Package 'intradayModel'

July 22, 2025

Title Modeling and Forecasting Financial Intraday Signals

Version 0.0.1

Date 2023-05-20

Description Models, analyzes, and forecasts financial intraday signals. This package currently supports a univariate state-space model for intraday trading volume provided by Chen (2016) <doi:10.2139/ssrn.3101695>.

Maintainer Daniel P. Palomar <daniel.p.palomar@gmail.com>

URL https://github.com/convexfi/intradayModel,
https://www.danielppalomar.com,
https://dx.doi.org/10.2139/ssrn.3101695

BugReports https://github.com/convexfi/intradayModel/issues

License Apache License (== 2.0)

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 2.10)

Imports ggplot2, magrittr, patchwork, reshape2, scales, xts, zoo, utils

Suggests knitr, rmarkdown, R.rsp, testthat (>= 3.0.0), cleanrmd, devtools

VignetteBuilder knitr, rmarkdown, R.rsp

LazyData true

Config/testthat/edition 3

NeedsCompilation no

Author Shengjie Xiu [aut],
Yifan Yu [aut],
Daniel P. Palomar [cre, aut, cph]

Repository CRAN

Date/Publication 2023-05-22 08:30:02 UTC

Contents

`intradayModel-package` *intradayModel: Modeling and Forecasting Financial Intraday Signals*

Description

This package uses state-of-the-art state-space models to facilitate the modeling, analyzing and forecasting of financial intraday signals. It currently offers a univariate model for intraday trading volume, with new features on intraday volatility and multivariate models in development.

Functions

`fit_volume`, `decompose_volume`, `forecast_volume`, `generate_plots`

Data

`volume_aapl`, `volume_fdx`

Help

For a quick help see the README file: GitHub-README.

Author(s)

Shengjie Xiu, Yifan Yu and Daniel P. Palomar decompose_volume *Decompose Intraday Volume into Several Components*

Description

This function decomposes the intraday volume into daily, seasonal, and intraday dynamic components according to (Chen et al., 2016). If `purpose` = “`analysis`” (aka Kalman smoothing), the optimal components are conditioned on both the past and future observations. Its mathematical expression is $\hat{x}_\tau = E[x_\tau|\{y_j\}_{j=1}^M]$, where M is the total number of bins in the dataset.

If `purpose` = “`forecast`” (aka Kalman forecasting), the optimal components are conditioned on only the past observations. Its mathematical expression is $\hat{x}_{\tau+1} = E[x_{\tau+1}|\{y_j\}_{j=1}^\tau]$.

Three measures are used to evaluate the model performance:

- Mean absolute error (MAE): $\frac{1}{M}\sum_{\tau=1}^M |\hat{y}_\tau - y_\tau|$;
- Mean absolute percent error (MAPE): $\frac{1}{M}\sum_{\tau=1}^M \frac{|\hat{y}_\tau - y_\tau|}{y_\tau}$;
- Root mean square error (RMSE): $\sqrt{\sum_{\tau=1}^M \frac{(\hat{y}_\tau - y_\tau)^2}{M}}$.

Usage

```
decompose_volume(purpose, model, data, burn_in_days = 0)
```

Arguments

`purpose`	String `"analysis"/"forecast"`. Indicates the purpose of using the provided model.
`model`	A model object of class "`volume_model`" from `fit_volume()`.
`data`	An n_bin * n_day matrix or an `xts` object storing intraday volume.
`burn_in_days`	Number of initial days in the burn-in period for forecast. Samples from the first `burn_in_days` are used to warm up the model and then are discarded.

Value

A list containing the following elements:

- `original_signal`: A vector of original intraday volume;
- `smooth_signal` / `forecast_signal`: A vector of smooth/forecast intraday volume;
- `smooth_components` /`forecast_components`: A list of smooth/forecast components: daily, seasonal, intraday dynamic, and residual components.
- `error`: A list of three error measures: mae, mape, and rmse.

Author(s)

Shengjie Xiu, Yifan Yu and Daniel P. Palomar

References

Chen, R., Feng, Y., and Palomar, D. (2016). Forecasting intraday trading volume: A Kalman filter approach. Available at SSRN 3101695.

Examples

```
library(intradayModel)
data(volume_aapl)
volume_aapl_training <- volume_aapl[, 1:20]
volume_aapl_testing <- volume_aapl[, 21:50]
model_fit <- fit_volume(volume_aapl_training, fixed_pars = list(a_mu = 0.5, var_mu = 0.05),
                        init_pars = list(a_eta = 0.5))

# analyze training volume
analysis_result <- decompose_volume(purpose = "analysis", model_fit, volume_aapl_training)

# forecast testing volume
forecast_result <- decompose_volume(purpose = "forecast", model_fit, volume_aapl_testing)

# forecast testing volume with burn-in
forecast_result <- decompose_volume(purpose = "forecast", model_fit, volume_aapl[, 1:50],
                             burn_in_days = 20)
```

`fit_volume` *Fit a Univariate State-Space Model on Intraday Trading Volume*

Description

The main function for defining and fitting a univaraite state-space model on intraday trading volume. The model is proposed in (Chen et al., 2016) as

$$\mathbf{x}_{\tau+1} = \mathbf{A}_\tau \mathbf{x}_\tau + \mathbf{w}_\tau,$$

$$y_\tau = \mathbf{C}\mathbf{x}_\tau + \phi_\tau + v_\tau,$$

where

- $\mathbf{x}_\tau = [\eta_\tau, \mu_\tau]^\top$ is the hidden state vector containing the log daily component and the log intraday dynamic component;
- $\mathbf{A}_\tau = \begin{bmatrix} a_\tau^\eta & 0 \\ 0 & a^\mu \end{bmatrix}$ is the state transition matrix with $a_\tau^\eta = \begin{cases} a^\eta & t = kI, k = 1, 2, \dots \\ 0 & \text{otherwise;} \end{cases}$
- $\mathbf{C} = [1, 1]$ is the observation matrix;
- ϕ_τ is the corresponding element from $\boldsymbol{\phi} = [\phi_1, \dots, \phi_I]^\top$, which is the log seasonal component;
- $\mathbf{w}_\tau = [\epsilon_\tau^\eta, \epsilon_\tau^\mu]^\top \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_\tau)$ represents the i.i.d. Gaussian noise in the state transition, with a time-varying covariance matrix $\mathbf{Q}_\tau = \begin{bmatrix} (\sigma_\tau^\eta)^2 & 0 \\ 0 & (\sigma^\mu)^2 \end{bmatrix}$ and $\sigma_\tau^\eta = \begin{cases} \sigma^\eta & t = kI, k = 1, 2, \dots \\ 0 & \text{otherwise;} \end{cases}$

- $v_\tau \sim \mathcal{N}(0, r)$ is the i.i.d. Gaussian noise in the observation;
- $\mathbf{x}_1$ is the initial state at $\tau = 1$, and it follows $\mathcal{N}(\mathbf{x}_0, \mathbf{V}_0)$.

In the model, $\boldsymbol{\Theta} = \{a^\eta, a^\mu, \sigma^\eta, \sigma^\mu, r, \boldsymbol{\phi}, \mathbf{x}_0, \mathbf{V}_0\}$ are treated as parameters. The model is fitted by expectation-maximization (EM) algorithms. The implementation follows (Chen et al., 2016), and the accelerated scheme is provided in (Varadhan and Roland, 2008). The algorithm terminates when `maxit` is reached or the condition $\|\Delta\boldsymbol{\Theta}_i\| \leq$ abstol is satisfied.

Usage

```
fit_volume(
  data,
  fixed_pars = NULL,
  init_pars = NULL,
  verbose = 0,
  control = NULL
)
```

Arguments

`data` An n_bin * n_day matrix or an `xts` object storing intraday trading volume.

`fixed_pars` A list of parameters' fixed values. The allowed parameters are listed below,

- `"a_eta"`: a^η of size 1 ;
- `"a_mu"`: a^μ of size 1 ;
- `"var_eta"`: σ^η of size 1 ;
- `"var_mu"`: σ^μ of size 1 ;
- `"r"`: r of size 1 ;
- `"phi"`: $\boldsymbol{\phi} = [\phi_1, \ldots, \phi_I]^\top$ of size I ;
- `"x0"`: $\mathbf{x}_0$ of size 2 ;
- `"V0"`: $\mathbf{V}_0$ of size 2 * 2 .

`init_pars` A list of unfitted parameters' initial values. The parameters are the same as `fixed_pars`. If the user does not assign initial values for the unfitted parameters, default ones will be used.

`verbose` An integer specifying the print level of information during the algorithm (default `1`). Possible numbers:

- `"0"`: no output;
- `"1"`: show the iteration number and $\|\Delta\boldsymbol{\Theta}_i\|$;
- `"2"`: 1 + show the obtained parameters.

`control` A list of control values of EM algorithm:

- `acceleration`: TRUE/FALSE indicating whether to use the accelerated EM algorithm (default TRUE);
- `maxit`: Maximum number of iterations (default `3000`);
- `abstol`: Absolute tolerance for parameters' change $\|\Delta\boldsymbol{\Theta}_i\|$ as the stopping criteria (default `1e-4`)
- `log_switch`: TRUE/FALSE indicating whether to record the history of convergence progress (defalut TRUE).

Value

A list of class "volume_model" with the following elements (if the algorithm converges):

- par: A list of parameters' fitted values.
- init: A list of valid initial values from users.
- par_log: A list of intermediate parameters' values if log_switch = TRUE.
- converged: A list of logical values indicating whether each parameter is fitted.

Author(s)

Shengjie Xiu, Yifan Yu and Daniel P. Palomar

References

Chen, R., Feng, Y., and Palomar, D. (2016). Forecasting intraday trading volume: A Kalman filter approach. Available at SSRN 3101695.

Varadhan, R., and Roland, C. (2008). Simple and globally convergent methods for accelerating the convergence of any EM algorithm. Scandinavian Journal of Statistics, 35(2), 335–353.

Examples

```
library(intradayModel)
data(volume_aapl)
volume_aapl_training <- volume_aapl[, 1:20]

# fit model with no prior knowledge
model_fit <- fit_volume(volume_aapl_training)

# fit model with fixed_pars and init_pars
model_fit <- fit_volume(volume_aapl_training, fixed_pars = list(a_mu = 0.5, var_mu = 0.05),
                        init_pars = list(a_eta = 0.5))

# fit model with other control options
model_fit <- fit_volume(volume_aapl_training, verbose = 2,
  control = list(acceleration = FALSE, maxit = 1000, abstol = 1e-4, log_switch = FALSE))
```

---

forecast_volume *Forecast One-bin-ahead Intraday Volume*

Description

This function forecasts one-bin-ahead intraday volume. Its mathematical expression is $\hat{y}_{\tau+1} = E[y_{\tau+1}|\{y_j\}_{j=1}^{\tau}]$. It is a wrapper of decompose_volume() with purpose = "forecast".

Usage

```
forecast_volume(model, data, burn_in_days = 0)
```

Arguments

`model`	A model object of class "`volume_model`" from `fit_volume()`.
`data`	An n_bin * n_day matrix or an `xts` object storing intraday volume.
`burn_in_days`	Number of initial days in the burn-in period. Samples from the first `burn_in_days` are used to warm up the model and then are discarded.

Value

A list containing the following elements:

- `original_signal`: A vector of original intraday volume;
- `forecast_signal`: A vector of forecast intraday volume;
- `forecast_components`: A list of the three forecast components: daily, seasonal, intraday dynamic, and residual components.
- `error`: A list of three error measures: mae, mape, and rmse.

Author(s)

Shengjie Xiu, Yifan Yu and Daniel P. Palomar

References

Chen, R., Feng, Y., and Palomar, D. (2016). Forecasting intraday trading volume: A Kalman filter approach. Available at SSRN 3101695.

Examples

```
library(intradayModel)
data(volume_aapl)
volume_aapl_training <- volume_aapl[, 1:20]
volume_aapl_testing <- volume_aapl[, 21:50]
model_fit <- fit_volume(volume_aapl_training, fixed_pars = list(a_mu = 0.5, var_mu = 0.05),
                        init_pars = list(a_eta = 0.5))

# forecast testing volume
forecast_result <- forecast_volume(model_fit, volume_aapl_testing)

# forecast testing volume with burn-in
forecast_result <- forecast_volume(model_fit, volume_aapl[, 1:50], burn_in_days = 20)
```

generate_plots *Plot Analysis and Forecast Result*

Description

Generate plots for the analysis and forecast results.

Usage

```
generate_plots(analysis_forecast_result)
```

Arguments

`analysis_forecast_result`
: Analysis/forecast result from `decompose_volume()` or `forecast_volume()`.

Value

A list of `patchwork` objects:

- `components`: Plot of components of intraday volume;
- `log_components`: Plot of components of intraday volume in their log10 scale;
- `original_and_smooth` / `original_and_forecast`: Plot of the original and the smooth/forecast intraday volume.

Author(s)

Shengjie Xiu, Yifan Yu and Daniel P. Palomar

Examples

```
library(intradayModel)
data(volume_aapl)
volume_aapl_training <- volume_aapl[, 1:20]
volume_aapl_testing <- volume_aapl[, 21:50]

# obtain analysis and forecast result
model_fit <- fit_volume(volume_aapl_training, fixed_pars = list(a_mu = 0.5, var_mu = 0.05),
                        init_pars = list(a_eta = 0.5))
analysis_result <- decompose_volume(purpose = "analysis", model_fit, volume_aapl_training)
forecast_result <- forecast_volume(model_fit, volume_aapl_testing)

# plot the analysis and forecast result
generate_plots(analysis_result)
generate_plots(forecast_result)
```

volume_aapl — *15-min Intraday Volume of AAPL*

Description

A 26 * 124 matrix including 15-min trading volume of AAPL from 2019-01-02 to 2019-06-28.

Usage

```
data(volume_aapl)
```

Format

A 26 * 124 matrix.

Source

barchart

volume_fdx — *15-min Intraday Volume of FDX*

Description

An `xts` object including 15-min trading volume of FDX from 2019-07-01 to 2019-12-31.

Usage

```
data(volume_fdx)
```

Format

An `xts` object.

Source

barchart

Index