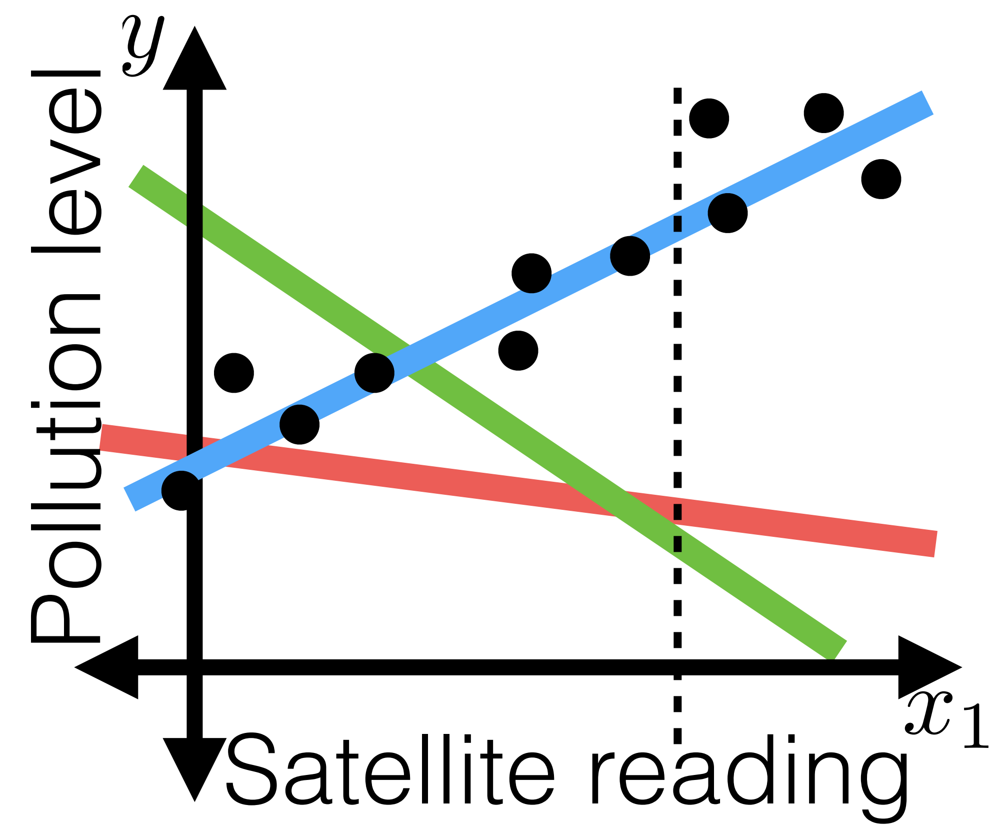
Gradient Descent

Prof. Tamara Broderick

Edited From 6.036 Fall21 Offering

Recall



- A general ML approach:
  - Collect data
  - Choose hypothesis class
  - Choose "good" hypothesis by minimizing training loss + regularizer
- Example: ridge regression

linear regression hypothesis

squared loss $L(g,a) = (g-a)^2$

e.g.

$$f(\Theta) = J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

squared-norm as regularizer

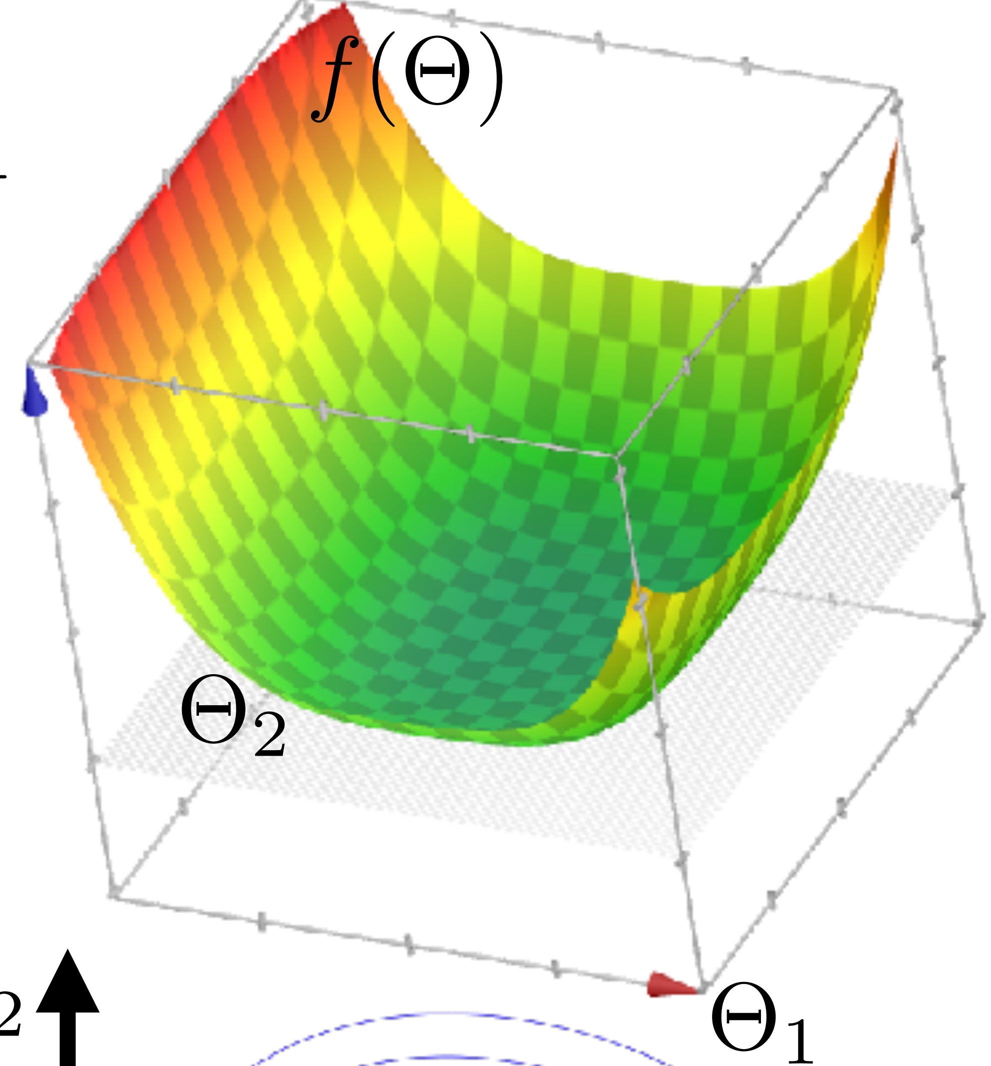
- "All models are wrong, but some are useful" -George Box
- Limitations of a closed-form solution for objective minimizer
  - Other hypotheses or loss or regularizer: maybe no closed-form solution, or difficult

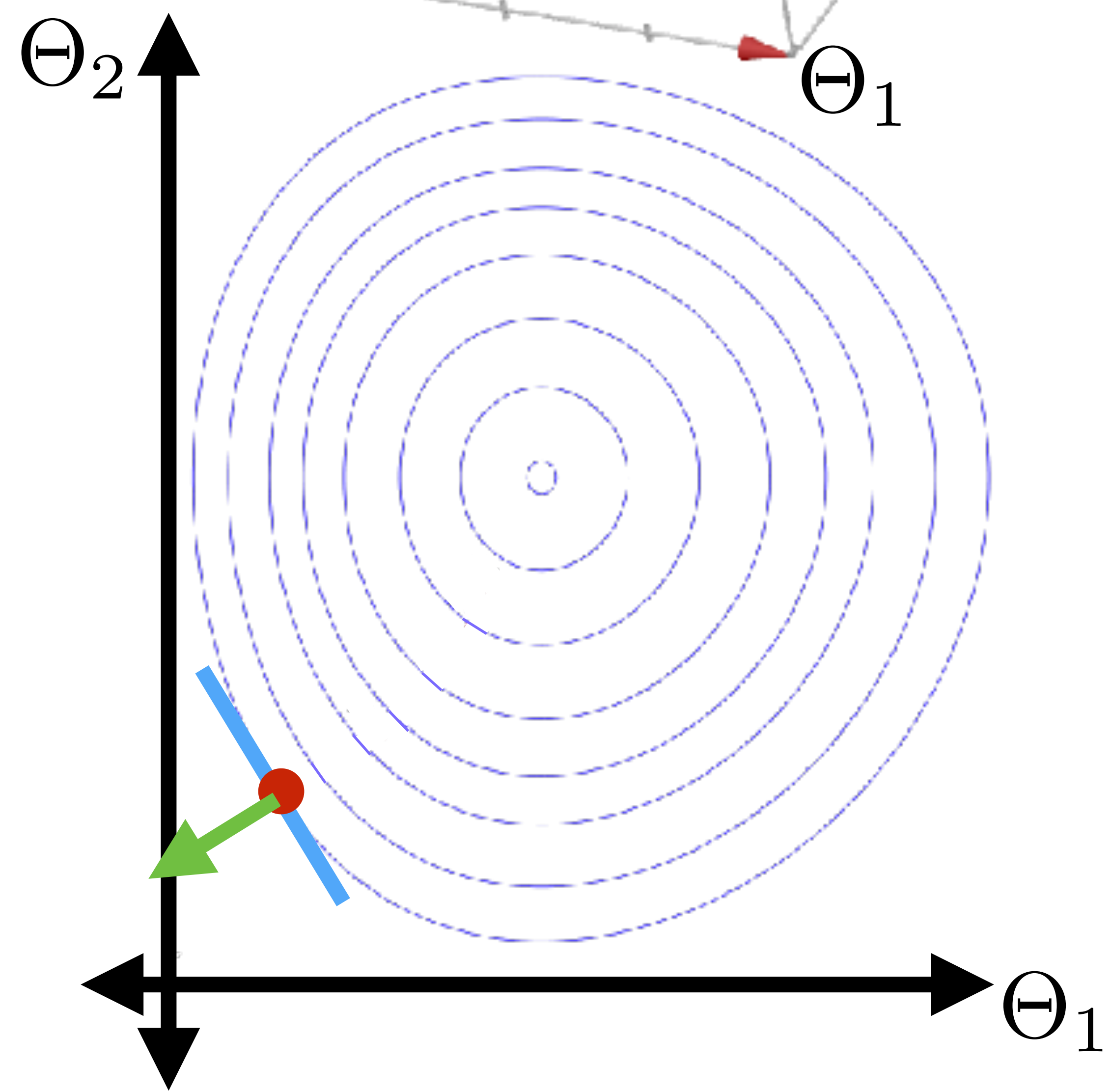
e.g. $L(g,a) =$
$$\begin{cases} (g-a)^2 \text{ if } g > a \\ 5(g-a)^2 \text{ if } g \leq a \end{cases}$$

  - Can be too slow to run, even in ridge regression

Gradient descent

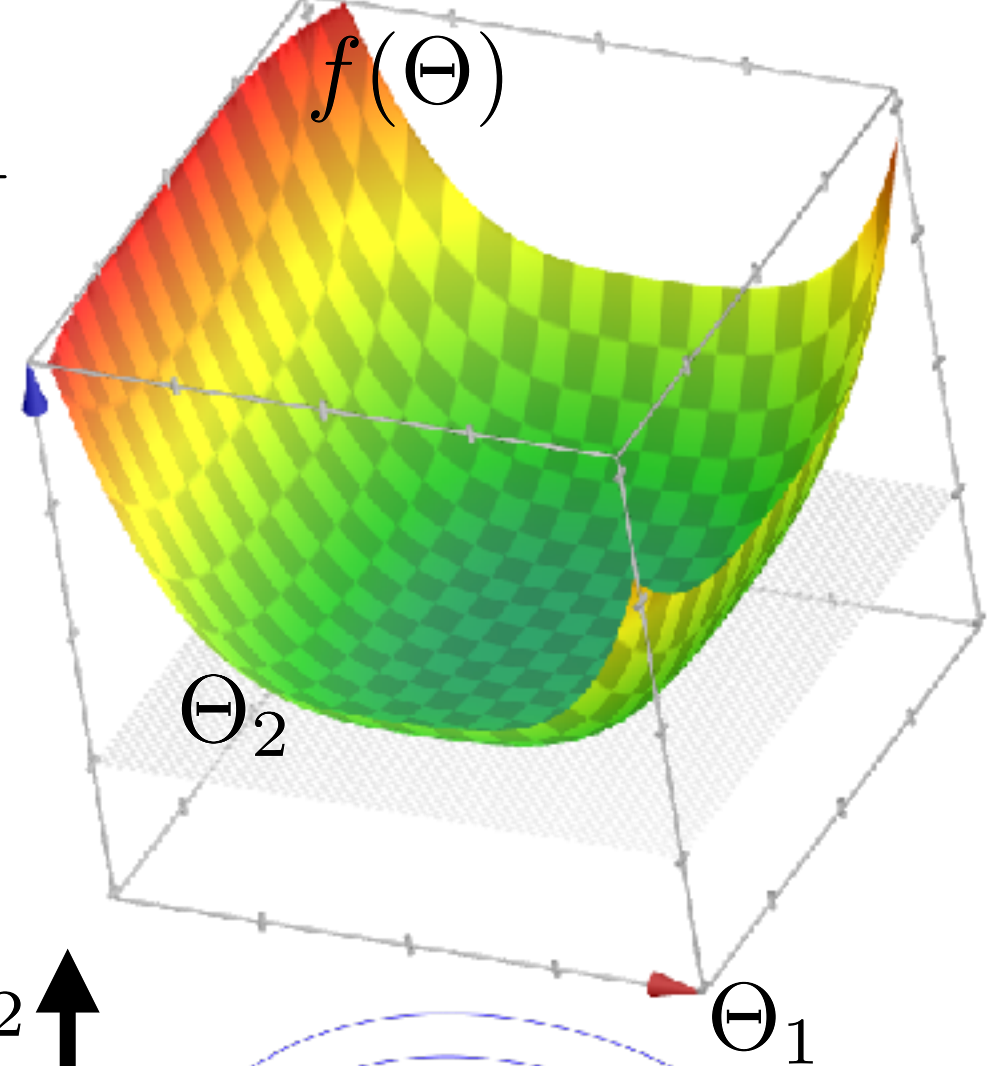
- Gradient $\nabla_\Theta f = \left[\dfrac{\partial f}{\partial \Theta_1}, \ldots, \dfrac{\partial f}{\partial \Theta_m}\right]^\top$
    - with $\Theta \in \mathbb{R}^m$

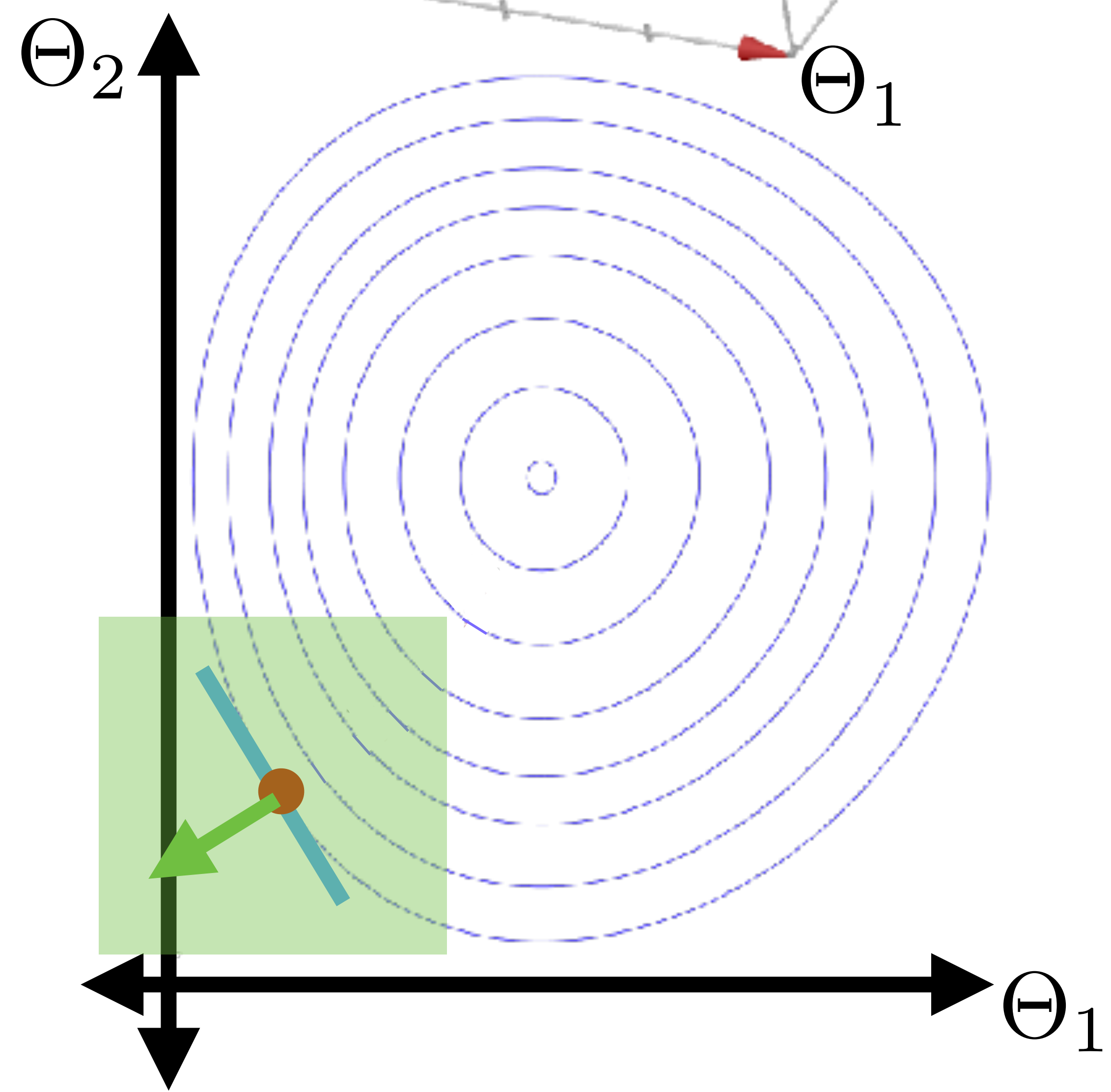




Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$



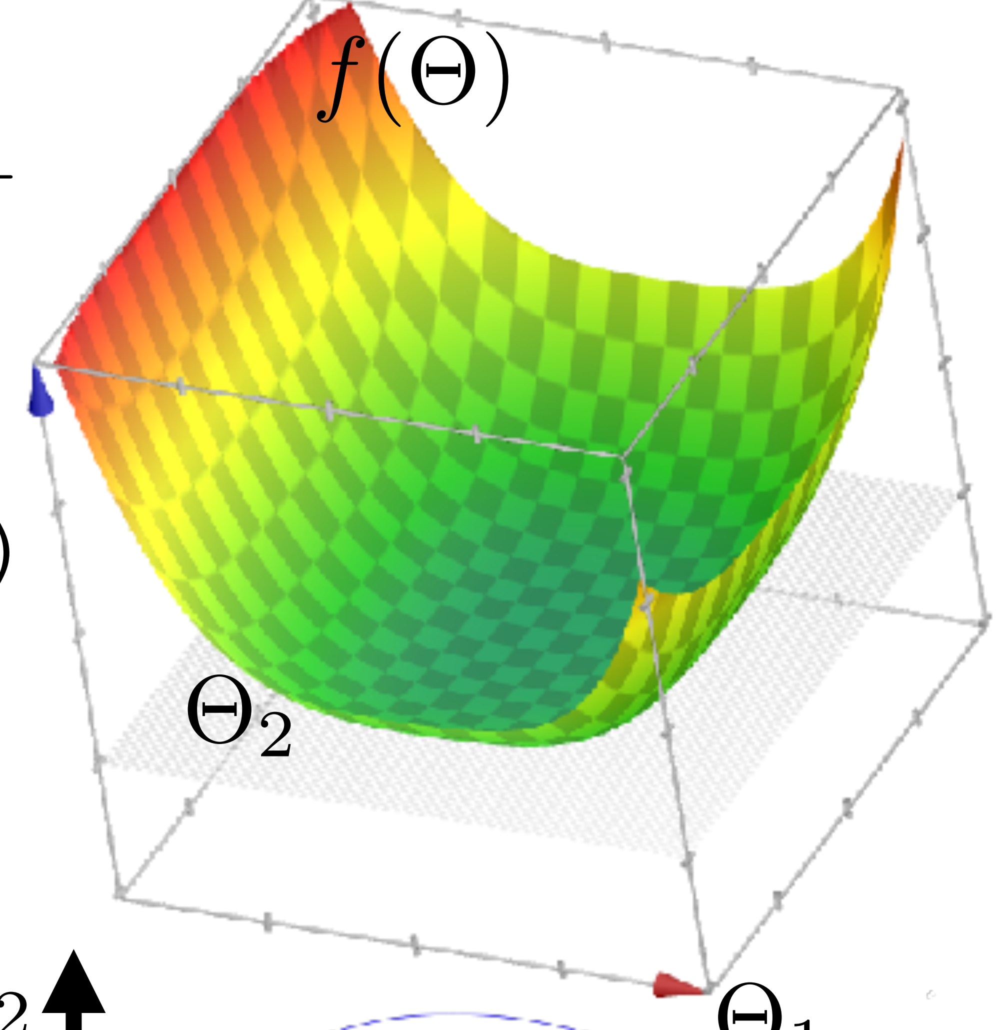


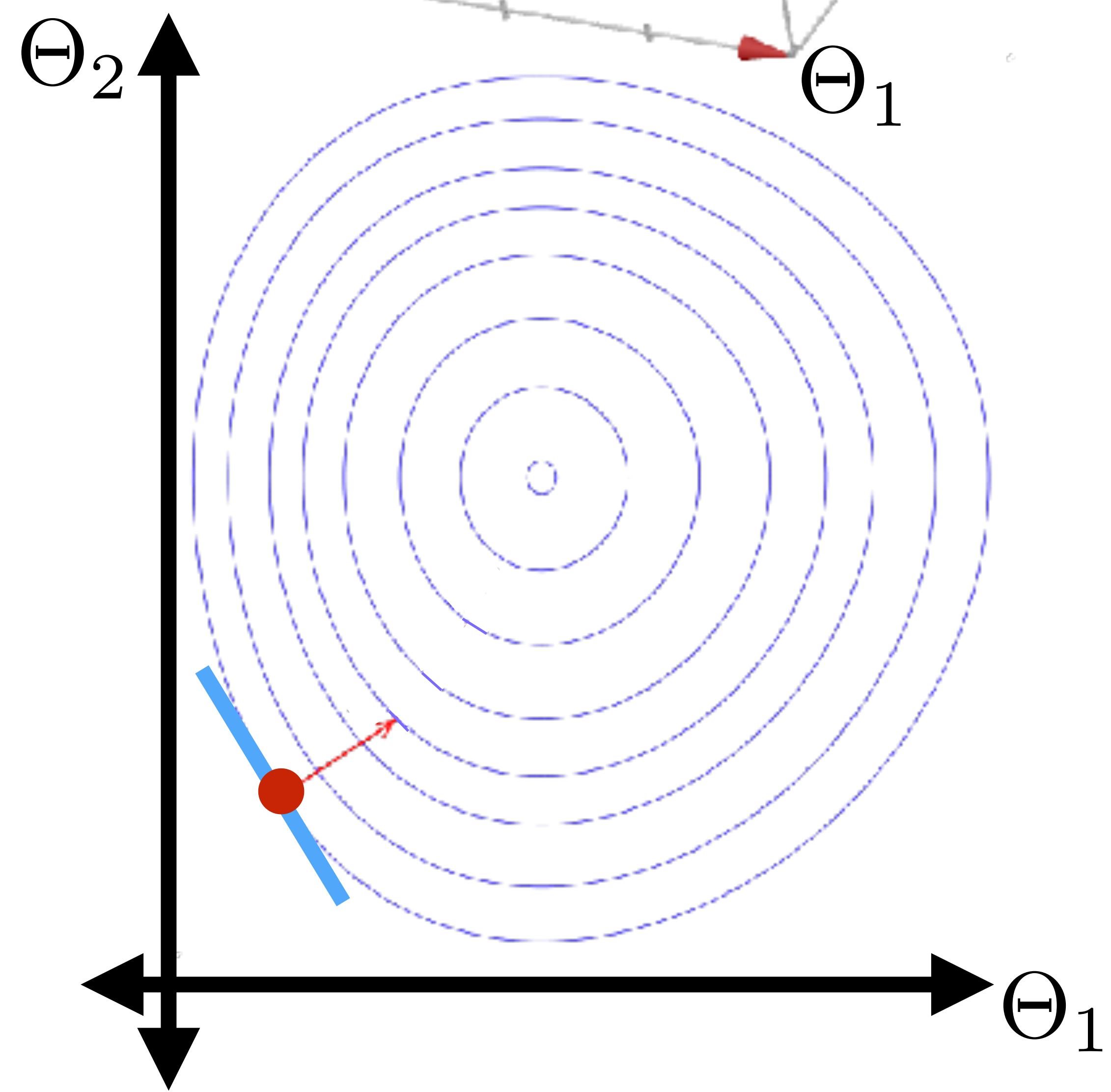
Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

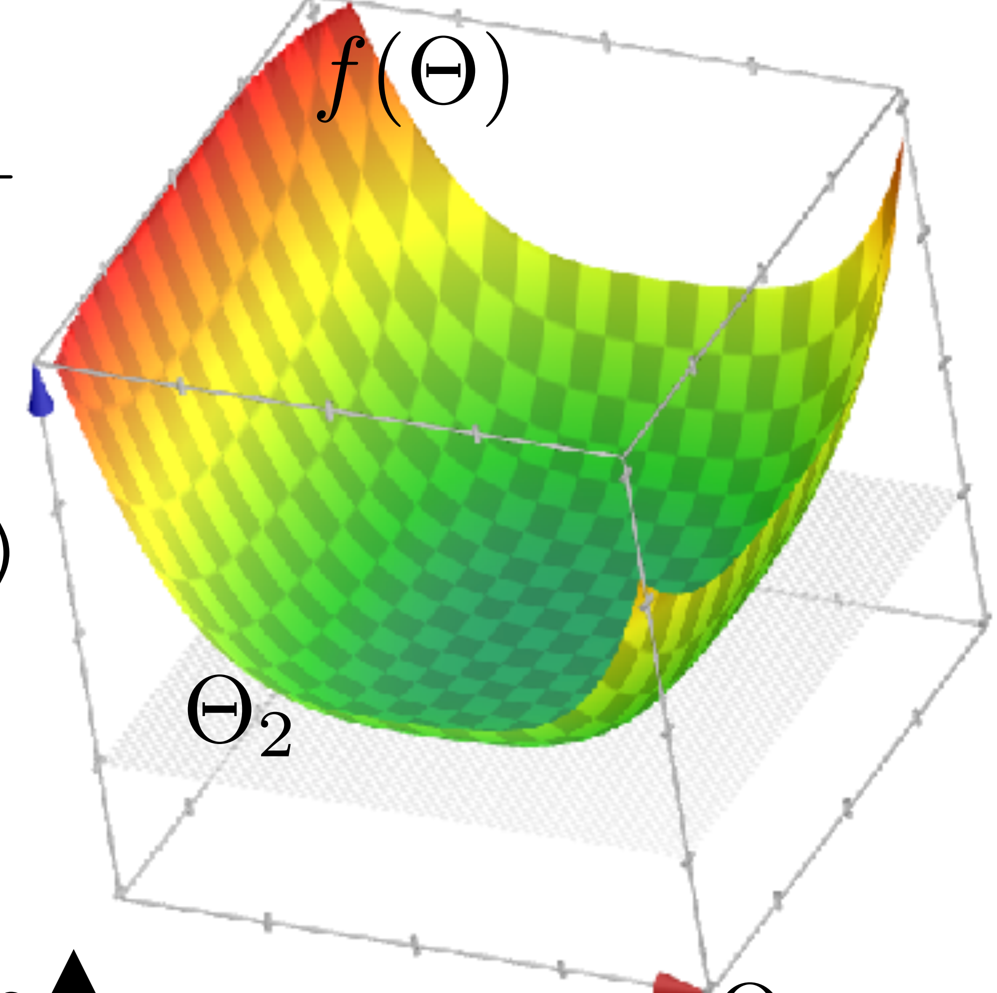
  `Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$





Gradient descent

- Gradient $\nabla_{\Theta} f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^{\top}$
    - with $\Theta \in \mathbb{R}^m$

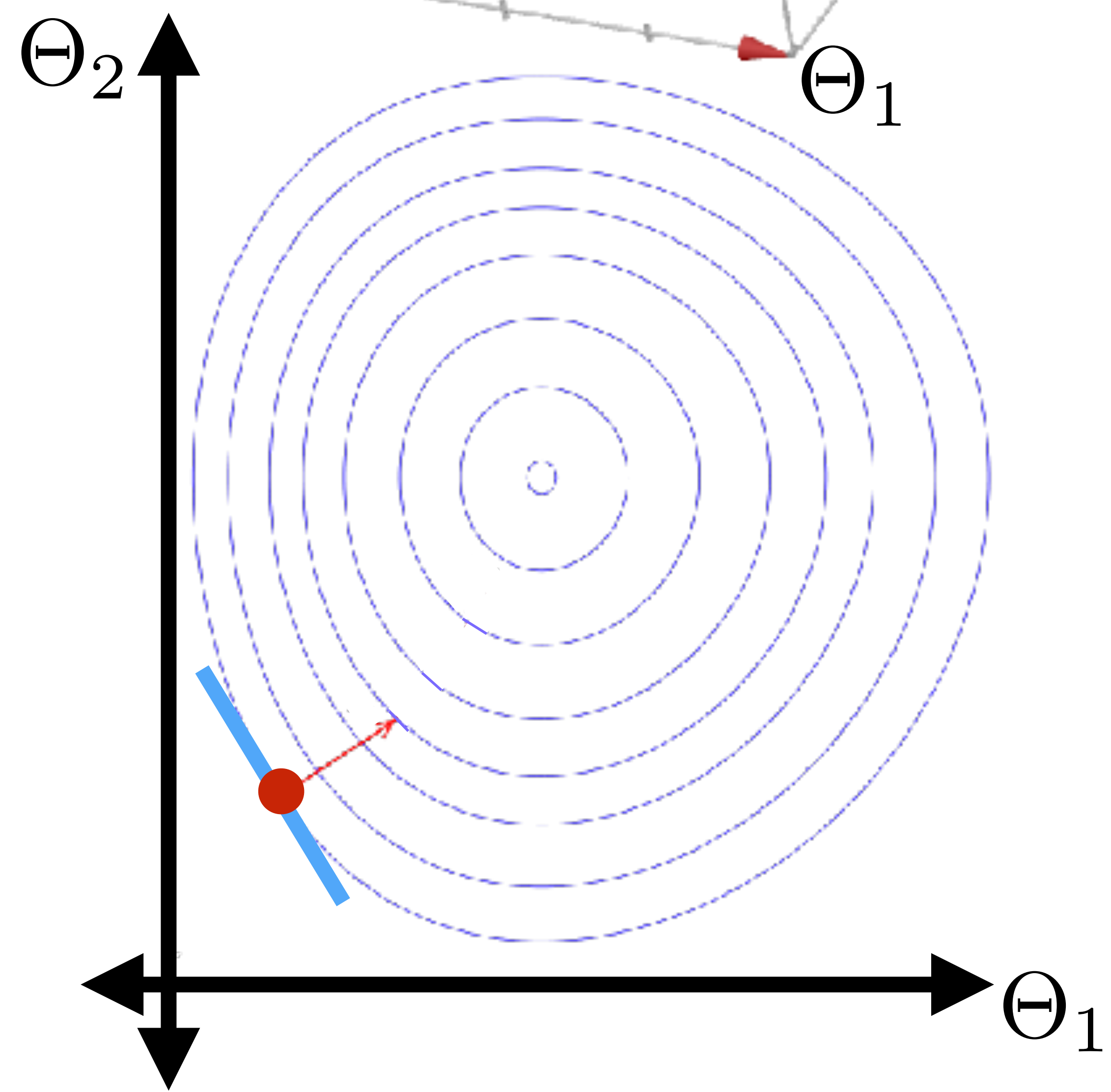



```
Gradient-Descent (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
```

$\texttt{Gradient-Descent}\,(\Theta_{\text{init}}, \eta, f, \nabla_{\Theta} f, \epsilon)$

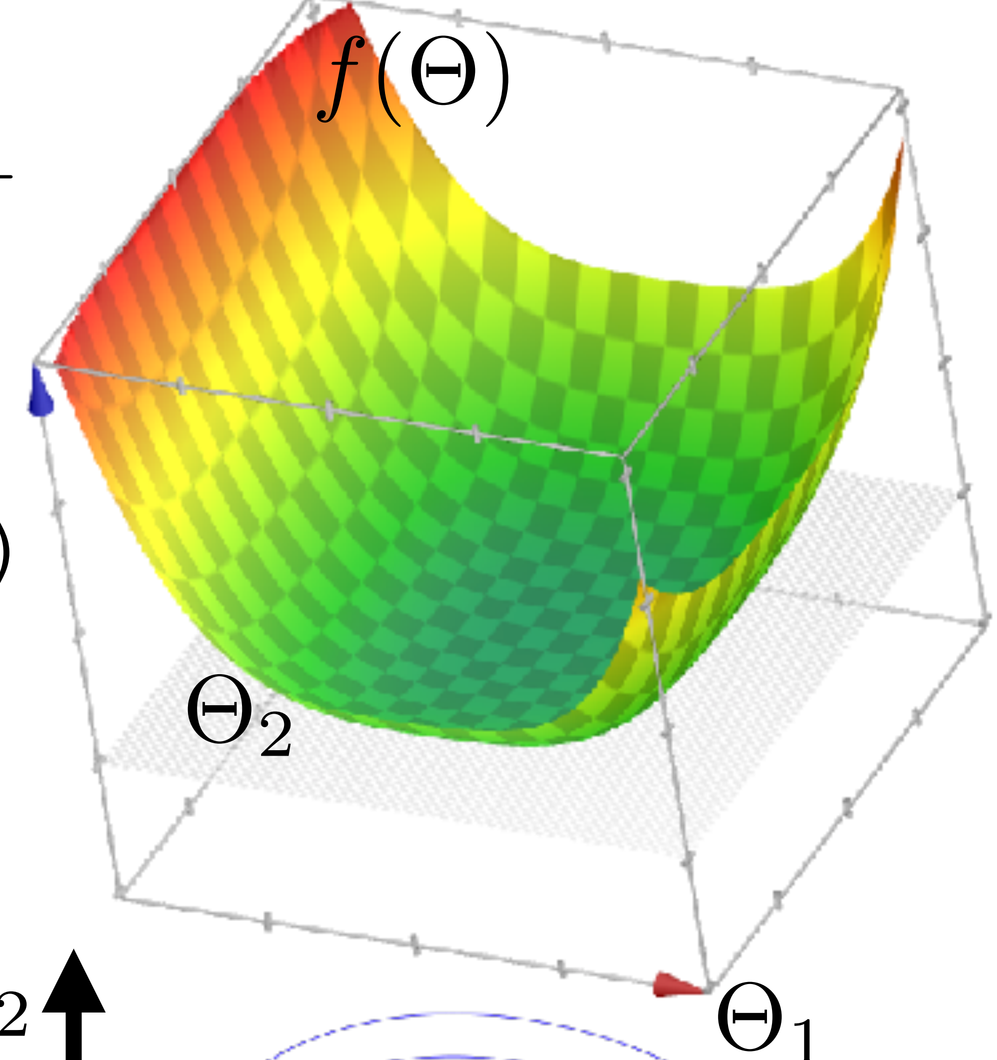
$\quad\texttt{Initialize}\ \Theta^{(0)} = \Theta_{\text{init}}$

$\quad\texttt{Initialize t = 0}$



Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$



`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

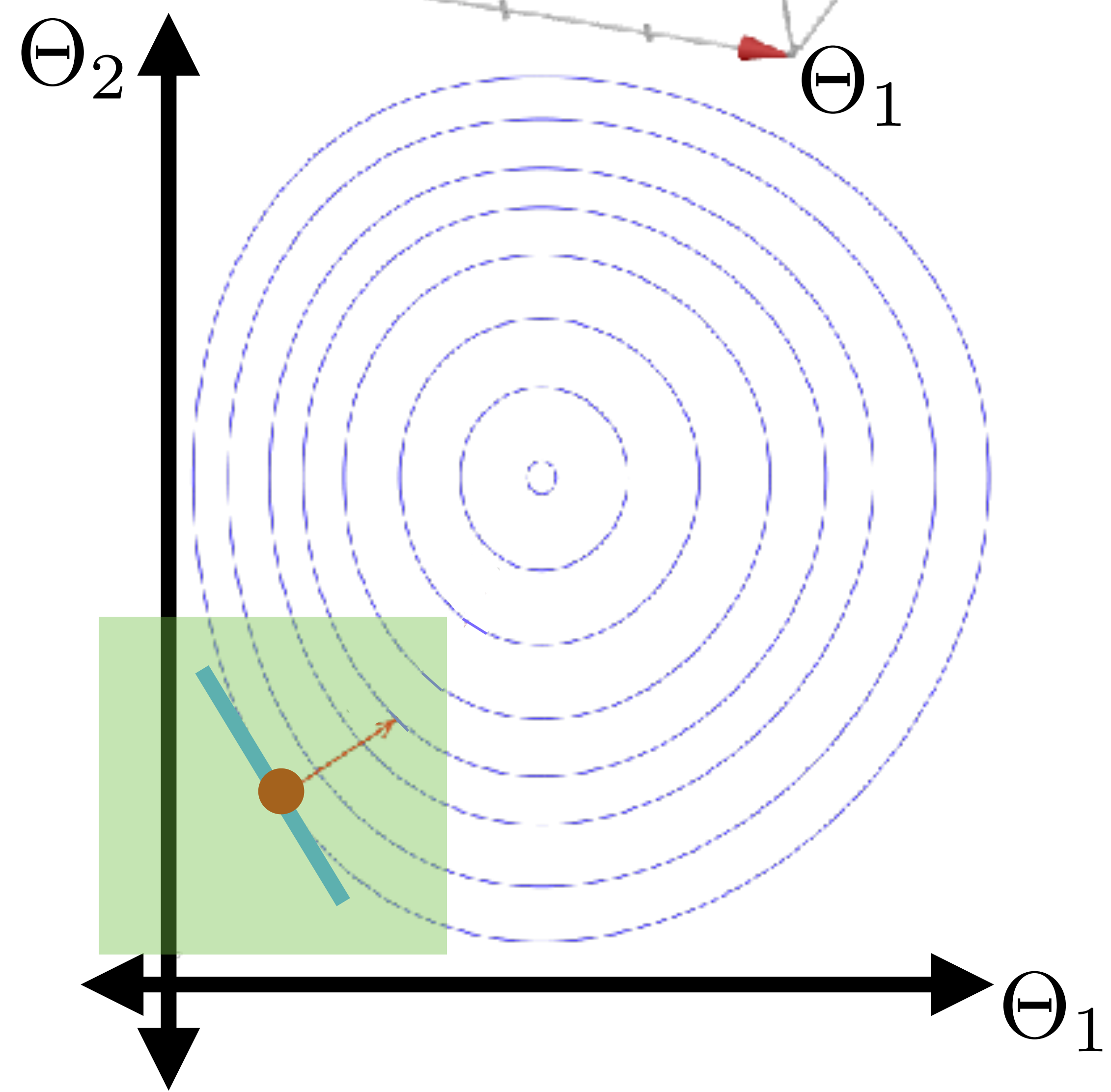
  `Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

  `Initialize t = 0`

  `repeat`

    `t = t + 1`

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

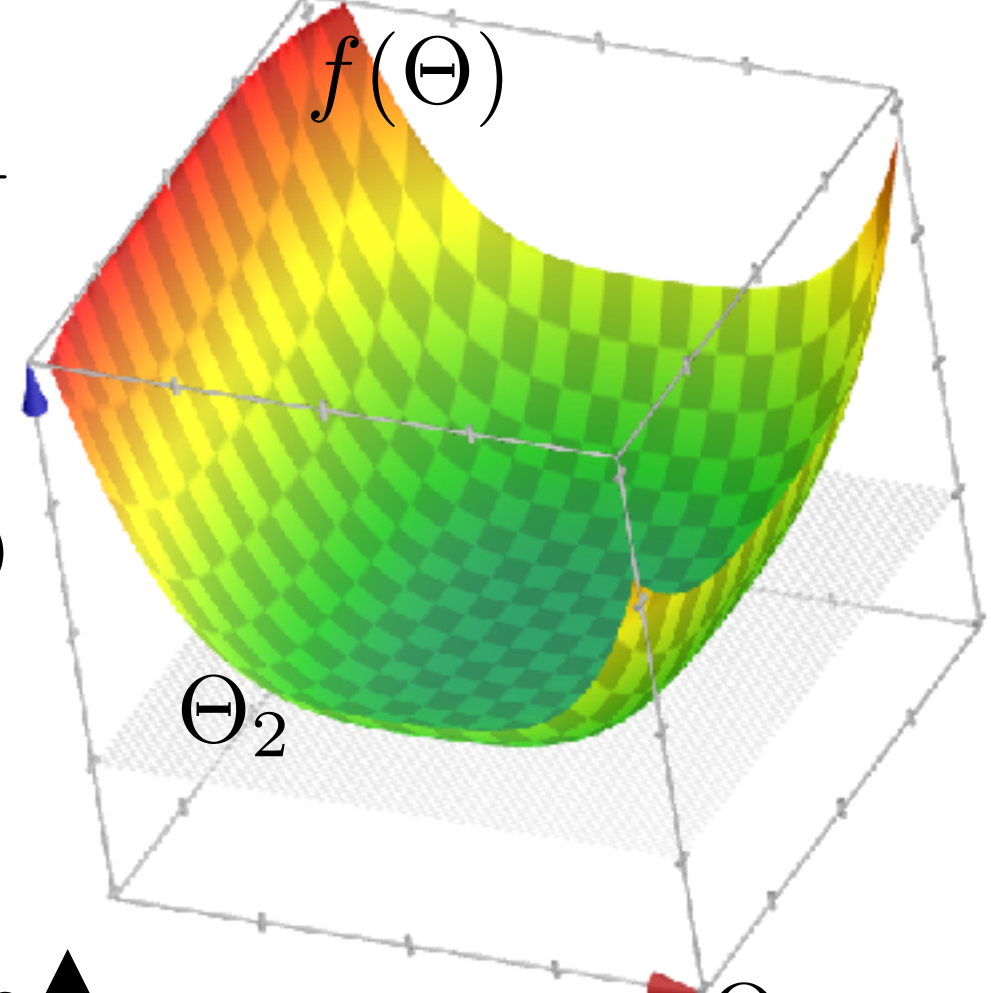


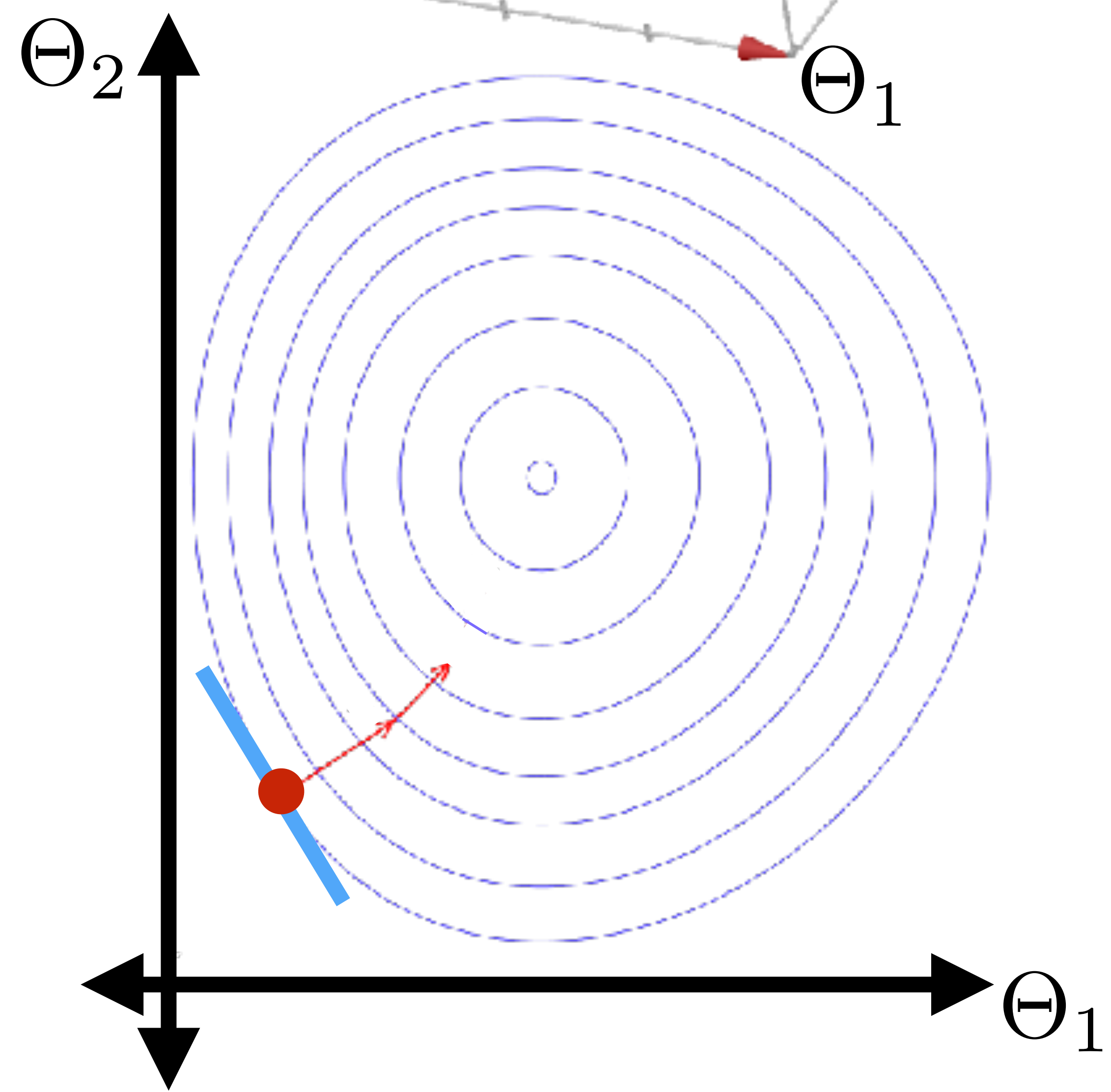
Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

```
Gradient-Descent (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t-1) - η ∇_Θ f(Θ^(t-1))
```

$$\texttt{Gradient-Descent}\,(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$$
$$\texttt{Initialize}\ \Theta^{(0)} = \Theta_{\text{init}}$$
$$\texttt{Initialize t = 0}$$
repeat
$$\texttt{t = t + 1}$$
$$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$$



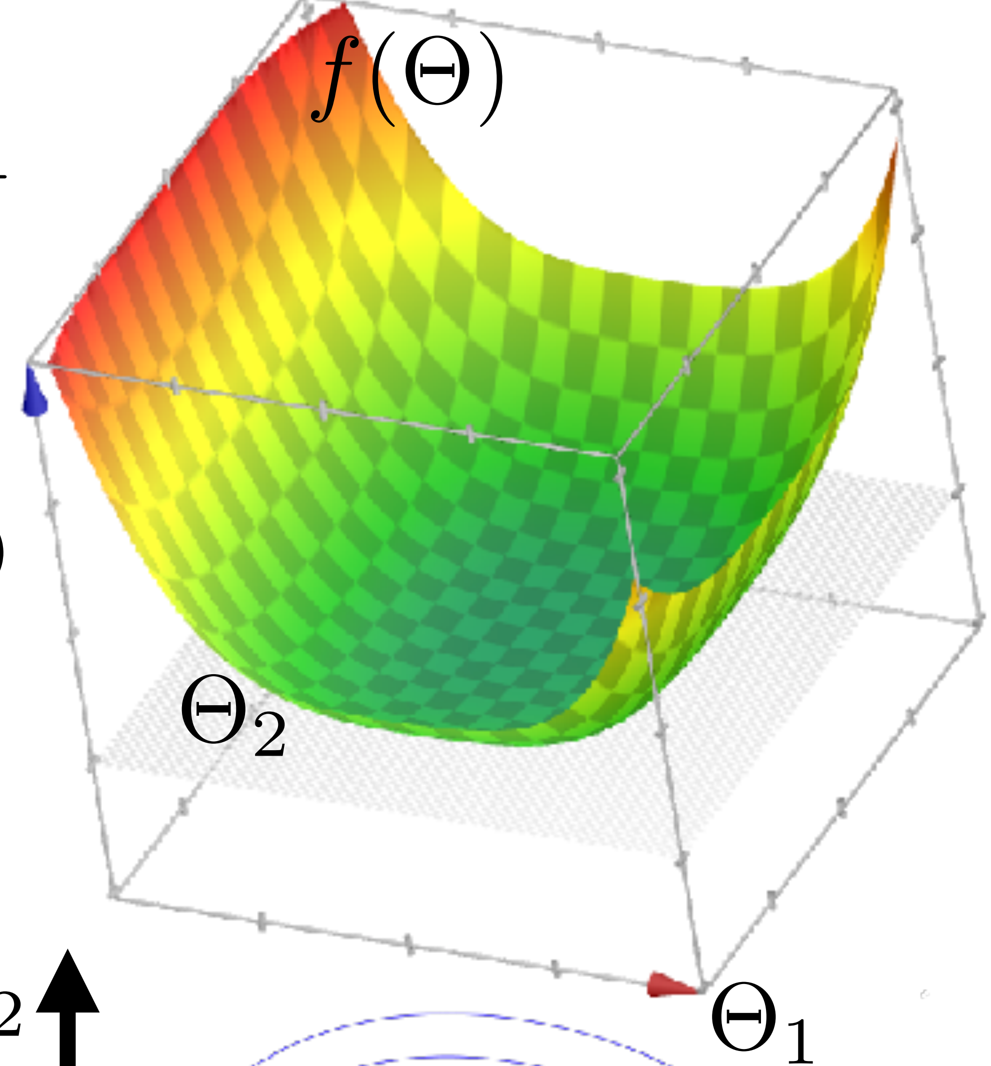


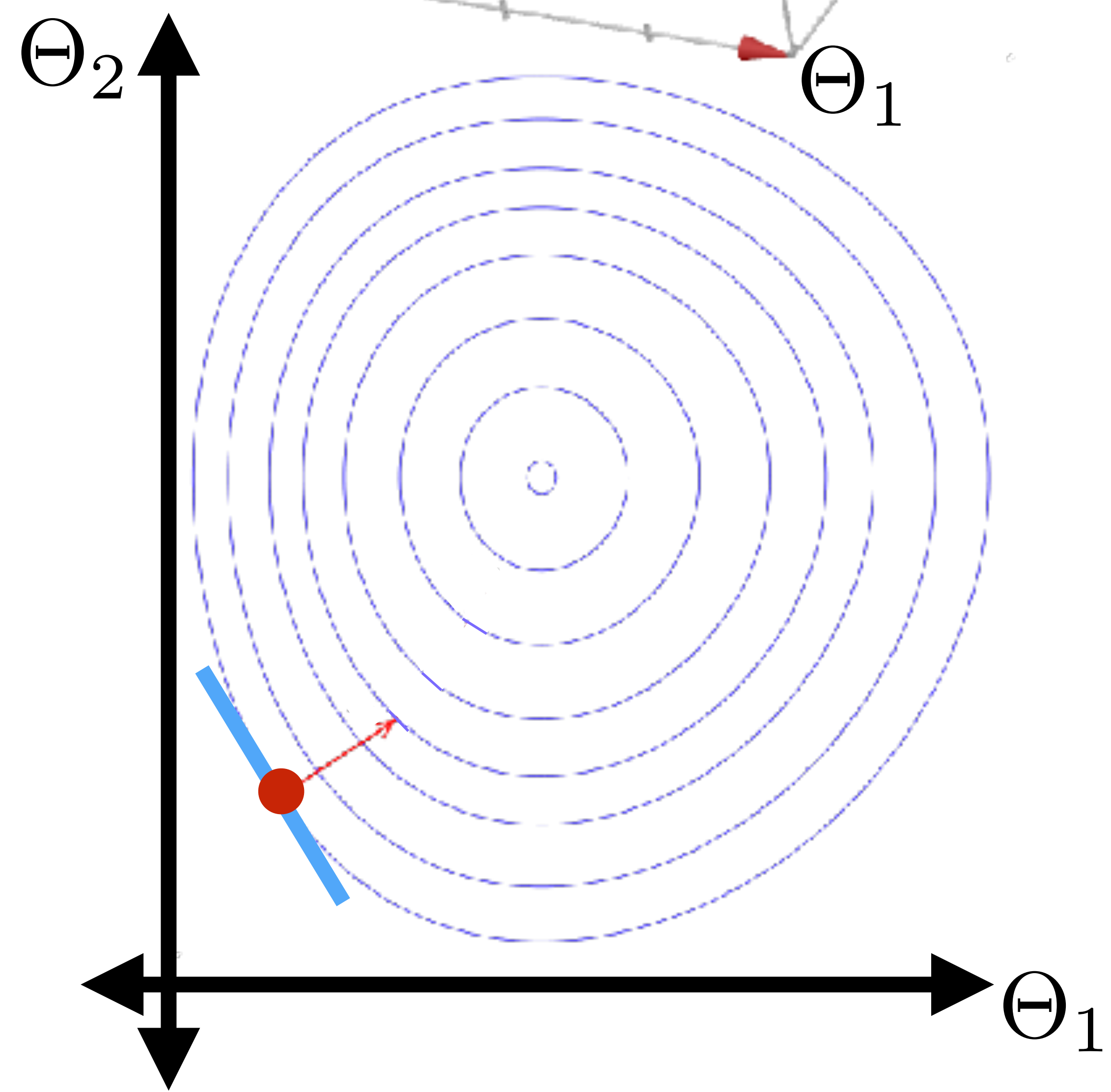
Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

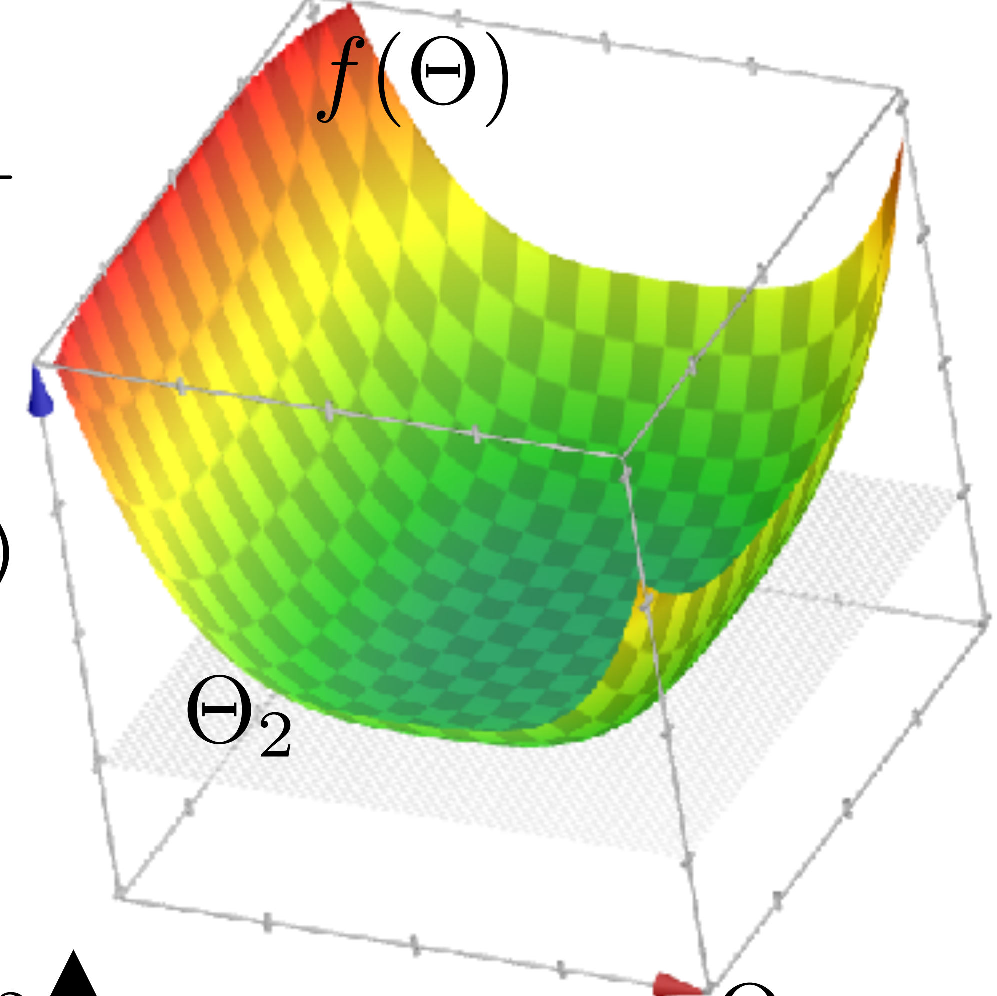
  `Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$



`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

  `Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

  `Initialize t = 0`

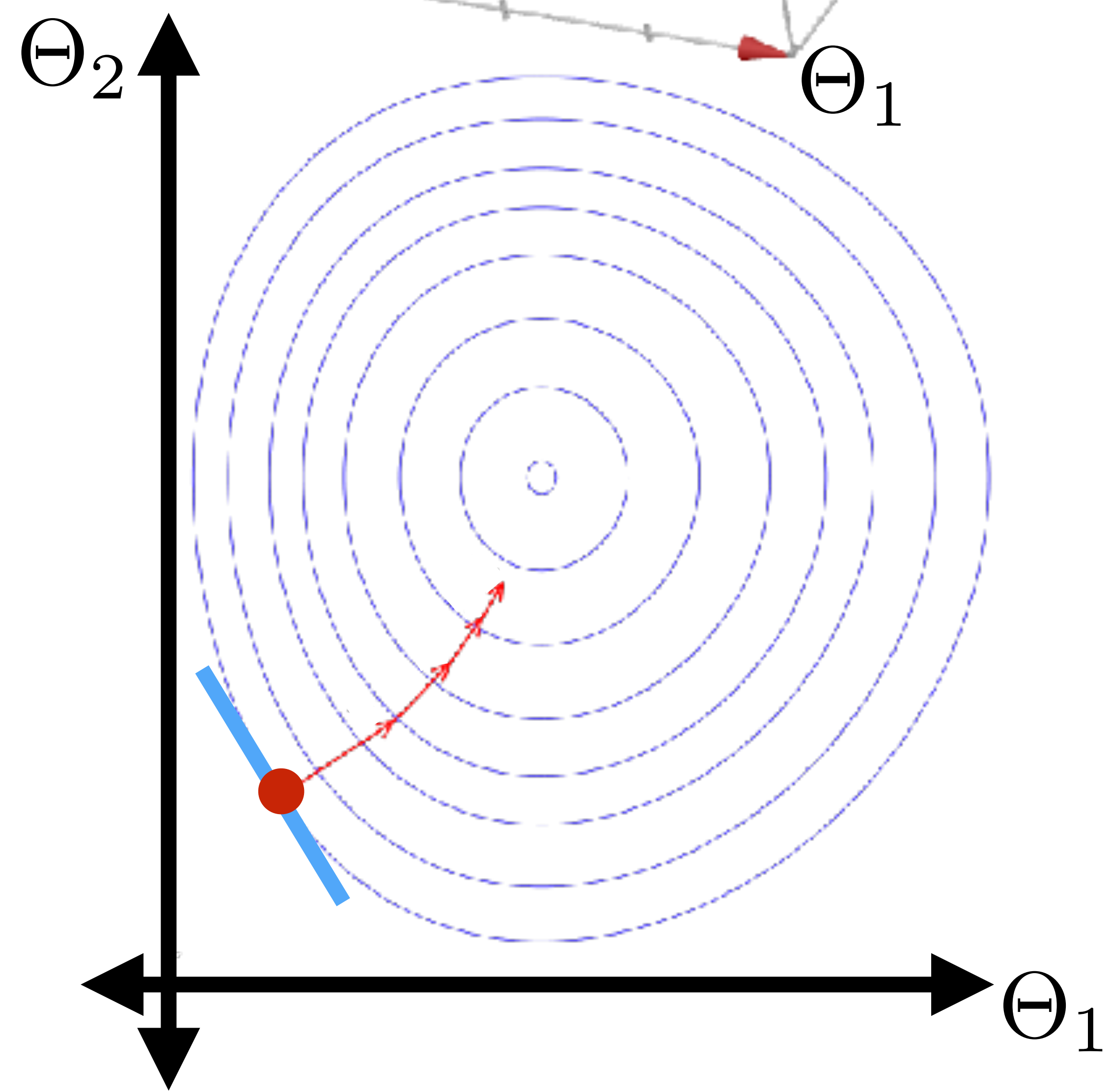
  `repeat`

    `t = t + 1`

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

  `until` $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

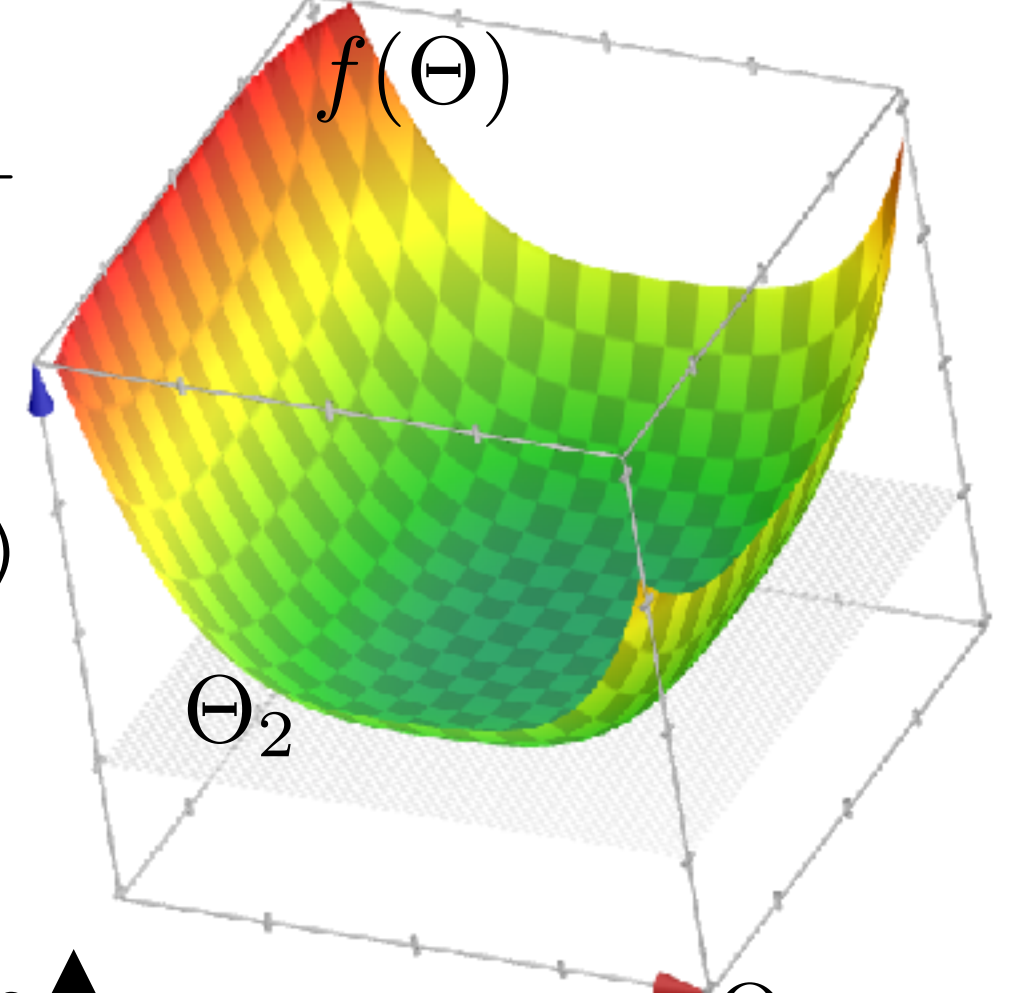
  `Return` $\Theta^{(t)}$



- Other possible stopping criteria:

Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$




```
Gradient-Descent (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t-1) − η ∇_Θ f(Θ^(t-1))
  until |f(Θ^(t)) − f(Θ^(t-1))| < ε
  Return Θ^(t)
```

$\texttt{Gradient-Descent}\,(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

$\quad$ Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

$\quad$ Initialize $t = 0$

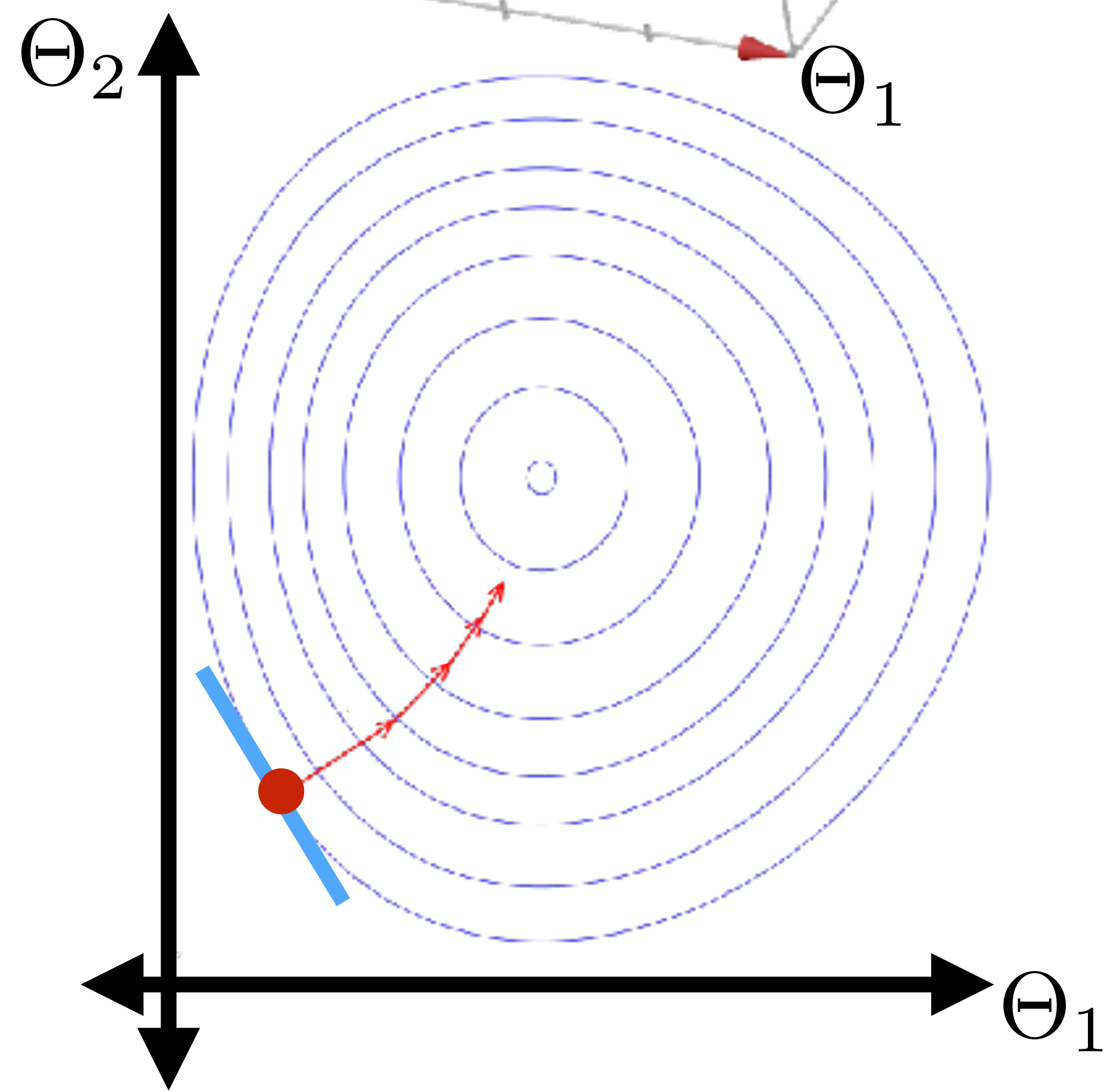
$\quad$ **repeat**

$\quad\quad t = t + 1$

$\quad\quad \Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

$\quad$ **until** $\left|f(\Theta^{(t)}) - f(\Theta^{(t-1)})\right| < \epsilon$

$\quad$ **Return** $\Theta^{(t)}$



- Other possible stopping criteria:
  - Max number of iterations T
  - $\|\Theta^{(t)} - \Theta^{(t-1)}\| < \epsilon$
  - $\|\nabla_\Theta f(\Theta^{(t)})\| < \epsilon$

Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

  `Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

  `Initialize t = 0`

  `repeat`

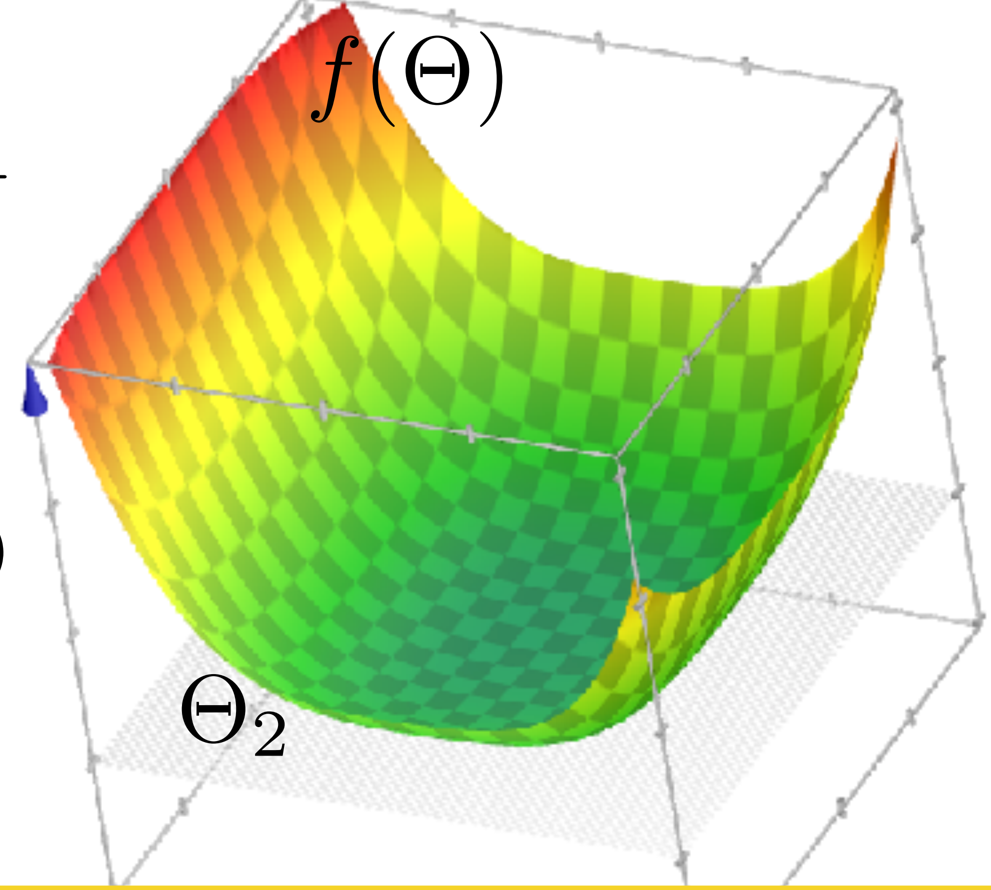
    `t = t + 1`

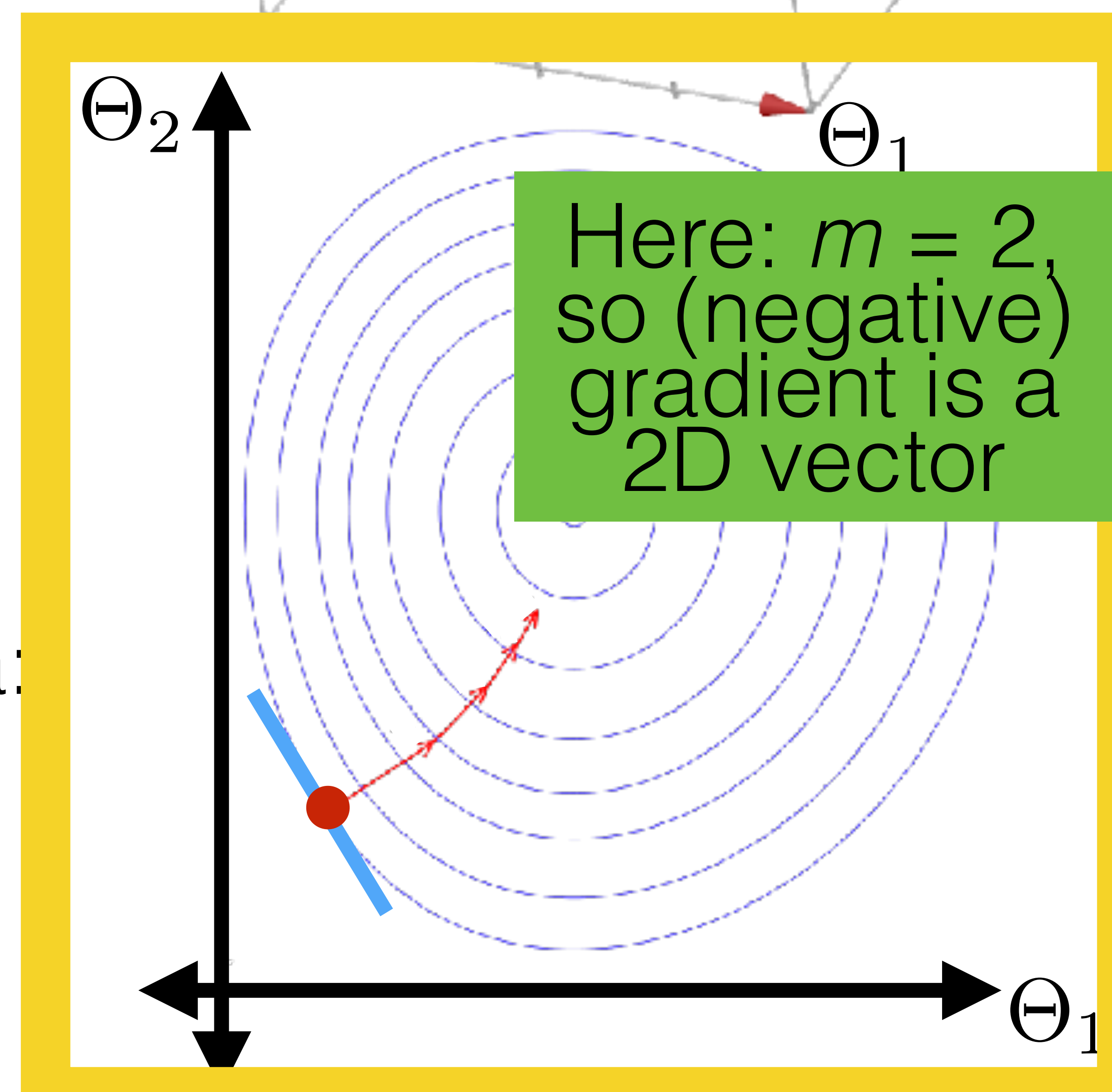
    $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

  `until` $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

  `Return` $\Theta^{(t)}$

- Other possible stopping criteria:
  - Max number of iterations T
  - $\|\Theta^{(t)} - \Theta^{(t-1)}\| < \epsilon$
  - $\|\nabla_\Theta f(\Theta^{(t)})\| < \epsilon$





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon$

  `Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

  `Initialize t = 0`

  `repeat`

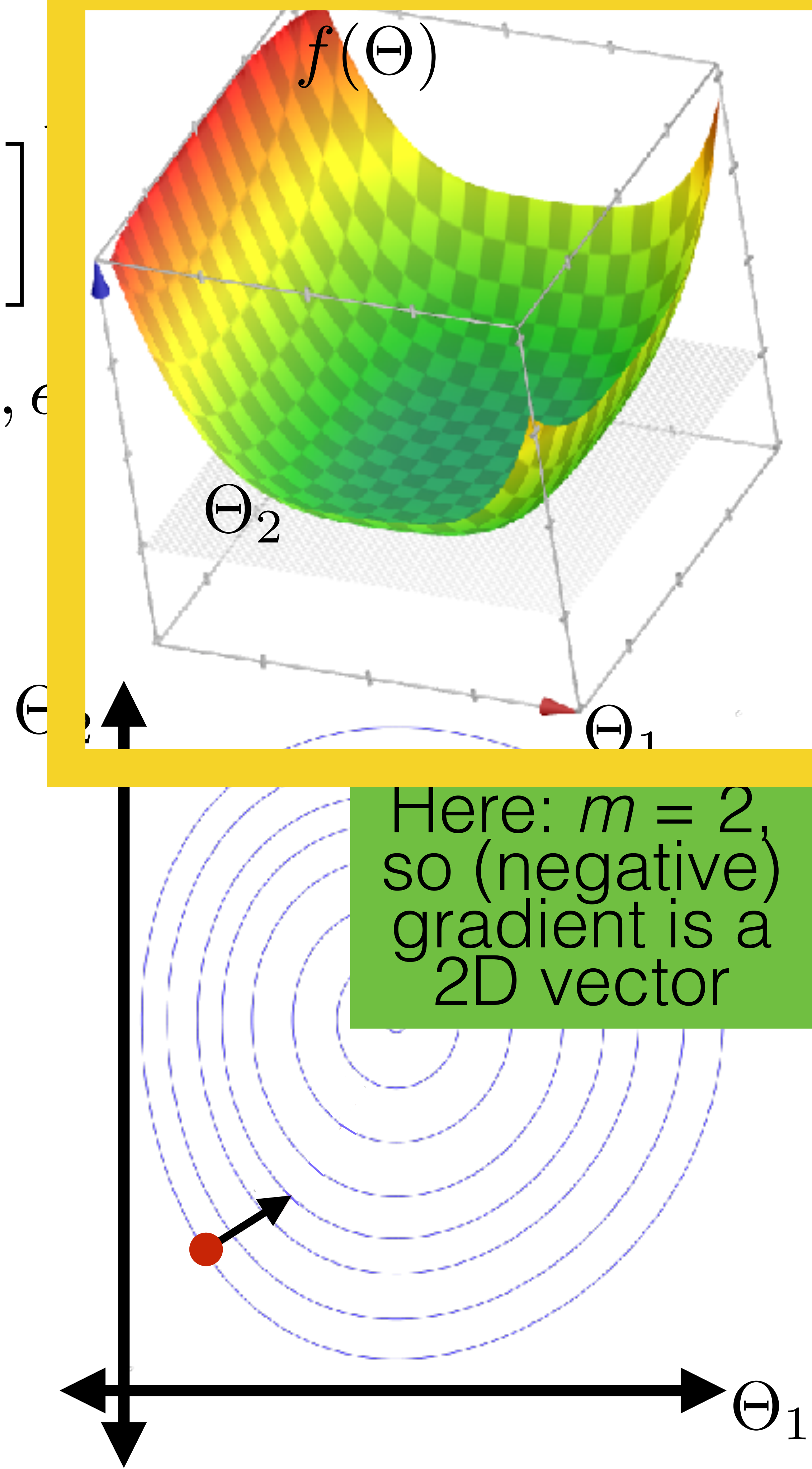
    `t = t + 1`

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

  `until` $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

  `Return` $\Theta^{(t)}$

- Other possible stopping criteria:
  - Max number of iterations T
  - $\|\Theta^{(t)} - \Theta^{(t-1)}\| < \epsilon$
  - $\|\nabla_\Theta f(\Theta^{(t)})\| < \epsilon$

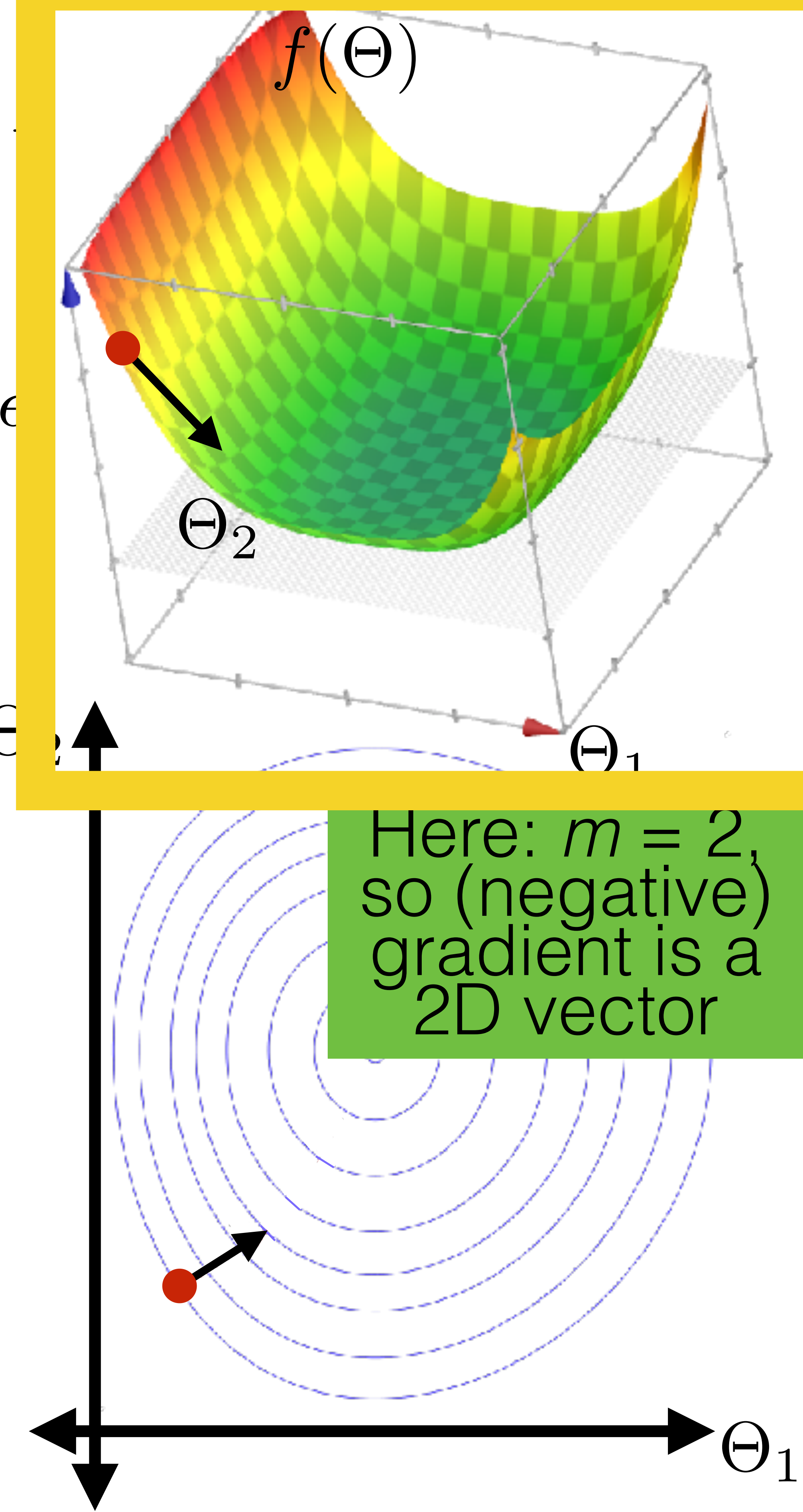


Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^T$
  - with $\Theta \in \mathbb{R}^m$

```
Gradient-Descent (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t−1) − η ∇_Θ f(Θ^(t−1))
  until |f(Θ^(t)) − f(Θ^(t−1))| < ε
  Return Θ^(t)
```

- Other possible stopping criteria:
  - Max number of iterations T
  - $\|\Theta^{(t)} - \Theta^{(t-1)}\| < \epsilon$
  - $\|\nabla_\Theta f(\Theta^{(t)})\| < \epsilon$



Here: $m = 2$, so (negative) gradient is a 2D vector

Gradient descent

- Gradient $\nabla_{\Theta} f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]$
  - with $\Theta \in \mathbb{R}^m$

Gradient-Descent $(\Theta_{\text{init}}, \eta, f, \nabla_{\Theta} f, \epsilon$

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Initialize t = 0

repeat

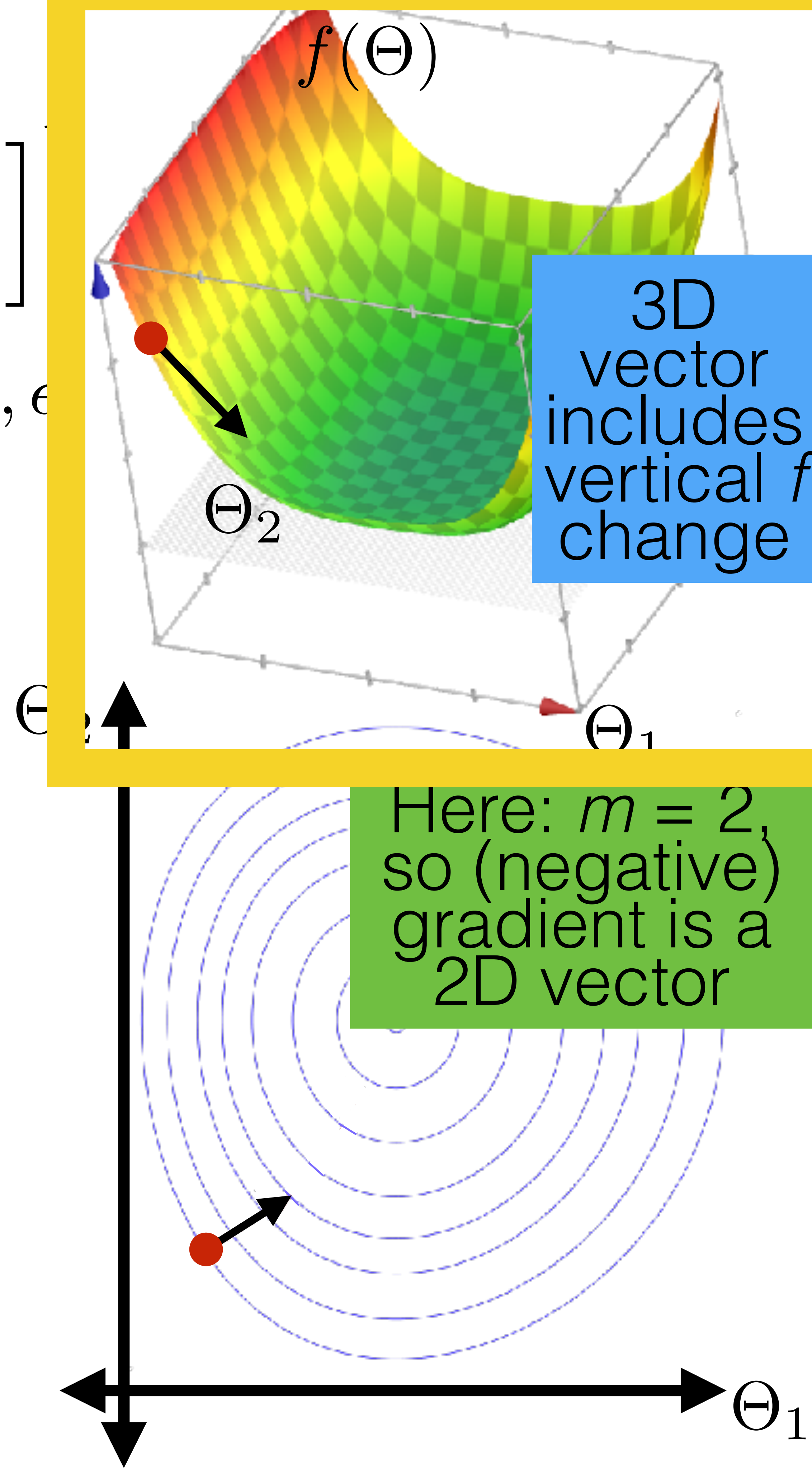
t = t + 1

$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_{\Theta} f(\Theta^{(t-1)})$

until $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

Return $\Theta^{(t)}$

- Other possible stopping criteria:
  - Max number of iterations T
  - $\|\Theta^{(t)} - \Theta^{(t-1)}\| < \epsilon$
  - $\|\nabla_{\Theta} f(\Theta^{(t)})\| < \epsilon$

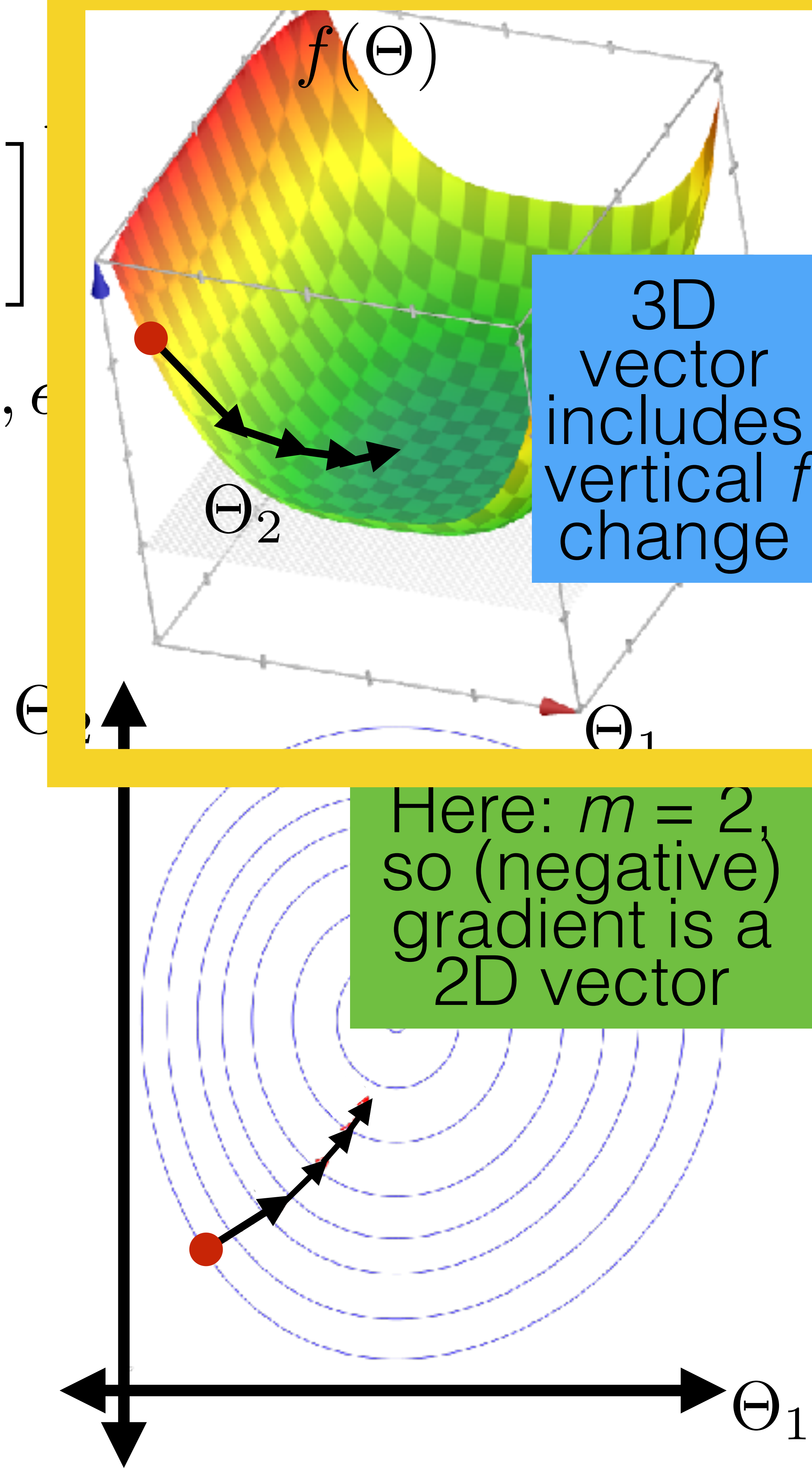


Gradient descent

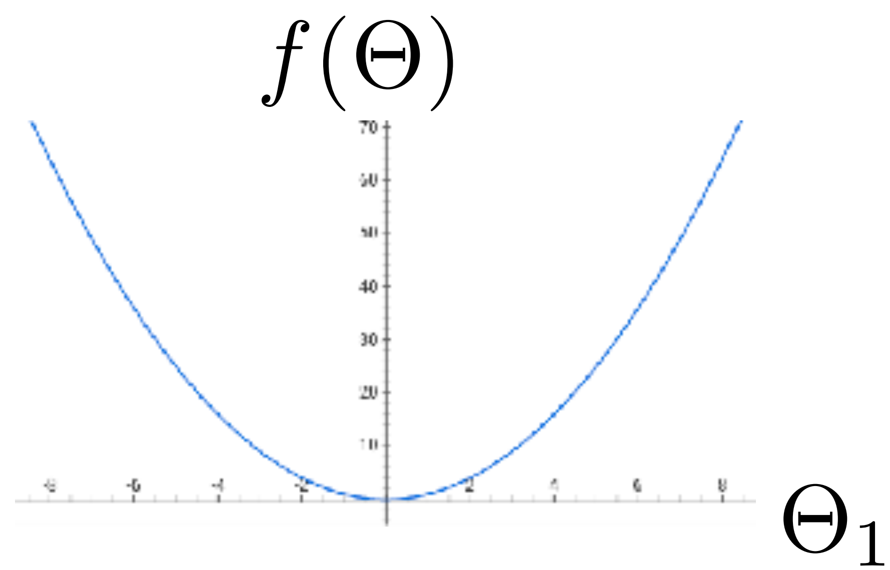
- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]$
  - with $\Theta \in \mathbb{R}^m$

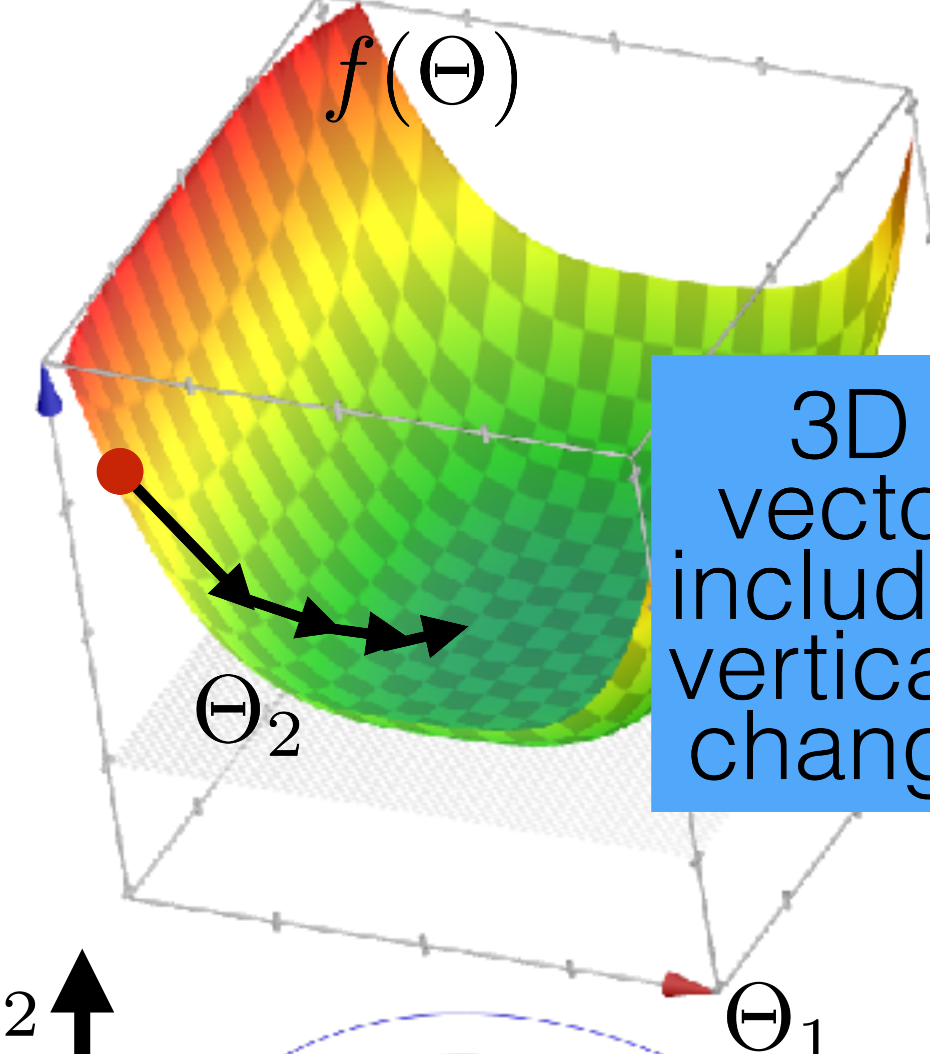
```
Gradient-Descent (Θ_init, η, f, ∇_Θ f, ε
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t-1) − η ∇_Θ f(Θ^(t-1))
  until |f(Θ^(t)) − f(Θ^(t-1))| < ε
  Return Θ^(t)
```

- Other possible stopping criteria:
  - Max number of iterations T
  - $\|\Theta^{(t)} - \Theta^{(t-1)}\| < \epsilon$
  - $\|\nabla_\Theta f(\Theta^{(t)})\| < \epsilon$

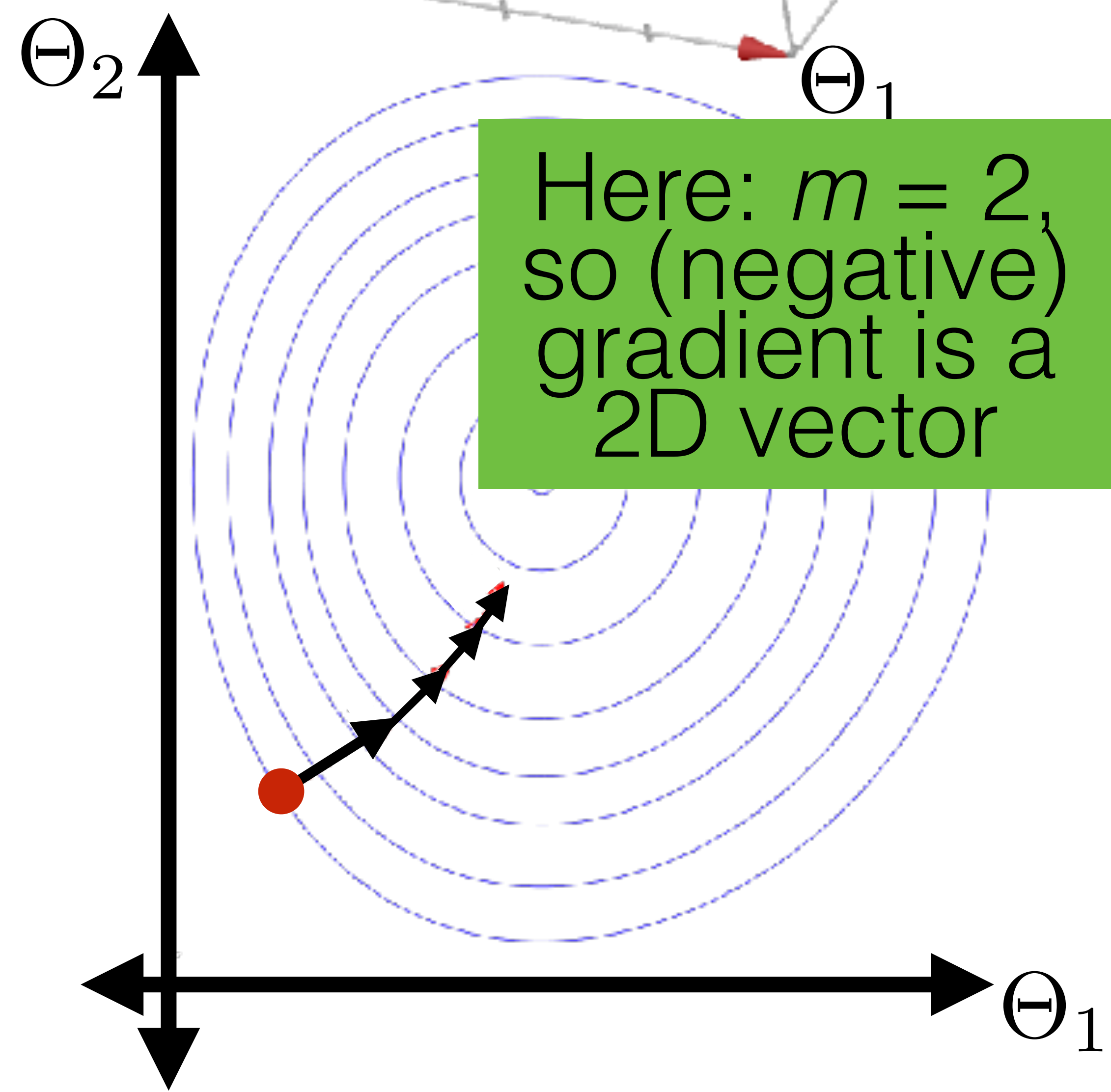


Gradient descent



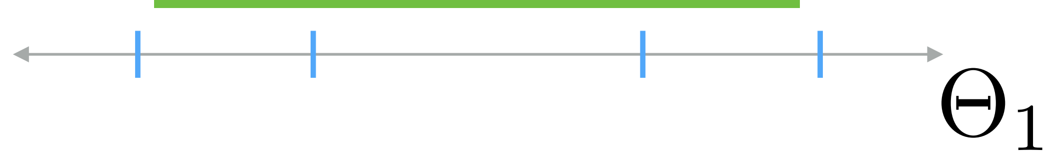


3D vector includes vertical f change

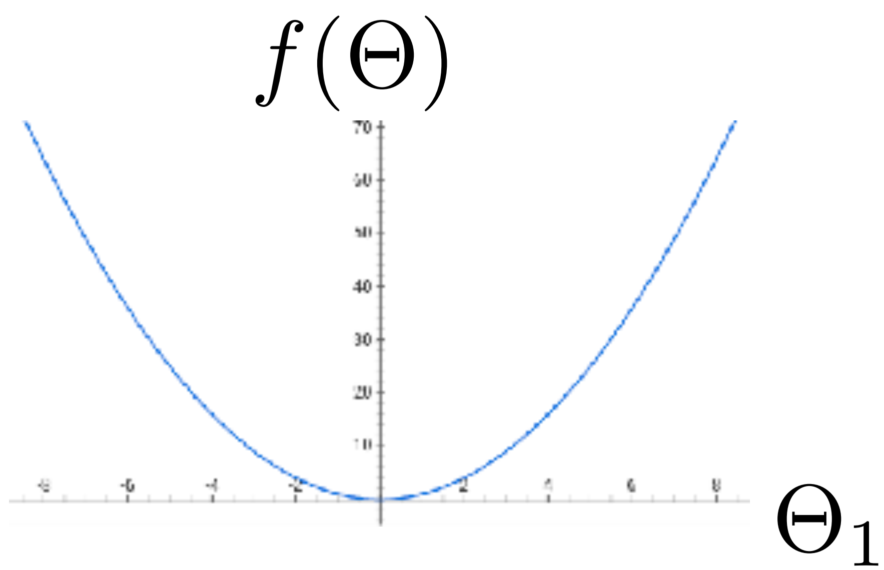


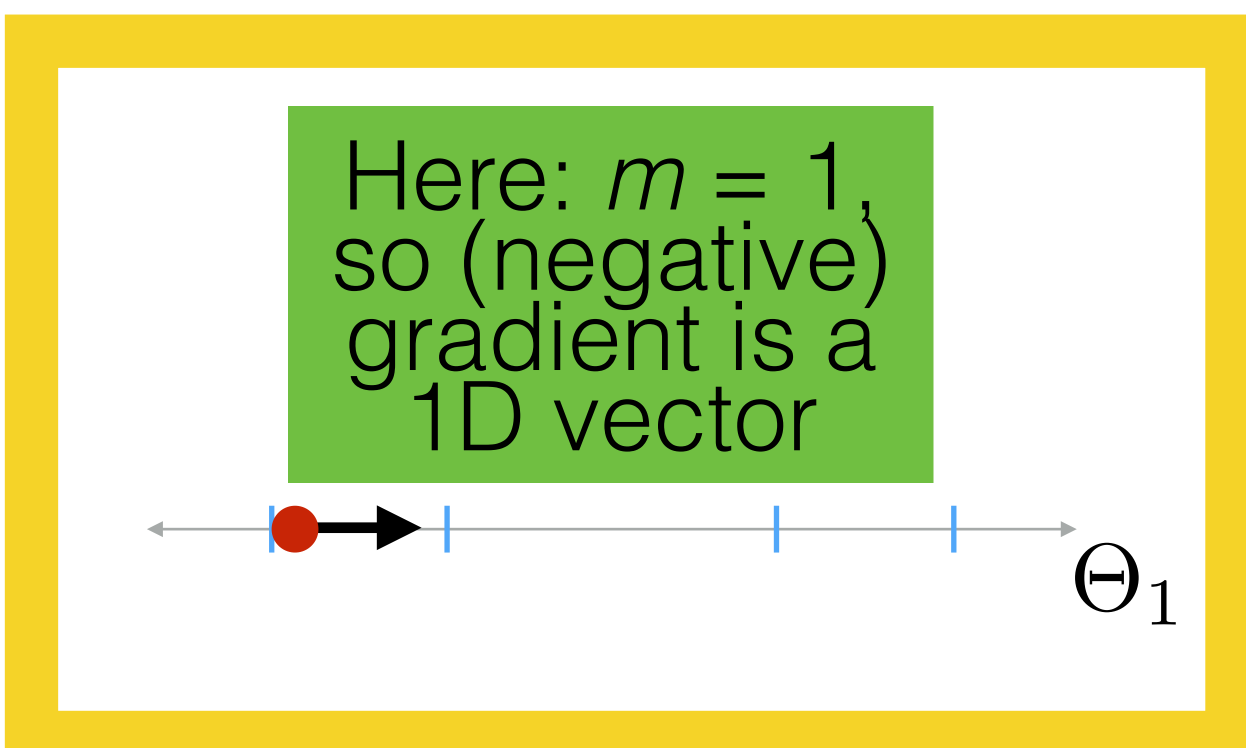
Here: m = 2, so (negative) gradient is a 2D vector

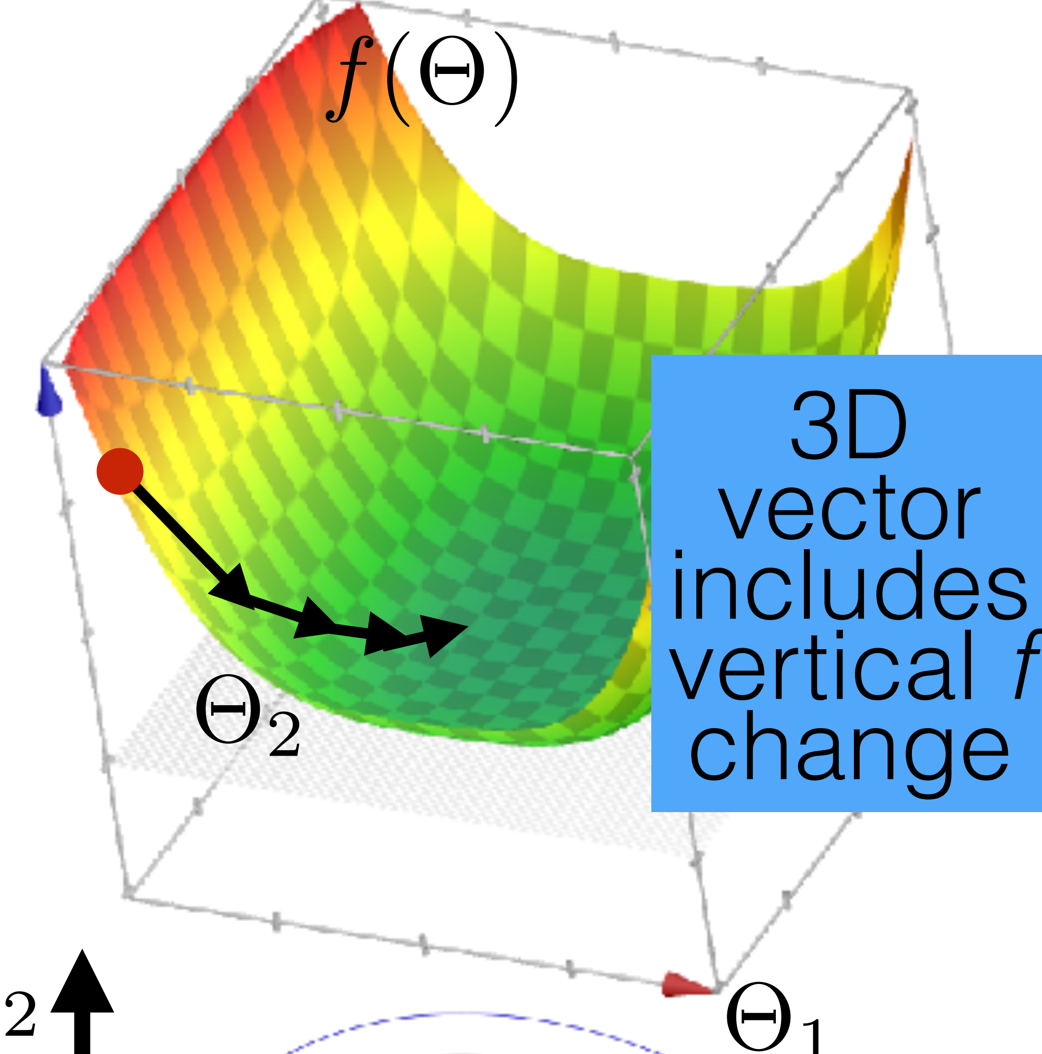
Here: m = 1, so (negative) gradient is a 1D vector

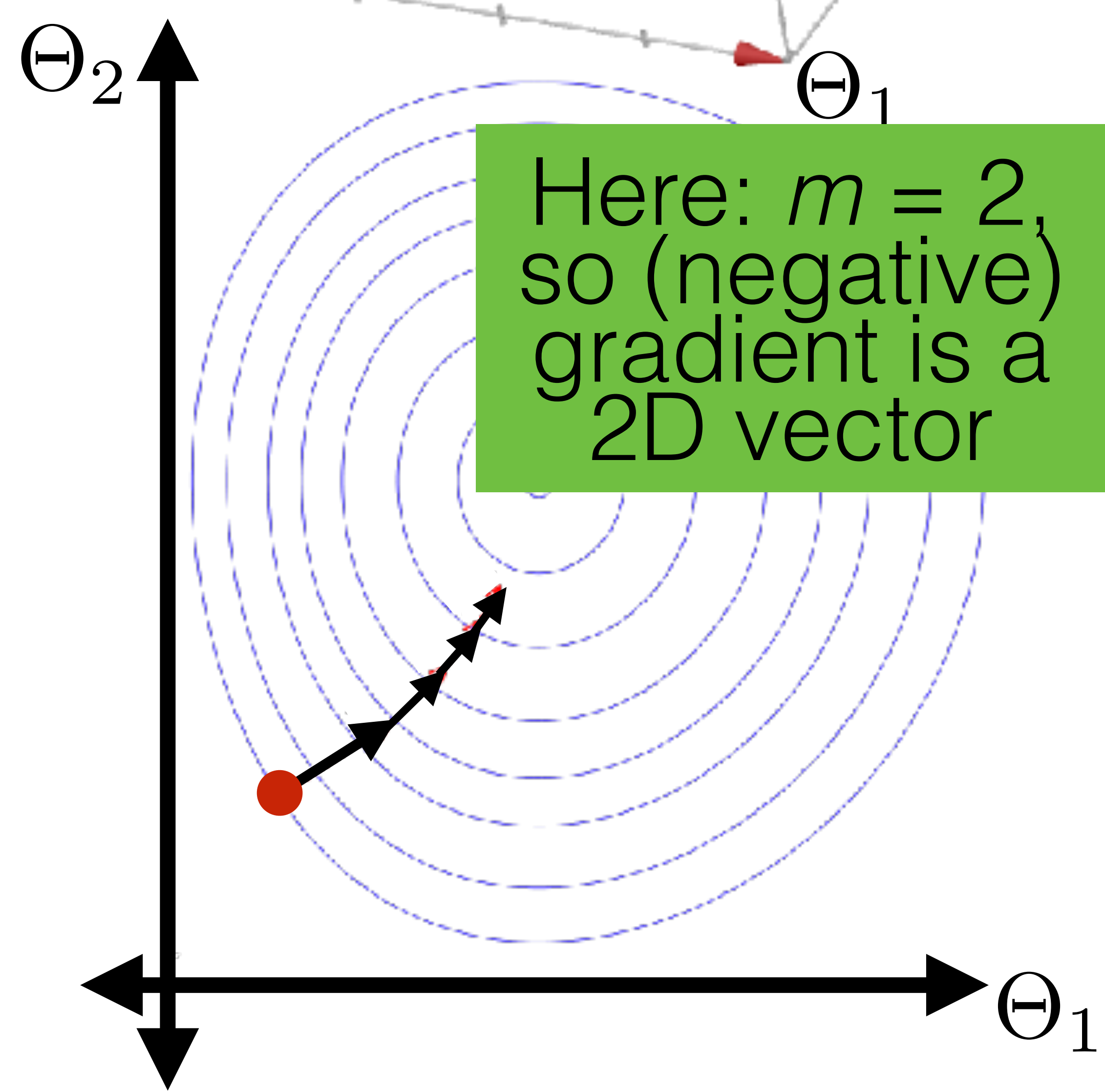


Gradient descent

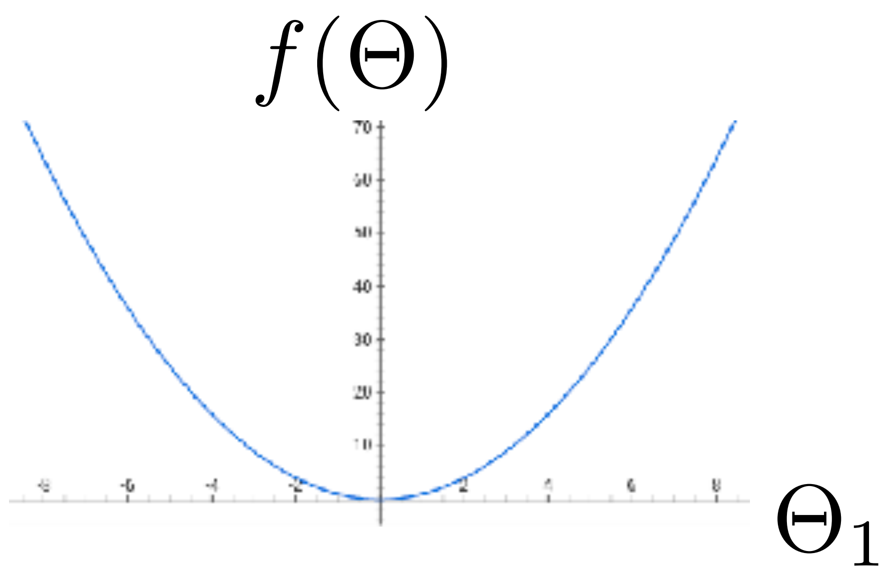


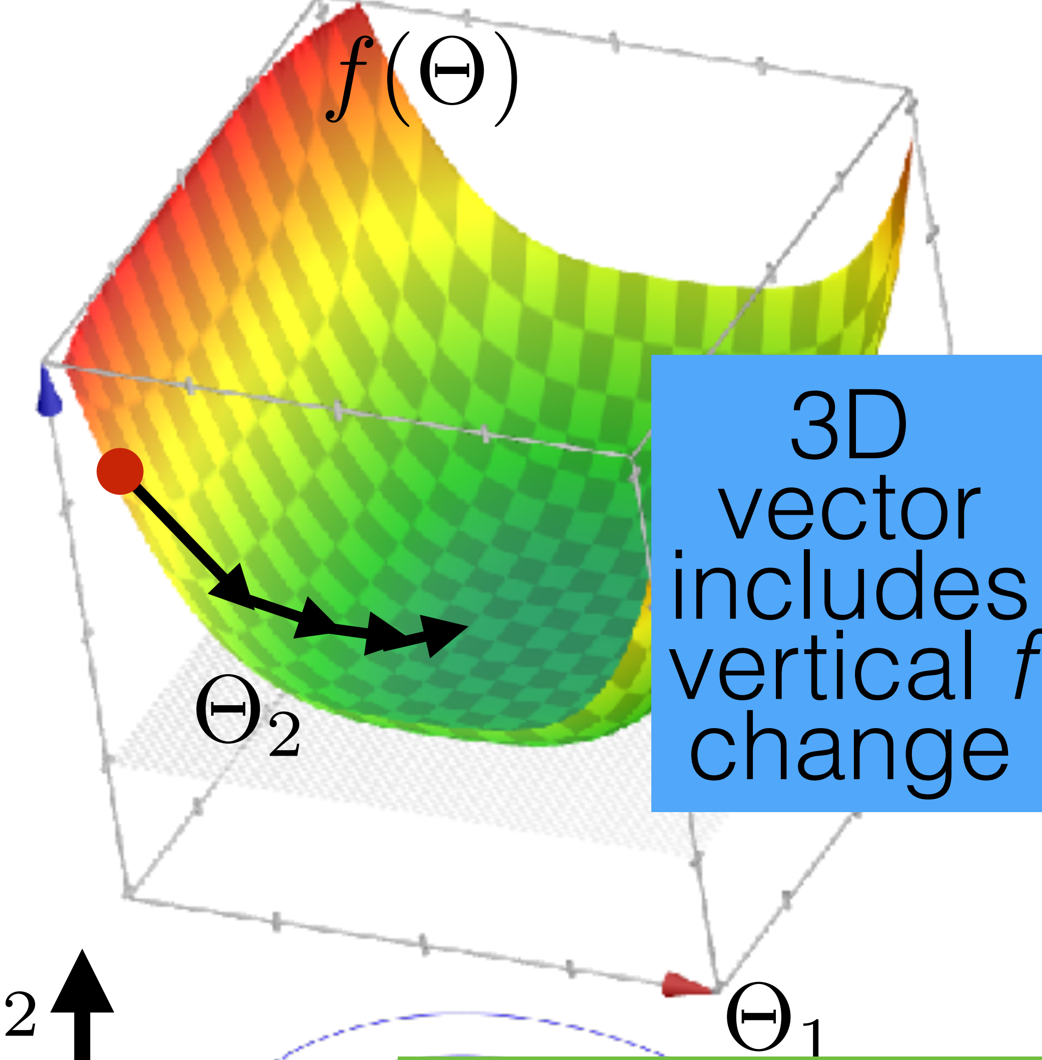




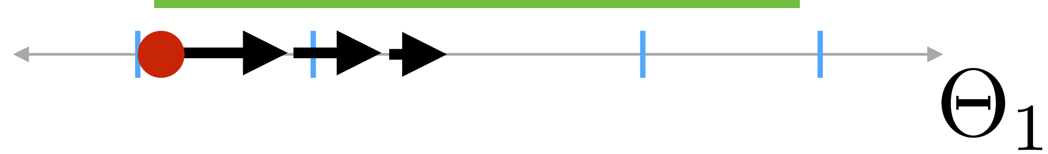


Gradient descent

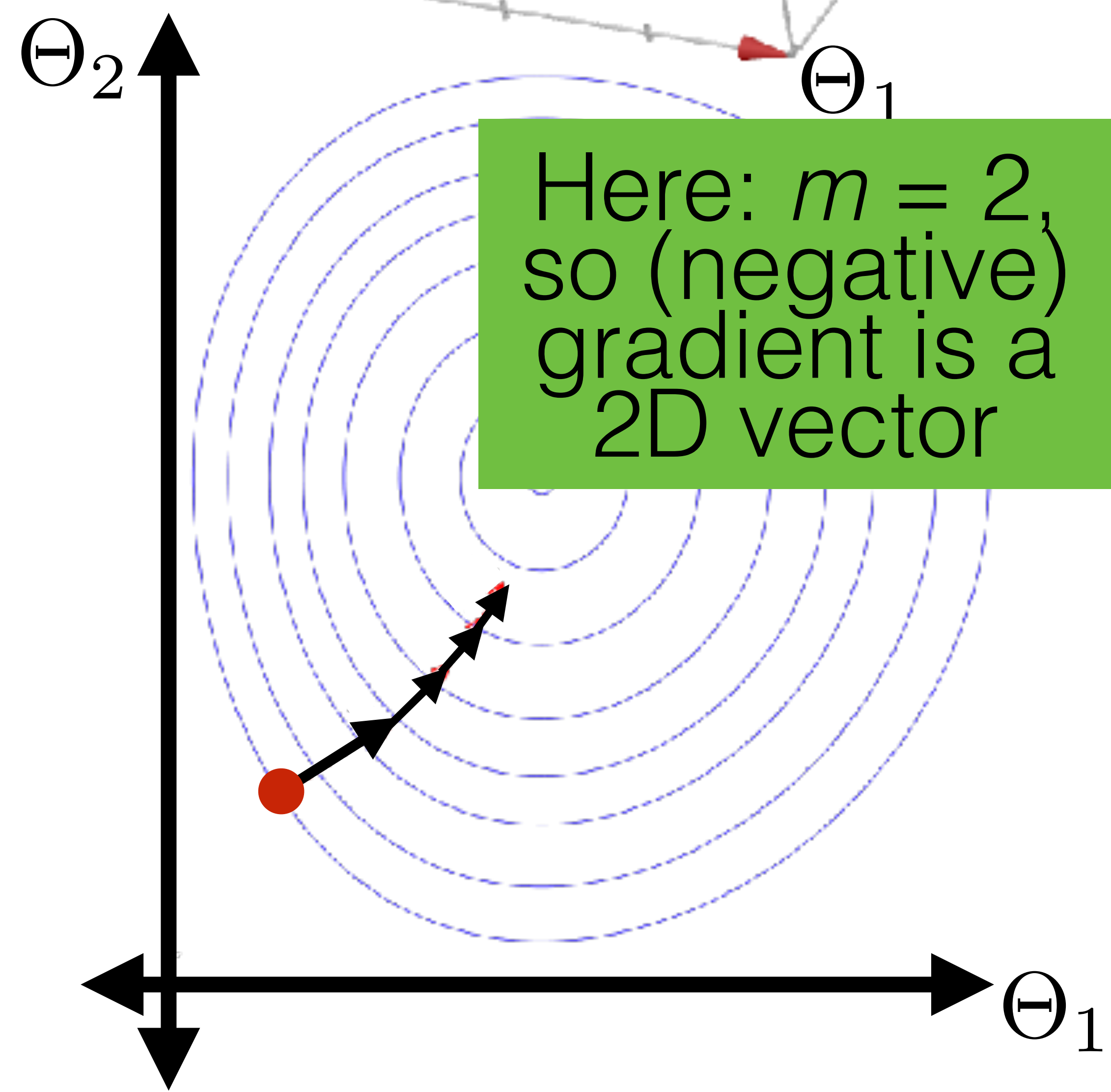




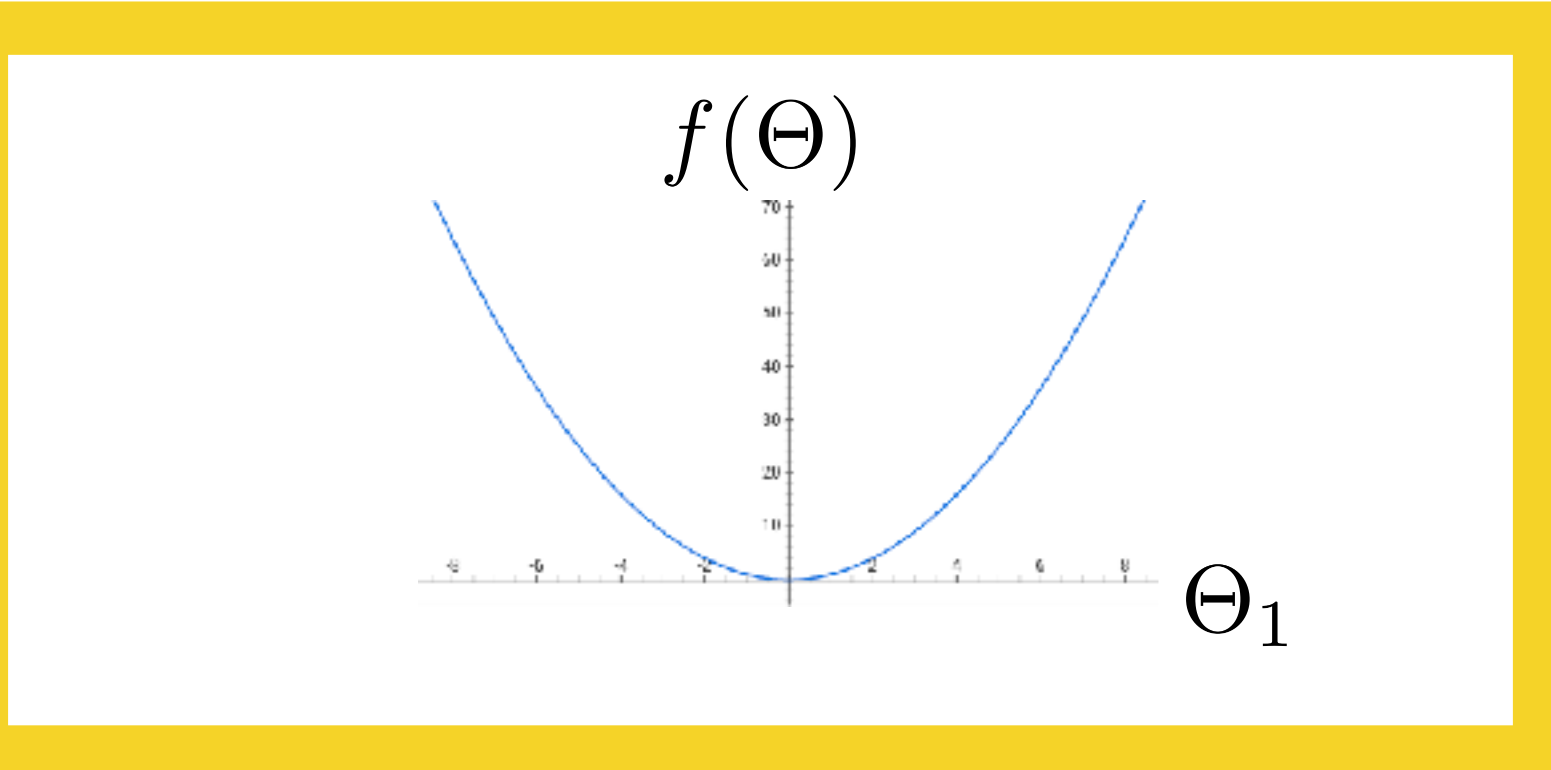
Here: $m = 1$, so (negative) gradient is a 1D vector

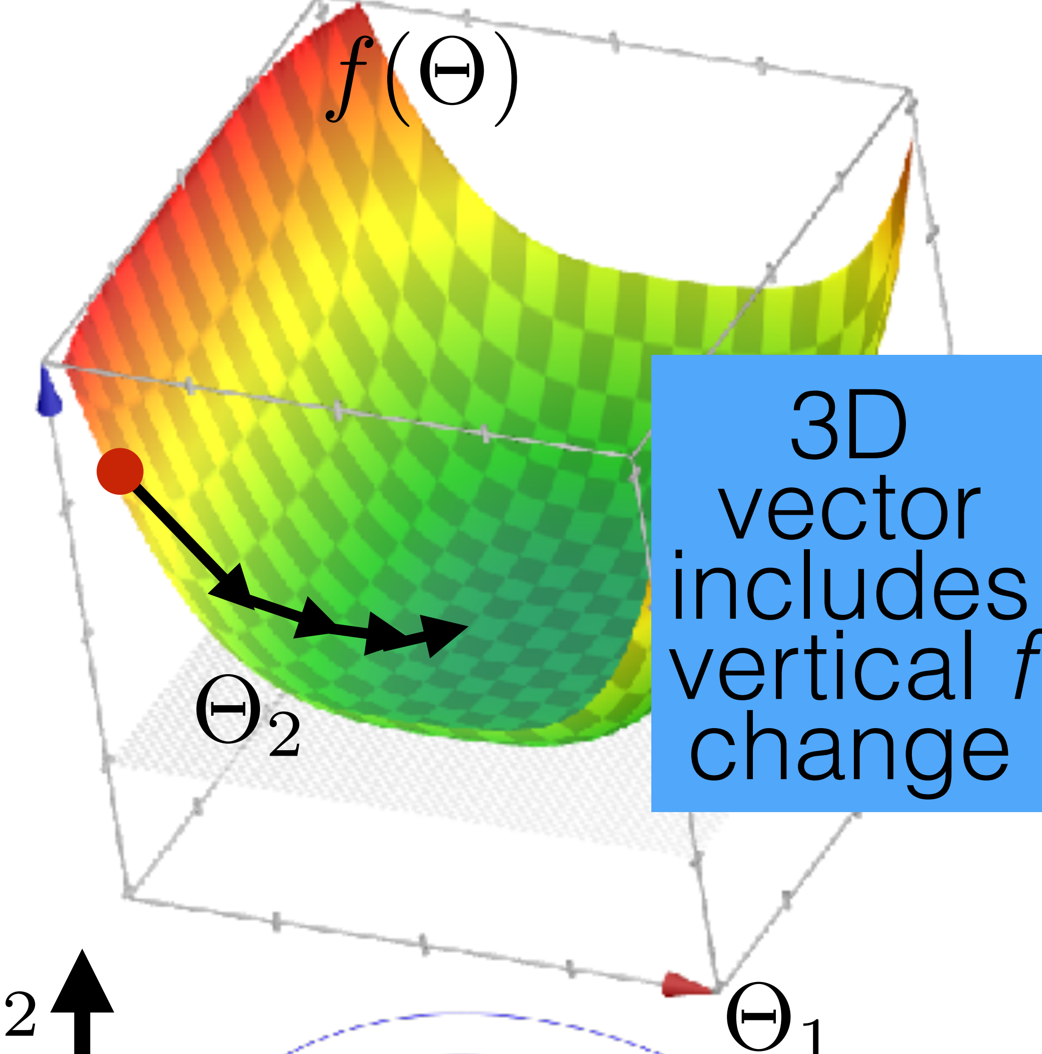


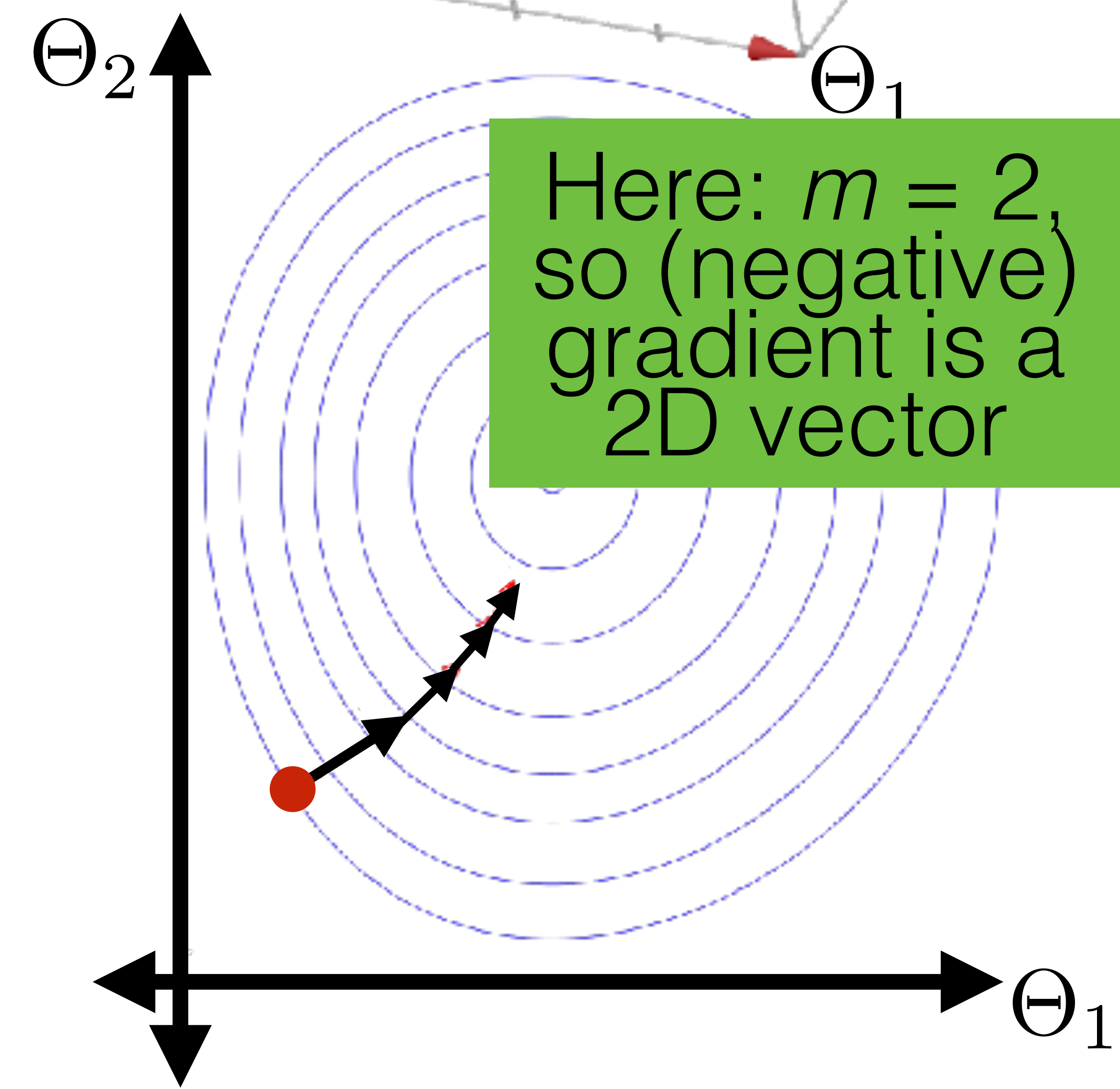
Here: $m = 2$, so (negative) gradient is a 2D vector



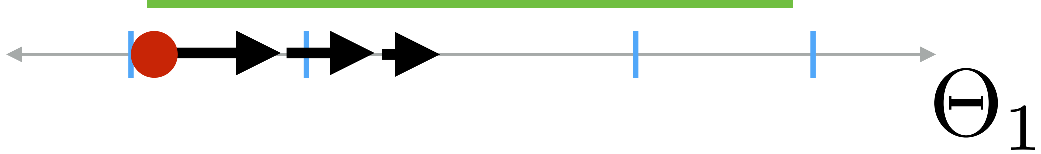
Gradient descent



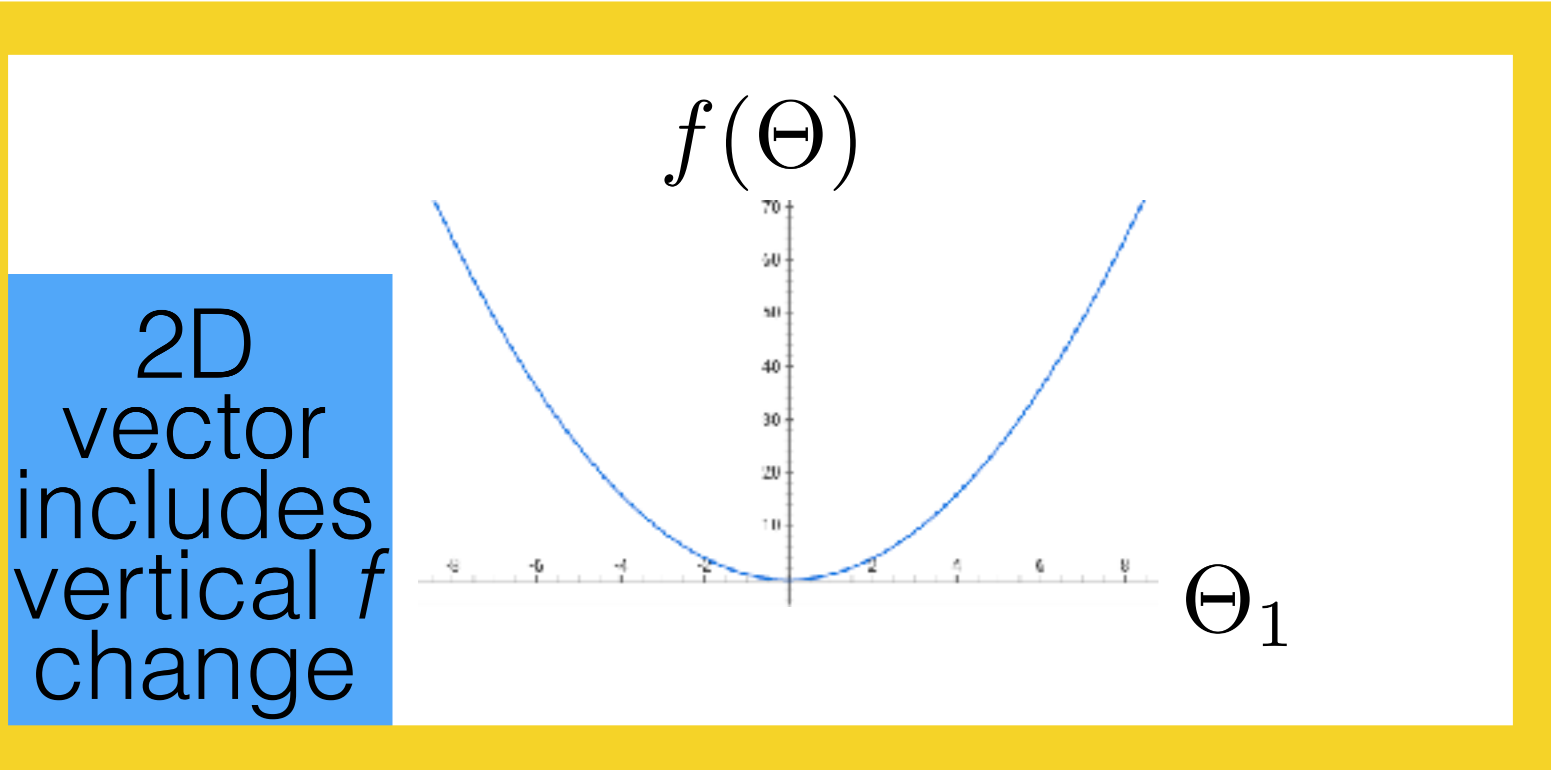


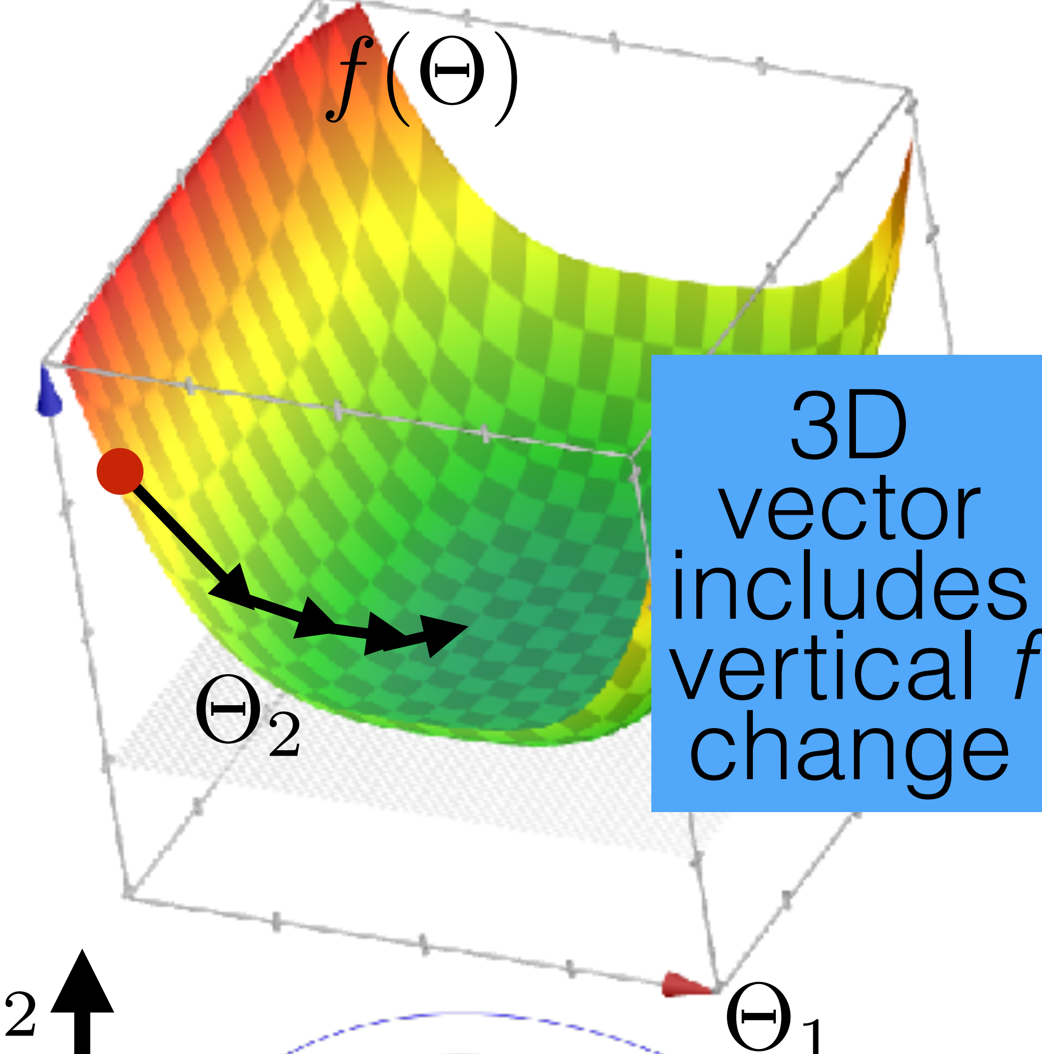


Here: m = 1, so (negative) gradient is a 1D vector

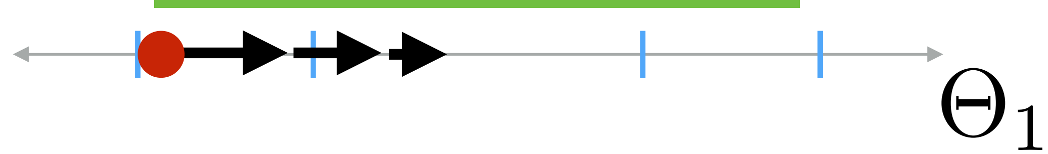


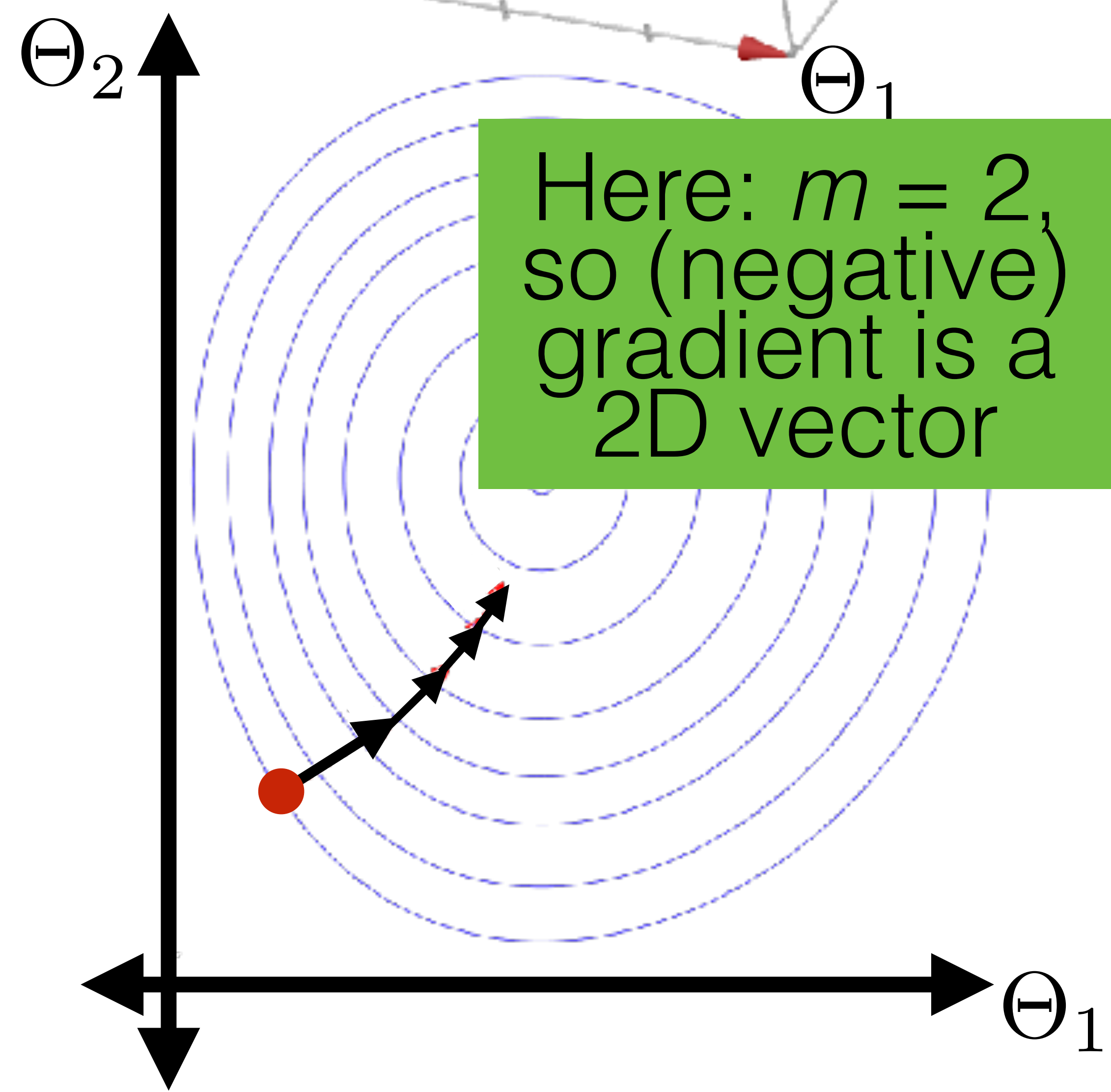
Gradient descent





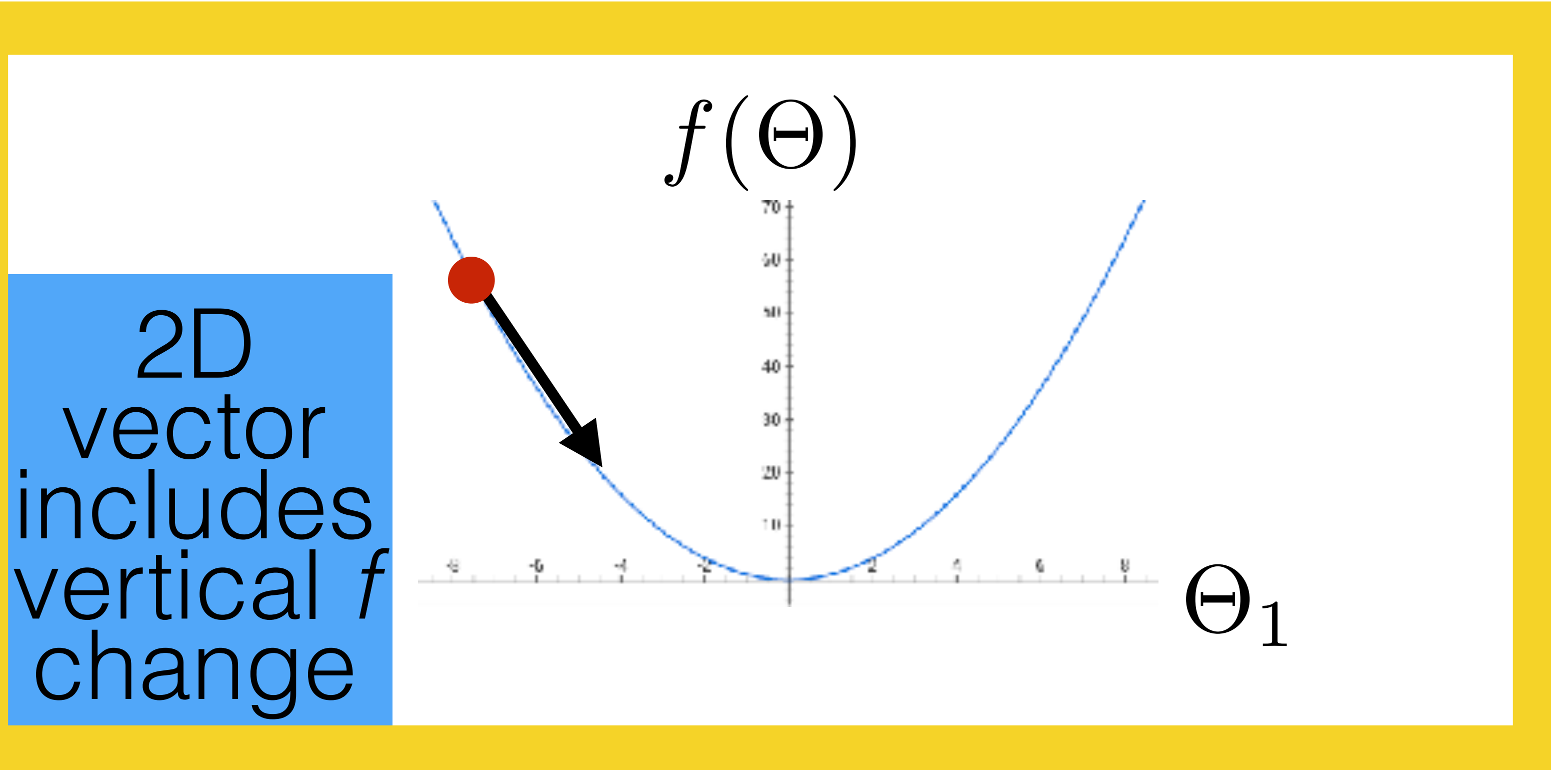
Here: m = 1, so (negative) gradient is a 1D vector

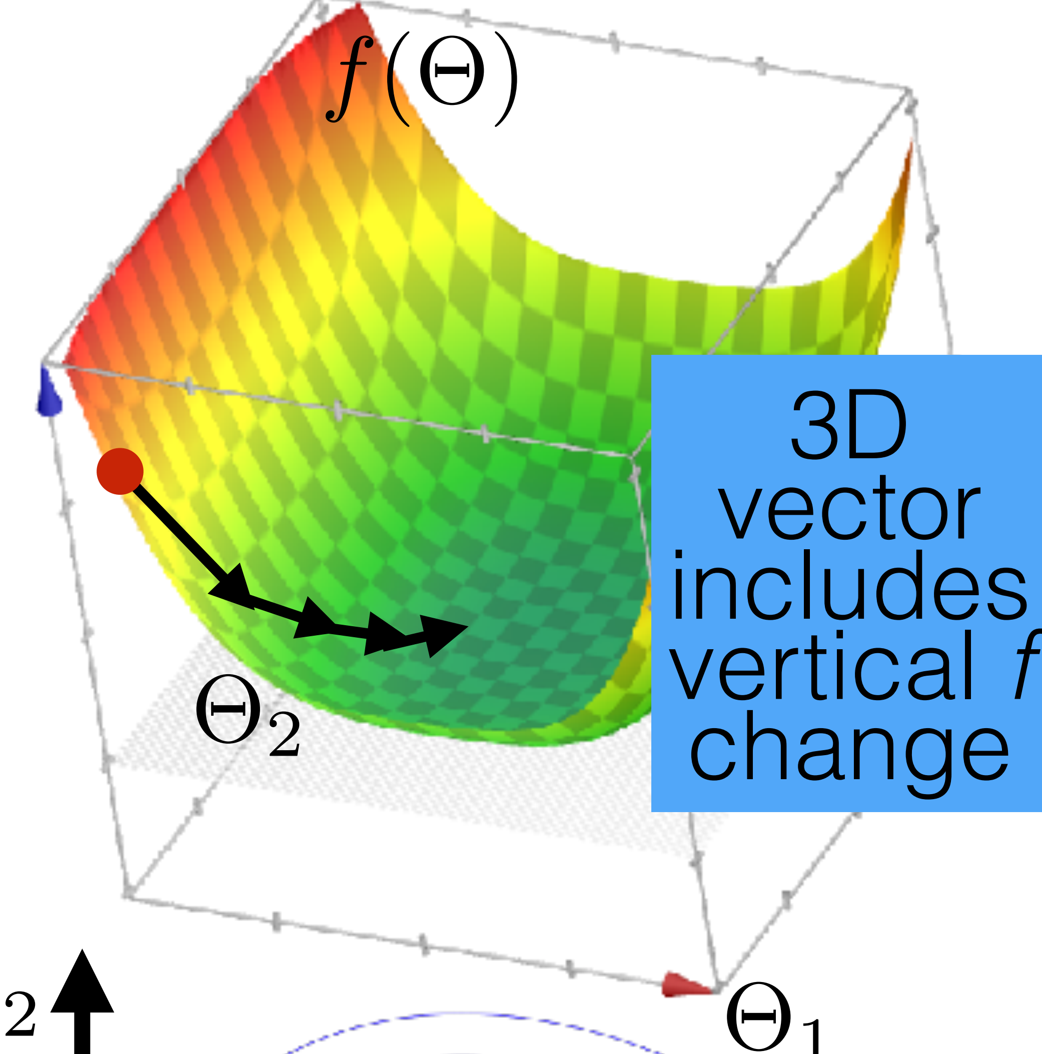


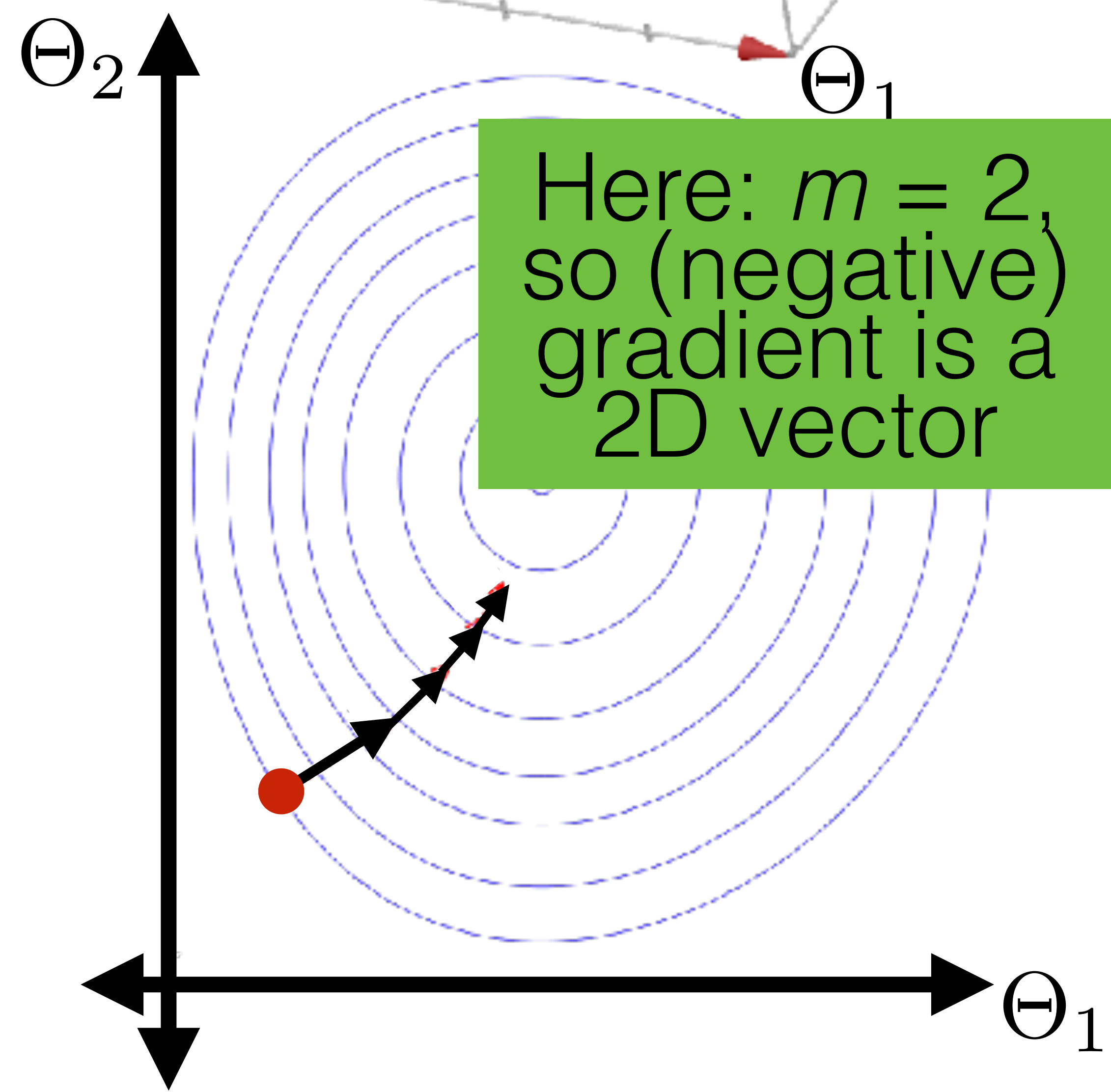


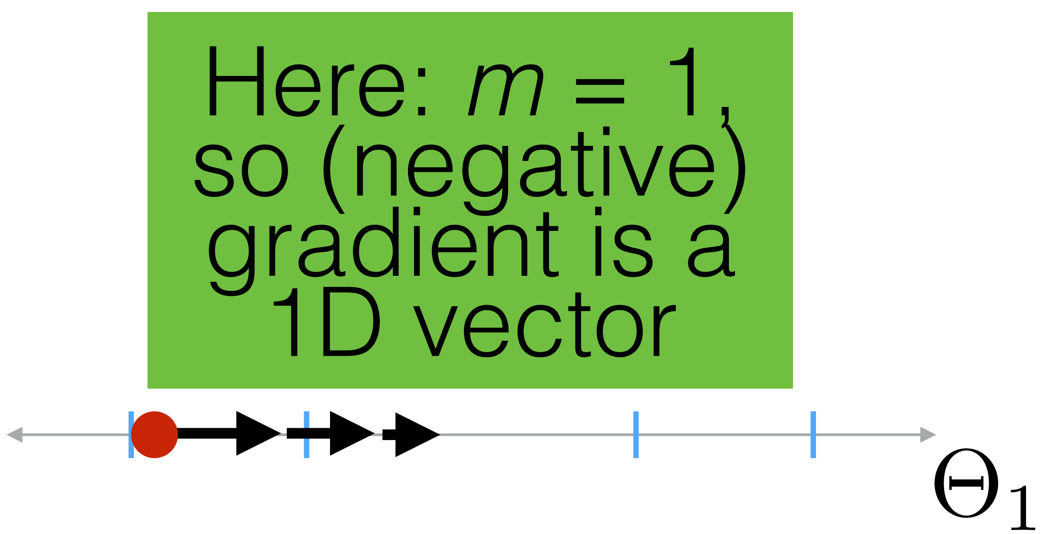
Here: m = 2, so (negative) gradient is a 2D vector

Gradient descent

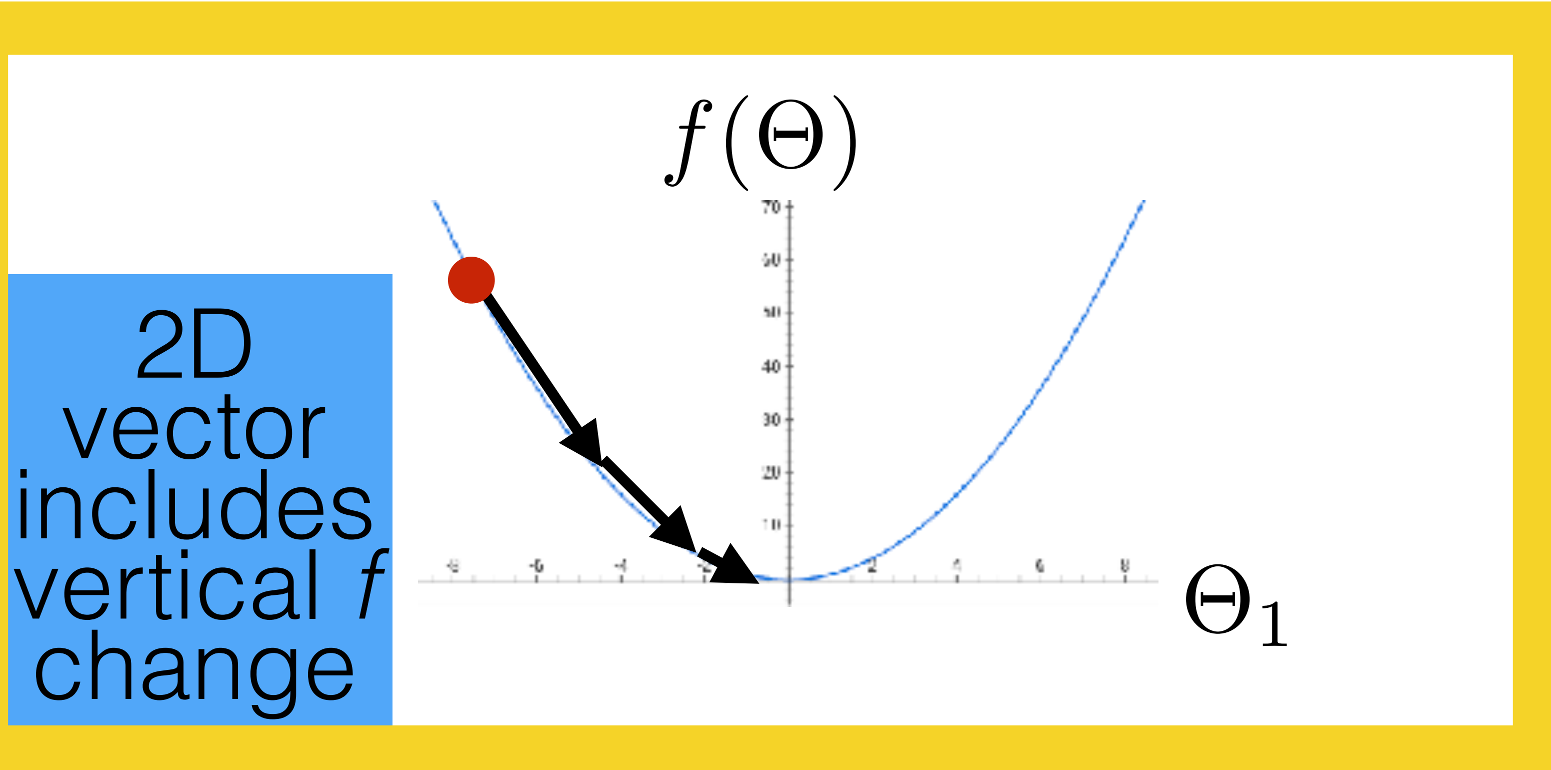


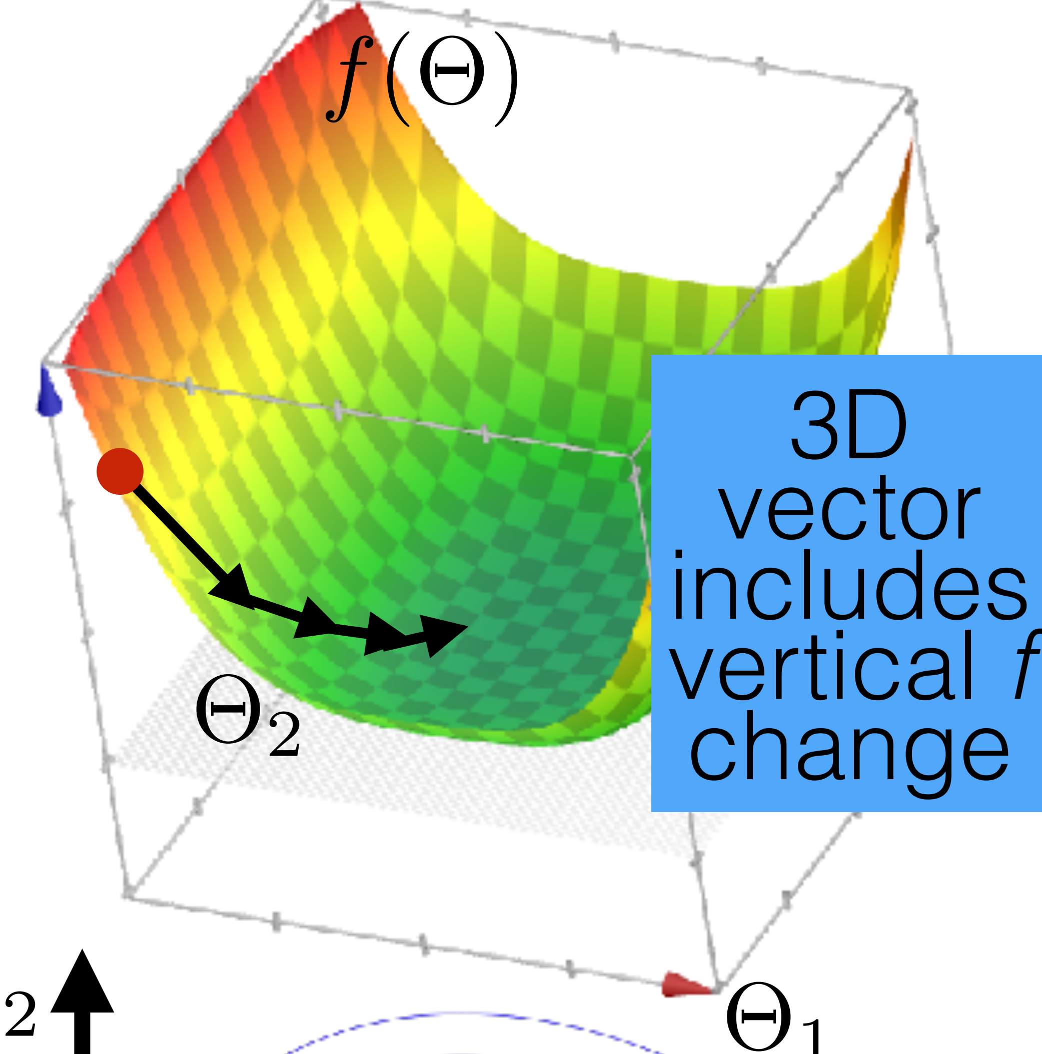




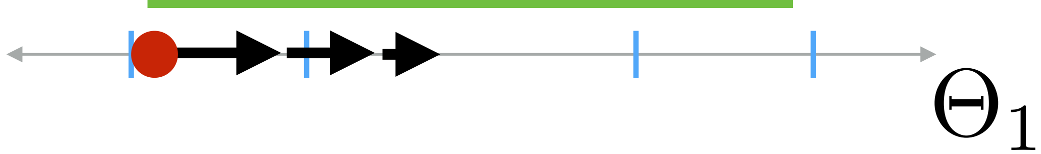


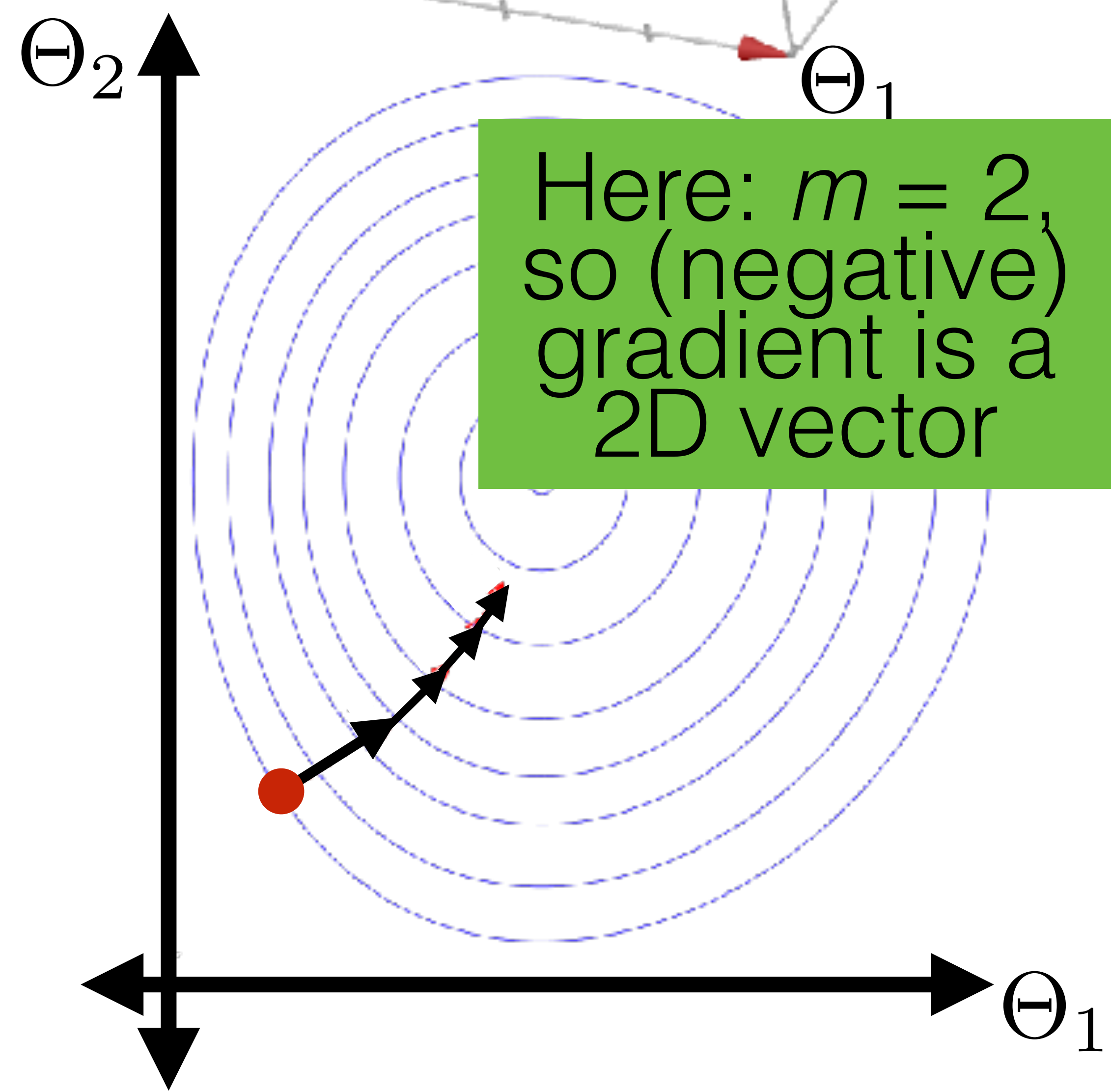
Gradient descent





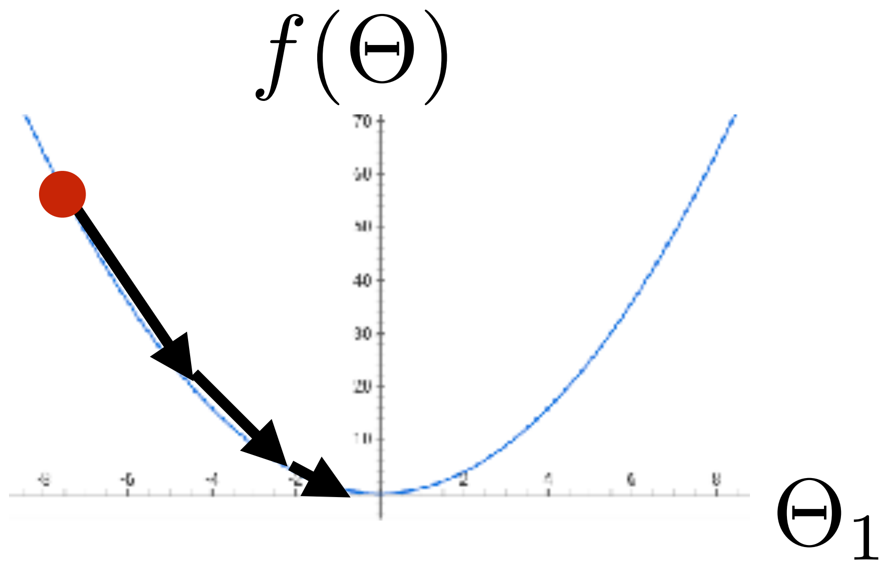
Here: m = 1, so (negative) gradient is a 1D vector





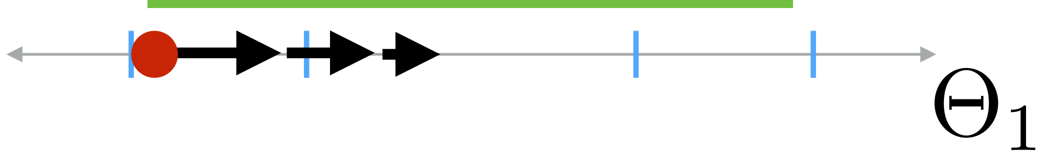
Here: m = 2, so (negative) gradient is a 2D vector

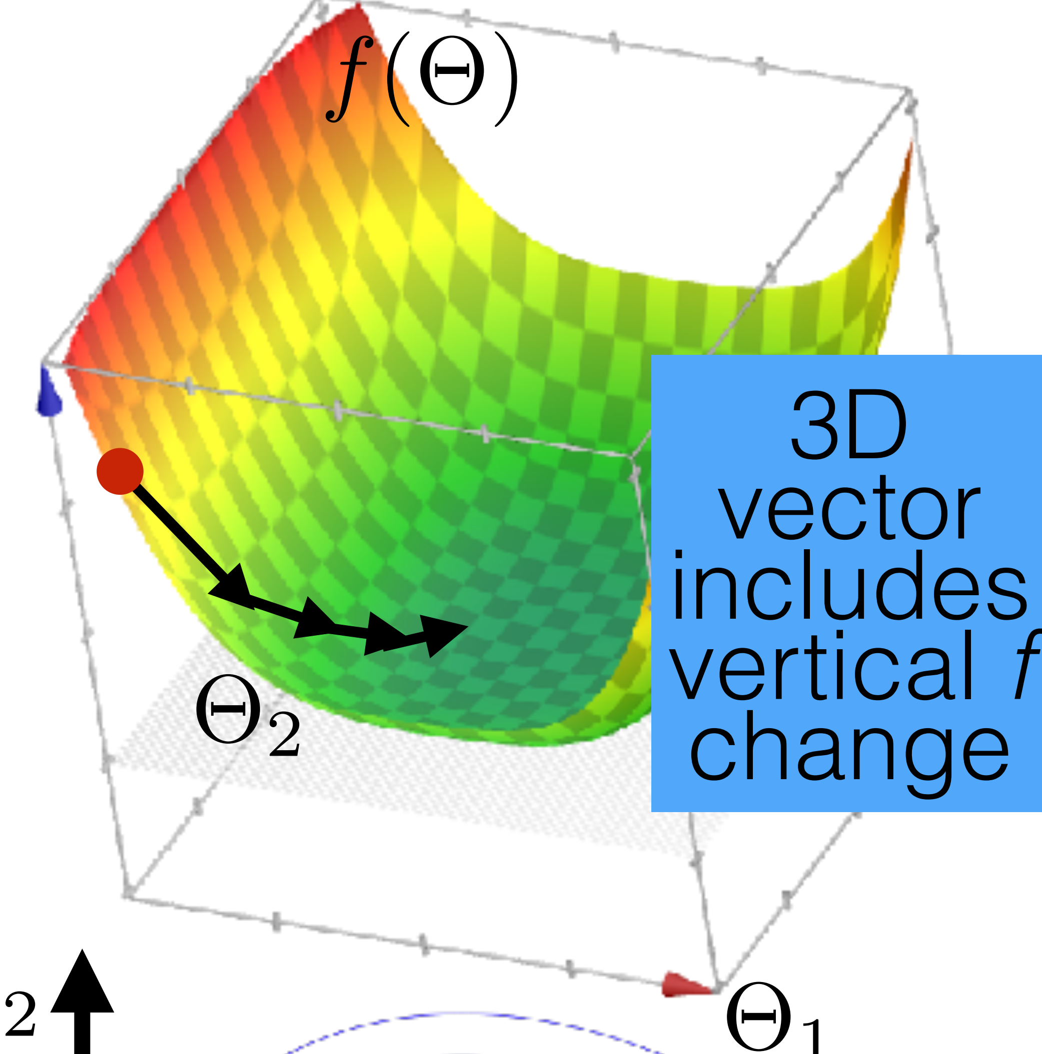
Gradient descent



2D vector includes vertical f change

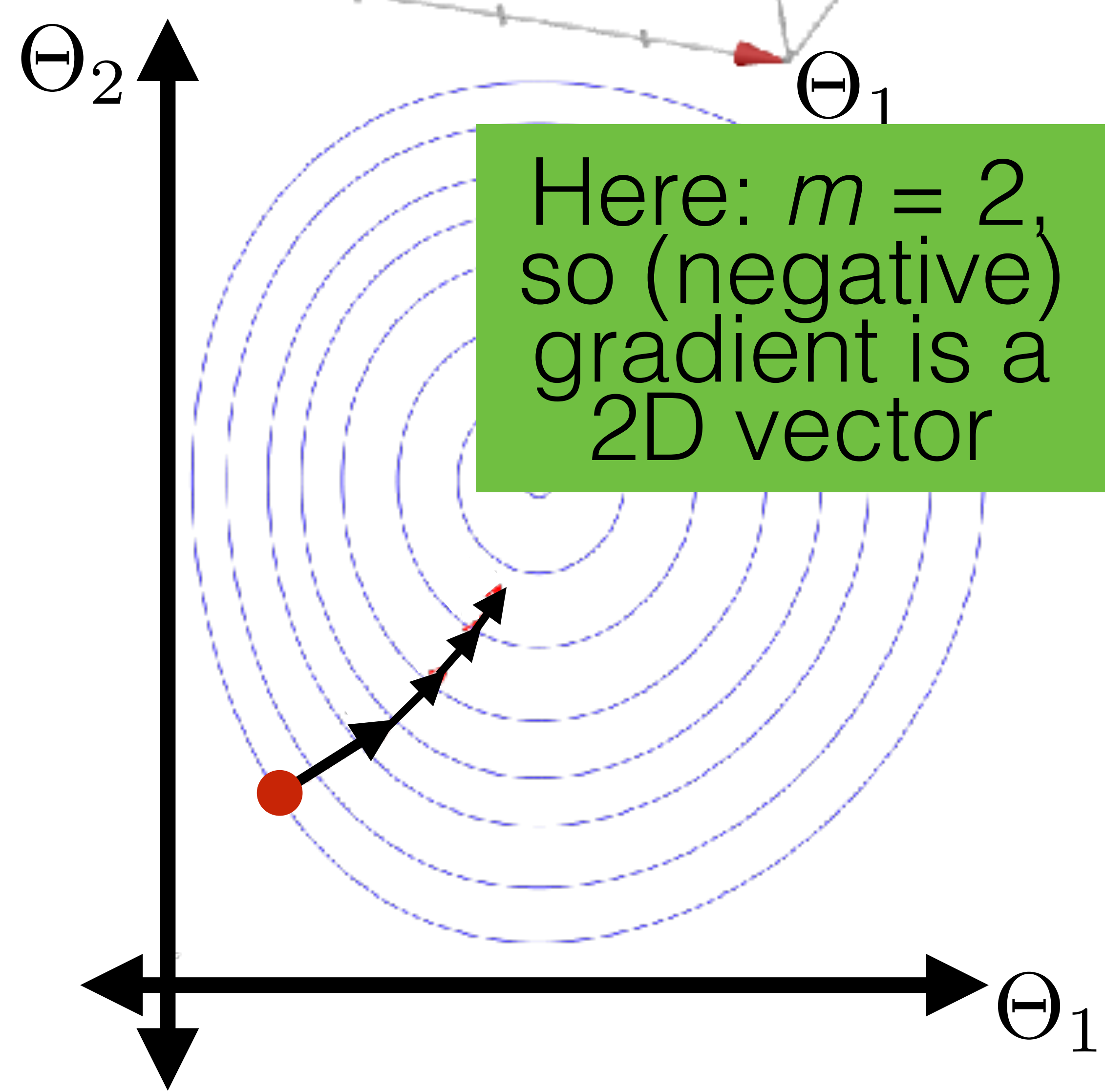
Here: m = 1, so (negative) gradient is a 1D vector





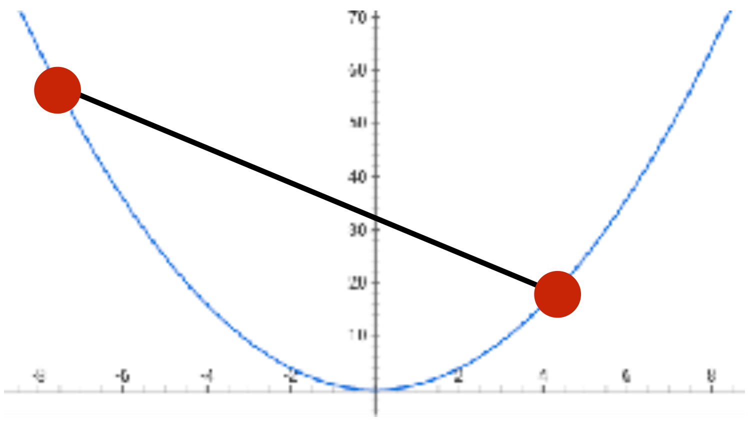
3D vector includes vertical f change

Here: m = 2, so (negative) gradient is a 2D vector



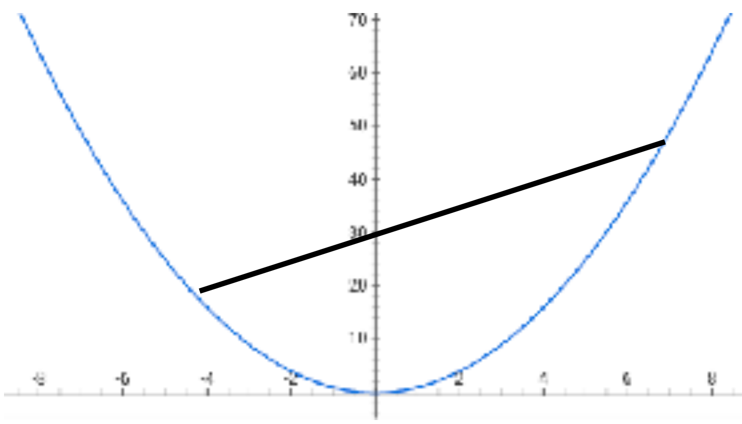
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



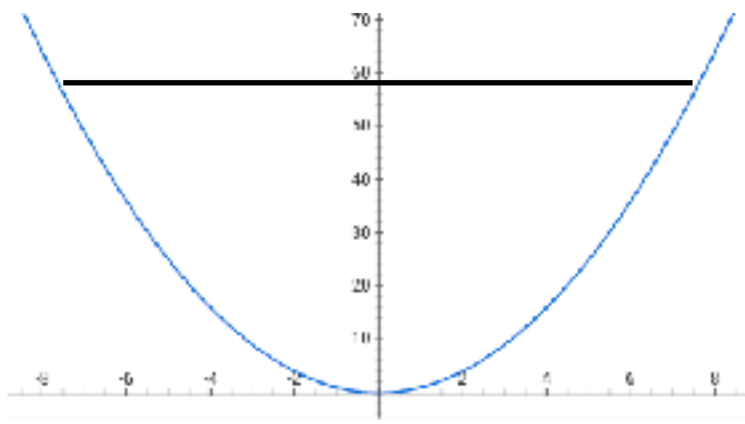
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



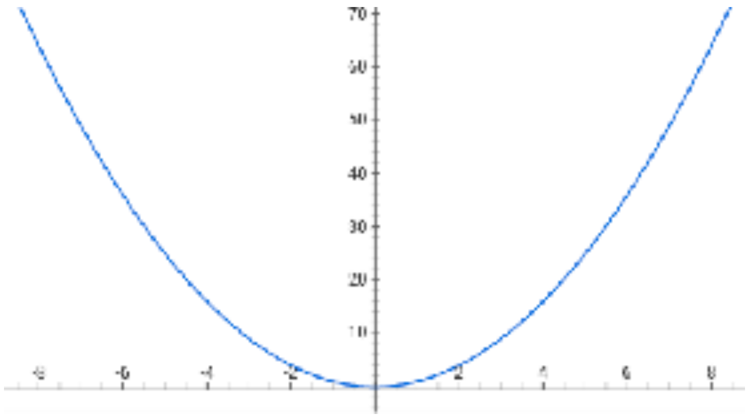
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



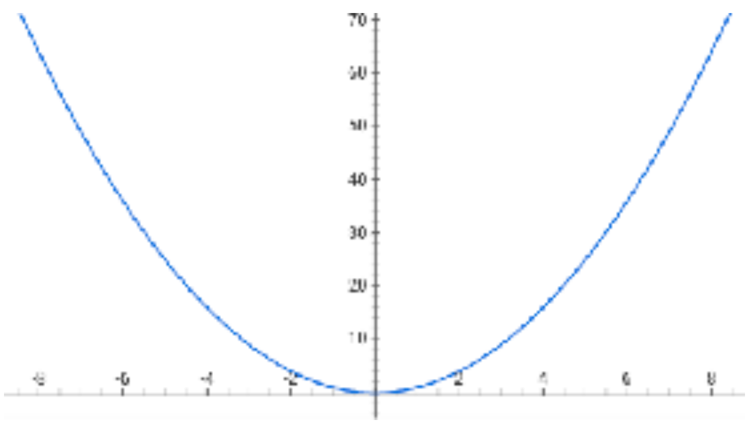
Gradient descent properties

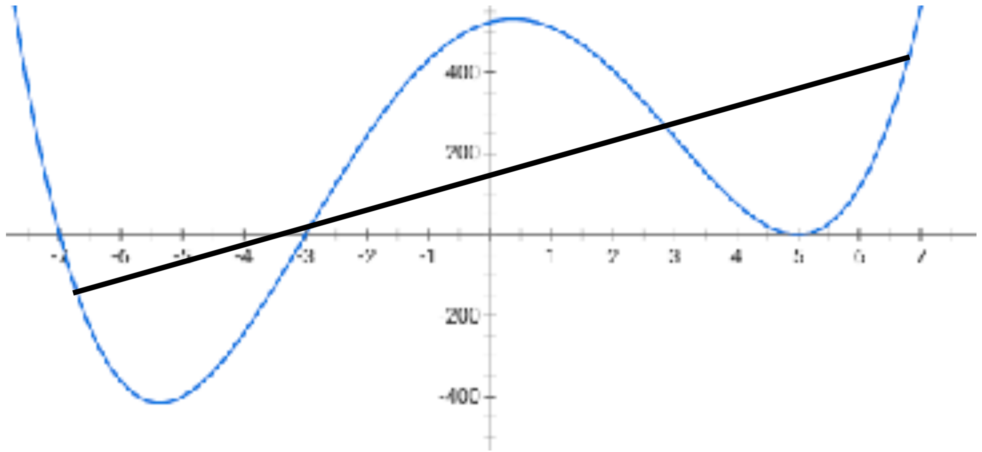
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



Gradient descent properties

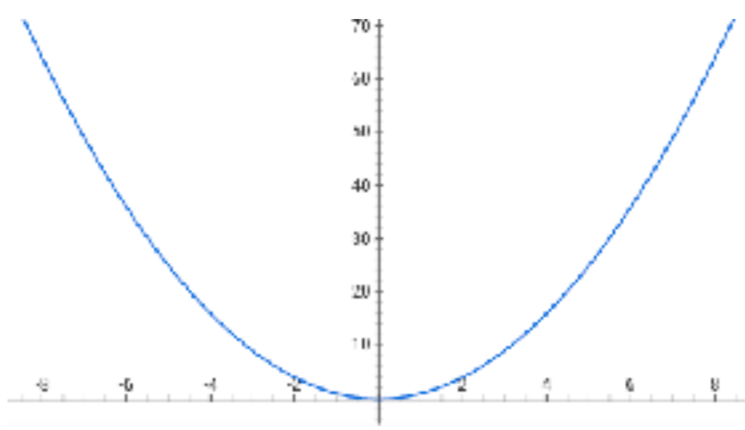
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

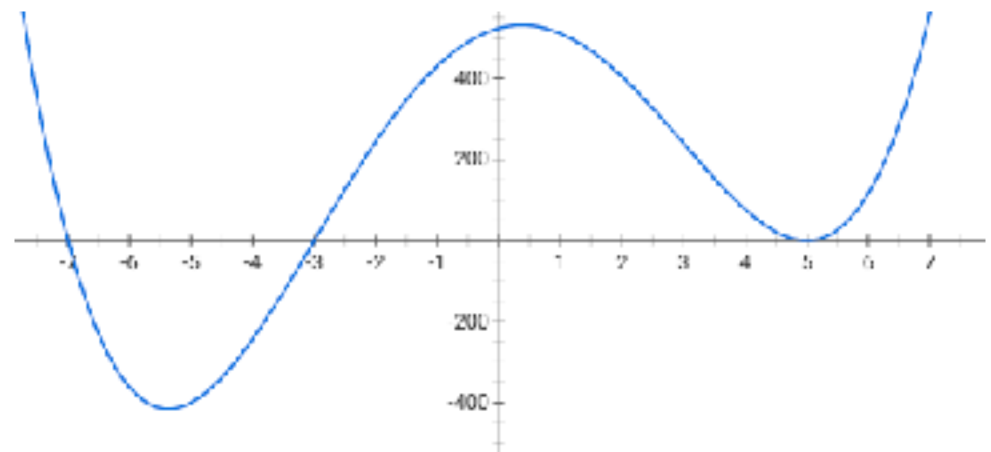


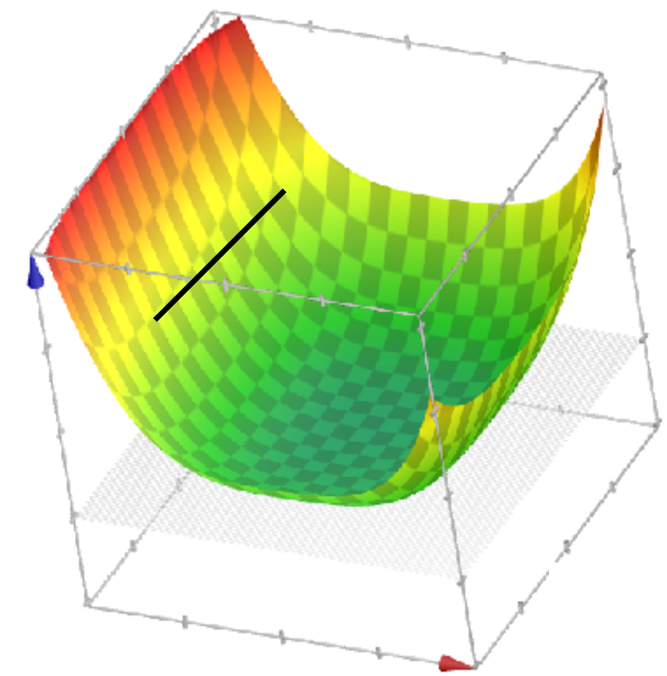


Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

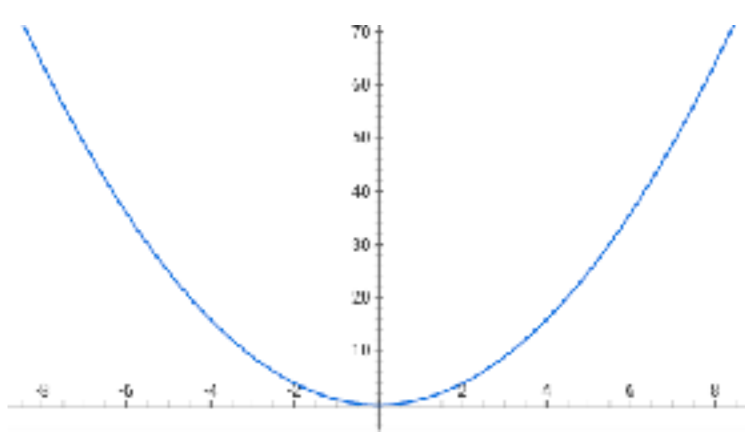


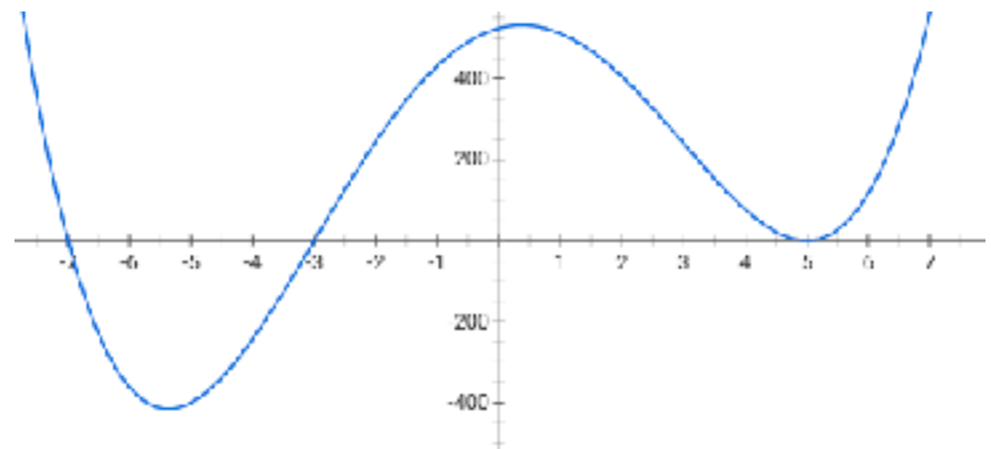


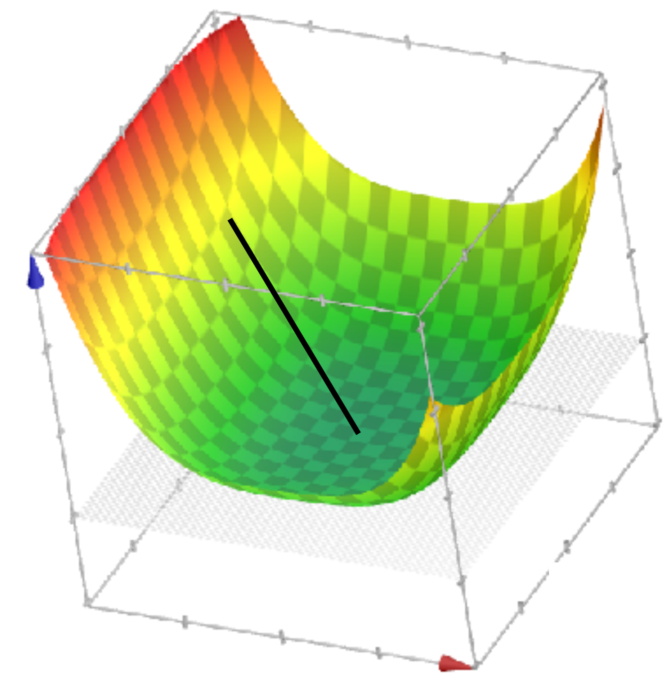


Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

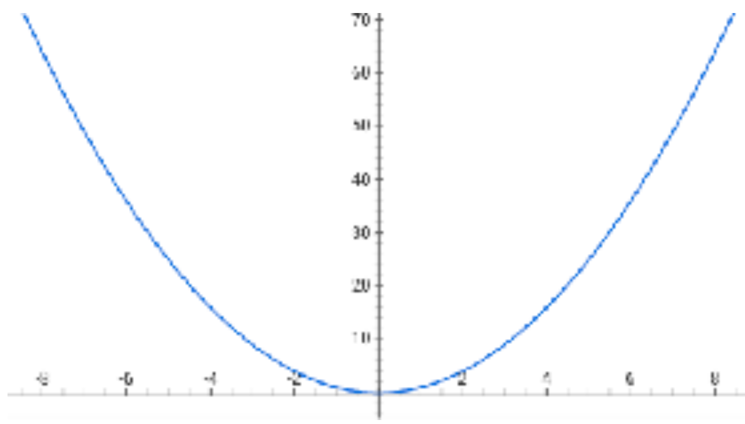


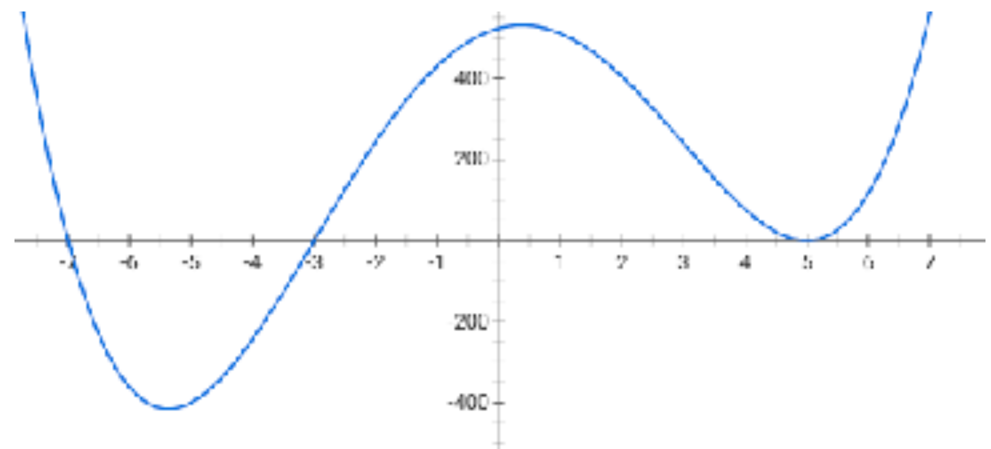


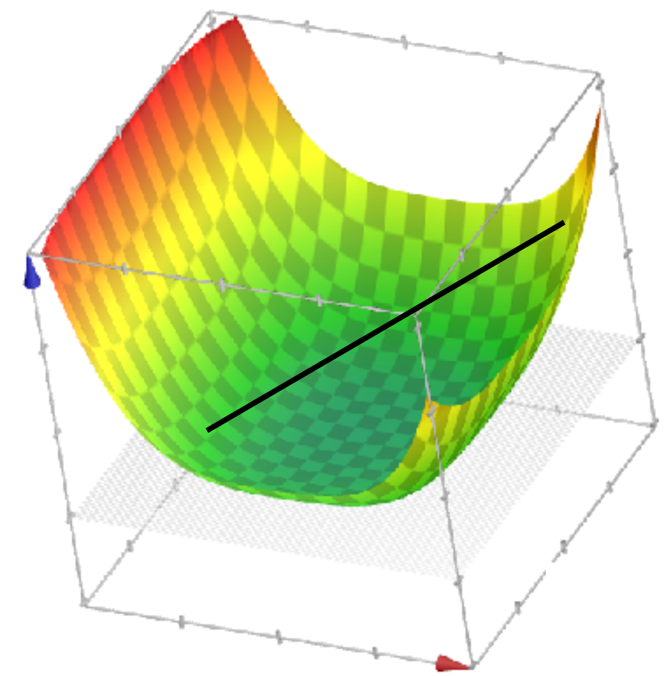


Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

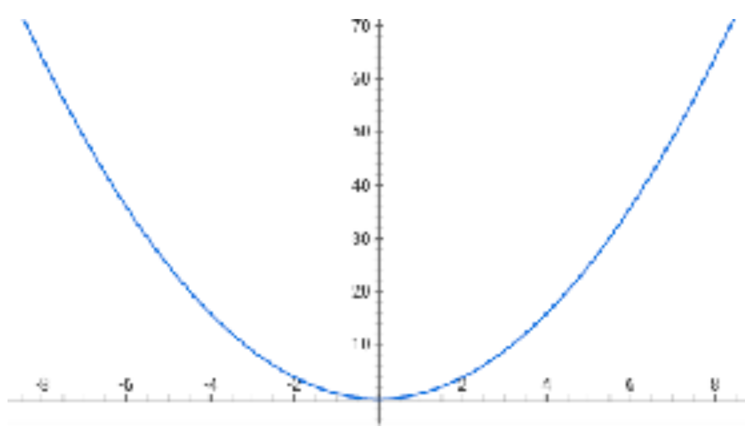


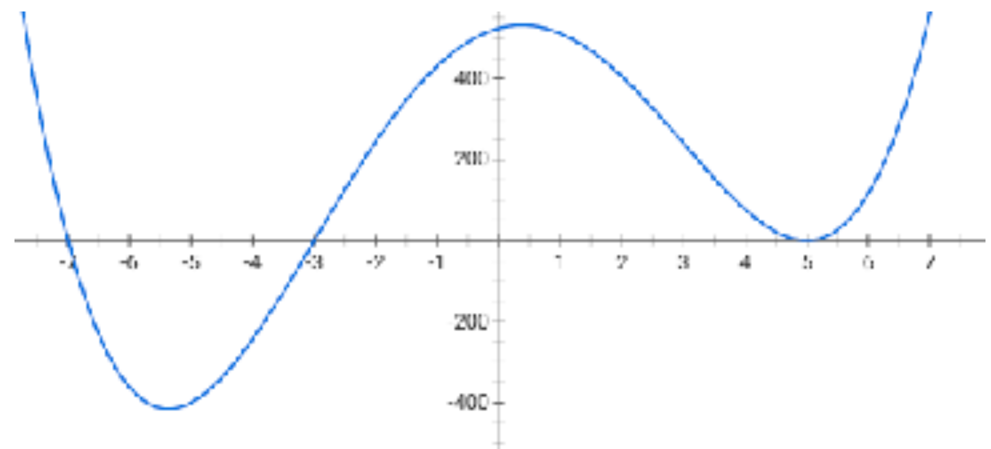


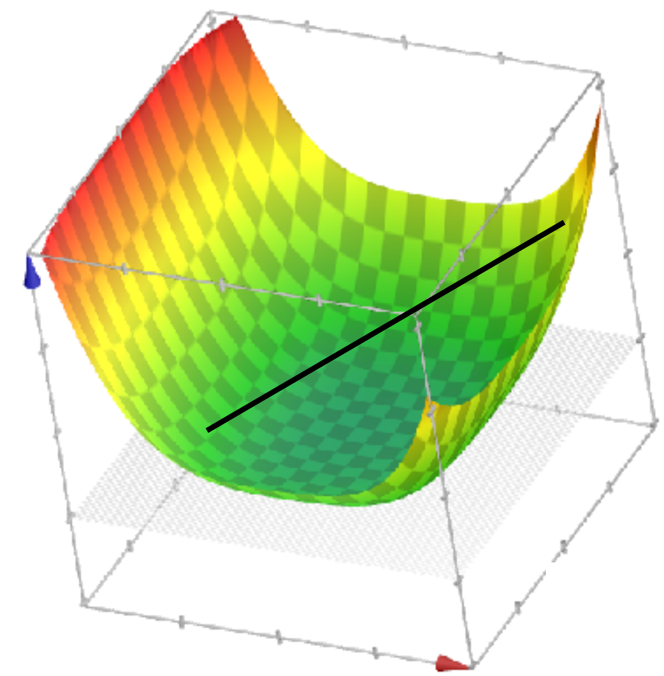


Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

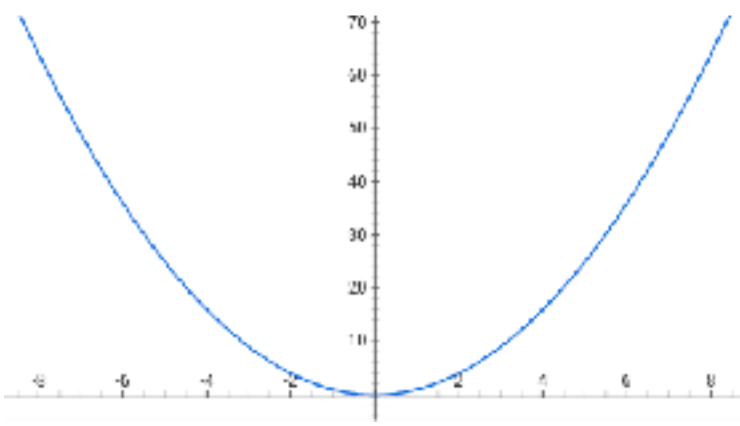


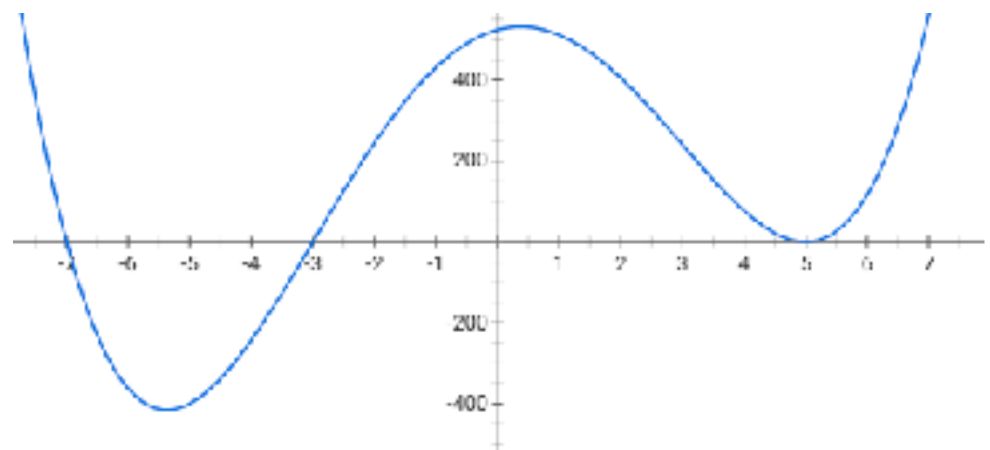


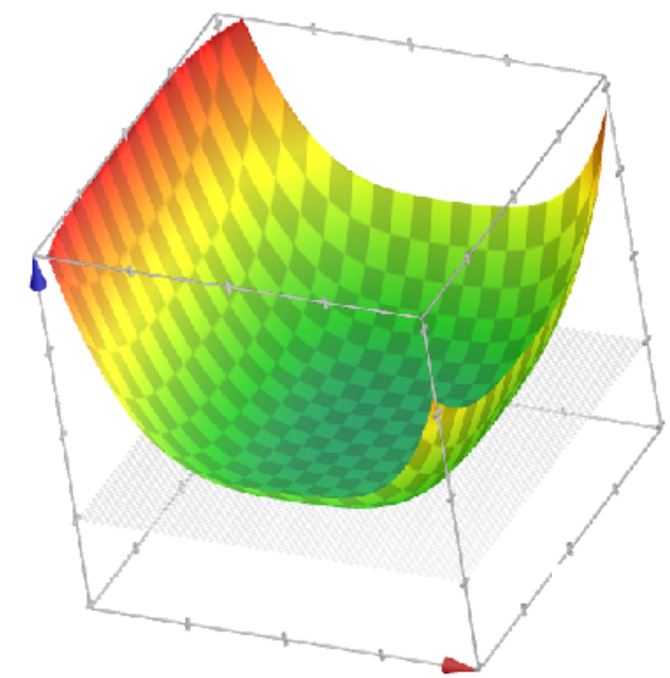


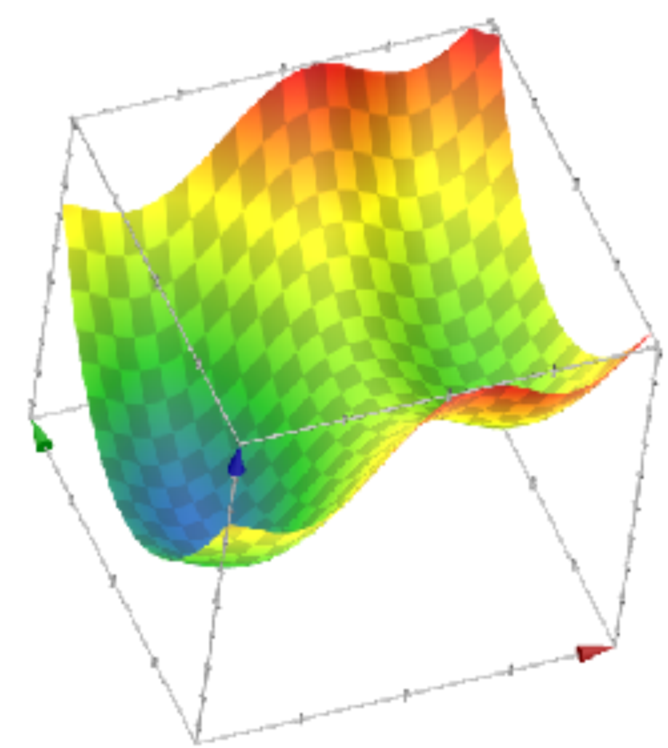
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



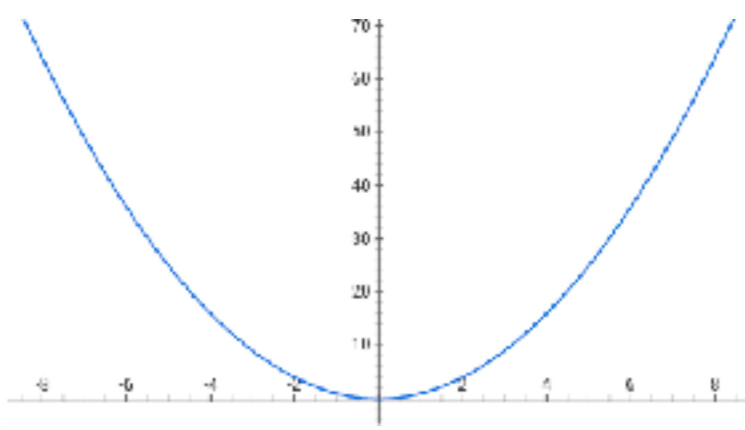


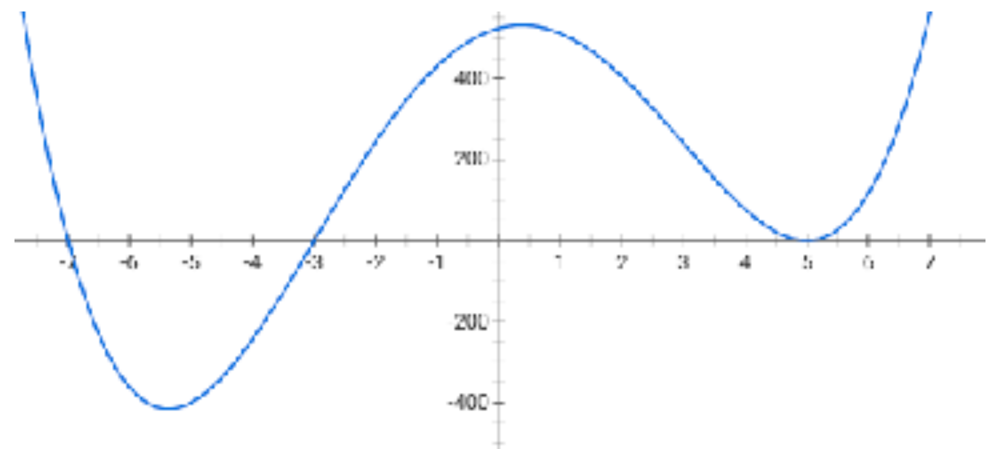


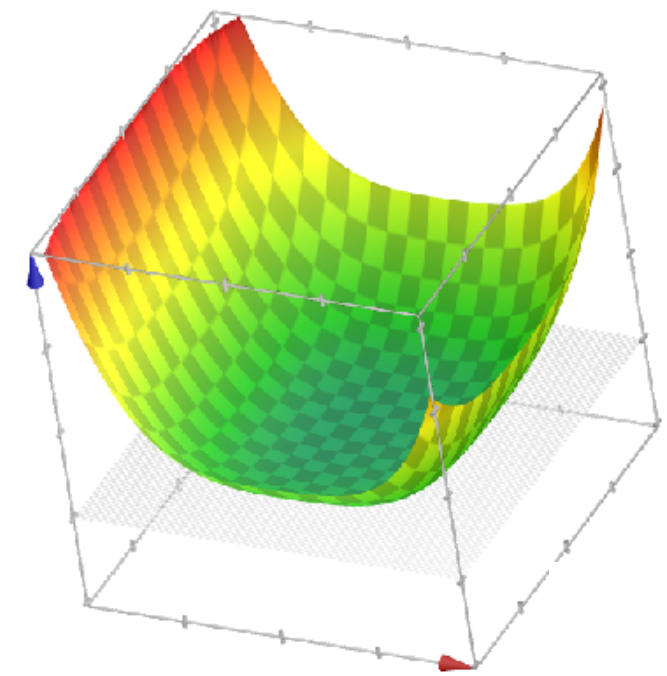


Gradient descent properties

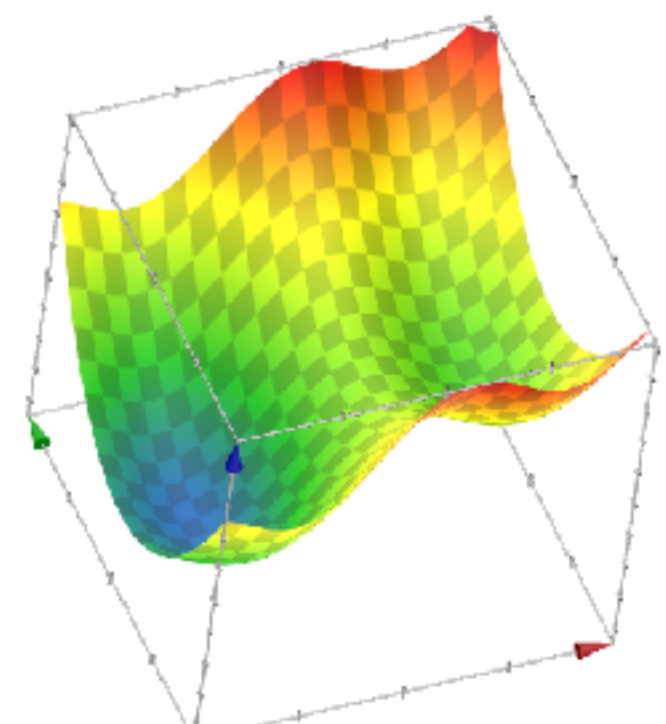
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





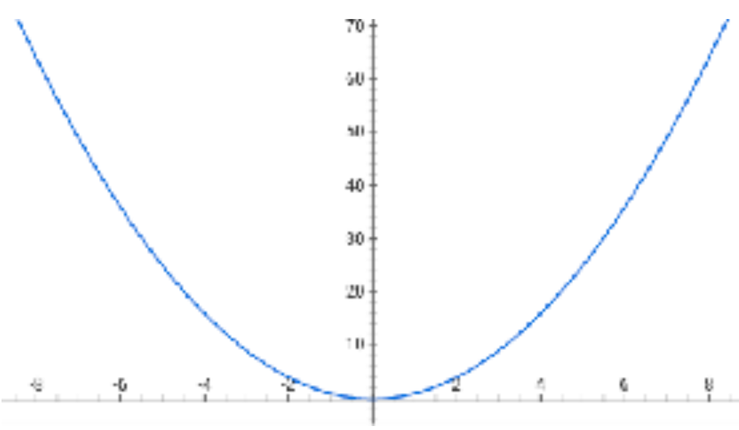


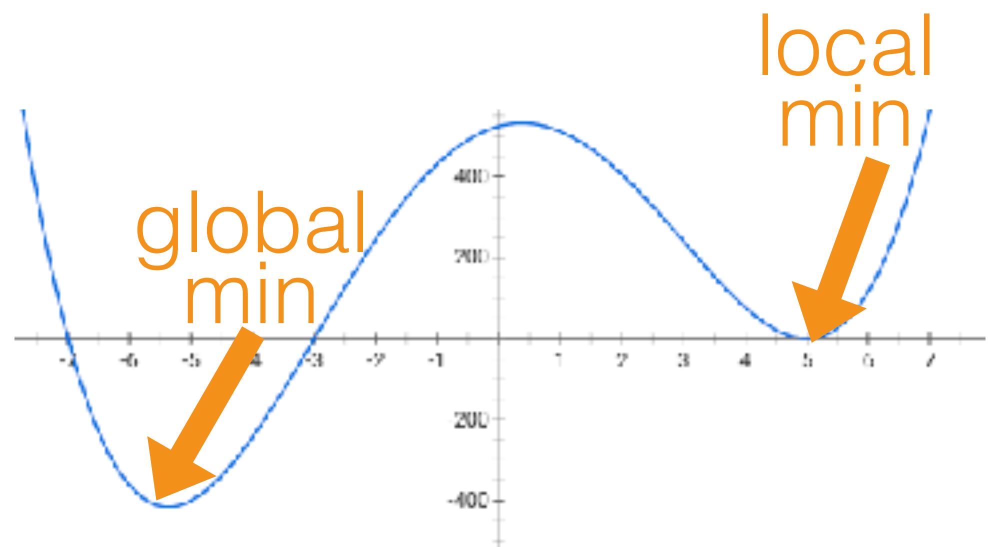
- **Theorem**: Gradient descent performance

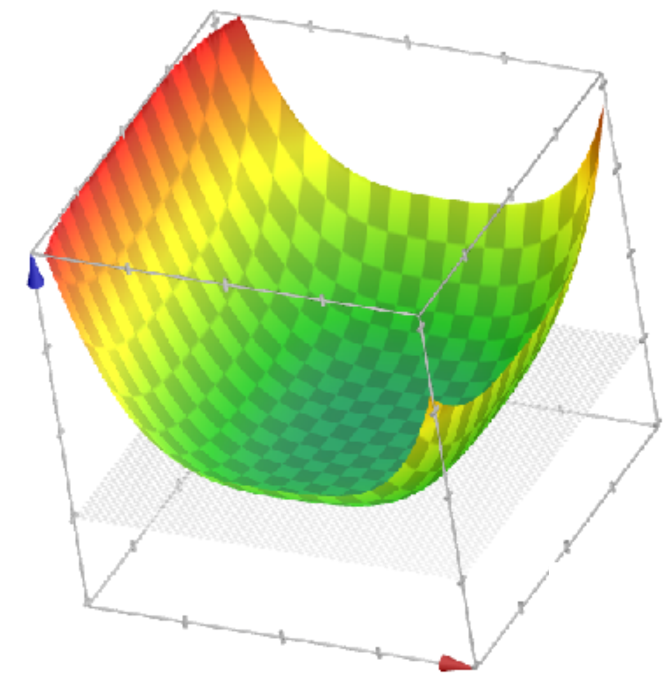


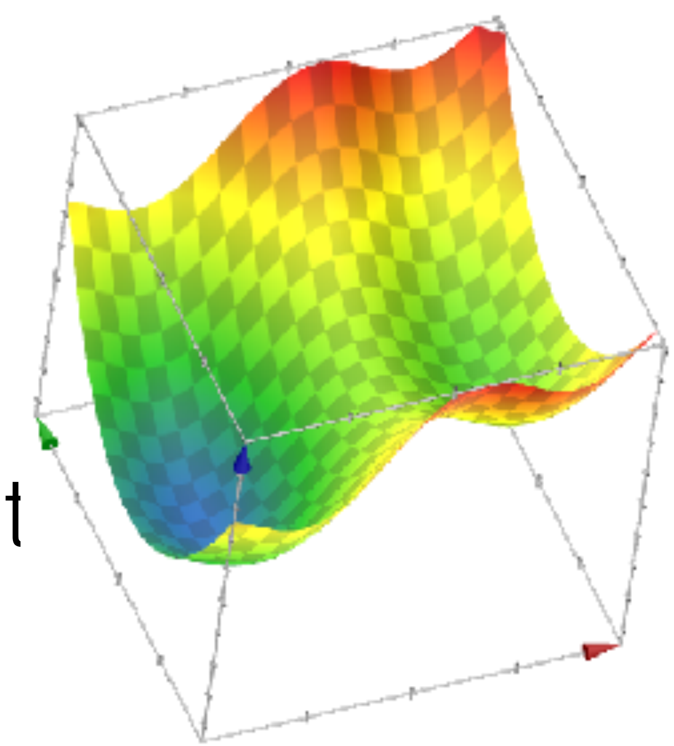
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





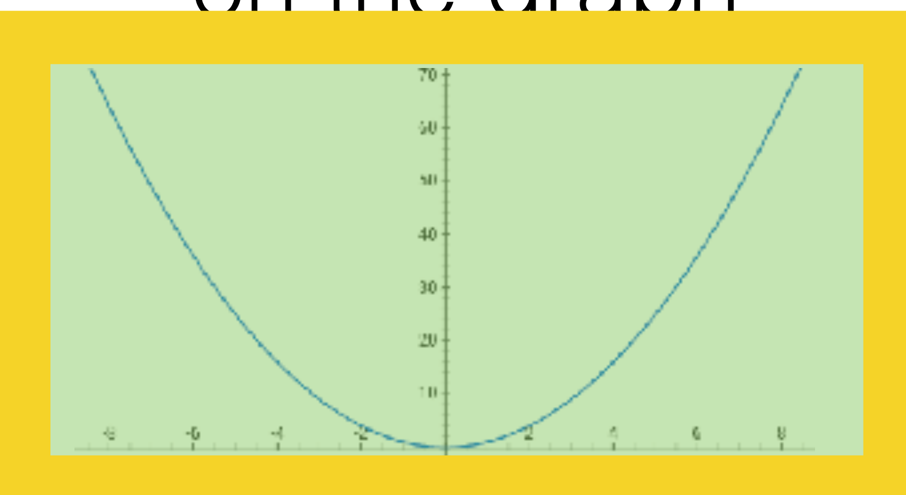


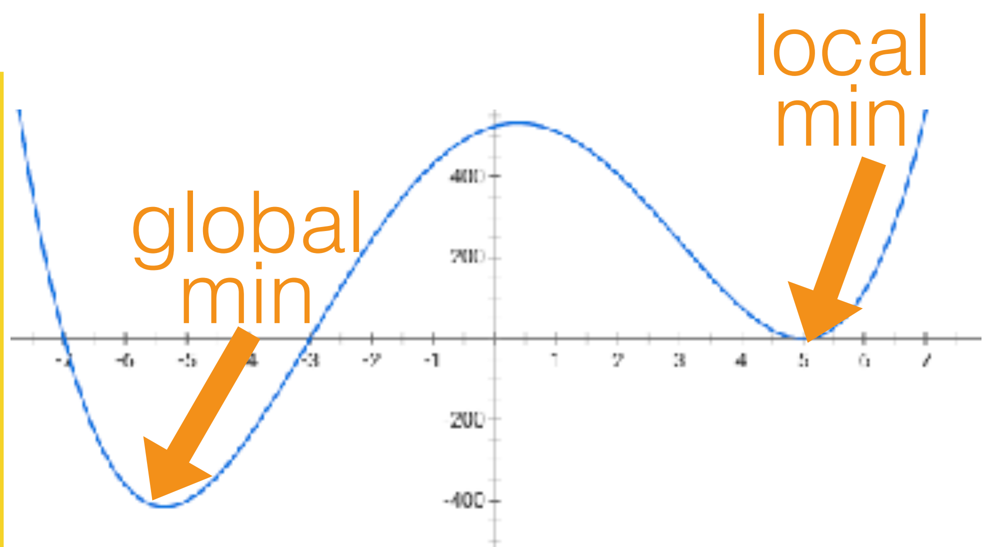


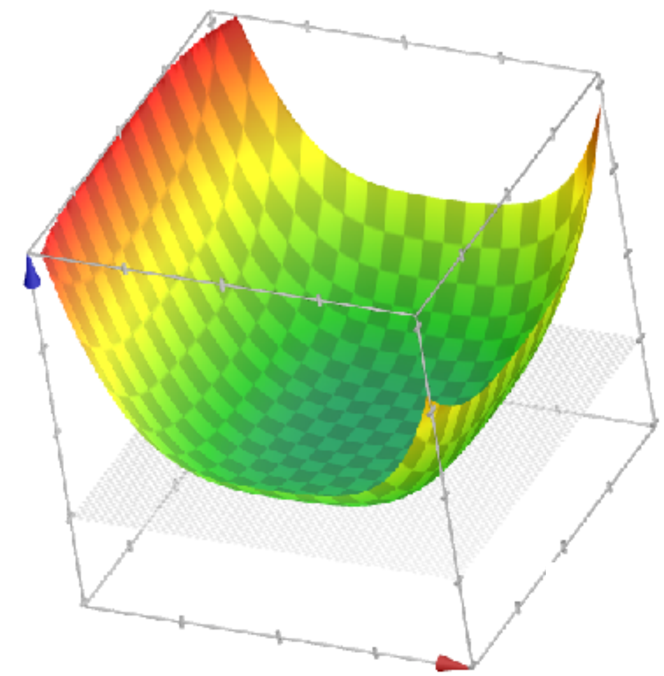
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

Gradient descent properties

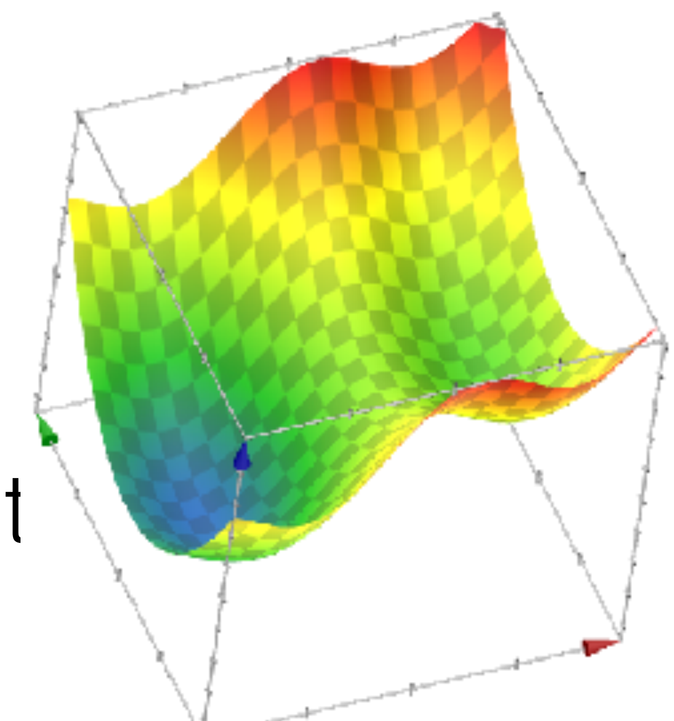
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





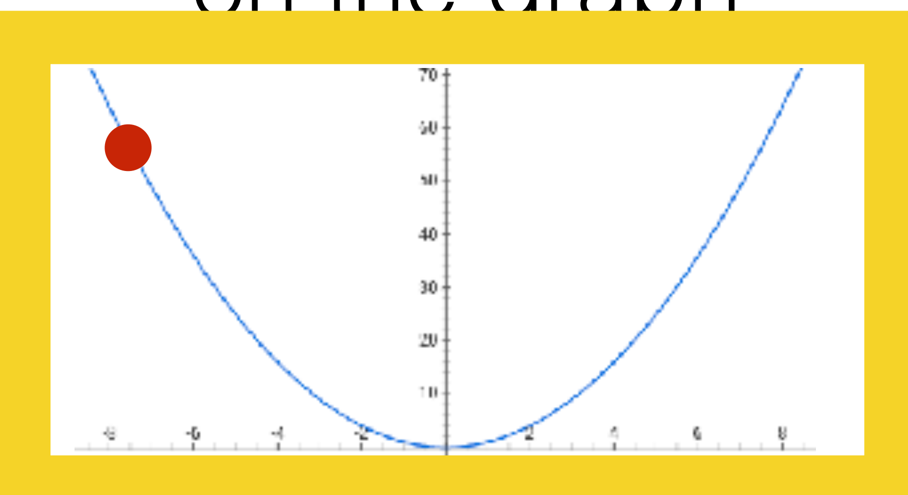


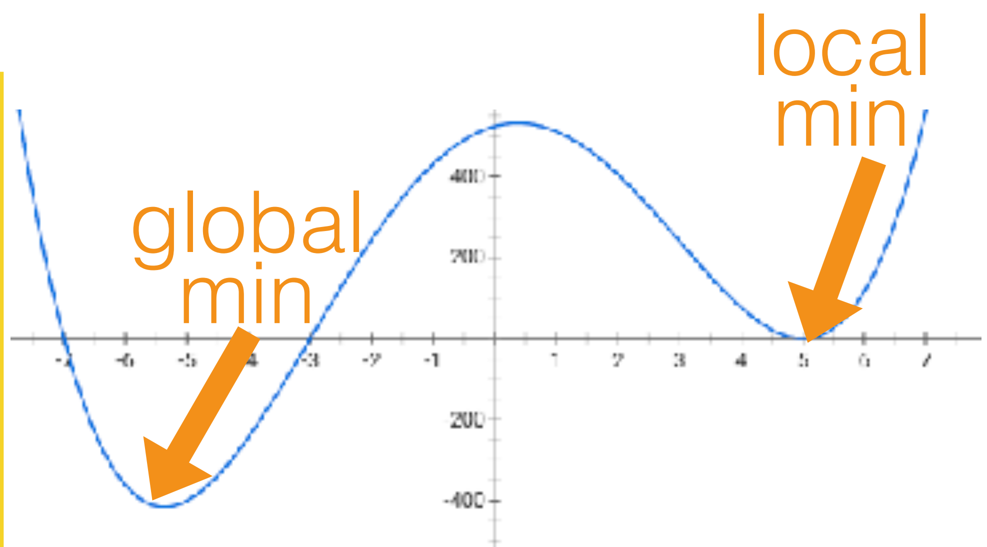
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

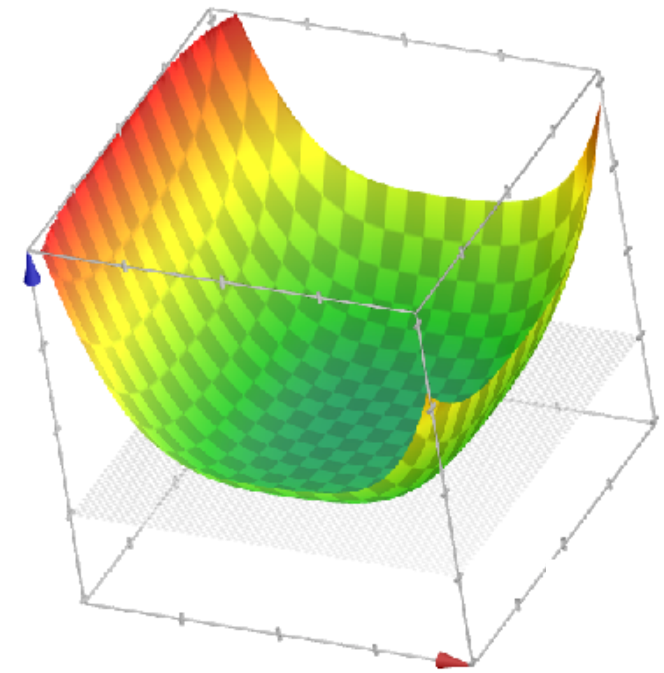


Gradient descent properties

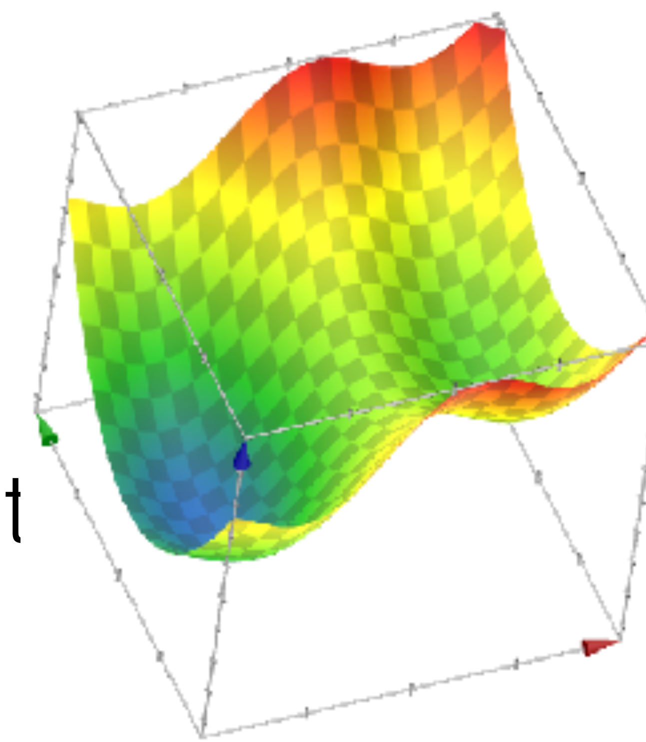
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





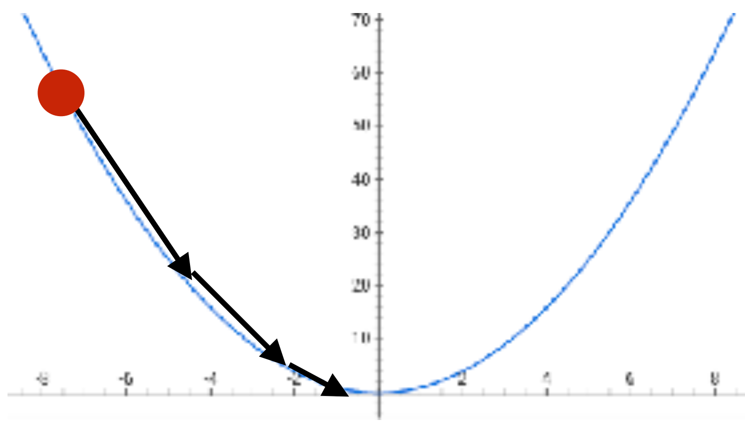


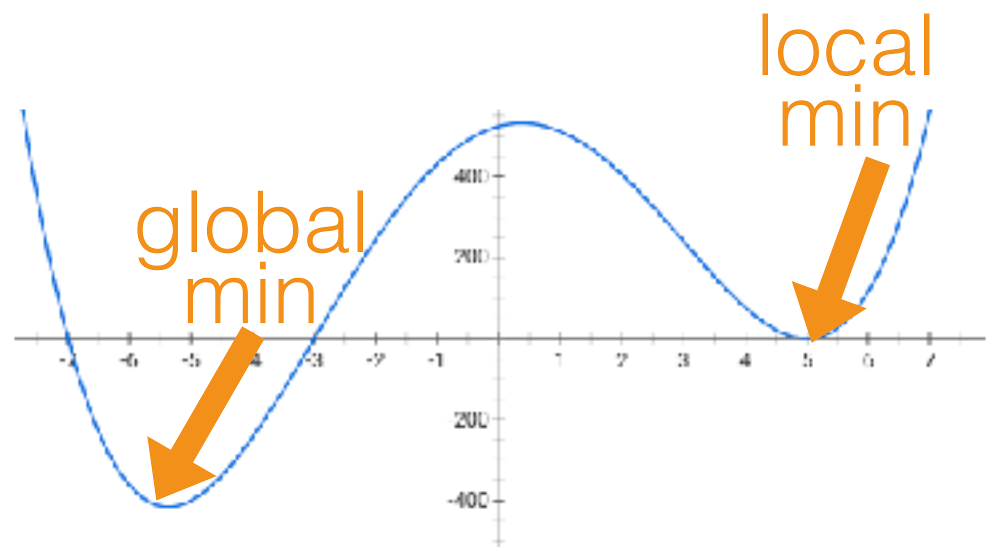
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently “smooth” and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

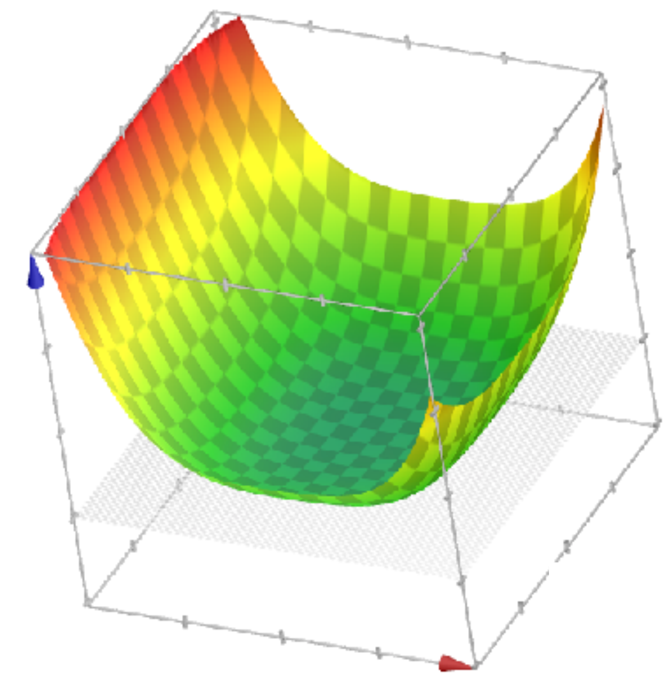


Gradient descent properties

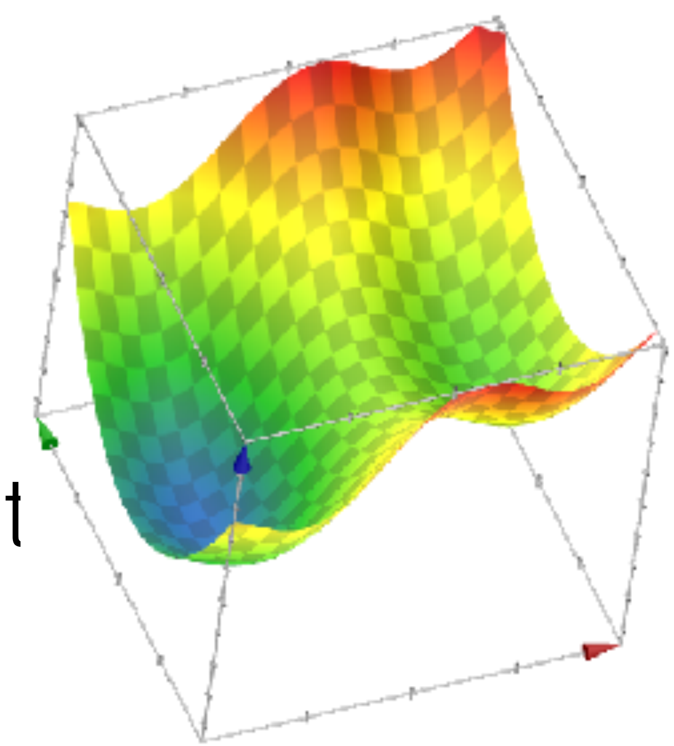
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





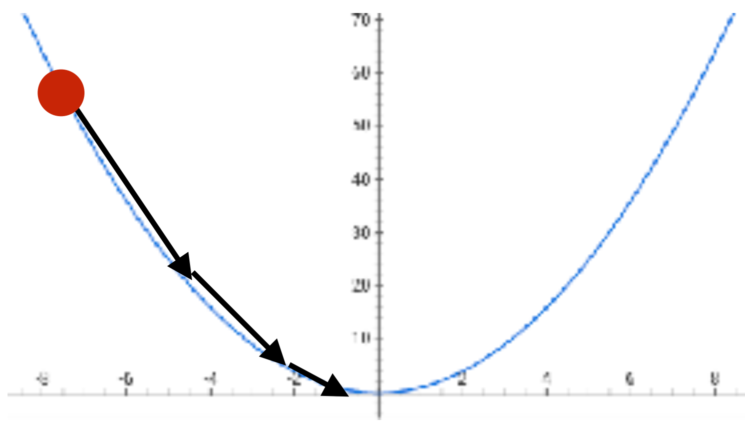


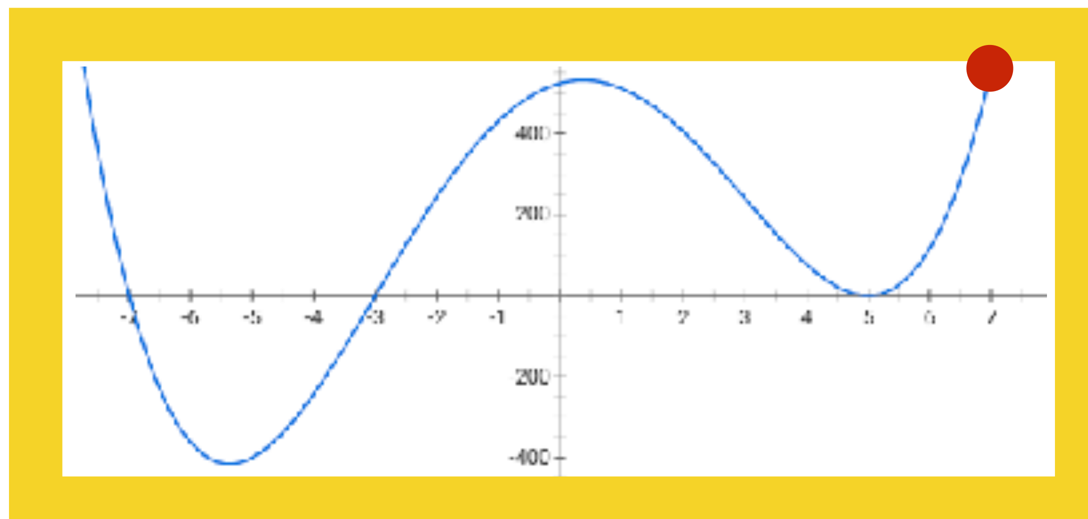
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

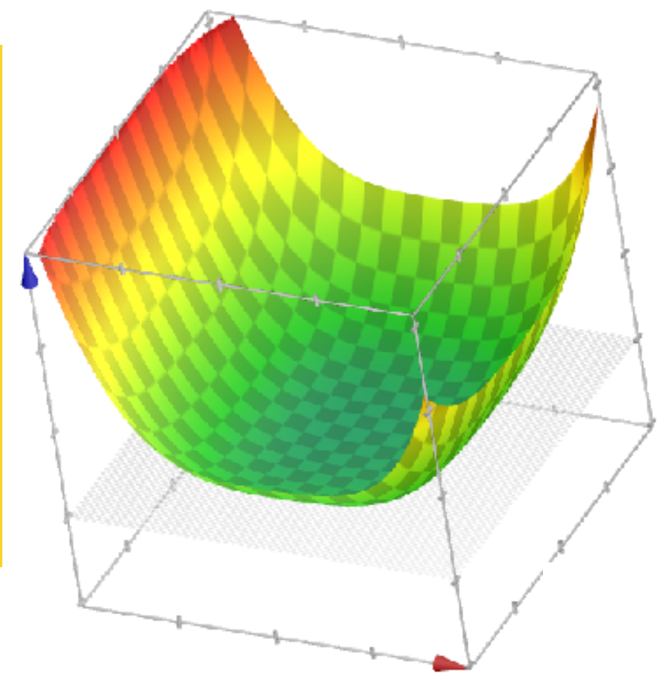


Gradient descent properties

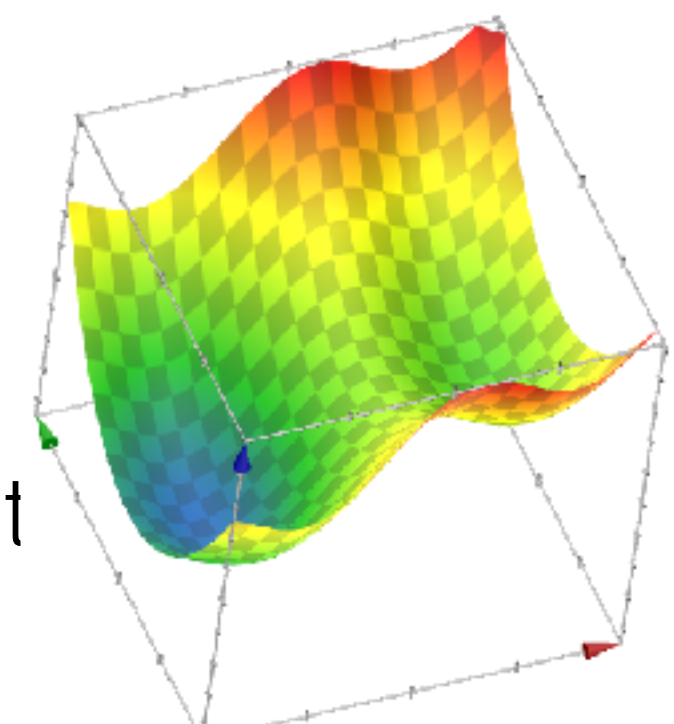
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





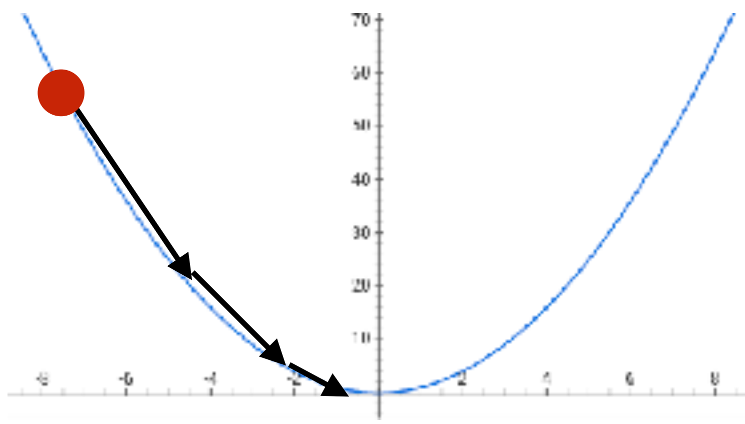


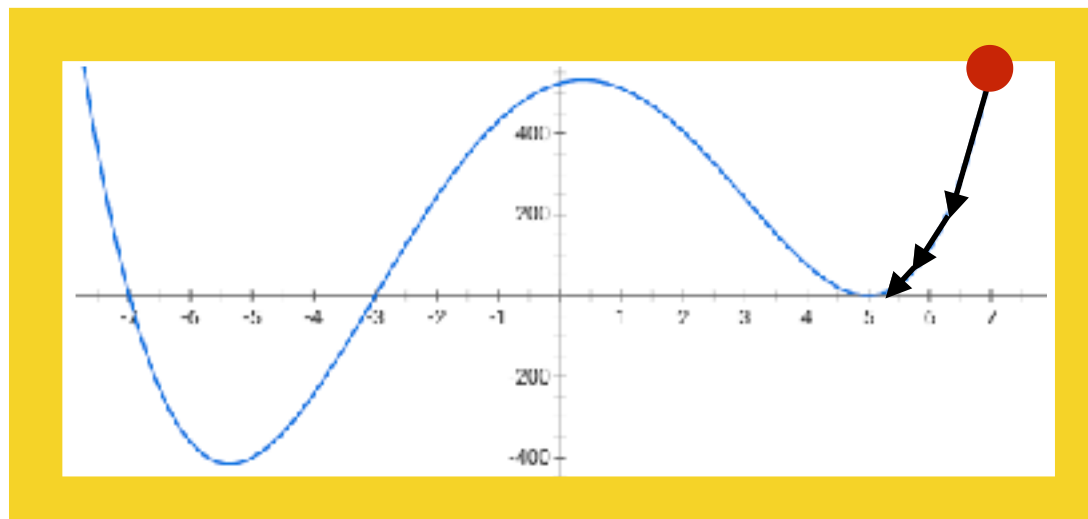
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

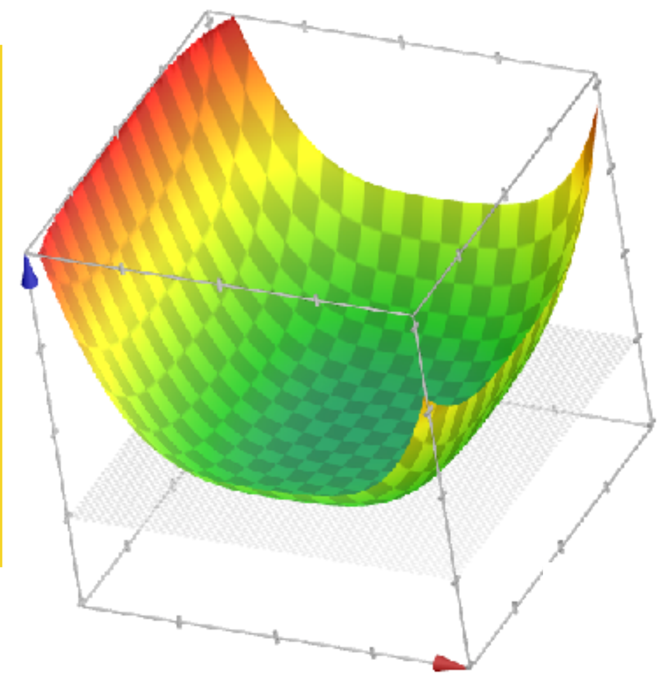


Gradient descent properties

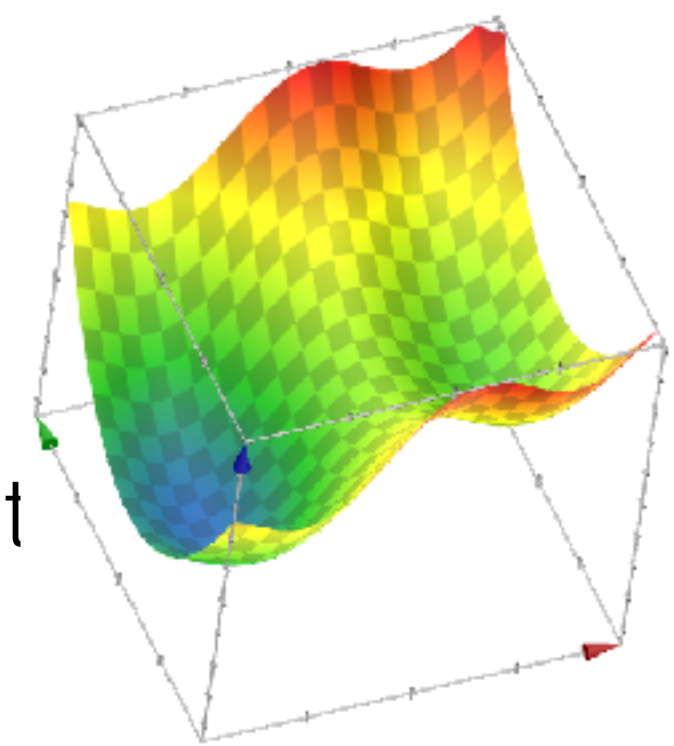
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





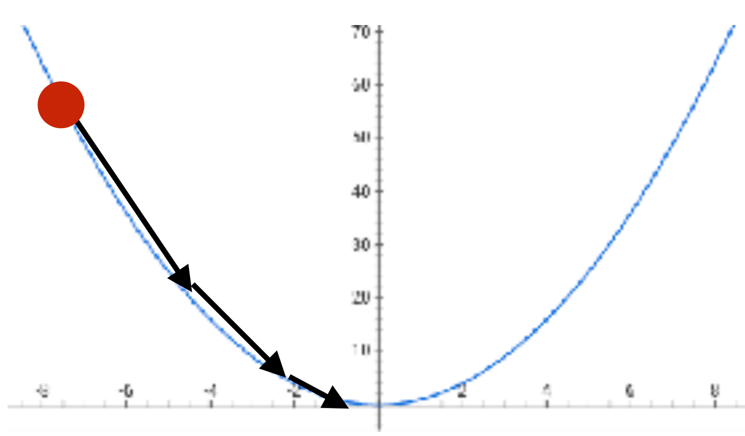


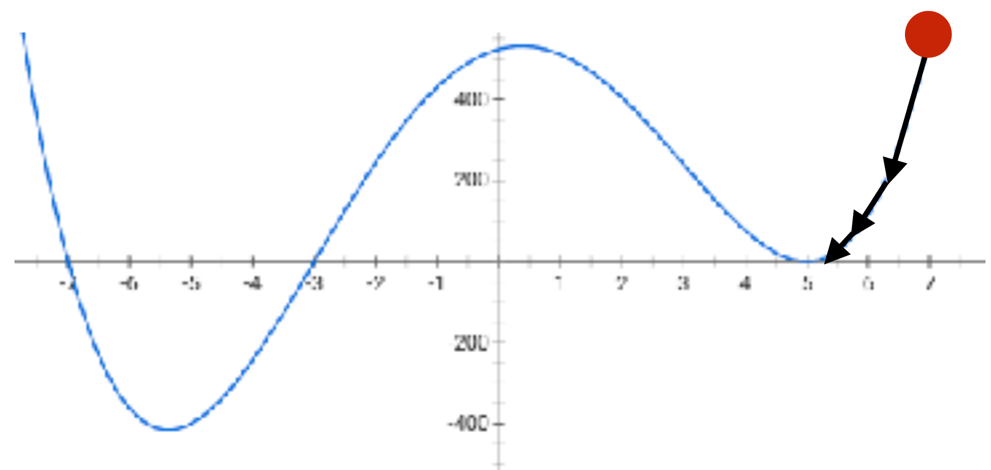
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently “smooth” and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

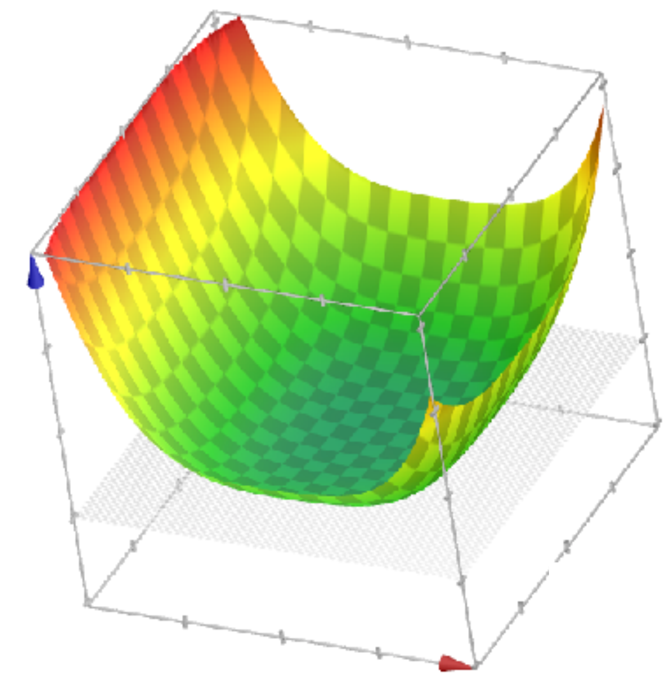


Gradient descent properties

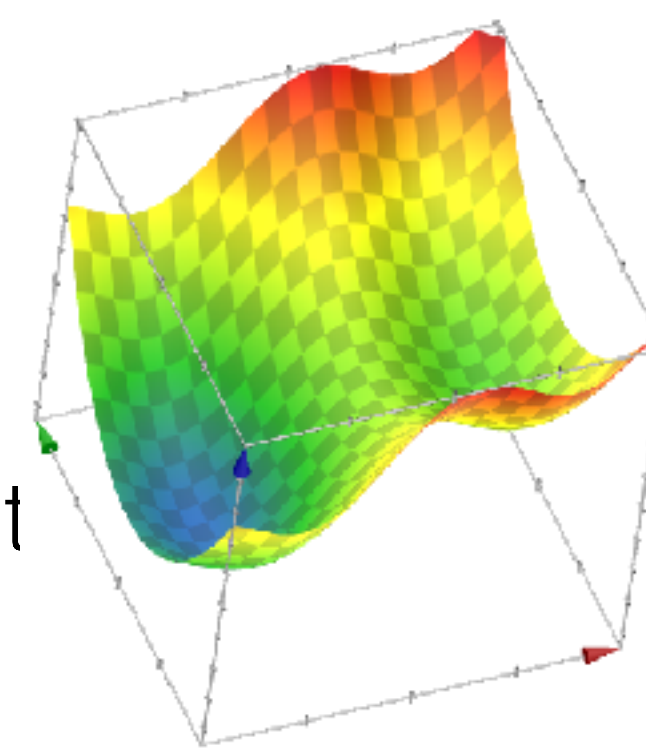
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





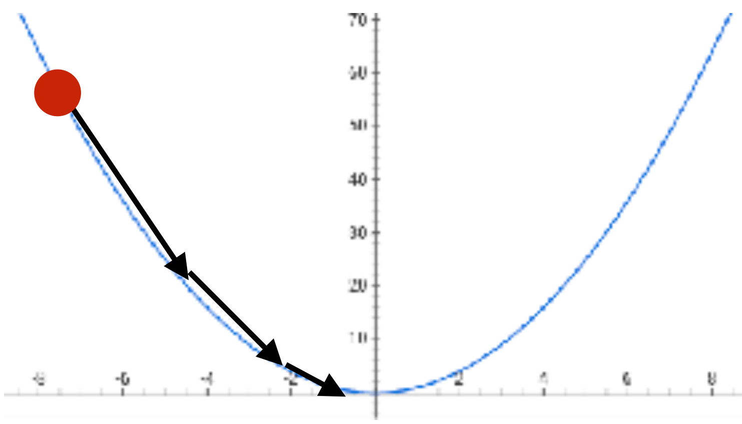


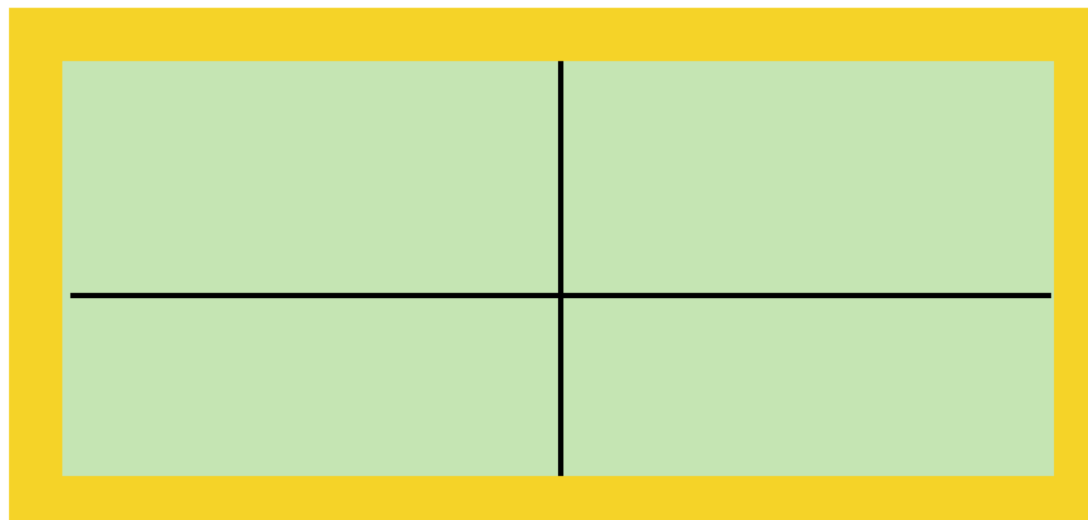
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

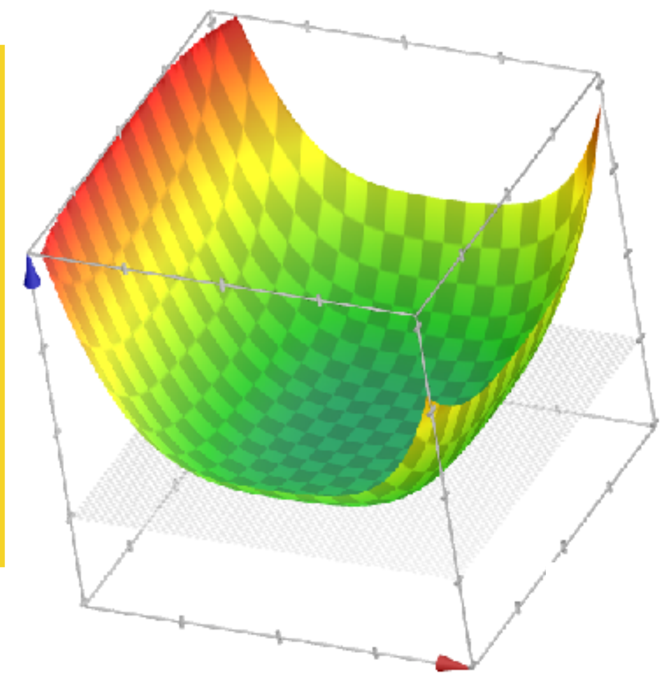


Gradient descent properties

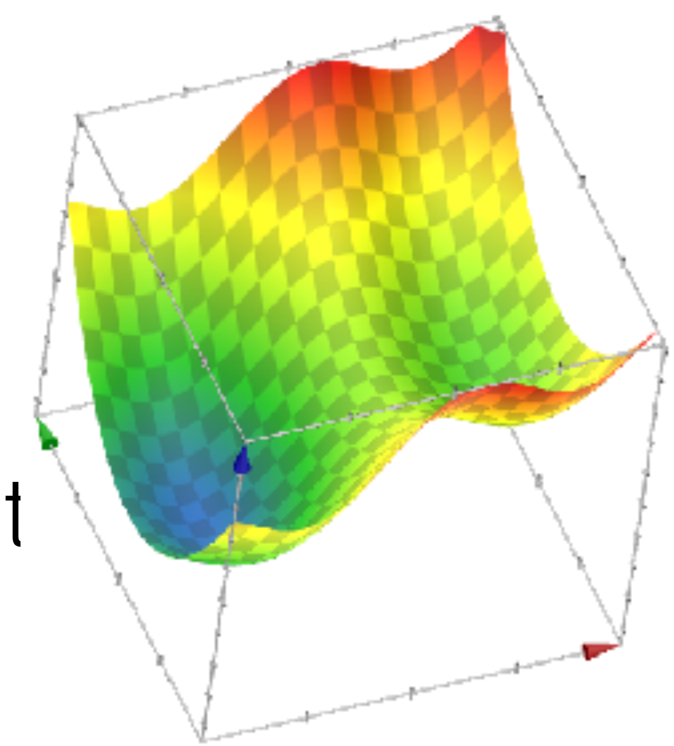
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





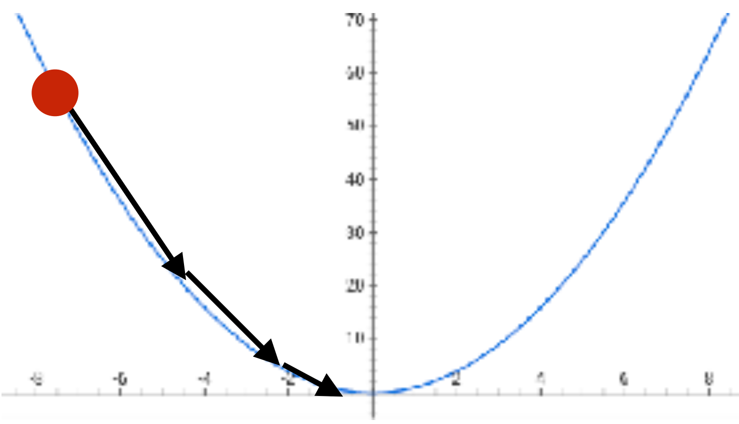


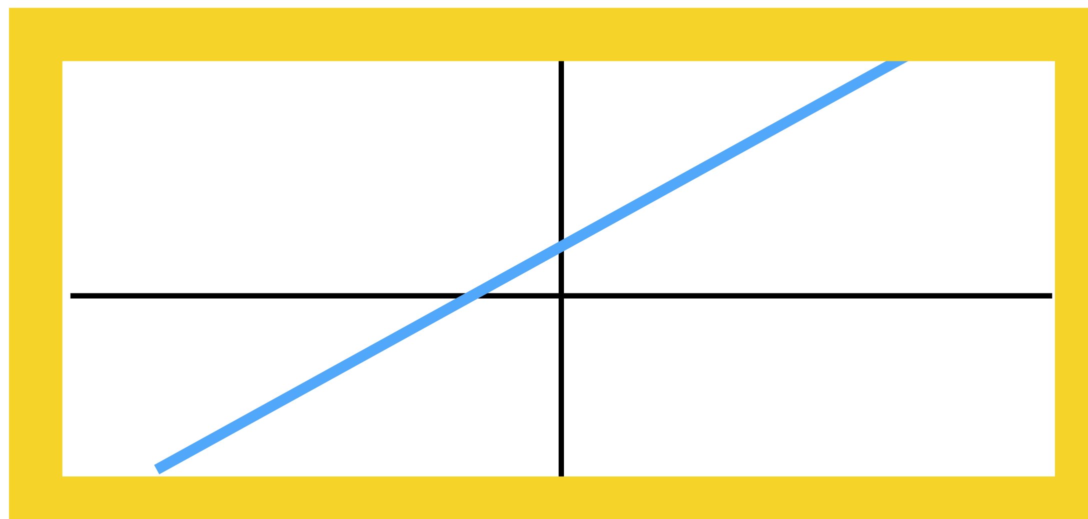
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

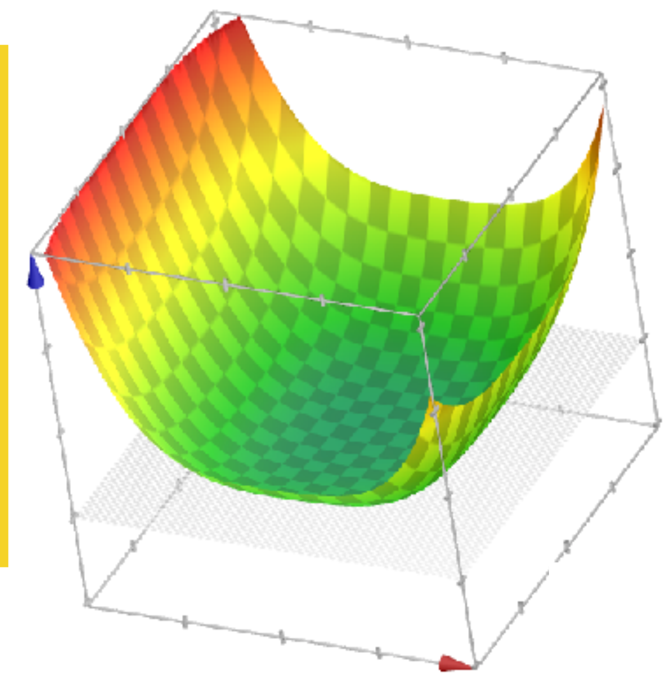


Gradient descent properties

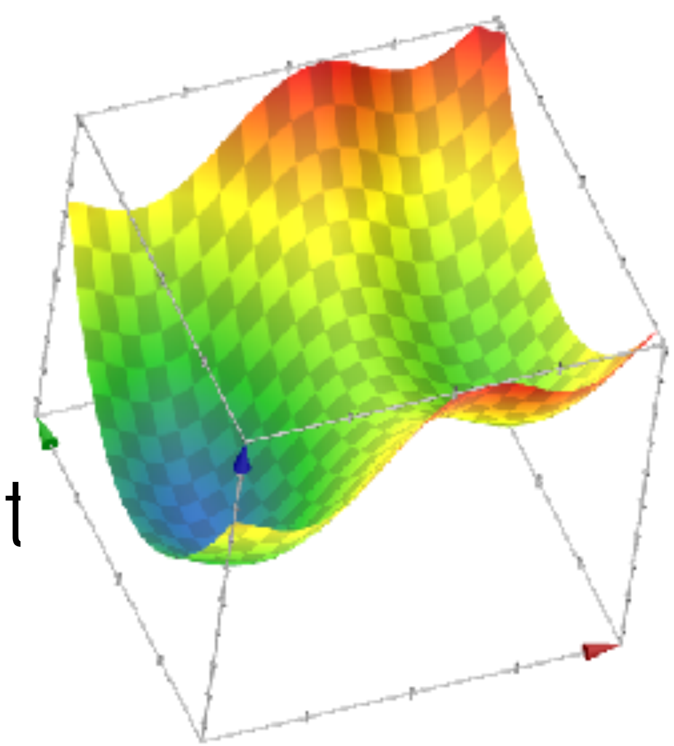
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





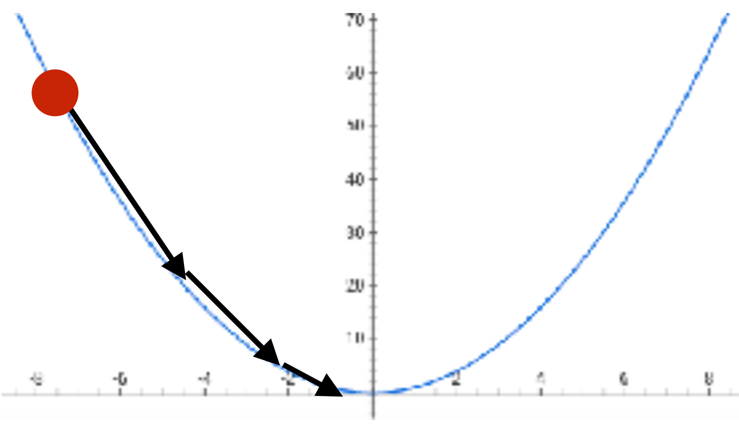


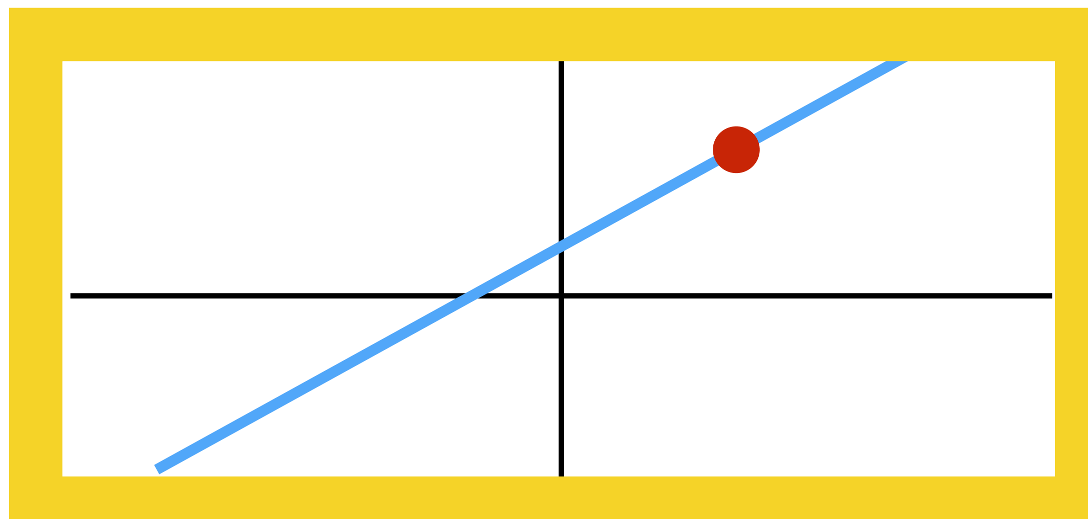
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

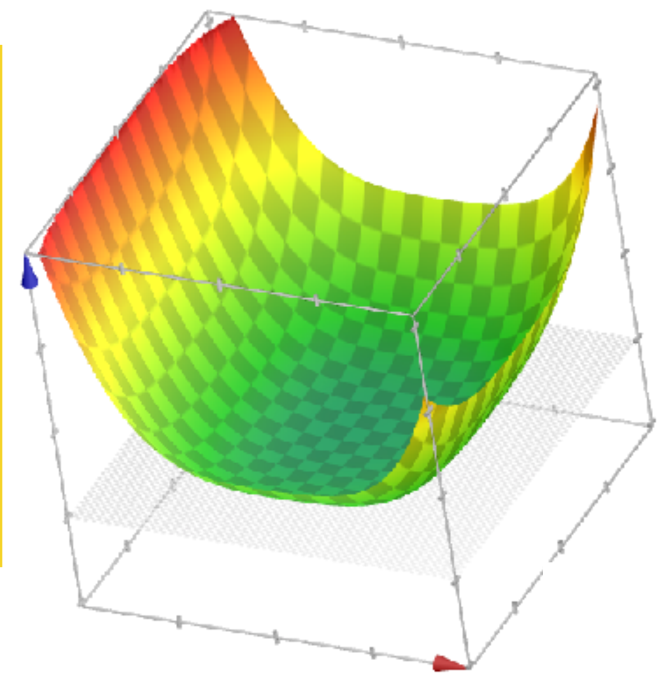


Gradient descent properties

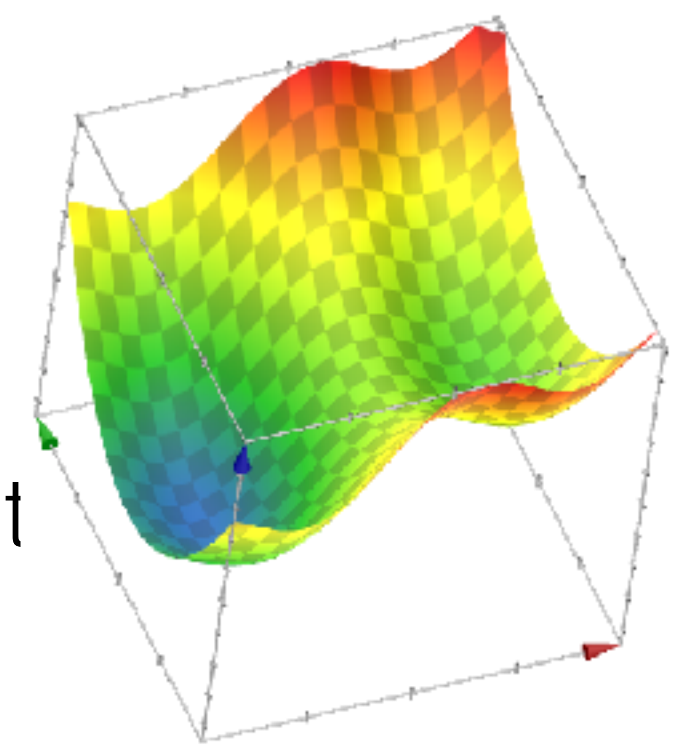
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





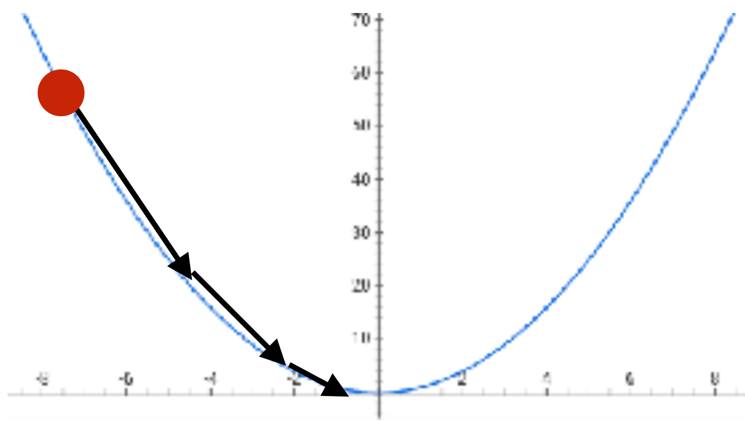


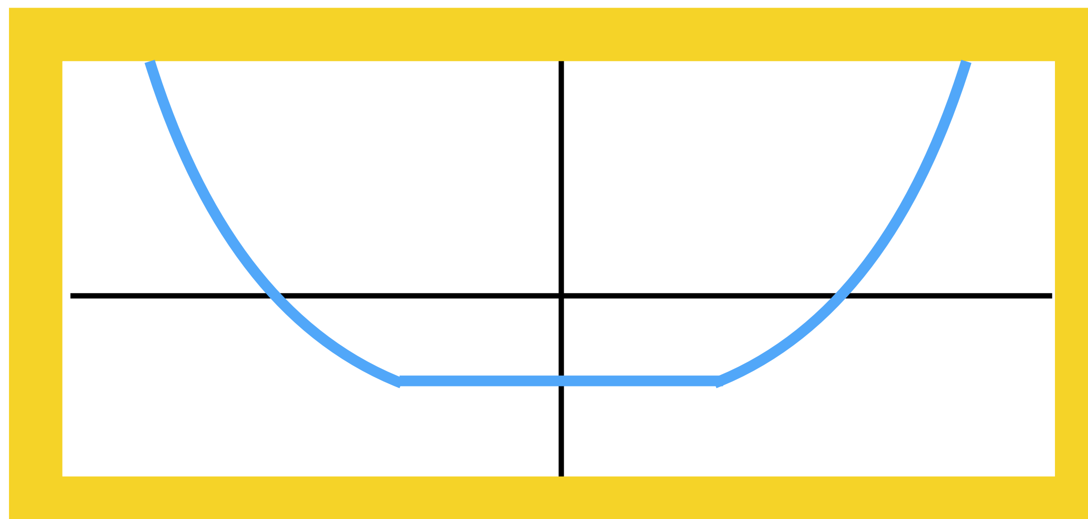
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

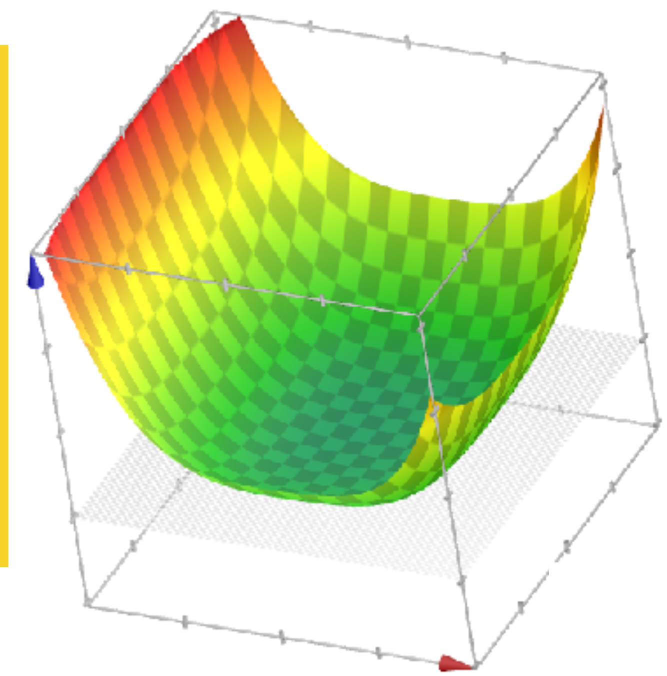


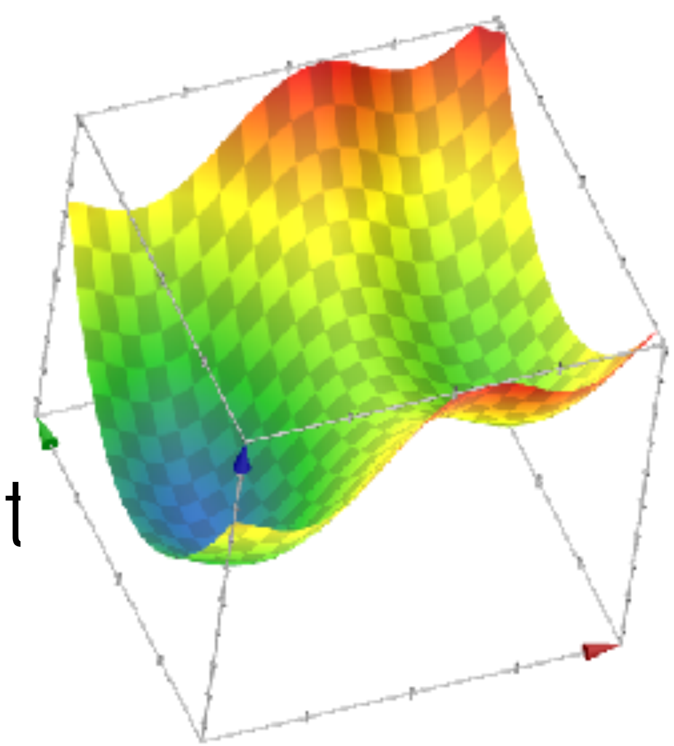
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





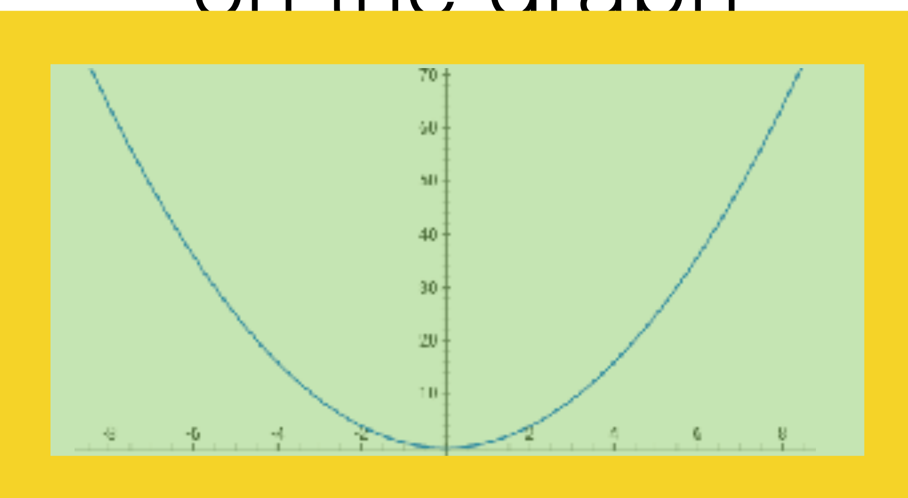


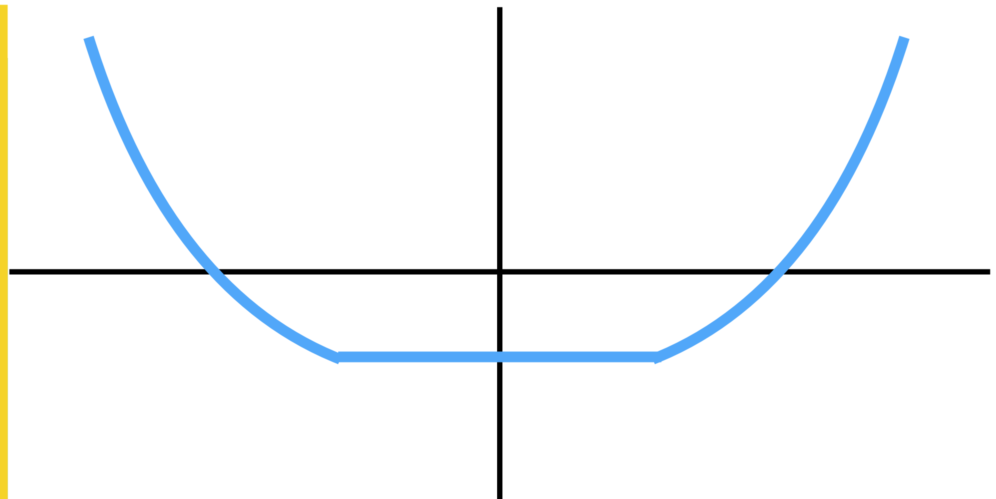


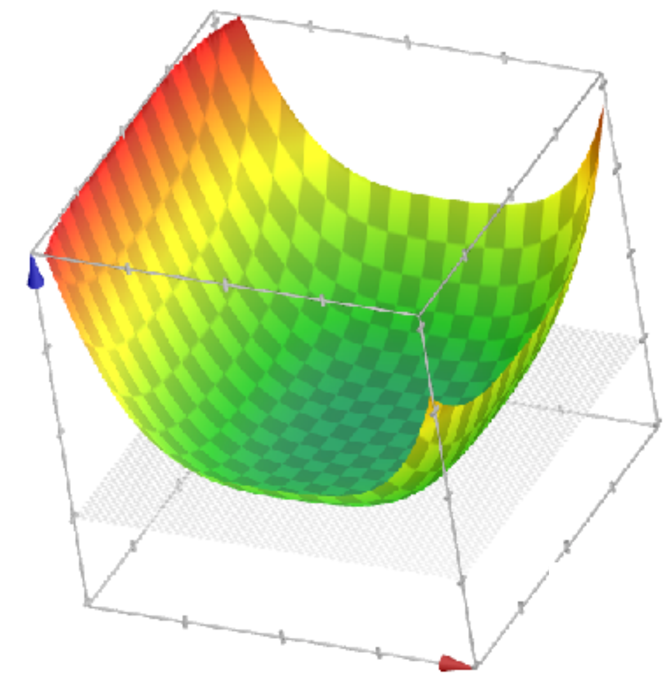
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently “smooth” and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

Gradient descent properties

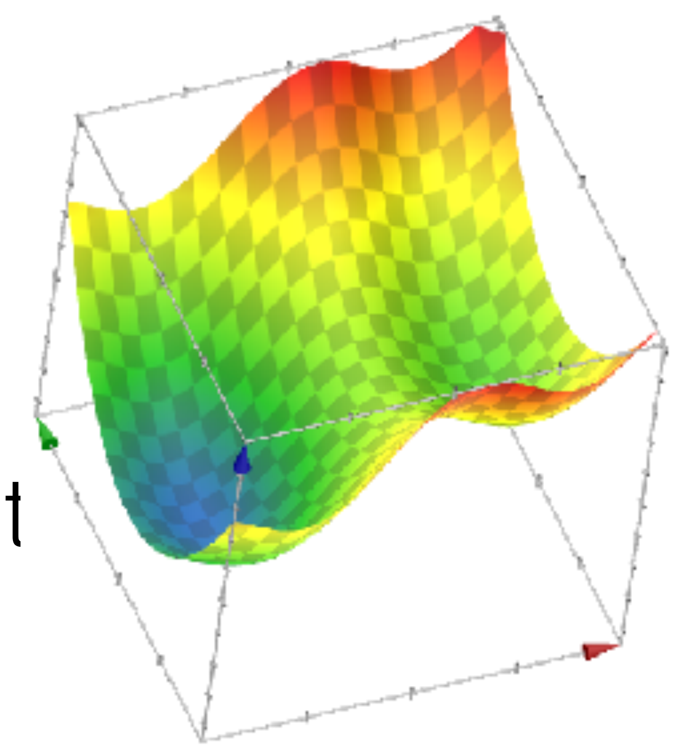
- A function f on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of f lies above or on the graph





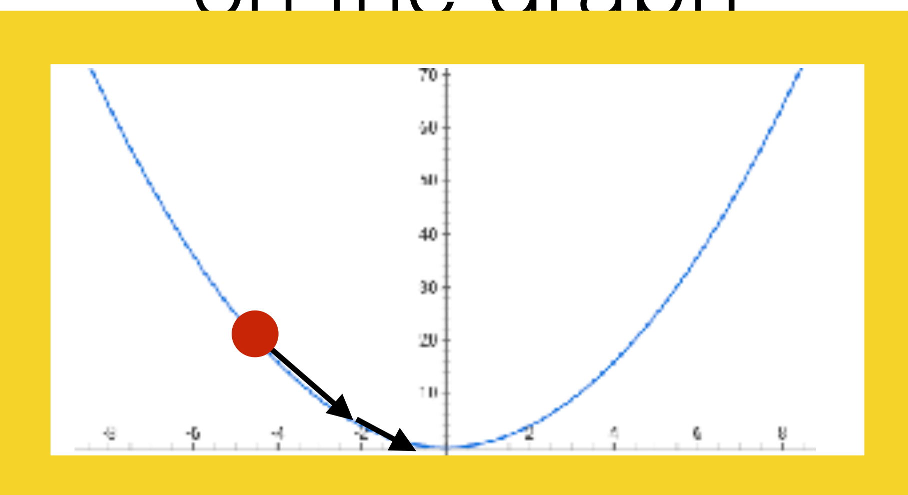


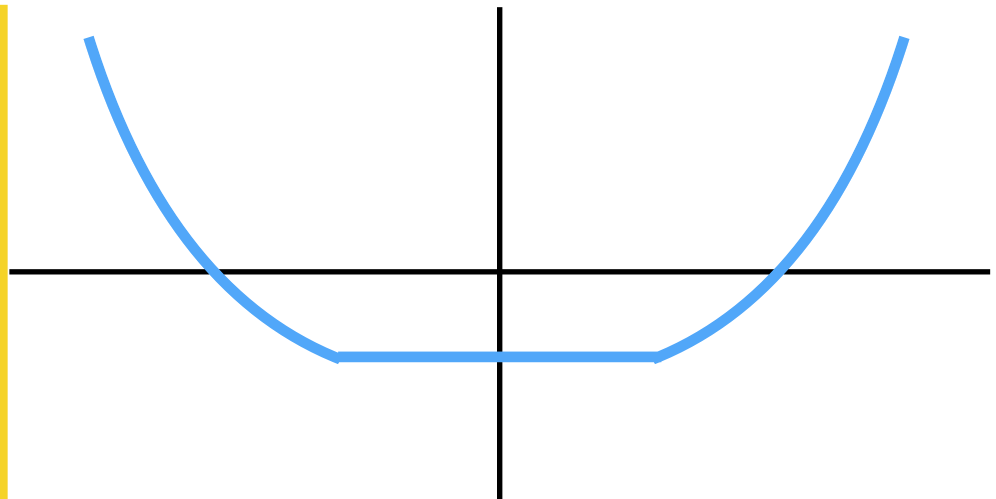
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - f is sufficiently "smooth" and convex
    - f has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

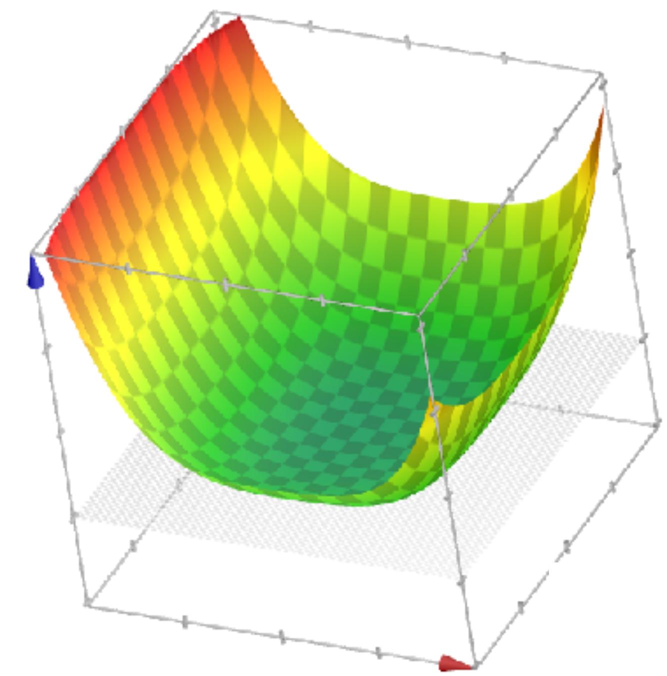


Gradient descent properties

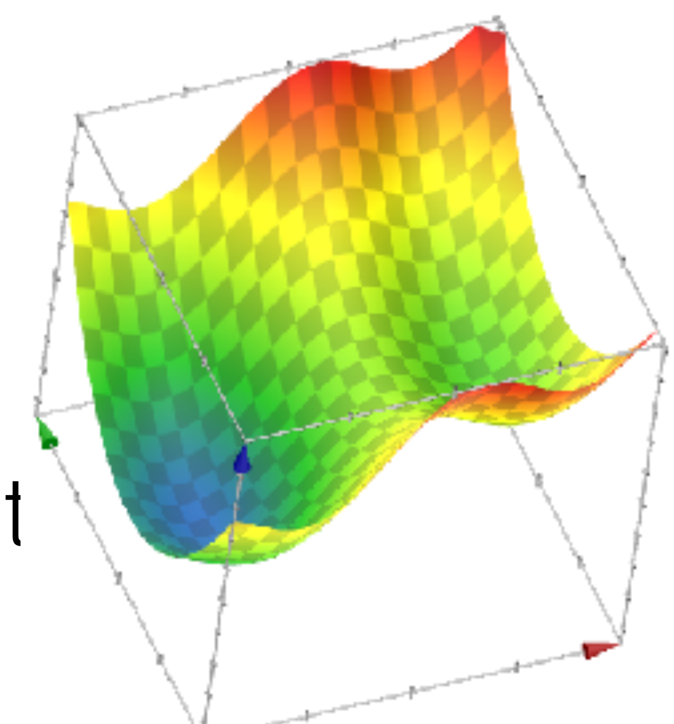
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





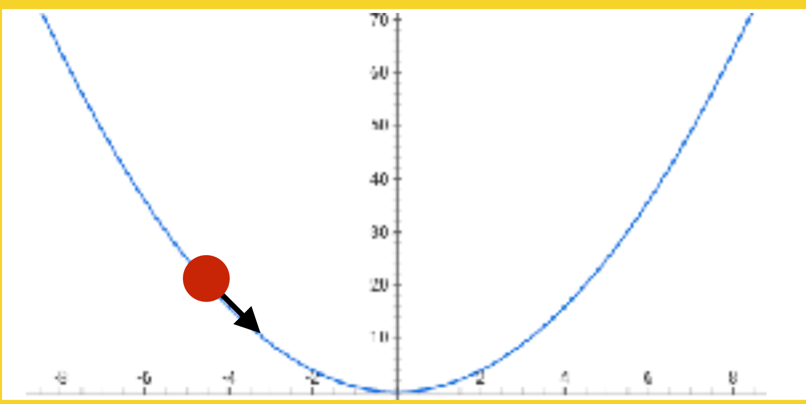


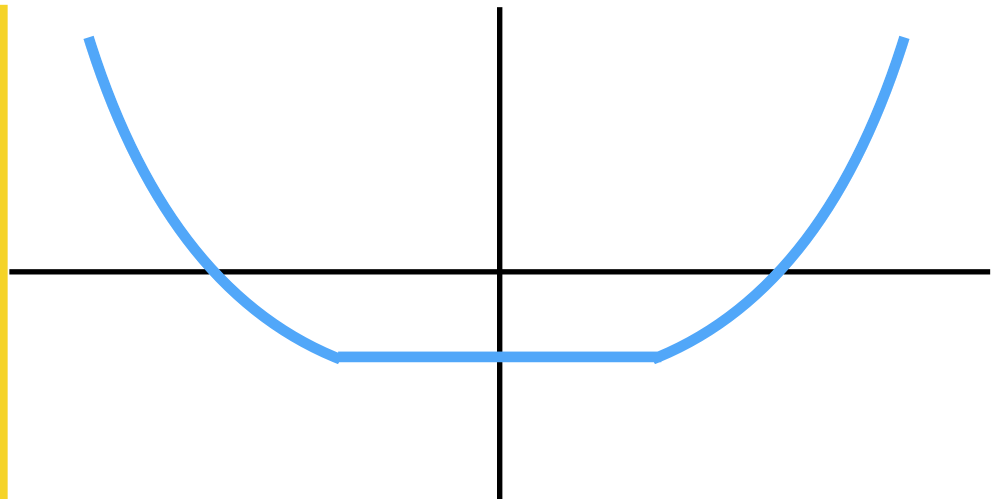
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

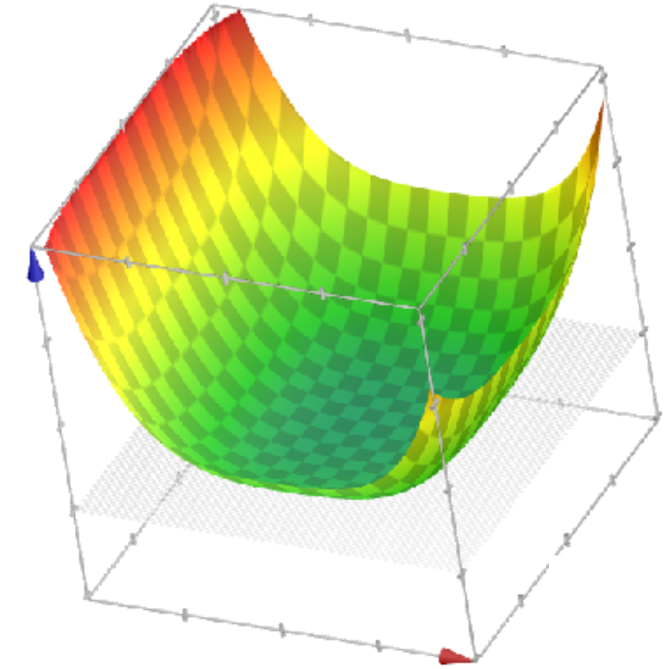


Gradient descent properties

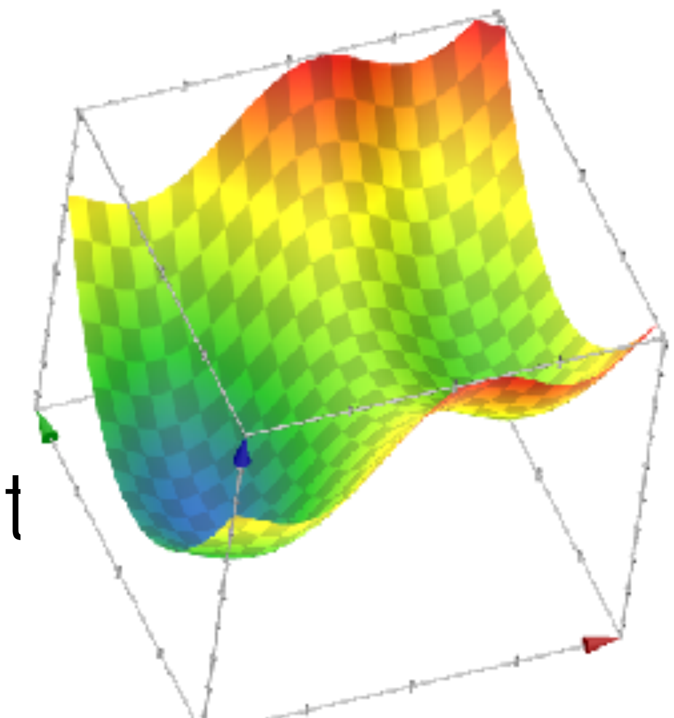
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





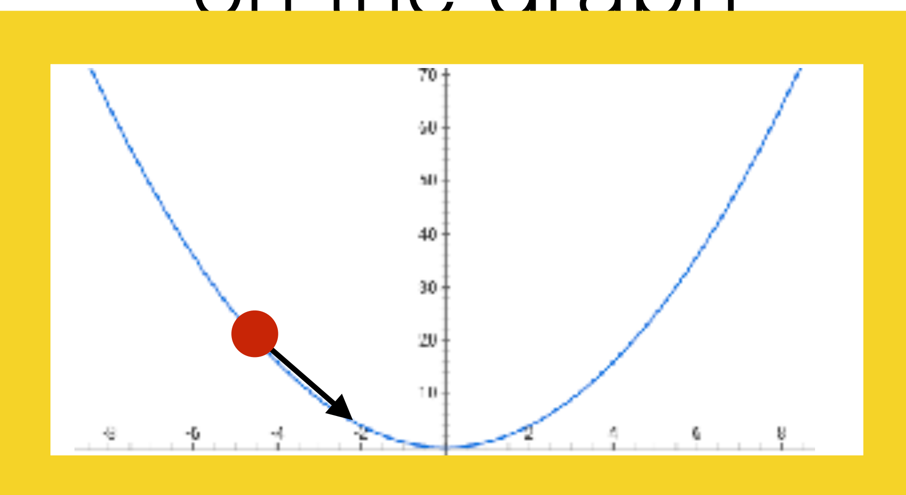


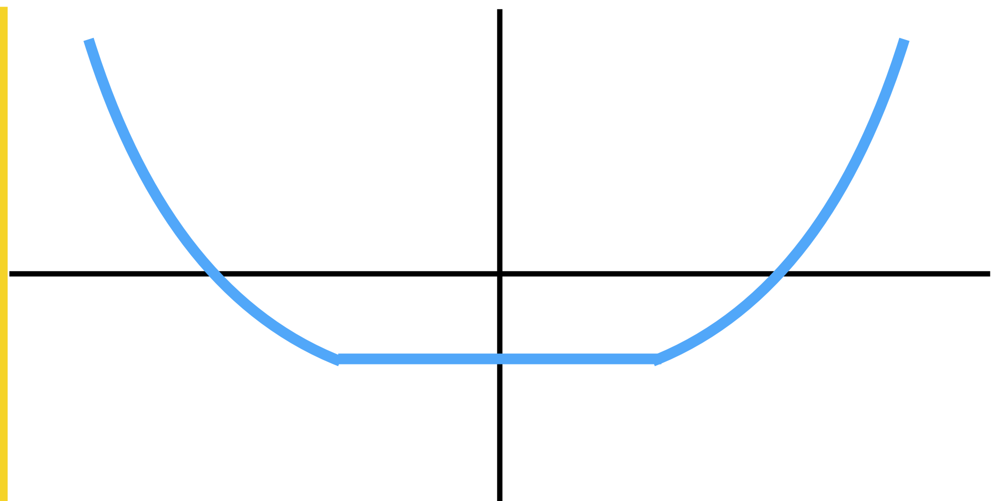
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

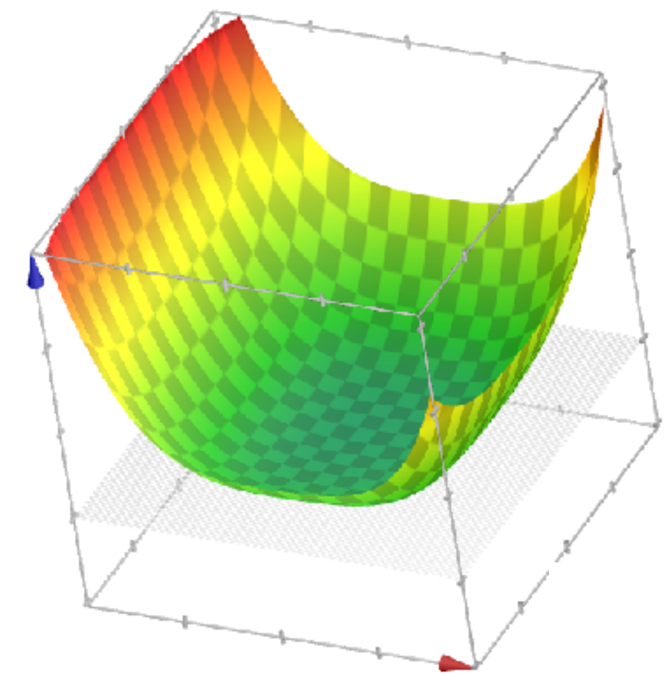


Gradient descent properties

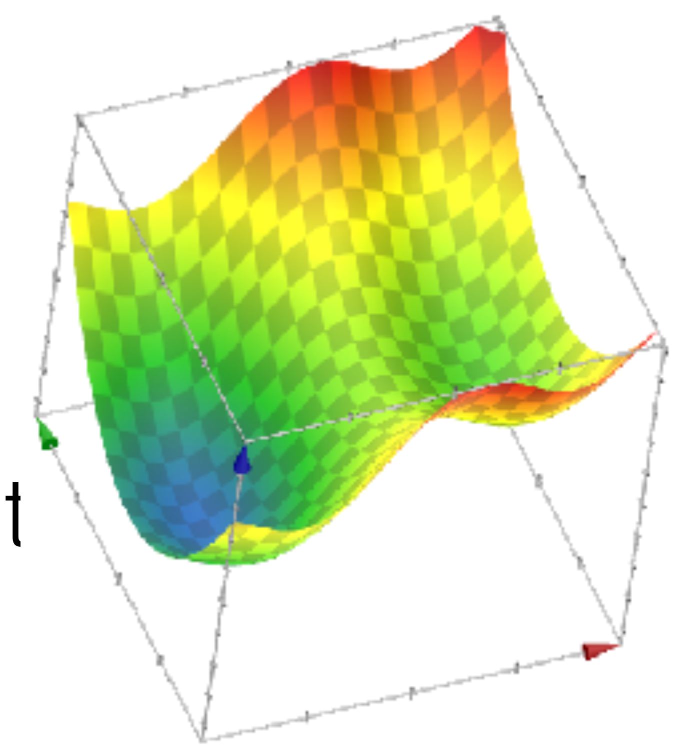
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





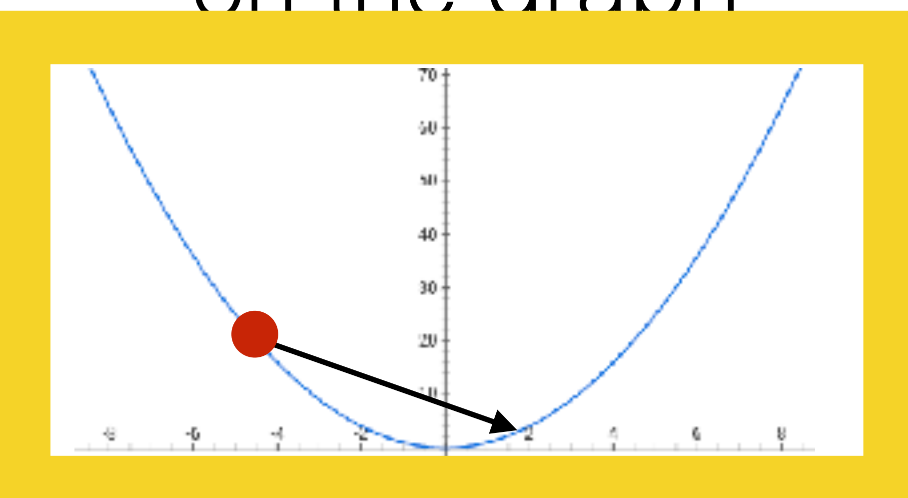


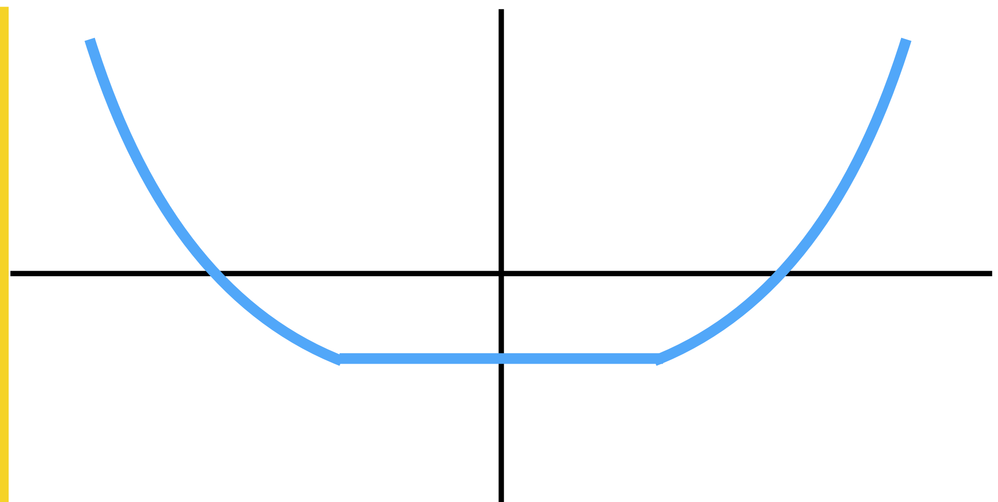
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

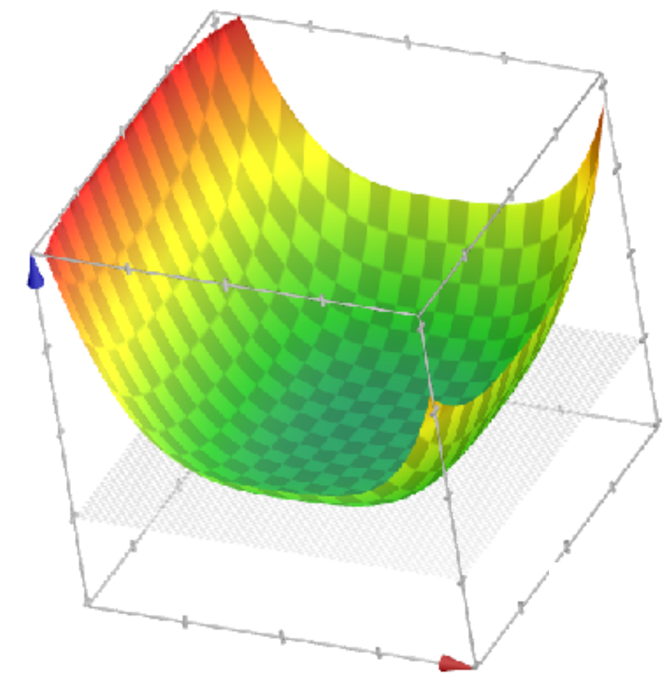


Gradient descent properties

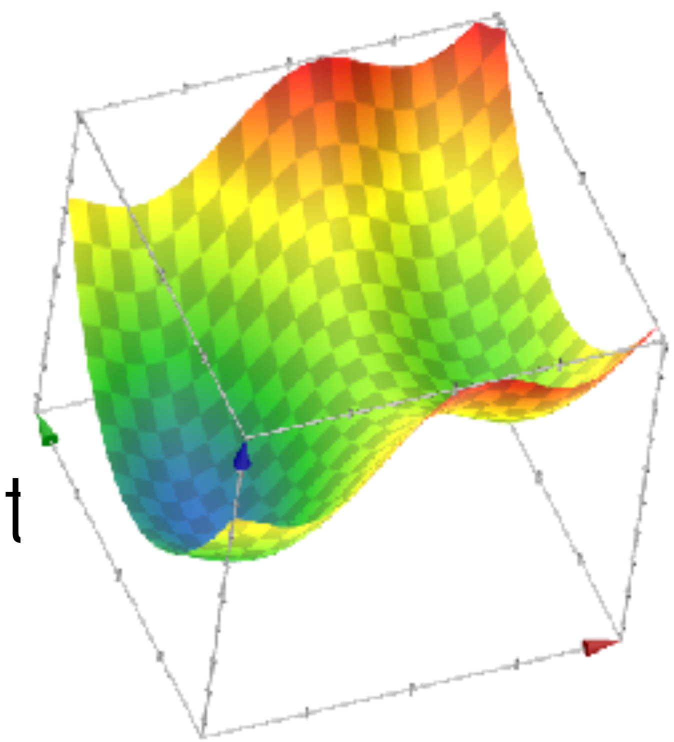
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





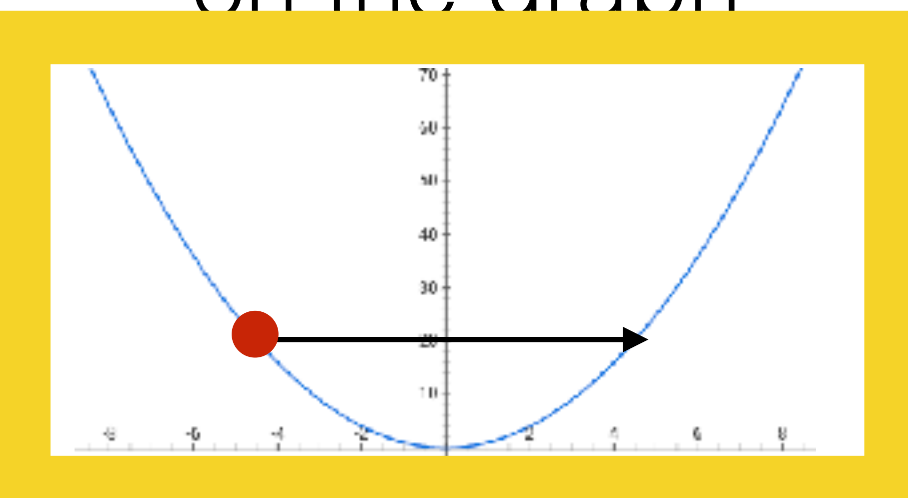


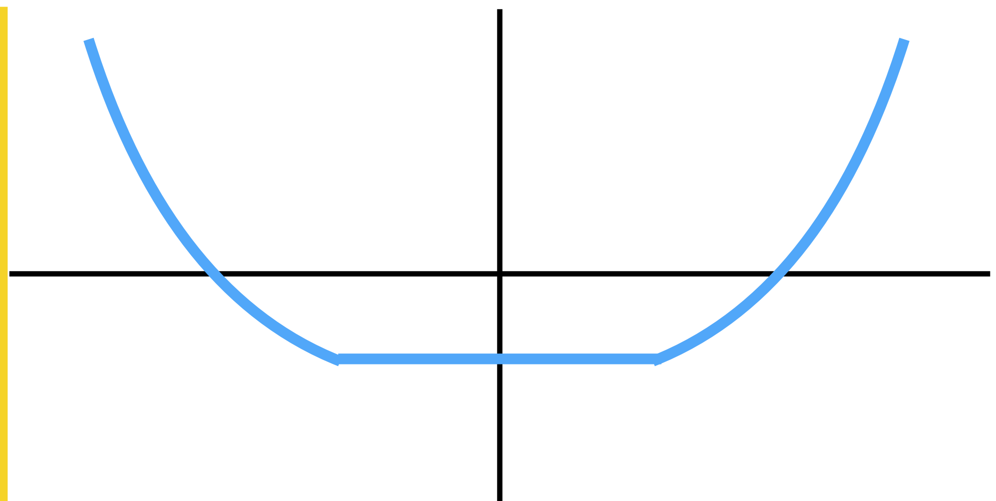
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

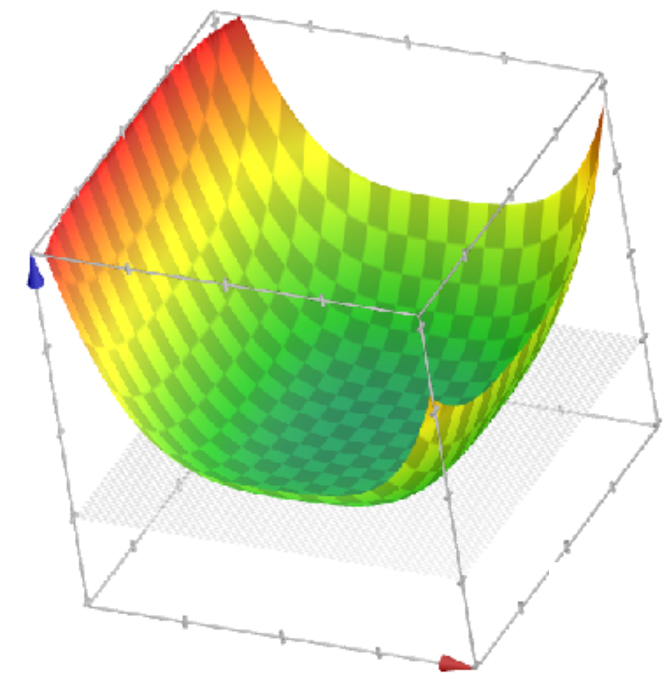


Gradient descent properties

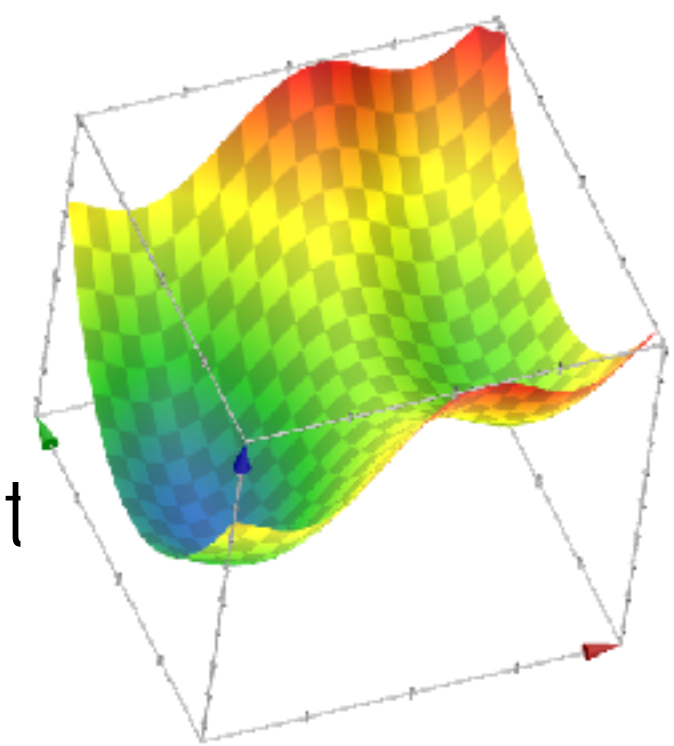
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





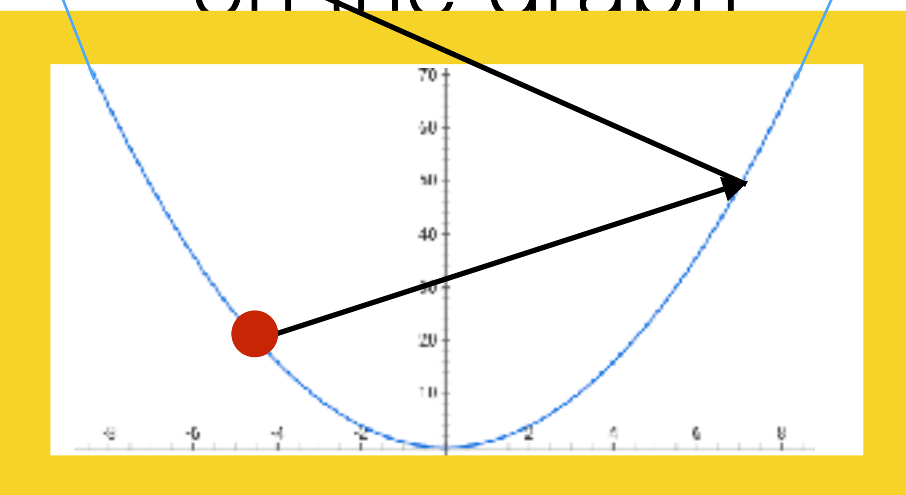


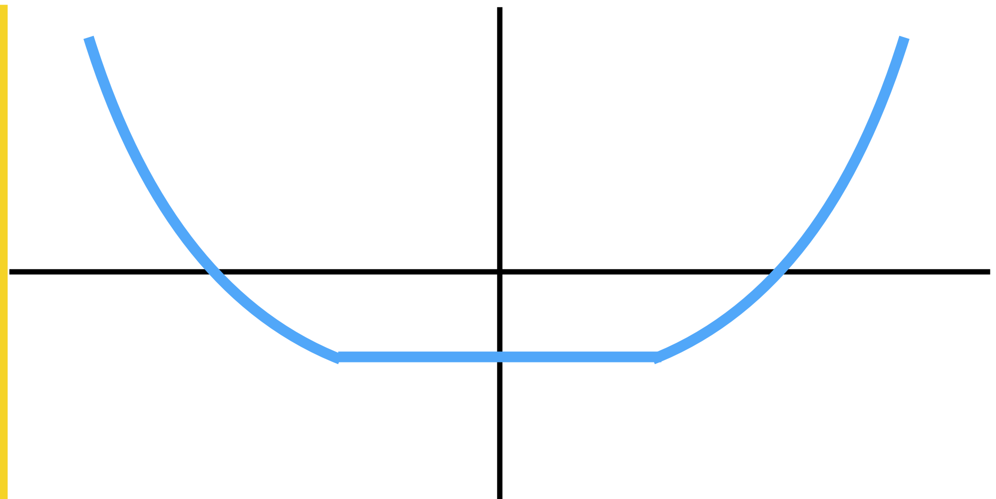
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently “smooth” and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

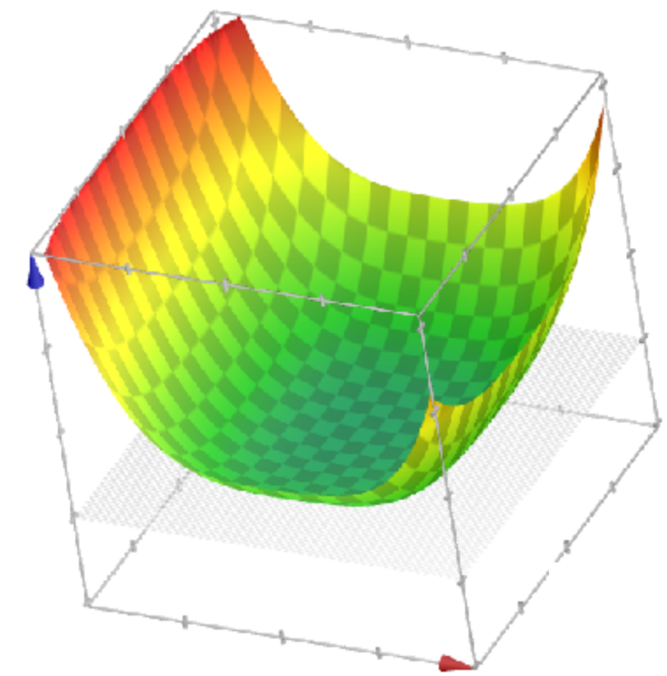


Gradient descent properties

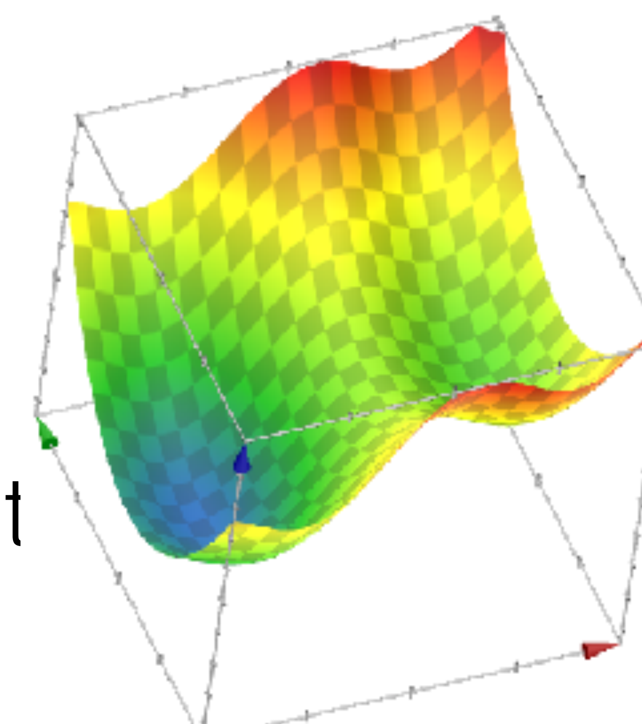
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





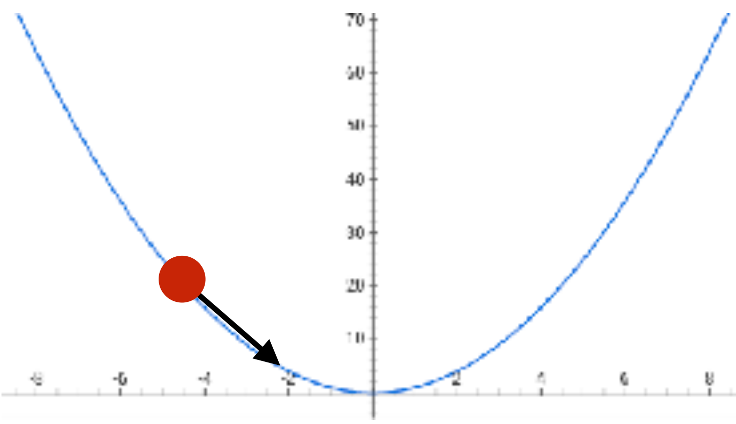


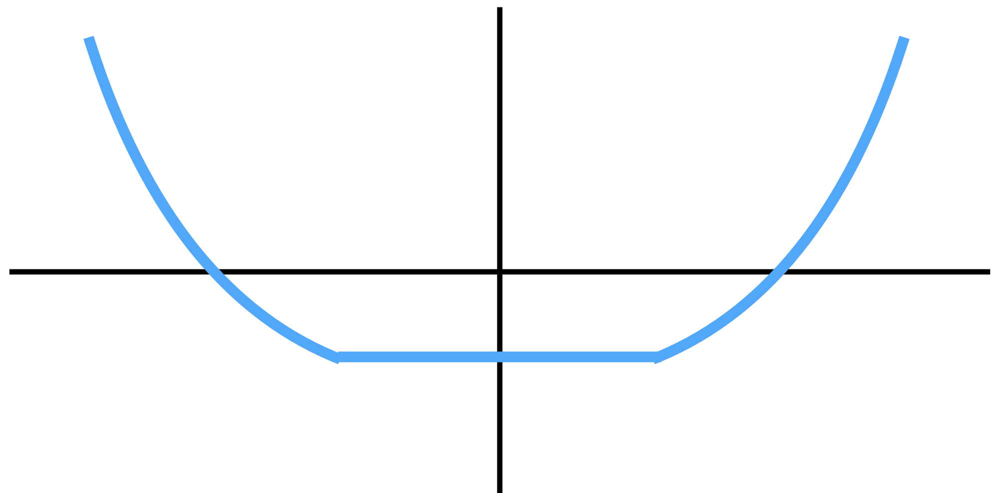
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

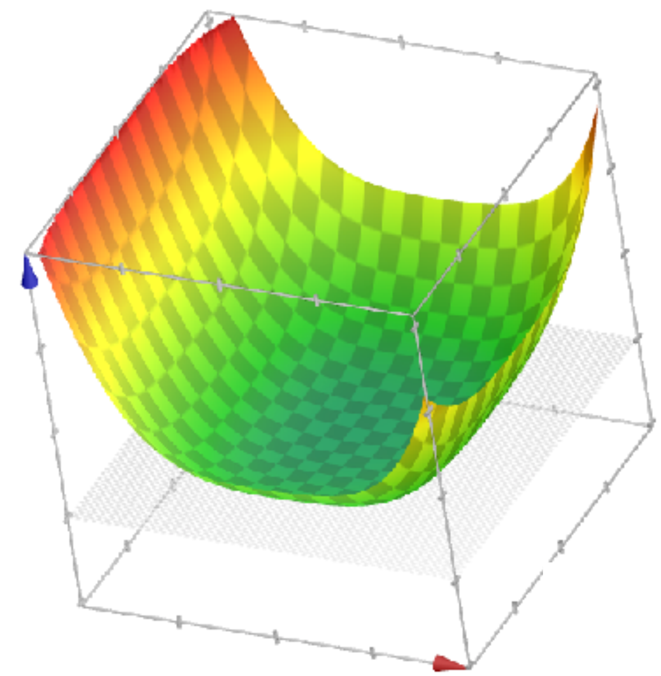


Gradient descent properties

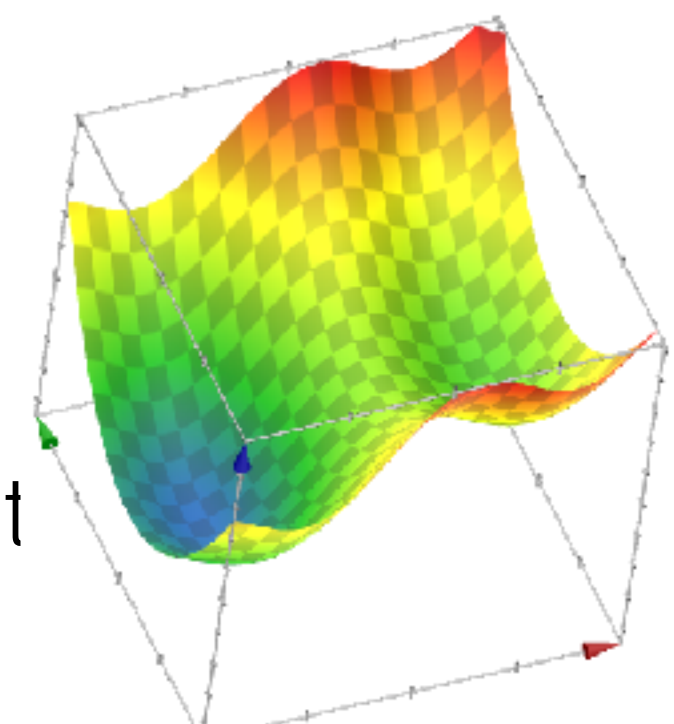
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





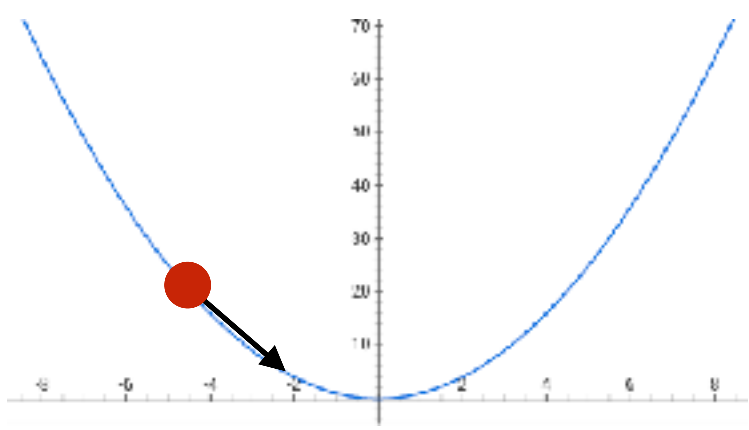


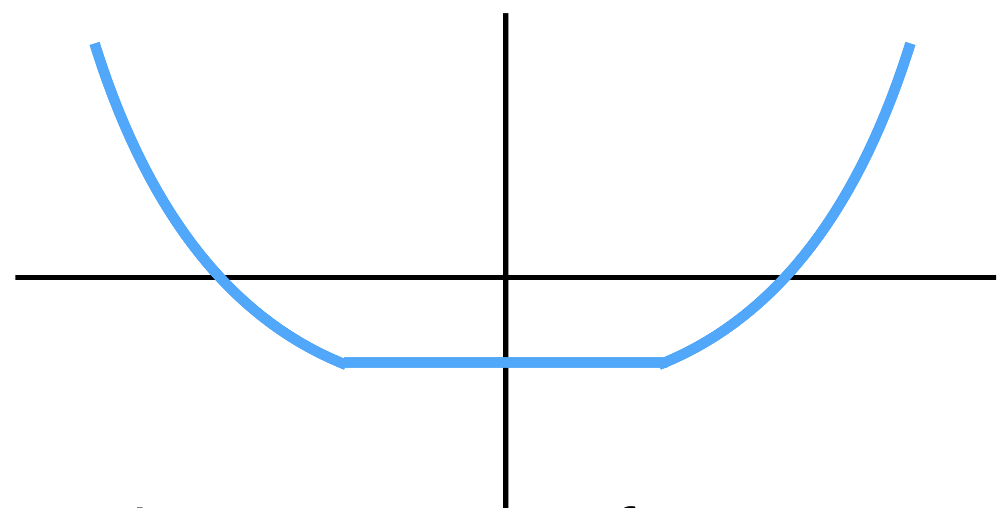
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

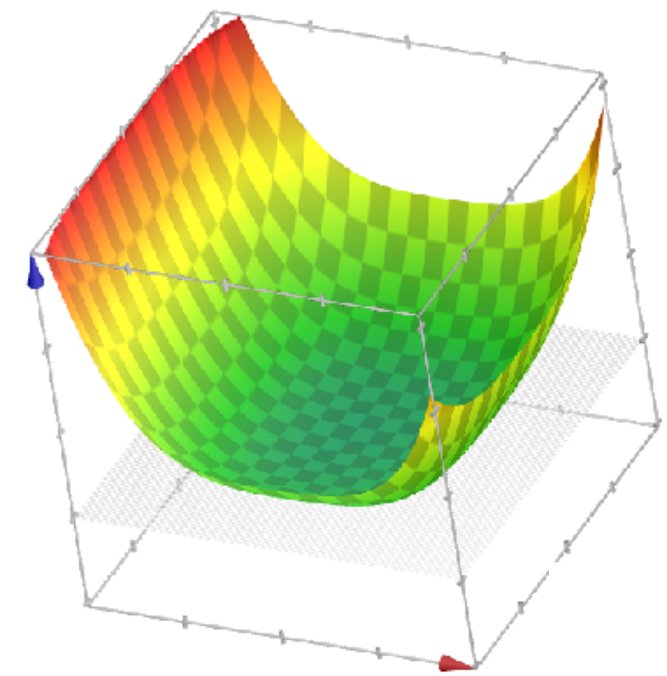


Gradient descent properties

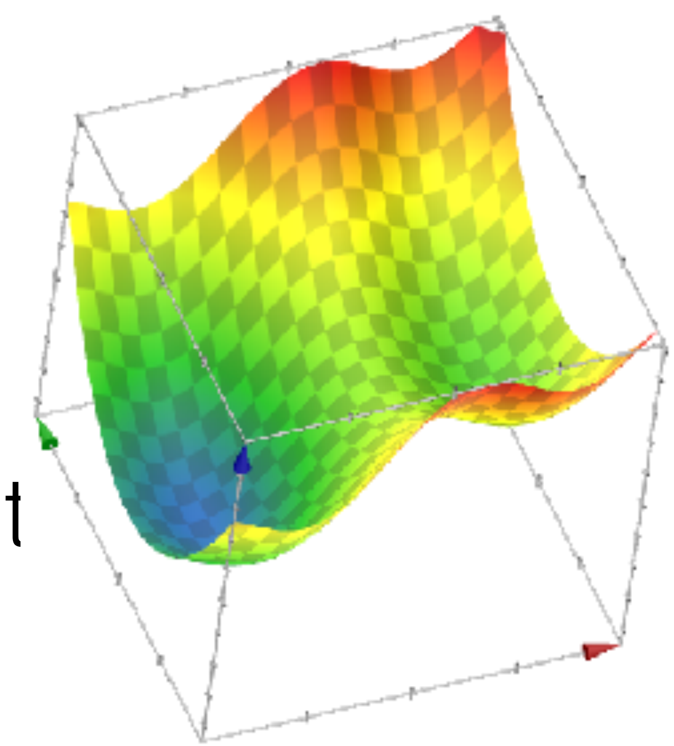
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





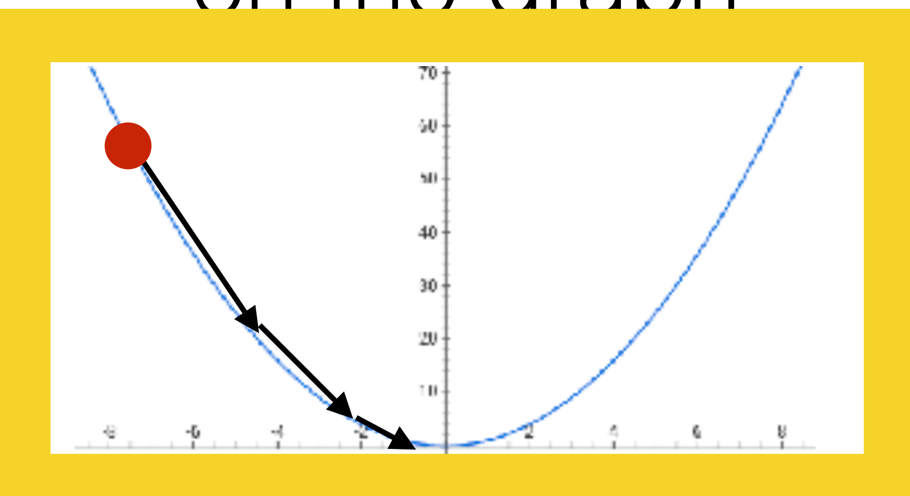


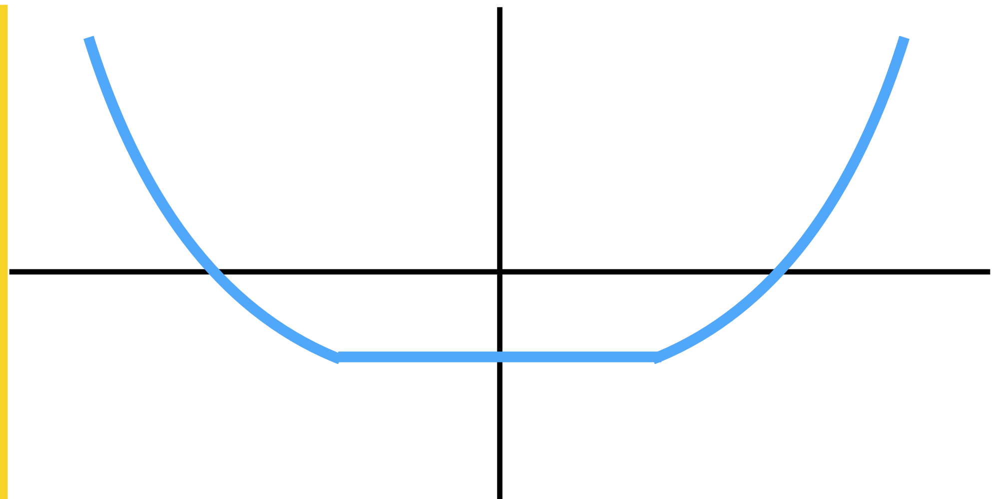
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ

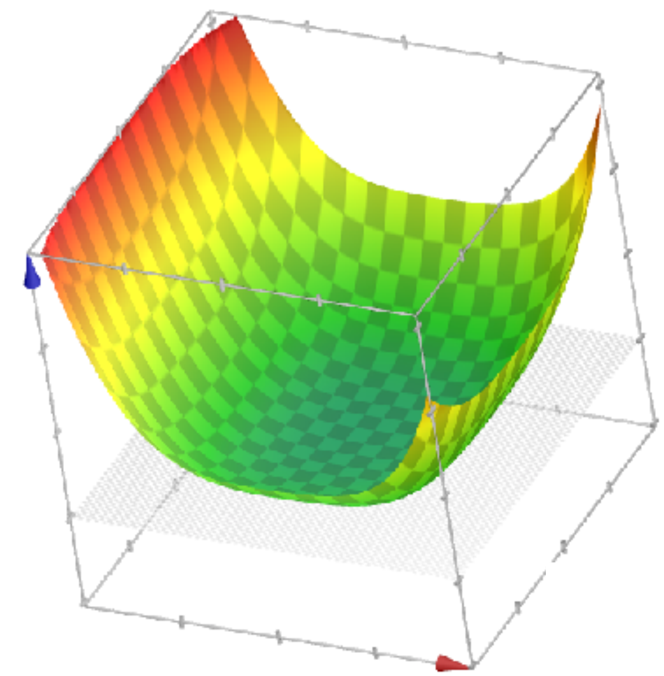


Gradient descent properties

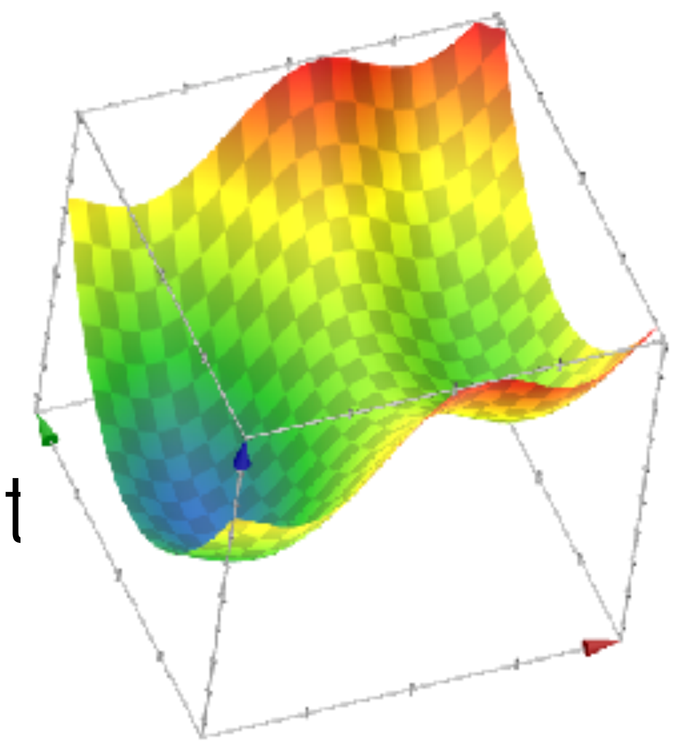
- A function f on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of f lies above or on the graph





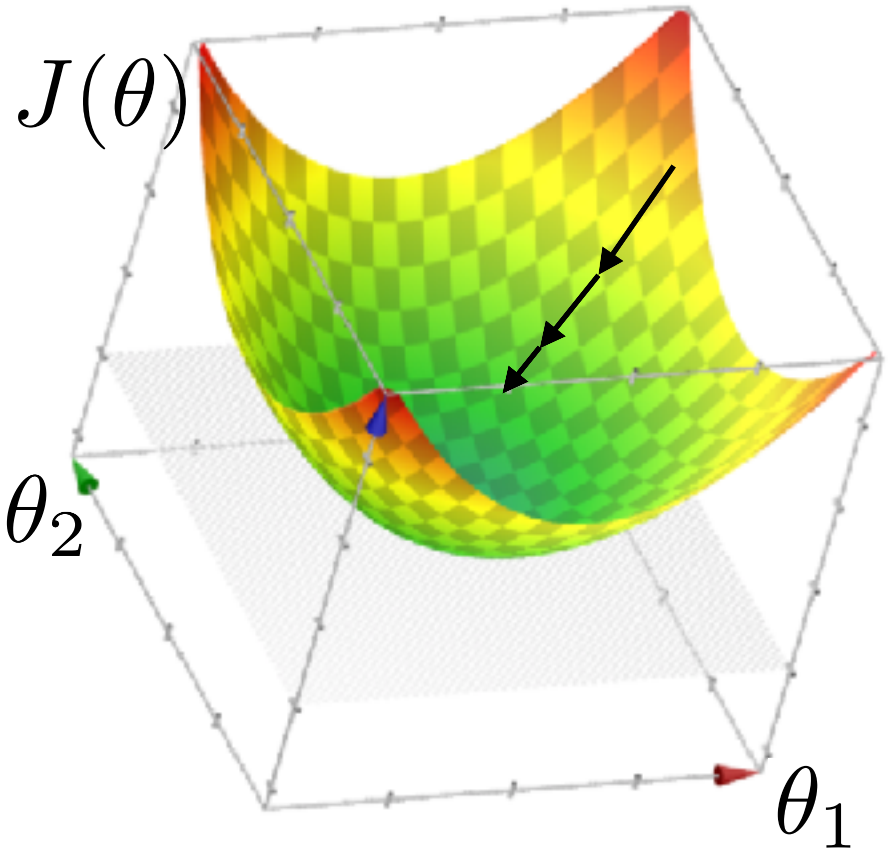


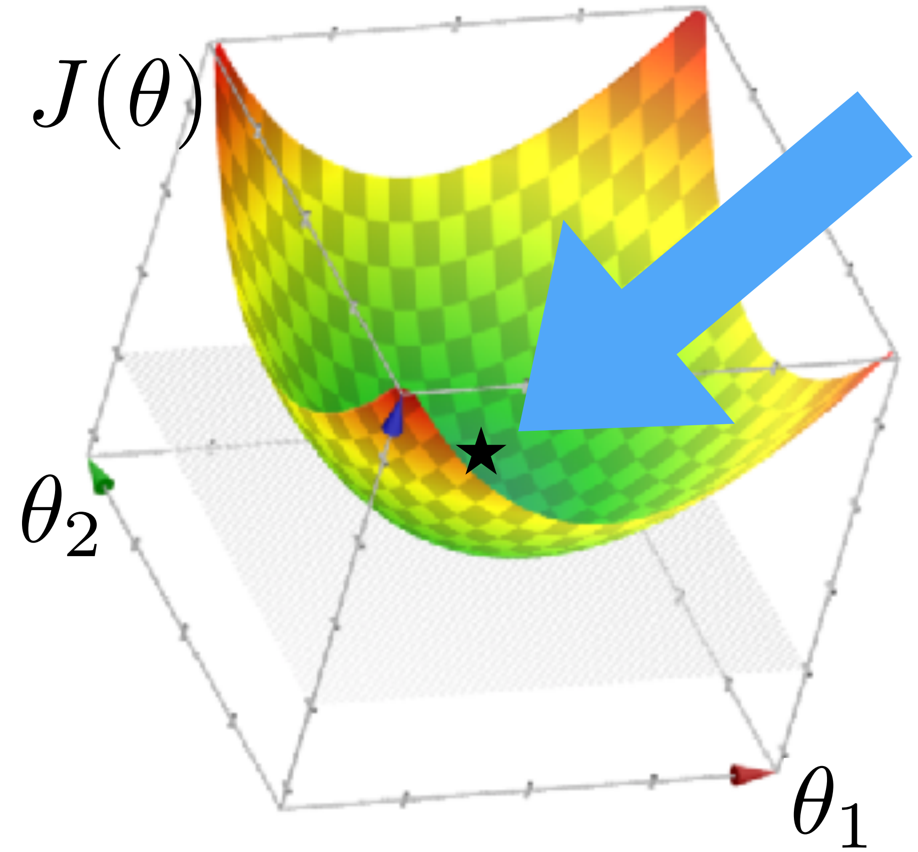
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - f is sufficiently "smooth" and convex
    - f has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ



Optimizing ridge regression

- Gradient descent vs. analytical/closed-form/direct solution





- Accuracy doesn't mean anything without running time
- Running time doesn't mean anything without accuracy
- Need to measure accuracy for the running time we have
  - Recall: closed-form solution (if no offset)

$$\theta = \underbrace{(\tilde{X}^\top \tilde{X} + n\lambda I)}_{\text{dxd}}{}^{-1} \tilde{X}^\top \tilde{Y}$$

Matrix inversion running time: $O(d^3)$

Gradient descent for ridge regression

- Gradient descent with *f* = ridge regression objective
  - For the moment, assume no offset (can extend)

Gradient-Descent($\Theta_{\text{init}}, \eta, f, \nabla_\Theta f$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  Initialize t = 0

  repeat

   t = t + 1

   $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

  until stopping criterion

  Return $\Theta^{(t)}$

Gradient descent for ridge regression

- Gradient descent with *f* = ridge regression objective
  - For the moment, assume no offset (can extend)

Gradient-Descent($\Theta_{\text{init}}, \eta, f, \nabla_\Theta f$)

Initialize $\theta^{(0)} = \theta_{\text{init}}$

Initialize t = 0

repeat

t = t + 1

$$\theta^{(t)} = \theta^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} 2 \left[\theta^{(t-1)\top} x^{(i)} - y^{(i)} \right] x^{(i)} + 2\lambda \theta^{(t-1)} \right\}$$

Exercise: Check!

until stopping criterion

Return $\Theta^{(t)}$

Gradient descent for ridge regression

- Gradient descent with *f* = ridge regression objective
  - For the moment, assume no offset (can extend)

```
Gradient-Descent( Θ_init, η, f, ∇_Θ f )
  Initialize θ^(0) = θ_init
  Initialize t = 0
  repeat
    t = t + 1
    θ^(t) = θ^(t-1) - η{ (1/n) Σ_{i=1}^{n} 2[θ^(t-1)ᵀ x^(i) - y^(i)] x^(i) + 2λθ^(t-1) }
  until stopping criterion
  Return θ^(t)
```

$$\theta^{(t)} = \theta^{(t-1)} - \eta\left\{\frac{1}{n}\sum_{i=1}^{n} 2\left[\theta^{(t-1)\top}x^{(i)} - y^{(i)}\right]x^{(i)} + 2\lambda\theta^{(t-1)}\right\}$$

Exercise: Check!

Gradient descent for ridge regression

- Gradient descent with *f* = ridge regression objective
  - For the moment, assume no offset (can extend)

```
RidgeRegression-Gradient-Descent( θ_init, η )
  Initialize θ^(0) = θ_init
  Initialize t = 0
  repeat
   t = t + 1
```

$$\theta^{(t)} = \theta^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} 2 \left[\theta^{(t-1)\top} x^{(i)} - y^{(i)} \right] x^{(i)} + 2\lambda \theta^{(t-1)} \right\}$$

```
  until stopping criterion
  Return θ^(t)
```

Exercise: Check!

Gradient descent for ridge regression

- Gradient descent with f = ridge regression objective
  - For the moment, assume no offset (can extend)

```
RidgeRegression-Gradient-Descent( θ_init, η )
  Initialize θ^(0) = θ_init
  Initialize t = 0
  repeat
    t = t + 1
```

$$\theta^{(t)} = \theta^{(t-1)} - \eta\left\{\frac{1}{n}\sum_{i=1}^{n} 2\left[\theta^{(t-1)\top}x^{(i)} - y^{(i)}\right]x^{(i)} + 2\lambda\theta^{(t-1)}\right\}$$

```
  until stopping criterion
  Return θ^(t)
```

Exercise: Check!

- No more matrix inversion! (see lab)

Gradient descent for ridge regression

- Gradient descent with *f* = ridge regression objective
  - For the moment, assume no offset (can extend)

```
RidgeRegression-Gradient-Descent( θ_init, η )
  Initialize θ^(0) = θ_init
  Initialize t = 0
  repeat
    t = t + 1
    θ^(t) = θ^(t-1) - η{ (1/n) Σ_{i=1}^{n} 2[θ^(t-1)T x^(i) - y^(i)] x^(i) + 2λθ^(t-1) }
  until stopping criterion
  Return θ^(t)
```

$$\theta^{(t)} = \theta^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} 2 \left[\theta^{(t-1)\top} x^{(i)} - y^{(i)} \right] x^{(i)} + 2\lambda \theta^{(t-1)} \right\}$$

Exercise: Check!

- No more matrix inversion! (see lab)
- But have to look at all *n* data points every step
- How to better handle large *n*?

Gradient descent for ridge regression

- Gradient descent with *f* = ridge regression objective
  - For the moment, assume no offset (can extend)

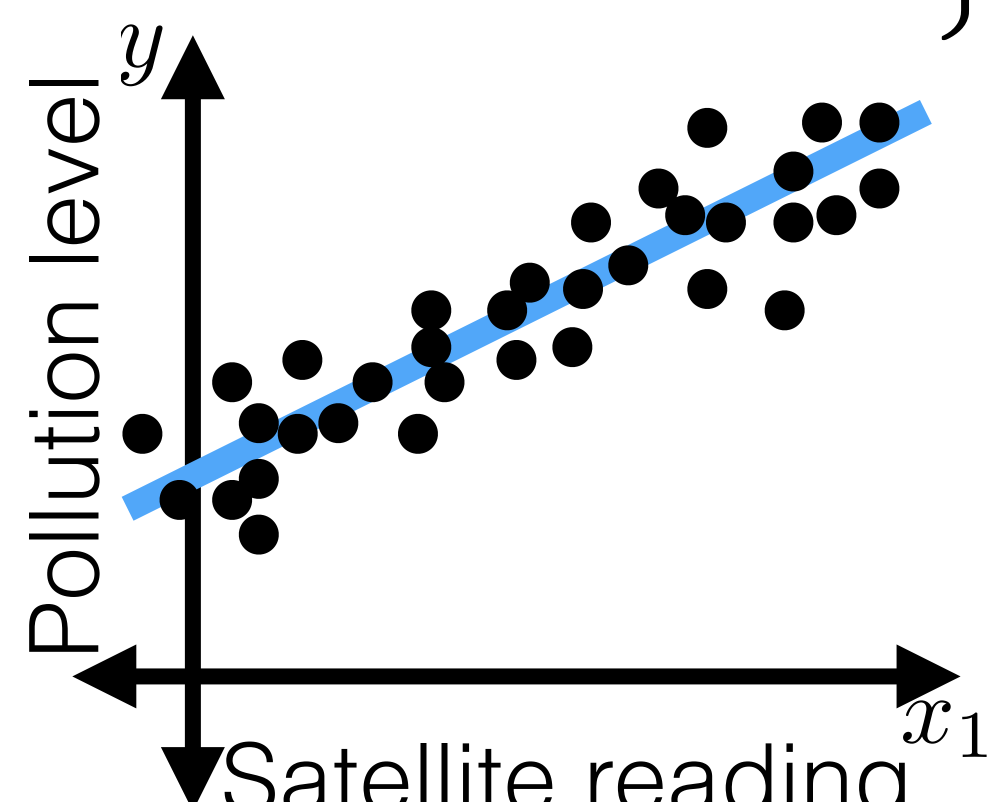
```
RidgeRegression-Gradient-Descent(θ_init, η)
  Initialize θ^(0) = θ_init
  Initialize t = 0
  repeat
    t = t + 1
```

$$\theta^{(t)} = \theta^{(t-1)} - \eta\left\{\frac{1}{n}\sum_{i=1}^{n} 2\left[\theta^{(t-1)\top}x^{(i)} - y^{(i)}\right]x^{(i)} + 2\lambda\theta^{(t-1)}\right\}$$

```
  until stopping criterion
  Return θ^(t)
```

Exercise: Check!

- No more matrix inversion! (see lab)
- But have to look at all *n* data points every step
- How to better handle large *n*?



Gradient descent for ridge regression

- Gradient descent with *f* = ridge regression objective
  - For the moment, assume no offset (can extend)

RidgeRegression-Gradient-Descent($\theta_{\text{init}}, \eta$)

Initialize $\theta^{(0)} = \theta_{\text{init}}$

Initialize t = 0

repeat

t = t + 1

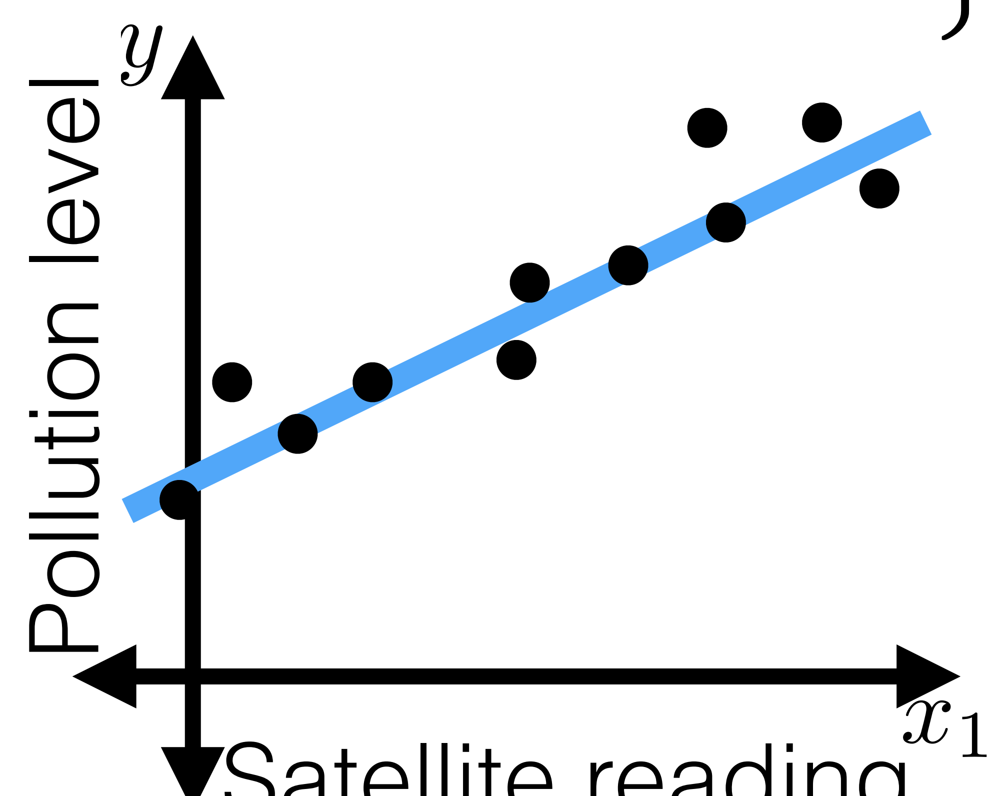
$$\theta^{(t)} = \theta^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} 2 \left[\theta^{(t-1)\top} x^{(i)} - y^{(i)} \right] x^{(i)} + 2\lambda\theta^{(t-1)} \right\}$$

until stopping criterion

Return $\theta^{(t)}$

Exercise: Check!

- No more matrix inversion! (see lab)
- But have to look at all *n* data points every step
- How to better handle large *n*?



Gradient descent for ridge regression

- Gradient descent with *f* = ridge regression objective
  - For the moment, assume no offset (can extend)

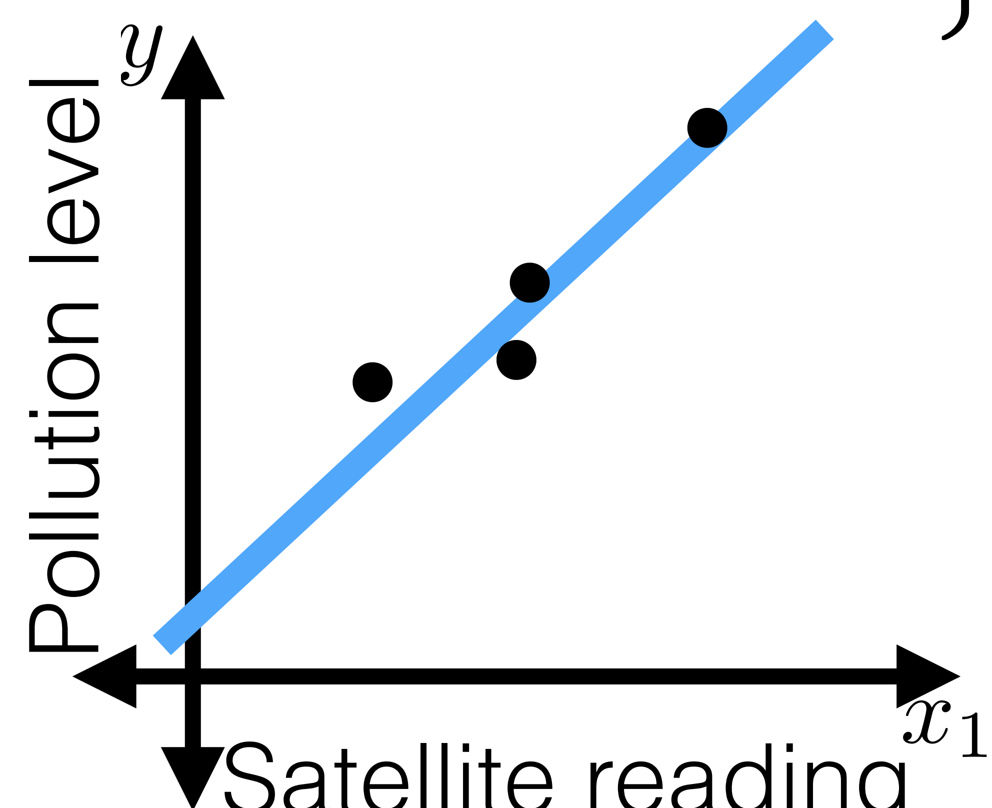
```
RidgeRegression-Gradient-Descent( θ_init, η )
  Initialize θ^(0) = θ_init
  Initialize t = 0
  repeat
    t = t + 1
```

$$\theta^{(t)} = \theta^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} 2\left[\theta^{(t-1)\top} x^{(i)} - y^{(i)}\right] x^{(i)} + 2\lambda\theta^{(t-1)} \right\}$$

```
  until stopping criterion
  Return θ^(t)
```

Exercise: Check!

- No more matrix inversion! (see lab)
- But have to look at all *n* data points every step
- How to better handle large *n*?



Stochastic gradient descent

- Linear regression objective (with $\lambda = 0$):

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\Theta)$$

Stochastic gradient descent

- Linear regression objective (with $\lambda = 0$):

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\Theta)$$

Stochastic gradient descent

- Linear regression objective (with $\lambda = 0$):

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\Theta)$$

- Stay tuned for more examples
- Compare to training error defn.

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

    randomly select i from {1,…,n} (with equal probability)

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta(t)\nabla_\Theta f_i(\Theta^{(t-1)})$

Compare to gradient descent update:
$\Theta^{(t)} = \Theta^{(t-1)} - \eta\nabla_\Theta f(\Theta^{(t-1)})$

Stochastic gradient descent

- Linear regression objective (with $\lambda = 0$):

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\Theta)$$

- Stay tuned for more examples
- Compare to training error defn.

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

    randomly select i from {1,…,n} (with equal probability)

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta(t)\nabla_\Theta f_i(\Theta^{(t-1)})$

Compare to gradient descent update:

$$\Theta^{(t)} = \Theta^{(t-1)} - \eta\nabla_\Theta f(\Theta^{(t-1)})$$

Stochastic gradient descent

- Linear regression objective (with $\lambda = 0$):

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\Theta)$$

- Stay tuned for more examples
- Compare to training error defn.

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

    randomly select i from {1,…,n} (with equal probability)

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta(t)\nabla_\Theta f_i(\Theta^{(t-1)})$

Compare to gradient descent update:

$$\Theta^{(t)} = \Theta^{(t-1)} - \eta\nabla_\Theta f(\Theta^{(t-1)})$$

Stochastic gradient descent

- Linear regression objective (with $\lambda = 0$):

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\Theta)$$

- Stay tuned for more examples
- Compare to training error defn.

```
Stochastic-Gradient-Descent(Θ_init, η, T)
  Initialize Θ^(0) = Θ_init
  for t = 1 to T
    randomly select i from {1,…,n} (with equal probability)
    Θ^(t) = Θ^(t−1) − η(t) ∇_Θ f_i(Θ^(t−1))
```

$\Theta^{(t)} = \Theta^{(t-1)} - \eta(t)\nabla_\Theta f_i(\Theta^{(t-1)})$

Compare to gradient descent update:

$\Theta^{(t)} = \Theta^{(t-1)} - \eta\nabla_\Theta f(\Theta^{(t-1)})$

Stochastic gradient descent

- Linear regression objective (with $\lambda = 0$):

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\Theta)$$

- Stay tuned for more examples
- Compare to training error defn.

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

    randomly select i from {1,…,n} (with equal probability)

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta(t) \nabla_\Theta f_i(\Theta^{(t-1)})$

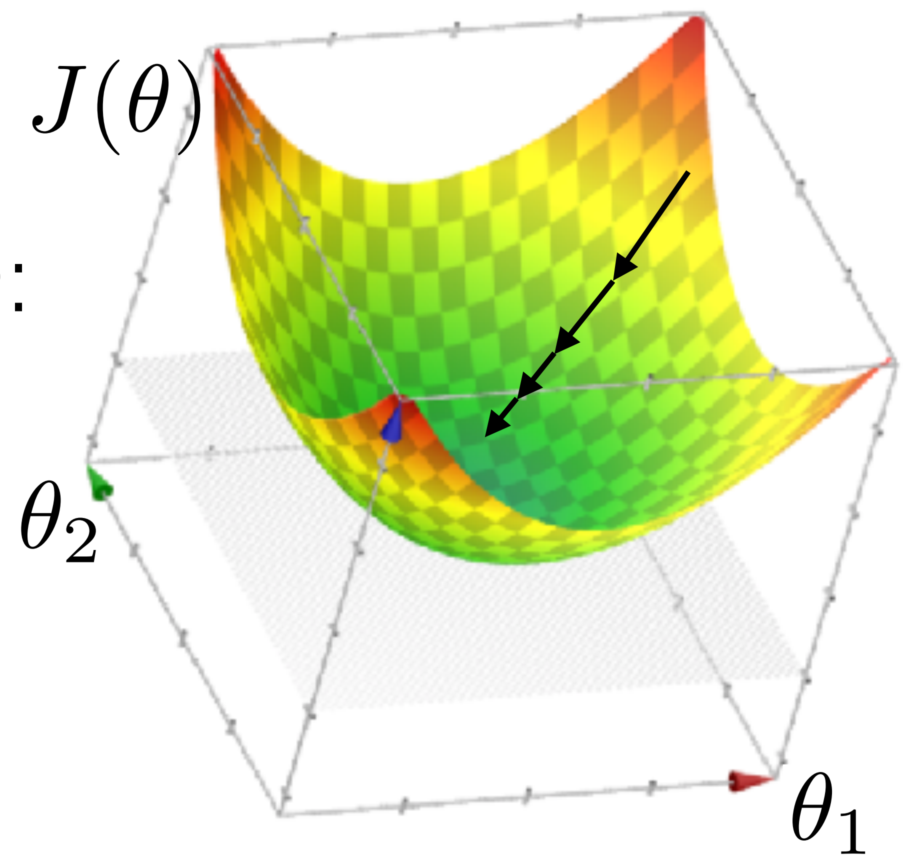
  Return $\Theta^{(t)}$

Compare to gradient descent update:
$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

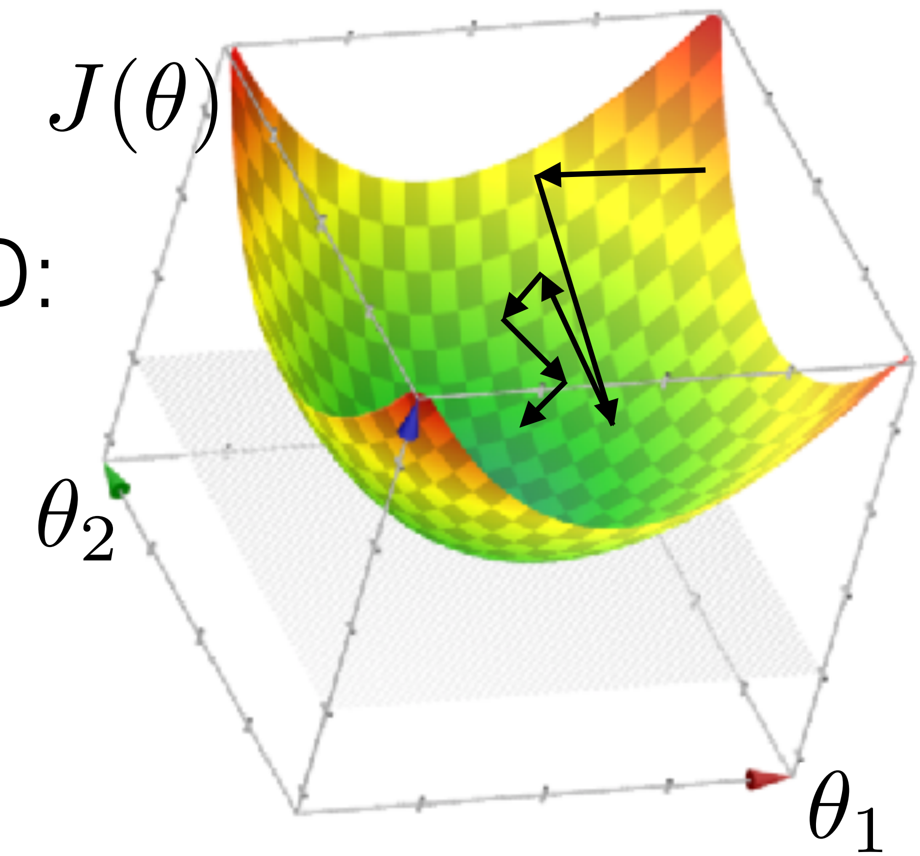
- Commonly used with “minibatches”

Stochastic gradient descent (SGD) properties

- GD:



- SGD:



- **Theorem**: SGD performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is "nice" & convex, has a unique global minimizer
    - $\sum_{t=1}^{\infty} \eta(t) = \infty, \sum_{t=1}^{\infty} (\eta(t))^2 < \infty$
      - e.g. $\eta(t) = \alpha(\tau_0 + t)^{-\kappa} (\kappa \in (0.5, 1])$
  - **Conclusion**: If run long enough, stochastic gradient descent will return a value within $\tilde{\epsilon}$ of the global minimizer