C and C++ problems for Computational Physics

Shahnoor

March 12, 2017

Contents

Chapter 1

Series

1.1 Fibonacci sequence

C.N.1.1.1 A continued fraction representation for the **Golden Ratio** ϕ is given by (as a sequence)

$$
\begin{aligned}
x_1 &= 1 + \frac{1}{1} \\
x_2 &= 1 + \frac{1}{1 + \frac{1}{1}} \\
x_3 &= 1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{1}}} \\
&\cdots \\
\phi &= 1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{\cdots}}}}
\end{aligned}
\tag{1.1}
$$

The sequence x_n converges to $\phi = 1.6180339887498948482045868343656...$ as the number of iterations goes to infinity.

(a) Write a C or C++ function **double mygold(int n)**, that evaluates the n-th order iterated value of ϕ i.e. x_n, using the above algorithm.

(b) The Golden Ratio may be used to compute the Fibonacci numbers using the relation below:

$$f_n = \frac{1}{2\phi - 1}\left(\phi^{n+1} - (1-\phi)^{n+1}\right) \tag{1.2}$$

for $n = 2, 3, 4$

This is an amazing equation. The right-hand side involves powers and quotients of irrational numbers, but the result is a sequence of integers. Write a C or C++ function **double myfibo(int n)** that evaluates the n-th order Fibonacci number using the above algorithm and approximating to the nearest integer value.

(c) Call *myfibo(n)* in a complete C or C++ program and output Fibonacci numbers for $n \in [1, 11]$. For you comparison, the corresponding Fibonacci numbers are: $\{1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144\}$.

C.N.1.1.2 The **Fibonacci numbers** may be generated using the relation given below

$$F(n) = \frac{\phi_+^n - (-\phi_-)^n}{\sqrt{5}} \tag{1.3}$$

where

$$\phi_+ = \frac{\sqrt{5}+1}{2} \tag{1.4}$$

and,

$$\phi_- = \frac{\sqrt{5}-1}{2} \tag{1.5}$$

$$\tag{1.6}$$

Write a C++ function *myfibo(int n)* that will calculate the n-th Fibonacci number. Then use the function to calculate the first 100 Fibonacci numbers and write them in a data file.

C.N.1.1.3 Fibonacci Wonder

The golden ratio is given by

$$\phi_+ = \frac{\sqrt{5}+1}{2} \tag{1.7}$$

The golden angle is given by

$$g = \frac{360}{\phi_+^2} \tag{1.8}$$

(a) plot a graph of ratio of Fibonacci sequence ,i.e. f_{n+1}/f_n vs n where $n \in [0:25]$ to show that the ratio of Fibonacci sequence approaches the golden ratio.

(b) **The Sunflower form**

$$r_n = \sqrt{n} \tag{1.9}$$

$$\theta_n = ng \tag{1.10}$$

where r is the distance from center and $n = 0, 1, 2, \ldots$ and g is the golden angle. Using the relation below plot a graph of scattered points with $n = 500, 1000, 1500, 2000$ points

$$x_n = r\cos(\theta_n) \tag{1.11}$$

$$y_n = r\sin(\theta_n) \tag{1.12}$$

$$\tag{1.13}$$

(c) **The Spiral form**

$$x_n = a\cos(n\pi/180)\exp(nb) \tag{1.14}$$

$$y_n = a\sin(n\pi/180)\exp(nb) \tag{1.15}$$

where $n = 0, 1, 2, \ldots$. Using $a = 0.001$ and $b = \pi/90$ plot a graph of $n = 2500$ points.

The sample graph is given below

1.2 Value of Pi π

C.N.1.2.1 The **Vite's formula** for π is given by

$$\frac{2}{\pi} = \frac{\sqrt{2}}{2}\frac{\sqrt{2+\sqrt{2}}}{2}\frac{\sqrt{2+\sqrt{2+\sqrt{2}}}}{2}\ldots = \prod_{i=1}^{\infty}\frac{a_i}{2} \tag{1.16}$$

where $a_i = \sqrt{2+a_{i-1}}$ with the initial condition $a_1 = \sqrt{2}$

shahnoor3pl@gmail.com

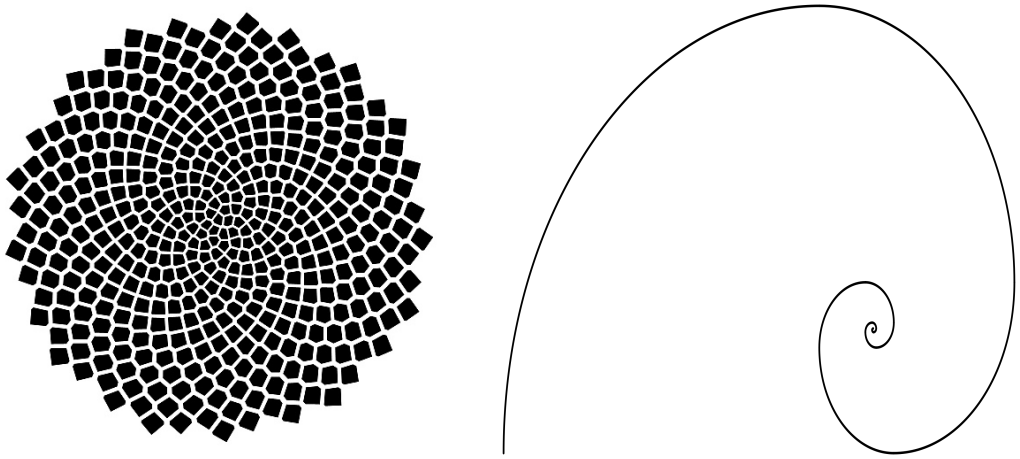


Figure 1.1: The sunflower form (left) and spiral form (right) of Fibonacci sequence

(a) Write a C or C++ function **double my2bpi(int n)**, that evaluates the value of $2/\pi$ with the first n-terms in the product using the above formula.

(b) Call the function *my2bpi(n)* in a complete C or C++ program and evaluate $x_n = my2bpi(n)$ with $n = 2, 4, 6, 8, 10, 12, 14, 16, 18, 20$ and write the values in a data file.

(c) Plot the values of $|x_n - 2/\pi|$ for different values of n using gnuplot. Save your plot as a postscript file.

C.N.1.2.2 (a) The **Vite's formula** for π is given by

$$\frac{2}{\pi} = \frac{\sqrt{2}}{2}\frac{\sqrt{2+\sqrt{2}}}{2}\frac{\sqrt{2+\sqrt{2+\sqrt{2}}}}{2}\ldots = \prod_{i=1}^{\infty}\frac{a_i}{2} \tag{1.17}$$

where $a_i = \sqrt{2 + a_{i-1}}$ with the initial condition $a_1 = \sqrt{2}$

Write a C++ function **double mypi1(int n)** that evaluates the value of π with the first n terms in the product using the above formula.

(b) The **Gauss-Legendre algorithm for calculating** π is given by

$$\begin{aligned}
a_0 &= 1, \\
b_0 &= \frac{1}{\sqrt{2}}, \\
t_0 &= 1/4, \\
p_0 &= 1, \\
a_{i+1} &= \frac{a_i + b_i}{2} \\
b_{i+1} &= \sqrt{a_i b_i} \\
t_{i+1} &= t_i - p_i(a_{i+1}^2 - b_{i+1}^2) \\
p_{i+1} &= 2p_i \\
\pi &\approx \frac{(a_{i+1} + b_{i+1})^2}{4t_{i+1}}
\end{aligned} \tag{1.18}$$

Write a C++ function **double mypi2(int n)** that evaluates the value of π with the first nth iterated value using the above formula.

(c) Among the two algorithms you used to calculate the value of π, By including the header file $< ctime >$ you can get access to predefined functions for measuring the time needed to run a process on your computer. $clock\_t$ is a data type defined in $< ctime >$. By writing $t = clock()$ anywhere in your code, you can store in the variable t the number of clock ticks elapsed since a particular moment. Use this hint to find the durations needed for finding π using Vite's formula and Gauss-Legendre formula.

C.N.1.2.3 The **Gauss-Legendre algorithm for calculating** π is given by

$$\begin{aligned} a_0 &= 1, \\ b_0 &= \frac{1}{\sqrt{2}}, \\ t_0 &= 1/4, \\ p_0 &= 1, \\ a_{i+1} &= \frac{a_i + b_i}{2} \\ b_{i+1} &= \sqrt{a_i b_i} \\ t_{i+1} &= t_i - p_i(a_{i+1}^2 - b_{i+1}^2) \\ p_{i+1} &= 2p_i \\ \pi &\approx \frac{(a_{i+1} + b_{i+1})^2}{4t_{i+1}} \end{aligned} \tag{1.19}$$

(a) Write a C++ function **double mypi(int n)** that evaluates the value of π with the first nth iterated value using the above formula.

(b) Call the function double mypi(n) in a complete C or C++ program and evaluate $x_n = mypi(n)$ with $n = 2, 4, 6, 8, 10, 12, 14, 16, 18, 20$ and write the values in a data file.

(c) Plot the values of $|x_n - \pi|$ for different values of n using gnuplot. Save your plot as a postscript file.

C.N.1.2.4 The value of pi is given by

$$\pi = \frac{\prod_{i=1}^{\infty}\left(1 + \frac{1}{4i^2-1}\right)}{\sum_{i=1}^{\infty}\left(\frac{1}{4i^2-1}\right)} \tag{1.20}$$

(a) Write a C or C++ function **double mypi(int n)**, that evaluates the value of π with the n-terms in the summation and product using the above formula.

(b) Call the function *mypi(n)* in a complete C or C++ program and evaluate $x_n = mypi(n)$ with $n = 2, 4, 6, 8, 10, 12, 14, 16, 18, 20$ and write the values in a data file.

(c) Plot the values of $|x_n - \pi|$ for different values of n using gnuplot. Save your plot as a postscript file.

1.3 Trigonometric Function

C.N.1.3.1 The **hyperbolic cosecant** or $cosech(x)$ can be expanded as

$$\operatorname{csch}(x) = \frac{1}{x} + 2x\sum_{k=1}^{\infty}\frac{(-1)^k}{x^2 + (\pi k)^2} \tag{1.21}$$

for $x \neq 0$

(a) Define a C or C++ function **double cosech(double x, int n)** that evaluates the series expansion for the Fresnel integral using the expansion up to the n-th term in the series for a given value of the arguments x and n.

(b) Plot your function *cosech(x,n)* with $n = 20$, for $x \in [0.1, 1.0]$using gnuplot and save the plot as a postscript file.

C.N.1.3.2 A continued **fraction representation for the function tangent** or $tan(x)$ (as a sequence of fractions) is given by

$$\begin{aligned} c_0 &= x \\ c_1 &= \frac{x}{1 - x^2} \\ c_2 &= \frac{x}{1 - \frac{x^2}{3 - x^2}} \\ c_3 &= \frac{x}{1 - \frac{x^2}{3 - \frac{x^2}{5 - x^2}}} \\ \tan(x) &= \frac{x}{1 - \frac{x^2}{3 - \frac{x^2}{5 - \frac{x^2}{7 - \frac{x^2}{9 - \cdots}}}}} \end{aligned} \tag{1.22}$$

(a) Write a C or C++ function **double mytan(double x,int n)**, that evaluates the n-th order iterated value of *tan(x)* using the above algorithm.

(b) Call the function **double mytan(double x, int n)** in a complete C or C++ program and evaluate **tan(x)** with $n \in [0, 20]$ and write the values in a data file.

(c) Plot the values of $|mytan(x, n) - tan(x)|$ at $x = \pi/4$, versus n in the range $[0, 20]$ using gnuplot. Save your plot as a postscript file

C.N.1.3.3 A **continued fraction representation for the function arctangent** or $tan^{-1}(x)$ (as a sequence of fractions) is given by

$$\begin{aligned} c_0 &= x \\ c_1 &= \frac{x}{1 + (1x)^2} \\ c_2 &= \frac{x}{1 + \frac{(1x)^2}{3 + (2x)^2}} \\ c_3 &= \frac{x}{1 + \frac{(1x)^2}{3 + \frac{(2x)^2}{5 + (3x)^2}}} \\ &\cdots \\ \tan^{-1}(x) &= \frac{x}{1 + \frac{(1x)^2}{3 + \frac{(2x)^2}{5 + \frac{(3x)^2}{7 + \frac{(4x)^2}{9 + \cdots}}}}} \end{aligned} \tag{1.23}$$

(a) Write a C or C++ function **double myarctan(double x,int n)**, that evaluates the n-th order iterated value of $tan^{-1}(x)$ using the above algorithm.

(b) Call the function **double myarctan(double x, int n)** in a complete C or C++ program and evaluate $tan^{-1}(x)$ with $n \in [0, 20]$ and write the values in a data file.

(c) Plot the values of $\left|myarctan(x,n) - tan^{-1}(x)\right|$ at $x = 1.0$, versus n in the range $[0, 20]$ using gnuplot. Save your plot as a postscript file.

C.N.1.3.4 The sine function has the following infinite product representation

$$\sin(x) = x\prod_{k=1}^{\infty}(1 - \frac{x^2}{k^2\pi^2}) \tag{1.24}$$

(a) Write a C or C++ function **double mysine(double x,int n)** that evaluates the sine function from the above definition using only the first n terms in the product.

(b) Plot the difference $|\sin(x) - \text{mysine(x, n)}|$ vs. n for $n \in [10, 100]$ at $x = \pi/4, \pi/2$ in the same plot.

C.N.1.3.5 The **hyperbolic cosecant** can be defied as

$$\cosh(x) = \prod_{k=1}^{n}\left(1 + \frac{4x^2}{(2k-1)^2\pi^2}\right) = \left(1 + \frac{4x^2}{\pi^2}\right)\left(1 + \frac{4x^2}{9\pi^2}\right)\ldots \tag{1.25}$$

$$\cosh(x) = \sum_{k=0}^{n}\frac{x^{2k}}{(2k)!} = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \ldots \tag{1.26}$$

Write two C or C++ functions that can evaluate these. Compare the result of above two equations.

C.N.1.3.6 The sine function has the following infinite product representation

$$\sin(x) = x\left(1 - \frac{x^2}{\pi^2}\right)\left(1 - \frac{x^2}{4\pi^2}\right)\left(1 - \frac{x^2}{9\pi^2}\right)\ldots = x\prod_{k=1}^{\infty}(1 - \frac{x^2}{k^2\pi^2}) \tag{1.27}$$

(a) Write a C or C++ function **double prodsine(double x, int n)** that evaluates the sine function from the above definition using only the first n terms in the product (i.e. $\sin(x) \approx x\prod_{k=1}^{n}(1 - \frac{x^2}{k^2\pi^2})$).

(b) Plot the difference fabs(sin(x) - prodsine(x,n)) vs. n for $n \in [10, 100]$ at $x = \pi/4, \pi/2$ in the same plot and save the plot as a postscript file.

(c) Putting $x = \pi/2$, we get the **Walli's famous formula** for $\pi/2$ as

$$\frac{\pi}{2} = \prod_{k=1}^{\infty}\left[\frac{(2k)^2}{(2k-1)(2k+1)}\right] = \frac{2.2}{1.3}\frac{4.4}{3.5}\frac{6.6}{5.7}\ldots \tag{1.28}$$

Write a C or C++ function **double Walli(int n)** that approximately evaluates $\pi/2$ using the first n terms in the product above. How many terms is required to get an accuracy for $\pi/2$ to within 1 in 10^6? Hint: Take the value of *pi* defined in the cmath or ¡math.h¿ library as exact.

1.4 Special Function

1. The **spherical Bessel function of the first kind** jn(x) is defined by the infinite series representation as:

$$j_n(x) = x^n\sum_{k=0}^{\infty}\frac{(-x^2/2)^k}{k!(2n+2k+1)!!} \tag{1.29}$$

where $k! = 1.2.3\ldots k$ is the factorial of k and $(2s+1)!!$ is the double factorial of $2s+1$ defined as $(2s+1)!! = (2s+1).(2s-1).(2s-3)\ldots 3.1$.

(a) Write a C or C++ function **double jn(double x, int n,int M)** that evaluates the spherical Bessel function of the first kind $j_n(x)$ using the above definitions and using the first $(M+1)$ terms in the infinite

shahnoor3pl@gmail.com

series. Take suitable value of M for evaluating the function $j_n(x)$. You may want to define your own factorial and double factorial functions in evaluating the series.

(b) In a complete C or C++ program write the values of x in the first column and the values of the function $j_n(x)$ in consecutive columns for different values of n. Plot, using gnuplot the functions $j_n(x)$ in the range $x \in [0, 10]$ using the above C C++ function for $n = 0, 1, 2, 3$ in the same plot. Save the two plot as postscript files.

2. The **Laguerre polynomials** $L_n(x)$ has the series representation:

$$L_n(x) = \sum_{k=0}^{n} \frac{(-1)^k n!}{(n-k)!k!k!} x^k \tag{1.30}$$

where $n = 0, 1, 2, 3, \ldots$ are integers. (a) Write a C or C++ function **double laguerren(double x, int n)** that evaluates the Laguerre polynomials $L_n(x)$ using the above definition. You will need to define your own factorial function for this.

(b) In a complete C or C++ program write the values of x in the first column and the values of the function $L_n(x)$ in consecutive columns for $n = 0, 1, 2, 3, 4$. Plot, using gnuplot the functions $L_n(x)$ in the range $x \in [0, 5]$ using the above C/C++ function for different $n = 0, 1, 2, 3, 4$ in the same plot. Save the plot as a postscript file.

3. The **Chebyshev polynomial of type I of order n** is defined by the series expansion

$$T_n(x) = \frac{n}{2} \sum_{m=0}^{[n/2]} (-1)^m \frac{(n-m-1)!}{m!(n-2m)!} (2x)^{n-2m} \tag{1.31}$$

where $[n/2]$ denotes the greatest positive integer less than (when n is odd) or equal to (when n is even) $n/2$. (For example, $[n/2] = 2$, when $n = 4$ and $[n/2] = 2$, when $n = 5$.) (a) Write a C or C++ function **double mycheb(double x, int n)** that evaluates the Chebyshev polynomial of type I, $T_n(x)$ using the above definition. (Hint: You may want to define your own function **double fact(int n)** to evaluate the factorials in the series or do otherwise.)

(b) Call *mycheb(n,x)* in a complete C or C++ program to write the values of $T_n(x)$ for $n = 1, 2, 3, 4$ in different columns of a file for $x \in [-1, 1]$. Plot the functions $T_n(x)$ for $n = 1, 2, 3, 4$ with respect to x using gnuplot and save the file as a postscript file.

4. e: 3 hours. Total Marks: 25 1. The $Ber_n(x)$ and $Bei_n(x)$ (or **Kelvin functions**) functions are the real and the imaginary parts of the **spherical Bessel's function** $J_n(xexp(3\pi\imath/4))$ and are defined by the infinite series representations (for $n = 0$) as:

$$\mathrm{Ber}(x) = 1 + \sum_{n=1}^{\infty} \frac{(-1)^n (x/2)^{4n}}{[(2n)!]^2} \tag{1.32}$$

$$\mathrm{Bei}(x) = \sum_{n=0}^{\infty} \frac{(-1)^n (x/2)^{4n+2}}{[(2n+1)!]^2} \tag{1.33}$$

$$\tag{1.34}$$

Here $(2n)!$ represents the factorial of $2n$, etc. (a) Write two C or C++ functions **double ber(double x, int n)** and **double ber(double x, int n)** that evaluate the Ber and Bei functions using the above definitions and using the first n-th term in the infinite series. You may want to define your own factorial function in evaluating the series. Start with $i_0 = 0$in the iterations.

(b) Plot, using gnuplot the functions *Ber(x)* and *Bei(x)* in the range $x \in [-10, 10]$using the above C/C++ functions for $n = 2, 4, 8, 10$ in the same plot for each function. Save the two plots as postscript files.

5. The **Bernoulli Polynomials** have fourier series representations

$$B_{2n}(x) = (-1)^{n+1}\frac{2(2n)!}{(2\pi)^{2n}}\sum_{k=1}^{\infty}\frac{\cos(2\pi kx)}{k^{2n}}, \text{ even order} \tag{1.35}$$

$$B_{2n+1}(x) = (-1)^{n+1}\frac{2(2n)!}{(2\pi)^{2n+1}}\sum_{k=1}^{\infty}\frac{\sin(2\pi kx)}{k^{2n+1}}, \text{ odd order} \tag{1.36}$$

(a) Write a C or C++ function **double bernpoly(double x, int s, int M)** that evaluates the Bernoulli polynomials of order s using the above Fourier series representations using the first M terms in the infinite series. Take suitable value of M for evaluating the function $B_s(x)$. [Hint: For $s = 2n$, use $n = s/2$, and for $s = 2n+1$, use $n = (s-1)/2$.]

(b) In a complete C or C++ program write the values of x in the first column and the values of the functions $B_s(x)$ for different values of $s = 2, 3, 4, 5$ in consecutive columns. Plot, using gnuplot the functions $B_s(x)$ in the range $x \in [0, 1]$ for $s = 2, 3, 4, 5$ using the above C/C++ function. Save the plot(s) as postscript files.

6. A continued fraction representation for the **upper incomplete gamma function**
$\Gamma(n, x) = \int_x^{\infty} t^{n-1}\exp(-t)dt$
(where the upper limit is fixed) is given by (as a sequence of fractions):

$$\begin{aligned} c_0 &= x^n e^{-x} \\ c_1 &= \frac{x^n e^{-x}}{x+1-n+1(n-1)} \\ c_2 &= \frac{x^n e^{-x}}{x+1-n+\frac{1(n-1)}{x+3-n+2(n-2)}} \\ &\ldots \\ \Gamma(n,x) &= \frac{x^n e^{-x}}{x+1-n+\frac{1(n-1)}{x+3-n+\frac{2(n-2)}{x+5-n+\frac{3(n-3)}{x+7-n+\ldots}}}} \end{aligned} \tag{1.37}$$

(a) Write a C or C++ function **double ui_gamma(int n, double x, int s)**, that evaluates the s-th order iterated value of $\Gamma(n, x)$ using the above algorithm.

(b) Call the function *double ui_gamma(int n, double x, int s)* in a complete C or C++ program and evaluate $\Gamma(n, x)$ with $s \in [0, 20]$ at $n = 2$ and $x = 1$. Save the plot in a postscript file. [Hint: $\Gamma(2, 1) = 0.7357588824$]

(c) Using **gnuplot**, plot the values of *ui_gamma(n, x, s)* versus x in the range $[0, 5]$ with s=10 and for $n = 0, 1, 2, 3$ in the same plot. Save your plot as a postscript file.

7. The **incomplete beta function** $B_x(p, q)$ is defined by the series expansion

$$B_x(p,q) = x^p\left[\frac{1}{p} + \frac{1-q}{p+1}x + \frac{(1-q)(2-q)}{2!(p+2)}x^2 + \ldots + \frac{(1-q)(2-q)\ldots(n-q)}{n!(p+n)}x^n + \ldots\right] \tag{1.38}$$

where $0 \le x \le 1$, $p > 0$, $q > 0$.

(a) Write a C++ function double **prod(double q, int n)** that evaluates $(1-q)(2-q)\ldots(n-q)$ given the values of q and n as arguments of the function (not from the standard input *cin*).

shahnoor3pl@gmail.com

(b) Define a function **double myfact(int n)** that recursively evaluates the factorial of an integer n using the relation $n! = n(n-1)!$, with $0! = 1$.

(c) Write a C++ program that approximately evaluates the incomplete beta function from the above series expansion using the first 21 terms [i.e. $B_x(p,q) = \sum_{n=0}^{20} \frac{(1-q)(2-q)...(n-q)}{n!(p+n)} x^{n+p}$].

8. The **Harmonic number** H_x for $x > 0$, integer or not, is given by

$$H_x = x \sum_{k=1}^{\infty} \frac{1}{k(x+k)} \tag{1.39}$$

which may be used to calculate the **Euler Mascheroni** constant $\gamma = 0.57721566490153286060\ldots$ and the digamma function $\psi(x)$ as

$$\gamma = \int_0^1 H_x dx \tag{1.40}$$

$$\psi(x) = H_x - \gamma - \frac{1}{x} \tag{1.41}$$

(a) Write a C++ function **double harmonic(double x, int m)** that calculates the Harmonic number H_x using the first m terms of the series above (i.e. the m-th partial sum) for any double type of argument x.

(b) Using the above function *harmonic(x, m)*, calculate the **Euler Mascheroni** constant γ by writing a C++ function **double euler(int m)** as a function of m using the above integral representation. Take number of steps equal to 1000.

(c) Plot the absolute error $|\text{euler(m)} - \gamma|$ with respect to m for $m \in [10, 10^3]$ taking appropriate steps in m. Save the plot as a postscript file.

(d) Write another C++ function **double digamma(double x, int m)** that uses *harmonic(x, m)* and the value of γ given above and $m = 1000$ to calculate $\psi(x)$ and plot $\psi(x)$ as a function of x for $x \in [0.001, 5]$. Save the plot as a postscript file.

9. The **error function** $erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x exp(-t^2)dt$ has the following infinite sum representation containing finite product as :

$$erf(x) = \frac{2}{\sqrt{\pi}} \left(x + \sum_{i=1}^{\infty} \frac{x}{2i+1} \prod_{k=1}^{i} \frac{(-x^2)}{k} \right) \tag{1.42}$$

(a) Write a C or C++ function **double error(double x, int n)** that evaluates the series expansion of the error function from the above expansion using up to the n-th term in the series for a given value of the arguments x and n. Call the function in a complete C or C++ code.

(b) Write the values of your function *error(x,n)* with $n = 5, 10, 20$ for $x \in [0.0, 2.0]$in three columns of a file and plot the data for the three cases in a single plot using gnuplot and save the plot as a postscript file.

(c) Can you guess the value of $\lim_{x\to\infty} error(x)$. Using your program above evaluate $\lim_{x\to R} error(x)$ by taking R large and find the asymptotic value.

1.5 others

C.N.1.5.1 A series expansion for the **Fresnel integral** $C(x) = \int_0^x \cos(\pi t^2/2)dt$ is given by :

$$C(x) = \sum_{k=0}^{\infty} \frac{(-1)^k (\pi/2)^{2k} x^{4k+1}}{(2k)!(4k+1)} \tag{1.43}$$

where $(2k)! = 1.2.3.4.5...(2k)$ is the factorial of $(2k)$.

(a) Define a C or C++ function **double fresnel1(double x, int n)** that evaluates the series expansion for the Fresnel integral using the expansion up to the n-th term in the series for a given value of the arguments x and n. You may want to define your own function **factorial(n)** to calculate n! and call it appropriately.

(b) Plot your function *fresnel(x,n)* with n = 20, for $x \in [0.0, 2.0]$ using gnuplot and save the plot as a postscript file.

(c) Can you guess the value of $\lim_{x\to\infty} C(x)$. Using your program above evaluate $\lim_{x\to R} C(x)$ by taking R large and find the asymptotic value.

C.N.1.5.2 The **zeta function** $\zeta(s)$ has an infinite series representation as:

$$\zeta(s) = \frac{1}{1-2^{1-s}} \sum_{k=1}^{\infty} \frac{(-1)^{k-1}}{k^s} \tag{1.44}$$

(a) Write a C or C++ function **double zeta(int s,int M)** that evaluates the zeta function from the above definition and using the first M terms in the infinite series. Take suitable value of M for evaluating the function $\zeta(s)$.

(b) In a complete C or C++ program write the values of s in the first column and the values of the function $\zeta(s)$ in second column. Plot, using gnuplot the functions $\zeta(s)$ in the range $s \in [2, 5]$and in the range $s \in [0.1, 0.8]$using the above C/C++ function. Save the two plots as postscript files.

C.N.1.5.3 The Fourier series representation of the sawtooth wave $f(x) = x$ for $x \in [-\pi, \pi]$ with $f(x \pm 2\pi) = f(x)$, is given by

$$f(x) = 2\left[\sin(x) - \frac{\sin(2x)}{2} + \frac{\sin(3x)}{3} - \ldots + (-1)^{n+1}\frac{\sin(nx)}{n} + \ldots\right] \tag{1.45}$$

where $x \in [-\pi, \pi]$

(a) Write a C++ program that evaluates the n-th partial sum (i.e. sum of the first n terms), where $n = 2, 3, 4, ..., 8$, of the series, for $x \in [-\pi, \pi]$ and writes the values in a file. Take the increment of x to be $\pi/20$. Your program should write the values of x in the first column, the partial sum for $n = 2$ in the second column of the file, that for $n = 3$ in the third column, etc.

(b) Plot the partial sums (for $n = 2, 3, 4, ..., 8$) in a single plot, using gnuplot, for $x \in [-\pi, \pi]$. Save your plot as a postscript file with your roll number included in the name of the file.

C.N.1.5.4 Consider the recursive sequence defined by $x_0 = p$ and $x_{n+1} = p + p/x_n$. Its asymptotic value $X = \lim_{n\to\infty} x_n$ is equal to the **infinite continued fraction** (where p is an integer):

$$X = p + \cfrac{p}{p + \cfrac{p}{p + \cfrac{p}{p+\ldots}}} \tag{1.46}$$

(a) Write a C++ function **double contfrac(double p, int n)**, that evaluates the n-th order iterated value of the continued fraction. You may use recursive calling of function to evaluate the *confrac(p, n)*.

(b) Plot the values of x_n for $p = 2, 3$ in a single plot, for different values of n using gnuplot or any other available plotting program. Save your plot as a postscript file. (For comparison $p = 2$, $X \approx 2.73205\ldots$ and for $p = 3, X \approx 3.79129\ldots$).

C.N.1.5.5 The **Bernoulli numbers** for even integers $m = 2n$ are defined by the following series expansion:

$$B_m = B_{2n} = \frac{(-1)^{n-1}2(2n)!}{(2\pi)^{2n}} \sum_{p=1}^{\infty} p^{-2n} \tag{1.47}$$

where $m = 2, 4, 6..., m = 2n$

(a) Define a function funcpdouble myfact(int m) that recursively evaluates the factorial of an integer m using the relation $m! = m(m-1)!$, with $0! = 1$. Use this to calculate $(2n)!$.

(b) Now write a function **double bernoulli(int n)** that calculates the Bernoulli numbers for $m = 2n$ using up to the first 20 terms in the series. Test your code by outputting the values of B_{2n}, where $n = 1, 2, 3, 4, \ldots, 20$. For your reference, a few Bernoulli numbers are given: $B_2 = 1/6, B_4 = -1/30, B_6 = 1/42, B_8 = -1/30, B_{10} = 5/66, B_{12} = -691/2730$, etc.

(c) Define a function **double testfunc(double x, int N)** that evaluates the series

$$f(x) = 1 - \frac{x}{2} + \sum_{n=1}^{N} B_{2n} \frac{x^{2n}}{(2n)!} \tag{1.48}$$

and print the values of $f(x)$ from the above definition, the exact result $x/(e^x - 1)$ using the $< cmath >$ function $exp()$ for $x \in [0.5, 2.0]$, $N = 30$ and the positive-valued difference between the two results. Take the increment of x to be 0.05.

C.N.1.5.6 The **Stirling's approximation** to the factorial of a large number n is given by

$$n! \approx \sqrt{2\pi n} n^n e^{-n} \left(1 + \frac{1}{12n} + \frac{1}{288n^2} - \frac{139}{51840n^3} + \ldots\right) \tag{1.49}$$

(a) Write a C++ function **double myfactorial(int n)** that recursively evaluates the factorial of an integer n from the definition $n! = n(n-1)!$, with $0! = 1$.

(b) Write a C++ program that evaluates $n!$ using the above formula.

(c) Print the positive-valued difference of the two values of $n!$ calculated as above, divided by the exact value found from the function *myfactorial()*, for $n = 2, 4, 6, \ldots 12$.

C.N.1.5.7 The complete **elliptical integral** of the second kind $E(m)$ has the series expansion

$$E(m) = \frac{\pi}{2}\left[1 - \sum_{n=1}^{\infty}\left[\frac{(2n-1)!!}{(2n)!!}\right]^2 \frac{m^n}{2n-1}\right], \; 0 \le m < 1 \tag{1.50}$$

The double factorials are defined as $n!! = n(n-2)!!$ with $0!! = (-1)!! = 1$.

(a) Write a C++ function **double dfact(int n)** that evaluates the double factorial of n using the definition given above.

(b) Write a C++ function **double ellipticE(double m)**, that approximately evaluates $E(m)$ from the above series expansion using the first 20 terms [i.e. $K(m) \approx (\pi/2)(1 - \sum_{n=1}^{20}[(2n-1)!!/(2n)!!]2mn/(2n-1))$].

(c) Print the values of $E(m)$ for $m \in [0, 1.0]$. Take the increment of m to be 0.05.

C.N.1.5.8 The **Riemann zeta function** is defined as

$$\zeta(s) = \sum_{n=1}^{\infty} n^{-s} \tag{1.51}$$

where $s > 1$

(a) Write a program that approximately calculates the Zeta function $\zeta(s)$ for $s = 2, 4, 6, 10, \ldots 20$. Use the first 30 terms i.e. $\sum_{n=1}^{30} n^{-s}$, to evaluate $\zeta(s)$.

(b) The Euler-Mascheroni constant is defined in terms of the ζ-function by

$$\gamma = \sum_{s=2}^{\infty} (-1)^s \frac{\zeta(s)}{s} \tag{1.52}$$

Evaluate approximately γ by including the first 19 terms i.e. using $\sum_{s=2}^{20} (-1)^s \frac{\zeta(s)}{s}$.

C.N.1.5.9 The n-th **Fermat number** is defined as

$$F_n = 2^{2^n} + 1 \tag{1.53}$$

(a) Define a function **double fermatn(int n)** that calculates the n-th Fermat number using the above definition. In doing so, use either the built-in function *pow()* or you may want to define your own function that returns a floating point number a raised to some integer power m i.e. a^m.

(b) Write another function **double myfunc(int n)** that returns the value of the following function $G(n)$ with the product evaluated up to the first n-terms:

$$G(n) = \prod_{i=0}^{n} \left(1 - \frac{1}{F_n}\right) \tag{1.54}$$

(c) Write the value of $G(n)$ as a function of n for $n \in [0, 5]$ in a file and use this data to plot $G(n)$ with respect to n using gnuplot. Save your plot as a postscript file.
What is the asymptotic value of the product?

C.N.1.5.10 The following infinite series is a representation of $\ln 2$:

$$\ln 2 = \frac{1}{2} + \frac{1}{2.2^2} + \frac{1}{3.2^3} + \frac{1}{4.2^4} + \ldots = \sum_{i=1}^{\infty} \frac{1}{i.2^i} \tag{1.55}$$

(a) Write a C or C++ function **double myln2(int n)**, that evaluates the value of $\ln 2$ with the first n-terms in the sum using the above formula.

(b) Call the function *myln2(n)* in a complete C or C++ program and evaluate $x_n = myln2(n)$ with $n = 1, 2, \ldots, 30$ and write the values in a data file.

(c) Plot the values of $|x_n - \ln 2|$ for different values of n using gnuplot. Save your plot as a postscript file.

C.N.1.5.11 The **hypergeometric function** ${}_2F_1(a, b, c; x)$ is defined in terms of the series

$${}_2F_1(a, b, c; x) = \sum_{n=0}^{\infty} \frac{(a)_n (b)_n}{(c)_n} \frac{x^n}{n!} = 1 + \frac{ab}{c}\frac{x}{1!} + \frac{a(a+1)b(b+1)}{c(c+1)}\frac{x^2}{2!} + \ldots \tag{1.56}$$

where $c \neq 0, -1, -2, \ldots$ and $(a)_n$ is the **Pochhammer symbol** defined by $(a)_n = a(a+1)(a+2)\ldots(a+n-1)$, with $(a)_0 = 1$.

shahnoor3pl@gmail.com

(a) Write a C or C++ function **double poch(double a, int n)** that evaluates $(a)_n$ given the values of a and n as arguments of the function (not from the standard input using cin or scanf).

(b) Write a C or C++ program that approximately evaluates the hypergeometric function using the above series expansion with the first 21 terms (i.e. ${}_2F_1(a,b,c;x) = \sum_{n=0}^{20} \frac{(a)_n (b)_n}{(c)_n} \frac{x^n}{n!}$.

(c) Plot ${}_2F_1(a,b,c;-x)$ for $x \in [0,1.0]$ using the above series. Compare your plot with the exact result ${}_2F_1(a,b,c;-x) = \ln(1+x)$. Save the plots of your function and that of ln(1 + x) in a single postscript file.

shahnoor3pl@gmail.com

Chapter 2

Function

C.N.2.0.1 In theory of probability, the **gamma distribution** function is given by

$$f(x;\alpha,\beta) = \frac{1}{\beta^{\alpha}\Gamma(\alpha)} x^{\alpha-1}\exp(-x/\beta) \tag{2.1}$$

where $x > 0$; $\alpha, \beta > 0$
and $\Gamma(\alpha) = (\alpha-1)!$, for integer α, is the gamma function. (a) Write a C or C++ function **double gammadist(double x, int alpha, double beta)** that evaluates the beta distribution function using the above definition and the factorial function.

(b) Write a C or C++ program that evaluates the gamma distribution function for different values of α and $\beta = 2$. Using the function **double gammadist(x, alpha, beta)**, evaluate (i) $\langle x\rangle$ i.e. the mean value of x and (ii) $\sigma^2 = \langle x^2\rangle - \langle x\rangle^2$, i.e. the variance of x, using different $\alpha = 1, 2, 3, 4, ...$ and $\beta = 2$. Take the range of x as $[0, X]$, where X is a sufficiently large number. You may use any numerical integration method of your choice. (c) Plot (i) $\langle x\rangle$ and (ii) σ^2 with respect to α using gnuplot and save the plots in a postscript file.

C.N.2.0.2 In theory of probability, the **beta distribution** function is given by

$$f(x;p,q) = \frac{1}{B(p,q)} x^{p-1}\,(x-1)^{q-1} \tag{2.2}$$

where $0 \le x \le 1$; $\quad p, q > 0$
and $B(p,q) = \int_0^1 t^{p-1}\,(t-1)^{q-1}\,dt$ is the beta function.

(a) Write a C or C++ function **double betadist(double x, double p, double q)** that evaluates the beta distribution function using the above two definitions. You may use any numerical integration method of your choice.

(b) Write a C or C++ program that evaluates the beta distribution function for different values of p, q and $x \in [0, 1]$ Take (p, q) equal to $(2, 2)$ and $(3, 3)$. Plot $f(x; p, q)$ with respect to x for $x \in [0, 1]$ using gnuplot and save the plots in a postscript file.

C.N.2.0.3 The period of a **simple pendulum** for large angle amplitude (θ_M) may be written as

$$T = 2\pi\sqrt{\frac{L}{g}}\frac{1}{M\left(\sqrt{(1-sin^2(\theta_M/2))}\right)} \tag{2.3}$$

where $0 \le \theta_M < \pi$ and $M(x)$ is the **arithmetic-multiplicative mean**, defined as
$M(x) = \lim_{n\to\infty} a_n$, where $a_1 = x$, $b_1 = 1$, $a_{n+1} = \frac{a_n+b_n}{2}$, $b_{n+1} = \sqrt{(a_n b_n)}$

(a) Write a C or C++ function **double artmul(double x, int n)**, that evaluates the n-th order iterated value of the above sequence that approximates the arithmetic-multiplicative mean $M(x)$ at x.

(b) Write a C or C++ function **double pendulumT(double thetaM)**, that evaluates the period T for a given θ_M as the argument using the above definition and the function **double artmul(double x, int n)**. Choose the value of L/g such that, as $\theta_M \to 0$, $T = 1s$ and take a large value of n. Call the function *pendulumT()* from the *main()* function with different values of θ_M as input.

(c) Using the above function, plot T vs. θ_M for $\theta_M \in [0, \pi/2]$. Hint: Check values: $\theta_M = [10\,\text{deg}, 50\,\text{deg}, 90\,\text{deg}] \Longrightarrow T = [1.00193, 1.05033, 1.18258]$.

C.N.2.0.4 In celestial mechanics, **Kepler's Equation** relates the mean anomaly M to the eccentric anomaly E of an elliptical orbit of eccentricity e as:

$$M = E - e\sin(E) \tag{2.4}$$

The orbital eccentricities (e) of the first six planets in the solar system are 0.2056, 0.0068, 0.0167, 0.0934, 0.0483, 0.0560; the mean eccentricity of the Moon's orbit is 0.0549.

(a) Write a C or C++ function **double eccanom(double M, double e)** that will calculate the eccentric anomalies E for a planet or the Moon for given values of M and e using any suitable root finding method.

(b) Call *eccanom(M,e)* in a complete C or C++ program and write the values of E in different columns of a file for each planet and the Moon when the mean anomaly $M \in [\pi/4, \pi/2]$ radians and varying M in steps of $\pi/80$. Plot the values in a single plot using gnuplot and save the plot as a postscript file.

(c) Are there values for the eccentric anomaly for which $E = M$? If so, what are they?

C.N.2.0.5 The **modified Struve's function** $M_\nu(x)$ has the integral representation given by

$$M_\nu(x) = -\frac{2(x/2)^\nu}{\sqrt{\pi}(\nu - 1/2)!}\int_0^1 \exp(-xt)(1-t^2)^{\nu-1/2}dt \tag{2.5}$$

where $\nu > -1/2$. For this problem, take $\nu = n = 1, 2, 3, 4, \ldots$etc.

(a) Using any suitable numerical integration scheme, write a C or C++ function **double modstruv(double x, int n)** that evaluates the modified Struve's function $M_n(x)$, where n is a positive integer. Use suitable number of grid points for $t \in [0, 1]$. You need to define your own function for calculating the factorial.

(b) Call the above function *modstruv(x,n)* in a complete C or C++ program and write the values of x in the first column and the function values $M_n(x)$ in consecutive columns for $n = 1, 2, 3, \ldots$ for $x \in [-2, 2]$ in a file. Plot $M_n(x)$ versus x for $n = 1, 2, 3$ using gnuplot in a single plot and save the plot as a postscript file.

C.N.2.0.6 The **modified Besssel function of the first kind** $I_\nu(x)$ has the integral representation given by

$$I_\nu(x) = \frac{1}{\pi^1/2(\nu - 1/2)!}\left(\frac{x}{2}\right)^\nu \int_{-1}^1 \exp(xp)(1-p^2)^{\nu-1/2}dp \tag{2.6}$$

where $\nu > -1/2$. For this problem, take $\nu = n = 1, 2, 3, 4, \ldots$ etc.

(a) Using any suitable numerical integration scheme, write a C or C++ function **double BesselI(double x, int n)** that evaluates the modified Besssel function of the first kind $I_n(x)$, where n is a positive integer. Use suitable number of grid points beetwen the interval $[-1, 1]$. You need to define your own function for calculating the factorial.

(b) Call the above function *BesselI(x,n)* in a complete C or C++ program and write the values of x in the first column and the function values $I_n(x)$ in consecutive columns for $n = 1, 2, 3, \ldots$ for $x \in [0, 3]$in a file. Plot $I_n(x)$ versus x for $n = 1, 2, 3$ using gnuplot in a single plot and save the plot as a postscript file.

Chapter 3

Recursive relations

C.N.3.0.1 The **Laguerre polynomial** may be defined recursively

$$\begin{aligned} L_0(x) &= 1 \\ L_1(x) &= -x + 1 \\ L_{n+1}(x) &= 2L_n(x) - L_{n-1}(x) - \frac{(1+x)L_n(x) - L_{n-1}(x)}{n+1} \end{aligned} \tag{3.1}$$

where $n = 2, 3, 4 \ldots$

(a) Write a C or C++ function **double laguerre(int n, double x)** to calculate $L_n(x)$ using the above definition.

(b) Use the above function *laguerre(n, x)* to plot $L_n(x)$ for $x = -5$ to 10 and $n = 1, 2, 3, 4$. [In the plot x-range should be [-5:10] and y-range should be [-10:20]].
The sample plot is given below

(c) Write a function that will solve this problem using loop.

C.N.3.0.2 The Recursive relation of **Legendre Polynomial** is defined as

$$(n+1)P_{n+1}(x) = (2n+1)xP_n(x) - nP_{n-1}(x) \tag{3.2}$$

with $P_0(x) = 1$ and $P_1(x) = x$
(a) Write a C or C++ function **double legendre(int n, double x)** that will evaluate the n-th value of Legendre Polynomial at x.

(b) plot graph of Legendre Polynomial for $n = 0, 1, 2, 3, 4, 5$ using $x \in [-1, 1]$. The sample graph is given below

(c) Write a function that will solve this problem using loop.

C.N.3.0.3 The **Hermite Polynomial** may be defined recursively by

$$H_n(x) = 2xH_{n-1}(x) - 2(n-1)H_{n-2}(x) \quad \text{,where } n = 2, 3, 4, \ldots \tag{3.3}$$

with $H_0(x) = 1$ and $H_1(x) = 2x$

(a)Write a function **double hermite(double x, int n)** that recursively calculates the Hermite polynomial of order n at some point x.

(b) plot graph of Hermite Polynomial for $n = 0, 1, 2, 3, 4, 5$ using $x \in [-3, 3]$.[In the plot x-range should be [-3:3] and y-range should be [-50:50] and [-100:100] and [-200:200] for three separate graph.
Sample figure is given below.

(c) Write a function that will solve this problem using loop.

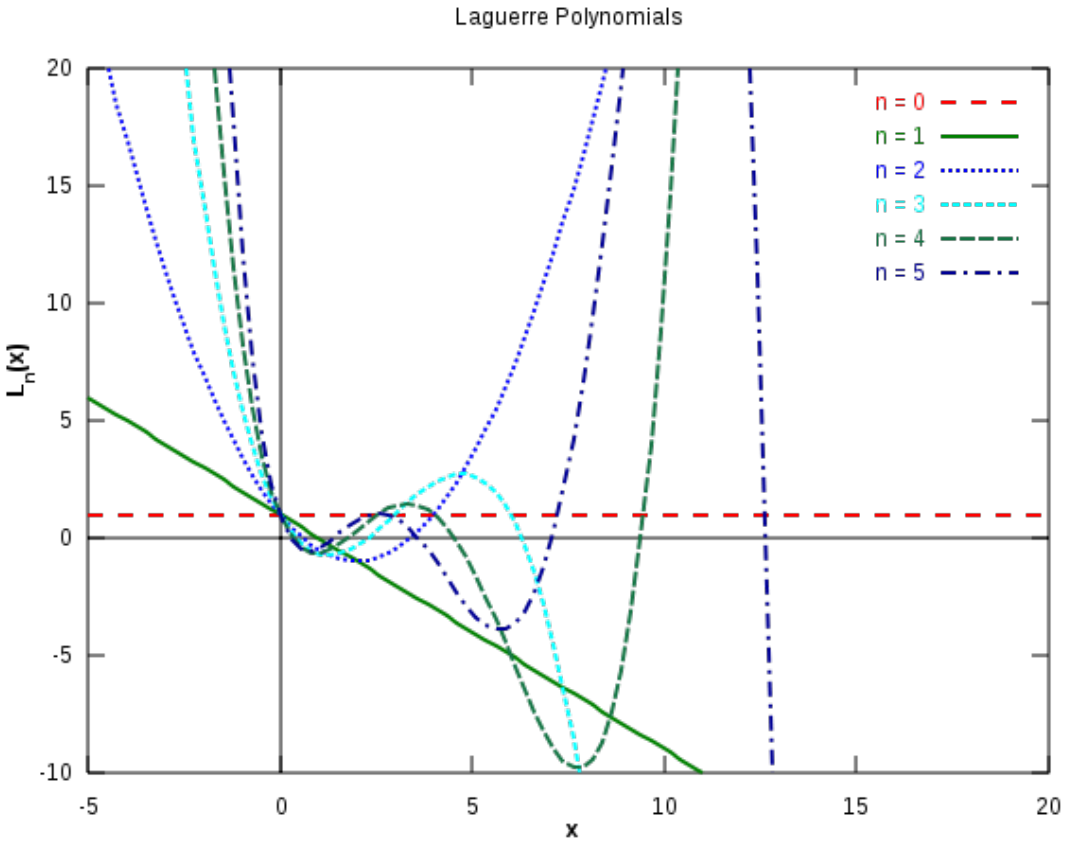


Figure 3.1: The Laguerre polynomial

C.N.3.0.4 Write a C or C++ program that can compute the **Fibonacci series** using recursive function.

C.N.3.0.5 The **Chebyshev polynomials of the first kind** is defined as

$$T_0(x) = 1 \tag{3.4}$$
$$T_1(x) = x \tag{3.5}$$
$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x) \tag{3.6}$$

(a) Write a C or C++ function that will evaluate Chebyshev polynomial recursively.

(b) Write a C or C++ function that will evaluate Chebyshev polynomial using loop.

(c) Plot Chebyshev polynomial for $n = 0, 1, 2, 3, 4, 5$ using gnuplot. [Hint: set y-range from -2 to 2 and set x-range from -2 to 2 for better viewing].
Sample figure is given below.

C.N.3.0.6 The **Chebyshev polynomials of the second kind** is defined as

$$U_0(x) = 1 \tag{3.7}$$
$$U_1(x) = 2x \tag{3.8}$$
$$U_{n+1} = 2xU_n - U_{n-1} \tag{3.9}$$

(a) Write a C or C++ function that will evaluate Chebyshev polynomial recursively.

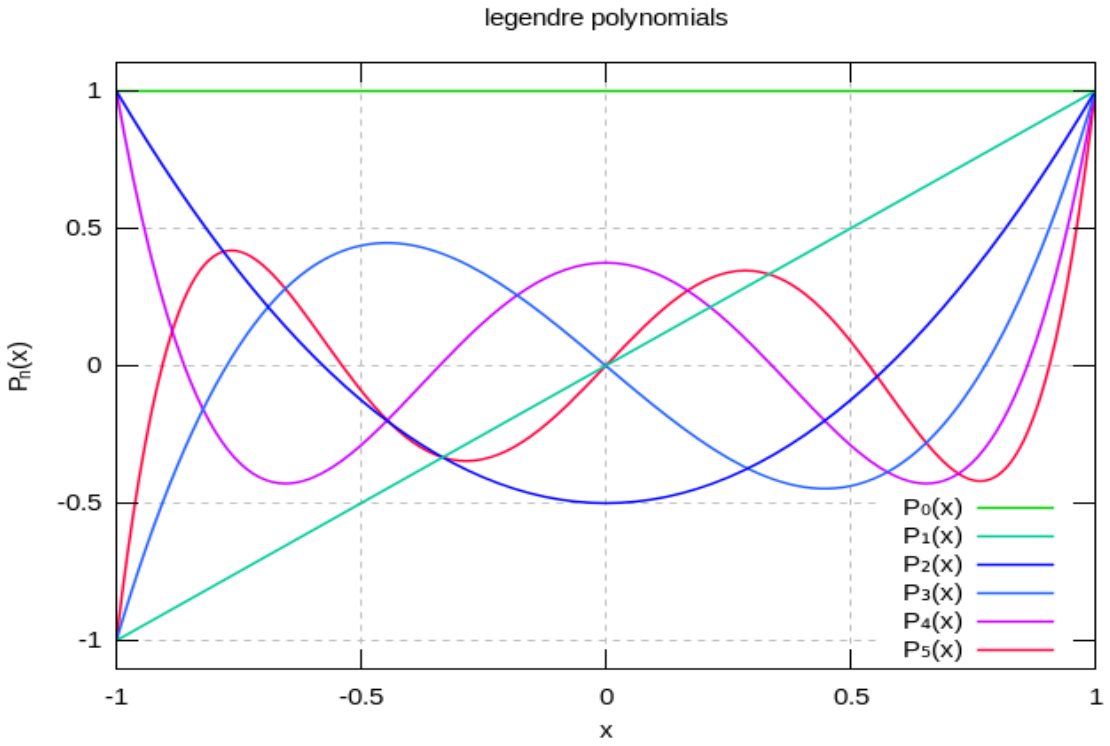


Figure 3.2: The Legendre polynomial

(b) Write a C or C++ function that will evaluate Chebyshev polynomial using loop.

(c) Plot Chebyshev polynomial for $n = 0, 1, 2, 3, 4, 5$ using gnuplot. [Hint: set y-range from -4 to 4 and set x-range from -1 to 1 for better viewing].
Sample figure is given below.

C.N.3.0.7 A general second order recurrence relation for a function of n may be written as

$$\begin{aligned} T(0) &= \alpha \\ T(1) &= \beta \\ T(n) + aT(n-1) + bT(n-2) = 0 \text{ , where n=2,3,4,...} \end{aligned} \tag{3.10}$$

(a) Write a function **double recur(double a, double b, double alpha, double beta, int n)**, that recursively calculates the function $T(n)$ given an integer n and the parameters a, b, α, β.

(b) Use your function to write the value of $T(n)$ as a function of n for $(\alpha, \beta) = (0, 1), (0, 2), (1, 4), (1, -1), (1, 2)$ with $(a, b) = (4, -3), (10, -25)$ in a file with respect to change of n. Plot $T(n)$ as a function of n for these set of values of the parameters. Save your plot as a postscript file.

(c) Now write the same function which will use loop to evaluate the above recursive relation. Also plot the graph. [Hint : use array and loop]

C.N.3.0.8 The **shifted Chebyshev polynomials** are defined recursively as

$$\begin{aligned} T_0^*(x) &= 1 \\ T_1^*(x) &= 2x - 1 \\ T_n^*(x) &= (4x - 2)T_{n-1}^*(x) - T_{n-2}^*(x) \end{aligned} \tag{3.11}$$

where $n = 2, 3, 4, \ldots$

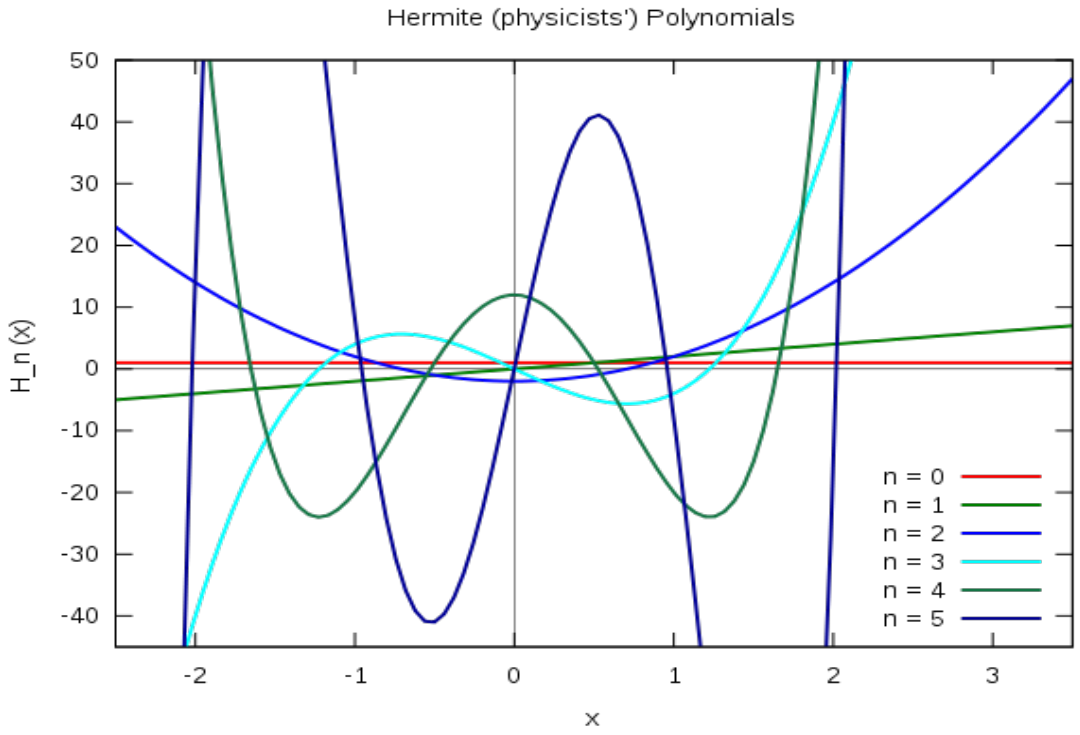


Figure 3.3: Hermite polynomial

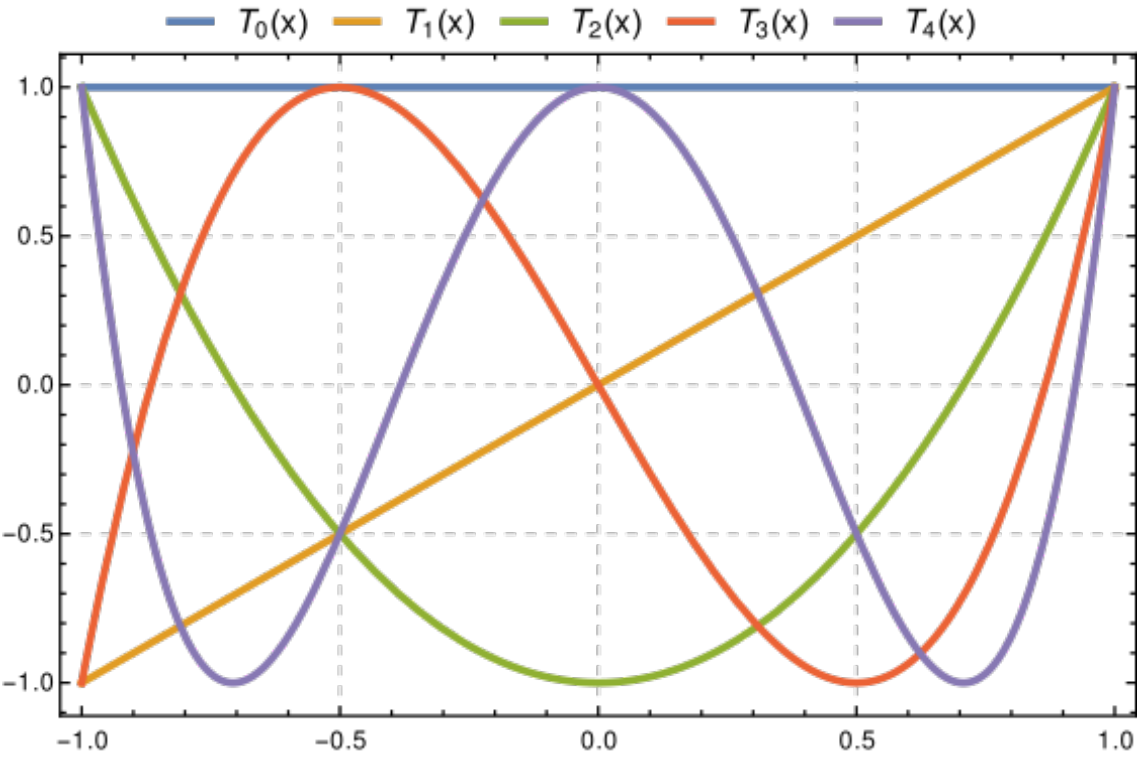


Figure 3.4: Chebyshev Polynomials of the First Kind

shahnoor3pl@gmail.com

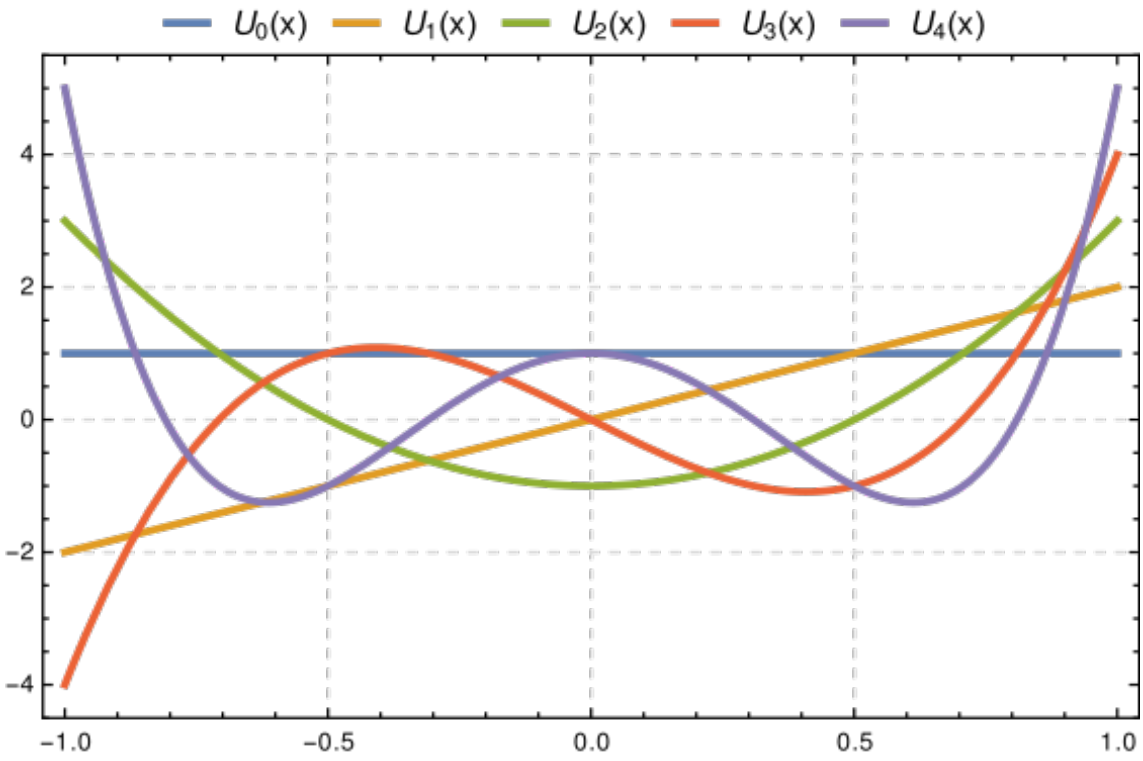


Figure 3.5: Chebyshev Polynomials of the Second Kind

(a) Write a function **double shifcheb(double x, int n)**, that recursively calculates the shifted Chebyshev Polynomial of order n, at some point x, given an integer n from the standard input.

(b) Use your function to write the value of x in the first column and the value of $T_1^*(x)$ in the second column, $T_2^*(x)$ in the third column, $T_3^*(x)$ in the fourth column and $T_4^*(x)$ in the fifth column of a file for $x \in [0, 1]$taking an increment of 0.01. Plot the function $T_n^*(x)$ for $n = 1, 2, 3, 4$ using gnuplot in a single plot. Save that plot as a postscript file.

(c) Plot $(3T_0^*(x) + 4T_1^*(x) + T_2^*(x))/8$ and $(10T_0^*(x) + 15T_1^*(x) + 6T_2^*(x) + T_3^*(x))/32$ as functions of x for $x \in [0, 1]$ taking an increment of 0.01 and save the plots as postscript files. Predict the behaviour of the plots as power functions of x i.e. x^p and identify the exponent p in each case.

(c) Write a function that uses loop to evaluate the above recursion relation and repeat step (b) to plot and compare the results.

C.N.3.0.9 A numerical method for finding the square root of a positive real number b is given by the recursive algorithm:

$$\xi_{n+1} = \frac{1}{2}\left[x_n + \frac{b}{x_n}\right] \tag{3.12}$$

with $x_0 = 1$ and $n = 0, 1, 2, ...$

(a) Write a function **double mysqroot(double b, int n)** that finds the square root of the positive real number b for a given number of iterations n and using the above algorithm. Use $n = 10$ for evaluation purpose.

(b) Plot the difference between the square root of $b = 100.0$ found from your function and from the built-in function sqrt() (defined in the header file ¡cmath¿ in C++ or ¡math.h¿ in C), for $n = 1, 2, 3, \ldots 20$.

shahnoor3pl@gmail.com

(c) Write the absolute value of the error i.e. difference between the square root calculated from your function and that from the built-in function *sqrt()* as a function of the number of iterations n, in a file. Plot the absolute value of error with respect to n using gnuplot and save the plot as a postscript file.

(d) Use your function **double mysqroot(double b, int n)** to output (e.g., the first 100 terms of) the sequence

$$d_i = \left(\sqrt{i} - \sqrt{i-1}\right) - \frac{1}{\sqrt{i} + \sqrt{i-1}} \tag{3.13}$$

for$i = 1, 2, 3 \ldots 100$ Use $n = 10$ in *mysqroot(double b, int n)*.

C.N.3.0.10 Consider the simple **Logistic Map** defined by the recursive relation

$$x_{n+1} = a - x_n^2 \tag{3.14}$$

where a is a parameter.

(a) Write a C or C++ function **double logistic(double a, double x0, int n)** that calculates the n-th order iterated value of x starting from a given x_0 and using a given a.

(b) In the *main()* function, call the function *logistic(a,x0,n)* with $a = 0.5$, $a = 1.476$ and $a = 2.0$ starting from $x_0 = 0.5$ for $n \in [0, 100]$. Write the value of n in the first column of a file, and successive values x_n for different a in different columns of the file.

(c) Plot the successive values of x_n with respect to n for different a using gnuplot. Save the file(s) as postscript file.

(d) Now change the program and write the values of x_{n-1} and x_n in two different columns of another datafile. Plot x_n vs x_{n-1} for $a = 2$ using gnuplot and save the plot in a postscript file.

C.N.3.0.11 Consider the simple **Logistic Map** defined by the quadratic recursive relation

$$x_{n+1} = rx_n(1 - x_n) \tag{3.15}$$

where r is a positive constant sometimes known as the biotic potential.

(a) Write a C or C++ function **double logistic(double r, double x0, int n)** that calculates the n-th order iterated value of x starting from a given x_0 and using a given r.

(b) Now take a random number $x \in [0, 1]$ and write a C or C++ function **double fixed(double r, double x)** that uses the function *logistic(r,x,1000)* that calculates the so called fixed point of the map for a given x and r.

(c) Now take 100 values of x (i.e. run a loop 100 times with different random numbers generated) and each time calculate the fixed point for a fixed r. Take $r = 0.1, 0.5, 2.0, 3.0, 3.25, 3, 5, 3.75, 3.85$ and plot the values of the fixed points with respect to r. Save the plot as a postscript file. (Use with points option in gnuplot).

Chapter 4

Vectors and Matrix

C.N.4.0.1 Write a C or C++ program that can multiply two matrices A and B. where A is an 3×3 matrix and B is an 3×3 matrix.

Now modify your program so that it can multiply $A = m \times n$ matrix and $B = n \times l$ matrix to give a new matrix $C = m \times l$;

C.N.4.0.2 (a) The **Levi-civita** symbol is defined as

$$\epsilon_{ijk} = \frac{1}{2}(i-j)(j-k)(k-i) \tag{4.1}$$

which has the value of $+1$ if ijk is an even permutation of 123, -1 if ijk is an odd permutation of 123 and zero otherwise. Write a C or C++ function **int epsn(int i, int j, int k)** that will return the value of Levi-civita symbol.

(b) The determinant of a 3×3 matrix can be written as

$$\det(a) = \epsilon_{ijk} a_{1i} a_{2j} a_{3k} \tag{4.2}$$

write a C or C++ function **double det(double a[3][3])** that takes an argument a $2D$ array and returns its determinant using above definition.

C.N.4.0.3 Write a C or C++ function that can find the determinant of a 3×3 matrix.
Now modify your program so that it can find the determinant of a $n \times n$ matrix.

Chapter 5

Differentiation

C.N.5.0.1

Chapter 6

Integration

6.1 Monte Carlo Integration

C.N.6.1.1 The area under the curve of a given function is defined as $A = R\frac{n}{N}$, where, R is the area of the rectangle enclosing the curve, n is the number of random points that hit inside of the curve and N is the total number of random points that hit inside of the whole area. This method is known as **Monte Carlo Integration**

(a) Write a C++ function *area(f, a, b, c, d, N)* to calculate the area of a given curve. Here, f is any function representing the curve, $b-a$ and $d-c$ correspond to the sides of the rectangle and N is the total number of random points.

(b) Consider the following integral

$$I = \int_0^{1/5} \exp(-5x^2) \tag{6.1}$$

Solve the integral using the area under the curve and the rectangle with sides 0.20 and 1.0. The exact value of the integral is 0.18743000575959573673.

(c) Plot the absolute error as a function of N for $N \in [103, 107]$. Save the plot as a postscript file.

C.N.6.1.2 Random numbers may be used to calculate integrals. This method is known as **Monte Carlo Integration**. Consider the following integral

$$I = \int_{-1}^{1} \frac{2}{1+x^2} dx \tag{6.2}$$

(a) Generate a sequence of pair of random numbers (x, y) with $x \in [-1, 1]$ and $y \in [1, 2]$. Write a C or C++ function **double integral2(int N)** to calculate the value of the integral by comparing the area under the curve y = 2 1+x2 and that of the rectangular region with sides 2 and 1 using N pairs of random numbers, i.e. N points.

(b) Increase the number of points used to evaluate the area and find the errors in each case. Take $N \in [10, 107]$. The exact value of the integral is $\pi/2$.

(c) Write the number of points and the errors in two columns of a file. Plot the error vs the number of points used and the save the plot as a postscript file.

C.N.6.1.3 Random numbers may be used to calculate integrals. This method is known as **Monte Carlo Integration**. Consider the following integral

$$I = \int_0^{\pi} \frac{1}{1+\sin^2(x)} dx \tag{6.3}$$

(a) Generate a sequence of pair of random numbers (x, y) with $x \in [0, \pi]$ and $y \in [0, 1]$. Write a C or C++ function **double integral2(int N)** to calculate the value of the integral by comparing the area under the curve $y = 1/(1 + sin^2(x))$ and that of the rectangular region with sides π and 1 using N pairs of random numbers, i.e. N points.

(b) Increase the number of points used to evaluate the area and find the errors in each case. Take $N \in [10, 10^7]$. The exact value of the integral is $\pi/\sqrt{2}$.

(c) Write the number of points and the errors in two columns of a file. Plot the error vs the number of points used and the save the plot as a postscript file.

C.N.6.1.4 Random numbers may be used to calculate integrals. This method is known as **Monte Carlo Integration**. Consider the following integral

$$I = \int_{-\pi}^{\pi} \frac{\exp(-x^2)}{1 + cos^2(x)} dx \tag{6.4}$$

(a) Generate a sequence of pair of random numbers (x, y) with $x \in [-\pi, \pi]$and $y \in [0, 0.5]$. Write a C or C++ function double integral2(int N) to calculate the value of the integral by comparing the area under the curve $y = exp(-x^2)/(1 + cos^2(x))$ and that of the rectangular region with sides 2π and 0.5 using N pairs of random numbers, i.e. N points.

(b) Increase the number of points used to evaluate the area and find the errors in each case. Take $N \in [10, 10^7]$. The exact value of the integral is 11.0964423232264471296 . . ..

(c) Write the number of points and the errors in two columns of a file. Plot the error vs the number of points used and the save the plot as a postscript file.

C.N.6.1.5 Random numbers may be used to calculate integrals. This method is known as **Monte Carlo Integration**. Consider the Fresnel integrals $C(z)$ and $S(z)$:

$$C(z) = \int_0^z \cos(\frac{\pi x^2}{2}) dx \tag{6.5}$$

$$S(z) = \int_0^z \sin(\frac{\pi x^2}{2}) dx \tag{6.6}$$

$$\tag{6.7}$$

(a) Generate a sequence of pair of random numbers (x, y) with $x \in [0, z]$and $y \in [0, 1]$. Write C or C++ functions **double C(double z, int N)** and **double C(double z, int N)** to calculate the value of the Fresnel integrals by comparing the areas under the curves $y = cos(\pi x^2)/2$ and $y = sin(\pi x^2)/2$ with that of the rectangular region with sides z and 1 using N pairs of random numbers, i.e. N points.

(b) Increase the number of points used to evaluate the area and find the errors in each case. Take $N \in [10, 107]$. The exact value of the integrals for $z = 1$ are $C(1) = 0.77989340037682282947 \ldots$ and $S(1) = 0.43825914739035476608 \ldots$, respectively.

(c) Write the number of points and the errors in two consecutive columns of a file. Plot the errors vs the number of points used and the save the plot as a postscript file.

C.N.6.1.6 Random numbers may be used to calculate integrals. This method is known as **Monte Carlo Integration**. Consider the following integral

$$I(\mu, \beta) = \int_0^{5\mu} \frac{dx}{1 + \exp[(x - \mu)/\beta]} \tag{6.8}$$

(a) Generate a sequence of pair of random numbers (x, y) with $x \in [0, 5\mu]$ and $y \in [0, 1.0]$. Write a C++ function **double integral2(int N)** to calculate the value of the integral by comparing the area under the curve $y = \frac{1}{1+\exp[(x-\mu)/\beta]}$ and that of the rectangular region with sides 5μ and 1.0 using N pairs of random numbers, i.e. N points. Take $\mu = 0.1, 0.2, 0.5, 1.0$ and $\beta = 0.2$.

(b) Take $N = 10^7$ and evaluate the integral for the above set of parameters. $N \in [10, 10^7]$. The approximate values of the integral are
$0.20783406594692304377, 0.25902235192008261874, 0.45150680017215039621, 0.78994280204332406688$.

(c) Write the number of points and the errors in two columns of a file (taking $\mu = 1.0$ and $\beta = 0.2$) and increase the number of points used to evaluate the area. Find the errors in each case. Plot the error vs the number of points used and the save the plot as a postscript file.

C.N.6.1.7 Random numbers may be used to calculate integrals. This method is known as **Monte Carlo Integration**. Consider the following integral

$$I = \int_{-\pi}^{\pi} (\exp(-|x|/\pi) + \exp(|x|/\pi)) \sin(|x|) dx \tag{6.9}$$

(a) Generate a sequence of pair of random numbers (x, y) with $x \in [-\pi, \pi]$ and $y \in [0.0, 5/2]$. Write a C or C++ function double **integral2(int N)** to calculate the value of the integral by comparing the area under the curve $y = (\exp(-|x|/\pi) + \exp(|x|/\pi)) \sin(|x|)$ and that of the rectangular region with sides 2π and 2.5 using N pairs of random numbers, i.e. N points.

(b) Increase the number of points used to evaluate the area and find the errors in each case. Take $N \in [10, 10^7]$. The approximate value of the integral is 9.2364722392959185640.

(c) Write the number of points and the errors in two columns of a file. Plot the error vs the number of points used and the save the plot as a postscript file.

C.N.6.1.8 Random numbers may be used to calculate integrals. This method is known as **Monte Carlo** Integration. Consider the following integral

$$I = \int_{-\pi}^{\pi} \left(\frac{1}{1 + \sin^2(x)} + \frac{1}{1 + \cos^2(x)} \right) dx \tag{6.10}$$

(a) Generate a sequence of pair of random numbers (x, y) with $x \in [-\pi, \pi]$ and $y \in [0.0, 3/2]$. Write a C or C++ function **double integral2(int N)** to calculate the value of the integral by comparing the area under the curve $y = \frac{1}{1+\sin^2(x)} + \frac{1}{1+\cos^2(x)}$ and that of the rectangular region with sides 2π and 1.5 using N pairs of random numbers, i.e. N points.

(b) Increase the number of points used to evaluate the area and find the errors in each case. Take $N \in [10, 10^7]$. The approximate value of the integral is $2\pi\sqrt{2}$.

(c) Write the number of points and the errors in two columns of a file. Plot the error vs the number of points used and the save the plot as a postscript file.

C.N.6.1.9 Random numbers may be used to calculate integrals. This method is known as **Monte Carlo Integration**. Consider the following integral

$$I = \int_{-\pi}^{\pi} \frac{(1 + \cos(x)) \sin(|2x|)}{1 + |\sin(2x)|} dx \tag{6.11}$$

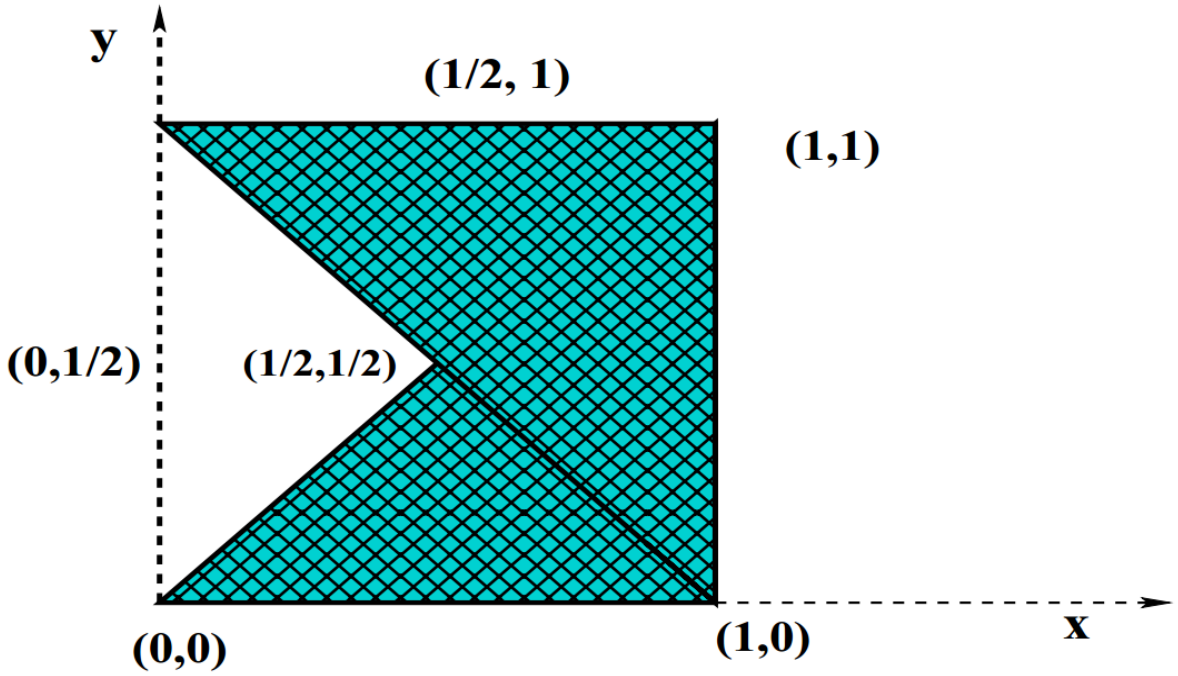
(a) Generate a sequence of pair of random numbers (x, y) with $x \in [-\pi, \pi]$and $y \in [-0.3, 0.9]$. Write a C or C++ function **double integral2(int N)** to calculate the value of the integral by comparing the area under the curve $y = \frac{(1+\cos(x))\sin(|2x|)}{1+|\sin(2x)|}$ and that of the rectangular region with sides 2π and 1.2 using

N pairs of random numbers, i.e. N points.

(b) Increase the number of points used to evaluate the area and find the errors in each case. Take $N \in [10, 107]$. The approximate value of the integral is 1.5070990394390779464.

(c) Write the number of points and the errors in two columns of a file. Plot the error vs the number of points used and the save the plot as a postscript file.

C.N.6.1.10 (a) Write a C or C++ program that evaluates the area of the filled/shaded region in the following figure using random numbers: You may use the built-in random number generator in C i.e. *rand()* and divide



it by the maximum integer random number RAND_MAX to get a random number between $(0, 1)$ for this purpose. Seed the *rand()* function by calling the *srand(seed)* function with a seed.

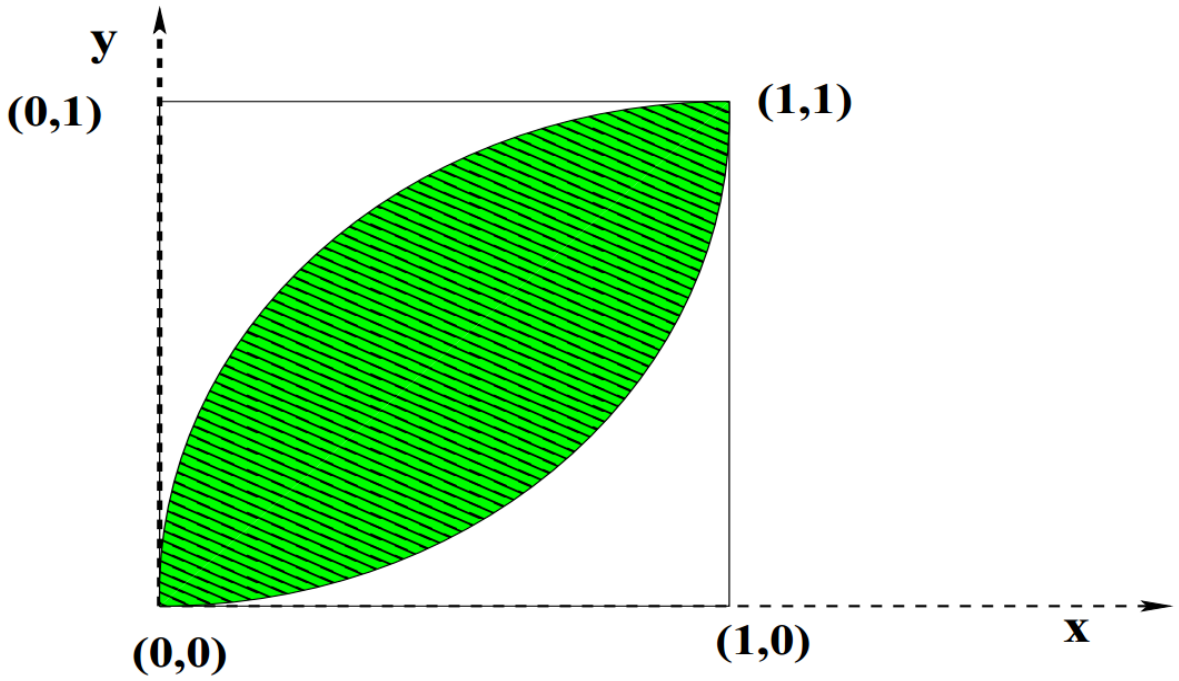
(b) Increase the number of points used to evaluate the area and find the errors in each case.

(c) Write the number of points and the errors in two columns of a file. Plot the error vs the number of points used and the save the plot as a postscript file.

C.N.6.1.11 (a) Write a C or C++ program that evaluates the area of the filled/shaded region in the following figure using random numbers: You may use the built-in random number generator in C i.e. *rand()* and divide it by the maximum integer random number RAND_MAX to get a random number between $(0, 1)$ for this purpose. Seed the *rand()* function by calling the *srand(seed)* function with a seed.

(b) Increase the number of points used to evaluate the area and find the errors in each case.

(c) Write the number of points and the errors in two columns of a file. Plot the error vs the number of points used and the save the plot as a postscript file.



6.2 Trapezoidal rule, Simpson's rule and Romberg method

C.N.6.2.1 Compute the following integrals

$$I_1 = \int_{\pi/4}^{\pi/2} \frac{\cos(x)\ln(\sin(x))}{1+\sin(x)} dx \tag{6.12}$$

$$I_2 = \int_{0}^{3\pi/8} \tan(x) dx \tag{6.13}$$

using Simpson's-3/8-rule and Simpson's 1/3 rule and compare the results.

C.N.6.2.2 The **Laplace's method for evaluating integrals** of the type $\int_a^b \exp(Mf(x))dx$, where $f(x)$ is an at least twice-differentiable function with a global maximum, and M is a large number is given by:

$$\int_a^b \exp(Mf(x))dx \approx exp(Mf(x_0))\sqrt{\frac{2\pi}{M\,|f''(x_0)|}} \tag{6.14}$$

where at x_0, the function $f(x)$ has a global maximum and hence $f'(x_0) = 0$.

(a) Define a function *f(x)* in C or C++ as **double f(double x)** which is given by

$$f(x) = \frac{sin(x)}{x} = \begin{cases} 1 & \text{if } x = 0 \\ \sin(x)/x & \text{if } x \neq 0 \end{cases} \tag{6.15}$$

(b) Using any suitable numerical integration scheme, write a C or C++ function **double numlpc(double M)** that evaluates the above integral $\int_a^b \exp(Mf(x))dx$ for a given value of the positive parameter M, between the limits $a = -10$, $b = 10$. You will need the above definition of the function **double f(double x)**.

(c) Write another function **double aprlpc(double M)** that evaluates the above approximate value of the integral for the positive parameter M. Use $f(x_0) = 1$, $f''(x_0) = -1/3$.

(d) Call the above two functions *numlpc(M)* and *aprlpc(M)* in a complete C or C++ program and write the values of x in the first column and the function values *numlpc(M)* and *aprlpc(M)* in consecutive columns

in a file. Plot *numlpc(M)* and *aprlpc(M)* versus x using gnuplot in a single plot and save the plot as a postscript file.

C.N.6.2.3 The **Dawson integrals** appear in physics in various fields including spectroscopy, heat conduction, electrical conduction, propagation of electromagnetic radiation along the earth's surface, etc. The Dawson integrals are defined as

$$D_+(x) = F(x) = \exp(-x^2) \int_0^x \exp(t^2) dt \tag{6.16}$$

$$D_-(x) = \exp(x^2) \int_0^x \exp(-t^2) dt \tag{6.17}$$

$$\tag{6.18}$$

(a) Write two C or C++ functions **double dawpos(double x)** and **double dawneg(double x)** that returns the Dawson integrals $D_+(x)$ and $D_-(x)$, respectively, at some points x, using any suitable numerical integration scheme. Take suitable number of grid points.

(b) In a complete C or C++ program write the values of x in the first column and the values of the functions $D_+(x)$ in the second and $D_-(x)$ in third column. Plot, using gnuplot the functions $D_+(x)$ in the range $x \in [-6, 6]$ and $D_-(x)$ in the range $x \in [-2, 2]$ using the above C/C++ functions. Save the plots as postscript files.

C.N.6.2.4 Using **Simpson's-1/3** rule, integrate the function given below from $-pi$ to π, you have to write a C++ function to do the integration and this function has to be general enough to take any integrand, lower and upper limits and the number of subintervals.

$$F(x) = (\exp(-|x|/\pi) + \exp(|x|/\pi) \sin(x) dx \tag{6.19}$$

Now, use **Romberg's trick** to improve your result obtained from the Simpson's-1/3 rule.

C.N.6.2.5 Use a suitable numerical integration scheme to evaluate the following integrals:

$$I_1 = \int_0^{\pi/2} \frac{1}{1 + \sin(x)} dx \quad \text{Ans : } 1 \tag{6.20}$$

$$I_2 = \int_3^5 \frac{1}{\sqrt{x^2 - 4}} dx \quad \text{Ans : } 0.6043755868532041837 \tag{6.21}$$

$$I_3 = \int_{-2}^2 x^3 e^x dx \quad \text{Ans : } 19.920852960852582746 \tag{6.22}$$

$$I_4 = \int_{0.5}^{1.5} \frac{2 - 2x + \sin(x - 1) + x^2}{1 + (x - 1)^2)} dx \quad \text{Ans : } 1 \tag{6.23}$$

C.N.6.2.6 The period of a **simple pendulum** for large angle amplitude (θ_M) is given in terms of the **complete elliptic integral of the first kind** $K(m)$ as

$$T = 4 \left(\frac{L}{g}\right)^{1/2} K\left(\sin^2(\frac{\theta_M}{2})\right) = 4 \left(\frac{L}{g}\right)^{1/2} \int_0^{\pi/2} \left(1 - \sin^2\left(\frac{\theta}{2}\right) \sin^2(\phi)\right)^{-1/2} d\phi \tag{6.24}$$

where

$$K(x) = \int_0^{\pi} \frac{1}{\sqrt{(1 - x \sin^2(\phi))}} d\phi \tag{6.25}$$

where $0 \leq \theta_M \leq \pi$.

(a) Using **Simpson's 1/3-rd rule**, write a C++ function **double pendulumT(double thetaM)**, that evaluates the period T for a given θ_M as the argument using the above definition. Choose the value of

L/g such that, as $\theta_M \to 0$, $T = 1s$.

(b) Call the function *pendulumT()* from the *main()* function with different values of θ_M as input.

(c) Using the above function, plot T vs. θ_M for $\theta_M \in [0, \pi/2]$. Save the plot as a postscript file. Hint: Check values: $\theta_M = [10°, 50°, 90°] \Rightarrow T = [1.00193, 1.05033, 1.18258]$.

C.N.6.2.7 Write a C or C++ program to solve the following integral

$$\int_0^{3\pi/8} \tan(x)dx \tag{6.26}$$

(a) using **Trapezoidal rule**, **Simpson's 1/3**, **Simpson's 3/8** rule and **Romberg's trick**.
[*Ans.* 0.96054717892973049468]

(b) Write a function that will use loop to evaluate Integral using **Romberg's trick** for **Trapezoidal rule**, **Simpson's 1/3** and **Simpson's 3/8**.

C.N.6.2.8 The **electrostatic potential** at a point (x, y) on the $xy-$plane due to a uniform surface charge distribution ρ in the square region $-1 \leq x' \leq 1$, $-1 \leq y' \leq 1$ is given by

$$V(x, y) = \frac{\rho}{4\pi \in_0} \int_{-1}^{1} \int_{-1}^{1} \frac{1}{\sqrt{(x - x')^2 + (y - y')^2}} dx'dy' \tag{6.27}$$

(a) Write a C or C++ program that evaluates the electrostatic potential at the point (x, y) using any suitable numerical integration technique. Take $\rho/(4\pi \in_0) = 1$

(b) Save data from your program for $V(y, x)$ at the three values of $y = 1.5, 2.0, 2.5$ in the range $x \in (1.5, 10)$. Plot the data using gnuplot and save the plot as a postscript file.

C.N.6.2.9 **Debye's theory of solids** gives the heat capacity of a solid at temperature T to be

$$C_v = 9V\rho k_B \left(\frac{T}{\theta_D}\right) \int_0^{\theta_D/T} \frac{x^4 e^x}{(e^x - 1^2)} dx \tag{6.28}$$

where V is the volume of the solids, ρ is the number density of the atoms, k_B is the Boltzmann's constant, and θ_D is the so-called Debye temperature, a property of solids that depends on their density and speed of sound.

(a) Write a C++ function $C_v(t)$ that evaluates C_v for a given value of temperature, for a sample consisting of 1000 cubic centimeters of solid aluminium, which has a number density of $\rho = 6.022 \times 10^{28}\ m^{-3}$ and a Debye temperature of $\theta_D = 428\ K$. Use **Romberg** integration technique to evaluate the integral.

(b) Also evaluate the integral using **Simpson's 1/3** method and then compare the results.

C.N.6.2.10 Use suitable numerical integration scheme to evaluate the following integrals

$$I_1 = \int_3^5 \frac{dx}{\sqrt{x^2 - 4}} \quad \text{Ans: } 0.6043755868532041837\ldots \tag{6.29}$$

$$I_2 = \int_{-2}^{2} x^3 e^x dx \quad \text{Ans: } 19.920852960852582746\ldots \tag{6.30}$$

$$I_3 = \int_{0.5}^{1.5} \frac{2 - 2x + \sin(x - 1) + x^2}{1 + (x - 1)^2} dx \quad \text{Ans: } 1 \tag{6.31}$$

C.N.6.2.11 Compute the following integrals correct to 10 decimal places

$$I_1 = \int_{\pi/4}^{\pi/2} \frac{\cos(x) \ln \sin(x)}{1 + \sin(x)} dx \quad \text{Ans : -0.0265799631930357879 8...} \tag{6.32}$$

$$I_2 = \int_0^{3\pi/8} \tan(x) dx \qquad \text{Ans : } 0.96054717892973049468... \tag{6.33}$$

(i) Using Trapezoidal rule and Romberg's trick.
(ii) Using Simpson's 1/3 rule and Romberg's trick.
(iii) Using Simpson's 3/8 rule and Romberg's trick.

shahnoor3pl@gmail.com

Chapter 7

Root finding

C.N.7.0.1 Neutron transport theory

In neutron transport theory the critical length of a fuel rod in a reactor is determined by the roots of thr equation:

$$cot(x) = (x^2 - 1)/(2x) = (x - 1/x)/2 \tag{7.1}$$

(a) Using gnuplot plot $cot(x)$ (x in radians) and on the same plot, graph $(x^2 - 1)/(2x)$. Plot the functions with different colored lines and save the plot as a postscript file. How many intersections are there?

(b) Using any suitable root finding method, write a C or C++ program that finds the the smallest positive root of this equation. Does your answer tally with the root(s) that you get from the inspection of the plot above?

C.N.7.0.2 In circuit with a diode and a resistor of resistance 10Ω connected in series with a dc voltage source of $5V$, the current in the diode is modelled as $i = i_s(\exp(-V/(\eta V_T)) - 1)$, where V is the potential difference across the diode, is is the reverse saturation current equal to $8.3 \times 10^{-6} A$, η is a parameter, taken as $\eta = 2$ and V_T is the knee voltage equal to $0.7V$ for silicon semiconductor-diode. To solve for the current in the circuit, we need to apply **Kirchhoff's voltage law** around the circuit which gives

$$-5 + 1.4\ln(i/i_s + 1.0) + 10.0i = 0 \tag{7.2}$$

(a) Write a C or C++ program that solves for the current in the circuit in the forward bias case (i.e. the above equation) using any suitable root finding method.

(b) Modify the program such that the value of the current is written in a file for successive iterations in the root finding algorithm. Write the number of iteration in the first column, the value of the current in the previous step in the second column and the value of the current in the current iteration/step in the third column of a file.

(c) Plot the values of $|i_n - i_{n-1}|$ vs n, using gnuplot, where n is the iteration number. Save the plot as a postscript file.

C.N.7.0.3 The **force between the Na+ and Cl-** ions can be modeled as

$$f(r) = -\frac{q^2}{4\pi \in_0 r^2} + \frac{V_0}{r_0}\exp(-r/r_0) = -132.276(r_0^2/r^2) + 3303.03\exp(-r/r_0) \tag{7.3}$$

where $q^2/(4\pi \in_0) = 14.4\mathring{A}\ eV$, $V_0 = 1090$eV and $r_0 = 0.33\mathring{A}$.

(a) Plot $f(x)$ vs. $x(= r/r_0)$ in the range $x \in [0.15, 10]$ and save the plot as a postscript file.

(b) Using the Newton-Raphson method or Secant method, write a C++ program that finds the equilibrium length between the Na+ and Cl- ions. Does your answer tally with the root(s) that you get from the inspection of the plot above?

C.N.7.0.4 The magnetization m in the **Ising model** in the mean field approximation is given by the transcendental equation

$$m = \tanh\left(\frac{Jm}{T}\right) \tag{7.4}$$

where J is called the exchange interaction parameter and T is the temperature. Take $J = 100$, $T = 10$, both in their respective m.k.s. unit. For for $T > J$ the only solution is $m = 0$ but for T below J there are two other solutions, with equal and opposite magnetization.

(a) Plot m and $\tanh\left(\frac{Jm}{T}\right)$ in the same plot and save it as a postscript file.

(b) Write a C++ program to find the non-zero root (either positive or negative) of the above transcendental equation using NewtonRaphson root finding algorithm. Use the plot from part (a) to determine an initial guess. Note that, in Newton-Raphson method, you have to use the exact derivative of the function.

C.N.7.0.5 Using **bisection method**, find the root of

$$xe^x = 1 \text{ with } x_0 = (1, 2) \tag{7.5}$$

$$x^3 - 6x^2 + 11x - 6 = 0 \text{ wtih } x_0 = (0, 2) \tag{7.6}$$

$$e^x = x^2 \text{ with } x_0 = (0.5, 1) \tag{7.7}$$

You have to write your own C++ function for the root and this function has to be general enough to take any function, initial guesses and the number of iterations.

C.N.7.0.6 In a certain Quantum Mechanical problems of **particles moving in a potential well**, the energies are given by the transcendental equation

$$-\cot(x) = \frac{1}{x}\sqrt{2mV_0(\frac{a^2}{\hbar^2}) - x^2} \tag{7.8}$$

where $2mV_0\left(\frac{a^2}{\hbar^2}\right) = 10.498597$ and $E = 0.952508225x^2 - 10$

(a) Plot $-\cot(x)$ and $\frac{1}{x}\sqrt{2mV_0(\frac{a^2}{\hbar^2}) - x^2}$ in the same plot and save it as a postscript file.

(b) Write a C or C++ program to find the lowest bound state i.e. energy eigenvalue from the above equations using any suitable root finding algorithm. [Newton-Raphson method Ans. $E = -4.80098$ and Bisection finding method Ans. $E = -4.67299$!?!?!?]

C.N.7.0.7 In **fluid dynamics**, a relationship between the Mach number M and the flow area A is given by the **Zucrow-Hoffman law** as

$$\varepsilon = \frac{A}{A^*} = \frac{1}{M}\left[\frac{2}{\gamma+1}\left(1 + \frac{\gamma-1}{2}M^2\right)\right]^{(\gamma+1)/(2(\gamma-1))} \tag{7.9}$$

where A^* is the choking area (i.e., the area where $M = 1$) and γ is the specific heat ratio of the flowing gas. For each value of ε, two values of M exist, one less than unity (i.e., subsonic flow) and one greater than unity (i.e., supersonic flow).

(a) Calculate both the values of M for $\varepsilon = 10.0$ and $\gamma = 1.4$ by Newton's method. For the subsonic root, let the initial guess $M_0 = 0.2$. For the supersonic root, let $M_0 = 5.0$.

(b) Now solve for the Mach numbers using bi-section method. Compare your results with those obtained from the Newton-Raphson's method.

Chapter 8

Ordinary Differential Equation

8.1 Euler's method and Improved Euler's method (RK-2)

C.N.8.1.1 SHO

The Euler's method for solving the second order ordinary differential equation

$$\frac{d^2x}{dt^2} + \omega^2 x = 0 \tag{8.1}$$

written as a set of coupled first order difference equations is

$$x_{n+1} = x_n + f(x_n, t_n)(t_{n+1} - t_n) = x_n + v_n h$$
$$v_{n+1} = v_n + g(v_n, x_n, t_n)h = v_n - \omega^2 x_n h$$

Consider the initial conditions $x_0 = 0$, $v_0 = 1$.

(a) Write a C/C++ code that solves the above initial value problem. Your code should write the values of t, x and v in different columns of a file.

(b) Plot the values of $v = \frac{dx}{dt}$ vs. x for $\omega = 1, 4, 9, 16$.

C.N.8.1.2 Many modifications of the **Lotka-Volterra predator-prey model** have been proposed to more accurately reflect what happens in nature. In one such model, the number of rabbits r and foxes f are assumed to be governed by the following equations, in which the number of rabbits can be prevented from growing indefinitely as :

$$\frac{dr}{dt} = 2\left(1 - \frac{r}{R}\right)r - \alpha r f \tag{8.2}$$
$$\frac{df}{dt} = -f + \alpha r f \tag{8.3}$$

where $r(0) = r_0$ and $f(0) = f_0$.
Since $\alpha > 0$, $\frac{dr}{dt}$ is negative whenever $r \geq R$. Consequently, the number of rabbits can never exceed R, a maximum value.

(a) Write a C or C++ program that solves the above coupled differential equations to get r and f as functions of the the time t. Take, $\alpha = 0.01$, $R = 400$, $r_0 = 300$ and $f_0 = 150$. Write the values of t, r and f on the first three columns of a datafile.

(b) Plot (i) f and r versus time t and (ii) r versus f for 20 cycles using gnuplot and save the plots as postscript files.

(c) Now call the program with the r/R term in the first equation dropped and redraw the above two plots. What are the differences between the two cases?

C.N.8.1.3 The **predator-prey** equations for the prey and predator populations are:

$$\frac{dx}{dt} = ax - \alpha xy \tag{8.4}$$

$$\frac{dy}{dt} = -cy + \gamma xy \tag{8.5}$$

where, x and y are the populations of the prey and predator respectively, at time t and a is the growth rate of the prey and c is the death rate of the predator and α and γ are measures of the effect of the interaction between the two species. consider the initial conditions $x_0 = 1.0$ and $y_0 = 0.5$ and the parameter values $a = 0.5$, $\alpha = 0.05$, $c = 1.0$, $\gamma = 0.9$.

(a) Write a C++ program to solve the above differential equation using either the improved Euler method or the 4th order Runge-kutta method.

(b) Plot $x(t)$ and $y(t)$ vs time t for at least a few periods of the population change. Plot x vs y. Save the plots as postscript files and interpret your results.

C.N.8.1.4 Consider the damped harmonic oscillator equation

$$\frac{d^x}{dt^2} + 0.1\frac{dx}{dt} + 16x = 0 \tag{8.6}$$

The initial conditions are given by: $x(0) = 1$ and $x'(0) = 0$.

(a) Solve the above differential equation using any suitable numerical algorithm.

(b) Plot the displacement x vs. time t graph for at least two full oscillations.

(c) Plot the velocity $v = \frac{dx}{dt}$ vs. time t graph for the same interval as above.

C.N.8.1.5 In Fluid dynamics, the **Navier-Stokes equations** may be reduced to a set of three coupled first order ordinary differential equations, known as the Lorenz Model:

$$\frac{dx}{dt} = \sigma(y - x) \tag{8.7}$$

$$\frac{dy}{dt} = -xz + rx - y \tag{8.8}$$

$$\frac{dz}{dt} = xy - bz \tag{8.9}$$

where, x, y, z are called the Lorenz variables, derived from the temperature, density, and velocity variables in the original Navier-Stokes equations. The parameters σ (Prandtl number), r(Rayleigh number) and b (Physical proportion) are measures of the temperature difference across the fluid and other fluid parameters. Take, $\sigma = 10$ and $b = 8/3$, which correspond to cold water.

(a) Write a C or C++ program that solves the above coupled differential equations to get x, y and z as functions of the the time t. Take, the initial conditions $x_0 = 1$, $y_0 = z_0 = 0$ and the parameter $r = 5$. Write the values of t, x, y and z on the first four columns of a datafile.

(b) Plot (i) z versus time t and (ii) z versus x for $t \in [0, 100]$ using gnuplot for three values of $r = 5, 10, 25$. Save the plots as postscript files. Which plot is qualitatively different from the other two for one particular value of r?

C.N.8.1.6 A **lunar lander is falling freely** toward the surface of the moon. If $x(t)$ represents the distance of the lander from the center of the moon (in meters, with t in seconds), then $x(t)$ satisfies the initial-value problem

$$\frac{d^2x}{dt^2} = 4 - \frac{k}{x^2} \tag{8.10}$$

where $k = 4.9044 \times 10^{12}$ is a constant and and the initial values are $x(0) = 1,781,870m$ from the moon's center, $\dot{x}(0) = -450m/s$.

(a) Write a C or C++ program that solves the above differential equation for $x(t)$ using any suitable numerical method. [One of the best option is to use RK-4 method]

(b) Write the values of $x(t)$ and $x'(t)$ vs t in a file for $t > 0$ and for long enough such that the craft lands on the surface. Plot x and x' as functions of t using gnuplot.

(c) Change your code such that it will print out on the screen the time t_f taken for the craft to land on the moon (i.e. when $x(t) = 1,740,000$) and the lander's velocity at touchdown.

C.N.8.1.7 The time evolution of the concentration of two chemical species, x and y in the **Belousov-Zhabotinski** reaction, is described by the equations

$$\frac{dx}{dt} = A + x^2y - (B+1)x \tag{8.11}$$

$$\frac{dy}{dt} = Bx - x^2y \tag{8.12}$$

where $A = 0.9$, $B = 0.65$ and the initial conditions are $x(0) = 0.5$ and $y(0) = 0.8$.

(a) Write a C++ program to solve the above two coupled differential equations by any suitable method.

(b) Plot (i) x vs. t, (ii) y vs. t and (c) $y(t)$ vs. $x(t)$ for $t \in [0, 10]$. What are the saturation concentrations?

C.N.8.1.8 A **body falling** through a resistive medium is described by

$$\frac{dv}{dt} = g - av \tag{8.13}$$

where $v = v(t)$ is the velocity of the body, t is the time and $g = 9.8m/s^2$, $a = 0.2s^{-1}$. 2 (a) Solve the above differential equation using any suitable numerical algorithm. Take the initial condition as v(0) = 0.

(b) Write the values of the velocity $v(t)$ for time $t = 0$ s to $t = 40s$, in a file and plot v vs. t. From the plot, approximately determine the terminal velocity.

C.N.8.1.9 Consider the **Schrödinger equation** :

$$\frac{d^2\Psi}{dx^2} + V(x)\Psi(x) = 0 \tag{8.14}$$

(a) Write a C++/C program that solves the differential equation of the above type given the values of function Ψ at the first two initial grid points i.e. Ψ_0 and Ψ_1. Take $V(x) = 2m(E + V_0)$ for $|x| > a$ and $V(x) = 2mE$ for $|x| < a$, where $m = 1$, $E = 2$, $V_0 = 1$ and $a = 1.5$ and use any physically meaningful values of Ψ_0 and Ψ_1. [Hint : use Runge-Kutta method of 4-th order]

(b) Modify your code to write the values of the position x and that of the wave function $\Psi(x)$ in a file. Plot $\Psi(x)$ vs x for $x \in [-2a, 2a]$ and save the plots.

8.2 Runge Kutta method of 4-th Order

C.N.8.2.1 The differential equation of motion of a particle of unit mass moving in an **anharmonic potential** is given by

$$\frac{d^2x}{dt} + kx\,(1 + \exp(-x/a)) = 0 \tag{8.15}$$

where $k = 1.0$ and $a = 0.05$. Consider the initial conditions $x(0) = 1$ and $v(0) = \dot{x}(0) = 0$. (a) Using the fourth order **Runge-Kutta (RK4)** method solve the above differential equation (you can not apply Euler's method for this problem).

(b) Plot the position x vs time t graph and from there determine the time period of oscillations when the maximum amplitude equals to (i) $x(0) = 0.1$ (ii) $x(0) = 1$ and (iii) $x(0) = 10$. Save the plots for each case as postscript files.

(c) Can we make a good clock using a mass connected to a such a "spring" ? For a simple harmonic oscillator, does the period depend on the amplitude of oscillation? Justify your claim.

C.N.8.2.2 (a) The differential equation of motion of a particle of unit mass moving in an **anharmonic potential** is given by

$$\frac{d^2x}{dt^2} + kx + ax^3 = 0 \tag{8.16}$$

where $k = 1.0$ and $a = 0.05$. Consider the initial conditions $x(0) = 1$ and $v(0) = \dot{x} = 0$.

(a) Using the fourth order Runge-Kutta (RK4) method solve the above differential equation (you can not apply Euler's method for this problem).

(b) Plot the position x vs. time t graph and from there determine the time period of oscillations when the maximum amplitude equals to (i) $x(0) = 0.1$ (ii) $x(0) = 1$ and (iii) $x(0) = 10$. Save the plots for each case as postscript files.

(c) For a simple harmonic oscillator, does the period depend on the amplitude of oscillation? Justify your claim.

C.N.8.2.3 Suppose that for a **mass-spring system**, the force is given by the usual spring force $F = -kx$ but the inertia is dependent on the speed i.e. $m = M\exp(v/\omega)$. The differential equation of motion of such an oscillator is given by

$$\frac{d^2x}{dt^2} + (\frac{k}{M})x\exp\left((-dx/dt)/\omega\right) = 0 \tag{8.17}$$

where $v = dx/dt$ is the speed of the particle, ω is a characteristics speed of the particle in the system. Take $M = 1.0$,$k = 1.0$ and $\omega = 5.0$. Consider the initial conditions $x(0) = 1$ and $v(0) = \dot{x}(0) = 0$.

(a) Using the fourth order **Runge-Kutta (RK4)** method solve the above differential equation (you can not apply Euler's method for this problem).

(b) Plot the position x vs. time t graph and from there determine the time period of oscillations when the maximum amplitude equals to (a) $x(0) = 0.1$ (b) $x(0) = 1$ and (c) $x(0) = 10$. Save the plots for each case as postscript files.

(c) Can we make a good clock using a mass connected to a such a "mass"-spring system ? For a simple harmonic oscillator, does the period depend on the amplitude of oscillation? Justify your claim.

C.N.8.2.4 Using **Runge-Kutta** method of 4th order solve the following 2nd order differential equation Consider the following second order ordinary differential equation :

$$\frac{d^2x(t)}{dt^2} = -\cos(2t)x(t) + \sin(2t) \tag{8.18}$$

The initial conditions are given by $x(0) = 0.2$, $x'(0) = 0.0$. (a) Write a C++ program applying Runge-Kutta algorithm to solve the above differential equation.

(b) Plot the displacement x vs. time t and v vs. t graph for the interval $t \in [0.0, 10.0]$ and save it as a postscript file.

C.N.8.2.5 The **flux in a transformer** ϕ satisfies the differential equation

$$\frac{d^2\phi}{dt^2} + \omega_0^2\phi + b\phi^3 = \frac{\omega}{N}E\cos(\omega t) \tag{8.19}$$

where $E\cos(\omega t)$ is the sinusoidal source voltage, N is the number of turns in the primary winding, and ω_0 and b are parameters of the transformer design.

(a) Write a C++ program that solves the above differential equation for ϕ using the fourth order Runge-Kutta (RK4) method.

(b) Write the values of ϕ and $\dot{\phi}$ as a function of t in a file for $t \in [0, 5T]$ where T is the period of the signal. Take $E = 165$, $\omega = 120\pi$, $N = 600$, $\omega_0^2 = 83$ and $b = 0.14$. Use any suitable initial conditions. Plot ϕ and $\dot{\phi}$ as functions of t is the range mentioned above.

(c) Repeat (a) and (b) for the source term $E\sin(\omega t)$

C.N.8.2.6 A pair of **chemical reactions**, $A \to B$ and $B \to C$ take place in a batch reactor, starting with pure A at a concentration $C_A(0) = 1.00$mol/liter. The following equations describe how the concentrations $C_A(t)$, ,$C_B(t)$, and $C_C(t)$ vary with time (sec). The notation $CA0$ will be used to represent the derivative dCA/dt.

$$C'_A(t) = -0.1C_A(t) \tag{8.20}$$
$$C'_B(t) = 0.1C_A(t) - 0.2C_B(t) \tag{8.21}$$
$$C'_C(t) = 0.2C_B(t) \tag{8.22}$$

with initial conditions $C_A(0) = 1.0$, $C_B(0) = 0.0$, $C_C(0) = 0.0$ Write a C++ program that will solve the above set of ODEs using Runge-Kutta (RK4) method. Plot CA,CB,CC at times from $t = 0$sec to $t = 40$sec on the same plot.

8.3 Numerov's method

Numerov's method is efficient for solving equations of the form $y''(t) + F(t)y(t) = G(t)$. And if $y(0) = y0$ and $y'(0) = y'_0$ are known then

$$y_{n+1} = \frac{(2 - \frac{5}{6}h^2F_n)y_n - \left(1 + \frac{h^2}{12}F_{n-1}\right)y_{n-1} - \frac{h^2}{12}\left(G_{n-1} - 10G_n - G_{n+1}\right)}{\left(1 + \frac{h^2}{12}F_{n+1}\right)} \tag{8.23}$$

One way to start the Numerov's method is

$$y_1 = \frac{y_0\left(1 + \frac{F_2h^2}{24}\right) + hy'_0\left(1 + \frac{F_2h^2}{24}\right) + \frac{h^2}{24}(7G_0 - 7F_0y_0 + 6G_1 - G2) + \frac{h^4F_2}{36}(G_0 - F_0y_0 + 2G_1)}{\left(1 + \frac{F_1h^2}{4} + \frac{F_1F_2h^4}{18}\right)} \tag{8.24}$$

C.N.8.3.1 Consider an oscillator :

$$\frac{d^2u(t)}{dt^2} = -u(t) + \sin(t) \tag{8.25}$$

The initial conditions are given by $u(0) = 1.0$, $u'(0) = -5.0$.

(a) Write a C++ program applying Numerov's algorithm to solve the above differential equation.

(b) Plot the displacement x vs. time t graph for the interval $t \in [0.0, 20.0]$ and save it as a postscript file.

C.N.8.3.2 Consider the following second order ordinary differential equation :

$$\frac{d^2x(t)}{dt^2} = -\cos(2t)x(t) + \sin(2t) \tag{8.26}$$

The initial conditions are given by $x(0) = 0.2$, $x'(0) = 0.0$. (a) Write a C++ program applying Numerov's algorithm to solve the above differential equation.

(b) Plot the displacement x vs. time t graph for the interval $t \in [0.0, 10.0]$ and save it as a postscript file.

C.N.8.3.3 Consider the following second order ordinary differential equation :

$$\frac{d^2\phi(t)}{dt^2} + 2t\phi(t) = t^3 \tag{8.27}$$

The initial conditions are given by $\phi(0) = 2.0$, $\phi'(0) = -3.0$.

(a) Write a C++ program applying Numerov's algorithm to solve the above differential equation.

(b) Plot the displacement ϕ vs. time t graph for the interval $t \in [0.0, 10.0]$ and save it as a postscript file.

C.N.8.3.4 Consider the **Schrödinger equation** :

$$\frac{d^2\Psi}{dx^2} + V(x)\Psi(x) = 0 \tag{8.28}$$

(a) Write a C++/C **numerov** of double type that solves the differential equation of the above type given the values of function Ψ at the first two initial grid points i.e. Ψ_0 and Ψ_1. Take $V(x) = 2m(E + V_0)$ for $|x| > a$ and $V(x) = 2mE$ for $|x| < a$, where $m = 1$, $E = 2$, $V_0 = 1$ and $a = 1.5$ and use any physically meaningful values of Ψ_0 and Ψ_1.

(b) Modify your code to write the values of the position x and that of the wave function $\Psi(x)$ in a file. Plot $\Psi(x)$ vs x for $x \in [-2a, 2a]$ and save the plots.

Chapter 9

Miscellaneous

9.1 Quantum Mechanics

C.N.9.1.1 A **square potential well and the transmission coefficient** for it are given by

$$V(x) = \begin{cases} 0 & \text{if } x < 0 \\ -V_0 & \text{if } 0 < x \geq a \\ 0 & \text{if } x > a \end{cases} \tag{9.1}$$

$$T = \left[1 + \frac{1}{4\epsilon(\epsilon+1)} \sin^2(\alpha\sqrt{1+\epsilon})\right]^{-1} \tag{9.2}$$

where $\alpha = (2mV_0a^2/\hbar^2)^{1/2}$, $\epsilon = E/V_0$.

(a) Using gnuplot, plot the transmission coefficient T and the reflection coefficient $R = 1 - T$, as functions of the reduced energy $E = E/V_0$ for $E \in [0, 2]$in the same plot. Save the plot as a postscript file.

(b) Write a C or C++ program that finds the resonant energies E which will give perfect transmission i.e. $T = 1$, with $\alpha = 25.598$ and $V_0 = 25\ eV$, using any suitable root finding algorithm. Find the first four energy values. [Hint: Resonant values are $E \approx 0.22, 0.51, \ldots$]

C.N.9.1.2 For a **square-well potential** given by

$$V(r) = \begin{cases} -V_0 & \text{for } r < R \\ 0 & \text{for } r > R \end{cases} \tag{9.3}$$

the energy eigenvalues $|E| = V_0(1 - \xi^2)$, lying in the range $0 < E < V_0 \rightarrow 0 < \xi < 1$, are given by the equation:

$$\tan(x_0\xi) = f_l(x_0, \xi) \tag{9.4}$$

where $l = 0, 1, 2, \ldots$ is the angular momentum quantum number and the functions $f_l(x_0, \xi)$ for $l = 0, 1$ are given by

$$f_0(x_0, \xi) = -\frac{\xi}{\sqrt{1-\xi^2}} \tag{9.5}$$

$$f_1(x_0, \xi) = \frac{x_0\xi}{1 + (\xi^2/(1-\xi^2))\left(1 + x_0\sqrt{1-\xi^2}\right)} \tag{9.6}$$

(a) Plot the three functions $\tan(x_0\xi)$, $f_0(x_0, \xi)$ and $f_1(x_0, \xi)$, in the same plot for $x_0 = 10$ and in the range $0 < \xi < 1$ and save it as a postscript file. Besides the trivial case for $\xi = 0$, there should be three points of intersection, between the two graphs of $\tan(x_0\xi)$ and $f_l(x_0, \xi)$ for both $l = 0, 1$. Take

$x_0^2 = 2mV_0R^2/\hbar^2 = 100$.

(b) Write a C or C++ program to find all the six possible energy states taking $V_0 = 1$ using any suitable root finding algorithm. [Hint:A few roots of the equation $\tan(x_0\xi) = f_l(x_0, \xi)$ are near $\xi = 0.28, 0.56, 0.84$ for $l = 0$.]

C.N.9.1.3 The energy eigenvalues E for a particle of mass m in a **one-dimensional square-well potential** of half-size a and depth V_0 satisfy the following transcendental equation

$$\sqrt{2mE + V_0}\tan(\sqrt{2m(E + V_0)})\frac{a}{\hbar} - \sqrt{-2mE} = f(E) = 0 \tag{9.7}$$

The parameter values are chosen to be representative of the energy levels of a neutron or proton in a nucleus. Take $\hbar = 1$, $m = 938(MeV)$, $a = 197.3(MeV)$, $V_0 = 70(MeV)$ and note that E should be negative for bound system.

(a) Using gnuplot plot $\sqrt{2mE + V_0}\tan(\sqrt{2m(E + V_0)})\frac{a}{\hbar}$ and $\sqrt{-2mE}$ with respect to $E \in [-4, 0]$ in the same plot and save it as a postscript file. Can you identify the singularities of the functions? Change the x-range of your plot to identify the first two solutions of the above equation.

(b) Write a C or C++ program to find the lowest bound state i.e. energy eigenvalue E from the above equations using any suitable root finding algorithm.

9.2 Nuclear Physics

C.N.9.2.1 The **semi-empirical mass formula** for nuclear binding energy is given by:

$$B = aA - bA^{2/3} - s\frac{(A - 2Z)^2}{A} - \frac{dZ^2}{A^{1/3}} - \frac{\delta}{A^{1/2}} \tag{9.8}$$

where $a = 15.835(MeV)$, $b = 18.33(MeV)$, $s = 23.20(MeV)$, $d = 0.714(MeV)$, $\delta = 0$ for even-odd nuclei, $+11.2(MeV)$ for odd-odd nuclei and $-11.2(MeV)$ for eevn-even nuclei.

(a) Write a C or C++ function **double getA(double B, int Z)** that will give the mass number A given the binding energy B and the atomic number Z. Use any suitable root-finding method.

(b) Use the above function in a complete C or C++ program to estimate the mass numbers of deuteron and tritium given their binding energies as $B = 2.224589(MeV)$ and $B = 8.481821(MeV)$, respectively.

9.3 Statistical Mechanics

C.N.9.3.1 Atoms in an ideal classical gas follow the **Maxwellian velocity distribution** given by

$$f(v) = \left(\frac{m}{2\pi kT}\right)^3 / 24\pi v^2 \exp(-mv^2/(2kT)) \tag{9.9}$$

$$f(x) = \sqrt{\frac{2}{\pi}}x^2\frac{\exp(-x^2/(2a^2))}{a^3} \tag{9.10}$$

where m = mass of the gas molecules, k = Boltzman constant, $a = \sqrt{kT/m}$ is the distribution parameter and $x = v$. The average speed and the root-mean-square speed are defined as

$$\bar{v} = < v > = \int_0^\infty vf(v)dv \tag{9.11}$$

$$v_{rms} = \sqrt{< v^2 >} = \left(\int_0^\infty v^2 f(v)dv\right)^{1/2} \tag{9.12}$$

For practical purpose, take the upper linit of the integration as $5a$ instead of ∞.

(a) Write a C or C++ function **void velocities(double a, double *vavg, double *vrms)**, that evaluates the average speed $< v >$ and the root-mean square speed $\sqrt{< v^2 >}$ as a function of a, using any suitable numerical integration scheme.

(b) Plot the average speed and rms speed with respect to a for $a \in [0.01, 2]$. Save the plots as postscript files. From the plots, find the formula for the average speed and rms speed in terms of a.

C.N.9.3.2 The **Van der Waals'** equation of state for a vapor is

$$\left(p + \frac{a}{v^2}\right)(v - b) = RT \tag{9.13}$$

which can be rearranged in the form:

$$Pv^3 - (Pb + RT)v^2 + av - ab = 0 \tag{9.14}$$

where P is the pressure, v is the specific volume, T is the temperature, R is the gas constant, and a and b are empirical constants.

(a) Write a C++ code that will find the specific volume v given the values of P, T, R, a, b. Use the ideal gas law, $Pv = RT$ to obtain the initial guess (or guesses) for the root.

(b) Plot Pv^3 and $(Pb + RT)v^2 - av + ab$ with respect to v and check from the plot whether the result you got is reasonable or not.

9.4 Optics

C.N.9.4.1 The **knife-edge diffraction pattern** is described by

$$I = 0.5T_0\{[C(u_0) + 0.5]^2 + [S(u_0) + 0.5]^2\} \tag{9.15}$$

where $C(u)$ and $S(u)$ are Fresnel integrals are defined as

$$C(x) = \int_0^x \cos\frac{\pi u^2}{2} du \tag{9.16}$$

$$S(x) = \int_0^x \sin\frac{\pi u^2}{2} du \tag{9.17}$$

Here I_0 is the incident intensity, I is the diffracted intensity and u_0 is proportional to the distance away from the knife edge (measures at right angles to the incident beam).

(a) Write a C or C++ function **double knedge(double u0, double I0)** that returns the diffracted intensity given the distance u_0 and I_0 as inputs.

(b) Calculate I/I_0 for u_0 varying from -1.0 to $+4.0$ in steps of 0.1 and write them in a file. Plot your results of I/I_0 vs u_0 from the datafile using gnuplot and save the file as a postscript file. (Check value: $u_0 = 1.0$ gives $I/I_0 = 1.259226$)

9.5 Mathematics

C.N.9.5.1 (a) Write a C or C++ function that can find the **GCD** (i.e. **Greatest Common Divisor**)

(b) Write a C or C++ function **LCM** (i.e. **Least Common Multiple**) of two numbers recursively.

(c) Now repeat (a) and (b) using loop.

9.6 Mixed

C.N.9.6.1 The area A inside the closed curve $y^2 + x^2 = \cos(x)$ is given by

$$A = 4\int_0^{\alpha} (\cos(x) - x^2)^{1/2} dx \tag{9.18}$$

where α is the positive root of the equation $\cos(x) = x^2$.

(a) Write a function **double getroot()** that returns the positive root of the above transcendental equation using any suitable root finding algorithm.

(b) Write a C + + function that evaluates the area A of the curve defined above.

(c) Use Romberg's trick to compute the area A with an absolute error less than 0.05 applied to the Trapezoidal rule.