TagGraph

Defining complex proteomes with string

TagGraph is the first algorithm of its kind which leverages the efficiency of FM-indexed sequence databases to rapidly and sensitively interpret large mass spectrometry data sets. One major advantage of this approach is its ability to discover and localize post-translational modifications and other amino acid alterations within peptide sequences, without anticipating them *a priori.*

The following describes basic installation and operation procedures for the TagGraph software, version 1.8.

Set up linux environment

1. If you are already running a compatible form of linux (so far, we've tested CentOS v 7), skip to step I.5
2. Install virtual machine infrastructure (E.g., hyper-v on windows 10)
   a. Follow general instructions here: https://www.tenforums.com/tutorials/2087-hyper-v-virtualization-setup-use-windows-10-a.html#Part1
   b. Follow supplemental instructions here: https://www.tenforums.com/tutorials/2291-hyper-v-vm-install-centos-linux-windows-10-a.html#Part1
3. Download CentOS operating system

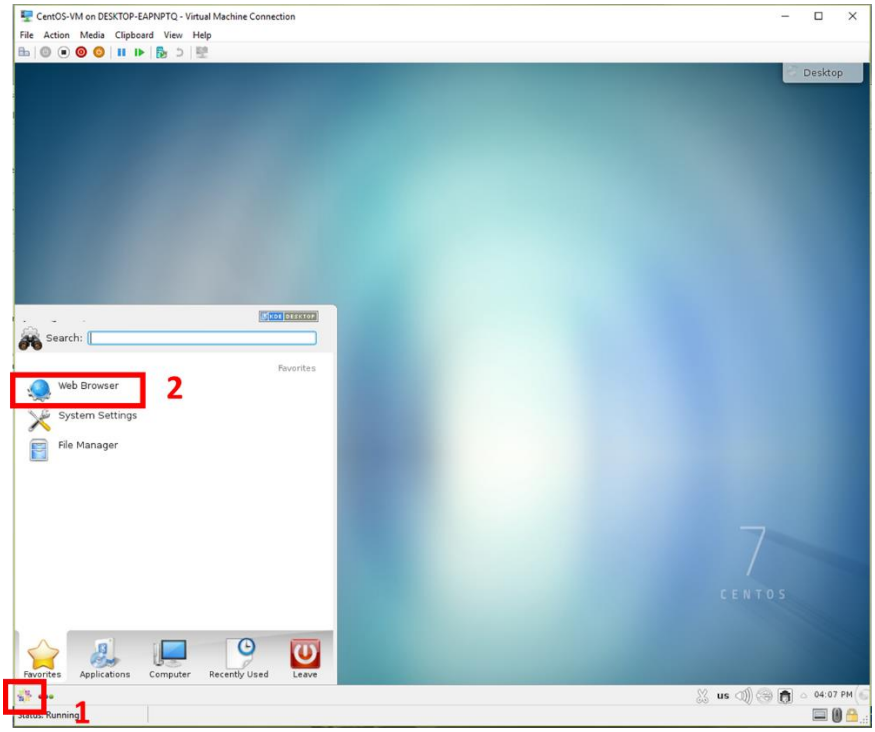
https://www.centos.org/download/

NB: Full download ~ 4.1 GB as of CentOS7.1611, March 21, 2017

4. Create CentOS virtual machine by continuing with the tutorial noted in Step 2b.

NB: be sure to follow instructions pertaining to internet connectivity, as this will be needed to install prerequisites for TagGraph, and TagGraph itself.

1. If your host computer is behind a firewall, you will need to register your virtual machine with the network. This can probably be accomplished by pointing the KDE web browser to any web page.



   a. Click Kickoff button (1)
   b. Select Web Browser (2)
   c. Enter any valid address in URL field (e.g., www.stanford.edu)

d. Register virtual computer with network if prompted.

5. Install non-standard packages with “yum” or pip commands which TagGraph will need
   A. pip (python installer)
      [sudo] easy_install pip
   B. lxml, numpy, pyteomics
      [sudo] pip install lxml==4.2.1
      [sudo] pip install numpy==1.7.1
      [sudo] pip install pyteomics==3.5.1
   C. sqlalchemy
      [sudo] pip install sqlalchemy
   D. pandas
      [sudo] pip install pandas==0.22.0
   E. pympler
      [sudo] pip install pympler==0.5
   F. MySQL-python
      [sudo] yum -y install MySQL-python
   G. networkx
      [sudo] pip install networkx==1.11
   H. pymzml
      [sudo] pip install pymzml==0.7.8
   I. matplotlib
      [sudo] pip install matplotlib

I. Install TagGraph

A. Download TagGraph program

A. Use web browser to navigate to taggraph repository: https://sourceforge.net/projects/taggraph/

B. Download:

i. Source code (TagGraph.1.8.tar.gz)

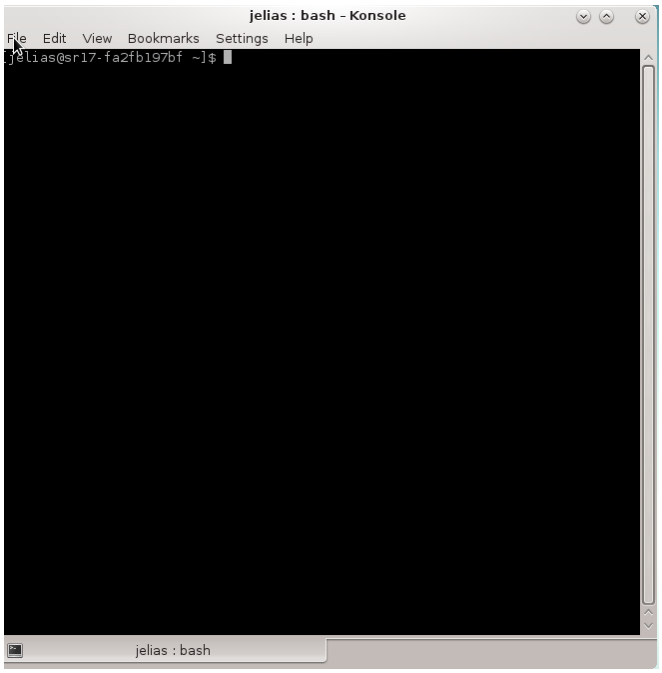
ii. Sample input (for testing): sampleInputFiles.1.8.tar.gz

iii. Sample output (for verification): sampleOutputFiles.1.8.tar.gz

C. Open Konsole terminal:

i. Kickoff button -> Search for “Konsole”

ii. Select Konsole



B. Verify software was installed correctly

A. Verify taggraph file was saved: by typing “ls”

B. Unzip TagGraph file:

tar –xzvf TagGraph.1.8.tar.gz

C. Navigate to the new folder that was created

D. Verify all required modules have been installed:

Run python checkPythonPackages.py

if any packages are missing, run the indicated command to install it or seek assistance from your system administrator.

II. Index fasta file for FM-index search

A. Download or transfer valid fasta-formatted protein sequence file
   a. E.g., the file “human_uniprot_12092014_crap.fasta used in the TagGraph manuscript, and included in the sampleOutputFiles.tar.gz download
   b. Download other fasta-formatted protein sequences from http://www.uniprot.org

B. Run FM indexer:

   python BuildFMIndex.py –f <fasta location> -o <desired output location with prefix Of the FMIndex file; default is to save the fasta file to the fmindexer’s working directory>

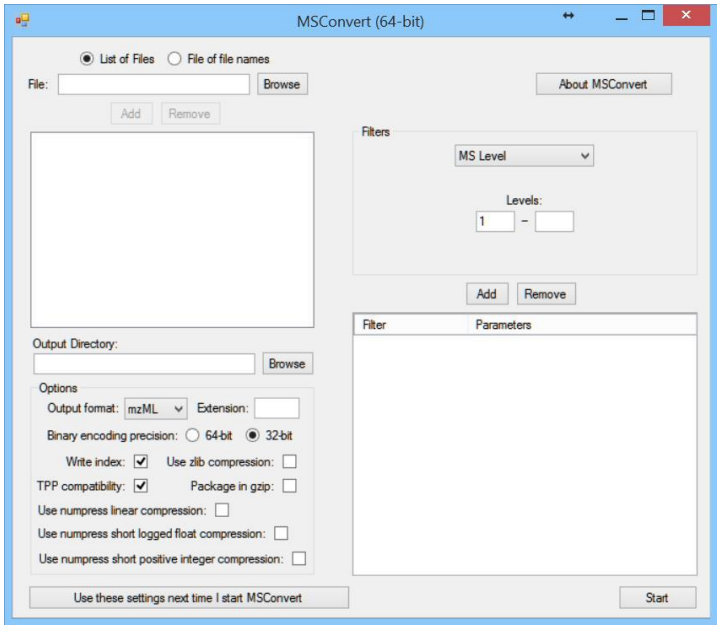
```
building index.
- remapping alphabet.
- creating cumulative counts C[].
- performing bwt.
- sample SA locations.
- creating bwt output.
- create RRR wavelet tree over bwt.
build FM-Index done. (0.021 sec)
space usage:
- remap_reverse: 21 bytes (0.10%)
- C: 1028 bytes (4.66%)
- Suffixes: 1484 bytes (6.72%)
- Positions: 1488 bytes (6.74%)
- Sampled: 1276 bytes (5.78%)
- T_bwt: 16512 bytes (74.78%)
input Size n = 23738 bytes

index Size = 22081 bytes (0.93 n)
writing FM Index to file 'protein.fm.1'
```

III. Gather input files

A. Generate peak lists from raw mass spectra data files

TagGraph natively interprets mzXML and mzML formats. Several tools are available for converting vendor-specific raw data formats to either of these standards. We have tested and recommend using msconvert, a component of the proteowizard suite of tools: http://proteowizard.sourceforge.net/downloads.shtml



1. Select raw data file(s) to be converted
2. Select local output directory
3. Select 32-bit, and check write index, TPP compatibility. Do not check Use zlib compression or Package in gzip
4. Click “Start”

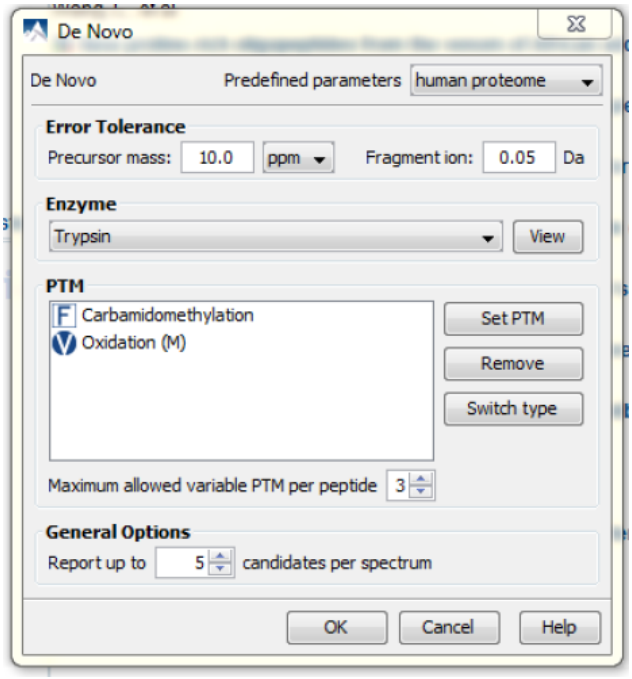
B. Generate de novo search results

TagGraph should in theory be able to consider any de novo sequence as input. We have found that PEAKS (ver 5 or later) produces strong results, in part due to its ability to consider a wide mass range and non-tryptic peptide options. TagGraph currently natively parse de novo results in the PEAKS .csv output file format (detailed below) or the pepXML standard. Converting output from other de novo sequencing engines to either of these formats should be compliant with TagGraph operation, provided that the following search parameters are approximately followed:

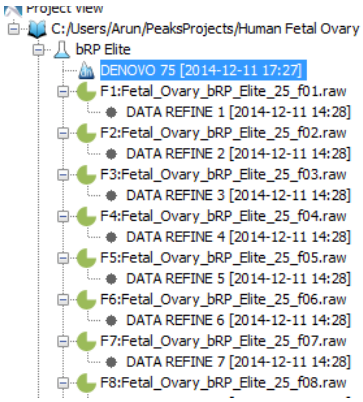
PEAKS de novo search parameters:

1. Set enzyme to enzyme used in your sample (only Trypsin and LysCsupported now)

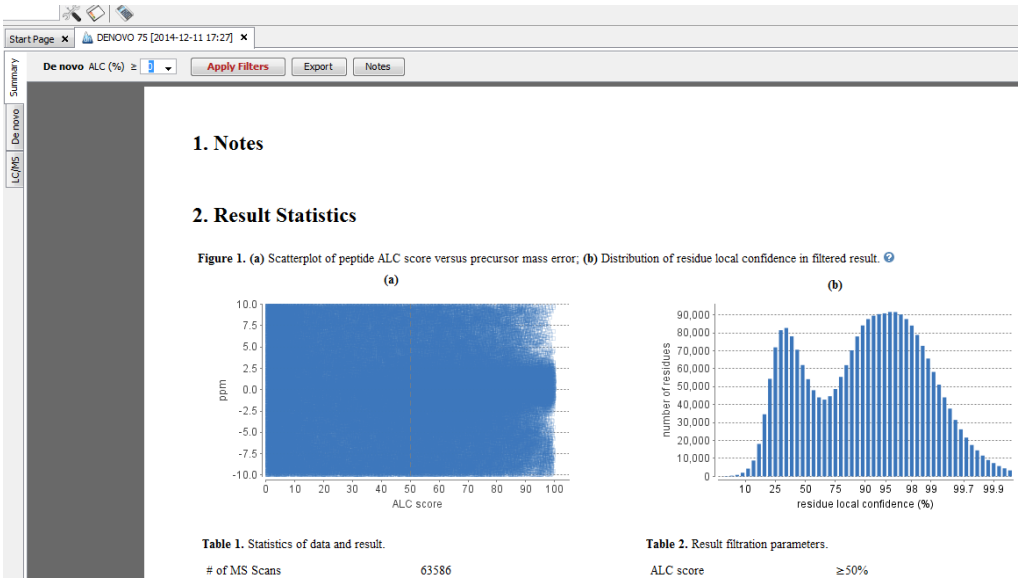
MODS MUST BE SET EXACTLY AS INDICATED!



Export Step 1:
Double click the DENOVO entry under the project to open the de novo results

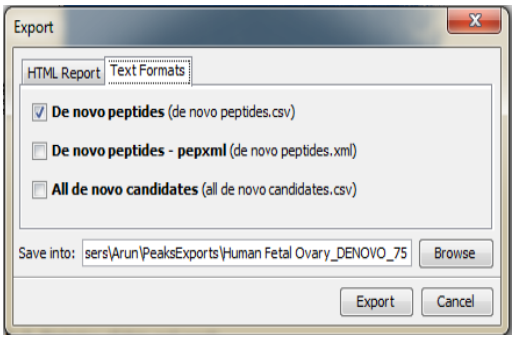


Export Step 2:
Change the ALC cutoff from 50 to 0 and then click the “Apply Filters” button (in red). This is important so that PEAKS exports all de novo results, not just the ones it thinks are high quality.



Export Step 3:

Click the “Export” button next to “Apply Filters”. Export using the settings shown here, upload the “de novo peptides.csv” file to the web server along with the corresponding mzXMLfile and watch the magic happen.



1. CSV file format

See deNovo/peaks/ within sampleInputFiles.1.8.tar.gz tarball for example.

2. pepXML file format

NB: If raw (as opposed to mzXML or mzML) data files are searched directly by de novo software, you will need to edit the de novo output file (.csv or .pepXML) so that the file extensions they list match the TagGraph input (mzXML or mzML).

e.g., for Thermo raw files:

sed ‘s/\.raw/.mzXML/g’ *original de novo file >altered de novo file*

TagGraph can also take the Deepnovo, NOVOR, or Kaiko’s result, with user define mapping from the [Denovo] section in the configuration file describe below, for the mapping column position.

A custom script created to apply the scanNum to the NOVOR result if the result file from the run showed “0” for scanNum; Please run the python script: novorAddScanNumber.py with the .mgf, and the NOVOR result file to get the scanNum filed populated before you run TagGraph.

IV. Set up TagGraph running parameters

A. TagGraph parameter file: (config/PEAKS.mzML.ini, or config/ deepNovo.ini in sampleInputFiles.1.8.tar.gz tarball)

```
##### General Settings #####
[General]
# Not used at this time, place-holder for the capabilities to only run the TG step, or the
EM step if TG was run previously
runTG = True
runEM = True

# If set to True the copies of the input mzML/mzXML input files are removed from the
output tree when the run has finished.
cleanInputDataFilesFromOutput = True
cleanIntermediateFiles = False

# output file
generatePepXML = False      # Boolean: print tab-delimited .txt file only, or pepXML
                            # format too?
outputPerFraction = False   # possibly useful for large data sets: produce per-fraction
                            # ouput in addition to aggregated output across all
                            # fractions?
FDRCutoff = 0.01            # FDR Threshold to apply to output
logEMCutoff = 2             # p-value threshold to apply to output (equivalent to p=0.99
                            # that identification is correct)
DisplayProteinNum = 3       # maximum number of proteins to list which contain
                            # identified peptide
##### END General Settings #####

##### De Novo Settings #####

[DeNovo]

# File mapping:

# If de novo sequencing program creates a single output text file, simply
enter that text file;

# Otherwise, if de novo sequencing program generates one text output per
input raw data file and there are multiple raw data input files, enter the
folder which contains all de novo files: e.g.:

# de_novo= /project_folder/de_novo_results/denovo_output.csv OR
de_novo=/project_folder/de_novo_results/

de_novo =A375_Dataset/

# If de novo sequencing program generates one text output per input file,
enter mapping here, linking each de novo file to the root name of the
corresponding input raw data file via a pipe (|) separating each de novo /
```

raw file pair by semicolons, e.g.: denovoMZML =216.csv|ath017216;217.csv|ath017217, otherwise, leave blank.

denovoMZML =216.csv|ath017216

De novo output file parsing

De novo output file parsing: Header lines to ignore

Enter number lines in de novo input file which contains descriptive information (e.g., column headers) to ignore. e.g., Ignore rows= 1

ignoreRows =1

#delimiter for the denovo file. It can be a tab, a comma, etc.

denovoFileDelimiter=

Column assignment

Column assignment: Multiple raw file / Fraction designation:

When searching multiple input raw files in one batch, does the de novo software assign a unique identifier to each input raw file (e.g., a fraction number)? If so, enter the column number which contains this information (left-most column = 1) e.g., fractionID= 1

fractionID =?

Column assignment: Scan designation

Enter column number which contains scan mapping information for given peptide (left-most column = 1) e.g., scan= 10

Scan =10

De novo output file parsing: Scan parsing:

#If de novo sequencing program aggregates multiple scans which contributed to a peptide identification (e.g., "2333;2567;2673"), enter the character which delimits each scan (i.e., "/"). e.g., scanSplitter= ;

scanSplitter =";"

Column assignment: Charge designation

Enter column number which contains charge assignment information for given peptide (left-most column = 1) e.g., charge= 7

Charge   =7

Column assignment: Peptide sequence

Enter column number which contains peptide sequence (left-most column = 1) e.g., peptide = 3

peptide =3

De novo output file parsing: Peptide parsing:

If de novo sequencing program enters a character between each returned amino acid (e.g., "P.E.P.T.I.D.E"), enter that character (i.e., "."). TagGraph will ignore this character in the peptide sequence: e.g., denovoDelimeter= .

peptideDelimiter=,

#optional parameters:

rt =?

precursor_mz =6

ALCinPCT =5

LC =11

End De Novo Settings

TagGraph Settings
[TagGraph]
# An integer number of data files to be used as fractions, and the path to each (mz[X]ML) data file with 2-digit counter appended to 'fraction' as key
dataDirectory = A375_Dataset/

#give full path to target data set directory
# Name to be used for this experiment and its output files and directories. Must not contain spaces.
ExperimentName = A375

# Location of input de novo search results exported from de novo sequencing program in .csv or .pepXML format.
de_novo = A375_Dataset/de_novo_peptides_mzXML.csv

# Path to folder TagGraph will create to store output. Should not already exist.
Output = A375_Dataset/EM_output

# Initialization file used to configure TAG-GRAPH. See below for detailed information.
init = resources/TAG_GRAPH_Tryp_CysCarbam_MetOx.ini

# Location and name of fmindex to search. This fmindex location and name should describe the root file name and extension prefix (lacking any numbers).
Fmindex = resources/FMIndices/20141209_UniHUMAN_cRAP_ILEq.fm

# Expected standard deviation in ppm error distributions of fragment ions. Recommend 5 for HCD 30,000 resolution
Ppmstd = 10

Maximum absolute deviation (Da) between experimental and database modification mass for TagGraph to consider modification as a candidate match. Recommend 0.1.

```
Modtolerance = 0.1

# Maximum number of times a de novo-produced substring can occur in the protein
sequence database for TagGraph to consider it as an unmodified peptide match.
# Recommend less than 1,000 for single organism database (i.e., human uniprot) and
5,000 for nr or 6-frame translations
Maxcounts = 400

# Maximum number of times a de novo-produced substring can occur in the protein
sequence database for TagGraph to consider it as a modified peptide match.
# Recommend 200 for single organism database (i.e. human uniprot) and 1,000 for nr or
6-frame translations
modmaxcounts = 200

# Location of pickled (python-serialized) unimod dictionary
unimoddict = resources/unimodDict_noLabels_20160724.pck

# Path to pickled (python-serialized) probabilistic model file.
model =
resources/AllChargeDist_posOnlyDependence_20150808_HumanProt500000.pck

# Path to pickled (python-serialized) model configuration file.
Config = resources/AllChargeDist_posOnlyDependence_20150808.txt
##### End TagGraph Settings #####

##### EM Settings #####
[EM]
# Number of iterations in initial EM overall results. Recommend 20 iterations.
initIterations = 20

# Maximum number of expectation maximization iterations for FDR assignment.
Recommend 100 iterations.
maxIterations = 100

# Filename Prefix to use for the output EM results files. Must not contain spaces.
resultsPrefix = EM_Results
##### End EM Settings #####
```

B. TagGraph initialization file (see init\Tag_Graph_Tryp_MetOx.ini in sampleInputFiles.1.8.tar.gz tarball)

```
; Add in default values for command line parameters here

[Parameters]




; Add enzyme info here

; Specificity is regular expression with a semicolon used to indicate cleavage site
```

[Enzyme]

Name: Tryp

Specificity: K|R;.*

; AA_name: AA_1_letter_abbrev AA_3_letter_abbrev AA_Elemental_Comp AA_monoisotopic_mass AA_avg_mass

; Add any additional amino acids other than the twenty original ones here

[Amino Acids]

Selenocysteine: U Sec C3H7NO2Se 150.953636 151.0388

Pyrrolysine: O Pyr C12H21N3O3 237.147727 237.3018

Xeucine: J Xle C6H11ON 113.08406 113.1594

; mod_name: AA mod_mass

; use N-Term for N-terminus and C-Term for C-terminus

[Static Mods]

; mod_name: AA(can be list of AAs such as STY, etc.) mod_mass overide_static_mod mod_symbol

; mod_symbol optional and will be chosen automatically if not given

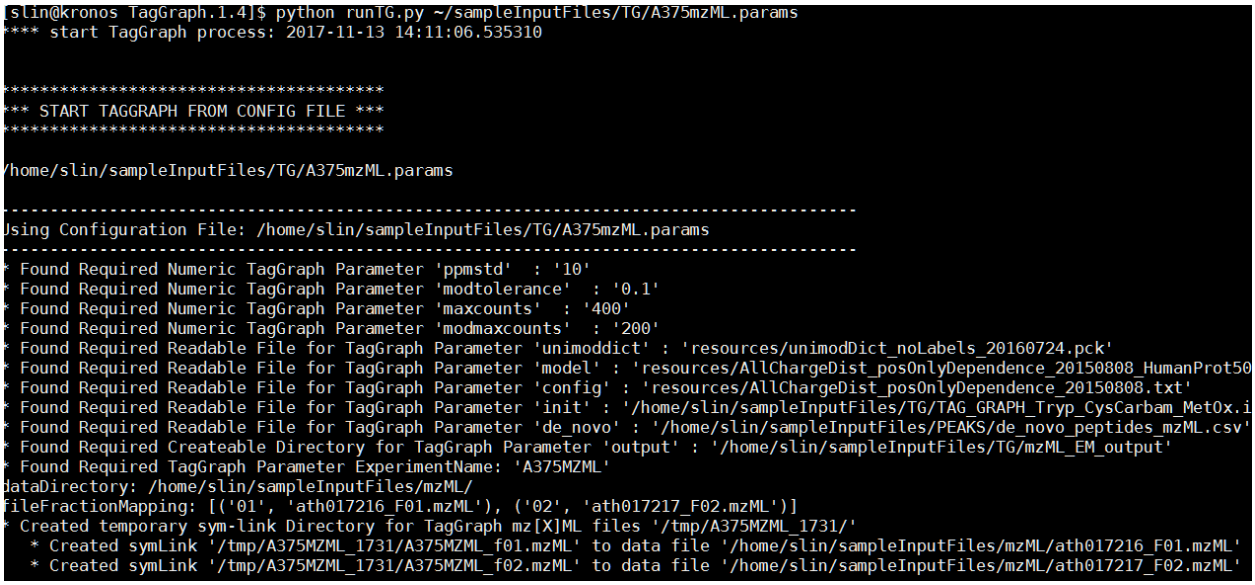
; override_static_mod is either 0 or 1, 1 means add mod_mass to original AA mass, not statically modified mass

[Diff Mods]

Oxidation: M 15.994915 0 #

V. Run TagGraph

Command to run the taggraph:
cd TagGraph.1.8
python runTG.py pathToYourParameterFile



VI. Output description

Output Overview

Taggraph writes the output to the output directory specified within the parameter file.

Description of files and folders in the output

Results files:
Taggraph outputs a run log: runReport.log. The log lists if the taggraph run was successful using three parameters, reported towards the end of this file:

1)Spectrum score x^2 > 1? Yes: PASS. No: FAIL - Match between high-scoring peptides and spectra is unusually low
2)Tag length x^2 > 1? Yes: PASS. No: FAIL - Unexpectedly few long de novo alignments with database
3)Mod size x^2 < 1? Yes: PASS. No: FAIL - TagGraph had to insert far more large modifications than would be expected.

Results are written out to three files listing the peptides assigned to each scan.
results.db – raw output file with no filtering. Open only with appropriate viewer for .db files
<SAMPLE>_TopResults.tdv – unformatted output of top hits after EM-score based filtering
<SAMPLE>_TopResults.txt – Formatted version of the .tdv output, listing out every modification in separate columns. Output fields in this file described below:

Output Fields
File: Sample

ScanF	Charge	Retention Time	Obs M+H	Theo M+H	PPM	PPM Upper Bound	PPM Lower Bound	PPM In Range
FL0016200.mzXML:76771	2	139.03	1035.58	1035.59	-2.8849	1.8	-4.6	Yes
FL0016200.mzXML:84366	2	148.42	1496.82	1496.82	-1.2811	1.8	-4.6	Yes
FL0016200.mzXML:80488	2	144.12	1808	1808.01	-1.846	1.8	-4.6	Yes
FL0016200.mzXML:71335	2	131.81	1172.53	1172.53	-2.9744	1.8	-4.6	Yes
FL0016200.mzXML:32843	2	77.44	1022.58	1022.58	-2.5695	1.8	-4.6	Yes
FL0016200.mzXML:18835	2	51.88	813.461	813.462	-1.4845	1.8	-4.6	Yes
FL0016200.mzXML:31552	2	75.22	1492.66	1492.66	-0.5196	1.8	-4.6	Yes
FL0016200.mzXML:29790	2	72.19	1521.74	1521.74	-0.0404	1.8	-4.6	Yes
FL0016200.mzXML:80159	4	143.72	2611.37	2611.37	-2.1408	1.8	-4.6	Yes

#ScanF
Number of peptide's MS2 fragmentation scan listed as: Filename.format:Scan#

#Charge
Charge state of precursor ion that gave rise to the peptide.

#Retention time
Peptide retention time (mins)

#Obs M+H

Inferred singly-charged ion mass, based on observed precursor ion's m/z ratio and charge

#Theo M+H
Computed peptide's singly-charged ion mass, based on its amino acid sequence and any additional modifications

#PPM
Parts-per-million mass deviation between observed and theoretical peptide masses

#PPM Upper
Upper bound of precursor mass tolerance based on all files submitted together in TagGraph instance

#PPM Lower
Upper bound of precursor mass tolerance based on all files submitted together in TagGraph instance

#PPM In Range
Specifies if calculated precursor error within specified tolerance [Yes/No]

FDR	EM Probability	1-lg10 EM	Spectrum Score	Alignment Score	Composite Score	Unique Siblings	Context Mod Variants	Num Mod Occurrences
0	0.9999997	6.51735	9.351939	9	18.351939	350	1	6116
0	0.99999999	8.26582	8.196586	12	20.196586	62	1	6116
0	1	49.0427	8.500118	16	24.500118	66	9	6116
0	0.99999999	7.92573	7.884147	10	17.884147	350	2	6116
0	0.99999977	6.63157	8.833676	9	17.833676	855	1	6116
0	1	9.0336	9.152572	7	16.152572	855	4	6116
0	1	12.0068	8.270654	13	21.270654	125	5	6116
0	1	11.3179	9.382939	13	22.382939	120	2	6116
0	1	51.7194	6.919787	23	29.919787	855	7	6116

#FDR
PSM false discovery rate

#EM Probability
Expectation Maximization (EM)-estimated probability of indicated peptide identification being correct [0-1]
1 = high probability
0 = low probability

#1-lg10 EM
Log-transformed, inverse probability of indicated peptide identification being correct, as computed by the expectation maximization-optimized Bayesian network; calculated as log10(1-EM probability)

#Spectrum Score
Score describing the extent to which a peptide agrees with the input spectrum

#Alignment Score
Heuristic score describing the extent to which the de novo sequence agrees with the database context sequence

#Composite Score
Composite score combining the spectrum score and the alignment score

#Unique Siblings
Number of other unique peptides found from the same protein that produced the current peptide-spectrum match

#Context Mod Variants
The number of unique peptides present in the dataset with the same base peptide sequence

#Num Mod Occurrences
The number of times the indicated modification(s) occur in the data set

Context	Mod Context	Mods	Mod Ambig Edges	Mod Ranges	Proteins	De Novo Peptide	De Novo Score	Matching Tag Length	Num Matches	N-terminus	Final Peptide	C-terminus	mod Pos0	mod Name0	modAA0	mod Mass0
APEIIFFAK	APEIIFFAK	[]	[]	[]	1:: ['sp|P02768|	APEIIFFAK	99	9	1	Y	APEIIFFAK	R				
TFEIVYEEII	TFEIVYEEII	[]	[]	[]	1:: ['sp|Q14624|	TFEIVYEEII	99	12	1	K	TFEIVYEEII	R				
ITVIHQDW	ITVIHQDW	[]	[]	[]	3:: ['sp|P01861|	ITVIHQDW	99	16	3	R	ITVIHQDW	E				
IDDFAAFVE	IDDFAAFVE	[]	[]	[]	1:: ['sp|P02768|	IDDFAAFVE	99	10	1	V	IDDFAAFVE	C				
IIAHWSPAI	IIAHWSPAI	[]	[]	[]	1:: ['sp|P04114|	IAHWSPAI	99	9	1	E	IAHWSPAI	I				
‡.TPAIHFK.‡	‡.TPAIHFK.‡	[]	[]	[]	1:: ['sp|P04114|	TPAIHFK	99	7	1	R	TPAIHFK	S				
SCGDICNFI	SCGDICNFI	[]	[]	[]	1:: ['sp|P04003|	SCGDICNFF	99	13	1	R	2]GDIC[57	I	3	ethyla	C	57
DEWSVNS	DEWSVNS	[]	[]	[]	1:: ['sp|P02787|	DEWSVNSV	99	13	1	R	02]DEWSV	I	2	ethyla	C	57
IANIINSEEI	IANIINSEEI	[]	[]	[]	1:: ['sp|P04114|	IANIINSEEI	99	23	1	K	IANIINSEEI	I				

#Context
Peptide sequence from input FASTA sequence database, with flanking amino acids (modifications not specified)

#Mod Context
Peptide sequence from input FASTA sequence database with modified residue designated with "-".

#Mods
Modifications assigned to TagGraph-resolved peptide: Nested series are of the format:

> *"[(('Mod1 name from Unimod if exists', Mod1 delta mass from Unimod if it exists, Mod1 delta mass vs. Unimod if exists), (Mod1 target amino acid from Unimod if exists, Mod1 target amino acid location on peptide from Unimod if exists), indexed location of Mod1 on peptide sequence counting from zero), (('Mod2 name from Unimod if exists'…, indexed location of Mod2 on peptide sequence counting from zero),(...)].* "

Isobaric substitution" describes mass-neutral differences between the de novo-identified peptide and its database counterpart.

#Mod Ambig Edges

Indication of mass gaps TagGraph needed to reconcile between de novo and database sequences

#Mod Ranges
Indication of the possible modification positions TagGraph needed to consider when choosing the final modification location

#Proteins
List of proteins from FASTA sequence database containing indicated peptide

#De Novo Peptide
Peptide predicted by de novo sequencing algorithm (PEAKS) – derived from the de novo sequencing output .pepXML file(s)

#De Novo Score
Peptide de novo score (ALC) [0-100] by de novo sequencing algorithm (PEAKS). Higher score = higher confidence in de novo prediction

#Matching Tag Length
Length of the longest contiguous string in common between fasta and de novo sequence candidates

#Num Matches
Number of peptide candidates considered from fasta file considering spectrum and de novo peptide with indicated Matching Tag Length.

#N-terminus
Flanking residue on the peptide's N-terminus

#Final Peptide
Peptide assigned to spectrum by Taggraph, incorporating all amino acid substitution, and decimal representation of modification between brackets

#C-terminus
Flanking residue on the peptide's C-terminus

#modPosX
Modification position [X] on peptide

#modNameX
Name of modification at position [X] from Unimod if available

#modAAX
Modified amino acid at position [X]

#modMassX
Modification mass shift at position [X]

Taggraph currently outputs files and folders that summarize models used, modifications listed by rank of occurrences, and per scan scores as reported by TAGGRAPH before and after EM-ranking. These are for auditing purposes only and will be removed from the output in future versions. These are listed below:

Folder <SAMPLE> **containing TG parsed files**
<SAMPLE>_CHECK.txt.12
fileFractionMapping.pck
TagGraph.pep.xml
File containing user defined TG and EM parameters at input
TG_params_<SAMPLE>.txt
Files listing mods (single and combination, indicating delta mass, position, AA)
<SAMPLE>_addPlausibleMods_poss_combo_mods.tdv
<SAMPLE>_addPlausibleMods_poss_single_mods.tdv
EM results – Models End has a ranked list of mods
EM_Results_EMProbs_BEFORERERANK.tdv
EM_Results_EMProbs_END_TOPONLY.tdv
EM_Results_MODELS_BEFORERERANK.log
EM_Results_MODELS_END.log
EM_Results_RERANKSTATS_ONE.tdv
EM_Results_RERANKSTATS_TWO.tdv

VII. Known issues

Peptides that are found with long matching strings spanning hundreds or thousands of proteins (e.g., MHC genes) are not reported