Package ‘lle’

February 20, 2015

Type Package
Title Locally linear embedding
Version 1.1
Date 2012-03-21
Author Holger Diedrich, Dr. Markus Abel (Department of Physics, University Potsdam)
Maintainer Holger Diedrich <holgerniedrich@gmx.net>
Depends scatterplot3d, MASS, snowfall
Suggests rgl

Description LLE is a non-linear algorithm for mapping high-dimensional data into a lower dimensional (intrinsic) space. This package provides the main functions to performs the LLE algorithm including some enhancements like subset selection, calculation of the intrinsic dimension etc.

License GPL-3
LazyLoad yes
Repository CRAN
Date/Publication 2012-03-21 12:52:58
NeedsCompilation no

R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>calc_k</td>
<td>2</td>
</tr>
<tr>
<td>find_coords</td>
<td>3</td>
</tr>
<tr>
<td>find_nn_eps</td>
<td>4</td>
</tr>
<tr>
<td>find.nn_k</td>
<td>5</td>
</tr>
<tr>
<td>find_weights</td>
<td>5</td>
</tr>
<tr>
<td>lle</td>
<td>6</td>
</tr>
<tr>
<td>lle_rectangular</td>
<td>8</td>
</tr>
<tr>
<td>lle_scurve</td>
<td>9</td>
</tr>
<tr>
<td>lle_scurve_data</td>
<td>10</td>
</tr>
</tbody>
</table>
Description

Calculates of optimal number of neighbours by using the algorithm proposed by Kayo (see Ref.). Therefore the LLE algorithm is performed for every k-neighbourhood size.

Usage

calc_k(X, m, kmin=1, kmax=20, plotres=TRUE, parallel=FALSE, cpus=2, iLLE=FALSE)

Arguments

- **x** matrix object containing the input data.
- **m** intrinsic dimension of the data.
- **kmin** minimal value of k.
- **kmax** maximal value of k.
- **plotres** a logical values indicating whether to plot the result.
- **parallel** a logical values indicating whether to use parallel computation on multiple cpu cores. See `snowfall`.
- **cpus** number of cpus cores used for parallel computation.
- **iLLE** a logical values indicating wheather to use improved LLE (very CPU intensive). See `lle`.

Details

Since the calculation of the optimal number of neighbours m is a step that is normally applied before the execution of LLE itself, the intrinsic dimension may be unknown. In this case, a good guess is sufficient. If no good estimation can be made the largest plausible value should be chosen.

Value

- **res** dataframe containing the number of neighbours and the calculated parameter $\rho$. The number of neighbours belonging to the smallest value of $\rho$ should be chosen.
find_coords

References
Locally linear embedding algorithm - extensions and applications / Olga Kayo / Universitatis Oulu-ensis, Oulu, Finland / 2006

Examples

```
## Not run:
data( lle_scurve_data )
x <- lle_scurve_data
calc_k( x, 2, 1, 15 )

data( lle_scurve_data )
x <- lle_scurve_data
calc_k( x, 2, 1, 15, FALSE, TRUE, 4 )
## End(Not run)
```

find_coords Calculate embedded data.

Description
Calculates the embedded data of dimension \( m \) using the weight matrix.

Usage
```
find_coords(wgts, nns, N, n, m)
```

Arguments
- `wgts` weight matrix calculated by `find_weights`.
- `nns` matrix of neighbours calculated by `find_nn_k` or `find_nn_eps`.
- `N` number of samples.
- `n` dimension of the original data.
- `m` intrinsic dimension of the data.

Value
```
Y matrix containing the embedded data.
```

Examples
```
data( lle_scurve_data )
X <- lle_scurve_data
nns <- find_nn_k(X,5)
wgts <- find_weights(nns,X,2)
Y <- find_coords( wgts$wgts, nns, dim(X)[1], 3, 2 )
```
find_nn_eps

Find nearest neighbours in epsilon environment.

Description

Finds the nearest points for every point in an environment with radius eps.

Usage

find_nn_eps(x, eps)

Arguments

x matrix object containing the input data.
eps size of epsilon environment around $x_i$.

Details

A good value for eps strongly depends on the scaling of the data. Therefore we recommend to use the R function scale.

Value

neighbours matrix with $N$ rows and columns. If distance between $x_i$ and $x_j$ is smaller than eps, $\text{neighbours}[i,j]$ is one, else zero.

See Also

find_nn_k

Examples

data( l1e_scurve_data )
X <- l1e_scurve_data
neighbours <- find_nn_eps( X, 0.5 )
table( rowSums(neighbours) )
**find_nn_k**

*Find k nearest neighbours.*

**Description**

Finds the nearest k points for every point of the input data.

**Usage**

```r
find_nn_k(X, k, iLLE = FALSE)
```

**Arguments**

- `X`: matrix object containing the input data.
- `k`: number of neighbours.
- `iLLE`: a logical values indicating wheater to use improved LLE. See `lle`.

**Value**

- `neighbours`: matrix with \( N \) rows and columns. If \( x_j \) is a neighbour of \( x_i \) then \( \text{neighbours}[i,j] \) is one, else zero.

**See Also**

`find_nn_eps`

**Examples**

```r
data(lle_scurve_data)
X <- lle_scurve_data
neighbours <- find_nn_k(X, 5)
table(rowSums(neighbours))
```

---

**find_weights**

*Calculate weight matrix.*

**Description**

Calculates the weights for every neighbour of \( x_i \).

**Usage**

```r
find_weights(nns, X, m, reg = 2, ss = FALSE, p = 0.5, id = FALSE, v = 0.99)
```
Arguments

- **nns**: matrix of nearest neighbours using `find_nn_k` or `find_nn_eps`.
- **X**: matrix object containing the input data.
- **m**: intrinsic dimension of the data. See **lle**.
- **reg**: regularisation method. See **lle**.
- **ss**: a logical values indicating whether to perform subset selection. See **lle**.
- **p**: amount of data remaining after subset selection. See **lle**.
- **id**: a logical values indicating whether to calculate the intrinsic dimension. See **lle**.
- **v**: threshold parameter for intrinsic dimension. See details.

Value

A list containing the following variables:

- **X**: input data, can change if subset selection is applied
- **weights**: weight matrix. If \( x_i \) is neighbour of \( x_j \) then \(-1 < weights[i, j] < 1\), else zero.
- **choise**: index vector of kept data while subset selection
- **id**: additionally to the (optional) printed intrinsic dimension, the vector of intrinsic dimension for every data point is returned by the function, so that the vector can easily be plotted manually.

Examples

```r
data( lle_scurve_data )
X <- lle_scurve_data
nns <- find_nn_k(X, 5)
weights <- find_weights(nns, X, 2, 2)
```

**lle**  
*Locally linear embedding main function.*

Description

Performs all steps of LLE algorithm by calling the functions of the package.

Usage

```r
lle(X, m, k, reg = 2, ss = FALSE, p = 0.5, id = FALSE, nnk = TRUE, eps = 1, iLLE = FALSE, v = 0.99)
```
Arguments

- **X**: matrix object containing the input data.
- **m**: intrinsic dimension of the data. This parameter mainly influences the visualisation of the results. The real intrinsic dimension will be calculated automatically.
- **k**: number of neighbours. Optimal number can be calculated using the `calc_k`.
- **reg**: regularisation method. Choose between 1, 2 and 3, by default 2. See details.
- **ss**: a logical values indicating wheather to perform subset selection. See details.
- **p**: amount of data remaining after subset selection. Values between 0 and 1.
- **id**: a logical values indicating wheather to calculate the intrinsic dimension.
- **nnk**: a logical values indicating wheather to use k nearest neighbours method. If false, epsilon environment neighbourhood will be used.
- **eps**: epsilon radius if parameter nnk is FALSE.
- **iLLE**: a logical values indicating wheater to use improved LLE after Wang. See details.
- **v**: threshold parameter for intrinsic dimension. See details.

Details

This is the main function to execute the LLE algorithm. Given a data matrix with \( N \) rows (samples) and \( n \) columns (dimensions), the embedded data of dimension \( m \) will be calculated.

As described above, the parameter \( m \) influences the visualisation of the embedded data. Therefore see `plot_lle`.

If \( id \) is true, the intrinsic dimension of the data is automatically calculated during the execution of the function. Since the intrinsic dimension is calculated for every data point \( x_i \) the result of this calculation consists of a vector with length \( N \). The used approach is to calculate the mean and the mode of this vector as representation of the overall intrinsic dimension of the data.

The \( reg \) parameter allows the decision between different regularisation methods. As one step of the LLE algorithm, the inverse of the Gram-matrix \( G \in \mathbb{R}^{k \times k} \) has to be calculated. The rank of \( G \) equals \( m \) which is mostly smaller than \( k \) - this is why a regularisation \( G^{(i)} + r \cdot I \) should be performed. The calculation of regularisation parameter \( r \) can be done using different methods:

- \( reg=1 \): standardized sum of eigenvalues of \( G \), see Ref. 1), Ch. 3.2
- \( reg=2 \): trace of Gram-matrix divided by \( k \), see Ref. 2), Ch. 5.2
- \( reg=3 \): constant value 3*10e-3

There is no theoretical evidence which method is best to use but several empirical analyses have shown that method \#2 works the most reliable.

The most time-consuming step of LLE consists in the calculation of the eigenvalues and -vectors of matrix \( M \in \mathbb{R}^{N \times N} \) in the `find_coords` function. To reduce the dimension of matrix \( M \), which means to reduce the number of samples \( N \) in a reliable way, Ref. 1 proposes a subset selection algorithm, which is integrated in the `lle` function. The amount of data that is kept is represented by parameter \( p \).

Improved LLE (iLLE) is an extension of the LLE algorithm described in Ref. 3. It raises the required amount of memory and time, but makes the algorithm less dependent on the number of neighbours.

Calculating the intrinsic dimension strongly depends on a threshold value \( v \). The best value for this
parameter depends on the origin of the data. For very accurate data a value beyond 0.99 is proposed, for very raw data a value of 0.9 is proposed. This parameter should be varied if a specific intrinsic dimension is expected and other results are calculated. Higher values of $v$ lead to a higher number of calculate intrinsic dimensions.

**Value**

A list containing the following variables:

- $X$ input data, can change if subset selection is applied
- $Y$ embedded data
- `choise` (only if `ss`=TRUE) index vector of kept data while subset selection
- `id` (only if `id`=TRUE) vector of intrinsic dimension for every data point.

**References**

2) Automated Local Linear Embedding with an application to microarray data / Elisa Grilli / Universita di Bologna, Italy / 2005
3) Improved Locally Linear Embedding Through New Distance Computing / Heyong Wang et al / Sun Yat-sen University, China / 2010

**Examples**

```r
# perform LLE
data( lle_scurve_data )
X <- lle_scurve_data
results <- lle( X=X, m=2, k=12, reg=2, ss=FALSE, id=TRUE, v=0.9 )
str(results)

# plot results and intrinsic dimension (manually)
split.screen( c(2,1) )
screen(1)
plot( results$Y, main="embedded data", xlab=expression(y[1]), ylab=expression(y[2]) )
screen(2)
plot( results$id, main="intrinsic dimension", type="l", xlab=expression(x[i]), ylab="id", lwd=2 )
```

---

**Example on a rectangular signal.**
**lle_rectangular**

**Description**

A rectangular signal with variable width and distance is generated and analysed (see figure). Every sample has dimension \( n = 500 \). In the example \( N \) samples are generated, that only differ by the two parameters width and distance. As a result it can be seen that the data can be embedded from a 500- into a 2-dimensional space which’s limits are specified by the limits of the two parameters.

![Diagram](image)

**Usage**

\[
\text{lle_rectangular}(N = 40, k = 5, v = 0.9)
\]

**Arguments**

- **\( N \)**: number of samples.
- **\( k \)**: number of neighbours. See **lle**.
- **\( v \)**: threshold parameter for intrinsic dimension. See **lle**.

**Examples**

```r
## Not run:
lle_rectangular()
lle_rectangular(30, 10, 0.8) \\
## End(Not run)
```

**lle_scurve**

*Example application on a three-dimensional S-curve dataset.*

**Description**

Performs the LLE algorithm on a three dimensional S-curve, which is a standard example for data embedding algorithms.

**Usage**

\[
\text{lle_scurve}(N = 800, k = 12, ss = FALSE, p = 0.5, reg = 2, 
\quad ILL = FALSE, v = 0.8)
\]
**Arguments**

- \( N \)  
  number of samples. See `lle`.

- \( k \)  
  number of neighbours. See `lle`.

- \( ss \)  
  a logical values indicating wheather to perform subset selection. See `lle`.

- \( p \)  
  amount of data remaining after subset selection. See `lle`.

- \( \text{reg} \)  
  regularisation method. See `lle`.

- \( \text{iLLE} \)  
  a logical values indicating wheather to use iLLE. See `lle`.

- \( v \)  
  threshold parameter for intrinsic dimension. See `lle`.

**Examples**

- `lle_scurve()`
- `lle_scurve( N=1800, k=11, ss=TRUE )`

---

**Description**

Syntheticly generated empiric data of a three-dimensional S-curve.

**Usage**

- `data(lle_scurve_data)`

**Format**

- matrix with dimensions \( n = 1000, m = 3 \)

**See Also**

- `lle_scurve`
**lle_sound**

**Description**

Performs the LLE algorithm on a soundfile. As a first step, the sound is scanned piecewise with time windows sized $t$. In every scanning step, the time windows is shifted by $dt$. This generates an overall data set with dimension $n = t$ and sample size $N \approx t/dt$. This dataset can then be embedded using LLE algorithm.

**Usage**

```r
lle_sound(t = 500, dt = 20, k = 25, reg = 2, ss = FALSE,
         p = 0.5, id = TRUE)
```

**Arguments**

- **t**: time window used to scan.
- **dt**: time window shift.
- **k**: number of neighbours. See `lle`.
- **reg**: regularisation method. See `lle`.
- **ss**: a logical values indicating whether to perform subset selection. See `calc_k`.
- **p**: amount of data remaining after subset selection. See `lle`.
- **id**: a logical values indicating whether to calculate the intrinsic dimension. See `lle`.

**Examples**

```r
## Not run:
lle_sound()
lle_sound(200, 20, 20)

## End(Not run)
```

**lle_spiral**

**Example application on a n-dimensional spiral**

**Description**

In this example nine three-dimensional spirals are generated. Every spiral has a higher number of windings than the previous. The first spiral’s intrinsic dimension is determined as $m = 1$. It can be observed that a higher number of windings with constant sampling rate leads to a higher calculated intrinsic dimension. Due to this fact, the last spiral’s intrinsic dimension is computed as $m = 2$.

**Usage**

```r
lle_spiral()
```
### lle_swissrole

**Example application on a swissrole.**

**Description**

Like the S-curve this is a standard example to demonstrate the functioning of LLE.

**Usage**

```r
lle_swissrole(N = 1500, k = 10, ss = FALSE, p = 0.5, reg = 2, iLLE = FALSE, v = 0.8)
```

**Arguments**

- `N`: number of samples.
- `k`: number of neighbours. See `lle`.
- `ss`: a logical values indicating wheather to perform subset selection. See `lle`.
- `p`: amount of data remaining after subset selection. See `lle`.
- `reg`: regularisation method. See `lle`.
- `iLLE`: a logical values indicating wheater to use iLLE. See `lle`.
- `v`: threshold parameter for intrinsic dimension. See `lle`.

**Examples**

```r
## not run:
lle_swissrole()
lle_swissrole( 3000, k=12, ss=TRUE )

## End(not run)
```

### lle_wave

**wavefile**

**Description**

synthetically generated wavefile with 10 modes.

**Usage**

```r
data(lle_wave)
```

**Format**

numeric vector of length 17640 (0.4 seconds, 44 kHz)

**Details**

To analyse an acustic signal a wavefile with 10 modes was generated and transformed into text.
Description

Function for plotting LLE results either in static or in dynamic way.

Usage

plot_lle( Y, X, print = FALSE, col = 3, name = as.numeric(Sys.time()), angle = 60, inter = FALSE )

Arguments

Y matrix object with calculated embedded data.
X matrix object with original data.
print a logical values indicating wheather to plot the graphical results to a file.
col string or number dtermining the plotting colours.
name (if print==true) filename.
angle (if inter==false) angle between x- and y-axis in scatterplot3d. See documenta-
tion of scatterplot3d.
inter a logical values indicating wheather to use interactive 3D-plots. See rgl.

Details

col determines the way that the points in the plot are coloured. Choosing a string name of a colour
leads to a monocoloured plot. Choosing a number between leads to a colour gradient plot build
up by \( N \) colours (taking only the rainbow colours into account). Choosing a numeric vector with
length \( N \) leads to points coloured respecting to the values in the vector (for unique colours only
values between 0 and 600 should be used).
If inter==false, two plots are generated in one window. The left plot is the plot of the original
data. These will only be plotted if \( n \in \{1, 2, 3\} \). The right plot shows the embedded data. These
will only be plotted \( m \in \{1, 2, 3\} \).
if inter==true, one interactive plot of the embedded data will be shown. This plot can be scrolled
and zoomed. It requires OpenGL drivers.

Examples

data( lle_scurve_data )
X <- lle_scurve_data
Y <- lle( X, m=2, k=12 )$Y
plot_lle( Y, X, FALSE, col="red", inter=TRUE )
Index

calc_k, 2, 7, 11
find_coords, 3
find_nn_eps, 3, 4, 5, 6
find_nn_k, 3, 4, 5, 6
find_weights, 3, 5

lle, 2, 5, 6, 6, 9–12
lle_rectangular, 8
lle_scurve, 9, 10
lle_scurve_data, 10
lle_sound, 11
lle_spiral, 11
lle_swissrole, 12
lle_wave, 12

plot_lle, 7, 13