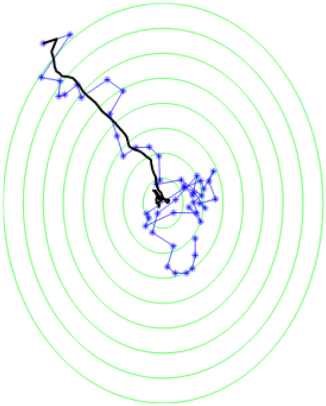
Optimization in Machine Learning

First order methods: SGD



Learning goals

- SGD
- Stochasticity
- Convergence
- Batch size

STOCHASTIC GRADIENT DESCENT

NB: We use g instead of f as objective, bc. f is used as model in ML.

$g : \mathbb{R}^d \to \mathbb{R}$ objective, g **average over functions**:

$$g(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} g_i(\mathbf{x}), \qquad g \text{ and } g_i \text{ smooth}$$

Stochastic gradient descent (SGD) approximates the gradient

$$\begin{aligned} \nabla_{\mathbf{x}}\, g(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \nabla_{\mathbf{x}}\, g_i(\mathbf{x}) \quad &:= \quad \mathbf{d} \quad \text{by} \\ \frac{1}{|J|} \sum_{i \in J} \nabla_{\mathbf{x}}\, g_i(\mathbf{x}) \quad &:= \quad \hat{\mathbf{d}}, \end{aligned}$$

with random subset $J \subset \{1, 2, ..., n\}$ of gradients called **mini-batch**.
This is done e.g. when computing the true gradient is **expensive**.

STOCHASTIC GRADIENT DESCENT

Algorithm Basic SGD pseudo code

Initialize $\mathbf{x}^{[0]}$, $t = 0$
while stopping criterion not met **do**
    Randomly shuffle indices and partition into minibatches $J_1, ..., J_K$ of size m
    for $k \in \{1, ..., K\}$ **do**
        $t \leftarrow t + 1$
        Compute gradient estimate with J_k: $\hat{\mathbf{d}}^{[t]} \leftarrow \frac{1}{m} \sum_{i \in J_k} \nabla_{\mathbf{x}} g_i(\mathbf{x}^{[t-1]})$
        Apply update: $\mathbf{x}^{[t]} \leftarrow \mathbf{x}^{[t-1]} - \alpha \cdot \hat{\mathbf{d}}^{[t]}$
    end for
end while

- Instead of drawing batches randomly we might want to go through the g_i sequentially (unless g_i are sorted in any way)
- Updates are computed faster, but also more stochastic:
  - In the simplest case, batch-size $m := |J_k|$ is set to $m = 1$
  - If n is a billion, computation of update is a billion times faster
  - **But** (later): Convergence rates suffer from stochasticity!

SGD IN ML

In ML, we perform ERM:

$$\mathcal{R}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} \underbrace{L\left(y^{(i)}, f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\right)}_{g_i(\boldsymbol{\theta})}$$

- for a data set

$$\mathcal{D} = \left(\left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(n)}, y^{(n)}\right)\right)$$

- a loss function $L(y, f(\mathbf{x}))$, e.g., L2 loss $L(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$,
- and a model class f, e.g., the linear model $f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right) = \boldsymbol{\theta}^\top \mathbf{x}$.

SGD IN ML

For large data sets, computing the exact gradient

$$\mathbf{d} = \frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} L \left(y^{(i)}, f \left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta} \right) \right)$$

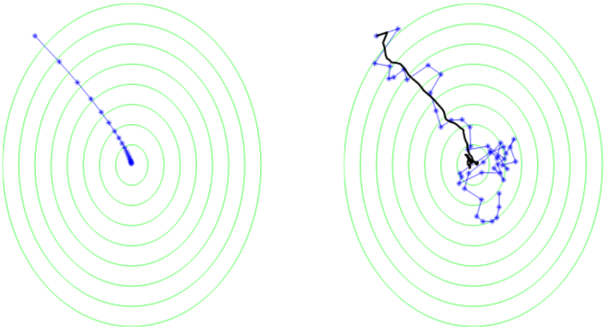
may be expensive or even infeasible to compute and is approximated by

$$\hat{\mathbf{d}} = \frac{1}{m} \sum_{i \in J} \nabla_{\boldsymbol{\theta}} L \left(y^{(i)}, f \left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta} \right) \right),$$

for $J \subset 1, 2, ..., n$ random subset.

NB: Often, maximum size of J technically limited by memory size.

STOCHASTICITY OF SGD



Minimize $g(x_1, x_2) = 1.25(x_1 + 6)^2 + (x_2 - 8)^2$.
Left: GD. **Right:** SGD. Black line shows average value across multiple runs.
(Source: Shalev-Shwartz et al., Understanding Machine Learning, 2014.)

STOCHASTICITY OF SGD

Assume batch size $m = 1$ (statements also apply for larger batches).

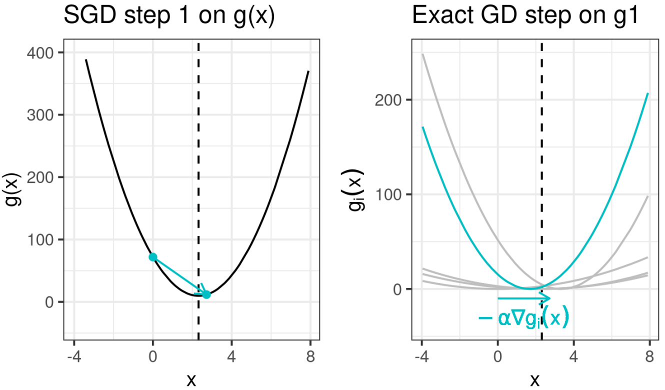
- **(Possibly) suboptimal direction:** Approximate gradient $\hat{\mathbf{d}} = \nabla_{\mathbf{x}} g_i(\mathbf{x})$ might point in suboptimal (possibly not even a descent!) direction
- **Unbiased estimate:** If J drawn i.i.d., approximate gradient $\hat{\mathbf{d}}$ is an unbiased estimate of gradient $\mathbf{d} = \nabla_{\mathbf{x}} g(\mathbf{x}) = \sum_{i=1}^{n} \nabla_{\mathbf{x}} g_i(\mathbf{x})$:

$$\begin{aligned}\mathbb{E}_i\left[\nabla_{\mathbf{x}} g_i(\mathbf{x})\right] &= \sum_{i=1}^{n} \nabla_{\mathbf{x}} g_i(\mathbf{x}) \cdot \mathbb{P}(i = i) \\ &= \sum_{i=1}^{n} \nabla_{\mathbf{x}} g_i(\mathbf{x}) \cdot \frac{1}{n} = \nabla_{\mathbf{x}} g(\mathbf{x}).\end{aligned}$$

Conclusion: SGD might perform single suboptimal moves, but moves in “right direction” **on average**.

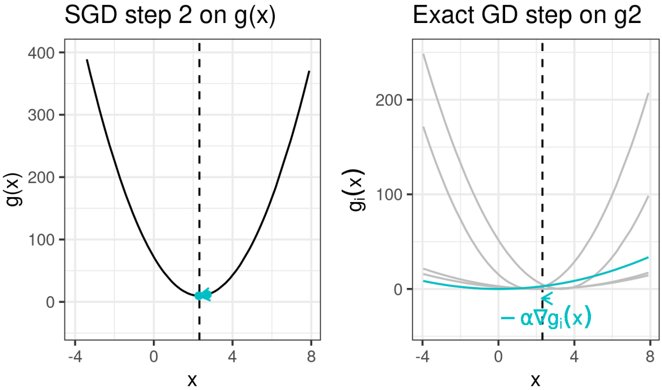
ERRATIC BEHAVIOR OF SGD

Example: $g(\mathbf{x}) = \sum_{i=1}^{5} g_i(\mathbf{x})$, g_i quadratic. Batch size $m = 1$.



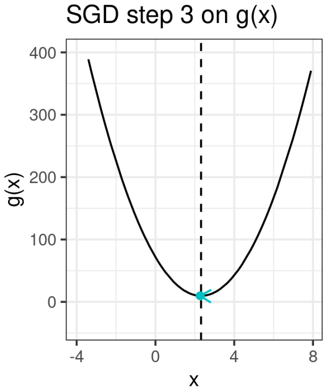
ERRATIC BEHAVIOR OF SGD

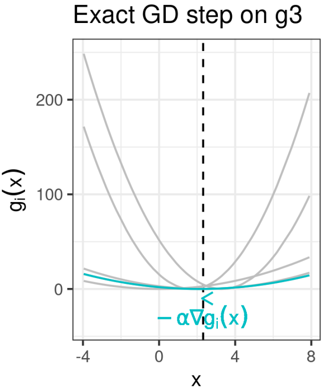
Example: $g(\mathbf{x}) = \sum_{i=1}^{5} g_i(\mathbf{x})$, g_i quadratic. Batch size $m = 1$.



ERRATIC BEHAVIOR OF SGD

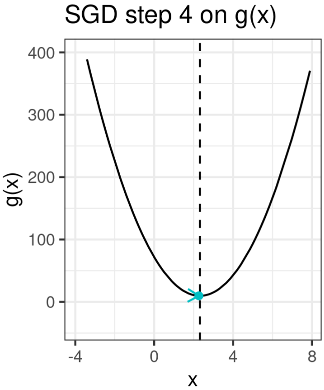
Example: $g(\mathbf{x}) = \sum_{i=1}^{5} g_i(\mathbf{x})$, g_i quadratic. Batch size $m = 1$.

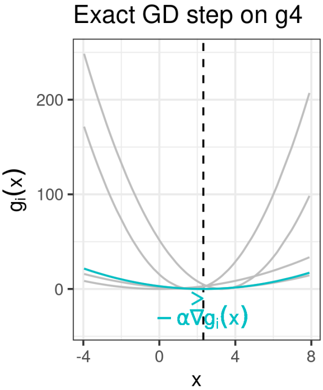




ERRATIC BEHAVIOR OF SGD

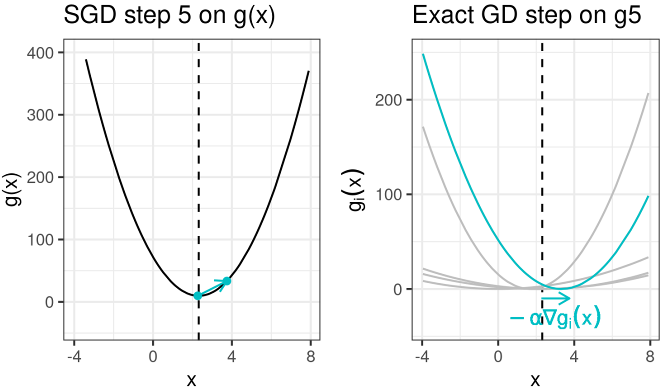
Example: $g(\mathbf{x}) = \sum_{i=1}^{5} g_i(\mathbf{x})$, g_i quadratic. Batch size $m = 1$.





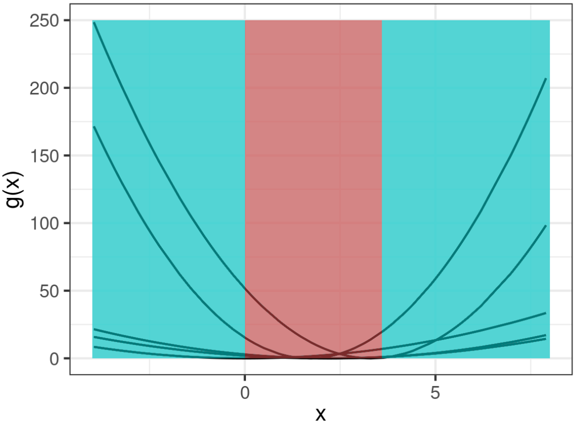
ERRATIC BEHAVIOR OF SGD

Example: $g(\mathbf{x}) = \sum_{i=1}^{5} g_i(\mathbf{x})$, g_i quadratic. Batch size $m = 1$.



In iteration 5, SGD performs a suboptimal move away from the minimum.

ERRATIC BEHAVIOR OF SGD



Blue area: Each $-\nabla g_i(\mathbf{x})$ points towards minimum.
Red area ("confusion area"): $-\nabla g_i(\mathbf{x})$ might point away from minimum and perform a suboptimal move.

ERRATIC BEHAVIOR OF SGD

- At location $\mathbf{x}$, "confusion" is captured by variance of gradients

$$\frac{1}{n}\sum_{i=1}^{n} \|\nabla_{\mathbf{x}} g_i(\mathbf{x}) - \nabla_{\mathbf{x}} g(\mathbf{x})\|^2$$

- If term is 0, next step goes in gradient direction (for each i)
- If term is small, next step *likely* goes in gradient direction
- If term is large, next step likely goes in direction different than gradient

CONVERGENCE OF SGD

As a consequence, SGD has worse convergence properties than GD.

But: Can be controlled via **increasing batches** or **reducing step size**.

The larger the batch size m

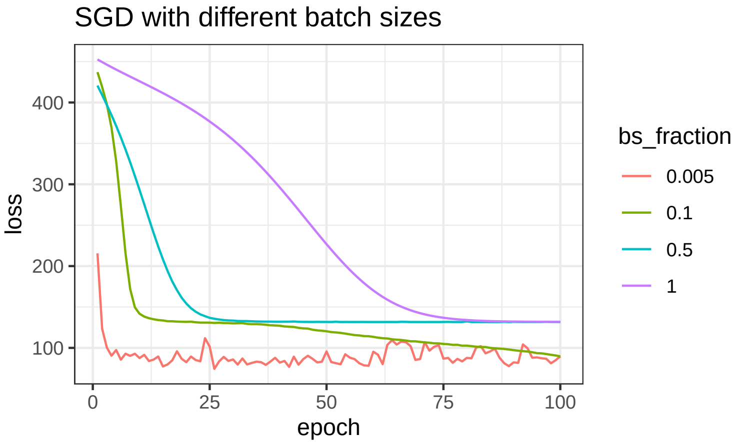
- the better the approximation to $\nabla_{\mathbf{x}} g(\mathbf{x})$
- the lower the variance
- the lower the risk of performing steps in the wrong direction

The smaller the step size α

- the smaller a step in a potentially wrong direction
- the lower the effect of high variance

As maximum batch size is usually limited by computational resources (memory), choosing the step size is crucial.

EFFECT OF BATCH SIZE



SGD for a NN with batch size $\in \{0.5\%, 10\%, 50\%\}$ of the training data.
The higher the batch size, the lower the variance.