

nf-core

A community effort to collect a curated set of analysis pipelines built using Nextflow.

nf-core

Presentation inspired from the nf-core documentation



Nicolas Fontrodona

February 1, 2022



What is nf-core ?

nf-core pipelines

Launching a nf-core pipeline

Documentation of nf-core pipelines

nf-core helper tools

Creating a nf-core pipeline

nf-core pipelines on the PSMN

nf-core utility

What is nf-core ?



A community effort to collect analysis pipelines built with **Nextflow** that follow strict guidelines.

What is nf-core ?



A community effort to collect analysis pipelines built with **Nextflow** that follow strict guidelines.

Nextflow is a workflow manager able to manage the parallelisation of complex pipelines

What is nf-core ?



A community effort to collect analysis pipelines built with **Nextflow** that follow strict guidelines.

Nextflow is a workflow manager able to manage the parallelisation of complex pipelines

Nf-core as three targets:

- ▶ **Facilities**: Automated and efficient pipelines with reproducibility of results
- ▶ **Single users**: Documented, portable, and easy to use pipelines
- ▶ **Developers**: Write a nextflow pipeline using templates and helper tools

What is nf-core ?



The nf-core community develops pipelines for various bioinformatics analyzes that adhere to strict guidelines¹:

- ▶ Documentation
- ▶ CI-testing
- ▶ Stable release
- ▶ Packaged processes
- ▶ Portable and reproducible

¹Source: <https://nf-co.re/>



The nf-core pipelines are listed here: <https://nf-co.re/pipelines>:

Pipelines

Browse the **58** pipelines that are currently available as part of nf-core.

Available Pipelines

Can you think of another pipeline that would fit in well? [Let us know!](#)

Search keywords: Filter: Released 33 Under development 20 Archived 5 Sort: Last Release Alphabetical Stars Display:

nf-core/eager ✓

★ 61

[adna](#) [ancient-dna-analysis](#) [ancientdna](#) [genome](#) [metagenomics](#)
[pathogen-genomics](#) [population-genetics](#)

A fully reproducible and state-of-the-art ancient DNA analysis pipeline

Version 2.4.2 Published 4 days ago

nf-core/cutandrun ✓

★ 20

[cutandrun](#) [cutandrun-seq](#) [cutandtag](#) [cutandtag-seq](#)

Analysis pipeline for CUT&RUN and CUT&TAG experiments that includes QC, support for spike-ins, IgG controls, peak calling and downstream analysis.

Version 1.1 Published 1 week ago



Prerequisites to launch a nf-core pipeline:

- ▶ Nextflow
- ▶ A software packaging tool (that can be managed by Nextflow)
 - ▶ Pipeline softwares don't need to be installed on your computer
 - ▶ They are packaged inside containers (or environments) and downloaded when the pipeline needs them
 - ▶ Key step for **reproducibility**
 - ▶ All nf-core pipelines can work with Docker and Singularity and most of them have support for 'Conda'

Launching a nf-core pipeline



Launching the Rnaseq pipeline on test data:

```
1 $ # Install nextflow
2 $ curl -s https://get.nextflow.io | bash
3 $ mkdir bin; mv nextflow ./bin/
4 $ # Launching a rnaseq pipeline with test inputs
5 $ bin/nextflow run nf-core/rnaseq -profile test,singularity
6 N E X T F L O W ~ version 21.10.6
7 Launching 'nf-core/rnaseq' [festering_fermi] - revision: 646723c70f
   [master]
8 ...
9 [c7/8318de] process > NFCORE_RNASEQ:RNASEQ:MULTIQC (1)
   [100%] 1 of 1
10 Pulling Singularity image https://depot.galaxyproject.org/
   singularity/multiqc:1.11--pyhdfd78af_0 [cache /media/Data/
   Projects/cours/nf-core/work/singularity/depot.galaxyproject.org-
   singularity-multiqc-1.11--pyhdfd78af_0.img]
11 ...
12 -[nf-core/rnaseq] Pipeline completed successfully -
13 ...
14 Completed at: 28-janv.-2022 17:53:44
15 Duration      : 17m 58s
16 CPU hours     : 0.4
17 Succeeded    : 202
```

Launching a nf-core pipeline

Additional data



By default the command `nextflow run mypipe [options]` runs the latest version of the pipeline `mypipe`.

To run a particular version of the pipeline you can use the `-r/-revision` flag. For example with the pipeline `rnaseq`:

```
1 bin/nextflow run nf-core/rnaseq -profile test,singularity -r 3.4
```

Launching a nf-core pipeline

Additional data



Also note that the code of the pipeline is not downloaded in your working directory:

```
1 $ bin/nextflow list
2 nf-core/rnaseq
3 $ bin/nextflow info nf-core/rnaseq
4 project name: nf-core/rnaseq
5 repository  : https://github.com/nf-core/rnaseq
6 local path  : /home/nicolas/.nextflow/assets/nf-core/rnaseq
7 main script : main.nf
8 description : Nextflow RNA-Seq analysis pipeline, part of the nf-
   core community.
9 author      : Phil Ewels, Rickard Hammaren
```

To do so you can use :

```
1 bin/nextflow clone nf-core/rnaseq -r 3.4
```

Additional data about nextflow's command line interface can be found [here](#)

Documentation of nf-core pipelines



The documentation of a nf-core pipeline can be easily found by going on this url '<https://nf-co.re/pipelines>' and clicking on a pipeline name.

nf-core/rnaseq 🔗 link

RNA sequencing analysis pipeline using STAR, RSEM, HISAT2 or Salmon with gene/isoform counts and extensive quality control.

📄 00:00

🚀 Launch version 3.5

<https://github.com/nf-core/rnaseq>

➔ Introduction **📄 Results** 📄 Usage docs 📄 Parameter docs 📄 Output docs 📄 Releases & Statistics 3.5

Introduction

nf-core/rnaseq is a bioinformatics pipeline that can be used to analyse RNA sequencing data obtained from organisms with a reference genome and annotation.

On release, automated continuous integration tests run the pipeline on a full-sized dataset obtained from the ENCODE Project Consortium on the AWS cloud infrastructure. This ensures that the pipeline runs on AWS, has sensible resource allocation defaults set to run on real-world datasets, and permits the persistent storage of results to benchmark between pipeline releases and other analysis sources. The results obtained from running the full-sized tests individually for each `aligner` option can be viewed on the [nf-core website](#) e.g. the results for running the pipeline with `aligner: star_sataion` will be in a folder called `aligner_star_sataion` and so on.

The pipeline is built using [Nextflow](#), a workflow tool to run tasks across multiple compute infrastructures in a very portable manner. It uses Docker/Singularity containers making installation trivial and results highly reproducible. The [Nextflow DSL2](#) implementation of this pipeline uses one container per process which makes it much easier to maintain and update software dependencies. Where possible, these processes have been submitted to and installed from [nf-core/modules](#) in order to make them available to all nf-core pipelines, and to everyone within the Nextflow community!

🔍 command

📅 4 clones in last 2 years

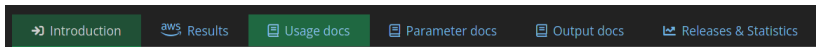
51023

| stars | watchers |
|-------|----------|
| 416 | 82 |

| last release | last updated |
|--------------|---------------|
| open issues | 3 days ago |
| 37 | pull requests |
| | 414 |

👤 collaborators

Documentation of nf-core/rnaseq pipeline



- ▶ **Introduction:** Describes how the program works and what are its steps
- ▶ **Usage docs:** Describes how to use the pipeline
- ▶ **Parameters docs:** Documentation for all available parameters
- ▶ **Output docs:** Describes the output produced by the pipeline

You can display the parameters doc with the following command:

```
1 $ bin/nextflow run nf-core/rnaseq -r 3.5 --help
```



To ease the management of nf-core pipelines the nf-core team has created the **Helper Tools**. They can help you to:

- ▶ List available pipelines and versions
- ▶ Run a pipeline with interactive parameter prompt
- ▶ View software licences in a pipeline
`nf-core licences PIPELINE_NAME`
- ▶ Create a pipeline that follow the nf-core best practices
- ▶ And more...

Let's review some commands of these tools

To learn more about these tools, check the [Helper Tools documentation](#)



Three possible kind of installations:

► conda

```
1 $ # installation with conda
2 $ conda install nf-core
3 $ # create an environment with nf-core/tools and nextflow
4 $ conda create --name nf-core python=3 nf-core nextflow
5 $ # Activate the new environment to run nf-core tools commands
6 $ conda activate nf-core
```

► python package index

```
1 $ pip install nf-core
```

► Docker

```
1 $ # Display the help of nf-core/tools where PATH is a folder
   # that will contain .nfcORE and .nextflow cache directories
2 $ docker run -it -v $PWD:$PWD -w $PWD -u $(id -u):$(id -g) --rm
   -e HOME=$PATH nfcORE/tools
3 $ # Creating an alias
4 $ alias nfcORE="docker run -it -v $PWD:$PWD -w $PWD -u $(id -u):
   $(id -g) --rm -e HOME=$PWD nfcORE/tools"
5 $ nfcORE --help
```

nf-core helper tools

Listing pipelines



```
1 $ nf-core list | head -n 50
2 |-----|
3 | Pipeline | Stars | Latest | Released | Last Pulled |
4 | Name | | Release | | |
5 |-----|-----|-----|-----|
6 | eager | 61 | 2.4.2 | 6 days ago | - |
7 |-----|-----|-----|-----|
8 | eager | 61 | 2.4.2 | 6 days ago | - |
9 | cutandrun | 20 | 1.1 | 1 weeks ago | - |
10 | mhcquant | 19 | 2.2.0 | 2 weeks ago | - |
11 $ # You can display only the pipelines with a given list of keywords
12 # in title description and topics
13 $ nf-core list rna-seq rnaseq
14 | Pipeline Name | Stars | Latest Release | Released |
15 |-----|-----|-----|-----|
16 | rnaseq | 417 | 3.5 | 1 months ago |
17 | smrnaseq | 35 | 1.1.0 | 8 months ago |
18 | dualrnaseq | 5 | 1.0.0 | 12 months ago |
19 | scflow | 12 | dev | - |
20 $ # The pipelines are sorted by the column Released by default. You
21 # can change that
22 $ nf-core list -s pulled # sort by when you last pulled a local copy
23 $ nf-core list -s name # sort by name
24 $ nf-core list -s stars # sort by github stars
```




Nf-core pipelines can have a lot of parameters. Nf-core helper tools help you to launch a pipeline and chose its parameters by using either a **web-based graphical interface** or an interactive **command-line wizard tool**.

1. Those tools display the documentation for each parameters
2. They validate your inputs
3. They save out parameters into a file named `nf-params.json` and used by nextflow with the flag `-params-file`



Using the wizard tool:

```
1 $ nf-core launch rnaseq
2 ...
3 ? Select release / branch: 3.5 [release]
4 ? Nextflow command-line flags
5 General Nextflow flags to control how the pipeline runs.
6 These are not specific to the pipeline and will not be saved in any
   parameter file. They are just used when building the nextflow
   run launch command.
7 (Use arrow keys)
8 Continue >>
9 ...
10 ? Input/output options
11 Define where the pipeline should find input data and save output
   data.
12 (Use arrow keys)
13 input
14 ...
15 ? --input
16 Path to comma-separated file containing information about the
   samples in the experiment.
17 You will need to create a design file with information about the...
18
19 ?
```

nf-core helper tools

Launch a pipeline



Using the web interface:

```
1 $ nf-core launch rnaseq
2 ...
3 ? Select release / branch: 3.5 [release]
4 ? Choose launch method Web based
```

Nextflow command-line flags Launch

Nextflow command-line flags

General Nextflow flags to control how the pipeline runs.

These are not specific to the pipeline and will not be saved in any parameter file. They are just used when building the 'nextflow run' launch command.

?
Unique name for this nextflow run

Configuration profile

Work directory for intermediate files

True False ?
Resume previous run, if found

>_ Input/output options

Define where the pipeline should find input data and save output data.

?
Path to comma-separated file containing information about the samples in the experiment.

Creating a nf-core pipeline

Guidelines



If you want to create a nf-core pipeline it has to follow some guidelines:

- ▶ Workflow specificity
- ▶ Workflow size
- ▶ Use the template `nf-core create`
- ▶ Built with Nextflow
- ▶ MIT Licence
- ▶ Software packaged with docker
- ▶ Continuous integration testing
- ▶ Stable release tags
- ▶ Common pipeline structure and usage (Standard file, folder, and parameters names)
- ▶ A unique workflow that runs in a single command (Not multiple separate workflows)
- ▶ Great documentation
- ▶ Contact to the person responsible for the pipeline
- ▶ No failure with `nf-core lint` tests. Checks for all nf-core community guidelines
- ▶ Workflow name in lowercase without punctuation
- ▶ Credits and Acknowledgements
- ▶ Be in touch with the community
- ▶ Other recommended features

Check [this page](#) for more details



Example with the rnaseq pipeline:

```
1 $ curl -s https://get.nextflow.io | bash # installation of nextflow
2 $ mkdir bin; mv nextflow ./bin/ # moving it into ./bin
3 $ bin/nextflow clone nf-core/rnaseq -r 3.5 .# cloning the pipeline
```

Now, we need to define a config file to run the pipeline on the PSMN. Let's create a config file `psmn.config` into the `rnaseq/conf` folder.

Note

The config files of nf-core pipelines are always located into the directory `pipeline_name/conf` (except for the main config file `nextflow.config` located in the root directory of the pipeline).



conf/psmn.config

```
1 singularity.enabled = true
2 singularity.cacheDir = "./bin"
3 singularity.runOptions = "--bind /Xnfs,/scratch"
4 errorStrategy = 'retry'
5 maxRetries = 3
6 params.singularity_pull_docker_container = true // build images from
   docker containers
7 process{
8   ext.singularity_pull_docker_container = true // build images from
   docker containers
9   withLabel: process_high {
10     // see next slide
11   }
12   withLabel: process_medium {
13     // see next slide
14   }
15   withLabel: 'process_low|process_long' {
16     // see next slide
17   }
18 }
```

nf-core pipelines on the PSMN



```
1  withLabel: process_high {
2    executor = "sge"
3    clusterOptions = "-cwd -V"
4    memory = "150GB"
5    cpus = 32
6    time = "24h"
7    queue = "CLG6242deb384*,CLG5218deb192*,CLG6226Rdeb192*"
8    penv = "openmp32"
9  }
10 withLabel: process_medium {
11   executor = "sge"
12   clusterOptions = "-cwd -V"
13   memory = "20GB"
14   cpus = 8
15   time = "15h"
16   queue = "CLG6242deb384C,CLG6226Rdeb192D,CLG5218deb192D,
17         SLG6142deb384C"
18   penv = "openmp8"
19 }
19 withLabel: 'process_low|process_long' {
20   executor = "sge"
21   clusterOptions = "-cwd -V"
22   memory = "16GB"
23   cpus = 1
24   time = "30h"
25   queue = "monointeldeb128"
26 }
```



To run a test on the psmn, it's highly advisable to set the `queue` and `cpus` parameters to `monointeldeb128` and `1` respectively for all processes. Remove also the `penv` parameter. Let's create another file `psmn_test.config` dedicated to test the pipeline. This file has the following content:

```
1  //--snip (same thing as the file psmn.config)
2  withLabel: process_high {
3    //--snip
4    memory = "16GB" // reduce the available memory for the process
5    cpus = 1
6    time = "2h" // You can also set the maximum duration to 2h
7    queue = "monointeldeb128"
8  }
9  withLabel: process_medium {
10   //--snip
11   memory = "16GB"
12   cpus = 1
13   time = "2h" // You can also set the maximum duration to 2h
14   queue = "monointeldeb128"
15 }
16 }
```




To be able to combine our newly defined config file to those defined in `conf` directory, for example test input parameters, we can add the following line inside `rnaseq/nextflow.conf` under the `profiles` section:

```
1 profiles {
2   psmn_test { includeConfig 'conf/psmn_test.config' }
3   psmn { includeConfig 'conf/psmn.config' }
4   // other profiles
5 }
```

Then we can run a test with:

```
1 $ bin/nextflow rnaseq/main.nf -profile psmn_test, test
```

Then you can run the pipeline with your file:

```
1 $ bin/nextflow rnaseq/main.nf -profile psmn [PARAM]
```

It can be annoying to create those configuration files yourself, that's why I developed a tool [nf-core utility](#).



Utility dedicated to automate the creation of config file in the psmn.
Git repository: https://gitbio.ens-lyon.fr/nfontrod/nfcore_utility

Steps of the script

1. Download nextflow into a bin folder inside the output_folder



Utility dedicated to automate the creation of config file in the psmn.
Git repository: https://gitbio.ens-lyon.fr/nfontrod/nfcore_utility

Steps of the script

1. Download nextflow into a bin folder inside the output_folder
2. Clone the pipeline code into a folder named pipeline_name_version



Utility dedicated to automate the creation of config file in the psmn.
Git repository: https://gitbio.ens-lyon.fr/nfontrod/nfcore_utility

Steps of the script

1. Download nextflow into a bin folder inside the `output_folder`
2. Clone the pipeline code into a folder named `pipeline_name_version`



Utility dedicated to automate the creation of config file in the psmn.
Git repository: https://gitbio.ens-lyon.fr/nfontrod/nfcore_utility

Steps of the script

1. Download nextflow into a bin folder inside the output_folder
2. Clone the pipeline code into a folder named pipeline_name_version
3. **Create configuration files** under the pipeline_name_version/conf folder
 - ▶ **psmn_test.config** to test the pipeline on the psmn
 - ▶ **psmn.config** to launch the pipeline on the psmn with your own inputs



Utility dedicated to automate the creation of config file in the psmn.
Git repository: https://gitbio.ens-lyon.fr/nfontrod/nfcore_utility

Steps of the script

1. Download nextflow into a bin folder inside the output_folder
2. Clone the pipeline code into a folder named pipeline_name_version
3. **Create configuration files** under the pipeline_name_version/conf folder
 - ▶ **psmn_test.config** to test the pipeline on the psmn
 - ▶ **psmn.config** to launch the pipeline on the psmn with your own inputs
4. **Add those profiles** inside nextflow.config file



Utility dedicated to automate the creation of config file in the psmn.
Git repository: https://gitbio.ens-lyon.fr/nfontrod/nfcore_utility

Steps of the script

1. Download nextflow into a bin folder inside the output_folder
2. Clone the pipeline code into a folder named pipeline_name_version
3. **Create configuration files** under the pipeline_name_version/conf folder
 - ▶ **psmn_test.config** to test the pipeline on the psmn
 - ▶ **psmn.config** to launch the pipeline on the psmn with your own inputs
4. **Add those profiles** inside nextflow.config file
5. Give you examples of commands to test the pipeline or to run it with your data



The tool is packaged into a singularity image on the PSMN.
To easily use the tool, create an alias on your `$HOME/.profile`:

```
1 # To add in \code{~/.profile}
2 alias nfutil='singularity exec --pwd 'pwd' -B 'pwd':'pwd' /Xnfs/abc/
   singularity/lbmc-nfcore_utility-latest.img python3 /script/nf-
   core_4_psmn.py'
3 # source your .profile then with
4 # source ~/.profile
```




```
1 $nfutil --help
2 usage: nf-core_4_psmn.py [-h] -p STR [-o STR] [-s STR] [-l STR] [-v
   STR]
3
4 Download a nf-core pipeline in a given folder along with nextflow
   and the singularity image needed to make the pipeline work.
   Then,
5 create a config file to launch the pipeline in the psmn.
6
7 Optional arguments:
8 -h, --help                show this help message and exit
9 -o, --output_folder STR  The folder where the pipeline will
   be downloaded (default '.')
10 -s, --singularity_cachedir STR The folder where the singularity
   image will be downloaded (default
11 ".")
12 -l, --level STR          The level of information to display
   (default INFO)
13 -v, --version STR        The version of the pipeline to
   download (default 'latest')
14
15 Required arguments:
16 -p, --pipeline_name STR  The name of the pipeline to download
   . It can be preceded by nf-core/ but it's not
17 mandatory.
```



Preparing the pipeline rnaseq v3.4:

```
1 $ nfutil -p rnaseq -s ./bin -o . -v 3.4
2 downloading nextflow...
3 downloading the pipeline rnaseq
4 ...
5 Creating a psmn config files
6 The next step is to test the pipeline:
7
8 -To run a test on your computer using docker enter :
9 export NXF_SINGULARITY_CACHEDIR=bin; bin/nextflow rnaseq_3.4/main.nf
   -c rnaseq_3.4/nextflow.config -profile test,docker
10 -To run a test on your computer using singularity enter :
11 export NXF_SINGULARITY_CACHEDIR=bin; bin/nextflow rnaseq_3.4/main.nf
   -c rnaseq_3.4/nextflow.config -profile test,singularity --
   singularity_pull_docker_container
12 -To run a test on the psmn enter:
13 export NXF_SINGULARITY_CACHEDIR=bin; bin/nextflow rnaseq_3.4/main.nf
   -profile psmn_test,test
14
15 To run the pipeline with your data run:
16 export NXF_SINGULARITY_CACHEDIR=bin; bin/nextflow rnaseq_3.4/main.nf
   -profile psmn [INPUT_PARAMS]
17 Where [INPUT_PARAMS] are the input parameter for the pipeline. Check
   the documentation at https://nf-co.re/rnaseq
```



Launch a test with :

```
1 $ bin/nextflow rnaseq_3.4/main.nf -profile psmn_test, test
```

Launch the pipeline with your data:

```
1 $ bin/nextflow rnaseq_3.4/main.nf -profile psmn [INPUT_PARAMS]
```