Analysing Genome Alignments with Gubbins

Contents

1. Introduction

Since the introduction of high-throughput, second-generation DNA sequencing technologies, there has been an enormous increase in the size of datasets being used for estimating bacterial population phylodynamics. Although many phylogenetic techniques are scalable to hundreds of bacterial genomes, methods which have been used for mitigating the effect of mechanisms of horizontal sequence transfer on phylogenetic reconstructions cannot cope with these new datasets. Gubbins (Genealogies Unbiased By recomBinations In Nucleotide Sequences) is an algorithm that iteratively identifies loci containing elevated densities of base substitutions while concurrently constructing a phylogeny based on the putative point mutations outside of these regions. Simulations demonstrate the algorithm generates highly accurate reconstructions under realistic models of short-term bacterial evolutiondiversification of sequences through both point mutation and recombination, and can be run in only a few hours on alignments of hundreds of bacterial genome sequences.

Gubbins is suitable for use on alignments of hundreds of sequenceson large whole genome alignments. It can be used on alignments of individual loci, but will only be able to identify recombinations that are short relative to the length of the locus. The time taken for the algorithm to converge on a stable solution increases approximately quadratically with the number of samples when RAxML is used to construct phylogenies; this increase can be ameliorated to some extent by using FastTree to construct at least the initial phylogeny. The input should be a whole genome sequence alignment; there is no need to remove accessory genome loci, as the algorithm should be able to cope with missing data.

Gubbins cannot distinguish elevated densities of polymorphisms arising through recombination from other potential causes. These may be assembly or alignment errors, mutational hotspots or regions of the genome with relaxed selection. Such false positives are more likely to arise on longer branches within a phylogeny; it is recommended that populations be subdivided into smaller groups of less diverse samples that can each be independently analysed with Gubbins. For some discussion of potential confounding factors in the analysis of such population genomic datasets, please see (1).

2. Requirements

Gubbins is a command line program designed to be run on Linux or Max OSX systems and requires Python version 2.7 or greater. Gubbins can also be run on Windows operating systems using a Bio-Linux virtual machine. Gubbins is dependent upon independent phylogenetics software (RAxML (2) and FastTree2 (3)) and sequence reconstruction software (FastML (4)) that are included in the installation package.

On large alignments (e.g. hundreds or thousands of whole genome sequences) Gubbins may take hours or days to converge and require several gigabytes of RAM. Please note that the first iteration will usually be the slowest, as the starting phylogeny is constructed using all polymorphic sites. Subsequent phylogenies do not use information from base substitutions introduced by putative recombinations and are typically faster.

In order to view the outputs of Gubbins, we would recommend:

- Seaview (http://doua.prabi.fr/software/seaview) for viewing sequence alignments
- FigTree (http://tree.bio.ed.ac.uk/software) for viewing phylogenies
- Artemis (https://www.sanger.ac.uk/resources/software/artemis/) for viewing annotation (.embltab and .gff) files

3. Installation

Installation instructions can be found in the relevant INSTALL document included in the package.

4. Input files

The main input file required for Gubbins is a FASTA format alignment. The header of each sequence should be unique. The alignment should be flush, with the '-' character used to indicate missing data or gaps. When specifying a starting tree, the Newick format should be used.

5. Running Gubbins

To run Gubbins with default settings:

```
run_gubbins.py [FASTA alignment]
```

Processing options:

Option	Description
--tree_builder, -t	The algorithm to use in the construction of phylogenies in the analysis; can be 'raxml', to use RAxML, 'fasttree', to use FastTree, or 'hybrid', to use FastTree for the first iteration and RAxML in all subsequent iterations. Default is RAxML
--iterations, -i	The maximum number of iterations to perform; the algorithm will stop earlier than this if it converges on the same tree in two successive iterations. Deafult is 5.
--min_snps, -m	The minimum number of base substitutions required to identify a recombination. Default is 3.
--converge_method, -z	Criteria to use to know when to halt iterations (weighted_robinson_foulds ,robinson_foulds , recombination). Default is weighted_robinson_foulds.

Output options:

Option	Description
--use_time_stamp, -u	Include a time stamp in the name of output files to avoid overwriting previous runs on the same input file. Default is to not include a time stamp.
--verbose, -v	Print debugging messages. Default is off.

--no_cleanup, -n	Do not remove files from intermediate iterations. This option will also keep other files created by RAxML, FastML and FastTree, which would otherwise be deleted. Default is to only keep files from the final iteration.

robinson_foulds: measures the difference in tree shape excluding branch lengths and halts when the difference between the current tree and any previous tree is zero.

weighted_robinson_foulds: measures the difference in tree shape including branch lengths and halts when the difference between the current tree and any previous tree is zero. This is the default method.

recombination: compares the coordinates of all the recombination's in each taxon(leaf node) with the recombination's found in all previous iterations

More information can be found in (9).

6. Output files

When running Gubbins in the default mode, the output files will have names of the form "X.S", where "X" is the prefix taken from the input FASTA file and "S" is a suffix from the list below. If the "-u" flag is used, then the files will have the form "X.Y.S", where "Y" is the time stamp. If the "-n" option is used, then intermediate files are retained, as detailed below; if one file is generated per iteration, the file names will be of the form "RAxML_result.X.iteration_I.S", where "I" denotes the iteration.

Output file suffices:

Suffix	Description
.recombination_predictions.embl	Recombination predictions in EMBL file format.
.recombination_predictions.gff	Recombination predictions in GFF format
.branch_base_reconstruction.embl	Base substitution

	reconstruction in EMBL format.
.summary_of_snp_distribution.vcf	VCF file summarising the distribution of SNPs
.per_branch_statistics.csv	Per branch reporting of the base substitutions inside and outside recombination events.
.filtered_polymorphic_sites.fasta	FASTA format alignment of filtered polymorphic sites used to generate the phylogeny in the final iteration.
.filtered_polymorphic_sites.phylip	Phylip format alignment of filtered polymorphic sites used to generate the phylogeny in the final iteration.
.final_tree.tree	Newick format file containing the final phylogeny.
. node_labelled.tre	Newick format file containing the final tree with nodes annotated for comparison with FastML's sequence reconstruction and the per branch statistics CSV file.
.aln.start (-n mode only)	FASTA format alignment of polymorphic sites in the input FASTA file.
.output_tree (-n mode only)	These files (one per non-final iteration) contain the intermediate output trees from FastML in the same format as the final tree in the .node_labelled.tre file.

.ancestor.tre (-n mode only)	These files (one per iteration) contain the parent and child nodes corresponding to all nodes in the tree in tab-delimited format. The node names correspond to those used in per branch statistics CSV file.
.seq.joint.txt (-n mode only)	FastML joint reconstructions (one per iteration) of the bases of the variable sites for each node in the tree. Names of internal nodes correspond to those in .output_tree and .ancestor_tree.
.prob.joint.txt (-n mode only)	These files (one per iteration) contain the FastML joint reconstruction log likelihoods for each variable site and a total joint log likelihood.

A log.txt file is also created by FastML.

7. Viewing output files

The ".gff", ".tab" and ".branch_snps.tab" files can be read onto any sequence from the alignment using the freely-available genome browsing software Artemis. Artemis can also display a summary of the SNP distribution by reading in the VCF file. The package also includes a Python script for constructing PDFs to visualise the distribution of base substitutions or recombinations relative to the phylogeny:

```
gubbins_drawer.py –o [output PDF file name] –t [Newick
format tree file] [tab file]
```

The .stats file contains summary statistics for each branch in the tree in tab-delimited format. For each branch, the following statistics are reported:

Statistic	Description
Node	Name of the node subtended by the branch. This can either be one of the taxa included in the input alignment, or an internal node, which will be named with an N followed by a number. The .output_tree output file contains node labels corresponding to these names, and the ancestor-descendent relationships are also described in the .node_labelled.tre output file.
Total SNPs	Total number of base substitutions reconstructed onto the branch
Num of SNPs inside recombinations	Number of base substitutions reconstructed onto the branch that fall within a predicted recombination (r).
Num of SNPs outside recombinations	Number of base substitutions reconstructed onto the branch that fall outside of a predicted recombination. i.e. predicted to have arisen by point mutation (m).
Num of Recombination Blocks	Total number of recombination blocks reconstructed onto the branch
Bases in recombinations	Total length of all recombination events reconstructed onto the branch
r/m	The ratio of base substitutions predicted to have been imported through recombination to those occurring through point mutation. This value gives a measure of the relative impact of recombination and mutation on the variation accumulated on the branch.

rho/theta	The ratio of the number of recombination events to point mutations on a branch; a measure of the relative rates of recombination and point mutation.
Genome Length	The total number of aligned bases between the ancestral and descendent nodes for the branch excluding any missing data or gaps in either.

8. Examples

Two example whole genome bacterial alignments are included in the Gubbins package. These are:

File	Description
ST239.aln	An alignment of 14 methicillin resistant *Staphylococcus aureus* isolates from the MLST sequence type (ST) 239. These isolates are a subset of the isolates sequenced in (5, 6).
PMEN1.aln	An alignment of 11 *Steptococcus pneumoniae* genomes from the PMEN1 lineage. These isolates are a subset of those analysed in (7).

Example 1: ST239

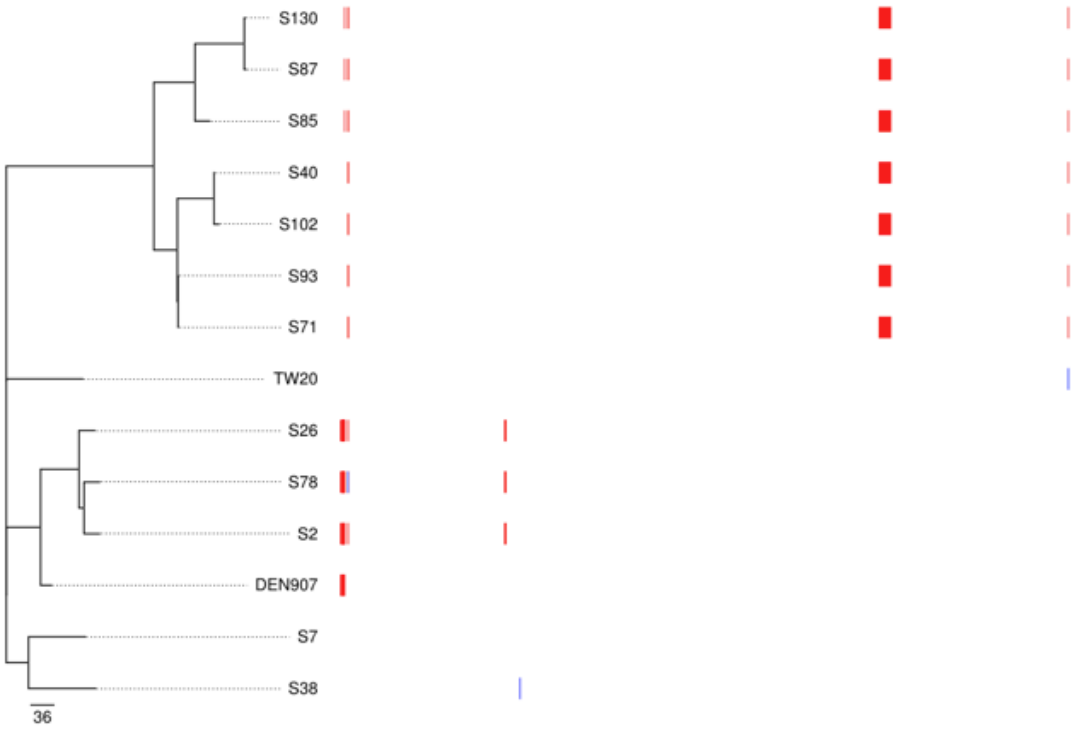
To run a Gubbins analysis of the ST239 data using the default parameters:

```
_gubbins.py ST239.aln
```

To create a showing the predicted regions of recombination against a phylogenetic reconstruction based on the final iteration of the Gubbins analysis:

```
gubbins_drawer.py -o ST239.pdf -t ST239.tre RAxML_result.
```

The resulting (below) shows the phylogeny of ST239 on the left. For each isolate, blocks representing the regions identified as recombinations by gubbins are indicated by coloured blocks. Blue blocks are unique to a single isolate while red blocks are shared by multiple isolates. The horizontal position of the blocks represents their position in the alignment.



Example 2: PMEN1

To run an analysis of the PMEN1 data with a maximum of 10 iterations:

```
_gubbins.py -i 10 PMEN1.aln
```

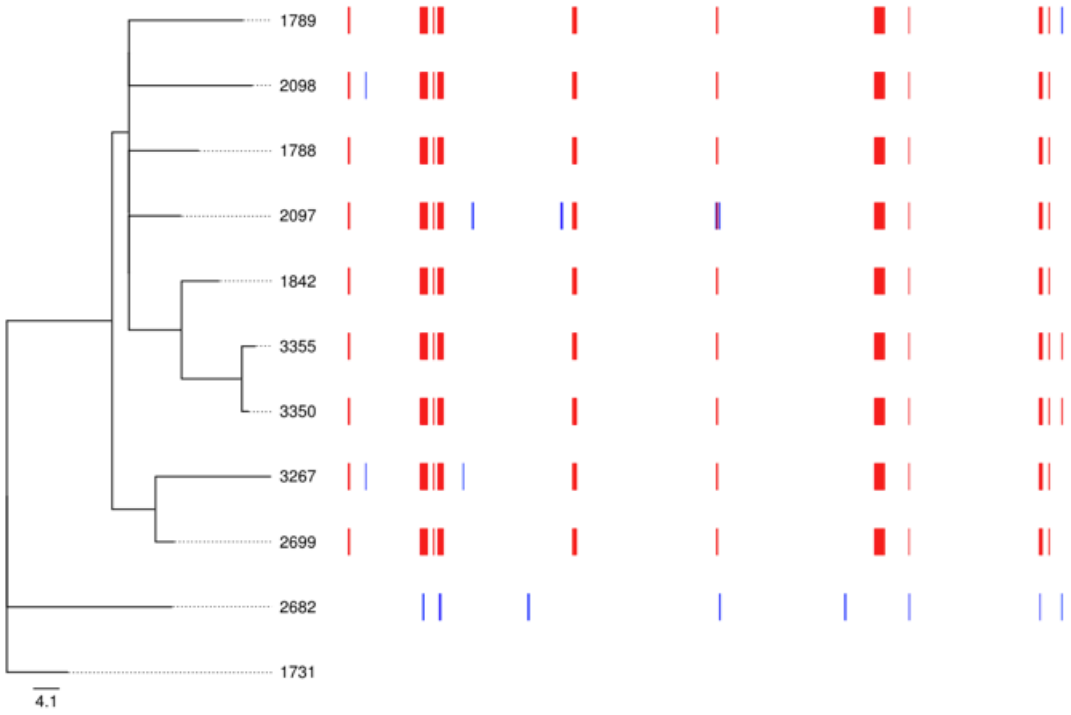
To create a showing the predicted regions of recombination against a phylogenetic reconstruction based on the final iteration of the Gubbins analysis:

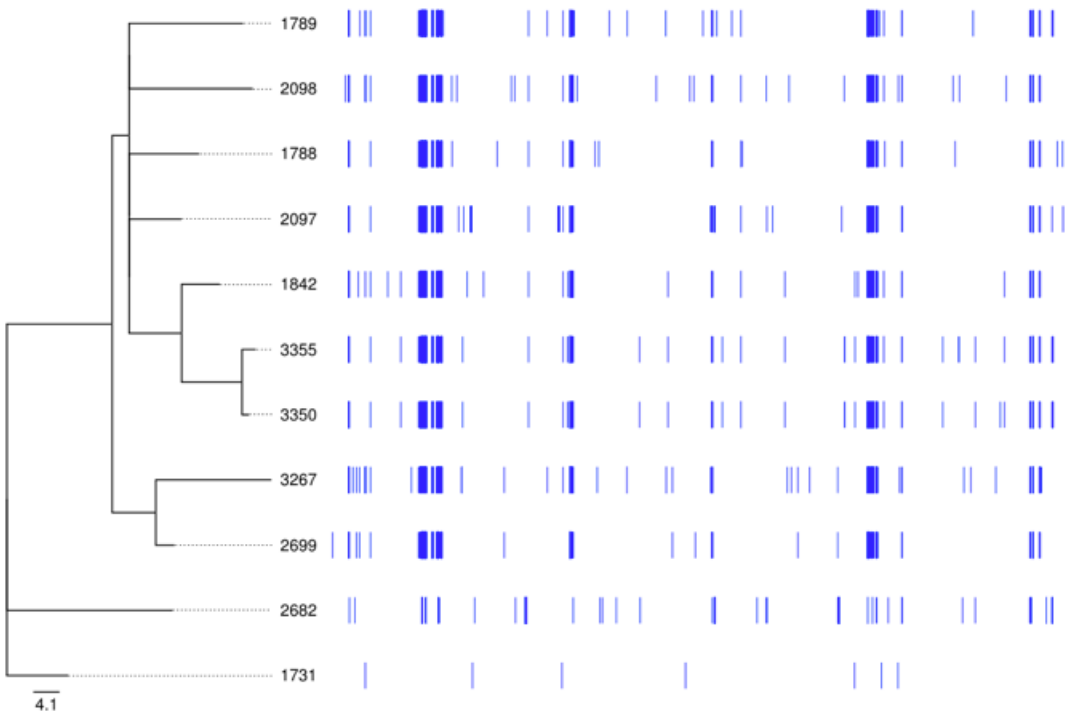
```
gubbins_drawer.py -o PMEN1.pdf -t PMEN1.tre
```

To create a showing the reconstructed SNPs against the same tree:

```
gubbins_drawer.py -o PMEN1_SNPs.pdf -t PMEN1.trebranch_
```

The two files, PMEN1.pdf and PMEN1_SNPs.pdf are shown below, illustrating the correlation between reconstructed recombination regions and SNP density.





9. Troubleshooting

RAxML returns a non-zero exit status – check the sequence identifiers in the FASTA alignment don't contain any unusual characters, such as '#', that are not accepted by RAxML.
Gubbins fails to converge on a stable solution – are you Fookes?this may be due to the limit of genetic resolution between very closely related samples. It may be necessary to obtain higher resolution data, if possible, or otherwise prune the dataset to remove very closely related isolates in order to obtain a stable solution.
Gubbins takes a long time to analyse the data – if Gubbins is taking a long time to analyse a large dataset, it is worth using FastTree rather than RAxML to construct the initial, or all, phylogenies; removing identical sequences from the dataset; removing sequences with a high proportion of missing data that may be difficult to reconstruction, or splitting the dataset into multiple sets of more closely related samples using a fast, phylogeny-independent approach (e.g. BAPS (8)).

Gubbins exits shortly after starting with a segmentation fault or runs out of memory later in the process - This can be caused by Gubbins running out of memory. The most memory-intensive stage is running RAxML, and the RAxML memory calculator (http://sco.h-its.org/exelixis/web/software/raxml/) can be used to check you

requirements. To solve this problem, you will either need to use a more highly powered machine, or split your data into more closely-related subsets.

The 'gubbins' python module cannot be imported - check that your PYTHONPATH variable (use the command "`echo $PYTHONPATH`") includes the directory in which the 'gubbins' python module is stored.

10. Citing Gubbins

Croucher N. J., Page A. J., Connor T. R., Delaney A. J., McQuillan J. A., Bentley S. D., Parkhill J., Harris S.R. Rapid phylogenetic analysis of large samples of recombinant bacterial whole genome sequences using Gubbins (2014), *Nuc. Acids Res.*, gku1196.

11. References

1. **Croucher NJ, Harris SR, Grad YH, Hanage WP.** 2013. Bacterial genomes in epidemiology—present and future. Philos. Trans. R. Soc. B Biol. Sci. 368.
2. **Stamatakis A, Ludwig T, Meier H.** 2005. RAxML-III: a fast program for maximum likelihood-based inference of large phylogenetic trees. Bioinformatics, 2004/12/21 ed. 21:456–463.
3. **Price MN, Dehal PS, Arkin AP.** 2010. FastTree 2–approximately maximum-likelihood trees for large alignments. PLoS One 5:e9490.
4. **Ashkenazy H, Penn O, Doron-Faigenboim A, Cohen O, Cannarozzi G, Zomer O, Pupko T.** 2012. FastML: a web server for probabilistic reconstruction of ancestral sequences. Nucleic Acids Res. 40:W580–W584.
5. **Harris SR, Feil EJ, Holden MT, Quail MA, Nickerson EK, Chantratita N, Gardete S, Tavares A, Day N, Lindsay JA, Edgeworth JD, de Lencastre H, Parkhill J, Peacock SJ, Bentley SD.** 2010. Evolution of MRSA during hospital transmission and intercontinental spread. Science (80-.)., 2010/01/23 ed. 327:469–474.
6. **Castillo-Ramírez S, Corander J, Marttinen P, Aldeljawi M, Hanage WP, Westh H, Boye K, Gulay Z, Bentley SD, Parkhill J.** 2012. Phylogeographic variation in recombination rates within a global clone of Methicillin-Resistant Staphylococcus aureus (MRSA). Genome Biol. 13:R126.
7. **Croucher NJ, Harris SR, Fraser C, Quail MA, Burton J, van der Linden M, McGee L, von Gottberg A, Song JH, Ko KS, Pichon B, Baker S, Parry CM, Lambertsen LM, Shahinas D, Pillai DR, Mitchell TJ, Dougan G, Tomasz A, Klugman KP, Parkhill J, Hanage WP, Bentley SD.** 2011. Rapid pneumococcal evolution in response to clinical interventions. Science (80-.)., 2011/01/29 ed. 331:430–434.

8. **Tang J, Hanage WP, Fraser C, Corander J.** 2009. Identifying currents in the gene pool for bacterial populations using an integrative approach. PLoS Comput Biol, 2009/08/08 ed. 5:e1000455.

9. **Tree Comparison.** http://www.monkeyshines.co.uk/blog/archives/891