RISE Documentation

Release 1.0.0

Junior Maih

October 16, 2014

CONTENTS

CHAPTER

ONE

INTRODUCTION

1.1 RISE at a Glance

1.1.1 What is RISE?

RISE is the acronym for **R**ationality **I**n **S**witching **E**nvironments.

It is an object-oriented Matlab toolbox primarily designed for solving and estimating nonlinear dynamic stochastic general equilibirium (**DSGE**) or more generally Rational Expectations(**RE**) models with **switching parameters**.

Leading references in the field include various papers by Roger Farmer, Dan Waggoner and Tao Zha and Eric Leeper among others.

RISE uses perturbation to approximate the nonlinear Markov Switching Rational Expectations (**MSRE**) model and solves it using efficient algorithms.

RISE also implements special cases of the general Switching MSRE model. This includes

- **VAR**s with and without switching parameters
- **SVAR**s with and without switching paramters
- **Time-varying parameter VAR**s
- etc.

1.1.2 Motivation for RISE development

- The world is not constant, it is switching

1.2 Capabilities of RISE

1.2.1 DSGE modeling

- constant parameters
- **switching parameters**
  - exogenous switching
  - endogenous switching
- **optimal policy (with and without switching)**

    - discretion
    - commitment
    - loose commitment
    - optimized simple rules
- Deterministic simulation
- Stochastic simulation
- higher-order perturbations

1.2.2 VAR modeling

- **constant parameters**
    - zero restrictions
    - sign restrictions
    - restrictions on lag structure
    - linear restrictions
- **switching parameters**
    - linear restrictions

1.2.3 SVAR modeling

- constant parameters
- **switching parameters**
    - linear restrictions

1.2.4 Time-Varying parameter VAR modeling

Under implementation

1.2.5 Smooth transition VAR modeling

Not yet implemented

1.2.6 Forecasting and Conditional Forecasting

1.2.7 Global sensitivity analysis

- Monte carlo filtering
- High dimensional model representation

1.2.8 Maximum Likelihood and Bayesian Estimation

- linear restrictions
- nonlinear restrictions

1.2.9 Time series

1.2.10 Reporting

1.3 How RISE works

1.3.1 Object orientation

1.3.2 Basic principles

- you can pass different options at any time

1.4 Background and mathematical formulations

1.5 Using this documentation

1.5.1 how to find help

1.5.2 Road map

1.6 Citing RISE in your research

CHAPTER

TWO

GETTING STARTED WITH RISE

2.1 Installation guide

2.1.1 Software requirements

I order to use RISE, the following software will need to be installed:

- Matlab version ? or higher
- MikTex (Windows users) MacTex (mac users)

2.1.2 How to obtain RISE

There are (at least) two ways to acquire RISE:

The zip file option

1. Go online to https://github.com/jmaih/RISE_toolbox
2. download the zip file and unzip it in some directory on your computer.

This option is not recommended but is convenient for people who are not allowed to install new software on their machines/laptop.

Github for the bleeding-edge installation (highly recommended)

1. Go to http://windows.github.com if you are a windows user or to http://mac.github.com if you are a mac user
2. Create an account online through the website and download the Github program
3. Sign in both online and on the github on your machine. It is obvious online, but on your machine, just go to Github>Preference>Account
4. Go online to https://github.com/jmaih/RISE_toolbox
5. Look for an icon with title 'Clone in Desktop' (or possibly clone in mac). There are options to locate where the repository will reside

The reason why this option is recommended is that you don't need to re-download the whole toolbox every time a marginal update is made. With one click and within seconds you can have the version of the toolbox on your computer updated.

The git option (never tested!!!)

The following has never been tested and so the syntax might be wrong:

```
git clone https://github.com/jmaih/RISE_toolbox.git
```

Testing your installation

More on this later...

2.1.3 Loading and starting RISE

1. Locate the RISE_toolbox directory and add its path to matlab in the command window as

   ```
   addpath('C:/Users/JMaih/GithubRepositories/RISE_toolbox')
   ```

2. You will need to adapt this path to conform with the location of the toolbox on your machine.
3. run rise_startup()

2.1.4 Updating RISE

New features are constantly added, efficiency is improved, users sometimes report bugs that are corrected. All this makes it necessary to update RISE every now and then in order to keep abreast of the latest changes and developments.

However, updating RISE depends on precisely how you installed it in the first place:

- If you downloaded a zip file, you will have to redownload a zip file even if the recent change was just an added comma.
- if instead you invested in opening a github account, with one click you will be able to update just the changes you don't have.
- with git, you would just execute the command

  ```
  git pull
  ```

2.2 Troubleshooting

2.3 RISE basics/basic principles

1. create an empty RISE object e.g.

   ```
   tao=rise.empty(0);
   ```

2. run methods(rise) or methods(tao) to see the functions/methods that can be applied to a RISE object
3. run those methods on r". e.g. "irf(r)", simulate(r)", solve(r)", etc. this will give you the default options of each method and tell you how you can modify the behavior of the method

2.4 Tutorial: A toy example

2.4.1 Foerster, Rubio-Ramirez, Waggoner and Zha (2014)

They consider the following model:

$$E_t\left[\begin{array}{c} 1-\beta\frac{\left(1-\frac{\kappa}{2}(\Pi_t-1)^2\right)Y_t}{\left(1-\frac{\kappa}{2}(\Pi_{t+1}-1)^2\right)Y_{t+1}}\frac{1}{e^{\mu_{t+1}}}\frac{R_t}{\Pi_{t+1}} \\ (1-\eta)+\eta\left(1-\frac{\kappa}{2}\left(\Pi_t-1\right)^2\right)Y_t+\beta\kappa\frac{\left(1-\frac{\kappa}{2}(\Pi_t-1)^2\right)}{\left(1-\frac{\kappa}{2}(\Pi_{t+1}-1)^2\right)}\left(\Pi_{t+1}-1\right)\Pi_{t+1}-\kappa\left(\Pi_t-1\right)\Pi_t \\ \left(\frac{R_{t-1}}{R_{ss}}\right)^{\rho}\Pi_t^{(1-\rho)\psi}\exp\left(\sigma\varepsilon_t\right)-\frac{R_t}{R_{ss}} \end{array}\right]=0$$

$$with$$

$$\mu_{t+1}=\bar{\mu}+\sigma\hat{\mu}_{t+1}.$$

The first equation is an Euler equation, the second equation a Phillips curve and the third equation a nonlinear Taylor rule.

The switching parameters are μ and ψ.

2.4.2 The RISE code

The RISE code with parameterization is given by

```
endogenous PAI,Y,R

exogenous EPS_R

parameters a_tp_1_2, a_tp_2_1, betta, eta, kappa, mu, mu_bar, psi, rhor, sigr
parameters(a,2) mu, psi

model
        1-betta*(1-.5*kappa*(PAI-1)^2)*Y*R/((1-.5*kappa*(PAI(+1)-1)^2)*Y(+1)*exp(mu)*PAI(+1));

        1-eta+eta*(1-.5*kappa*(PAI-1)^2)*Y+betta*kappa*(1-.5*kappa*(PAI-1)^2)*(PAI(+1)-1)*PAI(+1)/(1-
        -kappa*(PAI-1)*PAI;

        (R(-1)/steady_state(R))^rhor*(PAI/steady_state(PAI))^((1-rhor)*psi)*exp(sigr*EPS_R)-R/steady_


steady_state_model(unique,imposed)
    PAI=1;
    Y=(eta-1)/eta;
    R=exp(mu_bar)/betta*PAI;


parameterization
        a_tp_1_2,1-.9;
        a_tp_2_1,1-.9;
        betta, .99;
        kappa, 161;
        eta, 10;
        rhor, .8;
        sigr, 0.0025;
        mu_bar,0.02;
        mu(a,1), 0.03;
```

```
        mu(a,2), 0.01;
        psi(a,1), 3.1;
        psi(a,2), 0.9;
```

2.4.3 Running the example

Assume this example is saved in a file named frwz_nk.rs . The to run this example in Matlab, we run the following commands:

```
frwz=rise('frwz_nk'); % load the model and its parameterization

frwz=solve(frwz); % Solving the model

print_solution(frwz) % print the solution
```

2.5 How to find help?

2.6 Where to go from here

CHAPTER

THREE

RISE CAPABILITIES

3.1 Overview

3.2 Markov switching DSGE modeling

3.3 Markov switching SVAR modeling

3.4 Markov switching VAR modeling

3.5 Smooth transition VAR modeling

3.6 Time-varying parameter modeling

3.7 Maximum Likelihood and Bayesian Estimation

3.8 Differentiation

3.8.1 numerical differentiation

3.8.2 Symbolic differentiation

3.8.3 Automatic/Algorithmic differentiation

3.9 Time series

3.10 Reporting

3.11 Derivative-free optimization

3.12 Global sensitivity analysis

3.12.1 Monte Carlo filtering

3.12.2 High dimensional model representation

CHAPTER

FOUR

THE MARKOV SWITCHING DSGE INTERFACE

4.1 The general framework

The general form of the models is:

$$E_t \sum_{r_{t+1}=1}^{h} \pi_{r_t,r_{t+1}}\left(I_t\right) \tilde{d}_{r_t}\left(b_{t+1}\left(r_{t+1}\right), b_t\left(r_t\right), b_{t-1}, \varepsilon_t, \theta_{r_{t+1}}\right) = 0$$

- The switching of the parameters is governed by Markov processes and can be endogenous.
- Agents can have information about future events

4.2 The model file

4.2.1 Conventions

4.2.2 Variable declarations

4.2.3 Expressions

- **parameters and variables**
    - inside the model
    - outside the model
- operators
- **functions**
    - built-in functions
    - external/user-defined functions

4.2.4 model declaration

- model equations

- endogenous transition probabilities
- auxiliary parameters/variables
- inequality restrictions

4.2.5 auxiliary variables

4.2.6 initial and terminal conditions

4.2.7 shocks on exogenous variables

4.2.8 other general declarations

4.3 steady state

- finding the steady state with the RISE nonlinear solver
- using a steady state file
- using the steady state model

4.4 getting information about the model

4.5 deterministic simulation

4.6 stochastic solution and simulation

- computing the stochastic solution
- typology and ordering of variables
- first-order approximation
- second-order approximation
- third-order approximation
- fourth-order approximation
- fifth-order approximation

4.7 Estimation

4.8 Forecasting and conditional forecasting

4.9 Optimal policy

- optimal simple rules

- Commitment, discretion and loose commitment

CHAPTER

FIVE

MARKOV SWITCHING DYNAMIC STOCHASTIC GENERAL EQUILIBRIUM MODELING

5.1 methods

- [check_derivatives](dsge/check_derivatives)
- [check_optimum](dsge/check_optimum)
- [compute_steady_state](dsge/compute_steady_state)
- [create_estimation_blocks](dsge/create_estimation_blocks)
- [draw_parameter](dsge/draw_parameter)
- [dsge](dsge/dsge)
- [estimate](dsge/estimate)
- [filter](dsge/filter)
- [forecast](dsge/forecast)
- [forecast_real_time](dsge/forecast_real_time)
- [get](dsge/get)
- [historical_decomposition](dsge/historical_decomposition)
- [irf](dsge/irf)
- [is_stable_system](dsge/is_stable_system)
- [isnan](dsge/isnan)
- [load_parameters](dsge/load_parameters)
- [log_marginal_data_density](dsge/log_marginal_data_density)
- [log_posterior_kernel](dsge/log_posterior_kernel)
- [log_prior_density](dsge/log_prior_density)
- [monte_carlo_filtering](dsge/monte_carlo_filtering)
- [posterior_marginal_and_prior_densities](dsge/posterior_marginal_and_prior_densities)
- [posterior_simulator](dsge/posterior_simulator)

- [print_estimation_results](dsge/print_estimation_results)
- [print_solution](dsge/print_solution)
- [prior_plots](dsge/prior_plots)
- [report](dsge/report)
- [resid](dsge/resid)
- [set](dsge/set)
- [set_solution_to_companion](dsge/set_solution_to_companion)
- [simulate](dsge/simulate)
- [simulate_nonlinear](dsge/simulate_nonlinear)
- [simulation_diagnostics](dsge/simulation_diagnostics)
- [solve](dsge/solve)
- [solve_alternatives](dsge/solve_alternatives)
- [stoch_simul](dsge/stoch_simul)
- [theoretical_autocorrelations](dsge/theoretical_autocorrelations)
- [theoretical_autocovariances](dsge/theoretical_autocovariances)
- [variance_decomposition](dsge/variance_decomposition)

5.2 properties

- [definitions] -
- [equations] -
- [folders_paths] -
- [dsge_var] -
- [filename] -
- [legend] -
- [endogenous] -
- [exogenous] -
- [parameters] -
- [observables] -
- [markov_chains] -
- [options] -
- [estimation] -
- [solution] -
- [filtering] -

5.3 Synopsis and description on methods

5.3.1 check_derivatives

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.3.2 check_optimum

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/check_optimum is inherited from superclass RISE_GENERIC

5.3.3 compute_steady_state

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.3.4 create_estimation_blocks

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.3.5 draw_parameter

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/draw_parameter is inherited from superclass RISE_GENERIC

---

dsge

– no help found

5.3.6 estimate

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/estimate is inherited from superclass RISE_GENERIC

5.3.7 filter

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.3.8 forecast

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/forecast is inherited from superclass RISE_GENERIC

5.3.9 forecast_real_time

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.3.10 get

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/get is inherited from superclass RISE_GENERIC

---

historical_decomposition Computes historical decompositions of a DSGE model

5.4 Syntax

```
[Histdec,obj]=history_dec(obj)
[Histdec,obj]=history_dec(obj,varargin)
```

5.5 Inputs

- obj : [rise|dsge|rfvar|svar] model(s) for which to compute the decomposition. obj could be a vector of models
- varargin : standard optional inputs **coming in pairs**. Among which: - **histdec_start_date** : [char|numeric|{''}] : date at which the

  decomposition starts. If empty, the decomposition starts at he beginning of the history of the dataset

5.6 Outputs

- Histdec : [struct|cell array] structure or cell array of structures with the decompositions in each model. The decompositions are given in terms of: - the exogenous variables - **InitialConditions** : the effect of initial conditions - **risk** : measure of the effect of non-certainty equivalence - **switch** : the effect of switching (which is also a shock!!!) - **steady_state** : the contribution of the steady state

5.7 Remarks

- the elements that do not contribute to any of the variables are automatically discarded.
- **N.B** : a switching model is inherently nonlinear and so, strictly speaking, the type of decomposition we do for linear/linearized constant-parameter models is not feasible. RISE takes an approximation in which the variables, shocks and states matrices across states are averaged. The averaging weights are the smoothed probabilities.

5.8 Examples

See also:

Help for dsge/historical_decomposition is inherited from superclass RISE_GENERIC

5.8.1 irf

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.8.2 is_stable_system

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.8.3 isnan

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/isnan is inherited from superclass RISE_GENERIC

5.8.4 load_parameters

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/load_parameters is inherited from superclass RISE_GENERIC

5.8.5 log_marginal_data_density

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/log_marginal_data_density is inherited from superclass RISE_GENERIC

5.8.6 log_posterior_kernel

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/log_posterior_kernel is inherited from superclass RISE_GENERIC

5.8.7 log_prior_density

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/log_prior_density is inherited from superclass RISE_GENERIC

5.8.8 monte_carlo_filtering

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.8.9 posterior_marginal_and_prior_densities

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/posterior_marginal_and_prior_densities is inherited from superclass RISE_GENERIC

5.8.10 posterior_simulator

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/posterior_simulator is inherited from superclass RISE_GENERIC

5.8.11 print_estimation_results

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/print_estimation_results is inherited from superclass RISE_GENERIC

5.8.12 print_solution

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.8.13 prior_plots

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/prior_plots is inherited from superclass RISE_GENERIC

REPORT assigns the elements of interest to a rise_report.report object

5.9 Syntax

::

- REPORT(rise.empty(0)) : displays the default inputs
- REPORT(obj,destination_root,rep_items) : assigns the reported elements in rep_items to destination_root
- REPORT(obj,destination_root,rep_items,varargin) : assigns varargin to obj before doing the rest

5.10 Inputs

- obj : [rise|dsge]
- destination_root : [rise_report.report] : handle for the actual report
- rep_items : [char|cellstr] : list of desired items to report. This list can only include : 'endogenous', 'exogenous', 'observables', 'parameters', 'solution', 'estimation', 'estimation_statistics', 'equations', 'code'

5.11 Outputs

none

5.12 Description

5.13 Examples

See also:

Help for dsge/report is inherited from superclass RISE_GENERIC

5.13.1 resid

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.13.2 set

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.13.3 set_solution_to_companion

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.13.4 simulate

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.13.5 simulate_nonlinear

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.13.6 simulation_diagnostics

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/simulation_diagnostics is inherited from superclass RISE_GENERIC

5.13.7 solve

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.13.8 solve_alternatives

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

5.13.9 stoch_simul

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/stoch_simul is inherited from superclass RISE_GENERIC

5.13.10 theoretical_autocorrelations

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/theoretical_autocorrelations is inherited from superclass RISE_GENERIC

5.13.11 theoretical_autocovariances

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/theoretical_autocovariances is inherited from superclass RISE_GENERIC

5.13.12 variance_decomposition

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for dsge/variance_decomposition is inherited from superclass RISE_GENERIC

CHAPTER

SIX

REDUCED-FORM VAR MODELING

6.1 methods

- [check_identification](rfvar/check_identification)
- [check_optimum](rfvar/check_optimum)
- [draw_parameter](rfvar/draw_parameter)
- [estimate](rfvar/estimate)
- [forecast](rfvar/forecast)
- [get](rfvar/get)
- [historical_decomposition](rfvar/historical_decomposition)
- [irf](rfvar/irf)
- [isnan](rfvar/isnan)
- [load_parameters](rfvar/load_parameters)
- [log_marginal_data_density](rfvar/log_marginal_data_density)
- [log_posterior_kernel](rfvar/log_posterior_kernel)
- [log_prior_density](rfvar/log_prior_density)
- [msvar_priors](rfvar/msvar_priors)
- [posterior_marginal_and_prior_densities](rfvar/posterior_marginal_and_prior_densities)
- [posterior_simulator](rfvar/posterior_simulator)
- [print_estimation_results](rfvar/print_estimation_results)
- [prior_plots](rfvar/prior_plots)
- [report](rfvar/report)
- [rfvar](rfvar/rfvar)
- [set](rfvar/set)
- [set_solution_to_companion](rfvar/set_solution_to_companion)
- [simulate](rfvar/simulate)
- [simulation_diagnostics](rfvar/simulation_diagnostics)
- [solve](rfvar/solve)

- [stoch_simul](rfvar/stoch_simul)
- [structural_form](rfvar/structural_form)
- [template](rfvar/template)
- [theoretical_autocorrelations](rfvar/theoretical_autocorrelations)
- [theoretical_autocovariances](rfvar/theoretical_autocovariances)
- [variance_decomposition](rfvar/variance_decomposition)

6.2 properties

- [identification] -
- [structural_shocks] -
- [nonlinear_restrictions] -
- [constant] -
- [nlags] -
- [legend] -
- [endogenous] -
- [exogenous] -
- [parameters] -
- [observables] -
- [markov_chains] -
- [options] -
- [estimation] -
- [solution] -
- [filtering] -

6.3 Synopsis and description on methods

6.3.1 check_identification

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

6.3.2 check_optimum

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/check_optimum is inherited from superclass RISE_GENERIC

6.3.3 draw_parameter

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/draw_parameter is inherited from superclass RISE_GENERIC

6.3.4 estimate

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/estimate is inherited from superclass RISE_GENERIC

6.3.5 forecast

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/forecast is inherited from superclass RISE_GENERIC

6.3.6 get

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/get is inherited from superclass RISE_GENERIC

---

historical_decomposition Computes historical decompositions of a DSGE model

6.4 Syntax

```
[Histdec,obj]=history_dec(obj)
[Histdec,obj]=history_dec(obj,varargin)
```

6.5 Inputs

- obj : [rise|dsge|rfvar|svar] model(s) for which to compute the decomposition. obj could be a vector of models
- varargin : standard optional inputs **coming in pairs**. Among which: - **histdec_start_date** : [char|numeric|{''}] : date at which the

  decomposition starts. If empty, the decomposition starts at he beginning of the history of the dataset

6.6 Outputs

- Histdec : [struct|cell array] structure or cell array of structures with the decompositions in each model. The decompositions are given in terms of: - the exogenous variables - **InitialConditions** : the effect of initial conditions - **risk** : measure of the effect of non-certainty equivalence - **switch** : the effect of switching (which is also a shock!!!) - **steady_state** : the contribution of the steady state

6.7 Remarks

- the elements that do not contribute to any of the variables are automatically discarded.
- **N.B** : a switching model is inherently nonlinear and so, strictly speaking, the type of decomposition we do for linear/linearized constant-parameter models is not feasible. RISE takes an approximation in which the variables, shocks and states matrices across states are averaged. The averaging weights are the smoothed probabilities.

6.8 Examples

See also:

Help for rfvar/historical_decomposition is inherited from superclass RISE_GENERIC

6.8.1 irf

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/irf is inherited from superclass RISE_GENERIC

6.8.2 isnan

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/isnan is inherited from superclass RISE_GENERIC

6.8.3 load_parameters

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/load_parameters is inherited from superclass RISE_GENERIC

6.8.4 log_marginal_data_density

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/log_marginal_data_density is inherited from superclass RISE_GENERIC

6.8.5 log_posterior_kernel

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/log_posterior_kernel is inherited from superclass RISE_GENERIC

6.8.6 log_prior_density

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/log_prior_density is inherited from superclass RISE_GENERIC

6.8.7 msvar_priors

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/msvar_priors is inherited from superclass SVAR

6.8.8 posterior_marginal_and_prior_densities

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/posterior_marginal_and_prior_densities is inherited from superclass RISE_GENERIC

6.8.9 posterior_simulator

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/posterior_simulator is inherited from superclass RISE_GENERIC

6.8.10 print_estimation_results

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/print_estimation_results is inherited from superclass RISE_GENERIC

6.8.11 prior_plots

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/prior_plots is inherited from superclass RISE_GENERIC

REPORT assigns the elements of interest to a rise_report.report object

6.9 Syntax

::

- REPORT(rise.empty(0)) : displays the default inputs
- REPORT(obj,destination_root,rep_items) : assigns the reported elements in rep_items to destination_root
- REPORT(obj,destination_root,rep_items,varargin) : assigns varargin to obj before doing the rest

6.10 Inputs

- obj : [rise|dsge]
- destination_root : [rise_report.report] : handle for the actual report
- rep_items : [char|cellstr] : list of desired items to report. This list can only include : 'endogenous', 'exogenous', 'observables', 'parameters', 'solution', 'estimation', 'estimation_statistics', 'equations', 'code'

6.11 Outputs

none

6.12 Description

6.13 Examples

See also:

Help for rfvar/report is inherited from superclass RISE_GENERIC

---

rfvar

– no help found

6.13.1 set

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/set is inherited from superclass RISE_GENERIC

6.13.2 set_solution_to_companion

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/set_solution_to_companion is inherited from superclass SVAR

6.13.3 simulate

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/simulate is inherited from superclass RISE_GENERIC

6.13.4 simulation_diagnostics

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/simulation_diagnostics is inherited from superclass RISE_GENERIC

6.13.5 solve

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

6.13.6 stoch_simul

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/stoch_simul is inherited from superclass RISE_GENERIC

---

structural_form finds A structural form given the imposed restrictions

6.14 Syntax

```
newobj=structural_form(obj)
newobj=structural_form(obj,initcall)
```

6.15 Inputs

- **obj** : [rfvar] : reduced form VAR object
- varargin : standard optional inputs **coming in pairs**. Among which:
  - **restrict_lags : [cell array|{''}] : restrictions on the lag** structure. There are two equivalent syntaxes for this:
    - {'var_name1@var_name2{lag}'}

    * {'alag(var_name1,var_name2)'} : here alag should be understood as a-lag, where lag is the "lag" e.g. a1(infl,unemp) means unemp does not enter the infl equation at lag 1.

- **restrict_irf_sign : [cell array|{''}] : sign restrictions on the** impulse responses. The general syntax is {'var_name{period}@shock_name','sign'} and the default period is "0" (for contemporaneous). That means {'var_name{0}@shock_name','+'} and {'var_name@shock_name','+'} are equivalent
- **restrict_irf_zero : [cell array|{''}] : zero restrictions on the** impulse responses. The general syntax is {'var_name{period}@shock_name'} and the default period is "0" (for contemporaneous). That means {'var_name{0}@shock_name'} and {'var_name@shock_name'} are equivalent
- **structural_shocks : [cell array|{''}] : List of structural** shocks. The shock names can be entered with or without their description. For instance : - {'E_PAI','E_U','E_MP'} - {'E_PAI','''inflation shock''','E_U','''unempl shock''','E_MP'}
- **irf_sample_max** [[numeric|{10000}]][maximum number of trials in] the drawing of rotation matrices

6.16 Outputs

- newobj : [rfvar]: new rfvar object with the drawn structural form

6.17 Description

- RISE automatically orders the endogenous variables alphabetically and tags each equation with one of the endogenous variables. This may be useful for understanding the behavior of **restrict_lags** above.
- The Choleski identification scheme is not implemented per se. The user has to explicitly enter the zeros in the right places. This gives the flexibility in implementing the restrictions. For instance, one could imagine a scheme in which choleski restrictions hold only in the long run.
- With only zero restrictions, one cannot expect the impulse responses to automatically have the correct sign. The rotation imposes zero restrictions but not the sign. If you would like to have correctly-signed impulse responses there are two choices: - explicitly add sign restrictions - multiply the impulse responses for the wrongly-signed shock with minus.
- Many periods can be entered simultaneously. For instance 'var_name{0,3,5,10:20,inf}@shock_name'
- long-run restrictions are denoted by "inf". For instance 'var_name{inf}@shock_name'
- Identification for Markov switching VARs is not implemented/supported.

6.18 Examples

See also:

---

template

  – no help found

6.18.1 theoretical_autocorrelations

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/theoretical_autocorrelations is inherited from superclass RISE_GENERIC

6.18.2 theoretical_autocovariances

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/theoretical_autocovariances is inherited from superclass RISE_GENERIC

6.18.3 variance_decomposition

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for rfvar/variance_decomposition is inherited from superclass RISE_GENERIC

CHAPTER

SEVEN

STRUCTURAL VAR MODELING

7.1 methods

- [check_optimum](svar/check_optimum)
- [draw_parameter](svar/draw_parameter)
- [estimate](svar/estimate)
- [forecast](svar/forecast)
- [get](svar/get)
- [historical_decomposition](svar/historical_decomposition)
- [irf](svar/irf)
- [isnan](svar/isnan)
- [load_parameters](svar/load_parameters)
- [log_marginal_data_density](svar/log_marginal_data_density)
- [log_posterior_kernel](svar/log_posterior_kernel)
- [log_prior_density](svar/log_prior_density)
- [msvar_priors](svar/msvar_priors)
- [posterior_marginal_and_prior_densities](svar/posterior_marginal_and_prior_densities)
- [posterior_simulator](svar/posterior_simulator)
- [print_estimation_results](svar/print_estimation_results)
- [prior_plots](svar/prior_plots)
- [report](svar/report)
- [set](svar/set)
- [set_solution_to_companion](svar/set_solution_to_companion)
- [simulate](svar/simulate)
- [simulation_diagnostics](svar/simulation_diagnostics)
- [solve](svar/solve)
- [stoch_simul](svar/stoch_simul)
- [svar](svar/svar)

- [template](svar/template)
- [theoretical_autocorrelations](svar/theoretical_autocorrelations)
- [theoretical_autocovariances](svar/theoretical_autocovariances)
- [variance_decomposition](svar/variance_decomposition)

7.2 properties

- [constant] -
- [nlags] -
- [legend] -
- [endogenous] -
- [exogenous] -
- [parameters] -
- [observables] -
- [markov_chains] -
- [options] -
- [estimation] -
- [solution] -
- [filtering] -

7.3 Synopsis and description on methods

7.3.1 check_optimum

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/check_optimum is inherited from superclass RISE_GENERIC

7.3.2 draw_parameter

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/draw_parameter is inherited from superclass RISE_GENERIC

7.3.3 estimate

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/estimate is inherited from superclass RISE_GENERIC

7.3.4 forecast

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/forecast is inherited from superclass RISE_GENERIC

7.3.5 get

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/get is inherited from superclass RISE_GENERIC

---

historical_decomposition Computes historical decompositions of a DSGE model

7.4 Syntax

```
[Histdec,obj]=history_dec(obj)
[Histdec,obj]=history_dec(obj,varargin)
```

7.5 Inputs

- obj : [rise|dsge|rfvar|svar] model(s) for which to compute the decomposition. obj could be a vector of models
- varargin : standard optional inputs **coming in pairs**. Among which: - **histdec_start_date** : [char|numeric|{''}] : date at which the

    decomposition starts. If empty, the decomposition starts at he beginning of the history of the dataset

7.6 Outputs

- Histdec : [struct|cell array] structure or cell array of structures with the decompositions in each model. The decompositions are given in terms of: - the exogenous variables - **InitialConditions** : the effect of initial conditions - **risk** : measure of the effect of non-certainty equivalence - **switch** : the effect of switching (which is also a shock!!!) - **steady_state** : the contribution of the steady state

7.7 Remarks

- the elements that do not contribute to any of the variables are automatically discarded.
- **N.B** : a switching model is inherently nonlinear and so, strictly speaking, the type of decomposition we do for linear/linearized constant-parameter models is not feasible. RISE takes an approximation in which the variables, shocks and states matrices across states are averaged. The averaging weights are the smoothed probabilities.

7.8 Examples

See also:

Help for svar/historical_decomposition is inherited from superclass RISE_GENERIC

7.8.1 irf

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/irf is inherited from superclass RISE_GENERIC

7.8.2 isnan

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/isnan is inherited from superclass RISE_GENERIC

7.8.3 load_parameters

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/load_parameters is inherited from superclass RISE_GENERIC

7.8.4 log_marginal_data_density

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/log_marginal_data_density is inherited from superclass RISE_GENERIC

7.8.5 log_posterior_kernel

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/log_posterior_kernel is inherited from superclass RISE_GENERIC

7.8.6 log_prior_density

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/log_prior_density is inherited from superclass RISE_GENERIC

7.8.7 msvar_priors

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

7.8.8 posterior_marginal_and_prior_densities

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/posterior_marginal_and_prior_densities is inherited from superclass RISE_GENERIC

7.8.9 posterior_simulator

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/posterior_simulator is inherited from superclass RISE_GENERIC

7.8.10 print_estimation_results

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/print_estimation_results is inherited from superclass RISE_GENERIC

7.8.11 prior_plots

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/prior_plots is inherited from superclass RISE_GENERIC

REPORT assigns the elements of interest to a rise_report.report object

7.9 Syntax

::

- REPORT(rise.empty(0)) : displays the default inputs
- REPORT(obj,destination_root,rep_items) : assigns the reported elements in rep_items to destination_root
- REPORT(obj,destination_root,rep_items,varargin) : assigns varargin to obj before doing the rest

7.10 Inputs

- obj : [rise|dsge]
- destination_root : [rise_report.report] : handle for the actual report
- rep_items : [char|cellstr] : list of desired items to report. This list can only include : 'endogenous', 'exogenous', 'observables', 'parameters', 'solution', 'estimation', 'estimation_statistics', 'equations', 'code'

7.11 Outputs

none

7.12 Description

7.13 Examples

See also:

Help for svar/report is inherited from superclass RISE_GENERIC

7.13.1 set

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/set is inherited from superclass RISE_GENERIC

7.13.2 set_solution_to_companion

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

7.13.3 simulate

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/simulate is inherited from superclass RISE_GENERIC

7.13.4 simulation_diagnostics

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/simulation_diagnostics is inherited from superclass RISE_GENERIC

7.13.5 solve

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

7.13.6 stoch_simul

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/stoch_simul is inherited from superclass RISE_GENERIC

---

svar

– no help found

---

template

– no help found

7.13.7 theoretical_autocorrelations

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/theoretical_autocorrelations is inherited from superclass RISE_GENERIC

7.13.8 theoretical_autocovariances

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/theoretical_autocovariances is inherited from superclass RISE_GENERIC

7.13.9 variance_decomposition

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

Help for svar/variance_decomposition is inherited from superclass RISE_GENERIC

CHAPTER

EIGHT

TIME SERIES

8.1 Constructor

- [ts](ts/ts)

8.2 Visualization

- [head](ts/head)
- [index](ts/index)
- [describe](ts/describe)
- [display](ts/display)
- [jbtest](ts/jbtest)
- [kurtosis](ts/kurtosis)
- [isfinite](ts/isfinite)
- [isinf](ts/isinf)
- [isnan](ts/isnan)
- [ge](ts/ge)
- [get](ts/get)
- [gt](ts/gt)
- [le](ts/le)
- [lt](ts/lt)
- [max](ts/max)
- [mean](ts/mean)
- [median](ts/median)
- [min](ts/min)
- [mode](ts/mode)
- [ne](ts/ne)
- [quantile](ts/quantile)

- [range](ts/range)
- [skewness](ts/skewness)
- [sum](ts/sum)
- [tail](ts/tail)
- [var](ts/var)
- [std](ts/std)
- [spectrum](ts/spectrum)
- [sort](ts/sort)

8.3 Graphing

- [bar](ts/bar)
- [barh](ts/barh)
- [boxplot](ts/boxplot)
- [hist](ts/hist)
- [plot](ts/plot)
- [plotyy](ts/plotyy)

8.4 Calculus

- [acos](ts/acos)
- [acosh](ts/acosh)
- [acot](ts/acot)
- [acoth](ts/acoth)
- [aggregate](ts/aggregate)
- [allmean](ts/allmean)
- [apply](ts/apply)
- [asin](ts/asin)
- [asinh](ts/asinh)
- [atan](ts/atan)
- [atanh](ts/atanh)
- [bsxfun](ts/bsxfun)
- [corr](ts/corr)
- [corrcoef](ts/corrcoef)
- [cos](ts/cos)
- [cosh](ts/cosh)
- [cot](ts/cot)

- [coth](ts/coth)
- [cov](ts/cov)
- [cumprod](ts/cumprod)
- [cumsum](ts/cumsum)
- [decompose_series](ts/decompose_series)
- [eq](ts/eq)
- [exp](ts/exp)
- [hpfilter](ts/hpfilter)
- [interpolate](ts/interpolate)
- [intersect](ts/intersect)
- [log](ts/log)
- [minus](ts/minus)
- [mpower](ts/mpower)
- [mrdivide](ts/mrdivide)
- [mtimes](ts/mtimes)
- [plus](ts/plus)
- [power](ts/power)
- [rdivide](ts/rdivide)
- [sin](ts/sin)
- [sinh](ts/sinh)
- [transform](ts/transform)
- [times](ts/times)
- [uminus](ts/uminus)

8.5 Lookarounds

- [pages2struct](ts/pages2struct)
- [subsasgn](ts/subsasgn)
- [subsref](ts/subsref)

8.6 Utilities

- [and](ts/and)
- [cat](ts/cat)
- [collect](ts/collect)
- [ctranspose](ts/ctranspose)
- [double](ts/double)

- [drop](ts/drop)
- [dummy](ts/dummy)
- [expanding](ts/expanding)
- [fanchart](ts/fanchart)
- [horzcat](ts/horzcat)
- [nan](ts/nan)
- [numel](ts/numel)
- [ones](ts/ones)
- [rand](ts/rand)
- [randn](ts/randn)
- [regress](ts/regress)
- [reset_start_date](ts/reset_start_date)
- [rolling](ts/rolling)
- [automatic_model_selection](ts/automatic_model_selection)
- [transpose](ts/transpose)
- [zeros](ts/zeros)
- [values](ts/values)
- [step_dummy](ts/step_dummy)

8.7 properties

- [varnames] -
- [start] -
- [finish] -
- [frequency] -
- [NumberOfObservations] -
- [NumberOfPages] -
- [NumberOfVariables] -

8.8 Synopsis and description on methods

8.8.1 acos

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.2 acosh

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.3 acot

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.4 acoth

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.5 aggregate

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.6 allmean

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.7 and

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.8 apply

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.9 asin

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.10 asinh

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.11 atan

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.12 atanh

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.13 automatic_model_selection

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.14 bar

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.15 barh

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.16 boxplot

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.8.17 bsxfun

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

---

cat concatenates time series along the specified dimension

8.9 Syntax

:: db=cat(1,db1,db2,...,dbn) db=cat(2,db1,db2,...,dbn) db=cat(3,db1,db2,...,dbn)

8.10 Inputs

- **dim** [1|2|3] : dimension along which concatenation is done
- **db1, db2,...,dbn** [ts] : time series

8.11 Outputs

- **db** [ts] : time series with concatenated series

8.12 Description

- all times series must be of the same frequency
- Concatenation along the second dimension requires that variables have the same number of columns if no names are specified
- if names are specified in the first time series, then names should be specified in all of the others as well.
- empty time series are discarded but there should be at least one non-empty time series

8.13 Examples

See also:

8.13.1 collect

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.2 corr

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.3 corrcoef

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.4 cos

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.5 cosh

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.6 cot

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.7 coth

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.8 cov

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.9 ctranspose

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.10 cumprod

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.11 cumsum

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.12 decompose_series

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.13 describe

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.14 display

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.15 double

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.16 drop

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.17 dummy

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.18 eq

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.19 exp

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.20 expanding

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.21 fanchart

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.22 ge

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.23 get

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.24 gt

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.25 head

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.26 hist

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.27 horzcat

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.28 hpfilter

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.29 index

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.30 interpolate

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.31 intersect

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.32 isfinite

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.33 isinf

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.34 isnan

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.35 jbtest

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.36 kurtosis

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.37 le

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.38 log

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.39 lt

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.40 max

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.41 mean

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.42 median

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.43 min

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.44 minus

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.45 mode

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.46 mpower

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.47 mrdivide

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.48 mtimes

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.49 nan

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.50 ne

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.13.51 numel

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

---

ones overloads ones for ts objects

8.14 Syntax

:: db=ts.ones(start_date,varargin)

8.15 Inputs

- **start_date** : [numeric|char]: a valid time series (ts) date
- varargin : [numeric]: arguments to matlab's **ones** function.

8.16 Outputs

- **db** : [ts]: a time series

8.17 Description

- this is a static method and so it has to be called with the **ts.** prefix
- ts.ones does not allow more than 3 dimensions

8.18 Examples

db=ts.ones(1990,10,1) db=ts.ones('1990',10,3) db=ts.ones('1990Q3',10,5,100)

See also:

8.18.1 pages2struct

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.2 plot

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.3 plotyy

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.4 plus

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.5 power

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.6 quantile

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.7 rand

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.8 randn

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.9 range

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.10 rdivide

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.11 regress

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.12 reset_start_date

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.13 rolling

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.14 sin

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.15 sinh

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.16 skewness

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.17 sort

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.18 spectrum

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.19 std

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.20 step_dummy

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.21 subsasgn

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.22 subsref

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.23 sum

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.24 tail

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.25 times

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.26 transform

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

transpose

– no help found

ts

– no help found

8.18.27 uminus

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.28 values

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.29 var

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

8.18.30 zeros

H1 line

Syntax

Inputs

Outputs

Description

Examples

See also:

CHAPTER

NINE

MARKOV CHAIN MONTE CARLO FOR BAYESIAN ESTIMATION

9.1 Metropolis Hastings

9.2 Gibbs sampling

9.3 Marginal data density

9.3.1 Laplace approximation

9.3.2 Modified harmonic mean

9.3.3 Waggoner and Zha (2008)

9.3.4 Mueller

9.3.5 Chib and Jeliazkov

CHAPTER

TEN

DERIVATIVE-FREE OPTIMIZATION

- differential evolution
- bee algorithm
- biogeography
- studga
- ants

CHAPTER

ELEVEN

MONTE CARLO FILTERING

11.1 methods

- [addlistener](mcf/addlistener)
- [cdf](mcf/cdf)
- [cdf_plot](mcf/cdf_plot)
- [correlation_patterns_plot](mcf/correlation_patterns_plot)
- [delete](mcf/delete)
- [eq](mcf/eq)
- [findobj](mcf/findobj)
- [findprop](mcf/findprop)
- [ge](mcf/ge)
- [gt](mcf/gt)
- [isvalid](mcf/isvalid)
- [kolmogorov_smirnov_test](mcf/kolmogorov_smirnov_test)
- [le](mcf/le)
- [lt](mcf/lt)
- [mcf](mcf/mcf)
- [ne](mcf/ne)
- [notify](mcf/notify)
- [scatter](mcf/scatter)

11.2 properties

- [lb] -
- [ub] -
- [nsim] -
- [procedure] -

- [parameter_names] -
- [samples] -
- [is_behaved] -
- [nparam] -
- [is_sampled] -
- [check_behavior] -
- [number_of_outputs] -
- [user_outputs] -
- [known_procedures] -

11.3 Synopsis and description on methods

ADDLISTENER Add listener for event. el = ADDLISTENER(hSource, 'Eventname', Callback) creates a listener for the event named Eventname, the source of which is handle object hSource. If hSource is an array of source handles, the listener responds to the named event on any handle in the array. The Callback is a function handle that is invoked when the event is triggered.

el = ADDLISTENER(hSource, PropName, 'Eventname', Callback) adds a listener for a property event. Eventname must be one of the strings 'PreGet', 'PostGet', 'PreSet', and 'PostSet'. PropName must be either a single property name or cell array of property names, or a single meta.property or array of meta.property objects. The properties must belong to the class of hSource. If hSource is scalar, PropName can include dynamic properties.

For all forms, addlistener returns an event.listener. To remove a listener, delete the object returned by addlistener. For example, delete(el) calls the handle class delete method to remove the listener and delete it from the workspace.

See also MCF, NOTIFY, DELETE, EVENT.LISTENER, META.PROPERTY, EVENTS, DYNAMICPROPS

Help for mcf/addlistener is inherited from superclass HANDLE

Reference page in Help browser doc mcf/addlistener

---

cdf

– no help found

---

cdf_plot

– no help found

---

correlation_patterns_plot

– no help found

DELETE Delete a handle object. The DELETE method deletes a handle object but does not clear the handle from the workspace. A deleted handle is no longer valid.

DELETE(H) deletes the handle object H, where H is a scalar handle.

See also MCF, MCF/ISVALID, CLEAR

Help for mcf/delete is inherited from superclass HANDLE

Reference page in Help browser doc mcf/delete

11.3.1 eq

== (EQ) Test handle equality. Handles are equal if they are handles for the same object.

H1 == H2 performs element-wise comparisons between handle arrays H1 and H2. H1 and H2 must be of the same dimensions unless one is a scalar. The result is a logical array of the same dimensions, where each element is an element-wise equality result.

If one of H1 or H2 is scalar, scalar expansion is performed and the result will match the dimensions of the array that is not scalar.

TF = EQ(H1, H2) stores the result in a logical array of the same dimensions.

See also MCF, MCF/GE, MCF/GT, MCF/LE, MCF/LT, MCF/NE

Help for mcf/eq is inherited from superclass HANDLE

FINDOBJ Find objects matching specified conditions. The FINDOBJ method of the HANDLE class follows the same syntax as the MATLAB FINDOBJ command, except that the first argument must be an array of handles to objects.

HM = FINDOBJ(H, <conditions>) searches the handle object array H and returns an array of handle objects matching the specified conditions. Only the public members of the objects of H are considered when evaluating the conditions.

See also FINDOBJ, MCF

Help for mcf/findobj is inherited from superclass HANDLE

Reference page in Help browser doc mcf/findobj

FINDPROP Find property of MATLAB handle object. p = FINDPROP(H,'PROPNAME') finds and returns the META.PROPERTY object associated with property name PROPNAME of scalar handle object H. PROPNAME must be a string. It can be the name of a property defined by the class of H or a dynamic property added to scalar object H.

If no property named PROPNAME exists for object H, an empty META.PROPERTY array is returned.

See also MCF, MCF/FINDOBJ, DYNAMICPROPS, META.PROPERTY

Help for mcf/findprop is inherited from superclass HANDLE

Reference page in Help browser doc mcf/findprop

11.3.2 ge

>= (GE) Greater than or equal relation for handles. H1 >= H2 performs element-wise comparisons between handle arrays H1 and H2. H1 and H2 must be of the same dimensions unless one is a scalar. The result is a logical array of the same dimensions, where each element is an element-wise >= result.

If one of H1 or H2 is scalar, scalar expansion is performed and the result will match the dimensions of the array that is not scalar.

TF = GE(H1, H2) stores the result in a logical array of the same dimensions.

See also MCF, MCF/EQ, MCF/GT, MCF/LE, MCF/LT, MCF/NE

Help for mcf/ge is inherited from superclass HANDLE

11.3.3 gt

> (GT) Greater than relation for handles. H1 > H2 performs element-wise comparisons between handle arrays H1 and H2. H1 and H2 must be of the same dimensions unless one is a scalar. The result is a logical array of the same dimensions, where each element is an element-wise > result.

If one of H1 or H2 is scalar, scalar expansion is performed and the result will match the dimensions of the array that is not scalar.

TF = GT(H1, H2) stores the result in a logical array of the same dimensions.

See also MCF, MCF/EQ, MCF/GE, MCF/LE, MCF/LT, MCF/NE

Help for mcf/gt is inherited from superclass HANDLE

ISVALID Test handle validity. TF = ISVALID(H) performs an element-wise check for validity on the handle elements of H. The result is a logical array of the same dimensions as H, where each element is the element-wise validity result.

A handle is invalid if it has been deleted or if it is an element of a handle array and has not yet been initialized.

See also MCF, MCF/DELETE

Help for mcf/isvalid is inherited from superclass HANDLE

Reference page in Help browser doc mcf/isvalid

11.3.4 kolmogorov_smirnov_test

tests the equality of two distributions using their CDFs

11.3.5 le

<= (LE) Less than or equal relation for handles. Handles are equal if they are handles for the same object. All comparisons use a number associated with each handle object. Nothing can be assumed about the result of a handle comparison except that the repeated comparison of two handles in the same MATLAB session will yield the same result. The order of handle values is purely arbitrary and has no connection to the state of the handle objects being compared.

H1 <= H2 performs element-wise comparisons between handle arrays H1 and H2. H1 and H2 must be of the same dimensions unless one is a scalar. The result is a logical array of the same dimensions, where each element is an element-wise >= result.

If one of H1 or H2 is scalar, scalar expansion is performed and the result will match the dimensions of the array that is not scalar.

TF = LE(H1, H2) stores the result in a logical array of the same dimensions.

See also MCF, MCF/EQ, MCF/GE, MCF/GT, MCF/LT, MCF/NE

Help for mcf/le is inherited from superclass HANDLE

11.3.6 lt

< (LT) Less than relation for handles. H1 < H2 performs element-wise comparisons between handle arrays H1 and H2. H1 and H2 must be of the same dimensions unless one is a scalar. The result is a logical array of the same dimensions, where each element is an element-wise < result.

If one of H1 or H2 is scalar, scalar expansion is performed and the result will match the dimensions of the array that is not scalar.

TF = LT(H1, H2) stores the result in a logical array of the same dimensions.

See also MCF, MCF/EQ, MCF/GE, MCF/GT, MCF/LE, MCF/NE

Help for mcf/lt is inherited from superclass HANDLE

---

mcf

– no help found

11.3.7 ne

~= (NE) Not equal relation for handles. Handles are equal if they are handles for the same object and are unequal otherwise.

H1 ~= H2 performs element-wise comparisons between handle arrays H1 and H2. H1 and H2 must be of the same dimensions unless one is a scalar. The result is a logical array of the same dimensions, where each element is an element-wise equality result.

If one of H1 or H2 is scalar, scalar expansion is performed and the result will match the dimensions of the array that is not scalar.

TF = NE(H1, H2) stores the result in a logical array of the same dimensions.

See also MCF, MCF/EQ, MCF/GE, MCF/GT, MCF/LE, MCF/LT

Help for mcf/ne is inherited from superclass HANDLE

NOTIFY Notify listeners of event. NOTIFY(H,'EVENTNAME') notifies listeners added to the event named EVENTNAME on handle object array H that the event is taking place. H is the array of handles to objects triggering the event, and EVENTNAME must be a string.

NOTIFY(H,'EVENTNAME',DATA) provides a way of encapsulating information about an event which can then be accessed by each registered listener. DATA must belong to the EVENT.EVENTDATA class.

See also MCF, MCF/ADDLISTENER, EVENT.EVENTDATA, EVENTS

Help for mcf/notify is inherited from superclass HANDLE

Reference page in Help browser doc mcf/notify

---

scatter

– no help found

CHAPTER

TWELVE

HIGH DIMENSIONAL MODEL REPRESENTATION

12.1 methods

- [estimate](hdmr/estimate)
- [first_order_effect](hdmr/first_order_effect)
- [hdmr](hdmr/hdmr)
- [metamodel](hdmr/metamodel)
- [plot_fit](hdmr/plot_fit)
- [polynomial_evaluation](hdmr/polynomial_evaluation)
- [polynomial_integration](hdmr/polynomial_integration)
- [polynomial_multiplication](hdmr/polynomial_multiplication)

12.2 properties

- [N] -
- [Nobs] -
- [n] -
- [output_nbr] -
- [theta] -
- [theta_low] -
- [theta_high] -
- [g] -
- [x] -
- [expansion_order] -
- [pol_max_order] -
- [poly_coefs] -

- [Indices] -
- [coefficients] -
- [aggregate] -
- [f0] -
- [D] -
- [sample_percentage] -
- [optimal] -
- [param_names] -

12.3 Synopsis and description on methods

---

estimate

– no help found

---

first_order_effect

– no help found

---

hdmr

– no help found

---

metamodel

– no help found

---

plot_fit

– no help found

12.3.1 polynomial_evaluation

later on, the function that normalizes could come in here so that the normalization is done according to the hdmr_type of polynomial chosen.

12.3.2 polynomial_integration

polynomial is of the form a0+a1*x+...+ar*x^r the integral is then a0*x+a1/2*x^2+...+ar/(r+1)*x^(r+1)

12.3.3 polynomial_multiplication

each polynomial is of the form a0+a1*x+...+ar*x^r

CHAPTER

THIRTEEN

CONTRIBUTING TO RISE

13.1 contributing new code

13.2 contributing by helping maintain existing code

13.3 other ways to contribute

13.4 recommended development setup

13.5 RISE structure

13.6 useful links, FAQ, checklist

CHAPTER

FOURTEEN

ACKNOWLEDGEMENTS

Many people have, oftentimes unknowingly, provided help in the form of reporting bugs, making suggestions, asking challenging questions, etc. I would like to single out a few of them but the list is far from exhaustive:

- Dan Waggoner
- Doug Laxton
- Eric Leeper
- Jesper Linde
- Jim Nason
- Kjetil Olsen
- Kostas Theodoridis
- Leif Brubakk
- Marco Ratto
- Michel Juillard
- Pablo Winnant (dolo)
- Pelin Ilbas
- Raf Wouters
- Tao Zha

CHAPTER

FIFTEEN

BIBLIOGRAPHY

CHAPTER

SIXTEEN

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*