Selectable Taxon Ortholog Retrieval Iteratively (STORI)
User's Guide

Welcome to the STORI! This algorithm is a new way to retrieve protein families. The unique aspect of our method is an iterative search of "family space". We consider a protein family and its potentially paralogous families as a Markov chain whose future state (future grouping of sequence accessions) depends only on the present state (present grouping of sequence accessions). After repeated iteration, the family membership can converge to a steady state. We assess convergence by measuring the agreement between two parallel chains, whose initial states were randomized. Because family optimization occurs iteratively, this algorithm bypasses precomputation of reciprocal best hits.

The first step is to make sure that Perl is configured properly[1]. The run environment for these scripts needs Perl to have access to several modules from CPAN: Statistics::Descriptive, Data::Dumper, List::MoreUtils, Time::Elapse, LWP::Simple, Bio::SeqIO, and Getopt::Long. If you do not have root access, and these modules are not already functional, then do a non-root Perl module installation to some location in your home directory. We've included a separate text file with the commands that worked on our system (nonroot-cpan.txt).

Look over the scripts in the STORI directory and change the file paths as appropriate for your system[2]. Here is a list of the different paths that STORI needs to run, as we configured them for our system. These directories are found at the beginning of at least one of each script:

```
/tmp/jstern7
/nv/hp10/jstern7/perl5/lib/perl5
/nv/hp10/jstern7
/nv/hp10/jstern7/STORI
/nv/hp10/jstern7/STORI/getParentTaxa.pl
/nv/hp10/jstern7/STORI/STORIcontrol_job_statistics.txt
/nv/hp10/jstern7/STORI/job_data_STORI.txt
/nv/hp10/jstern7/STORI/checkSTORI.pl
/nv/hp10/jstern7/STORI/checkSTORI-noseqs.pl
/nv/hp10/jstern7/STORI/continueSTORIfast_t.pl
/nv/hp10/jstern7/STORI/continueSTORI_48hr.pl
/nv/hp10/jstern7/STORI/beginSTORI.pl
/nv/hp10/jstern7/STORI/GetMissingSeqs.pl
/nv/hp10/jstern7/STORI/STORI-pbs_t
/nv/hp10/jstern7/STORI/taxids_GIs.txt
/nv/hp10/jstern7/STORI/makeblastdb
/nv/hp10/jstern7/STORI/blastp
/nv/hp10/jstern7/STORI/blastdbcmd
/nv/hp10/jstern7/STORI/bp_nrdb_SHA.pl
/nv/hp10/jstern7/STORI/STORI.pl
```

[1] When executing Perl scripts, it might be necessary to "module load perl" at the beginning of your terminal session depending on your computing environment. Note that 'module' in the context of this command is different from a Perl module that one would download from http://www.cpan.org/.

[2] We wrote this algorithm intending it for use on a cluster with the Torque/Moab job scheduler, although we see no reason why it could not be adopted for use with a different scheduler.

```
/nv/hp10/jstern7/scratch/universal120312
/nv/hp10/jstern7/scratch/universal120312/blast
/nv/hp10/jstern7/scratch/universal120312/hits
/nv/hp10/jstern7/clustalw21/clustalw2
/nv/hp10/jstern7/clustalo/clustalo
```

Also, make sure that every path refers to a file or folder that actually exists. If you run into difficulty with the setup, it is probably due to an incorrect path.

The next step to setting up STORI is building its database. Use getFastas.pl, getFastas.pbs, and taxids_GIs.txt. Be sure to make changes as applicable to your system (i.e. the file paths)[3]. Also, set up a project directory on a file system with fast read/write access[4], and create empty subdirectories called "blast", "fasta", and "hits". E.g., our project directory "scratch/universal120312" contains these three subdirectories.

Downloading the sequences for the default taxa list takes about 24 hours[5]. Once this script completes, the end of the file retrieval_log.txt will have a table showing the fraction of each taxon successfully downloaded. Some taxa may not have downloaded fully[6]. Protein sequences from these taxa can be downloaded manually from NCBI Protein. Go to www.ncbi.nlm.nih.gov/protein and paste the query part of the url (txidXX[orgn]) from the log file into the search field. Hit "Search." Click Send To>File>FASTA>Create File[7].

Cull the redundancy from the downloaded FASTA files, and turn them into BLAST databases using makeNr.pl[8,9]. After finishing[10] makeNr.pl, archive the project directory[11], and move the archive to a backup volume.

---

[3] These scripts depend on blastdbcmd, blastp, and makeblastdb, which are executables from NCBI's excellent BLAST suite, version 2.2.25+. They should work as is, but if you run into problems, see the documentation at: ftp://ftp.ncbi.nih.gov/blast/

[4] Actually, STORI is set up to copy the databases to a node's local /tmp volume, which should be faster than scratch. But this will only work if such a volume exists.

[5] Once getFastas.pl finishes downloading the default taxa set, the size of the fastas/ dir will be about 2 GB. To reiterate, please set $projectDir to a location in scratch space, because scratch disks are faster than normal storage, and STORI will make many random reads from $projectDir.

[6] You should also check the size of the files in the fasta directory using "ls –lht". If you know that some taxon has 15168 protein sequences at NCBI, but its FASTA file is only 142 KB, something went wrong. The automated retrieval of protein sequence data remains challenging (Stein, 2002; Dessimoz et al., 2012). An alternative to retrieval from NCBI is the Reference Proteomes from the Quest for Orthologs website.

[7] To upload these FASTA files from a local machine (Mac or PC) to a cluster, we use the SFTP client Cyberduck.

[8] Make sure that the hits/ directory contains a file for every taxon – else the downstream script getParentTaxa.pl will fail. As long as the getFastas.pl result was satisfactory, this will be fine.

[9] This script is mostly a wrapper for BioPerl's bp_nrdb.pl by Dr. Jason Stajich.

[10] Run time is an hour or so. makeNr.pl may fail to create the BLAST database for a taxon if this taxon's FASTA file deviates from the FASTA format. We encountered a problem with txid9 (*Buchnera aphidicola*) because an entry for GI # 15616631 contained two carriage returns. We deleted this entry by hand and re-ran the script.

[11] E.g., tar -czf universal120312.tar.gz universal120312

STORIcontrol is for starting, stopping, or pausing runs. STORIstats is for checking progress and viewing results[12]. STORIcontrol and STORIstats are meant to run occasionally on a head node[13].

In a typical use of STORIcontrol, we launch it from the shell with

```
>perl ~/STORI/STORIcontrol.pl
```

Next, we start a retrieval:

```
STORI>start <run-name> <scratch/dir> <taxa file> <windowSize>
<finalMaxFams>
```

For example, we can retrieve the Bacterial tRNA synthetases with the command:

```
STORI>start all_tRNAsynthetase_bact_1x
/nv/hp10/jstern7/scratch/STORI_runfiles bacteria 4 50
```

The name of the run is "all_tRNAsynthetase_bact_1x". Its data files will be stored in /nv/hp10/jstern7/scratch/STORI_runfiles[14]. For this run, STORI will use the Taxon IDs specified in the text file taxa-master[bacteria].txt[15]. The size of the search window is 4 taxa. The maximum number of allowable families is 50.

STORI makes a request of us:

```
Please enter an expression to match with protein names:
```

We enter:

```
Please enter an expression to match with protein names:
tRNA\s[sS]ynthetase
```

STORI uses Perl regular expressions; in this case, matches will be protein names containing either "tRNA synthetase" or "tRNA Synthetase"[16].

Next, STORI asks:

```
what offset factor? (usually 3)
```

and we enter:

```
what offset factor? (usually 3)
```

---

[12] We added some "pre-alpha" functions to STORIstats for comparing family distance, which require Clustalw, Clustalo, Belvu, and ssearch36. (STORIstats will still report retrieval results if these programs are not installed.)

[13] If doing more extensive distance comparisons, run STORIstats on a compute node in an interactive session.

[14] Note that this path was absent from the earlier list and that in this example we had previously created the run directory, i.e. `mkdir ~/scratch/STORI_runfiles`.

[15] The taxa files need to be in the same directory as the STORI scripts, and should be named according to the format: "taxa-master[<user specified clade name>].txt". Note that STORI will have problems if an ID in this taxa file does not have a corresponding BLAST database or hitDir file.

[16] We developed STORI for research purposes. To use this code in a production environment, one would need to improve the front end. User inputs to a Perl script can be exploited to compromise network security.

```
1
```

(We'll explain offset factor below.) STORI next uses blastdbcmd to search the FASTA deflines for our input string. From the matching entries, STORI picks two randomized samples, each containing `<finalMaxFams>` sequences[17], and will use the protein sequences of each sample as the initial state of two independent chains.

```
satisfied?
yes
```

(We could have typed "no" to repeat the search.)

```
3 2 1>blastoff
```

STORI begins two parallel, independent runs. Each chain is a serial PBS job submitted using msub.

Now let's try retrieving Eumetazoan hemoglobin.

```
STORI>start hemoglobin_eumetazoa_1x_STORI
/nv/hp10/jstern7/scratch/STORM3_runfiles eumetazoa 4 20
[...]
Please enter an expression to match with protein names: [hH]emoglobin
```

Hemoglobin presents in nearly every Eumetazoan, but what is its evolutionary provenance? Is it possible that hemoglobin resulted from a gene duplication prior/during Eumetazoa radiation, and that the evidence of this duplication remains in the form of a lower-eukaryote paralog? Let us attempt to find out...

```
start hemoglobin_euk_8x_STORI /nv/hp10/jstern7/scratch/STORM3_runfiles
eukaryota 4 20
```

Previously, our offset factor was 1, but here it will be 8. This change makes the initial state of the chains more influential to the rest of the run. We have found that adding influence to these initial seed sequences can prevent families from disappearing during iteration. Such disappearance is common when a protein is absent from a large portion of the subject taxa. For Eumetazoa, the seeds do not need a "handicap", because there won't be much opportunity for more conserved families to push them out. However, when the subject taxa are a diverse selection of Eukaryotes, the conserved families may push out hemoglobin.

To stop a run, we could type[18]:

```
stop hemoglobin_euk_8x_STORI
```

[17] Taxa are randomly picked without replacement until the # of sequences is ≥ the maximum number of families (a value specified by the user).

[18] This feature has not been tested thoroughly and should be used with care.

STORIcontrol should be run about once a day, depending on the parameters of the retrieval runs. STORIcontrol is responsible for judging convergence, and it can run in background (using GNU screen). If not running in background, it is fine to just run periodically[19].

Now we will run STORIstats.pl to check on the progress of our runs. Before doing so it is usually good to run STORIcontrol.pl once, so that the file job_data_STORI.txt is updated[20].

```
>perl ~/STORI/STORIstats.pl
```

The most important commands are `show`, `summarize`, `annotate`, and `rename`. These commands are best explained by example:

```
STORI> show runs
showing the runs
1: hemoglobin_eumetazoa_1x_STORI 0.85
2: all_tRNAsynthetase_bact_1x_STORI 0.77
3: hemoglobin_euk_8x_STORI 0.51
(0 converged runs)
(0 paused runs)
STORI> summarize hemoglobin_eumetazoa_1x_STORI
12 families added to clipboard.
STORI> Name 6
[...]
STORI> show clipboard
0: hemoglobin_subunit_zeta
3: myoglobin_Danio_rerio
4: hemoglobin_eumetazoa_1x_STORI_orphph53_0
6: hemoglobin_eumetazoa_1x_STORI_orphh162_3
7: hemoglobin_subunit_alpha
9: PREDICTED_hemoglobin_subunit
11: cullinassociated_NEDD8dissociated_protein
STORI> annotate 3
[...]
STORI> rename 3 myoglobin
STORI> annotate 4
[...]
STORI> rename 4 cytoglobin
STORI> annotate 6
[...]
STORI> rename 6 neuroglobin
STORI> annotate 9
[...]
STORI> rename 9 hemoglobin_epsilon
STORI> annotate 11
[...]
STORI> show clipboard
```

[19] For users familiar with MrBayes, the "chain swapping" step of STORI is facilitated by STORIcontrol; therefore, this script must either run in background on the head node, or be manually run by the user about once daily. STORIcontrol must run repeatedly in order for the runs to run.

[20] However, if STORIcontrol submits any new PBS jobs, then it may take a few hours for data from their corresponding runs to be accessible to STORIstats.

```
0: hemoglobin_subunit_zeta
3: myoglobin
4: cytoglobin
6: neuroglobin
7: hemoglobin_subunit_alpha
9: hemoglobin_epsilon
11: cullinassociated_NEDD8dissociated_protein
STORI> show clipboard -all eumetazoa.txt
showing entire clipboard using org file eumetazoa.txt
[...]
```

What we did is take STORI's latest forecast of family organization and save it to a clipboard. We had STORIstats attempt to name each family automatically, and we corrected its mistakes by looking at the deflines ourselves and using our brains. Then we outputted the clipboard with a formatting amenable to copying and pasting in Excel or OpenOffice. To download an alignment, we could head over to http://www.ncbi.nlm.nih.gov/tools/cobalt/ and submit the accessions from one of the families. Note that the clipboard will disappear when we close STORIstats.

Eventually[21], these runs will converge, at which point they will no longer be displayed as an active run. They will be accessible with the command "`show converged`".

Caveat Emptor

1. Use of msub, Torque, Moab. Changes in command syntax for this software could break STORI in pretty obvious ways; eg, jobs don't start
2. As a corollary of #1, STORIcontrol relies on PBS run files to be in a specific location and to have resource use information output to them in a specific format.
3. Low tolerance for typos
4. Stop command needs more testing.
5. Future algorithmic improvement: the Merge and MergeRecursive subroutines in STORI.pl are costly, because they exhaustively compare family members with one another. For very large numbers of taxa, the Merge function may not complete before the wall-clock limit is reached. This will not be disastrous, but to avoid it, the wall-clock limit should be adaptive to the number of taxa desired and the number of families desired.

[21] For the runs in this example, probably 10 days. Other runs could take longer or shorter. If you want something fast, make a new taxa list of 20 archaea and retrieve 4 highly conserved families. This run should finish in less than 2 days, and it would be best to keep STORIcontrol running the whole time.