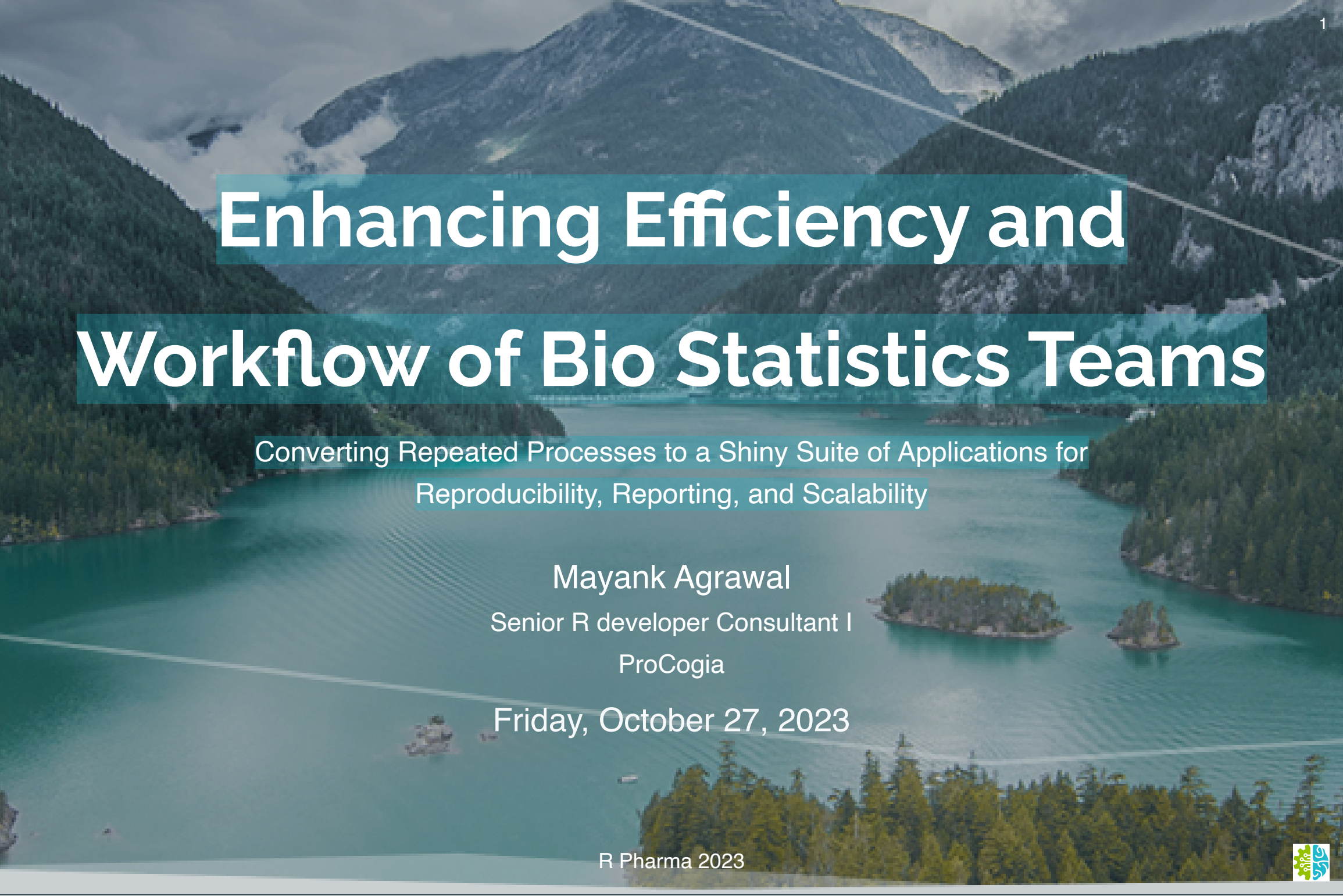

Enhancing Efficiency and Workflow of Bio Statistics Teams

Converting Repeated Processes to a Shiny Suite of Applications for Reproducibility, Reporting, and Scalability

Mayank Agrawal

Senior R developer Consultant I

ProCogia

Friday, October 27, 2023

Phase 2 Trails

> *A Phase 2 trial answers the question, "Does Drug X improve Disease Y?"*

- Phase 2 clinical trials assess the safety and efficacy of a new drug or drug combination for a specific medical condition.
- **Goal**: Determine appropriate dose and treatment plan for Phase 3 testing.
- **Phase 2a**: Involves fewer patients, generally 100-300 patients to focus on dose-response relationships and optimal dosing frequency.
- **Phase 2b**: Rigorously assesses drug's effectiveness in disease treatment, prevention, or diagnosis.
- Assess **therapeutic effectiveness** in a specific patient group for potential Phase 3 study.
- Also used to assess and review **safety parameters** for **potential adverse events** that might have been missed in a particular patient group.

Challenges

Dose Simulations for Phase 2 trails are often complex and time consuming with repeated similar workflow steps for each new variation of the dosage trail.

This often leads to

- Delayed Analysis and reporting
- Longer study time (years)
- Delayed Time to Market

resulting in

- Hindered workflows
- Lower productivity
- Repeated boring processes

Key Issues

- Manual processes
- Lengthy simulation times
- Scalability constraints
- Limited collaboration
- Reporting challenges
- Flexibility
- Reproducibility

Meeting the Challenges

Empowering Biostatisticians with discipline and the right set of tools.

- Collaborated with Biostats to understand their pain points and challenges.
- An operational framework rooted in **Agile development** methodology, with a focus on **empowering Biostats** at its core.
- Enhanced workflow efficiency, reproducibility and productivity among the biostats team.
  - Reproducible workflows with RStudio Projects, Git, `R`, `Shiny`, `renv` and `CRAN` packages.
  - Scalable applications built with Software Engineering principles.
  - Improved automated interactive reporting.
  - Automated testing framework for validated analysis results.
- The framework operated like a well-oiled machine, effectively engaging Biostats with discipline and empowering them with new tools in the expanding ecosystem.

Getting Started

- Define scope of each process.
- Document repetition rate, importance, and time investment for each work request.
- Identify the most time-consuming, yet simplest workflow.
- Develop a **MVP** (Minimum Viable Product)
  - Showcase a demo with the smallest workflow.
  - This aids leaders in visualizing the impact of approval.
- Integrate workflows incrementally from small to large.
- Continuously improve process through rapid iterations, responding to user feedback, ideas and feature requests.

Framework Principles

- Showcase early application design outlines using draw.io for UI layout prototyping.
- Invest time to establish a standard application template layout for the ecosystem with the organizational color scheme.
- Create smaller, independent Proof of Concepts (POCs) for new feature requests.
- Define the flow of reactivity for the overall application.
- Prioritize user-friendliness: if it's not intuitive, it won't be used.
- Prioritize user-requests based on need, impact, time, effort and complexity.
- Enable consistent reporting with parameterized Markdown/Quarto for dynamic MS Word reports, following organizational templates.
- Streamline workflow by adopting standard coding principles and agile project management.

Tools Selection

- `Shiny` for building interactive web apps straight from R.
  - Enhanced collaboration with Biostats in `R`; their preferred language for analysis and visualization of complex clinical trial data.
  - Seamless integration of Tables, Listings, and Graphs (TLGs).
  - Enabled real-time updates with quick release cycles, crucial for adapting to evolving trial needs.
  - Enhanced decision-making thru interactive tools.
- `renv` as a package management tool to ensure reproducibility across the team and environments.
  - `CRAN` published and maintained packages to ensure accurate and consistent results.

Enhancing Scalability

- Implement **modularization** and **functional programming** for a **plug-and-play** development format across applications.
- Enable multiple studies to be added concurrently using standard **git branching strategies**, involving multiple concurrent developers.
- **Async Programming**: Evaluate longer simulations in a **separate R process** preventing app performance issues.
- Take a step further and deploy simulation functions as internal APIs with **Plumber**.
- Write your **custom JavaScript and R bindings** for implementing unique feature requests.
- Approach feature requests as a blend of **web development**, **software engineering**, and **R development**.

Ensuring Reproducibility with Automated Testing

- Writing Test Cases (Inputs, Expected Outputs)
- Full Stack Testing
  - testthat for back-end testing,
  - shinytest2 for front end testing, and
  - shinyloadtest for load testing.
- Types of Tests: **Unit**, **Functional**, **Integration**, and **End-to-End**
- Continuous Integration for Testing with Git branching strategies.

- Benefits
  - Early bug detection
  - Efficiency and speed
  - Consistent and repeatable testing
  - Increased test coverage
  - Regression testing capabilities
  - Greater confidence in release stability

Enhancing Adoption

- Invest time to create an in-depth GitHub ReadMe providing comprehensive project reproducibility instructions.
- Include application workflow **GIFs** in announcement emails.
- Create detailed application interaction [**user manuals**{style="text-decoration:underline"} with screenshots and highlights for each step.
- Conduct regular (quarterly) **training sessions** to provide guidance, answer questions, and assist users with new features.
  - Record and share them for easier re-visit.
- Continuously engage user base for better ROI and on boarding.
- Prioritize most requested user features for each sprint.
- Have a team of Application Champions available for queries/requests.

Thank you

- Slides available on GitHub Pages at https://bit.ly/r-pharma-2023
- Quarto presentation code available on GitHub at https://bit.ly/github-r-pharma-2023
- Connect and/or send me a DM for a follow up question or catch up
  - Linkedin: mayank-agrawal-7jan
  - X (previously Twitter): mayank7jan
  - Mastodon: mayank7j

References - R Packages

- shinyDashboard, bslib, bs4dash for standard dashboard template.
- rmarkdown and Quarto for parameterized reporting.
- renv for package management in a R project.
- glue for interpreted string literals for dynamic reporting.
- Async programming: callr, mirai, crew, coro, future and promises.
- plumber for API creation.
- httr2 for API calls.
- pins for shareable secured publishing of data, models, and R objects
- testthat, shinytest2 and shinyloadtest for testing.
- dplyr for data manipulation.
- ggplot2, plotly and echarts4r for visualization.
- profvis for code profiling and time estimation.