

# Linear Least Squares I

Steve Marschner

Cornell CS 322

# Outline

Linear Fitting

Examples of linear fitting problems

Solving linear least squares problems

Difficulties in least squares fitting

Summary

# Linear systems

We have been looking at systems

$$y_i = f_i(x_1, \dots, x_n) \quad \text{for } i = 1 \dots n$$

or

$$\mathbf{y} = \mathbf{f}(\mathbf{x}) \quad \text{where } \mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^n$$

which, when  $\mathbf{f}$  is linear, read

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{where } \mathbf{A} \in \mathbb{R}^{n \times n}$$

# Square linear systems

The equation  $\mathbf{Ax} = \mathbf{b}$  with  $\mathbf{A}$  an  $n \times n$  matrix is a *square* linear system. Generally we expect this system to have exactly one solution.

$$\boxed{\mathbf{A}} \begin{array}{c} | \\ \mathbf{x} \\ | \end{array} = \begin{array}{c} | \\ \mathbf{b} \\ | \end{array}$$

(If  $\mathbf{A}$  is singular, there might be no solution or many solutions.)

## Non-square systems

If  $\mathbf{A}$  is  $n \times m$ , and  $n \neq m$ , the system is called (surprise!) *non-square* or *rectangular* and is generally either *overdetermined* or *underdetermined*.

$$\begin{array}{c} \boxed{\mathbf{A}} \\ \mathbf{x} \\ \mathbf{b} \end{array} =$$

overdetermined

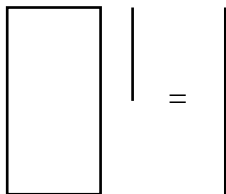
$$\begin{array}{c} \boxed{\mathbf{A}} \\ \mathbf{x} \\ \mathbf{b} \end{array} =$$

underdetermined

(If  $\mathbf{A}$  is singular, you can't necessarily tell whether a system is over- or underdetermined from its shape.)

# Overdetermined systems

Today, we're interested in the overdetermined case:  $m > n$ , more knowns than unknowns.

A diagram representing the equation Ax ≈ b. On the left is a tall, narrow rectangle representing matrix A. To its right is a vertical line representing vector x. To the right of x is an equals sign. To the right of the equals sign is another vertical line representing vector b. The vertical line for b is shorter than the vertical line for x, indicating that the number of equations (m) is greater than the number of unknowns (n).

$$\mathbf{Ax} \approx \mathbf{b}$$

Generally such an equation will have no exact solution, and we are in the business of finding a compromise.

# Linear regression

Experiment to find thermal expansion coefficient with metal bar and a torch:

- measure temperature of bar, record as  $T_1$ .
- measure length of bar, record as  $L_1$ .
- crank up heat, wait for a bit.
- measure temperature  $T_2$  and length  $L_2$ .
- repeat for many trials.

The data is  $n$  pairs  $(T_i, L_i)$ .

The hypothesis is that  $L(T) = L_0(1 + \alpha T)$ , where  $L_0$  is the bar's nominal length, and we want to estimate  $\alpha$ .

# Linear regression

To put this in the standard form, we have a set of given data points  $(x_i, y_i)$  and we believe that  $y = mx + b$ . (Here  $x$  is  $T$ ,  $y$  is  $L$ , and  $m$  is  $L_0\alpha$ .)

We believe that if there were no experimental uncertainty the model would fit the data exactly, but since there is noise the best we can do is minimize error. The problem is

$$\min_{m,b} \sum_i (mx_i + b - y_i)^2$$

To make this look like our standard problem we use the HW2 trick:

$$mx + b = \begin{bmatrix} x & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix}$$



# Linear regression

Stacking the data points into a matrix results in:

$$\min_{m,b} \left\| \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \right\|^2$$

which is a linear least squares problem in the standard form.

## Polynomial regression

Suppose the model we expect to fit our data  $(x_i, y_i)$  is a cubic polynomial rather than a straight line:

$$p(x) = ax^3 + bx^2 + cx + d$$

We want to find  $a$ ,  $b$ ,  $c$ , and  $d$  to best match the data:

$$\min_{a,b,c,d} \sum_i (ax_i^3 + bx_i^2 + cx_i + d - y_i)^2$$

Thinking of the coefficients as variables and the variables as coefficients, we can write this:

$$\min_{a,b,c,d} \left\| \begin{bmatrix} x_1^3 & x_1^2 & x_1 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_n^3 & x_n^2 & x_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \right\|^2$$

# Fitting with basis functions

This same approach works for any set of functions you want to add together to approximate some data:

$$y_i \approx \sum_j a_j b_j(x_i)$$

This works for any  $b_j$ s, such as monomials (which we just saw), sines and cosines, etc.

# Economic prediction

So far we have looked at a single independent variable, with complexity arising from the type of model. Some problems have many independent variables.

Moler's problem 5.11 has an example of an economic application. We would like to be able to predict total employment from a set of other economic measures:

- $x_1$ : GNP implicit price deflator
- $x_2$ : Gross National Product
- $x_3$ : Unemployment
- $x_4$ : Size of armed forces
- $x_5$ : Population
- $x_6$ : Year

## Economic prediction

We'd like to approximate  $y$ , the total employment, as a linear combination of the others:

$$y \approx \beta_0 + \sum_j \beta_j x_j$$

We have historical data available for many years, and so we can set up a system with a row for each year, each of which reads

$$y = \begin{bmatrix} 1 & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_6 \end{bmatrix}$$

with more than 7 years of data, this will be an overdetermined system that can be solved by least squares. Then  $y$  can be predicted in future years for which only the  $x$ s are available.

# Least squares fitting

The basic approach is to look for an  $\mathbf{x}$  that makes  $\mathbf{Ax}$  close to  $\mathbf{b}$ :

$$x^* = \min_x \text{distance}(\mathbf{Ax}, \mathbf{b}).$$

How to measure distance?

Usually by the magnitude of the difference:

$$x^* = \min_x \text{size}(\mathbf{Ax}, \mathbf{b})$$

How we measure “size” determines what kind of answer we get.

# Least squares fitting

The default way to measure size is with a vector norm, such as the familiar Euclidean distance (2-norm):

$$x^* = \min_x \|\mathbf{Ax} - \mathbf{b}\|$$

which expands out to

$$\begin{aligned} x^* &= \min_x \sqrt{\sum_i (\mathbf{a}_i \cdot \mathbf{x} - b_i)^2} \\ &= \min_x \sum_i (\mathbf{a}_i \cdot \mathbf{x} - b_i)^2. \end{aligned}$$

Since we only care about the minimum value, we can drop the square root, and our problem is to minimize the sum of squares.

# Why least squares?

Why are we using this sum-of-squares metric for error?

- Because it is the right norm for the problem?  
maybe with some strong assumptions. . .
- Because it corresponds to a familiar notion of distance?  
getting closer. . .
- Because it results in a problem that's really easy to solve?  
bingo!

Don't let its elegance seduce you into thinking that a least squares solution is the Right Answer for every fitting problem.



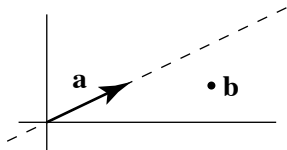
## Solving a $2 \times 1$ least squares system

Let's look at an example for  $n = 1, m = 2$ :

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} x \approx \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad \text{or} \quad \mathbf{a}x \approx \mathbf{b}$$

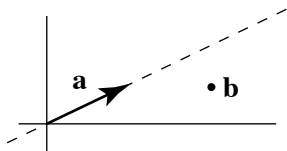
In this case we are taking a scalar multiple of a single vector  $\mathbf{a}$  and trying to come close to a point  $\mathbf{b}$ .

Here is a picture of the situation:



# Solving a $2 \times 1$ least squares system

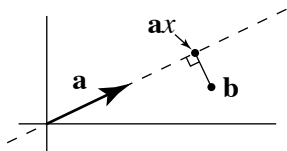
What is the closest point on this line to  $b$ ?



It is the orthogonal projection of  $b$  on to the line.

# Solving a $2 \times 1$ least squares system

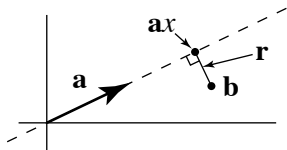
What is the closest point on this line to  $b$ ?



It is the orthogonal projection of  $\mathbf{b}$  on to the line.

# Solving a $2 \times 1$ least squares system

What is the closest point on this line to  $\mathbf{b}$ ?



It is the orthogonal projection of  $\mathbf{b}$  on to the line.

If  $\mathbf{ax}^*$  is the closest point to  $\mathbf{b}$ , then the residual  $\mathbf{r} = \mathbf{ax}^* - \mathbf{b}$  must be orthogonal to  $\mathbf{a}$ :

$$\mathbf{a} \cdot \mathbf{r} = 0 \quad ; \quad \mathbf{a} \cdot (\mathbf{ax}^* - \mathbf{b}) = 0 \quad ; \quad \mathbf{a} \cdot \mathbf{ax}^* = \mathbf{a} \cdot \mathbf{b}$$

## Solving a $2 \times 1$ least squares system

So the  $2 \times 1$  case boils down to

$$\mathbf{a} \cdot \mathbf{a}x^* = \mathbf{a} \cdot \mathbf{b}$$

Some interpretations of this:

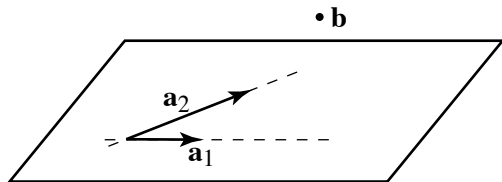
- Residual is orthogonal to  $\mathbf{a}$ .
- The vectors  $\mathbf{a}x^*$  and  $\mathbf{b}$  have the same component in the  $\mathbf{a}$  direction.
- (If  $\|\mathbf{a}\| = 1$ )  $x^*$  is the component of  $\mathbf{b}$  in the  $\mathbf{a}$  direction.

## Solving a $3 \times 2$ least squares system

Now we can graduate to the  $3 \times 2$  case:

$$\mathbf{Ax} \approx \mathbf{b} \quad \text{or} \quad \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \end{bmatrix} \mathbf{x} \approx \mathbf{b} \quad \text{or} \quad \mathbf{a}_1 x_1 + \mathbf{a}_2 x_2 \approx \mathbf{b}$$

Geometrically, this is finding the point on the plane *spanned* by  $\mathbf{a}_1$  and  $\mathbf{a}_2$  that is closest to  $\mathbf{b}$ .

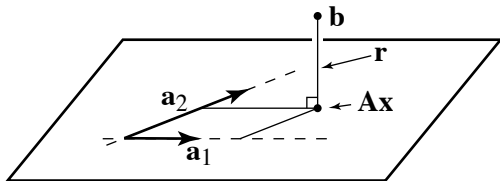


## Solving a $3 \times 2$ least squares system

Now we can graduate to the  $3 \times 2$  case:

$$\mathbf{Ax} \approx \mathbf{b} \quad \text{or} \quad [\mathbf{a}_1 \quad \mathbf{a}_2] \mathbf{x} \approx \mathbf{b} \quad \text{or} \quad \mathbf{a}_1 x_1 + \mathbf{a}_2 x_2 \approx \mathbf{b}$$

Geometrically, this is finding the point on the plane *spanned* by  $\mathbf{a}_1$  and  $\mathbf{a}_2$  that is closest to  $\mathbf{b}$ .



Now the residual is orthogonal to the plane—which is to say, it is orthogonal to both columns of  $\mathbf{A}$ .

## Solving a $3 \times 2$ least squares system

If  $\mathbf{A} = [\mathbf{a}_1 \quad \mathbf{a}_2]$  then being orthogonal to the columns of  $\mathbf{A}$  is two statements:

$$\begin{array}{l} \mathbf{a}_1 \cdot \mathbf{r} = 0 \\ \mathbf{a}_2 \cdot \mathbf{r} = 0 \end{array} \quad \text{or} \quad \begin{array}{l} \mathbf{a}_1^T \mathbf{r} = 0 \\ \mathbf{a}_2^T \mathbf{r} = 0 \end{array}$$

Another way to say this is

$$\mathbf{A}^T \mathbf{r} = 0$$

and if we expand out  $\mathbf{r}$  we get

$$\mathbf{A}^T (\mathbf{A}\mathbf{x} - \mathbf{b}) = 0 \quad ; \quad \mathbf{A}^T \mathbf{A}\mathbf{x} = \mathbf{A}^T \mathbf{b}$$

This statement is known as the *normal equations* of the least-squares system.



# Normal equations

For any  $n$  and  $m$ , the residual is orthogonal to all  $n$  the columns of  $\mathbf{A}$ . So the normal equations

$$\mathbf{A}^T \mathbf{A} \mathbf{x} = \mathbf{A}^T \mathbf{b}$$

hold for any overdetermined system.

This equation is an  $n \times n$  system that you can solve using, e.g., the LU decomposition.

Note: we threw out the long dimension!

→ fitting a plane to a million 3D points is still a  $3 \times 3$  system.

**Caution:** This method is only for “easy” problems! (And hard fitting problems come along more often than you might think.) More on this later.

## Normal equations, alternate derivation

Here is another derivation some find easier to remember:

$$\mathbf{x}^* = \min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|^2 = \min_{\mathbf{x}} (\mathbf{Ax} - \mathbf{b})^T (\mathbf{Ax} - \mathbf{b})$$

At the minimum the derivative of  $\|\mathbf{Ax} - \mathbf{b}\|^2$  with respect to  $\mathbf{x}$  has to be zero:

$$\begin{aligned} 0 &= \nabla \|\mathbf{Ax} - \mathbf{b}\|^2 = \nabla [(\mathbf{Ax} - \mathbf{b})^T (\mathbf{Ax} - \mathbf{b})] \\ &= \nabla [\mathbf{x}^T \mathbf{A}^T \mathbf{Ax} - 2\mathbf{x}^T \mathbf{A}^T \mathbf{b} + \mathbf{b}^T \mathbf{b}] \end{aligned}$$

A little sleight of hand is involved in differentiating these matrix expressions as if they were scalars (but write it out to verify):

$$0 = 2\mathbf{A}^T \mathbf{Ax} - 2\mathbf{A}^T \mathbf{b}$$

# LLS in Matlab

Solving a LLS system in Matlab is simple:

$$x = A \setminus b$$

We've seen that the backslash operator solves square systems. For an underdetermined system it computes the least squares solution (using a method better than the normal equations).

# Multi-RHS least squares systems

As with square systems we can have a system

$$\mathbf{AX} \approx \mathbf{B}$$

which amounts to a set of separate problems sharing the same matrix.

# Difficulties in least squares fitting

Fitting using least squares is elegant and efficient. But there are some pitfalls:

- Sensitivity to outliers
  - If uncertainties are truly Gaussian, least squares is optimal
  - One bad point will completely break the result
- Choosing the right size parameter space
  - too few parameters: model is underparameterized (does not fit the data)
  - too many parameters: overfitting (model distorts to accommodate minor variations and noise)

# Summary

- Overdetermined systems have many applications, including many fitting problems.
- When the problems are linear there is a very clean and simple way to find the optimum, if we adopt the sum-of-squares error metric.
- The normal equations convert an overdetermined system into a square system  
But only for easy problems! More stable methods later. . .