Package: crops (via r-universe)

September 13, 2024

Type Package

Title Changepoints for a Range of Penalties (CROPS)

Version 1.0.3

Date 2022-08-05

Maintainer Daniel Grose <dan.grose@lancaster.ac.uk>

Description Implements the Changepoints for a Range of Penalties (CROPS) algorithm of Haynes et al. (2017) <doi:10.1080/10618600.2015.1116445> for finding all of the optimal segmentations for multiple penalty values over a continuous range.

License GPL

Imports sets, reshape, tidyverse, memoise, ggplot2, magrittr, methods, cowplot, tibble, Rdpack

Suggests fpop, pacman

RdMacros Rdpack

RoxygenNote 7.1.2

NeedsCompilation no

Author Daniel Grose [aut, cre], Paul Fearnhead [aut], Idris Eckley [ctb]

Repository CRAN

Date/Publication 2022-08-05 17:10:01 UTC

Contents

crops *Generic implementation of the crops algorithm (ref goes here).*

Description

Provides a generic implementation of the crops (changepoints for a range of penalties) algorithm of Haynes et al. (2014) which efficiently searches a range of penalty values in multiple changepoint problems. The crops algorithm finds the optimal segmentations for a different number of segments without incurring as large a computational cost as solving the constrained optimisation problem for a range of values for the number of changepoints. To make the method generic, the user must provide a function that maps a penalty value to the results obtained by a penalised cost changepoint method, and formats these results in a specific way. This interface to the generic method is similar to that as used by the **optimx** package.

Usage

```
crops(method, beta_min, beta_max, max_iterations = 20, ...)
```

Arguments

`method`	A function mapping a penalty value to the results obtained by a penalised cost changepoint method. The function must return a list containing the cost and a vector of changepoint locations corresponding to the optimal segmentation as determined by a penalised cost changepoint method.
`beta_min`	A positive numeric value indicating the smallest penalty value to consider.
`beta_max`	A positive numeric value indicating the maximum penalty value to consider.
`max_iterations`	Positive non zero integer. Limits the maximum number of iterations of the crops algorithm to `max_iterations`. Default value is `max_iterations=20`
`...`	Additional parameters to pass to the underlying changepoint method if required.

Value

An instance of an S4 class of type `crops.class`.

References

Haynes K, Eckley IA, Fearnhead P (2017). "Computationally Efficient Changepoint Detection for a Range of Penalties." *Journal of Computational and Graphical Statistics*, **26**(1), 134-143. doi:10.1080/10618600.2015.1116445.

Nash JC, Varadhan R (2011). "Unifying Optimization Algorithms to Aid Software System Users: optimx for R." *Journal of Statistical Software*, **43**(9), 1–14. https://www.jstatsoft.org/v43/i09/.

Nash JC (2014). "On Best Practice Optimization Methods in R." *Journal of Statistical Software*, **60**(2), 1–14. https://www.jstatsoft.org/v60/i02/.

Nash JC (2021). *optimx: Expanded Replacement and Extension of the 'optim' Function*. R package version 2021-6.12.

Maidstone R, Hocking T, Rigaill G, Fearnhead P (2017). "On optimal multiple changepoint algorithms for large data." *Statistics and Computing*, **27**. https://link.springer.com/article/10.1007/s11222-016-9636-3.

Rigaill G (2019). *fpop: Segmentation using Optimal Partitioning and Function Pruning*. R package version 2019.08.26.

Examples

```
# generate some simple data
set.seed(1)
N <- 100
data.vec <- c(rnorm(N), rnorm(N, 2), rnorm(N))

# example one - calling fpop via crops using global scope
# need the fpop library
library(pacman)
p_load(fpop)
# create a function to wrap a call to fpop for use with crops
fpop.for.crops<-function(beta)
    {
       # Note - this code is taken from the example in the fpop package
       fit <- Fpop(data.vec, beta)
       end.vec <- fit$t.est
       change.vec <- end.vec[-length(end.vec)]
       start.vec <- c(1, change.vec+1)
       segs.list <- list()
       for(seg.i in seq_along(start.vec))
           {
            start <- start.vec[seg.i]
            end <- end.vec[seg.i]
            seg.data <- data.vec[start:end]
            seg.mean <- mean(seg.data)
            segs.list[[seg.i]] <- data.frame(
                                  start, end,
                                  mean=seg.mean,
                                  seg.cost=sum((seg.data-seg.mean)^2))
            }
        segs <- do.call(rbind, segs.list)
        return(list(sum(segs$seg.cost),segs$end[-length(segs$end)]))
    }

# now use this wrapper function with crops
res<-crops(fpop.for.crops,0.5*log(300),2.5*log(300))
# print summary of analysis
summary(res)
# summarise the segmentations
segmentations(res)
# visualise the segmentations
plot(res)
# overlay the data on the segmentations
```

```
df <- data.frame("x"=1:300,"y"=data.vec)
plot(res,df)
```

`plot` *Visualisation of data, costs, penalty values and changepoint locations.*

Description

Plot methods for an S4 object returned by `crops`. The plot can also be combined with the original data if required.

Usage

```
## S4 method for signature 'crops.class,data.frame'
plot(x, y)

## S4 method for signature 'crops.class,missing'
plot(x)
```

Arguments

`x`	An instance of an S4 class produced by `crops`.
`y`	A dataframe containing the locations and values of the data points. The data plot is plotted below, and is aligned with, the changepoint locations.

Value

A ggplot object. Note - if no changepoints are detected in the penalty interval [beta_min,beta_max], then the value returned is NULL.

See Also

`crops`.

Examples

```
# see the crops example

# see the crops example
```

print *Pretty printing for crops results*

Description

Pretty prints a summary of a crops result

Usage

```
## S4 method for signature 'crops.class'
print(x)
```

Arguments

x	An instance of an S4 class produced by crops.

See Also

crops.

Examples

```
# see the crops example
```

segmentations *Summary of segmentations by penalty value*

Description

Produces a summary of the segmentations for each penalty value in the form of a data frame.

Usage

```
segmentations(object)
```

Arguments

object	An instance of an S4 class produced by crops.

Value

A data frame containing the penalties, costs, penalised costs, and changepoint locations. Note - if no changepoints are detected in the penalty interval [beta_min,beta_max], then the value returned is NULL.

See Also

crops.

Examples

```
# see the crops example
```

---

subset *Subset crops results based on penalty values*

Description

Removes entries from a crops result that fall outside a specified range of penalty values. The subset function can be useful for simplifying plots and the details produced by segmentations.

Usage

```
## S4 method for signature 'crops.class'
subset(x, beta_min = 0, beta_max = Inf)
```

Arguments

x	An instance of an S4 class produced by crops.
beta_min	A positive numeric value specifying the minimum penalty value for entries in the crops result. Default value is 0.
beta_max	A positive numeric value specifying the maximum penalty value for entries in the crops result. Default value is Inf.

Value

An instance of the S4 class type crops.class. This is the same type as produced by the crops function.

---

summary *Summary of crops result*

Description

Prints a short summary of a crops result.

Usage

```
## S4 method for signature 'crops.class'
summary(object)
```

Arguments

object	An instance of an S4 class produced by crops.

See Also

crops.

Examples

```
# see the crops example
```

---

unique *Remove duplicate entries from a crops result*

Description

Removes duplicate entries from a crops result. A duplicate entry is one having the same number of changepoints as another entry. Note that the changepoint locations and the associated penalty and cost values are not taken into consideration. The unique function can be useful for simplifying plots and the details produced by segmentations.

Usage

```
## S4 method for signature 'crops.class'
unique(x)
```

Arguments

x	An instance of an S4 class produced by crops.

Value

An instance of the S4 class type crops.class. This is the same type as produced by the crops function.

Index