Icahn School of Medicine at Mount Sinai LINCS Center for Drug Toxicity Signatures

Standard Operating Procedure:
Computation of Protein Structural Signatures from Differential Gene Expression Data

DToxS SOP Index: M-1.0
Last Revision: 11/15/2016
Written By: Rayees Rahman, Avner Schlessinger

Approvals (Date): Joseph Goldfarb (DATE)
Marc Birtwistle (DATE)
Eric Sobie (DATE)
Ravi Iyengar (DATE)

Quality Control (QC) steps are indicated with green highlight.

Metadata recording is highlighted with yellow highlight and superscript indices.

How `structural-signatures.sh` works:

a) Given a supplied gene list or dataset, `structural-signatures.sh` will first look up structural features for the protein coded for by each gene in the precomputed databases downloaded. Or given a DEG data file, `structural-signatures.sh` will select the top *n* genes sorted by lowest p-value and look up structural features for the protein coded for by each gene in the precomputed database downloaded. Where *n* is the value of the number of genes option (-g) in `structural-signatures.sh`

b) Next, if the -b option is used, `structural-signatures.sh` will then look up structural features for each gene in the precomputed database for a random set of *n* genes. Where *n* is the value of the number of genes option (-g) in `structural-signatures.sh` or the total number of genes in a gene list file.

c) `structural-signatures.sh` will then repeat the previous step *i* number of times. Where *i* is the number of bootstraps specified.
   i) This is done to generate a random background distribution of structural features of the same size as the number of input genes for statistical testing.
   ii) Unless -p is specified, `structural-signatures.sh` will do this step one at a time.

d) Finally, `structural-signatures.sh` will perform several calculations and statistical tests to produce the final outputs.

1) BASH, version 4.1.2, shell environment is required for this protocol. This protocol has been tested using the following BASH environments.
   a) Ubuntu 14.04 LTS
   b) CentOS 6.2
   c) Windows 10 using Cygwin 2.0.4
   d) Windows 10 using Windows Subsystem for Linux
2) Download and install the following programs for either Windows or Linux:
   a) R[1] (version 3.3.0 was used to generate signatures)
   b) Perl[2] (version 5.10 was used to genertate signatures)
3) Go to https://www.dropbox.com/s/4w8cv27l4emgecq/structural-signatures-111116.tar.gz?dl=0 and download the latest version of the gzip compressed structural signatures pipeline, called: `structural-signatures.111116.tar.gz`. This file includes:
   a) The scripts for the structural signatures pipeline, which takes either a gene list or a DEG dataset file as described in **DToxS SOP CO-4.1 Identification of Differentially Expressed Genes**, and outputs the enrichment of several structural features.
   b) Example files to run the analysis on.
   c) An installation script, explained in *Step 6.*
4) Using a system shell navigate to the directory where `structural-signatures.111116.tar.gz` was downloaded and use this command to extract the file:
   `tar -xzf structural-signatures.111116.tar.gz`
   This command will create a new directory. To verify enter the `ls` command into the shell. There should be a directory called `structural-signatures/` (QA/QC1).
5) Enter the `structural-signatures/` directory generated in *Step 4* using the command:
   `cd structural-signatures/`
   There should be the following files and folders:
   a) `test_data/` folder containing example inputs to the pipeline.
   b) `bin/` folder containing scripts and data needed to run the pipeline.
   c) `install_structural-signatures.sh` file to install the databases and set the environmental variables needed to run the structural signatures program.
   d) `README.txt` file containing further information to install and run the structural signatures program.
   e) `structural-signatures.sh` the main structural signatures program that generates data describing the enrichment of several structural features given a gene list or a DEG dataset file.
   To verify that these files and folders are present, enter the `ls` command into the shell (QA/QC1).
6) Internet access is required for this step. To install the required databases to run the structural signatures pipeline you must run the `install_structural-signatures.sh` file by using the command:
   `./install_structural-signatures.sh`
   This command will download the necessary files and set important environmental variables to run the pipeline.

This script will install the pipeline to the current directory. To install the pipeline in another directory, move the `structural-signatures/`directory to your directory of choice and input the `./install_structural-signatures.sh` command.

When running the installation script, the following message will be shown:

The default answer is 'no'. To begin installation input any one of these commands: `y, yes, YES, Y` or `Yes` and press enter. The script will run for several minutes and many messages will come up during the installation, the installation is complete when the following message is seen:

```
❯❯❯ ./install_structural-signatures.sh
Checking environment
This script downloads databases and sets up the environmental variables for the structural
signatures pipeline.
This script may take a while to finish, are you sure you want to continue? [y/N]
```

```
Total bytes read: 3026186240 (2.9GiB, 9.4MiB/s)
Creating install.directory
Done! Please read the README or use the -h option for help running structural-signatures.sh
```

If there is an error message see QA/QC2 for help.

7) To verify that the installation was successful enter the `ls` command into the shell. There should now be a `Database/` directory and an `install.directory` file which is used by the structural signatures pipeline. (QA/QC3)

   The directory has precomputed structural features generated by using the HHPRED[3], Predict Protein[4] , IUPRED[5] and COILS[6] software on the human proteome obtained from UniProt[7]

8) Running `structural-signatures.sh`

After installation, `structural-signatures.sh` can now be run by entering:

`./structural-signatures.sh`

This will produce the following output:

```
Working Directory: /mnt/c/Users/rayee/Desktop/3_schles/lincs/nov_2016/sop/structural-signatures
Usage:
        -d Dataset
               *required*
        -n Name (prefix) for output files
               *required*
        -b number of Bootstraps
               <default 0>
        -g number of differencially expressed Genes sorted by p-value
               <default all differencitally expressed genes>
        -p number of Parallel bootstraps to run
               <default 1>
        -l switch if gene List is being used instead of DToXs data
               *required if a gene list is being used*
        -t Type of genes in DToXs data <OVER> <UNDER> <BOTH>
               *Required if input is DToXs*
        -h print Help
```

If any other output is shown see QA/QC4 for more information.

a) Running `structural-signatures.sh` requires the following inputs.
   i) Either a DEG data set file as described in **DToxS SOP CO-4.1 Identification of Differentially Expressed Genes** or a gene set file with UniProt identifiers.
   ii) A name for the output files **[prefix].**

b) Depending on the type of input, different arguments must be given.
   i) If a DEG data set file is used, two arguments must be specified:
      (1) The type of genes must be specified using the -t argument.
      Either overexpressed genes, <OVER>.
      Under expressed genes, <UNDER>.
      Or both, <BOTH>.
      (2) The number of genes to analyze using -g. If no value is given, all genes will be analyzed.
      (3) An example run using the DEG dataset file: Human.D-Hour.48-Plate.4-Calc-CTRL.SOR.tsv and analyzing the top 100 overexpressed genes with the output filename **[prefix]** "CER" would use the command:
      `./structural-signatures.sh -d Human.D-Hour.48-Plate.4-Calc-CTRL.CER.tsv  -t OVER -n CER -g 100`
   ii) If a gene set file is used one argument must be specified:
      (1) The switch -l must be specified.
      (2) The gene set must use UniProt identifiers and must have each gene separated by a newline ("\n"). An example of such a file can be seen in the `test_data/apoptosis.gene.list` file which is a list of signature genes found in the apoptosis pathway. This file can be obtained from the Molecular Signatures Database[8].
      (3) Note that the -t and -g arguments are not used.
      (4) For example, if we were to obtain the enrichment of structural features for the `apoptosis.gene.list` file in the `test_data/` directory and have the output

file [**prefix]** as “apoptosis-structural-features” the following command would be used:

```
./structural-signatures.sh -d test_data/apoptosis.gene.list
-l -n apoptosis-structural-features
```

c) `structural-signatures.sh` has two modes: one computing 3D structural features only, and the other computing both 3D structural features and 2D features.

3D structural features are the enrichment of Class, Fold, Superfamily and Family features as described by the SCOPe[9] hierarchy.

2D features are the enrichment of helical, sheet, loop and disordered residues as well as the enrichment of coiled-coil motifs, transmembrane helices and disordered regions for a set of genes. It is recommended to use a bootstrap of at least 100 to generate a representative null distribution.

i) To run 3D structural features only mode, **do not** give an argument to -b.
ii) To enter the second mode, give an argument to -b.
   (1) The argument -b specifies the number of bootstraps to generate a null distribution for calculating p-values for 2D features. If no argument is given to -b, then no null distribution is generated for the 2D features, thus no enrichment of these features may be calculated. See *Steps 9a viii-xi.*

The previous two examples both showed command examples for `structural-signatures.sh` run in the first mode. An example run of `structural-signatures.sh` to obtain 2D structural features using the `Human.D-Hour.48-Plate.4-Calc-CTRL.CER.tsv` dataset with 1000 bootstraps, measuring the top 500 overexpressed genes and having the output files with the **[prefix]** “D-CER” would use the command:

```
./structural-signatures.sh -d Human.D-Hour.48-Plate.4-Calc-
CTRL.CER.tsv -b 1000 -g 500 -t OVER -n D-CER
```

d) Depending on the number of bootstraps and the number of genes analyzed per bootstrap, `structural-signatures.sh` can take several hours to run.

i) You can parallelize the bootstrapping process in `structural-signatures.sh` to run faster by using the -p argument. For example, to run the previous command 10 bootstraps at a time, we can use the following command:

```
./structural-signatures.sh -d Human.D-Hour.48-Plate.4-Calc-
CTRL.IMA.tsv -b 1000 -g 500 -t OVER -n D-CER -p 10
```

By default, `structural-signatures.sh` will run 1 bootstrap per iteration. The maximum number of parallel bootstraps is the total number of cpu cores the computer has. Exceeding this number may not improve overall compute time.

9) `structural-signatures.sh` output:

a) `structural-signatures.sh` produces several output files placed in the `output/` directory. They are as follows:

i) **[prefix].info**
Contains the class, fold, families, superfamilies, number of secondary strucuture elements, number of disordered proteins, number of disordered regions, number of coils and number of transmembrane helicies information about each protein coded for in the supplied gene list or dataset processed in a "|" separated format.

ii) **[prefix].class.csv**
Contains the counts, fold change and pvalue for each of the structural classes found in proteins coded for in the supplied gene list or dataset in a comma separated format.

iii) **[prefix].folds.csv**
Contains the counts, fold change and pvalue for each of the structural folds found in proteins coded for in the supplied gene list or dataset in a comma separated format.

iv) **[prefix].superfamily.csv**
Contains the counts, fold change and pvalue for each of the structural super-families found in proteins coded for in the supplied gene list or dataset in a comma separated format.

v) **[prefix].family.csv**
Contains the counts, fold change and pvalue for each of the of structural families found in proteins coded for in the supplied gene list or dataset in a comma separated format.

vi) **[prefix].tot.genes.with.disordered.regions**
Contains the total number of proteins with disordered regions coded for in the supplied gene list or dataset.

vii) **[prefix].tmh.total**
Contains the total number of transmembrane helices found in proteins coded for in the supplied gene list or dataset.

viii) **[prefix].coils.csv**
Contains the fold change and pvalue of the representation of coiled-coil regions in the proteins coded for by the supplied gene list or dataset. This file is only present if the -b option is used.

ix) **[prefix].disordered.residues.csv**
Contains the fold change and pvalue of the representation of disordered residues in the proteins coded for in the supplied gene list or dataset in comma separated format. This file is only present if the -b option is used.

x) **[prefix].disordered.regions.csv**
Contains the fold change and pvalue of the the representation of disordered regions greater than 30 residues in the proteins coded for in the supplied gene list or dataset in comma separated format. This file is only present if the -b option is used.

xi) **[prefix].secondary.structure.csv**
Contains the fold change and pvalue of the of helices (represented by the letter H), strands (represented by the letter E) and loops (represented by the letter L) in the supplied gene list or dataset in a comma separated format. This file is only present if the -b option is used.

10) Error messages in `structural-signatures.sh`
   a) For common error messages see QA/QC5 for more information.

11) How `structural-signatures.sh` works:
   a) Given a supplied gene list or dataset, `structural-signatures.sh` will first look up structural features for the protein coded for by each gene in the precomputed databases

downloaded. Or given a DEG data file, `structural-signatures.sh` will select the top *n* genes sorted by lowest p-value and look up structural features for the protein coded for by each gene in the precomputed database downloaded. Where *n* is the value of the number of genes option (-g) in `structural-signatures.sh`

b) Next, if the -b option is used, `structural-signatures.sh` will then look up structural features for each gene in the precomputed database for a random set of *n* genes. Where *n* is the value of the number of genes option (-g) in `structural-signatures.sh` or the total number of genes in a gene list file.

c) `structural-signatures.sh` will then repeat the previous step *i* number of times. Where *i* is the number of bootstraps specified.
   i) This is done to generate a random background distribution of structural features of the same size as the number of input genes for statistical testing.
   ii) Unless -p is specified, `structural-signatures.sh` will do this step one at a time.

d) Finally, `structural-signatures.sh` will perform several calculations and statistical tests to produce the final outputs.

Metadata

1. **R**
   Version 3.3.1 “Bug in Your Hair”
   http://www.r-project.org
2. **Perl**
   Version 5.18.2
   http://www.perl.org
3. **HHSuite**
   Version 3.0
   https://toolkit.tuebingen.mpg.de/hhpred/help_ov
4. **Predict Protein**
   https://www.predictprotein.org/
5. **IUPRED**
   http://iupred.enzim.hu
6. **COILS**
   Version 2.2
   http://embnet.vital-it.ch/software/COILS_form.html
7. **UniProt**
   http://uniprot.org
8. **Molecular Signatures Database**
   Version 5.2
   http://software.broadinstitute.org/gsea/msigdb/
9. **SCOP Extended**
   Version 2.06
   http://scop.berkeley.edu/

Quality Assurance/Control Steps (QC)

QA/QC1: Verification that structural signatures package has been correctly extracted.

QA/QC2: Error messages from running `install_structural-signatures.sh`

1)

```
R/Rscript is required to run this pipeline.
Please install the latest version of R and make sure Rscript is accesible from the command line.
```

The statistical software R must be installed to run this script.

Solution: On Ubuntu or Windows Subsystem for Linux you may install R using this command:

```
sudo apt-get -y install r-base
```

Otherwise go to https://cran.r-project.org/ to install R for your system.

2)

```
It seems like structural-signatures.sh is already installed!
Aborting
```

This error occurs if you run `install_structural-signatures.sh` more than once in a specific directory.

Solution: To reinstall `structural-signatures.sh,` delete the present directory and follow *Steps 4-6*

3)

```
Incorrect input, aborting
```

An incorrect input was given at the start of the script.

Solution: The only vaild inputs are: `n,N,NO,No,no,Y,YES,y,yes,Yes.`

QA/QC3: Verification that `install_structural-signatures.sh` worked as expected.

QA/QC4: Error messages when running `structural-signatures.sh` without any input parameters.

1)

```
In order run structural_signatures.sh you must specify these directories in this script:
        $IUPRED directory to IUPRED predictions
        $PROF directory to predict protein predictions
        $HHR directory to hhpred output
        $SECSTUCT directory to secstruct.sh
Please edit secstruct.sh to include these directories
Or run the install_structural-signatures.sh script
```

Various environmental variables must be specified in order for `structural-signatures.sh` to run. This error can be due to a varitey of reasons:

a) `install_structural-signatures.sh` was not run at all.

Solution: Run `install_structural-signatures.sh`

b) `install.directory` file was removed

Solution: Create a file called `install.directory` inside the `structural-signatures/` directory (where the `Database/` directory is). Inside the file input the full directory to the `structural-signatures/` directory.

c) Running `structural-signatures.sh` outside of the installation directory.

Solution: To run `structural-signatures.sh` outside of the installation directory you must then edit the source code of `structural-signatures.sh` and input the full directory to each of the databases between lines 17-21.

2)

```
R/Rscript is required to run this pipeline.
Please install the latest version of R and make sure Rscript is accesible from the command line.
```

See **QA/QC2-1**

QA/QC5: Error messages when running `structural-signatures.sh` with input parameters.

1) This message constantly showing:

```
Usage:
        -d Dataset
               *required*
        -n Name (prefix) for output files
               *required*
        -b number of Bootstraps
               <default 0>
        -g number of differencially expressed Genes sorted by p-value
               <default all differencitally expressed genes>
        -p number of Parallel bootstraps to run
               <default 1>
        -l switch if gene List is being used instead of DToXs data
               *required if a gene list is being used*
        -t Type of genes in DToXs data <OVER> <UNDER> <BOTH>
               *Required if input is DToXs*
        -h print Help
```

This is usually caused by forgetting to input a required parameter such as -n or -d.

Solution: Make sure you have all required parameters set.

2)

```
You need to specify what types of genes to extract from DToXs data using -t
        <OVER>expressed genes
        <UNDER>expressed genes
        <BOTH>
```

This is caused by not inputting the -t argument when using a DEG dataset.

Solution: When using a DEG dataset remember to set -t to either <BOTH>, <OVER> or <UNDER>.

3)

```
Number of parallels (-p) cannot be greater than the total number of bootstraps (-b)
See -h for help
```

This usually occurs when you set the number of parallel runs greater than the total number of bootstraps.

Solution: Set the number of parallel runs to less than the total number of bootstraps.

4)

```
It looks like you are not using a DToXs DEG file as input.
If you are inputing a gene list you must use the -l option!
**All genes must be UNIPROT identifiers otherwise they may not be discovered by the script!**
```

This is error typically triggers when you input a gene list but forget to input the -l option.

Solution: If you are using a gene list and not a DEG dataset don't forget to set the -l option as well!

5)

```
not recognized please use either OVER, UNDER or BOTH for -t (case sensitive)
```

This error happens when you give -t something other than OVER, UNDER or BOTH.

Solution: When using -t OVER, UNDER or BOTH are the only arguments, they are also case sensitive!

6)

```
X out of Y found
```

Where X and Y are numbers. This is not an error, but just a message that not all of the genes in the input data was found in the precomputed database. Typically the amount not found is < 10%, however if it is more than 10% double check if all the genes in the input gene list are UniProt Identifiers! You can use http://www.uniprot.org/uploadlists/ to convert between different gene names and identifiers to UniProt IDs.

Correspondance:

Rayeees Rahman
E-mail: rayees.rahman@icahn.mssm.edu

Avner Schlessinger
E-mail: avner.schlessinger@mssm.edu