Using the IARC nextflow bioinformatics pipelines Day 1

Matthieu Foll
IARC course: May 23-24 2018

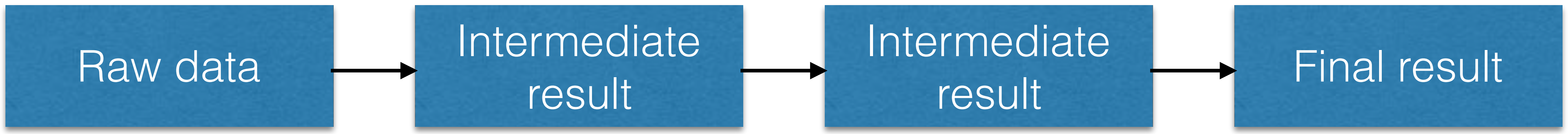
International Agency for Research on Cancer
Lyon, France

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.)

What is a pipeline/workflow?

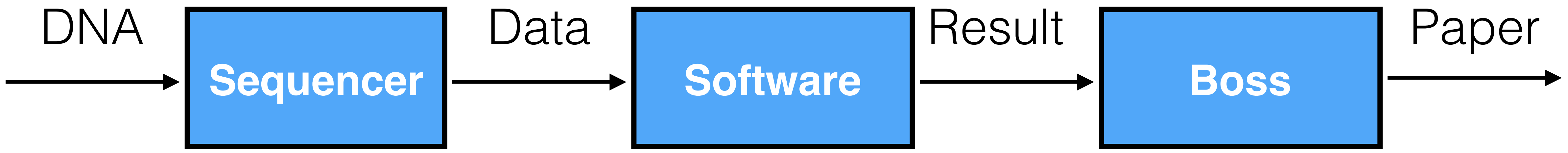


- A number of steps to analyse data
  - Each step can be a software, a script etc.
- Steps are linked by input/output
- One often need to run the same workflow for several samples

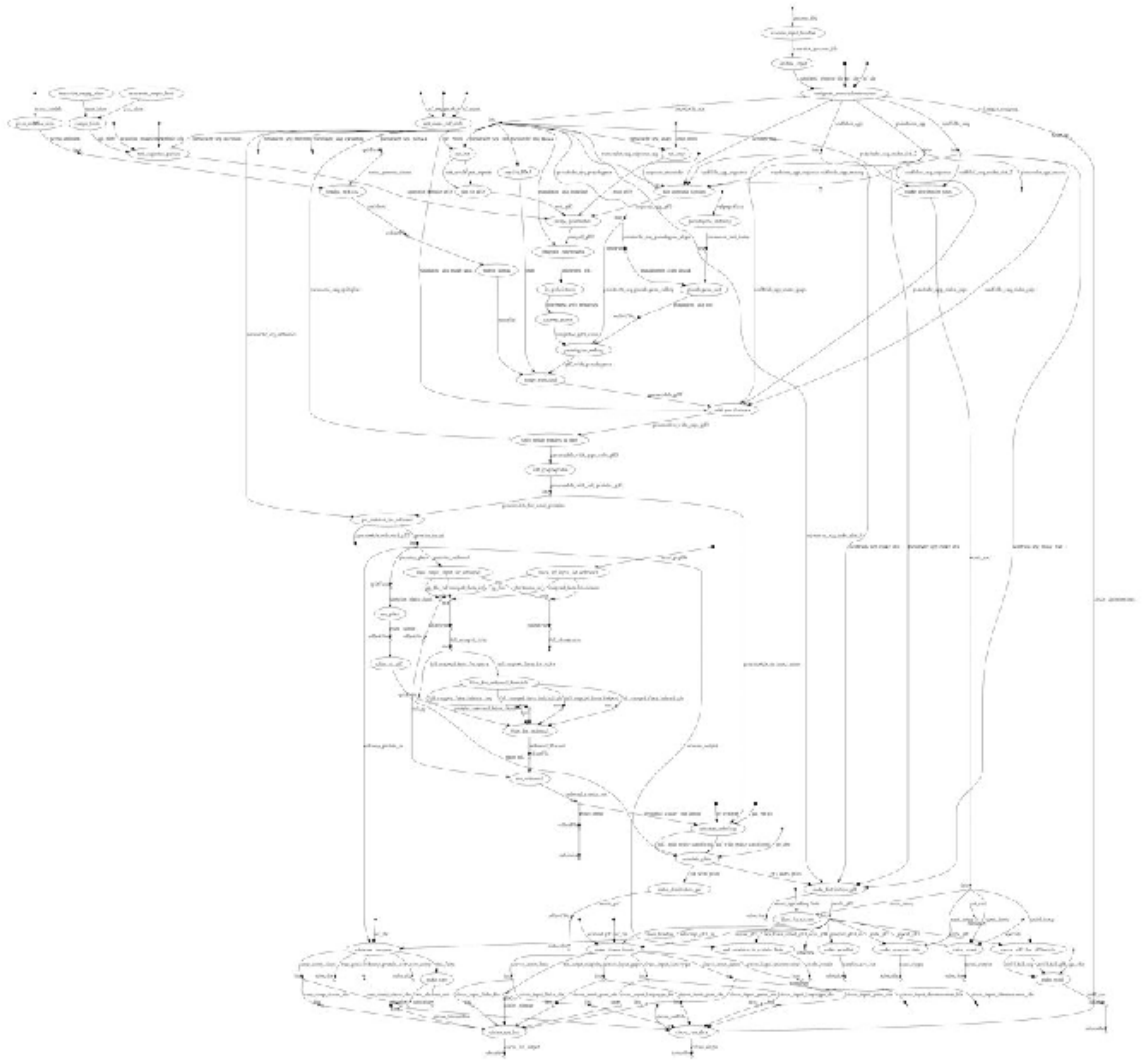
International Agency for Research on Cancer

World Health Organization

Data analysis workflow: point of view of the boss



Data analysis workflow: point of view of the bioinformatician



Why using a workflow manager?

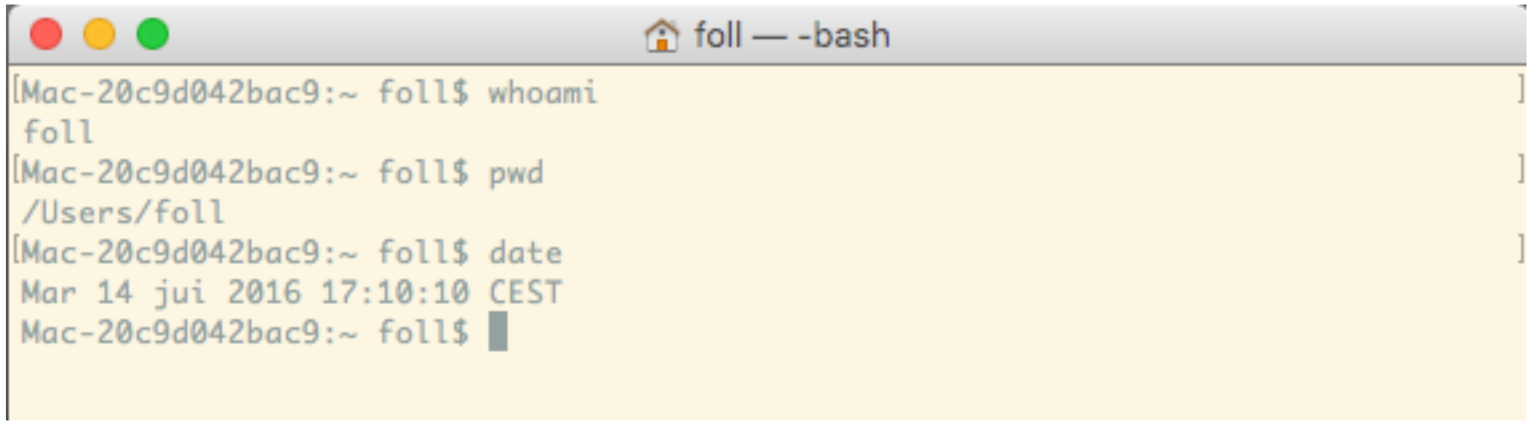
- Automates these steps:
  - launching each step one after the other and in parallel
  - monitoring and reporting
- Make the whole process:
  - Reproducible: easy to share
  - Portable: same pipeline on my laptop, on a HPC cluster or in the cloud (without touching the code)
  - Scalable: deal with parallelism (without touching the code)

“It's not really for the benefit of other people. Experience shows the principal beneficiary of reproducible research is you the author yourself”

-Jon Claerbout

What is the Unix command line?

- A shell that executes the commands we type in
- A terminal: a tool to interact with the shell
- Example: who, where, when?



```
Mac-20c9d042bac9:~ foll$ whoami
foll
Mac-20c9d042bac9:~ foll$ pwd
/Users/foll
Mac-20c9d042bac9:~ foll$ date
Mar 14 jui 2016 17:10:10 CEST
Mac-20c9d042bac9:~ foll$
```

Why Unix?

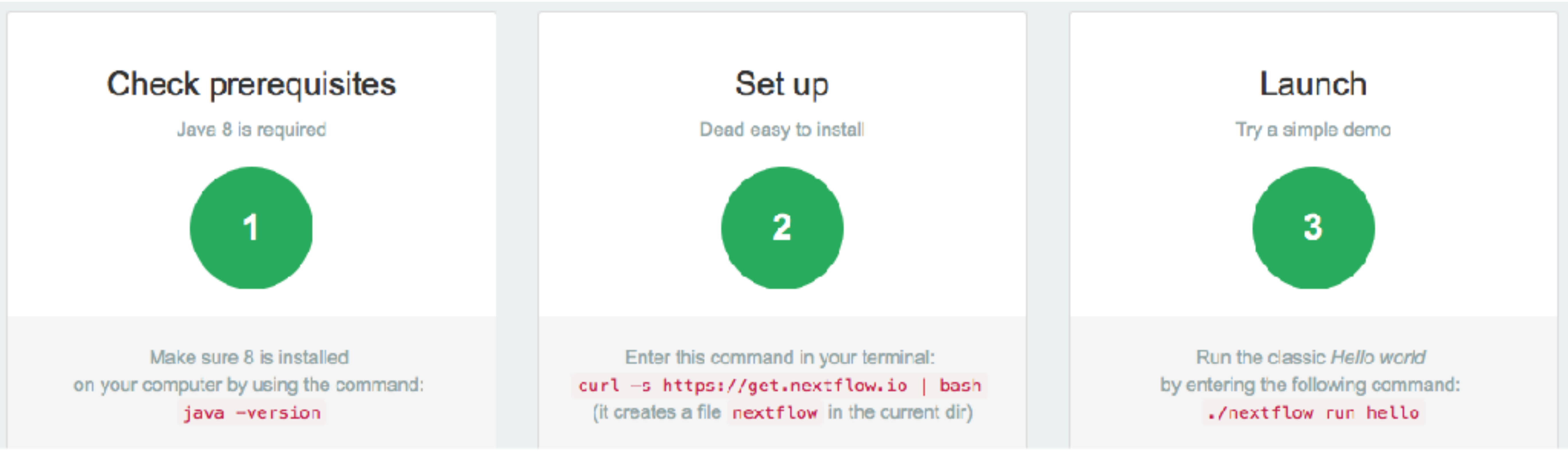
- Automatable and therefore scalable and repeatable
- Extensible
- Integrates will with other technologies (R, Python etc.)
- Ubiquitous (95% of the top 500 supercomputers)
- Very powerful tools (try to open a 1TB file in Excel)

A few things to know

- File extensions (e.g. ".txt") mean nothing in Unix
- File names can contain spaces, or even newline chars, but you should always avoid them
- Different "encoding" exist for text files. UTF-8 is the most common today.
- Windows and Unix text files encode differently new lines.
- You should always use a "real" text editor (e.g. Notepad++ for Windows, TextWrangler/Atom for Mac, Atom for Linux).

Paolo Di Tommaso,
Nextflow Lead

- Command line/text based workflow management system
- Dataflow programming model:
  - **Processes** (software/scripts) receiving (inputs) and emitting (outputs) through **Channels**
- Very lightweight. Installation:



Check prerequisites

Java 8 is required

1

Make sure 8 is installed on your computer by using the command:

```
java -version
```

Set up

Dead easy to install

2

Enter this command in your terminal:

```
curl -s https://get.nextflow.io | bash
```

(it creates a file `nextflow` in the current dir)

Launch

Try a simple demo

3

Run the classic *Hello world* by entering the following command:

```
./nextflow run hello
```

Features

Nextflow is built around the idea that Linux is the *lingua franca* of data science.

Fast prototyping

Nextflow allows you to write a computational pipeline by making it simpler to put together many different tasks.

You may reuse your existing scripts and tools and you don't need to learn a new language or API to start using it.

Reproducibility

Nextflow supports Docker and Singularity containers technology.

This, along with the integration of the GitHub code sharing platform, allows you to write self-contained pipelines, manage versions and to rapidly reproduce any former configuration.

Portable

Nextflow provides an abstraction layer between your pipeline's logic and the execution layer, so that it can be executed on multiple platforms without it changing.

It provides out of the box executors for SGE, LSF, SLURM, PBS and HTCondor batch schedulers and for Kubernetes and Amazon AWS cloud platforms.

Unified parallelism

Nextflow is based on the *dataflow* programming model which greatly simplifies writing complex distributed pipelines.

Parallelisation is implicitly defined by the processes input and output declarations. The resulting applications are inherently parallel and can scale-up or scale-out, transparently, without having to adapt to a specific platform architecture.

Continuous checkpoints

All the intermediate results produced during the pipeline execution are automatically tracked.

This allows you to resume its execution, from the last successfully executed step, no matter what the reason was for it stopping.

Stream oriented

Nextflow extends the Unix pipes model with a fluent DSL, allowing you to handle complex stream interactions easily.

It promotes a programming approach, based on functional composition, that results in resilient and easily reproducible pipelines.

Nextflow in practice

- A software written in Groovy, a language for the java platform (java 8 required)
- An executable bash script "`nextflow`" (putting it in a directory in `$PATH` is a good idea)
- A hidden folder in `~/.nextflow` containing dependencies dowloaded automatically
- Last version (0.29.1) weights 34.1MB
- No compilation needed, no need to be root. Can be installed offline using a self-contained package version (see on GitHub release page).

Nextflow in practice

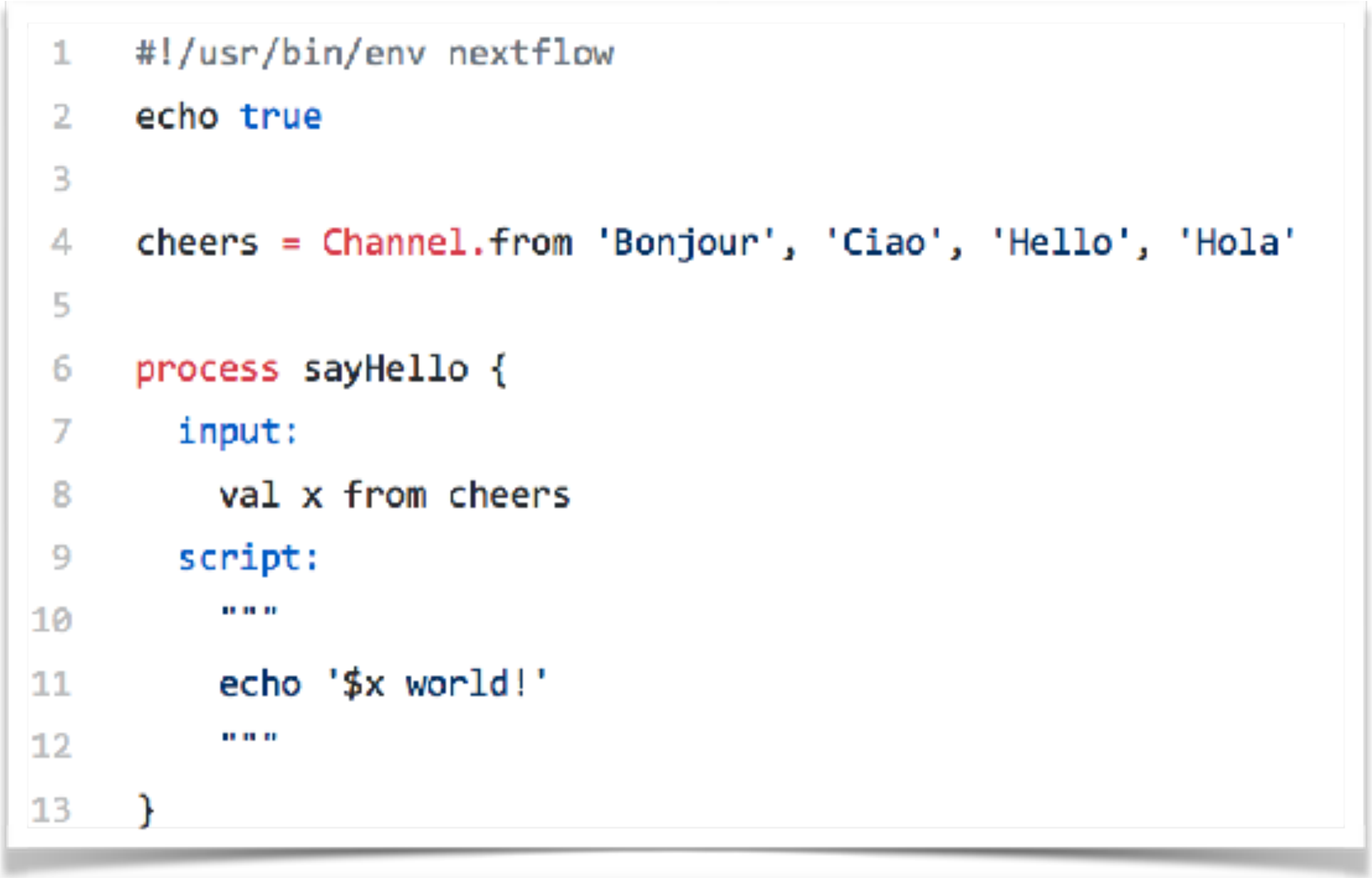
- Install: `wget -qO- https://get.nextflow.io | bash`
- Launch: `nextflow [options] COMMAND [arg…]`
- List options and commands: `nextflow -h`
- List options of command: `nextflow COMMAND -h`

clean	Clean up project cache and work directories
clone	Clone a project into a folder
cloud	Manage Nextflow clusters in the cloud
config	Print a project configuration
drop	Delete the local copy of a project
help	Print the usage help for a command
info	Print project and system runtime information
kuberun	Execute a workflow in a Kubernetes cluster (experimental)
list	List all downloaded projects
log	Print executions log and runtime info
pull	Download or update a project
run	Execute a pipeline project
self-update	Update nextflow runtime to the latest available version
view	View project script file(s)

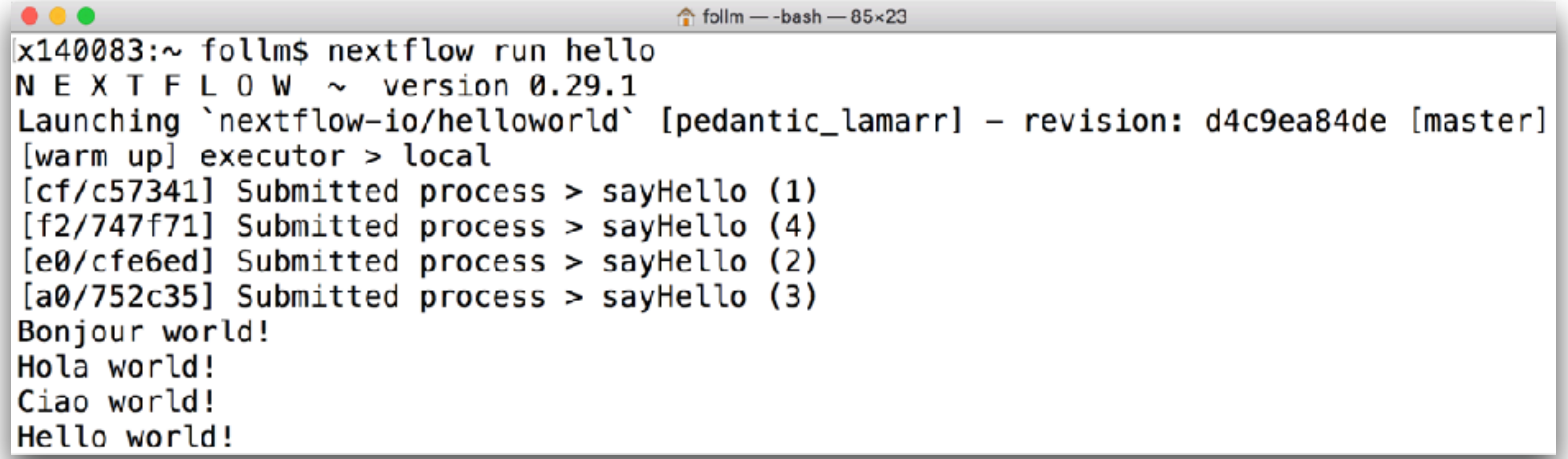
What is a nextflow pipeline?

- A text file (=“script”) describing ***what*** the pipeline will do:
  - *processes* and *channels*
  - parameters
  - usually ends with “`.nf`”
- Optional configuration file(s) defining ***how*** the pipeline will run
  - One global config affecting all pipelines and specific ones
  - CPU, memory, reporting, job scheduler, container etc.

hello world



```
#!/usr/bin/env nextflow
echo true

cheers = Channel.from 'Bonjour', 'Ciao', 'Hello', 'Hola'

process sayHello {
  input:
    val x from cheers
  script:
    """
    echo '$x world!'
    """
}
```



```
x140083:~ follm$ nextflow run hello
N E X T F L O W  ~  version 0.29.1
Launching `nextflow-io/helloworld` [pedantic_lamarr] - revision: d4c9ea84de [master]
[warm up] executor > local
[cf/c57341] Submitted process > sayHello (1)
[f2/747f71] Submitted process > sayHello (4)
[e0/cfe6ed] Submitted process > sayHello (2)
[a0/752c35] Submitted process > sayHello (3)
Bonjour world!
Hola world!
Ciao world!
Hello world!
```

GitHub

- Web hosting service for source code
- Using version control system *git*
- Collaboration features: bug tracking, feature requests, task management etc.
- Users can be part of ***organizations***
- ***repositories*** are the basic units of a GitHub project:
  - they belong to a user or an organisation
  - they can contain folders and files (scripts, images, data sets etc.)
  - a README file is automatically displayed when present
  - they usually also contain a LICENCE file
  - public repositories are free

International Agency for Research on Cancer

World Health Organization

github SOCIAL CODING

Nextflow and GitHub

- One repository hosts one Nextflow pipeline:
  - Nextflow script **`main.nf`**
  - Configuration file **`nextflow.config`**
  - A **`bin/`** folder containing external scripts
  - README
  - LICENCE

Nextflow and GitHub

- If a Nextflow project is hosted at **`http://github.com/user/repo`**, it can be executed with:

```
nextflow run user/repo
```

Containers

- Nextflow scripts can:
  - contain directly some code
  - call external scripts hosted in the `bin/` folder of a repository
  - call any software but it needs to be **installed on your machine**
- Docker and singularity solve the third case by packaging all the software needed in a "**container**"
- Nextflow makes the use of containers transparent

- “Build, Ship, and Run Any App, Anywhere"
- “Docker is a tool that can package an **application and its dependencies** in a virtual container that can run in isolation on any Linux server, whether on premises, public cloud, private cloud etc.”
- DockerHub can host images (free when public)
- Created (in France) by Solomon Hykes, first release March 2013, open source.

International Agency for Research on Cancer

World Health Organization

Example

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

Nextflow is doing this for you!!!

becomes:

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

A Virtual Machine?

- “Docker uses the resource isolation features of the Linux kernel such as cgroups and kernel namespaces, and a union-capable file system such as OverlayFS and others to allow independent "containers" to run within a single Linux instance, avoiding the overhead of starting and maintaining virtual machines”

Singularity

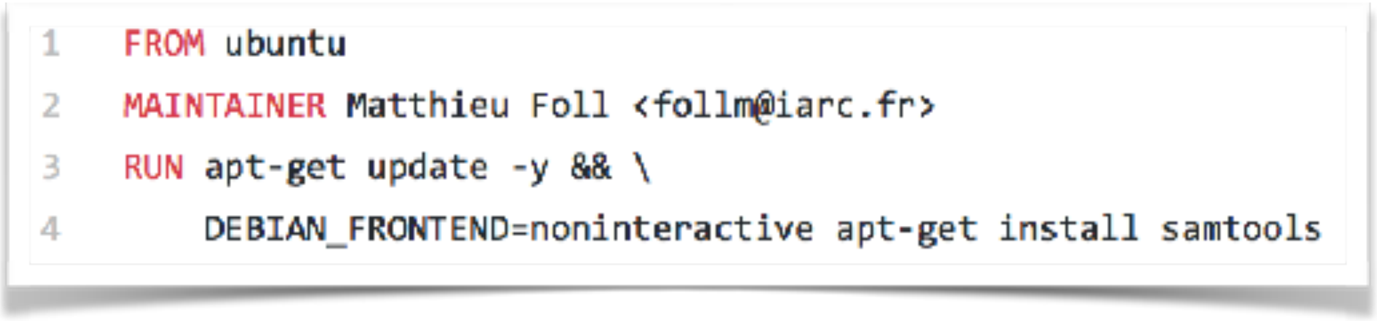
- Users running docker need to have sudo access
- This is not something we can do on shared machines like HPC cluster
- Singularity solves this problem
- And the good news is that it can easily run docker containers!

International Agency for Research on Cancer

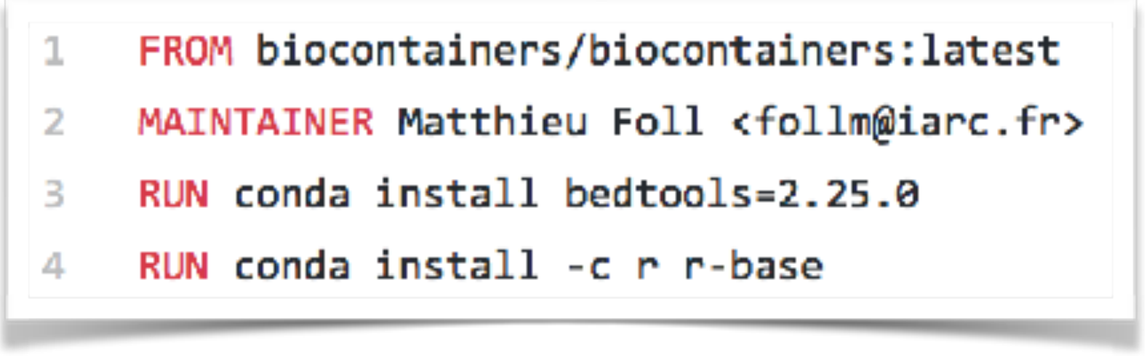
World Health Organization

GitHub and DockerHub

- A docker container can be build from a simple text file called a "`Dockerfile`"
- When hosted on GitHub, a `Dockerfile` can be automatically build by DockerHub and hosted there:



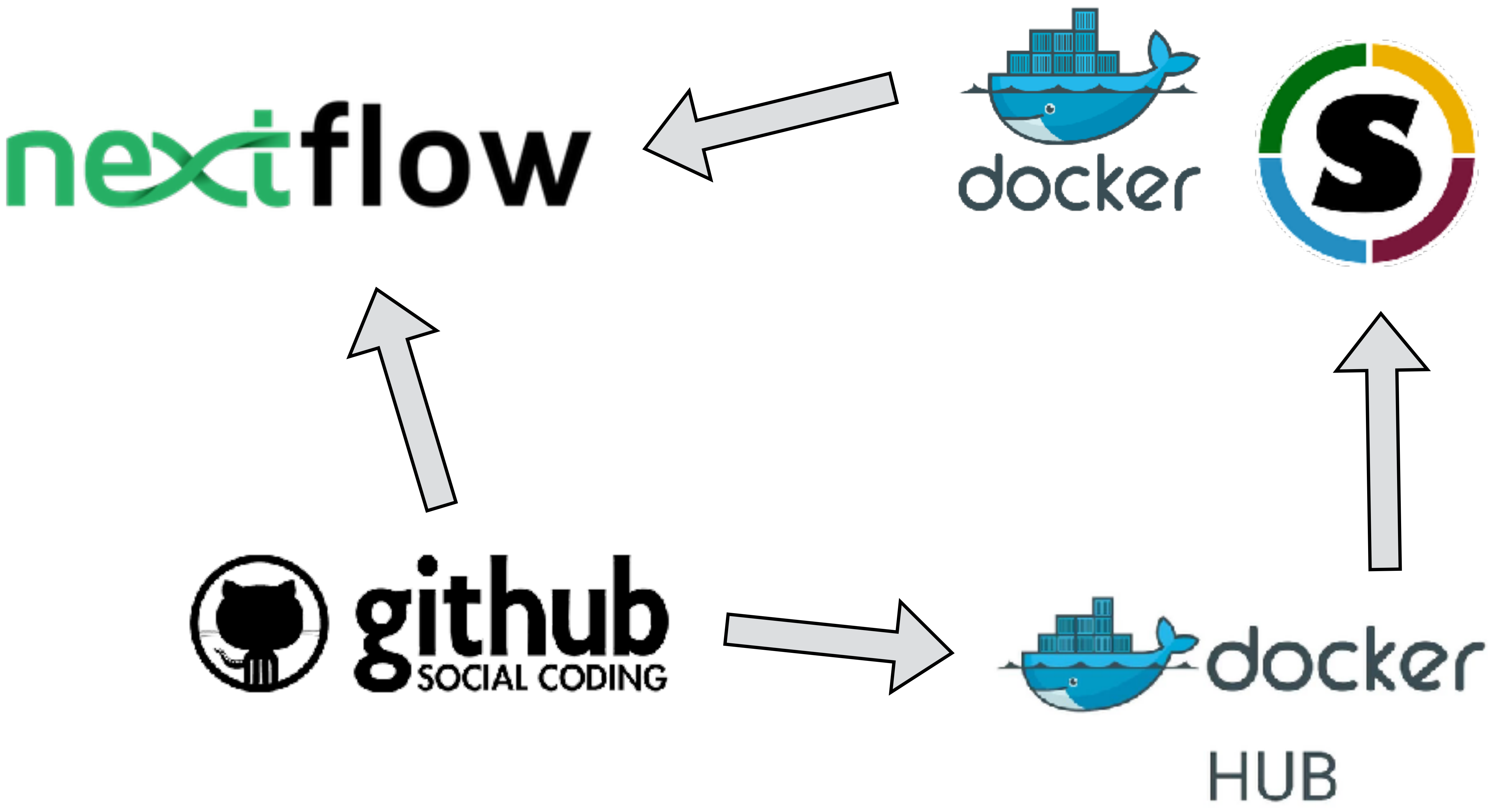
```
FROM ubuntu
MAINTAINER Matthieu Foll <follm@iarc.fr>
RUN apt-get update -y && \
    DEBIAN_FRONTEND=noninteractive apt-get install samtools
```



```
FROM biocontainers/biocontainers:latest
MAINTAINER Matthieu Foll <follm@iarc.fr>
RUN conda install bedtools=2.25.0
RUN conda install -c r r-base
```

International Agency for Research on Cancer
World Health Organization

A self-contained pipeline



Example

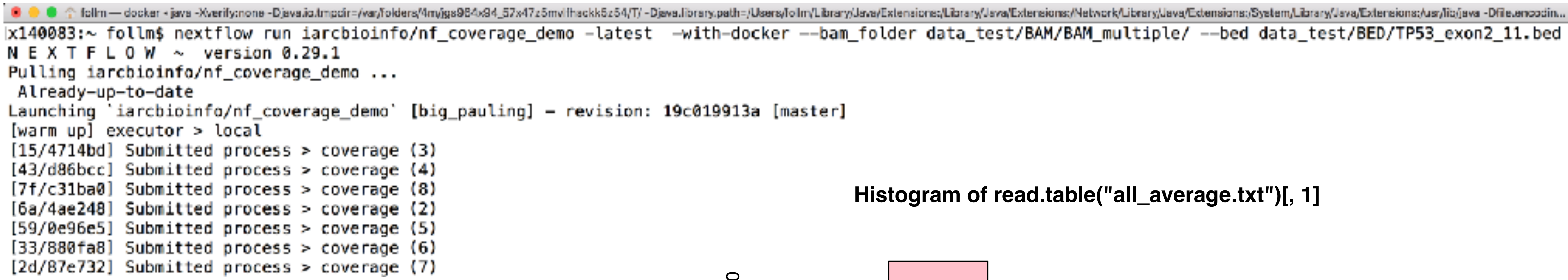
- Pipeline plotting (*output pdf*) the average depth in a genomic region (*input bed file*) over a series of alignment file (*input BAM files*) using a histogram.
- 2 processes for each BAM file (split):
  - use *bedtools* software to calculate the coverage at each position indicated in the bed file
  - use a *awk* command to calculate the average over the region (possible to pipe but kept separated here)
- 1 final process (join):
  - collect all results and use a R script to plot the histogram

International Agency for Research on Cancer

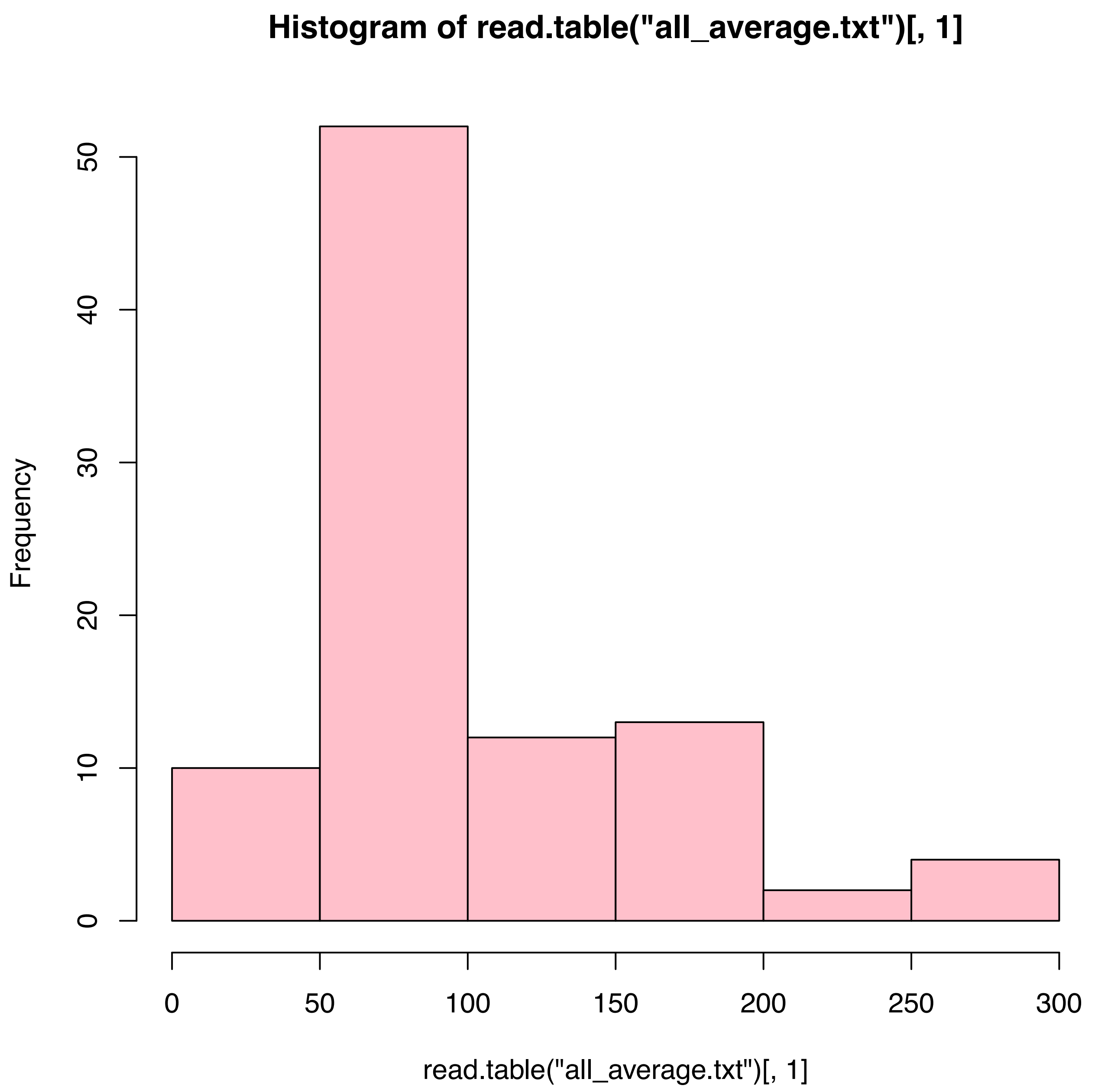
World Health Organization

github SOCIAL CODING: iarcbioinfo/nf_coverage_demo

Example



```
follm — docker ◂ java -Xverify:none -Djava.io.tmpdir=/var/folders/4m/jgs984x94_57x47z5mvlhsckk5z54/T/ -Djava.library.path=/Users/follm/Library/Java/Extensions:/Library/Java/Extensions:/Network/Library/Java/Extensions:/System/Library/Java/Extensions:/usr/lib/java -Dfile.encodin...
x140083:~ follm$ nextflow run iarcbioinfo/nf_coverage_demo -latest  -with-docker --bam_folder data_test/BAM/BAM_multiple/ --bed data_test/BED/TP53_exon2_11.bed
N E X T F L O W  ~  version 0.29.1
Pulling iarcbioinfo/nf_coverage_demo ...
 Already-up-to-date
Launching `iarcbioinfo/nf_coverage_demo` [big_pauling] - revision: 19c019913a [master]
[warm up] executor > local
[15/4714bd] Submitted process > coverage (3)
[43/d86bcc] Submitted process > coverage (4)
[7f/c31ba0] Submitted process > coverage (8)
[6a/4ae248] Submitted process > coverage (2)
[59/0e96e5] Submitted process > coverage (5)
[33/880fa8] Submitted process > coverage (6)
[2d/87e732] Submitted process > coverage (7)
```



Nextflow options: –
Pipeline params: --

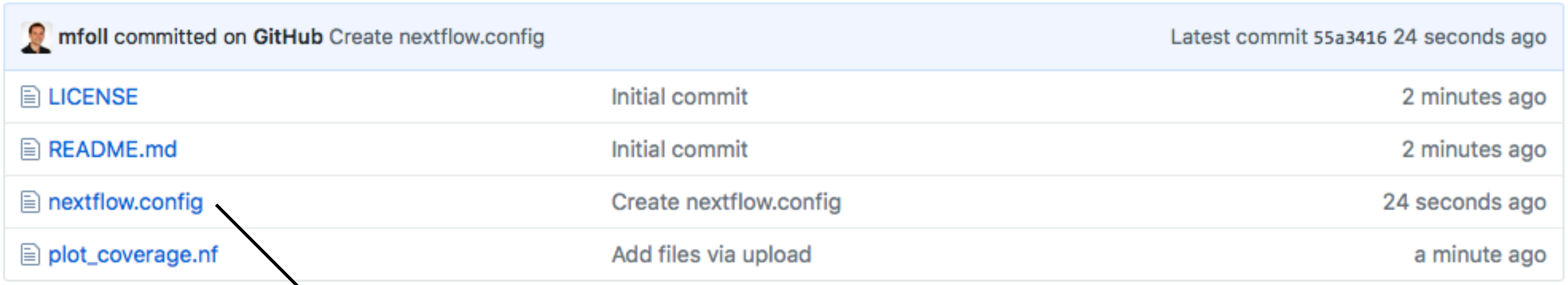
International Agency for Research on Cancer

World Health Organization

Practical

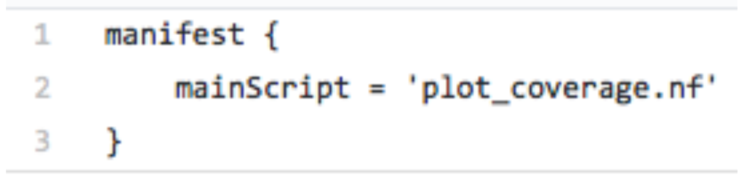
- Explore the GitHub repository to understand its structure
- Run it using test data and docker
- Get familiar with nextflow documentation
- Run other pipelines: RNA-seq-nf and platypus-nf

Running from GitHub

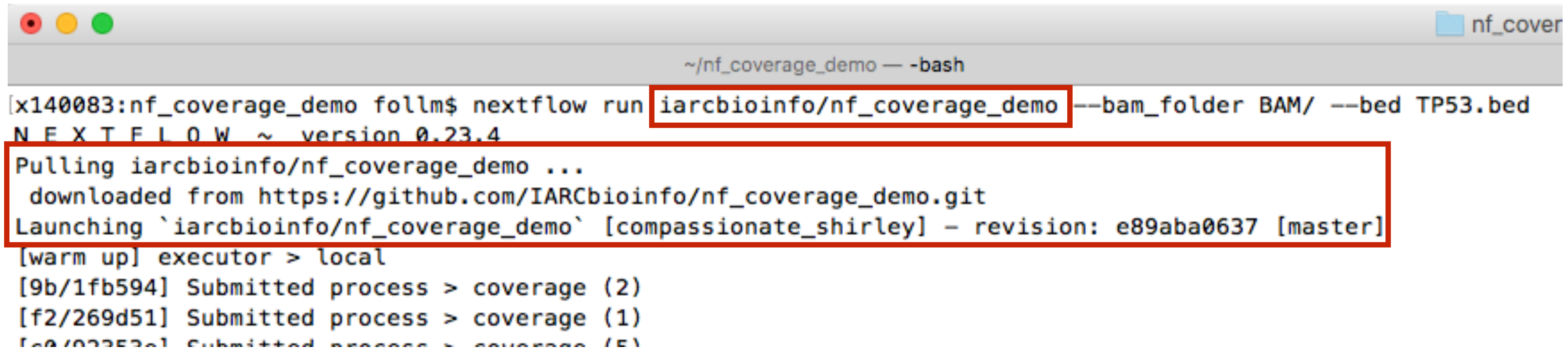


mfoll committed on GitHub Create nextflow.config — Latest commit 55a3416 24 seconds ago

LICENSE	Initial commit	2 minutes ago
README.md	Initial commit	2 minutes ago
nextflow.config	Create nextflow.config	24 seconds ago
plot_coverage.nf	Add files via upload	a minute ago



```
manifest {
    mainScript = 'plot_coverage.nf'
}
```



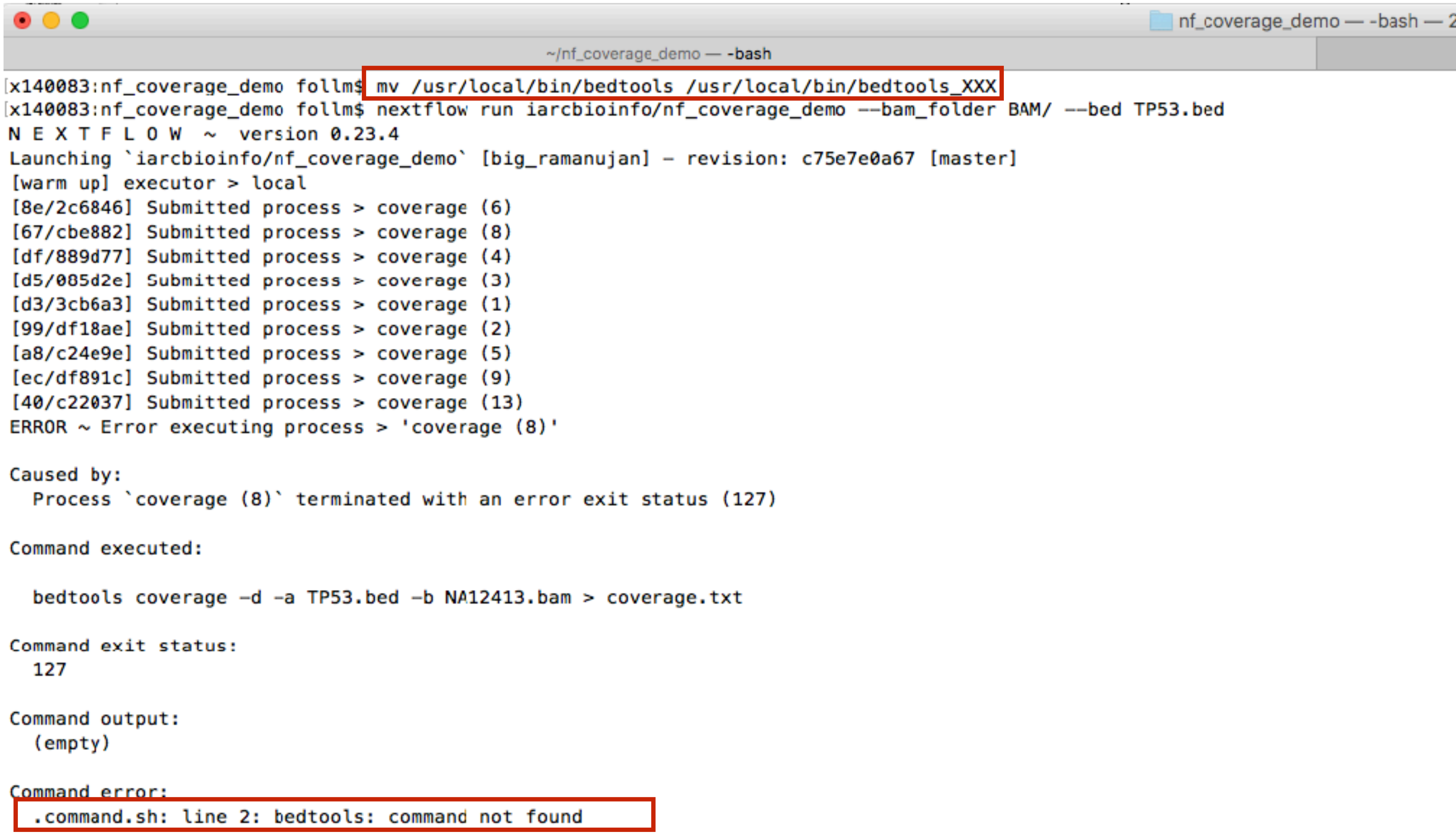
~/nf_coverage_demo — -bash

```
x140083:nf_coverage_demo follm$ nextflow run iarcbioinfo/nf_coverage_demo --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Pulling iarcbioinfo/nf_coverage_demo ...
 downloaded from https://github.com/IARCbioinfo/nf_coverage_demo.git
Launching `iarcbioinfo/nf_coverage_demo` [compassionate_shirley] - revision: e89aba0637 [master]
[warm up] executor > local
[9b/1fb594] Submitted process > coverage (2)
[f2/269d51] Submitted process > coverage (1)
```

International Agency for Research on Cancer

World Health Organization

Running with Docker



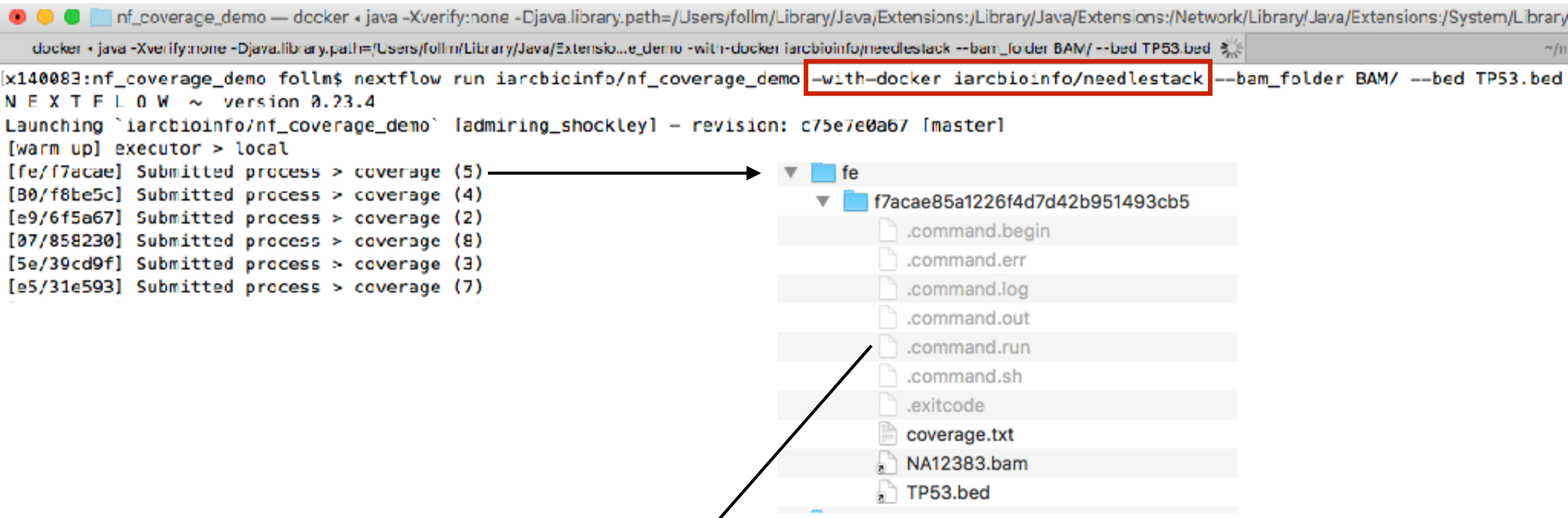
nf_coverage_demo — -bash — 2

~/nf_coverage_demo — -bash

```
[x140083:nf_coverage_demo follm$ mv /usr/local/bin/bedtools /usr/local/bin/bedtools_XXX
[x140083:nf_coverage_demo follm$ nextflow run iarcbioinfo/nf_coverage_demo --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Launching `iarcbioinfo/nf_coverage_demo` [big_ramanujan] - revision: c75e7e0a67 [master]
[warm up] executor > local
[8e/2c6846] Submitted process > coverage (6)
[67/cbe882] Submitted process > coverage (8)
[df/889d77] Submitted process > coverage (4)
[d5/085d2e] Submitted process > coverage (3)
[d3/3cb6a3] Submitted process > coverage (1)
[99/df18ae] Submitted process > coverage (2)
[a8/c24e9e] Submitted process > coverage (5)
[ec/df891c] Submitted process > coverage (9)
[40/c22037] Submitted process > coverage (13)
ERROR ~ Error executing process > 'coverage (8)'

Caused by:
  Process `coverage (8)` terminated with an error exit status (127)

Command executed:

  bedtools coverage -d -a TP53.bed -b NA12413.bam > coverage.txt

Command exit status:
  127

Command output:
  (empty)

Command error:
  .command.sh: line 2: bedtools: command not found
```

Organization

Running with Docker



```
nf_coverage_demo — docker ◂ java -Xverify:none -Djava.library.path=/Users/follm/Library/Java/Extensions:/Library/Java/Extensions:/Network/Library/Java/Extensions:/System/Library
docker ◂ java -Xverify:none -Djava.library.path=/Users/follm/Library/Java/Extensio...e_demo -with-docker iarcbioinfo/needlestack --bam_folder BAM/ --bed TP53.bed
[x140083:nf_coverage_demo follm$ nextflow run iarcbioinfo/nf_coverage_demo -with-docker iarcbioinfo/needlestack --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Launching `iarcbioinfo/nf_coverage_demo` [admiring_shockley] - revision: c75e7e0a67 [master]
[warm up] executor > local
[fe/f7acae] Submitted process > coverage (5)
[B0/f8be5c] Submitted process > coverage (4)
[e9/6f5a67] Submitted process > coverage (2)
[07/858230] Submitted process > coverage (8)
[5e/39cd9f] Submitted process > coverage (3)
[e5/31e593] Submitted process > coverage (7)
```

```
▼ fe
  ▼ f7acae85a1226f4d7d42b951493cb5
      .command.begin
      .command.err
      .command.log
      .command.out
      .command.run
      .command.sh
      .exitcode
      coverage.txt
      NA12383.bam
      TP53.bed
```

```
docker run -i -v /Users/follm/nf_coverage_demo:/Users/follm/nf_coverage_demo -v "$PWD":"$PWD" -w "$PWD"
           --entrypoint /bin/bash --name $NXF_BOXID iarcbioinfo/needlestack
           -c "/bin/bash -ue /Users/follm/nf_coverage_demo/work/fe/f7acae85a1226f4d7d42b951493cb5/.command.sh"
```

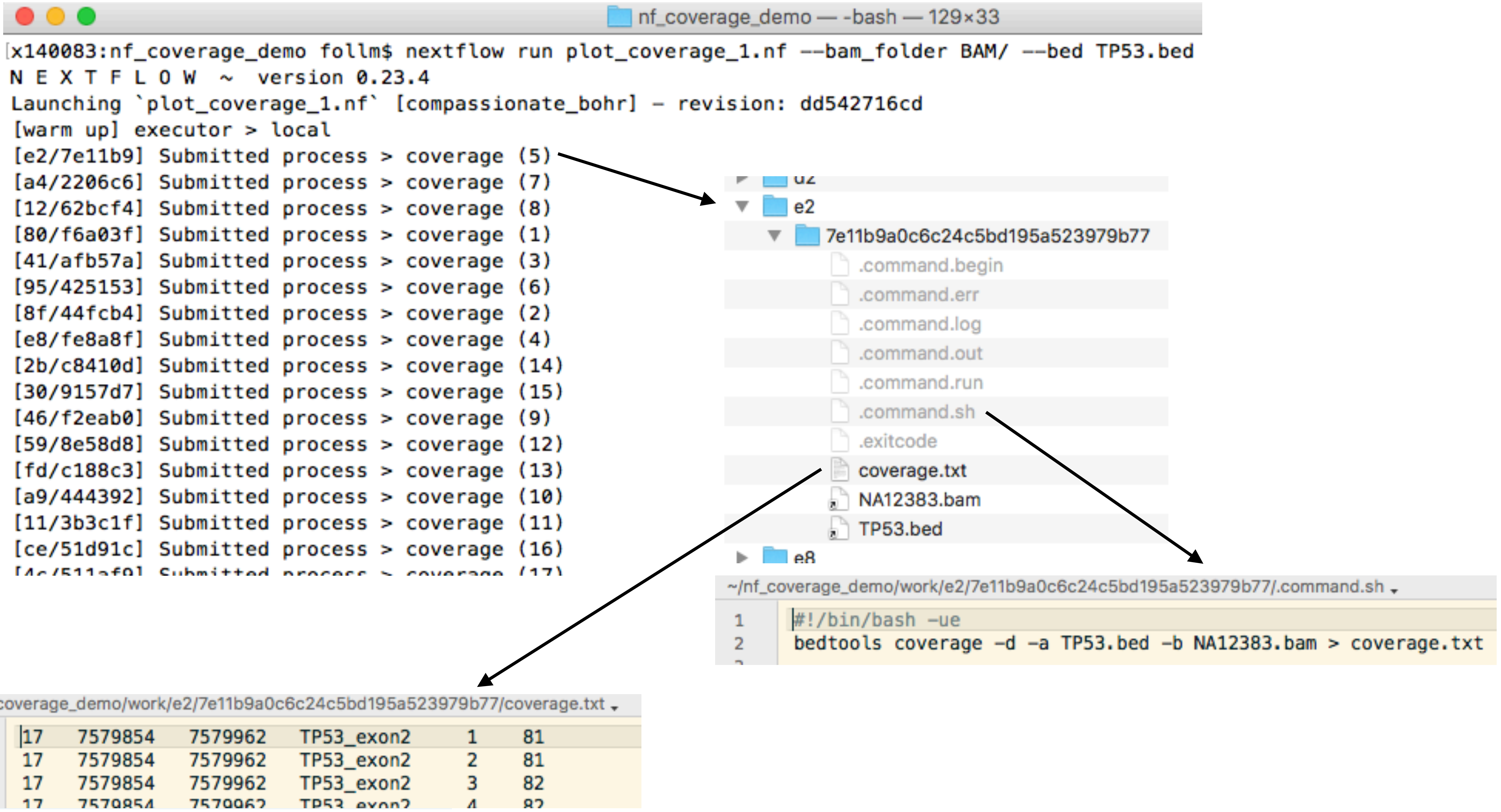
International Agency for Research on Cancer

World Health Organization

How nextflow is *working*?

- Each process runs in isolation in a sub-sub-directory of the `work` directory
- All input files are staged here using symbolic links
- All output files are written here
- stdout and stderr are written in files here
- exitcode
- script
- log

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)
```

```
▼ 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
```

Configuration

- When a pipeline script is launched Nextflow looks for a file named `nextflow.config` in the current directory and in the script base directory (if it is not the same as the current directory). Finally it checks for the file `~/.nextflow/config`.
- An extra configuration file can also be provided by using the command line option `-c <config file>`
- If you want to ignore any default configuration files and use only the custom one use the command line option `-C <config file>`

Example

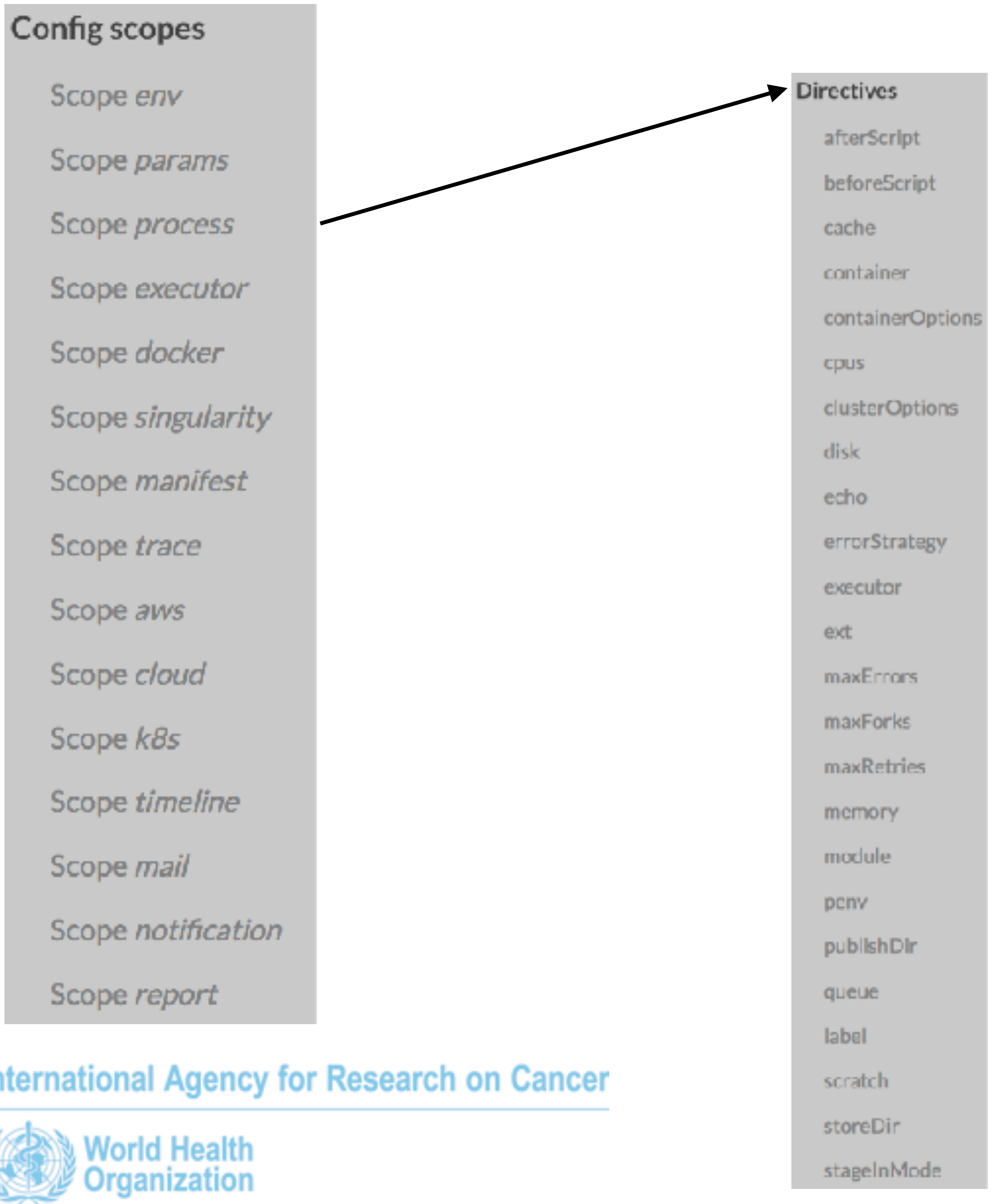
```
process {
        scratch = false
        errorStrategy = 'retry'
        maxErrors = 10000
        memory = { task.exitStatus == 130 ? 4.GB * task.attempt : 4.GB }
        maxRetries = { task.exitStatus == 130 ? 4 : 2 }
}

executor {
        name = 'lsf'
        perJobMemLimit = true
        queueSize = 50
}

trace {
    enabled = true
}

timeline {
    enabled = true
}

report {
        enabled = true
}

mail {
        from = 'username@iarc.fr'
        smtp.host = 'email.iarc.fr'
        smtp.port = 587
        smtp.user = 'username'
        smtp.password = 'XXXXXXXXXXX'
}

notification {
        enabled = true
        to = 'username@iarc.fr'
}

singularity {
        cacheDir = '/data/username/.singularity/'
}
```

International Agency for Research on Cancer

World Health Organization

Note: some options can directly be set in nextflow command line options like "`-with-trace`"

Config scopes



Config scopes

- Scope *env*
- Scope *params*
- Scope *process*
- Scope *executor*
- Scope *docker*
- Scope *singularity*
- Scope *manifest*
- Scope *trace*
- Scope *aws*
- Scope *cloud*
- Scope *k8s*
- Scope *timeline*
- Scope *mail*
- Scope *notification*
- Scope *report*

Directives

- afterScript
- beforeScript
- cache
- container
- containerOptions
- cpus
- clusterOptions
- disk
- echo
- errorStrategy
- executor
- ext
- maxErrors
- maxForks
- maxRetries
- memory
- module
- penv
- publishDir
- queue
- label
- scratch
- storeDir
- stageInMode

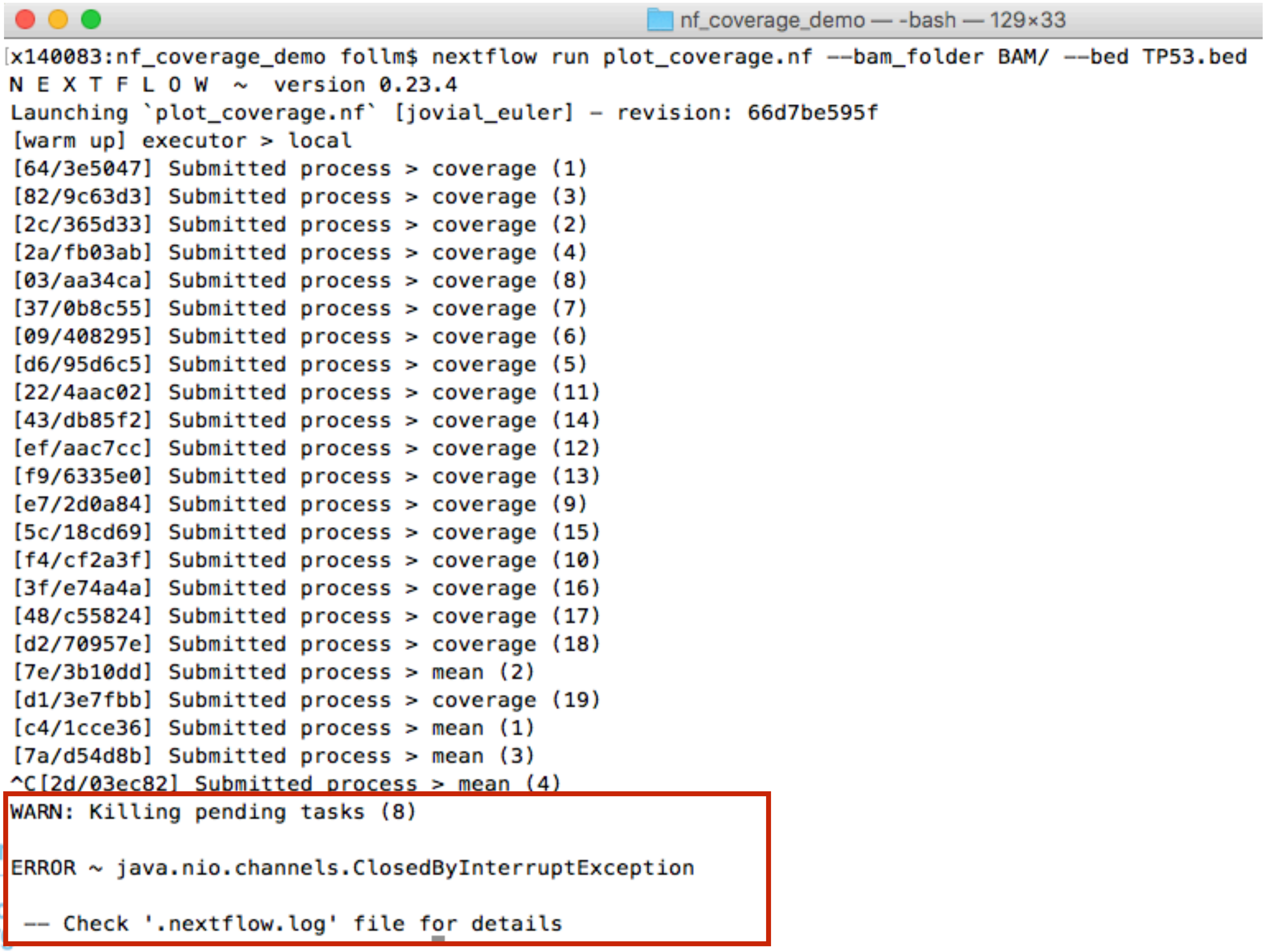
International Agency for Research on Cancer

World Health Organization

Practical

- Explore the `work` directory for one example of each of the three processes
- Add a config file to get a *timeline* and email *notification*
- Add a config file with the parameters of the pipeline
- Kill the pipeline while running (Ctrl+c) and resume it
- Look at the `clean` command and clean your work folder using the `history` command

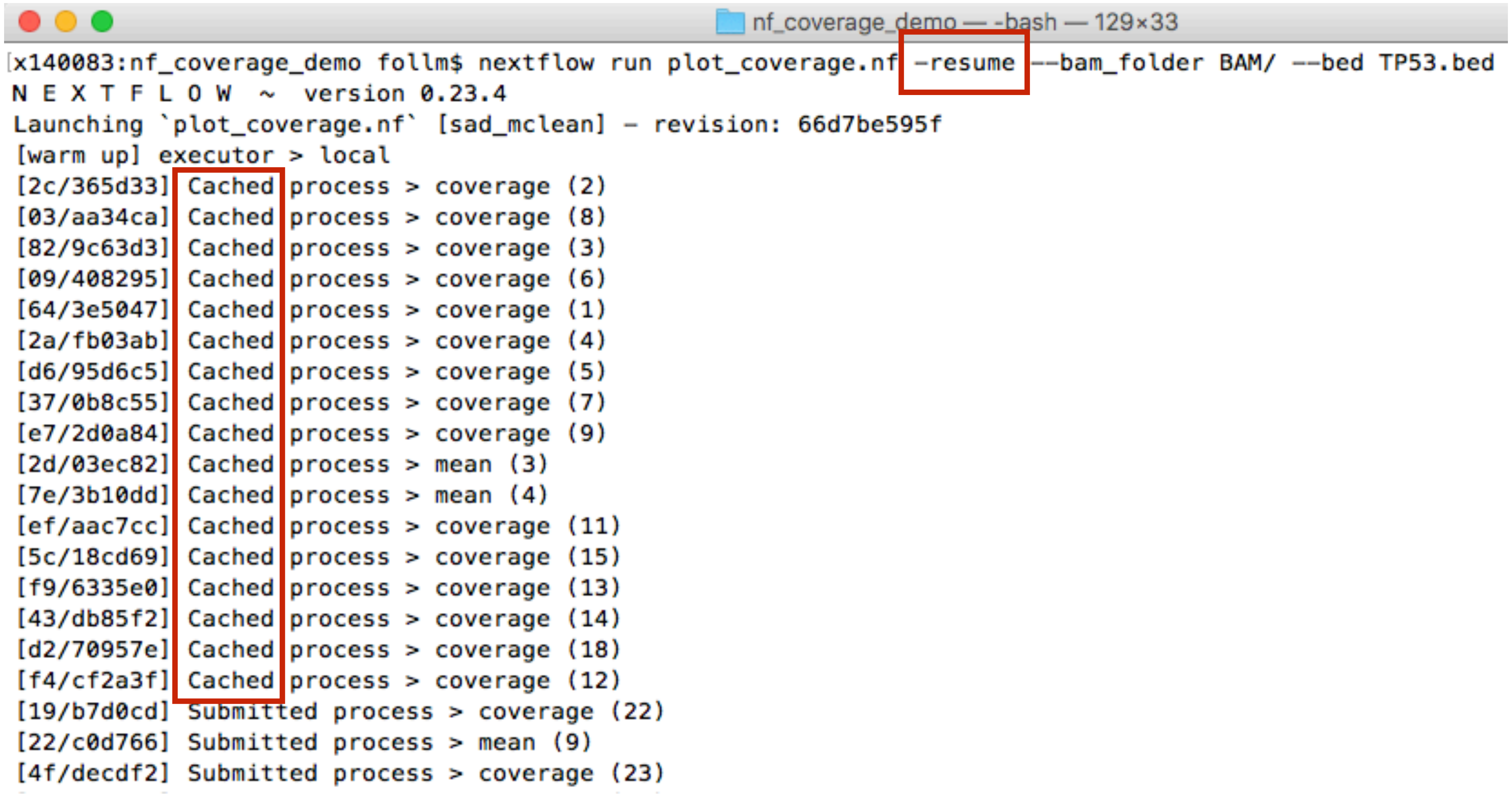
Pipeline crashed!



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` [jovial_euler] - revision: 66d7be595f
[warm up] executor > local
[64/3e5047] Submitted process > coverage (1)
[82/9c63d3] Submitted process > coverage (3)
[2c/365d33] Submitted process > coverage (2)
[2a/fb03ab] Submitted process > coverage (4)
[03/aa34ca] Submitted process > coverage (8)
[37/0b8c55] Submitted process > coverage (7)
[09/408295] Submitted process > coverage (6)
[d6/95d6c5] Submitted process > coverage (5)
[22/4aac02] Submitted process > coverage (11)
[43/db85f2] Submitted process > coverage (14)
[ef/aac7cc] Submitted process > coverage (12)
[f9/6335e0] Submitted process > coverage (13)
[e7/2d0a84] Submitted process > coverage (9)
[5c/18cd69] Submitted process > coverage (15)
[f4/cf2a3f] Submitted process > coverage (10)
[3f/e74a4a] Submitted process > coverage (16)
[48/c55824] Submitted process > coverage (17)
[d2/70957e] Submitted process > coverage (18)
[7e/3b10dd] Submitted process > mean (2)
[d1/3e7fbb] Submitted process > coverage (19)
[c4/1cce36] Submitted process > mean (1)
[7a/d54d8b] Submitted process > mean (3)
^C[2d/03ec82] Submitted process > mean (4)
WARN: Killing pending tasks (8)

ERROR ~ java.nio.channels.ClosedByInterruptException

 -- Check '.nextflow.log' file for details
```

But can be resumed…

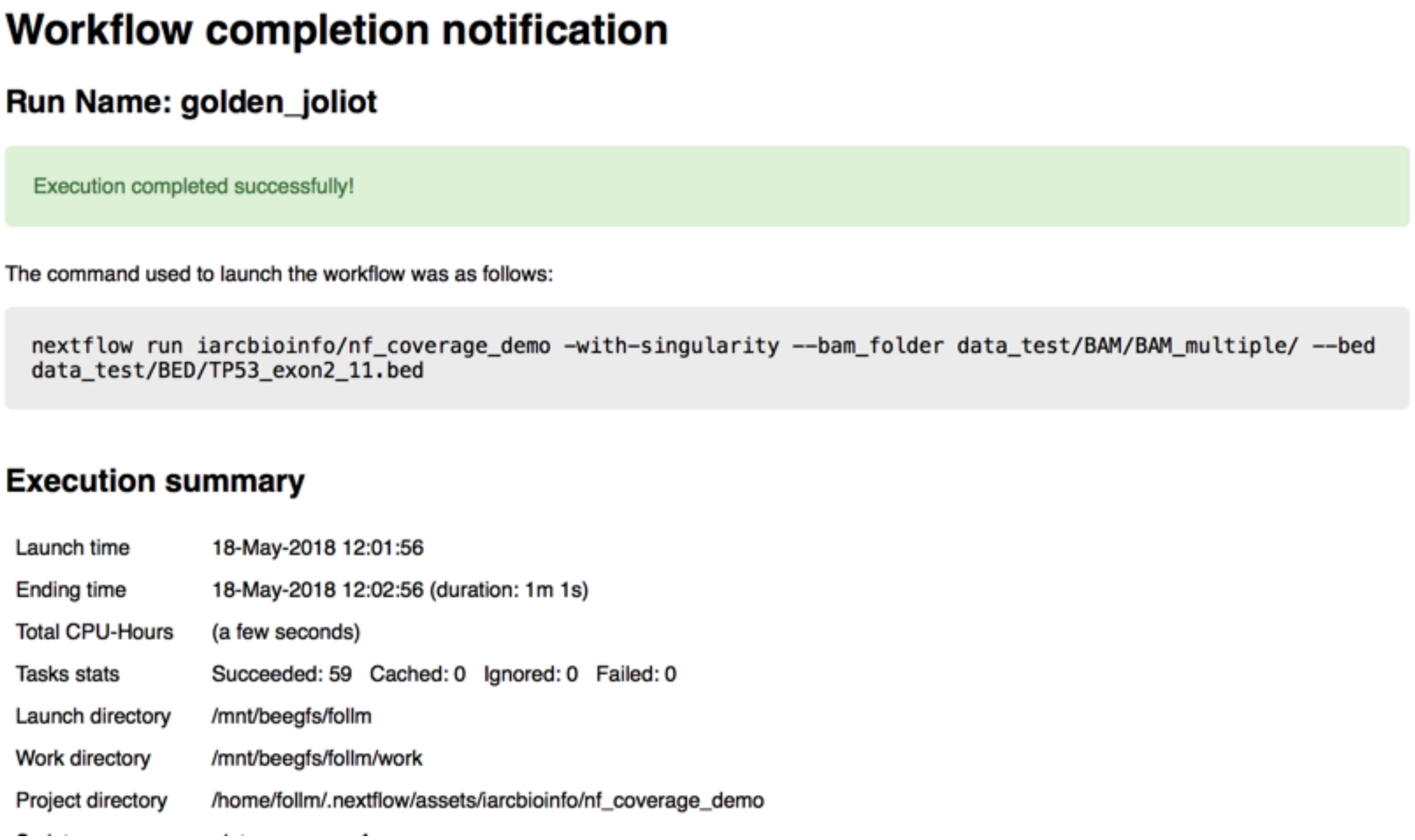

nf_coverage_demo — -bash — 129×33

```
x140083:nf_coverage_demo follm$ nextflow run plot_coverage.nf -resume --bam_folder BAM/ --bed TP53.bed
N E X T F L O W  ~  version 0.23.4
Launching `plot_coverage.nf` [sad_mclean] - revision: 66d7be595f
[warm up] executor > local
[2c/365d33] Cached process > coverage (2)
[03/aa34ca] Cached process > coverage (8)
[82/9c63d3] Cached process > coverage (3)
[09/408295] Cached process > coverage (6)
[64/3e5047] Cached process > coverage (1)
[2a/fb03ab] Cached process > coverage (4)
[d6/95d6c5] Cached process > coverage (5)
[37/0b8c55] Cached process > coverage (7)
[e7/2d0a84] Cached process > coverage (9)
[2d/03ec82] Cached process > mean (3)
[7e/3b10dd] Cached process > mean (4)
[ef/aac7cc] Cached process > coverage (11)
[5c/18cd69] Cached process > coverage (15)
[f9/6335e0] Cached process > coverage (13)
[43/db85f2] Cached process > coverage (14)
[d2/70957e] Cached process > coverage (18)
[f4/cf2a3f] Cached process > coverage (12)
[19/b7d0cd] Submitted process > coverage (22)
[22/c0d766] Submitted process > mean (9)
[4f/decdf2] Submitted process > coverage (23)
```

International Agency for Research on Cancer

World Health Organization

Notification (`-with-notification`)



Workflow completion notification

Run Name: golden_joliot

Execution completed successfully!

The command used to launch the workflow was as follows:

```
nextflow run iarcbioinfo/nf_coverage_demo -with-singularity --bam_folder data_test/BAM/BAM_multiple/ --bed
data_test/BED/TP53_exon2_11.bed
```

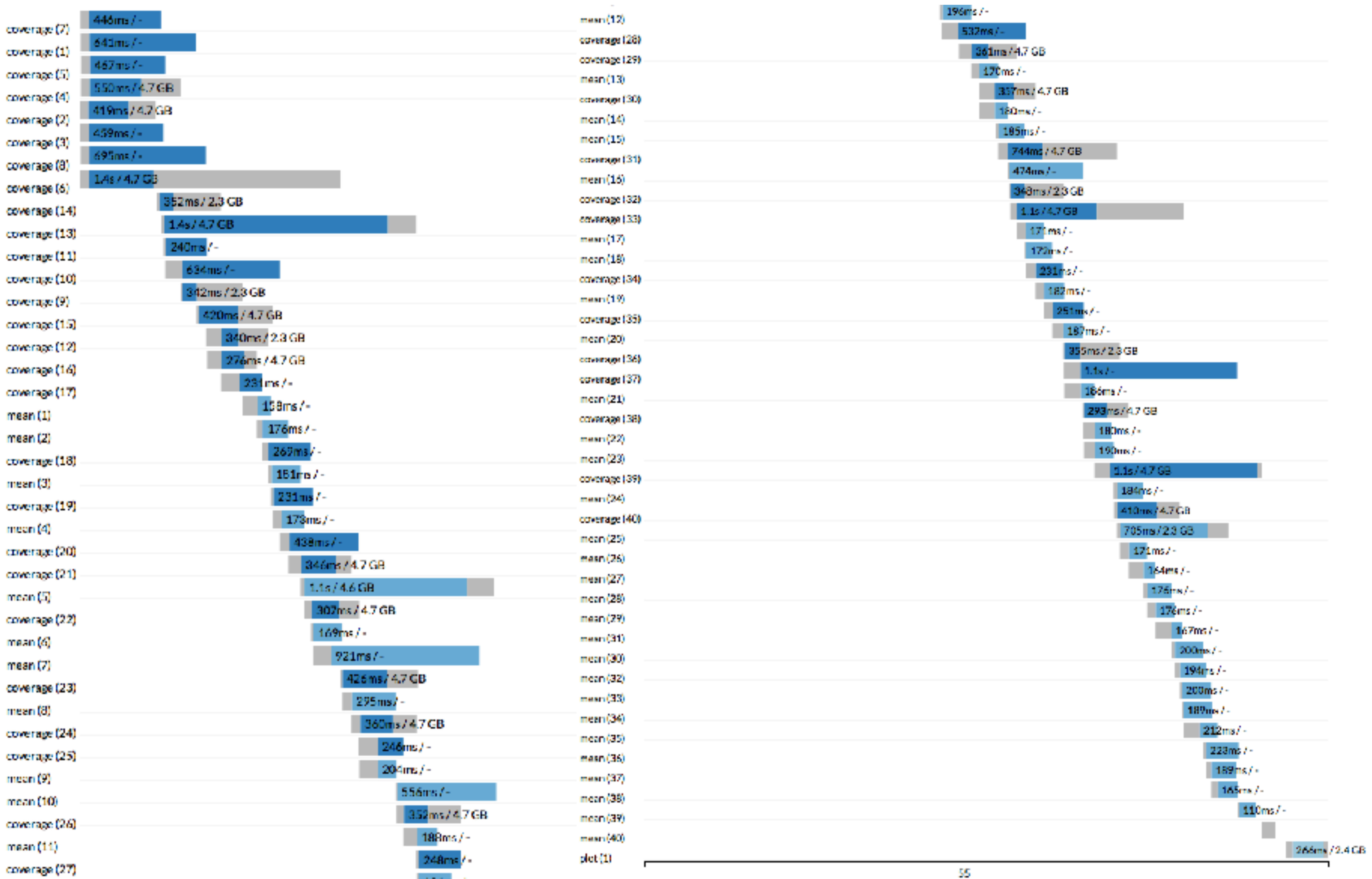
Execution summary

Launch time	18-May-2018 12:01:56
Ending time	18-May-2018 12:02:56 (duration: 1m 1s)
Total CPU-Hours	(a few seconds)
Tasks stats	Succeeded: 59 Cached: 0 Ignored: 0 Failed: 0
Launch directory	/mnt/beegfs/follm
Work directory	/mnt/beegfs/follm/work
Project directory	/home/follm/.nextflow/assets/iarcbioinfo/nf_coverage_demo

Timeline (`-with-timeline`)

Processes execution timeline

Launch time: 03 Mar 2017 11:16
Elapsed time: 4.9s


coverage (7) 446ms / -
coverage (1) 641ms / -
coverage (5) 467ms / -
coverage (4) 550ms / 4.7 GB
coverage (2) 419ms / 4.2 GB
coverage (3) 459ms / -
coverage (8) 695ms / -
coverage (6) 1.4s / 4.7 GB
coverage (14) 352ms / 2.3 GB
coverage (13) 1.4s / 4.7 GB
coverage (11) 240ms / -
coverage (10) 634ms / -
coverage (9) 312ms / 2.3 GB
coverage (15) 420ms / 4.7 GB
coverage (12) 340ms / 2.3 GB
coverage (16) 276ms / 4.7 GB
coverage (17) 231ms / -
mean (1) 158ms / -
mean (2) 176ms / -
coverage (18) 267ms / -
mean (3) 181ms / -
coverage (19) 231ms / -
mean (4) 173ms / -
coverage (20) 438ms / -
coverage (21) 346ms / 4.7 GB
mean (5) 1.1s / 4.6 GB
coverage (22) 302ms / 4.7 GB
mean (6) 169ms / -
mean (7) 921ms / -
coverage (23) 426ms / 4.7 GB
mean (8) 275ms / -
coverage (24) 360ms / 4.7 GB
coverage (25) 246ms / -
mean (9) 204ms / -
mean (10) 556ms / -
coverage (26) 352ms / 4.7 GB
mean (11) 188ms / -
coverage (27) 248ms / -
mean (12) 196ms / -
coverage (28) 532ms / -
coverage (29) 361ms / 4.7 GB
mean (13) 170ms / -
coverage (30) 357ms / 4.7 GB
mean (14) 180ms / -
mean (15) 185ms / -
coverage (31) 744ms / 4.7 GB
mean (16) 474ms / -
coverage (32) 348ms / 2.3 GB
coverage (33) 1.1s / 4.7 GB
mean (17) 171ms / -
mean (18) 172ms / -
coverage (34) 231ms / -
mean (19) 182ms / -
coverage (35) 251ms / -
mean (20) 187ms / -
coverage (36) 355ms / 2.3 GB
coverage (37) 1.1s / -
mean (21) 186ms / -
coverage (38) 293ms / 4.7 GB
mean (22) 180ms / -
mean (23) 190ms / -
coverage (39) 1.1s / 4.7 GB
mean (24) 184ms / -
coverage (40) 410ms / 4.7 GB
mean (25) 705ms / 2.3 GB
mean (26) 171ms / -
mean (27) 164ms / -
mean (28) 175ms / -
mean (29) 176ms / -
mean (31) 167ms / -
mean (30) 200ms / -
mean (32) 194ms / -
mean (33) 200ms / -
mean (34) 189ms / -
mean (35) 212ms / -
mean (36) 223ms / -
mean (37) 189ms / -
mean (38) 165ms / -
mean (39) 110ms / -
mean (40)
plot (1) 266ms / 2.4 GB
5s