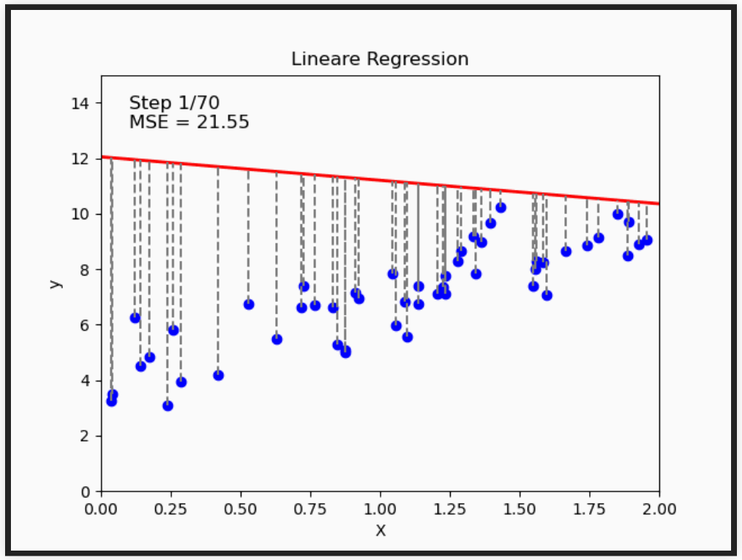
Optimierung

Mathematics of Machine Learning

Wiederholung - Empirical Risk Minimization (ERM)

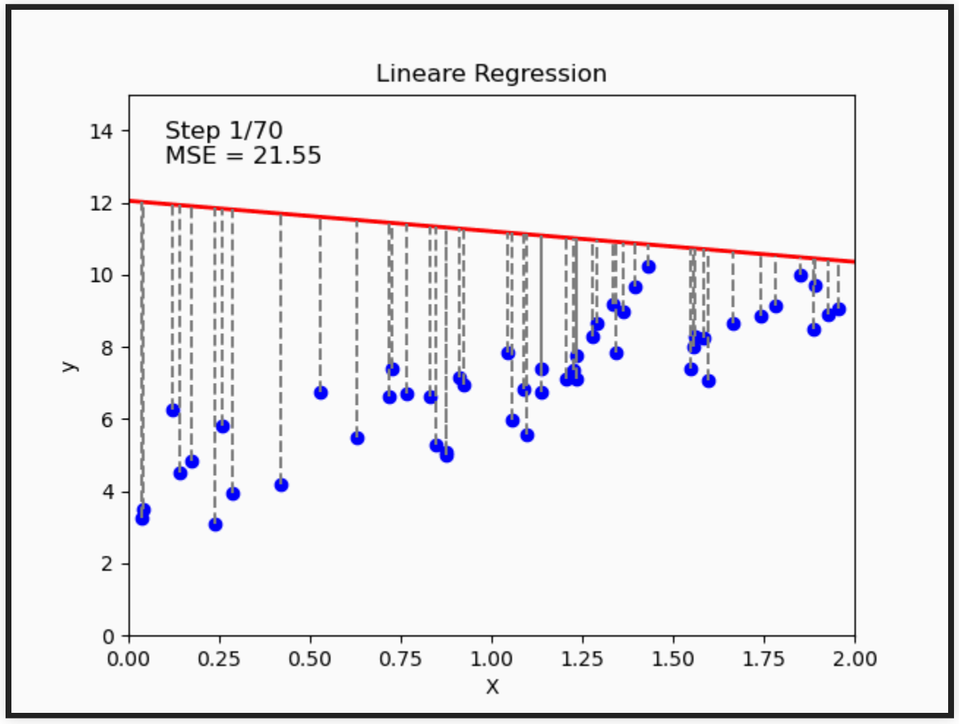
Idee - mit Mean-Squarred Error $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$



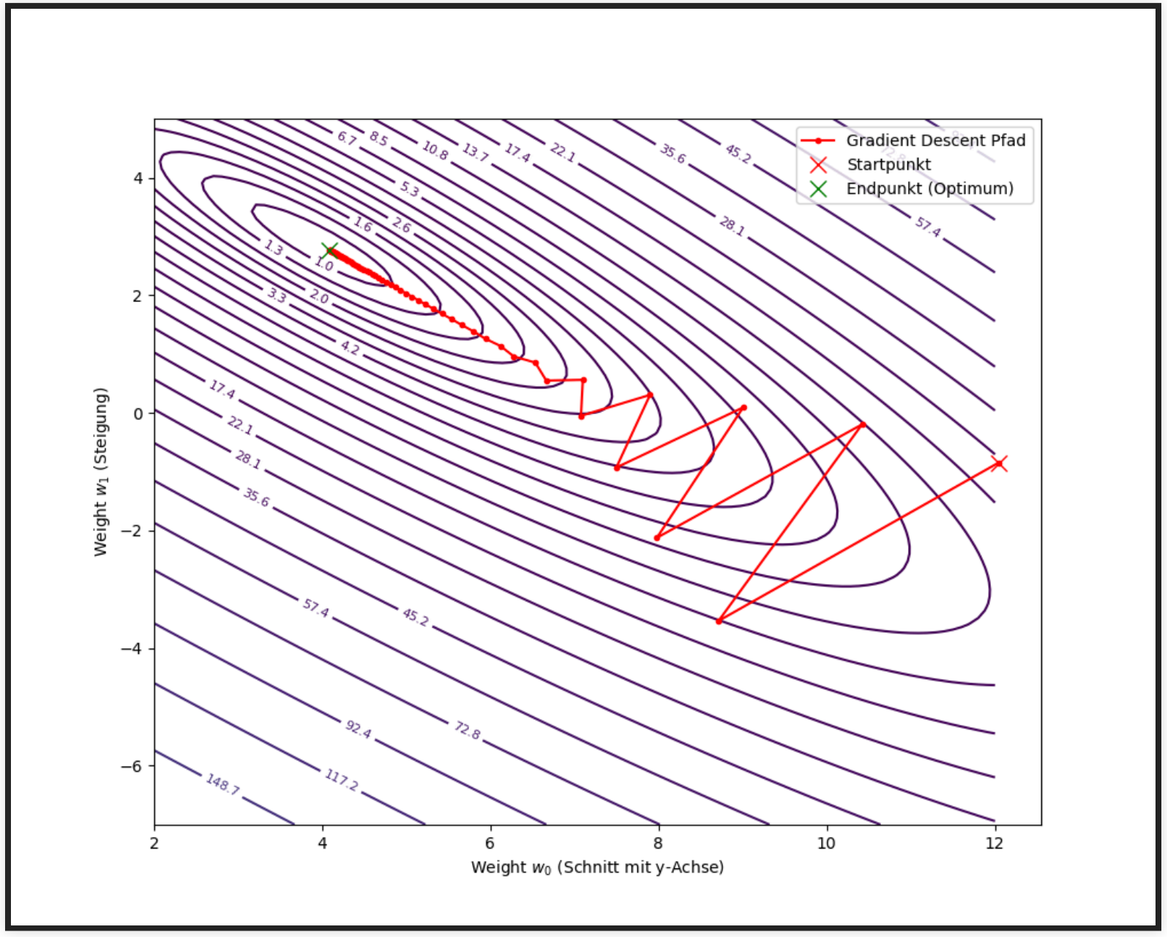
ERM - Formel

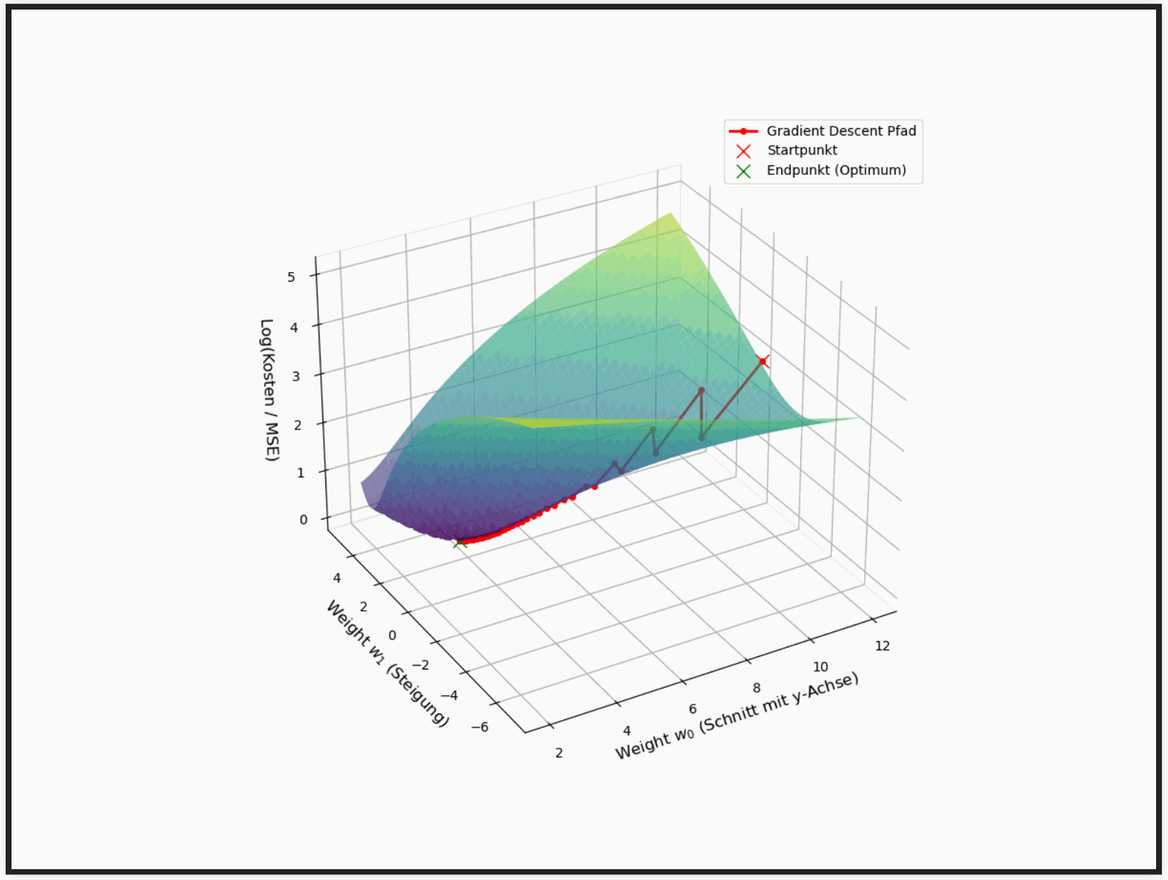
$$\min_{w \in \mathbb{R}^P} \frac{1}{m} \sum_{i=1}^{m} \ell(f_w(x_i), y_i)$$

- $w \in \mathbb{R}^P$ sind die trainierbaren **Parameter**
- (x_i, y_i) sind die **Trainingsdaten**
- $\ell(f_w(x_i), y_i)$ die **Loss Funktion**
- $\frac{1}{m} \sum_{i=1}^{m} \ell(f_w(x_i), y_i)$ durchschnittlicher Trainingsfehler



Landschaft der Kostenfunktion





generische Kostenfunktion

- $C : \mathbb{R}^P \to \mathbb{R}_+$
- im folgenden sei $C(w) = L(f_w) = \frac{1}{m} \sum_{i=1}^{m} \ell(f_w(x_i), y_i)$
- Ziel: Finde w^*, das $C(w)$ minimiert

Definitionen

C ist **differenzierbar**, falls für alle $i = 1, \ldots, P$ der Grenzwert existiert:

$$\frac{\partial C}{\partial w_i} = lim_{h \to 0} \frac{C(w_1, \ldots, w_i + h, \ldots, w_P) - C(w)}{h}$$

Gradient ist der Vektor aller partiellen Ableitungen:

$$\nabla_w C(w) = \begin{pmatrix} \frac{\partial C}{\partial w_1} \\ \vdots \\ \frac{\partial C}{\partial w_P} \end{pmatrix}$$

Notation: $\mathcal{C}^k(M)$, bezeichnet die Menge aller k-mal stetig differenzierbaren Funktionen auf M.

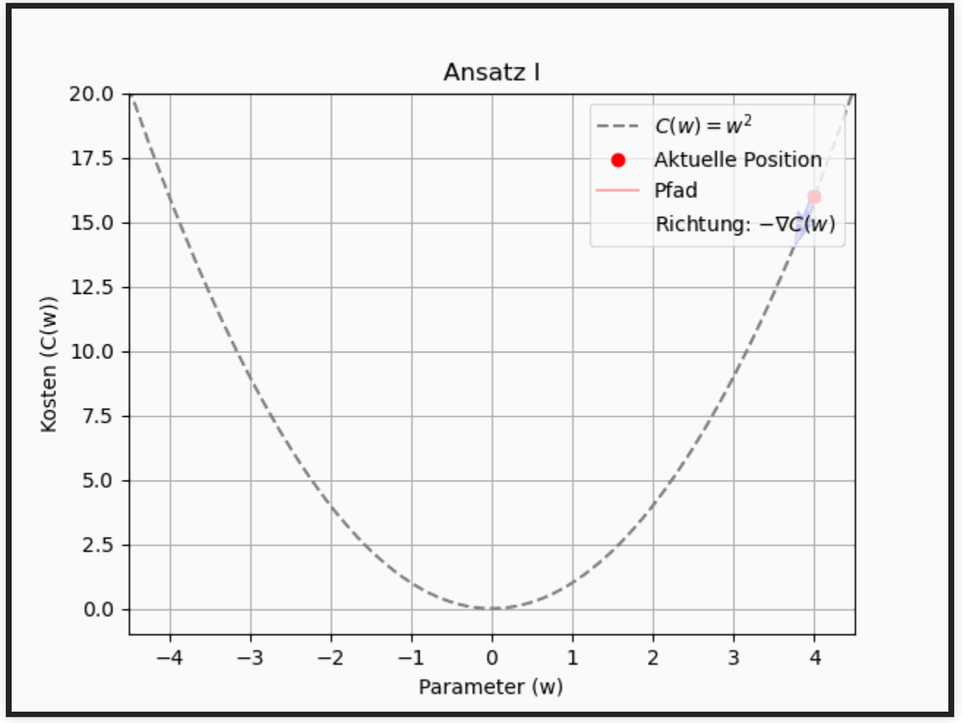
Im folgenden sei $C \in \mathcal{C}^1(\mathbb{R}^P)$

Geometrische Bedeutung des Gradienten

- der Gradient zeigt die Richtung des stärksten Anstiegs von C
- $\Rightarrow -\nabla_w C(w)$ ist die Richtung des stärksten Abfalls 💡

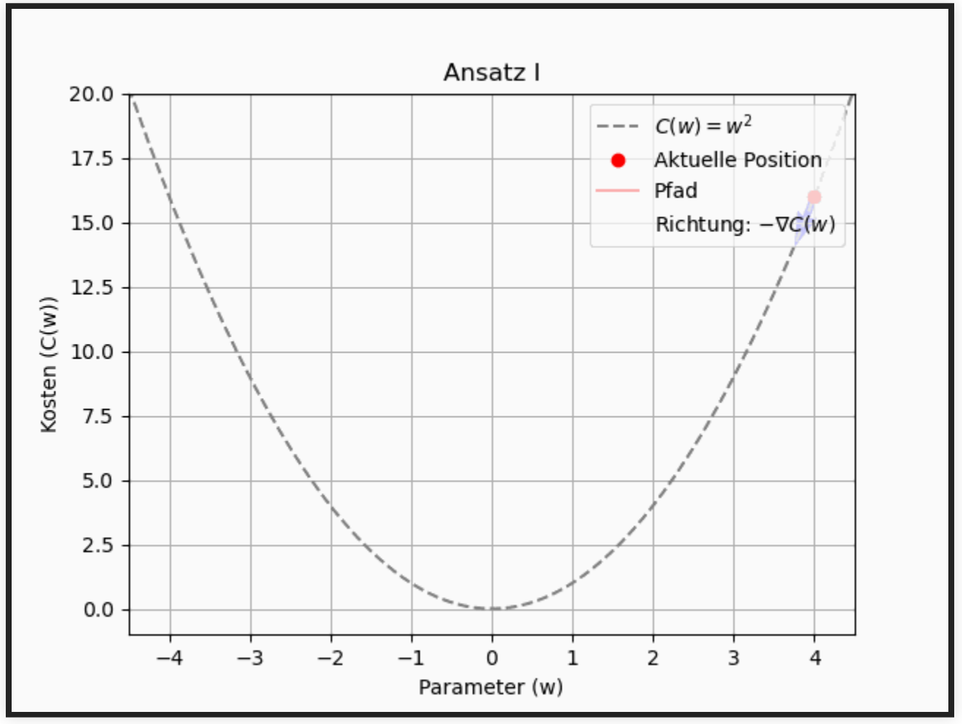
Ansatz 1

- starte an einem beliebigen Punkt $w_0 \in \mathbb{R}^P$
- gehe immer in Richtung $-\nabla_w C(w)$, solange bis… ❓



Ansatz 1 - Idee Abbruchkriterium

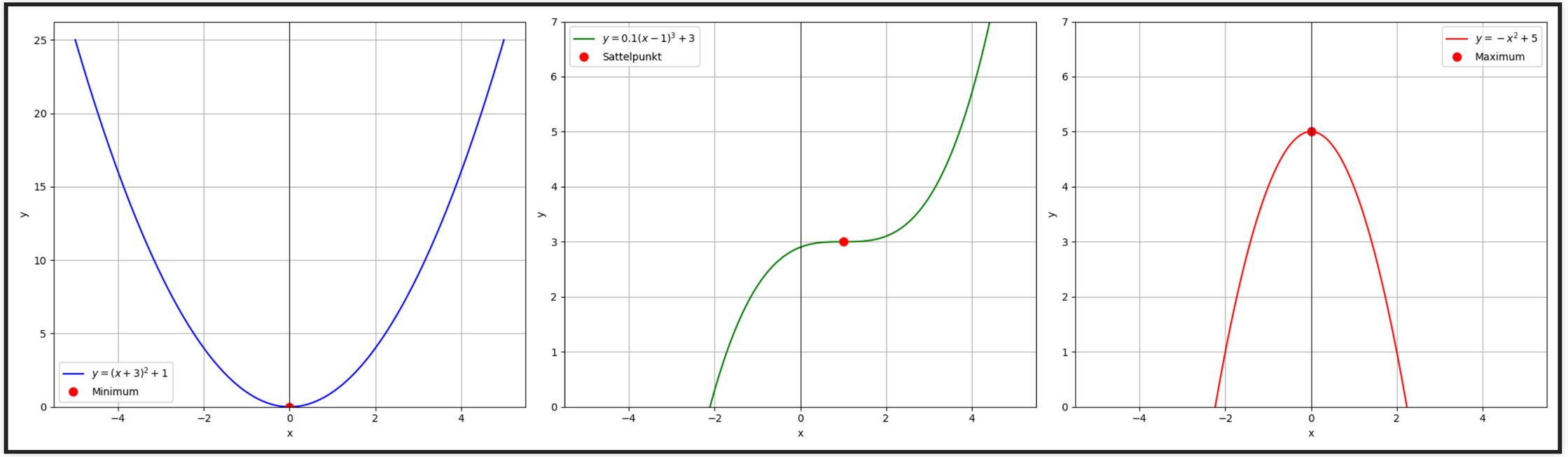
- wenn $\nabla_w C(w) = 0$ befinden wir uns an einem Punkt der in keiner Richtung lokal ansteigt oder abfällt
- $\Rightarrow$ gehe immer in Richtung $-\nabla_w C(w)$, solange bis $\nabla_w C(w) = 0$



Kritische Punkte

Kritischer Punkt: Ein Punkt w_*, an dem der Gradient Null ist:

$$\nabla_w C(w_*) = (0, \ldots, 0)^T \ .$$



Ein krischer Punkt ist entweder ein lokales Minimum, ein lokales Maximum oder ein Sattelpunkt.

Lokales/Globales Minimum:

Ein Punkt w_* heißt …

- **lokales Minimum** von C, wenn es eine Umgebung U von w_* in $\mathbb{R}^P$ gibt, s.d. $C(w_*) \leq C(w), \forall w \in U$.
- **striktes lokales Minimum**, wenn $C(w_*) < C(w), \forall w \in U \backslash \{w_*\}$.
- **globales Minimum**, wenn für alle w aus dem Definitionsbereich gilt das $C(w_*) \leq C(w)$.
- **Sattelpunkt** ein kritischer Punkt der kein lokales Optimum ist

Analog kann man (strikte) lokale/globale Maxima definieren

Notwendige Optimalitätsbedingung erster Ordnung

Ist w_* ein lokales Minimum von C, so gilt $\nabla C(w_*) = 0$, d.h. w_* ist ein kritischer Punkt von C.

Definition: Gradientenabstieg - Gradient Descent (GD)

Gegeben sei ein **Startpunkt** $w_0 \in \mathbb{R}^P$ und eine **Lernrate** $\eta > 0$. Die Parameter werden für $k = 0, 1, 2, \ldots$ wie folgt aktualisiert:

$$w_{k+1} = w_k - \eta \nabla C(w_k)$$

- ein einzelner Schritt $w_k \to w_{k+1}$ wird **Gradient Descent Step** genannt
- nach festgelegten Anzahl von $K \in \mathbb{N}$ Schritten ist der Algorithmus beendet

$$\mathcal{A}_{GD} : w_0 \to w_K$$

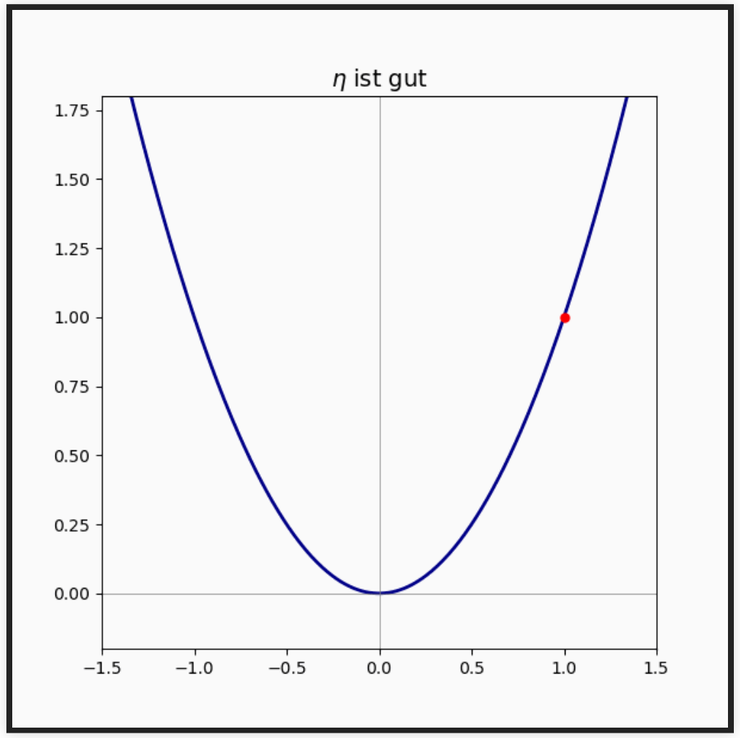
Bemerkung

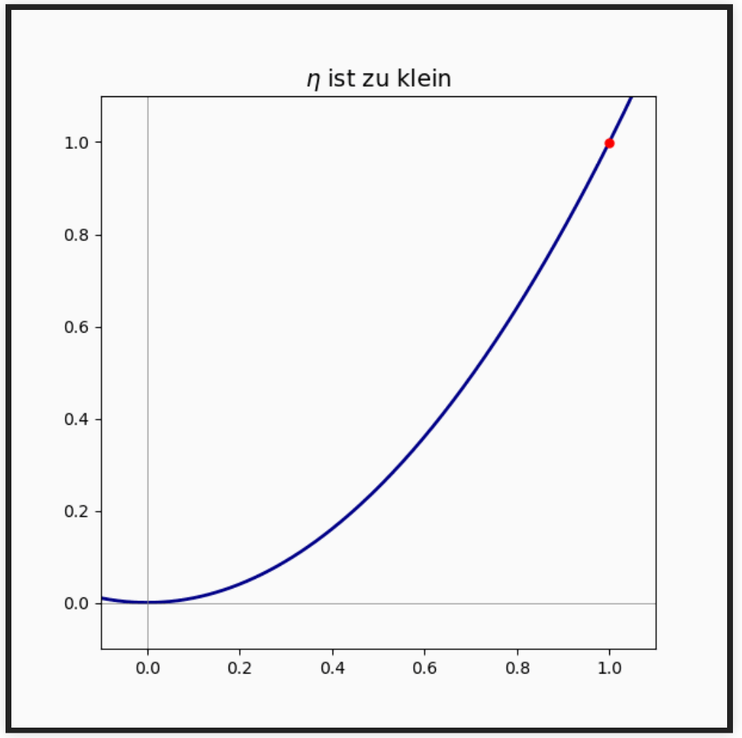
Im ERM framework sieht die Update Formel dann folgendermaßen aus,

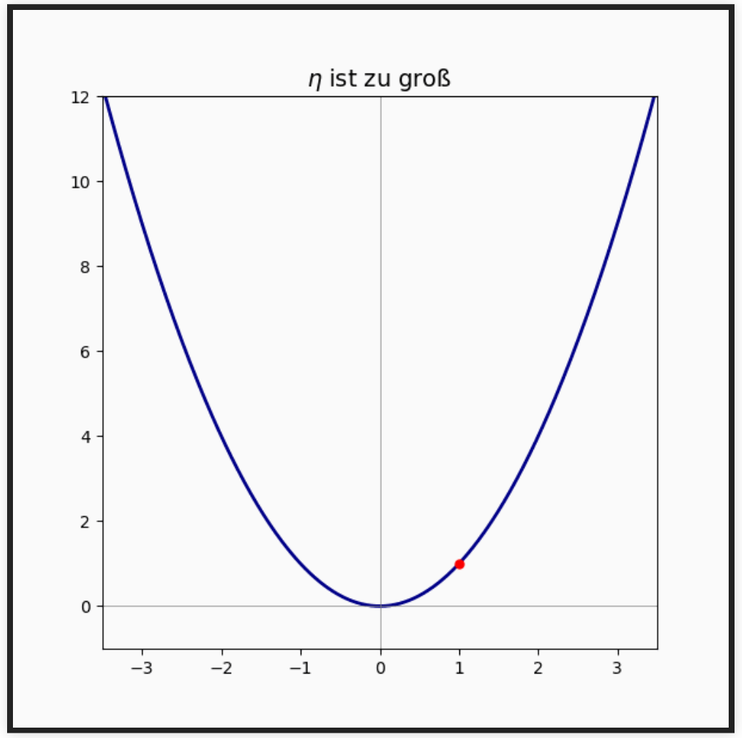
$$w_{k+1} = w_k - \eta \underbrace{\left(\frac{1}{m} \sum_{i=1}^{m} \nabla_w l(f_{w_k}(x_i), y_i) \right)}_{\text{Gesamtgradient } \nabla C(w_k)}$$

Bemerkung

- Hyperparameter von $\mathcal{A}_{GD}$: Lernrate η, Schrittanzahl K







Lösung - Hessian Matrix

Ist C zwei mal differenzierbar, bezeichnen wir mit

$$\mathcal{H}_w = \left(\partial_{w_i}\partial_{w_j} C(w)\right)_{1\leq i,j\leq P}$$

die **Hesse-Matrix** von C in w.

Eigenschaften

- beschreibt die **Krümmung** der Funktion
- **symmetrisch** (Satz von Schwarz) $\Rightarrow$ hat nur reele Eigenwerte λ_i und die Eigenvektoren e_i sind orthogonal zueinander

$\mathcal{H}_w$ ist **positiv definit** (semi-definit) $\Leftrightarrow \lambda_i > 0 (\geq 0), \forall i \in \{1, \ldots, P\}$

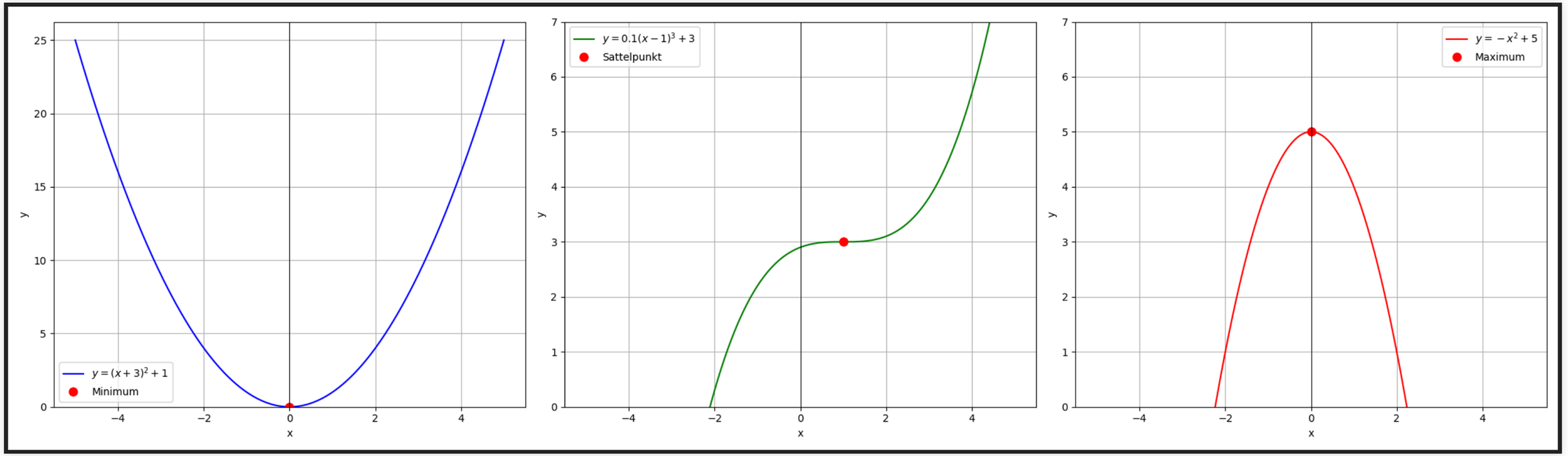
$\mathcal{H}_w$ ist **negativ definit** (semi-definit) $\Leftrightarrow \lambda_i < 0 (\leq 0), \forall i \in \{1, \ldots, P\}$

Optimalitätsbedingung zweiter Ordnung

Sei $M \subset \mathbb{R}^P$ offen und $C \in \mathcal{C}^2(M)$. Weiterhin gelte $w_* \in M$ und $\nabla C(w_*) = 0$.

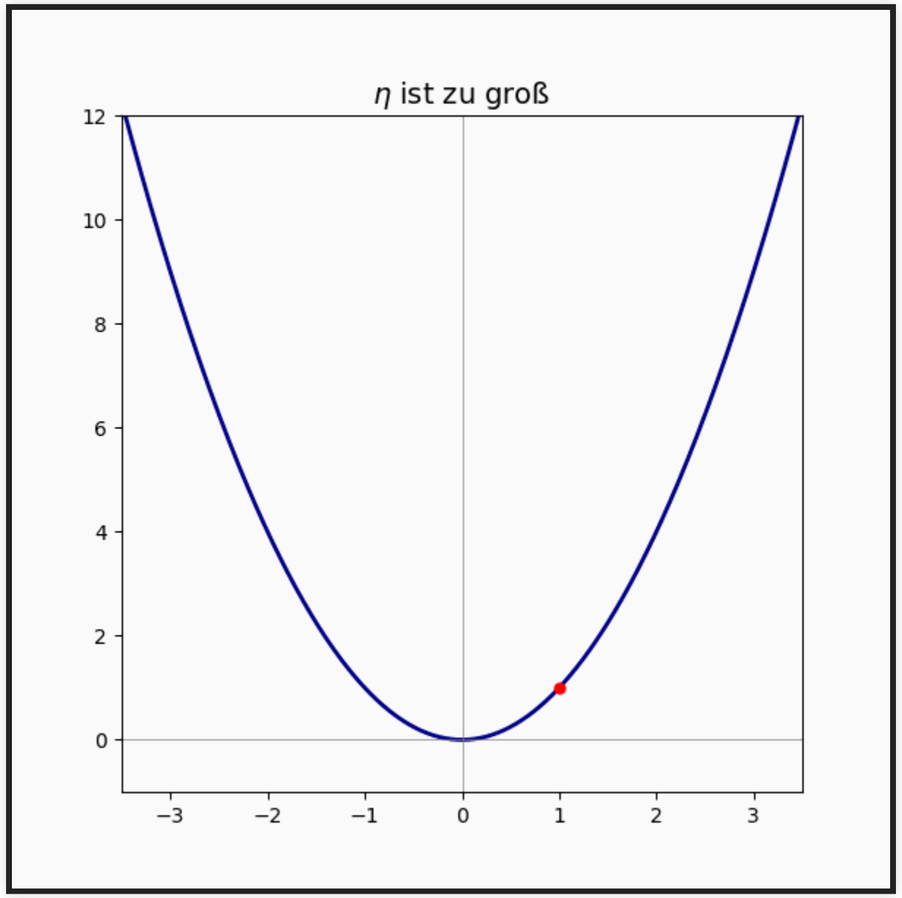
1. **(Hinreichende Bedingung)**: Ist $\mathcal{H}_{w_*}$ positiv (negativ) definit, so ist w_* ein lokales Minimum (Maximum) von C.
2. **(Notwendige Bedingung)**: Ist w_* lokales Minimum von C, so ist $\mathcal{H}_{w_*}$ positiv semidefinit.

Wenn $\mathcal{H}_{w_*}$ positive und negative Eigenwerte hat, dann ist w_* ein Sattelpunkt.


$y = (x+3)^2 + 1$
Minimum
x
y
$y = 0.1(x-1)^3 + 3$
Sattelpunkt
x
y
$y = -x^2 + 5$
Maximum
x
y

Limitierung

$$w_{k+1} = w_k - \eta \nabla C(w_k)$$



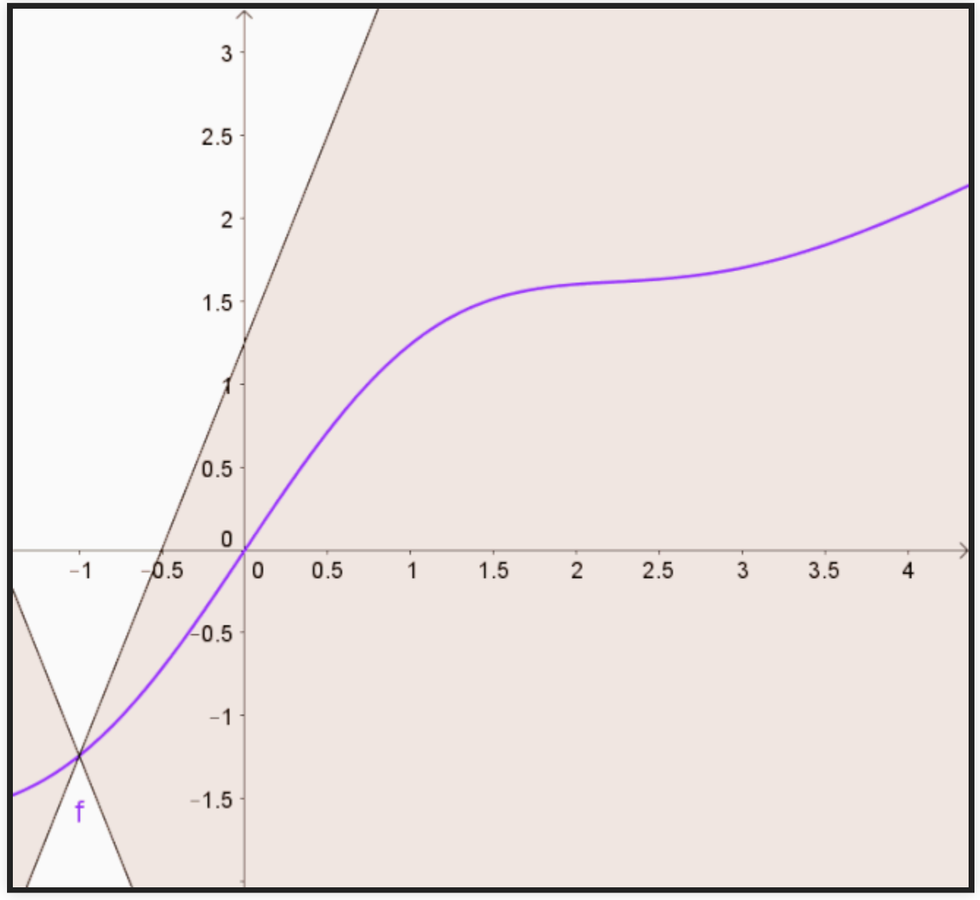
L-Lipschitz-Stetigkeit

- stärkere Form der Stetigkeit
- kontrolliert die Steigung global
- $f : \mathbb{R}^m \to \mathbb{R}^n$ ist **L-Lipschitz stetig** für $L > 0$, wenn

$$||f(x) - f(y)|| \leq L||x - y||, \forall x, y \in \mathbb{R}^m \Leftrightarrow ||\nabla f(x)|| \leq L$$

geometrische Intuition L-Lipschitz-Stetigkeit

$$||f(x) - f(y)|| \leq L||x - y||, \forall x, y \in \mathbb{R}^m \Leftrightarrow ||\nabla f(x)|| \leq L$$



- eine differenzierbare Funktion heißt **lipschitz-glatt**, wenn ihre Ableitung Lipschitz stetig ist

$$||\nabla f(x) - \nabla f(y)|| \leq L||x - y||, \forall x, y \in \mathbb{R}^m$$

- sei $C \in \mathcal{C}^2$ und $\forall w \in \mathbb{R}^P$, lass $\lambda_{max}(w), \lambda_{min}(w)$ der größte/ kleinste Eigenwert von $\mathcal{H}_w$ sein $\Rightarrow$

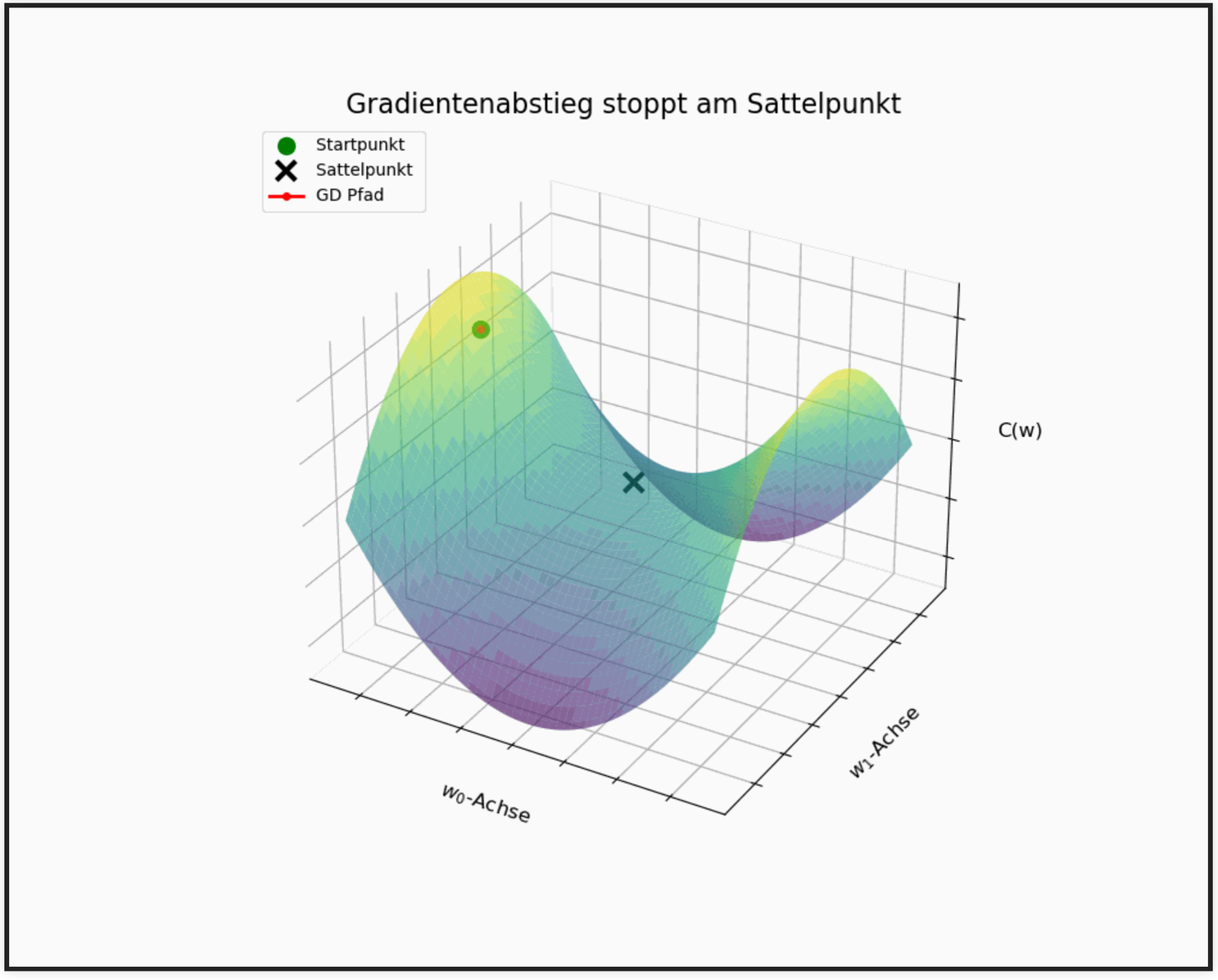
$$\boldsymbol{sup}_{w \in \mathbb{R}^P}\{|\lambda_{max}(w)|, |\lambda_{min}(w)| \leq L < \infty \Leftrightarrow C\}$$

  ist **L-glatt**

Satz - Konvergenz des Gradientenabstiegs

Sei...

- $C \in \mathcal{C}^2$
- $\eta > 0$ und einer Schrittanzahl $K \in \mathbb{N}$
- wenn C L-glatt ist und die Lernrate $\eta < 2/L$ erfüllt, dann :
  - konvergiert der Wert der Kostenfunktion, d.h. $lim_{K \to \infty} C(w_K)$ existiert
  - $lim_{K \to \infty} \nabla C(w_K) = 0$


Gradientenabstieg stoppt am Sattelpunkt
Startpunkt
Sattelpunkt
GD Pfad
C(w)
w_0-Achse
w_1-Achse

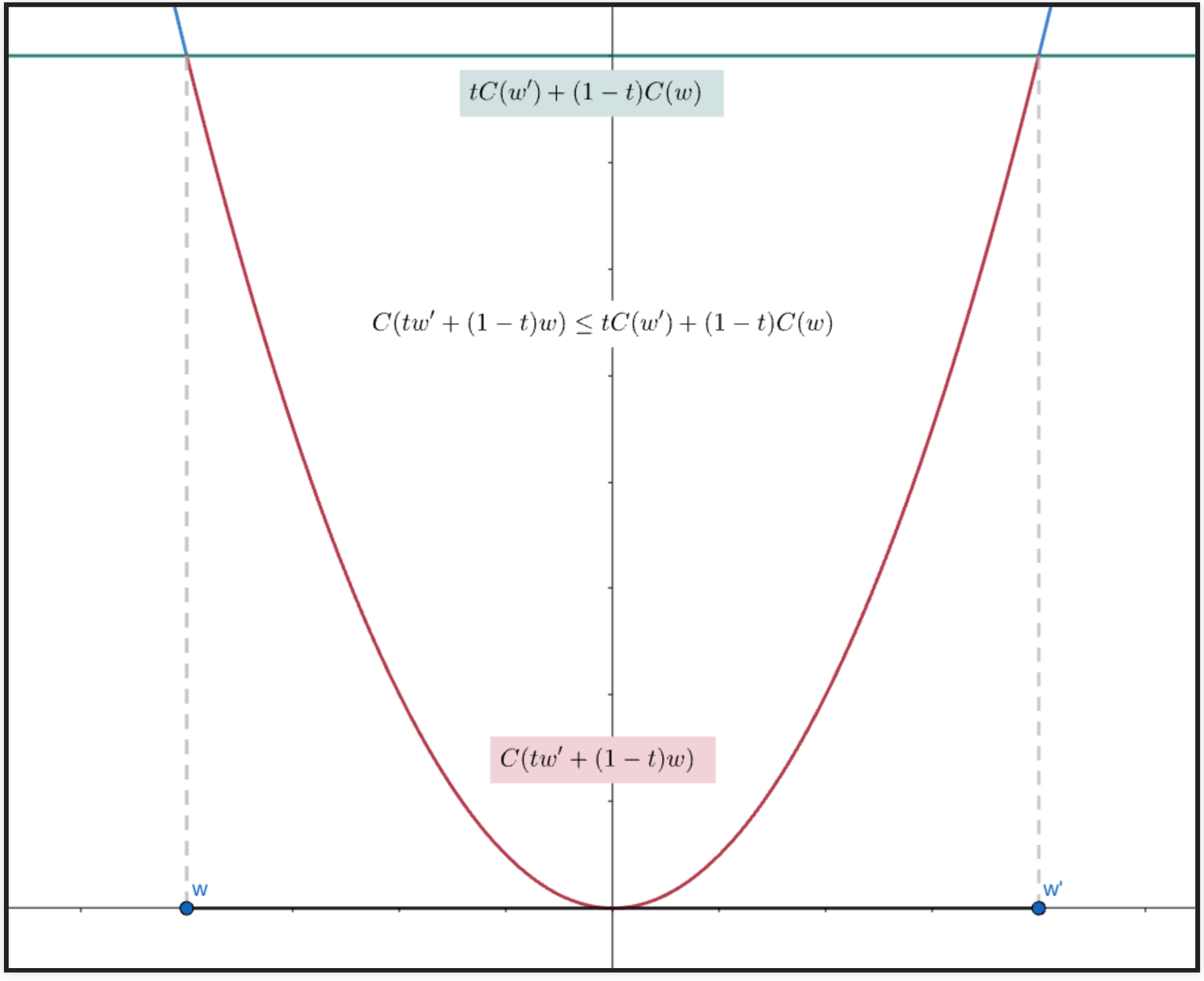
Konvexe Funktionen

- sei $M \subset \mathbb{R}^P$ eine nichtleere Teilmenge
- $C : M \to \mathbb{R}$ heißt **konvex**, falls folgende Bedingung erfüllt ist

$$C(tw' + (1-t)w) \leq tC(w') + (1-t)C(w), \forall w, w' \in M, t \in (0,1)$$

- C heißt **strikt konvex**, falls

$$C(tw' + (1-t)w) < tC(w') + (1-t)C(w), \forall w \neq w', t \in (0,1)$$


$tC(w') + (1-t)C(w)$
$C(tw' + (1-t)w) \leq tC(w') + (1-t)C(w)$
$C(tw' + (1-t)w)$
w
w'

Konvexe Funktionen

- sei $M \subset \mathbb{R}^P$ eine nichtleere Teilmenge
- $C : M \to \mathbb{R}$ heißt **konvex**, falls folgende Bedingung erfüllt ist

$$C(tw' + (1-t)w) \leq tC(w') + (1-t)C(w), \forall w, w' \in M, t \in (0,1)$$

- C heißt **strikt konvex**, falls

$$C(tw' + (1-t)w) < tC(w') + (1-t)C(w), \forall w \neq w', t \in (0,1)$$

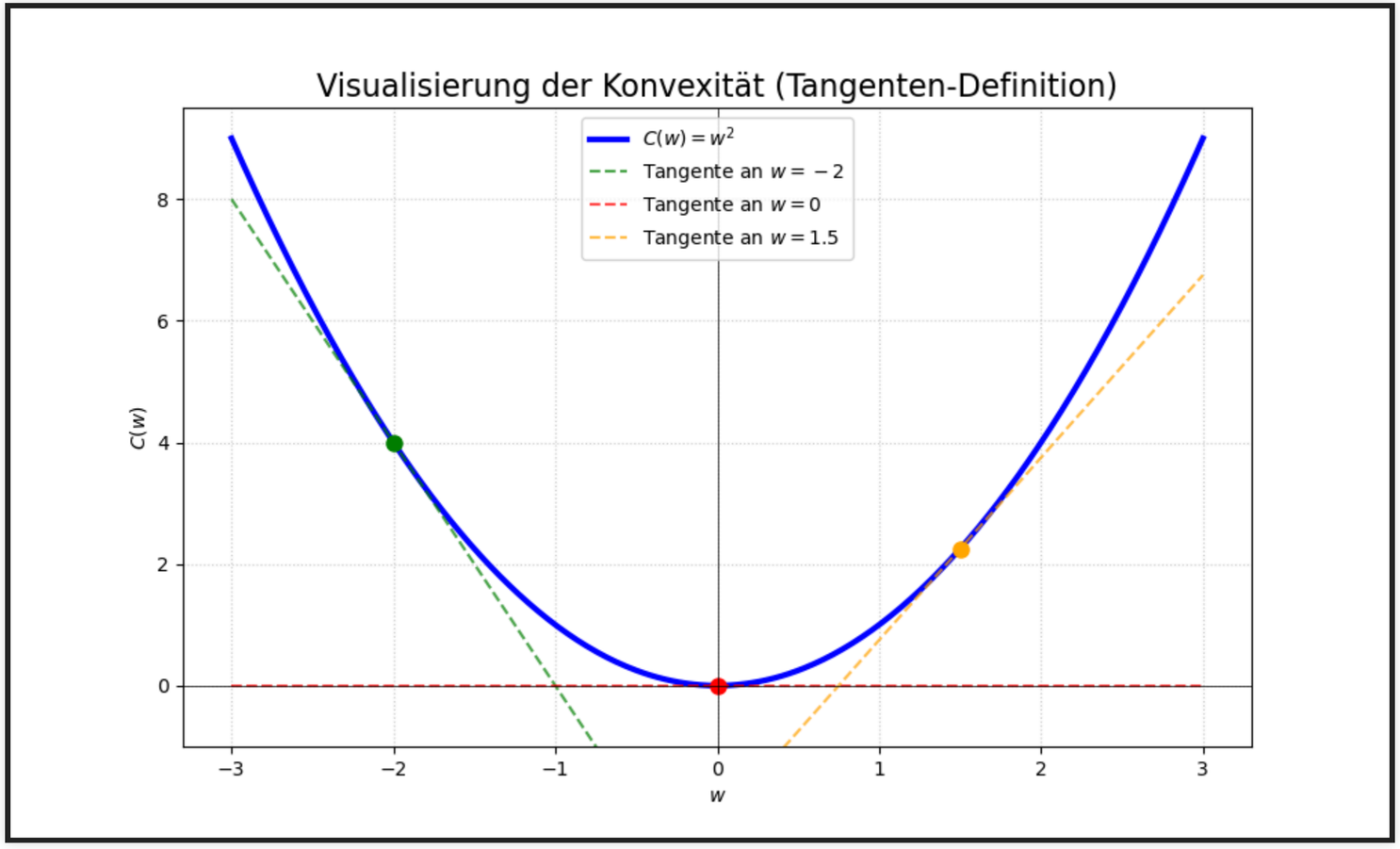
wenn C differenzierbar ist auf M, kann Konvexität auch folgendermaßen definiert werden:

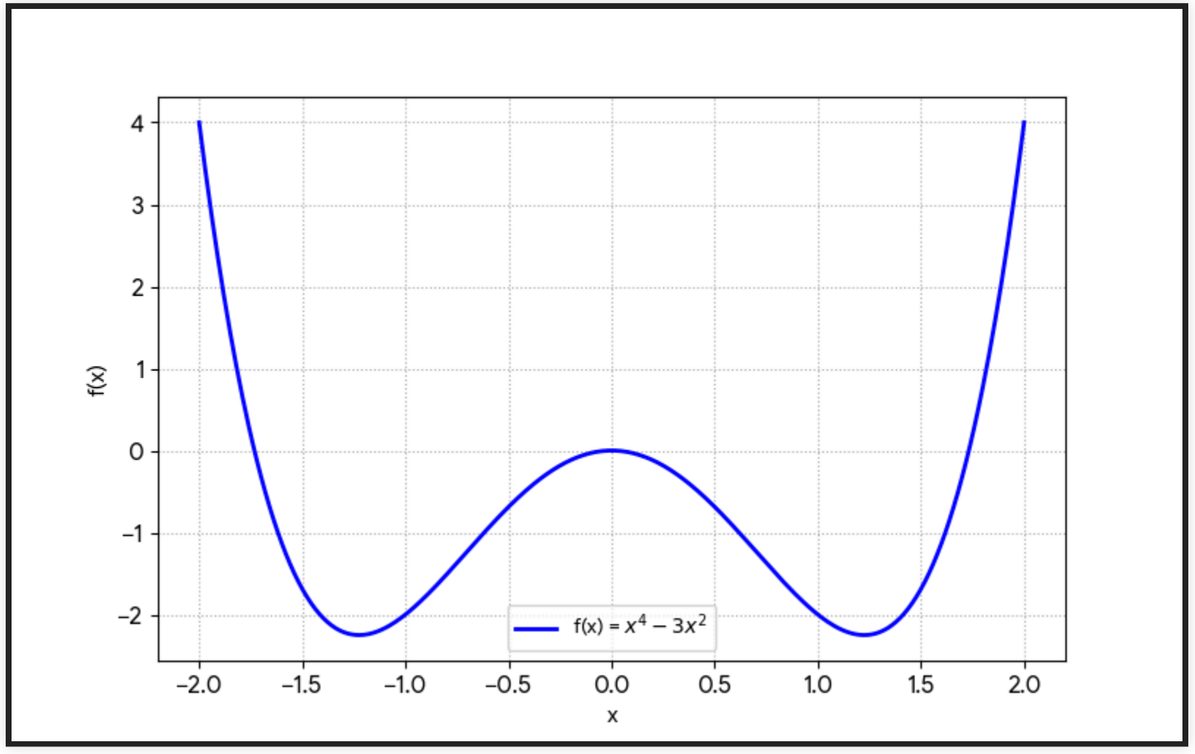
$$C(w') - C(w) \geq (w - w') \cdot \nabla_w C(w), \forall w, w' \in M$$

und umgeformt ergibt das

$$C(w') \geq \underbrace{C(w) + \nabla_w C(w) \cdot (w' - w))}_{\text{Tangente an der Stelle } w}$$

$$C(w') \geq \underbrace{C(w) + \nabla_w C(w) \cdot (w' - w))}_{\text{Tangente an der Stelle } w}$$




4
3
2
1
0
−1
−2
f(x)
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
x
f(x) = $x^4 - 3x^2$

Satz - Minima konvexer Funktionen

- ist C **konvex**, dann ist jeder kritische Punkt von C ein globales Minimum
- ist C **strikt konvex**, dann besitzt C höchstens ein globales Minimum

Was bisher geschah…

- L-Glattheit garantiert Konvergenz zu einem kritischen Punkt
- Konvexität garantiert das jeder kritische Punkt globales Minimum ist

1. Satz über Konvergenzgeschwindigkeit von GD

Annahmen:

- $C \in \mathcal{C}^2$
- C konvex und L-glatt
- $\eta \leq \frac{1}{L}$

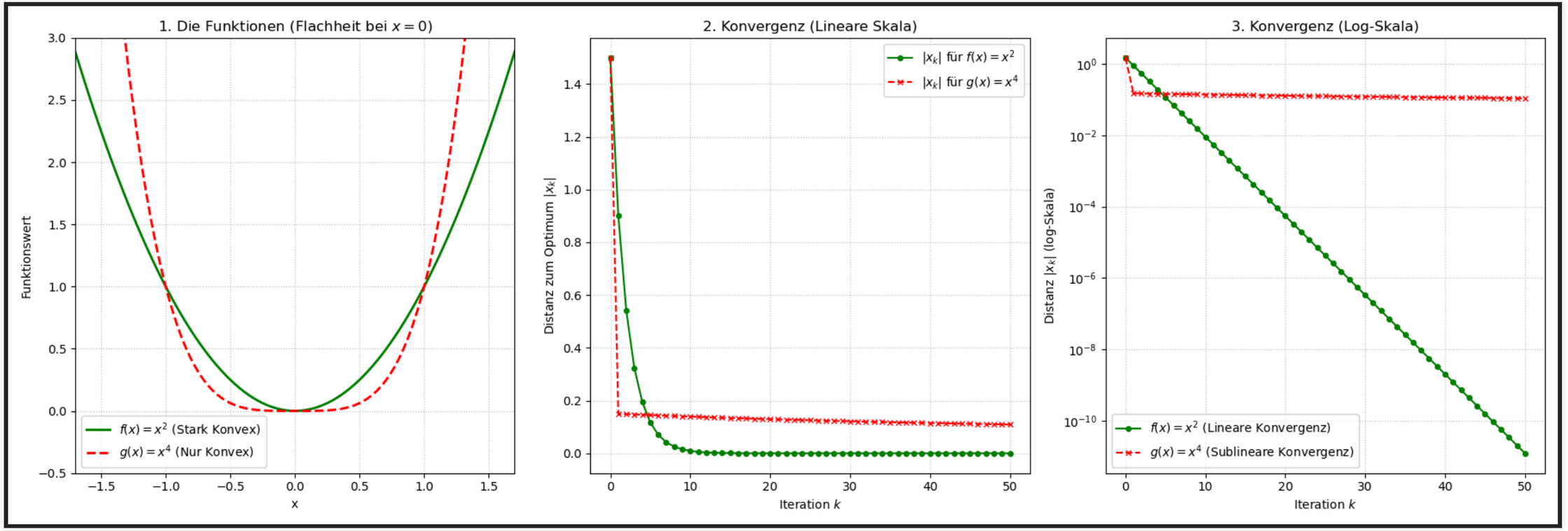
Ergebnis:

- nach K Schritten ist der Fehler $C(w_K)$ beschränkt durch:

$$0 \leq C(w_K) - C(w_*) \leq \frac{||w_0 - w_*||^2}{2\eta K}$$

- Konvergenzgeschwindigkeit von $\mathcal{O}(1/K)$

❓ geht es noch schneller ❓



Sei $C : \mathbb{R}^P \to \mathbb{R}$ und $C \in \mathcal{C}^2$ wenn der kleinste Eigenwert $\lambda_P(w)$ von $\mathcal{H}_w$ ist, s.d. $inf_{w \in \mathbb{R}^P} \lambda_P(w) \geq c > 0$, dann heißt C **c-stark konvex**.

Für alle $w, w' \in \mathbb{R}^P$ gilt dann:

$$C(w') - C(w) \geq (w' - w)\nabla_w C(w) + \frac{c}{2}||w' - w||^2,$$

2. Satz über Konvergenzgeschwindigkeit von GD

Annahmen:

- $\mathcal{C}$ ist L-glatt
- $\mathcal{C}$ ist c-stark konvex
- Lernrate ist $\eta \leq 1/L$.

Ergebnis:

- nach K Schritten ist der Fehler $C(w_K)$ beschränkt durch:

$$C(w_K) - C(w_*) \leq (1 - \eta c)^K \cdot (C(w_0) - C(w_*))$$

- Konvergenzgeschwindigkeit von $\mathcal{O}((1 - \eta c)^K)$

❓ ist das Problem jetzt perfekt gelöst ❓

unsere bisherige Update-Formel …

$$w_{k+1} = w_k - \eta \cdot \nabla C(w_k) \text{ mit } \nabla C(w_k) = \frac{1}{m} \sum_{i=1}^{m} \nabla_w l(f_{w_k}(x_i), y_i)$$

Problem

in der Praxis ist m oft riesig (Millionen oder Milliarden)

Konsequenz

Berechnung von $w_k \rightarrow w_{k+1}$ ist extrem rechenintensiv und in der Praxis viel zu langsam

Idee

schätzen des wahren Gradienten $\nabla C(w_k)$, anhand eines einzigen, zufällig ausgewählten Datenpunkt $(\tilde{x}_k, \tilde{y}_k)$ aus dem Datensatz S

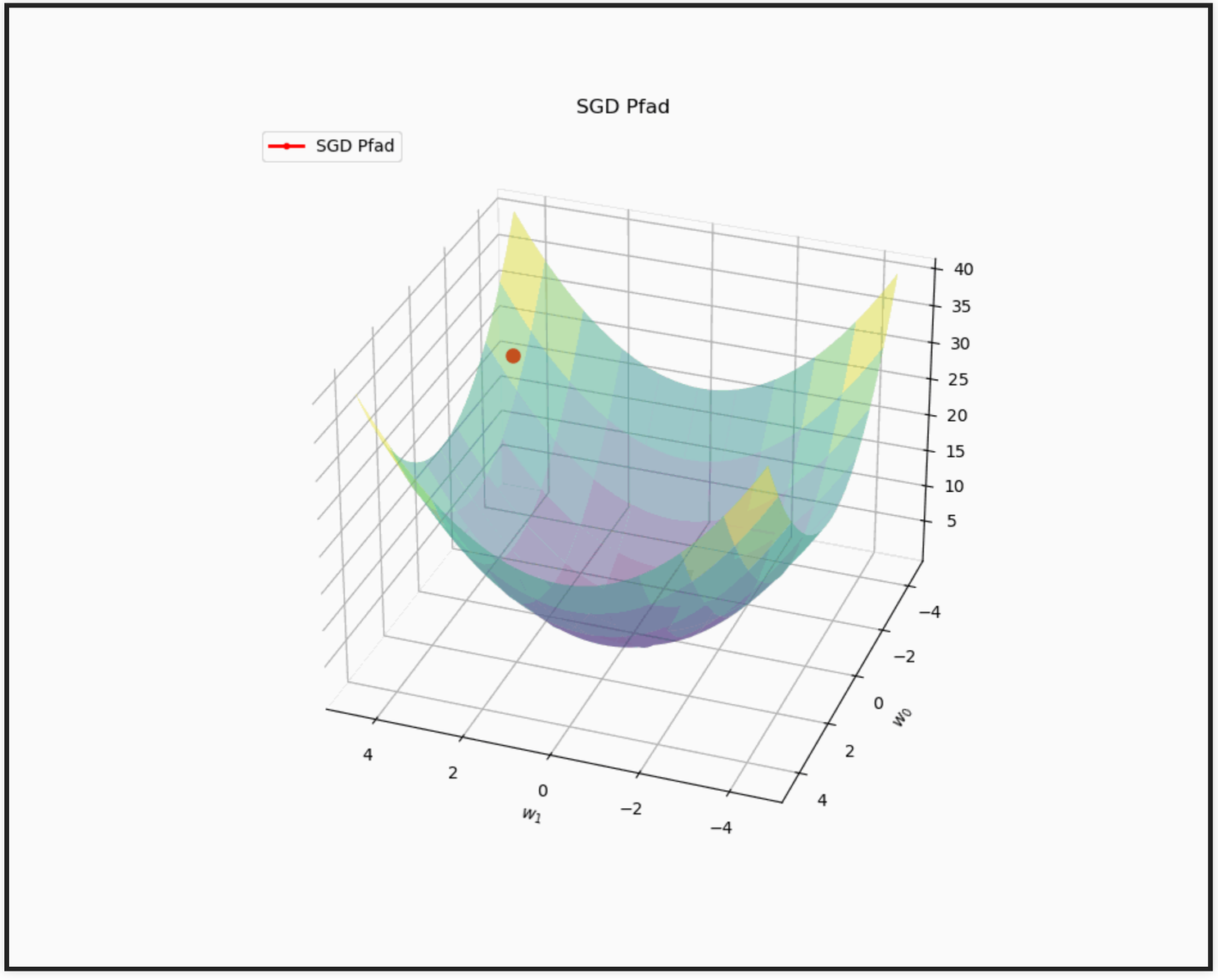
Stochastic Gradient Descent (SGD)

- wähle in jedem Schritt k ein Datenpaar $(\tilde{x}_k, \tilde{y}_k)$ zufällig und gleichverteilt aus dem gesamten Datensatz S aus
- berechne das Update nur basierend auf diesem einen Punkt:

$$w_{k+1} = w_k - \eta \cdot \nabla_w l(f_{w_k}(\tilde{x}_k), \tilde{y}_k)$$

Warum funktioniert das ?

- stochastische Gradient ist ein erwartungstreuer Schätzer für den wahren Gradienten
- $$\mathbb{E}[w_{k+1}^{SDG} | w_k^{SDG} = w_k^{GD}] = w_{k+1}^{GD}$$


SGD Pfad
SGD Pfad
40
35
30
25
20
15
10
5
4
2
0
−2
−4
w_1
4
2
0
−2
−4
w_0

Bemerkung

- in der Praxis wird nur selten ein einziger Punkt verwendet

Mini-Batch SGD

- wähle einen kleinen Mini-Batch B von b zufälligen Datenpunkten
- berechne den Gradienten als Durchschnitt über diesen Mini-Batch:

$$w_{k+1} = w_k - \eta \cdot \frac{1}{b} \sum_{(\tilde{x},\tilde{y}) \in B} \nabla_w l(f_{w_k}(\tilde{x}), \tilde{y})$$

Vorteile - Mini-Batch SGD

- geringere Varianz
- Effizienz

Konvergenz von SGD

Annahmen:

- $\mathcal{C} \in \mathcal{C}^2$ und $\exists w_* \in \mathbb{R}^P$, s.d. $C(w_*) = \frac{1}{m} \sum_{i=1}^{m} l(f_{w_*}(x_i), y_i) = 0$
- $(w_k)_{k \geq 0}$ durch SGD auf C erzeugt
- $\overline{w}_k := \frac{1}{k} \sum_{i=0}^{k-1} w_i$
- $w \mapsto l_w(x_i)$ L-glatt und konvex $\forall i = 1, \ldots, m$
- $\eta < \frac{1}{2L}$

Ergebnis: $\forall k \geq 1$

- $\mathbb{E}[C(\overline{w}_k)] \leq \frac{||w_0 - w_*||^2}{2\eta(1-2\eta L)k}$
- Konvergenzrate ist $\mathcal{O}(1/K)$

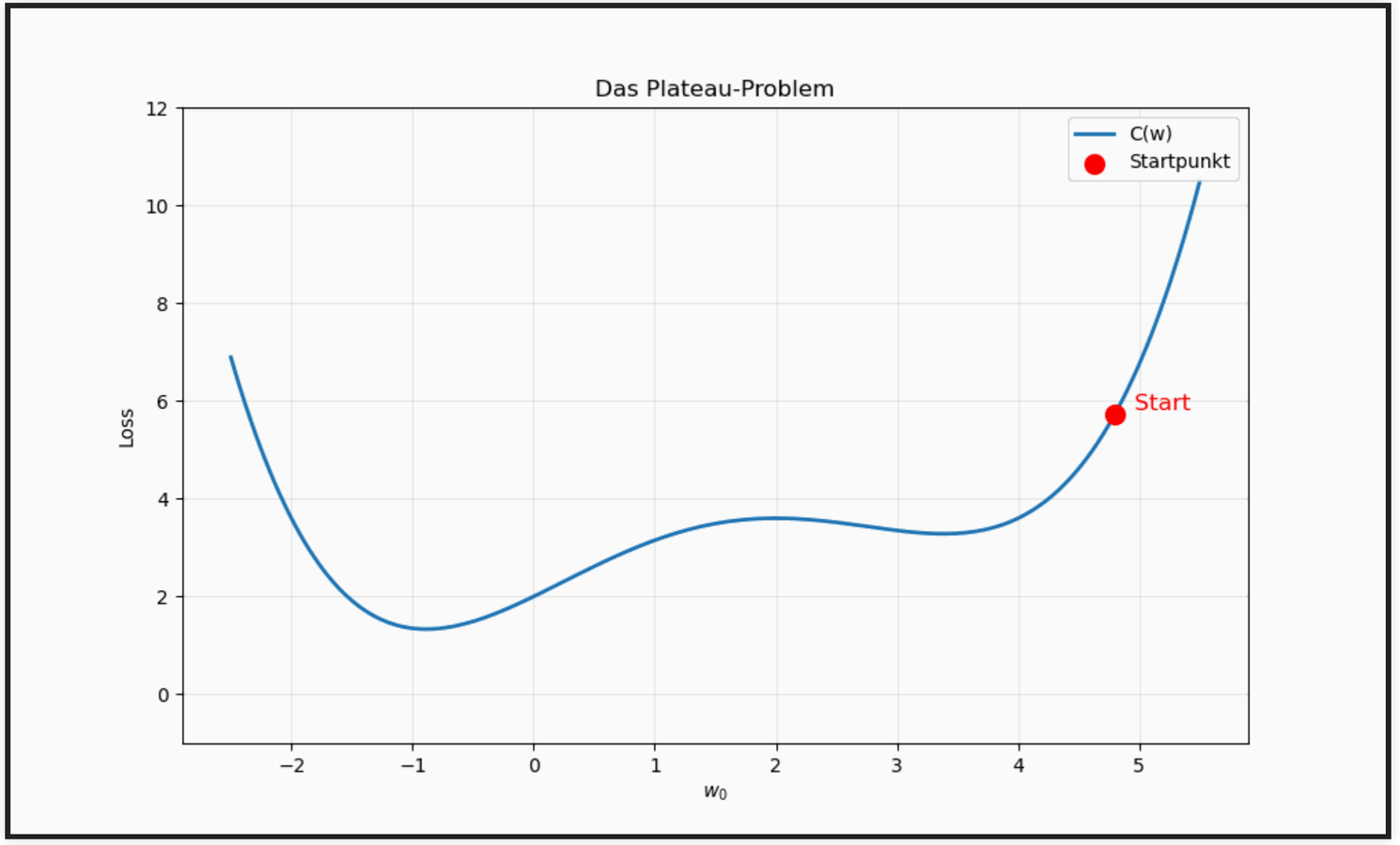
Konvergenz von SDG

Annahmen:

- $\mathcal{C} \in \mathcal{C}^2$ und $\exists w_* \in \mathbb{R}^P$, s.d. $C(w_*) = \frac{1}{m} \sum_{i=1}^{m} l(f_{w_*}(x_i), y_i) = 0$
- $(w_k)_{k \geq 0}$ durch SGD auf C erzeugt
- $\overline{w}_k := \frac{1}{k} \sum_{i=0}^{k-1} w_i$
- $w \mapsto l_w(x_i)$ L-glatt und c-stark konvex $\forall i = 1, \ldots, m$
- $\eta < \frac{1}{2L}$

Ergebnis: $\forall k \geq 1$

- $\mathbb{E}[||w_k - w_*||^2] \leq (1 - \eta c)^k ||w_0 - w_*||^2$
- Konvergenzrate $\mathcal{O}(\rho^K), \rho < 1$


Das Plateau-Problem
C(w)
Startpunkt
Start
Loss
w_0
12
10
8
6
4
2
0
−2
−1
0
1
2
3
4
5

Der Heavy Ball Algorithmus

Die Update-Formel mit Momentum:

$$w_{k+1} = w_k \underbrace{-\eta \nabla C(w_k)}_{\text{Gradienten-Update}} + \underbrace{\beta(w_k - w_{k-1})}_{\text{Momentum-Term}}$$

Parameter:

- $\beta \in [0, 1)$ **Momentum-Parameter** - bestimmt wie viel vom letzen Schritt behalten wird
  - $\beta = 0$: Standard Gradient Descent
  - $\beta \approx 0.9$: Typischer Wert (viel Schwung)

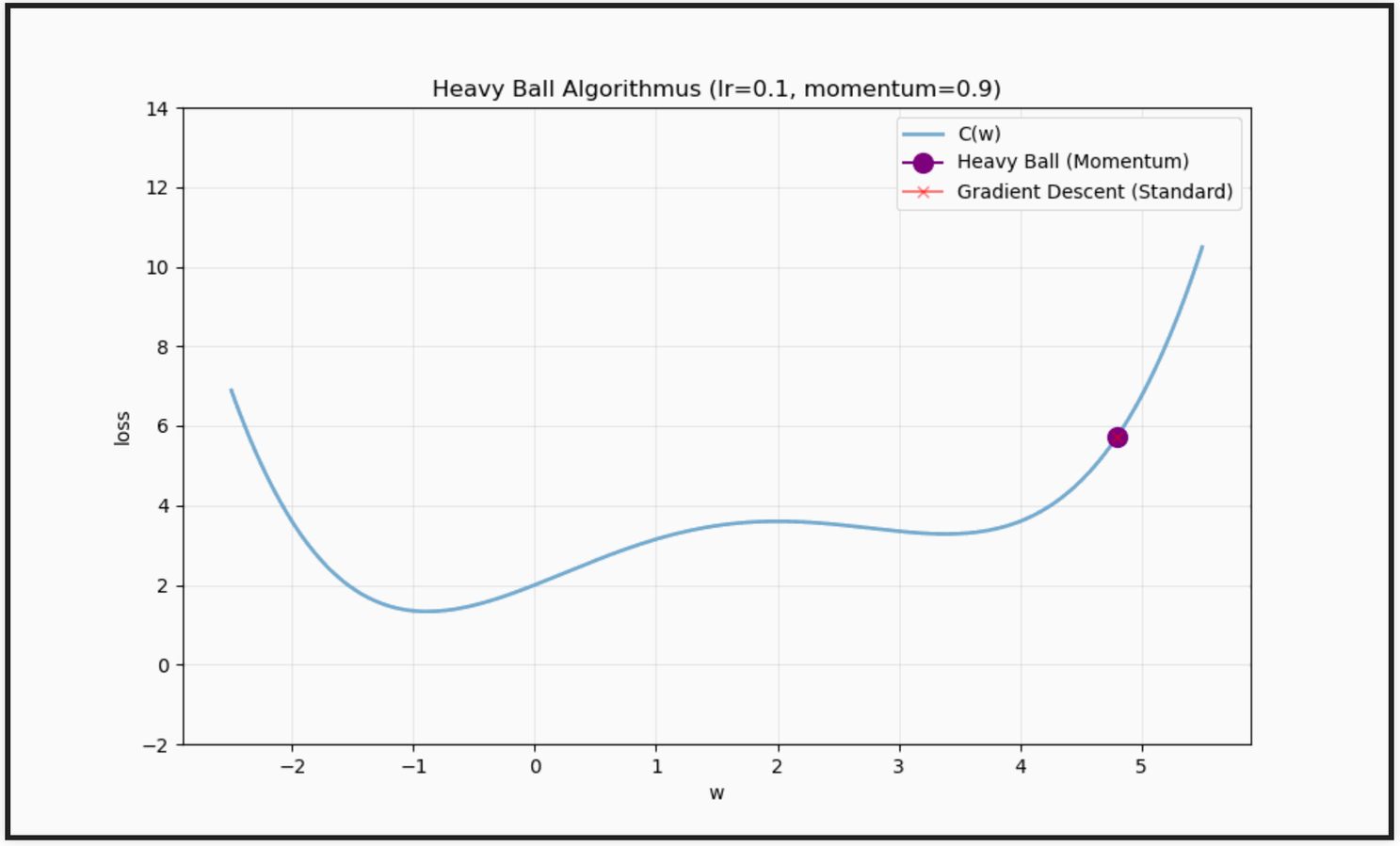
Das exponentielle Gedächtnis

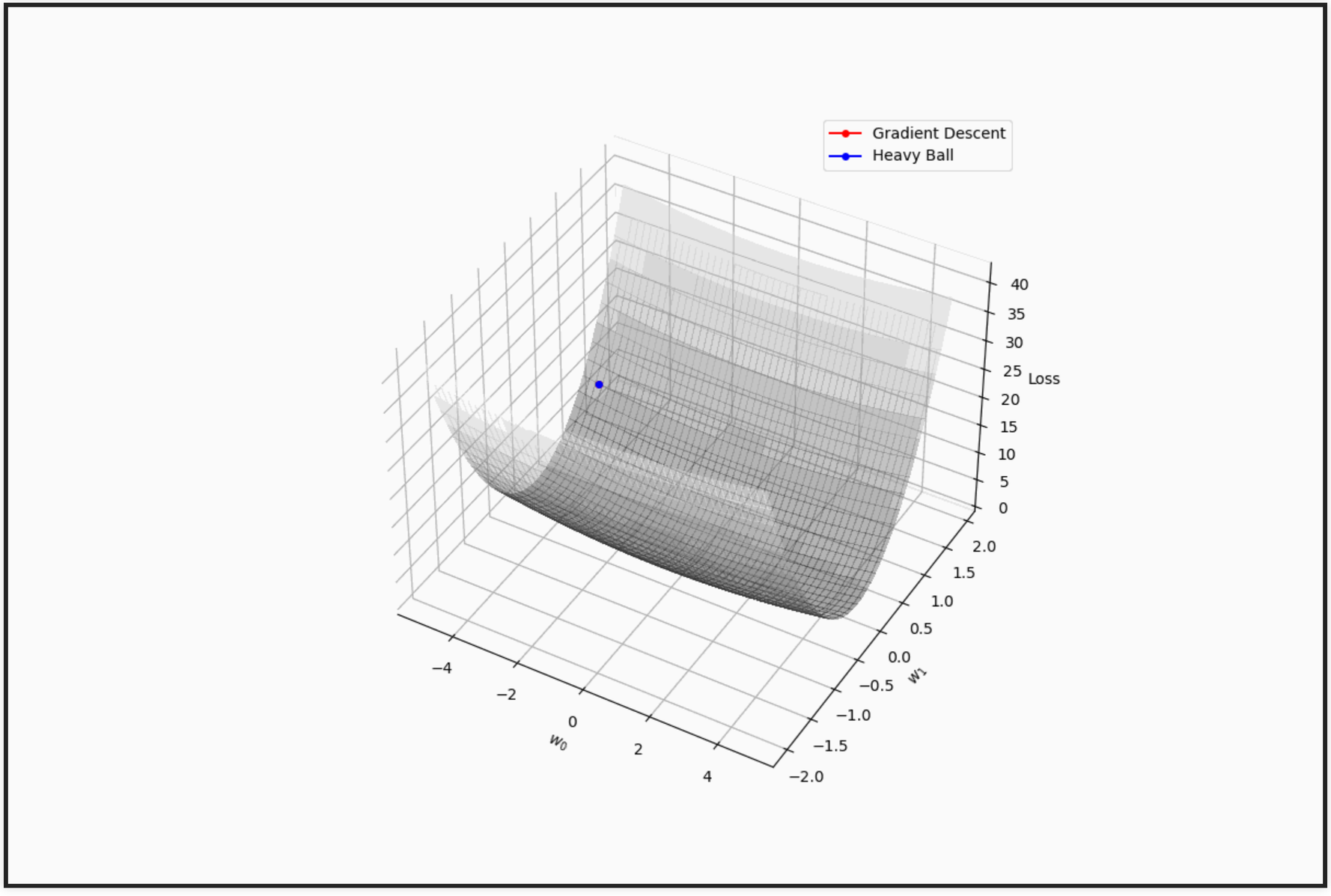
$$w_1 - w_0 = -\eta \nabla C(w_0)$$

$$w_2 - w_1 = -\eta \nabla C(w_1) + \beta \underbrace{(w_1 - w_0)}_{\text{ersetzen}} = -\eta(\nabla C(w_1) + \beta \nabla C(w_0))$$

$$\vdots$$

$$w_k - w_{k-1} = -\eta \sum_{i=1}^{k} \beta^{k-i} \nabla C(w_{i-1})$$


Heavy Ball Algorithmus (lr=0.1, momentum=0.9)
C(w)
Heavy Ball (Momentum)
Gradient Descent (Standard)
loss
w
14
12
10
8
6
4
2
0
−2
−2
−1
0
1
2
3
4
5


Gradient Descent
Heavy Ball
40
35
30
25
Loss
20
15
10
5
0
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
w_1
−4
−2
0
2
4
w_0

Bemerkung

- **L-glatte, konvexe Funktionen**: Konvergenzrate $O(1/k)$
- **L-glatte, c-stark konvexe Funktionen**: Konvergenzrate

$$O(r^k) \quad \text{für ein } r \in (0,1)$$

- gibt weitere Algorithmen, die das Momentum-Konzept nutzen, wie z.B. Adam

Konvergenz - im allgemeinen Fall

- keine Garantie für Konvergenz zu gobalen oder lokalen Minimum
- garantiert nur Konvergenz zu stationären Punkt $\nabla C(w_*)$
- über die Konvergenzgeschwindigkeit kann keine Aussage getroffen werden
- aber ist $C(w)$ L-glatt, konvergiert die Norm des Gradienten gegen 0

$$\min_{0 \leq k \leq K} \|\nabla C(w^k)\| \leq \sqrt{\frac{L(C(w_0) - C(w_*))}{\eta(K+1)}}$$

Bisher: $w \in \mathbb{R}^P$

Problem: in der Realität selten völlig frei von Nebenbedingungen

Optimierung mit Nebenbedingungen

$$
\begin{aligned}
\min_{w} \quad & C(w), \\
\text{s. t.} \quad & h_i(w) = 0,\ i = 1, \ldots, r, \\
\text{s. t.} \quad & f_i(w) \leq 0,\ i = 1, \ldots, p.
\end{aligned}
$$

Lösung : Lagrange Multiplikatoren

- wandeln das beschränkte Problem in ein unbeschränktes Problem um

$$\mathcal{L}(w, \alpha, \beta) = C(w) + \sum \alpha_i h_i(w) + \sum \beta_i f_i(w)$$
$$\alpha \in \mathbb{R}^r, \beta \in [0, \infty)^p$$

- α_i, β_i - Lagrange Multiplikatoren

Primale Problem (Min-Max)

- primale Funktion $g_{\text{prim}}(w) := \max_{\alpha,\beta \geq 0} \mathcal{L}(w, \alpha, \beta)$
- Optimierungsproblem wird zu: $\min_w g_{\text{prim}}(w)$

Analyse von $g_{\text{prim}}(w)$:

- **Fall 1**: w verletzt Regel (z.B. $h_i(w) \neq 0) \Rightarrow \max_\alpha(\cdots + \alpha_i h_i(w)) \to \infty$ $g_{\text{prim}}(w) = \infty$ (Unendliche Strafe)
- **Fall 2**: w hält Regeln ein $\Rightarrow$ Alle Strafterme sind 0 (für h) oder ≤ 0 (für f)
- Maximum (bei $\beta = 0$) ist genau $C(w)$

Duales Problem

- duale Funktion $g_{\text{dual}}(\alpha, \beta) := \min_w \mathcal{L}(w, \alpha, \beta)$
- duale Problem maximiert diese untere Schranke :

$$d^* = \max_{\alpha, \beta \geq 0} g_{\text{dual}}(\alpha, \beta)$$

$$\mathcal{L}(w, \alpha, \beta) \leq g_{\text{prim}}(w), \forall w \in \mathbb{R}^P, \alpha \in \mathbb{R}^r, \beta \in [0, \infty)^p$$

$$\underbrace{\min_{w \in \mathbb{R}^P} \mathcal{L}(w, \alpha, \beta)}_{g_{\text{dual}}(\alpha, \beta)} \leq \mathcal{L}(w, \alpha, \beta) \leq g_{\text{prim}}(w)$$

$$\Rightarrow g_{\text{dual}}(\alpha, \beta) \leq \min_w g_{\text{prim}}(w) = p^*$$

$$\Rightarrow d^* \leq p^*$$

in einigen Fällen ist $d^* = p^*$, das nennt man **starke Dualität**

Karush-Kuhn-Trucker (KKT) Bedingung

Bedingung	Formel
1. Stationarität	$\nabla_w \mathcal{L}(w^*) = 0$
2. Zulässigkeit	$h_i(w^*) = 0$ $f_i(w^*) \leq 0$
3. Duale Zulässigkeit	$\beta_i^* \geq 0$
4. Komplementärer Schlupf	$\beta_i^* \cdot f_i(w^*) = 0$

Frage: Wann können wir uns auf KKT verlassen?

1. Die Notwendigkeit ($\Rightarrow$)

- **Wenn** Starke Dualität gilt ($d^* = p^*$),
- **Dann** erfüllen die optimalen Lösungen (w^*, α^*, β^*) zwingend die KKT-Bedingungen.

2. Die Hinlänglichkeit ($\Leftarrow$)

- **Wenn** alle Funktionen (h_i, f_i) **affin** (linear) sind,
- **Dann** reicht das Erfüllen der KKT-Bedingungen aus, um sicher zu sein: Wir haben das Optimum gefunden!

Zusammenfassung: Der “Optimierungs-Werkzeugkasten”

1. **Der Motor:** Gradient Descent (GD) nutzt die Ableitung erster Ordnung, um das Minimum zu finden.
2. **Der Turbo:** Stochastic GD (SGD) ermöglicht Training auf riesigen Datenmengen durch Approximation.
3. **Das Momentum :** Heavy Ball hilft über flache Täler und aus lokalen Minima heraus.
4. **Die Regeln:** Lagrange und KKT erlauben uns, harte Nebenbedingungen mathematisch exakt einzuhalten.

Vielen Dank für die Aufmerksamkeit

Fragen?

“Optimization is the bridge between data and intelligence.”