Introduction to Machine Learning
[Fall 2022]

Diving Deeper into Gradients

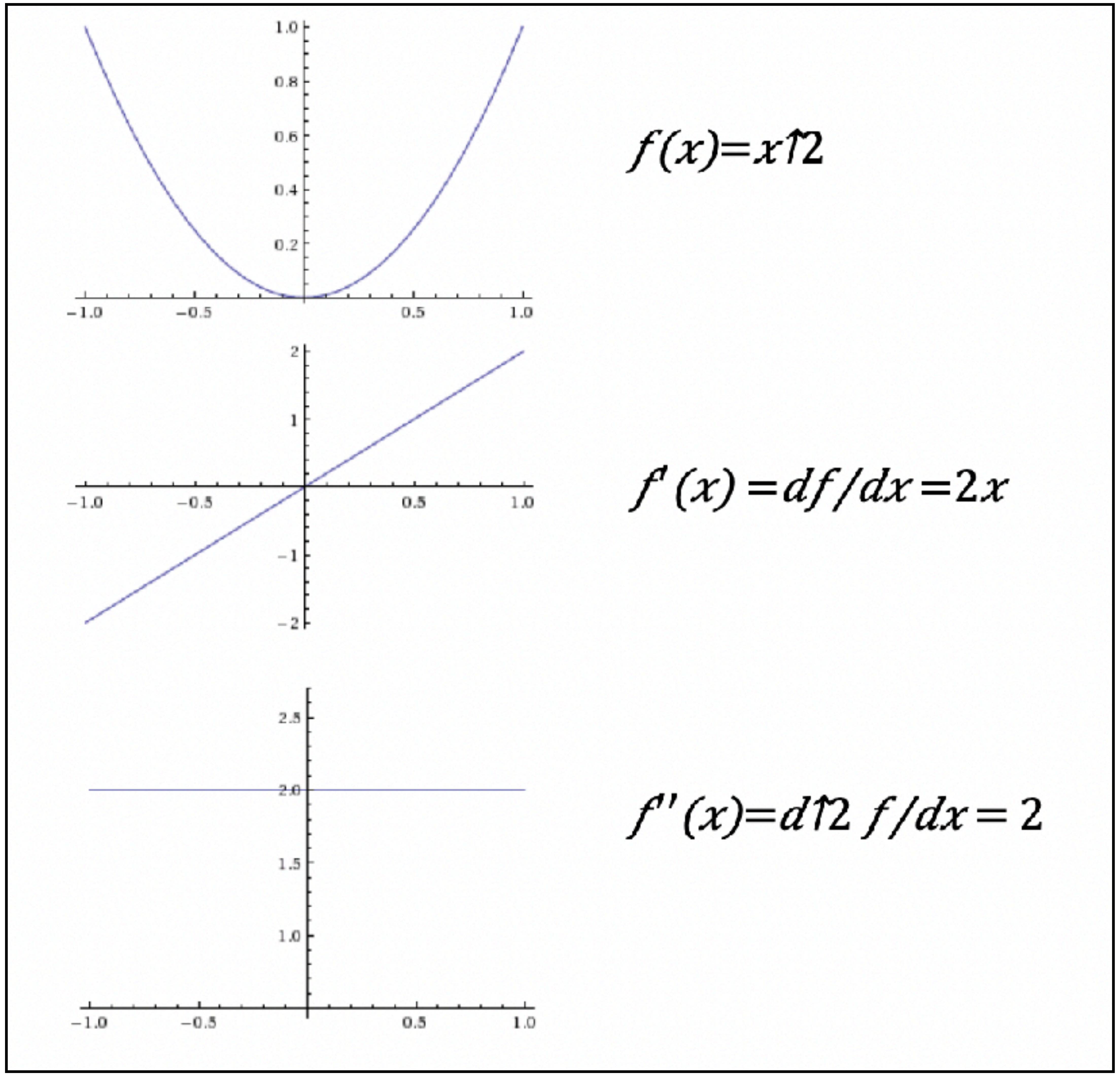
October 18, 2022

Lerrel Pinto

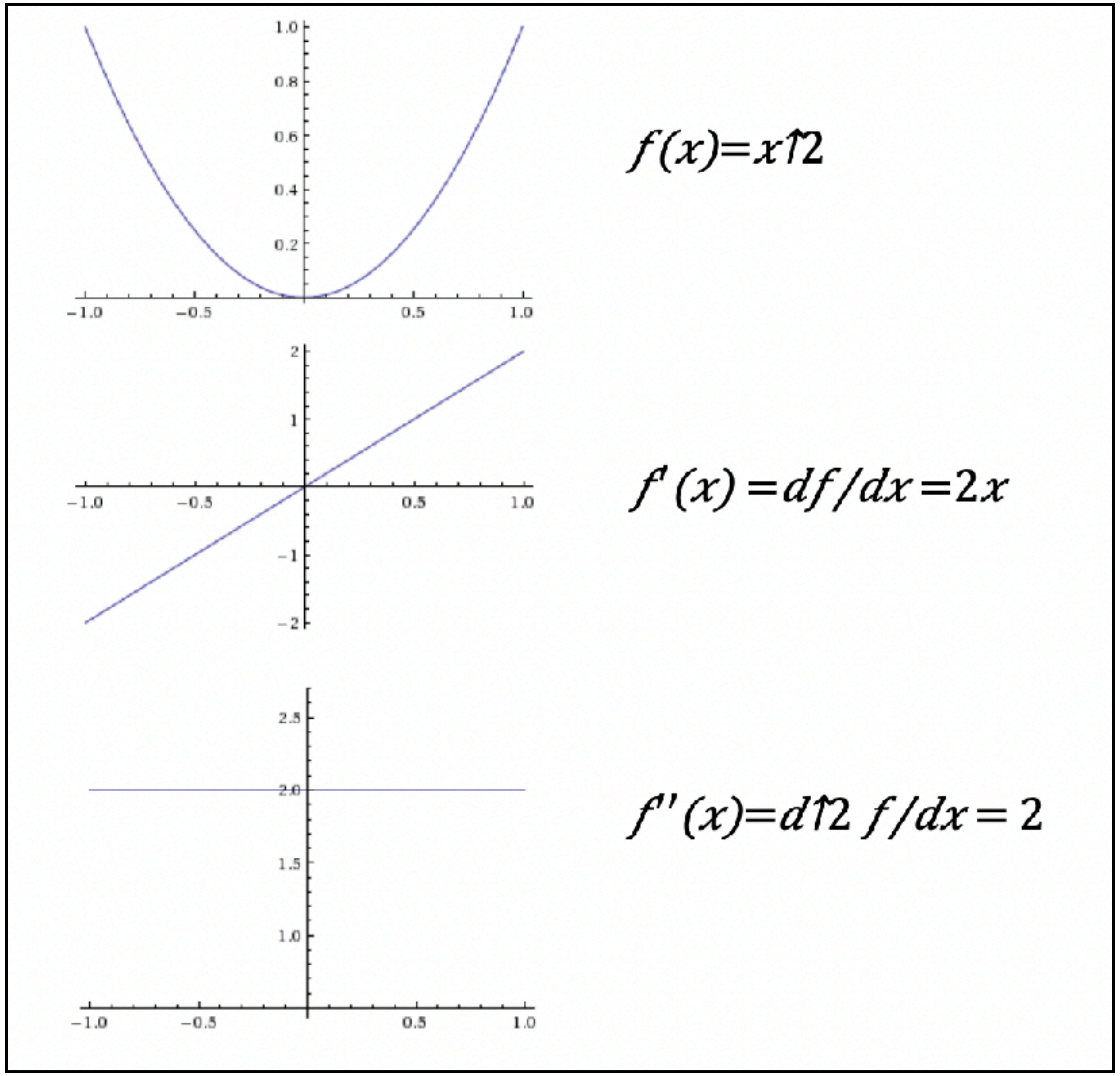
Topics for today

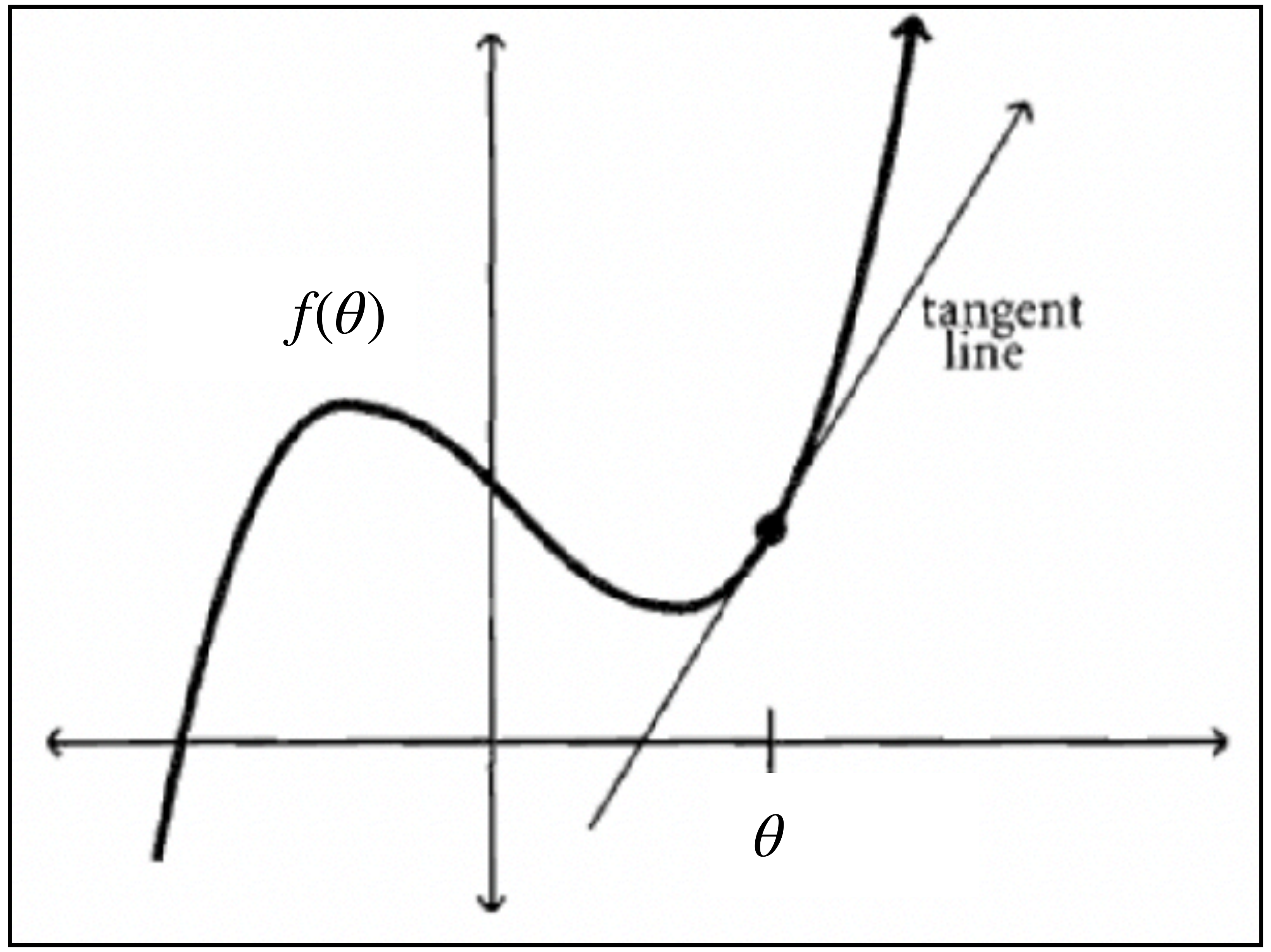
- Fundamentals for gradients
- Gradients with high-dimensional inputs
- Examples of gradient descent

Recap: Derivatives



Recap: Derivatives





Gradient Descent Algorithm

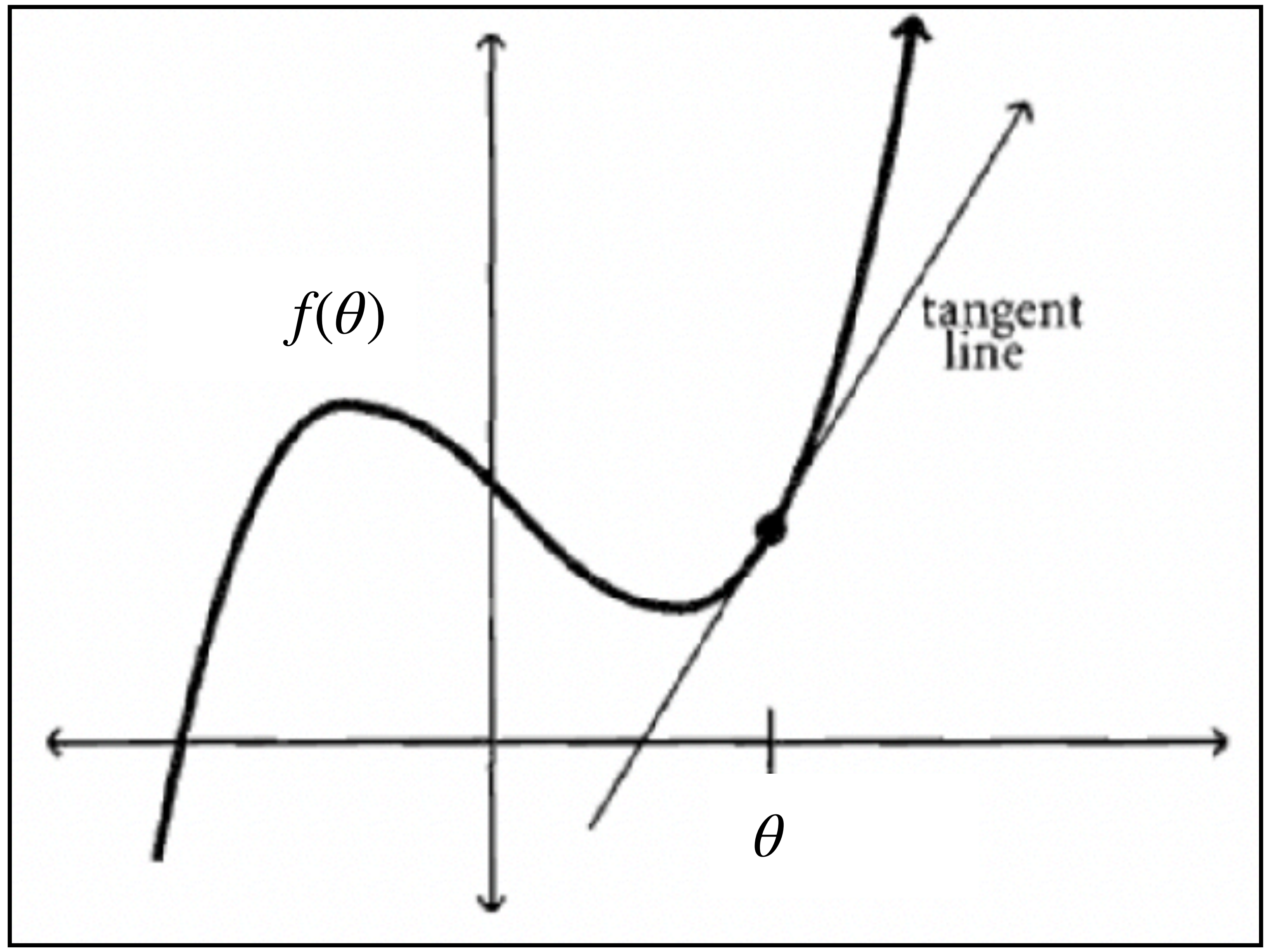
- Given: cost / loss/ objective function $f(\theta)$.

- Goal: find θ^* such that $f(\theta^*) = \min_{\theta} f(\theta)$.

Gradient Descent Algorithm

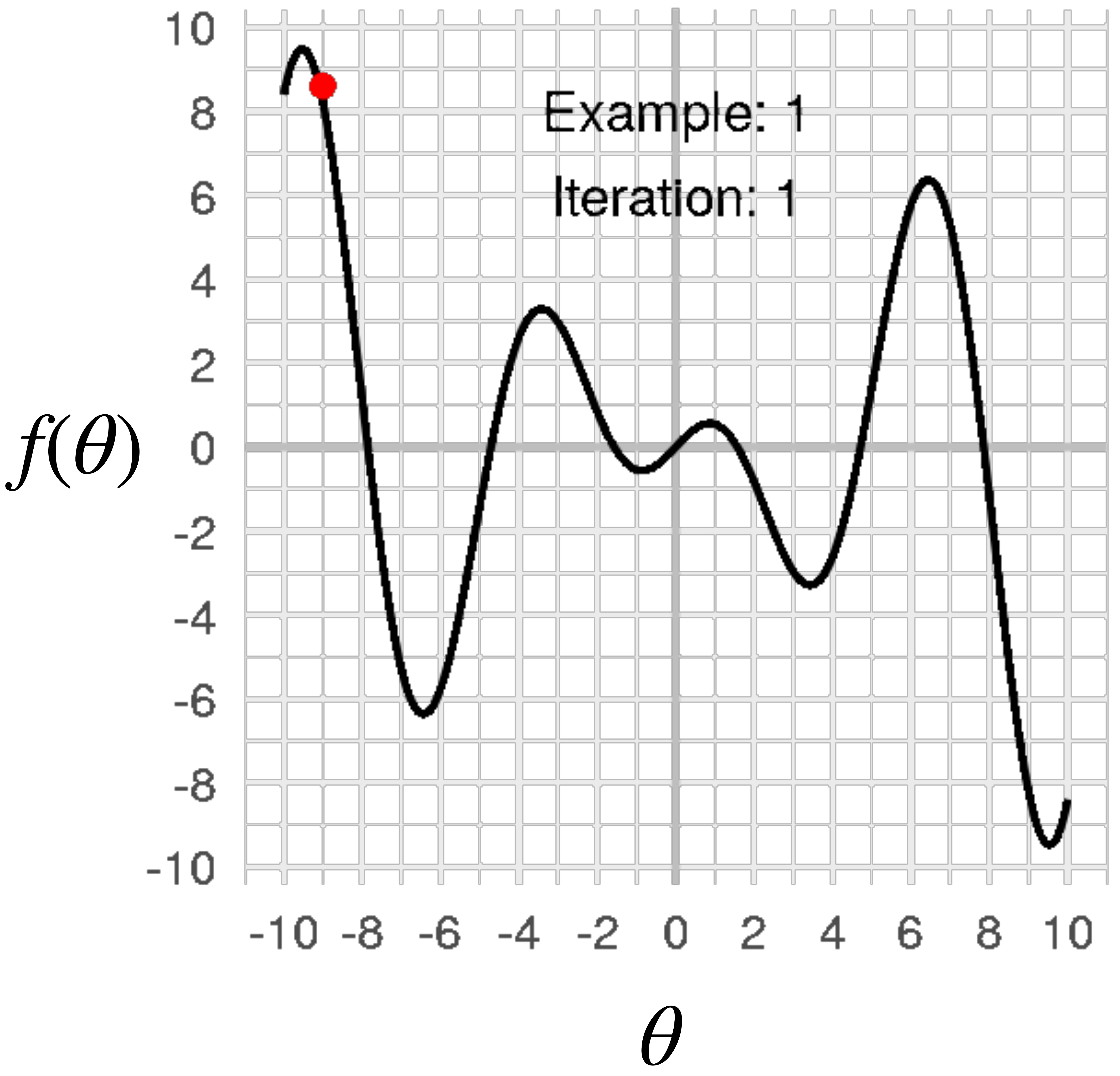
- Given: cost / loss/ objective function $f(\theta)$.
- Goal: find θ^* such that $f(\theta^*) = \min_{\theta} f(\theta)$.
- Gradient descent solution:
  - Start from initial guess θ^0 and learning rate α
  - Update $\theta^{i+1} \leftarrow \theta^i - \alpha \frac{df(\theta)}{d\theta}$
  - Repeat until change in θ is small, or maximum number of steps reached.

Gradient Descent Algorithm



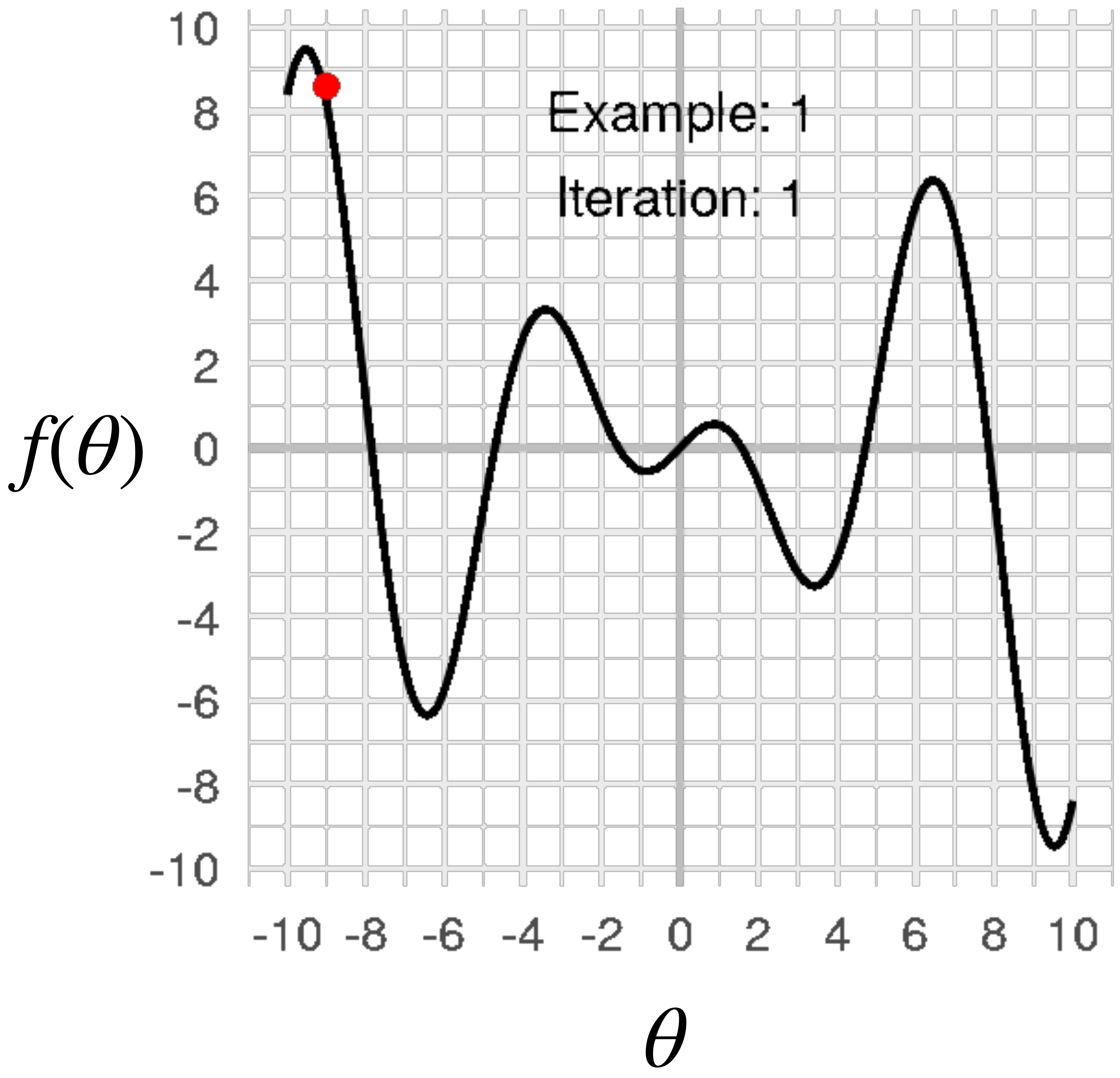
- Given: cost / loss/ objective function $f(\theta)$.
- Goal: find $\theta*$ such that $f(\theta*) = \min_{\theta} f(\theta)$.
- Gradient descent solution:
  - Start from initial guess θ^0 and learning rate α
  - Update $\theta^{i+1} \leftarrow \theta^i - \alpha \frac{df(\theta)}{d\theta}$
  - Repeat until change in θ is small, or maximum number of steps reached.

Gradient Descent Algorithm



Credits: Charles Bordet

Gradient Descent Algorithm

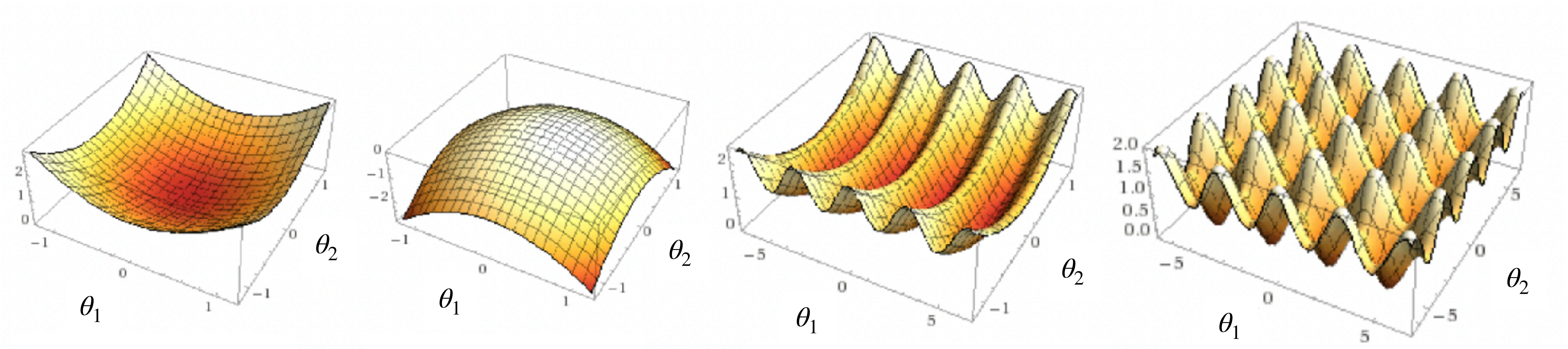


Credits: Charles Bordet

Interactive demo:

https://uclaacm.github.io/gradient-descent-visualiser/

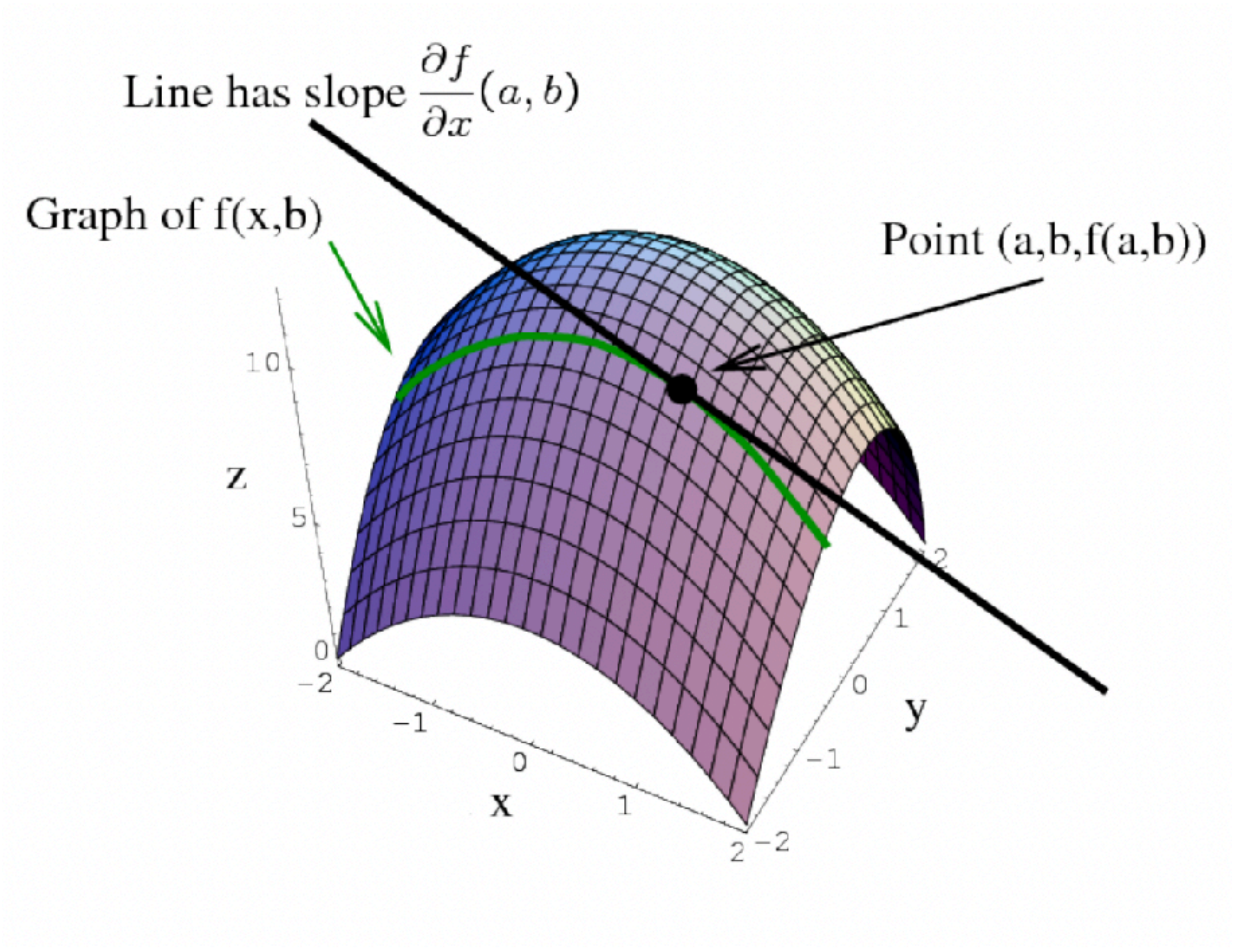
What happens when number of dimensions is high?



$f(\theta_1, \theta_2) = \theta_1^2 + \theta_2^2$ $\quad f(\theta_1, \theta_2) = -\theta_1^2 - \theta_2^2$ $\quad f(\theta_1, \theta_2) = cos(\theta_1)^2 + \theta_2^2$ $\quad f(\theta_1, \theta_2) = cos(\theta_1)^2 + cos(\theta_2)^2$

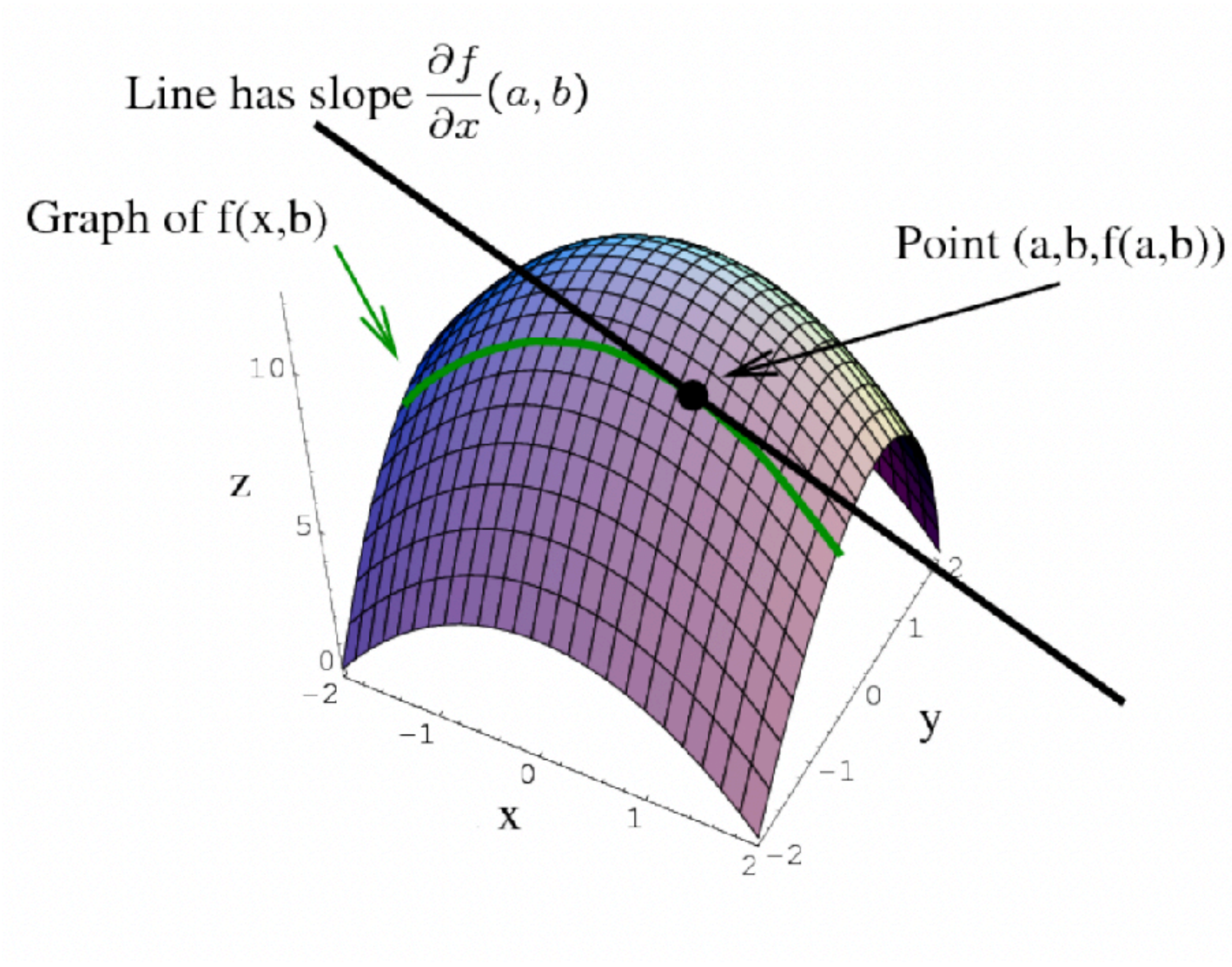
Credits: Michail Michailidis & Patrick Maiden

What happens when number of dimensions is high?



Credits: Michail Michailidis & Patrick Maiden

What happens when number of dimensions is high?



$$f(x, y) = 9 - x^2 - y^2$$

$$f(x, y) = 9 - x^2 - c^2$$

$$\frac{\partial f(x, y)}{\partial x} = -2x$$

$$f(x, y) = 9 - c^2 - y^2$$

$$\frac{\partial f(x, y)}{\partial y} = -2y$$

Gradient vector:

$$\nabla f(x, y) = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial x})$$

$$\nabla f(x, y) = (-2x, -2y)$$

Credits: Michail Michailidis & Patrick Maiden

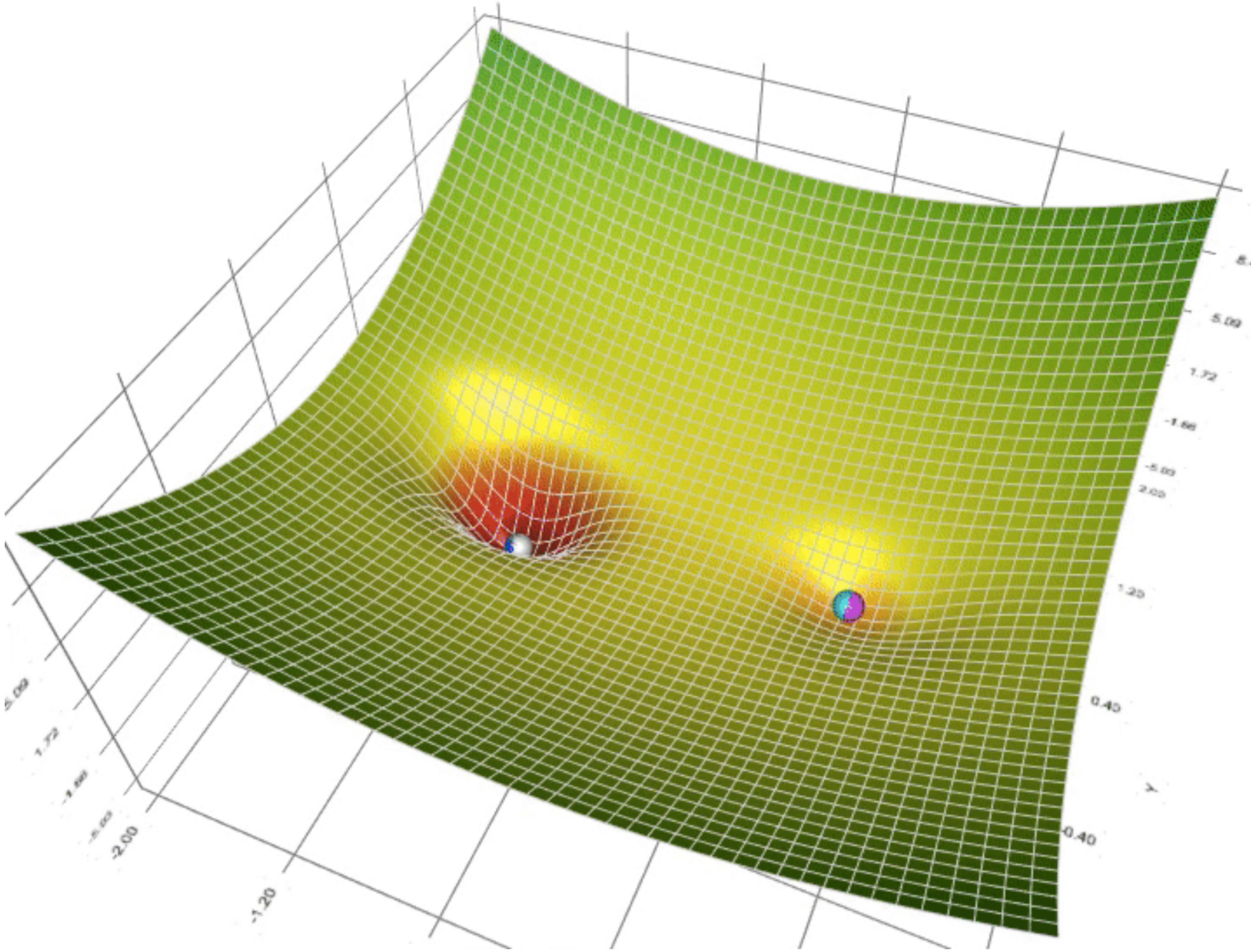
Gradient Descent Algorithm with multiple params

- Given: cost / loss/ objective function $f(\overrightarrow{\theta})$. Where $\overrightarrow{\theta} \in \mathbb{R}^d$.
- Goal: find $\overrightarrow{\theta}*$ such that $f(\overrightarrow{\theta}*) = \min_{\overrightarrow{\theta}} f(\overrightarrow{\theta})$.

Gradient Descent Algorithm with multiple params

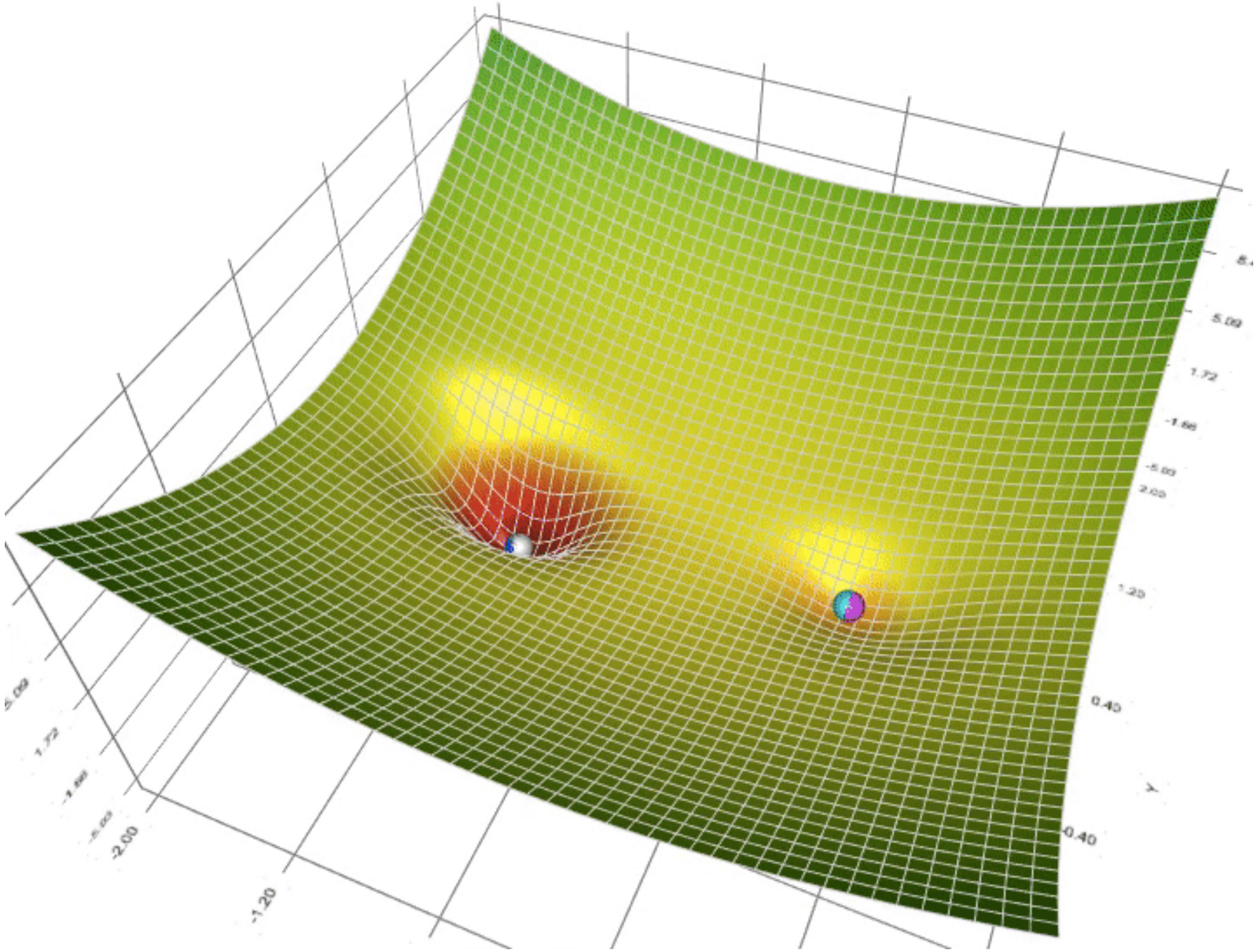
- Given: cost / loss/ objective function $f(\overrightarrow{\theta})$. Where $\overrightarrow{\theta} \in \mathbb{R}^d$.
- Goal: find $\overrightarrow{\theta}*$ such that $f(\overrightarrow{\theta}*) = \min_{\overrightarrow{\theta}} f(\overrightarrow{\theta})$.
- Gradient descent solution:
    - Start from initial guess $\overrightarrow{\theta}^0$ and learning rate α
    - Update $\overrightarrow{\theta}^{i+1} \leftarrow \overrightarrow{\theta}^i - \alpha \nabla f(\overrightarrow{\theta})$
    - Repeat until change in θ is small, or maximum number of steps reached.

Gradient Descent Algorithm Visualization



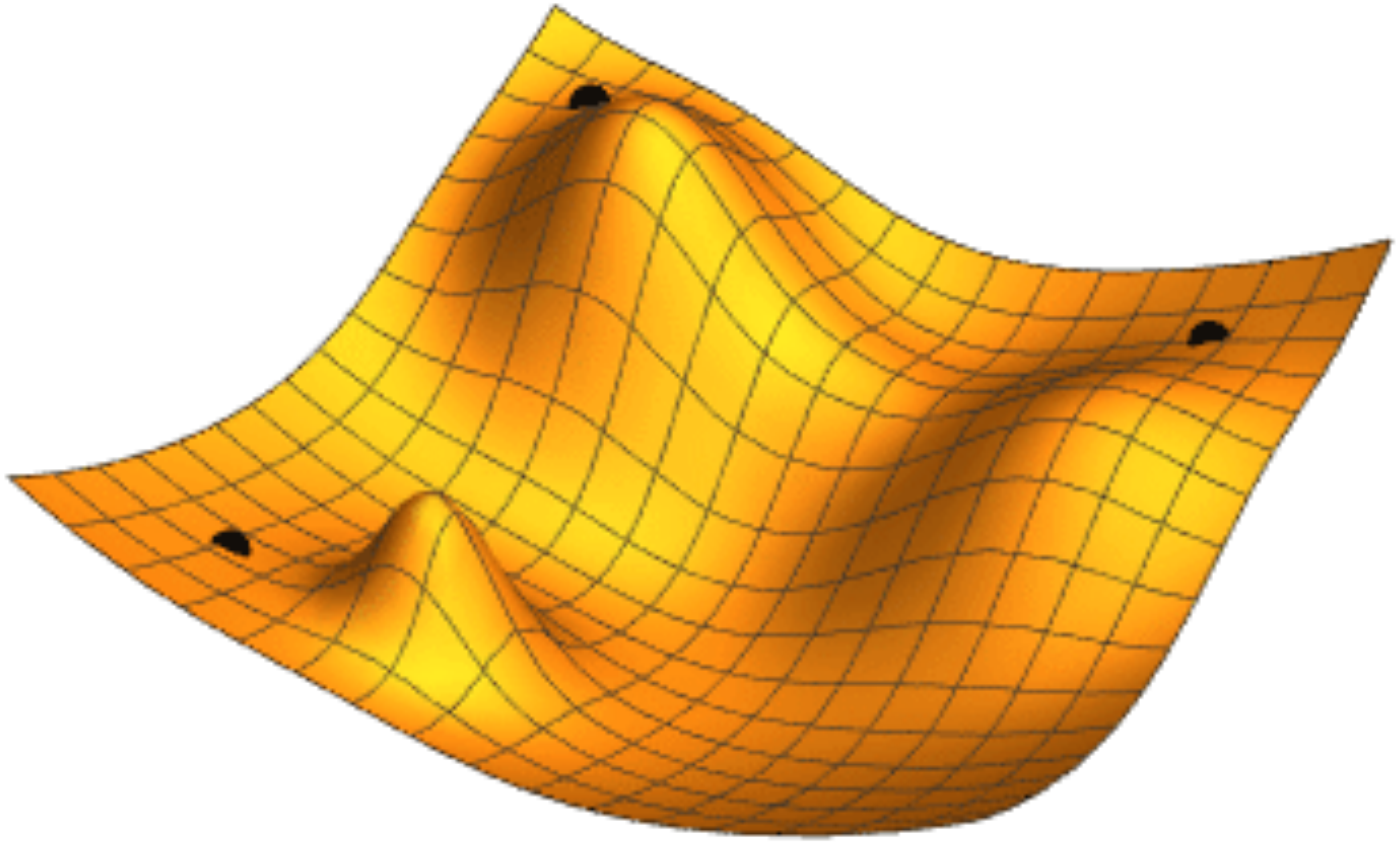
Credits: Lili Jiang

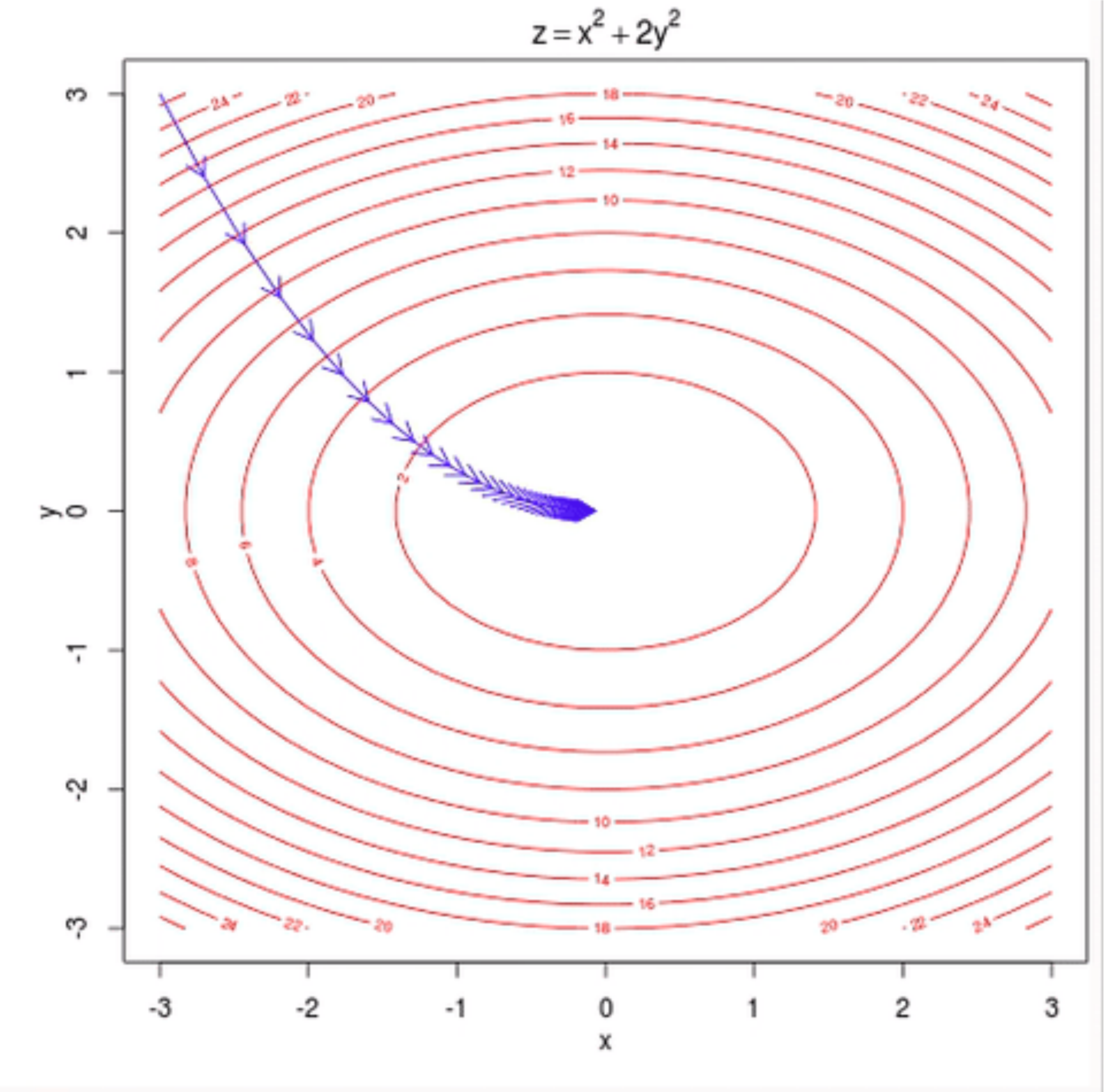
Gradient Descent Algorithm Visualization



Credits: Lili Jiang

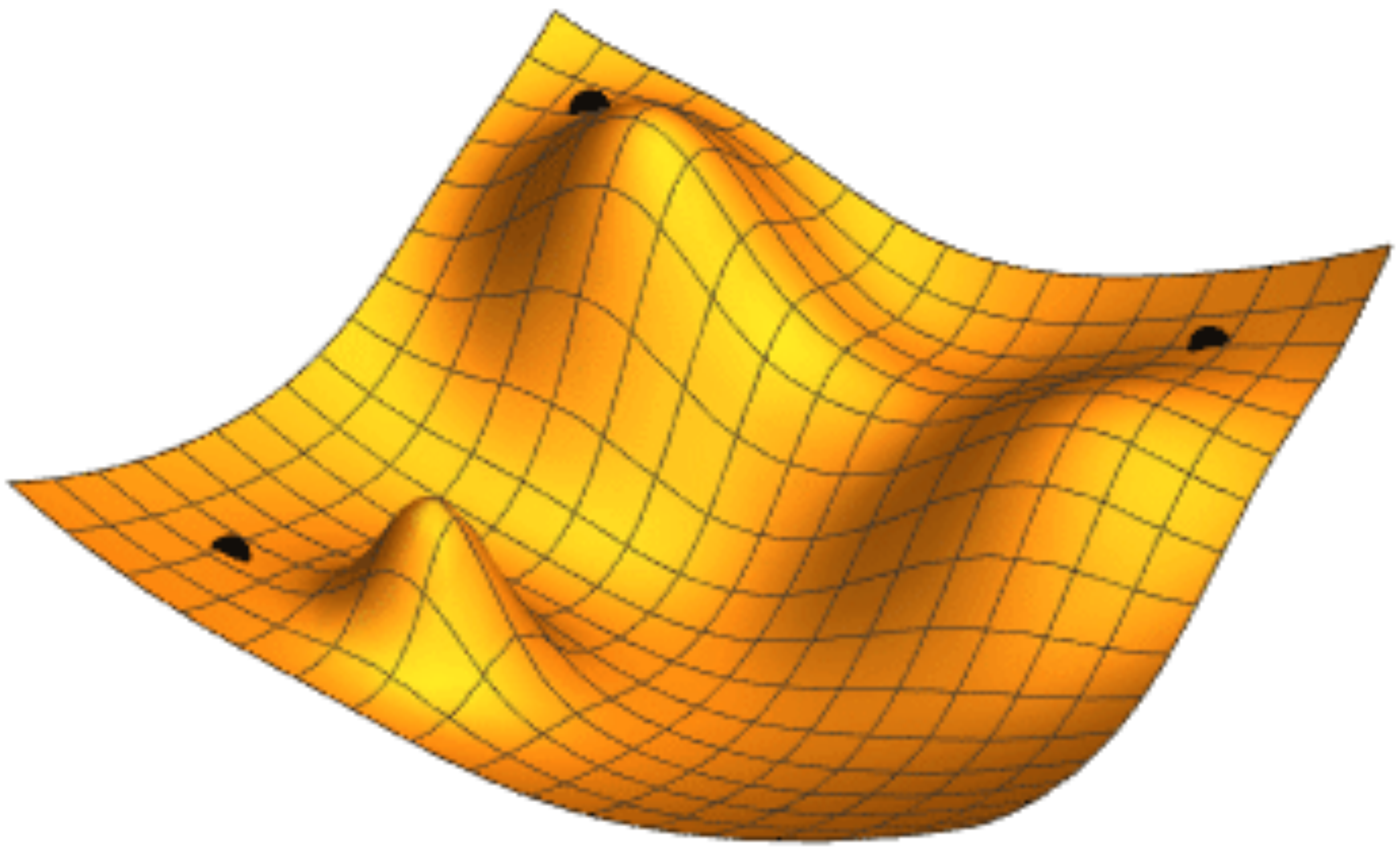
Gradient Descent Algorithm Visualization

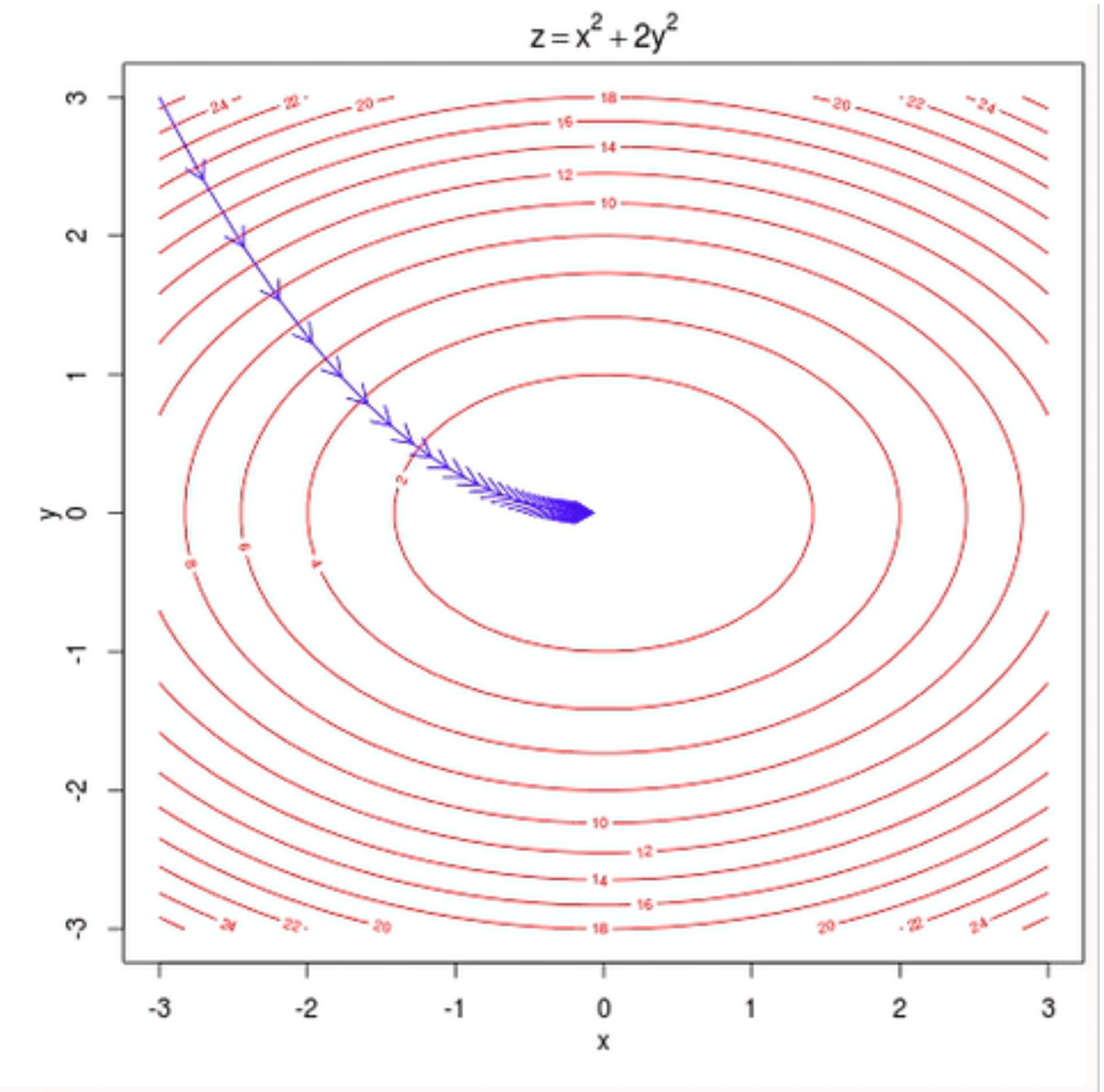




Credits: Wikimedia, Hoang Duong

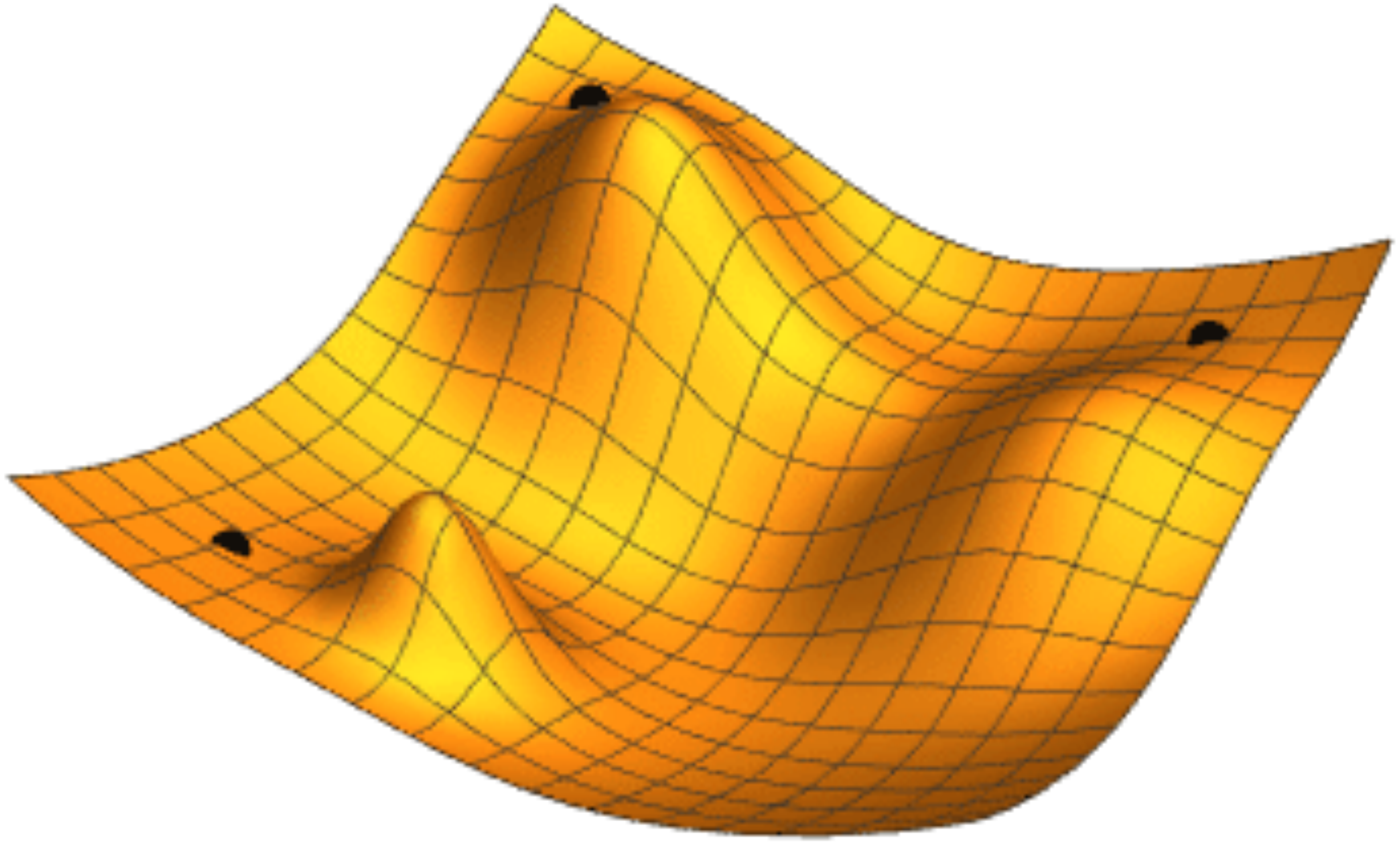
Gradient Descent Algorithm Visualization

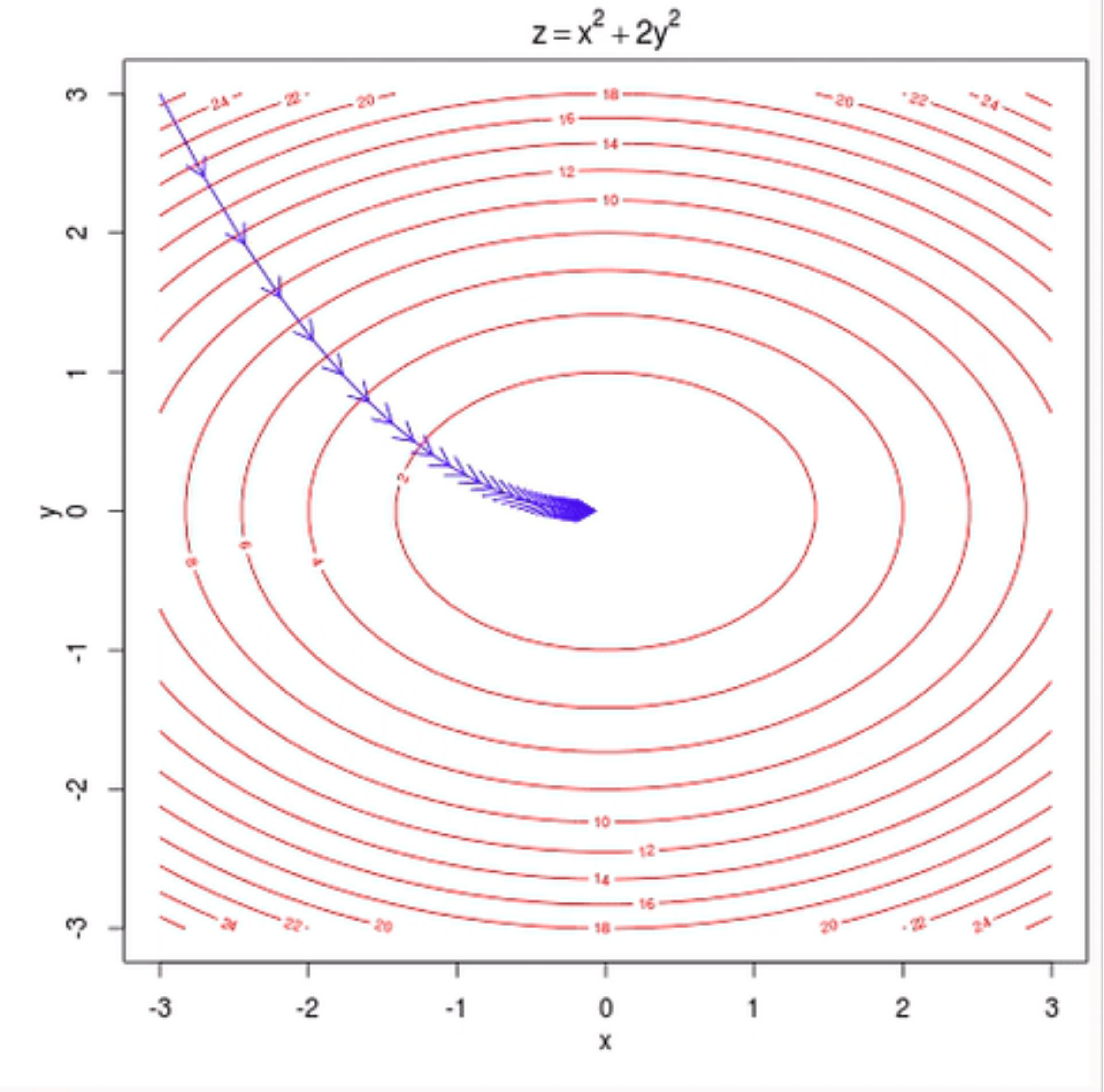




Credits: Wikimedia, Hoang Duong

Gradient Descent Algorithm Visualization





Credits: Wikimedia, Hoang Duong

Linear Regression with Gradient Descent

- Input data: $X \in \mathbb{R}^{d \times n}$, $Y \in \mathbb{R}^n$, where $(\vec{x} \in \mathbb{R}^d, y \in \mathbb{R}^1)$ corresponds to a data point.
  - $n \rightarrow$ # of data points, $d \rightarrow$ # of features / input dim.
- Goal: to find $\vec{w} \in \mathbb{R}^d$ such that $< \vec{w}, \vec{x} > = y$
  - Minimize $||X^\top \vec{w} - Y||^2$
- Analytic Solution: $\vec{w} = (XX^\top)^{-1}XY$

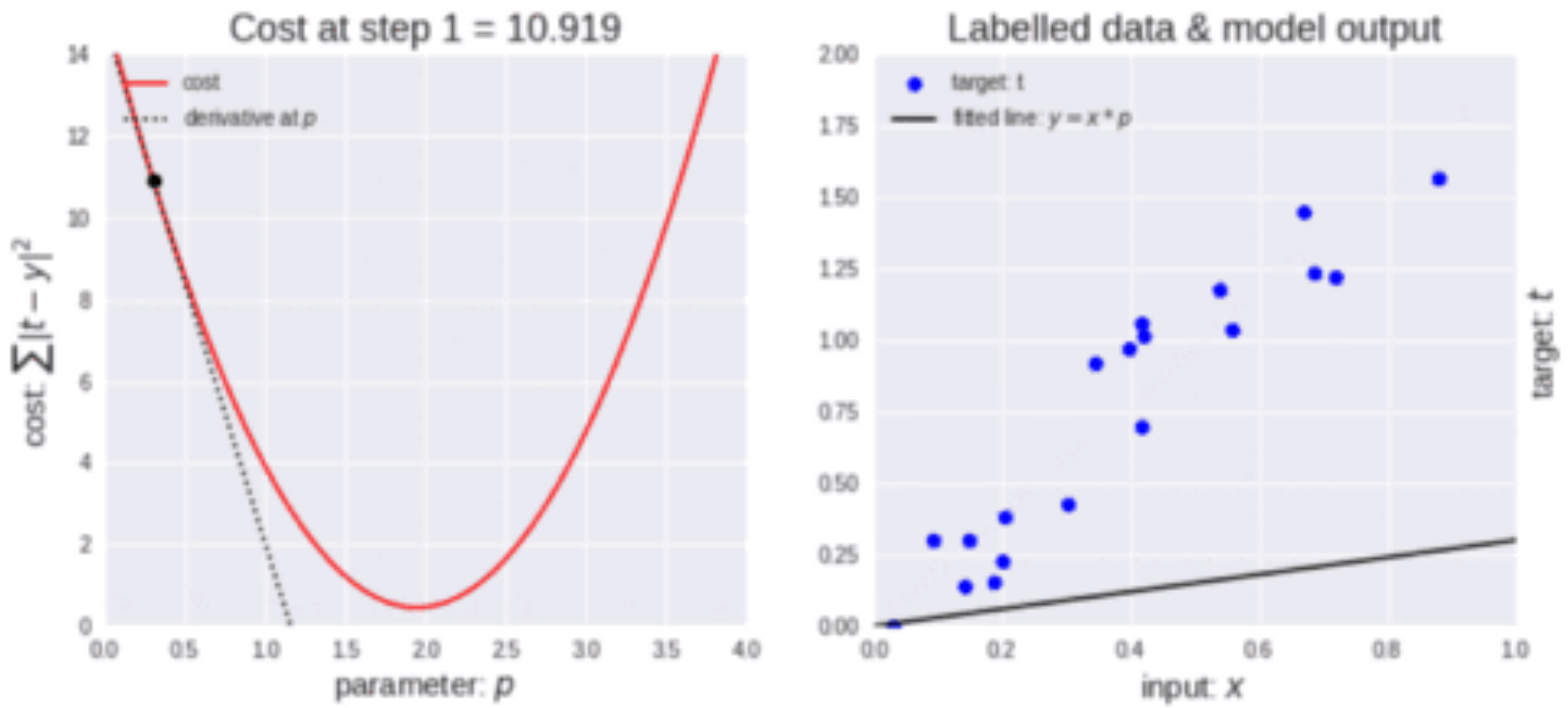
Linear Regression with Gradient Descent

- Input data: $X \in \mathbb{R}^{d \times n}$, $Y \in \mathbb{R}^{n}$, where $(\vec{x} \in \mathbb{R}^{d}, y \in \mathbb{R}^{1})$ corresponds to a data point.
    - $n \rightarrow$ # of data points, $d \rightarrow$ # of features / input dim.
- Goal: to find $\vec{w} \in \mathbb{R}^{d}$ such that $< \vec{w}, \vec{x} > = y$
    - Minimize $\|X^{\top}\vec{w} - Y\|^{2}$
- Loss function $f(\vec{w}) = \|X^{\top}\vec{w} - Y\|^{2} = (X^{\top}\vec{w} - Y)^{\top}(X^{\top}\vec{w} - Y)$

Linear Regression with Gradient Descent

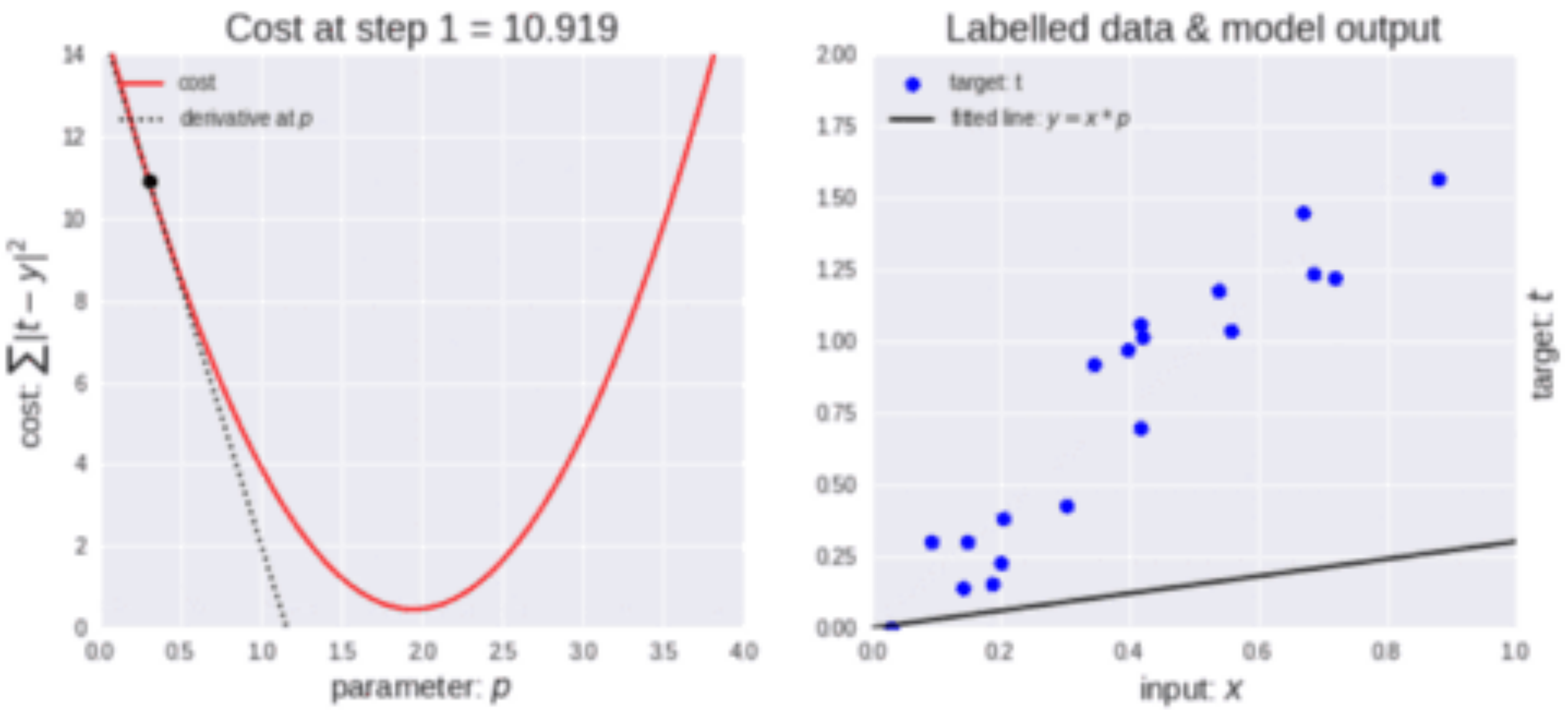
- Loss function $f(\overrightarrow{w}) = ||X^{\mathsf{T}}\overrightarrow{w} - Y||^2 = (X^{\mathsf{T}}\overrightarrow{w} - Y)^{\mathsf{T}}(X^{\mathsf{T}}\overrightarrow{w} - Y)$
    - Set $f(\overrightarrow{w}) = \sum_{i=1}^{n} (x^i . \overrightarrow{w} - y^i)^2$
    - Compute $\frac{\partial f}{\partial w_k} = \sum_{i=1}^{n} 2(\overrightarrow{w}^T x^i - y^i)x_k^i$
    - Compute $\nabla f = [\frac{\partial f}{\partial w_1}, \cdots, \frac{\partial f}{\partial w_k}, \cdots, \frac{\partial f}{\partial w_d}]$
    - Use gradient descent

Linear Regression with Gradient Descent



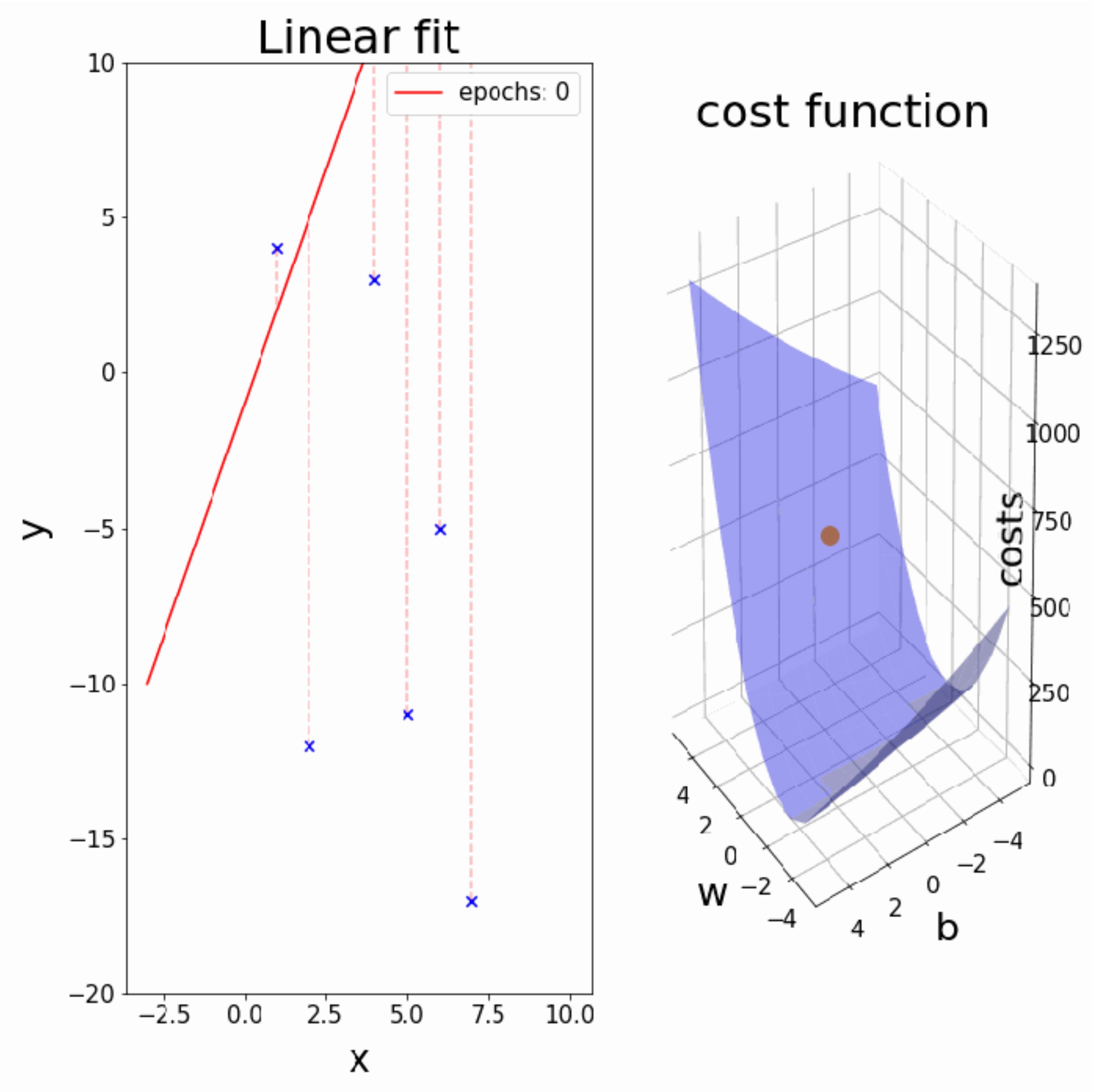
Credits: https://www.kdnuggets.com/

Linear Regression with Gradient Descent



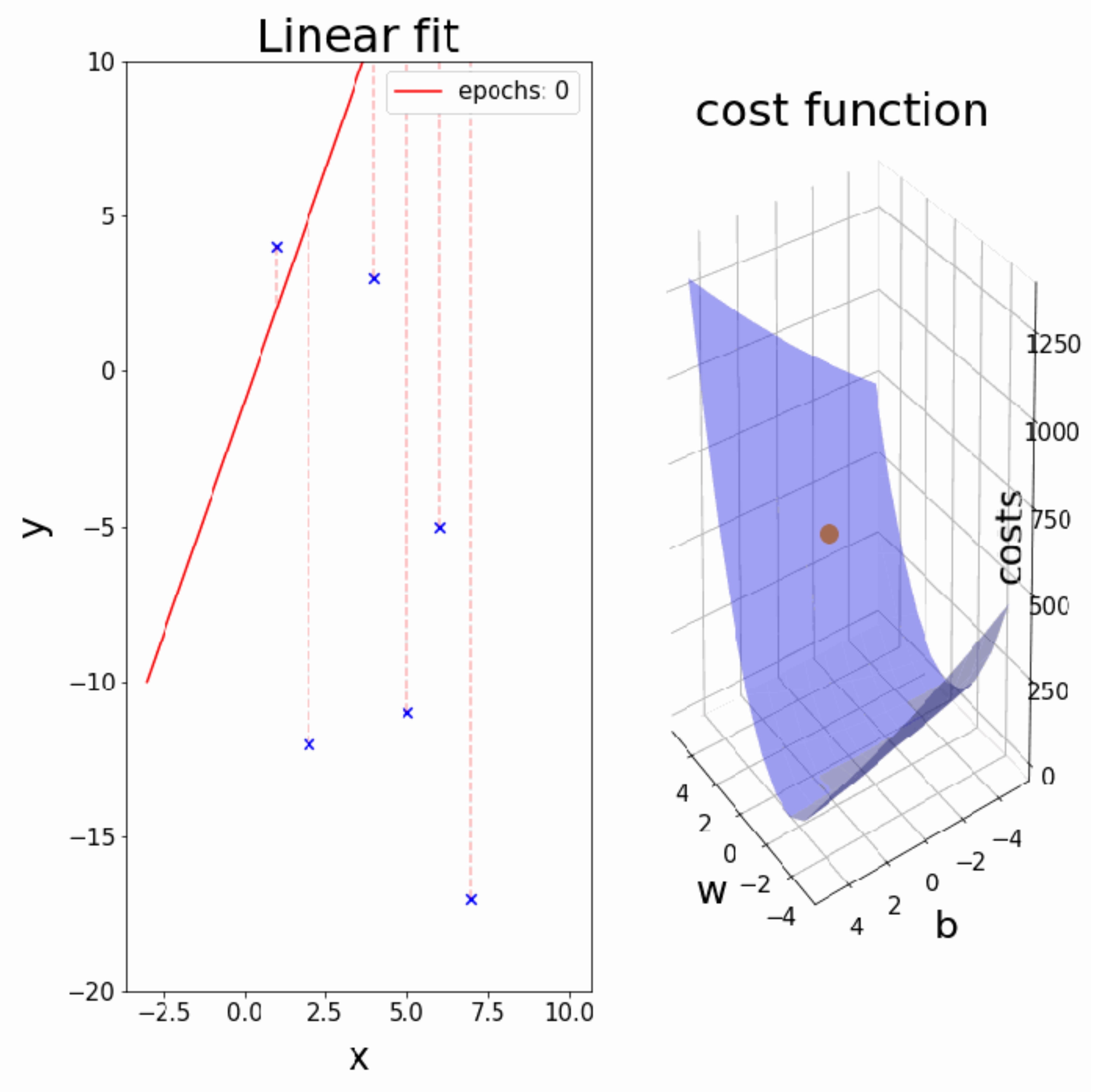
Credits: https://www.kdnuggets.com/

Linear Regression with Gradient Descent



Credits: Tobias Roeschl

Linear Regression with Gradient Descent

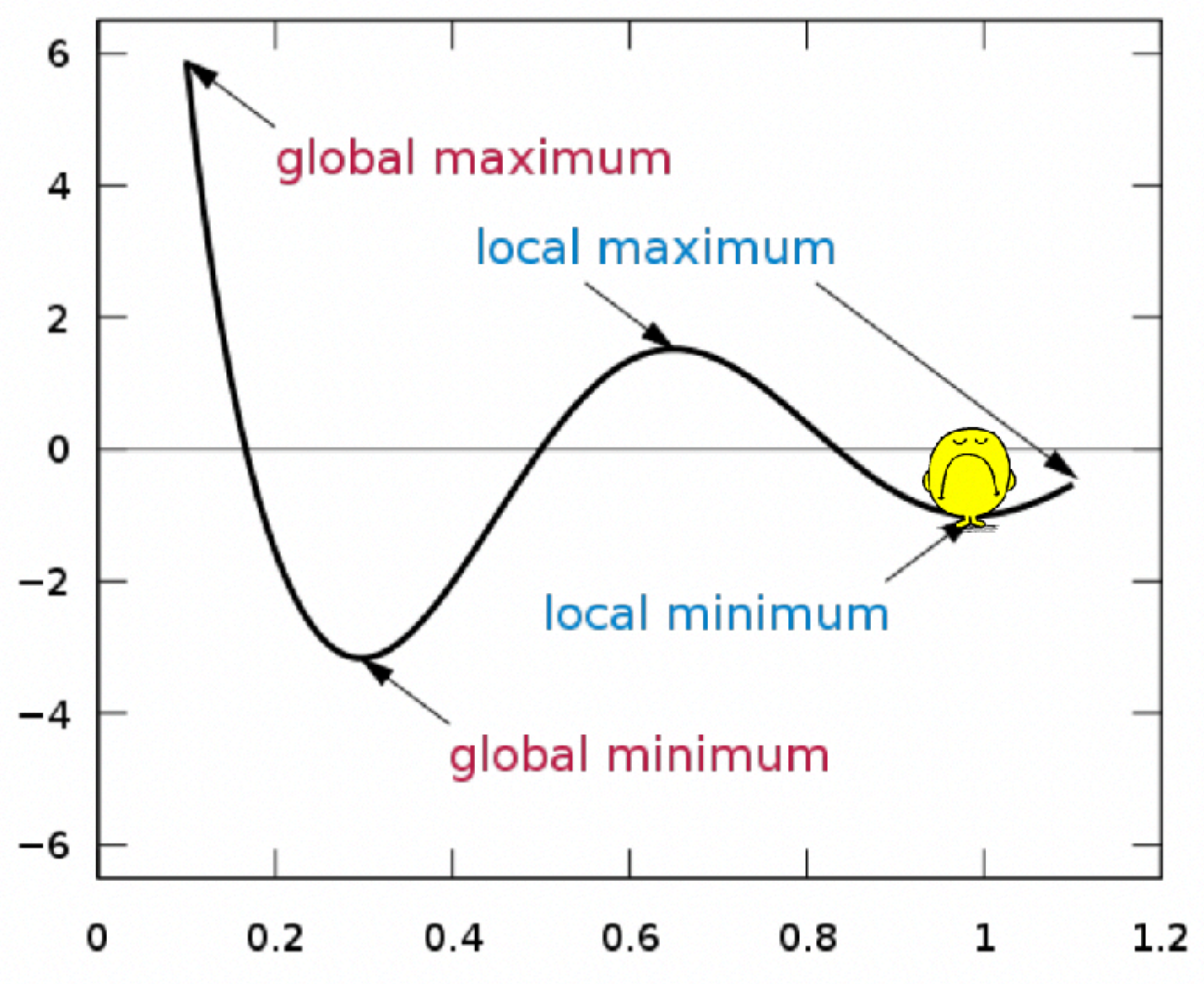


Credits: Tobias Roeschl

Gradient Descent Algorithm for Machine Learning

- Given: cost / loss/ objective function $f(\vec{\theta}, D)$. Where $\vec{\theta} \in \mathbb{R}^d$.
- Goal: find $\vec{\theta}*$ such that $f(\vec{\theta}*, D) = \min_{\vec{\theta}} f(\vec{\theta}, D)$.
- Gradient descent solution:
  - Start from initial guess $\vec{\theta}^0$ and learning rate α
  - Update $\vec{\theta}^{i+1} \leftarrow \vec{\theta}^i - \alpha \nabla f(\vec{\theta}, D)$
  - Repeat until change in θ is small, or maximum number of steps reached.

Key issue: Local minima

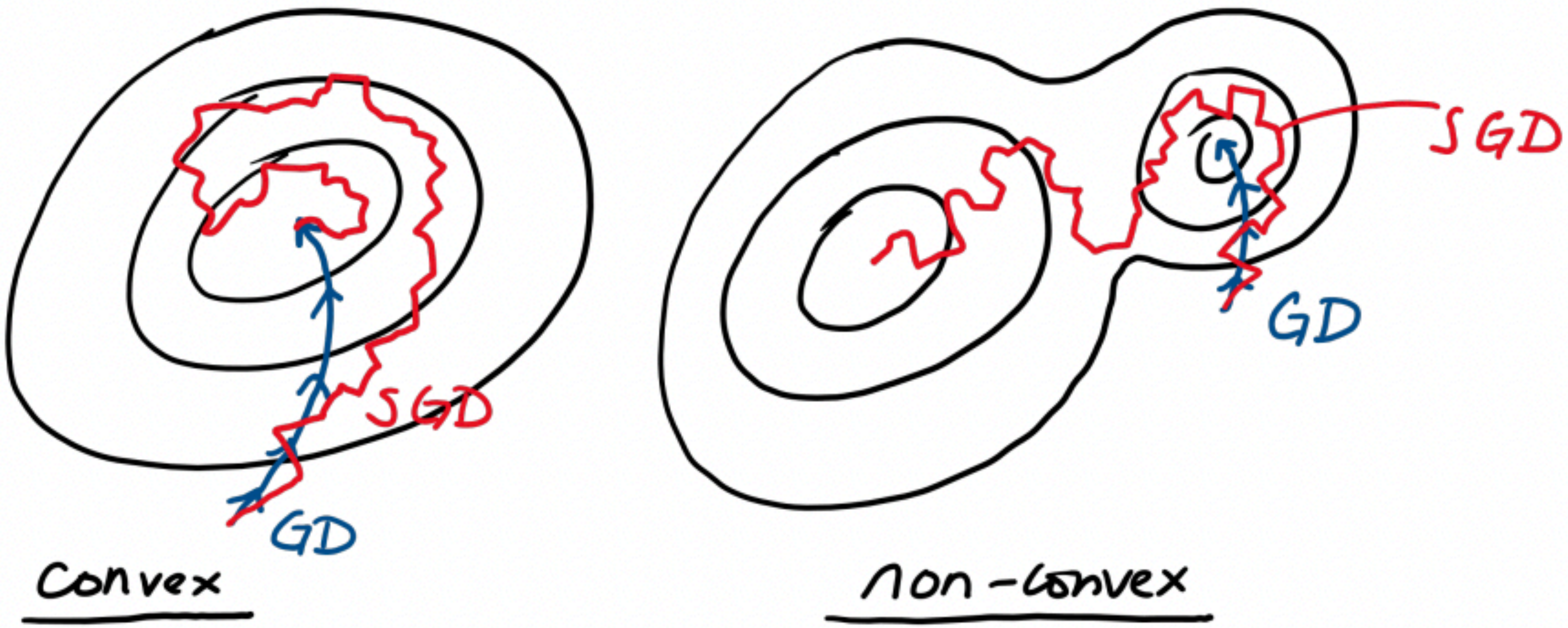


Credits: Michail Michailidis & Patrick Maiden

One solution: Stochastic Gradient Descent (SGD)

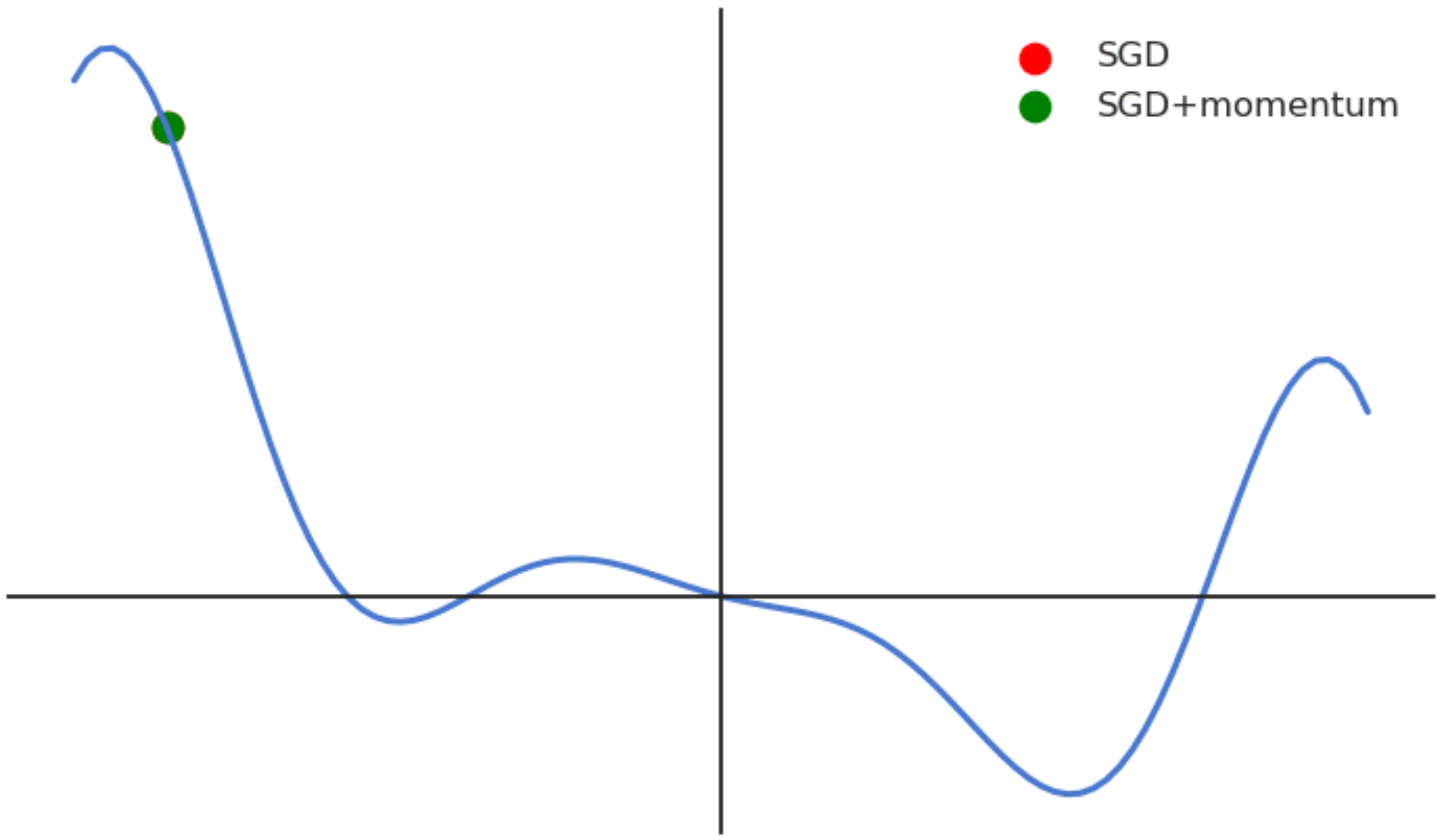
- Gradient descent solution:
  - Start from initial guess $\overrightarrow{\theta}^0$ and learning rate α
  - Update $\overrightarrow{\theta}^{i+1} \leftarrow \overrightarrow{\theta}^i - \alpha \nabla f(\overrightarrow{\theta}, D)$
- SGD:
  - Sample single or multiple datapoints $d \sim D$
  - Update $\overrightarrow{\theta}^{i+1} \leftarrow \overrightarrow{\theta}^i - \alpha \nabla f(\overrightarrow{\theta}, d)$

One solution: Stochastic Gradient Descent (SGD)



Credits: Stanley Chan

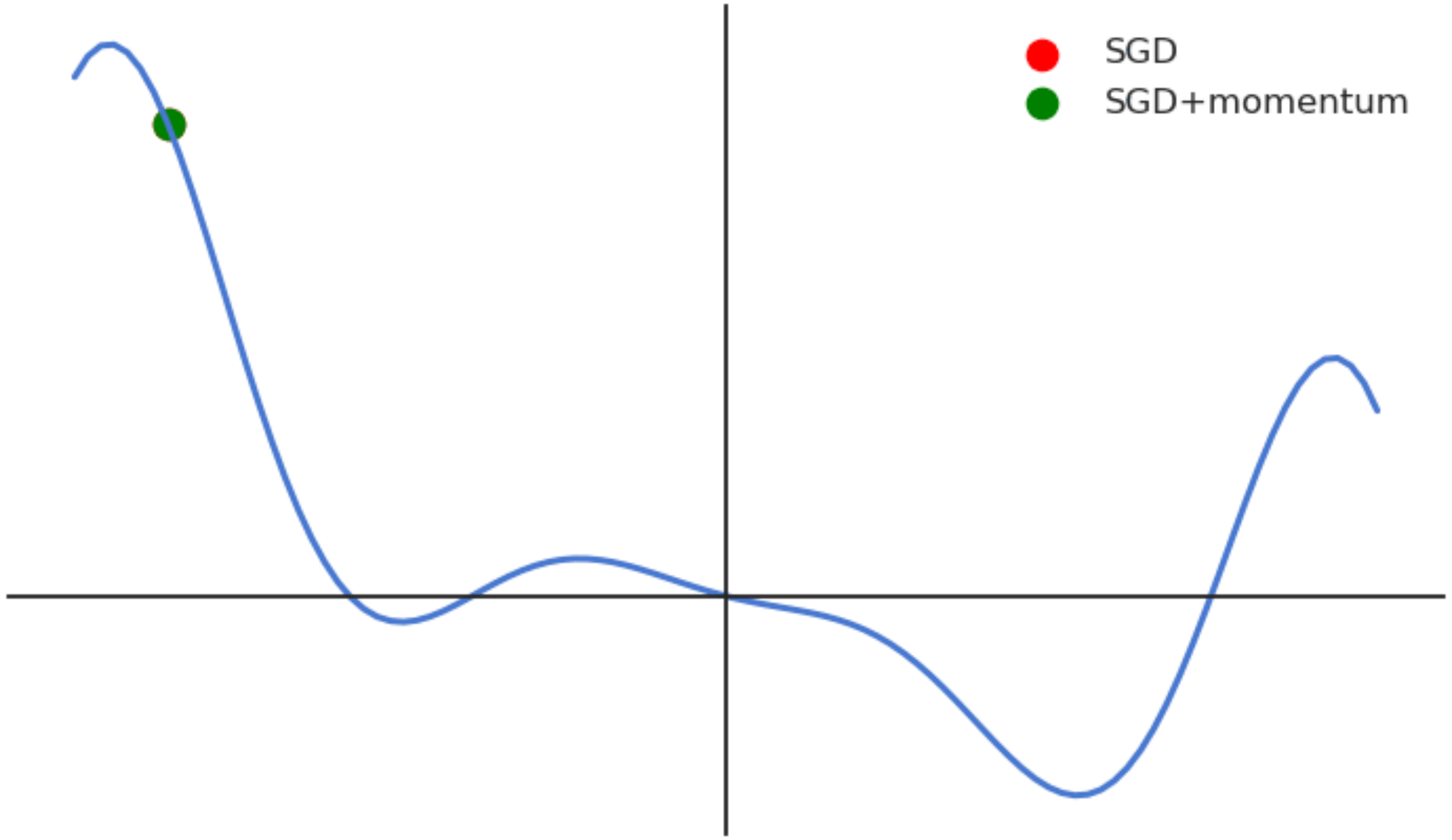
Another solution: Momentum



$$v^{t+1} = \beta v^t + \alpha \nabla f(\overrightarrow{w})$$

$$\overrightarrow{\theta}^{t+1} \leftarrow \overrightarrow{\theta}^{t} - v^{t+1}$$

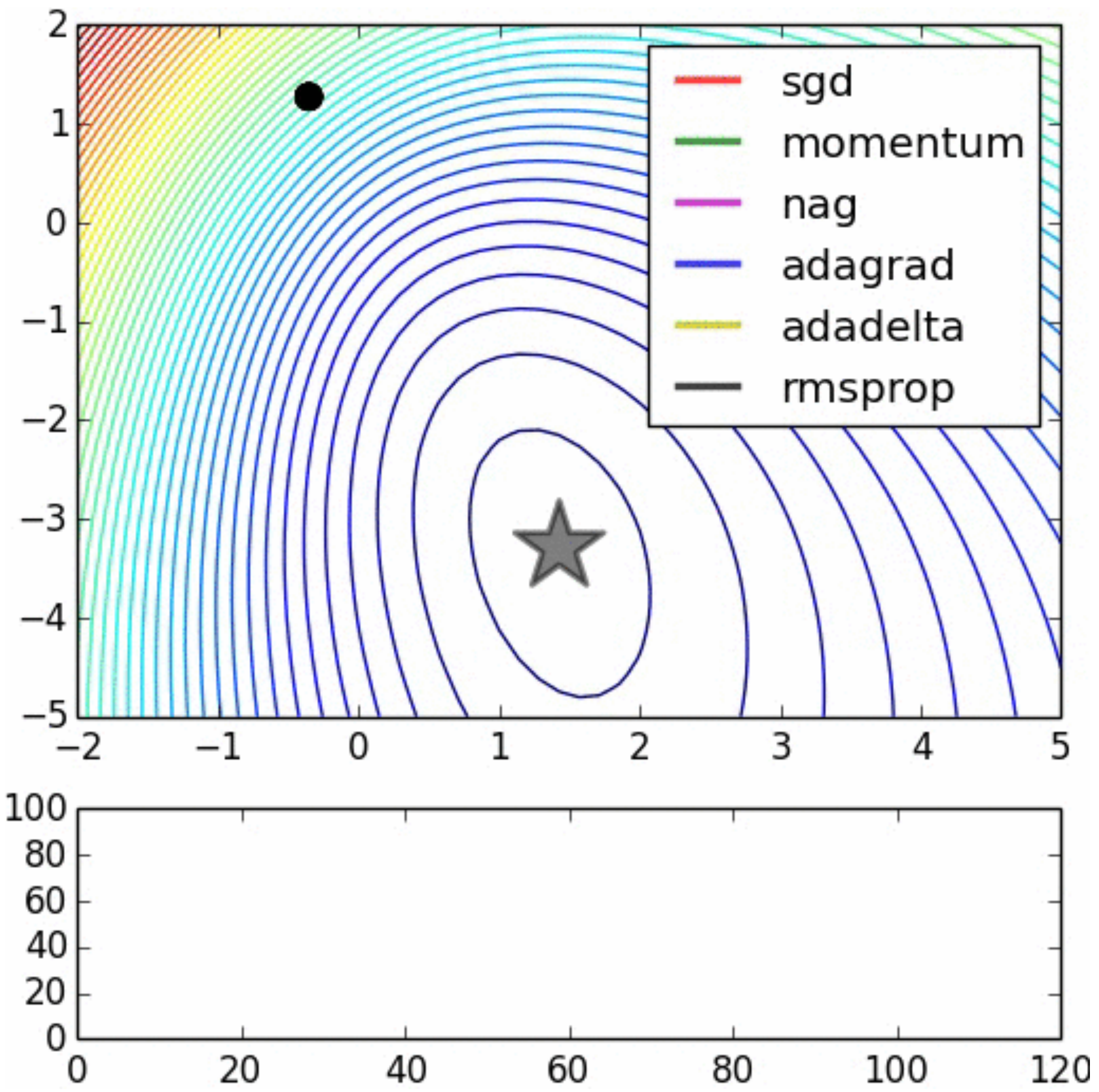
Another solution: Momentum



$$v^{t+1} = \beta v^t + \alpha \nabla f(\overrightarrow{w})$$

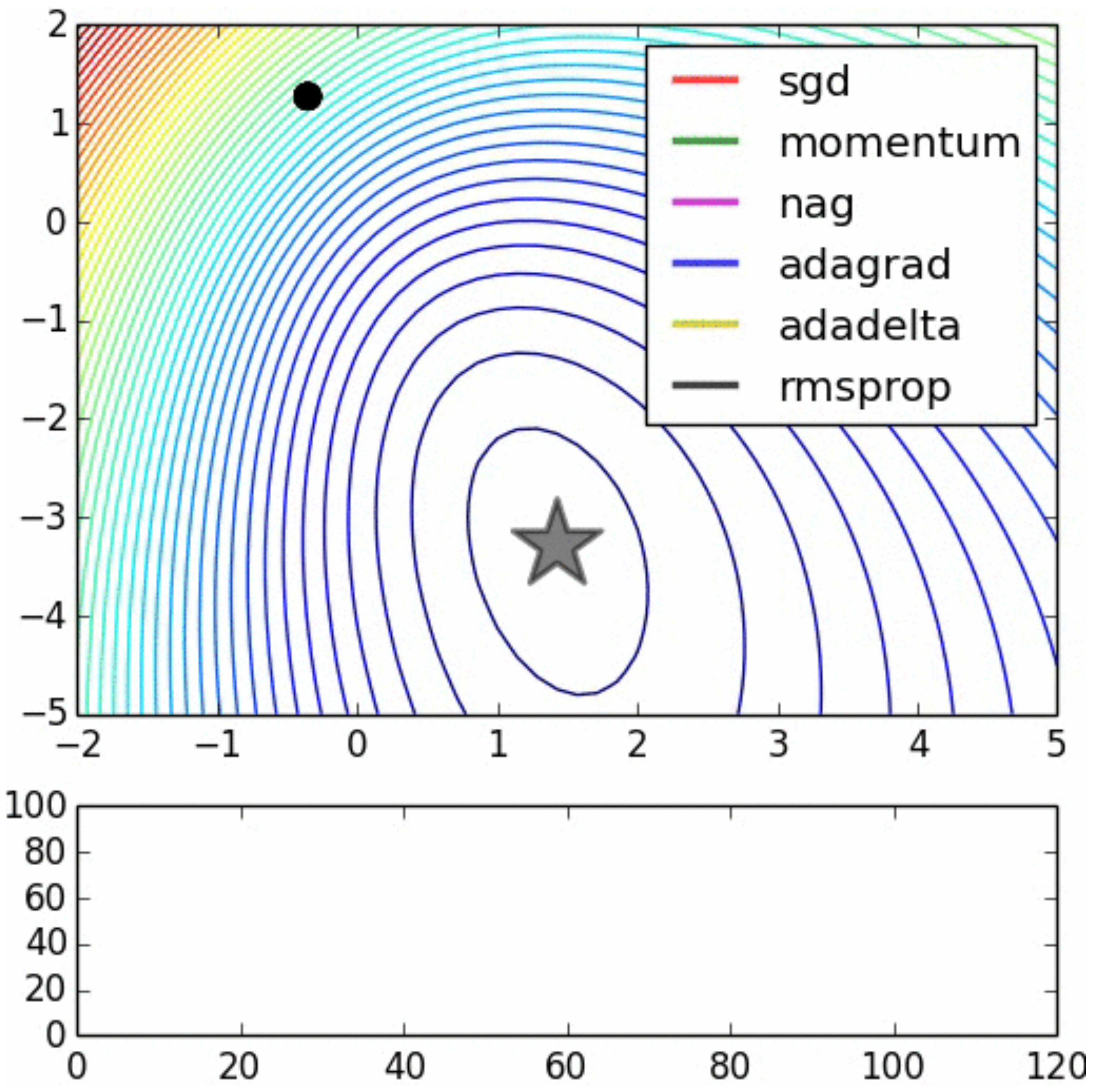
$$\overrightarrow{\theta}^{t+1} \leftarrow \overrightarrow{\theta}^t - v^{t+1}$$

More solutions have been proposed!



Credits: Alec Radford

More solutions have been proposed!



Credits: Alec Radford

Additional Reading

- Interactive tutorial: https://uclaacm.github.io/gradient-descent-visualiser/
- Book chapter: https://www.cs.utah.edu/~jeffp/IDABook/T6-GD.pdf
- SGD + variants: https://ruder.io/optimizing-gradient-descent

Questions?