Linear combinations

Abstract goal Given vectors $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n \in \mathbb{R}^m$ and a 'target vector' $\mathbf{b} \in \mathbb{R}^m$, we look for coefficients $x_1, x_2, \ldots, x_n$ such that

$$\mathbf{a}_1 x_1 + \cdots + \mathbf{a}_n x_n = \mathbf{b}.$$

Example

You want to build a balanced meal containing 40 grams of sugar, 30 grams of protein and 20 grams of fats. You have available ingredient 1 which contains 10 grams of sugar, 20 of protein and 3 of fats, and ingredient 2 which contains 5 grams of sugar, 1 of protein and 1 of fats.

$$\begin{array}{l} \text{sugars} \to \\ \text{prot} \to \\ \text{fats} \to \end{array} \begin{bmatrix}10\\20\\3\end{bmatrix} x_1 + \begin{bmatrix}5\\1\\1\end{bmatrix} x_2 = \begin{bmatrix}40\\30\\20\end{bmatrix}.$$

Solvability

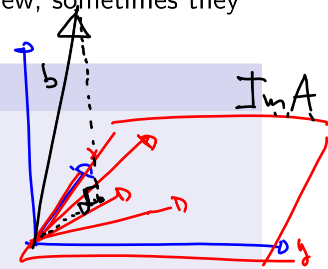
$$Ax = b, \quad A = \overbrace{\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_n. \end{bmatrix}}^{\text{invertible}}$$

Solvable (from linear algebra) when we have $n = m$ linearly independent vectors.

Not always the case: sometimes they are too few, sometimes they are not linearly independent.

Example

$$\begin{bmatrix} 4 \\ 3 \\ 0 \end{bmatrix} x_3 + \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix} x_1 + \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix} x_2 = \begin{bmatrix} 5 \\ 5 \\ 1 \end{bmatrix}$$



is not solvable. [geometric interpretation: spanning plane]

Not even if I add $\begin{bmatrix} 4 \\ 3 \\ 0 \end{bmatrix}$.

Linear least squares problems

Even if I cannot get $\begin{bmatrix} 5 \\ 5 \\ 1 \end{bmatrix}$, maybe I can get $\begin{bmatrix} 5 \\ 5 \\ 0 \end{bmatrix}$...

Problem

What is the closest I can get?

$$A = \left[a_1 \,\middle|\, a_2 \,\middle|\, \cdots \,\middle|\, a_n \right] \in \mathbb{R}^{m \times n}$$

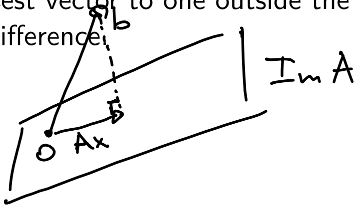
$$\boxed{\min_{x \in \mathbb{R}^n} \|Ax - b\|.}$$

$Ax =$ lin. comb.

(Remember: always $\|\cdot\|_2$ for us.)

Always solvable (will see later) — it is a convex problem.

Geometric interpretation: closest vector to one outside the spanning plane. Orthogonal difference.



Matlab divisions

We will see various algorithms — first, some examples where we let Matlab do the work.

A step back: two division operators in Matlab

```
>> 5 / 2
ans =
   2.5000e+00
>> 5 \ 2
ans =
   4.0000e-01
```

$= \frac{5}{2}$

$\frac{2}{5}$

Mnemonic: divide the 'number above' by the 'number below' the bar.

$$5/2 = \frac{5}{2} \qquad 5\backslash 2 = \frac{2}{5}$$

Linear systems in Matlab

The same operators solve linear systems:

```
>> [1 2; 3 4] \ [5; 6]
ans =
  -4.0000e+00
   4.5000e+00
```

$A \backslash b \leftrightarrow A^{-1} \cdot b$

$w / A \leftrightarrow w \cdot A^{-1}$

Finds the vector x such that

$$Ax = b, \quad A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad b = \begin{bmatrix} 5 \\ 6 \end{bmatrix}.$$

$x = A^{-1} b$

$= \cancel{b A^{-1}}$

Functionally equivalent to $A^{-1}b$ (but not implemented as `inv(A)*b` — there are faster and more stable ways).

There is also `w / A`, which does wA^{-1}, when it makes sense, e.g., when $w = v^T$ is a row vector.

Linear least squares problems

The same operators solve least squares problems.

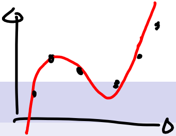
```
>> [2 1; 1 3; 0 0] \ [5; 5; 1]
ans =
   2.0000e+00
   1.0000e+00
```

$$\underbrace{\min_{x=\left[\begin{smallmatrix}x_1\\x_2\end{smallmatrix}\right]\in\mathbb{R}^2} \left\| \begin{bmatrix}2\\1\\0\end{bmatrix} x_1 + \begin{bmatrix}1\\3\\0\end{bmatrix} x_2 - \begin{bmatrix}5\\5\\1\end{bmatrix} \right\|}.$$

$$\begin{bmatrix}2 & 1\\1 & 3\\0 & 0\end{bmatrix} \backslash \begin{bmatrix}5\\5\\1\end{bmatrix}$$

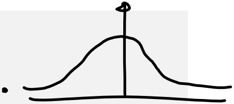
$$\underset{3\times 2}{A} \quad \underset{2\times 1}{x} \quad \approx \quad \underset{3\times 1}{b}$$

Example 1: polynomial fitting



Problem

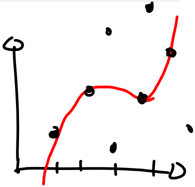
Given pairs (x_i, y_i) such that y_i is 'almost equal to' $ax_i^3 + bx_i^2 + cx_i + d$, find a, b, c, d.



```
% 1000 random points in [-10, 10], sorted
>> x = sort(20*rand(1000,1) - 10);
% degree-3 polynomial plus random noise
>> y = 0.02*x.^3 - x + 1 + randn(1000,1);
>> plot(x, y)
```

$0 \cdot x^2$

```
for i=1:1000
    y(i) = 0.02*x(i)^3 - x(i)+1
end
```

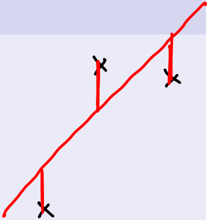


Least squares fitting

Problem

Given pairs (x_i, y_i), find a, b, c, d that minimize

$$\sum_{i=1}^{m}(\boxed{a}x_i^3 + \boxed{b}x_i^2 + \boxed{c}x_i + \boxed{d} - y_i)^2.$$



$a \log(x_i)$

"But it's not a linear problem": actually it is:

$$\min_{a,b,c,d} \left\| \begin{bmatrix} x_1^3 & x_1^2 & x_1 & 1 \\ x_2^3 & x_2^2 & x_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_m^3 & x_m^2 & x_m & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \right\| =$$

$$= \sqrt{\sum_i (a x_i^3 + b x_i^2 + c x_i + d - y_i)^2}$$

Matlab solution

```
>> A = [x.^3 x.^2 x ones(size(x))];
>> p = A \ y
p =
   1.9842e-02
  -5.9348e-04
  -9.9320e-01
   1.0230e+00
```

$$\begin{bmatrix} x_1^3 & x_1^2 & x_1 & 1 \\ & \vdots & & \\ x_{1000}^3 & x_{1000}^2 & \cdots & \end{bmatrix} \backslash\, y$$

Not quite bad.

```
>> plot(x,y,x,A*p)
```

More difficult

about 10 signal about 100 noise

Now with 100× as much noise...

```
>> y = 0.02*x.^3 - x + 1 + 100 * randn(1000,1);
>> p = A \ y
p =
   1.5762e-03
   5.1916e-02
   4.7983e-01
  -7.1315e+00
>> plot(x,y,x,A*p)
```

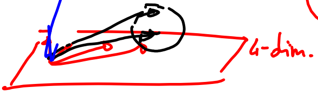
mean 0, variance 1

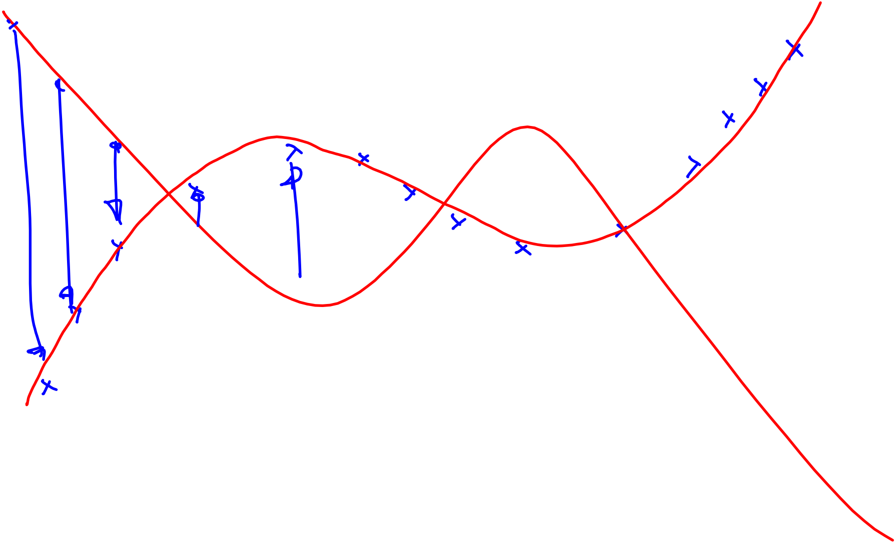
mean 0, variance 100^2

General idea: signal-to-noise ratio is related to the accuracy we can get.

Rough geometric idea: solution $=$ projection on a plane; vectors are almost orthogonal to this plane.

$\mathbb{R}^{1000}$





The statistics behind it

Statistical problem: given observations y_i, what are the values of a, b, c, d that 'most likely' produce it?

$e_i = y_i - ax_i^3 - bx_i^2 - \dots$

If noise $=$ random Gaussian with same variance, 'most likely' (maximum likelihood) means minimizing

Most likely errors: those who minimize $\sum e_i^2$

$$\sum_{i=1}^{m} (\underline{ax_i^3 + bx_i^2 + cx_i + d - y_i})^2,$$

i.e., Euclidean square norm.

If the variances are different, e.g.,

```
>> y(1) = 0.02*x(1)^3 - x(1) + 1 + randn();
>> y(2) = 0.02*x(2)^3 - x(2) + 1 + 5*randn();
```

we'd better rescale rows. (Ask a statistician)

$$\left\| \begin{bmatrix} \frac{1}{100} X_1^3 & X_1^2 & X_1 & 1 \\ \vdots & & & \\ \frac{1}{100} X_{500}^3 & X_{500}^2 & X_{500} & 1 \\ \hline X_{501}^3 & X_{501}^2 & X_{500} & 1 \\ \vdots & & & \\ X_{1000}^3 & X_{1000}^2 & X_{1000} & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} - \begin{bmatrix} \frac{1}{100} y_1 \\ \vdots \\ \frac{1}{100} y_{500} \\ \hline y_{501} \\ \vdots \\ y_{1000} \end{bmatrix} \right\|^2 =$$

$$= \sum_{1} \frac{1}{Var_i} \left(a x_i^3 + b x_i^2 + c x_i + d - y_i \right)^2$$

$$Var_i = \begin{cases} 100^2 \\ 1 \end{cases}$$

Salary estimation

salaries.csv: contains number of points made, rebounds taken, fouls committed by 399 NBA players in season 2015–2016, and the salaries guaranteed by their contract in the next season.
(Source: basketball-reference.com)

Is it true that the best-performing players are paid more? Which of these statistics has a larger impact?

Linear model: $(\text{salary}) \approx (\text{rebounds})\boxed{x_1} + (\text{fouls})\boxed{x_2} + (\text{points})\boxed{x_3}$. + error

$$\sum_{p \in \text{players}} \left(\boxed{x_1}(\text{rebounds})_p + \boxed{x_2}(\text{fouls})_p + \boxed{x_3}(\text{points})_p - (\text{salary})_p \right)^2$$

x_1 and x_3 should be positive and x_2 negative, intuitively. (?)

Matlab example

```
% separator: ',', skip 1 row, 1 column.
>> M = dlmread('salaries.csv', ',', 1, 1)
>> A = M(:, 1:3);
>> b = M(:, 4);
>> x = A \ b
ans =
   1.3285e+04
  -2.6578e+04
   9.5162e+03
```

```
>> [value, position] = min(A*x-b)
value =
  -1.8864e+07
position =
   271
```

Player #271 is paid 18M$ more than he would deserve...