EP 501 Homework 4: Least Squares and Interpolation

October 20, 2020

Instructions:

- Complete all listed steps to the problems.
- Submit all source code, which must be either MATLAB or Python, and output (results printed to the screen or plotted) via Canvas.
- Results must be compiled into a single .pdf file which contains descriptions of the calculations that you have done alongside the results.
- Source codes must run and produce the same essential output presented in your documents.
- Discussing the assignment with others is fine, but you must not copy the code of another student *verbatim*; this is considered an academic integrity violation.
- During submission on the canvas website compress all of your files for a given assignment into a single .zip file, e.g. assignment4.zip . I should be able to run your codes by unzipping the files and then opening them and running them in the appropriate developer environment (MATLAB or Spyder).
- You may use, *verbatim* or modified, any of the example codes from one of the course repositories contained on the GitHub organization website `https://github.com/Zettergren-Courses`.
- For demonstrating that your code is correct when you turn in the assignment, you must use the test problems referenced in the assignment, namely test_lsq.mat and test_interp.mat.

Purpose of this assignment:

- Learn principles behind data fitting and polynomial approximation.
- Develop good coding and documentation practices, such that your programs are easily understood by others.
- Hone skills of developing, debugging, and testing your own software
- Learn how to build programs on top of existing codes

1. This problem concerns least squares and data fitting and requires use of the example dataset from the repository, `test_lsq.mat`, which provides data for variables x_i, y_i, σ_{yi} referenced below.

   (a) Write a program that performs a linear least squares fit of a set of data to a polynomial of arbitrary order n. You may use any functions in the course repository or that you have written for your homework for solving the required system of equations; however, you may not use the built-in MATLAB functions for your solution (only to check the results).

   (b) Use your fitting program to fit the test data (y_i located at independent variable positions x_i) to a line and a quadratic form. Plot your results and the data on the same axis so they can be easily compared. Test your results against the built-in Matlab functions `polyfit` and `polyval` (`numpy.polyfit` and `numpy.poly1d` if using Python). Compare the error vectors and residuals for the two fits.

   (c) A rigorous way of deciding between preferred functional forms in fits (from some set of options like linear, quadratic, cubic, quartic, etc.) is to define a *goodness-of-fit* statistic that quantifies how effective a particular form is a describing a given data set. This goodness statistic should balance the need to fit the data (by having more parameters (unknowns) in the fit against the fact that of course one can fit a set of data given enough unknowns (a problem referred to as "overfitting the data"). The simplest and most commonly used goodness of fit statistic is the *reduced Chi-squared statistic* defined by:

$$\chi^2_\nu = \frac{1}{\nu} \sum_i \frac{\left(y_i - f(x_i)\right)^2}{\sigma^2_{y,i}}, \tag{1}$$

   where, as before, x_i are the points of the independent variable at which data are sampled, y_i are the data, $f(\cdot)$ is the function which is being fitted to the data (linear, quadratic, cubic, etc.), σ_i is the uncertainty of the data and ν is the number of degrees of freedom in your fit (equal to the number of data points minus number of parameters being estimated). Write a function that evaluates χ^2_ν for a polynomial fit of order n.

   (d) Use your goodness-of-fit statistic to determine whether is best (in the statistical sense) to fit these data with a linear, quadratic, or cubic polynomial. Show how you reached your decision.

2. This problem concerns bilinear interpolation methods and requires use of the grid (variables `xg,yg`) and data samples (`f2D`) from test_interp.mat.

   (a) Write a function that takes in a grid of points describing some independent variable (say x_i), and a point to which the data are to be interpolated x' and finds the index i into the array x_i such that: $x_i \leq x' \leq x_{i+1}$.

   (b) Use the function from part (a) to construct an additional function that works over a 2D grid x, y. I.e. given two grids x_i, y_j find the indices i, j such that: $x_i \leq x' \leq x_{i+1}$, $y_j \leq y' \leq y_{j+1}$.

   (c) Use your results from parts a and b to create a bilinear interpolation function that takes in a sequence of data points $\{x'_k, y'_k\}$ to which data are being interpolated, a grid x_i, y_j, and a dataset f_{ij} that is defined over this grid and produces bilinearly interpolated values of f_k at the points $\{x'_k, y'_k\}$. Write your program so that the input points are simply a flat list and not necessarily a 2D grid of points (you can always `reshape` the results later if needed).

   (d) Test your results against Matlab's bilinear interpolation function (`interp2`) and show that you get the same result. Use the test data from the repository (test_interp.mat). The source grid data are stored in `xg,yg`, while the value of the function at those points is in `f2D`. `xgi,ygi` are the densely sampled grid point to which the data are to be interpolated for this test.