A Practical Introduction to Control, Numerics and Machine Learning

Day 3

Summer School IFAC CPDE 2022

Workshop on Control of Systems Governed by Partial Differential Equations

Daniël Veldman

Chair in Dynamics, Control, and Numerics, Friedrich-Alexander-University Erlangen-Nürnberg

Contents

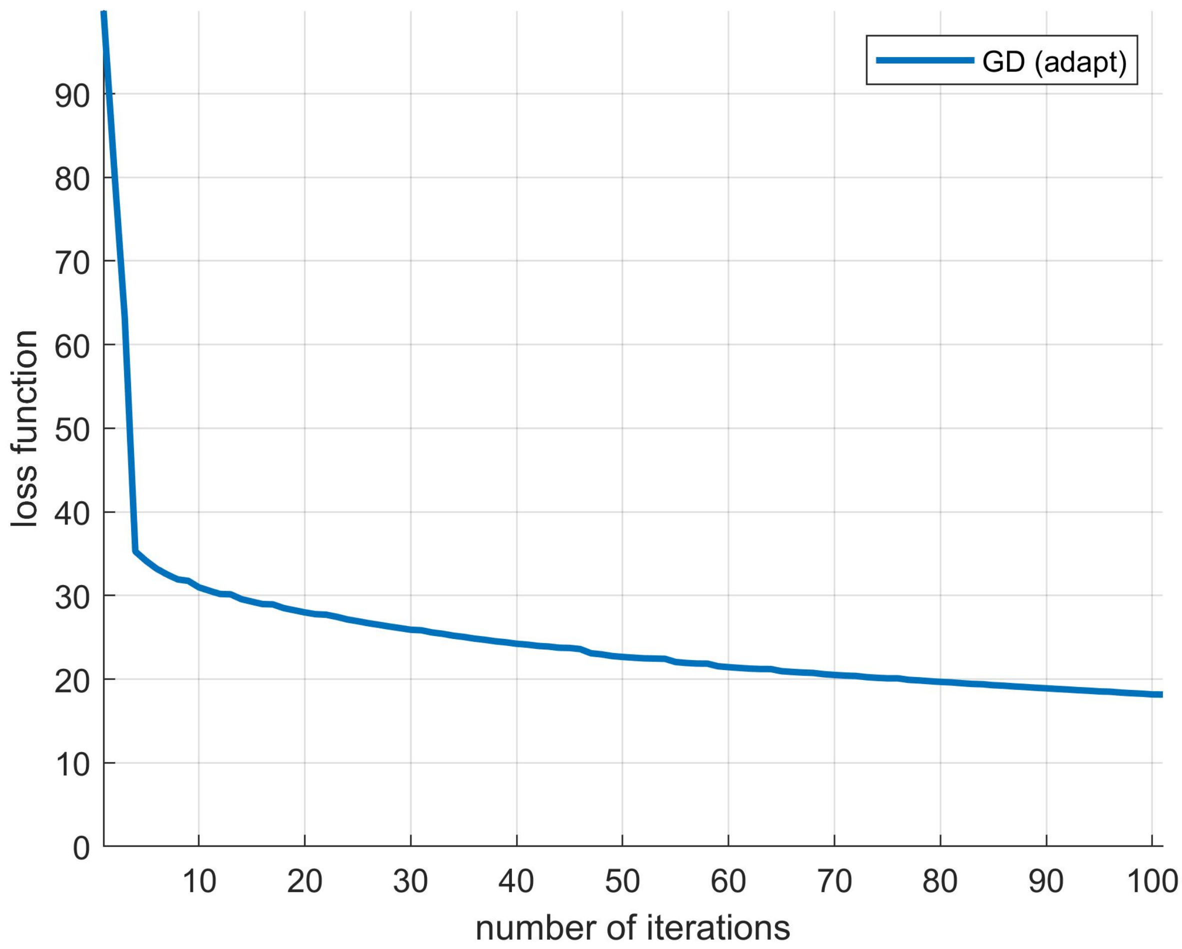
3.A Convergence analysis for gradient descent

Pseudo code for gradient descent with adaptive step size

- Choose an initial guess u_0
- Choose an initial step size β
- Compute $J_0 = J(u_0)$.
- for i = 1: `max_iters`
-     Compute $g_0 = \nabla J(u_0)$.
-     Set $J_1 = \infty$ and $\beta = 4\beta$.
-     while $J_1 > J_0$
-         Set $\beta = \beta/2$.
-         Set $u_1 = u_0 - \beta g_0$.
-         Compute $J_1 = J(u_1)$.
-     if convergence conditions are satisfied
-         Return u_1, J_1.
-     Set $u_0 = u_1$
-     Set $J_0 = J_1$

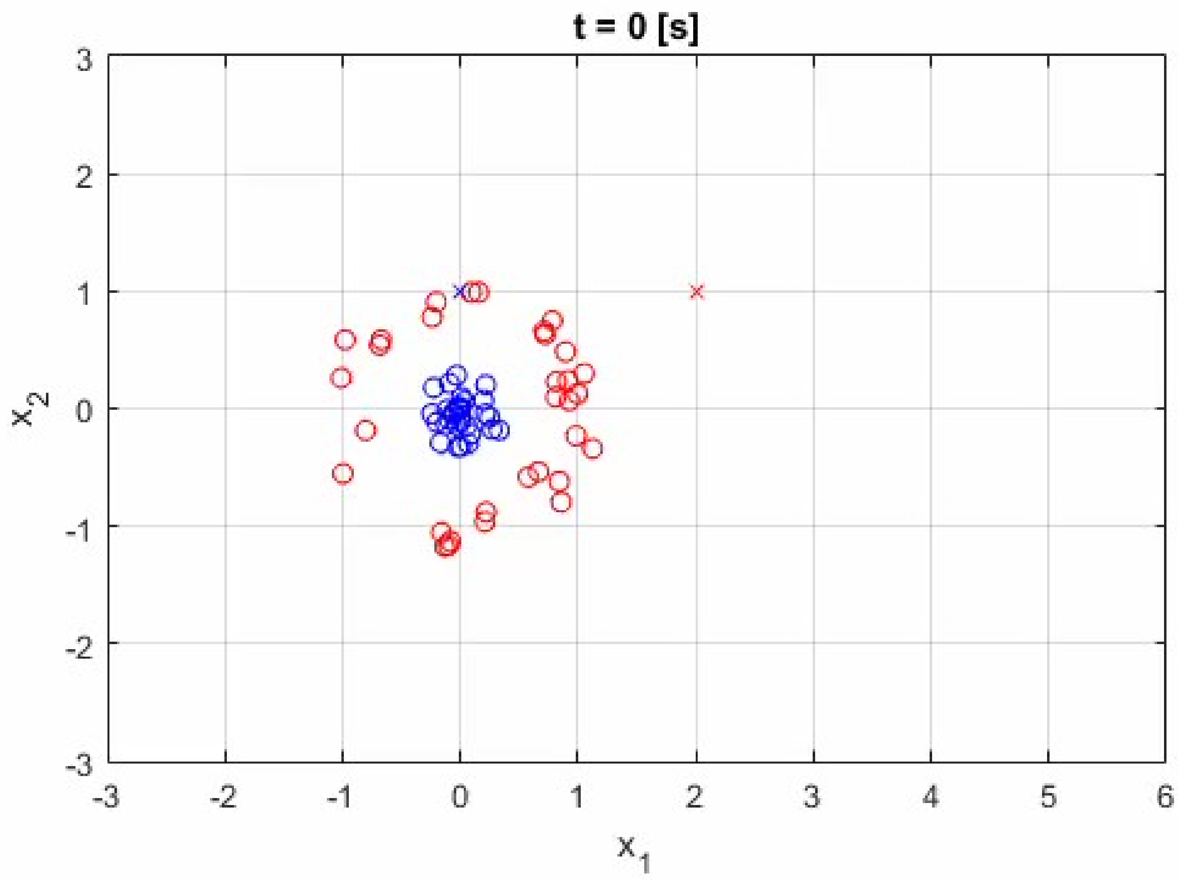
Example: 100 iterations of gradient descent with adaptive step size

Training of a ResNet with 100 hidden layers in $\mathbb{R}^2$ on 64 data points.



GD (adapt)	12.9 s

Example: 100 iterations of gradient descent with adaptive step size

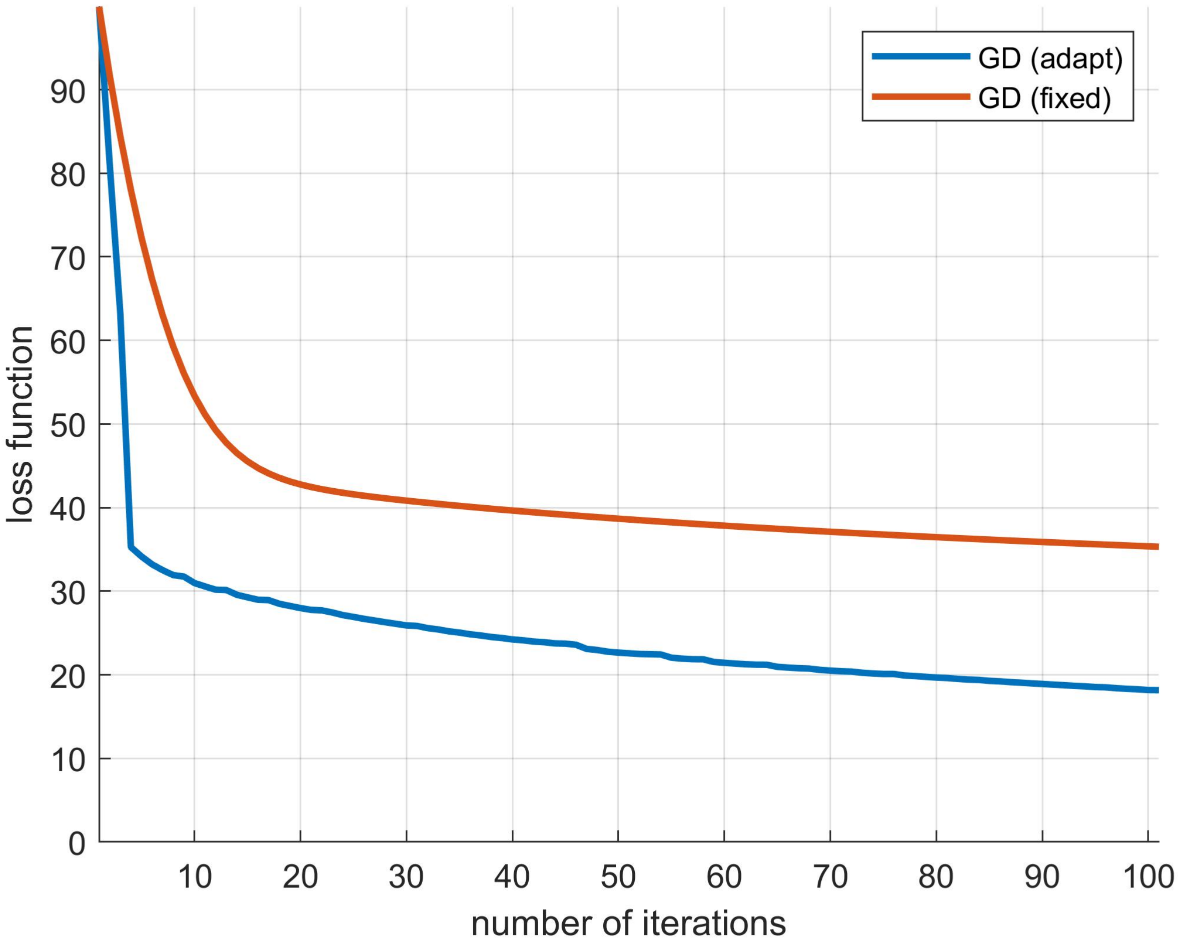


Pseudo code for gradient descent with adaptive step size

- Choose an initial guess u_0
- Choose a step size β
- Compute $J_0 = J(u_0)$.
- for i = 1: `max_iters`
- Compute $g_0 = \nabla J(u_0)$.
- Set $u_1 = u_0 - \beta g_0$.
- Compute $J_1 = J(u_1)$.
- if convergence conditions are satisfied
- Return u_1, J_1.
- Set $u_0 = u_1$
- Set $J_0 = J_1$

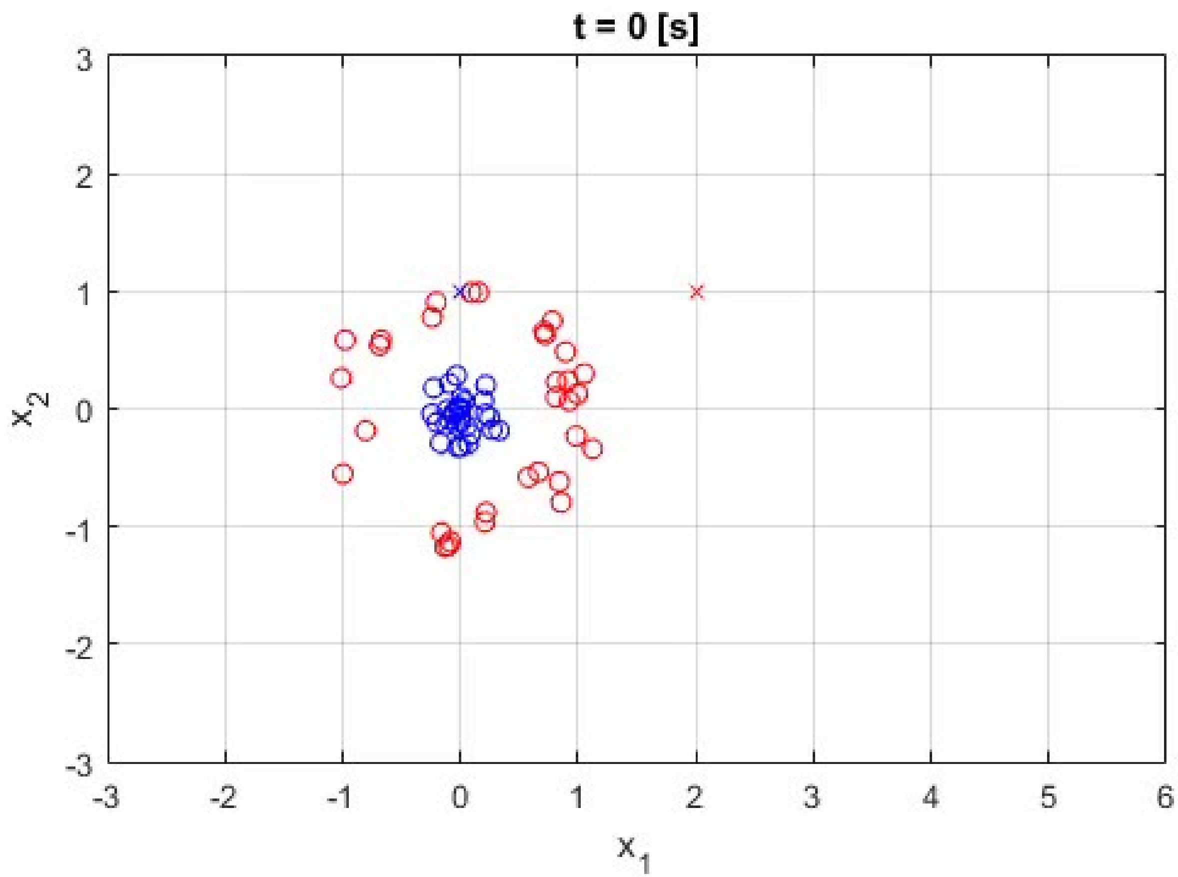
Example: 100 iterations of gradient descent with a fixed step size

Training of a ResNet with 100 hidden layers in $\mathbb{R}^2$ on 64 data points.



GD (adapt)	12.9 s
GD (fixed)	7.0 s

Example: 100 iterations of gradient descent with a fixed step size



Convergence analysis for gradient descent

We return to the more abstract optimization problem:

$$\min_{u \in \mathbb{R}^M} J(u).$$

Denote the minimizer by u^*.

For simplicity, we consider a gradient descent algorithm with a fixed step size β

$$u_{k+1} = u_k - \beta \nabla J(u_k).$$

Convergence analysis for gradient descent

We return to the more abstract optimization problem:

$$\min_{u\in\mathbb{R}^M} J(u).$$

Denote the minimizer by u^*.

For simplicity, we consider a gradient descent algorithm with a fixed step size β

$$u_{k+1} = u_k - \beta\nabla J(u_k).$$

Two assumptions:

- The functional J is α-convex, i.e.

$$J(\theta u + (1-\theta)v) \leq \theta J(u) + (1-\theta)J(v) - \frac{\alpha\theta(1-\theta)}{2}|u-v|^2, \qquad \theta\in[0,1].$$

- The gradient $\nabla J(u)$ is Lipschitz, i.e. there is an $L>0$ such that for all u and v

$$|\nabla J(u) - \nabla J(v)| \leq L|u-v|.$$

Theorem

$$|u_k - u^*|^2 \leq \left(1 - 2\alpha\beta + \beta^2L^2\right)^k |u_0 - u^*|^2$$

Observation 1

The functional J is α-convex:

$$J(\theta u + (1-\theta)v) \leq \theta J(u) + (1-\theta)J(v) - \frac{\alpha\theta(1-\theta)}{2}|u-v|^2.$$

Subtract expand the brackets on the LHS and subtract $J(v)$ on both sides:

$$J(v+\theta(u-v)) - J(v) \leq \theta J(u) - \theta J(v) - \frac{\alpha\theta(1-\theta)}{2}|u-v|^2.$$

Divide by θ and take the limit $\theta \to 0$:

$$\langle \nabla J(v), u-v\rangle = \lim_{\theta\to 0}\frac{J(v+\theta(u-v)) - J(v)}{\theta} \leq J(u) - J(v) - \frac{\alpha}{2}|u-v|^2.$$

We conclude

$$\langle \nabla J(v), u-v\rangle \leq J(u) - J(v) - \frac{\alpha}{2}|u-v|^2.$$

Observation 2

From the previous slide:

$$\langle \nabla J(v), u - v \rangle \leq J(u) - J(v) - \frac{\alpha}{2}|u - v|^2.$$

Because this holds for all u and v, we may interchange u and v to obtain:

$$\langle \nabla J(u), v - u \rangle \leq J(v) - J(u) - \frac{\alpha}{2}|v - u|^2.$$

Adding these two equations, we find

$$\langle \nabla J(v) - \nabla J(u), u - v \rangle \leq -\alpha|u - v|^2.$$

Proof

Theorem

$$|u_k - u^*|^2 \leq \left(1 - 2\alpha\beta + \beta^2 L^2\right)^k |u_0 - u^*|^2$$

$$\begin{aligned}
|u_{k+1} - u^*|^2 &= \langle u_{k+1} - u^*, u_{k+1} - u^* \rangle \\
&= \langle u_k - \beta \nabla J(u_k) - u^*, u_k - \beta \nabla J(u_k) - u^* \rangle \\
&= \langle u_k - u^*, u_k - u^* \rangle - 2\beta \langle \nabla J(u_k), u_k - u^* \rangle + \beta^2 \langle \nabla J(u_k), \nabla J(u_k) \rangle
\end{aligned}$$

Proof

Theorem

$$|u_k - u^*|^2 \leq \left(1 - 2\alpha\beta + \beta^2 L^2\right)^k |u_0 - u^*|^2$$

$$\begin{aligned}|u_{k+1} - u^*|^2 &= \langle u_{k+1} - u^*, u_{k+1} - u^*\rangle \\ &= \langle u_k - \beta\nabla J(u_k) - u^*, u_k - \beta\nabla J(u_k) - u^*\rangle \\ &= \langle u_k - u^*, u_k - u^*\rangle - 2\beta\langle\nabla J(u_k), u_k - u^*\rangle + \beta^2\langle\nabla J(u_k), \nabla J(u_k)\rangle\end{aligned}$$

Using that $\nabla J(u^*) = 0$ and Observation 2, we find

$$-\langle\nabla J(u_k), u_k - u^*\rangle = -\langle\nabla J(u_k) - \nabla J(u^*), u_k - u^*\rangle \leq -\alpha|u_k - u^*|^2.$$

Again using that $\nabla J(u^*) = 0$ and the Lipschitz continuity of $\nabla J(u)$, we also have that

$$\langle\nabla J(u_k), \nabla J(u_k)\rangle = |\nabla J(u_k) - \nabla J(u^*)|^2 \leq L^2|u_k - u^*|^2.$$

Proof

Theorem

$$|u_k - u^*|^2 \leq \left(1 - 2\alpha\beta + \beta^2 L^2\right)^k |u_0 - u^*|^2$$

$$\begin{aligned} |u_{k+1} - u^*|^2 &= \langle u_{k+1} - u^*, u_{k+1} - u^* \rangle \\ &= \langle u_k - \beta \nabla J(u_k) - u^*, u_k - \beta \nabla J(u_k) - u^* \rangle \\ &= \langle u_k - u^*, u_k - u^* \rangle - 2\beta \langle \nabla J(u_k), u_k - u^* \rangle + \beta^2 \langle \nabla J(u_k), \nabla J(u_k) \rangle \end{aligned}$$

Using that $\nabla J(u^*) = 0$ and Observation 2, we find

$$-\langle \nabla J(u_k), u_k - u^* \rangle = -\langle \nabla J(u_k) - \nabla J(u^*), u_k - u^* \rangle \leq -\alpha |u_k - u^*|^2.$$

Again using that $\nabla J(u^*) = 0$ and the Lipschitz continuity of $\nabla J(u)$, we also have that

$$\langle \nabla J(u_k), \nabla J(u_k) \rangle = |\nabla J(u_k) - \nabla J(u^*)|^2 \leq L^2 |u_k - u^*|^2.$$

Inserting these two results back into the original expression, we conclude

$$|u_{k+1} - u^*|^2 \leq \left(1 - 2\alpha\beta + \beta^2 L^2\right) |u_k - u^*|^2$$

The result now follows by induction over k.

3.B Stochastic gradient descent (SGD)

Stochastic gradient descent

For stochastic gradient descent, assume that the cost functional is of the form

$$J(u) = \sum_{i=1}^{I} J_i(u).$$

Typical in machine learning: each $J_i(u)$ corresponds to a training sample.

Stochastic gradient descent

For stochastic gradient descent, assume that the cost functional is of the form

$$J(u) = \sum_{i=1}^{I} J_i(u).$$

Typical in machine learning: each $J_i(u)$ corresponds to a training sample.

Therefore also

$$\nabla J(u) = \sum_{i=1}^{I} \nabla J_i(u).$$

Stochastic gradient descent

For stochastic gradient descent, assume that the cost functional is of the form

$$J(u) = \sum_{i=1}^{I} J_i(u).$$

Typical in machine learning: each $J_i(u)$ corresponds to a training sample.

Therefore also

$$\nabla J(u) = \sum_{i=1}^{I} \nabla J_i(u).$$

If all the $J_i(u)$ are similar,

$$\nabla J(u) \approx \tilde{\nabla} J(u) = I \nabla J_j(u),$$

for a randomly selected $j \in \{1, 2, \ldots, I\}$.

Stochastic gradient descent

For stochastic gradient descent, assume that the cost functional is of the form

$$J(u) = \sum_{i=1}^{I} J_i(u).$$

Typical in machine learning: each $J_i(u)$ corresponds to a training sample.

Therefore also

$$\nabla J(u) = \sum_{i=1}^{I} \nabla J_i(u).$$

If all the $J_i(u)$ are similar,

$$\nabla J(u) \approx \tilde{\nabla} J(u) = I \nabla J_j(u),$$

for a randomly selected $j \in \{1, 2, \ldots, I\}$.

Note that

$$\mathbb{E}[\tilde{\nabla} J(u)] = \sum_{i=1}^{I} I \nabla J_i(u) \mathbb{P}[j = i] = \sum_{i=1}^{I} \nabla J_i(u) = \nabla J(u),$$

because $\mathbb{P}[j = i] = 1/I$.

Pseudo code for stochastic gradient descent

In each iteration, take a step in the direction of the stochastic gradient:

$$u_{k+1} = u_k - \beta \tilde{\nabla} J(u_k)$$

- Choose an initial guess u_0
- Choose a step size β
- Compute $J_0 = J(u_0)$.
- for k = 1: `max_iters`
-     for j = 1: I
-         Select randomly an index $i \in \{1, 2, \ldots, I\}$
-         Compute $g_0 = I \nabla J_i(u_0)$.
-         Set $u_0 = u_0 - \beta g_0$.

Convergence analysis: assumptions

Three assumptions:

- The functional J is α-convex, i.e.

$$J(\theta u + (1-\theta)v) \leq \theta J(u) + (1-\theta)J(v) - \frac{\alpha\theta(1-\theta)}{2}|u-v|^2, \qquad \theta \in [0,1].$$

- The gradient $\nabla J(u)$ is Lipschitz, i.e. there is an $L > 0$ such that for all u and v

$$|\nabla J(u) - \nabla J(v)| \leq L|u-v|.$$

- The variance of the stochastic gradient is bounded, i.e. there is a σ such that for all u

$$\mathbb{E}[|\tilde{\nabla} J(u) - \nabla J(u)|^2] \leq \sigma^2.$$

Convergence analysis: proof (1/2)

In each iteration, take a step in the direction of the stochastic gradient:

$$u_{k+1} = u_k - \beta \tilde{\nabla} J(u_k)$$

It then follows that

$$\begin{aligned} |u_{k+1} - u^*|^2 &= \langle u_{k+1} - u^*, u_{k+1} - u^* \rangle \\ &= \langle u_k - \beta \tilde{\nabla} J(u_k) - u^*, u_k - \beta \tilde{\nabla} J(u_k) - u^* \rangle \\ &= \langle u_k - u^*, u_k - u^* \rangle - 2\beta \langle \tilde{\nabla} J(u_k), u_k - u^* \rangle + \beta^2 \langle \tilde{\nabla} J(u_k), \tilde{\nabla} J(u_k) \rangle \end{aligned}$$

Convergence analysis: proof (1/2)

In each iteration, take a step in the direction of the stochastic gradient:

$$u_{k+1} = u_k - \beta \tilde{\nabla} J(u_k)$$

It then follows that

$$\begin{aligned}
|u_{k+1} - u^*|^2 &= \langle u_{k+1} - u^*, u_{k+1} - u^* \rangle \\
&= \langle u_k - \beta \tilde{\nabla} J(u_k) - u^*, u_k - \beta \tilde{\nabla} J(u_k) - u^* \rangle \\
&= \langle u_k - u^*, u_k - u^* \rangle - 2\beta \langle \tilde{\nabla} J(u_k), u_k - u^* \rangle + \beta^2 \langle \tilde{\nabla} J(u_k), \tilde{\nabla} J(u_k) \rangle
\end{aligned}$$

Taking the expectation, using that $\mathbb{E}[\tilde{\nabla} J(u_k) \mid u_k] = \nabla J(u_k)$, it follows that

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = |u_k - u^*|^2 - 2\beta \langle \nabla J(u_k), u_k - u^* \rangle + \beta^2 \mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k].$$

Convergence analysis: proof (1/2)

In each iteration, take a step in the direction of the stochastic gradient:

$$u_{k+1} = u_k - \beta\tilde{\nabla} J(u_k)$$

It then follows that

$$\begin{aligned}|u_{k+1} - u^*|^2 &= \langle u_{k+1} - u^*, u_{k+1} - u^*\rangle \\ &= \langle u_k - \beta\tilde{\nabla} J(u_k) - u^*, u_k - \beta\tilde{\nabla} J(u_k) - u^*\rangle \\ &= \langle u_k - u^*, u_k - u^*\rangle - 2\beta\langle\tilde{\nabla} J(u_k), u_k - u^*\rangle + \beta^2\langle\tilde{\nabla} J(u_k), \tilde{\nabla} J(u_k)\rangle\end{aligned}$$

Taking the expectation, using that $\mathbb{E}[\tilde{\nabla} J(u_k) \mid u_k] = \nabla J(u_k)$, it follows that

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = |u_k - u^*|^2 - 2\beta\langle\nabla J(u_k), u_k - u^*\rangle + \beta^2\mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k].$$

For the second term on the RHS, Observation 2 shows that

$$-\langle\nabla J(u_k), u_k - u^*\rangle = -\langle\nabla J(u_k) - \nabla J(u^*), u_k - u^*\rangle \leq -\alpha|u_k - u^*|^2.$$

Therefore,

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = (1 - 2\alpha\beta)|u_k - u^*|^2 + \beta^2\mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k].$$

Convergence analysis: proof (2/2)

From the previous slide:

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = (1 - 2\alpha\beta)|u_k - u^*|^2 + \beta^2 \mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k].$$

Convergence analysis: proof (2/2)

From the previous slide:

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = (1 - 2\alpha\beta)|u_k - u^*|^2 + \beta^2 \mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k].$$

For the third term on the RHS, note that

$$|\tilde{\nabla} J(u_k)|^2 \leq |\tilde{\nabla} J(u_k) - \nabla J(u_k)|^2 + 2\langle \tilde{\nabla} J(u_k) - \nabla J(u_k), \nabla J(u_k)\rangle + |\nabla J(u_k)|^2.$$

Convergence analysis: proof (2/2)

From the previous slide:

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = (1 - 2\alpha\beta)|u_k - u^*|^2 + \beta^2 \mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k].$$

For the third term on the RHS, note that

$$|\tilde{\nabla} J(u_k)|^2 \leq |\tilde{\nabla} J(u_k) - \nabla J(u_k)|^2 + 2\langle \tilde{\nabla} J(u_k) - \nabla J(u_k), \nabla J(u_k)\rangle + |\nabla J(u_k)|^2.$$

Taking the expectation using that $\mathbb{E}[\tilde{\nabla} J(u_k) \mid u_k] = \nabla J(u_k)$, it follows that

$$\mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k] \leq \mathbb{E}[|\tilde{\nabla} J(u_k) - \nabla J(u_k)|^2 \mid u_k] + |\nabla J(u_k)|^2 \leq \sigma^2 + |\nabla J(u_k)|^2.$$

Convergence analysis: proof (2/2)

From the previous slide:

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = (1 - 2\alpha\beta)|u_k - u^*|^2 + \beta^2 \mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k].$$

For the third term on the RHS, note that

$$|\tilde{\nabla} J(u_k)|^2 \leq |\tilde{\nabla} J(u_k) - \nabla J(u_k)|^2 + 2\langle \tilde{\nabla} J(u_k) - \nabla J(u_k), \nabla J(u_k)\rangle + |\nabla J(u_k)|^2.$$

Taking the expectation using that $\mathbb{E}[\tilde{\nabla} J(u_k) \mid u_k] = \nabla J(u_k)$, it follows that

$$\mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k] \leq \mathbb{E}[|\tilde{\nabla} J(u_k) - \nabla J(u_k)|^2 \mid u_k] + |\nabla J(u_k)|^2 \leq \sigma^2 + |\nabla J(u_k)|^2.$$

Again using that $\nabla J(u^*) = 0$ and the Lipschitz continuity of $\nabla J(u)$, we also have that

$$|\nabla J(u_k)|^2 = \langle \nabla J(u_k), \nabla J(u_k)\rangle = |\nabla J(u_k) - \nabla J(u^*)|^2 \leq L^2 |u_k - u^*|^2.$$

Convergence analysis: proof (2/2)

From the previous slide:

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = (1 - 2\alpha\beta)|u_k - u^*|^2 + \beta^2 \mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k].$$

For the third term on the RHS, note that

$$|\tilde{\nabla} J(u_k)|^2 \leq |\tilde{\nabla} J(u_k) - \nabla J(u_k)|^2 + 2\langle \tilde{\nabla} J(u_k) - \nabla J(u_k), \nabla J(u_k)\rangle + |\nabla J(u_k)|^2.$$

Taking the expectation using that $\mathbb{E}[\tilde{\nabla} J(u_k) \mid u_k] = \nabla J(u_k)$, it follows that

$$\mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k] \leq \mathbb{E}[|\tilde{\nabla} J(u_k) - \nabla J(u_k)|^2 \mid u_k] + |\nabla J(u_k)|^2 \leq \sigma^2 + |\nabla J(u_k)|^2.$$

Again using that $\nabla J(u^*) = 0$ and the Lipschitz continuity of $\nabla J(u)$, we also have that

$$|\nabla J(u_k)|^2 = \langle \nabla J(u_k), \nabla J(u_k)\rangle = |\nabla J(u_k) - \nabla J(u^*)|^2 \leq L^2 |u_k - u^*|^2.$$

Inserting the resulting estimate for the third term, it follows that

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = (1 - 2\alpha\beta + \beta^2 L^2)|u_k - u^*|^2 + \beta^2\sigma^2.$$

Convergence analysis: proof (2/2)

From the previous slide:

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = (1 - 2\alpha\beta)|u_k - u^*|^2 + \beta^2 \mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k].$$

For the third term on the RHS, note that

$$|\tilde{\nabla} J(u_k)|^2 \leq |\tilde{\nabla} J(u_k) - \nabla J(u_k)|^2 + 2\langle \tilde{\nabla} J(u_k) - \nabla J(u_k), \nabla J(u_k) \rangle + |\nabla J(u_k)|^2.$$

Taking the expectation using that $\mathbb{E}[\tilde{\nabla} J(u_k) \mid u_k] = \nabla J(u_k)$, it follows that

$$\mathbb{E}[|\tilde{\nabla} J(u_k)|^2 \mid u_k] \leq \mathbb{E}[|\tilde{\nabla} J(u_k) - \nabla J(u_k)|^2 \mid u_k] + |\nabla J(u_k)|^2 \leq \sigma^2 + |\nabla J(u_k)|^2.$$

Again using that $\nabla J(u^*) = 0$ and the Lipschitz continuity of $\nabla J(u)$, we also have that

$$|\nabla J(u_k)|^2 = \langle \nabla J(u_k), \nabla J(u_k) \rangle = |\nabla J(u_k) - \nabla J(u^*)|^2 \leq L^2 |u_k - u^*|^2.$$

Inserting the resulting estimate for the third term, it follows that

$$\mathbb{E}[|u_{k+1} - u^*|^2 \mid u_k] = (1 - 2\alpha\beta + \beta^2 L^2)|u_k - u^*|^2 + \beta^2 \sigma^2.$$

Convergence of SGD

If β is such that $|1 - 2\alpha\beta + \beta^2 L^2| < 1$, then

$$\mathbb{E}[|u_k - u^*|^2] \leq |1 - 2\alpha\beta + \beta^2 L^2|^k |u_0 - u^*|^2 + \beta \frac{\sigma^2}{2\alpha - \beta L^2}.$$

Convergence of SGD

$$\mathbb{E}[|u_k - u^*|^2] \le |1 - 2\alpha\beta + \beta^2 L^2|^k |u_0 - u^*|^2 + \beta \frac{\sigma^2}{2\alpha - \beta L^2}.$$

Observe:

- the variance $\mathbb{E}[|\tilde{\nabla} J(u) - \nabla J(u)|^2] \le \sigma^2$ leads to an offset that does not converge to zero for $k \to \infty$.
  But the offset can be reduced by choosing the step size β smaller.
- The convergence rate $1 - 2\alpha\beta + \beta^2 L^2$ is the same as for gradient descent, but the cost for one iteration is reduced by a factor $1/I$.
- One epoch is defined as I iterations SGD.
  $\Rightarrow$ The computational cost for one epoch of SGD is approximately the same as one iteration of GD.
  Convergence rate per epoch is

$$|1 - 2\alpha\beta + \beta^2 L^2|^I.$$

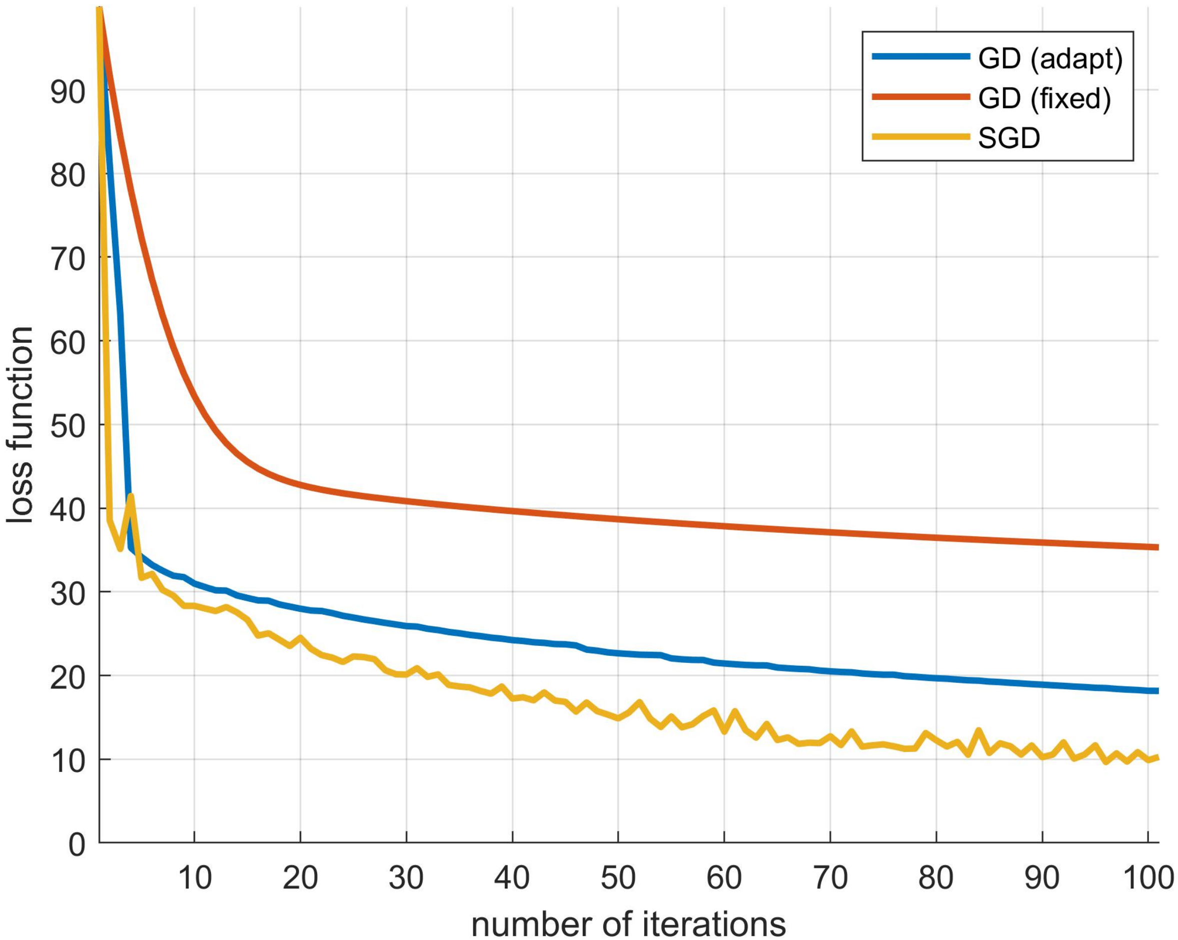
When the offset is sufficiently small,
the computational efficiency of SGD is much higher than the one of GD.

Pseudo code for stochastic gradient descent

- Choose an initial guess u_0
- Choose a step size β
- Compute $J_0 = J(u_0)$.
- for k = 1: `max_iters`
-     for j = 1: I
-         Select randomly an index $i \in \{1, 2, \dots, I\}$
-         Compute $g_0 = I\nabla J_i(u_0)$.
-         Set $u_0 = u_0 - \beta g_0$.

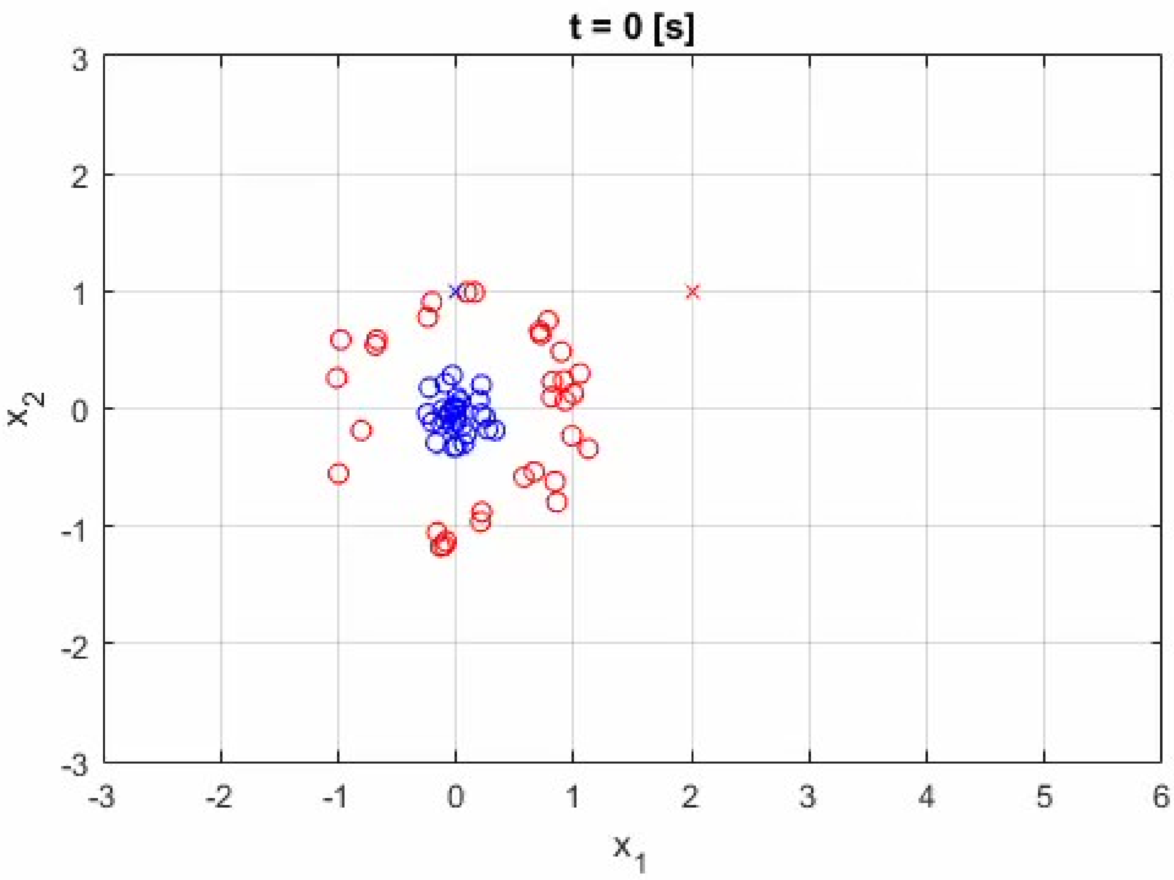
Example: 100 epochs of stochastic gradient descent

Training of a ResNet with 100 hidden layers in $\mathbb{R}^2$ on 64 data points.



GD (adapt)	12.9 s
GD (fixed)	7.0 s
SGD	11.1 s

Example: 100 epochs of stochastic gradient descent



Mini-batch methods

Setting:

$$J(u) = \sum_{i=1}^{I} J_i(u), \qquad \Rightarrow \qquad \nabla J(u) = \sum_{i=1}^{I} \nabla J_i(u).$$

Now define the stochastic gradient as the average of b randomly chosen gradients

$$\tilde{\nabla} J(u) = \frac{I}{b} \sum_{j \in \mathcal{B}} \nabla J_j(u),$$

where $\mathcal{B}$ is a randomly selected subset of $\{1, 2, \ldots, I\}$ of size b.

Mini-batch methods

Setting:

$$J(u) = \sum_{i=1}^{I} J_i(u), \qquad \Rightarrow \qquad \nabla J(u) = \sum_{i=1}^{I} \nabla J_i(u).$$

Now define the stochastic gradient as the average of b randomly chosen gradients

$$\tilde{\nabla} J(u) = \frac{I}{b} \sum_{j \in \mathcal{B}} \nabla J_j(u),$$

where $\mathcal{B}$ is a randomly selected subset of $\{1, 2, \dots, I\}$ of size b.

Again, it holds that

$$\mathbb{E}[\tilde{\nabla} J(u)] = \nabla J(u),$$

so it still makes sense to do updates as

$$u_{k+1} = u_k - \beta \tilde{\nabla} J(u_k).$$

Mini-batch methods: advantages and disadvantages

- **Disadvatange:**
  The computational cost is now b times higher than for SGD.
  $\Rightarrow$ An epoch is now consists of I/b iterations.

  Because $\mathbb{E}[\tilde{\nabla} J(u)] = \nabla J(u)$, the convergence rate is $|1 - 2\alpha\beta + \beta^2 L^2|$ per iteration.
  The convergence rate per epoch is thus

  $$|1 - 2\alpha\beta + \beta^2 L^2|^{I/b}.$$

  The convergence rate is lower than for SGD!

Mini-batch methods: advantages and disadvantages

- **Disadvatange:**
  The computational cost is now b times higher than for SGD.
  $\Rightarrow$ An epoch is now consists of I/b iterations.

  Because $\mathbb{E}[\tilde{\nabla} J(u)] = \nabla J(u)$, the convergence rate is $|1 - 2\alpha\beta + \beta^2 L^2|$ per iteration. The convergence rate per epoch is thus

  $$|1 - 2\alpha\beta + \beta^2 L^2|^{I/b}.$$

  The convergence rate is lower than for SGD!

- **Advantage:**
  The variance is reduced by a factor $1/b$, i.e. it now holds that

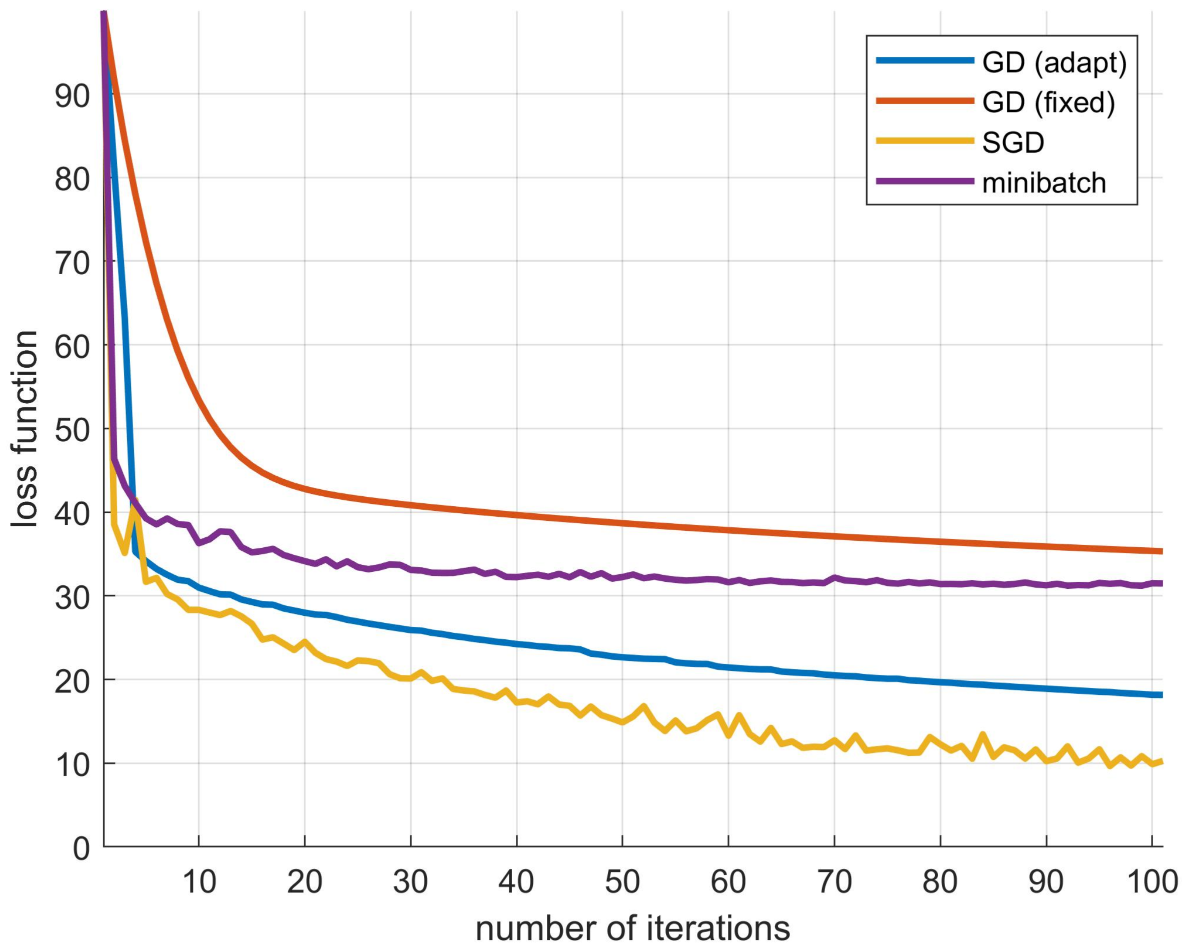
  $$\mathbb{E}[|\tilde{\nabla} J(u) - \nabla J(u)|^2] \leq \frac{\sigma_{\mathrm{SGD}}^2}{b}.$$

Pseudo code for stochastic gradient descent with batch size b

- Choose an initial guess u_0
- Choose a step size β and batch size b
- Compute $J_0 = J(u_0)$.
- for k = 1: `max_iters`
- for j = 1: I/b
- Select a random subset $\mathcal{B}$ of $i \in \{1, 2, \ldots, I\}$ of size b.
- Compute $g_0 = I/b \sum_{i \in \mathcal{B}} \nabla J_i(u_0)$.
- Set $u_0 = u_0 - \beta g_0$.

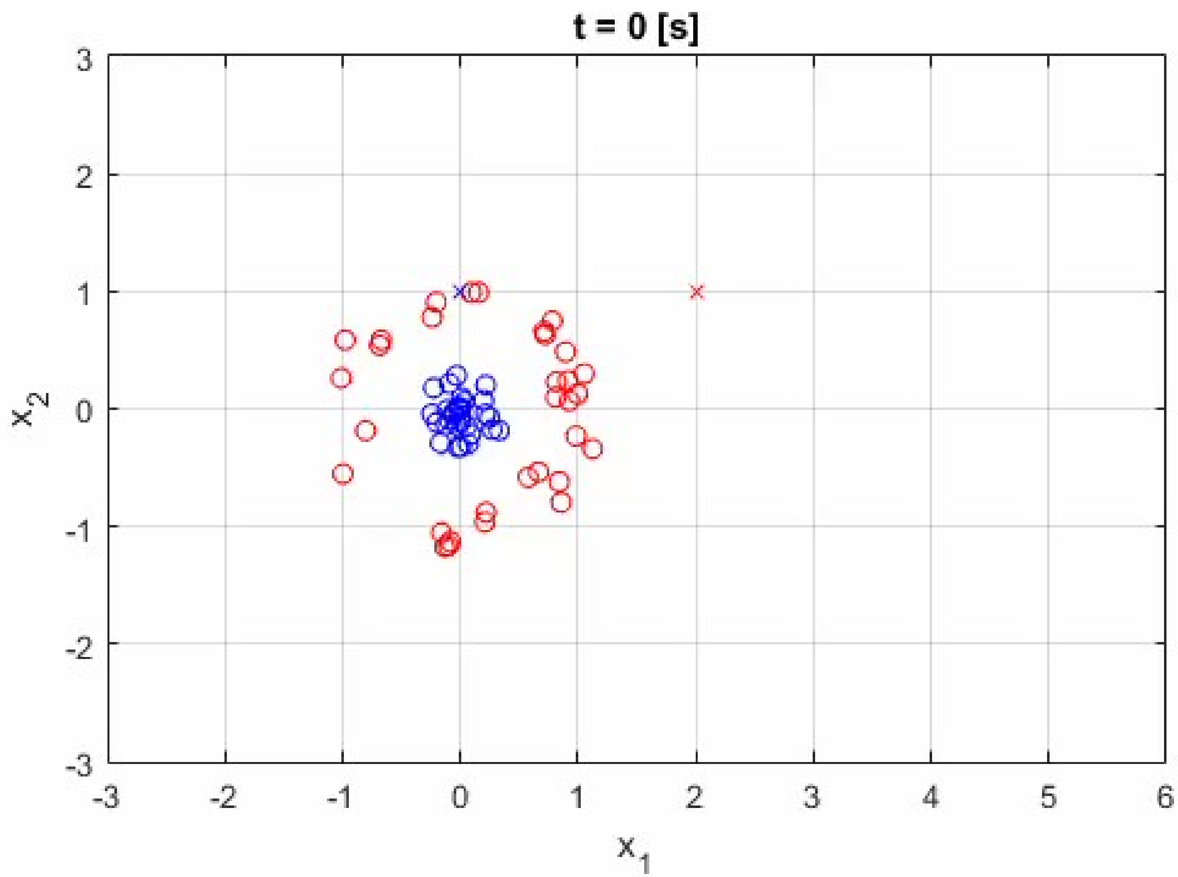
Example: 100 epochs of stochastic gradient descent with batch size 4

Training of a ResNet with 100 hidden layers in $\mathbb{R}^2$ on 64 data points.



GD (adapt)	12.9 s
GD (fixed)	7.0 s
SGD	11.1 s
minibatch	10.1 s

Example: 100 epochs of stochastic gradient descent with batch size 4



3.C SGD with momentum and ADAM

SGD with momentum

Problem in SGD: the gradient changes rapidly in each iteration.
This leads to a highly oscillatory trajectory.

Idea: to reduce oscillations, take an average over the previously computed gradients.
However, we should also ‘forget’ gradients that have been computed too long ago.

SGD with momentum

Problem in SGD: the gradient changes rapidly in each iteration.
This leads to a highly oscillatory trajectory.

Idea: to reduce oscillations, take an average over the previously computed gradients.
However, we should also 'forget' gradients that have been computed too long ago.

So now do updates as

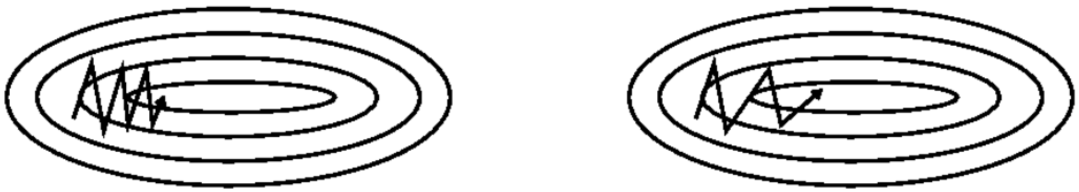
$$u_{k+1} = u_k - \beta v_k$$

where

$$\begin{aligned} v_k &= \tilde{\nabla} J(u_k) + \gamma \tilde{\nabla} J(u_{k-1}) + \gamma^2 \tilde{\nabla} J(u_{k-2}) + \ldots + \gamma^k \tilde{\nabla} J(u_0) \\ &= \tilde{\nabla} J(u_k) + \gamma v_{k-1}, \end{aligned}$$

for some $\gamma \in (0,1)$. Typically, $\gamma = 0.9$ or $\gamma = 0.99$.

Interpretation



(a) SGD without momentum (b) SGD with momentum

- Gradient descent is a man walking down a hill. He follows the steepest path downwards; his progress is slow, but steady.
- Momentum is a heavy ball rolling down the same hill. The added inertia acts both as a smoother and an accelerator, dampening oscillations and causing us to barrel through narrow valleys, small humps and local minima.

Pseudo code for gradient descent with momentum

- Choose an initial guess u_0 and set $v_0 = 0$.
- Choose a step size $\beta > 0$ and momentum parameter $\gamma \in (0, 1)$.
- for k = 1: `max_iters`
-     for j = 1: I
-         Select randomly an index $i \in \{1, 2, \dots, I\}$
-         Compute $g_0 = I\nabla J_i(u_0)$.
-         Set $v_0 = g_0 + \gamma v_0$
-         Set $u_0 = u_0 - \beta v_0$.

Pseudo code for gradient descent with momentum (alternative)

The iterations

$$u_{k+1} = u_k - \beta v_k, \qquad v_k = \tilde{\nabla} J(u_k) + \gamma v_{k-1}$$

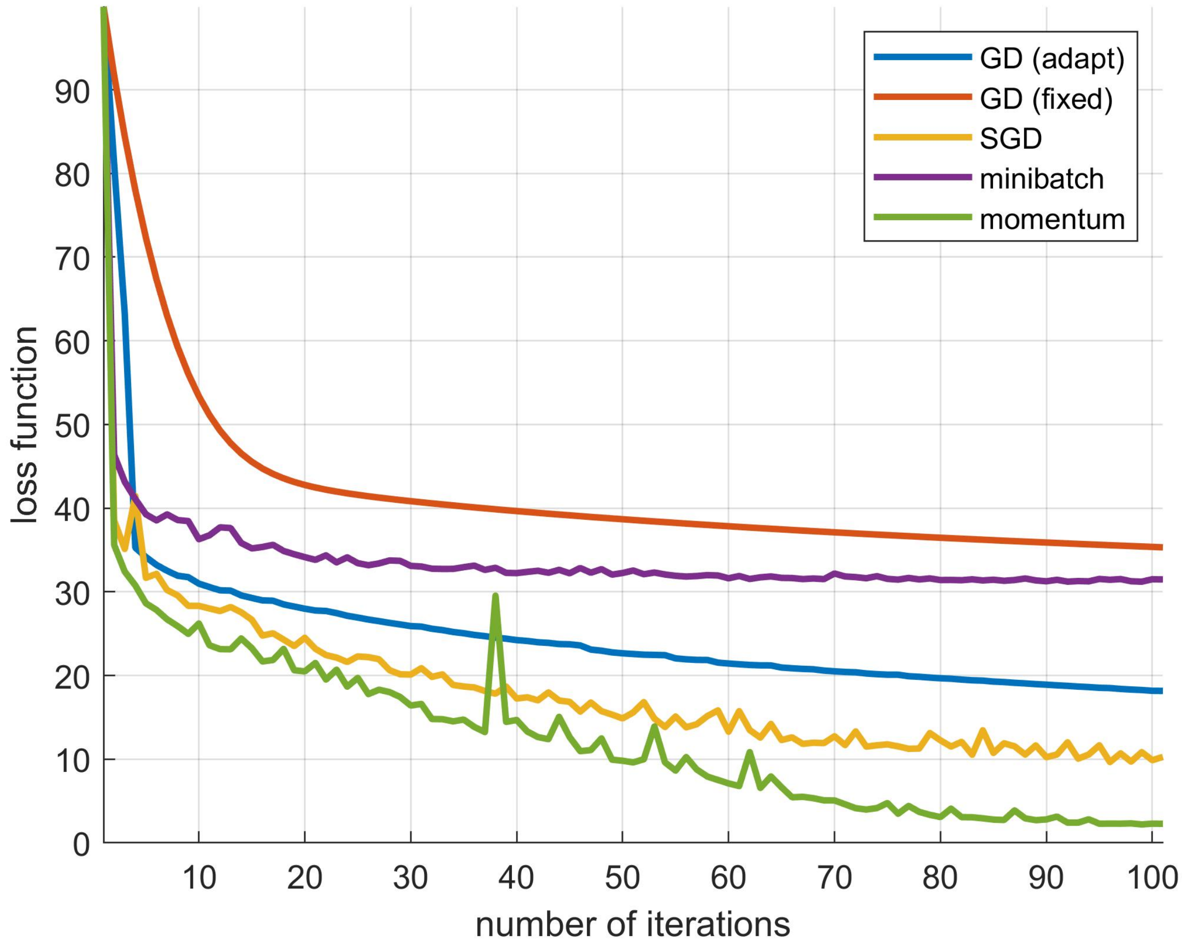
can be rewritten as

$$u_{k+1} = u_k - \beta \tilde{\nabla} J(u_k) + \gamma(u_k - u_{k-1}).$$

- Choose an initial guess $u_0 = 0$ and set $u_1 = u_0$.
- Choose a step size $\beta > 0$ and momentum parameter $\gamma \in (0, 1)$.
- Select randomly an index $i \in \{1, 2, \dots, I\}$.
- for k = 1: `max_iters`
-     for j = 1: I
-         Select randomly an index $i \in \{1, 2, \dots, I\}$
-         Compute $g_1 = I \nabla J_i(u_1)$.
-         Set $u_2 = u_1 - \beta g_1 + \gamma(u_1 - u_0)$.
-         Set $u_0 = u_1$.
-         Set $u_1 = u_2$.

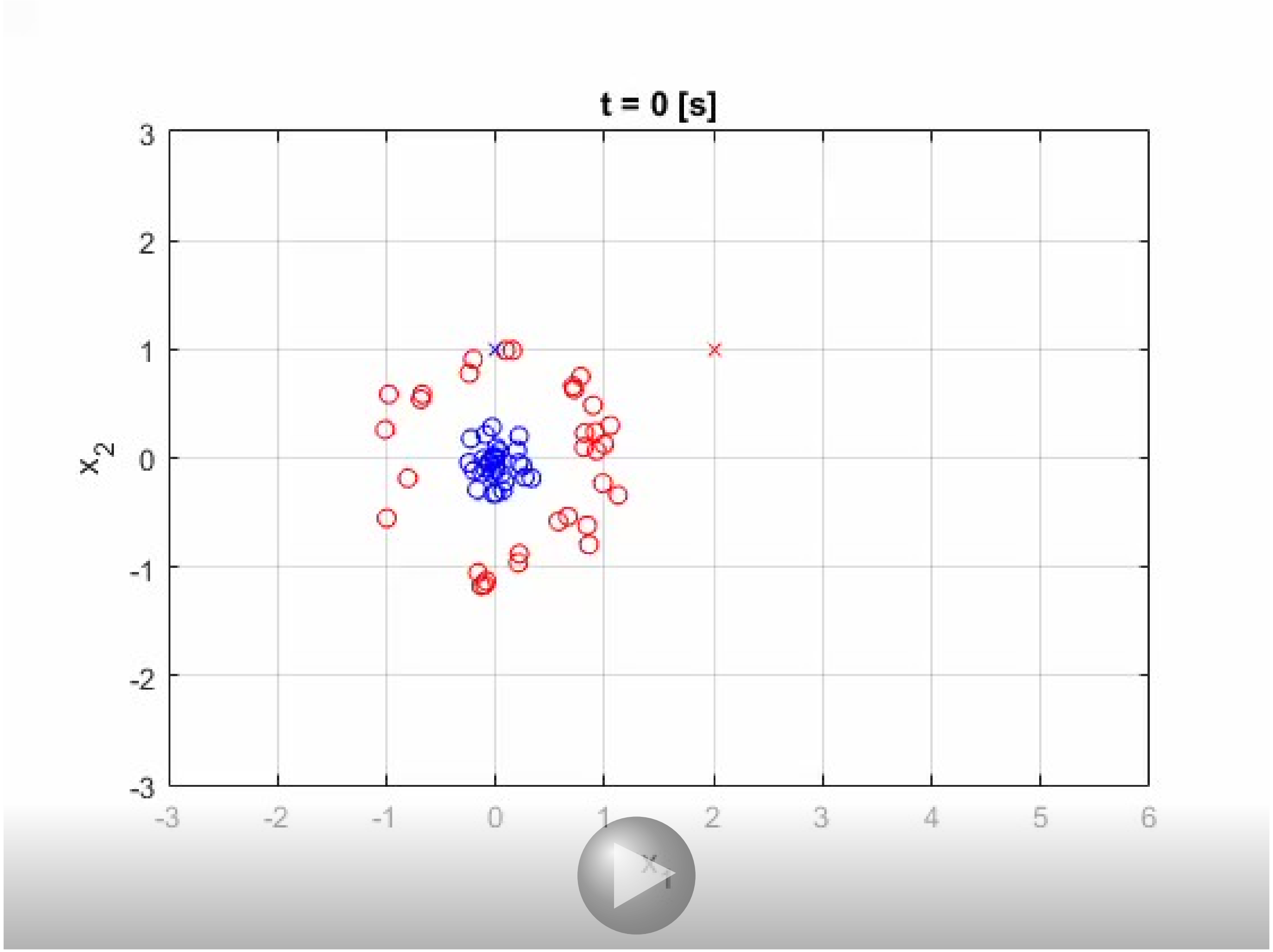
Example: 100 epochs of stochastic gradient descent with momentum

Training of a ResNet with 100 hidden layers in $\mathbb{R}^2$ on 64 data points.



GD (adapt)	12.9 s
GD (fixed)	7.0 s
SGD	11.1 s
minibatch	10.1 s
momentum	11.0 s

Example: 100 epochs of stochastic gradient descent with momentum



ADAptive Moment estimation (ADAM)

Idea: estimate the first and second moment of the gradient, i.e. estimations $\tilde{m}_k$ and $\tilde{v}_k$ such that

$$\tilde{m}_k \approx \mathbb{E}[\tilde{\nabla} J(u_k)], \qquad \tilde{v}_k \approx \mathbb{E}[\tilde{\nabla} J(u_k) \odot \tilde{\nabla} J(u_k)],$$

where $\odot$ denotes the component-wise product of vectors.

ADAptive Moment estimation (ADAM)

Idea: estimate the first and second moment of the gradient,
i.e. estimations $\tilde{m}_k$ and $\tilde{v}_k$ such that

$$\tilde{m}_k \approx \mathbb{E}[\tilde{\nabla} J(u_k)], \qquad \tilde{v}_k \approx \mathbb{E}[\tilde{\nabla} J(u_k) \odot \tilde{\nabla} J(u_k)],$$

where $\odot$ denotes the component-wise product of vectors.

Then the update is computed as

$$u_{k+1} = u_k - \beta \frac{\tilde{m}_k}{\sqrt{\tilde{v}_k} + \varepsilon},$$

for some (small) $\varepsilon > 0$. Note that

- the square root in $\sqrt{\tilde{v}_k}$ is computed component-wise,
- the division in $\tilde{m}_k / (\sqrt{\tilde{v}_k} + \varepsilon)$ is computed component-wise.

ADAptive Moment estimation (ADAM)

Idea: estimate the first and second moment of the gradient, i.e. estimations $\tilde{m}_k$ and $\tilde{v}_k$ such that

$$\tilde{m}_k \approx \mathbb{E}[\tilde{\nabla} J(u_k)], \qquad \tilde{v}_k \approx \mathbb{E}[\tilde{\nabla} J(u_k) \odot \tilde{\nabla} J(u_k)],$$

where $\odot$ denotes the component-wise product of vectors.

Then the update is computed as

$$u_{k+1} = u_k - \beta \frac{\tilde{m}_k}{\sqrt{\tilde{v}_k} + \varepsilon},$$

for some (small) $\varepsilon > 0$. Note that

- the square root in $\sqrt{\tilde{v}_k}$ is computed component-wise,
- the division in $\tilde{m}_k / (\sqrt{\tilde{v}_k} + \varepsilon)$ is computed component-wise.

Observe that if $\tilde{m}_k = \nabla J(u_k)$, $\tilde{v}_k = \nabla J(u_k) \odot \nabla J(u_k)$, and $\varepsilon = 0$, the update reduces to $u_{k+1} = u_k - \beta \mathrm{sign}(\nabla J(u_k))$.
Note that $-\mathrm{sign}(\nabla J(u_k))$ is a descent direction because

$$\langle \nabla J(u_k), -\mathrm{sign}(\nabla J(u_k)) \rangle = -|\nabla J(u_k)|_1 \leq 0.$$

Estimation of the first and second moments

Define m_k as

$$
\begin{aligned}
m_k &= (1-\beta_1)\tilde{\nabla} J(u_k) + \beta_1(1-\beta_1)\tilde{\nabla} J(u_{k-1}) + \ldots + \beta_1^k(1-\beta_1)\tilde{\nabla} J(u_0) \\
&= (1-\beta_1)\tilde{\nabla} J(u_k) + \beta_1 m_{k-1}.
\end{aligned}
$$

Estimation of the first and second moments

Define m_k as

$$\begin{aligned} m_k &= (1-\beta_1)\tilde{\nabla} J(u_k) + \beta_1(1-\beta_1)\tilde{\nabla} J(u_{k-1}) + \ldots + \beta_1^k(1-\beta_1)\tilde{\nabla} J(u_0) \\ &= (1-\beta_1)\tilde{\nabla} J(u_k) + \beta_1 m_{k-1}. \end{aligned}$$

Note that

$$\mathbb{E}[|\tilde{\nabla} J(u_k) - \tilde{\nabla} J(u_{k-1})|] = O(\beta).$$

Therefore,

$$\begin{aligned} \mathbb{E}[m_k] = \mathbb{E}\left[(1-\beta_1)\sum_{j=0}^{k} \beta_1^{k-j}\tilde{\nabla} J(u_j)\right] &= \mathbb{E}\left[(1-\beta_1)\sum_{j=0}^{k} \beta_1^{k-j}\tilde{\nabla} J(u_k) + O(\beta)\right] \\ = \mathbb{E}[\tilde{\nabla} J(u_k)](1-\beta_1)\sum_{j=0}^{k} \beta_1^{k-j} + O(\beta) &= \mathbb{E}[\tilde{\nabla} J(u_k)](1-\beta_1^{k+1}) + O(\beta). \end{aligned}$$

Estimation of the first and second moments

Define m_k as

$$\begin{aligned} m_k &= (1-\beta_1)\tilde{\nabla} J(u_k) + \beta_1(1-\beta_1)\tilde{\nabla} J(u_{k-1}) + \ldots + \beta_1^k(1-\beta_1)\tilde{\nabla} J(u_0) \\ &= (1-\beta_1)\tilde{\nabla} J(u_k) + \beta_1 m_{k-1}. \end{aligned}$$

Note that

$$\mathbb{E}[|\tilde{\nabla} J(u_k) - \tilde{\nabla} J(u_{k-1})|] = O(\beta).$$

Therefore,

$$\begin{aligned} \mathbb{E}[m_k] &= \mathbb{E}\left[(1-\beta_1)\sum_{j=0}^{k}\beta_1^{k-j}\tilde{\nabla} J(u_j)\right] = \mathbb{E}\left[(1-\beta_1)\sum_{j=0}^{k}\beta_1^{k-j}\tilde{\nabla} J(u_k) + O(\beta)\right] \\ &= \mathbb{E}[\tilde{\nabla} J(u_k)](1-\beta_1)\sum_{j=0}^{k}\beta_1^{k-j} + O(\beta) = \mathbb{E}[\tilde{\nabla} J(u_k)](1-\beta_1^{k+1}) + O(\beta). \end{aligned}$$

Practical implementation:

$$m_k = (1-\beta_1)\tilde{\nabla} J(u_k) + \beta_1 m_{k-1}, \qquad \tilde{m}_k = \frac{m_k}{1-\beta_1^{k+1}}.$$

And similarly for the second order moments:

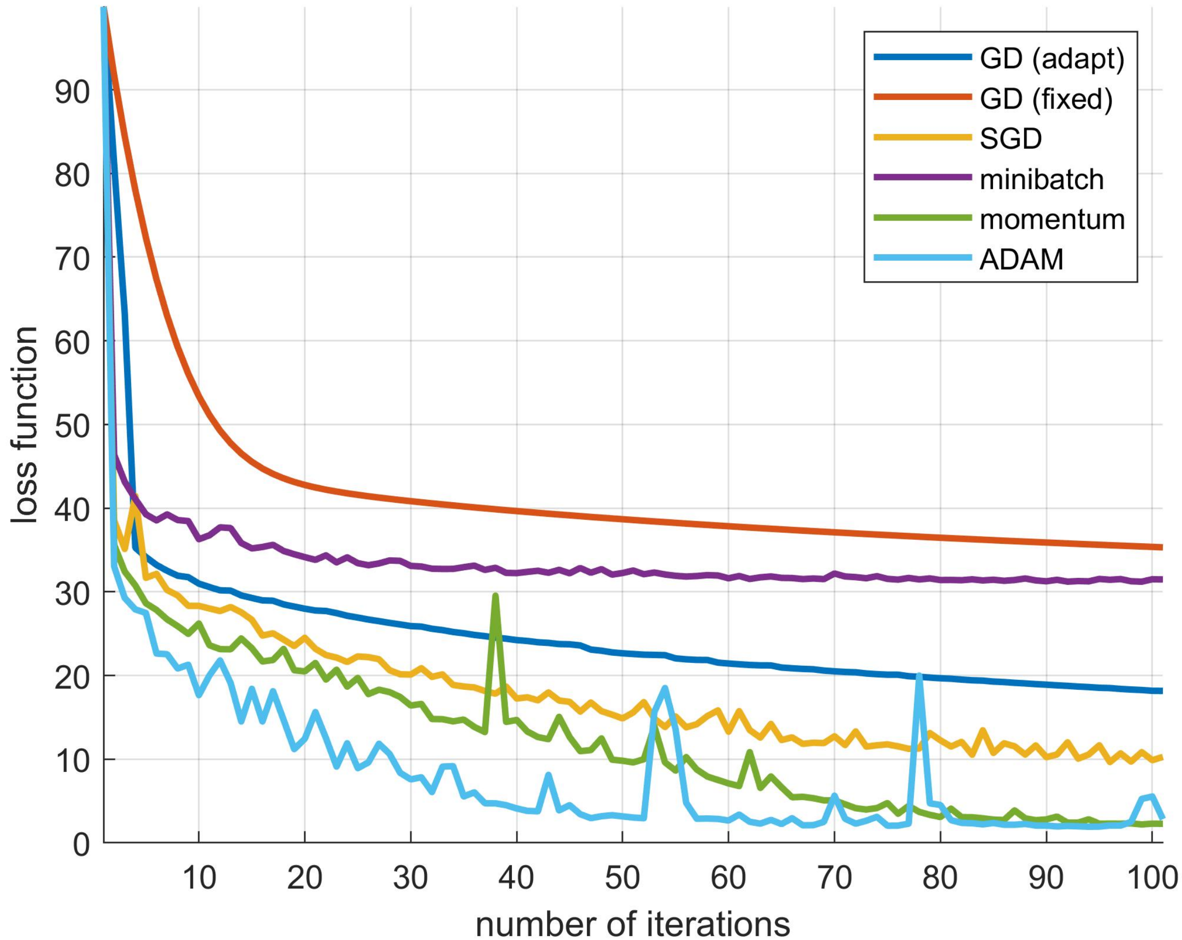
$$v_k = (1-\beta_1)\tilde{\nabla} J(u_k) \odot \tilde{\nabla} J(u_k) + \beta_2 v_{k-1}, \qquad \tilde{v}_k = \frac{v_k}{1-\beta_2^{k+1}}.$$

Pseudo code for gradient descent with momentum

- Choose an initial guess u_0 and set $m_0 = 0$ and $v_0 = 0$.
- Choose a step size $\beta > 0$ and momentum parameter $\gamma \in (0, 1)$.
- for k = 1: `max_iters`
- for j = 1: I
- Select randomly an index $i \in \{1, 2, \ldots, I\}$
- Compute $g_0 = I\nabla J_i(u_0)$.
- Set $m_0 = (1-\beta_1)g_0 + \beta_1 m_0$.
- Set $v_0 = (1-\beta_2)g_0 \odot g_0 + \beta_2 v_0$.
- Set $\tilde{m}_0 = m_0/(1-\beta_1^k)$.
- Set $\tilde{v}_0 = v_0/(1-\beta_2^k)$.
- Set $u_0 = u_0 - \beta\tilde{m}_0/(\sqrt{\tilde{v}_0} + \varepsilon)$.

Example: 100 epochs of stochastic gradient descent with momentum

Training of a ResNet with 100 hidden layers in $\mathbb{R}^2$ on 64 data points.



GD (adapt)	12.9 s
GD (fixed)	7.0 s
SGD	11.1 s
minibatch	10.1 s
momentum	11.0 s
ADAM	11.1 s

Example: 100 epochs of stochastic gradient descent with momentum