Package 'ptw'

May 26, 2018

Type Package

Title Parametric Time Warping

Version 1.9-13

Description Parametric Time Warping aligns patterns, i.e. it aims to put corresponding features at the same locations. The algorithm searches for an optimal polynomial describing the warping. It is possible to align one sample to a reference, several samples to the same reference, or several samples to several references. One can choose between calculating individual warpings, or one global warping for a set of samples and one reference. Two optimization criteria are implemented: RMS (Root Mean Square error) and WCC (Weighted Cross Correlation). Both warping of peak profiles and of peak lists are supported.

License GPL (>= 2)

Imports nloptr, graphics, grDevices, stats

NeedsCompilation yes

Author Jan Gerretzen [ctb],
Paul Eilers [aut],
Hans Wouters [ctb],
Tom Bloemberg [aut],
Ron Wehrens [aut, cre]

Maintainer Ron Wehrens <ron.wehrens@gmail.com>

Repository CRAN

Date/Publication 2018-05-25 22:17:57 UTC

R topics documented:

`asysm` *Trend estimation with asymmetric least squares*

Description

Estimates a trend based on asymmetric least squares. In this case used to estimate the baseline of a given spectrum.

Usage

```
asysm(y, lambda = 1e+07, p = 0.001, eps = 1e-8, maxit = 25)
```

Arguments

`y`	data: either a vector or a data matrix containing spectra as rows
`lambda`	smoothing parameter (generally 1e5 - 1e8)
`p`	asymmetry parameter
`eps`	numerical precision for convergence
`maxit`	max number of iterations. If no convergence is reached, a warning is issued.

Details

Asymmetric least squares (not to be confused with alternating least squares) assigns different weights to the data points that are above and below an iteratively estimated trendline, respectively. In this case, the asymmetry parameter p (0 <= p <= 1) is the weight for points above the trendline, whereas 1-p is the weight for points below it. Naturally, p should be small for estimating baselines. The parameter lambda controls the amount of smoothing: the larger it is, the smoother the trendline will be.

Value

An estimated baseline

Author(s)

Paul Eilers, Jan Gerretzen

References

Eilers, P.H.C. Eilers, P.H.C. (2004) "Parametric Time Warping", Analytical Chemistry, **76** (2), 404 – 411.

Boelens, H.F.M., Eilers, P.H.C., Hankemeier, T. (2005) "Sign constraints improve the detection of differences between complex spectral data sets: LC-IR as an example", Analytical Chemistry, **77**, 7998 – 8007.

Examples

```
data(gaschrom)
plot(gaschrom[1,], type = "l", ylim = c(0, 100))
lines(asysm(gaschrom[1,]), col = 2)
```

---

baseline.corr *Baseline Correction using asymmetric least squares*

Description

This function estimates a baseline using asymmetric least squares and subtracts it from the data.

Usage

```
baseline.corr(y, ...)
```

Arguments

y	signal(s) to correct. This can be a vector (containing one signal) or a matrix of signals(one signal per row)
...	other arguments to the asysm function.

Value

ycorr	baseline corrected signal(s): a vector or a matrix of the same dimension as the input signal(s)

Author(s)

Paul Eilers, Jan Gerretzen

References

Eilers, P.H.C. (2004) "Parametric Time Warping", Analytical Chemistry, **76** (2), 404 – 411.

Examples

```
data(gaschrom)
plot(gaschrom[1,], type = "l", ylim = c(0, 100))
lines(baseline.corr(gaschrom[1,]), col = 2)
```

---

bestref *Identification of optimal reference*

Description

This function calculates a similarity matrix and returns the sample number that is most similar to all other samples. This is possibly preferable as a reference sample since warping then may be kept to a minimum. Either RMS or WCC may be used as similarity functions.

Usage

```
bestref(x, optim.crit = c("WCC", "RMS"),
               trwdth=20, wghts = NULL, smooth.param = 0)
```

Arguments

`x`	data matrix or array of signals, specified row-wise. In case of an array, the third dimension should differentiate between the different samples
`optim.crit`	either `"WCC"` or `"RMS"`
`trwdth`	the width of the triangle in the WCC criterion, given as a number of data points. Default: 20
`wghts`	Optional weights vector in the WCC criterion; will be calculated from the triangle width if necessary. Sometimes it is more efficient to pre-calculate it and give it as an argument
`smooth.param`	smoothing parameter for smoothing the signal when `optim.crit` equals `"RMS"`. If no smoothing is required, set this to 0

Value

A list containing two elements:

`best.ref`	the index of the best reference(s)
`crit.values`	the qualities as measured by either RMS or WCC

Author(s)

Jan Gerretzen, Ron Wehrens

Examples

```
data(gaschrom)
bestref(gaschrom)
bestref(gaschrom, optim.crit = "WCC", trwdth = 50)
bestref(gaschrom, optim.crit = "RMS")
bestref(gaschrom, optim.crit = "RMS", smooth.param = 1e5)
```

`calc.multicoef` *Calculation of warping coefficients when applying more than one warping function successively*

Description

Applying two (or more) warping function after each other can be described with one warping function of a higher warping degree. This function provides the coefficients of this higher degree warping function.

Usage

```
calc.multicoef(coef1, coef2)
```

Arguments

`coef1`	vector containing the warping coefficients of the first applied warping function
`coef2`	vector containing the warping coefficients of the second applied warping function

Details

This function uses Pascal's simplex to calculate the new warping coefficients.

When applying three warping functions successively (first a, then b and finally c - here a, b and c are vectors of warping coefficients), first calculate the new coefficients for b and c, and afterwards the coefficients for a with these new coefficients. So the coefficients for the total warping function can be calculated via `calc.multicoef(a, calc.multicoef(b, c))`.

Value

a vector containing the corrected warping coefficients

Author(s)

Jan Gerretzen

References

Bloemberg, T.G., et al. (2010) "Improved parametric time warping for Proteomics", Chemometrics and Intelligent Laboratory Systems, **104** (1), 65 – 74.

See Also

calc.zerocoef

Examples

```
data(gaschrom)
ref <- gaschrom[1,]
samp <- gaschrom[16,]
coef1 <- c(100,1.1, 1e-5)
coef2 <- c(25, 0.95, 3.2e-5)
gaschrom.ptw <- ptw(ref, samp, init.coef = coef1, try = TRUE)
ref.w <- gaschrom.ptw$reference
samp.w <- gaschrom.ptw$warped.sample
samp.w[is.na(samp.w)] <- 0
gaschrom.ptw2 <- ptw(ref.w, samp.w, init.coef = coef2, try = TRUE)
plot(c(gaschrom.ptw2$warped.sample), type = "l")

corr.coef <- calc.multicoef(coef1, coef2)
gaschrom.ptw3 <- ptw(ref, samp, init.coef = corr.coef, try = TRUE)
lines(c(gaschrom.ptw3$warped.sample), col = 2, lty = 2)
```

---

calc.zerocoef *Correction for warping coefficients when using zeropadding*

Description

This function calculates the warping coefficients for the original range of the data, based on the warping of zero-filled data. Only needed when zeros are added in the beginning of the signal.

Usage

```
calc.zerocoef(coef, zeros)
```

Arguments

coef	vector of warping coefficients of a PTW-calculation on a set of signals with zeros added to the beginning of the signal
zeros	the number of zeros added

Value

a vector containing the corrected warping coefficients

Author(s)

Jan Gerretzen

References

Bloemberg, T.G., et al. (2010) "Improved parametric time warping for Proteomics", Chemometrics and Intelligent Laboratory Systems, **104** (1), 65 – 74.

See Also

padzeros calc.multicoef

Examples

```
data(gaschrom)
gaschrom.zf <- padzeros(gaschrom, 250)
ref <- gaschrom[1,]
samp <- gaschrom[16,]
ref.zf <- gaschrom.zf[1,]
samp.zf <- gaschrom.zf[16,]
gaschrom.ptw <- ptw(ref.zf, samp.zf)
layout(matrix(1:2,2,1, byrow=TRUE))
plot(gaschrom.ptw)
corr.coef <- calc.zerocoef(gaschrom.ptw$warp.coef, 250)
gaschrom.ptw2 <- ptw(ref, samp, init.coef = corr.coef, try = TRUE)
plot(gaschrom.ptw2)
```

---

coda *Chromatogram selection using the CODA algorithm*

Description

The CODA algorithm calculates a so-called MCQ (Mass Chromatogram Quality) value for every row of the input. High MCQ values correspond with those chromatograms not containing spikes and/or a baseline.

Usage

```
coda(x, window = 5, smoothing = c("median", "mean"))
```

Arguments

x	data matrix containing chromatograms in the rows
window	width of the smoothing window
smoothing	type of smoothing: whether to use running means or running medians

Details

The MCQ value of a spectrum is the inner product between the standardized, smoothed chromatogram, and the length-scaled chromatogram. In literature, a cut-off of 0.85 has been reported to work well in selecting useful chromatograms, although this is strongly data-set dependent.

References

Windig, W., Phalp, J., Payna, A. (1996) "A noise and background reduction method for component detection in liquid chromatography/mass spectrometry", Analytical Chemistry, **68**, 3602 – 3606.

Examples

```
data(gaschrom)
coda(gaschrom)
```

`difsm` *Smoothing with a finite difference penalty*

Description

This function smoothes signals with a finite difference penalty of order 2.

Usage

```
difsm(y, lambda)
```

Arguments

`y`	signal to be smoothed: a vector
`lambda`	smoothing parameter: larger values lead to smoothing

Value

smoothed signal: a vector

Author(s)

Paul Eilers, Jan Gerretzen

References

Eilers, P.H.C. (2004) "Parametric Time Warping", Analytical Chemistry, **76** (2), 404 – 411.

Eilers, P.H.C. (2003) "A perfect smoother", Analytical Chemistry, **75**, 3631 – 3636.

Examples

```
data(gaschrom)
plot(gaschrom[1,], type = "l", ylim = c(0, 100))
lines(difsm(gaschrom[1,], lambda = 1e5), col = 2)
lines(difsm(gaschrom[1,], lambda = 1e6), col = 3)
lines(difsm(gaschrom[1,], lambda = 1e7), col = 4)
```

gaschrom *16 calibration GC traces*

Description

The object gaschrom contains 16 calibration GC traces, measured at 5,000 time points. A peak-picked version is available as object gaschrom.st (see example).

Usage

```
data(gaschrom)
```

Source

Claire Boucon, Unilever

References

Eilers, P.H.C. (2004) "Parametric Time Warping", Analytical Chemistry, **76** (2), 404 – 411.

Examples

```
data(gaschrom)

## the gaschrom.st object has been generated in the following way:
## Not run:
pick.peaks <- function(x, span) {
  span.width <- span * 2 + 1
  loc.max <- span.width + 1 -
      apply(embed(x, span.width), 1, which.max)
  loc.max[loc.max == 1 | loc.max == span.width] <- NA

  pks <- loc.max + 0:(length(loc.max)-1)
  pks <- pks[!is.na(pks)]
  pks.tab <- table(pks)

  pks.id <- as.numeric(names(pks.tab)[pks.tab > span])

  cbind(rt = pks.id, I = x[pks.id])
}

gaschrom <- t(apply(gaschrom, 1, baseline.corr))
gaschrom.st <- lapply(1:nrow(gaschrom),
                      function(ii)
                      pick.peaks(gaschrom[ii,], span = 11))
## remove peaks with an intensity below 10
gaschrom.st <- lapply(gaschrom.st,
                      function(pk)
                      pk[pk[,2] >= 10,])
```

```
## End(Not run)
plot(gaschrom[1,], type = "l", xlim = c(3000, 3500), ylim = c(0, 200))
abline(h = 10, lty = 2, col = 2)
abline(v = gaschrom.st[[1]], col = 4)
```

`lcms` *Parts of 3 proteomic LC-MS samples*

Description

The `lcms` data consists of a 100 x 2000 x 3 array `lcms`, a vector `time` of length 2000 and a vector `mz` of length 100. The LC-MS data in the array are a subset of a larger set measured on a tryptic digest of E. coli proteins. Peak picking leads to the object ldms.pks (see example section).

Usage

```
data(lcms)
```

Source

Nijmegen Proteomics Facility, Department of Laboratory Medicine, Radboud University Nijmegen Medical Centre

References

Bloemberg, T.G., et al. (2010) "Improved parametric time warping for Proteomics", Chemometrics and Intelligent Laboratory Systems, **104** (1), 65 – 74.

Examples

```
## the lcms.pks object is generated in the following way:
## Not run:
data(lcms)
pick.peaks <- function(x, span) {
  span.width <- span * 2 + 1
  loc.max <- span.width + 1 -
      apply(embed(x, span.width), 1, which.max)
  loc.max[loc.max == 1 | loc.max == span.width] <- NA

  pks <- loc.max + 0:(length(loc.max)-1)
  pks <- pks[!is.na(pks)]
  pks.tab <- table(pks)

  pks.id <- as.numeric(names(pks.tab)[pks.tab > span])

  cbind(rt = pks.id, I = x[pks.id])
}

## bring all samples to the same scale, copied from ptw man page
```

```
lcms.scaled <- aperm(apply(lcms, c(1,3),
                           function(x) x/mean(x) ), c(2,1,3))
lcms.s.z <- aperm(apply(lcms.scaled, c(1,3),
                        function(x) padzeros(x, 250) ), c(2,1,3))
lcms.pks <- lapply(1:3,
                   function(ii) {
                     lapply(1:nrow(lcms.s.z[,,ii]),
                            function(jj)
                            cbind("mz" = jj,
                                  pick.peaks(lcms.s.z[jj,,ii], 5)))
                   })

## End(Not run)
```

`mzchannel2pktab` *Conversion between peak lists from hyphenated MS (LCMS, GCMS, ...) data and input for* `stptw`.

Description

Function pktab2mzchannel allows to split a list of peaks into several sublists, for instance on the basis of m/z values. The result can be aligned with `stptw`. The peak list can be obtained from packages like `XCMS`. The reverse function, mzchannel2pktab, simply gathers all peak positions in one matrix.

Usage

```
pktab2mzchannel(pktab, Ivalue = "maxo", masses = NULL, nMasses = 0, massDigits = 2)
mzchannel2pktab(mzchannels)
```

Arguments

`pktab`	a peak table as generated for example by XCMS. Necessary information: m/z value (column name "mz"), retention time (column name "rt") and intensity.
`Ivalue`	the name of the intensity value to be used. Default is "maxo", one of the columns generated by the XCMS package.
`masses`	a vector of specific masses
`nMasses`	an optional number limiting the number of mass channels. When both `masses` and `nMasses` are defined, the former takes preference
`massDigits`	number of significant mass digits - if no `masses` are supplied this number determines the number of distinct categories in the output
`mzchannels`	a list of peak matrices, e.g. the output of `pktab2mzchannel`

Value

Function `pktab2mzchannel` returns a list of peak matrices; list elements have the name of the mz value that they represent. Function `mzchannel2pktab` returns one peak matrix where all masses are in a specific column.

Author(s)

Ron Wehrens

Examples

```
data(lcms)
## first couple of peaks in the first three channels
(smallset <- lapply(lcms.pks[[1]][1:3], head))
## all in one data matrix
allpeaks <- mzchannel2pktab(smallset)
## and back again
pktab2mzchannel(allpeaks, Ivalue = "I")
```

---

padzeros *Pad matrix with zeros*

Description

Adds zeros to the left side of a matrix or vector, to its right side, or to both sides.

Usage

```
padzeros(data, nzeros, side="both")
```

Arguments

data	the original matrix or vector
nzeros	number of columns to add on one side
side	to which side to add the zeros - choose between "both", "left" or "right"

Details

When data is a numeric vector, it is converted to a matrix of a single row.

Value

A matrix with the same number of rows as the original matrix, and extra columns containing zeros on the specified side or sides

Author(s)

Jan Gerretzen

References

Bloemberg, T.G., et al. (2010) "Improved parametric time warping for Proteomics", Chemometrics and Intelligent Laboratory Systems, **104** (1), 65 – 74.

Examples

```
data(lcms)
lcms.z1 <- padzeros(lcms[75,,1], 250, side="left")
lcms.z2 <- padzeros(lcms[75,,1], 250, side="right")
lcms.z3 <- padzeros(lcms[75,,1], 250, side="both")
zeros <- rep(0, 250)

layout(matrix(1:4,2,2, byrow=TRUE))
plot(lcms[75,,1], type="l", main="Original signal")

plot(as.vector(lcms.z1), type="l", main="Padzeros left side")
points(1:250, zeros, col=2, lwd=0.08)

plot(as.vector(lcms.z2), type="l", main="Padzeros right side")
points(2001:2250, zeros, col=2, lwd=0.08)

plot(as.vector(lcms.z3), type="l", main="Padzeros both sides")
points(1:250, zeros, col=2, lwd=0.08)
points(2251:2500, zeros, col=2, lwd=0.08)
```

`plot.ptw` *Plot a ptw object*

Description

The function plots a `ptw` object. It shows either the original and warped signals, or the warping function.

Usage

```
## S3 method for class 'ptw'
plot(x, what = c("signal", "function"),
         type = c("simultaneous", "individual"), ask = TRUE, ...)
```

Arguments

`x`	object of class 'ptw'
`what`	`"signal"` plots the reference, sample and warped sample signal(s); `"function"` plots the warping function as warped 'time' - 'time' for the forward warping mode and as 'time' - warped 'time' for the backward warping mode.
`type`	`"simultaneous"` plots all signals or warping functions in one plot; `"individual"` generates multiple plots
`ask`	logical, whether to ask before showing a new plot
`...`	further arguments to the plotting functions

Author(s)

Jan Gerretzen, Ron Wehrens, Tom Bloemberg

Examples

```
data(gaschrom)
ref <- gaschrom[1:8,]
samp <- gaschrom[9:16,]
gaschrom.ptw <- ptw(ref, samp, warp.type = "individual",
              optim.crit = "RMS", init.coef = c(0, 1, 0, 0))
plot(gaschrom.ptw)
plot(gaschrom.ptw, what = "function")
```

`predict.ptw` *Prediction of warped signals*

Description

Given a ptw object, predict either the signal at a certain warped time, or the warped time itself.

Usage

```
## S3 method for class 'ptw'
predict(object, newdata, what = c("response", "time"),
  RTref = NULL, ...)
```

Arguments

`object`	An object of class "ptw"
`newdata`	Optional vector or matrix of new data points. If `what` equals "response", the new data should be a vector or matrix of intensities. If `what` equals "time", the new data is a vector of time points (a matrix of time points makes no sense...).
`what`	Either "response", in which case the function returns the warped signal, or "time", and then the function returns the warped time axis. That is, the time point in the warped sample corresponding to the given time point in the original sample.
`RTref`	Optional vector of retention times in the reference.
`...`	Further arguments, at the moment not used.

Value

The function returns a matrix (possibly containing only one row) of either warped time points or signals, warped according to the warping function defined in `object`. When warping signals individually, `predict.ptw` will check the dimension of `newdata`: if this is a vector or a matrix of one row, every single warping function will be applied to the one row. If the number of rows equals the number of warping functions, each row will be warped with its corresponding function. If the number of rows does not match the number of warping functions and is not equal to one, an error is given.

Author(s)

Ron Wehrens

References

Eilers, P.H.C. "Parametric Time Warping." Anal. Chem., 2004, 76, 404-411

Bloemberg, T.G. et al. "Improved parametric time warping for proteomics." Chemom. Intell. Lab. Syst., 2010, 104, pp. 65-74

See Also

ptw

Examples

```
## educational example, contributed by zeehio (Sergio Oller)
x1 <- c(rep(0, 5), 1,1,1, 20, 40, 20, 1, 1, 1, rep(0, 5))
x2 <- c(rep(0, 6), 1,1,1, 20, 40, 20, 1, 1, 1, rep(0, 4))
time <- 1:length(x1)
## get time-warped object. Default: 'forward' warping, also works
## with backward warping
w1b <- ptw(ref = x1, samp = x2)
## predict intensities of object x2 after warping at the times used in x1
x2wb <- predict(w1b, newdata = x2, what = "response")
## predict times where the original elements of x2 will end up
t2wb <- as.numeric(predict(w1b, newdata = time, what = "time"))

graphics.off()
par(mfrow = c(2,1))
plot(x1, type = "h", col = 2, lwd = 2, main = "Orig data")
points(x2, type = "h", col = 4)

plot(x1, type = "h", col = 2, lwd = 2, main = "Backward warping")
points(c(x2wb), type = "h", col = 4) # what = "response"
points(t2wb, x2, col = 4)            # what = "time"

## more relevant example
data(gaschrom)
## Global warping: all samples warped with the same function
ref <- gaschrom[1,]
samp <- gaschrom[14:16,]
gp <- ptw(ref, samp, init.coef = c(0, 1), warp.type = "global")
matplot(t(samp), type = "l", xlim = c(2200, 2400), lty = 1, col = 1:3)
lines(ref, type = "l", col = "gray", lwd = 2)
## plot predicted warped signal directly
matlines(t(predict(gp)), lty = 2, col = 1:3)
## plot original signal at warped time axis
matlines(t(predict(gp, newdata = 2001:2600, what = "time")),
         t(samp[,2001:2600]), col = 1:3, lwd = 3, lty = 2) ## OK
## result: good alignment with ref, differences between three profiles persist
```

```
## Individual warping: all samples warped individually
gp <- ptw(ref, samp, init.coef = c(0, 1), warp.type = "indiv")
predict(gp, what = "time", newdata = 2001:2600)
matplot(t(samp), type = "l", xlim = c(2200, 2400), lty = 1, col = 1:3)
lines(ref, type = "l", col = "gray", lwd = 2)
matlines(t(predict(gp, what = "time")),
         t(samp), col = 1:3, lty = 2)
## result: each individual profile is aligned to the ref

## How would samples 11:13 be warped using the coefficients from samples
## 14:16 (silly but just to make the point)?
samp.pred <- predict(gp, what = "response", newdata = gaschrom[11:13,])
```

---

ptw *Parametric Time Warping*

Description

The main functions of the ptw package, performing parametric time warping of one or more samples. Features in the samples are optimally aligned with features in the reference(s). One may align a single sample to a single reference, several samples to a single reference, and several samples to several references. In the latter case, the number of references and samples should be equal. One may require that all samples are warped with the same warping function, or one may allow individual warpings for all samples.

Usage

```
ptw(ref, samp, selected.traces,
    init.coef = c(0, 1, 0), try = FALSE,
    warp.type = c("individual", "global"),
    optim.crit = c("WCC", "RMS"),
    mode = c("forward", "backward"),
    smooth.param = ifelse(try, 0, 1e05),
    trwdth = 20, trwdth.res = trwdth,
    verbose = FALSE, ...)
## S3 method for class 'ptw'
summary(object, ...)
## S3 method for class 'ptw'
print(x, ...)
```

Arguments

ref	reference. Either a vector (containing one reference signal) or a matrix (one reference per row). If more than one reference is specified, the number of reference signals must equal the number of sample signals.
samp	sample. A vector (containing one sample signal) or a matrix (one sample per row).

`selected.traces`
: optional vector containing the row numbers to use from `ref` (if more than one reference signal is specified) and `samp`.

`init.coef`
: starting coefficients. The first number is the zeroth-order coefficient (i.e., a constant shift); further numbers indicate linear, quadratic, ... stretches. The default is to start from the identity warping using a quadratic function (`c(0, 1, 0)`)

`try`
: if `try = TRUE`, `ptw` does not optimize the warping but returns a `ptw` object containing the warping for `init.coef`. Default: `FALSE`

`warp.type`
: default is to treat samples and references as single entities and align them individually and independently. Using the argument `warp.type = "global"` leads to one alignment function; the samples are warped simultaneously to the reference(s). Also see details

`optim.crit`
: either `"WCC"` or `"RMS"`. In both cases, the optimal value of the alignment leads to a value of 0. For "WCC", this means that `1 - WCC` is optimized rather than `WCC` (where the optimal value equals 1)

`mode`
: either "forward" or "backward": the latter was the original implementation, basically for a point i in the original signal predicting the point j in the signal that would be in position i in the warped signal. The interpretation of the coefficients is counterintuitive. Therefore the default is "forward", simply predicting the location (time) in the warped signal of a particular point. Apart from possible numerical optimisation issues, both warpings should give the same net result.

`smooth.param`
: smoothing parameter for smoothing the reference and sample when `optim.crit` equals `"RMS"`. If no smoothing is required, set this to 0. The default is to use smoothing in the optimization mode, and no smoothing otherwise

`trwdth`
: the width of the triangle in the WCC criterion during the optimization, given as a number of data points. Default: 20

`trwdth.res`
: the width of the triangle in the WCC calculation in the calculation of the quality of the final result. Default: equal to `trwdth`

`verbose`
: logical, default is `FALSE`. Whether to give output during the optimisation, which may be useful for large data sets

`...`
: further arguments to optim

`x, object`
: an object of class "ptw"

Details

Function `ptw` and friends is meant for profile data, where intensities have been recorded at regular time points; function `stptw` is meant for lists of peaks, for instance obtained after peak-picking the profile data. The latter option is less flexible (Euclidean distance and backward warping have not been implemented) but is much faster, especially for larger data sets.

In the optimization mode (`try = FALSE`), the function optimizes the warping coefficients using the chosen criterion (either "WCC" or "RMS"). For "RMS", the data are smoothed before the optimization, but the quality of the final warping is measured on the unsmoothed data. For "WCC", the warping is performed using `trwdth` as the triangle width, but the quality of the final solution is measured using `trwdth.res`.

If `try = TRUE` is used as an argument, the function does not start an optimization, but just calculates the warping for the given warp function (`init.coef`); if `smooth.param` is larger than zero for the RMS criterion, the RMS of the smoothed patterns is calculated. The WCC criterion uses `trwidth.res` as the triangle width in this case.

Five situations can be distinguished:

1. One sample and one reference: this obviously leads to one warping function regardless of the setting of `warp.type`.
2. Several samples, all warped to the same single reference, each with its own warping function: this is the default behaviour (`warp.type = "individual"`)
3. Several samples, warped to an equal number of references (pair-wise), with their own warping functions: this is the default behaviour (`warp.type = "individual"`)
4. Several samples, warped to one reference, with one warping function (`warp.type = "global"`)
5. Several samples, warped to an equal number of references (pair-wise), with one warping function (`warp.type = "global"`)

Value

A list of class "ptw" containing:

`reference`	the reference(s) used as input
`sample`	the sample(s) used as input
`warped.sample`	the warped sample
`warp.coef`	the warping coefficients
`warp.fun`	the warped indices
`crit.value`	the value of the chosen criterion, either "WCC" or "RMS"
`optim.crit`	the chosen criterion, either "WCC" or "RMS"
`warp.type`	the chosen type of warping, either "individual" or "global"

Author(s)

Jan Gerretzen, Ron Wehrens

References

Eilers, P.H.C. (2004) "Parametric Time Warping", Analytical Chemistry, **76** (2), 404 – 411.

Bloemberg, T.G., et al. (2010) "Improved parametric time warping for Proteomics", Chemometrics and Intelligent Laboratory Systems, **104** (1), 65 – 74.

See Also

`WCC`, `RMS`, `select.traces`, `gaschrom`, `lcms`

Examples

```
data(gaschrom)
ref <- gaschrom[1,]
samp <- gaschrom[16,]
gaschrom.ptw <- ptw(ref, samp)
summary(gaschrom.ptw)

## same with sticks (peak lists)
refst <- gaschrom.st[1]
sampst <- gaschrom.st[16]
gaschrom.st.ptw <- stptw(refst, sampst, trwdth = 100)
summary(gaschrom.st.ptw)

## Not run:
## comparison between backward and forward warping
gaschrom.ptw <- ptw(ref, samp, init.coef = c(0, 1, 0, 0), mode = "backward")
summary(gaschrom.ptw)
gaschrom.ptw <- ptw(ref, samp, init.coef = c(-10, 1, 0, 0), mode = "forward")
summary(gaschrom.ptw)

## ############################
## many samples warped on one reference
ref <- gaschrom[1,]
samp <- gaschrom[2:16,]
gaschrom.ptw <- ptw(ref, samp, warp.type = "individual", verbose = TRUE,
                    optim.crit = "RMS", init.coef = c(0, 1, 0, 0))
summary(gaschrom.ptw)

## "individual" warping not implemented for sticks; do separate warpings
## instead
refst <- gaschrom.st[1]
sampst <- gaschrom.st[2:16]
gaschrom.st.ptw.list <- lapply(sampst,
                               function(smpl)
                               stptw(refst, list(smpl), trwdth = 100))
t(sapply(gaschrom.st.ptw.list, coef))

## ############################
## several samples on several references individually
ref <- gaschrom[1:8,]
samp <- gaschrom[9:16,]
gaschrom.ptw <- ptw(ref, samp, warp.type = "individual",
                    optim.crit = "RMS", init.coef = c(0, 1, 0, 0))
summary(gaschrom.ptw)

## stick version
gaschrom.st.ptw.list <-
  mapply(function(x, y)
         stptw(list(x), list(y), trwdth = 100),
         gaschrom.st[1:8], gaschrom.st[9:16],
         SIMPLIFY = FALSE)
t(sapply(gaschrom.st.ptw.list, coef))
```

```
gaschrom.ptw <- ptw(ref, samp, warp.type = "global",
                    optim.crit = "WCC", init.coef = c(0, 1, 0))
summary(gaschrom.ptw)

## ############################
## several samples on several references: one, global warping
refst <- gaschrom.st[1:8]
sampst <- gaschrom.st[9:16]
gaschrom.st.ptw <- stptw(refst, sampst, trwdth=100, init.coef = c(0, 1, 0))
summary(gaschrom.st.ptw)

## End(Not run)

## ############################
## Example of a three-way data set
# first bring all samples to the same scale
data(lcms)
## Not run:
lcms.scaled <- aperm(apply(lcms, c(1,3),
                           function(x) x/mean(x) ), c(2,1,3))
# add zeros to the start and end of the chromatograms
lcms.s.z <- aperm(apply(lcms.scaled, c(1,3),
                        function(x) padzeros(x, 250) ), c(2,1,3))


# define a global 2nd degree warping
warp1 <- ptw(lcms.s.z[,,2], lcms.s.z[,,3], warp.type="global")
warp.samp <- warp1$warped.sample
warp.samp[is.na(warp.samp)] <- 0
# refine by adding 5th degree warpings for individual chromatograms
warp2 <- ptw(lcms.s.z[,,2], warp.samp, init.coef=c(0,1,0,0,0,0))
warp.samp2 <- warp2$warped.sample
warp.samp2[is.na(warp.samp2)] <- 0
# compare TICs
layout(matrix(1:2,2,1, byrow=TRUE))
plot(colSums(lcms.s.z[,,2]), type="l", ylab = "",
     main = "TIC: original data")
lines(colSums(lcms.s.z[,,3]), col=2, lty=2)
plot(colSums(lcms.s.z[,,2]), type="l", ylab = "",
     main = "TIC: warped data")
lines(colSums(warp.samp2), lty=2, col=2)

## End(Not run)

## ############################
## stick version of this warping - note that the peaks have been picked
## from the scaled profiles. Note that here we need to take list
## elements: every sample is a list of mz channels.
warp1.st <- stptw(lcms.pks[[2]], lcms.pks[[3]], trwdth = 100)
summary(warp1.st)
```

ptwgrid *Calculate RMS or WCC values on a grid*

Description

Calculates values of the chosen optimization criterion (RMS or WCC) on a grid defined by the coefficients of the warping function.

Usage

```
ptwgrid(ref, samp, selected.traces,
coef.mins, coef.maxs, coef.lengths,
optim.crit = c("WCC", "RMS"),
smooth.param = 1e05,
trwdth = 20)
```

Arguments

`ref`	reference. Either a vector (containing one reference signal) or a matrix (one reference per row). If more than one reference is specified, the number of reference signals must equal the number of sample signals
`samp`	sample. A vector (containing one sample signal) or a matrix (one sample per row). If more than one sample is specified, the number of sample signals must equal the number of reference signals
`selected.traces`	optional vector containing the row numbers to use from `ref` (if more than one reference signal is specified) and `samp`
`coef.mins`	a vector containing the respective minimal values of coefficients for the grid. The first number is the minimal zeroth-order coefficient (i.e., a constant shift); further numbers indicate the minimal linear, quadratic, ... stretches
`coef.maxs`	a vector containing the maximal values of coefficients for the grid
`coef.lengths`	a vector of the same length as coef.maxs and coef.mins containing the number of steps in which to vary the respective coefficients between their minimum and maximum value
`optim.crit`	either `"WCC"` or `"RMS"`. In both cases, the optimal value of the alignment leads to a value of 0. For "WCC", this means that `1 - WCC` is optimized rather than `WCC` (where the optimal value equals 1)
`smooth.param`	smoothing parameter for smoothing the reference and sample when `optim.crit` equals `"RMS"`. If no smoothing is required, set this to 0.
`trwdth`	the width of the triangle in the WCC criterion during the optimization, given as a number of data points. Default: 20

Details

In contrast to ptw, only the three situations leading to one warping function are distinguished here:

1. One sample and one reference;
2. Several samples, warped to an equal number of references (pair-wise);
3. Several samples, warped to a single reference.

Which situation is applicable is determined from the dimensions of `ref` and `samp`.

Value

An array of dimension `length(coef.mins)` and maximal indices per dimension specified by `coef.lengths`

Author(s)

Tom Bloemberg, Jan Gerretzen, Ron Wehrens

See Also

ptw

Examples

```
## Not run:
data(gaschrom)
mygrid <- ptwgrid(gaschrom[1,], gaschrom[16,],
                  coef.mins = c(-10, .9), coef.max = c(10, 1.1),
                  coef.lengths = c(21, 21))
image(seq(-10, 10, length = 21),
      seq(.9, 1.1, length = 21),
      mygrid)

## End(Not run)
```

RMS *Quality criteria for comparing patterns with shifts*

Description

Functions to compare patterns with shifted features. These functions compare warped sample patterns to one or more reference patterns. RMS returns the usual root-mean-squared difference measure; WCC returns 1-wcc, where wcc indicates the weighted cross-correlation. Perfect alignment leads to a value of 0 for both criteria.

Internal function, not meant to be called directly by the user. In particular, note that the identity warping may lead to slightly different estimates than a direct comparison of the reference and sample signals - a warping even slightly outside the original range of `1 : ncol(ref)` leads to NA values.

Usage

```
RMS(warp.coef, ref, samp, B, mode)
WCC(warp.coef, ref, samp, B, trwdth = 20, wghts, mode, ref.acors = NULL)
```

Arguments

warp.coef	a vector of warping coefficients
ref	reference signal; a matrix with one or more rows. If the number of rows is greater than one, it should equal the number of rows in samp
samp	sample signal; a matrix with one or more rows
B	basis for warping function
mode	either "forward" (new implementation, also used for warping peak lists) or "backward" (classical implementation).
trwdth	triangle width for the WCC function, expressed in the number of data points
wghts	optional weights vector, will be calculated from triangle width if necessary. Sometimes it is more efficient to pre-calculate it and give it as an argument
ref.acors	autocorrelation of the reference. Since the reference is often unchanged over multiple evaluations (e.g., during an optimization), it is useful to pre-calculate this number

Details

All patterns in samp are warped using the same warping function, and then compared to ref, either pair-wise (when ref and samp are of the same size), or with the one pattern in ref.

Value

One number - either the WCC or RMS value

Author(s)

Jan Gerretzen, Tom Bloemberg, Ron Wehrens

References

Eilers, P.H.C. (2004) "Parametric Time Warping", Analytical Chemistry, **76** (2), 404 – 411.

de Gelder, R., Wehrens, R. and Hageman, J.A. (2001) "A generalized expression for the similarity of spectra: Application to powder diffraction pattern classification", Journal of Computational Chemistry, **22**, 273 – 289.

See Also

WCC

`select.traces` *Select traces from a data set according to several criteria*

Description

For alignment purposes, it may be useful to select traces which show clear features, and to throw away traces that contain mainly noise. This function implements three ways to achieve this: CODA, a criterion similar to varimax, and a criterion based on the highest intensity.

Usage

```
select.traces(X, criterion = c("coda", "var", "int"),
              window = 5, smoothing = c("median", "mean"))
```

Arguments

`X`	a data matrix or an array. The first dimension signifies the traces from which a selection is to be made. If X is a matrix, the first usually corresponds to samples and the second dimension is the spectral dimension. If X is an array, the data are assumed to come from a hyphenated experiment, with the first dimension the chromatographic dimension, the second the spectral dimension and the third dimension corresponding to samples
`criterion`	either Windig's CODA algorithm, a criterion calculating the variances of the length-scaled spectra, or a criterion giving the height of the highest peak
`window, smoothing`	arguments to the coda function.

Details

The CODA criterion in essence selects traces with no baseline and no spikes, but still containing significant intensities. The variance criterion aims at something similar: it calculates the variance (or standard deviation) of every trace after length scaling - traces with a high value show few highly structured features, whereas traces with a low value show noise or a significant baseline. The intensity criterion simply returns the intensity of the highest peak. The latter two criteria are simpler than CODA but implicitly assume that the traces have been preprocessed (i.e., spikes have been removed).

Value

The function returns a list with components

`crit.val`	a vector containing the values of the criterion for all traces. If X is an array, the function is recursively applied to all samples (elements of the third dimension) - the results are multiplied to obtain one criterion value per trace
`trace.nrs`	the order of the traces (from large to small)

Author(s)

Ron Wehrens

See Also

coda

Examples

```
data(lcms)
ntrace <- dim(lcms)[1]
lcms.selection <- select.traces(lcms[,,1:2], criterion = "var")
good <- lcms.selection$trace.nrs[1]
bad <- lcms.selection$trace.nrs[ntrace]

par(mfrow = c(1,2))
matplot(lcms[good,,1:2], type = 'l', lty = 1)
matplot(lcms[bad,,1:2], type = 'l', lty = 1)
```

---

warp.time *Transform time according to a given warping function*

Description

Warp time points according to a warping function.

Usage

```
warp.time(tp, coef)
```

Arguments

tp	A vector of time points, not necessarily equidistant.
coef	The coefficients of the parametric time warping function: the first coefficient is the constant shift, the second the linear stretch etcetera.

Value

The function returns a vector of warped time points.

Author(s)

Ron Wehrens

References

Wehrens, R. et al. (2015) "Fast parametric warping of peak lists", Bioinformatics. DOI: 10.1093/bioinformatics/btv299.

Examples

```
time <- 1:100
## simple shift and compression
warp.time(time, c(-10, .99))
```

---

wcc *Weighted auto- and cross-correlation measures*

Description

Functions to calculate weighted auto- and cross-correlation measures. The wcc is a suitable measure for the similarity of two patterns when features may be shifted. Identical patterns lead to a wcc value of 1.

Functions `wcc` and `wac` are meant for profile data (intensities measured at equidistant time points), whereas `wcc.st` and `wac.st` are meant for peak lists. In general, wcc values calculated for profiles will be higher since they will also include the large similarity in the empty spaces, i.e., parts of the profiles where no peaks are present (and that will appear to be perfectly aligned), whereas the peak-based version concentrates only on the peaks.

Usage

```
wcc(pattern1, pattern2, trwdth, wghts = NULL, acors1 = NULL, acors2 = NULL)
wac(pattern1, trwdth, wghts = NULL)
```

Arguments

`pattern1, pattern2`	input patterns, typically spectra. Vectors
`trwdth`	triangle width, given in the number of data points for the profile functions, and in the actual retention times for the stick-based warpings.
`wghts`	optional weights vector, will be calculated from triangle width if necessary. Sometimes it is more efficient to pre-calculate it and give it as an argument
`acors1, acors2`	autocorrelations of the input patterns. If not provided, they are calculated

Details

Functions `wcc` and `wac` are defined such that the triangle width stands for the number of points on one side of **and including** the current point. Thus, a `trwdth` of 0 signifies a non-existent triangle and results in an error; a `trwdth` equal to 1 only includes the current point with weight 1 and no neighbouring points. For the stick-based equivalents, the units of the time axis are used for the triangle width.

Value

One number, the weighted autocorrelation or crosscorrelation

Author(s)

Ron Wehrens

References

de Gelder, R., Wehrens, R. and Hageman, J.A. (2001) "A generalized expression for the similarity of spectra: Application to powder diffraction pattern classification", Journal of Computational Chemistry, **22**, 273 – 289.

Examples

```
data(gaschrom)
wcc(gaschrom[1,], gaschrom[2,], 20)

wcc.st(gaschrom.st[[1]], gaschrom.st[[2]], 20)
```

---

whit1 *Weighted Whittaker smoothing with a first order finite difference penalty*

Description

This function smoothes signals with a finite difference penalty of order 1.

Usage

```
whit1(y, lambda, w)
```

Arguments

y	signal to be smoothed: a vector
lambda	smoothing parameter: larger values lead to more smoothing
w	weights: a vector of same length as y. Default weights are equal to one

Value

smoothed signal: a vector

Author(s)

Paul Eilers, Jan Gerretzen

References

Eilers, P.H.C. (2004) "Parametric Time Warping", Analytical Chemistry, **76** (2), 404 – 411.

Eilers, P.H.C. (2003) "A perfect smoother", Analytical Chemistry, **75**, 3631 – 3636.

Examples

```
data(gaschrom)
plot(gaschrom[1,], type = "l", ylim = c(0, 100))
lines(whit1(gaschrom[1,], lambda = 1e1), col = 2)
lines(whit1(gaschrom[1,], lambda = 1e2), col = 3)
lines(whit1(gaschrom[1,], lambda = 1e3), col = 4)
```

---

whit2 *Weighted Whittaker smoothing with a second order finite difference penalty*

Description

This function smoothes signals with a finite difference penalty of order 2.

Usage

```
whit2(y, lambda, w)
```

Arguments

y	signal to be smoothed: a vector
lambda	smoothing parameter: larger values lead to more smoothing
w	weights: a vector of same length as y. Default weights are equal to one

Value

smoothed signal: a vector

Author(s)

Paul Eilers, Jan Gerretzen

References

Eilers, P.H.C. (2004) "Parametric Time Warping", Analytical Chemistry, **76** (2), 404 – 411.

Eilers, P.H.C. (2003) "A perfect smoother", Analytical Chemistry, **75**, 3631 – 3636.

Examples

```
data(gaschrom)
plot(gaschrom[1,], type = "l", ylim = c(0, 100))
lines(whit2(gaschrom[1,], lambda = 1e5), col = 2)
lines(whit2(gaschrom[1,], lambda = 1e6), col = 3)
lines(whit2(gaschrom[1,], lambda = 1e7), col = 4)
```

Index