Documentation
Version 1 - February 2019

PHOSPHOKINASE

A Comprehensive Kinase search platform covering protein kinases, associated inhibitors, and phosphosites.

Produced by Bayliss:
Hajar, Friha, Omer and Gabriel.

About

With increasing research into the Kinome and in particular kinase signalling networks, the need for a comprehensive Kinase platform integrating Kinase data is in demand. Phosphokinase is a fast and user-friendly interface that allows the user to search Kinases, associated Inhibitors, and Kinase phosphorylation sites. Phosphokinase was developed by Bayliss – a group of 4 students that are part of the MSc Bioinformatics Programme at Queen Mary University of London, under the supervision of Professor Conrad Bessant and Dr Fabrizio Smeraldi.

All search features of the Phosphokinase website are linked to a curated database with information on Kinases (names, aliases, location and classification), Inhibitor information (names, structural properties and representation), and all known phosphorylation site for each kinase (including substrate location and site of modified residue).

A key interactable feature of Phosphokinase is the two-step 'PP Tool', this allows the user to upload Phosphoproteomic data, define threshold parameters, and then visualise the analysis of the data. The output is divided into three components; a total summary represented as a volcano plot of phosphorylation sites and the Kinases known to phosphorylate at the given modified residue, a filtered Kinase only volcano plot and relative kinase activity table that can be downloaded for future use.

Contents

Software Schematics

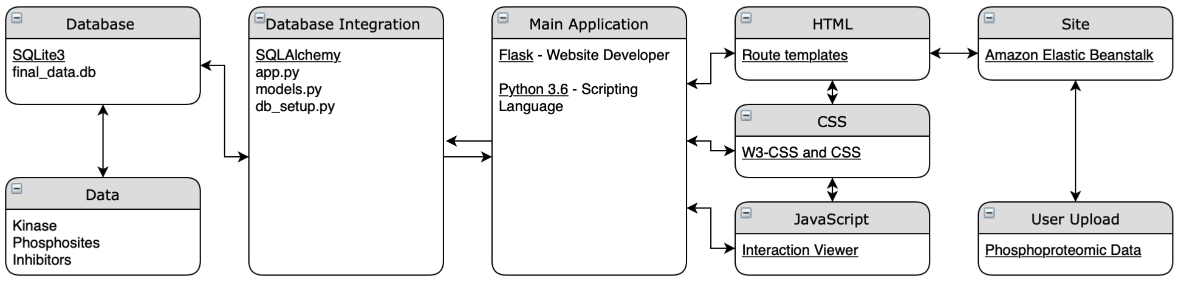

Figure 1: Phosphokinase software schematics and component integration.

Figure 1 shows the main components and their integration to form the final functional software. The database holding the information on kinases, phosphosites, and associated inhibitors was made using SQLite3. The database was connected to Flask using SQLAlchemy – a python toolkit and Object relational mapper that is designed for efficient database access that considers relationships in the database, not just a collection of tables. The main application is run on Flask using Jinja2 as template engine, Python 3.6 and integrates data from SQL using Flask-SQLAlchemy (main.py is the executable file). Website routes are defined using the HTML language and static files such as CSS (maintain consistency in website design) and javascript files (.js) are interlinked with the main application. Together the application renders the final site that also features a user upload to feed data back into the software.

Website Architecture

Figure 2 represents a concise summary of the various routes within the website and how they are linked. The home page has direct routes to three search engines in addition to a phosphoproteomic data analysis tool (PP Tool). Each search tool outputs a page with information allocated into different tabs, these are shown by the dotted lines. Internal links that allow for interactivity between site pages are also shown.

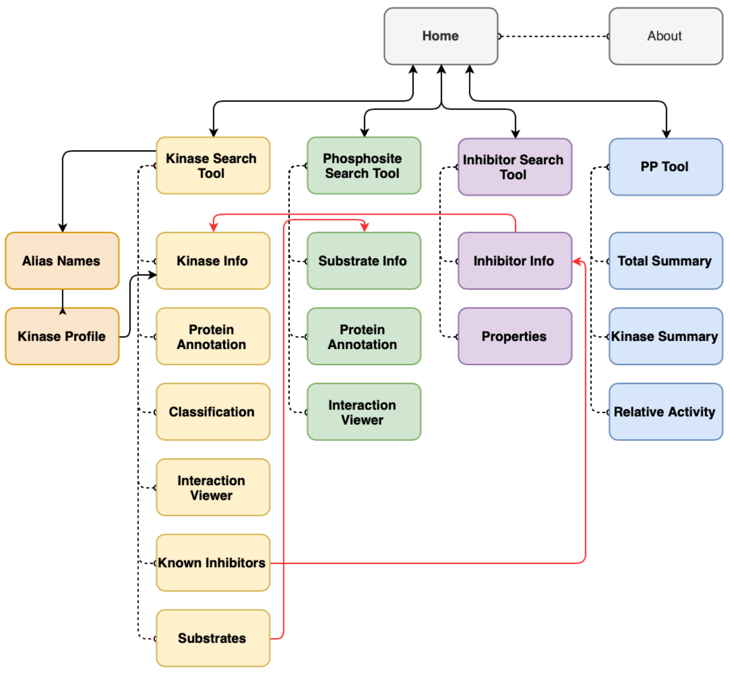

Figure 2: Phosphokinase web routes and links shown by solid black arrows. Web content shown by dotted lines. Internal hyperlinks shown in red.

Running Phosphokinase

Phosphokinase has been successfully deployed and can be used on: http://bayliss.eu-west-2.elasticbeanstalk.com.

Running Phosphokinase locally:

In order to run phosphokinase, download the final_website directory from the github page. Phosphokinase can then run locally using:

```
$ git clone https://github.com/startswithH/teamBayliss
$ cd final_website

$ pip install Flask
$ FLASK_APP=main.py
$ flask run

 * Running on http://localhost:5000/
```

Packages Used:

Flask: Flask is a microframework for Python based on Werkzeug and Jinja 2 templates. Flask was used because of its easy integration with python, and because of the simplicity and straightforwardness of the application making it easy to learn for beginners as well as set up and run. Flask also runs on python 2.7 and 3.6 and so was able to handle the script and code the website uses. In addition, Flask is based on Jinja 2 templating (Jinja2 works with Python 2.6.x, 2.7.x and >= 3.3) which shares a language similar to python, this made it easy to integrate and run code on the main application as well as on the HTML pages.

The libraries used in Flask include: sqlalchemy, flask-WTForms, flask_table, pandas, ipython, requests, numpy, matplotlib, bokeh and werkzeug.utils.

JavaScript: Phosphokinase runs opensource JavaScript libraries to provide an interactive experience for the user, as well as providing another platform for more detailed information about protein kinases. More information on what these .js files include and how they run is outlined later in the 'Phosphokinase Features' section. Libraries were searched for using the BioJs site – a library of more than a hundred javascript components for the field of Biology and Bioinformatics. The two libraries used are: ProtVista – Protein Annotation[1] and Interaction Viewer (https://github.com/ebi-uniprot/interaction-viewer).

Structure and Design: W3-CSS/CSS/HTML

HTML tags were used to define the components of each page. HTML was coupled to CSS languages to deliver a professional website. W3-CSS is a CSS framework that also contains built in responsiveness. This was used as it is smaller and faster than CSS and with a few commands produced a clean result. This meant there was no need to manually update CSS. In some cases, CSS was used to generate professional search fields, and to work with the JavaScript and other interactive elements.

Data Collection

Kinase Data

Initial data for kinases was obtained via the Kinase website (http://kinase.com/web/current/). This information included general information about the kinase. General information on the names of the discovered human kinases were found on a specific section of the website (http://kinase.com/web/current/human/). The data file used in order to collect the kinase information was the S1 file – available in an excel and tab-delimited format. However, some specific information (i.e. location) was missing and was thus gathered using Uniprot API (https://www.uniprot.org/). The kinase data used to make up the database was curated, and any pseudogenes were removed.

Phosphosite Data

All phosphorylation data was obtained from the PhosphoSitePlus[3] website (https://www.phosphosite.org/homeAction.action). The data used was found on the download section of the website (https://www.phosphosite.org/staticDownloads). Two datasets were downloaded and used, these were the 'Kinase_Substrate_Dataset.gz' and 'Phosphorylation_site_dataset.gz'. However, only data that showed kinase-kinase and protein-kinase phosphorylation relationships were obtained. The data was sorted to contain the relevant information such as, gene name, kinase, kinase accession, substrate, substrate accession, sub-gene, and the sites of phosphorylation. This data was primarily used for the phosphoproteomic data analysis section. This information also allows for the user to search for a specific substrate which is known to be phosphorylated by a kinase.

Inhibitor Data

The inhibitor data for each kinase was gathered from the Published Kinase Inhibitor Set (PKIS) which is a collection of 367 compounds[2] available by GSK (https://www.ebi.ac.uk/chembldb/extra/PKIS/). These compounds have been published in literature and evidence surrounding kinase inhibition is also provided. The data contained information on each of the inhibitor such as ChEMBL ID, structural representations (INCHI and SMILES) as well as more refined properties (e.g. molecular weight and Lipinski scores). Rows with repeated ChEMBL Ds were removed and target names of the inhibitor were changed to match the kinase names on the 'kinase_information' table.

Database Schema

SQLite3 was utilised due its simplicity in database construction and data retrieval via SQLAlchemy-Flask. Separate tables were created for each part of the database. The sorted information was stored as separate CSV files which were read and added into tables identified by their column names. The tables were joined using the kinase and inhibitor names (as seen in Figure 3) to ensure the data is connected and allows fast browsing.

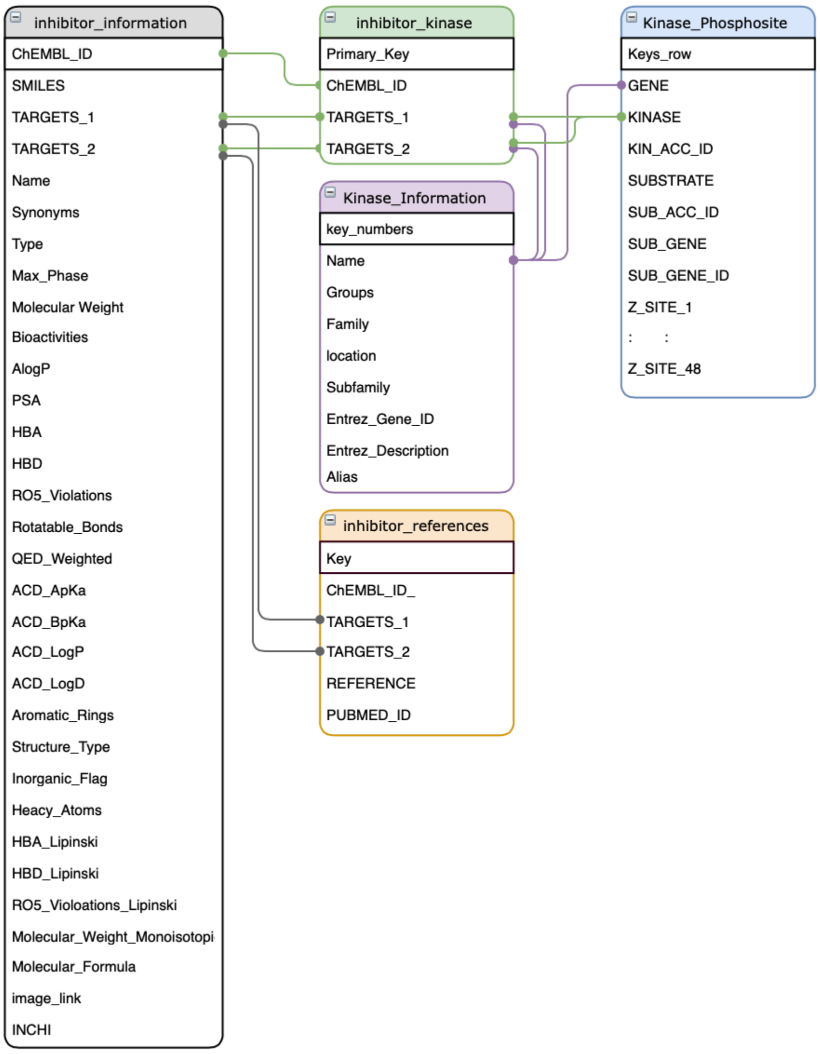

Figure 3 Phosphokinase database schematics: Shown is 5 tables and the links represented by solid lines.

SQLAlchemy was used to retrieve information from the database because of the ease of access when using python. Two types of queries were used to access the data.

1) Simple query: This involved searching the database given a search string from the user. This type of query forms the bases of all the search functions.

2) Join query: This was used to obtain information from other tables given a single search parameter value. This was used in the Kinase search to retrieve information about the Inhibitor Targets and kinase substrates.

Phosphokinase Features

Kinase search function

The kinase search engine allows the user to search for kinases based on three features; protein kinase name, alias name, and gene name search. The protein kinase and gene name search queries a search string against the 'kinase_information' table, the result is information on that specific kinase including the name, family, subcellular location, and alias names.
Alias search: In some cases, due to the development of different name standards the names used in the kinase search may not output any results hence the alias search feature. The alias search queries the alias names for each kinase and searches for similar matches, the result is an alias page that allows the user to select their desired kinase.

Inhibitor search function

Phosphokinase also features an inhibitor search engine that allows the user to find out more information about GSK inhibitors that are known to inhibit kinases. This includes information on structural properties, and structural representations. This feature is useful if the user already knows the ChEMBL ID or the GSK name.

Internal Hyperlinks

A useful feature of Phosphokinase is the interactivity of the websites and the smooth transition between different routes. The use of internal hyperlinks means that the user has the ability to navigate various components of the web with just a click. For example, after searching for a desired kinase the user can navigate through the inhibitors known to inhibit that kinase and select the desired inhibitor to find out more about it.

External Hyperlinks

In some cases, Phosphokinase may not provide the desired level of information about a particular kinase, or the user may be prompted to find out more about a kinase that they have searched into. Phosphokinase provides access to external websites for each kinase using either the Entrez ID to link to the NCBI website or the accession ID to access information from the Uniprot website.

Protein visualisation package and browsing Phosphosites

ProtVista is an invaluable tool that allows for the visualisation of a protein sequence as well as various features such as disease variants or sites of post translational modifications based on the Uniprot Knowledgebase, proteomics data, and datasets with information on disease variants. one of the main features of thie viewer that is especially useful for users interested in the Kinome is the 'PTM' section. This contains the kinase domain of the protein and also contains sites that are phosphorylated including position and residue. The hover feature allows the user to find out the exact kinase that phosphorylates the protein at that specific position.
ProtVista features in the phosphosite search element of the website in which the user can search for a substrate which is known to be phosphorylated by a kinase. The data presented is the name, gene name, genomic location and ProtVista. Using ProtVista the user can view all known phosphorylated sites of the substrate/protein.

Protein Interaction Viewer

Kinase phosphorylation forms part of a greater signalling network. In order to understand this network a key component is to assess the proteins the kinase interacts with. Protein interaction viewer searches for the accession ID of a given protein and outputs any binary interactions the protein has with other proteins derived from the EBI IntAct database. Another feature is the ability to filter the interactions based on subcellular location and/or by the disease involvement of a given protein. The hover feature allows the user to view more information on the binary interaction and also review the research and publications behind that interaction.

PP Tool: Uploading Phosphoproteomic data

This tool provides users with a tool to upload quantitative phosphoprotemics data to visualise the data and obtain estimations for the relative human kinase activity in the sample[5]. The user data file type accepted is the tsv file data. This should be presented with the columns arranged in the following order: Substrate, Control mean, inhibitor mean, fold change, p-value, control CV, inhibitor CV. If an incompatible file type is uploaded, an error message is displayed. The 'Substrate' Column accepts UniProtKB entry names and gene names.

Upon upload, the data is processed. The tool processes data from phosphorylation on serine, threonine and tyrosine residues. Data rows with "None" and displaying phosphorylation by methionine are parsed out since methionine residues near phosphorylation sites are oxidised and not phosphorylated[4]. Substrate labelled with "_HUMAN" are converted to their gene names using Uniprot API for standardisation. Those that were not converted by Uniprot API, are parsed out as they are likely to be experimental proteins that do not have any kinases matched to the phosphorylation sites in the dataset. Phosphorylation sites of substrates are then matched to the protein kinases that target these sites using data from the database table 'Kinase_Phosphosite' which is converted into a CSV file.

For visualisation, 2 volcano plots are generated, 'Total Protein Summary' shows all the data points. The second volcano plot, 'Filtered Kinase Summary' has only data points of substrate/phosphosite matched with their known kinases. The log transformation was performed on fold change and p-values where the volcano plots show $\log_2$ fold change (x-axis) plotted against $\log_{10}$ p-value.

The user can see the top 30 most significant kinases in the 'Relative Kinase Activity' output table sorted by ascending Fold Change of each kinase. This is useful as the table would feature the kinases most inhibited in the experiment. The full table of results can be downloaded as a CSV file. An explanation of the columns and calculations in the 'Relative Kinase Activity' output table is summarised below.

Total_Control_mean	The sum of mean phosphopeptide MS1 peak areas in the control (untreated) treatment, matched to a kinase.
Total_Control_mean	The sum of mean phosphopeptide MS1 peak areas in the control (untreated) treatment, matched to a kinase.
Total_Inhibitor_mean	The sum of mean phosphopeptide MS1 peak areas in the inhibitor treatment, matched to a kinase.
Relative_control_activity	In the control treatment, is estimated by the sum of the mean phosphopeptide MS1 peak area, matched to an individual kinase, divided by the sum of the mean phosphopeptide MS1 peak areas in the dataset for all kinase-substrate matches.
Relative_inhibitor_activity	In the inhibitor treatment, is estimated by the sum of the mean phosphopeptide MS1 peak area, matched to an individual kinase, divided by the sum of the mean phosphopeptide MS1 peak areas in the dataset for all kinase-substrate matches.
FC_kinase	Calculated by the Total_Inhibitor_mean divided by the Total_ Control_mean for each kinase.

PP Tool: Parametrisation

At the upload page, the user can enter the fold change, p-value and coefficient of variance and background noise thresholds to customise the display of the results. With fold change threshold parameter, the user can define values below which they consider to be inhibited. With p-values, user can define p-value threshold (usually 0.01 or 0.05). Values above the threshold will be coloured grey in the volcano plots and filtered out from the results table. Data with coefficient of variation values above the user set threshold are filtered out. With the background noise parameter, user can define values below which they consider to be noise or insignificant low-level phosphorylation.

Deploying Phosphokinase to the web:

Amazon Web Services (AWS) has many services in order to deploy an application, Elastic Beanstalk was the service used. It allows for the quick deployment of an application to the AWS Cloud without the need to learn about the infrastructure that is needed to run the application. Elastic Beanstalk handles the details of capacity provisioning, load balancing, scaling, and application health monitoring. Elastic Beanstalk has the advantage of supporting applications developed in many languages, in this case, the language used is Python.

In order to deploy the application, it must be first prepared. Elastic Beanstalk has many advantages, however, in order to deploy any application correctly you must ensure that you follow any and all instructions. The name of the app deployed was changed from main.py to application.py as the WSGI path automatically searches the path for an app named application.py. This is the default option, and you can change it by modifying the WSGIPath in the root of your app folder. In the application.py file, the WSGI will look for an application object, therefore, it is strongly recommended you assign your Flask app to application, with the following line have application assigned to app. This extra line of code is simple and further resolves the issue of having to rename any object to application within the file.

Once this is done, you will need to create a requirement.txt file, this is so the elastic beanstalk knows which packages to load into your virtual environment. This is a fairly straightforward process. First you must install the pipreqs package, do this with the following command "pip install pipreqs" which is to be executed in your terminal. With pipreqs installed, simply run the pipreqs command as followed "pipreqs /path/to/app" which will generate a requirements.txt file for you. In order to generate a requirements.txt file, you may also use pip freeze, however, this will save all packages in the environment including those that you don't use in your current project unless you are using a virtual environment. The usage of pipreqs allows for you to create a requirements text file based on imports just used for that particular project.

The file contents must now be compressed into a zip file. This zip is then simply uploaded to elastic beanstalk, which then does everything else for you and will deploy your application. Elastic Beanstalk is a resource which is on the face of it simple to use, however does currently suffer from limited documentation. Therefore, you may encounter many unique errors which will require a process of trial and error in order to successfully deploy your application.

Local Deployment

If you wish to locally deploy the application, you must first ensure that you have the correct imports. Therefore, using pipreqs, generate a requirements.txt file. Then in your terminal simply run "pip install -r /path/to/requirements.txt" which will import all the necessary libraries you need in to run the application locally. Having done this, now run the main application file using python, this will deploy the website locally.

Limitations

Web Feature limitations:

PP Tool: Time duration for results to return is dependent on speed of Uniprot API. During our testing, the time taken for results to return was between 0.5 to 8 minutes. Another limitation of this tool is the user upload feature, currently, the code only runs for the first 7 columns (or less) of the data given. This means, if the user uploads a dataset with more than one inhibitor data only the dataset for the first inhibitor will be analysed.

Kinase Substrate: Due to difference in names between the Phosphosite data and the kinase data, it is not possible to retrieve all the substrates for kinases. This is because when 'join query' is applied, it searches for the kinase in the kinase information table against the gene name in the phosphosite table, however if the kinase name (e.g. YES) is different from the gene name (e.g YES1) then there will be no substrates output. Although this is not a serious problem it just limits the interactivity of the site. The phosphosites search accounts for this (however without a direct link) and does show the phosphosites for each protein.
Inhibitors: Because of time constraints, the SQLAlchemy join query only searches the kinase against 'TARGETS1' of the 'inhibitor_information' and not 'TARGETS2', this means that some data for the second target for a particular inhibitor is missing. Provided more time this is a quick fix.

Database limitations:

Inhibitor Data: No synonyms, standardised GSK names and ChEMBL IDs only, this means that when the user uploads their data using PP Tool there is no way to retrieve information about the inhibitor given the inhibitor database Phosphokinase runs on.

PP Tool: After the user uploads the data, the tool searches through the kinase_substrate_filtered *csv* file (containing phosphorylation sites of kinases) instead of searching the database. The main limitation of the PP tool is that it works with the 'Kinase_Phosphosite' table from the database but it has to be converted into a CSV file for the calculation to work. Due to time constraints a relational link between the kinase table ('kinase_information') and the Phosphosite data table (Kinase_Phosphosite) was made but not used. This means that the join query command was not used which would have sped up the search a lot faster. Currently the algorithm checks through 48 phosphorylation site columns which significantly slows the process even further, coupled with checking for proteins using Uniprot API that do not exist on our database.

Technical Solutions:

One of the greatest challenges in Bioinformatics is the standardisation of protein names, and we account for this through our website by including an alias search.

User Journey:

A challenge when developing the website was to consider the user experience and navigation across pages. Initially each page was static and there was no interaction for the user to navigate between the different results for kinase/inhibitors/sites, hence the need for internal hyperlinks and the creation of individual profile pages.

Kinase nomenclature:

Standardisation of names remains a challenge in the field of Science. For database integration and for a smooth user experience the need for a standard naming approach was important. To achieve this, all the targets from the inhibitor table were converted into the names from the kinase table, as well as the names that were uploaded by the user were checked against the kinase table.

Design – Screen Resolution:

Phosphokinase successfully runs on smaller devices such as mobile phones with no detrimental effects on design and user navigation experience. However, the site must be used in landscape mode to reach full functionality.

Uploaded phosphoproteomic file

The PP tool allows the user to upload a .tsv file with phosphoproteomic data with columns arranged in the following order: Substrate, Control_mean, Inhibitor_mean, Fold_Change, p_value, control_CV, treatment_CV. In some cases, the file may not have these 7 columns required to run the analysis, to overcome this, the python script accounts for this error and allows the user to upload files with 5 columns that are missing coefficient of variance data. For data files missing CV data, a default value of 9.9 and default CV threshold of 10 is applied as an extreme measure. As CV values are usually between the range of 0-2. This allows the data to be processed unfiltered as coefficient of variance is unlikely to be of interest if not included

Erroneous Fold Change data

Further, in some cases, the fold change data that is fed into the PP tool is incorrectly calculated which affects the analysis of the results. The python script works to resolve this error by recalculating the fold change values to ensure these are correct.

Data validation – search fields

Empty search string values output errors and causes the site to stop, the user must return to the homepage and search again. In order to overcome this, data validation checks have been added onto the search fields to prevent the user from entering incorrect values.

Parameters

For a succinct analysis of the user input data and personalised results, the integration of parameters was essential. A parameter section was introduced after the user uploads the data to allow them to define their own parameters and thresholds for the analysis. This is compiled into the python script to produce a concise set of results. In addition, a background noise filter was added to remove low level phosphorylation results that may affect the calculated fold change.

Future developments

Although Phosphokinase has an interaction viewer, to further develop this one option includes the ability to filter out the kinase proteins. However, in order to fully understand the phosphorylation network an advancement of the software would be to generate a full kinase interaction network for the known kinases. This would include the kinases that phosphorylate and those that are phosphorylated. A package to achieve this would include using Cytoscape where the user can select nodes and analyse neighbours, but also visualise the whole network. In addition, it would be useful if the user can filter out the kinases based on their location.

The integration of PP tool to the rest of the website would be a major development, for example by taking the inhibitor used as a search string and querying the inhibitor database for that inhibitor, in addition to querying the kinases that are known to phosphorylate the substrates given by the user at particular sites. Error messages can be implemented in the future to generate if a wrong file format is uploaded (i.e. not .tsv file, incorrect number of columns, more than one inhibitor dataset or incorrect fold change calculations).

Another development can include the ability for the user to add more data to the database. For example, if a new protein kinase was discovered the user can have the feature to upload this data, after curation by the developers and publication checks this can then go live to the website. Similarly, if there are new inhibitors known to inhibitors kinases these can also be uploaded. User can also upload a tsv file that handles multiple inhibitor phosphoproteomics data.

Reference

1. Watkins, X., Garcia, L., Pundir, S. and Martin, M. (2017). ProtVista: visualization of protein sequence annotations. Bioinformatics, 33(13), pp.2040-2041.
2. Drewry DH, Willson TM, Zuercher WJ. Seeding collaborations to advance kinase science with the GSK Published Kinase Inhibitor Set (PKIS). Curr Top Med Chem. 2014;14(3):340–2.
3. Hornbeck P.V., Kornhauser J.M., Tkachev S., Zhang B., Skrzypek E., Murray B., Latham V., Sullivan M. PhosphoSitePlus: A comprehensive resource for investigating the structure and function of experimentally determined post-translational modifications in man and mouse. Nucleic Acids Res. 2011;40.
4. Veredas, F., Cantón, F., & Aledo, J. (2017). Methionine residues around phosphorylation sites are preferentially oxidized in vivo under stress conditions. *Scientific Reports*, *7*(1). doi: 10.1038/srep40403
5. Wilkes EH, Casado P, Rajeeve V et al.(2017). Kinase activity ranking using phosphoproteomics data (KARP) quantifies the contribution of protein kinases to the regulation of cell viability. *Mol Cell Proteomics* vol. 16, (9) 1694-1704.