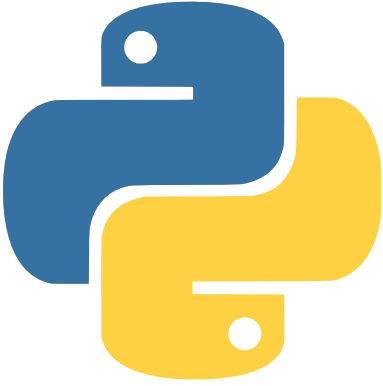
Universität Stuttgart
Projekt digit@L – BOOST. SKILLS. SUPPORT.



Programmierkurs Python

matplotlib: Weitere Anwendungsmöglichkeiten

Dominik Göddeke

IANS SimTech

Stiftung Innovation in der Hochschullehre

matplotlib: Weitere Anwendungsmöglichkeiten

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt
  - Subplots: Kombination mehrerer Plots in einer Graphik

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt
  - Subplots: Kombination mehrerer Plots in einer Graphik
  - Manuelle Annotationen innerhalb eines Plots

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt
  - Subplots: Kombination mehrerer Plots in einer Graphik
  - Manuelle Annotationen innerhalb eines Plots
  - Säulendiagramme

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt
  - Subplots: Kombination mehrerer Plots in einer Graphik
  - Manuelle Annotationen innerhalb eines Plots
  - Säulendiagramme
  - Histogramme

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt
  - Subplots: Kombination mehrerer Plots in einer Graphik
  - Manuelle Annotationen innerhalb eines Plots
  - Säulendiagramme
  - Histogramme
  - Scatterplots

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt
  - Subplots: Kombination mehrerer Plots in einer Graphik
  - Manuelle Annotationen innerhalb eines Plots
  - Säulendiagramme
  - Histogramme
  - Scatterplots
  - Bildbearbeitung

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt
  - Subplots: Kombination mehrerer Plots in einer Graphik
  - Manuelle Annotationen innerhalb eines Plots
  - Säulendiagramme
  - Histogramme
  - Scatterplots
  - Bildbearbeitung
  - Interaktive Plots

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt
  - Subplots: Kombination mehrerer Plots in einer Graphik
  - Manuelle Annotationen innerhalb eines Plots
  - Säulendiagramme
  - Histogramme
  - Scatterplots
  - Bildbearbeitung
  - Interaktive Plots
- Lediglich Kratzen an der Oberfläche

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt
  - Subplots: Kombination mehrerer Plots in einer Graphik
  - Manuelle Annotationen innerhalb eines Plots
  - Säulendiagramme
  - Histogramme
  - Scatterplots
  - Bildbearbeitung
  - Interaktive Plots
- Lediglich Kratzen an der Oberfläche
  - `https://matplotlib.org/stable/gallery/index.html`

Weitere Anwendungsmöglichkeiten

- In diesem Abschnitt
  - Subplots: Kombination mehrerer Plots in einer Graphik
  - Manuelle Annotationen innerhalb eines Plots
  - Säulendiagramme
  - Histogramme
  - Scatterplots
  - Bildbearbeitung
  - Interaktive Plots
- Lediglich Kratzen an der Oberfläche
  - `https://matplotlib.org/stable/gallery/index.html`
  - `https://matplotlib.org/stable/tutorials/index.html#tutorials`

Subplots: Kombination mehrerer Plots in einer Graphik

Subplots: Kombination mehrerer Plots in einer Graphik

- Anwendungsbeispiel: mehrere Plots untereinander oder nebeneinander in einer Grafik

Subplots: Kombination mehrerer Plots in einer Graphik

- Anwendungsbeispiel: mehrere Plots untereinander oder nebeneinander in einer Grafik
- `subplot(n,m,p)`

Subplots: Kombination mehrerer Plots in einer Graphik

- Anwendungsbeispiel: mehrere Plots untereinander oder nebeneinander in einer Grafik
- `subplot(n,m,p)`
  - `n,m`: Anzahl der Plots, n Zeilen und m Spalten

Subplots: Kombination mehrerer Plots in einer Graphik

- Anwendungsbeispiel: mehrere Plots untereinander oder nebeneinander in einer Grafik
- `subplot(n,m,p)`
  - `n,m`: Anzahl der Plots, n Zeilen und m Spalten
  - `p` Auswahl des Plots im Gitter

Subplots: Kombination mehrerer Plots in einer Graphik

- Anwendungsbeispiel: mehrere Plots untereinander oder nebeneinander in einer Grafik
- `subplot(n,m,p)`
  - `n,m`: Anzahl der Plots, n Zeilen und m Spalten
  - `p` Auswahl des Plots im Gitter
- Achsen teilbar, viele andere Anpassungen möglich

Subplots: Kombination mehrerer Plots in einer Graphik

- Anwendungsbeispiel: mehrere Plots untereinander oder nebeneinander in einer Grafik
- `subplot(n,m,p)`
  - `n,m`: Anzahl der Plots, n Zeilen und m Spalten
  - `p` Auswahl des Plots im Gitter
- Achsen teilbar, viele andere Anpassungen möglich
- Details: `https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.subplot.html`

Codebeispiel

Manuelle Annotationen innerhalb eines Plots

Manuelle Annotationen innerhalb eines Plots

- Didaktisch sinnvoll: „herummalen“ in einem Plot

Manuelle Annotationen innerhalb eines Plots

- Didaktisch sinnvoll: „herummalen“ in einem Plot
  - Markierung bestimmter Aspekte, Steuerung der Aufmerksamkeit

Manuelle Annotationen innerhalb eines Plots

- Didaktisch sinnvoll: „herummalen“ in einem Plot
  - Markierung bestimmter Aspekte, Steuerung der Aufmerksamkeit
- `annotate()`, mit Argumenten in dieser Reihenfolge

Manuelle Annotationen innerhalb eines Plots

- Didaktisch sinnvoll: „herummalen“ in einem Plot
  - Markierung bestimmter Aspekte, Steuerung der Aufmerksamkeit
- `annotate()`, mit Argumenten in dieser Reihenfolge
  - Zeichenkette mit Annotationstext

Manuelle Annotationen innerhalb eines Plots

- Didaktisch sinnvoll: „herummalen“ in einem Plot
  - Markierung bestimmter Aspekte, Steuerung der Aufmerksamkeit
- `annotate()`, mit Argumenten in dieser Reihenfolge
  - Zeichenkette mit Annotationstext
  - Koordinaten, im Koordinatensystem der Achsenbeschriftung entspricht

Manuelle Annotationen innerhalb eines Plots

- Didaktisch sinnvoll: „herummalen“ in einem Plot
  - Markierung bestimmter Aspekte, Steuerung der Aufmerksamkeit
- `annotate()`, mit Argumenten in dieser Reihenfolge
  - Zeichenkette mit Annotationstext
  - Koordinaten, im Koordinatensystem der Achsenbeschriftung entspricht
  - Koordinaten des eigentlichen Texts für Pfeilchen zwischen beiden Koordinaten

Manuelle Annotationen innerhalb eines Plots

- Didaktisch sinnvoll: „herummalen“ in einem Plot
  - Markierung bestimmter Aspekte, Steuerung der Aufmerksamkeit
- `annotate()`, mit Argumenten in dieser Reihenfolge
  - Zeichenkette mit Annotationstext
  - Koordinaten, im Koordinatensystem der Achsenbeschriftung entspricht
  - Koordinaten des eigentlichen Texts für Pfeilchen zwischen beiden Koordinaten
  - Eigenschaften des Pfeils in Form eines `dict`

Manuelle Annotationen innerhalb eines Plots

- Didaktisch sinnvoll: „herummalen“ in einem Plot
  - Markierung bestimmter Aspekte, Steuerung der Aufmerksamkeit
- `annotate()`, mit Argumenten in dieser Reihenfolge
  - Zeichenkette mit Annotationstext
  - Koordinaten, im Koordinatensystem der Achsenbeschriftung entspricht
  - Koordinaten des eigentlichen Texts für Pfeilchen zwischen beiden Koordinaten
  - Eigenschaften des Pfeils in Form eines `dict`
- Details: `https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.annotate.html`

Codebeispiel

Säulendiagramme

Säulendiagramme

- `bar()`, mit Argumenten in dieser Reihenfolge

Säulendiagramme

- `bar()`, mit Argumenten in dieser Reihenfolge
  - `x`: darzustellende Abszissen, bspw. als Liste oder NumPy-ndarray

Säulendiagramme

- `bar()`, mit Argumenten in dieser Reihenfolge
  - `x`: darzustellende Abszissen, bspw. als Liste oder NumPy-ndarray
  - `height`: darzustellende Ordinaten, identisch indiziert wie `x`

Säulendiagramme

- `bar()`, mit Argumenten in dieser Reihenfolge
  - `x`: darzustellende Abszissen, bspw. als Liste oder NumPy-ndarray
  - `height`: darzustellende Ordinaten, identisch indiziert wie `x`
- Details: `https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.bar.html`

Codebeispiel

Histogramme

Histogramme

- Zweck: automatisches „Einsortieren“ in gegebene Wertebereiche

Histogramme

- Zweck: automatisches „Einsortieren“ in gegebene Wertebereiche
- `hist()`, mit folgenden Argumenten

Histogramme

- Zweck: automatisches „Einsortieren“ in gegebene Wertebereiche
- `hist()`, mit folgenden Argumenten
  - `x`: Eingabewerte, bspw. als Liste oder NumPy-ndarray

Histogramme

- Zweck: automatisches „Einsortieren“ in gegebene Wertebereiche
- `hist()`, mit folgenden Argumenten
  - `x`: Eingabewerte, bspw. als Liste oder NumPy-ndarray
  - `bins (optional)`: Anzahl der „Eimer“ zur Sortierung

Histogramme

- Zweck: automatisches „Einsortieren“ in gegebene Wertebereiche
- `hist()`, mit folgenden Argumenten
  - `x`: Eingabewerte, bspw. als Liste oder NumPy-ndarray
  - `bins (optional)`: Anzahl der „Eimer“ zur Sortierung
    - Falls Zahl (Default 10): interpretiert als Anzahl Eimer, Intervall der Eingabewerte äquidistant unterteilt

Histogramme

- Zweck: automatisches „Einsortieren“ in gegebene Wertebereiche
- `hist()`, mit folgenden Argumenten
  - `x`: Eingabewerte, bspw. als Liste oder NumPy-ndarray
  - `bins (optional)`: Anzahl der „Eimer“ zur Sortierung
    - Falls Zahl (Default 10): interpretiert als Anzahl Eimer, Intervall der Eingabewerte äquidistant unterteilt
    - Falls geordnete Liste: definiert bins als halboffene Intervalle

Histogramme

- Zweck: automatisches „Einsortieren“ in gegebene Wertebereiche
- `hist()`, mit folgenden Argumenten
  - `x`: Eingabewerte, bspw. als Liste oder NumPy-ndarray
  - `bins (optional)`: Anzahl der „Eimer“ zur Sortierung
    - Falls Zahl (Default 10): interpretiert als Anzahl Eimer, Intervall der Eingabewerte äquidistant unterteilt
    - Falls geordnete Liste: definiert bins als halboffene Intervalle
  - `rwidth (optional)`: Breite der Säulen, relativ zur Gesamtbreite

Histogramme

- Zweck: automatisches „Einsortieren“ in gegebene Wertebereiche
- `hist()`, mit folgenden Argumenten
  - `x`: Eingabewerte, bspw. als Liste oder NumPy-ndarray
  - `bins (optional)`: Anzahl der „Eimer“ zur Sortierung
    - Falls Zahl (Default 10): interpretiert als Anzahl Eimer, Intervall der Eingabewerte äquidistant unterteilt
    - Falls geordnete Liste: definiert bins als halboffene Intervalle
  - `rwidth (optional)`: Breite der Säulen, relativ zur Gesamtbreite
- Details: `https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.hist.html`

Codebeispiel

Punktwolken

Punktwolken

- „Primitivste“ graphische Darstellung von Daten

Punktwolken

- „Primitivste“ graphische Darstellung von Daten
- Wenn (initial) kein funktionaler Zusammenhang zugrunde liegt oder bekannt

Punktwolken

- „Primitivste“ graphische Darstellung von Daten
- Wenn (initial) kein funktionaler Zusammenhang zugrunde liegt oder bekannt
- Daten im $\mathbb{R}^2$ oder $\mathbb{R}^3$, Plot entsprechend

Punktwolken

- „Primitivste“ graphische Darstellung von Daten
- Wenn (initial) kein funktionaler Zusammenhang zugrunde liegt oder bekannt
- Daten im $\mathbb{R}^2$ oder $\mathbb{R}^3$, Plot entsprechend
- `scatter(x, y)`, mit Argumenten

Punktwolken

- „Primitivste“ graphische Darstellung von Daten
- Wenn (initial) kein funktionaler Zusammenhang zugrunde liegt oder bekannt
- Daten im $\mathbb{R}^2$ oder $\mathbb{R}^3$, Plot entsprechend
- `scatter(x, y)`, mit Argumenten
  - `x` und `y`: Vektoren oder Listen oder allgemein identisch indizierbar

Punktwolken

- „Primitivste“ graphische Darstellung von Daten
- Wenn (initial) kein funktionaler Zusammenhang zugrunde liegt oder bekannt
- Daten im $\mathbb{R}^2$ oder $\mathbb{R}^3$, Plot entsprechend
- `scatter(x, y)`, mit Argumenten
  - `x` und `y`: Vektoren oder Listen oder allgemein identisch indizierbar
  - Datum identifiziert als Tupel `(x[i],y[i])`, analog in 3D

Punktwolken

- „Primitivste“ graphische Darstellung von Daten
- Wenn (initial) kein funktionaler Zusammenhang zugrunde liegt oder bekannt
- Daten im $\mathbb{R}^2$ oder $\mathbb{R}^3$, Plot entsprechend
- `scatter(x, y)`, mit Argumenten
  - `x` und `y`: Vektoren oder Listen oder allgemein identisch indizierbar
  - Datum identifiziert als Tupel `(x[i],y[i])`, analog in 3D
- Details: `https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.scatter.html`

Codebeispiel

Bilddateien

Bilddateien

- Unterpaket `image`: Methoden zur Ein- und Ausgabe von Bildern

Bilddateien

- Unterpaket `image`: Methoden zur Ein- und Ausgabe von Bildern
- Bild: im Wesentlichen 2D Array bei Grauwertbildern oder Farbindex-Bildern, bzw. 3D/4D Array bei RGB oder RGBA Farbbildern

Bilddateien

- Unterpaket `image`: Methoden zur Ein- und Ausgabe von Bildern
- Bild: im Wesentlichen 2D Array bei Grauwertbildern oder Farbindex-Bildern, bzw. 3D/4D Array bei RGB oder RGBA Farbbildern
- Ermöglicht Bildbearbeitung in Python, bspw. Kantenerkennung, Weichzeichnung oder andere Filter

Codebeispiel

Interaktive Plots – direkt im Code

Impressum, Danksagung und Quellen

Stiftung Innovation in der Hochschullehre

Gefördert durch die Stiftung Innovation in der Hochschullehre im Rahmen des Projekts digit@L, https://stiftung-hochschullehre.de

SimTech

Gefördert mit Mitteln der Deutschen Forschungsgemeinschaft (EXC 2075 - 390740016) im Rahmen der Exzellenzstrategie

---

Autor: Dominik Göddeke, IANS, Universität Stuttgart

ZOERR

Veröffentlicht auf dem Zentralen OER Repositorium Baden-Württemberg, https://www.zoerr.de