hoggormplot Documentation

Release 0.12.0

Oliver Tomic

Oct 01, 2020

Content

hoggormplot is a complementary plotting package designed specifically for visualising results of multivariate statistical models computed with the hoggorm package.

CHAPTER 1

Quickstart

1.1 Requirements

Make sure that Python 3.5 or higher is installed. A convenient way to install Python and many useful packages for scientific computing is to use the Anaconda distribution.

Required Python packages

- numpy
- hoggorm >= 0.11.0
- matplotlib >= 2.1.1

1.2 Installation and updates

1.2.1 Installation

Install hoggormplot easily from the command line from the PyPI - the Python Packaging Index.

```
pip install hoggormplot
```

1.2.2 Upgrading

To upgrade hoggormplot from a previously installed older version execute the following from the command line:

```
pip install --upgrade hoggormplot
```

If you need more information on how to install Python packages using pip, please see the pip documentation.

1.3 Documentation

- Documentation at Read the Docs
- Jupyter notebooks with examples of how to use hoggormplot

1.4 Example

```
import hoggormplot as hopl

# Compute PCA model with
# - 5 components
# - standardised/scaled variables
# - KFold cross validation with 4 folds
>>> model = ho.nipalsPCA(arrX=myData, numComp=5, Xstand=True, cvType=["Kfold", 4])

# Extract results from PCA model
>>> scores = model.X_scores()
>>> loadings = model.X_loadings()
>>> cumulativeCalibratedExplainedVariance_allVariables = model.X_cumCalExplVar_
↪indVar()
>>> cumulativeValidatedExplainedVariance_total = model.X_cumValExplVar()

# Plot results with HoggormPlot
# Get multiple plots with the main hoggormplot function
>>> hopl.plot(model, plots=[1, 2, 3, 6], cumulative=True, line=True)
>>> hopl.plot(model)
>>> hopl.plot(model, plots=['scores', 'loadings', 'explainedVariance'],␣
↪cumulative=True)
```

CHAPTER 2

Main plotting function

This is the main plotting function for visualsation of results from models computed with the hoggorm package. All convenience functions make use of the main plotting function.

hoggormplot.main_plot.**plot**(*model, comp=[1, 2], plots=[1, 2, 3, 4], which=[], line=False, weights=False, cumulative=True, individual=False, validated=[], objNames=[], XvarNames=[], YvarNames=[], newX=[], newY=[], newObjNames=[], figsize=None*)

This is the main plot function that generates plots that visualise results from PCA, PCR, PLSR and PLSR2 models computed with the Hoggorm package.

Parameters

- **model** (*nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm*) – The statistical results of the submitted model will be visualized.
- **comp** (*list, optional*) – The list contains components to be displayed. Defaults to [1,2].
- **plots** (*list, optional*) – Integers in this list define which plots are to be generated (combined in a single figure). The list defaults to [1, 2, 3, 4]. The following options are available:
  - 1 : Scores (default: X)
  - 2 : Loadings (default: X)
  - 3 : Correlation loadings (default: Both (X & Y))
  - 4 : Biplot (default: X)
  - 5 : Regression coefficients
  - 6 : Explained variance (default: Y)
  - 7 : Prediction

  Instead of integers, string arguments may be useds. The following options are available:

  - 'scores'
  - 'loadings'
  - 'correlationLoadings'
  - ' biplot'
  - 'coeffs'
  - 'explainedVariance'
  - 'predict'
- **which** (*list, optional*) – This list may contain one string argument. The following options are available:
  - 'X'
  - 'Y'
  - 'Both' (defaults listed with 'plots' parameter)
- **line** (*boolean, optional*) – When set to 'line=TRUE' loadings (weights) will be plotted as lines / spectra instead of as points in a scatter plot.
- **weights** (*boolean, optional*) – When set to 'weights=TRUE' loading weights will be plotted instead of loadings.
- **cumulative** (*boolean, optional*) – When set to 'cumulative=TRUE' explained variances will be plotted cumulatively instead of per component.
- **individual** (*boolean, optional*) – When set to 'individual=TRUE' explained variances will be plotted per variable instead of for all variables together.
- **validated** (*list, optional*) – When set to 'validated=[TRUE]' validated values are plotted if applicable (scores => False, explainedVariance => True).
- **objNames** (*list, optional*) – Object names may be provided in this list.
- **XvarNames** (*list, optional*) – Names of variables in array X may be provided in this list.
- **YvarNames** (*list, optional*) – Names of variables in Y may be provided in this list.
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type Multiple plots.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.plot(myModel, plots=[1, 2, 3, 6], cumulative=True, line=True)
>>> hopl.plot(myModel)
>>> hopl.plot(myModel, plots=['scores', 'loadings', 'explainedVariance'],␣
↪cumulative=True)
```

CHAPTER 3

Convenience plotting functions

This is an overview of available **convenience plotting functions** in hoggormplot. They are based on the main plot function `hoggorm.main_plot.plot` and provide simplified interfaces for interactive use.

3.1 Explained variance plot

Note: The two functions below, i.e. `.explVar` and `.explainedVariance` are two options of how to generate explained variance plots. They generate identical plots when provided the same input paramters or arguments.

`hoggormplot.conv_explainedVariancePlot.`**`explVar`**(*model*, *which=[]*, *cumulative=True*, *individual=False*, *validated=[]*, *figsize=None*)

This function generates explained variances plots of hoggorm models.

Parameters

- **`model`** (*`nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm.`*) –
- **`which`** (*`list, optional`*) – This list may contain one string argument. The following options are available:
  - `'X'`
  - `'Y'`
  - `'Both'` (defaults listed with 'plots' parameter)
- **`cumulative`** (*`boolean, optional`*) – When set to `'cumulative=TRUE'` explained variances will be plotted cumulatively instead of per component.
- **`individual`** (*`boolean, optional`*) – When set to `'individual=TRUE'` explained variances will be plotted per variable instead of for all variables together.

- **validated** (*list, optional*) – When set to `'validated=[TRUE]'` validated values are plotted if applicable (scores => False, explainedVariance => True).
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type An explained variance plot based on the input Hoggorm model.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.explVar(myModel, which=['Both'], individual=False)
>>> hopl.explVar(myModel)
>>> hopl.explVar(myModel, cumulative=True)
```

hoggormplot.conv_explainedVariancePlot.**explainedVariance**(*model*, *which=[]*, *cumulative=True*, *individual=False*, *validated=[]*, *figsize=None*)

This function generates explained variances plots of hoggorm models.

Parameters

- **model** (*nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm.*) –
- **which** (*list, optional*) – This list may contain one string argument. The following options are available:
  - `'X'`
  - `'Y'`
  - `'Both'` (defaults listed with 'plots' parameter)
- **cumulative** (*boolean, optional*) – When set to `'cumulative=TRUE'` explained variances will be plotted cumulatively instead of per component.
- **individual** (*boolean, optional*) – When set to `'individual=TRUE'` explained variances will be plotted per variable instead of for all variables together.
- **validated** (*list, optional*) – When set to `'validated=[TRUE]'` validated values are plotted if applicable (scores => False, explainedVariance => True).
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type An explained variance plot based on the input Hoggorm model.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.explainedVariance(myModel, which=['Both'], individual=False)
```

(continues on next page)

(continued from previous page)

```
>>> hopl.explainedVariance(myModel)
>>> hopl.explainedVariance(myModel, cumulative=True)
```

3.2 Scores plot

hoggormplot.conv_scoresPlot.**scores**(*model, comp=[1, 2], which=[], objNames=[], newX=[], newY=[], newObjNames=[], figsize=None*)

This is a convenience function that generates scores plots of hoggorm models.

Parameters

- **model** (*nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm*) – The statistical results of the submitted model will be visualized.
- **comp** (*list, optional*) – The list contains components to be displayed. Defaults to [1,2].
- **which** (*list, optional*) – This list may contain one string argument. The following options are available:
  - `'X'`
  - `'Y'`
  - `'Both'` (defaults listed with 'plots' parameter)
- **objNames** (*list, optional*) – Object names may be provided in this list.
- **newX** (*list, optional*) – The list contains one array of new measurement data X.
- **newY** (*list, optional*) – The list contains one array of new measurements of Y.
- **newObjNames** (*list, optional*) – The list contains object names of new measurement data in X or Y.
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type A scores plot.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.scores(myModel, comp=[1, 3])
>>> hopl.scores(myModel)
```

3.3 Loadings plot

hoggormplot.conv_loadingsPlot.**loadings**(*model, comp=[1, 2], which=[], line=False, weights=False, XvarNames=[], YvarNames=[], figsize=None*)

This is a convenience function that generates loadings plots of hoggorm models.

Parameters

- **model** (*nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm*) – The statistical results of the submitted model will be visualized.
- **comp** (*list, optional*) – The list contains components to be displayed. Defaults to [1,2].
- **which** (*list, optional*) – This list may contain one string argument. The following options are available:
  - `'X'`
  - `'Y'`
  - `'Both'` (defaults listed with 'plots' parameter)
- **line** (*boolean, optional*) – When set to `'line=TRUE'` loadings (weights) will be plotted as lines / spectra instead of as points in a scatter plot.
- **weights** (*boolean, optional*) – When set to `'weights=TRUE'` loading weights will be plotted instead of loadings.
- **XvarNames** (*list, optional*) – Names of variables in array X may be provided in this list.
- **YvarNames** (*list, optional*) – Names of variables in Y may be provided in this list.
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type A loadings plot.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.loadings(myModel, comp=[2,4], which=['Both'])
>>> hopl.loadings(myModel)
>>> hopl.loadings(myModel, line=True, weights=False)
```

3.4 Correlation loadings plot

hoggormplot.conv_correlationLoadingsPlot.**correlationLoadings**(*model, comp=[1, 2], which=[], XvarNames=[], YvarNames=[], figsize=None*)

This is a convenience plot function which generates correlation loadings plots of hoggorm models.

Parameters

- **model** (*nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm*) – The statistical results of the submitted model will be visualized.
- **comp** (*list, optional*) – The list contains components to be displayed. Defaults to [1,2].
- **which** (*list, optional*) – This list may contain one string argument. The following options are available:
  - `'X'`
  - `'Y'`
  - `'Both'` (defaults listed with 'plots' parameter)
- **XvarNames** (*list, optional*) – Names of variables in array X may be provided in this list.
- **YvarNames** (*list, optional*) – Names of variables in Y may be provided in this list.
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type A correlation loadings plot based on the input hoggorm model.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.correlationLoadings(myModel, comp=[2,4], which=['Both'])
>>> hopl.correlationLoadings(myModel)
```

3.5 Biplot

hoggormplot.conv_biPlot.**biplot**(*model, comp=[1, 2], which=[], objNames=[], XvarNames=[], YvarNames=[], figsize=None*)

This is a convenience plot function which generates a bi-plot of hoggorm models.

Parameters

- **model** (*nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm.*) –

- **comp** (*list, optional*) – The list contains components to be displayed. Defaults to [1,2].
- **which** (*list, optional*) – This list may contain one string argument. The following options are available:
  - 'X'
  - 'Y'
  - 'Both' (defaults listed with 'plots' parameter)
- **objNames** (*list, optional*) – Object names may be provided in this list.
- **XvarNames** (*list, optional*) – Names of variables in array X may be provided in this list.
- **YvarNames** (*list, optional*) – Names of variables in Y may be provided in this list.
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type A bi-plot based on the input hoggorm model.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.biplot(myModel, comp=[2,4], which=['Both'])
>>> hopl.biplot(myModel)
```

3.6 Coefficients plot

Note: The two functions below, i.e. `.coeff` and `.coefficients` are two options of how to generate explained variance plots. They generate identical plots when provided the same input paramters or arguments.

hoggormplot.conv_coefficientsPlot.**coefficients**(*model, comp=[1], figsize=None*)

This is a convenience plot function which generates coefficients plots of hoggorm models.

Parameters

- **model** (*nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm*) –
- **comp** (*list, optional*) – The list contains components to be displayed. Defaults to [1].
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type A coefficients plot based on the input Hoggorm model.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.coefficients(myModel, comp=[1, 2, 3])
>>> hopl.coefficients(myModel)
>>> hopl.coefficients(myModel, comp=[2])
```

hoggormplot.conv_coefficientsPlot.**coeffs**(*model, comp=[1], figsize=None*)

This is a convenience plot function which generates coefficients plots of hoggorm models. Note that this convenience function is identical to `hoggorm.resPlotting.coefficients`.

Parameters

- **model** (*nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm*) –
- **comp** (*list, optional*) – The list contains components to be displayed. Defaults to [1].
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type A coefficients plot based on the input Hoggorm model.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.coeffs(myModel, comp=[1, 2, 3])
>>> hopl.coeffs(myModel)
>>> hopl.coeffs(myModel, comp=[2])
```

3.7 Loading weights plot

hoggormplot.conv_loadingWeightsPlot.**loadingWeights**(*model, comp=[1, 2], which=[], line=False, weights=True, XvarNames=[], YvarNames=[], figsize=None*)

This is a convenience function that generates loading weights plots of hoggorm models.

Parameters

- **model** (*nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm*) – The statistical results of the submitted model will be visualized.
- **comp** (*list, optional*) – The list contains components to be displayed. Defaults to [1,2].
- **which** (*list, optional*) – This list may contain one string argument. The following options are available:

  – `'X'`

  - 'Y'
  - 'Both' (defaults listed with 'plots' parameter)
- **line** (*boolean, optional*) – When set to 'line=TRUE' loadings (weights) will be plotted as lines / spectra instead of as points in a scatter plot.
- **weights** (*boolean, optional*) – When set to 'weights=TRUE' loading weights will be plotted instead of loadings.
- **XvarNames** (*list, optional*) – Names of variables in array X may be provided in this list.
- **YvarNames** (*list, optional*) – Names of variables in Y may be provided in this list.
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type A loadings weights plot.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.loadingWeights(myModel, comp=[2,4], which=['Both'])
>>> hopl.loadingWeights(myModel)
>>> hopl.loadingWeights(myModel, line=True, weights=True)
```

3.8 Predict plot

hoggormplot.conv_predictPlot.**predict**(*model, comp=[1, 2], objNames=[], newX=[], newY=[], newObjNames=[], figsize=None*)

This is a convenience function that generates plots of predicted vs. original values of hoggorm models.

Parameters

- **model** (*nipalsPCR/nipalsPLSR1/nipalsPLSR2 class object computed in Hoggorm*) – The statistical results of the submitted model will be visualized.
- **comp** (*list, optional*) – The list contains components to be displayed. Defaults to [1,2].
- **newX** (*list, optional*) – The list contains one array of new measurement data X.
- **newY** (*list, optional*) – The list contains one array of new measurements of Y.
- **newObjNames** (*list, optional*) – The list contains object names of new measurement data in X or Y.
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Returns

Return type A predicted vs. measured plot.

Examples

```
>>> import hoggorm as ho
>>> import hoggormplot as hopl
>>> myModel = ho.nipalsPLS2(arrX=my_X_data, arrY=my_Y_data, cvType=["loo"])
>>> hopl.predict(myModel)
>>> hopl.predict(myModel, comp=[3, 4])
```

CHAPTER 4

SMI plotting function

This is the SMI plotting function.

hoggormplot.SMI_plot.**plotSMI**(*smi*, *pc='max'*, *significance=True*, *X1name='X1'*, *X2name='X2'*, *B=10000*, *fontscale=1*, *figsize=None*)

Diamond plot for Similarity of matrices index (SMI)

Parameters

- **smi** (*SMI class object*) – the SMI results to visualize.
- **pc** (*list, optional*) – the number of components to display, default is 'max' = all components.
- **significance** (*boolean, optional*) – plotting of significance symbols, added by default.
- **X1name** (*list, optional*) – name of the first data set, default = 'X1'.
- **X2name** (*list, optional*) – name of the second data set, default = 'X2'.
- **B** (*int, optional*) – number of permutations to use with significance testing.
- **fontscale** (*double, optional*) – scaling parameter for significance symbols and component labels.
- **figsize** (*tuple, optional*) – Sets figure width and height in inches

Examples

```
>>> import numpy as np
>>> import hoggorm as ho
>>> import hoggormplot as hop
```

```
>>> X1 = ho.center(np.random.rand(100,300))
>>> U, s, V = np.linalg.svd(X1, 0)
>>> X2 = np.dot(np.dot(np.delete(U, 2, 1), np.diag(np.delete(s, 2))), np.delete(V,
↪ 2, 0))
```

(continues on next page)

(continued from previous page)

```
>>> smiOP = ho.SMI(X1, X2, ncomp1=10, ncomp2=10)
>>> print(smiOP.smi[:4,:4])
```

```
>>> hop.plotSMI(smiOP)
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

h

Index