Roary user manual

By Andrew Page based on version 3.3.3 (2-Oct-2015)

Roary is a high speed stand alone pan genome pipeline, which takes annotated assemblies in GFF3 format (produced by Prokka (Seemann, 2014)) and calculates the pan genome. Using a standard desktop PC, it can analyse datasets with thousands of samples, something which is computationally infeasible with existing methods, without compromising the quality of the results. 128 samples can be analysed in under 1 hour using 1 GB of RAM and a single processor. To perform this analysis using existing methods would take weeks and hundreds of GB of RAM. Roary is not intended for meta-genomics or for comparing extremely diverse sets of genomes.

Citation and further details of the method

Andrew J. Page, Carla A. Cummins, Martin Hunt, Vanessa K. Wong, Sandra Reuter, Matthew T. G. Holden, Maria Fookes, Daniel Falush, Jacqueline A. Keane, Julian Parkhill (2015), "Roary: Rapid large-scale prokaryote pan genome analysis", Bioinformatics, doi:10.1093/bioinformatics/btv421

Installation instructions

Details on how to install Roary can be found here:
https://github.com/sanger-pathogens/Roary/blob/master/README.md

Inputs

Roary takes GFF3 files as input. They must contain the nucleotide sequence at the end of the file.

Input files from Prokka

All GFF3 files created by Prokka are valid with Roary and this is the recommended way of generating the input files. Each input file should have a unique prefix for the gene IDs (--prefix) to make it easier for you to identify where genes came from.

Input files from GenBank

On NCBI's website, GFF3 files only contain annotation and not the nucleotide sequence so cannot be used. You need to download the GenBank files plus nucleotide sequence and convert them. When downloading, click on the ‘show sequence’ option, ‘Update View’ then ‘Send’ to a ‘File’ of type ‘GenBank’. You can then use the Bio::Perl script ‘bp_genbank2gff3.pl’ to convert to GFF3. Just be aware that mixing different gene prediction methods and annotation pipelines can give noisier results.

Input files from GenBank draft WGS

Install Bio-RetrieveAssemblies which will allow you to download draft WGS assemblies from GenBank.

sudo cpanm -f Bio::RetrieveAssemblies

To download all Salmonella typhi annotated assemblies as GFF3 files:

retrieve_assemblies -a -f gff typhi

Software usage

To run the software and create a pan genome you use the '*roary'* script. It takes in GFF files and outputs various analysis.

roary

```
Usage:   roary [options] *.gff

Options: -p INT    number of threads [1]
         -o STR    clusters output filename [clustered_proteins]
         -f STR    output directory [.]
         -e        create a multiFASTA alignment of core genes using PRANK
         -n        fast core gene alignment with MAFFT, use with -e
         -i        minimum percentage identity for blastp [95]
         -cd FLOAT percentage of isolates a gene must be in to be core [99]
         -z        dont delete intermediate files
         -t INT    translation table [11]
         -v        verbose output to STDOUT
         -y        add gene inference information to spreadsheet
         -g INT    maximum number of clusters [50000]
         -qc       generate QC report with Kraken
         -k STR    path to Kraken database for QC, use with -qc
         -w        print version and exit
         -a        check dependancies and exit
         -h        this help message
```

For example:
Default usage

```
roary *.gff
```

Quickly generate a core gene alignment using 8 threads

```
roary -e --mafft -p 8 *.gff
```

Save results to a different directory

```
roary –f output_dir *.gff
```

Change the minimum blastp percentage identity. Its not advised to go below 90% unless you know what your doing.

```
roary –i 90 *.gff
```

Check that the software is installed correctly.

```
roary -a
```

query_pan_genome

Perform set operations on the pan genome to see the gene differences between groups of isolates.

```
Options: -g STR    groups filename [clustered_proteins]
         -a STR    action
(union/intersection/complement/gene_multifasta/difference) [union]
         -c FLOAT  percentage of isolates a gene must be in to be core [99]
         -o STR    output filename [pan_genome_results]
         -n STR    comma separated list of gene names for use with
gene_multifasta action
         -i STR    comma separated list of filenames, comparison set one
         -t STR    comma separated list of filenames, comparison set two
         -v        verbose output to STDOUT
         -h        this help message
```

Examples:
Union of genes found in isolates

```
    query_pan_genome -a union *.gff
```

Intersection of genes found in isolates (core genes)

```
    query_pan_genome -a intersection *.gff
```

Complement of genes found in isolates (accessory genes)

```
    query_pan_genome -a complement *.gff
```

Extract the sequence of each gene listed and create multi-FASTA files

```
    query_pan_genome -a gene_multifasta -n gryA,mecA,abc *.gff
```

Gene differences between sets of isolates

```
    query_pan_genome -a difference --input_set_one 1.gff,2.gff --input_set_two
3.gff,4.gff,5.gff
```

iterative_cdhit

Iteratively cluster a set of proteins with CD-hit, lower the threshold each time and extracting core genes (1 per isolate) to another file, and remove them from the input proteins file.

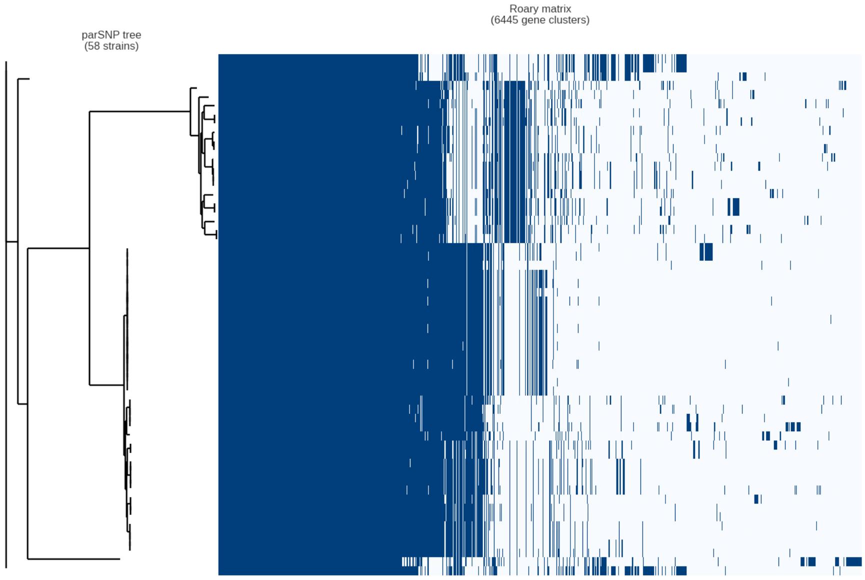
```
Options: -p INT   number of threads [1]
         -m STR   output filename for combined proteins [_combined_files]
         -n INT   number of isolates [1]
         -c STR   cd-hit output filename [_clustered]
         -f STR   output filename for filtered sequences
[_clustered_filtered.fa]
         -l FLOAT lower bound percentage identity [98.0]
         -u FLOAT upper bound percentage identity [99.0]
         -s FLOAT step size for percentage identity [0.5]
         -v       verbose output to STDOUT
         -h       this help message
```

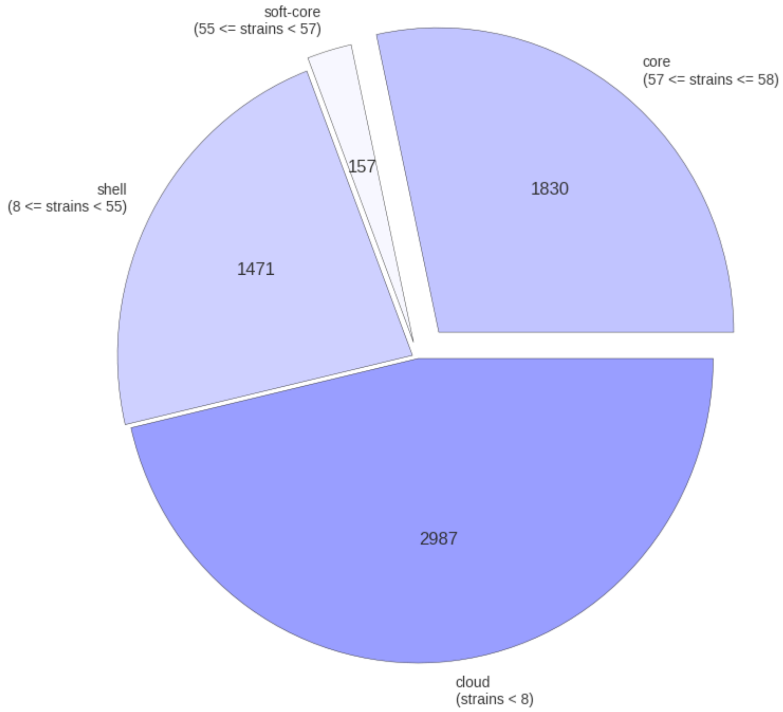
roary_plots.py

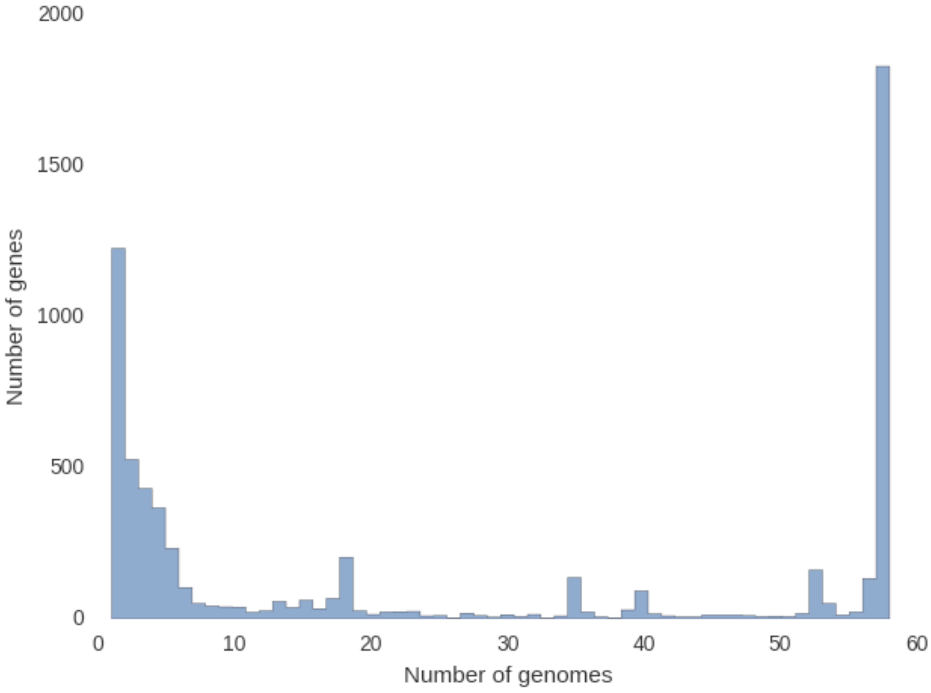
This contributed script by Marco Galardini is not installed by default but can be very useful. Additional details can be found here:
https://github.com/sanger-pathogens/Roary/tree/master/contrib/roary_plots
It provides 3 figures, showing the tree compared to a matrix with the presence and absence of core and accessory genes. The next is an pie chart of the

breakdown of genes and the number of isolate they are present in. And finally there is a graph with the frequency of genes versus the number of genomes.

```
roary_plots.py my_tree.tre gene_presence_absence.csv
```




soft-core
(55 <= strains < 57)
core
(57 <= strains <= 58)
shell
(8 <= strains < 55)
157
1830
1471
2987
cloud
(strains < 8)


2000
1500
1000
500
0
Number of genes
0
10
20
30
40
50
60
Number of genomes

Recipe for using Roary

1.) Annotate FASTA files with PROKKA
2.) Roary –e –mafft *.gff
3.) FastTree –nt –gtr core_gene_alignment.aln > my_tree.newick

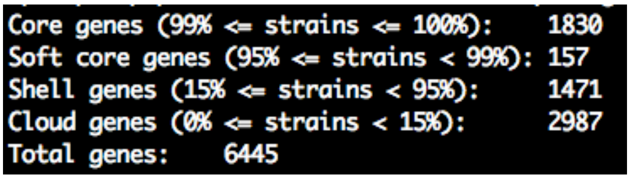
Output files

Table of output files and brief description

File	Description
summary_statistics.txt	Number of genes in the core and accessory
gene_presence_absence.csv	Spreadsheet with presence and absence of genes in each sample. Open in Excel.
pan_genome_reference.fa	FASTA file of nucleotide sequences with 1 sequence for every gene.
*.Rtab	Tab files for use in R
accessory_binary_genes.fa.newick	Tree in Newick format based on the binary presence and absence of genes in the accessory.
accessory_graph.dot	A graph in DOT format of how genes are linked together at the contig level in the accessory genome
core_accessory_graph.dot	A graph in DOT format of how genes are linked together at the contig level in the pan genome
clustered_proteins	Groups file where each line lists the sequences in a cluster
core_gene_alignment.aln	Multi-FASTA alignment of core genes

Summary Statistics

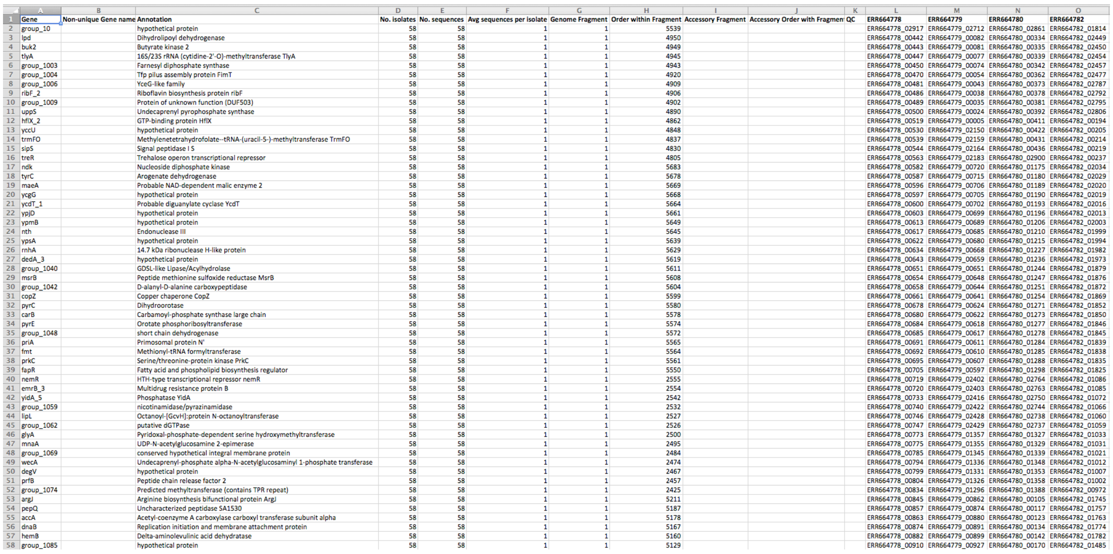
A text file with an overview of the genes and how frequently they occur in the input isolates. If the number of core genes is 0 it can indicate you have some contamination. Likewise if the total number of genes is very high.



```
Core genes (99% <= strains <= 100%):    1830
Soft core genes (95% <= strains < 99%): 157
Shell genes (15% <= strains < 95%):     1471
Cloud genes (0% <= strains < 15%):      2987
Total genes:    6445
```

Gene Presence and absence spreadsheet

The gene presence and absence spreadsheet lists each gene and which samples it is present in. The view below shows how it looks in Excel.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Gene	Non-unique Gene name	Annotation	No. isolates	No. sequences	Avg sequences per isolate	Genome Fragment	Order within Fragment	Accessory Fragment	Accessory Order with Fragmen	QC	ERR664778	ERR664779	ERR664780	ERR664782
2	group_10		hypothetical protein	58	58	1	1	5539				ERR664778_02917	ERR664779_02712	ERR664780_02861	ERR664782_01814
3	lpd		Dihydrolipoyl dehydrogenase	58	58	1	1	4950				ERR664778_00442	ERR664779_00082	ERR664780_00334	ERR664782_02449
4	buk2		Butyrate kinase 2	58	58	1	1	4949				ERR664778_00443	ERR664779_00081	ERR664780_00335	ERR664782_02450
5	tlyA		16S/23S rRNA (cytidine-2'-O)-methyltransferase TlyA	58	58	1	1	4945				ERR664778_00447	ERR664779_00077	ERR664780_00339	ERR664782_02454
6	group_1003		Farnesyl diphosphate synthase	58	58	1	1	4943				ERR664778_00450	ERR664779_00074	ERR664780_00342	ERR664782_02457
7	group_1004		Tfp pilus assembly protein FimT	58	58	1	1	4920				ERR664778_00470	ERR664779_00054	ERR664780_00362	ERR664782_02477
8	group_1006		YceG-like family	58	58	1	1	4909				ERR664778_00481	ERR664779_00043	ERR664780_00373	ERR664782_02787
9	ribF_2		Riboflavin biosynthesis protein ribF	58	58	1	1	4906				ERR664778_00486	ERR664779_00038	ERR664780_00378	ERR664782_02792
10	group_1009		Protein of unknown function (DUF503)	58	58	1	1	4902				ERR664778_00489	ERR664779_00035	ERR664780_00381	ERR664782_02795
11	uppS		Undecaprenyl pyrophosphate synthase	58	58	1	1	4890				ERR664778_00500	ERR664779_00024	ERR664780_00392	ERR664782_02806
12	hflX_2		GTP-binding protein HflX	58	58	1	1	4862				ERR664778_00519	ERR664779_00005	ERR664780_00411	ERR664782_00194
13	yccU		hypothetical protein	58	58	1	1	4848				ERR664778_00530	ERR664779_02150	ERR664780_00422	ERR664782_00205
14	trmFO		Methylenetetrahydrofolate--tRNA-(uracil-5-)-methyltransferase TrmFO	58	58	1	1	4837				ERR664778_00539	ERR664779_02159	ERR664780_00431	ERR664782_00214
15	sipS		Signal peptidase I S	58	58	1	1	4830				ERR664778_00544	ERR664779_02164	ERR664780_00436	ERR664782_00219
16	treR		Trehalose operon transcriptional repressor	58	58	1	1	4805				ERR664778_00563	ERR664779_02183	ERR664780_02900	ERR664782_00237
17	ndk		Nucleoside diphosphate kinase	58	58	1	1	5683				ERR664778_00582	ERR664779_00720	ERR664780_01175	ERR664782_02034
18	tyrC		Arogenate dehydrogenase	58	58	1	1	5678				ERR664778_00587	ERR664779_00715	ERR664780_01180	ERR664782_02029
19	maeA		Probable NAD-dependent malic enzyme 2	58	58	1	1	5669				ERR664778_00596	ERR664779_00706	ERR664780_01189	ERR664782_02020
20	ycgG		hypothetical protein	58	58	1	1	5668				ERR664778_00597	ERR664779_00705	ERR664780_01190	ERR664782_02019
21	ycdT_1		Probable diguanylate cyclase YcdT	58	58	1	1	5664				ERR664778_00600	ERR664779_00702	ERR664780_01193	ERR664782_02016
22	ypjD		hypothetical protein	58	58	1	1	5661				ERR664778_00603	ERR664779_00699	ERR664780_01196	ERR664782_02013
23	ypmB		hypothetical protein	58	58	1	1	5649				ERR664778_00613	ERR664779_00689	ERR664780_01206	ERR664782_02003
24	nth		Endonuclease III	58	58	1	1	5645				ERR664778_00617	ERR664779_00685	ERR664780_01210	ERR664782_01999
25	ypsA		hypothetical protein	58	58	1	1	5639				ERR664778_00622	ERR664779_00680	ERR664780_01215	ERR664782_01994
26	rnhA		14.7 kDa ribonuclease H-like protein	58	58	1	1	5629				ERR664778_00634	ERR664779_00668	ERR664780_01227	ERR664782_01982
27	dedA_3		hypothetical protein	58	58	1	1	5619				ERR664778_00643	ERR664779_00659	ERR664780_01236	ERR664782_01973
28	group_1040		GDSL-like Lipase/Acylhydrolase	58	58	1	1	5611				ERR664778_00651	ERR664779_00651	ERR664780_01244	ERR664782_01879
29	msrB		Peptide methionine sulfoxide reductase MsrB	58	58	1	1	5608				ERR664778_00654	ERR664779_00648	ERR664780_01247	ERR664782_01876
30	group_1042		D-alanyl-D-alanine carboxypeptidase	58	58	1	1	5604				ERR664778_00658	ERR664779_00644	ERR664780_01251	ERR664782_01872
31	copZ		Copper chaperone CopZ	58	58	1	1	5599				ERR664778_00661	ERR664779_00641	ERR664780_01254	ERR664782_01869
32	pyrC		Dihydroorotase	58	58	1	1	5580				ERR664778_00678	ERR664779_00624	ERR664780_01271	ERR664782_01852
33	carB		Carbamoyl-phosphate synthase large chain	58	58	1	1	5578				ERR664778_00680	ERR664779_00622	ERR664780_01273	ERR664782_01850
34	pyrE		Orotate phosphoribosyltransferase	58	58	1	1	5574				ERR664778_00684	ERR664779_00618	ERR664780_01277	ERR664782_01846
35	group_1048		short chain dehydrogenase	58	58	1	1	5572				ERR664778_00685	ERR664779_00617	ERR664780_01278	ERR664782_01845
36	priA		Primosomal protein N'	58	58	1	1	5565				ERR664778_00691	ERR664779_00611	ERR664780_01284	ERR664782_01839
37	fmt		Methionyl-tRNA formyltransferase	58	58	1	1	5564				ERR664778_00692	ERR664779_00610	ERR664780_01285	ERR664782_01838
38	prkC		Serine/threonine-protein kinase PrkC	58	58	1	1	5561				ERR664778_00695	ERR664779_00607	ERR664780_01288	ERR664782_01835
39	fapR		Fatty acid and phospholipid biosynthesis regulator	58	58	1	1	5550				ERR664778_00705	ERR664779_00597	ERR664780_01298	ERR664782_01825
40	nemR		HTH-type transcriptional repressor nemR	58	58	1	1	2555				ERR664778_00719	ERR664779_02402	ERR664780_02764	ERR664782_01086
41	emrB_3		Multidrug resistance protein B	58	58	1	1	2554				ERR664778_00720	ERR664779_02403	ERR664780_02763	ERR664782_01085
42	yidA_5		Phosphatase YidA	58	58	1	1	2542				ERR664778_00733	ERR664779_02416	ERR664780_02750	ERR664782_01072
43	group_1059		nicotinamidase/pyrazinamidase	58	58	1	1	2532				ERR664778_00740	ERR664779_02422	ERR664780_02744	ERR664782_01066
44	lipL		Octanoyl-[GcvH]:protein N-octanoyltransferase	58	58	1	1	2527				ERR664778_00746	ERR664779_02428	ERR664780_02738	ERR664782_01060
45	group_1062		putative dGTPase	58	58	1	1	2526				ERR664778_00747	ERR664779_02429	ERR664780_02737	ERR664782_01059
46	glyA		Pyridoxal-phosphate-dependent serine hydroxymethyltransferase	58	58	1	1	2500				ERR664778_00773	ERR664779_01357	ERR664780_01327	ERR664782_01033
47	mnaA		UDP-N-acetylglucosamine 2-epimerase	58	58	1	1	2495				ERR664778_00775	ERR664779_01355	ERR664780_01329	ERR664782_01031
48	group_1069		conserved hypothetical integral membrane protein	58	58	1	1	2484				ERR664778_00785	ERR664779_01345	ERR664780_01339	ERR664782_01021
49	wecA		Undecaprenyl-phosphate alpha-N-acetylglucosaminyl 1-phosphate transferase	58	58	1	1	2474				ERR664778_00794	ERR664779_01336	ERR664780_01348	ERR664782_01012
50	degV		hypothetical protein	58	58	1	1	2467				ERR664778_00799	ERR664779_01331	ERR664780_01353	ERR664782_01007
51	prfB		Peptide chain release factor 2	58	58	1	1	2457				ERR664778_00804	ERR664779_01326	ERR664780_01358	ERR664782_01002
52	group_1074		Predicted methyltransferase (contains TPR repeat)	58	58	1	1	2425				ERR664778_00834	ERR664779_01296	ERR664780_01388	ERR664782_00972
53	argJ		Arginine biosynthesis bifunctional protein ArgJ	58	58	1	1	5211				ERR664778_00845	ERR664779_00862	ERR664780_00105	ERR664782_01745
54	pepQ		Uncharacterized peptidase SA1530	58	58	1	1	5187				ERR664778_00857	ERR664779_00874	ERR664780_00117	ERR664782_01757
55	accA		Acetyl-coenzyme A carboxylase carboxyl transferase subunit alpha	58	58	1	1	5178				ERR664778_00863	ERR664779_00880	ERR664780_00123	ERR664782_01763
56	dnaB		Replication initiation and membrane attachment protein	58	58	1	1	5167				ERR664778_00874	ERR664779_00891	ERR664780_00134	ERR664782_01774
57	hemB		Delta-aminolevulinic acid dehydratase	58	58	1	1	5160				ERR664778_00882	ERR664779_00899	ERR664780_00142	ERR664782_01782
58	group_1085		hypothetical protein	58	58	1	1	5129				ERR664778_00910	ERR664779_00927	ERR664780_00170	ERR664782_01485

Column	Description
A	The gene name, which is the most frequently occurring gene name from the sequences in the cluster. If there is no gene name, then it is given a generic unique name 'group_XXX'.
B	A non unique gene name, where sequences with the same gene name have ended up in different groups. It might be because of split genes, or miss annotation.
C	Functional annotation
D	Number of isolates represented in the cluster
E	Number of sequences in the cluster
F	Average number of sequences per isolate. This is normally 1. If this is greater than 1 then there is over clustering and the paralogs couldn't be split.
G	Genome fragment, where there is evidence at the contig level that the genes are linked.
H	Order within fragment, combined with the genome fragment this gives an indication of the order of genes within the graph. In Excel, sort on Column G and H.
I	Accessory Fragment is where core genes are excluded and there is evidence at contig level that the genes are linked.
J	Accessory order with fragment, combined with the Accessory fragment this gives an indication of the order of genes within the accessory graph. In Excel, sort on columns I and J.
K	Comments on the quality of the cluster. Miss predictions are noted, as are single genes on single contigs, which can be evidence of low level contamination.
Others	Presence and absence of genes in each sample, with the corresponding source Gene ID.

Pan genome reference

This is a FASTA file which contains a single representative nucleotide sequence from each of the clusters in the pan genome (core and accessory). The name of each sequence is the source sequence ID followed by the cluster it came from.

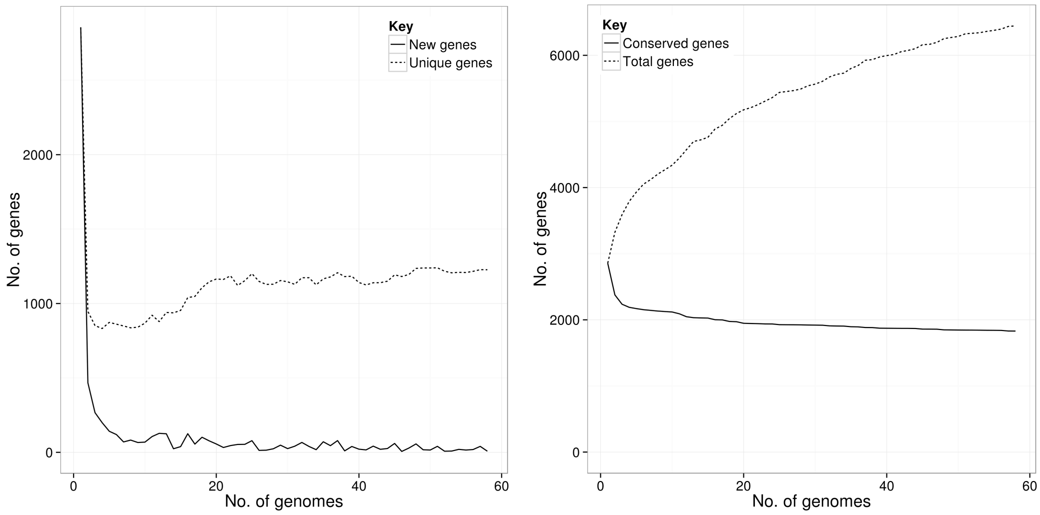
This file can be of use for reference guided assembly, whole genome MLST or for mapping raw reads to it.

Accessory binary genes tree

This is a tree created using the binary presence and absence of accessory genes. It is in Newick format and can be viewed in FigTree. It is only a quick and dirty tree to roughly group isolates together based on their accessory genome and is in no way reliable other than to give a quick insight into the data. If you want a more accurate tree you need to use the core gene alignment as your starting point.

Rtab files

There is an additional script called 'create_pan_genome_plots.R' which requires R and the ggplot2 library. It takes in the *.Rtab files and produces graphs on how the pan genome varies as genomes are added (in random orders).



Core gene alignment

If you pass in the '-e' parameter to roary, a multi-FASTA alignment of all of the core genes is created. By default it uses PRANK (Löytynoja, 2014) which performs a codon aware alignment. It is slow but accurate. If you pass in '-e --mafft' it will use MAFFT which performs a nucleotide alignment. It is very fast but less accurate. This can then be used as input to build a phylogenetic tree. To reduce the memory and run time, you can pre filter the alignment using snp_sites https://github.com/sanger-pathogens/snp_sites
Just be aware that recombination is not taken care of with this method.

Software availability

All of the source code is available under the GNU GPL 3 open source license from: https://github.com/sanger-pathogens/Roary

Contact us

If you have any queries about Roary or wish to report any bugs please email roary@sanger.ac.uk

If you are having problems installing the software you should contact your local system administrator in the first instance as they will be best placed to assist you.

FAQ

Strange errors

Check the dependencies with ‘roary -a’. If theres something missing, then you’ll need to install it.

cdhit seg faults

Old versions of cdhit have a bug, so you need to use at least version 4.6.1. The cdhit packages for Ubuntu 12.04 seem to be effected, so installing from the source is the only option.

Kraken installed via homebrew throws an error.

Theres a bug and you'll need to install it from source on older versions of OSX (like Mountain Lion).

Why dont you bundle a Kraken database for the QC?

Its massive (2.7GB currently) and changes as RefSeq is updated. The authors have prebuilt databases and details about how to make your own.

References

Löytynoja,A. (2014) Phylogeny-aware alignment with PRANK. *Methods Mol. Biol.*, **1079**, 155–170.

Seemann,T. (2014) Prokka: rapid prokaryotic genome annotation. *Bioinformatics*, **30**, 2068–2069.