Package ‘tapkee’

October 14, 2022

Type Package

Title Wrapper for 'tapkee' Dimension Reduction Library

Version 1.2

Date 2020-12-20

Author Alexey Shipunov

Maintainer Alexey Shipunov <dactylorhiza@gmail.com>

Description Wrapper for using 'tapkee' command line utility,
it allows to run it from inside R and catch the results for further analysis and plotting.
'Tapkee' is a program for fast dimension reduction, see 'package?tapkee' and <http://tapkee.lisitsyn.me/>
for installation and other details.

SystemRequirements 'tapkee' (http://tapkee.lisitsyn.me/)

Suggests scatterplot3d, rgl, R.rsp

VignetteBuilder R.rsp

License GPL (>= 2)

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2021-01-04 12:00:02 UTC

R topics documented:

tapkee-package *'tapkee' command line utility installation*

Description

Here is the description of how to install 'tapkee' utility on different operation systems.

General instructions

Download Executablle files for macOS, Windows and Linux are available here: https://github.com/lisitsyn/tapkee/releases/ta

Specific instructions for Linux

Downloaded binary: Linux binary is a 64 bit version which works on Ubuntu 16.04 LTS and likely will work on other systems. Depends on 'ldd tapkee', install dependencies (you might need to install at least "libarpack2"). Copy 'tapkee' binary to where system will find it ('echo $PATH').

Specific instructions for macOS

1) In Terminal.app, run:
$ echo $PATH

2) Copy 'tapkee' into one of mentioned directories, e.g.:
$ cp tapkee /usr/local/bin

3) Run:
$ tapkee -h
If you see the list of 'tapkee' options, everything is OK. If not, you need to check (4) and (5).

4) If you have the message similar to:
$ dyld: Library not loaded: ...
install eigen and arpack with Homebrew (install Homebrew first, google how to do it):
$ brew install arpack && brew install eigen
then run
$ tapkee -h
again.

5) You might also run:
$ echo $DYLD_LIBRARY_PATH
to see where installed libraries should be located.

Specific instructions for Windows

1) Download the 'tapkee-win.exe' executable which works under Windows 10. Please also make sure that you installed "Microsoft Visual C++ Redistributable 32-bit" (google the name to obtain link) to get required DLLs.

2) Find where did you place the executable file. Open command prompt and direct it to this folder, for example (if the 'tapkee-win.exe' is in "C:\Users\Me\Downloads" directory):

> c: > cd "C:\Users\Me\Downloads"

Then check is the executable works. Type

> tapkee-win.exe -h

Long message with options should appear. If not, you are either not in the proper place (double check where is executable file and then "cd" to there), or there are some other problems (e.g., required DLLs are not installed properly).

Next, find the place where to install 'tapkee-win.exe'. Best is to start the command prompt window and run there:

> echo

Install the executable (and change its name to just 'tapkee.exe') into one of folders which are in the list. For example, if "C:\Some dir" is in the list (entries there are separated with semicolon), run:

> copy tapkee-win.exe "C:\Some dir\tapkee.exe"

You might also try to install everything in the R working folder (to find it, run 'getwd()' inside R) but this is less usable.

3) Now open new command prompt window and run:

> tapkee -h

If you see the list of 'tapkee' options, everything is OK. If not, you need to check (1) and (2) again.

`Gen.dr.data` *Generates 3D data*

Description

Generates typical 3D dimension reduction data

Usage

```
Gen.dr.data(type, N=1000)
```

Arguments

`type`	one of "swissroll", "scurve" (S-curve), "helix", "ssphere" (severed sphere)
`N`	number of data points

Details

'Gen.dr.data()' generates some frequently used 3D data. Formulas taken partly from 'tapkee' 'borsch' script and partly from Python 'scikit-learn'.

Author(s)

Alexey Shipunov

Examples

```
# generate four variants of 3D data

SC <- Gen.dr.data("scurve")
SR <- Gen.dr.data("swissroll")
HX <- Gen.dr.data("helix")
SS <- Gen.dr.data("ssphere")

# plot them (requires packages 'scatterplot3d' and 'rgl')

if (requireNamespace("rgl", quietly = TRUE)) {
 COL <- colorRampPalette(c("green", "orange"))(1000)
 scatterplot3d::scatterplot3d(SC, color=COL, pch=20, cex.symbols=1.4)
} else {
  cat("Please install 'rgl' package to see the plot")
}

if (requireNamespace("rgl", quietly = TRUE)) {
 rgl::plot3d(SR, col=rainbow(1100))
} else {
 cat("Please install 'rgl' package to see the plot")
}

if (requireNamespace("rgl", quietly = TRUE)) {
 rgl::plot3d(HX, col=rainbow(1100))
} else {
 cat("Please install 'rgl' package to see the plot")
}

if (requireNamespace("rgl", quietly = TRUE)) {
 rgl::plot3d(SS, col=rainbow(1100))
} else {
  cat("Please install 'rgl' package to see the plot")
}
```

Tapkee *Tapkee wrapper*

Description

R wrapper for the ’tapkee’ dimension reduction library

Usage

```
Tapkee(data, method="pca", td=2, verbose=FALSE, add="", prefix="Dim", rm=TRUE)
```

Arguments

`data`	R numerical matrix or data frame (will be converted into matrix)
`method`	'tapkee' method, run "system('tapkee -h')" for the list, default is "pca"
`td`	Number of dimensions to output, default is 2
`verbose`	If TRUE, 'tapkee' is verbose, defalut is FALSE
`add`	'tapkee' additional arguments as character string: see "system('tapkee -h')"
`prefix`	Variable name prefix in the resulted data frame, default is "Dim"
`rm`	Remove temp files (but temp folder will be removed anyway in the end of R session), default is TRUE

Details

Interface (wrapper) for the 'tapkee', flexible and efficient C++ template library for dimension reduction. 'tapkee' is extremely fast comparing with other DR tools.

For methods used in 'tapkee', run 'vignette(tapkee_methods)'.

Users should install 'tapkee' independently from author Web site (https://github.com/lisitsyn/tapkee) or associated GitHub (https://github.com/lisitsyn/tapkee). Run 'package?tapkee' or help("tapkee-package") for details related with your operation system. If 'tapkee' is not installed, Tapkee() will fail gracefully and output the input data with warning.

Please note that "[warning] The neighborhood graph is not connected" message in most cases means that 'tapkee' run was unsuccessful. As a result, Tapkee() might return the matrix of NaN's. One of possible workarounds is to specify the higher number of neigbors ('-k' option, default is 10). See below for the example.

Note that the wrapper catches only one (main) type of 'tapkee' utility outputs. For other possible output types (see 'tapkee -h' for explanation), run 'tapkee' without wrapper.

Value

Data frame with number of columns equal to number of dimensions given and "prefix" column names prefixes.

Author(s)

Alexey Shipunov

References

Sergey Lisitsyn and Christian Widmer and Fernando J. Iglesias Garcia. Tapkee: An Efficient Dimension Reduction Library. Journal of Machine Learning Research, 14: 2355-2359, 2013.

See Also

`tapkee-package`

Examples

```
## 'tapkee' vs. R base functions
system.time(Tapkee(iris[, -5], method="mds"))
system.time(cmdscale(dist(iris[, -5])))

## How to use 'add' option
plot(Tapkee(iris[, -5], "isomap", add="-k 47"), col=iris[, 5])

## 'tapkee' methods as of March 2019:
TM <- c(
"lle", # 1) locally_linear_embedding (lle),
"npe", # 2) neighborhood_preserving_embedding (npe),
"ltsa", # 3) local_tangent_space_alignment (ltsa),
"lltsa", # 4) linear_local_tangent_space_alignment (lltsa),
"hlle", # 5) hessian_locally_linear_embedding (hlle),
"la", # 6) laplacian_eigenmaps (la),
"lpp", # 7) locality_preserving_projections (lpp),
"dm", # 8) diffusion_map (dm),
"isomap", # 9) isomap (isomap),
"l-isomap", # 10) landmark_isomap (l-isomap),
"mds", # 11) multidimensional_scaling (mds),
"l-mds", # 12) landmark_multidimensional_scaling (l-mds),
"spe", # 13) stochastic_proximity_embedding (spe),
"kpca", # 14) kernel_pca (kpca),
"pca", # 15) pca (pca),
"ra", # 16) random_projection (ra),
"fa", # 17) factor_analysis (fa),
"t-sne", # 18) t-stochastic_neighborhood_embedding (t-sne),
"ms") # 19) manifold_sculpting (ms)

## Iris example
oldpar <- par(mfrow=c(4, 5), mar=c(1, 1, 3, 1), xaxt="n", yaxt="n")
for (n in c(1:18)) {
plot(Tapkee(iris[, -5], method=TM[n], add="-k 50"),
 col=iris[, 5], pch=20, main=TM[n], xlab="", ylab="")
}
plot(iris[, 1:2], col=iris[, 5], pch=20, main="iris[, 1:2]", xlab="", ylab="")
par(oldpar)

## Generate typical 3D data
SR <- Gen.dr.data("swissroll")
SC <- Gen.dr.data("scurve")
HX <- Gen.dr.data("helix")
SS <- Gen.dr.data("ssphere")

## This will separate colors better
COL <- rainbow(1100)[1:1000]

## 3D plot (if no 'scatterplot3d' package, plots XY axes)
T3D <- function(dat, title, col=COL) {
 if (requireNamespace("scatterplot3d", quietly = TRUE)) {
 scatterplot3d::scatterplot3d(dat, color=col, main=title, pch=20, xlab="", ylab="", zlab="",
  axis=FALSE, tick.marks=FALSE, label.tick.marks=FALSE, mar=c(1, 1, 3, 1))
} else {
 plot(dat[, 1:2], col=col, main=paste(title, "(2D projection)"), pch=20)
 warning("Please install 'scatterplot3d' package to see the 3D plot")
}}

## Swiss Roll
oldpar <- par(mfrow=c(4, 5))
T3D(SR, title="Swiss Roll")
for (n in 1:18) plot(Tapkee(SR, method=TM[n]), col=COL, pch=20, main=TM[n],
 xlab="", ylab="", xaxt="n", yaxt="n")
par(oldpar)

## S-Curve
oldpar <- par(mfrow=c(4, 5))
T3D(SC, title="S-Curve")
for (n in 1:18) plot(Tapkee(SC, method=TM[n]), col=COL, pch=20, main=TM[n],
 xlab="", ylab="", xaxt="n", yaxt="n")
par(oldpar)

## Helix
oldpar <- par(mfrow=c(4, 5))
T3D(HX, title="Helix")
for (n in 1:18) plot(Tapkee(HX, method=TM[n]), col=COL, pch=20,
 main=TM[n], xlab="", ylab="", xaxt="n", yaxt="n")
par(oldpar)

## Severed Sphere
oldpar <- par(mfrow=c(4, 5))
T3D(SS, title="Severed Sphere", col=rainbow(nrow(SS)))
for (n in 1:18) plot(Tapkee(SS, method=TM[n]), col=rainbow(nrow(SS)), pch=20,
 main=TM[n], xlab="", ylab="", xaxt="n", yaxt="n")
par(oldpar)
```

Index