Homework 3

Ilia Kamyshev

February 24, 2021

Contents

1 Convexity preservation

Given

$$h(x) = \max\{f(x), g(x)\},\ f(x) \text{ and } g(x) \text{ are convex}$$

Function $h(x)$ is convex iff folowing holds $h(\lambda x_0 + (1-\lambda)x_1) \leq \lambda h(x_0) + (1-\lambda)h(x_1)$ for $\lambda \in [0, 1]$. It is clear that if $\max\{f(\lambda x_0 + (1-\lambda)x_1), g(\lambda x_0 + (1-\lambda)x_1)\} = f(\lambda x_0 + (1-\lambda)f(x_1)$, then from convexity of f we will have:

$$f(\lambda x_0 + (1-\lambda)x_1) \leq \lambda f(x_0) + (1-\lambda)f(x_1)$$

This is true for g too:

$$g(\lambda x_0 + (1-\lambda)x_1) \leq \lambda g(x_0) + (1-\lambda)g(x_1)$$

Thus, applying pointwise max function:

$$\max\{f(\lambda x_0 + (1-\lambda)x_1), g(\lambda x_0 + (1-\lambda)x_1)\} \leq \lambda \max\{f(x_0), g(x_0)\} + (1-\lambda)\max\{f(x_1), g(x_1)\}$$

In terms of $h(x) = \max\{f(x), g(x)\}$:

$$h(\lambda x_0 + (1-\lambda)x_1) \leq \lambda h(x_0) + (1-\lambda)h(x_1)\blacksquare$$

Another interesting proof came from epigraph of a function (set of points above its graph + graph). It is known that any function can be reconstructed from its epigraph. So if f is convex, then epi f is also convex. Same for g. Pointwise max in terms of sets is their intersection, so epi h = epi $f \cap$ epi g. As

it's known from set theory, intersection of convex sets is also convex set (*proof:* cross a line between two points in epi f and all the points between will be below this line since the set is convex, in the intersection set same points belong to epi f and to epi g, then convexity holds also for this set■).

2 One dimensional problem

Given

$$\min_{z\in\mathbf{R}} \frac{1}{m}\sum_{i}^{m}(z-x_i)^2 = \min_{z\in\mathbf{R}} f(z)$$

The neccessary condition is $\nabla_z f(z)|_{z^*} = 0$:

$$\frac{2}{m}\sum_{i}^{m}(z^*-x_i) = \frac{2}{m}mz^* - \frac{2}{m}\sum_{i}^{m}x_i = 0$$

$$z^* - \frac{1}{m}\sum_{i}^{m}x_i = 0 \implies z^* = \frac{1}{m}\sum_{i}^{m}x_i$$

The sufficient condition of minima $\nabla^2 f(z) = 1 - 0 > 0$ holds. Thus,

$$f(z^*) = \frac{1}{m}\sum_{i}^{m}(\frac{1}{m}\sum_{j}^{m}x_j - x_i) = \frac{m}{m^2}\sum_{j}^{m}x_j - \frac{1}{m}x_i = \frac{1}{m}\sum_{j}^{m}x_j - \frac{1}{m}\sum_{i}^{m}x_i$$

Since j and i loop over same values:

$$f(z^*) = \frac{1}{m}\sum_{j}^{m}x_j - \frac{1}{m}\sum_{i}^{m}x_i = 0$$

Answer: $z^* = \frac{1}{m}\sum_{i}^{m}x_i$, $f(z^*) = 0$

3 Linear regression

Given

$$\min_{a,b} \frac{1}{m}\sum_{i}^{m}(ax_i + b - y_i)^2 = \min_{a,b} f(a,b)$$

So $\nabla_b f(a,b) = 0$ and $\nabla_a f(a,b) = 0$:

$$\nabla_b f(a,b) = \frac{2}{m}\sum_{i}^{m}(ax_i + b - y_i) = 0$$

$$\nabla_a f(a,b) = \frac{2}{m}\sum_{i}^{m}(ax_i + b - y_i)x_i = 0$$

Equation for b:

$$\nabla_b f(a,b) = \frac{2}{m}\sum_{i}^{m}(ax_i + b - y_i) = 0$$

$$b = \sum_{i}^{m}(y_i - ax_i)$$

Substite b to $\nabla_a f(a,b) = 0$:

$$\sum_i^m (ax_i + \sum_j^m (y_j - ax_j) - y_i)x_i = 0$$

Let $\frac{1}{m}\sum_j^m y_j = \mathbb{E}[y]$ and $\frac{1}{m}\sum_j^m x_j = \mathbb{E}[x]$:

$$\sum_i^m (ax_i + m(\mathbb{E}[y] - a\mathbb{E}[x]) - y_i)x_i = 0$$

$$\sum_i^m (a(x_i^2 - x_i\mathbb{E}[x]) + mx_i\mathbb{E}[y]) = \sum_i^m y_i x_i$$

$$a\sum_i^m (x_i^2 - x_i\mathbb{E}[x]) + m\sum_i^m (x_i\mathbb{E}[y]) = \sum_i^m y_i x_i$$

$$a = \frac{\sum_i^m y_i x_i - m\sum_i^m (x_i\mathbb{E}[y])}{\sum_i^m (x_i^2 - x_i\mathbb{E}[x])}$$

Dividing numerator and denominator by m:

$$a = \frac{\frac{1}{m}\sum_i^m y_i x_i - m\mathbb{E}[x]\mathbb{E}[y]}{\frac{1}{m}\sum_i^m (x_i^2 - x_i\mathbb{E}[x])}$$

Intuively, the numerator can be rewritten as $\text{cov}(x,y) = \frac{1}{m}\sum_i^m (x_i - \mathbb{E}[x])(y_i - \mathbb{E}[y])$, and the denominator as $\text{var}(x) = \frac{1}{m}\sum_i^m (x_i - \mathbb{E}[x])^2$. Expanding $\text{cov}(x,y)$:

$$\frac{1}{m}\sum_i^m (x_i y_i - x_i\mathbb{E}[y] - y_i\mathbb{E}[x] + \mathbb{E}[x]\mathbb{E}[y])$$

Thus, numerator can be rewritten as:

$$\frac{1}{m}\sum_i^m (y_i x_i - x_i\mathbb{E}[y] - y_i\mathbb{E}[x] + \mathbb{E}[x]\mathbb{E}[y] + x_i\mathbb{E}[y] + y_i\mathbb{E}[x] - 2\mathbb{E}[x]\mathbb{E}[y]) =$$

$$= \frac{1}{m}\sum_i^m (x_i - \mathbb{E}[x])(y_i - \mathbb{E}[y]) + \frac{1}{m}\sum_i^m (x_i\mathbb{E}[y] + y_i\mathbb{E}[x] - 2\mathbb{E}[x]\mathbb{E}[y]) =$$

$$= \text{cov}(x,y) + \mathbb{E}[x]\mathbb{E}[y] + \mathbb{E}[y]\mathbb{E}[x] - 2\mathbb{E}[x]\mathbb{E}[y] = \text{cov}(x,y) + 2\mathbb{E}[x]\mathbb{E}[y] - 2\mathbb{E}[x]\mathbb{E}[y] = \text{cov}(x,y)$$

Similarly we can rewrite denominator:

$$\frac{1}{m}\sum_i^m (x_i^2 - x_i\mathbb{E}[x]) = \frac{1}{m}\sum_i^m (x_i^2 - 2x_i\mathbb{E}[x] + \mathbb{E}[x]^2) + \frac{1}{m}\sum_i^m (x_i\mathbb{E}[x] - \mathbb{E}[x]^2) =$$

$$= \text{var}(x) + \mathbb{E}[x]\mathbb{E}[x] - \mathbb{E}[x]^2 = \text{var}(x)$$

Thus,

$$a = \frac{\text{cov}(x,y)}{\text{var}(x)}$$

And finally b:

$$b = m(\mathbb{E}[y] - \frac{\text{cov}(x,y)}{\text{var}(x)}\mathbb{E}[x])$$

4 Armijo condition

Given

$$\min\ 2x_1^4 + 3x_2^4 + 2x_1^2 + 4x_2^2 + x_1x_2 - 3x_1 - 2x_2\ x \in \mathbb{R}^2$$

Gradient method with Armijo condition is described in Algorithm 1.

Algorithm 1: Steepest-descend with inexact line search under Armijo condition

$x(0) = [0, 0]^T$ - initial guess;
f - objective function;
L^0 - initial norm of the gradient;
$\epsilon = 10^{-3}$ - tolerance;
$\hat{t} \leftarrow 1$ - initial step-size for inexact line search algorihm;
$\gamma \leftarrow 0.9$ - step-size decay;
$\alpha \leftarrow 0.1$;
while $L(k) > \epsilon$ **do**
  $t \leftarrow \hat{t}$;
  $p(k) \leftarrow -\nabla f(x(k))$;
  while $f(x(k) + tp(k)) > f(x(k)) + \alpha t(\nabla f(x(k)), p(k))$ **do**
    $t \leftarrow \gamma t$;
  end
  $x(k+1) \leftarrow x(k) - t\nabla f(x(k))$;
  $L(k+1) \leftarrow ||\nabla f(x(k+1))||$;
end
Result: $x(k+1)$

The gradient of given function is:

$$\nabla f(\mathbf{x}) = \begin{bmatrix} 8x_1^3 + 4x_1 + x_2 - 3 \\ 12x_2^3 + 8x_2 + x_1 - 2 \end{bmatrix}$$

This algorithm converged in 28 iterations under parameters given. The minimal value $f(x^*) = -1.0139$ and the $x^* = [0.4815, 0.1809]^T$. Results are visualized at Figure 1.

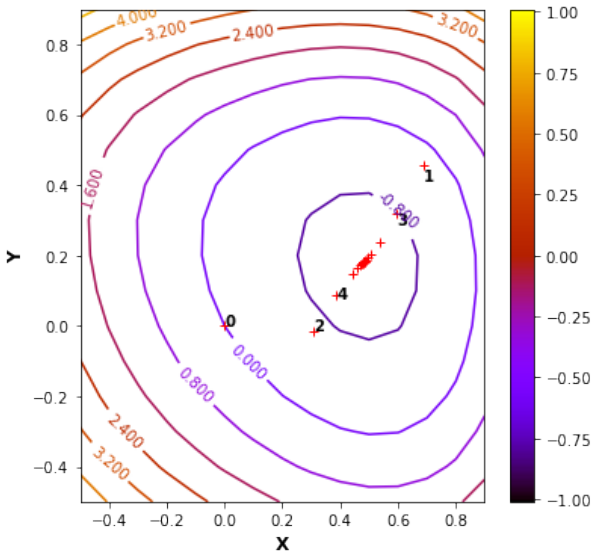

Figure 1: Iterative solution of the Algorithm 1

5 More algorithms is all you need

Given

$$f(\mathbf{x}) = \mathbf{x}^T\mathbf{A}\mathbf{x},\ \mathbf{A} \succ\ 0 \text{ and symmetric}$$

$$a_{ij} \in \mathbf{A} = \frac{1}{i+j-1},\ i = 1..5,\ j = 1..5$$

Initial parameters:

$$\nabla f(\mathbf{x}) = 2\mathbf{A}\mathbf{x}$$

$$\text{Initial guess } x(0) = [1, 1, 2, 3, 5]^T$$

$$\text{Tolerance } \epsilon = 10^{-4}$$

$$\text{Stop criteria } ||\nabla f(x(k))||$$

5.1 Gradient descend with constant step-size $t = \frac{1}{L}$

From Lipschitz continuity $||\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})|| = ||2\mathbf{A}\mathbf{x} - 2\mathbf{A}\mathbf{y}|| \leq L||\mathbf{x} - \mathbf{y}||$ we may conclude that $||2\mathbf{A}(\mathbf{x} - \mathbf{y})|| \leq 2||\mathbf{A}|| \cdot ||\mathbf{x} - \mathbf{y}|| \leq L||\mathbf{x} - \mathbf{y}|| \implies L = 2||\mathbf{A}||$. Thus,

$$t(k) = \frac{1}{2||\mathbf{A}||}$$

This algorithm converged in 9952 iterations with output $x^* = [-0.0048, 0.0156, 0.1297, -0.3998, 0.2685]$ and $f(x^*) = 0.0$.

5.2 Gradient descend with backtracking line search (Armijo condition)

This algorithm 1 was already listed before. Here $\gamma = 0.9$ and $\alpha = 0.3$ were set. This algorithm converged in 6161 iterations with output $x^* = [-0.0019, -0.0105, 0.1692, -0.3838, 0.2335]^T$ and $f(x^*) = 0.0$.

5.3 Steepest descend with exact line search

From the updating rule $x(k+1) = x(k) - t(k)\nabla f(x(k))$ we may find optimal t such that $t = \min_{t(k)} f(x(k) - t(k)\nabla f(x(k))$:

$$\nabla_{t(k)} f(x(k+1)) = (x(k) - t\nabla f(x(k)))^T A(x(k) - t\nabla f(x(k))) = 0$$

$$\nabla_{t(k)}((x(k)^T A - t\nabla f(x(k))^T A)(x(k) - t\nabla f(x(k)))) = 0$$

$$\nabla_{t(k)}(x(k)^T Ax(k) - tx(k)^T A\nabla f(x(k)) - t\nabla f(x(k))^T Ax(k) + t^2\nabla f(x(k))^T A\nabla f(x(k))) = 0$$

$$0 - x(k)^T A\nabla f(x(k)) - \nabla f(x(k))^T Ax(k) + 2t\nabla f(x(k))^T A\nabla f(x(k)) = 0$$

$$t = \frac{1}{2}\frac{x(k)^T A\nabla f(x(k)) + \nabla f(x(k))^T Ax(k)}{\nabla f(x(k))^T A\nabla f(x(k))}$$

$$t = \frac{1}{2}\frac{(Ax(k), \nabla f(x(k))) + (A\nabla f(x(k)), x(k))}{(A\nabla f(x(k)), \nabla f(x(k)))}$$

$$t = \frac{(Ax(k), \nabla f(x(k)))}{(A\nabla f(x(k)), \nabla f(x(k)))} = \frac{1}{2}\frac{||\nabla f(x(k))||^2}{(A\nabla f(x(k)), \nabla f(x(k)))}$$

This algorithm converged in 2338 iterations with output $x^* = [-0.0042, 0.0108, 0.1371, -0.3969, 0.262]^T$ and $f(x^*) = 0.0$.

5.4 Conjugate gradient method

The general derivation of this method is presented in many papers for solving $Ax = b$ [One-Minute Derivation of The Conjugate Gradient Algorithm], in our case since $\nabla f(x) = 2Ax$ this equation turns out to be $Ax = b/2 = 0$. Here the updating rule is $x(k+1) = x(k) + t(k)p(k)$. Where $p(k+1) = r(k) + \beta(k)p$ (where $\beta(k) = \frac{||r(k+1)||^2}{||r(k)||^2}$) and $r(k+1) = r(k) - t(k)Ap(k)$. Note that to account term 2 we have two choices: 1) divide initial $r(0)$ by $2 \to r(0) = -2Ax/2 = -Ax$; 2) divide $t(k)$ by 2 and multiply (A, p) by 2 in equation for $r(k+1)$. In order to preserve original structure of the algorithm we will choose 1st choice. From conjugancy we assume that $r(k)^T r(k+1) = 0$, thus:

$$r(k)^T r(k+1) = 0 = r(k)^T r(k) - t(k) r(k)^T Ap(k)$$

$$t = \frac{r(k)^T r(k)}{r(k)^T Ap(k)} = \frac{||r(k)||^2}{(Ap(k), r(k))}$$

This algorithm converged in 4 iterations with output $x^* = [0.0029, -0.0556, 0.241, -0.3653, 0.1791]^T$ and $f(x^*) = 0.0$.

5.5 Heavy-Ball method

Here the updating rule is as follows: $x(k+1) = x(k) - t(k)\nabla f(x(k)) + \beta(k)(x(k) - x(k-1))$ with momentum term $\beta(k)(x(k) - x(k-1))$. Here some restrictions $0 < t(k) < \frac{2(1+\beta(k))}{L}$, $0 \leq \beta(k) < 1$. More precisely, $\beta(k)$ and α from $t(k) = \alpha\frac{2(1+\beta(k))}{L}$ are tunable. After gridsearch $\beta(k) = 0.9217$ and $t(k) = 0.9986$ ($\alpha = 0.8143$). This algorithm converged in 251 iterations with output $x^* = [-0.0037, 0.0072, 0.143, -0.3954, 0.2578]^T$ and $f(x^*) = 0.0$.

5.6 Nesterov Fast Gradient method

Here $x(k+1) = y - t(k)\nabla f(y)$, where point $y = (1 - \theta(k))x(k) + \theta(k)v(k)$ with $v(k+1) = x(k) + \frac{1}{\theta(k)}(x(k+1) - x(k))$. $\theta(k)$ is chosen such that $\frac{1-\theta(k)}{\theta(k)^2}t(k) \leq \frac{t(k-1)}{\theta(k-1)^2}$. Idea of an algorithm was taken from this source. Algorithm implemented for $t(k) = \frac{1}{L}$. This algorithm converged in 1381 iterations with output $x^* = [-0.0048, 0.0156, 0.1298, -0.3998, 0.2684]^T$ and $f(x^*) = 0.0$.

5.7 Summary

The notebook is available on github:

https://github.com/arx7ti/optimization-course/blob/main/Homework3.ipynb.

From Figure 2 we may conclude that such methods as Conjugate Gradients, Heavy-Ball and Nesterov fast gradient have better performances than the others. Quick convergance of Heavy-Ball algorihm is due to the momentum term which scales down the previous step, that helps in control of oscillations. The success of the conjugate gradient is that it decomposes search direction into orthogonal components, this possible for convex problems, thus, quick convergancy is guaranteed, whereas not neccessarly for non-convex problems. As for Nesterov method this is second-order method, thus it converges in $\mathcal{O}(\frac{1}{k^2})$.

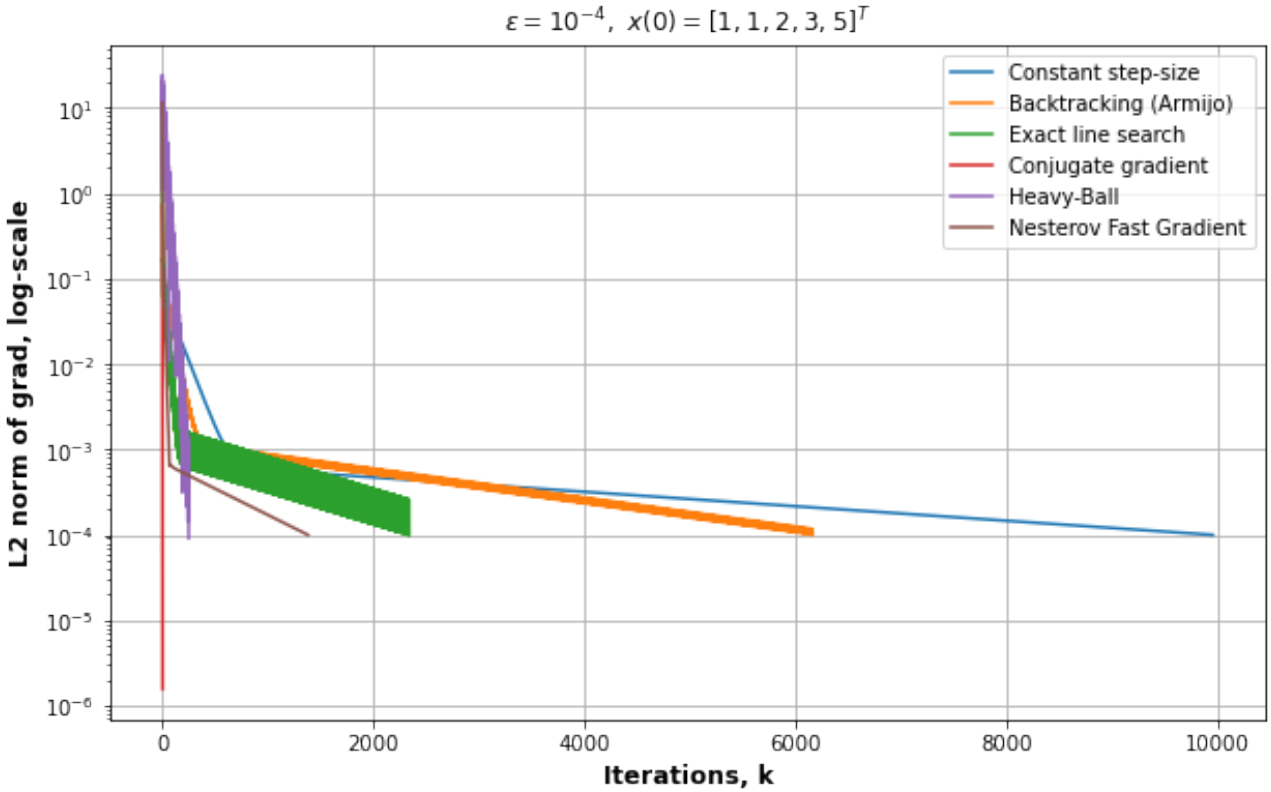

Figure 2: Convergence curves for each method