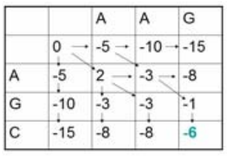
Patterns

Tore Samuelsson 2010

Patterns - comparison to method of aligning evolutionary related sequences

Dynamic programming algorithm. Designed to deal with closely as well as distantly related sequences, taking into account gaps (indel mutations) and substitution matrix information



		A	A	G
	0	-5	-10	-15
A	-5	2	-3	-8
G	-10	-3	-3	-1
C	-15	-8	-8	-6

Applications in:

- Sequence assembly
- Classification
- Prediction of function
- Comparative genomics
- Phylogeny / Evolutionary history

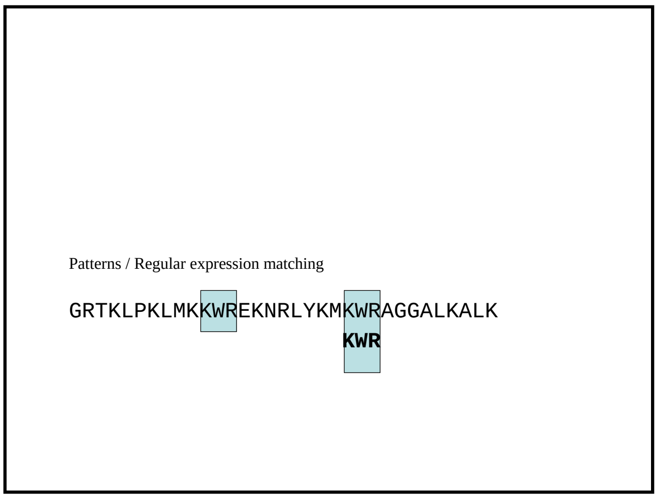
Patterns

Regular expression "Regexp" "Regex"

Patterns / Regular expression matching

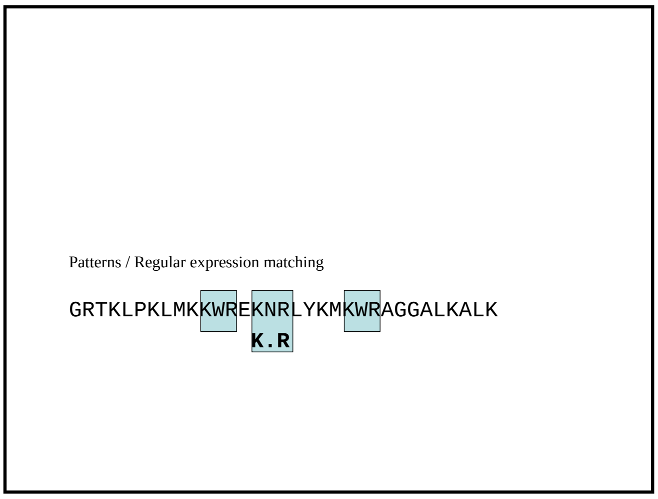
`GRTKLPKLMKKWREKNRLYKMKWRAGGALKALK`

Is there a match to **`KWR`** in this string?


Patterns / Regular expression matching

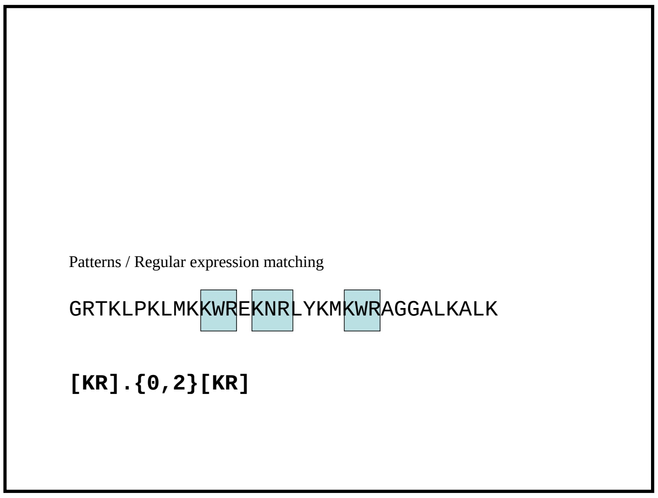
GRTKLPKLMKKWREKNRLYKMKWRAGGALKALK

KWR


Patterns / Regular expression matching

GRTKLPKLMKKWREKNRLYKMKWRAGGALKALK

K.R


Patterns / Regular expression matching

GRTKLPKLMKKWREKNRLYKMKWRAGGALKALK

[KR].{0,2}[KR]

Regular expressions and NFA / DFA

Regular expressions are related to theory of *automata (machines)*:
Regular expressions can be translated into :

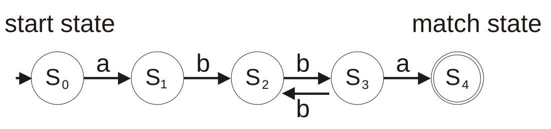
Non-deterministic Finite Automaton (NFA) &

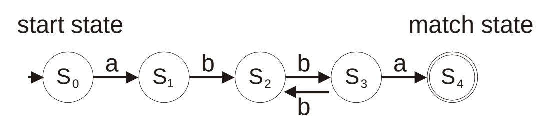
Deterministic Finite Automaton (DFA)

(and this is what happens in Perl and other utilities that make use of regexs)

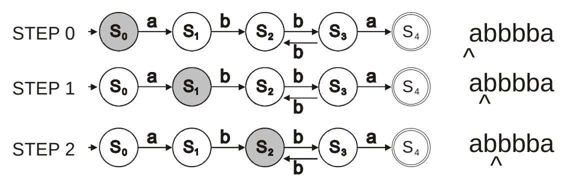
Regular expressions and finite state automata (state machines)

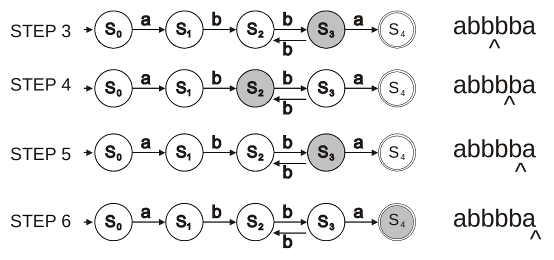
Regexp : a(bb)+a





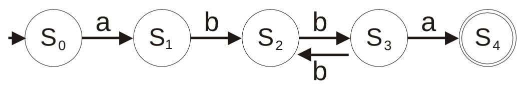
Regexp a(bb)+a Testing the string "abbbba"



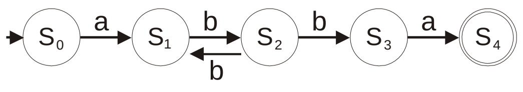


Deterministic and non-deterministic state machines

This machine (automaton) is ***deterministic*** because in any state each possible input character leads to at most *one* new state.



This machine is equivalent to the previous one, but is ***non-deterministic*** because in the state S2 it has multiple choices for b *:*



Applications of patterns/regexps

When are patterns useful in bioinformatics?

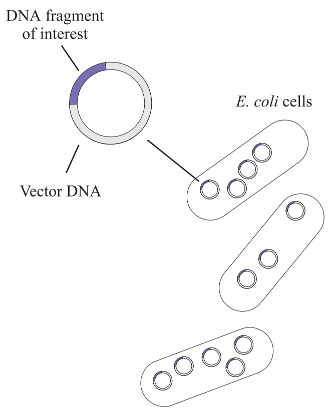
* Finding patterns in DNA and protein sequences
- Restriction enzyme sites (DNA)
- Transcription factor binding sites (DNA)
- PROSITE, using patterns to infer biological function from protein sequences

* Finding patterns to extract information from text files of bioinformatics applications, such as GenBank reports and BLAST output.

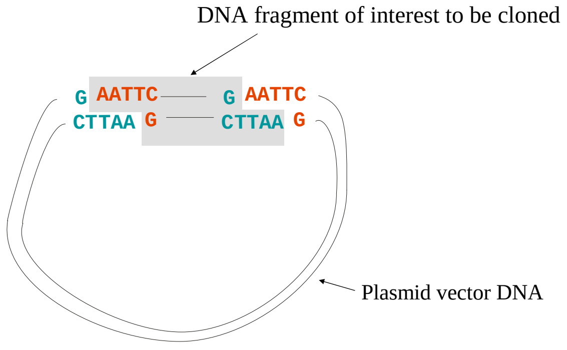
Recognition sites of restriction enzymes

```
EcoRI -GAATTC-
      -CTTAAG-

BamHI -GGATCC-
      -CCTAGG-

XhoII -RGATCY-    R = A or G
      -YCTAGR-    Y = C or T

PpuMI -RGGWCCY-   W = A or T
      -YCCWGGR-

FokI -GGATGNNNNNNNNNNNNNNN-
     -CCTACNNNNNNNNNNNNNNN-
```

Cloning

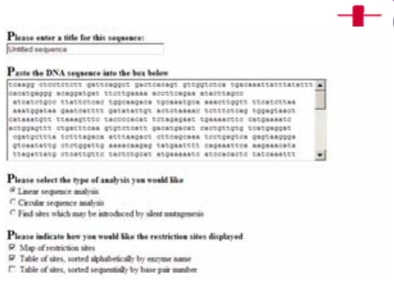


Restriction enzymes are used in cloning



Cutting in silico

http://rna.lundberg.gu.se/cutter2/



Please enter a title for this sequence:

Untitled sequence

Paste the DNA sequence into the box below

```
tcaagg ctcctctctt gattcaggct gactcacagt gttggtctca tgacaaattatttatattt
cacatgaggg acaggatgat ttcttgaaaa accttcagaa atacttagcc
 atcatctgcc ttattctcac tggcaagaca tgcaaatgca aaacttggtt ttcatcttaa
 aaatggatas gaatcatttt gatatattgt actctaaaac tctttctcag tggagtaact
cataaatgtt ttaaagtttc taccccacat tctagagaat tgaaaacttc catgaaaatc
actggagttt ctgatttcaa gtgtctcatt gacatgacat cactgttgtg tcatgaggat
 cgatgcttta tctttagaca atttaagact cttcagcaaa tcctgagtca gagtaaggga
 gtcaatattg ctctggattg aaaacaagag tatgaatttt cagaaattca aagaaacata
 ttagattatg ctcattgttc tactctgcat atgaaaaatc atcccactc tatcaaattt
```

Please select the type of analysis you would like

[✓ Linear sequence analysis]
Circular sequence analysis
Find sites which may be introduced by silent mutagenesis

Please indicate how you would like the restriction sites displayed

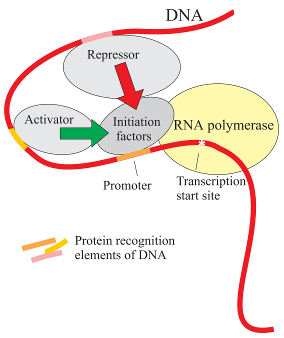
[✓ Map of restriction sites]
[✓ Table of sites, sorted alphabetically by enzyme name]
Table of sites, sorted sequentially by base pair number

Web-cutter output

```
                                            Eco31I   PshBI
            BseRI                          BspHI     AsnI
tcaaggctcctctcttgattcaggctgactcacagtgttggtctcatgacaaattaatttatatttcacatgagg base pairs
agttccgaggagagaactaagtccgactgagtgtcacaaccagagtactgtttaattaaatataaagtgtactcc 1 to 75
                                           RcaI      VspI
                                            BsaI     AseI



                             Eco57I                                       NspI
gacaggatgatttcttgaaaaaccttcagaaatacttagccatcatctgccttattctcactggcaagacatgca base pairs
ctgtcctactaaagaactttttggaagtctttatgaatcggtagtagacggaataagagtgaccgttctgtacgt 76 to 150
                                                                           MslI
```

Transcription factor databases contain information about DNA sequences recognized by transcription factors

Regulation of transcription



Transcription factor binding sites - example:

Estrogen response element

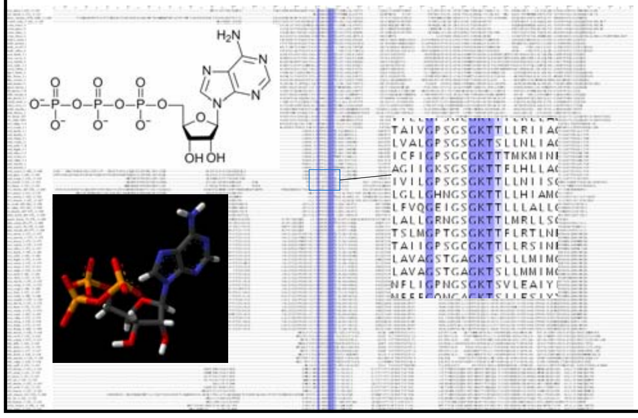
5' **AGGTCA**NNN**TGACCT**

Transcription factor databases contain information about DNA sequences recognized by transcription factors

```
Name                         Sequence                                           Trans. Factor
------------------------    ---------------------------------------------      -------------
UAS(G)-pMH100              0 CGGAGTACTGTCCTCCG                                  0 ! GAL4
TFIIIC-Xls-5S.1            0 TGGATGGGAG                                         0 ! TFIIIC
HSE_CS_inverted_repeat     0 CTNGAANNTTCNAG                                     0 ! HSTF
ZDNA_CS                    0 GCGTGTGCA                                          0 ! unknown
GCN4-his3-189              0 ATGACTCAT                                          0 ! GCN4
Ad5_EIA_element_I          0 AGGAAGTGAAA                                        0 ! unknown
Ad5_EIA_element_II         0 GGGCGTAACCGAGTAAGATTTGGCCATTTTC                    0 ! unknown
BPV-E2_CS1                 0 ACCGNNNNCGGT                                       0 ! (BPV-E2)
Alb_DEI                    0 GATTTTGTAATGG                                      0 ! C/EBP
Alb_DEII                   0 TTTTTGGCAAAGAT                                     0 ! CTF/NF-1
Alb_DEIII                  0 GCAAGGGATTTAGTT                                    0 ! unknown
Alb_PEI                    0 TGGTTAATGATCTACAGT                                 0 ! APF/HNF1
BPV-E2_CS2                 0 ACCNNNNNNGGT                                       0 ! BPV-E2
CACA                       0 CACACACACA                                         0 ! unknown
dc_box                     0 TSATTTGCAT                                         0 ! unknown
GALV_E                     0 AGAAATAGATGAGTCAACAG                               0 ! unknown
GCRE                       0 TGACTC                                             0 ! GCN4
Pit-1_CS1                  0 WTATYCAT                                           0 ! Pit-1
```

Protein patterns and the PROSITE database

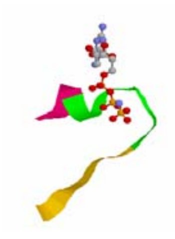
Proteins that bind the nucleotides ATP or GTP share a short sequence motif



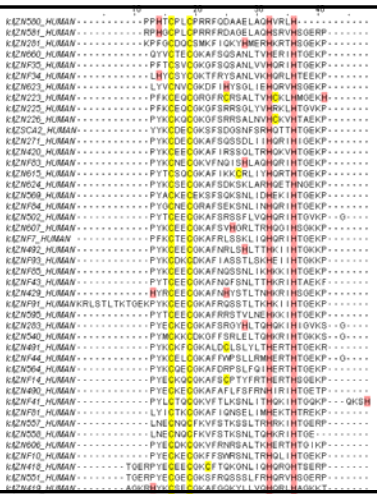
Entry in PROSITE for the ATP/GTP binding site motif

```
ID   ATP_GTP_A; PATTERN.
AC   PS00017;
DT   APR-1990 (CREATED); APR-1990 (DATA UPDATE); NOV-1990 (INFO
UPDATE).
DE   ATP/GTP-binding site motif A (P-loop).
PA   [AG]-x(4)-G-K-[ST].
CC   /TAXO-RANGE=ABEPV;
3D   1EFM; 1ETU; 1Q21; 2Q21; 4Q21; 5Q21; 6Q21;
DO   PDOC00017;
```

[AG]-x(4)-G-K-[ST]



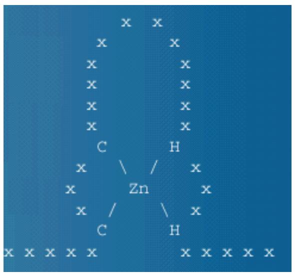
More PROSITE: a category of zinc finger proteins



```
lcl|ZN580_HUMAN------------PPHTCPLCPRRFQDAAELAQHVRLH-----------
lcl|ZN581_HUMAN------------RPHGCPLCPRRFRDAGELAQHSRVHSGERP-------
lcl|ZN281_HUMAN------------KPFGCDQCSMKFIQKYHMERHKRTHSGEKP-------
lcl|ZN660_HUMAN-------------QYVCTECGKAFSQSANLTVHERIHTGEKP-------
lcl|ZNF35_HUMAN-------------PFTCSVCGKGFSQSANLVVHQRIHTGEKP-------
lcl|ZNF34_HUMAN-------------LHYCSYCGKTFRYSANLVKHQRLHTEEKP-------
lcl|ZN623_HUMAN-------------LYVCNVCGKDFIHYSGLIEHQRVHSGEKP-------
lcl|ZN223_HUMAN-------------PFKCEQCGRGFRCRSALTVHCKLHMGEKH-------
lcl|ZN225_HUMAN-------------PFKCEQCGKGFSRRSGLYVHRKLHTGVKP-------
lcl|ZN226_HUMAN-------------PYKCKQCGKGFSRRSALNVHCKVHTAEKP-------
lcl|ZSCA2_HUMAN-------------YYKCDECGKSFSDGSNFSRHQTTHTGEKP-------
lcl|ZN271_HUMAN-------------PYKCDECGKAFSQSSDLIIHQRIHIGEKP-------
lcl|ZN420_HUMAN-------------PYKCEECGKAFIRSSQLTRHQKVHTGEKP-------
lcl|ZNF83_HUMAN-------------PYKCNECGKVFNQISHLAQHQRIHTGEKP-------
lcl|ZN615_HUMAN-------------PYTCSQCGKAFIKKCRLIYHQRTHTGEKP-------
lcl|ZN624_HUMAN-------------PYKCSECGKAFSDKSKLARHQETHNGEKP-------
lcl|ZN569_HUMAN-------------PYACKECEKSFSQKSNLIDHEKIHTGEKP-------
lcl|ZNF84_HUMAN-------------PYGCNECGRAFSEKSNLINHQRIHTGEKP-------
lcl|ZN502_HUMAN-------------PYTCEECGKAFSRSSFLVQHQRIHTGVKP--G----
lcl|ZN607_HUMAN-------------PYKCEECGKAFSVHGRLTRHQGIHSGKKP-------
lcl|ZNF7_HUMAN -------------PFKCTECGKAFRLSSKLIQHQRIHTGEKP-------
lcl|ZN492_HUMAN-------------PYKCEECGKAFNRLSHLTTHKIIHTGKKP-------
lcl|ZNF93_HUMAN-------------PYKCDKCDKAFIASSTLSKHEIIHTGKKP-------
lcl|ZNF85_HUMAN-------------PYKCEECGKAFNQSSNLIKHKKIHTGEKP-------
lcl|ZNF43_HUMAN-------------PYTCEECGKAFNQFSNLTTHKRIHTAEKF-------
lcl|ZN429_HUMAN-------------HYRCEECGKAFNHYSTLTNHKRIHSGEKP-------
lcl|ZNF91_HUMANKRLSTLTKTGEKPYKCEECGKAFRQSSTLTKHKIIHTGEKP-------
lcl|ZN595_HUMAN-------------PYTCEECGKAFRRSTVLNEHKKIHTGEKP-------
lcl|ZN283_HUMAN-------------PYECKECGKAFSRGYHLTQHQKIHIGVKS--G----
lcl|ZN540_HUMAN-------------PYMCKKCDKGFFSRLELTQHKRIHTGKKS--G----
lcl|ZN491_HUMAN-------------PYKCKFCGKALDCLSLYLTHERTHTGEKR-------
lcl|ZNF44_HUMAN-------------PYKCELCGKAFFWPSLLRMHERTHTGEKP--G----
lcl|ZN564_HUMAN-------------PYKCQECGKAFDRPSLFQIHERTHTGEKP-------
lcl|ZNF14_HUMAN-------------PYECKQCGKAFSCPTYFRTHERTHSGEKP-------
lcl|ZN490_HUMAN-------------PYECKECGKAFAFLFSFRNHIRIHTGETP-------
lcl|ZNF41_HUMAN-------------PYLCTQCGKVFTLKSNLITHQKIHTGQKP---QKSH
lcl|ZNF81_HUMAN-------------LYICTKCGKAFIQNSELIMHEKTHTREKP-------
lcl|ZN557_HUMAN-------------LNECNQCFKVFSTKSSLTRHRKIHTGERP-------
lcl|ZN558_HUMAN-------------LNECNQCFKVFSTKSNLTQHKRIHTGE---------
lcl|ZN606_HUMAN-------------PYECDKCGKVFRNRSALTKHERTHTGIKP-------
lcl|ZNF10_HUMAN-------------PYECKECGKFFSWRSNLTRHQLIHTGEKP-------
lcl|ZN418_HUMAN---------TGERPYECEECGKCFTQKGNLIQHQRGHTSERP-------
lcl|ZN551_HUMAN---------TGERPYECGECGKSFRQSSSLFRHQRVHSGERP-------
lcl|ZN419_HUMAN---------AGKRHYKCSECGKAFGQKYLLVQHQRLHAGKKT-------
```

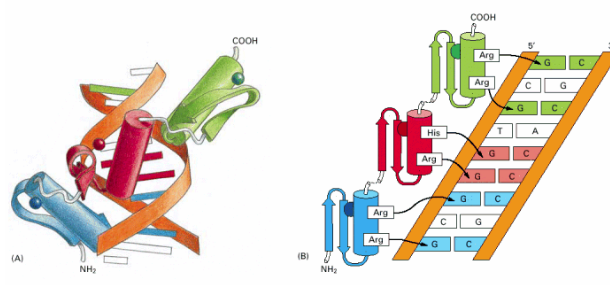
PROSITE entry of C2H2 type of zinc finger protein

```
ID   ZINC_FINGER_C2H2; PATTERN.
AC   PS00028;
DT   APR-1990 (CREATED); JUN-1994 (DATA UPDATE); NOV-1997 (INFO UPDATE).
DE   Zinc finger, C2H2 type, domain.
PA   C-x(2,4)-C-x(3)-[LIVMFYWC]-x(8)-H-x(3,5)-H.
NR   /RELEASE=35,69113;
```



```
           x  x
         x      x
        x        x
        x        x
        x        x
        x        x
         C      H
       x  \   /   x
      x     Zn     x
       x  /    \  x
         C      H
x x x x x          x x x x x
```

Recognition of DNA by zinc-finger binding protein

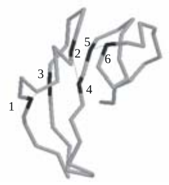


Zinc finger binding motifs are extremely common.
Among 23000 human genes, 845 have C2H2 zinc finger domains !!

More PROSITE: Disulfide bonds in EGF domain

- First identified in epidermial growth factor, but present in many other proteins
- Function unclear, but found in extracellular part of membrane proteins
- Includes six cysteins involved in disulfide bonds

```
                2                          4     5                            6
                +-----------------+     +-------------------+
                |                          |     |                            |
C-x(0,48)-C-x(3,12)-C-x(1,70)-C-x(1,6)-C-x2-G-x(0,21)-G-x2-C
|                          |
+-----------------+
1                          3
```



PROSITE

ExPASy Home page
Site Map
Search ExPASy
Contact us
Hosted by SIB Switzerland | Mirror sites: Australia | Brazil | Canada | China | Korea
Search PROSITE for
Go
Clear

PROSITE
Database of protein families and domains

http://www.expasy.org/prosite/ (Swiss Institute of Bioinformatics)

Release 20.67, of 05-Oct-2010 (contains
1308 *patterns* and 909 *profiles/matrices*).

A - x(4,7) - G -x (5,6) - D

"Non-probabilistic"

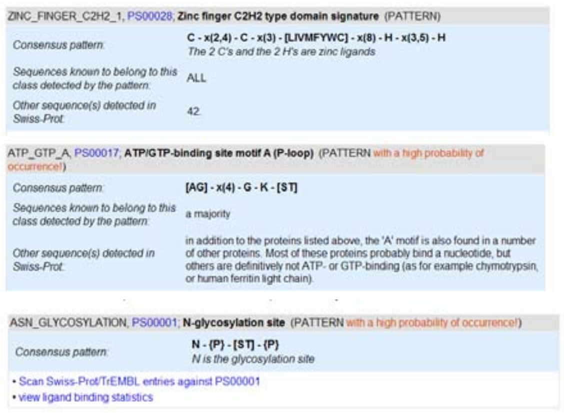
```
A  10    0   0   10  5  4
K  0     0   0   0   0  0
G  0     0   0   0   1  4
L  0     10  10  0   4  2
.. etc.
```

"Probabilistic"

PROSITE syntax

		Perl
D	matches D	
[AS]	matches A or S	
x	any symbol (amino acid)	.
x(3) or x3	any three symbols (amino acids)	.{3}
{AT}	matches any symbol *except* A and T	[^AT]
[AS]2	Matches two positions with either A or S (i.e. AA, AS , SS, or SA)	[AS]{2}
x(3,7)	a sequence of symbols between 3 and 7 in length (for instance GAT, GCRE, PPLKM, GTTREC or PPPPPPP)	.{3,7}

Variation in classification accuracy



ZINC_FINGER_C2H2_1, PS00028; **Zinc finger C2H2 type domain signature** (PATTERN)

Consensus pattern:	**C - x(2,4) - C - x(3) - [LIVMFYWC] - x(8) - H - x(3,5) - H** *The 2 C's and the 2 H's are zinc ligands*
Sequences known to belong to this class detected by the pattern:	ALL
Other sequence(s) detected in Swiss-Prot:	42

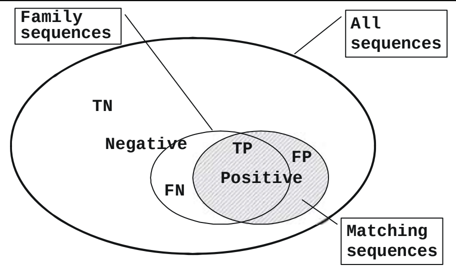
ATP_GTP_A, PS00017; **ATP/GTP-binding site motif A (P-loop)** (PATTERN with a high probability of occurrence!)

Consensus pattern:	**[AG] - x(4) - G - K - [ST]**
Sequences known to belong to this class detected by the pattern:	a majority
Other sequence(s) detected in Swiss-Prot:	in addition to the proteins listed above, the 'A' motif is also found in a number of other proteins. Most of these proteins probably bind a nucleotide, but others are definitively not ATP- or GTP-binding (as for example chymotrypsin, or human ferritin light chain).

ASN_GLYCOSYLATION, PS00001; **N-glycosylation site** (PATTERN with a high probability of occurrence!)

Consensus pattern:	**N - {P} - [ST] - {P}** *N is the glycosylation site*

- Scan Swiss-Prot/TrEMBL entries against PS00001
- view ligand binding statistics



* True positive - member of the protein family that is matched by the pattern

* False positive - a protein which is *not* a member of the protein family is matched by the pattern

* True negative - a protein which is *not* a member of the protein family is *not* matched by the pattern

* False negative - member of the protein family is *not* matched by pattern

Classification accuracy

Specificity : $\frac{TP}{(TP+FP)}$

Sensitivity : $\frac{TP}{(TP+FN)}$

- Ex: The flavodoxin family (PS00201)

True positives:	51
False positives:	4
False negatives:	9

$\frac{51}{(51+4)}$ » 0.93

$\frac{51}{(51+9)}$ » 0.85

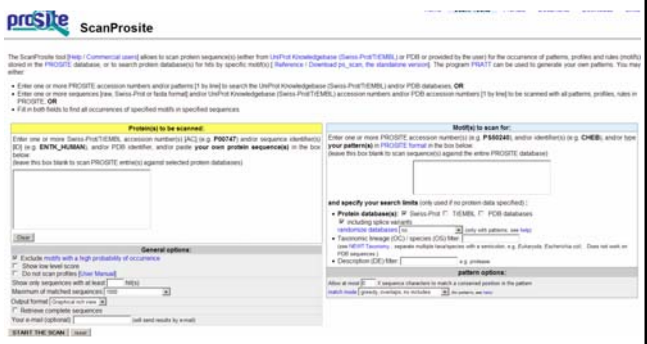
Important application of databases like PROSITE

Biological information may be extracted from a protein sequence.

A ‘new’ protein sequence has been identified using bioinformatics methods (like in a genome project). A scan of PROSITE using this sequence (regular expression matching) can give important clues as to the biological function of the protein.

Common applications of PROSITE (ScanProsite/Motifscan)

* Search a sequence for all PROSITE patterns/profiles
* Search database with a pattern/profile



ScanProsite

The ScanProsite tool [Help / Commercial users] allows to scan protein sequence(s) (either from UniProt Knowledgebase (Swiss-Prot/TrEMBL) or PDB or provided by the user) for the occurrence of patterns, profiles and rules (motifs) stored in the PROSITE database, or to search protein database(s) for hits by specific motif(s) [Reference / Download ps_scan, the standalone version]. The program PRATT can be used to generate your own patterns. You may either:

- Enter one or more PROSITE accession numbers and/or patterns [1 by line] to search the UniProt Knowledgebase (Swiss-Prot/TrEMBL) and/or PDB databases, **OR**
- Enter one or more sequences [raw, Swiss-Prot or fasta format] and/or UniProt Knowledgebase (Swiss-Prot/TrEMBL) accession numbers and/or PDB accession numbers [1 by line] to be scanned with all patterns, profiles, rules in PROSITE, **OR**
- Fill in both fields to find all occurrences of specified motifs in specified sequences.

Protein(s) to be scanned:

Enter one or more Swiss-Prot/TrEMBL accession number(s) [AC] (e.g. **P00747**) and/or sequence identifier(s) [ID] (e.g. **ENTH_HUMAN**), and/or PDB identifier, and/or paste **your own protein sequence(s)** in the box below
(leave this box blank to scan PROSITE entrie(s) against selected protein databases)

Clear

General options:

- [✓] Exclude motifs with a high probability of occurrence
- [ ] Show low level score
- [ ] Do not scan profiles [User Manual]

Show only sequences with at least [ ] hit(s)
Maximum of matched sequences: 1000
Output format: Graphical rich view
[ ] Retrieve complete sequences
Your e-mail (optional): [ ] (will send results by e-mail)

START THE SCAN | reset

Motif(s) to scan for:

Enter one or more PROSITE accession number(s) (e.g. **PS00240**), and/or identifier(s) (e.g. **CHEB**), and/or type **your pattern(s)** in PROSITE format in the box below
(leave this box blank to scan sequence(s) against the entire PROSITE database)

and specify your search limits (only used if no protein data specified) :

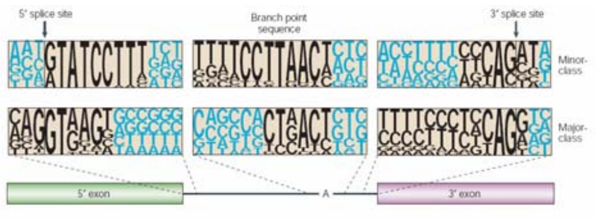
- **Protein database(s):** [✓] Swiss-Prot [ ] TrEMBL [ ] PDB databases
  [✓] including splice variants
  randomize databases: no (only with patterns, see help)
- Taxonomic lineage (OC) / species (OS) filter: [ ]
  (see NEWT Taxonomy ; separate multiple taxa/species with a semicolon, e.g. Eukaryota; Escherichia coli ; Does not work on PDB sequences)
- Description (DE) filter: [ ] e.g. protease

pattern options:

Allow at most [0] X sequence characters to match a conserved position in the pattern
match mode: greedy, overlaps, no includes (for patterns, see help)

More applications of patterns ...

Finding minor introns

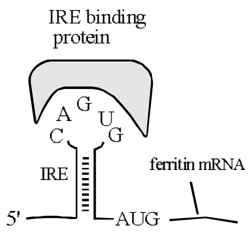


A class of U12 (minor) introns are highly conserved in sequence:

```
   RT|**AT**CCTYT................TTCCTTTR......**AC|**Y
exon |  5'          intron                    3'| exon

[AG]TATCCT[CT]T.{30,1500}TTCCTT[AG]......AC[TC]
```

More applications of patterns ...

Finding the iron responsive element



CAGUG[CU].{10,200}AUG

... and finding ORFs

```
AUG(...){0,200}?(UAA|UAG|UGA)
```

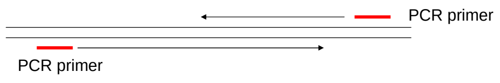
match as few characters as possible

More applications of patterns ...

When BLAST will fail to identify regions of identity in a genome

For instance: Find occurrences of 'AGCTGCAAAAA'

To find such short regions of identity BLAST is often difficult to use. There are however applications when you want to find such matches, like when searching for matches to oligonucleotides used in PCR or probes used in microarrays. Or you may want to search for matches to very short peptide sequences potentially encoded by the genome.



More applications of patterns ...

Repeats ...

like in the Huntingtin protein

---CAGCAGCAGCAGCAGCAGCAGCAGCAGCAGCAGCAGCAGCAG---

```
(...)\1{8,20}
```

stored in memory

recall what is in memory

Regular expressions are used in ...

* Patterns as used in PROSITE

* programming and scripting languages, for instance *perl*

* operating system tools , for instance *grep*

* text editors

* EMBOSS program *fuzzpro*

UNIX utility grep / egrep

```
egrep "[AG].{4}GK[TS]" /dbs/swissprot
```

will print all lines containing matching the pattern
corresponding to the ATP/GTP binding site motif in PROSITE

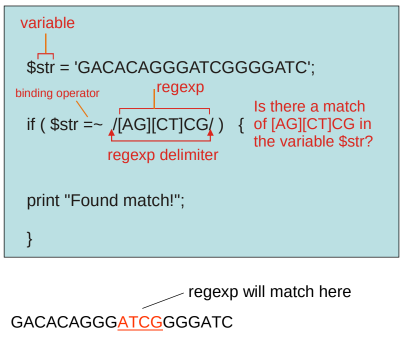
/dbs/swissprot is the Swissprot protein sequence database
in FASTA format

Sample session with 'fuzzpro'

```
% fuzzpro
Protein pattern search
Input sequence(s): tsw:*
Search pattern: [FY]-[LIV]-G-[DE]-E-A-Q-x-[RKQ](2)-G
Number of mismatches [0]:
Output report [100k_rat.fuzzpro]:
```

Perl

Example code that uses a regular expression:

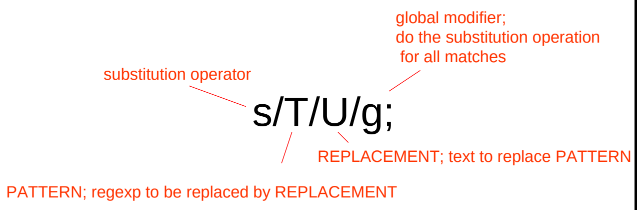


```
$str = 'GACACAGGGATCGGGGATC';
if ( $str =~ /[AG][CT]CG/ )  {
print "Found match!";
}
```

The substitution and transliteration operators

Transcribing DNA into RNA.

```
$dna = 'GCAATGG';
print "The DNA sequence is $dna\n";
$rna = $dna;
$rna =~ s/T/U/g;
print "and the RNA sequence is $rna\n";
```



global modifier; do the substitution operation for all matches
substitution operator
s/T/U/g;
REPLACEMENT; text to replace PATTERN
PATTERN; regexp to be replaced by REPLACEMENT

The substitution and transliteration operators

Count the bases in a DNA sequence using tr

```
$dna = 'GCAATGNGATTACTTCG';

$basecount = ($dna =~ tr/ACGT//;)
$nonbase = length($dna)-$basecount;
print "There are $basecount As,Cs,Ts,and Gs \n";
print "There is/are $nonbase other symbol(s) \n";
```

`tr/ACGT//;`

operation does not change the string!

Metacharacters

.	match any character
^	match beginning of string
$	match end of string
?	optional match

Quantifiers

*	Any number of characters, including zero
+	One or more characters
{m,n}	minimum m , maximum n characters

Reading a file in PERL and checking for regular expresssions

Program will read the file with the name 'myfile' and that has a sequence in EMBL format. The program will print to the screen all lines starting with 'FT', i.e lines with feature table information.

```
open IN, 'myfile';

while (<IN>) {

if (/^FT/) {print ;}

}
close IN;
```

Example of how regexps may be used to parse the output from a BLAST search

'bl2seq' is a program to blast two sequences against each other.
This is an output using two sequences seq1 and seq2:

```
Query= seq1
         (31 letters)
>seq2
         Length = 29

 Score = 42.1 bits (21), Expect = 1e-10
 Identities = 27/29 (93%)
 Strand = Plus / Plus

Query: 3  acgacgtacacgactagtcaggcggagct 31
          ||||||| |||||||||||| ||||||||
Sbjct: 1  acgacgttcacgactagtcacgcggagct 29

.... (and some more text ) ......
```

Example of how regexps may be used to parse the output from a BLAST search

Let's say you have many output files like this and you want to make a script to present the names of the two sequences as well as the Expect value listed. Here's an example of code: (we assume that the blast output is in a file called 'bl2seq.out')

```
open IN, 'bl2seq.out';
while (<IN>) {  # we read one line at a time, each
         # line is stored in the default variable $_)
chomp; # remove the end of line character in $_
if (/Query= (.*)/) {print "$1 "; }
if (/^>(.*)/) {print "$1 ";}
if (/Expect = (.*)/) {print "$1\n";}
}
close IN;
```

.* means any number (including zero) of any characters;
the regexp algorithm is ***greedy*** so it will try to find the longest substring that matches

Regular expression match basics

1. Quantifiers * + ? {m,n} are *greedy*

Perl example:

```
$str = "GGAAGG";
$str =~ s/(G.*G)//;
```

$str is now empty

2. The match that begins earliest (leftmost) wins

Perl example:

```
$str = "GGAAGG";
$str =~ s/GG/XX/;
print $str;
```

$str is now XXAAGG