

# Discover acceleration of gradient descent

Daniil Merkulov

Optimization methods. MIPT

## Coordinate shift

Consider the following quadratic optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}_{++}^d.$$

## Coordinate shift

Consider the following quadratic optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}_{++}^d.$$

- Firstly, without loss of generality we can set  $c = 0$ , which will not affect optimization process.



## Coordinate shift

Consider the following quadratic optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}_{++}^d.$$

- Firstly, without loss of generality we can set  $c = 0$ , which will not affect optimization process.
- Secondly, we have a spectral decomposition of the matrix  $A$ :

$$A = Q \Lambda Q^\top$$



## Coordinate shift

Consider the following quadratic optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}_{++}^d.$$

- Firstly, without loss of generality we can set  $c = 0$ , which will not affect optimization process.
- Secondly, we have a spectral decomposition of the matrix  $A$ :

$$A = Q \Lambda Q^\top$$

- Let's show, that we can switch coordinates in order to make an analysis a little bit easier. Let  $\hat{x} = Q^\top(x - x^*)$ , where  $x^*$  is the minimum point of initial function, defined by  $Ax^* = b$ . At the same time  $x = Q\hat{x} + x^*$ .

$$\begin{aligned} f(\hat{x}) &= \frac{1}{2} (Q\hat{x} + x^*)^\top A (Q\hat{x} + x^*) - b^\top (Q\hat{x} + x^*) \\ &= \frac{1}{2} \hat{x}^\top Q^\top A Q \hat{x} + (x^*)^\top A Q \hat{x} + \frac{1}{2} (x^*)^\top A (x^*) - b^\top Q \hat{x} - b^\top x^* \\ &= \frac{1}{2} \hat{x}^\top \Lambda \hat{x} \end{aligned}$$



# Polyak Heavy ball method

Let's introduce the idea of momentum, proposed by Polyak in 1964. Recall that the momentum update is

$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1}).$$



# Polyak Heavy ball method

Let's introduce the idea of momentum, proposed by Polyak in 1964. Recall that the momentum update is

$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1}).$$

Which is in our case is

$$\hat{x}_{k+1} = \hat{x}_k - \alpha \Lambda \hat{x}_k + \beta(\hat{x}_k - \hat{x}_{k-1}) = (I - \alpha \Lambda + \beta I) \hat{x}_k - \beta \hat{x}_{k-1}$$



# Polyak Heavy ball method



Let's introduce the idea of momentum, proposed by Polyak in 1964. Recall that the momentum update is

$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1}).$$

Which is in our case is

$$\hat{x}_{k+1} = \hat{x}_k - \alpha \Lambda \hat{x}_k + \beta(\hat{x}_k - \hat{x}_{k-1}) = (I - \alpha \Lambda + \beta I) \hat{x}_k - \beta \hat{x}_{k-1}$$

This can be rewritten as follows

$$\hat{x}_{k+1} = (I - \alpha \Lambda + \beta I) \hat{x}_k - \beta \hat{x}_{k-1},$$

$$\hat{x}_k = \hat{x}_k.$$

Let's use the following notation  $\hat{z}_k = \begin{bmatrix} \hat{x}_{k+1} \\ \hat{x}_k \end{bmatrix}$ . Therefore  $\hat{z}_{k+1} = M \hat{z}_k$ , where the iteration matrix  $M$  is:

$$M = \begin{bmatrix} I - \alpha \Lambda + \beta I & -\beta I \\ I & 0_d \end{bmatrix}.$$



## Reduction to a scalar case

Note, that  $M$  is  $2d \times 2d$  matrix with 4 block-diagonal matrices of size  $d \times d$  inside. It means, that we can rearrange the order of coordinates to make  $M$  block-diagonal in the following form. Note that in the equation below, the matrix  $M$  denotes the same as in the notation above, except for the described permutation of rows and columns. We use this slight abuse of notation for the sake of clarity.

## Reduction to a scalar case

Note, that  $M$  is  $2d \times 2d$  matrix with 4 block-diagonal matrices of size  $d \times d$  inside. It means, that we can rearrange the order of coordinates to make  $M$  block-diagonal in the following form. Note that in the equation below, the matrix  $M$  denotes the same as in the notation above, except for the described permutation of rows and columns. We use this slight abuse of notation for the sake of clarity.



where  $\hat{x}_k^{(i)}$  is  $i$ -th coordinate of vector  $\hat{x}_k \in \mathbb{R}^d$  and  $M_i$  stands for  $2 \times 2$  matrix. This rearrangement allows us to study the dynamics of the method independently for each dimension. One may observe, that the asymptotic convergence rate of the  $2d$ -dimensional vector sequence of  $\hat{z}_k$  is defined by the worst convergence rate among its block of coordinates. Thus, it is enough to study the optimization in a one-dimensional case.

## Reduction to a scalar case

For  $i$ -th coordinate with  $\lambda_i$  as an  $i$ -th eigenvalue of matrix  $W$  we have:

$$M_i = \begin{bmatrix} 1 - \alpha\lambda_i + \beta & -\beta \\ 1 & 0 \end{bmatrix}.$$

The method will be convergent if  $\rho(M) < 1$ , and the optimal parameters can be computed by optimizing the spectral radius

$$\alpha^*, \beta^* = \arg \min_{\alpha, \beta} \max_{\lambda \in [\mu, L]} \rho(M) \quad \alpha^* = \frac{4}{(\sqrt{L} + \sqrt{\mu})^2}; \quad \beta^* = \left( \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}} \right)^2.$$

It can be shown, that for such parameters the matrix  $M$  has complex eigenvalues, which forms a conjugate pair, so the distance to the optimum (in this case,  $\|z_k\|$ ), generally, will not go to zero monotonically.

## Heavy ball quadratic convergence

We can explicitly calculate the eigenvalues of  $M_i$ :

$$\lambda_1^M, \lambda_2^M = \lambda \left( \begin{bmatrix} 1 - \alpha\lambda_i + \beta & -\beta \\ 1 & 0 \end{bmatrix} \right) = \frac{1 + \beta - \alpha\lambda_i \pm \sqrt{(1 + \beta - \alpha\lambda_i)^2 - 4\beta}}{2}.$$

When  $\alpha$  and  $\beta$  are optimal  $(\alpha^*, \beta^*)$ , the eigenvalues are complex-conjugated pair  $(1 + \beta - \alpha\lambda_i)^2 - 4\beta \leq 0$ , i.e.  $\beta \geq (1 - \sqrt{\alpha\lambda_i})^2$ .

$$\operatorname{Re}(\lambda_1^M) = \frac{L + \mu - 2\lambda_i}{(\sqrt{L} + \sqrt{\mu})^2}; \quad \operatorname{Im}(\lambda_1^M) = \frac{\pm 2\sqrt{(L - \lambda_i)(\lambda_i - \mu)}}{(\sqrt{L} + \sqrt{\mu})^2}; \quad |\lambda_1^M| = \frac{L - \mu}{(\sqrt{L} + \sqrt{\mu})^2}.$$

And the convergence rate does not depend on the stepsize and equals to  $\sqrt{\beta^*}$ .

# Heavy ball method summary

- Ensures accelerated convergence for strongly convex quadratic problems

# Heavy ball method summary

- Ensures accelerated convergence for strongly convex quadratic problems
- Local accelerated convergence was proved in the original paper.

# Heavy ball method summary

- Ensures accelerated convergence for strongly convex quadratic problems
- Local accelerated convergence was proved in the original paper.
- Recently was proved, that there is no global accelerated convergence for the method.

# Heavy ball method summary

- Ensures accelerated convergence for strongly convex quadratic problems
- Local accelerated convergence was proved in the original paper.
- Recently was proved, that there is no global accelerated convergence for the method.
- Method was not extremely popular until the ML boom



# Heavy ball method summary

- Ensures accelerated convergence for strongly convex quadratic problems
- Local accelerated convergence was proved in the original paper.
- Recently was proved, that there is no global accelerated convergence for the method.
- Method was not extremely popular until the ML boom
- Nowadays, it is de-facto standard for practical acceleration of gradient methods, even for the non-convex problems (neural network training)

# Nesterov accelerated gradient

$$x_{k+1} = x_k - \alpha \nabla f(x_k) \quad (\text{GD})$$

$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1}) \quad (\text{HB})$$

$$\begin{cases} y_{k+1} = x_k + \beta(x_k - x_{k-1}) \\ x_{k+1} = y_{k+1} - \alpha \nabla f(y_{k+1}) \end{cases} \quad (\text{NAG})$$