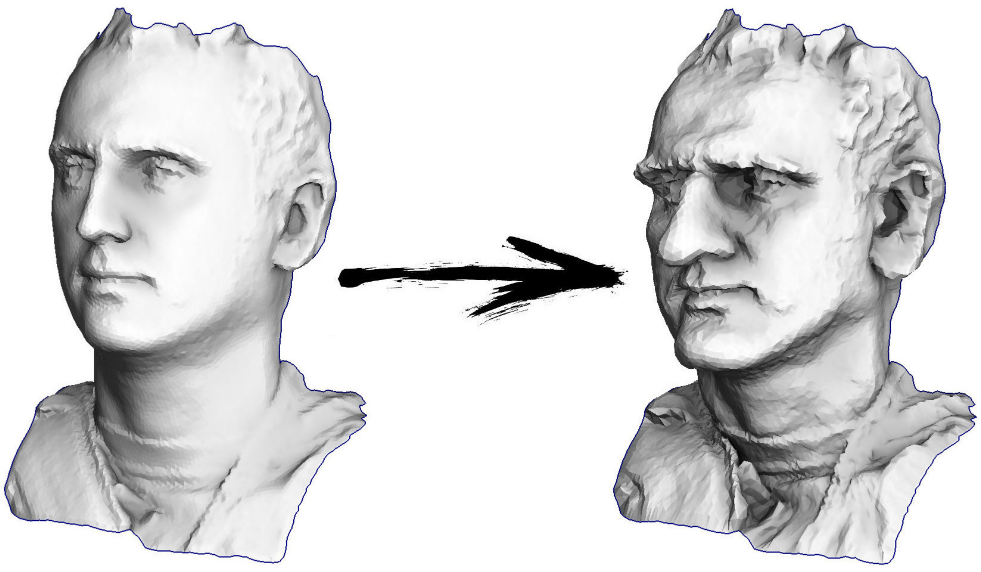
Least squares for programmers

with illustrations

Dmitry Sokolov and Nicolas Ray



Chapter 1

Is probability theory well-founded or do you believe in the theory of evolution?

This short section is not mandatory for the understanding of the main course; the idea behind is to warm up before attacking code and formulae. I'll start with the least squares methods through the maximum likelihood estimation; this requires some (at least superficial) knowledge of probability theory. So, right from the beginning, I would like to digress a little.

I was once asked if I believed in evolutionary theory. Take a short break, think about how you would answer. Being puzzled by the question, I have answered that I find it plausible. Scientific theory has little to do with faith. In short, a theory only builds a model of the world, and there is no need to believe in it. Moreover, the Popperian criterion[] requires a scientific theory be able to be falsifiable. A solid theory must possess, first of all, the power of prediction. For example, if you genetically modify crops in such a way that they produce pesticides themselves, it is only logical that pesticide-resistant insects would appear. However, it is much less obvious that this process can be slowed down by growing regular plants side by side with genetically modified plants. Based on evolutionary theory, the corresponding modelling has made this prediction[], and it seems to have been validated[].

Wait, what is the connection? As I mentioned earlier, the idea is to approach the least squares through the principle of maximum likelihood. Let us illustrate by example. Suppose we are interested in penguins body height, but we are only able to measure a few of these magestic birds. It is reasonable to introduce the body height distribution model into the task; most often it is supposed to be normal. A normal distribution is characterized by two parameters: the average value and the standard deviation. For each fixed value of parameters, we can calculate the probability that the measurements we made would be generated. Then, by varying the parameters, we will find those that maximize the probability.

Thus, to work with maximum likelihood we need to operate in the notions of probability theory. We will informally define the concept of probability and plausibility, but I would like to focus on another aspect first. I find it surprisingly rare to see people paying attention to the word *theory* in "probability theory".

What are the origins, values and scope of probabilistic estimates? For example, Bruno de Finetti said that the probability is nothing but a subjective analysis of the probability that something will happen, and that this probability does not exist out of mind. It's a person's willingness to bet on something to happen. This opinion is directly opposed to the view of people adhering to the classical/frequentist interpretation of probabilty. They assume that the same event can be repeated many times, and the "probability" of a particular result is associated with the frequency of a particular outcome during repeated well-defined random experiment trials. In addition to subjectivists and frequentists, there are also objectivists who argue that probabilities are real aspects of the universe, and not a mere measurement of the observer's degree of confidence.

In any case, all three scientific schools in practice use the same apparatus based on Kolmogorov's axioms. Let us provide an indirect argument, from a subjectivistic point of view, in favor of the probability theory based on Kolmogorov's axioms. We will list the axioms later, first assume that we have a bookmaker who takes bets on the next World Cup. Let us have two events: a = Uruguay will be the champion, b = Germany wins the cup. The bookmaker estimates the chances of the Uruguayan team to win at 40%, and the chances of the German team at 30%. Clearly, both Germany and Uruguay cannot win at the same time, so the chance of $a \wedge b$ is zero. At the same time, the bookmaker thinks that the probability that either Uruguay or Germany (and not Argentina or Australia) will win is 80%. Let's write it down in the following form:

$$P(a) = .4 \qquad P(a \wedge b) = 0 \qquad P(b) = .3 \qquad P(a \vee b) = .8$$

If the bookmaker asserts that his degree of confidence in the event a is equal to 0.4, i.e., $P(a) = 0.4$, then the player can choose whether he will bet on or against the statement a, placing amounts that are compatible with the degree of confidence of the bookmaker. It means that the player can make a bet on the event a, placing \$4 against \$6 of the bookmaker's money. Or the player can bet \$6 on the event $\neg a$ against \$4 of bookmaker's money.

If the bookmaker's confidence level does not accurately reflect the state of the world, we can expect that in the long run he will lose money to players whose beliefs are more accurate. However, it is very curious that in this particular example, the player has a winning strategy: he can make the bookmaker lose money for *any* outcome. Let us illustrate it:

Player's bets		Result for the bookmaker			
Bet event	Bet amount	$a \wedge b$	$a \wedge \neg b$	$\neg a \wedge b$	$\neg a \wedge \neg b$
a	4-6	-6	-6	4	4
b	3-7	-7	3	-7	3
$\neg(a \vee b)$	2-8	2	2	2	-8
		-11	-1	-1	-1

The player makes three bets, and independently of the outcome, he always wins. Please note that in this case we do not even take into account whether Uruguay or Germany were favorits or outsiders, the loss of the bookmaker is guaranteed! This unfortunate (for the bookmaker) situation happened because he did not respect the third axiom of Kolmogorov, let us list all three of them:

- $0 \leq P(a) \leq 1$: all probabilities range from 0 to 1.
- $P(true) = 1$, $P(false) = 0$: true statements have probability of 1 and false probability of 0.
- $P(a \vee b) = P(a) + P(b) - P(a \wedge b)$: this one is also very intuitive. All cases where the statement a is true, together with those where b is true, cover all those cases where the statement $a \vee b$ is true; however the intersection $a \wedge b$ is counted twice in the sum, therefore it is necessary to subtract $P(a \wedge b)$.

Let us define the word "event" as "a subset of the unit square". Define the word "probability of event" as "area of the corresponding subset". Roughly speaking, we have a large dartboard, and we close our eyes and shoot at it. The chances that the dart hits a given region of the dartboard are directly proportional to the area of the region. A true event in this case is the entire square, and false events are those of zero measure, for example, any given point. Figure 1.1 illustrates the axioms.

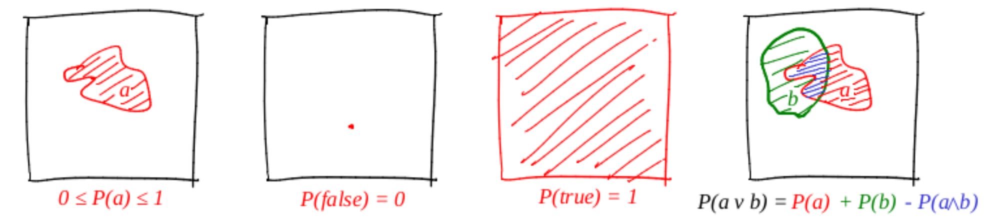


Figure 1.1: A graphical illustration for the Kolmogorov's axioms

In 1931, de Finetti proved a very strong proposition:

If a bookmaker is guided by beliefs which break the axioms of the theory of probability, then there exists such a combination of bets of the player which guarantees the loss for the bookmaker (a prize for the player) at each bet.

Probability axioms can be considered as the limiting set of probabilistic beliefs that some agent can adhere to. Note that if a bookmaker respects Kolmogorov's axioms, it does not imply that he will win (leaving aside the fees), however, if he does not respect the axioms, he is guaranteed to lose. Other arguments have been put forward in favour of the probability theory; but it is the practical success of probability-based reasoning systems that has proved to be very attractive.

To conclude the digression, it seems reasonable to base our reasoning on the probability theory. Now let us proceed to maximum likelihood estimation, thus motivating the least squares.

Chapter 2

Maximum likelihood through examples

2.1 First example: coin toss

Let us consider a simple example of coin flipping, also known as the Bernoulli's scheme. We conduct n experiments, two events can happen in each one ("success" or "failure"): one happens with probability p, the other one with probability $1-p$. Our goal is to find the probability of getting exactly k successes in these n experiments. This probability is given by the Bernoulli's formula:

$$P(k;n,p) = C_n^k p^k (1-p)^{n-k}$$

Let us take an ordinary coin ($p = 1/2$), flip it ten times ($n = 10$), and count how many times we get the tails:

$$P(k) = C_{10}^k \frac{1}{2^k}\left(1-\frac{1}{2}\right)^{10-k} = \frac{C_{10}^k}{2^{10}}$$

Figure 2.1, left shows what a probability density graph looks like.

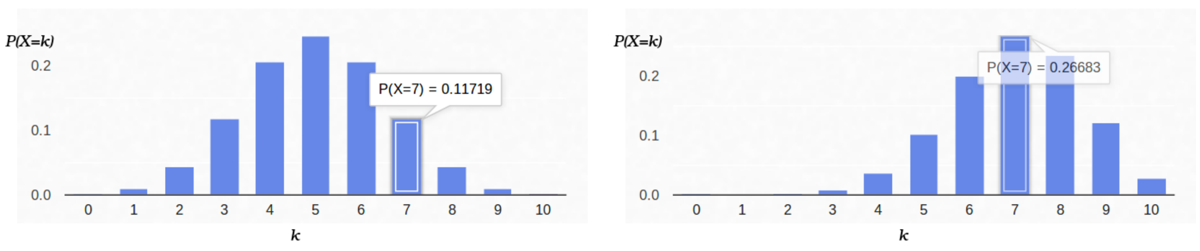


Figure 2.1: **Left:** probability density graph for the Bernoulli's scheme with $p = 1/2$. **Right:** probability density graph for the Bernoulli's scheme with $p = 7/10$.

Thus, if we have fixed the probability of "success" (1/2) and also fixed the number of experiments (10), then the possible number of "successes" can be any integer between 0 and 10, but these outcomes are not equiprobable. It is clear that five "successes" are much more likely to happen than none. For example, the probability encountering seven tails is about 12%.

Now let us look at the same problem from a different angle. Suppose we have a real coin, but we do not know its distribution of a priori probability of "success"/"failure". However, we can toss it ten times and count the number of "successes". For example, we have counted seven tails. Would it help us to evaluate p?

We can try to fix $n = 10$ and $k = 7$ in the Bernoulli's formula, leaving p as a free parameter:

$$\mathcal{L}(p) = C_{10}^7 p^7 (1-p)^3$$

Then the Bernoulli's formula can be interpreted as the plausibility of the parameter being evaluated (in this case p). I have even changed the function notation, now it is denoted as $\mathcal{L}$ (likelihood). That is being said, the likelihood is the probability to generate the observation data (7 tails out of 10 experiments) for the given value of the parameter(s). For example, the likelihood of a balanced coin ($p = 1/2$) with seven tails

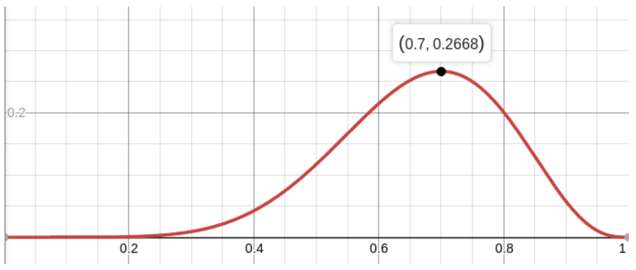


Figure 2.2: The plot of the likelihood function $\mathcal{L}(p)$ for the observation data with 7 tails out of 10 experiments.

out of ten tosses is approximately 12%. Figure 2.2 plots the likelihood function for the observation data with 7 tails out of 10 experiments.

So, we are looking for the parameter value that maximizes the likelihood of producing the observations we have. In our particular case, we have a function of one variable, and we are looking for its maximum. In order to make things easier, I will not search for the maximum of $\mathcal{L}$, but for the maximum of $\log \mathcal{L}$. The logarithm is a strictly monotonous function, so maximizing both is equivalent. The logarithm has a nice property of breaking down products into sums that are much more convenient to differentiate. So, we are looking for the maximum of this function:

$$\log \mathcal{L}(p) = \log C_{10}^7 + 7 \log p + 3 \log(1-p)$$

That's why we equate it's derivative to zero:

$$\frac{d \log \mathcal{L}}{dp} = 0$$

The derivative of $\log x = \frac{1}{x}$, therefore:

$$\frac{d \log \mathcal{L}}{dp} = \frac{7}{p} - \frac{3}{1-p} = 0$$

That is, the maximum likelihood (about 27%) is reached at the point $p = 7/10$. Just in case, let us check the second derivative:

$$\frac{d^2 \log \mathcal{L}}{dp^2} = -\frac{7}{p^2} - \frac{3}{(1-p)^2}$$

In the point $p = 7/10$ it is negative, therefore this point is indeed a maximum of the function $\mathcal{L}$:

$$\frac{d^2 \log \mathcal{L}}{dp^2}(0.7) \approx -48 < 0$$

Figure 2.1 shows the probability density graph for the Bernoulli's scheme with $p = 7/10$.

2.2 Second example: analog-to-digital converter (ADC)

Let us imagine that we have a constant physical quantity that we want to measure; for example, it can be a length to measure with a ruler or a voltage with a voltmeter. In the real world, any measurement gives *an approximation* of this value, but not the value itself. The methods I am describing here were developed by Gauss at the end of the 18th century, when he measured the orbits of celestial bodies [1]. [**?**]

For example, if we measure the battery voltage N times, we get N different measurements. Which of them should we take? All of them! So, let us say that we have N measurements U_j:

$$\{U_j\}_{j=1}^N$$

[1]Note that Legendre has published an equivalent method in 1805, whereas Gauss' first publication is dated by 1809. Gauss has always claimed that he had been using the method since 1795, and this is a very famous priority dispute [**?**] in the history of statistics. There are, however, numerous evidence to support the thesis that Gauss possessed the method before Legendre, but he was late in his communication.

Let us suppose that each measurement U_j is equal to the real value plus the Gaussian noise. The noise is characterized by two parameters — the center of the Gaussian bell and its "width". In this case, the probability density can be expressed as follows:

$$p(U_j) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(U_j - U)^2}{2\sigma^2}\right)$$

That is, having N measurements U_j, our goal is to find the parameters U and σ that maximize the likelihood. The likelihood (I have already applied the logarithm) can be written as follows:

$$\begin{aligned}\log \mathcal{L}(U,\sigma) &= \log\left(\prod_{j=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(U_j - U)^2}{2\sigma^2}\right)\right) = \\ &= \sum_{j=1}^{N} \log\left(\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(U_j - U)^2}{2\sigma^2}\right)\right) = \\ &= \sum_{j=1}^{N}\left(\log\left(\frac{1}{\sqrt{2\pi}\sigma}\right) - \frac{(U_j - U)^2}{2\sigma^2}\right) = \\ &= -N\left(\log\sqrt{2\pi} + \log\sigma\right) - \frac{1}{2\sigma^2}\sum_{j=1}^{N}(U_j - U)^2\end{aligned}$$

And then everything is strictly as it used to be, we equate the partial derivatives to zero:

$$\frac{\partial \log \mathcal{L}}{\partial U} = \frac{1}{\sigma^2}\sum_{j=1}^{N}(U_j - U) = 0$$

The most plausible estimation of the unknown value U is the simple average of all measurements:

$$U = \frac{\sum\limits_{j=1}^{N} U_j}{N}$$

And the most plausible estimation of σ turns out to be the standard deviation:

$$\frac{\partial \log \mathcal{L}}{\partial \sigma} = -\frac{N}{\sigma} + \frac{1}{\sigma^3}\sum_{j=1}^{N}(U_j - U)^2 = 0$$

$$\sigma = \sqrt{\frac{\sum\limits_{j=1}^{N}(U_j - U)^2}{N}}$$

Such a convoluted way to obtain a simple average of all measurements... In my humble opinion, the result is worth the effort. By the way, averaging multiple measurements of a constant value in order to increase the accuracy of measurements is quite a standard practice. For example, ADC averaging. Note that the hypothesis of Gaussian noise is not necessary in this case, it is enough to have an unbiased noise.

2.3 Third exampe, still 1D

Let us re-consider the previous example with a small modification. Let us say that we want to measure the resistance of a resistor. We have a bench top power supply with current regulation. That is, we control the current flowing through the resistance and we can measure the voltage required for this current. So, our "ohmmeter" evaluates the resistance through N meausrements U_j for each reference current I_j:

$$\{I_j, U_j\}_{j=1}^{N}$$

If we draw these points on a chart (Figure 2.3), the Ohm's law tells us that we are looking for the slope of the blue line that approximates the measurements.

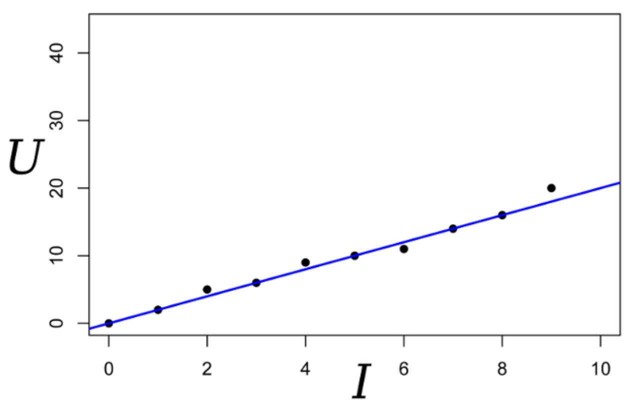


Figure 2.3: Having N meausrements U_j for each reference current I_j, we are looking for the slope of the blue line that approximates the measurements through the Ohm's law.

Let us write the expression of the (logarithm of) likelihood of the parameters:

$$\log \mathcal{L}(R,\sigma) = \log\left(\prod_{j=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(U_j - RI_j)^2}{2\sigma^2}\right)\right) =$$
$$= -N\left(\log\sqrt{2\pi} + \log\sigma\right) - \frac{1}{2\sigma^2}\sum_{j=1}^{N}(U_j - RI_j)^2$$

As usual, we equate the partial derivatives to zero:

$$\frac{\partial \log \mathcal{L}}{\partial R} = -\frac{1}{2\sigma^2}\sum_{j=1}^{N} -2I_j(U_j - RI_j) =$$
$$= \frac{1}{\sigma^2}\left(\sum_{j=1}^{N} I_jU_j - R\sum_{j=1}^{N} I_j^2\right) = 0$$

Then the most plausible resistance R can be found with the following formula:

$$R = \frac{\sum\limits_{j=1}^{N} I_jU_j}{\sum\limits_{j=1}^{N} I_j^2}$$

This result is somewhat less obvious than the simple average of all measurements in the previous example. Note that if we take one hundred measurements with $\approx 1A$ reference current and one measurement with $\approx 1kA$ reference current, then the first hundred measurements would barely affect the result. Let's remember this fact, we will need it later.

2.4 Fourth example: back to the least squares

You have probably already noticed that in the last two examples, maximizing the logarithm of the likelihood is equivalent to minimizing the sum of squared estimation errors. Let us consider one more example. Say we want to calibrate a spring scale with a help of reference weights. Suppose we have N reference weights of mass x_j; we weigh them with the scale and measure the length of the spring. So, we have N spring lengths y_j:

$$\{x_j, y_j\}_{j=1}^{N}$$

Hooke's law tells us that spring stretches linearly on the force applied; this force includes the reference weight and the weight of the spring itself. Let us denote the spring stiffness as a, and the spring length streched under under its own weight as b. Then we can express the plausibility of our measurements (still under the Gaussian measurement noise hypothesis) in this way:

$$\log \mathcal{L}(a,b,\sigma) = \log\left(\prod_{j=1}^{N} \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(y_j - ax_j - b)^2}{2\sigma^2}\right)\right) =$$
$$= -N\left(\log\sqrt{2\pi} + \log\sigma\right) - \frac{1}{2\sigma^2}\sum_{j=1}^{N}(y_j - ax_j - b)^2$$

Maximizing the likelihood of $\mathcal{L}$ is equivalent to minimizing the sum of the squared estimation error, i.e., we are looking for the minimum of the function S defined as follows:

$$S(a,b) = \sum_{j=1}^{N}(y_j - ax_j - b)^2$$

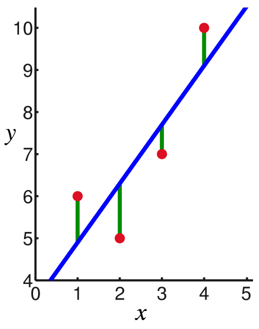


Figure 2.4: To calibrate the spring scale, we can solve the linear regression problem.

Figure 2.4 illustrates the formula: we are looking for such a straight line that minimizes the sum of squared lengths of green segments. And then the derivation is quite straightforward:

$$\frac{\partial S}{\partial a} = \sum_{j=1}^{N} 2x_j(ax_j + b - y_j) = 0$$
$$\frac{\partial S}{\partial b} = \sum_{j=1}^{N} 2(ax_j + b - y_j) = 0$$

We obtain a system of two linear equations with two unknowns:

$$\begin{cases} a\sum\limits_{j=1}^{N} x_j^2 + b\sum\limits_{j=1}^{N} x_j & = \sum\limits_{j=1}^{N} x_j y_j \\ a\sum\limits_{j=1}^{N} x_j + bN & = \sum\limits_{j=1}^{N} y_j \end{cases}$$

Use your favorite method to obtain the following solution:

$$a = \frac{N\sum\limits_{j=1}^{N} x_j y_j - \sum\limits_{j=1}^{N} x_j \sum\limits_{j=1}^{N} y_j}{N\sum\limits_{j=1}^{N} x_j^2 - \left(\sum\limits_{j=1}^{N} x_j\right)^2}$$
$$b = \frac{1}{N}\left(\sum_{j=1}^{N} y_j - a\sum_{j=1}^{N} x_j\right)$$

Conclusion

The least squares method is a particular case of maximizing likelihood in cases where the probability density is Gaussian. If the density is not Gaussian, the least squares approach can produce an estimate different from the MLE (maximum likelihood estimation). By the way, Gauss conjectured that the type of noise is of no importance, and the only thing that matters is the independence of trials.

As you have already noticed, the more we parameters we have, the more cumbersome the analytical solutions are. Fortunately, we are not living in XVIII century anymore, we have computers! Next we will try to build a geometric intuition on least squares, and see how can least squares problems be efficiently implemented.

Chapter 3

Introduction to systems of linear equations

3.1 The Jacobi and Gauss-Seidel iterative methods

$$\begin{cases} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = b_2 \\ & & & & \vdots & & & \\ a_{n1}x_1 & + & a_{n2}x_2 & + & \cdots & + & a_{nn}x_n & = b_n \end{cases}$$

$$\begin{aligned} x_1 &= \frac{1}{a_{11}}(b_1 - a_{12}x_2 - a_{13}x_3 - \cdots - a_{1n}x_n) \\ x_2 &= \frac{1}{a_{22}}(b_2 - a_{21}x_1 - a_{23}x_3 - \cdots - a_{2n}x_n) \\ &\vdots \\ x_n &= \frac{1}{a_{nn}}(b_n - a_{n1}x_1 - a_{n2}x_2 - \cdots - a_{n,n-1}x_{n-1}) \end{aligned}$$

$$\vec{x}^{(0)} = \left(x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}\right)$$

$$\vec{x}^{(1)} = \left(x_1^{(1)}, x_2^{(1)}, \ldots, x_n^{(1)}\right)$$

$$\begin{aligned} x_1^{(1)} &= \frac{1}{a_{11}}(b_1 - a_{12}x_2^{(0)} - a_{13}x_3^{(0)} - \cdots - a_{1n}x_n^{(0)}) \\ x_2^{(1)} &= \frac{1}{a_{22}}(b_2 - a_{21}x_1^{(0)} - a_{23}x_3^{(0)} - \cdots - a_{2n}x_n^{(0)}) \\ &\vdots \\ x_n^{(1)} &= \frac{1}{a_{nn}}(b_n - a_{n1}x_1^{(0)} - a_{n2}x_2^{(0)} - \cdots - a_{n,n-1}x_{n-1}^{(0)}) \end{aligned}$$

$$\vec{x}^{(k)} = \left(x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)}\right)$$

$$x_i^{(k)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1, j\neq i}^{n} a_{ij}x_j^{(k-1)}\right), \quad \text{for } i = 1, 2, \ldots, n$$

$$x_i^{(k)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}\right), \quad \text{for } i = 1, 2, \ldots, n$$

3.2 Smooth an array

The time has come to write some code. Let us examine the following program:

```python
# initialize the data array
x = [0.] * 32
x[0] = x[31] = 1.
x[18] = 2.

# smooth the array values
for iter in range(128):
    x[0] = x[1]
    for i in range(1, len(x)-1):
        x[i] = (x[i-1]+x[i+1])/2.
    x[-1] = x[-2]
```

Top left image of the Figure 3.1 shows the initialization of the array x.

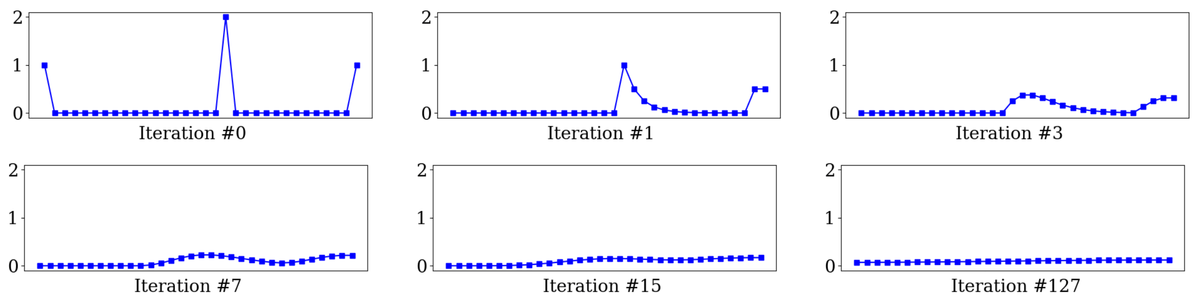


Figure 3.1: Smoothing an array: first 128 iterations of the program from section 3.2.

$$
\left\{
\begin{array}{ccccccccccl}
-x_0 & +2x_1 & -x_2 & & & & & & & & = 0 \\
 & -x_1 & +2x_2 & -x_3 & & & & & & & = 0 \\
 & & -x_2 & +2x_3 & -x_4 & & & & & & = 0 \\
 & & & & & \ddots & & & & & \vdots \\
 & & & & & & -x_{27} & +2x_{28} & -x_{29} & & = 0 \\
 & & & & & & & -x_{28} & +2x_{29} & -x_{30} & = 0 \\
 & & & & & & & & -x_{29} & +2x_{30} \quad -x_{31} & = 0
\end{array}
\right.
$$

3.3 Add some constraints

```python
# initialize the data array
x = [0.] * 32
x[0] = x[31] = 1.
x[18] = 2.

# smooth the array values other than at indices 0,18,31
for iter in range(128):
    for i in range(0, len(x)):
        if i in [0,18,31]: continue
        x[i] = (x[i-1]+x[i+1])/2.
```

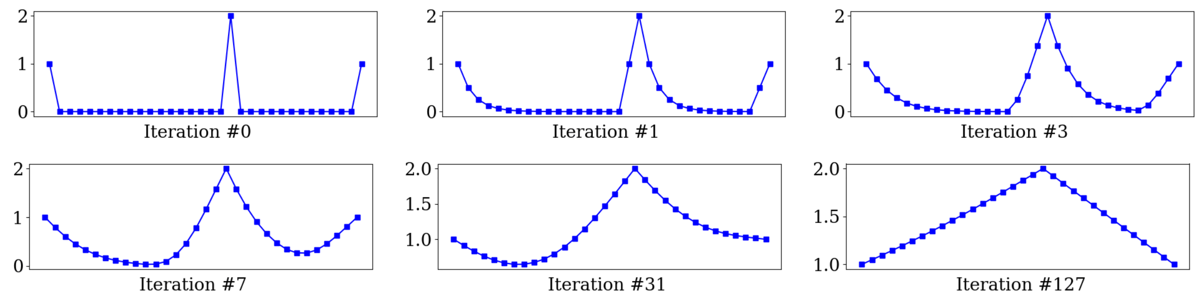


Figure 3.2: Smoothing an array.

$$
\left\{
\begin{array}{l}
2x_1 \quad -x_2 = x_0 \\
-x_1 \quad +2x_2 \quad -x_3 = 0 \\
-x_2 \quad +2x_3 \quad -x_4 = 0 \\
\ddots \qquad \vdots \\
-x_{15} \quad +2x_{16} \quad -x_{17} = 0 \\
-x_{16} \quad +2x_{17} = x_{18} \\
+2x_{19} \quad -x_{20} = x_{18} \\
-x_{19} \quad +2x_{20} \quad -x_{21} = 0 \\
\ddots \qquad \vdots \\
-x_{27} \quad +2x_{28} \quad -x_{29} = 0 \\
-x_{28} \quad +2x_{29} \quad -x_{30} = 0 \\
-x_{29} \quad +2x_{30} = x_{31}
\end{array}
\right.
$$

3.4 RHS

```python
# initialize the data array
x = [0.] * 32
x[0] = x[31] = 1.
x[18] = 2.

c = 23./32**2 # the curvature to prescribe, some arbitrary constant

# smooth the array values other than at indices 0,18,31
for iter in range(128):
    for i in range(0, len(x)):
        if i in [0,18,31]: continue
        x[i] = (x[i-1]+x[i+1]+c)/2.
```

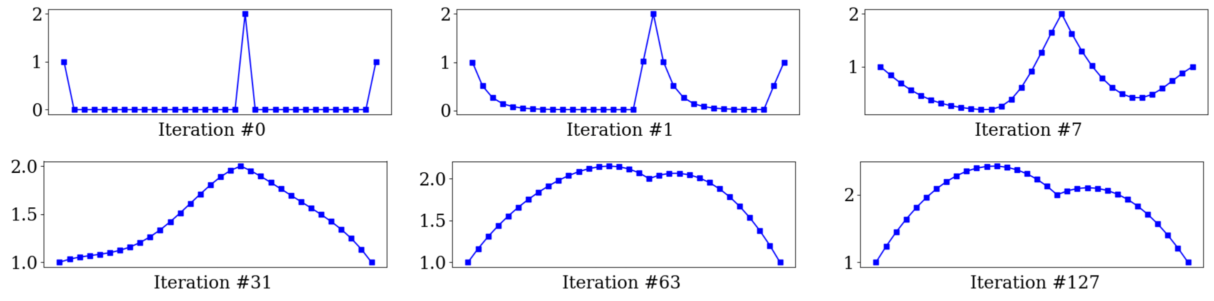


Figure 3.3: Smoothing an array.

3.5 3D example

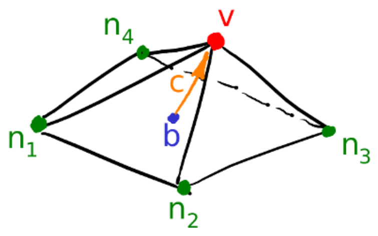


Figure 3.4: .

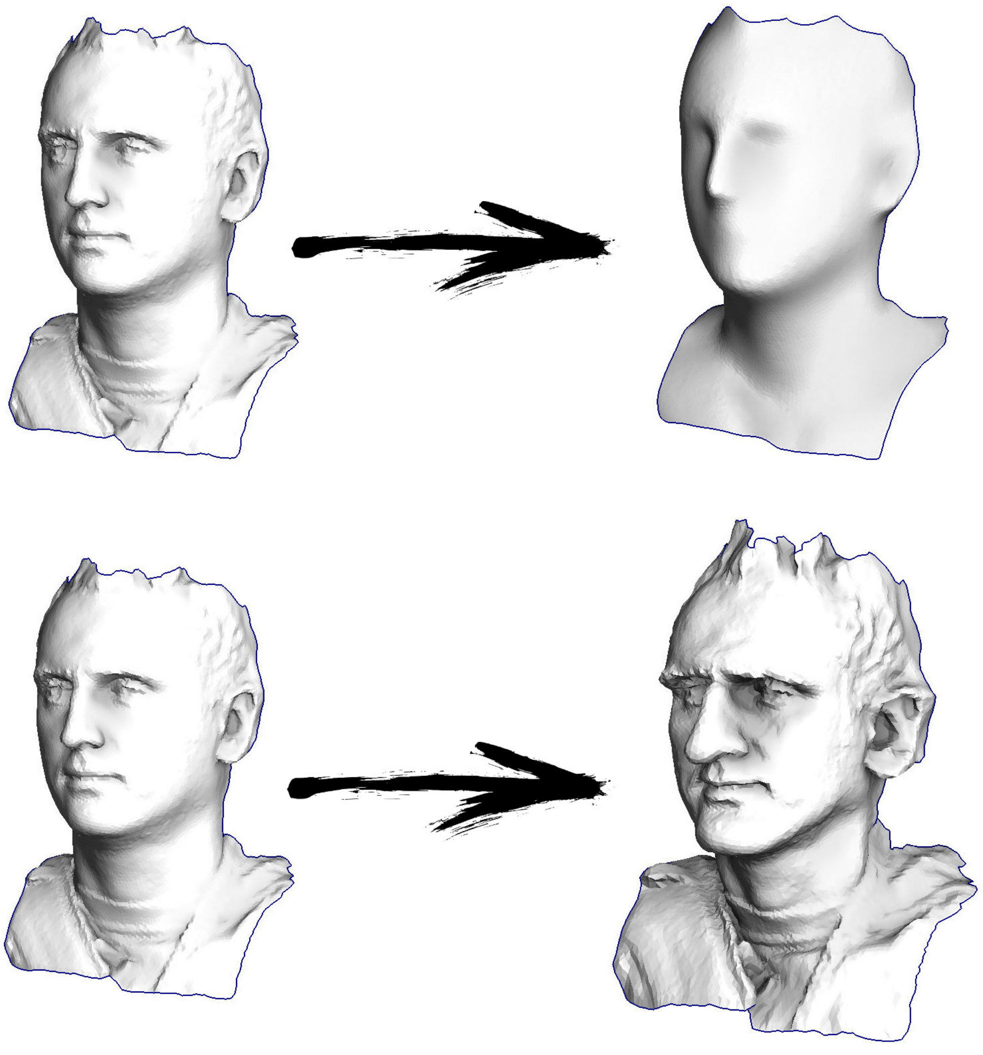


Figure 3.5: .

Chapter 4

Finite elements example: the Galerkin weighted residual method

In this section we will consider a simple boundary value problem and its relation to systems of linear equations. The boundary value problem is the problem of finding a solution to a given differential equation satisfying boundary conditions at the the boundary of a region. Let us start with a ground truth function defined as

$$\varphi(x) := \frac{1}{6}x^3 + \frac{1}{2}x^2 + \frac{1}{3}x.$$

This function is unknown, and we list it here to compare our solution to the ground truth. Let us define a function $f(x) := x + 1$. It is simply the second derivative of $\varphi(x)$, but remember that $\varphi(x)$ is not known! So, the problem is to find a function $\varphi(x)$ with prescribed second derivative and constrained at the limits of a region. One possible instance of the boundary value problem can be written as follows:

$$\begin{cases} \dfrac{d^2}{dx^2}\varphi(x) = f(x), & 0 < x < 1 \\ \varphi(0) = 0, & \varphi(1) = 1 \end{cases}$$

Here $f(x)$ is known, $\varphi(x)$ is unknown, but we have specified its values at the boundary.

In general, finite elements method searches for an *approximated* solution of the problem. Let us split the interval in n parts, these subsegments are the *elements*. For example, for $n = 3$ we can define four equispaced nodes x_i:

$$x_0 := 0, \quad x_1 := \frac{1}{3}, \quad x_2 := \frac{2}{3}, \quad x_3 := 1.$$

Note that equal distance is chosen here for the simplicity of presentation, and is not a requirement for the method. Then a set of functions is to be defined over the elements; the approximated solution $\tilde{\varphi}$ is defined as a weighted sum of these functions.

For example, let us choose a set of linear functions:

$$w_i(x) := \begin{cases} \dfrac{x - x_{i-1}}{x_i - x_{i-1}}, & x_{i-1} \le x \le x_i \\ \dfrac{x_{i+1} - x}{x_{i+1} - x_i}, & x_i \le x \le x_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

TODO : *figure, hat functions*

Then the approximate solution is a linear combination of the weighting functions:

$$\tilde{\varphi}(x) := 0 \cdot w_0(x) + b \cdot w_1(x) + c \cdot w_2(x) + 1 \cdot w_3(x),$$

where the coefficients of $w_0(x)$ and $w_3(x)$ are the direct consequence of the boundary condition $\varphi(0) = 0$, $\varphi(1) = 1$; note that b and c give the values of the approximation for the points $x = \frac{1}{3}$ and $x = \frac{2}{3}$, respectively.

$$\int_0^1 w_i \cdot \left(\frac{d^2\tilde{\varphi}}{dx^2} - f \right) dx = 0, \quad i = 1 \dots n-1$$

TODO : *figure, residual orthogonal to the solution space* Note that $w_i(x)$ is equal to zero outside the interval $[x_{i-1}, x_{i+1}]$, this allows us to rewrite the system:

$$\int\limits_{x_{i-1}}^{x_{i+1}} w_i \cdot \frac{d^2\tilde{\varphi}}{dx^2}\,dx - \int\limits_{x_{i-1}}^{x_{i+1}} w_i \cdot f\,dx = 0, \quad i = 1\dots n-1$$

Most often both integrals are evaluated numerically, or symbolic calculations are peformed over the left integral only. The left integral is a dot product between the basis functions and the differential operator defined on a combination of basis functions; it depends on the choice of the basis and can be precomputed. As our problem is very simple, we will find the solution symbolically.

Attention! The next step will be done on a slippery ground. Our weighting functions are not differentiable everywhere, and caution must be taken. Anyhow, let us integrate by parts the first integral:

$$\int\limits_{x_{i-1}}^{x_{i+1}} w_i \cdot \frac{d^2\tilde{\varphi}}{dx^2}\,dx = \int\limits_{x_{i-1}}^{x_i} w_i \cdot \frac{d^2\tilde{\varphi}}{dx^2}\,dx + \int\limits_{x_i}^{x_{i+1}} w_i \cdot \frac{d^2\tilde{\varphi}}{dx^2}\,dx =$$

$$= \underbrace{\lim_{\varepsilon\to 0} w_i \cdot \frac{d\tilde{\varphi}}{dx}\Big|_{x_{i-1}+\varepsilon}^{x_i-\varepsilon} + \lim_{\varepsilon\to 0} w_i \cdot \frac{d\tilde{\varphi}}{dx}\Big|_{x_i+\varepsilon}^{x_{i+1}-\varepsilon}}_{=0} - \int\limits_{x_{i-1}}^{x_{i+1}} \frac{dw_i}{dx} \cdot \frac{d\tilde{\varphi}}{dx}\,dx =$$

$$= -\int\limits_{x_{i-1}}^{x_{i+1}} \frac{dw_i}{dx} \cdot \frac{d\tilde{\varphi}}{dx}\,dx$$

It allows us to rewrite the system of equations:

$$\int\limits_{x_{i-1}}^{x_{i+1}} \frac{dw_i}{dx} \cdot \frac{d\tilde{\varphi}}{dx}\,dx + \int\limits_{x_{i-1}}^{x_{i+1}} w_i \cdot f\,dx = 0, \quad i = 1\dots n-1 \tag{4.1}$$

$$\frac{d\tilde{\varphi}}{dx}(x) := \begin{cases} 3b, & 0 < x < \frac{1}{3} \\ -3b+3c, & \frac{1}{3} < x < \frac{2}{3} \\ -3c+3, & \frac{2}{3} < x < 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{dw_1}{dx}(x) := \begin{cases} 3, & 0 < x < \frac{1}{3} \\ -3, & \frac{1}{3} < x < \frac{2}{3} \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{dw_2}{dx}(x) := \begin{cases} 3, & \frac{1}{3} < x < \frac{2}{3} \\ -3, & \frac{2}{3} < x < 1 \\ 0 & \text{otherwise} \end{cases}$$

Then the system (4.1) can be rewritten as follows:

$$\begin{cases} \int\limits_0^{1/3} 3\cdot 3b\,dx + \int\limits_{1/3}^{2/3} -3\cdot(-3b+3c)\,dx + \int\limits_0^{1/3} 3x(x+1)\,dx + \int\limits_{1/3}^{2/3} (2-3x)(x+1)\,dx = 0 \\ \int\limits_{1/3}^{2/3} 3\cdot(-3b+3c)\,dx + \int\limits_{2/3}^{1} -3\cdot(-3c+3)\,dx + \int\limits_{1/3}^{2/3} (3x-1)(x+1)\,dx + \int\limits_{2/3}^{1} (3-3x)(x+1)\,dx = 0 \end{cases}$$

$$\begin{cases} 2b - c = -\dfrac{4}{27} \\ -b + 2c = \dfrac{22}{27} \end{cases}$$

$b = \frac{14}{81}$, $c = \frac{40}{81}$

Chapter 5

Minimization of quadratic functions and linear systems

Recall that the main goal is to study least squares, therefore our main tool will be the minimization of quadratic functions; however, before we start using this power tool, we need to find where its on/off button is located. First of all, we need to recall what a matrix is; then we will revisit the definition of a positive numbers, and only then we will attack minimization of quadratic functions.

5.1 Matrices and numbers

In this sections, matrices will be omnipresent, so let's remember what it is. Do not peek further down the text, pause for a few seconds, and try to formulate what the matrix is.

5.1.1 Different interpretations of matrices

The answer is very simple. A matrix is just a locker that stores stuff. Each piece of stuff lies in its own cell, cells are grouped in rows and columns. In our particular case, we store real numbers; for a programmer the easiest way to imagine a matrix A is something like:

```
float A[m][n];
```

Why would we need a storage like this? What does it describe? Maybe I will upset you, but the matrix by itself does not describe anything, it stores stuff. For example, you can store coefficients of a function in it. Let us put aside matrices for a second imagine that we have a number a. What does it mean? Who knows what it means... For example, it can be a coefficient inside a function that takes one number as an input and gives another number as an output:

$$f(x) : \mathbb{R} \to \mathbb{R}$$

One possible instance of such a function a mathematicion could write down as:

$$f(x) = ax$$

In the programmers' world it would look something like this:

```
float f(float x) {
    return a*x;
}
```

On the other hand, why this function and not another one? Let's take another one!

$$f(x) = ax^2$$

A programmer would write it like this:

```
float f(float x) {
    return x*a*x;
}
```

One of these functions is linear and the other is quadratic. Which one is correct? Neither one. The number a does not define it, it just stores a value! Build the function you need.

The same thing happens to matrices, they give storage space when simple numbers (scalars) do not suffice, a matrix is a sort of an hyper-number. The addition and multiplication operations are defined over matrices just as over numbers.

Let us suppose that we have a 2×2 matrix A:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

The matrix does not mean anything by itself, for example, it can be interpreted as a linear function:

$$f(x) : \mathbb{R}^2 \to \mathbb{R}^2, \quad f(x) = Ax$$

Here goes the programmer's view on the function:

```
vector<float> f(vector<float> x) {
    return vector<float>{a11*x[0] + a12*x[1],  a21*x[0] + a22*x[1]};
}
```

This function maps a two-dimensional vector to a two-dimensional vector. Graphically, it is convenient to imagine it as an image transformation: we give an input image, and the output is the stretched and/or rotated (maybe even mirrored!) version. The top row of Figure 5.1 provides few different examples of this interpretation of matrices.

On the other hand, nothing prevents to interpret the matrix A as a function that maps a vector to a scalar:

$$f(x) : \mathbb{R}^2 \to \mathbb{R}, \quad f(x) = x^\top A x = \sum_i \sum_j a_{ij} x_i x_j$$

Note that the square is not very well defined for the vectors, so I cannot write x^2 as I wrote in the case of ordinary numbers. For those who are not at ease with matrix multiplications, I highly recommend to revisit it right now and check that the expression $x^\top A x$ indeed produces a scalar value. To this end, we can explicitly put brackets $x^\top A x = (x^\top A)x$. Recall that in this particular example x is a two-dimensional vector (stored in a 2×1 matrix). Let us write all the matrix dimensions explicitly:

$$\underbrace{\underbrace{\left(\underbrace{x^\top}_{1\times 2} \times \underbrace{A}_{2\times 2}\right)}_{1\times 2} \times \underbrace{x}_{2\times 1}}_{1\times 1}$$

Returning to the cozy world of programmers, we can write the same quadratic function as follows:

```
float f(vector<float> x) {
    return x[0]*a11*x[0] + x[0]*a12*x[1] + x[1]*a21*x[0] + x[1]*a22*x[1];
}
```

5.1.2 What is a positive number?

Allow me to ask a very stupid question: what is a positive number? We have a great tool called the predicate "greater than" $>$. Do not be in a hurry to answer that the number a is positive if and only if $a > 0$, it would be too easy. Let us define the positivity as follows:

Definition 1. *The real number a is positive if and only if for all non-zero real $x \in \mathbb{R}$, $x \neq 0$ the condition $ax^2 > 0$ is satisfied.*

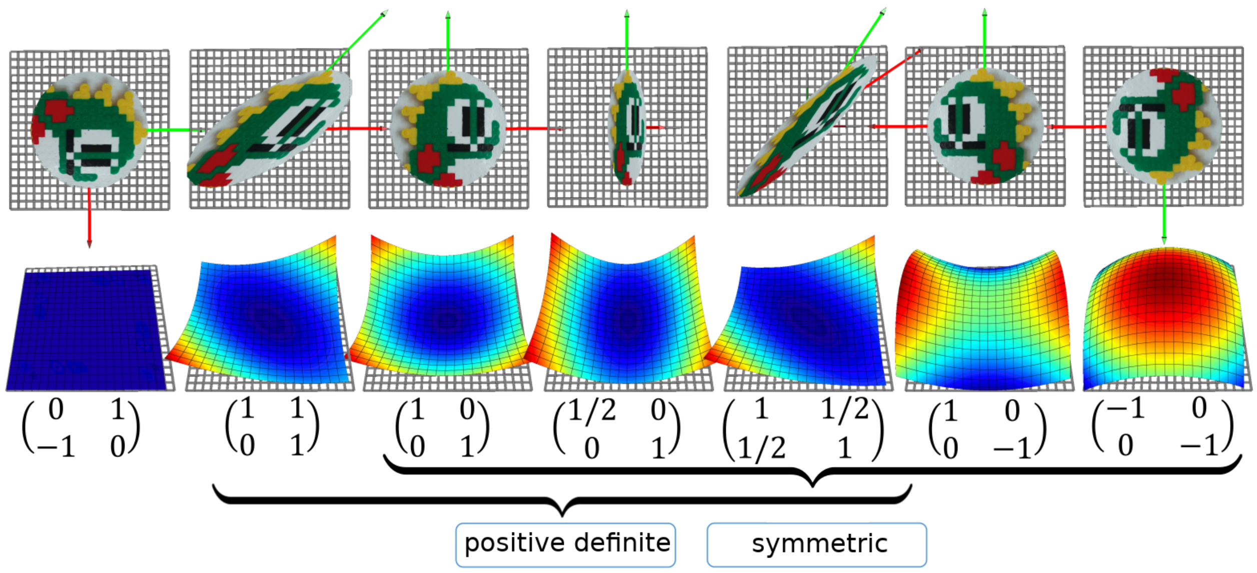

$\begin{pmatrix}0 & 1\\-1 & 0\end{pmatrix}$
$\begin{pmatrix}1 & 1\\0 & 1\end{pmatrix}$
$\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}$
$\begin{pmatrix}1/2 & 0\\0 & 1\end{pmatrix}$
$\begin{pmatrix}1 & 1/2\\1/2 & 1\end{pmatrix}$
$\begin{pmatrix}1 & 0\\0 & -1\end{pmatrix}$
$\begin{pmatrix}-1 & 0\\0 & -1\end{pmatrix}$
positive definite
symmetric

Figure 5.1: Seven examples of 2×2 matrices, some of them are positive definite and/or symmetric. **Top row:** the matrices are interpreted as linear functions $f(x) : \mathbb{R}^2 \to \mathbb{R}^2$. **Middle row:** the matrices are interpreted as quadratic functions $f(x) : \mathbb{R}^2 \to \mathbb{R}$.

Looks pretty awkward, but it applies perfectly to the matrices:

Definition 2. *The square matrix A is called positive definite if for any non-zero x the condition $x^\top Ax > 0$ is met, i.e. the corresponding quadratic form is strictly positive everywhere except at the origin.*

What do we need the positivity for? As we have already mentioned, our main tool will be the minimization of quadratic functions. It would be nice to be sure that the minimum exists! For example, the function $f(x) = -x^2$ clearly has no minimum, because the number -1 is not positive, both branches of the parabola $f(x)$ look down. Positive definite matrices guarantee that the corresponding quadratic forms form a paraboloid with a (unique) minimum. Refer to the Figure 5.1 for an illustration.

Thus, we will work with a generalization of positive numbers, namely, positive definite matrices. Moreover, in our particular case, the matrices will be symmetric! Note that quite often, when people talk about positive definiteness, they also imply symmetry. This can be partly explained by the following observation (optional for the understanding of the rest of the text):

A digression on quadratic forms and matrix symmetry

Let us consider a quadratic form $x^\top Mx$ for an arbitrary matrix M. Next we add and subtract a half ot its transpose:

$$M = \underbrace{\frac{1}{2}(M + M^\top)}_{:=M_s} + \underbrace{\frac{1}{2}(M - M^\top)}_{:=M_a} = M_s + M_a$$

The matrix M_s is symmetric: $M_s^\top = M_s$; the matrix M_a is antisymmetric: $M_a^\top = -M_a$. A remarkable fact is that for any antisymmetric matrix the corresponding quadratic form is equal to zero everywhere. This follows from the following observation:

$$q = x^\top M_a x = (x^\top M_a^\top x)^\top = -(x^\top M_a x)^\top = -q$$

It means that the quadratic form $x^\top M_a x$ equals q and $-q$ at the same time, and the only way to have this condition is to have $q \equiv 0$. From this fact it follows that for an arbitrary matrix M the corresponding quadratic form $x^\top Mx$ can be expressed through the symmetric matrix M_s as well:

$$x^\top Mx = x^\top (M_s + M_a)x = x^\top M_s x + x^\top M_a x = x^\top M_s x.$$

5.2 Ищем минимум квадратичной функции

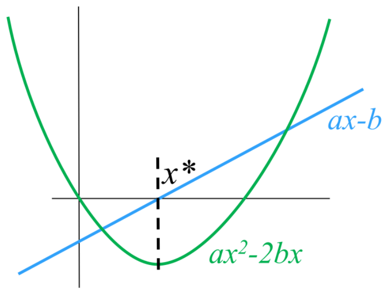


Figure 5.2: В одномерном мире решение x^* уравнения $ax - b = 0$ также является решением проблемы $\arg\min_x(ax^2 - 2bx)$.

Вернёмся ненадолго в одномерный мир; я хочу найти минимум функции $f(x) = ax^2 - 2bx$. Число a положительно, поэтому минимум существует; чтобы его найти, приравняем нулю соответствующую производную: $\frac{d}{dx}f(x) = 0$. Продифференцировать одномерную квадратичную функцию труда не составляет: $\frac{d}{dx}f(x) = 2ax - 2b = 0$; поэтому наша проблема сводится к решению уравнения $ax - b = 0$, откуда путём страшных усилий получам решение $x^* = b/a$. Рисунок 5.2 иллюстрирует эквивалентность двух проблем: решение x^* уравнения $ax - b = 0$ совпадает с решением уравнения $\arg\min_x(ax^2 - 2bx)$.

Я клоню к тому, что нашей глобальной целью является минимизация квадратичных функций, мы же про наименьшие квадраты говорим. Но при этом что мы действительно умеем делать хорошо, так это решать линейные уравнения (и системы линейных уравнений). И очень здорово, что одно эквивалентно другому! Единственное, что нам осталось, так это убедиться, что эта эквивалентность в одномерном мире распространяется и на случай n переменных. Чтобы это проверить, для начала докажем три теоремы.

5.2.1 Три теоремы, или дифференцируем матричные выражения

Первая теорема гласит о том, что матрицы 1×1 инварианты относительно транспонирования:

Theorem 1. $x \in \mathbb{R} \Rightarrow x^\top = x$

Доказательство настолько сложное, что для краткости я вынужден его опустить, но попробуйте найти его самостоятельно.

Вторая теорема позволяет дифференцировать линейные функции. В случае вещественной функции одной переменной мы знаем, что $\frac{d}{dx}(bx) = b$, но что происходит в случае вещественной функции n переменных?

Theorem 2. $\nabla b^\top x = \nabla x^\top b = b$

То есть, никаких сюрпризов, просто матричная запись того же самого школьного результата. Доказательство крайне прямолинейное, достаточно просто записать определение градиента:

$$\nabla(b^\top x) = \begin{bmatrix} \frac{\partial(b^\top x)}{\partial x_1} \\ \vdots \\ \frac{\partial(b^\top x)}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial(b_1x_1+\dots+b_nx_n)}{\partial x_1} \\ \vdots \\ \frac{\partial(b_1x_1+\dots+b_nx_n)}{\partial x_n} \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} = b$$

Применив первую теорему $b^\top x = x^\top b$, мы завершаем доказательство.

Теперь переходим к квадратичным формам. Мы знаем, что в случае вещественной функции одной переменной $\frac{d}{dx}(ax^2) = 2ax$, а что будет в случае квадратичной формы?

Theorem 3. $\nabla(x^\top Ax) = (A + A^\top)x$

Кстати, обратите внимание, что если матрица A симметрична, то теорема гласит, что $\nabla(x^\top Ax) = 2Ax$.

Это доказательство тоже прямолинейно, просто запишем квадратичную форму как двойную сумму:

$$x^\top Ax = \sum_i \sum_j a_{ij} x_i x_j$$

А затем посмотрим, чему равна частная производная этой двойной суммы по переменной x_i:

$$\begin{aligned}
\frac{\partial(x^\top Ax)}{\partial x_i} &= \frac{\partial}{\partial x_i}\left(\sum_{k_1}\sum_{k_2} a_{k_1k_2}x_{k_1}x_{k_2}\right) = \\
&= \frac{\partial}{\partial x_i}\left(\underbrace{\sum_{k_1\neq i}\sum_{k_2\neq i} a_{ik_2}x_{k_1}x_{k_2}}_{k_1\neq i, k_2\neq i} + \underbrace{\sum_{k_2\neq i} a_{ik_2}x_i x_{k_2}}_{k_1=i,k_2\neq i} + \underbrace{\sum_{k_1\neq i} a_{k_1 i}x_{k_1}x_i}_{k_1\neq i,k_2=i} + \underbrace{a_{ii}x_i^2}_{k_1=i,k_2=i}\right) = \\
&= \sum_{k_2\neq i} a_{ik_2}x_{k_2} + \sum_{k_1\neq i} a_{k_1 i}x_{k_1} + 2a_{ii}x_i = \\
&= \sum_{k_2} a_{ik_2}x_{k_2} + \sum_{k_1} a_{k_1 i}x_{k_1} = \\
&= \sum_j (a_{ij} + a_{ji})x_j
\end{aligned}$$

Я разбил двойную сумму на четыре случая, которые выделены фигурными скобками. Каждый из этих четырёх случаев дифференцируется тривиально. Осталось сделать последнее действие, собрать частные производные в вектор градиента:

$$\nabla(x^\top Ax) = \begin{bmatrix} \frac{\partial(x^\top Ax)}{\partial x_1} \\ \vdots \\ \frac{\partial(x^\top Ax)}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \sum\limits_j (a_{1j} + a_{j1})x_j \\ \vdots \\ \sum\limits_j (a_{nj} + a_{jn})x_j \end{bmatrix} = (A + A^\top)x$$

5.2.2 Минимум квадратичной функции и решение линейной системы

Давайте не забывать направление движения. Мы видели, что при положительном числе a в случае вещественных функций одной перменной решить уравнение $ax = b$ или минимизировать квадратичную функцию $\arg\min\limits_x(ax^2 - 2bx)$ — это одно и то же.

Мы хотим показать соответствующую связь в случае симметричной положительно определённой матрицы A. Итак, мы хотим найти минимум квадратичной функции

$$\arg\min_{x\in\mathbb{R}^n}(x^\top Ax - 2b^\top x).$$

Ровно как и в школе, приравняем нулю производную:

$$\nabla(x^\top Ax - 2b^\top x) = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Оператор Гамильтона линеен, поэтому мы можем переписать наше уравнение в следующем виде:

$$\nabla(x^\top Ax) - 2\nabla(b^\top x) = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

Теперь применим вторую и третью теоремы о дифференцировании:

$$(A + A^\top)x - 2b = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

Вспомним, что A симметрична, и поделим уравнение на два, получаем нужную нам систему линейных уравнений:

$$Ax = b$$

Ура-ура, перейдя от одной переменной к многим, мы не потеряли ровным счётом ничего, и можем эффективно минимизировать квадратичные функции!

5.3 Переходим к наименьшим квадратам

Наконец мы можем перейти к основному содержанию этой лекции. Представьте, что у нас есть две точки на плоскости (x_1, y_1) и (x_2, y_2), и мы хотим найти уравнение прямой, проходящей через эти две точки. Уравнение прямой можно записать в виде $y = \alpha x + \beta$, то есть, наша задача это найти коэффициенты α и β. Это упражнение для седьмого-восьмого класса школы, запишем систему уравнений:

$$\begin{cases} \alpha x_1 + \beta = y_1 \\ \alpha x_2 + \beta = y_2 \end{cases}$$

У нас два уравнения с двумя неизвестными (α и β), остальное известно. В общем случае решение существует и единственно. Для удобства перепишем ту же самую систему в матричном виде:

$$\underbrace{\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \end{bmatrix}}_{:=A} \underbrace{\begin{bmatrix} \alpha \\ \beta \end{bmatrix}}_{:=x} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}}_{:=b}$$

Получим уравнение типа $Ax = b$, которое тривиально решается $x^* = A^{-1}b$.

А теперь представим, что у нас есть **три** точки, через которые нужно провести прямую:

$$\begin{cases} \alpha x_1 + \beta = y_1 \\ \alpha x_2 + \beta = y_2 \\ \alpha x_3 + \beta = y_3 \end{cases}$$

В матричном виде эта система запишется следующим образом:

$$\underbrace{\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \end{bmatrix}}_{:=A\,(3\times 2)} \underbrace{\begin{bmatrix} \alpha \\ \beta \end{bmatrix}}_{:=x\,(2\times 1)} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}}_{:=b\,(3\times 1)}$$

Теперь у нас матрица A прямоугольная, и у неё просто не существует обратной! Это совершенно нормально, так как у нас всего две переменных и три уравнения, и в общем случае эта система не имеет решения. Это совеpешенно нормальная ситуация в реальном мире, когда точки - это данные зашумлённых измерений, и нам нужно найти параметры α и β, которые наилучшим образом *приближают* данные измерений. Мы этот пример уже рассматривали в первой лекции, когда калибровали безмен. Но тогда у нас решение было чисто алгебраическим и очень громоздким. Давайте попробуем более интуитивный способ.

Нашу систему можно записать в следующем виде:

$$\alpha \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}_{:=\vec{i}} + \beta \underbrace{\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}}_{:=\vec{j}} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

Или, более кратко,

$$\alpha\vec{i} + \beta\vec{j} = \vec{b}.$$

Векторы $\vec{i}$, $\vec{j}$ и $\vec{b}$ известны, нужно найти скаляры α и β. Очевидно, что линейная комбинация $\alpha\vec{i} + \beta\vec{j}$ порождает плоскость, и если вектор $\vec{b}$ в этой плоскости не лежит, то точного решения не существует; однако же мы ищем приближённое решение.

Давайте определим ошибку решения как $\vec{e}(\alpha, \beta) := \alpha\vec{i} + \beta\vec{j} - \vec{b}$. Нашей зачей является найти такие значения параметров α и β, которые минимизируют длину вектора $\vec{e}(\alpha, \beta)$. Иначе говоря, проблема записывается как $\arg\min\limits_{\alpha,\beta} \|\vec{e}(\alpha, \beta)\|$. Иллюстрация дана на рисунке 5.3.

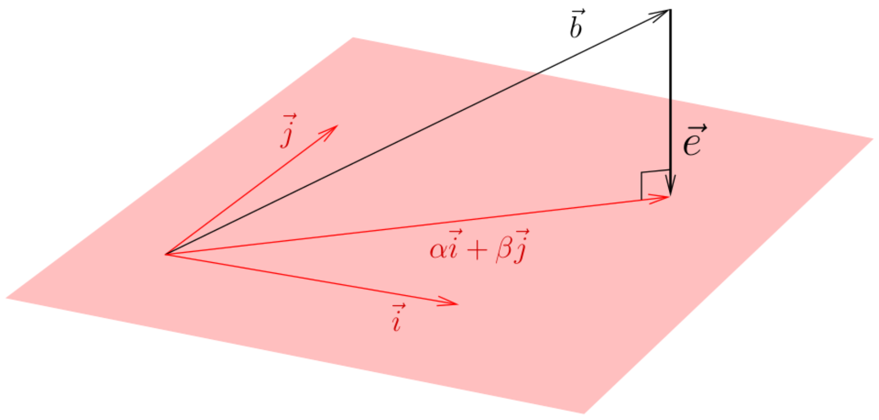


Figure 5.3: Имея заданные векторы $\vec{i}$, $\vec{j}$ и $\vec{b}$, мы стараемся минимизировать длину вектора ошибки $\vec{e}$. Очевидно, что его длина минимизируется, если он перпендикулярен плоскости, натянутой на векторы $\vec{i}$ и $\vec{j}$.

Но постойте, где же наименьшие квадраты? Сейчас будут. Функция извлечения корня $\sqrt{\cdot}$ монотонна, поэтому $\arg\min\limits_{\alpha,\beta} \|\vec{e}(\alpha, \beta)\| = \arg\min\limits_{\alpha,\beta} \|\vec{e}(\alpha, \beta)\|^2$!

Вполне очевидно, что длина вектора ошибки минимизируется, если он перпендикулярен плоскости, натянутой на векторы $\vec{i}$ и $\vec{j}$, что можно записать, приравняв нулю соответствующие скалярные произведения:

$$\begin{cases} \vec{i}^\top \vec{e}(\alpha, \beta) = 0 \\ \vec{j}^\top \vec{e}(\alpha, \beta) = 0 \end{cases}$$

В матричном виде эту же самую систему можно записать как

$$\begin{bmatrix} x_1 & x_2 & x_3 \\ 1 & 1 & 1 \end{bmatrix} \left(\alpha \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \beta \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

или же

$$\begin{bmatrix} x_1 & x_2 & x_3 \\ 1 & 1 & 1 \end{bmatrix} \left(\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Но мы на этом не остановимся, так как запись можно ещё больше сократить:

$$A^\top (Ax - b) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

И самая последняя трансформация:

$$A^\top Ax = A^\top b.$$

Давайте посчитаем размерности. Матрица A имеет размер 3×2, поэтому $A^\top A$ имеет размер 2×2. Матрица b имеет размер 3×1, но вектор $A^\top b$ имеет размер 2×1. То есть, в общем случае матрица $A^\top A$ обратима! Более того, $A^\top A$ имеет ещё пару приятных свойств.

Theorem 4. $A^\top A$ *симметрична.*

Это совсем нетрудно показать:

$$(A^\top A)^\top = A^\top (A^\top)^\top = A^\top A.$$

Theorem 5. $A^\top A$ *положительно полуопределена:* $\forall x \in \mathbb{R}^n \quad x^\top A^\top Ax \geq 0.$

Это следует из того факта, что $x^\top A^\top Ax = (Ax)^\top Ax > 0$.

Кроме того, $A^\top A$ положительно определена в том случае, если A имеет линейно независимые столбцы (ранг A равен количеству переменных в системе).

Итого, для системы с двумя неизвестными мы доказали, что минимизация квадратичной функции $\arg\min\limits_{\alpha,\beta} \|\vec{e}(\alpha,\beta)\|^2$ эквивалентна решению системы линейных уравнений $A^\top Ax = A^\top b$. Разумеется, всё это рассуждение применимо и к любому другому количеству переменных, но давайте ещё раз компактно запишем всё вместе простым алгебраическим подсчётом. Мы начнём с проблемы наименьших квадратов, придём к минимизации квадратичной функции знакомого нам вида, и из этого сделаем вывод об эквивалентности решению системы линейных уравнений. Итак, поехали:

$$\begin{aligned}
\arg\min \|Ax - b\|^2 &= \arg\min (Ax-b)^\top (Ax-b) = \\
&= \arg\min (x^\top A^\top - b^\top)(Ax-b) = \\
&= \arg\min (x^\top A^\top Ax - b^\top Ax - x^\top A^\top b + \underbrace{b^\top b}_{\text{const}}) = \\
&= \arg\min (x^\top A^\top Ax - 2b^\top Ax) = \\
&= \arg\min (x^\top \underbrace{(A^\top A)}_{:=A'} x - 2\underbrace{(A^\top b)}_{:=b'}{}^\top x)
\end{aligned}$$

Таким образом, проблема наименьших квадратов $\arg\min \|Ax-b\|^2$ эквивалентна минимизации квадратичной функции $\arg\min(x^\top A'x - 2b'^\top x)$ с (в общем случае) симметричной положительно определённой матрицей A', что, в свою очередь, эквивалентно решению системы линейных уравнений $A'x = b'$. Уфф. Теория закончилась.

Chapter 6

Conjugate gradient for geometry processing

$$\left\{\begin{array}{cccccccccccccccccc}
2x_1 & -x_2 & & & & & & & & & & & & & & & = x_0 \\
-x_1 & +2x_2 & -x_3 & & & & & & & & & & & & & & = 0 \\
 & -x_2 & +2x_3 & -x_4 & & & & & & & & & & & & & = 0 \\
 & & & & \ddots & & & & & & & & & & & & \vdots \\
 & & & & & -x_{15} & +2x_{16} & -x_{17} & & & & & & & & & = 0 \\
 & & & & & & -x_{16} & +2x_{17} & & & & & & & & & = x_{18} \\
 & & & & & & & & +2x_{19} & -x_{20} & & & & & & & = x_{18} \\
 & & & & & & & & -x_{19} & +2x_{20} & -x_{21} & & & & & & = 0 \\
 & & & & & & & & & & & \ddots & & & & & \vdots \\
 & & & & & & & & & & & & -x_{27} & +2x_{28} & -x_{29} & & = 0 \\
 & & & & & & & & & & & & & -x_{28} & +2x_{29} & -x_{30} & = 0 \\
 & & & & & & & & & & & & & & -x_{29} & +2x_{30} & = x_{31}
\end{array}\right.$$

$$\left\{\begin{array}{|c|ccccccc|c|ccccccc|c|c}
-x_0 & +x_1 & & & & & & & & & & & & & & & & = 0 \\
 & -x_1 & +x_2 & & & & & & & & & & & & & & & = 0 \\
 & & -x_2 & +x_3 & & & & & & & & & & & & & & = 0 \\
 & & & -x_3 & +x_4 & & & & & & & & & & & & & = 0 \\
 & & & & & \ddots & & & & & & & & & & & & \vdots \\
 & & & & & & -x_{16} & +x_{17} & & & & & & & & & & = 0 \\
 & & & & & & & -x_{17} & +x_{18} & & & & & & & & & = 0 \\
 & & & & & & & & -x_{18} & +x_{19} & & & & & & & & = 0 \\
 & & & & & & & & & -x_{19} & +x_{20} & & & & & & & = 0 \\
 & & & & & & & & & & & \ddots & & & & & & \vdots \\
 & & & & & & & & & & & & -x_{27} & +x_{28} & & & & = 0 \\
 & & & & & & & & & & & & & -x_{28} & +x_{29} & & & = 0 \\
 & & & & & & & & & & & & & & -x_{29} & +x_{30} & & = 0 \\
 & & & & & & & & & & & & & & & -x_{30} & +x_{31} & = 0
\end{array}\right.$$

$$\left\{\begin{array}{llllllllllllll l}
+x_1 & & & & & & & & & & & & & & = +x_0 \\
-x_1 & +x_2 & & & & & & & & & & & & & = 0 \\
 & -x_2 & +x_3 & & & & & & & & & & & & = 0 \\
 & & -x_3 & +x_4 & & & & & & & & & & & = 0 \\
 & & & & \ddots & & & & & & & & & & \vdots \\
 & & & & & -x_{16} & +x_{17} & & & & & & & & = 0 \\
 & & & & & & -x_{17} & & & & & & & & = -x_{18} \\
 & & & & & & & +x_{19} & & & & & & & = +x_{18} \\
 & & & & & & & -x_{19} & +x_{20} & & & & & & = 0 \\
 & & & & & & & & & \ddots & & & & & \vdots \\
 & & & & & & & & & & -x_{27} & +x_{28} & & & = 0 \\
 & & & & & & & & & & & -x_{28} & +x_{29} & & = 0 \\
 & & & & & & & & & & & & -x_{29} & +x_{30} & = 0 \\
 & & & & & & & & & & & & & -x_{30} & = -x_{31}
\end{array}\right.$$

$$\left\{\begin{array}{lllllllllllll l}
-x_0 & +x_1 & & & & & & & & & & & & = 0 \\
 & -x_1 & +x_2 & & & & & & & & & & & = 0 \\
 & & & \ddots & & & & & & & & & & \vdots \\
 & & & & -x_{16} & +x_{17} & & & & & & & & = 0 \\
 & & & & & -x_{17} & +x_{18} & & & & & & & = 0 \\
 & & & & & & -x_{18} & +x_{19} & & & & & & = 0 \\
 & & & & & & & -x_{19} & +x_{20} & & & & & = 0 \\
 & & & & & & & & & \ddots & & & & \vdots \\
 & & & & & & & & & & -x_{29} & +x_{30} & & = 0 \\
 & & & & & & & & & & & -x_{30} & +x_{31} & = 0 \\
100 \cdot x_0 & & & & & & & & & & & & & = 100 \cdot 1 \\
 & & & & & & 100 \cdot x_{18} & & & & & & & = 100 \cdot 2 \\
 & & & & & & & & & & & & 100 \cdot x_{31} & = 100 \cdot 1
\end{array}\right.$$

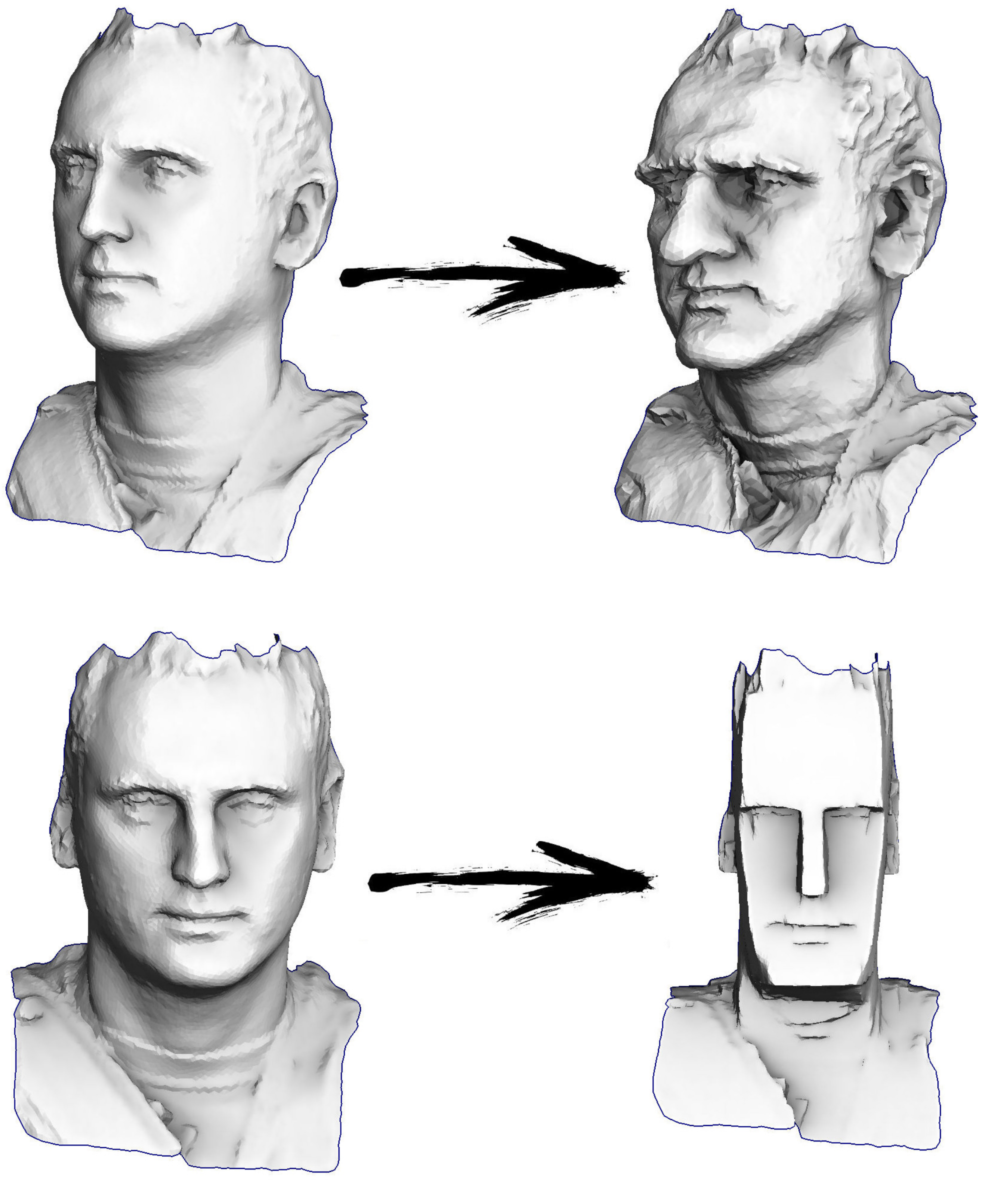


Figure 6.1: .

Chapter 7

From least squares to neural networks

7.1 Logistic regression

Two possible classes encoded as $y \in \{0,1\}$ and assume

$$p(y=1|x,w) = \frac{1}{1+e^{-w^\top x}},$$

where w is a $(m+1)$-parameter vector and the last element of x is the constant 1.

Follows that

$$p(y=0|x,w) = 1 - P(y=1|x,w) = \frac{1}{1+e^{w^\top x}}$$

Given n independently distributed data points x_i with corresponding labels y_i, we can write the log-likelihood as

$$\log \mathcal{L}(w) = \log \prod_{i=1}^n p_i(w)^{y_i}(1-p_i(w))^{1-y_i} = \sum_{i=1}^n \left(y_i w^\top x_i - \log\left(1+e^{w^\top x}\right)\right),$$

where $p_i(w) := p(y_i = 1|x_i, w)$.

To maximize $\log \mathcal{L}$, we set its derivatives to 0 and obtain

$$\frac{\partial \log \mathcal{L}}{\partial w}(w) = \sum_{i=1}^n x_i(y_i - p_i(w)) = 0,$$

so we have $m+1$ non-linear equations in w. We can rewrite the system as

$$\frac{\partial \log \mathcal{L}}{\partial w}(w) = X^\top (y-p) = 0, \tag{7.1}$$

where X is the $n \times (m+1)$ matrix whose rows are x_i, $y := \begin{bmatrix} y_1 & \dots & y_n \end{bmatrix}^\top$ and $p := \begin{bmatrix} p_1(w) & \dots & p_n(w) \end{bmatrix}^\top$.

We can solve the system (7.1) iteratively using Newton-Raphson steps:

$$w^{(k+1)} = w^{(k)} - \left(\frac{\partial^2 \log \mathcal{L}}{\partial w \partial w^\top}\left(w^{(k)}\right)\right)^{-1} \frac{\partial \log \mathcal{L}}{\partial w}\left(w^{(k)}\right) \tag{7.2}$$

Let V be $n \times n$ diagonal matrix with $V_{i,i} = p_i(w)(1-p_i(w))$, it is straightforward to verify that the Hessian matrix has the following expression:

$$\frac{\partial^2 \log \mathcal{L}}{\partial w \partial w^\top}(w) = -X^\top V X.$$

Then (7.2) becomes:

$$\begin{aligned} w^{(k+1)} &= w^{(k)} + \left(X^\top V^{(k)} X\right)^{-1} X^\top (y - p^{(k)}) \\ &= \left(X^\top V^{(k)} X\right)^{-1} X^\top V^{(k)} \left(X w^{(k)} + {V^{(k)}}^{-1}(y-p^{(k)})\right) \\ &= \left(X^\top V^{(k)} X\right)^{-1} X^\top V^{(k)} z^{(k)}, \end{aligned} \tag{7.3}$$

where $z^{(k)} := Xw^{(k)} + {V^{(k)}}^{-1}(y - p^{(k)})$ and where $V^{(k)}$ and $p^{(k)}$ are V and p, respectively, evaluated at $w^{(k)}$.

Note that $w^{(k+1)}$ as given by (7.3) also satisfies

$$w^{(k+1)} = \arg\min_{w} \left(z^{(k)} - Xw\right)^{\top} V^{(k)} \left(z^{(k)} - Xw\right),$$

a weighted least-squares problem, and hense iterating (7.3) is often called iteratively reweighted least squares. It is easy to see that the Hessian matrix $X^{\top}VX$ is positive definite (it follows from the fact that $V > 0$), and thus is a convex optimization problem.

Note that convergence fails if the two classes are linearly separable. In that case we can handle this via a regularization.