Physics 441 Assignments

1. Comparison of First Derivative Calculations

One can calculate the first derivative of polynomials exactly, or by using a numerical approximation. We want to compare the results for the function f(x)=x2 at x=1. We define the difference between the exact and so-called “forward difference” numerical approximations as:



The assignment is to create a plot of Δh vs. h for values of h between 10-20 and 100, first by using the forward difference approximation, and then also by using the “centered difference” approximation, as well as the “improved centered difference approximation.

Use double precision variables in all cases.

1. Root-finding Using Newton’s Method

This assignment has multiple parts:

Begin by considering the polynomial:



which as we discussed in class, has three real roots at x=-2.4,0.5, and 1.5

1. Write a standalone C++ program which calculates the roots of this polynomial using the bisection algorithm. The program should fill an array of 10,000 data points between x=-4.0 and x=4.0. Then, the program should do a sparse search of this array, i.e. once every n\_search elements, to look for sign changes, and subsequently using the bisection algorithm to find the root. The program should also keep track of the approximate total number of instructions taken to find the three roots (n\_steps).
2. Create a plot, using a ROOT macro, of the number of steps taken (n\_steps) vs. the sparsification size (n\_search), for n\_search values between 1 and 10,000.
3. Deduce the source of the various patterns that you observe in part b). Hint: You might consider adjusting the number of data points in the original array, as well as the coefficients of the polynomial, slightly.
4. Create a single ROOT macro that both finds the roots of the polynomial, as well as creates the plot from part b)
5. Modify the macro created in part d) to allow for the possibility to find the roots of other polynomials or functions.
6. Curve Fitting Part I – Linear Fit – No Uncertainties

Write a program to fit the following data using a linear least-squares fitting algorithm. There are no uncertainties on the data points themselves. Your should calculate the fit parameters, as well as the uncertainties on the fit parameters.

X Y

1 1.6711

2 2.00994

3 2.26241

4 2.18851

5 2.33006

6 2.42660

7 2.48424

8 2.63729

9 2.77163

10 2.89610

11 2.89083

12 3.08081

13 3.05305

14 3.24079

15 3.36212

1. Curve Fitting Part II – Linear Fit – Uncertainties on the Data

Using the data from part I, assign an uncertainty, Δy, to each data point. Start with having equal uncertainties on each data point. Modify the program from part I to take into account the uncertainties. You should find that the fit parameters will not change in value, but the uncertainties on the fit parameters will now be somewhat larger.

Next, assign non-equal uncertainties. Now, you should find that both the fit parameters and the uncertainties on the fit parameters should change should both change.

1. Curve Fitting Part III – Higher Order Fits

Modify the code that you wrote for part II to be able to fit the data with a polynomial of arbitrary order. Include uncertainties on the data points, as well.

1. Curve Fitting Part IV – Non-Polynomial Fits

Write a program to fit the following data:

X Y ΔY

.038 0.050 .010

.194 0.127 .010

.425 0.094 .030

.626 0.2122 .030

1.253 0.2729 .030

2.500 0.2665 .020

3.740 0.3317 .010

The fitting function should be of the form:



Determine both A and B, as well as the uncertainties on A and B.

1. Differential Equations Part I – Projectile Motion

I am providing to you a program that will calculate the motion of a projectile under the influence of gravity, including air resistance. It uses the “forward derivative” method for calculating derivatives. The program does the following:

1. Takes as input the initial height above the ground, the initial speed, and the initial launch angle.
2. Calculates the path of the projectile (i.e. the (x,y) position of the particle) for the case of zero air resistance, and including air resistance.
3. Creates (x,y) plots of these two paths, for comparison.
4. Creates plots of x vs. t and y vs. t for both paths, for comparison.

The assignment is to take this program and modify it to do the following:

Include the effect of the object “bouncing” of the ground. In order to do this, you will have to make some assumptions:

* 1. The coefficient of restitution between the projectile and the ground should be, say, 0.30 in the y-direction and 0.90 in the x-direction.
  2. When a “bounce” happens, the projectile reverses its y-velocity, subject to the coefficient of restitution in the y-direction, and its x-velocity is reduced according to the coefficient of restitution in the x-direction.
  3. The object has to “stop” at some point – say when the total velocity has dropped below some threshold … use 0.5 m/s.

Run the code and produce plots for the following situation: h0=2m, v0=40 m/s, theta0=45 degrees.

1. Differential Equations Part II – Simple Pendulum

I am providing you with a program that will calculate the “path” of a simple pendulum over time in the case of zero air resistance. The final output is a plot of θ vs. t for several oscillations of the pendulum. The assignment is to modify the program to include air resistance. You should be able to take advantage of the algorithm that was used in the program of part 7 above.

Create a plot of theta vs. t for several oscillations for the case of theta0=45 degrees, m=1kg, radius of mass=0.10m, length of pendulum=9.807m.