Contents

1 Quick Start

This shows section shows you how to quickly get started with gladiator-nf.

1.1 Tutorial

1.1.1 Cloning the directory

If you are using the development version of this, you will wnat to clone this directory, and 'tangle' (create the scripts). This will require 'make', emacs 27 or greater, and the coreutils

```
git clone https://github.com/elolab/gladiator-nf
cd gladiator-nf
yes '//' | make tangle
```

1.1.2 Example Data

For this example we will use MSV000090837 from , just 210820_Grad090_LFQ_A_01.raw and 210820_Grad090_LFQ_B_01.raw, so that you dont need to use that much disk space for trying this out. While the massive repo provides mzml files, which you could use, we are gonna start from the raw for educational purposes. Lets say Download the data

```
mkdir RAW
wget --no-directories  --directory-prefix=RAW ftp://massive-ftp.ucsd.edu/v05/MSV0000908

mkdir fasta
wget --directory-prefix=fasta  'ftp://massive-ftp.ucsd.edu:/v05/MSV000090837/sequence/i
```

1.1.3 Convert RAW to mzml

We will want to convert the raw files to mzml, so we will want to run the following command in the container

```
COMMAND='find . -iname '"'"'*.raw'"'"' -print0 | xargs -P5 -0 -i wine msconvert {} --fi
IMAGEID=$(podman pull docker://chambm/pwiz-skyline-i-agree-to-the-vendor-licenses:3.0.2
echo  $COMMAND > ./convert.sh
chmod +x convert.sh
podman run -it  -v $PWD:$PWD -w $PWD $IMAGEID /bin/bash  ./convert.sh
```

1.1.4 Setting up your experiments config file

Create a config file with the following contents, adjusting as needed. Lets name it `myconfig.nf`

```
params {
    // this is fragment mass tolerance in Dalton, 0.02 is a sensible default
    fragment_mass_tolerance=0.02
    // This is in parts per million, ppm
    precursor_mass_tolerance=10
    // this is cutoffrate used by mayu for finding the peptide probability
    protFDR=0.01
    // retention time information, this traml is a good default.
    irt_traml_file='ftp://PASS00289:XY4524h@ftp.peptideatlas.org:/SGS/assays/OpenSWATH_
    use_irt=false
    max_missed_cleavages=1
    libgen_method='diaumpire'
    pyprophet_use_legacy=false
    pyprophet_fixed_seed=false
    // this would make your subsample ratio 1 / your number of samples
    pyprophet_subsample_ratio=null
}
```

1.1.5 Calling gladiator

Note that you will have to quote wildcards, because nextflow's java expects to handle the wildcard expansion,rather than your shell handling the expandsion

```
# if you are using podman
NXF_VER=22.10.1 nextflow -c myconfig.nf \
         -c config/podman.nf \
         run gladiator.nf \
         --fastafiles='fasta/*.fasta' \
         --diafiles='MZML/*.mzML'
```

if you are using singularity, use `config/singularity.nf` in lieu of `config/podman.nf`. if you have the guix package manager, you can use `config/guix.nf` .

You will want to have java / openjdk 17, 18, or 19. Java 20 and later will give you a groovy error.

1.2 Results

In the results directory , which you can specify with `--outdir=/path/to/results`, which defaults to `./results`, you will find two files

- `dia/DIA-peptide-matrix.tsv` This contains the intensitities on the peptide level, the first column is the peptide identifier, and the the other columns are the intensities in that sample. For example:

"ProteinName_FullPeptideName"	"210820_Grad090_LFQ_A_01.mzML"	"210820_Grad090_LFQ_B_01.r
"1/A2I7N3_LAVSHVIHK"	624895	
"3/sp¦P08729¦K2C7_HUMAN/Q3KNV1/P08729_VDALNDEINFLR"	2037142	1

- `dia/DIA-protein-matrix.tsv` Here the intensities are on the protein group level, with the first column containing the protein group name and size.

ProteinName	210820_Grad090_LFQ_A_01.mzML	210820_Grad090_LFQ_B_01.mzML
1/sp¦O14561¦ACPM_HUMAN	1410100	1359820
2/sp¦P0CX49¦RL18A_YEAST/sp¦P0CX50¦RL18B_YEAST	99249410	44970930

(read `|` instead of `¦`)

2 About this document

This is a literate programming (org-mode) document that describes the the nextflow implemantiation of gladiator (`https://github.com/elolab/glaDIAtor`) and all needed template files. This directory should already contain the tangled files. To learn more about org-mode, see `https://orgmode.org/`.

2.1 Writing Style

Sentences with the first person plural ("we") as subject or with implied third person (it reads as "[The program] . . . "), are notes about the development process or an explanation of the program, whereas sentences with the second person as subject ("You" e.g. "You might try setting foo to 3") are instructions to the end user.

2.2 Trouble Shooting

When you encounter an error, go to the section of this document that this error occurred, there we will describe fixes for errors that occur in that step. You can also search for the text the error message gave, we will usually include it in that section.

2.3 Tangling

in order to turn this file into the needed files run, you'll need to have emacs and gnu make installed, and then run

```
yes '//' | make tangle
```

alternatively, if you have gnu guix installed, you can run

```
make SHELL=guix tangle
```

The following is a list of all files this document tangles into

gwl-gladiator.scm
gladiator.nf
diaumpireconfig.txt
comet_template.txt
xtandem-template.xml
tpp-5.2-fix.diff
irt.txt
pyprophet-legacy-requirements.txt
pyprophet-legacy-standalone.dockerfile
install-R-packages.R
config/singularity.nf
config/singularity-local.nf
config/docker.nf
config/docker-local.nf
config/podman.nf
config/podman-local.nf
nextflow.config
nextflow.tags

2.4 Building the containers

Containers are available from the public registry, but you can also build them yourself. The containers are defined in terms guix manifests, rather than Dockerfiles or the like, so you will need to install guix in order to build the containers yourself.

```
# if you want singularity images
make SHELL=guix singularity-containers
# if you want docker images
make SHELL=guix docker-containers
```

If you want to pass arguments to the step that's responsible for building the containers, (like e.g. guix style transformations), you can specify those with make variable `GUIX_PACK_FLAGS`. e.g.

```
make SHELL=guix 'GUIX_PACK_FLAGS=--with-patch=tpp=/path/to/my/tpp/patchfile.patch' sing
```

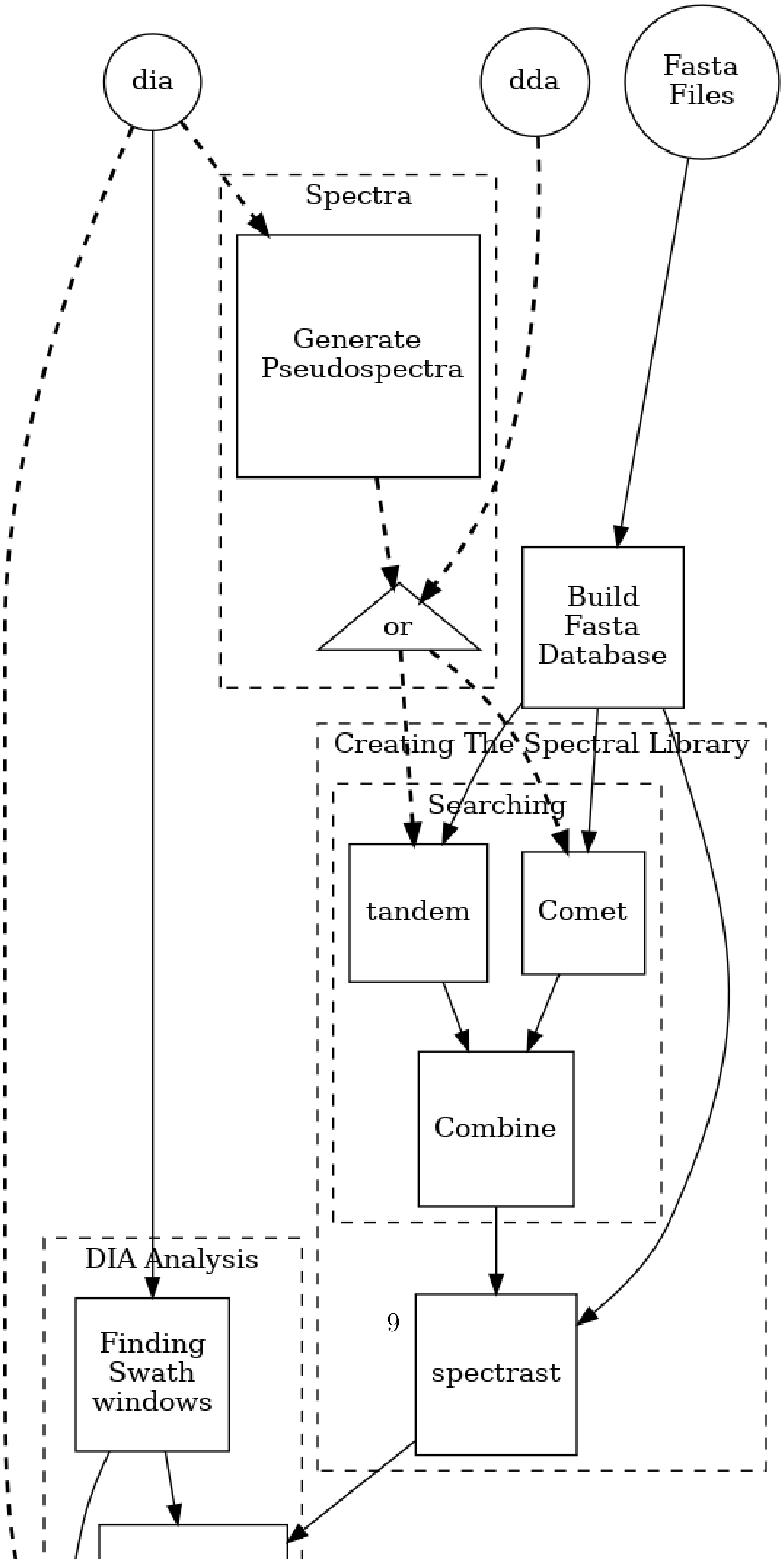
2.5 License

This program is free software, under GPL3 or later.

```
/*
 * Copyright (C) 2025
```

```
*
* This program is free software: you can redistribute it and/or modify
* it under the terms of the GNU General Public License as published by
* the Free Software Foundation, either version 3 of the License, or
* (at your option) any later version.
*
* This program is distributed in the hope that it will be useful,
* but WITHOUT ANY WARRANTY; without even the implied warranty of
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
* GNU General Public License for more details.
*
* You should have received a copy of the GNU General Public License
* along with this program.  If not, see <http://www.gnu.org/licenses/>.
*/
```

3 Overview of this pipeline



4 Preprocessing Data

We will not distribute the vendored msconvert, but if you have DDA-files you need to convert froma propriatry format, to mzmxml, following the picking peaks step, and you can use the docker image of `dockerhub:chambm/pwiz-skyline-i-agree-to-the-ve` You can convert your DIA-files with the same container following "Converting Dia Raw with Msconvert"

4.1 Picking Peaks

```
mkdir -p MZXML-pwiz
for f in RAW/*.wiff; do
    wine qtofpeakpicker --resolution=2000 --area=1 --threshold=1 --smoothwidth=1.1 --i
done
```

4.2 Converting Dia Raw with Msconvert

```
mkdir -p MZML-pwiz
find . -iname '*.wiff' -print0 | xargs -P5 -0 -i wine msconvert {} --filter 'titleMake
```

5 Analysis [4/4]

Gladiator paper: `https://doi.org/10.1038/s43705-022-00137-0` diatools: `https://doi.org/10.1021/acs.jproteome.9b00606`

5.1 Headers

```
(setq org-babel-tangle-lang-exts
      (cl-remove-duplicates
      (append
       '(("scheme" . "scm"))
       org-babel-tangle-lang-exts
       )
      :test 'equal))
```

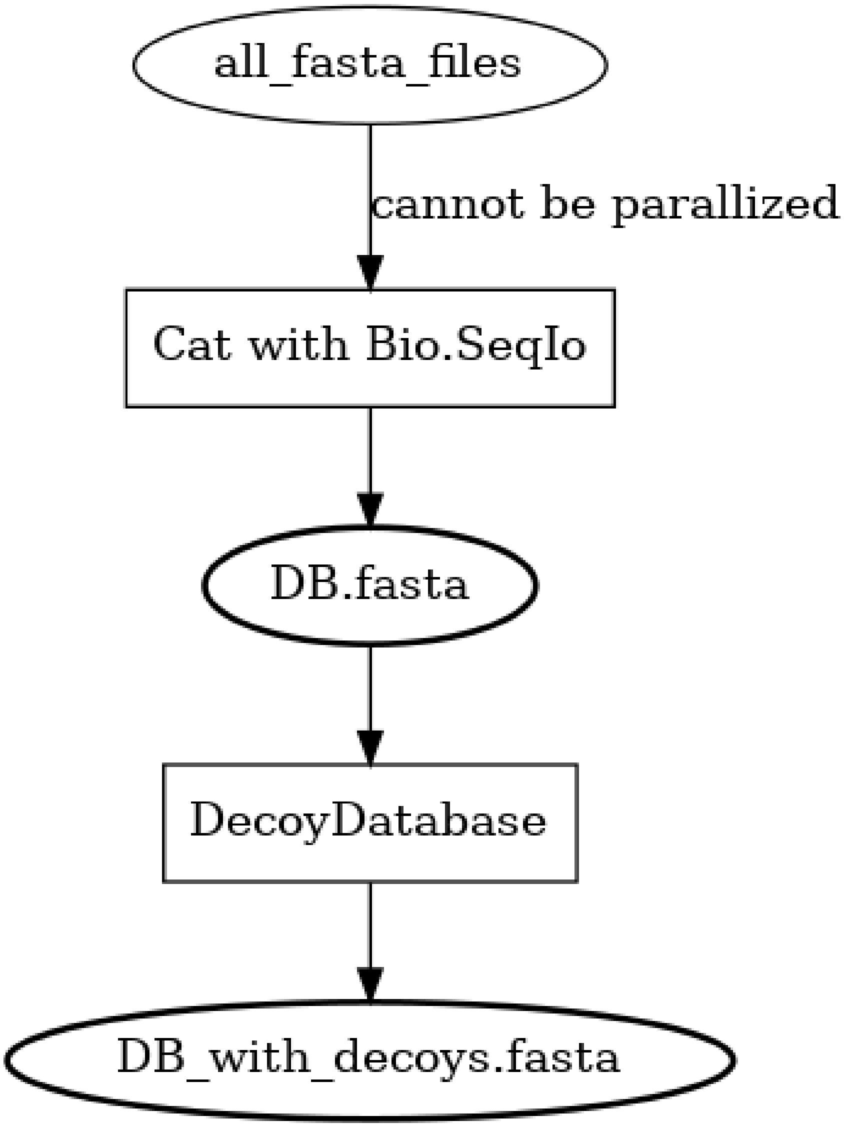
```
(define-module (workflow)
  #:use-module (gwl workflows)
  #:use-module (gwl processes)
  #:use-module (gwl utils)
  #:use-module (gwl sugar))
```

```
<<gwl-vars>>
```

```
<<nf-sdrf-handling>>
<<nf-function-definitions>>
<<nf-vars>>
```

5.2 building database

Overview build_database definition



5.2.1 Combining Fasta Files

```
from Bio import SeqIO
def join_fasta_files(input_files, output_file):
    IDs = set()
    seqRecords = []
    for filename in input_files:
        records = SeqIO.index(filename, "fasta")
        for ID in records:
            if ID not in IDs:
                seqRecords.append(records[ID])
                IDs.add(ID)
            else:
                print("Found duplicated sequence ID " + str(ID) + ", skipping this sequ

    SeqIO.write(seqRecords, output_file, "fasta")
```

```
process JoinFastaFiles {
    input:
    file fasta_files from fasta_files_ch.toSortedList()
    output:
    file 'joined_database.fasta' into joined_fasta_database_ch

    """
    #!/usr/bin/env python3
    <<py-joinfastafiles>>
    join_fasta_files("$fasta_files".split(" "), 'joined_database.fasta')
    """
}
```

```
params.fastafiles='fasta/*.fasta'
```

```
Channel.fromPath(params.fastafiles).set{fasta_files_ch}
```

This was how we could set the fasta_files_ch to be in the same order as the original bruderer run

```
Channel.from([
    "fasta/Q7M135.fasta",
    "fasta/irtfusion.fasta",
    "fasta/trypsin.fasta",
    "fasta/uniprot_human_2017_04_05.fasta",
    "fasta/Bruderer_QS-spike-in-proteins.fasta"])
    .map{file(it)}
    .set({fasta_files_ch})
```

```
("join-fasta-files"
 "python"
 "biopython")
```

```
(define (join-fasta-files fasta-files)
  (make-process
   (name "join-fasta-files")
   (synopsis "Join fasta files into one file")
   (packages
    (cdr (quote
      <<gwl-joinfastafiles-deps>>)))
   (inputs (files fasta-files))
   (outputs "joined-fasta.fasta")
   # python
{
<<py-joinfastafiles>>
join_fasta_files({{inputs}}.split(" "),{{outputs}})
}))

(define fasta-files
  '("Q7M135.fasta" "trypsin.fasta"))

(join-fasta-files fasta-files)
```

5.2.2 Adding Decoys

```
fasta_db_with_decoys = Channel.value()
process BuildFastaDatabase {
    input:
    file joined_fasta_db from joined_fasta_database_ch
    output:
    file "DB_with_decoys.fasta" into joined_fasta_with_decoys_ch
    """
    DecoyDatabase -in $joined_fasta_db -out DB_with_decoys.fasta
    """
}
```

DecoyDatabase package is from OpenMs/utils https://abibuilder.informatik.uni-tuebingen.de/archive/openms/Documentation/release/latest/html/UTILS_DecoyDatabase.html https://github.com/OpenMS/OpenMS.git License: BSD-3 clause (Not in guix, but uses cmake as build-program, should be relatively easy to define)

```
(define create-database-with-decoys
  (make-process
   (name "create-database-with-database")
   (synopsis "Add decoys to fasta database")
   (inputs "joined-fasta.fasta")
   (outputs "DB-with-decoys.fasta")
   (packages )
   # sh
     {
      DecoyDatabase -in $inputs -out $outputs
                    }))
```

```
create-database-with-decoys
```

5.3 Input DIA files

Here we redirect the dia files to the

```
Channel
    .fromPath(params.diafiles)
    .multiMap{
        it -> swath_windows: osw: it}
    .set{dia_mzml_files_ch}
```

5.4 Creating Swath window files

`glaDIAtor/workflow.py` outputs files `swath-windows.txt`, `truncated-swath-windows.txt`

If you are using FAIMS split MZMLs, the mzml might not contain `isolationWindow` elements, in that case you can provide your own tab-separated file of swathwindows.

5.4.1 Branching if user supplied windows

```
// optional swath windows file thats a tab-separated file
// where the first column is the isolation window lower offset
// and the second column  is the isolation window upper offset
// this file is normally automatically generated in the MakeSwathWindows steps
// but if your mzML does not provide isolationWindow
params.swath_windows_file=''
```

```
if (params.swath_windows_file) {
  <<nf-regularize-user-swath-windows>>
} else  {
    <<nf-infer-swath-windows>>
}
```

5.4.2 Making truncated-swath-windows and swath-windows from user-supplied swath-windows

Here we do some mangling so that the user inputed swathwindows is in the same format as the one that would be generated by us.

We keeep `FS` to the default so that awk will happily accept whitespace as field separator, (be it normal spaces or tabs), but we output with tab as separators.

```
process RegularizeUserSwathWindow {
    input:
    path user_swath_windows, stageAs: 'userSwathWindow.txt' from Channel.fromPath(param
    output:
    file swath_windows into swath_windows_ch
    script:
    swath_windows="swath-windows.txt"
    """
    sort -n $user_swath_windows | awk 'BEGIN {OFS="       "} {print \$1,\$2}' >  $swat
    """
}
```

5.4.3 Inferring Windows from mzml files

If the user didnt supply a swath windows file, we infer it from the mzml file

```
import xml.etree.ElementTree as ET
import os

def read_swath_windows(dia_mzML):

    print ("DEBUG: reading_swath_windows: ", dia_mzML)

    context = ET.iterparse(dia_mzML, events=("start", "end"))

    windows = {}
    for event, elem in context:

        if event == "end" and elem.tag == '{http://psi.hupo.org/ms/mzml}precursor':
            il_target = None
            il_lower = None
            il_upper = None
            isolationwindow = elem.find('{http://psi.hupo.org/ms/mzml}isolationWindow')
            if isolationwindow is None:
                raise RuntimeError("Could not find isolation window; please supply --sw
            for cvParam in isolationwindow.findall('{http://psi.hupo.org/ms/mzml}cvPara
                name = cvParam.get('name')
                value = cvParam.get('value')

                if (name == 'isolation window target m/z'):
                    il_target = value
                elif (name == 'isolation window lower offset'):
                    il_lower = value
                elif (name == 'isolation window upper offset'):
                    il_upper = value

            ionList = elem.find('{http://psi.hupo.org/ms/mzml}selectedIonList')

            selectedion = ionList.find('{http://psi.hupo.org/ms/mzml}selectedIon')

            if selectedion:

                for cvParam in selectedion.findall('{http://psi.hupo.org/ms/mzml}cvPara
                    name = cvParam.get('name')
                    value = cvParam.get('value')

                    if (name == 'selected ion m/z'):
                        if not il_target:
                            il_target = value

            if not il_target in windows:
                windows[il_target] = (il_lower, il_upper)
            else:
                lower, upper = windows[il_target]
                assert (il_lower == lower)
                assert (il_upper == upper)
                return windows

    return windows

def create_swath_window_files(cwd, dia_mzML):
    windows = read_swath_windows(dia_mzML)
    swaths = []
    for x in windows:
        target_str = x
        lower_str, upper_str = windows[x]
        target = float(target_str)
        lower = float(lower_str)
        upper = float(upper_str)
        assert (lower > 0)
        assert (upper > 0)
        swaths.append((target - lower, target + upper))
    swaths.sort(key=lambda tup: tup[0])
    # here we use chr(10) (equivalent to slash n), and chr(9) (equivalent to slash t)
    newline_character = chr(10)
    tab_character = chr(9)
    with open(os.path.join(cwd, "swath-windows.txt"), "w") as fh_swaths:
        for lower,upper in swaths:
            fh_swaths.write(str(lower) + tab_character + str(upper)  + newline_characte
    return fh_swaths
```

```
process InferSwathWindows {
    input:
    file diafile from dia_mzml_files_ch.swath_windows.first()
    output:
    file "swath-windows.txt" into  swath_windows_ch
    shell:
    '''
    #!/usr/bin/env python3
    <<py-makeswathwindows>>
    swaths = create_swath_window_files(".","!{diafile}")
    '''
}
```

we'll have to get minswath and maxswath by reading "swath-windows.txt"

5.4.4 Making the non-overlapping swath-windows

Openswath requires non-overlapping windows, so we create them here.

```
BEGIN {OFS="        "}
function max(a,b){
    if(a > b)
        return a
    return b
}
NR==1 {
    # we start with the special case that the boundary for the first entry
    # should be unchanged
    prev_upper=$1
    # and we add the column names
    print "LowerOffset","HigherOffset"
}
{
    if (prev_upper > $2)
    {
        print "There is a a window thats a subwindow of the previous window"
        exit 1
    }
    print(max($1,prev_upper),$2)
    prev_upper=$2
}

process InferNonOverlappingSwathWindows {
    input:
    file swath_windows from swath_windows_ch.first()
    output:
    file truncated_swath_windows into truncated_swath_windows_ch
    script:
    truncated_swath_windows="truncated_swath_windows.txt"
    ''' awk '
    <<awk-infer-non-overlapping-swath-windows>>' ''' + "$swath_windows > $truncated_swa
}
```

5.5 Library Generation

There are various way to generate spectral libraries from DIA data / DDA dat. Here we make the distinction between deconvolution methods and other library generation methods.

The following is a list of the methods we support,

```
[ "dda","custom", "deepdia", "diaumpire","diams2pep"]
```

And you can adjust the following parameter

```
// one or more of [ "dda","custom", "deepdia", "diaumpire","diams2pep"] seperated by c
// will default to "dda" if ddafiles are supplied
// othewise to "deepdia"
params.libgen_method = null

// TODO: raise an error if params.libgen_method  is not a supported method
libgen_methods_validate_params(params)

// returns all libgen methods that we supplor
def libgen_methods_get_existing (){
    return [ "dda","custom", "deepdia", "diaumpire","diams2pep"]
}


def libgen_method_any_pseudospectra_method_is_enabled(params){
    def pseudospectra_methods = ["diams2pep","diaumpire"]
    return pseudospectra_methods.inject(false) { acc, val -> acc || libgen_method_is_en
}



def libgen_methods_validate_params(params){
    if(params.libgen_method != null){
        def invalid_methods = params.libgen_method.split(",").findAll({           !libg
        if(invalid_methods)
            raise RunTimeError("Invalid libgen methods specified: " + invalid_methods.
    }
}
def libgen_method_is_enabled(method, params){
    // method to use if the user didnt specify anything
    def fallback_method = "diaumpire";
    if (params.libgen_method){
        return params.libgen_method.split(",").contains(method)
    }
    switch (method) {
            case "dda": return !!params.ddafiles;
            case "custom": return !!params.speclib;
        default: return (method == fallback_method) && !params.ddafiles && !params.spe
    }
}
```

```
def libgen_method_is_exclusively_enabled(method, params) {
    return libgen_methods_get_existing().inject(true) { acc, val -> acc && ( libgen_met
}
```

5.5.1 Building {Pseudo-,}Spectral library from (Pseudo)-Spectra [5/5]

This section covers seval ways of deconvolution for making spectral libraries for later usage by open swath.

The below block handles the logic of dealing with the various deconvolution methods, sending diafiles to all input channels, and getting output mgf from output channels.

```
deconvolution_methods = []
<<nf-deconvolution-handling>>
deconv_input_chs = deconvolution_methods*.input.findAll({it != null})
if(deconv_input_chs){
    Channel
        .fromPath(params.diafiles)
        .into(
            deconv_input_chs
                .inject() { acc, val -> acc << val })
}
deconv_output_chs = deconvolution_methods*.output.findAll({it != null})
for(ch:deconv_output_chs)
    Channel.create().set(ch.clone())
if(deconv_output_chs){
    Channel.empty()
        .mix(*(deconv_output_chs*.call()))
        .multiMap{ it -> spectrast: comet: xtandem: it }
        .set{maybespectra_ch}
}

if(libgen_method_any_pseudospectra_method_is_enabled(params) || libgen_method_is_enable
```

Using DDA Data

```
if(libgen_method_is_enabled("dda",params)){
    deconvolution_methods += [output: { dda_files_ch } ]
}
```

```
if(libgen_method_is_enabled("dda",params)){
    Channel.fromPath(params.ddafiles).tap(dda_files_ch)
}
```

Creating Pseudospectra with DIAumpire `glaDIAtor/workflow.py`

[width=.9]img/dot-create-pseudospectra

https://github.com/Nesvilab/DIA-Umpire/tree/master https://github.com/Nesvilab/DIA-Umpire/raw/gh-pages/DIA_Umpire_Manual_v2.0.pdf

Problems you might encounter during this step

1. Out of Memory in Dia-umpire Dia-umpire, which we use here for pseudo-spectra creation, has pretty extreme memory requirements, in your config file you can set the process specific memory (required to be in Gigabyes) e.g.

   ```
   process {
     withName: 'GeneratePseudoSpectra'
     {
           time='96h'
           memory='400 GB'
     }
   }
   ```

   see also The Nextflow documentation about process memory

2. `MzmlToMzxml` processing error. If you get an error of in `MzmltoMzxml`, that can mean that something went wrong when you used `msconvert` to convert from the propriatary format to mzml

Steps that are run DIAumpire is `Apache 2` licensed.

```
if(libgen_method_is_enabled("diaumpire",params)){
    deconvolution_methods += [output: {diaumpire_pseudospectra_ch},
                              input:  {dia_mzml_files_for_diaumpire_ch}]
}
```

```
if(libgen_method_is_enabled("diaumpire",params)){
```

MGF = Mascot Generic Format https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3518119/

```
// create mzxml
process MzmlToMzxml {
    input:
    file diafile from dia_mzml_files_for_diaumpire_ch
    output:
    file "*.mzXML" into dia_mzxml_files_for_diaumpire_ch
    """
    msconvert $diafile --32 --zlib --filter "peakPicking false 1-" --mzXML
    """
}

process GeneratePseudoSpectra  {
    memory '16 GB'
    input:
    file diafile from dia_mzxml_files_for_diaumpire_ch
    path diaumpireconfig from diaumpireconfig_ch.first()
    output:
    // we flatten here becuase a single mzxml might result in multiple mgf files
    file "*.mgf" into diaumpire_pseudospectra_mgf_ch mode flatten

    """
    # we set \$1 to the number of gigs of memory
    set -- $task.memory
    if command -v diaumpire-se;
    then
        diaumpire-se  -Xmx\$1g -Xms\$1g $diafile $diaumpireconfig
    else
        java -Xmx\$1g -Xms\$1g -jar /opt/dia-umpire/DIA_Umpire_SE.jar $diafile $diaumpi
    fi
    """
}

process DiaUmpireMgfToMzxml {
    input:
    file mgf from diaumpire_pseudospectra_mgf_ch
    output:
    file "*.mzXML" into diaumpire_pseudospectra_ch
    when:
    // excluding empty files
    mgf.size()  > 0
    """
    msconvert $mgf --mzXML
    """
}
```

```
("generate-pseudo-spectra"
 "dia-umpire"
 "pwiz") ;; the free one
```

though this might also be done with openms's `FileConverter` ? which is more conventionally build `https://abibuilder.informatik.uni-tuebingen.de/archive/openms/Documentation/release/latest/html/TOPP_FileConverter.html` mstools

```
params.diaumpireconfig='diaumpireconfig.txt'
// glob to DIA mzmML files, e.g. "DIA/*.mzML"
// MANDATORY to be set if not set by SDRF file
params.diafiles = null
// OPTIONAL glob to mzXML dda files
// e.g. "DDA/*.mzXML"
// if left unset, then pseudospectra will be used.
params.ddafiles = null

// so that this is a singleton channel
diaumpireconfig_ch = Channel.fromPath(params.diaumpireconfig)

} // end of diaumpire guard
```

Creating Pseudospectra with diams2pep `https://github.com/SS2proteome/DIA-MS2pep` `https://doi.org/10.52601/bpr.2022.220011`

```
// fragment tolarance for diam2spep in ppm
// (other tools require it in dalton)
params.diams2pep_fragment_tolerance = null

if(libgen_method_is_enabled("diams2pep",params)){
    deconvolution_methods += [output: { diams2pep_pseudospectra},
                              input: { diams2pep_input_mzml}]
}

if(libgen_method_is_enabled("diams2pep",params)){
```

Do we need msconvert to convert to a friendly mzml file? According to DIA-MS2PEP's readme we need

```
mzML=true
zlib=true
mz64=true
inten64=true
simAsSpectra=true
filter=’’peakPicking vendor msLevel=1-2"
```

trying with filter "cwt" because we don't ship vendors.

```
process convert_for_DIAMS2PEP {
    input:
    file mzml from diams2pep_input_mzml
    output:
    // there is no good way in nextflow that makes a UUUID that persists across -resume
    // task.hash is forgotten in resume, as is task.id.
    tuple val("${mzml.baseName}"), path(ofile) into diams2pep_mgf_mzml, diams2pep_windo
    script:
    ofile="converted/${mzml.baseName}.mzML"
    """
    mkdir -p converted
    msconvert --mzML --mz64 --zlib --inten64 --simAsSpectra --filter "peakPicking cwt m
    """
}

process convert_mgf_for_DIAMS2PEP {
    input:
    tuple val(hash), path(mzml) from diams2pep_mgf_mzml
    output:
    tuple val(hash), path("${mzml.baseName}.mgf") into diams2pep_mgf

    """
    msconvert --mgf $mzml
    """
}
```

`DIA_acquistion_window_generator.pl` is not a typo, this is how it is in the original repo.

```
process DIAMS2PEP_window {
    input:
    tuple val(hash), path(mzml) from  diams2pep_window_mzml
    output:
    tuple val(hash), path("${mzml}.DIA_acquisition_window.txt") into diams2pep_window

    """
    DIA_acquistion_window_generator.pl $mzml
    """
}
```

Looking at the source, this creates an mgf file for every window that was detected. which would not be known before running `DIA_acquisition_window_generator.pl`

`DIA_pesudo_MS2.pl`.pl= is not a typo, this is how it is in the original repo

```
if(params.diams2pep_fragment_tolerance == null)
    raise RunTimeError("DIAMSM2PEP enabled but no diams2pep_fragment_tolerance specifie
process DIAMS2PEP_generate_pseudo {
    input:
    tuple val(hash), path(mzml), path(mgf), path(acq_window) from diams2pep_for_pseudo_
    val tolerance from Channel.value(params.diams2pep_fragment_tolerance)
    output:
    file "mgf-output/*.mgf" into diams2pep_pseudospectra_mgf mode flatten
    """
    mkdir -p mgf-output
    DIA_pesudo_MS2_multiforks.pl ${mzml.baseName} mgf-output $tolerance ${task.cpus}
    """
}

// this will use the default container because we need msconvert
process MgfToMzml_DIAMS2PEP {
    input:
    file mgf from diams2pep_pseudospectra_mgf
    output:
    file "*.mzXML" into diams2pep_pseudospectra
    """
    msconvert --mzXML $mgf
    """
}

} // end of diams2pep guard
```

Choosing the MS Sequence database search engine: Comet/Xtandem

```
params.search_engines = ["comet","xtandem"]
```

Depending on your experimental machine, the precursor and fragment tolerances are different. These are parameters to all search engines unsed.

Some scientific papers use `mmu` which is equal to 1 milidalton 0.001 Dalton

```
// Float or Int; in ppm ; eg. params.precursor_mass_tolerance=10
params.precursor_mass_tolerance=null
// Float or Int; in Dalton; e.g. parames.fragment_mass_tolerance=0.2
params.fragment_mass_tolerance=null
```

Notably, a more stringent (lower) tolerance increases memorary usage by comet.

the maximum number of allowed missed cleavages is also passed to all search engines. **Mandatory** if you want to use these search engines.

```
// Int, if you are using the comet, this can by at maximum 5,
params.max_missed_cleavages=null

max_missed_cleavages = Channel.value(params.max_missed_cleavages)
```

However, in this developers experience, `xtandem` will crash when using another `max-missed-cleavages` then 1 so you would put e.g. the following in your config file.

```
params.search_engines = ["comet"]
params.max_missed_cleavages= 2
```

Comet `glaDIAtor/UI/ui/__init__.py` `glaDIAtor/workflow.py` `https://github.com/UWPR/Comet`

The following is the template file for the parameters passed to comet. you can change fields for things that we don't give parameters for (so where the value is not "@...@"), in order to change behaviour of comet specific to your use case. For more information, see comet's documentation: `https://comet-ms.sourceforge.net/parameters/parameters_202101/`

Here we set the above as the default parameter template. If you customized or have your own the comet config, you can point to it with this.

```
params.comet_template="comet_template.txt"
```

```
process MakeCometConfig {
    // should we instead return a tuple here of fastadb and config
    // because the config.txt refers to it?
    input:
    val max_missed_cleavages
    file fastadb_with_decoy from joined_fasta_with_decoys_ch.first()
    path template from Channel.fromPath(params.comet_template)
    output:
    file "comet_config.txt" into comet_config_ch
    """
    sed 's/@DDA_DB_FILE@/$fastadb_with_decoy/g;s/@FRAGMENT_MASS_TOLERANCE@/$params.frag
    """

}
```

setting memory & error strategy like this prevents caching even with process.cache='lenient' maybe because the task.attempt = 1 is tried first

```
process Comet {
    // we probably also want to publish thees
    memory { 5.GB * 2 *  task.attempt }
    errorStrategy { task.exitStatus in 137..137 ? 'retry' : 'terminate' }
    maxRetries 2
    input:
    file comet_config from comet_config_ch.first()
    // future dev: we can .mix with DDA here?
    // though we might need to tag for DDA / Pseudo
    // so that xinteract
    file mzxml from maybespectra_ch.comet
    file fastadb_with_decoy from joined_fasta_with_decoys_ch.first()
    output:
    file("${mzxml.baseName}.pep.xml") into comet_pepxml_ch
    file mzxml into xinteract_comet_mzxml_ch
    when:
    params.search_engines.contains("comet")

    """
    if command -v command-ms;
    then
      comet-ms -P$comet_config $mzxml
    else
```

```
      comet -P$comet_config $mzxml
    fi
    """
}

process XinteractComet {
    memory '16 GB'
    time '5h'
    // memory usage scales with the number of input files
    // find the correct usage per input file or size
    // also for xinteractxtandem
    // usage there seems to be a lot smaller
    // as input files seems to be smaller
    input:
    file pepxmls from comet_pepxml_ch.toSortedList()
    // the filename of needed fastdadb was defined in cometcfg
    // and stored in pepxml in the comet-ms step
    // -a suppplies the absulute path to the data directory where the mzxmls
    // rather than reading wherer the mfrom the xmls
    // where the mzxml are, because its not very
    // nextflow to look outside the cwd.
    file fastadb_with_decoy from joined_fasta_with_decoys_ch.first()
    file mzxmls from  xinteract_comet_mzxml_ch.toSortedList()
    output:
    file "interact_comet.pep.xml" into comet_search_results_ch
    when:
    pepxmls.size() > 0
    """
    xinteract -a\$PWD -OARPd -dDECOY_ -Ninteract_comet.pep.xml $pepxmls
    """
}
```

Xtandem `glaDIAtor/UI/ui/__init__.py` `glaDIAtor/workflow.py`

As with tandem, you an adjust the file "xtandem-template.xml" to suit your needs, values with `@...@` are automatically replaced. See also the documentation of xtandem here: `https://www.thegpm.org/TANDEM/api/`

```
<?xml version="1.0"?>
<bioml label="x! taxon-to-file matching list">
  <taxon label="DB">
```

```
    <file format="peptide" URL="%s" />
  </taxon>
</bioml>

<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="tandem-input-style.xsl"?>
<bioml>
<note>list path parameters</note>

<note>spectrum parameters</note>
        <note type="input" label="spectrum, fragment monoisotopic mass error">@FRAGMEN
        <note type="input" label="spectrum, parent monoisotopic mass error plus">@PRECU
        <note type="input" label="spectrum, parent monoisotopic mass error minus">@PREC
        <note type="input" label="spectrum, parent monoisotopic mass isotope error">yes
        <note type="input" label="spectrum, fragment monoisotopic mass error units">Dal
        <note>The value for this parameter may be 'Daltons' or 'ppm': all other values
        <note type="input" label="spectrum, parent monoisotopic mass error units">ppm</
                <note>The value for this parameter may be 'Daltons' or 'ppm': all other
        <note type="input" label="spectrum, fragment mass type">monoisotopic</note>
                <note>values are monoisotopic|average </note>

<note>spectrum conditioning parameters</note>
        <note type="input" label="spectrum, dynamic range">100.0</note>
                <note>The peaks read in are normalized so that the most intense peak
                is set to the dynamic range value. All peaks with values of less that
                1, using this normalization, are not used. This normalization has the
                overall effect of setting a threshold value for peak intensities.</note
        <note type="input" label="spectrum, total peaks">50</note>
                <note>If this value is 0, it is ignored. If it is greater than zero (le
                then the number of peaks in the spectrum with be limited to the 50 most
                peaks in the spectrum. X! tandem does not do any peak finding: it only
                limits the peaks used by this parameter, and the dynamic range paramete
        <note type="input" label="spectrum, maximum parent charge">4</note>
        <note type="input" label="spectrum, use noise suppression">yes</note>
        <note type="input" label="spectrum, minimum parent m+h">500.0</note>
        <note type="input" label="spectrum, minimum fragment mz">150.0</note>
        <note type="input" label="spectrum, minimum peaks">15</note>
        <note type="input" label="spectrum, threads">40</note>
        <note type="input" label="spectrum, sequence batch size">1000</note>
```

```
<note>residue modification parameters</note>
        <note type="input" label="residue, modification mass">57.022@C</note>
                <note>The format of this parameter is m@X, where m is the modfication
                mass in Daltons and X is the appropriate residue to modify. Lists of
                modifications are separated by commas. For example, to modify M and C
                with the addition of 16.0 Daltons, the parameter line would be
                +16.0@M,+16.0@C
                Positive and negative values are allowed.
                </note>
        <note type="input" label="residue, potential modification mass">16@M</note>
                <note>The format of this parameter is the same as the format
                for residue, modification mass (see above).</note>
        <note type="input" label="residue, potential modification motif"></note>
                <note>The format of this parameter is similar to residue, modification
                with the addition of a modified PROSITE notation sequence motif specif
                For example, a value of 80@[ST!]PX[KR] indicates a modification
                of either S or T when followed by P, and residue and the a K or an R.
                A value of 204@N!{P}[ST]{P} indicates a modification of N by 204, if it
                is NOT followed by a P, then either an S or a T, NOT followed by a P.
                Positive and negative values are allowed.
                </note>

<note>protein parameters</note>
        <note type="input" label="protein, taxon">other mammals</note>
                <note>This value is interpreted using the information in taxonomy.xml.<
        <note type="input" label="protein, cleavage site">[RK]|{P}</note>
                <note>this setting corresponds to the enzyme trypsin. The first charact
                in brackets represent residues N-terminal to the bond - the '|' pipe -
                and the second set of characters represent residues C-terminal to the
                bond. The characters must be in square brackets (denoting that only
                these residues are allowed for a cleavage) or french brackets (denoting
                that these residues cannot be in that position). Use UPPERCASE characte
                To denote cleavage at any residue, use [X]|[X] and reset the
                scoring, maximum missed cleavage site parameter (see below) to somethin
                </note>
        <note type="input" label="protein, modified residue mass file"></note>
        <note type="input" label="protein, cleavage C-terminal mass change">+17.002735<
        <note type="input" label="protein, cleavage N-terminal mass change">+1.007825</
        <note type="input" label="protein, N-terminal residue modification mass">0.0</
        <note type="input" label="protein, C-terminal residue modification mass">0.0</
```

```
                <note type="input" label="protein, homolog management">no</note>
                <note>if yes, an upper limit is set on the number of homologues kept for a part
                <note type="input" label="protein, quick acetyl">no</note>
                <note type="input" label="protein, quick pyrolidone">no</note>

        <note>model refinement parameters</note>
                <note type="input" label="refine">yes</note>
                <note type="input" label="refine, modification mass"></note>
                <note type="input" label="refine, sequence path"></note>
                <note type="input" label="refine, tic percent">20</note>
                <note type="input" label="refine, spectrum synthesis">yes</note>
                <note type="input" label="refine, maximum valid expectation value">0.1</note>
                <note type="input" label="refine, potential N-terminus modifications">+42.01056


                <note type="input" label="refine, potential C-terminus modifications"></note>
                <note type="input" label="refine, unanticipated cleavage">yes</note>
                <note type="input" label="refine, potential modification mass"></note>
                <note type="input" label="refine, point mutations">no</note>
                <note type="input" label="refine, use potential modifications for full refineme
                <note type="input" label="refine, point mutations">no</note>
                <note type="input" label="refine, potential modification motif"></note>
                <note>The format of this parameter is similar to residue, modification mass,
                        with the addition of a modified PROSITE notation sequence motif specifi
                        For example, a value of 80@[ST!]PX[KR] indicates a modification
                        of either S or T when followed by P, and residue and the a K or an R.
                        A value of 204@N!{P}[ST]{P} indicates a modification of N by 204, if it
                        is NOT followed by a P, then either an S or a T, NOT followed by a P.
                        Positive and negative values are allowed.
                        </note>

        <note>scoring parameters</note>
                <note type="input" label="scoring, minimum ion count">4</note>
                <note type="input" label="scoring, maximum missed cleavage sites">@MAX_MISSED_C
                <note type="input" label="scoring, x ions">no</note>
                <note type="input" label="scoring, y ions">yes</note>
                <note type="input" label="scoring, z ions">no</note>
                <note type="input" label="scoring, a ions">no</note>
                <note type="input" label="scoring, b ions">yes</note>
                <note type="input" label="scoring, c ions">no</note>
                <note type="input" label="scoring, cyclic permutation">no</note>
                        <note>if yes, cyclic peptide sequence permutation is used to pad the sc
                <note type="input" label="scoring, include reverse">no</note>
                        <note>if yes, then reversed sequences are searched at the same time as
                <note type="input" label="scoring, cyclic permutation">no</note>
                <note type="input" label="scoring, include reverse">no</note>

        <note>output parameters</note>
                <note type="input" label="output, log path"></note>
                <note type="input" label="output, message">testing 1 2 3</note>
                <note type="input" label="output, one sequence copy">no</note>
                <note type="input" label="output, sequence path"></note>
                <note type="input" label="output, path">output.xml</note>
                <note type="input" label="output, sort results by">protein</note>
                        <note>values = protein|spectrum (spectrum is the default)</note>
                <note type="input" label="output, path hashing">no</note>
                        <note>values = yes|no</note>
                <note type="input" label="output, xsl path">tandem-style.xsl</note>
                <note type="input" label="output, parameters">yes</note>
                        <note>values = yes|no</note>
                <note type="input" label="output, performance">yes</note>
                        <note>values = yes|no</note>
                <note type="input" label="output, spectra">yes</note>
                        <note>values = yes|no</note>
                <note type="input" label="output, histograms">yes</note>
                        <note>values = yes|no</note>
                <note type="input" label="output, proteins">yes</note>
                        <note>values = yes|no</note>
                <note type="input" label="output, sequences">yes</note>
                        <note>values = yes|no</note>
                <note type="input" label="output, one sequence copy">no</note>
                        <note>values = yes|no, set to yes to produce only one copy of each prot
                <note type="input" label="output, results">valid</note>
                        <note>values = all|valid|stochastic</note>
                <note type="input" label="output, maximum valid expectation value">0.1</note>
                        <note>value is used in the valid|stochastic setting of output, results
                <note type="input" label="output, histogram column width">30</note>
                        <note>values any integer greater than 0. Setting this to '1' makes cut
                        into spread sheet programs easier.</note>
        <note type="description">ADDITIONAL EXPLANATIONS</note>
```

```
        <note type="description">Each one of the parameters for X! tandem is entered as
                        node. In the current version of X!, keep those note nodes
                        on a single line.
        </note>
        <note type="description">The presence of the type 'input' is necessary if a not
                        an input parameter.
        </note>
        <note type="description">Any of the parameters that are paths to files may requ
                        particular installation. Full path names usually cause the leas
                        but there is no reason not to use relative path names, if that
                        most convenient.
        </note>
        <note type="description">Any parameter values set in the 'list path, default pa
                        reset by entries in the normal input file, if they are present
                        the default set is used.
        </note>
        <note type="description">The 'list path, taxonomy information' file must exist
                </note>
        <note type="description">The directory containing the 'output, path' file must
                </note>
        <note type="description">The 'output, xsl path' is optional: it is only of use
                </note>
</bioml>

<?xml version="1.0"?>
<bioml>
        <note>
        Each one of the parameters for x! tandem is entered as a labeled note node.
        Any of the entries in the default_input.xml file can be over-ridden by
        adding a corresponding entry to this file. This file represents a minimum
        input file, with only entries for the default settings, the output file
        and the input spectra file name.
        See the taxonomy.xml file for a description of how FASTA sequence list
        files are linked to a taxon name.
        </note>

        <note type="input" label="list path, default parameters">%s</note>
        <note type="input" label="list path, taxonomy information">%s</note>

        <note type="input" label="protein, taxon">DB</note>
        <note type="input" label="spectrum, path">%s</note>

        <note type="input" label="output, path">%s</note>
</bioml>
```

We are making the `xtandem_taxonomy` xml in the same process because its kinda a pseudo dependency

```
process MakeXtandemConfig {
    input:
    file template from Channel.fromPath(params.xtandem_template)
    file fastadb_with_decoy from joined_fasta_with_decoys_ch.first()
    val max_missed_cleavages
    output:
    file "xtandem_config.xml" into xtandem_config_ch
    """
    sed 's/@DDA_DB_FILE@/$fastadb_with_decoy/g;s/@FRAGMENT_MASS_TOLERANCE@/$params.frag
    """
}


process XTandem {
    when:
    params.search_engines.contains("xtandem")

    input:
    file mzxml from maybespectra_ch.xtandem
    file tandem_config from xtandem_config_ch.first()
    file fastadb_with_decoy from joined_fasta_with_decoys_ch.first()
    output:
    file("${mzxml.baseName}.tandem.pep.xml") into xtandem_pepxml_ch
    file mzxml into xinteract_xtandem_mzxml_ch
    """
    printf '
    <<taxonomy-template>>'  $fastadb_with_decoy | tail -n+2 > xtandem_taxonomy.xml

    printf '
    <<xtandem-input-template>>' $tandem_config xtandem_taxonomy.xml $mzxml ${mzxml.base
    tandem input.xml
    Tandem2XML ${mzxml.baseName}.TANDEM.OUTPUT.xml ${mzxml.baseName}.tandem.pep.xml
    """
}

process XinteractXTandem {
    memory '16 GB'
    input:
    file pepxmls from xtandem_pepxml_ch.toSortedList()
    // the filename of needed fastdadb was defined in cometcfg
    // and stored in pepxml in the comet-ms step
    // -a suppplies the absulute path to the data directory where the mzxmls
    // rather than reading wherer the mfrom the xmls
    // where the mzxml are, because its not very
    // nextflow to look outside the cwd.
    file fastadb_with_decoy from joined_fasta_with_decoys_ch.first()
    file mzxmls from  xinteract_xtandem_mzxml_ch.toSortedList()
    output:
    file "interact_xtandem.pep.xml" into xtandem_search_results_ch
    when:
    pepxmls.size() > 0
    """
    xinteract -a\$PWD -OARPd -dDECOY_ -Ninteract_xtandem.pep.xml $pepxmls
    """
}
```

If you customized or have your own the xtandem template, you can point to it with this.

```
params.xtandem_template="xtandem-template.xml"
```

# pepxml	size pepxml (GiB)
69	0.7890625

Why is this much smaller than comet?

We patch tpp to accept comet 2019015

```
diff -Naur release_5-2-0/extern/Makefile release_5-2-0_mod/extern/Makefile
--- release_5-2-0/extern/Makefile        2020-07-29 23:43:45.483620066 +0300
+++ release_5-2-0_mod/extern/Makefile        2020-07-29 23:47:41.796860274 +0300
@@ -339,7 +339,7 @@
```

```
 #
 # http://comet-ms.sourceforge.net/
 #
-COMET_VER := 2018014
+COMET_VER := 2019015
 COMET_ZIP := $(TPP_EXT)/comet_source_$(COMET_VER).zip
 COMET_SRC := $(BUILD_SRC)/comet_source_$(COMET_VER)
 .PHONY: comet comet-source comet-clean
```

Joining Comet & Xtandem `glaDIAtor/workflow.py` `glaDIAtor/workflow.py` the tap seems to hap after nextflow has stopped, look more into this.

possible causes: Queue remains open when data is staged from an external source · Issue #2502 · nextflow-io/nextflow · GitHub Parent nextflow process doesn't exit after all compute tasks are complete · Issue #1230 · nextflow-io/nextflow · GitHub

```
// we handle the one or two engines case
// DSL2 incompat
// would be in workflow body

if (params.search_engines.size() > 1) {
    process CombineSearchResults {
        publishDir "${params.outdir}/speclib"
        when:

        input:
        file xtandem_search_results from xtandem_search_results_ch
        file comet_search_results from comet_search_results_ch
        output:
        file "lib_iprophet.peps.xml" into combined_search_results_ch
        """
        InterProphetParser DECOY=DECOY_ THREADS=${task.cpus} $xtandem_search_results $c
        """
    }
} else if (params.search_engines.contains("comet")) {
    combined_search_results_ch = comet_search_results_ch
} else if (params.search_engines.contains("xtandem")) {
    combined_search_results_ch =xtandem_search_results_ch
} else {
    combined_search_results_ch = Channel.create()
}
```

Building Specral Library glaDIAtor/UI/ui/__init__.py glaDIAtor/workflow.py Inputs from Creating Swath window files http://www.openswath.org/en/latest/docs/openswath.html

Mayu https://doi.org/10.5167/uzh-28712 https://doi.org/10.1074/mcp.M900317-MCP200

> GENERAL: Mayu is a software package to determine protein identification false discovery rates (protFDR) and peptide identification false discovery rates (pepFDR) additionally to the peptide-spectrum match false discovery rate (mFDR).

Here is what happens in mayu: For a pepxml file with peptide-spectrum-matches PSM (type of (spectrum,peptide,probability), where the probality is based on the similarity of the theoratical spectrum, mayu determines the peptide-spectrum-match False Detection Rate (mFDR), and protein identification false discovery rates (protFDR). We select a protFDR for which mayu finds a matching mFDR level (no higher than the -G flag) and it will filter everything with a higher mFDR level In the output csv the score column is the the probability in PSM (in mayu documentation "discrimant")

We find the lowest probability that still has an mFDR that matched the above, and that is what we use as the filtering criterian in spectrast

This is what we will than filter on with specrtrast

Hmhf why can't mayu return deterministic filenames. (It incorporates the mayu version number in the filename grumbl), it follows the pattern

```
my $psm_file_base = $out_base . '_psm_';
my $id_csv_file = $psm_file_base
                . $fdr_type
                . $fdr_value . '_'
                . $target_decoy . '_'
                . $version . '.csv';
```

```
process  FindMinimumPeptideProbability {
    input:
    file combined_search_results from combined_search_results_ch.first()
    file fastadb_with_decoy from joined_fasta_with_decoys_ch.first()
    val max_missed_cleavages
    output:
```

```
    env PEPTIDEPROBABILITY into  minimum_peptide_probability
    /* explanation of paramaters
     -G  $params.protFDR           | maximum allowed mFDR of $params.protFDR
     -P protFDR=$params.protFDR:t   |  print out PSMs of targets who have a protFDR of
     -
     -H | defines the resolution of error analysis (mFDR steps)
     -I number of missed cleavages used for database search
     -M | file name base
     */
    script:
    prefix="filtered"
    // you can change this to a glob-pattern (e.g. "*") for future-proofing
    mayu_version="1.07"
    psm_csv="${prefix}_psm_protFDR${params.protFDR}_t_${mayu_version}.csv"
    """
    Mayu.pl -verbose -A $combined_search_results -C $fastadb_with_decoy -E DECOY_ -G $p
    # test if psm_csv was made
    test -e $psm_csv || exit 1
    # test if the results arent empty
    test `wc -l $psm_csv | cut -d' ' -f1` -gt 1 || exit 1
    PEPTIDEPROBABILITY=`cat $psm_csv | cut -f 5 -d ',' |tail -n+2 |sort -u | head -n1`
    """
}
```

Note that sort requires `$TMPDIR` to actually exists and be writable, `$TMPDIR` (the envvar) is inherited from the parent env when run in a container, but not mounted (at least not in Singularity), so if `$TMPDIR` does not exist in the container, this will crash.

```
// sensible values = floats between 0 and 1
// target FDR for mayu
// this is equivalent to the "pvalue" parameter in the original (python) gladiator imp
// which is labeld as "Spectral library building FDR" in the UI
params.protFDR=0.01
```

Spectrast `http://tools.proteomecenter.org/wiki/index.php?title=Software:SpectraST`

Spectrast in `SpectraSTPepXMLLibImporter.cpp` `readFromFile` `processSearchHit` will read the mzxmls contained in the pepxml. It defaults to looking for the mzxml in the CWD otherwise it checks the path in the `base_name` property of msms_run_summary element in `<search_summary` so we need again give the `maybespectra_ch` on. From the above url

> - Creating Consensus Libraries
> - Importing the raw spectra into SpectraST
>
> […] Remember that the .mzXML files must be in the same directories as their corresponding .pepXML files.

Spectrast is from `tpp` Note that spectrast flags are single-dash multi-lettered underscored argument-concatenated.thanks. Its argument-parser is **very** funky, so be careful here. It also doesn't check if illegal flags are given, they will pass silently instead, **grumble.**

1. Converting traml into spectrast friendly format

   ```
   BEGIN {FS="	"; OFS="	"}
   NR==1 {
       for (i=1; i<=NF; i++) {
           f[$i] = i
       }
   }
   NR>1 { print $(f["PeptideSequence"]), $(f["NormalizedRetentionTime"]) }
   ```

   ```
   BEGIN {FS="	"; OFS="	"}
   # we set the column names so that we can look them up later
   NR==1 {
       for (i=1; i<=NF; i++) {
           f[$i] = i
       }
   }
   # use only the last entry in the table per peptide sequence
   NR>1 {
       irt_by_sequence[$(f["PeptideSequence"])] = $(f["NormalizedRetentionTime"])
       peptide_sequences[$(f["PeptideSequence"])]=$(f["PeptideSequence"])
   }
   END {
       for (sequence in peptide_sequences)
           print(sequence,irt_by_sequence[sequence])
   }
   ```

   TargetedFileConverter from `OpenMS`

```
process CreateSpectrastIrtFile {
    input:
    file irt_traml from Channel.fromPath(params.irt_traml_file)
    output:
    file ("irt.txt") into irt_txt_ch
    script:
    intermediate_tsv="intermediate_irt.tsv"
    """
    TargetedFileConverter -in $irt_traml -out_type tsv -out $intermediate_tsv
    """ + '''  awk '
    BEGIN {FS="        "; OFS="        "}
    NR==1 {
        for (i=1; i<=NF; i++) {
            f[$i] = i
        }
    }
    NR>1 { print $(f["PeptideSequence"]), $(f["NormalizedRetentionTime"]) }' ''' +
}
```

2. Running Spectrast

```
// spectrast will create *.splib, *.spidx, *.pepidx,
// note that where-ever a splib goes, so must its spidx and pepidx
///and they must have the same part
process SpectrastCreateSpecLib {
    input:
    file irtfile from irt_txt_ch
    file combined_search_results from combined_search_results_ch.first()
    file fastadb_with_decoy from joined_fasta_with_decoys_ch.first()
    file spectra from maybespectra_ch.spectrast.toSortedList()
    val cutoff from minimum_peptide_probability
    output:
    tuple file ("${prefix}_cons.splib"), file("${prefix}_cons.spidx") into spectra
    file("${prefix}_cons.sptxt") into consensus_lib_sptxt_ch
    script:
    prefix = "SpecLib"
    to_run = "spectrast -cN${prefix} -cIHCD -cf\"Protein! ~ DECOY_\" -cP$cutoff -c
    if (params.use_irt)
        to_run += "-c_IRT$irtfile "
    to_run +=  "$combined_search_results" // spectrast really wants its input-file
    to_run += "\n spectrast -cN${prefix}_cons -cD$fastadb_with_decoy -cIHCD -cAC $
}
```

from original gladiator implementation source is unclear; author forgot.

```
// white-space-delimited file of peptide-sequences and internal retention times
// whether or not to use the retention-
params.use_irt=true
params.irt_traml_file = "iRTAssayLibrary.TraML"
```

Here we forward declary `consensus_pseudospectra_openswath_library_tsv` so that we can later redirect it.

```
consensus_pseudospectra_openswath_library_tsv = Channel.create()

process Spectrast2OpenSwathTsv {
 /*
     Choice parts of sprectrast2.tsv --help

     spectrast2tsv.py
     ----------------
     This script is used as filter from spectraST files to swath input files.
     python spectrast2tsv.py [options] spectrast_file(s)

     -d                  Remove duplicate masses from labeling
     -e                  Use theoretical mass
     -k    output_key    Select the output provided. Keys available: openswath, pe
     -l    mass_limits   Lower and upper mass limits of fragment ions. Example: -l
     -s    ion_series    List of ion series to be used. Example: -s y,b

     -w    swaths_file   File containing the swath ranges. This is used to remove
     -n    int           Max number of reported ions per peptide/z. Default: 20
     -o    int           Min number of reported ions per peptide/z. Default: 3
     -a    outfile       Output file name (default: appends _peakview.txt)
     */
    input:
    file swath_windows from swath_windows_ch.first()
    file sptxt from consensus_lib_sptxt_ch.first()
    output:
    file consensus_pseudospectra_openswath_library_tsv
    script:
    consensus_pseudospectra_openswath_library_tsv="SpecLib_cons_openswath.tsv"
    """
    MINWINDOW=`head -n1 $swath_windows | cut -d'        ' -f1`
    MAXWINDOW=`tail -n1 $swath_windows | cut -d'        ' -f2`
    spectrast2tsv.py -l \$MINWINDOW,\$MAXWINDOW -s y,b -d -e -o 6 -n 6 -w $swath_w
    """
}
```

5.5.2 Building Spectral library from Machine learning

DeepDIA can predict the spectral library from peptide lists See also its documentation `https://github.com/lmsac/DeepDIA/raw/master/README.md` `https://github.com/lmsac/DeepDIA/raw/master/docs/predict_detectability.md` `https://doi.org/10.1038/s41467-019-13866-z`

```
if(libgen_method_is_enabled("deepdia",params))  {

params._deepdia_url = "https://github.com/lmsac/DeepDIA/raw/c5ad2aa50218fcdfd1d44171470
// float or null
// if null, do not use minimum detectability filtering
// if a float, filter
params.deepdia_min_detectability = null
params.deepdia_detectability_model = "${params._deepdia_url}/data/models/detectability/

// list tuples in the form of
// [charge, model, peptidelist]
params.deepdia_ms2_entries = [
    ["2",
     "${params._deepdia_url}/data/models/charge2/epoch_035.hdf5",
     ],
    ["3",
     "${params._deepdia_url}/data/models/charge3/epoch_034.hdf5",
     ]]

Channel
    .from(params.deepdia_ms2_entries)
    .map( {
            charge, model ->
            tuple(charge, file(model))})
    .set{deepdia_ms2_models}
```

```
process DeepDIADigestProtein
{
    input:
    file joined_fasta from joined_fasta_database_ch
    output:
    file deepdia_peptide_list
    script:
    deepdia_peptide_list="deepdia_peptide_list.csv"
    """
    digest_proteins.py --in $joined_fasta --out $deepdia_peptide_list --no-group_duplic
    """
}
```

If we do detectability filtering we mix the filtered peptides with the models, otherwise the unfiltered.

```
deepdia_peptide_list = Channel.create()
if (params.deepdia_min_detectability != null){
    deepdia_peptide_list.set{deepdia_prefilt_peptide_list}
    deepdia_filtered_peptide_list = Channel.create()
    deepdia_filtered_peptide_list
        .tap{deepdia_peptides_for_retention_pred}
        .tap{deepdia_peptides_for_library}
        .combine(deepdia_ms2_models)
        .set{deepdia_ms2_inputs_ch}
} else {
    deepdia_peptide_list
        .tap{deepdia_peptides_for_retention_pred}
        .tap{deepdia_peptides_for_library}
        .combine(deepdia_ms2_models)
        .set{deepdia_ms2_inputs_ch}
}
```

So here we predict detectability of peptides and filter by them, if requested

```
if (params.deepdia_min_detectability != null){
    // we seperate these two so that --resume allows for easy tweaking of --minimum-de
    process DeepDIATrainDetectibility {
        memory '64 GB'
        input:
```

```
        file model from Channel.fromPath(params.deepdia_detectability_model)
        file deepdia_prefilt_peptide_list
        output:
        set file(deepdia_detectability_prediction), file(deepdia_prefilt_peptide_list)
        script:
        deepdia_detectability_prediction="${deepdia_prefilt_peptide_list.baseName}.dete
        "predict_detectability.py --in $deepdia_prefilt_peptide_list --model $model --
    }

    process DeepDIAMinimumDetectabilityFiltering
    {
        input:
        set file(detectability_prediction), file(prefilt_peptide_list) from deepdia_det
        val min_detectability from Channel.value(params.deepdia_min_detectability)
        output:
        file deepdia_filtered_peptide_list
        script:
        deepdia_filtered_peptide_list="deepdia_filtered_peptide_list.csv"
        """
        filter_peptide_by_detectability.py --peptide $prefilt_peptide_list --detect $de
        """
    }
}
```

Then we predict the ms2

```
process DeepDIAPredictCharge {
    memory '64 GB'
    input:
    set file(peptides),charge,file(model)  from deepdia_ms2_inputs_ch
    output:
    file deepdia_ions
    script:
    deepdia_ions="predictions.charge.${charge}.ions.json"
    """
    predict_ms2.py --charge $charge --in $peptides --model $model --out $deepdia_ions
    """
}
```

```
params.deepdia_irt_model =
    "${params._deepdia_url}/data/models/irt/epoch_082.hdf5"
```

```
// params.deepdia_peptides =
//     "${params._deepdia_url}/data/peptides/Pan_human.peptide.csv"

deepdia_irt_model = Channel.fromPath(params.deepdia_irt_model)
deepdia_speclib = Channel.create()

process DeepDIAPredictRetention {
    memory '32 GB'
    input:
    file deepdia_irt_model
    file deepdia_peptides_for_retention_pred
    output:
    file predicted_rt
    script:
    predicted_rt="prediction.irt.csv"
    """
    predict_rt.py --in $deepdia_peptides_for_retention_pred --model $deepdia_irt_model
    """
}

process DeepDIAPredictionsToLibrary {
    memory '32 GB'
    input:
    file predicted_rt
    file ions from deepdia_ions.toSortedList()
    file deepdia_peptides_for_library
    output:
    file deepdia_speclib
    script:
    deepdia_speclib="speclib.tsv"
    """
    build_assays_from_prediction.py --peptide ${deepdia_peptides_for_library} --rt ${pr
    convert_assays_to_openswath.py --in prediction.assay.pickle --out ${deepdia_speclib
    """
}

deepdia_speclib.set{speclib_tsv_for_decoys}

}  // end of deepdia guard
```

5.5.3 Supplying a custom spectral library

You can also supply a custom spectral library by setting `=--speclib` to a glob/path of your customly generated library. This should be in a format that openms's TargetedFileConverte understand, so MaxQuant or OpenMS works.

To check, one can manually inspect whether `TargetedFileConverter --in yourlibrary --out speclib.tsv` looks proper. Pay special attention to the modified peptides field, if this is being parsed correctly. One bug observed by this developer is that this field is repeated after not being parsed correctly. See also `https://abibuilder.cs.uni-tuebingen.de/archive/openms/Documentation/release/2.7.0/html/classOpenMS_1_1TransitionTSVFile.html`

```
// if params.deconvolution method is set
// set this to to spectral libraries tsvs in maxquant / openms / any input format that
params.speclib = null

if (libgen_method_is_enabled("custom", params)){
    Channel.fromPath(params.speclib).set{speclib_tsv_for_decoys}
}
```

5.5.4 combining various spectral libraries into one.

would need to use openms TargetedFileConverter to convert to openswath like tsv, then msproteomicstools tsv2spectrast.py to turn into spectrast. Then merge merge with spectrast with either -cJA or -cJU `http://tools.proteomecenter.org/wiki/index.php?title=Software:SpectraST#Creating_Consensus_Libraries` See aso schubert et al (`https://doi.org/10.1038/nprot.2015.015`)

5.6 OpenSwathDecoys

specrast2tsv.py is from `msproteomicstools` OpenSwathDecoyGenerator from `OpenMS topp`

```
// optional
params.openswath_transitions = ""
// Minimum decoy fraction for open swath decoy generator
// if left unset, gladiator might pick an appropriate one depending on your deconvolut
// should be a fraction between 0.0 and 1.0
params.oswdg_min_decoy_fraction = null
```

```
if(params.oswdg_min_decoy_fraction != null) {
    Channel.value("-min_decoy_fraction ${params.oswdg_min_decoy_fraction}").set{oswdg_a
} else if (libgen_method_is_enabled("deepdia",params)){
 Channel.value("-min_decoy_fraction 0.0").set{oswdg_args}
} else {
  Channel.value("").set{oswdg_args}
}
```

With deepDIA method used there seems to be some problems with generating decoys, so we set `-min_decoy_fraction` to `0.0` in this case.

```
process AddDecoysToOpenSwathTransitions {
    input:
    file speclib_tsv from speclib_tsv_for_decoys.first()
    val oswdg_args
    output:
    file outputfile into openswath_transitions_ch
    script:
    outputfile="SpecLib_cons_decoys.pqp"

    """
    TargetedFileConverter -in $speclib_tsv -out SpecLib_cons.TraML
    OpenSwathDecoyGenerator -decoy_tag DECOY_ -in SpecLib_cons.TraML -out $outputfile
    """
}
```

`TargetedFileConverter` from `openms`

> /usr/bin/TargetedFileConverter: error while loading shared libraries: libQt5Core.so.5: cannot open shared object file: No such file or directory

Here we might actually not need TargetedFileConverter, can give tsv directly to OpenSwathDecoyGenerator. and pass result tsv to OpenSwathWorkflow as `-tr`.

5.7 Building Dia Matrix

https://openswath.org/_/downloads/en/latest/pdf/ https://openswath.org/_/downloads/en/latest/htmlzip/

5.7.1 OpenSwathWorkflow

Creates tsv with `-out_tsv` `glaDIAtor/workflow.py`

Using a the cache will decrease memomry usage at the cost of writes & time

```
// wheter to use -readOptions cacheWorkingInMemory in OSW
// this actually crashes so disabled
params.osw_use_cache = false
// extra flags to pass to OSW
params.osw_extra_flags =  ""
```

The transitions size is larger if the deconvolution method is deepDIA, which will consume more memory.

to use `-out_osw`, `-tr` needs to be a pqp file, If we are using legacy pyprophet we will need to create a tsv

```
process OpenSwathWorkflow_legacy {
        memory { 16.GB * (libgen_method_is_enabled("deepdia",params) ? 2 : 1 )}
        input:
        file dia_mzml_file from dia_mzml_files_ch.osw
        // file openswath_transitions from Channel.fromPath("data_from_original/bruder
        file openswath_transitions from openswath_transitions_ch_for_legacy.first()
        file swath_truncated_windows from truncated_swath_windows_ch.first()
        file irt_traml from Channel.fromPath(params.irt_traml_file).first()
        output:
        file dia_tsv_file  into openswath_tsv_ch
        script:
        dia_tsv_file = "${dia_mzml_file.baseName}-DIA.tsv"
        to_execute =
            "OpenSwathWorkflow " +
            "-force " +
            "-in $dia_mzml_file " +
            "-tr $openswath_transitions " +
            "-threads ${task.cpus} " +
            "-min_upper_edge_dist 1 " +
            "-sort_swath_maps " +
            "-out_tsv ${dia_tsv_file} " +
            "-swath_windows_file $swath_truncated_windows " +
            params.osw_extra_flags + " "
        if (params.use_irt)
```

```
            to_execute +=  "-tr_irt $irt_traml "
        to_execute
}
```

If we are using nonlegacy pyprophet we will need to create an osw

```
openswath_osw_indirect_ch = Channel.create()
openswath_osw_indirect_ch.multiMap{ it ->
    pyprophet_subsample: pyprophet_score  : it}.set{openswath_osw_ch}
process OpenSwathWorkflow {
    memory { 16.GB * (libgen_method_is_enabled("deepdia",params) ? 2 : 1 )}
    input:
    file dia_mzml_file from dia_mzml_files_ch.osw
    // file openswath_transitions from Channel.fromPath("data_from_original/bruderer-pu
    file openswath_transitions from openswath_transitions_ch_for_nonlegacy.first()
    file swath_truncated_windows from truncated_swath_windows_ch.first()
    file irt_traml from Channel.fromPath(params.irt_traml_file).first()
    output:
    file dia_osw_file  into openswath_osw_indirect_ch
    script:
    dia_osw_file = "${dia_mzml_file.baseName}-DIA.osw"
    to_execute =
        "OpenSwathWorkflow " +
        "-force " +
        "-in $dia_mzml_file " +
        "-tr $openswath_transitions " +
        "-threads ${task.cpus} " +
        "-min_upper_edge_dist 1 " +
        "-sort_swath_maps " +
        "-out_osw ${dia_osw_file} " +
        "-swath_windows_file $swath_truncated_windows " +
        params.osw_extra_flags + " "

    if (params.use_irt)
        to_execute +=  "-tr_irt $irt_traml "
    to_execute
}
```

Then here we choose which one to use

```
// we will need the osw to go to various processes
if (params.pyprophet_use_legacy){
    openswath_transitions_ch.into{openswath_transitions_ch_for_legacy}
 <<nf-openswathworkflow-for-pyprophet-legacy>>
} else {
    openswath_transitions_ch.into{openswath_transitions_ch_for_nonlegacy;openswath_tran
  <<nf-openswathworkflow-for-pyprophet-nonlegacy>>
}
```

(Apparently these two cant have the same name, even if they are conditionally declared

> Extraction windows have a gap. Will abort (override with -force)

OpenSwathWorkflow invocation can output tsv XOR osw, not both. You will get exit status 8

> Error: Unexpected internal error (Either out_features, out_tsv or out_osw needs to be set (but not two or three at the same time))

What is the relation between `irt.txt` and `iRTAssayLibrary.TraML`

The `irt.txt` seems to contain pairs of `CompoundList/Peptide@sequence` and `Compoundlist/Peptide/RetentionTimeList/cvParam[@name="normalized retention time"]@value` of the traml file, except

> LFLQFGAQGSPFLK 98.0897

is not present in the traml file.

traml was not gotten from here: `https://db.systemsbiology.net/sbeams/cgi/PeptideAtlas/PASS_View?identifier=PASS00779` `file://ftp:PASS00779@ftp.peptideatlas.org:/` (but is comparable)

> <OA> The files have been downloaded from net and for now they are intended to be used "as is". If there becomes a need to modify those, then there will be a need figure out how to do it. So, it is very much unexplored how to generate those files for now. [10:56] <NA> looks like the .txt contains pairs of "sequence - Normalized retention time" from the traml. The only one in the .txt that isnt in the traml seems to be LFLQFGAQGSPFLK [10:59] <NA> where did you download them froM? [11:00] <OA> That is a good question. I guess it was some example dataset for openswath, but I don't remember which. If you have time and interest, you could try to figure out how iRT assay library should be built. [11:07]

Retention time normalization¶

The retention time normalization peptides are provided using the optional parameter tr_irt in TraML format. We suggest to use the iRTassays.TraML file provided in the tutorial dataset, if the Biognosys iRT-kit was used during sample preparation.

If the iRT-kit was not used, it is highly recommended to use or generate a set of endogenous peptides for RT normalization. A recent publication [5] provides such a set of CiRT peptides suitable for many eukaryotic samples. The TraML file from the supplementary information can be used as input for tr_irt. Since not all CiRT peptides might be found, the flag RTNormalization:estimateBestPeptides should be set to improve initial filtering of poor signals. Further parameters for optimization can be found when invoking OpenSwathWorkflow –helphelp under the RTNormalization section. Those do not require adjustment for most common sample types and LC-MS/MS setups, but might be useful to tweak for specific scenarios.

5 Röst HL, Liu Y, D'Agostino G, Zanella M, Navarro P, Rosenberger G, Collins BC, Gillet L, Testa G, Malmström L, Aebersold R. TRIC: an automated alignment strategy for reproducible protein quantification in targeted proteomics. Nat Methods. 2016 Sep;13(9):777-83. doi: 10.1038/nmeth.3954. Epub 2016 Aug 1. PMID: 27479329

`https://doi.org/10.1038/nmeth.3954` `file://ftp:PASS00788@ftp.peptideatlas.org:/`

also not from here

It seems the traml is based on `file://ftp:PASS00289@ftp.peptideatlas.org:/SGS/assays/OpenSWATH_SM4_iRT_AssayLibrary.TraML` which has also has the same irt times as iRT.txt, except its still missign `LFLQFGAQGSPFLK`, The same irt.txt can be found in `https://github.com/CaronLab/Allele-specific-library-scripts/blob/main/iRT.txt`, published before gladiator, with the retention times also hardcoded in `https://github.com/msproteomicstools/msproteomicstools/raw/master/analysis/spectral_libs/spectrast_updateiRTs.py`

`OpenMs/src/tests/class_tests/openms/data/MRMDecoyGenerator_input.TraML` has the same irts and also contains `LFLQFGAQGSPFLK` so that might also be a good target.

5.7.2 Pyprophet

https://github.com/PyProphet/pyprophet pyprophet License: 3-clause BSD https://doi.org/10.1093/bioinformatics/btu686 https://doi.org/10.1038/s42003-023-04977-x Pyprophet aggregates various quality scores into one score d_score (discriminant_score)

is based on mprophet https://doi.org/10.1038/nmeth.1584

Legacy Pyprophet

```
if (params.pyprophet_use_legacy)
    process pyprophet_legacy {
    publishDir "${params.outdir}/pyprophet/", pattern: "*.csv"
    publishDir "${params.outdir}/reports/pyprophet/", pattern: "*.pdf"
    input:
    file dia_tsv from openswath_tsv_ch
    output:
    file dscore_csv into pyprophet_legacy_ch
    // just for publishing
    file "${dia_tsv.baseName}_report.pdf"
    script:
    seed="928418756933579397"

    dscore_csv="${dia_tsv.baseName}_with_dscore.csv"
    """
    pyprophet --random_seed=${seed} --delim=tab --export.mayu ${dia_tsv} --ignore.inva
    """
}
```

WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
DIA_summary_stat.csv WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
DIA_full_stat.csv WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
DIA_with_dscore.csv WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
DIA_with_dscore_filtered.csv WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mz
DIA_report.pdf WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
DIA_cutoffs.txt WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
DIA_svalues.txt WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
DIA_qvalues.txt WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
DIA_dscores_top_target_peaks.txt WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T
DIA_dscores_top_decoy_peaks.txt WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T
DIA_mayu.cutoff WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-

> DIA_mayu.fasta WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
> DIA_mayu.csv WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
> DIA_scorer.bin WRITTEN: B_D140314_SGSDSsample1_R01_MHRM_T0.mzML-
> DIA_weights.txt

If you get an error here as

> raise Exception("got empty input file")

try running with `params.use_irt=false`

nonlegacy prypophet `http://www.openswath.org/en/latest/docs/pyprophet.html` In order to make the pyrophet step less intensive, by default we subsample to `1 / nr_samples`, as suggested in the pyprophet guide. However you can supply your own here

```
// The ratio (0,1] to subsample by in pyprophet.
// leave to null to use 1 / nr_samples
params.pyprophet_subsample_ratio = null
```

we calculate the subsample ratio from the number of runs

```
if(params.pyprophet_subsample_ratio == null){
    Channel.value(1 / Math.max(Channel.fromPath(params.diafiles).toSortedList().size()
        .set{subsample_ratio}
} else {
    Channel.value(params.pyprophet_subsample_ratio).set{subsample_ratio}
}
```

In `pyprophet subsample` and `pyprophet score` (of nonlegacy pypropet), we will need to pass `--test` to not have random behaviour. This will make the sql behaviour behaviour non-random. Because sql doesn't allow for settign a seed, pyprophet will just select the first `subsample_ratio` of transitions, This can result in no decoy peptides being in the subsampled osw's. This is indicated by the error of pyprophet:

```
Error: At least 10 decoy groups and 10 target groups are required.
```

This leaves you with one of three choices:

1. passing =`--pyprophet_subsample_ratio=1.0`, sacrificing runtime.
2. passing =`--pyprophet_fixed_seed=false`, sacrificing reproducibility.

3. passing =--pyprophet_use_legacy=true, sacrificing "up-to-dateness".

```
params.pyprophet_fixed_seed=true
params.pyprophet_use_legacy=false
```

Below we are following the steps of http://www.openswath.org/en/latest/docs/pyprophet.html#scaling-up

Anything in pyprophet that is not invoked with an --out flag will overwrite the --in file, here we only do that when the --in-file is created in the process

```
if (!params.pyprophet_use_legacy)
    {

process pyprophet_subsample {
    input:
    file dia_osw_file from openswath_osw_ch.pyprophet_subsample
    val subsample_ratio
    output:
    file subsampled_osw
    script:
    subsampled_osw="${dia_osw_file.baseName}.osws"
    pyprophet_seed_flag=(params.pyprophet_fixed_seed ? "--test" : "--no-test")
    """
    pyprophet subsample $pyprophet_seed_flag --in=$dia_osw_file --out=$subsampled_osw -
    """
}
```

pyprophet score requires significantly more memory than pyprophet merge should we split this?

If pyprophet merge or pyprophet score complains about no decoys being found, you can try without passing =--pyprophet_fixed_seed to gladiator.

```
process pyprophet_learn_classifier {
    input:
    file subsampled_osws from subsampled_osw.toSortedList()
    file openswath_transitions from openswath_transitions_ch_for_pyprophet.first()
    output:
    file osw_model
    script:
```

```
    pyprophet_seed_flag=(params.pyprophet_fixed_seed ? "--test" : "--no-test")
    osw_model="model.osw"
    """
    pyprophet merge --template=$openswath_transitions --out=$osw_model $subsampled_osws
    pyprophet score  $pyprophet_seed_flag --in=$osw_model --level=ms1ms2
    """
}

scored_osw_indirect_ch =Channel.create()
scored_osw_indirect_ch.multiMap{it ->
    reduce: backpropagate: it}.set{scored_osw_ch}

process pyprophet_apply_classifier {
    input:
    file osw_model from osw_model.first()
    file osw from openswath_osw_ch.pyprophet_score
    output:
    file scored_osw into scored_osw_indirect_ch
    script:
    pyprophet_seed_flag=(params.pyprophet_fixed_seed ? "--test" : "--no-test")
    scored_osw="${osw.baseName}.scored.${osw.Extension}"
    """
    pyprophet score $pyprophet_seed_flag --in=$osw --out=$scored_osw --apply_weights=$o
    """
}

process pyprophet_reduce {
    input:
    file scored_osw from scored_osw_ch.reduce
    output:
    file reduced_scored_osw
    script:
    reduced_scored_osw="${file(scored_osw.baseName).baseName}.${scored_osw.Extension}r"
    """
    pyprophet reduce --in=$scored_osw --out=$reduced_scored_osw
    """
}

process pyprophet_control_error {
    input:
```

```
    file reduced_scored_osws from reduced_scored_osw.toSortedList()
    file osw_model from osw_model.first()
    output:
    file osw_global_model
    script:
    osw_global_model="model_global.osw"
    """
    pyprophet merge --template=$osw_model --out=$osw_global_model $reduced_scored_osws
    pyprophet peptide --context=global --in=$osw_global_model
    pyprophet protein --context=global --in=$osw_global_model
    """
}

process pyprophet_backpropagate {
    input:
    file osw_scored from scored_osw_ch.backpropagate
    file osw_global_model from osw_global_model.first()
    output:
    file dscore_tsv into pyprophet_nonlegacy_ch
    script:
    base="${file(osw_scored.baseName).baseName}"
    backproposw="${base}.backprop.osw"
    dscore_tsv="${base}.tsv"
    """
    pyprophet backpropagate --in="$osw_scored" --apply_scores="$osw_global_model" --out
    # we supply --format=legacy_merged so that pyprophet export respect the --out param
    pyprophet export --in=$backproposw --format=legacy_merged --out=$dscore_tsv
    """
}
```

https://github.com/PyProphet/pyprophet/issues/49

}

KeyError: 'HOME' in `pyprophet_subsample` If in this section you get the error

> Command error: File "/gnu/store/lvip6h5pamjwmvnkwg60sjb63ph8698k-python-3.9.9/lib/python3.9/os.py", line 679, in getitem raise KeyError(key) from None KeyError: 'HOME'

and you are using podman or docker, you will want to add to your config file the following:

```
process {
    withName: 'pyprophet_.*' {
        containerOptions= '--env HOME="$PWD"'
    }}
```

This will set your home directory to the running directory in pyprophet processes (Note, single quotes (’), rather than double-quotes ("), to prevent expansion

Choosing between legacy and nonlegacy pyprophet Here we choose between legacy or nonlegacy pyprophet

```
if (params.pyprophet_use_legacy)
    pyprophet_legacy_ch.set{pyprophet_ch}
else
    pyprophet_nonlegacy_ch.set{pyprophet_ch}
```

5.7.3 feature-alignment

http://www.openswath.org/en/latest/docs/tric.html of msproteomicstools/analysis/alignmen

Note: If things fail here, because the right fdr cannot be reached, you can try changing precursor_mass_tolerance and fragment_mass_tolerance earlier upstream.

Additionally you can try setting =--use_irt to false, if you are getting a zero-division errors because the input tsvs are empty.

```
process feature_alignment
{
    publishDir "${params.outdir}/dia/"
    input:
    file dscore_csvs from pyprophet_ch.toSortedList()
    output:
    file outfile into feature_alignment_ch
    script:
    outfile = "DIA-analysis-results.csv"

    if (params.use_irt) {
        realign_method = "diRT"
    } else {
        realign_method = "linear"
    }

    "feature_alignment.py " +
        "--method best_overall " +
        "--max_rt_diff 90 " +
        "--target_fdr $params.tric_target_fdr " +
        "--max_fdr_quality $params.tric_max_fdr " +
        "--in $dscore_csvs " +          // will this break on filenames with spaces
        " --realign_method $realign_method " +
        "--out $outfile"
}

// Target FDR used in TRIC alignment in dirT mode [default 0.01]
// This was "trig_target_pvalue" in the original python gladiator implementation
params.tric_target_fdr=0.01
// Maximum FDR for TRIC alignment in dirT mode [default 0.05]
// This was "trig_max_pvalue" in the original python gladiator implementation
params.tric_max_fdr=0.05
```

5.7.4 Swath2stats

```
suppressPackageStartupMessages(library(SWATH2stats))
suppressPackageStartupMessages(library(data.table))

remove_irt <- function(df)
  df[grep("iRT", df[["ProteinName"]], invert=TRUE, fixed=TRUE),, drop=FALSE]

## original gladiator decoy removing behaviour
remove_decoy_strict <- function(df,decoyprefix)
  df[grep(decoyprefix, df[["ProteinName"]], invert=TRUE, fixed=TRUE),, drop=FALSE]


remove_decoy_loose <- function(df)
  df[!df[["decoy"]],, drop = FALSE]


basename_sans_ext <- function(f)
  unlist(strsplit(basename(f), ".",fixed=TRUE))[[1]]
```

```
main <- function(diafile,
                 strict_checking=FALSE,
                 peptideoutputfile="",
                 proteinoutputfile="",
                 decoyprefix="DECOY_")
{
  remove_decoy <- `if`(strict_checking,
                       function(df) remove_decoy_strict(df,decoyprefix),
                       remove_decoy_loose)
  filtered_data <-
    data.table::fread(diafile,header=TRUE) |>
    data.frame(stringsAsFactors = FALSE) |>
    within(run_id <- basename(filename)) |>
    SWATH2stats::reduce_OpenSWATH_output() |>
    remove_irt() |>
    remove_decoy()

  # Writing output
  filtered_data |>
    SWATH2stats::write_matrix_peptides(filename = basename_sans_ext(peptideoutputfile)
    write.table(sep="\t",file=peptideoutputfile,row.names = FALSE)

  filtered_data |>
    SWATH2stats::write_matrix_proteins(filename = basename_sans_ext(proteinoutputfile)
    write.table(sep="\t",file=proteinoutputfile,row.names = FALSE)
}
```

Where the field "Decoy" is 1, thats a decoy generated by `OpenSwathDecoyGenerator`, rather than from the fastadataasee of `DecoyDatabase`

```
process swath2stats {
    publishDir "${params.outdir}/dia/"
    input:
    file dia_score from feature_alignment_ch

    output:
    file peptide_matrix
    file protein_matrix
    script:
    strict_checking=params.swath2stats_strict_checking
    peptide_matrix="DIA-peptide-matrix.tsv"
    protein_matrix="DIA-protein-matrix.tsv"

    """
    #!/usr/bin/env Rscript
    """ +
        '''
<<r-swath2stats>>
        ''' +
        """
    main("$dia_score", strict_checking = as.logical("$strict_checking"),
        peptideoutputfile="$peptide_matrix",
        proteinoutputfile="$protein_matrix",
        decoyprefix="DECOY_")

    """
}

// whether to exclude in final DIA matrices
// proteins of which
// any peptide can be a decoy.
// this is the default behaviour of original gladiator implementation.
// if set to false, just instead remove anything
// that has the "decoy" column set to false
params.swath2stats_strict_checking=true
```

Dependencies

```
("swath2stats"
 "r-minimal"
 "r-swath2stats")
```

The above as an R-script

```
#!/usr/bin/env Rscript

.libPaths("/opt/Rlibs/")
```

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install(version = "3.15",ask=FALSE,update=FALSE)
BiocManager::install("SWATH2stats", ask=FALSE,update=FALSE)
```

6 Configuration for backends

6.1 Defining the config template

Here we define what the registry is that hosts our images

```
docker.io/elolabfi/
```

This is the template that we fill with the container manager program to be used (%p), the gladiator container uri to be used (%G), and the pyprophet legacy container to be used (%P).

```
%p.enabled=true
process {
   container='%G'
   withName: 'DeepDIA.*' {
      container='%D'
   }

   // for nonlegacy pyprophet processes
   withName: 'pyprophet_.*' {
      container='%N'
   }

   // because this is the more specific rule
   // we apply it last, so that it overrides the 'pyprophet_.*' rule if
   // this rule also applies
    withName: pyprophet_legacy {
container='%P'
   }
   // for perl-diamspep
   withName: 'DIAMS2PEP_.*'{
   // there is really no good single letter here
    container='%V'
   }
}
```

And here we write the logic that fills the template

```
(defvar container-property-alist
  (let ((remote-registry registry))
    `(("remote"
       ("singularity"
        ("access-protocol" . "docker://")
        ("registry" . ,remote-registry)
        ("container-suffix" . ":0.1.4-0-f7abd8"))
       ("docker"
        ("access-protocol" . "")
        ("registry" . ,remote-registry)
        ("container-suffix" . ":0.1.4-0-f7abd8"))
       ("podman"
        ("access-protocol" . "docker://")
        ("registry" . ,remote-registry)
        ("container-suffix" . ":0.1.4-0-f7abd8")))
      ("local"
       ("singularity"
        ("access-protocol" . "file://")
        ("registry" . "containers/")
        ("container-suffix" . ".simg"))
       ("docker"
        ("access-protocol" . "")
        ("registry" . "localhost/")
        ("container-suffix" . ""))
       ("podman"
        ("access-protocol" . "")
        ("registry" . "localhost/")
        ("container-suffix" . ""))))))

(defun construct-container-uri (basename program-name locality)
  (let ((properties
         (alist-get
          program-name
          (alist-get locality container-property-alist nil nil 'equal)
          nil nil 'equal)))
    (unless properties
      (error "Could not find properties for '%s' '%s'" program-name locality))
    (format
     "%s%s%s%s"
     (alist-get "access-protocol" properties nil nil 'equal)
     (alist-get "registry" properties  nil nil 'equal)
     basename
     (alist-get "container-suffix" properties nil nil 'equal))))
```

```
(cl-defun construct-container-config (template program-name locality &key (gladiator-in
  (let ((program-name-key
         program-name))
    (format-spec template
               `((?p . ,program-name)
                 (?G .
                     ,(construct-container-uri gladiator-image
                                                program-name-key
                                                locality))
                 (?D .
                     ,(construct-container-uri "deepdia"
                                                program-name-key
                                                locality))
                 (?N .
                     ,(construct-container-uri "pyprophet"
                                                program-name-key
                                                locality))
                 (?V .
                     ,(construct-container-uri
                       "diams2pep"
                       program-name-key
                       locality))

                 (?P .
                     ,(construct-container-uri
                       "pyprophet-legacy"
                       program-name-key
                       locality))))))
```

```
(construct-container-config template program-name locality :gladiator-image gladiator-i
```

```
docker://registry.gitlab.utu.fi/elixirdianf/gladiator-notes/test
```

```
https://registry.gitlab.utu.fi/elixirdianf/gladiator-notes/gladiator
```

6.2 Per backend config files

So here we fill the templates that we defined in the previous section. If you are reading this as an org-mode file, this headings default tangle argument makes each blocks tangle output file based on the name of the block

This block is whats common between singularity

```
singularity.runOptions = '-B $TMPDIR:/tmp'
singularity.autoMounts=true
```

This block is whats common between singularity and podman. On certain machines (reported on Ubuntu 22.04.5 LTS, 6.2.0-26-generic, with podman 3.4.4)

matplotlib in pyprophet in podman tries to read the home directory. which does not exist in the container, which will result in the error described here.

```
process {
    withName: 'pyprophet_.*' {
        containerOptions= '--env HOME="$PWD"'
    }
}
```

```
nil
singularity.runOptions = '-B $TMPDIR:/tmp'
singularity.autoMounts=true
```

#+end_src

```
<<construct-container-config("singularity","local")>>
<<singularity-general-options>>
```

```
nil
singularity.runOptions = '-B $TMPDIR:/tmp'
singularity.autoMounts=true
```

```
<<construct-container-config("docker","remote")>>
<<dockerlike-general-options>>
```

```
<<construct-container-config("docker","local")>>
<<dockerlike-general-options>>
```

```
<<construct-container-config("podman","remote")>>
<<dockerlike-general-options>>
```

```
<<construct-container-config("podman","local")>>
<<dockerlike-general-options>>

process {
   beforeScript="source <(guix time-machine -C $projectDir/ci/guix/gladiator-guix-chann
   withName: "DeepDIA.*" {
      beforeScript="source <(guix time-machine -C $projectDir/ci/guix/deepdia-channels
   }

   // for nonlegacy pyprophet processes
   withName: "pyprophet_.*" {
      beforeScript="source <(guix time-machine -C $projectDir/ci/guix/gladiator-guix-ch
   }

   // because this is the more specific rule
   // we apply it last, so that it overrides the "pyprophet_.*" rule if
   // this rule also applies
    withName: pyprophet_legacy {
        beforeScript="source <(guix time-machine -C $projectDir/ci/guix/pyprophet-legac
   }
   // for perl-diamspep
   withName: "DIAMS2PEP_.*"{
   // there is really no good single letter here
        beforeScript="source <(guix time-machine -C $projectDir/ci/guix/diams2pep-chann
   }

   withName: "DIAtracer" {
     beforeScript="source <(guix time-machine -C $projectDir/ci/guix/diatracer-channels
   }
}
```

6.3 Backend Specific Issues

6.3.1 Podman specific Issues

If you get the error in any process

> Error: OCI runtime error: the requested cgroup controller ‘cpu‘
> is not available

This is because nextflow sets the number of CPU’s podman can use, but you do not have the cpu cgroup in podman.

If, when you do

```
podman info  --format={{".Host.CgroupControllers"}}
```

you do not see cpu listed, but when you do

```
podman --cgroup-manager cgroupfs info  --format={{".Host.CgroupControllers"}}
```

you do see cpu listed, you will want to add to your nextflow config file:

```
podman.engineOptions="--cgroup-manager cgroupfs"
```

7 SDRF support

You can supply a path to an sdrf file with `--sdrf=filename`

```
// path to a sdrf file
params.sdrf = null
```

The standard is defined here `https://github.com/bigbio/proteomics-metadata-standard` See the `annoted-projects` directory for examples, Good examples is `PXD003977`

```
import nextflow.splitter.CsvSplitter

def readSDRF(filename,params)
{
    def sdrf_file=file(filename)
    def options=[:]
    def errors=[]
    def warnings=[]
    def handlers =[
        this.&parseSDRFFileUri,
        {_sdrf_fields,_params ->
            parseSDRFTolerance(_sdrf_fields, _params,
                                                    field_name = "comment[fragment mas
        {_sdrf_fields,_params ->
            parseSDRFTolerance(_sdrf_fields, _params,
                                                    field_name = "comment[precursor ma
    ]
    def sdrf_fields = new CsvSplitter().target(sdrf_file.text).options(header:true,sep
    sdrf_fields = sdrf_fields.collect(
        {entry ->
```

```
            entry.collectEntries(
                {key, row ->
                    [key,
                     (row =="not available" ||
                      row == "not applicable") ?
                     null :
                     row]})})


    for(handler in handlers) {
        def val = handler(sdrf_fields, params);
        errors += val.errors
        warnings+= val.warnings
        options+=val.values
    }
    for(warning in warnings)
        print("WARNING: " + warnings)
    for(error in errors)
        print("ERROR: " + error)
    if(errors)
        throw new Exception("Errors in SDRF file, see above messages")
    return options
}
```

Fields we handle:

7.1 comment[file uri]

We can handle remote and local files in comment[associated file uri] as long as they are in mzML format. This can be used i.l.o params.diafiles (though other libre formats would also be possible if we add an msconvert step first)

```
this.&parseSDRFFileUri,

def parseSDRFFileUri(sdrf_fields, params){
    def raise_error_on_non_mzml=true
    def retval= [values:[:],warnings:[],errors:[]]
    def field_name = 'comment[file uri]'
    def flag_name = '--diafiles'
    def files = sdrf_fields*.get(field_name)
    // here we intentionally don't break the switch statement so that we can accumulat
    // so that the user can know all at once.
    switch (true) {
        case (params.diafiles):
            retval.values+=[diafiles:params.diafiles]
            break;
    case(files == null):
            retval.errors += ["No column named '$field_name' in sdrf. Add one or suppl
            break;
        case (isallnull(files) || !files):
            retval.errors += ["All retval.values are missing in  supplied in sdrf '$fi
        case (!isallnull(files) && issomenull(files)):
            retval.errors += ["Some entries in sdrf '$field_name'  are not given, fill
        case (!isallnull(files) && raise_error_on_non_mzml && files.any({x -> x  && fil
            retval.errors += ["Some entries in sdrf  '$field_name' are not mzML files"]
            // if there were no retval.errors
            // then the files field in the sdrf is correct
        case(!retval.errors):
            retval.values+=[diafiles:files]
            break;
        case(retval.errors):
            retval.values+=[diafiles:null]
            break;
    }
    return retval
}
```

7.2 comment[precursor mass tolerance], comment[fragment mass tolerance]

We require (for now) precursor mass tolerance to be ppm, and to be the same for all the samples.

```
{_sdrf_fields,_params ->
    parseSDRFTolerance(_sdrf_fields, _params,
                                            field_name = "comment[fragment mass tolera
{_sdrf_fields,_params ->
    parseSDRFTolerance(_sdrf_fields, _params,
                                            field_name = "comment[precursor mass tolera

def parseSDRFTolerance(sdrf_fields, params,  field_name = "comment[precursor mass toler
    def retval = [values:[:], warnings:[], errors:[]]
    def tolerances = sdrf_fields*.get(field_name)
    switch(true){
        case(params.get(flag_name) != null):
            retval.values[flag_name] = params.get(field_name);
            break
        case(tolerances == null):
            retval.errors += ["No column named '$field_name' in sdrf. Add one or supply
            break;
        case (isallnull(tolerances) || !tolerances):
            retval.errors += ["All values are missing in  supplied in sdrf '$field_nam
        case (!isallnull(tolerances) && !tolerances.every()):
            retval.errors += ["Some entries in sdrf '$field_name'  are not given, fill
        case(tolerances.any({it && it.split().size() > 1 && !(supported_units.contains
            retval.errors += ["Some entries in sdrf '$field_name' have an unsupported u
        case(tolerances.any({it && it.split().size() < 2})):
            retval.warnings += ["Some entries in sdrd '$field_name' do not have a unit
        case(tolerances.unique().size() != 1):
            retval.errors += ["Gladiator currently requires the $field_name to be the s
    case(!retval.errors):
            retval.values[flag_name] = tolerances.unique()[0].split()[0]
    }
    return retval
}
```

7.3 Utility functions

```
def isallnull(value)
{
    value == null || value.every({x-> x==null})
}
def issomenull(value)
{
    value == null || value.any({x -> x == null})
}
```

7.4 Testing

```
print(readSDRF("annotated-projects/PXD003977/PXD003977.sdrf.tsv",params))
```

7.5 Updating params

```
if (params.sdrf){
    params = readSDRF(params.sdrf, params) + params
}
```

8 Putting it together

8.1 gwl

```
(make-workflow
 (name "my-workflow")
 (processes
  (auto-connect
   (list
    <<gwl-proc>>
    ))))
```

8.2 nf

according to `nextflow/docs/process.rst`, the cache looks at the timestamp for cache-correctness, which can be inconsistent in on shared file systems, setting `process.cache` to 'lenient' maybe will work around this? See alos

> "'lenient"' Enable caching. Cache keys are created indexing input files path and size attributes (this policy provides a workaround for incorrect caching invalidation observed on shared file systems due to inconsistent files timestamps; requires version 0.32.x or later).

there is also 'deep' hashing, based on file content and the undocumented 'sha256' hashing bashed on the shasum of the file See also here: `https://www.nextflow.io/blog/2019/troubleshooting-nextflow-resume.html`

```
process.cache='lenient'
```

though this will break if you replace input files with files with the same path but different content. 'sha256' might be the best one?

The results files will all be output in the following directory

```
// directory where the  results will be output to
params.outdir = "./results"
```

```
<<nf-params>>
```

9 Troubleshooting

- Podman specific Issues
- KeyError: 'HOME' in `pyprophet_subsample`