NAME

phosfox − find peptide−level differences in PTMs

SYNOPSIS

```
phosfox job−file.json
```

DESCRIPTION

PhosFox performs a peptide-level protein post-translational modification (PTM) comparison between case and control settings. It detects which peptides have been *uniquely modified* (for example, uniquely phosphorylated) in case and control samples, and writes the differences into HTML-formatted report files. Peptides and proteins can be also written into separate files for further analyses. Both qualitative and quantitative data sets can be analyzed. Peptide lists originating from different search engines can be also compared.

If you are not familiar with the concepts and the process, please read Söderholm et al. first.

The program is controlled by a *job file*, which is the sole, and mandatory, command line argument. See ''JOB FILE'' for details.

JOB FILE

A job file is a plain text file in JSON (<http://www.json.org>) format that describes the experimental set-up and the parameters for the program. Here is an example job file with all possible parameters:

```
{
    "fasta_files": [ "HUMAN.fasta", "rat.protein.faa" ],
    "database_files": [ "swissprot.tsv", "phosphosite.tsv" ],
    "acids": "STYK",
    "modifications": [ "acetyl", "phospho" ],
    "analysis_name": "My job",
    "report_prefix": "my−report",
    "output_prefix": "my−prefix",
    "log_file": "my−log.txt",
    "verbose": true,

    "inputs": [
        {
            "filename": "paragon−cases.tsv",
            "type": "case",
            "search_engine": "Paragon"
        },
        {
            "filename": "mascot−cases.csv",
            "type": "case",
            "search_engine": "Mascot"
        },
        {
            "filename": "paragon−controls.tsv",
            "type": "control",
            "search_engine": "Paragon"
        },
        {
            "filename": "mascot−controls.csv",
            "type": "control",
            "search_engine": "Mascot"
        },
        {
            "filename": "sequest.txt",
            "type": "quantified",
```

```
                "search_engine": "SEQUEST"
            }
        ],

        "cutoffs": {
            "Conf": 80,
            "IonScore": 80,
            "pep_score": 0.8
        },

        "quantification_cutoffs": [
            {
                "labels": "115/113",
                "low": 0.5,
                "high": 2.0
            },
            {
                "labels": "116/114",
                "low": 0.75,
                "high": 1.25
            }
        ]
    }
```

Job parameters can be divided into four groups: generic parameters, output parameters, input files, and cut-offs. The parameters are optional with the exception of `fasta_files`, `inputs`, and `report_prefix`.

Generic parameters

`fasta_files`

A list of one or more FASTA files containing the protein sequences. The peptides are matched against the sequences in these files to determine the exact modification sites.

SwissProt and NCBI GenInfo FASTA files are accepted.

Filenames can be specified with full, absolute paths, or with relative paths, in which case they are sought relative to the current working directory in which PhosFox is executed.

`database_files`

(Optional.) Known PTM database files in plain text format. If given, novel modification sites will be detected. ‘‘Novel’’ means that the correspoding site has not been reported in the scientific literature according to the given database files.

See the *DATABASES.md* file in the PhosFox distribution for details about these files. Filenames can be specified with full, absolute paths, or with relative paths, in which case they are sought relative to the current working directory in which PhosFox is executed.

If this parameter is given, protein identifiers in the input files are expected to have UniProt accessions.

`acids`

(Optional.) A string of amino acids that are checked for modification events. The default is `STY` (serines, threonines, and tyrosines).

`modifications`

(Optional.) A list of post-translational modifications that are examined. Possible values are

- `phospho`: Phosphorylation. (This is the default.)
- `acetyl`: Acetylation.

Each modification type will generate a separate report file. See `report_prefix`.

Output parameters

`analysis_name`

(Optional.) Analysis name. This is used as the title in the produced report.

`report_prefix`

A prefix for report file names. Produced reports will be named as *report_prefix−modification.html* for each modification in `modifications`. See ‘‘REPORT FILES’’ for details.

`output_prefix`

(Optional.) A prefix for case and control peptide and protein list file names. If given, modified proteins and peptides in case and control samples will be writted into four separate files, with one such set for each search engine. These files can be used to compare the protein/peptide identifications between search engines. See ‘‘PROTEIN AND PEPTIDE FILES’’ for details.

`log_file`

(Optional.) If given, the logging output will be written into this file. The default is to print the log to standard error.

`verbose`

(Optional.) Verbosity flag (`true` or `false`). If true, the log will contain more information. The default is false: only warnings and errors will be logged.

Input files

`inputs`

A list of JSON objects. Each object specifies a single input file (peptide list). The object must have the following elements:

- `filename`: The name of the peptide list file. This file must be either Mascot-like CSV file, or tab-separated tabular file. See ‘‘INPUT FORMATS’’.
- `type`: The sample type from which the input peptides were identified. This must be one of `case`, `control`, or `quantified`. Quantified peptides are interpreted as case or control peptides depending on quantification cut-offs.
- `search_engine`: The search engine that was used to detect the peptides in the input file. Each search engine gets its own case and control columns in the output report. Note that `search_engine` is case-sensitive (eg. `Mascot` and `MASCOT` are different).

Cut-offs

`cutoffs`

(Optional.) ‘‘Generic’’ cut-offs used for filtering out peptides. These are usually quality-control thresholds. Cut-offs must be specified in a single JSON object, whose elements are interpreted as *value name*/*threshold* pairs. Examples:

```
"Conf": 80
```

Filter out peptides with `Conf` value less than 80. (This is a typical ProteinPilot QC field.)

```
"pep_score": 0.8
```

Filter out peptides with `pep_score` value less than 0.8. (This is a typical Mascot QC field.)

The cut-offs are global in the sense that they are applied to all input files. It is not possible to specify separate cut-offs for each input file. See also ‘‘INPUT FORMATS’’.

`quantification_cutoffs`

(Optional.) List of JSON objects defining quantification labels and cut-offs. These cut-offs, or thresholds, are used to divide quantified peptides into cases and controls. Each object defines one field (label pair) and two limits:

- `labels`: A pair of quantification labels. Labels must be of form `X/Y`, where `X` refers to case label, and `Y` refers to control label.

- `low`: If the `X/Y` ratio is *below* this threshold, the corresponding peptide is interpreted as control peptide.
- `high`: If the `X/Y` ratio is *above* this threshold, the corresponding peptide is interpreted as case peptide.

Example:

```
"quantification_cutoffs": [
    {
        "labels": "115/113",
        "low": 0.5,
        "high": 2.0
    },
    {
        "labels": "116/114",
        "low": 0.75,
        "high": 1.25
    }
]
```

In this example, case samples have been labeled with 115 and 116, and control samples have been labeled with 113 and 114. For each peptide, the following comparisons are done:

- If 115/113 < 0.5, the peptide is interpreted as control peptide.
- If 115/113 > 2.0, the peptide is interpreted as case peptide.
- If 0.5 <= 115/113 <= 2.0, the peptide is interpreted as both case and control peptide.
- If 116/114 < 0.75, the peptide is interpreted as control peptide.
- If 116/114 > 1.25, the peptide is interpreted as case peptide.
- If 0.75 <= 116/114 <= 1.25, the peptide is interpreted as both case and control peptide.

If you do not want to mix label pairs (because they represent separate experimental settings, for example), use only one label pair at a time.

INPUT FORMATS

Peptide lists for *PhosFox* must be tabular, plain text files. Many proteomics analysis platforms can export identified peptides into such files. *PhosFox* expects to find certain fields (columns) from the input files. Specifically, the program recognizes the following formats:

- Comma-separated (CSV) files, similar to ones produced by *Mascot* software from Matrix Science. They have one peptide per line, and the data values are separated by commas.

  The file *must* have a header line with fields `prot_hit_num` (protein hit number), `prot_acc` (accession), `pep_seq` (peptide sequence), and `pep_var_mod_pos` (peptide modifications). These fields can be in any order.

  Everything before the header line is ignored, and everything after the header line is assumed to be peptide data.

- Tab-separated (TSV) files. These are simple tabular files: one line per peptide, and the data values separated by tab characters.

  As with the CSV files, the file *must* have a header line with fields `Accessions` (*or* `Protein Group Accessions`), `Sequence`, and `Modifications`. These fields can be in any order. The header line must be the first non-empty line in the file that has at least three tab-sebarated values (fields).

The following post-translational modification strings are recognized:

- *MODIFICATION*(*ACID*)@*OFFSET*, eg. Phospho(S)@4
- *ACID OFFSET*(*MODIFICATION*), eg. T2(Phospho)
- ‘‘Mascot-notation’’, eg. 0.0000200.0. *PhosFox* tries to parse modification identifiers from the header data.

If you specify ‘‘generic’’ cut-offs (see ‘‘Cut-offs’’), the header should contain the corresponding fields. For example, if you specify a cut-off for Conf value, the header should contain Conf field. It is not an error if there is no matching field; in that case the cut-off is ignored for that input file.

REPORT FILES

The report files are standard HTML files that can be opened in any web browser. They also work with many spreadsheet programs, such as *LibreOffice Calc* and *Microsoft Excel*. A separate report file will be written for each modification given in modifications in the job file. A report file contains a table of all identified peptides that fulfill the speficied cut-offs, and have at least one modified amino acid among the acids parameter. Peptides that share exactly the same modification sites are grouped together as a single peptide.

For example, if phosphorylations and acetylations are examined, two report files will be produced. The other one will contain phosphorylated peptides, and the another one will contain acetylated peptides. Note that the same peptide can have both acetylations and phosphorylations, and hence appear in both report files.

The report table contains the following columns:

- Protein id: protein accession (UniProt or GenInfo).
- Protein name: short name or RefSeq locus.
- Protein description: longer description.
- Peptide sequence: Matched peptide sequence. Modification sites are underlined. If uniprot_file parameter is given, modification sites are checked for ‘‘novelty’’ (see ‘‘Generic parameters’’). Novel sites are additionally typed in bold.
- Modification sites: Locations of the modifications in the corresponding protein sequence. Location 1 corresponds to the first amino acid of the protein sequence. Bold typeface indicates novel modification sites.
- Cases: A set of counts, one for each specified search engine, indicating how many times the corresponding peptide was identified by the search engine in the case sample.
- Controls: A set of counts, one for each specified search engine, indicating how many times the corresponding peptide was identified by the search engine in the control sample.
- Unique case: If 1, the peptide is unique to the case sample, otherwise 0. Unique case rows are colored in red.
- Unique control: If 1, the peptide is unique to the control sample, otherwise 0. Unique control rows are colored in blue.
- Acid(s): The number of modified acid(s) in the peptide for each amino acid given in acids in the job file.

PROTEIN AND PEPTIDE FILES

If output_prefix parameter is specified in the job file (see ‘‘Output parameters’’), *PhosFox* writes the following four files for each modification and search engine combination:

- *output_prefix−modification−search_engine−case−peptides*
- *output_prefix−modification−search_engine−control−peptides*
- *output_prefix−modification−search_engine−case−proteins*
- *output_prefix−modification−search_engine−control−proteins*

modification and *search_engine* are replaced with the corresponding modification type and search engine

name.

Files with *−control−proteins* suffix contain identifiers, one per line, of *uniquely modified proteins* identified from the control sample by the correspoding engine. Similarly, files with *−case−proteins* suffix contain uniquely modified proteins identified from the case sample.

Files with *−peptides* suffix contain modified peptides identified from the corresponding sample by the corresponding engine. All modified peptides that satisfy the cut-offs (if any) are included regardless of whether they are uniquely modified or not. There is one line for each peptide. Peptides that share exactly the same modification sites are grouped together as a single peptide, just like in the report file. Each line looks like this:

```
Q9BQ69 154 155S 162T 164
1      2   3    4    5
```

In this example there are five fields (1−5 above):

1: Accession of the corresponding protein.
2: The start position of the peptide in the corresponding protein sequence. Location 1 corresponds to the first amino acid of the protein sequence.
3 and 4: Modification sites, the letter denotes the modified amino acid.
5: The end position of the peptide in the corresponding protein sequence.

DIAGNOSTICS

PhosFox logs may include warnings and errors encountered during processing. By default, these messages are printed to standard error. Each message is prefixed by the program name and time stamp of the event. Possible warning and error messages are listed below.

cannot open any peptide list
: None of the input files specified in the job file could be opened and/or recognized as a valid peptide list. Check the file names and formats (see ‘‘INPUT FORMATS’’).

cannot open job file ’*filename*’
: The supplied job file *filename* cannot be opened for reading. Often the error message includes the cause as well.

cannot open log file ’*filename*’
: The specified log file *filename* cannot be opened for writing. Often the error message includes the cause as well.

cannot open database file *filename*: ...
: The specified database file cannot be opened for reading.

cannot open output file ’*filename*’: ...
: The specified file cannot be opened for writing.

cannot parse job file, stopped at line: ...
: The job file is not proper JSON; there is a syntax error. The error is hopefully close to the line that is printed in the message.

cannot recognize ’*filename*’ as valid peptide list
: The given input file *filename* is not in supported format. See ‘‘INPUT FORMATS’’.

invalid modification line *N* in file *filename*
: The specified modification is not properly formatted in the specified file. See the *DATABASES* file for details on the database file formats.

invalid quantification cutoff labels ...
: Some quantification cut-off has an invalid label pair (*labels* attribute). Labels must be of form `X/Y`, where `X` and `Y` are case and control labels.

malformed job file ’*filename*’
: The supplied job file *filename* is most likely not (even closely) valid JSON. Check that you have specified the correct job file and check its formatting.

missing input type
: Some input file does not have type attribute (`case`, `control`, or `quantified`).

missing quantification cutoff labels
: Some quantification cut-off does not have *labels* attribute.

missing quantification cutoff low limit
: Some quantification cut-off does not have *low* attribute.

missing quantification cutoff high limit
: Some quantification cut-off does not have *high* attribute.

multiple (*N*) matches for *sequence* at *location*
: The peptide sequence *sequence* matched *N* times ($N > 1$) to the corresponding protein sequence. Only the first match is used. The reported modification sites are not necessary correct for such peptides.

no match for *sequence* at *location*, skipping
: The peptide sequence *sequence* does not match to the corresponding protein sequence. Check that you have specified a correct FASTA file in the job file.

no sequence for *id* at *location*, skipping
: The protein sequence for protein with identifier *id* does not exist. Check that you have specified a correct FASTA file in the job file.

sequence cannot be determined to be in either case or control sample at *location*, skipping
: The peptide sequence *sequence* cannot be categorized as case nor control peptide due to missing quantification cut-off or invalid quantification data. (This occurs only with quantified inputs.)

unknown input type '*type*' was given
: Some input file does not have a correct *type* attribute (`case`, `control`, or `quantified`).

unknown modifications *mods* given
: Post-translational modifications are incorrectly specified in the job file. See ''JOB FILE''.

CAVEATS

All existing files are silently overwritten, if their names happen to clash with the output file names (see ''Output parameters''). Be particularly careful with the `output_prefix` and `report_prefix` parameters.

Peptides with accessions matching CON (contaminated) or REV (reverse) are silently discarded.

BUGS

When looking for novel modification sites, peptides that do not have UniProt accessions, or otherwise do not have a matching identifier in the supplied database files are considered ''novel''.

RESTRICTIONS

Only GenBank and SwissProt FASTA files can be used.

AUTHOR

Petteri Hintsanen <petterih@iki.fi>.

COPYRIGHT AND LICENSE

SEE ALSO

Sandra Söderholm, Petteri Hintsanen, Tiina Öhman, Tero Aittokallio, Tuula A. Nyman: PhosFox: a bioinformatics tool for peptide-level processing of LC−MS/MS−based phosphoproteomic data. Proteome Science 2014, 12:36, doi:10.1186/1477−5956−12−36.

Web site: <http://bitbucket.org/phintsan/phosfox>