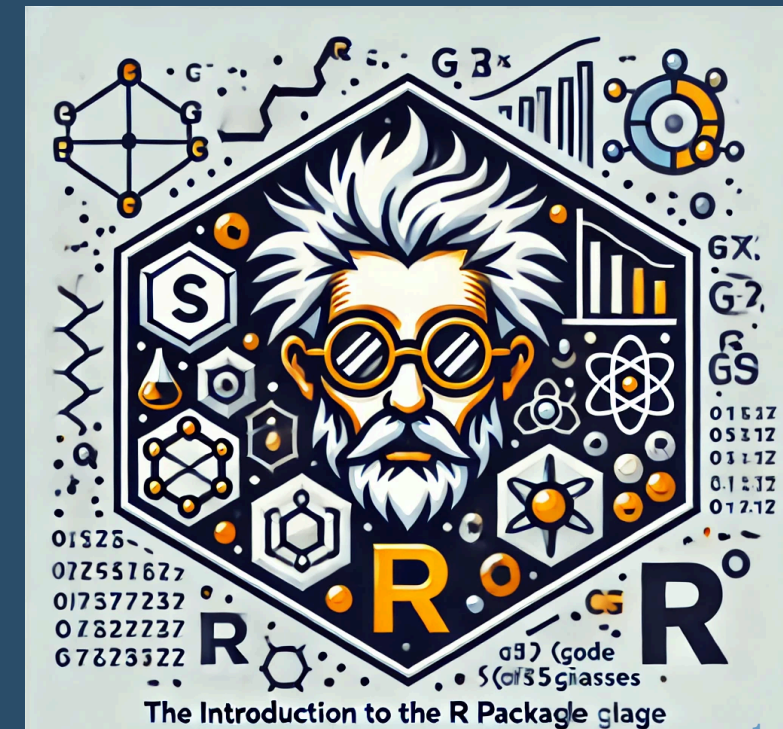


Teil II – Pakete in R

Grundlagen der Datenanalyse und Statistik mit R | WS 2024/25


Prof. Dr. Daniel Schnitzlein



Was sind R-Pakete?

- R besteht aus einer Basiskomponente, die durch sogenannte R-Pakete erweitert werden kann um bestimmte Aufgaben zu lösen.
- Das wichtigste Repository für R-Pakete, das Comprehensive R Archive Network (CRAN), listet zur Zeit 21.517 (Oktober 2024) verfügbare Pakete auf (zuzüglich einer Unmenge von Paketen auf Github).
- Für alle (die meisten) verfügbaren Pakete ist eine umfangreiche Dokumentation verfügbar.
- Warum?
- Möglichkeiten den Überblick zu behalten: CRAN Task Views (siehe nächste Seite), Social Media Posts, R-User Konferenzen (meist auch via YouTube verfügbar) oder Diskussionen in Foren und/oder GitHub.

CRAN Task Views



CRAN

[Mirrors](#)

[What's new?](#)

[Search](#)

[CRAN Team](#)

About R

[R Homepage](#)

[The R Journal](#)

Software

[R Sources](#)

[R Binaries](#)

[Packages](#)

[Task Views](#)

[Other](#)

Documentation

[Manuals](#)

[FAQs](#)

[Contributed](#)

Donations

[Donate](#)

CRAN Task Views

CRAN task views aim to provide guidance which packages on CRAN are relevant for tasks related to a certain topic. They give a brief overview of the included packages which can also be automatically installed using the [ctv](#) package. The views are intended to have a sharp focus so that it is sufficiently clear which packages should be included (or excluded) - and they are *not* meant to endorse the "best" packages for a given task.

To automatically install the views, the [ctv](#) package needs to be installed, e.g., via

```
install.packages("ctv")
```

and then the views can be installed via `install.views` or `update.views` (where the latter only installs those packages are not installed and up-to-date), e.g.,

```
ctv::install.views("Econometrics")
ctv::update.views("Econometrics")
```

To query information about a particular task view on CRAN from within R or to obtain the list of all task views available, respectively, the following commands are provided:

```
ctv::ctv("Econometrics")
ctv::available.views()
```

The resources available from the [CRAN Task View Initiative](#) provide further information on how to contribute to existing task views and how to propose new task views.

Topics

ActuarialScience	Actuarial Science
Agriculture	Agricultural Science
Bayesian	Bayesian Inference
CausalInference	Causal Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
Databases	Databases with R
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
DynamicVisualizations	Dynamic Visualizations and Interactive Graphics
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
Epidemiology	Epidemiology
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
ExtremeValue	Extreme Value Analysis
Finance	Empirical Finance
FunctionalData	Functional Data Analysis
GraphicalModels	Graphical Models
HighPerformanceComputing	High-Performance and Parallel Computing with R
Hydrology	Hydrological Data and Modeling
MachineLearning	Machine Learning & Statistical Learning
MedicalImaging	Medical Image Analysis
MetaAnalysis	Meta-Analysis

Was bedeuten Pakete für R?

- Pakete erweitern den Funktionsumfang von R erheblich. Pakete werden von der Community und/oder Unternehmen erstellt und gepflegt.
- **Vorteil**: Durch die aktive und breite Community stellt R auch für hochspezialisierte Fragestellungen Lösungen bereit, die proprietäre Software u.U. nicht enthalten würde.
- **Vorteil**: Deutlich höhere Entwicklungsgeschwindigkeit neuer Methoden.
- **Nachteil**: Nicht alle Pakete sind zwingend miteinander kompatibel. Fallen die Akteure, die ein Paket betreuen aus, kann es sein, dass die Funktionalität verloren geht, wenn sich R weiterentwickelt.
- **Nachteil**: Ebenso umfasst Versionierung nicht nur R sondern auch alle Pakete.
- **Realität**: Viele populäre Pakete werden in Gruppen zusammengefasst entwickelt und gepflegt. Oft sind zentrale Figuren entweder bei Unternehmen angestellt, die einen Bezug zu R haben, oder die R extensiv nutzen.

Es gibt Pakete die R erweitern, z.B. die **easystats** Paket-Gruppe und ...



... es gibt Pakete die auch die Art und Weise verändern wie mit R gearbeitet wird, z.B. die **tidyverse** Pakete.



Die Rolle von Posit (früher RStudio)

- Die Entwicklung von R zu der Verbreitung die R heute hat, wäre ohne die grafische IDE RStudio die 2011 von der damals gleichnamigen Firma auf den Markt gebracht wurde nicht denkbar gewesen.
- RStudio hat aber auch massiv die Entwicklung von R von einem “Spielplatz für Akademiker” hin zu einer zuverlässigen Grundlage für Produktivsysteme vorangetrieben.
- RStudio verdient dabei ihr Geld mit der Bereitstellung von R Entwicklungsumgebungen (u.a. RStudio) für Unternehmen.
- Inzwischen hat sich RStudio in Posit umbenannt und hat neben dem Fokus auf R auch Python im Blick.
- Viele hoch innovative Pakete in R und Python werden direkt oder indirekt von Posit entwickelt.
- Dadurch ergibt sich natürlich eine gewisse Macht aber auch eine gewisse Notwendigkeit Standards zu setzen (siehe Tidyverse).
- Nicht unbedingt in der ganzen R Community beliebt.

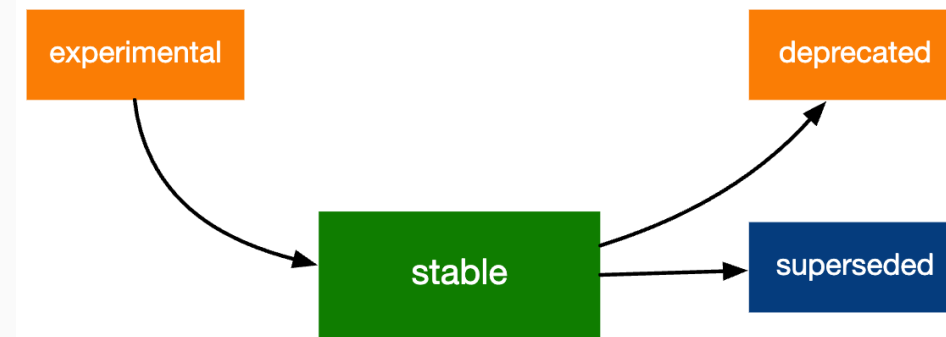
Beispiel: Lebenszyklus für Funktionen

- Ein weit verbreitetes Problem mit Funktionen in R Paketen war lange Zeit (und ist es stellenweise immer noch), dass diese teilweise sehr kurzfristig durch neue Versionen ersetzt werden.
- Insbesondere in Produktivumgebungen ist das sehr ärgerlich, wenn z.B. Dashboards nach Updates neue angepasst werden müssen.
- Idealerweise kündigen die jeweiligen Administratoren der Pakete solche Änderungen an. Jedoch ist das nicht immer so.
- Posit/RStudio hat um diesem Problem zu begehen einen Lebenszyklus für Funktionen/Pakete etc. eingeführt.

Lifecycle stages

Source: [vignettes/stages.Rmd](#)

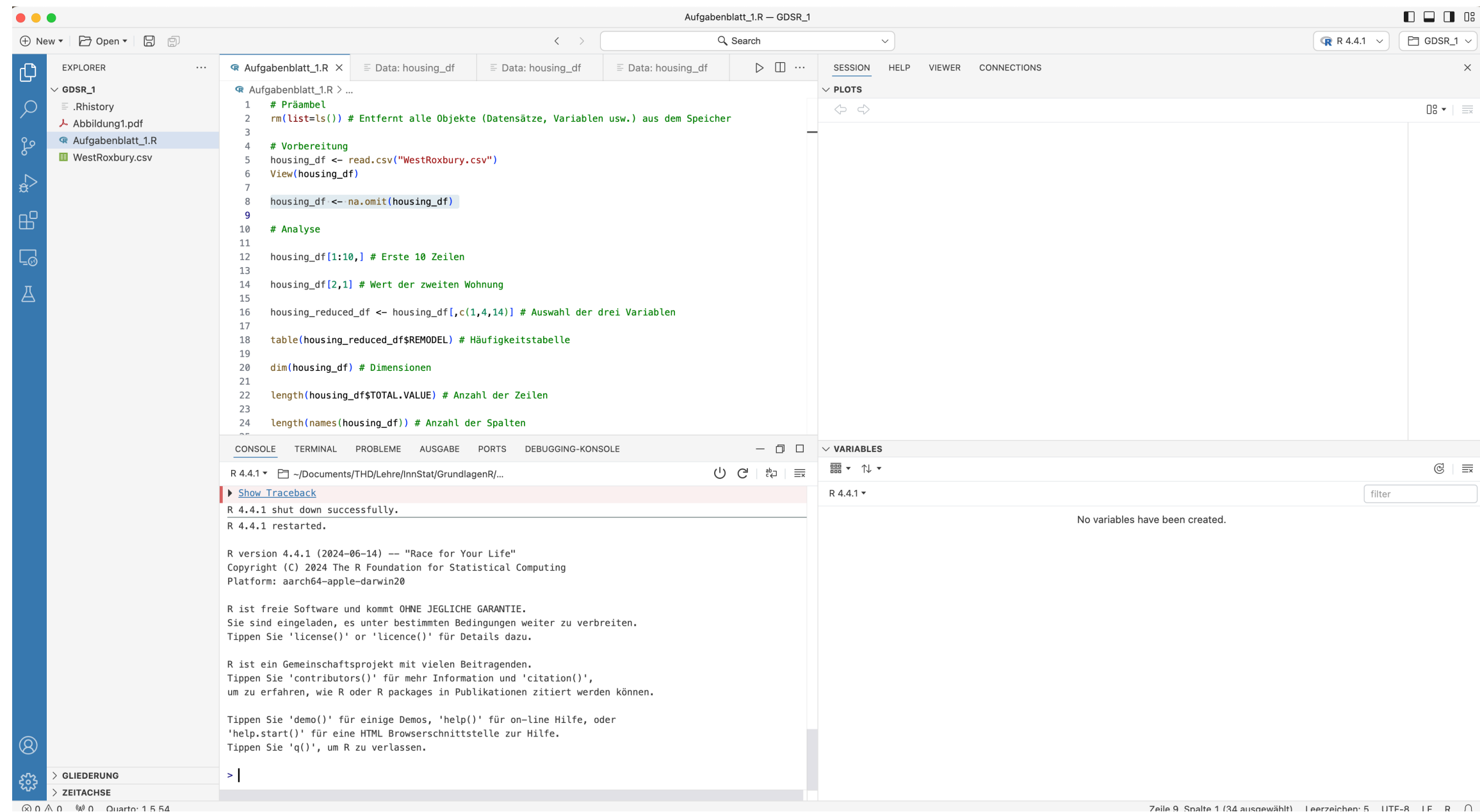
This vignette describes the four primary stages of the tidyverse lifecycle: stable, deprecated, superseded, and experimental.



A diagram showing the transitions between the four main stages: experimental can become stable and stable can become deprecated or superseded.

The lifecycle stages can apply to packages, functions, function arguments, and even specific values of a function argument. However, you'll mostly see them used to label individual functions, so that's the language we use below.

Beispiel: Positron



Beispiel: Tidyverse (1)



At a high level, the tidyverse is a language for solving data science challenges with R code. Its primary goal is to facilitate a conversation between a human and a computer about data.

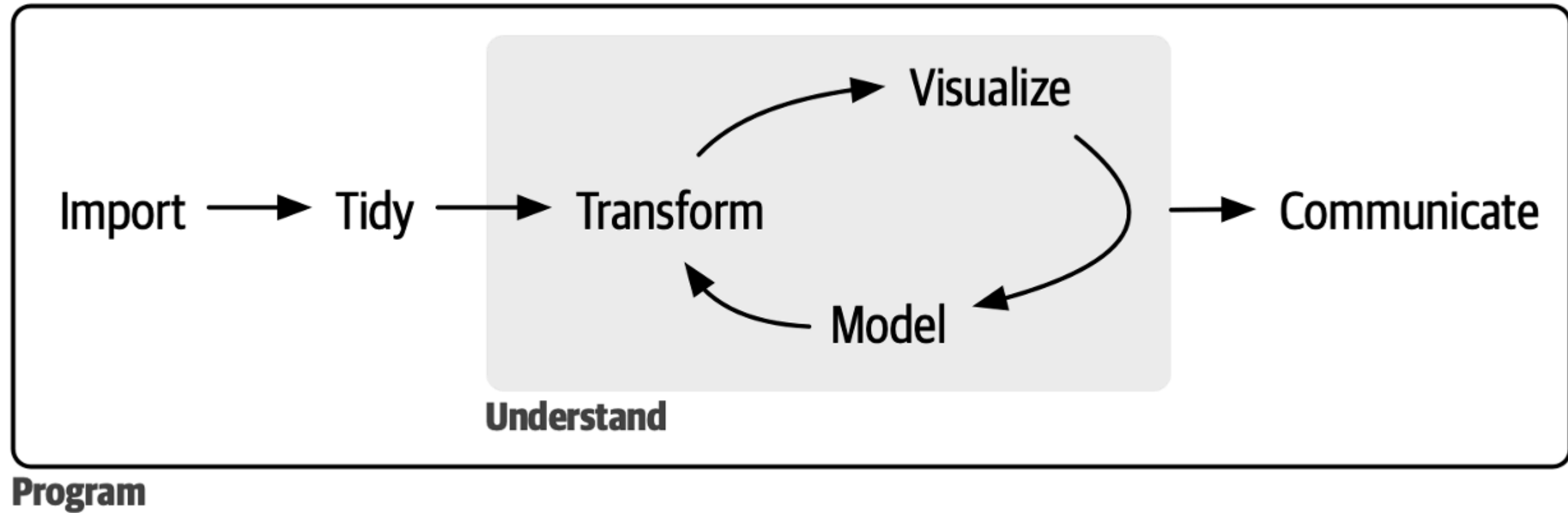
Less abstractly, the tidyverse is a collection of R packages that share a high-level design philosophy and low-level grammar and data structures, so that learning one package makes it easier to learn the next.

(Wickham et al. 2019)

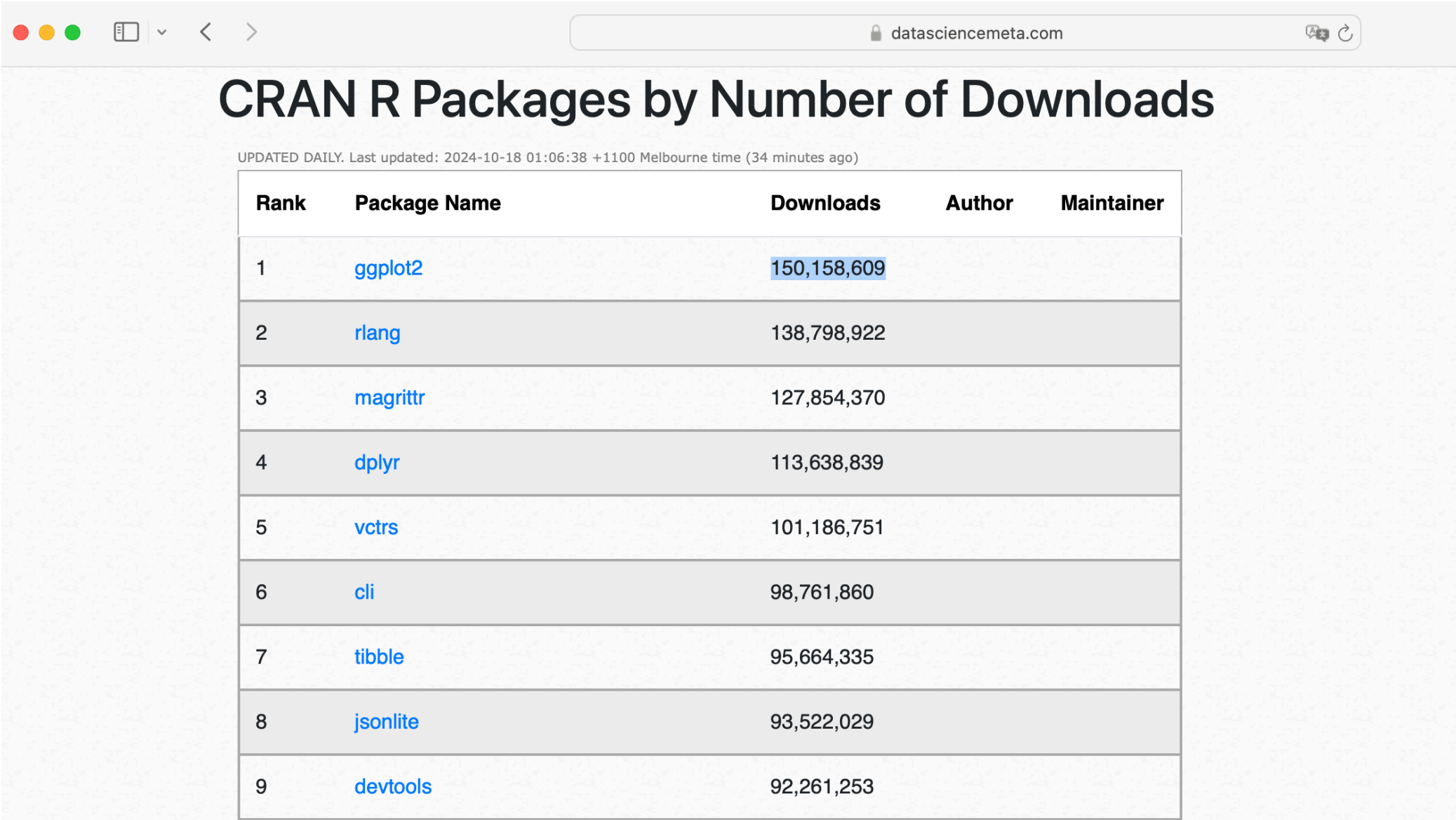
Beispiel: Tidyverse (2)

- Das Ziel ist, wenn Daten in der Form vorliegen, wie das Tidyverse die Daten gerne hätte (sog. tidy Datenformat), dann sollten alle Funktionen aller Tidyverse-Pakete und der sich auf das Tidyverse berufenden Pakete out-of-the-box funktionieren.
- Alle Funktionen des Tidyverse sollen einer gemeinsamen Logik im Funktionsaufbau folgen.
- Die Tidyverse Pakete orientieren sich dabei an einem genormten Ablauf einer Datenanalyse (siehe nächste Folie).
- Dieser hohe Standardisierungsgrad hat dazu beigetragen z.B. ggplot2 zum beliebtesten R-Paket zumachen mit aktuell (Oktober 2024) 150 Mio. Downloads (über CRAN).
- Posit wendet dabei die Standards sowohl auf R als auch auf Python an. Mittelfristig ist das Ziel einen nahtlosen Austausch zwischen Python und R Paketen zu ermöglichen.

Beispiel: Tidyverse (3)



Die wichtigsten Pakete nach Downloads



Rank	Package Name	Downloads	Author	Maintainer
1	ggplot2	150,158,609		
2	rlang	138,798,922		
3	magrittr	127,854,370		
4	dplyr	113,638,839		
5	vctrs	101,186,751		
6	cli	98,761,860		
7	tibble	95,664,335		
8	jsonlite	93,522,029		
9	devtools	92,261,253		