Highly specific prediction of phosphorylation sites in proteins

Matthias Koenig\textsuperscript{1,*} and Niels Grabe\textsuperscript{2}

\textsuperscript{1}CeVis, University of Bremen, Universitaetsallee 29, 28359 Bremen, Germany and
\textsuperscript{2}AgeLab Pharma GmbH, Grandweg 64, 22529 Hamburg, Germany

Received on May 8, 2003; revised and accepted on July 27, 2004
Advance Access publication August 5, 2004

ABSTRACT
Summary: The prediction of significant short functional protein sequences has inherent problems. In predicting phosphorylation sites, problems came from the shortness of phosphorylation sites, the difficulties in maintaining many different predefined models of binding sites, and the difficulties of obtaining highly sensitive predictions and of obtaining predictions with a constant sensitivity and specificity. The algorithm presented in this paper overcomes these problems.

The proposed algorithm PHOSITE is based on the case-based sequence analysis. This enables the prediction of phosphorylation sites with constant specificity and sensitivity. Furthermore, this method leads not only to the prediction of phosphorylation sites in general but also predicts the most probable type of kinase involved.

Availability: The tool PHOSITE implementing the presented method can be evaluated under the website http://www.phosite.com

Contact: koenig@cevis.uni-bremen.de

INTRODUCTION

This paper discusses an algorithm for the prediction of significant short functional protein sequences. This algorithm is described using the example of predicting phosphorylation sites in unknown protein sequences.

Phosphorylation and dephosphorylation processes are important mechanisms regulating cellular functions. Phosphorylation serves to effect critical post-translational modification of proteins having profound effects on their functions, which in turn governs the metabolic processes in a cell and tissue. Protein phosphorylation is an energy-dependent chemical process in which a phosphate moiety is transferred from a high-energy phosphate donor like ATP, GTP, etc. to an acceptor amino acid like tyrosine, serine or threonine in a protein chain. Because phosphorylation sites are short (no longer than $\sim$10 residues), usual sequence alignment tools [e.g. BLAST (Altschul \textit{et al.}, 1990) or FASTA (Pearson and Lipman, 1988)] yield no useful results due to a large number of irrelevant hits.

For the prediction of functional patterns in protein sequences, several approaches have emerged. Methods of the motif approach like PROSITE (Hofmann \textit{et al.}, 1999), BLOCKS (Henikoff, 1999) or PRINTS (Attwood \textit{et al.}, 1999) use a large database of local sequence patterns (or motifs) in order to help identifying given functions or activities of proteins. In the profile approach, a pre-calculated profile (or weight matrix) that is a table of position-specific amino acid weights and gap costs are applied to calculate a similarity score between the profile and a sequence (Gribskov and Devereux, 1991). SCANSITE (Yaffe \textit{et al.}, 2001) applies such pre-calculated matrices generated from peptide library and phage display experimental data for similarity scoring. Hidden Markov models (HMMs) such as PFAM (Bateman \textit{et al.}, 2000) and artificial neural networks found a further category for the considered prediction task.

Nevertheless, it is possible to convert a PFAM HMM into a profile (Bucher \textit{et al.}, 1996), both producing the same raw scores when run against the same sequence. In a similar manner, weight matrices can be mapped onto certain architectures of neural networks where each weight of a neural network corresponds to a value of a weight matrix. An implementation of neural networks for the prediction of phosphorylation sites is realized by NETPHOS (Blom \textit{et al.}, 1999).

As against the profile or pattern approaches, the NETPHOS approach has the advantage of automatic learning. There is no need for maintaining and using several sets of models by hand. On the other hand, the pattern/matrix approach has the advantage of being able to yield more accurate and individual models for the prediction of phosphorylation sites. Nevertheless, these approaches have some drawbacks. Either one allows automatic learning or one provides a set of detailed models. Additionally, both approaches rely on prebuilt models whose underlying known phosphorylation sites had been compiled previously by rather unknown conditions.

SYSTEMS AND METHODS

The method called PHOSITE presented here approaches the difficulty arising prediction appearing from the shortness of phosphorylation sites by case-based model construction. The

\*To whom correspondence should be addressed.
difficult arising from maintaining many predefined different models of binding sites is solved by a fully automatic construction of case-based models. Further, the problem of obtaining predictions with a constant sensitivity and specificity is solved by controlling a few parameters in the case-based model construction. These are applied in the same way for all models and thus yield predictions of a constant accuracy.

The method described here uses any database $D$ [e.g. the PHOSPHOBASE (Kreegipuu et al., 1999)] of biologically proved phosphorylation sites (pps) to predict so far unknown phosphorylation sites (ups) in a given longer protein sequence ($lps$). The method consists of several steps and is a further development for the analysis of phosphorylation sites of AliBaba2 (Grabe, 2002) for transcription factor binding sites:

1. **Preprocessing**: $pps$ are extracted from database $D$ and grouped by the three possible acceptor residues tyrosine, serine and threonine, and subcategorized by kinase classes (e.g. PKC phosphorylation).
2. **Pairwise alignment**: All $pps$ are pairwise aligned to the protein sequence $lps$ to be analysed yielding a set of aligned $pps$ (apps).
3. **Segmentation**: The aligned known binding sites (apps) from the database $D$ are grouped in segments $Gi$, according to their function (i.e. acceptor residue and kinase class) and position relative in $lps$.
4. **Matrix construction**: All $apps$ contained in one segment are aligned, leading to a set of matrices. All $apps$ of a matrix are highly similar to each other, while the according matrix is highly similar to the $ups$ and highly conserved.

**Algorithm**

**Pairwise alignment** is done by applying the PAM 250 matrix (Gonnet et al., 1992). To circumvent negative values here, an offset is added to each PAM entry. Although other matrices could be used, the PAM 250 matrix is chosen here.

The known binding sites are pairwise aligned to the unknown $lbs$ $u$. A matching score between $u$ and each database item $pps$ is calculated. A user-defined threshold compared with the score decides whether the item $pps$ is selected for the construction of a matrix. Considering a $pps$ $s$ with length $n$ and a $lbs$ $u$, the pairwise similarity is given by:

$$\text{pairsim}(u, s) := \sum_{i=1}^{n} \text{PAM}[u(i), s(i)].$$

This results in a set $P$ of aligned pairwise phosphorylation sites (apps) with $\text{pairsim}$ exceeding a user-defined threshold $\text{min_pairsim}$.

A segmentation of the set $P$ is required to structure $P$ by function and position. The classification by function is done by class (acceptor residues, e.g. S-14) and subclass (kinase, e.g. PKC).

The results from the alignment step are divided in blocks $B$ with regard to class and subclass. A pairwise alignment $p \in P$ between $u$ and a binding site is described by a pair $<\text{site_id, site_class}>$.

Then, each block $B$ is split into segments $Gi$ using the position of each $p \in P$, according to the position in the analysed sequence $u$. If $p1, p2 \in B$ overlap a distinct length, $p1$ and $p2$ are put into the same segment $Gi$. Therefore, each $Gi$ contains the $p \in P$ from binding sites that are bound to similarly structured domains and found at similar positions.

**Matrix construction** is done using a gap-free heuristic alignment. Phosphorylation sites of each segment $Gi$ are aligned in sequential order of each site’s similarity. If sites overlap, multiple matrices are constructed for each segment $Gi$. A matrix has 20 rows, each for one amino acid, and $n$ columns, each for a position of the potential $ups$. An item $plA$ stands for the probability of an amino acid $A$ at the position $l$. Gaps are not taken into consideration because of the short length of the phosphorylation sites.

Adopting the strategy of hierarchical clustering, we successively merge binding sites of $Gi$ to a large matrix while this matrix fulfills three requirements. First, the matrix’ average conservation should be above a given minimum conservation ($\text{min_news}$). Second, the matrix should have a minimum similarity to $u$ ($\text{min_sim}$). Third, there should be a minimum number of sites ($\text{min_sup}$) for the support of the sequence.

The conservation is calculated by the average information content (Shannon, 1948):

$$C_l = \log_2(20) + \sum_A p_{lA} \cdot \log_2(p_{lA})$$

$$\text{conservation} := \frac{1}{s} \sum_{l=1}^{s} C_l \quad (1)$$

Here, $C_l$ denotes the conservation of column $l$ in the matrix ($p_{lA}$) with width $s$. The sum runs over all amino acids in column $l$. The calculated column conservation $C_l$ describes the certainty of only one amino acid to occur in column $l$ of the matrix. $C_l$ can be normalized to the range of $[0..100\%]$. From the single $C_l$ the matrix conservation is derived by averaging over all columns $l$ from 1 to $s$.

The measure of the similarity between the built matrix $m$ and the potential phosphorylation site $ups$ is calculated by the following scoring function, similar to the method used by the program MatInspector (Werner, 1999):

$$\text{similarity}(u, m) = \frac{\sum_l C_l p_{lB}}{\sum_l \max_B(p_{lB})} \quad (2)$$

Here, $B_l$ denotes the amino acid residue at position $l$ of $ups$. Alternatively, other measure functions are possible, e.g. the Berg and von Hippel measure (Berg and von Hippel, 1987).
The matrix is constructed as follows: the method decides for each site whether to use it in an already existing matrix or to create a new one. Poorly supported matrices are discarded at the end, as shown in Algorithm 1.

In matrix_construction, the matrix candidate \( m' \) emerges from the already generated matrix \( m \) extended by \( p \). The width of every matrix is chosen by a parameter and stays the same for each matrix, although phosphorylation sites can differ in length. With the parameter \( \text{min}_\text{cons} \), a minimal conservation is required, which means that a minimal amount of information is needed for the existence of a binding site. Accordingly, \( \text{min}_\text{sim} \) denotes the minimal similarity required and \( \text{min}_\text{sup} \) stands for the minimal support required. If the similarity and conservation of an existing matrix candidate \( m' \) are greater than specific thresholds, \( m' \) is accepted instead of the original matrix \( m \). A new matrix is generated from \( p \) in the other case (line 9) and added to \( M \).

At last, Algorithm 2 [\text{generate_matrix}(p)] is applied for generating the extended matrix.

The variable input parameters of the overall method are a minimum threshold \( \text{min}_\text{pairsim} \) for selecting aligned potential phosphorylation sites \( \text{apps} \), a minimum similarity \( \text{min}_\text{sim} \) between matrix \( m \) and the longer protein sequence to be analysed \( \text{lps} \), minimum support \( \text{min}_\text{sup} \) denoting the minimum number of known phosphorylation sites contained in one matrix, minimum conservation \( \text{min}_\text{cons} \) of matrices, minimum overlap \( \text{min}_\text{overlap} \) between an existing matrix and a known phosphorylation site, the input database \( D \) of known phosphorylation sites (hierarchically structured), and the long protein sequence \( \text{lps} \) to be analysed. The parameters enable users to incrementally narrow an analysis down, as it might be reasonable to start with a more tolerant search in some cases (e.g. lower \( \text{min}_\text{sim} \)).

Example of Algorithm

The database of proved phosphorylation sites \( \text{apps} \) is structured as shown in Figure 1. The first level defines the kind of post-translational modification (here 1 = phosphorylation sites), the second level the phosphorylated residue (1 = S, 2 = T, 3 = Y) and the third level the acting kinase (1.1.2 = Phosphorylation, at S, PKA is acting). A zero is used to indicate whether the kinase is unknown. Each known phosphorylation site has its name extended, describing its database origin \( \text{B034} = \text{A16}\@\text{P11217}\_\text{15} \) indicates \( \text{B034} = \text{A16} \) stems from SWISS-PROT database, protein P11217, amino acid 15).
The result of the first step is a pairwise alignment of known phosphorylation sites to the sequence to be analysed. The protein sequence \( u = \text{NTSDTCTSPQRGGALAGGV T} \) will be analysed in this example. Therefore, the following five proved phosphorylation sites \( \text{A001-D, B240-A, B267-A, B267-C and B268-E} \) from the database yield acceptable scores of a pairwise alignment to \( u \). In our example, the length of the database entries and the matrix width is \( n = 9 \) and the minimal overlap is \( \text{min}\_\text{overlap} = 8 \).

In Figure 2, for each site the position in \( u \) is given by the variable \( xp \). The variable \( \text{pwsim} \) shows the calculated pairwise similarity score from Equation (2). The index given by \( xb \) refers to the beginning of the alignment.

All aligned proved phosphorylation sites are then grouped by function, which leads to only one functional block of group 1.1.3 in this example. Further, the aligned proved phosphorylation sites (apps) of this block are grouped into positional segments by sorting them by their position \( xp \), measuring the overlap length \( d \) between subsequently following apps and grouping them according to \( d \) and the threshold \( \text{min}\_\text{overlap} (=8 \text{ for example}) \).

We use \( \text{min}\_\text{pairsim} = 75 \) in our example shown in Figure 2. The first apps appears at position \( xp=1 \). A pairwise similarity of \( \text{pwsim}=75 \) is calculated at position \( xb=1 \) of the apps B267-C. This apps builds the segment \( G_1 = \{ \text{B267-C/xp=1} \} \). The overlap of 7 of the next apps (A001-D/xp=3) and \( G_1 \) is lower than \( \text{min}\_\text{pairsim} \). Thus, a new segment \( G_2 \) is generated. The next three apps overlap totally with \( G_2 \) and thus are placed in \( G_2 \). Segment \( G_1 \) consists of only one apps. In our example, we demand a minimal support \( \text{min}\_\text{sup} = 3 \). For this reason \( G_1 \) is discarded.

The matrices in this example have a length of 8 using the minimum overlap length \( \text{min}\_\text{overlap} \). The matrix construction starts by sorting all apps from \( G_2 \) according to their \( \text{pwsim} \) value: 1. \( \text{A001-D/xp=3} \); 2. \( \text{B240-A/xp=3} \); 3. \( \text{B267-A/xp=3} \); and 4. \( \text{B268-E/xp=3} \). From the first apps the matrix \( m_1 \) is built. This matrix \( m_1 \) for \( G_2 \) is initially constructed from \( \text{A001-D/xp=3} \). We use the alignment positions as the matrix’ columns and the frequencies of the amino acid residues as its rows. Next, the apps \( \text{B240-A/xp=3} \) is tried for enhancing \( m_1 \). As shown in Algorithm 1, a matrix candidate \( m_1' \) is calculated that can replace \( m_1 \). The thresholds for the matrix construction are assumed as follows: \( \text{min}\_\text{cons} = 75\% \) and \( \text{min}\_\text{sim} = 60 \). Applying Equation (1) to matrix \( m_1' \) a conservation of 88\% is calculated. Equation (2) yields a similarity of 93. As both thresholds are exceeded, the matrix candidate \( m_1' \) is accepted and replaces \( m_1 \). Similar steps follow with the apps \( \text{B-267-A/xp=3} \), which is placed in \( m_1 \), as well.

Integrating the apps \( \text{B268-E/xp=3} \) into the matrix \( m_1 \) would not fulfil our threshold constraints, because the conservation = 74\% is too low. Thus, the matrix \( m_1 \) remains unchanged, consisting of \( \{ \text{A001-D/xp=3, B240-A/xp=3, B267-A/xp=3} \} \) and instead a new matrix \( m_2 \) is created with \( \text{B268-E/xp=3} \). Of the two generated matrices, only \( m_1 \) has a support of greater than or equal to the minimal support \( \text{min}\_\text{sup} = 3 \). Hence, the other matrix is discarded. The result of our algorithm is shown in Figure 3. Here, the protein kinase \( p34\text{cdc2} \) is predicted for

---

**Fig. 2.** Example for pairwise alignment.

<table>
<thead>
<tr>
<th>seq:</th>
<th>( \text{NTSDTCTSPQRGGALAGGV T} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( xp=1 )</td>
<td>( xb=1 )</td>
</tr>
<tr>
<td>1.1.3</td>
<td>( +\text{B267-C} )</td>
</tr>
<tr>
<td>( xp=3 )</td>
<td>( xb=0 )</td>
</tr>
<tr>
<td>1.1.3</td>
<td>( +\text{A001-D} )</td>
</tr>
<tr>
<td>( xp=3 )</td>
<td>( xb=0 )</td>
</tr>
<tr>
<td>1.1.3</td>
<td>( +\text{B240-A} )</td>
</tr>
<tr>
<td>( xp=3 )</td>
<td>( xb=1 )</td>
</tr>
<tr>
<td>1.1.3</td>
<td>( +\text{B267-A} )</td>
</tr>
<tr>
<td>( xp=3 )</td>
<td>( xb=0 )</td>
</tr>
<tr>
<td>1.1.3</td>
<td>( +\text{B268-E} )</td>
</tr>
</tbody>
</table>

**Fig. 3.** The result of chosen example.

---

The matrices in this example have a length of 8 using the minimum overlap length \( \text{min}\_\text{overlap} \). The matrix construction starts by sorting all apps from \( G_2 \) according to their \( \text{pwsim} \) value: 1. \( \text{A001-D/xp=3} \); 2. \( \text{B240-A/xp=3} \); 3. \( \text{B267-A/xp=3} \); and 4. \( \text{B268-E/xp=3} \). From the first apps the matrix \( m_1 \) is built. This matrix \( m_1 \) for \( G_2 \) is initially constructed from \( \text{A001-D/xp=3} \). We use the alignment positions as the matrix’ columns and the frequencies of the amino acid residues as its rows. Next, the apps \( \text{B240-A/xp=3} \) is tried for enhancing \( m_1 \). As shown in Algorithm 1, a matrix candidate \( m_1' \) is calculated that can replace \( m_1 \). The thresholds for the matrix construction are assumed as follows: \( \text{min}\_\text{cons} = 75\% \) and \( \text{min}\_\text{sim} = 60 \). Applying Equation (1) to matrix \( m_1' \) a conservation of 88\% is calculated. Equation (2) yields a similarity of 93. As both thresholds are exceeded, the matrix candidate \( m_1' \) is accepted and replaces \( m_1 \). Similar steps follow with the apps \( \text{B-267-A/xp=3} \), which is placed in \( m_1 \), as well.

Integrating the apps \( \text{B268-E/xp=3} \) into the matrix \( m_1 \) would not fulfil our threshold constraints, because the conservation = 74\% is too low. Thus, the matrix \( m_1 \) remains unchanged, consisting of \( \{ \text{A001-D/xp=3, B240-A/xp=3, B267-A/xp=3} \} \) and instead a new matrix \( m_2 \) is created with \( \text{B268-E/xp=3} \). Of the two generated matrices, only \( m_1 \) has a support of greater than or equal to the minimal support \( \text{min}\_\text{sup} = 3 \). Hence, the other matrix is discarded. The result of our algorithm is shown in Figure 3. Here, the protein kinase \( p34\text{cdc2} \) is predicted for
the sequence $u$, because most of the apps generating this matrix are from the subclass of this kinase.

**IMPLEMENTATION**

The program PHOSITE was implemented in C++. The source code compiles and runs on UNIX derivates. The program has a command-line interface and, thus can be controlled by external programs via CGI scripts. PHOSITE requires a database format as described earlier. Conversion of other formats can be achieved by preprocessing them.

**DISCUSSION**

Owing to PHOSITE’s multiple parameters, the results of analysing an unknown protein sequence can differ strongly. We tried to find a way to get an impression of the statistical prediction error of PHOSITE in general. Further, we compared the presented method PHOSITE with SCANSITE (Yaffe et al., 2001) and NETPHOS (Blom et al., 1999) in two examples.

We chose the complete PHOSPHOBASE (Kreegipuu et al., 1999) database (version 2.0, release date December 1998) as input data for estimating sensitivity and specificity of PHOSITE. The database entries were used as positive test examples. To obtain the negative test examples, phosphorylation acceptor residues were cut out of the protein sequences referred to in PHOSPHOBASE.

Sensitivity (true positives) is calculated as the number of correct predictions of phosphorylation sites in an unknown sequence by a Jackknife (leave-one-out) approach.

Specificity (false positives per 1000 amino acids) is measured as the number of phosphorylation sites predicted in the negative example dataset built from PHOSPHOBASE.

The parameter $min\_cons$ is indirectly presented in the following statistics because it is needed for calculating sensitivity and specificity. Parameters that are not analysed in the following statistics are set fixed using the reference parameters $min\_pairsim = 85$, $min\_sim = 70$, $min\_sup = 3$, $min\_overlap = 9$ else are explicitly given. Figure 4 displays the prediction accuracy in dependence on the parameter $min\_pairsim$. With growing specificity/sensitivity, a $min\_pairsim$ of 85 yields the most true positives with the fewest false positives. Figure 5 shows that the ratio between true and false positives increases with growing $min\_sim$. Good ratios are obtained with $min\_sim \geq 75$, e.g. with $min\_sim = 75$ and $min\_cons = 80$, we get 155 true positives and only 1.49 false positives per 1000 amino acid residues. The influence of $min\_sup$ is very high and is displayed in Figure 6. It is very important to consider the values of the conservation $min\_cons$ in this figure. The support has a similar effect on the sensitivity to specificity ratio as the conservation has, e.g. the ratio of $min\_sup = 3$ and $min\_cons = 80$ behaves similar to the one of $min\_sup = 5$ and $min\_cons = 70$. The dependence of the parameter $min\_overlap$ is given in Figure 7. As shown, the best overall ratio is achieved with $min\_overlap = 9$.

Hence, a good parameter set to start an analysis is $min\_pairsim = 85$, $min\_cons = 75$, $min\_sim = 75$, $min\_sup = 3$ and $min\_overlap = 9$.

Further, we compared our method with NETPHOS and SCANSITE by analysing the proteins CBP
Prediction of phosphorylation sites

Fig. 5. Influence of min_sim on prediction accuracy.

Fig. 6. Influence of min_sup on prediction accuracy.

(SwissProt-Id: Q92793) and PTEN (O00633). Both are not contained in the PHOSPHOBASE, so all three approaches have to deal with the unknown data. CBP was used as an example for NETPHOS originally (Hofmann et al., 1999). The results in Table 1 show that NETPHOS, SCANSITE and PHOSITE (depending on the parameters) find many similar potential phosphorylation sites.

There are three known phosphorylation sites in PTEN: at positions 366, 370 and 385. The last two are phosphorylated by a CKII kinase. NETPHOS predicts these sites with the scores 0.732 at position 366, 0.966 at 370 and 0.989 at 385, respectively. In low-stringency levels, SCANSITE predicts CKII at position 385, a phosphorylation site at position 366, but no hit for position 370. PHOSITE predicts eight sites as shown in Figure 8. The predicted site is denoted by the most probable kinase followed by the positions for beginning and ending of the binding site. If no clear assumption about the probable binding kinase can be made, the kinase is denoted
Fig. 7. Influence of min\(_{\text{overlap}}\) on prediction accuracy.

![Graph showing influence of min\(_{\text{overlap}}\) on prediction accuracy.]

Fig. 8. The result of analysing PTEN with the parameters: min\(_{\text{pairsim}}\) = 85, min\(_{\text{sup}}\) = 3, min\(_{\text{cons}}\) = 70, min\(_{\text{overlap}}\) = 9 and min\(_{\text{sim}}\) = 70.

Table 1. Number of S/T/Y sites

<table>
<thead>
<tr>
<th>S/T/Y sites</th>
<th>000633</th>
<th>Q92793</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) NETPHOS</td>
<td>27</td>
<td>127</td>
</tr>
<tr>
<td>(b) SCANSITE</td>
<td>6</td>
<td>61</td>
</tr>
<tr>
<td>(c) PHOSITE</td>
<td>8</td>
<td>59</td>
</tr>
<tr>
<td>Similar a and b</td>
<td>3</td>
<td>35</td>
</tr>
<tr>
<td>Similar a and c</td>
<td>6</td>
<td>36</td>
</tr>
<tr>
<td>Similar b and c</td>
<td>1</td>
<td>19</td>
</tr>
</tbody>
</table>

(a) Predicted sites by NETPHOS with score above 0.5; (b) by SCANSITE with medium stringency; and (c) by PHOSITE with min\(_{\text{pairsim}}\) = 85, min\(_{\text{sup}}\) = 3, min\(_{\text{cons}}\) = 70, min\(_{\text{overlap}}\) = 9 and min\(_{\text{sim}}\) = 70; similar predicted sites by the different approaches.

PHOSITE offers a good specificity and sensitivity ratio and the freedom for the user to modify an understandable set of parameters. As SCANSITE does, PHOSITE gives an important hint regarding which kinase is most likely to interact with a particular site.

PHOSITE prediction accuracy depends on the database information available (here PHOSPHOBASE). But PHOSITE does not need a previously compiled model because its model is built on the fly. Furthermore, independent from the task of predicting phosphorylation sites, other application areas of PHOSITE’s approach can be imagined, e.g. prediction of glycolysis sites with a suitable database.

Besides these advantages, PHOSITE’s approach can occasionally lead to a prediction of non-S/T/Y sites under very tolerant parameter settings, e.g. min\(_{\text{sup}}\) < 3. These can be filtered out by a simple post-processing step checking the predicted site for consistency.

PHOSITE reaches good specificity at the cost of sensitivity. Thus, with strict parameter sets and a request of less than 0.1 false positives per 1000 amino acid residues, PHOSITE predicts only 30–40 true positives in the Jackknife tests.
CONCLUSION
We have described here a new algorithm called PHOSITE to predict phosphorylation sites applicable to protein sequences. The presented method allows the user to tailor the prediction properties. This enables the user to achieve results depending on the favoured sensitivity and specificity ratio. Additionally, PHOSITE is able to predict the most probable kinase phosphorylating the predicted binding site.

Thinking beyond phosphorylation analysis, the proposed approach could be a valuable extension for automatic sequence annotation systems.

REFERENCES