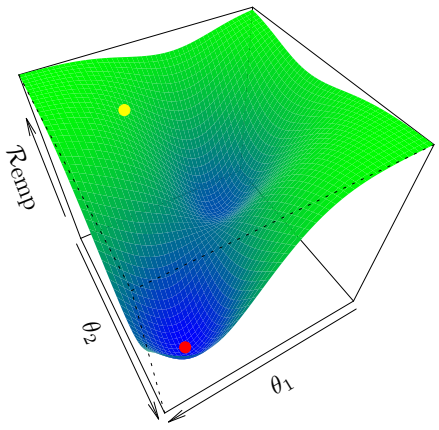
# Introduction to Machine Learning

# ML-Basics: Optimization

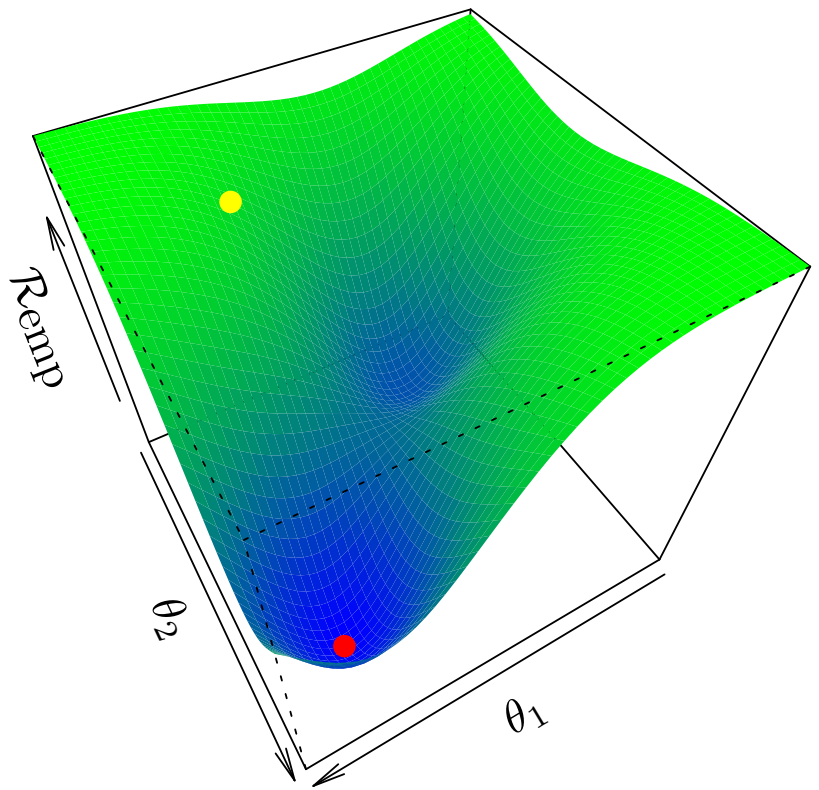



## Learning goals

- Understand how the risk function is optimized to learn the optimal parameters of a model
- Understand the idea of gradient descent as a basic risk optimizer

# LEARNING AS PARAMETER OPTIMIZATION

- We have seen, we can operationalize the search for a model $f$ that matches training data best, by looking for its parametrization $\boldsymbol{\theta} \in \Theta$ with lowest empirical risk $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$.
- Therefore, we usually traverse the error surface downwards; often by local search from a starting point to its minimum.



# LEARNING AS PARAMETER OPTIMIZATION

The ERM optimization problem is:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta} \in \Theta} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}).$$

For a **(global) minimum** $\hat{\boldsymbol{\theta}}$ it obviously holds that

$$\forall \boldsymbol{\theta} \in \Theta : \quad \mathcal{R}_{\text{emp}}(\hat{\boldsymbol{\theta}}) \leq \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}).$$

This does not imply that $\hat{\boldsymbol{\theta}}$ is unique.

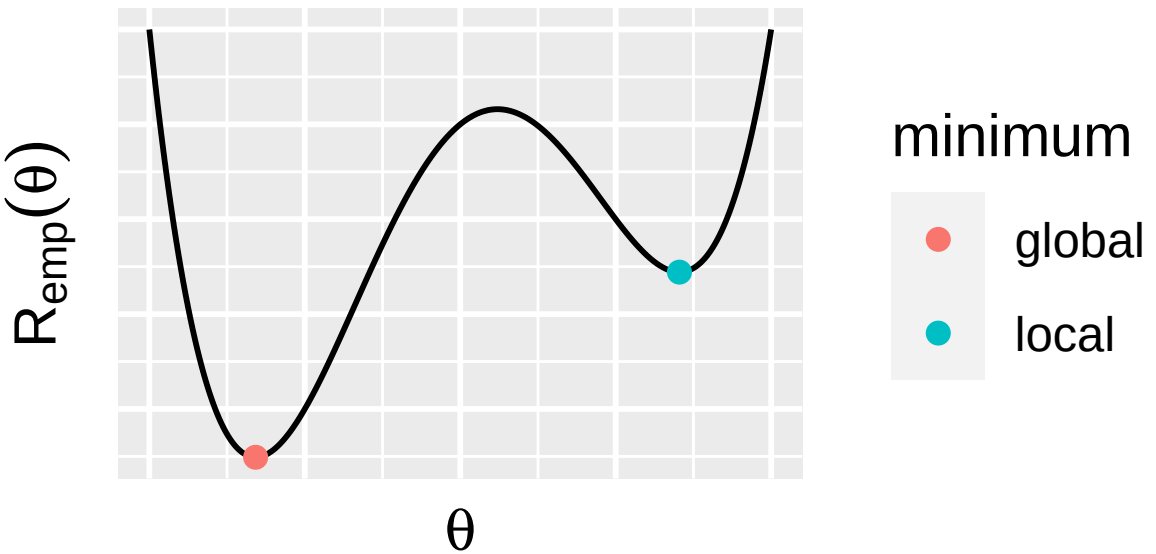
Which kind of numerical technique is reasonable for this problem strongly depends on model and parameter structure (continuous params? uni-modal $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$?). Here, we will only discuss very simple scenarios.

# LOCAL MINIMA

If $\mathcal{R}_{\text{emp}}$ is continuous in $\boldsymbol{\theta}$ we can define a **local minimum** $\hat{\boldsymbol{\theta}}$:

$$\exists \epsilon > 0 \; \forall \boldsymbol{\theta} \text{ with } \left\| \hat{\boldsymbol{\theta}} - \boldsymbol{\theta} \right\| < \epsilon : \quad \mathcal{R}_{\text{emp}}(\hat{\boldsymbol{\theta}}) \leq \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}).$$

Clearly every global minimum is also a local minimum. Finding a local minimum is easier than finding a global minimum.

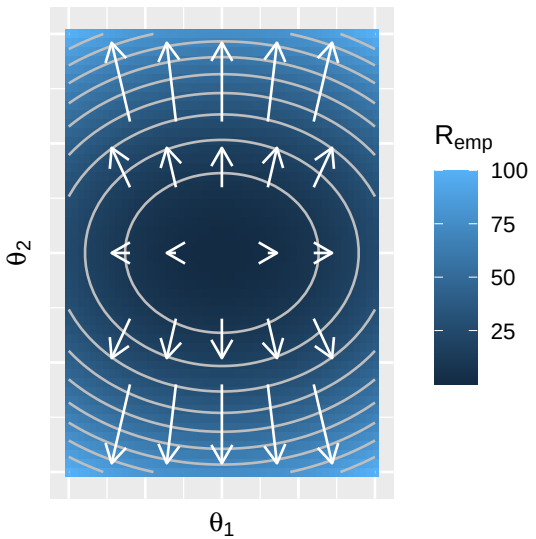


# LOCAL MINIMA AND STATIONARY POINTS

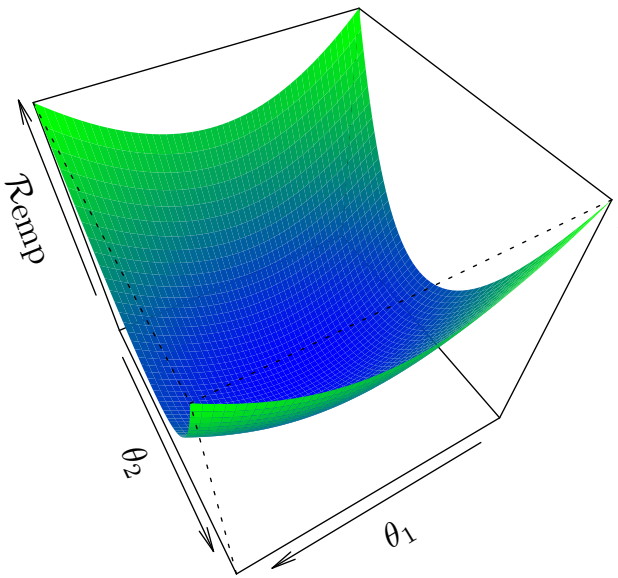
If $\mathcal{R}_{\text{emp}}$ is continuously differentiable in $\boldsymbol{\theta}$ then a **sufficient condition** for a local minimum is that $\hat{\boldsymbol{\theta}}$ is **stationary** with 0 gradient, so no local improvement is possible:

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{R}_{\text{emp}}(\hat{\boldsymbol{\theta}}) = 0$$

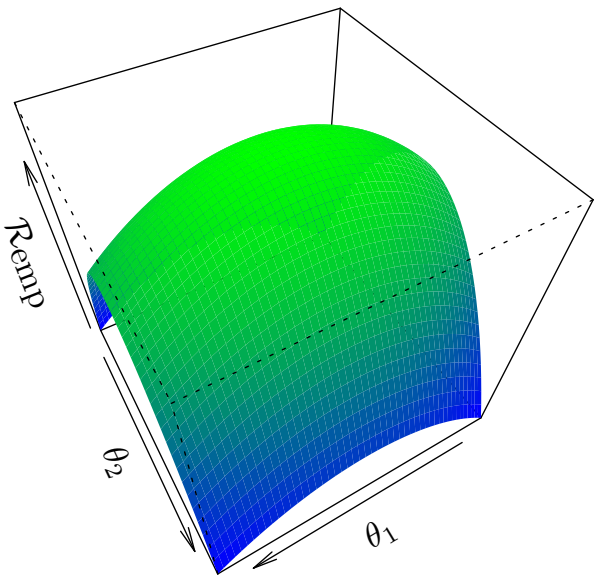
and the Hessian $\frac{\partial^2}{\partial \boldsymbol{\theta}^2} \mathcal{R}_{\text{emp}}(\hat{\boldsymbol{\theta}})$ is positive definite. While the neg. gradient points into the direction of fastest local decrease, the Hessian measures local curvature of $\mathcal{R}_{\text{emp}}$.




$\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$




const. pos. def. Hessian




const. neg. def. Hessian

# LEAST SQUARES ESTIMATOR

Now, for given features $\mathbf{X} \in \mathbb{R}^{n \times p}$ and target $\mathbf{y} \in \mathbb{R}^n$, we want to find the best linear model regarding the squared error loss, i.e.,

$$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \|\mathbf{X}\boldsymbol{\theta} - \mathbf{y}\|_2^2 = \sum_{i=1}^{n} (\boldsymbol{\theta}^\top \mathbf{x}^{(i)} - y^{(i)})^2 \ .$$

With the sufficient condition for continously differentiable functions it can be shown that the **least squares estimator**

$$\hat{\boldsymbol{\theta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

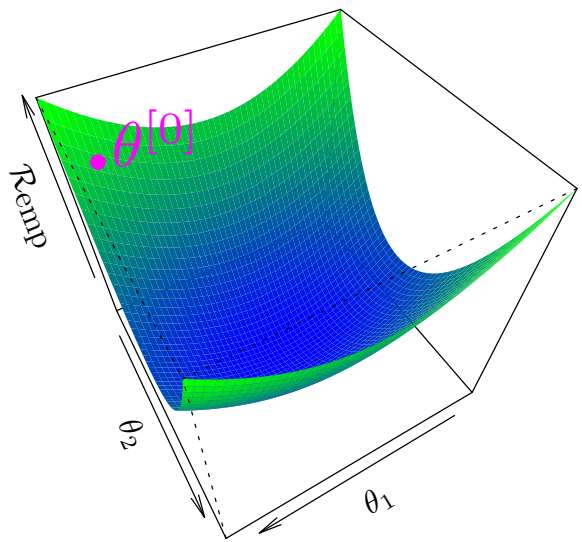
is a local minimum of $\mathcal{R}_{\text{emp}}$. If $\mathbf{X}$ is full-rank, $\mathcal{R}_{\text{emp}}$ is strictly convex and there is only one local minimum - which is also global.

**Note:** Often such analytical solutions in ML are not possible, and we rather have to use iterative numerical optimization.

# GRADIENT DESCENT

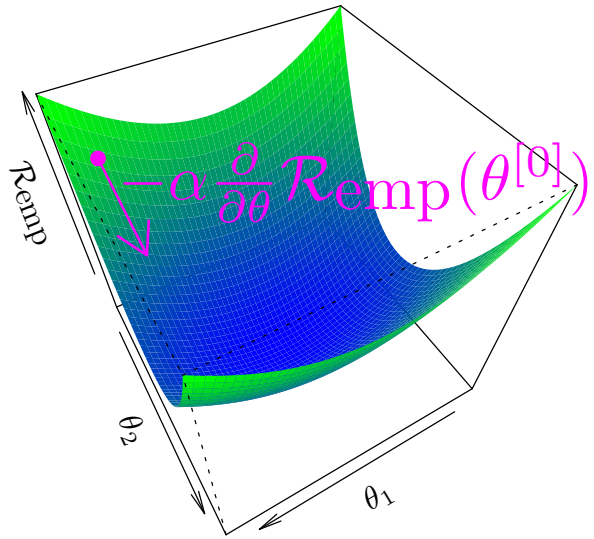
The simple idea of GD is to iteratively go from the current candidate $\boldsymbol{\theta}^{[t]}$ in the direction of the negative gradient, i.e., the direction of the steepest descent, with learning rate $\alpha$ to the next $\boldsymbol{\theta}^{[t+1]}$:

$$\boldsymbol{\theta}^{[t+1]} = \boldsymbol{\theta}^{[t]} - \alpha \frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}^{[t]}).$$

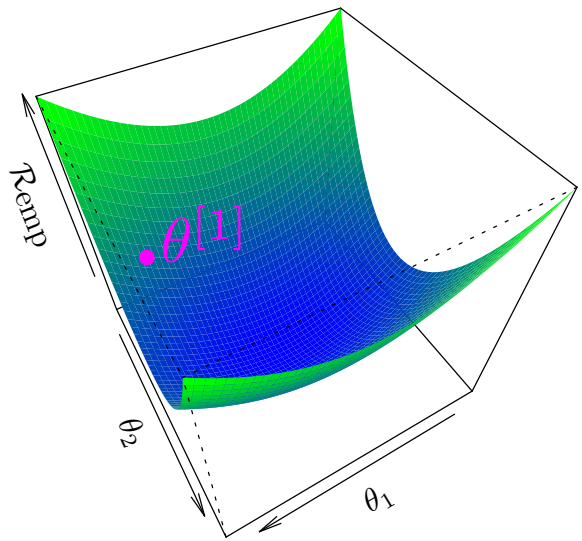



We choose a random start $\boldsymbol{\theta}^{[0]}$ with risk $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}^{[0]}) = 76.25$.

# GRADIENT DESCENT - EXAMPLE

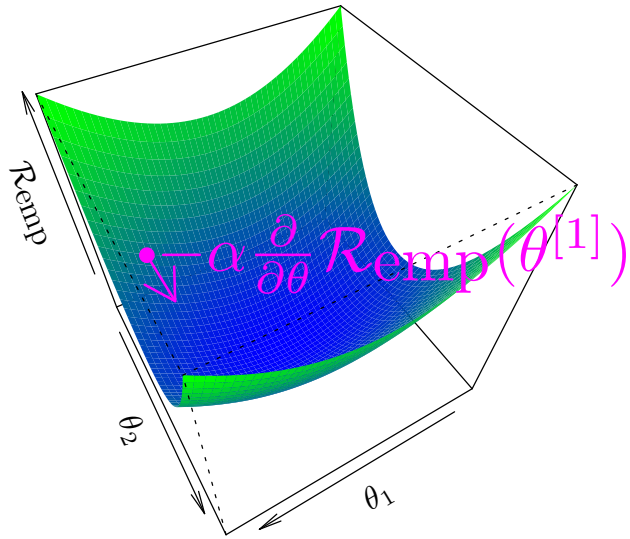



Now we follow in the direction of the negative gradient at $\theta^{[0]}$.

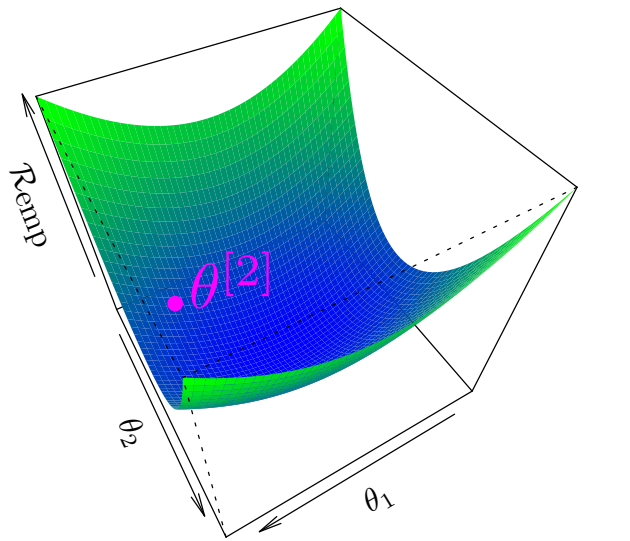



We arrive at $\theta^{[1]}$ with risk
$\mathcal{R}_{\text{emp}}(\theta^{[1]}) \approx 42.73$.
We improved:
$\mathcal{R}_{\text{emp}}(\theta^{[1]}) < \mathcal{R}_{\text{emp}}(\theta^{[0]})$.

# GRADIENT DESCENT - EXAMPLE

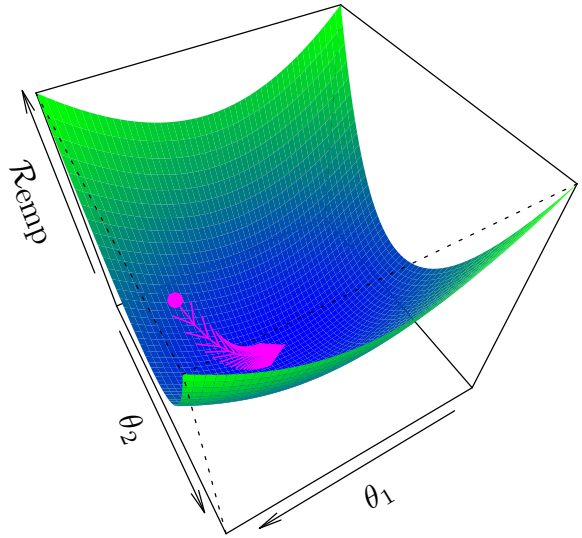



Again we follow in the direction of the negative gradient, but now at $\boldsymbol{\theta}^{[1]}$.

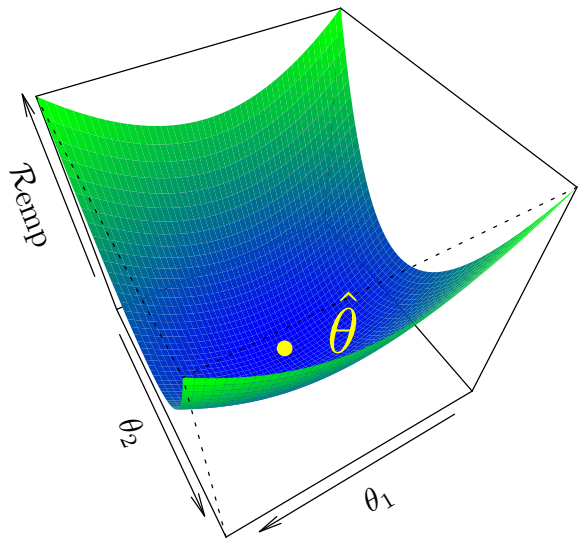



Now $\boldsymbol{\theta}^{[2]}$ has risk $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}^{[2]}) \approx 25.08$.

# GRADIENT DESCENT - EXAMPLE



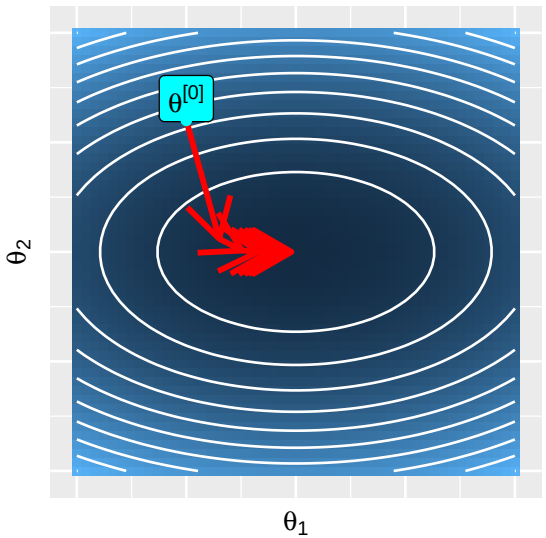

We iterate this until some form of convergence or termination.



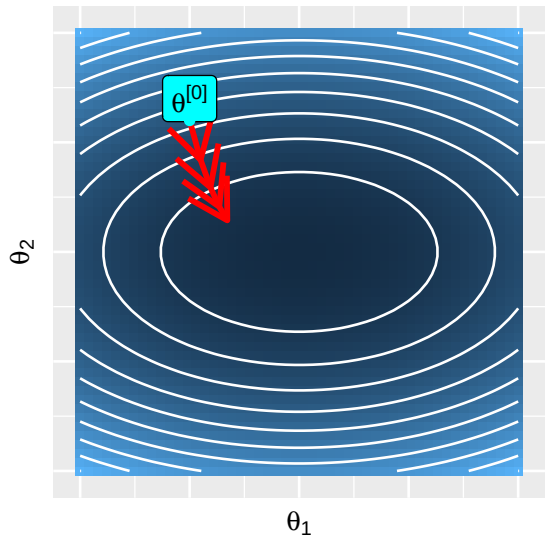

We arrive close to a stationary $\hat{\theta}$ which is hopefully at least a local minimum.

# GRADIENT DESCENT - LEARNING RATE

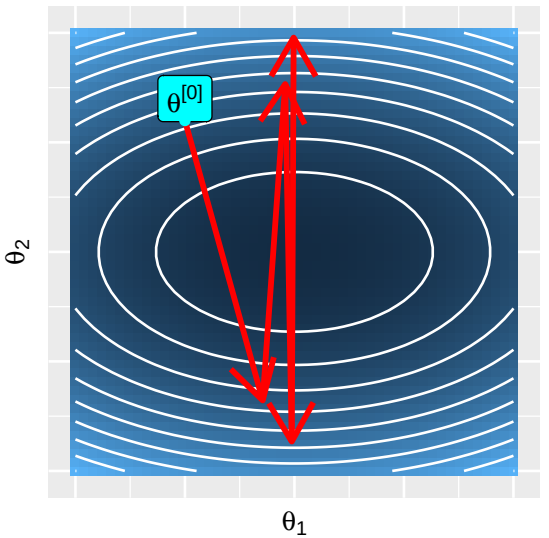
- The negative gradient is a direction that looks locally promising to reduce $\mathcal{R}_{\text{emp}}$.
- Hence it weights components higher in which $\mathcal{R}_{\text{emp}}$ decreases more.
- However, the length of $-\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{R}_{\text{emp}}$ measures only the local decrease rate, i.e., there are no guarantees that we will not go "too far".
- We use a learning rate $\alpha$ to scale the step length in each iteration. Too much can lead to overstepping and no converge, too low leads to slow convergence.
- Usually, a simple constant rate or rate-decrease mechanisms to enforce local convergence are used




good convergence for $\alpha_1$




poor convergence for $\alpha_2 (< \alpha_1)$




no convergence for $\alpha_3 (> \alpha_1)$

# FURTHER TOPICS

- GD is a so-called first-order method. Second-order methods use the Hessian to refine the search direction for faster convergence.
- There exist many improvements of GD, e.g., to smartly control the learn rate, to escape saddle points, to mimic second order behavior without computing the expensive Hessian.
- If the gradient of GD is not derived from the empirical risk of the whole data set, but instead from a randomly selected subset, we call this **stochastic gradient descent** (SGD). For large-scale problems this can lead to higher computational efficiency.