Package ‘bulletr’

April 26, 2017

Title  Algorithms for Matching Bullet Lands
Version  0.1
License  GPL-3
Date  2017-04-25
Description  Analyze bullet lands using nonparametric methods. We provide a reading routine for x3p files (see <http://www.openfmc.org> for more information) and a host of analysis functions designed to assess the probability that two bullets were fired from the same gun barrel.

URL  https://github.com/erichare/bulletr
BugReports  https://github.com/erichare/bulletr/issues
Imports  xml2, zoo, ggplot2, plyr, dplyr, reshape2, plotly, robustbase, smoother
Depends  R (>= 3.1)
RoxygenNote  6.0.1
LazyData  true
NeedsCompilation  no
Author  Eric Hare [aut, cre], Heike Hofmann [aut]
Maintainer  Eric Hare <erichare@iastate.edu>
Repository  CRAN
Date/Publication  2017-04-26 05:31:52 UTC

R topics documented:

boot_fit_loess .................................................. 2
br411 ......................................................... 3
bulletAlign ..................................................... 3
bulletCheckCrossCut .......................................... 4
bulletGetMaxCMS ............................................. 4
bulletGetMaxCMS_nist ....................................... 5
boot_fit_loess

Fit a LOESS model with bootstrap samples

Description

Fit a LOESS model with bootstrap samples

Usage

boot_fit_loess(bullet, groove, B = 1000, alpha = 0.95)

Arguments

bullet        Bullet as returned from fortify_x3p
groove       Groove as returned from get_grooves
B            number of Bootstrap samples
alpha        The significance level
3d topological surface measurements for one land of a bullet from the Hamby study

Description
Some more info - not sure at the moment which bullet this is. Describe structure.

Usage

bulletAlign(data, value = "l30")

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>data frame consisting of at least two surface crosscuts as given by function bulletSmooth.</td>
</tr>
<tr>
<td>value</td>
<td>string of the variable to match. Defaults to l30, the variable returned from function bulletSmooth.</td>
</tr>
</tbody>
</table>

Value
list consisting of a) the maximal cross correlation, b) the lag resulting in the highest cross correlation, and c) same data frame as input, but y vectors are aligned for maximal correlation between the
bulletCheckCrossCut  Identifying a reliable cross section

Description

Should be changed: x should just indicate lower and upper limit. That is cleaner and should speed things up as well.

Usage

bulletCheckCrossCut(path, bullet = NULL, distance = 25, xlimits = c(50, 500), minccf = 0.9, span = 0.03)

Arguments

- path: path to an x3p file
- bullet: If passed in, the actual bullet already loaded
- distance: positive numeric value indicating the distance between cross sections to use for a comparison
- xlimits: vector of values between which to check for cross sections in a stable region
- minccf: minimal value of cross correlation to indicate a stable region
- span: The span for the loess smooth function

bulletGetMaxCMS  Identify the number of maximum CMS between two bullet lands

Description

Identify the number of maximum CMS between two bullet lands

Usage

bulletGetMaxCMS(lof1, lof2, column = "resid", span = 35)

Arguments

- lof1: dataframe of smoothed first signature
- lof2: dataframe of smoothed second signature
- column: The column which to smooth
- span: positive number for the smoothfactor to use for assessing peaks.

Value

list of matching parameters, data set of the identified striae, and the aligned data sets.
**bulletGetMaxCMS_nist**  
*Identify the number of maximum CMS between two bullet lands*

**Description**
Identify the number of maximum CMS between two bullet lands

**Usage**
bulletGetMaxCMS_nist(lof1, lof2, column = "resid", span = 35)

**Arguments**
- **lof1**: dataframe of smoothed first signature
- **lof2**: dataframe of smoothed second signature
- **column**: The column which to smooth
- **span**: positive number for the smoothfactor to use for assessing peaks.

**Value**
list of matching parameters, data set of the identified striae, and the aligned data sets.

---

**bulletSmooth**  
*Smooth the surface of a bullet*

**Description**
Smooth the surface of a bullet

**Usage**
bulletSmooth(data, span = 0.03, limits = c(-5, 5))

**Arguments**
- **data**: data frame as returned by the function processBullets
- **span**: width of the smoother, defaults to 0.03
- **limits**: vector of the form c(min, max). Results will be limited to be between these values.

**Value**
data frame of the same form as the input extended by the vector l30 for the smooth.
**CMS**  
*Table of the number of consecutive matches*

**Description**  
Table of the number of consecutive matches

**Usage**  
CMS(match)

**Arguments**  
match is a Boolean vector of matches/non-matches

**Value**  
a table of the number of the CMS and their frequencies

**Examples**  
x <- rbinom(100, size = 1, prob = 1/3)  
CMS(x == 1) # expected value for longest match is 3

---

**fit_loess**  
*Fit a loess curve to a bullet data frame*

**Description**  
First, the surface measurements of the bullet land is trimmed to be within left and right groove as specified by vector groove. A loess regression is fit to the remaining surface measurements and residuals are calculated. The most extreme 0.25 The result is called the signature of the bullet land.

**Usage**  
fit_loess(bullet, groove, span = 0.75)

**Arguments**  
bullet The bullet object as returned from fortify_x3p  
groove vector of two numeric values indicating the location of the left and right groove.  
span The span to use for the loess regression

**Value**  
a list of a data frame of the original bullet measurements extended by loess fit, residuals, and standard errors and two plots: a plot of the fit, and a plot of the bullet’s land signature.
**fortify_x3p**

*Convert a list of x3p file into a data frame*

**Description**

x3p format consists of a list with header info and a 2d matrix of scan depths. fortify_x3p turn the matrix into a variable within a data frame, using the parameters of the header as necessary.

**Usage**

```r
fortify_x3p(x3p)
```

**Arguments**

- `x3p` a file in x3p format as return by function `read_x3p`

**Value**

data frame with variables `x`, `y`, and `value`

**Examples**

```r
data(br411)
br411_fort <- fortify_x3p(br411)
head(br411_fort)
```

---

**getCircle**

*Estimate center and radius*

**Description**

Assuming the variables `x` and `y` are describing points located on a circle, the function uses a likelihood approach to estimate center and radius of the circle.

**Usage**

```r
getCircle(x, y)
```

**Arguments**

- `x` numeric vector of values
- `y` numeric vector of values

**Value**

three dimensional vector of the circle center `(x0, y0)` and the radius
getTwist

Estimate the twist in a bullet land

Description

Estimation of the twist in a barrel follows roughly the process described by Chu et al (2010). At the moment, twist is estimated from a single land - but the twist should be the same for the whole barrel. Therefore all lands of the same barrel should have the same twist. A note on timing: at the moment calculating the twist rate for a bullet land takes several minutes. XXX TODO XXX make the different methods a parameter. Also, accept other input than the path - if we start with the flattened bulletland we get results much faster.

Usage

getTwist(path, bullet = NULL, twistlimit = NULL, cutoff = 0.75)

Arguments

- path to a file in x3p format
- bullet data in x3p format as returned by function read_x3p
- twistlimit Constraint the possible twist value
- cutoff Use this for the quantile cutoff

Value

numeric value estimating the twist

Examples

## Not run:
# execution takes several minutes
load("data/b1.rda")
twist <- getTwist(path="barrel 1 bullet 1", bullet = b1, twistlimit=c(-2,0)*1.5625)

## End(Not run)

get_bullet

Deprecated function use get_crosscut

Description

Deprecated function use get_crosscut

Usage

get_bullet(path, x = 243.75)
get_crosscut

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>The path to the x3p file</td>
</tr>
<tr>
<td>x</td>
<td>The crosscut value</td>
</tr>
</tbody>
</table>

Description

Read a crosscut from a 3d surface file

Usage

get_crosscut(path = NULL, x = 243.75, bullet = NULL)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>path to an x3p file. The path will only be considered, if bullet is not specified.</td>
</tr>
<tr>
<td>x</td>
<td>level of the crosscut to be taken. If this level does not exist, the crosscut with the closest level is returned.</td>
</tr>
<tr>
<td>bullet</td>
<td>alternative access to the surface measurements.</td>
</tr>
</tbody>
</table>

Value

data frame

grooves

Find the grooves of a bullet land

Description

Find the grooves of a bullet land

Usage

grooves(bullet, smoothfactor = 15, adjust = 10, groove_cutoff = 400, mean_left = NULL, mean_right = NULL, mean_window = 100)
**Arguments**

- **bullet**: data frame with topological data
- **smoothfactor**: The smoothing window to use
- **adjust**: positive number to adjust the grooves
- **groove_cutoff**: The index at which a groove cannot exist past
- **mean_left**: If provided, the location of the average left groove
- **mean_right**: If provided, the location of the average right groove
- **mean_window**: The window around the means to use

---

**get_peaks**

*Identify the location and the depth of peaks and heights at a crosscut*

---

**Description**

Identify the location and the depth of peaks and heights at a crosscut

**Usage**

```
get_peaks(loessdata, column = "resid", smoothfactor = 35, striae = TRUE, window = TRUE)
```

**Arguments**

- **loessdata**: export from rollapply
- **column**: The column which should be smoothed
- **smoothfactor**: set to default of 35. Smaller values will pick up on smaller changes in the crosscut.
- **striae**: If TRUE, show the detected striae on the plot
- **window**: If TRUE, show the window of the striae on the plot

**Value**

list of several objects:
get_peaks_nist

Identify the location and the depth of peaks and heights at a crosscut

Description

Identify the location and the depth of peaks and heights at a crosscut

Usage

get_peaks_nist(loessdata, column = "resid", smoothfactor = 35, striae = TRUE, window = TRUE)

Arguments

loessdata export from rollapply
column The column which should be smoothed
smoothfactor set to default of 35. Smaller values will pick up on smaller changes in the cross-cut.
striae If TRUE, show the detected striae on the plot
window If TRUE, show the window of the striae on the plot

Value

list of several objects:

maxCMS Number of maximum consecutively matching striae

Description

Number of maximum consecutively matching striae

Usage

maxCMS(match)

Arguments

match is a Boolean vector of matches/non-matches

Value

an integer value of the maximum number of consecutive matches
Examples

```r
x <- rbinom(100, size = 1, prob = 1/3)
CMS(x == 1) # expected value for longest match is 3
maxCMS(x==1)
```

Description

Plot a bullet land using plotly

Usage

```r
plot_3d_land(path, bullet = NULL)
```

Arguments

- **path**: The path to the x3p file
- **bullet**: If not null, use this pre-loaded bullet

Description

Estimate predictions and residuals for a circle fit of x and y

Usage

```r
predCircle(x, y, resid.method = "response")
```

Arguments

- **x**: vector of numeric values
- **y**: vector of numeric values
- **resid.method**: character, one of "response" or "ortho"(gonal)

Value

- data frame with predictions and residuals
**predSmooth**

Estimate predictions and residuals for a smooth of x and y

**Description**

Fit a smooth line through x and y, find predictive values and residuals.

**Usage**

predSmooth(x, y)

**Arguments**

- **x**: vector of numeric values
- **y**: vector of numeric values

**Value**

data frame with predictions and residuals

**processBullets**

Process x3p file

**Description**

x3p file of a 3d topological bullet surface is processed at surface crosscut x, the bullet grooves in the crosscuts are identified and removed, and a loess smooth is used (see ?loess for details) to remove the big structure.

**Usage**

processBullets(bullet, name = "", x = 100, grooves = NULL, span = 0.75, window = 0, ...)

**Arguments**

- **bullet**: file as returned from read_x3p
- **name**: name of the bullet
- **x**: (vector) of surface crosscuts to process.
- **grooves**: The grooves to use as a two element vector, if desired
- **span**: The span for the loess fit
- **window**: The mean window around the ideal crosscut
- **...**: Additional arguments, passed to the get_grooves function
sample_x3p

Value
data frame

Examples
data(br411)
br411_processed <- processBullets(br411, name = "br411")

---

read_x3p

Read an x3p file as an R Data Frame

Description
Read an x3p file as an R Data Frame

Usage
read_x3p(path, profiley = TRUE)

Arguments
path The file path to the x3p file
profiley If TRUE, rotate the matrix (if necessary) to ensure a profile is taken across y

Examples
## Not run:
br411 <- read_x3p("Br4 Bullet 4-1.x3p")
## End(Not run)

---

sample_x3p

Sample every X element of a data frame

Description
Sample every X element of a data frame in x and y direction

Usage
sample_x3p(dframe, byxy = c(2, 2))
smoothloess

**Arguments**

- `df` (data frame) - data frame with x and y variable
- `byxy` (vector) - numeric value indicating the sampling resolution. If a single number, the same resolution is used for x and y.

**Value**

- Subset of the input variable

**Examples**

```r
data(brT11)
brT11_fort <- fortify_xSp(brT11)
brT11_sample <- sample_xSp(brT11_fort, byxy = c(4, 4))
head(brT11_sample)
```

---

**smoothloess**  
*Predict smooth from a fit*

**Description**

Predict smooth from a fit

**Usage**

```
smoothloess(x, y, span, sub = 2)
```

**Arguments**

- `x` - X values to use
- `y` - Y values to use
- `span` - The span of the loess fit
- `sub` - Subsample factor
striation_identify  
*Match striation marks across two cross sections based on previously identified peaks and valleys*

**Description**

Match striation marks across two cross sections based on previously identified peaks and valleys

**Usage**

`striation_identify(lines1, lines2)`

**Arguments**

- `lines1`: data frame as returned from `get_peaks` function. Data frames are expected to have the following variables: `xmin`, `xmax`, `group`, `type`, `bullet`, `heights`
- `lines2`: data frame as returned from `get_peaks` function. Data frames are expected to have the following variables: `xmin`, `xmax`, `group`, `type`, `bullet`, `heights`

**Value**

data frame of the same form as `lines1` and `lines2`, but consisting of an additional variable of whether the striation marks are matches

---

unfortify_x3p  
*Convert a data frame into an x3p file*

**Description**

Convert a data frame into an x3p file

**Usage**

`unfortify_x3p(df)`

**Arguments**

- `df`: A data frame produced by `fortify_x3p`

**Value**

An x3p object
Examples

data(br411)
br411_fort <- fortify_x3p(br411)
br411_unfort <- unfortify_x3p(br411_fort)
identical(br411_unfort, br411)
Index

*Topic datasets
  br411, 3

boot_fit_loess, 2
br411, 3
bulletAlign, 3
bulletCheckCrossCut, 4
bulletGetMaxCMS, 4
bulletGetMaxCMS_nist, 5
bulletSmooth, 5

CMS, 6

fit_loess, 6
fortify_x3p, 7

get_bullet, 8
get_crosscut, 9
get_grooves, 9
get_peaks, 10
get_peaks_nist, 11
getCircle, 7
getTwist, 8

maxCMS, 11

plot_3d_land, 12
predCircle, 12
predSmooth, 13
processBullets, 13

read_x3p, 14

sample_x3p, 14
smoothloess, 15
striation_identify, 16

unfortify_x3p, 16