

Cronograma

Introdução à Programação em R com GitHub, ChatGPT e Claude

Vinícius Silva Junqueira

2025-10-08

Sumário

1	Informações Gerais do Curso	3
2	Pré-requisitos Técnicos	4
2.1	Instalações obrigatórias (para todos os sistemas)	4
3	Estrutura Pedagógica e Filosofia	5
4	Pacotes R Necessários	6
4.1	Instalação no Dia 1 a 3 (núcleo do curso)	6
4.2	Pacotes de IA (Dia 4)	6
5	Dia 1: 17/11 (Segunda) — Fundamentos e Ambiente de Trabalho	7
5.1	19h00 - 20h30 Ambientação e Setup Completo	7
5.2	20h30 - 20h50 Intervalo	7
5.3	20h50 - 22h00 Fundamentos do R	7
6	Dia 2: 18/11 (Terça) — Lógica, Funções e Introdução ao Tidyverse	8
6.1	19h00 - 20h30 Programação em R	8
6.2	20h30 - 20h50 Intervalo	8
6.3	20h50 - 22h00 Introdução ao Tidyverse	8
7	Dia 3: 24/11 (Segunda) — Transformação, I/O e Visualização	9
7.1	19h00 - 20h30 Transformação e I/O de Dados	9
7.2	20h30 - 20h50 Intervalo	9
7.3	20h50 - 22h00 Visualização com ggplot2	9
8	Dia 4: 25/11 (Terça) — Integração do ChatGPT e Claude no RStudio	10
8.1	19h00 - 19h30 Conceitos e modelos	10
8.2	19h30 - 20h15 Configuração de chaves e ambiente	10
8.3	20h15 - 20h30 Intervalo	10
8.4	20h30 - 21h00 RStudio + gptstudio (ChatGPT)	10
8.5	21h00 - 21h30 RStudio + chattr (Claude)	11
8.6	21h30 - 22h00 Exercício guiado de integração	11

9	Troubleshooting por Sistema Operacional	13
9.1	Windows	13
9.2	macOS	13
9.3	Linux (Ubuntu/Debian)	13
9.4	Problemas comuns (todos os SO)	13
10	Recursos Adicionais	14
11	Contato	15

1 Informações Gerais do Curso

Carga horária total: 16 horas

Datas: 17/11, 18/11, 24/11 e 25/11

Horário: 19h00 às 22h00 (com intervalo de 20 minutos às 20h30)

Tempo líquido por dia: 2h40 de conteúdo efetivo

Público-alvo: Iniciantes de diversas áreas (ciências agrárias, saúde, economia, biologia, ciências sociais)

Material: Repositório GitHub com datasets, scripts e exercícios

Repositório original: <https://github.com/viniciusjunqueira/curso-r-github-ia>

Sistemas Operacionais: Windows, macOS e Linux (curso compatível com todos)

Estrutura do curso (visão geral):

- Dia 1: Fundamentos de R + Ambiente reprodutível (RStudio, Git, GitHub, fork).
- Dia 2: Lógica, condicionais, funções, tidyverse básico.
- Dia 3: Transformações com tidyr/dplyr, I/O e visualização com ggplot2.
- Dia 4: Integração prática do ChatGPT e do Claude dentro do RStudio.

2 Pré-requisitos Técnicos

2.1 Instalações obrigatórias (para todos os sistemas)

1. R (versão 4.3 ou superior)
Download: <https://cran.r-project.org/>
2. RStudio Desktop (2023.09+)
Download: <https://posit.co/download/rstudio-desktop/>
3. Git
Windows: <https://git-scm.com/download/win>
macOS: verifique com `git --version` (ou use o instalador)
Linux (Ubuntu/Debian):

```
sudo apt update  
sudo apt install git
```
4. Conta no GitHub
<https://github.com/signup>

3 Estrutura Pedagógica e Filosofia

- Multiplataforma: conteúdo compatível com Windows, macOS e Linux
- RStudio como IDE padrão
- Teoria seguida de prática guiada
- Commits diários no fork do aluno
- IA como ferramenta para explicação, depuração e geração de material
- Portabilidade: `here::here()` e projetos `.Rproj`

4 Pacotes R Necessários

4.1 Instalação no Dia 1 a 3 (núcleo do curso)

```
install.packages(c(
  "tidyverse", "here", "janitor", "skimr",
  "readxl", "writexl", "rmarkdown", "knitr",
  "lubridate", "scales", "patchwork", "broom",
  "palmerpenguins"
))
```

4.2 Pacotes de IA (Dia 4)

```
install.packages(c("gptstudio", "chattr", "httr2", "jsonlite"))
```

5 Dia 1: 17/11 (Segunda) — Fundamentos e Ambiente de Trabalho

5.1 19h00 - 20h30 | Ambientação e Setup Completo

Apresentação do curso, objetivos e metodologia.

Por que R, GitHub e IA.

Checklist de instalações (R, RStudio, Git).

Configuração do Git (user.name/user.email).

Autenticação no GitHub (PAT recomendado).

Fazer fork do repositório do curso.

Clonar o fork no RStudio (Projeto .Rproj).

Verificar `git remote -v` apontando para o fork do aluno.

5.2 20h30 - 20h50 | Intervalo

5.3 20h50 - 22h00 | Fundamentos do R

Objetos e estruturas básicas: vetores, listas, data.frames, fatores.

Funções básicas: `c()`, `length()`, `class()`, `typeof()`.

Exploração: `str()`, `head()`, `tail()`, `names()`, `dplyr::glimpse()`, `summary()`.

Indexação: `[]`, `$`, subsetting lógico.

Prática guiada: criar vetores e data.frames, manipular objetos.

Commit sugerido:

```
git add scripts/01_fundamentos.R
git commit -m "Fundamentos do R - Dia 1"
git push origin main
```

6 Dia 2: 18/11 (Terça) — Lógica, Funções e Introdução ao Tidyverse

6.1 19h00 - 20h30 | Programação em R

Operadores lógicos e relacionais.

Condicionais: if/else, ifelse() (vetorizado), dplyr::case_when().

Loops e funções: for vs. vetorização, criação de funções, validação de entradas.

Boas práticas e debugging: snake_case, comentários, leitura de traceback.

Mini demonstração de como a IA pode explicar um erro simples.

6.2 20h30 - 20h50 | Intervalo

6.3 20h50 - 22h00 | Introdução ao Tidyverse

Filosofia tidyverse e uso de pipes (%>% e |>).

Verbos essenciais do dplyr: filter(), select(), mutate(), arrange(), summarize(), group_by().

Datas com lubridate: ymd/dmy/mdy, year/month/wday, today/now.

Exemplo integrado com palmerpenguins.

Commit sugerido:

```
git commit -m "Lógica, funções e tidyverse - Dia 2"
```


7 Dia 3: 24/11 (Segunda) — Transformação, I/O e Visualização

7.1 19h00 - 20h30 | Transformação e I/O de Dados

Reshape com tidyr: `pivot_longer()`, `pivot_wider()`, `separate()`, `unite()`.

Tratamento de NAs: `is.na()`, `drop_na()`, `replace_na()`, `fill()`.

Leitura/Escrita: `readr::read_csv()`, `read_csv2()`, `readxl::read_excel()`.

Portabilidade com `here::here()` e organização de projetos.

Ferramentas úteis: `janitor::clean_names()`, `skimr::skim()`.

7.2 20h30 - 20h50 | Intervalo

7.3 20h50 - 22h00 | Visualização com ggplot2

Gramática de gráficos: camadas, `aesthetics`, `geoms` comuns.

Gráficos: dispersão, barras, `boxplot`, linhas, histograma/densidade.

Combinação com `patchwork`, formatos com `scales`.

Personalização e salvamento: `theme_*`, `labs()`, `ggsave()`.

Commit sugerido:

```
git commit -m "Transformação, I/O e visualização - Dia 3"
```

8 Dia 4: 25/11 (Terça) — Integração do ChatGPT e Claude no RStudio

Objetivo do dia: capacitar o aluno a usar ChatGPT (OpenAI) e Claude (Anthropic) diretamente no RStudio para explicar erros, revisar e gerar código, criar rascunhos de relatórios e automatizar pequenas rotinas via API.

8.1 19h00 - 19h30 | Conceitos e modelos

Panorama rápido sobre LLMs, APIs, limites e custos.

Boas práticas de uso responsável de IA: privacidade, dados sensíveis, versionamento de código gerado.

Comparação prática: quando usar ChatGPT e quando usar Claude.

8.2 19h30 - 20h15 | Configuração de chaves e ambiente

Variáveis de ambiente no R: uso de `~/.Renviron` e `Sys.getenv()`.

Criação de chaves de API e configuração local.

Nomes convencionados: - `OPENAI_API_KEY` para ChatGPT (OpenAI) - `ANTHROPIC_API_KEY` para Claude (Anthropic)

Exemplo de `~/.Renviron`:

```
OPENAI_API_KEY=coloque_sua_chave_aqui
ANTHROPIC_API_KEY=coloque_sua_chave_aqui
```

Teste rápido no R:

```
Sys.getenv("OPENAI_API_KEY")
Sys.getenv("ANTHROPIC_API_KEY")
```

Instalação de pacotes:

```
install.packages(c("gptstudio", "chattr", "httr2", "jsonlite"))
```

8.3 20h15 - 20h30 | Intervalo

8.4 20h30 - 21h00 | RStudio + gptstudio (ChatGPT)

Abertura dos Addins do gptstudio no RStudio (chat pane e code assistant).

Uso no editor: seleção de código e prompt de revisão.

Exemplos típicos: explicar erro, refatorar função, gerar testes unitários simples.

Script de exemplo (via API manual com httr2):

```
library(httr2); library(jsonlite)

endpoint <- "https://api.openai.com/v1/chat/completions"
prompt <- "Explique o que este código faz e sugira melhorias:\n\nx <- 1:10; mean(x)"
body <- list(
  model = "gpt-4o-mini",
  messages = list(list(role="user", content=prompt))
)
```

```
req <- request(endpoint) |>
  req_method("POST") |>
  req_headers(Authorization = paste("Bearer", Sys.getenv("OPENAI_API_KEY"))) |>
  req_body_json(body)

resp <- req_perform(req)
json <- resp_body_json(resp)
cat(json$choices[[1]]$message$content)
```

8.5 21h00 - 21h30 | RStudio + chattr (Claude)

Fluxo básico com chattr::chat_claude().

Exemplos práticos: explicar um traceback, sugerir validação de argumentos, gerar esqueleto de RMarkdown.

Script de exemplo (API manual com httr2):

```
library(httr2); library(jsonlite)

endpoint <- "https://api.anthropic.com/v1/messages"
prompt <- "Revise a função abaixo e a torne mais robusta a NAs.\n\nsoma_media <- function(x){ s

body <- list(
  model = "claude-3-5-sonnet-latest",
  max_tokens = 300,
  messages = list(list(role="user", content=prompt))
)

req <- request(endpoint) |>
  req_method("POST") |>
  req_headers(
    Authorization = paste("Bearer", Sys.getenv("ANTHROPIC_API_KEY")),
    "anthropic-version" = "2023-06-01",
    "content-type" = "application/json"
  ) |>
  req_body_json(body)

resp <- req_perform(req)
json <- resp_body_json(resp)
cat(json$content[[1]]$text)
```

8.6 21h30 - 22h00 | Exercício guiado de integração

Tarefa 1: usar gptstudio para revisar um script curto de dplyr e propor 2 melhorias.

Tarefa 2: usar chattr para gerar uma função em R que: - receba um data.frame e uma coluna numérica - remova NAs, retorne média e desvio-padrão com nomes claros - inclua validação de tipos e mensagens de erro úteis

Entrega esperada no fork do aluno: - scripts/04_ia_integracao_gptstudio.R - scripts/04_ia_integracao_claude.R

- docs/relatorio_ia.Rmd com um parágrafo descrevendo o que a IA sugeriu, o que foi adotado e por quê.

Checklist final: - variáveis de ambiente lidas com Sys.getenv() - addins do gptstudio funcionando - chamada mínima via httr2 para cada API - commit e push no fork

```
git add scripts/04_*.R docs/relatorio_ia.Rmd
git commit -m "Integração ChatGPT e Claude no RStudio (Dia 4)"
git push origin main
```

9 Troubleshooting por Sistema Operacional

9.1 Windows

Git não encontrado: reinstalar Git (opção Git from the command line...).

Acentos estranhos: garantir UTF-8 ao salvar, ou usar locale(encoding="latin1") quando necessário.

Pacotes com erro: tentar instalar na biblioteca do usuário.

9.2 macOS

xcrun error com Git: `xcode-select --install`

LaTeX não encontrado: `tinytex::install_tinytex()`

9.3 Linux (Ubuntu/Debian)

Compilação de pacotes:

```
sudo apt install build-essential libcurl4-openssl-dev libssl-dev libxml2-dev  
sudo apt install libfontconfig1-dev libharfbuzz-dev libfribidi-dev
```

9.4 Problemas comuns (todos os SO)

Permission denied ao fazer push: corrigir origin para o fork do aluno:

```
git remote set-url origin https://github.com/SEU-USUARIO/curso-r-github-ia.git
```

API Key não encontrada: configurar no ~/.Renviron e reiniciar o R.

10 Recursos Adicionais

R for Data Science (2e): <https://r4ds.hadley.nz/>

Happy Git with R: <https://happygitwithr.com/>

Cheatsheets Posit: <https://posit.co/resources/cheatsheets/>

palmerpenguins: <https://allisonhorst.github.io/palmerpenguins/>

Datasets: TidyTuesday, Brasil.io, Kaggle

11 Contato

Instrutor: Vinícius Silva Junqueira

Email: junqueiravinicius@hotmail.com

GitHub: <https://github.com/viniciusjunqueira/curso-r-github-ia>

Lattes: <http://lattes.cnpq.br/4686677580216927>

LinkedIn: www.linkedin.com/in/junqueiravinicius

Última atualização: 2025-10-08

Versão: 6.0 – Dia 4 100% integração com IA no RStudio