



Programmierung und Deskriptive Statistik

BSc Psychologie WiSe 2023/24

Belinda Fleischmann

Inhalte basieren auf Programmierung und Deskriptive Statistik von Dirk Ostwald, lizenziert unter CC BY-NC-SA 4.0

Herzlich willkommen!

Aufnahme läuft.



(1) Einführung

Formalia

Grundbegriffe der Informatik

R und Visual Studio Code

Selbstkontrollfragen

Formalia

Grundbegriffe der Informatik

R und Visual Studio Code

Selbstkontrollfragen

Homepage der Abteilung für Methodenlehre I



INSTITUT | STUDIUM | FORSCHUNG | PERSONEN

Home > Institut > Abteilungen des Ins... > Methodenlehre I : Experimentelle und Neuro... > Forschung | Lehre | Team

Methodenlehre I : Experimentelle und Neurowissenschaftliche Psychologie

Forschung



Lehre



Team



C1. Programmierung und Deskriptive Statistik

- Einführung in die datenanalytische Programmierung
- Einführung in die Auswertung deskriptiver Statistiken mit R in Visual Studio Code

C2. Analyse und Dokumentation


- Praktische Analyse empirischer Daten
- Dokumentation empirischer Studien und Analysen

- Termine: Mittwochs in Raum G05-307
 - Gruppe 1 um 13-15 Uhr
 - Gruppe 2 um 11-13 Uhr
- Kursmaterialien (Folien, Videos, Source Code) auf der [Kurswebseite](#)
- Code auf [Github](#)
- Ankündigungen über die [Moodleseite](#)
- Vorherige Iteration des Kurses: [PDS \(WS 2022/23\)](#)
- Empfohlene Vorbereitung: [Vorkurs "Mathematische Grundlagen"](#)
- Q&A im [Mattermost-Channel](#)
 - Einmalige [Registrierung](#) zum Team "bsc-psy-2023"
- Zweiteiliger Leistungsnachweis: Unbenotet, Multiple Choice, digitales Format
 - Teil 1 vor der Weihnachtspause
 - Teil 2 am Semesterende

Termine

Datum	Einheit	Thema
11.10.23	Einführung	(1) Einführung
18.10.23	R Grundlagen	(2) R und Visual Studio Code
25.10.23	R Grundlagen	(2) R und Visual Studio Code
01.11.23	R Grundlagen	(3) Vektoren
08.11.23	R Grundlagen	(4) Matrizen
15.11.23	R Grundlagen	(5) Listen und Dataframes
22.11.23	R Grundlagen	(6) Datenmanagement
29.11.23	Deskriptive Statistik	(7) Häufigkeitsverteilungen
06.12.23	Deskriptive Statistik	(8) Verteilungsfunktionen und Quantile
13.12.23	Deskriptive Statistik	(9) Maße der zentralen Tendenz
20.12.23	<i>Leistungsnachweis Teil 1</i>	
20.12.23	Deskriptive Statistik	(10) Maße der Datenvariabilität
	Weihnachtspause	
10.01.24	Deskriptive Statistik	(11) Anwendungsbeispiel (Deskriptive Statistik)
17.01.24	Inferenzstatistik	(12) Anwendungsbeispiel (Parameterschätzung, Konfidenzintervalle)
24.01.24	Inferenzstatistik	(13) Anwendungsbeispiel (Hypothesentest)
25.01.24	<i>Leistungsnachweis Teil 2</i>	

Webseite des Kurses (Folien, Videos)



OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG

INSTITUT FÜR PSYCHOLOGIE

Sitemap Impressum Kontakt

Suchbegriff

🔍

INSTITUT | STUDIUM | FORSCHUNG | PERSONEN

DIREKTLINKS ▾

Home > Methodenlehre I: E. > Lehre > Wintersemester 2024 > Programmierung und Deskriptive Statistik

Programmierung und Deskriptive Statistik

Dieser Kurs gibt eine Einführung in die datenanalytische Programmierung und die Auswertung deskriptiver Statistiken mit `R` in `Visual Studio Code`.

Nach der [Studien- und Prüfungsordnung](#) für den BSc Psychologie (06/2020) und dem [Modulhandbuch](#) für den BSc Psychologie (09/2020) entspricht dieser Kurs dem Modul C1 Computergestützte Datenanalyse.

Der Quarto Code der Vorlesungstollen ist auf [GitHub](#) verfügbar.

Als weiterführende Literatur zur Programmierung mit R werden empfohlen:


- Sauer, S. (2019) [Moderne Datenanalyse mit R](#)
- Wickham, H. (2019) [Advanced R](#)
- Cotton, R. (2013) [Learning R](#).
- Murrell, P. (2021) [R Graphics](#)

Als Einstieg in die Deskriptive Statistik bieten sich an

- Henze, N. (2016) [Deskriptive Statistik](#)
- Fahrmeier et al. (2016) [Statistik, Kapitel 1 - 3](#)

Vorlesungseinheiten

(1) Einführung



Kontakt

Abteilungsleitung

• Prof. Dr. Dirk Ostwald
dirk.ostwald@ovgu.de
Tel.: + 49 391 67 57370

Abteilungsassistentenz

• Birgit Müller
birgit.mueller@ovgu.de
Tel.: +49 391 67 58464

Anschrift

Otto-von-Guericke-Universität
Magdeburg
Institut für Psychologie
Universitätsplatz 2
Gebäude 24
39106 Magdeburg

[Anfahrt](#)

Letzte Änderung: 06.10.2023 - Ansprechpartner: [Webmaster](#)

Git-repository des Kurses (Folien, Source Codes)

<> **Code** Issues Pull requests Actions Projects Security Insights

main 1 branch 0 tags Go to file Code

belindamef remove notes and data from repo 489994 4 minutes ago 15 commits

1_Einfuehrung	Delete adv folders from repo	20 hours ago
Abbildungen	Merge branch "main" of github.com:belindamef/prog-und-deskr-st...	3 weeks ago
Header.tex	Update	3 weeks ago
README.md	Update README	9 minutes ago
R_common.R	Add templates	last month
Referenzen.bib	Add templates	last month

☰ README.md

Programmierung und Deskriptive Statistik (WiSe 2023/2024)

Allgemeine Informationen

Dieses Repository enthält alle Skripte für den Kurs "Programmierung und Deskriptive Statistik" (Modul C1) im Wintersemester 2023/2024 bei Belinda Fleischmann an der OVGU Magdeburg.

Alle Lehrmaterialien werden im Sinne der Open Education frei über die ["Website der Abteilung"](#) bereit gestellt.

About

No description, website, or topics provided.

- Readme
- Activity
- 0 stars
- 1 watching
- 0 forks

Report repository

Releases

No releases published

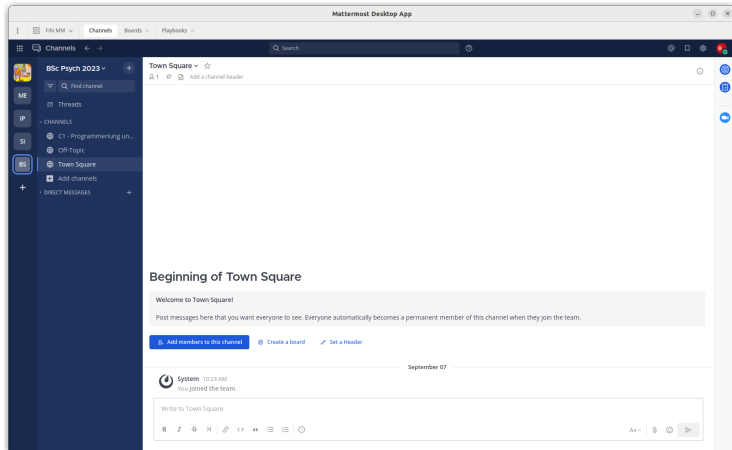
Packages

No packages published

Languages

TeX 99.8% R 0.2%

Mattermost-Team BSc Psych 2023



Q & A

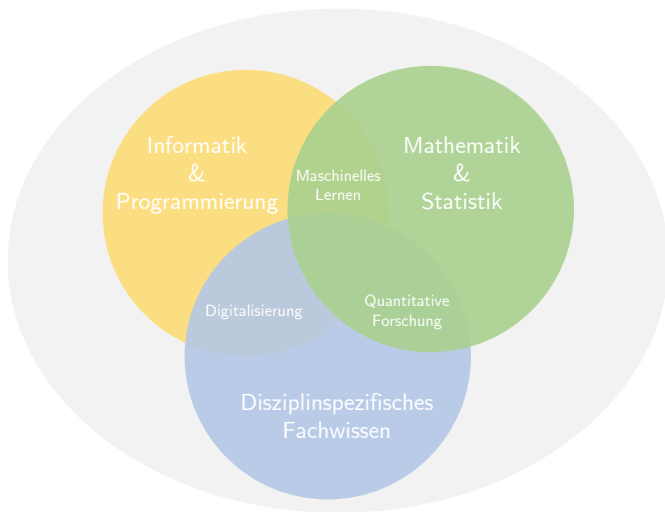
Formalia

Grundbegriffe der Informatik

R und Visual Studio Code

Selbstkontrollfragen

Zentrale Komponenten der Datenwissenschaft



Formalia

Grundbegriffe der Informatik

- Datenanalyse
- Informatik
- Rechnerarchitektur
- Algorithmen und Programme

R und Visual Studio Code

Selbstkontrollfragen

Formalia

Grundbegriffe der Informatik

- **Datenanalyse**
- Informatik
- Rechnerarchitektur
- Algorithmen und Programme

R und Visual Studio Code

Selbstkontrollfragen

- Wissenschaftliche Daten liegen heutzutage als digitale Daten vor.
- Digitale Daten werden mit Hilfe eines Computers analysiert.
- Zur Analyse von digitalen Daten schreibt man Computerprogramme.
- Diese Computerprogramme heißen Datenanalyseskripte.

1. Einlesen und Bereinigen eines digitalen Datensatzes.
2. Berechnung und Visualisierung deskriptiver Statistiken.
3. Probabilistische Modellierung und Inferenz.
4. Dokumentation und Präsentation der Ergebnisse.

Typische Werkzeuge zur Analyse psychologischer Daten

- **R** (frei, Datenwissenschaft, Statistik, Psychologie)
- **Python** (frei, Datenwissenschaft, Anwendung)
- **Matlab** (kommerziell, Engineering, Neuroimaging)

Altmodisch

- **SPSS** (kommerziell, Sozialwissenschaften, Psychologie)
- **JMP** (kommerziell, Biologie, Psychologie)
- **STATA** (kommerziell, Wirtschaftswissenschaften)

Programmiersprachen Trends

PYPL Index (Stand: September 2023)

Worldwide, Sept 2023 :

Rank	Change	Language	Share	1-year trend
1		Python	27.99 %	+0.1 %
2		Java	15.9 %	-1.1 %
3		JavaScript	9.36 %	-0.1 %
4		C#	6.67 %	-0.4 %
5		C/C++	6.54 %	+0.3 %
6		PHP	4.91 %	-0.4 %
7		R	4.4 %	+0.2 %
8		TypeScript	3.04 %	+0.2 %
9	↑↑	Swift	2.64 %	+0.6 %
10		Objective-C	2.15 %	+0.1 %
11	↑↑	Rust	2.12 %	+0.5 %
12	↓↓↓	Go	2.0 %	-0.1 %
13	↓	Kotlin	1.78 %	-0.0 %
14		Matlab	1.58 %	+0.1 %
15		Ruby	1.05 %	-0.1 %

- PopularitY of Programming Language
- Basierend auf GoogleSuchanfragen zu Programmiersprachentutorials

- Dokumentation aller Schritte von Rohdaten bis zur Datenvisualisierung.
- Reproduktion wissenschaftlicher Ergebnisse durch Dritte.
- Essentieller Teil wissenschaftlicher Publikationen.
- Essentieller Teil täglicher wissenschaftlicher Arbeit.

- Die Digitalisierung betrifft insbesondere auch die Wissenschaft.
- Forschungsdatenmanagement ist eine akute Herausforderung.
- Programmierung als zentrales Handwerkszeug wissenschaftlicher Arbeit.
- Informatikkenntnisse sind in der Arbeitswelt unverzichtbar.
- Dies gilt auch für Psychotherapeut:innen (z.B. Online-Intervention).

Formalia

Grundbegriffe der Informatik

- Datenanalyse
- **Informatik**
- Rechnerarchitektur
- Algorithmen und Programme

R und Visual Studio Code

Selbstkontrollfragen

Informatik (engl. Computer Science)

Bei der Informatik handelt es sich um die Wissenschaft von der systematischen Darstellung, Speicherung, Verarbeitung und Übertragung von Informationen, wobei besonders die automatische Verarbeitung mit Computern betrachtet wird. Sie ist zugleich Grundlagen- und Formalwissenschaft als auch Ingenieurdisziplin.

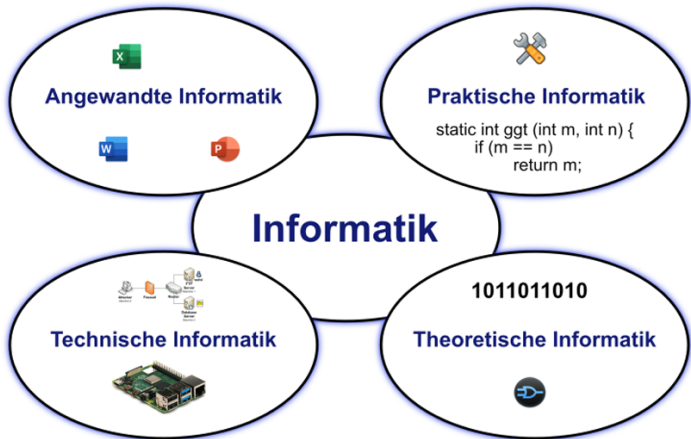
Wikipedia

Computer

- Maschinen zum Datenspeichern und Ausführen einfacher Datenoperationen.
- Einfache Operationen mit extrem hoher Geschwindigkeit.
- Universalität durch Speicherung von Daten und Programmen.

Algorithmen und Programme

- *Programme* sind in einer *Programmiersprache* verfasste *Algorithmen*.
- Algorithmen sind Folgen von Anweisungen durchzuführender Operationen.
- Bei Algorithmen unterscheidet man
 - Beschreibung (Kochrezept, IKEA Bauanleitung, R Skript)
 - Anweisungen (“Mehl und Wasser vermengen”, $o - - -, x = c(1,2,3)$)
 - Durchführung (Kochvorgang, Zusammenbau, R Skript laufen lassen)



Hattenhauer (2020) Informatik

Technische Informatik

- Mikroprozessortechnik, Rechnerarchitektur, Netzwerktechnik

Theoretische Informatik

- Automatentheorie, Berechenbarkeitstheorie, Komplexitätstheorie

Praktische Informatik

- Programmierung, Algorithmen, Datenbanken

Angewandte Informatik

- Anwendungssoftware, Human-Computer-Interaction, Informatik und Gesellschaft

Maschinelles Lernen und Künstliche Intelligenz

- Datenanalyse aus Sicht der Informatik

Computervisualistik

- Bilderkennung und Bildsynthese, Virtuelle Realität, Augmented Reality

Computerlinguistik

- Spracherkennung und Sprachsynthese

Bioinformatik

- Lebenswissenschaften, Genomik, Bildgebende Verfahren der Medizin

Formalia

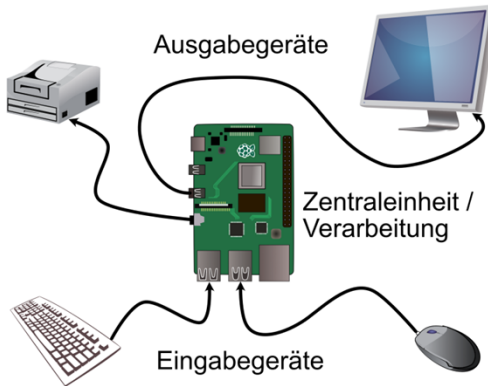
Grundbegriffe der Informatik

- Datenanalyse
- Informatik
- **Rechnerarchitektur**
- Algorithmen und Programme

R und Visual Studio Code

Selbstkontrollfragen

EVA-Prinzip: Eingabe → Verarbeitung → Ausgabe



Hattenhauer (2020) Informatik

Zentraleinheit eines Computers

Auch Hauptplatine, Motherboard oder Mainboard genannt



Figure 1: Hattenhauer (2020) Informatik

Hattenhauer (2020) Informatik

Zentraleinheit eines Computers

CPU (Central Processing Unit/Mikroprozessor)

- Rechenwerk, Steuerwerk, und Leitwerk des Systems
- Cache (flüchtiger schneller Speicher)
- Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz

RAM (Random Access Memory)

- Temporärer, flüchtiger Arbeitsspeicher des Systems
- Begrenzt, z.B. 16 GB

Massenspeicher

- Stationärer Speicher des Systems
- SSD (Solid State Drive), Cloudspeicher

GPU (Graphical Processing Unit)

- Leistungsstarke, speziell für Visualisierung optimierte Prozessoren
- Unterstützung der CPU in manchen Anwendungen, z.B. Neuronale Netze

Von Neumann-Architektur

Abstraktion eines Rechensystems mit wohldefinierten Komponenten und Datenflüsse.

Rechner := Steuerwerk, Rechenwerk, Speicher, Eingabewerk, Ausgabewerk.

Zentrale Eigenschaften

- Struktur des Rechners unabhängig von dem zu bearbeitenden Problem.
- Daten, Programme, Zwischen- und Endergebnisse liegen im gleichen Speicher.
- Speicher ist in gleichgroße nummerierte (adressierte) Zellen unterteilt.
- Über die Adresse einer Speicherzelle kann deren Inhalt abgerufen verändert werden.
- Ein Programm ist eine Reihe von Befehlen.
- Aufeinanderfolgende Befehle eines Programms liegen in benachbarten Speicherzellen und werden entsprechend nacheinander aufgerufen.

→ Die Architektur eines Rechners impliziert das Grundprinzip der Programmierung:
Befehle werden streng sequentiell abgearbeitet.

Formalia

Grundbegriffe der Informatik

- Datenanalyse
- Informatik
- Rechnerarchitektur
- **Algorithmen und Programme**

R und Visual Studio Code

Selbstkontrollfragen

Vom Realweltproblem zum Programm

Realweltproblem

- Das Problem, das mithilfe eines Computers gelöst werden soll.
- z.B. Auswertung von Fragebogendaten einer psychologischen Studie.

Problemspezifikation

- Genaue sprachliche Fassung des Realweltproblems.
- z.B. Methodenteil einer wissenschaftlichen Publikation.

Algorithmus

- Folge von Anweisungen zur Lösung des Problems.
- z.B. Dateneinlesen, deskriptive Statistiken berechnen, T-Test durchführen.

Programm

- Ein Algorithmus, der von einem Computer ausgeführt werden kann.
- Eine in einer Programmiersprache verfasste Textdatei.

Definition (Algorithmus)

Ein *Algorithmus* ist eine Folge von Anweisungen, um aus gewissen Eingabedaten bestimmte Ausgabedaten herzuleiten, wobei folgende Bedingungen erfüllt sein müssen

- *Fintheit*. Die Anweisungsfolge muss in einem endlichen Text vollständig beschrieben sein.
- *Effektivität*. Jede Anweisung muss tatsächlich ausführbar sein.
- *Terminierung*. Der Algorithmus endet nach endlich vielen Anweisungen.
- *Determiniertheit*. Der Ablauf des Algorithmus ist zu jedem Punkt fest vorgeschrieben.

Wenn E die Menge der zulässigen Eingabedaten und A die Menge der zulässigen Ausgabedaten bezeichnet, dann ist ein Algorithmus eine Funktion

$$f : E \rightarrow A, e \mapsto f(e) \quad (1)$$

Umgekehrt heißen Funktionen, die durch einen Algorithmus beschrieben werden können, *berechenbare Funktionen*.

Bemerkung

- Effektivität sollte nicht mit Effizienz verwechselt werden.

Eine Programmiersprache

- ... bestimmt die Regeln, denen ein Programm gehorchen muss.
- ... definiert eine Syntax, also Vokabular und Programmaufbau.
- ... definiert Semantik, also die Bedeutung der erlaubten Anweisungen.

```
#if [ -z "$USER_NAME" -o -z "$USER_TYPE" -o -z "$GROUP" ]
if [ -z "$USER_NAME" -o -z "$USER_TYPE" ]
then
#     echo "Please set the user name, type and group"
     echo "Please set the user name and type"
     exit 1
fi

# generate a random password
# -y: include special characters
# -n: include numbers
# -l: one generated passwords per Line
#pwgen -y 15 -n 5 -l
echo "Propositions for random passwords to use in next step:"
pwgen -s -n -l 15 5

# add the user
# requires password to be given via input
adduser --firstuid 1000 --lastuid 9999 --no-create-home ${USER_NAME}
```

Maschinensprache

- Elementare Operationsbefehle (z.B. Speichern, Vergleichen, Addieren)
- Elementare Operationsbefehle werden als Binärzahlen kodiert

Addiere Inhalt R1 zu Inhalt R2 \Rightarrow 1001 0010

Erhöhe Inhalt R1 um 1 \Rightarrow 1001 0110

Übertrage Inhalt R1 nach R3 \Rightarrow 0010 0011

- Programme in Maschinensprache heißen *Maschinenprogramme*.
- De facto führt ein Computer nur Maschinenprogramme aus.
- Für Menschen ist die Programmierung in Maschinensprache mühselig.

Höhere Programmiersprache

- An die menschliche Sprache angelehnte Wörter und Sätze
- Interpreter oder Compiler übersetzen Programme in Maschinensprache
- R, Python, Matlab, C++, Java, FORTRAN, COBOL,...

1. Generation (1GL)

- Maschinensprachen
- 10110000 01100001 (hexadezimaler Darstellung des Ausdrucks "B0 61")

2. Generation (2GL)

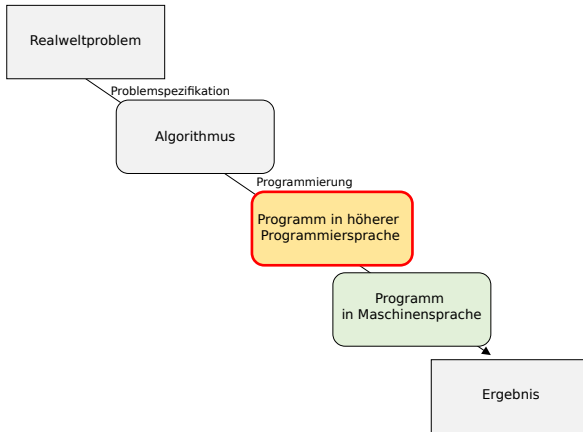
- Assemblersprachen ab 1950, erste Form der symbolischen Programmierung
- Bspw. "MOV AI, 61H" # Intel-Prozessor-spezifische Sprache

3. Generation (3GL)

- Höhere Programmiersprachen ab 1970 wie FORTRAN, C, C++, Java
- Programmierfreundlich, prozessor-unabhängig

4. Generation (4GL)

- Höhere Programmiersprachen ab 1980 wie Python, Matlab, R
- Codeoverhead Minimierung, Automation, Flexibilität, Multiparadigmatisch



Imperative Programmierung

- Problemlösungsweg wird als Folge von *Anweisungen (Befehlen)* vorgegeben.
- Befehle verarbeiten Daten, die mithilfe von *Variablen* adressiert werden.
 - **Prozedurale imperative Programmierung**
 - Daten und sie manipulierende Befehle werden separat behandelt.
 - Prozeduren (Funktionen) bilden das zentrale Strukturkonzept.
 - **Objektorientierte imperative Programmierung**
 - Daten und manipulierende Befehle werden als *Objekte* zusammengefasst.
 - Objekte bilden das zentrale Strukturkonzept.
- Praktisch liegen oft Mischformen vor.

Kompilierte Programmiersprachen

- Gesamter Quellcode wird *vor der Ausführung* in Maschinensprache übersetzt.
- Das Übersetzungsprogramm heißt *Compiler*.
- Der übersetzte Maschinencode wird vom Prozessor ausgeführt.
- Das ausführbare Programm wird nicht übersetzt und läuft schnell.
- Bei Änderungen des Quellcodes muss neu kompiliert werden.
- Beispiele für kompilierte Sprachen sind Java, C, C++.

Interpretierte Programmiersprachen

- Quellcode wird *während der Ausführung* in maschinennahe Sprache übersetzt.
- Das Ausführungsprogramm heißt *Interpreter*.
- Das Programm läuft aufgrund der Interpretation langsamer.
- Bei Änderungen des Quellcodes muss nicht neu interpretiert werden.
- Beispiele für interpretierte Sprachen sind Python und R.

Formalia

Grundbegriffe der Informatik

R und Visual Studio Code

Selbstkontrollfragen

Was ist R?

- Eine Programmiersprache und ein Softwarepaket.
- Entwickelt von Ihaka and Gentleman (1996).
- Freier Dialekt der proprietären Software S (Becker, Chambers, and Wilks (1988)).
- Weiterentwickelt und gepflegt durch [R Core Team](#) und [R Foundation](#)
- Interpretierte imperative 4GL Sprache.
- Optimiert und populär für statistische Datenanalysen.
- Große Community mit etwa 20.000 beigetragenen [R Paketen](#) (Erweiterungen)
- Evolviert und konservativ im Kern, konsistent und progressiv in R Paketen.

Wie bekomme ich R?

Über cran.r-project.org die geeignete Version herunterladen und installieren.



CRAW
Mimry
What's new?
Search
CRAN Team

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Link Views
Other

Documentation
Manuals
FAQs
Contributed

Download and Install R
Precompiled binary distributions of the base system and contributed packages, Windows and Mac users most likely want one of these versions of R: <ul style="list-style-type: none">• Download R for Linux (Debian, Fedora/Redhat, Ubuntu)• Download R for macOS• Download R for Windows
R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.
Source Code for all Platforms
Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it! <ul style="list-style-type: none">• The latest release (2022-06-23, Funny-Looking Kid) R-4.2.1.tar.gz, read what's new in the latest version.• Sources of R.alpha and beta releases (daily snapshots, created only in time periods before a planned release).• Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.• Source code of older versions of R is available here.• Contributed extension packages
Questions About R
<ul style="list-style-type: none">• If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

What are R and CRAN?

R is 'GNU S', a freely available language and environment for statistical computing and graphics which provides a wide variety of statistical and graphical techniques: linear and nonlinear modelling, statistical tests, time series analysis, classification, clustering, etc. Please consult the [R project homepage](#) for further information.

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. Please use the CRAN [mirror](#) nearest to you to minimize network load.

Submitting to CRAN

To "submit" a package to CRAN, check that your submission meets the [CRAN Repository Policy](#) and then use the [web form](#).

If this fails, send an email to CRAN-submissions@R-project.org following the policy. Please do not attach submissions to emails, because this will clutter up the mailboxes of half a dozen people.

Was kann man mit R machen?

- Datensätze laden, manipulieren, und speichern.
- Eine Vielzahl von Berechnungen an verschiedenen Datenstrukturen durchführen.
- Eine Vielzahl statistischer Analysemethoden auf Daten anwenden.
- Datenanalyseskripte schreiben und Abbildungen generieren.
- Präsentationen [RMarkdown](#) und Bücher [RBookdown](#) erstellen.
- Wissenschaftliche Berichte mit [Quarto](#) erstellen.

Was kann man mit R (bisher) nicht so gut machen?

- In einer ansprechenden Umgebung programmieren (\Rightarrow Visual Studio Code).
- Scientific Computing (\Rightarrow Python, Matlab, Julia).
- Psychologische Experimente programmieren (\Rightarrow Python, Matlab)

Wie bekomme ich Hilfe zu R?

- Googlen
- ChatGPT
- stackoverflow.com
- r-project.org/help.html
- Während der Programmierung und bei bekanntem Funktionsnamen über die Kommandozeile:

```
?mean          # Zeigt Hilfe zu der Funktion "mean()"  
help(mean)     # Zeigt Hilfe zu der Funktion "mean()"  
browseVignettes() # Zeigt Vignetten aller installierten Pakete im Browser  
browseVignettes("knitr") # Zeigt Vignetten des Paktes "knitr"
```

- rseek.org
- rstudio.com/resources/cheatsheets
- r-bloggers.com

Was ist Visual Studio Code (VSCode)?

- VSCode ist ein kostenloser Quelltext-Editor von Microsoft.
- VSCode ist eine Softwareentwicklungsumgebung (Integrated Development Environment, IDE)
- Seit 2015 für Windows, macOS und Linux verfügbar.
- Seit 2018 ist VSCode der beliebteste Editor laut jährlicher Stackoverflow Umfragen.
- Ein Microsoftprodukt ist damit auch der beliebteste Editor der Linuxwelt.
- Über Extensions kann VSCode als IDE für beliebige Sprachen genutzt werden.
- Zum Beispiel funktioniert VSCode als IDE für R, Python, Julia, Shell, Quarto, etc.
- VSCode ist Community-based und sehr konfigurierbar.
- VSCode ist über Microsoft's GitHub über Endgeräte synchronisierbar.

Wie bekomme ich VSCode?

Über code.visualstudio.com heruntergeladen und installiert.

Visual Studio Code Docs Updates Blog API Extensions FAQ Learn

Search Docs Download

Version 1.83 is now available! Read about the new features and fixes from September.

Code editing. Redefined.

Free. Built on open source. Runs everywhere.

[.deb](#) Debian, Ubuntu... [.rpm](#) Red Hat, Fedora...

Web, insiders editor, or other platforms

By using VS Code, you agree to its license and privacy statement.

IntelliSense Run and Debug Built-in Git Extensions

Wie benutze ich VSCode?

Online Dokumentation: code.visualstudio.com/docs

The screenshot shows the Visual Studio Code documentation website. At the top, there is a navigation bar with the Visual Studio Code logo, links for Docs, Updates, Blog, API, Extensions, FAQ, and Learn, a search bar, and a Download button. Below the navigation bar, a banner for 'Version 1.82' is visible. The main content area is titled 'Getting Started' and contains a paragraph describing Visual Studio Code as a lightweight but powerful source code editor. To the left of the main content is a sidebar with a table of contents including Overview, Setup, Get Started, User Guide, Source Control, Terminal, Languages, Node.js / JavaScript, TypeScript, Python, Java, C++, C#, Docker, Data Science, Azure, Remote, and Dev Containers. To the right of the main content is another sidebar titled 'Getting Started' with links for VS Code in Action, Top Extensions, First Steps, Keyboard Shortcuts, Downloads, Privacy, Subscribe, Ask questions, Follow @code, Request features, Report issues, and Watch videos. The main content area features a code editor snippet showing JavaScript code for an Express.js server with a dropdown menu for 'Intelligent Code Completion' showing various methods like get, getMaxListeners, arguments, engine, length, merge, purge, settings, toString, and defaultConfiguration. Below the code editor is a section titled 'Intelligent Code Completion' with the text 'Code smarter with IntelliSense - completions for variables, methods, and imported modules.' and a progress indicator.

Visual Studio Code Docs Updates Blog API Extensions FAQ Learn

Search Docs Download

Version 1.82 is now available! Read about the new features and fixes from September.

Getting Started

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as C++, C#, Java, Python, PHP, Go, .NET). Begin your journey with VS Code with these [introductory videos](#).

Visual Studio Code in Action

```
4 var server = express();
5 server.use(bodyParser.json);
6
7 server.get
8
9
10
11
12
13
14
15
16
17
18
```

Intelligent Code Completion

Code smarter with IntelliSense - completions for variables, methods, and imported modules.

GETTING STARTED

- VS Code in Action
- Top Extensions
- First Steps
- Keyboard Shortcuts
- Downloads
- Privacy
- Subscribe
- Ask questions
- Follow @code
- Request features
- Report issues
- Watch videos

1. R installieren.
2. VSCode installieren.
3. VSCode für R startklar machen (Anleitung [hier](#)).

Formalia

Grundbegriffe der Informatik

R und Visual Studio Code

Selbstkontrollfragen

Selbstkontrollfragen

1. Geben Sie die typische Struktur einer computergestützten Datenanalyse wieder.
2. Erläutern Sie den Begriff "Datenanalyseskript".
3. Definieren Sie den Begriff "Informatik".
4. Erläutern Sie die Akronyme CPU, RAM, SSD, und GPU.
5. Nennen Sie wesentliche Aspekte der Von-Neumann Rechnerarchitektur.
6. Definieren Sie den Begriff Algorithmus.
7. Erläutern Sie den Zusammenhang von Algorithmen und Programmen.
8. Was bezeichnen die Syntax und Semantik einer Programmiersprache?
9. Differenzieren Sie die Begriffe "Maschinensprache" und "höhere Programmiersprache".
10. Skizzieren Sie Prinzipien der prozeduralen und objektorientierten imperativen Programmierung.
11. Skizzieren Sie die Entwicklung der Programmiersprachen der ersten bis vierten Generation.
12. Differenzieren Sie die Begriffe der kompilierten und der interpretierten Programmiersprachen.

- Becker, Richard A., John M. Chambers, and Allen Reeve Wilks. 1988. *The New S Language: A Programming Environment for Data Analysis and Graphics*. Reprint. London: Chapman & Hall.
- Ihaka, Ross, and Robert Gentleman. 1996. "R: A Language for Data Analysis and Graphics." *Journal of Computational and Graphical Statistics* 5 (3): 2999–2314.