

---

# **mygmm Documentation**

***Release 0.1***

**Stanislav Khrapov**

August 13, 2015







**class** `mygmm.gmm.GMM(momcond)`  
GMM estimation class.

### Attributes

<code>momcond</code>	Moment function
----------------------	-----------------

### Methods

---

`gmme`  
`st`

---

**\_\_init\_\_** (*momcond*)  
Initialize the class.

**Parameters** `momcond` : function

Moment function. Should return:

- **array (nobs x nmoms)** moment function values
- **(optionally) array (nmoms x nparams)** derivative of moment function average across observations.

**gmme** (*theta\_start*, *bounds=None*, *constraints=()*, *iter=2*, *method='BFGS'*, *kernel='Bartlett'*, *band=None*, *names=None*, *\*\*kwargs*)  
Multiple step GMM estimation procedure.

**Parameters** `theta_start` : array

Initial parameters

**bounds** : list of tuples

Bounds on parameters

**constraints** : dict or sequence of dict

Equality and inequality constraints. See `scipy.optimize.minimize`

**iter** : int

Number of GMM steps

**method** : str

Optimization method

**kernel** : str

Type of kernel for HAC. Currently implemented: SU, Bartlett, Parzen, Quadratic

**band** : int

Truncation parameter for HAC

**names** : list of str

Parameter names

**Returns** instance of Results

Estimation results

**class** `mygmm.results.Results` (*opt\_out=None*, *var\_theta=None*, *nmoms=None*, *names=None*)  
Class to hold estimation results.

#### Attributes

theta	Parameter estimate
degf	Degrees of freedom
jstat	J-statistic
stde	Standard errors
tstat	t-statistics
jpval	p-value of the J test
opt	Optimization output

## Symbols

`__init__()` (mygmm.gmm.GMM method), 1

## G

GMM (class in mygmm.gmm), 1

`gmtest()` (mygmm.gmm.GMM method), 1

## R

Results (class in mygmm.results), 1