Report Group 5

Ancient HGT to Firm5 (from unknown donor)

Joaquim, Virginie & Kamil

We are going to investigate which genes of our Firm5 group are potentially acquired by ancient horizontal gene transfer from unknown donor.

Our GitHub repository is: https://github.com/KamilSJaron/SAGE_Firm5_specific_HGT.git
On Vital-IT, you can access the same repo by typing:

cd /scratch/beegfs/monthly/mls_2016/claivaz_ricci/SAGE_Firm5_specific_HGT

In this folder, README.md comprises all informations about our folders and files. To make it short, data contains all inputs and outputs files whereas scripts contains all scripts for data analysis.
Folders are in red, files (inputs and outputs) in green and scripts in blue.

Summary:

We are going to select all gene families which contains orthologous genes if at least 80% of all strains in a given group of bees are present. Then, we are going to extract each protein sequences of all orthologous genes of each group of bees and blast them against RefSeq database (ncbi database). Then, we are going to parse blast results according to parameters. Thanks to blast results, we are going to determine hierarchical taxonomy and hierarchical taxonomy distance for each blast hit. This will allow us to plot % of identity of blast hits against hierarchical taxonomy distance. In the end, all bast hits corresponding to 'Lactobacillus' (in addition to supposed contaminations) will be discarded as we are interested in HGT from species other than Lactobacillus. From blast results, we will perform 2 different analysis. The first one is constructing phylogenetic trees of gene families potentially acquired by HGT and the second one is to predict genomic islands of gene families potentially acquired by HGT.

Group of bees:

Bumble_bees: B_pascuorum+B_bohemicus
('F225','F230','F233','F234','F236','F237')+('F228', 'F245','F246','F247')
Honey_bees: A_mellifera
('JF72','JF73','JG30','JF74','JF75','JF76','F259','F260','F261','F262','F263','L183','L184','L185','L186')
Bumble_Honey_bees: Bumble_bees+Honey_bees
Outgroup: other strains which are not Bumble_bees and Honey_bees
('JG29','LA14','LA2','LDB','LGAS','LHV','LJP','WANG')

Steps:

1. For each group of bees (Bumble_bees, Honey_bees, Bumble_Honey_bees, Outgroup), subgroup_sort.py looks into each Gene_family_* (each line) of GeneFamilies.txt. If a

Gene_family_* line contains at least 80% of all strains present in the group of bees, it stores the Gene_family_* line into its corresponding file.
Input: GeneFamilies.txt (in data)
Outputs: Bumble_bees.txt, Honey_bees.txt, Bumble_Honey_bees.txt and Outgroup.txt (in data/sort_group)

2. For each group of bees and for each Gene_family_*, proteinseqextract.py extracts all protein sequences for further 'Blast' step. This function creates a folder for each group of bees which comprises fasta files for each Gene_family_* containing protein sequences.
Inputs: outputs of subgroup_sort.py in data/sort_group (Bumble_bees.txt, Honey_bees.txt, Bumble_Honey_bees.txt or Outgroup.txt), all_proteins.fasta (in mls_2016/blast folder)
Outputs: folders in data of Bumble_bees_proteins, Honey_bees_proteins, Bumble_Honey_bees_proteins - each folder contains .fasta files of gene families (Gene_family_*.fasta) - a file concerns one gene family and contains all reference_genome|protein_ID and their sequence in fasta format

3. For every protein sequences present in each Gene_family_*.fasta files in each folder of group of bees (data/Bumble_bees_proteins, data/Honey_bees_proteins, data/Bumble_Honey_bees_proteins), blast_gene_families.sh and function_blast.sh perform blastp against RefSeq database (module Blast is previously added on Vital-IT, see blast_gene_families.sh). RefSeq database is suitable as it is big (and manually curated) and it will allow us to access taxonomic informations from NCBI for further 'Phylogeny' step. blast_gene_families.sh runs blastp command specifying queries and outputs. function_blast.sh creates output folders (data/blast/Bumble_bees_proteins, data/blast/Honey_bees_proteins, data/blast/Bumble_Honey_bees_proteins) and assigns variables used in blast_gene_families.sh (variables correspond to every Gene_family_*.fasta files present in each folder of group bees and stored in list_files.txt).
Inputs: outputs of proteinseqextract.py - Gene_family_*.fasta in data/Bumble_bees_proteins, data/Honey_bees_proteins, data/Bumble_Honey_bees_proteins (to select Gene_family_*.fasta, function_blast.sh will go through list_files.txt - created in the terminal: ***ls Gene_family_* > list_files.txt****)*
Outputs: Gene_family_*.out of its corresponding Gene_family_*.fasta of its corresponding folder of group of bees (data/Bumble_bees_proteins, data/Honey_bees_proteins, data/Bumble_Honey_bees_proteins) in its corresponding output folders (data/blast/Bumble_bees_proteins, data/blast/Honey_bees_proteins, data/blast/Bumble_Honey_bees_proteins) - each Gene_family_*.out file contains blast hits for each reference_genome|protein_ID's sequence - for each reference_genome|protein_ID's sequence (query): 5 first lines summarizing blastp information (version of BLASTP, query, database, fields and the total number of found hits) - fields correspond to the name of each resulting column (total of 14 columns) - subsequent lines correspond to every hits of the query

4. For every blast results of each reference_genome|protein_ID's sequence (query) present in each Gene_family_*.out files in each folder of group of bees (data/blast/Bumble_bees_proteins, data/blast/Honey_bees_proteins, data/blast/Bumble_Honey_bees_proteins), blast_hits_extract.py parses hits according to different parameters (threshold_alignment_length, threshold_ID, threshold_eval, threshold_bitscore):

- threshold_alignment_length : constant = 0.8
- threshold_ID : starts at 50
- threshold _eval : 0.00001
- threshold_bitscore : 0

For a given Gene_family_*.out, if there are less than 25 blast hits with parameters below, we relax the threshold_ID by multiplying it by 0.8 until reaching at least 25 blast hits. blast_hits_extract.sh creates output folders (data/parsed_blast/Bumble_bees_proteins, data/parsed_blast/Honey_bees_proteins, data/parsed_blast/Bumble_Honey_bees_proteins) and assigns parameters as below.
Inputs: outputs of blast_gene_families.sh and function_blast.sh - Gene_family_*.out in data/blast/Bumble_bees_proteins, data/blast/Honey_bees_proteins, data/blast/Bumble_Honey_bees_proteins (to select Gene_family_*.out, blast_hits_extract.py will go through list_files.txt - created in the terminal: ***ls Gene_family_* > list_files.txt****)*
Outputs: Gene_family_*_parsed.out of its corresponding Gene_family_*.out of its corresponding folder of group of bees (data/blast/Bumble_bees_proteins, data/blast/Honey_bees_proteins, data/blast/Bumble_Honey_bees_proteins) in its corresponding output folders (data/parsed_blast/Bumble_bees_proteins, data/parsed_blast/Honey_bees_proteins, data/parsed_blast/Bumble_Honey_bees_proteins) - each Gene_family_*_parsed.out file contains the 'best' blast hits for each reference_genome|protein_ID's sequence - the header is '# Query_ID Subject_titles %_Identity Alignment_length evalue bit_score' - for each 'best' hits, informations are stored in its corresponding column

5. From every **parsed** blast results of each reference_genome|protein_ID's sequence (query) present in each Gene_family_*_parsed.out files in each folder of group of bees (data/parsed_blast/Bumble_bees_proteins, data/parsed_blast/Honey_bees_proteins, data/parsed_blast/Bumble_Honey_bees_proteins), extract_taxonomy_hierarchy.py first creates a list of subject IDs (hit IDs, for example 'WP_052726720'). Subsequently, thanks to Entrez (from Bio package), the function extracts the hierarchical taxonomy of each subject IDs (from protein database of NCBI, for example: 'Bacteria;Firmicutes;Bacilli;Lactobacillales;Lactobacillaceae;Lactobacillus'). *The example ('WP_052726720') is used as reference hierarchical taxonomy.* Finally, the function looks for the most recent taxa which is shared between each hierarchical taxonomy of each subject IDs and the reference hierarchical taxonomy. Hierarchical taxonomy distance are subjective:

- Lactobacillus = 1
- Lactobacillaceae = 2
- Lactobacillales = 3
- Bacilli = 4
- Firmicutes = 5
- Bacteria = 6
- None = 7 (which either corresponds to Archae or Eukaryota - contaminations)

The function creates a summary file (hierarchical_taxonomy.txt) which is located in data/parsed_blast.
Inputs: Gene_family_*_parsed.out of its corresponding folder of group of bees (data/parsed_blast/Bumble_bees_proteins, data/parsed_blast/Honey_bees_proteins, data/parsed_blast/Bumble_Honey_bees_proteins)

Outputs: hierarchical_taxonomy.txt in data/parsed_blast - the first column corresponds to the strain IDs, the second one corresponds to the hierarchical taxonomy and the third one corresponds to the subjective hierarchical taxonomy distance

6. For every **parsed** blast results of each reference_genome|protein_ID's sequence (query) present in each Gene_family_*_parsed.out files in each folder of group of bees (data/parsed_blast/Bumble_bees_proteins, data/parsed_blast/Honey_bees_proteins, data/parsed_blast/Bumble_Honey_bees_proteins), plot_identity_taxonomy.R extracts its percentage of identity and its hierarchical taxonomy distance (in hierarchical_taxonomy.txt) and plot them against each other. Moreover, for each Gene_family_*_parsed.out, on one hand, we will perform linear regression model in addition to polynomial model and investigate if models differ. If they differ, it would mean that polynomial model is the best one and suggest potential horizontal gene transfer.
Inputs: Gene_family_*_parsed.out of its corresponding folder of group of bees (data/parsed_blast/Bumble_bees_proteins, data/parsed_blast/Honey_bees_proteins, data/parsed_blast/Bumble_Honey_bees_proteins) and hierarchical_taxonomy.txt in data/parsed_blast
*Outputs: Bumble_taxo_plot.pdf, Honey_taxo_plot.pdf, Bumble_Honey_taxo_plot.pdf and potential_HGT.txt in data/parsed_blast - each *.pdf file contains all plots of a given bee group - potential_HGT.txt contains a list of Gene_family_*_parsed.out that are orthologous genes potentially acquired by HGT*

7. Manually, we decide which gene families are good candidates to investigate if orthologous genes are acquired by HGT. These gene family candidates are listed in list_files_HGT_to_trees.txt (in data/parsed_blast). Gene_family_1058_parsed.out and Gene_family_991_parsed.out (data/parsed_blast/Bumble_Honey_bees_proteins) correspond to 'confirmed' HGT candidates. Gene_family_1099_parsed.out (data/parsed_blast/Bumble_Honey_bees_proteins) and Gene_family_1674_parsed.out (data/parsed_blast/Bumble_bees_proteins) correspond to 'supposed' HGT candidates. Gene_family_1048_parsed.out (data/parsed_blast/Bumble_Honey_bees_proteins) corresponds to non-HGT candidate. First, extract_seq_blast_hits.py extracts the protein sequence of every blast hits of selected Gene_family_*_parsed.out (listed in list_files.txt). From this, align_potential_HGT.sh aligns the sequence of every blast hits of selected Gene_family_*_parsed.out (listed in list_files.txt) in its folder of group of bees (data/phylogeny_potential_HGT/Bumble_bees_proteins, data/phylogeny_potential_HGT/Honey_bees_proteins, data/phylogeny_potential_HGT/Bumble_Honey_bees_proteins). Then, tree_wo_bootstrap.sh runs RAXML tool to infer phylogenetic trees (without bootstrapping) of all aligned protein sequences of every blast hits of selected Gene_family_*_parsed.out (in its bee folder in data/phylogeny_potential_HGT).
extract_seq_blast_hits.py:
Inputs: list_files.txt in data/parsed_blast, Gene_family_*_parsed.out of its corresponding folder of group of bees (data/parsed_blast/Bumble_bees_proteins, data/parsed_blast/Honey_bees_proteins, data/parsed_blast/Bumble_Honey_bees_proteins) listed in list_files.txt
Outputs: amino_acid_seq_Gene_family_*_parsed.fasta in data/parsed_blast - each output file contains protein sequences of every blast hits of a selected gene family. The ID name of each sequence is defined as follows: StrainName_SujectID_HierarchicalTaxonomyDistance

align_potential_HGT.sh:
Inputs: amino_acid_seq_Gene_family_*_parsed.fasta in data/parsed_blast
Outputs: Gene_family_*_parsed.multifasta in its corresponding folder of group of bees (data/phylogeny_potential_HGT/Bumble_bees_proteins, data/phylogeny_potential_HGT/Honey_bees_proteins, data/phylogeny_potential_HGT/Bumble_Honey_bees_proteins) - each output file contains aligned protein sequences of every blast hits of a selected gene family
tree_wo_bootstrap.sh:
Inputs: Gene_family_*_parsed.multifasta in its corresponding folder of group of bees (data/phylogeny_potential_HGT/Bumble_bees_proteins, data/phylogeny_potential_HGT/Honey_bees_proteins, data/phylogeny_potential_HGT/Bumble_Honey_bees_proteins)
*Outputs: *ML_wo_bootstrap_Gene_family_*_parsed.out in data/phylogeny_potential_HGT/RAxML_results*
>> RAxML_bestTree.ML_wo_bootstrap_Gene_family_*_parsed.out can then be opened using FigTree tool (available at http://tree.bio.ed.ac.uk/software/figtree/)

8. The inference of genomic islands is performed using Islandviewer tool (available at http://www.pathogenomics.sfu.ca/islandviewer/). For this purpose, two reference genomes from Bumble bees group (***F5_237*** and ***F5_245***) and one from Honey bees group (***Lb_183***) are analysed. For each referenge genome, extract_coordinates_function_from_gbkfiles.py extracts the protein name of every genes present in each Gene_family_* from GeneFamilies.txt (in data). Thanks to GeneBank files (mls_2016/genome_files), the coordinates and the function of each protein are recovered. Then, we map manually the specific position of the protein onto its reference genome. Results are present in data/IslandViewerResults, *.png shows the circular representation of each reference genome and the position of the genomic islands, *.tsv contains the coordinates of each genomic island and which method allows this inference. Moreover, MappingIntoGenome.txt sums up the different results for this part.
Input: GeneFamilies.txt in data, GeneBank files in mls_2016/genome_files
*Outputs: *.png files , *.tsv files and MappingIntoGenome.txt in data/IslandViewerResults*