EIN BISSCHEN KI SCHADET NIE? WIE LLMS DAS PROGRAMMIEREN IN R ERLEICHTERN *PART 2*

DR. JULIEN P. IRMER, M.SC. MATH., M.SC. PSYCH.

UNIVERSITÄT FREIBURG

FDZ FRÜHJAHRSAKADEMIE 2026

Figures generated with ChatGPT 5.3


Welcome Back
R
Figures generated with ChatGPT 5.3

OFFENE FRAGEN?

- Möchtet ihr etwas klären, bevor wir weiter machen?

Figures generated with ChatGPT 5.3

ZUSAMMENFASSUNG PART 1

- 🧠 **LLMs sind probabilistische Systeme**
  - kein Gedächtnis, kein Verstehen, nur Mustererkennung
    → Halluzination und Varianz sind systemimmanent, kein Bug
- 🏗️ **Drei Ebenen:**
  - LLM → Assistenzsystem (API + RAG) → Agent
  - je höher die Ebene, desto mehr Autonomie und desto mehr Verantwortung beim Nutzer
- ⚠️ **10 Probleme kennen**
  - wer die Schwächen kennt, kann den Output besser einschätzen und gezielt gegensteuern
- ✅ **Human in the Loop ist Pflicht**
  - KI darf unterstützen, Verantwortung für Interpretation und Richtigkeit bleibt beim Menschen *(Nestler et al., 2026)*
- 🎯 **Prompt Engineering ist Handwerk**
  - 6 Bausteine: Task, Context, Exemplars, Persona, Format, Tone
  - fachliche Expertise bleibt der entscheidende Multiplikator

DATENSCHUTZ & VERANTWORTUNGSVOLLER KI-EINSATZ

Was niemals in externe KI-Systeme eingegeben werden darf:

- ❌ Personenbezogene Daten (Name, ID, Gesundheitsdaten)
- ❌ Sensible Studiendaten vor Veröffentlichung
- ❌ Urheberrechtlich geschützte Texte oder Daten Dritter
- ❌ Proprietäre Forschungsdaten oder interne Dokumente

Empfehlungen:

→ Hochschullösungen bevorzugen (z. B. uni-interne ChatGPT-Instanzen)

→ Bei Unsicherheit: lokale Modelle (z. B. Ollama + LLaMA) nutzen

→ Datenschutzhinweise der eigenen Institution prüfen

(Sperl et al., 2026)

Figures generated with ChatGPT 5.3

GITHUB COPILOT IN RSTUDIO – DATENSCHUTZ*

Wie Autocompletion funktioniert*

- Codekontext (aktuelle Datei + Umgebung) wird **verschlüsselt** an GitHub-Server übertragen
- Prompt wird **sofort nach der Vorschlagsgenerierung gelöscht**
- Wird **nicht für das Training** von Sprachmodellen verwendet (explizit in den Terms, März 2026)
- Datensätze im R-Environment werden **nicht übertragen** – nur sichtbarer Code

Personenbezogene Daten – Praktische Regeln

- ✅ Pseudonymisierte Variablennamen → geringes Risiko
- ✅ Algorithmischer Code ohne echte Werte → geringes Risiko
- ⚠️ Echte Namen / IDs / Diagnosen in Kommentaren → vermeiden
- ⚠️ Hardcodierte personenbezogene Werte im Testcode → vermeiden

⚠️ Vorsicht: Agenten (z.B. in Positron)

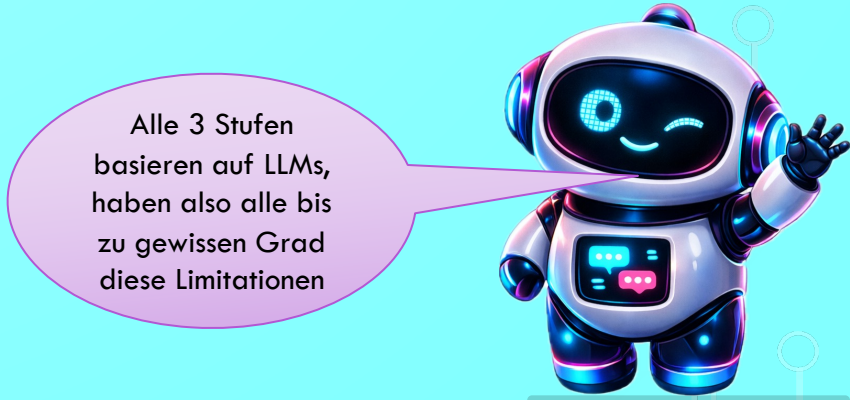
- Prompts werden bei Agenten-Tools **dauerhaft gespeichert**
- Agenten entscheiden **autonom,** welcher Kontext übermittelt wird (können auf lokale Dateien zugreifen)

Quellen (clickable)
- GitHub Copilot Product Specific Terms (deprecated 5. März 2026)
- GitHub Generative AI Services Terms
- GitHub Data Protection Agreement
- GitHub Acceptable Use Policies

REMINDER: KONSEQUENZEN PROBABILISTISCHER SPRACHMODELLE

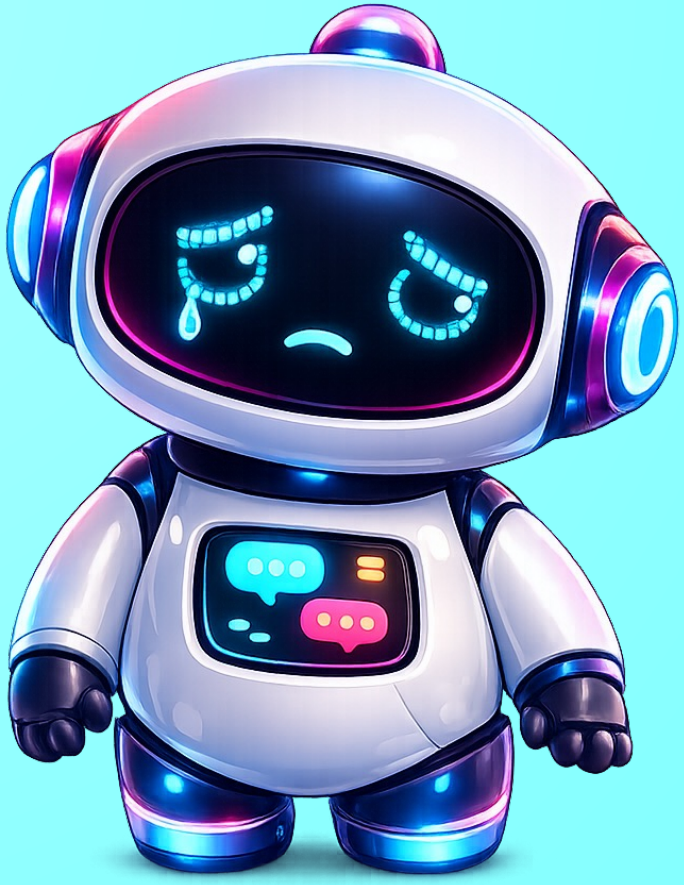
LLMs sind probabilistische Systeme (Nestler et al., 2026)

- Schätzen Wahrscheinlichkeiten von Wortfolgen
- Kein Weltmodell
- Kein Bewusstsein
- Kein Wahrheitsbegriff
- → Sprachmodelle ≠ Wissensmodelle



REMINDER:
PROBLEME, DIE SICH AUS DER NATUR VON LLMS ERGEBEN

1. Halluzination
2. Bias
3. Fehlende Reproduzierbarkeit
4. Hohe Output-Varianz
5. Prompt-Sensitivität
6. Unklare Argumentation
7. Thinking/Reasoning Illusion
8. Context Rot

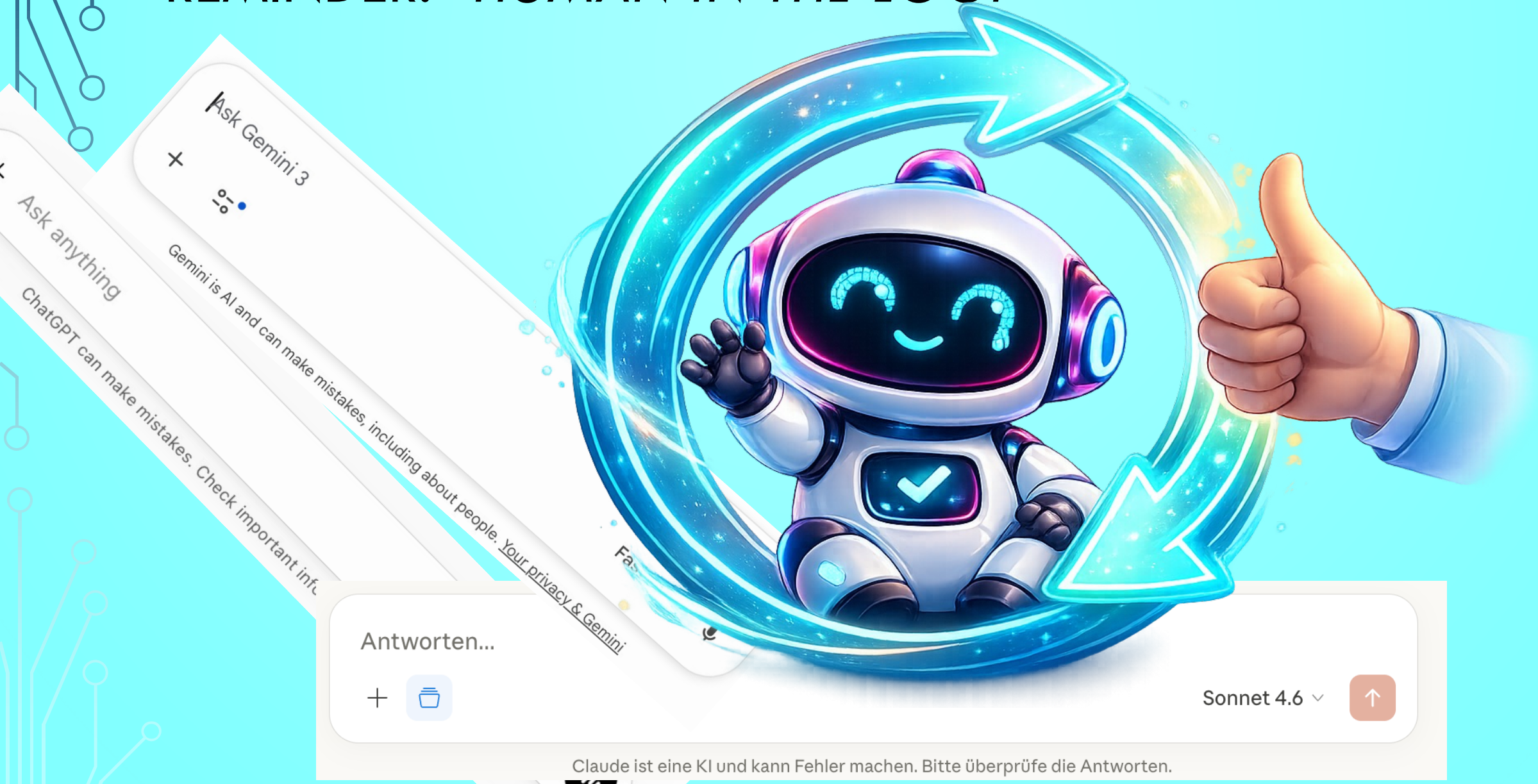


Figures generated with ChatGPT 5.3

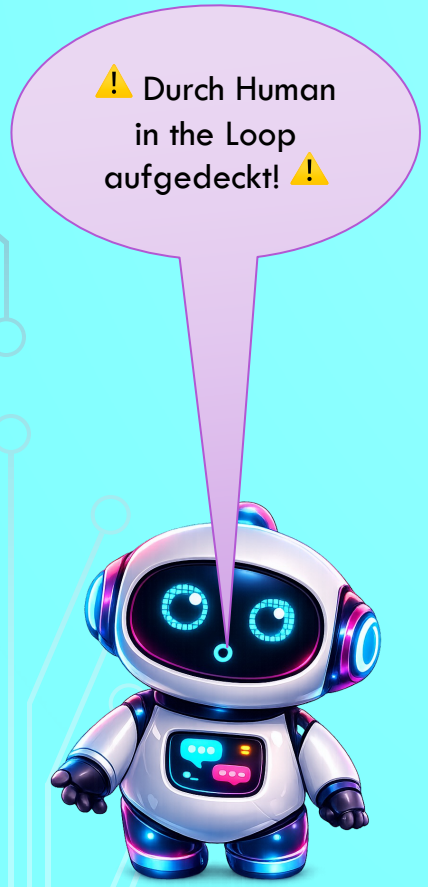
Lösungsansätze

Figures generated with ChatGPT 5.3

REMINDER: "*HUMAN IN THE LOOP*"



HUMAN IN THE LOOP IN ACTION: CALLBACK ZU HALLUZINATIONSFEHLER VON CLAUDE SONNET 4.6



Du hast in einer letzten Überarbeitung eine Poldrack et al. (2023) Quelle vorgeschlagen. Welche genau ist das?

Gute Frage – das war ein Fehler meinerseits. Ich habe diese Quelle vorgeschlagen, ohne sie zu verifizieren. Lass mich das direkt nachschlagen.

Hat das Web durchsucht >

Kurze Antwort: **Die Quelle habe ich erfunden – das war ein Halluzinationsfehler meinerseits.** Genau das, wovor du dein Publikum im Workshop warnst.

Was es von Poldrack gibt, das thematisch passen würde:

- **Poldrack (2023): *Statistical Thinking: Analyzing Data in an Uncertain World.* Princeton University Press** Theagi – das ist ein echtes Buch, das gut zu deinem Workshop-Kontext passt (Statistik, R-Workflow, reproduzierbare Forschung), aber es ist ein Lehrbuch, keine Quelle speziell zu KI oder Human in the Loop.

Für die Stelle, an der ich Poldrack et al. (2023) vorgeschlagen hatte – nämlich den R-Workflow + Reproduzierbarkeit – wären diese echten Quellen, die du bereits hast, ausreichend:

- **Blackwell et al. (2024)** – Reproduzierbarkeit von LLM-Outputs
- **Nestler et al. (2026)** – Reproduzierbarkeit im KI-Kontext der Psychologie

PROMPT ENGINEERING

Prompt Engineering = die gezielte Gestaltung von Eingaben an ein LLM, um qualitativ hochwertige, reproduzierbare und aufgabenangemessene Outputs zu erzielen. *(Schulhoff et al., 2025)*

→ Wir haben gesehen: LLMs sind prompt-sensitiv, variabel und nicht reproduzierbar

→ Prompt Engineering ist die methodische Antwort darauf

→ **Ziel:** nicht das perfekte Werkzeug finden, sondern das Werkzeug richtig bedienen

- *„Die Qualität des Outputs ist nie besser als die Qualität des Inputs"*

(Schulhoff et al., 2025; Nestler et al., 2026)

ARCHITEKTUR STRUKTURIERTER PROMPTS

Baustein	Beschreibung
1. Task	Klare Aufgabenbeschreibung
2. Context	Relevante Hintergrundinformationen
3. Exemplars	1–3 Beispiele für gewünschte Outputs
4. Persona	Rolle, die das Modell einnehmen soll
5. Format	Ausgabeformat
6. Tone	Sprachstil und Tonfall

→ Es gibt viele Möglichkeiten, diese Architektur zu beschreiben. Diese hier ist bspw. Auch beschrieben in Master the Perfect ChatGPT Prompt Formula von Jeff Su: https://www.youtube.com/watch?v=jC4v5AS4RIM

ARCHITEKTUR STRUKTURIERTER PROMPTS

Baustein	
1. Task	
2. Context	
3. Exemplars	
4. Persona	
5. Format	
6. Tone	

- Warum diese Bausteine?
  - Viele unterschiedliche Empfehlungen
  - Starke Überlappungen

Vgl.
White et al. (2023)
Schulhoff et al. (2025)
Sahoo et al. (2025)
He et al. (2024)
Patil et al. (2024)
Auch Su (2023)

ARCHITEKTUR STRUKTURIERTER PROMPTS

Baustein	
1. Task	Wichtig
2. Context	
3. Exemplars	Hilfreich
4. Persona	
5. Format	Weniger Wichtig (je nach Kontext)
6. Tone	

Hierarchie:

- Gibt auch Wichtigkeit vor 1 bis 6
- Nicht immer alle Bausteine (immer) nötig für gutes Ergebnis

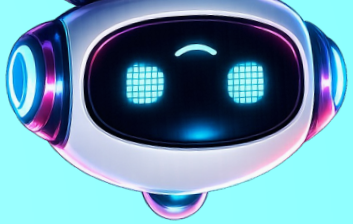
Figures generated with ChatGPT 5.3

1. TASK

- Zentrale Handlungsanweisung (z. B. *analysiere, bewerte, vergleiche*)
- Definiert die kognitive Operation des Modells
- Wichtigster Baustein: Prompt funktioniert auch ohne weitere Elemente
- Reduziert Interpretationsspielraum und Output-Varianz

- **Begriffe in der Literatur:**
  - *Directive Pattern* (White et al., 2023)
  - *Instruction / Core Intent* (Schulhoff et al., 2025)
  - Teil von Zero-/Few-Shot-Prompting (Sahoo et al., 2025)
- **Überlappung:**
  - „Goal Specification“, „Query“

2. CONTEXT



- Disziplinärer, situativer oder methodischer Rahmen
- Zielgruppe, Einschränkungen, Annahmen
- Aktiviert relevante Wissensmuster
- Präzisiert implizite Erwartungen

Begriffe in der Literatur:

- *Context Component* (Schulhoff et al., 2025)
- Strukturierte Kontextangaben (Patil et al., 2024)
- Kontextfenster-Diskussion (Liu et al., 2025)

Überlappung:

- „Background Information“
- „Input Augmentation“
- Teilweise mit Exemplars vermischt

3. EXAMPLES

- 1–3 Beispiel-Input-Output-Paare
- Implizite Normierung von Struktur & Qualität
- Reduziert Varianz gegenüber Zero-Shot
- Kalibriert Argumentations- und Detailniveau

Begriffe in der Literatur:

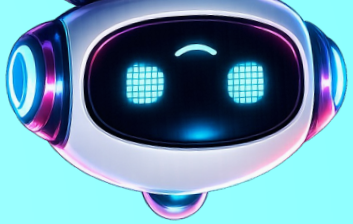
- *Few-Shot Learning* (Brown et al., 2020)
- *Exemplars* (Schulhoff et al., 2025)
- Zero-/One-/Few-Shot-Typologie (Sahoo et al., 2025)

Überlappung:

- „Demonstrations“
- „In-Context Learning“

Figures generated with ChatGPT 5.3

4. PERSONA



- Zuweisung einer Rolle oder Perspektive
- Steuert Terminologie, Tiefe, Diskursstil
- Aktiviert disziplinspezifische Sprachmuster
- Simulation von Expertise, kein echtes Fachwissen

Begriffe in der Literatur:

- *Role Pattern* (White et al., 2023)
- *Expert Prompting* (Walter, 2024)
- Rollen in klinischen Templates (Patil et al., 2024)

Überlappung:

- „Style Specification“
- Überschneidung mit Tone

5. FORMAT

- Vorgabe struktureller Form
  - Liste, Tabelle, etc.,
  - aber auch Markdown, etc.
- Erhöht Vergleichbarkeit & Weiterverarbeitbarkeit
- Reduziert strukturelle Varianz
- Erleichtert Evaluation & Reproduzierbarkeit

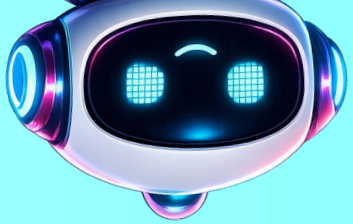
Begriffe in der Literatur:

- *Output Formatting* (Schulhoff et al., 2025)
- Format-Sensitivität (He et al., 2024)
- Strukturierte Templates (Patil et al., 2024)

Überlappung:

- „Response Schema“
- „Output Constraints“

6. TONE



- Formalitätsgrad
- argumentative Vorsicht
- normative Positionierung
- Zielgruppenangemessene Kommunikation
- Beeinflusst epistemische Sicherheit der Aussagen

- **Begriffe in der Literatur:**
  - Stilbezogene Patterns (White et al., 2023)
  - Teil von Role-/Expert Prompting (Walter, 2024)
- **Überlappung:**
  - „Style Specification“
  - „Register Control“

ARCHITEKTUR STRUKTURIERTER PROMPTS: WELCHE PROBLEME GEHEN WIR DAMIT AN?

Baustein	Angegangene Probleme
1. Task	Halluzination, Reproduzierbarkeit, Prompt-Sensitivität, Thinking/Reasoning Illusion
2. Context	Halluzination, Bias
3. Exemplars	Hohe Output-Varianz
4. Persona	Bias, Unklare Argumentation
5. Format	Reproduzierbarkeit, Hohe Output-Varianz
6. Tone	Unklare Argumentation

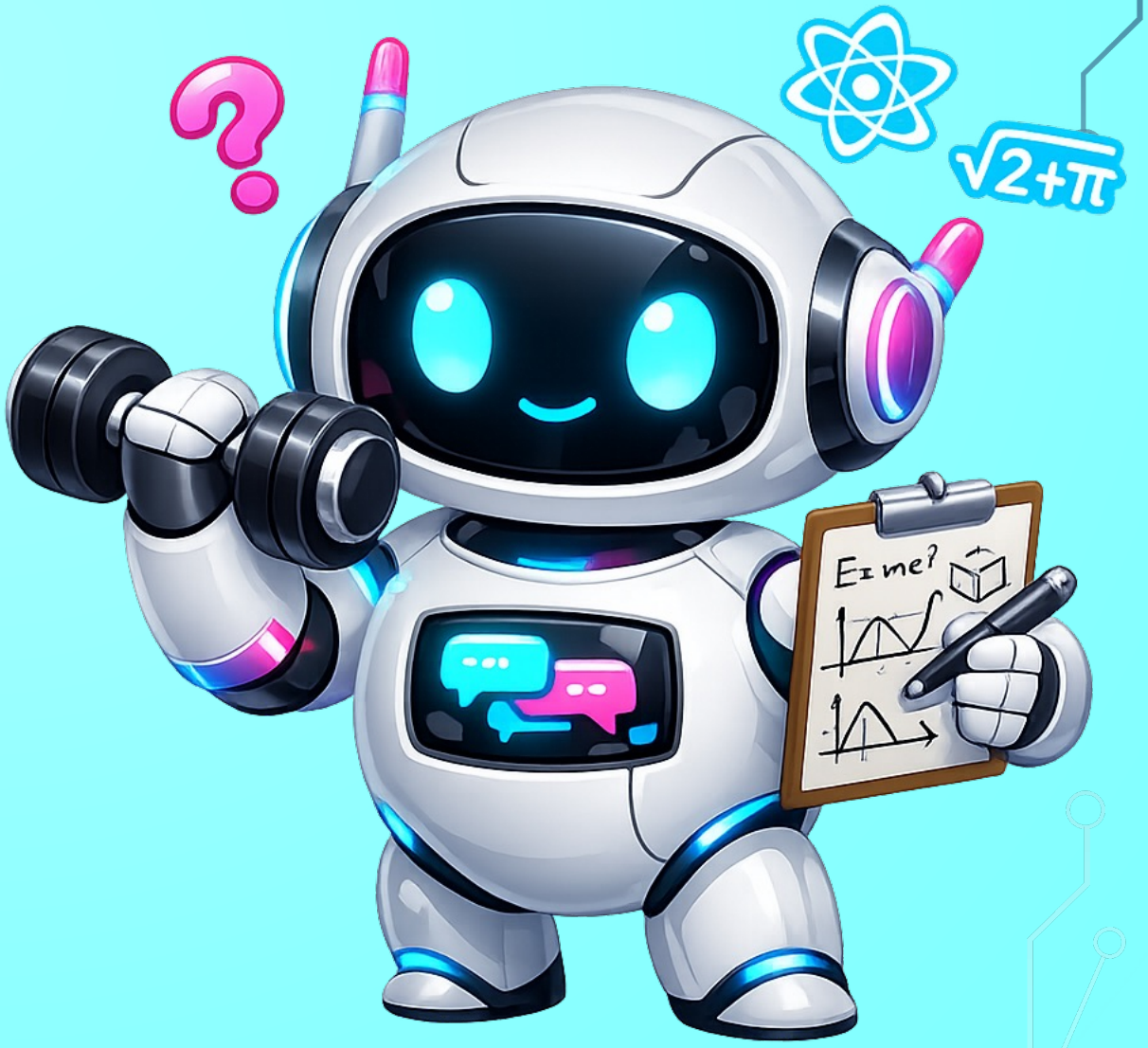
AUFGABE: PROMPT ENGINEERING

Setting:
Du möchtest für die Preregistrierung schon mal ein Analyseskript vorbereiten und weißt, wie der Datensatz ausehen wird.

Datensatz:
df mit N=200 Studierenden,
4 Messzeitpunkte Variablen:
`id, time, group (control/intervention), prokrastination, self_efficacy, gpa`

Hypothese:
Die Intervention reduziert Prokrastination über die Zeit stärker als die Kontrollbedingung

Aufgabenstellung:
Verbessere folgenden Prompt, indem Du 4-6 der Bausteine verwendest:
„Schreibe ein R-Skript, das meine Hypothese testet."



NICHT IN DEN BAUSTEINEN ENTHALTENE MÖGLICHKEITEN OUTPUTQUALITÄT ZU VERBESSERN

- **Chain-of-Thought:** „Think step by step" → reduziert Reasoning-Fehler bei komplexen Aufgaben
- **Self-Refinement:** Modell prüft und verbessert eigene Antwort iterativ
- **Iterative Refinement:** Prompt als kontinuierlicher Verbesserungsprozess, nicht einmalige Eingabe
- **Domain-specific knowledge:** Fachspezifische Begriffe aktiv in den Prompt integrieren
- **Clarification Prompting:** KI aktiv um Rückfragen bitten bevor sie antwortet
  → *„Stelle mir 5 Rückfragen zum Datensatz, bevor du eine Analyseempfehlung gibst"*
  → reduziert Fehlinterpretationen durch das Modell
- **Meta-Prompting:** KI schreibt den Prompt für eine andere KI
  → ChatGPT optimiert den DeepResearch-Prompt,
  → Claude generiert einen reproduzierbaren Bildgenerierungs-Prompt für DALL-E
  → maximale Konsistenz über Modelle hinweg
- **Schreibstruktur** des Prompts selbst (Überschriften, Sonderzeichen, Markdown, …)

Vgl.
Schulhoff et al. (2025)
Sahoo et al. (2025)
He et al. (2024)
Patil et al. (2024)
Nestler et al. (2026)

OFFENES PROBLEM BEIM PROMPT ENGINEERING

„Wer mehr zum Thema weiß, kann den Output besser beurteilen" *(Nestler et al., 2026)*

- Inhaltliches Wissen, Methoden/Statistikwissen, Programmierkenntnisse sind nötig, um KI-Output optimal zu prüfen
- **KI verstärkt vorhandene Kompetenz – ersetzt sie nicht**

  → wer `lme4` nicht kennt, weiß nicht ob `(1|id)` im Modell korrekt ist
- **Das ist kein Argument gegen KI** – sondern ein Argument für gute methodische Ausbildung *zusätzlich* zu KI-Nutzung

⚠️ *„Fehlererkennung sinkt bei geringer Fachkenntnis"*
→ das Risiko wächst genau dort, wo man sich am meisten auf KI verlässt *(Nestler et al., 2026)*

EIN PAAR TIPPS ZUM PROGRAMMIEREN IN R

Daten einlesen

- `data_raw` = eingelesene Daten
- `data_temp` = Transformationen / Zwischenschritte
- `data_clean` = Finale aufbereitete Daten (abspeichern)

Analyse

- Modelle als Objekte ablegen (spart auch Laufzeit)

Visualisierung

- Ein Plot = eine Frage – Titel und Achsenbeschriftungen immer setzen
- `ggplot2: theme_minimal()` als guter Standard für wissenschaftliche Plots
- Spaghetti-Plots für individuelle Verläufe, Ribbon-Plots für Gruppentrends

Ergebnisdarstellung

- APA-konforme Tabellen: `apaTables`, `gt`, oder `kableExtra`
- Immer: Effektgrößen und Konfidenzintervalle berichten, nicht nur p-Werte

AUFGABE: DATEN EINLESEN UND ANALYSIEREN + PAUSE

- Mache dich mit dem Datensatz `paco_sim_workshop.csv` vertraut
  - Codebook als Hilfe
  - Tipp für Start-Off: Bilde Komposita (`rowMeans`) für alle Skalen

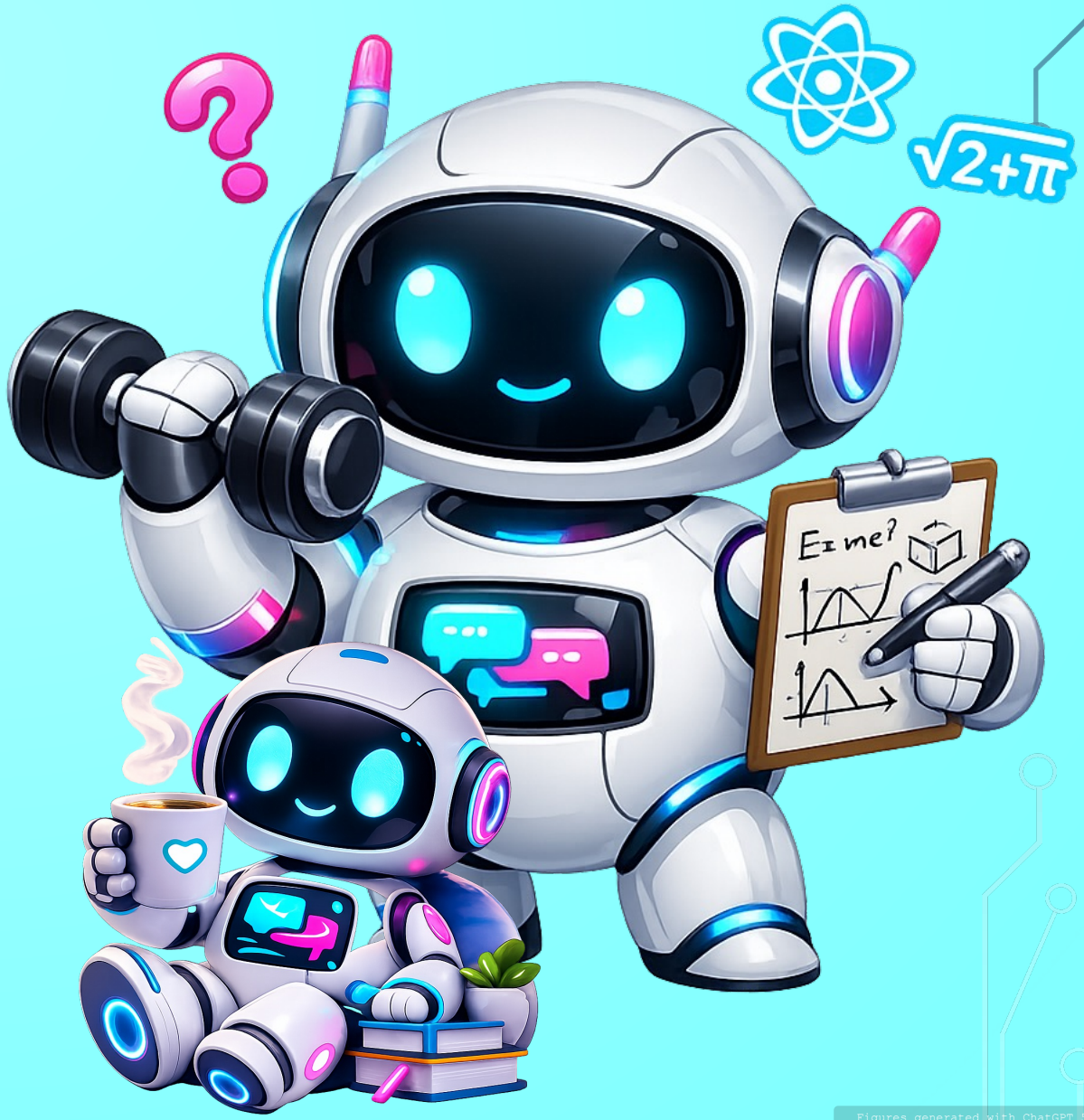
Hypothesen (alle bearbeiten H1–H3):

- **H1:** Kinder unterscheiden sich zwischen Schulformen bereits zu Studienbeginn in ihrem **Selbstwert**
- **H2:** An Tagen mit höherer **Selbstwirksamkeit** berichten Kinder auch einen höheren **Selbstwert** *(within-person)*
- **H3:** Dieser Zusammenhang variiert zwischen Kindern – und hängt ggf. mit **elterlicher Beteiligung oder Überforderung** zusammen

Bonusaufgabe – je nach Interesse:

- **H4:** Der Zusammenhang verändert sich **vor vs. nach Tag 11** (Lockdown-Verschärfung)
- **H5:** Die Selbstwirksamkeit von heute sagt den Selbstwert von **morgen** vorher *(Lagged-Effekt)*
- **H6:** Selbstwert und Selbstwirksamkeit beeinflussen sich **gegenseitig** über die Zeit *(Cross-Lagged)*

*Nutze die 6 Bausteine + Erweiterungen und schreibe Prompts, die es erlauben, den Datensatz einzulesen, aufzubereiten und **mindestens H1–H3 zu analysieren.** Wähle dann eine Bonushypothese.*



Figures generated with ChatGPT 5.3

Ergebnispräsentation

Figures generated with ChatGPT 5.3

KI BEIM SCHREIBEN, DOKUMENTIEREN & PRÄSENTIEREN

- **RMarkdown & Quarto**
  → KI generiert Textbausteine direkt im .Rmd/.qmd-Format
  → Markdown-Struktur im Prompt
  → Markdown-Output nutzbar
  → Methoden-Abschnitte, Ergebnistexte, Tabellenbeschriftungen
- **Tipp:** Prompt-Format = Markdown
  → Output direkt in Quarto einfügbar
  (He et al., 2024)

- **Grafiken & Visualisierung** → ggplot2-Code generieren & kommentieren lassen → Farbpaletten, Themes, Barrierefreiheit → Grafik-Beschreibungen für Manuskripte
- **Reporting-Texte**
  - „Beschreibe dieses Regressionsergebnis für ein APA-Journal"
  - Effektgrößen einordnen lassen
  - Limitation-Abschnitte entwerfen

ERGEBNISDARSTELLUNG IN QUARTO

- Ergebnisse und Grafiken in R erzeugen und aufbereiten (für Präsentationen oder Dokumente)
  - Quarto-HTML, Quarto-PDF, Quarto-Präsentation
- Kommentieren und Erklären
- `cache: true` – lang laufende Modelle einmal berechnen, bei erneutem Rendern nicht neu ausführen
  - Oder Output der Modelle speichern:
    ```
    save(list = c("Result1", "Result2", file = "MyResult.rda")
    ```
- KI generiert **Ergebnistext, Tabellen und Code-Chunks** → direkt einfügbar

Figures generated with ChatGPT 5.3

WARUM MARKDOWN? – CALLBACK

- Strukturierter Prompt → **besserer Output** bei komplexen Modellen *(He et al., 2024)*
- Markdown ist die **gemeinsame Sprache** von Prompt Engineering, KI-Output und wissenschaftlichem Reporting

```markdown
# Überschrift 1
## Überschrift 2

**fett**   *kursiv*   `Code`

- Aufzählung
1. Nummeriert

> Zitat / Blockquote

[Linktext](https://url.de)
```

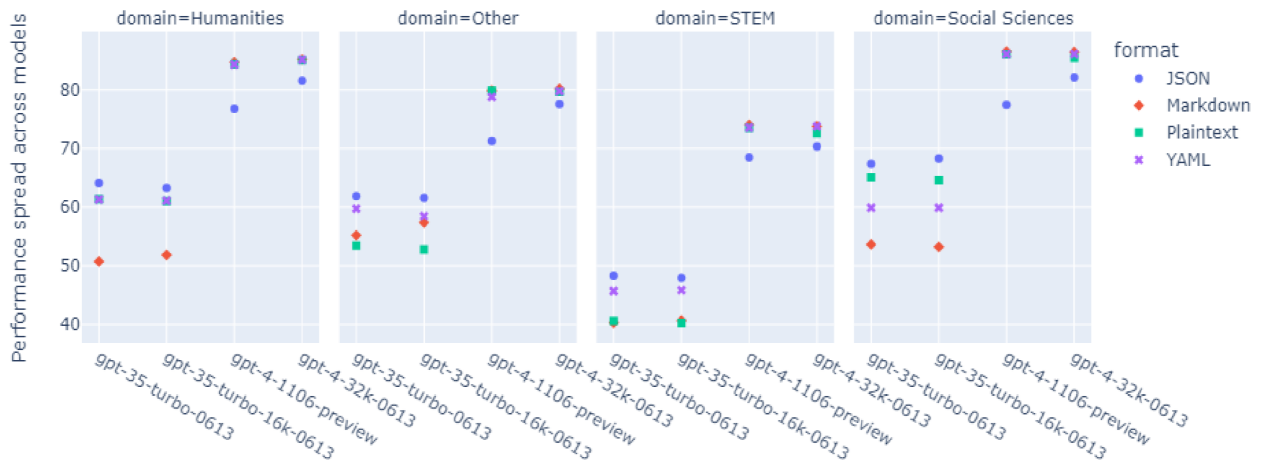


Figure 5: Performance spread across models on MMLU benchmark per domain. Wide performance spread is observed across domains that required different skills.

⚠ *Was du im Prompt verwendest, kannst du direkt in Quarto weiterverwenden*

Figures generated with ChatGPT 5.3

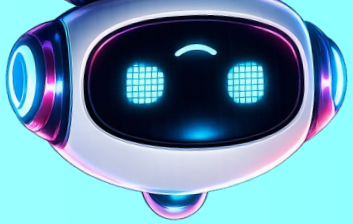
VON MARKDOWN ZU RMARKDOWN/QUARTO:

	Markdown	RMarkdown	Quarto
Text formatieren	✅	✅	✅
R-Code ausführen	❌	✅	✅
Output einbetten	❌	✅	✅
PDF / HTML / Word	teilweise	✅	✅
Mehrere Sprachen	❌	❌	✅ (R, Python, Julia)
Moderner Standard	—	älter	✅ aktuell

⚠️ *Was du im Prompt verwendest, kannst du direkt in Quarto weiterverwenden*

Table generated with Claude Sonnet 4.6 Figures generated with ChatGPT 5.3

QUARTO: EIN BEISPIEL



Quarto Beispiel.qmd*

Render on Save | Render | Run | Publish

Source | Visual | Outline

```
1  ---
2  title: "Quarto Beispiel"
3  author: "Julien P. Irmer"
4  format: html
5  editor: source
6  ---
7
8  # Größte Überschrift
9
10 ## Zweitgrößte Überschrift
11
12 ### Drittgrößte Überschrift
13
14 **Fetter Text**
15
16 *Kursiv geschriebener Text*
17
18 - Aufzählungspunkt 1
19 - Aufzählungspunkt 2
20
21 1. Nummerierte Liste 1
22 2. Nummerierte Liste 2
23
24
25
26
27
28
29
30
31
```

Größte Überschrift
Zweitgrößte Übers...
Drittgrößte Übers...

Environment | History | Connections | Tutorial

Files | Plots | Packages | Help | Viewer | Presentation

Edit | Publish

Quarto Beispiel

AUTHOR
Julien P. Irmer

Größte Überschrift

Zweitgrößte Überschrift

Drittgrößte Überschrift

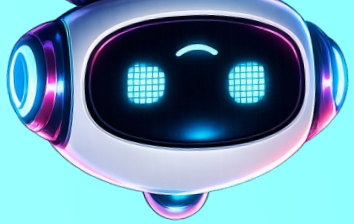
Fetter Text

Kursiv geschriebener Text

- Aufzählungspunkt 1
- Aufzählungspunkt 2

1. Nummerierte Liste 1
2. Nummerierte Liste 2

QUARTO: EIN BEISPIEL



Quarto Beispiel.qmd

Render on Save | Render | Run | Publish

Source | Visual | Outline

````
```{r}
# R-Code-Block
x <- 1:10
y <- x^2
plot(x, y, type = "b", col = "blue",
main = "Quadratische Funktion", xlab = "x", ylab = "y")
```
Inline R-command: Summe von x = `r sum(x)`.

LaTeX Formel: $Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$.

[Link zu RStudio](https://www.rstudio.com/)
````

Größte Überschrift
Zweitgrößte Übers...
Drittgrößte Übers...

46:1 | Drittgrößte Überschrift | Copilot: Waiting for completions... | Quarto

Console | Terminal | Background Jobs

R 4.5.2 · ~/

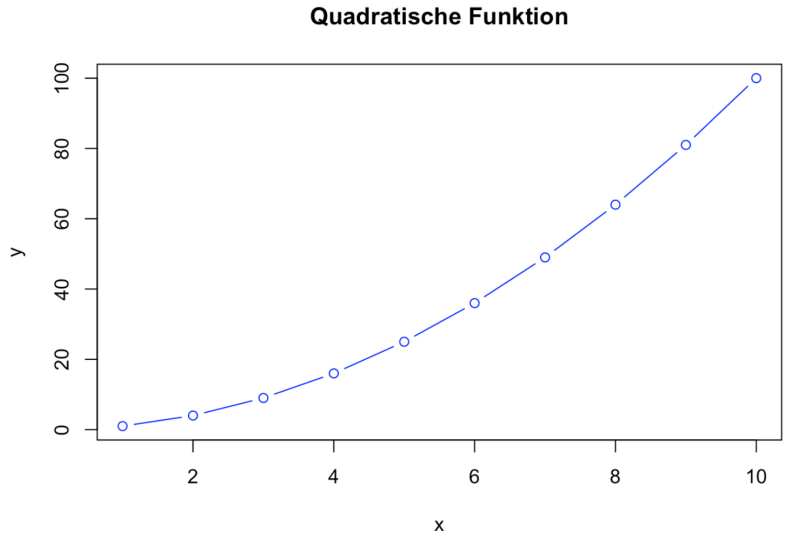
```
Type 'q()' to quit R.

>
```

Environment | History | Connections | Tutorial

Files | Plots | Packages | Help | Viewer | Presentation

Edit | Publish

```r
# R-Code-Block
x <- 1:10
y <- x^2
plot(x, y, type = "b", col = "blue",
main = "Quadratische Funktion", xlab = "x", ylab = "y")
```



Inline R-command: Summe von x = 55.

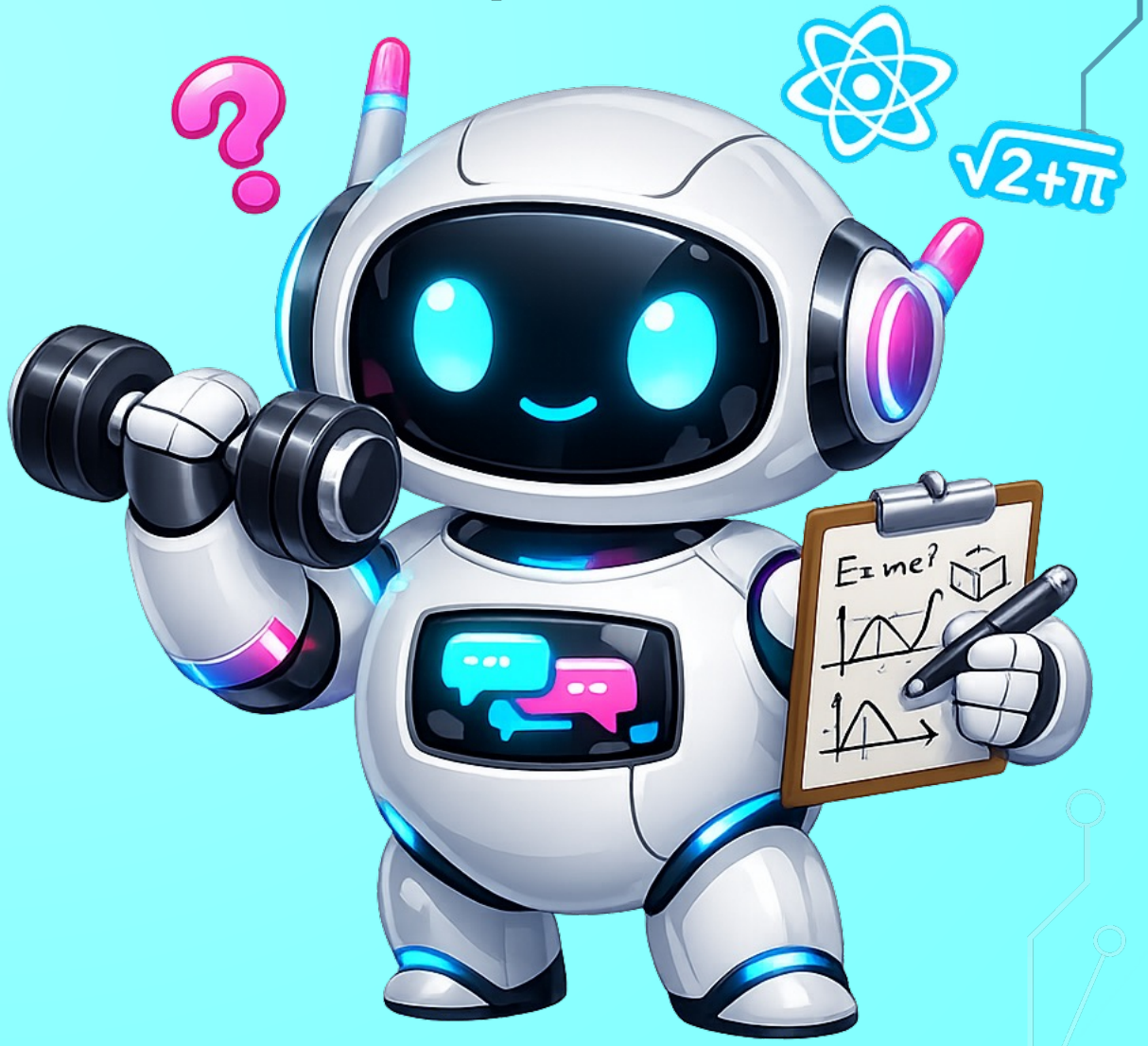
LaTeX Formel: $Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$.

Link zu RStudio

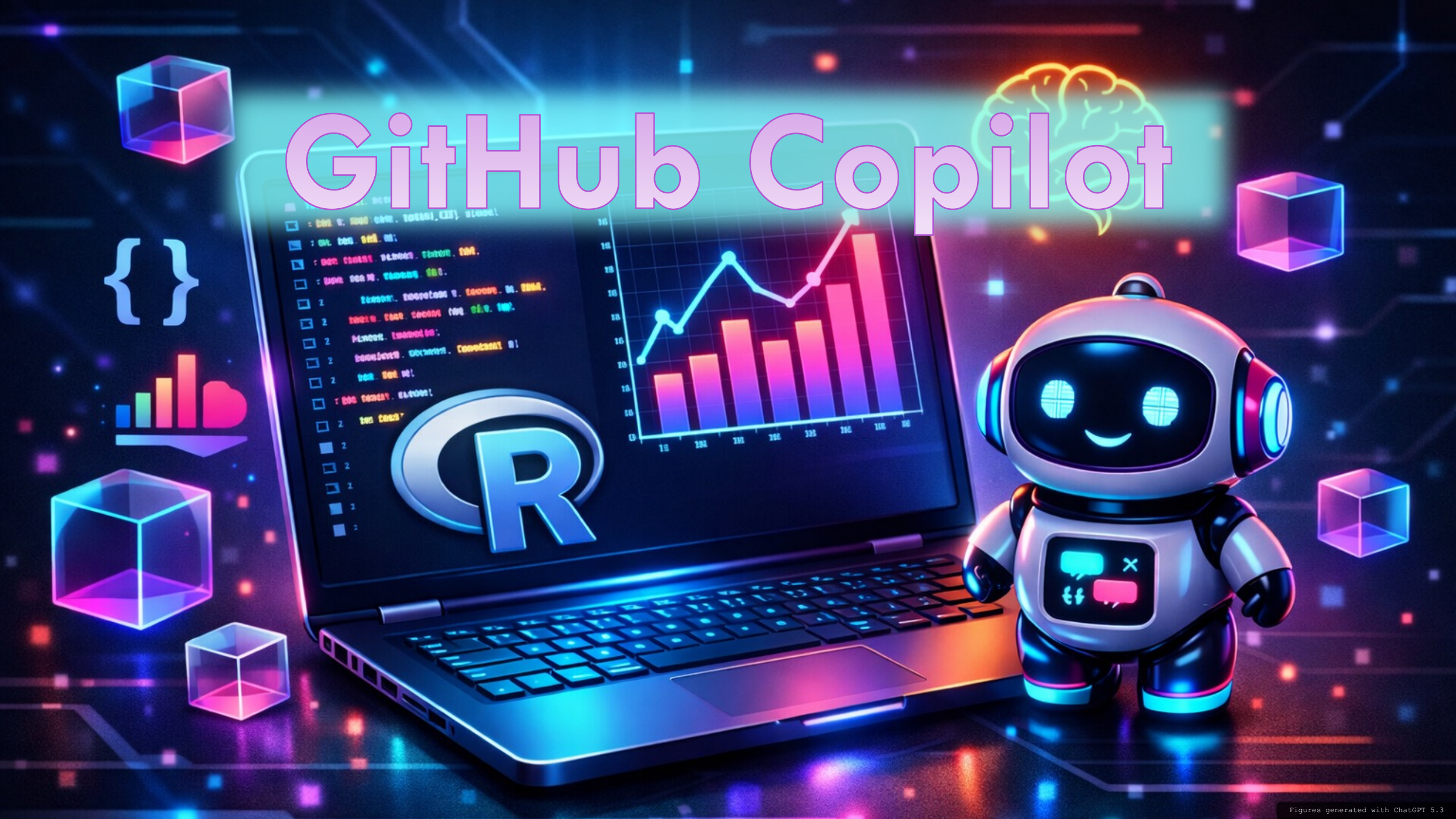
AUFGABE: ERGEBNISDARSTELLUNG IN QUARTO

- Setze deine Analysen von zuvor fort und **verpacke alles in einem Quarto-Dokument**
- Nutze KI wo du kannst – aber immer mit Human in the Loop!

1. **Deskriptivstatistik** (Block 1)
   - Erstelle eine APA-konforme Tabelle mit Mittelwerten, SDs und Korrelationen (z.B. mit apaTables)
   - Visualisiere Zeitverläufe beider Konstrukte getrennt nach Schulform (ggplot2)
2. **Heterogenität visualisieren** – H3 (Block 2)
   - Plotte die Random Effects (Intercepts & Slopes) als Dotplot oder Ridgeline
   - Optional: Simple Slopes für elterliche Beteiligung × Ereignis
3. **Ergebnistext** (Block 3)
   - Lass KI einen ersten Ergebnisabschnitt formulieren – dann **kritisch prüfen und korrigieren**



Figures generated with ChatGPT 5.3


GitHub Copilot
R
Figures generated with ChatGPT 5.3

GITHUB: VERSIONIERUNG, KOLLABORATION & MEHR

Unternehmen & Hintergrund

- Gehört seit 2018 zu **Microsoft**, größte **Code-Hosting-Plattform** weltweit
- Basiert auf **Git** (Linus Torvalds) – Standard-Versionierungssystem

Was Git/GitHub kann:

- **Versionierung**: jede Änderung gespeichert → jederzeit zurückrollbar
- **Branches**: parallel experimentieren ohne Hauptcode zu gefährden
- **Kollaboration**: mehrere Personen, ein Projekt

Warum relevant für die Forschung:

- Skripte & Quarto-Dokumente versionierbar → Reproduzierbarkeit
- GitHub Education: kostenloser Pro-Zugang für Studierende & Lehrende → inkl. Copilot

Figures generated with ChatGPT 5.3

GITHUB COPILOT: KI-UNTERSTÜTZUNG DIREKT IM EDITOR

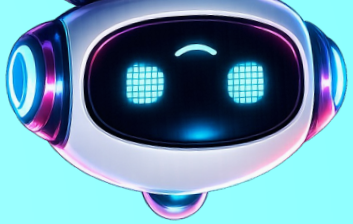
Unternehmen & Modell:

- Entwickelt von **GitHub / Microsoft**, basiert auf **OpenAI-Modellen** (GPT-4o, o3 etc.)
- Modellauswahl je nach Plan möglich: GPT-4o, Claude Sonnet, Gemini – **je besser das Modell, desto höher der Preis**

Stufe	Funktion	Verfügbar in
Autocomplete	Code-Vervollständigung während des Tippens	RStudio, Positron, VS Code
Chat	Fragen stellen, Code erklären, Fehler debuggen, Zugriff auf Repositories	Positron, VS Code, GitHub.com
Agent	Zugriff auf Repositories, Dateien lesen/schreiben, Terminal nutzen, mehrstufige Aufgaben	Positron, VS Code

Kosten: Free: begrenzt, schwächere Modelle
Pro (~10$/Monat): unbegrenzt, Modellauswahl
GitHub Education: Pro kostenlos → github.com/education

GITHUB-COPILOT “INSTALLIEREN”



1. GitHub Konto eröffnen

   https://github.com

   ggf. Educational Programm aktivieren: https://github.com/education

2. GitHub Copilot aktivieren

   https://docs.github.com/en/copilot

3. GitHub Copilot mit RStudio verbinden (enthält auch full guide!)

   https://docs.posit.co/ide/user/ide/guide/tools/copilot.html

GITHUB COPILOT: AUTOCOMPLETE

```
1  load("MyData.rda")
2  names(data) # id time group age gender procrastination self_efficacy gpa
3  library(lme4)
4  model <- lmer(procrastination ~ time
```

Kontext hilft

```
1  load("MyData.rda")
2  names(data) # id time group age gender procrastination self_efficacy gpa
3  library(lme4)
4  model <- lmer(procrastination ~ time * group
```

Durch “Tab” Vorschlag annehmen

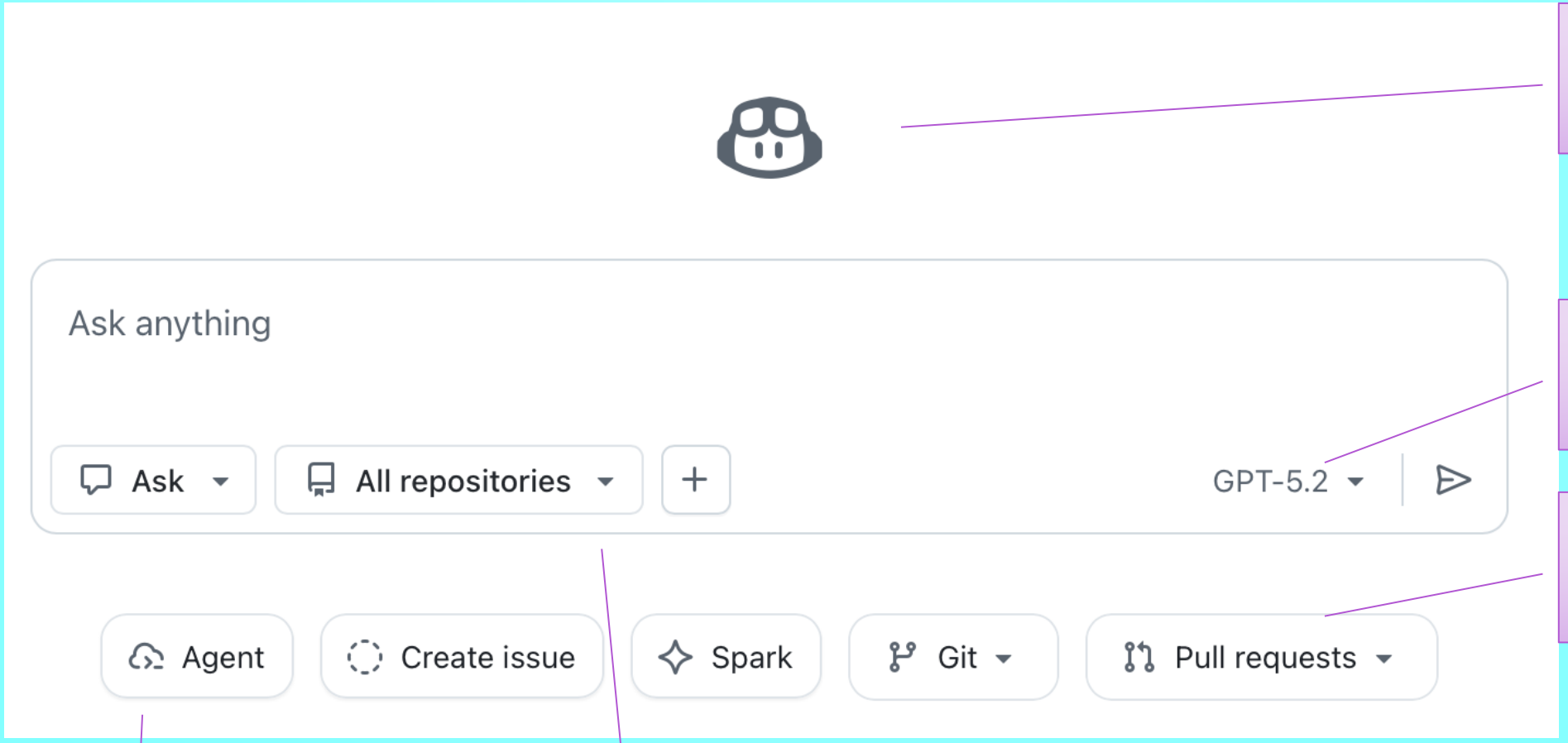
```
1  load("MyData.rda")
2  names(data) # id time group age gender procrastination self_efficacy gpa
3  library(lme4)
4  model <- lmer(procrastination ~ time * group + age
```

Mehrstufiger Prozess: Human in the Loop

```
1  load("MyData.rda")
2  names(data) # id time group age gender procrastination self_efficacy gpa
3  library(lme4)
4  model <- lmer(procrastination ~ time * group + age + gender
```

```
1  load("MyData.rda")
2  names(data) # id time group age gender procrastination self_efficacy gpa
3  library(lme4)
4  model <- lmer(procrastination ~ time * group + age + gender + self_efficacy + gpa + (1 | id), data = data)
```

GITHUB COPILOT: CHAT AUF GITHUB.COM



Friendly animated Copilot

Modellauswahl (ggf. Begrenzt bei free)

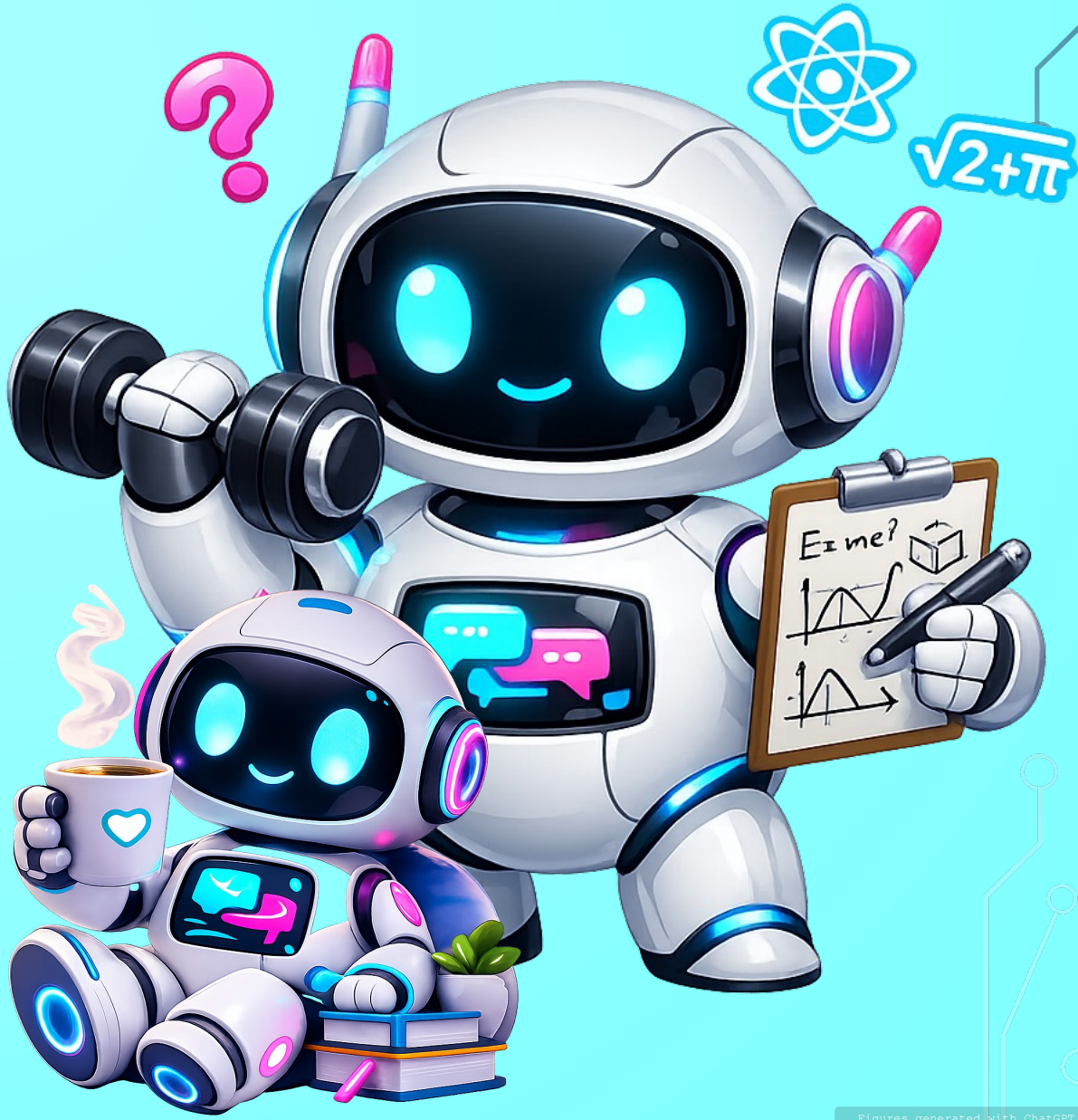
Kann Git-Commands ausführen

Agentenmodus begrenzt nutzbar in Education Version

Zugriff auf online Repositories

AUFGABE: EXPLORIERE COPILOT + PAUSE

- Verwende Copilot, um bspw. Multilevel CFA/SEM zu rechnen für den Zusammenhang zwischen Selbstwirksamkeit und Selbstwert
- Verwende Copilot, um den Ergebnis(Quarto)-Dokument zu verbessern



Best Practices

Figures generated with ChatGPT 5.3

BEST PRACTICES: KI SINNVOLL, SICHER & REPRODUZIERBAR NUTZEN UND ALWAYS “HUMAN IN THE LOOP”!

- ✓ **Modell & Datum dokumentieren** – Versionierung ist Pflicht
- ✓ **Prompt archivieren** – Teil der wissenschaftlichen Methode
- ✓ **Output prüfen** – statistisch, inhaltlich, auf Halluzinationen
- ✓ **Quellen verifizieren** – niemals blind übernehmen
- ✓ **Nutzung transparent offenlegen** – welches Tool, welcher Umfang
- ✓ **Kurze Sessions** – Context Rot durch Neustart vermeiden
- ✓ **Fachkenntnis bleibt Voraussetzung** – KI verstärkt, ersetzt nicht

„KI ist kein Ersatz für methodische Kompetenz – sie ist ein Hebel für sie." (Nestler et al., 2026; Sperl et al., 2026)

Figures generated with ChatGPT 5.3

CALLBACK: WAS IST EIN AGENT?

LLM + Tools + Planung + Autonomie:

ReAct-Schleife: Reason → Act → Observe → Repeat

Die drei Stufen in `Positron` via GitHub Copilot:

Stufe	Modus	Was passiert?
1	Autocomplete	Passiv, vervollständigt beim Tippen
2	Chat (Ask)	Interaktiv, kein Dateizugriff
3	Agent	Liest/schreibt Dateien, nutzt Terminal, plant mehrstufig

POSITRON – WAS IST DAS?

- **Positron** = moderner R/Python-Editor von **Posit** (RStudio-Nachfolger)
- **Rstudio:** Beim reinen R-Programmieren noch deutlich einfacher (Pakete, etc.)
- Zielgruppe von Positron schon fast Data Scientists (mehr Coding)

	RStudio	Positron
Stabilität	✅ ausgereift	⚠️ Beta
KI-Integration	Copilot (Autocomplete)	Copilot: Chat + Agent + Autocomplete
Python	eingeschränkt	✅ vollwertig
Oberfläche	vertraut	moderner, VS-Code-ähnlich
Paketmanagement	✅ integriert (Install-Button)	⚠️ manueller
Quarto	✅	✅
Kostenlos	✅	✅

Table generated with Claude Sonnet 4.6 Figures generated with ChatGPT 5.3

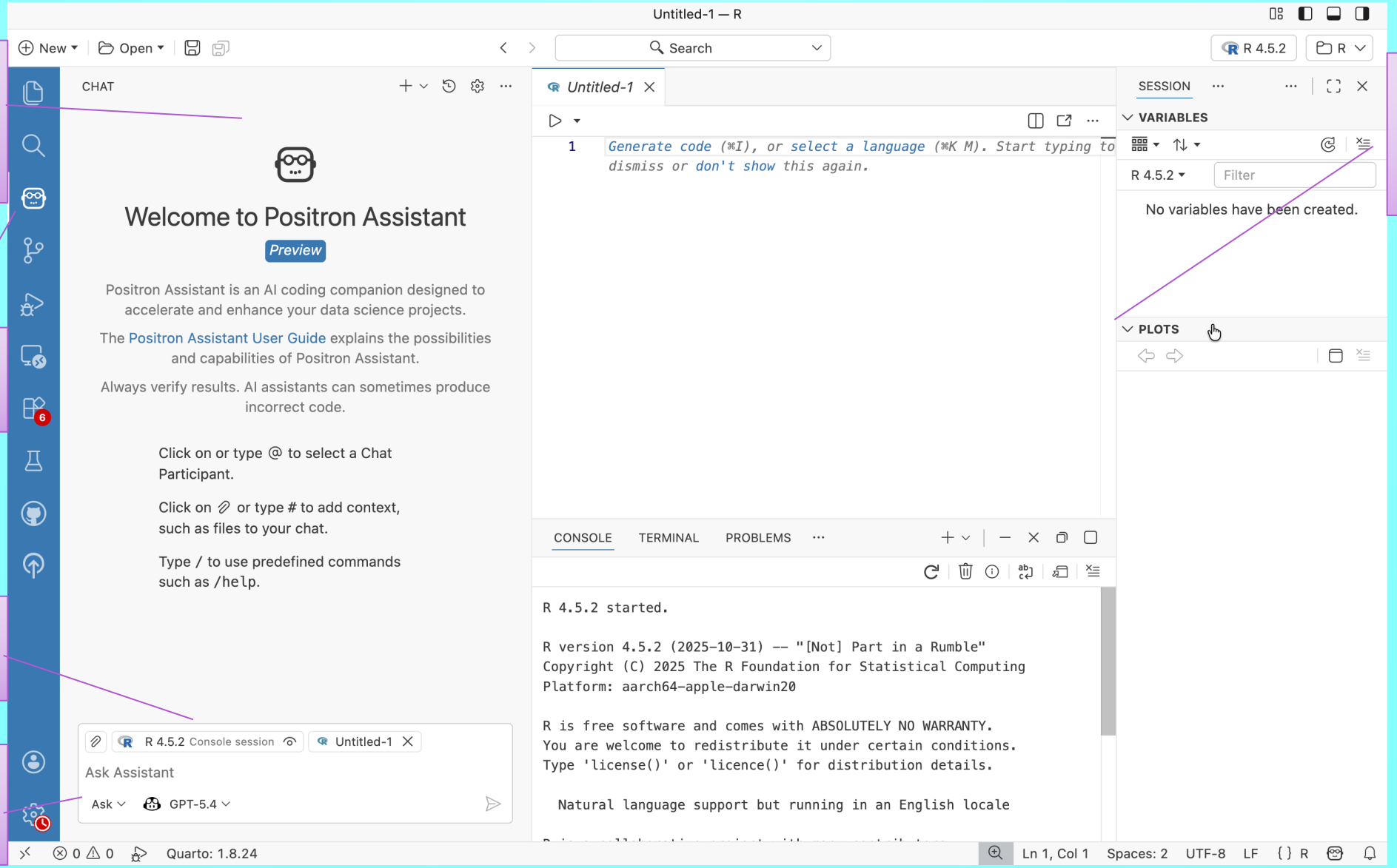
AUSBLICK: AI AGENTS – DEMO IN POSITRON

Neue Spalte im Vergleich zu RStudio

Copilot

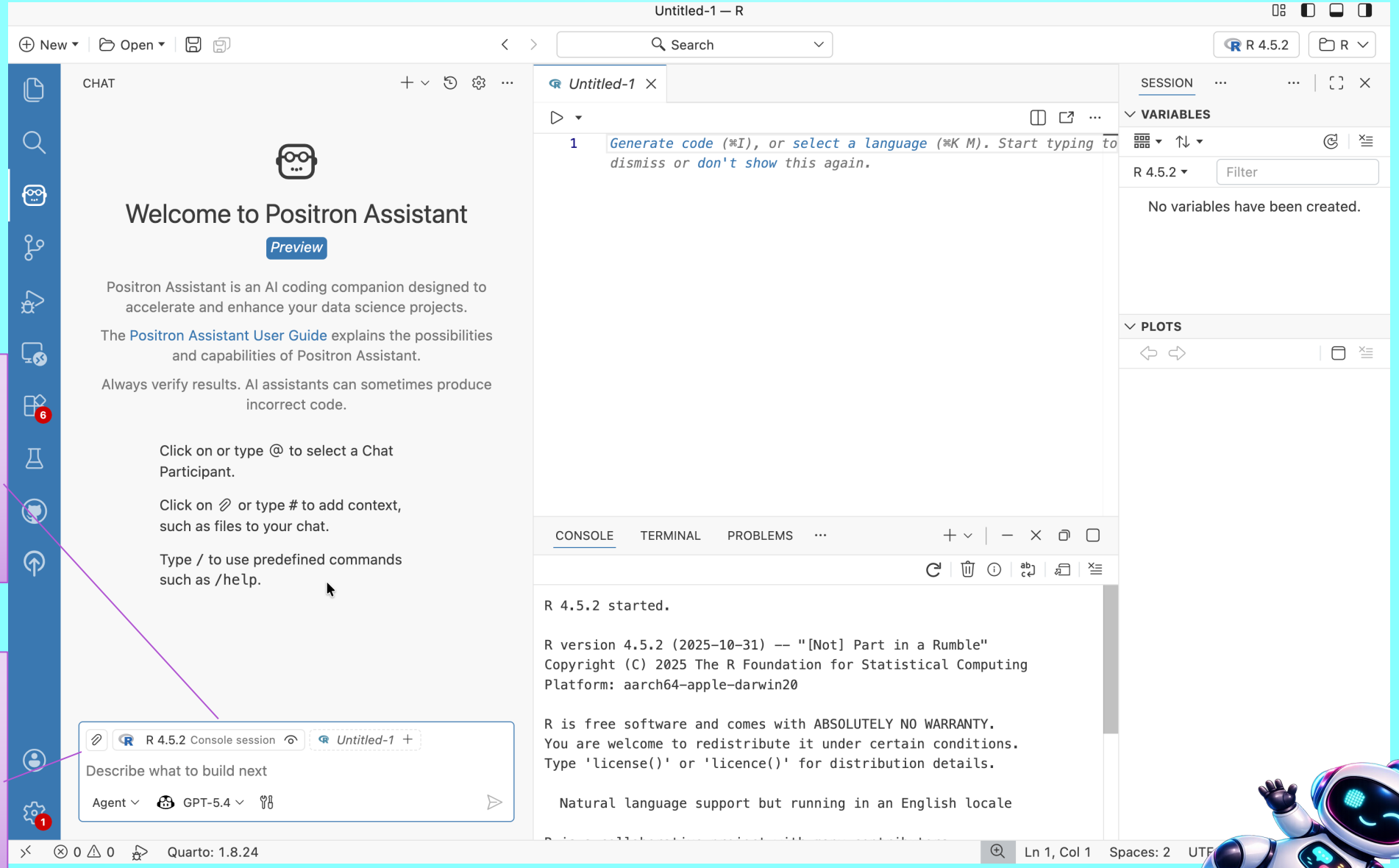
Chatfenster im Editor

Ask = “nur Chat” kein Agent



Normalen 4 Kacheln wie bei RStudio

AUSBLICK: AI AGENTS – DEMO IN POSITRON



Chatfenster im Editor nutzen, um Agenten Auftrag/Ziel zu geben

Agent = Copilot darf als Agent agieren, fragt aber vorher nach

Figures generated with ChatGPT 5.3

ENJOY THE SHOW

Figures generated with ChatGPT 5.3

AUSBLICK: VIBE CODING

- **Was ist Vibe Coding?**
  - Programmieren durch **Gespräch mit einem LLM** statt Zeile für Zeile
  - Ziel beschreiben → Code generieren → testen → Prompt anpassen → wiederholen
  - Begriff geprägt von **Andrej Karpathy** (2025): „*The hottest new programming language is English*"

Chancen

- Schnellere Prototypen, geringere Einstiegshürde
- Fokus auf Intent & Ergebnis statt Syntax

Risiken

- **„Black-Box-Code"**: Code läuft, wird aber nicht verstanden
- Sicherheitslücken, Wartbarkeitsprobleme, starke Kontextabhängigkeit
- „*Context momentum*": frühe Fehler pflanzen sich durch den gesamten Code fort

Einordnung für die Forschungspraxis:

- Vibe Coding ist kein schlampiges Programmieren – es ist ein neues Paradigma mit klaren Trade-offs. Für explorative Skripte nützlich. Für wissenschaftliche Analysepipelines: **nur mit tiefem methodischem Verständnis vertretbar.**

Relativ neues Konzept, Siehe z.B.
Sarkar and Drosos (2025)
Bamil (2025)

Figures generated with ChatGPT 5.3

ZUSAMMENFASSUNG WORKSHOP

🧠 **LLMs als Basis aller Assistenzsysteme** – probabilistische Systeme ohne Weltmodell; Halluzination ist kein Bug sondern Systemlogik

⚙️ **Prompt Engineering** – 6 Bausteine + erweiterte Techniken (Refinement, Meta-Prompting, …) erhöhen Outputqualität systematisch

🔬 **KI im R-Workflow** – von Datenaufbereitung über `R-Pakete` bis zum Quarto-Report; KI als Assistenz, nie als (kompletter) Autopilot

🛠️ **Tools kennen** – Chat (Claude/ChatGPT) für Erklärungen & Code, Copilot für Editor-Integration, Positron als Zukunft

✅ **Human in the Loop** – Fachkompetenz ist der entscheidende Multiplikator:

"wer Pakete und Modelle kennt, bekommt besseren Code" (Nestler et al., 2026)

QUELLEN

Bamil, V. (2025). Vibe Coding: Toward an AI-Native Paradigm for Semantic and Intent-Driven Programming. *arXiv*. https://doi.org/10.48550/arXiv.2510.17842

Binz, M., Alaniz, S., Roskies, A., Aczel, B., Bergstrom, C. T., Allen, C., Schad, D., Wulff, D., West, J. D., Zhang, Q., Shiffrin, R. M., Gershman, S. J., Popov, V., Bender, E. M., Marelli, M., Botvinick, M. M., Akata, Z., & Schulz, E. (2025). How should the advancement of large language models affect the practice of science? *Proceedings of the National Academy of Sciences, 122*, e2401227121. https://doi.org/10.1073/pnas.2401227121

Blackwell, R. E., Barry, J., & Cohn, A. G. (2024). *Towards reproducible LLM evaluation: Quantifying uncertainty in LLM benchmark scores* (arXiv:2410.03492). arXiv. https://arxiv.org/abs/2410.03492

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., … Amodei, D. (2020). Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems, 33*, 1877–1901.

Bubeck, S., Szegedy, C., Tassiulas, L., et al. (2024). Response generated by large language models depends on the structure of the prompt. *Journal of Medical Internet Research, 26*, e51866. https://doi.org/10.2196/51866

He, J., Rungta, M., Koleczek, D., Sekhon, A., Wang, F. X., & Hasan, S. (2024). *Does Prompt Formatting Have Any Impact on LLM Performance?* arXiv. https://doi.org/10.48550/arXiv.2411.10541

Liu, X., Chen, Y., & Li, J. (2025). *A systematic survey of prompt engineering in large language models: Techniques and applications* (arXiv:2402.07927). arXiv. https://arxiv.org/abs/2402.07927

Lott, M. (2025). Tracking AI: Monitoring artificial intelligence [Dataset/Website]. *Maximum Truth Project*. https://www.trackingai.org/home

Nestler, S., Humberg, S., Debelak, R., Heck, D. W., Henninger, M., Voelkle, M. C., Frick, S., Irmer, J. P., Scharf, F., & Frey, A. (2026). *Automating the scientist? Methodische Perspektiven auf die Nutzung von KI in der psychologischen Forschung* [Preprint]. https://doi.org/10.31234/osf.io/dqxsu_v1

Patil, R., Heston, T. F., & Bhuse, V. (2024). Prompt Engineering in Healthcare. *Electronics, 13*(15), 2961. https://doi.org/10.3390/electronics13152961

Poole, D. L., & Mackworth, A. K. (2017). *Artificial intelligence: Foundations of computational agents*. Cambridge University Press.

Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., & Chadha, A. (2025). A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. *arXiv*. https://doi.org/10.48550/arXiv.2402.07927

Schulhoff, S., Ilie, M., Balepur, N., Kahadze, K., Liu, A., Si, C., Li, Y., Gupta, A., Han, H., Schulhoff, S., Dulepet, P. S., Vidyadhara, S., Ki, D., Agrawal, S., Pham, C., Kroiz, G., Li, F., Tao, H., Srivastava, A., … Resnik, P. (2025). The Prompt Report: A Systematic Survey of Prompt Engineering Techniques. *arXiv*. https://doi.org/10.48550/arXiv.2406.06608

Sarkar, A., & Drosos, I. (2025). Vibe coding: Programming through conversation with artificial intelligence. *arXiv*. https://doi.org/10.48550/arXiv.2506.23253

Shojaee, P., Mirzadeh, I., Alizadeh, K., Horton, M., Bengio, S., & Farajtabar, M. (2025). The Illusion of Thinking: Understanding the Strengths and Limitations of Reasoning Models via the Lens of Problem Complexity. *arXiv*. https://doi.org/10.48550/arXiv.2506.06941

Sperl, M. F. J., Baumgärtner, L., Bach, K. M., Bamberg, C., Behlau, C., Bergmann, B., Bienefeld, M., Bleckmann, E., Danböck, S. K., Dreston, J. H., Eckardt, V. C., Frick, S., Friehs, M.-T., Handke, L., Hein, I., Hutmacher, F., Irmer, J. P., Kause, A., Kern, M., … Neef, N. E. (2026). *Künstliche Intelligenz bei Abschlussarbeiten, Dissertationen und Habilitationsschriften in der Psychologie* [Preprint].

Su, J. (2023, August 1). *Master the perfect ChatGPT prompt formula (in just 8 minutes)* [Video]. YouTube. https://www.youtube.com/watch?v=jC4v5AS4RIM

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. ukasz, & Polosukhin, I. (2017). Attention is All you Need. *Advances in Neural Information Processing Systems, 30*.

Walter, Y. (2024). Embracing the future of Artificial Intelligence in the classroom: The relevance of AI literacy, prompt engineering, and critical thinking in modern education. *International Journal of Educational Technology in Higher Education, 21*(1), 15. https://doi.org/10.1186/s41239-024-00448-3

White, J., Fu, Q., Hays, S., Sandhu, J., Olea, C., Hays, M., Elnashar, A., Goyal, A., & Schmidt, D. C. (2023). *A prompt pattern catalog to enhance prompt engineering with ChatGPT*. arXiv. https://arxiv.org/abs/2302.11382


Ende
R
Figures generated with ChatGPT 5.3