Moderne Methoden der Datenanalyse
– Ereignisklassifikation –

Roger Wolf

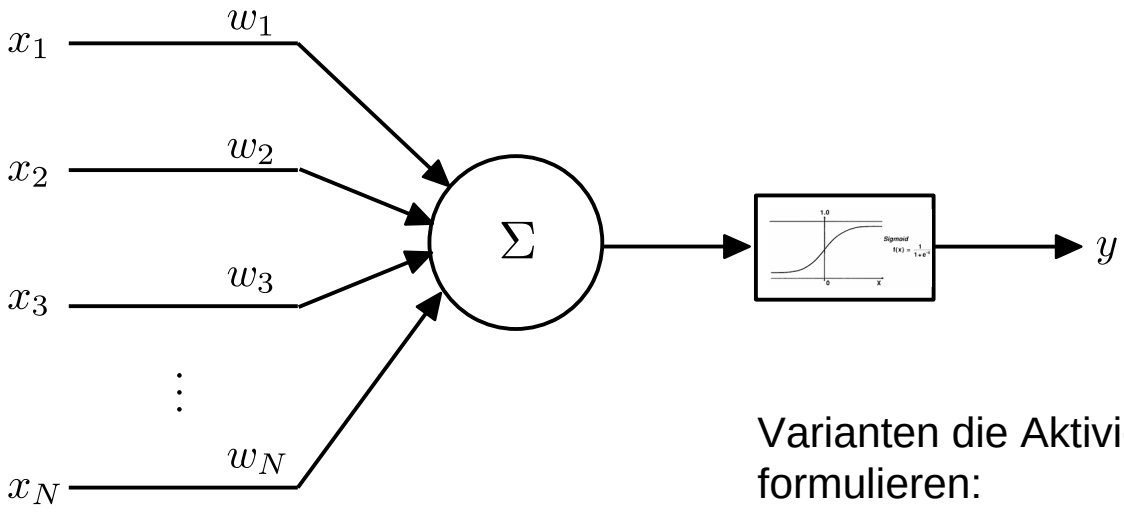
09. Juli 2021

Inhalt der Vorlesung

- Training als Optimierungsproblem.
- Statistisches Modell, Verlustfunktion.
- Gradientenabstiegsverfahren (*gradient descent*):
  - Schrittweite (*learning rate*).
  - Newton-Verfahren, Momentum methods, Nestorov-Verfahren.
  - Forward Pass.
  - Backward Pass.

Übergang zur kontinuierlichen Aktivierungsfunktion

- In der letzten Vorlesung haben wir den Übergang von einer einfachen Stufenfunktion zu einer kontinuierlichen Aktivierungsfunktion diskutiert:



$x_1 \dots x_N \in \mathbb{R}$

$w_1 \dots w_N \in \mathbb{R}$

Einheit feuert, wenn $\sum_i w_i x_i \geq T$

Varianten die Aktivierungslogik zu formulieren:

$$\sum_i w_i x_i \geq T \,,$$

$$\sum_i w_i x_i - T \geq 0 \,,$$

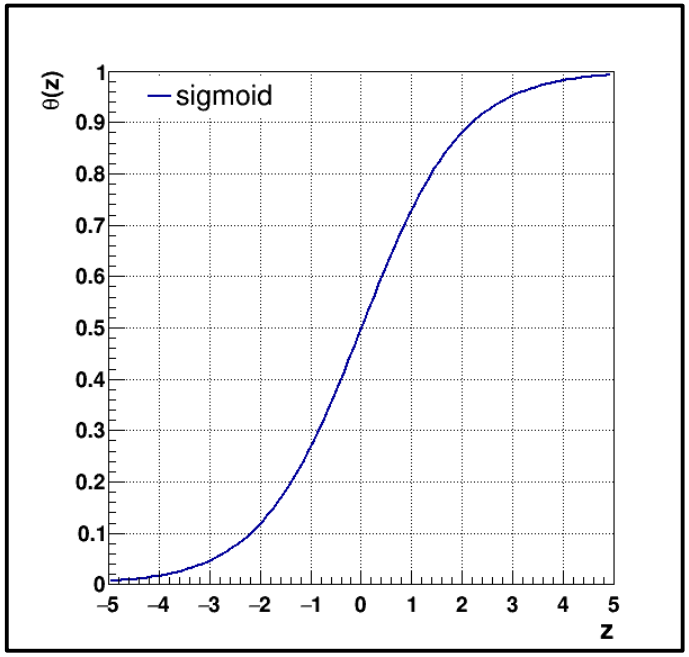
$$\theta\left(\sum_i w_i x_i - T\right)$$

Wir wählen als konkretes Beispiel im folgenden die Sigmoidfunktion.

Sigmoidfunktion

- Die Sigmoidfunktion (oder logistische Funktion) ist eine gängige Aktivierungsfunktion für Perceptrons:

$$\theta(z) = \frac{1}{1+\exp(-z)} = \frac{1}{2}\left(1+\tanh\left(\frac{z}{2}\right)\right) \quad ; \quad \frac{\mathrm{d}\theta}{\mathrm{d}z} = \theta(z)\left(1-\theta(z)\right)$$



- Bildet $\mathbb{R}$ auf $(0,1)$ ab.
- Entspricht stetigem Schwellenübergang.
- Dient zur Modellierung von Saturations-prozessen in der Statistik.
- Erlaubt Interpretation als bedingte Wahr-scheinlichkeitsdichte (siehe VL-11 Folie 30).

Training als Optimierungsproblem

- Durch die Verwendung differenzierbarer Aktivierungsfunktionen wird die MLP output Funktion $y\left(\{x_k\};\{w_j\}\right)$ ebenfalls differenzierbar in jeder Variablen.

Training als Optimierungsproblem

- Durch die Verwendung differenzierbarer Aktivierungsfunktionen wird die MLP output Funktion $y(\{x_k\};\{w_j\})$ ebenfalls differenzierbar in jeder Variablen.

- Die Anpassung der Gewichte des MLP $\{w_j\}$ an die Targetfunktion/kontur wird so zu einem multidimensionalen **Optimierungsproblem**.

Training als Optimierungsproblem

- Durch die Verwendung differenzierbarer Aktivierungsfunktionen wird die MLP output Funktion $y(\{x_k\};\{w_j\})$ ebenfalls differenzierbar in jeder Variablen.

- Die Anpassung der Gewichte des MLP $\{w_j\}$ an die Targetfunktion/kontur wird so zu einem multidimensionalen **Optimierungsproblem**.

  - Die Dimension dieses Problems ist i.a. sehr hoch.

Training als Optimierungsproblem

- Durch die Verwendung differenzierbarer Aktivierungsfunktionen wird die MLP output Funktion $y\left(\{x_k\};\{w_j\}\right)$ ebenfalls differenzierbar in jeder Variablen.

- Die Anpassung der Gewichte des MLP $\{w_j\}$ an die Targetfunktion/kontur wird so zu einem multidimensionalen **Optimierungsproblem**.

  - Die Dimension dieses Problems ist i.a. sehr hoch.

  - Die Targetfunktion/kontur (der Grundgesamtheit) ist i.a. nicht bekannt und/oder analytisch nicht darstellbar (*non-tractable*).

Training als Optimierungsproblem

- Durch die Verwendung differenzierbarer Aktivierungsfunktionen wird die MLP output Funktion $y(\{x_k\};\{w_j\})$ ebenfalls differenzierbar in jeder Variablen.

- Die Anpassung der Gewichte des MLP $\{w_j\}$ an die Targetfunktion/kontur wird so zu einem multidimensionalen **Optimierungsproblem**.

  - Die Dimension dieses Problems ist i.a. sehr hoch.

  - Die Targetfunktion/kontur (der Grundgesamtheit) ist i.a. nicht bekannt und/oder analytisch nicht darstellbar (*non-tractable*).

- Die Übereinstimmung von $y(\{x_k\};\{w_j\})$ mit der Targetfunktion/kontur wird durch die **Verlustfunktion** L (*loss function*) quantifiziert. Diese sollte ebenfalls so gewählt werden, dass sie differenzierbar ist.

Training als Optimierungsproblem

- Durch die Verwendung differenzierbarer Aktivierungsfunktionen wird die MLP output Funktion $y(\{x_k\};\{w_j\})$ ebenfalls differenzierbar in jeder Variablen.

- Die Anpassung der Gewichte des MLP $\{w_j\}$ an die Targetfunktion/kontur wird so zu einem multidimensionalen **Optimierungsproblem**.

  - Die Dimension dieses Problems ist i.a. sehr hoch.

  - Die Targetfunktion/kontur (der Grundgesamtheit) ist i.a. nicht bekannt und/oder analytisch nicht darstellbar (*non-tractable*).

- Die Übereinstimmung von $y(\{x_k\};\{w_j\})$ mit der Targetfunktion/kontur wird durch die **Verlustfunktion** L (*loss function*) quantifiziert. Diese sollte ebenfalls so gewählt werden, dass sie differenzierbar ist.

- **NB**: Die Verlustfunktion enthält i.a. die Targetfunktion/kontur garnicht, d.h. dass das Training nicht auf die Targetfunktion/kontur direkt erfolgen muss, was i.a. auch nicht der Fall ist.

Supervised learning

- Wir werden im Rahmen dieser Vorlesung das *supervised learning* diskutieren, d.h. für das *training* steht dem MLP die folgende Information für jedes Elementarereignis zur Verfügung:

Supervised learning

- Wir werden im Rahmen dieser Vorlesung das *supervised learning* diskutieren, d.h. für das *training* steht dem MLP die folgende Information für jedes Elementarereignis zur Verfügung:
  - Die Eingabewerte $\{x_k\}$ (*input features*).

Supervised learning

- Wir werden im Rahmen dieser Vorlesung das *supervised learning* diskutieren, d.h. für das *training* steht dem MLP die folgende Information für jedes Elementarereignis zur Verfügung:
  - Die Eingabewerte $\{x_k\}$ (*input features*).
  - Die wahre Zuordnung (Klassifikation)/der wahre Wert (Regression), auch *label* genannt.

Supervised learning

- Wir werden im Rahmen dieser Vorlesung das *supervised learning* diskutieren, d.h. für das *training* steht dem MLP die folgende Information für jedes Elementarereignis zur Verfügung:
  - Die Eingabewerte $\{x_k\}$ (*input features*).
  - Die wahre Zuordnung (Klassifikation)/der wahre Wert (Regression), auch *label* genannt.
  - Elementarereignisse der Trainingsstichprobe werden auch oft intuitiv „Beispiele“ (*examples*) genannt: „Das MLP lernt nicht aus expliziten Regeln, sondern aus Beispielen“.

Supervised learning

- Wir werden im Rahmen dieser Vorlesung das *supervised learning* diskutieren, d.h. für das *training* steht dem MLP die foldende Information für jedes Elementarereignis zur Verfügung:
  - Die Eingabewerte $\{x_k\}$ (*input features*).
  - Die wahre Zuordnung (Klassifikation)/der wahre Wert (Regression), auch *label* genannt.
  - Elementarereignisse der Trainingsstichprobe werden auch oft intuitiv „Beispiele“ (*examples*) genannt: „Das MLP lernt nicht aus expliziten Regeln, sondern aus Beispielen“.

Supervised learning: *Label* bekannt für jedes Ereignis.

Unsupervised learning

- Demgegenüber steht z.B. das *unsupervised learning*:

Unsupervised learning

- Demgegenüber steht z.B. das *unsupervised learning*:
  - Hier identifiziert (und/oder reproduziert) das MLP in der späteren Anwendung Strukturen, die es in den Tainingsdaten identifiziert hat.

Unsupervised learning

- Demgegenüber steht z.B. das *unsupervised learning*:
  - Hier identifiziert (und/oder reproduziert) das MLP in der späteren Anwendung Strukturen, die es in den Tainingsdaten identifiziert hat.
  - Beispiel: Zwei Tüten Gummibärchen. In einer Tüte gibt es **grüne** Gummibärchen, in der anderen nicht. Finde die grünen Gummibärchen!

Unsupervised learning

- Demgegenüber steht z.B. das *unsupervised learning*:
  - Hier identifiziert (und/oder reproduziert) das MLP in der späteren Anwendung Strukturen, die es in den Tainingsdaten identifiziert hat.
  - Beispiel: Zwei Tüten Gummibärchen. In einer Tüte gibt es **grüne** Gummibärchen, in der anderen nicht. Finde die grünen Gummibärchen!

  - An dieser Stelle ist wichtig, dass nicht (mehr) jedes „Bärchen“ *gelabelt* ist. Das MLP muss den Unterschied zwischen beiden Tüten selbst identifizieren.

Reinforcement learning

- Eine weitere Art des Lernens:
  - Hier erhält das Netz (i.a. mit „Verzögerung“) eine Belohnung/Bestrafung für seine Entscheidungen (Bsp. Machinelles Spielen: Schach, Go, …).

Labels

- Für die Klassifikation erfolgt das *labelling* der Trainingsdaten i.a. im *one-hot encoding*:

Labels

- Für die Klassifikation erfolgt das *labelling* der Trainingsdaten i.a. im *one-hot encoding*:
  - Binäre Klassifikation *(binary classification)*:

$$p(x_k) = \begin{cases} 1 & \text{Signal} \\ 0 & \text{Untergrund} \end{cases}$$

Labels

- Für die Klassifikation erfolgt das *labelling* der Trainingsdaten i.a. im *one-hot encoding*:
  - Binäre Klassifikation *(binary classification)*:

$$p(x_k) = \begin{cases} 1 & \text{Signal} \\ 0 & \text{Untergrund} \end{cases}$$

  - Multiklassifikation *(multi-classification)*:

$$p(x_k) = \left\{ \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix} \quad \cdots \quad \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} \right. \begin{matrix} \text{Kategorie–1} \\ \text{Kategorie–2} \\ \vdots \\ \text{Kategorie–n} \end{matrix}$$

Jeweils als n-komponentiger Vektor

Labels

- Für die Klassifikation erfolgt das *labelling* der Trainingsdaten i.a. im *one-hot encoding*:
  - Binäre Klassifikation *(binary classification)*:

$$p(x_k) = \begin{cases} 1 & \text{Signal} \\ 0 & \text{Untergrund} \end{cases}$$

  - Multiklassifikation *(multi-classification)*:

$$p(x_k) = \left\{ \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix} \quad \cdots \quad \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} \right. \begin{matrix} \text{Kategorie–1} \\ \text{Kategorie–2} \\ \vdots \\ \text{Kategorie–n} \end{matrix}$$

Jeweils als n-komponentiger Vektor

- Regression:
  - Hier entspricht das *label* dem (reellwertigen) wahren Wert: bei der Kalibration eines Kalorimeters z.B. der Energie eines eingschossenen Pions.

Verlustfunktion

- Die Verlustfunktion ist eine Teststatistik zur Abschätzung der Wahrscheinlichkeitsdichte der unbekannten Grundgesamtheit. Sie ist damit eine Schätzfunktion.

Verlustfunktion

- Die Verlustfunktion ist eine Teststatistik zur Abschätzung der Wahrscheinlichkeitsdichte der unbekannten Grundgesamtheit. Sie ist damit eine Schätzfunktion.

- Häufig beim *training* von MLPs verwendete Verlustfunktionen sind:

Verlustfunktion

- Die Verlustfunktion ist eine Teststatistik zur Abschätzung der Wahrscheinlichkeitsdichte der unbekannten Grundgesamtheit. Sie ist damit eine Schätzfunktion.

- Häufig beim *training* von MLPs verwendete Verlustfunktionen sind:

Kreuzentropie (*cross-entropy*) für (Multi-)Klassifikationsprobleme:

$$H(\{p_i\},\{y_i\}) = -\sum_{i=1}^{n} p_i \log(y_i)$$

n : Länge der Stichprobe

p_i : Label für Beispiel i

y_i : MLP output für Beispiel i

Verlustfunktion

- Die Verlustfunktion ist eine Teststatistik zur Abschätzung der Wahrscheinlichkeitsdichte der unbekannten Grundgesamtheit. Sie ist damit eine Schätzfunktion.

- Häufig beim *training* von MLPs verwendete Verlustfunktionen sind:

Kreuzentropie (*cross-entropy*) für (Multi-)Klassifikationsprobleme:

$$H(\{p_i\},\{y_i\}) = -\sum_{i=1}^{n} p_i \log(y_i)$$

n : Länge der Stichprobe

p_i : Label für Beispiel i

y_i : MLP output für Beispiel i

Mittleres Betragsquadrat (Mean Squared Error, MSE) für Regressionsprobleme:

$$\mathrm{MSE}(\{p_i\},\{y_i\}) = -\frac{1}{n}\sum_{i=1}^{n} (p_i - y_i)^2$$

n : Länge der Stichprobe

p_i : Wahrer Wert für Beispiel i

y_i : MLP output für Beispiel i

Verlustfunktion

- Die Verlustfunktion ist eine Teststatistik zur Abschätzung der Wahrscheinlichkeitsdichte der unbekannten Grundgesamtheit. Sie ist damit eine Schätzfunktion.

- Häufig beim *training* von MLPs verwendete Verlustfunktionen sind:

Kreuzentropie (*cross-entropy*) für (Multi-)Klassifikationsprobleme:

$$H(\{p_i\},\{y_i\}) = -\sum_{i=1}^{n} p_i \log(y_i)$$

n : Länge der Stichprobe

p_i : Label für Beispiel i

y_i : MLP output für Beispiel i

Maximum Likelihood Abschätzung für die labels $\{p_i\}$ für multinomial verteilte MLP Vorhersagen $\{y_i\}$.

Mittleres Betragsquadrat (Mean Squared Error, MSE) für Regressionsprobleme:

$$\mathrm{MSE}(\{p_i\},\{y_i\}) = -\frac{1}{n}\sum_{i=1}^{n} (p_i - y_i)^2$$

n : Länge der Stichprobe

p_i : Wahrer Wert für Beispiel i

y_i : MLP output für Beispiel i

Maximum Likelihood Abschätzung für die labels $\{p_i\}$ für normalverteilte MLP Vorhersagen $\{y_i\}$.

Kreuzentropie (binäre Klassifikation)

Wahrscheinlichkeit für Signal
bei Kenntnis der Wahrheit

$$p_i(x_k) = \begin{cases} 1 & \text{Signal} \\ 0 & \text{Untergrund} \end{cases}$$

Wahrscheinlichkeit für Untergrund
bei Kenntnis der Wahrheit

$$1 - p_i(x_k)$$

MLP-Abschätzung für $p_i(x_k)$

$$y_i(x_k)$$

MLP-Abschätzung für $1 - p_i(x_k)$

$$1 - y_i(x_k)$$

$$H(\{p_i\}, \{y_i\}) = -\sum_{i=1}^{n} \left(p_i \log(y_i) + (1 - p_i) \log(1 - y_i) \right)$$

Kreuzentropie (binäre Klassifikation)

Wahrscheinlichkeit für Signal bei Kenntnis der Wahrheit

$$p_i(x_k) = \begin{cases} 1 & \text{Signal} \\ 0 & \text{Untergrund} \end{cases}$$

Wahrscheinlichkeit für Untergrund bei Kenntnis der Wahrheit

$$1 - p_i(x_k)$$

MLP-Abschätzung für $p_i(x_k)$

$$y_i(x_k)$$

MLP-Abschätzung für $1 - p_i(x_k)$

$$1 - y_i(x_k)$$

$$H(\{p_i\},\{y_i\}) = -\sum_{i=1}^{n} \left(\underbrace{p_i \log(y_i)}_{\text{Richtig identifiziertes Signal}} + \underbrace{(1-p_i)\log(1-y_i)}_{\text{Richtig identifizierter Untergrund}} \right)$$

- Die Kreuzentropie quantifiziert beim *one-hot encoding* nur die richtig identifizierten Beispiele (vgl. mit Lernregel von Rosenblatt).

Binomialverteilung

- Wahrscheinlichkeit eines Bernoulli-Prozesses vom MLP als Signal oder Untergrund identifiziert zu werden:

$$P(\{y_i\},\{w_j\}) = y_i^{p_i}(1-y_i(x_k))^{1-p_i} = \begin{cases} y_i & \text{Signal} \\ 1-y_i & \text{Untergrund} \end{cases}$$

- Likelihood für n Bernoulli-Prozesse (unter Berücksichtigung der Reihenfolge):

$$\begin{aligned} \mathcal{L}(\{w_i\}) &= \prod_{i=1}^{n} P(\{y_i\},\{w_j\}) \\ &= \prod_{i=1}^{n} y_i^{p_i}(1-y_i(x_k))^{1-p_i}; \end{aligned}$$

$$\log(\mathcal{L}(\{w_i\})) = \underbrace{\sum_{i=1}^{n}\Big(p_i\log(y_i)+(1-p_i)\log(1-y_i)\Big)}_{\equiv -H(\{y_i\},\{p_i\})}$$

Maximum Likelihood Abschätzung für die labels $\{p_i\}$ für multinomial verteilte MLP Vorhersagen $\{y_i\}$.

Anmerkungen zur Verlustfunktion

- Kreuzentropie und MSE sind zwar häufig verwendete Verlustfunktionen. Die Wahl der Verlustfunktion steht Ihnen jedoch **frei**!

- Die Verlustfunktion wird in beiden Fällen auf dem Umfang n einer Stichprobe bestimmt.

- Die Verlustfunktion ist eine reellwertige Funktion auf einem hochdimensionalen Definitionsbereich.(1)

$$L : \Omega^k \longrightarrow \mathbb{R} \quad ; \quad \vec{x} \longrightarrow L(\vec{x}, \vec{w})$$

(1)

Diskrete *features* müssen beim training gesondert behandelt werden. Das werden wir im Rahmen dieser Vorlesung nicht weiter diskutieren können.

Minimierung der Verlustfunktion

- In einem hochdimensionalen Definitionsbereich steht der Gradient ∇L senkrecht auf die Tangentialebene zur Äquipotentialkontur der Verlustfunktion L und weist in Richtung des größten Anstieges von L. (Das müssen wir unter Physikern nicht weiter diskutieren.)

- Für das Minimum von L gilt:

$$\nabla_{\vec{w}} L = 0 \quad ; \quad H_{ij} = \left[\frac{\partial^2}{\partial w_i \partial w_j} L \right] \ (\text{positiv definit})$$

Gradientenabstiegsverfahren

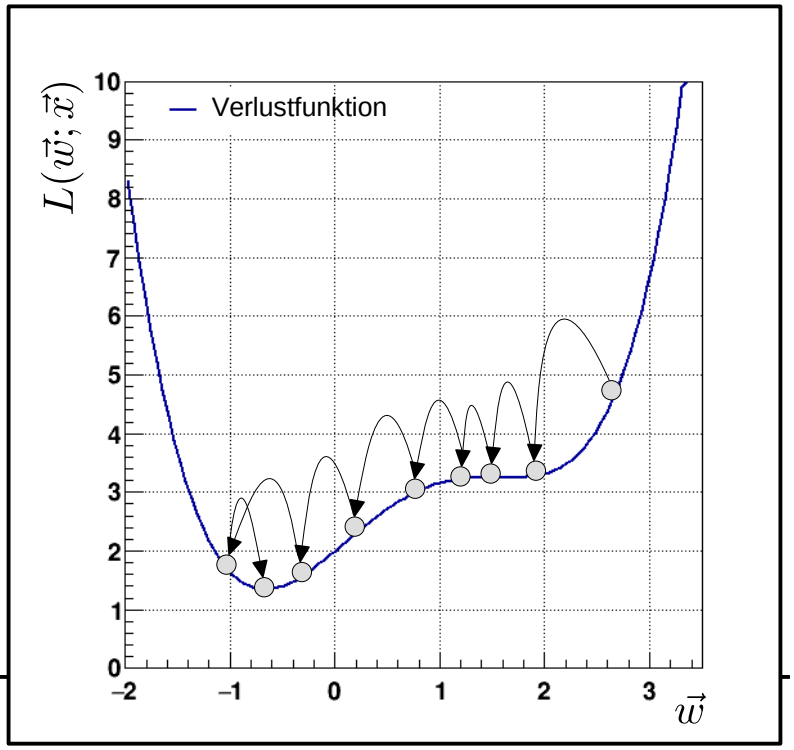
- Ein etabliertes numerisches Verfahren, um das Minimum zu finden besteht im Gradienten-abstieg:
  - Initialisiere Gewichte $\vec{w}^{(0)}$ zufällig.
  - Bilde den Gradienten im Punkt $\vec{w}^{(k)}$.

Aktualisiere wie folg:

$$\vec{w}^{(k+1)} = \vec{w}_k - \eta \nabla_{\vec{w}^{(k)}} L, \quad \eta > 0$$

solange:

$$|L(x_k) - L(x_{k-1})| > \epsilon$$



Gradientenabstiegsverfahren

- Ein etabliertes numerisches Verfahren, um das Minimum zu finden besteht im Gradienten-abstieg:
  - Initialisiere Gewichte $\vec{w}^{(0)}$ zufällig.
  - Bilde den Gradienten im Punkt $\vec{w}^{(k)}$.

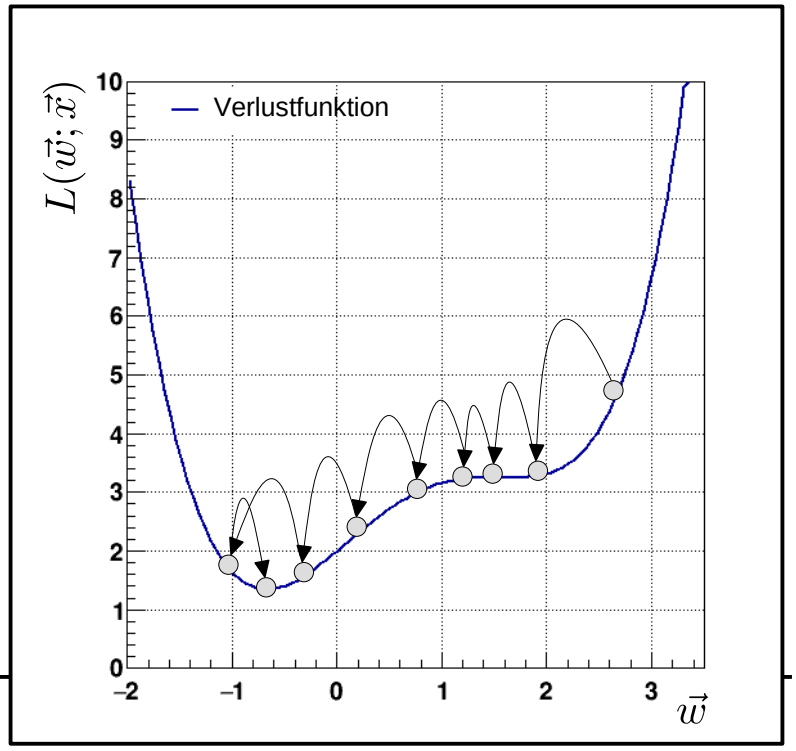
Aktualisiere wie folg:

$$\vec{w}^{(k+1)} = \vec{w}_k - \eta \nabla_{\vec{w}^{(k)}} L, \quad \eta > 0$$

solange:

$$|L(x_k) - L(x_{k-1})| > \epsilon$$

d.h. in Gegenrichtung des größten Anstiegs.



Gradientenabstiegsverfahren

- Ein etabliertes numerisches Verfahren, um das Minimum zu finden besteht im Gradienten-abstieg:
  - Initialisiere Gewichte $\vec{w}^{(0)}$ zufällig.
  - Bilde den Gradienten im Punkt $\vec{w}^{(k)}$.

Man bezeichnet η als **Schrittweite/Lernrate**.

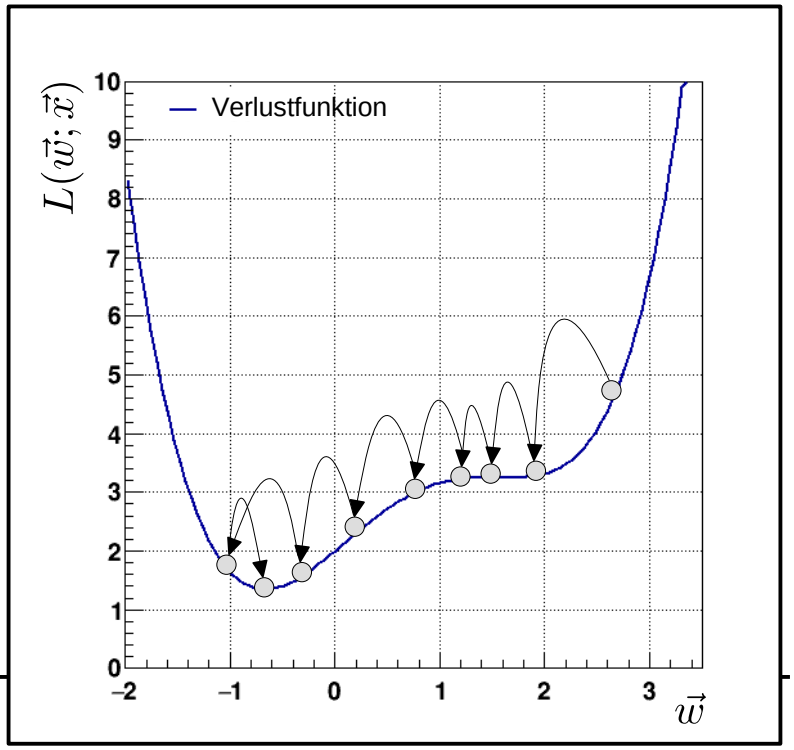
Aktualisiere wie folg:

$$\vec{w}^{(k+1)} = \vec{w}_k - \eta \nabla_{\vec{w}^{(k)}} L, \quad \eta > 0$$

solange:

$$|L(x_k) - L(x_{k-1})| > \epsilon$$

d.h. in Gegenrichtung des größten Anstiegs.

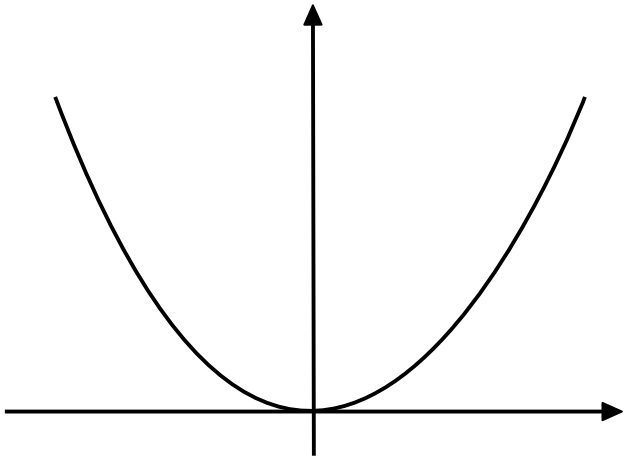


Optimale Schrittweite

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$

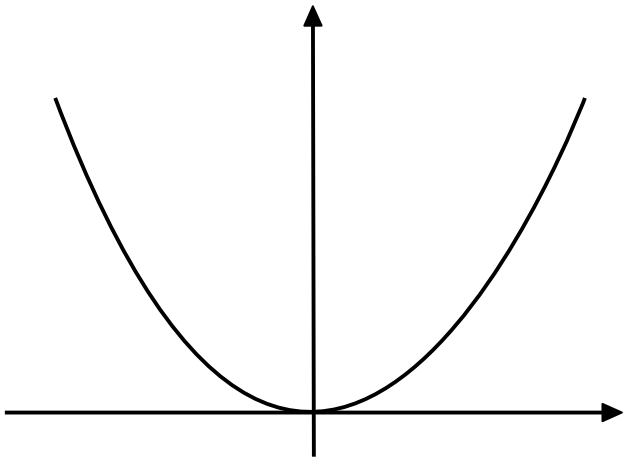
- Wie würden Sie die Schrittweite η wählen, um das Minimum der Funktion möglichst schnell zu erreichen und nach wieviel Schritten ist dies der Fall?



Optimale Schrittweite

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$



- Wie würden Sie die Schrittweite η wählen, um das Minimum der Funktion möglichst schnell zu erreichen und nach wieviel Schritten ist dies der Fall?

- Taylorreihe um einen beliebigen Wert x_0:

$$F\,(x) = F(x_0) + F'(x_0)\,(x - x_0) + \frac{1}{2}F''(x_0)(x_0)\,(x - x_0)^2$$

$$F'(x) = F'(x_0) + F''(x_0)\,(x - x_0) = 0$$

$$x_{\min} \quad = x_0 - \frac{F'(x_0)}{F''(x_0)}$$

Vergleich mit Aktualisierungsregel:

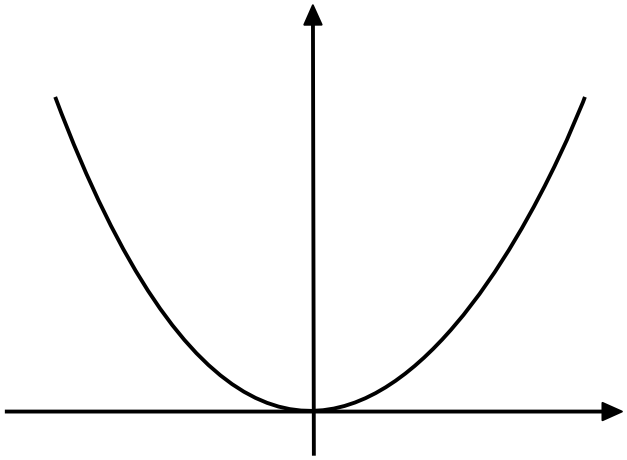
$$x_1 = x_0 - \eta F'(x_0) \qquad \Rightarrow \qquad \eta_{\text{opt}} = \frac{1}{F''(x_0)} \quad \text{mit: } F''(x) = a \quad (\forall x \in \mathbb{R})$$

Optimale Schrittweite

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$

- Wie würden Sie die Schrittweite η wählen, um das Minimum der Funktion möglichst schnell zu erreichen und nach wieviel Schritten ist dies der Fall?



- Taylorreihe um einen beliebigen Wert x_0:

$$F\,(x) = F(x_0) + F'(x_0)\,(x - x_0) + \frac{1}{2}F''(x_0)(x_0)\,(x - x_0)^2$$

$$F'(x) = F'(x_0) + F''(x_0)\,(x - x_0) = 0$$

$$x_{\min} = x_0 - \frac{F'(x_0)}{F''(x_0)}$$

D.h. Sie erreichen das Minimum einer Parabel immer mit einem Schritt, egal von wo Sie die Minimierung starten.

Vergleich mit Aktualisierungsregel:

$$x_1 = x_0 - \eta F'(x_0) \qquad \Rightarrow \qquad \eta_{\text{opt}} = \frac{1}{F''(x_0)} \quad \text{mit: } F''(x) = a \quad (\forall x \in \mathbb{R})$$

Diskussion Konvergenzverhalten

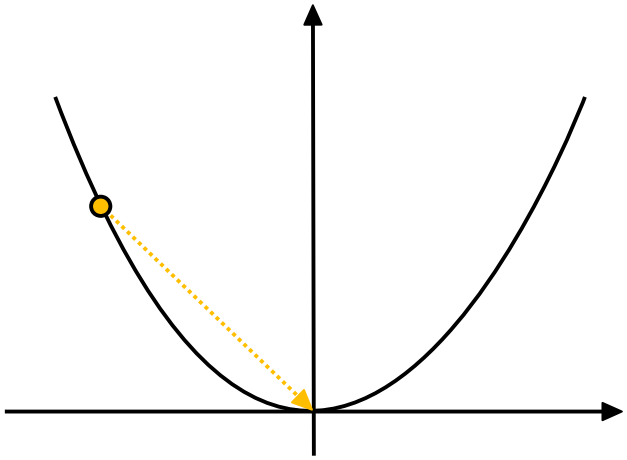
- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$

- Wir untersuchen im folgenden das Konvergenzverhalten für verschiedene Werte von η :

$$(1) \quad \eta = \frac{1}{F''(x)}$$

} Konvergenz nach einem Schritt



Diskussion Konvergenzverhalten

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

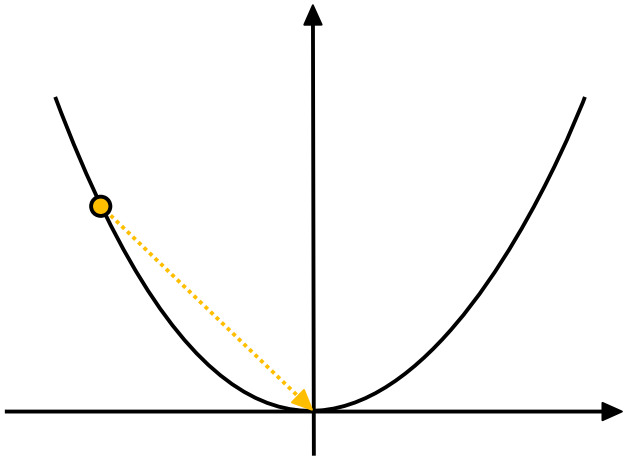
$$F(x) = \frac{1}{2}ax^2 + bx + c$$

- Wir untersuchen im folgenden das Konvergenzverhalten für verschiedene Werte von η :

$$(1) \quad \eta = \frac{1}{F''(x)}$$

$$(2) \quad \eta < \frac{1}{F''(x)}$$

} Konvergenz nach einem Schritt



Diskussion Konvergenzverhalten

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

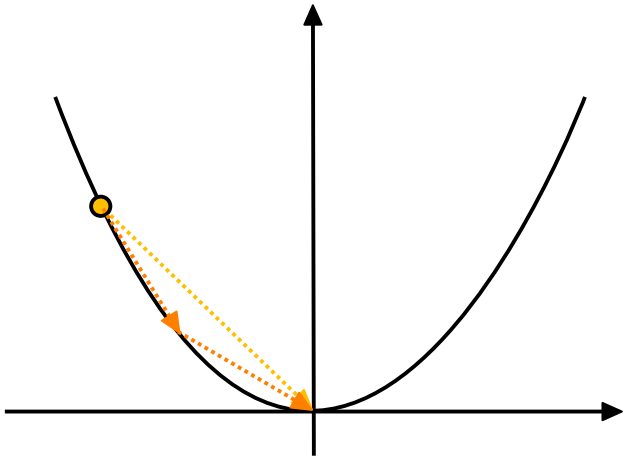
$$F(x) = \frac{1}{2}ax^2 + bx + c$$

- Wir untersuchen im folgenden das Konvergenzverhalten für verschiedene Werte von η :

$$(1) \quad \eta = \frac{1}{F''(x)}$$

$$(2) \quad \eta < \frac{1}{F''(x)}$$

} Konvergenz nach einem Schritt



Diskussion Konvergenzverhalten

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$

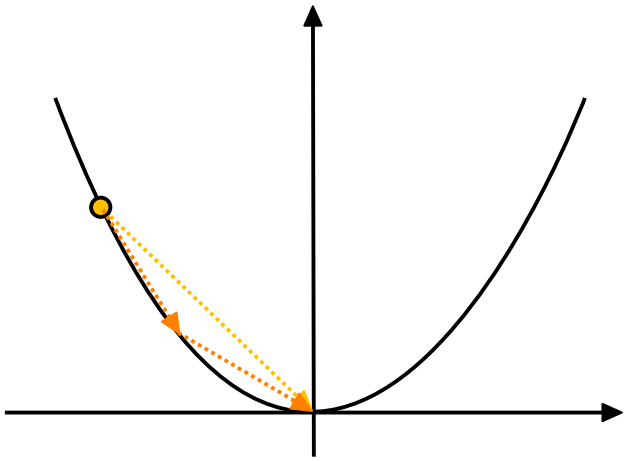
- Wir untersuchen im folgenden das Konvergenzverhalten für verschiedene Werte von η :

$$(1) \quad \eta = \frac{1}{F''(x)}$$

$$(2) \quad \eta < \frac{1}{F''(x)}$$

$$(3) \quad \eta > \frac{1}{F''(x)}$$

} Konvergenz nach einem Schritt



Diskussion Konvergenzverhalten

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

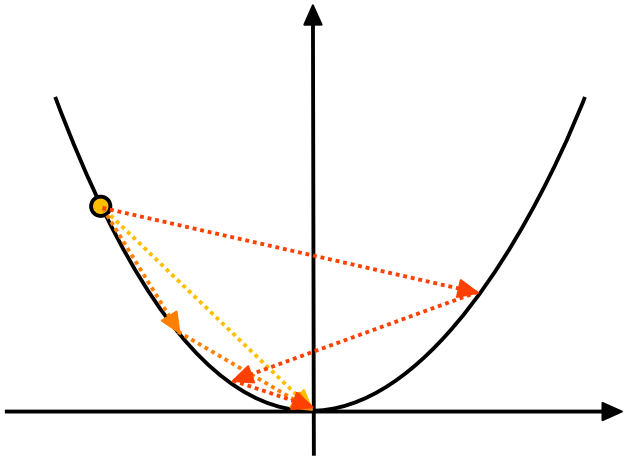
$$F(x) = \frac{1}{2}ax^2 + bx + c$$

- Wir untersuchen im folgenden das Konvergenzverhalten für verschiedene Werte von η :

(1) $\eta = \frac{1}{F''(x)}$ } Konvergenz nach einem Schritt

(2) $\eta < \frac{1}{F''(x)}$ } Konvergenz nach mehr als einem Schritt (t.w. oszillierend

(3) $\eta > \frac{1}{F''(x)}$ }



Diskussion Konvergenzverhalten

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$

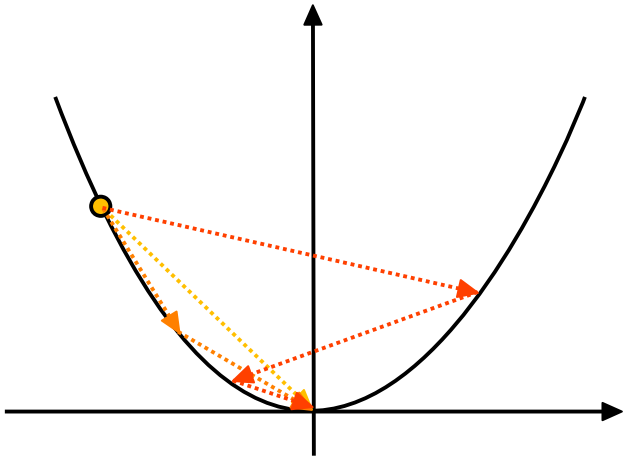
- Wir untersuchen im folgenden das Konvergenzverhalten für verschiedene Werte von η :

(1) $\eta = \dfrac{1}{F''(x)}$ } Konvergenz nach einem Schritt

(2) $\eta < \dfrac{1}{F''(x)}$ } Konvergenz nach mehr als einem Schritt (t.w. oszillierend

(3) $\eta > \dfrac{1}{F''(x)}$ }

(4) $\eta = 2\dfrac{1}{F''(x)}$

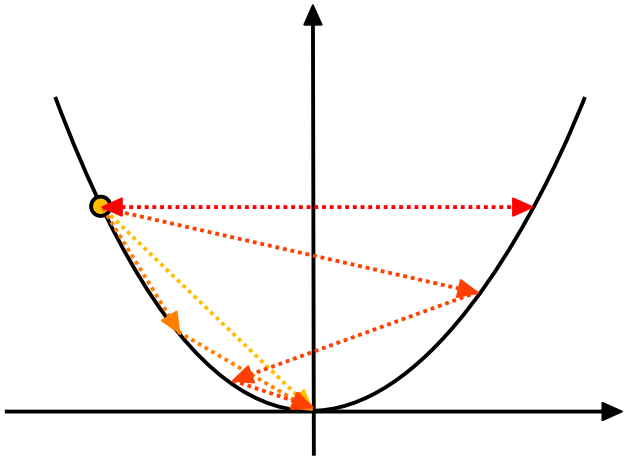


Diskussion Konvergenzverhalten

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$

- Wir untersuchen im folgenden das Konvergenzverhalten für verschiedene Werte von η :



(1) $\eta = \frac{1}{F''(x)}$ } Konvergenz nach einem Schritt

(2) $\eta < \frac{1}{F''(x)}$ } Konvergenz nach mehr als einem Schritt (t.w. oszillierend

(3) $\eta > \frac{1}{F''(x)}$ }

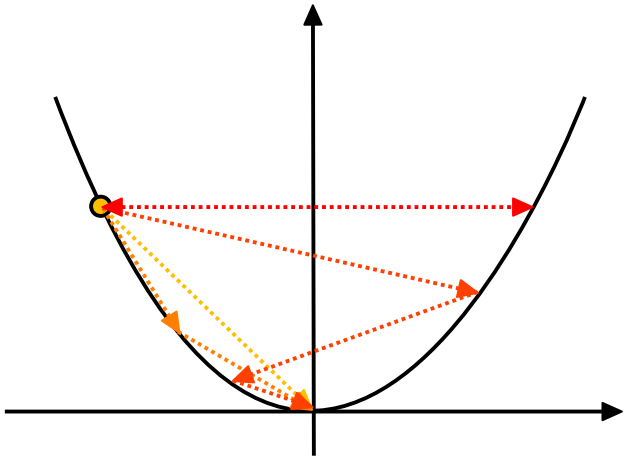
(4) $\eta = 2\frac{1}{F''(x)}$ } Stabile Oszillation

Diskussion Konvergenzverhalten

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$

- Wir untersuchen im folgenden das Konvergenzverhalten für verschiedene Werte von η :



(1) $\eta = \frac{1}{F''(x)}$ } Konvergenz nach einem Schritt

(2) $\eta < \frac{1}{F''(x)}$

(3) $\eta > \frac{1}{F''(x)}$ } Konvergenz nach mehr als einem Schritt (t.w. oszillierend

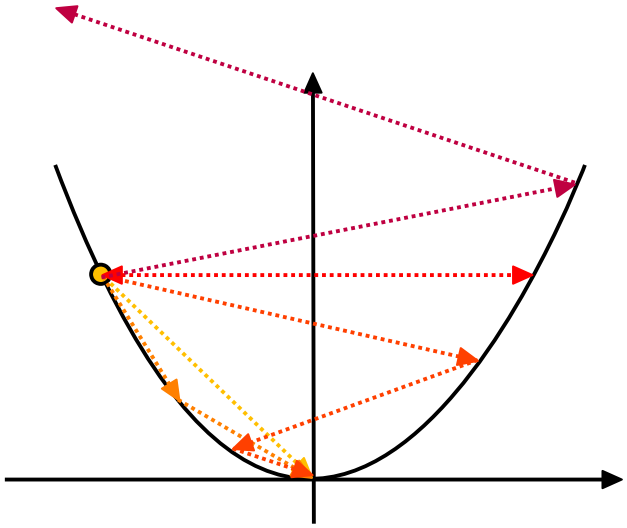
(4) $\eta = 2\frac{1}{F''(x)}$ } Stabile Oszillation

(5) $\eta > 2\frac{1}{F''(x)}$

Diskussion Konvergenzverhalten

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$



- Wir untersuchen im folgenden das Konvergenzverhalten für verschiedene Werte von η :

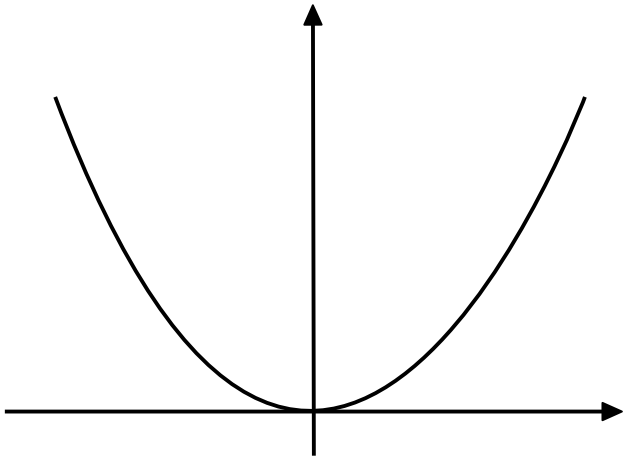
(1)	$\eta = \frac{1}{F''(x)}$	Konvergenz nach einem Schritt
(2)	$\eta < \frac{1}{F''(x)}$	Konvergenz nach mehr als einem Schritt (t.w. oszillierend
(3)	$\eta > \frac{1}{F''(x)}$	
(4)	$\eta = 2\frac{1}{F''(x)}$	Stabile Oszillation
(5)	$\eta > 2\frac{1}{F''(x)}$	Divergenz

Newtonverfahren

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$

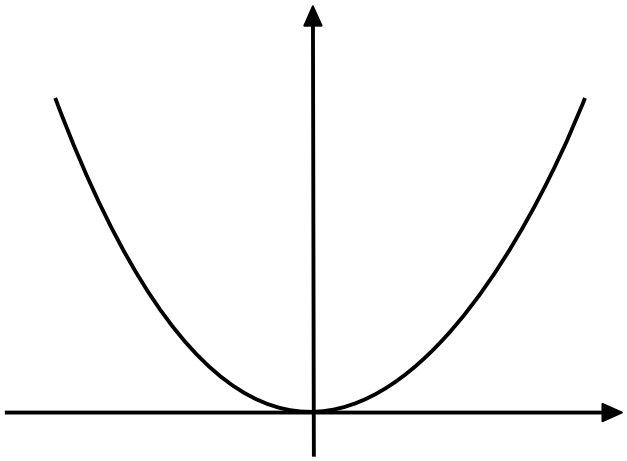
- Die Diskussion anhand von Polynomen zweiter Ordnung lässt sich auf beliebige zweifach differenzierbare Funktionen übertragen, solange man sie (nur) bis zur zweiten Ordnung in der Taylorreihe approximiert.



Newtonverfahren

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$



- Die Diskussion anhand von Polynomen zweiter Ordnung lässt sich auf beliebige zweifach differenzierbare Funktionen übertragen, solange man sie (nur) bis zur zweiten Ordnung in der Taylorreihe approximiert.

- Die Aktualisierungsregel des Gradientenabstiegs nimmt dabei die folgende Form an:

$$x_{k+1} = x_k - \frac{F'(x_k)}{F''(x_k)}$$

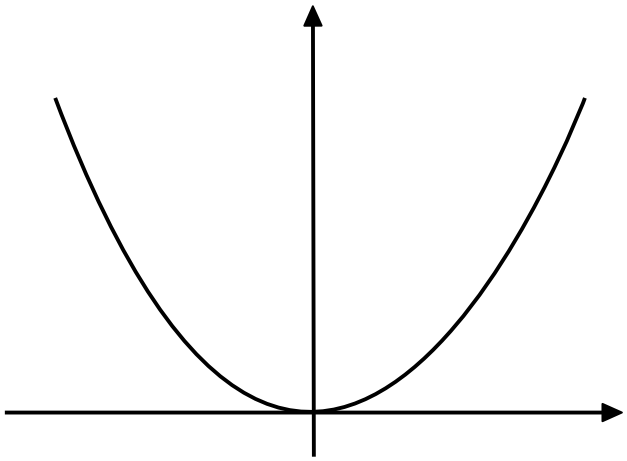
solange:

$$|F(x_k) - F(x_{k-1})| > \epsilon$$

Newtonverfahren

- Zur Diskussion der optimalen Schrittweite betrachten wir ein Polynom zweiter Ordnung:

$$F(x) = \frac{1}{2}ax^2 + bx + c$$



- Die Diskussion anhand von Polynomen zweiter Ordnung lässt sich auf beliebige zweifach differenzierbare Funktionen übertragen, solange man sie (nur) bis zur zweiten Ordnung in der Taylorreihe approximiert.

- Die Aktualisierungsregel des Gradientenabstiegs nimmt dabei die folgende Form an:

$$x_{k+1} = x_k - \frac{F'(x_k)}{F''(x_k)}$$

solange:

$$|F(x_k) - F(x_{k-1})| > \epsilon$$

Hierbei handelt es sich um das bekannte Newtonverfahren zur Nullstellenbestimmung angewandt auf $F'(x)$.

Newtonverfahren in d Dimensionen

- Ein allgemeines Polynom zweiter Ordnung in d Dimensionen hat die Form:

$$F(\vec{x}) = \frac{1}{2}\vec{x}A\vec{x}^{\intercal} + \vec{b}^{\intercal}\vec{x} + \vec{c}$$

- In der Aktualisierungsregel des Gradientenabstiegs wird dann die zweite Ableitung durch die **Hessematrix** ersetzt $H(\vec{x})$.

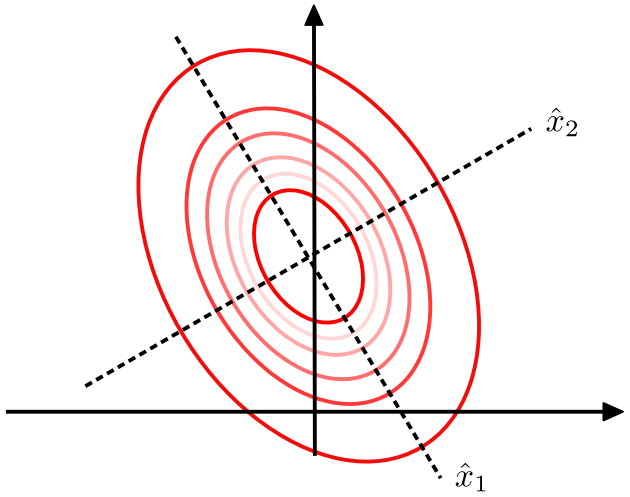
$$\vec{x}_{k+1} = \vec{x}_k - A^{-1}\vec{\nabla}F(x_k) = \vec{x}_k - H^{-1}(\vec{x}_k)\vec{\nabla}F(x_k)$$

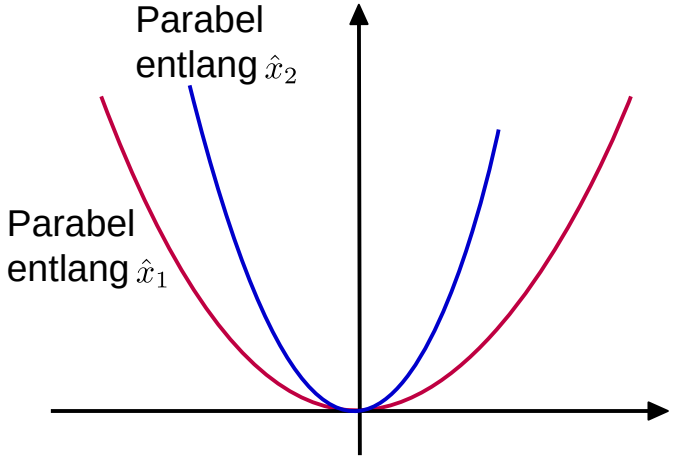
solange:

$$|F(\vec{x}_k) - F(\vec{x}_{k-1})| > \epsilon$$

Gradientenabstieg in d Dimensionen

- In d Dimensionen lässt sich jede Parabel auf ihre Hauptachsen transformieren. Schnitte entlang der Hauptachsen führen wiederum auf Parabeln.

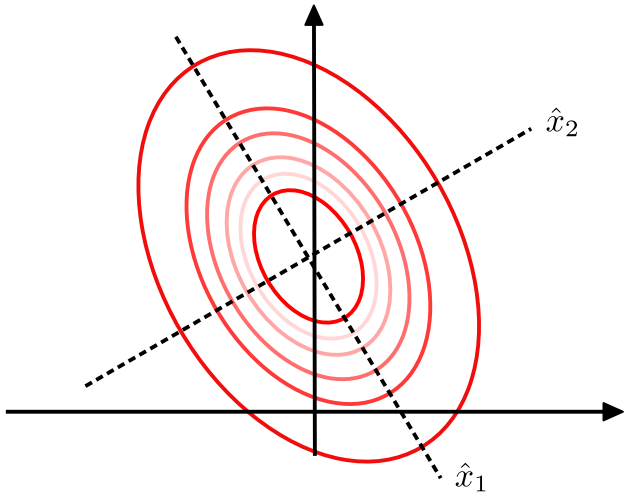


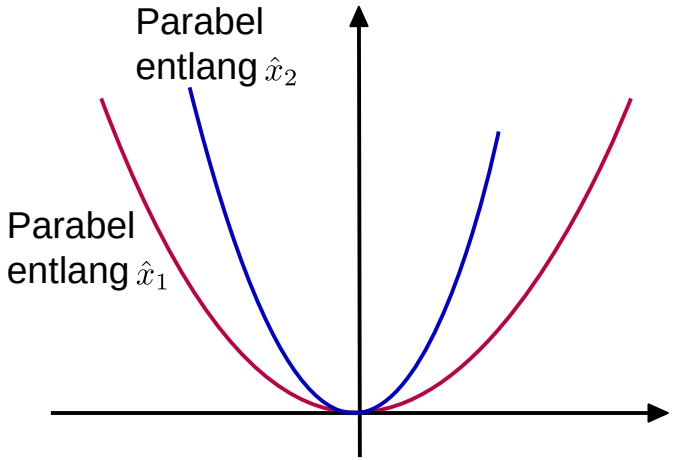


Gradientenabstieg in d Dimensionen

- In d Dimensionen lässt sich jede Parabel auf ihre Hauptachsen transformieren. Schnitte entlang der Hauptachsen führen wiederum auf Parabeln.

- **Problem**: Die Öffnungswinkel der Parabeln entlang der Hauptachsen – und damit verbunden die zweiten Richtungsableitungen entlang der Hauptachsen – sind i.a. unterschiedlich groß.



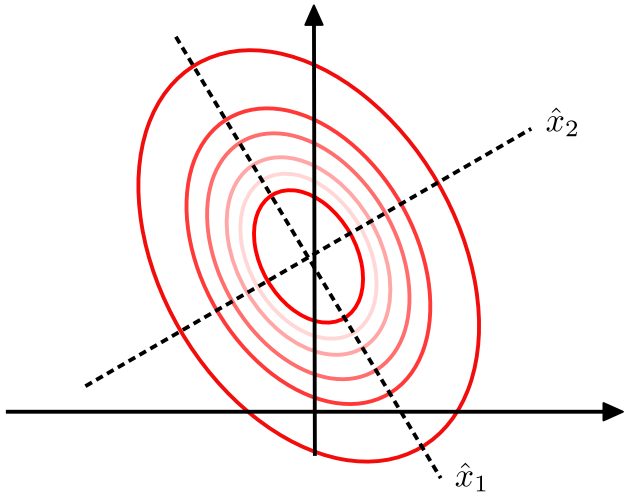


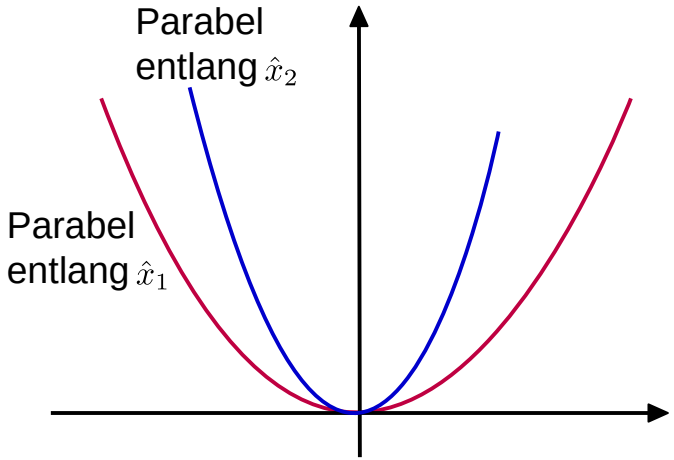
Gradientenabstieg in d Dimensionen

- In d Dimensionen lässt sich jede Parabel auf ihre Hauptachsen transformieren. Schnitte entlang der Hauptachsen führen wiederum auf Parabeln.

- **Problem**: Die Öffnungswinkel der Parabeln entlang der Hauptachsen – und damit verbunden die zweiten Richtungsableitungen entlang der Hauptachsen – sind i.a. unterschiedlich groß.

- Wie ist η zu wählen, um optimale und garantierte Konvergenz zu erreichen?





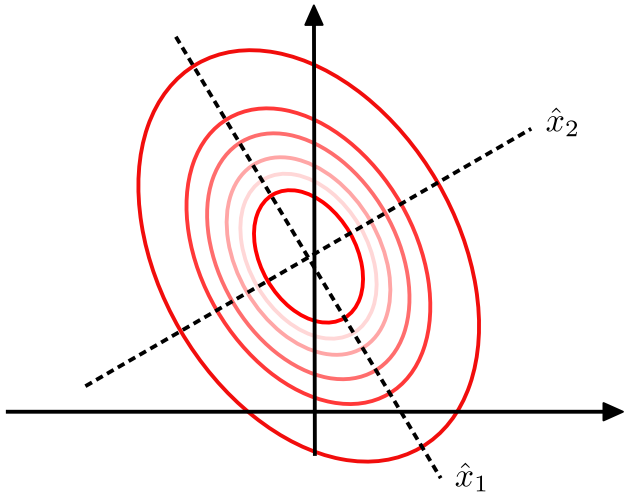
Gradientenabstieg in d Dimensionen

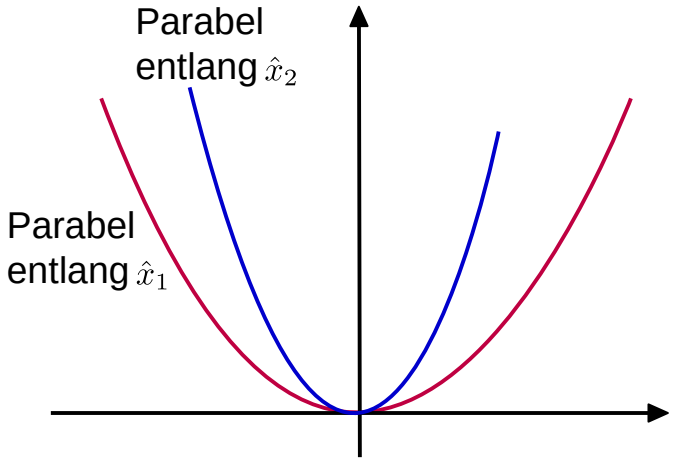
- In d Dimensionen lässt sich jede Parabel auf ihre Hauptachsen transformieren. Schnitte entlang der Hauptachsen führen wiederum auf Parabeln.

- **Problem**: Die Öffnungswinkel der Parabeln entlang der Hauptachsen – und damit verbunden die zweiten Richtungsableitungen entlang der Hauptachsen – sind i.a. unterschiedlich groß.

- Wie ist η zu wählen, um optimale und garantierte Konvergenz zu erreichen?

- Je höherdimensional der Definitionsbereich von $F(x)$ ist desto schwieriger lässt sich diese Frage im Rahmen des naiven Gradientenabstiegverfahrens beantworten.





Gradientenabstieg in d Dimensionen

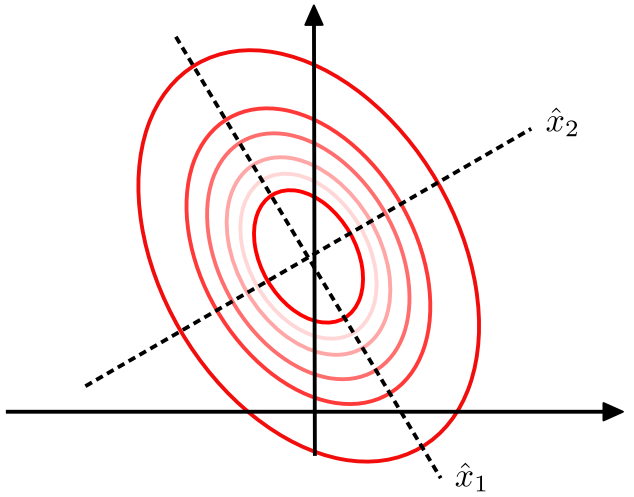
- In d Dimensionen lässt sich jede Parabel auf ihre Hauptachsen transformieren. Schnitte entlang der Hauptachsen führen wiederum auf Parabeln.

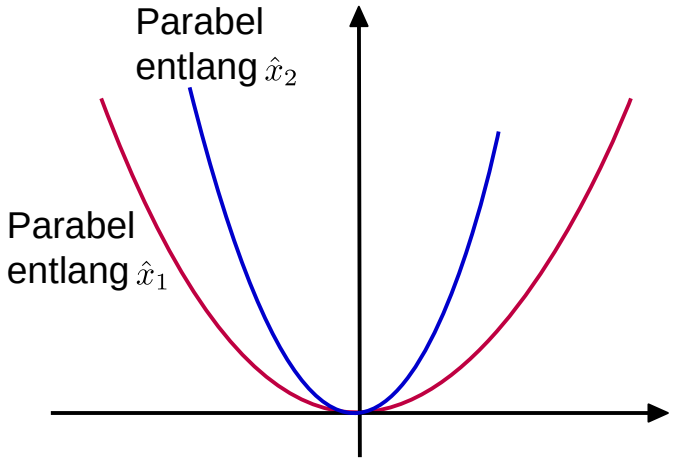
- **Problem**: Die Öffnungswinkel der Parabeln entlang der Hauptachsen – und damit verbunden die zweiten Richtungsableitungen entlang der Hauptachsen – sind i.a. unterschiedlich groß.

- Wie ist η zu wählen, um optimale und garantierte Konvergenz zu erreichen?

- Je höherdimensional der Definitionsbereich von $F(x)$ ist desto schwieriger lässt sich diese Frage im Rahmen des naiven Gradientenabstiegverfahrens beantworten.

- **NB**: Konvergenz ist umso schwieriger zu erreichen je größer die Konditionszahl der Hessematrix ist (d.h. je weiter die Eigenwerte der Hessematrix auseinander liegen).



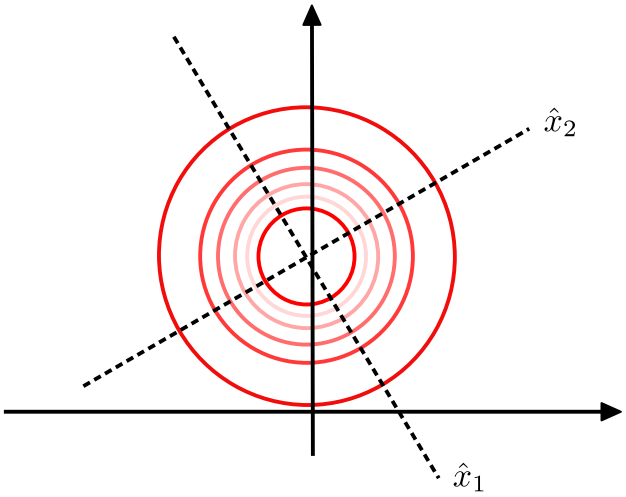


Gradientenabstieg in d Dimensionen

- In d Dimensionen lässt sich jede Parabel auf ihre Hauptachsen transformieren. Schnitte entlang der Hauptachsen führen wiederum auf Parabeln.

- **Problem**: Die Öffnungswinkel der Parabeln entlang der Hauptachsen – und damit verbunden die zweiten Richtungsableitungen entlang der Hauptachsen – sind i.a. unterschiedlich groß.

- Durch das Newtonverfahren wird das Problem automatisch gelöst. Die Multiplikation mit $H^{-1}(x)$ transformiert die Parabel auf eine Einheitsparabel.

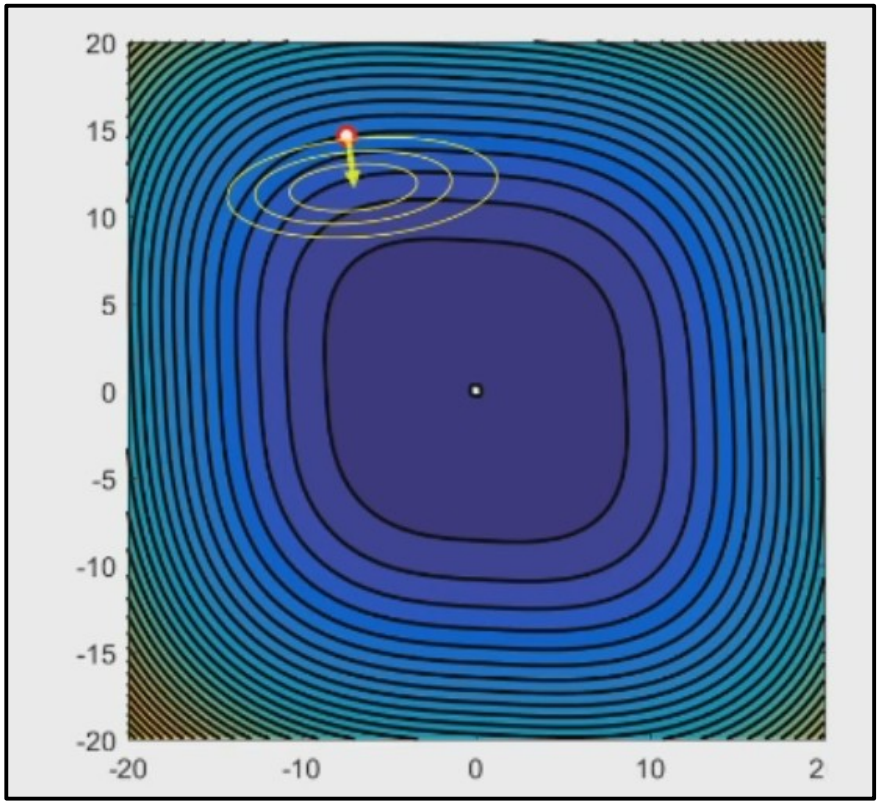


Sie können das auch durch die Aktualisierungsregel erkennen.

$$\vec{x}_{k+1} = \vec{x}_k - \underbrace{H^{-1}(\vec{x}_k)\vec{\nabla}F(x_k)}_{\text{Transformation des Gradienten mit } \eta \equiv 1}$$

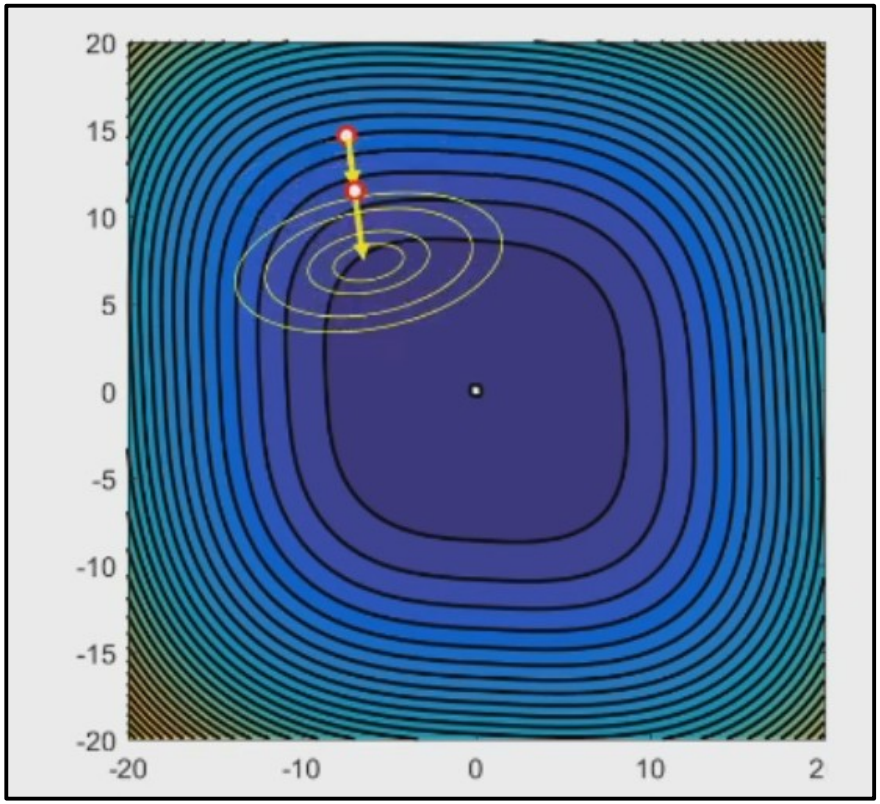
Newtonverfahren in 2 Dimensionen

- Die folgenden Folien veranschaulichen das Newtonverfahren in 2 Dimensionen:



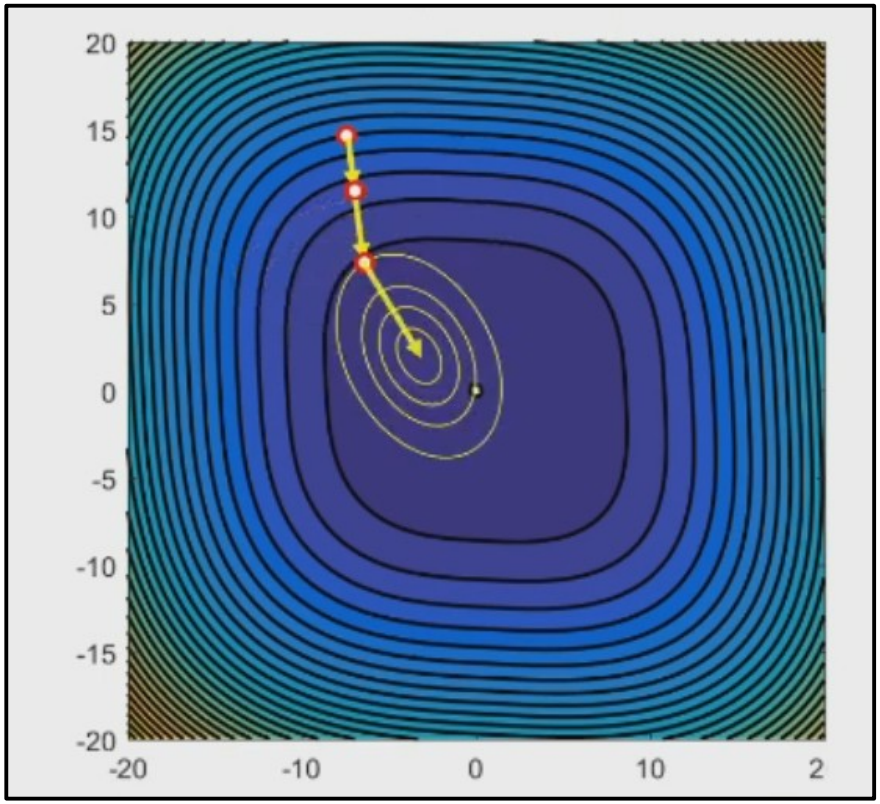
Newtonverfahren in 2 Dimensionen

- Die folgenden Folien veranschaulichen das Newtonverfahren in 2 Dimensionen:



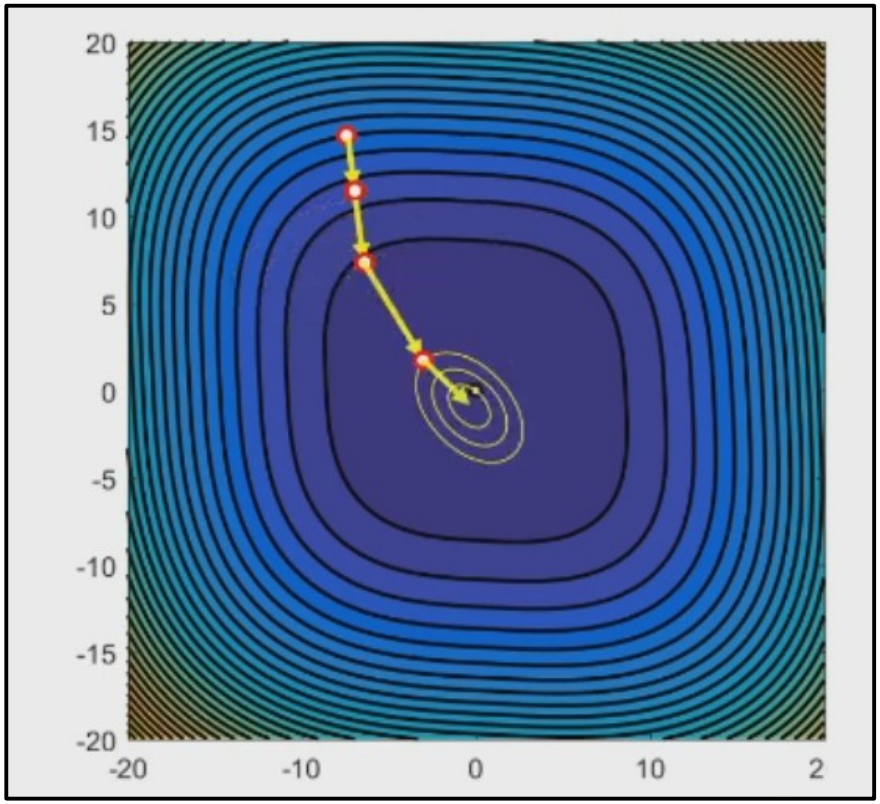
Newtonverfahren in 2 Dimensionen

- Die folgenden Folien veranschaulichen das Newtonverfahren in 2 Dimensionen:



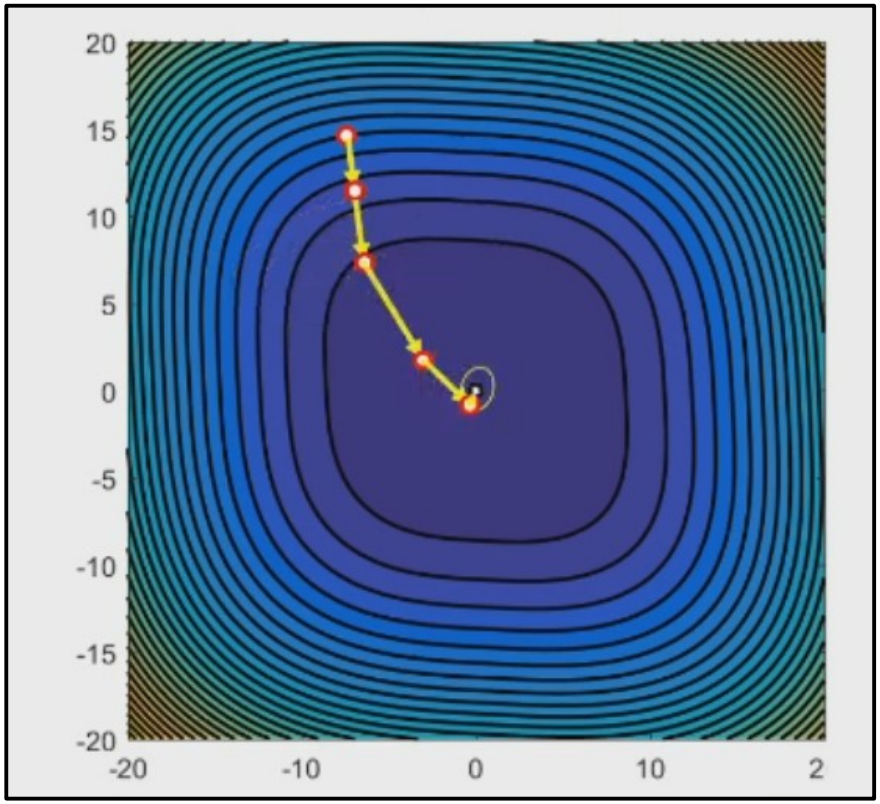
Newtonverfahren in 2 Dimensionen

- Die folgenden Folien veranschaulichen das Newtonverfahren in 2 Dimensionen:



Newtonverfahren in 2 Dimensionen

- Die folgenden Folien veranschaulichen das Newtonverfahren in 2 Dimensionen:



Newtonverfahren – Diskussion

- Das Newtonverfahren hat sehr gute Konvergenzeigenschaften für konvexe Funktionen.

Newtonverfahren – Diskussion

- Das Newtonverfahren hat sehr gute Konvergenzeigenschaften für konvexe Funktionen.
- Konvergenz in ein globales Minimum ist jedoch nicht garantiert.

Newtonverfahren – Diskussion

- Das Newtonverfahren hat sehr gute Konvergenzeigenschaften für konvexe Funktionen.
- Konvergenz in ein globales Minimum ist jedoch nicht garantiert.
- Probleme bestehen bei nicht konvexen Funktionen, bei denen insbesondere in höheren Dimensionen $H(x)$ nicht garantiert **positiv definit** ist.

Newtonverfahren – Diskussion

- Das Newtonverfahren hat sehr gute Konvergenzeigenschaften für konvexe Funktionen.
- Konvergenz in ein globales Minimum ist jedoch nicht garantiert.
- Probleme bestehen bei nicht konvexen Funktionen, bei denen insbesondere in höheren Dimensionen $H(x)$ nicht garantiert **positiv definit** ist.
- Zur Optimierung bei Problemen sehr hoher Dimensionalität (wie z.B. im Bereich maschienellen Lernens) ist das Newtonverfahren ungeeignet, weil es nicht nur die Berechnung sondern auch die Inversion von $H(x)$ voraussetzt.

Adaptive Schrittweite

- Eine andere Möglichkeit Konvergenzprobleme zu vermeiden, besteht darin (ggf. mit großer) fester Schrittweite zu beginnen und diese sukzessive zu verringern.

- Der Beginn mit großer Schrittweite soll dabei verhindern, durch unglückliche Wahl der Anfangsparameter in Nebenminima „gefangen“ zu bleiben.

- Dabei sollten die folgenden Bedingungen an die Schrittweiten bestehen:

$$\sum_{k=1}^{\infty} \eta_k \rightarrow \infty$$

Es muss möglich sein, durch eine unendlich lange Folge den gesamten Definitionsbereich der Funktion abzudecken.

Adaptive Schrittweite

- Eine andere Möglichkeit Konvergenzprobleme zu vermeiden, besteht darin (ggf. mit großer) fester Schrittweite zu beginnen und diese sukzessive zu verringern.

- Der Beginn mit großer Schrittweite soll dabei verhindern, durch unglückliche Wahl der Anfangsparameter in Nebenminima „gefangen“ zu bleiben.

- Dabei sollten die folgenden Bedingungen an die Schrittweiten bestehen:

$$\sum_{k=1}^{\infty} \eta_k \to \infty$$

Es muss möglich sein, durch eine unendlich lange Folge den gesamten Definitionsbereich der Funktion abzudecken.

$$\sum_{k=1}^{\infty} \eta_k^2 < \infty$$

Die Reihe soll trotzdem konvergent sein und die Elemente der Folge immer kleiner werden.

Adaptive Schrittweite

- Eine andere Möglichkeit Konvergenzprobleme zu vermeiden, besteht darin (ggf. mit großer) fester Schrittweite zu beginnen und diese sukzessive zu verringern.

- Der Beginn mit großer Schrittweite soll dabei verhindern, durch unglückliche Wahl der Anfangsparameter in Nebenminima „gefangen“ zu bleiben.

- Etablierte Schrittweitenalgorithmen sind:

$$\eta_k = \frac{\eta_0}{k+1} \qquad \text{(Linear)}$$

$$\eta_k = \frac{\eta_0}{(k+1)^2} \qquad \text{(Quadratisch)}$$

$$\eta_k = \eta_0\, e^{-\beta\, k} \quad \beta > 0 \qquad \text{(Exponentiell)}$$

Adaptive Schrittweite

- Eine andere Möglichkeit Konvergenzprobleme zu vermeiden, besteht darin (ggf. mit großer) fester Schrittweite zu beginnen und diese sukzessive zu verringern.

- Der Beginn mit großer Schrittweite soll dabei verhindern, durch unglückliche Wahl der Anfangsparameter in Nebenminima „gefangen“ zu bleiben.

- Etablierte Schrittweitenalgorithmen sind:

$$\eta_k = \frac{\eta_0}{k+1} \qquad \text{(Linear)}$$

$$\eta_k = \frac{\eta_0}{(k+1)^2} \qquad \text{(Quadratisch)}$$

$$\eta_k = \eta_0\, e^{-\beta\, k} \quad \beta > 0 \qquad \text{(Exponentiell)}$$

- **NB**: Eine im Bereich maschienellen Lernens verwendete Methode ist z.B. mit einer festen Schrittweite zu beginnen, die nach einer bestimmten Anzahl an initialen Schritten sukessive reduziert wird.

Impulsverfahren (engl. *momentum methods*)

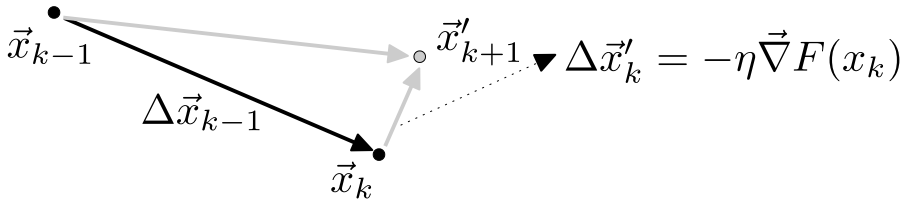
- Impulsverfahren beruhen auf der Annahme, dass die ursprüngliche Richtung bereits, eine gute Wahl zum Auffinden des Minimums war.

- Man behält also ein „Gedächtnis“ dieser Information in der Aktualisierungsregel des Gradientenabstiegs, z.B. durch (gewichtete) Mittelwertbildung:

$$\Delta\vec{x}_{k+1} = \beta\Delta\vec{x}_k - \eta\vec{\nabla}F(\vec{x}_k)$$
$$\vec{x}_{k+1} = \vec{x}_k + \Delta\vec{x}_{k+1}$$

solange:

$$|F(\vec{x}_k) - F(\vec{x}_{k-1})| > \epsilon$$



Einfacher Gradientenabstieg (gestrichene Variablen) in grau.

- Dieses einfache Verfahren verzichtet auf die zweite Ableitung. Es unterdrückt schnelle, stark variierende Änderungen im Abstieg und somit auch Oszillationen.

Impulsverfahren (engl. *momentum methods*)

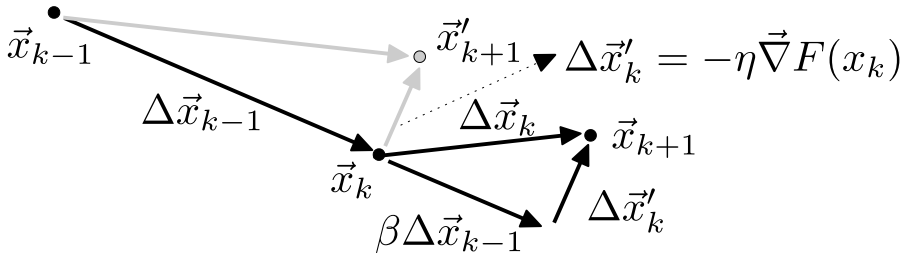
- Impulsverfahren beruhen auf der Annahme, dass die ursprüngliche Richtung bereits, eine gute Wahl zum Auffinden des Minimums war.

- Man behält also ein „Gedächtnis“ dieser Information in der Aktualisierungsregel des Gradientenabstiegs, z.B. durch (gewichtete) Mittelwertbildung:

$$\Delta\vec{x}_{k+1} = \beta\Delta\vec{x}_k - \eta\vec{\nabla}F(\vec{x}_k)$$
$$\vec{x}_{k+1} = \vec{x}_k + \Delta\vec{x}_{k+1}$$

solange:

$$|F(\vec{x}_k) - F(\vec{x}_{k-1})| > \epsilon$$



Einfacher Gradientenabstieg (gestrichene Variablen) in **grau**.
Gradientenabstieg nach Impulsverfahren in **schwarz**.

- Dieses einfache Verfahren verzichtet auf die zweite Ableitung. Es unterdrückt schnelle, stark variierende Änderungen im Abstieg und somit auch Oszillationen.

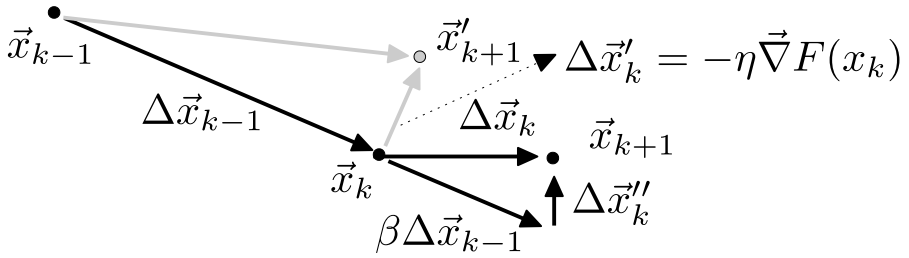
Nestorov's accelerated gradient (Y. Nesterov 1983)

- Eine Variation mit nachweisbar noch besserem Konvergenzverhalten besteht darin zuerst den Schritt aus dem vorherigen Abstieg durchzuführen und dort die Ableitung auszuwerten:

$$\Delta\vec{x}_{k+1} = \beta\Delta\vec{x}_k - \eta\vec{\nabla}F(\vec{x}_k + \beta\Delta\vec{x}_k)$$
$$\vec{x}_{k+1} = \vec{x}_k + \Delta\vec{x}_{k+1}$$

solange:

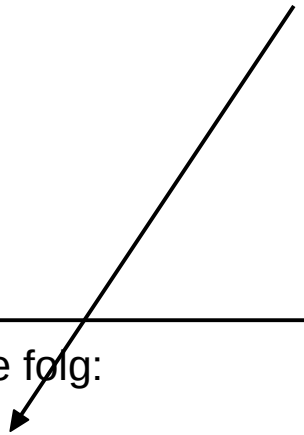
$$|F(\vec{x}_k) - F(\vec{x}_{k-1})| > \epsilon$$



Einfacher Gradientenabstieg (gestrichene Variablen) in **grau**.
Gradientenabstieg nach Impulsverfahren in **schwarz**.

Putting things together…

- Wir haben uns bis hierhin mit der optimalen Schrittweite für den Gradientenabstieg beschäftigt.



Aktualisiere wie folg:

$$\vec{w}^{(k+1)} = \vec{w}_k - \eta \nabla_{\vec{w}^{(k)}} L, \quad \eta > 0$$

solange:

$$|L(x_k) - L(x_{k-1})| > \epsilon$$

Putting things together…

- Wir haben uns bis hierhin mit der optimalen Schrittweite für den Gradientenabstieg beschäftigt.

- Im folgenden werden wir praktische Aspekte bei der Berechnung des Gradienten diskutieren.

Aktualisiere wie folg:

$$\vec{w}^{(k+1)} = \vec{w}_k - \eta \nabla_{\vec{w}^{(k)}} L, \quad \eta > 0$$

solange:

$$|L(x_k) - L(x_{k-1})| > \epsilon$$

Putting things together…

- Wir haben uns bis hierhin mit der optimalen Schrittweite für den Gradientenabstieg beschäftigt.

- Im folgenden werden wir praktische Aspekte bei der Berechnung des Gradienten diskutieren.

Aktualisiere wie folg:

$$\vec{w}^{(k+1)} = \vec{w}_k - \eta \nabla_{\vec{w}^{(k)}} L, \quad \eta > 0$$

solange:

$$|L(x_k) - L(x_{k-1})| > \epsilon$$

- Beachten Sie: wir bewegen uns immer noch innerhalb einer Iteration der Minimierung. D.h. wenn wir jetzt gleich $\nabla_{\vec{w}} L$ berechnen sind alle $\{w_j\}$ fix.

Ableitung an einem festen Punkt im MLP

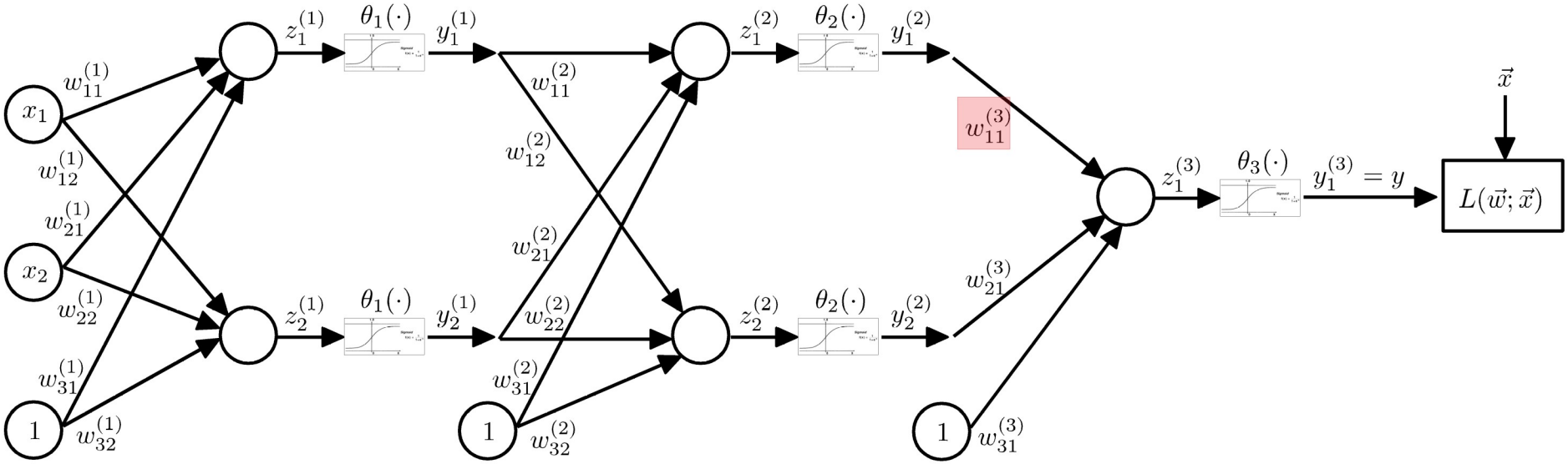
$$\frac{\mathrm{d}L}{\mathrm{d}w_{11}^{(3)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y} \cdot \frac{\mathrm{d}y}{\mathrm{d}z_1^{(3)}} \cdot \frac{\mathrm{d}z_1^{(3)}}{\mathrm{d}w_{11}^{(3)}}$$

$$\frac{\mathrm{d}L}{\mathrm{d}y} = \frac{1}{y} \qquad \text{(für Kreuzentropie)}$$

$$\frac{\mathrm{d}y}{\mathrm{d}z_1^{(3)}} = \theta_3(z)\,(1-\theta_3(z)) \qquad \text{(für Sigmoid)}$$

$$\frac{\mathrm{d}z_1^{(3)}}{\mathrm{d}w_{11}^{(3)}} = y_1^{(2)}$$

- Ableitung erfolgt produktweise nach Kettenregel.



Ableitung an einem festen Punkt im MLP

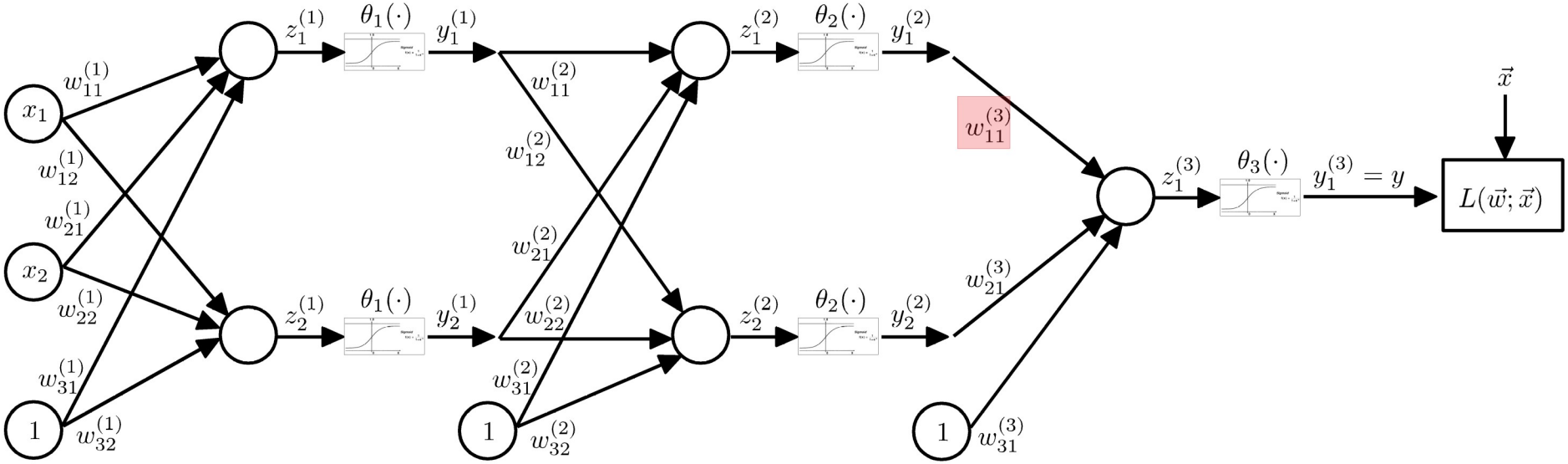
$$\frac{\mathrm{d}L}{\mathrm{d}w_{11}^{(3)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y} \cdot \frac{\mathrm{d}y}{\mathrm{d}z_1^{(3)}} \cdot \frac{\mathrm{d}z_1^{(3)}}{\mathrm{d}w_{11}^{(3)}}$$

$$\frac{\mathrm{d}L}{\mathrm{d}y} = \frac{1}{y}\bigg|_{y(\vec{w};\vec{x})} \qquad \text{(für Kreuzentropie)}$$

$$\frac{\mathrm{d}y}{\mathrm{d}z_1^{(3)}} = \theta_3(z)\,(1-\theta_3(z))\bigg|_{z=z_1^{(3)}(\vec{w};\vec{x})} \qquad \text{(für Sigmoid)}$$

$$\frac{\mathrm{d}z_1^{(3)}}{\mathrm{d}w_{11}^{(3)}} = y_1^{(2)}\bigg|_{y_1^{(2)}(\vec{w};\vec{x})}$$

- Ableitung erfolgt produktweise nach Kettenregel.
- Zur konkreten Bestimmung der Ableitung ist der Wert des Arguments notwendig.



Ableitung an einem festen Punkt im MLP

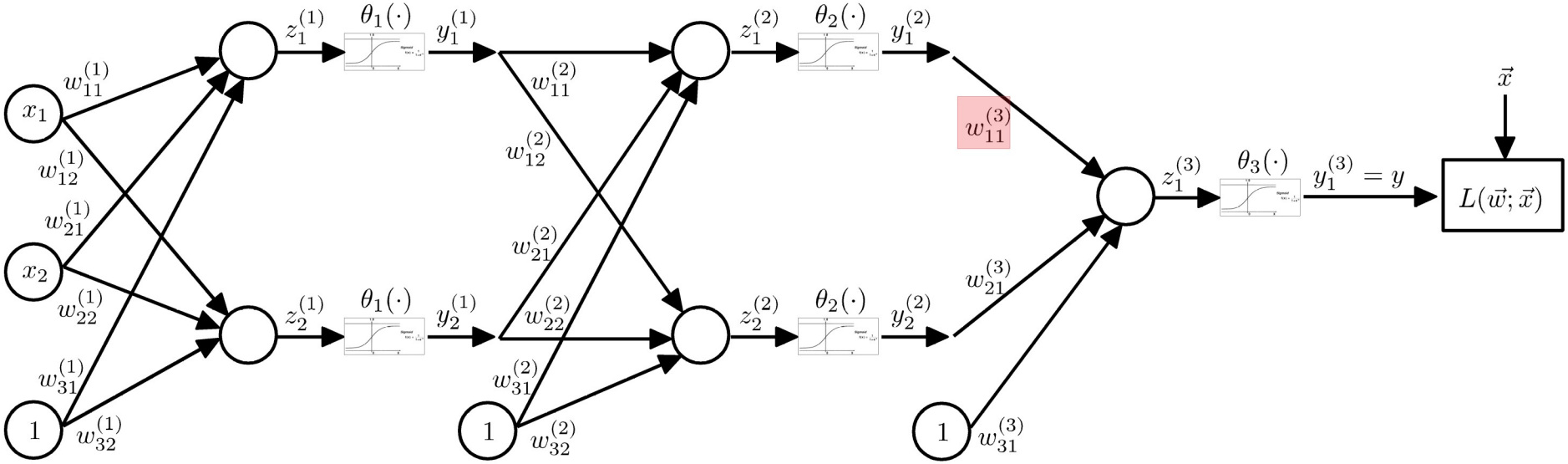
$$\frac{\mathrm{d}L}{\mathrm{d}w_{11}^{(3)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y} \cdot \frac{\mathrm{d}y}{\mathrm{d}z_1^{(3)}} \cdot \frac{\mathrm{d}z_1^{(3)}}{\mathrm{d}w_{11}^{(3)}}$$

$$\frac{\mathrm{d}L}{\mathrm{d}y} = \frac{1}{y}\bigg|_{y(\vec{w};\vec{x})} \qquad \text{(für Kreuzentropie)}$$

$$\frac{\mathrm{d}y}{\mathrm{d}z_1^{(3)}} = \theta_3(z)\,(1-\theta_3(z))\bigg|_{z=z_1^{(3)}(\vec{w};\vec{x})} \qquad \text{(für Sigmoid)}$$

$$\frac{\mathrm{d}z_1^{(3)}}{\mathrm{d}w_{11}^{(3)}} = y_1^{(2)}\bigg|_{y_1^{(2)}(\vec{w};\vec{x})}$$

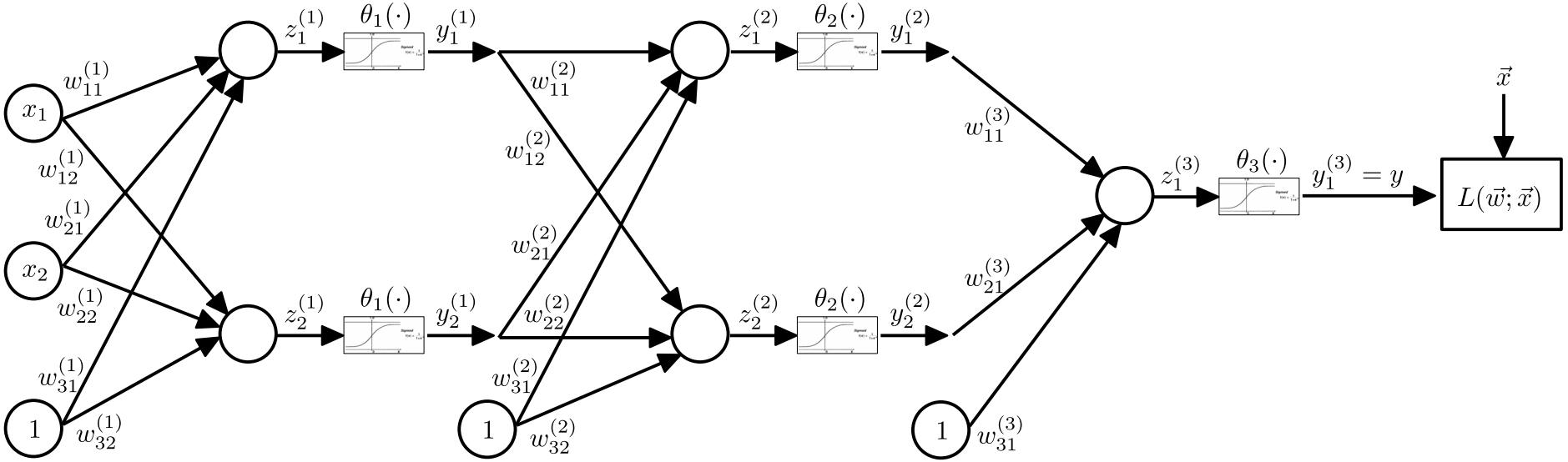
- Ableitung erfolgt produktweise nach Kettenregel.
- Zur konkreten Bestimmung der Ableitung ist der Wert des Arguments notwendig.
- D.h. jeder Gradientenabstieg erfordert zwei Schritte!



Schritt-1: Forward pass

- Berechne und speichere alle $\{z_i^{(k)}\}$ und $\{y_i^{(k)}\}$ für die gegebene MLP Konfiguration, Lage für Lage:

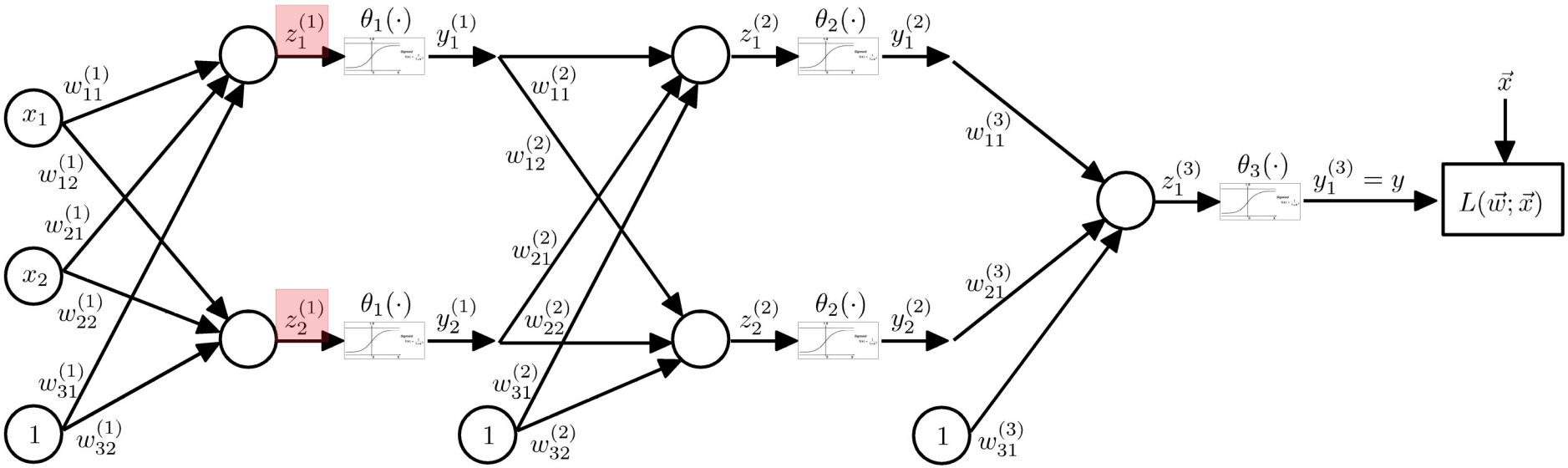
$$y(\vec{w};\vec{x}) = y_1^{(3)}(\vec{w};\vec{x}) = \theta_3\left(\sum w_{ij}^{(3)}\theta_2\left(\sum w_{ij}^{(2)}\theta_1\left(\sum w_{ij}^{(1)}x_i\right)\right)\right)$$



Schritt-1: Forward pass

- Berechne und speichere alle $\{z_i^{(k)}\}$ und $\{y_i^{(k)}\}$ für die gegebene MLP Konfiguration, Lage für Lage:

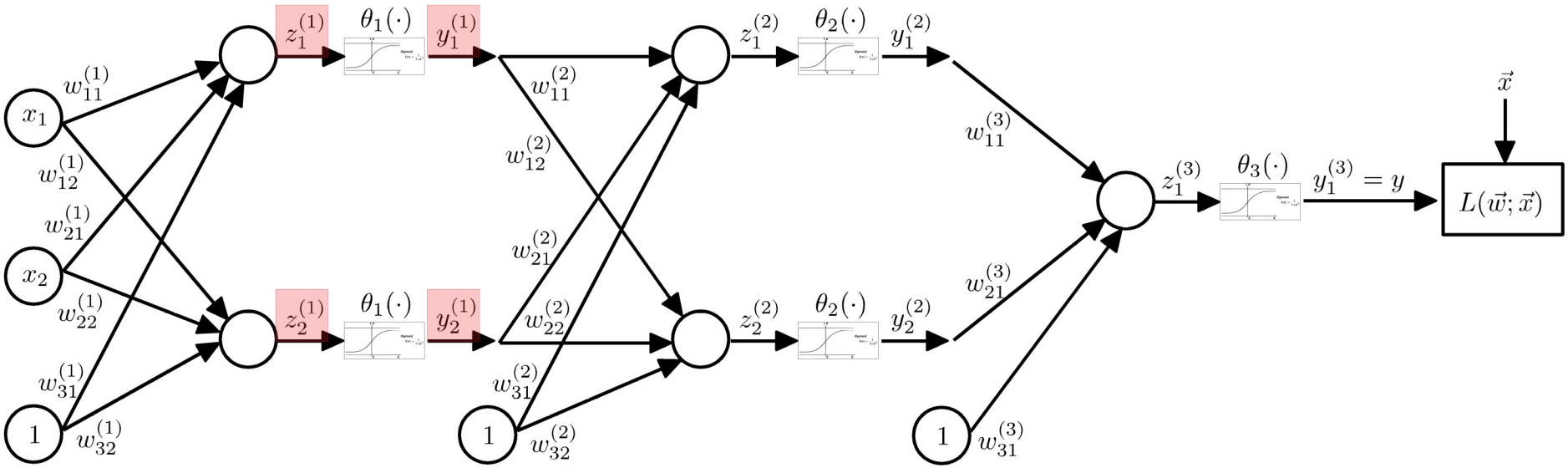
$$y(\vec{w};\vec{x}) = y_1^{(3)}(\vec{w};\vec{x}) = \theta_3\left(\sum w_{ij}^{(3)}\theta_2\left(\sum w_{ij}^{(2)}\theta_1\left(\sum w_{ij}^{(1)}x_i\right)\right)\right)$$



Schritt-1: Forward pass

- Berechne und speichere alle $\{z_i^{(k)}\}$ und $\{y_i^{(k)}\}$ für die gegebene MLP Konfiguration, Lage für Lage:

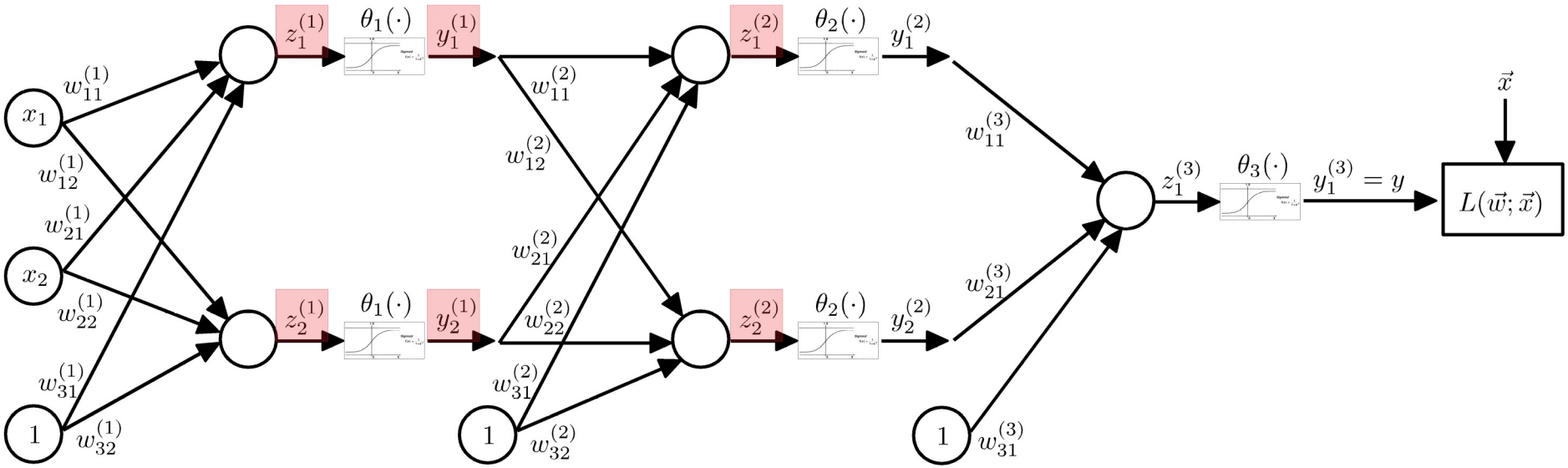
$$y(\vec{w};\vec{x}) = y_1^{(3)}(\vec{w};\vec{x}) = \theta_3\left(\sum w_{ij}^{(3)}\theta_2\left(\sum w_{ij}^{(2)}\theta_1\left(\sum w_{ij}^{(1)}x_i\right)\right)\right)$$



Schritt-1: Forward pass

- Berechne und speichere alle $\{z_i^{(k)}\}$ und $\{y_i^{(k)}\}$ für die gegebene MLP Konfiguration, Lage für Lage:

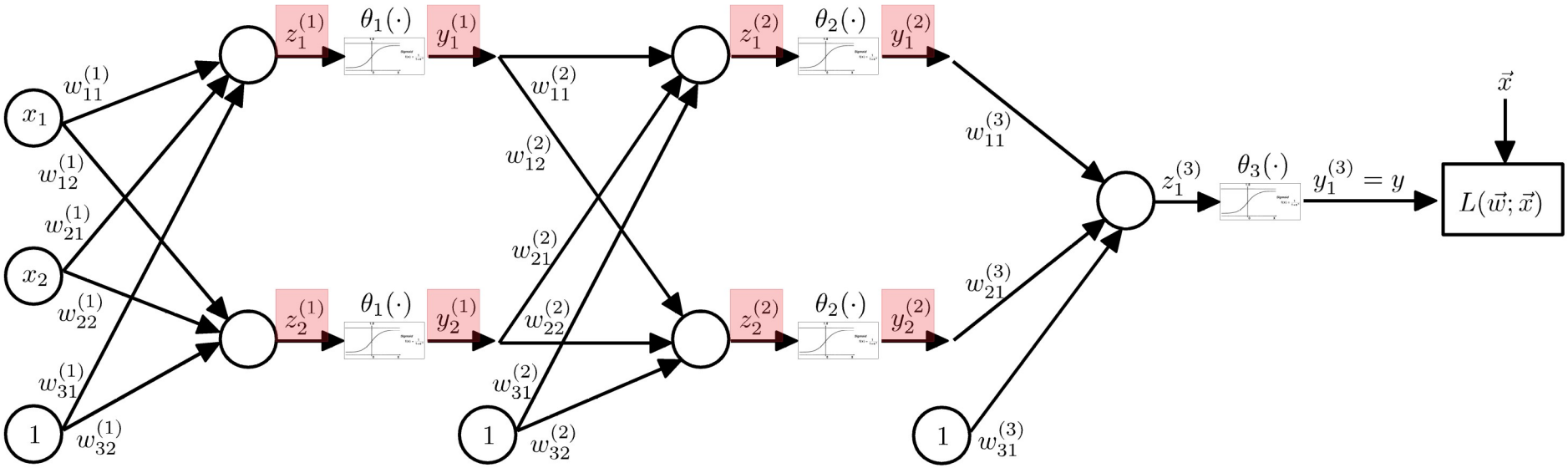
$$y(\vec{w};\vec{x}) = y_1^{(3)}(\vec{w};\vec{x}) = \theta_3\left(\sum w_{ij}^{(3)}\theta_2\left(\sum w_{ij}^{(2)}\theta_1\left(\sum w_{ij}^{(1)}x_i\right)\right)\right)$$



Schritt-1: Forward pass

- Berechne und speichere alle $\{z_i^{(k)}\}$ und $\{y_i^{(k)}\}$ für die gegebene MLP Konfiguration, Lage für Lage:

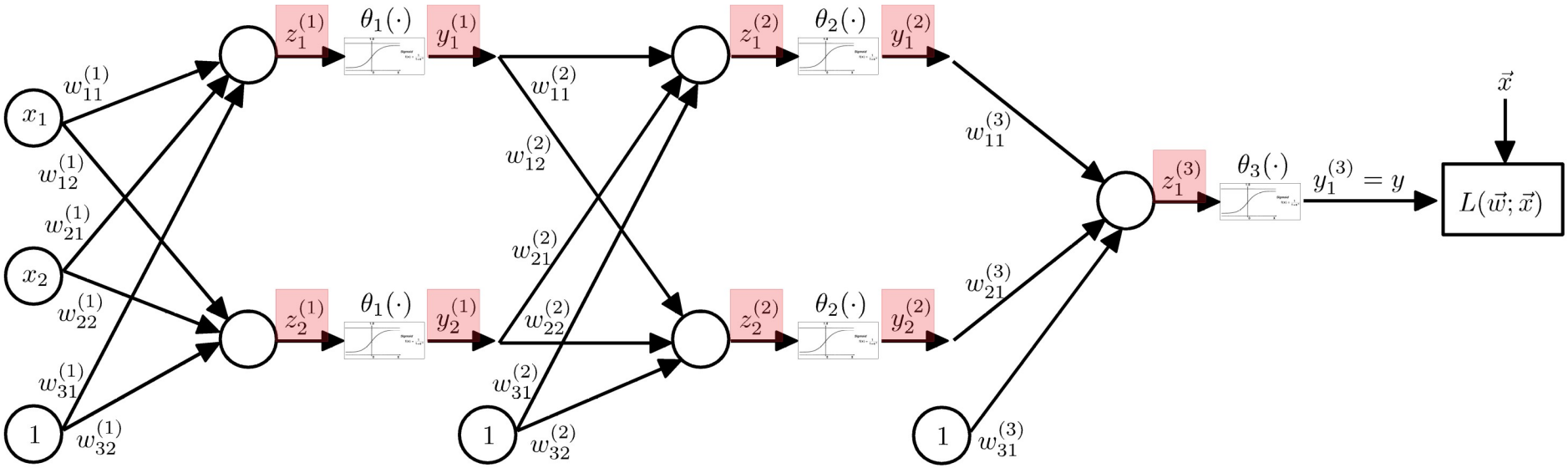
$$y(\vec{w};\vec{x}) = y_1^{(3)}(\vec{w};\vec{x}) = \theta_3\left(\sum w_{ij}^{(3)}\theta_2\left(\sum w_{ij}^{(2)}\theta_1\left(\sum w_{ij}^{(1)}x_i\right)\right)\right)$$



Schritt-1: Forward pass

- Berechne und speichere alle $\{z_i^{(k)}\}$ und $\{y_i^{(k)}\}$ für die gegebene MLP Konfiguration, Lage für Lage:

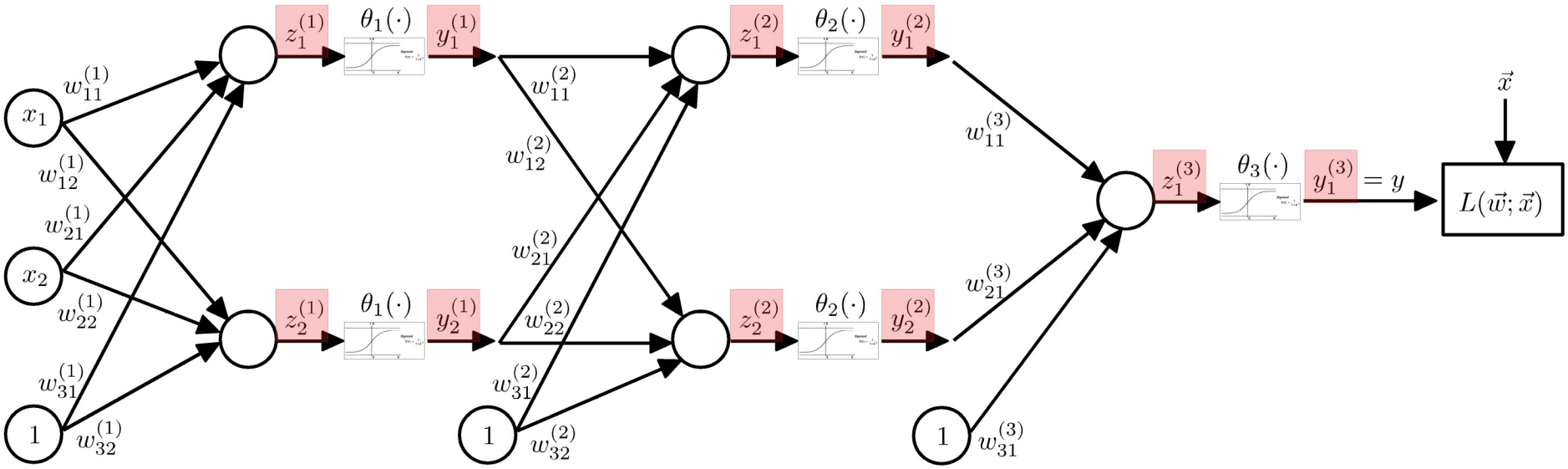
$$y(\vec{w};\vec{x}) = y_1^{(3)}(\vec{w};\vec{x}) = \theta_3\left(\sum w_{ij}^{(3)}\theta_2\left(\sum w_{ij}^{(2)}\theta_1\left(\sum w_{ij}^{(1)}x_i\right)\right)\right)$$



Schritt-1: Forward pass

- Berechne und speichere alle $\{z_i^{(k)}\}$ und $\{y_i^{(k)}\}$ für die gegebene MLP Konfiguration, Lage für Lage:

$$y(\vec{w};\vec{x}) = y_1^{(3)}(\vec{w};\vec{x}) = \theta_3\left(\sum w_{ij}^{(3)}\theta_2\left(\sum w_{ij}^{(2)}\theta_1\left(\sum w_{ij}^{(1)}x_i\right)\right)\right)$$



Gradientenabstieg – Teil 1

- Algorithmus für den Gradientenabstieg für ein MLP mit N Lagen:

Forward pass:

- Beginne mit: $y_j^{(0)} = x_j$

- Für jede Lage $k = 1 \ldots N$:

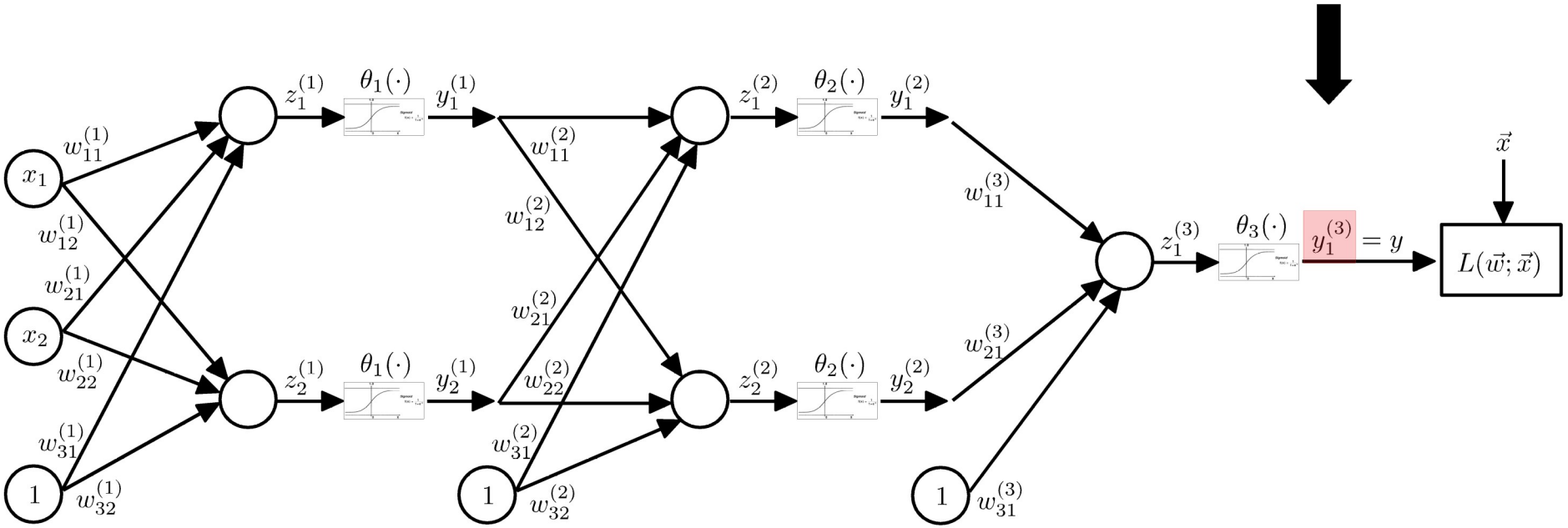
$$z_i^{(k)} = \sum_j y_j^{(k-1)} w_{ij}^{(k)}$$

$$y_i^{(k)} = \theta_k \left(z_i^{(k)} \right)$$

Schritt-2: Backward pass

- Berechne die Ableitungen nach den $\{w_{ij}^{(k)}\}$:

$$\frac{\mathrm{d}L}{\mathrm{d}y}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y_1^{(3)}}$$

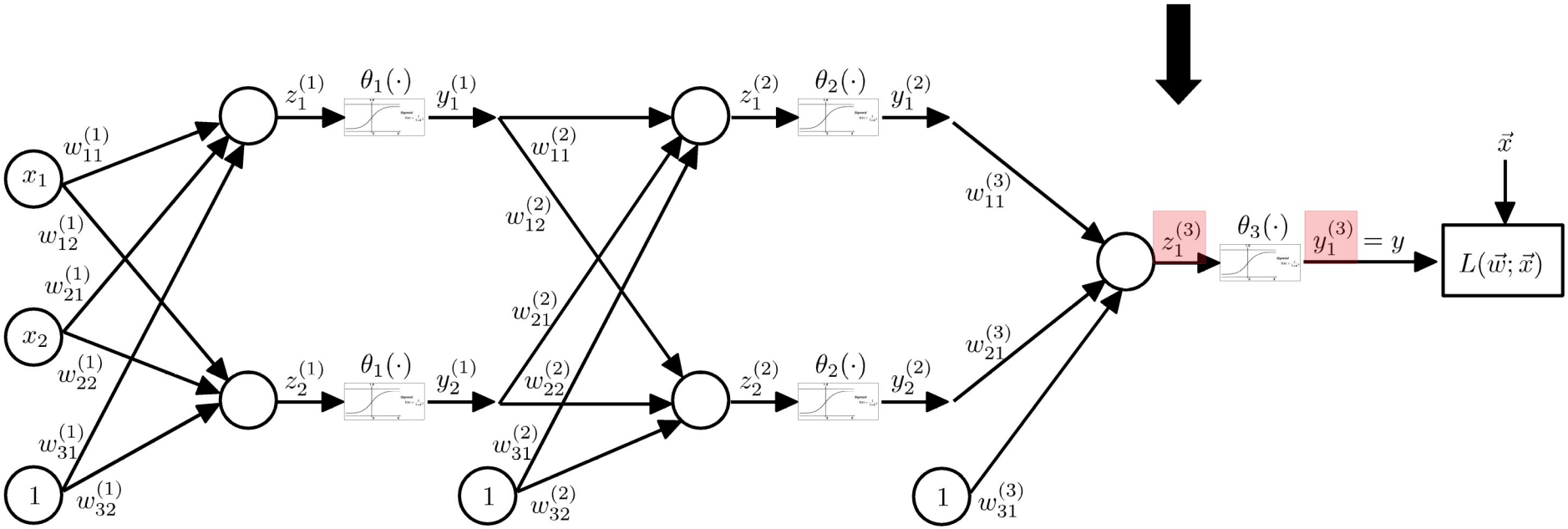


Schritt-2: Backward pass

- Berechne die Ableitungen nach den $\{w_{ij}^{(k)}\}$:

$$\frac{\mathrm{d}L}{\mathrm{d}z_1^{(3)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y_1^{(3)}} \cdot \theta_3'\left(z_1^{(3)}\right)$$

Bereits im vorherigen Schritt berechnet



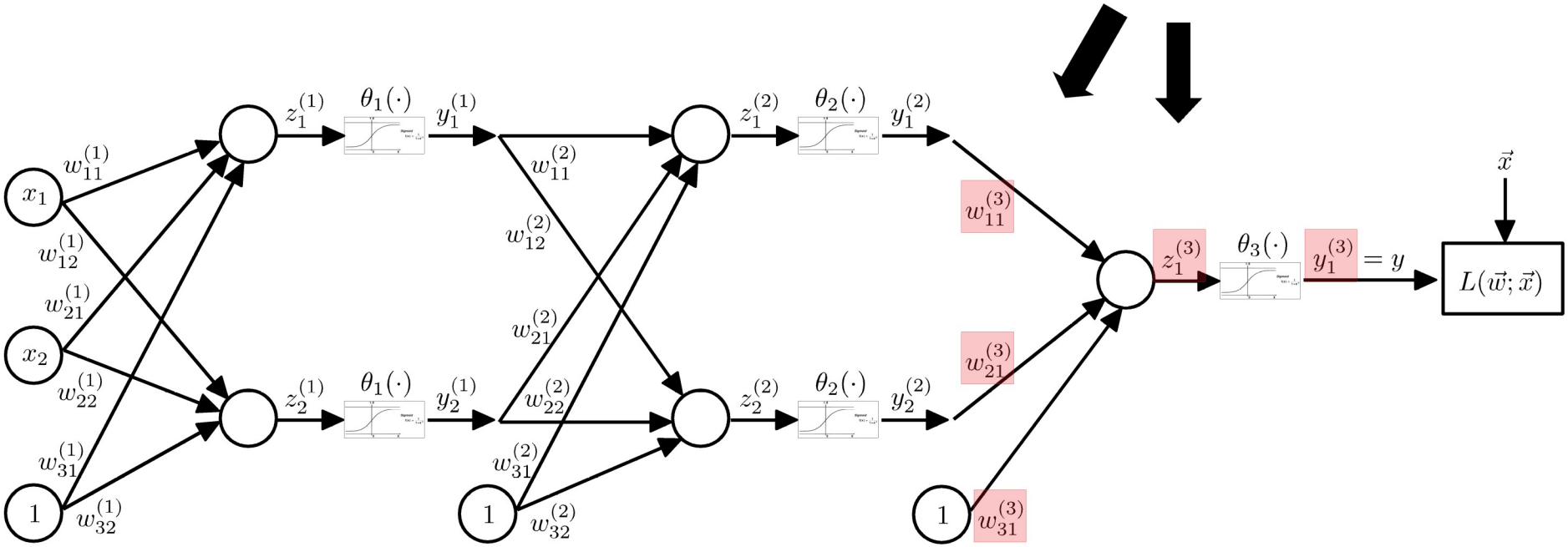
Schritt-2: Backward pass

- Berechne die Ableitungen nach den $\{w_{ij}^{(k)}\}$:

$$\frac{\mathrm{d}L}{\mathrm{d}z_1^{(3)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y_1^{(3)}} \cdot \theta_3'\left(z_1^{(3)}\right)$$

$$\frac{\mathrm{d}L}{\mathrm{d}w_{i1}^{(3)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}z_1^{(3)}} \cdot \frac{\mathrm{d}z_1^{(3)}}{\mathrm{d}w_{i1}^{(3)}} = \frac{\mathrm{d}L}{\mathrm{d}z_1^{(3)}} \cdot y_i^{(2)}$$

Bereits im vorherigen Schritt berechnet

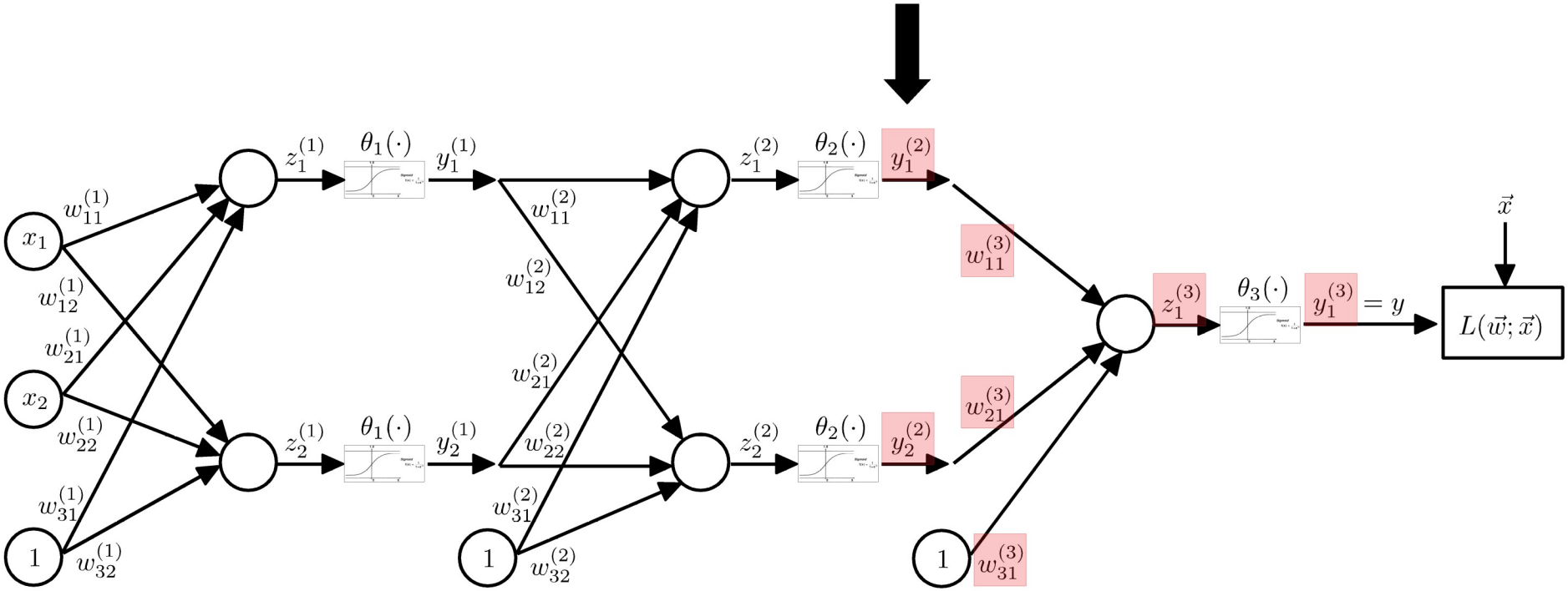


Schritt-2: Backward pass

- Berechne die Ableitungen nach den $\{w_{ij}^{(k)}\}$:

$$\frac{\mathrm{d}L}{\mathrm{d}y_i^{(2)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}z_1^{(3)}} \cdot \frac{\mathrm{d}z_1^{(3)}}{\mathrm{d}y_i^{(2)}} = \frac{\mathrm{d}L}{\mathrm{d}z_1^{(3)}} \cdot w_{i1}^{(3)}$$

Bereits im vorherigen Schritt berechnet

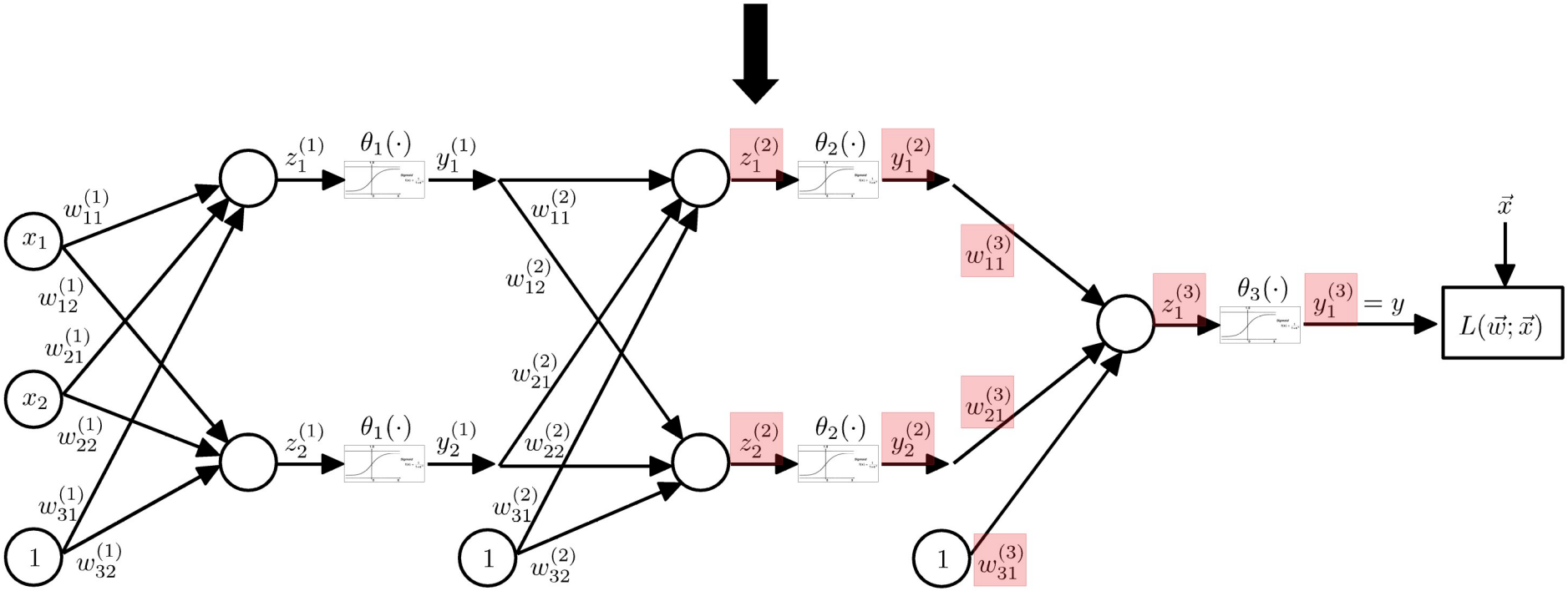


Schritt-2: Backward pass

- Berechne die Ableitungen nach den $\{w_{ij}^{(k)}\}$:

$$\frac{\mathrm{d}L}{\mathrm{d}z_i^{(2)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y_i^{(2)}} \cdot \theta_2'\left(z_i^{(2)}\right)$$

Bereits im vorherigen Schritt berechnet



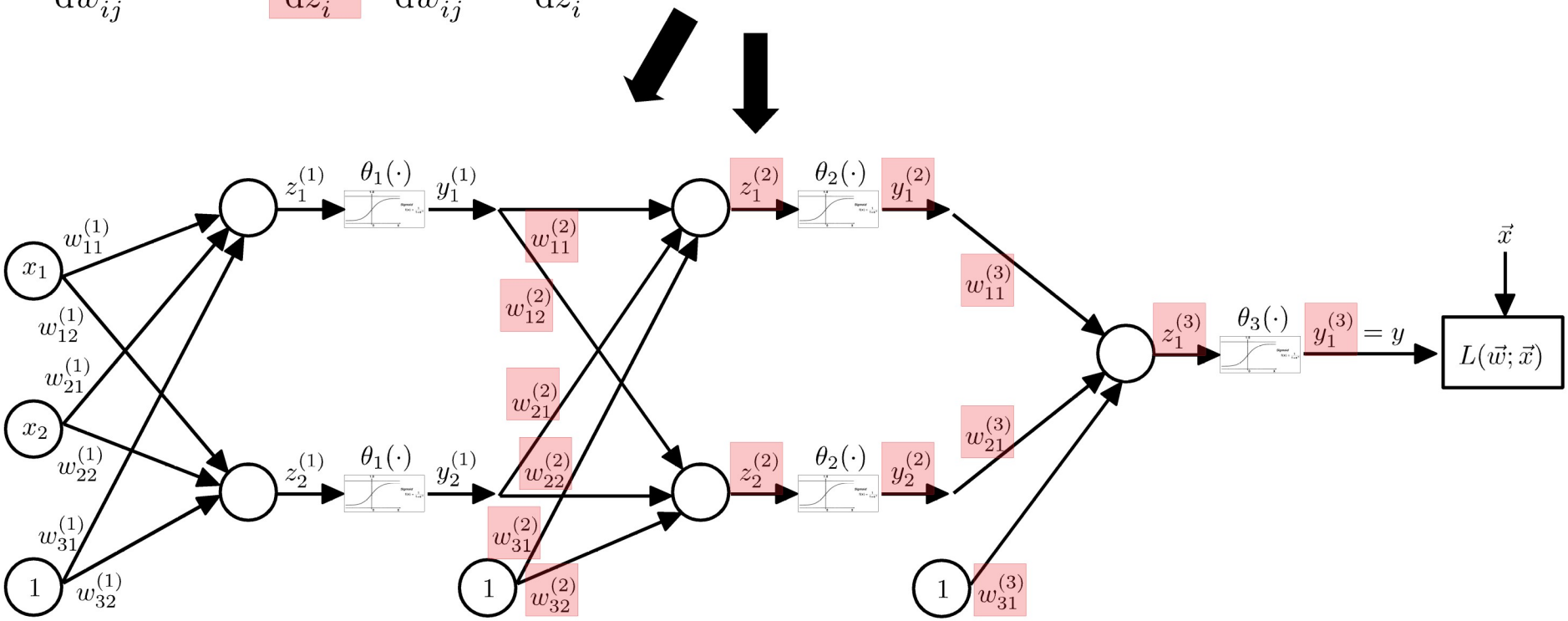
Schritt-2: Backward pass

- Berechne die Ableitungen nach den $\{w_{ij}^{(k)}\}$:

$$\frac{\mathrm{d}L}{\mathrm{d}z_i^{(2)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y_i^{(2)}} \cdot \theta_2'\left(z_i^{(2)}\right)$$

$$\frac{\mathrm{d}L}{\mathrm{d}w_{ij}^{(2)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}z_i^{(2)}} \cdot \frac{\mathrm{d}z_i^{(2)}}{\mathrm{d}w_{ij}^{(2)}} = \frac{\mathrm{d}L}{\mathrm{d}z_i^{(2)}} \cdot y_j^{(1)}$$

Bereits im vorherigen Schritt berechnet

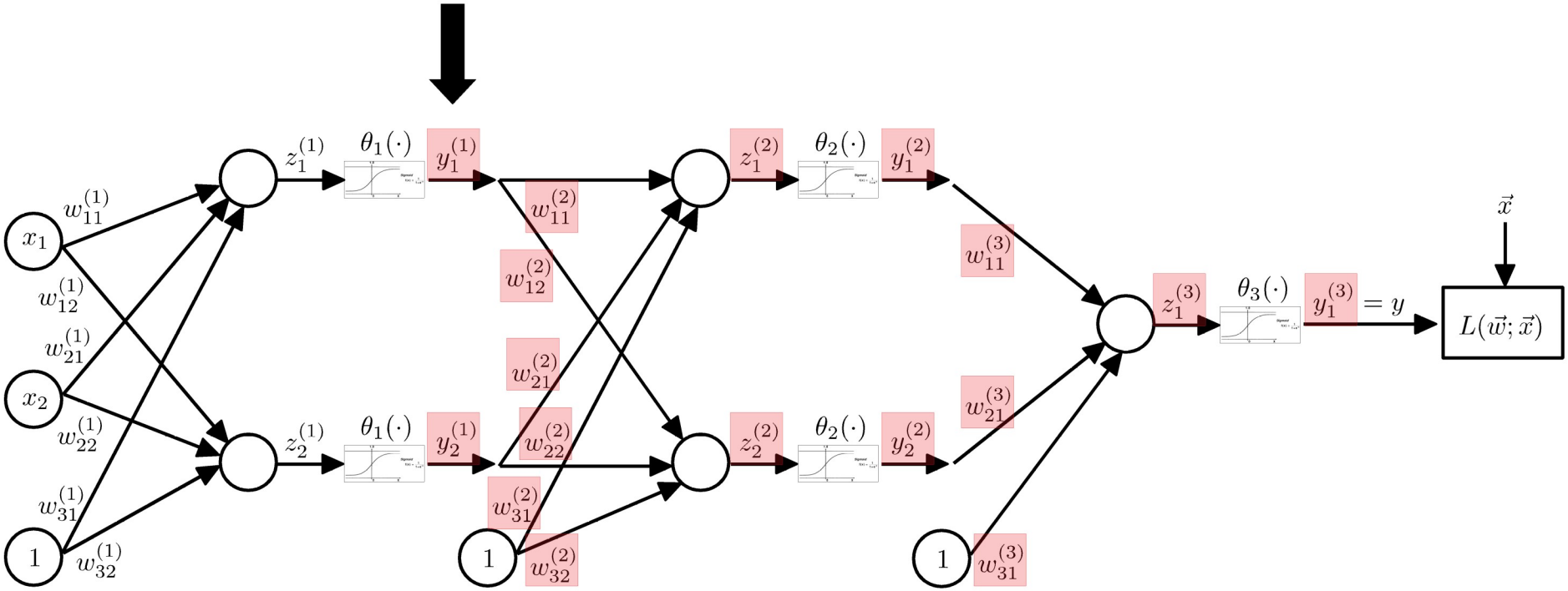


Schritt-2: Backward pass

- Berechne die Ableitungen nach den $\{w_{ij}^{(k)}\}$:

$$\frac{\mathrm{d}L}{\mathrm{d}y_i^{(1)}}(\vec{w};\vec{x}) = \sum_{j=1}^{2} \frac{\mathrm{d}L}{\mathrm{d}z_j^{(2)}} \cdot \frac{\mathrm{d}z_j^{(2)}}{\mathrm{d}y_i^{(1)}} = \sum_{j=1}^{2} \frac{\mathrm{d}L}{\mathrm{d}z_j^{(2)}} \cdot w_{ij}^{(2)}$$

Bereits im vorherigen Schritt berechnet

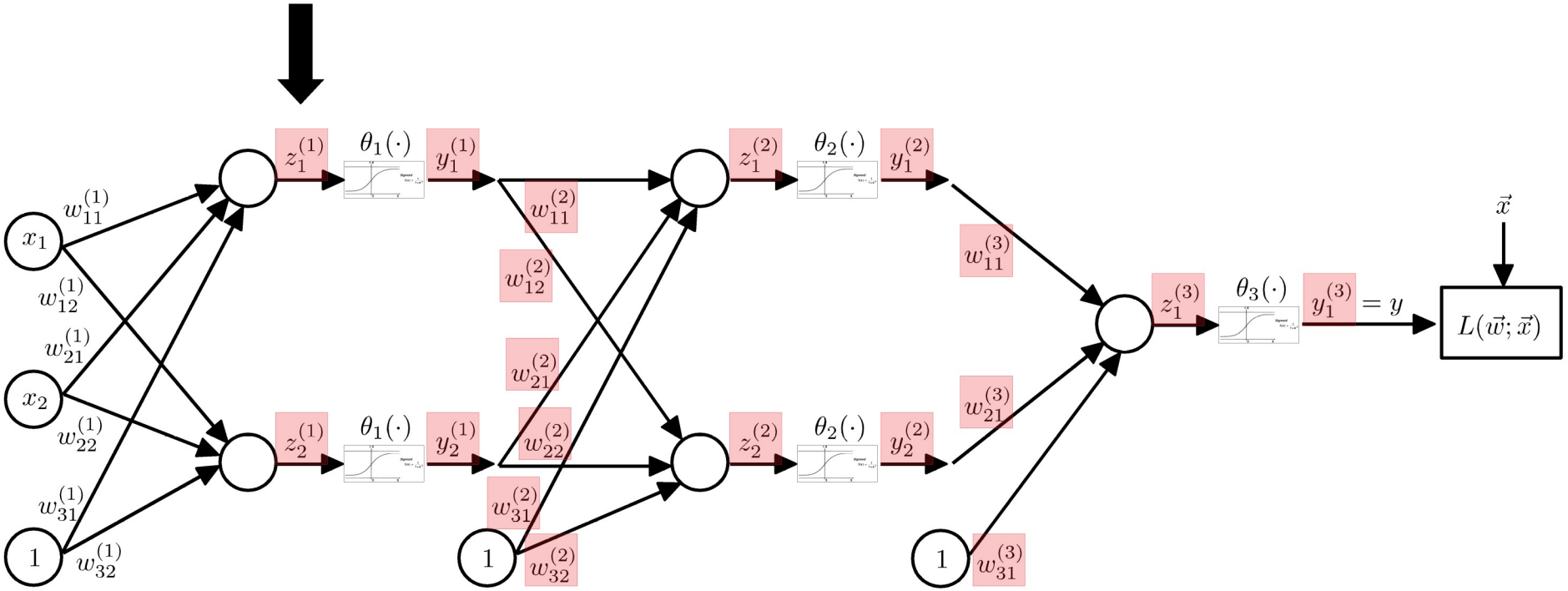


Schritt-2: Backward pass

- Berechne die Ableitungen nach den $\{w_{ij}^{(k)}\}$:

$$\frac{\mathrm{d}L}{\mathrm{d}z_i^{(1)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y_i^{(1)}} \cdot \theta_1'\left(z_i^{(1)}\right)$$

Bereits im vorherigen Schritt berechnet



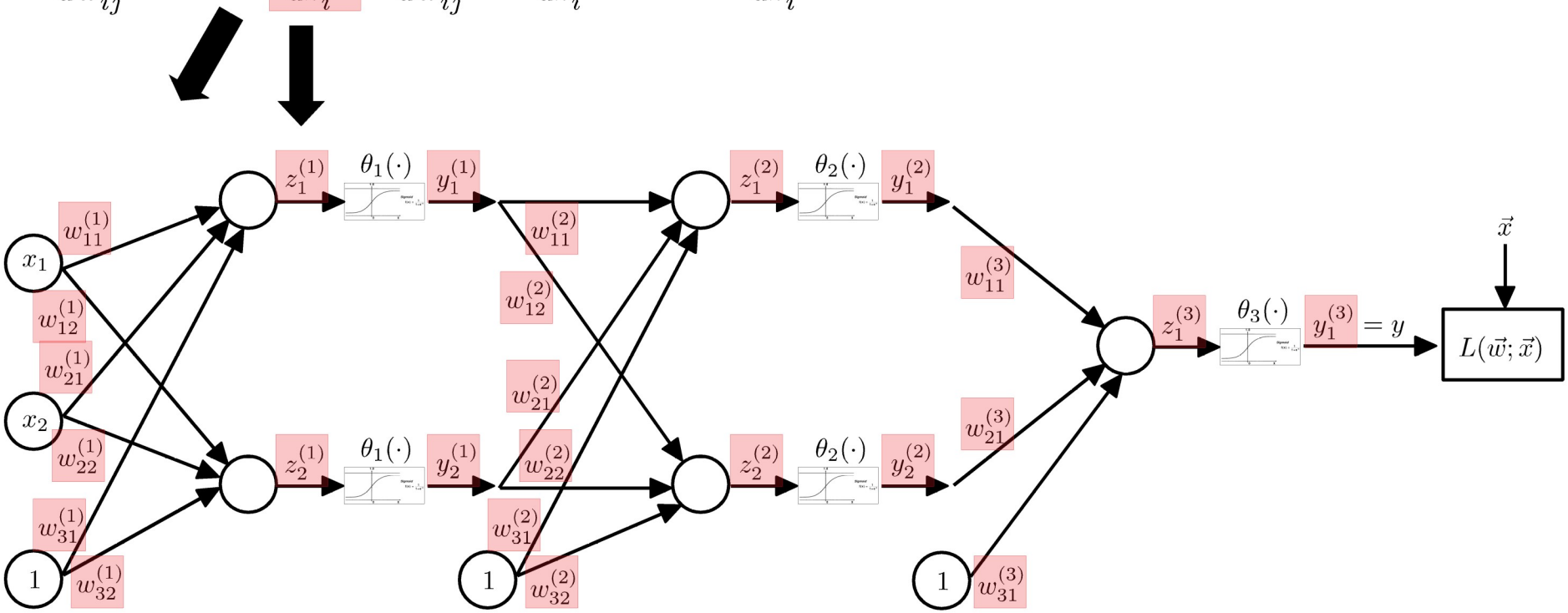
Schritt-2: Backward pass

- Berechne die Ableitungen nach den $\{w_{ij}^{(k)}\}$:

$$\frac{\mathrm{d}L}{\mathrm{d}z_i^{(1)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y_i^{(1)}} \cdot \theta_1'\left(z_i^{(1)}\right)$$

$$\frac{\mathrm{d}L}{\mathrm{d}w_{ij}^{(1)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}z_i^{(1)}} \cdot \frac{\mathrm{d}z_i^{(1)}}{\mathrm{d}w_{ij}^{(1)}} = \frac{\mathrm{d}L}{\mathrm{d}z_i^{(1)}} \cdot y_j^{(0)} = \frac{\mathrm{d}L}{\mathrm{d}z_i^{(1)}} \cdot x_j$$

Bereits im vorherigen Schritt berechnet



Gradientenabstieg – Teil 2

- Algorithmus für den Gradientenabstieg für ein MLP mit N Lagen:

Forward pass:

- Beginne mit: $y_j^{(0)} = x_j$

- Für jede Lage $k = 1 \ldots N$:

$$z_i^{(k)} = \sum_j y_j^{(k-1)} w_{ij}^{(k)}$$

$$y_i^{(k)} = \theta_k \left(z_i^{(k)} \right)$$

Backward pass:

- Beginne für Lage N:

$$\frac{\mathrm{d}L}{\mathrm{d}y}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y^{(N)}}$$

$$\frac{\mathrm{d}L}{\mathrm{d}z^N}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y^{(N)}} \cdot \theta'_N \left(z^N \right)$$

- Für jede Lage $k = (N-1) \ldots 0$:

$$\frac{\mathrm{d}L}{\mathrm{d}w_{ij}^{(k+1)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}z_i^{(k+1)}} \cdot y_j^{(k)}$$

$$\frac{\mathrm{d}L}{\mathrm{d}y_i^{(k)}}(\vec{w};\vec{x}) = \sum_{j=1}^{n} \frac{\mathrm{d}L}{\mathrm{d}z_j^{(k+1)}} \cdot w_{ij}^{(k+1)}$$

$$\frac{\mathrm{d}L}{\mathrm{d}z_i^{(k)}}(\vec{w};\vec{x}) = \frac{\mathrm{d}L}{\mathrm{d}y_i^{(k)}} \cdot \theta'_1 \left(z_i^{(k)} \right)$$

Backup