Reproducible Research Recipes

Collaborative Idea team members

Jennifer Molloy, Robert Haines, Adam McMaster, Fabian Renn, Robin Wilson

Hackday pitch leader

Robin Wilson

Context

Tool to support computational research across all domains

Problem

"One small change in your input data, one giant screw-up in your PhD viva"

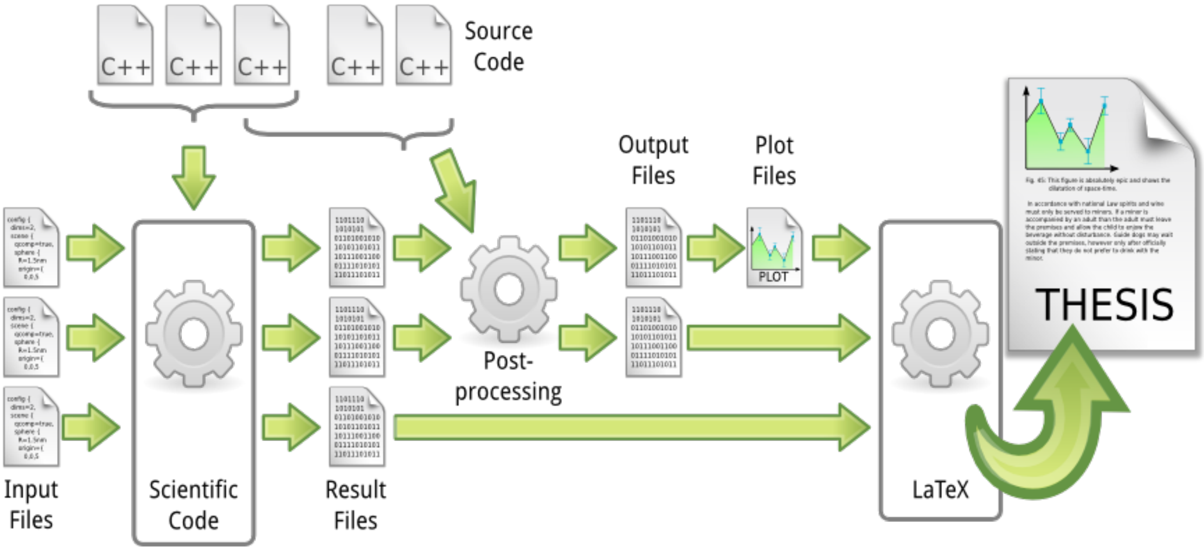
As scientists we often run into problems when we have multiple data sources, each of which undergoes multiple processing steps before final output such as plots, tables or even entire theses are produced. Small changes in input files or code can lead to large changes in these final outputs, but it is often difficult to spot this.
Putting your code together into a reproducible pipeline can solve this - particularly when dependency management is included which ensures that the right processes are re-run when bits of code or data change.
There are methods to solve this problem (such as make), but these generally come with a significant learning curve, and creating these sort of pipelines is very difficult - particularly for inexperienced users.

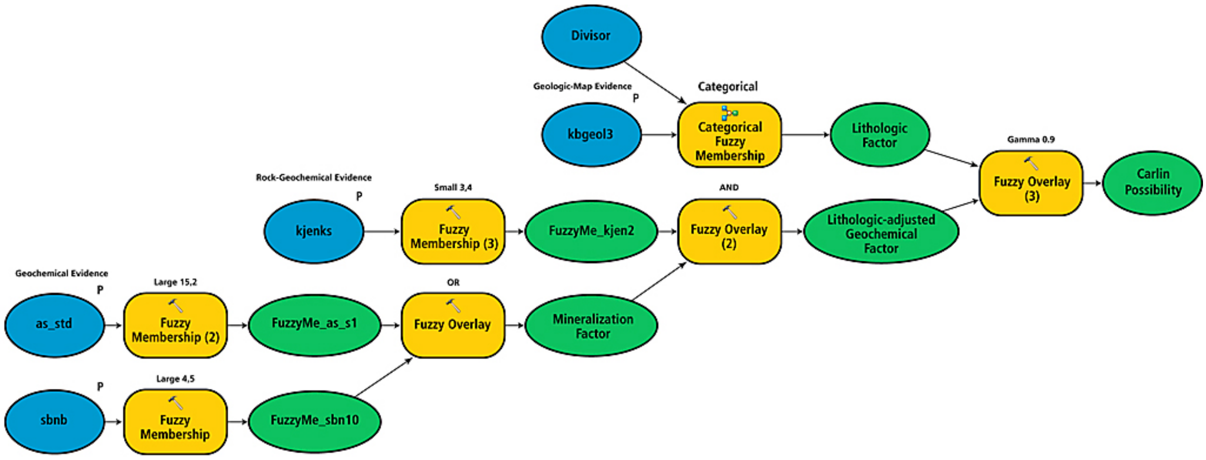
Solution

A simple graphical tool to sketch out dependencies between different tasks and then produce an executable pipeline. For example, this could be a GUI tool which allows users to drag and drop individual tasks and data into a pipeline, connect them to show the dependencies and then produces something like a Makefile which can then be run easily. The tool could also support generic rules to, for example, run a certain command to convert all .dat files to .png files by plotting them in R.

We realise there are pipeline tools currently available, but they're often very complex and become a programming environment of their own. With this tool we want to produce a really simple 'lightweight' interface for co-ordinating the tools you already use, rather than adding something else to learn!
A quick investigation shows that a tool like this does not currently exist.

Diagrams





Extensible Github views

Collaborative Idea team members

Jens Nielsen, Michael O'Hagan, Jane Charslesworth, Dominic Orchard, Ross Mounce

Hackday pitch leader

Jens Nielsen

Context

Github is great for sharing code and data. A further useful feature is the ability to view files in the browser and compare versions (“diffs”). Visualisations are extremely useful for communicating data.

Problem

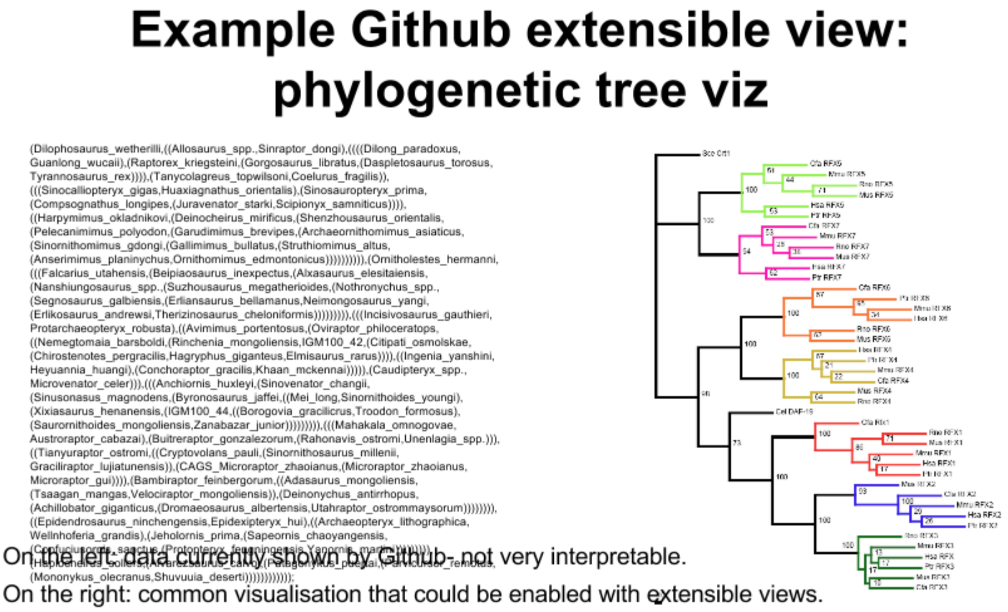
The supported file formats for Github ‘views’ is limited to a small number of (arbitrary) formats, including text files, images, CSV, GeoJSON, STL. It would be even better if researchers could render, and compare versions, for other common scientific formats within Github repositories, for example:

- phylogenetic trees (e.g. nexus/newick and see separate illustration)
- Matplotlib
- gnuplot
- protein structures (e.g. PDB files)
- NMR spectra (e.g. raw FID files)

But none of these are supported.

Solution
We propose extending github with a system for extensible user-defined ‘views’. This will allow scientists to develop views for their projects that can be shared within the community. Since the Github.com website is itself not open-source, the development process will need to first start on the ‘Gitlab’ clone (providing a roadmap for adoption into Github.com). The extension will need to take a raw file (of some format) as input, and produce HTML (+CSS/Javascript) as output (ideally without any system calls, for security reasons). To address safety concerns we would suggest using Github’s current limitations regarding e.g. file sizes as a starting point

Diagrams



Example Github extensible view: phylogenetic tree viz

(Dilophosaurus_wetherilli,((Allosaurus_spp.,Sinraptor_dongi),((((Dilong_paradoxus, Guanlong_wucaii),(Raptorex_kriegsteini,(Gorgosaurus_libratus,(Daspletosaurus_torosus, Tyrannosaurus_rex)))),(Tanycolagreus_topwilsoni,Coelurus_fragilis)), (((Sinocalliopteryx_gigas,Huaxiagnathus_orientalis),(Sinosauropteryx_prima, (Compsognathus_longipes,(Juravenator_starki,Scipionyx_samniticus)))), ((Harpymimus_okladnikovi,(Deinocheirus_mirificus,(Shenzhousaurus_orientalis, (Pelecanimimus_polyodon,(Garudimimus_brevipes,(Archaeornithomimus_asiaticus, (Sinornithomimus_gdongi,(Gallimimus_bullatus,(Struthiomimus_altus, (Anserimimus_planinychus,Ornithomimus_edmontonicus)))))))))),(Ornitholestes_hermanni, (((Falcarius_utahensis,(Beipiaosaurus_inexpectus,(Alxasaurus_elesitaiensis, (Nanshiungosaurus_spp.,(Suzhousaurus_megatherioides,(Nothronychus_spp., (Segnosaurus_galbiensis,(Erliansaurus_bellamanus,Neimongosaurus_yangi, (Erlikosaurus_andrewsi,Therizinosaurus_cheloniformis))))))))),(((Incisivosaurus_gauthieri, Protarchaeopteryx_robusta),((Avimimus_portentosus,(Oviraptor_philoceratops, ((Nemegtomaia_barsboldi,(Rinchenia_mongoliensis,IGM100_42,(Citipati_osmolskae, (Chirostenotes_pergracilis,Hagryphus_giganteus,Elmisaurus_rarus)))),((Ingenia_yanshini, Heyuannia_huangi),(Conchoraptor_gracilis,Khaan_mckennai))))),(Caudipteryx_spp., Microvenator_celer))),(((Anchiornis_huxleyi,(Sinovenator_changii, (Sinusonasus_magnodens,(Byronosaurus_jaffei,((Mei_long,Sinornithoides_youngi), (Xixiasaurus_henanensis,(IGM100_44,((Borogovia_gracilicrus,Troodon_formosus), (Saurornithoides_mongoliensis,Zanabazar_junior)))))))))),(((Mahakala_omnogovae, Austroraptor_cabazai),(Buitreraptor_gonzalezorum,(Rahonavis_ostromi,Unenlagia_spp.))), ((Tianyuraptor_ostromi,((Cryptovolans_pauli,(Sinornithosaurus_millenii, Graciliraptor_lujiatunensis)),(CAGS_Microraptor_zhaoianus,(Microraptor_zhaoianus, Microraptor_gui)))),(Bambiraptor_feinbergorum,((Adasaurus_mongoliensis, (Tsaagan_mangas,Velociraptor_mongoliensis)),(Deinonychus_antirrhopus, (Achillobator_giganticus,(Dromaeosaurus_albertensis,Utahraptor_ostrommaysorum)))))))), ((Epidendrosaurus_ninchengensis,Epidexipteryx_hui),((Archaeopteryx_lithographica, Wellnhoferia_grandis),(Jeholornis_prima,(Sapeornis_chaoyangensis, (Confuciusornis_sanctus,(Protopteryx_fengningensis,Yanornis_martini)))))))), (Haplocheirus_sollers,(Alvarezsaurus_calvoi,(Patagonykus_puertai,(Parvicursor_remotus, (Mononykus_olecranus,Shuvuuia_deserti))))))))))));

On the left: data currently shown by Github- not very interpretable.
On the right: common visualisation that could be enabled with extensible views.

Phylogeny of Scientific Codes

Collaborative Idea team members

Devasena Inupakutika, Laurent Gatto, Philip Fowler, Katalin Phimister, Daniele Tartarini

Hackday pitch leader

Laurent Gatto

Context

Scientific codes vary tremendously, even within a discipline. For example there are varying degrees of documentation and utility of error messages. In extreme cases some codes resemble black boxes.

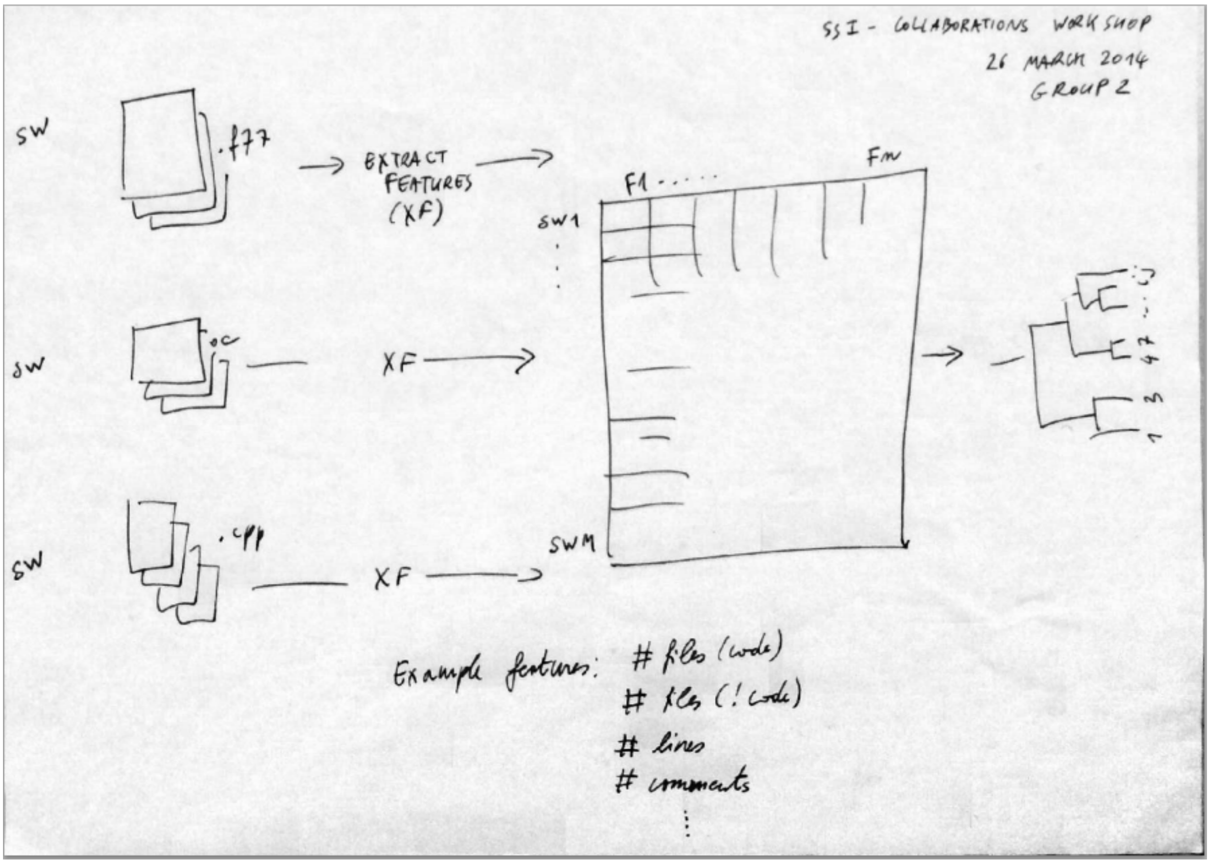
Problem

It is very hard to know which codes are "better" (or in a related way, which codes are more similar). Mind you, this isn't intended *entirely* seriously - but it would be fun to see if we could come up with a phylogenetic tree of the scientific codes from a particular scientific discipline, such as quantum chemistry or classical molecular dynamics or R packages. Having said that, we hope that the results would spur developers to improve their code, by e.g. providing comments for each method in a class etc.

Solution

- retrieve a set of scientific codes to study
- design and implement an algorithm to classify and compare the source code in plaintext. This might look at the amount of documentation included with the package, the proportion of the code that is documented, the number of variables that match dictionary entries, a distribution of words in the code ("for", "if" etc) and symbols ("{}, ";" etc).
- use to compare different packages in an, hopefully, automated way
- draw a phylogenetic tree and see what we can learn! (Hopefully we could infer which codes are of higher quality).
- publish it somewhere (like PNAS)

Diagrams



Crowd-sourced image annotation web site

Collaborative Idea team members
Jan Kim, Mike Jackson, Graham Etherington, Michael Fischer, Robyn Grant

Context
Any with image processing e.g. bioinformatics, medical etc, in which images require manual intervention.

Problem
Images require hand annotating via point-and-click using the mouse e.g. to mark up whiskers on rats, or tweak marked areas on the brain.
The number of images available (100s-1000s) means this is time-consuming and tedious.

Solution
Crowd-source - have volunteers do the annotation!
Support competitive motivators e.g.:

- Badging for numbers of images done.
- League tables of numbers done.

Promote collaborative motivators e.g.: Helping science.
Support manual review by the image set owner to assess the quality of the annotations.
Future: Evaluate multiple attempts on same images for goodness-of-fit to help devise more

intelligent automated methods in future.

Readmycode: Code review buddy finder

Collaborative Idea team members
James Hetherington, Leanne Wake, Nicolas Gruel, Jonathan Cooper, Sweitze Roffel

Hackday pitch leader
James Hetherington

Context
A: I'm a researcher who programs. I know my code could be better, but I don't know how to start finding out what I could improve and how. I want to find people to show my code to.
B: I run a code journal. I want to find programming researchers who are qualified to review submissions.

Problem
Given:

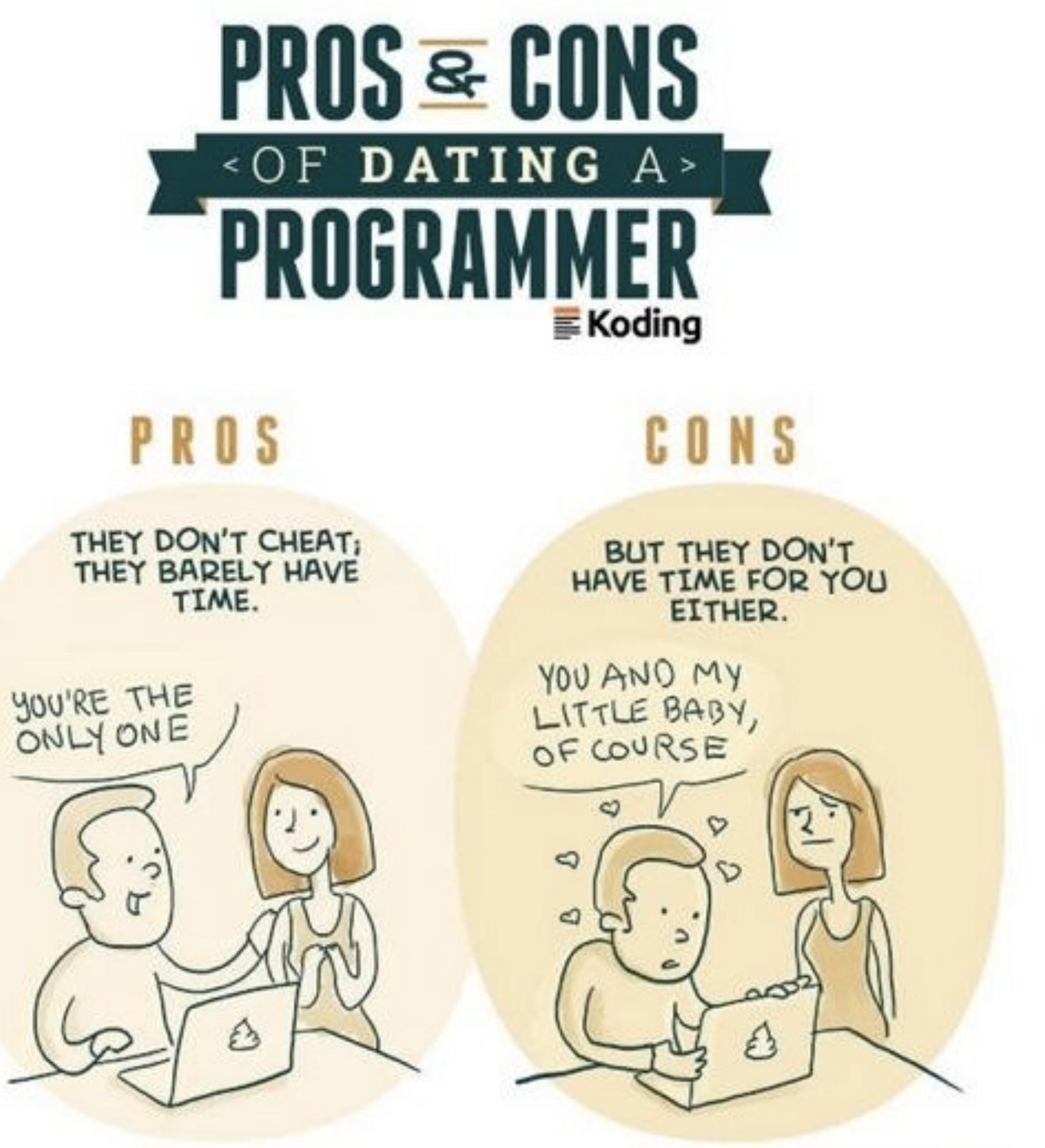
- A link to my code
- A description of me as a researcher

Find:
Other researchers who work

- in similar fields
- OR with similar techniques
- AND in the same programming language

Solution
From:
A git/hg repo or github URL
An ORCID or userid on academia.edu or Web of Science or similar
Use existing online searches and data to match and search.

Diagrams


PROS & CONS
OF DATING A
PROGRAMMER
Koding
PROS
CONS
THEY DON'T CHEAT; THEY BARELY HAVE TIME.
YOU'RE THE ONLY ONE
BUT THEY DON'T HAVE TIME FOR YOU EITHER.
YOU AND MY LITTLE BABY, OF COURSE

Digitalisation of hand-drawn chemical structures

Collaborative Idea team members
Paul Barrett, Derek Groen, Andreas Heger, Rob Davey

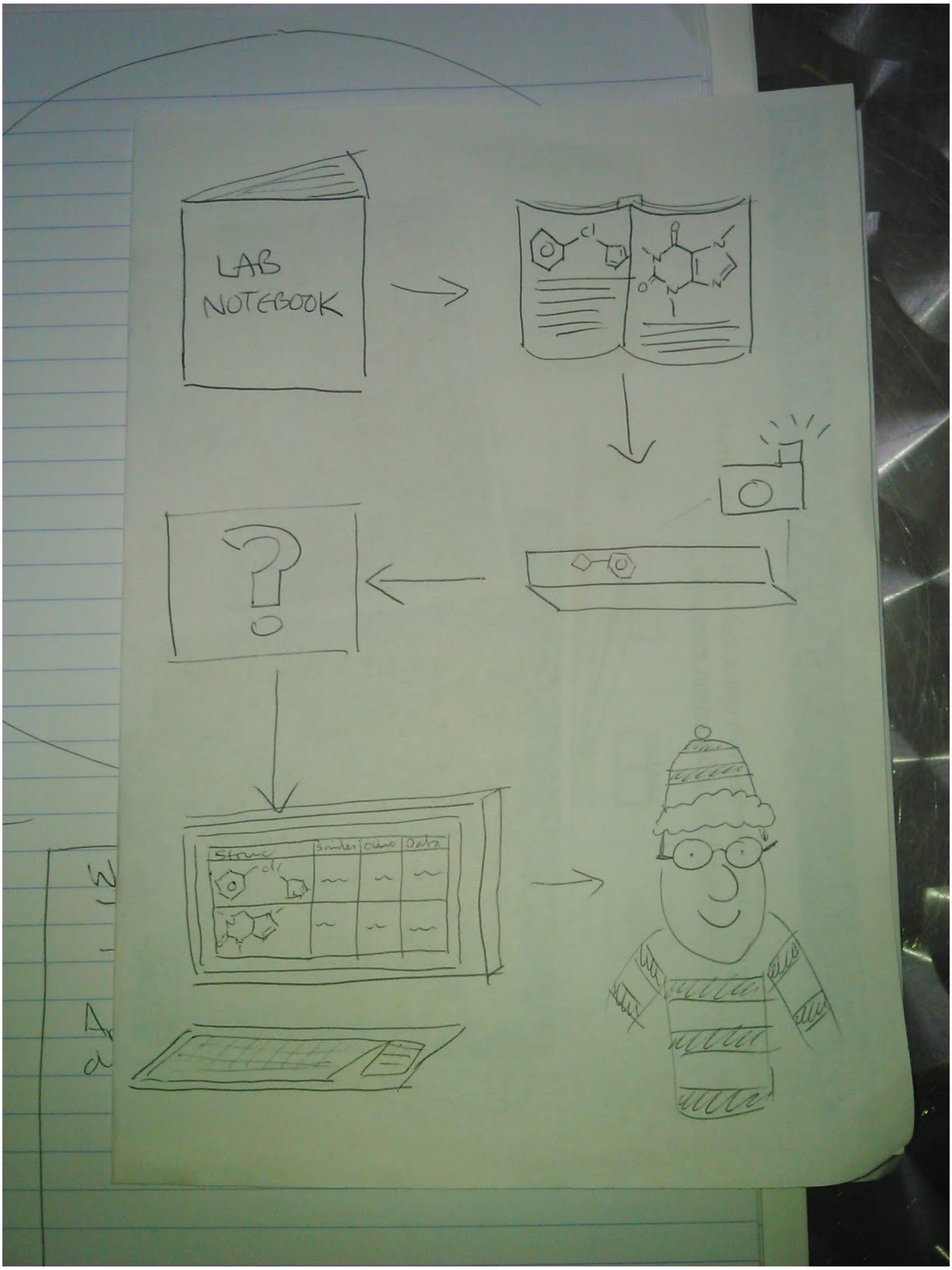
Hackday pitch leader
Derek Groen

Context
Much chemical data is represented as structure diagrams. Many of these are hand-drawn figures in paper lab notebooks. We envisage moving legacy chemistry lab notebooks to a new digital system would benefit from automated scanning, recognition and digitalisation of these structure doodles.

Problem
Scientists of different disciplines have a need for image scanning and recognition for a variety of purposes. One such example is legacy hand-drawn chemical structures in paper lab notebooks that hinder reproducibility and understanding of the research.

Solution
Taking an automated "Where's Wally" solution code as an example and abstract it out to recognise chemical structure drawings in scanned pages and produce digital versions, i.e. SMILES string, and potentially PDB format files and 3D representations.

Diagrams



Open Source Health Check

Collaborative Idea team members

Arfon Smith, Kywei Duan, James Spencer, Mark Basham

Context

Open source software?

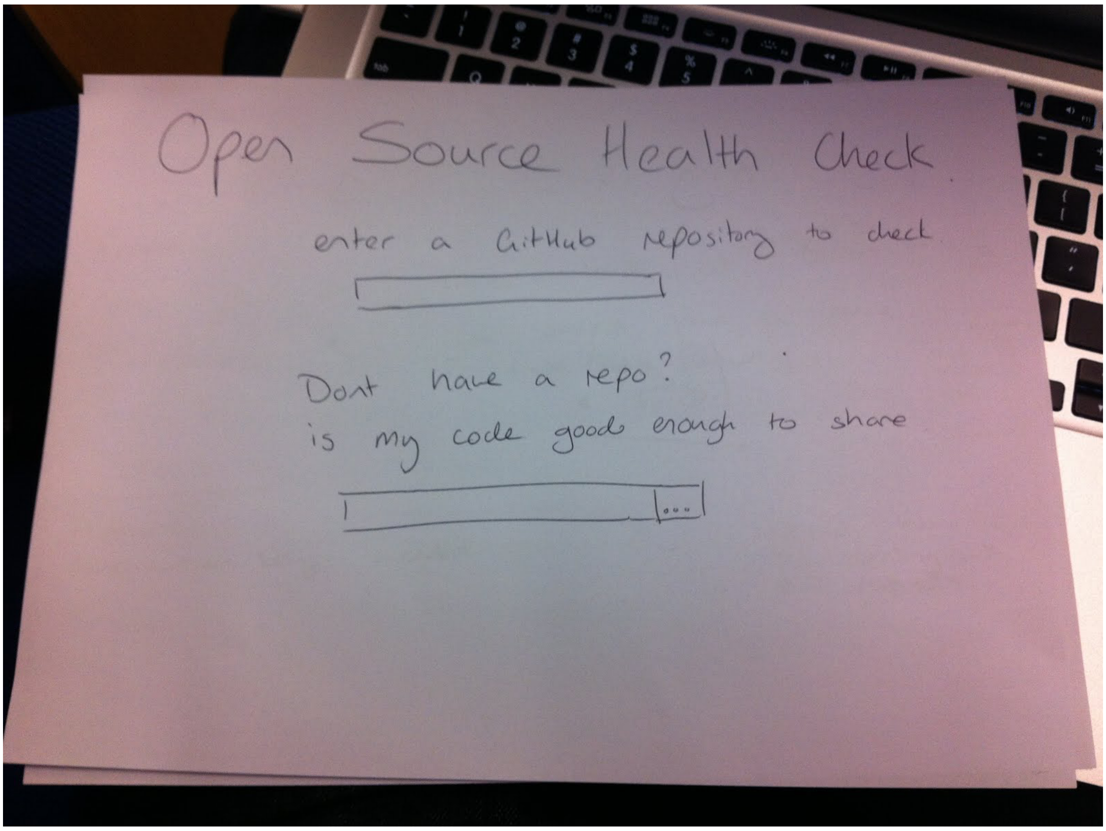
Problem

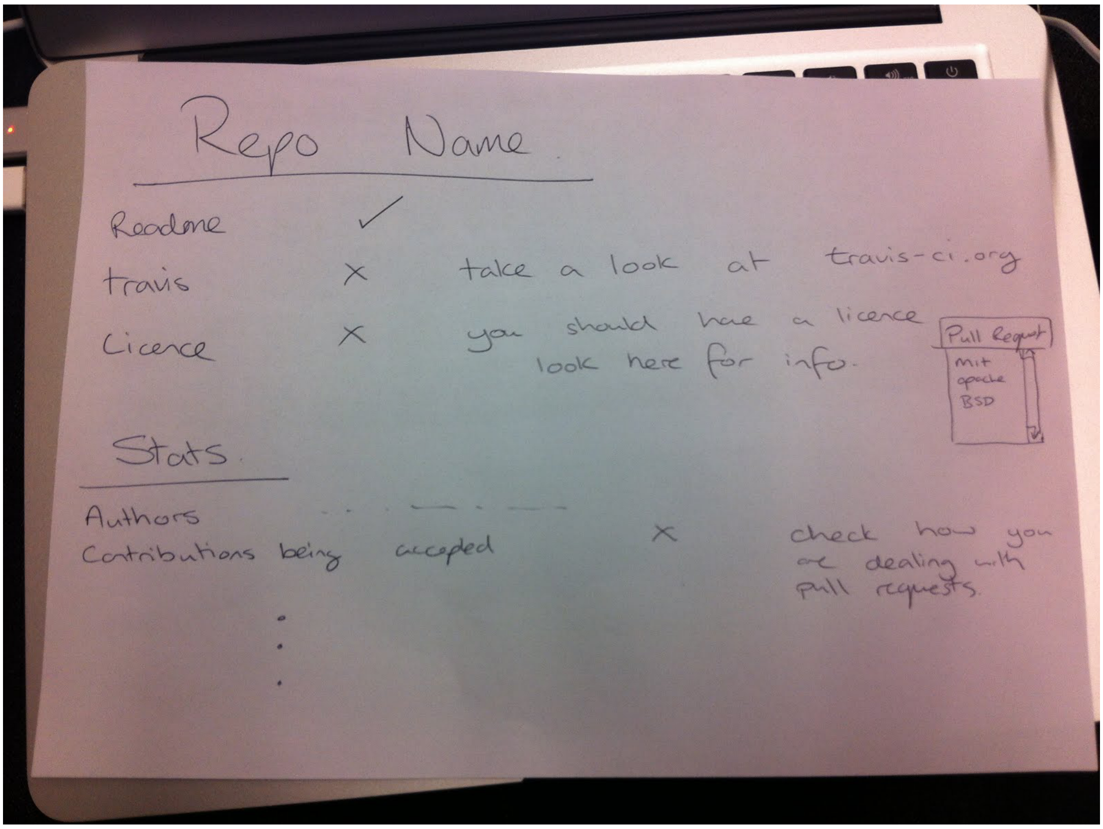
What steps do I need to take to make my open source project more shareable?
For example: Is there a licence? Is there a README? Is there service running automated tests? Are community contributions ever accepted?

Solution

We propose creating a website (or automated tool) that looks at an open source project and checks for key files/signatures such as a licence.txt, README.md, .travis.yml file that denote repository health.
If these files are missing then the Open Source Health Check Bot (OSHCB) opens a pull request on GitHub suggesting modifications necessary to improve the 'health' of the repository.
A possible extension to this is a simple tool that makes an assessment of a code repository that helps researchers with the question 'is my code good enough to share'. It always says YES.

Diagrams


Open Source Health Check
enter a GitHub repository to check
Dont have a repo?
is my code good enough to share
...



Repo Name

Readme ✓

travis ✗ take a look at travis-ci.org

Licence ✗ you should hae a licence look here for info.

Pull Request
mit
apache
BSD

Stats

Authors

Contributions being accepted ✗ check how you ae dealing with pull requests.

⋮

Peer review of software reproducibility

Collaborative Idea team members
Stephen Eglen, Jure Triglav, Neil Chue Hong, Grigori Fursin, Graham Klyne

Hackday pitch leader
Jure Triglav

Context
Published results based on software that others cannot successfully install and run cannot really be considered to be reproducible. But, full peer review of software is unrealistic to achieve, so are looking for an accessible proxy for this.

Problem
How to get a software description (e.g. a README?) that is good enough for someone else to install and reproduce the claimed results with given input.
Requirements:

- mechanisms to get to the right reviewers;
- testing process: needs to be constrained?
- a way to report back - feedback into product instructions (pull request?);
- mechanisms to credit reviewers (author credit on paper).

Solution
Potential solution:

- adjunct(s) to GitHub infrastructure?

- reporting though issues and pull requests
- ideally, feedback crystallised in automatic/executable configuration files

Matlab Toolbox Metadata

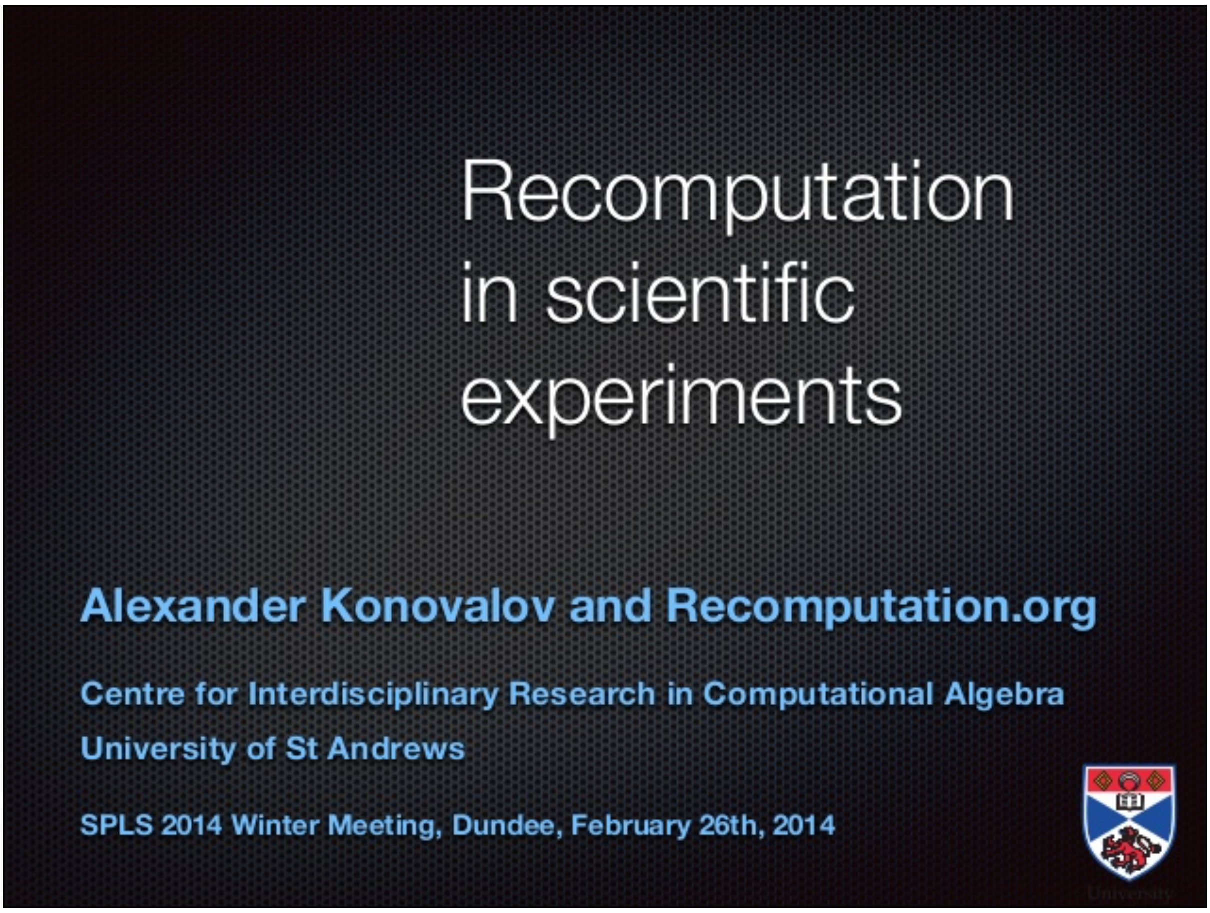
Collaborative Idea team members
Tim Parkinson, Liberty Foreman, Michael Fischer, Bruno Vieira, Marta Ribeiro
Context
Many users in a lab use Matlab scripts to analyse data but they may all use different versions of toolboxes from a range of commercial and in-house developers.
Problem
The different versions of the toolboxes may have different parameters or else make different assumptions as to default values and therefore may produce different results on different machines (even if those machines are identical hardware and OS).
(Same problem for Ruby and Ruby Gems).
Solution
a) find a way to make a Matlab script dump any version info for the toolboxes it is using.
b) set up some sort of register (website) for the toolboxes in use that describe the differentces between the toolboxes and between versions of each toolbox.

Diagrams

New recomputation.org experiment

Collaborative Idea team members
Ling Ge, Karen Porter, Filippo Mortari, Olexandr Konovalov, Ahmad Alam
Hackday pitch leader
Olexandr Konovalov
Context
Recomputable scientific experiments.
Problem
We suggest to pick up a computational experiment and make it reproducible. That could be an experiment from the area of expertise of team member(s), either from their own experience or from some (recent or classical) computational results in their field. For example, one could find a paper in which an experiment and software/data are described and start from that.
Solution
Make a virtual machine either in VirtualBox to run locally or (preferably) in Azure cloud. Configure it to run the experiment automatically after booting up. Try to address the issues of user friendliness and discovery.

Diagrams



Recomputation in scientific experiments

Alexander Konovalov and Recomputation.org

Centre for Interdisciplinary Research in Computational Algebra

University of St Andrews

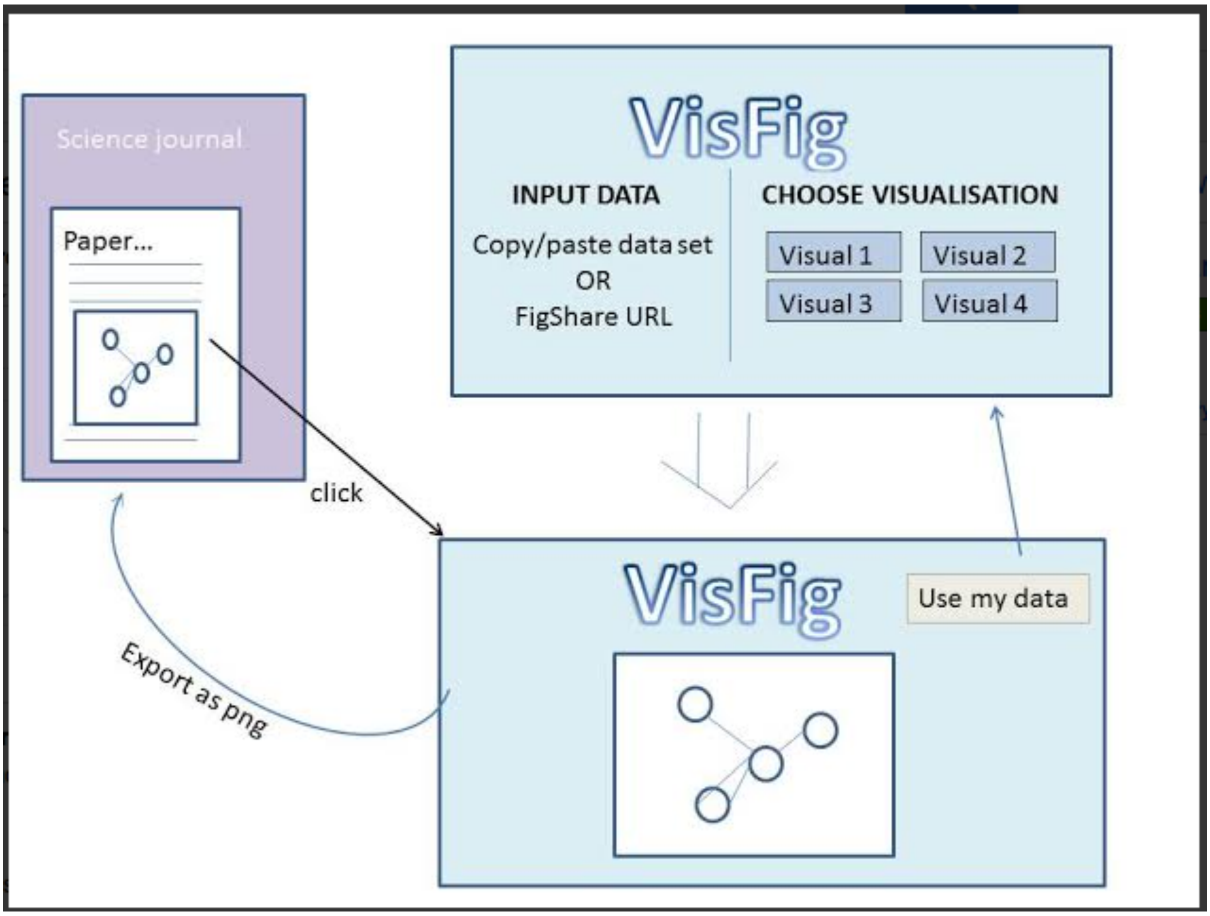
SPLS 2014 Winter Meeting, Dundee, February 26th, 2014

Dynamic figures to follow-up findings instantly

Collaborative Idea team members
Matthew Brack, Martin Hammitzsch, Manuel Corpas, Niall Beard
Hackday pitch leader
Niall Beard
Context
Make software creating figures available as service persistently.
Problem
Reproduce and replicate a figure from a paper with data used in the paper and 'my data' to follow-up findings presented in the paper.
Solution

- Build the figure on a website with open libraries and APIs, e.g. using Google chart API
- Preconfigure the dynamic figure using the paper data
- Mint a DOI to the website to enable the 'social process' / commitment behind the DOI universe
- Put link/DOI to website beneath figure in paper
- Link/DOI directs to website, preconfigured with data and display type and displays figure form paper
- Use button to change the data for same figure displaying other data

Diagram



Reproduciliteracy: Reculturing Research

Collaborative Idea team members
Alexandra Simperler, Ian Gent, Kenji Takeda, Sebastian Gibb, Tom Crick, Stephen Crouch

Hackday pitch leader
Ian Gent

Context
This is a problem that transcends education and research, from undergrads to postgraduate; people who aren't at the CW14 need to be convinced to spend time and money to learn about the importance, skills and tools associated with reproducibility in research.

Problem
As Carole Goble mentioned in her talk at the CW14, there is a multitude of manifestos and pledges, declarations for research software around recomputation, reproducibility, repeatability. It's easy to sign a manifesto or pledge, but how can we provide a cohesive, over-arching support that effects culture change. The other issue is that many scientists aren't good at selling ideas - how can we help them to sell reproducibility? How can this be sold to researchers as a good idea that will benefit them?

A lot of materials on these topics already exist (e.g. in Software Carpentry, others), but bringing them together in a meaningful way could make this easier for others to understand.

Solution
We can provide people with materials (slides, guides, documents) to help them understand and sell reproducibility to others within their own contact networks. We could use the hackday time to collate/develop such materials.

As a starting point, we could create a small set of support materials for SSI Fellows. This could include a slide for talks, a behavioural case study which illustrates the point and benefits of it.

Diagram