PRA1004 Scientific Computing - Lab Assignments

Frans Oliehoek
<frans.oliehoek@maastrichtuniversity.nl>

Week 3

Overview Lab 3

In this lab session, you will...

- ... use different *interpolation* methods on a dataset of the population of The Netherlands in the previous century as well as random data, that you generate yourself.
- ... learn how systems of linear equations play a crucial role in problems of *polynomial interpolation*, and how these systems can be solved.
- ... see that making an exact fit is not always the best thing to do. Especially in the presense of sensor noise, it may be preferrable to fitting lines and polynomials using *least-squares regression*.

As always, you are expected to hand in a report where you answer your questions with a short explanation toghether with the source code you produce in this lab session. (see the document with requirements on my website.)

Also, for this lab, you will need to download the lab kit from my website.

1 Computational Cost

1. Create a function called `C = MatMult(A,B)` that implement matrix multiplication using a triple for loop.
   - Make sure that you document the function, such that you can call "help MatMult" and get an informative text.
   - Also, you should do some checking of the input. In particular, use `size` to check whether AB is a valid matrix product. If the dimensions are incompatible throw an error "cannot multiply, sizes incorrect" using `error`.
   - Something that may help you when debugging, is to use the following test matrices:

```
A = reshape(1:6, 2, 3)
B = reshape(10:10:60, 3, 2)
```

(This type of test input will allow you to easily identify what goes wrong if there is a bug.)

2. Create a plot showing the run time of MatMult on random matrices of size 10, 20, ..., 100.

3. Also create a plot showing the runtime of Matlab's multiply operator on random matrices of size 100, 200, ..., 1000.

4. What reasons can you give to explain the difference?

2 Interpolation

A. Population Data

1. Load the data from 'population.txt', which contains the population (in thousands) of The Netherlands for the indicated years. Make a point plot of the data with the correct x and y axis.

2. Use the `interp1` function to perform a linear and spline interpolation. In particular, perform the interpolation for all years between 1900 and 2000. Use the help command to understand the exact syntax of the function.

3. Also perform a 'nearest' interpolation.

4. Plot the interpolations using lines in the same figure (use hold on/off to avoid/enable erasing previous plots, for an example see the script in section 3.3). What interpolation would you use?

5. Navigate to `http://en.wikipedia.org/wiki/Demographics_of_the_Netherlands` to get all of the data. Save it in your own text file that can be imported to Matlab/Octave.

6. Import the data and make a plot of your favorite interpolation alongside the actual numbers. Is it as you had expected?

B. Noisy Measurements

1. Load the data from 'noisy.m', plot the data (use a point plot).

2. What kind of interpolation would you want to use for this data?

3 Polynomial Interpolation

3.1 Random Data

Write a script that performs the following:

1. defines a variable for the number of points N.

2. defines `X=0:(N-1)`.
3. draws a random Y value in the range $[0, 10]$ for each of these N points.
4. plots these randomly drawn points.
5. performs a polynomial fit using `polyfit(X,Y,N-1)`.
6. defines `Xfull` as a denser grid on the x-axis.
7. Evaluates the polynomial fit at all the points in `Xfull` (use the `polyeval` function).
8. plots the resulting fit.

Run you script for a number of different values of N, what do you notice?

3.2 Population Data

Run polynomial interpolation on the population data. (include the plot in your report) What do you notice? (is the fit going through the data points?) What do you think that is the cause of this?

3.3 Solving Linear Systems

In this assignment we take away some of the magic of the polyfit function. In particular, we show how it is easy to find the polynomial interpolation.

- Open the file `linSysPolyFit.m` and inspect it. It should look like this:

```
N = 3;
X = [0, 1, 2]';
Y = [1.62299, 2.33874, 0.40348]';
hold off;
plot(X,Y,'+r', 'markersize', 25);
hold on;
% the data points induce a system of N equations of the form
%    a1*x^2 + a2*x + a3*1 = y
%
% Therefore, we solve the system using D * a = Y, where ...
% D = ... , a = ...
% first, construct C
% then solve the system of equations
a =
%plot it
Xfull = -0.5 :0.1: N-0.5;
Yfull = polyval(a, Xfull);
plot(Xfull, Yfull, '-b');
axis([-0.5, N-0.5, -2, 3]);
print('linSysPolyFit.eps', '-deps2')
%or you prefer a different format (e.g., .png):
%print('linSysPolyFit.png', '-dpng')
```

- You notice that the file defines 3 data point, of which the x and y coordinates are stored in X and Y.

Questions:

1. The comments mention that these points induce a system of $N(= 3)$ equations. Write down these 3 equations.
2. What is the general shape of the matrix `D` mentioned? What is its instantiation here?
3. What is the vector `a`?
4. Finish the implementation. Use `inv` to solve the system of equations. Store the found coefficients in `a`, such that plotting works correctly.

4 General Curve Fitting & Least Squares Solutions

As might be clear by now, it is not always the best choice to use an *interpolent* (i.e., a function that passes exactly through the data points). The reason is that this may give you a very complex function, especially when your measurements have some noise, while to underlying relation in fact is much simpler.

4.1 More Noisy Data

1. Redo the plot of the noisy data of assignment 2. Open up the basic fitting tool. Try some of the different fits.
   NOTE: the basic fitting tool is Matlab only. (However, you can also try and fit polynomials of order 1–4 in Octave.)
   (a) What seems to be the best model?
   (b) What is its sum of squared errors (SSE)?
2. The data was in fact generated from a (fictitious) series of noisy measurements. Run `GenerateNoisyMeasurements` to get a new data set. Plot it together with your previous best fit. Is it still a good fit?
3. Repeat the procedure a number of times, what do you now think is the best model? That is, make a prediction on the value for some x (say, $x = 100$), that I will generate (using the same `GenerateNoisyMeasurements` function) when correcting your report.
4. What (theoretical) assumption of least-squares fitting is violated by the measurements generated by `GenerateNoisyMeasurements`?

Notes: the problem of selecting what fitting function to use is called "model selection" and has a rich literature in statistics.

4.2 Solving a Least-Squares Problem

In this assignment we will use a different data generating function `GenerateNoisyMeasurements2`.

1. Generate and plot N=5 data points using a call to `GenerateNoisyMeasurements2(5)`. (tip: reuse code from `linSysPolyFit.m`).
2. The old code (i.e., your implementation of `linSysPolyFit.m`) tries to fit a 2nd order polynomial. What is the problem you encounter when use this script for the newly generated data set? (How many equations with how many unknowns are there?)
3. Fortunately, there is a solution: 'solving' the system of equations using ' \' (also called left division). Implement this. Plot the resulting fitted polynomial.
4. The matrix C is also called the *design matrix* and it contains the so-called basis functions. What basis functions did you use here?