Phys234, 2018, Problem set #6: Due Monday March 19, 3pm

Question 1 & 2 : Will be given at the start of the lab

Question 3:

The data used in the Questions 1-2 (that you'll get at the start of the lab) were generated with the following function, which is a variation of the build-in Matlab function `humps`

$$f(x) = \frac{1}{(x+0.9)^2 + 0.7} + \frac{1}{(x-1.1)^2 + 0.2}$$

Write a m-file function called `humpy.m` such that `y=humpy(x)` returns in y the value of this function $f(x)$. Your function should be flexible enough that if a vector x is passed to `humpy`, a vector y is returned.

To test your `humpy` function, write a function `fakedata.m` that takes in a integer number (`ndat`) as an input and returns: 1) vector `xdat` that contains `ndat` equally spaced values of x between -3 and 3, and 2) the vector `ydat` which is the evaluation of `humpy` at each value of `xdat`. In other words the commands:

```
>>ndat=11;
>>[xdat,ydat]=fakedata(ndat)
```

should reproduce the set of `xdat,ydat` values in the table of Question 1. (You should check this!)

Then write a function called `ps6q3.m` that calculates how well your linear (question 1) and cubic-spline (question 2) interpolations match the original function $f(x)$ when `ndat=11`. The fit is determined by

```
fit = norm(y-yinterp)/norm(y);
```

where vector `y` contains the exact values of $f(x)$, computed for 200 equally spaced values of `x` between $[-3, 3]$. The vector `yinterp` contains your interpolation function approximation of $f(x)$ at these same `x` points. The better the fit, the closer to zero `fit` should be. Your function `ps6q3.m` should print the value of `fit` for each interpolation.

Finally, your function `ps6q3.m` should also produce a plot that includes: 1) the 11 data points, 2) your cubic spline interpolation function, 3) the linear interpolation and 4) the exact function $f(x)$.

Question 4:

We now want to repeat what we did in Question 3, but for different choices of the number of "fake" data points `ndat`.

Write a function called `ps6q4.m` that does the following:

1) it loops over values of `ndat` between 5 and 30;
2) for each `ndat` it calculates the fit for both the linear and cubic-spline interpolation (again based on 200 interpolation points)
3) it produces a plot showing how the two fits change as a function of `ndat`

At the end of your `ps6q4.m` file, include as comments the answer to the following questions. What is the number of "fake" data (`ndat`) points required so that the cubic-spline interpolation fit is better than the linear fit? Why are both fits not monotonously decreasing with `ndat`? In other words, why does increasing `ndat` from 14 to 15, for example, results in a worsening of the fit, not an improvement?