UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE CIÊNCIAS EXATAS E DA NATUREZA
DEPARTAMENTO DE ESTATÍSTICA
DISCIPLINA: INTRODUÇÃO AOS SOFTWARES ESTATÍSTICOS

TÍTULO

Aila Soares Ferreira
matrícula: 20240045022

João Pessoa - PB

Julho 2025

1.0 INTRODUÇÃO

Com os avanços do *Big Data* - termo que descreve grandes volumes de dados de alta velocidade, complexos e variáveis que passaram a exigir técnicas e tecnologias avançadas para captura, armazenamento, distribuição, gerenciamento e análise de informações (Foundation 2012) - observamos que o valor desses dados só é plenamente aproveitado quando utilizado para tomada de decisões estratégicas. Para isso, as organizações precisam de processos eficientes que transformem dados brutos, muitas vezes dispersos e dinâmicos, em *insights* significativos (Gandomi e Haider 2015).

Além disso, conforme um projeto avança, arquivos são constantemente modificados e compartilhados entre colaboradores, exigindo gestão adequada para garantir integridade e acessibilidade. O gerenciamento de dados envolve não apenas armazenamento, mas também preparação e recuperação eficiente para análise (Bryan 2017).

Nesse contexto, este trabalho foi desenvolvido como requisito da disciplina Introdução aos Softwares Estatísticos, com o objetivo de integrar ferramentas fundamentais para a prática atual da estatística:

1. **Git e GitHub** - Sistema de controle de versões distribuído e plataforma de hospedagem
2. **Quarto** - Ferramenta para criação de relatórios dinâmicos e reprodutíveis
3. **Python** - Linguagem de programação para análise de dados

2.0 Controle de Versão

Antes de falarmos sobre Git e GitHub, precisamos entender o conceito de **controle de versão**. Esse sistema registra todas as mudanças em arquivos ao longo do tempo, permitindo que você volte a versões anteriores quando necessário. Ele guarda todo o histórico de alterações em um banco de dados especial. Se acontecer algum erro, é possível voltar atrás e comparar diferentes versões do projeto (Bryan e Hester 2020).

2.2 Benefícios do Controle de Versão

Os principais benefícios são:

1. **Histórico completo** de todas as mudanças:
    - Todas as alterações feitas por diferentes pessoas
    - Criação e exclusão de arquivos
    - Mudanças no conteúdo
2. **Ramificação (branching) e junção (merging)**:
    - Trabalhar em partes separadas do projeto
    - Criar "ramos" para desenvolver coisas novas
    - Juntar tudo depois no projeto principal
3. **Rastreamento**:

- Saber quem fez cada mudança
- Quando e por que foi feita
- Comparar versões diferentes

Os sistemas de controle de versão passaram por significativas melhorias nas últimas décadas. Dentre as soluções atualmente disponíveis, o Git se consolida como uma das ferramentas mais adotadas para controle de versão (Chacon e Straub 2014; Bryan e Hester 2020).

3.0 Git e GitHub

O Git é um sistema de controle de versão distribuído, criado por Linus Torvalds em 2005 com o objetivo de gerenciar o desenvolvimento colaborativo do kernel do Linux. Sua arquitetura permite que múltiplos desenvolvedores trabalhem simultaneamente em um projeto, mantendo um histórico completo e seguro das alterações realizadas.

Com o passar dos anos, o Git passou a ser amplamente adotado em diferentes áreas, incluindo a ciência de dados, onde tem se mostrado uma ferramenta eficiente no gerenciamento de projetos complexos. Esses projetos frequentemente envolvem uma coleção heterogênea de arquivos, como scripts de análise computacional, relatórios dinâmicos, bases de dados e visualizações gráficas (Bryan 2017) (Chacon e Straub 2014).

3.1 Fluxo de Trabalho Básico do Git

O fluxo de trabalho básico com Git pode ser resumido nas seguintes etapas:

1. **Modificação**
   - Edição de arquivos na árvore de trabalho
   - Criação, remoção ou alteração de documentos
2. **Preparação**
   - Seleção das mudanças que devem ser incluídas no próximo commit (`git add`)
   - Organização lógica das alterações
3. **Consolidação**
   - Registro definitivo das mudanças com uma mensagem descritiva (`git commit`)
   - Criação de pontos de restauração no histórico do projeto
4. **Sincronização**
   - Integração com repositórios remotos por meio dos comandos `git push` e `git pull`
   - Garantia de que todas as versões do projeto permaneçam consistentes entre os colaboradores

3.2 GitHub

O GitHub é uma plataforma de hospedagem de código-fonte que complementa o uso do Git, fornecendo uma interface gráfica baseada na web e recursos avançados para o gerenciamento de repositórios Git. Enquanto o Git é uma ferramenta local utilizada para registrar e controlar as alterações em arquivos, o GitHub funciona como um serviço remoto, permitindo que esses repositórios sejam armazenados, sincronizados e compartilhados pela internet.

Muitas operações podem ser feitas inteiramente no Github, como exemplo: **Visualizar e editar arquivos; revisar e gerenciar solicitações de mudanças; e Controlar permissões de acesso aos projetos**.

3.3 Comandos básicos: `git init`, `git clone`, `git add`, `git commit`, `git push`, `git pull`

3.3.1 `git init`

O comando `git init` cria um novo repositório do Git. Ele é utilizado para transformar uma pasta comum do computador em um repositório Git, ou seja, um ambiente com controle de versão. Isso permite acompanhar todas as alterações feitas no projeto ao longo do tempo.

A seguir, apresentamos um exemplo prático com base no desenvolvimento deste próprio relatório.

Exemplo prático: iniciando o Git em um projeto Quarto

Etapas

1. Criei uma nova pasta:
2. Dentro da pasta, criei um novo arquivo chamado `relatorio2.qmd`.
3. A seguir, abri o terminal dentro da própria pasta do projeto (isso pode ser feito direto pelo Visual Studio Code ou Positron).

Transformando em repositório Git:
Este é um bloco simples que funciona em qualquer PDF
`git init` - Comando para iniciar repositório

conceitos basicos em

CONCLUSÃO

REFERÊNCIAS

Bryan, Jennifer. 2017. «Excuse me, do you have a moment to talk about version control?» *PeerJ Preprints* 5: e3159v2. https://doi.org/10.7287/peerj.preprints.3159v2.

Bryan, Jennifer, e Jim Hester. 2020. *Happy Git and GitHub for the useR*. https://happygitwithr.com/.

Chacon, Scott, e Ben Straub. 2014. *Pro Git*. 2.ª ed. Apress. https://git-scm.com/book/en/v2.

Foundation, TechAmerica. 2012. *Big Data Recommendations*. TechAmerica.

Gandomi, A., e M. Haider. 2015. «Beyond the hype: Big data concepts, methods, and analytics». *International Journal of Information Management* 35 (2): 137–44.