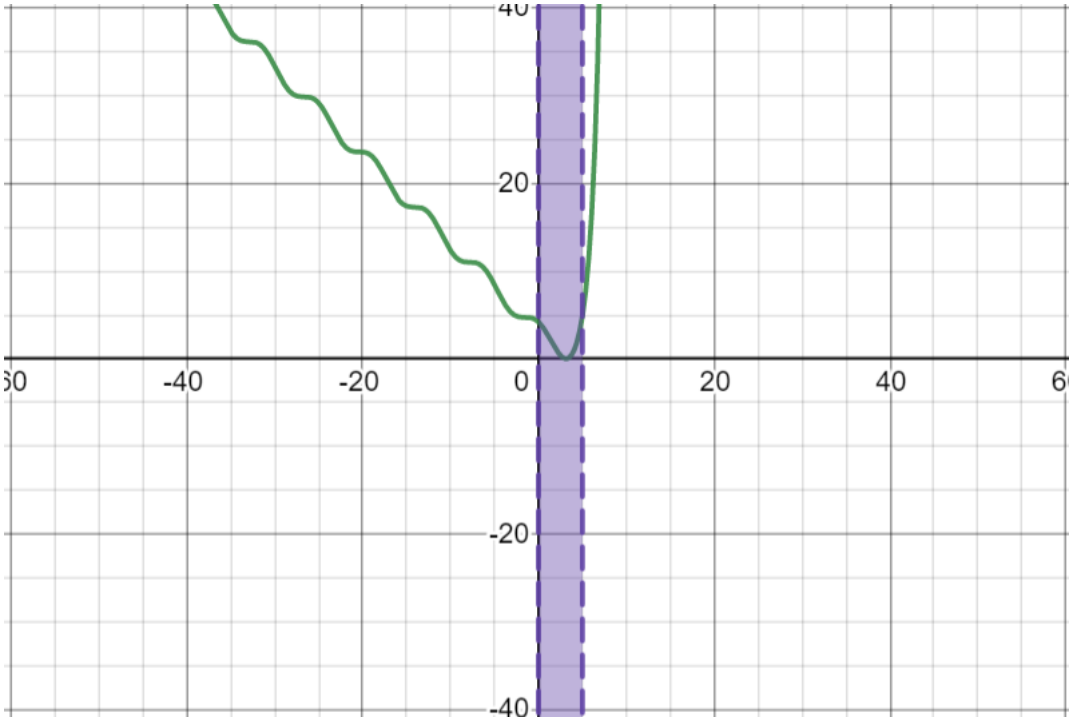


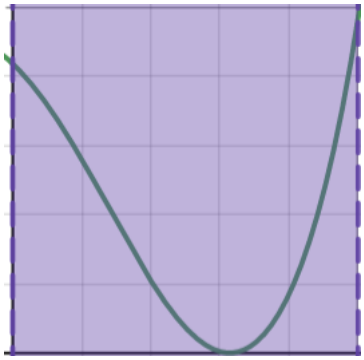

```
clear all
clc
syms R
% write the equation to be solved
f = log(R)*2.341077*10^(-4)+ log(R)^3*8.775468*10^(-8)-2.293775*10^(-3);
% derivative of the function
g = diff(f);
n = 3; % number of decimal places
epsilon = 0.5*10^(2-n);
R0 = 15000; % initial guess
%%
for i=1:100
f0 = vpa(subs(f,R,R0)); %function value at R0
f0_der = vpa(subs(g,R,R0));%function derivative at R0
y = R0 - f0/f0_der; % update the next guess
err = abs(y-R0);
if err<epsilon
break
end
R0 = y;
end
%%
y = y - rem(y,10^-n)
%fprintf('The Root is : %f ',y);
```

Ex.3 Matlab

Lupascu Felicia

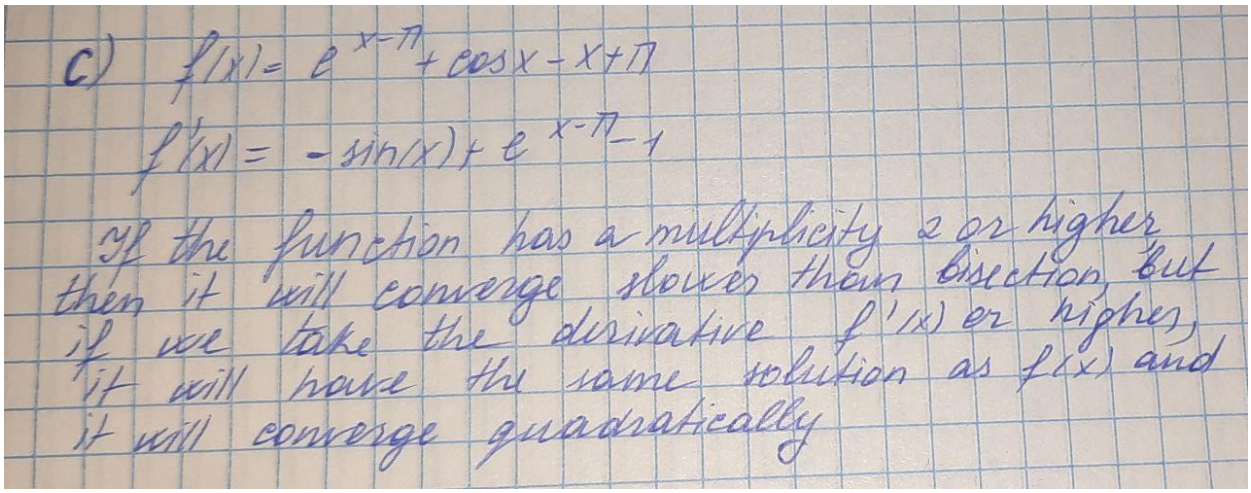
Ex.4





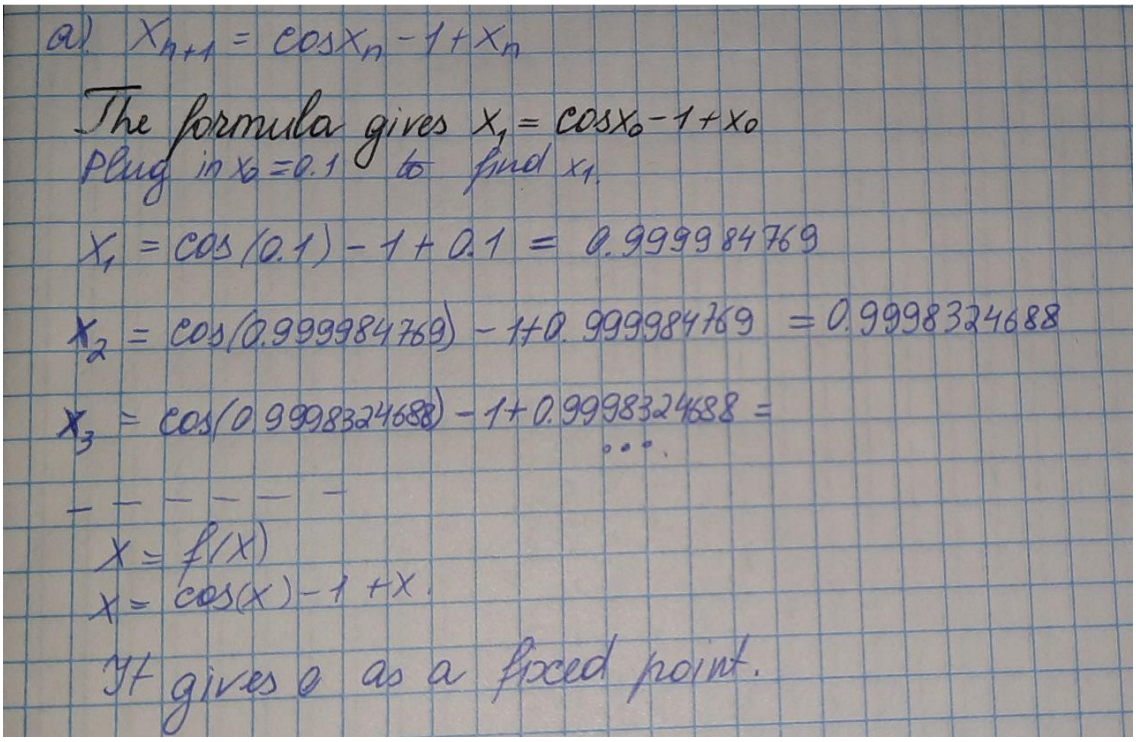
Order of convergence=
1.9617575539126844

It tends to 2 . It is quadratic.



```
function dummy=bisc_newton_secant()
clc;
clear all;

f=@(e^(x-pi)+cos(x)-x+pi); %function
fp=@(-sin(x)+e^(x-pi)-1); % derivative of unction

tol=1e-8;

a=0;
b=5; % interval
x0=5;
disp('Root by Bisection method')
y=bisecion(f,a,b,tol) % function calling

a1=0;
b1=5; % interval
disp('Root by fixed point method')
C1=fixedp(g,x0,tol) % function calling

disp('Root by Newton method')
y2=newt(f,x0,tol)% function calling

a2=0;
b2=5; % interval

disp('Root by Secant method')
y1=secn(f,a2,b2,tol)% function calling
```

Ex.5

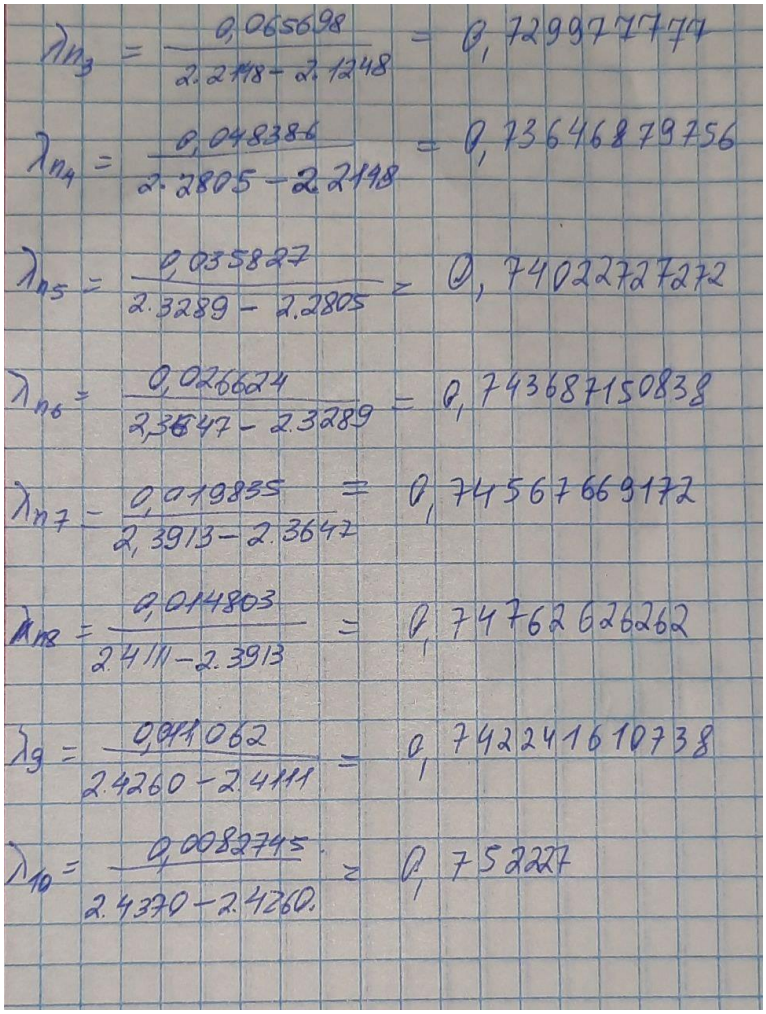


```cpp
// C++ program for implementation of Newton Raphson Method for
// solving equations
#include<bits/stdc++.h>
#define EPSILON 0.001
using namespace std;

// An example function whose solution is determined using
// Bisection Method. The function is xn+1 = cos xn - 1 + xn
double func(double x)
{
    return cos(x)-1+x;
}

// Derivative of the above function which is 1-sin(x)
double derivFunc(double x)
{
    return  1-sin(x);
}

// Function to find the root
void newtonRaphson(double x)
{
    double h = func(x) / derivFunc(x);
    while (abs(h) >= EPSILON)
    {
        h = func(x)/derivFunc(x);

        // x(i+1) = x(i) - f(x) / f'(x)
        x = x - h;
    }

    cout << "The value of the root is : " << x;
}

// Driver program to test above
int main()
{
    double x0 = -0.1; // Initial values assumed
    newtonRaphson(x0);
    return 0;
}
```

Ex.6

$$\lambda n = \frac{x_n - x_{n-1}}{x_{n-1} - x_{n-2}}$$

n	x_n	$x_n - x_{n-1}$	λn
0	2.0		
1	2.1248	0.124834	
2	2.2148	0.089944	0,720705
3	2.2805	0.065698	0.72997777
4	2.3289	0.048386	0.736468797
5	2.3647	0.035827	0,74022727272
6	2.3913	0.026624	0.74567669172
7	2.4111	0.019835	0.74567669172
8	2.4260	0.014803	0,747626262
9	2.4370	0.011062	0,74224161073
10	2.4453	0.0082745	0,752227



a)An iterative method is called convergent if the corresponding sequence converges for given initial approximations. Out function is convergent because it tends to the initial value 2.

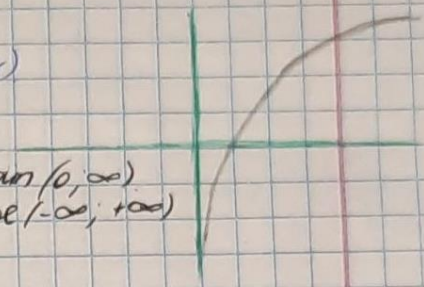
b) Yes.

c) When the condition is satisfied, Newton's method converges, and it also converges faster than almost any other alternative iteration scheme based on other methods of coverting the original f(x) to a function with a fixed point.

Ex.7

Given function is $f(x) = x + \ln x = 0$

Graph of $f(x) = x + \ln(x)$



domain $(0, \infty)$
Range $(-\infty, +\infty)$

① i) $x_2 = -\ln(x) = \varphi(x) \Rightarrow \varphi(x) = -\ln(x)$

$|\varphi'(x)| = |-\frac{1}{x}|$

Therefore $|\varphi'(x)| = |\frac{1}{x}| \not< 1$ in its domain.

$\Rightarrow x$ can not converge

③ iii) $x = \frac{x + e^{-x}}{2}$

$\varphi(x) = \frac{x + e^{-x}}{2}$; $|\varphi'(x)| = \frac{1}{2}|1 - e^{-x}| < 1$

$\forall x$ in the domain of f.

$\Rightarrow x = \frac{x + e^{-x}}{2}$ will converge

② ii) $x = e^{-x} = \varphi(x) = \varphi(x)$

$\varphi'(x) = |-e^{-x}| = |e^{-x}| < 1 \quad \forall x$ in the domain of f.

$\Rightarrow \varphi(x) = x = e^{-x}$ will converge definitely

② and ③ can be used to find the root of f.

A better formula to solve this equation by Newton-Raphson meth

Newton-Raphson method is given by:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

We have $f(x) = x + \ln(x)$

$f'(x) = 1 + \frac{1}{x}$.

$a_0 = 0.5$

Putting $n = 0$ in the above iterative formula we get

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = 0.5 - \frac{0.5 + \ln(0.5)}{1 + \frac{1}{0.5}} = 0.5 - \frac{-0.1931478}{3}$$

i.e x_1 = first iteration = 0.5643823933

Second iteration

$$x_3 = x_2 - \frac{f(x_2)}{f'(x_2)} = 0.5671389887 - \frac{0.5671389887 + \ln(0.5671389887)}{1 + \frac{1}{0.5671389887}} =$$

$$= 0.5671458314$$

Relative error $= |x_3 - x_2| = 0.0000068445432222$

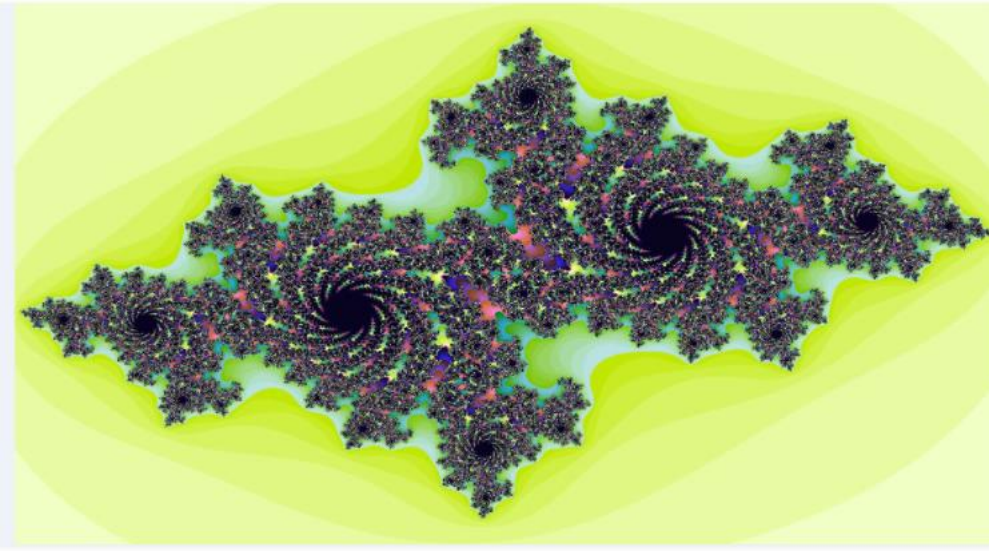
$x_3 = 0.5671458314$ is correct up to 5 digits.

x_3 is a root of the eq. $x + \ln(x) = 0$.

Lupascu Felicia

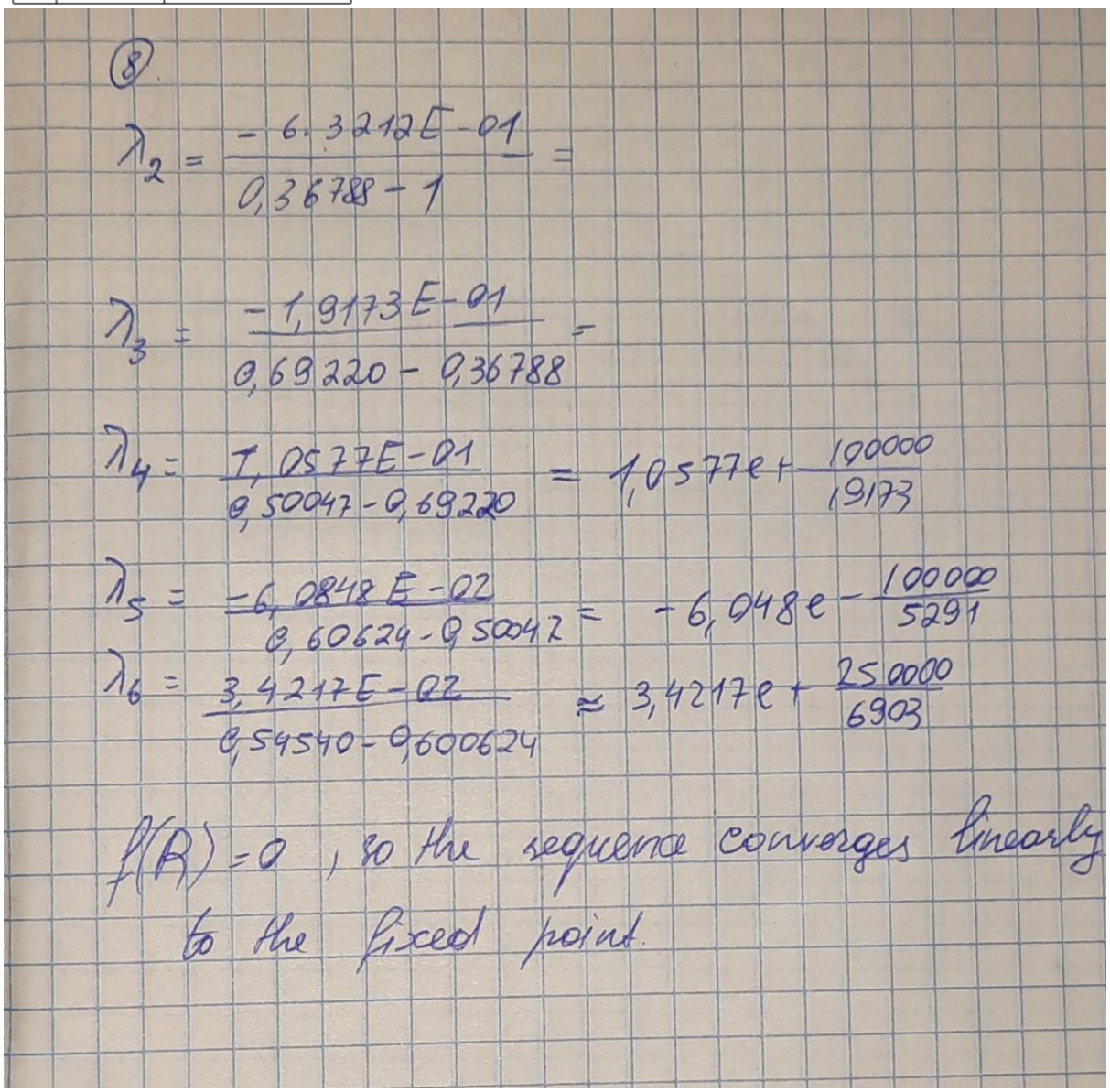
Ex.9

```python
# Python code for Julia Fractal
from PIL import Image

# driver function
if __name__ == "__main__":

    # setting the width, height and zoom
    # of the image to be created
    w, h, zoom = 1920,1080,1

    # creating the new image in RGB mode
    bitmap = Image.new("RGB", (w, h), "white")

    # Allocating the storage for the image and
    # loading the pixel data.
    pix = bitmap.load()

    # setting up the variables according to
    # the equation to create the fractal
    cX, cY = -0.7, 0.27015
    moveX, moveY = 0.0, 0.0
    maxIter = 255

    for x in range(w):
        for y in range(h):
            zx = 1.5*(x - w/2)/(0.5*zoom*w) + moveX
            zy = 1.0*(y - h/2)/(0.5*zoom*h) + moveY
            i = maxIter
            while zx*zx + zy*zy < 4 and i > 1:
                tmp = zx*zx - zy*zy + cX
                zy,zx = 2.0*zx*zy + cY, tmp
                i -= 1

            # convert byte to RGB (3 bytes), kinda
            # magic to get nice colors
            pix[x,y] = (i << 21) + (i << 10) + i*8

    # to display the created fractal
    bitmap.show()

```



Ex.8

n	x_n	$x_n - x_{n-1}$
0	1.00	
1	0.36788	$-6.3212E-01$
2	0.69220	$3.2432E-01$
3	0.50047	$-1.9173E-01$
4	0.60624	$1.0577E-01$
5	0.54540	$-6.0848E-02$
6	0.57961	$3.4217E-02$



We can also use Aitken extrapolation formula

Homework 1

Exercise 1.2

Number 12.1875

Value actually stored in float 12.1875

Binary representation 01000001010000110000000000000000

Hexadecimal representation 0x41430000

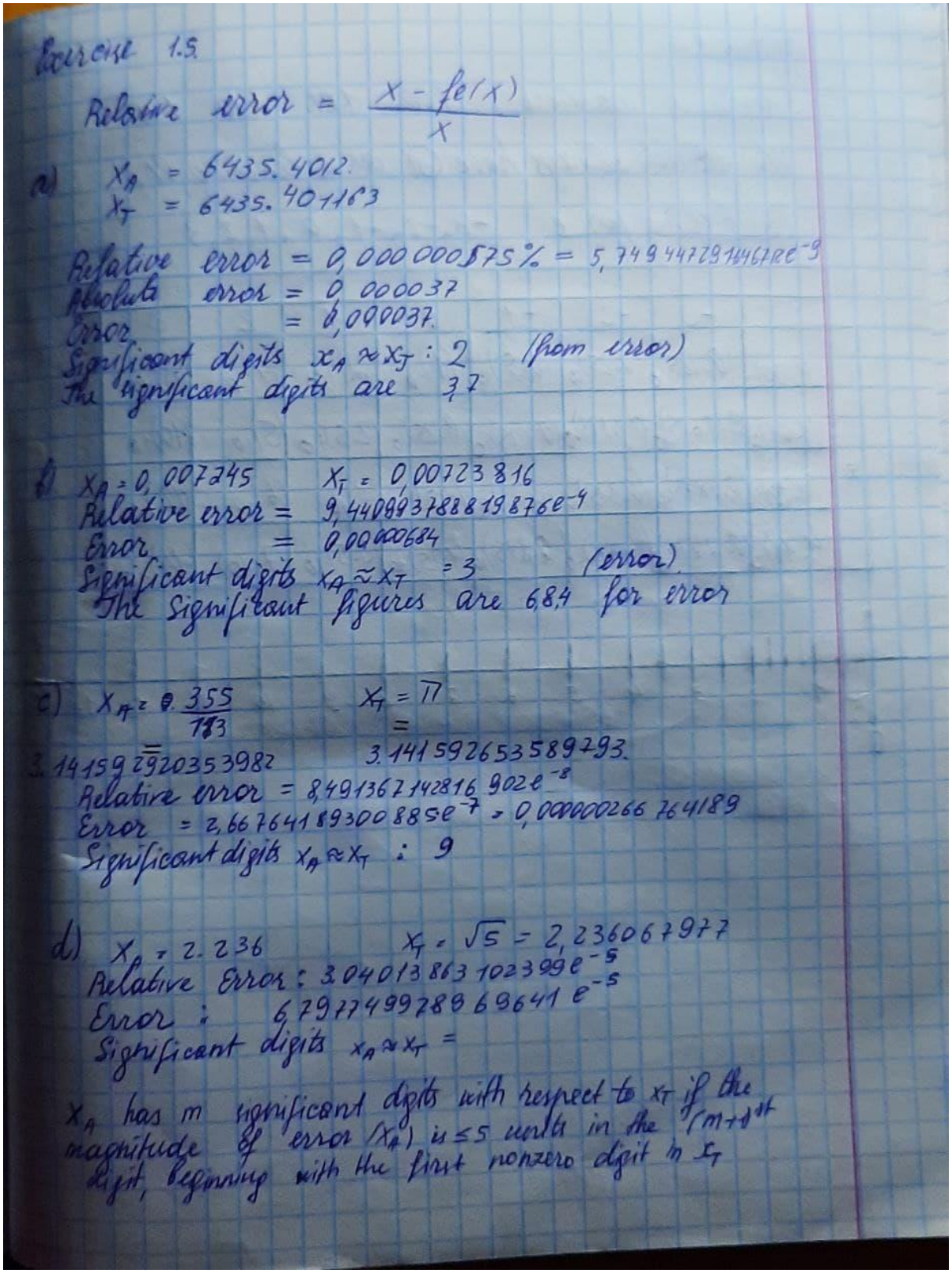
	Sign	Exponent	Mantissa
Value	+1	2^3	1.5234375
Encoded as	0	130	43090912

Exercise 1.3

Exponent: 7 bits
Sign 0 : 1 bit
Mantissa : 16 bits with no heading of the leading bit 1
The arithmetic used chopping

The single ~~form~~ precision IEEE format of x consists of a precision of 24 binary digits.

The exponent e is limited by $-126 < e < 127$



Exercise 1.5

Relative error = $\frac{x - fe(x)}{x}$

a) $x_A = 6435.4012$
$x_T = 6435.401163$

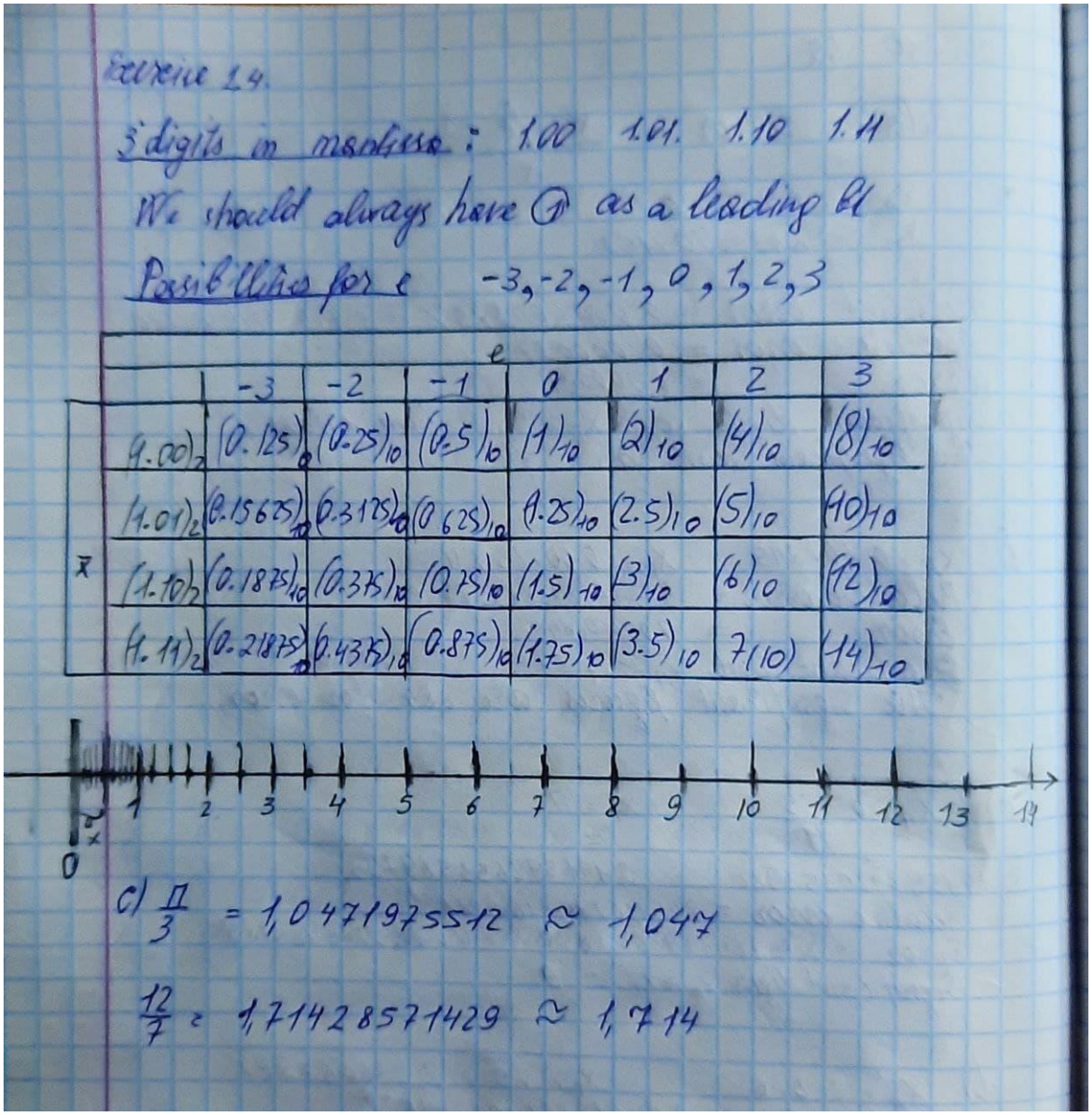
Relative error = 0,000000875% = $5,749447291467RE^{-9}$
Absolute error = 0,000037
Error = 0,000037
Significant digits $x_A \approx x_T$: 2 (from error)
The significant digits are 3,7

b) $x_A = 0,007245$ $x_T = 0,00723816$
Relative error = $9,440893788819876e^{-4}$
Error = 0,00000684
Significant digits $x_A \approx x_T$ = 3 (error)
The significant figures are 6,8,4 for error

c) $x_A = 0.\frac{355}{113}$ $x_T = \overline{11}$
=
3.14159292035398 3.14159265358979 3.
Relative error = $8,491367142816902e^{-8}$
Error = $2,667641893008855e^{-7} = 0,000000266764189$
Significant digits $x_A \approx x_T$: 9

d) $x_A = 2.236$ $x_T = \sqrt{5} = 2,236067977$
Relative Error: $3,040138631023996e^{-5}$
Error: $6,797749928969641e^{-5}$
Significant digits $x_A \approx x_T$ =

x_A has m significant digits with respect to x_T if the magnitude of error (x_A) is ≤ 5 units in the (m+1)st digit, beginning with the first nonzero digit in x_T



Exercise 14.

3 digits in mantissa: 1.00 1.01 1.10 1.11

We should always have (1) as a leading bit

Possibilities for e: -3, -2, -1, 0, 1, 2, 3

x \ e	-3	-2	-1	0	1	2	3
$(1.00)_2$	$(0.125)_{10}$	$(0.25)_{10}$	$(0.5)_{10}$	$(1)_{10}$	$(2)_{10}$	$(4)_{10}$	$(8)_{10}$
$(1.01)_2$	$(0.15625)_{10}$	$(0.3125)_{10}$	$(0.625)_{10}$	$(1.25)_{10}$	$(2.5)_{10}$	$(5)_{10}$	$(10)_{10}$
$(1.10)_2$	$(0.1875)_{10}$	$(0.375)_{10}$	$(0.75)_{10}$	$(1.5)_{10}$	$(3)_{10}$	$(6)_{10}$	$(12)_{10}$
$(1.11)_2$	$(0.21875)_{10}$	$(0.4375)_{10}$	$(0.875)_{10}$	$(1.75)_{10}$	$(3.5)_{10}$	7(10)	$(14)_{10}$

c) $\frac{\Pi}{3} = 1,047197551 2 \approx 1,047$

$\frac{12}{7} = 1,714285714 29 \approx 1,714$

Exercise 1.6

Avoid loss of significance errors in formulas

a) $\log(x) - \log(x-1)$ for large values of x

- We can avoid the loss of precision by multiplying the function by conjugate

$$\frac{\log(x) - \log(x-1)}{1} \cdot \frac{\log(x) + \log(x-1)}{\log(x) + \log(x-1)} =$$

$$= \frac{(\log(x) - \log(x-1))(\log(x) + \log(x-1))}{\log(x) + \log(x-1)}.$$

$f(100) =$ 0,0043648054 in first case

$f(100) =$ 0,004364805402450004 in second case

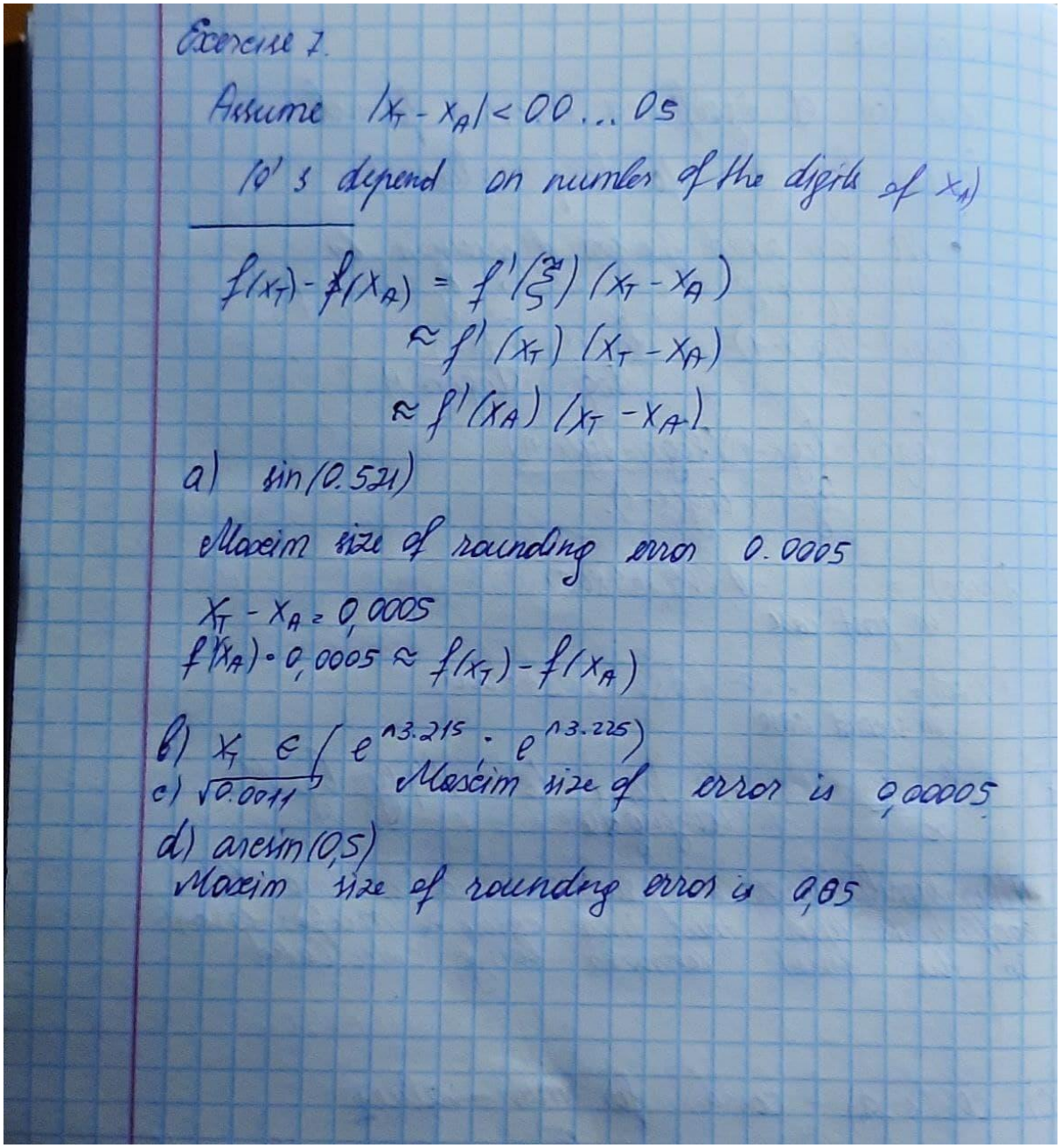
b) $\frac{e^x - 1}{x}$ We have to multiply by the conjugate!

Other similar errors are present in calculating other coefficients, and thus they cause a major error in the final answer being calculated

c) $\cos(x+d) - \cos(d)$ for small values

We have to substitute the expresion with something simmilar

$\cos(\alpha) = 1 - 2\sin^2\left(\frac{\alpha}{2}\right)$.



Exercise 1.

Assume $|x_T - x_A| < 00\ldots05$

10's depend on number of the digits of x_A)

$$f(x_T) - f(x_A) = f'(\xi)(x_T - x_A)$$
$$\approx f'(x_T)(x_T - x_A)$$
$$\approx f'(x_A)(x_T - x_A)$$

a) $\sin(0.521)$

Maxeim size of rounding error 0.0005

$x_T - x_A = 0,0005$

$f(x_A) \cdot 0,0005 \approx f(x_T) - f(x_A)$

b) $x_T \in (e^{\wedge 3.215} \cdot e^{\wedge 3.225})$

c) $\sqrt{0.0011}$ Maxeim size of error is 0,00005

d) $\arcsin(0,5)$

Maxeim size of rounding error is 0,05

Homework 3

Ex.3.1

x	2.0	4.5	5.25	7.81	9.2	10.6
y	7.2	7.1	6.0	5.0	3.5	5.0

For fifth polynomial interpolation we choose the value of y given by.

$$y(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5$$

Using the six points.

$x_0 = 2.00$ $\quad y(x_0) = 7.2$
$x_1 = 4.25$ $\quad y(x_1) = 7.1$
$x_2 = 5.25$ $\quad y(x_2) = 6.0$
$x_3 = 7.81$ $\quad y(x_3) = 5.0$
$x_4 = 9.20$ $\quad y(x_4) = 3.5$
$x_5 = 10.60$ $\quad y(x_5) = 5.0$

gives:

$$y(2.00) = a_0 + a_1(2.00) + a_2(2.00)^2 + a_3(2.00)^3 + a_4(2.00)^4 + a_5(2.00)^5 = 7.2$$
$$y(4.25) = a_0 + a_1(4.25) + a_2(4.25)^2 + a_3(4.25)^3 + a_4(4.25)^4 + a_5(4.25)^5 = 7.1$$
$$y(5.25) = a_0 + a_1(5.25) + a_2(5.25)^2 + a_3(5.25)^3 + a_4(5.25)^4 + a_5(5.25)^5 = 6.0$$
$$y(7.81) = a_0 + a_1(7.81) + a_2(7.81)^2 + a_3(7.81)^3 + a_4(7.81)^4 + a_5(7.81)^5 = 5.0$$
$$y(9.20) = a_0 + a_1(9.20) + a_2(9.20)^2 + a_3(9.20)^3 + a_4(9.20)^4 + a_5(9.20)^5 = 3.5$$
$$y(10.60) = a_0 + a_1(10.60) + a_2(10.60)^2 + a_3(10.60)^3 + a_4(10.60)^4 + a_5(10.60)^5 = 5.0$$

Writing the six equations in matrix form:

$$\begin{bmatrix} 1 & 2.00 & 4 & 8 & 16 & 32 \\ 1 & 4.25 & 18.063 & 76.766 & 326.25 & 1386.6 \\ 1 & 5.25 & 27.563 & 144.70 & 769.69 & 3988.4 \\ 1 & 7.81 & 60.996 & 476.38 & 3720.5 & 29057 \\ 1 & 9.20 & 84.64 & 778.69 & 7163.9 & 65908 \\ 1 & 10.60 & 112.36 & 1191.0 & 12625 & 133820 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} 7.2 \\ 7.1 \\ 6.0 \\ 5.0 \\ 3.5 \\ 5.0 \end{bmatrix}$$

Solving the above six equations gives

$a_0 = -30.898$
$a_1 = 41.344$
$a_2 = -15.855$
$a_3 = 2.7862$
$a_4 = -0.23091$
$a_5 = 0.0072923$

Hence

$$y(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5$$

$$= -30.898 + 41.344x - 15.855x^2 + 2.7862x^3 - 0.23091x^4 + 0.0072923x^5, \qquad 2 \le x \le 16$$

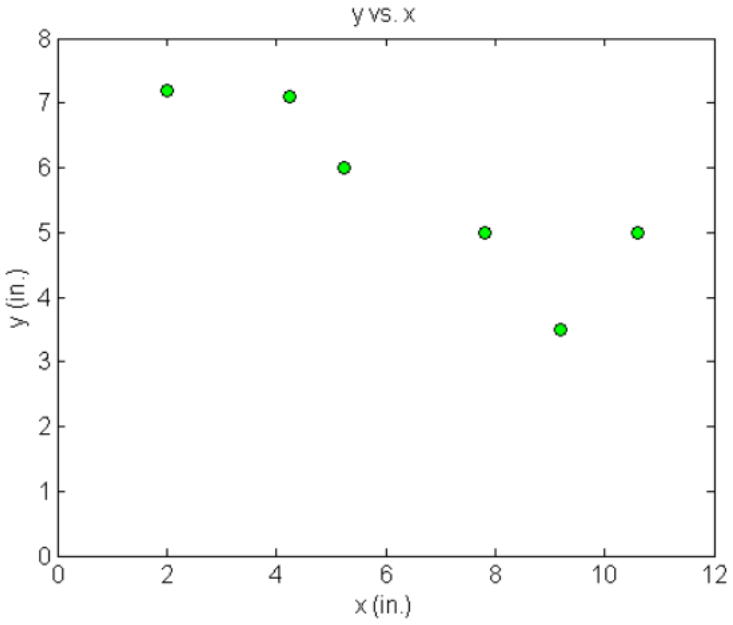


Figure 1 Location of holes on the rectangular plate.

Cubic Spline formula is

$$f(x)=\frac{(x_i-x)^3}{6h}M_{i-1}+\frac{(x-x_{i-1})^3}{6h}M_i+\frac{(x_i-x)}{h}\left(y_{i-1}-\frac{h^2}{6}M_{i-1}\right)+\frac{(x-x_{i-1})}{h}\left(y_i-\frac{h^2}{6}M_i\right)$$

We have $M_{i-1}+4M_i+M_{i+1}=\frac{6}{h^2}(y_{i-1}-2y_i-y_{i+1})$

Here $h=2.5$ $n=5$ $M_0=0$ $M_5=0$

Substitute $i=1$ in equation

$M_0+4M_1+M_2=\frac{6}{h^2}(y_0-2y_1+y_2) \Rightarrow 4M_1+M_2=-0.96$

$M_1+4M_2+M_3=\frac{6}{h^2}(y_1-2y_2+y_3) \Rightarrow M_1+4M_2+M_3=0.096$

$M_2+4M_3+M_4=\frac{6}{h^2}(y_2-2y_3+y_4) \Rightarrow M_2+4M_3+M_4=-0.48$

$M_3+4M_4+M_5=\frac{6}{h^2}(y_3-2y_4+y_5) \Rightarrow M_3+4M_4=2.88$

Substitute $i=1$ in eq, we get cubic spline 1st interval $[2\ 4\ 5]$

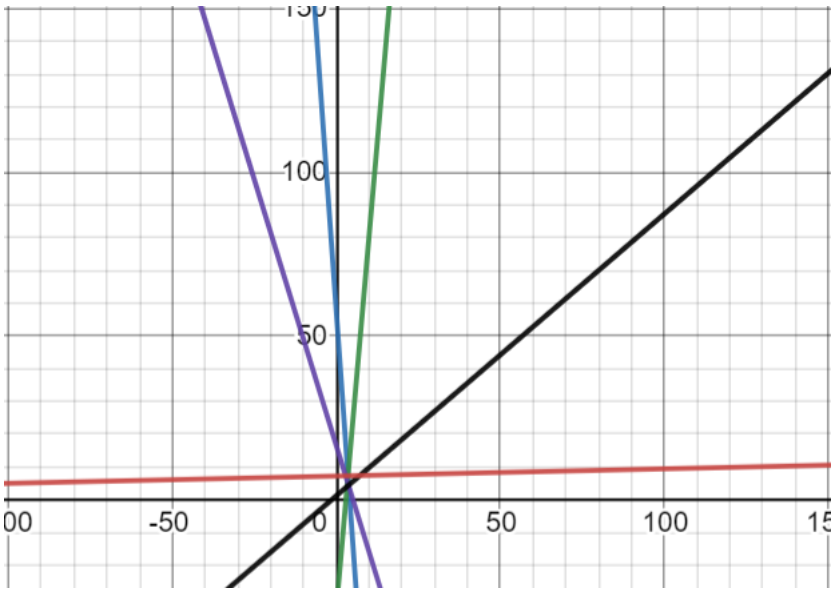
$f_1(x)=-0.0191x^3+0.1148x^2-0.1501x+7.1939$ for $2.0\le x\le 4.5$

$f_2(x)=0.0317x^3-0.4709x^2+1.7072x+1.776$ for $4.5\le x\le 5.25$

$f_3(x)=-0.0372x^3+0.6828x^2-4.5051x+16.3729$ for $5.25\le x\le 7.81$

$f_4(x)=0.0789x^3-1.9504x^2+15.087x-33.4957$ for $7.81\le x\le 9.2$

$f_5(x)=-0.0542x^3+1.7077x^2-17.3222x+57.3723$ for $9.2\le x\le 10.6$



Ex.3.2

a)

n	1	2	3	4	5
$\Gamma(n)$	1	1	2	6	24

```
import numpy as np
import matplotlib.pyplot as plt

plt.style.use('seaborn-poster')

%matplotlib inline
def divided_diff(x, y):
    '''
    function to calculate the divided
    differences table
    '''
    n = len(y)
    coef = np.zeros([n, n])
    # the first column is y
    coef[:,0] = y

    for j in range(1,n):
        for i in range(n-j):
            coef[i][j] = \
           (coef[i+1][j-1] - coef[i][j-1]) / (x[i+j]-x[i])

    return coef

def newton_poly(coef, x_data, x):
    '''
    evaluate the newton polynomial
    at x
    '''
    n = len(x_data) - 1
    p = coef[n]
    for k in range(1,n+1):
        p = coef[n-k] + (x -x_data[n-k])*p
    return p
x = np.array([1,2,3,4,5])
y = np.array([1,1,2,6,24])
# get the divided difference coef
a_s = divided_diff(x, y)[0, :]

# evaluate on new data points
#x_new = np.arange(1, 2,3,4,5)
#Y_new = newton_poly(a_s, x, x_new)

plt.figure(figsize = (12, 8))
plt.plot(x, y, 'bo')
#plt.plot(x_new, y_new)
```

$$
\begin{array}{llllll}
x_0 & y_0 & & & & \\
 & & f[x_1,x_0] & & & \\
x_1 & y_1 & & f[x_2,x_1,x_0] & & \\
 & & f[x_2,x_1] & & f[x_3,x_2,x_1,x_0] & \\
x_2 & y_2 & & f[x_3,x_2,x_1] & & f[x_4,x_3,x_2,x_1,x_0] \\
 & & f[x_3,x_2] & & f[x_4,x_3,x_2,x_1] & \\
x_3 & y_3 & & f[x_4,x_3,x_2] & & \\
 & & f[x_4,x_3] & & & \\
x_4 & y_4 & & & &
\end{array}
$$

x	y	1^{st} order	2^{nd} order	3^{rd} order	4^{th} order
1	1				
		0			
2	1		0.5		
		1		0.333333	
3	2		1.5		-0.458333
		4		-1.5	
4	6		-3		
		-2			
5	4				

Newton's divided difference interpolation formula

$$f(x) = y_0 + (x - x_0)f[x_0, x_1] + (x - x_0)(x - x_1)f[x_0, x_1, x_2] + (x - x_0)(x - x_1)(x - x_2)f[x_0, x_1, x_2, x_3] + (x - x_0)(x - x_1)(x - x_2)(x - x_3)f[x_0, x_1, x_2, x_3, x_4]$$

$$f(x) = 1 + (x - 1) \times 0 + (x - 1)(x - 2) \times 0.5 + (x - 1)(x - 2)(x - 3) \times 0.333333 + (x - 1)(x - 2)(x - 3)(x - 4) \times -0.458333$$

$$f(x) = 1 + (x - 1) \times 0 + \left(x^2 - 3x + 2\right) \times 0.5 + \left(x^3 - 6x^2 + 11x - 6\right) \times 0.333333 + \left(x^4 - 10x^3 + 35x^2 - 50x + 24\right) \times -0.458333$$

$$f(x) = 1 + (0) + \left(0.5x^2 - 1.5x + 1\right) + \left(0.333333x^3 - 2x^2 + 3.666667x - 2\right) + \left(-0.458333x^4 + 4.583333x^3 - 16.041667x^2 + 22.916667x - 11\right)$$

$$f(x) = -0.458333x^4 + 4.916667x^3 - 17.541667x^2 + 25.083333x - 11$$

$$f(x) = \frac{(x_i - x)^3}{6h} M_{i-1} + \frac{(x - x_{i-1})^3}{6h} M_i + \frac{(x_i - x)}{h}\left(y_{i-1} - \frac{h^2}{6} M_{i-1}\right) + \frac{(x - x_{i-1})}{h}\left(y_i - \frac{h^2}{6} M_i\right)$$

b)

We have $M_{i-1} + 4M_i + M_{i+1} = \frac{6}{h^2}(y_{i-1} - 2y_i + y_{i+1})$

$h=1$ $h=4$ $M_0=0$ $M_4=0$

$M_0 + 4M_1 + M_2 = \frac{6}{h^2}(y_0 - 2y_1 + y_2) \Rightarrow 4M_1 + M_2 = 6$

$M_1 + 4M_2 + M_3 = \frac{6}{h^2}(y_1 - 2y_2 + y_3) \Rightarrow M_1 + 4M_2 + M_3 = 18$

$M_2 + 4M_3 + M_4 = \frac{6}{h^2}(y_2 - 2y_3 + y_4) \Rightarrow M_2 + 4M_3 = 84$

$f_1(x) = 0.3036x^3 - 0.9107x^2 + 0.6071x + 1$ for $1 \le x \le 2$

$f_2(x) = -0.5178x^3 + 4.0178x^2 - 9.2498x + 7.5713$ for $2 \le x \le 3$

$f_3(x) = 3.7678x^3 - 34.5535x^2 + 106.4641x - 108.1426$ for $3 \le x \le 4$

$f_4(x) = -3.5536x^3 + 53.3035x^2 - 244.9639x + 360.428$ for $4 \le x \le 5$

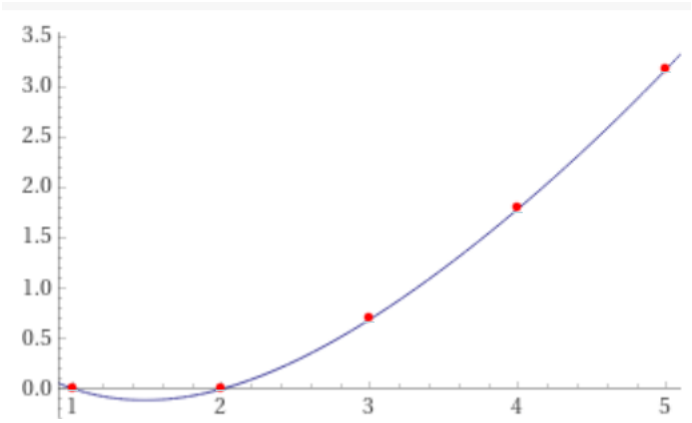
c)

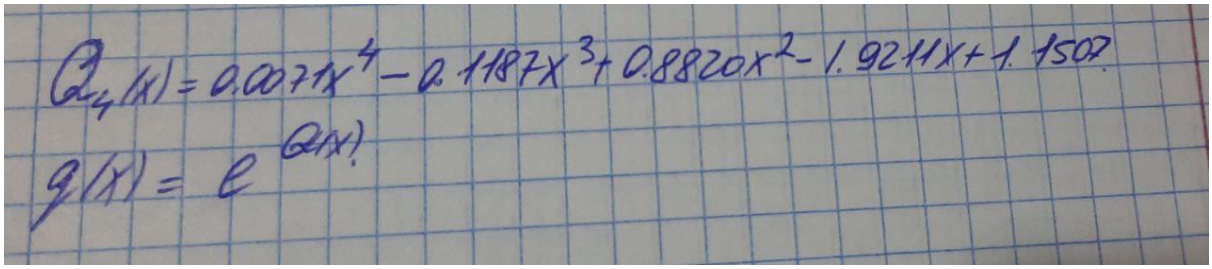
n	1	2	3	4	5
$\log \Gamma(n)$	0	0	$\log 2$	$\log 6$	$\log 24$

c) $\frac{1}{24}x^4 \log 24 - \frac{1}{6}x^4 \log 6 + \frac{1}{4}\log 2 - \frac{5}{12}x^3 \log 24 + \frac{11}{6}x^3 \log 6 -$

$- 3x^3 \log 2 + \frac{35}{24}x^2 \log(24) - \frac{41}{6}x^2 \log 6 + \frac{49}{4}x^2 \log 2 - \frac{25}{12}x \log 24 +$

$\frac{61}{6}x \log 6 - \frac{39}{2}x \log 2 + \log 24 - 5 \log 6 + 10 \log 2$



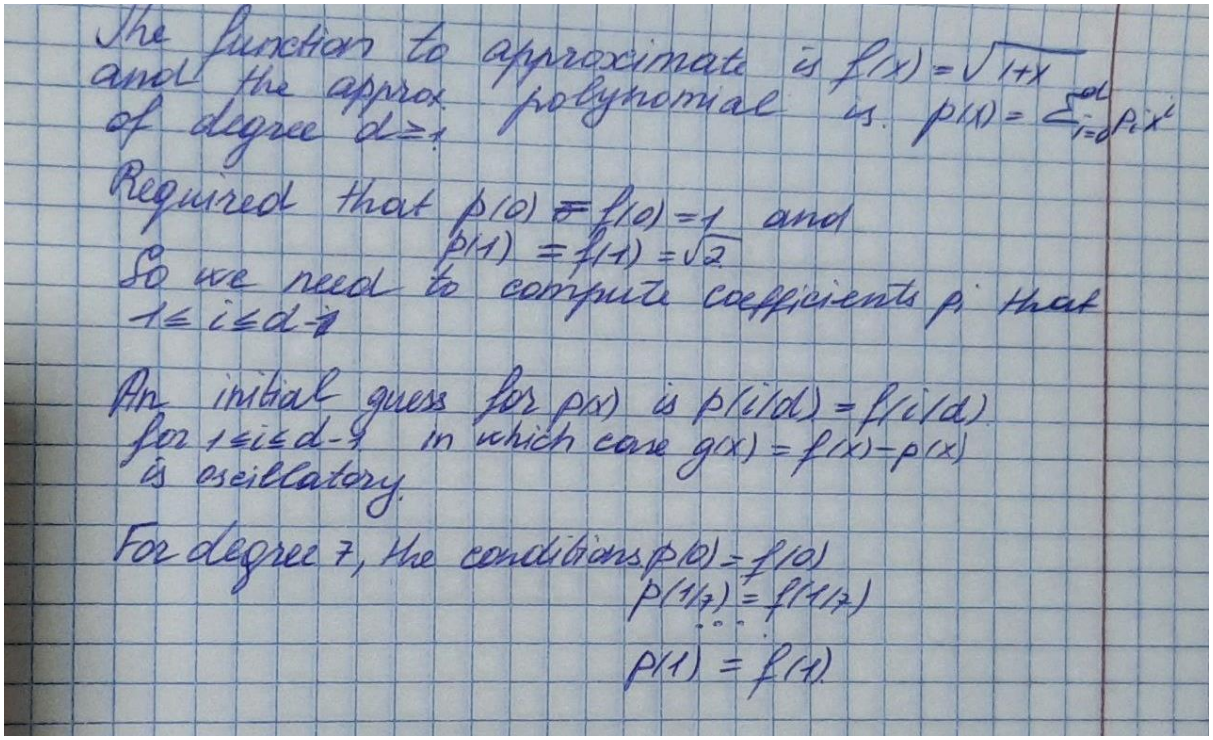


Ex.3.3

$$f(x) = \sqrt{x+1}$$

interval $[-1, 1]$

```
>> pkg load symbolic
>> syms n x
>> C=chebyshevU(8,x)
C = (sym)

       8         6         4         2
  256*x  - 448*x  + 240*x  - 40*x  + 1
```



d	coefficients			d	coefficients		
1	p_0	=	$+1$	2	p_0	=	$+1$
	p_1	=	$+4.1421356237309505 * 10^{-1}$		p_1	=	$+4.8563183076125260 * 10^{-1}$
	e	=	$+1.7766952966368793 * 10^{-2}$		p_2	=	$-7.1418268388157458 * 10^{-2}$
					e	=	$+1.1795695163108744 * 10^{-3}$
3	p_0	=	$+1$	4	p_0	=	$+1$
	p_1	=	$+4.9750045320242231 * 10^{-1}$		p_1	=	$+4.9955939832918816 * 10^{-1}$
	p_2	=	$-1.0787308044477850 * 10^{-1}$		p_2	=	$-1.2024066151943025 * 10^{-1}$
	p_3	=	$+2.4586189615451115 * 10^{-2}$		p_3	=	$+4.5461507257698486 * 10^{-2}$
	e	=	$+1.1309620116468910 * 10^{-4}$		p_4	=	$-1.0566681694362146 * 10^{-2}$
					e	=	$+1.2741170151556180 * 10^{-5}$
5	p_0	=	$+1$	6	p_0	=	$+1$
	p_1	=	$+4.9992197660031912 * 10^{-1}$		p_1	=	$+4.9998616695784914 * 10^{-1}$
	p_2	=	$-1.2378506719245053 * 10^{-1}$		p_2	=	$-1.2470733323278438 * 10^{-1}$
	p_3	=	$+5.6122776972699739 * 10^{-2}$		p_3	=	$+6.0388587356982271 * 10^{-2}$
	p_4	=	$-2.3128836281145482 * 10^{-2}$		p_4	=	$-3.1692053551807930 * 10^{-2}$
	p_5	=	$+5.0827122737047148 * 10^{-3}$		p_5	=	$+1.2856590305148075 * 10^{-2}$
	e	=	$+1.5725568940708201 * 10^{-6}$		p_6	=	$-2.6183954624343642 * 10^{-3}$
					e	=	$+2.0584155535630089 * 10^{-7}$
7	p_0	=	$+1$	8	p_0	=	$+1$
	p_1	=	$+4.9999754817809228 * 10^{-1}$		p_1	=	$+4.9999956583056759 * 10^{-1}$
	p_2	=	$-1.2493243476353655 * 10^{-1}$		p_2	=	$-1.2498490369914350 * 10^{-1}$
	p_3	=	$+6.1859954146370910 * 10^{-2}$		p_3	=	$+6.2318494667579216 * 10^{-2}$
	p_4	=	$-3.6091595023208356 * 10^{-2}$		p_4	=	$-3.7982961896432244 * 10^{-2}$
	p_5	=	$+1.9483946523450868 * 10^{-2}$		p_5	=	$+2.3642612312869460 * 10^{-2}$
	p_6	=	$-7.5166134568007692 * 10^{-3}$		p_6	=	$-1.2529377587270574 * 10^{-2}$
	p_7	=	$+1.4127567687864939 * 10^{-3}$		p_7	=	$+4.5382426960713929 * 10^{-3}$
	e	=	$+2.8072302919734948 * 10^{-8}$		p_8	=	$-7.8810995273670414 * 10^{-4}$
					e	=	$+3.9460605685825989 * 10^{-9}$

The numbers are coefficient pi for the polynomyal p(x), The table shows the maximum error for the approximation.

```
% construct N=8 data points
N=8;
xdata=linspace(-1,1,N);
```

```
ydata=sqrt(1+x)(xdata);

% construct many test points
xval=linspace(???,???,4001);
% construct the true test point values, for reference
yvalTrue=sinh(???);

% use Lagrange polynomial interpolation to evaluate
% the interpolant at the test points
yval=eval_lag(???,???,xval);

% plot reference values in thick green
plot(xval,yvalTrue,'g','linewidth',4);
hold on
% plot interpolation data points
plot(xdata,ydata,'k+');
% plot interpolant in thin black
plot(xval,yval,'k');
hold off

% estimate the approximation error of the interpolant
approximationError=max(abs(yvalTrue-yval))/max(abs(yvalTrue))
```

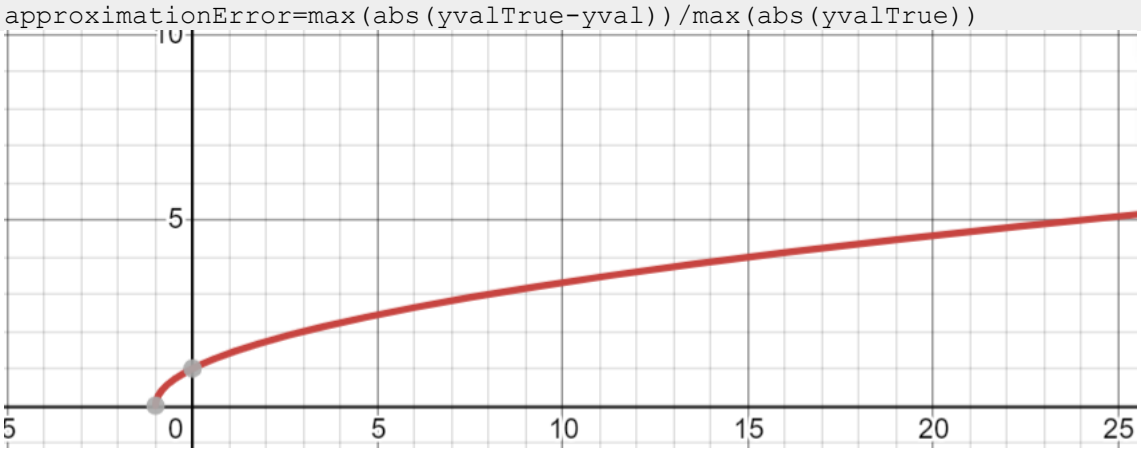
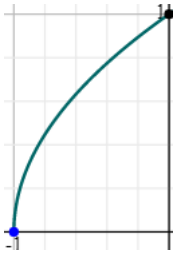


Figure 1 sqrt(x+1)



Homework 1

Due February 11, 19.00

Problem 1.1

All readings are posted on the course web page.

a) Read Lectures 0, 1.1, 1.2, 1.3.

b) Review how to do binary to decimal and vice-versa conversions.

c) Read definitions of Numerical Analysis by K. Atkinson and L. Trethenen.

d) Read the history of IEEE standard 754.

e) Read the paper on some common bugs related to computer representation of numbers.

Problem 1.2

Write the binary single precision IEEE floating-point expression for the number 12.1875. Specify sign σ, exponent E and mantissa.

Problem 1.3

Some microcomputers in the past used a binary floating-point format with 7 bits for the exponent and 1 bit for the sign σ. The mantissa contained 16 bits, with no hiding of the leading bit 1. The arithmetic used chopping. Determine the accuracy of the representation by finding the following:

a) machine epsilon;

b) integer M;

c) accuracy of the chopping operation.

Problem 1.4

Consider a binary floating-point representation with mantissa containing 3 digits without hiding the leading 1 and $-3_{10} \leqslant e \leqslant 3_{10}$.

a) List all numbers that can be stored exactly together with their decimal value.

b) Plot these numbers on real axis.

c) For this arithmetic, specify what are the corresponding floating-point representation of $\pi/3$ and $12/7$ if rounding is used.

d) Repeat c) if chopping is being used.

Problem 1.5

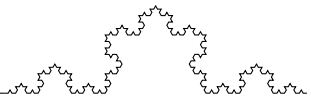
Calculate the error, relative error and the number of significant digits in the following approximations $x_A \approx x_T$.

a) $x_A = 6435.4012, \quad x_T = 6435.401163;$

b) $x_A = 0.007245, \quad x_T = 0.00723816;$

c) $x_A = 355/113, \quad x_T = \pi;$

a) $x_A = 2.236, \quad x_T = \sqrt{5}.$



Problem 1.6
Avoid loss-of-significance errors in the following formulas

$a)$ $\log(x) - \log(x-1)$ for large values of x;
$b)$ $\dfrac{e^x - 1}{x}$ for small values of x;
$c)$ $\cos(x+a) - \cos(a)$ for small values of x;

Problem 1.7
In the following function evaluations $f(x_A)$, assume the numbers x_A are correctly rounded to the number of digits shown. Bound the error $f(x_T) - f(x_A)$ and the relative error $Rel(x_A)$:

$a)$ $\sin(0.521)$;
$b)$ $e^{3.22}$;
$c)$ $\sqrt{0.0011}$;
$d)$ $\arcsin(0.5)$.

Homework 2

Due March 18, 19:00

Problem 2.1

The **error function** (also called Gauss error function) is defined by

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt.$$

Use Taylor series to approximate $\operatorname{erf}(x)$ with a polynomial $T_n(x)$. What is n, if the desired accuracy is 10^{-5}? Using this approximation, plot the graph of $\operatorname{erf}(x)$ on $[-3, 3]$.

Problem 2.2

Consider the sequence of Fibonacci numbers F_n:

$$F_0 = 1, \quad F_1 = 1, \quad F_n = F_{n-1} + F_{n-2}, \quad n = 2, 3, \ldots$$

Let $R_n = \frac{F_{n+1}}{F_n}$. It can be shown that

$$\lim_{n\to\infty} R_n = \frac{1+\sqrt{5}}{2} \equiv \phi,$$

which is known as **golden ratio**. Write a code that will compute numerically the first 40 terms of the sequence R_n together with errors $\phi - R_n$. In computations make sure that you are using IEEE double precision. Comment your results. What can be said on the order of convergence?

Problem 2.3

Thermistors are temperature-measuring devices based on the principle that the thermistor material exhibits a change in electrical resistance with a change in temperature. By measuring the resistance of the thermistor material, one can then determine the temperature. For a 10K3A Betatherm thermistor, the relationship between

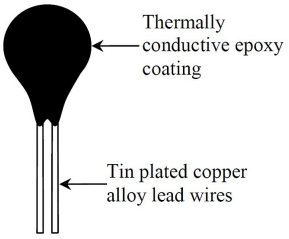


Figure 1: A typical thermistor

the resistance R of the thermistor and the temperature T is given by

$$\frac{1}{T} = 1.129241 \times 10^{-3} + 2.341077 \times 10^{-4} \log R + 8.775468 \times *10^{-8} \Big(\log R \Big)^3$$

where T is in Kelvin and R is in Ohms, and log denotes the natural logarithm. A thermistor error of no more than $\pm 0.01^o C$ is acceptable. To find the range of the resistance that is within this acceptable limit at $19^o C$, we need to solve

$$\frac{1}{19.01 + 273.15} = 1.129241 \times 10^{-3} + 2.341077 \times 10^{-4} \log R + 8.775468 \times *10^{-8} \Big(\log R \Big)^3 \tag{1}$$

$$\frac{1}{18.99 + 273.15} = 1.129241 \times 10^{-3} + 2.341077 \times 10^{-4} \log R + 8.775468 \times *10^{-8} \Big(\log R \Big)^3 \tag{2}$$

Write a computer routine implementing Newton's method and solve equations (1) and (2) using Newton's method with initial guess $R_0 = 15000$ and error tolerance of 10^{-5}.
What is the obtained range for resistance values?

Problem 2.4

Consider the function $f(x) = e^{x-\pi} + \cos x - x + \pi$.

a) Plot its graph on interval $[0, 5]$.

b) Apply Newton's method routine you developed in **Problem 2.3** to solve equation $f(x) = 0$ on interval $[0, 5]$. What can be said about its order of convergence? Argue why this is happening? How its convergence order can be improved?

c) Write a modified routine that will ensure **quadratic** convergence and apply it.

d) Instead of solving $f(x) = 0$, try to apply the fixed point iterations $x_{n+1} = e^{x_n - \pi} + \cos x_n + \pi$. Comment on your results.

Problem 2.5

a) Compute the fixed point $x_{n+1} = \cos x_n - 1 + x_n$ with initial guess $x_0 = 0.1$.

b) What can be said about the speed of convergence? Compare it with bisection method.

c) Write a modified computer routine that will speed up the convergence.

Problem 2.6

Newton's method is used to find the root α of $f(x) = 0$. The first 10 iterates are shown in the table below.

(1) What can be said about the order of convergence? Is it slower or faster than bisection method?
(2) What can be said about the root α to explain this convergence?
(3) Knowing function $f(x)$, how would you speed up the convergence?

n	x_n	$x_n - x_{n-1}$
0	2.0	
1	2.1248	0.124834
2	2.2148	0.089944
3	2.2805	0.065698
4	2.3289	0.048386
5	2.3647	0.035827
6	2.3913	0.026624
7	2.4111	0.019835
8	2.4260	0.014803
9	2.4370	0.011062
10	2.4453	0.0082745

Problem 2.7

For solving the equation $x + \ln x = 0$, there were proposed three methods:

$$
\begin{aligned}
(a) \quad x &= -\ln x \\
(b) \quad x &= e^{-x} \\
(c) \quad x &= \frac{x + e^{-x}}{2}
\end{aligned}
$$

(1) Which of the formulas can be used?
(2) Which of the formulas should be used?
(3) Give an even better formula!

Problem 2.8

Consider the following table of iterates from an iteration method which is convergent to a fixed point α of the function $g(x)$:

n	x_n	$x_n - x_{n-1}$
0	1.00	
1	0.36788	$-6.3212E-01$
2	0.69220	$3.2432E-01$
3	0.50047	$-1.9173E-01$
4	0.60624	$1.0577E-01$
5	0.54540	$-6.0848E-02$
6	0.57961	$3.4217E-02$

(1) Show that this is a linearly convergent iteration method.
(2) Find its rate of linear convergence. Is this method faster or slower than bisection method?
(3) Propose a way to accelerate the convergence of this method?

Problem 2.9

BONUS. Benout B. Mandelbrot, a famous mathematician is known as the inventor of *fractals* and this problem is dedicated to him. In this problem you will generate the so-called **quadratic Julia Sets**, a well-known fractal example.

Given two complex numbers, c and z_0 the following recursion (it is similar to fixed point iterations) is defined

$$z_n = z_{n-1}^2 + c.$$

For an arbitrary given choice of c and z_0, this recursion leads to a sequence of complex numbers z_1, z_2, z_3,... called the **orbit** of z_0. Depending on the exact choice of c and z_0, a large range of orbit patterns are possible.

For a given fixed c, most choices of z_0 yield orbits that tend towards infinity. (That is, $|z_n| \to \infty$ as $n \to \infty$).

For some values of c certain choices of z_0 yield orbits that eventually go into a periodic loop. Finally, some starting values yield orbits that appear to dance around the complex plane, apparently at random (an example of chaos). These initial values of z_0 make up the Julia set of this recursion, denoted by J_c.

Write a MATLAB/GNU Octave/Python script that visualizes a slightly different set, called the filled-in Julia set denoted by K_c, which is the set of all z_0 with orbits which do not tend towards infinity. The "normal" Julia set J_c is the edge of the filled-in Julia set. The figure below illustrates a filled-in Julia Set for one particular value of c.

a) It has been shown that if $|z_n| > 2$ for some n, then it is guaranteed that the orbit will tend to infinity. The value of n for which this becomes true is called the **escape velocity** of a particular z_0. Write a function that returns the escape velocity of a given z_0 and c. The function declaration should be: $n =$`EscVel`(z_0, c, N), where N is the maximum allowed escape velocity (i.e. if $|z_n| \leq 2$ for $n < N$, return N as the escape velocity, so you will prevent infinite loops).

b) To generate the filled-in Julia Set, write the following function

$$M = \texttt{JuliaSet}(z_{Max}, c, N),$$

where z_{Max} will be the maximum of the real and imaginary parts of the various values of z_0 for which we will compute escape velocities, c and N are the same as defined above, and M is the matrix that contains the escape velocity of various z_0.

- In this function, you first want to make a 500×500 matrix that contains complex numbers with real part between $-z_{Max}$ and z_{Max}, and imaginary part between $-z_{Max}$ and z_{Max}. Call this matrix Z. Make the imaginary part vary along the y-axis of this matrix. You can most easily do this by using `linspace` and `meshgrid` commands from MATLAB/GNU Octave, but you can also do it with a loop.
- For each element of Z, compute the escape velocity (by calling your `EscVel` function) and store it in the same location in a matrix M. When done, the matrix M should be the same size as Z and contain escape velocities with values between 1 and N.
- Run your `JuliaSet` function with various z_{Max}, c and N values to generate various fractals. To display the fractal nicely, use `imagesc` command to visualize $\arctan(0.1 * M)$, (taking the arctangent of M makes the image look nicer, you also can use `axisxy` command so that y values aren't flipped).

The figure below shows a Julia Set for $c = -.297491 + i * 0.641051$.

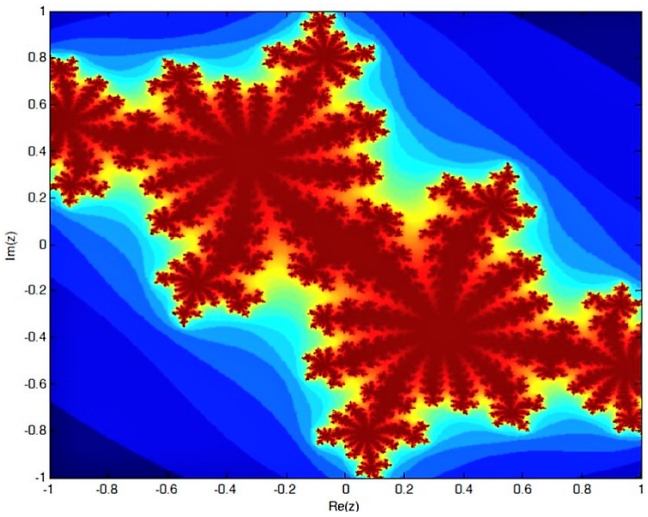


Figure 2: $M =$`JuliaSet`$(1, -.297491 + i * 0.641051, 100)$

Julia Sets are the boundaries of more general Mandelbrot sets. There is a chapter in Clive Moler textbook "Experiments with MATLAB" `www.mathworks.com/moler/exm/chapters.html` dedicated to Mandelbrot sets. The difference between Julia Sets and Mandelbrot set is presented in `www.karlsims.com/julia.html`, the figure below was taken from there.

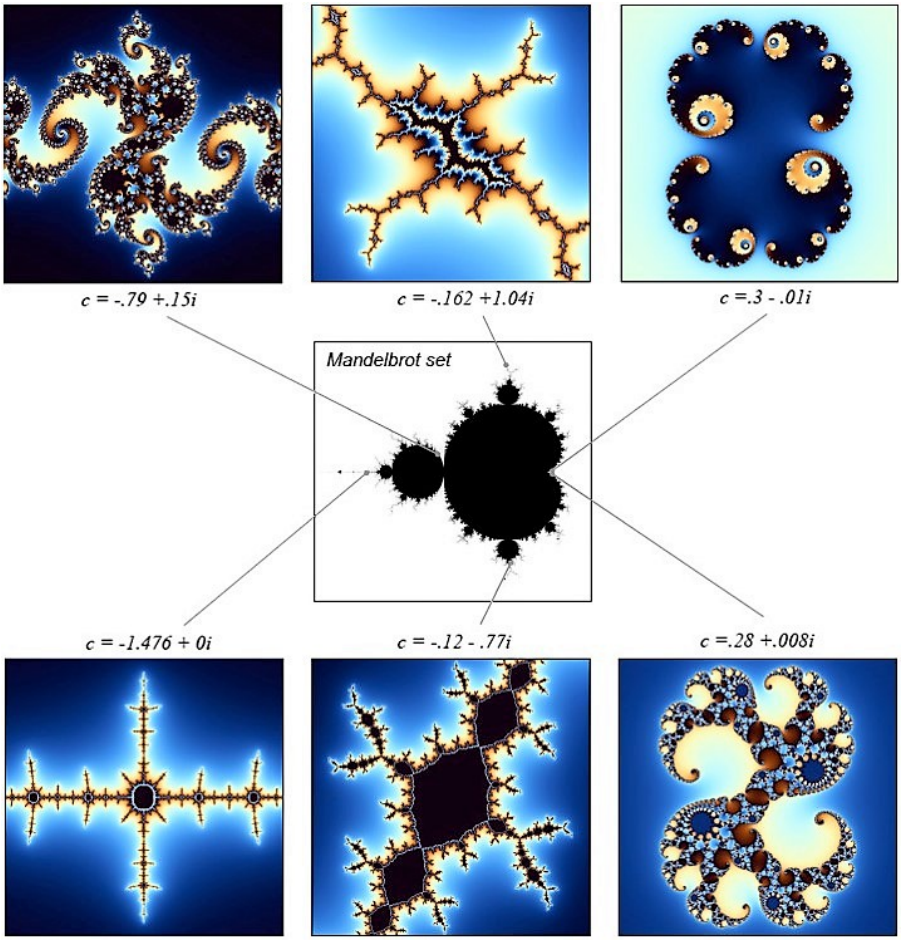


Figure 3: Different Julia Sets and their relation to Mandelbrot Set.

Homework 3

Due April 1, 19:00

Problem 3.1

A robot arm with a rapid laser is used to do a quick quality check, such as the radius of hole, on six holes located on a rectangular plate. Coordinates of the centers of the holes are given below

x	2.0	4.5	5.25	7.81	9.2	10.6
y	7.2	7.1	6.0	5.0	3.5	5.0

The path of the robot going from one point to another point needs to be smooth so as to avoid sharp jerks in the arm that can otherwise create premature wear and tear of the robot arm.

a) One such path can be the interpolating polynomial $P_5(x)$ that passes those points. Find it.

b) Other paths can be obtained by cubic spline interpolation. Find several cubic splines (see Answers to homework 3) interpolating these data, plot them together with $P_5(x)$ and propose the one with the shortest path.

Problem 3.2

The gamma function is defined as

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, dt$$

and it can be shown (integrate by parts) that $\Gamma(n) = (n-1)!\ \forall n \in \mathbb{N}$.
a) Write a computer programm that will calculate Newton's divided differences and then find (using Newton's formula) the interpolating polynomial $P_4(x)$ of the Γ function at the following points:

n	1	2	3	4	5
$\Gamma(n)$	1	1	2	6	24

b) Find a natural cubic spline $S(x)$ (using an available spline routine) that interpolates the same data.

c) Another approximation can be obtained by first calculating the polynomial $Q_4(x)$ that interpolates points $(n, \log\Gamma(n))$:

n	1	2	3	4	5
$\log\Gamma(n)$	0	0	$\log 2$	$\log 6$	$\log 24$

(here log denotes the natural logarithm) and then, consider $q(x) = e^{Q_4(x)}$.

d) Plot in the same figure the graphs of $\Gamma(x)$, $P_4(x)$, $S(x)$ and $q(x)$ on $[1, 5]$.

e) Compute with accuracy of at least 3 significant digits

$$\max_{x\in[1,5]} \Big|\Gamma(x) - P_4(x)\Big|, \quad \max_{x\in[1,5]} \Big|\Gamma(x) - S(x)\Big| \text{and} \quad \max_{x\in[1,5]} \Big|\Gamma(x) - q(x)\Big|.$$

Which of the 3 approximations is more accurate on $[1, 5]$?

HINTS. In MATLAB or GNU Octave the Γ function is known as `gamma`. In Python it is available in the `scipy.special` library. In order to complete d) you can compute the maximum from the graph, just make sure that sampling is done at sufficiently dense points that will ensure the needed accuracy.

Problem 3.3

Consider function $f(x) = \sqrt{x+1}$ on interval $[-1, 1]$. Find the near minimax polynomial approximation of degree 7 m_7 for this function. Also, find the polynomial interpolant $P_7(x)$ on evenly spaced points. Plot together function $f(x), m_7(x)$ and $P_7(x)$.

Homework 3

Due April 1, 19:00

Problem 3.1

A robot arm with a rapid laser is used to do a quick quality check, such as the radius of hole, on six holes located on a rectangular plate. Coordinates of the centers of the holes are given below

x	2.0	4.5	5.25	7.81	9.2	10.6
y	7.2	7.1	6.0	5.0	3.5	5.0

The path of the robot going from one point to another point needs to be smooth so as to avoid sharp jerks in the arm that can otherwise create premature wear and tear of the robot arm.

a) One such path can be the interpolating polynomial $P_5(x)$ that passes those points. Find it.

b) Other paths can be obtained by cubic spline interpolation. Find several cubic splines (see Answers to homework 3) interpolating these data, plot them together with $P_5(x)$ and propose the one with the shortest path.

Problem 3.2

The gamma function is defined as

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, dt$$

and it can be shown (integrate by parts) that $\Gamma(n) = (n-1)!\ \forall n \in \mathbb{N}$.
a) Write a computer programm that will calculate Newton's divided differences and then find (using Newton's formula) the interpolating polynomial $P_4(x)$ of the Γ function at the following points:

n	1	2	3	4	5
$\Gamma(n)$	1	1	2	6	24

b) Find a natural cubic spline $S(x)$ (using an available spline routine) that interpolates the same data.

c) Another approximation can be obtained by first calculating the polynomial $Q_4(x)$ that interpolates points $(n, \log\Gamma(n))$:

n	1	2	3	4	5
$\log\Gamma(n)$	0	0	$\log 2$	$\log 6$	$\log 24$

(here log denotes the natural logarithm) and then, consider $q(x) = e^{Q_4(x)}$.

d) Plot in the same figure the graphs of $\Gamma(x)$, $P_4(x)$, $S(x)$ and $q(x)$ on $[1, 5]$.

e) Compute with accuracy of at least 3 significant digits

$$\max_{x\in[1,5]} \Big|\Gamma(x) - P_4(x)\Big|, \quad \max_{x\in[1,5]} \Big|\Gamma(x) - S(x)\Big| \text{and} \quad \max_{x\in[1,5]} \Big|\Gamma(x) - q(x)\Big|.$$

Which of the 3 approximations is more accurate on $[1, 5]$?

HINTS. In MATLAB or GNU Octave the Γ function is known as `gamma`. In Python it is available in the `scipy.special` library. In order to complete d) you can compute the maximum from the graph, just make sure that sampling is done at sufficiently dense points that will ensure the needed accuracy.

Problem 3.3

Consider function $f(x) = \sqrt{x+1}$ on interval $[-1, 1]$. Find the near minimax polynomial approximation of degree 7 m_7 for this function. Also, find the polynomial interpolant $P_7(x)$ on evenly spaced points. Plot together function $f(x), m_7(x)$ and $P_7(x)$.

Practice problems 2

Problem 2.1
Let the interval used in the bisection method have the length $b - a = 3$. Find the number of midpoints c_n that must be calculated with the bisection method to obtain an approximate root within an error tolerance of 10^{-9}.

Problem 2.2
Imagine you are finding a root α satisfying $1 < \alpha < 2$. If you are using a binary computer with m digits in its significand, what is the smallest error tolerance that makes sense in finding an approximation to α? If the original interval is $[1, 2]$ how many halving are needed to find an approximation to α with the maximum accuracy possible for this computer?

Problem 2.3
Work out what the Newton iteration is for $f(x) = x^2$. What is the solution to $f(x) = 0$? Will the sequence generated by Newton method converge to solution? How quickly? Relate this to the theory of Newton method.

Problem 2.4
On most computers, the computation of $\sqrt{a}$ is based on Newton's method. Set up the Newton's iteration for solving $x^2 - a = 0$, and show that it can be written in the form

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right), \qquad n \geq 0.$$

Derive the error and relative error formulas:

$$\begin{aligned} \sqrt{a} - x_{n+1} &= -\frac{1}{2x_n}\left(\sqrt{a} - x_n\right)^2, \\ \text{Rel}(x_{n+1}) &= -\frac{\sqrt{a}}{2x_n}\left(\text{Rel}(x_n)\right)^2. \end{aligned}$$

For initial guess x_0 near $\sqrt{a}$, the last formula becomes

$$\text{Rel}(x_{n+1}) \approx -\frac{1}{2}\left(\text{Rel}(x_n)\right)^2$$

Assuming $\text{Rel}(x_0) = 0.1$, use this formula to estimate the relative error in $x_i, i = 1, 2, 3, 4$.

Problem 2.5
Derive formula

$$\text{Rel}(x_{n+1}) = \left(\text{Rel}(x_n)\right)^2$$

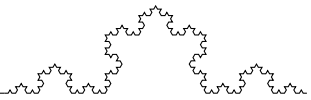
for the Newton's iterations used in computing $\frac{1}{b}$ for given b (formula was discussed in class without proof).

Problem 2.6
How many solutions are there to the equation $x = e^{-x}$? Will the iteration $x_{n+1} = e^{-x_n}$ converge for a suitable choice of x_0? Use Aitken extrapolation formula to estimate the error $\alpha - x_3$ for $x_0 = 0.57$.

Problem 2.7
The iteration

$$x_{n+1} = 2 - (1 + c)x_n + cx_n^3$$

will converge to $\alpha = 1$ for some values of c (provided that initial guess x_0 is chosen sufficiently close to α). Find the values of c for which convergence occurs. For what values of c, if any, convergence will be quadratic?



Problem 2.8

Consider the equation

$$x^7 - 28x^6 + 322x^5 - 1960x^4 + 6769x^3 - 13132x^2 - 5040 = 0$$

Change the coefficient of x^4 from -1960 to -1960.14. What is relative perturbation error in the coefficient of x^4? Calculate $\alpha(\varepsilon)$ for $\alpha(0) = 3$ and $\alpha(0) = 5$.

Problem 2.9

What is the order of convergence of the iteration

$$x_{n+1} = \frac{x_n(x_n^2 + 15)}{3x_n^2 + 5}$$

as it converges to the fixed point $\alpha = \sqrt{5}$?

Problem 2.10

Newton's method is used to find the root of $f(x) = 0$. The first few iterates are shown in the following table, giving a very slow speed of convergence. What can be said about the root α to explain the convergence? Knowing $f(x)$, how would you find an accurate value for α?

n	x_n	$x_{n-1} - x_n$
0	0.75	
1	0.752710	0.00271
2	0.754795	0.00208
3	0.756368	0.00157
4	0.757552	0.00118
5	0.758441	0.000889

Problem 2.11

Consider the following table of iterates from an iteration method which is convergent to a fixed point α of the function $g(x)$:

n	x_n	$x_n - x_{n-1}$
0	1.30499998	
1	1.25340617	$-5.159E-2$
2	1.21676284	$-3.664E-2$
3	1.19087998	$-2.588E-2$
4	1.17257320	$-1.831E-2$
5	1.15962919	$-1.294E-2$

(a) Does this appear to be a linearly convergent iteration method? If so, then estimate the rate of linear convergence. (b) Estimate the error in x_5 . (c) Give an improved estimate of α .

Practice set 2

ANSWERS

Problem 2.1

Use the error formula for bisection method to get:

$$\begin{aligned} |\alpha - c_n| \leq \frac{3}{2^n} \leq \varepsilon = 10^{-9} \\ \Leftrightarrow n \geq \frac{\ln\left(\frac{3}{10^{-9}}\right)}{\ln 2} \approx 31.48. \end{aligned}$$

Therefore $n \geq 32$.

Problem 2.2

The smallest error tolerance that makes sense (since the root is between 1 and 2, no subnormal numbers are used) is the machine epsilon, in this case $\varepsilon = 2^{-m}$. Since we should have

$$\begin{aligned} n &\geq \frac{\ln\left(\frac{1}{2^{-m}}\right)}{\ln 2} \\ &= \frac{\ln\left(2^m\right)}{\ln 2} \\ &= m, \end{aligned}$$

clearly the number of halvings will be m.

Problem 2.3

Solution, obviously is $\alpha = 0$. Newton's method is

$$\begin{aligned} x_{n+1} &= x_n - \frac{x_n^2}{2x_n} \\ &= \frac{1}{2}x_n = \frac{1}{2}\left(\frac{1}{2}x_{n-1}\right) = \frac{1}{2^2}\left(\frac{1}{2}x_{n-2}\right) \\ &= \dots \\ &= \frac{x_0}{2^{n+1}}, \end{aligned}$$

where x_0 is the initial guess. It can be seen that $x_{n+1} \to 0$ for any value of x_0, in other words, the Newton's method will converge, no matter what initial guess is chosen. In order to see the speed of convergence, first we can observe a similar behavior to the behavior of the bisection method (look for the error formula for bisection method).
Therefore, a linear convergence will be expected. Actually, we can easily derive a formula for the error in this case:

$$\begin{aligned} x_{n+1} - \alpha &= x_{n+1} \\ &= \frac{1}{2}x_n \\ &= \frac{1}{2}(x_n - \alpha) \end{aligned}$$

Thus, we have a linear convergence with linear rate $\frac{1}{2}$.
Theoretically, the convergence for Newton's method should be quadratic, but in this case the convergence speed is lower. (Because of the multiplicity of the root!)

Problem 2.4

Let a be given and set $\alpha = \sqrt{a}$ and $f(x) = x^2 - a$. Then for any $n \geq 0$ we have

$$\begin{aligned} x_{n+1} &= x_n - \frac{f(x_n)}{f'(x_n)} \\ &= x_n - \frac{x_n^2 - a}{2x_n} \\ &= \frac{1}{2}\left(x_n + \frac{a}{x_n}\right). \end{aligned}$$

In class we have shown the formula

$$\begin{aligned} \sqrt{a} - x_{n+1} &= -\frac{(\sqrt{a} - x_n)^2}{2} \cdot \frac{f"(c_n)}{f'(x_n)} \\ &= -\frac{(\sqrt{a} - x_n)^2}{2} \cdot \frac{2}{2x_n} \\ &= -\frac{(\sqrt{a} - x_n)^2}{2x_n}. \\ \mathrm{Rel}(x_{n+1}) &= \frac{\sqrt{a} - x_{n+1}}{\sqrt{a}} \\ &= \frac{-\frac{(\sqrt{a} - x_n)^2}{2x_n}}{\sqrt{a}} \\ &= -\frac{(\sqrt{a} - x_n)^2 \cdot \sqrt{a}}{2x_n (\sqrt{a})^2} \\ &= -\frac{\sqrt{a}}{2x_n}\left(\frac{\sqrt{a} - x_n}{\sqrt{a}}\right)^2 \\ &= -\frac{\sqrt{a}}{2x_n}(\mathrm{Rel}(x_n))^2. \end{aligned}$$

For initial guess x_0 near $\sqrt{a}$, the method was shown to converge and therefore $x_n \to \sqrt{a}$. Thus for n big enough, $x_n \approx \sqrt{a}$ and the last formula becomes

$$\mathrm{Rel}(x_{n+1}) \approx -\frac{1}{2}(\mathrm{Rel}(x_n))^2$$

Let $\mathrm{Rel}(x_0) = 0.1$, Then

$$\begin{aligned} \mathrm{Rel}(x_1) &\approx -\frac{1}{2}(\mathrm{Rel}(x_0))^2 = -0.005; \\ \mathrm{Rel}(x_2) &\approx -\frac{1}{2}(\mathrm{Rel}(x_1))^2 \approx -1.25 \cdot 10^{-5}; \\ \mathrm{Rel}(x_3) &\approx -\frac{1}{2}(\mathrm{Rel}(x_2))^2 \approx -7.8125 \cdot 10^{-11}; \\ \mathrm{Rel}(x_4) &\approx -\frac{1}{2}(\mathrm{Rel}(x_3))^2 \approx -3.0518 \cdot 10^{-21}. \end{aligned}$$

Problem 2.5

Let $f(x) = b - \frac{1}{x}$. Root is $\alpha = \frac{1}{b}$. The relative error according to its definition is

$$\begin{aligned} \mathrm{Rel}(x_{n+1}) &= \frac{\alpha - x_{n+1}}{\alpha} \\ &= \frac{\frac{1}{b} - x_{n+1}}{\frac{1}{b}} \\ &= 1 - bx_{n+1}. \end{aligned}$$

In class we showed that Newton's method for the equation $b - \frac{1}{x} = 0$ becomes

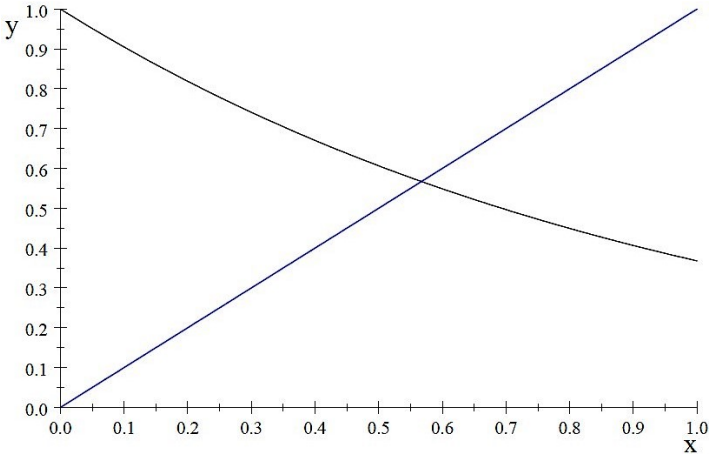
$$x_{n+1} = x_n(2 - bx_n).$$

Thus, combining the last two formulas we have

$$\begin{aligned} \mathrm{Rel}(x_{n+1}) &= 1 - bx_{n+1} \\ &= 1 - bx_n(2 - bx_n) \\ &= 1 - 2bx_n + b^2x_n^2 \\ &= (1 - bx_n)^2 \\ &= (\mathrm{Rel}(x_n))^2 \end{aligned}$$

Problem 2.6

From the graph it can be seen that we have only one root α between 0.5 and 0.7.



Let $g(x) = e^{-x} \Rightarrow g'(x) = -e^{-x}$. Since

$$\max_{x\in[0.5,0.7]} |g'(x)| = e^{-0.5} \approx 0.6065 < 1,$$

the fixed point iterates $x_{n+1} = e^{-x_n}$ will converge for $x_0 \in [0.5, 0.7]$ (Actually, any $x_0 > 0$ will do fine.). The Aitken extrapolation formula for the error $\alpha - x_3$ is

$$\alpha - x_3 \approx \frac{\lambda_3}{1 - \lambda_3}(x_3 - x_2),$$

where

$$\lambda_3 = \frac{x_3 - x_2}{x_2 - x_1}.$$

We have

i	x_i	λ_i
0	0.57	
1	$5.6553E-1$	
2	$5.6806E-1$	
3	$5.6662E-1$	$-5.6734E-1$

Thus

$$\alpha - x_3 \approx 5.2084E - 4.$$

Problem 2.7

Convergence will happen if

$$|g'(\alpha)| < 1$$

where $g(x) = 2 - (1+c)x + cx^3$ and $\alpha = 1$. Therefore, we have the following condition

$$\begin{aligned} |-(1+c)+3c| &< 1 \Leftrightarrow \\ |-1+2c| &< 1 \Leftrightarrow \\ -1 &< -1+2c < 1 \Leftrightarrow \\ 0 &< c < 1 \end{aligned}$$

Convergence will be at least quadratic, if $g'(1) = 0 \Leftrightarrow -1 + 2c = 0 \Leftrightarrow c = \frac{1}{2}$. It should be remarked that $g''(1) = 3x \Rightarrow g''(1) \neq 0$. Therefore, convergence will be exactly quadratic.

Problem 2.8

$$|\text{Rel}(-1960.14)| = \frac{-1960 - (-1960.14)}{-1960} = 7.1429E - 5$$

Consider the root $\alpha(0) = 3$. Let $g(x) = x^4$ and $f(x) = x^7 - 28x^6 + 322x^5 - 1960x^4 + 6769x^3 - 13132x^2 + 13680x - 5040$. Then $f'(x) = 7x^6 - 168x^5 + 1610x^4 - 7840x^3 + 20307x^2 - 26264x + 13680$ and $f'(3) = 660$. From the formula

$$\begin{aligned} \alpha(\varepsilon) &\approx \alpha(0) - \varepsilon \frac{g(\alpha(0))}{f'(\alpha(0))} \\ &= 3 - \varepsilon \frac{g(3)}{f'(3)} \\ &= 3 - 0.14 \cdot \frac{3^4}{660} \\ &\approx 2.9828. \end{aligned}$$

Consider the root $\alpha(0) = 5$. Then $f'(5) = 660$. Similarly,

$$\begin{aligned} \alpha(\varepsilon) &\approx 5 - \varepsilon \frac{g(5)}{f'(5)} \\ &= 5 - 0.14 \cdot \frac{5^4}{660} \\ &\approx 4.8674. \end{aligned}$$

Problem 2.9

Let

$$g(x) = \frac{x(x^2+15)}{3x^2+5}.$$

First, check that $\sqrt{5}$ is a fixed point for function g. Indeed $g(\sqrt{5}) = \sqrt{5}$. Then compute

$$\begin{aligned} g'(x) &= \frac{9x^4 - 30x^2 + 75}{(3x^2+5)^2} \\ g'(\sqrt{5}) &= 0. \end{aligned}$$

Thus, the order of convergence is at least quadratic.

$$\begin{aligned} g''(x) &= \frac{12x(-3x^4 + 31x^2 - 80)}{(3x^2+5)^4} \\ g''(\sqrt{5}) &= 0. \end{aligned}$$

Therefore, the order of convergence is at least cubic. It can be checked that

$$g^{(3)}(\sqrt{5}) \neq 0$$

So, the order of convergence is cubic.

Problem 2.10

n	x_n	$x_{n-1} - x_n$	λ_n
0	0.75		
1	0.752710	0.00271	
2	0.754795	0.00208	0.76753
3	0.756368	0.00157	0.75481
4	0.757552	0.00118	0.75159
5	0.758441	0.000889	0.75339

We can see that $\lambda_n \to 0.75 = \frac{3}{4}$. Therefore, we can say that our root have multiplicity 4. In order to find an accurate value of α, we compute the $f^{(3)}(x)$ and apply the Newton's method to this equation $f^{(3)}(x) = 0$. Since α is a simple root of $f^{(3)}(x)$,Newton′s method should converge much faster.

Problem 2.11

n	x_n	$x_n - x_{n-1}$	λ_n
0	1.30499998		
1	1.25340617	$-5.159E-2$	
2	1.21676284	$-3.664E-2$	$7.102E-1$
3	1.19087998	$-2.588E-2$	$7.063E-1$
4	1.17257320	$-1.831E-2$	$7.075E-1$
5	1.15962919	$-1.294E-2$	$7.06717E-1$

Since λ_n obviusly are converging to 0.707, we have $g'(\alpha) \approx 0.707$ and

$$\alpha - x_{n+1} \approx 0.707(\alpha - x_{n+1})$$

which means that convergence is linear with linear rate approximately 0.707. By Aitken error estimation formula we have

$$\begin{aligned} \alpha - x_5 &\approx \frac{\lambda_5}{1-\lambda_5}(x_5 - x_4) \\ &\approx \frac{0.706717}{1-0.76717} \cdot (-0.001294) \\ &\approx -3.92772E-3. \end{aligned}$$

By Aitken extrapolation formula

$$\begin{aligned} \alpha &\approx x_5 + \frac{\lambda_5}{1-\lambda_5}(x_5 - x_4) \\ &\approx 1.15962919 + (-3.92772E-3) \\ &= 1.15570147. \end{aligned}$$

Practice Set 3

ANSWERS

Problem 3.1

Consider a general cubic polynomial

$$P(x) = a + bx + cx^2 + dx^3.$$

Then

$$P'(x) = b + 2cx + 3dx^2.$$

From conditions

$$y_1 = P(0), \quad y_2 = P(1), \quad y_1' = P'(0), \quad y_2' = P'(1),$$

we get

$$\begin{aligned} y_1 &= P(0) = a, \\ y_2 &= P(1) = a + b + c + d, \\ y_1' &= P'(0) = b, \\ y_2' &= P'(1) = b + 2c + 3d, \end{aligned}$$

Solve this system with unknowns a, b, c and d

$$\begin{aligned} y_1 &= a, \\ y_2 &= a + b + c + d, \\ y_1' &= b, \\ y_2' &= b + 2c + 3d, \end{aligned}$$

and obtain solution

$$\begin{aligned} a &= y_1, \\ b &= y_1', \\ c &= -3y_1 + 3y_2 - 2y_1' - y_2', \\ d &= 2y_1 - 2y_2 + y_1' + y_2', \end{aligned}$$

Thus we have

$$\begin{aligned} P(x) &= y_1 + y_1'x + \big(-3y_1 + 3y_2 - 2y_1' - y_2'\big)x^2 + \big(2y_1 - 2y_2 + y_1' + y_2'\big)x^3 \\ &= \big(2x^3 - 3x^2 + 1\big)y_1 + \big(-2x^3 + 3x^2\big)y_2 + \big(x^3 - 2x^2 + x\big)y_1' + \big(x^3 - x^2\big)y_2' \\ &= \big(1 + 2x\big)\big(1 - x\big)^2 y_1 + x^2\big(3 - 2x\big)y_2 + x\big(1 - x\big)^2 y_1' + x^2\big(x - 1\big)y_2' \\ &= H_1(x)y_1 + H_2(x)y_2 + H_3(x)y_1' + H_4(x)y_2' \end{aligned}$$

with

$$\begin{aligned} H_1(x) &= \big(1 + 2x\big)\big(1 - x\big)^2, \\ H_2(x) &= x^2\big(3 - 2x\big), \\ H_3(x) &= x\big(1 - x\big)^2, \\ H_4(x) &= x^2\big(x - 1\big), \end{aligned}$$

Observe that

$$\begin{aligned} &H_1(0) = 1, && H_1(1) = 0, && H_1'(0) = 0, && H_1'(1) = 0, \\ &H_2(0) = 0, && H_2(1) = 1, && H_2'(0) = 0, && H_2'(1) = 0, \\ &H_3(0) = 0, && H_3(1) = 0, && H_3'(0) = 1, && H_3'(1) = 0, \\ &H_4(0) = 0, && H_4(1) = 0, && H_4'(0) = 0, && H_4'(1) = 1, \end{aligned}$$

Problem 3.2

Look for function $Q_1(x) = a + b\cos(\pi x) + c\sin(\pi x)$ such that

$$\begin{cases} a + b\cos(\pi \cdot 0) + c\sin(\pi \cdot 0) = Q_1(0) = 2 \\ a + b\cos(\pi \cdot \frac{1}{2}) + c\sin(\pi \cdot \frac{1}{2}) = Q_(\frac{1}{2}) = 5 \\ a + b\cos(\pi \cdot 1) + c\sin(\pi \cdot 1) = Q_1(1) = 4 \end{cases}$$

This leads to the following system

$$\begin{cases} a + b = 2 \\ a + c = 5 \\ a - b = 4 \end{cases}$$

that has solution $a = 3$, $b = -1$, $c = 2$. Therefore the function is $Q_1(x) = 3 - \cos(\pi x) + 2\sin(\pi x)$.

In order to find the quadratic interpolation polynomial use Newton's divided difference formula. First, need to compute Newton's divided differences:

x	y	D_1	D_2
0	2	6	–8
0.5	5	–2	
1	4		

And interpolating polynomial is $P_2(x) = 2 + 6x - 8x(x - \frac{1}{2}) = -8x^2 + 10x + 2$.

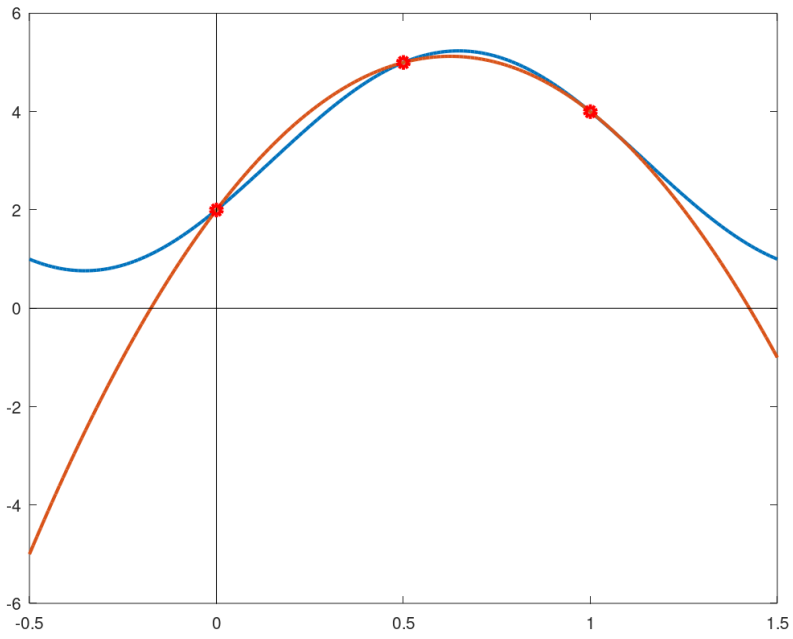


Figure 1: Graphs of quadratic interpolant $P_2(x) = -8x^2 + 10x + 2$ (red) and trigonometric interpolant $Q_1(x) = 3 - \cos(\pi x) + 2\sin(\pi x)$ (blue).

Problem 3.3

Use Newton's divided difference formula

x	y	D_1	D_2	D_3
0	–1	5	–7/2	13/15
1	4	–2	5/6	
2	2	4/3		
5	6			

Interpolating cubic polynomial is $P_3(x) = -1 + 5x - \frac{7}{2}x(x-1) + \frac{13}{15}x(x-1)(x-2)$.

Problem 3.4

Observe that data are satisfying relation $y = x^2$, i.e. interpolating points are located on parabola $y = x^2$. Since interpolation polynomial is unique, the answer is $P(x) = x^2$.

Problem 3.5

Let $N + 1$ interpolation points be

$$x_0 < x_1 < x_2 < \cdots < x_{N-1} < x_N$$

and let $L_i(x)$, $i = 0, 1, \ldots, N$ be $N + 1$ associated Lagrange basis functions. Need to prove that

$$\sum_{i=0}^{N} L_i(x) = 1. \tag{1}$$

Proof.
Rewrite the left side of the identity (1) in the form

$$1 \cdot L_0(x) + 1 \cdot L_1(x) + 1 \cdot L_2(x) + \cdots + 1 \cdot L_N(x) \tag{2}$$

and compare it with the formula for interpolation polynomial for the data (x_i, y_i), $i = 0, 1, \ldots, N$:

$$y_0 \cdot L_0(x) + y_1 \cdot L_1(x) + y_2 \cdot L_2(x) + \cdots + y_N \cdot L_N(x).$$

Obviuosly (2) represents the interpolation polynomial that interpolates points $\{(x_0, 1), (x_1, 1), (x_2, 1), \ldots, (x_N, 1)\}$. These points lie on the line $y = 1$, therfore, since interpolation polynomial is unique, it follows that interpolation polynomial given by (2) is identical function 1. So

$$1 \cdot L_0(x) + 1 \cdot L_1(x) + 1 \cdot L_2(x) + \cdots + 1 \cdot L_N(x) \equiv 1.$$

and identity (1) is proved. ■

Problem 3.6

Let $P_2(x)$ be the quadratic interpolation polynomial of function e^{-x^2} at the points $x_0 = 0$, $x_1 = \frac{1}{2}$ and $x_2 = 1$. The interpolation error is given by

$$e^{-x^2} - P_2(x) = \frac{x(x - \frac{1}{2})(x - 1)}{6} \left(e^{-x^2}\right)'''(\theta) \tag{3}$$

for some $\theta \in [0, 1]$. Let $h = \frac{1}{2}$. Using error formula (3) we get

$$\begin{aligned} \left|e^{-x^2} - P_2(x)\right| &= \frac{\left|x(x-h)(x-2h)\right|}{6} \left|\left(e^{-x^2}\right)'''(\theta)\right| \\ &\leq \frac{1}{6} \cdot \max_{x\in[0,1]} \left|x(x-h)(x-2h)\right| \cdot \max_{x\in[0,1]} \left|\left(e^{-x^2}\right)'''(x)\right| \end{aligned} \tag{4}$$

From Lecture 9 (pages $8 - 10$) we know that

$$\max_{x_0 \leq x \leq x_2} \left|(x - x_0)(x - x_1)(x - x_2)\right| = \frac{2h^3}{3\sqrt{3}}, \text{ with } h = x_1 - x_0 = x_2 - x_1.$$

Thus,

$$\max_{x\in[0,1]} \left|x(x-h)(x-2h)\right| = \frac{2 \cdot \frac{1}{2^3}}{3\sqrt{3}} = \frac{1}{12\sqrt{3}} \approx 0.048113 \tag{5}$$

Next compute the 3rd derivative of e^{-x^2}:

$$\begin{aligned} \left(e^{-x^2}\right)' &= -2xe^{-x^2} \\ \left(e^{-x^2}\right)'' &= (4x^2 - 2)e^{-x^2} \\ \left(e^{-x^2}\right)''' &= (-8x^3 + 12x)e^{-x^2} \end{aligned}$$

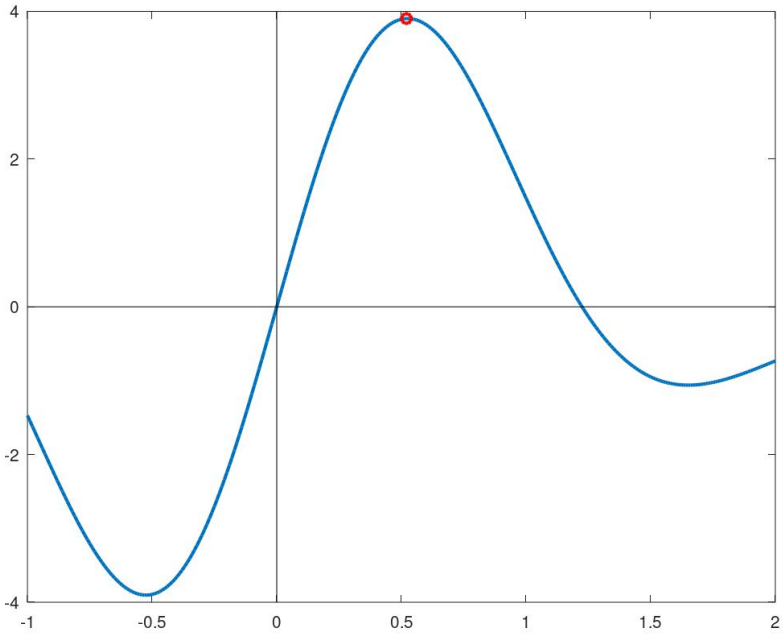


Figure 2: Graph of $(-8x^3+12x)e^{-x^2}$

In order to find the maximum of $(-8x^3+12x)e^{-x^2}$ on $[0,1]$, plot its graph.
It follows from the graph that

$$\max_{x\in[0,1]}\left|\left(e^{-x^2}\right)'''(x)\right| \approx 3.9032 \quad \text{at } x \approx 0.52 \tag{6}$$

Substituting (6) and (5) in inequaity (4) we obtain

$$\left|e^{-x^2}-P_2(x)\right| \le \frac{1}{6}\cdot 0.048113 \cdot 3.9032 \approx 0.031299 = 3.13E-2$$

Problem 3.7

In order to be a cubic spline, function $s(x)$ should have the properties (see Lecture 11):

1. $s(x)$ is a piecewise cubic polynomial;
2. $s(x)$, $s'(x)$ and $s''(x)$ should be continuous functions.

Let

$$s(x) = \begin{cases} (x-1)^3, & 0 \le x \le 1 \\ 2(x-1)^3, & 1 \le x \le 2 \end{cases}$$

Obviously condition 1 is satisfied. Then

$$s'(x) = \begin{cases} 3(x-1)^2, & 0 \le x \le 1 \\ 6(x-1)^2, & 1 \le x \le 2 \end{cases}$$

and

$$s''(x) = \begin{cases} 6(x-1), & 0 \le x \le 1 \\ 12(x-1), & 1 \le x \le 2 \end{cases}$$

It can be easily checked that $s_-(1)=0=s_+(1)$, $s'_-(1)=0=s'_+(1)$ and $s''_-(1)=0=s''_+(1)$. Thus, condition 2 is also satisfied. Therefore the function $s(x)$ is a cubic spline. Moreover, since $s''(0)=-6\neq 0$ and $s''(2)=12\neq 0$, this is not a "natural" cubic spline.

Problem 3.8

(a) Piecewise linear interpolant $P_{1,p}(x)$ is

$$P_{1,p}(x) = \begin{cases} \frac{1}{2}x, & 0 \le x \le \frac{1}{2} \\ \frac{3}{2}x - \frac{1}{2}, & \frac{1}{2} \le x \le 1 \\ -2x + 3, & 1 \le x \le 2 \\ -1, & 2 \le x \le 3 \end{cases}$$

(b) Use Newton's divided differences to find piecewise quadratic interpolant $P_{2,p}(x)$.

x	y	D_1	D_2
0	0	1/2	1
1/2	1/4	3/2	
1	1		

x	y	D_1	D_2
1	1	–2	1
2	–1	0	
3	–1		

On $\{0, 1/2, 1\}$ quadratic polynomial is $0 + 1/2(x-0) + 1(x-0)(x-1/2) = x^2$
and on $\{1, 2, 3\}$ quadratic polynomial is $1 + (-2)(x-1) + 1(x-1)(x-2) = x^2 - 5x + 5$.
Therefore, piecewise quadratic interpolant is:

$$P_{1,p}(x) = \begin{cases} x^2, & 0 \le x \le 1 \\ x^2 - 5x + 5, & 1 \le x \le 3 \end{cases}$$

(c) Since it was not specifically required to obtain the natural cubic spline analytically, we will use GNU Octave/MATLAB built-in functions to get it.

In GNU Octave define the interpolating data and apply *csape* function:

```
>>xx=[0, 0.5, 1, 2, 3];
>>yy=[0, 0.25, 1, -1, -1];
>>pp=csape(xx, yy, 'variational')
pp =
  scalar structure containing the fields:

    form = pp

    breaks =
       0.00000   0.50000   1.00000   2.00000   3.00000

    coefs =
       1.80952   0.00000   0.04762   0.00000
      -5.04762   2.71429   1.40476   0.25000
       2.52381  -4.85714   0.33333   1.00000
      -0.90476   2.71429  -1.80952  -1.00000

    pieces =  4
    order =  4
    dim =  1
```

Matrix `coefs` contains the coefficients of piecewise cubic polynomial:
if row i of matrix `coefs` is $[a\,b\,c\,d]$ then $s(x) = a(x-x_i)^3 + b(x-x_i)^2 + c(x-x_i) + d$ if $x \in [x_i, x_{i+1}]$

Therefore, we have the following natural cubic spline function that interpolates our data.

$$sn(x) = \begin{cases} 1.80952x^3 + 0.04762x, & 0 \le x \le \frac{1}{2} \\ -5.04762(x-0.5)^3 + 2.71429(x-0.5)^2 + 1.40476(x-0.5) + 0.25, & \frac{1}{2} \le x \le 1 \\ 2.52381(x-1)^3 - 4.85714(x-1)^2 + 0.33333(x-1) + 1.0, & 1 \le x \le 2 \\ -0.90476(x-2)^3 + 2.71429(x-2)^2 - 1.80952(x-2) - 1.0, & 2 \le x \le 3 \end{cases}$$

Let's plot all three interpolants:

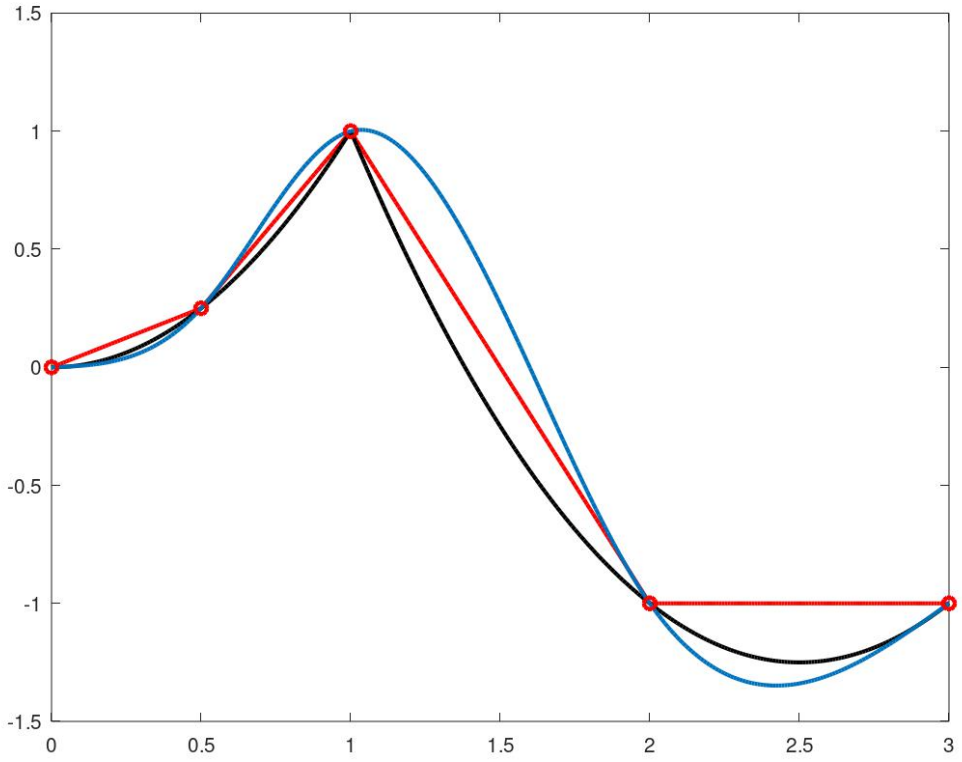


Figure 3: Graphs of piecewise linear interpolant (red), piecewise quadratic interpolant (black) and natural cubic spline interpolant (blue)

IMPORTANT REMARK:
Beware and read help/refference documentation initially, since, as it was mentioned in class lecture, there are variuous boundary conditions, and thus many types of spline functions exist. For example,

- **natural** (known in Matlab and GNU Octave as 'variational') cubic spline uses condition $s''(x_1) = s''(x_N) = 0$.
- **not-a-knot** cubic spline uses condition that $s'''(x)$ is continuous at x_2 and x_{N-1}.
- **complete** (also known as clamped) cubic spline uses condition $s'(x_1) = A$ and $s'(x_2) = B$ with values A and B provided beforehand.
- **periodic** cubic spline uses condition $s'(x_1) = s'(x_N)$.
- **second** cubic spline uses condition $s'(x_1) = C$ and $s'(x_2) = D$ with values A and B provided beforehand.

In order to obtain natural cubic spline we will use *csape* function provided by MATLAB library. For using it in GNU Octave you need to download and install additional package *splines*. Let Google be with you! I was able to do it, so will you. The available by default with GNU Octave basic distribution function *spline* can do only not-a-knot and clamped cubic splines.

As a bonus let's compare 3 different cubic spline interpolants.

- natural cubic spline obtained previously;
- clamped cubic spline (obtained using `spline` function from GNU Octave);
- not-a-knot cubic spline (aobtained using `spline` function).

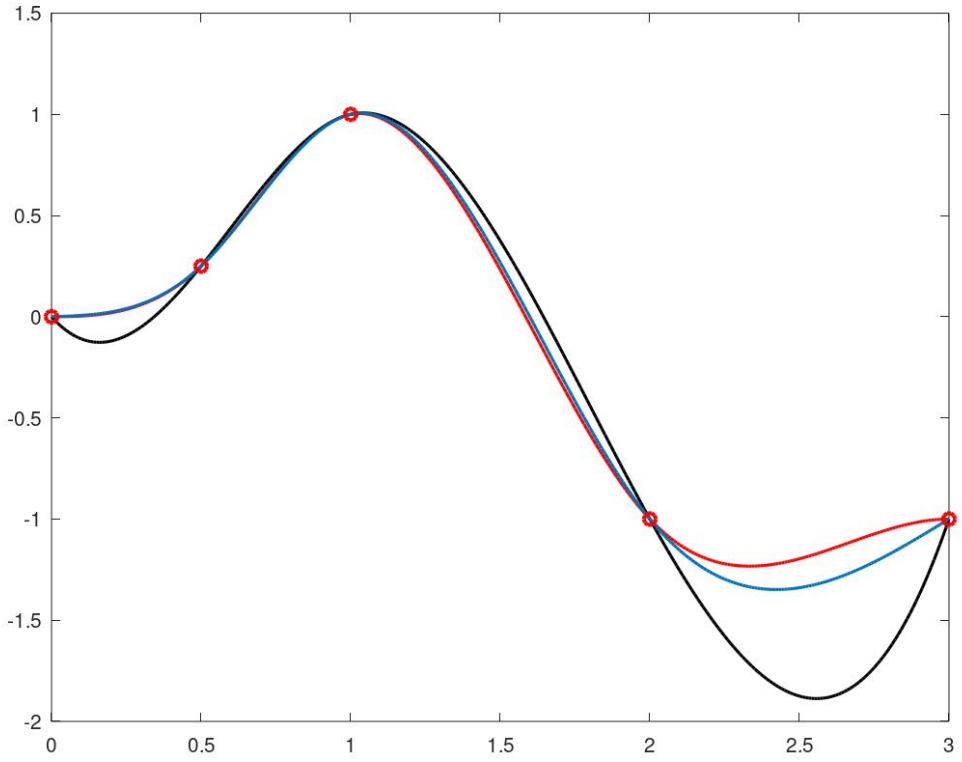


Figure 4: Natural cubic spline (blue), clamped cubic spline (red), not-a-knot cubic spline (black)

Problem 3.9
Consider the error formula for minimax approximation (see Lecture 10, page 5)

$$\rho_n(f) \leq \frac{\left(\frac{b-a}{2}\right)^{n+1}}{(n+1)!\,2^n} \max_{x\in[a,b]} \left|f^{(n+1)}(x)\right|$$

Substituting $a=-1$, $b=2$, $n=5$ we get error estimate formula

$$\rho_5(f) \leq \frac{\left(\frac{3}{2}\right)^6}{6!\,2^5} \max_{x\in[-1,2]} \left|f^{(6)}(x)\right| = \frac{3^6}{6!\,2^{11}} \max_{x\in[-1,2]} \left|f^{(6)}(x)\right|,$$

where $f(x)=e^{3x-1}$. Compute derivatives

$$\left(e^{3x-1}\right)' = 3e^{3x-1}, \quad \left(e^{3x-1}\right)'' = 9e^{3x-1}, \quad \ldots, \quad \left(e^{3x-1}\right)^{(6)} = 3^6 e^{3x-1}$$

Since

$$\max_{x\in[-1,2]} \left|f^{(6)}(x)\right| = 3^6 \max_{x\in[-1,2]} \left|e^{3x-1}\right| = 3^6 \cdot e^{3\cdot 2-1} = 3^6 e^5$$

error estiamte formula becomes

$$\rho_5(e^{3x-1}) \leq \frac{3^{12}e^5}{6!\,2^{11}} \approx 53.489$$

It seems that this is a very crude estimate, but keep in mind that fucntion e^{3x-1} on interval $[-1,2]$ grows fast from 0.018 up to 148.41. So we can repeat the same computations for $n=10$ and $n=15$ to get

$$\rho_{10}(e^{3x-1}) \leq \frac{3^{22}e^5}{11!\,2^{21}} \approx 5.564E-2, \quad \rho_{15}(e^{3x-1}) \leq \frac{3^{32}e^5}{16!\,2^{31}} \approx 6.1207E-6$$

Problem 3.10

Recall the triple recurrsion formula for Chebyshev polynomials (see Lecture10, page 8)

$$\begin{aligned} T_0(x) &= 1, \\ T_1(x) &= x, \\ T_{n+1}(x) &= (2x) \cdot T_n(x) - T_{n-1}(x), \end{aligned}$$

Evaluation of T_0 and T_1 is straightforward, then T_2 will need 2 multiplications and 1 addition, and every next polynomial will need 1 more multiplication and 1 more addition. Therefore, evaluation at a particular x of Chebyshev polynomials $T_0(x), T_1(x), T_2(x), \dots T_n(x)$ will need n multiplications and $n-1$ additions.

Problem 3.11

Let $q(x)$ be a polynomial of degree $n-1$ and consider the polynomial $x^n - q(x)$. It is a polynomial of degree n, moreover it is a monic polynomial (see Lecture 10). According to the **Theorem on minimum size property** from page 11, the degree n monic polynomial with the smallest maximum on $[-1,1]$ is the modified Chebyshev polynomial $\widetilde{T}_n(x)$, and its maximum value on $[-1,1]$ is $\frac{1}{2^{n-1}}$. Thus,

$$\max_{x\in[-1,1]} \left|x^n - q(x)\right| = \frac{1}{2^{n-1}}$$

and it is achieved for $q(x) = x^n - \widetilde{T}_n(x)$.

Problem 3.12

Proof.
Consider substitution

$$x = \cos\theta, \quad \theta = \arccos x, \quad \sqrt{1-x^2} = \sqrt{1-\cos^2\theta} = \sin\theta, \quad dx = d(\cos\theta) = -\sin\theta\, d\theta.$$

Then

$$\begin{aligned} \int_{-1}^{1} \frac{T_n(x)T_m(x)}{\sqrt{1-x^2}}\, dx &= \int_{\pi}^{0} \frac{\cos(n\theta)\cos(m\theta)(-\sin\theta)\, d\theta}{\sin\theta} \\ &= \int_0^{\pi} \cos(n\theta)\cos(m\theta)\, d\theta \\ &= 0, \text{ if } n \neq m. \end{aligned}$$

■

Practice Problems Set 3

Problem 3.1

Find a polynomial $P(x)$ of degree ≤ 3 for which

$$P(0) = y_1 \qquad P(1) = y_2$$
$$P'(0) = y_1' \qquad P'(1) = y_2'$$

with y_1, y_2, y_1', y_2' given constants.
The resulting polynomial is called **cubic Hermite interpolating polynomial**.
HINT: Write $P(x) = y_1 H_1(x) + y_2 H_2(x) + y_1' H_3(x) + y_2' H_4(x)$ with H_i cubic polynomials satisfying appropriate properties, in analogy with Lagrange interpolating polynomials.

Problem 3.2

Find the function $P(x) = a + b\cos(\pi x) + c\sin(\pi x)$, which interpolates the data

x	0	0.5	1
y	2	5	4

This is so-called **trigonometric interpolation**. Also, find the quadratic polynomial interpolating this data. In each instance, draw the graph of the interpolating function.

Problem 3.3

Find cubic polynomial interpolating the data

x	0	1	2	5
y	-1	4	2	6

Problem 3.4

Find the polynomial interpolating the data

x	-3	1	2	4	5
y	9	1	4	16	25

Problem 3.5

Prove that

$$\sum_{i=0}^{N} L_i(x) = 1,$$

where $L_i(x)$ are Lagrange basis functions associated to $N+1$ interpolation points.

Problem 3.6

Consider the polynomial interpolation of the function $f(x) = e^{-x^2}$ on $[0,1]$ at the points $x_0 = 0$, $x_1 = 0.5$ and $x_2 = 1$. Estimate the maximum of the polynomial interpolation error for $x \in [0,1]$, i.e. give an upper bound for this error.

Problem 3.7

Is the following a cubic spline on the interval $0 \leq x \leq 2$?

$$s(x) = \begin{cases} (x-1)^3, & 0 \leq x \leq 1 \\ 2(x-1)^3, & 1 \leq x \leq 2 \end{cases}$$

Problem 3.8
Consider the data

x	0	1/2	1	2	3
y	0	1/4	1	-1	-1

(a) Find the piecewise linear interpolating function for the data; (b) Find the piecewise quadratic interpolating function for the data. (c) Find the natural cubic spline that interpolates the data. Graph all three graphs for $0 \leq x \leq 3$.

Problem 3.9
Compute the error bound for the minimax approximation of the function $f(x) = e^{3x-1}$ on the $[-1, 2]$ and $n = 5$.

Problem 3.10
How many multiplications and additions are needed to compute Chebyshev polynomials $T_0(x), T_1(x), T_2(x), \ldots, T_n(x)$ for a particular value of x?

Problem 3.11
Let $q(x)$ be a polynomial of degree $\leq n-1$, and consider

$$\max_{-1 \leq x \leq 1} |x^n - q(x)|$$

What is the smallest possible value for this quantity? Solve for the $q(x)$ for which this value is attained.

Problem 3.12
For $n, m \geq 0$ and $n \neq m$ show

$$\int_{-1}^{1} \frac{T_n(x) T_m(x)}{\sqrt{1-x^2}} dx = 0$$

This is called the orthogonality property for the Chebyshev polynomials.

Homework 1

Problem 1.1

Let d_1d_2/m_1m_2 be your birthday. Find the binary single precision IEEE floating-point expression for the number $(d_1m_2)_{10}$. Also, find the binary double precision IEEE floating-point expression for the number $(d_2m_1)_{10}$. In both cases, specify significand, exponent and σ. Convert $(111\ldots1)_2$ to decimal form with the parentheses enclosing $(21 - m_1m_2)$ ones.

Problem 1.2

Some microcomputers in the past used a binary floating-point format with 8 bits for the exponent and 1 bit for the sign σ. The significand contained 31 bits, with no hiding of the leading bit 1. The arithmetic used rounding. To determine the accuracy of the representation, find the machine epsilon, integer M, and the largest number that can be represented exactly in this floating-point format. Also, find the accuracy of the rounding operation.

Problem 1.3

Consider a binary floating-point representation with significand containing 3 digits without hiding the leading 1 and $-1_{10} \leq e \leq 3_{10}$. List all numbers that can be stored exactly together with their decimal value. Plot these numbers on real axis. For this arithmetic, specify what are the corresponding floating-point representation of $\pi/4$ and $14/5$ if a) rounding is used; b) chopping is used?

Problem 1.4

Calculate the error, relative error and the number of significant digits in the following approximations $x_A \approx x_T$.

$a)$ $x_A = 28.271, \quad x_T = 28.254;$

$b)$ $x_A = 0.028271, \quad x_T = 0.028254;$

$c)$ $x_A = 19/7, \quad x_T = e;$

$d)$ $x_A = 1.414, \quad x_T = \sqrt{2}.$

Problem 1.5

Avoid loss-of-significance errors in the following formulas

$a)$ $\log(x+1) - \log(x)$ for large values of x;

$b)$ $\dfrac{e^x - 1}{x}$ for small values of x;

$c)$ $\sin(x+a) - \sin(a)$ for small values of x;

$d)$ $\sqrt[3]{x+1} - \sqrt[3]{x}$ for large values of x;

$e)$ $\sqrt{1 + \dfrac{1}{x}} - 1$ for large values of x.

Problem 1.6

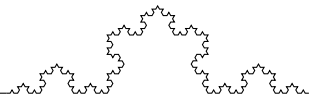
In the following function evaluations $f(x_A)$, assume the numbers x_A are correctly rounded to the number of digits shown. Bound the error $f(x_T) - f(x_A)$ and the relative error $Rel(x_A)$:

$a)$ $cos(1.473);$

$b)$ $e^{2.231};$

$c)$ $\sqrt{0.0275};$

$d)$ $\arctan(4.7869).$



Problem 1.7

Extra Bonus point Bound

$$\int_0^{\pi} \frac{t^2}{1+t^4}\,dt - \int_0^{22/7} \frac{t^2}{1+t^4}\,dt.$$

Hint: Define function

$$f(x) = \int_0^{x} \frac{t^2}{1+t^4}\,dt.$$

and then bound $f(\pi) - f(22/7)$.

Numerical Analysis

Homework 1. Answers to some problems.

Problem 1

$(2.8)_{10} = (10.1100110011001100110011001100110011\ldots)_2$
$= (1.0110011001100110011001100110011\ldots)_2 \cdot 2^1$
$\sigma = +1;\ e = 1; \overline{x} = (1.0110011001100110011001100110011\ldots)_2$

In single precision IEEE format $E = e + 127 = 128_{10} = (10000000)_2$;

1	10000000	01100110011001100110011

Double precission is done similarly.

$\underbrace{(111\ldots1)_2}_{21-8=13} = 2^{12} + 2^{11} + \ldots 2 + 2^0 = \frac{2^{13}-1}{2-1} = 2^{13} - 1 = 8191$

Problem 2 *Machine epsilon is the difference between* 1 *and the next larger number that can be stored in the given format. Therefore, in this format,*

Machine epsilon $= (1.00000\ldots.0001)_2 - 1_2$ $(29\ zeros) = (0.00000\ldots0001)_2 = 2^{-30}$

M, by definition given in class, is the largest integer having the property that any integer x smaller than M can be stored exactly in this floating-point format.

If 31 is the number of binary digits in the significand, then all integers less than or equal to $(11\ldots1)_2$ can be stored exactly, where this significand contains 31 digits, all 1. It equals to $2^{31} - 1$.

In addition, the number $2^{31} = (1.0\ldots0)_2 \cdot 2^{31}$ also stores exactly.

Note that there not enough digits to store $2^{31} + 1$, as this would require 32 binary digits in the significand.

Therefore $M = 2^{31} \approx 2.147 \cdot 10^9$.

The largest number that can be represented in this format is $(1.111\ldots111)_2 \cdot 2^e$, where significand contains 31 ones, and the exponent $e = (11111111)_2 - 127_{10} = 255 - 127 = 128_{10}$. It is approximately 10^{37}.

Since rounding is used, the bounds for ε are

$$-2^{-31} \leq \varepsilon \leq 2^{-31}$$

Problem 3 *List of numbers in decimal format:*

$0.5, 0.625, 0.75, 0.875,$
$1, 1.25, 1.5, 1.75,$
$2, 2.5,$
$3, 3.5,$
$4, 5, 6, 7, 8,$
$10, 12, 14$

$\pi/4 \approx 0.785398163$. In this arithmetic it will be stored as 0.75 if both rounding and chopping are used.

$14/5 = 2.8$ In this arithmetic it will be stored as 3 if rounding is used.and 2.5 in case of chopping.

Problem 4 *a)* $Error = 28.254 - 28.271 = -0.017; \quad Rel(x_A) = -\frac{0.017}{28.254} = -0.00060168.$

Three significant digits;
Similarly
b) Three significant digits; c) Three significant digits; d) Four significant digits.

Problem 5

$$a) \quad Use \quad \log(x+1) - \log(x) = \log \frac{x+1}{x}$$
$$b) \quad Use \quad Taylor \quad decomp. \quad for \quad e^x$$
$$c) \quad Use \ \sin(x+a) - \sin(a) = 2\cos\frac{x+2a}{2}\sin\frac{x}{2}$$
$$d) \quad Use \quad a - b = \frac{a^3 - b^3}{a^2 + ab + b^2}$$
$$e) \quad Use \quad = \frac{\left(\sqrt{1+\frac{1}{x}} - 1\right)\left(\sqrt{1+\frac{1}{x}} + 1\right)}{\sqrt{1+\frac{1}{x}} + 1} = \frac{\sqrt{x}}{\sqrt{x+1} + \sqrt{x}}$$

Problem 6 *Use formula*

$$|f(x_T) - f(x_A)| \approx |f"(x_A)| \cdot |x_T - x_A|$$

a) Since x_A was correctly rounded, $|x_T - x_A| \leq 0.0005$. Therefore, $|f(x_T) - f(x_A)| \leq |\sin 1.4713| \cdot 0.0005 \approx 4.9761 \cdot 10^{-4}$
$|rel(f(x_A))| \approx \frac{|f"(x_A)| \cdot |x_T - x_A|}{|f(x_A)|} \approx 0.0051$
b,c,d) are done similarly.

Problem 7 *Define*

$$f(x) = \int_0^x \frac{t^2}{1+t^4} dt. \quad Then \quad f'(x) = \frac{x^2}{1+x^4}$$

Thus $f(\pi) - f(22/7) \approx f'(x_A)(x_T - x_A) = f'\left(\frac{22}{7}\right)\left(\pi - \frac{22}{7}\right) = \frac{\left(\frac{22}{7}\right)^2}{1+\left(\frac{22}{7}\right)^4}\left(\pi - \frac{22}{7}\right) \approx -1.2672 \cdot 10^{-4}$.