procoil

October 25, 2011

`CCModel-class` *Class "CCModel"*

Description

S4 class representing a coiled coil prediction model

Objects from the Class

In principle, objects of this class can be created by calls of the form `new("CCModel")`, although it is probably never necessary to create such an object from scratch. The default model is stored in the object `PrOCoilModel`. An alternative model, `PrOCoilModelBA`, that is optimized for balanced accuracy is available too (see below). For future safety, other models can be loaded from files using the function `readCCModel`.

Discriminant function of model

Given a new coiled coil sequence x and a model, the discriminant function of the model is given as

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

where b is a constant, $N(p, x)$ denotes the number of occurrences of pattern p in sequence x, and $w(p)$ is the weight assigned to pattern p. P is the set of all patterns contained in the model. In the models used in the **procoil** package, the weights are computed from a support vector machine. Models can include kernel normalization or not. The formula above refers to the variant without kernel normalization. If kernel normalization is employed, the weights are computed in a different way and the discriminant function changes to

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

where $R(x)$ is a normalization value depending on the sample x. It is defined as follows:

$$R(x) = \sqrt{\sum_{p \in P} N(p, x)^2}$$

The **procoil** package does not consider arbitrary patterns, but only very specific ones: pairs of amino acids at fixed register 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 a 'L' (Leucine) at the next 'd' position. For instance, the GCN4 wildtype has one occurrence of this pattern:

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

Slots

`b`: Object of class `"numeric"` the value b as described above

`m`: Object of class `"integer"` the value m as described above

`scaling`: Object of class `"logical"` indicating whether the model should employ kernel normalization

`weights`: Object of class `"list"` storing all pattern weights; the patterns are stored in the format described above

Methods

predict `signature(object = "CCModel")`: see `predict`

weights `signature(object = "CCModel")`: see `weights`

show `signature(object = "CCModel")`: see `show-methods`

Default model PrOCoilModel

The **procoil** package provides a default coiled coil prediction model, `PrOCoilModel`.

The model was created with libSVM *[Chang and Lin, 2001]* using the coiled coil kernel with $m = 7$, $C = 8$, and kernel normalization on the BLAST-augmented data set. It is optimized for standard (unbalanced) accuracy, i.e. it tries to minimize the probability of misclassifications. Since dimers are more frequent in the data set, it slightly favors dimers for unknown sequences.

Alternative model PrOCoilModelBA

As mentioned above, the default model `PrOCoilModel` slightly favors dimers. This may be undesirable for some applications. For such cases, an alternative model `PrOCoilModelBA` is available that is optimized for balanced accuracy, i.e. it tries not to favor the larger class - dimers -, but may therefore prefer trimers in borderline cases. The overall misclassification probability is slightly higher for this model than for the default model `PrOCoilModel`.

The model `PrOCoilModelBA` was created with PSVM *[Hochreiter and Obermayer, 2006]* using the coiled coil kernel with $m = 8$, $C = 2$, $\varepsilon = 1.3$, class balancing, and kernel normalization on the BLAST-augmented data set.

Author(s)

Ulrich Bodenhofer <bodenhofer@bioinf.jku.at>

References

http://www.bioinf.jku.at/software/procoil/

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

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

Hochreiter, S., and Obermayer, K. (2006) Support vector machines for dyadic data. Neural Computation 18:1472-1510. DOI: 10.1162/neco.2006.18.6.1472

See Also

predict-methods, show-methods, weights-methods

Examples

```
showClass("CCModel")

## show summary of default model (optimized for accuracy)
PrOCoilModel

## show weight of pattern "N..La"
weights(PrOCoilModel)[["N..La"]]

## show the 10 patterns that are most indicative for trimers
## (as the weights are sorted in descending order in PrOCoilModel)
weights(PrOCoilModel)[1:10]

## show the 10 patterns that are most indicative for dimers
## (as the weights are sorted in descending order in PrOCoilModel)
nW <- length(weights(PrOCoilModel))
weights(PrOCoilModel)[nW:(nW - 9)]

## predict oligomerization of GCN4 wildtype
GCN4wt<-predict(PrOCoilModel,
                "MKQLEDKVEELLSKNYHLENEVARLKKLV",
                "abcdefgabcdefgabcdefgabcdefga")

## show summary of alternative model (optimized for balanced accuracy)
PrOCoilModelBA

## show weight of pattern "N..La"
weights(PrOCoilModelBA)[["N..La"]]

## show the 10 patterns that are most indicative for trimers
## (as the weights are sorted in descending order in PrOCoilModelBA)
weights(PrOCoilModelBA)[1:10]

## show the 10 patterns that are most indicative for dimers
## (as the weights are sorted in descending order in PrOCoilModelBA)
nW <- length(weights(PrOCoilModelBA))
weights(PrOCoilModelBA)[nW:(nW - 9)]
```

`CCProfile-class` *Class "CCProfile"*

Description

S4 class for representing coiled coil prediction results

Objects from the Class

In principle, objects of this class can be created by calls of the form `new("CCProfile")`, although there is no need in doing so. Most importantly, the `predict` function of `procoil` stores its results in objects of this type.

Slots

`seq`: Object of class `"character"` containing the amino acid sequence for which the prediction has been made

`reg`: Object of class `"character"` containing the heptad register corresponding to the amino acid sequence for which the prediction has been made

`profile`: Array of numerical values representing the prediction profile for the sequence under consideration. This array has the same length as the sequence.

`b`: Object of class `"numeric"`; value b used in the discriminant function (see `CCModel` for details

`disc`: Object of class `"numeric"` containing the discriminant function value (see `CCModel` for details

`pred`: Object of class `"character"` containing the final classification. Upon a call to `predict`, it is either "trimer" or "dimer".

Methods

plot `signature(x = "CCProfile", y = "missing")`: see `plot`

plot `signature(x = "CCProfile", y = "CCProfile")`: see `plot`

profile `signature(fitted = "CCProfile")`: see `profile`

show `signature(object = "CCProfile")`: see `show`

Prediction profiles

As described in `CCModel`, the discriminant function of the coiled coil classifier is essentially a weighted sum of numbers of occurrences of certain patterns in the sequence under consideration, i.e. every pattern occurring in the sequence contributes a certain weight to the discriminant function. Since every such occurrence is uniquely linked to two specific residues in the sequence, every amino acid in the sequence contributes a unique weight to the discriminant function value which is nothing else but half the sum of weights of matching patterns in which this amino acid is involved. If we denote the contribution of each position i with $s_i(x)$, it follows immediately that

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

where L is the length of the sequence x.

Author(s)

Ulrich Bodenhofer <bodenhofer@bioinf.jku.at>

References

http://www.bioinf.jku.at/software/procoil/

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

See Also

CCModel, plot, plot, profile, show,

Examples

```
showClass("CCProfile")

## predict oligomerization of GCN4 wildtype
GCN4wt<-predict(PrOCoilModel,
                "MKQLEDKVEELLSKNYHLENEVARLKKLV",
                "abcdefgabcdefgabcdefgabcdefga")

## display summary of result
GCN4wt

## show raw prediction profile
profile(GCN4wt)

## plot profile
plot(GCN4wt)
```

---

plot-methods *Plotting prediction profiles*

Description

Functions for plotting prediction profiles

Usage

```
## S4 method for signature 'CCProfile,missing'
plot(x, col="red", rng=0,
             standardize=FALSE, shades=NULL,
             legend="", legend.pos="topright", ...)
## S4 method for signature 'CCProfile,CCProfile'
plot(x, y, col=c("red", "blue"), rng=0,
             standardize=FALSE, shades=NULL, legend=NULL,
             legend.pos="topright", ...)
```

Arguments

x	Object of class CCProfile to be plotted with plot
y	Object of class CCProfile to be plotted with plot (in case plot is called with two arguments to compare two profiles)
col	Character string containing the name of the color in which the profile should be plotted (in case plot is called with one CCProfile argument). A vector of character strings containing the names of the two colors in which the profiles should be plotted (in case plot is called with two CCProfile arguments).
rng	Argument that allows the user to preset the range of the profile plot. If 0 (default) or negative, the range is determined automatically from the values in the profile. Otherwise, the range is set to $[-rng, rng]$.
standardize	logical. If FALSE (default), the profile values s_i are displayed as they are with the value $y = -b/L$ superimposed as a light gray line. If TRUE, the whole profile is shifted by $-b/L$ and the light gray line is displayed at $y = 0$.
shades	Vector of at least two color specifications (default: NULL). If not NULL, the background area above and below the base line $y = -b/L$ are shaded in colors shades[1] and shades[2], respectively.
legend	A character string containing the legend/description of the profile (in case plot is called with one CCProfile argument). A vector of character strings containing the legends/descriptions of the profiles (in case plot is called with two CCProfile arguments). If empty, no legend is displayed.
legend.pos	Position specification for legend (if legend is specified). Can either be a vector with coordinates or a single keyword like "topright" (see legend).
...	all other arguments are passed to the standard plot command that is called internally to display the graphics window

Details

The plot function displays the profile as a step function over the sequence with the steps connected by vertical lines. The vertical plot range can be determined by the rng argument. The sequence and the heptad register are visualized below and above the profile, respectively. The value $-b/L$ and the light gray line has the following meaning: It is obvious that we can rewrite

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

as

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

so the discriminant function value $f(x)$ can be understood as the sum of values $s_i(x) - (-\frac{b}{L})$, i.e. the area between the constant value $-b/L$ and the prediction profile. If the area above the light gray line is greater than the area below the light gray line, the sequence is predicted as trimer, otherwise as dimer.

If plot is called with two CCProfile arguments, the two profiles are plotted together to facilitate a comparison of profiles (e.g. wild type sequences versus mutants).

Value

Both variants of `plot` do not return any value.

Author(s)

Ulrich Bodenhofer <bodenhofer@bioinf.jku.at>

References

http://www.bioinf.jku.at/software/procoil/

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

See Also

`procoil`, `CCModel`, `CCProfile`

Examples

```
## predict oligomerization of GCN4 wildtype
GCN4wt<-predict(PrOCoilModel,
                "MKQLEDKVEELLSKNYHLENEVARLKKLV",
                "abcdefgabcdefgabcdefgabcdefga")

## plot profile
plot(GCN4wt)

## predict oligomerization of a GCN4 mutant
GCN4m3<-predict(PrOCoilModel,
                "MKQLEDKVEELLSKIYHNENEVARLKKLV",
                "abcdefgabcdefgabcdefgabcdefga")

## plot two profiles
plot(GCN4wt, GCN4m3, legend=c("GCN4 wild type", "GCN4 mutant"))
```

`predict-methods` *Predict oligomerization of coiled coil segment*

Description

Function for predicting the oligomerization of a coiled coil segment

Usage

```
## S4 method for signature 'CCModel'
predict(object, seq, reg)
```

Arguments

object	The model to be considered; can either be one of the models included in the package (PrOCoilModel and PrOCoilModelBA) or any other model loaded or created by the user. For a detailed explanation of the two default models, seeCCModel.
seq	An amino acid sequence; valid characters are all uppercase letters except 'B', 'J', 'O', 'U', 'X', and 'Z'; invalid characters are tolerated, but ignored by prediction. This argument can be of any type that can be cast to a character string, including Biostrings objects.
reg	A character string denoting the heptad register; valid characters are the lowercase letters 'a' to 'g'. All characters must be in proper order, e.g. 'a' can only be followed by 'b', 'b' can only be followed by 'c', ..., 'g' can only be followed by 'a'. The register can start with any of the seven letters. It must always have the same length as the seq argument. If this argument is missing, predict looks whether the object passed as argument seq has an attribute reg. If this attribute is missing and if seq is of class BString or AAString, predict looks whether there is a reg component in the metadata slot of seq.

Details

The function predict is the most important one in the **procoil** package. It is used to apply a coiled coil prediction model to a new coiled coil sequence. It uses the discriminant function described in CCModel. By default the final classification is computed on the basis of the discriminant function value. If $f(x) >= 0$, x is predicted as trimer, otherwise as dimer.

Value

If the heptad register (supplied by argument reg or by one of the way described above) does not contain any dashes '-', predict returns a CCProfile object containing the classification result and the prediction profile. If the heptad register contains dashes '-', predict returns a list of CCProfile objects, each corresponding to the prediction profile of one of the coiled coil segments contained in the sequence. The names of the components are of the format "s_e", where 's' denotes the start position and 'e' denotes the end position of the respective coiled coil sequence in the original sequence. show returns an invisible NULL

Author(s)

Ulrich Bodenhofer <bodenhofer@bioinf.jku.at>

References

http://www.bioinf.jku.at/software/procoil/

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

See Also

procoil, CCModel, CCProfile

Examples

```
## predict oligomerization of GCN4 wildtype
GCN4wt<-predict(PrOCoilModel,
                "MKQLEDKVEELLSKNYHLENEVARLKKLV",
                "abcdefgabcdefgabcdefgabcdefga")

## show and plot results
GCN4wt
plot(GCN4wt)

## predict oligomerization of unknown sequence (Marcoil example)
MarcoilEx<-predict(PrOCoilModel,
                 "MGECDQLLVFMITSRVLVLSTLIIMDSRQVYLENLRQFAENLRQNIENVHSFLENLRADLENLRQKFPGKW
                 "--------------------------------abcdefgabcdefgabcdefgabcdefgabcdefg-----

## show and plot results
MarcoilEx
plot(MarcoilEx[[1]])
```

`procoil-package` *Prediction of Oligomerization of Coiled Coil Proteins*

Description

The **procoil** package allows to predict whether a coiled coil sequence (amino acid sequence plus heptad register) is more likely to form a dimer or more likely to form a trimer. The predict function not only computes the prediction itself, but also a profile which allows to determine the strengths to which the individual residues are indicative for either class. Profiles can also be visualized graphically using a plot function.

Details

The package defines two S4 classes, `CCModel` and `CCProfile`. The former's purpose to represent a coiled coil prediction model. The default model `PrOCoilModel` is pre-loaded when the package is loaded. An alternative model `PrOCoilModelBA` can be loaded on demand. Other models can be loaded with the function `readCCModel`. The `predict` function is used to predict the oligomerization of a coiled coil sequence (which consists of an amino acid sequence and a heptad register aligned to it). The result is stored in a `CCProfile` object. The resulting prediction profile can be visualized with `plot`.

Author(s)

Ulrich Bodenhofer <bodenhofer@bioinf.jku.at>

References

http://www.bioinf.jku.at/software/procoil/

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

Examples

```
## display summary of default model
PrOCoilModel

## predict oligomerization of GCN4 wildtype
GCN4wt<-predict(PrOCoilModel,
                "MKQLEDKVEELLSKNYHLENEVARLKKLV",
                "abcdefgabcdefgabcdefgabcdefga")

## display result
GCN4wt

## plot profile
plot(GCN4wt)

## predict oligomerization of unknown sequence (Marcoil example)
MarcoilEx<-predict(PrOCoilModel,
                 "MGECDQLLVFMITSRVLVLSTLIIMDSRQVYLENLRQFAENLRQNIENVHSFLENLRADLENLRQKFPGKW
                 "------------------------------abcdefgabcdefgabcdefgabcdefgabcdefg-----

## display result
MarcoilEx[[1]]

## plot profile
plot(MarcoilEx[[1]])
```

`profile-methods` *Retrieve prediction profile from a CCProfile object*

Description

Function for retrieving the prediction profile from a `CCProfile` object

Usage

```
## S4 method for signature 'CCProfile'
profile(fitted)
```

Arguments

`fitted`	the coiled coil prediction to be considered

Details

The function `profile` provides an accessor function for retrieving the prediction profile from a `CCProfile` object without the need to directly access the `profile` slot.

Value

`profile` returns a vector with the prediction profile.

Author(s)

Ulrich Bodenhofer <bodenhofer@bioinf.jku.at>

References

http://www.bioinf.jku.at/software/procoil/

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

See Also

procoil, CCProfile, predict-methods

Examples

```
## predict oligomerization of GCN4 wildtype
GCN4wt<-predict(PrOCoilModel,
                "MKQLEDKVEELLSKNYHLENEVARLKKLV",
                "abcdefgabcdefgabcdefgabcdefga")

## show raw prediction profile
profile(GCN4wt)
```

---

readCCModel *Reading a coiled coil predicton model from a file*

Description

Function to read a coiled coil prediction model from a file into a CCModel object.

Usage

```
readCCModel(file)
```

Arguments

file	the name of the file from which the model should be read

Details

This function is more or less for testing purposes and future safety. Once the package **procoil** is loaded, the default model PrOCoilModel is pre-loaded. The function readCCModel can be used to load alternative models that may be published in the future (or created manually by intensionally changing parameters). The file must be a comma-separated table with two columns. The left column must contain strings and the right column must contain numerical values. Entries in the first column that start with '_' refer to model parameters that are directly written into the corresponding slots of the resulting model. All other entries are interpreted as patterns (with the corresponding weights in the second columns). See CCModel for an overview of model parameters.

Value

Upon successful completion, the function returns a CCModel object. In case of an error, the function does not return any value.

Note

The PrOCoil model is available on on http://www.bioinf.jku.at/software/procoil/PrOCoilModel.patternmodel. in exactly the format the function readCCModel requires. Analogously for the alternative model optimized for balanced accuracy (see CCModel): http://www.bioinf.jku.at/software/procoil/PrOCoilModelBA.patternmodel

Author(s)

Ulrich Bodenhofer <bodenhofer@bioinf.jku.at>

References

http://www.bioinf.jku.at/software/procoil/

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

See Also

procoil, CCModel-class

Examples

```
## Not run: ## read example model file
testmodel<-readCCModel("http://www.bioinf.jku.at/software/procoil/PrOCoilModel.patternmod

## display summary of example model
testmodel

## display 10 heightes pattern weights
weights(testmodel)[1:10]
## End(Not run)
```

---

show-methods *Functions for CCModel class*

Description

Display information about CCModel and CCProfile objects

Usage

```
## S4 method for signature 'CCModel'
show(object)
## S4 method for signature 'CCProfile'
show(object)
```

Arguments

object	An object of class CCModel or CCProfile

Details

If called for an object of class CCModel, show displays a summary of data in the CCModel object. Since the number of pattern weights is potentially large, these values are not displayed by show. In order to see the pattern weights (for what reason ever), directly access the weights slot (see example below).

If called for an object of class CCProfile, show displays a summary of the prediction result, including the sequence, its heptad register, the discriminant function value, and the final prediction (trimer or dimer). The prediction profile itself is not displayed. To access the profile, directly access the profile slot (see also CCProfile).

Value

Both variants of show returns an invisible NULL

Author(s)

Ulrich Bodenhofer <bodenhofer@bioinf.jku.at>

References

http://www.bioinf.jku.at/software/procoil/

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

See Also

procoil, CCModel, CCProfile

Examples

```
## display summary of default model
show(PrOCoilModel)

## show all pattern weights
PrOCoilModel@weights

## predict oligomerization of GCN4 wildtype
GCN4wt<-predict(PrOCoilModel,
                "MKQLEDKVEELLSKNYHLENEVARLKKLV",
                "abcdefgabcdefgabcdefgabcdefga")

## show result
show(GCN4wt)
```

`weights-methods` *Retrieve pattern weights from a CCModel object*

Description

Function for retrieving the pattern weights from a `CCModel` object

Usage

```
## S4 method for signature 'CCModel'
weights(object)
```

Arguments

`object`	The model to be considered; can either be one of the models included in the package (`PrOCoilModel` and `PrOCoilModelBA`) or any other model loaded or created by the user. For a detailed explanation of the two default models, see `CCModel`.

Details

The function `weights` provides an accessor function for retrieving the pattern weights from a `CCModel` object without the need to directly access the `weights` slot.

Value

`weights` returns a list of pattern weights.

Author(s)

Ulrich Bodenhofer <bodenhofer@bioinf.jku.at>

References

http://www.bioinf.jku.at/software/procoil/

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

See Also

`procoil`, `CCModel`, `CCProfile`, `readCCModel`

Examples

```
## show weight of pattern "N..La"
weights(PrOCoilModel)[["N..La"]]

## show the 10 patterns that are most indicative for trimers
## (as the weights are sorted in descending order in PrOCoilModel)
weights(PrOCoilModel)[1:10]

## show the 10 patterns that are most indicative for dimers
```

```
## (as the weights are sorted in descending order in PrOCoilModel)
nW <- length(weights(PrOCoilModel))
weights(PrOCoilModel)[nW:(nW - 9)]

## show weight of pattern "N..La"
weights(PrOCoilModelBA)[["N..La"]]

## show the 10 patterns that are most indicative for trimers
## (as the weights are sorted in descending order in PrOCoilModelBA)
weights(PrOCoilModelBA)[1:10]

## show the 10 patterns that are most indicative for dimers
## (as the weights are sorted in descending order in PrOCoilModelBA)
nW <- length(weights(PrOCoilModelBA))
weights(PrOCoilModelBA)[nW:(nW - 9)]
```

Index