Reference Manual

Generated by Doxygen 1.6.1

Tue Oct 23 22:18:34 2012

Contents

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

Class Documentation

3.1 parameters Class Reference

Object to store parameter values for VFI problem.

`#include <global.h>`

Public Member Functions

- void load (const char *)

  Function to load VFI parameter values to parameters object.

Public Attributes

- REAL eta

  Coefficient of relative risk aversion.

- REAL beta

  Time discount factor.

- REAL alpha

  Share of capital in the production function.

- REAL delta

  Rate of capital depreciation.

- REAL mu

  TFP mean.

- REAL rho

  TFP persistence.

- REAL sigma

  TFP volatility.

- REAL lambda

  Number of standard deviations for AR1 approximation.

- int nk

  Number of values in capital grid.

- int nz

  Number of values in TFP grid.

- REAL tol

  Tolerance for convergence.

- char maxtype

  Maximization method - choices are 'g' (grid) and 'b' (binary search).

- int howard

  Number of howard steps to perform between maximizations - set howard = 1 if max = 'b'.

3.1.1 Member Function Documentation

3.1.1.1 void parameters::load (const char * *fileName*)

This function is a parameters class method which loads parameter values from a text file for storage in the object. The input file must have 13 lines, each line beginning with a parameter value, followed by a comma and a character string describing the parameter. The order of the parameters must correspond to the order in the parameters class description.

Parameters:

← ***fileName*** Name of file storing parameter values.

Returns:

Void.

The documentation for this class was generated from the following files:

- global.h
- parameters.cpp

Chapter 4

File Documentation

4.1 ar1.cpp File Reference

File containing AR1 function for the VFI problem. `#include "global.h"`

`#include <math.h>`

Functions

- void ar1 (const parameters &param, REAL *Z, REAL *P)

  Function to compute discrete AR1 approximation values and transition matrix.

4.1.1 Detailed Description

Author:

Eric M. Aldrich
ealdrich@ucsc.edu

Version:

1.0

Date:

23 Oct 2012

4.1.2 Function Documentation

4.1.2.1 void ar1 (const parameters & *param*, REAL * *Z*, REAL * *P*)

This function that computes a discrete AR1 approximation and transition matrix using the method of Tauchen (1986).

Parameters:

← ***param*** Object of class parameters.

→ ***Z*** Grid of AR1 values.

→ ***P*** AR1 transition matrix values.

Returns:

Void.

4.2 auxFuncs.h File Reference

Simple auxiliary functions. `#include <iostream>`

`#include <iomanip>`

Functions

- template<class T >
  void printMatrix (const bool colMaj, const int M, const int N, const REAL *X, const int printRows, const int printCols, const int digits)

  Function to print the elements of a matrix.

- template<class T >
  void printVector (const int N, const REAL *X, const int digits)

  Function to print the elements of a Thrust device vector.

4.2.1 Detailed Description

Author:

Eric M. Aldrich
ealdrich@ucsc.edu

Version:

1.0

Date:

18 July 2012

4.2.2 Function Documentation

4.2.2.1 template<class T > void printMatrix (const bool *colMaj*, const int *M*, const int *N*, const REAL * *X*, const int *printRows*, const int *printCols*, const int *digits*) [inline]

This functions prints a subset of the elements of a matrix to the screen.

Parameters:

← ***colMaj*** Boolean indicating if the matrix is stored in column-major format.

← ***M*** Number of rows in the data matrix.

← ***N*** Number of columns in the data matrix.

← ***X*** Array of matrix values.

← ***printRows*** Number of rows to print.

← ***printCols*** Number of columns to print.

← ***precision*** Number of significant digits to print.

Returns:

Void.

4.2.2.2 template<class T > void printVector (const int *N*, const REAL * *X*, const int *digits*) [inline]

This functions prints a subset of the elements of a vector to the screen.

Parameters:

← ***N*** Number of elements in the data matrix.

← ***X*** Array of vector values.

← ***precision*** Number of significant digits to print.

Returns:

Void.

4.3 global.h File Reference

Global header file.

Classes

- class parameters

  Object to store parameter values for VFI problem.

Typedefs

- typedef double **REAL**

Functions

- double curr_second (void)

  Basic timer function.

- void ar1 (const parameters &param, REAL *Z, REAL *P)

  Function to compute discrete AR1 approximation values and transition matrix.

- void kGrid (const parameters &param, const REAL *Z, REAL *K)

  Function to compute the values of an equally spaced capital grid.

- void vfInit (const parameters &param, const REAL *Z, REAL *V)

  Function to initialize value function.

Variables

- const float **singletype**
- const double **doubletype**
- const REAL **realtype**

4.3.1 Detailed Description

Author:

Eric M. Aldrich
ealdrich@ucsc.edu

Version:

1.0

Date:

23 Oct 2012

4.3.2 Function Documentation

4.3.2.1 void ar1 (const parameters & *param*, REAL * *Z*, REAL * *P*)

This function that computes a discrete AR1 approximation and transition matrix using the method of Tauchen (1986).

Parameters:

← ***param*** Object of class parameters.

→ ***Z*** Grid of AR1 values.

→ ***P*** AR1 transition matrix values.

Returns:

Void.

4.3.2.2 curr_second (void)

Returns:

Double precision value representing time.

4.3.2.3 void kGrid (const parameters & *param*, const REAL * *Z*, REAL * *K*)

This function computes an equally spaced capital grid. The upper and lower bounds are the deterministic steady-state values of capital at the highest and lowest values of the TFP process (respectively), scaled by 0.95 and 1.05 (respectively).

Parameters:

← ***param*** Object of class parameters.

← ***Z*** Grid of TFP values.

→ ***K*** Grid of capital values.

Returns:

Void.

4.3.2.4 void vfInit (const parameters & *param*, const REAL * *Z*, REAL * *V*)

This function initializes the value function at the deterministic steady state values for each level of TFP: conditional on a TFP level, the deterministic steady-state value of capital is computed, as well as the associated value function value.

Parameters:

← ***param*** Object of class parameters.

← ***Z*** Grid of TFP values.

→ ***V*** Matrix of value function values.

Returns:

Void.

4.4 kGrid.cpp File Reference

File containing function to create capital grid. `#include "global.h"`

`#include <math.h>`

Functions

- void kGrid (const parameters &param, const REAL *Z, REAL *K)

  Function to compute the values of an equally spaced capital grid.

4.4.1 Detailed Description

Author:

Eric M. Aldrich
ealdrich@ucsc.edu

Version:

1.0

Date:

23 Oct 2012

4.4.2 Function Documentation

4.4.2.1 void kGrid (const parameters & *param*, const REAL * *Z*, REAL * *K*)

This function computes an equally spaced capital grid. The upper and lower bounds are the deterministic steady-state values of capital at the highest and lowest values of the TFP process (respectively), scaled by 0.95 and 1.05 (respectively).

Parameters:

← ***param*** Object of class parameters.

← ***Z*** Grid of TFP values.

→ ***K*** Grid of capital values.

Returns:

Void.

4.5 parameters.cpp File Reference

File containing parameters class method for loading VFI parameter values. `#include "global.h"`

`#include <stdlib.h>`

`#include <vector>`

`#include <fstream>`

4.5.1 Detailed Description

Author:

Eric M. Aldrich
ealdrich@ucsc.edu

Version:

1.0

Date:

23 Oct 2012

4.6 timer.cpp File Reference

File containing basic timer function. `#include <stddef.h>`

`#include <sys/time.h>`

Functions

- double curr_second (void)

  Basic timer function.

4.6.1 Detailed Description

Author:

Kyle Spafford

Date:

19 November 2010

Public domain.

4.6.2 Function Documentation

4.6.2.1 double curr_second (void)

Returns:

Double precision value representing time.

4.7 vfInit.cpp File Reference

File containing function to initialize the value function. `#include "global.h"`

`#include <math.h>`

Functions

- void vfInit (const parameters &param, const REAL *Z, REAL *V)

  Function to initialize value function.

4.7.1 Detailed Description

Author:

Eric M. Aldrich
ealdrich@ucsc.edu

Version:

1.0

Date:

23 Oct 2012

4.7.2 Function Documentation

4.7.2.1 void vfInit (const parameters & *param*, const REAL * *Z*, REAL * *V*)

This function initializes the value function at the deterministic steady state values for each level of TFP: conditional on a TFP level, the deterministic steady-state value of capital is computed, as well as the associated value function value.

Parameters:

← ***param*** Object of class parameters.

← ***Z*** Grid of TFP values.

→ ***V*** Matrix of value function values.

Returns:

Void.