MATH1019 Linear Algebra and Statistics for Engineers
Laboratory Session 11

Learning outcomes for this session

This lab treats a topic – least squares – from Lecture 12, the notes for which are available on Blackboard. We have already seen the matlab command `A\b` used with square matrices, but the same matlab command is used for over-determined systems, i.e. systems where there are more equations than unknowns. It hardly ever happens that over-determined systems have a solution, and the idea of least squares is to choose $\boldsymbol{x}$ so as to make the 2-norm of $A\boldsymbol{x} - \boldsymbol{b}$ as small as possible. One of the solution methods is to form, and solve, the normal equations

$$A^T A\boldsymbol{x} = A^T\boldsymbol{b}.$$

Because for many of you the lab is before the lecture on this topic, this lab is mostly cut and paste **and think about what is happening**.
At the end of this session, you will be able to

1. Use matlab to find least squares solutions of over-determined sets of linear equations.
2. Appreciate some uses of matlab's Symbolic Toolbox.

Overview

1. Using matlab `x=A\b` for solving least squares
2. Least squares fitting of polynomials to data, matlab `polyfit`

Least squares for a 'pretend' example from surveying

Suppose we have an island with 3 mountains, called A, B and C. The respective heights of mountains A, B and C above sea-level given by

$$x_A \ , \ x_B \ , \ x_C$$

are required. A surveyor measures some differences in heights

$$h_1 \ , \ h_2 \ , \ h_3 \ , \ h_4 \ , \ h_5 \ , \ h_6$$

h_1 is the measurement of the height to top of A from the sea.
h_2 the height to top of B from the sea.
h_3 the height to top of C from the sea.
While at the top of A, the surveyor also measures:
h_4 how much higher B is than A,
h_6 how much higher C is than A.
Similarly when at the top of mountain B, the surveyor measures:
h_5 how much higher C is than B.

The overdetermined system of equations is

$$M\boldsymbol{x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_A \\ x_B \\ x_C \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \end{bmatrix} = \boldsymbol{h}$$

Of course, measurement errors in h_j mean that one can't solve $M\boldsymbol{x} = \boldsymbol{h}$ exactly, but instead one seeks $\boldsymbol{x}$ to minimize $\|M\boldsymbol{x} - \boldsymbol{h}\|_2^2$, i.e. least-squares.
After entering

```
M =sym([1,0,0; 0,1,0; 0,0,1; -1,1,0; 0,-1,1; -1,0,1])
syms h1 h2 h3 h4 h5 h6
hvec = sym([h1;h2;h3;h4;h5;h6])
```

Form the normal equations $M^T M\boldsymbol{x} = M^T\boldsymbol{h}$ with matrix `M'*M` and right-hand side `M'*hvec` (the superscript T denotes transpose, in matlab it is denoted with a prime `M'`) and solve them to find the least squares solution for the heights of the mountains.

Least squares fitting of straight lines to data

1. Suppose the data is $\{(x_i, y_i)\}_{i=1}^n$. We want to fit $y = mx + c$ such that

$$Q(m, c) = \|y - mx - c\|_2^2 = \sum_{i=1}^{n} (y_i - mx_i - c)^2$$

is as small as possible. There are many ways to solve for m and c, e.g.

- geometry – projections
- calculus, where at minimum for Q,

$$\frac{\partial Q}{\partial m} = 0 \quad \text{and} \quad \frac{\partial Q}{\partial c} = 0$$

  (See the optional extra item at the end of this document.)
- algebra – solving $A\boldsymbol{x} = \boldsymbol{y}$ by solving the 'normal equations' $A^T A\boldsymbol{x} = A^T\boldsymbol{y}$.

The problem of fitting a polynomial approximation to data occurs sufficiently often that matlab has a function for it, `polyfit`. Run the following example, which fits a least squares line (i.e. least squares polynomial of degree 1) to a set of data points, to see how to use it. (If you wish, adapt the code to any example you might have from any engineering topic or from the last set of MATH1019 Workshop 11b questions).

```
% https://www.varsitytutors.com/hotmath/hotmath_help/topics/line-of-best-fit

xdata = [ 8, 2, 11, 6, 5, 4, 12, 9, 6, 1]
ydata = [ 3, 10, 3, 6, 8, 12, 1, 4, 9, 14]

linfit= polyfit(xdata,ydata,1)
xrange= [0:0.1:12];
f1= polyval(linfit,xrange);
figure
plot(xdata,ydata,'rd',xrange,f1,'-')
title(' Example of a fit of a straight line ')
```

2. Use `polyfit` to fit a quadratic approximation (i.e. least squares polynomial of degree 2) to the `xdata`, `ydata` entered as above (you may need to consult the help on `polyfit`). Also plot the quadratic approximation.

Extra, optional, items

Least squares

In the MATH1020 Calculus for Engineers unit you will be using Matlab's Symbolic Toolbox. This can be used to derive the formulae for least squares fits of a line:

```
syms f x y k m c d
% d is the slope of the line, m number of points
f=symsum(str2sym('(y(k)-c-d*x(k))^2'),k,1,m)
fC=diff(f,c)
fD=diff(f,d)
solAns = solve([fC,fD],[d,c])
pretty(solAns.d)
pretty(solAns.c)
```

Other topics

Partly to illustrate that matlab live scripts can be used to present engineering mathematics attractively, there are a couple of documents, both involving solving linear equations, which you can look at. One document is on trusses (Trusses_Matlab.pdf), the other on the Wheatstone bridge electric circuit (Wheatstone_Matlab.pdf). Both these pdf files can be found in Blackboard under Laboratory Session 11, as well as the two worked examples Trusses_Worked_Example_Live_Script.mlx and Wheatstone_Worked_Example_Live_Script.mlx.