**The Large Scale Blast Score Ratio (LS-BSR) pipeline**

**Citation**

Jason W. Sahl, J. Gregory Caporaso, David A. Rasko, Paul S. Keim (2014). The large-scale blast score ratio (LS-BSR) pipeline: a method to rapidly compare genetic content between bacterial genomes. PeerJ PrePrints 2:e220v1.

**What does it do?**

The LS-BSR pipeline was designed as a way to quickly compare the genetic content between a large number of bacterial genomes. LS-BSR can calculate several pan-genome statistics in a population and the output can be easily visualized with a variety of third-party tools. Additionally, LS-BSR can be used to query a set of genes against a large set of genomes to identify gene distribution and conservation. LS-BSR was developed to be easy to run and interpret.

**Installation**

-The code is kept here:

<https://github.com/jasonsahl/LS-BSR.git>

-You can clone the repository to your own system with git:

$git clone <https://github.com/jasonsahl/LS-BSR.git>

-Enter the directory, then:

$python setup.py install

-if your install directory is /Users/jsahl/LS-BSR, run:

$export PYTHONPATH=/Users/jsahl/LS-BSR:$PYTHONPATH

-You can add this to your .bashrc or .profile

-You can test your installation by running the tests:

$python /Users/jsahl/LS-BSR/tests/test\_all\_functions.py

-If your installation is correct, all tests should pass

**Dependencies**

1. USEARCH (tested version is 6.0.307), path is passed as command-line option - **only required if a set of gene sequences is not supplied**. 32-bit version should be sufficient for most applications, including the analysis of 1000 *E. coli* genomes. Tested successfully with versions v7.0.959 and 7.0.1090. Can be freely obtained for academics/non-profits from: <http://www.drive5.com/usearch/>
2. BioPython, must be in $PYTHONPATH environmental variable. Can be freely obtained from: <http://biopython.org/wiki/Main_Page>
3. blastall (tested version is 2.2.25), must be in path as 'blastall'. Known issues seen with v2.2.26. 2.2.5 can be obtained from: <ftp://ftp.ncbi.nlm.nih.gov/blast/executables/release/2.2.25/>
4. Prodigal (tested version is 2.60), must be in path as 'prodigal' - **only required if a set of gene sequences is not supplied**. Can be obtained from: <https://code.google.com/p/prodigal/>
5. Numpy, must be in PythonPath. **Numpy is only required for the post-process compare matrices tool**. If you don't want to install numpy, you should be fine, but won't be able to run the compare script.

**Command Line options**

**-d DIRECTORY**, --directory=DIRECTORY : the directory to your fasta files, all must end in ".fasta". Can either be complete genomes or draft assemblies. Scaffolds are discouraged.

**-i ID**, de-replication clustering value for USEARCH, defaults to 0.9 (range from 0.0-1.0)

**-f FILTER**, whether to use BLAST filtering, default is "F" or filter, turn off with "T". Turning this to “T” should speed up the analysis, but may throw out highly repetitive sequences.

**-p PROCESSORS**, number of processors to use, defaults to 2.

**-g GENES**, if you have a list of genes to screen, supply a nucleotide fasta file. Each gene sequence must be in frame, or questionable results will be obtained (only true for TBLASTN). If this flag is not envoked, then the de novo gene prediction method is envoked

-**b BLAST**, which blast method to use. Default is 'tblastn', can be changed to 'blastn'. Can be used with either a list of supplied genes, or with the *de novo* method.

**-q PENALTY**, blast mismatch penalty, default is -4, only works with blastn. Optimized to return longer matches. Only certain q/r ratios are allowed. See BLAST documentation for more details.

**-r REWARD**, blast reward value, default is 5, only works with blastn. Optimized to return longer matches. Only certain q/r ratios are allowed. See BLAST documentation for more details.

**-l LENGTH**, minimum BSR value to be called a duplicate, defaults to 0.7. The BSR of the "duplicate" divided by the reference bit score must be greater than this value to be called a duplicate

**-m MAX\_PLOG**, maximum value to be called a paralog, defaults to 0.85. If the BSR value is greater than this value, then it is considered to be an ortholog

**-n MIN\_HLOG**, minimum BLAST ID to be called a homolog, defaults to 75. If the BLAST ID is below this value, it is considered a remote homolog

**-t F\_PLOG**, filter ORFs with a paralog from BSR matrix? Default is F (do not filter), values can be T (filter paralogs) or F

**-k KEEP**, keep or remove temp files, choose from T or F, defaults to False (F)

**Test data – give LS-BSR a whirl on small datasets**

-Test data is present in the test\_data directory. This data consists of:

- Genomes (4 *E.coli* genomes from 4 different pathogenic variants). Genomes are:

* H10407 - enterotoxigenic *E. coli* (ETEC)
* E2348/69 - enteropathogenic *E. coli* (EPEC)
* O157:H7 sakai - shiga toxin *E. coli* (STEC)
* SSON046 - Shigella sonnei

-Genes (5 different markers that delineate between the variants). These include:

* IpaH3 - Shigella invasion antigen. Mostly present in *Shigella* spp.
* LT - heat-labile toxin. Only present in ETEC (not all)
* ST2 - heat-stable toxin. Only present in ETEC (not all)
* bfpB - bundle forming pilus. Only present on plasmid in EPEC
* stx2a - shiga toxin. Present in STEC

-You can test out the LS-BSR functionality in 4 different ways:

1. Test the gene screen method with tblastn:

-enter test\_data directory, run LS-BSR

$python /Users/jsahl/LS-BSR/ls\_bsr.py -d genomes -g genes/ecoli\_markers.fasta

-the output should show how each gene is only present in the correct pathovar

1. Test the gene screen method with blastn:

-enter test\_data directory, run LS-BSR

$python /Users/jsahl/LS-BSR/ls\_bsr.py -d genomes -g genes/ecoli\_markers.fasta -b blastn

1. Test the de novo gene prediction method:

-enter test\_data directory

-run LS-BSR

python /Users/jsahl/LS-BSR/ls\_bsr.py -d genomes -u /usr/local/bin/usearch6

-To inspect the output, you can look up the following entries in the BSR matrix. They should correspond with the results obtained with the gene screen methods (TBLASTN only):

IpaH3 -> centroid\_1724 LT -> centroid\_11953 ST2 -> centroid\_19265 bfpB -> centroid\_1922 stx2a -> centroid\_7471

Sample output for each method is shown in the test\_data directory

\*Disclaimer

TGen and ITS Affiliates, representatives and employees make no representations, warranties, guarantees, or claims, of any kind or nature, as to the accuracy of, reliability, utility, performance, effectiveness, or otherwise, of the Software or the results obtained therefrom, nor do they assume any responsibility for the results, quality of results, or lack of results. THE SOFTWARE IS PROVIDED "AS IS" AND WITHOUT WARRANTY OF ANY KIND, WRITTEN OR ORAL, EXPRESS OR IMPLIED, DIRECT, INDIRECT, BY ESTOPPEL, OR OTHERWISE, AND SPECIFICALLY EXCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT OF ANY THIRD PARTY PATENT OR OTHER INTELLECTUAL PROPERTY RIGHTS, AND ANY OTHER WARRANTIES PROVIDED FOR BY THE UNIFORM COMMERCIAL CODE, USAGE IN THE INDUSTRY, OR COURSE OF DEALING BETWEEN THE PARTIES. TGEN AND IBBL DO NOT WARRANT THAT USE OF THE SOFTWARE WILL BE ERROR FREE, UNINTERRUPTED, SECURE, OR VIRUS-FREE. TO THE EXTENT TGEN AND IBBL MAY NOT AS A MATTER OF LAW DISCLAIM ANY WARRANTY, THE PARTIES AGREE THAT THE SCOPE AND DURATION OF ANY SUCH WARRANTY SHALL BE THE MINIMUM PERMITTED UNDER APPLICABLE LAW.

LIABILITY LIMITATION

TGEN SHALL NOT BE LIABLE FOR ANY DAMAGES WHATSOEVER, INCLUDING ACTUAL OR DIRECT DAMAGES, AND SHALL NOT IN ANY EVENT BE LIABLE FOR ANY INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF INAPPROPRIATE, SUBSTITUTE, OR ALTERNATIVE, SOFTWARE, LOSS OF USE, LOSS OF DATA, LOSS OF PROFITS, LOSS OF REVENUES, OR BUSINESS INTERRUPTION) HOWEVER CAUSED, RESULTING FROM, RELATED TO, OR ARISING OUT OF THE USE OF THE SOFTWARE, REGARDLESS OF FORESEEABILITY AND NOTWITHSTANDING WHETHER RECIPIENT HAS BEEN ADVISED OF THE POSSIBILITY OF ANY SUCH DAMAGES, AND REGARDLESS OF THE FORM OF THE CAUSE OF ACTION, WHETHER IN CONTRACT, BREACH OF WARRANTY, OR TORT (INCLUDING NEGLIGENCE, STRICT LIABILITY, OR OTHERWISE), OR OTHERWISE.