

# Chapter 14

## Robust Identification of Orthologues and Paralogues for Microbial Pan-Genomics Using GET\_HOMOLOGUES: A Case Study of pIncA/C Plasmids

Pablo Vinuesa and Bruno Contreras-Moreira

### Abstract

GET\_HOMOLOGUES is an open-source software package written in Perl and R to define robust core- and pan-genomes by computing consensus clusters of orthologous gene families from whole-genome sequences using the bidirectional best-hit, COGtriangles, and OrthoMCL clustering algorithms. The granularity of the clusters can be fine-tuned by a user-configurable filtering strategy based on a combination of blastp pairwise alignment parameters, hmmscan-based scanning of Pfam domain composition of the proteins in each cluster, and a partial synteny criterion. We present detailed protocols to fit exponential and binomial mixture models to estimate core- and pan-genome sizes, compute pan-genome trees from the pan-genome matrix using a parsimony criterion, analyze and graphically represent the pan-genome structure, and identify lineage-specific gene families for the 12 complete pIncA/C plasmids currently available in NCBI's RefSeq. The software package, license, and detailed user manual can be downloaded for free for academic use from two mirrors: <http://www.eead.csic.es/compbio/soft/gethoms.php> and <http://maya.ccg.unam.mx/soft/gethoms.php>.

**Key words** Orthologs, Paralogs, Pan-genomics, Comparative genomics, Bacterial genomes, pIncA/C plasmids, Core-genome, Pan-genome, Software, Open-source

---

### 1 Introduction

The advent of next-generation sequencing (NGS) technology has recently boosted the number of genome sequencing projects publicly available [1]. This trend empowers comparative genomics and pan-genomics approaches to genome analysis, motivating the development of more and better software tools for these tasks. Early within-species genome comparisons, such as those performed by the group of Fred Blattner on three *Escherichia coli* strains with contrasting ecological niches (the commensal K12, the uropathogen CFT073, and enterohemorrhagic EDL933), revealed an extensive “mosaic” genome structure [2]. They determined that only 39 % of their combined proteomes were shared by all three strains.

However, they found that the strains maintained remarkable synteny in the common, vertically inherited genome backbone, which is interrupted by the insertion of genomic islands that are acquired by horizontal gene transfer. Genes on these islands were found to be largely responsible for defining the lifestyles and niches of the strains. Three years after this landmark paper, Tettelin and colleagues were the first to introduce the concept of the pan-genome, the collective genetic repertoire of a certain species, developing first computational strategies to estimate its size [3]. Ever since, the microbial pan-genome has been a key topic in microbial genomics, as it has profound implication on how we understand bacterial evolution, niche adaptation, and population structure, with strong practical implications in areas such as epidemiology and vaccine development [4].

Here we present a detailed tutorial on the use of the open-source GET\_HOMOLOGUES software package [5], demonstrating some of its bioinformatic, statistical, and graphical capabilities for microbial pan-genomics. Protocols are provided to define robust orthologous gene families, fit exponential and mixture models to estimate core- and pan-genome sizes, analyze and graphically represent the pan-genome structure, and identify lineage-specific gene families for the 12 complete pInCA/C plasmids currently available in NCBI's RefSeq [6].

The package is released under a GNU General Public License and is written mainly in Perl and R. GET\_HOMOLOGUES is highly configurable; runs on MacOSX, and Linux operating systems; and was designed to take advantage of multiprocessor machines and computer clusters to distribute time-consuming blast+ [7] and HMMER3 [8] jobs. If constrained by RAM, the software implements the possibility to write data structures temporarily to disk using BerkeleyDB. Together these features make it possible to analyze large datasets of hundreds of microbial genomes on a dedicated server. Smaller sets up to ~50 bacterial genomes can be analyzed on a modern commodity desktop or laptop in reasonable time [5]. It automatically computes homologous gene families based on three alternative and well-established reciprocal BLAST hit algorithms (RBHAs): our own implementation of the bidirectional best-hit (BDHB) algorithm [5], COGtriangles [9], and OrthoMCL [10]. RBHAs are heuristic in nature [11, 12], but have been recently shown to produce highly accurate orthologous gene clusters when compared with tree-based methods, which are generally prohibitive for large datasets due to the computational burden they impose [13].

The GET\_HOMOLOGUES package bundles several auxiliary scripts to facilitate the interrogation of homologous gene clusters, and computation of pan-genome sets and pan-genome trees based on the pan-genomic presence-absence matrix. A unique feature of the software is its capacity to compute consensus core- and

pan-genomes, that is, to define these genome sets based on the joint evidence of any combination of the three abovementioned clustering algorithms. This generates very robust, although conservative clusters. The tightness of the clusters generated by each algorithm can be fine-tuned by controlling key blast parameters such as percentage overlap and identity of pairwise alignments and E-score cutoff value. It is also possible to make orthologous gene clusters even more stringent by imposing a partial synteny criterion and/or by scanning the Pfam domain composition of the clusters using hmmscan of the HMMER3 package. Several auxiliary scripts are provided for the statistical and graphical analysis of core- and pan-genomes, which can fit both exponential and binomial mixture models to the data to estimate the sizes of the core- and pan-genomes [3, 14, 15]. The package also bundles an installation script that takes care of the installation of most external dependencies, including the downloading and formatting of the latest Pfam database required by hmmscan for domain-scanning of proteins. A detailed manual with >40 pages documenting all the software's features and options makes the use of GET\_HOMOLOGUES reasonably user friendly.

To demonstrate some of the key features and capabilities of GET\_HOMOLOGUES, we present detailed protocols on the use of the main script `get_homologues.pl` and several auxiliary scripts bundled with the package to compute robust core- and pan-genome sets of 12 large, broad-host-range bacterial resistance plasmids of the IncA/C incompatibility group (pIncA/C) [16–19], statistically estimate the size of their core- and pan-genomes, graphically visualize the structure of the pan-genome, and identify genes specifically found in the two plasmids containing the *bla*NDM-1 (New Delhi metallo-beta-lactamase-1) gene [20]. The encoded protein is one of the most recently reported metallo-enzymes conferring resistance to all beta-lactams, including carbapenems, the last drug type in this class conferring nearly universal, anti-Gram-negative activity until the recent appearance of carbapenemases [21]. To make things worse, carbapenemase-producing bacteria are typically multidrug resistant (MDR) or even pan resistant [22], making the emergence and rapid spread of NDM a worldwide public health concern [18, 23]. Different plasmids, including those of the A/C incompatibility group, are largely involved in the rapid spread of NDM and other resistance genes such as *bla*<sub>CYM-2</sub>, *tetA*, *flo*, and *sul* [16, 18].

---

## 2 Materials

1. The protocol depends on the installation of the GET\_HOMOLOGUES software package (version 20140901 or later) [5] on a MacOSX, or Linux box. For larger datasets (>50 fully sequenced bacterial genomes), the software is best

run on a multiprocessor machine, with 8GB of RAM or more, or on a Linux computer cluster. For the demo dataset analyzed herein, a standard commodity laptop or desktop with 2 cores and 1GB of RAM will suffice. The package is freely available for academic purposes, but not for commercial or military use, as detailed in the license agreement, which can be found along with the software from two mirror servers: <http://maya.ccg.unam.mx/soft/gethoms.php> (Mexico) and <http://www.eead.csic.es/compbio/soft/gethoms.php> (Spain). Additionally the user will need to download the GenBank files for the selected pIncA/C plasmids, also available as a compressed tar file from the URL provided below: [http://maya.ccg.unam.mx/soft/protocols\\_gethom/methMolBiol2014\\_get\\_homologues.tgz](http://maya.ccg.unam.mx/soft/protocols_gethom/methMolBiol2014_get_homologues.tgz).

Subheading 3.1 provides detailed methods on how to unpack this file, which also contains all the code and auxiliary scripts used in this chapter.

2. Typographical conventions: `Monospaced text` will be used for all commands to be issued by the user, as well as directory names, program names, and output. The command prompt will be represented with the `$` symbol.

---

## 3 Methods

### 3.1 Downloading Selected pIncA/C Plasmid GenBank Files Using NCBI's Entrez System

The easiest way to get the GenBank files required for the protocols in this chapter is to download them from the URL provided below. Create a directory named `pIncAC/` to store the files, move into it, and use the following command to fetch the file:

```
# Make the directory, cd into it and save its
path for easy access later on
$ mkdir pIncAC && top_dir=$(pwd) && cd pIncAC
$ gbk_dir=$(pwd)
$ wget -c
http://maya.ccg.unam.mx/soft/protocols_gethom/
methMolBiol2014_get_homologues.tgz
```

Unpack the `*tgz` file and view the new directory's contents with the following command:

```
$ tar -xvzf methMolBiol2014_get_homologues.tgz
&& ls
```

### 3.2 Installing GET\_HOMOLOGUES and Its External Dependencies

After downloading the package from the closest of the abovementioned mirrors you will have to unzip and unpack it, change into the `get_homologues/` directory, and launch the install script with the following command issued from your terminal:

```
$ tar xvfz get_homologues_X.Y.tgz; cd get_
homologues_X.Y; ./install.pl
```

Note that `_X.Y` has to be changed to the actual distribution version you downloaded. Please follow the indications provided by the installation script in case some required dependency is missing. They should be enough to assist you with the installation of dependencies. The protocols presented below require a full installation of the external dependencies, which includes R and the latest version of the Pfam-A database [24]. Read “Subheading 2” of the manual (bundled with the distribution) if you need additional help on the installation process.

### **3.3 Computing Orthologous Gene Clusters for *plncA/C* Plasmids Using the BDBH Algorithm Under Default Settings**

We are now set to proceed with the actual calculations. The aim of this protocol is to compute orthologous gene clusters or families using the main script `get_homologues.pl` and its default clustering method (BDBH) under default parameter values. This is intentionally kept simple in order to focus the reader’s attention on the basic computational steps involved in the whole process. Make sure that you are working in the parental directory (one directory above) of `pIncAC/`, the directory in which we stored the GenBank files (**step 1** of Subheading 2). To display the program’s help menu simply type (*see Note 2*)

```
$ cd $top_dir
$ get_homologues.pl
```

Let us start by running a standard BDBH analysis with default parameter values (75 % pairwise alignment coverage [`-C 75`], E-value=1e-05 [`-E 1e-05`], using two threads or cores [`-n 2`] and retaining only clusters that contain at least one representative protein from each proteome analyzed [`-t number_of_proteomes`], running the analysis on the local machine [`-m local`]). This is as simple as issuing the following command from your terminal prompt:

```
$ get_homologues.pl -d pIncAC
```

The `get_homologues.pl` script will start extracting the CDSs from the GenBank files to generate replicon/genome-specific multi-FASTA files at the protein level (their proteomes; *see Note 3*), with sequences uniquely numbered to allow reusing of results if new proteomes are added. These are copied into a new directory named as the directory with the source GenBank files plus a “\_homologues” suffix (`pIncAC_homologues/` in our example). The script will then use these FASTA-formatted proteomes to generate blast databases by automatically calling `makeblastdb` from the `blast+package` [7]. Next, `blastp` will be called to make an all-against-all blast search, splitting jobs among the available threads. If your computer has more cores, you can use `-n <no_of_cores_to_use>` to speed up the process. In preparation for identifying bidirectional best-hits (BDBHs), the individual pairwise blast results are concatenated and sorted, so that all hits of a query are grouped together and ranked in terms of E-value.

Note that the BDBH algorithm requires a reference genome. If none is specified, `get_homologues.pl` will automatically select the smallest input file as the reference. The sorted blast table, which can be quite large, is then parsed in order to calculate alignment lengths, also managing hits with several multiple high-scoring segments. The resulting file is indexed for faster posterior data access, storing the first and last hits of every query. The algorithm starts by finding inparalogues [25] in the reference genome. These are operationally defined as bidirectional BDBHs found within the same genome from which the query protein derives, that is, better within-genome hits (obviously excluding the query protein itself) than those found in any other genomes included in the analysis. The inparalogues of a second proteome are labeled next, before identifying BDBHs between the reference genome and this second one. This process is repeated until all non-reference genomes were compared with the reference one, as depicted in Fig. 3 of the manual. All BDBHs found outside the reference genome for a particular protein are added to a cluster, labeled according to the reference protein name and written to disk. Note that these clusters will contain at least one representative of each proteome. A cluster that contains more members (proteins) than the number of proteomes compared indicates the presence of inparalogues in at least some non-reference proteomes. The BDBH clusters are all saved in a directory named in a fashion that makes it easy to identify the clustering algorithm and associated parameters used for that particular analysis. For example

```
EscherichiacolistrainSCEC2plasmidpSCEC2NC0223
77_f0_alltaxa_algBDBH_e0_
```

indicates the name of the reference genome, that no % length difference within cluster filtering was applied (`_f0_`), and that only clusters containing at least one member from all proteomes analyzed are considered. In our case that means that all clusters contain at least 12 protein sequences, one from each original proteome. Note that equivalent clusters of DNA sequences are also produced from input files if they are in GenBank format. These are therefore orthologous gene clusters, as defined by the BDBH algorithm. The flag `_e0_` indicates that clusters with inparalogues were allowed (default behavior). How can we find out how many orthologous gene clusters were found and the number of protein sequences each one contains? This is easy to answer using basic shell filtering commands. Let us first change into the directory (`cd`) containing the blast results (`pIncAC_homologues/`) and explore its contents by issuing the following commands (lines preceded with a hash symbol are simply comments that are ignored by the shell command interpreter):

```
# cd into blast results directory and save its
path in the variable $blast_dir
```

```

$ cd pIncAC_homologues
$ blast_dir=$(pwd)
# explore contents by file extension names
$ ls | cut -d\ . -f2 | sort | uniq -c
    1 cluster_list
    1
EscherichiacolistrainSCEC2plasmidpSCEC2NC02
2377_f0_alltaxa_algBDBH_e0_
    216 gbk
    1 tmp
    1 txt
# find which of those files are directories
$ find . -type d
    ./tmp
    ./EscherichiacolistrainSCEC2plasmidpSCEC2NC
022377_f0_alltaxa_algBDBH_e0_

```

Take some time to explore the contents of the different files. Due to space constraints we cannot explain the contents of all the intermediary files herein, but more information can be found in the manual. So lets cd into the directory containing the BDBH orthologous clusters obtained by running `get_homologues.pl` under default settings to explore the results in greater detail. Note that the output of some of the commands is truncated or not shown, in order to save space and trees:

```

# cd into the BDBH clusters directory (default
BDBH clusters)
$ cd EscherichiacolistrainSCEC2plasmidpSCEC2NC02
377_f0_alltaxa_algBDBH_e0_
# list contents (orthologous gene clusters) and
count them
$ ls
1238_repA.faa 1270_hypothetical_protein.faa
1241_putative_signal_peptide_peptidase_SppA.faa
1271_dsbc.faa
1242_DsbA-like_thioredoxin_domain_protein.faa
1287_protein_YbaA.faa
... output cut to save trees
$ ls | wc
    23 23 684

```

```

# how many genes does each orthologous cluster
contain?
$ grep -c '>' *faa
1238_repA.faa:12
1241_putative_signal_peptide_peptidase_SppA.
faa:12
1242_DsbA-like_thioredoxin_domain_protein.
faa:12
... output truncated
#which clusters contain inparalogues (in our
case > 12 sequences)?
$ grep -c '>' *faa | grep -v ':12'
1270_hypothetical_protein.faa:13
1271_dsbc.faa:13

```

The result of issuing these commands is that we found 23 clusters of orthologous proteins among the 12 plasmid proteomes, two of which contain 13 sequences (one cluster contains an inparalogue) and the remaining 21, twelve proteins, one from each source proteome. The question to answer now is the following: Which plasmids contain the loci with inparalogues?

```

# which plasmid proteome contains the locus with
inparalogues
# for orthologous cluster 1270_hypothetical_
protein.faa?
$ grep '>' 1270_hypothetical_protein.faa | cut
-d\| -f2,3 | sort | uniq -c
  1 [Aeromonas hydrophila] |
  1 [Escherichia coli]|APEC1990_61
  1 [Escherichia coli]|AR060302
  1 [Escherichia coli]|H4H
  1 [Escherichia coli]|NDM-1 Dok01
  1 [Escherichia coli]|PG010208
  1 [Escherichia coli]|SCEC2
  1 [Escherichia coli UMNK88]|UMNK88
  1 [Klebsiella pneumoniae] |
  1 [Klebsiella pneumoniae]|Kp7
  2 [Salmonella enterica]|AM04528
  1 [Salmonella enterica subsp. enterica
serovar Kentucky]|1643/10

```

That output reveals that the proteome of *Salmonella enterica* AM04528 is the one which contains two copies (inparalogues) for cluster 1270. Repeat the exercise for cluster 1271.



### 3.4 Computing Orthologous Gene Clusters for *plncA/C* Plasmids Using the BDBH Algorithm Imposing Homogeneous Pfam Domain Composition on Cluster Members

The default parsing parameters for blast results are quite stringent, imposing 75 % pairwise alignment coverage [-C 75] and an E-value value cutoff=1e-05 [-E 1e-05]. Depending on the divergence of the dataset to be analyzed, these parameters may be relaxed (divergent set) or made more stringent (within species). A less arbitrary and very powerful means of selecting *bona fide* orthologous clusters is imposing the restriction that all members have the same Pfam domain composition [24]. Due to the relatively tight link that exists between protein domain architecture and function, this restriction makes the resulting clusters more likely to contain functionally equivalent proteins [26]. This can be easily performed calling the `get_homologues.pl` script with the `-D` option, as shown below:

```
# Generate BDBH clusters containing proteins
with conserved Pfam domain composition

$ cd $stop_dir

$ nohup get_homologues.pl -d pIncAC -D &> log.
get_homologues_pIncAC_BDBH_C75D_allTaxa &

$ tail -f log.get_homologues_pIncAC_BDBH_C75D_
allTaxa
```

For a brief explanation of the additional shell commands and syntax used in this command line *see Note 4*. The `-D` option calls the Pfam-based HMMER domain scanning function implemented in GET\_HOMOLOGUES (*see Note 5*). Each protein from each source FASTA file will be scanned with `hmmScan` using the Pfam-A domain database [24]. The results are concatenated and parsed, generating a file containing strings of domain composition and order for each protein of all proteomes.

The `get_homologues.pl` script will notice that we are running a new analysis on the same input dataset and will therefore reuse as much of the previous calculations as possible. In this case, the script will reuse the all-versus-all `blastp` results from the previous run. However, the blast results are newly parsed, now taking into account the domain composition of the reciprocal best hits in order to construct the orthologous clusters. The new clustering results are saved in its own directory, named with a `_Pfam_` suffix, as shown below:

```
# cd into the Pfam-domain filtered BDBH cluster
directory

$ cd $blast_dir

$ cd

EscherichiacolistrainSCEC2plasmidpSCEC2NC
022377_f0_alltaxa_algBDBH_Pfam_e0_

# list contents (orthologous gene clusters) and
count them

$ ls && ls | wc
```

```

1238_repA.faa      1259_N-6_DNA_Methylase_family_
protein.faa
1298_traF.faa
    1241_putative_signal_peptide_peptidase_SppA.
faa 1260_hypothetical_protein.faa 1299_traH.faa
... output truncated
    22 22 658
#which clusters contain inparalogues
$ grep -c '>' *faa | grep -v ':12'
1270_hypothetical_protein.faa:13

```

Repeating similar commands as shown in the previous section we find that this new BDBH analysis uncovers 22 orthologous clusters (vs. 23 in the previous one), only one of which has 13 proteins (i.e., contains an inparalog). So the question to answer now is the following: Which are the clusters from the standard BDBH analysis that do not contain a homogeneous Pfam domain composition? This can be easily answered with the following shell commands:

```

# generate two files listing the clusters found
by the standard and Pfam-domain filtered BDBH
clusters
$ ls *faa > Pfam_filtered_BDBH_clusters.list
$ ls
../EscherichiacolistrainSCEC2plasmid
pSCEC2NC022377_f0_alltaxa_algBDBH_e0_/*faa | \
sed
's#../EscherichiacolistrainSCEC2plasmi
dpSCEC2NC022377_f0_alltaxa_algBDBH_e0_###' \
> standard_BDBH_clusters.list
# find the difference between the two lists
$ diff standard_BDBH_clusters.list
Pfam_filtered_BDBH_clusters.list | grep '<'
< 1262_topB.faa
< 1271_dsbc.faa
< 1293_site-specific_recombinase-_
phage_integrase_family.faa

```

This result demonstrates the higher stringency of the Pfam domain-composition filtering strategy. It also suggests that 1270\_hypothetical\_protein.faa may be a true inparalogue that has recently been duplicated, without changing its Pfam domain composition and ordering.

If the user wishes to obtain orthologous BDBH gene clusters containing only single-copy genes, any of the previous `get_homologues.pl` commands could have been expanded with the `-e` flag, which excludes clusters with inparalogues. We leave this exercise for the reader.

**3.5 Computing  
a Robust Strict Core  
Genome  
and the Corresponding  
Clusters  
of Orthologous Gene  
Clusters  
with Homogeneous  
Pfam-Domain  
Composition  
for *pIncA/C* Plasmids  
Using the Intersection  
Between BDBH, COG,  
and OrthoMCL Gene  
Families**

We have recently shown that the definition of orthologous clusters and their composition are variable depending on the clustering method used [5, 27]. Technical details aside, it is clear that the most robust orthologous gene clusters would be those recognized by all three clustering algorithms currently implemented in GET\_HOMOLOGUES. We will now run `get_homologues.pl` sequentially, to obtain the COG and OrthoMCL clusters of any size by using the `-t 0` option (only valid for these two algorithms, but not for BDBH, since the latter requires that the reference genome is always present in the clusters). This option is required when we are interested in computing pan-genome sizes and the frequency distribution of pan-genomic cluster sizes, the pan-genome structure (note that by default `-t` is set to the number of all proteomes). The auxiliary script `compare_clusters.pl` can then be used to produce intersection pan-genome matrices, including the computation of consensus core genomes. We will also use the `-c` flag for genome composition analysis, that is, to obtain tables of re-sampled core- and pan-genome sizes which can be used by the auxiliary script `plot_pancore_matrix.pl` to fit Tettelin [3] or Willenbrock [28] exponential decay models to estimate core genome sizes, and the exponential Tettelin model [28] to get estimates and graphical plots of the pan-genome size. The next code snippets show the use of `get_homologues.pl` to call the three clustering algorithms combined with `compare_clusters.pl` to parse them in order to obtain consensus clusters. Make sure that you are just above the `pIncAC/` directory holding the GenBank files and issue the following command:

```
$ cd $stop_dir
$ nohup get_homologues.pl -d pIncAC -G -D -t 0
-c &> log.get_homologues_pIncAC_GDt0c && get_
homologues.pl -d pIncAC -M -D -t 0 -c &> log.
get_homologues_pIncAC_MDt0c && get_homologues.
pl -d pIncAC -D -c &> log.get_homologues_pIncAC_
BDBH_Dc &
```

This command will sequentially call the main script `get_homologues.pl` to run the COG, OrthoMCL, and BDBH algorithms under stringent conditions of homogeneous Pfam-domain composition (`-D`), reporting core- and pan-genome composition (`-c`), and in the case of the former two clustering methods, reporting clusters of all sizes (`-t 0`) (*see Note 6*). Note that running two jobs simultaneously on the same input directory might produce unexpected results, so it is not encouraged.

This will run very quickly, as we have already performed all `blastp` runs and Pfam-based `hmmsearch` searches for domain composition. We are now ready to use the auxiliary `compare_clusters.pl` script that will read the contents of the three directories containing the BDBH, COG, and OrthoMCL clustering results to compute the consensus single-copy orthologous gene families, by using the following code snippet:

```
# generate the consensus single-copy orthologous gene clusters with compare_clusters.pl
$ cd $blast_dir
$ compare_clusters.pl -d
EscherichiacolistrainSCEC2plasmidpSCEC2NC022377_f0_0taxa_algCOG_Pfam_e0_,EscherichiacolistrainSCEC2plasmidpSCEC2NC022377_f0_0taxa_algOMCL_Pfam_e0_,EscherichiacolistrainSCEC2plasmidpSCEC2NC022377_f0_alltaxa_algBDBH_Pfam_e0_ -o intersect_core_BCM_Dt12 -t 12 -m
```

The `-d` option is used to pass the script the names of the three directories containing the source clusters. Option `-o` is required to provide an output directory to hold the resulting cluster information, the corresponding FASTA files, and a PDF file with a Venn diagram showing the results of the parsing analysis. Option `-t 12` tells the script to report only the clusters with the indicated number of proteomes (all in our case). The following code snippets show how to explore the contents of the newly generated results directory which we have named `intersect_core_BCM_Dt12/`

```
# cd into the intersect_core_BCM_Dt12 directory and explore its contents
$ cd intersect_core_BCM_Dt12 && ls && ls *faa | wc
1238_repA.faa1297_uvrD-REP_helicase_N-terminal_domain_protein.faa
1241_putative_signal_peptide_peptidase_SppA.faa
1298_traF.faa
1242_DsbA-like_thioredoxin_domain_protein.faa
1299_traH.faa
... output truncated
    18 18 553
# confirm that all 18 clusters contain only one sequence from each plasmid/proteome
$ grep '>' *faa | cut -d\| -f2,3 | sort | uniq -c
    18 [Aeromonas hydrophila]|
    18 [Escherichia coli]|APEC1990_61
    18 [Escherichia coli]|AR060302
    18 [Escherichia coli]|H4H
```

```

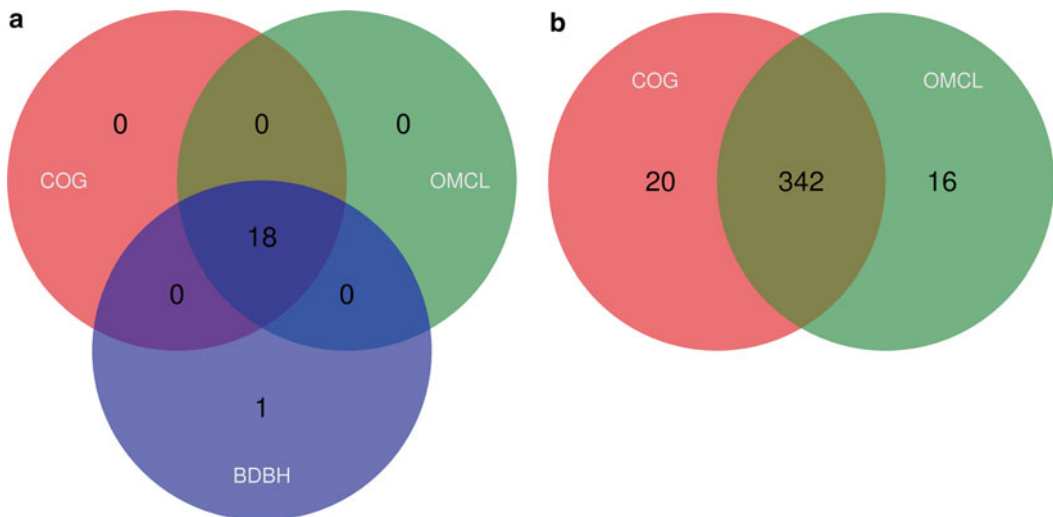
18 [Escherichia coli]|NDM-1 Dok01
18 [Escherichia coli]|PG010208
18 [Escherichia coli]|SCEC2
18 [Escherichia coli UMNK88]|UMNK88
18 [Klebsiella pneumoniae]|
18 [Klebsiella pneumoniae]|Kp7
18 [Salmonella enterica]|AM04528
18 [Salmonella enterica subsp. enterica
serovar Kentucky]|1643/10

```

This quick analysis shows that there are 18 consensus orthologous clusters, each having a single sequence from each plasmid/proteome. Figure 1a shows the results of a Venn analysis of the composition of the clusters generated by each of the three clustering algorithms. This figure shows that only the BDBH algorithm detected an additional cluster, as we have learned in previous sections.

**3.6 Computing Robust Consensus Pan-Genome Clusters as the Intersection of Homologous Gene Clusters Generated by the COG and OrthoMCL Algorithms, with Pfam-Based Domain Scanning**

This exercise is similar to the previous one, except that here we are interested in defining a consensus pan-genome, that is, the set of clusters of any size consistently detected by the COG and OrthoMCL algorithms with Pfam-based domain scanning. To do so we will call the auxiliary `compare_clusters.pl` script with the `-t 0` option, which, as stated before, can only be used with these clustering algorithms, which do not require a reference genome to be included in each cluster. For this very reason they are better suited for computing the pan-genome cluster composition and hence statistically estimate its theoretical size. Issue the



**Fig. 1** Venn analyses of the consensus core- (a) and pan-genomes (b) computed from the intersection of the clusters found by the indicated algorithms

following command from the `pIncAC_homologues/` directory to get the results:

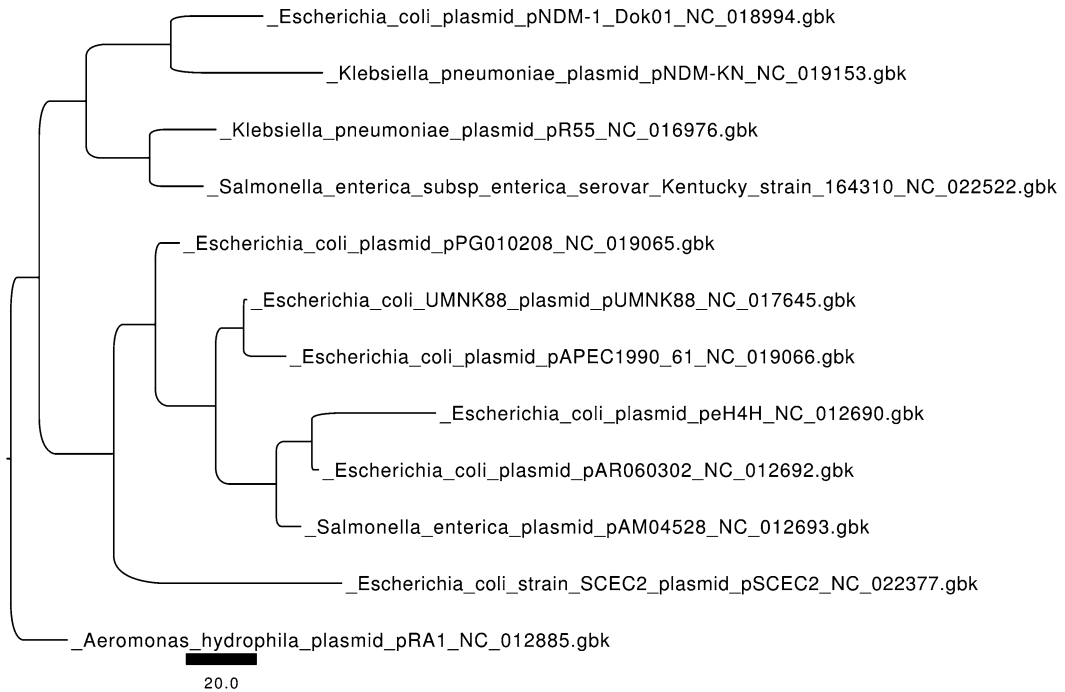
```
$ cd $blast_dir
[$ compare_clusters.pl -d
EscherichiacolistrainSCEC2plasmidpSCEC2NC0223
77_f0_0taxa_algCOG_Pfam_e0_,Escherichiacolistr
ainSCEC2plasmidpSCEC2NC022377_f0_0taxa_algOMCL_
Pfam_e0_ -o intersect_pan_CM_Dt0 -t 0 -m -T &>
log.comp_clusters_intersect_pan_CM_Dt0 &
```

Note that here we are redirecting the script's output to a file named `log.comp_clusters_intersect_pan_CM_Dt0` for later inspection. Notice also the use of the `-m` flag to tell the script that we want it to compute the pan-genome matrix. This is a table containing the presence-absence data for each gene (columns) and proteome/genome (rows). If R [29] is installed on the system, the script will run a Venn analysis and generate the corresponding Venn diagram, shown in Fig. 1b.

From the output saved in `log.comp_clusters_intersect_pan_CM_Dt0` we can see that the COG algorithm yielded 362 pan-genomes clusters, OrthoMCL 358, and 342 were predicted by both as graphically represented in Fig. 1b (*see Note 7*). The pan-genome matrix is also provided in PHYLIP format, which can then be used by `parse` (bundled with the `GET_HOMOLOGUES` package) from the PHYLIP package [30] to compute pan-genomic parsimony trees, as we have shown previously [5, 27] and detailed in the `GET_HOMOLOGUES` manual. Using the `-T` flag will do this automatically. Figure 2 shows such a pan-genomic parsimony tree depicting the relationships among the 12 pIncA/C plasmids based on the presence-absence matrix of homologous gene clusters. That is, this phylogeny depicts the phylogenetic relationships among plasmids based on their gene content.

### **3.7 Statistical Estimation of the Theoretical Core- and Pan-Genome Sizes by Fitting Exponential Models (Tettelin and Willenbrock)**

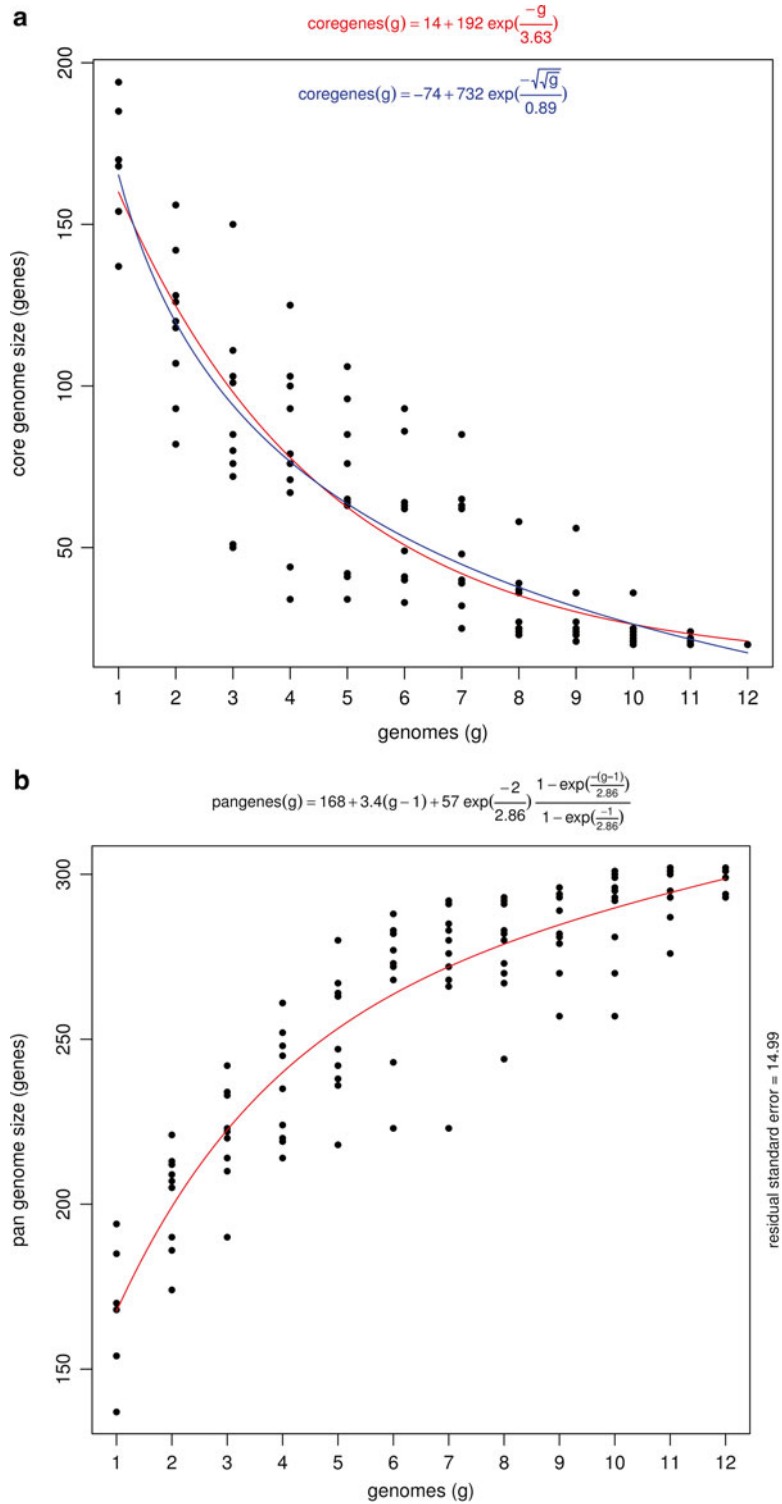
Other features of the `GET_HOMOLOGUES` package that we want to demonstrate herein are its graphical and statistical capabilities, which are based on the powerful statistical and graphical computing environment R [29]. You may recall that in Subheading 3.5 we ran `get_homologues.pl` with the `-c` option enabled. As we will show now, this had the effect of generating three tab-delimited text files called `core_genome*tab` and `pan_genome*tab` found in the `pIncAC_homologues/` directory, where `*` stands for the clustering algorithm used to generate them. These files contain the results of ten sampling experiments, in which genomes are randomly ordered and sequentially added to the pan-genome pool, keeping track of novel genes contributed by each genome (pan) and those already found in previous clusters (core), a strategy first introduced by Tettelin and colleagues in their seminal work on



**Fig. 2** Pan-genome tree depicting the relationships among plncA/C plasmids based on the presence–absence pan-genome matrix. The phylogeny was recovered under standard Fitch parsimony and rooted in the reference pRA1 plasmid found in *Aeromonas hydrophila*, a non-enteric gamma-proteobacterium (*Aeromonadales*, *Aeromonadaceae*) strain recovered as a fish pathogen

*Streptococcus* pan-genomics [3]. These tables can be read by the auxiliary script `plot_pancore_matrix.pl`, which will convert them to R data frames to fit the exponential models of Tettelin et al. [15] and Willenbrock et al. [28]. These models are used to estimate the theoretical size of the core and pan-genomes. The following commands will fit the models and generate the files corresponding to the core- and pan-genome graphs, which are shown in Fig. 3a, b:

```
# find the names of the pancore tab files in
pIncAC_homologues/
$ ls *tab
core_genome_algBDBH_Pfam.tab core_genome_algOMCL_
Pfam.tab pan_genome_algCOG_Pfam.tab
core_genome_algCOG_Pfam.tab pan_genome_algBDBH_
Pfam.tab pan_genome_algOMCL_Pfam.tab
# visualize the contents of the core and pan-
genome size files
# obtained by randomly sampling 10 genomes based
on OMCL clustering
```



**Fig. 3** Statistical estimation and graphical display of core-genome (a) and pan-genome (b) sizes obtained by fitting exponential functions [3, 28] to resamplings of the core- and pan-genome clusters



```

$ for file in *OMCL*Pfam.tab; do echo "# $file";
cat $file; echo; echo; done
# core_genome_algOMCL_Pfam.tab
g1    g2    g3    g4    g5    g6    g7    g8
g9    g10   g11   g12
154   107   101   100   96    93    65    25
21    21    20    20
168   129   87    61    53    39    38    38
38    38    21    20
161   65    57    55    41    41    24    24
22    22    21    20
... output truncated
# pan_genome_algOMCL_Pfam.tab
g1    g2    g3    g4    g5    g6    g7    g8
g9    g10   g11   g12
154   205   210   219   267   272   272   273
279   293   293   299
168   245   252   256   268   287   287   288
296   297   298   301
... output truncated
# use the *tab files computed based on the OrthoMCL
clustering results to fit
# both the Tettelin and Willenbrock exponential
decay functions to the core genome
# resampling data.
$ plot_pancore_matrix.pl -i core_genome_algOMCL_
Pfam.tab -f core_both

```

Note that due to the random sampling of the proteomes performed to compute the core- and pan-genome sizes, the actual output you get may be somewhat different, particularly in the first columns. The script also generates log files with the details of the statistical analysis. As an example, let us inspect one such file:

```

$ less core_genome_algOMCL.tab_core_both.log
# core_Tettelin fit converged
# residual standard error = 18.33
~ coregenes(g) == "14" + "192" exp(frac(-g,
"3.63"))...
output truncated
# core_Willenbrock fit converged
# residual standard error = 17.88

```

```
~ coregenes(g) == "-74" + "732" exp(frac(-
sqrt(sqrt(g)), "0.89"))
... output truncated
```

Based on the residual standard error, these results show that the Willenbrock model has a slightly better fit than the Tettelin model for this dataset.

### **3.8 Fitting Mixture Models to Estimate Pan-Genome Sizes and Graphical Analysis of the Pan-Genome Structure**

The exponential models fitted to the core- and pan-genome re-sampling data demonstrated in Subheading 3.7 have been criticized by some authors [14], based on two objections: (1) Exponential models implicitly assume an infinite size for “open” pan-genomes [3, 15] and (2) they also imply that the pan-genome structure basically consists of two “compartments,” the universally distributed core-genome genes and the less conserved “accessory genes” that conform the “flexible genome.” Although the gene pool available to species with open pan-genomes is certainly impressively large [31], it is not realistic to assume that it is infinite [14]. Further, large-scale comparative genomics studies have consistently revealed that the structure of the microbial pan-genome has certainly more classes than just the core and flexible components [32]. In the latter class the frequency distribution of the taxa in homologous gene clusters varies strongly, but in their seminal work, Koonin and Wolf [32] show that on a coarse scale, the flexible components can be grouped in the shell and cloud components, the latter corresponding to genes present in very few proteomes/genomes of those analyzed.

The auxiliary script `parse_pangenome_matrix.pl` was designed to analyze the structure of the pan-genome, computing and plotting the strict core, relaxed core, shell, and cloud components of the pan-genome. The command lines shown below will illustrate the usage of the `parse_pangenome_matrix.pl` script to graphically explore the structure of the pan-genome of pInCA/C plasmids using the consensus COG and OrthoMCL clusters with Pfam-domain filtering computed in Subheading 3.6. We move into the `intersect_pan_CM_Dt0/` directory and issue the following command:

```
#first cd into the dir holding the consensus COG-
OrthoMCL pangenome
$ cd intersect_pan_CM_Dt0
# Fit mixture model and plot core-cloud-shell
pan-genome composition graphics
# saving the output to the file pan-genome_struc-
ture_analysis.out
$ parse_pangenome_matrix.pl -m pangenome_matrix_
t0.tab -s &> pan-genome_structure_analysis.out
```

The script returns the following files:

```
# find the output files just generated by the script
$ ls -ltr
pangenome_matrix_t0__softcore_list.txt
pangenome_matrix_t0__shell_list.txt
pangenome_matrix_t0__shell_input.txt
pangenome_matrix_t0__core_list.txt
pangenome_matrix_t0__cloud_list.txt
pangenome_matrix_t0__shell_estimates.tab
pangenome_matrix_t0__shell_circle.png
pangenome_matrix_t0__shell_circle.pdf
pangenome_matrix_t0__shell.png
pangenome_matrix_t0__shell.pdf
pan-genome_structure_analysis.out
```

Let us explore the `pangenome_matrix_t0_*_list.txt` files to find both conserved and plasmid-specific genes. In the first category we would expect for example to find the plasmid replication and mobilization genes (*rep* and *tra*). The following code will do the job:

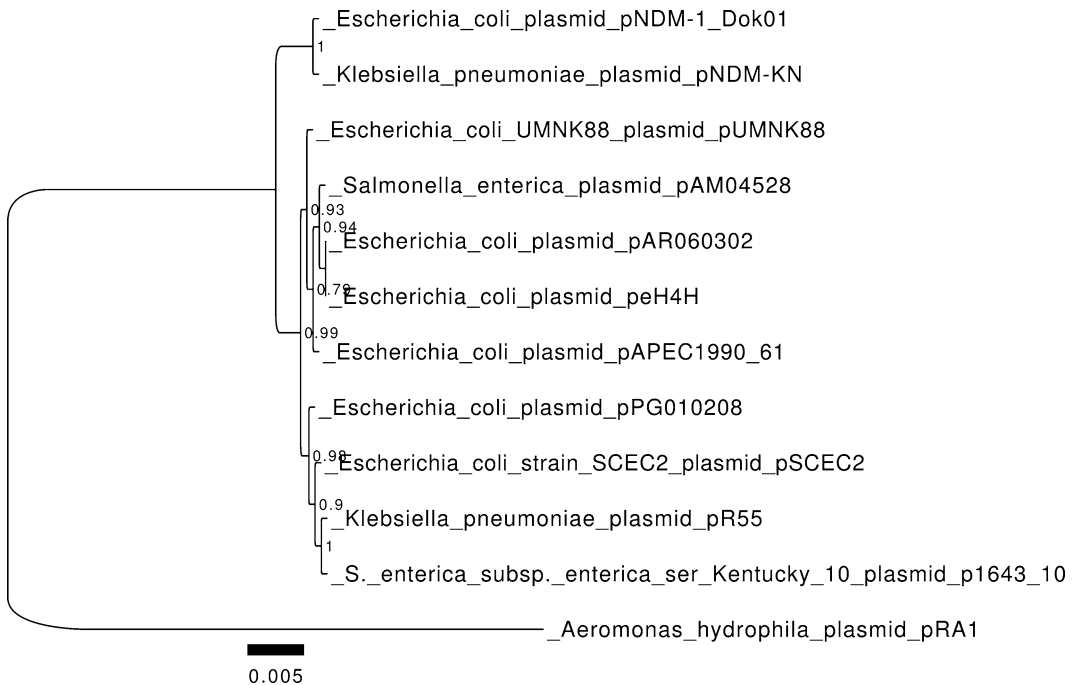
```
# inspect the pangenome_matrix_t0_*_list.txt for
the presence plasmid replication and mobilization
genes
$ egrep 'mob|tra|rep' pangenome_matrix_t0*.txt |
egrep -v 'transpo|transcr|trans' | grep core.list
pangenome_matrix_t0__core_list.txt:1238_repA.faa
pangenome_matrix_t0__core_list.txt:1295_DNA_
replication_terminus_site-binding-_Ter_protein.
faa
pangenome_matrix_t0__core_list.txt:1298_traF.faa
pangenome_matrix_t0__core_list.txt:1299_traH.faa
pangenome_matrix_t0__core_list.txt:1300_traG.faa
pangenome_matrix_t0__softcore_list.txt:1263_
traI.faa
pangenome_matrix_t0__softcore_list.txt:1264_
traD.faa
pangenome_matrix_t0__softcore_list.txt:1268_
traB.faa
pangenome_matrix_t0__softcore_list.txt:1269_
traV.faa
pangenome_matrix_t0__softcore_list.txt:1272_
traC.faa
```

```

pangenome_matrix_t0__softcore_list.txt:1273_
traF.faa
pangenome_matrix_t0__softcore_list.txt:1274_
traW.faa
pangenome_matrix_t0__softcore_list.txt:1276_
traU.faa
pangenome_matrix_t0__softcore_list.txt:1277_
traN.faa
... output cut.

```

As expected, most of these genes are part of the core-genome, although some are also part of the shell-genome. There are practical implications for defining such a set of bona fide core-genome sequences. They could for example be used (at the DNA level) to design degenerate PCR primers for the detection, typing, and phylogenetic analysis of pIncA/C plasmids. This task could be very easily performed with the `primers4clades` web server [33, 34]. Another key use of this set of proteins is for phylogenetic analysis



**Fig. 4** Maximum likelihood phylogeny of plncA/C plasmids based on the concatenation of the 18 consensus core-genome computed from the intersection of BDBH, COGtriangles, and OMCL clusters and Pfam domain-scanning enabled. The tree search was performed under the LG matrix with empirical frequencies + proportion of invariant sites + gamma correction of among-site rate variation using the BEST move in PhyML3

to unravel the evolutionary relationships between the plasmids under study and infer the evolutionary pathways that have shaped the final replicons, including the gain and loss of gene clusters. Figure 4 shows a maximum likelihood phylogeny inferred from the concatenation of the 18 strict core loci (*see Note 8*).

Now let us interrogate the lists to search for some interesting and famous antimicrobial resistance genes, like beta-lactamases and tetracycline resistance genes (*bla* and *tet* genes):

```
# inspect the pangenome_matrix_t0__*_list.txt
for the presence of bla or tet genes
$ egrep 'bla|lactamase|tet|tetracycline'
pangenome_matrix_t0__*_list.txt
pangenome_matrix_t0__cloud_list.txt:546_tetA.faa
pangenome_matrix_t0__cloud_list.txt:867_
blaNDM-1.faa
pangenome_matrix_t0__cloud_list.txt:870_blaTEM-1.
faa
pangenome_matrix_t0__cloud_list.txt:1611_
blaCXA-21.faa
pangenome_matrix_t0__shell_list.txt:1252_tetA.
faa
pangenome_matrix_t0__shell_list.txt:1253_tetR.
faa
```

As expected, the antibiotic resistance genes are part of the cloud and shell gene pools.

Let us now inspect the output from the script, which was redirected to the `pan-genome_structure_analysis.out` file. Files in Linux or Unix systems can be viewed for example with `less pan-genome_structure_analysis.out`. We will focus on the mixture-model analysis section, which is displayed below:

```
# pan-genome size estimates (Snipen mixture
model PMID:19691844): pangenome_matrix_t0__
shell_estimates.tab
Core.size Pan.size BIC LogLikelihood
2 components 19 343 2836.97081559449 -1409.73319
169165
3 components 12 401 1600.53932337316 -785.682634
843925
4 components 0 475 1516.28702925272 -737.721677
046643
5 components 0 484 1528.05926969358 -737.7729865
30008
6 components 0 500 1540.93353594771 -738.375308
92001
```

```

7 components 0 482 1551.33188934561 -737.7396
748819
8 components 0 478 1563.60197249279 -738.039905
718426
9 components 0 472 1574.81631547103 -737.8122664
70482
10 components 0 434 1596.0860933087 -742.6123446
52257

```

Based on the Bayesian Information Criterion (BIC) of the different components (second column from the right), this analysis shows that the best fit corresponds to a model with 4 components (as it has the lowest BIC value), followed by that with 5 components, at a distance of 11.7 AIC units (*see Note 9*). This analysis therefore strongly suggests that there are more than just two pan-genome components, which is consistent with the graphical analysis of cluster-size frequency distribution shown in Fig. 5a, b. The size of the pan-genome is estimated to be around 475 genes. The consensus core-genome size is estimated to be much smaller, around 0 genes, which clearly seems a strong underestimation. These results highlight the importance of refining all models to find more realistic and useful core- and pan-genome size estimates.

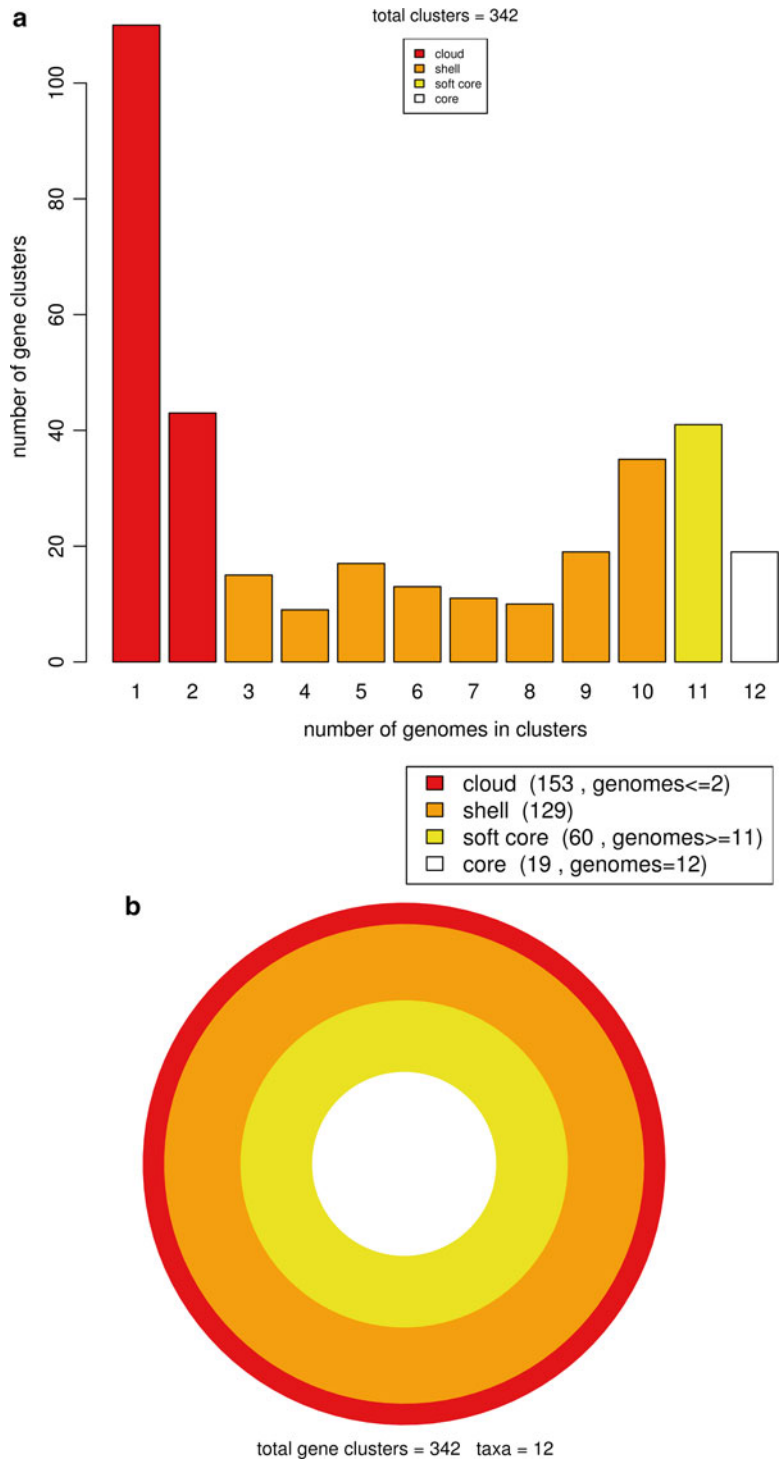
### 3.9 Identification of Lineage-Specific Genes in Consensus Pan-Genome Matrices Using `parse_pangenome_matrix.pl`

The `parse_pangenome_matrix.pl` script was designed to perform basic comparative genomics tasks. It can be used to compare two pan-genome sets to identify lineage-specific genes and lineage-specific gene expansions in one subset (A), as compared to the other one (B). From the inspection of the `pangenome_matrix_t0_cloud_list.txt` file we did in the previous section, we found that the *bla*<sub>NDM</sub> genes were part of the cloud-genome. It is trivial to find the plasmids that contain them, using the following `grep` command:

```

# find the plasmids containing the NDM-1 genes
$ grep '>' 867_blaNDM-1.faa
>GI:410502926 |[Escherichia coli]|NDM-1 Dok01|
blaNDM-1|NA|NC_018994(195560):139825-140637:
-1 ^,GeneID:13876866^ Escherichia coli plasmid
pNDM-1_Dok01, complete sequence.|neighbours:GI:
410502925(-1),GI:410502927(-1)|neighbour_
genes:hypothetical protein,IS903 transposase|
>GI:410656145 |[Klebsiella pneumoniae]|Kp7|NDM-
1|NA|NC_019153(162746):108108-108920:-1 ^,GeneID:
13914405^ Klebsiella pneumoniae plasmid pNDM-KN,
complete sequence.|neighbours:GI:410656144(-1),
GI:410656146(-1)|neighbour_genes:bleMBL,
insertion element ISKpn14|

```



**Fig. 5** Graphical analysis of the structure of the *plnC/A/C* pan-genome protein space. Panel **a** depicts a bar plot showing the absolute size frequencies of orthologous clusters as predicted by the OMCL algorithm. Panel **b** shows a circle plot depicting the relative sizes (cluster numbers) contained in the core, soft-core, shell, and cloud genomes

This makes clear that only two plasmids contain the genes. We can now generate two lists of plasmid genomes: list A will contain the names of the GenBank files containing the *bla*<sub>N<sub>DM</sub>-1</sub> genes, and list B the names of the rest of the files. Generate such lists with the following code, working within the directory holding the \*gbk files (pInC/):

```
# 1. Generate the lists of genomes to be compared
for lineage specific genes (in list A vs. B)
panGmat_dir=$(pwd)
cd $gbk_dir
$ ls *gbk | grep pNDM > listA_pNDB
$ ls *gbk | grep -v pNDM > listB_nonNDB
$ cd $panGmat_dir
```

Now we are ready to run to find the genes specific to the “A” list of plasmids:

```
# 2. parse the pangenome matrix file to find the
listA-specific genes
$ parse_pangenome_matrix.pl -A $gbk_dir/listA_
pNDM -B $gbk_dir/listB_nonNDB -g -m pangenome_
matrix_t0.tab -p _Escherichia_coli_plasmid_
pNDM1_Dok01_NC_018994
```

Now we can inspect the output file’s content to see how many and which are the genes that are found only in the pInC/A/C plasmids containing the bla<sub>N<sub>DM</sub></sub> genes:

```
$ cat
pangenome_matrix_t0__Escherichia_coli_plasmid_
pNDM1_Dok01_NC_018994_pangenes_list.txt
# genes present in set A and absent in B (19):
846_armA.faa
862_groES.faa
863_hypothetical_protein.faa
864_hypothetical_protein.faa
865_trpF.faa
866_hypothetical_protein.faa
867_blaNDM-1.faa
879_Rhs_family_protein.faa
883_Tn7-like_transposition_protein_A.faa
884_Tn7-like_transposition_protein_B.faa
885_Tn7-like_transposition_protein_C.faa
886_hypothetical_protein.faa
888_type_I_site-specific_deoxyribonuclease-
_HsdR_family.faa
```



```
889_hypothetical_protein.faa
890_hypothetical_protein.faa
891_putative_type_I_restriction-modification_
system_restriction_subunit.faa
892_hypothetical_protein.faa
893_type_I_restriction-modification_system-_M_
subunit.faa
894_hypothetical_protein.faa
```

The sequential numbering of several genes (862–867 and 883–894) suggests that most of the list “A”-specific genes are clustered in two regions. The first one, containing the blaNDM-1 gene, also contains the well-known proteins GroES and TrpF. The first one is a component of the GroEL-GroES chaperonin complex. The *groS* gene is one of a network of 93 genes believed to play a role in promoting the stress-induced mutagenesis (SIM) response of *E. coli* K-12 (for more details see <http://ecocyc.org/ECOLI/NEW-IMAGE?type=GENE&object=EG10600>). TrpF (synonym of TrpC) is a bifunctional phosphoribosylanthranilate isomerase/indole-3-glycerol phosphate synthase. It carries out the third and fourth steps in the tryptophan biosynthesis pathway (for more details see <http://ecocyc.org/ECOLI/NEW-IMAGE?type=GENE&object=EG11026>). It is certainly somewhat surprising to find these two genes on a resistance plasmid. Readers interested in more details about these interesting findings are referred to the original publications describing the two NDM plasmids used in this chapter [16, 19].

### 3.10 Conclusions and Perspectives

In this chapter we have demonstrated some of the capabilities of the GET\_HOMOLOGUES software, focusing in the detection of orthologs, the statistical evaluation and graphical analysis of the core- and pan-genome compartments, and the detection of lineage-specific genes in the pan-genome matrix. These features demonstrate the flexibility and robustness of the software, and highlight its ease of use. There are several other interesting features, such as the analysis of syntenic intergenic regions, the use of the synteny criterion to define orthologs, and the use of the BerkeleyDB system to trade speed for RAM when analyzing very large genomic datasets, which are well documented in the manual and have been published elsewhere [5, 27]. Altogether these features make GET\_HOMOLOGUES a useful, versatile, flexible, and powerful piece of software that allows nonspecialists to make rigorous and detailed analyses of microbial pan-genomics and comparative genomics. Future development of the software will focus on including more statistical analyses and expanding its graphical capabilities.

## 4 Notes

1. The set of 12 GenBank files used in this chapter were downloaded from NCBI's RefSeq database [6] and further processed using the following protocol. Point your browser to the URL <http://www.ncbi.nlm.nih.gov/nuccore/> and type the following query string into the text box: "incA/C[*text*] AND plasmid[*titl*] AND complete sequence[*titl*] AND 90000[SLEN]: 200000[SLEN] AND srcdb\_refseq\_known[PROP]." This will search for pIncA/C plasmids in NCBI's RefSeq database. The results are displayed in the summary format. In the upper right corner click "Send to ->File; Format ->Accession List" and save the list of RefSeq accession numbers to the working directory on your hard drive with the name `accNo.list`. To fetch the actual GenBank files cd into the directory holding your `accNo.list` file (we will use the directory name `pIncAC/herein`) and type the following shell one-liner on your command prompt (all in one line):

```
$ for acc in $(cat accNo.list); do accBase=$(cut
-d\. -f1); wget -c
```

```
ftp://ftp.ncbi.nlm.nih.gov/genomes/Plasmids/
${accBase}.gbk; done
```

This should fetch the desired GBK files. If you wish, you can rename those files with the file's DEFINITION line using the auxiliary shell script `rename_gbk_files_with_DEFINITION_line.sh`

These simple scripts are bundled with the `*tgz` file mentioned in **step 1** of Subheading **2**.

2. This is assuming that you have added the directory containing the distribution to your `PATH` variable (as explained in the manual bundled with the package). Otherwise you will need to precede the program name with the full path, like `$HOME/path/to/get_homologues_XXX/get_homologues.pl`.
3. `Get_homologues.pl` can also work with the genome's `faa` or `ffn` files, that is, the fasta files in for the CDSs in protein or nucleotide version, respectively. Please check the manual for all accepted combinations of input formats. It should be noted that specialized functionality like the extraction of orthologous intergenic spacers or the use of the synteny criterion to filter orthologs will not work here, as the software relies on the GenBank annotations to determine the identity of the neighboring genes. See the manual for more details.
4. The `nohup` (no hang-up) command allows a second command provided as argument to be executed even after you exit from a shell session. This is very useful when you are running large jobs on a server. You issue your command and can log out of the session

without killing your process. The `&> log.get_homologues_pIncAC_BDBH_C75D_allTaxa` & syntax tells the shell to redirect the standard output and standard error streams to the `log.get_homologues_pIncAC_BDBH_C75D_allTaxa` file, while the last ampersand asks the shell to run the whole process in the background. Finally, the `log.get_homologues_pIncAC_BDBH_C75D_allTaxa` command allows us to continuously follow the last ten lines of the growing log file. A CTRL-C will close (kill) the tail command to exit from it. Then execute the file instructions calling `bash` with the following command: `bash get_homol_batch_pIncAC.cmd`. After issuing this command, you can log out of your session, if you wish. The script will run in the background, calling `get_homologues.pl` sequentially to run the three clustering algorithms.

5. The latest version of Pfam-A domain database can be downloaded from the Sanger ftp site during the package installation process. The database will be automatically formatted with `hmmpress` during the installation process, making it ready to use (see the `db/directory` within your `get_homologues.X.Y./directory`).
6. It is convenient to save complex command lines like this to a file for later reference or even use them as a template to create similar commands for other datasets. Open an editor and type or paste the code reproduced below

```
nohup get_homologues.pl -d pIncAC -G -n 2 -t
0 &> log.get_homologues_pIncAC_Gn2t0 && get_
homologues.pl -d pIncAC -M -n 2 -t 0 -c &>
log.get_homologues_pIncAC_Mn26t0 && get_
homologues.pl -d pIncAC -n 2 &> log.get_homo-
logues_pIncAC_BDBHn2 &
```

and name the file `get_homol_batch_pIncAC.cmd`. The command file can then be executed with this simple line:

```
$ bash get_homol_batch_pIncAC.cmd.
```

7. We have found that the COGtriangles clustering algorithm will consistently generate a larger number of unique clusters than the OMCL algorithm [5]. Most of these COG-specific clusters are actually singletons, consisting of single or pairs of proteins that were not merged into a proper cluster because at least three proteins from distinct organisms/proteomes are required to form a COG triangle [9, 35].
8. The individual clusters were aligned using `muscle` as in Subheading 3.8 [36] under default parameter values with the following command (assumes that `muscle` is installed on the system and in `PATH`):

```
$ for file in *faa; do muscle < $file >
${file%faa}_musAln.FAA; done
```

The original ordering of the strains in the alignments was reestablished and the alignments concatenated. The concatenated

alignment was then subjected to a maximum-likelihood tree search using PhyML3 [37] under the LG model, estimating amino-acid frequencies, proportion of invariant sites, and the shape parameter of the gamma distribution to model among-site rate variation. The search was started from a BioNJ tree using the BEST moves algorithm. The tree was visualized and edited with FigTree [38].

9. With the R package “qpcR” it is very easy to compute Akaike weights. Simply generate a vector of AIC values, here called `AIC.vals`, and pass it to the function `akaike.weights()`. For more information, see for example <http://www.inside-r.org/packages/cran/qpcR/docs/akaike.weights>

The R commands and output are shown below:

```
# call library qpcR
> library(qpcR)

# create a vector with the AIC values, in this
# case the three best ones (those with 3, 4 and
# 5 components, respectively) from the mixture
# model analysis in section 3.8
> AIC.vals<-c(1600.53932337316, 1516.287029
25272, 1528.05926969358)

# pass the vector AIC.vals to the akaike.
# weights function.
> akaike.weights(AIC.vals)
$deltaAIC
[1] 84.25229 0.00000 11.77224
$rel.LL
[1] 5.068119e-19 1.000000e+00 2.777733e-03
$weights
[1] 5.054080e-19 9.972300e-01 2.770038e-03
```

The output on the last line shows that the model with four classes has a relative weight of >99 % and second best (five components) a marginal 0.027 %, making the four-class model the clear winner.

---

## Acknowledgements

We thank Romualdo Zayas, Víctor del Moral, and Alfredo J. Hernández at CCG-UNAM for technical support. We also thank David M. Kristensen and the development team of OrthoMCL for permission to use their code in our project. Funding for this work was provided by the Fundación ARAID, Consejo Superior de Investigaciones Científicas (grant 200720I038), DGAPA-PAPIIT UNAM-México (grant IN211814), and CONACyT-México (grant 179133).

## References

1. Pagani I, Liolios K, Jansson J et al (2012) The Genomes OnLine Database (GOLD) v. 4: status of genomic and metagenomic projects and their associated metadata. *Nucleic Acids Res* 40:D571–D579
2. Welch RA, Burland V, Plunkett G 3rd et al (2002) Extensive mosaic structure revealed by the complete genome sequence of uropathogenic *Escherichia coli*. *Proc Natl Acad Sci U S A* 99:17020–17024
3. Tettelin H, Masignani V, Cieslewicz MJ et al (2005) Genome analysis of multiple pathogenic isolates of *Streptococcus agalactiae*: implications for the microbial “pan-genome”. *Proc Natl Acad Sci U S A* 102:13950–13955
4. Mira A, Martin-Cuadrado AB, D'Auria G et al (2010) The bacterial pan-genome: a new paradigm in microbiology. *Int Microbiol* 13:45–57
5. Contreras-Moreira B, Vinuesa P (2013) GET\_HOMOLOGUES, a versatile software package for scalable and robust microbial pangenome analysis. *Appl Environ Microbiol* 79:7696–7701
6. Tatusova T, Ciufu S, Fedorov B et al (2014) RefSeq microbial genomes database: new representation and annotation strategy. *Nucleic Acids Res* 42:D553–D559
7. Camacho C, Coulouris G, Avagyan V et al (2009) BLAST+: architecture and applications. *BMC Bioinformatics* 10:421
8. Eddy SR (2009) A new generation of homology search tools based on probabilistic inference. *Genome Inform* 23:205–211
9. Kristensen DM, Kannan L, Coleman MK et al (2010) A low-polynomial algorithm for assembling clusters of orthologous groups from intergenomic symmetric best matches. *Bioinformatics* 26:1481–1487
10. Li L, Stoeckert CJ Jr, Roos DS (2003) OrthoMCL: identification of ortholog groups for eukaryotic genomes. *Genome Res* 13:2178–2189
11. Altenhoff AM, Dessimoz C (2012) Inferring orthology and paralogy. *Methods Mol Biol* 855:259–279
12. Kristensen DM, Wolf YI, Mushegian AR et al (2011) Computational methods for gene orthology inference. *Brief Bioinform* 12:379–391
13. Wolf YI, Koonin EV (2012) A tight link between orthologs and bidirectional best hits in bacterial and archaeal genomes. *Genome Biol Evol* 4:1286–1294
14. Snipen L, Almoy T, Ussery DW (2009) Microbial comparative pan-genomics using binomial mixture models. *BMC Genomics* 10:385
15. Tettelin H, Riley D, Cattuto C et al (2008) Comparative genomics: the bacterial pan-genome. *Curr Opin Microbiol* 11:472–477
16. Carattoli A, Villa L, Poirel L et al (2012) Evolution of IncA/C bla<sub>CMY</sub>(-2)-carrying plasmids by acquisition of the bla<sub>NDM</sub>(-1) carbapenemase gene. *Antimicrob Agents Chemother* 56:783–786
17. Fricke WF, Welch TJ, McDermott PF et al (2009) Comparative genomics of the IncA/C multidrug resistance plasmid family. *J Bacteriol* 191:4750–4757
18. Johnson TJ, Lang KS (2012) IncA/C plasmids: an emerging threat to human and animal health? *Mob Genet Elements* 2:55–58
19. Sekizuka T, Matsui M, Yamane K et al (2011) Complete sequencing of the bla<sub>(NDM-1)</sub>-positive IncA/C plasmid from *Escherichia coli* ST38 isolate suggests a possible origin from plant pathogens. *PLoS One* 6:e25334
20. Poirel L, Hombrouck-Alet C, Freneaux C et al (2010) Global spread of New Delhi metallo-beta-lactamase 1. *Lancet Infect Dis* 10:832
21. Nordmann P, Poirel L, Walsh TR et al (2011) The emerging NDM carbapenemases. *Trends Microbiol* 19:588–595
22. Poirel L, Bonnin RA, Nordmann P (2011) Analysis of the resistome of a multidrug-resistant NDM-1-producing *Escherichia coli* strain by high-throughput genome sequencing. *Antimicrob Agents Chemother* 55:4224–4229
23. Moellering RC Jr (2010) NDM-1 – a cause for worldwide concern. *N Engl J Med* 363:2377–2379
24. Finn RD, Tate J, Mistry J et al (2008) The Pfam protein families database. *Nucleic Acids Res* 36:D281–D288
25. Sonnhammer EL, Koonin EV (2002) Orthology, paralogy and proposed classification for paralog subtypes. *Trends Genet* 18:619–620
26. Forslund K, Pekkari I, Sonnhammer EL (2011) Domain architecture conservation in orthologs. *BMC Bioinformatics* 12:326
27. Vinuesa P, Contreras-Moreira B (2014) Pangenomic analysis of the *Rhizobiales* using the GET\_HOMOLOGUES software package. In: De Bruijn FJ (ed) *Biological nitrogen fixation 7*. Wiley/Blackwell, Hoboken, NJ
28. Willenbrock H, Hallin PF, Wassenaar TM et al (2007) Characterization of probiotic

- Escherichia coli* isolates with a novel pan-genome microarray. *Genome Biol* 8:R267
29. R Development Core Team (2012) R: a language and environment for statistical computing. <http://www.R-project.org>. Vienna, Austria
  30. Felsenstein J (2004) PHYLIP (phylogeny inference package). In: Distributed by the author. Department of Genetics, University of Washington, Seattle
  31. Kaas RS, Friis C, Ussery DW et al (2012) Estimating variation within the genes and inferring the phylogeny of 186 sequenced diverse *Escherichia coli* genomes. *BMC Genomics* 13:577
  32. Koonin EV, Wolf YI (2008) Genomics of bacteria and archaea: the emerging dynamic view of the prokaryotic world. *Nucleic Acids Res* 36: 6688–6719
  33. Contreras-Moreira B, Sachman-Ruiz B, Figueroa-Palacios I et al (2009) primers4clades: a web server that uses phylogenetic trees to design lineage-specific PCR primers for metagenomic and diversity studies. *Nucleic Acids Res* 37:W95–W100
  34. Sachman-Ruiz B, Contreras-Moreira B, Zozaya E et al (2011) Primers4clades, a web server to design lineage-specific PCR primers for gene-targeted metagenomics. In: de Bruijn FJ (ed) *Handbook of molecular microbial ecology I: metagenomics and complementary approaches*. Wiley/Blackwell, Hoboken, NJ, pp 441–452
  35. Tatusov RL, Koonin EV, Lipman DJ (1997) A genomic perspective on protein families. *Science* 278:631–637
  36. Edgar RC (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res* 32:1792–1797
  37. Guindon S, Dufayard JF, Lefort V et al (2010) New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of PhyML 3.0. *Syst Biol* 59: 307–321
  38. Rambaut A (2009) FigTree v1.4.0. Available from <http://tree.bio.ed.ac.uk/software/figtree/>