Using the IARC nextflow bioinformatics pipelines Day 2

Matthieu Foll
IARC course: May 23-24 2018

International Agency for Research on Cancer
Lyon, France

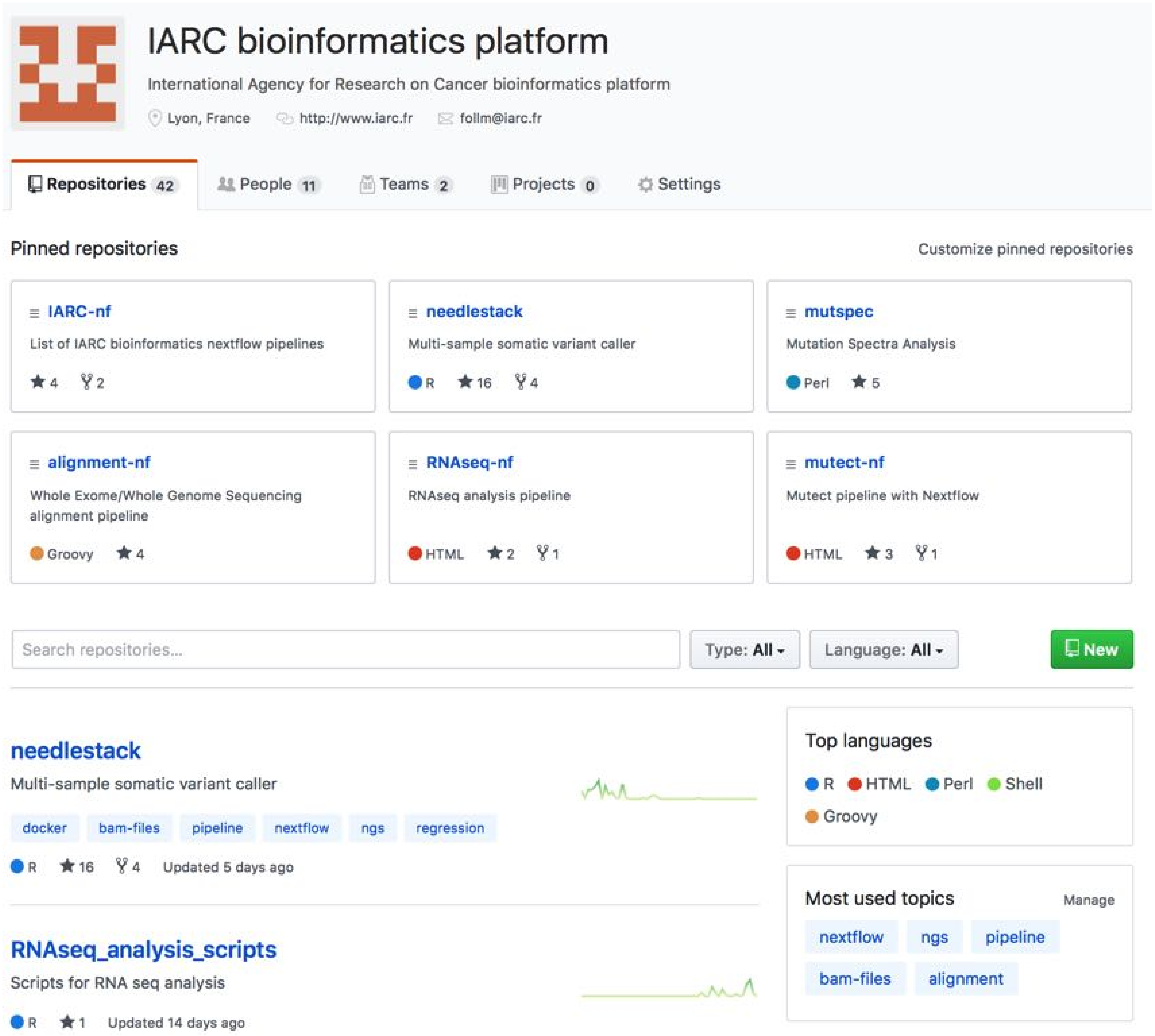
Agenda

	Wednesday 23 May
09:00-10:00	Introduction to bioinformatics pipelines, nextflow, docker, Github and the IARC organization
10:00-10:30	Practical application: running your first pipeline
10:30-11:00	Break
11:00-11-30	The hidden structure of nextflow: work folder and configuration
11:30-12:30	Practical application: configuring, crashing, resuming and debugging pipelines

	Thursday 24 May
09:00-09:30	Introduction to HPC clusters and running pipelines on a cluster.
09:30-10:30	Practical application: trace and visualise pipeline execution with log files.
10:30-11:00	Break
11:00-11h30	Introduction to the nextflow language: understanding what the pipelines are doing
11:30-12:30	Practical application: advanced usages toward reproducibility (choosing a container, Github releases and branches, modifying a pipeline etc.)

International Agency for Research on Cancer

World Health Organization



IARC bioinformatics platform

International Agency for Research on Cancer bioinformatics platform

Lyon, France http://www.iarc.fr follm@iarc.fr

Pinned repositories

IARC-nf
List of IARC bioinformatics nextflow pipelines
★ 4 2

needlestack
Multi-sample somatic variant caller
● R ★ 16 4

mutspec
Mutation Spectra Analysis
● Perl ★ 5

alignment-nf
Whole Exome/Whole Genome Sequencing alignment pipeline
● Groovy ★ 4

RNAseq-nf
RNAseq analysis pipeline
● HTML ★ 2 1

mutect-nf
Mutect pipeline with Nextflow
● HTML ★ 3 1

needlestack
Multi-sample somatic variant caller
docker bam-files pipeline nextflow ngs regression
● R ★ 16 4 Updated 5 days ago

RNAseq_analysis_scripts
Scripts for RNA seq analysis
● R ★ 1 Updated 14 days ago

Top languages
● R ● HTML ● Perl ● Shell ● Groovy

Most used topics
nextflow ngs pipeline bam-files alignment

Our philosophy

- “Do It Once, Do It Right, And Use It Everywhere”
- “Keep it simple, stupid” (KISS principle):
  - most systems work best if they are kept simple
  - simplicity should be a key goal in design
  - code easier to maintain and to understand

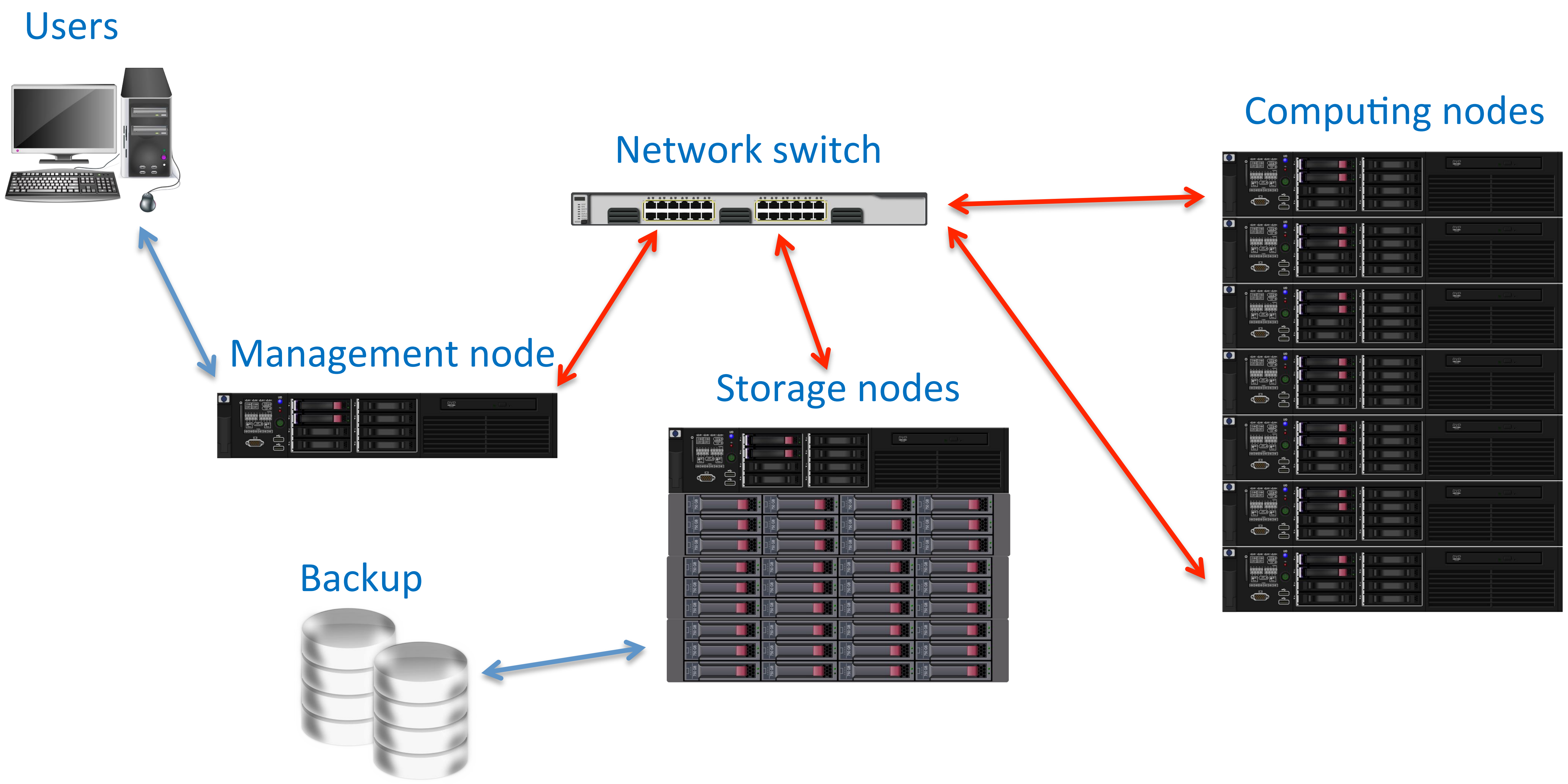
Our design

- Too much automation is not for us:
  - Hard to read, to maintain and to keep modular
  - eg: we prefer to have one alignment pipeline; one variant calling pipeline; one annotation pipeline; one QC pipeline.
- One pipeline = one GitHub repository
- Docker containers hosted on DockerHub, compatible with Singularity
- CircleCI for tests and deployment
- Standardised readme, params, help etc. (one shared template)
- Use GitHub issues and releases
- Master branch ← beta branch ← dev branch

In practice

- Entry point: GitHub group
  - https://github.com/IARCbioinfo
- One central repo references all nextflow pipelines:
  - https://github.com/IARCbioinfo/IARC-nf
  - List pipelines with a short description
  - One pipeline = one repo, ends with “-nf”
  - Common instructions to use the pipelines (install nextflow, configuration, basic usage, docker…)
- A “template-nf” nextflow “hello-world” repo

High Performance Computing (HPC) cluster



International Agency for Research on Cancer

World Health Organization

Job scheduler

- Computer program for controlling unattended background program execution of jobs
- Choose host based on:
  - Compute resource availability
  - Execution time allocated to user
  - Number of simultaneous jobs allowed for a user
  - Job priority
  - …
- SGE, SLURM, PBS, Torque, **LSF** etc.

LSF most useful commands

- `bsub command` to launch a job. Useful options:
  - `-I`: Interactive job
  - `-oo:` output file name (overwrite)
  - `-eo`: error file name (overwrite)
  - `-J`: job name
  - `-m`: choose the hosts (or list of hosts, e.g. `bsub -m "cn08 cn09 cn10"`)
  - `-n`: number of CPUs
  - `-R "rusage[mem=XXX]" -M XXX`: ask for XXX MB of RAM
- `bjobs` to monitor jobs. Useful options:
  - `-w` for the full name
  - `-l` option for more details
  - `-u username` for a given user (or `-u all` for all users)

Nextflow is doing this for you!!!

International Agency for Research on Cancer

World Health Organization

Example

```
samtools view -H NA06984.bam
```

with docker becomes:

```
docker run -it --rm -v $PWD:$PWD -w $PWD --entrypoint /bin/bash samtools_img -c
                                                              "samtools view -H NA06984.bam"
```

with LSF becomes:

Nextflow is doing this for you!!!

```
bsub -oo out.txt -eo err.txt -n 1 -R "rusage[mem=1000]" -M 1000 -J samtools
                                                             “samtools view -H NA06984.bam”
```

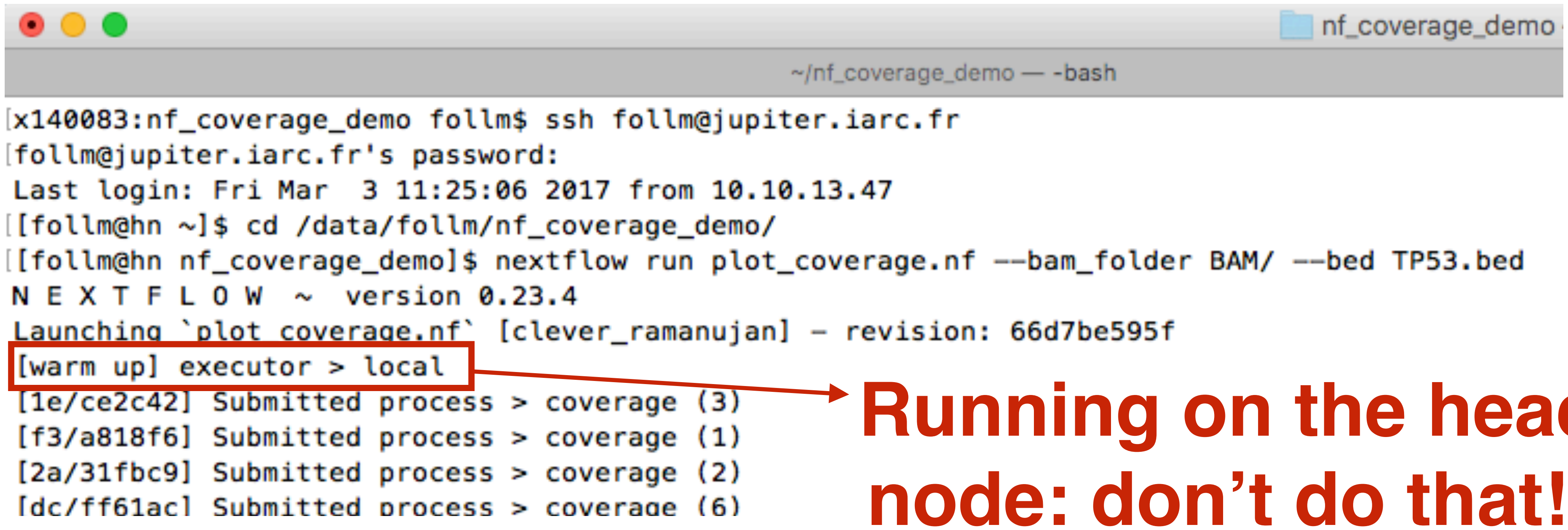
with docker and LSF becomes:

```
bsub -oo out.txt -eo err.txt -n 1 -R "rusage[mem=1000]" -M 1000 -J samtools
 "docker run -it --rm -v $PWD:$PWD -w $PWD --entrypoint /bin/bash samtools_img -c
                                                         \"samtools view -H NA06984.bam\" "
```

Hum… but we use singularity on our cluster… The syntax is actually different!
(`singularity exec` …)

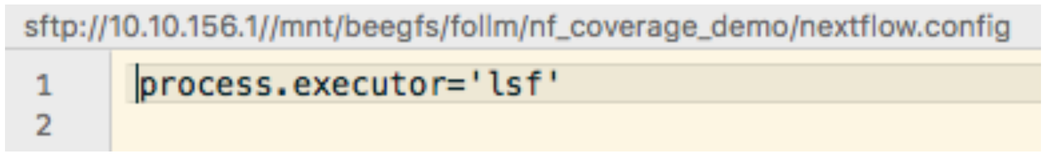
International Agency for Research on Cancer
World Health Organization

Running on a cluster



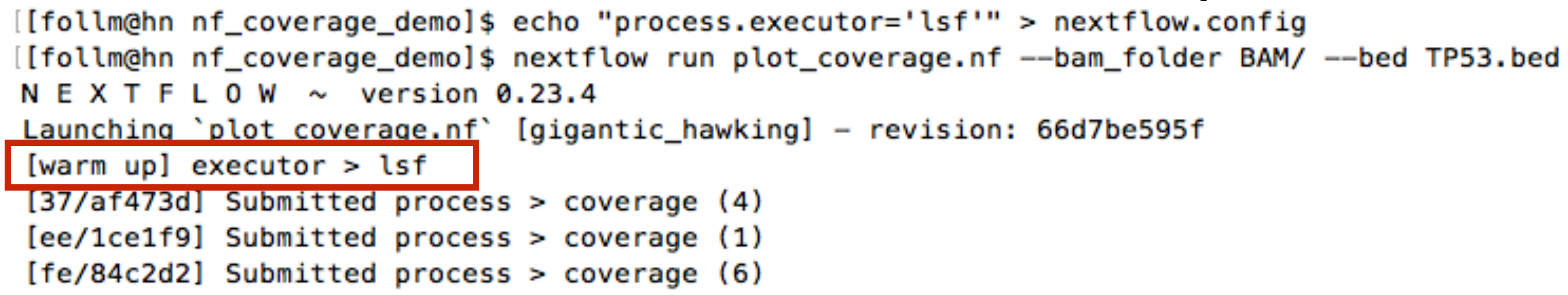
```
x140083:nf_coverage_demo follm$ ssh follm@jupiter.iarc.fr
follm@jupiter.iarc.fr's password:
Last login: Fri Mar  3 11:25:06 2017 from 10.10.13.47
[follm@hn ~]$ cd /data/follm/nf_coverage_demo/
[follm@hn nf_coverage_demo]$ nextflow run plot_coverage.nf --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Launching `plot_coverage.nf` [clever_ramanujan] - revision: 66d7be595f
[warm up] executor > local
[1e/ce2c42] Submitted process > coverage (3)
[f3/a818f6] Submitted process > coverage (1)
[2a/31fbc9] Submitted process > coverage (2)
[dc/ff61ac] Submitted process > coverage (6)
```

Running on the head-node: don't do that!!!



sftp://10.10.156.1//mnt/beegfs/follm/nf_coverage_demo/nextflow.config

```
process.executor='lsf'
```



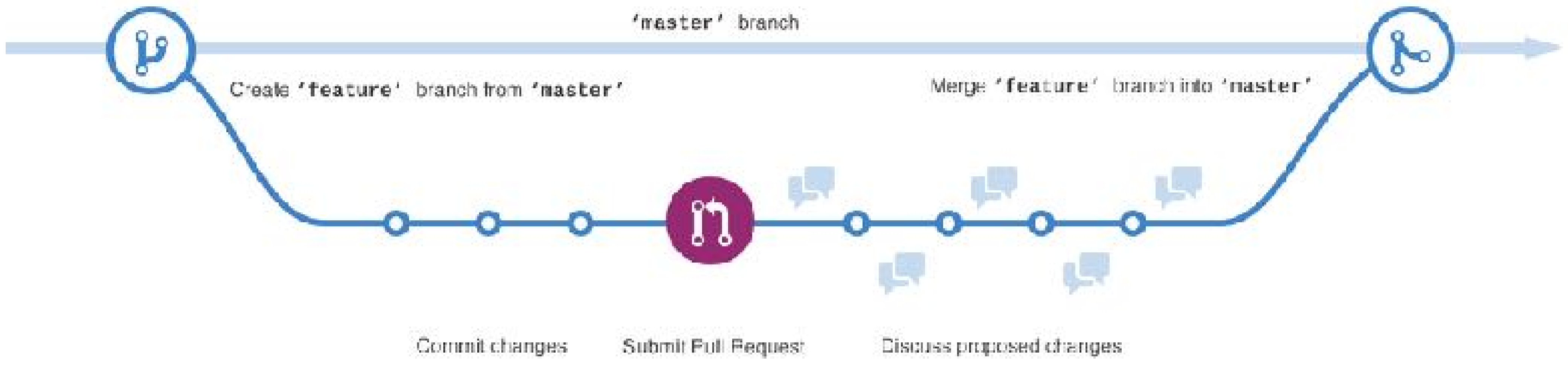
```
[follm@hn nf_coverage_demo]$ echo "process.executor='lsf'" > nextflow.config
[follm@hn nf_coverage_demo]$ nextflow run plot_coverage.nf --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Launching `plot_coverage.nf` [gigantic_hawking] - revision: 66d7be595f
[warm up] executor > lsf
[37/af473d] Submitted process > coverage (4)
[ee/1ce1f9] Submitted process > coverage (1)
[fe/84c2d2] Submitted process > coverage (6)
```

Control queue size with **-qs** option

International Agency for Research on Cancer
World Health Organization

Speaking GitHub language

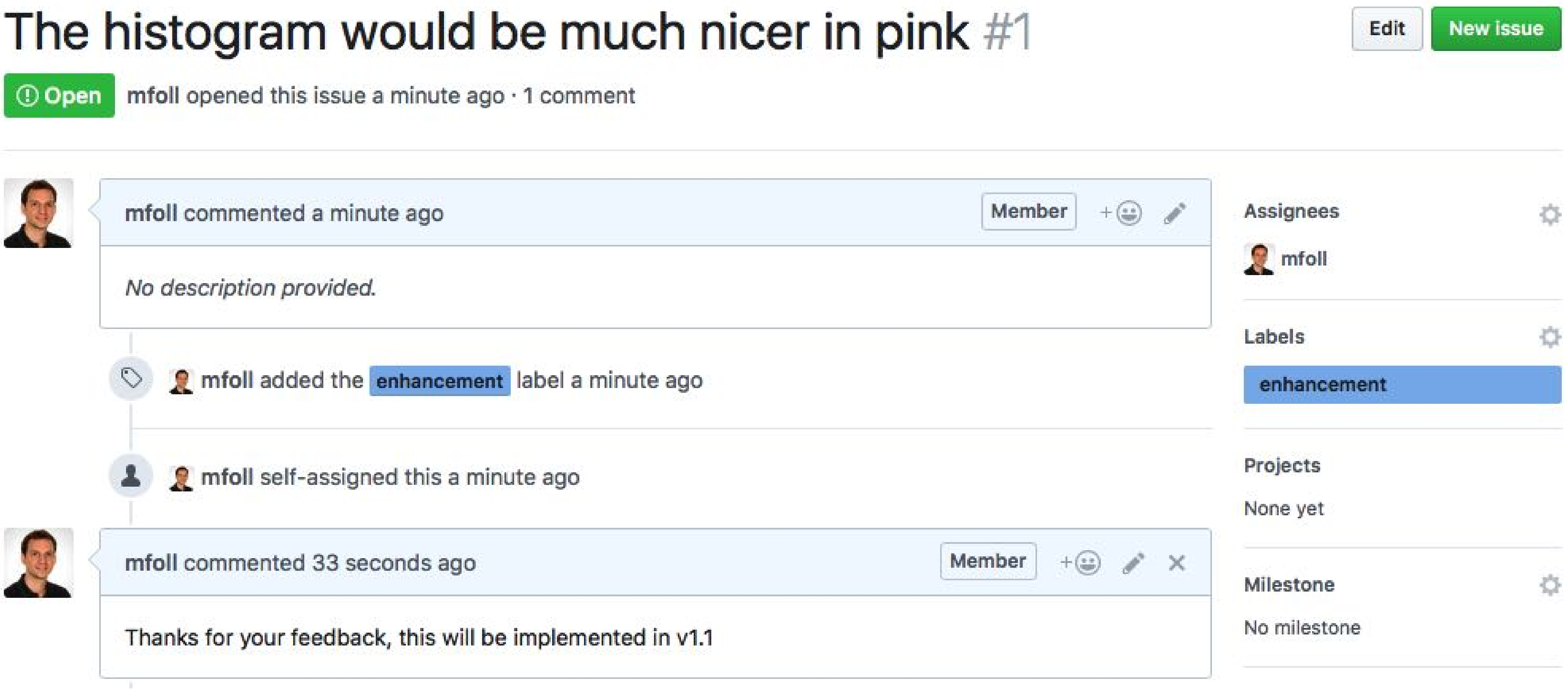
- ***Branches***: master branch considered definitive, use other branches to experiment.
- Changes are called ***commit****:* one can ***push*** or ***pull*** commits to repository or a branch. Commits have unique IDs (hash).
- ***Pull requests***: ask someone to ***merge*** the commits you did into their branch.
- ***Releases*** are user friendly commit IDs with ***tag*** names
- ***Issues*** are used to discuss bugs, features etc.


'master' branch
Create 'feature' branch from 'master'
Merge 'feature' branch into 'master'
Commit changes
Submit Pull Request
Discuss proposed changes

Nextflow and GitHub

- Any branch, tag or commit ID can be used to specify the revision you want to execute using the `-r` option:
  ```
  nextflow run nextflow-io/hello -r mybranch
  nextflow run nextflow-io/hello -r v1.1
  ```
- Be careful with docker version used:
  - Docker also has `tags` and hash IDs
  - If well done, a particular pipeline version will point to the right container automatically using the config file
  - If not you can manually specify the docker tag using:
    ```
    nextflow run -with-docker user/repo:tag
    ```

Social coding



The histogram would be much nicer in pink #1

Open mfoll opened this issue a minute ago · 1 comment

mfoll commented a minute ago

No description provided.

mfoll added the enhancement label a minute ago

mfoll self-assigned this a minute ago

mfoll commented 33 seconds ago

Thanks for your feedback, this will be implemented in v1.1

Assignees: mfoll

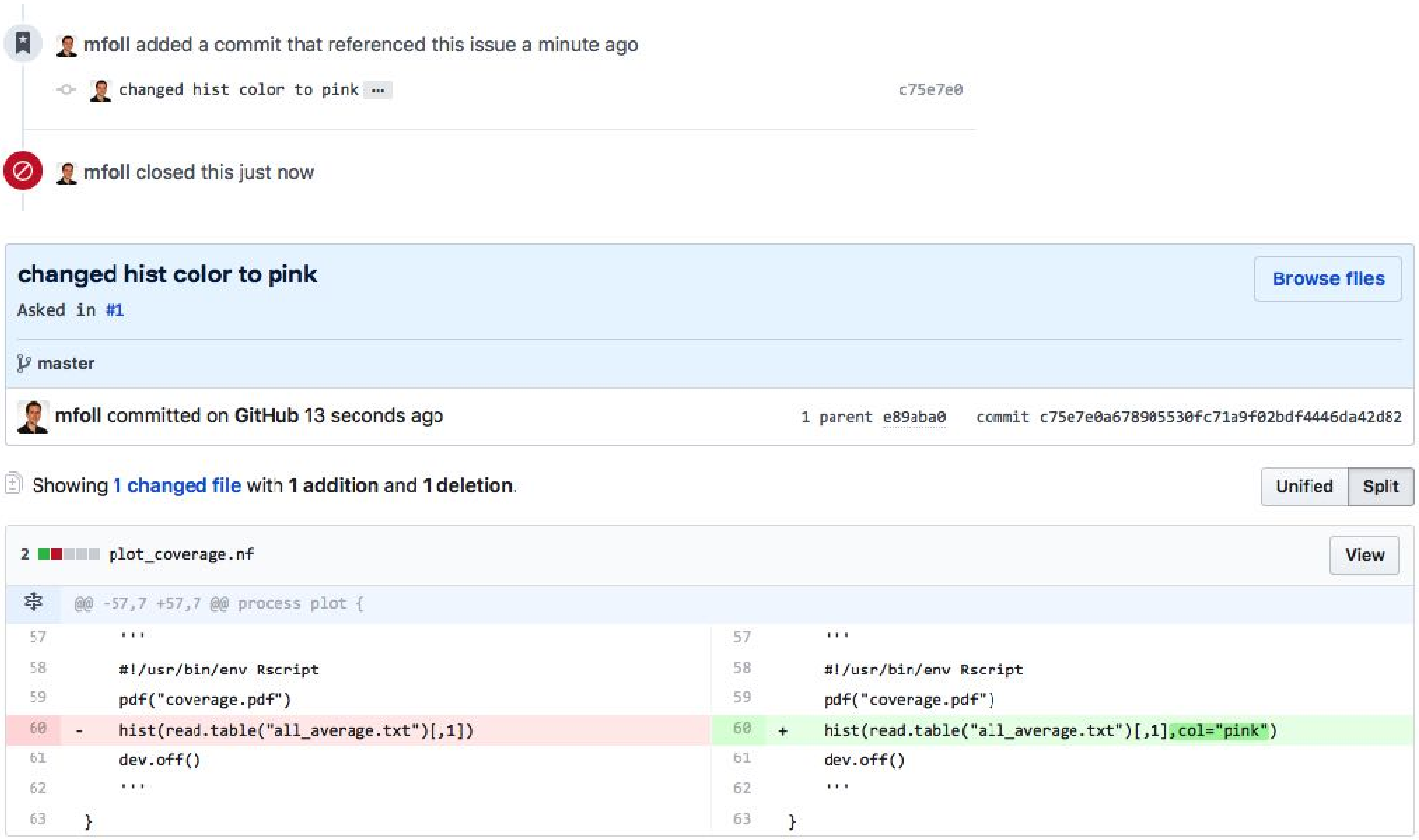
Labels: enhancement

Projects: None yet

Milestone: No milestone

International Agency for Research on Cancer

World Health Organization



mfoll added a commit that referenced this issue a minute ago

changed hist color to pink ··· c75e7e0

mfoll closed this just now

changed hist color to pink

Asked in #1

master

mfoll committed on GitHub 13 seconds ago 1 parent e89aba0 commit c75e7e0a678905530fc71a9f02bdf4446da42d82

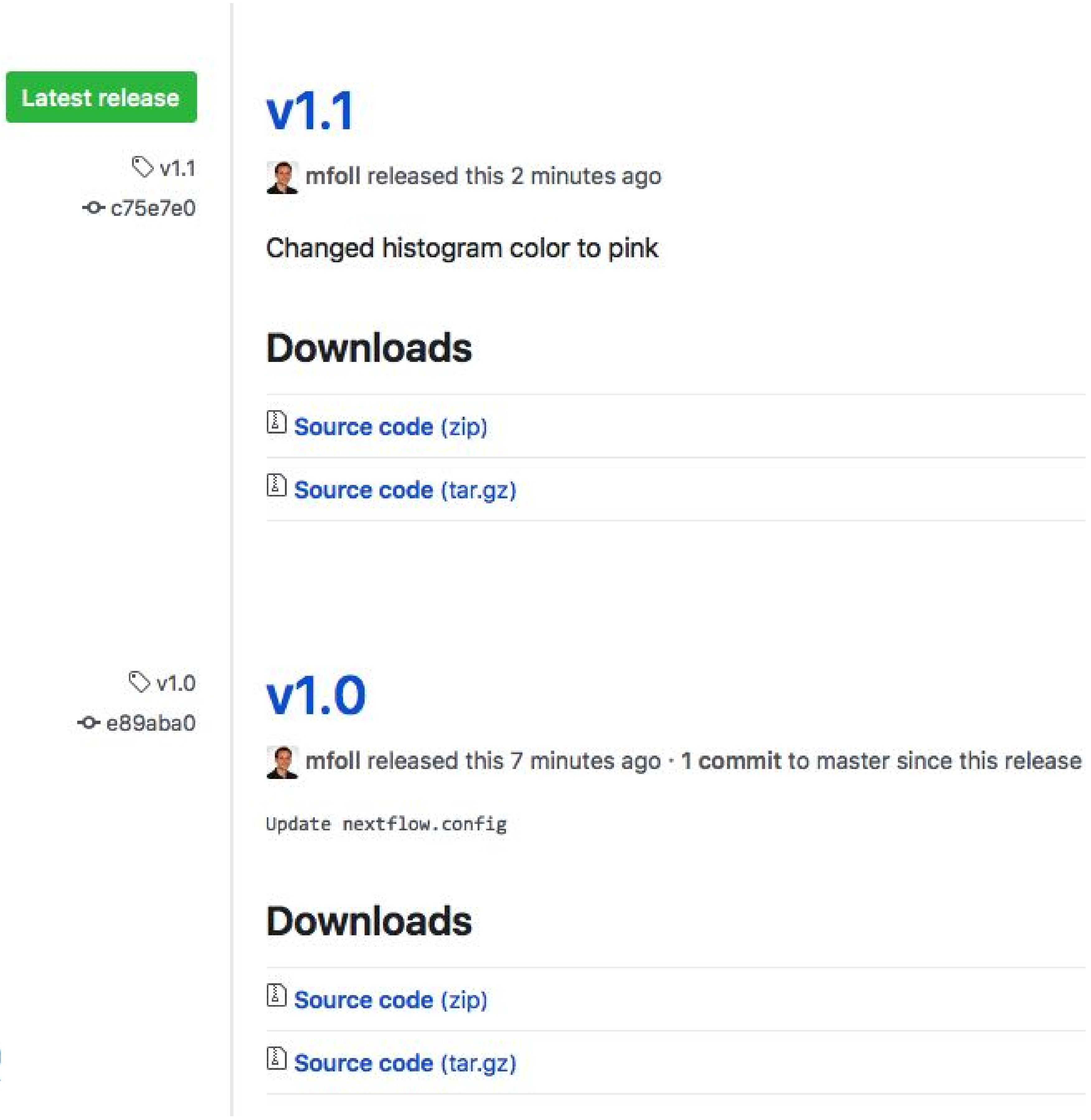
Showing 1 changed file with 1 addition and 1 deletion.

2 plot_coverage.nf

```
@@ -57,7 +57,7 @@ process plot {
     '''
     #!/usr/bin/env Rscript
     pdf("coverage.pdf")
-    hist(read.table("all_average.txt")[,1])
+    hist(read.table("all_average.txt")[,1],col="pink")
     dev.off()
     '''
 }
```

International Agency for Research on Cancer

World Health Organization



Latest release

v1.1

c75e7e0

v1.1

mfoll released this 2 minutes ago

Changed histogram color to pink

Downloads

Source code (zip)

Source code (tar.gz)

v1.0

e89aba0

v1.0

mfoll released this 7 minutes ago · **1 commit** to master since this release

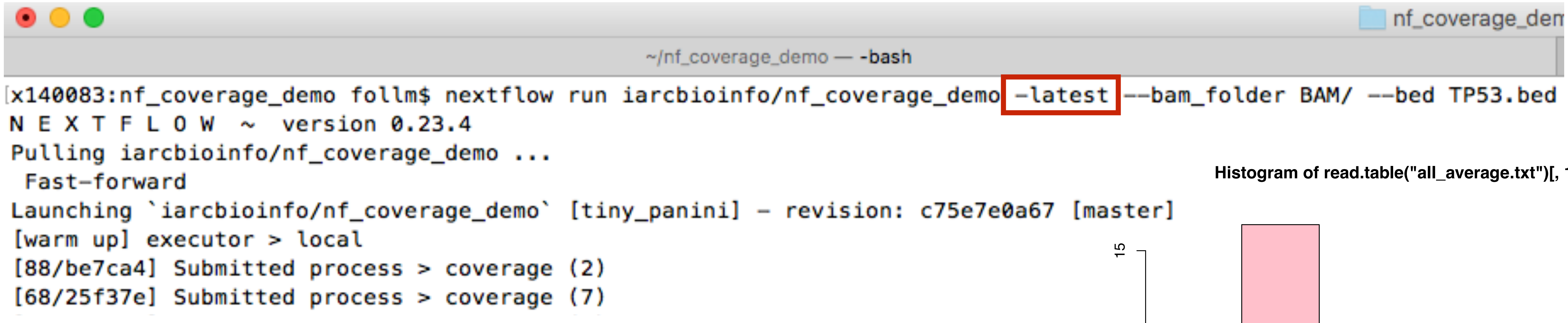
```
Update nextflow.config
```

Downloads

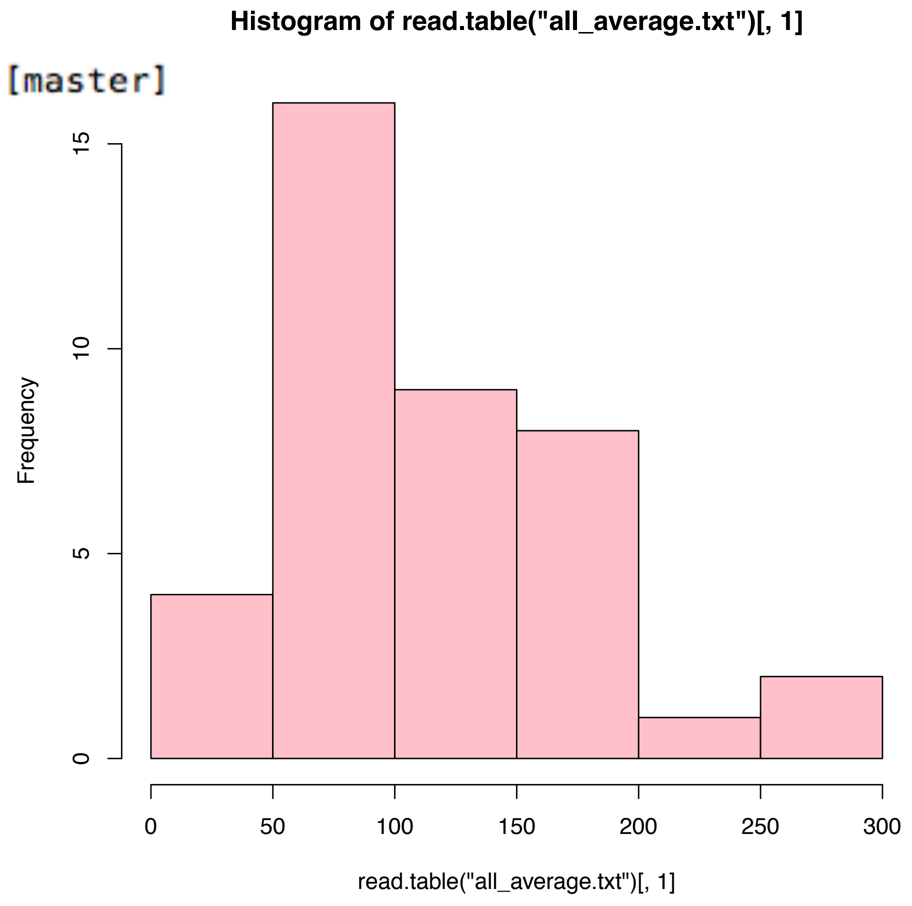
Source code (zip)

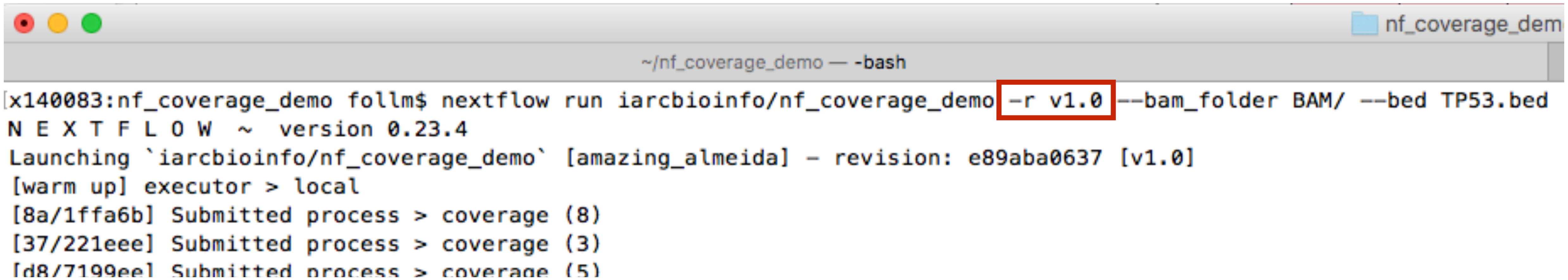
Source code (tar.gz)

Running specific versions



```
~/nf_coverage_demo — -bash
x140083:nf_coverage_demo follm$ nextflow run iarcbioinfo/nf_coverage_demo -latest --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Pulling iarcbioinfo/nf_coverage_demo ...
 Fast-forward
Launching `iarcbioinfo/nf_coverage_demo` [tiny_panini] - revision: c75e7e0a67 [master]
[warm up] executor > local
[88/be7ca4] Submitted process > coverage (2)
[68/25f37e] Submitted process > coverage (7)
```



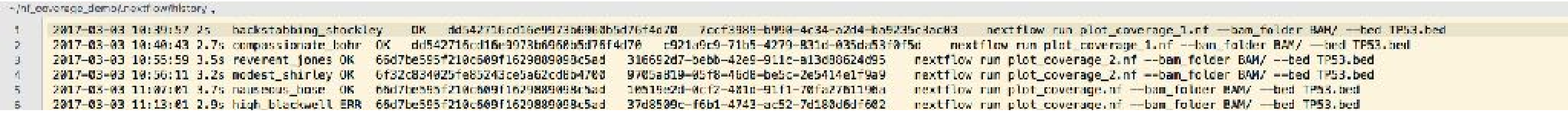


```
~/nf_coverage_demo — -bash
x140083:nf_coverage_demo follm$ nextflow run iarcbioinfo/nf_coverage_demo -r v1.0 --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Launching `iarcbioinfo/nf_coverage_demo` [amazing_almeida] - revision: e89aba0637 [v1.0]
[warm up] executor > local
[8a/1ffa6b] Submitted process > coverage (8)
[37/221eee] Submitted process > coverage (3)
[d8/7199ee] Submitted process > coverage (5)
```

Nextflow outputs

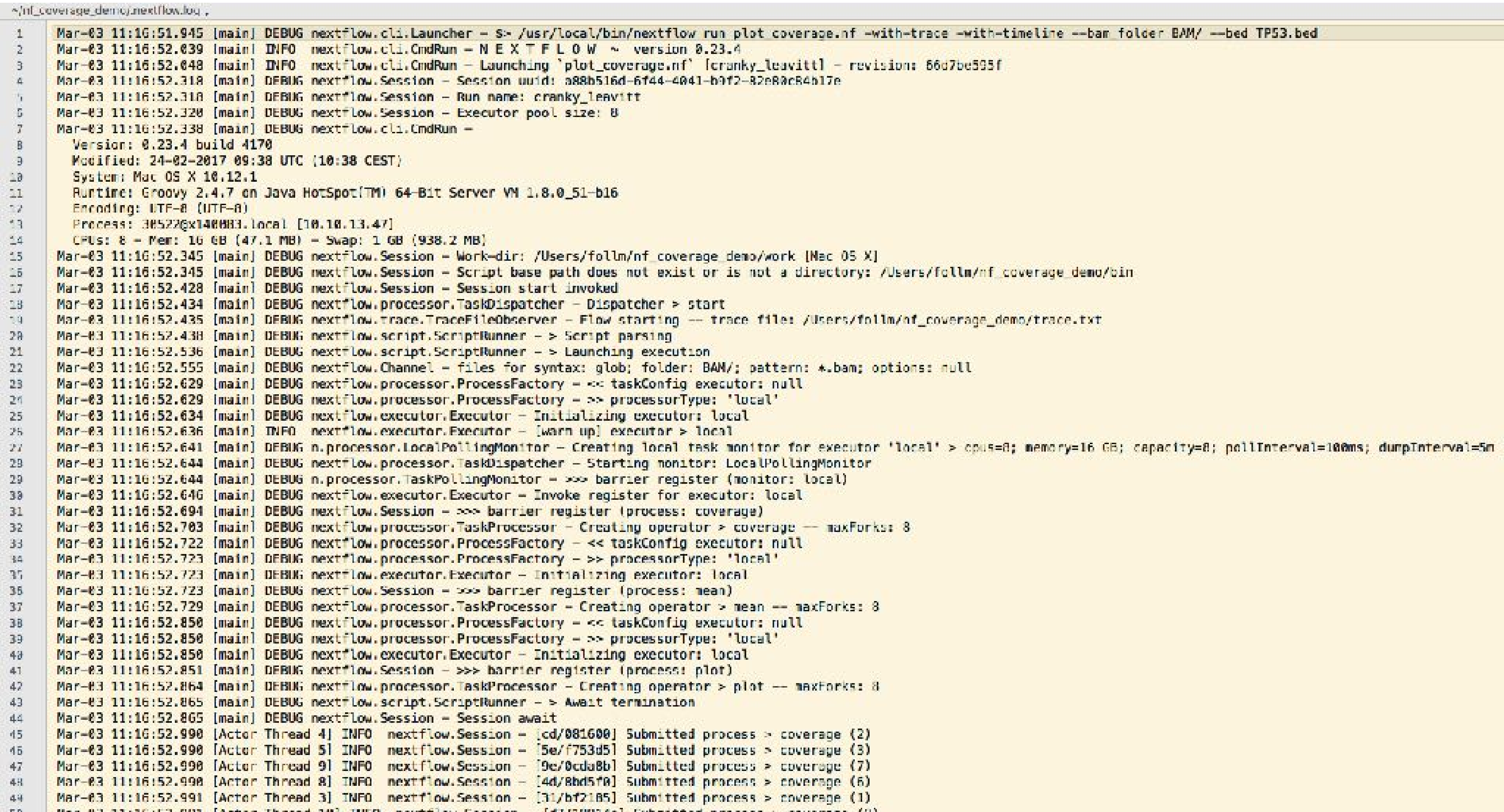
- stdout (what you see on the screen)
- work folder
- log file: `.nextflow.log`
- history: `.nextflow/history`
- Optional:
    - trace
    - timeline
    - report
    - email notification

History and log



~/nf_coverage_demo/.nextflow.history

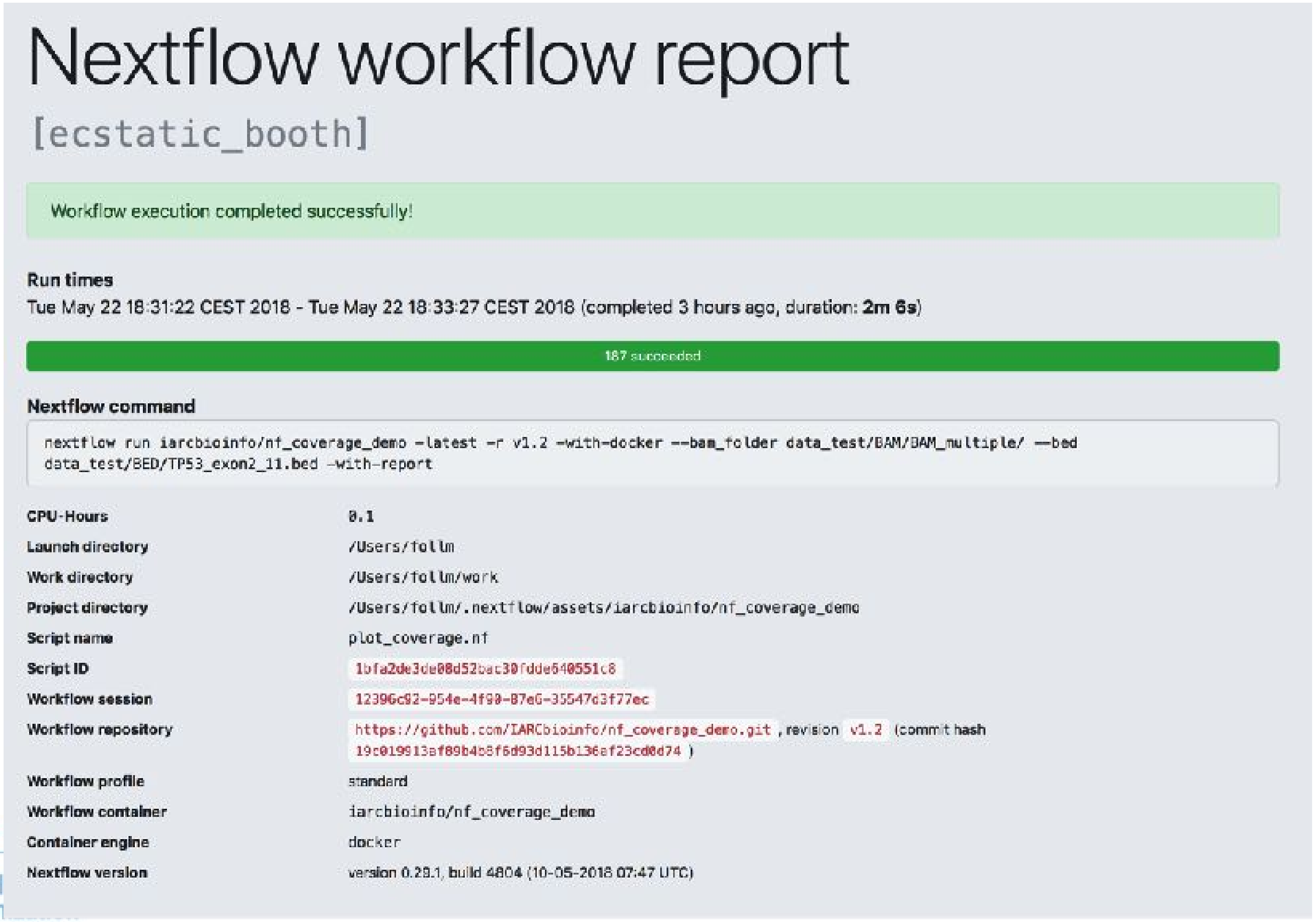
```
2017-03-03 10:39:57 2s   backstabbing_shockley   OK   dd542716cd16e9973b6968b5d76f4d70   7ccf3985-b990-4c34-a2d4-ba5235c3ac03   nextflow run plot_coverage_1.nf --bam_folder BAM/ --bed TP53.bed
2017-03-03 10:40:43 2.7s compassionate_bohr  OK   dd542716cd16e9973b6968b5d76f4d70   c921a9c9-71b5-4279-831d-035da53f0f5d   nextflow run plot_coverage_1.nf --bam_folder BAM/ --bed TP53.bed
2017-03-03 10:55:59 3.5s reverent_jones OK   66d7be595f210c509f1629089059c5ad   316692d7-bebb-42e9-911c-a13d08624d95   nextflow run plot_coverage_2.nf --bam_folder BAM/ --bed TP53.bed
2017-03-03 10:56:11 3.2s modest_shirley OK   6f32c834025fe8524Gce5a62cd6b4700   9705a819-05f8-46d8-be5c-2e5414e1f9a9   nextflow run plot_coverage_2.nf --bam_folder BAM/ --bed TP53.bed
2017-03-03 11:07:01 3.7s nauseous_hose  OK   66d7be595f210c509f1629089059c5ad   10515e2d-0cf2-481c-91f1-70fa27611984   nextflow run plot_coverage.nf --bam_folder BAM/ --bed TP53.bed
2017-03-03 11:13:01 2.9s high_blackwell ERR  66d7be595f210c509f1629089059c5ad   37d8509c-f6b1-4743-ac52-7d188d6df682   nextflow run plot_coverage.nf --bam_folder BAM/ --bed TP53.bed
```



~/nf_coverage_demo/.nextflow.log

```
Mar-03 11:16:51.945 [main] DEBUG nextflow.cli.Launcher - $> /usr/local/bin/nextflow run plot_coverage.nf -with-trace -with-timeline --bam_folder BAM/ --bed TP53.bed
Mar-03 11:16:52.039 [main] INFO  nextflow.cli.CmdRun - N E X T F L O W  ~  version 0.23.4
Mar-03 11:16:52.048 [main] INFO  nextflow.cli.CmdRun - Launching `plot_coverage.nf` [cranky_leavitt] - revision: 66d7be595f
Mar-03 11:16:52.318 [main] DEBUG nextflow.Session - Session uuid: a88b516d-6f44-4041-b0f2-82e80c64b17e
Mar-03 11:16:52.318 [main] DEBUG nextflow.Session - Run name: cranky_leavitt
Mar-03 11:16:52.320 [main] DEBUG nextflow.Session - Executor pool size: 8
Mar-03 11:16:52.338 [main] DEBUG nextflow.cli.CmdRun -
  Version: 0.23.4 build 4170
  Modified: 24-02-2017 09:38 UTC (10:38 CEST)
  System: Mac OS X 10.12.1
  Runtime: Groovy 2.4.7 on Java HotSpot(TM) 64-Bit Server VM 1.8.0_51-b16
  Encoding: UTF-8 (UTF-8)
  Process: 38522@x140003.local [10.10.13.47]
  CPUs: 8 - Mem: 16 GB (47.1 MB) - Swap: 1 GB (938.2 MB)
Mar-03 11:16:52.345 [main] DEBUG nextflow.Session - Work-dir: /Users/follm/nf_coverage_demo/work [Mac OS X]
Mar-03 11:16:52.345 [main] DEBUG nextflow.Session - Script base path does not exist or is not a directory: /Users/follm/nf_coverage_demo/bin
Mar-03 11:16:52.428 [main] DEBUG nextflow.Session - Session start invoked
Mar-03 11:16:52.434 [main] DEBUG nextflow.processor.TaskDispatcher - Dispatcher > start
Mar-03 11:16:52.435 [main] DEBUG nextflow.trace.TraceFileObserver - Flow starting -- trace file: /Users/follm/nf_coverage_demo/trace.txt
Mar-03 11:16:52.438 [main] DEBUG nextflow.script.ScriptRunner - > Script parsing
Mar-03 11:16:52.536 [main] DEBUG nextflow.script.ScriptRunner - > Launching execution
Mar-03 11:16:52.555 [main] DEBUG nextflow.Channel - files for syntax: glob; folder: BAM/; pattern: *.bam; options: null
Mar-03 11:16:52.629 [main] DEBUG nextflow.processor.ProcessFactory - << taskConfig executor: null
Mar-03 11:16:52.629 [main] DEBUG nextflow.processor.ProcessFactory - >> processorType: 'local'
Mar-03 11:16:52.634 [main] DEBUG nextflow.executor.Executor - Initializing executor: local
Mar-03 11:16:52.636 [main] INFO  nextflow.executor.Executor - [warm up] executor > local
Mar-03 11:16:52.641 [main] DEBUG n.processor.LocalPollingMonitor - Creating local task monitor for executor 'local' > cpus=8; memory=16 GB; capacity=8; pollInterval=100ms; dumpInterval=5m
Mar-03 11:16:52.644 [main] DEBUG nextflow.processor.TaskDispatcher - Starting monitor: LocalPollingMonitor
Mar-03 11:16:52.644 [main] DEBUG n.processor.TaskPollingMonitor - >>> barrier register (monitor: local)
Mar-03 11:16:52.646 [main] DEBUG nextflow.executor.Executor - Invoke register for executor: local
Mar-03 11:16:52.694 [main] DEBUG nextflow.Session - >>> barrier register (process: coverage)
Mar-03 11:16:52.703 [main] DEBUG nextflow.processor.TaskProcessor - Creating operator > coverage -- maxForks: 8
Mar-03 11:16:52.722 [main] DEBUG nextflow.processor.ProcessFactory - << taskConfig executor: null
Mar-03 11:16:52.723 [main] DEBUG nextflow.processor.ProcessFactory - >> processorType: 'local'
Mar-03 11:16:52.723 [main] DEBUG nextflow.executor.Executor - Initializing executor: local
Mar-03 11:16:52.723 [main] DEBUG nextflow.Session - >>> barrier register (process: mean)
Mar-03 11:16:52.729 [main] DEBUG nextflow.processor.TaskProcessor - Creating operator > mean -- maxForks: 8
Mar-03 11:16:52.850 [main] DEBUG nextflow.processor.ProcessFactory - << taskConfig executor: null
Mar-03 11:16:52.850 [main] DEBUG nextflow.processor.ProcessFactory - >> processorType: 'local'
Mar-03 11:16:52.850 [main] DEBUG nextflow.executor.Executor - Initializing executor: local
Mar-03 11:16:52.851 [main] DEBUG nextflow.Session - >>> barrier register (process: plot)
Mar-03 11:16:52.864 [main] DEBUG nextflow.processor.TaskProcessor - Creating operator > plot -- maxForks: 8
Mar-03 11:16:52.865 [main] DEBUG nextflow.script.ScriptRunner - > Await termination
Mar-03 11:16:52.865 [main] DEBUG nextflow.Session - Session await
Mar-03 11:16:52.990 [Actor Thread 4] INFO  nextflow.Session - [cd/081600] Submitted process > coverage (2)
Mar-03 11:16:52.990 [Actor Thread 5] INFO  nextflow.Session - [5e/f753d5] Submitted process > coverage (3)
Mar-03 11:16:52.990 [Actor Thread 9] INFO  nextflow.Session - [9e/0cda8b] Submitted process > coverage (7)
Mar-03 11:16:52.990 [Actor Thread 8] INFO  nextflow.Session - [4d/8bd5f0] Submitted process > coverage (6)
Mar-03 11:16:52.991 [Actor Thread 3] INFO  nextflow.Session - [31/bf2105] Submitted process > coverage (1)
```

Report (`-with-report`)



Nextflow workflow report

[ecstatic_booth]

Workflow execution completed successfully!

Run times

Tue May 22 18:31:22 CEST 2018 - Tue May 22 18:33:27 CEST 2018 (completed 3 hours ago, duration: **2m 6s**)

187 succeeded

Nextflow command

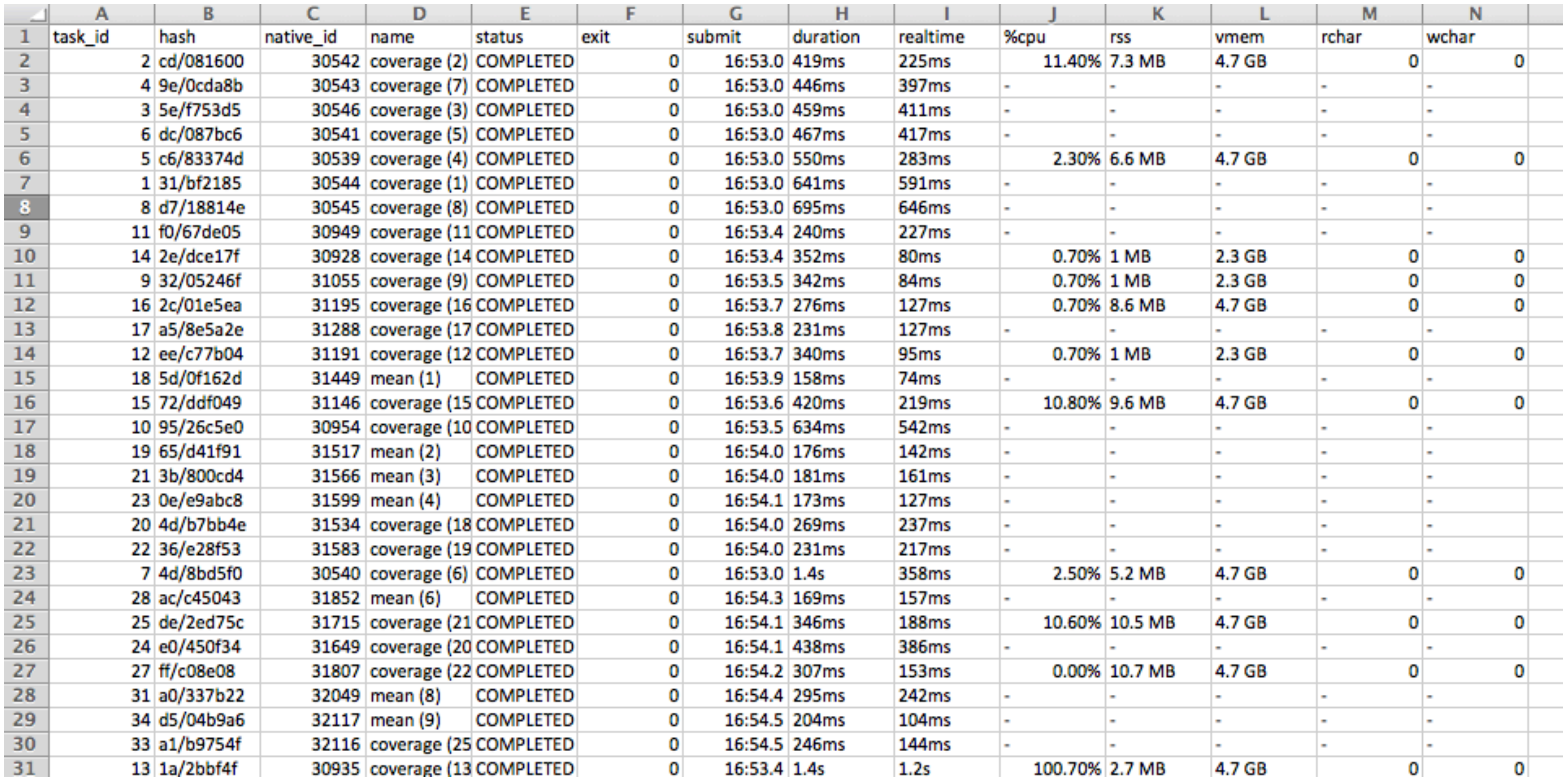
```
nextflow run iarcbioinfo/nf_coverage_demo -latest -r v1.2 -with-docker --bam_folder data_test/BAM/BAM_multiple/ --bed
data_test/BED/TP53_exon2_11.bed -with-report
```

CPU-Hours	`0.1`
Launch directory	`/Users/follm`
Work directory	`/Users/follm/work`
Project directory	`/Users/follm/.nextflow/assets/iarcbioinfo/nf_coverage_demo`
Script name	`plot_coverage.nf`
Script ID	`1bfa2de3de08d52bac30fdde640551c8`
Workflow session	`12396c92-954e-4f90-87e6-35547d3f77ec`
Workflow repository	`https://github.com/IARCbioinfo/nf_coverage_demo.git` , revision `v1.2` (commit hash `19c019913af89b4b8f6d93d115b136af23cd0d74`)
Workflow profile	standard
Workflow container	`iarcbioinfo/nf_coverage_demo`
Container engine	`docker`
Nextflow version	version 0.29.1, build 4804 (10-05-2018 07:47 UTC)

International

World Organ

Trace (-with-trace)



task_id	hash	native_id	name	status	exit	submit	duration	realtime	%cpu	rss	vmem	rchar	wchar
2	cd/081600	30542	coverage (2)	COMPLETED	0	16:53.0	419ms	225ms	11.40%	7.3 MB	4.7 GB	0	0
4	9e/0cda8b	30543	coverage (7)	COMPLETED	0	16:53.0	446ms	397ms	-	-	-	-	-
3	5e/f753d5	30546	coverage (3)	COMPLETED	0	16:53.0	459ms	411ms	-	-	-	-	-
6	dc/087bc6	30541	coverage (5)	COMPLETED	0	16:53.0	467ms	417ms	-	-	-	-	-
5	c6/83374d	30539	coverage (4)	COMPLETED	0	16:53.0	550ms	283ms	2.30%	6.6 MB	4.7 GB	0	0
1	31/bf2185	30544	coverage (1)	COMPLETED	0	16:53.0	641ms	591ms	-	-	-	-	-
8	d7/18814e	30545	coverage (8)	COMPLETED	0	16:53.0	695ms	646ms	-	-	-	-	-
11	f0/67de05	30949	coverage (11	COMPLETED	0	16:53.4	240ms	227ms	-	-	-	-	-
14	2e/dce17f	30928	coverage (14	COMPLETED	0	16:53.4	352ms	80ms	0.70%	1 MB	2.3 GB	0	0
9	32/05246f	31055	coverage (9)	COMPLETED	0	16:53.5	342ms	84ms	0.70%	1 MB	2.3 GB	0	0
16	2c/01e5ea	31195	coverage (16	COMPLETED	0	16:53.7	276ms	127ms	0.70%	8.6 MB	4.7 GB	0	0
17	a5/8e5a2e	31288	coverage (17	COMPLETED	0	16:53.8	231ms	127ms	-	-	-	-	-
12	ee/c77b04	31191	coverage (12	COMPLETED	0	16:53.7	340ms	95ms	0.70%	1 MB	2.3 GB	0	0
18	5d/0f162d	31449	mean (1)	COMPLETED	0	16:53.9	158ms	74ms	-	-	-	-	-
15	72/ddf049	31146	coverage (15	COMPLETED	0	16:53.6	420ms	219ms	10.80%	9.6 MB	4.7 GB	0	0
10	95/26c5e0	30954	coverage (10	COMPLETED	0	16:53.5	634ms	542ms	-	-	-	-	-
19	65/d41f91	31517	mean (2)	COMPLETED	0	16:54.0	176ms	142ms	-	-	-	-	-
21	3b/800cd4	31566	mean (3)	COMPLETED	0	16:54.0	181ms	161ms	-	-	-	-	-
23	0e/e9abc8	31599	mean (4)	COMPLETED	0	16:54.1	173ms	127ms	-	-	-	-	-
20	4d/b7bb4e	31534	coverage (18	COMPLETED	0	16:54.0	269ms	237ms	-	-	-	-	-
22	36/e28f53	31583	coverage (19	COMPLETED	0	16:54.0	231ms	217ms	-	-	-	-	-
7	4d/8bd5f0	30540	coverage (6)	COMPLETED	0	16:53.0	1.4s	358ms	2.50%	5.2 MB	4.7 GB	0	0
28	ac/c45043	31852	mean (6)	COMPLETED	0	16:54.3	169ms	157ms	-	-	-	-	-
25	de/2ed75c	31715	coverage (21	COMPLETED	0	16:54.1	346ms	188ms	10.60%	10.5 MB	4.7 GB	0	0
24	e0/450f34	31649	coverage (20	COMPLETED	0	16:54.1	438ms	386ms	-	-	-	-	-
27	ff/c08e08	31807	coverage (22	COMPLETED	0	16:54.2	307ms	153ms	0.00%	10.7 MB	4.7 GB	0	0
31	a0/337b22	32049	mean (8)	COMPLETED	0	16:54.4	295ms	242ms	-	-	-	-	-
34	d5/04b9a6	32117	mean (9)	COMPLETED	0	16:54.5	204ms	104ms	-	-	-	-	-
33	a1/b9754f	32116	coverage (25	COMPLETED	0	16:54.5	246ms	144ms	-	-	-	-	-
13	1a/2bbf4f	30935	coverage (13	COMPLETED	0	16:53.4	1.4s	1.2s	100.70%	2.7 MB	4.7 GB	0	0

International Agency for Research on Cancer

World Health Organization

Practical

- Run pipelines on the cluster using Singularity `-with-singularity`
- Explore all nextflow outputs/logs, including in the `work` directory
- Try to run without docker/singularity
- Try to run v1.0, still with docker/singularity
- Try different combinations of pipeline version and container version
- Look at the help of the `nextflow` commands to find useful options (in particular `run -latest, -qs`, `-bg`)
- Look at the `clone`, `drop`, `list` and `pull` commands to manage pipelines
- Try to modify a pipeline (feel free to send me a PR!)
- Check if a new version of nextflow itself is available using `nextflow self-update`

International Agency for Research on Cancer

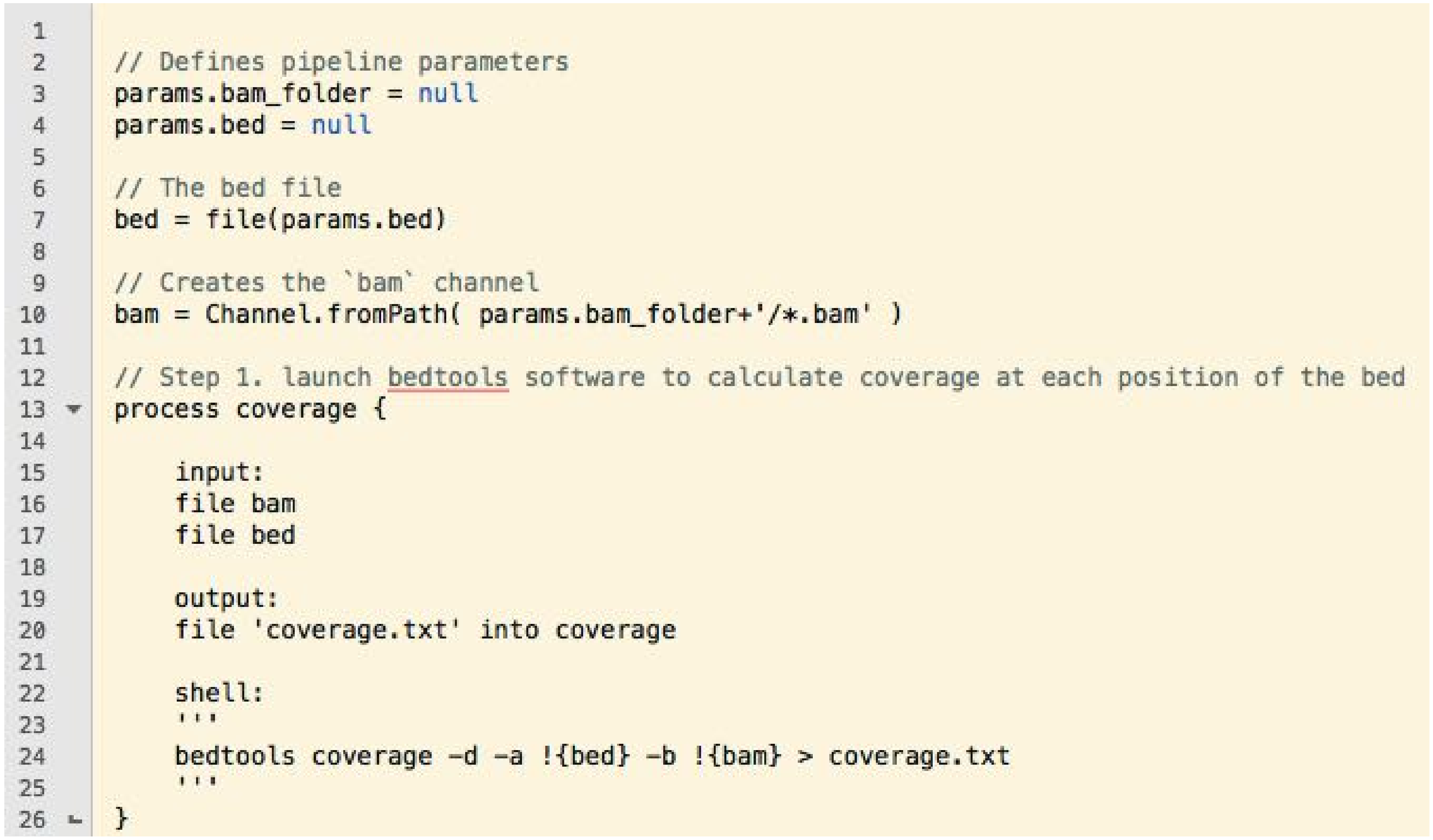
World Health Organization

Dataflow programming

- Traditionally, a program is a series of operations happening in a specific order (“sequential programming”)
- Dataflow programming emphasizes the movement of data and models programs as a series of connections:
  - Explicitly defined inputs and outputs connect operations, which function like black boxes.
  - An operation runs as soon as all of its inputs become valid. Thus, dataflow languages are inherently parallel.

https://en.wikipedia.org/wiki/Dataflow_programming

Our first process

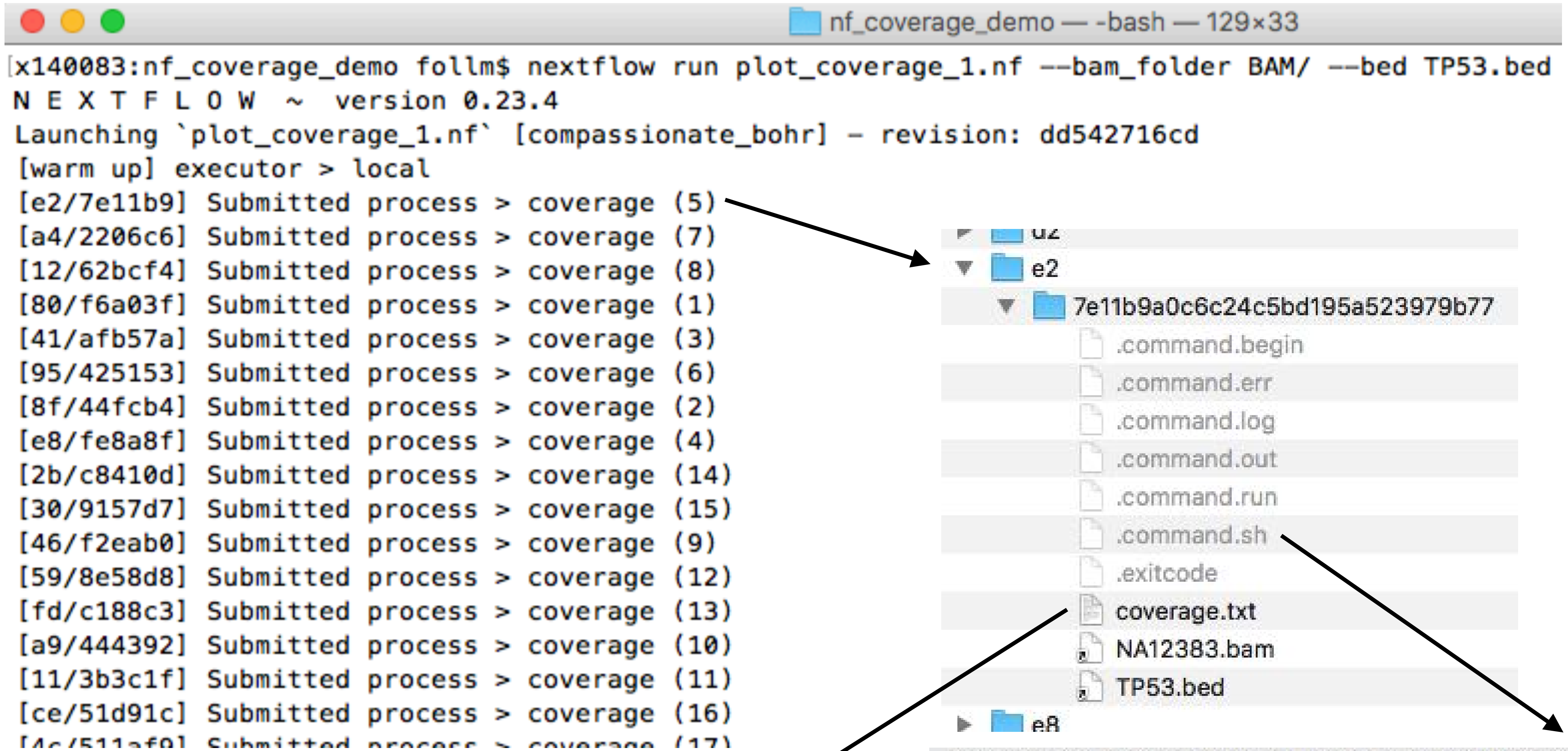


```
// Defines pipeline parameters
params.bam_folder = null
params.bed = null

// The bed file
bed = file(params.bed)

// Creates the `bam` channel
bam = Channel.fromPath( params.bam_folder+'/*.bam' )

// Step 1. launch bedtools software to calculate coverage at each position of the bed
process coverage {

    input:
    file bam
    file bed

    output:
    file 'coverage.txt' into coverage

    shell:
    '''
    bedtools coverage -d -a !{bed} -b !{bam} > coverage.txt
    '''
}
```

International Agency for Research on Cancer

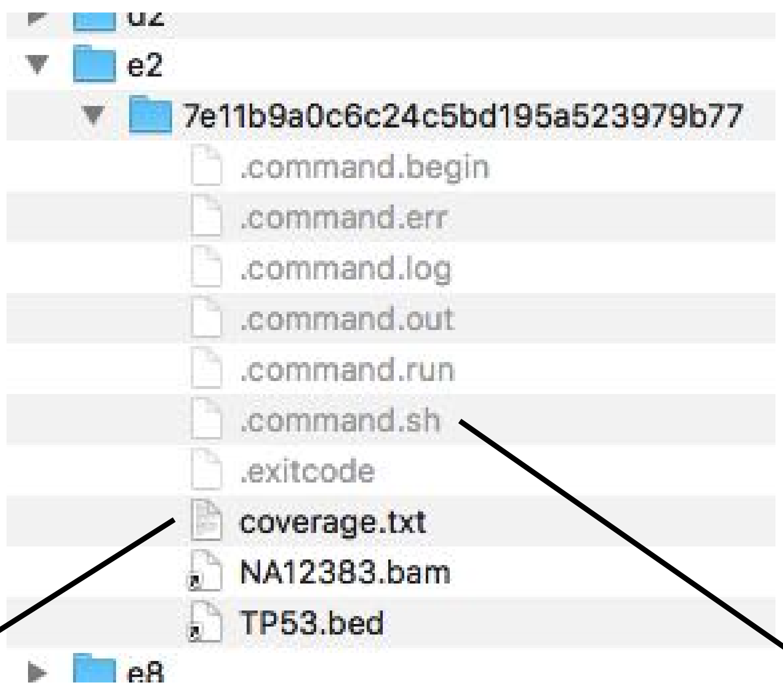
World Health Organization

Running the pipeline

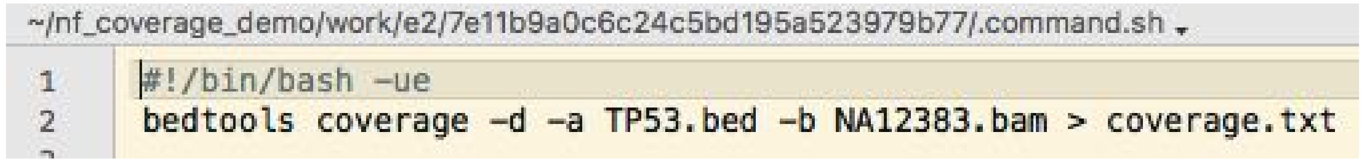


nf_coverage_demo — -bash — 129×33

```
x140083:nf_coverage_demo follm$ nextflow run plot_coverage_1.nf --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Launching `plot_coverage_1.nf` [compassionate_bohr] - revision: dd542716cd
[warm up] executor > local
[e2/7e11b9] Submitted process > coverage (5)
[a4/2206c6] Submitted process > coverage (7)
[12/62bcf4] Submitted process > coverage (8)
[80/f6a03f] Submitted process > coverage (1)
[41/afb57a] Submitted process > coverage (3)
[95/425153] Submitted process > coverage (6)
[8f/44fcb4] Submitted process > coverage (2)
[e8/fe8a8f] Submitted process > coverage (4)
[2b/c8410d] Submitted process > coverage (14)
[30/9157d7] Submitted process > coverage (15)
[46/f2eab0] Submitted process > coverage (9)
[59/8e58d8] Submitted process > coverage (12)
[fd/c188c3] Submitted process > coverage (13)
[a9/444392] Submitted process > coverage (10)
[11/3b3c1f] Submitted process > coverage (11)
[ce/51d91c] Submitted process > coverage (16)
[4c/511af9] Submitted process > coverage (17)
```

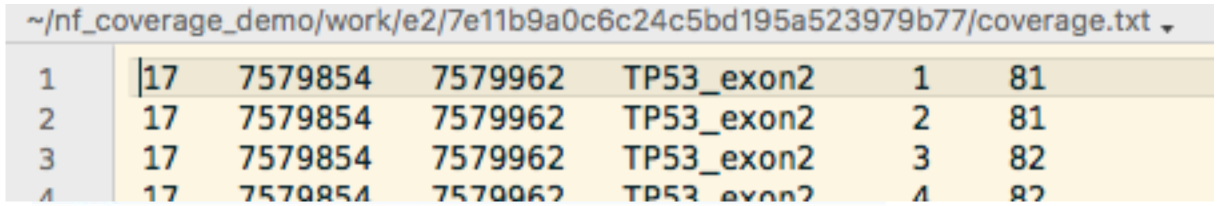


```
▶ d2
▼ e2
  ▼ 7e11b9a0c6c24c5bd195a523979b77
      .command.begin
      .command.err
      .command.log
      .command.out
      .command.run
      .command.sh
      .exitcode
      coverage.txt
      NA12383.bam
      TP53.bed
▶ e8
```



~/nf_coverage_demo/work/e2/7e11b9a0c6c24c5bd195a523979b77/.command.sh

```
#!/bin/bash -ue
bedtools coverage -d -a TP53.bed -b NA12383.bam > coverage.txt
```

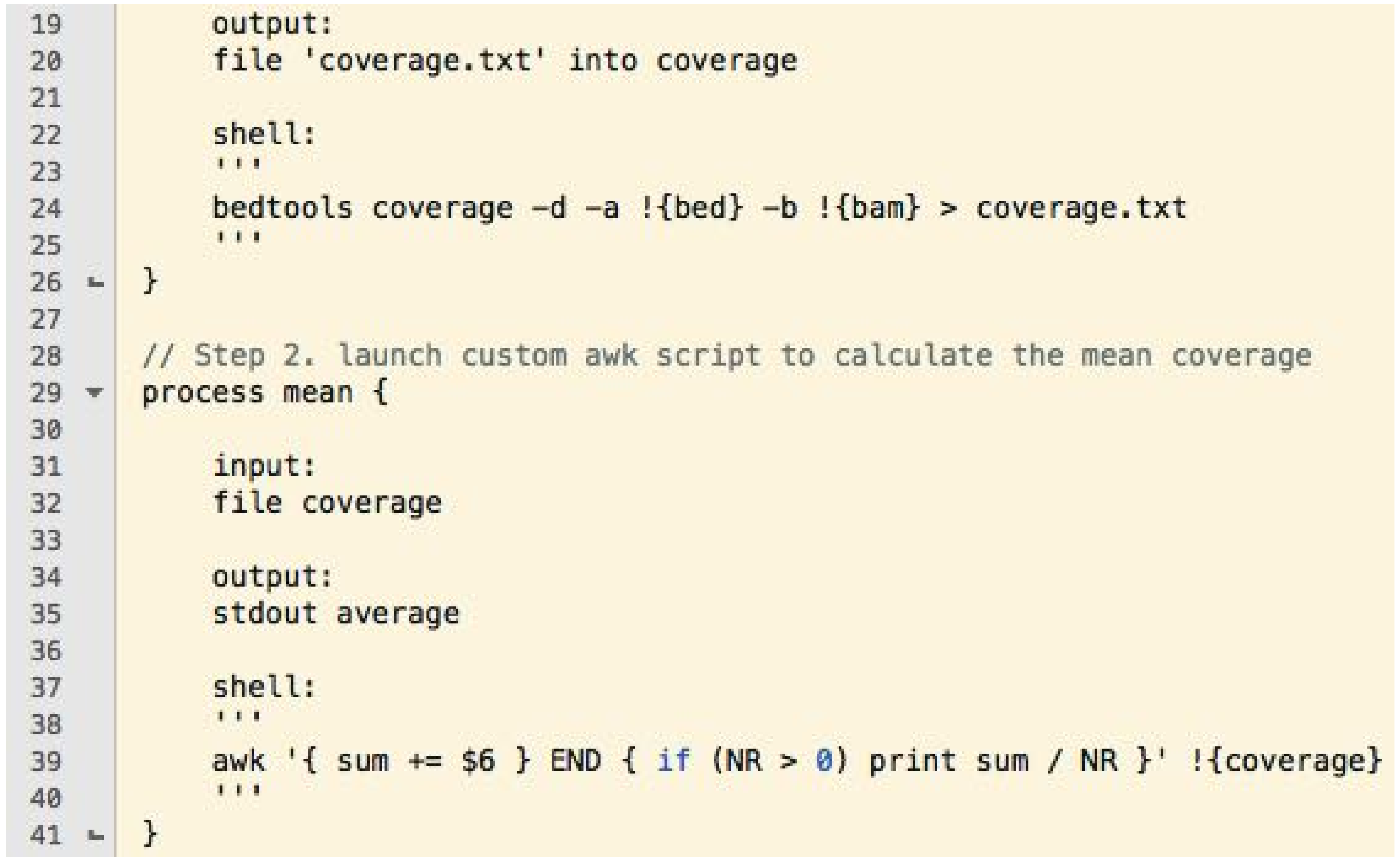


~/nf_coverage_demo/work/e2/7e11b9a0c6c24c5bd195a523979b77/coverage.txt

```
17    7579854    7579962    TP53_exon2    1    81
17    7579854    7579962    TP53_exon2    2    81
17    7579854    7579962    TP53_exon2    3    82
17    7579854    7579962    TP53_exon2    4    82
```

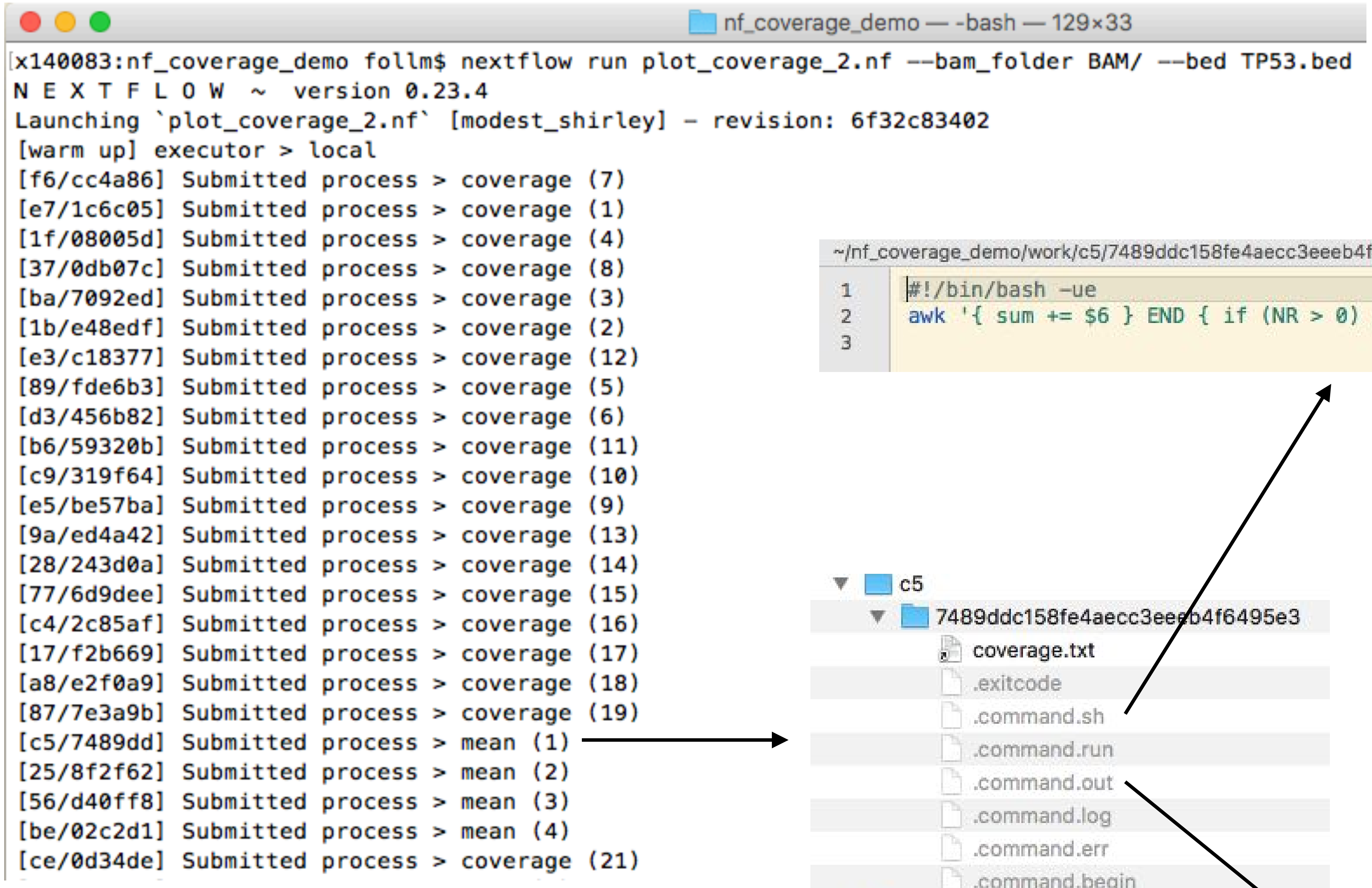
World Health Organization

Adding a second process



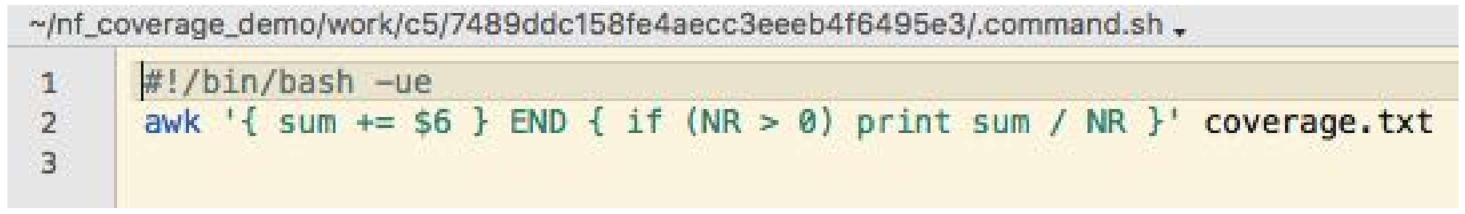
```
    output:
    file 'coverage.txt' into coverage

    shell:
    '''
    bedtools coverage -d -a !{bed} -b !{bam} > coverage.txt
    '''
}

// Step 2. launch custom awk script to calculate the mean coverage
process mean {

    input:
    file coverage

    output:
    stdout average

    shell:
    '''
    awk '{ sum += $6 } END { if (NR > 0) print sum / NR }' !{coverage}
    '''
}
```

Running the pipeline



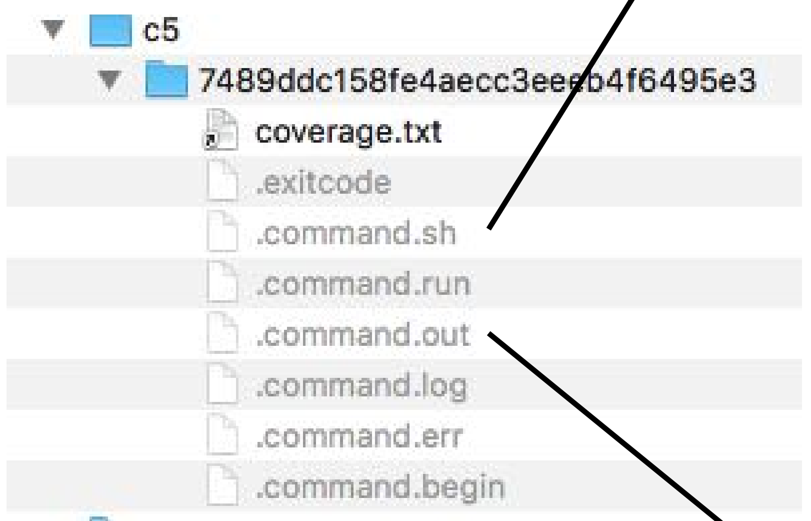
nf_coverage_demo — -bash — 129×33

```
x140083:nf_coverage_demo follm$ nextflow run plot_coverage_2.nf --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Launching `plot_coverage_2.nf` [modest_shirley] - revision: 6f32c83402
[warm up] executor > local
[f6/cc4a86] Submitted process > coverage (7)
[e7/1c6c05] Submitted process > coverage (1)
[1f/08005d] Submitted process > coverage (4)
[37/0db07c] Submitted process > coverage (8)
[ba/7092ed] Submitted process > coverage (3)
[1b/e48edf] Submitted process > coverage (2)
[e3/c18377] Submitted process > coverage (12)
[89/fde6b3] Submitted process > coverage (5)
[d3/456b82] Submitted process > coverage (6)
[b6/59320b] Submitted process > coverage (11)
[c9/319f64] Submitted process > coverage (10)
[e5/be57ba] Submitted process > coverage (9)
[9a/ed4a42] Submitted process > coverage (13)
[28/243d0a] Submitted process > coverage (14)
[77/6d9dee] Submitted process > coverage (15)
[c4/2c85af] Submitted process > coverage (16)
[17/f2b669] Submitted process > coverage (17)
[a8/e2f0a9] Submitted process > coverage (18)
[87/7e3a9b] Submitted process > coverage (19)
[c5/7489dd] Submitted process > mean (1)
[25/8f2f62] Submitted process > mean (2)
[56/d40ff8] Submitted process > mean (3)
[be/02c2d1] Submitted process > mean (4)
[ce/0d34de] Submitted process > coverage (21)
```

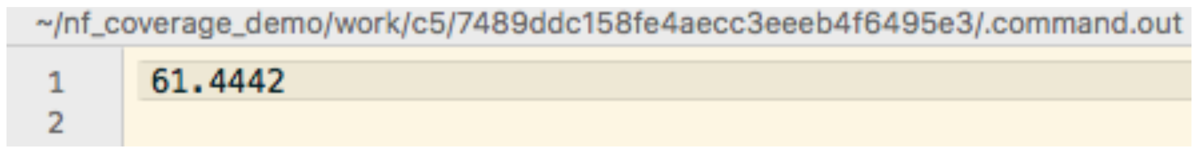


~/nf_coverage_demo/work/c5/7489ddc158fe4aecc3eeeb4f6495e3/.command.sh

```
#!/bin/bash -ue
awk '{ sum += $6 } END { if (NR > 0) print sum / NR }' coverage.txt
```



```
▼ c5
  ▼ 7489ddc158fe4aecc3eeeb4f6495e3
      coverage.txt
      .exitcode
      .command.sh
      .command.run
      .command.out
      .command.log
      .command.err
      .command.begin
```



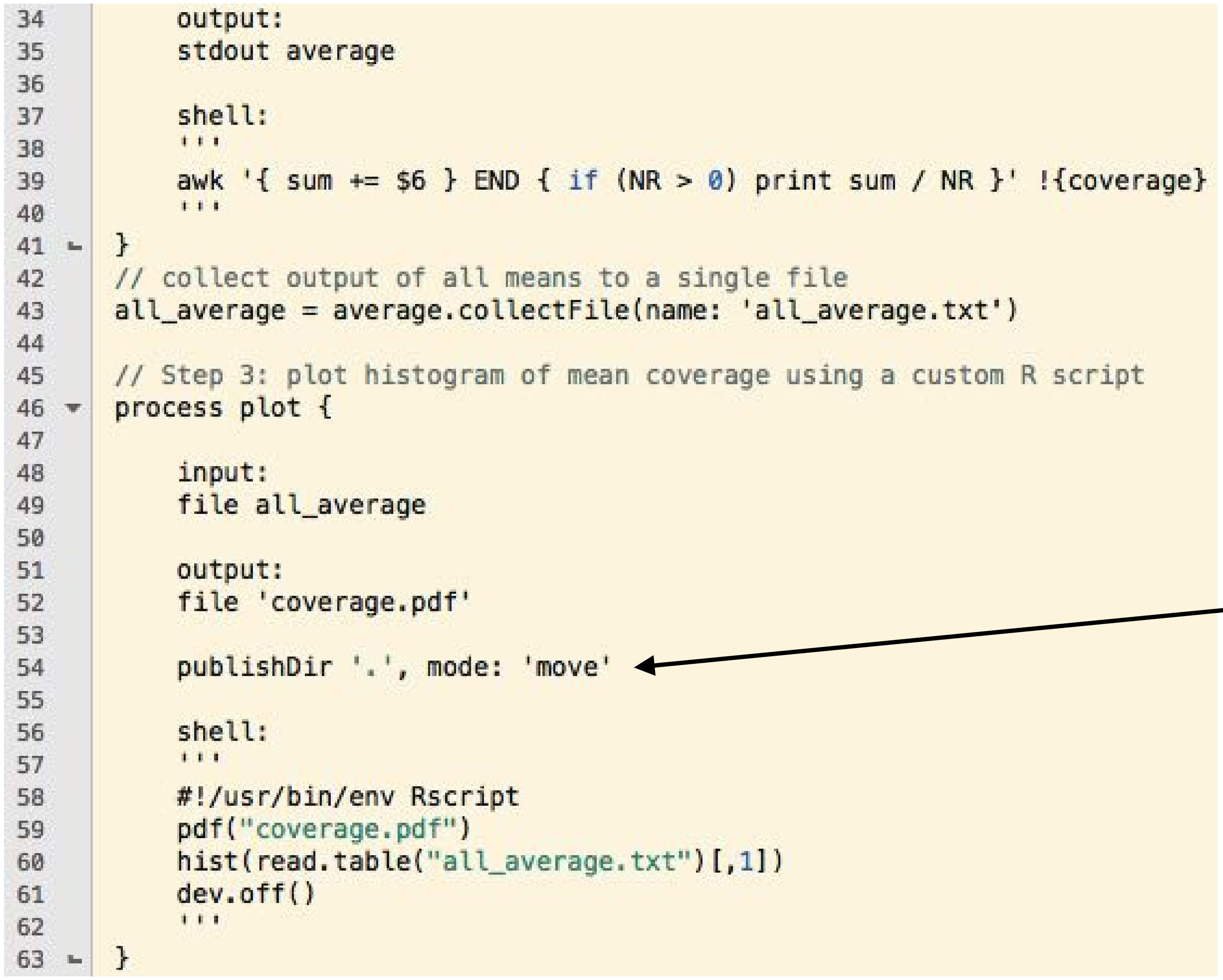
~/nf_coverage_demo/work/c5/7489ddc158fe4aecc3eeeb4f6495e3/.command.out

```
61.4442
```

International Agency for Research on Cancer

World Health Organization

The final process



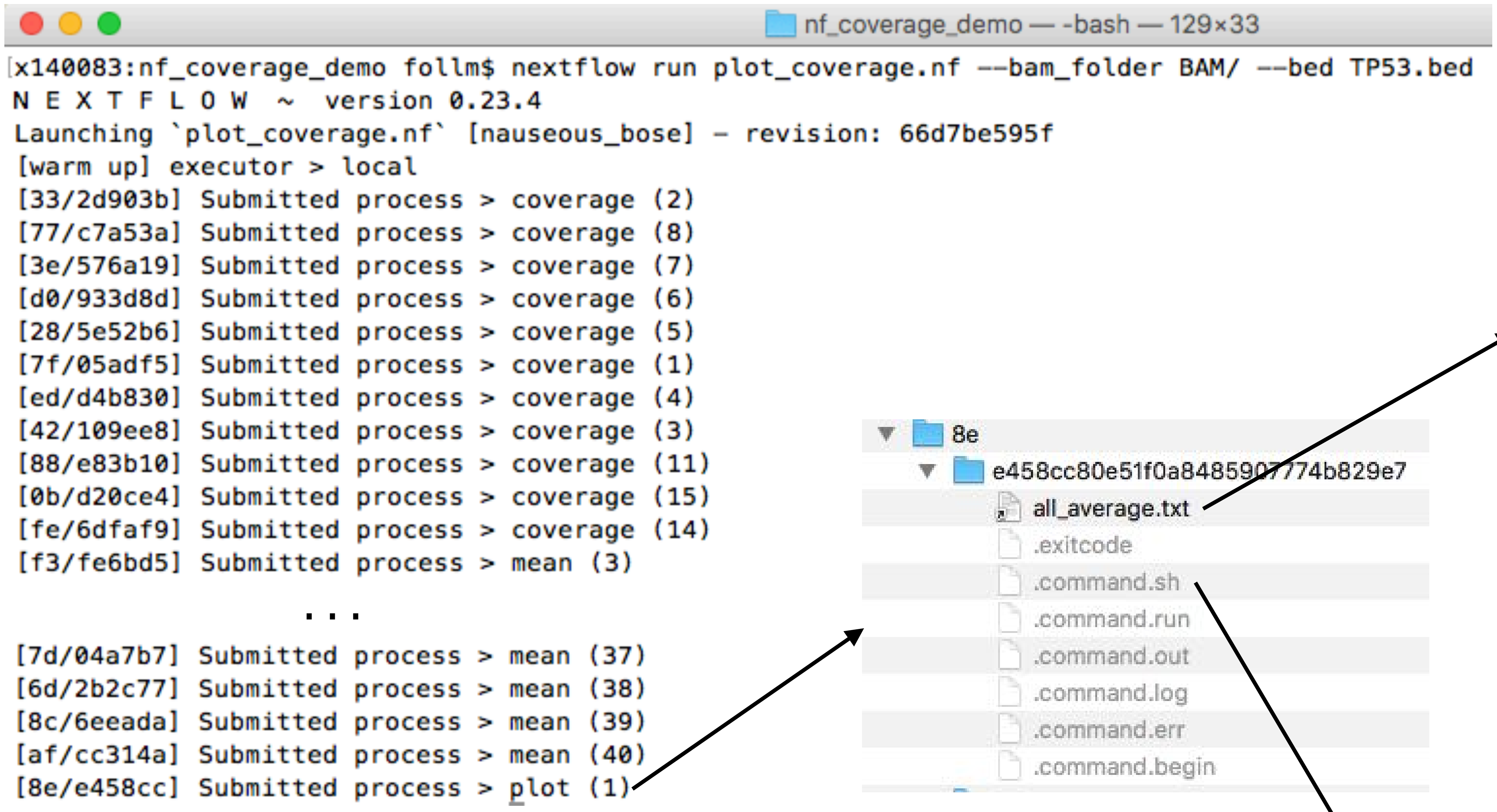
```
    output:
    stdout average

    shell:
    '''
    awk '{ sum += $6 } END { if (NR > 0) print sum / NR }' !{coverage}
    '''
}
// collect output of all means to a single file
all_average = average.collectFile(name: 'all_average.txt')

// Step 3: plot histogram of mean coverage using a custom R script
process plot {

    input:
    file all_average

    output:
    file 'coverage.pdf'

    publishDir '.', mode: 'move'

    shell:
    '''
    #!/usr/bin/env Rscript
    pdf("coverage.pdf")
    hist(read.table("all_average.txt")[,1])
    dev.off()
    '''
}
```

This is our final result

International Agency for Research on Cancer

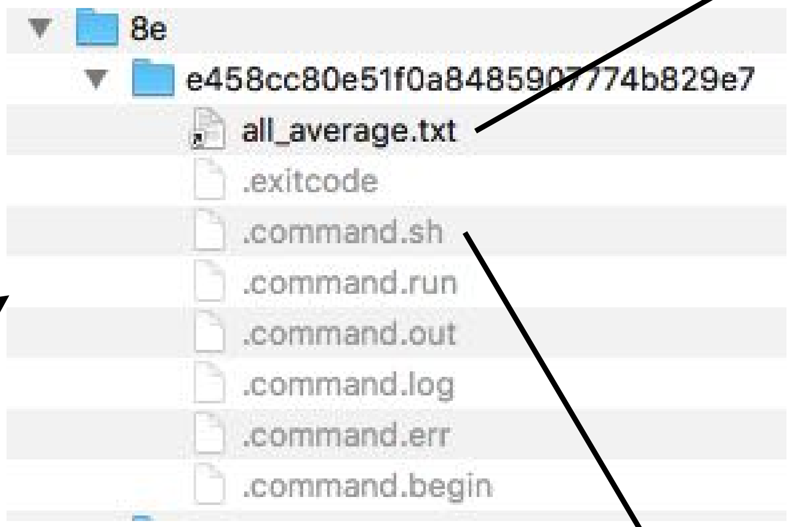
World Health Organization

Running the pipeline

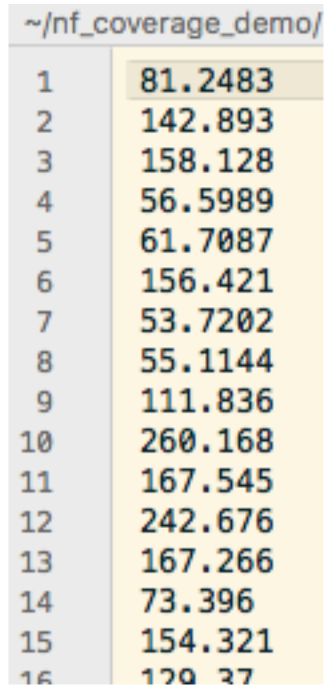


nf_coverage_demo — -bash — 129×33

```
x140083:nf_coverage_demo follm$ nextflow run plot_coverage.nf --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Launching `plot_coverage.nf` [nauseous_bose] - revision: 66d7be595f
[warm up] executor > local
[33/2d903b] Submitted process > coverage (2)
[77/c7a53a] Submitted process > coverage (8)
[3e/576a19] Submitted process > coverage (7)
[d0/933d8d] Submitted process > coverage (6)
[28/5e52b6] Submitted process > coverage (5)
[7f/05adf5] Submitted process > coverage (1)
[ed/d4b830] Submitted process > coverage (4)
[42/109ee8] Submitted process > coverage (3)
[88/e83b10] Submitted process > coverage (11)
[0b/d20ce4] Submitted process > coverage (15)
[fe/6dfaf9] Submitted process > coverage (14)
[f3/fe6bd5] Submitted process > mean (3)

                             ...

[7d/04a7b7] Submitted process > mean (37)
[6d/2b2c77] Submitted process > mean (38)
[8c/6eeada] Submitted process > mean (39)
[af/cc314a] Submitted process > mean (40)
[8e/e458cc] Submitted process > plot (1)
```

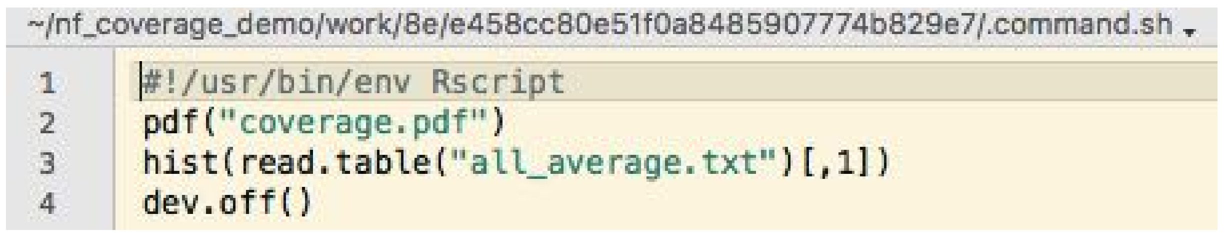


```
▼ 8e
  ▼ e458cc80e51f0a8485907774b829e7
      all_average.txt
      .exitcode
      .command.sh
      .command.run
      .command.out
      .command.log
      .command.err
      .command.begin
```



~/nf_coverage_demo/

```
81.2483
142.893
158.128
56.5989
61.7087
156.421
53.7202
55.1144
111.836
260.168
167.545
242.676
167.266
73.396
154.321
129.37
```



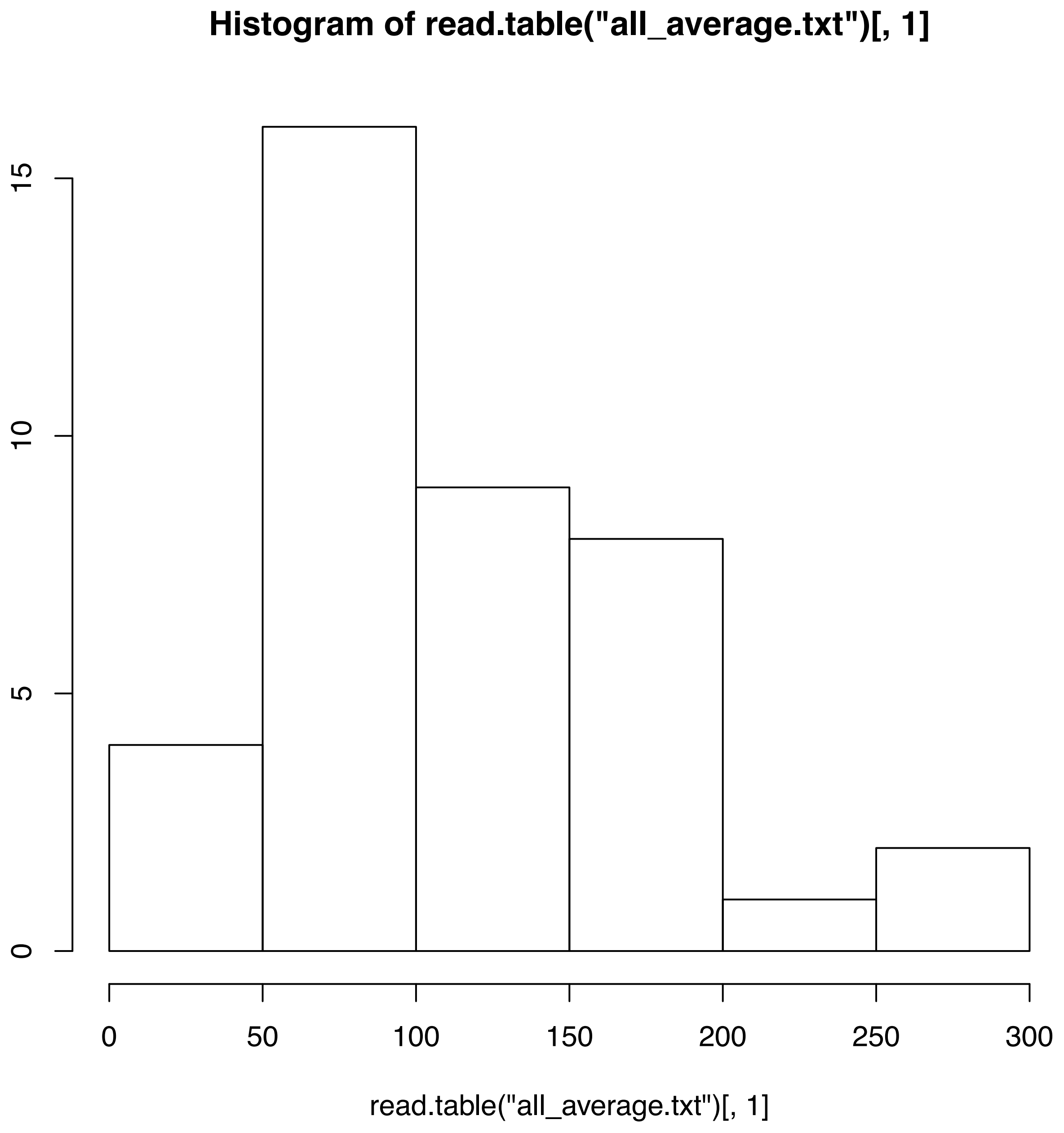
~/nf_coverage_demo/work/8e/e458cc80e51f0a8485907774b829e7/.command.sh

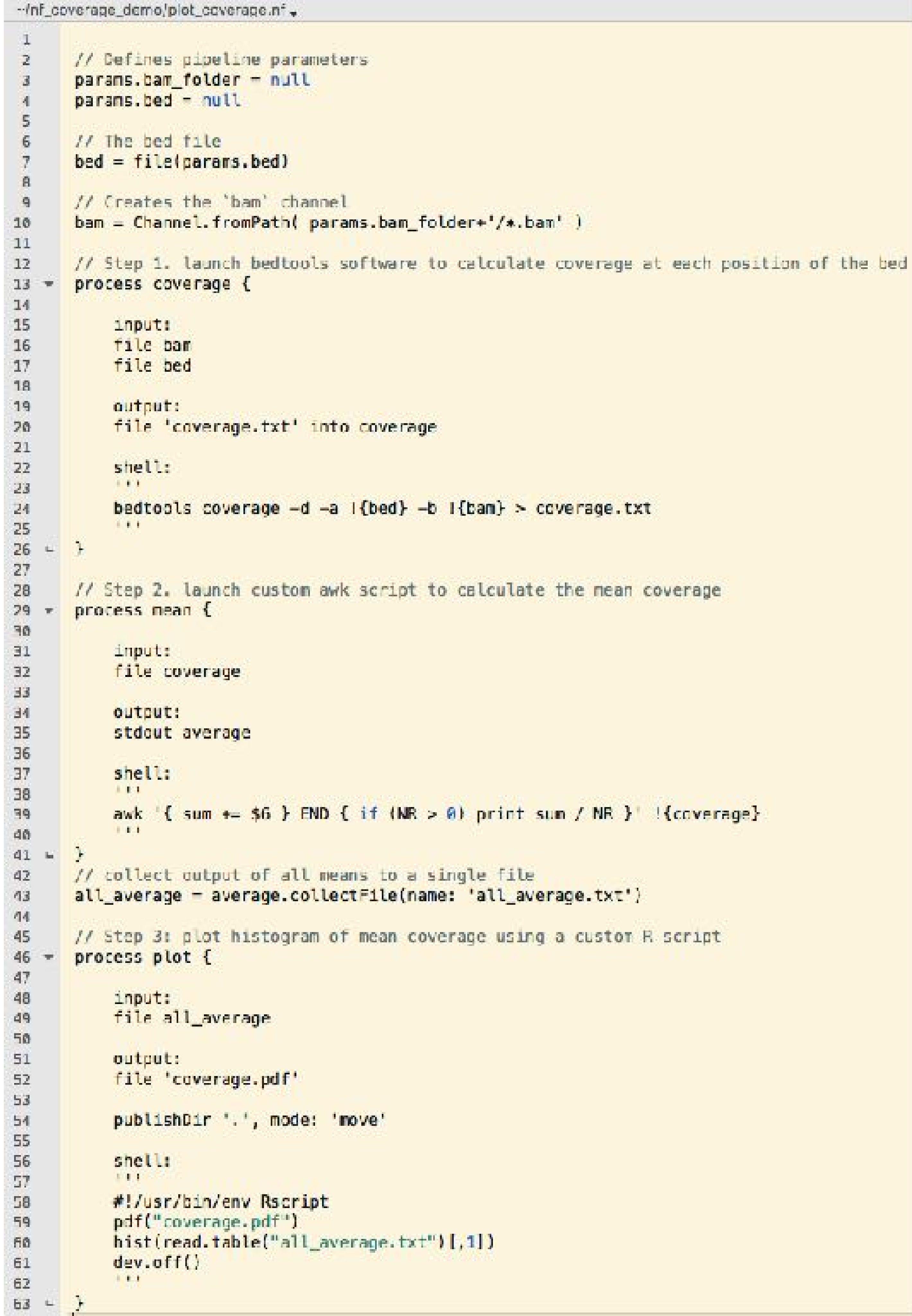
```
#!/usr/bin/env Rscript
pdf("coverage.pdf")
hist(read.table("all_average.txt")[,1])
dev.off()
```

International Agency for Research on Cancer

World Health Organization

Result





```
// Defines pipeline parameters
params.bam_folder = null
params.bed = null

// The bed file
bed = file(params.bed)

// Creates the 'bam' channel
bam = Channel.fromPath( params.bam_folder+'/*.bam' )

// Step 1. launch bedtools software to calculate coverage at each position of the bed
process coverage {

    input:
    file bam
    file bed

    output:
    file 'coverage.txt' into coverage

    shell:
    '''
    bedtools coverage -d -a !{bed} -b !{bam} > coverage.txt
    '''
}

// Step 2. launch custom awk script to calculate the mean coverage
process mean {

    input:
    file coverage

    output:
    stdout average

    shell:
    '''
    awk '{ sum += $6 } END { if (NR > 0) print sum / NR }' !{coverage}
    '''
}
// collect output of all means to a single file
all_average = average.collectFile(name: 'all_average.txt')

// Step 3: plot histogram of mean coverage using a custom R script
process plot {

    input:
    file all_average

    output:
    file 'coverage.pdf'

    publishDir '.', mode: 'move'

    shell:
    '''
    #!/usr/bin/env Rscript
    pdf("coverage.pdf")
    hist(read.table("all_average.txt")[,1])
    dev.off()
    '''
}
```

International Agency for Research on Cancer

World Health Organization

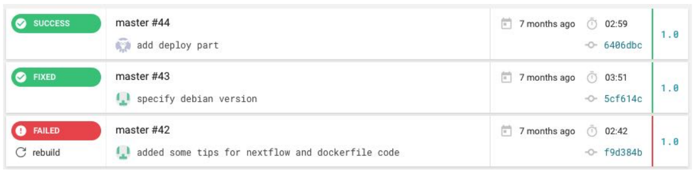
Full life cycle

1 A **developer** modifies the pipeline and pushes the changes to github (*commit*).

This is the only manual action he has to perform as all the following steps are triggered automatically

2 **Github** tracks the *changes* and creates a unique identifier for this *commit*

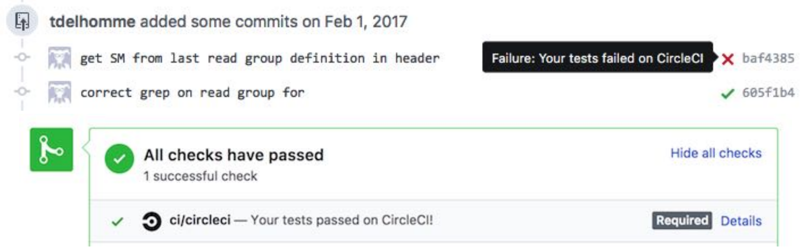
3 **Circleci** starts running *tests* in the cloud to check if the changes haven't broken something and are still producing expected results on some test datasets



✓ SUCCESS	master #44 add deploy part	7 months ago 02:59 6406dbc	1.0
✓ FIXED	master #43 specify debian version	7 months ago 03:51 5cf614c	1.0
! FAILED rebuild	master #42 added some tips for nextflow and dockerfile code	7 months ago 02:42 f9d384b	1.0

4 **Circleci** sends feedback on the pipeline github webpage about the results of the test

circleci passing



tdelhomme added some commits on Feb 1, 2017

get SM from last read group definition in header — Failure: Your tests failed on CircleCI ✗ baf4385

correct grep on read group for ✓ 605f1b4

All checks have passed
1 successful check
Hide all checks

✓ ci/circleci — Your tests passed on CircleCI! Required Details

5 If circleci checks have passed, a new **docker** container is build and hosted on DockerHub

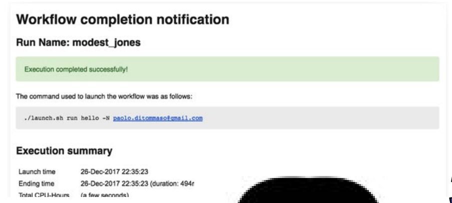
1 A **user** runs the pipeline using *nextflow*. If something goes wrong he opens an *issue*.

2 **Nextflow** downloads the pipeline from github

3 **Nextflow** downloads the docker container with all the *software* from DockerHub

4 **Nextflow** runs each *process* of the pipeline inside the container on the available *executors* (workstation, HPC, cloud)

5 **Nextflow** sends an execution *report*



Workflow completion notification

Run Name: modest_jones

Execution completed successfully!

The command used to launch the workflow was as follows:

```
./launch.sh run hello -N paolo.ditommaso@gmail.com
```

Execution summary

Launch time	26-Dec-2017 22:35:23
Ending time	26-Dec-2017 22:35:23 (duration: 494r
Total CPU-Hours	(a few seconds)

International Agency for Research on Cancer

World Health Organization

Final word

Don’t try to twist nextflow to fit your current habits, give it a chance to actually change your habits