Linear Algebra

Linear least squares

Monday, March 11th
Math 365
Week #8
Is there a line which fits the data in some optimal sense?

Slope = ?

y-intercept = ?

Linear regression
Linear regression

When are the points and the line “close” in some sense? We could try to minimize the “projected” distance of each point to the line.
We could minimize the vertical distance to the line

**Question**: What is the slope and y-intercept of the line that minimizes this vertical distance?
Linear regression

Questions we might have:

• What is the model we are trying to fit?
• What do we know?
• What don’t we know?
• What do we mean by “distance”
• How to compute the distance,
• How do we describe a “collective distance”
• What kind of minimum we are after,

Finally,

• What algorithm do we use to find our unknowns?
What does our model look like?

Our independent variable is $x$ and the dependent variable is $y$, and the model we are trying to fit is

$$y = mx + b$$

Our goal is to find $m$ and $b$
One approach is to attempt to “solve” the system of equations given by

\[ mx_1 + b = y_1 \]
\[ mx_2 + b = y_2 \]
\[ mx_3 + b = y_3, \]
\[ \vdots \]
\[ mx_N + b = y_N \]

An exact solution does not exist in general.
Over-determined system

The linear system is given by

$$\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \\ \vdots \\ x_N & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} m \\ b \end{bmatrix}$$

Solve the normal equations

$$A^T A \hat{x} = A^T b$$
Normal equations

\[ A^T A \hat{x} = A^T b \]

\[ \hat{x} = (A^T A)^{-1} A^T b \]

to get \( m = \hat{x}_1 \) and \( b = \hat{x}_2 \)

This system is invertible as long as \( A \) has “full column rank”
The vertical distance from a point \((x_i, y_i)\) to the line is given by ...

Vertical distance to the line: 
\[ d_i = |y_i - (mx_i + b)| \]
Minimize “collective” distance

Find the $m$ and the $b$ that minimises all of the distances.

What do we mean by “all” distances? What will make us happy?

- Minimizing the sum of all the distances?
- Minimizing the sum of the distances squared?
- Do we want to selectively weight the importance of some data points over the others?

We might be happy with any choice. So what is easiest mathematically?
Minimize sum of squares

The normal equation solution minimises the sum of the squared distances:

\[ D(m, b) = \frac{1}{2} \sum_{i=1}^{N} (y_i - (mx_i + b))^2 \]

From Calculus III, we know that at a minimum of a multivariable function, we have

\[ \frac{\partial D}{\partial m} = 0 \]

\[ \frac{\partial D}{\partial b} = 0 \]

We use the squared distance rather than absolute value because the absolute value function is not differentiable at its minimum.
Minimize sum of squares

\[
\frac{\partial D}{\partial m} = \sum_{i=1}^{N} (y_i - (mx_i + b))(-x_i) = 0
\]

\[
\frac{\partial D}{\partial b} = \sum_{i=1}^{N} (y_i - (mx_i + b)) = 0
\]

These equations are exactly satisfied at the \( \hat{\mathbf{x}} \), the solution to the normal equations.

\[
\begin{bmatrix}
\sum x_i^2 & \sum x_i \\
\sum x_i & N
\end{bmatrix}
\begin{bmatrix}
m \\
b
\end{bmatrix}
= 
\begin{bmatrix}
\sum x_i y_i \\
\sum y_i
\end{bmatrix}
\]

\[
A^T A \quad \hat{\mathbf{x}} \quad A^T \mathbf{b}
\]
Another way to view what least squares is doing:

\[ \hat{x} = \min_{x \in \mathbb{R}^2} \frac{1}{2} \| Ax - b \|^2 \]

We are minimizing the squared length of the vector \( Ax - b \).

A linear algebra view:

In linear algebra terms, \( b \) is not in general in the column space \( C(A) \) of \( A \). But we can "project" \( b \) onto \( C(A) \) and get the vector \( A\hat{x} \) in \( C(A) \) that most closely resembles \( b \).
In Matlab, we can solve the normal equations very easily using the backslash operator

```matlab
load data; % Load x,y into memory
plot(x,y,'.','markersize',20);
A = [x ones(length(x),1)];
b = y;
xhat = A\b;
m = xhat(1);
b = xhat(2);
plot(x,m*x + b,'r','linewidth',2);
xlabel('x data (m)','fontsize',18,'fontweight','bold');
ylabel('y data (t)','fontsize',18,'fontweight','bold');
title('Distribution of data','fontsize',18,'fontweight','bold');
daspect([1 1 1]);
```

Other methods: `polyfit` or `linsolve`.

These also solve the normal equations as above.