EXCURSUS: NUMERICAL OPTIMIZATION

We are searching for the parameter $\boldsymbol{\theta} \in \Theta$ which minimizes the empirical risk

$$\min_{\boldsymbol{\theta} \in \Theta} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta} \in \Theta} \sum_{i=1}^{n} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\right)$$

What if there is no closed-form solution to the problem above?

$\longrightarrow$ Numerical Optimization

Gradient Descent

GRADIENT DESCENT

- To find local minima, gradient descent takes steps in direction of the **negative gradient** of $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$ at the current point $\boldsymbol{\theta}^{[t]}$.
- The iterative update rule is

$$\boldsymbol{\theta}^{[t+1]} = \boldsymbol{\theta}^{[t]} - \alpha^{[t]} \cdot \nabla_{\boldsymbol{\theta}} \, \mathcal{R}_{\text{emp}}(\boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{[t]}} \, ,$$

  where the **step-size** $\alpha^{[t]} \in [0, 1]$ needs to be chosen (fixed, line-search, ...).
- As it uses the gradient, gradient descent is a first-order iterative optimization algorithm.

GD IN ML AND PSEUDO-RESIDUALS

By using the chain rule we see that

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) &= \sum_{i=1}^{n} \underbrace{\left.\frac{\partial L\left(y^{(i)}, f\right)}{\partial f}\right|_{f=f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)}}_{=-\tilde{r}^{(i)}} \cdot \nabla_{\boldsymbol{\theta}} f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right) \\ &= -\sum_{i=1}^{n} \tilde{r}^{(i)} \cdot \nabla_{\boldsymbol{\theta}} f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\end{aligned}$$

For risk minimization, the update rule for the parameter $\boldsymbol{\theta}$ is

$$\begin{aligned}\boldsymbol{\theta}^{[t+1]} &\leftarrow \boldsymbol{\theta}^{[t]} - \alpha^{[t]} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}}\, L\left(y^{(i)}, f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\right)\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{[t]}} \\ \boldsymbol{\theta}^{[t+1]} &\leftarrow \boldsymbol{\theta}^{[t]} + \alpha^{[t]} \sum_{i=1}^{n} \tilde{r}^{(i)} \cdot \nabla_{\boldsymbol{\theta}} f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{[t]}}\end{aligned}$$

$\alpha^{[t]} \in [0, 1]$ is called “learning rate” in this context.

GRADIENT DESCENT FOR HUBER LOSS

Example: Huber Loss

We can calculate the pseudo-residuals for the Huber loss

$$\tilde{r}^{(i)} = \begin{cases} -\frac{\partial}{\partial f}\frac{1}{2}\left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right)^2 = \left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right) & \text{if } |y^{(i)} - f\left(\mathbf{x}^{(i)}\right)| \leq \delta \\ -\frac{\partial}{\partial f}\delta|y^{(i)} - f\left(\mathbf{x}^{(i)}\right)| - \frac{1}{2}\delta^2 = \delta \cdot \text{sgn}\left(y^{(i)} - f\left(\mathbf{x}^{(i)}\right)\right) & \text{otherwise} \end{cases}$$

For the constant model $f(\mathbf{x}) = \theta$ this results in the following update rule:

$$\theta^{[t+1]} \quad \leftarrow \quad \theta^{[t]} + \alpha^{[t]} \sum_{i=1}^{n} \tilde{r}^{(i)}$$

The steps of gradient descent are shown in the following plots. The orange line shows the “optimal” constant model w.r.t. Huber loss, the black line shows the iterations of gradient descent.

GRADIENT DESCENT FOR HUBER LOSS

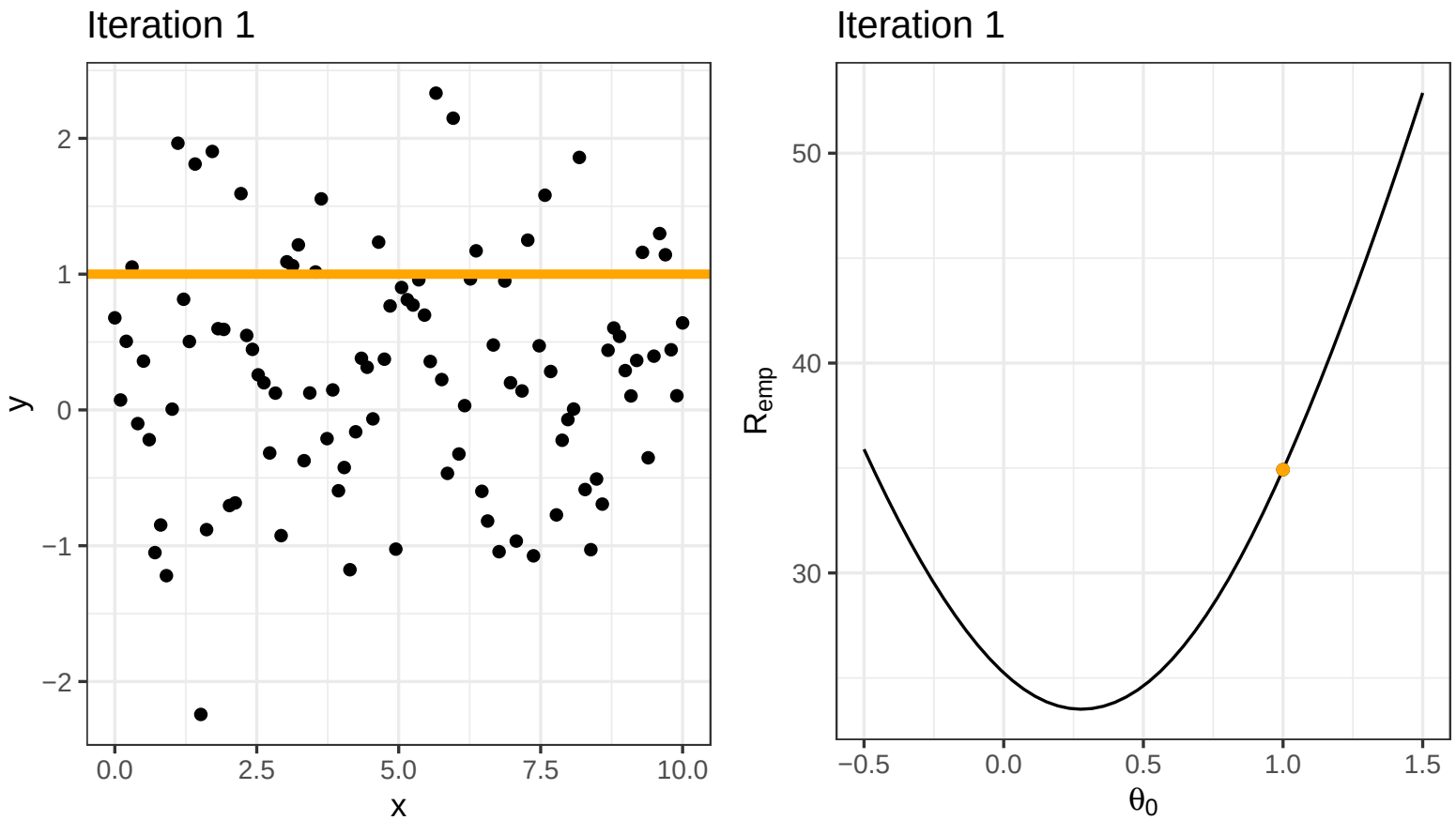

Figure: Huber loss optimized by gradient descent for a constant model $f(\mathbf{x}) = \theta$.

GRADIENT DESCENT FOR HUBER LOSS

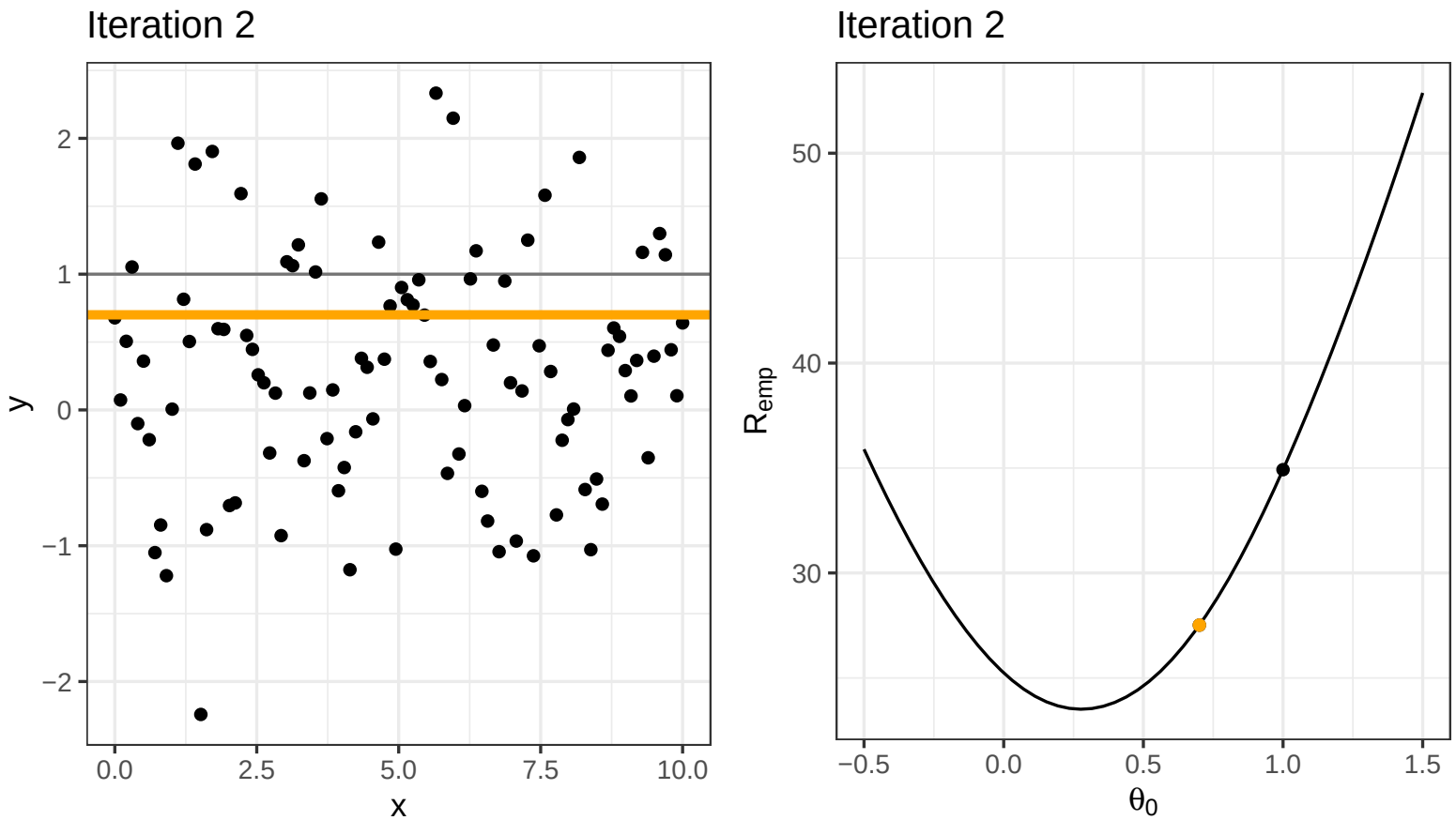

Figure: Huber loss optimized by gradient descent for a constant model $f(\mathbf{x}) = \theta$.

GRADIENT DESCENT FOR HUBER LOSS

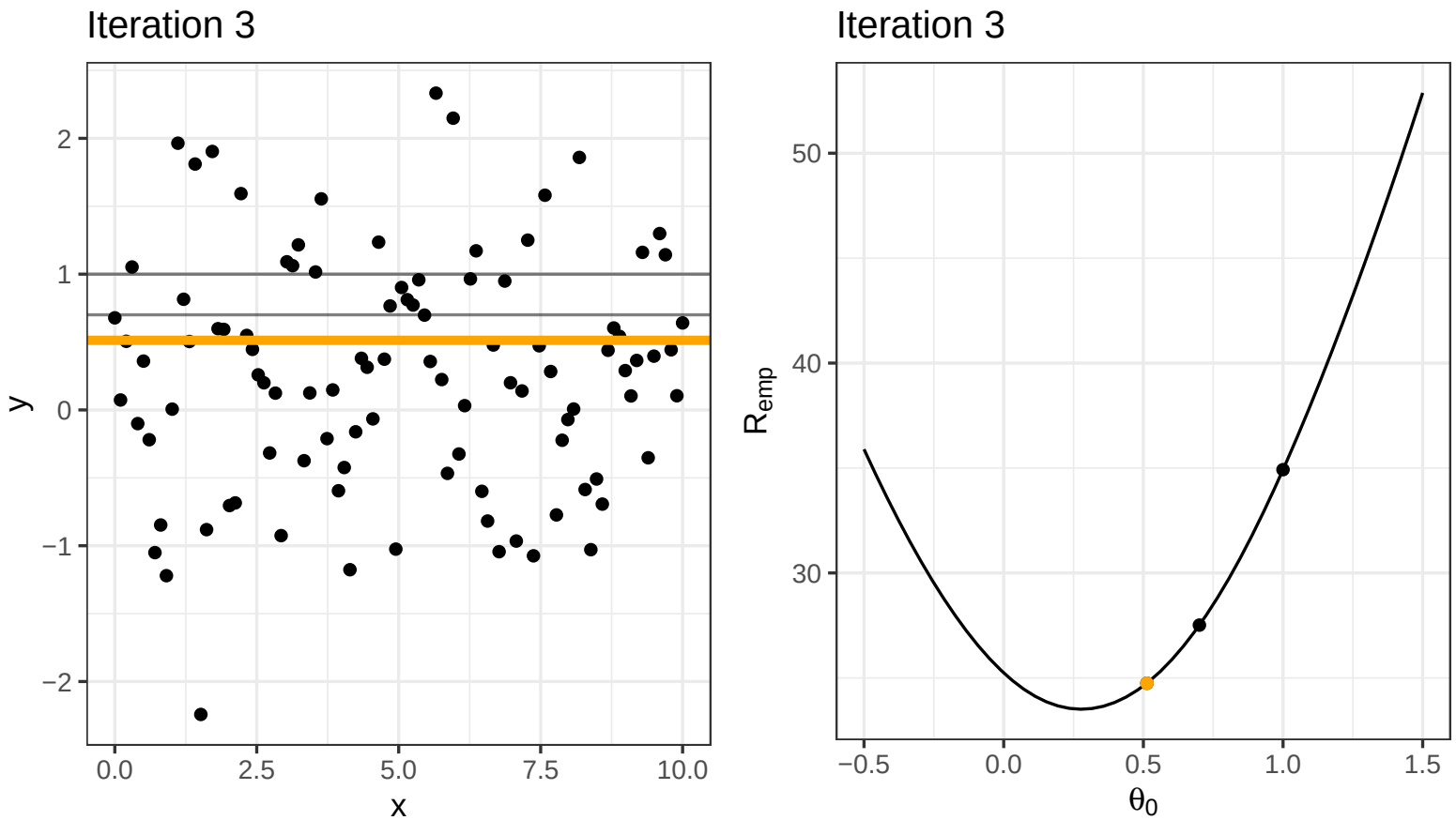

Figure: Huber loss optimized by gradient descent for a constant model $f(\mathbf{x}) = \theta$.

GRADIENT DESCENT FOR HUBER LOSS

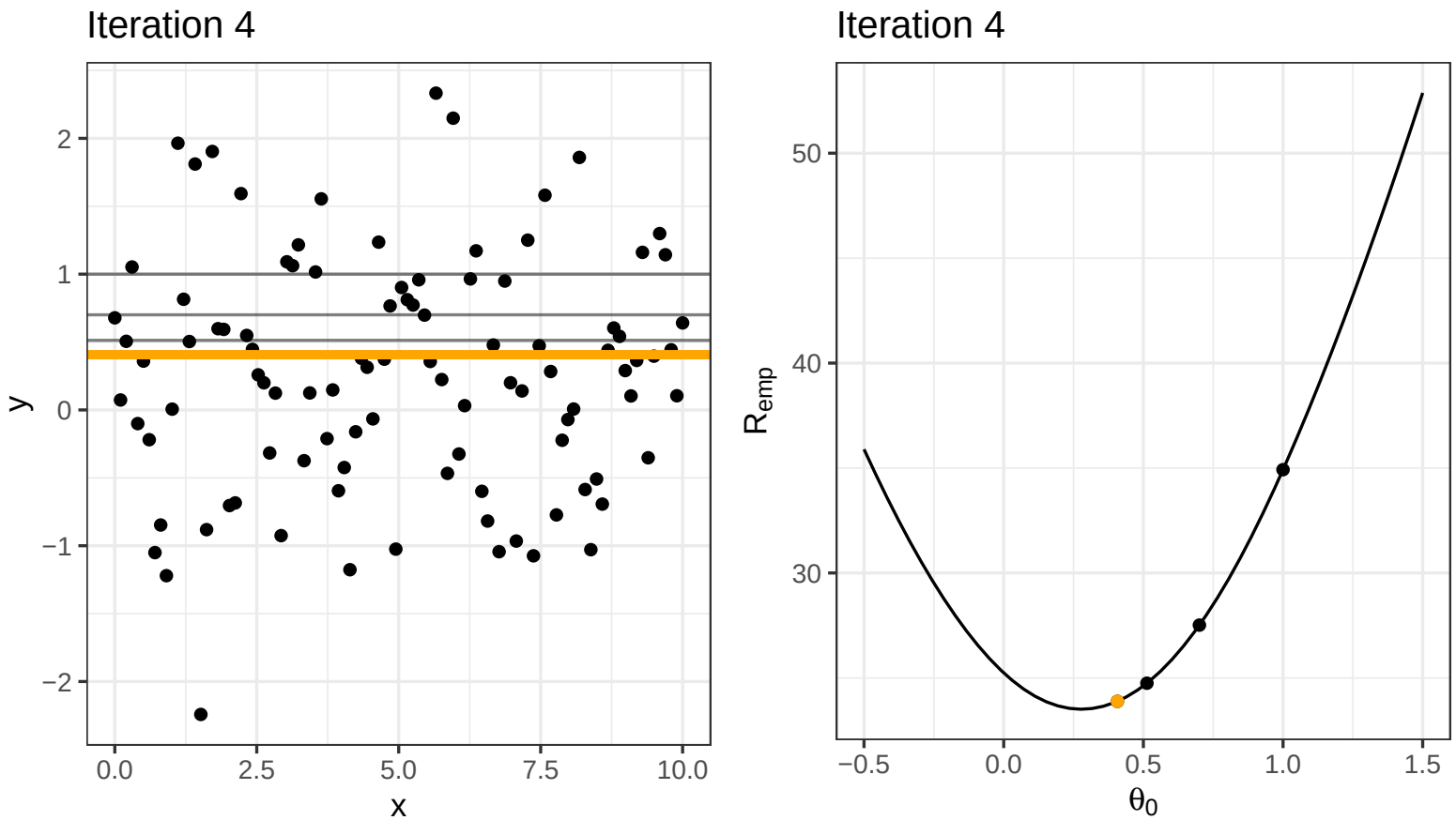

Figure: Huber loss optimized by gradient descent for a constant model $f(\mathbf{x}) = \theta$.

GRADIENT DESCENT FOR HUBER LOSS

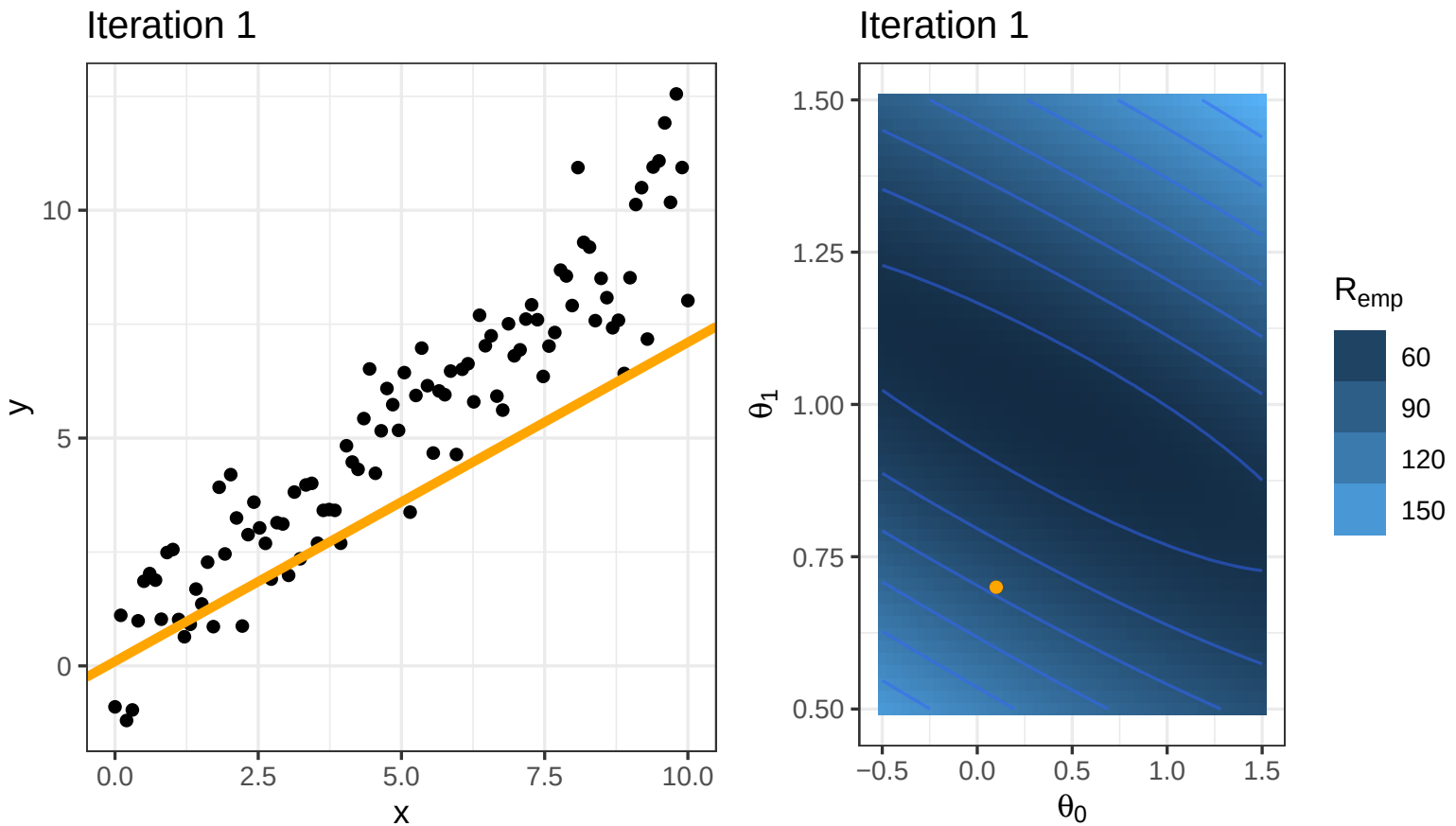

Figure: Huber loss optimized by gradient descent for a linear model $f(\mathbf{x}) = \theta^\top \mathbf{x}$.

GRADIENT DESCENT FOR HUBER LOSS

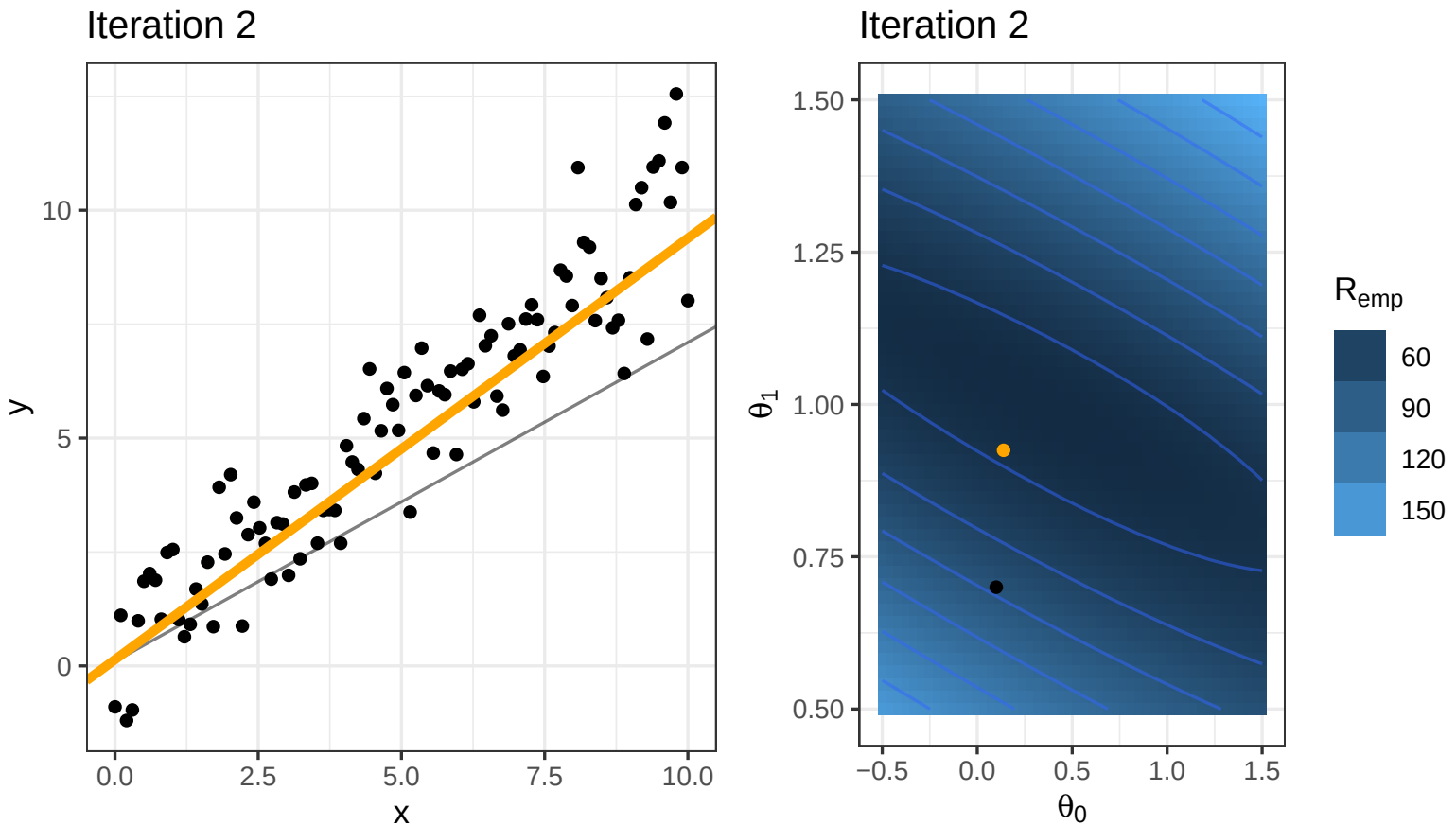

Figure: Huber loss optimized by gradient descent for a linear model $f(\mathbf{x}) = \theta^\top \mathbf{x}$.

GRADIENT DESCENT FOR HUBER LOSS

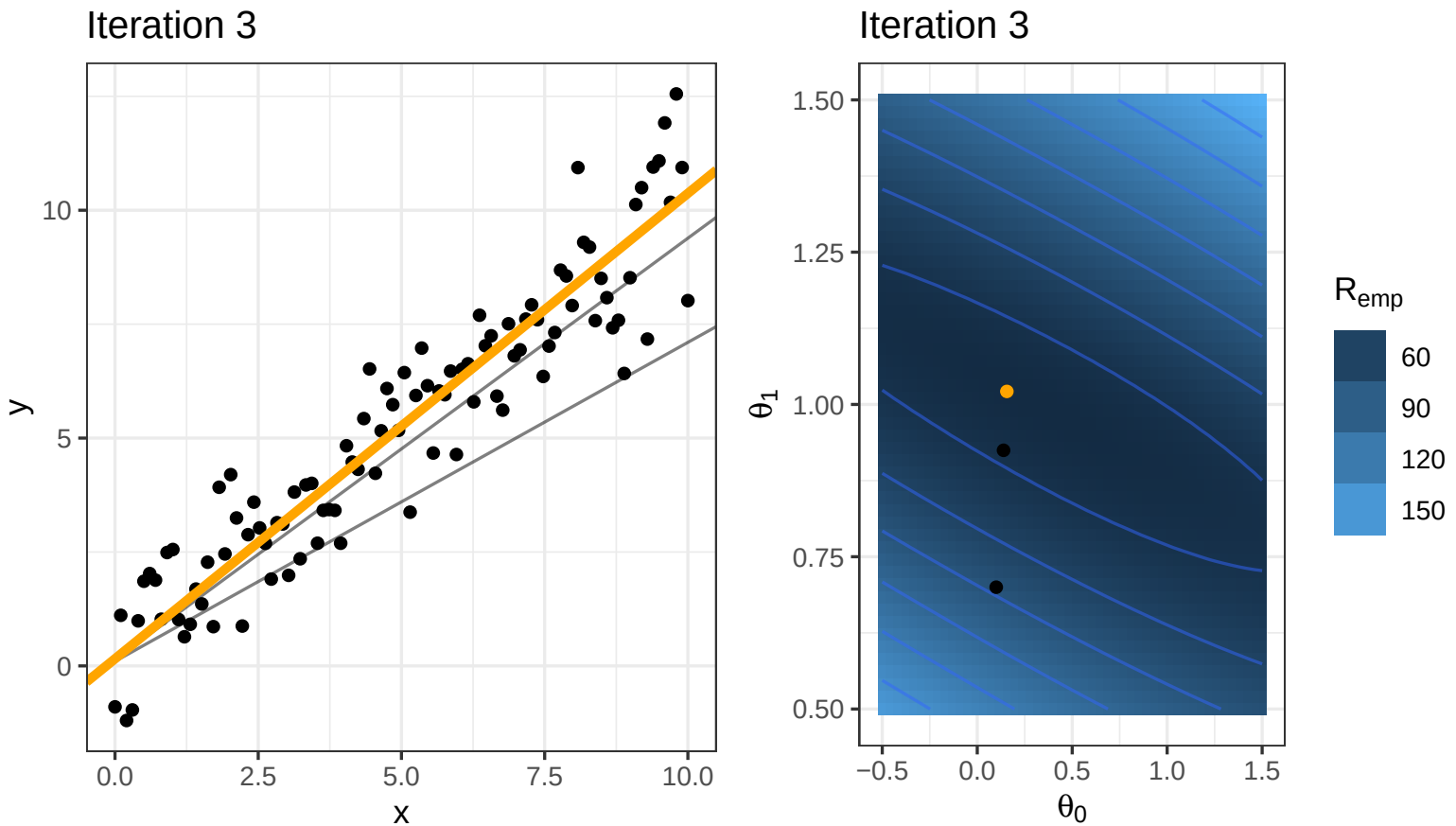

Figure: Huber loss optimized by gradient descent for a linear model $f(\mathbf{x}) = \theta^\top \mathbf{x}$.

GRADIENT DESCENT FOR HUBER LOSS

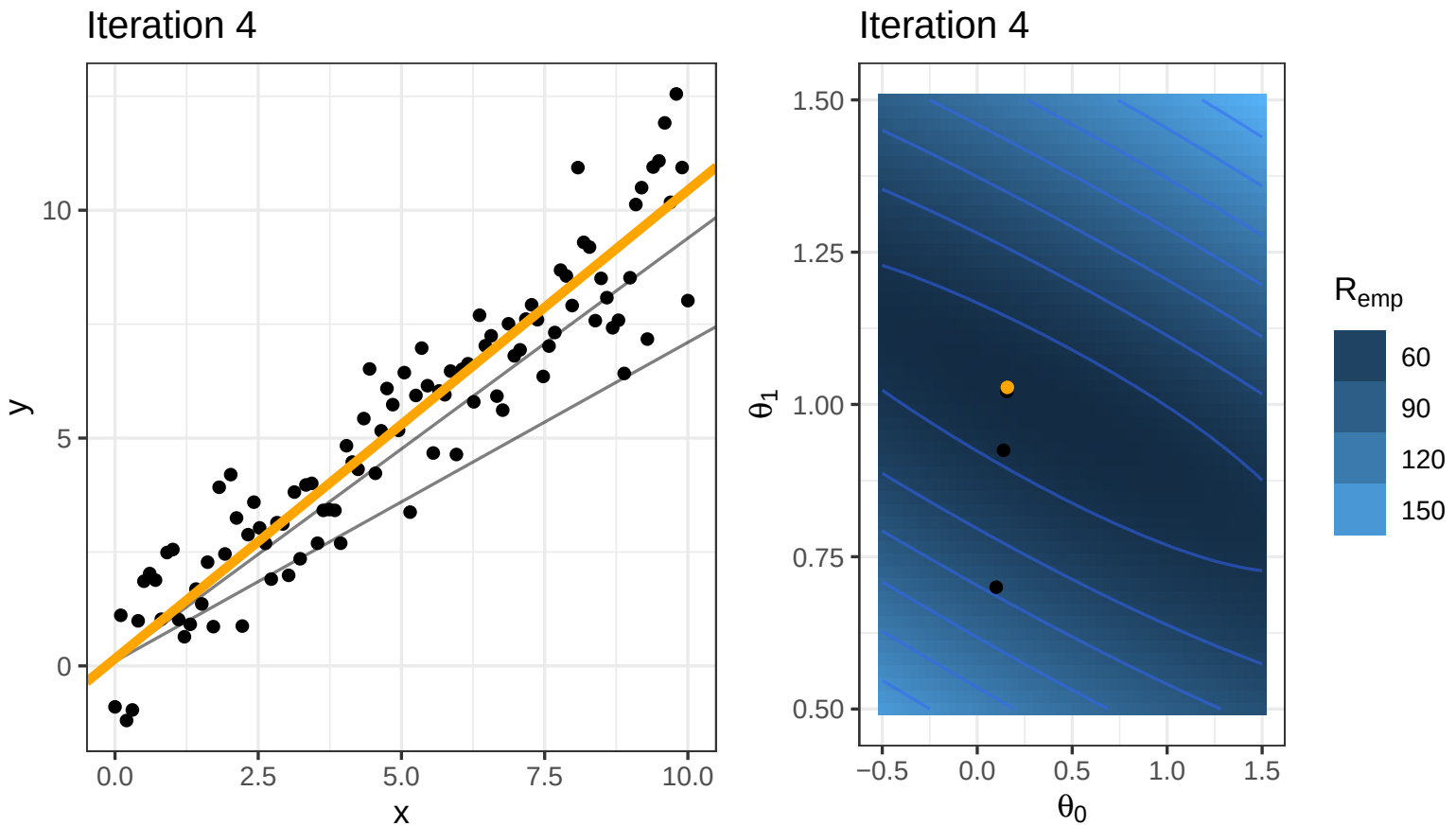

Figure: Huber loss optimized by gradient descent for a linear model $f(\mathbf{x}) = \theta^\top \mathbf{x}$.

Stochastic Gradient Descent

STOCHASTIC GRADIENT DESCENT

- **Stochastic Gradient Descent (SGD)** is a stochastic approximation of gradient descent.
- SGD is applied if $\sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}^{[t]}} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\right)$ is expensive in terms of evaluations (every summand needs to be evaluated).
- SGD approximates the gradient using just a random observation i leading to a simplified updating rule:

$$\boldsymbol{\theta}^{[t+1]} \leftarrow \boldsymbol{\theta}^{[t]} - \alpha^{[t]} \nabla_{\theta} \, L\left(y^{(i)}, f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\right)\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{[t]}}$$

- The sequence of parameters $\{\boldsymbol{\theta}^{[1]}, \boldsymbol{\theta}^{[2]}, \ldots\}$ is stochastic since it depends on the randomly drawn observation in every step.

SGD MINI BATCHES

- Stochastic gradient is computationally cheap compared to standard gradient descent, but might be very noisy.
- A trade-off between standard gradient descent (uses all observations for computation of the gradient) and stochastic gradient (uses one observation for approximation of the gradient) is **mini-batch gradient descent**.
- Mini-batch gradient descent uses a set of randomly drawn observations $I \subset \{1, 2, ..., n\}$ for approximation of the gradient

$$\boldsymbol{\theta}^{[t+1]} \leftarrow \boldsymbol{\theta}^{[t]} - \alpha^{[t]} \sum_{i \in I} \nabla_\theta \, L\left(y^{(i)}, f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\right)\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{[t]}}$$

For further details on multivariate optimization see CIM1 - Statistical Computing.