NDK_HodrickPrescotFilter

Last Modified on 07/07/2016 12:22 pm CDT

- C/C++
- .Net

```
int __stdcall NDK_HodrickPrescotFilter(double * X,
                                       size_t   N,
                                       BOOL     bAscending,
                                       double   lambda
                                      )
```

computes cyclical component of given time series using the Hodrick's Prescott filter.

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]	**bAscending**	is the time order in the data series (i.e. the first data point's corresponding date (earliest date=1 (default), latest date=0)).
[in]	**lambda**	is the multiplier used to penalize the variation in the trend component. If missing, a default is used based on data frequency.

Remarks

1. The time series is homogeneous or equally spaced.
2. The time series may include missing values (NaN) at either end.
3. The Hodrick-Prescott filter is used to obtain a smoothed-curve representation of a time series, one that is more sensitive to long-term than to short-term fluctuations
4. In sum, The Hodrick-Prescott filter is a mathematical tool used to separate the cyclical component of a time series from raw data: \[y_t = c_t + \tau_t \] Where:
   - \() t=1,2,\cdots , T\).
   - \(y_t\) is the input time series.
   - \(c_t\) is the cyclical component.
   - \(\tau_t\) is the trend component.
5. Hodrick and Prescott (1997) suggest the following criterion to reveal the unobserved components, \(\tau_t\) and \(c_t\), conditional on a choice of "smoothing parameter" \(\lambda\): \[\min_{\tau}\left(\sum_{t = 1}^T {(y_t - \tau _t)^2 } + \lambda \sum_{t = 2}^{T - 1} {[(\tau _{t+1} -

\tau _t) - (\tau _t - \tau _{t - 1})]^2 }\right) \]

6. An expert judgment for the choice of \(lambda\) is necessary. In general, the close is \(lambda\) to zero, the closer is filtered trend to the original series. Likewise, if \(lambda\) approaches infinity, the filtered trend becomes a straight line.
7. If lambda is zero or negative, NxHP return #VALUE!
8. In the event that lambda and data frequency are missing, lambda is set to a default value of 1600.
9. The input data must be properly seasonal adjusted prior to HP filtering.
10. HP Analysis is purely historical and static (closed domain). The filter causes misleading predictions when used dynamically since the algorithm changes (during iteration for minimization) the past state (unlike a moving average) of the time series to adjust for the current state regardless of the size of \lambda used.
11. In comparison to other techniques, such as the production function approach or the Kalman filter, the HP filter forms a fast and easy to use alternative.

Requirements

Header	SFSDK.H
Library	SFSDK.LIB
DLL	SFSDK.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")]