Software Manual

Institute of Bioinformatics, Johannes Kepler University Linz

PrOCoil — A Web Service and an R Package for Predicting the Oligomerization of Coiled Coil Proteins

Ulrich Bodenhofer

Institute of Bioinformatics, Johannes Kepler University Linz
Altenberger Str. 69, 4040 Linz, Austria
procoil@bioinf.jku.at

Version 1.16.0, March 21, 2014

Scope and Purpose of this Document

This document is a user manual for PrOCoil, the software suite accompanying the paper [10]. It provides a gentle introduction into how to use PrOCoil. Not all features of the R package are described in full detail. Such details can be obtained from the documentation enclosed in the R package. Further note the following: (1) this is not an introduction to coiled coil proteins; (2) this is not an introduction to R; (3) this is not an introduction to support vector machines. If you lack the background for understanding this manual, you first have to read introductory literature on the subjects mentioned above.

Contents

1 Introduction

This user manual describes the PrOCoil software suite that accompanies the paper [10]. This software is concerned with analyzing coiled coil sequences in terms of their oligomerization behavior. PrOCoil does not only provide a prediction, but also detailed insights into which residues or sub-sequences are responsible for the predicted oligomerization.

PrOCoil is available both as an easy-to-use Web interface and an R package. Both variants offer the same prediction and analysis facilities. The following table summarizes the essential differences:

PrOCoil Web interface	PrOCoil R package
can be used instantly on any computer with Internet access and a Web browser	requires R and installation of package `procoil`
supports only the standard PrOCoil model	supports the standard PrOCoil model and the alternative model optimized for balanced accuracy; other models can be loaded from files
graphics are produced in a non-customizable standard format	graphics are customizable
every sequence must be analyzed separately	analyses can be automated and run in batch mode through R's scripting environment
beside standard input (amino acid sequence + aligned heptad annotation), also PairCoil2 output format is supported; Marcoil output can be used upon a separate pre-processing step	only standard input (amino acid sequence + aligned heptad annotation)

We recommend beginners to use the Web interface. Experienced users can benefit from the greater flexibility of the R package. R package users can use the Web interface for converting PairCoil2 and/or Marcoil output into the input format the R package requires.

2 Input Data

As already mentioned, PrOCoil predicts whether a given coiled coil segment of an amino acid sequence is more likely to form a dimer or trimer. Such a segment has to consist of an amino acid (sub-)sequence and an aligned heptad annotation of the same length. As an example, the GCN4 yeast transcription factor is a dimer consisting of two equal sequences (i.e. it is a homo-dimer), the coiled coil parts of which (according to SOCKET [15]) look as follows:

```
MKQLEDKVEELLSKNYHLENEVARLKKLV
abcdefgabcdefgabcdefgabcdefga
```

The heptad annotation is essential for PrOCoil to work and cannot be omitted. The letters 'a'–'g' correspond to the usual annotation of positions within the heptad motif. PrOCoil can also process

full amino acid sequences containing one or more coiled coil segments as sub-sequences, where the coiled coil segments must be annotated with the usual symbols ‘a’–‘g’ and non-coiled coil residues must be annotated with dashes ‘-’. As an example, let us consider the sample sequence used in the Marcoil Web interface[1] [6] (annotation derived from running Marcoil on this sequence with a cut-off of 50%):

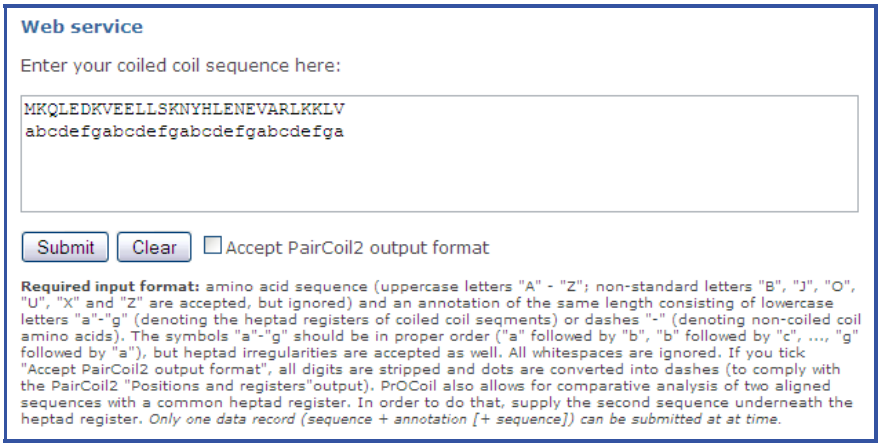
```
MGECDQLLVFMITSRVLVLSTLIIMDSRQVYLENLRQFAENLRQNIENVHSFLENLRADLENLRQKFPGKWYSAMPGRHG
------------------------------abcdefgabcdefgabcdefgabcdefgabcdefg--------------
```

If the heptad annotation contains dashes, PrOCoil extracts all coiled coil regions (contiguous non-dash regions) and classifies them separately. From the Marcoil sample sequence, PrOCoil will extract and classify the following coiled coil segment (residues 32–66 of the above example):

```
LENLRQFAENLRQNIENVHSFLENLRADLENLRQK
abcdefgabcdefgabcdefgabcdefgabcdefg
```

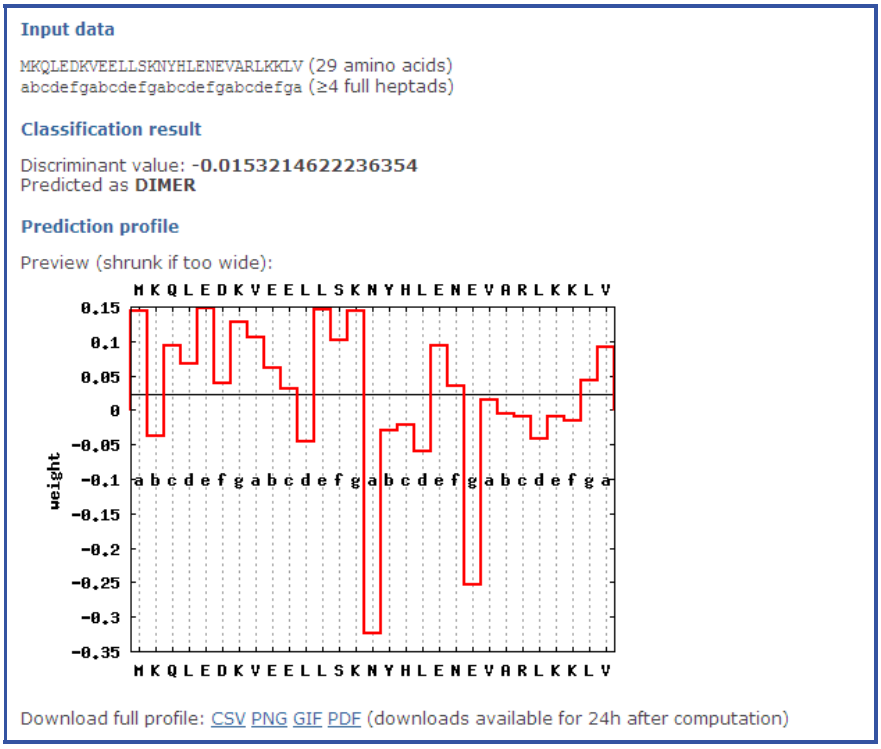
3 Predictions Using the Web Interface

The PrOCoil Web interface can be accessed directly via the PrOCoil homepage.[2] The page contains an input field that is pre-filled with the GCN4 wild type as an example:



The PrOCoil server first checks your input for validity (see Section 2). If the input is correct, all coiled coil segments are extracted from the input and analyzed separately. For the standard example, the GCN4 wild type, the output looks as follows:

[1] http://www.isrec.isb-sib.ch/webmarcoil/webmarcoilC1.html
[2] http://www.bioinf.jku.at/software/procoil/



The output of the PrOCoil Web interface is structured into three sections:

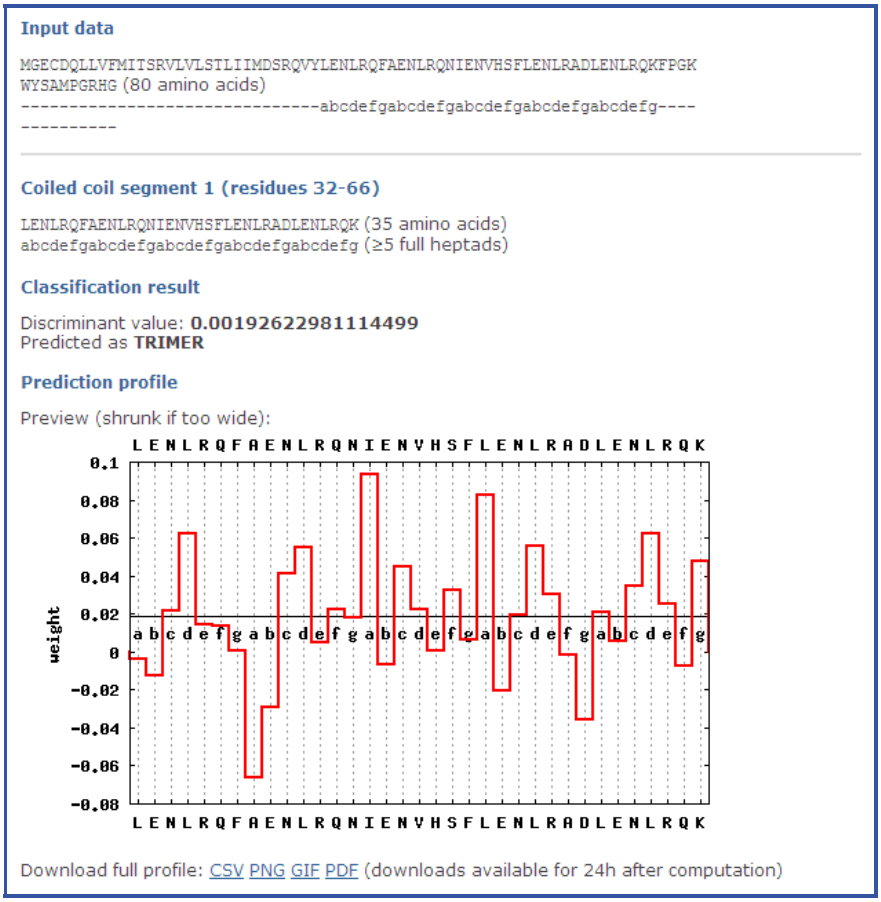
Input data: amino acid sequence and heptad annotation of the coiled coil segment;

Classification result: discriminant function value and final classification; a positive value means that the sequence is classified as trimer, a negative value means that the sequence is classified as dimer. The higher the absolute value, the clearer the oligomerization tendency is. The closer the value is to zero, the more the sequence can be considered a borderline case.

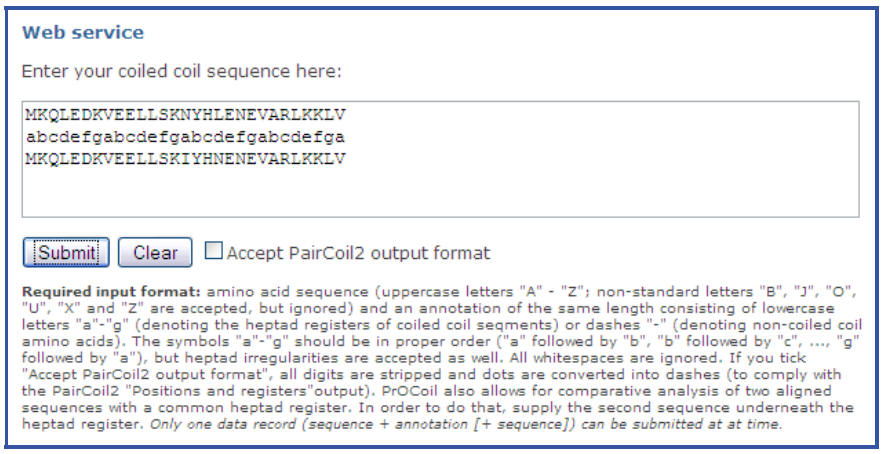
Prediction profile: this plot shows the contribution of each residue to the discriminant function value. The more positive a value is for a residue, the more this residue contributes to an oligomerization tendency towards trimers. The more negative the value is, the more this residue contributes to an oligomerization tendency towards dimers. The horizontal line corresponds to the base line of the classifier (see 6.3). The discriminant function value is actually obtained as the area above the grey baseline minus the area below the grey baseline. The links below the prediction profile plot allow for downloading the profile in various formats.

Note: Values in the prediction profile cannot be understood as general rules for which oligomerization behavior a given amino acid at a given heptad position is indicative for. PrOCoil takes pairwise interactions of amino acids into account. Therefore, the values in the prediction profile are always to be considered in the context of the given sequence. The same amino acid at the same heptad position might have a completely different value in the prediction profile of a different sequence.

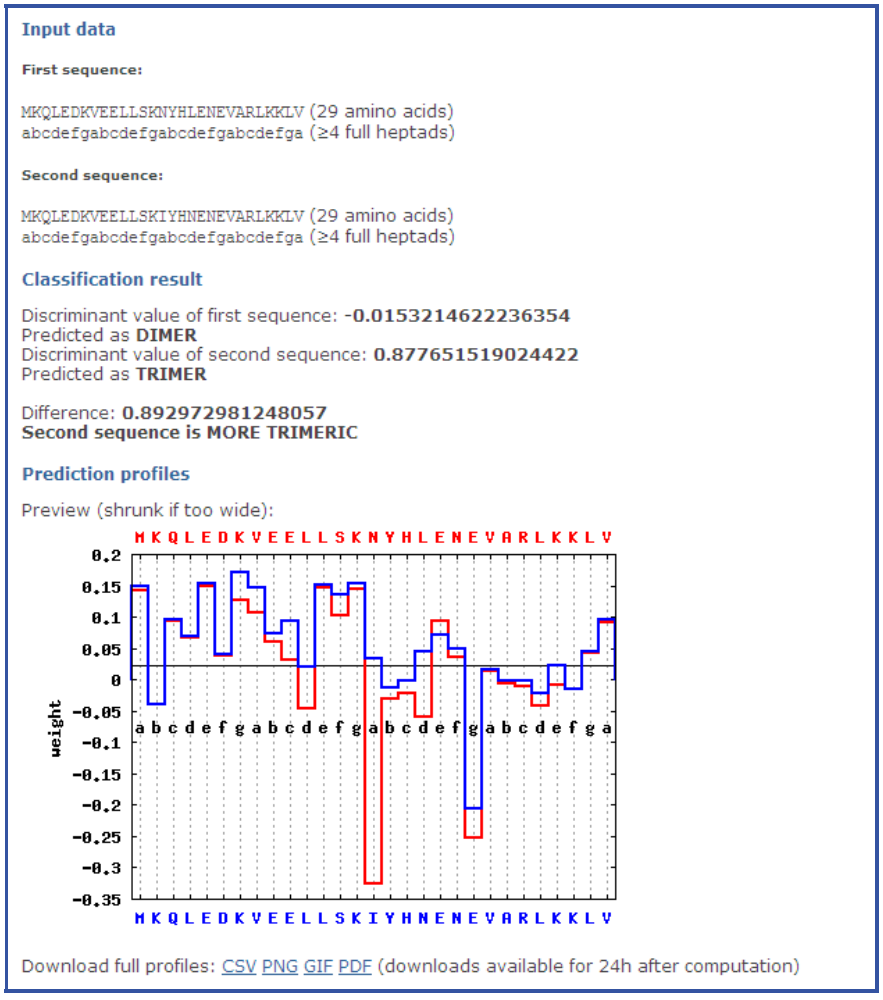
If the heptad annotation contains at least one dash '-', PrOCoil first extracts all coiled coil segments, i.e. all contiguous sub-sequences with no dashes in the heptad annotation. Then all these coiled coil segments are analyzed independently and the results are presented sequentially in the order they appear in the input sequence in the same format as above. The only notable difference is that the section "Input data" shows the complete input sequence, whereas the extracted coiled coil segment is displayed in a separate section "Coiled coil segment" along with its consecutive number and the corresponding start and end positions in the input sequence. For the Marcoil sample sequence, the output looks as follows:



The PrOCoil Web interface also facilitates easy analysis of mutations of coiled coil segments. This feature is limited to two sequences at a time. The two sequences must have the same length and heptad register. If these conditions are met, the second sequence can simply be written underneath the heptad register (with the first sequence remaining above the heptad register):



The output is structured as described for single sequences. The section “Classification result” shows discriminant function values and predictions of both sequences along with a comparison whether the second sequence is more dimeric or more trimeric than the first sequence. Only one profile plot is produced in which both profiles are visualized, the profile of the first sequence in red and the second sequence’s profile in blue:

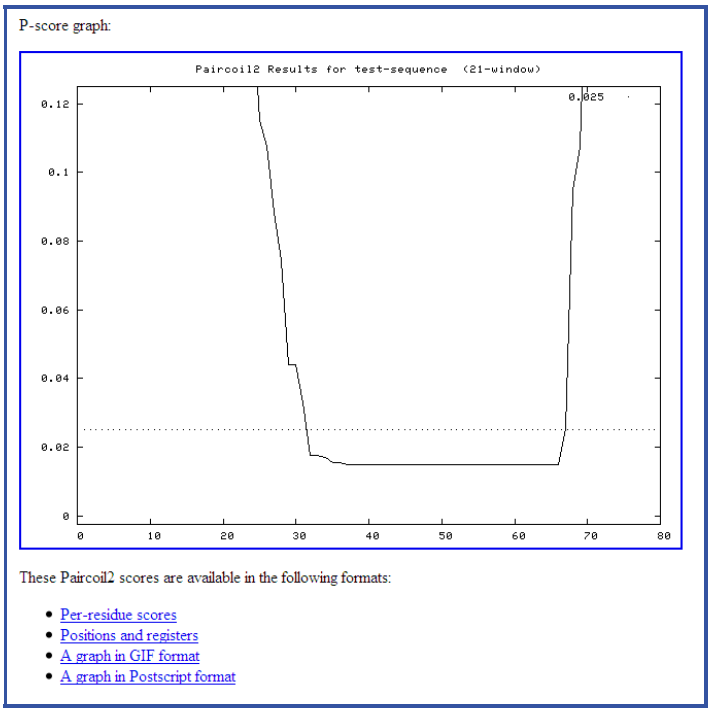


4 Preprocessing Predicted Coiled Coil Segments Using the Web interface

We expect PrOCoil to be used mainly for sequences the exact 3D structures of which are unknown (otherwise the correct oligomerization can be determined using SOCKET [15]). For structurally unresolved sequences, a coiled coil predictor must be used first to determine the coiled coil segments and their presumed heptad annotations. Currently the two most commonly used predictors for this task are **Marcoil** [6] and **PairCoil2** [1, 11]. Both programs have their own output formats that do not comply with the format described in Section 2. In order to ease integration with Marcoil and PairCoil2, the PrOCoil Web interface offers preprocessing tools that allow to use the outputs of Marcoil and PairCoil2 for processing with PrOCoil.

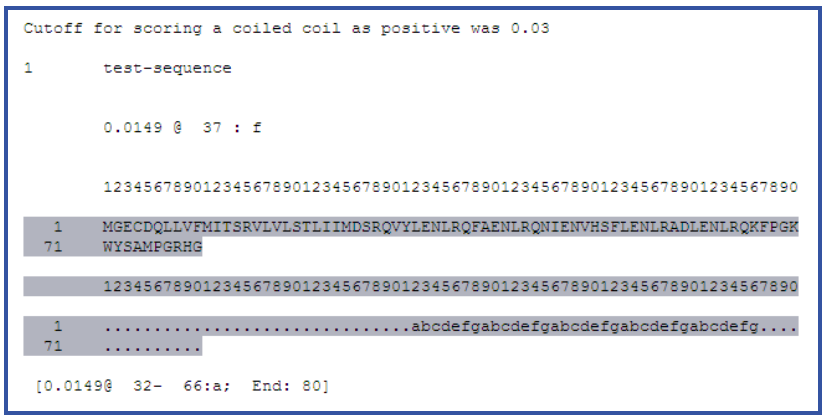
4.1 Processing PairCoil2 results

The PairCoil2 Web interface[3] produces the following kind of output for the Marcoil sample sequence:

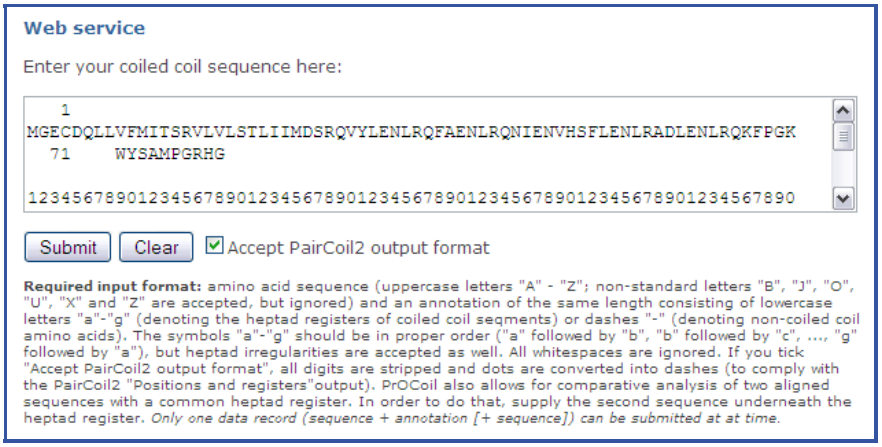


If you click the link “Positions and registers”, a text file with the predicted heptad annotation is displayed. Select the lines with the amino acids and heptad annotations in the following way:

[3] `http://groups.csail.mit.edu/cb/paircoil2/paircoil2.html`



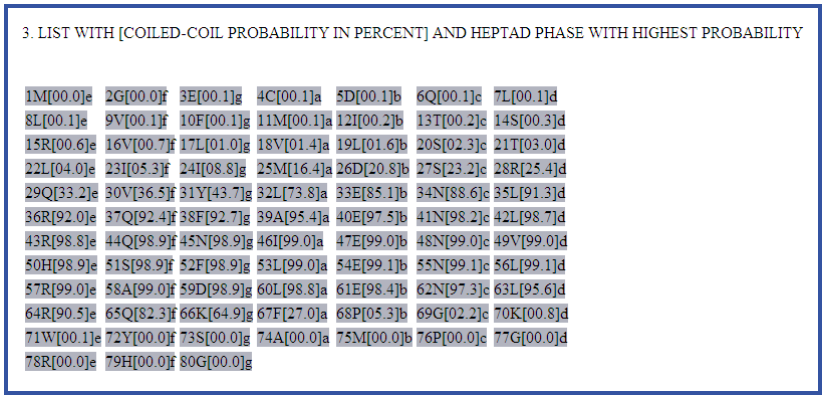
Now copy this selection into the input field of the PrOCoil Web interface. If you check “Accept PairCoil2 output format”, you can directly process the PairCoil2 output with PrOCoil:



4.2 Processing Marcoil results

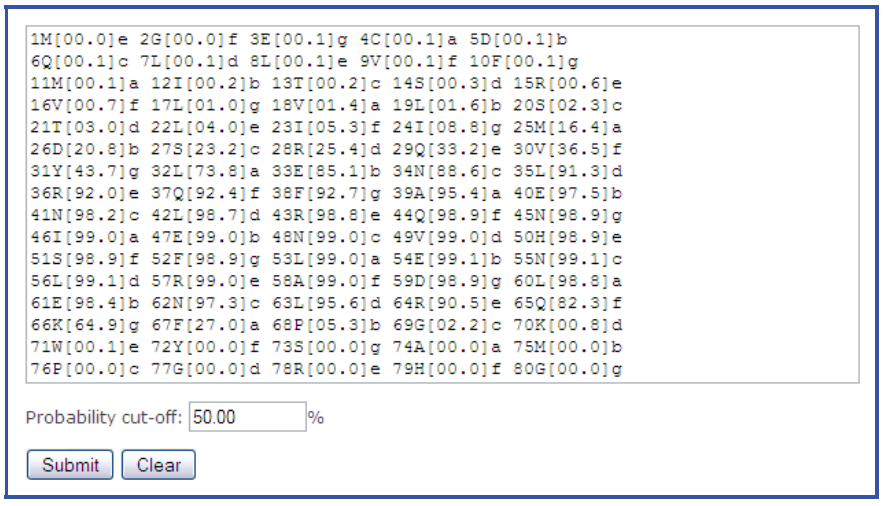
If you submit a sequence to the Marcoil Web interface,[4] the results are shown as an HTML text in your Web browser. Scroll down to the section “3. LIST WITH [COILED-COIL PROBABILITY IN PERCENT] AND HEPTAD PHASE WITH HIGHEST PROBABILITY ” and select the whole section (without the heading) as follows:

[4] http://www.isrec.isb-sib.ch/webmarcoil/webmarcoilC1.html

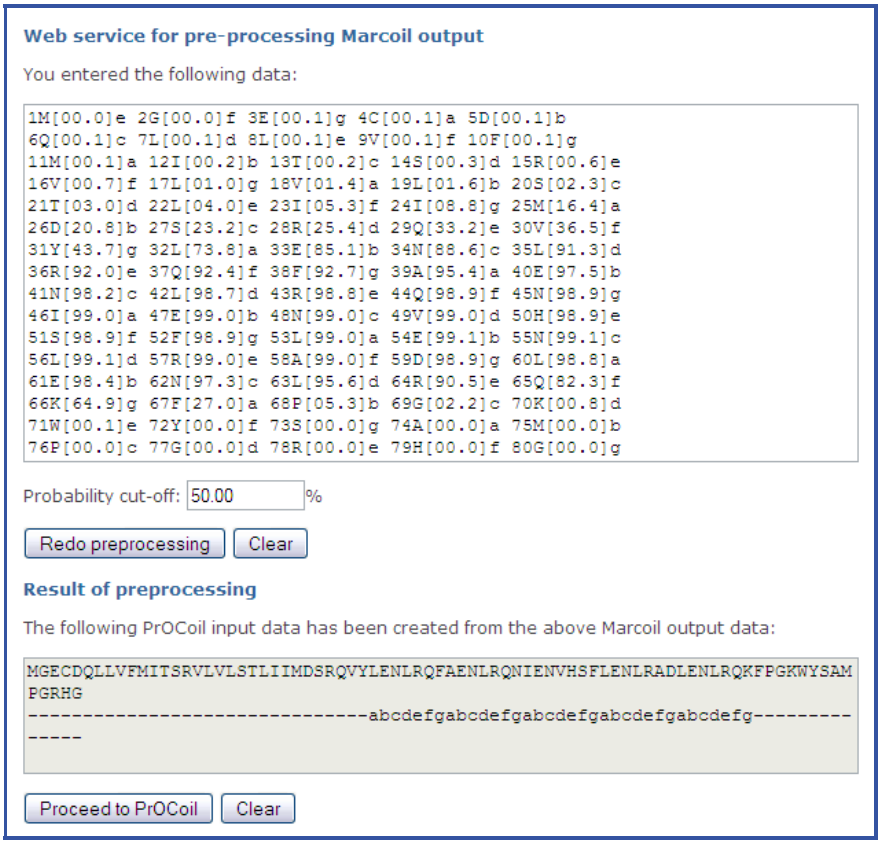

3. LIST WITH [COILED-COIL PROBABILITY IN PERCENT] AND HEPTAD PHASE WITH HIGHEST PROBABILITY

1M[00.0]e 2G[00.0]f 3E[00.1]g 4C[00.1]a 5D[00.1]b 6Q[00.1]c 7L[00.1]d
8L[00.1]e 9V[00.1]f 10F[00.1]g 11M[00.1]a 12I[00.2]b 13T[00.2]c 14S[00.3]d
15R[00.6]e 16V[00.7]f 17L[01.0]g 18V[01.4]a 19L[01.6]b 20S[02.3]c 21T[03.0]d
22L[04.0]e 23I[05.3]f 24I[08.8]g 25M[16.4]a 26D[20.8]b 27S[23.2]c 28R[25.4]d
29Q[33.2]e 30V[36.5]f 31Y[43.7]g 32L[73.8]a 33E[85.1]b 34N[88.6]c 35L[91.3]d
36R[92.0]e 37Q[92.4]f 38F[92.7]g 39A[95.4]a 40E[97.5]b 41N[98.2]c 42L[98.7]d
43R[98.8]e 44Q[98.9]f 45N[98.9]g 46I[99.0]a 47E[99.0]b 48N[99.0]c 49V[99.0]d
50H[98.9]e 51S[98.9]f 52F[98.9]g 53L[99.0]a 54E[99.1]b 55N[99.1]c 56L[99.1]d
57R[99.0]e 58A[99.0]f 59D[98.9]g 60L[98.8]a 61E[98.4]b 62N[97.3]c 63L[95.6]d
64R[90.5]e 65Q[82.3]f 66K[64.9]g 67F[27.0]a 68P[05.3]b 69G[02.2]c 70K[00.8]d
71W[00.1]e 72Y[00.0]f 73S[00.0]g 74A[00.0]a 75M[00.0]b 76P[00.0]c 77G[00.0]d
78R[00.0]e 79H[00.0]f 80G[00.0]g

These data cannot be processed by PrOCoil directly, since coiled coil segments must first be singled out by applying a probability cut-off threshold. For this purpose, a separate input field is available at the bottom of the PrOCoil Web page. Copy the selection into this field and choose a probability cut-off (by default 50%):



After clicking “Submit”, an output page is displayed that shows the result of applying the chosen probability cut-off in a read-only field at the bottom:



If you are satisfied with the result, you can directly pass the data to PrOCoil by clicking "Proceed to PrOCoil". If you think you should have used a different threshold, your input is displayed in the input field on top of the page. Select a different threshold until the result meets your expectation and then pass the data to PrOCoil.

5 PrOCoil R Package

5.1 Installation

The PrOCoil R package (current version: 1.16.0) is available via Bioconductor. The simplest way to install the package is the following:

```
> source("http://www.bioconductor.org/biocLite.R")
> biocLite("procoil")
```

If you wish to install the package manually instead, you can download the package archive that fits best to your computer system from the Bioconductor homepage.

5.2 Getting started

To load the PrOCoil package, enter

```
> library(procoil)
```

in your R session. If this command terminates without any error message or warning, you can be sure that the PrOCoil package has been installed successfully. If so, the PrOCoil package is ready for use now and you can start predicting the oligomerization of coiled coils.

As a first example, the following command makes a prediction for the GCN4 wild type already used above:

```
> GCN4wt <- predict(PrOCoilModel, "MKQLEDKVEELLSKNYHLENEVARLKKLV", "abcdefgabcdefgabcdefgal
```

The object `PrOCoilModel` is an object of class `CCModel` in which the PrOCoil model is stored. Since `predict` is a generic function that determines the function to call from the type of the first argument, it is essential that you provide an object of class `CCModel` as first argument of `predict`. For more information about `CCModel` objects, enter `help(CCModel)`.

The second argument is obviously the amino acid sequence and the third argument is the heptad annotation. The order of the two latter may be changed by naming arguments:

```
> GCN4wt <- predict(PrOCoilModel, reg="abcdefgabcdefgabcdefgabcdefga",
+                   seq="MKQLEDKVEELLSKNYHLENEVARLKKLV")
```

Note that the argument `seq` need not be a plain character string: anything that can be cast into a character string works, in particular, objects of class `BString` or `AAString` (see package `Biostrings` from Bioconductor[5]).

The model and the sequence are compulsory arguments. To supply a heptad register is compulsory too, but it can also be supplied by means of an attribute or `Biostrings` metadata (see 5.5.5).

[5] `http://www.bioconductor.org/`

As mentioned in Section 2, the sequence and the register annotation must have equal lengths. The function `predict` creates an object of class `CCProfile` in which prediction results along with various additional information is stored. To obtain more information about this class, enter `help(CCProfile)`. Basic information about the result can be displayed by `show(GCN4wt)` or simply by entering the name of the object:

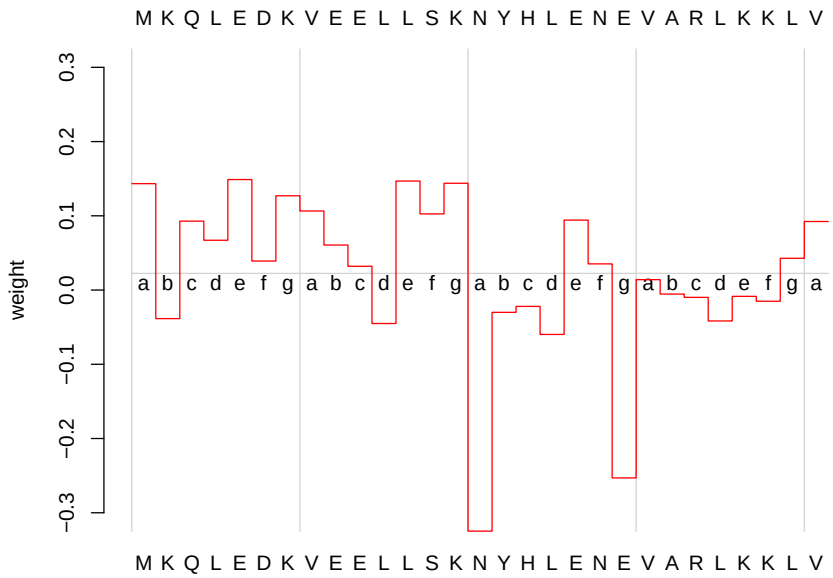
```
> GCN4wt
```

```
CCProfile object
        Sample: MKQLEDKVEELLSKNYHLENEVARLKKLV
                abcdefgabcdefgabcdefgabcdefga
        discriminant function value = -0.01532146
        predicted as dimer
```

The discriminant value and its interpretation are the same as described in Section 3 above.

A prediction profile can be plotted simply as follows:

```
> plot(GCN4wt)
```



The `plot` function for `CCProfile` objects provides various ways for customizing the plot and for writing directly to graphics files (see also 5.5.4 and the documentation available via `help("plot-methods")`).

5.3 Predictions for non-trimmed sequences containing coiled coil segments

Like the Web version described above, the R package is also capable of handling sequences that contain non-coiled-coil sub-sequences. If the heptad annotation contains at least one dash "-", `predict` first extracts all coiled coil segments, i.e. all contiguous sub-sequences with no dashes in the heptad annotation. Then all these coiled coil segments are analyzed independently and the results are returned as a list, the components of which are the prediction results of the coiled coil segments in the order they appear in the sequence. Let us consider the Marcoil sample sequence again:

```
> res <- predict(PrOCoilModel,
+ "MGECDQLLVFMITSRVLVLSTLIIMDSRQVYLENLRQFAENLRQNIENVHSFLENLRADLENLRQKFPGKWYSAMPGRHG",
+ "------------------------------abcdefgabcdefgabcdefgabcdefgabcdefg-------------")
```

The returned object `res` is a list of `CCProfile` objects:

```
> res
```

```
$`32_66`

CCProfile object
        Sample: LENLRQFAENLRQNIENVHSFLENLRADLENLRQK
                abcdefgabcdefgabcdefgabcdefgabcdefg
        discriminant function value = 0.00192623
        predicted as trimer
```

Each component of the returned list is named "s_e", where s and e are the start and end positions of the coiled coil segment in the original sequence. In the above example, therefore, '`32_66`' means that the coiled coil segment consists of residues 32–66 in the original sequence.

The prediction profile must be plotted for each coiled coil segment separately. This can be done by accessing the list components explicitly:

```
> plot(res[[1]])
```

Note: `predict` returns a list whenever it finds at least one dash in the heptad annotation, regardless of whether there is only one or more than one coiled coil segment in the data.

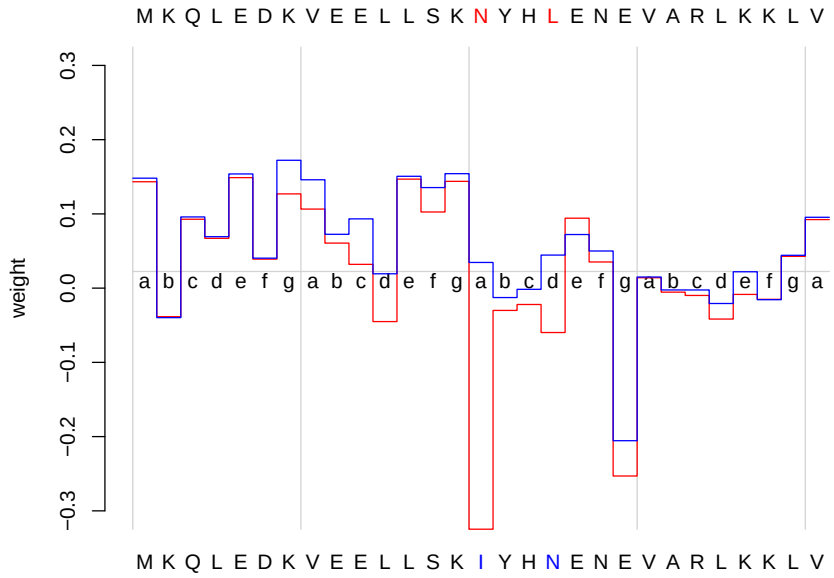
5.4 Comparative mutation analysis

The PrOCoil R package allows not only to consider input sequences completely independently of one another, but also allows for overlaying profiles of aligned coiled coil segments. This feature is useful, for example, for studying effects of different mutations. As an example, we consider a double mutation of GCN4:

```
> GCN4m <- predict(PrOCoilModel, "MKQLEDKVEELLSKIYHNENEVARLKKLV", "abcdefgabcdefgabcdefgabc
> GCN4m


CCProfile object
        Sample: MKQLEDKVEELLSKIYHNENEVARLKKLV
                abcdefgabcdefgabcdefgabcdefga
        discriminant function value = 0.8776515
        predicted as trimer


> plot(GCN4wt, GCN4m)
```



By default, the profile corresponding of the first argument passed to `plot` is plotted in red and the second argument's profile is plotted in blue. The sequence of the first argument is shown above the graph and the second argument's sequence is shown below the graph. Black letters correspond to residues that are the same in the two sequences, whereas mutations are highlighted in color. More information on how to customize plots is available in 5.5.4 and the documentation available via `help("plot-methods")`.

Note: profile overlays only work if the two sequences have been equally long and if their heptad annotations are exactly the same.

5.5 Miscellanea

5.5.1 Processing predicted coiled coil segments with the R package

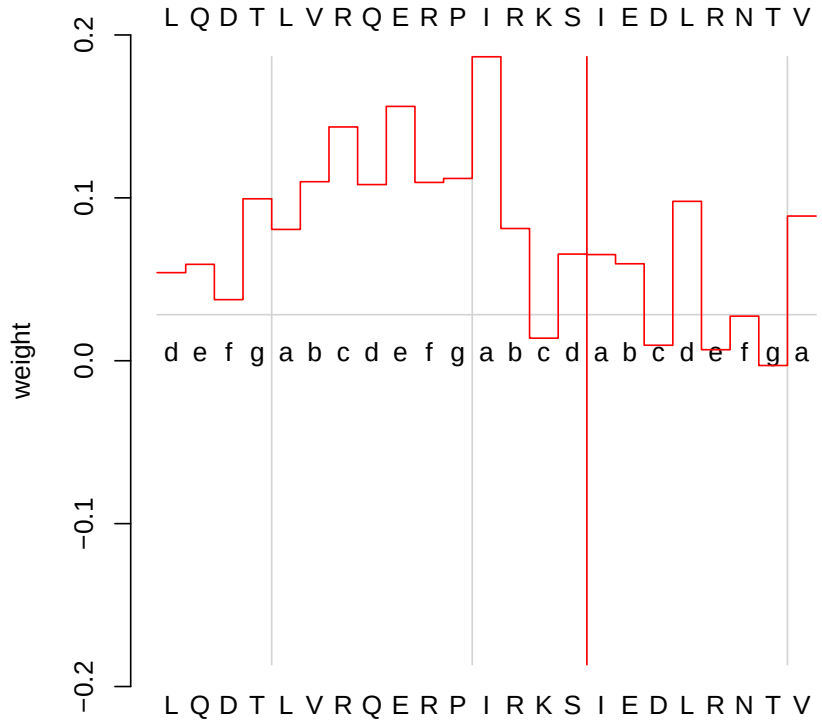
The PrOCoil R package itself is not able to process Marcoil or PairCoil2 output. Users who want to process prediction results obtained from these programs are recommended to use the PrOCoil Web interface to convert Marcoil or PairCoil2 output (see Section 4) into the correct input format that can be processed by the PrOCoil R package.

5.5.2 Heptad irregularities

Some coiled coils occurring in nature have heptad irregularities, e.g. incomplete heptads in which an '`a`' position follows after a '`d`' position. PrOCoil can also process heptad annotations with such

irregularities. In the profile plots created by the PrOCoil Web interface, such irregularities can only be seen in the register labels in the middle of the plot. In the profile plots created by the PrOCoil R package, heptad irregularities are additionally visualized by a vertical red line between the two positions that do not conform to the usual regular pattern.

```
> plot(predict(PrOCoilModel, "LQDTLVRQERPIRKSIEDLRNTV", "defgabcdefgabcdabcdefga"))
```



5.5.3 Alternative models

Trimers occur less frequently than dimers. Correspondingly, there were more dimers and trimers in the data set used for training the default model `PrOCoilModel`. Since this model was optimized for (standard) classification accuracy, it tends to concentrate on classifying the larger class — dimers — properly. That is why the model performs better in terms of specificity/true negative rate (correctly classified dimers) than in terms of sensitivity/true positive rate (correctly classified trimers). In case one wants to attribute equal importance to sensitivity and specificity, we have trained a second model that is optimized for *balanced accuracy*, i.e., the average of sensitivity and specificity (see publication supplement of [10]). This model can be used simply by calling `predict`

with PrOCoilModelBA as first argument (instead of PrOCoilModel which is the standard PrOCoil model). Like the standard model PrOCoilModel, the alternative model PrOCoilModelBA is automatically available once the PrOCoil R has been loaded.

For transparency and future safety , the PrOCoil R package further provides an open interface for loading custom classification models from a file. For this purpose, the function readCCModel is available. In order to see how such a model file should be organized, the two models included in the PrOCoil R package are available for download at:

```
http://www.bioinf.jku.at/software/procoil/PrOCoilModel.patternmodel
http://www.bioinf.jku.at/software/procoil/PrOCoilModelBA.patternmodel
```

It is also possible to read them directly using the readCCModel command:

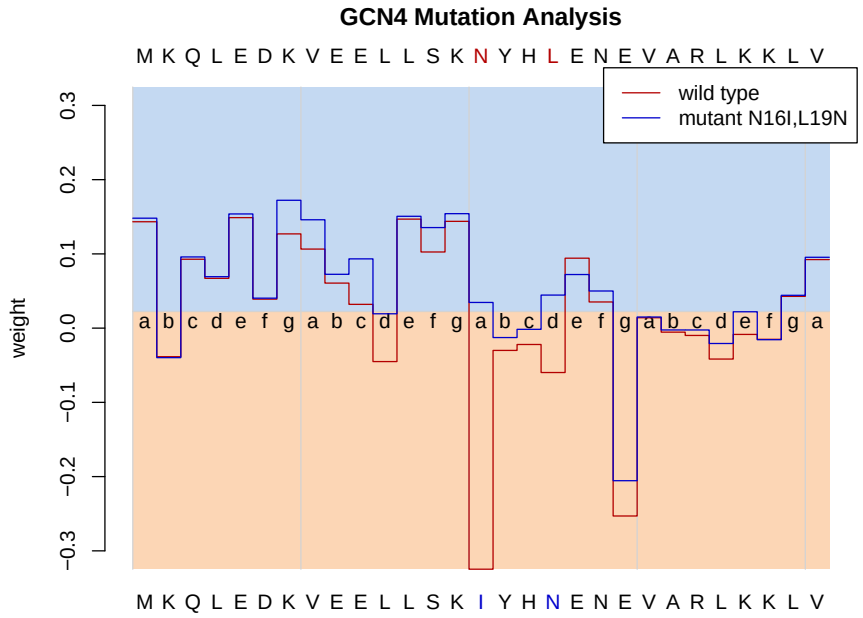
```
> readCCModel("http://www.bioinf.jku.at/software/procoil/PrOCoilModel.patternmodel")
```

```
CCModel object
        coiled coil kernel with m=7
        with kernel normalization
        18172 patterns
        b = -0.650577
```

5.5.4 Customizing and saving plots

The PrOCoil R package provides several opportunities to customize plots. For more detailed documentation, see help("plot-methods"). We simply provide an example here that uses legends, shading, custom coloring of profiles, and a plot header:

```
> plot(GCN4wt, GCN4m, legend=c("wild type", "mutant N16I,L19N"),
+      col=c(rgb(0.7,0,0), rgb(0,0,0.8)), main="GCN4 Mutation Analysis",
+      shades=c(rgb(0.77,0.85,0.95),rgb(0.99,0.84,0.71)))
```



R graphics have a fixed size and aspect ratio by default. That is why PrOCoil prediction profiles appear quite stretched for shorter and constricted for longer sequences. When exporting profile plots to graphics files, we therefore recommend to adjust the graphics dimensions in order to achieve a consistent and optically pleasing apperance:

- Use a fixed value for the *height* of the graphics device; we recommend a value between 5" and 7" for vector formats (PDF and (encapsulated) PostScript) and a few hundred pixels for bitmap image formats (BMP, JPEG, TIFF, PNG).
- Scale the *width* according to the length of the sequence; we recommend 1/24 of the graphic's height per residue as a rule of thumb.

Examples:

```
> pdf(file="GCN4wt.pdf",height=6,width=nchar(GCN4wt@seq)*6/24)
> plot(GCN4wt)
> dev.off()
> bmp(file="GCN4wt.bmp",height=480,width=nchar(GCN4wt@seq)*480/24)
> plot(GCN4wt)
> dev.off()
```

5.5.5 Alternative ways of supplying heptad registers to `predict`

As mentioned briefly in Subsection 5.2, the `predict` function requires a string that contains the heptad register annotation of the sequence to be classified. Normally, this string is passed as the

argument reg. Alternatively, one can attach the heptad register to the sequence object by means of the reg attribute:

```
> GCN4wtseq<-"MKQLEDKVEELLSKNYHLENEVARLKKLV"
> attr(GCN4wtseq,"reg")<-"abcdefgabcdefgabcdefgabcdefga"
> predict(PrOCoilModel,GCN4wtseq)

CCProfile object
        Sample: MKQLEDKVEELLSKNYHLENEVARLKKLV
                abcdefgabcdefgabcdefgabcdefga
        discriminant function value = -0.01532146
        predicted as dimer
```

The same works if a Biostrings object is supplied to predict:

```
> require(Biostrings)
> GCN4wtseq2<-AAString("MKQLEDKVEELLSKNYHLENEVARLKKLV")
> attr(GCN4wtseq2,"reg")<-"abcdefgabcdefgabcdefgabcdefga"
> predict(PrOCoilModel,GCN4wtseq2)

CCProfile object
        Sample: MKQLEDKVEELLSKNYHLENEVARLKKLV
                abcdefgabcdefgabcdefgabcdefga
        discriminant function value = -0.01532146
        predicted as dimer
```

For Biostrings objects, the register can also be supplied by a component named reg as part of the metadata slot:

```
> GCN4wtseq3<-AAString("MKQLEDKVEELLSKNYHLENEVARLKKLV")
> GCN4wtseq3@metadata$reg<-"abcdefgabcdefgabcdefgabcdefga"
> predict(PrOCoilModel,GCN4wtseq3)

CCProfile object
        Sample: MKQLEDKVEELLSKNYHLENEVARLKKLV
                abcdefgabcdefgabcdefgabcdefga
        discriminant function value = -0.01532146
        predicted as dimer
```

The predict function proceeds as follows: if the argument reg is present, it takes the content of this argument as heptad register. If the argument reg is missing, predict takes the reg attribute of the seq argument if this attribute exists. If the attribute does not exist and if seq is of class BString or AAString, then predict looks for a component named reg in the metadata slot of seq. If no heptad register is found in any of these three places, predict quits with an error message.

6 More Details About PrOCoil

6.1 How the prediction works

PrOCoil's classification models are based on support vector machines [3, 5, 12, 13] with a kernel designed specifically for coiled coil classification:

$$k(x, y) = \sum_{p \in P} N(p, x) \cdot N(p, y)$$

In this formula, x and y are the two input sequences that are to be compared, P is a set of coiled coil-specific patterns, and $N(p, x)$ is the number of occurrences of pattern p in sequence x.

PrOCoil uses the following set of patterns P: pairs of amino acids at fixed heptad positions with no more than a maximum number m of residues in between. Internally, these patterns are represented as strings with an amino acid letter on the first position, then a certain number of wildcards (between 0 and m as noted above), then the second amino acid letter, and finally a letter '`a`'–'`g`' denoting the heptad register position of the first amino acid, e.g. "`N..La`". This pattern matches a coiled coil sequence if the sequence has an '`N`' (Asparagine) at an '`a`' position and an '`L`' (Leucine) at the next '`d`' position.[6] For instance, the GCN4 wild type has one occurrence of this pattern:

```
MKQLEDKVEELLSKNYHLENEVARLKKLV
abcdefgabcdefgabcdefgabcdefga
             N..L
             a  d
```

So, obviously, PrOCoil considers pair interactions of amino acids at given heptad positions which are no more than m positions apart. The kernel described above resembles the spatial sample kernel [9] (however, with a heptad-specific property) and the kernel described in [7] (however, considering interactions within one sequence and not restricting to a particular subset of interactions).

The two models included in PrOCoil further employ kernel normalization [14] to correct for different sequence lengths. This means that the support vector machines have been trained with the kernel

$$k'(x, y) = \frac{k(x, y)}{\sqrt{k(x, x) \cdot k(y, y)}},$$

where $k(.,.)$ is the *coiled coil kernel* described above.

Using an explicit representation of the feature mapping underlying the kernel, the support vector machines have been transformed into linear classifiers on sequence features (cf. [2] for details). Thus, PrOCoil's prediction models consist of weights for specific sequence features (i.e. *patterns*) and a constant offset. Let us assume that x denotes a new coiled coil sequence. Without kernel normalization, the discriminant function value of the new sequence x is given as

$$f(x) = b + \sum_{p \in P} N(p, x) \cdot w(p),$$

[6] The heptad position of the second amino acid is tacitly assumed to comply to the regular heptad repeat. If the sequence has irregularities in its heptad annotation, PrOCoil only considers the patterns which match in both positions.

where b is the constant offset of the support vector machine and $w(p)$ is the weight of pattern p.

If kernel normalization is employed, the following, slightly more complicated, representation is obtained:

$$f(x) = b + \left(\sum_{p \in P} N(p,x) \cdot w(p) \right) \Big/ \underbrace{\sqrt{\sum_{p \in P} N(p,x)^2}}_{=R(x)}$$

Obviously, $R(x)$ is the value that corrects for the sequence length. The longer the sequence, the larger $R(x)$.

6.2 PrOCoil's built-in models

As already mentioned above, PrOCoil provides a default model that is optimized for classification accuracy (object `PrOCoilModel` in the PrOCoil R package) and an alternative model that is optimized for balanced accuracy (object `PrOCoilModelBA` in the PrOCoil R package). Both models have been trained on the same training set which consisted of verified coiled coils from the PDB enriched by putatively similar sequences that were obtained by BLAST in conjunction with Marcoil. The default model was created with libSVM's C-SVM implementation [4] using the normalized coiled coil kernel with $m = 7$ and a penalty parameter of $C = 8$. The alternative mode was created with PSVM [8] using the normalized coiled coil kernel with $m = 8$, class balancing, and regularization parameters $C = 2$ and $\varepsilon = 1.3$.

In the PrOCoil R package, all parameters describing a model are stored in the corresponding `CCModel` object:

```
> PrOCoilModel

CCModel object
        coiled coil kernel with m=7
        with kernel normalization
        18172 patterns
        b = -0.650577

> weights(PrOCoilModel)["N..La"]

$N..La
[1] -1.192731

> PrOCoilModelBA

CCModel object
        coiled coil kernel with m=8
        with kernel normalization
        21089 patterns
        b = -0.3294237
```

```
> weights(PrOCoilModelBA)["N..La"]


$N..La
[1] -2.731276
```

In both models, the pattern weights are sorted decreasingly, i.e., from most trimeric to most dimeric. Thus, the user also has easy access to the most indicative patterns. Here we provide an example how to extract the 25 most trimeric (i.e. the first 25 patterns in the list) and the 25 most dimeric patterns (i.e. the last 25 patterns in the list) from the PrOCoil model:

```
> noP<-length(weights(PrOCoilModel))
> names(weights(PrOCoilModel))[1:25]


 [1] "E..Ie"     "S.Lf"      "E....Kb"   "I...Id"    "L...Vd"
 [6] "I..Ed"     "M......Va" "N..Ve"     "E.Ke"      "R...Lg"
[11] "V...Lg"    "V...Ld"    "Q.Ec"      "AIc"       "Q..Eb"
[16] "K....Eg"   "V......Va" "DAg"       "IEa"       "V....Sa"
[21] "M...Id"    "L.Ke"      "I.Ia"      "K...Mg"    "E....Ae"
```

```
> names(weights(PrOCoilModel))[noP:(noP-24)]


 [1] "E...Lg"     "L...Nd"     "N..La"      "E.Ee"
 [5] "N.....Ea"   "K..La"      "L...Kd"     "K.....Le"
 [9] "E......Ec"  "V......Na"  "K...Ea"     "E.Ec"
[13] "AEa"        "K.......Kg" "K....Na"    "EVg"
[17] "E....Lc"    "EKf"        "VKa"        "K.......Lc"
[21] "AAf"        "Q...Ig"     "K.....Ea"   "KEb"
[25] "V......Ka"
```

6.3 How prediction profiles are obtained

Regardless of whether kernel normalization is employed or not, the essential component of the discriminant function is the sum

$$\sum_{p \in P} N(p, x) \cdot w(p).$$

It is obvious that every match of a pattern p contributes $w(p)$ to the sum. In order to find out the extent to which each residue contributes to the final classification, we can rewrite the sum as

$$\sum_{p \in P} N(p, x) \cdot w(p) = \sum_{i=1}^{L} c_i(x), \tag{1}$$

where L is the length of the sequence x and $c_i(x)$ is the contribution of the i-th residue of x. The contribution $c_i(x)$ can simply be computed as half the sum of weights of patterns matching the

i-th residue.[7] The weights $c_i(x)$ (or $c_i(x)/R(x)$ in case kernel normalization is employed) can be understood as a profile that can be plot over the sequence. We have already introduced these values as *prediction profiles* above. In order to have a unified terminology, let us denote the prediction profiles as

$$s_i(x) = \begin{cases} c_i(x)/R(x) & \text{if kernel normalization is employed,} \\ c_i(x) & \text{otherwise.} \end{cases}$$

A positive values $s_i(x)$ indicates that the i-th residue is participating more strongly in patterns that are indicative for trimers. A negative values $s_i(x)$ indicates that the i-th residue is participating more strongly in patterns that are indicative for dimers. A value $s_i(x)$ or zero or close to zero either means that the i-th residue is not participating in any indicative patterns or that it participates in dimer and trimer patterns the contributions of which compensate/cancel each other.

PrOCoil computes prediction profiles each time it computes a prediction. It is clear that we can recover the discriminant value $f(x)$ as follows:

$$f(x) = b + \sum_{i=1}^{L} s_i(x)$$

If $b \neq 0$, which is the normal case, the values $s_i(x)$ do not provide enough information to infer the final classification from the prediction profile. In order to facilitate a more sensible analysis, let us reformulate the above equality as

$$f(x) = \sum_{i=1}^{L} \big(s_i(x) + \frac{b}{L}\big) = \sum_{i=1}^{L} \big(s_i(x) - (-\frac{b}{L})\big).$$

Hence, if we plot the profile values $s_i(x)$ along with a horizontal line at $-\frac{b}{L}$, we can recover the discriminant value $f(x)$ as the difference of the area above the $-\frac{b}{L}$ minus the area below $-\frac{b}{L}$. The value $-\frac{b}{L}$ exactly corresponds to the "base line" mentioned above.

7 How to Cite PrOCoil

If you use PrOCoil for research that is published later, you are kindly asked to cite it as follows:

> C. C. Mahrenholz, I. G. Abfalter, U. Bodenhofer, R. Volkmer, and S. Hochreiter. Complex networks govern coiled coil oligomerization — predicting and profiling by means of a machine learning approach. *Mol. Cell. Proteomics* 10(5), 2011. DOI: 10.1074/mcp.M110.004994

To obtain this reference in BibTEX format, you can enter the following into your R session:

```
> toBibtex(citation("procoil"))
```

[7] As all patterns match two distinct residues, each weight must be split to the two residues in order to ensure that the equality (1) holds.

References

[1] B. Berger, D. B. Wilson, E. Wolf, T. Tonchev, M. Milla, and P. S. Kim. Predicting coiled coils by use of pairwise residue correlations. *Proc. Natl. Acad. Sci. USA*, 92:8259–8263, 1995.

[2] U. Bodenhofer, K. Schwarzbauer, M. Ionescu, and S. Hochreiter. Modeling position specificity in sequence kernels by fuzzy equivalence relations. In J. P. Carvalho, D. Dubois, U. Kaymak, and J. M. C. Sousa, editors, *Proc. Joint 13th IFSA World Congress and 6th EUSFLAT Conference*, pages 1376–1381, Lisbon, July 2009.

[3] C. J. M. Burges. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.*, 2:121–167, 1998.

[4] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[5] C. Cortes and V. N. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1986.

[6] M. Delorenzi and T. Speed. An HMM model for coiled-coil domains and a comparison with PSSM-based predictions. *Bioinformatics*, 18(4):617–625, 2002.

[7] J. H. Fong, A. E. Keating, and M. Singh. Predicting specificity in bZIP coiled-coil protein interactions. *Genome Biol.*, 5:R11, 2004.

[8] S. Hochreiter and K. Obermayer. Support vector machines for dyadic data. *Neural Comput.*, 18:1472–1510, 2006.

[9] P. Kuksa, P.-H. Huang, and V. Pavlovic. A fast, large-scale learning method for protein sequence classification. In *8th Int. Workshop on Data Mining in Bioinformatics*, pages 29–37, Las Vegas, NV, 2008.

[10] C. C. Mahrenholz, I. G. Abfalter, U. Bodenhofer, R. Volkmer, and S. Hochreiter. Complex networks govern coiled coil oligomerization — predicting and profiling by means of a machine learning approach. *Mol. Cell. Proteomics* 10(5), 2011.

[11] A. V. McDonnell, T. Jiang, A. E. Keating, and B. Berger. Paircoil2: improved prediction of coiled coils from sequence. *Bioinformatics*, 22(3):356–358, 2006.

[12] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Trans. Neural Networks*, 12(2):181–201, 2001.

[13] B. Schölkopf and A. J. Smola. *Learning with Kernels*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2002.

[14] B. Schölkopf, K. Tsuda, and J.-P. Vert, editors. *Kernel Methods in Computational Biology*. MIT Press, Cambridge, MA, 2004.

[15] J. Walshaw and D. N. Woolfson. SOCKET: a program for identifying and analysing coiled-coil motifs within protein structures. *J. Mol. Biol.*, 307(5):1427–1450, 2001.