Programming for Instrumentation

PHYS 4600 Fall 2021. Final Exam

Clone the PHYS4600F21EXAM repository from my github. Create a new branch with your name.

Once you have finished commit your changes and push back to github with the commit message “EXAM FINISHED”. ***Remember to push back to <u>your</u> branch not main or master.***

You only have to do 2 of the 3 problems. If you try all 3 I will give you the marks for your best 2. You may use any of the notes we have taken and you may also refer to or reuse your previous code from the course.

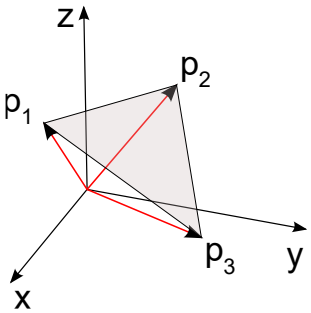
If you are unsure of the syntax for a particular command you may look it up on line. You may not look up specific algorithms, full code solutions etc.

Please include a comment at the top of each source file with your name.

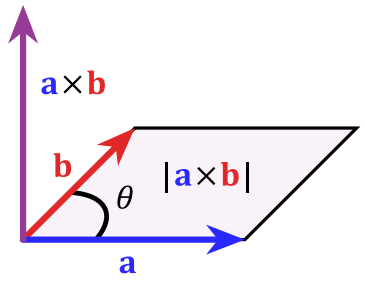
Problem 1: Vectors

For this question you will use the files in the folder "vectors" in the exam repo.

In this question you will write a program to find the area of a triangle given the position vectors of its 3 vertices in 3 dimensions.



To do so you can make use of the cross-product. Given two vectors a and b, the cross product is a vector axb which points perpendicular to both a and b and had a magnitude equal to the area of the parallelogram made by two vectors. But a and b also make two of the sides of a triangle which is half of the parallelogram.



So given 3 points (position vectors) of the two sides of the triangle, use vector subtraction to find vectors from one of the corners to the other 2 corners. Then find the magnitude of the cross-product of those two vectors and half it to get the area of the triangle formed by the 3 points.

For example in to find the area of the triangle in the first figure you would find the vectors from p1 to p2 and p1 to p3. The half the magnitude of the cross product of these two would be the area of the triangle.

The folder "vectors" contains a file "vectorops.c" and header "vectorops.h" as well as a "main.c" file and a makefile. Your task is to write and test a function in main.c which can calculate the area of a triangle given 3 points. Points should be stored as 3-element arrays of doubles.

To do this you will first have to write functions for some of the basic vector operations needed to do the calculation. These are in vectorops.c

To test your program you should find the area of several triangles. Have your program print out the point coordinates and areas of at least 3 triangles.

Math reminder

$$\vec{a} \times \vec{b} = \begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{k} \\ x_a & y_a & z_a \\ x_b & y_b & z_b \end{vmatrix} = (y_a z_b - y_b z_a)\hat{\imath} - (x_a z_b - x_b z_a)\hat{\jmath} + (x_a y_b - x_b y_a)\hat{k}$$

Problem 2: Monte-carlo integration

You may remember from integral calculus that the average value <f> of a mathematical function f between two values of x is given by:

$$\langle f \rangle = \frac{1}{b-a}\int_a^b f(x).dx$$

We can rearrange this:

$$(b-a)\langle f\rangle = \int_a^b f(x).\,dx$$

So if we could find the average value of the function in a different way on the left hand side, we could calculate the value of the definite integral.

In Montecarlo integration you calculate the f(x) for N random values of x and then take the mean of the results. So:

$$(b-a)\frac{1}{N}\sum_{i=0}^{N} f(x_i) \approx \int_a^b f(x).\,dx$$

Where the x_i are randomly chosen values for x between a and b. This method can be used to find the values of integrals which cannot be found in simple closed form. It is not very efficient for functions with only one variable but with multidimensional integrals of functions of multiple variables it is quite fast compared to other methods.

In the montecarlo folder there is a single c file.

Write a C function to do the integration between a and b for N values.

You will first need to write a function to generate a random double precision floating point x value between a and b. The stdlib.h function rand() produces an *integer* value between 0 and RAND_MAX. RAND_MAX is a constant also provided in stdlib.

The function you are integrating is in the function "function" – integrate it by hand between the limits 0 and pi to confirm your function is working. Determine how many N are needed for your function to converge to within 1% of the analytical solution.

Problem 3: Calculator in Python

For this problem. Create a simple calculator in Python. Use a Tk GUI with buttons for the numerals 0-9, the addition and subtraction operations, an "=" button and a clear button. Use a label for the display. As the user presses buttons the numerals should appear on the display.

Bonus for implementing: chained operators (i.e. 5 + 6 + 7 + ….) with the answer appearing after each new operator.