Automated pipelines for low-coverage whole genome sequencing

Nicolas Lou, Oct 2025

Intended learning outcomes

- By the end of this session, you will be able to
  - appreciate the power of workflow management tools in bioinformatics
  - have a basic understanding of Snakemake
  - be aware of several published pipelines for high-throughput sequencing data and lcWGS
  - test run a lcWGS pipeline with a toy dataset

What are automated bioinformatic pipelines and why using them?

What are some common frustrations of manual workflows?

- Constant monitoring and intervention
- Prone to errors
- Lack of reproducibility
- Headaches in software installation, conflicts, and versioning
- Inefficient parallelization
- Complicated interface with job schedulers

key Snakemake concepts

- rule
- input
- output
- wildcard
- snakefile
- config file

```
SAMPLES = ["A", "B"]


rule all:
    input:
        "plots/quals.svg"


rule bwa_map:
    input:
        "data/genome.fa",
        "data/samples/{sample}.fastq"
    output:
        "mapped_reads/{sample}.bam"
    shell:
        "bwa mem {input} | samtools view -Sb - > {output}"


rule samtools_sort:
    input:
        "mapped_reads/{sample}.bam"
    output:
        "sorted_reads/{sample}.bam"
    shell:
        "samtools sort -T sorted_reads/{wildcards.sample} "
        "-O bam {input} > {output}"


rule samtools_index:
    input:
        "sorted_reads/{sample}.bam"
    output:
        "sorted_reads/{sample}.bam.bai"
    shell:
        "samtools index {input}"


rule bcftools_call:
    input:
        fa="data/genome.fa",
        bam=expand("sorted_reads/{sample}.bam", sample=SAMPLES),
        bai=expand("sorted_reads/{sample}.bam.bai", sample=SAMPLES)
    output:
        "calls/all.vcf"
    shell:
        "bcftools mpileup -f {input.fa} {input.bam} | "
        "bcftools call -mv - > {output}"


rule plot_quals:
    input:
        "calls/all.vcf"
    output:
        "plots/quals.svg"
    script:
        "scripts/plot-quals.py"
```

From fastq to bam

snakemake 8.15.2 | CI passing | platform linux-64 | osx-64 | release v0.15.0 | license GPLv3 | doi 10.1093/bioinformatics/btac600

Snakemake workflow for variant calling from raw sample sequences, with lots of bells and whistles - for sampled individuals, and for pool sequencing.

Advantages:

- One command to run the whole pipeline!
- Many tools to choose from for each step
- Simple configuration via a single file
- Automatic download of tool dependencies
- Resuming from failing jobs

From fastq to bam

Process and available tools:

- Read trimming (single or paired end)
  - AdapterRemoval
  - Cutadapt
  - fastp
  - SeqPrep
  - skewer
  - trimmomatic
- Read mapping
  - bwa mem
  - bwa aln
  - bwa mem2
  - Bowtie2
- Optional read filtering, clipping, duplication removal, and quality score recalibration
  - samtool view
  - BamUtil clipOverlap
  - Picard MarkDuplicates
  - DeDup
  - GATK BaseRecalibrator (BQSR)
  - samtools mpileup
- Damage profiling (optional; e.g., for ancient DNA)
  - mapDamage
  - DamageProfiler
- Variant calling and genotyping
  - GATK HaplotypeCaller / GATK GenotypeGVCFs
  - freebayes
  - bcftools call
- Variant filtering
  - GATK VariantFiltration
  - GATK VariantRecalibrator (VQSR)
  - bcftools filter
- Frequency calling (for pool sequencing data, as an alternative to variant calling)
  - HAF-pipe
- Quality control, statistics, SNP annotation, reporting
  - FastQC
  - samtool stats
  - samtool flagstat
  - QualiMap
  - Picard CollectMultipleMetrics
  - bcftools stats
  - snpEff
  - VEP (Ensembl Variant Effect Predictor)
  - MultiQC

A simple config file and a single command

- https://github.com/moiexpositoalonsolab/grenepipe/blob/master/config/config.yaml
- Then run
  - `snakemake [other options] all_qc`

From bam to population genetic inference

loco-pipe: a Snakemake pipeline for low-coverage whole-genome sequencing

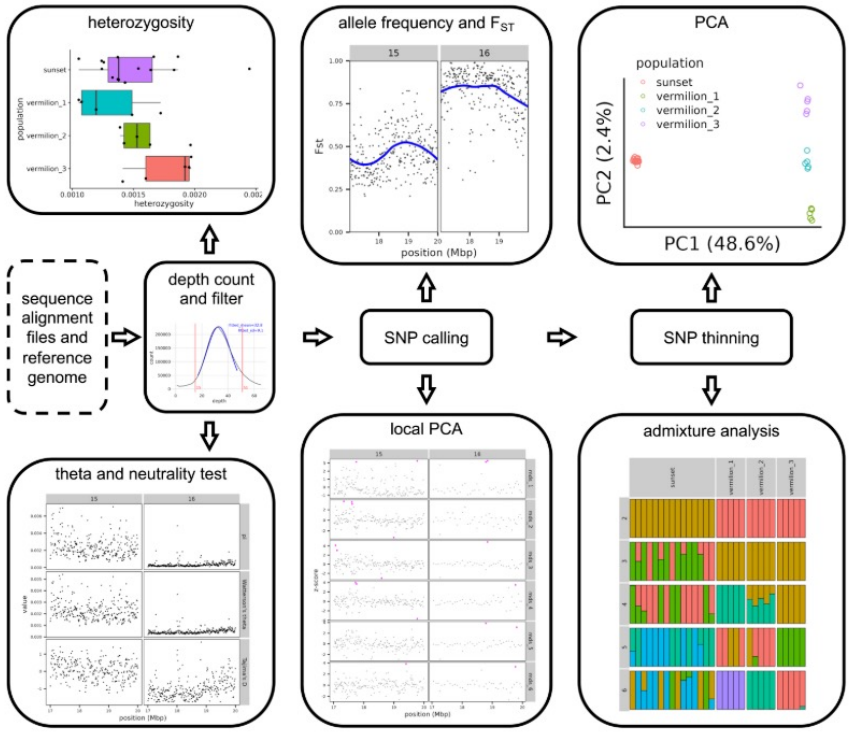
loco-pipe is an automated Snakemake pipeline that streamlines a set of essential population genomic analyses for **lo**w-**co**verage whole genome sequencing (lcWGS) data.

Key features

- Streamlining of several essential population genomic analyses
- Can be launched with a single line of code
- Incorporation of key filtering steps and best practices for low-coverage data
- Key results are plotted automatically for visual inspection
- Easy customization through a configuration file
- A quick start guide with an example dataset
- Extensive in-line annotation along with a detailed user's manual
- Flexible architecture that allows for the addition of new features
- Inheritance of the many benefits offered by Snakemake, including
  - High computational efficiency achieved through massive parallelization
  - Seamless integration with common job schedulers on computer clusters
  - The ability to automatically continue from the last failed or interrupted job
  - Built-in software management system and robust file structure

Currently supported functionalities

- Depth counting
- SNP calling
- Allele frequency estimation
- Site frequency spectrum (SFS) estimation
- Fixation index (Fst) estimation
- Principal component analysis (PCA)
- Admixture analysis
- Theta estimation
- Neutrality test statistics
- Heterozygosity estimation
- Local PCA analysis



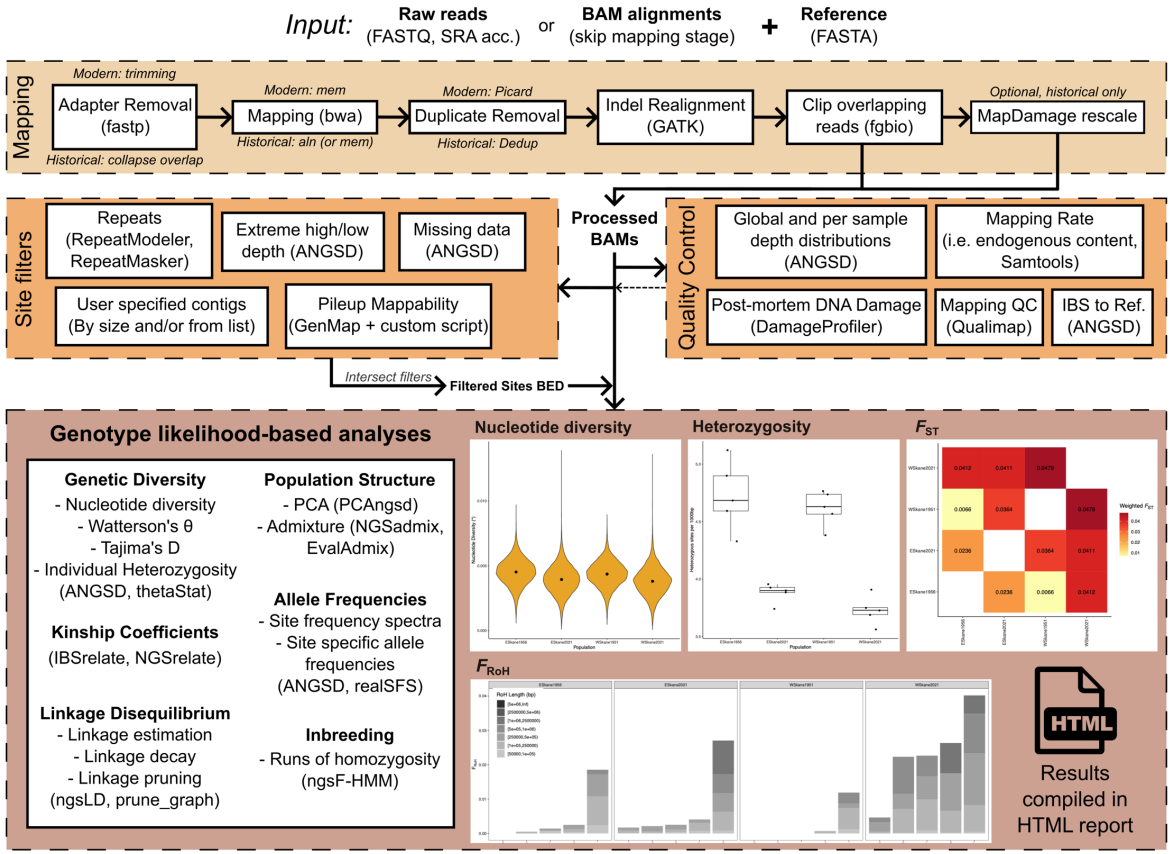
From bam to population genetic inference

Welcome to the documentation for PopGLen

PopGLen is aimed at enabling users to run population genomic analyses on their data within a genotype likelihood framework in an automated and reproducible fashion. Genotype likelihood based analyses avoid genotype calling, instead performing analyses on the likelihoods of each possible genotype, incorporating uncertainty about the true genotype into the analysis. This makes them especially suited for datasets with low coverage or that vary in coverage.

This pipeline was developed in initially to make my own analyses easier. I work with many species being mapped to their own references within the same project. I developed this pipeline so that I could ensure standardized processing for datasets within the same project and to automate the many steps that go into performing these analyses. As it needed to fit many datasets, it is generalizable and customizable through a single configuration file, uses a common workflow utilized by ANGSD users, and provides reporting features to quickly assess sample quality, filtering, and results. As such, it is applicable to a wide variety of studies that utilize population genomics on whole genome data.

This webpage contains documentation so others can utilize the features of this pipeline, including a short getting started page, a detailed configuration summary, a tutorial to walk users through running all analyses in the pipeline, as well as more specialized documentation describing how to configure the pipeline to run on a HPC cluster, how to modify the requested resources for jobs, how to extend the workflow with additional rules, and the file paths for all major output files for reference.



Setting up loco-pipe

- a contiguous reference genome & bam files
- install conda
- clone loco-pipe repo from GitHub
- create loco-pipe and lostruct conda environments
- set up file structure according to instructions
- prepare a sample table and a chromosome table
- edit the configuration file
- (Optional) edit the cluster configuration file

loco-pipe config file

- https://github.com/sudmantlab/loco-pipe/blob/main/config.yaml

Now let’s do some live coding

- Set up loco-pipe on AWS
- Do a dry run using the built-in test data
- Modify the config file and check what changes