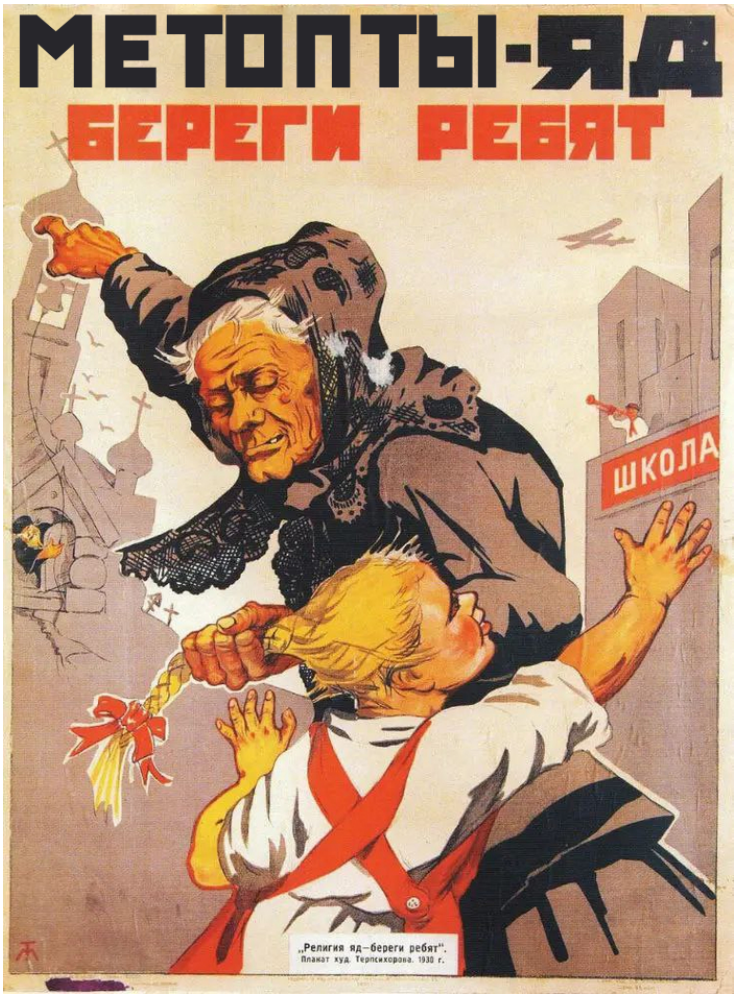
OPTIMIZATION METHODS ABSTRACT

CUB, AST MASTERS 1 курс
LECTOR: DMITRY KROPOTOV

AUTHORS:

Andrei K-dizzled Kozyrev, Ivan klimoza Klimov



SPRING 2024

Оглавление

1 Introduction

This course tries to answer such questions that occur while doing some practical tasks in machine learning such as:

- Which optimization routine to use in a given situation?
- How do I convert one optimization problem to another?
- How do I solve a given optimization problem?

Definition. Let's start with a simple *Unconstrained optimization problem*:

$$f(x) \to \min_x, x \in \mathbb{R}^n, f \in \mathcal{C}^1$$

It's quite clear on specific examples, that when introducing an optimization technique, we cannot garantee to achieve the global minimum/maximum. We can only hope to find a local minimum/maximum.

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} \in \mathbb{R}^n$$

$$\nabla^2 f(x) = \left\{ \frac{\partial^2 f}{\partial x_i \partial x_j} \right\}_{i,j=1}^n \in \mathbb{R}^{n \times n} (\in \mathcal{C}^2)$$

Theorem. *Necessary condition for a local minimum:* $f \in \mathcal{C}^1 (\in \mathcal{C}^2), x_0 \in \text{intdom } f$, x_0 — *local minimum* $\Longrightarrow \nabla f(x_0) = 0, \nabla^2 f(x_0) \geqslant 0$

Доказательство. $\exists \varepsilon > 0$ such that $\forall x \in O_\varepsilon(x_0) \qquad f(x) \geqslant f(x_0)$ (O_ε – vicinity ε)

$$f(x + \alpha d) = f(x) + \nabla f(x)^T d\alpha + \overline{o}(\alpha^2) \geqslant f(x_0)$$

$$\nabla f(x_0)^T d\alpha + \overline{o}(\alpha^2) \geqslant 0$$
$$\nabla f(x_0)^T d + \overline{o}(\alpha) \geqslant 0 \qquad | \ \alpha \to 0$$
$$\nabla f(x_0)^T d \geqslant 0 \quad \forall d \in \mathbb{R}^n \Longrightarrow \nabla f(x_0) = 0$$

■

Theorem. *Sufficient condition for a local minimum:* $f \in \mathcal{C}^2, x_0 \in \text{intdom } f, \nabla f(x_0) = 0, \nabla^2 f(x_0) > 0 \Longrightarrow x_0$ — *local minimum*

To solve the optimization problem one may use linear regression, defined like so:

$$Q(w) = \|Xw - y\|_2^2 \to \min_w \Longrightarrow w_{opt} = \underbrace{(X^T X)^{-1}}_{D \times D} X^T y$$

Why it could be bad? It is hard to store a matrix of D by D size in memory. The complexity wil be $\mathcal{O}(ND^2)$

1.1 Machine Precision

Floating points on machines are stored using the mantissa. A technical standard for floating-point arithmetic is the IEEE Standard for Floating-Point Arithmetic, also known as IEEE 754. Let us have a quick overview of the standard. In IEEE basic decimal types, known as floats are also stored using 32 bits. The first bit is a sign bit, the next 8 bits define the exponent E and the rest 23 bits are the mantissa M:

The value of the actual number that it represents is calculated as follows:

$$(-1)^{\text{Sign}} \cdot \left(1 + \frac{M}{2^{23}}\right) \cdot 2^{E-127}$$

So, whats the point here. If we look at how dense is the exponential scale, we see that the bigger are the numbers, the higher is the error, i.e. the distance betweeen 2 (∼neighbouring) given numbers on the exponential scale grows when these numbers get bigger.

$$\left|\frac{f(x) - x}{x}\right| \leqslant \varepsilon_m$$

Here ε_m is known as machine precision.

Some popular values:

- $\varepsilon_m \approx 10^{-7}$ for single precision if using 32 bits and $\varepsilon_m \approx 10^{-3}$ — 16 bits
- $\varepsilon_m \approx 10^{-16}$ for double precision — 64 bits

Say you have an oracle that that takes a point as input and computes the value of function f at this point and its gradient at this point. That's our optimization routine. How can we do a sanity check of our oracle optimizer?

We can check it this way:

$$f(x + \varepsilon d) + f(x) + \nabla f(x)^T d\varepsilon + \underline{O}(\varepsilon^2)$$
$$\nabla f(x)^T d = \frac{f(x + \varepsilon d) - f(x)}{\varepsilon} + \underline{O}(\varepsilon)$$
$$d = e_i \qquad \frac{\partial f}{\partial x_i} \approx \frac{f(x + \varepsilon e_i) - f(x)}{\varepsilon}$$

1.2 Convergence rate

One other important paramter of an optimization algorithm, used, for example for evaluating the performance of the algorithm, is the convergence rate.

Yet, sometimes, different algorithms have different convergence rates for different functions, so it is important to evaluate algorithms for a class of functions, not just for a single function.

Say we have a sequence of points $\{x_k\}$, and we want to measure the distance between the current point and the optimal point.

$$\begin{aligned} r_k &= f(x_k) - f_{opt} \\ &= \|x_k - x_{opt}\| \\ &= \|\nabla f(x_k)\| \end{aligned}$$

Definition. $\{r_k\}$ has linear convergence rate if $\exists X, q \in (0,1), k_0$ such that $\forall k \geqslant k_0 \quad r_k \leqslant Cq^k$

For a linear convergent algorithm, each step requires a constant number of iterations. So the quality of the algorithm really depends on this constant. Whether it is big or small.

$$\begin{aligned} r_k &= Cq^k \\ \log r_k &= \log C + k \log q \end{aligned}$$

Definition. $\{r_k\}$ has sublinear convergence rate if it doesn't have a linear convergence rate $\forall q \in (0,1)$.

Definition. $\{r_k\}$ has superlinear convergence rate if it has a linear convergence rate $\forall q \in (0,1)$

Notice: An example of a superlinear convergent algorithm is the Newton's method.

Notice: For a superlinear convergent algorithm, with each iteration we get a higher amount of bits, that we have already got from all previous iterations.

Theorem. *(Test of ratios) Lets consider a sequence $\{r_k\}$, $r_k > 0$ and:*

$$\alpha := \overline{\lim_{k\to\infty}} \frac{r_{k+1}}{r_k} \qquad \beta := \underline{\lim_{k\to\infty}} \frac{r_{k+1}}{r_k}$$

Then:

1. *If $\alpha < 1$, then $\{r_k\}$ has a linear convergence rate*
2. *If $\beta \geqslant 1$, then $\{r_k\}$ has a sublinear convergence rate*
3. *If $\beta < 1, \alpha > 1$, we cannot say anything*

Let's consider an example. A sequence is $\{r_k\} = \frac{1}{k}$. Then:

$$\frac{r_{k+1}}{r_k} = \frac{k}{k+1} \underbrace{\longrightarrow}_{k\to\infty} 1 \Longrightarrow \text{sublinear}$$

Theorem. *(Criterion of roots) Lets consider a sequence $\{r_k\} \underbrace{\longrightarrow}_{k\to\infty} 0$, $r_k \geqslant 0$ and:*

$$\alpha := \overline{\lim_{k\to\infty}} \sqrt[k]{r_k}$$

Then:

1. *If $\alpha < 1$, then $\{r_k\}$ has a linear convergence rate*
2. *If $\alpha = 0$, then $\{r_k\}$ has a superlinear convergence rate*
3. *If $\alpha = 1$, then $\{r_k\}$ has a sublinear convergence rate*

Definition. *Line search* We have $f(x) \to \min_x, x \in \mathbb{R}^n, f \in \mathcal{C}^1$:

$$x_{k+1} = x_k + \alpha_k d_k, d_k \in \mathbb{R}^n, \alpha_k \in \mathbb{R}_+$$
$$\text{Requirement for } d_k : \nabla f(x_k)^T d_k < 0$$

So:

$$\arg\min \underbrace{f(x_k + \alpha d_k)}_{\varphi(\alpha)} = \alpha_k$$

There are several conditions for searching for a local minimum of $\varphi(\alpha)$. These conditions give us ranges of α which we will consider while searching for the minimum afterwards.

One of them is the Armijo condition.

Definition. *Armijo condition*:

$$\varphi(\alpha) \leqslant \varphi(0) + c_1 \alpha \varphi'(0)$$
$$c_1 \in (0, 1)$$

Also there are weak and strong Wolfe conditions.

Definition. *Weak Wolfe condition*:

$$\varphi(\alpha) \geqslant c_2 \varphi'(0)$$
$$c_2 \in (c_1, 1)$$

Definition. *Strong Wolfe condition*:

$$|\varphi'(\alpha)| \leqslant c_2 |\varphi'(0)|$$
$$c_2 \in (c_1, 1)$$

Theorem. *If we have $\varphi \in \mathcal{C}^1, \varphi'(0) < 0, \varphi(\alpha) > -\infty \quad \forall \alpha \in [0, \alpha_0]$ and $0 < c_1 < c_2 < 1$, then α_k satisfies both the Armijo and the weak/strong Wolfe conditions.*

Доказательство. Let's consider the Armijo condition.

$$\varphi(\alpha) = \varphi(0) + \alpha\varphi'(0) + \overline{o}(\alpha) \leqslant \varphi(0) + c_1 \alpha \varphi'(0)$$
$$\overline{o}(\alpha) + \leqslant \alpha \varphi'(0)(c_1 - 1)$$

$$\psi(\alpha) = \varphi(0) + c_1\alpha\varphi'(0) - \varphi(\alpha)$$
$$\psi(0) = 0, \psi(\alpha_{\max}) = 0, \psi(\alpha) > 0 \quad \forall\alpha \in (0, \alpha_{\max})$$
$$\exists\hat{\alpha} \in (0, \alpha_{\max}) : \psi'(\hat{\alpha}) = 0$$
$$\psi'(\alpha) = c_1\varphi'(0) - \varphi'(\alpha) = 0$$
$$\varphi'(\alpha) = c_1\varphi'(0) \geqslant c_2\varphi'(0) \qquad \forall c_2 \geqslant c_1$$

■

And now let's discuss how do we find the particular values of α_k that satisfy the conditions. There are several methods for that.

Definition. *Backtracking algorithm*:

- $\alpha = \alpha_{start}$
- Repeat:
  - if $\varphi(\alpha) \leqslant \varphi(0) + c_1\alpha\varphi'(0)$, then return α
  - else $\alpha = \beta\alpha, \beta \in (0, 1)$

Notice: Say you use line_search from scipy. Sometimes it can return you None. What do you do? You can use backtracking algorithm as it never fails completely.

What are the stopping conditions for the expansion procedure?

- $\varphi(\alpha) > \varphi(0) + c_1\alpha\varphi'(0)$
- $\varphi'(\alpha) > 0$
- $\varphi(\alpha_{prev}) < \varphi(\alpha), \qquad c_1 < c_2, \varphi'(\alpha_{prev}) > c_1\varphi'(0)$

We do an expansion procedure and then the zooming procedure.

Definition. *Expansion procedure*:

- $\alpha_0 = 0, \alpha_1 \in (0, \alpha_{max})$
- Repeat:
  - Compute $\varphi(\alpha_i)$
  - If $\varphi(\alpha_i) > \varphi(0) + c_1\alpha_i\varphi'(0)$ or $\varphi(\alpha_i) > \varphi(\alpha_{i-1})$, then zoom$(\alpha_{i-1}, \alpha_i)$
  - Compute $\varphi'(\alpha_i)$
  - If $|\varphi'(\alpha_i)| \leqslant c_2|\varphi'(0)|$, then $\alpha_k = \alpha_i$ and stop
  - If $\varphi'(\alpha_i) \geqslant 0$, then zoom$(\alpha_{i-1}, \alpha_i)$
  - Pick $\alpha_{i+1} \in (\alpha_i, \alpha_{max})$
  - $i++$

On the zooming procedure we are doing aka binary search and reducing the interval of the search.

Definition. *Zoom procedure*: Depends on two arguments: $\alpha_{low}, \alpha_{high}$. Requirements for the arguments: α_{low} should satisfy the Armijo condition ans has the least function value we have seen so far. Also $\varphi'(\alpha_{low})(\alpha_{high} - \alpha_{low}) < 0$.

- Repeat:
  - Choose $\alpha \in (\alpha_{low}, \alpha_{high})$
  - Compute $\varphi(\alpha)$
  - If $\varphi(\alpha) > \varphi(0) + c_1\alpha\varphi'(0)$ or $\varphi(\alpha) > \varphi(\alpha_{low})$, then $\alpha_{high} = \alpha$
  - Compute $\varphi'(\alpha)$
  - If $|\varphi'(\alpha)| \leqslant c_2|\varphi'(0)|$, then $\alpha_k = \alpha$ and stop
  - If $\varphi'(\alpha)(\alpha_{high} - \alpha_{low}) \geqslant 0$, then $\alpha_{high} = \alpha_{low}$
  - $\alpha_{low} = \alpha$