Nextflow fundamentals training

Albert Palleja Caro

on behalf of

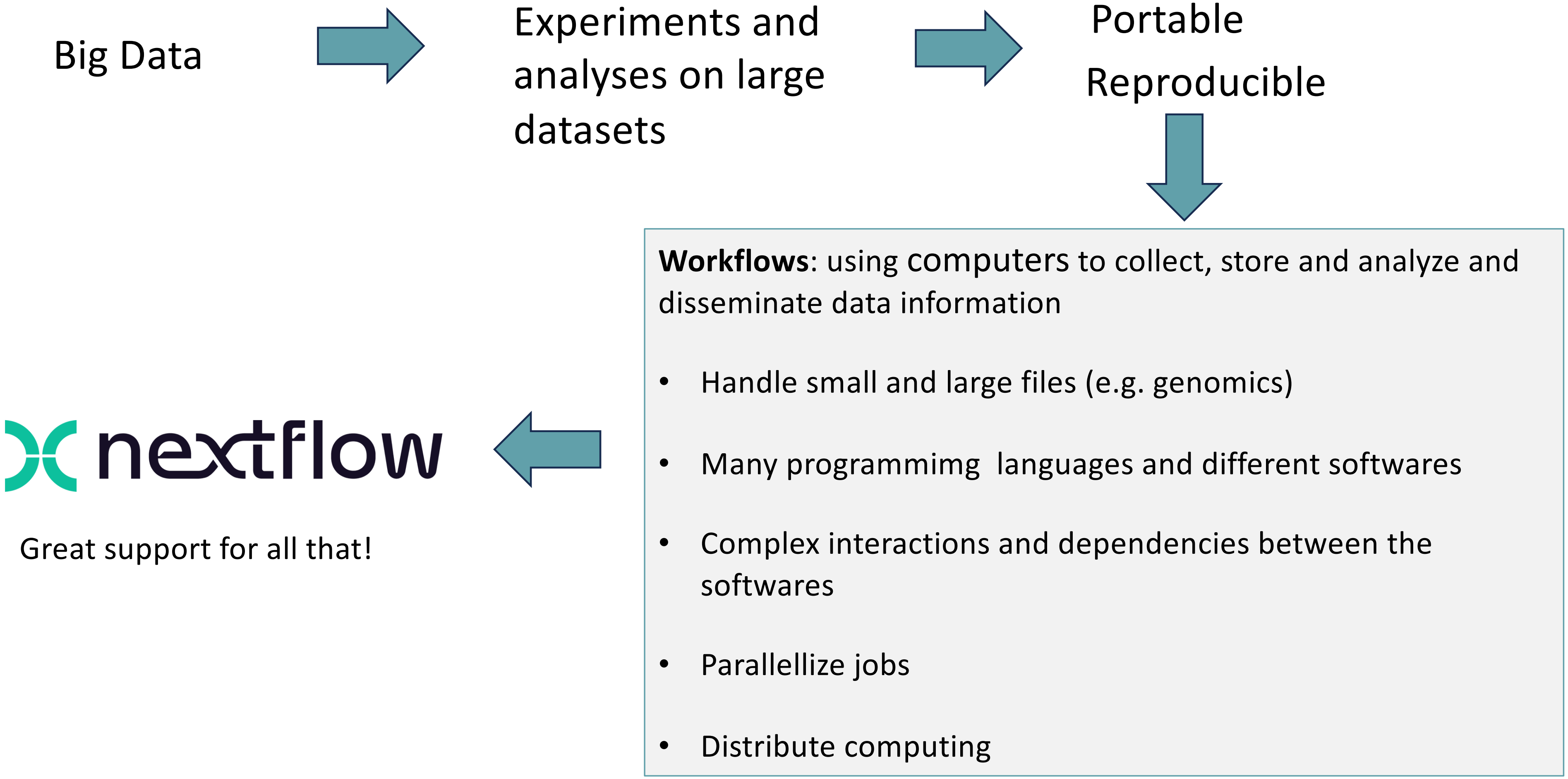
Data Science Platform

27th November 2024

Outline of the theoretical part

- What is Nextflow?
- Behind nextflow
- Why do we need a Community effort?
- Nextflow - Basic concepts
- Nextflow’s core features
- A Nextflow script
- Installing Nextflow and running the first scripts
- Nextflow and Azure batch
- nf-metagenomics pipeline
- Resources

Context



Big Data

Experiments and analyses on large datasets

Portable

Reproducible

Workflows: using computers to collect, store and analyze and disseminate data information

- Handle small and large files (e.g. genomics)
- Many programmimg languages and different softwares
- Complex interactions and dependencies between the softwares
- Parallellize jobs
- Distribute computing

nextflow

Great support for all that!

- Nextflow is a **software** - Workflow orchestrator engine
- Nextflow is a Domain-specific **language** (DSL) built on top of Groovy
- It allows writing data-intensive computational workflows
- It accommodates many languages, software environments, and computing environments
- It is oriented to bioinformatics analyses
- There is an active community giving support, organizing trainings and developing and maintaining 113 different bioinformatics analyses (on date: 27th November 2024)

Behind Nextflow

#Open Science

Make science more open!

Behind Nextflow

Open source

- We need to know software, params, options, method used and have access to the source code to be sure of what we are doing.
- Nextflow is open source.

Behind Nextflow

Open Data

- Which data are we handling?
- How was our data measured, which technology generated it, what are the units, etc…
- Findable, Accessible, Interoperable, Reusable (FAIR) data

Behind Nextflow

Open community

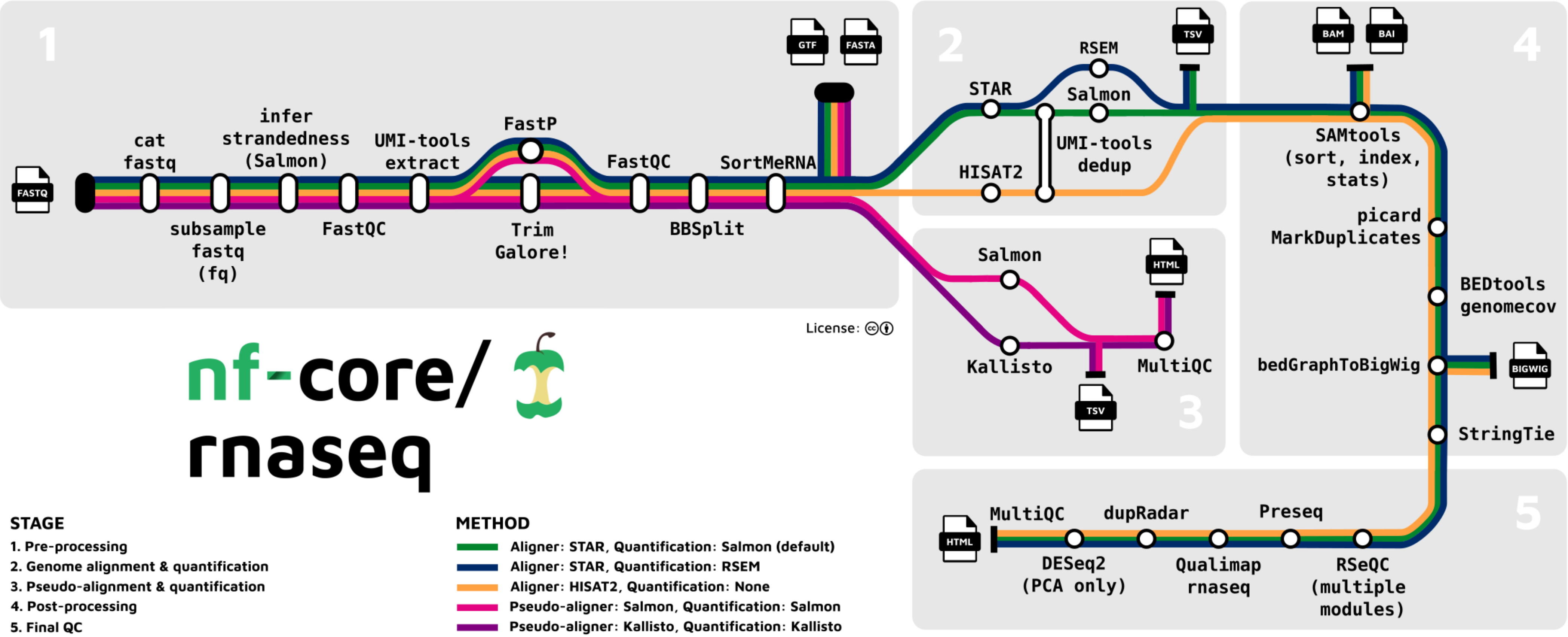
- Nextflow advocates for an open community with multiple channels to interact and connect people, forums, Slack channels, conferences, hackathons regularly organize
- nf-core initiative have developed, curate and maintain 118 bioinformatics pipelines

Behind Nextflow

Genomics workflows

- First Nextflow pipelines built for Genomics data processing characteryzed by large files per samples, different formats (binary, tab separated, etc..) parallel processess, different sorftware and languages (python, R, matlab, …) involved, nedd to take decisions for the next processes
- Nextflow can handle all that

A nextflow pipeline in metro map



STAGE

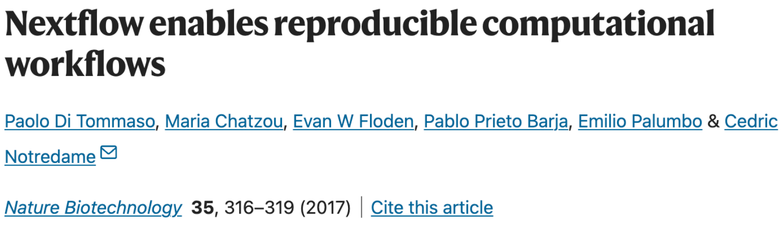
1. Pre-processing
2. Genome alignment & quantification
3. Pseudo-alignment & quantification
4. Post-processing
5. Final QC

METHOD

- 🟩 Aligner: STAR, Quantification: Salmon (default)
- 🟦 Aligner: STAR, Quantification: RSEM
- 🟧 Aligner: HISAT2, Quantification: None
- 🟥 Pseudo-aligner: Salmon, Quantification: Salmon
- 🟪 Pseudo-aligner: Kallisto, Quantification: Kallisto

Why do we need a Community effort?

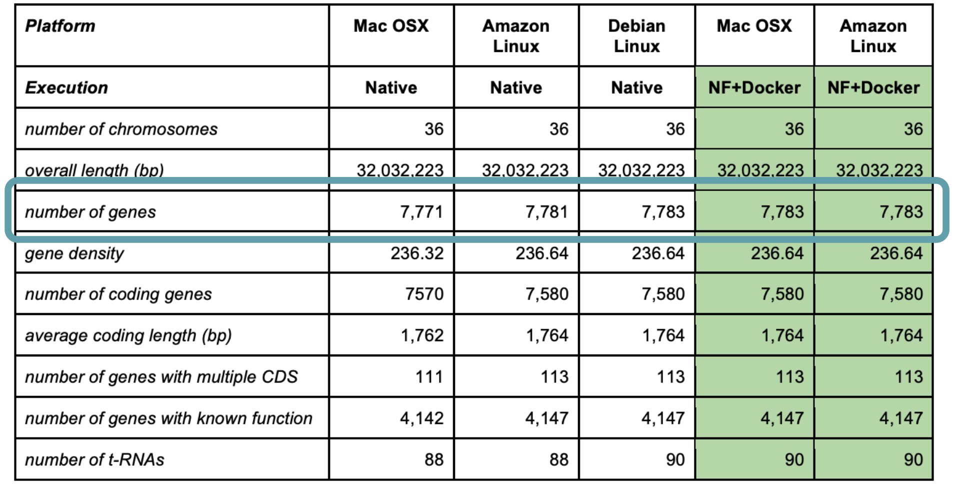
Reproducibility and even repeatability is challenging even if you are very careful keeping the same, params, versions, etc…!!!



Nextflow enables reproducible computational workflows

Paolo Di Tommaso, Maria Chatzou, Evan W Floden, Pablo Prieto Barja, Emilio Palumbo & Cedric Notredame

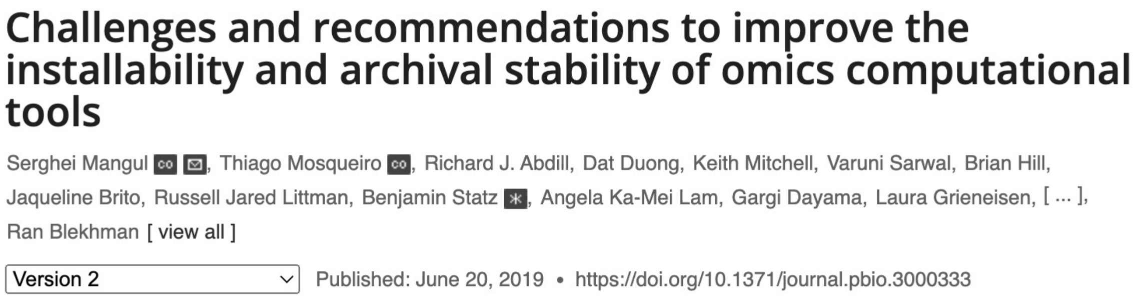
Nature Biotechnology **35**, 316–319 (2017) | Cite this article



Platform	**Mac OSX**	**Amazon Linux**	**Debian Linux**	**Mac OSX**	**Amazon Linux**
Execution	**Native**	**Native**	**Native**	**NF+Docker**	**NF+Docker**
number of chromosomes	36	36	36	36	36
overall length (bp)	32,032,223	32,032,223	32,032,223	32,032,223	32,032,223
number of genes	7,771	7,781	7,783	7,783	7,783
gene density	236.32	236.64	236.64	236.64	236.64
number of coding genes	7570	7,580	7,580	7,580	7,580
average coding length (bp)	1,762	1,764	1,764	1,764	1,764
number of genes with multiple CDS	111	113	113	113	113
number of genes with known function	4,142	4,147	4,147	4,147	4,147
number of t-RNAs	88	88	90	90	90

- Gene annotation differences depending on which operation system the pipeline was executed
- This did not happen when using Nextflow + Docker on different environments

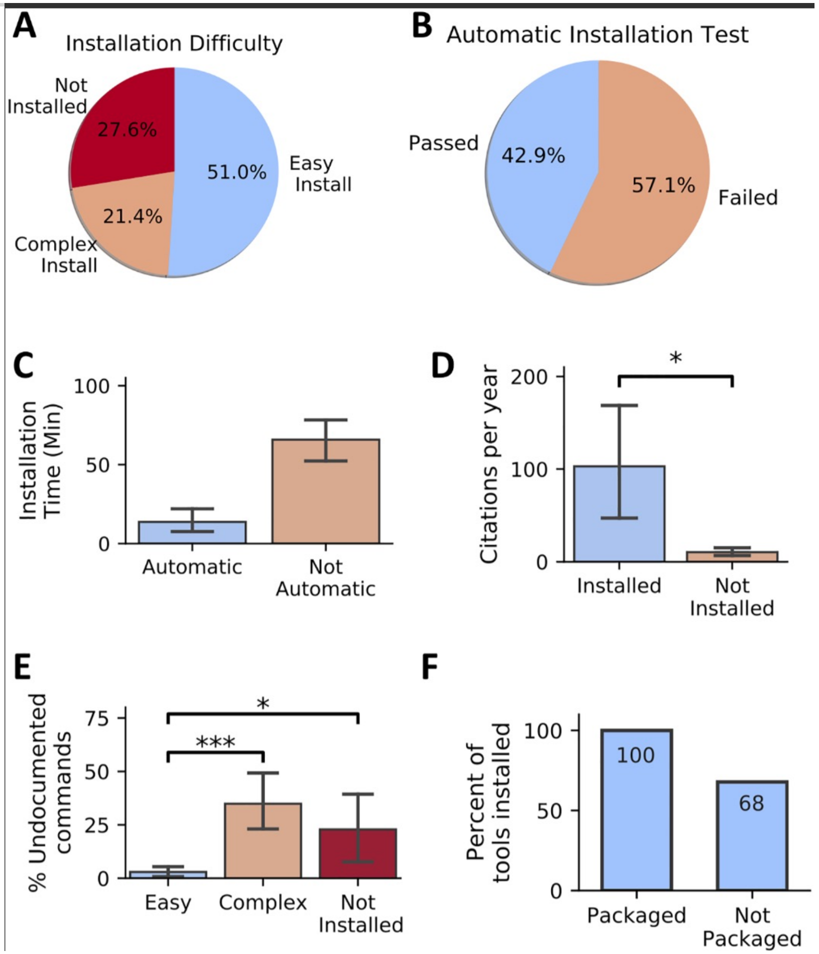
Why do we need a Community effort?



Challenges and recommendations to improve the installability and archival stability of omics computational tools

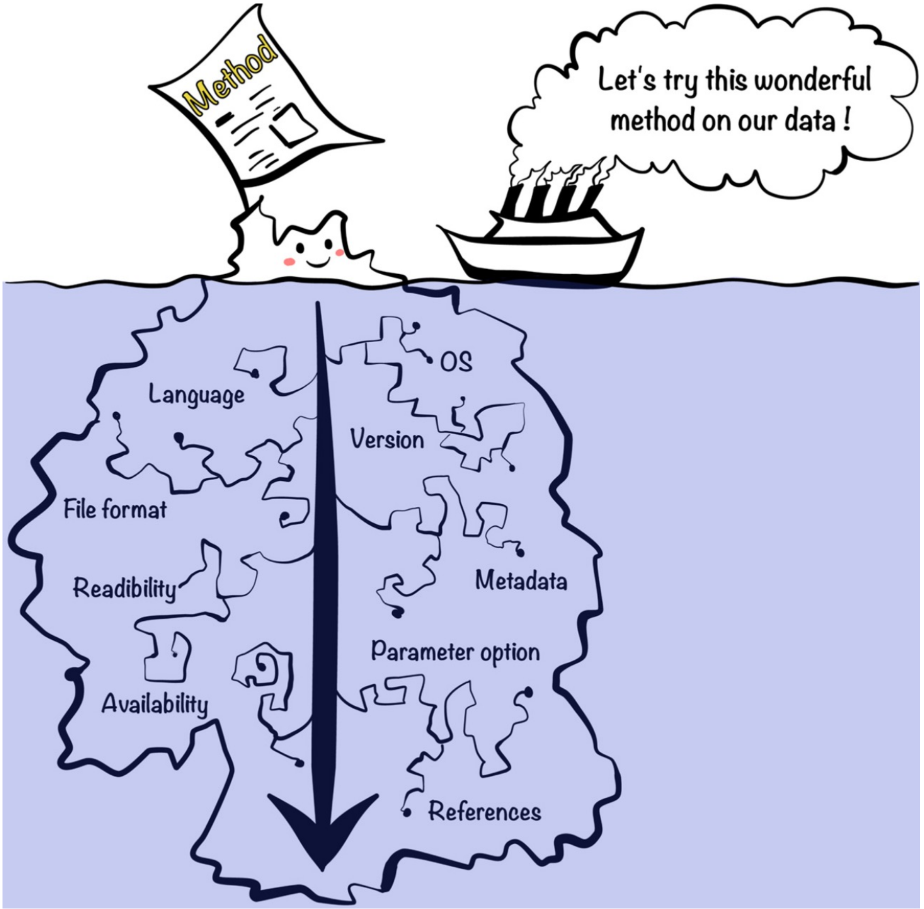
Serghei Mangul, Thiago Mosqueiro, Richard J. Abdill, Dat Duong, Keith Mitchell, Varuni Sarwal, Brian Hill, Jaqueline Brito, Russell Jared Littman, Benjamin Statz, Angela Ka-Mei Lam, Gargi Dayama, Laura Grieneisen, [...], Ran Blekhman [view all]

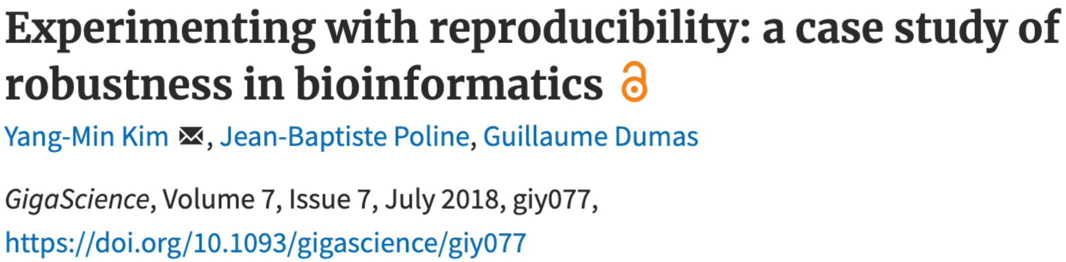
Version 2 Published: June 20, 2019 • https://doi.org/10.1371/journal.pbio.3000333



- 28% of all omics software resources were not accessible through URLs published
- Among the tools found, 49% were difficult to install or could not be installed at all!

Why do we need a Community effort?





Experimenting with reproducibility: a case study of robustness in bioinformatics

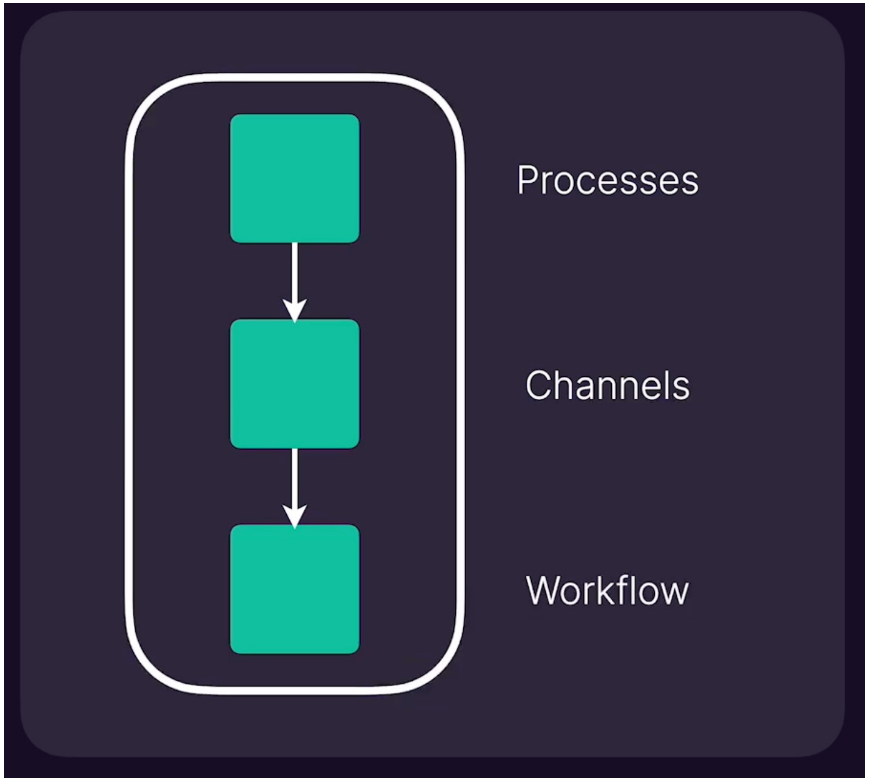
Yang-Min Kim, Jean-Baptiste Poline, Guillaume Dumas

GigaScience, Volume 7, Issue 7, July 2018, giy077, https://doi.org/10.1093/gigascience/giy077

- "First we tried to rerun the analysis with the code and the data provided by the authors. Second we reimplemented the whole method in a python package…"

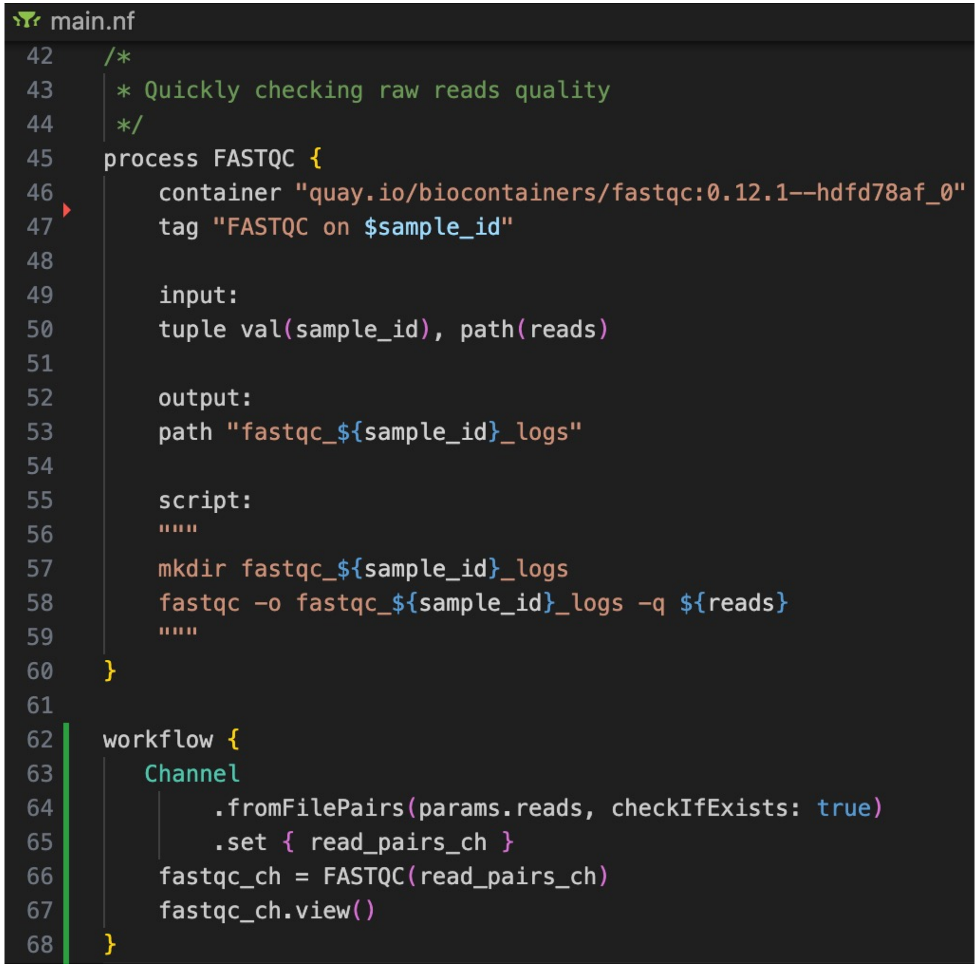
Nextflow is based on few primitives

- **Process** is every step in your pipeline, and they are executed independently isolated from each other
- We need something to share inputs and outputs between processes, these are the **channels** and there are different ways to build those
- When you have a set of processes and channels coonecting them we have a **workflow**



A Nextflow script

The .nf files are workflow scripts



```
/*
 * Quickly checking raw reads quality
 */
process FASTQC {
    container "quay.io/biocontainers/fastqc:0.12.1--hdfd78af_0"
    tag "FASTQC on $sample_id"

    input:
    tuple val(sample_id), path(reads)

    output:
    path "fastqc_${sample_id}_logs"

    script:
    """
    mkdir fastqc_${sample_id}_logs
    fastqc -o fastqc_${sample_id}_logs -q ${reads}
    """
}

workflow {
   Channel
        .fromFilePairs(params.reads, checkIfExists: true)
        .set { read_pairs_ch }
    fastqc_ch = FASTQC(read_pairs_ch)
    fastqc_ch.view()
}
```

Directives

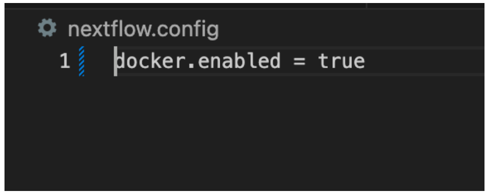
Code block

Process

Channel

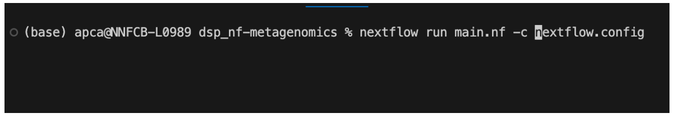
Workflow

The file nextflow.config is a configuration file that sets minimal environment properties



```
docker.enabled = true
```

How to run it:



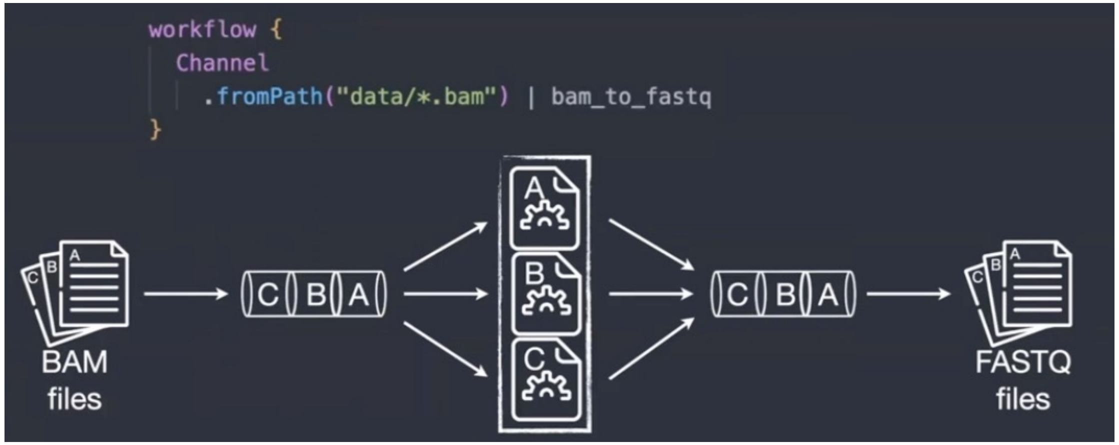
```
○ (base) apca@NNFCB-L0989 dsp_nf-metagenomics % nextflow run main.nf -c nextflow.config
```

Nextflow – Interesting features

- Implicit **parallelism** (tasks in a process are run by default in parallel)

- **Reentrancy** (resume partial runs, do not need to rerun the entire pipeline when something went wrong, it starts from wherever it stopped)

- **Reusability** (reuse different modules, subworkflows, written and containerized by the Nextflow community)



```
workflow {
  Channel
    .fromPath("data/*.bam") | bam_to_fastq
}
```

Languages, Software and computing environments

- Nextflow it supports many version controlled you can host your code there and run it directly from Nextflow

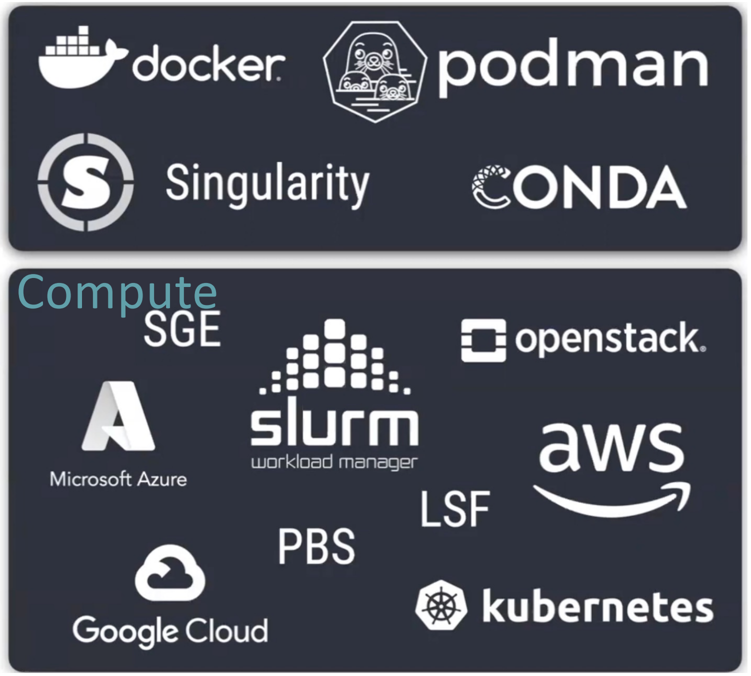
- Nextflow supports most of the containerizing options like Docker, Singularity or Conda environments, etc …

- Same thing for computing environments, it supports supports the main cloud infrastructures and most of the job schedulers of HPC

Languages

git
GitHub
Bitbucket
GitLab
Gitea
AWS CodeCommit
Azure Repos

Software

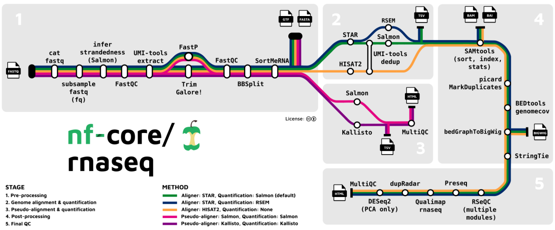


Nextflow’s core features

- **Reproducible** between runs
  - integration with code management tools
  - all packages downloaded, organized in containers, and control over computing environment
- **Portable** between systems
  - you can write the code in your laptop and can run everywhere (HPC, cloud)
  - works with most of computing environments
- **Scalable**
  - it can be run for 10 on your laptop or thousands of samples in an HPC or the cloud
- **Integration** of existing tools, systems, and industry standards

nf-core

A global community effort to collect a curated set of open-source analysis pipelines built using Nextflow.
https://nf-co.re/



118 **pipelines** available (27th November 2024) to process and analyse many different data types

Modules: 1,163 reusable components that can be integrated into pipelines

Subworkflows: 72 pre-assembled combinations of modules aimed at streamlining commonly used workflows

nf-core

Tools

- Running pipelines
- Writing pipelines
- Testing/linting
- Validating
- Automation

Community

- Developing with the community
- Standards – Use common templates
- Best practices defined
- Followed globally and through many social media channels (Slack, hackathons,…)
- Collaboration – No duplicate pipelines within nf-core
- Different modules and subworkflows need to be compatible