GraftM

https://github.com/geronimp/graftM

Version 0.4.2

Joel A. Boyd, Dr. Ben Woodcroft

Table of Contents

For Impatient Peoples

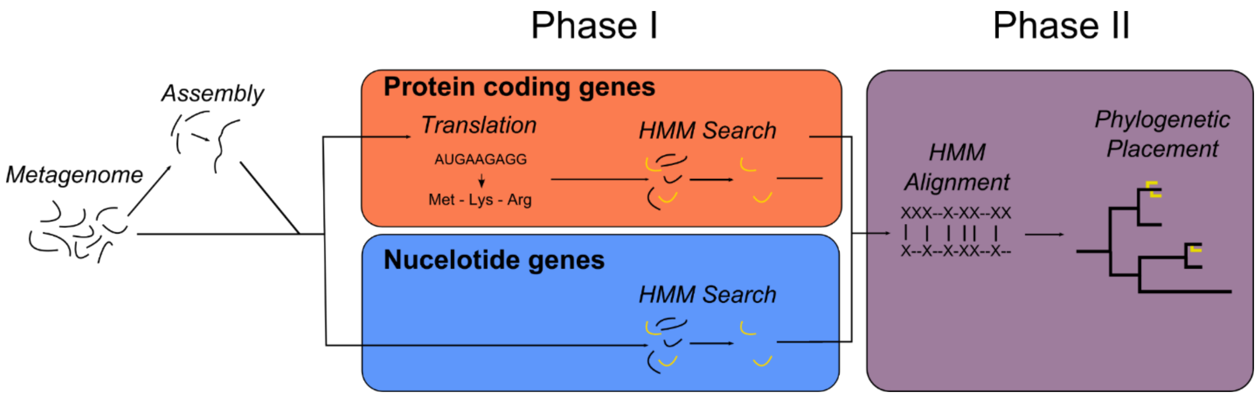
Running GraftM can be as simple as:

```
$ graftM graft --forward <READS> --graftm_package <PACKAGE>
```

Introduction to the way GraftM works

This is a pipeline that can rapidly generate community profiles based on genetic marker genes using approaches that are independent of traditional pairwise sequence comparisons.

Basically, it can be split into two phases. The first is the identification of reads using Hidden Markov Models (HMMs). The second phase is the placement of these reads into phylogenetic trees. Finally, a community composition is then derived from the taxonomic placement of each read.



A more detailed workflow can be found on the following page. As input, GraftM can take short read data (100 b.p.) and assembled contigs. Data can be either metagenomic, metatranscriptomic, or protein sequence.

GraftM Packages

GraftM requires what is called a GraftM package to run. GraftM packages essentially contain two things. The first is a HMM that is used in Phase I, and the second is a reference package (refpkg) containing a phylogenetic tree that is used by pplacer in the placement step. The taxonomy within each of these trees have been curated based off an in-house genome tree database. All taxonomy follows the GreenGenes format. In the case of 16S rRNA, the HMM and the tree were built from the GreenGenes database clustered at the 97% level. The tree was also curated using green genes taxonomy.

Eukaryotes

It should be noted that at the moment, GraftM does *not* have any package corresponding to Eukaryotic 18S. This is because any type of taxonomy scheme available, has no consistent taxonomic rank information, which is essential in the annotation of a phylogenetic tree. Some regions are known to be conserved between Eukaryotic 18S and the 16S of Prokaryotes. This means some 18S will likely be picked up in the search phase of GraftM if you have Eukaryotes in your sample. GraftM will do the best it can to filter out these 18S reads with a Eukaryote specific HMM.

Running GraftM, and options

Graft

GraftM is the standard analysis pipeline described above.

Inputs

Inputs may be provided as in fasta (.fa) or fastq, gzipped (.fq.gz). Sequences protein or nucleotide sequence may be provided. Multiple datasets may be used in one run, just comma separate the files as follows:

```
$ GraftM graft --forward reads1.fa,reads2.fa --graftM_package mcrA.gpkg
```

Running options

`--forward`

The reads you wish to run through GraftM, either in fasta (.fa) or fastq, gzipped format (.fq.gz). If you would like to run multiple samples at once, GraftM can handle this. Just provide it with a comma separated list of the file names, with no spaces.

`--reverse`

If your data are paired end, you may wish to provide the reverse reads. If you are running more than one dataset through, ensure that the **order of the files matches the order that was provided for the `--forward` flag.**

`--eval`

Here you can specify the evalue cutoff for the hmmsearch if the default cutoff (1e-05) is not sufficient.

`--threads`

Here you can specify the number of threads used.

`--placements_cutoff`

This flag allows you to change the likelihood cutoff for phylogenetic placement of reads (0 – 1). It should be mentioned that while using a likelihood cutoff below the default (0.75) may yield more resolved taxonomic placement, too much confidence should not be placed in these results.

`--graftm_package`

This is the GraftM package containing the HMM, reference package (explained in section XXX).

`--force`

Use this flag if you wish to overwrite a previous run with the same name.

`--input_sequence_type`

Specify if you are providing protein or nucleotide sequence data. If this flag is not provided, graftM will attempt to auto predict the sequence type. In the off chance that GraftM mis-interprets a file, you may overwrite this prediction with this flag.

`--seach_and_align_only`

Use this flag to stop before the alignment, and after the search and alignment step. This will provide hits from the Phase I of GraftM, both unaligned and aligned to the HMM.

`--search_only`

Use this flag to stop after the search step, yielding unaligned sequences that were found in Phase I of GraftM.

`--check_total_euks`

Use this flag to run an 18S specific HMM on the sample, to provide an estimation of the eukaryote abundance in the sample. Note it will not classify the 18S reads found.

`--output_directory`

Here you can specify the name of the output directory provided by GraftM when it runs.

`--version`

Return the version of the GraftM you are using.

Outputs

The output of GraftM will be a directory named after the input file (e.g. results from a file named `my_reads.fq.gz` will be placed in a directory named `my_reads`), unless a name was specified with `--output` flag. The contents of this folder will slightly differ, depending on which pipeline was run:

Nucleotide pipeline:

- `my_reads_count_table.txt`
    - A tally of reads for each lineage detected within your sample

- `my_reads_coverage_table.txt`
    - The average coverage of each lineage detected of the HMM, based off the average length of each read

- `my_reads_relative_table.txt`
    - The read counts in the above count table, normalised by total reads detected

- `my_reads_hits.fa`
    - Reads that were detected as hits at the search phase of GraftM

- `my_reads_hits.fa`
    - The reads above, aligned to the HMM used for searching

- `my_reads_krona.html`
    - A krona plot (Ondov *et al.* 2011) providing a graphic presentation of your community structure, as determined by GraftM with the marker gene used.

- `basic_stats.txt`
    - A table with basic run statistics, such as a breakdown of the runtime for each step in GraftM, total number of reads detected and extent of 18S contamination detected. If anything would be useful to you to add to this table, please request it via the GitHub page (https://github.com/geronimp/graftM)

- `my_reads_placements.jplace`
    - A .jplace file produced by pplacer in Phase II of graftM. This file may be used for further analysis, such as the visualisation of placements, or running edge PCAs with guppy.

- `my_reads_placements.guppy`
    - A tabulated format of the placements in the .jplace file.

Protein pipeline will provide the above, with the addition of:

- `my_reads_orf.fa`
    - The ORFs of each nucleotide sequence detected (found in `my_reads_hits.fa`)

- `my_reads.aln.fa`
    - If the protein pipeline was run, this file will contain the aligned ORFs only.

Assemble

Assemble may be used after running graft to attempt to assemble the reads recovered by GraftM. The basic principle is, you provide assemble with the output directory provided by graft and it will read the placement file, parse the reads into groups of the lowest taxonomic ranking possible, and attempt to assemble them using various approaches. The most basic command would be:

```
$ GraftM assemble --graft_run <graft_output>
```

Again multiple input folders may be given to bolster the assembly. Just pass these to the flag as a comma separated list, as with `graft`.

Options

`--graft_run`

The output directory created by `graft` after it has finished.

`--kmer`

If you're using a velvet assembly, the kmer can be specified here. The default it 51.

`--assembly_type`

Choose from phrap or velvet.

`--finish`

Follow a velvet assembly with a phrap assembly (THIS IS DODGY)

Outputs

`assemble` will contain files with the following prefixes:

> <GreenGenes_Taxonomic_Rank>_<Name_Of_Lineage>_<Number_Of_Reads>_~
>
> For example:
>
>> G_Methanoregula_2041_~

This will be followed by a suffix describing the file/folder. For a velvet assembly:

- `reads.fa`
  - The unassembled reads in fasta format
- `assembly.fa`
  - The assembled reads in fasta format. This file may contain nothing if the assembly was unsuccessful.
- `velvet_assembly/`
  - The processing file generated by velvet for assembly.

For a phrap assembly:

- `reads.fa`
- `assembly.fa`
- Processing files associated with phrap

This will be followed by the

Manage

This simple utility is, as of the moment, simple a subcommand that allows to easily access the sequences GraftM has detected for each lineage. Say, you're interested in Betaproteobacteria and you wanted all the the sequences within this class. You would need to run the following program to retrieve the reads corresponding to this lineage:

```
$ graftM manage --seq Betaproteobacteria  --profile <graft_output>
```

And this will make a file containing all sequences assigned at the Betaproteobacteria Class, or at a lower rank. If you're interested in more than one lineage, you may provide a file containing one lineage per line, like so:

```
$ cat methanogens.txt
  Methanosaeta
  Methanosarcina
  Methanoflorens
  ...
```

You will have to supply the `--file` flag, so `manage` knows how to interpret this input:

```
$ graftM manage --seq methanogens.txt –file --profile <graft_output>
```

Finally, if you wish to obtain the reads that were assigned at a given taxonomic rank, however, for whatever reason, could not be classified at a lower rank (for example, reads that were assigned at the Betaproteobacteria level, and no lower) you can supply the `--non_cumil` flag, and the fasta file with just those reads will be written.

A note on read counts, and further points on analysis

When running GraftM on short read metagenomic data a few considerations must be met if you are considering comparisons between multiple samples, or across different marker genes on the same, or different samples.

Across samples

The size of metagenome datasets vary across run. While a given organism may not be any more abundant across two samples, the one that was sequenced deeper (i.e. more sequence data)

Across marker genes

The length of genes differ considerably, and this is reflected in the length of HMM used by GraftM. It is intuitive then, that genes that comprise more sequence will attract more reads when GraftM is run on Metagenomic data. Consider normalising for the length of the gene you used, before drawing comparisons between marker genes.

Another, age old consideration is copy number variation between species. More copies mean more reads, and while some programs exist that predict 16S copy number (Angly *et al,* 2014), other marker genes are relatively unexplored. Unfortunately, I can offer no remedy. Too bad.

References and Acknowledgments

Jol pls..