Conjugate gradient method

Seminar

Optimization for ML. Faculty of Computer Science. HSE University

Strongly convex quadratics

Consider the following quadratic optimization problem:

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

Optimality conditions:

$$\nabla f(x^*) = Ax^* - b = 0 \iff Ax^* = b$$

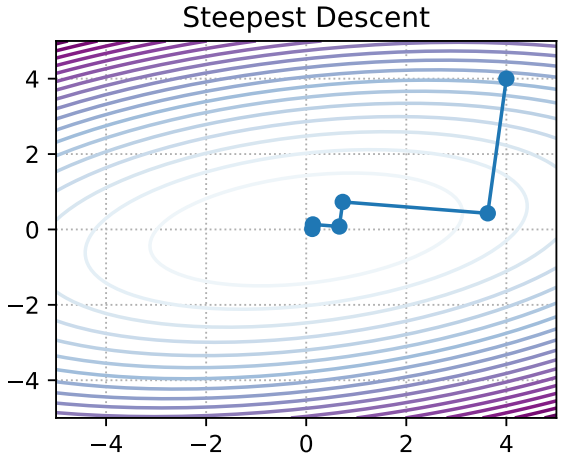
Strongly convex quadratics

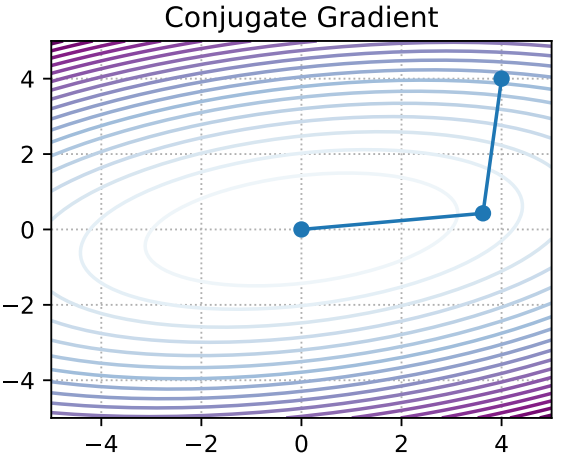
Consider the following quadratic optimization problem:

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

Optimality conditions:

$$\nabla f(x^*) = Ax^* - b = 0 \iff Ax^* = b$$





Overview of the CG method for the quadratic problem

1) **Initialization.** $k = 0$ and $x_k = x_0$, $d_k = d_0 = -\nabla f(x_0)$.

Overview of the CG method for the quadratic problem

1) **Initialization.** $k = 0$ and $x_k = x_0$, $d_k = d_0 = -\nabla f(x_0)$.

2) **Optimal Step Length.** By the procedure of *line search* we find the optimal length of step. This involves calculate α_k minimizing $f(x_k + \alpha_k d_k)$:

$$\alpha_k = -\frac{d_k^\top (Ax_k - b)}{d_k^\top A d_k}$$

Overview of the CG method for the quadratic problem

1) **Initialization.** $k = 0$ and $x_k = x_0$, $d_k = d_0 = -\nabla f(x_0)$.

2) **Optimal Step Length.** By the procedure of *line search* we find the optimal length of step. This involves calculate α_k minimizing $f(x_k + \alpha_k d_k)$:

$$\alpha_k = -\frac{d_k^\top (Ax_k - b)}{d_k^\top A d_k}$$

3) **Algorithm Iteration.** Update the position of x_k by moving in the direction d_k, with a step size α_k:

$$x_{k+1} = x_k + \alpha_k d_k$$

Overview of the CG method for the quadratic problem

1) **Initialization.** $k = 0$ and $x_k = x_0$, $d_k = d_0 = -\nabla f(x_0)$.

2) **Optimal Step Length.** By the procedure of *line search* we find the optimal length of step. This involves calculate α_k minimizing $f(x_k + \alpha_k d_k)$:

$$\alpha_k = -\frac{d_k^\top (Ax_k - b)}{d_k^\top A d_k}$$

3) **Algorithm Iteration.** Update the position of x_k by moving in the direction d_k, with a step size α_k:

$$x_{k+1} = x_k + \alpha_k d_k$$

4) **Direction Update.** Update the $d_{k+1} = -\nabla f(x_{k+1}) + \beta_k d_k$, where β_k is calculated by the formula:

$$\beta_k = \frac{\nabla f(x_{k+1})^\top A d_k}{d_k^\top A d_k}.$$

Overview of the CG method for the quadratic problem

1) **Initialization.** $k = 0$ and $x_k = x_0$, $d_k = d_0 = -\nabla f(x_0)$.

2) **Optimal Step Length.** By the procedure of *line search* we find the optimal length of step. This involves calculate α_k minimizing $f(x_k + \alpha_k d_k)$:

$$\alpha_k = -\frac{d_k^\top (Ax_k - b)}{d_k^\top A d_k}$$

3) **Algorithm Iteration.** Update the position of x_k by moving in the direction d_k, with a step size α_k:

$$x_{k+1} = x_k + \alpha_k d_k$$

4) **Direction Update.** Update the $d_{k+1} = -\nabla f(x_{k+1}) + \beta_k d_k$, where β_k is calculated by the formula:

$$\beta_k = \frac{\nabla f(x_{k+1})^\top A d_k}{d_k^\top A d_k}.$$

5) **Convergence Loop.** Repeat steps 2-4 until n directions are built, where n is the dimension of space (dimension of x).

Optimal Step Length

Exact line search:

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}^+} f(x_{k+1}) = \arg\min_{\alpha \in \mathbb{R}^+} f(x_k + \alpha d_k)$$

Optimal Step Length

Exact line search:

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}^+} f(x_{k+1}) = \arg\min_{\alpha \in \mathbb{R}^+} f(x_k + \alpha d_k)$$

Let's find an analytical expression for the step α_k:

$$f(x_k + \alpha d_k) = \frac{1}{2}(x_k + \alpha d_k)^\top A(x_k + \alpha d_k) - b^\top(x_k + \alpha d_k) + c$$
$$= \frac{1}{2}\alpha^2 d_k^\top A d_k + d_k^\top(Ax_k - b)\alpha + \left(\frac{1}{2}x_k^\top A x_k + x_k^\top d_k + c\right)$$

Optimal Step Length

Exact line search:

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}^+} f(x_{k+1}) = \arg\min_{\alpha \in \mathbb{R}^+} f(x_k + \alpha d_k)$$

Let's find an analytical expression for the step α_k:

$$f(x_k + \alpha d_k) = \frac{1}{2}(x_k + \alpha d_k)^\top A(x_k + \alpha d_k) - b^\top(x_k + \alpha d_k) + c$$
$$= \frac{1}{2}\alpha^2 d_k^\top A d_k + d_k^\top(Ax_k - b)\alpha + \left(\frac{1}{2}x_k^\top A x_k + x_k^\top d_k + c\right)$$

We consider $A \in \mathbb{S}^d_{++}$, so the point with zero derivative on this parabola is a minimum:

$$\left(d_k^\top A d_k\right)\alpha_k + d_k^\top(Ax_k - b) = 0 \iff \alpha_k = -\frac{d_k^\top(Ax_k - b)}{d_k^\top A d_k}$$

Direction Update

We update the direction in such a way that the next direction is A - orthogonal to the previous one:

$$d_{k+1} \perp_A d_k \iff d_{k+1}^\top A d_k = 0$$

Direction Update

We update the direction in such a way that the next direction is A - orthogonal to the previous one:

$$d_{k+1} \perp_A d_k \iff d_{k+1}^\top A d_k = 0$$

Since $d_{k+1} = -\nabla f(x_{k+1}) + \beta_k d_k$, we choose β_k so that there is A - orthogonality:

$$d_{k+1}^\top A d_k = -\nabla f\left(x_{k+1}\right)^\top A d_k + \beta_k d_k^\top A d_k = 0 \iff \beta_k = \frac{\nabla f\left(x_{k+1}\right)^\top A d_k}{d_k^\top A d_k}$$

Direction Update

We update the direction in such a way that the next direction is A - orthogonal to the previous one:

$$d_{k+1} \perp_A d_k \iff d_{k+1}^\top A d_k = 0$$

Since $d_{k+1} = -\nabla f(x_{k+1}) + \beta_k d_k$, we choose β_k so that there is A - orthogonality:

$$d_{k+1}^\top A d_k = -\nabla f\left(x_{k+1}\right)^\top A d_k + \beta_k d_k^\top A d_k = 0 \iff \beta_k = \frac{\nabla f\left(x_{k+1}\right)^\top A d_k}{d_k^\top A d_k}$$

Lemma 1

All directions of construction using the procedure described above are orthogonal to each other:

$$d_i^\top A d_j = 0, \text{ if } i \neq j$$

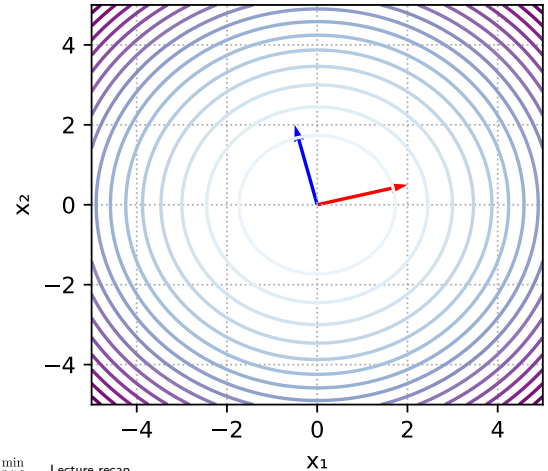
$$d_i^\top A d_j > 0, \text{ if } i = j$$

A-orthogonality

v_1 and v_2 are orthogonal

$v_1^T v_2 = 0.00$

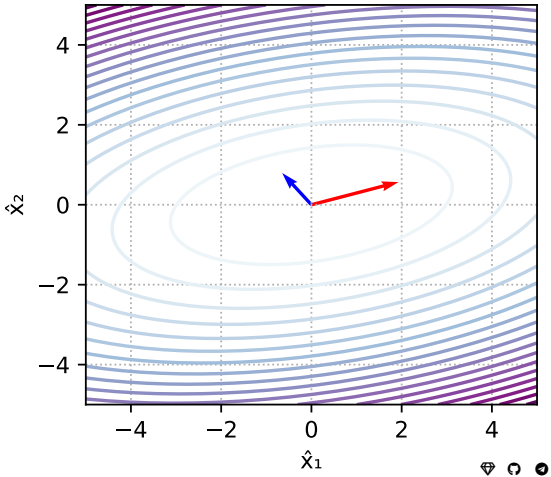
$v_1^T A v_2 = 1.19$



$\hat{v}_1$ and $\hat{v}_2$ are A-orthogonal

$\hat{v}_1^T \hat{v}_2 = -0.80$

$\hat{v}_1^T A \hat{v}_2 = -0.00$



Convergence of the CG method

Lemma 2

Suppose, we solve n-dimensional quadratic convex optimization problem. The conjugate directions method:

$$x_{k+1} = x_0 + \sum_{i=0}^{k} \alpha_i d_i,$$

where $\alpha_i = -\dfrac{d_i^\top (Ax_i - b)}{d_i^\top A d_i}$ taken from the line search, converges for at most n steps of the algorithm.

CG method in practice

In practice, the following formulas are usually used for the step α_k and the coefficient β_k:

$$\alpha_k = \frac{r_k^\top r_k}{d_k^\top A d_k} \qquad \beta_k = \frac{r_{k+1}^\top r_{k+1}}{r_k^\top r_k},$$

where $r_k = b - Ax_k$, since $x_{k+1} = x_k + \alpha_k d_k$ then $r_{k+1} = r_k - \alpha_k A d_k$. Also, $r_i^T r_k = 0, \forall i \neq k$ (**Lemma 5** from the lecture).

CG method in practice

In practice, the following formulas are usually used for the step α_k and the coefficient β_k:

$$\alpha_k = \frac{r_k^\top r_k}{d_k^\top A d_k} \qquad \beta_k = \frac{r_{k+1}^\top r_{k+1}}{r_k^\top r_k},$$

where $r_k = b - Ax_k$, since $x_{k+1} = x_k + \alpha_k d_k$ then $r_{k+1} = r_k - \alpha_k A d_k$. Also, $r_i^T r_k = 0, \forall i \neq k$ (**Lemma 5** from the lecture).

Let's get an expression for β_k:

$$\beta_k = \frac{\nabla f\left(x_{k+1}\right)^\top A d_k}{d_k^\top A d_k} = -\frac{r_{k+1}^\top A d_k}{d_k^\top A d_k}$$

CG method in practice

In practice, the following formulas are usually used for the step α_k and the coefficient β_k:

$$\alpha_k = \frac{r_k^\top r_k}{d_k^\top A d_k} \qquad \beta_k = \frac{r_{k+1}^\top r_{k+1}}{r_k^\top r_k},$$

where $r_k = b - Ax_k$, since $x_{k+1} = x_k + \alpha_k d_k$ then $r_{k+1} = r_k - \alpha_k A d_k$. Also, $r_i^T r_k = 0, \forall i \neq k$ (**Lemma 5** from the lecture).

Let's get an expression for β_k:

$$\beta_k = \frac{\nabla f\left(x_{k+1}\right)^\top A d_k}{d_k^\top A d_k} = -\frac{r_{k+1}^\top A d_k}{d_k^\top A d_k}$$

Numerator: $r_{k+1}^\top A d_k = \frac{1}{\alpha_k} r_{k+1}^\top \left(r_k - r_{k+1}\right) = [r_{k+1}^\top r_k = 0] = -\frac{1}{\alpha_k} r_{k+1}^\top r_{k+1}$

Denominator: $d_k^\top A d_k = \left(r_k + \beta_{k-1} d_{k-1}\right)^\top A d_k = \frac{1}{\alpha_k} r_k^\top \left(r_k - r_{k+1}\right) = \frac{1}{\alpha_k} r_k^\top r_k$

CG method in practice

In practice, the following formulas are usually used for the step α_k and the coefficient β_k:

$$\alpha_k = \frac{r_k^\top r_k}{d_k^\top A d_k} \qquad \beta_k = \frac{r_{k+1}^\top r_{k+1}}{r_k^\top r_k},$$

where $r_k = b - Ax_k$, since $x_{k+1} = x_k + \alpha_k d_k$ then $r_{k+1} = r_k - \alpha_k A d_k$. Also, $r_i^T r_k = 0, \forall i \neq k$ (**Lemma 5** from the lecture).

Let's get an expression for β_k:

$$\beta_k = \frac{\nabla f\left(x_{k+1}\right)^\top A d_k}{d_k^\top A d_k} = -\frac{r_{k+1}^\top A d_k}{d_k^\top A d_k}$$

Numerator: $r_{k+1}^\top A d_k = \frac{1}{\alpha_k} r_{k+1}^\top \left(r_k - r_{k+1}\right) = [r_{k+1}^\top r_k = 0] = -\frac{1}{\alpha_k} r_{k+1}^\top r_{k+1}$

Denominator: $d_k^\top A d_k = \left(r_k + \beta_{k-1} d_{k-1}\right)^\top A d_k = \frac{1}{\alpha_k} r_k^\top \left(r_k - r_{k+1}\right) = \frac{1}{\alpha_k} r_k^\top r_k$

Question

Why is this modification better than the standard version?

CG method in practice. Pseudocode

$\mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$

if $\mathbf{r}_0$ is sufficiently small, then return $\mathbf{x}_0$ as the result

$\mathbf{d}_0 := \mathbf{r}_0$

$k := 0$

repeat

$\alpha_k := \dfrac{\mathbf{r}_k^\mathsf{T}\mathbf{r}_k}{\mathbf{d}_k^\mathsf{T}\mathbf{A}\mathbf{d}_k}$

$\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k\mathbf{d}_k$

$\mathbf{r}_{k+1} := \mathbf{r}_k - \alpha_k\mathbf{A}\mathbf{d}_k$

if $\mathbf{r}_{k+1}$ is sufficiently small, then exit loop

$\beta_k := \dfrac{\mathbf{r}_{k+1}^\mathsf{T}\mathbf{r}_{k+1}}{\mathbf{r}_k^\mathsf{T}\mathbf{r}_k}$

$\mathbf{d}_{k+1} := \mathbf{r}_{k+1} + \beta_k\mathbf{d}_k$

$k := k + 1$

end repeat

return $\mathbf{x}_{k+1}$ as the result

Exercise 1

Write iterations of the conjugate gradient method for a quadratic problem

$$f(x) = \frac{1}{2}x^T Ax - b^T x \longrightarrow \min_{x \in \mathbb{R}^n}$$

and run experiments for several matrices A. See code here .

Non-linear conjugate gradient method

In case we do not have an analytic expression for a function or its gradient, we will most likely not be able to solve the one-dimensional minimization problem analytically. Therefore, step 2 of the algorithm is replaced by the usual line search procedure. But there is the following mathematical trick for the fourth point:

For two iterations, it is fair:

$$x_{k+1} - x_k = cd_k,$$

where c is some kind of constant. Then for the quadratic case, we have:

$$\nabla f(x_{k+1}) - \nabla f(x_k) = (Ax_{k+1} - b) - (Ax_k - b) = A(x_{k+1} - x_k) = cAd_k$$

Expressing from this equation the work $Ad_k = \frac{1}{c}\left(\nabla f(x_{k+1}) - \nabla f(x_k)\right)$, we get rid of the "knowledge" of the function in step definition β_k, then point 4 will be rewritten as:

$$\beta_k = \frac{\nabla f(x_{k+1})^\top (\nabla f(x_{k+1}) - \nabla f(x_k))}{d_k^\top (\nabla f(x_{k+1}) - \nabla f(x_k))}.$$

This method is called the Polak-Ribier method.

Exercise 2

Write iterations of the Polack-Ribier method and run experiments for several μ in binary logistic regression:

$$f(x) = \frac{\mu}{2}\|x\|_2^2 + \frac{1}{m}\sum_{i=1}^{m} log(1 + exp(-y_i\langle a_i, x\rangle)) \longrightarrow \min_{x \in \mathbb{R}^n}$$

See code here .

A pathological example

Let $t \in (0, 1)$ and

$$W = \begin{bmatrix} t & \sqrt{t} & & & \\ \sqrt{t} & 1+t & \sqrt{t} & & \\ & \sqrt{t} & 1+t & \sqrt{t} & \\ & & \ddots & \ddots & \ddots \\ & & & \sqrt{t} & 1+t \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Since W invertible, there exists a unique solution to $Wx = b$. Solving it by conjugate gradient descent gives us rather bad convergence. During the CG process, the error grows exponentially (!), until it suddenly becomes zero as the unique solution is found.

Residual $\|Wx_k - b\|^2$ grows exponentially as $(1/t)^k$ until the n iteration, after which it drops sharply to zero.

See experiment here .

Another computational experiments

Let's see another examples here . The code taken from .