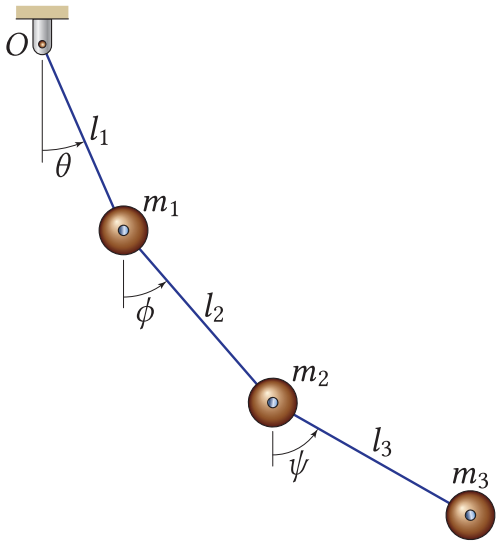
ESC 407
Computational Methods in Engineering Science
Homework 5
Numerical Quadrature

Due Date: Thursday, November 12 at 11:59 p.m.

Problem 1

(34 pts)

A triple pendulum is outfitted with accelerometers that measure the angular acceleration of each mass as it swings in the vertical plane. The pendulum is released with $\theta(0) = \pi/2$, $\phi(0) = \pi$, $\psi(0) = \pi/4$ $\dot{\theta}(0) = 0$, $\dot{\phi}(0) = 1\,\text{rad/s}$, and $\dot{\psi}(0) = -1\,\text{rad/s}$. The angular acceleration of each mass is measured 1000 times per second for six seconds and the data is written to a `.csv` file in which the first column is time, the second is $\ddot{\theta}$, the third is $\ddot{\phi}$, and the fourth is $\ddot{\psi}$. The file is named `accel_data.csv` and it can be found on the Homework 5 assignment page on Canvas.



Write a MATLAB function called `trap` that uses the trapezoidal rule to find the *cumulative* integral of data at n discrete points. The formula that you will use to get the cumulative integral for $t_0 < t_i \leq t_n$ is given by*

$$\int_0^x f(x)\,dx \approx y_0 + (x_2 - x_1)\frac{f(x_1) + f(x_2)}{2} + (x_3 - x_2)\frac{f(x_2) + f(x_3)}{2} + \cdots + (x_n - x_{n-1})\frac{f(x_{n-1}) + f(x_n)}{2}.$$

The function should take as inputs

- a single column vector of abscissas, that is, the x_i's,
- a single column vector of ordinates, that is, the $f(x_i)$'s or y_i's,
- a scalar that is the initial condition on y, that is, $f(x_0)$ or y_0.

The output should be a vector `int_y` that is the integral of $f(x)$ or y at every x. So your function definition should look something like

```
function int_y = trap(x,y,yi)
```

The function should check that the `x` and `y` vectors are the same length and should throw and error if they are not.

The goal is to estimate $\dot{\theta}(t)$, $\dot{\phi}(t)$, $\dot{\psi}$, $\theta(t)$, $\phi(t)$, and $\psi(t)$ by calling your `trap` function with a script that is written for the given data. For example, the x's in the above equations are t's

*Note that this formula works for equally- or unequally-spaced abscissas, but in the data I am giving you the data is equally-spaced. Therefore, $x_i - x_{i-1} = h$ is constant.

in the given data, there are three y's in the given data, one is a vector containing $\ddot{\theta}(t)$, the second is a vector containing $\ddot{\phi}(t)$, and the third is a vector containing $\ddot{\psi}(t)$. Write a script called `trap_run` that does all of the following.

(a) Plots the accelerations as a function of time in a `figure` window.

(b) Integrate the angular accelerations to get estimates for the angular velocities as a function of time, and then plot the angular velocities versus time in a second `figure` window.

(c) Integrate the angular velocities to obtain estimates for angles θ, ϕ, and ψ as functions of time. Plot the angles versus time in a third `figure` window.

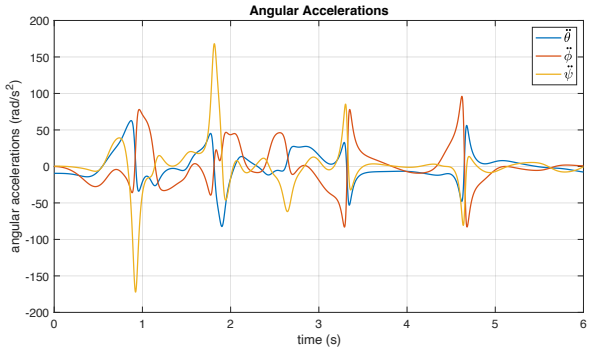
(d) In a fourth `figure` window, plot the trajectories of m_1, m_2, and m_3. Use $l_1 = l_2 = l_3 = 1$ to generate your plot. Also put a marker where each mass starts and where each mass ends its motion. Include those markers in your `legend`.

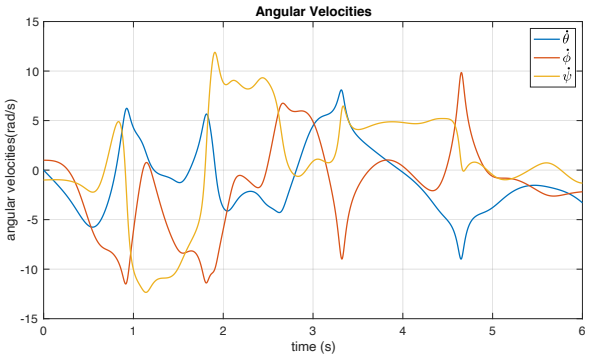
(e) Finally, use MATLAB's `cumtrapz` function to generate the plots of the trajectories of m_1, m_2, and m_3. Include the starting and end points of each mass as in part (d). Comment on the similarities and differences between the two trajectory plots.

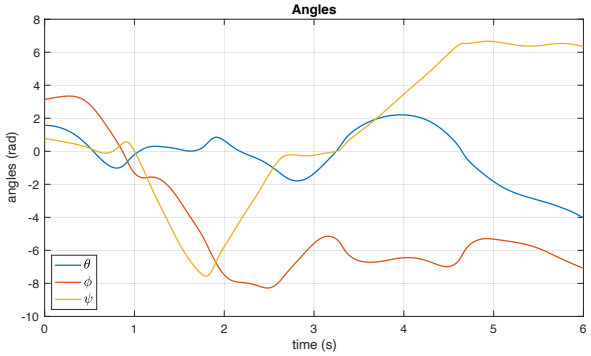
Make sure your all your plots have a `legend`, the axes are all labeled, and each `figure` has a `title`.

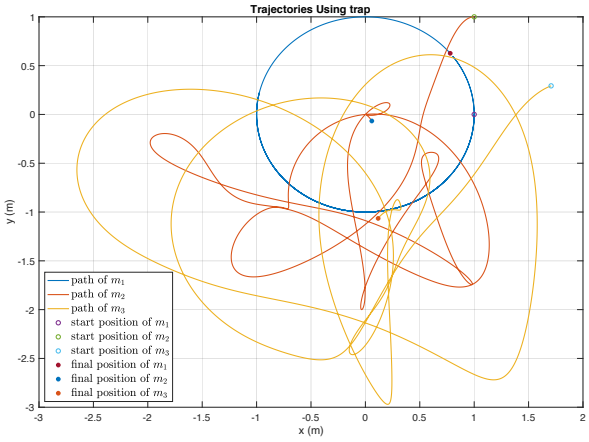
Answers

The plots should look like this.









Problem 2 (34 pts)

We are now going to approximate the integral

$$I = \int_0^{2\pi} x^4 e^{-x} \cos^2 x \sin x \, dx, \tag{1}$$

using composite Simpson's rule, the trapezoidal rule, and MATLAB's built-in function `integral`.

(a) Write a MATLAB function called `csr` that uses composite Simpson's rule to evaluate the integral of an arbitrary function $f(x)$ on an interval $a \le x \le b$. The function should look something like

```
function integ = csr[f,a,b,n]
```

where `f` is $f(x)$ as a function handle, `a` and `b` are the limits of integration, `n` is the number of subintervals used to estimate the integral, and `integ` is the integral approximation. The spacing between nodes is $h = (b-a)/n$, and don't forget that *composite Simpson's rule requires that n be even*, which should be an error check in your function.

To check your function, write a script called `csr_run` that calls `csr` to integrate $\sin x$ from 0 to π using 10 subintervals. It should report the approximate value of the integral. You should get:

```
integral approximation is 2.000109517
```

(b) Now add a section to your `csr_run` script that approximates the integral in Eq. (1). It should do the following

(i) call your `csr` function with with increasing values of n, $n = 2, 4, 6, \ldots$ (that is, decreasing values of h), until the approximate relative error in your estimate is less than 10^{-9},

(ii) use MATLAB's `integral` function with its default tolerances to do the same integral,

(iii) compute the *true* relative error between MATLAB's result and yours.

The script should report the final estimate of the integral, the value of n required to achieve the estimate, the *approximate* relative error of your final result, and the *true* relative error. Here is what I get:

```
number of subintervals required is 246
estimate of the integral is -1.641455791
approximate relative error is 9.61497e-10
true relative error is 2.89647e-08
```

(c) Next, apply your `trap` function to estimate the integral using the same number of intervals that your `csr` function needed;* report the integral approximation and the true relative error. Here is what I get:

*You should not change your `trap` function to do this, though you will now need to generate the information you send to it.

```
trap estimate of the integral with 246 intervals is
   -1.641297507
true relative error is 9.64003e-05
```

(d) Finally, apply your `trap` function with increasing values of n, $n = 1, 2, 3, \ldots$, until it achieves an approximate relative error of 10^{-9}; it should report the number of intervals required, the integral approximation, the final approximate relative error, and the true relative error. Here is what I get:

```
number of subintervals required is 2269
estimate of the integral is -1.641453884
approximate relative error is 9.99379e-10
true relative error is 1.13305e-06
```

Problem 3

(32 pts)

Write a MATLAB function `gauss` that uses Gauss-Legendre quadrature to integrate a function $f(x)$ from $x = a$ to $x = b$. As a reminder, given the abscissas $\{x_1, x_2, \ldots, x_n\}$ and corresponding weights $\{w_1, w_2, \ldots, w_n\}$ for n-point Gauss-Legendre quadrature over $[-1, 1]$, to apply the rule over the interval $[a, b]$ requires the change of variables

$$t = \frac{a+b}{2} + \frac{b-a}{2}x \quad \text{and} \quad dt = \frac{b-a}{2}dx.$$

Then the relationship

$$\int_a^b f(t)\,dt = \int_{-1}^{1} f\left(\frac{a+b}{2} + \frac{b-a}{2}x\right)\frac{b-a}{2}\,dx,$$

is used to obtain the quadrature formula

$$\int_a^b f(t)\,dt = \frac{b-a}{2}\sum_{i=1}^{n} w_i f\left(\frac{a+b}{2} + \frac{b-a}{2}x_i\right).$$

The MATLAB function should take the following inputs

- the function $f(x)$ as a function handle,
- a and b,
- a row vector of Gauss-Legendre abscissas,
- a row vector of corresponding weights,
- the desired approximate relative error.

The function should approximate the integral of f for Gauss-Legendre abscissas and weights starting with $n = 2$ and increasing n until the desired relative error is achieved. The output should be

(a) the final estimate of the integral,

(b) the final approximate relative error,

(c) the final value of n that achieved the desired relative error.

The function definition should look something like this:

```
function [g_quad,rel_err,n] = gauss(f,a,b,A,W,err)
```

I have generated all the Gauss-Legendre abscissas and weights for $n = 2, 3, 4, \ldots, 256$ and have put them in an $32{,}895 \times 2$ array, in ascending order from $n = 2$ in rows 1 and 2 to $n = 256$ in rows 32,640 through 32,895. For example, the abscissas and weights for $n = 5$ are in rows 10–14. The array is available in a MATLAB file named `GL.dat`,* which can be downloaded from Canvas and then read into MATLAB using

```
load GL.dat -ascii
```

MATLAB will read in the data and place it in the variable named `GL`. Before you call your `gauss` function, you will need to put the column containing the abscissas in the variable `A` and the column containing the weights in the variable `W`. Test your function by integrating $\sin x$ from 0 to π with a desired relative error of 1×10^{-9} in the Command Window or using a simple script. You should achieve the results shown below (the abscissas and weights have already been stored in `A` and `W`)

```
>> f = @(x) sin(x);
>> [g_quad,rel_err,n] = gauss(f,0,pi,A,W,1e-9)
g_quad =
   2.000000000001791
rel_err =
     2.622601025234911e-10
n =
     7
```

Next, write a MATLAB script called `gauss_run` that estimates the following integrals to an approximate relative error tolerance of 1×10^{-9}:

(a) the integral in Problem 2,

```
gauss estimate of the integral is -1.64145574369
approximate relative error is 3.133125e-11
value of n to achieve this error is 16
Matlab's estimate of the integral is -1.64145574367
```

(b) $\displaystyle\int_1^{\pi} \left(x^2 + x + 1\right) e^{-x^2/2}\, dx,$

*It was created by saving the array named `GL` using `save GL.dat GL -ascii -double`

```
gauss estimate of the integral is 1.97444303056
approximate relative error is 3.037572e-11
value of n to achieve this error is 9
Matlab's estimate of the integral is 1.97444303056
```

(c) $\displaystyle\int_0^5 \frac{xe^{-x}\sin x}{x^2+9}\,dx,$

```
gauss estimate of the integral is 0.04923039651
approximate relative error is 9.055116e-10
value of n to achieve this error is 12
Matlab's estimate of the integral is 0.04923039650
```

(d) $\displaystyle\int_1^2 x^3 \cos\left(\frac{2\pi}{\ln 2}\ln x\right)\,dx,$

```
estimate of the integral is 0.61118994781
approximate relative error is 7.580775e-11
value of n to achieve this error is 10
Matlab's estimate of the integral is 0.61118994781
```

(e) $\displaystyle\int_{0.001}^{\pi/2-0.001} \ln x \sec x\,dx.$

```
estimate of the integral is 1.41001085297
approximate relative error is 9.196807e-10
value of n to achieve this error is 190
Matlab's estimate of the integral is 1.41001086519
```

Do the same integrals in Mathematica (`Integrate`) or MATLAB (`integral`) to verify that you are getting the correct results. Submit your Mathematica or MATLAB code with your assignment.