PATRIC Command Line Interface (CLI) Tutorial

Table of Contents

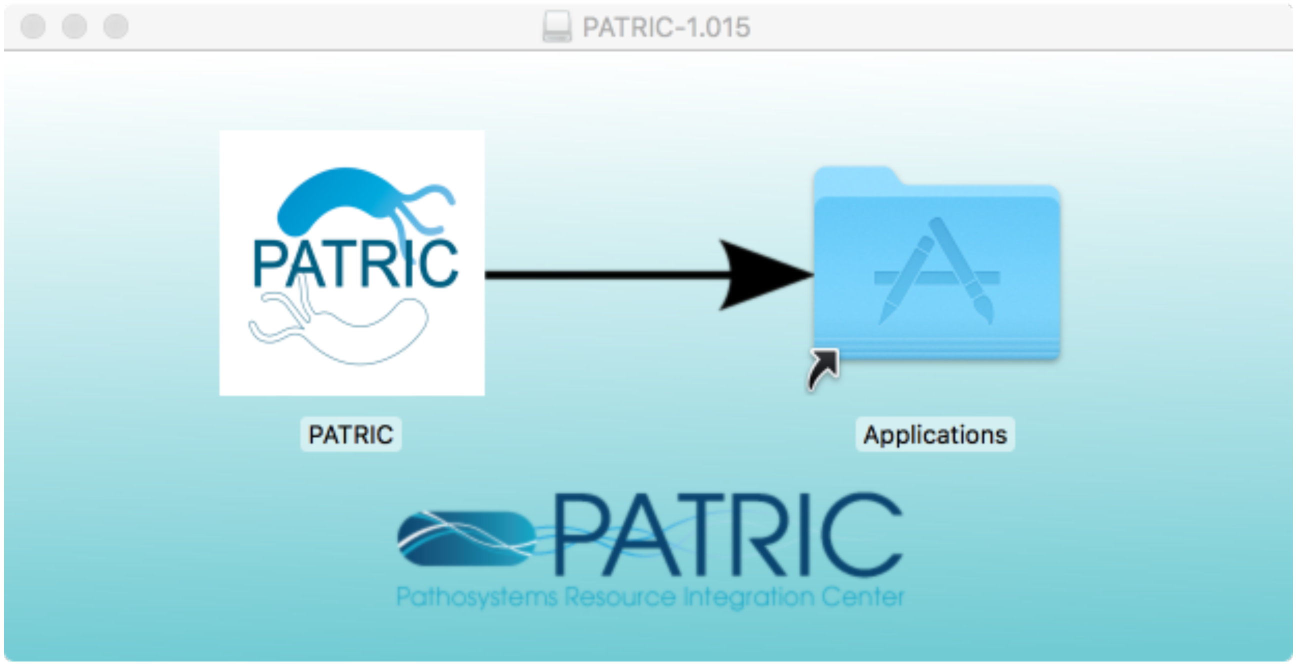
1. Installing the CLI Release

We currently only have a Mac OSX release of the CLI package, but we should soon have a Windows version as well. In the past we have created Ubuntu packaging for this technology as well and may do so if there is a demand for it.

The releases are available at the PATRIC3 github site.

Download the latest version of the PATRIC dmg (disk image) file. Click on the downloaded file to open it, and drag the PATRIC icon on to the Application folder icon. This will install into your Applications folder.

Then doubleclick on the PATRIC icon in the Applications folder. This will bring up a new Terminal window that is configured for access to the PATRIC command line tools.



Note. You may need to authorize to run this app. Go to System Preference -> Security & Privacy -> "open anyway".

---

2. Command Line Introduction

Most of the PATRIC command line tools take as input a file containing a single column set or a tab-separated table and they output a modified table. The most common modification is the addition of one of more columns. We create "pipelines" of these tools to implement fairly complex transformations leading to the final table containing the desired output. We begin with accessing information about genomes.

Accessing Genome Information

Consider the following example.

```
$ p3-all-genomes

genome.genome_id
1390.176
1398.26
1345597.3
282669.3
...
```

`p3-all-genomes` takes no input and is what we call a generator; it returns the set of all genome ids in PATRIC. Notice that the first line is a header identifying the columns in the table. Most p3 commands expect this header. If you were interested in certain data about the genomes, you would use the command `p3-get-genome-data`, which takes as input a set of genome ids, like this;

```
$ p3-all-genomes | p3-get-genome-data

genome.genome_id    genome.genome_id    genome.genome_name  genome.taxon_id
genome.genome_status    genome.gc_content
1390.176    1390.176    Bacillus amyloliquefaciens strain B425  1390    WGS 45.7
1398.26 1398.26 Bacillus coagulans strain B4098 1398    WGS 47.39
1345597.3   1345597.3   Helicobacter pylori SA216A  1345597 WGS 39.02
282669.3    282669.3    Psychrobacter cibarius strain W1    282669  WGS 70.43
...
```

`p3-all-genomes` is used to generate a set of input ids that we pipe into `p3-get-genome-data`, which returns a 6 column table with data about the genome; the genome_id, genome_name, taxon_id, genome_status and gc_content. You can control which of these fields is returned with the `-a` argument.

```
$ p3-all-genomes | p3-get-genome-data -a genome_name

genome.genome_id    genome.genome_name
1390.176    Bacillus amyloliquefaciens strain B425
1398.26 Bacillus coagulans strain B4098
1345597.3   Helicobacter pylori SA216A
282669.3    Psychrobacter cibarius strain W1
1349753.3   Caldimonas taiwanensis NBRC 104434
1285191.3   Desulfotomaculum intricatum strain NBRC 109411
...
```

If you were interested in only Streptococcus genomes, you could use the match command like this;

```
$ p3-all-genomes | p3-get-genome-data -a genome_name | p3-match -c2 Streptococcus

genome.genome_id    genome.genome_name
1313.7195   Streptococcus pneumoniae strain 2842STDY5753638
1313.7189   Streptococcus pneumoniae strain 2842STDY5643920
1313.7203   Streptococcus pneumoniae strain 2842STDY5643723
1313.7208   Streptococcus pneumoniae strain 2842STDY5643999
1313.7199   Streptococcus pneumoniae strain 2842STDY5644588
1313.7207   Streptococcus pneumoniae strain 2842STDY5643980
...
```

Here, we retrieved the id of all genomes in PATRIC, piped that output to get the name and produce a two column table, and piped that table to a command to match for the string Streptococcus in column 2, thus filtering the table to contain only Streptococcus genomes.

There are other ways to accomplish this, but the example serves to demonstrate what we mean by creating pipelines of commands and producing tables of information.

Accessing Features

If you want to look into the features of a genome, you would use the `p3-get-genome-features` command. `p3-genome-features` takes as input a set of genome ids. The following example uses the `p3-echo` command to generate input for `p3-get-genome-features`.

```
$ p3-echo -t genome.genome_id 282669.3 | p3-get-genome-features | head

genome.genome_id    feature.patric_id   feature.feature_type    feature.location
feature.product
282669.3    fig|282669.3.repeat.1   repeat_region   1..127  repeat region
282669.3    fig|282669.3.repeat.2   repeat_region   586..712    repeat region
282669.3    fig|282669.3.peg.4  CDS complement(1..909)  Aspartyl-tRNA(Asn)
amidotransferase subunit A (EC 6.3.5.6) @ Glutamyl-tRNA(Gln) amidotransferase subunit
A (EC 6.3.5.7)
282669.3    fig|282669.3.repeat.3   repeat_region   1..127  repeat region
282669.3    fig|282669.3.repeat.4   repeat_region   805..931    repeat region
282669.3    fig|282669.3.repeat.5   repeat_region   869..1006   repeat region
282669.3    fig|282669.3.repeat.6   repeat_region   1..127  repeat region
282669.3    fig|282669.3.repeat.7   repeat_region   1110..1236  repeat region
282669.3    fig|282669.3.repeat.8   repeat_region   1..127  repeat region
```

Notice that the command returns all information about features by default. If you were only interested in the feature ids, you would specify that with the -a option.

```
$ p3-echo -t genome.genome_id 282669.3 | p3-get-genome-features -a patric_id | head
genome.genome_id    feature.patric_id

282669.3    fig|282669.3.repeat.1
282669.3    fig|282669.3.repeat.2
282669.3    fig|282669.3.peg.4
282669.3    fig|282669.3.repeat.3
282669.3    fig|282669.3.repeat.4
282669.3    fig|282669.3.repeat.5
282669.3    fig|282669.3.repeat.6
282669.3    fig|282669.3.repeat.7
282669.3    fig|282669.3.repeat.8
```

Since this returns all feature types, it might be desirable to limit the features returned to a specific type. Here, we return the ids of only the pegs in a Genome by using the --equal option.

```
$ p3-echo -t genome.genome_id 282669.3 | p3-get-genome-features --equal
feature_type,CDS -a patric_id | head
genome.genome_id    feature.patric_id

282669.3    fig|282669.3.peg.4
282669.3    fig|282669.3.peg.43
282669.3    fig|282669.3.peg.72
282669.3    fig|282669.3.peg.83
282669.3    fig|282669.3.peg.90
282669.3    fig|282669.3.peg.117
282669.3    fig|282669.3.peg.179
282669.3    fig|282669.3.peg.207
282669.3    fig|282669.3.peg.214
```

In this tutorial we have introduced the basics of using the PATRIC Command Line Interface (CLI) and how to access data relating to genomes and features.

In the following tutorials, you will learn what all the commands are and how to use them to explore the PATRIC website, to build collections of data and to apply bioinformatic tools against your data.

Getting help on commands

Generally, commands support inline help; passing `-h` or `–help` gives you options you can provide.

```
$ p3-all-genomes --help
p3-all-genomes.pl [-aehr] [long options...]
    -a STR... --attr STR...               field(s) to return
    -e STR... --eq STR... --equal STR...  search constraint(s) in the
                                          form field_name,value
    --in STR...                           any-value search constraint(s)
                                          in the form
                                          field_name,value1,value2,...,valueN
    -r STR... --required STR...           field(s) required to have values
    -h --help                             display usage information
```

For example, p3-all-genomes allows you search PATRIC genomes with `–eq` argument. So, command line below provides exactly same results with the first example. This example performs better. Can you answer Why?

```
$ p3-all-genomes --eq genome_name,Streptococcus | p3-get-genome-data -a genome_name
$ p3-all-genomes --eq genome_name,Streptococcus --attr genome_name
```

3. Logging in to PATRIC at the Command Line

For operations on private data, such as that data stored in your workspace, you will need to be logged in. You can check to see if you are already logged in using the `p3-whoami` command:

```
$ p3-whoami
You are logged in as:
brettin@patricbrc.org
```

If you were not logged in, when you run the `p3-whoami` command you would see that you are not logged in. If this were the case, you would not be able to upload to your workspace or see any of the data in your workspace for example.

```
$ p3-whoami
You are not logged in.
```

If you are not logged in, you can do so using the `p3-login` command. Notice that your user name contains the suffix `@patricbrc.org`. This suffix is required so that your PATRIC credentials are used rather than your RAST credentials.

```
$ p3-login brettin@patricbrc.org
Password: ****
Logged in with username brettin@patricbrc.org
$ p3-whoami
You are logged in as:
brettin@patricbrc.org
$ p3-logout
Logged in as:
public
$ p3-whoami
You are not logged in.
```

4. Creating Genome Groups

Creating a genome group in your workspace allows you to create logical groups of genomes for downstream comparative analysis. In this example, you will create two genome groups. One genome group will contain only Streptococcus aureus, and the second genome group will contain all Streptococcus genomes except those from Streptococcus aureus.

This example assumes familiarity with a few other commands in the CLI to create the input file containing the genomes to be put in the new genome group. You do not need to be familiar with these commands, just the format of the input file. The format of the input file is simply a list of genome ids, one per line with no other information in the file.

```
$ p3-all-genomes --in genome_name,aureus > Staphylococcus.aureus.genomes
$ p3-all-genomes --in genome_name,Staphylococcus > Staphylococcus.all.genomes
$ a_not_b Staphylococcus.all.genomes Staphylococcus.aureus.genomes >
Staphylococcus.not.aureus.genomes

$ wc *.genomes
    9257   41089  399759 Staphylococcus.all.genomes
    8383   36955  356756 Staphylococcus.aureus.genomes
     876    4146   43099 Staphylococcus.not.aureus.genomes
   18516   82190  799614 total
```

Let's take a quick look at a few entries in each file using the unix head command. In this case, the first five lines of each file are displayed. Notice that the header exists in two of the three files. It is not in the file created with the a_not_b command because the header appeared in both input files to the a_not_b command.

```
$ head -n 5 *.genomes
==> Staphylococcus.all.genomes <==
genome.genome_id
904758.3
904731.3
904739.3
904750.3

==> Staphylococcus.aureus.genomes <==
genome.genome_id
904758.3
904731.3
904739.3
904750.3

==> Staphylococcus.not.aureus.genomes <==
1000590.6
1008454.3
1034809.4
1078083.3
1115805.3
```

The input files to create genome groups are almost ready. The input to `p3-put-genome-group` is a single column file with the only values being genome ids. The header needs to be removed. We can easily do this with an editor, or use the unix grep command with the -v option. For this example, we will just edit the file with an editor and remove the header.

The `p3-put-genome-group` takes the list of genome ids on standard input. Using the unix cat command we can send the contents of our newly created files of genome ids to the `p3-put-genome-group` command with the following command.

```
$ cat Staphylococcus.not.aureus.genomes | p3-put-genome-group "Staphylococcus not aureus"
$ cat Staphylococcus.aureus.genomes | p3-put-genome-group "Staphylococcus aureus"
```

The two newly created genome groups are now visible and usable in your workspace using on the PATRIC web site, as well as accessible using the command line interface.

```
$ p3-get-genome-group "Staphylococcus aureus"
904758.3
904731.3
904739.3
904750.3
904763.3
904725.3
904734.3
904737.3
904754.3
904768.3
<...>
```

5. Uploading data from the command line into your workspace

The PATRIC workspace is a place where you can upload your data so that it can be integrated with existing public data in PATRIC while at the same time maintaining privacy. As an example, you can upload contigs to your workspace, annotate them using the PATRIC annotation service, and then compare the annotated results with publicly annotated genomes in PATRIC.

In this example, I am going to upload contigs from my mac to my workspace in PATRIC using the command line interface rather than using the web interface for uploading contigs to my workspace.

Uploading data

The ws-create command is used to upload data to my workspace:

```
$ ws-create -h
ws-create.pl [-ahopu] [long options...]
         -p --permission   Permissions for folders created
         -u --useshock     Upload file to shock and store link in workspace
         -o --overwrite    Overwrite existing destination object
         --wsurl           Workspace URL
         -a --admin        Run as administrator
         -h --help         Show this usage message
$ ws-create -u /brettin@patricbrc.org/home/MinhashDev/UC.MICU.02.30.fastq Reads
/home/brettin/assemblies/UC.MICU.02.30.fastq
```

In this case, I used the -u option to tell the system to upload the genomes into the Shock bulk storage system; for any file larger than a few kilobytes we recommend that this option be used.

If I want to verify the uploaded reads, I can use do a listing on the folder that I uploaded the reads to.

```
$ ws-ls /brettin@patricbrc.org/home/MinhashDev

Name                 Owner                 Type       Moddate             Size
User perm Global perm
UC.MICU.02.30.msh    brettin@patricbrc.org job_result 2016-10-01T03:04:01       1513 o
        n
.UC.MICU.02.30.msh   brettin@patricbrc.org folder     2016-10-01T01:19:34          0 o
        n
UC.MICU.02.30.fastq brettin@patricbrc.org reads      2016-09-30T21:34:12 8284314281 o
        n
```

The folder listing includes fields such as owner, the object's name (specified by first positional argument in the ws-create command) type (specified by the second positional argument) and permissions (specified by the third positional argument).

6. Computing Signature Clusters: an Application of the Command-Line Tools

Introduction: What is a Signature Cluster?

In this tutorial, we show how to use a tool that we have created to help you locate clusters of genes that distinguish genomes from two designated sets of genomes. For example, suppose that you have a set of genomes from a given species and a second set from different species in the same genus. In this case, we might look for chromosomal clusters that occur in most genomes from the specific genus, but almost never occur in genomes from a different species in the same genus. This is just one of a growing set of tools you can use to access PATRIC data, but we think of it as extremely interesting.

So, the general operation we are implementing might be described as follows:

1. Define a set of closely-related genomes (usually a set of genomes from a single species). Call this set **GS1**.
2. Define a second set of genomes which will be used for comparison and call it **GS2**. Typically this would be a set establishing a "context". The usual contents of **GS2** would be genomes from the same genus, but different species.
3. Then define the notion of *signature family* as a protein family in which all members (or almost all members) occur in all genomes in **GS1**, but no (or very few) genomes in **GS2**.
4. Finally, define a *signature cluster* as a set of instances of signature families that occur close to one another on the contigs of a genome in **GS1**. Since a signature cluster contains only signature families, by definition it can occur in **GS1**, but only very seldom in **GS2**. We will argue that the signature clusters are very effective for locating chromosomal clusters that are very local phylogentically and correspond to molecular machines that are quite different from those that include the core cellular machinery. They are things like

- virulence factors,
- antibiotic fabrication mechanisms,
- prophages,
- special transportation cassets,
- and so forth.

How to Compute Signature Clusters

In this short tutorial we will compute signature clusters for *Streptococcus pyogenes*. The actual computation can be done for any genus and species for which you have enough genomes (say, 20 within the species and 20 from different species within the same genus).

Step 1: Defining GS1 and GS2

The following three lines of code create three tables encoding genome sets. We have included "head" statements to show that each row in each table contains two fields: a genome id and a genome name.

```
$ p3-all-genomes --attr genome_name --eq 'genome_name,Streptococcus' >
all.strep.genomes
head all.strep.genomes
genome.genome_id    genome.genome_name
1313.7014   Streptococcus pneumoniae P310839-218
208435.3    Streptococcus agalactiae 2603V/R
171101.6    Streptococcus pneumoniae R6
160490.10   Streptococcus pyogenes M1 GAS
568814.3    Streptococcus suis BM407
862971.3    Streptococcus anginosus C238
888833.3    Streptococcus australis ATCC 700641
864569.5    Streptococcus bovis ATCC 700338
482234.3    Streptococcus canis FSL Z3-227

$ p3-match --col 2 pyogenes < all.strep.genomes > pyogenes
head pyogenes
genome.genome_id    genome.genome_name
160490.10   Streptococcus pyogenes M1 GAS
1314.192    Streptococcus pyogenes strain NGAS322
798300.3    Streptococcus pyogenes MGAS15252
864568.3    Streptococcus pyogenes ATCC 10782
1314.198    Streptococcus pyogenes strain NGAS743
1314.197    Streptococcus pyogenes strain NGAS596
1314.196    Streptococcus pyogenes strain NGAS327
1314.168    Streptococcus pyogenes strain 19615
301451.4    Streptococcus pyogenes serotype M18 strain CPost

$ p3-match --col 2 pyogenes --reverse < all.strep.genomes > not.pyogenes
head not.pyogenes
genome.genome_id    genome.genome_name
1313.7014   Streptococcus pneumoniae P310839-218
208435.3    Streptococcus agalactiae 2603V/R
171101.6    Streptococcus pneumoniae R6
568814.3    Streptococcus suis BM407
862971.3    Streptococcus anginosus C238
888833.3    Streptococcus australis ATCC 700641
864569.5    Streptococcus bovis ATCC 700338
482234.3    Streptococcus canis FSL Z3-227
862969.3    Streptococcus constellatus subsp. pharyngis C1050
```

The first command looks at all of the PATRIC genomes, keeps only those which have 'Streptococcus' within the *genome_name* field, and writes out one line for each extracted *Streptococcus* genome. This is actually a fairly complex incantation, so we urge you to try to construct the corresponding command for a different species (say, *Staphylococcus*).

Then the *p3-match* commands create a list of *Streptococcus pyogenes* genomes and a set of *Streptococcus* genomes that are not from the *pyogenes* species.

Please construct corresponding sets for *Staphylococcus aureus* (that is, construct the two files **aureus** and **not.aureus**).

Once you have constructed your genome sets, verify that they include what appear to be a reasonable collection of genomes.

Computing Signature Clusters

Now that we have **GS1** and **GS2** defined, we can compute the signature clusters using something like

```
$ p3-related-by-clusters --gs1 pyogenes  \
                        --gs2 not.pyogenes \
                        --sz1 20 \
                        --sz2 20 \
                        --min 0.8 \
                        --max 0.1 \
                        --iterations 2 \
                        --output Strep
```

Let us briefly discuss the process being requested:

1. First, we take 20 random genomes from **GS1** and 20 from **GS2** (these sizes are specified by **sz1** and **sz2**) Then, we compute the protein families that occur in at least 80% of the genomes in **GS1**, but none of the genomes in **GS2** (the thresholds are specified by the **min** and **max** arguments). These are the *signature families* that we will use to search for *signature clusters*.
2. Then we compute the desired signature clusters, base on the reandomly selected genome sets.
3. We save the clusters computed; this is called a single *iteration*. We redo the selection of random genomes, computation of signature families, and computation of signature clusters (added to a growing set), until we have completed the requested number of iterations (in our example, we specified "2").

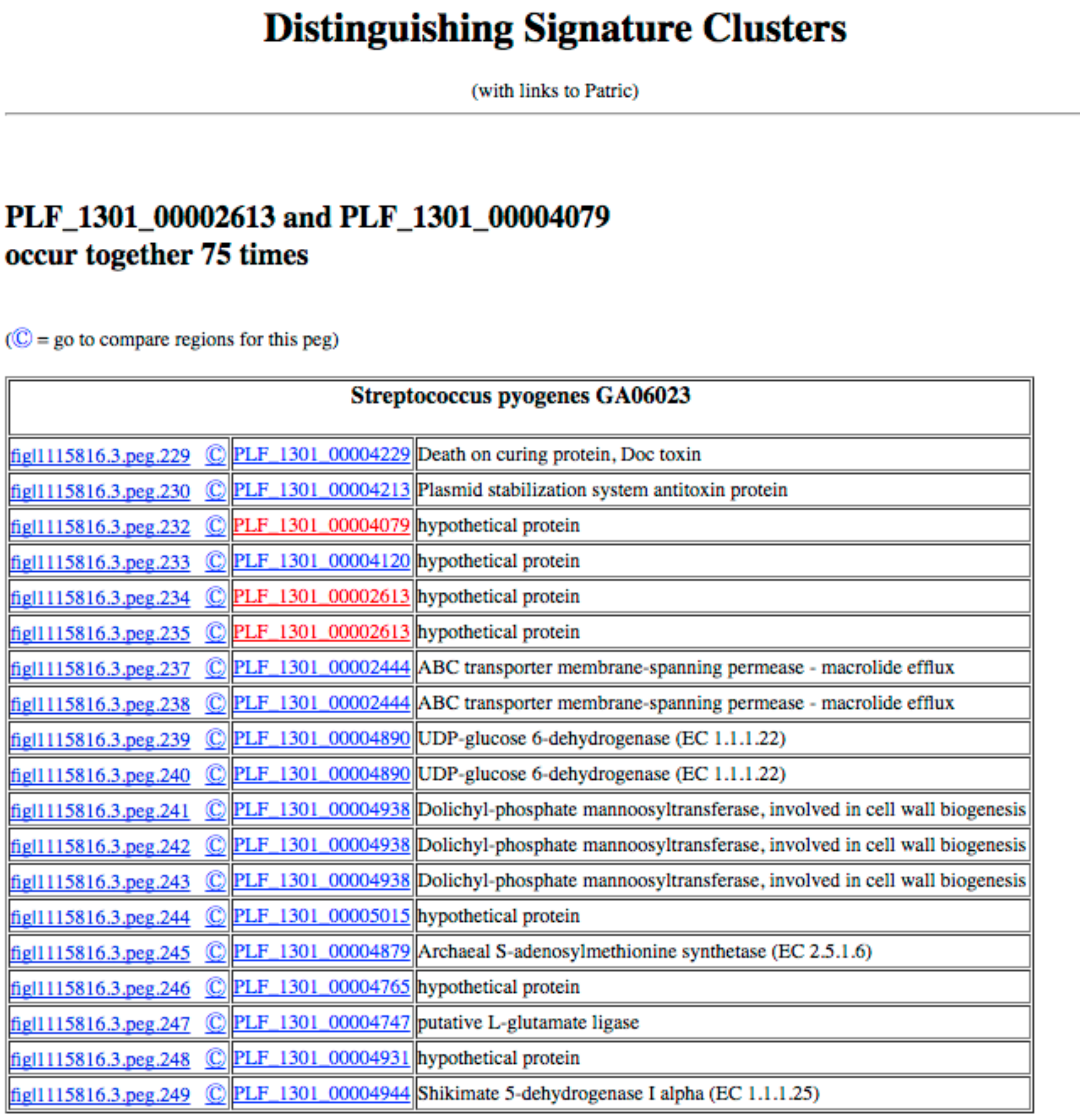
Thus, we build up a collection of signature clusters recorded in the designated ouput directory.

Looking at the Results

To look at the computed signature clusters, use something like

```
$ p3-format-results -d Strep | p3-aggregates-to-html > clusters.html
$ open clusters.html
```

The results will look something like this:



Distinguishing Signature Clusters

(with links to Patric)

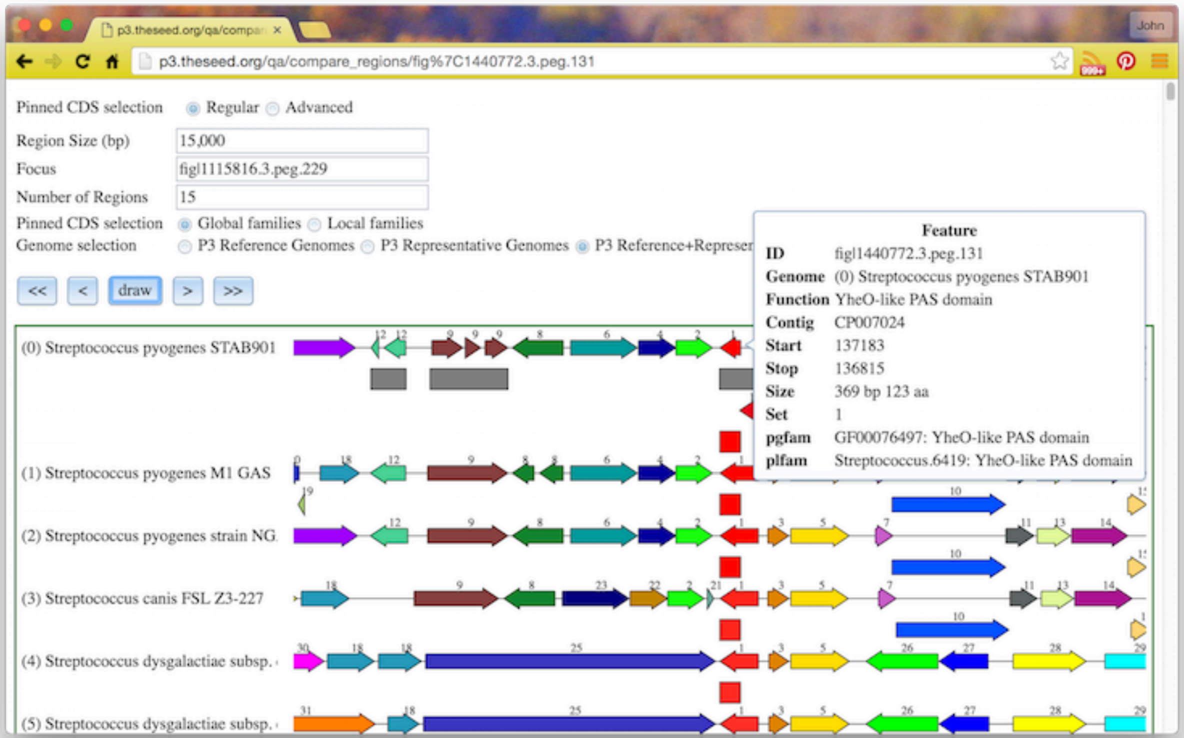
PLF_1301_00002613 and PLF_1301_00004079 occur together 75 times

(Ⓒ = go to compare regions for this peg)

Streptococcus pyogenes GA06023

fig|1115816.3.peg.229	Ⓒ	PLF_1301_00004229	Death on curing protein, Doc toxin
fig|1115816.3.peg.230	Ⓒ	PLF_1301_00004213	Plasmid stabilization system antitoxin protein
fig|1115816.3.peg.232	Ⓒ	PLF_1301_00004079	hypothetical protein
fig|1115816.3.peg.233	Ⓒ	PLF_1301_00004120	hypothetical protein
fig|1115816.3.peg.234	Ⓒ	PLF_1301_00002613	hypothetical protein
fig|1115816.3.peg.235	Ⓒ	PLF_1301_00002613	hypothetical protein
fig|1115816.3.peg.237	Ⓒ	PLF_1301_00002444	ABC transporter membrane-spanning permease - macrolide efflux
fig|1115816.3.peg.238	Ⓒ	PLF_1301_00002444	ABC transporter membrane-spanning permease - macrolide efflux
fig|1115816.3.peg.239	Ⓒ	PLF_1301_00004890	UDP-glucose 6-dehydrogenase (EC 1.1.1.22)
fig|1115816.3.peg.240	Ⓒ	PLF_1301_00004890	UDP-glucose 6-dehydrogenase (EC 1.1.1.22)
fig|1115816.3.peg.241	Ⓒ	PLF_1301_00004938	Dolichyl-phosphate mannoosyltransferase, involved in cell wall biogenesis
fig|1115816.3.peg.242	Ⓒ	PLF_1301_00004938	Dolichyl-phosphate mannoosyltransferase, involved in cell wall biogenesis
fig|1115816.3.peg.243	Ⓒ	PLF_1301_00004938	Dolichyl-phosphate mannoosyltransferase, involved in cell wall biogenesis
fig|1115816.3.peg.244	Ⓒ	PLF_1301_00005015	hypothetical protein
fig|1115816.3.peg.245	Ⓒ	PLF_1301_00004879	Archaeal S-adenosylmethionine synthetase (EC 2.5.1.6)
fig|1115816.3.peg.246	Ⓒ	PLF_1301_00004765	hypothetical protein
fig|1115816.3.peg.247	Ⓒ	PLF_1301_00004747	putative L-glutamate ligase
fig|1115816.3.peg.248	Ⓒ	PLF_1301_00004931	hypothetical protein
fig|1115816.3.peg.249	Ⓒ	PLF_1301_00004944	Shikimate 5-dehydrogenase I alpha (EC 1.1.1.25)

If you click on the feature ID, you will be taken to the Patric Feature Page for that feature:



Pinned CDS selection [✓ Regular] Advanced

Region Size (bp) 15,000

Focus fig|1115816.3.peg.229

Number of Regions 15

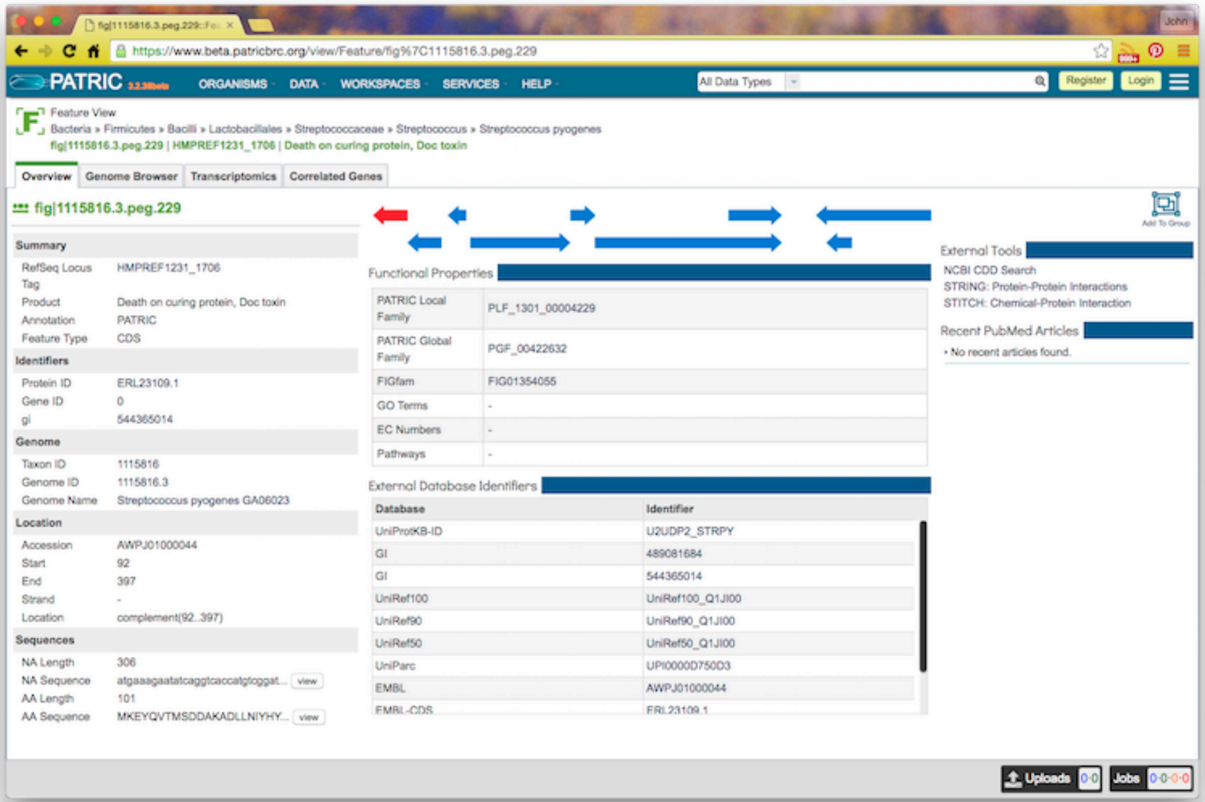
Pinned CDS selection [✓ Global families] Local families

Genome selection P3 Reference Genomes P3 Representative Genomes [✓ P3 Reference+Represen]

	Feature
ID	fig|1440772.3.peg.131
Genome	(0) Streptococcus pyogenes STAB901
Function	YheO-like PAS domain
Contig	CP007024
Start	137183
Stop	136815
Size	369 bp 123 aa
Set	1
pgfam	GF00076497: YheO-like PAS domain
plfam	Streptococcus.6419: YheO-like PAS domain

p3.theseed.org/qa/compare_regions/fig%7C1440772.3.peg.131
<< < draw > >>
(0) Streptococcus pyogenes STAB901
(1) Streptococcus pyogenes M1 GAS
(2) Streptococcus pyogenes strain NG
(3) Streptococcus canis FSL Z3-227
(4) Streptococcus dysgalactiae subsp.
(5) Streptococcus dysgalactiae subsp.

If you click the circled C on a feature, you will see a "Compare Regions" screen centered on that feature, like this:



Feature View

Bacteria » Firmicutes » Bacilli » Lactobacillales » Streptococcaceae » Streptococcus » Streptococcus pyogenes

fig|1115816.3.peg.229 | HMPREF1231_1706 | Death on curing protein, Doc toxin

fig|1115816.3.peg.229

Summary	
RefSeq Locus Tag	HMPREF1231_1706
Product	Death on curing protein, Doc toxin
Annotation	PATRIC
Feature Type	CDS
Identifiers	
Protein ID	ERL23109.1
Gene ID	0
gi	544365014
Genome	
Taxon ID	1115816
Genome ID	1115816.3
Genome Name	Streptococcus pyogenes GA06023
Location	
Accession	AWPJ01000044
Start	92
End	397
Strand	-
Location	complement(92..397)
Sequences	
NA Length	306
NA Sequence	atgaaagaatatcaggtcaccatgtggat... view
AA Length	101
AA Sequence	MKEYQVTMSDDAKADLLNIYHY... view

Functional Properties

PATRIC Local Family	PLF_1301_00004229
PATRIC Global Family	PGF_00422632
FIGfam	FIG01354055
GO Terms	-
EC Numbers	-
Pathways	-

External Database Identifiers

Database	Identifier
UniProtKB-ID	U2UDP2_STRPY
GI	489081684
GI	544365014
UniRef100	UniRef100_Q1JI00
UniRef90	UniRef90_Q1JI00
UniRef50	UniRef50_Q1JI00
UniParc	UPI0000D750D3
EMBL	AWPJ01000044
EMBL-CDS	ERL23109.1

External Tools

NCBI CDD Search

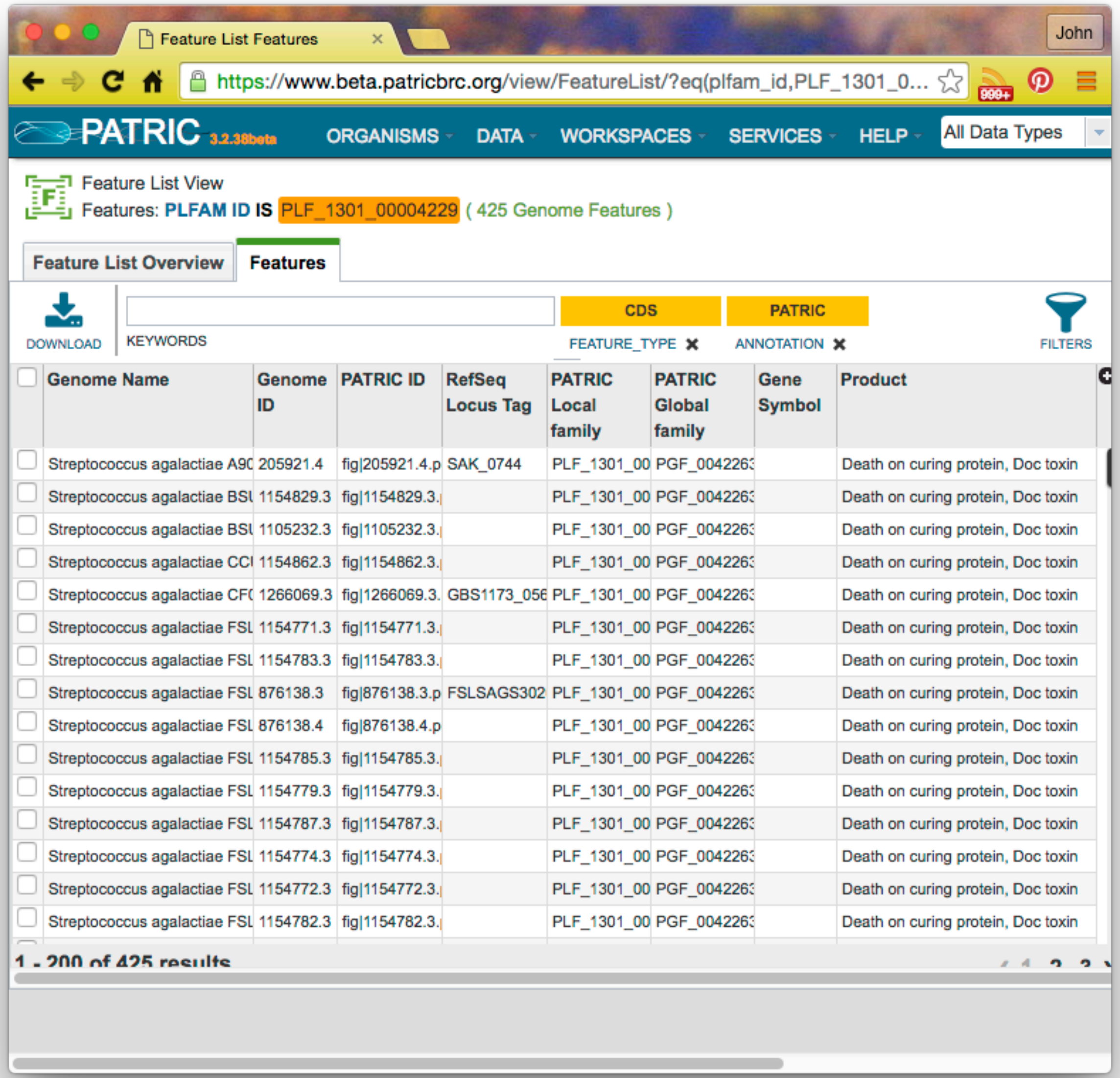
STRING: Protein-Protein Interactions

STITCH: Chemical-Protein Interaction

Recent PubMed Articles

- No recent articles found.

If you click on a family id, you will be taken to a Patric Family Page:



Feature List View
Features: PLFAM ID IS PLF_1301_00004229 (425 Genome Features)

Genome Name	Genome ID	PATRIC ID	RefSeq Locus Tag	PATRIC Local family	PATRIC Global family	Gene Symbol	Product
Streptococcus agalactiae A90	205921.4	fig|205921.4.p	SAK_0744	PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae BSU	1154829.3	fig|1154829.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae BSU	1105232.3	fig|1105232.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae CCI	1154862.3	fig|1154862.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae CF(	1266069.3	fig|1266069.3.	GBS1173_056	PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae FSL	1154771.3	fig|1154771.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae FSL	1154783.3	fig|1154783.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae FSL	876138.3	fig|876138.3.p	FSLSAGS302	PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae FSL	876138.4	fig|876138.4.p		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae FSL	1154785.3	fig|1154785.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae FSL	1154779.3	fig|1154779.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae FSL	1154787.3	fig|1154787.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae FSL	1154774.3	fig|1154774.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae FSL	1154772.3	fig|1154772.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin
Streptococcus agalactiae FSL	1154782.3	fig|1154782.3.		PLF_1301_00	PGF_004226		Death on curing protein, Doc toxin

1 - 200 of 425 results

Summary

We have implemented a tool that, given two sets of genomes, will compute the signature clusters that occur (or tend to occur) in genomes from one set but not in genomes from the other. The sets of genomes are tken from the current release of the PATRIC database.

We have illustrated one intended use: finding the signature clusters that distinguish a species from other species within a phylogenetic context (the genus).