Stofftransportmodellierung im hydrogeologischen Kontext auf Basis eines künstlichen neuronalen Netzes

Semesterarbeit im Kurs
„Programmieren für Data Science“

Alexander Prinz
Studiengang: Data Science
Fachbereich: Informatik und Sprachen

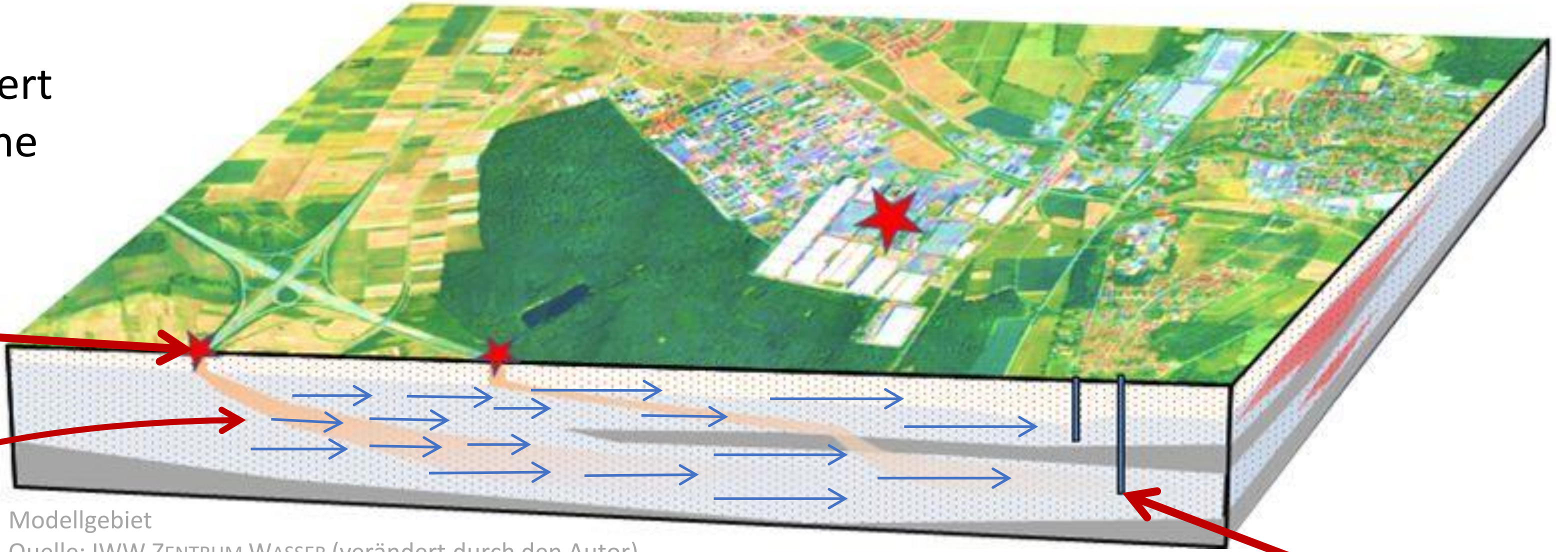
Hochschule Anhalt
Anhalt University of Applied Sciences

- Motivation
  - Beispielszenario zur Anwendung von Stofftransportmodellierung im Grundwasserbereich
- Methoden
  - Konventionelle Methode und ihr Nutzen als Trainingsdatenquelle
    - numerische Lösung der Stofftransportgleichung
    - exemplarische Lösung der Stofftransportgleichung als Animation
    - Implementierung des Solvers und des semistochastischen Permeabilitätsfeldgenerators
  - Künstliches neuronales Netz als mögliche alternative Methode
    - Parametrisierung und Topologie
- Ergebnisse
  - Präsentation/Diskussion einiger exemplarischer Ergebnisse
- Fragen und Antworten

Motivation

Beispielszenario zur Anwendung von Stofftransportmodellierung

Gegeben:
LKW mit x m³
Chemikalie verliert durch Unfall seine Ladung an diesem Ort.



Modellgebiet
Quelle: IWW Zentrum Wasser (verändert durch den Autor)

Grundwasserleiter:
Hydraulisch leitende Formation aus permeablen Sedimenten wie Kies, Sand, Silt und einer Wasserschicht über einer impermeablen Schicht (oftmals bestehend ausTon)

Fragestellung:
Kann man „voraussagen“, ob, und wenn ja, wann an diesem Ort die Chemikalie in welcher Konzentration ankommt?

Methoden

Der Stofftransport kann durch die Lösung einer *hinreichend genau* beschreibenden Differentialgleichung modelliert werden.

Stofftransportgleichung
berücksichtigt in der Grundwassermodellierung: Diffusion, Advektion sowie Quellen/Senken

$$\frac{\partial c}{\partial t} = \underline{\nabla(D \cdot \nabla(c))} + \underline{(-\nabla(v_A \cdot c))} + \underline{\sigma}$$

Substitution der Abstandsgeschwindigkeit v_A durch implizites Darcy-Gesetz: $v_A = \kappa \cdot \nabla(h)$ führt zu:

$$\frac{\partial c}{\partial t} = \nabla(D \cdot \nabla(c)) - \nabla(\kappa \cdot \nabla(h) \cdot c) + \sigma$$

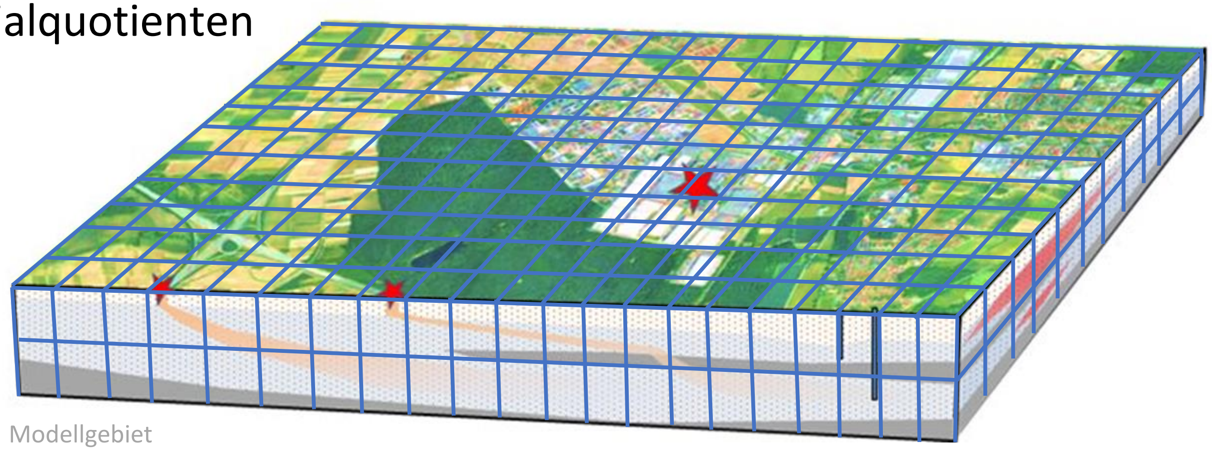
$c = c(\vec{x}, t)$:= Konzentration, t := Zeitpunkt, D:= Diffusionskoeffizient (konstant über Ort und Zeit), $v_A(\vec{x})$:= Abstandsgeschwindigkeit, $\kappa = \kappa(\vec{x})$:= Permeabilitätsfeld,
$h = h(\vec{x})$:= Höhe am Ort $\vec{x}$, ∇:= Nabla-Operator, $\sigma = \sigma(\vec{x}, t)$:= Inhomogenität (Quelle/Senke)

Methoden

- Modellparameter sind häufig nicht stetig.
  -> Stofftransportgleichung nicht analytisch lösbar

- Daher:
  Numerische Lösung der Stofftransportgleichung
  durch Approximation des Differentialquotienten

$$\frac{\partial c}{\partial t} \approx \frac{c^{t+1} - c^{t}}{\Delta t}$$



Modellgebiet
Quelle: IWW Zentrum Wasser (verändert durch den Autor)

Einschrittverfahren zum Lösen von Anfangswertproblemen.
Es gibt verschiedene Lösungsverfahren
Aus Zeitlichen Gründen habe ich das einfachste (explizites Eulerverfahren) angewendet/implementiert.

Methoden

- Solverimplementierung

$$\frac{c_{i,j}^{t+1} - c_{i,j}^{t}}{\Delta t} = D_{i,j}\left(\frac{c_{i+1,j}^{t} - 2c_{i,j}^{t} + c_{i-1,j}^{t}}{(\Delta x_1)^2} + \frac{c_{i,j+1}^{t} - 2c_{i,j}^{t} + c_{i,j-1}^{t}}{(\Delta x_2)^2}\right)$$
$$-c_{i,j}^{t}\left(\frac{\kappa_{i+1,j} - \kappa_{i-1,j}}{2\Delta x_1} + \frac{\kappa_{i,j+1} - \kappa_{i,j-1}}{2\Delta x_2}\right)\left(\frac{h_{i+1,j} - h_{i-1,j}}{2\Delta x_1} + \frac{h_{i,j+1} - h_{i,j-1}}{2\Delta x_2}\right)$$
$$-\kappa_{i,j}\left(\frac{h_{i+1,j} - h_{i-1,j}}{2\Delta x_1} + \frac{h_{i,j+1} - h_{i,j-1}}{2\Delta x_2}\right)\left(\frac{\boldsymbol{c_{i+1,j}^{t} - c_{i-1,j}^{t}}}{\boldsymbol{2\Delta x_1}} + \frac{\boldsymbol{c_{i,j+1}^{t} - c_{i,j-1}^{t}}}{\boldsymbol{2\Delta x_2}}\right)$$
$$+\sigma_{i,j}^{t}$$

Animation der Lösung der Stofftransportgleichung anhand zweier Beispiele auf Basis unterschiedlicher Permeabilitätsfelder…

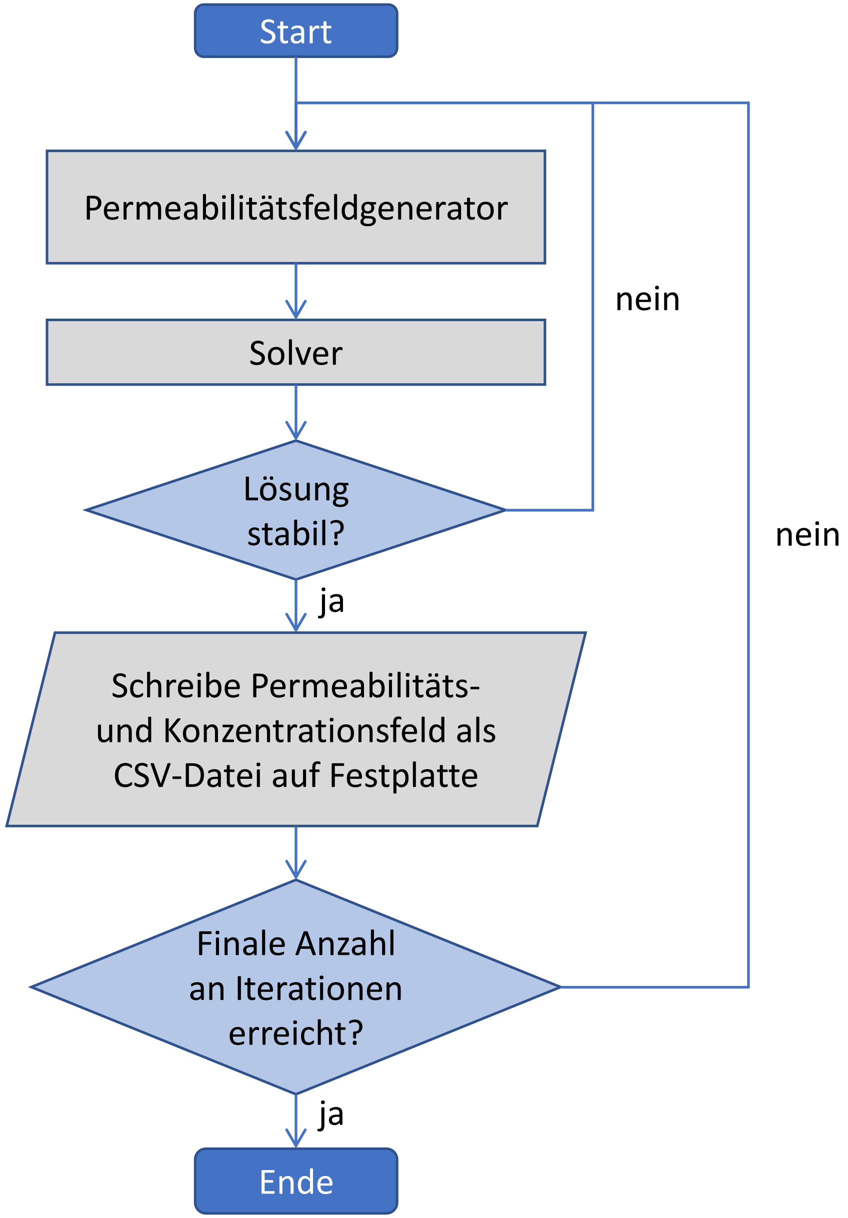
Methoden

- Automatisiertes Generieren von Trainingsdaten durch semistochastisch erzeugte Permeabilitätsfelder gekoppelt mit dem Solver.
  - Es wurde für jeden Fall die Lösung zum 3000sten Zeitschritt berechnet.
  - Instabile Lösungen wurden durch entsprechende Stabilitätskriterien (Neuman‘sches Kriterium, Maximumsnorm) evaluiert und falls nötig verworfen.
  - Valide Lösungen wurden als Ausgabedateien (Permeabilitätsfeld und Konzentrationsfeld) im CSV-Format gespeichert.
  - Permeabilitätsfelder wurden zudem als PNG-Datei zu Visualisierungszwecken pro erzeugten Datensatz gespeichert.

…Noch ein wenig Anschauliches dazu.

Methoden

Schema des Programmablaufs zum Generieren der Trainingsdaten.



Gerechnet wurde anfangs auf zwei Raspberry Pi 4
-> 3 Wochen für 4677 so erzeugte Datentupel.

…dann auf Desktop-Rechner
-> 6 h für weitere 3611 Datentupel.

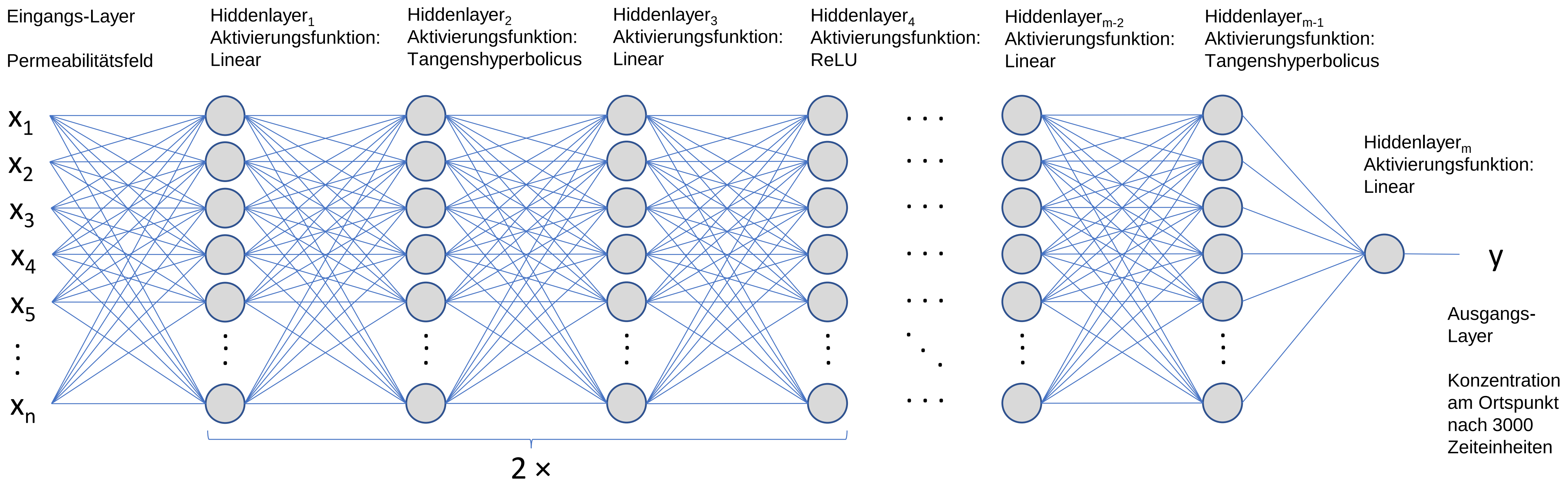
Methoden

Das neuronale Netz

- PyTorch als verwendetes Framework
- Gerechnet wurde final auf einer GPU (GeForce RTX 2060 mit 1920 Shader-Einheiten)
  …dazu by the way ein kleiner Benchmark:
  -> ca. 6 mal schneller als auf Desktop-Rechner-CPU (i9 9900k, 16 Threads / @3.6 GHz)
  -> ca. 18 mal schneller als auf Laptop-CPU (i5, 8 Threads @1.6 GHz
- Zwei Trainingsdatensätze zu 4677 und 8288 Lösungen wurden in des Modell einbezogen
- 2500 Eingabeneuronen (da 50×50 Gitterzellen)
- 11 Hiddenlayer mit unterschiedlichen Aktivierungsfunktionen
- Ein Ausgabeneuron (Konzentrationswert an der definierten Gitterzelle)
- Mehrere Modelle definiert durch unterschiedliche Orte wurden automatisiert gerechnet.

Methoden

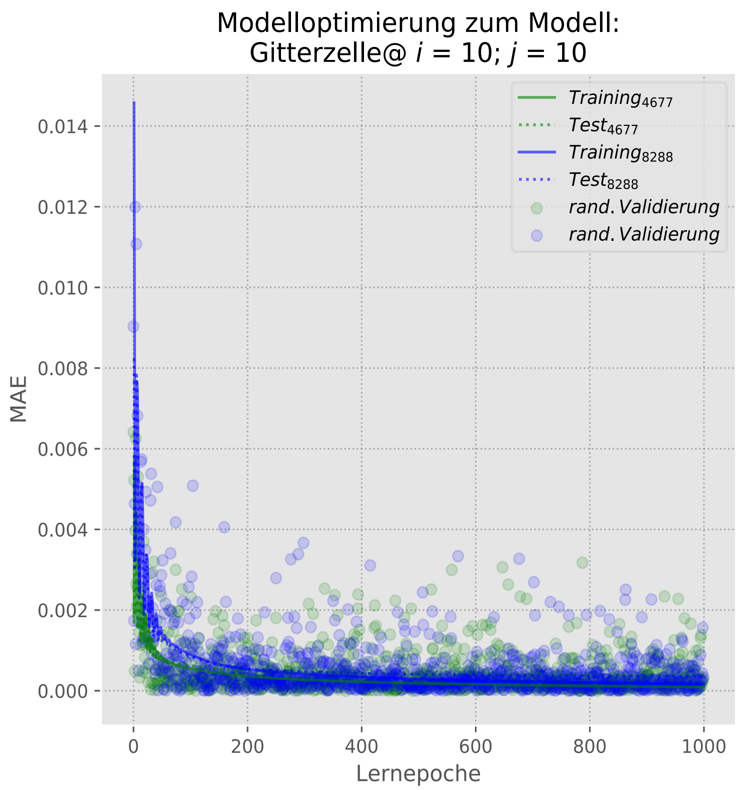
Das künstlich neuronale Netz



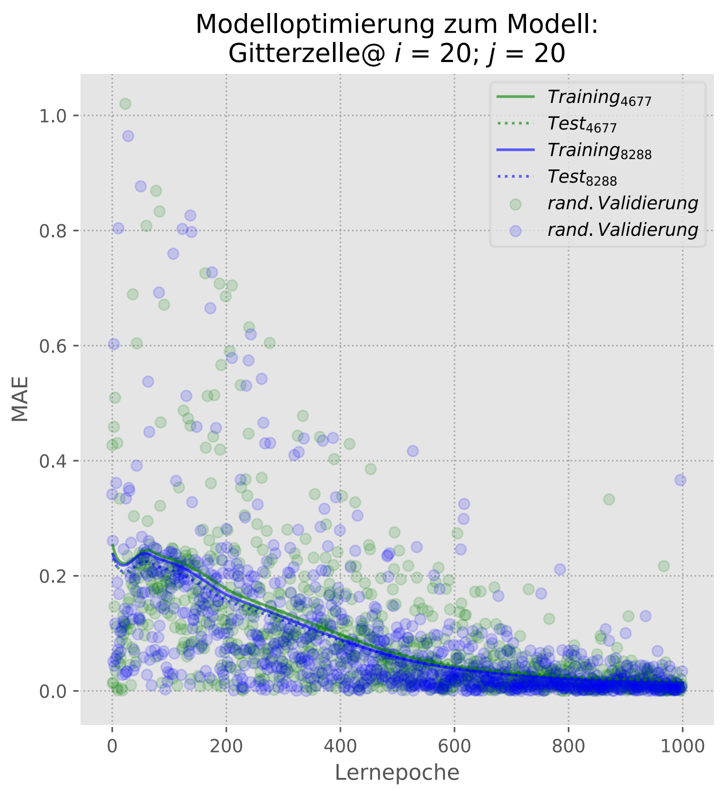
- klassisches Feedforeward-Netz
  -> allen Neuronen sind mit den Neuronen der folgenden Schicht verbunden

Ergebnisse/Diskussion

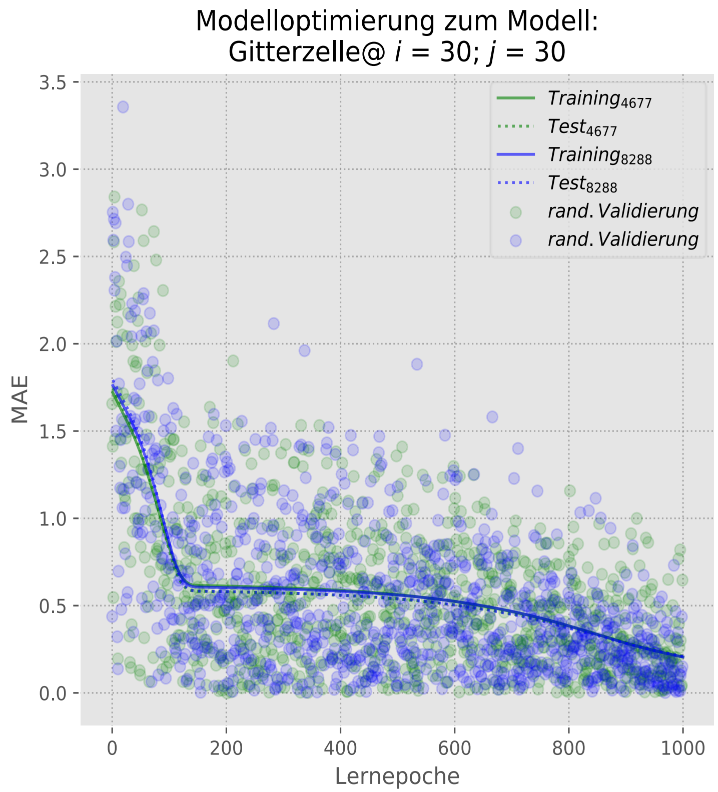
Drei Modellergebnisse:
einziger veränderter Parameter ist die Gitterzellenkoordinate



Modellfehlerverhalten über den Lernzeitraum von 1000 Lernepochen an der Gitterzelle i=10, j=10



Modellfehlerverhalten über den Lernzeitraum von 1000 Lernepochen an der Gitterzelle i=20, j=20



Modellfehlerverhalten über den Lernzeitraum von 1000 Lernepochen an der Gitterzelle i=30, j=30