NDK_BOXCOX

Last Modified on 04/21/2016 12:27 pm CDT

- C/C++
- .Net

```
int __stdcall NDK_BOXCOX(double * X,
                         size_t   N,
                         double * lambda,
                         double * alpha,
                         int      retTYpe,
                         double * retVal
                        )
```

Computes the Box-Cox transformation, including its inverse.

Returns

status code of the operation

Return values

NDK_SUCCESS Operation successful

NDK_FAILED Operation unsuccessful. See Macros for full list.

Parameters

[in,out] **X** is the univariate time series data (a one dimensional array).

[in] **N** is the number of observations in X.

[in] **lambda** is the input power parameter of the transformation, on a scale from 1 to 0. If omitted, a default value of 0 is assumed.

[in] **alpha** is the input shift parameter for X. If omitted, the default value is 0.

[in] **retTYpe** is a number that determines the type of return value: 1 (or missing)=Box-Cox, 2=inverse Box-Cox, 3= LLF of Box-Cox.

Value	Return Type
1 or omitted	Box-Cox Transform
2	Inverse of Box-Cox transform
3	LLF of Box-Cox transform

[out] **retVal** is the calculated log-likelihood value of the transform (retType=3).

Remarks

1. **Box-Cox** transform is perceived as a useful data (pre)processing technique used to stabilize variance and make the data more normally distributed.
2. The **Box-Cox** transformation is defined as follows: \[T\left (x_{t}; \lambda, \alpha \right) = \begin{cases} \dfrac{\left (x_{t} + \alpha \right)^{\lambda}-1}{\lambda} & \text{ if } \lambda \neq 0 \\ \log \left (x_t + \alpha \right) & \text{ if } \lambda= 0 \end{cases} \] Where:
   - \(x_{t}\) is the value of the input time series at time \(t\)

- \(\lambda\) is the input scalar value of the Box-Cox transformation
- \(\alpha\) is the shift parameter
- \(\left(x_t +\alpha \right) \gt 0\) for all t values.

3. The BOXCOX function accepts a single value or an array of values for X.
4. The shift parameter must be large enough to make all the values of X positive.

Requirements

Header	SFSDK.H
Library	SFSDK.LIB
DLL	SFSDK.DLL

Examples

```
int NDK_BOXCOX(double[]     pData,
               UIntPtr      nSize,
               out double   lambda,
               out double   fAlpha,
               int          argRetType,
               out double   retVal
              )
```

Namespace:	NumXLAPI
Class:	SFSDK
Scope:	Public
Lifetime:	Static

Computes the Box-Cox transformation, including its inverse.

Returns

status code of the operation

Return values

NDK_SUCCESS	Operation successful
NDK_FAILED	Operation unsuccessful. See Macros for full list.

Parameters

[in,out]	**pData**	is the univariate time series data (a one dimensional array).
[in]	**nSize**	is the number of observations in pData.
[in]	**lambda**	is the input power parameter of the transformation, on a scale from 1 to

0. If omitted, a default value of 0 is assumed.

[in] **fAlpha** is the input shift parameter for pData. If omitted, the default value is 0.

[in] **argRetType** is a number that determines the type of return value: 1 (or missing)=Box-Cox, 2=inverse Box-Cox, 3= LLF of Box-Cox.

Value	Return Type
1 or omitted	Box-Cox Transform
2	Inverse of Box-Cox transform
3	LLF of Box-Cox transform

[out] **retVal** is the calculated log-likelihood value of the transform (retType=3).

Remarks

1. **Box-Cox** transform is perceived as a useful data (pre)processing technique used to stabilize variance and make the data more normally distributed.
2. The **Box-Cox** transformation is defined as follows: \[T\left (x_{t}; \lambda, \alpha \right) = \begin{cases} \dfrac{\left (x_{t} + \alpha \right)^{\lambda}-1}{\lambda} & \text{ if } \lambda \neq 0 \\ \log \left (x_t + \alpha \right) & \text{ if } \lambda= 0 \end{cases} \] Where:
   - \(x_{t}\) is the value of the input time series at time \(t\)
   - \(\lambda\) is the input scalar value of the Box-Cox transformation
   - \(\alpha\) is the shift parameter
   - \(\left(x_t +\alpha \right) \gt 0\) for all t values.
3. The BOXCOX function accepts a single value or an array of values for X.
4. The shift parameter must be large enough to make all the values of X positive.

Exceptions

Exception Type	Condition
None	N/A

Requirements

Namespace	NumXLAPI
Class	SFSDK
Scope	Public
Lifetime	Static
Package	NumXLAPI.DLL

Examples

References

Hamilton, J .D.; Time Series Analysis , Princeton University Press (1994), ISBN 0-691-04289-6

Tsay, Ruey S.; Analysis of Financial Time Series John Wiley & SONS. (2005), ISBN 0-471-690740

See Also

[template("related")]