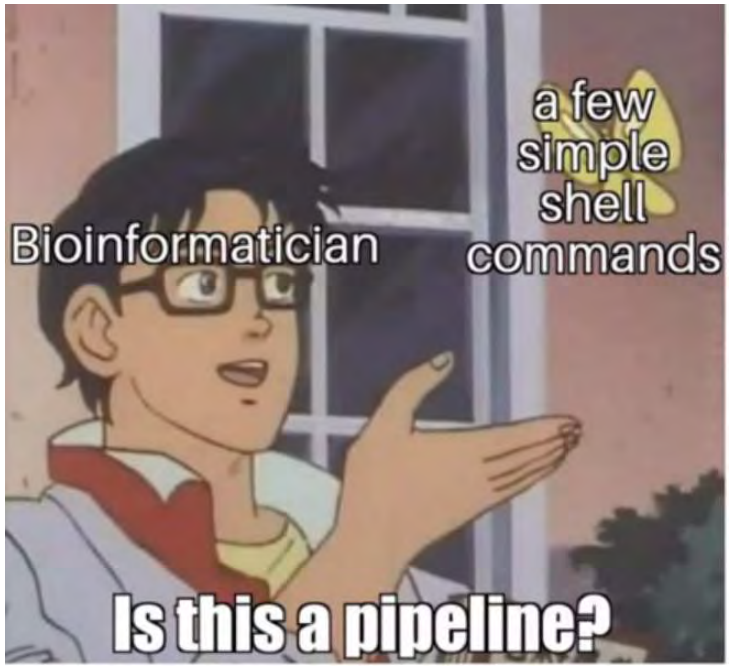
Пайпланы и пайпланы в биоинформатике

Пайплайны

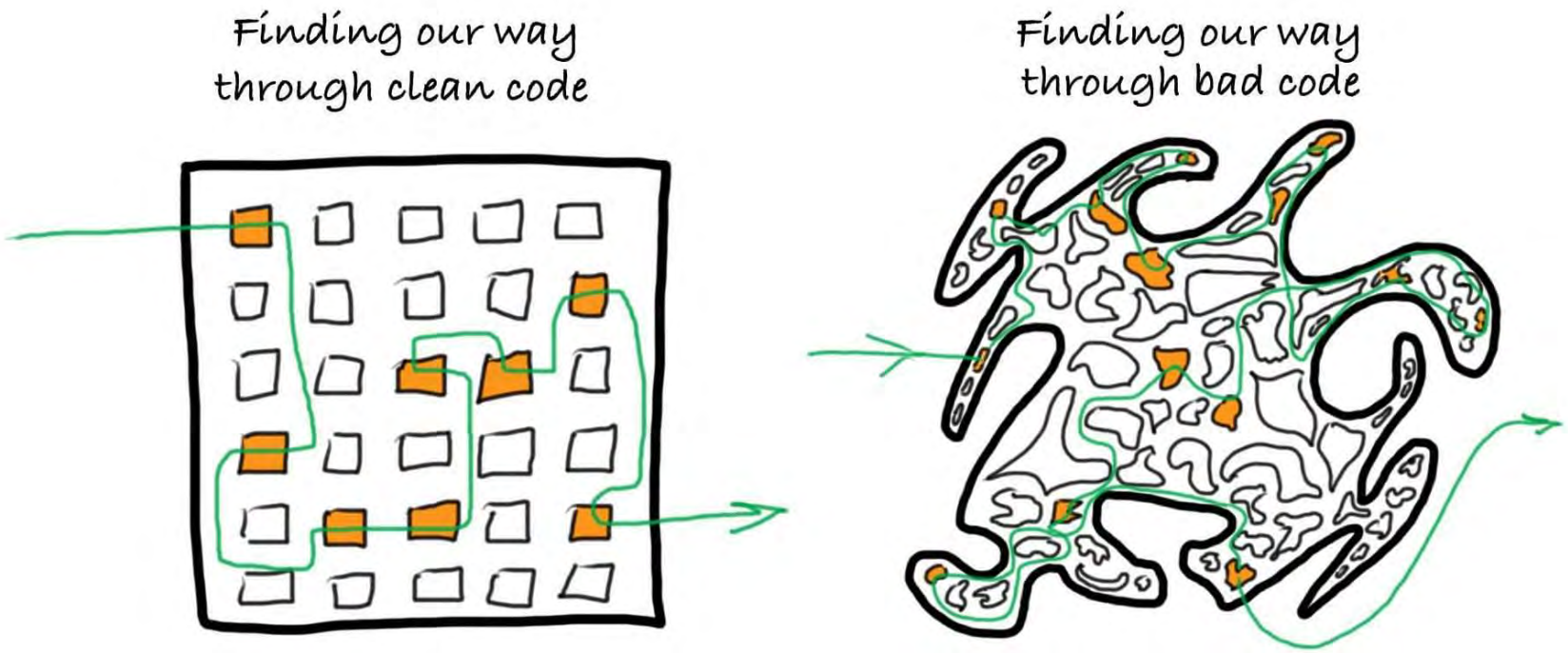
Набор инструментов, объединенных для анализа и обработки данных

- Детерминированный
- Использует кэши
- Масштабируемый
- Простые и гибкие

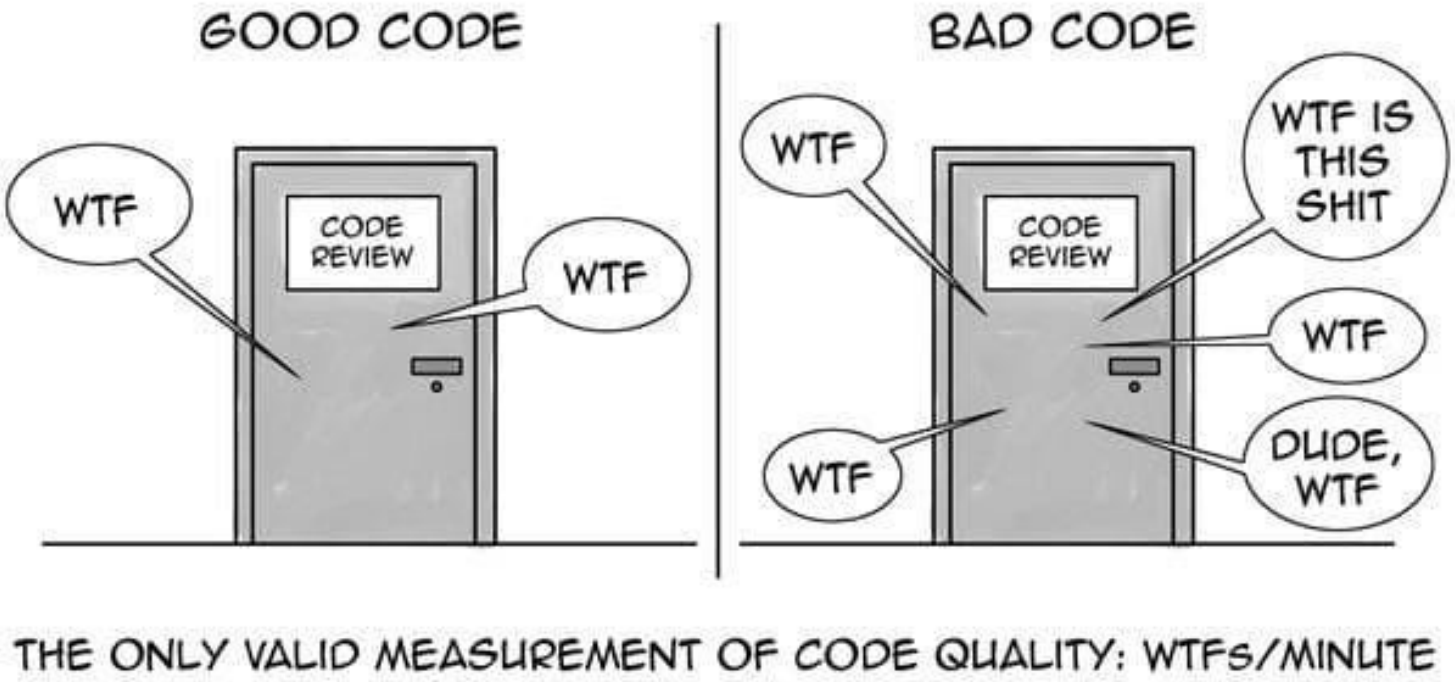
Не нужно реализовывать функции снова и снова

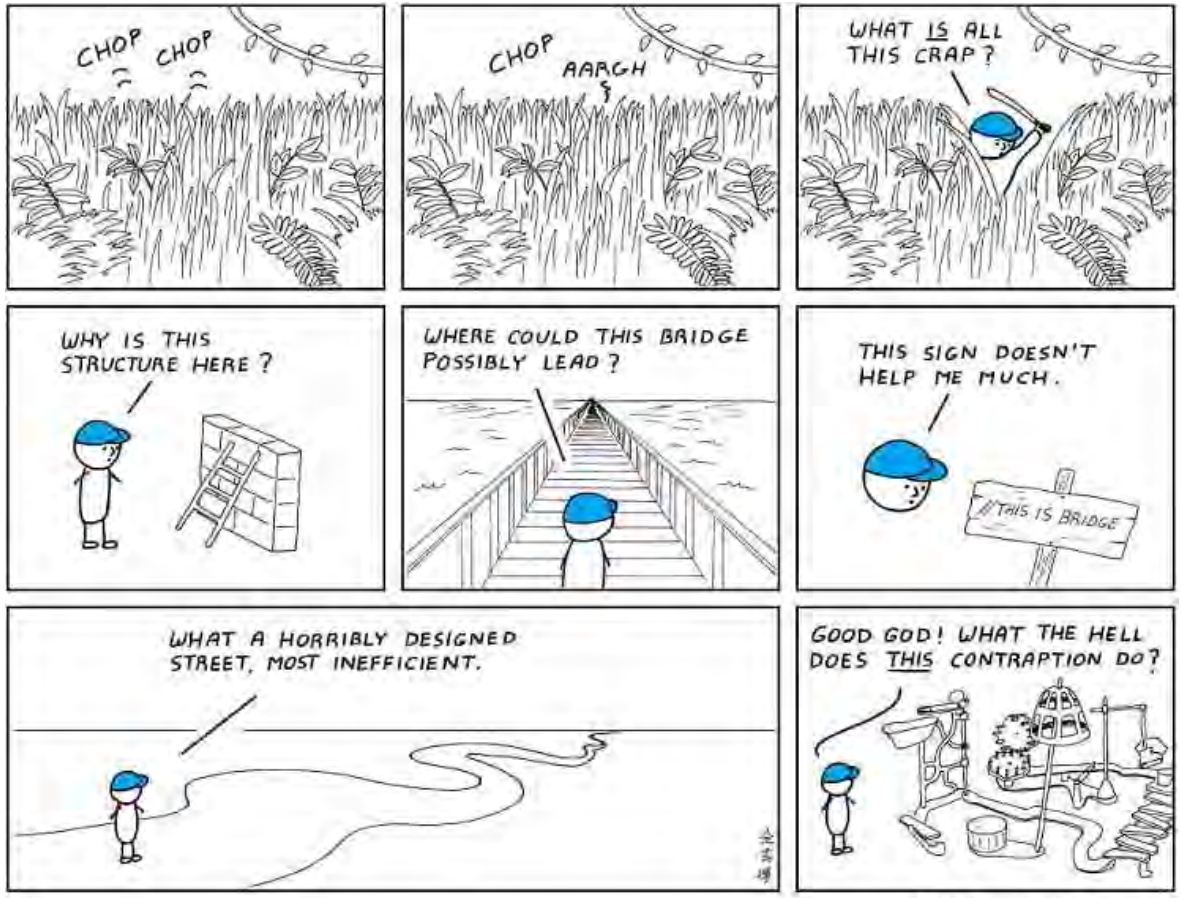


Зачем? У меня уже есть мои скрипты



Хороший или плохой





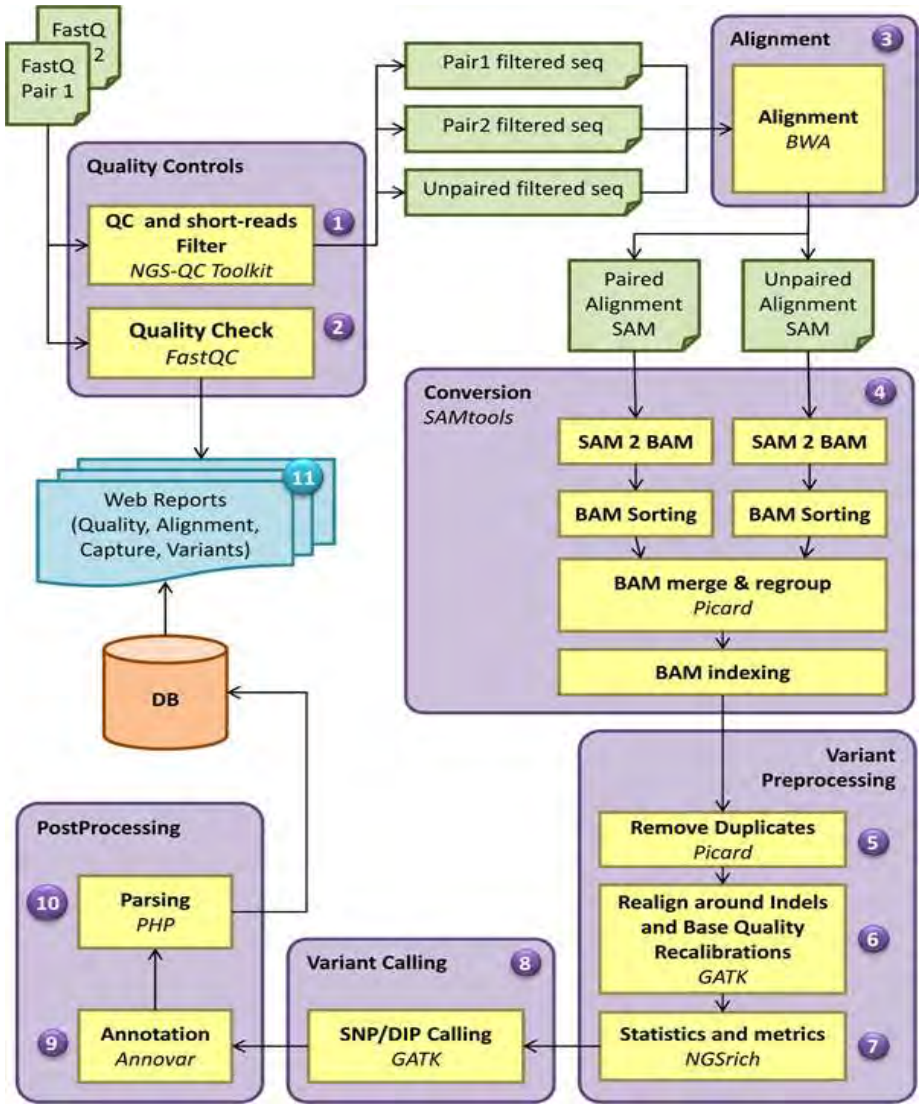
I hate reading
other people's code.

Критерий	Пайплайн	Кастомный код
Гибкость	Высокая гибкость с использованием конфигурационных файлов.	Зависит от уровня навыков пользователя в программировании и настройке.
Скорость разработки	Быстрый старт с использованием готовых правил и компонентов.	Требует времени на создание и отладку собственных скриптов.
Контроль и прозрачность	Четкий контроль зависимостей и прозрачность выполнения задач.	Полный контроль, но требует аккуратности в организации кода.
Сообщество и поддержка	Активное сообщество, множество готовых решений и обновлений.	Ограниченная поддержка, зависит от опыта пользователя.
Работа с уникальными задачами	Может быть адаптирован для уникальных и специфических задач.	Идеально подходит для уникальных и сложных задач, если написан правильно.
Зависимость от ресурсов	Требуется доступ к высокопроизводительным вычислительным ресурсам.	Зависит от конфигурации и требований пользователя.
Обучение и навыки	Требуется изучение синтаксиса, но относительно доступен.	Требует глубоких навыков в программировании и биоинформатике.

Почему важны пайплайны

- Экономия Времени и Ресурсов
- Стандартизация и Надежность
- Обновления и Поддержка
- Общие Протоколы и Процессы
- Доступность для Неопытных Пользователей
- Расширенные Возможности

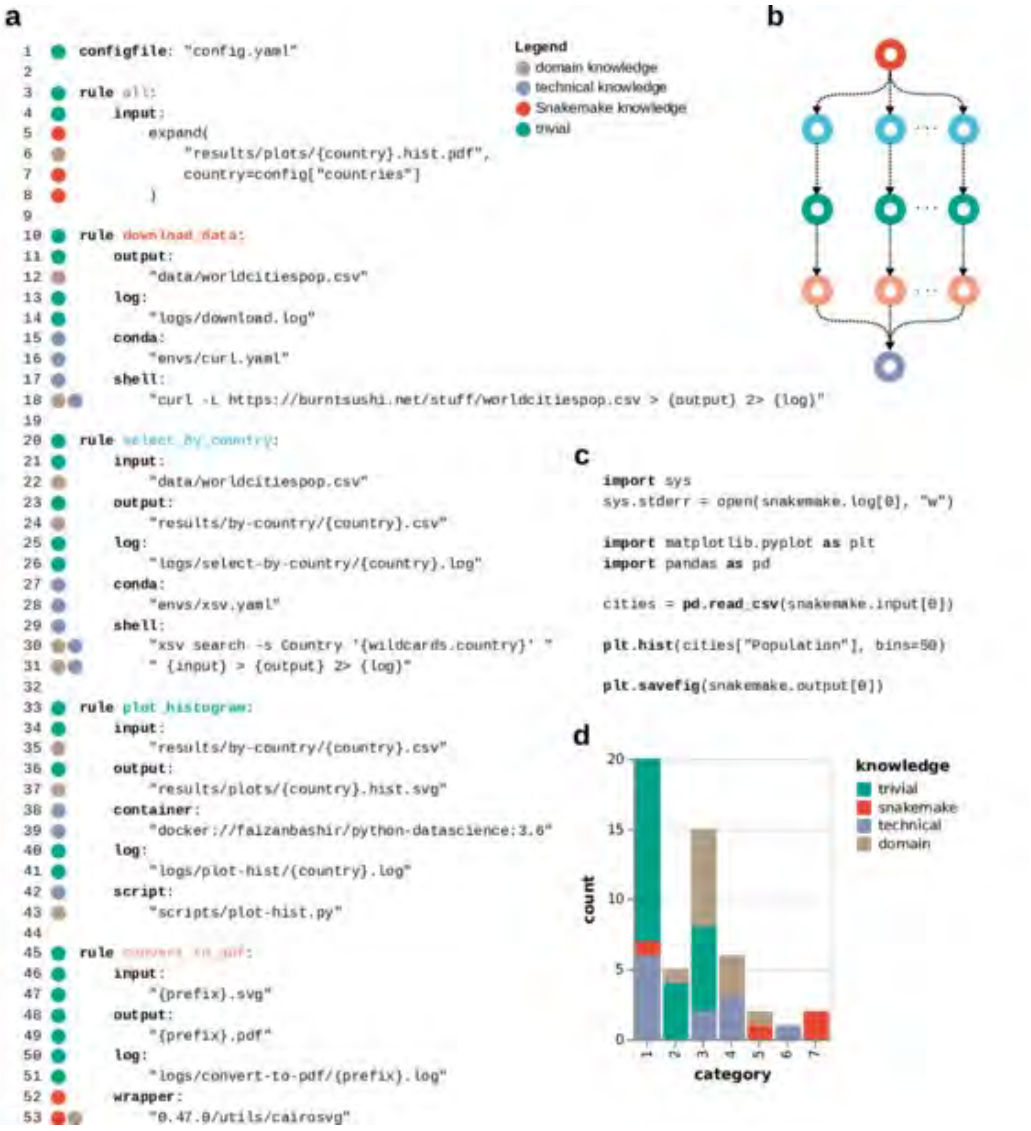
Вычислительные графы



Пайпланы в биоинформатике

Snakemake

Snakemake


a
b
c
d
Legend
domain knowledge
technical knowledge
Snakemake knowledge
trivial
configfile: "config.yaml"
rule all:
input:
expand(
"results/plots/{country}.hist.pdf",
country=config["countries"]
)
rule download_data:
output:
"data/worldcitiespop.csv"
log:
"logs/download.log"
conda:
"envs/curl.yaml"
shell:
"curl -L https://burntsushi.net/stuff/worldcitiespop.csv > {output} 2> {log}"
rule select_by_country:
input:
"data/worldcitiespop.csv"
output:
"results/by-country/{country}.csv"
log:
"logs/select-by-country/{country}.log"
conda:
"envs/xsv.yaml"
shell:
"xsv search -s Country '{wildcards.country}' "
" {input} > {output} 2> {log}"
rule plot_histogram:
input:
"results/by-country/{country}.csv"
output:
"results/plots/{country}.hist.svg"
container:
"docker://faizanbashir/python-datascience:3.6"
log:
"logs/plot-hist/{country}.log"
script:
"scripts/plot-hist.py"
rule convert_to_pdf:
input:
"{prefix}.svg"
output:
"{prefix}.pdf"
log:
"logs/convert-to-pdf/{prefix}.log"
wrapper:
"0.47.0/utils/cairosvg"
import sys
sys.stderr = open(snakemake.log[0], "w")
import matplotlib.pyplot as plt
import pandas as pd
cities = pd.read_csv(snakemake.input[0])
plt.hist(cities["Population"], bins=50)
plt.savefig(snakemake.output[0])
knowledge
trivial
snakemake
technical
domain
count
category
0
5
10
15
20
1
2
3
4
5
6
7

Snakemake workflow catalog

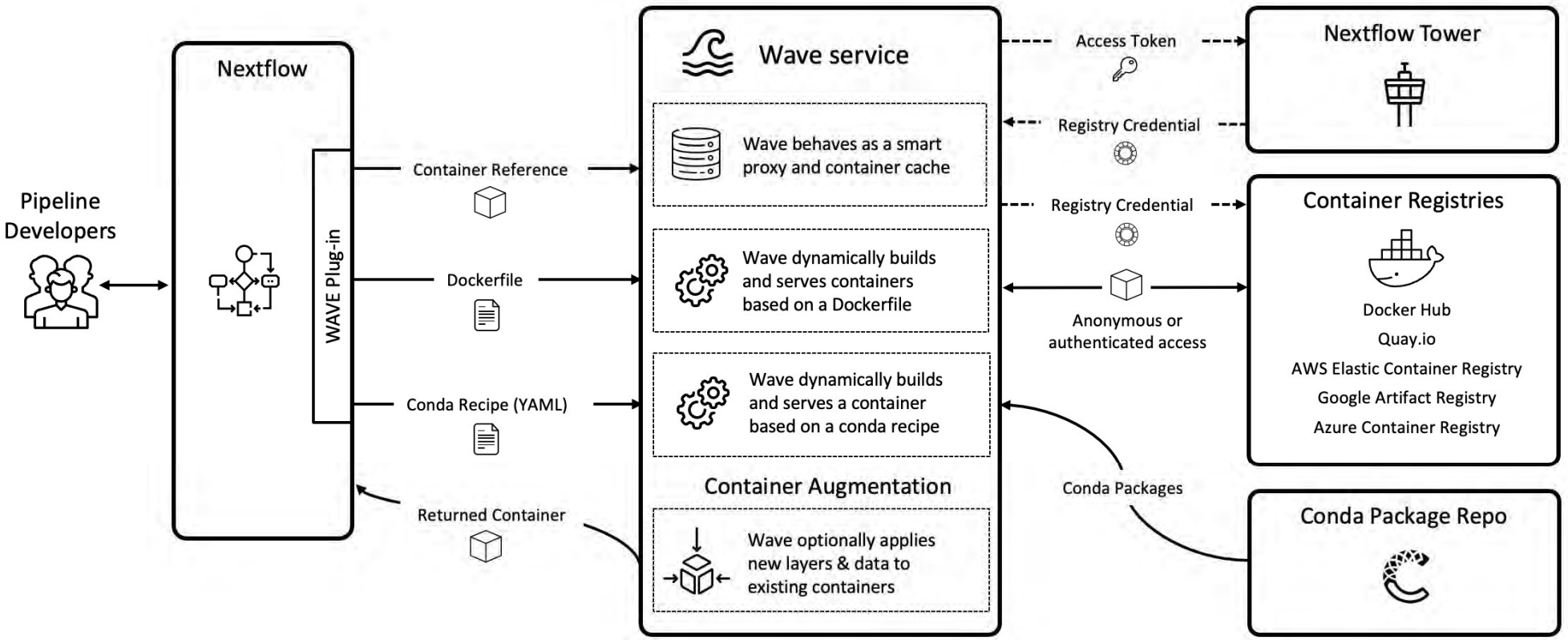
A comprehensive catalog of standards compliant, public, Snakemake workflows

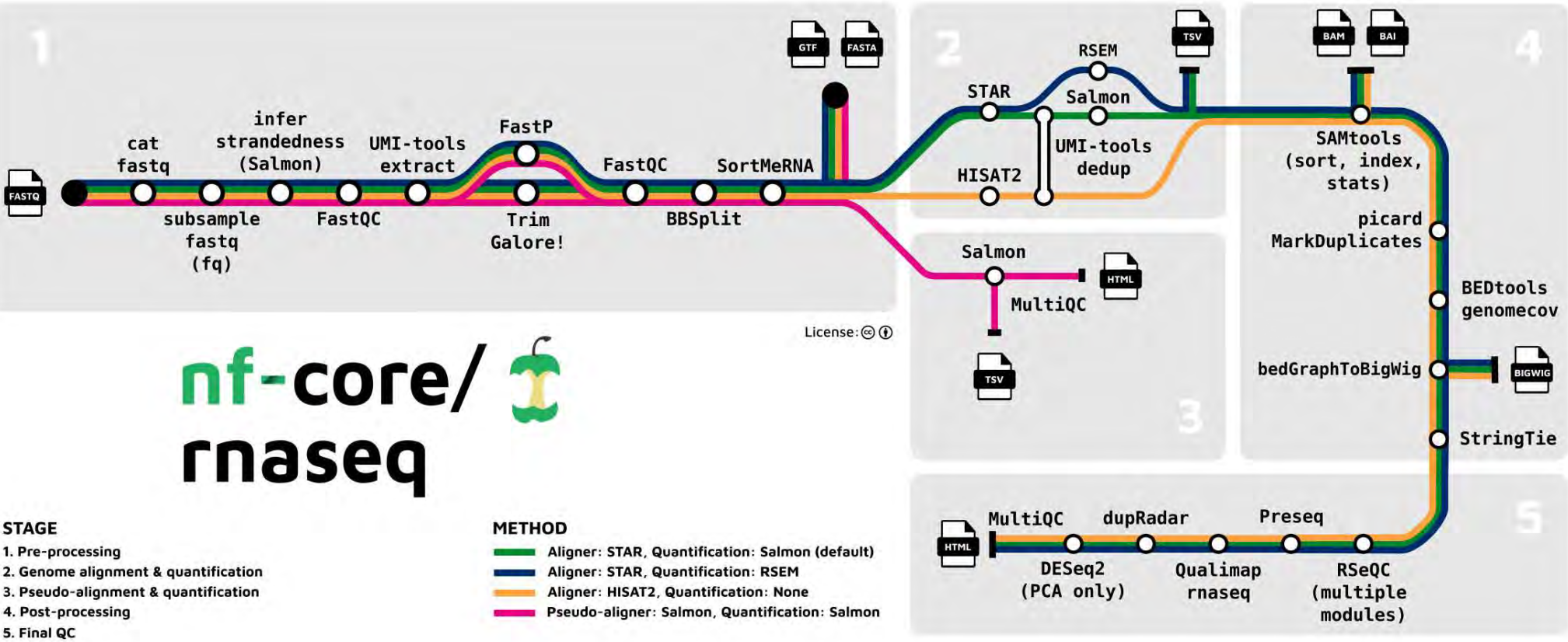
Standardized usage 196 | All workflows 2561

Search

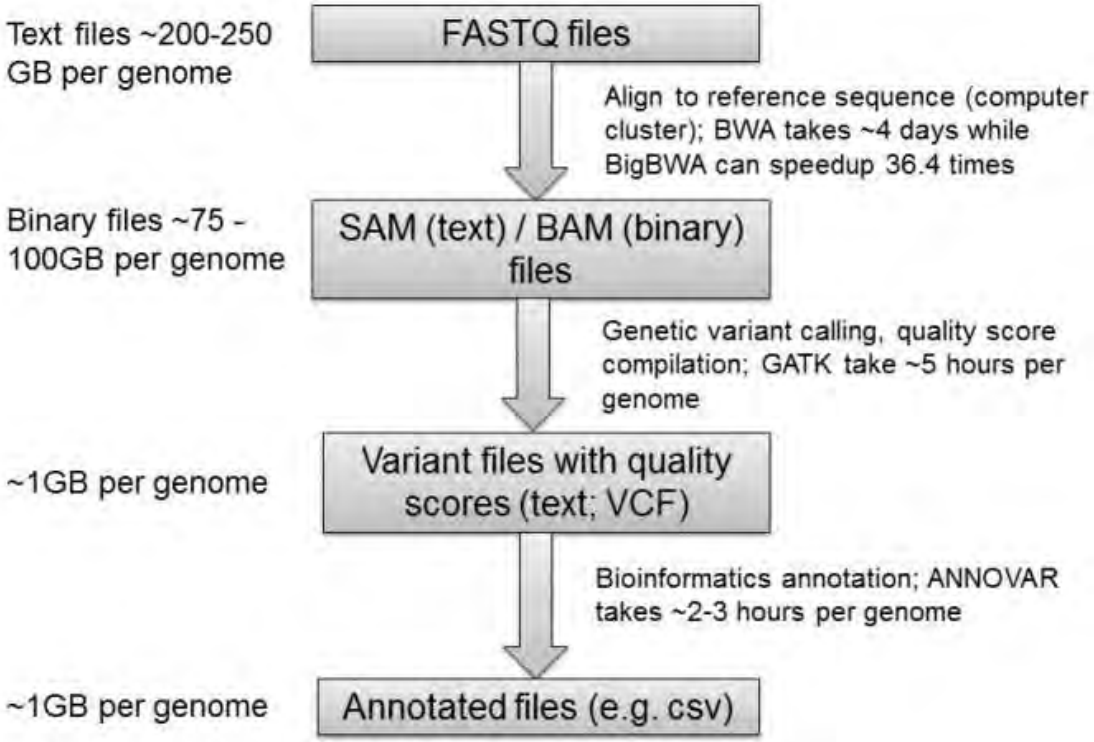
	Workflow	Description	Topics	QC	Stars	Watchers
Usage	snakemake-workflows/rna-seq-star-deseq2	RNA-seq workflow using STAR and DESeq2	snakemake, sciworkflows, reproducibility, gene-expression-analysis, deseq2	license MIT; last commit november; linting passed; formatting passed	282	11
Usage	snakemake-workflows/dna-seq-gatk-variant-calling	This Snakemake pipeline implements the GATK best-practices workflow	reproducibility, snakemake, sciworkflows, genomic-variant-calling, gatk	license MIT; last commit may 2021; linting passed; formatting failed	207	9

Nextflow


Pipeline Developers
Nextflow
WAVE Plug-in
Container Reference
Dockerfile
Conda Recipe (YAML)
Returned Container
Wave service
Wave behaves as a smart proxy and container cache
Wave dynamically builds and serves containers based on a Dockerfile
Wave dynamically builds and serves a container based on a conda recipe
Container Augmentation
Wave optionally applies new layers & data to existing containers
Access Token
Registry Credential
Registry Credential
Anonymous or authenticated access
Conda Packages
Nextflow Tower
Container Registries
Docker Hub
Quay.io
AWS Elastic Container Registry
Google Artifact Registry
Azure Container Registry
Conda Package Repo


1
2
3
4
5
FASTQ
cat fastq
infer strandedness (Salmon)
UMI-tools extract
FastP
FastQC
SortMeRNA
GTF
FASTA
subsample fastq (fq)
FastQC
Trim Galore!
BBSplit
STAR
RSEM
Salmon
HISAT2
UMI-tools dedup
TSV
BAM
BAI
SAMtools (sort, index, stats)
picard MarkDuplicates
BEDtools genomecov
bedGraphToBigWig
BIGWIG
StringTie
Salmon
MultiQC
HTML
TSV
License:
nf-core/rnaseq
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
MultiQC
dupRadar
Preseq
DESeq2 (PCA only)
Qualimap rnaseq
RSeQC (multiple modules)

Семинар


Text files ~200-250 GB per genome
FASTQ files
Align to reference sequence (computer cluster); BWA takes ~4 days while BigBWA can speedup 36.4 times
Binary files ~75 - 100GB per genome
SAM (text) / BAM (binary) files
Genetic variant calling, quality score compilation; GATK take ~5 hours per genome
~1GB per genome
Variant files with quality scores (text; VCF)
Bioinformatics annotation; ANNOVAR takes ~2-3 hours per genome
~1GB per genome
Annotated files (e.g. csv)