Linear combinations

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

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

Example

A certain food is a mixture of ingredient A, which contains 10 grams of sugars, 20 of protein and 3 of fats, and ingredient B, which contains 5 grams of sugars, 1 of protein and 1 of fats. A lab analysis reveals that the mixture contains 40 grams of sugars, 30 grams of protein and 20 grams of fats. What is the amount of each ingredient?

$$\begin{bmatrix} 10 \\ 20 \\ 3 \end{bmatrix} x_1 + \begin{bmatrix} 5 \\ 1 \\ 1 \end{bmatrix} x_2 = \begin{bmatrix} 40 \\ 30 \\ 20 \end{bmatrix}.$$

Solvability

$$A\mathbf{x} = \mathbf{y}, \quad A = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_n. \end{bmatrix}$$

Solvable for each $\mathbf{y}$ when we have $n = m$ linearly independent vectors (invertibility, from linear algebra).

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

Example

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

is not solvable. [geometric interpretation: spanning plane]

Not even if I add $\begin{bmatrix} 4 \\ 3 \\ 0 \end{bmatrix}$, $\begin{bmatrix} 12 \\ -8 \\ 0 \end{bmatrix}$, ...

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?

$$\min_{\mathbf{x} \in \mathbb{R}^n} \|A\mathbf{x} - \mathbf{y}\|.$$

(Here, $\|v\|_2 = v_1^2 + v_2^2 + \cdots + v_n^2$.)

Geometric interpretation: closest vector to $\mathbf{y}$ inside the hyperplane $\operatorname{Im}(A)$. We obtain it by orthogonal projection.

The obstructions are not always as visible as in our first example; for instance, all columns of A may have zero sum, instead of a zero component...

Matlab divisions

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

First of all: Matlab has two division operators

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

Mnemonic: one divides the number above the bar by the number below.

Linear systems in Matlab

The same operators solve linear systems:

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

Finds the vector $\mathbf{x}$ such that

$$A\mathbf{x} = \mathbf{y}, \quad A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} 5 \\ 6 \end{bmatrix}.$$

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

There is also `X / A`, which computes XA^{-1}, when the product makes sense, e.g., when $X = \mathbf{v}^T$ 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
```

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

Before speaking about algorithms, we will show a few applications of this problem.

Example 0: Linear regression in machine learning (prof. Micheli). Apart from notation change,

$$\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{y}\|^2 \iff \min_{\mathbf{w}} \|X\mathbf{w} - \mathbf{y}\|^2.$$

Example 1: Salary estimation

`salaries.csv`: contains number of points made, rebounds taken, fouls committed by 399 NBA players in season 2015–2016, and the salaries they earn.
(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: $(\text{salary}) \approx (\text{rebounds})x_1 + (\text{fouls})x_2 + (\text{points})x_3$.

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

Our intuition suggests that x_1 and x_3 should be positive, and x_2 may be negative.

Matlab example

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

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

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

Example 2: polynomial fitting

Problem

Given pairs (x_i, y_i) such that y_i is almost equal to $ax_i^3 + bx_i^2 + cx_i + d$, recover the unknown coefficients 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)
```

Least squares fitting

Problem

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

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

It does not look like a linear problem, but it is: the x_i are parameters and a, b, c, d are our unknowns:

$$\min_{\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \in \mathbb{R}^4} \left\| \begin{bmatrix} x_1^3 & x_1^2 & x_1 & 1 \\ x_2^3 & x_2^2 & x_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_m^3 & x_m^2 & x_m & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \right\|$$

(Last column of ones: bias, in machine learning terms.)

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
```

This is not too different from the values we started with; and actually these numbers give a lower error than the ones we used to construct the example, $\begin{bmatrix} 0.02 & 0 & -1 & 1 \end{bmatrix}$.

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

More difficult

Now with $100\times$ 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)
```

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

Geometric idea: if there is no noise, $\mathbf{y}$ lies on the plane $\operatorname{Im} A$; noise moves it away from the plane.

The statistics behind it

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

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

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

i.e., the squared Euclidean norm.

Remark This works because the variance of the added noise is the same on each entry. If they 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 should rescale rows to have more accuracy. (Ask a statistician for more detail.)