Package ‘HilbertVis’

May 29, 2024

Version 1.62.0
Date 2013-10-07
Title Hilbert curve visualization
Author Simon Anders <sanders@fs.tum.de>
Maintainer Simon Anders <sanders@fs.tum.de>
Depends R (>= 2.6.0), grid, lattice
Suggests IRanges, EBImage
Description Functions to visualize long vectors of integer data by means of Hilbert curves
License GPL (>= 3)
URL http://www.ebi.ac.uk/~anders/hilbert
biocViews Visualization
git_url https://git.bioconductor.org/packages/HilbertVis
git_branch RELEASE_3_19
git_last_commit 2c928e6
git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-29

Contents

hilbertCurve ......................................................... 2
hilbertImage ....................................................... 3
makeRandomTestData .............................................. 4
makeWiggleVector ................................................ 5
plotHilbertCurve .................................................. 6
plotLongVector ..................................................... 6
showHilbertImage ................................................... 7
shrinkVector ......................................................... 8

Index 10


**Description**

These functions calculate the Hilbert curve in its finite approximations. `hilbertCurvePoint` gives the coordinates of one point and `hilbertCurve` returns an array with the coordinates of all $4^lv$ points. The functions are not needed for `hilbertImage` and only provided for demonstration purposes. `plotHilbertCurve` makes use of them.

**Usage**

```r
hilbertCurve( lv )
hilbertCurvePoint( t, lv )
```

**Arguments**

- `lv` The iteration level. A Hilbert curve of level $lv$ spans a square with side length $2^lv$ (coordinates ranging from 0 to $2^lv-1$) and has $4^lv$ points.
- `t` The point index in the Hilbert curve. Must be an integer in $0:(4^lv-1)$.

**Value**

- `hilbertCurvePoint` returns a vector of two integer numbers, both in the range $0:(2^lv-1)$, indicating the coordinates of point $t$. `hilbertCurve` returns a matrix with $4^lv$ rows and 2 columns, giving all points of the curve at level $lv$.

**Author(s)**

Simon Anders, EMBL-EBI, <sanders@fs.tum.de>

**See Also**

`plotHilbertCurve`

**Examples**

```r
hilbertCurvePoint( 67, 4 )
hilbertCurve( 4 )
```
hilbertImage

Produce a matrix that visualizes a long data vector along a Hilbert curve

Description

Calculate a Hilbert curve visualization of a long data vector and return it as a square matrix.

Usage

hilbertImage(data, level = 9, mode = "absmax")

Arguments

data: A (potentially very long) vector of numerical data.
level: The level of the Hilbert curve, determining the size of the returned matrix.
mode: The binning mode. See `shrinkVector` for details.

Details

See the package vignette for an explanation of this visualization technique.

Value

A matrix of dimension $2^{\text{level}} \times 2^{\text{level}}$. Each matrix element corresponds to a bin of consecutive elements of the data vector, the bins arranged to follow the Hilbert curve of the given level. By default, the value of a matrix element is either the largest or smallest element in the bin, whichever is larger by absolute value. (See `shrinkVector` for other possible binning modes.)

To display such a matrix graphically, you can use the standard functions `image` or `levelplot` but the function `showHilbertImage` may be more convenient.

Note

For an interactive GUI to explore a Hilbert curve visualisation, use the function `hilbertDisplay` in the `HilbertVisGUI` package.

Author(s)

Simon Anders, EMBL-EBI, sanders@fs.tum.de

Examples

```r
# Get a vector with example data
dataVec <- makeRandomTestData()

# Plot it in conventional (linear) fashion
plotLongVector(dataVec)
```
# makeRandomTestData

Make random test data.

```r

makeRandomTestData

makeRandomTestData(len = 1e+07, numPeaks = 500)
```

## Arguments

- `len`: Length of the vector.
- `numPeaks`: Number of peaks to be placed in the vector.

## Value

A vector, of type 'numeric', with sample data.

## See Also

`hilbertImage`
**makeWiggleVector**

*generate a "wiggle vector" from start/end/value data*

**Description**
Given intervals in the form of a "start" and an "end" vectors and corresponding values, generate a "wiggle vector" of a given length that contains the specified values in the vector elements indicated by the intervals.

**Usage**

```r
makeWiggleVector(start, end, value, chrlength)
```

**Arguments**

- **start**
  The start coordinates of the intervals. As usual in R, these are 1-based.

- **end**
  The end coordinates of the intervals. As usual, the end points are included.

- **value**
  The values to be put in the wiggle vector. Where intervals overlap, the values are added.

- **chrlength**
  The desired length of the returned vector.

**Value**

A vector as described above.

**Author(s)**

Simon Anders, EMBL-EBI, sanders\@fs.tum.de

**See Also**

For a value vector containing only ones, this function acts similar as the pileup function in the ShortRead package.

**Examples**

```r
intervalStarts <- c(3,10,17,22)
intervalEnds <- c(7,13,20,26)
values <- c(2, 1.5, .3, 4)
chrlength <- 30
wig <- makeWiggleVector(intervalStarts, intervalEnds, values, chrlength)
# The same effect can be achieved with the following R code, which, however
# is much slower:
wig2 <- numeric(chrlength)
for( i in 1:length(values) )
  wig2[ intervalStarts[i]:intervalEnds[i] ] <-
    wig2[ intervalStarts[i]:intervalEnds[i] ] + values[i]
# Let's check that we got the same:
all( wig == wig2 )
```
plotHilbertCurve  Plotting the Hilbert curve (for demonstration purposes).

Description

This function plots the Hilbert curve fractal at a chosen iteration level in order to give you an impression how it looks like.

Usage

plotHilbertCurve( lv, new.page = TRUE )

Arguments

lv The iteration level. A Hilbert curve of level lv spans a square with side length $2^lv$ (coordinates ranging from 0 to $2^lv-1$) and has $4^lv$ points. Values $lv > 7$ will take very long and yield a cluttered mesh of indistinguishable lines.

new.page Boolean indicating whether to start a new graphics page (default: yes).

Value

An invisible NULL is returned. Furthermore, a plot is created.

Author(s)

Simon Anders, EMBL-EBI, <sanders@fs.tum.de>

See Also

hilbertCurve

Examples

plotHilbertCurve( 3 )

plotLongVector  A simple function to plot a very long vector.

Description

This function does basically the same as just calling plot(vec) but is much faster in case of a very long vector. This is because it first calls shrinkVector.

Usage

plotLongVector(vec, offset = 1, shrinkLength = 4000, xlab = "", ylab = "", ...)


**showHilbertImage**

**Arguments**

- **vec**
  The numerical vector to be plotted. May be an ordinary or an IRanges::Rle vector.
- **offset**
  The x axis is labelled with numbers from offset to offset+length(vec)-1.
- **shrinkLength**
  To which length to shrink the vector before plotting it. Should be at least the width of your plot in pixels.
- **xlab**
  The label of the x axis, to be passed to `plot`.
- **ylab**
  The label of the y axis, to be passed to `plot`.
- **...**
  Further arguments to be passed to `plot`.

**Value**

Invisible Null and a plot.

**Author(s)**

Simon Anders, EMBL-EBI, sanders@fs.tum.de

**Examples**

```r
plotLongVector( rep( 1:100000, 20 ) )
```

**showHilbertImage**

*display a hilbert*

**Description**

A convenient wrapper around `levelplot` to display a hilbert image matrix as it is returned by `hilbertImage`. Alternatively to `levelplot`, `EBImage` is available as well.

**Usage**

```r
showHilbertImage( mat,
                  palettePos = colorRampPalette(c("white", "red"))(300),
                  paletteNeg = colorRampPalette(c("white", "blue"))(300),
                  maxPaletteValue = max(abs(mat)),
                  mode = c("lattice", "EBImage", "EBImage-batch") )
```

**Arguments**

- **mat**
  The matrix to be displayed. In principle this can be any matrix, but typically, it is one returned by `hilbertImage`.
- **palettePos**
  The colour palette to be used for the positive entries in `mat` (including 0).
- **paletteNeg**
  The colour palette to be used for the negative entries in `mat`. 
shrinkVector

maxPaletteValue
The absolute value to which the right end of the palettes should correspond.
(The left ends correspond to 0.)

mode
For mode "lattice", the function levelplot from the lattice package is used.
An (invisible) lattice object is returned that can be displayed with show. In
interactive mode, the image is displayed automatically. For mode "EBImage"
the image is displayed with the EBImage package, and for "EBImage-batch",
the same image is produced and not displayed but rather returned as a value
suitable to be passed to EBImage's display function.

Value
A lattice or EBImage graphics object. For all modes except “EBImage-batch” it is marked “invisi-
ble”.

Author(s)
Simon Anders, EMBL-EBI (sanders\@fs.tum.de)

See Also
hilbertImage

Examples
  # See ?hilbertImage for examples.

shrinkVector(vec, newLength, mode = c("max", "min", "absmax", "mean"))

Description
Given a (potentially very long) vector, the vector is partitioned into a given number of (up to round-
ing errors) equally long bins, and a vector summarizing each of the bins with one number it returned.

Usage
shrinkVector(vec, newLength, mode = c("max", "min", "absmax", "mean"))

Arguments
vec
The vector to be shrunk. May be an ordinary numeric or integer vector or an
IRanges::Rle vector.

newLength
The desired size of the return vector, i.e., the number of partitions

mode
the summerization mode: 'max': take the maximal value of each bin; 'min':
take the minimal value of each bin; 'absmax': take the value with largest abso-
lute value; 'mean': take the mean of the bin values.
**shrinkVector**

**Value**

A vector of length `newLength` with the summary values of each of the bin of `vector`.

**Author(s)**

Simon Anders, EMBL-EBI (sanders\@fs.tum.de)

**See Also**

`plotLongVector`, `Rsamtools::pileup`, `HilbertVisGui::simpleLinPlot`

**Examples**

```r
shrinkVector( 100000 + 1:1000, 17 )
```
Index

hilbertCurve, 2, 6
hilbertCurvePoint (hilbertCurve), 2
hilbertDisplay, 3
hilbertImage, 2, 3, 7, 8
HilbertVisGui::simpleLinPlot, 9
makeRandomTestData, 4
makeWiggleVector, 5

plot, 7
plotHilbertCurve, 2, 6
plotLongVector, 6, 9

Rsamtools::pileup, 9

showHilbertImage, 3, 7
shrinkVector, 3, 6, 8