Lee Katz

lyve-SET
========

LYVE version of the Snp Extraction Tool (SET), a method of using hqSNPs to create a phylogeny. NML, part of PHAC, has the original version of SET (https://github.com/apetkau). However, I have been updating it since inception and so the results might differ slightly.

SET is meant to be run on a cluster but will just run system calls if qsub is not present.

Requirements
----

* Perl, multithreaded
* RAxML
* PhyML
* FreeBayes
* Smalt
* CG-Pipeline
* Samtools
* Schedule::SGELK (installed at the same time if you download with git)

Installation

```
    $ mkdir ~/bin
    $ cd ~/bin
    $ git clone --recursive https://github.com/lskatz/lyve-SET.git
    $ export PATH=$PATH:~/bin/lyve-SET # you might also put this line into your .bash_profile or other login script
```

TO UPDATE, in case any updates or fixes are made, run this command at any time from the base lyve-SET directory.

```
    $ git pull -u --recurse-submodules=yes
```

Usage

```
    Usage: launch_set.pl -ref reference.fasta [-b bam/ -v vcf/ -t tmp/ -reads reads/ -m msa/]
      Where parameters with a / are directories
      -r where fastq and fastq.gz files are located
      -b where to put bams
      -v where to put vcfs
      --msadir multiple sequence alignment and tree files (final output)
      -numcpus number of cpus
      -numnodes maximum number of nodes
      -w working directory where qsub commands can be stored.
```

Default: CWD

-a allowed flanking distance in bp. Nucleotides this close together cannot be considered as high-quality.

--nomsa to not make a multiple sequence alignment

--notrees to not make phylogenies

-q '-q long.q' extra options to pass to qsub. This is not sanitized.

--noclean to not clean reads before mapping (faster, but you need to have clean reads to start with)

Lyve-SET is modular and so the individual scripts can be run too. For example, you can run launch_smalt.pl or launch_snap.pl to run mapping alone; however, indexing the reference fasta takes place in launch_set.pl. To get usage help on any of these scripts, run the script with no options.

Citing lyve-SET

To cite lyve-SET, please reference this site and cite the Haiti Anniversary paper. Lyve-SET also makes use of the tools shown above in the prerequisites. If you feel like your study relied heavily on any of those tools, please don't forget to cite them!

https://github.com/lskatz/lyve-SET

Katz LS, Petkau A, Beaulaurier J, Tyler S, Antonova ES, Turnsek MA, Guo Y, Wang S, Paxinos EE, Orata F, Gladney LM, Stroika S, Folster JP, Rowe L, Freeman MM, Knox N, Frace M, Boncy J, Graham M, Hammer BK, Boucher Y, Bashir A, Hanage WP, Van Domselaar G, Tarr CL. 2013. Evolutionary dynamics of Vibrio cholerae O1 following a single-source introduction to Haiti. mBio 4(4):e00398-13. doi:10.1128/mBio.00398-13.

Aaron Petkau.

GitHub This repository Explore Features Enterprise Blog Sign up Sign in

apetkau / core-phylogenomics

★ Star 0 Fork 0

Pipeline for identifying core SNPs and building a phylogenetic tree

250 commits 2 branches 0 releases 2 contributors

Code
Issues 2
Pull Requests 0
Wiki
Pulse
Graphs
Network

HTTPS clone URL
https://github.com/

You can clone with HTTPS or Subversion.

Clone in Desktop
Download ZIP

master core-phylogenomics / +

Updated vcf2pseudoalign

apetkau authored a month ago latest commit ebf23821b2

bin	Changed header for nml_translate.pl	a year ago
etc	Added a bit more install documentation	5 months ago
lib	Updated vcf2pseudoalign	a month ago
perl_bin	Changed link for more documentation	4 months ago
t	Printing out the temporary output directory	a month ago
.gitignore	added gitignore so we can ignore temp file from vim and emacs	7 months ago
.gitmodules	Updating locations of submodule and other links	5 months ago
INSTALL.md	Changed shell from sh to bash	5 months ago
README.md	Updating locations of submodule and other links	5 months ago

README.md

Core SNP Phylogenomics

The Core SNP phylogenomics pipeline provides a pipeline for identifying high-quality core SNPs among a set of bacterial isolates and generating phylogenetic trees from these SNPs. The pipeline takes as input a reference genome (in FASTA format) and a set of DNA sequencing reads (in FASTQ format) and proceeds through a number of different stages to find core SNPs.

Authors

The Core SNP Pipeline was developed by the following individuals: Aaron Petkau, Gary Van Domselaar, Philip Mabon, and Lee Katz.

Quick Start

Command

If you have a set of DNA sequencing reads, **fastq_reads/*.fastq**, and a reference file containing the DNA sequence of the genome to use for reference mapping, **reference.fasta**, then the following command can be used to generate a core SNP phylogeny.

```
snp_phylogenomics_control --mode mapping --input-dir fastq_reads/ --output pipeline_out --reference reference
```

Output

Once the pipeline is finished main output files you will want to look at include:

- **pipeline_out/pseudoalign/pseudoalign-positions.tsv**: A tab-separated file containing all variants identified and the positions of each variant on the reference genome.

Example:

```
#Chromosome Position    Status  Reference   isolate1    isolate2
contig1 20  valid   A   A   T
contig2 30  filtered-coverage   A   -   A
```

- **pipeline_out/pseudoalign/matrix.csv**: A tab-separated file containing a matrix of high-quality SNP distances between isolates.

Example:

```
strain  isolate1    isolate2
isolate1    0   5
isolate2    5   0
```

- **pipeline_out/pseudoalign/pseudoalign.phy**: An alignment of variants for each input isolate in phylip format.
- **pipeline_out/phylogeny/pseudoalign.phy_phyml_tree.txt**: A phylogenetic tree of the above alignment file generated using PhyML.

Alternative Commands

Alternatively, in addition to the above files, if you have a set of assembled contigs from isolates, **fasta_contigs/*.fasta**, that you wish to include in the analysis, you can run the pipeline with:

```
snp_phylogenomics_control --mode mapping --input-dir fastq_reads/ --contig-dir fasta_contigs/  --output pipel
```

In addition, if you have a pre-defined set of positions on the reference genome you wish to exclude (repetitive regions, etc) in a tab-separated values file format (see detailed documentation below for a description of the file format) you can run the pipeline with the command:

```
snp_phylogenomics_control --mode mapping --input-dir fastq_reads/ --contig-dir fasta_contigs/  --invalid-pos
```

bad_positions.tsv:

```
#Contig start   end
contig1 50  100
contig2 75  100
```

Stages

The core SNP pipeline proceeds through the following stages:

1. Reference mapping using SMALT.
2. Variant calling using FreeBayes.
   i. For any assembled contigs passed to the pipeline, generates variant call files (VCF) using MUMMer alignments.
3. Checking variant calls and depth of coverage using SAMTools.
4. Aligning high-quality SNPs into a meta-alignment (pseudoalignment) of phylogenetically informative sites.
   i. If an invalid positions file is passed, remove any SNPs within the invalid positions.

5. Building a phylogenetic tree with PhyML.

Tutorial

For a step-by-step tutorial on how to run the core SNP pipeline with some example data, please see https://github.com/apetkau/core-phylogenomics-tutorial.

Installation

Please refer to the Installation document.

Details

The core SNP pipeline has a number of different modes of operation in addition to building core SNP phylogenies from reference mapping and variant calling. These modes of operation are controlled by the **--mode** parameter. A list of these different modes of operation is given below.

- **prepare-fastq**: Can be used to remove low-quality reads from FASTQ files and reduce the data size. This can be used for a basic quality check of reads before performing reference mapping and variant calling.
- **mapping**: Builds a core SNP phylogeny using reference mapping and variant calling.
- **orthomcl**: Builds a core orthologous gene SNP phylogeny by multiply aligning orthologs identified using OrthoMCL and extracting phylogenetically informative sites.
- **blast**: Builds a core orthologous gene SNP phylogeny by multiply aligning orthologs identified using a single-directional BLAST and extracting phylogenetically informative sites.

In addition to the above modes, data analysis can be re-submitted from any stage using the **--resubmit** parameter.

Mode: Prepare-FASTQ

This mode can be used to do some basic quality checks on FASTQ files before running through the reference mapping pipeline. The stages that are run are as follows:

1. Removes poor quality reads and trims ends of reads.
2. Randomly samples reads from FASTQ files to reduce the dataset size to a user-configurable maximum coverage (estimated based on the reference genome length).
3. Runs FastQC on the quality-filtered and reduced reads dataset.
4. Generates a report for each of the isolates.

Command

To run this mode, the following command can be used.

```
snp_phylogenomics_control --mode prepare-fastq --input-dir fastq/ --output cleaned_out --reference reference.
```

The main output you will want to use includes:

- **cleaned_out/fastqc/fastqc_stats.csv**: A tab-deliminated report from FastQC on each isolate.
- **cleaned_out/downsampled_fastq**: A directory containing all the cleaned and reduced FASTQ files. This directory can be used as input to the mapping mode of the pipeline.

Input

The following is a list of input files and formats for the **prepare-fastq** mode of the pipeline.

- **--input-dir fastq_reads/**: A directory containing FASTQ-formatted DNA sequence reads. Only one file per isolate (paired-end mapping not supported).

Example:

```
fastq_reads/
    isolate1.fastq
    isolate2.fastq
    isolate3.fastq
```

- **--reference reference.fasta**: A reference FASTA file. This is used to estimate the coverage of each isolate for reducing the amount of data in each FASTQ file.
- **--config options.conf**: A configuration file in YAML format. This is used to defined specific parameters for each stage of the *prepare-fastq* mode.

Example:

```
%YAML 1.1
---
max_coverage: 200
trim_clean_params: '--numcpus 4 --min_quality 20 --bases_to_trim 10 --min_avg_quality 25 --min_length 36 -p 1
drmaa_params:
    general: "-V"
    trimClean: "-pe smp 4"
```

Output

- **--output cleaned_out**: Defines the output directory to store the files for each stage.

The output directory structure looks as follows:

```
cleaned_out/
    downsampled_fastq/
    fastqc/
    initial_fastq_dir/
    log/
    reference/
    run.properties
    stages/
```

A description of each of the directories and files are:

- **downsampled_fastq/**: A directory containing the quality-filtered and data reduced fastq files.
- **fastqc/**: A directory containing any of the FastQC results.
- **initial_fastq_dir/**: A directory containing links to the initial input fastq files.
- **log/**: A directory containing log files for each of the stages.
- **reference/**: A directory containing the input reference file.
- **run.properties**: A file containing all the parameters used to quality-filter the fastq files.
- **stages/**: A directory containing files used to defined which stages of the *prepare-fastq* mode have been completed.

Mode: Mapping

Input

The following is a list of input files and formats for the pipeline.

- **--reference reference.fasta**: A FASTA file containing the genome to be used for reference mapping.

Example:

```
>contig1
ATCGATCGATCGATCG
ATCGATCGATCGATCG
```

- **--input-dir fastq_reads/**: A directory containing FASTQ-formatted DNA sequence reads. Only one file per isolate (paired-end mapping not supported). The file name is used as the name in the final phylogenetic tree.

Example:

```
fastq_reads/
    isolate1.fastq
    isolate2.fastq
    isolate3.fastq
```

- **--contig-dir contig_fasta/**: A directory containing assembled contigs to include for analysis. Variants will be called using MUMMer. Only one file per isolate.

Example:

```
contig_fasta/
    isolate1.fasta
    isolate2.fasta
    isolate3.fasta
```

- **--invalid-pos bad_positions.tsv**: A tab-separated values file format containing a list of positions to exclude from the analysis. Any SNPs in these positions will be marked as 'invalid' in the variant table and will be excluded from the matrix of SNP distances and the alignment used to generate the phylogeny. The contig IDs used in this file must correspond to the IDs used in the reference FASTA file.

Example:

```
#ContigID   Start   End
contig1 1   500
contig2 50  100
```

- **--config options.conf**: A configuration file which can be used to override the default parameters for the different stages. This file must be in YAML format.

Example:

```
%YAML 1.1
---
min_coverage: 15
freebayes_params: '--pvar 0 --ploidy 1 --left-align-indels --min-mapping-quality 30 --min-base-quality 30 --m
smalt_index: '-k 13 -s 6'
smalt_map: '-n 24 -f samsoft -r -1'
vcf2pseudo_numcpus: 4
vcf2core_numcpus: 24
trim_clean_params: '--numcpus 4 --min_quality 20 --bases_to_trim 10 --min_avg_quality 25 --min_length 36 -p 1
drmaa_params:
    general: "-V"
    vcf2pseudoalign: "-pe smp 4"
    vcf2core: "-pe smp 24"
    trimClean: "-pe smp 4"
```

Output

The detailed output directory tree looks as follows:

```
pipeline_out/
```

```
    fastq/
    invalid/
    log/
    mapping/
    mpileup/
    phylogeny/
        pseudoalign.phy_phyml_stats.txt
        pseudoalign.phy_phyml_tree.txt
        pseudoalign.phy_phyml_tree.txt.pdf
    pseudoalign/
        matrix.csv
        pseudoalign.fasta
        pseudoalign.phy
        pseudoalign-positions.tsv
    reference/
    run.properties
    sam/
    stages/
    vcf/
    vcf2core/
        contig1.gff
        contig1.png
    vcf-split/
```

The description of each of these directories/files are as follows:

- **fastq/**: A directory containing links to each of the input fastq files.
- **invalid/**: A directory containing the invalid positions file used if it was passed to the pipeline.
- **log/**: Log files for every stage of the pipeline.
- **mapping/**: Files for each isolate containing the SMALT reference-mapping information.
- **mpileup/**: Files generated from 'samtools mpileup' for each isolate.
- **phylogeny/**: Files generated from PhyML when building the phylogeny.
    - **pseudoalign.phy_phyml_stats.txt**: A statistics file generated by PhyML.
    - **pseudoalign.phy_phyml_tree.txt**: The phylogenetic tree generated by PhyML in Newick format.
    - **pseudoalign.phy_phyml_tree.txt.pdf**: A PDF of the phylogenetic tree, rendered using Figtree.
- **pseudoalign/**: Contains the "pseudoalignment" of only phylogenetically informative sites used to generate the phylogeny, as well as other information about each of the sites.
    - **matrix.csv**: A matrix of SNP distances between each isolate.
    - **pseudoalign.fasta**: An alignment of phylogenetically informative sites in FASTA format.
    - **pseudoalign.phy**: An alignment of phylogenetically informative sites, in phylip format.
    - **pseudoalign-positions.tsv**: A tab-separated values file containing a list of all positions identified by the pipeline.
- **reference/**: A directory containing links to the reference FASTA file used by some of the tools.
- **run.properties**: A properties file containing all the parameters used for the pipeline, in YAML format.
- **sam/**: SAM formated files generated by SMALT.
- **stages/**: A directory of files indicating which stages have been completed by the pipeline.
- **vcf/**: The VCF files produced by FreeBayes.
- **vcf2core/**: Files used to generate an image of the core genome.
    - **contig1.gff**: A GFF formatted file listing core genome locations on each contig.
    - **contig1.png**: An image showing the core genome locations for each contig rendered using GView.
- **vcf-split/**: VCF files split up so that one single SNP is represented by one line.

The **matrix.csv** file lists high-quality SNP distances between each combination of isolates. An example of this file is given below.

Example: *matrix.csv*

```
strain  isolate1    isolate2
isolate1    0   5
isolate2    5   0
```

The **pseudoalign-positions.tsv** file lists all SNPs found within the pipeline and the corresponding contig/position combination. The **status** column lists the status of each position. Only *valid* position statuses are used to generate the alignment files. The *filtered-coverage* status defines a position (indicated by a - character) which had insufficient coverage to be included as a core SNP. The *filtered-mpileup* status defines a position (indicated by an N) which had conflicting variant calls between FreeBayes and SAMTools mpileup. The *filtered-invalid* status indicates that this position was filtered out due to belonging to one of the invalid position regions passed to the pipeline.

Example: *pseudoalign-positions.tsv*

```
#Chromosome Position    Status  Reference   isolate1     isolate2
contig1 20  valid   A   A   T
contig2 5   filtered-coverage   A   -   A
contig2 35  filtered-mpileup    A   N   A
contig2 40  filtered-invalid    A   C   A
```

The **vcf2core/*.gff** files list the coordinates that were deteremend to be part of the core genome (based on the minimum coverage).

Example: *contig1.gff*

```
task_2  .   region  10  100 100 +   0
task_2  .   region  150 200 100 +   0
```

Resubmitting

In order to resubmit a particular run of the pipeline for data analysis from a particular stage the following command can be used.

```
snp_phylogenomics_control --resubmit output_dir/ --start-stage starting-stage
```

The *output_dir/* is the directory containing all the results of a previous run of the pipeline. The *start-stage* defines the starting stage for the new analysis. For more details on the particular stages to use please run the command:

```
snp_phylogenomics_control --help
```

Ådam Philipy

Rand, Hugh

From:	Timme, Ruth
Sent:	Monday, May 05, 2014 1:11 PM
To:	Strain, Errol; Rand, Hugh; Luo, Yan; Pettengill, James; Davis, Steven
Cc:	Payne, Justin *
Subject:	Re: New(?) bioinformatics tools

FYI - ParSNP and gingr come from adam philipy's group – their public release is expected in a month or so.

An email from earlier this year:

Hi all,
At some point over the past year I mentioned to you that my group was working on a new tool for computing core-genome alignments and trees. We are pleased to announce the first beta release of this tool, ParSNP, developed by Todd Treangen, and its associated visualization tool, Gingr, developed by Brian Ondov. You are one of a trusted set of collaborators who asked to receive the beta release (or had it forced upon you!). We are not yet ready to put out a public release, so please do not share outside of your group without asking.

ParSNP is an ultrafast core-genome alignment tool capable of aligning up to thousands of closely-related strains. As primary output ParSNP produces multi-alignments, SNP calls, and a core-genome tree. This is contained in a single, compressed "harvest" file that can be read by Gingr for interactive visualization of the alignment and tree. Common file formats such as VCF for SNPs, XMFA for alignments, and Newick for trees can also be extracted from the Harvest file to be analyzed with other tools. The available documentation for both tools is included in the ParSNP tarball. You can download them here:

ParSNP 64-bit Linux: http://cbcb.umd.edu/~amp/parsnp/parsnp_v1.0b-LINUX.tar.gz
ParSNP OS X: http://cbcb.umd.edu/~amp/parsnp/parsnp_v1.0b-OSX.tar.gz

Gingr 64-bit Linux: http://cbcb.umd.edu/~amp/parsnp/gingr.gz
Gingr OS X: http://cbcb.umd.edu/~amp/parsnp/gingr.app.zip

If you are interested, please play around with these tools. We would be very interested in your feedback, especially from a users perspective. Please feel free to reach out to me, Todd, or Brian (CC'ed) if you have any questions or have a dataset you would like us to look at. We look forward to hearing from you.

Best regards,
-Adam

-------.

Ruth E. Timme, PhD
Research Microbiologist, CFSAN/FDA
Office 4E-009
5100 Paint Branch Parkway
College Park, MD 20740
ruth.timme@fda.hhs.gov
(240) 402-2196

From: <Strain>, Ruth Timme <Errol.Strain@fda.hhs.gov>
Date: Monday, May 5, 2014 12:50 PM
To: "Rand, Hugh" <Hugh.Rand@fda.hhs.gov>, "Luo, Yan" <Yan.Luo@fda.hhs.gov>, James Pettengill <James.Pettengill@fda.hhs.gov>, "Davis, Steven" <Steven.Davis@fda.hhs.gov>
Cc: Ruth Timme <ruth.timme@fda.hhs.gov>, "Payne, Justin *" <Justin.Payne@fda.hhs.gov>
Subject: New(?) bioinformatics tools

I visited NCBI last week and heard about some tools that were new to me (but you already may be in the loop)

iMetAMOS - http://www.cbcb.umd.edu/software/imetamos-0 - Looks like a nice multiple assembly pipeline, basically a more polished version of our in-house version with some downstream metrics

gingr – can't find any good links outside of https://github.com/marbl/gingr, looked like a nice view for SNPs in multiple alignments

ParSNP – Core SNP finder for microbes, can't find any links, looked interesting.

Errol Strain, Ph.D.
Chief, Biostatistics Branch
Office of Analytics and Outreach
FDA Center for Food Safety and Applied Nutrition
5100 Paint Branch Pkwy
College Park, MD 20740
Office:240-402-2815

Adam Philipy

SEARCH

Home People News Research Education Resources Contact Us

iMetAMOS

iMetAMOS is an automated ensemble assembly pipeline; iMetAMOS encapsulates the process of running, validating, and selecting a single assembly from multiple assemblies. iMetAMOS packages several leading open-source tools into a single binary that automates parameter selection and execution of multiple assemblers, scores the resulting assemblies based on multiple validation metrics, and annotates the assemblies for genes and contaminants. iMetAMOS is available as a workflow within the metAMOS package starting with version 1.5.

Home Page:
iMetAMOS (http://www.cbcb.umd.edu/software/imetamos)

Keywords:
Sequencing (/areas/sequencing)
Genome/Metagenome Assembly (/areas/genomemetagenome-assembly)

Images: