Optimization

Marcin Kuta

Stationary points

Stationary point (critical point, equilibrium point)

$$\nabla F(x) = 0 \tag{1}$$

Theorem (First-order necessary conditions for minimum)

If x^* is a local minimizer and f is continuously differentiable in an open neighborhood of x^*, then $\nabla F(x^*) = 0$

Theorem (Second-order necessary conditions for minimum)

If x^* is a local minimizer and $\nabla^2 F$ is continuous in an open neighborhood of x^*, then $\nabla F(x^*) = 0$ and $\nabla^2 F(x^*)$ is positive semidefinite

Stationary points

Theorem (Second-order sufficient conditions)

Suppose that $\nabla^2 F$ is continuous in an open neighborhood of x^* and that $\nabla F(x^*) = 0$.
If $\nabla^2 F(x^*)$ is

- *positive definite, then x^* is a strict local minimizer of f.*
- *negative definite, then x^* is a strict local maximizer of f.*
- *indefinite, then x^* is a saddle point.*
- *singular, then various pathological situations can occur.*

Stationary points

$$\begin{vmatrix} a_{11} \end{vmatrix}, \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}, \ldots \begin{vmatrix} a_{11} & \ldots & a_{1n} \\ \ldots & \ldots & \ldots \\ a_{n1} & \ldots & a_{nn} \end{vmatrix}$$

If sequence of signs of determinants is

- all positive $\Rightarrow$ A positive definite $\Rightarrow$ minimum
- alternates, starting from negative $\Rightarrow$ A negative definite $\Rightarrow$ maximum
- any sign is wrong $\Rightarrow$ A indefinite $\Rightarrow$ saddle point.
- no sign is wrong, but one or more terms is 0 $\Rightarrow$ A is positive semidefinite or negative semidefinite $\Rightarrow$ more delicate work is needed

Methods

Solving equations	Optimization
bisection method	golden section search
secant method	successive parabolic interpolation
Newton's method	Newton's method
Conjugate Gradient method	Conjugate Gradient method

Convergence of optimization methods

Convergence

- sublinear

$$\lim_{n\to\infty} \frac{||x_{n+1} - x^*||}{||x_n - x^*||} = C, \quad C = 1 \tag{2}$$

- linear

$$\lim_{n\to\infty} \frac{||x_{n+1} - x^*||}{||x_n - x^*||} = C, \quad 0 < C < 1 \tag{3}$$

- superlinear

$$\lim_{n\to\infty} \frac{||x_{n+1} - x^*||}{||x_n - x^*||} = 0 \tag{4}$$

- quadratic

$$\lim_{n\to\infty} \frac{||x_{n+1} - x^*||}{||x_n - x^*||^2} = C, \quad C > 0 \tag{5}$$

Convergence of optimization methods

Convergence of order p at rate C

$$\lim_{k \to \infty} \frac{||x_{n+1} - x^*||}{||x_n - x^*||^p} = C, \;\; C > 0 \tag{6}$$

Examples:

- $x_n = 1/n$: sublinear convergence
- $x_n = 1/n^2$: sublinear convergence
- $x_n = a^n, \; 0 < a < 1$: linear convergence with $C = a$
- $x_n = a^{2n}, \; 0 < a < 1$: linear convergence with $C = a^2$
- $x_n = a^{n^2}, \; 0 < a < 1$: superlinear convergence
- $x_n = a^{2^n}, \; 0 < a < 1$: quadratic convergence

Convergence of optimization methods

Linear order of convergence ($p = 1$):

- $||x_n - x^*|| = O(C^n)$

Quadratic order of convergence ($p = 2$):

- the number of correct digits approximately doubles at each iteration.

Convergence of optimization methods

Method	Convergence
coordinate descent method	no convergence
golden section search	linear, $r = 1$, $C \approx 0.618$
successive parabolic interpolation	superlinear, $r \approx 1.324$
steepest descent	linear, $r = 1$
Newton's method	quadratic, $r = 2$
Quasi-Netwon methods	superlinear
–BFGS	superlinear
Conjugate Gradient method	linear

Gradient descent

- $f : \mathbb{R}^n \to \mathbb{R}$
- $x_k \in \mathbb{R}^n$
- $\nabla f(x) \in \mathbb{R}^n$
- $\alpha_k \in \mathbb{R}$

Gradient descent

Algorithm Gradient Descent

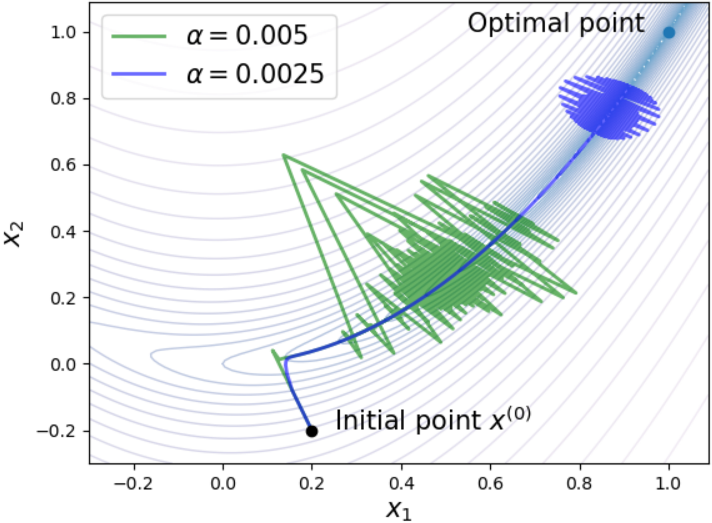
x_0 = initial guess
for $k \in \{0, 1, 2, \ldots\}$ **do**
    $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$
end for

- fixed learning rate: $\alpha_k = \alpha$
- exponentially decaying learning rate: $\alpha_k = \alpha\gamma^k$
- gradient descent with exact line search:

$$\alpha_k = \arg\min_{\alpha} f(x_k - \alpha \nabla f(x_k))$$

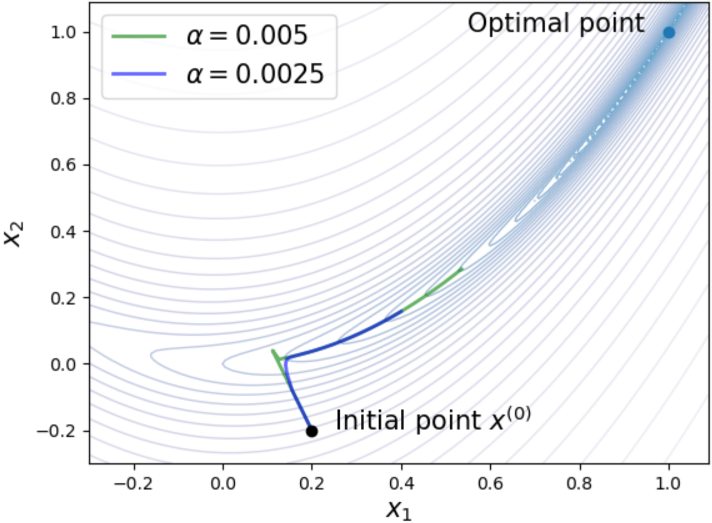
Gradient descent

Fixed learning rate: $\alpha_k = \alpha$



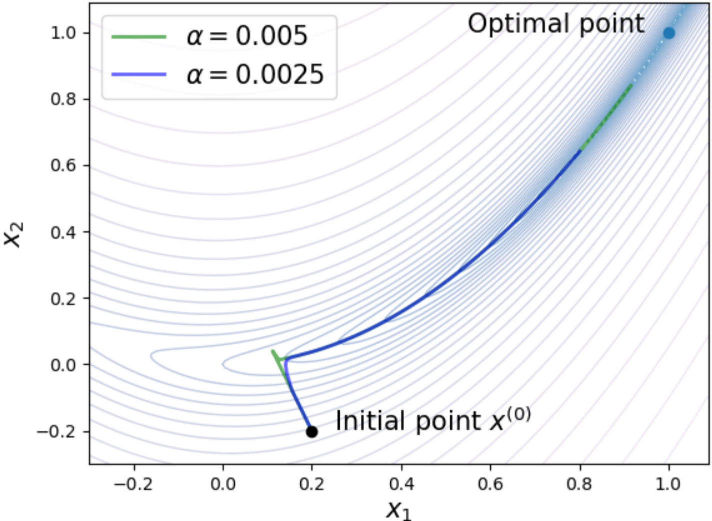
Gradient descent

Exponentially decaying learning rate: $\alpha_k = \alpha\gamma^k,\ \gamma = 0.99$



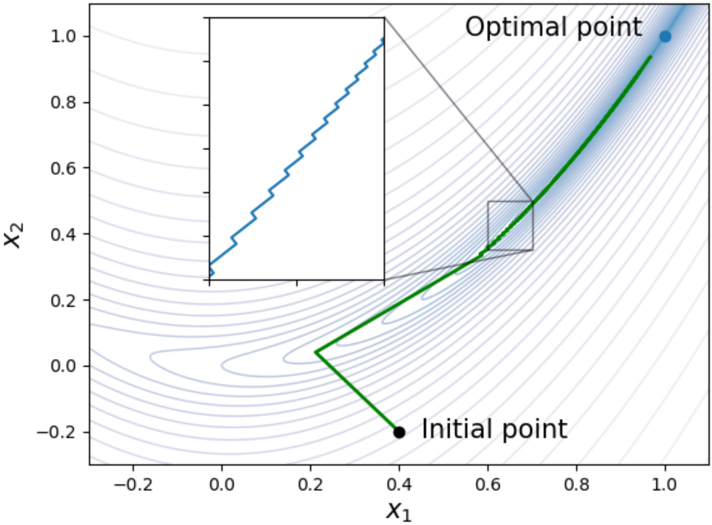
Gradient descent

Inverse time decaying learning rate: $\alpha_k = \frac{\alpha}{1+\gamma k}$, $\gamma = 0.01$



Gradient descent

Exact line search: $\alpha_k = \arg\min_\alpha f(x_k - \alpha \nabla f(x_k))$



Gradient descent with momentum

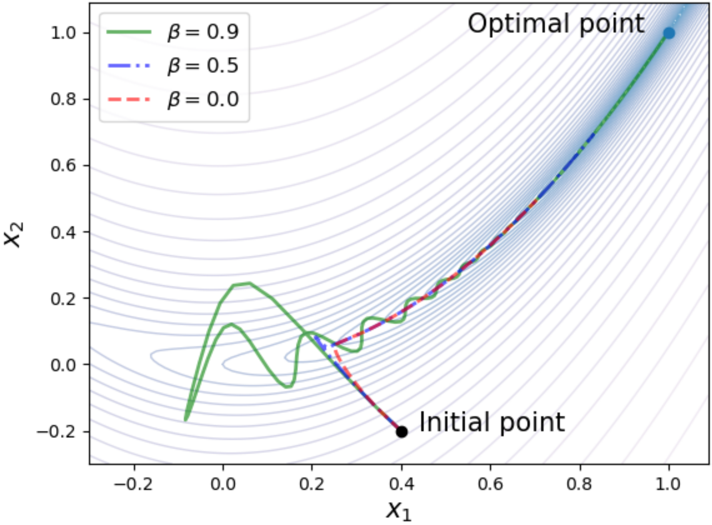
Algorithm Gradient Descent with momentum

$x_0 =$ initial guess
$v_0 = 0$
for $k \in \{0, 1, 2, \dots\}$ **do**
  $v_{k+1} = \beta v_k + (1-\beta)\nabla f(x_k)$ ▷ Momentum update
  $x_{k+1} = x_k - \alpha v_{k+1}$ ▷ Variable update
end for

Momentum applies exponential smoothing to the past gradient vectors:

$$v_{k+1} = (1-\beta)\sum_{i=0}^{k} \beta^{k-i}\nabla f(x_i)$$

Gradient descent with momentum



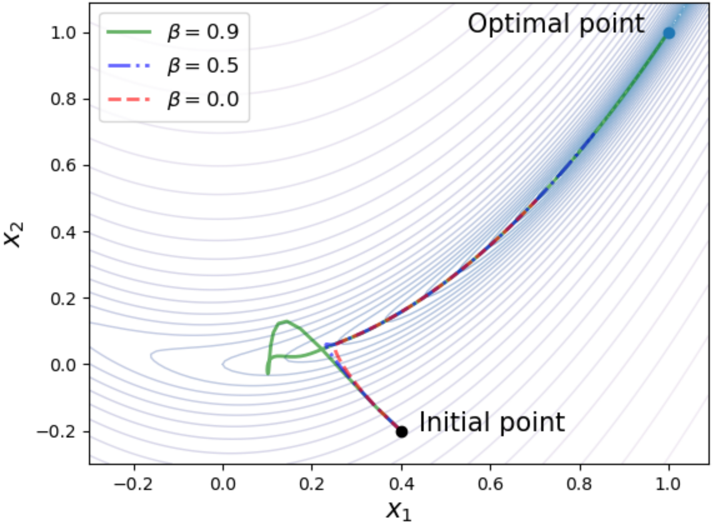
Gradient descent with Nesterov momentum

Algorithm Gradient Descent with Nesterov momentum

1. $x_0 =$ initial guess
2. $v_0 = 0$
3. **for** $k \in \{0, 1, 2, \dots\}$ **do**
4. $\quad v_{k+1} = \beta v_k + (1-\beta)\nabla f(\underbrace{x_k - \alpha\beta v_k}_{\text{predicted } \hat{x}_{k+1}})$ ▷ Momentum update
5. $\quad x_{k+1} = x_k - \alpha v_{k+1}$ ▷ Variable update
6. **end for**

Quantity $\nabla f(x_k - \alpha\beta v_k)$ is a look-ahead gradient.

Gradient descent with Nesterov momentum



ADAM – Adaptive Moment Estimation Algorithm

ADAM is based on three ideas:

- adaptive learning rates per coordinates, $\alpha \in \mathbb{R}^n$
- exponential smoothing (momentum or RMSProp)
- bias correction

ADAM – Adaptive Moment Estimation Algorithm

Algorithm ADAM

$x_0 =$ initial guess
$v_0 = 0$
$s_0 = 0$
for $k \in \{0, 1, 2, \dots\}$ **do**
  $v_{k+1} = \beta v_k + (1 - \beta)\nabla f(x_k)$ ▷ Momentum update
  $s_{k+1} = \gamma s_k + (1 - \gamma)\nabla f(x_k) \odot \nabla f(x_k)$ ▷ Second momentum ▷ update
  $\hat{v}_{k+1} = \frac{v_{k+1}}{1-\beta^{k+1}}$ ▷ Bias correction
  $\hat{s}_{k+1} = \frac{s_{k+1}}{1-\gamma^{k+1}}$ ▷ Bias correction
  $x_{k+1} = x_k - \frac{\alpha}{\sqrt{\hat{s}_{k+1}} + \epsilon} \odot \hat{v}_{k+1}$ ▷ Variable update
end for

Gradient descent and linear model

$$Ax \cong y \tag{7}$$

$$\min_x J(x) \tag{8}$$

$$J(x) = ||Ax - y||_2^2 = (Ax - y)^2 \tag{9}$$

$$\nabla J(x) = 2A^T(Ax - y) \tag{10}$$

$$x_{k+1} = x_k - \alpha \nabla (Ax_k - y)^2 = x_k - \alpha \cdot 2A^T(Ax_k - y) \tag{11}$$

$$x_{k+1} = (I - \alpha \cdot 2A^T A)x_k + \alpha \cdot 2A^T y \tag{12}$$

Gradient descent and linear model

Matrix $A^T A$ is symmetric and positive-definite matrix, thus its eigenvalues are real and positive.
Let λ_{min} and λ_{max} be the smallest and the largest eigenvalue of $A^T A$.
Gradient descent iteration (11) converges for

$$\alpha < \frac{1}{\lambda_{\text{max}}} \tag{13}$$

Optimal size of learning rate, for which minimum is achieved with the fewest number of iterations, equals

$$\alpha_{\text{opt}} = \frac{1}{\lambda_{\text{max}} + \lambda_{\text{min}}} \tag{14}$$

Computational cost

$$Ax \cong y, \quad A \in \mathbb{R}^{n \times m}, \quad y \in \mathbb{R}^{n \times 1} \tag{15}$$

Normal equation

- Cost: $O(nm^2) + O(m^3)$

Gradient descent

- Cost per iteration: $O(nm)$
- Number of iterations is $O(\log(\epsilon))$, where ϵ is threshold on accepted error

Newton's method

Let $q(x)$ be quadratic approximation of $f(x)$ around $x^{(k)}$:

$$q(x) = f(x^{(k)}) + (x - x^{(k)})f'(x^{(k)}) + \frac{(x - x^{(k)})^2}{2} f''(x^{(k)}) \quad (16)$$

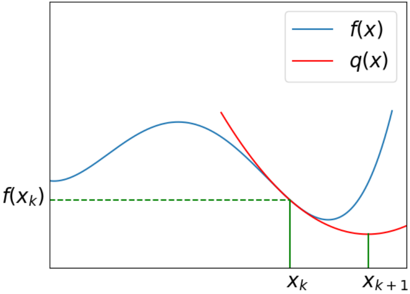
Setting $q'(x) = 0$ we get

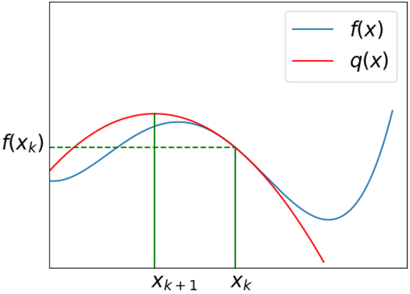
$$f'(x^{(k)}) + (x - x^{(k)})f''(x^{(k)}) = 0 \quad (17)$$

Solving (17) for x and setting $x_{k+1} = x$ we get

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)} \quad (18)$$

Newton's method





Newton's method

$$H_f(x) = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \cdots & \cdots & \ddots & \cdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1 +} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix} \tag{19}$$

$$\begin{aligned} H_f(x_k)s_k &= -\nabla f(x_k) \\ x_{k+1} &= x_k + s_k \end{aligned} \tag{20}$$

Stopping critera

1. absolute improvement

$$|f(x_k) - f(x_{k-1})| < \epsilon$$

2. relative improvement

$$\frac{|f(x_k) - f(x_{k-1})|}{|f(x_{k-1})|} < \epsilon$$

3. gradient magnitude

$$|\nabla f(x_k)| < \epsilon \text{ or } \alpha_k |\nabla f(x_k)| < \epsilon$$

4. number of iterations

References

[1] http://heath.cs.illinois.edu/scicomp/notes/cs450_chapt06.pdf

[2] http://www.benfrederickson.com/numerical-optimization

[3] Mykel Kochenderfer, Tim Wheeler,
Algorithms for Optimization, 2019
https://algorithmsbook.com/optimization

[4] Benoit Liquet, Sarat Moka, Yoni Nazarathy,
Mathematical Engineering of Deep Learning, 2024,
https://deeplearningmath.org

[5] Jorge Nocedal, Stephen J. Wright,
Numerical Optimization, 2006,
https://doi.org/10.1007/978-0-387-40065-5