Nov 05, 2024

GenFlow: A Bioinformatic Workflow for PhyloGenomic Analysis

DOI
https://dx.doi.org/10.17504/protocols.io.14egn6kepl5d/v1

Bradd Mendoza[1]

[1]Health Research Institute, University of Costa Rica

Bradd Mendoza
Health Research Institute, University of Costa Rica

DOI: https://dx.doi.org/10.17504/protocols.io.14egn6kepl5d/v1

Protocol Citation: Bradd Mendoza 2024. GenFlow: A Bioinformatic Workflow for PhyloGenomic Analysis. **protocols.io** https://dx.doi.org/10.17504/protocols.io.14egn6kepl5d/v1

Protocol status: In development
We implemented this protocol successfully, and potential upgrades are planned for future development.

Created: September 30, 2024

Last Modified: November 05, 2024

Protocol Integer ID: 108599

Keywords: bioinformatic workflow for phylogenomic analysis genflow, phylogenomic analysis genflow, phylogenomic tree, genomes in fasta format, phylogenomic tree in newick format, streamlined bioinformatics pipeline, genome, copy core gene, genflow github repository, bioinformatic workflow, genflow, gene, ncbi datasets command, fasta format, accession codes of the reference, including ncbi datasets command, accession code, average nucleotide identity value

Abstract

GenFlow is a streamlined bioinformatics pipeline that integrates multiple tools, including NCBI Datasets command-line, EDirect, Anvi'o, Prodigal, MAFFT, FastTree, and pyANI, to offer users an easy, one-command solution for performing phylogenomic and ANI analysis. Simply provide your genomes in FASTA format along with a text file listing the accession codes of the reference genomes for the analysis. The program will generate a phylogenomic tree in Newick format using all single-copy core genes shared among the genomes and a heatmap displaying the Average Nucleotide Identity values.
All the necessary instructions and data to perform the analysis are available in the **GenFlow GitHub repository**.

Set up conda environment

1 You will need to run some command lines to create a Conda environment with the package and all necessary dependencies.
All the information and data is availavable in the **GenFlow GitHub repository.**

2

Command

Conda Install

```
git clone https://github.com/braddmg/GenFlow.git
cd GenFlow
conda env create -f GenFlow.yml
conda activate GenFlow
bash setup.sh
conda deactivate
conda activate GenFlow
```

Input Data Preparation

3 You will only need two things:
1. The assembled genomes you want to compare against others in FASTA format. For this example, we
will use three genomes, that are in the folder Data. These are potential new subspecies of Aeromonas hydrophila.
2. A text file containing all the accession numbers of the reference genomes you wish to include in your analysis. You can obtain these accession numbers from scientific papers, GenBank, or both. For this example, we'll use a short dataset with 10 genomes of Aeromonas hydrophila. In the same folder is a dataset (large.txt) with a larger number of Aeromonas reference genomes for more comprehensive analysis.
You can use either dataset based on your computational capacity and time constraints.

The genome list contain NCBI accession numbers for reference genomes and should look like this:

Command

short.txt

```
cat Data/short.txt
GCF_000633175.1
GCF_000820325.1
GCF_016805405.1
GCA_963892115.1
GCF_002850695.3
GCF_022631195.1
GCF_000014805.1
GCF_029025785.1
GCF_002285935.1
GCF_023920205.1
GCF_000820005.1
GCF_028355655.1
```

Quick start for inexperienced users in the command Line

4 If you only want to generate visual representations like trees and other genomic plots, there's no need to run all the code step by step. We have prepared a bash script that automates the entire process for you. All you need to do is set up a few parameters and then wait for the results to be generated (it may take some time).

5 There is an example command with the short dataset:

Command

GenFlow

```
GenFlow -g short.txt -f INISA09F.fasta,INISA10F.fasta,INISA16F.fasta -
t 8 -G 0.8 -F 0.8
```

In this example:
-g specifies the file containing the accession numbers of the reference genomes.
-f specifies the FASTA files for the analysis.
-t specifies the number of threads to use (recommended 8).
-The -G and -F options specify the geometric and functional index values for core gene selection. In this case, we selected 0.8, which is also the default value for both options.
-As we are not employing the -N flag, the phylogenomic tree will be created with aminoacid sequences. But hey, if you fancy yourself a hardcore evolutionary biologist who swears by DNA sequences, just add the flag :)
-The MCL inflation parameter (-I) was not used, so the default value of 10 was employed. This value is recommended for identifying gene clusters in closely related genomes.
Refer to this tutorial for more information: **Anvio-pangenomics**

6 To see the final plots, just go to the last step.
Or continue with the protocol that explain each part of the code

Download reference genomes and add taxonomy name

7 After downloading the genomes, you might notice their names look like a jumble of letters and numbers, such as:

```
GCF_029025785.1_ASM2902578v1_genomic.fna
```

Not exactly the most user-friendly name, right? To make these filenames more informative, we can use esearch along with some bash commands to rename the files to something clearer that includes the Species, Strain, and Accession, such as:

```
Aeromonas_hydrophila_K533_GCF_029025785_1.fasta
```

Command

Download and rename genomes

```
#Download ncbi tool
curl -o datasets 'https://ftp.ncbi.nlm.nih.gov/pub/datasets/command-
line/v2/linux-amd64/datasets'
chmod +x datasets

#Create a folder to save some logs
mkdir logs

#Download your references genomes and place them in a new folder
named "Genomes"
./datasets download genome accession --inputfile short.txt >>
logs/download.log 2>&1
unzip ncbi*
rm *.zip
mkdir Genomes
mv ncbi_dataset/data/GC*/*.fna Genomes/
rm -r ncbi_dataset md5sum.txt README.md datasets dataformat
cp *.fasta Genomes
cd Genomes

#Download esearch tool and set the path
sh -c "$(curl -fsSL
https://ftp.ncbi.nlm.nih.gov/entrez/entrezdirect/install-edirect.sh)"
sh -c "$(wget -q
https://ftp.ncbi.nlm.nih.gov/entrez/entrezdirect/install-edirect.sh -
O -)" > ../logs/esearch.log 2>&1
echo "export PATH=\$HOME/edirect:\$PATH" >> $HOME/.bash_profile >>
../logs/esearch.log 2>&1
export PATH=${HOME}/edirect:${PATH} >> ../logs/esearch.log 2>&1

#Replace name with taxa
for f in GC* ; do term=$(echo $f | cut -f1,2 -d'_') ; esearch -db
assembly -query $term | esummary | \
xtract -pattern DocumentSummary -sep ' ' -element
Organism,Sub_value,AssemblyAccession | sed 's/ /_/g' | sed 's/://g' |
\
sed 's+/+_+g' | sed 's/,/_/g' |  sed 's/[.]//g' | sed 's/-/_/g' | sed
's/_([^)(]*)//' | sed 's/=//g' ; done > NEW
ls -1 *.fna > OLD
paste OLD NEW|while read OLD NEW;do mv ${OLD} ${NEW};done
```

```
for i in ls *GCF*; do mv $i $i\.fasta; done
rm OLD NEW
```

8 After running the code, you should have generated a series of FASTA files inside the "Intermediate" folder that look like this:

- Aeromonas_dhakensis_KN_Mc_6U21_GCF_002285935_1.fasta
- Aeromonas_dhakensis_b2_100_GCF_023920205_1.fasta
- Aeromonas_hydrophila_AC133_GCF_022631195_1.fasta
- Aeromonas_hydrophila_K533_GCF_029025785_1.fasta
- ...

Now you are ready for the next step!

Identify Single Copy Core Genes and extract them with Anvio

9 This part was tested in **Anvio** v7.1 and v8.0; if you have an older version, we do not know how it works.
Be prepared to exercise some patience, as importing genomes into the database format in Anvio can take a considerable amount of time.
Once the genomes are imported, coding sequences will be annotated using Prodigal. Anvio will utilize this information to identify all the single-copy core genomes shared across your datasets and produce a new FASTA file containing the amino acid sequences.
For our analysis, we will only retain genes with a geometric and functional index ≥0.8 for each gene. For more details, please refer to the relevant section of the **Anvio tutorial.**
A Friendly Reminder!!
If you utilize this pipeline, don't forget to cite the **Anvio**and **Prodigal** papers!

Command

Obtaining Single Copy Core Genes

```
#Reformat fasta files to be compatible with Anvio. New files will
have the extension ".fa"
for i in `ls -1 *.fasta | sed 's/.fasta//'`
do
anvi-script-reformat-fasta $i\.fasta \
                           -o $i\.fa \
                           -l 1000 \
                           --simplify-names --seq-type NT >>
../logs/reformat.log 2>&1
    anvi-gen-contigs-database -f $i\.fa -o $i\.db >>
../logs/database.log 2>&1
    anvi-run-hmms -c $i\.db -T 32 >> ../logs/hmms.log 2>&1
done
#Some bash codes to create a requires file called to perform
pangenomics (yes, we are doing that)
ls -1 *.db > path.txt
sed -i '1s/^/contigs_db_path\n/' path.txt
ls -1 *.db | sed 's/.db//' > name.txt
sed -i '1s/^/name\n/' name.txt
paste name.txt path.txt > external-genomes.txt
rm name.txt path.txt
#Save the genomes in a new Anvio database and perform pangenomics
anvi-gen-genomes-storage -e external-genomes.txt -o Filo-GENOMES.db
>> \
../logs/storage.log 2>&1
anvi-pan-genome -g Filo-GENOMES.db --project-name Filo --num-threads
32 >> \
../logs/pangenome.log 2>&1
#Create the new file with all SCCGs with a geometric and functional
index over 0.8
NG=$(ls *.fasta | wc -l)
anvi-get-sequences-for-gene-clusters -g Filo-GENOMES.db -p Filo/Filo-
PAN.db \
-o core-sequences.fasta \
--max-num-genes-from-each-genome 1 --min-num-genomes-gene-cluster-
occurs "${NG}" \
--concatenate-gene-clusters --min-geometric-homogeneity-index 0.8 \
--min-functional-homogeneity-index 0.8 >> ../logs/core.log 2>&1
```

10 The new FASTA file is named **proteins-sequences.fasta**. If you want to know how many genes were used to create this file, you can check the **core.log** file in the logs folder.
Since we gathered a large number of genes at this step, using amino acid sequences works well and requires less time and computational resources.

We will run the phylogenomic tree (finally)

11 We need to align the core-sequences file with maft, and then create the phylogenomic tree with Fasttre.
For this step we have a default parameter of 1,000 resamples and the JTT+CAT model for aminoacid sequences. But hey, if you fancy yourself a hardcore evolutionary biologist who swears by DNA sequences, you can still get your fix! Just add the **--report-DNA-sequences** option to the last code.

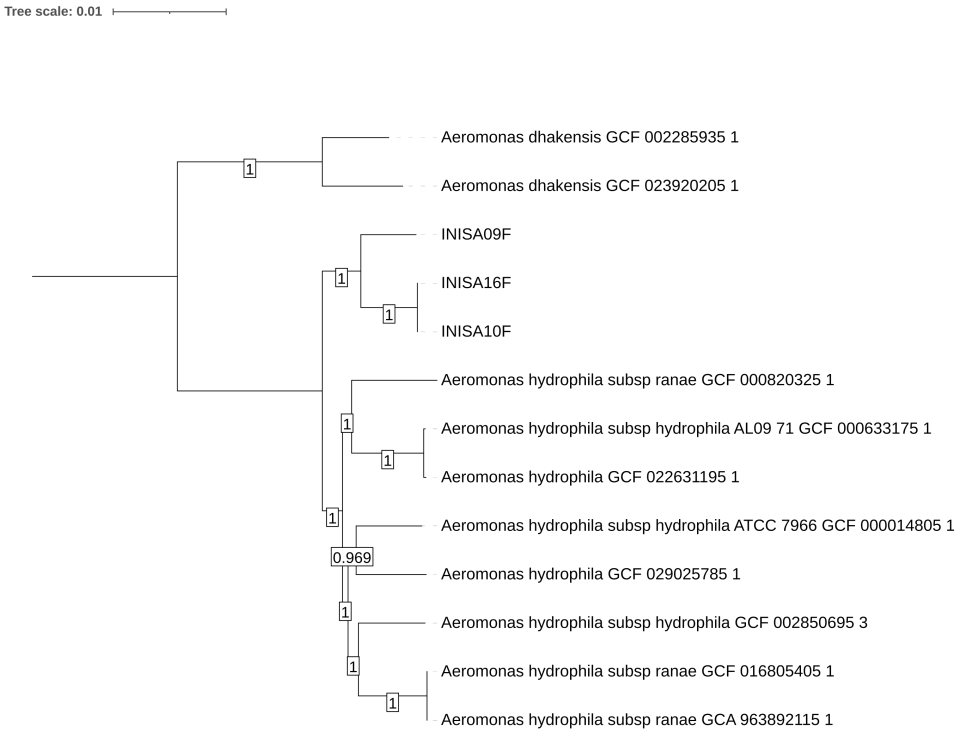
Command

Phylogenomic tree

```
mafft --retree 1 --thread 32 --maxiterate 0 core-sequences.fasta >
core-sequences-aligned.fasta
mkdir ../results
anvi-gen-phylogenomic-tree -f core-sequences-aligned.fasta \
                           -o ../results/phylogenomic-tree.txt
```

Phylogenomic Tree

12



You can observe that the genomes labeled as "INISA" form a distinct clade from the other subspecies. Bootstrap values, based on 1,000 resamplings performed by **FastTree**, provide support for these clades.

Some additional plots

13 We can also perform an Average Nucleotide Identity (ANI) analysis using **pyANI**. The next command will create a heatmap with ANI values of all genomes.

Command

pyANI

```
average_nucleotide_identity.py -i fa -o ../results/pyANI --labels
labels.txt --classes classes.txt -g --gmethod seaborn --gformat svg -
v -l ba_ANI.log >> ../logs/pyANI.log
```

Modifying the heatmap

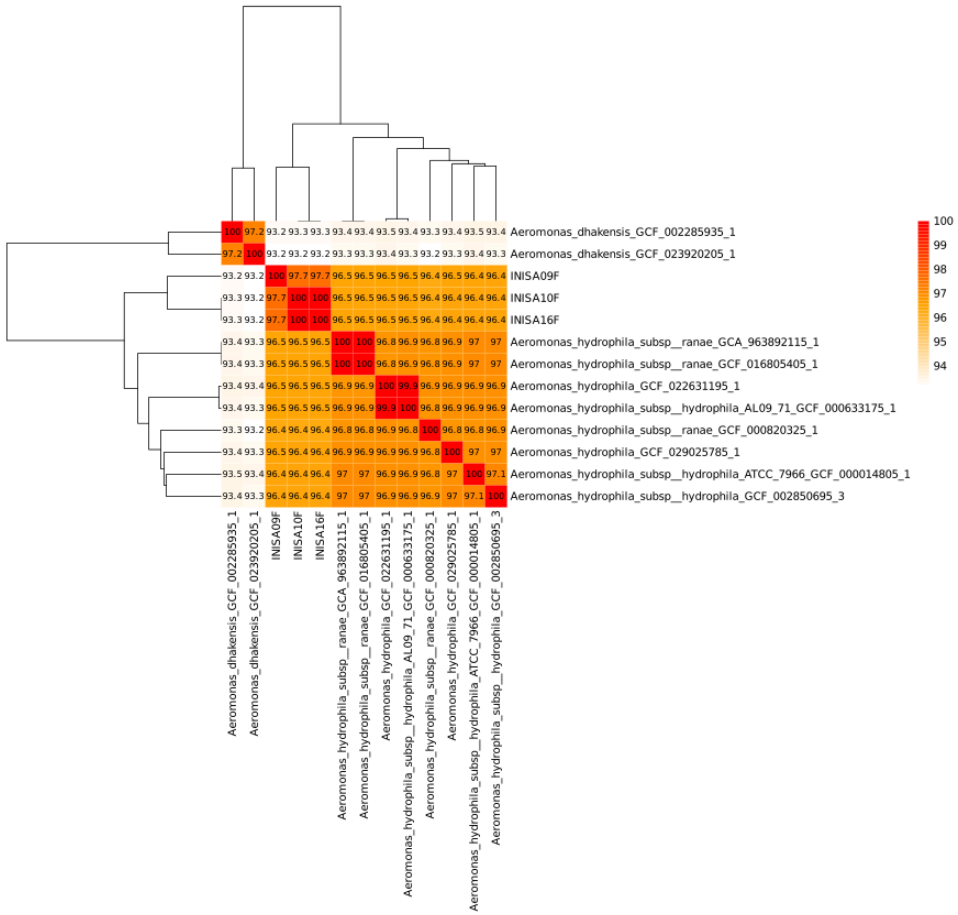
14 The results are generated in a matrix dataset containing all comparisons. We can then import this data into R to create more visually appealing heatmaps using the following code.

Command

Rplots

```
library(pheatmap)
library(viridis)
library(ggplot2)
library(viridis)
library(reshape2)
library(grid)
data <- read.table("ANIm_percentage_identity.tab", sep = "\t", header
= T, row.names = 1)

# Convert the data to a matrix
data_matrix <- as.matrix(data)

# Multiply all values by 100 for percentage
data_matrix <- data_matrix * 100

# Melt the data to long format
data_long <- melt(data_matrix)

# Custom formatting to remove decimal places for whole numbers
formatted_numbers <- apply(data_matrix, c(1, 2), function(x) {
  # Format numbers without decimal points
  if (x %% 1 == 0) {
    return(as.character(as.integer(x)))  # Convert to integer and
then to character
  } else {
    return(as.character(round(x, 1)))  # Round to one decimal place
for non-whole numbers
  }
})

# Create the heatmap
heatmap_plot <- pheatmap(
  data_matrix,
  cluster_rows = TRUE,           # Cluster the rows (to maintain
order)
  cluster_cols = TRUE,           # Cluster the columns
  clustering_method = "complete", # Complete linkage clustering
  clustering_distance_rows = "euclidean", # Euclidean distance for
rows
  clustering_distance_cols = "euclidean", # Euclidean distance for
```

```
columns
  color = colorRampPalette(c("white", "orange", "red"))(100), #
Gradient from white to orange to red
  display_numbers = formatted_numbers, # Use custom formatted numbers
  fontsize_number = 9,           # Increase font size of values to 8
  cellwidth = 20,                # Increase cell width for better
spacing
  cellheight = 20,               # Increase cell height for better
spacing
  scale = "none",                # No scaling of ANI values
  border_color = NA,             # Remove borders around cells
  angle_col = 90,                # Rotate x-axis labels 90 degrees
  legend = TRUE,                 # Title for the legend
  treeheight_row = 200,          # Increase row dendrogram height for
larger branches
  treeheight_col = 200,          # Increase column dendrogram height
for larger branches
  show_colnames = TRUE,          # Show x-axis labels
  show_rownames = TRUE,          # Show y-axis labels
  margin = c(10, 10, 10, 10),    # Increase margins (bottom, left,
top, right)
  number_color = "black"
)

# Save the heatmap to a PDF with appropriate dimensions
pdf("heatmap.pdf", width = 18, height = 18)  # Adjust size as needed

# Print the heatmap to the PDF device
print(heatmap_plot)

# Close the PDF device
dev.off()
```



This is the final heatmap. The main GenFlow function generates this heatmap, offering three different color palettes to choose from.

Protocol references

- **Anvi'o**: Eren, A. M., et al. (2015). Anvi'o: An advanced analysis and visualization platform for 'omics data. *PeerJ*, 3, e1319. https://doi.org/10.7717/peerj.1319
- **MAFFT**: Katoh, K., & Standley, D. M. (2013). MAFFT multiple sequence alignment software version 7: improvements in performance and usability. *Molecular Biology and Evolution*, 30(4), 772–780. https://doi.org/10.1093/molbev/mst010
- **FastTree**: Price, M. N., Dehal, P. S., & Arkin, A. P. (2010). FastTree 2 – Approximately Maximum-Likelihood Trees for Large Alignments. *PLoS ONE*, 5(3), e9490. https://doi.org/10.1371/journal.pone.0009490
- **NCBI EDirect**: Kans, J. (2013). Entrez Direct: E-utilities on the UNIX command line. In *Entrez Programming Utilities Help* (Internet). National Center for Biotechnology Information (US). **https://www.ncbi.nlm.nih.gov/books/NBK179288/**
- **pyANI**: Pritchard, L., Glover, R. H., Humphris, S., Elphinstone, J. G., & Toth, I. K. (2016). Genomics and taxonomy in diagnostics for food security: soft-rotting enterobacterial plant pathogens. *Analytical Methods*, 8, 12-24. https://doi.org/10.1039/C5AY02550H
- **PRODIGAL**: Hyatt, D., Chen, G.-L., Locascio, P. F., Land, M. L., Larimer, F. W., & Hauser, L. J. (2010). Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics*, 11, 119. https://doi.org/10.1186/1471-2105-11-119