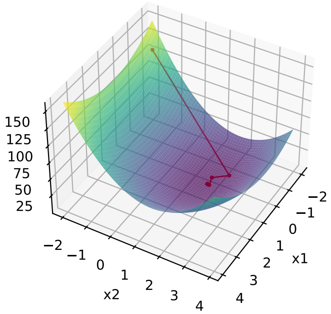
Optimization in Machine Learning

First order methods

GD – Example with adaptive step-length



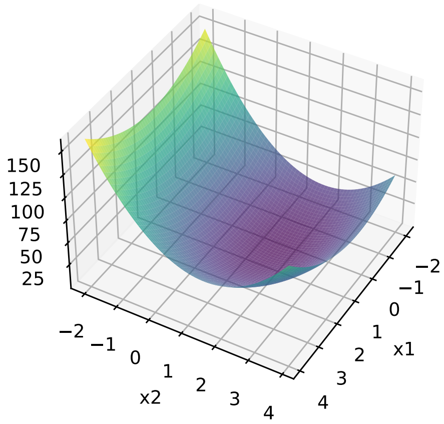
Learning goals

- Recap - Gradient Decent and Backtracking

MULTIVARIATE FUNCTION

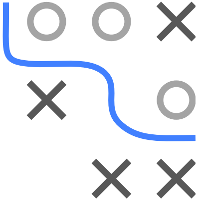
Let $f : \mathbb{R}^2 \to \mathbb{R}, (x_1, x_2) \mapsto (3x_2 - 5)^2 + (2x_1 - 1)^2 + x_1 x_2 - 5,$

$$\nabla f(\mathbf{x}) = \begin{pmatrix} 8x_1 - 4 + x_2 \\ 18x_2 - 30 + x_1 \end{pmatrix}$$



Function-plot

GRADIENT DECENT - ITERATION 1



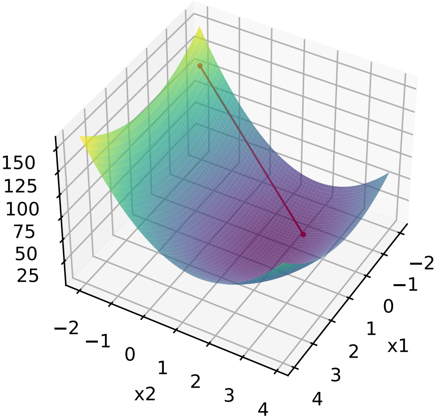
Start-point:

$$\mathbf{x}^0 = \begin{pmatrix} -1,5 \\ -1,7 \end{pmatrix},$$

$$f(\mathbf{x}^0) = (3(-1,7) - 5)^2 + (2(-1,5) - 1)^2 + (-1,5)(-1,7) - 5 = 115,56$$

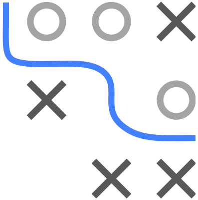
$$\mathbf{d}^0 = -\nabla f(\mathbf{x}^0) = -\begin{pmatrix} 8(-1,5) - 4 + (-1,7) \\ 18(-1,7) - 30 + (-1,5) \end{pmatrix} = -\begin{pmatrix} 17,7 \\ 62,1 \end{pmatrix},$$

$$\alpha_{init} = 0,5$$



Function-plot

STEP-LENGTH ADJUSTMENT WITH BACKTRACKING - ITERATION 1



Initial step-length: $\alpha = \alpha_{init} = 0,5$

$\mathbf{x}^1(\alpha) = \mathbf{x}^0 + \alpha \mathbf{d}^0 = \begin{pmatrix} 7,35 \\ 29,35 \end{pmatrix}$

$f(\mathbf{x}^1) = 7300,715$

$f_a(\mathbf{x}^1) = f(\mathbf{x}^0) + \alpha_{init}\gamma_1(\nabla f(\mathbf{x}^0)^T \mathbf{d}^0) = 115,56 + 0,5 \cdot 0,1 \cdot \begin{pmatrix} 17,7 & 62,1 \end{pmatrix} \begin{pmatrix} -17,7 \\ -62,1 \end{pmatrix} = -921,865$

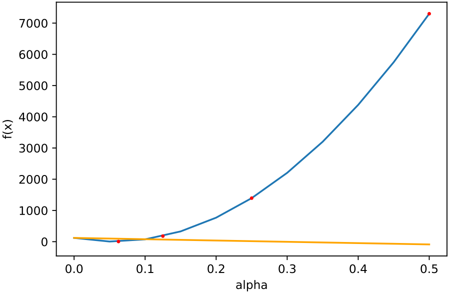
$\Rightarrow f(\mathbf{x}^1) > f_a(\mathbf{x}^1)$

$\Rightarrow \alpha = \tau \cdot \alpha = 0,5 \cdot 0,5 = 0,25$

$\Rightarrow$ restart at $\mathbf{x}^1(\alpha)$

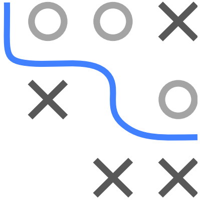
$\vdots$

$\alpha = 0,0625$



Armijo-rule

GRADIENT DECENT - ITERATION 2



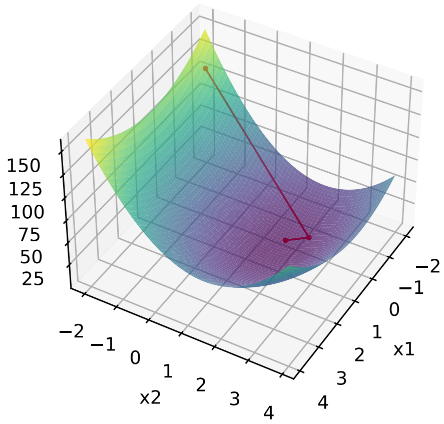
New solution:

$$\mathbf{x}^1 = \begin{pmatrix} -0,39375 \\ 2,18125 \end{pmatrix},$$

$$f(\mathbf{x}^1) = 4,71945312$$

$$\mathbf{d}^1 = -\nabla f(\mathbf{x}^1) = -\begin{pmatrix} 4,96875 \\ -8,86875 \end{pmatrix},$$

$$\alpha_{init} = 0,5$$



Function-plot

STEP-LENGTH ADJUSTMENT WITH BACKTRACKING - ITERATION 2

Initial step-length: $\alpha = \alpha_{init} = 0,5$

$\mathbf{x}^2(\alpha) = \mathbf{x}^1 + \alpha \mathbf{d}^1 = \begin{pmatrix} 7,35 \\ 29,35 \end{pmatrix}$

$f(\mathbf{x}^2) = 143.69281249$

$f_a(\mathbf{x}^2) = f(\mathbf{x}^1) + \alpha_{init}\gamma_1(\nabla f(\mathbf{x}^1)^T \mathbf{d}^1) = 4,71945312 + 0,5 \cdot 0,1 \cdot \begin{pmatrix} 4,96875 & -8,86875 \end{pmatrix} \begin{pmatrix} -4,96875 \\ 8,86875 \end{pmatrix} \approx -21.1163476562499 7$

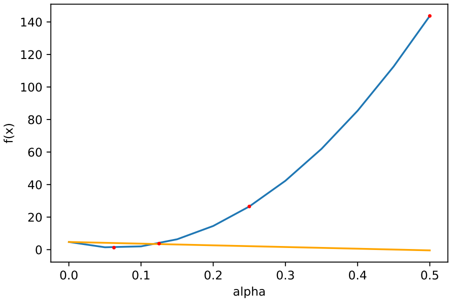
$\Rightarrow f(\mathbf{x}^1) > f_a(\mathbf{x}^1)$

$\Rightarrow \alpha = \tau \cdot \alpha = 0,5 \cdot 0,5 = 0,25$

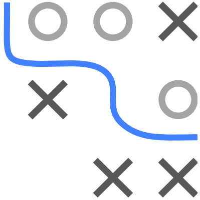
$\Rightarrow$ restart at $\mathbf{x}^1(\alpha)$

$\vdots$

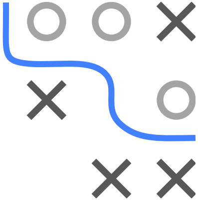
$\alpha = 0,0625$



Armijo-rule



GRADIENT DECENT - ITERATION 3



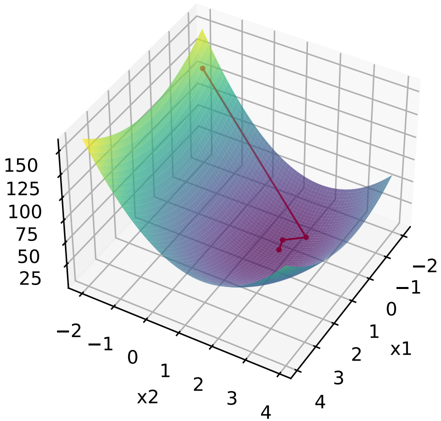
New solution:

$$\mathbf{x}^2 \approx \begin{pmatrix} -0,08320312 \\ 1,62695312 \end{pmatrix},$$

$$f(\mathbf{x}^2) \approx 1,2393304443359374$$

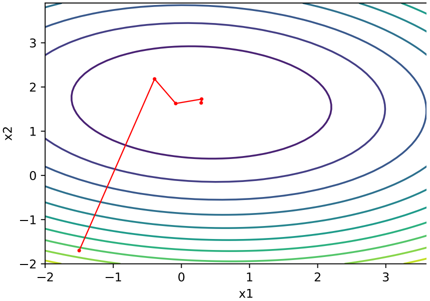
$$\mathbf{d}^1 = -\nabla f(\mathbf{x}^1) \approx \begin{pmatrix} 3,03867188 \\ 0,79804688 \end{pmatrix},$$

$$\alpha_{init} = 0,5$$

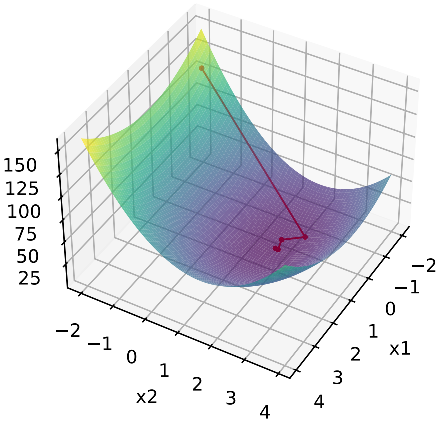


Function-plot

GRADIENT DECENT - ITERATION 4

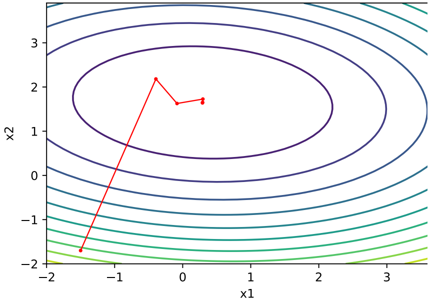


Contour-plot

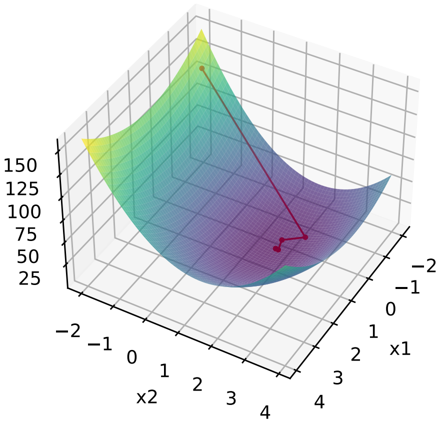


Function-plot

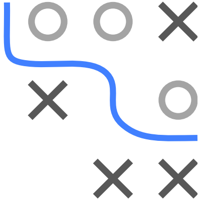
GRADIENT DECENT - ITERATION 5



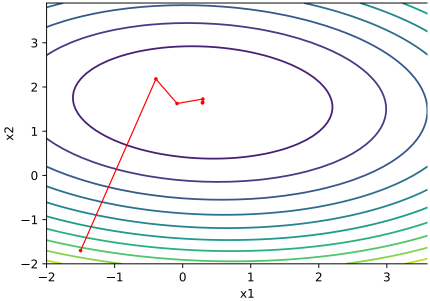
Contour-plot



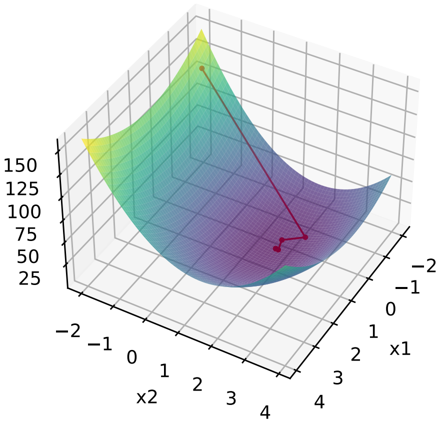
Function-plot



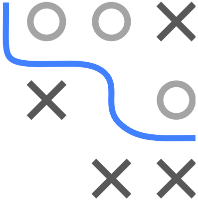
GRADIENT DECENT - ITERATION 6



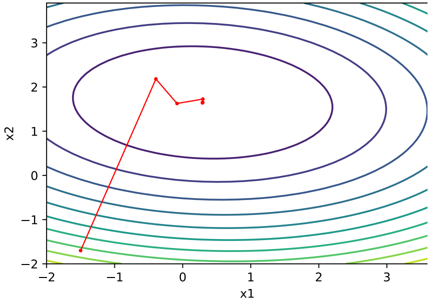
Contour-plot



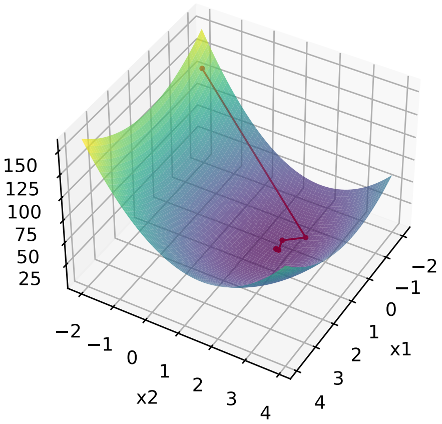
Function-plot



GRADIENT DECENT - ITERATION 7



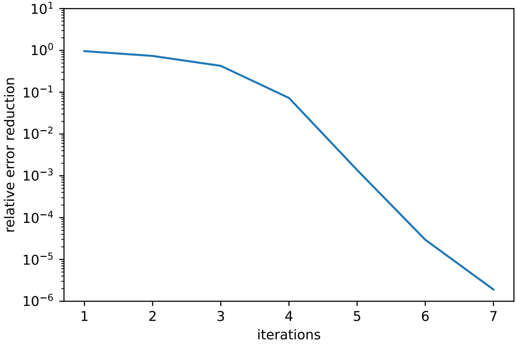
Contour-plot



Function-plot

GRADIENT DECENT

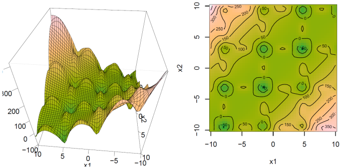
Convergence relative error: $\frac{f(x^{i-1})-f(x^{i})}{f(x^{i-1})}$



Convergence of relative error

Optimization in Machine Learning

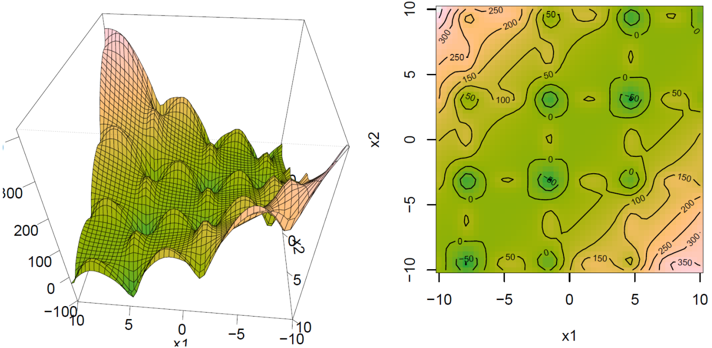
First order methods
GD – Multimodality and Saddle points



Learning goals

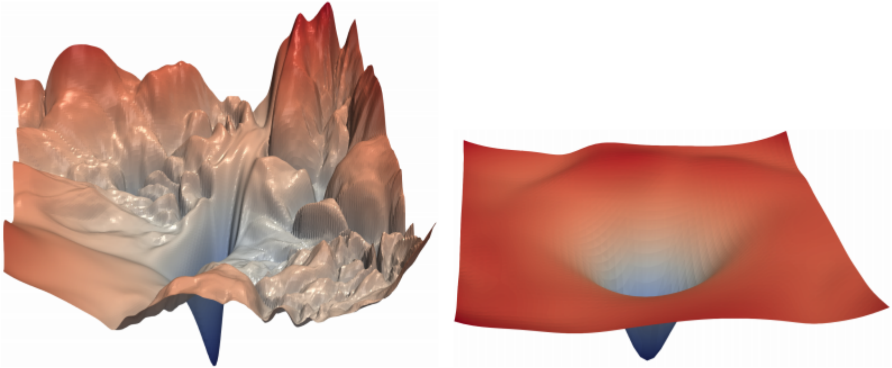
- Multimodality, GD result can be arbitrarily bad
- Saddle points, major problem in NN error landscapes, GD can get stuck or slow crawling

UNIMODAL VS. MULTIMODAL LOSS SURFACES



Snippet of a loss surface with many local optima

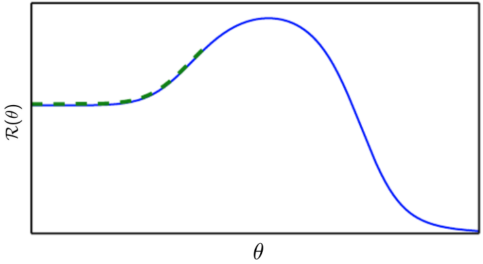
UNIMODAL VS. MULTIMODAL LOSS SURFACES / 2



In deep learning, we often find multimodal loss surfaces.
Left: Multimodal loss surface. **Right:** (Nearly) unimodal loss surface.
(Source: Hao Li et al., 2017.

GD: ONLY LOCALLY OPTIMAL MOVES

- GD makes only **locally** optimal moves
- It may move away from the global optimum

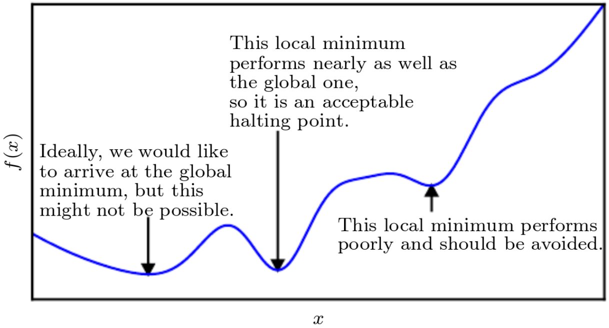


Source: Goodfellow et al., 2016

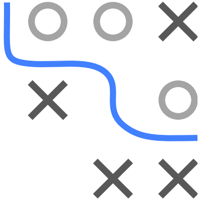
- Initialization on “wrong” side of the hill results in weak performance
- In higher dimensions, GD may move around the hill (potentially at the cost of longer trajectory and time to convergence)

LOCAL MINIMA

- **In practice:** Only local minima with high value compared to global minimium are problematic.

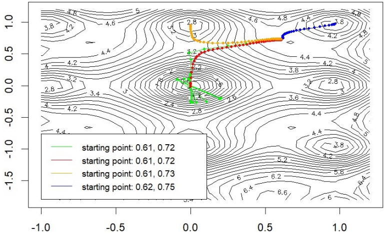


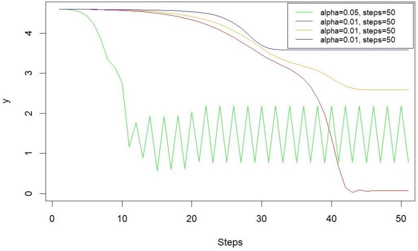
Source: Goodfellow et al., 2016



LOCAL MINIMA / 2

- Small differences in starting point or step size can lead to huge differences in the reached minimum or even to non-convergence





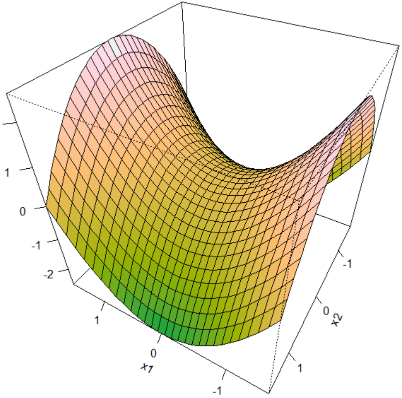
(Non-)Converging gradient descent for Ackley function

GD AT SADDLE POINTS

Example:

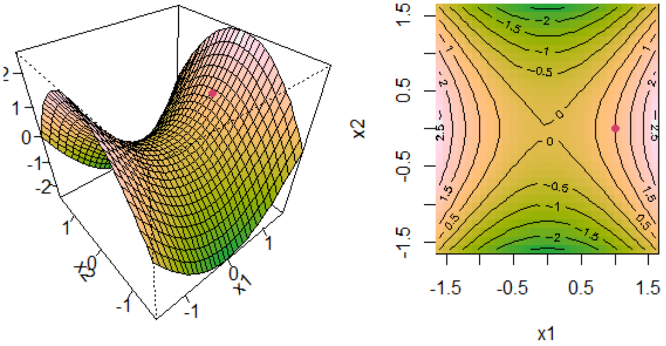
$$
\begin{aligned}
f(x_1, x_2) &= x_1^2 - x_2^2 \\
\nabla f(x_1, x_2) &= (2x_1, -2x_2)^\top \\
\boldsymbol{H} &= \begin{pmatrix} 2 & 0 \\ 0 & -2 \end{pmatrix}
\end{aligned}
$$

- Along x_1, curvature is positive ($\lambda_1 = 2 > 0$).
- Along x_2, curvature is negative ($\lambda_2 = -2 < 0$).



SADDLE POINT WITH GD

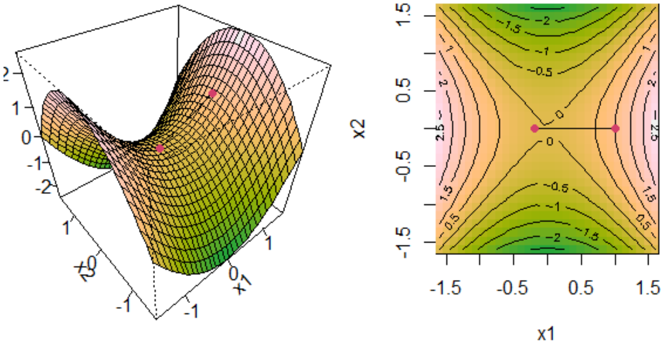
- How do saddle points impair optimization?
- Gradient-based algorithms **might** get stuck in saddle points



Red dot: Starting location

SADDLE POINT WITH GD

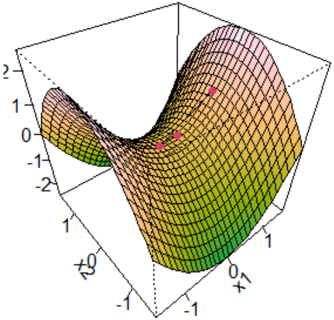
- How do saddle points impair optimization?
- Gradient-based algorithms **might** get stuck in saddle points

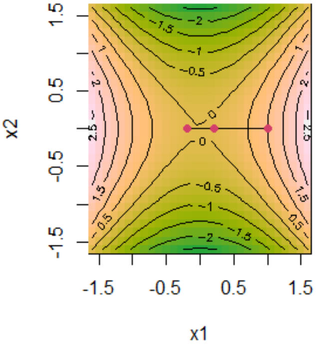


Step 1 ...

SADDLE POINT WITH GD

- How do saddle points impair optimization?
- Gradient-based algorithms **might** get stuck in saddle points

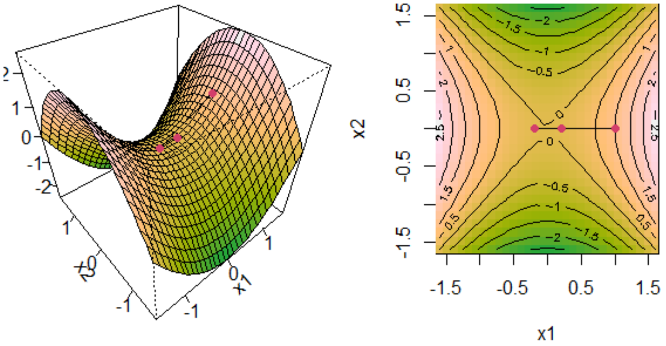




... Step 2 ...

SADDLE POINT WITH GD

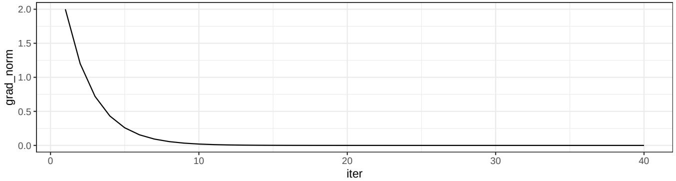
- How do saddle points impair optimization?
- Gradient-based algorithms **might** get stuck in saddle points



... Step 10 ... got stuck and cannot escape saddle point

SADDLE POINT WITH GD

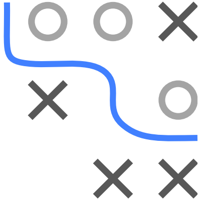
- How do saddle points impair optimization?
- Gradient-based algorithms **might** get stuck in saddle points

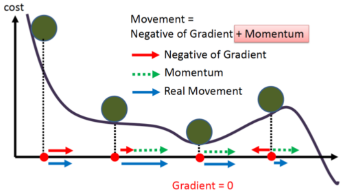


... Step 10 ... got stuck and cannot escape saddle point

Optimization in Machine Learning

First order methods
GD with Momentum

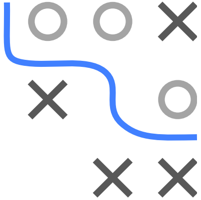




Learning goals

- Recap of GD problems
- Momentum definition
- Unrolling formula
- Examples
- Nesterov

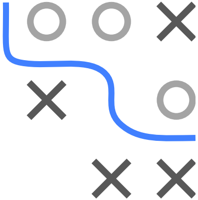
RECAP: WEAKNESSES OF GRADIENT DESCENT



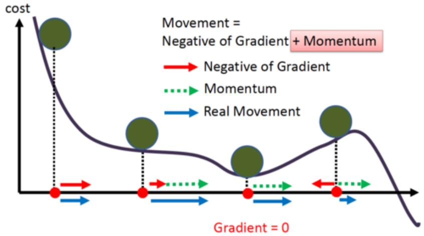
- **Zig-zagging behavior:** For ill-conditioned problems, GD moves with a zig-zag course to the optimum, since the gradient points approximately orthogonal in the shortest direction to the minimum.
- **Slow crawling:** may vanish rapidly close to stationary points (e.g. saddle points) and hence also slows down progress.
- **Trapped in stationary points:** In some functions GD converges to stationary points (e.g. saddle points) since gradient on all sides is fairly flat and the step size is too small to pass this flat part.

Aim: More efficient algorithms which quickly reach the minimum.

GD WITH MOMENTUM



- **Idea:** "Velocity" $\boldsymbol{\nu}$: Increasing if successive gradients point in the same direction but decreasing if they point in opposite directions

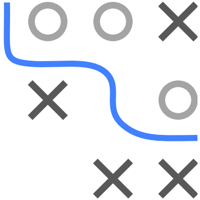


Source: Khandewal, *GD with Momentum, RMSprop and Adam Optimizer*, 2020.

- $\boldsymbol{\nu}$ is weighted moving average of previous gradients:

$$\boldsymbol{\nu}^{[t+1]} = \varphi \boldsymbol{\nu}^{[t]} - \alpha \nabla f(\mathbf{x}^{[t]})$$
$$\mathbf{x}^{[t+1]} = \mathbf{x}^{[t]} + \boldsymbol{\nu}^{[t+1]}$$

- $\varphi \in [0, 1]$ is additional hyperparameter

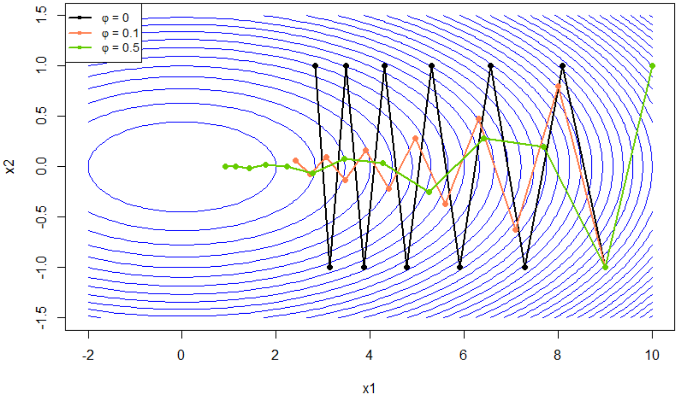


- Length of a single step depends on how large and aligned a sequence of gradients is
- Length of a single step grows if many successive gradients point in the same direction
- φ determines how strongly previous gradients are included in $\boldsymbol{\nu}$
- Common values for φ are 0.5, 0.9 and even 0.99
- In general, the larger φ is in relation to α, the more strongly previous gradients influence the current direction
- **Special case** $\varphi = 0$: "vanilla" gradient descent
- **Intuition:** GD with "short term memory" for the direction of motion

GD WITH MOMENTUM: ZIG-ZAG BEHAVIOUR

Consider a two-dimensional quadratic form $f(\mathbf{x}) = x_1^2/2 + 10x_2$.

Let $\mathbf{x}^{[0]} = (10, 1)^\top$ and $\alpha = 0.1$.

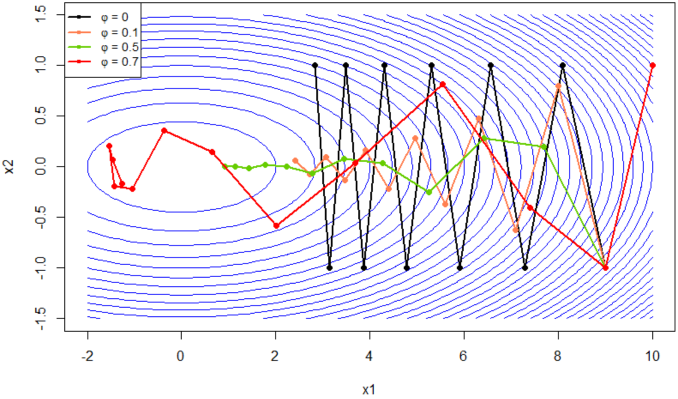


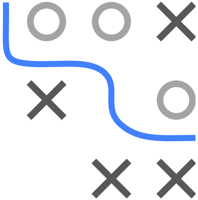
GD shows stronger zig-zag behaviour than GD with momentum.

GD WITH MOMENTUM: ZIG-ZAG BEHAVIOUR / 2

Caution:

- If momentum is too high, minimum is possibly missed
- We might go back and forth around or between local minima

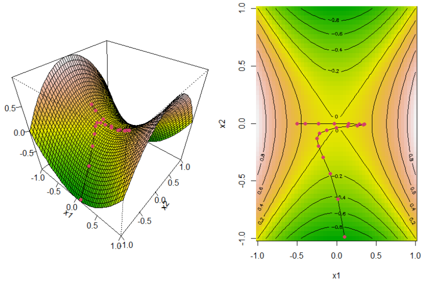




GD WITH MOMENTUM: SADDLE POINTS

Consider $f(\mathbf{x}) = x_1^2 - x_2^2$ with a saddle point at $(0, 0)^\top$.

Let $\mathbf{x}^{[0]} = (-1/2, 10^{-3})^\top$ and $\alpha = 0.1$.



GD was slowing down at the saddle point (vanishing gradient).
GD with momentum "breaks out" of the saddle point and moves on.

MOMENTUM: ANALYSIS

$$\boldsymbol{\nu}^{[1]} = \varphi \boldsymbol{\nu}^{[0]} - \alpha \nabla f(\mathbf{x}^{[0]})$$

$$\mathbf{x}^{[1]} = \mathbf{x}^{[0]} + \varphi \boldsymbol{\nu}^{[0]} - \alpha \nabla f(\mathbf{x}^{[0]})$$

MOMENTUM: ANALYSIS

$$\boldsymbol{\nu}^{[1]} = \varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]})$$

$$\mathbf{x}^{[1]} = \mathbf{x}^{[0]} + \varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]})$$

$$\begin{aligned}\boldsymbol{\nu}^{[2]} &= \varphi\boldsymbol{\nu}^{[1]} - \alpha\nabla f(\mathbf{x}^{[1]}) \\ &= \varphi(\varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]})) - \alpha\nabla f(\mathbf{x}^{[1]})\end{aligned}$$

$$\mathbf{x}^{[2]} = \mathbf{x}^{[1]} + \varphi(\varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]})) - \alpha\nabla f(\mathbf{x}^{[1]})$$

MOMENTUM: ANALYSIS

$$
\begin{aligned}
\boldsymbol{\nu}^{[1]} &= \varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]}) \\
\mathbf{x}^{[1]} &= \mathbf{x}^{[0]} + \varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]}) \\
\boldsymbol{\nu}^{[2]} &= \varphi\boldsymbol{\nu}^{[1]} - \alpha\nabla f(\mathbf{x}^{[1]}) \\
&= \varphi(\varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]})) - \alpha\nabla f(\mathbf{x}^{[1]}) \\
\mathbf{x}^{[2]} &= \mathbf{x}^{[1]} + \varphi(\varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]})) - \alpha\nabla f(\mathbf{x}^{[1]}) \\
\boldsymbol{\nu}^{[3]} &= \varphi\boldsymbol{\nu}^{[2]} - \alpha\nabla f(\mathbf{x}^{[2]}) \\
&= \varphi(\varphi(\varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]})) - \alpha\nabla f(\mathbf{x}^{[1]})) - \alpha\nabla f(\mathbf{x}^{[2]}) \\
\mathbf{x}^{[3]} &= \mathbf{x}^{[2]} + \varphi(\varphi(\varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]})) - \alpha\nabla f(\mathbf{x}^{[1]})) - \alpha\nabla f(\mathbf{x}^{[2]}) \\
&= \mathbf{x}^{[2]} + \varphi^3\boldsymbol{\nu}^{[0]} - \varphi^2\alpha\nabla f(\mathbf{x}^{[0]}) - \varphi\alpha\nabla f(\mathbf{x}^{[1]}) - \alpha\nabla f(\mathbf{x}^{[2]}) \\
&= \mathbf{x}^{[2]} - \alpha(\varphi^2\nabla f(\mathbf{x}^{[0]}) + \varphi^1\nabla f(\mathbf{x}^{[1]}) + \varphi^0\nabla f(\mathbf{x}^{[2]})) + \varphi^3\boldsymbol{\nu}^{[0]}
\end{aligned}
$$

MOMENTUM: ANALYSIS

$$
\begin{aligned}
\boldsymbol{\nu}^{[1]} &= \varphi \boldsymbol{\nu}^{[0]} - \alpha \nabla f(\mathbf{x}^{[0]}) \\
\mathbf{x}^{[1]} &= \mathbf{x}^{[0]} + \varphi \boldsymbol{\nu}^{[0]} - \alpha \nabla f(\mathbf{x}^{[0]}) \\
\boldsymbol{\nu}^{[2]} &= \varphi \boldsymbol{\nu}^{[1]} - \alpha \nabla f(\mathbf{x}^{[1]}) \\
&= \varphi(\varphi \boldsymbol{\nu}^{[0]} - \alpha \nabla f(\mathbf{x}^{[0]})) - \alpha \nabla f(\mathbf{x}^{[1]}) \\
\mathbf{x}^{[2]} &= \mathbf{x}^{[1]} + \varphi(\varphi \boldsymbol{\nu}^{[0]} - \alpha \nabla f(\mathbf{x}^{[0]})) - \alpha \nabla f(\mathbf{x}^{[1]}) \\
\boldsymbol{\nu}^{[3]} &= \varphi \boldsymbol{\nu}^{[2]} - \alpha \nabla f(\mathbf{x}^{[2]}) \\
&= \varphi(\varphi(\varphi \boldsymbol{\nu}^{[0]} - \alpha \nabla f(\mathbf{x}^{[0]})) - \alpha \nabla f(\mathbf{x}^{[1]})) - \alpha \nabla f(\mathbf{x}^{[2]}) \\
\mathbf{x}^{[3]} &= \mathbf{x}^{[2]} + \varphi(\varphi(\varphi \boldsymbol{\nu}^{[0]} - \alpha \nabla f(\mathbf{x}^{[0]})) - \alpha \nabla f(\mathbf{x}^{[1]})) - \alpha \nabla f(\mathbf{x}^{[2]}) \\
&= \mathbf{x}^{[2]} + \varphi^3 \boldsymbol{\nu}^{[0]} - \varphi^2 \alpha \nabla f(\mathbf{x}^{[0]}) - \varphi \alpha \nabla f(\mathbf{x}^{[1]}) - \alpha \nabla f(\mathbf{x}^{[2]}) \\
&= \mathbf{x}^{[2]} - \alpha(\varphi^2 \nabla f(\mathbf{x}^{[0]}) + \varphi^1 \nabla f(\mathbf{x}^{[1]}) + \varphi^0 \nabla f(\mathbf{x}^{[2]})) + \varphi^3 \boldsymbol{\nu}^{[0]} \\
\mathbf{x}^{[t+1]} &= \mathbf{x}^{[t]} - \alpha \sum_{j=0}^{t} \varphi^j \nabla f(\mathbf{x}^{[t-j]}) + \varphi^{t+1} \boldsymbol{\nu}^{[0]}
\end{aligned}
$$

MOMENTUM: INTUITION

Suppose momentum always observes the same gradient $\nabla f(\mathbf{x}^{[t]})$:

$$\begin{aligned}
\mathbf{x}^{[t+1]} &= \mathbf{x}^{[t]} - \alpha \sum_{j=0}^{t} \varphi^j \nabla f(\mathbf{x}^{[j]}) + \varphi^{t+1} \boldsymbol{\nu}^{[0]} \\
&= \mathbf{x}^{[t]} - \alpha \nabla f(\mathbf{x}^{[t]}) \sum_{j=0}^{t} \varphi^j + \varphi^{t+1} \boldsymbol{\nu}^{[0]} \\
&= \mathbf{x}^{[t]} - \alpha \nabla f(\mathbf{x}^{[t]}) \frac{1 - \varphi^{t+1}}{1 - \varphi} + \varphi^{t+1} \boldsymbol{\nu}^{[0]} \\
&\to \mathbf{x}^{[t]} - \alpha \nabla f(\mathbf{x}^{[t]}) \frac{1}{1 - \varphi} \qquad \text{for } t \to \infty.
\end{aligned}$$

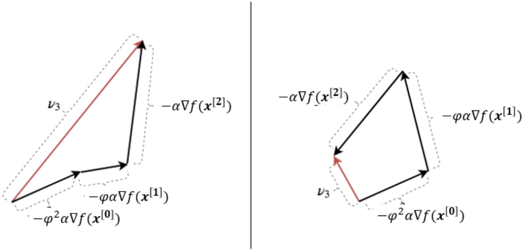
Momentum accelerates along $-\nabla f(\mathbf{x}^{[t]})$ to terminal velocity yielding step size $\alpha/(1 - \varphi)$.

Example: Momentum with $\varphi = 0.9$ corresponds to a tenfold increase in original step size α compared to vanilla gradient descent

MOMENTUM: INTUITION / 2

Vector $\boldsymbol{\nu}^{[3]}$ (for $\boldsymbol{\nu}^{[0]} = 0$):

$$\begin{aligned}\boldsymbol{\nu}^{[3]} &= \varphi(\varphi(\varphi\boldsymbol{\nu}^{[0]} - \alpha\nabla f(\mathbf{x}^{[0]})) - \alpha\nabla f(\mathbf{x}^{[1]})) - \alpha\nabla f(\mathbf{x}^{[2]}) \\ &= -\varphi^2\alpha\nabla f(\mathbf{x}^{[0]}) - \varphi\alpha\nabla f(\mathbf{x}^{[1]}) - \alpha\nabla f(\mathbf{x}^{[2]})\end{aligned}$$

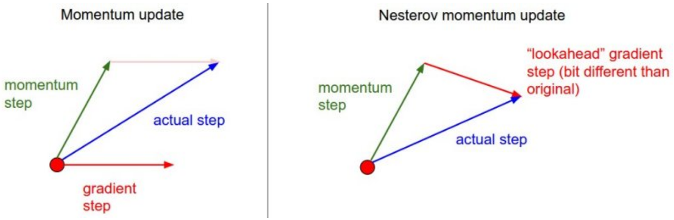


Successive gradients pointing in same/different directions increase/decrease velocity.
Further geometric intuitions and detailed explanations:
`https://distill.pub/2017/momentum/`

NESTEROV ACCELERATED GRADIENT

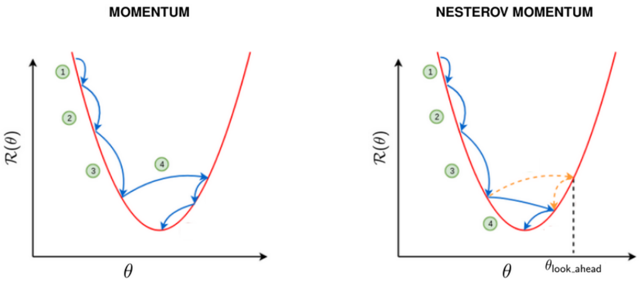
- Slightly modified version: **Nesterov accelerated gradient**
- Stronger theoretical convergence guarantees for convex functions
- Avoid moving back and forth near optima

$$\boldsymbol{\nu}^{[t+1]} = \varphi \boldsymbol{\nu}^{[t]} - \alpha \nabla f(\mathbf{x}^{[t]} + \varphi \boldsymbol{\nu}^{[t]})$$
$$\mathbf{x}^{[t+1]} = \mathbf{x}^{[t]} + \boldsymbol{\nu}^{[t+1]}$$



Nesterov momentum update evaluates gradient at the "look-ahead" position. (Source: `https://cs231n.github.io/neural-networks-3/`)

MOMENTUM VS. NESTEROV



GD with momentum (**left**) vs. GD with Nesterov momentum (**right**).
Near minima, momentum makes a large step due to gradient history.
Nesterov momentum "looks ahead" and reduces effect of gradient history.
(Source: Chandra, 2015)