specL - Prepare Peptide Spectrum Matches for Use in Targeted Proteomics

Christian Panse*
Christian Trachsel†
Jonas Grossmann‡

October 13, 2014

*cp@fgcz.ethz.ch
†christian.trachsel@fgcz.ethz.ch
‡jg@fgcz.ethz.ch

Contents

1 Introduction **3**

2 Workflow **3**
2.1 Prologue - How to get the input for the specL package? 3
2.2 Input: Redundant plus non-redundant blib files 3
2.3 Protein ID annotation using FASTA 5
2.4 Generate the spectrum library (assay) 6
2.5 Normalizing the retention time using iRT peptides 8
2.6 Output: R console or csv file 10
2.7 Epilogue - What can I do with that library now? 10

3 Acknowledgement **10**

4 To Do for next releases **10**

5 Session information **10**

1 Introduction

Targeted proteomics is a fast evolving field in proteomics science and was even elected as method of the year in 2012 [1]. Especially targeted methods like SWATH [1] open promising perspectives for the identification and quantification of peptides and proteins. All targeted methods have in common the need of precise MS coordinates composed of precursor mass, fragment masses, and retention time. The combination of this information is kept in so called assays or spectra libraries. Here we present an R package able to produce such libraries out of peptide identification results (Mascot (dat), TPP (pep.xml and mzXMLs), ProteinPilot (group), Omssa (omx)). . *specL* is an easy to use, versatile and flexible function, which can be integrated into already existing commercial or non commercial analysis pipelines for targeted proteomics data analysis. Some examples of today's pipelines are ProteinPilot combined with Peakview (ABSciex), Spectronaut (Biognosys) or OpenSwath [2].

In the following vignette it is described how the *specL* package can be used for the included data sets `peptideStd` and `peptideStd.redundant`.

2 Workflow

2.1 Prologue - How to get the input for the specL package?

Since peptide identification (using, e.g., Mascot, Sequest, xTandem!, Omssa, ProteinPilot) usually creates result files which are heavily redundant and therefore unsuited for spectrum library building, the search results must first be filtered. To create non-redundant input files, we use the BiblioSpec [3] algorithm implemented in Skyline [4]. A given search result (e.g. Mascot.DAT file) is loaded into the software Skyline and is redundancy filtered. The 'Skyline workflow step' provides two sqlite readable files as output named `'*.blib'` and `'*.redundant.blib'`. These files are ideally used as ideal input for this packages. Note here, that Skyline is very flexible when it comes to peptide identification results. It means with Skyline you can build the spectrum library files for almost all search engines (even from other spectrum library files such as spectraST [5]).

The first step which has to be performed on the R shell is loading *specL* library.

```
> library(specL)
> data(peptideStd)
```

2.2 Input: Redundant plus non-redundant blib files

for demonstration *specL* contains the two data sets namely `peptideStd` and `peptideStd.redundant`. This is a data set which comes from two standard run experiment which are routinely used to check the liquid chromatographic system is still working appropriate. The sample consists of a digest of the

[1] http://www.nature.com/nmeth/journal/v10/n1/pdf/nmeth.2329.pdf, 2014-09-22

Fetuin protein (Bos taurus, uniprot id: P12763). 40 femtomole are loaded on column. Mascot was used to search and identify the respective peptides.

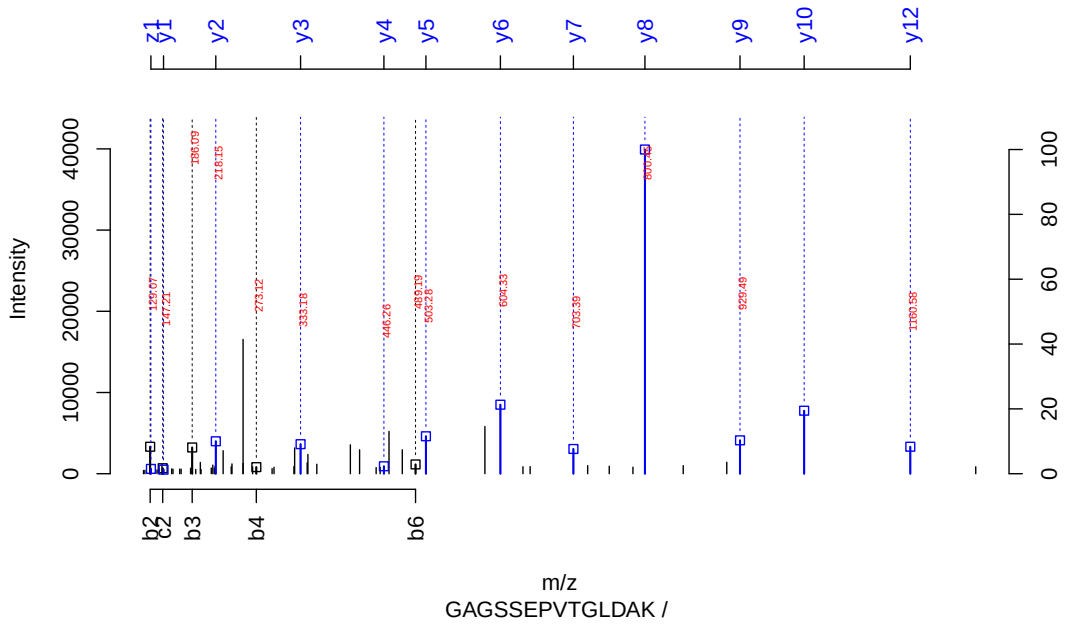
```
> demoIdx <- 40
> str(peptideStd[[demoIdx]])
```

```
List of 11
 $ peaks             : int 63
 $ mZ                : num [1:63] 120 122 129 129 130 ...
 $ intensity         : num [1:63] 401 420 3321 901 592 ...
 $ peptideSequence   : chr "GAGSSEPVTGLDAK"
 $ charge            : int 2
 $ pepmass           : num 645
 $ fileName          : chr "s:\\p1239\\Proteomics\\QEXACTIVE_3\\ctrachse_20140910_Nuclei_di
 $ proteinInformation: chr ""
 $ rtinseconds       : num 9.13
 $ varModification   : num [1:14] 0 0 0 0 0 0 0 0 0 0 ...
 $ mascotScore       : num 41.5
```

For both `peptideStd`, `peptideStd.redandant` data sets the Skyline software was used to generate the bibliospec files which contain the peptide sequences with the respective peptide spectrum match (PSM). The read.bibliospec function was used to read the blib files into R.

The peptide spectrum match can be displayed using the *protViz* `peakplot` function.

```
> res.peakplot <- peakplot(peptideSequence=peptideStd[[demoIdx]]$peptideSequence,
+                 spec=peptideStd[[demoIdx]],
+                 ion.axes=TRUE)
>
```



2.3 Protein ID annotation using FASTA

The information to which protein a peptide-spectrum-match belongs (PSM) is not stored by BiblioSpec. Therefore specL provides the `annotateProteinID` function which uses R's internal `grep` to 'reassigning' the protein information. Therefore a `fasta` object has to be loaded into the R system using `read.fasta` of the seqinr package. For this, not necessarily, the same `fasta` file needs to be provided as in the original database search.

The following lines demonstrate a simple sanity check with a single FASTA style formated protein entry. Also it demonstrates the use case how to identify entries in the R-object which are from one or a few proteins of interest.

```
>  irtFASTAseq <- paste(">zz|ZZ_FGCZCont0260|",
+ "iRT_Protein_with_AAAAK_spacers concatenated Biognosys\n",
+ "LGGNEQVTRAAAAKGAGSSEPVTGLDAKAAAAKVEATFGVDESNAKAAAAKYILAGVENS",
+ "KAAAAKTPVISGGPYEYRAAAAKTPVITGAPYEYRAAAAKDGLDAASYYAPVRAAAAKAD",
+ "VTPADFSEWSKAAAAKGTFIIDPGGVIRAAAAKGTFIIDPAAVIRAAAAKLFLQFGAQGS",
+ "PFLK\n")
> Tfile <- file();  cat(irtFASTAseq, file = Tfile);
> fasta.irtFASTAseq <-read.fasta(Tfile, as.string=TRUE, seqtype="AA")
> close(Tfile)
```

As expected the `peptideStd` data, e.g., our demo Object, does not contain any protein information yet.

```
> peptideStd[[demoIdx]]$proteinInformation
```

```
[1] ""
```

The protein information can be added as follow:

```
> peptideStd <- annotateProteinID(peptideStd,
+     fasta=fasta.irtFASTAseq)
```

The following lines show now the object indices of those entries which do hav a protein information now.

```
> (idx<-which(unlist(lapply(peptideStd,
+     function(x){nchar(x$proteinInformation)>0}))))
```

```
[1]  23  40  52 109 110 118
```

As expected, there are now a number of peptide sequences annotated with the protein ID.

```
> peptideStd[[demoIdx]]$proteinInformation
```

```
[1] "zz|ZZ_FGCZCont0260|"
```

Please note, that the default digest pattern is defined as

```
> digestPattern = "(([RK])|(^)|(^M))"
```

for tryptic peptides. For other enzymes, the pattern has to be adapted. For example, for semi-tryptic identifications use `digestPattern = ""`.

2.4 Generate the spectrum library (assay)

`genSwathIonLib` is the main contribution of the *specL* package. It generates the spectra library used in a targeted data extraction workflow from a mass spectrometric measurement.

Generation of the spec Library with default settings.

```
> res.genSwathIonLib <- genSwathIonLib(data=peptideStd,
+                                      data.fit=peptideStd.redundant)
```

The determined mass spec coordinates of the selected tandem mass spectrum `demoIdx` look like this:

```
> res.genSwathIonLib@ionlibrary[[demoIdx]]
```

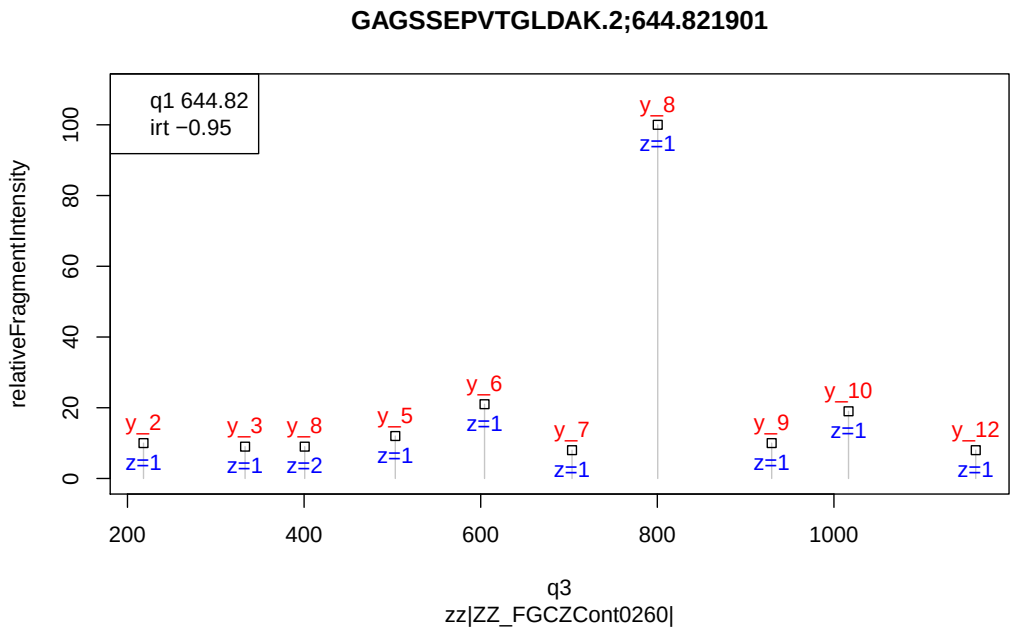
```
An "specL" object.


content:
group_id = GAGSSEPVTGLDAK.2;644.821901
peptide_sequence = GAGSSEPVTGLDAK
proteinInformation = zz|ZZ_FGCZCont0260|
q1 = 644.8219
q3 = 800.4497 604.3285 1016.522 503.2805 929.4925 218.1497 400.7282 333.176
1160.581 703.3948
q3.in_silico = 800.4512 604.3301 1016.526 503.2824 929.4938 218.1499 400.7295
333.1769 1160.579 703.3985
decoy = NA NA NA NA NA NA NA NA NA NA
prec_z = 2
frg_type = y y y y y y y y y y
frg_nr = 8 6 10 5 9 2 8 3 12 7
frg_z = 1 1 1 1 1 1 2 1 1 1
relativeFragmentIntensity = 100 21 19 12 10 10 9 9 8 8
irt = -0.95
peptideModSeq =
mZ.error = 0.001514 0.00156 0.003685 0.001914 0.001318 0.000207 0.001313
0.000856 0.001846 0.003686
filename =
s:\p1239\Proteomics\QEXACTIVE_3\ctrachse_20140910_Nuclei_diff_extraction_methods\20140910_


size:
Memory usage: 4400 bytes
```

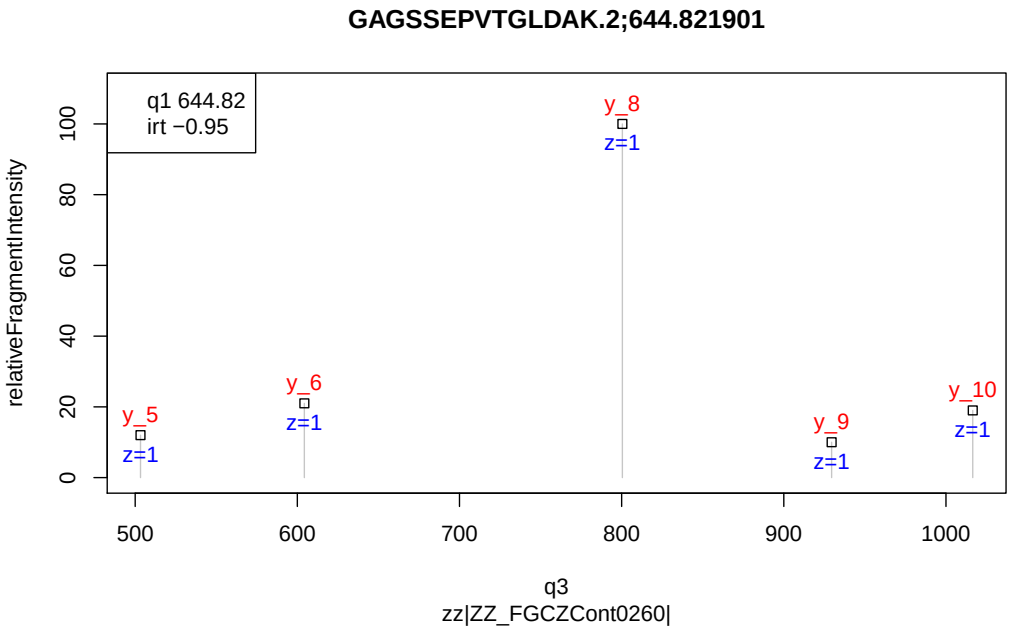
It can be displayed using the `specL::plot` function.

```
> plot(res.genSwathIonLib@ionlibrary[[demoIdx]])
```



The following code considers only the top five y ions.

```
> # define customized fragment ions
> # for demonstration lets consider only the top five singly charged y ions.
>
> r.genSwathIonLib.top5 <- genSwathIonLib(peptideStd,
+     peptideStd.redundant, topN=5,
+     fragmentIonFUN=function (b, y) {
+       return( cbind(y1_=y) )
+       }
+     )
> plot(r.genSwathIonLib.top5@ionlibrary[[demoIdx]])
>
```



2.5 Normalizing the retention time using iRT peptides

Retention time is a very important parameter in targeted data extraction. However retention times are not easy to transfer between different reverse phase columns or HPLC systems. To make transfer applicable and account for inter run shift in retention time Biognosys [6] invented the iRT normalization based on iRT / HRM peptides. For this, a set of well behaving peptides (good flying properties, good fragmention characteristics, completely artifical) which cover the whole rt-gradient are spiked into each sample. For this set of peptide an idependent retention time (dimension less) is suggested by Biognosys. With this at hand, the set of peptides can later be used to apply a linear regression model to adapt all measured retention times into an independent retention time scale.
If the identification results contain iRT peptides the package supports the conversion to the iRT scale. For this (if the identification result are based on multiple input files) the redundant BiblioSpec file is required where all iRT peptides from all measurements are stored. For the most representative spectrum in the non-redundant R-object the original filename is identified and the respective linear model for this one particular MS experiment is applied to normalize the retention time to the iRT scale. The iRT peptides as well as their independent retention times are stored in the `iRTpeptides` object.

specL uses by default the iRT peptide table to normalize into the independent retention time but could also be extended or changed to custom iRT peptides if available.

```
> iRTpeptides
```

The method genSwathIonLib uses:

```
> fit <- lm(formula = rt ~ aggregateInputRT * fileName, data=m)
```

to build the linear models for each MS measurement individually. For defining `m` both data sets were aggregated over the attributes `peptide` and `fileName` using the `mean` operator.

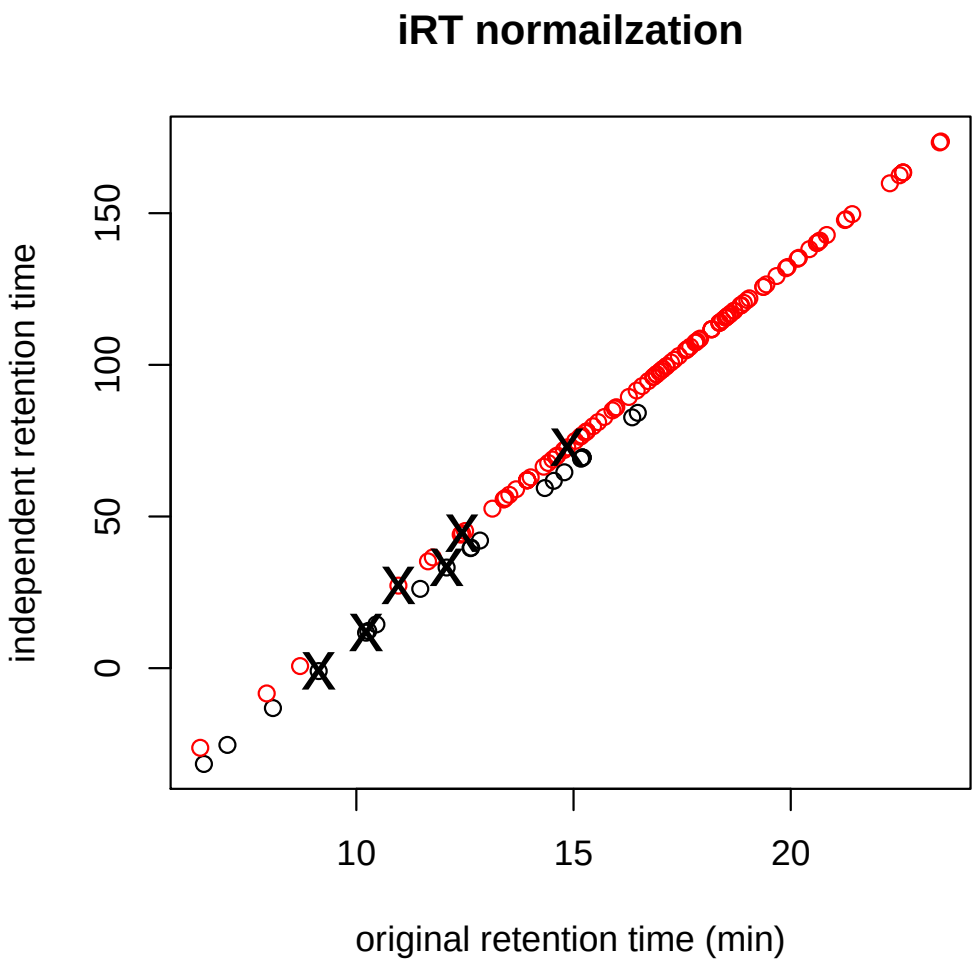
```
> data<-aggregate(df$rt, by=list(df$peptide, df$fileName), FUN=mean)
> data.fit<-aggregate(df.fit$rt, by=list(df.fit$peptide, df.fit$fileName), FUN=mean)
```

Afterwards the following join operator was applied.

```
> m <- merge(iRT, data.fit, by.x='peptide', by.y='peptide')
```

The following graph displays the normalized retention time versus the measured retention time after applying the calculated model to the two data sets.

```
> # calls the plot method for a specLSet object
> plot(res.genSwathIonLib)
> # graph the iRT peptides of "zz|ZZ_FGCZCont0260|"
> points(res.genSwathIonLib@rt.normalized[idx] ~ res.genSwathIonLib@rt.input[idx],
+        col='black', lwd=4, pch="x", cex=2)
```



Shown are original retention time (in minutes) and iRT (dimensionless) for two standard run experiments (color black and red). Indicated with black **X** are the iRT peptides which are the base for the regression.

2.6 Output: R console or csv file

The output can be written as an ASCII text file.

```
> write.Spectronaut(res.genSwathIonLib, file="specL-Spectronaut.txt")
```

2.7 Epilogue - What can I do with that library now?

The specL output text file can directly be used as input (assay) for the Spectronaut software from Biognosys or with minimal reshaping for Peakview. Alternatively it can be used as a basis for script based construction of SRM/MRM assays.

3 Acknowledgement

The authors thank all colleagues of the Functional Genomics Center Zuerich (FGCZ), and especial thank goes to our test users Sira Echevarría Zomeño (Swiss Federal Institute of Technology in Zurich (ETHZ)), Tobias Kockmann (Swiss Federal Institute of Technology in Zurich (ETHZ)) and Stephan Michalik (Ernst-Moritz-Arndt-Universität Greifswald, Germany).

4 To Do for next releases

- new option for `specL::genSwathIonLib`; Exclude fragment ions from precursor `window = TRUE, FALSE`
- new option for `specL::genSwathIonLib`; Calculate `q1.in-silico` as an alternative to `q1`
- new option for `specL::genSwathIonLib`; Predict transitions for heavy labeled peptides using information from light peptides `predictHeavy = TRUE,FALSE, LabelFile = "fileWithHeavyAA"`
- new export function into TraML format for compatibility with OpenSWATH [2]
- streamline modsequence, e.g. `AAAMASATTM`$[16.0]$`LTTK` for compatibility with peakView V2.0
- replace *seqinr* `read.fasta` by using *Biostrings* `readAAStringSet` to handle fasta files

5 Session information

An overview of the package versions used to produce this document are shown below.

- R version 3.1.1 Patched (2014-09-25 r66681), `x86_64-unknown-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`

- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: BiocParallel 1.0.0, DBI 0.3.1, RSQLite 0.11.4, protViz 0.2.9, seqinr 3.0-7, specL 1.0.0
- Loaded via a namespace (and not attached): BBmisc 1.7, BatchJobs 1.4, BiocGenerics 0.12.0, BiocStyle 1.4.0, Rcpp 0.11.3, base64enc 0.1-2, brew 1.0-6, checkmate 1.4, codetools 0.2-9, digest 0.6.4, fail 1.2, foreach 1.4.2, iterators 1.0.7, parallel 3.1.1, sendmailR 1.2-1, stringr 0.6.2, tools 3.1.1

References

[1] L. C. Gillet, P. Navarro, S. Tate, H. Rost, N. Selevsek, L. Reiter, R. Bonner, and R. Aebersold. Targeted data extraction of the MS/MS spectra generated by data-independent acquisition: a new concept for consistent and accurate proteome analysis. *Mol. Cell Proteomics*, 11(6):O111.016717, Jun 2012.

[2] H. L. Rost, G. Rosenberger, P. Navarro, L. Gillet, S. M. Miladinovi?, O. T. Schubert, W. Wolski, B. C. Collins, J. Malmstrom, L. Malmstrom, and R. Aebersold. OpenSWATH enables automated, targeted analysis of data-independent acquisition MS data. *Nat. Biotechnol.*, 32(3):219–223, Mar 2014.

[3] B. Frewen and M. J. MacCoss. Using BiblioSpec for creating and searching tandem MS peptide libraries. *Curr Protoc Bioinformatics*, Chapter 13:Unit 13.7, Dec 2007.

[4] B. MacLean, D. M. Tomazela, N. Shulman, M. Chambers, G. L. Finney, B. Frewen, R. Kern, D. L. Tabb, D. C. Liebler, and M. J. MacCoss. Skyline: an open source document editor for creating and analyzing targeted proteomics experiments. *Bioinformatics*, 26(7):966–968, Apr 2010.

[5] H. Lam, E. W. Deutsch, J. S. Eddes, J. K. Eng, S. E. Stein, and R. Aebersold. Building consensus spectral libraries for peptide identification in proteomics. *Nat. Methods*, 5(10):873–875, Oct 2008.

[6] C. Escher, L. Reiter, B. MacLean, R. Ossola, F. Herzog, J. Chilton, M. J. MacCoss, and O. Rinner. Using iRT, a normalized retention time for more targeted measurement of peptides. *Proteomics*, 12(8):1111–1121, Apr 2012.