

Lehre und Lernen auf souveränen Plattformen: KI-Toolbox, LiteLLM und bwJupyter

Philip Hoyer, Matthias Leander-Knoll, Suleiman Elkhoury,
Paul Hoyer,
Scientific Computing Center (SCC),
06.10.2026



1. KI-Toolbox

2. bwJupyter

- Einführung in Jupyter Notebooks
- Landesdienst bwJupyter
- bwJupyter und KI

Einführung in Jupyter Notebooks

Struktur

Jupyter Notebooks sind **interaktive Dokumente** mit

- Code
- Text (Markdown & HTML)
- Formeln (LaTeX)
- Visualisierungen

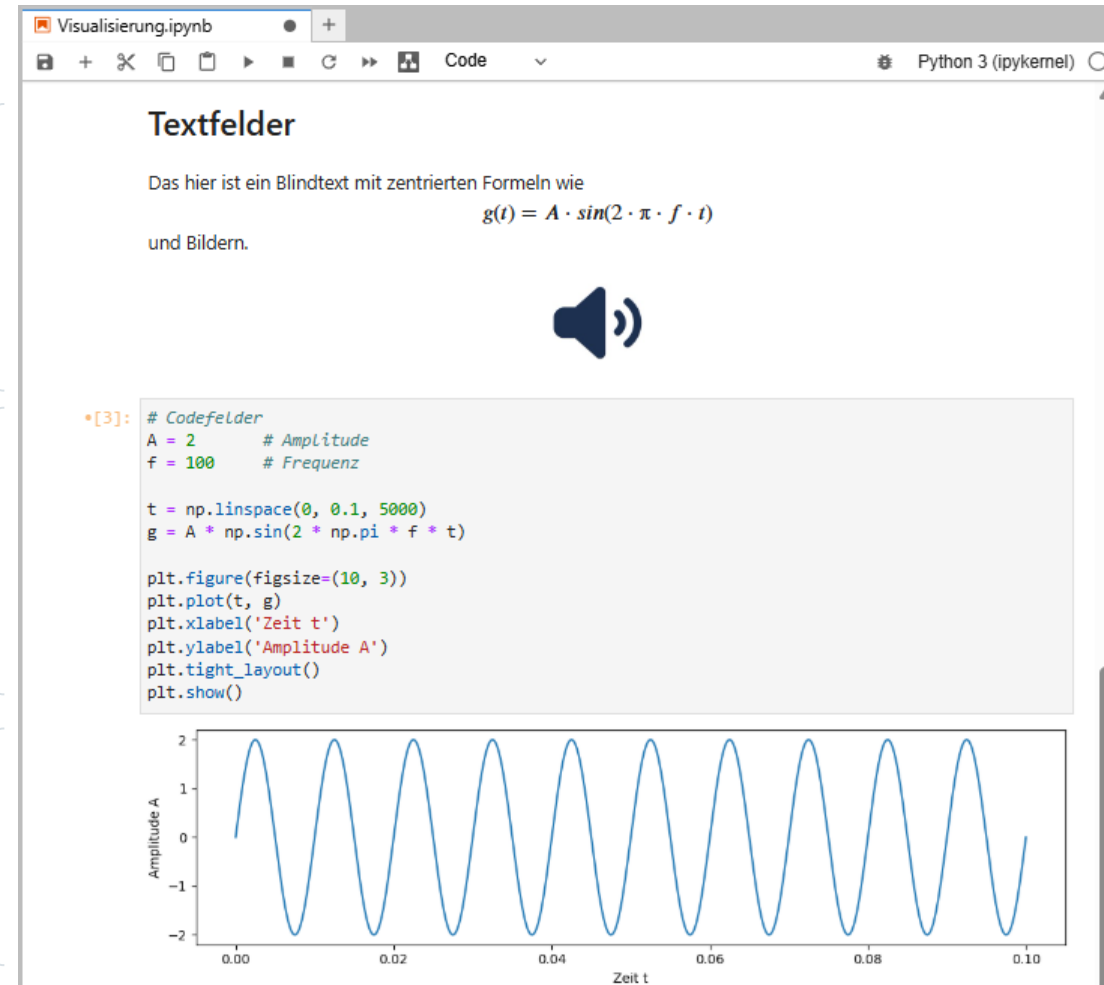
Es werden verschiedene **Programmiersprachen** unterstützt:

- Julia
- Python
- R
- Java, C/C++, Stata, ...

Textfelder

Codefelder

Ausgabe



Einführung in Jupyter Notebooks

Use Cases



1. Jupyter Notebooks als **Computational Essays**

- Computergestützte Modellierungs- und Programmierprozesse und dessen Ergebnisse dokumentieren und reproduzierbar machen
- Authentisches wissenschaftliches Arbeiten
- Bietet Struktur bei ergebnisoffener forschungsorientierter Arbeit

Einführung in Jupyter Notebooks

Use Case: Computational Essays

compEssay.ipynb

Kamerabasierte Artikelerkennung in dm-Filialen

I. Beschreibung des realen Problems

Im Einzelhandel gibt es viele Routinetätigkeiten, die viel Zeit und Personal in Anspruch nehmen, wie z.B. die Regalpflege oder Inventur. Jedoch wird die Anzahl der zur Verfügung stehenden Fachkräfte durch den demografischen Wandel immer geringer, weshalb man immer mehr versucht, die Mitarbeiter durch technische Hilfsmittel zu unterstützen. Unser Ziel war es daher, optimale Positionen für Kameras zu finden, die in einer dm-Filiale fehlende Artikel in Regalen feststellen können, damit man zielgerichtet Artikel wieder auffüllen kann. Dies sollte für 90-100% der Artikel möglich sein.



Zentral waren bei uns folgende Fragen:

- Wo sind die optimalen Kamerapositionen, wenn man annimmt, dass die Kameras auf einer Höhe positioniert werden?
- Wie kann man das nun auf die dritte Dimension übertragen, um die jeweils notwendige Höhe und Anordnung der Kameras zu bestimmen?

II. Die Daten

Die Produkte in den Filialregalen sind in einer Tabelle gegeben und nach `aisle` (Gang), `side` (linke oder rechte Seite des Gangs) und `shelf` (einzelne Regalabschnitte in einer Regalreihe) sortiert. Jede Zeile entspricht einem Artikel im Regal und für alle vier Ecken des Sichtfensters des Artikels sind die Koordinaten `x`, `y` und `z` angegeben. Am Anfang eines Gangs sitzt immer der Ursprung, von dem aus die `x`-Achse den Gang entlang verläuft, während die `y`-Achse orthogonal zu ihr liegt und die `z`-Achse nach oben zeigt. Der Datensatz beinhaltet 6 Gänge mit jeweils zwei Seiten, in denen 525 verschiedene Artikel Platz finden.

+ 10 cells hidden

III. Modell 1: Tracing Mapping Algorithm

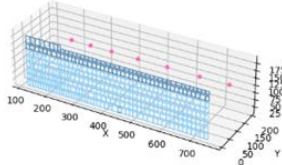
Trace Mapping Algorithm / x-Richtung, darauf aufbauend z-Richtung

Auch wenn diese Funktion nur für eine eindimensionale Positionierung der Kamera optimal ist, so kann man sie trotzdem ebenfalls auf die Gänge anwenden. Die dabei entstandenen Ergebnisse benötigen sehr wenige Kameras, haben aber oft in den oberen oder unteren Regalböden eine geringere Abdeckung.

```
[10]: # Erstelle einen Zähler, um die Anzahl an benötigten Kameras zu erfassen
counter = 0
covs = []

# Gruppieren die Produkte nach Reihe und Seite und führe dafür jeweils aus:
for w in df.groupby(['aisle', 'side']):
    # Führe die Methode 'set_cams_x' aus. Dabei werden die Daten aus dem entsprechenden Gang und von der entsprechenden Seite ausgeführt. Als Dateiname für das Bild werden ebenfalls Gang und Seite verwendet. Die Länge der Rückgabe, also die Anzahl an Kameras, wird dem Zähler hinzugefügt
    cams, cov = set_cams_x(w['aisle'] == w[0][0] & (w['side'] == w[0][1]), f'img{w[0][0]}-{w[0][1]}')
    counter += len(cams)
    covs += cov * len(df[(df['aisle'] == w[0][0]) & (df['side'] == w[0][1])])
# Schlussendlich wird die Anzahl an Kameras auf der Console ausgegeben.
print(f"Es werden insgesamt {counter} Kameras benötigt.")
print(f"Insgesamt werden {covs/len(df)} der Produkte abgedeckt.")

Finished calculating products.
Es sind 75.89042095890411 % aller Produkte durch die Kameras abgedeckt.
```



Einführung in Jupyter Notebooks

Use Cases



1. Jupyter Notebooks als **Computational Essays**

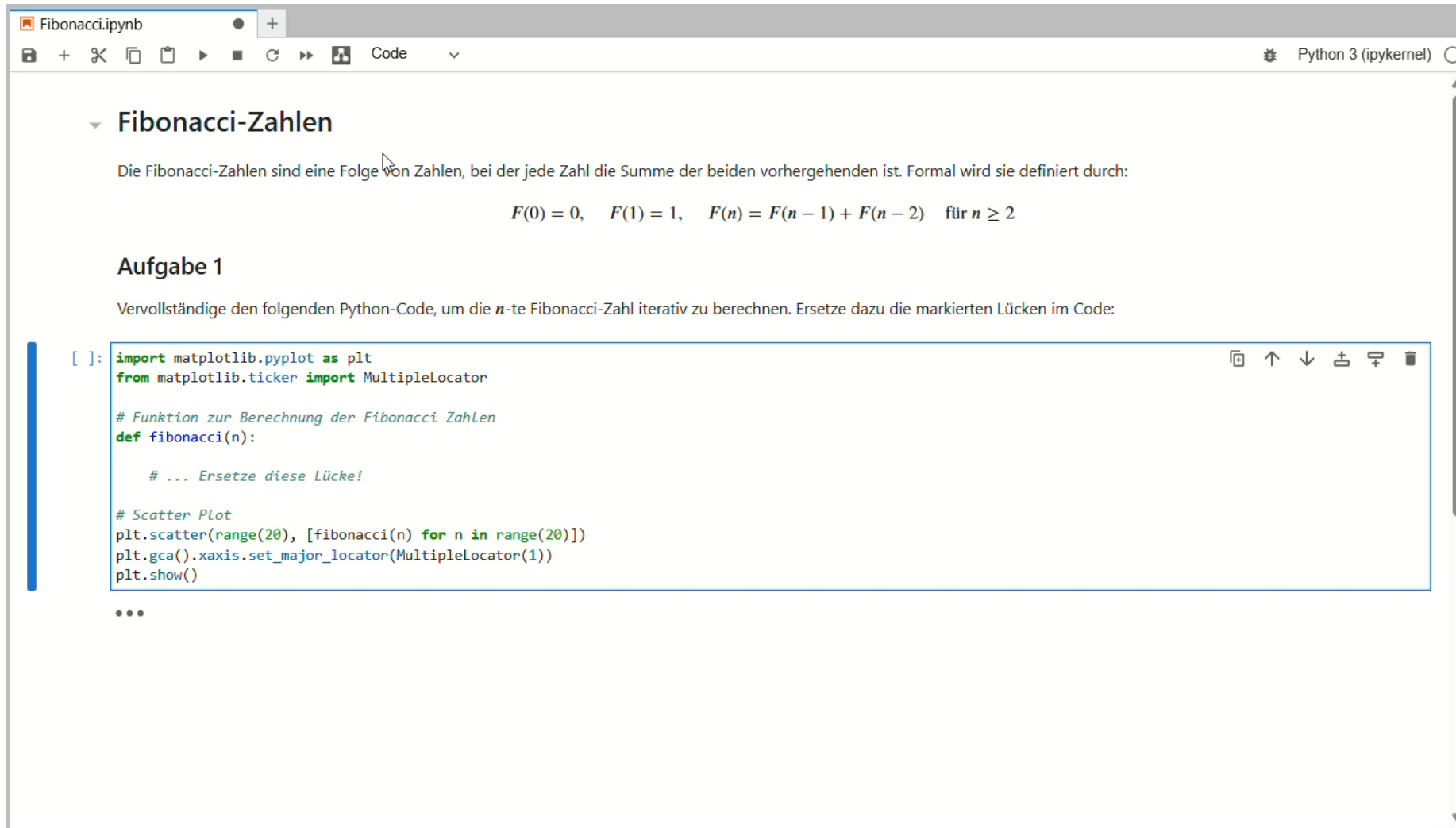
- Computergestützte Modellierungs- und Programmierprozesse und dessen Ergebnisse dokumentieren und reproduzierbar machen
- Authentisches wissenschaftliches Arbeiten
- Bietet Struktur bei ergebnisoffener forschungsorientierter Arbeit

2. Jupyter Notebooks als **Übungsblätter**

- „Fill in the gap“

Einführung in Jupyter Notebooks

Use Case: Übungsblatt



Fibonacci-Zahlen

Die Fibonacci-Zahlen sind eine Folge von Zahlen, bei der jede Zahl die Summe der beiden vorhergehenden ist. Formal wird sie definiert durch:

$$F(0) = 0, \quad F(1) = 1, \quad F(n) = F(n-1) + F(n-2) \quad \text{für } n \geq 2$$

Aufgabe 1

Vervollständige den folgenden Python-Code, um die n -te Fibonacci-Zahl iterativ zu berechnen. Ersetze dazu die markierten Lücken im Code:

```
[ ]: import matplotlib.pyplot as plt
from matplotlib.ticker import MultipleLocator

# Funktion zur Berechnung der Fibonacci Zahlen
def fibonacci(n):
    # ... Ersetze diese Lücke!

# Scatter Plot
plt.scatter(range(20), [fibonacci(n) for n in range(20)])
plt.gca().xaxis.set_major_locator(MultipleLocator(1))
plt.show()
```

...

Einführung in Jupyter Notebooks

Use Cases



1. Jupyter Notebooks als **Computational Essays**
 - Computergestützte Modellierungs- und Programmierprozesse und dessen Ergebnisse dokumentieren und reproduzierbar machen
 - Authentisches wissenschaftliches Arbeiten
 - Bietet Struktur bei ergebnisoffener forschungsorientierter Arbeit
2. Jupyter Notebooks als **Übungsblätter**
 - „Fill in the gap“
3. Jupyter Notebooks als **interaktives Skript**
 - Interaktiv ausführbare Elemente
 - Verknüpfung von Theorie und Praxis

Einführung in Jupyter Notebooks

Use Case: Interaktives Skript

56_DiscussionMC.ipynb

Code

Open in... Python 3 (ipykernel)

56 Discussion of Monte Carlo

- The MC method suffers from the curse of dimensionality, according to our definition. MC works, though, for the right class of functions, e.g. $g \in F_d^1$.
- Another specialty of MC is that we have an error equality

$$\varepsilon_{RMS} = \left(E \left[(Qg - \bar{Q}_N g)^2 \right] \right)^{\frac{1}{2}} = \frac{\sqrt{\text{Var}(g)}}{\sqrt{N}}$$

This formula has to be taken with a grain of salt. In the frequentist interpretation of randomness, it means that we need to repeat the MC experiment often enough to actually observe the rate $\frac{1}{2}$.

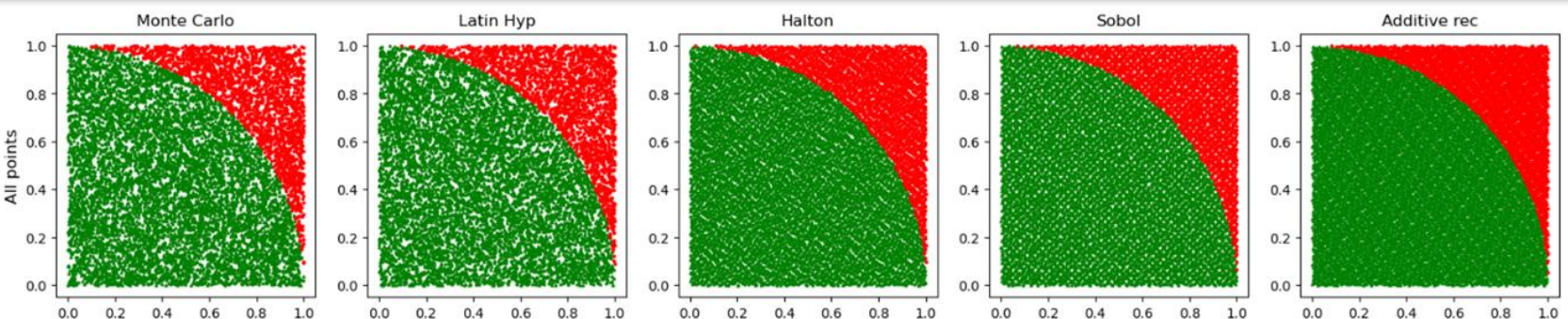
- The major advantage of MC (and MLMC) is that in high dimensions, it is the only possibility, because we simply cannot construct a grid. A non-trivial grid has at least 2^d points. Because $2^{10} \approx 10^3$, in 100 dimensions we need $2^{100} \approx (10^3)^{10} = 10^{30}$ grid points.

EXERCISE 56.1)

In this exercise, we revisit Exercise 6.1, in which we studied the convergence of (quasi) Monte-Carlo. Go over the exercise again and think about how your answers (especially regarding task (d)) differs from your answer when you started the course. Discuss which method should be used for which problem to mitigate the curse of dimensionality.

a) In this section we compare different random numbers to approximate the value of π . What does the code do exactly?

```
def generateSamples (N): ...
```



Einführung in Jupyter Notebooks

Use Cases



1. Jupyter Notebooks als **Computational Essays**
 - Computergestützte Modellierungs- und Programmierprozesse und dessen Ergebnisse dokumentieren und reproduzierbar machen
 - Authentisches wissenschaftliches Arbeiten
 - Bietet Struktur bei ergebnisoffener forschungsorientierter Arbeit
2. Jupyter Notebooks als **Übungsblätter**
 - „Fill in the gap“
3. Jupyter Notebooks als **interaktives Skript**
 - Interaktiv ausführbare Elemente
 - Verknüpfung von Theorie und Praxis
4. Jupyter Notebooks als **geführte Lernumgebung**
 - Automatische Rückmeldung
 - Differenzierungsmöglichkeiten

Einführung in Jupyter Notebooks

Use Case: Geführte Lernumgebung

AB5-SuS.ipynb

Python 3 (ipykernel)

Arbeitsblatt 5 | Bewertung des Spracherkenners

Auf Arbeitsblatt 4 haben wir ein erstes funktionierendes Modell für einen Spracherkennner erstellt, indem das Muster eines neuen Sprachsignals mit vorhandenen Referenzmustern abgeglichen wurde. Auf diesem Arbeitsblatt wollen wir bewerten, wie gut der Spracherkennner ist. Wir probieren den Spracherkennner dafür systematisch aus und lesen Sprachsignale ein, deren Wortbedeutung und somit deren Klassenzugehörigkeit bereits bekannt ist. So können wir überprüfen, wie gut die Wortbedeutung über unser entwickeltes Verfahren erkannt wird. Hierfür nutzen wir eine Strategie aus dem Bereich des überwachten Maschinellen Lernens und teilen in einem ersten Schritt unsere Daten in Trainings- und Testdaten auf. Die Trainingsdaten nutzen wir um Referenzmuster zu bilden. Die Testdaten werden zur Bewertung des Spracherkenners genutzt.

Das Diagramm zeigt den Prozess der Bewertung eines Spracherkenners. Es beginnt mit einer Datenbank 'Sprachsignale mit bekannter Wortbedeutung'. Diese werden in 'Trainingsdaten' und 'Testdaten' unterteilt. Die Trainingsdaten werden verwendet, um Referenzmuster (z.B. 'one', 'two', 'three', 'four') zu erstellen. Ein Testsignal wird mit diesen Mustern verglichen ('Abstand berechnen') und dann klassifiziert ('klassifizieren'). Das Ergebnis ist ein 'erkanntes Wort', das mit dem 'tatsächlichen Wort' verglichen wird.

```
graph TD; A[Sprachsignale mit bekannter Wortbedeutung] --> B[Trainingsdaten]; A --> C[Testdaten]; B --> D[„one“ „two“ „three“ „four“]; D --> E[erkanntes Wort]; C --> E; E --> F[tatsächliches Wort];
```

1) Die Trainings- und Testdaten

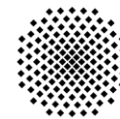
Im Hintergrund haben wir die Daten bereits in Trainings- und Testdaten unterteilt und die Testdaten zurückbehalten. Wir lesen nun sowohl Trainings- als auch Testdaten ein. Im folgenden Codefeld werden die Referenzmuster, die wir bereits auf Arbeitsblatt 3 aus den Trainingsdaten ermittelt haben, sowie deren jeweilige Klassenzugehörigkeit geladen. Außerdem werden die Testdaten eingelesen und wie in Arbeitsblatt 3 die zu den Signalen zugehörigen Muster gebildet. Die Klassenzugehörigkeit ist zwar auch bei den Testdaten bekannt, für die Klassifizierung wird diese

Landesdienst bwJupyter

Der JupyterHub für die Lehre



- bwJupyter als Maßnahmepaket aus dem **Dialogprozess** „Zukunftslabor Hochschulen in der digitalen Welt“
- **Kooperation** zwischen Karlsruher Institut für Technologie (KIT) und Universität Stuttgart
- **Förderung des Ministerium für Wissenschaft, Forschung und Kunst (MWK)** von April 2024 bis Dezember 2025
 - Bedarfsumfrage zu Anforderungen mit Dozierenden
 - Pilotphase an ausgewählten Hochschulen
 - Hardware-Beschaffung
 - Temporärer Zugang zu bwJupyter für alle Hochschulen des Landes
- Übergang vom MWK-geförderten Projekt zum **Regelbetrieb** eines Landesdienstes
- **Umlage der Kosten** auf teilnehmende Hochschulen



Universität Stuttgart



Baden-Württemberg
Ministerium für Wissenschaft,
Forschung und Kunst

15.165

Registrierte Nutzende

22

Teilnehmende Hochschulen

493

Aktive Profile von AI/ML bis Biologie und Medizin

Profile Selection

Shared Profiles

The shared profiles are created by lecturers. By clicking on a shared profile link, the profile is added to your list. You can remove profiles on the page [Profile Management](#).

FOKI_Vorstellung_KI_Assistent

Dieses Profil dient der Vorstellung des integrierten KI-Assistenten in den Profilen von bwJupyter und wurde für die Fortbildungsreihe FOKI entwickelt.

Start

KCDS Summer School 2026

This profile is used for the KCDS Summer School practical sessions on GNNs.

Start as group

Manage group

Grading enabled

Linear Algebra in Data Science

This profile contains the sample codes for the course "Linear Algebra in Data Science" held at the University of Konstanz in the winter term 2025/2026.

Start

System Engineering und KI - Praktikum

System Engineering und KI - Praktikum

Start

Collaborator

Grading enabled

Landesdienst bwJupyter

Feature-Übersicht



Dienst

-  Leicht zugänglich und geräteunabhängig
-  Begleitmaterialien für Dozierende
-  Eigene Profile für Dozierende
-  Schnelles und unkompliziertes Abgabesystem
-  Umfassendes und automatisches Bewertungssystem
-  Nutzung von Rechenpower (CPU, RAM, GPU)
-  Bereitstellung gängiger Pakete und Tools (z.B. PyTorch)
-  Kollaboratives Arbeiten und Gruppenarbeit in Echtzeit
-  Zugriff auf Lehr- und Lernmaterialien (ZOERR)
-  KI-Assistent

Projekt

-  Schneller Support
-  Offen für Feedback und Verbesserungsvorschläge
-  Transparente Entwicklung und Nutzung

bwJupyter und KI

Chat-Bot fürs Erstellen von Jupyter Notebooks



Chat-Bot mit lokalen Modellen der KI-Toolbox:

- Mistral Small 4
- DeepSeek V4 Flash
- GLM-5.3

Funktionalitäten:

- Fehler im Code beheben
- Notebooks erstellen
- Code erklären lassen
- (Fragen zu Dateien stellen)
- (Inline Completion)
- Chat-Oberfläche

Modellauswahl

The screenshot displays the JupyterLab environment. On the left, the Jupyter AI chat interface is visible, featuring a menu icon (a square with a circle inside) and a list of commands: `/ask`, `/clear`, `/export`, `/fix`, `/generate`, `/help`, and `/learn`. Below this list, there is a text input field labeled "Ask Jupyter AI" and a "Simple" toggle switch. On the right, the Launcher sidebar is open, showing a "Notebook" section with three options: "Python 3 (ipykernel)", "Julia 1.11.4", and "Pluto Notebook [?]", as well as an "R" option. A "Console" section is also visible at the bottom of the sidebar. The top of the interface shows a menu bar with "File", "Edit", "View", "Run", "Kernel", "Git", "Tabs", "Settings", and "Help". A gear icon in the top right corner of the main area is highlighted with a blue box, indicating the "Modellauswahl" (Model Selection) feature.

KI-Assistent

bwJupyter und KI

Chat-Bot fürs Erstellen von Jupyter Notebooks



Demo

bwJupyter und KI

Materialien zu KI und Maschinelles Lernen



- Über 2.800 OER-Materialien, davon 146 mit Jupyter Notebooks
 - Python-Einführungskurs
 - KI-Modell zur Hautkrebserkennung
 - Einführung in klassische Convolutional Neural Networks (CNN) anhand LeNet-5 und AlexNet
 - Einführung in moderne CNNs anhand InceptionV3
- Materialien können direkt über bwJupyter ausprobiert werden
- 40+ Workshops mit Fokus auf mathematische Modellierung echter Probleme für Schülerinnen und Schüler
- Geleitete Lernumgebung anhand Jupyter Notebooks
- Keine Programmierkenntnisse notwendig
- Workshops mit KI- und ML-Bezug
 - Alpha Zero (Reinforcement Learning)
 - Convolutional Neural Networks
 - Funktionen und KI (Neuronale Netze)
 - Lebenserwartung (Lineare Regression bis Neuronale Netze)
 - Spamfilter (Bayes-Klassifikator)
 - Wortvorschläge (N-Gramm-Modelle)

bwJupyter und KI

Materialien zu KI und Maschinelles Lernen



Demo

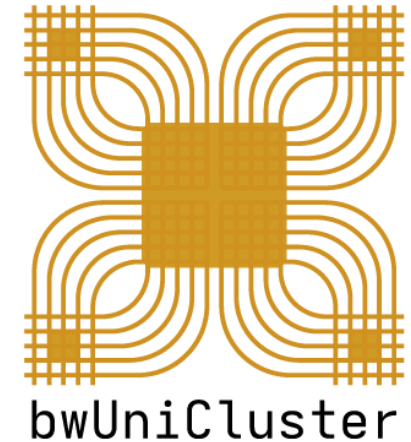
bwJupyter und KI

Einstieg in HPC und Übergang zum bwUniCluster



Niederschwelliger Einstieg in HPC durch bwJupyter

- Keine lokale Installation notwendig
- Verwandte Frameworks und Technologien werden verwendet
- Nutzung von GPUs „im kleinen Stile“



Verfügbare Compute-Ressourcen pro Nutzende

- 8 bis 16 GB RAM
- 2 CPU Kerne
- 6 GB vRAM (NVIDIA L40S)
- 2 GB persönliches Nutzerverzeichnis
- 10 GB geteiltes Profilverzeichnis

Beispiel-Profil mit Jupyter Notebooks



<https://tinyurl.com/bwj-hed>

bwJupyter und KI

Einstieg in HPC und Übergang zum bwUniCluster



Demo

bwJupyter Community Meeting

Save the Date



Neue
Features
vorstellen

Fragerunde



Live-
Demos

Verbesserungs-
vorschläge

Ausblick

Montag, den **12. Oktober 2026, 15 Uhr**
via Zoom

Vielen Dank für Ihre
Aufmerksamkeit.

 support@bwjupyter.de

 <https://hub.bwjupyter.de>

 <https://www.bwjupyter.de>

