NOV 07, 2022

SHARE

WORKS FOR ME 1

Phylogenetic classification of bacterial populations using a combined analysis of core SNPs and the presence/absence of accessory genes

DOI
dx.doi.org/10.17504/protocols.io.x54v9y13mg3e/v1

Jared Johnson[1], Chris Curtin[1], Joy Waite-Cusic[1]

[1]Oregon State University

Jared D Johnson

COMMENTS 0

ABSTRACT

Whole genome sequencing-based typing methods have become the gold standard for phylogenetic classifications of bacterial populations. The most common approach is to compare single nucleotide polymorphisms (SNPs) across the core genome of a cohort of isolates. Greater confidence in SNP-based population assignments can be gained by extending this analysis to include the accessory genome. This is often accomplished by comparing allelic differences between isolates using whole genome multi-locus sequence typing (wgMLST). Unfortunately, allele-based methods, like wgMLST, suffer from being highly species-specific, therefore limiting their current application to only commonly studied bacteria. This work aimed to address this limitation by developing an alternative population classification method based on the accessory genome that could be more universally applied, regardless of the bacterial species being studied. This was accomplished by comparing population assignments based on core SNPs to those derived from the phylogenetic analysis of gene presence/absence matrices output by common pangenome assembly software (i.e., Panaroo or Roary).

DOI

dx.doi.org/10.17504/protocols.io.x54v9y13mg3e/v1

EXTERNAL LINK

https://github.com/johnjare/panaroo_scripts

PROTOCOL CITATION

Jared Johnson, Chris Curtin, Joy Waite-Cusic 2022. Phylogenetic classification of bacterial populations using a combined analysis of core SNPs and the presence/absence of accessory genes. **protocols.io**
https://dx.doi.org/10.17504/protocols.io.x54v9y13mg3e/v1

FUNDERS ACKNOWLEDGEMENT

Western Dairy Center BUILD Dairy Program

KEYWORDS

phylogenetics, bacterial populations, pangenome, accessory genome, SNPs

CREATED

Apr 24, 2022

LAST MODIFIED

Nov 07, 2022

PROTOCOL INTEGER ID

61322

BEFORE STARTING

This protocol assumes you have the following software installed:

Required Software:

- Unicycler (https://github.com/rrwick/Unicycler)
- Prokka (https://github.com/tseemann/prokka)
- Panaroo (https://github.com/gtonkinhill/panaroo)
- SNP-sites (https://github.com/sanger-pathogens/snp-sites)
- RAxML (https://cme.h-its.org/exelixis/web/software/raxml/)

Optional Software:

- BUSCO (https://busco.ezlab.org)
- Quast (http://quast.sourceforge.net)

Downloading required scripts

1 Download scripts from https://github.com/johnjare/panaroo_scripts.

```
git clone https://github.com/johnjare/panaroo_scripts
```

Genome assembly and quality check:

2 Assemble bacterial genomes using Unicycler (https://github.com/rrwick/Unicycler). The standard input for Illumina-only assemblies is shown.

```
unicycler -1 short_reads_1.fastq.gz -2 short_reads_2.fastq.gz -o output_dir
```

3 Check the quality of the assembled genomes and any reference genomes included in the analysis using BUSCO (https://busco.ezlab.org) and Quast (http://quast.sourceforge.net). Standard inputs are shown. This step is optional.

```
busco -i assembly.fasta -o output_dir -m genome -l reference_id
```

```
quast.py -i assembly_directory/*.fasta -o output_directory -r
reference.fasta
```

4 Reject assemblies with large numbers of missing, fragmented, or duplicated (MFD) BUSCOs (e.g., assemblies with >1% MFD BUSCOs) and/or those with estimated genome sizes three standard deviations from the mean of high-quality reference genomes for that species.

Annotate assemblies

5 Annotate all genome assemblies, including reference genomes if included in your analysis, using Prokka (https://github.com/tseemann/prokka). The standard input is shown below.

```
prokka --kingdom Bacteria --outdir output_directory --genus bacterial_genus
--locustag isolate_id assembly.fasta
```

6 Move the resulting .gff annotation files to a single directory.

Pangenome assembly and quality check

7 Assemble the pangenome using Panaroo (https://github.com/gtonkinhill/panaroo). The standard input with core gene alignment using MAFFT is shown.

```
panaroo -i gff_directory/*.gff -o output_directory -a core --aligner mafft
```

8 Check for any problematic genome assemblies in the pangenome using pan_check.R (https://github.com/johnjare/panaroo_scripts/tree/main/pan_check). This script will output a heatmap like shown in Figure 1. Notice that isolate_10 in Figure 1 is missing several genes shared by all other isolates, probably indicating poor assembly quality.

```
Rscript pan_check.R gene_presence_absence.Rtab
```

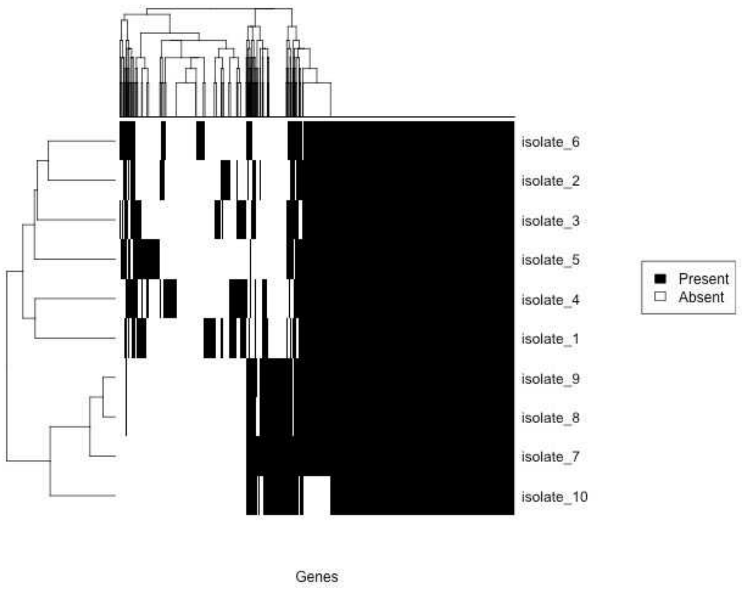

Figure 1. Example output from the 'pan_check.R' script.

9 If necessary, re-assemble the pangenome after removing any problematic genome assemblies identified in the previous step.

Extract SNPs and accessory genes

10 Extract single nucleotide polymorphisms (SNPs) from the core gene alignment using SNP-sites (https://github.com/sanger-pathogens/snp-sites). The standard inputs for the creation of an alignment (.aln) file and variant call format (.vcf) file are shown. The "-c" flag excludes gaps.

```
snp-sites -c core_gene_alignment.aln > core_snps.aln
```

```
snp-sites -c -v core_gene_alignment.aln > core_snps.vcf
```

11 Extract accessory genes using the accessory_binary_genes.py (https://github.com/johnjare/panaroo_scripts/tree/main/accessory_binary_genes). This script requires that Pandas is installed (https://pandas.pydata.org). The standard input for this command is shown. The output has a similar structure to a FASTA file but contains 1's and 0's in places of ACGT's.

```
python3 accessory_binary_genes.py -i gene_presence_absence.Rtab -o
accessory_genes.binary
```

Create phylogenetic trees

12 Create phylogenetic trees from the "accessory_genes.binary" and "core_snps.aln" files generated in the previous step using RAXML. The standard input using the BINCAT and GTRCAT models are shown. Adjust as necessary based on your available resources, preferred seed numbers, and number of bootstraps. The command for rooting the tree is also shown, though this step is optional.

12.1 **Accessory genome:**

```
raxml -s accessory_genes.bin -n output_file_name -m BINCAT -f a -T 2 -x 11
```

```
-N 100 -p 11
```

```
raxml -T 3 -f I -m GTRCAT -t input_file -n output_file_name
```

12.2 **SNPs:**

```
raxml -s core_snps.aln -n output_file_name -m GTRCAT -f a -T 2 -x 11 -N
100 -p 11
```

```
raxml -T 3 -f I -m GTRCAT -t input_file -n output_file_name
```

Calculating pairwise SNP and gene differences

13 Calculate pairwise gene and SNP differences using the pair_diff.py (https://github.com/johnjare/panaroo_scripts/tree/main/pair_diff). The standard input is shown.

```
python3 pair_diff.py -g gene_presence_absence.Rtab -s core_snps.vcf -o
gene_snp_differences.csv
```