

BioPatML - an XML description language for patterns in biological sequences

Stefan R Maetschke*, Michael W Towsey*¹ and James M Hogan¹

¹Faculty of Information Technology, Queensland University of Technology, GPO Box 2434, Brisbane, QLD 4001, Australia

Email: Stefan R Maetschke* - s.maetschke@qut.edu.au; Michael W Towsey* - m.towsey@qut.edu.au; James M Hogan - j.hogan@qut.edu.au;

*Corresponding author

Abstract

Background: A major challenge in computational biology is the description of biological systems in a way that allows their computational evaluation and exchange between institutes and applications. Recent modeling languages that describe various aspects of biological systems such as the genomic composition, spatiotemporal quantities or biochemical reactions predominately rely on XML (eXtensible Markup Language), as a standardized and well supported format. An exception are description languages for patterns in biological sequences that show a great diversity in format and function, which impedes the definition and the exchange of biological patterns.

Results: In this paper we introduce **BioPatML**, an XML-based pattern description language that supports a wide variety of patterns and allows the construction of complex, hierarchically structured patterns and pattern libraries. **BioPatML** unifies the diversity of current pattern description languages and fills a gap in the set of XML-based description languages for biological systems. The paper discusses the structure and elements of the language, and demonstrates its advantages on three applications. An XML schema, manual and **DIANA**, a command line tool to search **BioPatML** patterns in nucleotide and amino acid sequences, are available at <http://eresearch.fit.qut.edu.au/biopatml>.

Conclusions: **BioPatML** increases the power of classical pattern definition languages through principled aggregation. It furthermore simplifies the compilation of pattern libraries and promotes exchange of complex patterns. The language provides a convenient format to encapsulate pattern definitions and their annotations for integrated bioinformatic analyses.

Background

A major challenge in computational biology is the development of standardized, machine-readable descriptions of biological systems to enable the computational evaluation and the unhampered exchange of biological data. In this context a strong tendency toward XML (eXtensible Markup Language) as a description language for biological data can be observed. Examples are, CELLML for the mathematical description of biological systems, FIELDML to model time-varying and spatially-varying fields, SBML to represent biochemical reaction networks, CML to describe chemical concepts, CISML as an output format for motif detection software, and numerous other XML formats to describe genomic data such as INSDSEQ, BSML, AGAVE, DDBJ-XML, EMBLXML and CHADO-XML.

Representing data in XML has distinct advantages over proprietary data formats. XML is a standardized format (<http://www.w3.org/XML>) for which parsers are available in most modern programming languages and web-browsers. The syntax of an XML document can be validated against a DTD (Document Type Definition) or a schema and is easy to extend. Furthermore, XML documents are readily transformed, formatted and styled, utilizing extensible style sheet languages (XSL). Finally, many databases natively support XML or provide mappings for XML documents.

This paper introduces **BioPatML**, an XML based description language for patterns in biological sequences that offers significant advantages over other pattern description languages and that fills a gap in the set of description languages for biological systems.

Current pattern description languages for biological sequence patterns are predominantly subsets or minor extensions of regular expressions with limited modeling power. For instance, KANGAROO, [1] defines patterns as plain regular expressions with an extension to allow IUPAC ambiguity symbols [2]. However, mismatch thresholds, weighted gaps and inverted repeats are not supported. The similar PATMATCH [3] language is limited in the same way, although mismatch thresholds are provided.

Many other pattern languages exist, and broadly speaking, these may be categorized as Extended Regular Expressions – such as those outlined above – or Stochastic Grammars, such as the Profile Hidden Markov Models described in [?]. A number of specializations of these approaches exist for particular domains.

PROSITE¹ [4] and PEPAT [5] are dedicated to the description of peptide sequences. PROSITE is equivalent in expressive power to a regular expression but implements a slightly different syntax. PEPAT provides an extension to similarity scoring using substitution matrices, and supports the PROSITE syntax. Inverted repeats and weighted gaps are not implemented, however. 3OF5 [6] simplifies the description of frequently occurring patterns, such as n matches of a set of symbols in a sequence of length m .

The Profile-HMM [?] approach is equivalent in expressive power to a stochastic regular grammar, and is particularly popular as a format for protein families. Profile-HMM models are derived from sets of aligned peptide sequences, and stored as predefined patterns in databases such as PROSITE [4] and PFAM [7]. However, Profile-HMMs cannot describe overlapping or repeated patterns, and remain fundamentally probabilistic regular expressions i.e. they are *not* Stochastic Context Free Grammars.

PATSCAN [8] and its improved successor PATSEARCH [9,10] are more closely aligned with the present work. While fundamentally based on regular expression representations, these approaches support hierarchical aggregation of patterns, direct and inverted repeats, overlapping patterns, mismatch thresholds and position weight matrices. PATSEARCH permits insertions and deletions in matching a motif but does not support weighted gaps and similarity scoring is only partially available.

General purpose programming languages on the other hand, such as JAVA, PYTHON or PERL, which are frequently utilized to search for patterns, are not specifically designed for biological sequences and require significant effort to solve characteristic pattern matching problems, perceived as apparently simple and fundamental to the biologist.

BioPatML is a pattern description language which addresses these deficiencies, providing support for a broad range of component patterns, and a rich grammatical structure for their combination. The language thus allows a common representation for structured patterns equally useful in pattern databases and pattern search. The following section introduces the elements of **BioPatML** and demonstrates its use in three applications.

Results

The development of **BioPatML** was initiated by difficulties encountered while modeling bacterial promoters. For instance, the canonical model of a σ^{70} -promoter in *E. coli* consists of the motif TTGACA separated by a variable gap from the motif TATAAT [11] (see Figure ??). However, the two hexamers are poorly conserved, and the distribution of gap lengths is not uniform. Such a pattern cannot be described

¹Prosite patterns are used to store protein motifs in the PROSITE database at <http://au.expasy.org/prosite>.

accurately by a regular expression or a position weight matrix (PWM). Our primary goal was to develop an XML-based pattern description language that permits the consistent aggregation of different pattern description paradigms (such as PWMs, mismatch motifs and regular expressions).

The aggregation of patterns requires the solution of two problems: (1) the unification of the syntactic elements of different languages, and (2) the unification of language-specific scoring schemata. **BioPatML** addresses the first of these problems by wrapping classical patterns within XML tags. For instance, the regular expression "L.{6}L.{6}L.{6}L" used to describe the Leucine-zipper motif becomes instead `<Regex regex="L.{6}L.{6}L.{6}L"/>`.

The second problem is more difficult. Information theoretic similarity measures, such as information content or log-odd scores, are compatible with probabilistic structures such as PWMs, but not with classical regular expressions. Similarly, the number of mismatches between motif sequences is not directly comparable to the score from a PWM, although a sensible relation may be defined. Within **BioPatML**, some degree of comparability is assured by mapping all scoring schemes to the common fixed interval [0, 1], where a score of one indicates the best possible match for the defined pattern and a score of zero is the worst possible match. Interpretation of the score depends on the specific pattern and search criteria. The overall similarity score for a compound pattern is given by a weighted sum of the partial similarity scores, where the weightings are chosen by the user.

In the following we first describe the structure of a **BioPatML** document, before discussing the language elements and three example applications.

Document structure

A **BioPatML** pattern is represented by an XML document that is divided into the XML document *header* and the pattern *definition*. The pattern definition itself is composed of an optional *annotations* section, an optional set of *definitions* of sub-patterns and the *pattern description*:

```
<?xml version="1.0"?>
<BioPatML
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="BioPatML.xsd">

  <Definition name="..." >
    <Annotations>
      <...pattern annotations...>
    </Annotations>
    <Definitions>
      <...sub-definitions...>
    </Definitions>
    <...pattern description...>
  </Definition>
</BioPatML>
```

```
</Definition>
```

```
</BioPatML>
```

A pattern description is defined by an XML tag and attributes that represent the parameters of the pattern. For instance, the TATAAT motif of the -10 element (Pribnow-box) in a σ^{70} -promoter might be described as

```
...
<Motif motif="TATAAT" alphabet="DNA" threshold="0.7" />
...
```

where `motif`, `alphabet` and `threshold` are the parameters of the `<Motif>` pattern. **BioPatML** offers a wide range of different pattern types that are explained in more detail in Section ??.

A pattern definition can also contain a `<Definitions>` block with sub-definitions, either to define patterns that are used more than once or to import or to construct pattern libraries. The following example shows sub-definitions for the -10 and -35 element patterns of a σ^{70} -promoter:

```
...
<Definition name="sigma70" >
  <!-- sub-definitions -->
  <Definitions>
    <Definition name="-10element" >
      <Motif motif="TATAAT" alphabet="DNA" threshold="0.7" />
    </Definition>
    <Definition name="-35element" >
      <Motif motif="TTGACA" alphabet="DNA" threshold="0.7" />
    </Definition>
  </Definitions>
  ...
</Definition>
...
```

The pattern definitions within the `<Definitions>` block are addressed within a pattern description by the `<Use>` pattern. For instance, the canonical model of promoter, using the pattern definitions above, could be described as

```
...
<Series ... >
  <Use definition="-35element" />
  <Gap ... />
  <Use definition="-10element" />
</Series>
...
```

where the `<Series>` pattern defines a sequential arrangement of the promoter elements, and the `<Gap>` pattern describes the interjacent gap.

Since the `<Definitions>` block of an **BioPatML** documents can contain an arbitrary number of pattern definitions, **BioPatML** documents also serve as pattern libraries. An existing **BioPatML** library is

imported by the `<Import>` command. Provided that the promoter element definitions showed above were stored under "promoter.bpl" they could be used as follows:

```
...
<Definition name="Promoter" >
  <Definitions>
    <Import uri="promoter.bpl" />
  </Definitions>
  <Series ... >
    <Use definition="sigma70.-35element" />
    <Gap ... />
    <Use definition="sigma70.-10element" />
  </Series>
</Definition>
...
```

Pattern definitions within a library are addressed by a dot separated path, with the names of the encapsulating definitions as elements of the path. The hierarchy of definitions and sub-definitions can be arbitrary deep, allowing paths such as: `QUT.promoter.EColi.sigma70.-35element`. Note that the import function loads pattern libraries from locations specified by an URI (Uniform Resource Identifier) and therefore supports the construction of web-wide, centrally administered pattern libraries. A pattern definition can be augmented by annotations (such as author, date etc.), represented as name/value pairs. This format is flexible enough to translate annotation information of existing pattern description languages (e.g. PROSITE) into **BioPatML** without loss of meta-data.

```
...
<Annotations>
  <Annotation name="Author">S. Maetschke</Annotation>
  <Annotation name="Date">04.04.2007</Annotation>
  ...
</Annotations>
...
```

Language Elements

We now describe the elements of the language in more detail, with illustrative examples provided as appropriate. For clarity, we show only the pattern description and generally omit document headers, annotations and pattern sub-definitions if not required.

A listing of the principal language elements of **BioPatML** is given in Table ???. In this section, we shall first outline the high-level features of the language – management of alphabets, general element and attribute syntax – before proceeding to examine the language elements in more detail, grouping our discussion according to tag function.

Element	Description
<Any>	matches any sequence of the specified length
<Gap>	variable, (weighted) gap between patterns
<Composition>	composition of a region
<Motif>	motif with mismatches
<Regex>	regular expression
<Prosite>	regular expression in PROSITE syntax
<PWM>	position weight matrix
<Block>	block of aligned sequences
<Set>	set of alternative patterns
<Series>	ordered series of patterns
<Iteration>	iteration of a pattern
<Repeat>	direct or inverted repeat of a pattern match
<Logic>	boolean conjunction of pattern matches
<Void>	placeholder
<Constraint>	positional constraints for pattern matches
<Alignment>	alignment of pattern matches

Discussion of the *pattern definition* is deferred to the individual structural groupings, but we shall here consider a range of commonly occurring attributes which support specification and detection of patterns. Specifications typically incorporate a number of the following attributes:

- an optional **name**, which serves as an unique identifier of a pattern description;
- an **alphabet**, drawn from the predefined set of alphabets: (DNA, RNA, AA) – the selection automatically enables a set of ambiguity symbols;
- a similarity **threshold**, which controls whether or not a match is reported;
- a match **impact** which determines the weight of a component match score within the aggregated match scores, being defined as the weighted sum of the component scores;
- and a match **mode**, which controls handling of multiple matches. Structured patterns with variable gaps may match multiple times at the same position. The match **mode** (ALL, BEST) determines whether *all* those matches or only the *best* match is returned.

We now consider the simpler patterns, before examining the combination mechanisms of structured patterns and special patterns.

Region Patterns

Region patterns describe the length or contents of regions of a sequence, without being specific about position-related properties of the region. The simplest pattern of this type is the <Any> pattern, that

matches any subsequence, regardless of its content, whose length lies in the specified range [minimum..maximum].

```
<Any minimum="6" maximum="8" />
```

The <Gap> pattern extends the <Any> pattern by allowing gap weights to define preferences over a distribution of gap lengths. The pattern is typically used to define gaps of variable lengths between two patterns. The values of the weight array can be either integer or real and are internally normalized. Gaps furthermore support a real valued **increment** parameter to describe periodic properties of regions (e.g. α -helical protein structures show a typical periodicity of 3.6 residues per turn):

```
<Gap minimum="2" maximum="13" increment="3.6" >
  <Weights> 1 1 2 3 2 3 2 4 4 2 1 1 </Weights>
</Gap>
```

The <Any> and the <Gap> pattern ignore the content of the matched region. To model the overall composition of a region the <Composition> pattern is provided. It defines symbol weights to express preferences for specific amino acid or nucleotide within a region. The following, simplified example describes a hydrophobic protein region (e.g. a transmembrane domain):

```
<Composition name="transmembrane domain"
  minimum="15" maximum="25" increment="1.0"
  mode="BEST" alphabet="AA" threshold="0.7" >
  <Symbol letter="V" weight="1.0" />
  <Symbol letter="I" weight="1.0" />
  <Symbol letter="L" weight="1.0" />
  <Default weight="0.0" />
</Composition>
```

Motif Patterns

Motif patterns represent frequently reoccurring motifs in biological sequences. In contrast to the region patterns discussed above, motif patterns express position-specific assumptions of the content of a region. Strongly conserved motifs are readily described by the <Motif> pattern. The pattern matches successfully if the fraction of matching symbols is greater than or equal to the specified **threshold**. Ambiguity symbols and alternatives (specified in brackets) are supported. The following example of a Pribnow-box motif matches if 60% of the motif symbols match:

```
<Motif name="Pribnow-box"
  alphabet="DNA" motif="TA[AT]AAT" threshold="0.6" />
```

More complex motifs may be described using standard regular expressions inside the <Regex> pattern. However, in contrast to the <Motif> pattern, the matching process is based only on standard alpha-numeric characters since regular expressions do not natively support alphabets or ambiguity symbols.

```
<Regex name="C6 zinc-finger"
  regex="C.{2}C.{6}C.{5,6}C.{2}C.{6}C"/>
```

BioPatML also supports the simplified regular expressions of the PROSITE language:

```
<Prosite name="Leucine-zipper" alphabet="AA"
  prosite="C-x(2)-C-x(6)-C-x(5,6)-C-x(2)-C-x(6)-C" />
```

Yet each of these approaches is fundamentally character based, reporting matches according to whether each character is in its proper position. The Position Weight Matrix (PWM) allows a richer view of the matching problem, supporting a score based on the probability of the occurrence of a particular symbol at each specific position in the motif. PWMs are described by the `<PWM>` pattern. The match score of a PWM is automatically scaled to the interval $[0, 1]$, and a match is reported if the normalized score is greater than or equal to the defined threshold:

```
<PWM name="Pribnow-box" alphabet="DNA" threshold="0.6">
  <Row letter="a"> -2.4  1.3 -0.9  1.0  0.9 -2.2 </Row>
  <Row letter="c"> -1.0 -1.7 -0.8 -0.6 -0.0 -1.5 </Row>
  <Row letter="g"> -1.3 -1.7 -0.5 -0.5 -1.3 -1.9 </Row>
  <Row letter="t">  1.3 -1.2  1.0 -1.2 -0.9  1.4 </Row>
</PWM>
```

Another method for defining a PWM is by providing a list of aligned sequences, from which the weights are derived. Similar to pattern description within the PRINTS [12] or the BLOCKS [13] databases, **BioPatML** implements the `<Block>` pattern for this purpose:

```
<Block name="Pribnow-box"
  alphabet="DNA" threshold="0.7">
  <Sequence> TATAAT </Sequence>
  <Sequence> TAAAAT </Sequence>
  <Sequence> TATGAT </Sequence>
  <Sequence> TATAAA </Sequence>
  <Sequence> TCGAAT </Sequence>
</Block>
```

PWMs are frequently employed in the search for regulatory elements in DNA sequences, and more sophisticated searches often seek to boost the detection of the traditional dual-hexamer patterns by incorporating additional features local to the transcription start site. This combination of patterns lies at the heart of **BioPatML**, and is considered in detail in the following section.

Structured Patterns

Structured patterns are aggregations of sub-patterns. Each of the sub-patterns can be an arbitrarily complex **BioPatML** pattern itself.

Frequently a pattern may be profitably viewed as one from among a series of alternatives, for example nuclear localization signals [14]. The alternative patterns can be collected within the `<Set>` pattern. The `<Set>` pattern matches if at least one of the included patterns matches and achieves a similarity score greater than or equal to the defined `threshold`. The `mode` attribute specifies whether all possible matches of a set are returned (`mode="ALL"`) or only the best (`mode="BEST"`):

```
<Set name="nuclear localization signal"
  threshold="1.0" mode="BEST" >
  <Regex name="NLS:144" regex="RKCLQAGMNLEARKTKK" />
```

```

<Regex name="NLS:146" regex="RKKRKR" />
<Regex name="NLS:147" regex="RKKRK.{9}KAKKSK" />
<Regex name="NLS:156" regex="RKR[PLQMN]R[PLQMN]R"/>
...
</Set>

```

The patterns contained within a `<Set>` are unordered and it is sufficient if one of the sub-pattern matches for a successful match of the `<Set>` pattern. In contrast, the `<Series>` pattern is used to define an ordered series of sub-patterns that matches only if the mean similarity score of all embedded patterns is above the specified `threshold`. In the following example, only the two defined motifs influence the overall match score because the impact factor for the gap is zero. The `<Series>` threshold is set at zero to enable return of the highest scoring match, but note that a match will be returned only if *both* motifs exceed the motif match score of 0.7:

```

<Series name="canonical sigma70-promoter"
  mode="BEST" threshold="0.0">
  <Motif name="-35 element"
    alphabet="DNA" motif="TTGACA" threshold="0.7"/>
  <Gap minimum="15" maximum="21" impact="0.0"/>
  <Motif name="-10 element"
    alphabet="DNA" motif="TATAAT" threshold="0.7"/>
</Series>

```

The number of sub-patterns within a `<Series>` is fixed. When a variable number of matches of a pattern is required, the `<Iteration>` pattern can be employed. It matches successfully when the encapsulated pattern matches for the required `minimum` number of iterations and the mean similarity score over all matches is not below the `threshold`. No more than the specified `maximum` number of iterations will be performed. The following example pattern searches for the motifs `GA`, `GAGA` or `GAGAGA`:

```

<Iteration minimum="2" maximum="3" threshold="1.0">
  <Motif alphabet="DNA" motif="GA" threshold="1.0"/>
</Iteration>

```

The `<Iteration>` pattern independently matches the same pattern multiple times. To find repeats of previously matched patterns the `<Repeat>` pattern was implemented. To this purpose the pattern refers to a previously matched pattern, using the pattern name as an identifier. The pattern description below finds all palindromes of length 8 to 10 with a mismatch allowance of 30%. The `mode` attribute of the `<Repeat>` determines whether direct ("`DIRECT`") or inverted ("`INVERTED`") repeats are searched and the `pattern` attribute specifies the pattern, which match is to be repeated:

```

<Series name="palindrome" mode="ALL" threshold="0.0">
  <Any name="any" minimum="4" maximum="5"/>
  <Repeat pattern="any" mode="INVERTED" threshold="0.7" />
</Series>

```

In addition to ambiguity symbols the <Repeat> pattern furthermore supports non-standard, weighted pairings of symbols when matching repeats (see Section ??).

Boolean operations on the matches of multiple patterns are performed by the <Logic> pattern. Depending on the chosen operation ("AND", "OR", "EXOR") the <Logic> pattern matches if all, some or exactly one of its sub-pattern matches – provided that the mean similarity score of the matches is not below the specified **threshold**. The following simplified pattern searches for a hydrophobic stretch of amino acids, composed of at least 60% of Leucines (L) that contains a helix-helix interaction motif ("GxxxGxxxG"):

```

<Logic operation="AND" threshold="0.0">
  <Motif alphabet="AA" motif="LLLLLLLLLLLLLLLL" threshold="0.6"/>
  <Motif alphabet="AA" motif="xxxGxxxGxxxGxxx" threshold="1.0"/>
</Series>

```

Note that the <Logic> pattern operates on an arbitrary number of patterns, which are not required to match at the same length.

Special Patterns

Special patterns perform special functions outside the classification schemata for patterns utilized above.

The simplest of the special patterns is the <Void> pattern, that serves as a placeholder and always matches. It is frequently employed in pattern libraries to signalize the lack of a main pattern description:

```

<BioPatML ... >
  <Definition name="mylib" >
    <Definitions>
      <...pattern definitions...>
    </Definitions>
    <Void/>
  </Definition>
</BioPatML>

```

The <Constraint> pattern expresses positional constraints. It matches only at the position of the sequence that is specified by the **position** and the **offset** attributes of the pattern. The **position** attribute describes a symbolic position and can refer to the "START", "END" or "CENTER" of the sequence. The **offset** attribute defines an offset relative to the provided symbolic position. For instance, to identify membrane proteins that carry the classical di-lysine KKXX retrieval signal at the extreme C-terminus, the following pattern description can be used:

```

<Series mode="BEST" threshold="0.0">
  <Motif motif="KK" alphabet="AA" threshold="1.0"/>
  <Constraint position="END" offset="-2" />
</Series>

```

The `<Alignment>` pattern aligns the match of predecessor pattern with the match of a successor pattern.

The `pattern` attribute specifies the predecessor pattern and the `position` and the `offset` attributes

describe the location relative to the predecessor pattern, the start of the successor pattern is aligned with. The `<Alignment>` patterns is typically employed to describe overlapping pattern. For instance, the VP40 protein of the Ebola virus contains two overlapping L-domain motifs PTAP and PPxY. The PTAP motif appears in a distance of 7 residues to the N-terminal and the last residue of the motif overlaps with the first residue of the PPxY motif [15]. Such a pattern may be described as:

```

<Series mode="BEST" threshold="0.0">
  <Constraint position="START" offset="+7" />
  <Motif name="motif1" motif="PTAP" alphabet="AA" threshold="1.0" />
  <Alignment pattern="motif1" position="END" offset="-1" />
  <Motif name="motif2" motif="PPxY" alphabet="AA" threshold="1.0" />
</Series>

```

Note that the pattern description above allows to assign different similarity thresholds or impact factors to the two L-domain motifs to model differences in conservation or effectivity, which would not be possible for the combined motif PTAPPxY. This difference is of practical relevance, since the PTAP and PPxY are functionally redundant and can act independently as L-domains [15].

Example applications

In this section we demonstrate the use of **BioPatML** for three typical pattern search problems in DNA, RNA and amino acid sequences. More precisely we devise patterns to locate promoters and transcription start sites (TSS) in *E. coli*, Metazoan histone stem-loops in messenger RNA and internalization signals in transmembrane proteins.

Transcription start site prediction

Bacterial gene transcription is initiated when an enzyme complex, consisting of RNA polymerase and a sigma factor, binds to a promoter upstream of the actual transcription start site. The canonical model of a σ^{70} -promoter specifies two hexamers, separated by a gap of approximately 15-21bp, and located near positions -35 and -10 relative to the TSS (see Figure ??). The consensus sequence for the -35 motif is TTGACA, and that for the -10 motif is TATAAT [11].

The classical promoter model for *E. coli* shows an optimal gap length of 17bp between the two hexamers (we refer to this gap as the *spacer*) and a distance of 7bp between the -10 motif and the TSS (we refer to

this gap as the *discriminator*). The nucleotide at the TSS location is typically a purine [16]. Such a pattern can readily be expressed with **BioPatML**:

```
<Series mode="BEST" threshold="0.0">
  <PWM name="-35element" alphabet="DNA" threshold="0.0">
    <Row letter="a">-1.5 -2.3 -1.5 0.8 -0.8 0.9</Row>
    <Row letter="g">-1.5 -1.8 1.3 -0.6 -1.1 -0.0</Row>
    <Row letter="c">-0.8 -1.6 -0.5 -0.7 1.3 -1.0</Row>
    <Row letter="t"> 1.2 1.4 -0.5 -0.4 -0.5 -1.0</Row>
  </PWM>
  <Gap name="Spacer" minimum="15" maximum="21" impact="0.2">
    <Weights>
      0.15 0.16 0.20 0.16 0.09 0.12 0.11
    </Weights>
  </Gap>
  <PWM name="-10element" alphabet="DNA" threshold="0.0">
    <Row letter="a">-2.2 1.3 -0.8 1.1 1.2 -2.1</Row>
    <Row letter="g">-1.2 -1.1 -0.8 -0.5 -1.4 -2.0</Row>
    <Row letter="c">-0.5 -2.1 -0.9 -1.2 -0.4 -1.5</Row>
    <Row letter="t"> 1.2 -1.3 1.0 -1.1 -1.4 1.4</Row>
  </PWM>
  <Gap name="Discriminator" minimum="4" maximum="12" impact="0.2">
    <Weights>
      0.05 0.11 0.27 0.22 0.09 0.05 0.08 0.06 0.08
    </Weights>
  </Gap>
  <Motif name="TSS" impact="0.2"
    alphabet="DNA" motif="R" threshold="0.0"/>
</Series>
```

The weights of the PWMs and the gaps were extracted from [?]. ...

Metazoan histone stem-loop

Metazoan histone stem-loop elements are *cis*-regulatory elements in the 5'- or 3'-UTR regions of eukaryotic mRNAs that fold into a specific secondary structure [9].

...

To compare PATSEARCH and **BioPatML** we took this example from the PATSEARCH paper [9] and present the PATSEARCH pattern and the equivalent **BioPatML** pattern.

```
<Series name="stem-loop" mode="BEST" threshold="0.0">
  <Any name="5'flank1" minimum="0" maximum="1" />
  <Motif name="5'flank2" motif="mmm" alphabet="RNA" threshold="1.0"/>
  <Motif name="stem1" motif="ggyyy" alphabet="RNA" threshold="1.0"/>
  <Motif name="loop" motif="uhhuha" alphabet="RNA" threshold="1.0"/>
  <Repeat name="stem2" pattern="motif" mode="INVERTED" threshold="1.0" >
    <Pairing original="a" repeat="u" />
```

```

    <Pairing original="u" repeat="a" />
    <Pairing original="g" repeat="c" />
    <Pairing original="c" repeat="g" />
    <Pairing original="g" repeat="u" />
    <Pairing original="u" repeat="g" />
  </Repeat>
  <Motif name="3'flank1" motif="mmm" alphabet="RNA" threshold="1.0"/>
  <Any name="3'flank2" minimum="0" maximum="3" />
</Series>

```

PATSEARCH pattern:

```

r1={au,ua,gc,cg,gu,ug}
0..1 mmmm p1=ggyyy u hhuh a r1~p1 mm 0..3

```

...

Transmembrane protein internalization signal

```

<Series name="internalization" mode="BEST" threshold="0.0">
  <Composition name="transmembrane domain"
    minimum="15" maximum="25"
    mode="BEST" alphabet="AA" threshold="0.0" >
    <!-- Kyte-Doolittle hydrophobicity scale -->
    <Symbol letter="A" weight=" 1.8" />
    <Symbol letter="R" weight="-4.5" />
    ...
    <Symbol letter="V" weight=" 4.2" />
  </Composition>
  <Gap minimum="6" maximum="9" />
  <Motif motif="YXX[FYW]" alphabet="AA" threshold="1.0"/>
  <Gap minimum="0" maximum="1" />
  <Constraint position="END" />
</Series>

```

...

Conclusions

Authors contributions

Acknowledgements

We thank S. Mann and J. Sumitomo for administering the server. This work was supported by the Australian Research Council under its Special Research Initiative programme, and by the QUT Strategic Collaborative Programme in Bioinformatics.

References

1. Betel D, Hogue CWV: **Kangaroo—a pattern-matching program for biological sequences.** *BMC Bioinformatics* 2002, **3**:20.

2. Cornish-Bowden: **IUPAC-IUB SYMBOLS FOR NUCLEOTIDE NOMENCLATURE**. *Nucl. Acids Res* 1985, **13**:3021–3030.
3. Yan T, Yoo D, Berardini TZ, Mueller LA, Weems DC, Weng S, Cherry JM, Rhee SY: **PatMatch: a program for finding patterns in peptide and nucleotide sequences**. *Nucleic Acids Res* 2005, **33**(Web Server issue):W262–W266, [<http://dx.doi.org/10.1093/nar/gki368>].
4. Hulo N, Sigrist CJA, Saux VL, Langendijk-Genevaux PS, Bordoli L, Gattiker A, Castro ED, Bucher P, Bairoch A: **Recent improvements to the PROSITE database**. *Nucl. Acids Res* 2004, **32**(Database issue):D134–D137, [<http://dx.doi.org/10.1093/nar/gkh044>].
5. Jiang Y, Gao G, Fang G, Gustafson EL, Laverty M, Yin Y, Zhang Y, Luo J, Greene JR, Bayne ML, Hedrick JA, Murgolo NJ: **PepPat, a pattern-based oligopeptide homology search method and the identification of a novel tachykinin-like peptide**. *Mamm Genome* 2003, **14**(5):341–9, [<http://dx.doi.org/10.1007/s00335-002-3061-y>].
6. Seiler M, Mehrle A, Poustka A, Wiemann S: **The 3of5 web application for complex and comprehensive pattern matching in protein sequences**. *BMC Bioinformatics* 2006, **7**:144 (Epub), [<http://dx.doi.org/10.1186/1471-2105-7-144>].
7. Bateman A, Birney E, Cerruti L, Durbin R, Eddy SR, G-J S, Howe KL, Marshall M, Sonnhammer ELL: **The Pfam protein families database**. *Nucleic Acids Research* 2002, **30**:276–80.
8. Dsouza M, Larsen N, Overbeek R: **Searching for patterns in genomic data**. *Trends Genet* 1997, **13**(12):497–8.
9. Pesole G, Liuni S, D'Souza M: **PatSearch: a pattern matcher software that finds functional elements in nucleotide and protein sequences and assesses their statistical significance**. *Bioinformatics* 2000, **16**(5):439–450.
10. Grillo G, Licciulli F, Liuni S, Sbisà E, Pesole G: **PatSearch: A program for the detection of patterns and structural motifs in nucleotide sequences**. *Nucleic Acids Research* 2003, **31**(13):3608–3612.
11. Huerta AM, Collado-Vides J: **Sigma70 promoters in *Escherichia coli*: specific transcription in dense regions of overlapping promoter-like signals**. *J Mol Biol* 2003, **333**(2):261–278.
12. Attwood TK: **The PRINTS database: a resource for identification of protein families**. *Brief Bioinform* 2002, **3**(3):252–263.
13. Pietrokovski S, Henikoff JG, Henikoff S: **The Blocks database—a system for protein classification**. *Nucleic Acids Res* 1996, **24**:197–200.
14. Cokol M, Nair R, Rost B: **Finding nuclear localization signals**. *EMBO* 2000, **1**:411–415.
15. Licata JM, Simpson-Holley M, Wright NT, Han Z, Paragas J, Harty RN: **Overlapping motifs (PTAP and PPEY) within the Ebola virus VP40 protein function independently as late budding domains: involvement of host proteins TSG101 and VPS-4**. *J Virol* 2003, **77**(3):1812–1819.
16. Harley CB, Reynolds RP: **Analysis of *E. Coli* promoter sequences**. *Nucleic Acids Research* 1987, **15**(5):2343–2361.

Figures

Figure 1 - Sample figure title

A short description of the figure content should go here.

Figure 2 - Sample figure title

Figure legend text.

Tables

Table 1 - Sample table title

Here is an example of a *small* table in L^AT_EX using `\tabular{...}`. This is where the description of the table should go.

Element	Description
<Motif>	motif with mismatches

Table 2 - Sample table title

Large tables are attached as separate files but should still be described here.

Additional Files

Additional file 1 — Sample additional file title

Additional file descriptions text (including details of how to view the file, if it is in a non-standard format or the file extension). This might refer to a multi-page table or a figure.

Additional file 2 — Sample additional file title

Additional file descriptions text.