# Package 'fpop'

July 22, 2025

**Type** Package

**Title** Segmentation using Optimal Partitioning and Function Pruning

**Version** 2019.08.26

**Maintainer** Guillem Rigaill <guillem.rigaill@inra.fr>

**Description** A dynamic programming algorithm for the fast segmentation of univariate signals into piecewise constant profiles.
The 'fpop' package is a wrapper to a C++ implementation of the fpop (Functional Pruning Optimal Partioning) algorithm described in Maidstone et al. 2017 <doi:10.1007/s11222-016-9636-3>. The problem of detecting changepoints in an univariate sequence is formulated in terms of minimising the mean squared error over segmentations. The fpop algorithm exactly minimizes the mean squared error for a penalty linear in the number of changepoints.

**License** LGPL (>= 2.1)

**NeedsCompilation** yes

**Author** Guillem Rigaill [aut, cre],
Toby Hocking [aut],
Robert Maidstone [aut],
Michel Koskas [ctb],
Paul Fearnhead [aut]

**Repository** CRAN

**Date/Publication** 2019-08-27 07:00:03 UTC

## Contents

fpop-package . . . . . 2
Fpop . . . . . 2
fpop_analysis . . . . . 4
multiBinSeg . . . . . 4
retour_op . . . . . 5

**Index** **6**

`fpop-package` *Segmentation using optimal partioning and functional pruning*

## Description

A wrapper to a C implementation of optimal partioning with functional pruning

## Details

| | |
|---|---|
| Package: | fpop |
| Type: | Package |
| Title: | Segmentation using optimal partioning and functional pruning |
| Version: | 2014.7.16 |
| Depends: | methods, cghseg |
| SystemRequirements: | GNU GSL |
| Date: | 2014-02-26 |
| Author: | Guillem Rigaill |
| Maintainer: | Guillem Rigaill <rigaill@evry.inra.fr> |
| License: | LGPL (>= 2.1) |

## Author(s)

Guillem Rigaill

`Fpop` *Fpop*

## Description

Function calling the fpop algorithm, use functional pruning and optimal partioning to recover the best segmentation with respect to the L2 loss with a per change-point penalty of lambda. More precisely, this function computes the solution to argmin_m sum_i=1^n (x_i-m_i)^2 + lambda * sum_i=1^n-1 I(m_i != m_i+1), where the indicator function I counts the number of changes in the mean vector m.

## Usage

```
Fpop(x, lambda, mini = min(x), maxi = max(x))
```

## Arguments

| | |
|---|---|
| x | A vector of double : the signal to be segmented |
| lambda | Value of the penalty |
| mini | Min value for the mean parameter of the segment |
| maxi | Max value for the mean parameter of the segment |

## Value

Named list with the following elements: input data (signal, n, lambda, min, max), path (best previous segment end up to each data point), cost (optimal penalized cost up to each data point), t.est (vector of overall optimal segment ends), K (optimal number of segments), J.est (total un-penalized cost of optimal model). To see how cost relates to J.est, see definition of J.est in the R source code for this function.

## Author(s)

Guillem Rigaill, Toby Dylan Hocking

## Examples

```
set.seed(1)
N <- 100
data.vec <- c(rnorm(N), rnorm(N, 2), rnorm(N))
fit <- Fpop(data.vec, N)
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)
plot(data.vec)
with(segs, segments(start-0.5, mean, end+0.5, mean, col="green"))
with(segs[-1,], abline(v=start-0.5, col="green", lty="dotted"))
```

`fpop_analysis` *fpop analysis*

## Description

A function to count the number of intervals and or candidate segmentation at each step of fpop (under-developpemment)

## Usage

```
fpop_analysis(x, lambda, mini = min(x), maxi = max(x))
```

## Arguments

| | |
|---|---|
| `x` | A vector of double : the signal to be segmented |
| `lambda` | Value of the penalty |
| `mini` | Min value for the mean parameter of the segment |
| `maxi` | Max value for the mean parameter of the segment |

## Value

return a list with a vector containing the position of the change-points t.est

## Author(s)

Guillem Rigaill, Toby Dylan Hocking

`multiBinSeg` *multiBinSeg*

## Description

Binary segmentation of p profiles using the L2 loss

## Usage

```
multiBinSeg(geno, Kmax)
```

## Arguments

| | |
|---|---|
| `geno` | A matrix with p columns and n lines, each column is one of the profile |
| `Kmax` | Maximum number of change-points |

## Value

return an object with the successive change-points found by binseg t.est and the L2 cost J.est

### Author(s)

Guillem Rigaill, Toby Dylan Hocking

---

`retour_op` *retour op*

---

### Description

This function is used by the Fpop function to recover the best segment ends from 1:n from the C output.

### Usage

```
retour_op(path)
```

### Arguments

| | |
|---|---|
| `path` | the path vector of the "colibri_op_R_c C" function |

### Value

a vector with the best segment ends.

### Author(s)

Guillem Rigaill, Toby Dylan Hocking

# Index

∗ **package**
fpop-package, 2

Fpop, 2
fpop *(*fpop-package*)*, 2
fpop-package, 2
fpop_analysis, 4

multiBinSeg, 4

retour_op, 5