eLIFE

Publishing reproducible research

Emmy Tsang, Innovation Community Manager @ eLife
Twitter: @eLifeInnovation | Labs: elifesci.org/labs

hhmi Howard Hughes Medical Institute

wellcome

MAX-PLANCK-GESELLSCHAFT

Knut and Alice Wallenberg Foundation

What is eLife?

eLife as an experiment

- What happens if we open up peer review?
- How can we make reading papers easier?
- Can we simplify production with better software?
- Can we help early career researchers through our platform?

hhmi Howard Hughes Medical Institute

wellcome

MAX-PLANCK-GESELLSCHAFT

Knut and Alice Wallenberg Foundation

eLI

Helping scientists accelerate discovery by operating a platform for research communication that encourages and recognises the most responsible behaviours in science.

eLife Innovation’s mission

Drive open-source innovation for open science, through:

- Building open-source tools, platforms and technologies to improve the ways research is discovered, consumed, shared and evaluated

- Supporting a community of open-source innovators to develop these tools

What is reproducible research?

Reproducibility: a definition

		Data	
		Same	Different
Analysis	Same	Reproducible	Replicable
	Different	Robust	Generalisable

The Turing Way, Chapter 2: Reperoducibility https://the-turing-way.netlify.com/reproducibility/03/definitions.html

eLIFE

Why should you care?

Some code from my PhD...

```
#goi avg_exp heatmaps
avg_exp=FetchData(dataset, c("cluster", goi))
a=melt(avg_exp, id.vars="cluster", variable.name="goi", value.name="exp")
avg_exp=a %>% group_by(cluster, goi) %>% summarise(avg=mean(exp))
avg_exp_goi=avg_exp %>% group_by(goi) %>% summarise(range=max(avg)-min(avg), min=min(avg))
avg_exp=dplyr::left_join(avg_exp, avg_exp_goi, by="goi")
avg_exp=dplyr::mutate(avg_exp, norm_exp=(avg-min)/range)
ggplot(avg_exp, aes(x=cluster, y=goi,fill=norm_exp)) +
  geom_tile() +
  theme(axis.text.x=element_text(angle=90, hjust=1, vjust=0.5))

#DEx
cluster_perm=data.frame(permutations(n=length(levels(dataset@meta.data$cluster)), r=2, v=levels(dc
colnames(cluster_perm)=c("clust1", "clust2")
tests=c("wilcox", "t", "bimod", "MAST")
pairwise_clust=data.frame(goi=c(), clust1=c(), clust2=c(), test=c(), p_val=c())
for (i in 1:nrow(cluster_perm)) {
  for (j in tests) {
    pval=FindMarkers(dataset, ident.1=cluster_perm[i, 'clust1'], ident.2=cluster_perm[i, 'clust2']
    pval$goi=rownames(pval)
```

Why work reproducibly?

- Avoiding disaster
- Writing papers easier
- Convincing reviewers
- Facilitating continuity of work
- Building your reputation

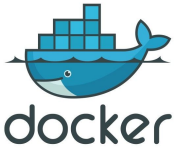
The Turing Way, Chapter 2: Reperoducibility https://the-turing-way.netlify.com/reproducibility/02/whycare.html

How to work reproducibly?

Document your code

Version control

Reproducible environments



Reproducible workflows

DATA CARPENTRY

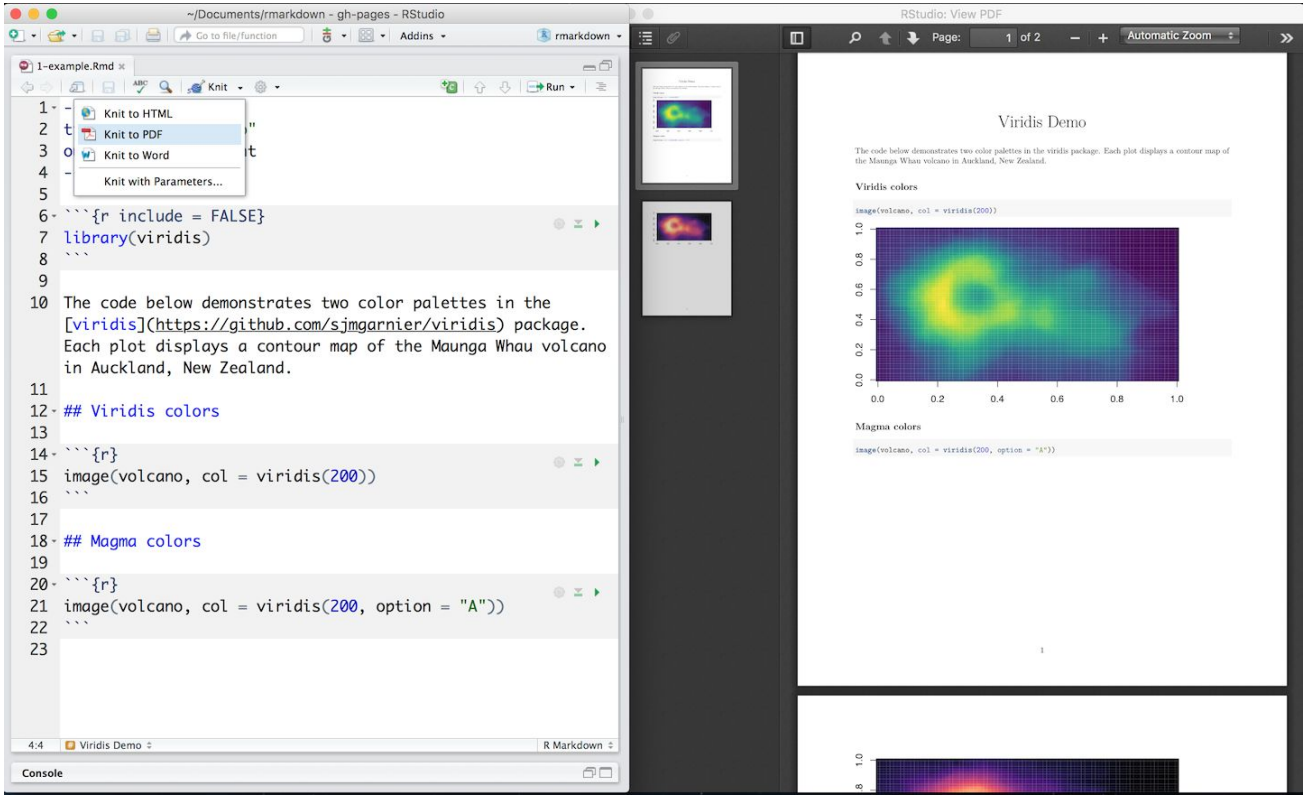
BUILDING COMMUNITIES TEACHING UNIVERSAL DATA LITERACY

The Turing Way: https://the-turing-way.netlify.com/

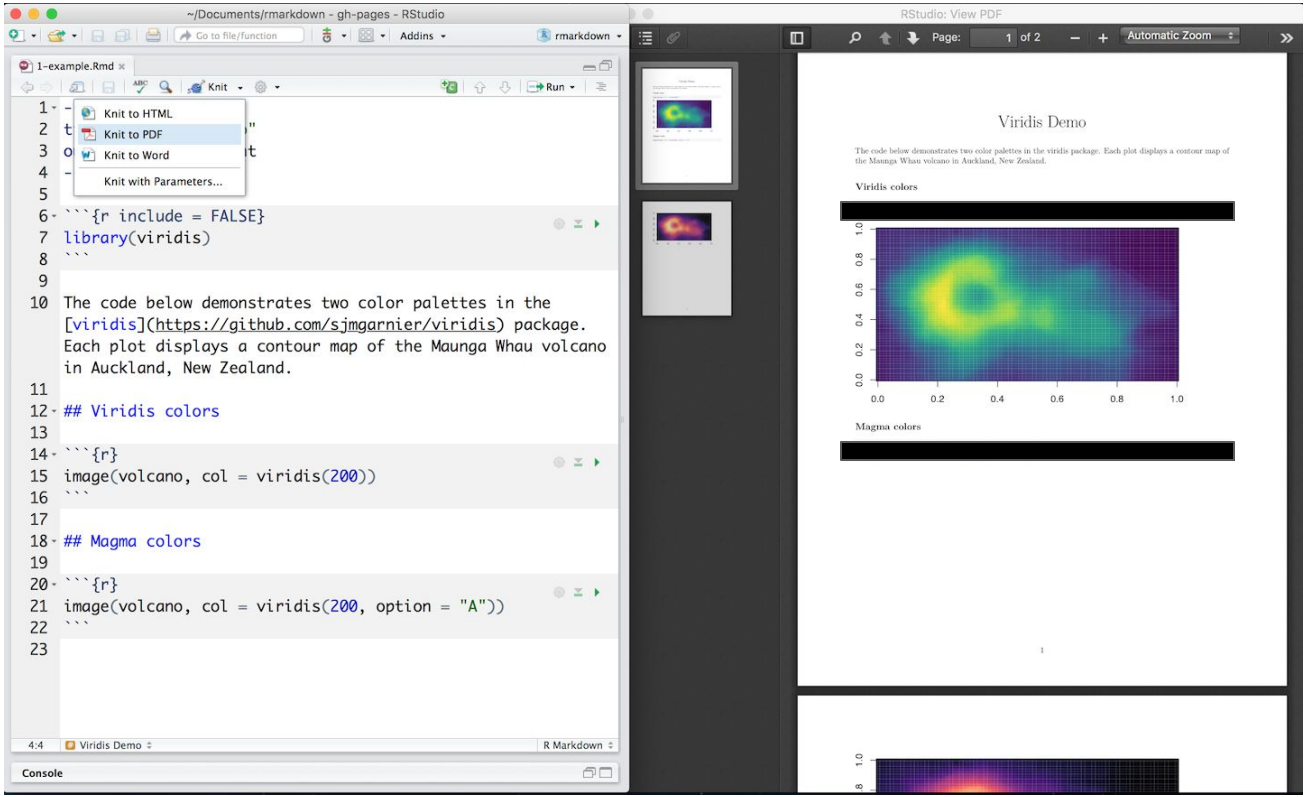
eLIFE

How can we make research more reusable and reproducible through publishing?

Is this you?


~/Documents/rmarkdown - gh-pages - RStudio
1-example.Rmd
Knit to HTML
Knit to PDF
Knit to Word
Knit with Parameters...
```{r include = FALSE}
library(viridis)
```
The code below demonstrates two color palettes in the [viridis](https://github.com/sjmgarnier/viridis) package. Each plot displays a contour map of the Maunga Whau volcano in Auckland, New Zealand.
Viridis colors
```{r}
image(volcano, col = viridis(200))
```
Magma colors
```{r}
image(volcano, col = viridis(200, option = "A"))
```
Viridis Demo
R Markdown
Console
RStudio: View PDF
Page: 1 of 2
Automatic Zoom
Viridis Demo
Viridis colors
Magma colors

Is this you?


~/Documents/rmarkdown - gh-pages - RStudio
1-example.Rmd
Knit to HTML
Knit to PDF
Knit to Word
Knit with Parameters...
```{r include = FALSE}
library(viridis)
```
The code below demonstrates two color palettes in the [viridis](https://github.com/sjmgarnier/viridis) package. Each plot displays a contour map of the Maunga Whau volcano in Auckland, New Zealand.
Viridis colors
```{r}
image(volcano, col = viridis(200))
```
Magma colors
```{r}
image(volcano, col = viridis(200, option = "A"))
```
Viridis Demo
R Markdown
Console
RStudio: View PDF
Viridis Demo
Viridis colors
Magma colors

Our vision: Reproducible Documents

- Encapsulates usable code and data within the flow of a manuscript.
- Delivers progressive enhancement from static research article, to full data and code interaction
- Future-proof: Platform, tool, language agnostic
- Accessible: Easy and accessible for everyone
- Encourage reuse of published research

Reproducible Document Stack

Demo: elifesci.org/reproducible-example

increase in expression upon c-Myc induction, in contrast to genes that were silent under low c-Myc conditions that did not change.

The Registered Report for the 2012 paper by Lin et al. described the experiments to be replicated (Figure 1B and 3E–F), and summarized the current evidence for these findings (Blum et al., 2015). Since that publication there have been additional studies investigating the ability c-Myc to influence the global gene expression output of cells. Similar to Lin et al. other studies have reported c-Myc dependent amplification of cellular RNA (Hart et al., 20152014; Hsu et al., 2015; Nie et al., 2012; Sabò et al., 2014), although this observation was not reported in all biological systems (Fagnocchi et al., 2016; Sabò et al., 2014; Walz et al., 2014). It has been suggested c-Myc regulates specific genes that indirectly lead to RNA amplification (Sabò et al., 2014; Sabò and Amati, 2014; Walz et al., 2014). This has also been suggested of MYCN (Duffy et al., 20142015). The reported differences could be a result of the intrinsic variation between cell lines in maintaining the transcriptome (Trakhtenberg et al., 2016). Indeed, a recent study reported that distinct transcriptional regulation can be accounted for by differences in promoter affinity under different c-Myc expression levels (Lorenzin et al., 2016).

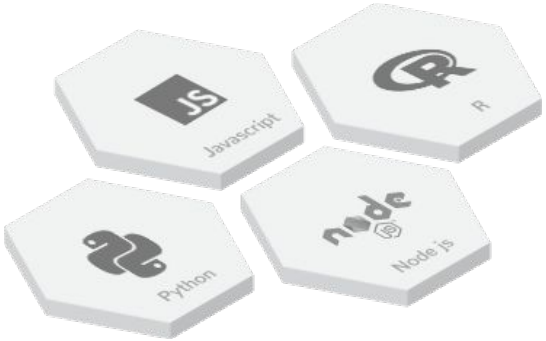
The outcome measures reported in this Replication Study will be aggregated with those from the other Replication Studies to create a dataset that will be examined to provide evidence about reproducibility of cancer biology research, and to identify factors that influence

Towards a scalable infrastructure for reproducible document publishing

1. Interoperable authoring and conversion tools
2. Portable reproducible documents
3. Reliable and performant reproducible execution environments
4. Publisher tools

Authoring and Conversion tools

- Authoring with Stencila Desktop: an intuitive, clean text editor built on top of Texture, with code cells and reproducible figures
  - “Mini” formula language for Excel-like graphing
- Conversion with Encoda: allow conversion from commonly used formats (e.g. Jupyter notebooks, R Markdown, Google Doc, LaTeX, PDF) to DAR



eLIFE

Encoda

Documents, markup, and notebook formats

Format	Status
Markdown	status alpha
RMarkdown	status alpha
Latex	status alpha
HTML	status alpha
PDF	-
Google Doc	status alpha

Tabular data and spreadsheet formats

Format	Status
CSV	status alpha
Yaml front matter for CSV CSVY	#25
Excel (.xlsx)	status alpha
OpenDocument Spreadsheet	status alpha
Tabular Data Package	status alpha

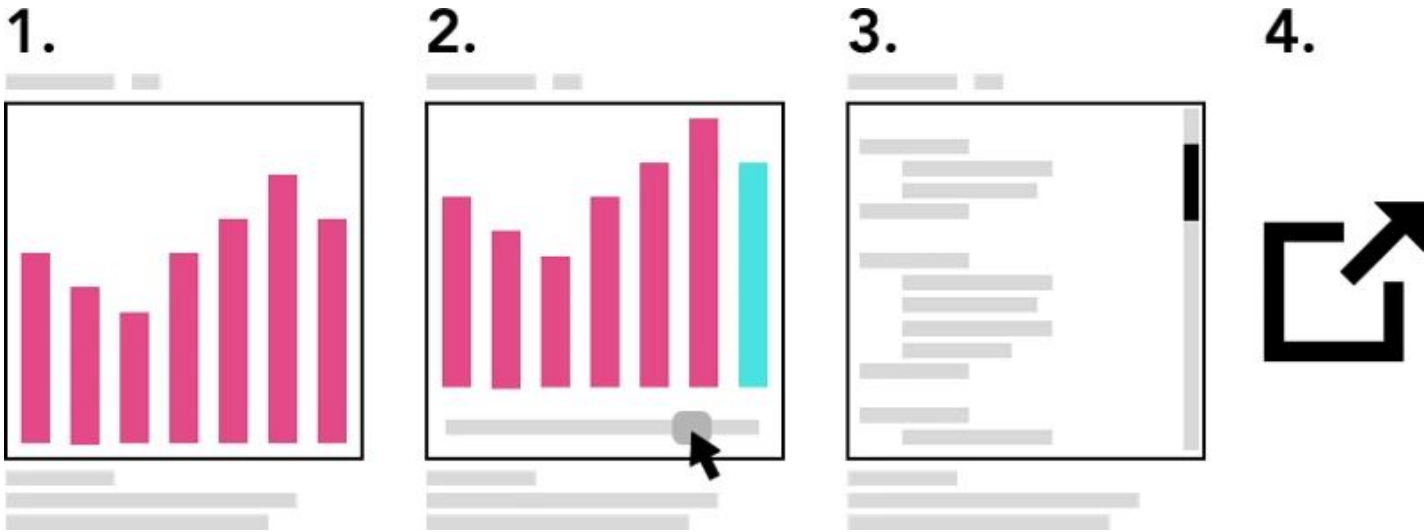
Portable reproducible documents

- DAR (Document ARchive) Container for text, code, data and media assets
- Standard JATS-XML based data formats
- Open format
- Extension to support R Markdown inline code cells to enhance interoperability

Reproducible execution environments

- Stencila Hub to provide reliant and performant execution environments to run live-code elements
- Building on top of existing technologies: Jupyter Kernels, Binder Hub, Docker

Publisher tools: Progressive enhancement via multi-level output



Plain .PNG
(e.g., mobile)

Interactive

Code / Data view
(edit, execute)

eLIFE

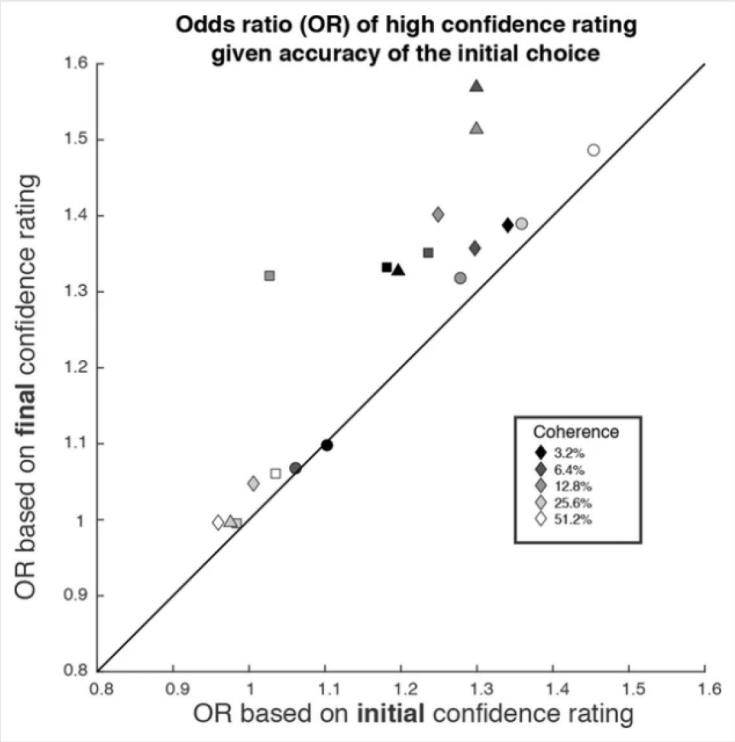
Casual reader

Side by side

Jump to

- Abstract
- eLife digest
- Introduction
- Results
- Discussion
- Materials and methods
- References
- Decision letter
- Author response
- Article and author information
- Metrics

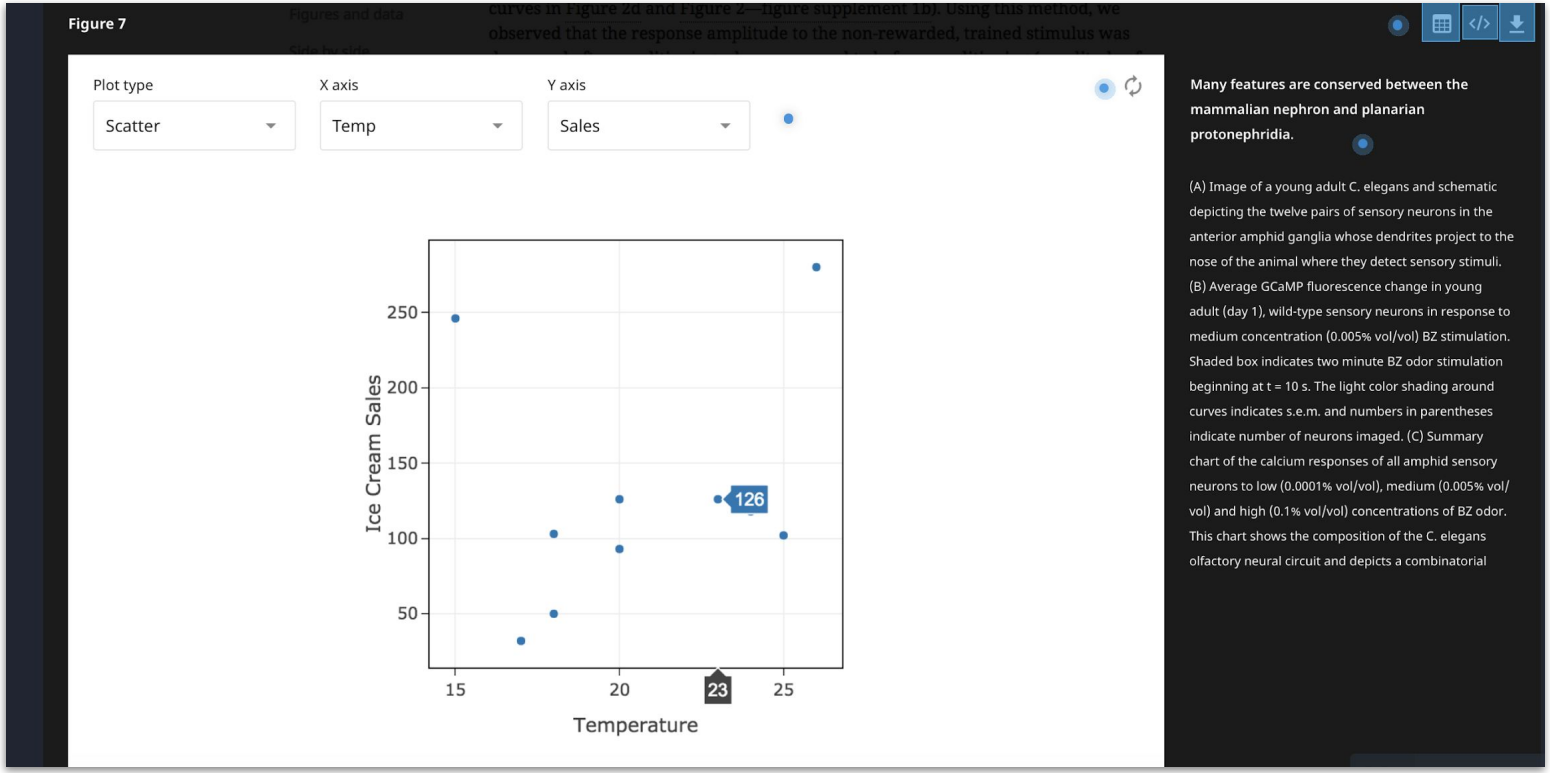
Figure 7



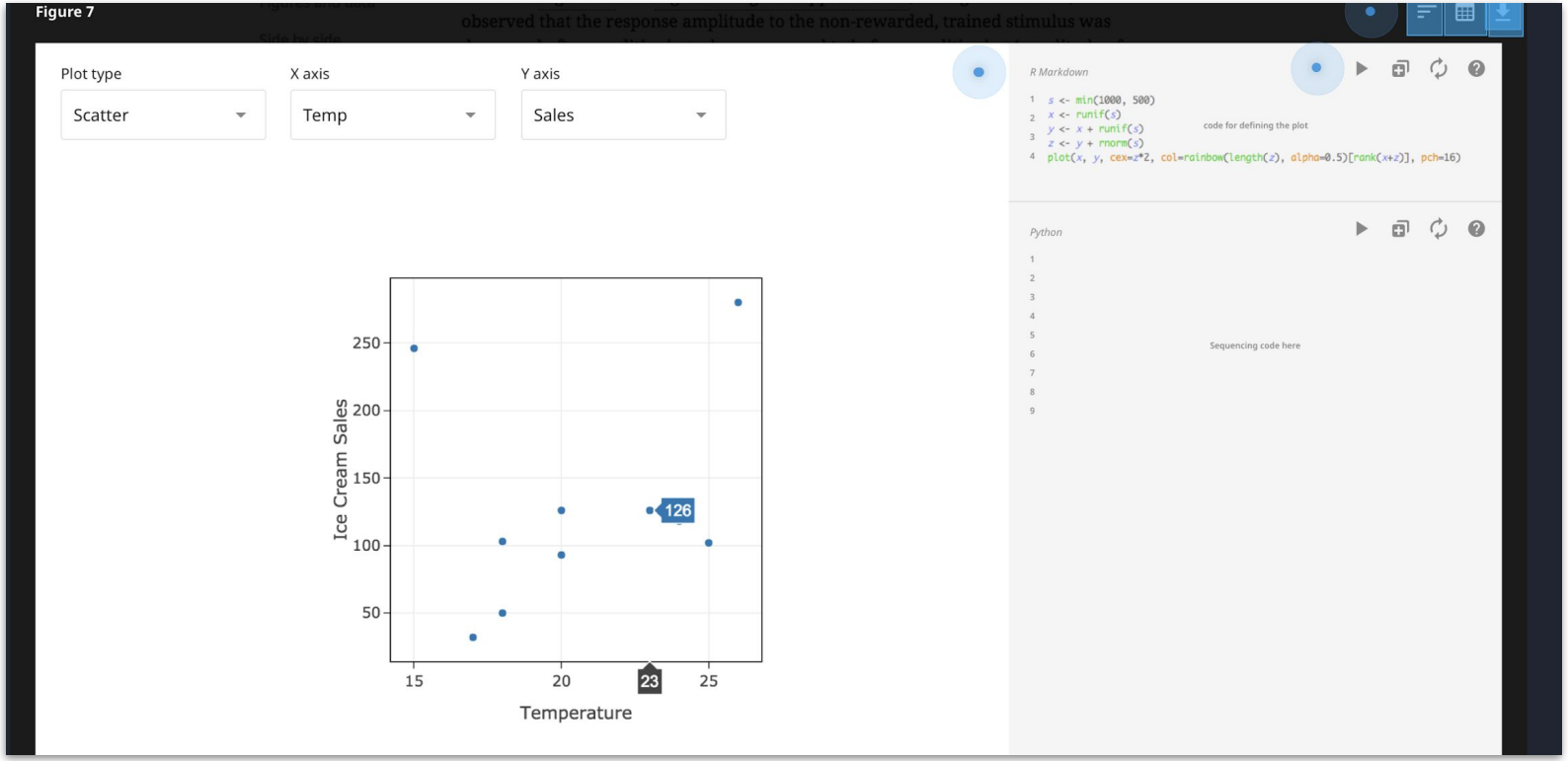
The possibility of change of confidence introduces an apparent dissociation between accuracy and

eLIFE

Interactive figures



Interacting with code and data



eLIFE

Publisher tools

- Web-based publishing of fully reproducible documents with in-browser code interaction and execution
  - Converted to HTML and served from Stencila Hub; or
  - Rendered by Javascript in browser using a Texture Reader interface
- Quick export to PDF for legacy systems
- Journal submission infrastructure integration

Workflow

1. **Authoring.** Author write articles in DAR or notebook format (e.g. Rmd, Jupyter)
2. **Uploading.** Author uploads article, and necessary data and code files, to a “project” on Stencila Hub.
3. **Building.** A reproducible execution environment is built for the article, based on the software packages used in it.
4. **Verification.** The article is executed within the reproducible execution environment to verify that it is indeed reproducible.
5. **Conversion.** Once the article has been verified as being reproducible, the author presses a “Create DAR” button (when not already using the format) to export their article to DAR ready for eLife’s production team to use.
6. **Publication.** A reproducible companion version of the article made available using publishers tools.

Core development principles

- Open
  - Not trying to “win” a tools race
- Interoperable
  - Easy for scientists to create / publishers to publish reproducible documents from multiple starting points
- Modular
  - Tools within the stack can be taken out and integrated into other pipelines
  - Minimise dependencies for reuse

eLI

Helping scientists accelerate discovery by operating a platform for research communication that encourages and recognises the most responsible behaviours in science.

You can help

- Share your use case
- Provide feedback
- Learn about progress and opportunities to help

Sign up: elifesci.org/RDSupdates

This will take you to a form asking for your consent to be added to a mailing list for ~bimonthly emails with updates about this project, including calls for contributions and feedback.

Research Practice Survey: www.surveymonkey.co.uk/r/RPSeLifeweb

Questions?

Email: e.tsang@elifesciences.org

Twitter: @eLifeInnovation / @emmy_ft

Labs: elifesci.org/labs

Stencila: stenci.la

Substance: substance.io

doi.org/10.6084/m9.figshare.9868625

Pandoc: pandoc.org

Pandoc a universal document converter

Donate

About

Installing

Getting started

Demos

Documentation

Help

Extras

Releases

About pandoc

If you need to convert files from one markup format into another, pandoc is your swiss-army knife. Pandoc can convert between the following formats:

(← = conversion from; → = conversion to; ↔ = conversion from and to)

Lightweight markup formats

- ↔ Markdown (including CommonMark and GitHub-flavored Markdown)
- ↔ reStructuredText
- → AsciiDoc
- ↔ Emacs Org-Mode
- ↔ Emacs Muse
- → Textile
- ← txt2tags

HTML formats

- ↔ (X)HTML 4
- ↔ HTML5

Ebooks

- ↔ EPUB version 2 or 3
- ↔ FictionBook2

Documentation formats

- → GNU TexInfo

Word processor formats

- ↔ Microsoft Word docx
- ↔ OpenOffice/LibreOffice ODT
- → OpenDocument XML
- → Microsoft PowerPoint

Page layout formats

- → InDesign ICML

Outline formats

- ↔ OPML

Wiki markup formats

- ↔ MediaWiki markup
- ↔ DokuWiki markup
- ← TikiWiki markup
- ← TWiki markup
- → Vimwiki markup
- → XWiki markup
- → ZimWiki markup

Encoda

Pandoc

pandoc sunspots.ipynb -o sunspots.docx

Getting started

OK, let's just dive right in and fill in details as we go. I'll be using Python for this exploration but will focus on the story and not the code.

First things first, let's load the sunspots data, which is easy to find (e.g. from NOAA) and conveniently included in a popular Python package for doing statistical work...

```
import statsmodels.api as sm
import pandas as pd
data_loader = sm.datasets.sunspots.load_pandas()
df = data_loader.data
```

df is shorthand for "dataframe", which we can think of as an Excel-like table of values. Dataframes have various methods that can be called to easily learn about the data contained in them, and we'll step through calling some of these methods. Below, we see that we have 309 pairs of (year, activity) to examine...

```
df
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 309 entries, 0 to 308
Data columns:
YEAR          309  non-null values
SUNACTIVITY   309  non-null values
dtypes: float64(2)
```

We can quickly inspect the first and last handful of values to get an idea of what the data look like...

```
df.head()
```

```
   YEAR  SUNACTIVITY
0  1700          5
1  1701         11
2  1702         16
3  1703         23
4  1704         36
```

```
df.tail()
```

```
     YEAR  SUNACTIVITY
304  2004         40.4
305  2005         29.8
306  2006         15.2
307  2007          7.5
308  2008          2.9
```

Stencila + Pandoc

stencila convert sunspots.ipynb sunspots.docx

Getting started

OK, let's just dive right in and fill in details as we go. I'll be using Python for this exploration but will focus on the story and not the code.

First things first, let's load the sunspots data, which is easy to find (e.g. from NOAA) and conveniently included in a popular Python package for doing statistical work...

df is shorthand for "dataframe", which we can think of as an Excel-like table of values. Dataframes have various methods that can be called to easily learn about the data contained in them, and we'll step through calling some of these methods. Below, we see that we have 309 pairs of (year, activity) to examine...

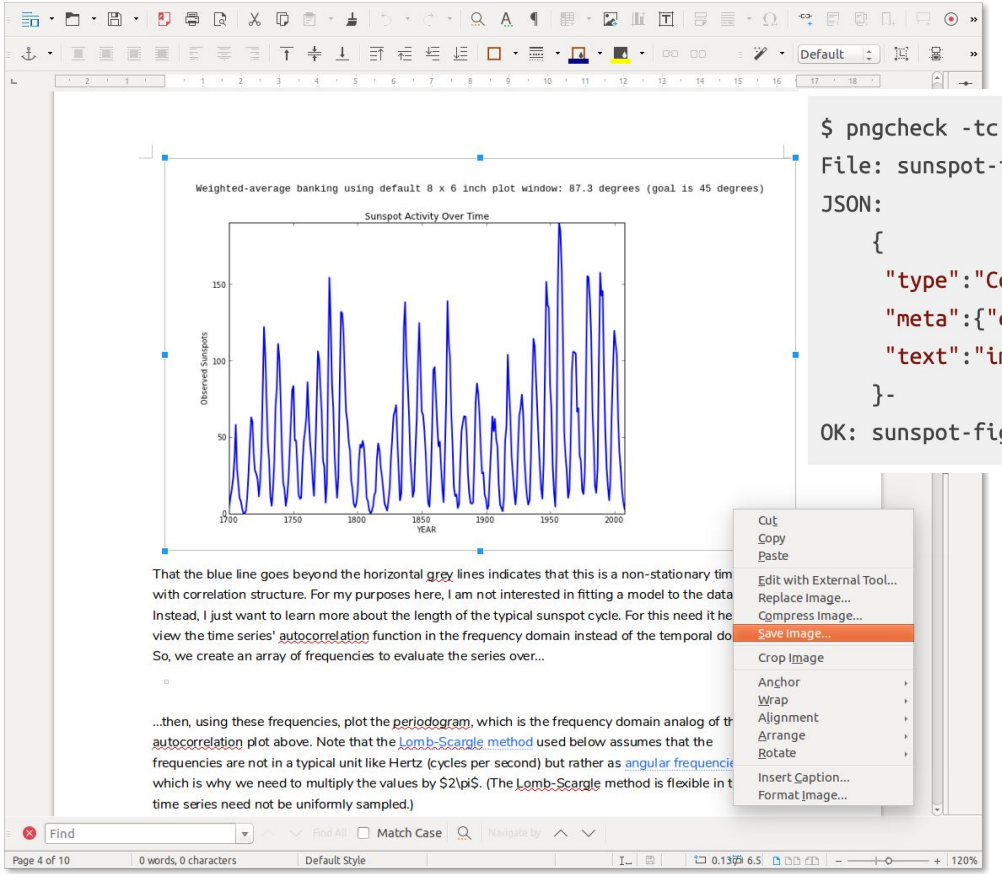
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 309 entries, 0 to 308
Data columns:
YEAR          309  non-null values
SUNACTIVITY   309  non-null values
dtypes: float64(2)
```

We can quickly inspect the first and last handful of values to get an idea of what the data look like...

```
0 1700 5
1 1701 11
2 1702 16
3 1703 23
4 1704 36
```


eLIFE

Encoda




```
$ pngcheck -tc sunspot-figure.png
File: sunspot-figure.png (50854 bytes)
JSON:
    {
     "type":"CodeChunk",
     "meta":{"execution_count":18},
     "text":"import scipy.optimize as spo\ntarget = np.radians(45)   \nsl
    }-
OK: sunspot-figure.png (832x518, 32-bit RGB+alpha, non-interlaced, 97.1%)
```

Dockta: smaller Docker images optimised for reproducible articles

- Performs static code analysis to determine package requirements.
- Uses package databases to determine package system dependencies and generate linked metadata
- Quicker installation of package dependencies

https://stencila.github.io/dockta/