argamma Documentation

Release 0.1

Stanislav Khrapov

July 06, 2015

Contents

The code is an implementation of ARG model given in *[R1]*. Its major features include:

- simulation of stochastic volatility and returns
- estimation using both MLE and GMM
- option pricing

CHAPTER 1

References

CHAPTER 2

Class documentation

class argamma.arg.**ARG**(*param=None*)

Class for ARG model.

$$E\left[\exp\left\{-uY_t\right\}|\, Y_{t-1}\right] = \exp\left\{-a(u)Y_{t-1} - b(u)\right\}$$

Attributes

vol	Volatility series
ret	Asset return series
param	Parameters of the model
maturity	Maturity of the option or simply time horizon. Fraction of a year, i.e. 30/365
riskfree	Risk-free rate of return per day

Methods

afun(uarg, param)	Function a().
bfun(uarg, param)	Function b().
cfun(uarg, param)	Function c().
plot_abc(uarg, param)	Plot a() and b() functions on the same plot.
vsim([nsim, nobs, param])	Simulate ARG(1) process for volatility.
vsim2([nsim, nobs, param])	Simulate ARG(1) process for volatility.
rsim([param])	Simulate returns given ARG(1) process for volatility.
load_data([vol, ret])	Load data into the model object.
estimate_mle([param_start, model, bounds])	Estimate model parameters via Maximum Likelihood.
estimate_gmm([param_start, model])	Estimate model parameters using GMM.
cos_restriction()	Restrictions used in COS method of option pricing.
charfun(varg)	Risk-neutral conditional characteristic function.
option_premium([vol, moneyness, maturity, ...])	Model implied option premium via COS method.

afun(*uarg*, *param*)

Function a().

$$a\left(u\right) = \frac{\rho u}{1 + cu}$$

Parameters **uarg** : array

Grid

param : ARGparams instance

Model parameters

Returns array

Same dimension as uarg

bfun(*uarg*, *param*)
Function b().

$$b(u) = \delta \log(1 + cu)$$

Parameters **uarg** : array

Grid

param : ARGparams instance

Model parameters

Returns array

Same dimension as uarg

cfun(*uarg*, *param*)
Function c().

$$c(u) = \delta \log\left\{1 + \frac{cu}{1-\rho}\right\}$$

Parameters **uarg** : array

Grid

param : ARGparams instance

Model parameters

Returns array

Same dimension as uarg

charfun(*varg*)
Risk-neutral conditional characteristic function.

Parameters **varg** : array

Grid for evaluation of CF. Real values only.

Returns array

Same dimension as varg

Notes

This method is used by COS method of option pricing

estimate_gmm(*param_start=None*, *model='vol'*, ***kwargs*)
Estimate model parameters using GMM.

Parameters **param_start** : ARGparams instance

Starting value for optimization

model : str

Type of the model to estimate. Must be in:

- 'vol'
- 'ret'
- 'joint'

uarg : array

Grid to evaluate a and b functions

zlag : int, optional

Number of lags in the instrument. Default is 1

Returns **param_final** : ARGparams instance

Estimated model parameters

mygmm.Results instance

GMM estimation results

estimate_mle(*param_start=None*, *model=None*, *bounds=None*)

Estimate model parameters via Maximum Likelihood.

Parameters **param_start** : ARGparams instance, optional

Starting value for optimization

model : str

Type of model to estimate. Must be in:

- 'vol'
- 'ret'
- 'joint'

bounds : list of tuples

Bounds on parameters, i.e. [(min, max)]

Returns **param_final** : ARGparams instance

Estimated parameters

results : OptimizeResult instance

Optimization output

load_data(*vol=None*, *ret=None*)

Load data into the model object.

Parameters **vol** : (nobs,) array

Volatility time series

ret : (nobs,) array

Return time series

option_premium(*vol=None*, *moneyness=None*, *maturity=None*, *riskfree=None*, *call=None*, *data=None*, *npoints=1024*)

Model implied option premium via COS method.

Parameters **vol** : array_like

Current variance per day

moneyness : array_like

Log-forward moneyness, np.log(strike/price) - riskfree * maturity

maturity : float, optional

Maturity of the option or simply time horizon. Fraction of a year, i.e. 30/365

riskfree : float, optional

Risk-free rate of return per day

call : bool array_like

Call/Put flag

data : pandas DataFrame, record array, or dictionary of arrays

Structured data. Mandatory labels: vol, moneyness, maturity, riskfree, call

npoints : int

Number of points on the grid. The more the better, but slower.

Returns array_like

Model implied option premium via COS method

rsim(*param=None*)

Simulate returns given ARG(1) process for volatility.

Parameters **param** : ARGparams instance

Model parameters

Returns (nobs, nsim) array

Simulated data

vsim(*nsim=1*, *nobs=100*, *param=None*)

Simulate ARG(1) process for volatility.

$$Z_t|Y_{t-1} \sim \mathcal{P}(\beta Y_{t-1})$$
$$Y_t|Z_t \sim \gamma(\delta + Z_t, c)$$

Parameters **nsim** : int

Number of series to simulate

nobs : int

Number of observations to simulate

param : ARGparams instance

Model parameters

Returns (nobs, nsim) array

Simulated data

vsim2(*nsim=1*, *nobs=100*, *param=None*)

Simulate ARG(1) process for volatility.

Uses non-central Chi-square distribution to simulate in one step.

Parameters **nsim** : int

Number of series to simulate

nobs : int

Number of observations to simulate

param : ARGparams instance

Model parameters

Returns (nobs, nsim) array

Simulated data

Bibliography

[R1] Stanislav Khrapov and Eric Renault (2014) “Affine Option Pricing Model in Discrete Time”, working paper, New Economic School. <http://goo.gl/yRVsZp>

[R2] Christian Gourieroux and Joann Jasiak (2006) “Autoregressive Gamma Processes”, 2006, *Journal of Forecasting*, 25(2), 129–152. doi:10.1002/for.978

[R3] Serge Darolles, Christian Gourieroux, and Joann Jasiak (2006) “Structural Laplace Transform and Compound Autoregressive Models” *Journal of Time Series Analysis*, 27(4), 477–503. doi:10.1111/j.1467-9892.2006.00479.x

a