

---

# **otpod Documentation**

***Release 0.0.1***

**Antoine Dumas**

April 11, 2016



<b>1</b>	<b>Contents:</b>	<b>1</b>
1.1	Documentation of the API . . . . .	1
1.1.1	Data analysis . . . . .	1
1.1.2	POD model . . . . .	6
1.2	Examples . . . . .	6
<b>2</b>	<b>Indices and tables</b>	<b>7</b>
	<b>Index</b>	<b>9</b>



## CONTENTS:

## 1.1 Documentation of the API

This is the user manual for the Python bindings to the otpod library.

### 1.1.1 Data analysis

---

*UnivariateLinearModelAnalysis* Linear regression analysis with residuals hypothesis tests.

---

#### UnivariateLinearModelAnalysis

**class UnivariateLinearModelAnalysis** (\*args)

Linear regression analysis with residuals hypothesis tests.

**Available constructors:**

UnivariateLinearModelAnalysis(*inputSample*, *outputSample*)

UnivariateLinearModelAnalysis(*inputSample*, *outputSample*, *noiseThres*, *saturationThres*, *resDistFact*, *boxCox*)

**Parameters** **inputSample** : 2-d sequence of float

Vector of the defect sizes, of dimension 1.

**outputSample** : 2-d sequence of float

Vector of the signals, of dimension 1.

**noiseThres** : float

Value for low censored data. Default is None.

**saturationThres** : float

Value for high censored data. Default is None

**resDistFact** : `openturns.DistributionFactory`

Distribution hypothesis followed by the residuals. Default is `openturns.NormalFactory`.

**boxCox** : bool or float

Enable or not the Box Cox transformation. If boxCox is a float, the Box Cox transformation is enabled with the given value. Default is False.

## Notes

This method automatically :

- computes the Box Cox parameter if *boxCox* is True,
- computes the transformed signals if *boxCox* is True or a float,
- builds the univariate linear regression model on the data,
- computes the linear regression parameters for censored data if needed,
- computes the residuals,
- runs all hypothesis tests.

## Methods

<code>drawBoxCoxLikelihood([name])</code>	Draw the loglikelihood versus the Box Cox parameter.
<code>drawLinearModel([model, name])</code>	Draw the linear regression prediction versus the true data.
<code>drawResiduals([model, name])</code>	Draw the residuals versus the defect values.
<code>drawResidualsDistribution([model, name])</code>	Draw the residuals histogram with the fitted distribution.
<code>drawResidualsQQplot([model, name])</code>	Draw the residuals QQ plot with the fitted distribution.
<code>getAndersonDarlingPValue()</code>	Accessor to the Anderson Darling test p-value.
<code>getBoxCoxParameter()</code>	Accessor to the Box Cox parameter.
<code>getBreuschPaganPValue()</code>	Accessor to the Breusch Pagan test p-value.
<code>getCramerVonMisesPValue()</code>	Accessor to the Cramer Von Mises test p-value.
<code>getDurbinWatsonPValue()</code>	Accessor to the Durbin Watson test p-value.
<code>getHarrisonMcCabePValue()</code>	Accessor to the Harrison McCabe test p-value.
<code>getIntercept()</code>	Accessor to the intercept of the linear regression model.
<code>getKolmogorovPValue()</code>	Accessor to the Kolmogorov test p-value.
<code>getR2()</code>	Accessor to the R2 value.
<code>getResiduals()</code>	Accessor to the residuals.
<code>getResidualsDistribution()</code>	Accessor to the residuals distribution.
<code>getSlope()</code>	Accessor to the slope of the linear regression model.
<code>getStandardError()</code>	Accessor to the standard error of the estimate.
<code>getZeroMeanPValue()</code>	Accessor to the Zero Mean test p-value.
<code>printResults()</code>	Print results of the linear analysis.
<code>saveResults(name)</code>	Save all analysis test results in a file.

### **drawBoxCoxLikelihood** (*name=None*)

Draw the loglikelihood versus the Box Cox parameter.

**Parameters** *name* : string

name of the figure to be saved with *transparent* option sets to True and *bbox\_inches='tight'*. It can be only the file name or the full path name. Default is None.

**Returns** *fig* : matplotlib.figure

Matplotlib figure object.

**ax** : matplotlib.axes

Matplotlib axes object.

## Notes

This method is available only when the parameter *boxCox* is set to True.

**drawLinearModel** (*model*='uncensored', *name*=None)

Draw the linear regression prediction versus the true data.

**Parameters** *model* : string

The linear regression model to be used, either *uncensored* or *censored* if censored threshold were given. Default is *uncensored*.

**name** : string

name of the figure to be saved with *transparent* option sets to True and *bbox\_inches*='tight'. It can be only the file name or the full path name. Default is None.

**Returns** *fig* : matplotlib.figure

Matplotlib figure object.

**ax** : matplotlib.axes

Matplotlib axes object.

**drawResiduals** (*model*='uncensored', *name*=None)

Draw the residuals versus the defect values.

**Parameters** *model* : string

The residuals to be used, either *uncensored* or *censored* if censored threshold were given. Default is *uncensored*.

**name** : string

name of the figure to be saved with *transparent* option sets to True and *bbox\_inches*='tight'. It can be only the file name or the full path name. Default is None.

**Returns** *fig* : matplotlib.figure

Matplotlib figure object.

**ax** : matplotlib.axes

Matplotlib axes object.

**drawResidualsDistribution** (*model*='uncensored', *name*=None)

Draw the residuals histogram with the fitted distribution.

**Parameters** *model* : string

The residuals to be used, either *uncensored* or *censored* if censored threshold were given. Default is *uncensored*.

**name** : string

name of the figure to be saved with *transparent* option sets to True and *bbox\_inches*='tight'. It can be only the file name or the full path name. Default is None.

**Returns** *fig* : matplotlib.figure

Matplotlib figure object.

**ax** : matplotlib.axes

Matplotlib axes object.

**drawResidualsQQplot** (*model='uncensored', name=None*)

Draw the residuals QQ plot with the fitted distribution.

**Parameters** **model** : string

The residuals to be used, either *uncensored* or *censored* if censored threshold were given. Default is *uncensored*.

**name** : string

name of the figure to be saved with *transparent* option sets to True and *bbox\_inches='tight'*. It can be only the file name or the full path name. Default is None.

**Returns** **fig** : `matplotlib.figure`

Matplotlib figure object.

**ax** : `matplotlib.axes`

Matplotlib axes object.

**getAndersonDarlingPValue** ()

Accessor to the Anderson Darling test p-value.

**Returns** **pValue** : `openturns.NumericalPoint`

Either the p-value for the uncensored case or for both cases.

**getBoxCoxParameter** ()

Accessor to the Box Cox parameter.

**Returns** **lambdaBoxCox** : float

The Box Cox parameter used to transform the data. If the transformation is not enabled None is returned.

**getBreuschPaganPValue** ()

Accessor to the Breusch Pagan test p-value.

**Returns** **pValue** : `openturns.NumericalPoint`

Either the p-value for the uncensored case or for both cases.

**getCramerVonMisesPValue** ()

Accessor to the Cramer Von Mises test p-value.

**Returns** **pValue** : `openturns.NumericalPoint`

Either the p-value for the uncensored case or for both cases.

**getDurbinWatsonPValue** ()

Accessor to the Durbin Watson test p-value.

**Returns** **pValue** : `openturns.NumericalPoint`

Either the p-value for the uncensored case or for both cases.

**getHarrisonMcCabePValue** ()

Accessor to the Harrison McCabe test p-value.

**Returns** **pValue** : `openturns.NumericalPoint`

Either the p-value for the uncensored case or for both cases.



**getIntercept ()**

Accessor to the intercept of the linear regression model.

**Returns** `intercept` : `openturns.NumericalPoint`

The intercept parameter for the uncensored and censored (if so) linear regression model.

**getKolmogorovPValue ()**

Accessor to the Kolmogorov test p-value.

**Returns** `pValue` : `openturns.NumericalPoint`

Either the p-value for the uncensored case or for both cases.

**getR2 ()**

Accessor to the R2 value.

**Returns** `R2` : `openturns.NumericalPoint`

Either the R2 for the uncensored case or for both cases.

**getResiduals ()**

Accessor to the residuals.

**Returns** `residuals` : `openturns.NumericalSample`

The residuals computed from the uncensored and censored linear regression model. The first column corresponds with the uncensored case.

**getResidualsDistribution ()**

Accessor to the residuals distribution.

**Returns** `distribution` : list of `openturns.Distribution`

The fitted distribution on the residuals, computed in the uncensored and censored (if so) case.

**getSlope ()**

Accessor to the slope of the linear regression model.

**Returns** `slope` : `openturns.NumericalPoint`

The slope parameter for the uncensored and censored (if so) linear regression model.

**getStandardError ()**

Accessor to the standard error of the estimate.

**Returns** `stderr` : `openturns.NumericalPoint`

The standard error of the estimate for the uncensored and censored (if so) linear regression model.

**getZeroMeanPValue ()**

Accessor to the Zero Mean test p-value.

**Returns** `pValue` : `openturns.NumericalPoint`

Either the p-value for the uncensored case or for both cases.

**printResults ()**

Print results of the linear analysis.

**saveResults (name)**

Save all analysis test results in a file.

**Parameters** `name` : string

Name of the file or full path name.

#### Notes

The file can be saved as a csv file. Separations are made with tabulations.

If *name* is the file name, then it is saved in the current working directory.

### 1.1.2 POD model

---

*UnivariateLinearRegressionPOD* doc

---

#### UnivariateLinearRegressionPOD

```
class UnivariateLinearRegressionPOD (*args)
    doc
```

#### Methods

---

*run()* Bla bla bla

---

```
run ()
    Bla bla bla
```

**Parameters** *sdfs* : float

*dfsdf*

**oko** : bool

*ture*

## 1.2 Examples

## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`



## D

drawBoxCoxLikelihood() (UnivariateLinearModelAnalysis method), 2  
drawLinearModel() (UnivariateLinearModelAnalysis method), 3  
drawResiduals() (UnivariateLinearModelAnalysis method), 3  
drawResidualsDistribution() (UnivariateLinearModelAnalysis method), 3  
drawResidualsQQplot() (UnivariateLinearModelAnalysis method), 4

## G

getAndersonDarlingPValue() (UnivariateLinearModelAnalysis method), 4  
getBoxCoxParameter() (UnivariateLinearModelAnalysis method), 4  
getBreuschPaganPValue() (UnivariateLinearModelAnalysis method), 4  
getCramerVonMisesPValue() (UnivariateLinearModelAnalysis method), 4  
getDurbinWatsonPValue() (UnivariateLinearModelAnalysis method), 4  
getHarrisonMcCabePValue() (UnivariateLinearModelAnalysis method), 4  
getIntercept() (UnivariateLinearModelAnalysis method), 4  
getKolmogorovPValue() (UnivariateLinearModelAnalysis method), 5  
getR2() (UnivariateLinearModelAnalysis method), 5  
getResiduals() (UnivariateLinearModelAnalysis method), 5  
getResidualsDistribution() (UnivariateLinearModelAnalysis method), 5  
getSlope() (UnivariateLinearModelAnalysis method), 5  
getStandardError() (UnivariateLinearModelAnalysis method), 5  
getZeroMeanPValue() (UnivariateLinearModelAnalysis method), 5

## P

printResults() (UnivariateLinearModelAnalysis method), 5

## R

run() (UnivariateLinearRegressionPOD method), 6

## S

saveResults() (UnivariateLinearModelAnalysis method), 5

## U

UnivariateLinearModelAnalysis (class in otpod), 1  
UnivariateLinearRegressionPOD (class in otpod), 6