

# GUIs for research software: Why are they relevant?

Dr Diego Alonso Álvarez  
Senior Research Software Engineer  
RCS-ICT, Imperial College London



**Imperial College**  
London

# Contents

1. Does my RS deserve a GUI?
2. Knowing your users
3. Pros and Cons of GUIs for RS



# A graphical user interface is about the user

If they do not like using it, then the GUI is either not needed or it is badly done

# Types of research software



Libraries



Frameworks



CLI Apps



Visual Studio Code



GUI Apps

# Types of research software: Libraries

- ▶ Toolboxes meant to be used in other software
- ▶ Often related to mathematical operations
- ▶ Or data manipulation
- ▶ Or accessing certain hardware
- ▶ Using them requires doing some coding



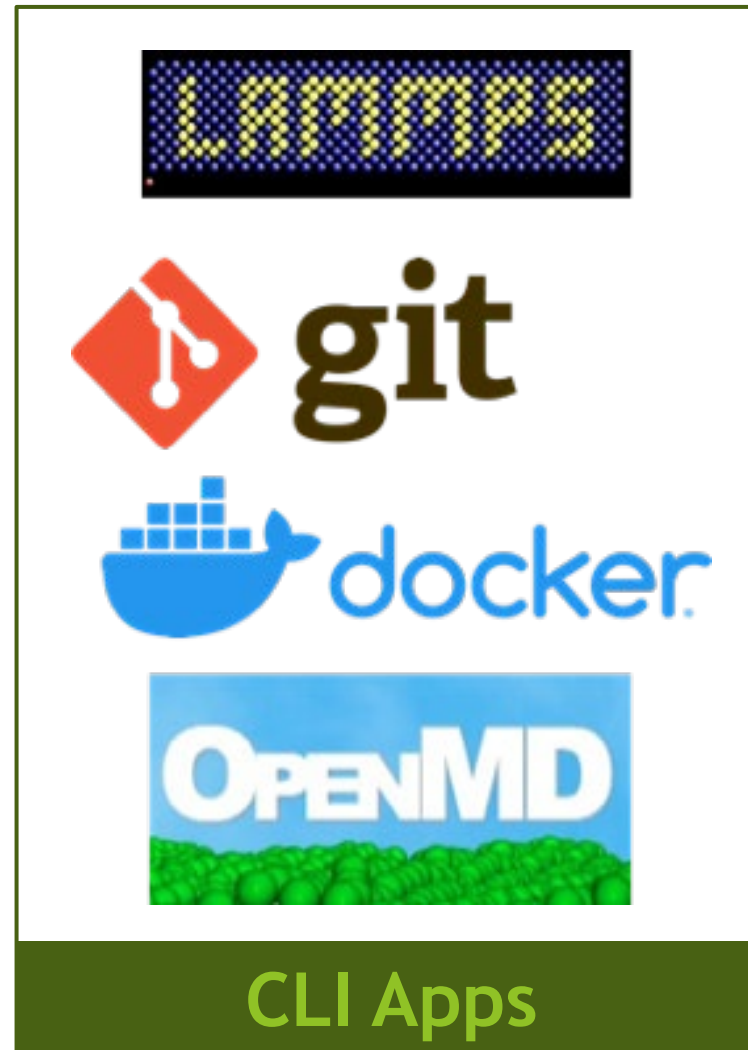
## Types of research software: Frameworks

- ▶ Tightly coupled with research in some area of knowledge
- ▶ Often bring to the table a variety of tools rather than a single functionality
- ▶ Include domain specific data in addition to code (universal constants, material properties, etc.)
- ▶ Using them requires doing some coding
- ▶ With an extra layer of code, parts of them could become apps



# Types of research software: CLI apps

- ▶ General purpose tools are often facilitators of research
- ▶ Domain specific tools are often HPC-oriented
- ▶ Do one thing only, although often with many arguments to modify the default behavior
- ▶ Do not require coding, but using the terminal



## Types of research software: GUI apps

- ▶ The GUI is the whole point of the application
- ▶ Most web apps fall into this category
- ▶ A wide range of people can often use them - at least, run them!





# Do they need a GUI?



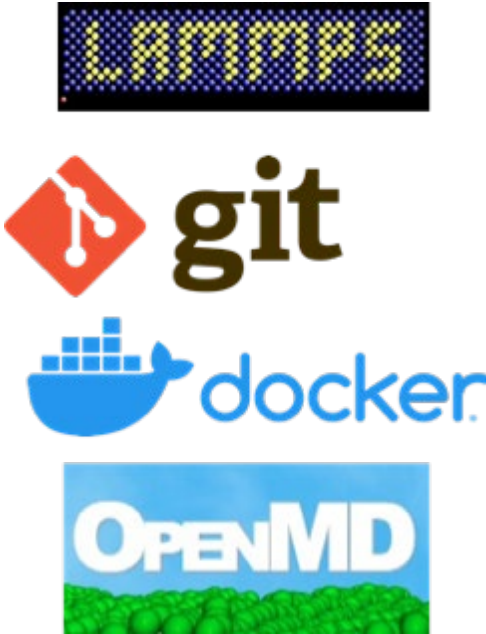
**sunpy**  
A Community Python  
Library for Solar Physics

The **Astropy**  
Project

**SOLCORE**

**PyBaMM**

**Frameworks**



**LAMMPS**

**git**

**docker**

**OPENMD**

**CLI Apps**

- ▶ Now that depends on the user's needs
- ▶ Having a GUI is not incompatible with other ways of using the software
- ▶ A GUI does not need to cover all functionality of the software

# One software: many ways of using it

Know your software to decide if a GUI  
makes sense at all

Know your users to decide if they will  
benefit from having a GUI



# Types of users

## **Anna, postdoc**

Likes to have things done. Needs publications to progress in her career and become a lecturer. Does not mind coding but is not one of her strengths. Prefer to spend time in the lab or outdoors doing field work than in front of her laptop.

## **Kristoff, postdoc**

Has 3 screens, a split keyboard and rarely uses the mouse. Hopes to finish his research in cybernetics and HCI on time for the RS Summer Science Exhibition.

## **Oaken, technician**

Is a perfectionist and is obsessed with efficiency. The less lab users need to do, the lower the chances they do something wrong. Thinks the software controlling the lab equipment is dreadful.

## **Elsa, Professor**

Manages a research group of 6 postdocs and a dozen students. She is writing 3 grant proposals and 2 review papers. She was a great coder but hasn't written a single line of code in 10 years. Need all results by yesterday.

## **Sven, postdoc**

Fond of open science and outreach. Should focus on his research, but enjoys too much showing what they do in his group at schools and public events. Does not know anything about coding but he is an excellent science communicator.

## **Olaf, PhD student**

Still figuring out the purpose of his PhD. Has lots of data to analyse in the HPC and has inherited some Fortran code from a former PhD to do it. Likes R... Fortran no so much. Knows he procrastinates way too much tuning his Linux workstation.



# Count with the user when creating something that is meant for the user

Ask them what they are trying to accomplish

- Users with different degrees of expertise
- Users from different fields
- Ask those who are not users yet, but that could become

Ask them what barriers/challenges are they facing when using the software

Ask about the GUI

- If there is already a GUI, ask for feedback about it
- If there is not, ask them to sketch how they imagine a potential GUI would be

# Generating user personas

User personas are semi-fictional characters based on current (or ideal) users

**Software:** A python framework to extract statistical data out of series of spectra taken from luminescent algae, among other things

User Persona	Who are they?	What is their main goal?	What is their main barrier or challenge when using the software?
Experimentalist postdoc	Postdoc working on a fix term contract in a big research group. Has some supervisory and teaching duties, as well.	Have the tons of data generated by the experiments analysed to get some statistical information and generate plots for their papers.	Cannot code. Often have to ask colleagues to write a script and run the tool for them with different options, but it is time consuming for everyone and problematic to transfer all that data from one computer to another.

<https://www.hotjar.com/blog/user-personas/>

<https://careerfoundry.com/en/blog/ux-design/how-to-define-a-user-persona/>

# Example: Git - Who uses what?

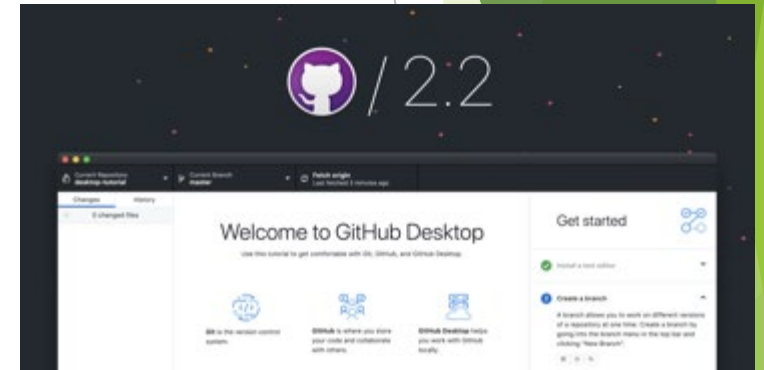
CLI app

Within IDEs

Standalone GUI app



Visual Studio Code





# GUIs for RS come with benefits... and a price to pay

Including a GUI is a key decision to be taken as soon as possible in the creation process. You need to plan for all that it involves, the good and the bad

A hand is shown interacting with a digital bar chart on a tablet screen. The chart consists of several white wireframe bars of varying heights. The background is a blurred blue and green, suggesting a professional or technological setting.

## The bright side of having a GUI

Accessibility

Usability

Reproducibility

Impact



# Accessibility

Separates user from developer

Accounts for users' needs

- Disabilities
- Language

No end-user knowledge of programming is required

- Non-coders
- Non-experts in that field
- Students/children

# Usability

## Learnability

- Reduces the learning curve and the time it takes to become proficient

## Efficiency

- Focus on research inputs/outputs not on technical details

## Memorability

- The visual dimension makes it easier to remember how to use the tool after a break

# Reproducibility

There is less scope for misusing functionality

- The user is offered only the options that can be used at that time

Can implement validation checks at user input

- Feedback and corrected values provided immediately

Input values explicit and visually accessible

# Impact

## Increases the user base

- Potential pool of new contributors
- More people providing feedback on usability and errors

## Increases citations

## Increases the visibility

- Researchers behind the tool
- RSE/RSE team developing it



# The dark side of having a GUI

Time

Complexity

New skills

Designer

# Time

## Translates to higher costs

- Development time
- Testing time
- Deployment

## Delays the delivery of the application

- To clients
- To showcase the app to stakeholders, investors or at a conference

# Complexity

## More code

- To provide tests for
- That can have bugs that need to be sorted out
- That depends on extra libraries with their own issues

## New architecture

- To maintain the separation of concerns
- That support multi-threaded execution
- To accommodate user interaction

# New Skills

## Coding skills

- New GUI toolkits/frameworks
- Separate programming languages

## Software architecture and techniques

- Gain knowledge on GUI-related software architectures
- Testing techniques for GUIs
- New development paradigms, like event-driven programming



# Designer

## Know the psychology of users

- Appropriate layout to maximise efficiency
- Familiar with accessibility issues of different types of users

## Good “taste” for creating a pleasant look&feel

- Deal with proportions and distribution of widgets
- Appropriate combination of palettes of colors and fonts
- Might be able to create custom logos and icons

## Not all software needs a GUI

- Having a GUI does not exclude other ways of using the software
- Only some parts of the software might benefit from a GUI

## Know your users

- Those who already are and those who could become one
- Count with them when deciding if including a GUI and what to include in it

## Pros and Cons of GUIs for RS

- A GUI brings benefits to the software at multiple levels, specially in terms of usability
- But it is a big undertaking, requiring time, effort and new skills

# Take away points