Homework 2

Ilia Kamyshev

February 19, 2021

Contents

1 Minimum

Given

$$f(x) = ax^2 + bx + c$$

This is a convex function, so the optimal solution is global and unique:

$$\frac{d}{dx} f(x) = 2ax + b = 0$$

$$x^* = -\frac{b}{2a}$$

The optimal value of $f(x)$ is as follows:

$$f(x^*) = \frac{b^2}{4a} - \frac{b}{a} + c = \frac{b}{a}(\frac{b}{4} - 1) + c$$

For the minimum following holds $\frac{d^2}{dx^2} f(x) > 0$:

$$\frac{d^2}{dx^2} f(x) = 2a > 0 \implies a > 0$$

Answer: the minimum is at $x^* = -\frac{b}{2a}$ and its value is $f(x^*) = \frac{b}{a}(\frac{b}{4} - 1) + c$ for $a > 0$, $b \in \mathbb{R}$ and $c \in \mathbb{R}$.

2 Gradient dimension

Given

$$h(x) = f(Ax),\ f : \mathbb{R}^m \to \mathbb{R},\ A \in \mathbb{R}^{m \times k}$$

Let's assign $y = Ax$ then $h(x) = (f \circ y)(x)$. The total derivative $Dh(\mathbf{x}) = Df(y(\mathbf{x}))Dy(\mathbf{x})$ (*matrix product*). The gradient is $\nabla(\circ) = (D(\circ))^T$. Thus,

$$Dh(\mathbf{x}) = Df(y)\mathbf{A}$$

$$\nabla_{\mathbf{x}} h(\mathbf{x}) = \mathbf{A}^T (Df(y))^T = \mathbf{A}^T \nabla_y f(y)$$

Since $f : \mathbb{R}^m \to \mathbb{R}$, then the dimension of $\nabla_y f(y)$ is $m \times 1$ (*denominator-layout*, the gradient is a column vector), and from formula above we can conclude that $(k \times m) \times (m \times 1) = k \times 1$

Answer: $k \times 1$

3 Gradient and Hessian

Given

$$f(x) = (x, c)^2,\ x \in \mathbb{R}^m$$

The inner product $(x, c)^2$ can be rewritten as $(x^T c)^2$. Let's assign $y = x^T c$, $g = y^2$ then $f(x) = g(y(x))$ or $f(x) = (g \circ y)(x)$. Thus, applying same technique as before:

1. $Df(x) = Dg(y(x))Dy(x)$. Here $Dg(y(x)) = D(y^2(x)) = 2y(x) = 2x^T c$ and $Dy(x) = c^T$. Therefore, $Df(x) = 2(x^T c)c^T$. The gradient $\nabla_x f(x) = (Df(x))^T = 2((x^T c)c^T)^T = 2c(c^T x)$.
2. The Hessian is $D(Df(x)) = D(2(x^T c)c^T) = 2cc^T$

   Answer: a) $2c(c^T x)$ b) $2cc^T$

4 Hessian matrix

Given

$$f(x) = g(Ax + b),\ g : \mathbb{R}^m \to \mathbb{R},\ \mathbb{A} \in \mathbb{R}^{m \times n},\ b \in \mathbb{R}^m,\ x \in \mathbb{R}^n$$

Let's assign $y = Ax + b$, then $f(x) = (g \circ y)(x)$. The total derivative is $Df(x) = Dg(y(x))Dy(x) = Dg(y(x))A$. The Hessian: $H_x(f(x)) = D(Df(x)) = D(Dg(y(x))A) = A^T D(Dg(y(x))) = A^T D^2 g(y) A = A^T H_y(g(y))A$.

Answer: $A^T H_y(g(y))A$

5 Optimal step-size problem

Given

$$f(\gamma) = (A(x+\gamma d), x+\gamma d) + (b, x+\gamma d), A \succ 0 \in \mathbb{R}^{n\times n},\ \ x, b, d \in \mathbb{R}^n$$

Since the function is quadratic and convex, there is a neccessary condition of local minima $f(\gamma) = 0$, in our case this solution will be also minimum (convexity and positive-definiteness of A). Let's rewrite:

$$f(\gamma) = (x+\gamma d)^T A^T (x+\gamma d) + b^T(x+\gamma d)$$

$$f(\gamma) = (x+\gamma d)^T Ax + (x+\gamma d)^T A\gamma d + b^T x + b^T\gamma d)$$

$$f(\gamma) = x^T Ax + \gamma d^T Ax + x^T A\gamma d + \gamma d^T A\gamma d + b^T x + b^T\gamma d$$

$$f(\gamma) = x^T Ax + \gamma x^T (d^T A)^T + x^T A\gamma d + \gamma^2 d^T Ad + b^T x + b^T\gamma d$$

$$f(\gamma) = x^T Ax + 2\gamma x^T Ad + \gamma^2 d^T Ad + b^T x + b^T\gamma d$$

The gradient is taken over scalar γ:

$$\nabla f(\gamma) = 2x^T Ad + 2\gamma d^T Ad + b^T d$$

From $\nabla f(\gamma^*) = 0$:

$$\gamma^* = -\frac{b^T d + 2x^T Ad}{2d^T Ad}$$

$$\gamma^* = -\frac{(b,d) + 2(Ax,d)}{2(Ad,d)}$$

Answer: $\gamma^* = -\frac{(b,d)+2(Ax,d)}{2(Ad,d)}$

6 Subdifferential

Given

$$[x^2-1]_+ = max(0, x^2-1)$$

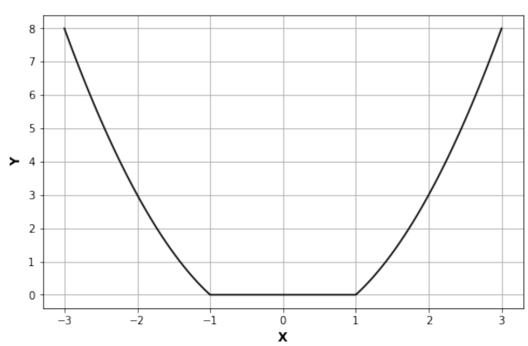

Figure 1: $max(0, x^2-1)$

The subgradient is vector v at point x_0 if following holds: $v(x-x_0) \le f(x) - f(x_0)$, the subdifferential at point x_0 is $\partial f(x_0) = \{v\}$. From Figure 1 we can see, that for $x > 1$ and $x < -1$ the $\partial f(x)$ is defined and equals $2x$. Similarly for interval $(-1,1)$ where $\partial f(x) = 0$. In points -1 and 1 we can place a set of tangents $[-2,0]$ and $[0,2]$ respectively.

Answer: $\partial f(x < -1) = \{2x : x < -1\}$; $\partial f(x > 1) = \{2x : x > 1\}$; $\partial f(x \in (-1,1)) = \{0 : x \in (-1,1)\}$; $\partial f(-1) = [-2,0]$; $\partial f(1) = [0,2]$

7 Steepest-descend 1

Given

$$f(x) = \frac{1}{2}x^TQx - x^Tb,\ b \in \mathbb{R}^n,\ Q \in \mathbb{R}^{n\times n},\ Q \succ 0$$

Suppose algorithm converges in 1 step, then x^1 is a solution. From neccessary condition we should get $\nabla f(x^1) = Qx^1 - b = 0$. Substitying $x^1 = x^0 - \alpha\nabla f(x^0)$ to $\nabla f(x^1) = 0$:

$$Q(x^0 - \alpha\nabla f(x^0)) - b = 0$$

Note that:

$$\nabla f(x^0) = Qx^0 - b$$

This turns out to be our eigenvector g^0 of Q from the problem statement. Continuing:

$$Q(x^0 - \alpha g^0) - b = Qx^0 - \alpha Qg^0 - b = (Qx^0 - b) - \alpha Qg^0 = 0$$

$$g^0\frac{1}{\alpha} = Qg^0$$

Since g^0 is the eigenvector of Q so $\frac{1}{\alpha}$ is eigenvalue. So since x^1 is the solution, then:

$$x^1 = x^0 - \alpha g^0$$

Multiply each side by Q:

$$Qx^1 = Qx^0 - \alpha Qg^0 = Qx^0 - \alpha\frac{1}{\alpha}g^0 = Qx^0 - g^0 = Qx^0 - Qx^0 + b$$

From where:

$$x^1 = Q^{-1}b$$

This holds if and only if g^0 is the eigenvector of Q.

8 Steepest-descend 2

Given

$$f(x, y) = x^2 + xy + 10y^2 - 22y - 5x$$

The steepest-descend algorihm is as follows:

Algorithm 1: Steepest-descend

x^0 - initial guess;
f - objective function;
L^0 - initial error;
ϵ - tolerance;
$\hat{\alpha} \leftarrow 1$ - initial step-size for backtracking algorihm;
$\gamma \in (0, 0.5)$ - backtracking algorihm parameter 1;
$\beta \in (0, 1)$ - backtracking algorihm parameter 2;
while $L^k > \epsilon$ **do**
  $\alpha \leftarrow \hat{\alpha}$;
  $p^k \leftarrow -\nabla f(x)$;
  while $f(x^k + \alpha p^k) > f(x^k) + \gamma\alpha(\nabla f(x^k), p^k)$ **do**
    $\alpha \leftarrow \beta\alpha$;
  end
  $x^{k+1} \leftarrow x^k - \alpha\nabla f(x^k)$;
  $L^{k+1} \leftarrow ||\nabla f(x^{k+1})||$
end
Result: x^{k+1}

For this assignment tolerance $\epsilon = 10^{-4}$ was chosen, $\gamma = 0.1$ and $\beta = 0.5$.

8.1 Starting point $x^0, y^0 = 1, 10$

Algorithm converged in 49 steps. Min value $f(x^*, y^*) = -16$. $[x^*, y^*] = [1.99, 0.99]^T$ (Figure 2). First 20 iterations are filled in table below:

k	x	y	f
1	0.56	-1.19	786.0000
2	0.88	1.64	37.0625
3	0.98	0.91	-11.4029
4	1.25	1.26	-14.7877
5	1.32	0.98	-14.9454
6	1.49	1.11	-15.5268
7	1.61	0.90	-15.6764
8	1.66	1.05	-15.6953
9	1.74	0.97	-15.8777
10	1.81	1.08	-15.9125
11	1.83	0.99	-15.9104
12	1.87	1.03	-15.9682
13	1.90	0.96	-15.9760
14	1.91	1.02	-15.9732
15	1.93	0.99	-15.9917
16	1.94	1.01	-15.9933
17	1.97	0.99	-15.9967
18	1.97	1.01	-15.9965
19	1.98	0.99	-15.9992
20	1.98	1.00	-15.9992

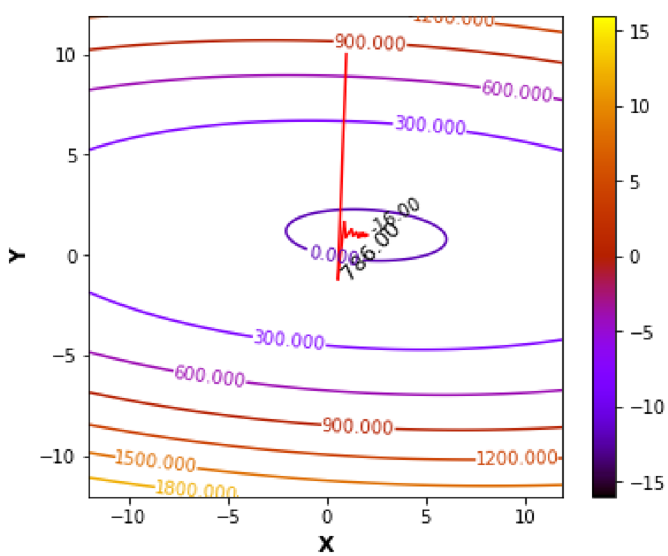

Figure 2: Steepest-descend for $x^0 = 1,\ y^0 = 10$

8.2 Starting point $x^0, y^0 = 10, 10$

Algorithm converged in 57 steps. Min value $f(x^*, y^*) = -16$. $[x^*, y^*] = [2.00, 0.99]^T$ (Figure 3). First 20 iterations are filled in table below:

k	x	y	f
1	8.44	-1.75	930.0000
2	7.80	1.29	83.3633
3	6.32	-0.15	20.1628
4	5.38	2.19	10.9656
5	4.89	0.49	13.6347
6	4.23	1.40	-6.5528
7	3.62	0.12	-8.5180
8	3.47	1.12	-7.0141
9	3.09	0.64	-13.5119
10	2.98	1.02	-13.8889
11	2.73	0.84	-15.0195
12	2.57	1.14	-15.3383
13	2.49	0.93	-15.3938
14	2.37	1.05	-15.7470
15	2.27	0.88	-15.8219
16	2.25	1.01	-15.8228
17	2.18	0.95	-15.9344
18	2.14	1.05	-15.9514
19	2.12	0.98	-15.9473
20	2.10	1.02	-15.9829

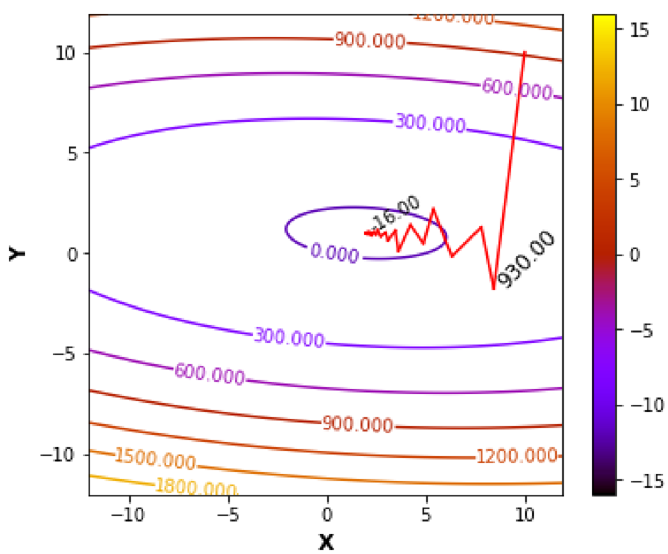

Figure 3: Steepest-descend for $x^0 = 10,\ y^0 = 10$

8.3 Starting point $x^0, y^0 = 10, 1$

Algorithm converged in 54 steps. Min value $f(x^*, y^*) = -16$. $[x^*, y^*] = [2.00, 1.00]^T$ (Figure 4). First 20 iterations are filled in table below:

k	x	y	f
1	6.00	-1.00	48.0000
2	5.62	1.25	32.0000
3	4.69	0.17	-1.3281
4	4.40	1.04	-4.1450
5	3.19	0.24	-10.1149
6	3.09	1.11	-9.7500
7	2.80	0.69	-14.5545
8	2.72	1.03	-14.6511
9	2.54	0.87	-15.4518
10	2.42	1.13	-15.6099
11	2.36	0.94	-15.6040
12	2.28	1.04	-15.8576
13	2.20	0.90	-15.8933
14	2.18	1.01	-15.8817
15	2.14	0.96	-15.9628
16	2.11	1.04	-15.9703
17	2.09	0.98	-15.9641
18	2.07	1.02	-15.9902
19	2.06	0.99	-15.9916
20	2.05	1.00	-15.9961

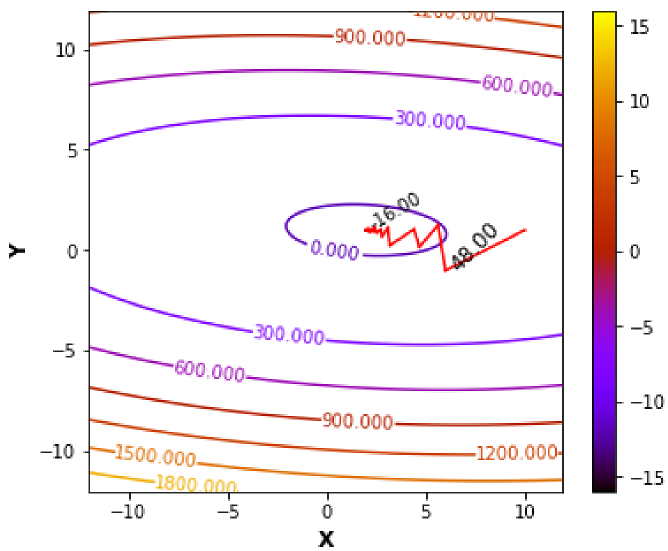

Figure 4: Steepest-descend for $x^0 = 10,\ y^0 = 1$

Answer: as we can note three different starting points lead to 49-57 iterations to converge. It means that in our configuration the steepest-descent was able to find same minima with small variation in number of iterations. This is due to the optimal-step size problem, we are not using constant α, but such that brings closest minimal value of function at x^{k+1}.

9 Steepest-descend 3

Given

$$f(x_1, x_2, \ldots, x_n) = \frac{1}{4}(x_1 - 1)^2 + \sum_{i=2}^{n}(2x_{i-1}^2 - x_i - 1)^2$$

9.1 Given $n = 3$ and $x^0 = [-1.5, 1, \ldots, 1]^T$

9.1.1 The first iteration of steepest-descend

This solution also holds for $n = 10$. The gradient components:

$$\frac{\partial}{\partial x_1} f(x^0) = \frac{1}{2}(x_1 - 1)8x_1(2x_1^2 - x_2 - 1) = -31.25$$

$$\frac{\partial}{\partial x_2} f(x^0) = 8x_2(2x_2^2 - x_3 - 1) - 2(2x_1^2 - x_2 - 1) = -5$$

$$\frac{\partial}{\partial x_i} f(x^0) = 0 \text{ for } i >= 3$$

9.1.2 Numerical solution

For this assignment same steepest-descent algorithm as in previous problem was chosen. The parameters are: tolerance $\epsilon = 10^{-3}$, $\gamma = 0.1$ and $\beta = 0.1$.

1. $\alpha^k = \text{argmin} f(x^k - \alpha \nabla f(x^k))$. Converged in 31 iterations. $x^* = [1.0067\ 1.0273\ 1.1108]^T$, $f(x^*) = 1.14 \cdot 10^{-5}$.
2. $\alpha^k = 0.1$. Algorithm diverged.
3. $\alpha^k = 0.5$. Algorithm diverged.
4. $\alpha^k = 1.0$ and stopping criteria $||x^{k+1} - x^k|| \leq 10^{-6}$. Algorithm diverged.

9.1.3 Use Python *scipy.optimize.minimize* to solve the problem

Algorithm (SLSQP) converged in 60 iterations. $x^* = [0.99999717, 0.99998847, 0.99995387]^T$, $f(x^*) = 2.046 \cdot 10^{-12}$.

9.2 Given $n = 10$ and $x^0 = [-1.5, 1, \ldots, 1]^T$

9.2.1 Numerical solution

The parameters are: tolerance $\epsilon = 10^{-3}$, $\gamma = 0.1$ and $\beta = 0.1$.

1. $\alpha^k = \text{argmin} f(x^k - \alpha \nabla f(x^k))$. Converged in 48 iterations. $x^* = [1.\ 1.\ 1.\ 1.\ 1.\ 1.\ 1.\ 1.\ 1.\ 1.\ 1.]^T$, $f(x^*) = 9.24 \cdot 10^{-9}$.
2. $\alpha^k = 0.1$. Algorithm diverged.
3. $\alpha^k = 0.5$. Algorithm diverged.
4. $\alpha^k = 1.0$ and stopping criteria $||x^{k+1} - x^k|| \leq 10^{-6}$. Algorithm diverged.

9.2.2 Use Python *scipy.optimize.minimize* to solve the problem

Algorithm (SLSQP) converged in 64 iterations. $x^* = [0.9995\ 0.9979\ 0.9913\ 1.0\ 1.0\ 1.0\ 1.0\ 1.0\ 1.0\ 1.0\ 1.0]^T$, $f(x^*) = 1.045 \cdot 10^{-7}$.

9.3 Conclusion

We can see that for relatively large α algorithm did not converge. This is easy to explain: each next x^{k+1} point jumps over the closest minimal values of f, and since the step-size is constant it cannot be increased/decreased to reach some minimal value in its neighbourhood as in optimal step-size formulation. Ideally, if α is constant it should be as small as possible (near to 0), but then number of iterations will significantly increase.

10 Comment

Python notebook can be found here https://github.com/arx7ti/optimization-course/blob/main/Homework2.ipynb