BIG Data, **BIG** responsibility

Maneage: Managing data lineage for long-term and archivable reproducibility

(Published in CiSE 23 (3), pp 82-91: DOI:10.1109/MCSE.2021.3072860, arXiv:2006.03018)

Mohammad Akhlaghi
Centro de Estudios de Física del Cosmos de Aragón (CEFCA), Teruel, Spain

Royal Observatory Coffee talk; Edinburgh
23rd of May 2023

Most recent slides available in link below (this PDF is built from Git commit c747d78-dirty):
`https://maneage.org/pdf/slides-intro-short.pdf`

Financiado por la Unión Europea-NextGenerationEU

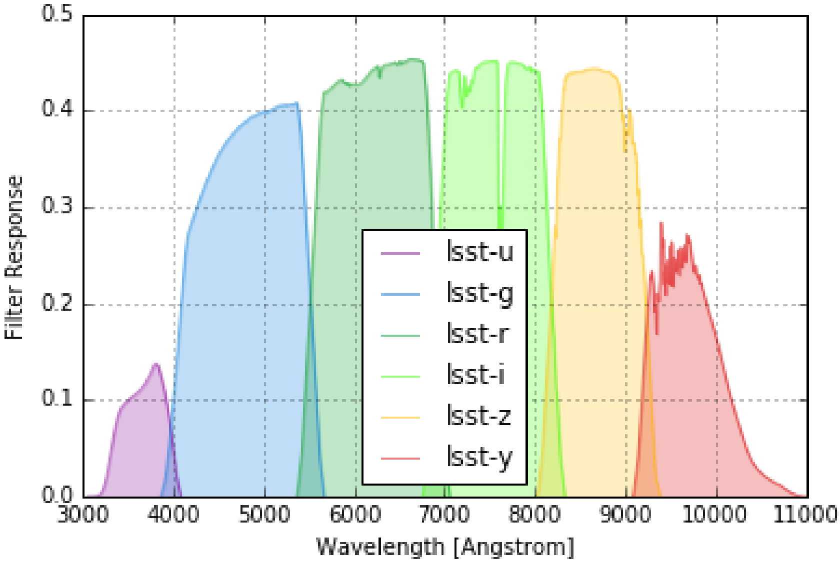
Unión Europea
Fondo Europeo de Desarrollo Regional
"Una manera de hacer Europa"

This project received financial support from the European Union's Horizon 2020 research and innovation grant agreement No. 777388 to the RDA EU 4.0 project.

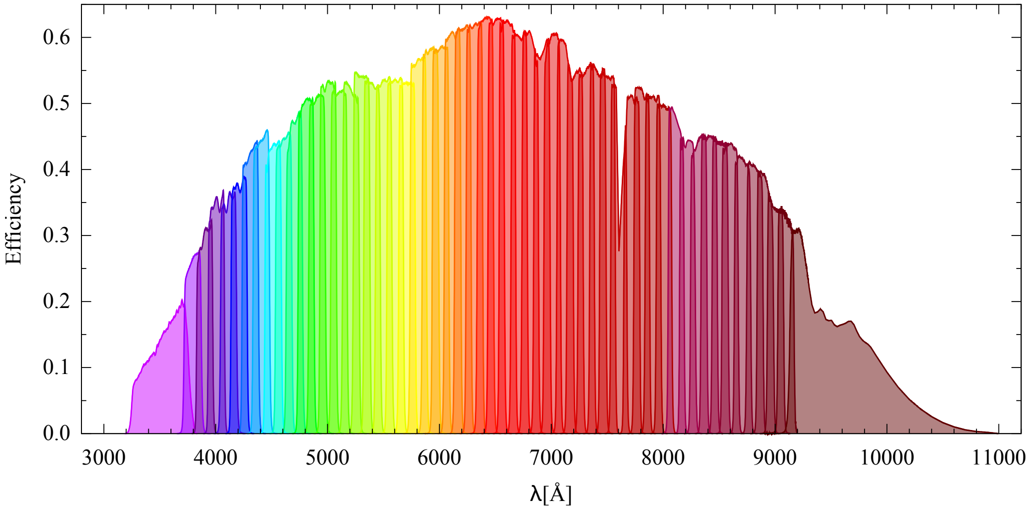
Our main project: **J-PAS** with Observatorio Astrofísico de Javalambre (OAJ)

J-PAS will observe the northern sky in 56 medium-band filters ($\sim$ 14nm):

LSST filter: 6 (image from speclite docs):

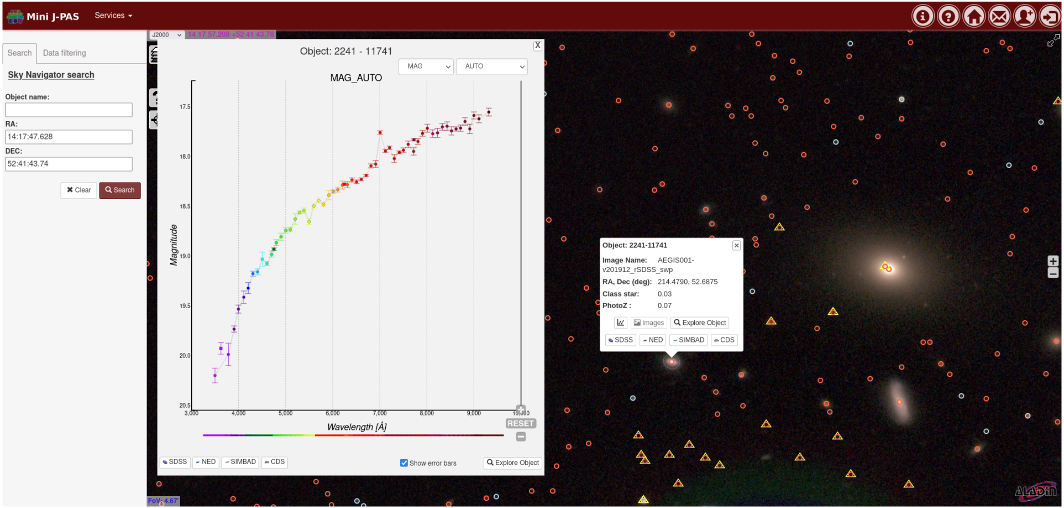


J-PAS filters: 56 (Bonoli+2021: 2021A&A...653A..31B)

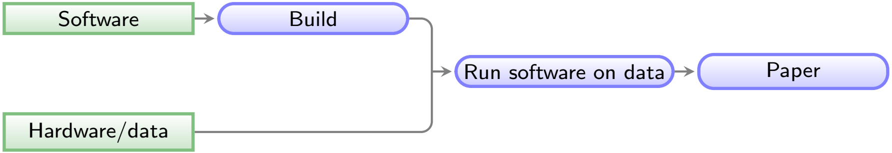


Result: photo-spectra of every pixel of the non-Galactic northern sky (like an IFU)!

http://archive.cefca.es/catalogues/minijpas-pdr201912/navigator.html

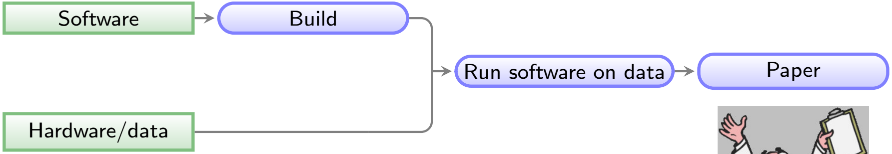


General outline of a project (after data collection)



Green boxes with sharp corners: *source*/input components/files.
Blue boxes with rounded corners: *built* components.
Red boxes with dashed borders: questions that must be clarified for each phase.

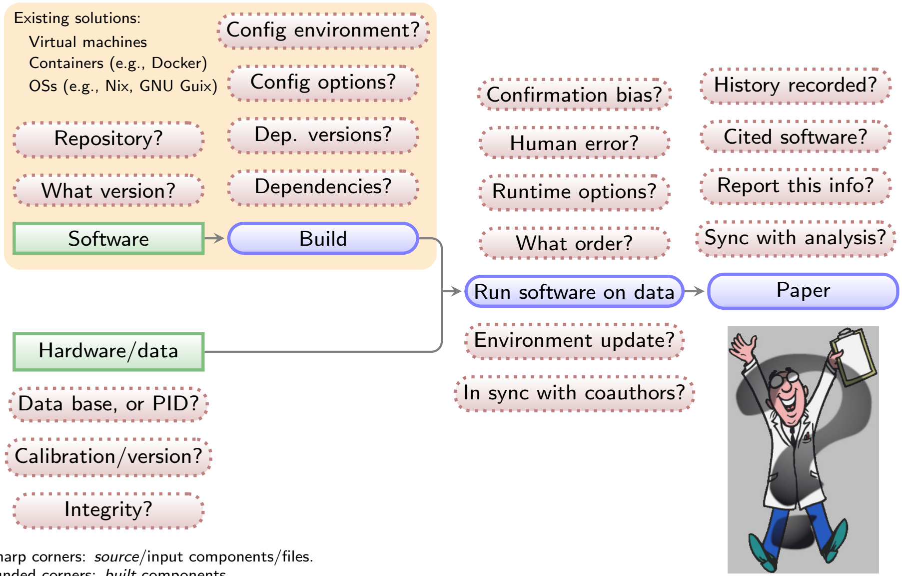
General outline of a project (after data collection)



https://heywhatwhatdidyousay.wordpress.com

Green boxes with sharp corners: *source*/input components/files.
Blue boxes with rounded corners: *built* components.
Red boxes with dashed borders: questions that must be clarified for each phase.

General outline of a project (after data collection)



https://heywhatwhatdidyousay.wordpress.com
http://pngimages.net

Green boxes with sharp corners: *source*/input components/files.
Blue boxes with rounded corners: *built* components.
Red boxes with dashed borders: questions that must be clarified for each phase.

Science is a tricky business

> Data analysis [...] is a human behavior. Researchers who hunt hard enough will turn up a result that fits statistical criteria, but their discovery will probably be a false positive.
>
> Five ways to fix statistics (Nature, 551, Nov 2017; DOI:10.1038/d41586-017-07522-z).

“Reproducibility crisis” in the sciences? (Baker 2016, Nature 533, 452, DOI:10.1038/533452a)

1576 researchers participated in a survey by Nature, 90% believed in a crisis!

Status	% agreed
Yes, a significant crisis	52
Yes, a slight crisis	38
Don't know	7
No, there is no crisis	3

Full PDF available at https://www.nature.com/articles/533452a.pdf

Notebooks are not long-term solutions (see appendices of Akhlaghi+2021: arXiv:2006.03018)

Results from run on May 10th, 2022:

```
[mohammad@akhlaghi software]$ bash Miniconda3-latest-Linux-x86_64.sh -b -p conda-install
PREFIX=/home/mohammad/tmp/software/conda-install
Unpacking payload ...
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/mohammad/tmp/software/conda-install

  added / updated specs:
    - _libgcc_mutex==0.1=main
    - _openmp_mutex==4.5=1_gnu
    - brotlipy==0.7.0=py39h27cfd23_1003
    - ca-certificates==2021.10.26=h06a4308_2
    - certifi==2021.10.8=py39h06a4308_2
    - cffi==1.15.0=py39hd667e15_1
    - charset-normalizer==2.0.4=pyhd3eb1b0_0
    - conda-content-trust==0.1.1=pyhd3eb1b0_0
    - conda-package-handling==1.7.3=py39h27cfd23_1
    - conda==4.11.0=py39h06a4308_0
    - cryptography==36.0.0=py39h9ce1e76_0
    - idna==3.3=pyhd3eb1b0_0
    - ld_impl_linux-64==2.35.1=h7274673_9
    - libffi==3.3=he6710b0_2
    - libgcc-ng==9.3.0=h5101ec6_17
    - libgomp==9.3.0=h5101ec6_17
    - libstdcxx-ng==9.3.0=hd4cf53a_17
    - ncurses==6.3=h7f8727e_2
    - openssl==1.1.1m=h7f8727e_0
    - pip==21.2.4=py39h06a4308_0
    - pycosat==0.6.3=py39h27cfd23_0
    - pycparser==2.21=pyhd3eb1b0_0
    - pyopenssl==21.0.0=pyhd3eb1b0_1
    - pysocks==1.7.1=py39h06a4308_0
    - python==3.9.7=h12debd9_1
    - readline==8.1.2=h7f8727e_1
    - requests==2.27.1=pyhd3eb1b0_0
    - ruamel_yaml==0.15.100=py39h27cfd23_0
    - setuptools==58.0.4=py39h06a4308_0
    - six==1.16.0=pyhd3eb1b0_0
    - sqlite==3.37.0=hc218d9a_0
    - tk==8.6.11=h1ccaba5_0
    - tqdm==4.62.3=pyhd3eb1b0_1
    - tzdata==2021e=hda174b7_0
    - urllib3==1.26.7=pyhd3eb1b0_0
    - wheel==0.37.1=pyhd3eb1b0_0
    - xz==5.2.5=h7b6447c_0
    - yaml==0.2.5=h7b6447c_0
    - zlib==1.2.11=h7f8727e_4
```

```
Collecting jupyterlab-pygments
  Downloading jupyterlab_pygments-0.2.2-py2.py3-none-any.whl (21 kB)
Collecting argon2-cffi-bindings
  Downloading argon2_cffi_bindings-21.2.0-cp36-abi3-manylinux_2_17_x86_64.manylin
ux2014_x86_64.whl (86 kB)
     |████████████████████████████████| 86 kB 3.1 MB/s
Requirement already satisfied: cffi>=1.0.1 in ./conda-install/lib/python3.9/site-
packages (from argon2-cffi-bindings->argon2-cffi->notebook->jupyter) (1.15.0)
Requirement already satisfied: pycparser in ./conda-install/lib/python3.9/site-pa
ckages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook->jupyter) (
2.21)
Collecting soupsieve>1.2
  Downloading soupsieve-2.3.2.post1-py3-none-any.whl (37 kB)
Collecting webencodings
  Downloading webencodings-0.5.1-py2.py3-none-any.whl (11 kB)
Collecting pyparsing!=3.0.5,>=2.0.2
  Downloading pyparsing-3.0.8-py3-none-any.whl (98 kB)
     |████████████████████████████████| 98 kB 3.4 MB/s
Collecting qtpy>=2.0.1
  Downloading QtPy-2.1.0-py3-none-any.whl (68 kB)
     |████████████████████████████████| 68 kB 2.9 MB/s
Collecting pure-eval
  Downloading pure_eval-0.2.2-py3-none-any.whl (11 kB)
Collecting asttokens
  Downloading asttokens-2.0.5-py2.py3-none-any.whl (20 kB)
Collecting executing
  Downloading executing-0.8.3-py2.py3-none-any.whl (16 kB)
Installing collected packages: traitlets, pyrsistent, attrs, wcwidth, tornado, py
zmq, python-dateutil, pure-eval, ptyprocess, parso, nest-asyncio, jupyter-core, j
sonschema, fastjsonschema, executing, entrypoints, asttokens, webencodings, stack
-data, soupsieve, pyparsing, pygments, prompt-toolkit, pickleshare, pexpect, nbfo
rmat, matplotlib-inline, MarkupSafe, jupyter-client, jedi, decorator, backcall, t
inycss2, psutil, pandocfilters, packaging, nbclient, mistune, jupyterlab-pygments
, jinja2, ipython, defusedxml, debugpy, bleach, beautifulsoup4, argon2-cffi-bindi
ngs, terminado, Send2Trash, prometheus-client, nbconvert, ipython-genutils, ipyke
rnel, argon2-cffi, notebook, widgetsnbextension, qtpy, jupyterlab-widgets, qtcons
ole, jupyter-console, ipywidgets, jupyter
Successfully installed MarkupSafe-2.1.1 Send2Trash-1.8.0 argon2-cffi-21.3.0 argon
2-cffi-bindings-21.2.0 asttokens-2.0.5 attrs-21.4.0 backcall-0.2.0 beautifulsoup4
-4.11.1 bleach-5.0.0 debugpy-1.6.0 decorator-5.1.1 defusedxml-0.7.1 entrypoints-0
.4 executing-0.8.3 fastjsonschema-2.15.3 ipykernel-6.13.0 ipython-8.3.0 ipython-g
enutils-0.2.0 ipywidgets-7.7.0 jedi-0.18.1 jinja2-3.1.2 jsonschema-4.5.1 jupyter-
1.0.0 jupyter-client-7.3.1 jupyter-console-6.4.3 jupyter-core-4.10.0 jupyterlab-p
ygments-0.2.2 jupyterlab-widgets-1.1.0 matplotlib-inline-0.1.3 mistune-0.8.4 nbcl
ient-0.6.3 nbconvert-6.5.0 nbformat-5.4.0 nest-asyncio-1.5.5 notebook-6.4.11 pack
aging-21.3 pandocfilters-1.5.0 parso-0.8.3 pexpect-4.8.0 pickleshare-0.7.5 promet
heus-client-0.14.1 prompt-toolkit-3.0.29 psutil-5.9.0 ptyprocess-0.7.0 pure-eval-
0.2.2 pygments-2.12.0 pyparsing-3.0.8 pyrsistent-0.18.1 python-dateutil-2.8.2 pyz
mq-22.3.0 qtconsole-5.3.0 qtpy-2.1.0 soupsieve-2.3.2.post1 stack-data-0.2.0 termi
nado-0.13.3 tinycss2-1.1.1 tornado-6.1 traitlets-5.2.0 wcwidth-0.2.5 webencodings
-0.5.1 widgetsnbextension-3.6.0
(base) [mohammad@akhlaghi software]$ pip install jupyter
```

Notebooks are not long-term solutions (see appendices of Akhlaghi+2021: arXiv:2006.03018)

```
[mohammad@akhlaghi software]$ bash Miniconda3-latest-Linux-x86_64.sh -b -p conda-install
PREFIX=/home/mohammad/tmp/software/conda-install
Unpacking payload ...
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/mohammad/tmp/software/conda-install

  added / updated specs:
    - _libgcc_mutex==0.1=main
    - _openmp_mutex==4.5=1_gnu
    - brotlipy==0.7.0=py39h27cfd23_1003
    - ca-certificates==2021.10.26=h06a4308_2
    - certifi==2021.10.8=py39h06a4308_2
    - cffi==1.15.0=py39hd667e15_1
    - charset-normalizer==2.0.4=pyhd3eb1b0_0
    - conda-content-trust==0.1.1=pyhd3eb1b0_0
    - conda-package-handling==1.7.3=py39h27cfd23_1
    - conda==4.11.0=py39h06a4308_0
    - cryptography==36.0.0=py39h9ce1e76_0
    - idna==3.3=pyhd3eb1b0_0
    - ld_impl_linux-64==2.35.1=h7274673_9
    - libffi==3.3=he6710b0_2
    - libgcc-ng==9.3.0=h5101ec6_17
    - libgomp==9.3.0=h5101ec6_17
    - libstdcxx-ng==9.3.0=hd4cf53a_17
    - ncurses==6.3=h7f8727e_2
    - openssl==1.1.1m=h7f8727e_0
    - pip==21.2.4=py39h06a4308_0
    - pycosat==0.6.3=py39h27cfd23_0
    - pycparser==2.21=pyhd3eb1b0_0
    - pyopenssl==21.0.0=pyhd3eb1b0_1
    - pysocks==1.7.1=py39h06a4308_0
    - python==3.9.7=h12debd9_1
    - readline==8.1.2=h7f8727e_1
    - requests==2.27.1=pyhd3eb1b0_0
    - ruamel_yaml==0.15.100=py39h27cfd23_0
    - setuptools==58.0.4=py39h06a4308_0
    - six==1.16.0=pyhd3eb1b0_0
    - sqlite==3.37.0=hc218d9a_0
    - tk==8.6.11=h1ccaba5_0
    - tqdm==4.62.3=pyhd3eb1b0_1
    - tzdata==2021e=hda174b7_0
    - urllib3==1.26.7=pyhd3eb1b0_0
    - wheel==0.37.1=pyhd3eb1b0_0
    - xz==5.2.5=h7b6447c_0
    - yaml==0.2.5=h7b6447c_0
    - zlib==1.2.11=h7f8727e_4
```

```
Collecting jupyterlab-pygments
  Downloading jupyterlab_pygments-0.2.2-py2.py3-none-any.whl (21 kB)
Collecting argon2-cffi-bindings
  Downloading argon2_cffi_bindings-21.2.0-cp36-abi3-manylinux_2_17_x86_64.manylin
ux2014_x86_64.whl (86 kB)
     |████████████████████████████████| 86 kB 3.1 MB/s
Requirement already satisfied: cffi>=1.0.1 in ./conda-install/lib/python3.9/site-
packages (from argon2-cffi-bindings->argon2-cffi->notebook->jupyter) (1.15.0)
Requirement already satisfied: pycparser in ./conda-install/lib/python3.9/site-pa
ckages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook->jupyter) (
2.21)
Collecting soupsieve>1.2
  Downloading soupsieve-2.3.2.post1-py3-none-any.whl (37 kB)
Collecting webencodings
  Downloading webencodings-0.5.1-py2.py3-none-any.whl (11 kB)
Collecting pyparsing!=3.0.5,>=2.0.2
  Downloading pyparsing-3.0.8-py3-none-any.whl (98 kB)
     |████████████████████████████████| 98 kB 3.4 MB/s
Collecting qtpy>=2.0.1
  Downloading QtPy-2.1.0-py3-none-any.whl (68 kB)
     |████████████████████████████████| 68 kB 2.9 MB/s
Collecting pure-eval
  Downloading pure_eval-0.2.2-py3-none-any.whl (11 kB)
Collecting asttokens
  Downloading asttokens-2.0.5-py2.py3-none-any.whl (20 kB)
Collecting executing
  Downloading executing-0.8.3-py2.py3-none-any.whl (16 kB)
Installing collected packages: traitlets, pyrsistent, attrs, wcwidth, tornado, py
zmq, python-dateutil, pure-eval, ptyprocess, parso, nest-asyncio, jupyter-core, j
sonschema, fastjsonschema, executing, entrypoints, asttokens, webencodings, stack
-data, soupsieve, pyparsing, pygments, prompt-toolkit, pickleshare, pexpect, nbfo
rmat, matplotlib-inline, MarkupSafe, jupyter-client, jedi, decorator, backcall, t
inycss2, psutil, pandocfilters, packaging, nbclient, mistune, jupyterlab-pygments
, jinja2, ipython, defusedxml, debugpy, bleach, beautifulsoup4, argon2-cffi-bindi
ngs, terminado, Send2Trash, prometheus-client, nbconvert, ipython-genutils, ipyke
rnel, argon2-cffi, notebook, widgetsnbextension, qtpy, jupyterlab-widgets, qtcons
ole, jupyter-console, ipywidgets, jupyter
Successfully installed MarkupSafe-2.1.1 Send2Trash-1.8.0 argon2-cffi-21.3.0 argon
2-cffi-bindings-21.2.0 asttokens-2.0.5 attrs-21.4.0 backcall-0.2.0 beautifulsoup4
-4.11.1 bleach-5.0.0 debugpy-1.6.0 decorator-5.1.1 defusedxml-0.7.1 entrypoints-0
.4 executing-0.8.3 fastjsonschema-2.15.3 ipykernel-6.13.0 ipython-8.3.0 ipython-g
enutils-0.2.0 ipywidgets-7.7.0 jedi-0.18.1 jinja2-3.1.2 jsonschema-4.5.1 jupyter-
1.0.0 jupyter-client-7.3.1 jupyter-console-6.4.3 jupyter-core-4.10.0 jupyterlab-p
ygments-0.2.2 jupyterlab-widgets-1.1.0 matplotlib-inline-0.1.3 mistune-0.8.4 nbcl
ient-0.6.3 nbconvert-6.5.0 nbformat-5.4.0 nest-asyncio-1.5.5 notebook-6.4.11 pack
aging-21.3 pandocfilters-1.5.0 parso-0.8.3 pexpect-4.8.0 pickleshare-0.7.5 promet
heus-client-0.14.1 prompt-toolkit-3.0.29 psutil-5.9.0 ptyprocess-0.7.0 pure-eval-
0.2.2 pygments-2.12.0 pyparsing-3.0.8 pyrsistent-0.18.1 python-dateutil-2.8.2 pyz
mq-22.3.0 qtconsole-5.3.0 qtpy-2.1.0 soupsieve-2.3.2.post1 stack-data-0.2.0 termi
nado-0.13.3 tinycss2-1.1.1 tornado-6.1 traitlets-5.2.0 wcwidth-0.2.5 webencodings
-0.5.1 widgetsnbextension-3.6.0
(base) [mohammad@akhlaghi software]$ pip install jupyter
```

Results from run on May 10th, 2022:

Conda setup:
39 dependencies

Notebooks are not long-term solutions (see appendices of Akhlaghi+2021: arXiv:2006.03018)

```
[mohammad@akhlaghi software]$ bash Miniconda3-latest-Linux-x86_64.sh -b -p conda-install
PREFIX=/home/mohammad/tmp/software/conda-install
Unpacking payload ...
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/mohammad/tmp/software/conda-install

  added / updated specs:
    - _libgcc_mutex==0.1=main
    - _openmp_mutex==4.5=1_gnu
    - brotlipy==0.7.0=py39h27cfd23_1003
    - ca-certificates==2021.10.26=h06a4308_2
    - certifi==2021.10.8=py39h06a4308_2
    - cffi==1.15.0=py39hd667e15_1
    - charset-normalizer==2.0.4=pyhd3eb1b0_0
    - conda-content-trust==0.1.1=pyhd3eb1b0_0
    - conda-package-handling==1.7.3=py39h27cfd23_1
    - conda==4.11.0=py39h06a4308_0
    - cryptography==36.0.0=py39h9ce1e76_0
    - idna==3.3=pyhd3eb1b0_0
    - ld_impl_linux-64==2.35.1=h7274673_9
    - libffi==3.3=he6710b0_2
    - libgcc-ng==9.3.0=h5101ec6_17
    - libgomp==9.3.0=h5101ec6_17
    - libstdcxx-ng==9.3.0=hd4cf53a_17
    - ncurses==6.3=h7f8727e_2
    - openssl==1.1.1m=h7f8727e_0
    - pip==21.2.4=py39h06a4308_0
    - pycosat==0.6.3=py39h27cfd23_0
    - pycparser==2.21=pyhd3eb1b0_0
    - pyopenssl==21.0.0=pyhd3eb1b0_1
    - pysocks==1.7.1=py39h06a4308_0
    - python==3.9.7=h12debd9_1
    - readline==8.1.2=h7f8727e_1
    - requests==2.27.1=pyhd3eb1b0_0
    - ruamel_yaml==0.15.100=py39h27cfd23_0
    - setuptools==58.0.4=py39h06a4308_0
    - six==1.16.0=pyhd3eb1b0_0
    - sqlite==3.37.0=hc218d9a_0
    - tk==8.6.11=h1ccaba5_0
    - tqdm==4.62.3=pyhd3eb1b0_1
    - tzdata==2021e=hda174b7_0
    - urllib3==1.26.7=pyhd3eb1b0_0
    - wheel==0.37.1=pyhd3eb1b0_0
    - xz==5.2.5=h7b6447c_0
    - yaml==0.2.5=h7b6447c_0
    - zlib==1.2.11=h7f8727e_4
```

```
Collecting jupyterlab-pygments
  Downloading jupyterlab_pygments-0.2.2-py2.py3-none-any.whl (21 kB)
Collecting argon2-cffi-bindings
  Downloading argon2_cffi_bindings-21.2.0-cp36-abi3-manylinux_2_17_x86_64.manylin
ux2014_x86_64.whl (86 kB)
     |████████████████████████████████| 86 kB 3.1 MB/s
Requirement already satisfied: cffi>=1.0.1 in ./conda-install/lib/python3.9/site-
packages (from argon2-cffi-bindings->argon2-cffi->notebook->jupyter) (1.15.0)
Requirement already satisfied: pycparser in ./conda-install/lib/python3.9/site-pa
ckages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook->jupyter) (
2.21)
Collecting soupsieve>1.2
  Downloading soupsieve-2.3.2.post1-py3-none-any.whl (37 kB)
Collecting webencodings
  Downloading webencodings-0.5.1-py2.py3-none-any.whl (11 kB)
Collecting pyparsing!=3.0.5,>=2.0.2
  Downloading pyparsing-3.0.8-py3-none-any.whl (98 kB)
     |████████████████████████████████| 98 kB 3.4 MB/s
Collecting qtpy>=2.0.1
  Downloading QtPy-2.1.0-py3-none-any.whl (68 kB)
     |████████████████████████████████| 68 kB 2.9 MB/s
Collecting pure-eval
  Downloading pure_eval-0.2.2-py3-none-any.whl (11 kB)
Collecting asttokens
  Downloading asttokens-2.0.5-py2.py3-none-any.whl (20 kB)
Collecting executing
  Downloading executing-0.8.3-py2.py3-none-any.whl (16 kB)
Installing collected packages: traitlets, pyrsistent, attrs, wcwidth, tornado, py
zmq, python-dateutil, pure-eval, ptyprocess, parso, nest-asyncio, jupyter-core, j
sonschema, fastjsonschema, executing, entrypoints, asttokens, webencodings, stack
-data, soupsieve, pyparsing, pygments, prompt-toolkit, pickleshare, pexpect, nbfo
rmat, matplotlib-inline, MarkupSafe, jupyter-client, jedi, decorator, backcall, t
inycss2, psutil, pandocfilters, packaging, nbclient, mistune, jupyterlab-pygments
, jinja2, ipython, defusedxml, debugpy, bleach, beautifulsoup4, argon2-cffi-bindi
ngs, terminado, Send2Trash, prometheus-client, nbconvert, ipython-genutils, ipyke
rnel, argon2-cffi, notebook, widgetsnbextension, qtpy, jupyterlab-widgets, qtcons
ole, jupyter-console, ipywidgets, jupyter
Successfully installed MarkupSafe-2.1.1 Send2Trash-1.8.0 argon2-cffi-21.3.0 argon
2-cffi-bindings-21.2.0 asttokens-2.0.5 attrs-21.4.0 backcall-0.2.0 beautifulsoup4
-4.11.1 bleach-5.0.0 debugpy-1.6.0 decorator-5.1.1 defusedxml-0.7.1 entrypoints-0
.4 executing-0.8.3 fastjsonschema-2.15.3 ipykernel-6.13.0 ipython-8.3.0 ipython-g
enutils-0.2.0 ipywidgets-7.7.0 jedi-0.18.1 jinja2-3.1.2 jsonschema-4.5.1 jupyter-
1.0.0 jupyter-client-7.3.1 jupyter-console-6.4.3 jupyter-core-4.10.0 jupyterlab-p
ygments-0.2.2 jupyterlab-widgets-1.1.0 matplotlib-inline-0.1.3 mistune-0.8.4 nbcl
ient-0.6.3 nbconvert-6.5.0 nbformat-5.4.0 nest-asyncio-1.5.5 notebook-6.4.11 pack
aging-21.3 pandocfilters-1.5.0 parso-0.8.3 pexpect-4.8.0 pickleshare-0.7.5 promet
heus-client-0.14.1 prompt-toolkit-3.0.29 psutil-5.9.0 ptyprocess-0.7.0 pure-eval-
0.2.2 pygments-2.12.0 pyparsing-3.0.8 pyrsistent-0.18.1 python-dateutil-2.8.2 pyz
mq-22.3.0 qtconsole-5.3.0 qtpy-2.1.0 soupsieve-2.3.2.post1 stack-data-0.2.0 termi
nado-0.13.3 tinycss2-1.1.1 tornado-6.1 traitlets-5.2.0 wcwidth-0.2.5 webencodings
-0.5.1 widgetsnbextension-3.6.0
(base) [mohammad@akhlaghi software]$ pip install jupyter
```

Results from run on May 10th, 2022:

Conda setup:
39 dependencies

Jupyter (with Pip):
61 dependencies

Notebooks are not long-term solutions (see appendices of Akhlaghi+2021: arXiv:2006.03018)

```
[mohammad@akhlaghi software]$ bash Miniconda3-latest-Linux-x86_64.sh -b -p conda-install
PREFIX=/home/mohammad/tmp/software/conda-install
Unpacking payload ...
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/mohammad/tmp/software/conda-install

  added / updated specs:
    - _libgcc_mutex==0.1=main
    - _openmp_mutex==4.5=1_gnu
    - brotlipy==0.7.0=py39h27cfd23_1003
    - ca-certificates==2021.10.26=h06a4308_2
    - certifi==2021.10.8=py39h06a4308_2
    - cffi==1.15.0=py39hd667e15_1
    - charset-normalizer==2.0.4=pyhd3eb1b0_0
    - conda-content-trust==0.1.1=pyhd3eb1b0_0
    - conda-package-handling==1.7.3=py39h27cfd23_1
    - conda==4.11.0=py39h06a4308_0
    - cryptography==36.0.0=py39h9ce1e76_0
    - idna==3.3=pyhd3eb1b0_0
    - ld_impl_linux-64==2.35.1=h7274673_9
    - libffi==3.3=he6710b0_2
    - libgcc-ng==9.3.0=h5101ec6_17
    - libgomp==9.3.0=h5101ec6_17
    - libstdcxx-ng==9.3.0=hd4cf53a_17
    - ncurses==6.3=h7f8727e_2
    - openssl==1.1.1m=h7f8727e_0
    - pip==21.2.4=py39h06a4308_0
    - pycosat==0.6.3=py39h27cfd23_0
    - pycparser==2.21=pyhd3eb1b0_0
    - pyopenssl==21.0.0=pyhd3eb1b0_1
    - pysocks==1.7.1=py39h06a4308_0
    - python==3.9.7=h12debd9_1
    - readline==8.1.2=h7f8727e_1
    - requests==2.27.1=pyhd3eb1b0_0
    - ruamel_yaml==0.15.100=py39h27cfd23_0
    - setuptools==58.0.4=py39h06a4308_0
    - six==1.16.0=pyhd3eb1b0_0
    - sqlite==3.37.0=hc218d9a_0
    - tk==8.6.11=h1ccaba5_0
    - tqdm==4.62.3=pyhd3eb1b0_1
    - tzdata==2021e=hda174b7_0
    - urllib3==1.26.7=pyhd3eb1b0_0
    - wheel==0.37.1=pyhd3eb1b0_0
    - xz==5.2.5=h7b6447c_0
    - yaml==0.2.5=h7b6447c_0
    - zlib==1.2.11=h7f8727e_4
```

```
Collecting jupyterlab-pygments
  Downloading jupyterlab_pygments-0.2.2-py2.py3-none-any.whl (21 kB)
Collecting argon2-cffi-bindings
  Downloading argon2_cffi_bindings-21.2.0-cp36-abi3-manylinux_2_17_x86_64.manylin
ux2014_x86_64.whl (86 kB)
     |████████████████████████████████| 86 kB 3.1 MB/s
Requirement already satisfied: cffi>=1.0.1 in ./conda-install/lib/python3.9/site-
packages (from argon2-cffi-bindings->argon2-cffi->notebook->jupyter) (1.15.0)
Requirement already satisfied: pycparser in ./conda-install/lib/python3.9/site-pa
ckages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook->jupyter) (
2.21)
Collecting soupsieve>1.2
  Downloading soupsieve-2.3.2.post1-py3-none-any.whl (37 kB)
Collecting webencodings
  Downloading webencodings-0.5.1-py2.py3-none-any.whl (11 kB)
Collecting pyparsing!=3.0.5,>=2.0.2
  Downloading pyparsing-3.0.8-py3-none-any.whl (98 kB)
     |████████████████████████████████| 98 kB 3.4 MB/s
Collecting qtpy>=2.0.1
  Downloading QtPy-2.1.0-py3-none-any.whl (68 kB)
     |████████████████████████████████| 68 kB 2.9 MB/s
Collecting pure-eval
  Downloading pure_eval-0.2.2-py3-none-any.whl (11 kB)
Collecting asttokens
  Downloading asttokens-2.0.5-py2.py3-none-any.whl (20 kB)
Collecting executing
  Downloading executing-0.8.3-py2.py3-none-any.whl (16 kB)
Installing collected packages: traitlets, pyrsistent, attrs, wcwidth, tornado, py
zmq, python-dateutil, pure-eval, ptyprocess, parso, nest-asyncio, jupyter-core, j
sonschema, fastjsonschema, executing, entrypoints, asttokens, webencodings, stack
-data, soupsieve, pyparsing, pygments, prompt-toolkit, pickleshare, pexpect, nbfo
rmat, matplotlib-inline, MarkupSafe, jupyter-client, jedi, decorator, backcall, t
inycss2, psutil, pandocfilters, packaging, nbclient, mistune, jupyterlab-pygments
, jinja2, ipython, defusedxml, debugpy, bleach, beautifulsoup4, argon2-cffi-bindi
ngs, terminado, Send2Trash, prometheus-client, nbconvert, ipython-genutils, ipyke
rnel, argon2-cffi, notebook, widgetsnbextension, qtpy, jupyterlab-widgets, qtcons
ole, jupyter-console, ipywidgets, jupyter
Successfully installed MarkupSafe-2.1.1 Send2Trash-1.8.0 argon2-cffi-21.3.0 argon
2-cffi-bindings-21.2.0 asttokens-2.0.5 attrs-21.4.0 backcall-0.2.0 beautifulsoup4
-4.11.1 bleach-5.0.0 debugpy-1.6.0 decorator-5.1.1 defusedxml-0.7.1 entrypoints-0
.4 executing-0.8.3 fastjsonschema-2.15.3 ipykernel-6.13.0 ipython-8.3.0 ipython-g
enutils-0.2.0 ipywidgets-7.7.0 jedi-0.18.1 jinja2-3.1.2 jsonschema-4.5.1 jupyter-
1.0.0 jupyter-client-7.3.1 jupyter-console-6.4.3 jupyter-core-4.10.0 jupyterlab-p
ygments-0.2.2 jupyterlab-widgets-1.1.0 matplotlib-inline-0.1.3 mistune-0.8.4 nbcl
ient-0.6.3 nbconvert-6.5.0 nbformat-5.4.0 nest-asyncio-1.5.5 notebook-6.4.11 pack
aging-21.3 pandocfilters-1.5.0 parso-0.8.3 pexpect-4.8.0 pickleshare-0.7.5 promet
heus-client-0.14.1 prompt-toolkit-3.0.29 psutil-5.9.0 ptyprocess-0.7.0 pure-eval-
0.2.2 pygments-2.12.0 pyparsing-3.0.8 pyrsistent-0.18.1 python-dateutil-2.8.2 pyz
mq-22.3.0 qtconsole-5.3.0 qtpy-2.1.0 soupsieve-2.3.2.post1 stack-data-0.2.0 termi
nado-0.13.3 tinycss2-1.1.1 tornado-6.1 traitlets-5.2.0 wcwidth-0.2.5 webencodings
-0.5.1 widgetsnbextension-3.6.0
(base) [mohammad@akhlaghi software]$ pip install jupyter
```

Results from run on May 10th, 2022:

Conda setup:
39 dependencies

Jupyter (with Pip):
61 dependencies

Web browser has more dependencies; with fluid/evolving web technologies.

Notebooks are not long-term solutions (see appendices of Akhlaghi+2021: arXiv:2006.03018)

```
[mohammad@akhlaghi software]$ bash Miniconda3-latest-Linux-x86_64.sh -b -p conda-install
PREFIX=/home/mohammad/tmp/software/conda-install
Unpacking payload ...
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/mohammad/tmp/software/conda-install

  added / updated specs:
    - _libgcc_mutex==0.1=main
    - _openmp_mutex==4.5=1_gnu
    - brotlipy==0.7.0=py39h27cfd23_1003
    - ca-certificates==2021.10.26=h06a4308_2
    - certifi==2021.10.8=py39h06a4308_2
    - cffi==1.15.0=py39hd667e15_1
    - charset-normalizer==2.0.4=pyhd3eb1b0_0
    - conda-content-trust==0.1.1=pyhd3eb1b0_0
    - conda-package-handling==1.7.3=py39h27cfd23_1
    - conda==4.11.0=py39h06a4308_0
    - cryptography==36.0.0=py39h9ce1e76_0
    - idna==3.3=pyhd3eb1b0_0
    - ld_impl_linux-64==2.35.1=h7274673_9
    - libffi==3.3=he6710b0_2
    - libgcc-ng==9.3.0=h5101ec6_17
    - libgomp==9.3.0=h5101ec6_17
    - libstdcxx-ng==9.3.0=hd4cf53a_17
    - ncurses==6.3=h7f8727e_2
    - openssl==1.1.1m=h7f8727e_0
    - pip==21.2.4=py39h06a4308_0
    - pycosat==0.6.3=py39h27cfd23_0
    - pycparser==2.21=pyhd3eb1b0_0
    - pyopenssl==21.0.0=pyhd3eb1b0_1
    - pysocks==1.7.1=py39h06a4308_0
    - python==3.9.7=h12debd9_1
    - readline==8.1.2=h7f8727e_1
    - requests==2.27.1=pyhd3eb1b0_0
    - ruamel_yaml==0.15.100=py39h27cfd23_0
    - setuptools==58.0.4=py39h06a4308_0
    - six==1.16.0=pyhd3eb1b0_0
    - sqlite==3.37.0=hc218d9a_0
    - tk==8.6.11=h1ccaba5_0
    - tqdm==4.62.3=pyhd3eb1b0_1
    - tzdata==2021e=hda174b7_0
    - urllib3==1.26.7=pyhd3eb1b0_0
    - wheel==0.37.1=pyhd3eb1b0_0
    - xz==5.2.5=h7b6447c_0
    - yaml==0.2.5=h7b6447c_0
    - zlib==1.2.11=h7f8727e_4
```

```
Collecting jupyterlab-pygments
  Downloading jupyterlab_pygments-0.2.2-py2.py3-none-any.whl (21 kB)
Collecting argon2-cffi-bindings
  Downloading argon2_cffi_bindings-21.2.0-cp36-abi3-manylinux_2_17_x86_64.manylin
ux2014_x86_64.whl (86 kB)
     |████████████████████████████████| 86 kB 3.1 MB/s
Requirement already satisfied: cffi>=1.0.1 in ./conda-install/lib/python3.9/site-
packages (from argon2-cffi-bindings->argon2-cffi->notebook->jupyter) (1.15.0)
Requirement already satisfied: pycparser in ./conda-install/lib/python3.9/site-pa
ckages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook->jupyter) (
2.21)
Collecting soupsieve>1.2
  Downloading soupsieve-2.3.2.post1-py3-none-any.whl (37 kB)
Collecting webencodings
  Downloading webencodings-0.5.1-py2.py3-none-any.whl (11 kB)
Collecting pyparsing!=3.0.5,>=2.0.2
  Downloading pyparsing-3.0.8-py3-none-any.whl (98 kB)
     |████████████████████████████████| 98 kB 3.4 MB/s
Collecting qtpy>=2.0.1
  Downloading QtPy-2.1.0-py3-none-any.whl (68 kB)
     |████████████████████████████████| 68 kB 2.9 MB/s
Collecting pure-eval
  Downloading pure_eval-0.2.2-py3-none-any.whl (11 kB)
Collecting asttokens
  Downloading asttokens-2.0.5-py2.py3-none-any.whl (20 kB)
Collecting executing
  Downloading executing-0.8.3-py2.py3-none-any.whl (16 kB)
Installing collected packages: traitlets, pyrsistent, attrs, wcwidth, tornado, py
zmq, python-dateutil, pure-eval, ptyprocess, parso, nest-asyncio, jupyter-core, j
sonschema, fastjsonschema, executing, entrypoints, asttokens, webencodings, stack
-data, soupsieve, pyparsing, pygments, prompt-toolkit, pickleshare, pexpect, nbfo
rmat, matplotlib-inline, MarkupSafe, jupyter-client, jedi, decorator, backcall, t
inycss2, psutil, pandocfilters, packaging, nbclient, mistune, jupyterlab-pygments
, jinja2, ipython, defusedxml, debugpy, bleach, beautifulsoup4, argon2-cffi-bindi
ngs, terminado, Send2Trash, prometheus-client, nbconvert, ipython-genutils, ipyke
rnel, argon2-cffi, notebook, widgetsnbextension, qtpy, jupyterlab-widgets, qtcons
ole, jupyter-console, ipywidgets, jupyter
Successfully installed MarkupSafe-2.1.1 Send2Trash-1.8.0 argon2-cffi-21.3.0 argon
2-cffi-bindings-21.2.0 asttokens-2.0.5 attrs-21.4.0 backcall-0.2.0 beautifulsoup4
-4.11.1 bleach-5.0.0 debugpy-1.6.0 decorator-5.1.1 defusedxml-0.7.1 entrypoints-0
.4 executing-0.8.3 fastjsonschema-2.15.3 ipykernel-6.13.0 ipython-8.3.0 ipython-g
enutils-0.2.0 ipywidgets-7.7.0 jedi-0.18.1 jinja2-3.1.2 jsonschema-4.5.1 jupyter-
1.0.0 jupyter-client-7.3.1 jupyter-console-6.4.3 jupyter-core-4.10.0 jupyterlab-p
ygments-0.2.2 jupyterlab-widgets-1.1.0 matplotlib-inline-0.1.3 mistune-0.8.4 nbcl
ient-0.6.3 nbconvert-6.5.0 nbformat-5.4.0 nest-asyncio-1.5.5 notebook-6.4.11 pack
aging-21.3 pandocfilters-1.5.0 parso-0.8.3 pexpect-4.8.0 pickleshare-0.7.5 promet
heus-client-0.14.1 prompt-toolkit-3.0.29 psutil-5.9.0 ptyprocess-0.7.0 pure-eval-
0.2.2 pygments-2.12.0 pyparsing-3.0.8 pyrsistent-0.18.1 python-dateutil-2.8.2 pyz
mq-22.3.0 qtconsole-5.3.0 qtpy-2.1.0 soupsieve-2.3.2.post1 stack-data-0.2.0 termi
nado-0.13.3 tinycss2-1.1.1 tornado-6.1 traitlets-5.2.0 wcwidth-0.2.5 webencodings
-0.5.1 widgetsnbextension-3.6.0
(base) [mohammad@akhlaghi software]$ pip install jupyter
```

Results from run on May 10th, 2022:

Conda setup: 39 dependencies

Jupyter (with Pip): 61 dependencies

Web browser has more dependencies; with fluid/evolving web technologies.

They can contain binary components.

The dependency tree (Matplotlib is *only one* dependency of Jupyter)

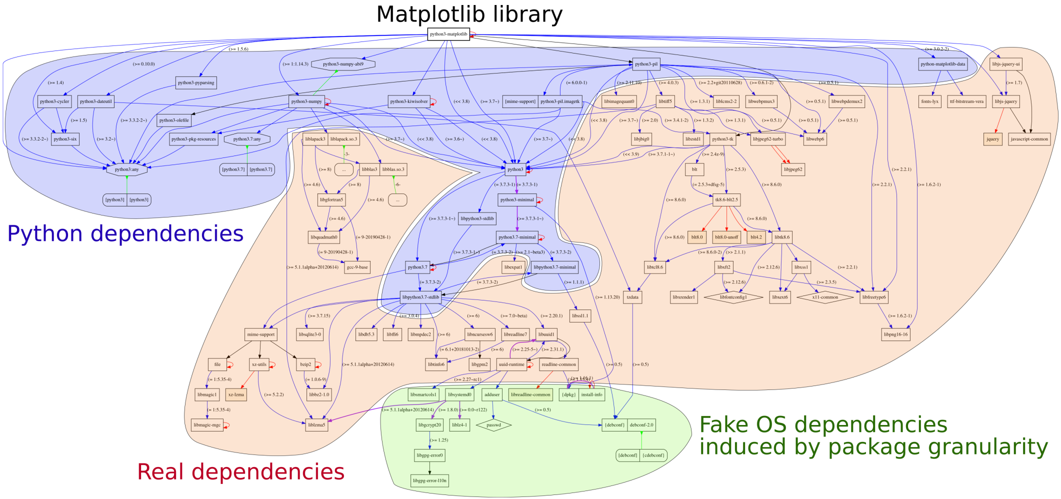


Fig. 1. Transitive dependencies of the software environment required by a simple "`import matplotlib`" command in the Python 3 interpreter.

From "Attributing and Referencing (Research) Software: Best Practices and Outlook from Inria" (Alliez et al. 2020, CiSE, DOI:10.1109/MCSE.2019.2949413).

Are containers the solution? Yes, but ... for the short term

Are containers the solution? Yes, but ... for the short term

- Containers are large (many giga-bytes)

Are containers the solution? Yes, but ... for the short term

- Containers are large (many giga-bytes)
  - Expensive to archive!

Are containers the solution? Yes, but ... for the short term

- Containers are large (many giga-bytes)
  - Expensive to archive!
  - Example: SHARE (enabling remote connection to Virtual machines with project environment):
    - 2nd place in Elsevier's "Executable paper grand challenge" of 2011.
    - SHARE's image repository was taken offline in 2019!
    - Even the challenge webpage is no longer available: http://www.executablepapers.com

Are containers the solution? Yes, but ... for the short term

- Containers are large (many giga-bytes)
  - Expensive to archive!
  - Example: SHARE (enabling remote connection to Virtual machines with project environment):
    - 2nd place in Elsevier's "Executable paper grand challenge" of 2011.
    - SHARE's image repository was taken offline in 2019!
    - Even the challenge webpage is no longer available: http://www.executablepapers.com
- Container are binary (tailored to certain kernels+CPUs)

Are containers the solution? Yes, but ... for the short term

- Containers are large (many giga-bytes)
  - Expensive to archive!
  - Example: SHARE (enabling remote connection to Virtual machines with project environment):
    - 2nd place in Elsevier's "Executable paper grand challenge" of 2011.
    - SHARE's image repository was taken offline in 2019!
    - Even the challenge webpage is no longer available: http://www.executablepapers.com
- Container are binary (tailored to certain kernels+CPUs)
  - Only guarantee the Long Term Release kernels.
    - Become un-readable, multi-gigabyte binary blobs in $\sim$ 10 years!
    - Even if you store them on Zenodo!

Are containers the solution? Yes, but … for the short term

- Containers are large (many giga-bytes)
  - Expensive to archive!
  - Example: SHARE (enabling remote connection to Virtual machines with project environment):
    - 2nd place in Elsevier's "Executable paper grand challenge" of 2011.
    - SHARE's image repository was taken offline in 2019!
    - Even the challenge webpage is no longer available: http://www.executablepapers.com
- Container are binary (tailored to certain kernels+CPUs)
  - Only guarantee the Long Term Release kernels.
    - Become un-readable, multi-gigabyte binary blobs in $\sim$ 10 years!
    - Even if you store them on Zenodo!
  - Only on common CPUs architectures.

Are containers the solution? Yes, but ... for the short term

- Containers are large (many giga-bytes)
  - Expensive to archive!
  - Example: SHARE (enabling remote connection to Virtual machines with project environment):
    - 2nd place in Elsevier's "Executable paper grand challenge" of 2011.
    - SHARE's image repository was taken offline in 2019!
    - Even the challenge webpage is no longer available: http://www.executablepapers.com
- Container are binary (tailored to certain kernels+CPUs)
  - Only guarantee the Long Term Release kernels.
    - Become un-readable, multi-gigabyte binary blobs in $\sim$ 10 years!
    - Even if you store them on Zenodo!
  - Only on common CPUs architectures.
- Containers themselves are hard to reproduce.
  - Example: 2020CSE....22a.102M use '`FROM ubuntu:16.04`', but if run today, images are from 2021.

For longevity issues with Jupyter, Conda, Containers and etc ...

As well as a survey of depreciated/abandoned/lost solutions since the 1990s ...

... see the appendices in arXiv:2006.03018

Our solution: CiSE 23 (3), pp 82-91: DOI:10.1109/MCSE.2021.3072860, arXiv:2006.03018

EDITORS: Lorena A. Barba, labarba@gwu.edu
Sandra Gesing, sandra.gesing@nd.edu

SPECIAL TRACK: REPRODUCIBLE RESEARCH

Toward Long-Term and Archivable Reproducibility

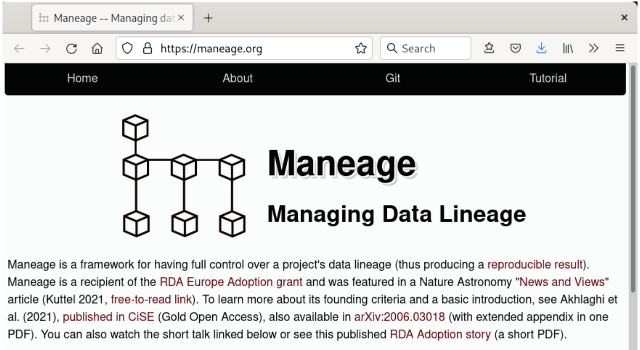
Mohammad Akhlaghi, *Instituto de Astrofísica de Canarias, La Laguna, Tenerife, 38205, Spain*
Raúl Infante-Sainz, *Universidad de La Laguna, La Laguna, Tenerife, 38205, Spain*
Boudewijn F. Roukema, *Nicolaus Copernicus University, Torun 87-100, Poland*
Mohammadreza Khellat, *Ideal-Information, PC 133 Al Khuwair, Muscat, Oman*
David Valls-Gabaud, *Paris Observatory, Paris 75014, France*
Roberto Baena-Gallé, *Universidad Internacional de La Rioja, Logroño 26006, Spain*

Analysis pipelines commonly use high-level technologies that are popular when created, but are unlikely to be readable, executable, or sustainable in the long term. A set of criteria is introduced to address this problem: completeness (no execution requirement beyond a minimal Unix-like operating system, no administrator privileges, no network connection, and storage primarily in plain text); modular design; minimal complexity; scalability; verifiable inputs and outputs; version control; linking analysis with narrative; and free and open-source software. As a proof of concept, we introduce "Maneage" (managing data lineage), enabling cheap archiving, provenance extraction, and peer verification that has been tested in several research publications. We show that longevity is a realistic requirement that does not sacrifice immediate or short-term reproducibility. The caveats (with proposed solutions) are then discussed and we conclude with the benefits for the various stakeholders. This article is itself a Maneage'd project (project commit 313db0b). Appendices—*Two comprehensive appendices that review the longevity of existing solutions are available as supplementary "Web extras," which are available in the IEEE Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/MCSE.2021.3072860.* Reproducibility—*All products available in* `zenodo.4913277`*, the Git history of this paper's source is at* `git.maneage.org/paper-concept.git`*, which is also archived in Software Heritage: swh:1:dir:33fea87068c1612daf011f161b97787b9a0df39f. Clicking on the SWHIDs in the digital format will provide more "context" for same content.*

Reproducible research has been discussed in the sciences for at least 30 years.[1,2] Many reproducible workflow solutions (hereafter, "solutions") have been proposed, which mostly rely on the common technology of the day, starting with Make and Matlab libraries in the 1990s, Java in the 2000s, and mostly shifting to Python during the past decade.

However, these technologies develop fast, e.g., code written in Python 2 (which is no longer officially maintained) often cannot run with Python 3. The cost of staying up to date within this rapidly evolving landscape is high. Scientific projects, in particular, suffer the most: Scientists have to focus on their own research domain, but to some degree, they need to understand the technology of their tools because it determines their results

Digital Object Identifier 10.1109/MCSE.2021.3072860
Date of publication 13 April 2021; date of current version 15 June 2021.



https://maneage.org

Recognition 1: RDA adoption grant (2019) to IAC for Maneage

For Maneage, the IAC is selected as a Top European organization funded to adopt RDA Recommendations and Outputs.

- Research Data Alliance was launched by the European Commission, NSF, National Institute of Standards and Technology, and the Australian Government's Department of Innovation.
- RDA Outputs are the technical and social infrastructure solutions developed by RDA Working Groups or Interest Groups that enable data sharing, exchange, and interoperability.

Recognition 2: "News and Views" in Nature Astronomy (DOI:10.1038/s41550-021-01402-3)

news & views

Check for updates

REPRODUCIBILITY

No expiration date

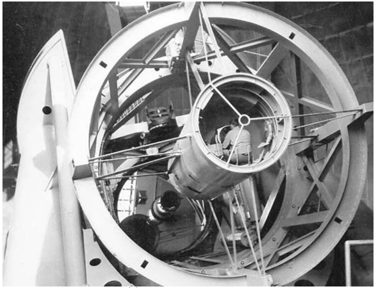
The short lifespan of software puts a time limit on the reproducibility of computational research. To extend software longevity, guidelines and tools to preserve scientific workflows and analysis are helpful, but the challenge is to get researchers to use them.

Michelle M. Kuttel

Free-to-read link: https://rdcu.be/cmYVx

Replicability (hardware/statistical)

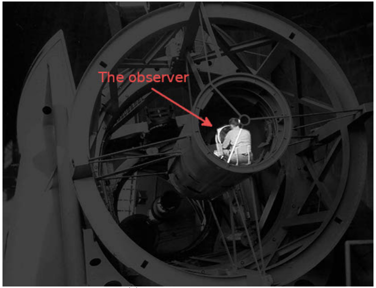
- Involves data collection.
- Inherently includes measurements errors (can never be exactly reproduced).
- Example: Raw telescope image/spectra.
- **NOT DISCUSSED HERE.**



http://slittlefair.staff.shef.ac.uk

Replicability (hardware/statistical)

- Involves data collection.
- Inherently includes measurements errors (can never be exactly reproduced).
- Example: Raw telescope image/spectra.
- **NOT DISCUSSED HERE.**



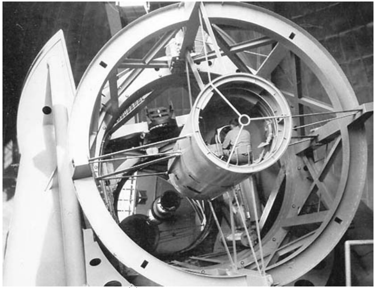
http://slittlefair.staff.shef.ac.uk

Definitions & Clarification

(from the National Academies report in 2019, DOI:10.17226/25303)

Replicability (hardware/statistical)

- Involves data collection.
- Inherently includes measurements errors (can never be exactly reproduced).
- Example: Raw telescope image/spectra.
- **NOT DISCUSSED HERE.**



http://slittlefair.staff.shef.ac.uk

Reproducibility (Software/Deterministic)

- Involves data analysis, or simulations.
- Starts after data is collected/digitized.
- Example: $2+2=4$ (i.e., sum of datasets).
- **DISCUSSED HERE.**

Wikimedia Commons

Founding criteria

Basic/simple principle:

Science is defined by its METHOD, not its result.

Founding criteria

Basic/simple principle:

Science is defined by its METHOD, not its result.

- **Complete/self-contained:**
  - Only dependency should be POSIX tools (discards Conda or Jupyter which need Python).
  - Must not require root permissions (discards tools like Docker or Nix/Guix).
  - Should be non-interactive or runnable in batch (user interaction is an incompleteness).
  - Should be usable without internet connection.

Founding criteria

Basic/simple principle:

Science is defined by its METHOD, not its result.

- **Complete/self-contained:**
  - Only dependency should be POSIX tools (discards Conda or Jupyter which need Python).
  - Must not require root permissions (discards tools like Docker or Nix/Guix).
  - Should be non-interactive or runnable in batch (user interaction is an incompleteness).
  - Should be usable without internet connection.
- **Modularity:** Parts of the project should be re-usable in other projects.

Founding criteria

Basic/simple principle:

Science is defined by its METHOD, not its result.

- **Complete/self-contained:**
  - Only dependency should be POSIX tools (discards Conda or Jupyter which need Python).
  - Must not require root permissions (discards tools like Docker or Nix/Guix).
  - Should be non-interactive or runnable in batch (user interaction is an incompleteness).
  - Should be usable without internet connection.
- **Modularity:** Parts of the project should be re-usable in other projects.
- **Plain text:** Project's source should be in plain-text (binary formats need special software)
  - This includes high-level analysis.
  - It is easily publishable (very low volume of $\times$100KB), archivable, and parse-able.
  - Version control (e.g., with Git) can track project's history.

Founding criteria

Basic/simple principle:

Science is defined by its METHOD, not its result.

- **Complete/self-contained:**
  - Only dependency should be POSIX tools (discards Conda or Jupyter which need Python).
  - Must not require root permissions (discards tools like Docker or Nix/Guix).
  - Should be non-interactive or runnable in batch (user interaction is an incompleteness).
  - Should be usable without internet connection.
- **Modularity:** Parts of the project should be re-usable in other projects.
- **Plain text:** Project's source should be in plain-text (binary formats need special software)
  - This includes high-level analysis.
  - It is easily publishable (very low volume of $\times$100KB), archivable, and parse-able.
  - Version control (e.g., with Git) can track project's history.
- **Minimal complexity:** Occum's rasor: "Never posit pluralities without necessity".
  - Avoiding the fashionable tool of the day: tomorrow another tool will take its place!
  - Easier learning curve, also doesn't create a generational gap.
  - Is compatible and extensible.

Founding criteria

Basic/simple principle:

Science is defined by its METHOD, not its result.

- **Complete/self-contained:**
  - Only dependency should be POSIX tools (discards Conda or Jupyter which need Python).
  - Must not require root permissions (discards tools like Docker or Nix/Guix).
  - Should be non-interactive or runnable in batch (user interaction is an incompleteness).
  - Should be usable without internet connection.
- **Modularity:** Parts of the project should be re-usable in other projects.
- **Plain text:** Project's source should be in plain-text (binary formats need special software)
  - This includes high-level analysis.
  - It is easily publishable (very low volume of $\times$100KB), archivable, and parse-able.
  - Version control (e.g., with Git) can track project's history.
- **Minimal complexity:** Occum's rasor: "Never posit pluralities without necessity".
  - Avoiding the fashionable tool of the day: tomorrow another tool will take its place!
  - Easier learning curve, also doesn't create a generational gap.
  - Is compatible and extensible.
- **Verifable inputs and outputs:** Inputs and Outputs must be automatically verified.

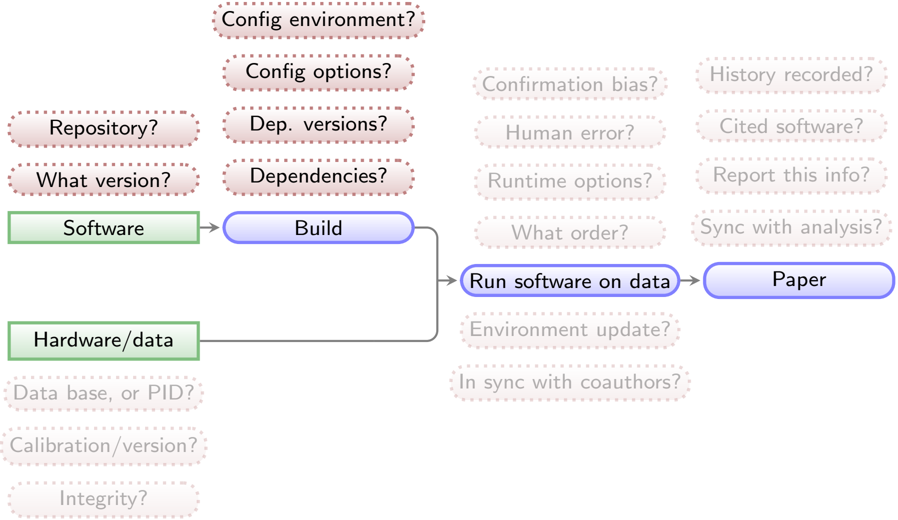
Founding criteria

Basic/simple principle:

Science is defined by its METHOD, not its result.

- **Complete/self-contained:**
  - Only dependency should be POSIX tools (discards Conda or Jupyter which need Python).
  - Must not require root permissions (discards tools like Docker or Nix/Guix).
  - Should be non-interactive or runnable in batch (user interaction is an incompleteness).
  - Should be usable without internet connection.
- **Modularity:** Parts of the project should be re-usable in other projects.
- **Plain text:** Project's source should be in plain-text (binary formats need special software)
  - This includes high-level analysis.
  - It is easily publishable (very low volume of $\times$100KB), archivable, and parse-able.
  - Version control (e.g., with Git) can track project's history.
- **Minimal complexity:** Occum's rasor: "Never posit pluralities without necessity".
  - Avoiding the fashionable tool of the day: tomorrow another tool will take its place!
  - Easier learning curve, also doesn't create a generational gap.
  - Is compatible and extensible.
- **Verifable inputs and outputs:** Inputs and Outputs must be automatically verified.
- **Free and open source software:** Free software is essential: non-free software is not configurable, not distributable, and dependent on non-free provider (which may discontinue it in N years).

General outline of a project (after data collection)



Green boxes with sharp corners: *source*/input components/files.
Blue boxes with rounded corners: *built* components.
Red boxes with dashed borders: questions that must be clarified for each phase.

Predefined/exact software tools

Reproducibility & software

Reproducing the environment (specific software versions, build instructions and dependencies) is also critically important for reproducibility.

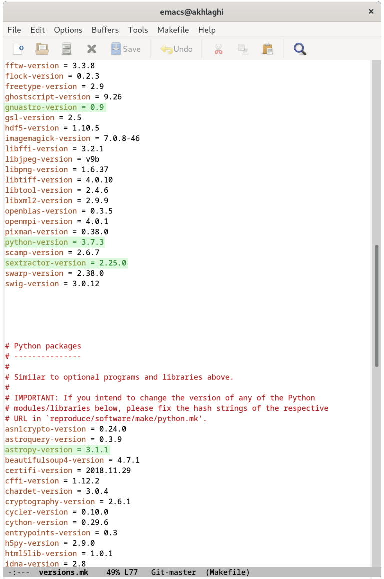
- *Containers* or *Virtual Machines* are a binary black box.
  - e.g., with ‘`FROM ubuntu:16.04`’ (released in April 2016),
  - in a `Dockerfile`, the OS image will come from (updated monthly!): `https://partner-images.canonical.com/core/xenial`
- Maneage installs fixed versions of all necessary research software.
  - Including their dependencies.
  - All the way down to the C compiler.
- Installs similar environment on GNU/Linux, or macOS systems.
- Works like a package manager (e.g., `apt`, `brew` or Conda).
  - ... but (!), its not a third party package manager.
  - Build instructions are within same analysis project.
  - e.g., see Conda’s build of Gnuastro (its gets updated behind your back): `https://anaconda.org/conda-forge/gnuastro/files`
- Source code of all software in Maneage is archived on zenodo.3883409.

Predefined/exact software tools

Reproducibility & software

Reproducing the environment (specific software versions, build instructions and dependencies) is also critically important for reproducibility.

- *Containers* or *Virtual Machines* are a binary black box.
  - e.g., with '`FROM ubuntu:16.04`' (released in April 2016),
  - in a `Dockerfile`, the OS image will come from (updated monthly!): `https://partner-images.canonical.com/core/xenial`
- Maneage installs fixed versions of all necessary research software.
  - Including their dependencies.
  - All the way down to the C compiler.
- Installs similar environment on GNU/Linux, or macOS systems.
- Works like a package manager (e.g., `apt`, `brew` or Conda).
  - ... but (!), its not a third party package manager.
  - Build instructions are within same analysis project.
  - e.g., see Conda's build of Gnuastro (its gets updated behind your back): `https://anaconda.org/conda-forge/gnuastro/files`
- Source code of all software in Maneage is archived on zenodo.3883409.



Advantages of this build system

- Project runs in fixed/controlled environment: custom build of Bash, Make, GNU Coreutils (`ls`, `cp`, `mkdir` and etc), AWK, or SED, LaTeX, etc.
- No need for root/administrator permissions (on servers or super computers).
- Whole system is built automatically on any Unix-like operating system (less 2 hours).
- Dependencies of different projects will not conflict.
- Everything in plain text (human & computer readable/archivable).

https://natemowry2.wordpress.com

Software citation automatically generated in paper (including Astropy)

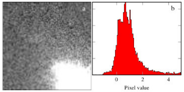


Figure 2: (a) An example image of the Wide-Field Planetary Camera 2, on board the Hubble Space Telescope from 1993 to 2009. This is one of the sample images from the FITS standard webpage, kept as examples for this file format. (b) Histogram of pixel values in (a).

removes the necessity to add further dependencies (to create the plots) to your project. There are high-level language libraries like Matplotlib which also generate plots. However, the problem is that they require many dependencies (Python, Numpy and etc). Installing these dependencies from source, is not easy and will harm the reproducibility of your paper. Note that after several years, the binary files of these high-level libraries, that you easily install today, will no longer be available in common repositories. Therefore building the libraries from source is the only option to reproduce your results.

Furthermore, since PGFPlots is built by LaTeX it respects all the properties of your text (for example line width and fonts and etc). Therefore the final plot blends in your paper much more nicely. It also has a wonderful manual[5].

This template also defines two LaTeX macros that allow you to mark text within your document as *new* and *notes*. For example, this text has been marked as `new`. If you comment the line (by adding a '`%`' at the start of the line or simply deleting the line) that defines `highlightchanges`, then the one that was marked `new` will become black (totally blend in with the rest of the text) and the one marked `tonote` will not be in the final PDF. You can thus use `highlightchanges` to easily make copies of your research for existing coauthors (who are just interested in the new parts or notes) and new co-authors (who don't want to be distracted by these issues in their first time reading).

2. NOTICE AND CITATIONS

To encourage other scientists to publish similarly reproducible papers, please add a notice close to the start of your paper or in the end of the abstract clearly mentioning that your work is fully reproducible.

For the time being, we haven't written a specific paper only for this template. Until then, we would be grateful if you could cite the first paper that used the early versions of this template: Akhlaghi and Ichikawa (2015).

After publication, don't forget to upload all the necessary data, software source code and the project's source to a long-lasting host like Zenodo (https://zenodo.org).

[5] http://mirrors.ctan.org/graphics/pgf/contrib/pgfplots/doc/pgfplots.pdf

3. ACKNOWLEDGEMENTS

Please include the following two paragraphs in the Acknowledgement section of your paper. This reproducible paper template was developed in parallel with Gnuastro, so it benefited from the same grants. If you don't use Gnuastro in your final/customized project, please remove it from the paragraph below, only mentioning the reproducible paper template.

This research was partly done using GNU Astronomy Utilities (Gnuastro, ascl.net/1801.009), and the reproducible paper template v0-364-g268dfc0-dirty. Work on Gnuastro and the reproducible paper template has been funded by the Japanese Ministry of Education, Culture, Sports, Science, and Technology (MEXT) scholarship and its Grant-in-Aid for Scientific Research (21244012, 24253003), the European Research Council (ERC) advanced grant 339659-MUSICOS, European Union's Horizon 2020 research and innovation programme under Marie Sklodowska-Curie grant agreement No 721463 to the SUNDIAL ITN, and from the Spanish Ministry of Economy and Competitiveness (MINECO) under grant number AYA2016-76219-P.

This research was done with the following free software programs and libraries: Bzip2 1.0.6, CFITSIO 3.45, CMake 3.14.2, cURL 7.63.0, Discoteq flock 0.2.3, File 5.36, FreeType 2.9, Git 2.21.0, GNU Astronomy Utilities 0.9 (Akhlaghi and Ichikawa 2015), GNU AWK 5.0.0, GNU Bash 5.0.7, GNU Binutils 2.32, GNU Compiler Collection (GCC) 9.1.0, GNU Coreutils 8.31, GNU Diffutils 3.7, GNU Findutils 4.6.0.199-e3fc, GNU Grep 3.3, GNU Gzip 1.10, GNU Integer Set Library 0.18, GNU Libtool 2.4.6, GNU M4 1.4.18, GNU Make 4.2.90, GNU Multiple Precision Arithmetic Library 6.1.2, GNU Multiple Precision Complex library, GNU Multiple Precision Floating-Point Reliably 4.0.2, GNU NCURSES 6.1, GNU Readline 8.0, GNU Scientific Library 2.5, GNU Sed 4.7, GNU Tar 1.32, GNU Wget 1.20.3, GNU Which 2.21, GPL Ghostscript 9.26, HDF5 library 1.10.5, ImageMagick 7.0.8-46, Libbsd 0.9.1, Libgit2 0.26.0, Libjpeg v9b, Libpng 1.6.37, Libtiff 4.0.10, Lzip 1.20, Metastore (forked) 1.1.2-23-fa9170b, OpenBLAS 0.3.5, Open MPI 4.0.1, OpenSSL 1.1.1a, PatchELF 0.9, pkg-config 0.29.2, Python 3.7.3, Unzip 6.0, WCSLIB 6.2, XZ Utils 5.2.4, Zip 3.0 and Zlib 1.2.11. Within Python, the following modules were used: Astropy 3.1.1 (Astropy Collaboration et al. 2013; Astropy Collaboration et al. 2018), Cycler 0.10.0, Cython 0.29.6 (Behnel et al. 2011), h5py 2.9.0, Kiwisolver 1.0.1, Matplotlib 3.0.2 (Hunter 2007), Numpy 1.16.2 (van der Walt et al. 2011), pkgconfig 1.5.1, PyParsing 2.3.1, python-dateutil 2.8.0, Scipy 1.2.1 (Oliphant 2007; Millman and Aivazis 2011), Setuptools 40.8.0, Setuptools-scm 3.2.0 and Six 1.12.0. The LaTeX source of the paper was compiled to make the PDF using the following packages: biber 2.12, biber 2.12, biblatex 3.12, biblatex 3.12, caption 2018-10-05, caption 2018-10-05, courier 2016-06-24, courier 2016-06-24, csquotes 5.2d, datetime 2.60, datetime 2.60, ec 1.0, ec 1.0, etoolbox 2.5f, etoolbox 2.5f, fancyhdr 3.10, fancyhdr 3.10, fmtcount 3.05, fmtcount 3.05, fontaxes 1.0d, fontaxes 1.0d, footmisc 5.5b, footmisc 5.5b, fp 2.1d, fp 2.1d, logreq 1.0, logreq 1.0, newtx 1.554, newtx 1.554, pgf 3.1.2, pgf 3.1.2, pgfplots 1.16, pgfplots 1.16, preprint 2011, preprint 2011, setspace 6.7a, setspace 6.7a, tex 3.14159265, tex 3.14159265, texgyre 2.501, texgyre 2.501, times 2016-06-24, times 2016-06-24, titlesec 2.10.2, titlesec 2.10.2, txfonts 2016-

06-24, txfonts 2016-06-24, ulem 2016-06-24, ulem 2016-06-24, xcolor 2.12, xcolor 2.12, xkeyval 2.7a and xkeyval 2.7a. We are very grateful to all their creators for freely providing this necessary infrastructure. This research (and many others) would not be possible without them.

References

Akhlaghi, M. and T. Ichikawa (Sept. 2015). *ApJS*, 220, 1.
Astropy Collaboration et al. (Oct. 2013). *A&A*, 558, A33.
Astropy Collaboration et al. (Sept. 2018). *AJ*, 156, 123.
Bacon, R. et al. (Nov. 2017). *A&A*, 608, A1.
Behnel, S. et al. (Mar. 2011). *CiSE*, 13, 31.
Hunter, J. D. (2007). *CiSE*, 9, 90.
Millman, K. J. and M. Aivazis (Mar. 2011). *CiSE*, 13, 9.
Oliphant, T. E. (May 2007). *CiSE*, 9, 10.
van der Walt, S. et al. (Mar. 2011). *CiSE*, 13, 22.

Software citation automatically generated in paper (including Astropy)

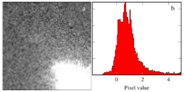


Figure 2: (a) An example image of the Wide-Field Planetary Camera 2, on board the Hubble Space Telescope from 1993 to 2009. This is one of the sample images from the FITS standard webpage, kept as examples for this file format. (b) Histogram of pixel values in (a).

removes the necessity to add further dependencies (to create the plots) to your project. There are high-level language libraries like Matplotlib which also generate plots. However, the problem is that they require many dependencies (Python, Numpy and etc). Installing these dependencies from source, is not easy and will harm the reproducibility of your paper. Note that after several years, the binary files of these high-level libraries, that you easily install today, will no longer be available in common repositories. Therefore building the libraries from source is the only option to reproduce your results.

Furthermore, since PGFPlots is built by LaTeX it respects all the properties of your text (for example line width and fonts and etc). Therefore the final plot blends in your paper much more nicely. It also has a wonderful manual[5].

This template also defines two LaTeX macros that allow you to mark text within your document as *new* and *notes*. For example, this text has been marked as `new`. If you comment the line (by adding a '`%`' at the start of the line or simply deleting the line) that defines `highlightchanges`, then the one that was marked `new` will become black (totally blend in with the rest of the text) and the one marked `tonote` will not be in the final PDF. You can thus use `highlightchanges` to easily make copies of your research for existing coauthors (who are just interested in the new parts or notes) and new co-authors (who don't want to be distracted by these issues in their first time reading).

2. NOTICE AND CITATIONS

To encourage other scientists to publish similarly reproducible papers, please add a notice close to the start of your paper or in the end of the abstract clearly mentioning that your work is fully reproducible.

For the time being, we haven't written a specific paper only for this template. Until then, we would be grateful if you could cite the first paper that used the early versions of this template: Akhlaghi and Ichikawa (2015).

After publication, don't forget to upload all the necessary data, software source code and the project's source to a long-lasting host like Zenodo (https://zenodo.org).

[5] http://mirrors.ctan.org/graphics/pgf/contrib/pgfplots/doc/pgfplots.pdf

3. ACKNOWLEDGEMENTS

Please include the following two paragraphs in the Acknowledgement section of your paper. This reproducible paper template was developed in parallel with Gnuastro, so it benefited from the same grants. If you don't use Gnuastro in your final/customized project, please remove it from the paragraph below, only mentioning the reproducible paper template.

This research was partly done using GNU Astronomy Utilities (Gnuastro, ascl.net/1801.009), and the reproducible paper template v0-364-g268dfc0-dirty. Work on Gnuastro and the reproducible paper template has been funded by the Japanese Ministry of Education, Culture, Sports, Science, and Technology (MEXT) scholarship and its Grant-in-Aid for Scientific Research (21244012, 24253003), the European Research Council (ERC) advanced grant 339659-MUSICOS, European Union's Horizon 2020 research and innovation programme under Marie Sklodowska-Curie grant agreement No 721463 to the SUNDIAL ITN, and from the Spanish Ministry of Economy and Competitiveness (MINECO) under grant number AYA2016-76219-P.

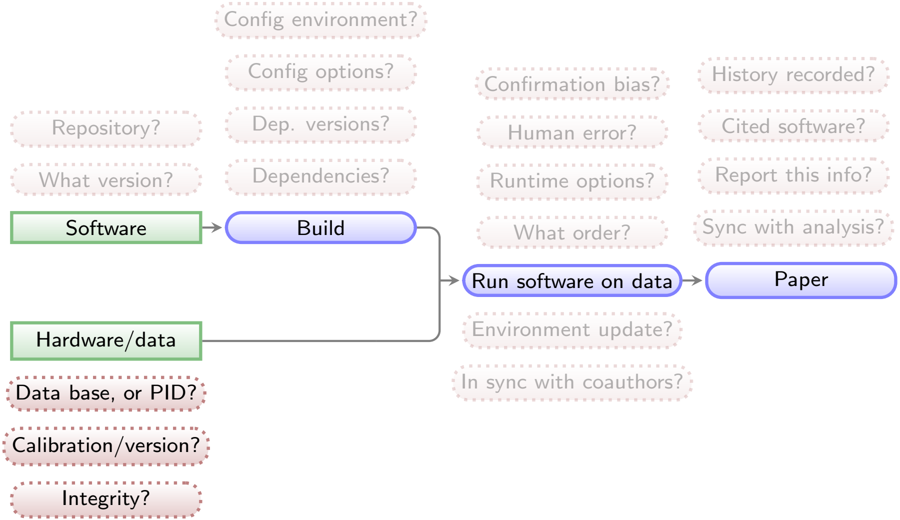
This research was done with the following free software programs and libraries: Bzip2 1.0.6, CFITSIO 3.45, CMake 3.14.2, cURL 7.63.0, Discoteq flock 0.2.3, File 5.36, FreeType 2.9, Git 2.21.0, GNU Astronomy Utilities 0.9 (Akhlaghi and Ichikawa 2015), GNU AWK 5.0.0, GNU Bash 5.0.7, GNU Binutils 2.32, GNU Compiler Collection (GCC) 9.1.0, GNU Coreutils 8.31, GNU Diffutils 3.7, GNU Findutils 4.6.0.199-e3fc, GNU Grep 3.3, GNU Gzip 1.10, GNU Integer Set Library 0.18, GNU Libtool 2.4.6, GNU M4 1.4.18, GNU Make 4.2.90, GNU Multiple Precision Arithmetic Library 6.1.2, GNU Multiple Precision Complex library, GNU Multiple Precision Floating-Point Reliably 4.0.2, GNU NCURSES 6.1, GNU Readline 8.0, GNU Scientific Library 2.5, GNU Sed 4.7, GNU Tar 1.32, GNU Wget 1.20.3, GNU Which 2.21, GPL Ghostscript 9.26, HDF5 library 1.10.5, ImageMagick 7.0.8-46, Libbsd 0.9.1, Libgit2 0.26.0, Libjpeg v9b, Libpng 1.6.37, Libtiff 4.0.10, Lzip 1.20, Metastore (forked) 1.1.2-23-fa9170b, OpenBLAS 0.3.5, Open MPI 4.0.1, OpenSSL 1.1.1a, PatchELF 0.9, pkg-config 0.29.2, Python 3.7.3, Unzip 6.0, WCSLIB 6.2, XZ Utils 5.2.4, Zip 3.0 and Zlib 1.2.11. Within Python, the following modules were used: Astropy 3.1.1 (Astropy Collaboration et al. 2013; Astropy Collaboration et al. 2018), Cycler 0.10.0, Cython 0.29.6 (Behnel et al. 2011), h5py 2.9.0, Kiwisolver 1.0.1, Matplotlib 3.0.2 (Hunter 2007), Numpy 1.16.2 (van der Walt et al. 2011), pkgconfig 1.5.1, PyParsing 2.3.1, python-dateutil 2.8.0, Scipy 1.2.1 (Oliphant 2007; Millman and Aivazis 2011), Setuptools 40.8.0, Setuptools-scm 3.2.0 and Six 1.12.0. The LaTeX source of the paper was compiled to make the PDF using the following packages: biber 2.12, biber 2.12, biblatex 3.12, biblatex 3.12, caption 2018-10-05, caption 2018-10-05, courier 2016-06-24, courier 2016-06-24, csquotes 5.2d, datetime 2.60, datetime 2.60, ec 1.0, ec 1.0, etoolbox 2.5f, etoolbox 2.5f, fancyhdr 3.10, fancyhdr 3.10, fmtcount 3.05, fmtcount 3.05, fontaxes 1.0d, fontaxes 1.0d, footmisc 5.5b, footmisc 5.5b, fp 2.1d, fp 2.1d, logreq 1.0, logreq 1.0, newtx 1.554, newtx 1.554, pgf 3.1.2, pgf 3.1.2, pgfplots 1.16, pgfplots 1.16, preprint 2011, preprint 2011, setspace 6.7a, setspace 6.7a, tex 3.14159265, tex 3.14159265, texgyre 2.501, texgyre 2.501, times 2016-06-24, times 2016-06-24, titlesec 2.10.2, titlesec 2.10.2, txfonts 2016-

06-24, txfonts 2016-06-24, ulem 2016-06-24, ulem 2016-06-24, xcolor 2.12, xcolor 2.12, xkeyval 2.7a and xkeyval 2.7a. We are very grateful to all their creators for freely providing this necessary infrastructure. This research (and many others) would not be possible without them.

References

Akhlaghi, M. and T. Ichikawa (Sept. 2015). *ApJS*, 220, 1.
Astropy Collaboration et al. (Oct. 2013). *A&A*, 558, A33.
Astropy Collaboration et al. (Sept. 2018). *AJ*, 156, 123.
Bacon, R. et al. (Nov. 2017). *A&A*, 608, A1.
Behnel, S. et al. (Mar. 2011). *CiSE*, 13, 31.
Hunter, J. D. (2007). *CiSE*, 9, 90.
Millman, K. J. and M. Aivazis (Mar. 2011). *CiSE*, 13, 9.
Oliphant, T. E. (May 2007). *CiSE*, 9, 10.
van der Walt, S. et al. (Mar. 2011). *CiSE*, 13, 22.

General outline of a project (after data collection)



Green boxes with sharp corners: *source*/input components/files.
Blue boxes with rounded corners: *built* components.
Red boxes with dashed borders: questions that must be clarified for each phase.

Input data source and integrity is documented and checked

Stored information about each input file:

- PID (where available).
- Download URL.
- MD5-sum to check integrity.

All inputs are downloaded from the given PID/URL when necessary (during the analysis).

MD5-sums are checked to make sure the download was done properly or the file is the same (hasn't changed on the server/source).

Example from the reproducible paper arXiv:1909.11230.
This paper needs three input files (two images, one catalog).


```
# Input files necessary for this project.
#
# This file is read by the configure script and running Makefiles.
#
# Copyright (C) 2018-2019 Mohammad Akhlaghi <mohammad@akhlaghi.org>
#
# Copying and distribution of this file, with or without modification, are
# permitted in any medium without royalty provided the copyright notice and
# this notice are preserved.  This file is offered as-is, without any
# warranty.


M51SDSSRURL   = https://dr12.sdss.org/sas/dr12/boss/photoObj/frames/301/3716/6
M51SDSSRIMAGE = frame-r-003716-6-0117.fits.bz2
M51SDSSRMD5   = 965da8bd861e94a9701521a11b2d80aa
M51SDSSRSIZE  = 2.8M

XDFF755WURL   = http://archive.stsci.edu/pub/hlsp/xdf
XDFF755WIMAGE = hlsp_xdf_hst_acswfc-60mas_hudf_f775w_v1_sci.fits
XDFF755WMD5   = 81408ed0949bd3a93c4bfe7e229472e6
XDFF775WSIZE  = 106M

UVUDFSEGURL   = https://asd.gsfc.nasa.gov/UVUDF
UVUDFSEGIMAGE = segmentation_map_rafelski_2015.fits.gz
UVUDFSEGMD5   = 29d5b3e5311b77512baf27db6ad0e11b
UVUDFSEGSIZE  = 1.3M
```

Input data source and integrity is documented and checked

Stored information about each input file:

- PID (where available).
- Download URL.
- MD5-sum to check integrity.

All inputs are downloaded from the given PID/URL when necessary (during the analysis).

MD5-sums are checked to make sure the download was done properly or the file is the same (hasn't changed on the server/source).

Example from the reproducible paper arXiv:1909.11230.
This paper needs three input files (two images, one catalog).

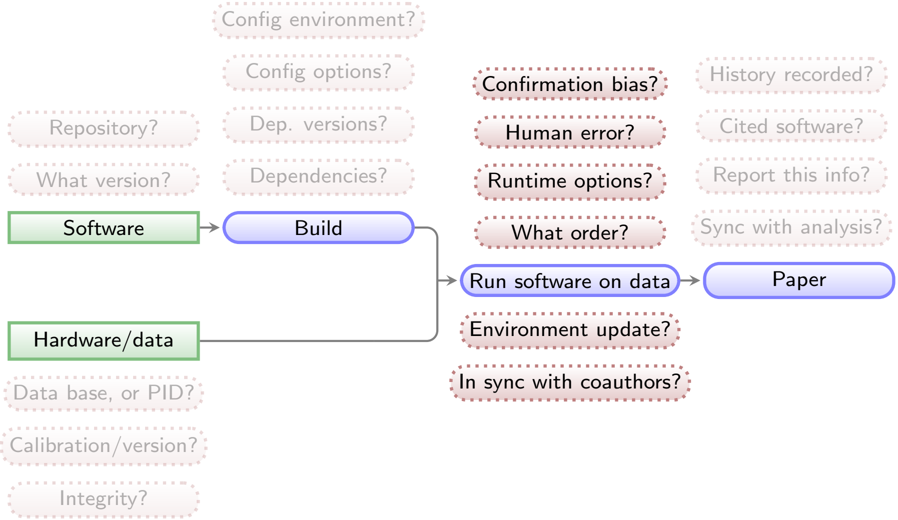
emacs@akhlaghi

File Edit Options Buffers Tools Makefile Help

```
# Input files necessary for this project.
#
# This file is read by the configure script and running Makefiles.
#
# Copyright (C) 2018-2019 Mohammad Akhlaghi <mohammad@akhlaghi.org>
#
# Copying and distribution of this file, with or without modification, are
# permitted in any medium without royalty provided the copyright notice and
# this notice are preserved.  This file is offered as-is, without any
# warranty.


M51SDSSRURL   = https://dr12.sdss.org/sas/dr12/boss/photoObj/frames/301/3716/6
M51SDSSRIMAGE = frame-r-003716-6-0117.fits.bz2
M51SDSSRMD5   = 965da8bd861e94a9701521a11b2d80aa
M51SDSSRSIZE  = 2.8M

XDFF755WURL   = http://archive.stsci.edu/pub/hlsp/xdf
XDFF755WIMAGE = hlsp_xdf_hst_acswfc-60mas_hudf_f775w_v1_sci.fits
XDFF755WMD5   = 81408ed0949bd3a93c4bfe7e229472e6
XDFF775WSIZE  = 106M

UVUDFSEGURL   = https://asd.gsfc.nasa.gov/UVUDF
UVUDFSEGIMAGE = segmentation_map_rafelski_2015.fits.gz
UVUDFSEGMD5   = 29d5b3e5311b77512baf27db6ad0e11b
UVUDFSEGSIZE  = 1.3M
```

-:--- INPUTS.mk All L1 Git-master (GNUmakefile)
For information about GNU Emacs and the GNU system, type C-h C-a.

General outline of a project (after data collection)



Green boxes with sharp corners: *source*/input components/files.
Blue boxes with rounded corners: *built* components.
Red boxes with dashed borders: questions that must be clarified for each phase.

Reproducible science: Maneage is managed through a Makefile

All steps (downloading and analysis) are managed by Makefiles (example from zenodo.1164774):

- Unlike a script which always starts from the top, a Makefile starts from the end and steps that don't change will be left untouched (not remade).
- A single *rule* can manage any number of files.
- Make can identify independent steps internally and do them in parallel.
- Make was designed for complex projects with thousands of files (all major Unix-like components), so it is highly evolved and efficient.
- Make is a very simple and small language, thus easy to learn with great and free documentation (for example GNU Make's manual).

Reproducible science: Maneage is managed through a Makefile

All steps (downloading and analysis) are managed by Makefiles (example from zenodo.1164774):

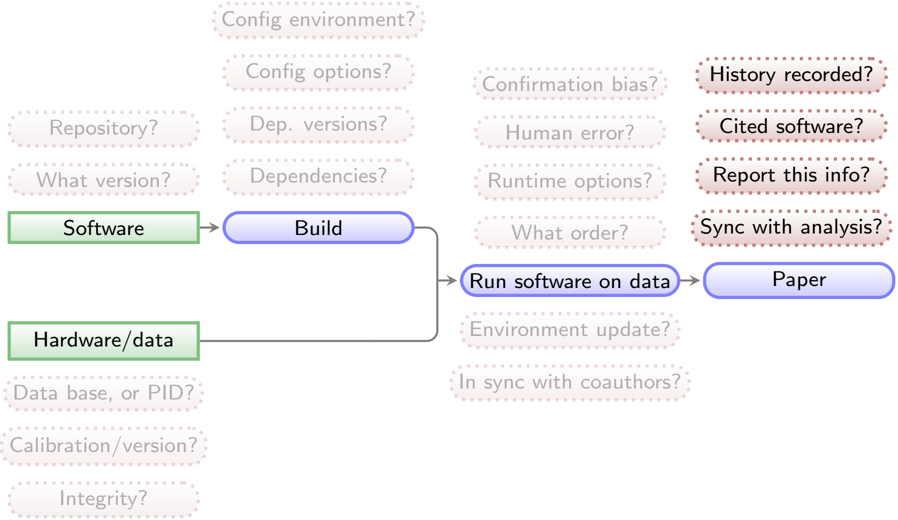
- Unlike a script which always starts from the top, a Makefile starts from the end and steps that don't change will be left untouched (not remade).
- A single *rule* can manage any number of files.
- Make can identify independent steps internally and do them in parallel.
- Make was designed for complex projects with thousands of files (all major Unix-like components), so it is highly evolved and efficient.
- Make is a very simple and small language, thus easy to learn with great and free documentation (for example GNU Make's manual).

Reproducible science: Maneage is managed through a Makefile

All steps (downloading and analysis) are managed by Makefiles (example from zenodo.1164774):

- Unlike a script which always starts from the top, a Makefile starts from the end and steps that don't change will be left untouched (not remade).
- A single *rule* can manage any number of files.
- Make can identify independent steps internally and do them in parallel.
- Make was designed for complex projects with thousands of files (all major Unix-like components), so it is highly evolved and efficient.
- Make is a very simple and small language, thus easy to learn with great and free documentation (for example GNU Make's manual).

General outline of a project (after data collection)



Green boxes with sharp corners: *source*/input components/files.
Blue boxes with rounded corners: *built* components.
Red boxes with dashed borders: questions that must be clarified for each phase.

Values in final report/paper

All analysis results (numbers, plots, tables) written in paper's PDF as LaTeX macros. They are thus updated automatically on any change.
Shown here is a portion of the NoiseChisel paper and its LaTeX source (arXiv:1505.01664).

```
\begin{equation}
  \label{tSNeq}
  \mathrm{S/N}_T=\frac{NF-NS_a}{\sqrt{NF+N\sigma_S^2}}
  =\frac{\sqrt{N}(F-S_a)}{\sqrt{F+\sigma_S^2}}.
\end{equation}

\noindent
See Section \ref{SNeqmodif} for the modifications required when
the input image is not in units of counts or has already been
Sky subtracted. The distribution of {\small S/N}$_T$ from the
objects in $R_s$ for the three examples in Figure \ref{dettf}
can be seen in column 5 (top) of that figure. Image processing
effects, mainly due to shifting, rotating, and re-sampling the
images for co-adding, on the real data further increase the size
and count, and hence, the {\small S/N} of false detections in
real, reduced/co-added images. A comparison of scales on the
{\small S/N} histograms between the mock ((a.5.1) and (b.5.1))
and real (c.5.1) examples in Figure \ref{dettf} shows the effect
quantitatively. In the histograms of Figure \ref{dettf}, the bin
with the largest number of false pseudo-detections respectively
has an {\small S/N} of $\onelargedettfmax$,
$\sensitivitycdettfmax$, and $\fourdettfmax$.
```

smaller than `--detsnminarea` are removed from the analysis in both R_s and R_d. In the examples in this section, it is set to `15`. Note that since a threshold approximately equal to the Sky value is used, this is a very weak constraint. For each pseudo-detection, $\mathrm{S/N}_T$ can be written as,

$$\mathrm{S/N}_T=\frac{NF-NS_a}{\sqrt{NF+N\sigma_S^2}}=\frac{\sqrt{N}(F-S_a)}{\sqrt{F+\sigma_S^2}}. \qquad (3)$$

See Section 3.3 for the modifications required when the input image is not in units of counts or has already been Sky subtracted. The distribution of $\mathrm{S/N}_T$ from the objects in R_s for the three examples in Figure 7 can be seen in column 5 (top) of that figure. Image processing effects, mainly due to shifting, rotating, and re-sampling the images for co-adding, on the real data further increase the size and count, and hence, the S/N of false detections in real, reduced/co-added images. A comparison of scales on the S/N histograms between the mock ((a.5.1) and (b.5.1)) and real (c.5.1) examples in Figure 7 shows the effect quantitatively. In the histograms of Figure 7, the bin with the largest number of false pseudo-detections respectively has an S/N of 1.89, 2.37, and 4.77.

The $\mathrm{S/N}_T$ distribution of detections in R_s provides a very ro-

Values in final report/paper

All analysis results (numbers, plots, tables) written in paper's PDF as LaTeX macros. They are thus updated automatically on any change.
Shown here is a portion of the NoiseChisel paper and its LaTeX source (arXiv:1505.01664).

```
\begin{equation}
  \label{tSNeq}
  \mathrm{S/N}_T=\frac{NF-NS_a}{\sqrt{NF+N\sigma_S^2}}
  =\frac{\sqrt{N}(F-S_a)}{\sqrt{F+\sigma_S^2}}.
\end{equation}

\noindent
See Section \ref{SNeqmodif} for the modifications required when
the input image is not in units of counts or has already been
Sky subtracted. The distribution of {\small S/N}$_T$ from the
objects in $R_s$ for the three examples in Figure \ref{dettf}
can be seen in column 5 (top) of that figure. Image processing
effects, mainly due to shifting, rotating, and re-sampling the
images for co-adding, on the real data further increase the size
and count, and hence, the {\small S/N} of false detections in
real, reduced/co-added images. A comparison of scales on the
{\small S/N} histograms between the mock ((a.5.1) and (b.5.1))
and real (c.5.1) examples in Figure \ref{dettf} shows the effect
quantitatively. In the histograms of Figure \ref{dettf}, the bin
with the largest number of false pseudo-detections respectively
has an {\small S/N} of $\onelargedettfmax$,
$\sensitivitycdettfmax$, and $\fourdettfmax$.
```

smaller than `--detsnminarea` are removed from the analysis in both R_s and R_d. In the examples in this section, it is set to `15`. Note that since a threshold approximately equal to the Sky value is used, this is a very weak constraint. For each pseudo-detection, $\mathrm{S/N}_T$ can be written as,

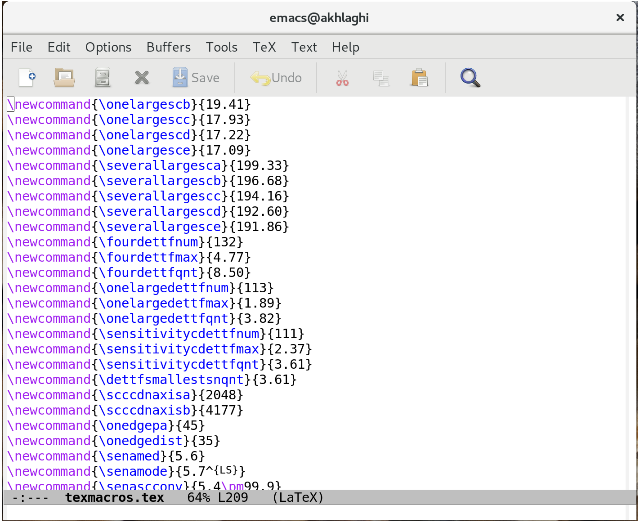
$$\mathrm{S/N}_T = \frac{NF - NS_a}{\sqrt{NF + N\sigma_S^2}} = \frac{\sqrt{N}(F - S_a)}{\sqrt{F + \sigma_S^2}}. \qquad (3)$$

See Section 3.3 for the modifications required when the input image is not in units of counts or has already been Sky subtracted. The distribution of $\mathrm{S/N}_T$ from the objects in R_s for the three examples in Figure 7 can be seen in column 5 (top) of that figure. Image processing effects, mainly due to shifting, rotating, and re-sampling the images for co-adding, on the real data further increase the size and count, and hence, the S/N of false detections in real, reduced/co-added images. A comparison of scales on the S/N histograms between the mock ((a.5.1) and (b.5.1)) and real (c.5.1) examples in Figure 7 shows the effect quantitatively. In the histograms of Figure 7, the bin with the largest number of false pseudo-detections respectively has an S/N of 1.89, 2.37, and 4.77.

The $\mathrm{S/N}_T$ distribution of detections in R_s provides a very ro-

Analysis step results/values concatenated into a single file.

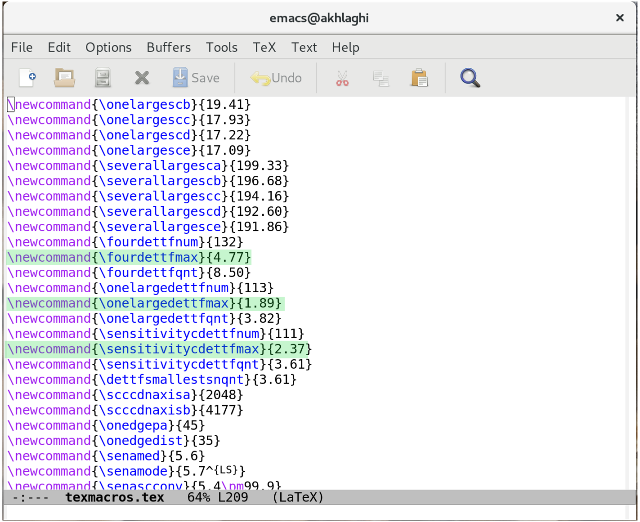
All LaTeX macros come from a single file.




```
\newcommand{\onelargescb}{19.41}
\newcommand{\onelargescc}{17.93}
\newcommand{\onelargescd}{17.22}
\newcommand{\onelargesce}{17.09}
\newcommand{\severallargesca}{199.33}
\newcommand{\severallargescb}{196.68}
\newcommand{\severallargescc}{194.16}
\newcommand{\severallargescd}{192.60}
\newcommand{\severallargesce}{191.86}
\newcommand{\fourdettfnum}{132}
\newcommand{\fourdettfmax}{4.77}
\newcommand{\fourdettfqnt}{8.50}
\newcommand{\onelargedettfnum}{113}
\newcommand{\onelargedettfmax}{1.89}
\newcommand{\onelargedettfqnt}{3.82}
\newcommand{\sensitivitycdettfnum}{111}
\newcommand{\sensitivitycdettfmax}{2.37}
\newcommand{\sensitivitycdettfqnt}{3.61}
\newcommand{\dettfsmallestsnqnt}{3.61}
\newcommand{\scccdnaxisa}{2048}
\newcommand{\scccdnaxisb}{4177}
\newcommand{\onedgepa}{45}
\newcommand{\onedgedist}{35}
\newcommand{\senamed}{5.6}
\newcommand{\senamode}{5.7^{LS}}
\newcommand{\senascconv}{5.4\pm99.9}
```

Analysis step results/values concatenated into a single file.

All LaTeX macros come from a single file.



Analysis results stored as LaTeX macros

The analysis scripts write/update the LaTeX macro values automatically.

```
# Numbers for dettf.tex:
sqnt=9999999
function dettfhist
{
    # Set the file name.
    if [ $2 == 4 ]; then                      obase=four;
    elif [ $2 = sensitivity3 ]; then    obase=sensitivityc;
    else                                      obase=$2;
    fi
    if [ $2 == onelarge ]; then ind="_7"; else ind="_12"; fi
    name=$1$2$ind"_detsn"$txt

    dettfnum=$(awk '/points binned in/{print $4; exit(0)}' $name)
    dettfqnt=$(awk '/quantile has a value of/{
                         printf("%.2f", $9); exit(0);}' $name)
    dettfmax=$(awk 'BEGIN { max=-999999 }
                        !/^#/ { if($2>max){max=$2; mv=$1} }
                        END { printf("%.2f", mv) }' $name)
    addtexmacro $obase"dettfnum" $dettfnum
    addtexmacro $obase"dettfmax" $dettfmax
    addtexmacro $obase"dettfqnt" $dettfqnt

    # Find the smallest S/N quantile:
    sqnt=$(echo " " | awk '{if('$dettfqnt'<'$sqnt') print '$dettfqnt'}')
}
for base in 4 onelarge sensitivity3
do dettfhist $texdir/dettf/ $base;  done
addtexmacro dettfsmallestsnqnt $sqnt
```

Analysis results stored as LaTeX macros

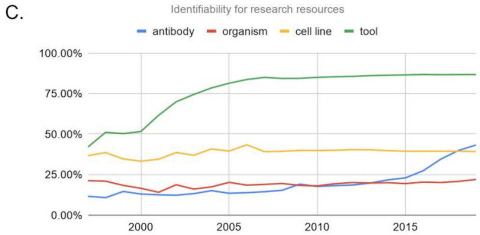
The analysis scripts write/update the LaTeX macro values automatically.

```
# Numbers for dettf.tex:
sqnt=9999999
function dettfhist
{
    # Set the file name.
    if [ $2 == 4 ]; then                        obase=four;
    elif [ $2 = sensitivity3 ]; then    obase=sensitivityc;
    else                                        obase=$2;
    fi
    if [ $2 == onelarge ]; then ind="_7"; else ind="_12"; fi
    name=$1$2$ind"_detsn"$txt

    dettfnum=$(awk '/points binned in/{print $4; exit(0)}' $name)
    dettfqnt=$(awk '/quantile has a value of/{
                     printf("%.2f", $9); exit(0);}' $name)
    dettfmax=$(awk 'BEGIN { max=-999999 }
                    !/^#/ { if($2>max){max=$2; mv=$1} }
                    END { printf("%.2f", mv) }' $name)
    addtexmacro $obase"dettfnum" $dettfnum
    addtexmacro $obase"dettfmax" $dettfmax
    addtexmacro $obase"dettfqnt" $dettfqnt

    # Find the smallest S/N quantile:
    sqnt=$(echo " " | awk '{if('$dettfqnt'<'$sqnt') print '$dettfqnt'}')
}
for base in 4 onelarge sensitivity3
do dettfhist $texdir/dettf/ $base;  done
addtexmacro dettfsmallestsnqnt $sqnt
```

Let's look at the data lineage to replicate Figure 1C (green/tool) of Menke+2020 (DOI:10.1101/2020.01.15.908111), as done in arXiv:2006.03018 for a demo.

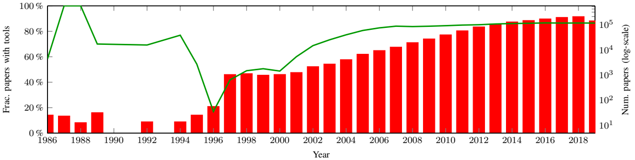
ORIGINAL PLOT

The Green plot shows the fraction of papers mentioning software tools from 1997 to 2019.

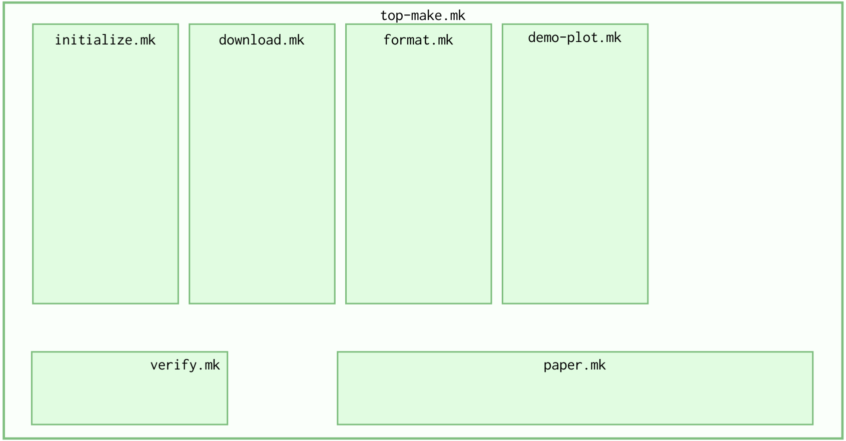


OUR enhanced REPLICATION

The green line is same as above but over their full historical range.
Red histogram is the number of papers studied in each year

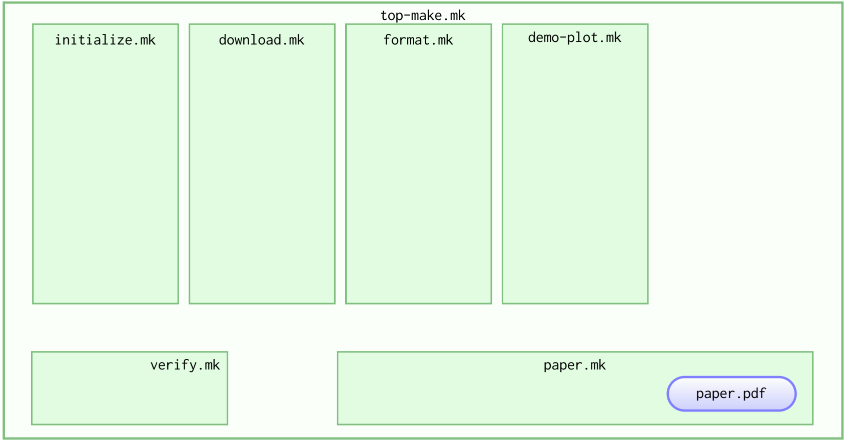


Makefiles (.mk) keep contextually separate parts of the project, all imported into top-make.mk



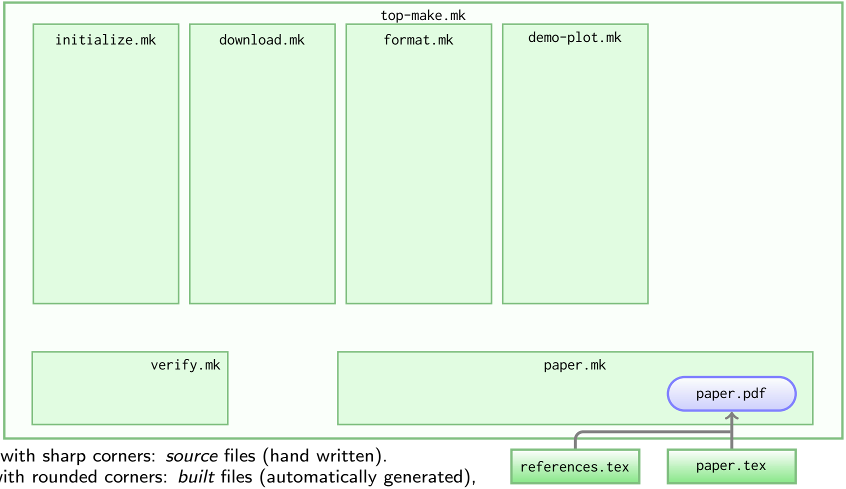
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

The ultimate purpose of the project is to produce a paper/report (in PDF).



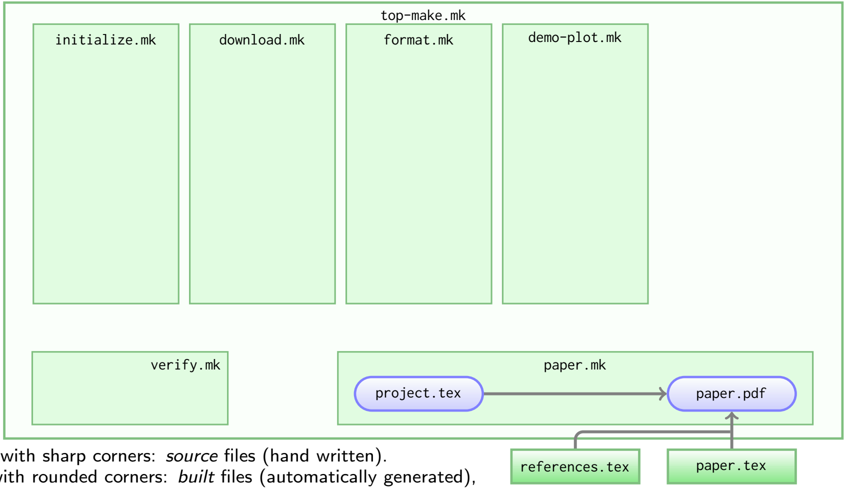
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

The narrative description, typography and references are in `paper.tex` & `references.tex`.



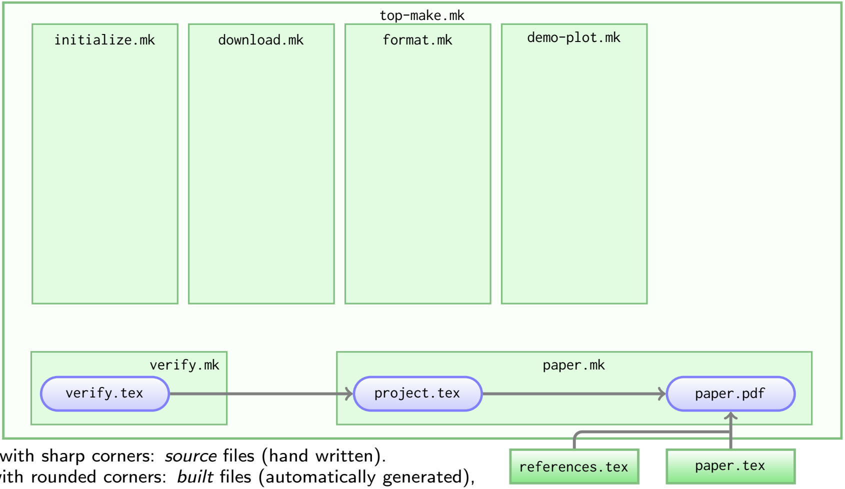
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

Analysis outputs (blended into the PDF as LaTeX macros) come from project.tex.



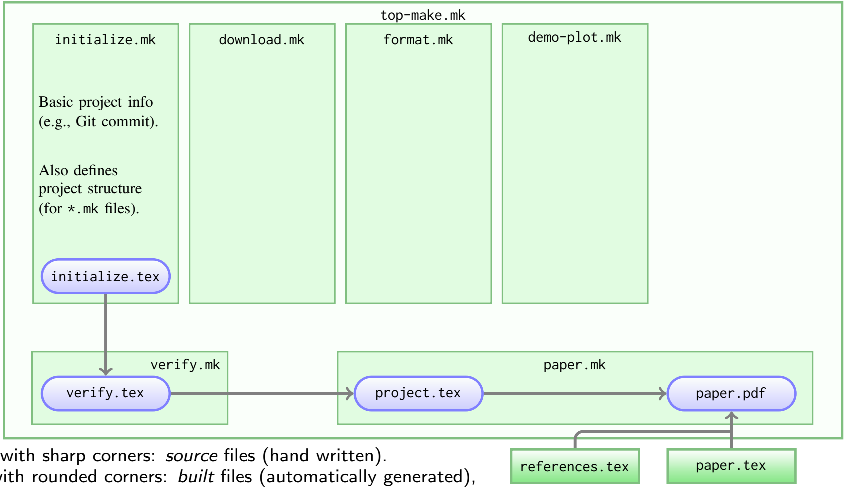
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

But analysis outputs must first be *verified* (with checksums) before entering the report/paper.



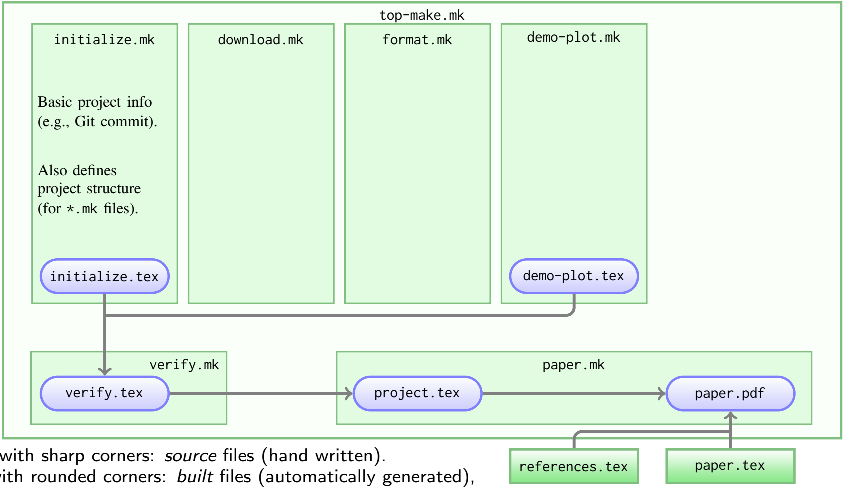
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

Basic project info comes from `initialize.tex`.



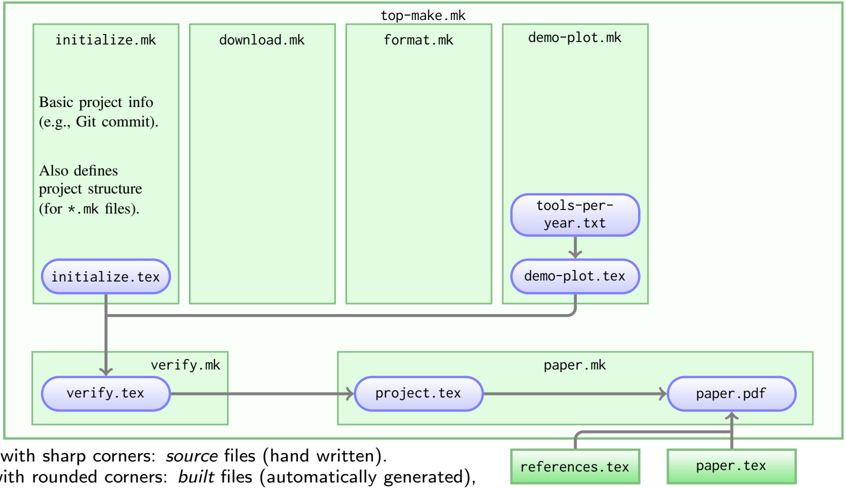
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

The paper includes some information about the plot.



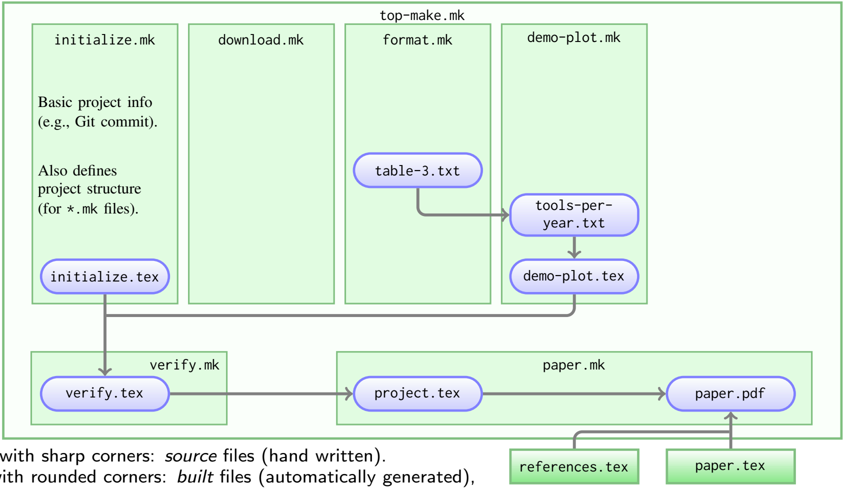
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

The final plotted data are calculated and stored in `tools-per-year.txt`.



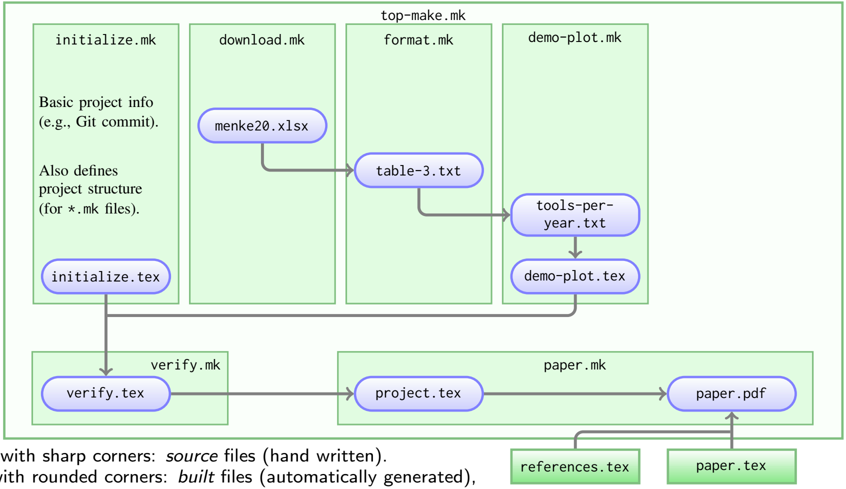
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

The plot's calculation is done on a formatted sub-set of the raw input data.



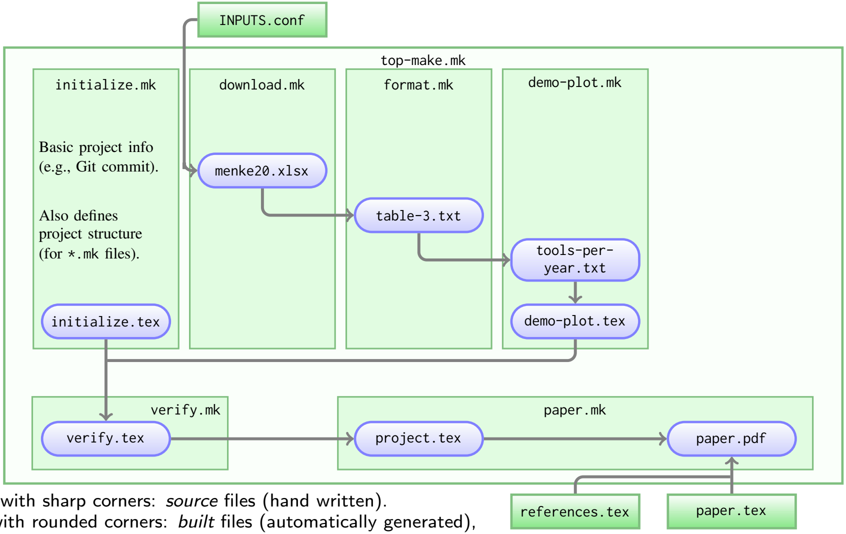
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

The raw data that were downloaded are stored in XLSX format.



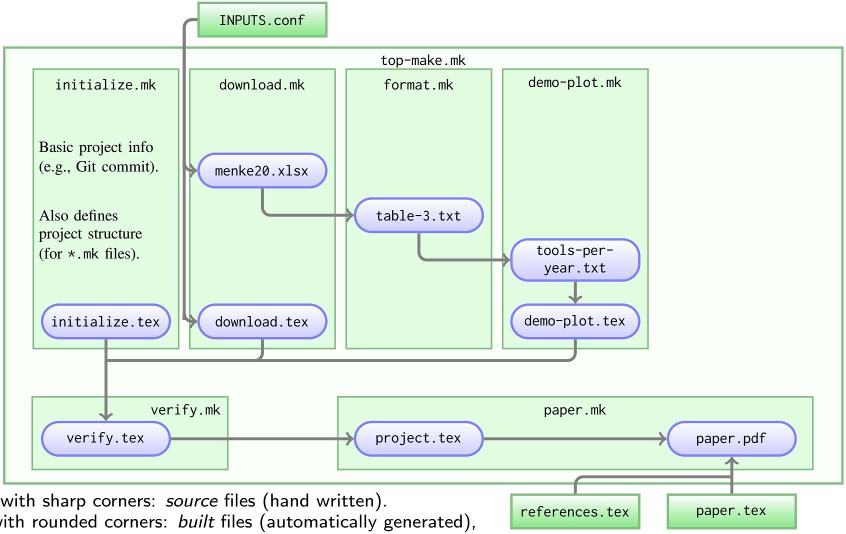
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

The download URL *and* a checksum to validate the raw inputs, are stored in `INPUTS.conf`.



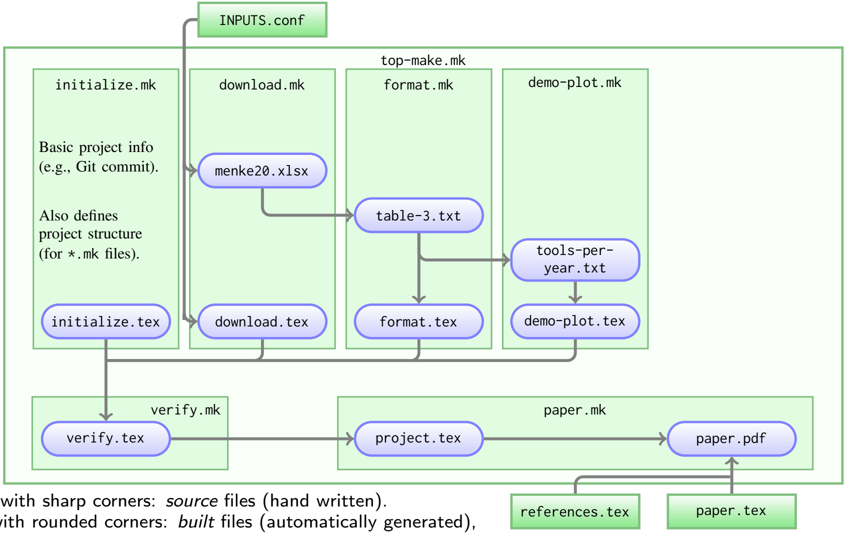
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

We also need to report the URL in the paper...



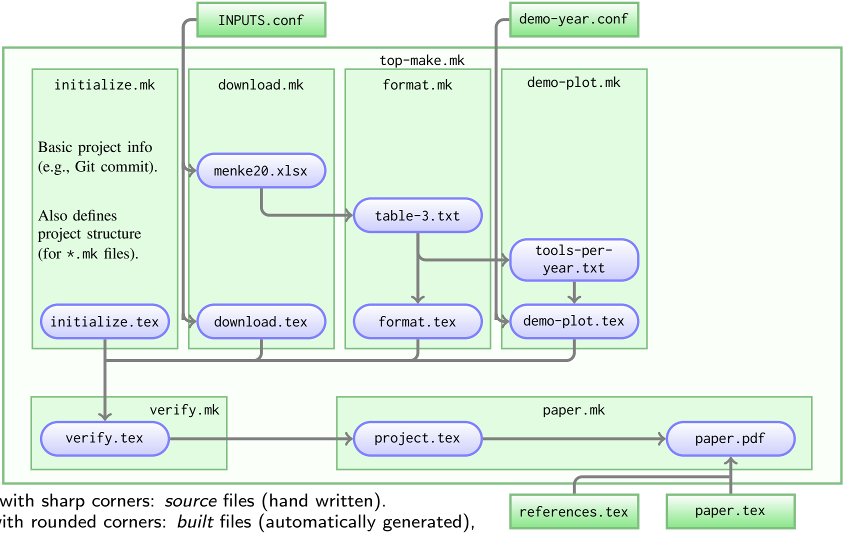
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

Some general info about the full dataset may also be reported.



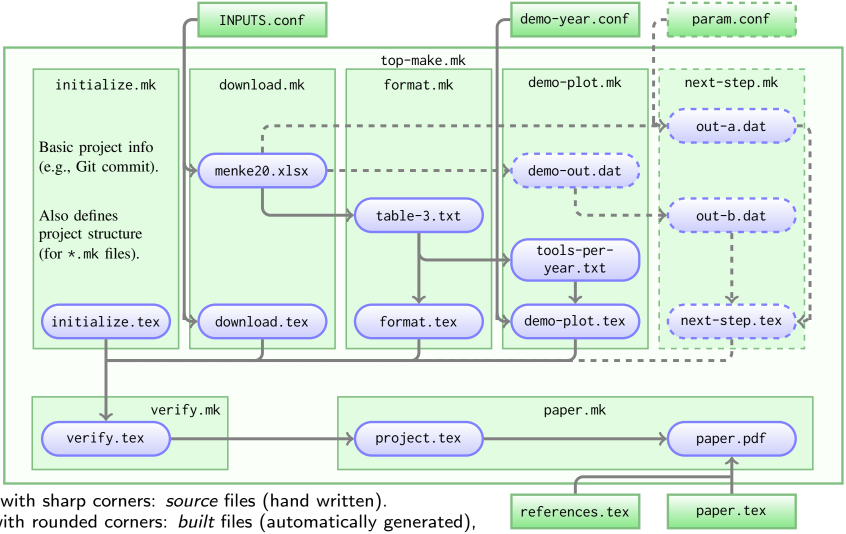
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

We report the number of papers studied in a special year, desired year is stored in .conf file.



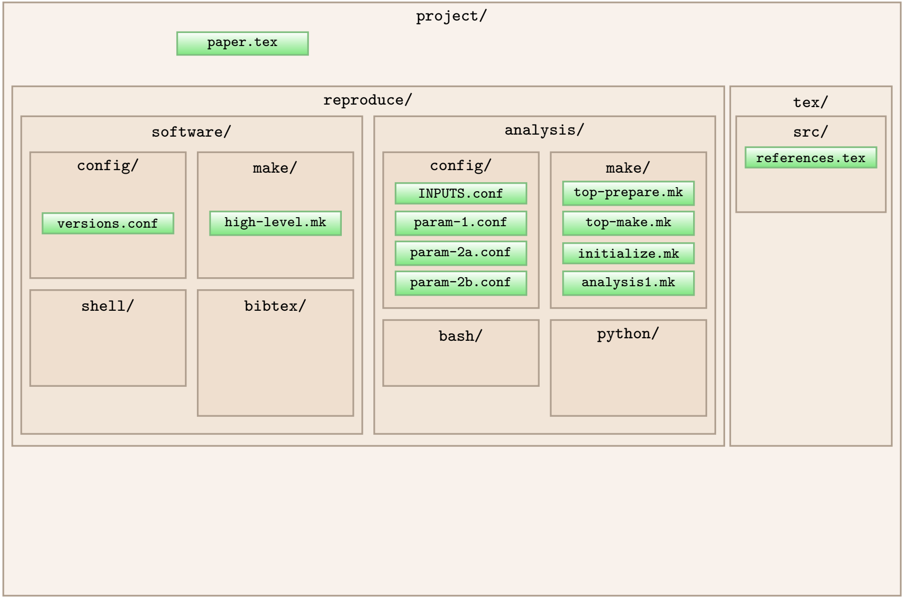
Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

It is very easy to expand the project and add new analysis steps (this solution is scalable)

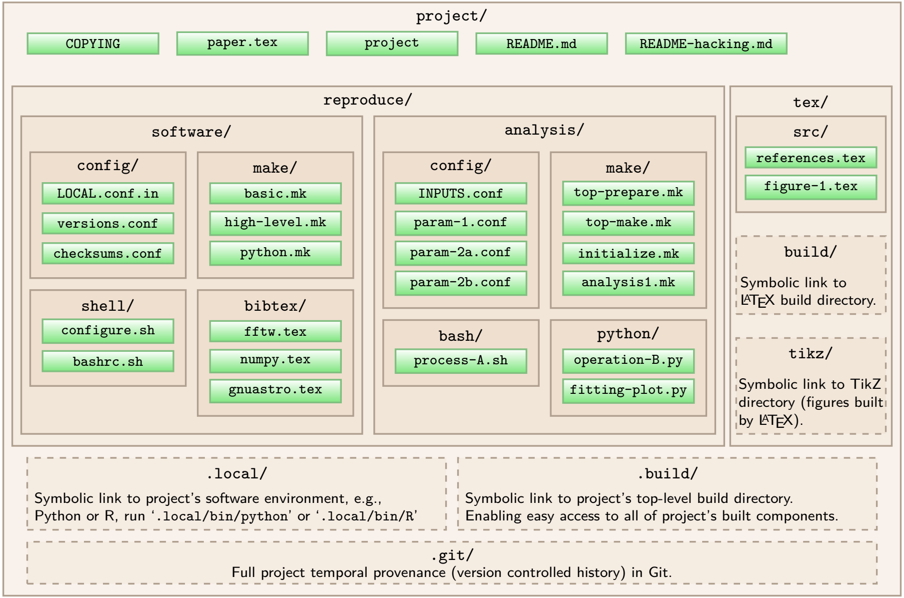


Green boxes with sharp corners: *source* files (hand written).
Blue boxes with rounded corners: *built* files (automatically generated),
built files are shown in the Makefile that contains their build instructions.

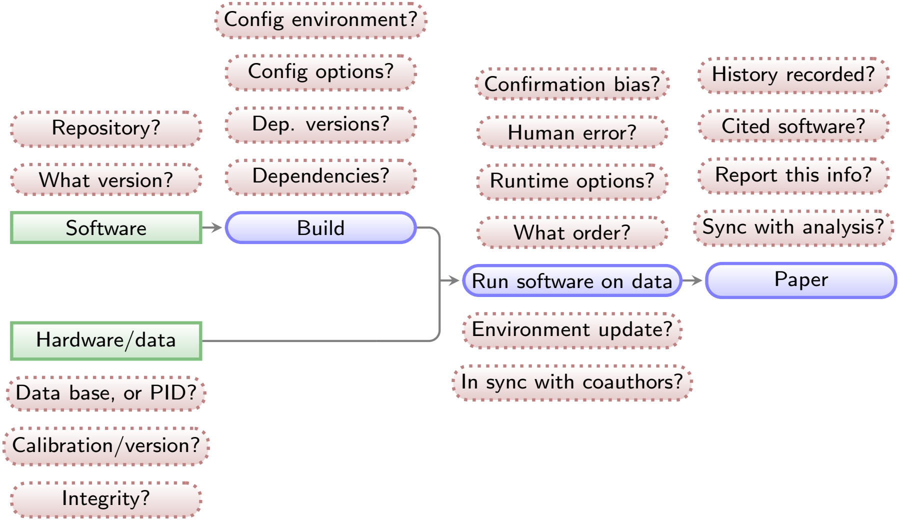
Files organized in directories by context (here are some of the files discussed before)



Files organized in directories by context (now with other project files and symbolic links)

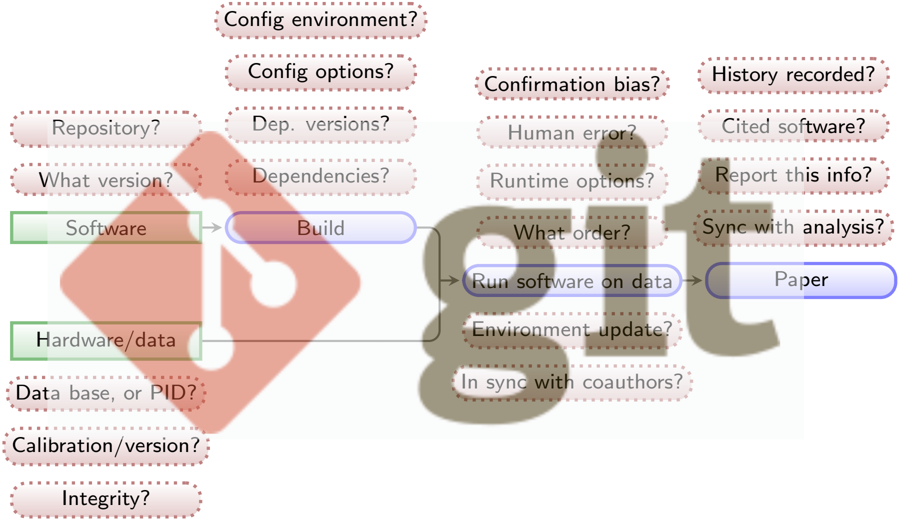


All questions have an answer now (in plain text: human & computer readable/archivable).



Green boxes with sharp corners: *source*/input components/files.
Blue boxes with rounded corners: *built* components.
Red boxes with dashed borders: questions that must be clarified for each phase.

All questions have an answer now (in plain text: so we can use Git to keep its history).



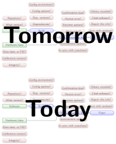
Green boxes with sharp corners: *source*/input components/files.
Blue boxes with rounded corners: *built* components.
Red boxes with dashed borders: questions that must be clarified for each phase.

New projects branch from Maneage

- The project (answers to questions above) will evolve.

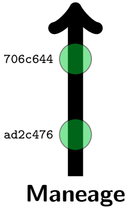
New projects branch from Maneage

- The project (answers to questions above) will evolve.



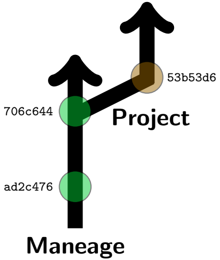
New projects branch from Maneage

- Each point of project's history is recorded with Git.

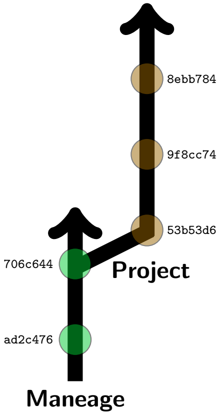


New projects branch from Maneage

- Each point of project's history is recorded with Git.
- New project: a branch from the template.
  Recall that every commit contains the following:
  - Instructions to download, verify and build software.
  - Instructions to download and verify input data.
  - Instructions to run software on data (do the analysis).
  - Narrative description of project's purpose/context.

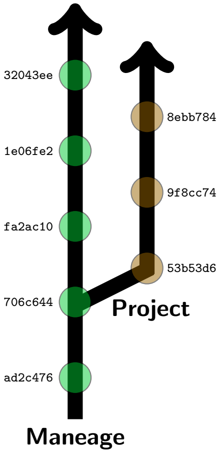


New projects branch from Maneage



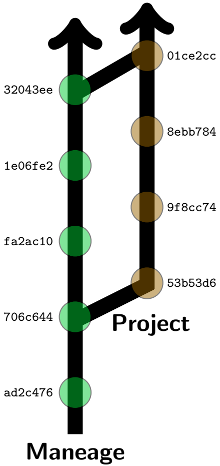
- Each point of project's history is recorded with Git.
- New project: a branch from the template.
  Recall that every commit contains the following:
  - Instructions to download, verify and build software.
  - Instructions to download and verify input data.
  - Instructions to run software on data (do the analysis).
  - Narrative description of project's purpose/context.
- Research progresses in the project branch.

New projects branch from Maneage



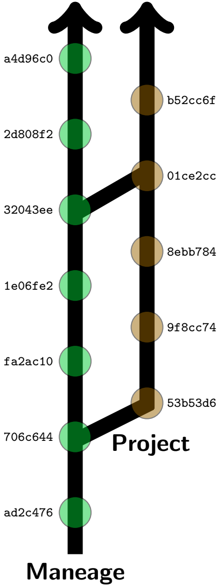
- Each point of project's history is recorded with Git.
- New project: a branch from the template.
  Recall that every commit contains the following:
  - Instructions to download, verify and build software.
  - Instructions to download and verify input data.
  - Instructions to run software on data (do the analysis).
  - Narrative description of project's purpose/context.
- Research progresses in the project branch.
- Template will evolve (improved infrastructure).

New projects branch from Maneage



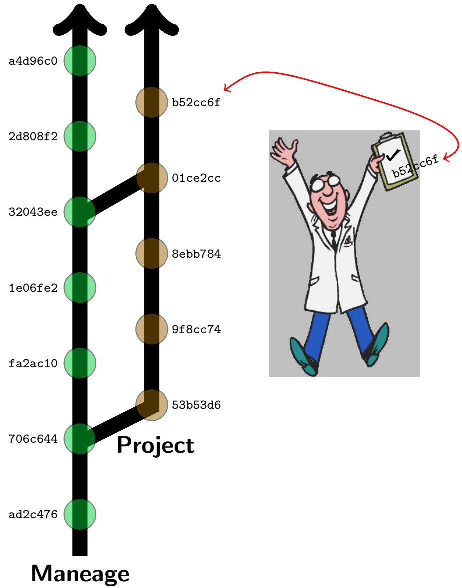
- Each point of project's history is recorded with Git.
- New project: a branch from the template.
  Recall that every commit contains the following:
  - Instructions to download, verify and build software.
  - Instructions to download and verify input data.
  - Instructions to run software on data (do the analysis).
  - Narrative description of project's purpose/context.
- Research progresses in the project branch.
- Template will evolve (improved infrastructure).
- Template can be imported/merged back into project.

New projects branch from Maneage



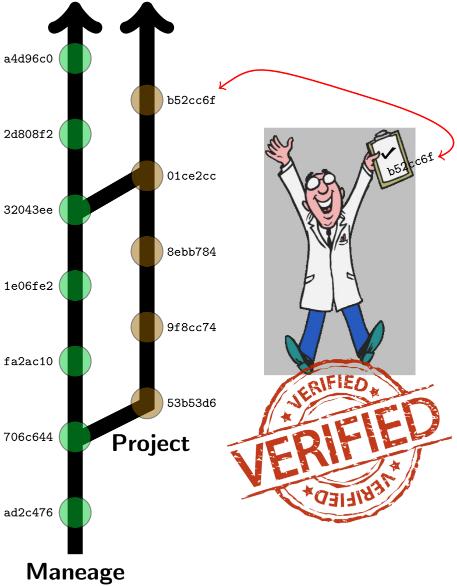
- Each point of project's history is recorded with Git.
- New project: a branch from the template.
  Recall that every commit contains the following:
  - Instructions to download, verify and build software.
  - Instructions to download and verify input data.
  - Instructions to run software on data (do the analysis).
  - Narrative description of project's purpose/context.
- Research progresses in the project branch.
- Template will evolve (improved infrastructure).
- Template can be imported/merged back into project.
- The template and project will evolve.
- During research this encourages creative tests
  (previous research states can easily be retrieved).
- Coauthors can work on same project in parallel
  (separate project branches).

New projects branch from Maneage



- Each point of project's history is recorded with Git.
- New project: a branch from the template.
  Recall that every commit contains the following:
  - Instructions to download, verify and build software.
  - Instructions to download and verify input data.
  - Instructions to run software on data (do the analysis).
  - Narrative description of project's purpose/context.
- Research progresses in the project branch.
- Template will evolve (improved infrastructure).
- Template can be imported/merged back into project.
- The template and project will evolve.
- During research this encourages creative tests (previous research states can easily be retrieved).
- Coauthors can work on same project in parallel (separate project branches).
- Upon publication, the Git checksum is enough to verify the integrity of the result.

New projects branch from Maneage



- Each point of project's history is recorded with Git.
- New project: a branch from the template.
  Recall that every commit contains the following:
  - Instructions to download, verify and build software.
  - Instructions to download and verify input data.
  - Instructions to run software on data (do the analysis).
  - Narrative description of project's purpose/context.
- Research progresses in the project branch.
- Template will evolve (improved infrastructure).
- Template can be imported/merged back into project.
- The template and project will evolve.
- During research this encourages creative tests
  (previous research states can easily be retrieved).
- Coauthors can work on same project in parallel
  (separate project branches).
- Upon publication, the Git checksum is enough to
  verify the integrity of the result.

"Verified" image from vectorstock.com

Two recent examples (publishing Git checksum in abstract)

The Realm of the Low-Surface-Brightness Universe
Proceedings IAU Symposium No. 355, 2019
D. Valls-Gabaud, I. Trujillo & S. Okamoto, eds.

DOI: 00.0000/X000000000000000X

Carving out the low surface brightness universe with NoiseChisel

Mohammad Akhlaghi[1,2]

[1]Instituto de Astrofísica de Canarias, C/ Vía Láctea, 38200 La Laguna, Tenerife, Spain.
email: `mohammad@akhlaghi.org`

[2]Facultad de Física, Universidad de La Laguna, Avda. Astrofísico Fco. Sánchez s/n, 38200 La Laguna, Tenerife, Spain.

Abstract. NoiseChisel is a program to detect very low signal-to-noise ratio (S/N) features with minimal assumptions on their morphology. It was introduced in 2015 and released within a collection of data analysis programs and libraries known as GNU Astronomy Utilities (Gnuastro). Over the last ten stable releases of Gnuastro, NoiseChisel has significantly improved: detecting even fainter signal, enabling better user control over its inner workings, and many bug fixes. The most important change may be that NoiseChisel's segmentation features have been moved into a new program called Segment. Another major change is the final growth strategy of its true detections, for example NoiseChisel is able to detect the outer wings of M51 down to S/N of 0.25, or 28.27 mag/arcsec^2 on a single-exposure SDSS image (r-band). Segment is also able to detect the localized HII regions as "clumps" much more successfully. Finally, to orchestrate a controlled analysis, the concept of a "reproducible paper" is discussed: this paper itself is exactly reproducible (snapshot v4-0-g8505cfd).

Keywords. galaxies: halos, galaxies: photometry, galaxies: structure, methods: data analysis, methods: reproducible, techniques: image processing, techniques: photometric

1. Introduction

Signal from the low surface brightness universe is buried deep in the datasets noise and thus requires accurate detection methods. In Akhlaghi and Ichikawa (2015) (henceforth AI15) a new method was introduced to detect such very low signal-to-noise ratio (S/N) signal from the images in a non-parametric manner. It allows accurate detection of the diffuse outer features of galaxies (that often have a different morphology from the centers). The software implementation of this method (NoiseChisel) is released as part of a larger collection of data analysis software known as GNU Astronomy Utilities† (Gnuastro). It was the first professional astronomical software to be independently refereed by an independent panel (GNU Evaluation committee) and fully conforms with the GNU Coding Standards‡.

Since its release, NoiseChisel has been used in many studies. For example Bacon et al. (2017) used it to identify objects that were missed by Rafelski et al. (2015) (henceforth R15), who used a combination of six SExtractor (Bertin and Arnouts 1996) runs with different configurations to avoid deblending problems, but still missed many sources with significant signal, see Figure 1. Borlaff et al. (2019), Mller et al. (2019), and Trujillo et al. (2019) used it for accurate flat field and Sky subtraction to create deeper co-added images in galaxy fields for optimal detection of the low surface brightness features. Calvi et al. (2019) used it to find Lyman-α emitters in spectra. For future studies, Laine et al.

† `https://www.gnu.org/s/gnuastro`
‡ `https://www.gnu.org/prep/standards`

Monthly Notices of the ROYAL ASTRONOMICAL SOCIETY

MNRAS **491,** 5317–5329 (2020) doi:10.1093/mnras/stz3111
Advance Access publication 2019 November 14

The Sloan Digital Sky Survey extended point spread functions

Raúl Infante-Sainz[1,2]★ Ignacio Trujillo[1,2] and Javier Román[1,2,3]

[1]*Instituto de Astrofísica de Canarias, c/ Vía Láctea s/n, E-38205 La Laguna, Tenerife, Spain*
[2]*Departamento de Astrofísica, Universidad de La Laguna, E-38205 La Laguna, Tenerife, Spain*
[3]*Instituto de Astrofísica de Andalucía (CSIC), Glorieta de la Astronomía, E-18008 Granada, Spain*

Accepted 2019 October 30. Received 2019 October 29; in original form 2019 September 10

ABSTRACT

A robust and extended characterization of the point spread function (PSF) is crucial to extract the photometric information produced by deep imaging surveys. Here, we present the extended PSFs of the Sloan Digital Sky Survey (SDSS), one of the most productive astronomical surveys of all time. By stacking ∼1000 images of individual stars with different brightness, we obtain the bidimensional SDSS PSFs extending over 8 arcmin in radius for all the SDSS filters (u, g, r, i, z). This new characterization of the SDSS PSFs is near a factor of 10 larger in extension than previous PSFs characterizations of the same survey. We found asymmetries in the shape of the PSFs caused by the drift scanning observing mode. The flux of the PSFs is larger along the drift scanning direction. Finally, we illustrate with an example how the PSF models can be used to remove the scattered light field produced by the brightest stars in the central region of the Coma cluster field. This particular example shows the huge importance of PSFs in the study of the low-surface brightness Universe, especially with the upcoming of ultradeep surveys, such as the Large Synoptic Survey Telescope (LSST). Following a reproducible science philosophy, we make all the PSF models and the scripts used to do the analysis of this paper publicly available (snapshot v0.4-0-gd966ad0).

Key words: instrumentation: detectors – methods: data analysis – techniques: image processing – techniques: photometric – galaxies: haloes.

1 INTRODUCTION

The point spread function (PSF) describes the response of an imaging system to the light produced by a point source. Real PSFs have complex structures as their shapes depend on the optical path that light takes as it travels through the atmosphere and multiple optical elements, mirrors, lenses, detectors, etc. For the vast majority of astronomical works, only a tiny portion of the PSF (i.e. normally a few inner arcseconds; see e.g. Trujillo et al. 2001a, b) is characterized. In practice, however, the light of both point and extended sources are spread over the entire detector due to the effect of the PSF at large radii. Therefore, it is necessary to have a good understanding of its structure along the entire detector (typically extending over arcminutes or more).

Extended PSFs have become a vital tool to obtain precise photometric information in modern astronomical surveys. For instance, Slater, Harding & Mihos (2009) modelled the extended PSF and the internal reflections produced by the stars of the Burrell Schmidt telescope and showed that virtually all the pixels of the image are dominated by the scattered light by both stars and galaxies at 29.5 mag arcsec^{-2} (V-band). Trujillo & Fliri (2016) also characterized and used the extended PSF of the 10.4 m Gran Telescopio Canarias (GTC) telescope to model and remove the scattered light in ultradeep observations of the UGC 00180 galaxy. Even more troublesome for low-surface brightness studies is the finding (see e.g. Trujillo & Bakos 2013; Sandin 2014, 2015) that the outer regions of astronomical objects are severely affected by their own scattered light produced by the convolution with the PSF. In order to correct this effect, Karabal et al. (2017) generated the PSF and models of the internal reflections from images of the Canada–France–Hawaii Telescope (CFHT) to de-convolve a sample of three galaxies and correct them from instrumental scattered light. More recently, Román, Trujillo & Montes (2019) characterized the PSFs of the Stripe 82 survey and used them to model and correct the scattered light field produced by stars to study the optical properties of the Galactic cirri. All the above works have shown that having an extended PSF is crucial when accurate photometric and structure properties of astronomical objects at low-surface brightness levels are required.

One of the most commonly used surveys for measuring photometric properties of astronomical objects is the Sloan Sky Digital Survey (SDSS; York et al. 2000), covering 14 555 deg^2 on the sky (just over 35 per cent of the full sky) in five photometric bands (u, g, r, i, and z). Although SDSS is a relatively shallow survey compared

★ E-mail: infantesainz@gmail.com

Two recent examples (publishing Git checksum in abstract)

The Realm of the Low-Surface-Brightness Universe
Proceedings IAU Symposium No. 355, 2019
D. Valls-Gabaud, I. Trujillo & S. Okamoto, eds.

DOI: 00.0000/X000000000000000X

Carving out the low surface brightness universe with NoiseChisel

Mohammad Akhlaghi[1,2]

[1]Instituto de Astrofísica de Canarias, C/ Vía Láctea, 38200 La Laguna, Tenerife, Spain.
email: mohammad@akhlaghi.org

[2]Facultad de Física, Universidad de La Laguna, Avda. Astrofísico Fco. Sánchez s/n, 38200 La Laguna, Tenerife, Spain.

Abstract. NoiseChisel is a program to detect very low signal-to-noise ratio (S/N) features with minimal assumptions on their morphology. It was introduced in 2015 and released within a collection of data analysis programs and libraries known as GNU Astronomy Utilities (Gnuastro). Over the last ten stable releases of Gnuastro, NoiseChisel has significantly improved: detecting even fainter signal, enabling better user control over its inner workings, and many bug fixes. The most important change may be that NoiseChisel's segmentation features have been moved into a new program called Segment. Another major change is the final growth strategy of its true detections, for example NoiseChisel is able to detect the outer wings of M51 down to S/N of 0.25, or 28.27 mag/arcsec^2 on a single-exposure SDSS image (r-band). Segment is also able to detect the localized HII regions as "clumps" much more successfully. Finally, to orchestrate a controlled analysis, the concept of a "reproducible paper" is discussed: this paper itself is exactly reproducible (snapshot v4-0-g8505cfd).

VERIFIED

Keywords. galaxies: halos, galaxies: photometry, galaxies: structure, methods: data analysis, methods: reproducible, techniques: image processing, techniques: photometric

1. Introduction

Signal from the low surface brightness universe is buried deep in the datasets noise and thus requires accurate detection methods. In Akhlaghi and Ichikawa (2015) (henceforth AI15) a new method was introduced to detect such very low signal-to-noise ratio (S/N) signal from the images in a non-parametric manner. It allows accurate detection of the diffuse outer features of galaxies (that often have a different morphology from the centers). The software implementation of this method (NoiseChisel) is released as part of a larger collection of data analysis software known as GNU Astronomy Utilities† (Gnuastro). It was the first professional astronomical software to be independently refereed by an independent panel (GNU Evaluation committee) and fully conforms with the GNU Coding Standards‡.

Since its release, NoiseChisel has been used in many studies. For example Bacon et al. (2017) used it to identify objects that were missed by Rafelski et al. (2015) (henceforth R15), who used a combination of six SExtractor (Bertin and Arnouts 1996) runs with different configurations to avoid deblending problems, but still missed many sources with significant signal, see Figure 1. Borlaff et al. (2019), Mller et al. (2019), and Trujillo et al. (2019) used it for calibrating flat field and Sky subtraction to create deeper co-added images in galaxy fields for optimal detection of the low surface brightness features. Calvi et al. (2019) used it to find Lyman-α emitters in spectra. For future studies, Laine et al.

† https://www.gnu.org/s/gnuastro
‡ https://www.gnu.org/prep/standards

Monthly Notices of the ROYAL ASTRONOMICAL SOCIETY

MNRAS **491**, 5317–5329 (2020) doi:10.1093/mnras/stz3111
Advance Access publication 2019 November 14

The Sloan Digital Sky Survey extended point spread functions

Raúl Infante-Sainz[1,2★] Ignacio Trujillo[1,2] and Javier Román[1,2,3]

[1]Instituto de Astrofísica de Canarias, c/ Vía Láctea s/n, E-38205 La Laguna, Tenerife, Spain
[2]Departamento de Astrofísica, Universidad de La Laguna, E-38205 La Laguna, Tenerife, Spain
[3]Instituto de Astrofísica de Andalucía (CSIC), Glorieta de la Astronomía, E-18008 Granada, Spain

Accepted 2019 October 30. Received 2019 October 29; in original form 2019 September 10

ABSTRACT

A robust and extended characterization of the point spread function (PSF) is crucial to extract the photometric information produced by deep imaging surveys. Here, we present the extended PSFs of the Sloan Digital Sky Survey (SDSS), one of the most productive astronomical surveys of all time. By stacking ∼1000 images of individual stars with different brightness, we obtain the bidimensional SDSS PSFs extending over 8 arcmin in radius for all the SDSS filters (u, g, r, i, z). This new characterization of the SDSS PSFs is near a factor of 10 larger in extension than previous PSFs characterizations of the same survey. We found asymmetries in the shape of the PSFs caused by the drift scanning observing mode. The flux of the PSFs is larger along the drift scanning direction. Finally, we illustrate with an example how the PSF models can be used to remove the scattered light field produced by the brightest stars in the central region of the Coma cluster field. This particular example shows the huge importance of PSFs in the study of the low-surface brightness Universe, especially with the upcoming of ultradeep surveys, such as the Large Synoptic Survey Telescope (LSST). Following a reproducible science philosophy, we make all the PSF models and the scripts used to do the analysis of this paper publicly available (snapshot v0.4-0-gd966ad0).

Key words: instrumentation: detectors – methods: data analysis – techniques: image processing – techniques: photometric – galaxies: haloes.

VERIFIED

1 INTRODUCTION

The point spread function (PSF) describes the response of an imaging system to the light produced by a point source. Real PSFs have complex structures as their shapes depend on the optical path that light takes as it travels through the atmosphere and multiple optical elements, mirrors, lenses, detectors, etc. For the vast majority of astronomical works, only a tiny portion of the PSF (i.e. normally a few inner arcseconds; see e.g. Trujillo et al. 2001a, b) is characterized. In practice, however, the light of both point and extended sources are spread over the entire detector due to the effect of the PSF at large radii. Therefore, it is necessary to have a good understanding of its structure along the entire detector (typically extending over arcminutes or more).

Extended PSFs have become a vital tool to obtain precise photometric information in modern astronomical surveys. For instance, Slater, Harding & Mihos (2009) modelled the extended PSF and the internal reflections produced by the stars of the Burrell Schmidt telescope and showed that virtually all the pixels of the image are dominated by the scattered light by both stars and galaxies at 29.5 mag arcsec^{-2} (V-band). Trujillo & Fliri (2016) also characterized and used the extended PSF of the 10.4 m Gran Telescopio Canarias (GTC) telescope to model and remove the scattered light in ultradeep observations of the UGC 00180 galaxy. Even more troublesome for low-surface brightness studies is the finding (see e.g. Trujillo & Bakos 2013; Sandin 2014, 2015) that the outer regions of astronomical objects are severely affected by their own scattered light produced by the convolution with the PSF. In order to correct this effect, Karabal et al. (2017) generated the PSF and models of the internal reflections from images of the Canada–France–Hawaii Telescope (CFHT) to de-convolve a sample of three galaxies and correct them from instrumental scattered light. More recently, Román, Trujillo & Montes (2019) characterized the PSFs of the Stripe 82 survey and used them to model and correct the scattered light field produced by stars to study the optical properties of the Galactic cirri. All the above works have shown that having an extended PSF is crucial when accurate photometric and structure properties of astronomical objects at low-surface brightness levels are required.

One of the most commonly used surveys for measuring photometric properties of astronomical objects is the Sloan Sky Digital Survey (SDSS; York et al. 2000), covering 14 555 deg^2 on the sky (just over 35 per cent of the full sky) in five photometric bands (u, g, r, i, and z). Although SDSS is a relatively shallow survey compared

★ E-mail: infantesainz@gmail.com

Publication of the project

A reproducible project using Maneage will have the following (plain text) components:

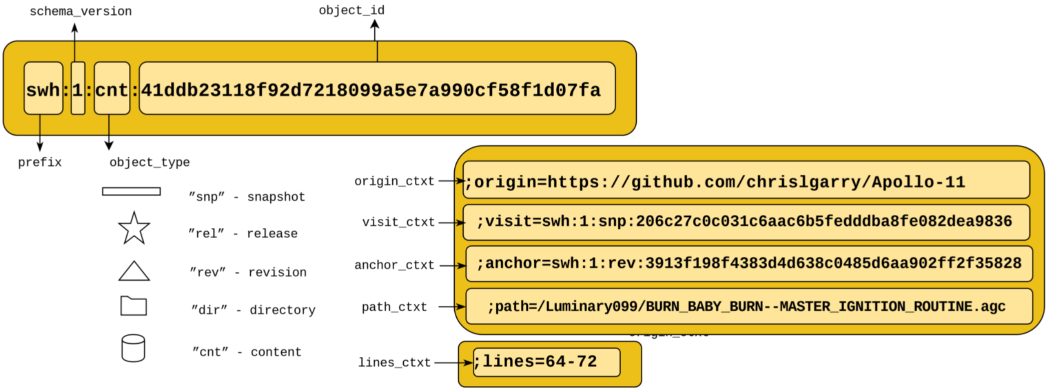
- Makefiles.
- LaTeX source files.
- Configuration files for software used in analysis.
- Scripts/programming files (e.g., Python, Shell, AWK, C).

The volume of the project's source will thus be negligible compared to a single figure in a paper (usually $\sim$ 100 kilo-bytes).

The project's pipeline (customized Maneage) can be published in

- arXiv: uploaded with the LaTeX source to always stay with the paper (for example arXiv:1909.11230, arXiv:1911.01430, arXiv:2006.03018, arXiv:2007.11779 arXiv:2010.03742, arXiv:2112.14174).
- Zenodo: Along with all the input datasets (many Gigabytes) and software (for example zenodo.6533902, also see comments in arXiv links above) and given a unique DOI.
- Software Heritage: to archive the full version-controlled history of the project. (for example swh:1:dir:33fea87068c1612daf011f161b97787b9a0df39fk)

Software Heritage IDs (SWHID); persistent identifier for source code (or any text!)



For more details, see SoftwareHeritage FAQ (at https://www.softwareheritage.org/faq)

Executing a Maneaged project (for example arXiv:2006.03018)

```
$ git clone https://gitlab.com/makhlaghi/maneage-paper    # Import the project.
```

Executing a Maneaged project (for example arXiv:2006.03018)

```
$ git clone https://gitlab.com/makhlaghi/maneage-paper    # Import the project.


$ ./project configure        # You will specify the build directory on your system,
                             # and it will build all software (about 1.5 hours).
```

Executing a Maneaged project (for example arXiv:2006.03018)

```
$ git clone https://gitlab.com/makhlaghi/maneage-paper    # Import the project.

$ ./project configure          # You will specify the build directory on your system,
                               # and it will build all software (about 1.5 hours).

$ ./project make               # Does all the analysis and makes final PDF.
```

Future prospects...

Adoption of reproducibility by many researchers will enable the following:

- A repository for education/training (PhD students, or researchers in other fields).
- Easy verification/understanding of other research projects (when necessary).
- Trivially test different steps of others' work (different configurations, software and etc).
- Science can progress incrementally (shorter papers actually building on each other!).
- Extract meta-data after the publication of a dataset (for future ontologies or vocabularies).
- Applying machine learning on reproducible research projects will allow us to solve some Big Data Challenges:
  - *Extract the relevant parameters automatically.*
  - *Translate the science to enormous samples.*
  - *Believe the results when no one will have time to reproduce.*
  - *Have confidence in results derived using machine learning or AI.*

Summary:

Maneage (https://maneage.org) is a customizable template that will for research or data reduction:

- Automatically downloads the necessary *software* and *data*.
- Builds the software in a closed environment.
- Runs the software on data to generate the final research results.
- Modification of part of the analysis will only result in re-doing that part, not the whole project.
- Using LaTeX macros, paper's figures, tables and numbers will be Automatically updated.
- The whole project is under version control (Git) to allow easy reversion to a previous state. This encourages tests/experimentation in the analysis.
- The Git commit hash of the project source, is printed in the published paper and saved on output data products. Ensuring the integrity/reproducibility of the result.
- These slides are available at https://maneage.org/pdf/slides-intro-short.pdf.
- Longer slides are available at https://maneage.org/pdf/slides-intro.pdf.
  - YouTube recording (May 2021): https://www.youtube.com/watch?v=XdhRUhoMqw0
- Matrix-protocol chat room: #maneage-general:matrix.org

For a technical description of Maneage's implementation, as well as a checklist to customize it, and tips on good practices, please see this page:
https://gitlab.com/maneage/project/-/blob/maneage/README-hacking.md