Matlab Toolbox Heterogeneous Agents Dynamic Programming

Fan Wang

2020-07-01

Contents

Preface

This is a work-in-progress Matlab package consisting of functions that facilitate Dynamic Programming and Related Tasks. Materials gathered from various projects in which Matlab code is used. Some of the solutions/algorithms are research outputs developed for specific research papers, other algorithms and methods are commonly-used. Files are the MEconTools repository. Matlab files are linked below by section with livescript files. Tested with Matlab 2019a (The MathWorks Inc, 2019).

Download and install the Matlab toolbox: MEconTools.mltbx

This bookdown file is a collection of mlx based vignettes for functions that are available from MEconTools. Each Vignette file contains various examples for invoking each function. The goal of this repository is to make it easier to find/re-use codes produced for various projects.

From other repositories: For dynamic borrowing and savings problems, see Dynamic Asset Repository; For code examples, see also R Example Code, Matlab Example Code, and Stata Example Code; For intro stat with R, see Intro Statistics for Undergraduates, and intro Math with Matlab, see Intro Mathematics for Economists. See here for all of Fan's public repositories.

The site is built using Bookdown (Xie, 2020).

Please contact FanWangEcon for issues or problems.

Chapter 1

Savings Dynamic Programming

1.1 FF_VFI_AZ_LOOP Dynamic Savings Problem Loop Common Grid

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ff_vfi_az_loop** from the **MEconTools Package.** This function solves the dynamic programming problem for a (a,z) model. Households can save a, and face AR(1) shock z. The problem is solved over the infinite horizon. This is the looped code, it is slow for larger state-space problems. The code uses common grid, with the same state space and choice space grids.

Links to Four Code:

Four Core Savings/Borrowing Dynamic Programming Solution Functions that are functions in the **MEconTools Package.** :

- Common Choice and States Grid : **ff_vfi_az_loop**, slow should use for testing new models
- Common Choice and States Grid : **ff_vfi_az_vec**, fast good for many purposes
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand :**ff_vfi_az_bisec_loop**, high precision even with small grid
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand : **ff_vfi_az_bisec_vec**, precision and speed

The four sample codes are written for the standard dynamic savings problem with AR(1) shock that is one of the core problems introduced in first sessions of graduate Economics courses. The code can be easily adapted to accomand multiple assets, savings and borrowing, discrete and continuous choice, etc. A large proportion of dynamic economic models are based on the underlying structure of solving a model with endogenous states and exogenous shocks, and that is what the (a,z) model does. In general, one should write looped code first to make sure the economics is correct, then vectorized code can be adopted to increase speed.

1.1.1 Test FF_VFI_AZ_LOOP Defaults

Call the function with defaults. By default, shows the asset policy function summary. Model parameters can be changed by the mp_params.

```
%mp_params
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('fl_crra') = 1.5;
mp_params('fl_beta') = 0.94;
% call function
ff_vfi_az_loop(mp_params);
```

```
Elapsed time is 1.291175 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
          i    idx    ndim    numel    rowN    colN     sum      mean      std     coefvari    min
          -    ---    ----    -----    ----    ----    -----    ------    -----    --------    ---

    ap    1     1      2       700     100      7      16864    24.091    14.08    0.58446      0

xxx TABLE:ap xxxxxxxxxxxxxxxxx
              c1         c2         c3         c4         c5         c6         c7
            -------    -------    -------    -------    -------    -------    ------

    r1            0          0          0          0          0    0.50505    2.0202
    r2            0          0          0    0.50505    0.50505     1.0101    2.5253
    r3      0.50505    0.50505    0.50505    0.50505     1.0101     1.5152    3.0303
    r4       1.0101     1.0101     1.0101     1.0101     1.5152     2.0202    3.5354
    r5       1.5152     1.5152     1.5152     1.5152     2.0202     2.5253    4.0404
    r96      45.455     45.455      45.96      45.96      45.96      46.97    48.485
    r97       45.96      45.96      45.96     46.465     46.465     47.475     48.99
    r98      46.465     46.465     46.465      46.97      46.97      47.98     48.99
    r99       46.97      46.97      46.97     47.475     47.475     48.485    49.495
    r100      47.475     47.475     47.475      47.98      47.98      48.99        50
```

1.1.2 Test FF_VFI_AZ_LOOP Speed Tests

Call the function with different a and z grid size, print out speed:

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_timer') = true;
mp_support('ls_ffcmd') = {};
```

A grid 50, shock grid 5:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 50;
mp_params('it_z_n') = 5;
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 0.223217 seconds.
```

A grid 100, shock grid 7:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 100;
mp_params('it_z_n') = 7;
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 1.284511 seconds.
```

A grid 200, shock grid 9:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 200;
mp_params('it_z_n') = 9;
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 6.325330 seconds.
```

1.1.3 Test FF_VFI_AZ_LOOP Control Outputs

Run the function first without any outputs;

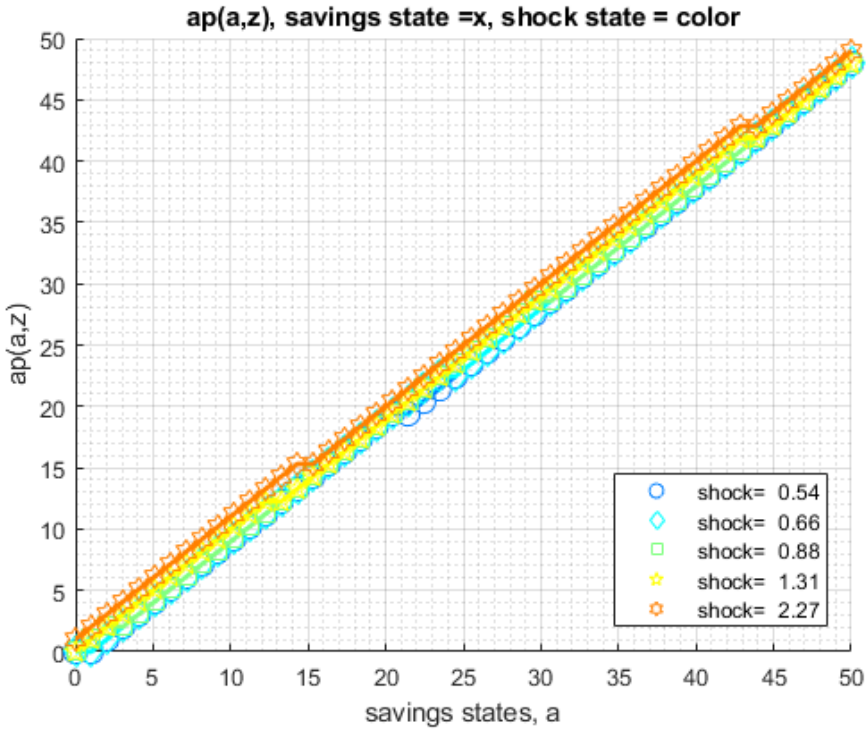
```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 50;
mp_params('it_z_n') = 5;
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_timer') = false;
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
```

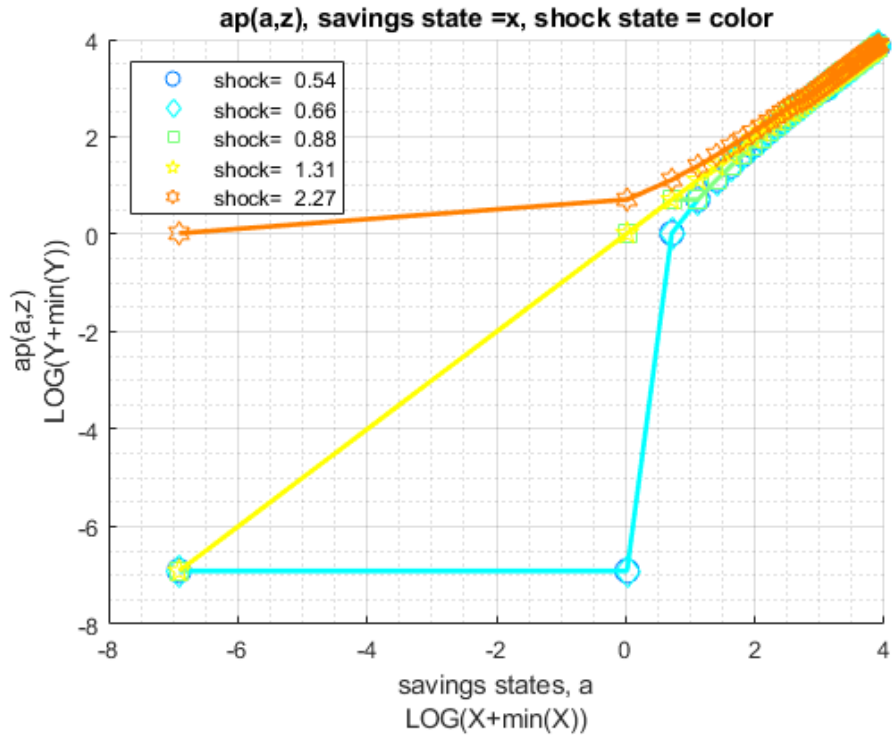
Run the function and show policy function for savings choice. For ls_ffcmd, ls_ffsna, ls_ffgrh, can include these: 'v', 'ap', 'c', 'y', 'coh', 'savefraccoh'. These are value, aprime savings choice, consumption, income, cash on hand, and savings fraction as cash-on-hand.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
% ls_ffcmd: summary print which outcomes
mp_support('ls_ffcmd') = {};
% ls_ffsna: detail print which outcomes
mp_support('ls_ffsna') = {'ap'};
% ls_ffgrh: graphical print which outcomes
mp_support('ls_ffgrh') = {'ap'};
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 0.313830 seconds.
xxx  ff_vfi_az_vec, outcome=ap  xxxxxxxxxxxxxxxxxxxxxxxxxx
```

group	a	mean_z_0_54195	mean_z_0_66401	mean_z_0_88162	mean_z_1_3095	mean_z
1	0	0	0	0	0	1.0
2	1.0204	0	0	1.0204	1.0204	2.0
3	2.0408	1.0204	1.0204	2.0408	2.0408	3.0
4	3.0612	2.0408	2.0408	2.0408	3.0612	4.0
5	4.0816	3.0612	3.0612	3.0612	4.0816	5.
6	5.102	4.0816	4.0816	4.0816	5.102	6.1
7	6.1224	5.102	5.102	5.102	6.1224	7.1
8	7.1429	6.1224	6.1224	6.1224	7.1429	8.1
9	8.1633	7.1429	7.1429	7.1429	8.1633	9.1
10	9.1837	8.1633	8.1633	8.1633	9.1837	10.
11	10.204	9.1837	9.1837	9.1837	10.204	11.
12	11.224	10.204	10.204	10.204	11.224	12.
13	12.245	11.224	11.224	11.224	12.245	13.
14	13.265	12.245	12.245	12.245	12.245	14.
15	14.286	13.265	13.265	13.265	13.265	15.
16	15.306	14.286	14.286	14.286	14.286	15.
17	16.327	15.306	15.306	15.306	15.306	16.
18	17.347	16.327	16.327	16.327	16.327	17.
19	18.367	17.347	17.347	17.347	17.347	18.
20	19.388	18.367	18.367	18.367	18.367	19.
21	20.408	19.388	19.388	19.388	19.388	20.
22	21.429	19.388	20.408	20.408	20.408	21.
23	22.449	20.408	21.429	21.429	21.429	22.
24	23.469	21.429	22.449	22.449	22.449	23.
25	24.49	22.449	22.449	23.469	23.469	24
26	25.51	23.469	23.469	24.49	24.49	25
27	26.531	24.49	24.49	25.51	25.51	26.
28	27.551	25.51	25.51	26.531	26.531	27.

29	28.571	26.531	26.531	27.551	27.551	28.
30	29.592	27.551	27.551	28.571	28.571	29.
31	30.612	28.571	28.571	28.571	29.592	30.
32	31.633	29.592	29.592	29.592	30.612	31.
33	32.653	30.612	30.612	30.612	31.633	32.
34	33.673	31.633	31.633	31.633	32.653	33.
35	34.694	32.653	32.653	32.653	33.673	34.
36	35.714	33.673	33.673	33.673	34.694	35.
37	36.735	34.694	34.694	34.694	35.714	36.
38	37.755	35.714	35.714	35.714	36.735	37.
39	38.776	36.735	36.735	36.735	37.755	38.
40	39.796	37.755	37.755	37.755	38.776	39.
41	40.816	38.776	38.776	38.776	39.796	40.
42	41.837	39.796	39.796	39.796	40.816	41.
43	42.857	40.816	40.816	40.816	41.837	42.
44	43.878	41.837	41.837	41.837	41.837	42.
45	44.898	42.857	42.857	42.857	42.857	43.
46	45.918	43.878	43.878	43.878	43.878	44.
47	46.939	44.898	44.898	44.898	44.898	45.
48	47.959	45.918	45.918	45.918	45.918	46.
49	48.98	46.939	46.939	46.939	46.939	47.
50	50	47.959	47.959	47.959	47.959	48





Run the function and show summaries for savings and fraction of coh saved:

```
mp_params('it_a_n') = 100;
mp_params('it_z_n') = 9;
mp_support('ls_ffcmd') = {'ap', 'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
mp_support('bl_vfi_store_all') = true; % store c(a,z), y(a,z)
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 1.867278 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coef
	-	---	----	-----	----	----	------	-------	-------	----
ap	1	1	2	900	100	9	21825	24.25	14.089	0.
savefraccoh	2	2	2	900	100	9	752.38	0.83597	0.13497	0.16

```
xxx TABLE:ap xxxxxxxxxxxxxxxxxx
```

	c1	c2	c3	c4	c5	c6	c7	c8	
	-------	-------	-------	-------	-------	-------	-------	------	-
r1	0	0	0	0	0	0	0.50505	1.5152	3
r2	0	0	0	0	0.50505	0.50505	1.0101	1.5152	3
r3	0.50505	0.50505	0.50505	0.50505	0.50505	1.0101	1.5152	2.0202	4
r4	1.0101	1.0101	1.0101	1.0101	1.0101	1.5152	2.0202	2.5253	4
r5	1.5152	1.5152	1.5152	1.5152	1.5152	2.0202	2.5253	3.0303	5
r96	45.455	45.455	45.455	45.96	45.96	45.96	46.465	47.475	4
r97	45.96	45.96	45.96	46.465	46.465	46.465	46.97	47.98	4
r98	46.465	46.465	46.465	46.465	46.97	46.97	47.475	48.485	
r99	46.97	46.97	46.97	46.97	47.475	47.475	47.98	48.99	

```
    r100     47.475     47.475     47.475     47.475      47.98      47.98     48.485     49.495

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1         c2         c3         c4         c5         c6         c7         c8
            -------    -------    -------    -------    -------    -------    -------    -------

    r1            0          0          0          0          0          0    0.24587    0.48182
    r2            0          0          0          0     0.3075    0.25444    0.39276    0.41371
    r3      0.30679    0.29486    0.27938    0.25939     0.2338    0.40362    0.49043     0.4833
    r4       0.4668    0.45285    0.43438    0.40981    0.37721    0.50166    0.56006    0.53755
    r5      0.56502    0.55132    0.53293    0.50802    0.47415    0.57101    0.61221    0.58103
    r96     0.91292     0.9117    0.90997    0.91752    0.91364    0.90746    0.90692    0.90732
    r97     0.91357    0.91236    0.91064    0.91812    0.91427    0.90815    0.90761    0.90799
    r98      0.9142      0.913     0.9113    0.90882    0.91489    0.90882    0.90828    0.90865
    r99     0.91482    0.91363    0.91195    0.90949    0.91549    0.90949    0.90894    0.90929
    r100    0.91543    0.91425    0.91258    0.91014    0.91609    0.91013    0.90959    0.90992
```

1.1.4 Test FF_VFI_AZ_LOOP Change Interest Rate and Discount

Show only save fraction of cash on hand:

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 50;
mp_params('it_z_n') = 5;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Solve the model with several different interest rates and discount factor:

```
% Lower Savings Incentives
mp_params('fl_beta') = 0.80;
mp_params('fl_r') = 0.01;
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 0.113265 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                  i    idx    ndim    numel    rowN    colN     sum       mean       std      coefv
                  -    ---    ----    -----    ----    ----    ------    -------    ------    -----

    savefraccoh   1     1      2       250      50      5      118.68    0.47472    0.2843    0.598

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1         c2         c3         c4         c5
            -------    -------    -------    -------    -------

    r1            0          0          0          0    0.10642
    r2            0          0          0          0     0.1064
    r3            0          0          0          0    0.10629
    r4            0          0          0          0      0.106
    r5            0          0          0          0    0.10543
    r46     0.79096    0.78787    0.78241    0.77191    0.74922
```

```
    r47    0.79553    0.79262    0.78747    0.77755    0.75606
    r48     0.7999    0.79715    0.79229     0.7829    0.76254
    r49    0.80407    0.80147    0.79687    0.78799    0.76868
    r50    0.80805    0.80559    0.80125    0.79284     0.7745

% Higher Savings Incentives
mp_params('fl_beta') = 0.95;
mp_params('fl_r') = 0.04;
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 0.327279 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                   i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                   -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh    1     1      2       250      50      5      160.99    0.64394    0.29947    0.46

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
             c1          c2          c3          c4          c5
            -------    -------    --------    -------    -------

    r1            0          0    0.024103    0.18484    0.40057
    r2            0          0    0.024094     0.1848    0.40051
    r3            0          0    0.024028    0.18446    0.40008
    r4            0          0    0.046583    0.18354    0.39894
    r5            0          0    0.045925    0.24935    0.39672
    r46    0.94526    0.94167     0.93533    0.92312    0.89672
    r47    0.94628    0.94291     0.93696    0.92548    0.90059
    r48    0.94722    0.94405     0.93846    0.92766    0.90418
    r49    0.94808    0.94511     0.93984    0.92966    0.90749
    r50    0.94888    0.94608     0.94111    0.93151    0.91056
```

1.1.5 Test FF_VFI_AZ_LOOP Changing Risk Aversion

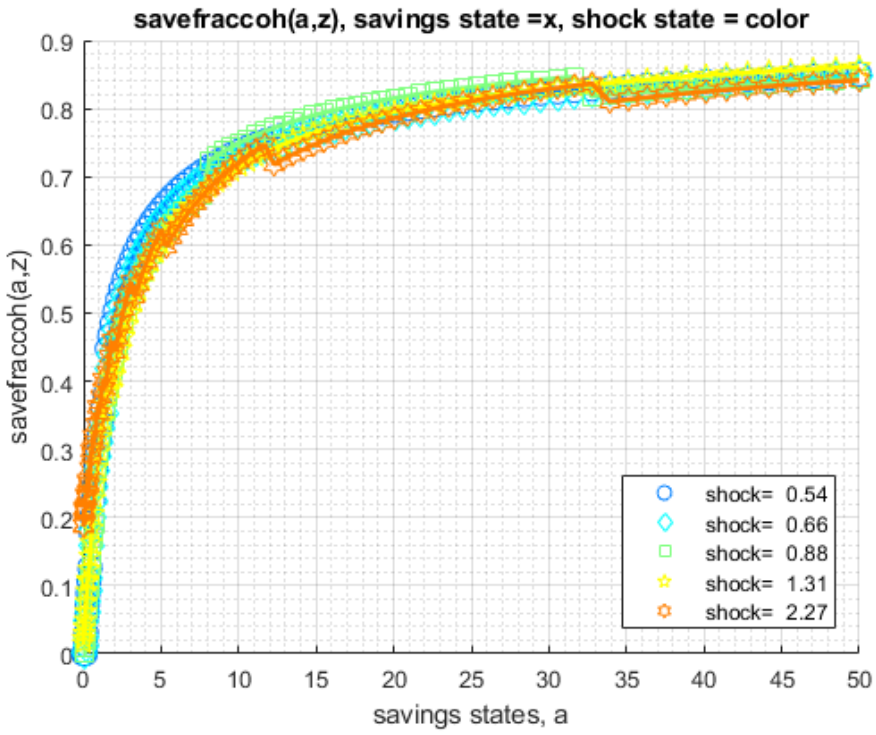
Here, again, show fraction of coh saved in summary tabular form, but also show it graphically.

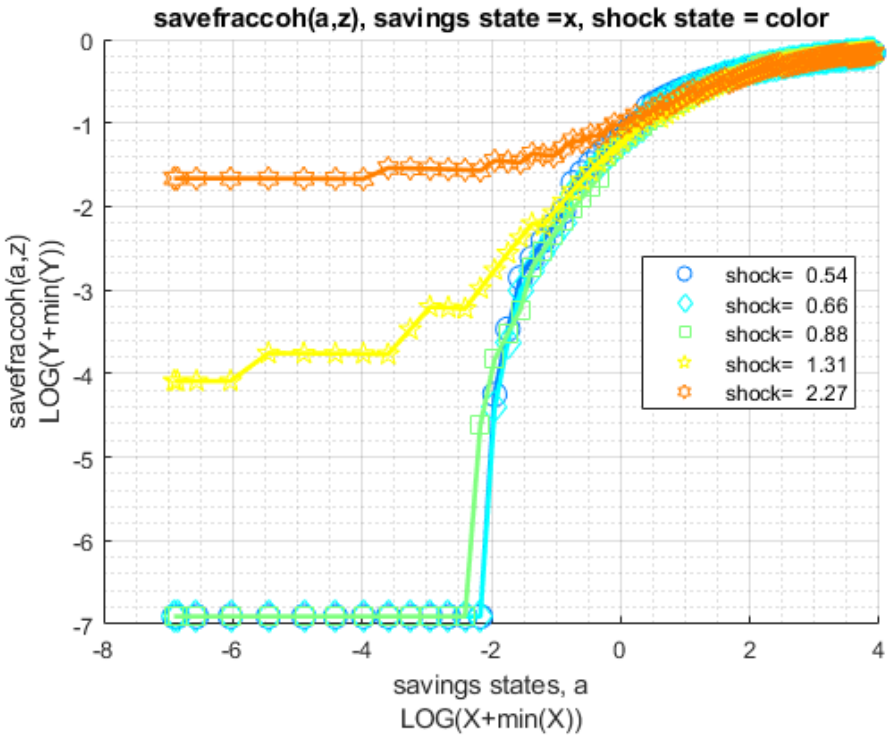
```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {'savefraccoh'};
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 100;
mp_params('it_z_n') = 5;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Solve the model with different risk aversion levels, higher preferences for risk:

```
% Lower Risk Aversion
mp_params('fl_crra') = 0.5;
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 0.581794 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

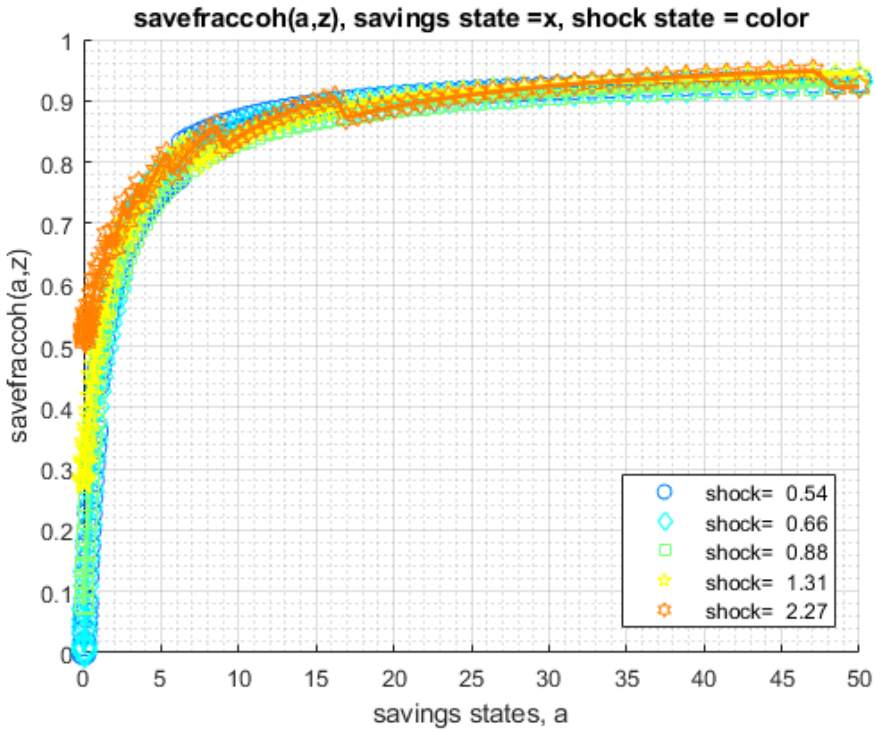
```
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i     idx     ndim     numel     rowN     colN      sum       mean        std       coef
                    -     ---     ----     -----     ----     ----     ------     -------     -------     ----

    savefraccoh     1      1       2        500      100       5       268.82     0.53764     0.29852     0.55

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5
            -------     -------     -------     --------     -------

    r1            0           0           0     0.015741     0.18847
    r2            0           0           0      0.01574     0.18847
    r3            0           0           0     0.015737     0.18844
    r4            0           0           0     0.015728     0.18838
    r5            0           0           0     0.022367     0.18825
    r96     0.84455     0.84169     0.83664      0.85445     0.83255
    r97     0.84611     0.84333     0.83842      0.85626     0.83496
    r98     0.84763     0.84493     0.84016      0.85803     0.83729
    r99     0.84911     0.84648     0.84185      0.85974     0.83956
    r100    0.85055       0.848     0.84349      0.86141     0.84176
```

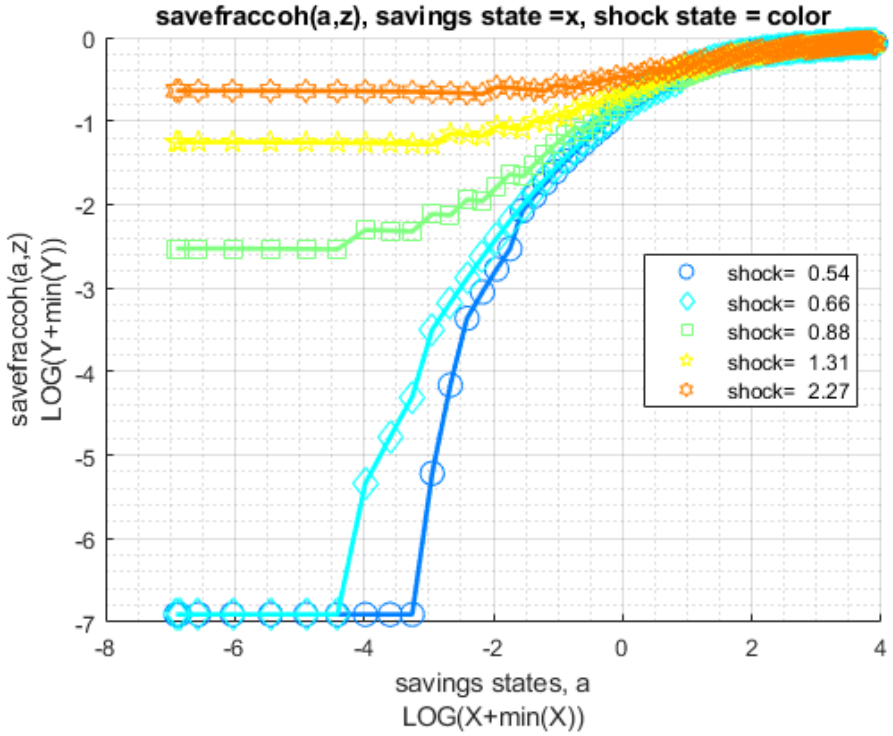




When risk aversion increases, at every state-space point, the household wants to save more.

```
% Higher Risk Aversion
mp_params('fl_crra') = 5;
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 0.937495 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i    idx    ndim    numel    rowN    colN     sum       mean        std       coef
                    -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh     1      1       2      500     100       5     335.64    0.67129    0.28688    0.42

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5
            -------    -------    --------    -------    -------

    r1            0          0    0.078907    0.28472    0.52731
    r2            0          0    0.078904    0.28471     0.5273
    r3            0          0    0.078878    0.28465    0.52723
    r4            0          0    0.078808    0.28448    0.52705
    r5            0          0    0.078672    0.28415    0.52669
    r96     0.93086    0.92771     0.92215    0.94079    0.94593
    r97     0.93161    0.92855     0.92315    0.94183    0.94739
    r98     0.93233    0.92936     0.92411    0.94283     0.9488
    r99     0.93303    0.93015     0.92505    0.94379    0.92164
    r100    0.93371    0.93091     0.92595    0.94471    0.92317
```





1.1.6 Test FF_VFI_AZ_LOOP with Higher Uncertainty

Increase the standard deviation of the Shock.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
```

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 100;
mp_params('it_z_n') = 5;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Lower standard deviation of shock:

```
% Lower Risk Aversion
mp_params('fl_shk_std') = 0.10;
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 0.957457 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i    idx    ndim    numel    rowN    colN     sum      mean       std      coefva
                    -    ---    ----    -----    ----    ----    -----    ------    -------    ------

    savefraccoh     1     1      2       500     100      5      294.1    0.5882    0.32083    0.5454

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5
            -------    -------    -------    --------    -------

    r1            0          0          0    0.034556    0.11424
    r2            0          0          0    0.034555    0.11424
    r3            0          0          0    0.034546    0.11422
    r4            0          0          0    0.034523    0.11416
    r5            0          0          0    0.034478    0.11404
    r96     0.89673    0.89421    0.91986     0.91499    0.90808
    r97     0.89789    0.89545    0.92093      0.9162    0.90948
    r98     0.89903    0.89665    0.92196     0.91737    0.91084
    r99     0.90013    0.89782    0.92295      0.9185    0.91215
    r100    0.90119    0.89896    0.92392     0.91959    0.91342
```

Higher shock standard deviation: low shock high asset save more, high shock more asset save less, high shock low asset save more:

```
% Higher Risk Aversion
mp_params('fl_shk_std') = 0.40;
ff_vfi_az_loop(mp_params, mp_support);

Elapsed time is 0.923630 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                    -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh     1     1      2       500     100      5      350.37    0.70073    0.26741    0.38

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5
            -------    -------    --------    -------    -------

    r1            0          0    0.030722    0.36969    0.77072
    r2            0          0     0.03072    0.36967    0.77071
```

r3	0	0	0.0307	0.36958	0.77068
r4	0	0	0.030646	0.36933	0.7706
r5	0	0	0.030543	0.36885	0.77044
r96	0.90975	0.90819	0.9038	0.91513	0.88687
r97	0.91053	0.90902	0.90476	0.91633	0.89076
r98	0.91129	0.90982	0.90569	0.9175	0.86794
r99	0.91204	0.91061	0.9066	0.91862	0.84583
r100	0.91276	0.91138	0.90748	0.91971	0.82439

1.2 FF_VFI_AZ_VEC Dynamic Savings Problem Vectorized Common Grid

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ff_vfi_az_vec** from the **MEconTools Package.** This function solves (vectorized) the dynamic programming problem for a (a,z) model. Households can save a, and face AR(1) shock z. The problem is solved over the infinite horizon.

The code uses common grid, with the same state space and choice space grids. **ff_vfi_az_bisec_vec** from the **MEconTools Package** solves the same problem but using continuous exact percentage asset choices, which is more precise than the solution here, and perhaps a little bit slower.

This is the vectorized code, its speed is much faster than the looped code. The function is designed to have small memory footprint and requires low computing resources, yet is fast.

Links to Four Code:

Four Core Savings/Borrowing Dynamic Programming Solution Functions that are functions in the **MEconTools Package.** :

- Common Choice and States Grid : **ff_vfi_az_loop**, slow should use for testing new models
- Common Choice and States Grid : **ff_vfi_az_vec**, fast good for many purposes
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand :**ff_vfi_az_bisec_loop**, high precision even with small grid
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand : **ff_vfi_az_bisec_vec**, precision and speed

1.2.1 Test FF_VFI_AZ_VEC Defaults

Call the function with defaults. By default, shows the asset policy function summary. Model parameters can be changed by the mp_params.

```
%mp_params
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('fl_crra') = 1.5;
mp_params('fl_beta') = 0.94;
ff_vfi_az_vec(mp_params);

Elapsed time is 0.128918 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
          i    idx    ndim    numel    rowN    colN     sum      mean      std     coefvari     min
          -    ---    ----    -----    ----    ----    -----    ------    -----    --------    ---

    ap    1     1      2       700     100      7      16864    24.091    14.08    0.58446      0
```

```
xxx TABLE:ap xxxxxxxxxxxxxxxxx
```

	c1	c2	c3	c4	c5	c6	c7
r1	0	0	0	0	0	0.50505	2.0202
r2	0	0	0	0.50505	0.50505	1.0101	2.5253
r3	0.50505	0.50505	0.50505	0.50505	1.0101	1.5152	3.0303
r4	1.0101	1.0101	1.0101	1.0101	1.5152	2.0202	3.5354
r5	1.5152	1.5152	1.5152	1.5152	2.0202	2.5253	4.0404
r96	45.455	45.455	45.96	45.96	45.96	46.97	48.485
r97	45.96	45.96	45.96	46.465	46.465	47.475	48.99
r98	46.465	46.465	46.465	46.97	46.97	47.98	48.99
r99	46.97	46.97	46.97	47.475	47.475	48.485	49.495
r100	47.475	47.475	47.475	47.98	47.98	48.99	50

1.2.2 Test FF_VFI_AZ_BISEC_VEC Speed Tests

Call the function with different a and z grid size, print out speed:

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_timer') = true;
mp_support('ls_ffcmd') = {};
```

A grid 200, shock grid 9:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 200;
mp_params('it_z_n') = 9;
ff_vfi_az_vec(mp_params, mp_support);

Elapsed time is 0.220867 seconds.
```

A grid 750, shock grid 15:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 750;
mp_params('it_z_n') = 15;
ff_vfi_az_vec(mp_params, mp_support);

Elapsed time is 3.573648 seconds.
```

A grid 600, shock grid 45:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 600;
mp_params('it_z_n') = 45;
ff_vfi_az_vec(mp_params, mp_support);

Elapsed time is 8.398580 seconds.
```

1.2.3 Test FF_VFI_AZ_VEC Control Outputs

Run the function first without any outputs;

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 50;
mp_params('it_z_n') = 5;
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_timer') = false;
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
```

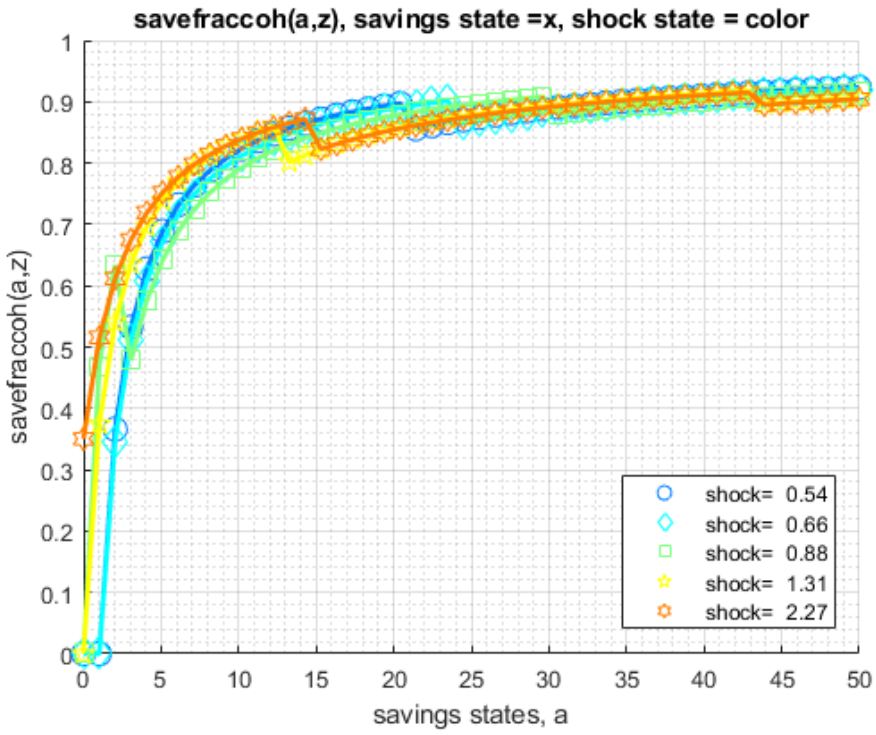
Run the function and show policy function for savings choice. For ls_ffcmd, ls_ffsna, ls_ffgrh, can include these: 'v', 'ap', 'c', 'y', 'coh', 'savefraccoh'. These are value, aprime savings choice, consumption, income, cash on hand, and savings fraction as cash-on-hand.

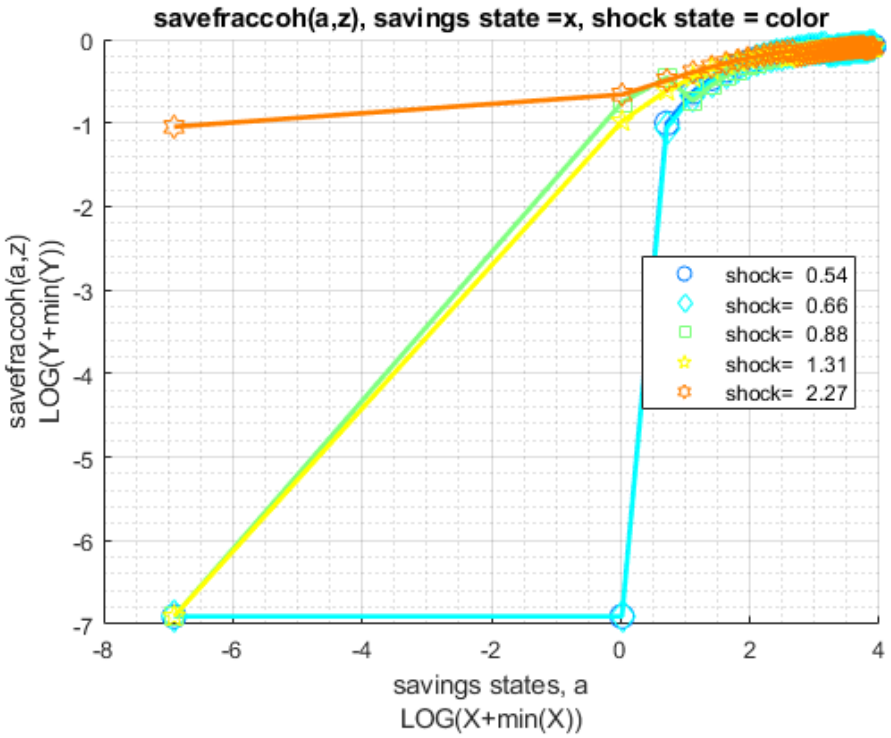
```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
% ls_ffcmd: summary print which outcomes
mp_support('ls_ffcmd') = {};
% ls_ffsna: detail print which outcomes
mp_support('ls_ffsna') = {'savefraccoh'};
% ls_ffgrh: graphical print which outcomes
mp_support('ls_ffgrh') = {'savefraccoh'};
ff_vfi_az_vec(mp_params, mp_support);
```

```
Elapsed time is 0.014484 seconds.
xxx  ff_vfi_az_vec, outcome=savefraccoh  xxxxxxxxxxxxxxxxxxxxxxxxxxx
```

group	a	mean_z_0_54195	mean_z_0_66401	mean_z_0_88162	mean_z_1_3095	mean_z
1	0	0	0	0	0	0.
2	1.0204	0	0	0.46928	0.37487	0.5
3	2.0408	0.36632	0.34687	0.63373	0.54163	0.6
4	3.0612	0.53265	0.51178	0.47837	0.63592	0.6
5	4.0816	0.62764	0.60816	0.57627	0.69655	0.7
6	5.102	0.68908	0.67137	0.64196	0.73882	0.7
7	6.1224	0.73208	0.71603	0.68909	0.76996	0.7
8	7.1429	0.76386	0.74926	0.72456	0.79387	0.7
9	8.1633	0.7883	0.77494	0.75221	0.81279	0.8
10	9.1837	0.80769	0.79539	0.77438	0.82815	0.8
11	10.204	0.82343	0.81206	0.79254	0.84086	0.
12	11.224	0.83648	0.82591	0.8077	0.85155	0.8
13	12.245	0.84747	0.83759	0.82053	0.86067	0.8
14	13.265	0.85685	0.84758	0.83155	0.80173	0.8
15	14.286	0.86495	0.85622	0.8411	0.81288	0.8
16	15.306	0.87201	0.86377	0.84947	0.82268	0.8
17	16.327	0.87823	0.87043	0.85685	0.83137	0.8
18	17.347	0.88374	0.87633	0.86342	0.83912	0.8
19	18.367	0.88866	0.88161	0.8693	0.84608	0.8
20	19.388	0.89309	0.88635	0.8746	0.85237	0.8
21	20.408	0.89708	0.89064	0.87939	0.85807	0.8
22	21.429	0.85567	0.89454	0.88375	0.86327	0.8
23	22.449	0.86096	0.89809	0.88773	0.86803	0.8
24	23.469	0.86581	0.90135	0.89138	0.87241	0.8
25	24.49	0.87026	0.86502	0.89474	0.87644	0.8
26	25.51	0.87436	0.8693	0.89784	0.88017	0.8
27	26.531	0.87816	0.87327	0.90071	0.88362	0.8
28	27.551	0.88168	0.87695	0.90338	0.88684	0.8
29	28.571	0.88496	0.88037	0.90586	0.88984	0.8
30	29.592	0.88802	0.88357	0.90818	0.89264	0.8
31	30.612	0.89087	0.88655	0.87896	0.89527	0.8
32	31.633	0.89355	0.88935	0.88197	0.89773	0.8
33	32.653	0.89606	0.89198	0.8848	0.90005	0.8
34	33.673	0.89843	0.89446	0.88747	0.90223	0.8
35	34.694	0.90065	0.89679	0.88998	0.90429	0.9
36	35.714	0.90275	0.89899	0.89235	0.90624	0.9
37	36.735	0.90474	0.90107	0.8946	0.90809	0.9
38	37.755	0.90662	0.90304	0.89673	0.90984	0.9
39	38.776	0.90841	0.90491	0.89874	0.9115	0.9

40	39.796	0.9101	0.90669	0.90066	0.91308	0.9
41	40.816	0.91171	0.90838	0.90249	0.91458	0.9
42	41.837	0.91325	0.90998	0.90422	0.91601	0.9
43	42.857	0.91471	0.91152	0.90588	0.91738	0.9
44	43.878	0.9161	0.91298	0.90746	0.89681	0.8
45	44.898	0.91743	0.91438	0.90897	0.89854	0.8
46	45.918	0.91871	0.91571	0.91042	0.90019	0.8
47	46.939	0.91993	0.91699	0.91181	0.90178	0.8
48	47.959	0.9211	0.91822	0.91313	0.9033	0.9
49	48.98	0.92222	0.91939	0.91441	0.90475	0.9
50	50	0.92329	0.92052	0.91563	0.90615	0.9





Run the function and show summaries for savings and fraction of coh saved:

```
mp_params('it_a_n') = 100;
mp_params('it_z_n') = 9;
mp_support('ls_ffcmd') = {'ap', 'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
mp_support('bl_vfi_store_all') = true; % store c(a,z), y(a,z)
ff_vfi_az_vec(mp_params, mp_support);

Elapsed time is 0.127807 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coef
	-	---	----	-----	----	----	------	-------	-------	----
ap	1	1	2	900	100	9	21825	24.25	14.089	0.
savefraccoh	2	2	2	900	100	9	752.38	0.83597	0.13497	0.16

```
xxx TABLE:ap xxxxxxxxxxxxxxxxxx
```

	c1	c2	c3	c4	c5	c6	c7	c8	
	-------	-------	-------	-------	-------	-------	-------	------	-
r1	0	0	0	0	0	0	0.50505	1.5152	3
r2	0	0	0	0	0.50505	0.50505	1.0101	1.5152	3
r3	0.50505	0.50505	0.50505	0.50505	0.50505	1.0101	1.5152	2.0202	4
r4	1.0101	1.0101	1.0101	1.0101	1.0101	1.5152	2.0202	2.5253	4
r5	1.5152	1.5152	1.5152	1.5152	1.5152	2.0202	2.5253	3.0303	5
r96	45.455	45.455	45.455	45.96	45.96	45.96	46.465	47.475	4
r97	45.96	45.96	45.96	46.465	46.465	46.465	46.97	47.98	4
r98	46.465	46.465	46.465	46.465	46.97	46.97	47.475	48.485	
r99	46.97	46.97	46.97	46.97	47.475	47.475	47.98	48.99	

```
    r100     47.475     47.475     47.475     47.475      47.98      47.98     48.485     49.495

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1         c2         c3         c4         c5         c6         c7         c8
            -------    -------    -------    -------    -------    -------    -------    -------

    r1            0          0          0          0          0          0    0.24587    0.48182
    r2            0          0          0          0     0.3075    0.25444    0.39276    0.41371
    r3      0.30679    0.29486    0.27938    0.25939     0.2338    0.40362    0.49043     0.4833
    r4       0.4668    0.45285    0.43438    0.40981    0.37721    0.50166    0.56006    0.53755
    r5      0.56502    0.55132    0.53293    0.50802    0.47415    0.57101    0.61221    0.58103
    r96     0.91292     0.9117    0.90997    0.91752    0.91364    0.90746    0.90692    0.90732
    r97     0.91357    0.91236    0.91064    0.91812    0.91427    0.90815    0.90761    0.90799
    r98      0.9142      0.913     0.9113    0.90882    0.91489    0.90882    0.90828    0.90865
    r99     0.91482    0.91363    0.91195    0.90949    0.91549    0.90949    0.90894    0.90929
    r100    0.91543    0.91425    0.91258    0.91014    0.91609    0.91013    0.90959    0.90992
```

1.2.4 Test FF_VFI_AZ_VEC Change Interest Rate and Discount

Show only save fraction of cash on hand:

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 750;
mp_params('it_z_n') = 9;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Solve the model with several different interest rates and discount factor:

```
% Lower Savings Incentives
mp_params('fl_beta') = 0.80;
mp_params('fl_r') = 0.01;
ff_vfi_az_vec(mp_params, mp_support);

Elapsed time is 0.771613 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                  i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                  -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh   1     1      2       6750     750      9      3318.8    0.49167    0.27768    0.56

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1         c2         c3         c4         c5         c6         c7         c8
            -------    -------    -------    -------    -------    -------    --------    -------

    r1            0          0          0          0          0          0    0.023475    0.13289
    r2            0          0          0          0          0          0    0.023475    0.13289
    r3            0          0          0          0          0          0    0.023475    0.13289
    r4            0          0          0          0          0          0    0.023475    0.13289
    r5            0          0          0          0          0          0    0.023475    0.13289
    r746     0.8044    0.80333    0.80182    0.79961    0.79626    0.79093      0.7887     0.7824
```

```
    r747    0.80465    0.80359    0.80209    0.79989    0.79655    0.79124     0.78903    0.78277
    r748    0.80491    0.80385    0.80235    0.80016    0.79683    0.79154     0.78936    0.78315
    r749    0.80517    0.80411    0.80262    0.80043    0.79712    0.79185     0.78969    0.78352
    r750    0.80542    0.80437    0.80288    0.80071     0.7974    0.79215     0.79002    0.78389

% Higher Savings Incentives
mp_params('fl_beta') = 0.95;
mp_params('fl_r') = 0.04;
ff_vfi_az_vec(mp_params, mp_support);

Elapsed time is 2.484993 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                   i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                   -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh    1     1      2      6750     750      9      4491.9    0.66547    0.28771    0.43

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1         c2         c3         c4          c5         c6         c7         c8
            -------    -------    -------    -------    --------    -------    -------    -------

    r1            0          0          0          0    0.031818    0.14726    0.31047    0.48484
    r2            0          0          0          0    0.031818    0.14726    0.31047    0.48484
    r3            0          0          0          0    0.031818    0.14726    0.31047    0.48484
    r4            0          0          0          0    0.031818    0.14726    0.31047    0.48484
    r5            0          0          0          0    0.031818    0.14726    0.31047    0.48484
    r746    0.92742       0.93     0.9283    0.92581     0.92578    0.92349    0.92443    0.91686
    r747     0.9275    0.93007    0.92838     0.9259     0.92588    0.92361    0.92457    0.91706
    r748    0.92757    0.93014    0.92846    0.92599     0.92598    0.92373    0.92472    0.91359
    r749    0.92764    0.93022    0.92854    0.92608     0.92608    0.92384    0.92115    0.91014
    r750    0.92772    0.93029    0.92862    0.92617     0.92618    0.92396     0.9213    0.90671
```

1.2.5 Test FF_VFI_AZ_VEC Changing Risk Aversion

Here, again, show fraction of coh saved in summary tabular form, but also show it graphically.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {'savefraccoh'};
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 750;
mp_params('it_z_n') = 9;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Solve the model with different risk aversion levels, higher preferences for risk:

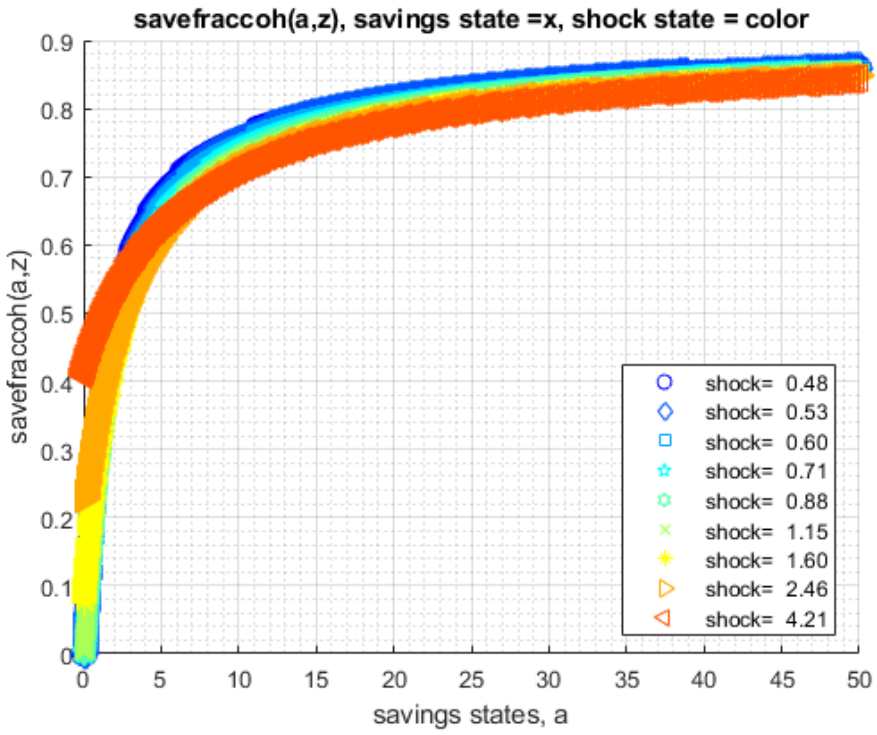
```
% Lower Risk Aversion
mp_params('fl_crra') = 0.5;
ff_vfi_az_vec(mp_params, mp_support);

Elapsed time is 1.991475 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

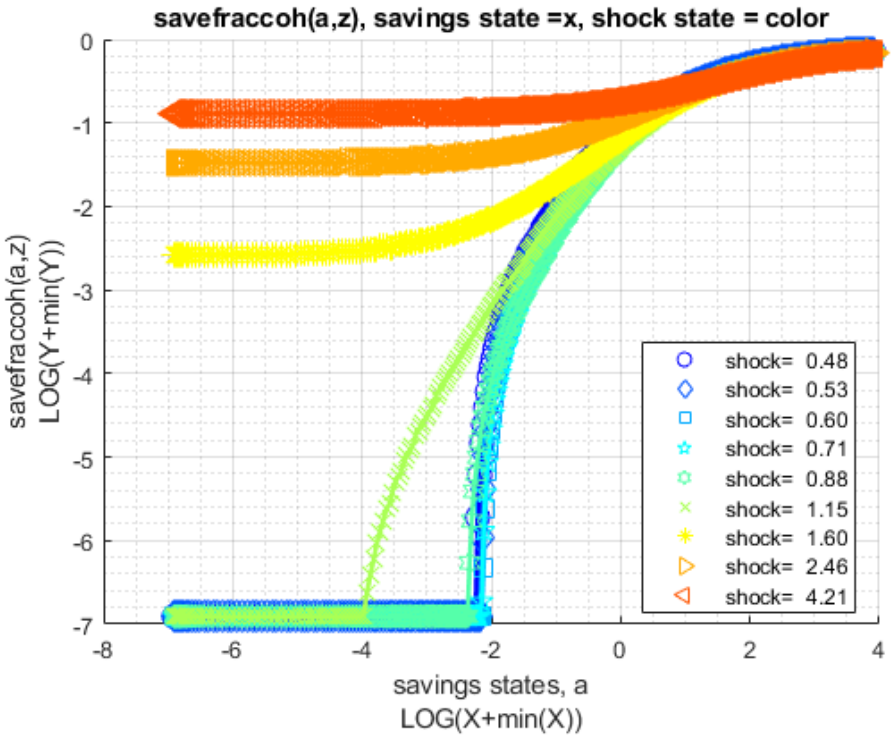
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coefv
savefraccoh	1	1	2	6750	750	9	3735.9	0.55347	0.2897	0.523

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx

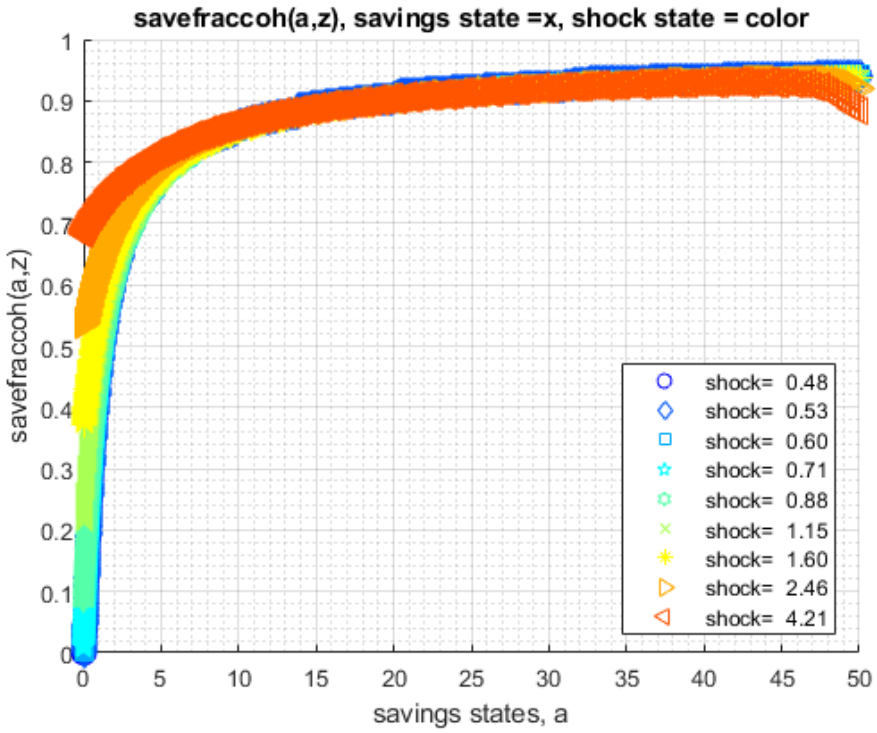
	c1	c2	c3	c4	c5	c6	c7	c8
r1	0	0	0	0	0	0	0.075021	0.22812
r2	0	0	0	0	0	0	0.075021	0.22812
r3	0	0	0	0	0	0	0.075021	0.22812
r4	0	0	0	0	0	0	0.075021	0.22812
r5	0	0	0	0	0	0	0.075021	0.22812
r746	0.85928	0.85816	0.85657	0.85425	0.85428	0.8522	0.84972	0.84635
r747	0.85946	0.85834	0.85676	0.85444	0.85449	0.85242	0.84997	0.84665
r748	0.85963	0.85852	0.85694	0.85464	0.85469	0.85264	0.85021	0.84694
r749	0.85981	0.8587	0.85713	0.85483	0.85489	0.85286	0.85046	0.84723
r750	0.85998	0.85888	0.85731	0.85502	0.85509	0.85307	0.8507	0.84752

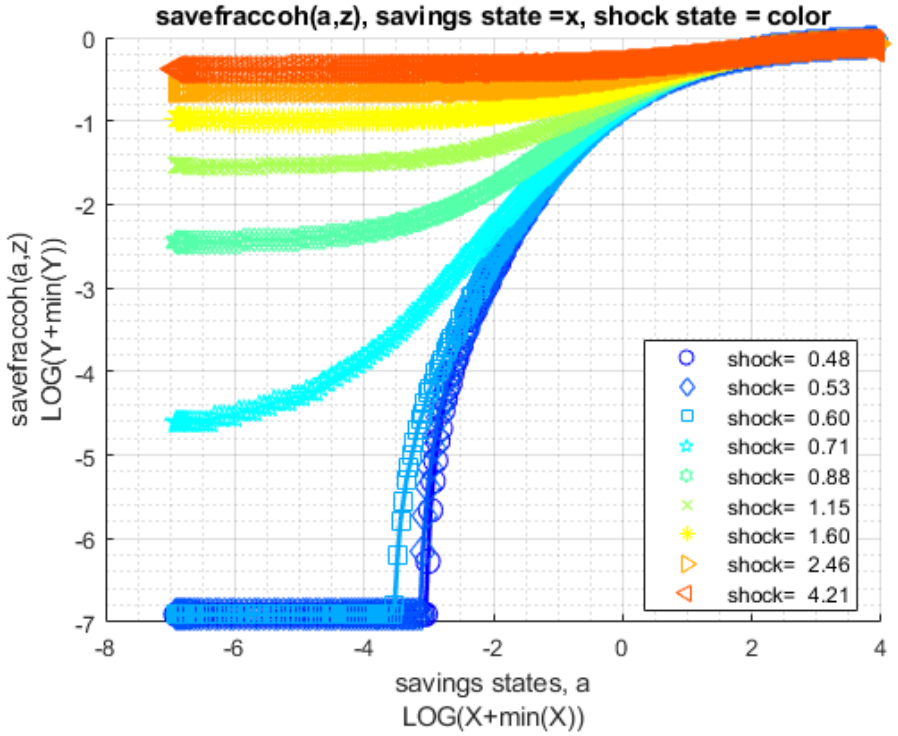




When risk aversion increases, at every state-space point, the household wants to save more.

```
% Higher Risk Aversion
mp_params('fl_crra') = 5;
ff_vfi_az_vec(mp_params, mp_support);

Elapsed time is 2.026442 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i     idx     ndim     numel     rowN     colN      sum       mean       std       coefv
                    -     ---     ----     -----     ----     ----     ------     ------    -------     -----

    savefraccoh     1      1       2        6750      750       9      4639.3     0.6873    0.28204     0.410

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxx
              c1          c2          c3           c4           c5          c6          c7          c8
            -------     -------     -------     ---------     --------    -------     -------     -------

    r1            0           0           0      0.008995     0.085095    0.21314     0.37277     0.53628
    r2            0           0           0      0.008995     0.085095    0.21314     0.37277     0.53628
    r3            0           0           0      0.008995     0.085095    0.21314     0.37277     0.53628
    r4            0           0           0      0.008995     0.085095    0.21314     0.37277     0.53628
    r5            0           0           0     0.0089949     0.085094    0.21314     0.37277     0.53628
    r746    0.94083      0.9396     0.94168       0.93912      0.93904     0.94041     0.93743     0.92949
    r747    0.94091     0.93969     0.94176       0.93921      0.93914     0.93674     0.93758     0.92969
    r748    0.94098     0.93977     0.94184       0.93931      0.93924     0.93686     0.93772     0.92618
    r749    0.94106     0.93985     0.94192        0.9394      0.93934     0.93699     0.93787     0.92269
    r750    0.94113     0.93993       0.942       0.93949      0.93944     0.93711     0.93801     0.91921
```





1.2.6 Test FF_VFI_AZ_VEC with Higher Uncertainty

Increase the standard deviation of the Shock.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
```

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 750;
mp_params('it_z_n') = 9;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Lower standard deviation of shock:

```
% Lower Risk Aversion
mp_params('fl_shk_std') = 0.10;
ff_vfi_az_vec(mp_params, mp_support);

Elapsed time is 2.065989 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i    idx    ndim    numel    rowN    colN    sum      mean        std      coefva
                    -    ---    ----    -----    ----    ----    ----    -------    -------    ------

    savefraccoh     1     1      2       6750     750      9      4026    0.59644    0.31533    0.5286

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5          c6          c7          c8
            -------    -------    -------    -------    -------    --------    --------    -------

    r1            0          0          0          0          0    0.012569    0.062884    0.13754
    r2            0          0          0          0          0    0.012569    0.062884    0.13754
    r3            0          0          0          0          0    0.012569    0.062884    0.13754
    r4            0          0          0          0          0    0.012569    0.062884    0.13754
    r5            0          0          0          0          0    0.012569    0.062884    0.13754
    r746    0.91375    0.91251    0.91101    0.91289    0.91057     0.91138     0.91136    0.91021
    r747    0.91387    0.91264    0.91114    0.91302    0.91072     0.91153     0.91152    0.91039
    r748    0.91399    0.91277    0.91127    0.91315    0.91086     0.91168     0.91168    0.91056
    r749    0.91411    0.91289     0.9114    0.91329      0.911     0.91183     0.91183    0.91073
    r750    0.91423    0.91302    0.91153    0.91342    0.91114     0.91197     0.91199     0.9109
```

Higher shock standard deviation: low shock high asset save more, high shock more asset save less, high shock low asset save more:

```
% Higher Risk Aversion
mp_params('fl_shk_std') = 0.40;
ff_vfi_az_vec(mp_params, mp_support);

Elapsed time is 2.184888 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                    -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh     1     1      2       6750     750      9      4687.4    0.69442    0.27109    0.39

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5          c6          c7          c8
            -------    -------    -------    -------    --------    -------    -------    -------

    r1            0          0          0          0    0.030619    0.24561    0.55369    0.80189
    r2            0          0          0          0    0.030619    0.24561    0.55369    0.80189
```

r3	0	0	0	0	0.030619	0.2456	0.55369	0.80189
r4	0	0	0	0	0.030619	0.2456	0.55369	0.80189
r5	0	0	0	0	0.030618	0.2456	0.55369	0.80189
r746	0.93365	0.93335	0.9328	0.93173	0.92941	0.92713	0.92079	0.8402
r747	0.93371	0.93341	0.93286	0.9318	0.92949	0.92723	0.92095	0.83734
r748	0.93378	0.93348	0.93293	0.93187	0.92957	0.92733	0.92111	0.83449
r749	0.93384	0.93354	0.933	0.93194	0.92965	0.92743	0.92127	0.83166
r750	0.9339	0.9336	0.93306	0.93201	0.92973	0.92753	0.92143	0.82883

1.3 FF_VFI_AZ_BISEC_LOOP Dynamic Savings Problem Loop Continuous Choice

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function:**ff_vfi_az_bisec_loop**from the **MEconTools Package.** This function solves the dynamic programming problem for a (a,z) model. Households can save a, and face AR(1) shock z. The problem is solved over the infinite horizon. This is the looped code, it is slow for larger state-space problems. The code uses continuous choices, solved with bisection. The state-space is on a grid, but choice grids are in terms of percentage of resources to save and solved exactly.

Links to Four Code:

Four Core Savings/Borrowing Dynamic Programming Solution Functions that are functions in the **MEconTools Package.** :

- Common Choice and States Grid : **ff_vfi_az_loop**, slow should use for testing new models
- Common Choice and States Grid : **ff_vfi_az_vec**, fast good for many purposes
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand :**ff_vfi_az_bisec_loop**, high precision even with small grid
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand : **ff_vfi_az_bisec_vec**, precision and speed

1.3.1 Test FF_VFI_AZ_BISEC_LOOP Defaults

Call the function with defaults. By default, shows the asset policy function summary. Model parameters can be changed by the mp_params.

```
%mp_params
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('fl_crra') = 1.5;
mp_params('fl_beta') = 0.94;
% call function
ff_vfi_az_bisec_loop(mp_params);

Elapsed time is 13.575906 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
          i    idx    ndim    numel    rowN    colN     sum      mean      std      coefvari     min
          -    ---    ----    -----    ----    ----    -----    ------    ------    --------    ---

    ap    1     1      2       700     100      7      15835    22.621    13.367    0.59091      0

xxx TABLE:ap xxxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5          c6          c7
            -------    -------    -------    -------    -------    -------    ------
```

r1	0	0	0	0	0	0.38021	1.4609
r2	0.19477	0.18872	0.19731	0.24709	0.41492	0.79311	1.8893
r3	0.54595	0.54109	0.55664	0.62239	0.81173	1.2132	2.3195
r4	1.0101	1.0101	1.0101	1.0189	1.2217	1.6363	2.7464
r5	1.4388	1.4362	1.459	1.5151	1.6354	2.0602	3.1804
r96	43.225	43.246	43.3	43.422	43.632	44.155	45.413
r97	43.69	43.71	43.765	43.887	44.096	44.618	45.879
r98	44.154	44.174	44.228	44.352	44.559	45.083	46.344
r99	44.618	44.638	44.693	44.815	45.024	45.548	46.809
r100	45.08	45.101	45.156	45.28	45.487	46.012	47.273

1.3.2 Test FF_VFI_AZ_BISEC_LOOP Speed Tests

Call the function with different a and z grid size, print out speed:

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_timer') = true;
mp_support('ls_ffcmd') = {};
```

A grid 50, shock grid 5:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 50;
mp_params('it_z_n') = 5;
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 4.733351 seconds.
```

A grid 100, shock grid 7:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 100;
mp_params('it_z_n') = 7;
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 13.889250 seconds.
```

A grid 200, shock grid 9:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 200;
mp_params('it_z_n') = 9;
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 38.195963 seconds.
```

1.3.3 Test FF_VFI_AZ_BISEC_LOOP Control Outputs

Run the function first without any outputs;

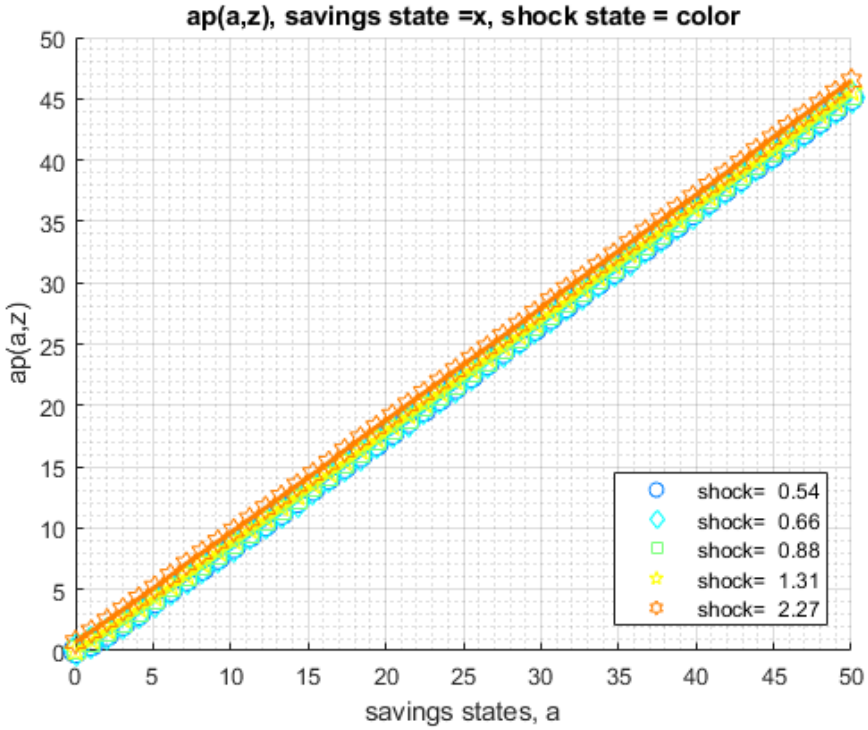
```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 50;
mp_params('it_z_n') = 5;
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_timer') = false;
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
```

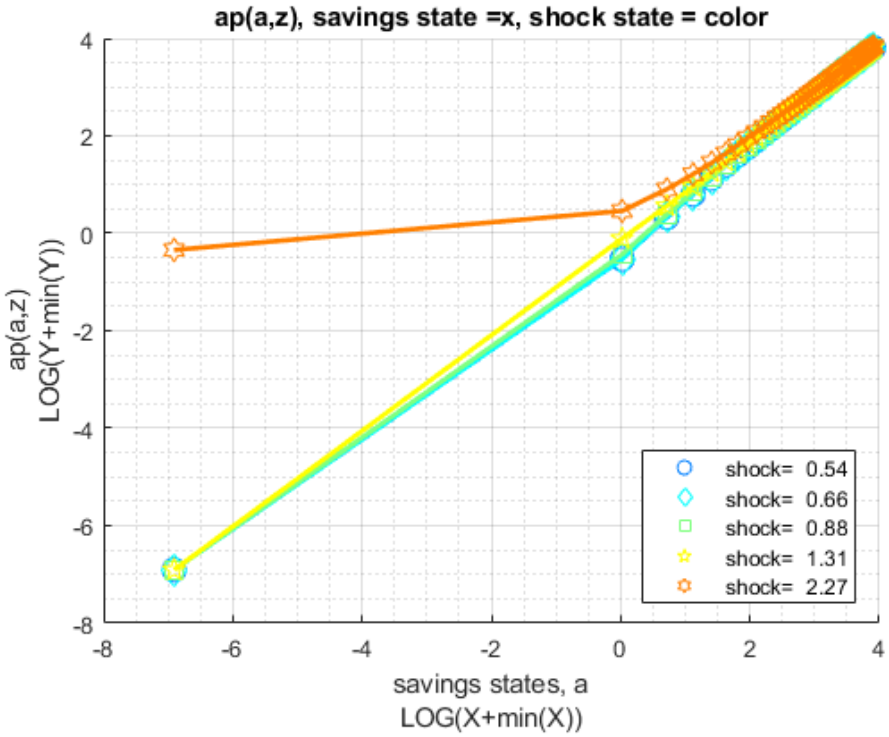
Run the function and show policy function for savings choice. For ls_ffcmd, ls_ffsna, ls_ffgrh, can include these: 'v', 'ap', 'c', 'y', 'coh', 'savefraccoh'. These are value, aprime savings choice, consumption, income, cash on hand, and savings fraction as cash-on-hand.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
% ls_ffcmd: summary print which outcomes
mp_support('ls_ffcmd') = {};
% ls_ffsna: detail print which outcomes
mp_support('ls_ffsna') = {'ap'};
% ls_ffgrh: graphical print which outcomes
mp_support('ls_ffgrh') = {'ap'};
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 4.728102 seconds.
xxx  ff_vfi_az_vec, outcome=ap  xxxxxxxxxxxxxxxxxxxxxxxxxx
```

group	a	mean_z_0_54195	mean_z_0_66401	mean_z_0_88162	mean_z_1_3095	mean_z
1	0	0	0	0	0	0.7
2	1.0204	0.58666	0.5889	0.64782	0.89696	1.
3	2.0408	1.3824	1.3926	1.4723	1.7501	2.
4	3.0612	2.2189	2.2357	2.3281	2.6228	3.
5	4.0816	3.0774	3.0995	3.2007	3.5069	4.
6	5.102	4.0815	4.0816	4.0833	4.3986	5.
7	6.1224	4.9896	5.0184	5.1021	5.2928	6.
8	7.1429	5.8564	5.8902	6.0116	6.1873	7.
9	8.1633	6.7406	6.7749	6.8956	7.1428	7.
10	9.1837	7.6346	7.6706	7.7922	8.1324	8.
11	10.204	8.5353	8.572	8.6944	9.0301	9.
12	11.224	9.4411	9.4787	9.6014	9.9343	10
13	12.245	10.35	10.389	10.512	10.845	11
14	13.265	11.263	11.302	11.427	11.761	12
15	14.286	12.245	12.245	12.344	12.68	13
16	15.306	13.224	13.264	13.265	13.6	14
17	16.327	14.141	14.182	14.286	14.523	15
18	17.347	15.052	15.097	15.228	15.446	16
19	18.367	15.971	16.014	16.147	16.37	17
20	19.388	16.889	16.933	17.065	17.347	18
21	20.408	17.811	17.856	17.989	18.34	19
22	21.429	18.735	18.779	18.912	19.265	20
23	22.449	19.66	19.705	19.838	20.187	2
24	23.469	20.586	20.632	20.765	21.113	21
25	24.49	21.513	21.559	21.693	22.041	22
26	25.51	22.449	22.488	22.622	22.97	
27	26.531	23.469	23.47	23.551	23.9	24
28	27.551	24.418	24.464	24.49	24.831	25
29	28.571	25.348	25.394	25.51	25.763	26
30	29.592	26.276	26.325	26.46	26.696	27
31	30.612	27.203	27.252	27.392	27.629	28
32	31.633	28.135	28.182	28.321	28.571	29
33	32.653	29.067	29.115	29.251	29.592	30
34	33.673	29.998	30.047	30.185	30.542	31
35	34.694	30.931	30.979	31.118	31.476	32
36	35.714	31.866	31.913	32.05	32.407	33
37	36.735	32.799	32.848	32.985	33.34	34
38	37.755	33.733	33.782	33.921	34.276	35
39	38.776	34.694	34.718	34.856	35.211	36
40	39.796	35.714	35.714	35.792	36.145	3
41	40.816	36.651	36.7	36.735	37.082	37
42	41.837	37.587	37.636	37.755	38.02	38

43	42.857	38.523	38.573	38.712	38.957	39
44	43.878	39.457	39.508	39.648	39.894	40
45	44.898	40.391	40.441	40.585	40.832	41
46	45.918	41.328	41.379	41.519	41.836	42
47	46.939	42.266	42.316	42.455	42.816	43
48	47.959	43.204	43.254	43.393	43.754	44
49	48.98	44.14	44.191	44.332	44.693	45
50	50	45.077	45.128	45.269	45.631	46





Run the function and show summaries for savings and fraction of coh saved:

```
mp_params('it_a_n') = 100;
mp_params('it_z_n') = 9;
mp_support('ls_ffcmd') = {'ap', 'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
mp_support('bl_vfi_store_all') = true; % store c(a,z), y(a,z)
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 18.110812 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coef
ap	1	1	2	900	100	9	20493	22.77	13.386	0.5
savefraccoh	2	2	2	900	100	9	701.94	0.77994	0.13136	0.16

```
xxx TABLE:ap xxxxxxxxxxxxxxxxx
```

	c1	c2	c3	c4	c5	c6	c7	c8	
r1	0	0	0	0	0	0	0.20716	0.89208	2
r2	0.19971	0.19144	0.18896	0.2007	0.24755	0.38215	0.61592	1.3126	2
r3	0.55145	0.54262	0.54255	0.5618	0.62321	0.77699	1.0303	1.7326	3
r4	1.0101	1.0101	1.0101	1.0101	1.0198	1.1844	1.5151	2.1613	3
r5	1.4445	1.436	1.4393	1.4657	1.5152	1.5944	1.9615	2.5895	4
r96	43.226	43.233	43.257	43.313	43.424	43.584	43.951	44.764	4
r97	43.69	43.697	43.722	43.776	43.888	44.048	44.444	45.227	
r98	44.155	44.161	44.186	44.241	44.352	44.512	44.933	45.692	4
r99	44.619	44.626	44.65	44.707	44.817	44.976	45.398	46.156	4
r100	45.081	45.088	45.114	45.169	45.28	45.454	45.861	46.621	4

```
xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
```

	c1	c2	c3	c4	c5	c6	c7	c8
r1	0	0	0	0	0	0	0.10085	0.28368
r2	0.17696	0.16018	0.14648	0.1404	0.15072	0.19253	0.23949	0.35842
r3	0.33498	0.31679	0.30013	0.28853	0.2885	0.31047	0.33348	0.41451
r4	0.46678	0.45284	0.43437	0.40981	0.38082	0.39214	0.42003	0.46007
r5	0.53868	0.52254	0.50624	0.49144	0.47417	0.45067	0.47554	0.49651
r96	0.86817	0.86713	0.86597	0.86469	0.86323	0.86054	0.85786	0.85551
r97	0.86845	0.86744	0.86631	0.865	0.86356	0.86091	0.8588	0.8559
r98	0.86875	0.86774	0.86662	0.86533	0.8639	0.86128	0.85966	0.8563
r99	0.86903	0.86805	0.86692	0.86567	0.86424	0.86161	0.86002	0.8567
r100	0.86927	0.86829	0.8672	0.86594	0.86454	0.86222	0.86036	0.85709

1.3.4 Test FF_VFI_AZ_BISEC_LOOP Change Interest Rate and Discount

Show only save fraction of cash on hand:

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
```

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 50;
mp_params('it_z_n') = 5;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Solve the model with several different interest rates and discount factor:

```
% Lower Savings Incentives
mp_params('fl_beta') = 0.80;
mp_params('fl_r') = 0.01;
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 1.421696 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                  i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                  -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh   1     1      2       250      50      5      94.272    0.37709    0.25552    0.67

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
             c1         c2         c3         c4           c5
           -------    -------    -------    -------    ----------

    r1           0          0          0          0    0.00014733
    r2           0          0          0          0    0.00014733
    r3           0          0          0          0     0.0011239
    r4           0          0          0          0     0.0011544
    r5           0          0          0          0     0.0039009
    r46    0.67805    0.67137    0.66298     0.6526       0.64094
    r47    0.67964    0.67329    0.66536     0.6555       0.64439
    r48     0.6811    0.67506    0.66752    0.65818        0.6476
    r49    0.68242    0.67671     0.6696    0.66075       0.65059
    r50    0.68364    0.67826    0.67158    0.66319       0.65336

% Higher Savings Incentives
mp_params('fl_beta') = 0.95;
mp_params('fl_r') = 0.04;
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 5.579251 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                  i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                  -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh   1     1      2       250      50      5      146.44    0.58575    0.29994    0.51

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
             c1         c2           c3           c4          c5
           -------    -------    ----------    --------    -------

    r1           0          0             0    0.086968     0.3134
    r2           0          0             0    0.086938     0.3135
```

```
    r3           0          0             0    0.086877    0.31423
    r4           0          0             0    0.091393    0.31621
    r5           0          0    0.00036095     0.10012    0.31765
    r46    0.87894     0.8773       0.87437     0.86796    0.86643
    r47    0.88136     0.8798       0.87717     0.87083    0.86933
    r48    0.88358    0.88215       0.87983     0.87348    0.87202
    r49    0.88566    0.88432        0.8823     0.87595    0.87455
    r50    0.88761    0.88633       0.88465     0.87827    0.87687
```

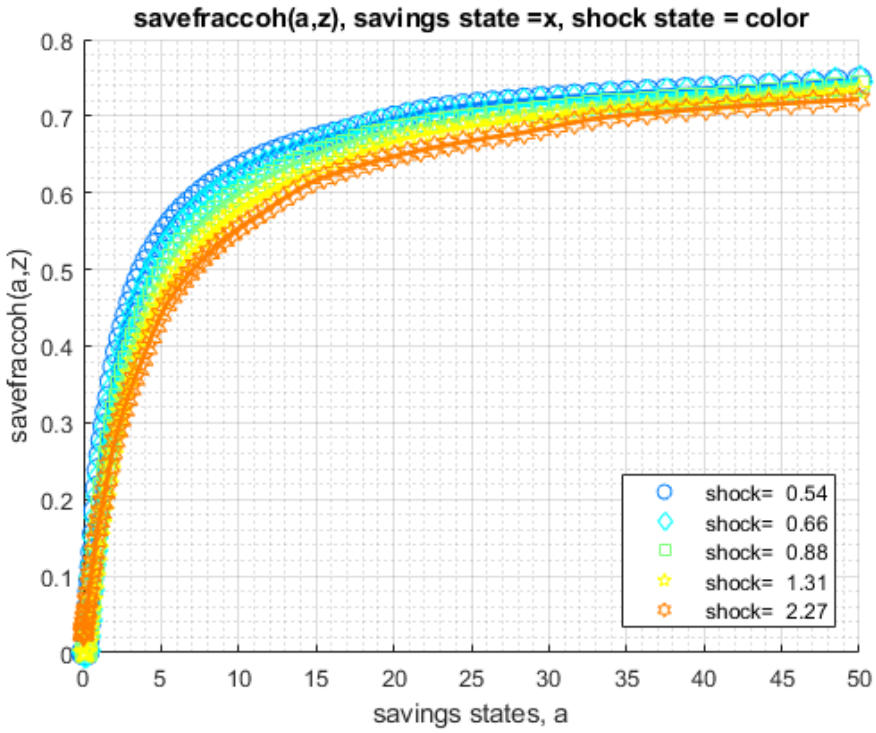
1.3.5 Test FF_VFI_AZ_BISEC_LOOP Changing Risk Aversion

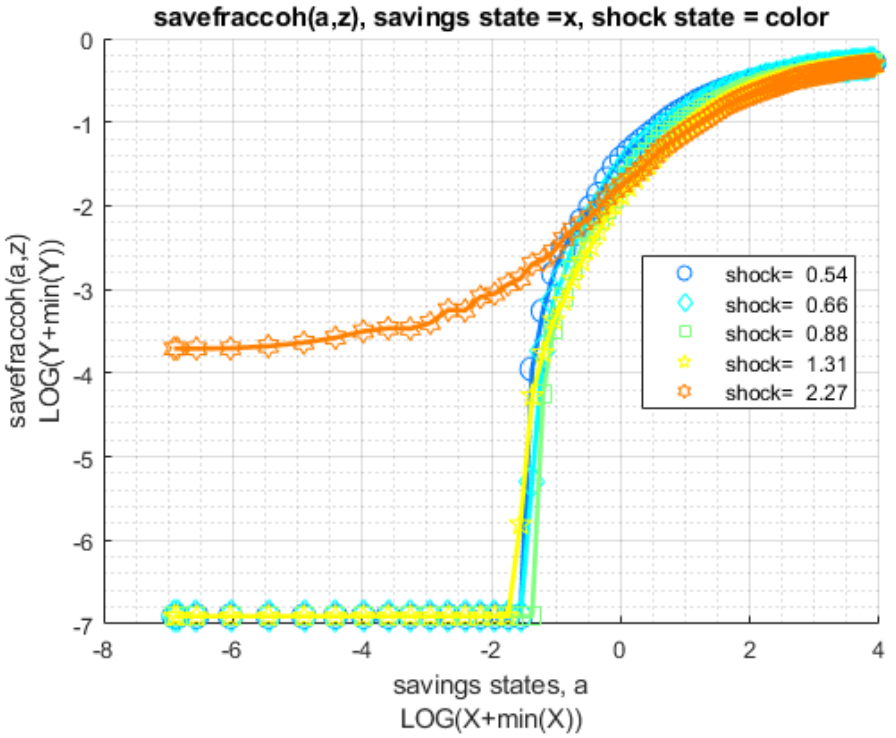
Here, again, show fraction of coh saved in summary tabular form, but also show it graphically.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {'savefraccoh'};
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 100;
mp_params('it_z_n') = 5;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Solve the model with different risk aversion levels, higher preferences for risk:

```
% Lower Risk Aversion
mp_params('fl_crra') = 0.5;
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 9.991698 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                   i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                   -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh    1     1      2       500     100      5      214.05    0.42811    0.27486    0.64

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxx
              c1         c2         c3         c4          c5
            -------    -------    -------    -------    --------

    r1            0          0          0          0    0.023554
    r2            0          0          0          0    0.023554
    r3            0          0          0          0    0.023554
    r4            0          0          0          0    0.023615
    r5            0          0          0          0    0.024256
    r96      0.7393    0.73551    0.73109    0.72261     0.71525
    r97     0.74049    0.73805    0.73374    0.72544     0.71702
    r98      0.7429    0.74052    0.73634    0.72825      0.7187
    r99     0.74525    0.74296    0.73887      0.731     0.72032
    r100    0.74757    0.74534    0.74113    0.73371     0.72187
```

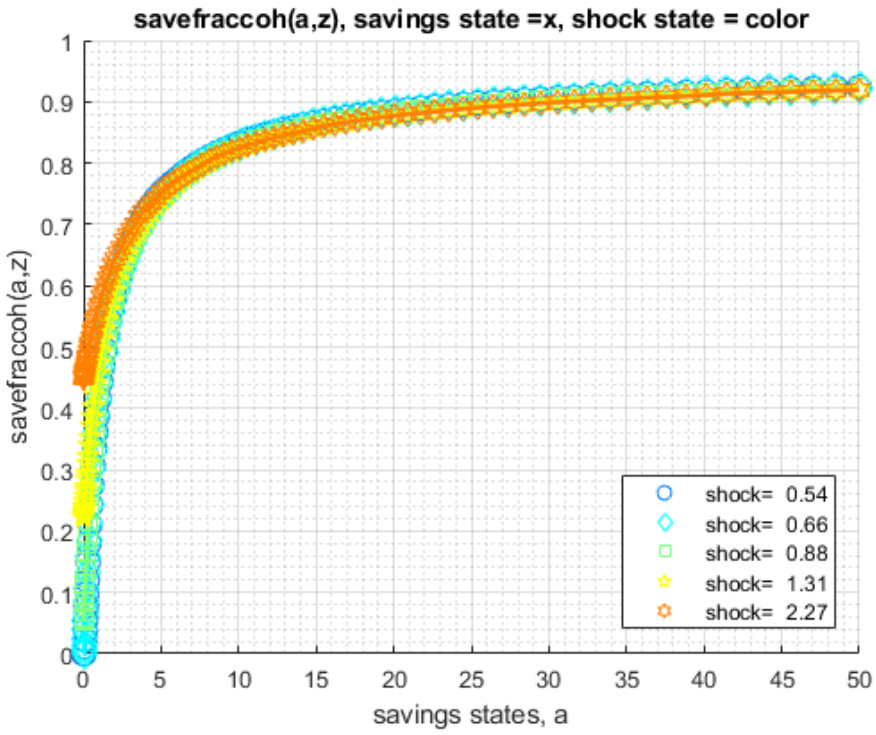


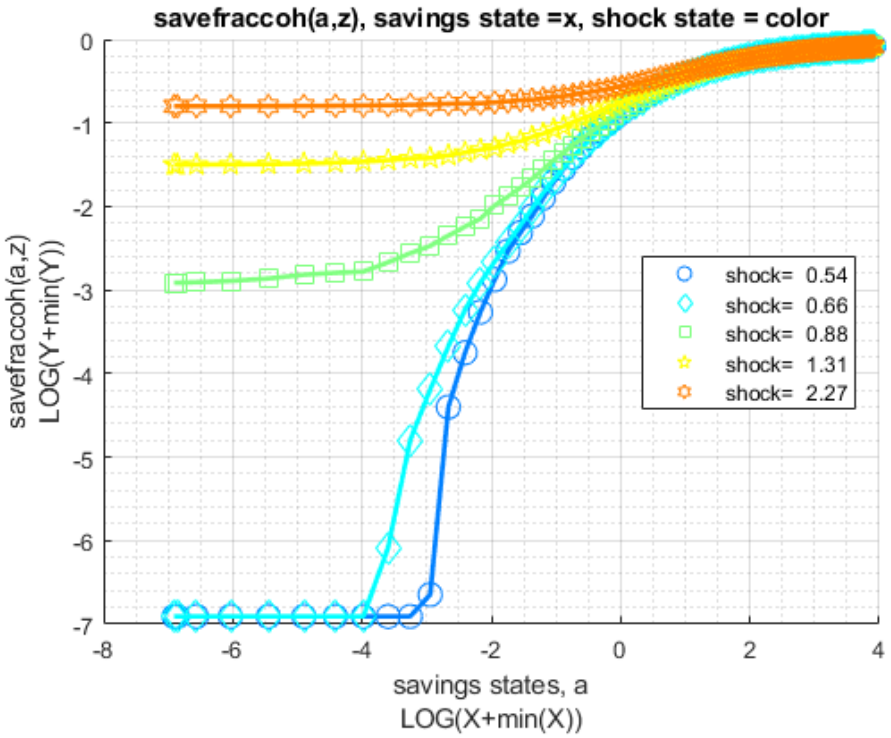


When risk aversion increases, at every state-space point, the household wants to save more.

```
% Higher Risk Aversion
mp_params('fl_crra') = 5;
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 8.955693 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
```

```
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i     idx     ndim     numel     rowN     colN      sum        mean         std        coef
                    -     ---     ----     -----     ----     ----     ------     -------     -------     ----

    savefraccoh     1      1       2        500      100       5       323.21     0.64642     0.28954     0.44

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5
            -------     -------     --------    -------     -------

    r1            0           0     0.053308    0.22219     0.44872
    r2            0           0     0.053338    0.22222     0.44875
    r3            0           0     0.053644     0.2224     0.44881
    r4            0           0     0.054498    0.22286     0.44899
    r5            0           0     0.056115    0.22378     0.44936
    r96     0.91981     0.91901      0.91813    0.91575     0.91523
    r97     0.92081     0.92002      0.91914    0.91682      0.9163
    r98     0.92176       0.921      0.92014    0.91785      0.9173
    r99     0.92268     0.92194      0.92112    0.91883     0.91831
    r100    0.92356     0.92286      0.92203    0.91981     0.91926
```





1.3.6 Test FF_VFI_AZ_BISEC_LOOP with Higher Uncertainty

Increase the standard deviation of the Shock.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 100;
mp_params('it_z_n') = 5;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Lower standard deviation of shock:

```
% Lower Risk Aversion
mp_params('fl_shk_std') = 0.10;
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 10.016136 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i    idx    ndim    numel    rowN    colN     sum       mean        std       coef
                    -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh     1     1      2       500     100      5      266.09    0.53217    0.31606    0.59

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5
            -------    -------    -------    -------    --------
```

```
    r1            0          0          0          0    0.027887
    r2            0          0          0          0    0.027887
    r3            0          0          0          0    0.027887
    r4            0          0          0          0    0.027857
    r5            0          0          0          0    0.027826
    r96     0.85941    0.85841    0.85734    0.85621     0.85218
    r97     0.86076    0.85978    0.85871    0.85764     0.85368
    r98     0.86204    0.86109    0.86008    0.85902     0.85511
    r99     0.86329    0.86234    0.86137    0.86036     0.85688
    r100    0.86448    0.86359    0.86262    0.86164     0.85862
```

Higher shock standard deviation: low shock high asset save more, high shock more asset save less, high shock low asset save more:

```
% Higher Risk Aversion
mp_params('fl_shk_std') = 0.40;
ff_vfi_az_bisec_loop(mp_params, mp_support);

Elapsed time is 10.186494 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                   i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                   -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh    1     1      2       500     100      5      324.66    0.64932    0.26596    0.40

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5
            -------    -------    -------    -------    -------

    r1            0          0          0    0.26564    0.67546
    r2            0          0          0    0.26568    0.67549
    r3            0          0          0    0.26586    0.67549
    r4            0          0          0    0.26635    0.67552
    r5            0          0          0    0.26732    0.67558
    r96     0.88071    0.87922    0.87495    0.86723    0.85859
    r97     0.88178    0.88032     0.8762    0.86826    0.86085
    r98     0.88282    0.88139    0.87739    0.86927    0.86317
    r99     0.88383    0.88212    0.87855    0.87028    0.86668
    r100    0.88483    0.88279    0.87937    0.87128    0.87003
```

1.4 FF_VFI_AZ_BISEC_VEC Dynamic Savings Problem Vectorized Continuous Exact

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ff_vfi_az_bisec_vec** from the **MEconTools Package.** This function solves the dynamic programming problem for a (a,z) model. Households can save a, and face AR(1) shock z. The problem is solved over the infinite horizon. This is a vectorized code, it is much faster for larger state-space problems then looped code.

The code uses continuous choices, solved with bi(multi)section. The state-space is on a grid, but choice grids are in terms of percentage of resources available, which is individual specific, to save and solved exactly up to ((1/(2)^16)*100=0.001525878) percentage of cash on hand. THe **ff_vfi_az_vec** from the **MEconTools Package** solves the same problem using vectorized common grid code where the choice

set and state space share the same grid.

This is the vectorized code, its speed is much faster than the looped code. The function is designed to have small memory footprint and requires low computing resources, yet is fast.

Links to Four Code:

Four Core Savings/Borrowing Dynamic Programming Solution Functions that are functions in the **MEconTools Package.** :

- Common Choice and States Grid : **ff_vfi_az_loop**, slow should use for testing new models
- Common Choice and States Grid : **ff_vfi_az_vec**, fast good for many purposes
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand :**ff_vfi_az_bisec_loop**, high precision even with small grid
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand : **ff_vfi_az_bisec_vec**, precision and speed

1.4.1 Test FF_VFI_AZ_BISEC_VEC Defaults

Call the function with defaults. By default, shows the asset policy function summary. Model parameters can be changed by the mp_params.

```
%mp_params
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('fl_crra') = 1.5;
mp_params('fl_beta') = 0.94;
% call function
ff_vfi_az_bisec_vec(mp_params);

Elapsed time is 0.341348 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
          i    idx    ndim    numel    rowN    colN     sum      mean      std      coefvari    min
          -    ---    ----    -----    ----    ----    -----    ------    ------    --------    ---

    ap    1     1      2       700     100      7      15835    22.621    13.367    0.59091      0

xxx TABLE:ap xxxxxxxxxxxxxxxxxx
              c1         c2         c3         c4         c5         c6         c7
            -------    -------    -------    -------    -------    -------    ------

    r1            0          0          0          0          0    0.38021    1.4609
    r2      0.19477    0.18872    0.19731    0.24709    0.41492    0.79311    1.8893
    r3      0.54595    0.54109    0.55664    0.62239    0.81173     1.2132    2.3195
    r4       1.0101     1.0101     1.0101     1.0189     1.2217     1.6363    2.7464
    r5       1.4388     1.4362      1.459     1.5151     1.6354     2.0602    3.1804
    r96      43.225     43.246       43.3     43.422     43.632     44.155    45.413
    r97       43.69      43.71     43.765     43.887     44.096     44.618    45.879
    r98      44.154     44.174     44.228     44.352     44.559     45.083    46.344
    r99      44.618     44.638     44.693     44.815     45.024     45.548    46.809
    r100      45.08     45.101     45.156      45.28     45.487     46.012    47.273
```

1.4.2 Test FF_VFI_AZ_BISEC_VEC Speed Tests

Call the function with defaults. By default, shows the asset policy function summary. Model parameters can be changed by the mp_params.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
```

```
mp_support('bl_timer') = true;
mp_support('ls_ffcmd') = {};
```

A grid 50, shock grid 5:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 50;
mp_params('it_z_n') = 5;
ff_vfi_az_bisec_vec(mp_params, mp_support);

Elapsed time is 0.188450 seconds.
```

A grid 750, shock grid 15:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 750;
mp_params('it_z_n') = 15;
ff_vfi_az_bisec_vec(mp_params, mp_support);

Elapsed time is 12.017243 seconds.
```

A grid 600, shock grid 45:

```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 600;
mp_params('it_z_n') = 45;
ff_vfi_az_bisec_vec(mp_params, mp_support);

Elapsed time is 22.719622 seconds.
```

1.4.3 Test FF_VFI_AZ_BISEC_VEC Control Outputs

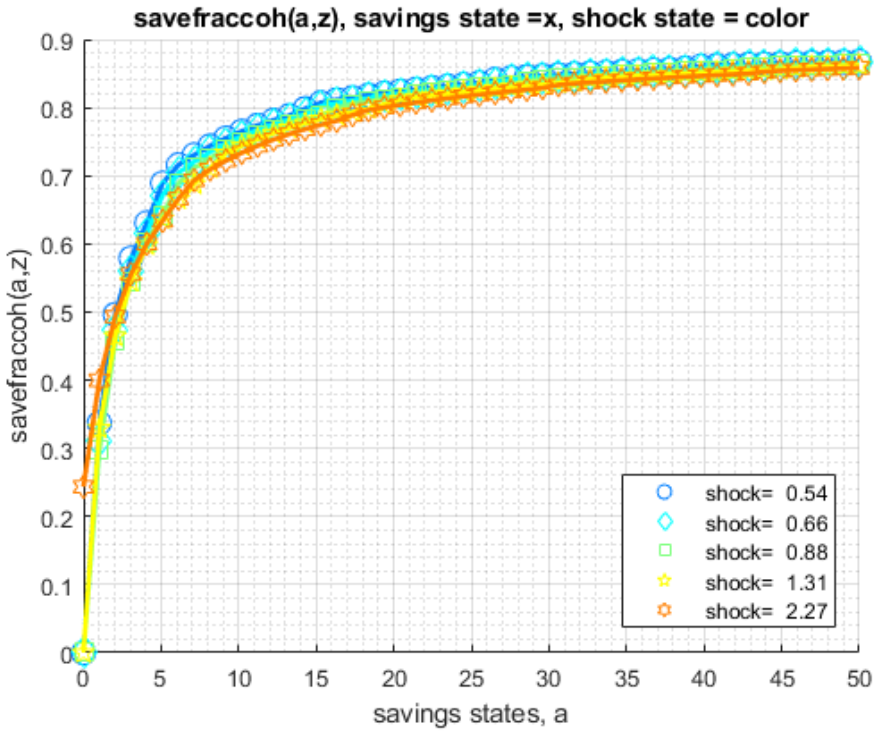
Run the function first without any outputs;

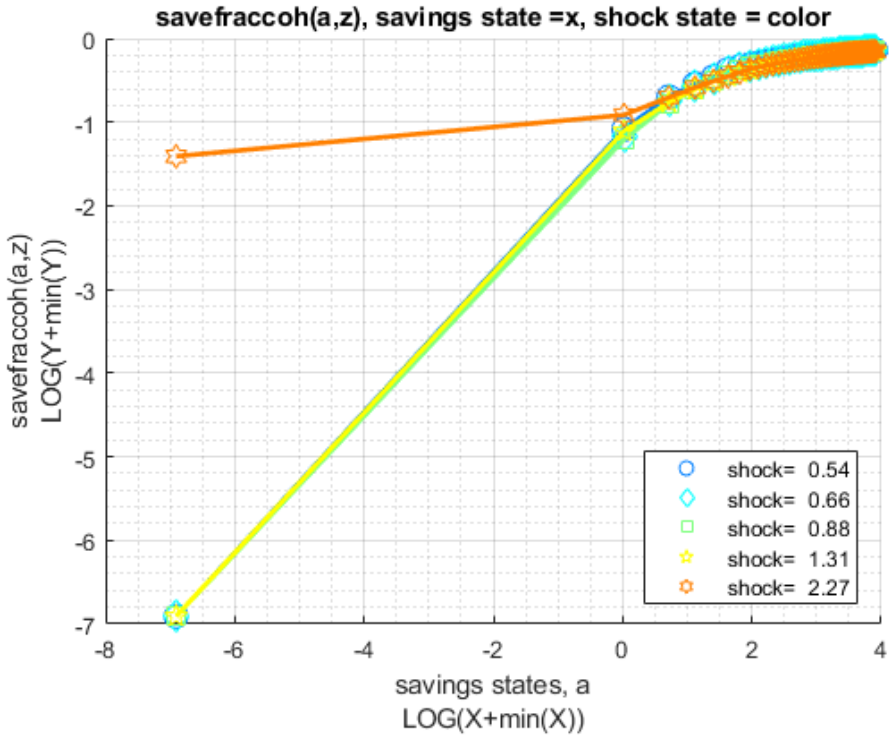
```
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 50;
mp_params('it_z_n') = 5;
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_timer') = false;
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
```

Run the function and show policy function for savings choice. For ls_ffcmd, ls_ffsna, ls_ffgrh, can include these: 'v', 'ap', 'c', 'y', 'coh', 'savefraccoh'. These are value, aprime savings choice, consumption, income, cash on hand, and savings fraction as cash-on-hand.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
% ls_ffcmd: summary print which outcomes
mp_support('ls_ffcmd') = {};
% ls_ffsna: detail print which outcomes
mp_support('ls_ffsna') = {'savefraccoh'};
% ls_ffgrh: graphical print which outcomes
mp_support('ls_ffgrh') = {'savefraccoh'};
ff_vfi_az_bisec_vec(mp_params, mp_support);

Elapsed time is 0.160923 seconds.
xxx  ff_vfi_az_vec, outcome=savefraccoh  xxxxxxxxxxxxxxxxxxxxxxxxxx
    group      a       mean_z_0_54195    mean_z_0_66401    mean_z_0_88162    mean_z_1_3095    mean_z
    -----    ------    --------------    --------------    --------------    -------------    ------

      1          0            0                 0                 0                0          0.2
```

2	1.0204	0.33724	0.31063	0.29793	0.32952	0.
3	2.0408	0.49626	0.47337	0.4572	0.46446	0.4
4	3.0612	0.57912	0.56065	0.5457	0.54484	0.5
5	4.0816	0.63096	0.61577	0.60252	0.59846	0.6
6	5.102	0.68907	0.67137	0.64222	0.63694	0.6
7	6.1224	0.71595	0.7043	0.6891	0.66563	0.6
8	7.1429	0.73066	0.72084	0.71144	0.68766	0.6
9	8.1633	0.74391	0.73503	0.72618	0.71119	0.7
10	9.1837	0.75538	0.74739	0.73918	0.73335	0.7
11	10.204	0.7653	0.75798	0.75032	0.74412	0.7
12	11.224	0.77394	0.76719	0.75999	0.75367	0.7
13	12.245	0.78147	0.77525	0.76847	0.76231	0.7
14	13.265	0.78816	0.78233	0.77598	0.77006	0.7
15	14.286	0.79841	0.79035	0.78266	0.77699	0.7
16	15.306	0.80723	0.80201	0.7888	0.78321	0.7
17	16.327	0.81135	0.8065	0.79972	0.78883	0.7
18	17.347	0.81474	0.81031	0.80534	0.79386	0.7
19	18.367	0.81815	0.81388	0.80918	0.79841	0.7
20	19.388	0.82121	0.81715	0.8126	0.805	0.8
21	20.408	0.82414	0.82026	0.81596	0.81172	0.8
22	21.429	0.82685	0.82313	0.81898	0.81492	0.8
23	22.449	0.82938	0.82584	0.82182	0.81776	0.8
24	23.469	0.83177	0.82838	0.8245	0.8205	0.8
25	24.49	0.83399	0.83073	0.827	0.8231	0.8
26	25.51	0.83634	0.83296	0.82935	0.82554	0.8
27	26.531	0.84156	0.83689	0.83155	0.82786	0.8
28	27.551	0.84394	0.84098	0.8339	0.83003	0.8
29	28.571	0.84553	0.84266	0.83875	0.8321	0.8
30	29.592	0.84693	0.84425	0.84107	0.83405	0.8
31	30.612	0.84821	0.84562	0.84266	0.83589	0.8
32	31.633	0.84956	0.84699	0.84409	0.83787	0.8
33	32.653	0.85084	0.84837	0.84547	0.84199	0.8
34	33.673	0.852	0.84962	0.84684	0.84391	0.8
35	34.694	0.85316	0.85081	0.84815	0.84528	0.8
36	35.714	0.85429	0.852	0.84934	0.8465	0.8
37	36.735	0.85532	0.85313	0.85053	0.84773	0.8
38	37.755	0.85633	0.8542	0.85169	0.84895	0.8
39	38.776	0.85795	0.85523	0.85279	0.85008	0.8
40	39.796	0.86091	0.85767	0.85383	0.85114	0.8
41	40.816	0.86176	0.85975	0.85499	0.85221	0.
42	41.837	0.86256	0.8606	0.85786	0.85325	0.
43	42.857	0.86332	0.86143	0.85917	0.85423	0.
44	43.878	0.86399	0.86216	0.85999	0.85517	0.8
45	44.898	0.86463	0.86283	0.86079	0.85609	0.8
46	45.918	0.86533	0.86356	0.86149	0.85831	0.8
47	46.939	0.86601	0.86427	0.86219	0.85996	0.8
48	47.959	0.86665	0.86494	0.86292	0.86073	0.8
49	48.98	0.86723	0.86558	0.86362	0.86146	0.8
50	50	0.86781	0.86619	0.86427	0.86216	0.8





Run the function and show summaries for savings and fraction of coh saved:

```
mp_params('it_a_n') = 100;
mp_params('it_z_n') = 9;
mp_support('ls_ffcmd') = {'ap', 'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
mp_support('bl_vfi_store_all') = true; % store c(a,z), y(a,z)
ff_vfi_az_bisec_vec(mp_params, mp_support);
```

```
Elapsed time is 0.443544 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i    idx    ndim    numel    rowN    colN     sum       mean        std       coef
                    -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    ap              1     1      2       900     100      9       20493      22.77     13.386      0.5
    savefraccoh     2     2      2       900     100      9      701.94    0.77994    0.13136     0.16

xxx TABLE:ap xxxxxxxxxxxxxxxxx
               c1          c2          c3          c4          c5          c6          c7          c8          
             -------     -------     -------     ------     -------     -------     -------     -------     -

    r1             0           0           0           0           0           0     0.20716     0.89208     2
    r2       0.19971     0.19144     0.18896      0.2007     0.24755     0.38215     0.61592      1.3126     2
    r3       0.55145     0.54262     0.54255      0.5618     0.62321     0.77699      1.0303      1.7326     3
    r4        1.0101      1.0101      1.0101      1.0101      1.0198      1.1844      1.5151      2.1613     3
    r5        1.4445       1.436      1.4393      1.4657      1.5152      1.5944      1.9615      2.5895     4
    r96       43.226      43.233      43.257      43.313      43.424      43.584      43.951      44.764     4
    r97        43.69      43.697      43.722      43.776      43.888      44.048      44.444      45.227
    r98       44.155      44.161      44.186      44.241      44.352      44.512      44.933      45.692     4
    r99       44.619      44.626       44.65      44.707      44.817      44.976      45.398      46.156     4
    r100      45.081      45.088      45.114      45.169       45.28      45.454      45.861      46.621     4

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
               c1          c2          c3          c4          c5          c6          c7          c8          
             -------     -------     -------     -------     -------     -------     -------     -------

    r1             0           0           0           0           0           0     0.10085     0.28368
    r2       0.17696     0.16018     0.14648      0.1404     0.15072     0.19253     0.23949     0.35842
    r3       0.33498     0.31679     0.30013     0.28853      0.2885     0.31047     0.33348     0.41451
    r4       0.46678     0.45284     0.43437     0.40981     0.38082     0.39214     0.42003     0.46007
    r5       0.53868     0.52254     0.50624     0.49144     0.47417     0.45067     0.47554     0.49651
    r96      0.86817     0.86713     0.86597     0.86469     0.86323     0.86054     0.85786     0.85551
    r97      0.86845     0.86744     0.86631       0.865     0.86356     0.86091      0.8588      0.8559
    r98      0.86875     0.86774     0.86662     0.86533      0.8639     0.86128     0.85966      0.8563
    r99      0.86903     0.86805     0.86692     0.86567     0.86424     0.86161     0.86002      0.8567
    r100     0.86927     0.86829      0.8672     0.86594     0.86454     0.86222     0.86036     0.85709
```

1.4.4 Test FF_VFI_AZ_BISEC_VEC Change Interest Rate and Discount

Show only save fraction of cash on hand:

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 750;
mp_params('it_z_n') = 9;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Solve the model with several different interest rates and discount factor:

```
% Lower Savings Incentives
```

```
mp_params('fl_beta') = 0.80;
mp_params('fl_r') = 0.01;
ff_vfi_az_bisec_vec(mp_params, mp_support);

Elapsed time is 2.064615 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                    -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh     1     1      2       6750     750      9      2573.6    0.38127    0.24694    0.64

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5          c6          c7          c8
            -------    -------    -------    -------    -------    -------    -------    --------

    r1            0          0          0          0          0          0          0    0.014734
    r2            0          0          0          0          0          0          0    0.014734
    r3            0          0          0          0          0          0          0    0.014734
    r4            0          0          0          0          0          0          0    0.014734
    r5            0          0          0          0          0          0          0    0.014734
    r746    0.68623    0.68354    0.68095    0.67686    0.67308    0.66722    0.66044     0.65098
    r747    0.68663    0.68364    0.68119    0.67698     0.6732    0.66734    0.66063     0.65117
    r748    0.68675     0.6837    0.68129    0.67711    0.67332    0.66749    0.66078     0.65138
    r749    0.68681    0.68379    0.68141     0.6772    0.67344    0.66764    0.66096     0.65184
    r750     0.6869    0.68385     0.6815    0.67759    0.67357    0.66777    0.66111     0.65233

% Higher Savings Incentives
mp_params('fl_beta') = 0.95;
mp_params('fl_r') = 0.04;
ff_vfi_az_bisec_vec(mp_params, mp_support);

Elapsed time is 8.355503 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                    -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh     1     1      2       6750     750      9      4047.5    0.59963    0.28766    0.47

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5          c6          c7          c8
            -------    -------    -------    -------    -------    --------    -------    -------

    r1            0          0          0          0          0    0.046381    0.17205    0.33791
    r2            0          0          0          0          0    0.046381    0.17205    0.33791
    r3            0          0          0          0          0    0.046381    0.17205    0.33791
    r4            0          0          0          0          0    0.046381    0.17205    0.33791
    r5            0          0          0          0          0    0.046381    0.17205    0.33791
    r746    0.88633    0.88548    0.88435    0.88337    0.88194     0.88041    0.87852    0.87629
    r747    0.88645     0.8856    0.88447    0.88349    0.88206     0.88053    0.87867    0.87644
    r748    0.88657    0.88575    0.88459    0.88361    0.88221     0.88068    0.87882    0.87659
    r749     0.8867    0.88587    0.88474    0.88377    0.88233     0.88084    0.87897    0.87675
    r750    0.88682    0.88599    0.88486    0.88389    0.88248     0.88096     0.8791     0.8769
```

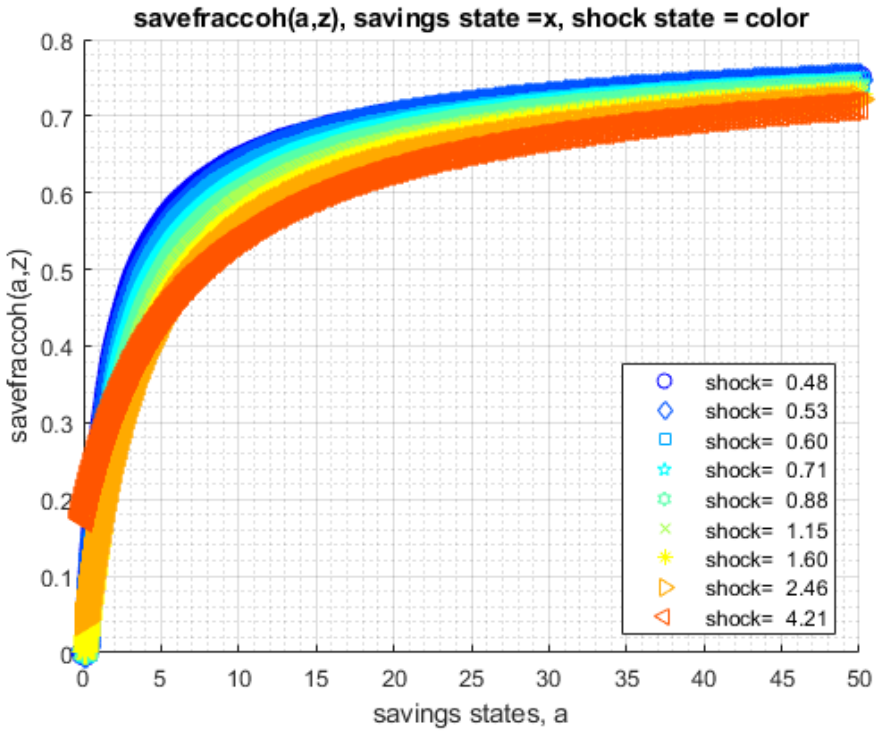
1.4.5 Test FF_VFI_AZ_BISEC_VEC Changing Risk Aversion

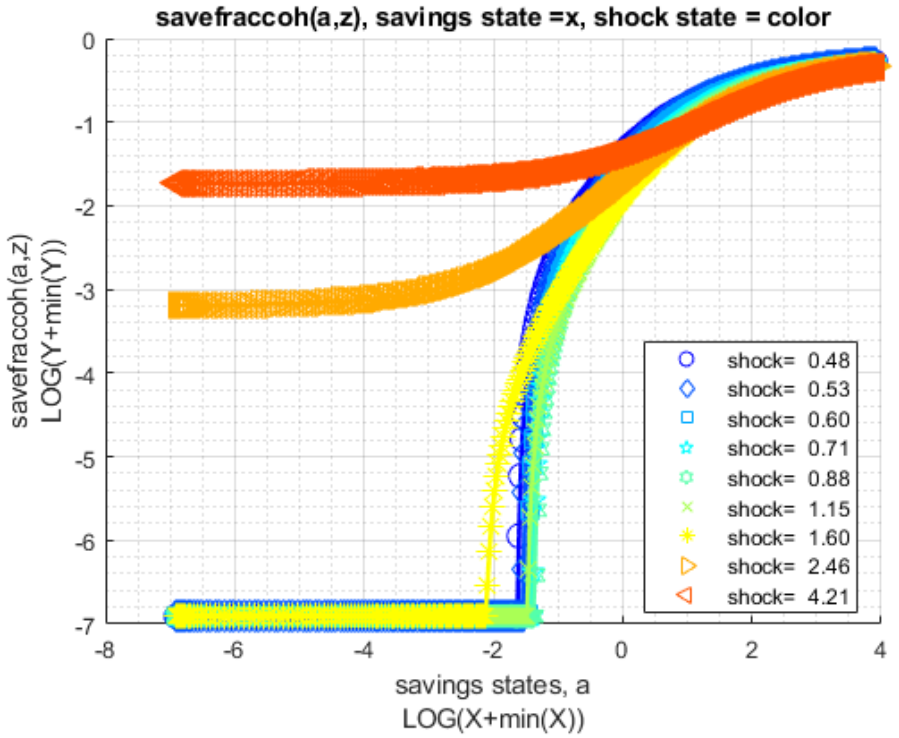
Here, again, show fraction of coh saved in summary tabular form, but also show it graphically.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {'savefraccoh'};
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 750;
mp_params('it_z_n') = 9;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Solve the model with different risk aversion levels, higher preferences for risk:

```
% Lower Risk Aversion
mp_params('fl_crra') = 0.5;
ff_vfi_az_bisec_vec(mp_params, mp_support);

Elapsed time is 6.947134 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                   i    idx    ndim    numel    rowN    colN     sum       mean        std      coef
                   -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh    1     1      2      6750     750      9      2940.8    0.43567    0.26675    0.61

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxx
              c1         c2         c3         c4         c5         c6         c7          c8
            -------    -------    -------    -------    -------    -------    -------    --------

    r1            0          0          0          0          0          0          0    0.040155
    r2            0          0          0          0          0          0          0    0.040155
    r3            0          0          0          0          0          0          0    0.040155
    r4            0          0          0          0          0          0          0    0.040155
    r5            0          0          0          0          0          0          0    0.040155
    r746    0.74928    0.74699    0.74427    0.74165    0.73826    0.73371    0.72828     0.72074
    r747    0.74949    0.74711     0.7444    0.74195    0.73844    0.73405    0.72847     0.72096
    r748    0.74958    0.74723    0.74452    0.74226     0.7386    0.73432    0.72865     0.72117
    r749    0.74971    0.74736    0.74467    0.74241    0.73875    0.73451    0.72883     0.72139
    r750    0.74983    0.74748    0.74491    0.74253     0.7389    0.73466    0.72905     0.72178
```

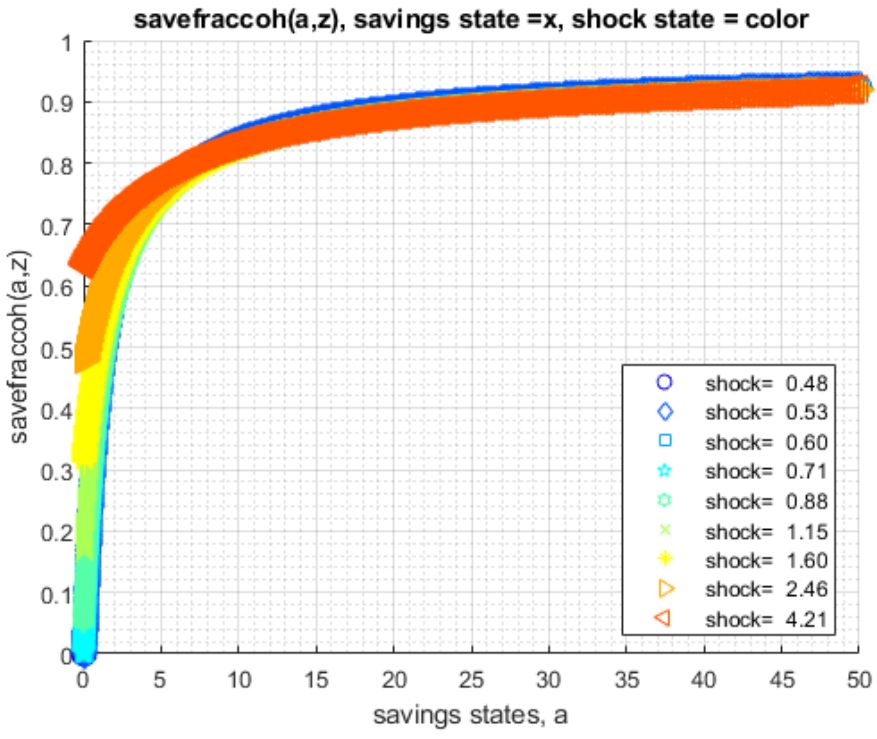


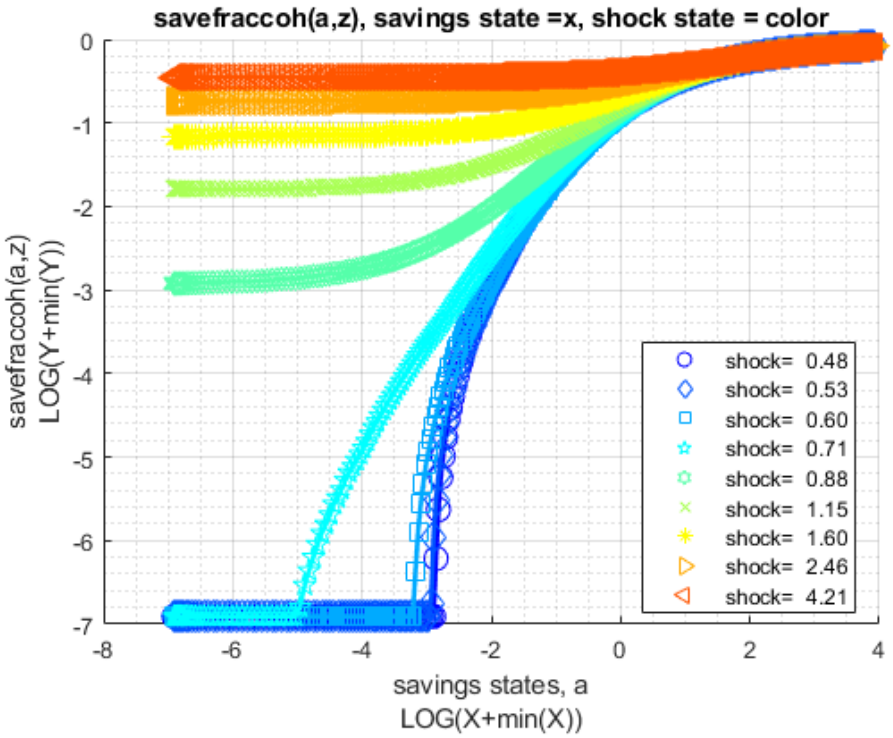


When risk aversion increases, at every state-space point, the household wants to save more.

```
% Higher Risk Aversion
mp_params('fl_crra') = 5;
ff_vfi_az_bisec_vec(mp_params, mp_support);

Elapsed time is 6.425400 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
```

```
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i     idx     ndim     numel     rowN     colN     sum      mean       std       coefvar
                    -     ---     ----     -----     ----     ----     ----     -------    ------    -------

    savefraccoh     1      1       2       6750      750       9       4449     0.65911    0.2826    0.42876

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5          c6          c7          c8
            -------     -------     -------     -------     -------     -------     -------     -------

    r1            0           0           0           0     0.05282     0.16466     0.31347     0.47728
    r2            0           0           0           0     0.05282     0.16466     0.31347     0.47728
    r3            0           0           0           0     0.05282     0.16466     0.31347     0.47728
    r4            0           0           0           0     0.05282     0.16466     0.31347     0.47728
    r5            0           0           0           0     0.05282     0.16466     0.31347     0.47728
    r746    0.92341     0.92298     0.92249     0.92176     0.92097      0.9202      0.9191     0.91825
    r747    0.92353      0.9231     0.92261     0.92188     0.92109     0.92033     0.91923      0.9184
    r748    0.92365     0.92319     0.92271       0.922     0.92121     0.92045     0.91935     0.91852
    r749    0.92377     0.92332     0.92283     0.92213     0.92133     0.92057      0.9195     0.91868
    r750    0.92387     0.92344     0.92295     0.92225     0.92145     0.92069     0.91962      0.9188
```





1.4.6 Test FF_VFI_AZ_BISEC_VEC with Higher Uncertainty

Increase the standard deviation of the Shock.

```
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {'savefraccoh'};
mp_support('ls_ffsna') = {};
mp_support('ls_ffgrh') = {};
mp_params = containers.Map('KeyType','char', 'ValueType','any');
mp_params('it_a_n') = 750;
mp_params('it_z_n') = 9;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
```

Lower standard deviation of shock:

```
% Lower Risk Aversion
mp_params('fl_shk_std') = 0.10;
ff_vfi_az_bisec_vec(mp_params, mp_support);

Elapsed time is 6.784360 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                    i    idx    ndim    numel    rowN    colN     sum       mean        std       coef
                    -    ---    ----    -----    ----    ----    ------    -------    -------    ----

    savefraccoh     1     1      2       6750     750      9      3617.7    0.53596    0.31083    0.57

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5          c6          c7           c8
            -------    -------    -------    -------    -------    -------    -------    --------
```

r1	0	0	0	0	0	0	0	0.034876
r2	0	0	0	0	0	0	0	0.034876
r3	0	0	0	0	0	0	0	0.034876
r4	0	0	0	0	0	0	0	0.034876
r5	0	0	0	0	0	0	0	0.034876
r746	0.8642	0.86359	0.86295	0.86192	0.86124	0.8603	0.85944	0.85835
r747	0.86436	0.86375	0.86314	0.8621	0.8614	0.86048	0.8596	0.85853
r748	0.86451	0.8639	0.86329	0.86225	0.86158	0.8607	0.85978	0.85871
r749	0.86466	0.86408	0.86344	0.86243	0.86173	0.86091	0.85996	0.85886
r750	0.86482	0.86424	0.86359	0.86259	0.86192	0.86112	0.86012	0.85905

Higher shock standard deviation: low shock high asset save more, high shock more asset save less, high shock low asset save more:

```
% Higher Risk Aversion
mp_params('fl_shk_std') = 0.40;
ff_vfi_az_bisec_vec(mp_params, mp_support);

Elapsed time is 7.804664 seconds.
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coefv
	-	---	----	-----	----	----	------	------	-------	-----
savefraccoh	1	1	2	6750	750	9	4755.4	0.7045	0.26237	0.372

```
xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxx
```

	c1	c2	c3	c4	c5	c6	c7	c8
	-------	-------	-------	-------	-------	-------	-------	-------
r1	0	0	0	0	0	0.152	0.44643	0.71928
r2	0	0	0	0	0	0.152	0.44643	0.71928
r3	0	0	0	0	0	0.152	0.44643	0.71928
r4	0	0	0	0	0	0.152	0.44643	0.71928
r5	0	0	0	0	0	0.152	0.44643	0.71928
r746	0.8914	0.89054	0.88944	0.88798	0.88599	0.88279	0.87788	0.87836
r747	0.89146	0.8906	0.8895	0.88807	0.88609	0.88288	0.878	0.87879
r748	0.89152	0.89066	0.88956	0.88813	0.88615	0.88297	0.87812	0.87919
r749	0.89158	0.89072	0.88963	0.88819	0.88624	0.88306	0.87824	0.87962
r750	0.89164	0.89079	0.88972	0.88828	0.8863	0.88316	0.87833	0.88001

Chapter 2

Summarize Policy and Value

2.1 FF_SUMM_ND_ARRAY Examples

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ff_summ_nd_array** from the **MEconTools Package**. This function summarizes policy and value functions over states.

2.1.1 Test FF_SUMM_ND_ARRAY Defaults

Call the function with defaults.

```
ff_summ_nd_array();
```

```
xxx  Summ over (a,z), condi age as cols, kids/marriage as rows  xxxxxxxxxxxxxxxxxxxxxxxxxx
    group    marry    kids    mean_age_18    mean_age_19    mean_age_20    mean_age_21
    _____    _____    ____    ___________    ___________    ___________    ___________

      1        0        1        0.52456        0.51689        0.48412        0.54526
      2        1        1        0.49355        0.52906         0.5583        0.47342
      3        0        2        0.49085        0.51315        0.45158        0.43201
      4        1        2        0.58096        0.50596        0.47985        0.58791
      5        0        3        0.57811         0.6068        0.55221        0.50677
      6        1        3        0.53023        0.49258        0.48728        0.43352
      7        0        4        0.50339        0.48449        0.53618        0.45993
      8        1        4        0.44418         0.5223        0.55657        0.48583
```

2.1.2 Test FF_SUMM_ND_ARRAY with Random 2 Dimensional Matrix

Summarize over 6 dimensional array, iteratively change how many dimensions to group over.

First, generate matrix:

```
st_title = "Random 2D dimensional Array Testing Summarizing";
rng(123)
mn_polval = rand(5,4);
bl_print_table = true;
ar_st_stats = ["mean"];
cl_mp_datasetdesc = {};
cl_mp_datasetdesc{1} = containers.Map({'name', 'labval'}, ...
    {'a', linspace(0,1,size(mn_polval,1))});
cl_mp_datasetdesc{2} = containers.Map({'name', 'labval'}, ...
    {'z', linspace(-1,1,size(mn_polval,2))});
disp(mn_polval);
```

```
    0.6965    0.4231    0.3432    0.7380
    0.2861    0.9808    0.7290    0.1825
    0.2269    0.6848    0.4386    0.1755
    0.5513    0.4809    0.0597    0.5316
    0.7195    0.3921    0.3980    0.5318
```

Second, show the entire matrix (no labels):

```
it_aggd = 0;
bl_row = 1;
ff_summ_nd_array(st_title, mn_polval, bl_print_table, ar_st_stats, it_aggd, bl_row);

xxx  Random 2D dimensional Array Testing Summarizing  xxxxxxxxxxxxxxxxxxxxxxxxxx
    group    vardim2    mean_vardim1_1    mean_vardim1_2    mean_vardim1_3    mean_vardim1_4    mean
    -----    -------    --------------    --------------    --------------    --------------    ----

      1          1          0.69647           0.28614           0.22685            0.55131          0
      2          2          0.42311           0.98076           0.68483            0.48093          0
      3          3          0.34318           0.72905           0.43857           0.059678          0
      4          4            0.738           0.18249           0.17545            0.53155          0
```

Third, rotate row and column, and now with labels:

```
it_aggd = 0;
bl_row = 1;
ar_permute = [2,1];
ff_summ_nd_array(st_title, mn_polval, bl_print_table, ar_st_stats, it_aggd, bl_row, ...
    cl_mp_datasetdesc, ar_permute);

xxx  Random 2D dimensional Array Testing Summarizing  xxxxxxxxxxxxxxxxxxxxxxxxxx
    group     a      mean_z__1    mean_z__0_33333    mean_z_0_33333    mean_z_1
    -----    ----    ---------    ---------------    --------------    --------

      1         0      0.69647          0.42311            0.34318         0.738
      2      0.25      0.28614          0.98076            0.72905       0.18249
      3       0.5      0.22685          0.68483            0.43857       0.17545
      4      0.75      0.55131          0.48093           0.059678       0.53155
      5         1      0.71947          0.39212            0.39804       0.53183
```

Fourth, dimension one as columns, average over dim 2:

```
it_aggd = 1;
bl_row = 1;
ff_summ_nd_array(st_title, mn_polval, bl_print_table, ar_st_stats, it_aggd, bl_row, ...
    cl_mp_datasetdesc);

xxx  Random 2D dimensional Array Testing Summarizing  xxxxxxxxxxxxxxxxxxxxxxxxxx
    group    x    mean_z__1    mean_z__0_33333    mean_z_0_33333    mean_z_1
    -----    -    ---------    ---------------    --------------    --------

      1      1     0.49605          0.59235             0.3937        0.43186
```

Fifth, dimension one as rows, average over dim 2:

```
it_aggd = 1;
bl_row = 0;
ff_summ_nd_array(st_title, mn_polval, bl_print_table, ar_st_stats, it_aggd, bl_row, ...
    cl_mp_datasetdesc);

xxx  Random 2D dimensional Array Testing Summarizing  xxxxxxxxxxxxxxxxxxxxxxxxxx
    group       z         sum       mean        std      coefvari      min         max
```

1	-1	2.4802	0.49605	0.22895	2.1666	0.22685	0.71947
2	-0.33333	2.9617	0.59235	0.24524	2.4154	0.39212	0.98076
3	0.33333	1.9685	0.3937	0.23907	1.6468	0.059678	0.72905
4	1	2.1593	0.43186	0.24575	1.7573	0.17545	0.738

Sixth, dimension two as rows, average over dim 1:

```
ar_permute = [2,1];
it_aggd = 1;
bl_row = 0;
ff_summ_nd_array(st_title, mn_polval, bl_print_table, ar_st_stats, it_aggd, bl_row, ...
    cl_mp_datasetdesc, ar_permute);
```

```
xxx  Random 2D dimensional Array Testing Summarizing  xxxxxxxxxxxxxxxxxxxxxxxxx
    group     a       sum       mean        std      coefvari      min         max
    -----    ----    ------    -------    -------    --------    --------    -------

      1        0     2.2007    0.55019    0.19636     2.8019      0.34318      0.738
      2      0.25    2.1784    0.54461    0.37514     1.4518      0.18249    0.98076
      3       0.5    1.5257    0.38143    0.23212     1.6432      0.17545    0.68483
      4      0.75    1.6235    0.40587    0.23269     1.7443     0.059678    0.55131
      5        1     2.0415    0.51036    0.15361     3.3226      0.39212    0.71947
```

2.1.3 Test FF_SUMM_ND_ARRAY with Random 6 Dimensional Matrix

Summarize over 6 dimensional array, iteratively change how many dimensions to group over.

First, generate matrix:

```
st_title = "Random ND dimensional Array Testing Summarizing";
rng(123)
mn_polval = rand(8,7,6,5,4,3);
bl_print_table = true;
ar_st_stats = ["mean"];
```

Second, summarize over the first four dimensions, row group others:

```
it_aggd = 4;
bl_row = 0;
ff_summ_nd_array(st_title, mn_polval, bl_print_table, ar_st_stats, it_aggd, bl_row);
```

```
xxx  Random ND dimensional Array Testing Summarizing  xxxxxxxxxxxxxxxxxxxxxxxxxx
    group    vardim5    vardim6     sum       mean        std      coefvari       min          max
    -----    -------    -------    ------    -------    -------    --------    ----------    -------

      1         1          1       836.78    0.49808    0.29255     1.7026     8.1888e-05    0.99964
      2         2          1       842.15    0.50128    0.28968     1.7305     6.7838e-05    0.99936
      3         3          1       831.45    0.49491    0.28851     1.7154     0.00091373    0.99989
      4         4          1        843.9    0.50232    0.28154     1.7842     0.00012471    0.99731
      5         1          2       838.99     0.4994     0.2911     1.7156     0.00029749    0.99938
      6         2          2       830.81    0.49453    0.28634     1.7271     0.00027113     0.9992
      7         3          2       832.59    0.49559    0.28682     1.7279     0.00035994    0.99936
      8         4          2       820.42    0.48835    0.29032     1.6821     0.00096259    0.99896
      9         1          3       870.56    0.51819    0.29111     1.7801      0.0010616    0.99951
     10         2          3       854.68    0.50874    0.28458     1.7877       0.001884    0.99965
     11         3          3       838.29    0.49898     0.2891      1.726       0.0019192    0.99945
     12         4          3       842.83    0.50169     0.2877     1.7438     0.00016871    0.99963
```

Third, summarize over the first four dimensions, column group 5th, and row group others:

```
it_aggd = 4;
bl_row = 1;
ff_summ_nd_array(st_title, mn_polval, bl_print_table, ["sum"], it_aggd, bl_row);

xxx  Random ND dimensional Array Testing Summarizing  xxxxxxxxxxxxxxxxxxxxxxxxxxx
    group    vardim6    sum_vardim5_1    sum_vardim5_2    sum_vardim5_3    sum_vardim5_4
    -----    -------    -------------    -------------    -------------    -------------

      1          1          836.78           842.15           831.45            843.9
      2          2          838.99           830.81           832.59           820.42
      3          3          870.56           854.68           838.29           842.83
```

Fourth, summarize over the first five dimensions, column group 6th, no row groups:

```
it_aggd = 5;
bl_row = 1;
ff_summ_nd_array(st_title, mn_polval, bl_print_table, ["mean", "std"], it_aggd, bl_row);

xxx  Random ND dimensional Array Testing Summarizing  xxxxxxxxxxxxxxxxxxxxxxxxxxx
    group    x    mean_vardim6_1    mean_vardim6_2    mean_vardim6_3    std_vardim6_1    std_vardim6
    -----    -    --------------    --------------    --------------    -------------    -----------

      1      1       0.49915           0.49447            0.5069           0.28805          0.28862
```

Fifth, summarize over all six dimensions, summary statistics over the entire dataframe:

```
it_aggd = 6;
bl_row = 0;
ff_summ_nd_array(st_title, mn_polval, bl_print_table, ar_st_stats, it_aggd, bl_row);

xxx  Random ND dimensional Array Testing Summarizing  xxxxxxxxxxxxxxxxxxxxxxxxxxx
    group    x     sum      mean        std      coefvari       min          max
    -----    -    -----    -------    -------    --------    ----------    -------

      1      1    10083    0.50017    0.28831     1.7349     6.7838e-05    0.99989
```

2.1.4 Test FF_SUMM_ND_ARRAY with Random 7 Dimensional Matrix with All Parameters

Given a random seven dimensional matrix, average over the 2nd, 4th and 5th dimensionals. Show as row groups the 3, 6 and 7th dimensions, and row groups the 1st dimension. Show Coefficient of Variation only.

```
st_title = "avg VALUE 2+4+5th dims. groups 3+6+7th dims, and row groups the 1st dim.";
rng(123)
mn_polval = rand(3,10,2,10,10,2,3);
ar_permute = [2,4,5,1,3,6,7];
bl_print_table = true;
ar_st_stats = ["coefvari"];
it_aggd = 3; % mean over 3 dims
bl_row = 1; % one var for row group
cl_mp_datasetdesc = {};
cl_mp_datasetdesc{1} = containers.Map({'name', 'labval'}, ...
    {'age', [18, 19, 20]});
cl_mp_datasetdesc{2} = containers.Map({'name', 'labval'}, ...
    {'savings', linspace(0,1,10)});
cl_mp_datasetdesc{3} = containers.Map({'name', 'labval'}, ...
    {'borrsave', [-1,+1]});
cl_mp_datasetdesc{4} = containers.Map({'name', 'labval'}, ...
    {'shocka', linspace(-5,5,10)});
cl_mp_datasetdesc{5} = containers.Map({'name', 'labval'}, ...
```

```
    {'shockb', linspace(-5,5,10)});
cl_mp_datasetdesc{6} = containers.Map({'name', 'labval'}, ...
    {'marry', [0,1]});
cl_mp_datasetdesc{7} = containers.Map({'name', 'labval'}, ...
    {'region', [1,2,3]});
% call function
ff_summ_nd_array(st_title, mn_polval, bl_print_table, ar_st_stats, it_aggd, bl_row, cl_mp_datasetdes

xxx  avg VALUE 2+4+5th dims. groups 3+6+7th dims, and row groups the 1st dim.  xxxxxxxxxxxxxxxxxxxx
    group    borrsave    marry    region    cv_age_18    cv_age_19    cv_age_20
    -----    --------    -----    ------    ---------    ---------    ---------

      1         -1         0        1        1.7607       1.7534       1.7065
      2          1         0        1        1.6566       1.7501       1.7042
      3         -1         1        1        1.6608       1.7658       1.7291
      4          1         1        1         1.756       1.7479       1.7606
      5         -1         0        2        1.7314       1.7506        1.786
      6          1         0        2        1.7347        1.728        1.738
      7         -1         1        2        1.7811        1.755       1.7568
      8          1         1        2        1.7445       1.7398       1.7746
      9         -1         0        3        1.7025       1.7286         1.69
     10          1         0        3          1.74       1.7549       1.7356
     11         -1         1        3        1.7147       1.7287       1.7341
     12          1         1        3        1.7919       1.7313       1.7452
```

Chapter 3

Distributional Analysis

3.1 FF_SIMU_STATS Examples

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ff_simu_stats** from the **MEconTools Package.** This is a gate-way function that computes mean, percentiles, covariance etc between several variables.

3.1.1 Test FF_SIMU_STATS Defaults

Call the function with defaults.

```
ff_simu_stats();
```

```
xxx tb_outcomes: all stats xxx
    OriginalVariableNames     cl_mt_pol_a    cl_mt_pol_c
    ----------------------    -----------    -----------

    {'mean'              }     -0.11081         8.8423
    {'sd'                }       4.1239         6.5845
    {'coefofvar'         }      -37.215        0.74466
    {'min'               }           -7        -6.3772
    {'max'               }            9         21.786
    {'pYis0'             }     0.064259              0
    {'pYls0'             }      0.54867       0.027329
    {'pYgr0'             }      0.38707        0.97267
    {'pYisMINY'          }     0.051764       0.015232
    {'pYisMAXY'          }     0.027329       0.046484
    {'p1'                }           -7        -6.3772
    {'p10'               }           -6        0.27238
    {'p25'               }           -3         5.2138
    {'p50'               }           -1         6.5321
    {'p75'               }            3         13.799
    {'p90'               }            5         16.887
    {'p99'               }            9         21.786
    {'fl_cov_cl_mt_pol_a'}       17.007        -22.084
    {'fl_cor_cl_mt_pol_a'}            1       -0.81327
    {'fl_cov_cl_mt_pol_c'}      -22.084         43.356
    {'fl_cor_cl_mt_pol_c'}     -0.81327              1
    {'fracByP1'          }       3.2699      -0.010985
    {'fracByP10'         }       5.9889      -0.013362
    {'fracByP25'         }       14.165       0.041007
    {'fracByP50'         }       16.208         0.1893
```

```
    {'fracByP75'         }       12.702         0.59539
    {'fracByP90'         }       6.6611          0.8307
    {'fracByP99'         }            1               1
```

3.1.2 Test FF_SIMU_STATS Four States-Points Matrix

Over some (a,z) states that is 3 by 3, c matrix, generate all stats

```
% Set Parameters
mt_x_of_s = [1, 2,  3.0;...
             3, 1,  1.5;...
             4, 3,  2.0];
mt_y_of_s = [2, -10, 9.0;...
             5, 1.1,3.0;...
             1, 3,  -1.5];
mt_z_of_s = [1.1, 2,3.3;...
             2.3, 1,1.5;...
             4, 2.5,2.0];
mp_cl_mt_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
mp_cl_mt_xyz_of_s('cl_mt_x_of_s') = {mt_x_of_s, zeros(1)};
mp_cl_mt_xyz_of_s('cl_mt_y_of_s') = {mt_y_of_s, zeros(1)};
mp_cl_mt_xyz_of_s('cl_mt_z_of_s') = {mt_z_of_s, zeros(1)};
mp_cl_mt_xyz_of_s('ar_st_y_name') = ["cl_mt_x_of_s", "cl_mt_y_of_s", "cl_mt_z_of_s"];
% Mass
rng(123);
mt_f_of_s = rand(size(mt_x_of_s));
mt_f_of_s = mt_f_of_s/sum(mt_f_of_s, 'all');
% Call Function
mp_cl_mt_xyz_of_s_out = ff_simu_stats(mt_f_of_s, mp_cl_mt_xyz_of_s);

xxx tb_outcomes: all stats xxx
     OriginalVariableNames     cl_mt_x_of_s    cl_mt_y_of_s    cl_mt_z_of_s
    ______________________    ____________    ____________    ____________

    {'mean'               }        2.0763          1.9323          2.0668
    {'sd'                 }        0.9071          5.2239          0.9042
    {'coefofvar'          }       0.43688          2.7034         0.43749
    {'min'                }             1             -10               1
    {'max'                }             4               9               4
    {'pYis0'              }             0               0               0
    {'pYls0'              }             0         0.20441               0
    {'pYgr0'              }             1         0.79559               1
    {'pYisMINY'           }       0.28039         0.10917         0.14247
    {'pYisMAXY'           }      0.044922         0.19422        0.044922
    {'p1'                 }             1             -10               1
    {'p10'                }             1             -10               1
    {'p25'                }             1             1.1             1.1
    {'p50'                }             2               2               2
    {'p75'                }             3               5             2.5
    {'p90'                }             3               9             3.3
    {'p99'                }             4               9               4
    {'fl_cov_cl_mt_x_of_s'}       0.82282           1.589         0.78646
    {'fl_cor_cl_mt_x_of_s'}             1         0.33534         0.95887
    {'fl_cov_cl_mt_y_of_s'}         1.589          27.289          1.8353
    {'fl_cor_cl_mt_y_of_s'}       0.33534               1         0.38856
    {'fl_cov_cl_mt_z_of_s'}       0.78646          1.8353         0.81758
    {'fl_cor_cl_mt_z_of_s'}       0.95887         0.38856               1
    {'fracByP1'           }       0.13504        -0.56498        0.068934
    {'fracByP10'          }       0.13504        -0.56498        0.068934
```

```
    {'fracByP25'          }       0.13504        -0.53456         0.14234
    {'fracByP50'          }       0.42991        -0.39181         0.43856
    {'fracByP75'          }       0.91346        0.095425         0.60296
    {'fracByP90'          }       0.91346               1         0.91306
    {'fracByP99'          }             1               1               1
```

3.1.3 Test FF_SIMU_STATS Four States-Points Matrix Single Column Inputs

Same as before, but now inputs are single column, should have identical results:

```
% Array Inputs
mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
mp_cl_mt_xyz_of_s('cl_mt_x_of_s') = {mt_x_of_s(:), zeros(1)};
mp_cl_mt_xyz_of_s('cl_mt_y_of_s') = {mt_y_of_s(:), zeros(1)};
mp_cl_mt_xyz_of_s('cl_mt_z_of_s') = {mt_z_of_s(:), zeros(1)};
mp_cl_mt_xyz_of_s('ar_st_y_name') = ["cl_mt_x_of_s", "cl_mt_y_of_s", "cl_mt_z_of_s"];
% Call Function
mp_cl_mt_xyz_of_s_out = ff_simu_stats(mt_f_of_s(:), mp_cl_mt_xyz_of_s);

xxx tb_outcomes: all stats xxx
     OriginalVariableNames     cl_mt_x_of_s    cl_mt_y_of_s    cl_mt_z_of_s
    ______________________    ____________    ____________    ____________

    {'mean'               }        2.0763          1.9323          2.0668
    {'sd'                 }        0.9071          5.2239          0.9042
    {'coefofvar'          }       0.43688          2.7034         0.43749
    {'min'                }             1             -10               1
    {'max'                }             4               9               4
    {'pYis0'              }             0               0               0
    {'pYls0'              }             0         0.20441               0
    {'pYgr0'              }             1         0.79559               1
    {'pYisMINY'           }       0.28039         0.10917         0.14247
    {'pYisMAXY'           }      0.044922         0.19422        0.044922
    {'p1'                 }             1             -10               1
    {'p10'                }             1             -10               1
    {'p25'                }             1             1.1             1.1
    {'p50'                }             2               2               2
    {'p75'                }             3               5             2.5
    {'p90'                }             3               9             3.3
    {'p99'                }             4               9               4
    {'fl_cov_cl_mt_x_of_s'}       0.82282           1.589         0.78646
    {'fl_cor_cl_mt_x_of_s'}             1         0.33534         0.95887
    {'fl_cov_cl_mt_y_of_s'}         1.589          27.289          1.8353
    {'fl_cor_cl_mt_y_of_s'}       0.33534               1         0.38856
    {'fl_cov_cl_mt_z_of_s'}       0.78646          1.8353         0.81758
    {'fl_cor_cl_mt_z_of_s'}       0.95887         0.38856               1
    {'fracByP1'           }       0.13504        -0.56498        0.068934
    {'fracByP10'          }       0.13504        -0.56498        0.068934
    {'fracByP25'          }       0.13504        -0.53456         0.14234
    {'fracByP50'          }       0.42991        -0.39181         0.43856
    {'fracByP75'          }       0.91346        0.095425         0.60296
    {'fracByP90'          }       0.91346               1         0.91306
    {'fracByP99'          }             1               1               1
```

3.1.4 Test FF_SIMU_STATS Print Many Details

The Same As before, but now control which percentiles and other details to display.

```
% Array Inputs
mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
mp_cl_ar_xyz_of_s('cl_ar_x_of_s') = {mt_x_of_s(:), zeros(1)};
mp_cl_ar_xyz_of_s('cl_ar_z_of_s') = {mt_z_of_s(:), zeros(1)};
mp_cl_ar_xyz_of_s('ar_st_y_name') = ["cl_ar_x_of_s", "cl_ar_z_of_s"];

% controls
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('bl_display_detail') = false;
mp_support('bl_display_final') = true;
mp_support('bl_display_drvm2outcomes') = false;
mp_support('ar_fl_percentiles') = [25 50 75];
mp_support('bl_display_drvstats') = true;
mp_support('bl_display_drvm2covcor') = false;

% Call Function
mp_cl_mt_xyz_of_s_out = ff_simu_stats(mt_f_of_s(:), mp_cl_ar_xyz_of_s, mp_support);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Summary Statistics for: cl_ar_x_of_s
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
----------------------------------------
fl_choice_mean
    2.0763

fl_choice_sd
    0.9071

fl_choice_coefofvar
    0.4369

fl_choice_prob_zero
     0

fl_choice_prob_below_zero
     0

fl_choice_prob_above_zero
     1

fl_choice_prob_max
    0.0449

tb_disc_cumu
    cl_ar_x_of_sDiscreteVal    cl_ar_x_of_sDiscreteValProbMass     CDF      cumsumFrac
    -----------------------    -------------------------------    ------    ----------

                1                          0.28039                28.039     0.13504
              1.5                          0.13561                 41.6      0.23301
                2                          0.20441                62.041     0.42991
                3                          0.33466                95.508     0.91346
                4                         0.044922                   100           1

    cl_ar_x_of_sDiscreteVal    cl_ar_x_of_sDiscreteValProbMass     CDF      cumsumFrac
    -----------------------    -------------------------------    ------    ----------

                1                          0.28039                28.039     0.13504
```

```
            1.5                        0.13561                41.6     0.23301
              2                        0.20441              62.041     0.42991
              3                        0.33466              95.508     0.91346
              4                       0.044922                 100           1

tb_prob_drv
    percentiles    cl_ar_x_of_sDiscreteValPercentileValues    fracOfSumHeldBelowThisPercentile
    -----------    ---------------------------------------    --------------------------------

        25                            1                                 0.13504
        50                            2                                 0.42991
        75                            3                                 0.91346

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Summary Statistics for: cl_ar_z_of_s
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
----------------------------------------
fl_choice_mean
    2.0668

fl_choice_sd
    0.9042

fl_choice_coefofvar
    0.4375

fl_choice_prob_zero
     0

fl_choice_prob_below_zero
     0

fl_choice_prob_above_zero
     1

fl_choice_prob_max
    0.0449

tb_disc_cumu
    cl_ar_z_of_sDiscreteVal    cl_ar_z_of_sDiscreteValProbMass     CDF      cumsumFrac
    -----------------------    -------------------------------    ------    ----------

              1                          0.14247                14.247     0.068934
            1.1                          0.13792                28.039      0.14234
            1.5                          0.13561                  41.6      0.24076
              2                          0.20441                62.041      0.43856
            2.3                          0.056663               67.708      0.50162
            2.5                          0.083786               76.086      0.60296
            3.3                           0.19422               95.508      0.91306
              4                         0.044922                  100           1

    cl_ar_z_of_sDiscreteVal    cl_ar_z_of_sDiscreteValProbMass     CDF      cumsumFrac
    -----------------------    -------------------------------    ------    ----------

              1                          0.14247                14.247     0.068934
            1.1                          0.13792                28.039      0.14234
            1.5                          0.13561                  41.6      0.24076
```

```
             2                    0.20441             62.041      0.43856
           2.3                   0.056663             67.708      0.50162
           2.5                   0.083786             76.086      0.60296
           3.3                    0.19422             95.508      0.91306
             4                   0.044922                100            1

tb_prob_drv
    percentiles    cl_ar_z_of_sDiscreteValPercentileValues    fracOfSumHeldBelowThisPercentile
    -----------    ---------------------------------------    --------------------------------

        25                          1.1                                0.14234
        50                            2                                0.43856
        75                          2.5                                0.60296

xxx tb_outcomes: all stats xxx
     OriginalVariableNames     cl_ar_x_of_s    cl_ar_z_of_s
    ----------------------    ------------    ------------

    {'mean'               }        2.0763          2.0668
    {'sd'                 }        0.9071          0.9042
    {'coefofvar'          }       0.43688         0.43749
    {'min'                }             1               1
    {'max'                }             4               4
    {'pYis0'              }             0               0
    {'pYls0'              }             0               0
    {'pYgr0'              }             1               1
    {'pYisMINY'           }       0.28039         0.14247
    {'pYisMAXY'           }      0.044922        0.044922
    {'p25'                }             1             1.1
    {'p50'                }             2               2
    {'p75'                }             3             2.5
    {'fl_cov_cl_ar_x_of_s'}       0.82282         0.78646
    {'fl_cor_cl_ar_x_of_s'}             1         0.95887
    {'fl_cov_cl_ar_z_of_s'}       0.78646         0.81758
    {'fl_cor_cl_ar_z_of_s'}       0.95887               1
    {'fracByP25'          }       0.13504         0.14234
    {'fracByP50'          }       0.42991         0.43856
    {'fracByP75'          }       0.91346         0.60296
```

3.2 FF_DISC_RAND_VAR_STATS Examples

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ff_disc_rand_var_stats** from the **MEconTools Package.** This function summarizes statistics of matrixes stored in a container map, as well as scalar, string, function and other values stored in container maps.

3.2.1 Test FF_DISC_RAND_VAR_STATS Defaults

Call the function with defaults.

```
ff_disc_rand_var_stats();
```

```
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Summary Statistics for: binom
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
----------------------------------------
```

```
fl_choice_mean
   -1.0000

fl_choice_sd
    2.5100

fl_choice_coefofvar
   -2.5100

fl_choice_prob_zero
    0.1416

fl_choice_prob_below_zero
    0.5888

fl_choice_prob_above_zero
    0.2696

fl_choice_prob_max
   2.0589e-16

tb_disc_cumu
    binomDiscreteVal    binomDiscreteValProbMass       CDF       cumsumFrac
    ----------------    ------------------------    ---------    ----------

          -10                 2.2539e-05           0.0022539    0.00022539
           -9                 0.00028979            0.031233     0.0028335
           -8                  0.0018008             0.21132       0.01724
           -7                  0.0072034             0.93166      0.067664
           -6                   0.020838              3.0155       0.19269
           -5                    0.04644              7.6595       0.42489
           -4                   0.082928              15.952       0.75661
           -3                    0.12185              28.138        1.1222
           -2                    0.15014              43.152        1.4224
           -1                    0.15729              58.881        1.5797

    binomDiscreteVal    binomDiscreteValProbMass    CDF    cumsumFrac
    ----------------    ------------------------    ---    ----------

           11                 6.0392e-06           100         1
           12                 1.0588e-06           100         1
           13                 1.5784e-07           100         1
           14                  1.973e-08           100         1
           15                 2.0293e-09           100         1
           16                 1.6725e-10           100         1
           17                 1.0619e-11           100         1
           18                 4.8762e-13           100         1
           19                 1.4412e-14           100         1
           20                 2.0589e-16           100         1

tb_prob_drv
    percentiles    binomDiscreteValPercentileValues    fracOfSumHeldBelowThisPercentile
    -----------    --------------------------------    --------------------------------

        0.1                       -8                                0.01724
          1                       -6                                0.19269
          5                       -5                                0.42489
         10                       -4                                0.75661
```

15	-4	0.75661
20	-3	1.1222
25	-3	1.1222
35	-2	1.4224
50	-1	1.5797
65	0	1.5797
75	1	1.4694
80	1	1.4694
85	2	1.3197
90	2	1.3197
95	3	1.1865
99	5	1.0412
99.9	7	1.0052

3.2.2 Test FF_DISC_RAND_VAR_STATS 0 and 1 Random Variable

The simplest discrete random variable has two values, zero or one. The probability of zero is 30 percent, and 70 percent is the probability of one.

```
% Parameters
% 1. specify the random variable
st_var_name = 'bernoulli';
ar_choice_unique_sorted = [0, 1];
ar_choice_prob = [0.3, 0.7];
% 2. percentiles of interest
ar_fl_percentiles = [0.1 5 25 50 75 95 99.9];
% 3. print resutls
bl_display_drvstats = true;
% Call Function
[ds_stats_map] = ff_disc_rand_var_stats(st_var_name, ...
    ar_choice_unique_sorted, ar_choice_prob, ...
    ar_fl_percentiles, bl_display_drvstats);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Summary Statistics for: bernoulli
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
----------------------------------------
fl_choice_mean
    0.7000

fl_choice_sd
    0.4583

fl_choice_coefofvar
    0.6547

fl_choice_prob_zero
    0.3000

fl_choice_prob_below_zero
     0

fl_choice_prob_above_zero
    0.7000

fl_choice_prob_max
    0.7000
```

```
tb_disc_cumu
    bernoulliDiscreteVal    bernoulliDiscreteValProbMass    CDF    cumsumFrac
    --------------------    ----------------------------    ---    ----------

              0                           0.3                30        0
              1                           0.7               100        1

    bernoulliDiscreteVal    bernoulliDiscreteValProbMass    CDF    cumsumFrac
    --------------------    ----------------------------    ---    ----------

              0                           0.3                30        0
              1                           0.7               100        1

tb_prob_drv
    percentiles    bernoulliDiscreteValPercentileValues    fracOfSumHeldBelowThisPercentile
    -----------    ------------------------------------    --------------------------------

        0.1                         0                                  0
          5                         0                                  0
         25                         0                                  0
         50                         1                                  1
         75                         1                                  1
         95                         1                                  1
       99.9                         1                                  1
```

3.2.3 Test FF_DISC_RAND_VAR_STATS with Poisson

Poisson random variable, with mean equals to ten, summarize over umsymmetric percentiles. Note that the poisson random variable has no upper bound.

```
% Parameters
% 1. specify the random variable
st_var_name = 'poisson';
mu = 10;
ar_choice_unique_sorted = 0:1:50;
ar_choice_prob = poisspdf(ar_choice_unique_sorted, mu);
% 2. percentiles of interest, unsymmetric
ar_fl_percentiles = [0.1 5 10 25 50 90 95 99 99.9 99.99 99.999 99.9999];
% 3. print resutls
bl_display_drvstats = true;
% Call Function
[ds_stats_map] = ff_disc_rand_var_stats(st_var_name, ...
    ar_choice_unique_sorted, ar_choice_prob, ...
    ar_fl_percentiles, bl_display_drvstats);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Summary Statistics for: poisson
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
----------------------------------------
fl_choice_mean
    10

fl_choice_sd
    3.1623

fl_choice_coefofvar
    0.3162
```

```
fl_choice_prob_zero
   4.5400e-05

fl_choice_prob_below_zero
     0

fl_choice_prob_above_zero
    1.0000

fl_choice_prob_max
   1.4927e-19

tb_disc_cumu
    poissonDiscreteVal    poissonDiscreteValProbMass      CDF      cumsumFrac
    ------------------    --------------------------    -------    ----------

            0                     4.54e-05             0.00454           0
            1                     0.000454             0.04994     4.54e-05
            2                      0.00227             0.27694    0.0004994
            3                    0.0075667              1.0336    0.0027694
            4                     0.018917              2.9253     0.010336
            5                     0.037833              6.7086     0.029253
            6                     0.063055              13.014     0.067086
            7                     0.090079              22.022      0.13014
            8                       0.1126              33.282      0.22022
            9                      0.12511              45.793      0.33282

    poissonDiscreteVal    poissonDiscreteValProbMass    CDF    cumsumFrac
    ------------------    --------------------------    ---    ----------

            41                    1.3571e-13            100        1
            42                    3.2313e-14            100        1
            43                    7.5146e-15            100        1
            44                    1.7079e-15            100        1
            45                    3.7953e-16            100        1
            46                    8.2506e-17            100        1
            47                    1.7554e-17            100        1
            48                    3.6572e-18            100        1
            49                    7.4636e-19            100        1
            50                    1.4927e-19            100        1

tb_prob_drv
    percentiles    poissonDiscreteValPercentileValues    fracOfSumHeldBelowThisPercentile
    -----------    ----------------------------------    --------------------------------

         0.1                       2                                0.0004994
           5                       5                                 0.029253
          10                       6                                 0.067086
          25                       8                                  0.22022
          50                      10                                  0.45793
          90                      14                                  0.86446
          95                      15                                  0.91654
          99                      18                                  0.98572
        99.9                      21                                  0.99841
       99.99                      24                                  0.99988
      99.999                      26                                  0.99998
         100                      28                                        1
```

```
% Print out full Stored Matrix
% Note that the outputs are single row arrays.
ff_container_map_display(ds_stats_map, 100, 100)

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: ds_stats_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                               i    idx    ndim    numel    rowN    colN     mean       std      coe
                               -    ---    ----    -----    ----    ----    -------    ------    ---

    ar_choice_perc_fracheld    1     1      2       12       1       12     0.62833     0.435    0.6
    ar_choice_percentiles      2     2      2       12       1       12       14.75    8.7399    0.5
    ar_fl_percentiles          3     3      2       12       1       12      64.499    42.887    0.6

xxx TABLE:ar_choice_perc_fracheld xxxxxxxxxxxxxxxxx
              c1           c2           c3          c4          c5          c6          c7          c8
          ---------    --------    --------    -------    -------    -------    -------    -------

    r1    0.0004994    0.029253    0.067086    0.22022    0.45793    0.86446    0.91654    0.98572

xxx TABLE:ar_choice_percentiles xxxxxxxxxxxxxxxxx
          c1    c2    c3    c4    c5    c6    c7    c8    c9    c10    c11    c12
          --    --    --    --    --    --    --    --    --    ---    ---    ---

    r1    2     5     6     8     10    14    15    18    21    24     26     28

xxx TABLE:ar_fl_percentiles xxxxxxxxxxxxxxxxx
          c1     c2    c3    c4    c5    c6    c7    c8     c9      c10      c11      c12
          ---    --    --    --    --    --    --    --    ----    -----    ------    ---

    r1    0.1    5     10    25    50    90    95    99    99.9    99.99    99.999    100

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: ds_stats_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                                 i     idx      value
                                 --    ---    ----------

    fl_choice_coefofvar           1     4        0.31623
    fl_choice_max                 2     5             50
    fl_choice_mean                3     6             10
    fl_choice_min                 4     7              0
    fl_choice_prob_above_zero     5     8        0.99995
    fl_choice_prob_below_zero     6     9              0
    fl_choice_prob_max            7    10     1.4927e-19
    fl_choice_prob_min            8    11       4.54e-05
    fl_choice_prob_zero           9    12       4.54e-05
    fl_choice_sd                 10    13         3.1623
```

3.3 FF_DISC_RAND_VAR_MASS2OUTCOMES Examples

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ff_disc_rand_var_mass2outcomes** from the **MEconTools Package.** This function generates sorted discrete random variable from state-space joint distri-

bution.

3.3.1 Test FF_DISC_RAND_VAR_MASS2OUTCOMES Defaults

Call the function with defaults.

```
ff_disc_rand_var_mass2outcomes();

INPUT f(a,z): mt_dist_bystates
    0.0289    0.0465    0.0228    0.0036    0.0001
    0.0241    0.0930    0.0857    0.0241    0.0015
    0.0080    0.0744    0.1285    0.0643    0.0074
    0.0013    0.0297    0.0964    0.0857    0.0186
    0.0001    0.0059    0.0361    0.0571    0.0232
    0.0000    0.0005    0.0054    0.0152    0.0116

INPUT y(a,z): mt_choice_bystates
    -5    -4    -5    -4    -4
    -3    -2    -3    -2    -3
    -1    -1    -1     0     0
     1     1     2     3     1
     4     3     3     4     3
     5     6     5     6     6

OUTPUT f(y): ar_choice_prob_byY
    0.0518
    0.0502
    0.1113
    0.1171
    0.2109
    0.0717
    0.0497
    0.0964
    0.1510
    0.0572
    0.0054
    0.0273

OUTPUT f(y,z): mt_choice_prob_byYZ
    0.0289         0    0.0228         0         0
         0    0.0465         0    0.0036    0.0001
    0.0241         0    0.0857         0    0.0015
         0    0.0930         0    0.0241         0
    0.0080    0.0744    0.1285         0         0
         0         0         0    0.0643    0.0074
    0.0013    0.0297         0         0    0.0186
         0         0    0.0964         0         0
         0    0.0059    0.0361    0.0857    0.0232
    0.0001         0         0    0.0571         0
    0.0000         0    0.0054         0         0
         0    0.0005         0    0.0152    0.0116

OUTPUT f(y,a): mt_choice_prob_byYA
    0.0518         0         0         0         0         0
    0.0502         0         0         0         0         0
         0    0.1113         0         0         0         0
         0    0.1171         0         0         0         0
         0         0    0.2109         0         0         0
         0         0    0.0717         0         0         0
```

```
         0         0         0    0.0497         0         0
         0         0         0    0.0964         0         0
         0         0         0    0.0857    0.0653         0
         0         0         0         0    0.0572         0
         0         0         0         0         0    0.0054
         0         0         0         0         0    0.0273

OUTPUT f(y) and y in table: tb_choice_drv_cur_byY
    binomtestOutcomes    probMassFunction
    -----------------    ----------------

           -5                0.051764
           -4                0.050217
           -3                 0.11126
           -2                 0.11706
           -1                 0.21092
            0                0.071696
            1                0.049682
            2                0.096388
            3                 0.15102
            4                0.057231
            5               0.0054256
            6                0.027329
```

3.3.2 Test FF_DISC_RAND_VAR_MASS2OUTCOMES Four States-Points

Over some (a,z) states that is 2 by 2, matrix or vectorized inputs identical results.

```
% Set Parameters
st_y_name = 'consumption';
% consumption matrix: c(a,z)
mt_c_of_s = [1,2;3,1];
% stationary mass over assets adn shocks: f(a,z)
mt_f_of_s = rand(size(mt_c_of_s));
mt_f_of_s = mt_f_of_s/sum(mt_f_of_s, 'all');
% Call Function
[ar_f_of_y, ar_y_unique_sorted] = ...
    ff_disc_rand_var_mass2outcomes(st_y_name, mt_c_of_s, mt_f_of_s);
% print
disp([ar_f_of_y ar_y_unique_sorted]);

    0.4039    1.0000
    0.2971    2.0000
    0.2990    3.0000
```

Same as before, but now inputs are single column:

```
% Call Function
[ar_f_of_y, ar_y_unique_sorted] = ...
    ff_disc_rand_var_mass2outcomes(st_y_name, mt_c_of_s(:), mt_f_of_s);
disp([ar_f_of_y ar_y_unique_sorted]);

    0.4039    1.0000
    0.2971    2.0000
    0.2990    3.0000
```

3.3.3 Test FF_DISC_RAND_VAR_MASS2OUTCOMES Conditional Mass Outputs

Same inputs as before, but now, also output additional conditional statistis, f(y, a), where a is the row state variable for f(a,z). For conditional statistics, must provide matrix based inputs.

```
% Set Parameters
st_y_name = 'consumption';
% consumption matrix: c(a,z)
mt_c_of_s = [1,2,0.5;
             3,1,2.0];
% stationary mass over assets adn shocks: f(a,z)
mt_f_of_s = rand(size(mt_c_of_s));
mt_f_of_s = mt_f_of_s/sum(mt_f_of_s, 'all');
% Call Function
[ar_f_of_y, ar_y_unique_sorted, mt_f_of_y_srow, mt_f_of_y_scol] = ...
    ff_disc_rand_var_mass2outcomes(st_y_name, mt_c_of_s, mt_f_of_s);
% print
disp([ar_f_of_y ar_y_unique_sorted]);

    0.2695    0.5000
    0.3765    1.0000
    0.2649    2.0000
    0.0891    3.0000

disp(mt_f_of_y_srow);

    0.2695         0
    0.1215    0.2550
    0.1217    0.1432
         0    0.0891

disp(mt_f_of_y_scol);

         0         0    0.2695
    0.1215    0.2550         0
         0    0.1217    0.1432
    0.0891         0         0
```

3.4 FF_DISC_RAND_VAR_MASS2COVCOR Examples

> Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ff_disc_rand_var_mass2covcor** from the **MEconTools Package.** This function calculates covariance and correlation based for two discrete random variables.

3.4.1 Test FF_DISC_RAND_VAR_MASS2COVCOR Defaults

Call the function with defaults.

```
ff_disc_rand_var_mass2covcor();

-----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: covvar_input_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                i    idx    ndim    numel    rowN    colN      mean        std       coefvari
                -    ---    ----    -----    ----    ----    --------    --------    --------    --
```

```
    mt_f_of_s    1     5      2       30       6       5      0.033333    0.035743     1.0723     3.
    mt_x_of_s    2     6      2       30       6       5       0.83333      5.3051     6.3661
    mt_y_of_s    3     7      2       30       6       5        8.3259      7.1913    0.86373

xxx TABLE:mt_f_of_s xxxxxxxxxxxxxxxxx
              c1            c2           c3          c4          c5
          ----------    ----------    ---------    ---------    ---------

    r1      0.028917      0.046484     0.022848    0.0036146     0.000119
    r2      0.024097      0.092967     0.085679     0.024097    0.0014875
    r3     0.0080324      0.074374      0.12852     0.064259    0.0074374
    r4     0.0013387       0.02975     0.096388     0.085679     0.018593
    r5    0.00011156     0.0059499     0.036146     0.057119     0.023242
    r6    3.7187e-06    0.00047599    0.0054218     0.015232     0.011621

xxx TABLE:mt_x_of_s xxxxxxxxxxxxxxxxx
          c1    c2    c3    c4    c5
          --    --    --    --    --

    r1    -7    -6    -7    -6    -6
    r2    -5    -3    -5    -3    -4
    r3    -2    -1    -1     0    -1
    r4     2     2     3     4     2
    r5     6     5     5     6     5
    r6     8     9     7     9     9

xxx TABLE:mt_y_of_s xxxxxxxxxxxxxxxxx
            c1          c2          c3         c4          c5
          ------    --------    ------    -------    -------

    r1    13.231      21.786    18.136      19.35     13.901
    r2     9.946      16.887    9.6914      15.71     8.6906
    r3    16.255      6.2166    13.799     5.2138     11.641
    r4    12.628      2.7525    6.5321    0.27238     13.357
    r5    5.8844      4.0352      6.05    0.14102    0.50318
    r6    3.5617    -0.72091    5.1855    -6.3772    -4.4805

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: covvar_input_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                i    idx     value
                -    ---    --------

    fl_x_mean   1     1     -0.11081
    fl_x_sd     2     2       4.1239
    fl_y_mean   3     3       8.8423
    fl_y_sd     4     4       6.5845

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: covvar_output_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                                i    idx    ndim    numel    rowN    colN      mean       std
                                -    ---    ----    -----    ----    ----    --------    ------

    mt_cov_component_weighted   1     1      2       30       6       5      -0.73612    1.0404
```

```
    mt_x_devi_from_mean          2     2      2       30       6       5       0.94415    5.3051
    mt_x_y_multiply              3     3      2       30       6       5       -31.321    36.564
    mt_y_devi_from_mean          4     4      2       30       6       5      -0.51644    7.1913

xxx TABLE:mt_cov_component_weighted xxxxxxxxxxxxxxxxx
              c1            c2           c3          c4           c5
          -----------    ---------    --------    ---------    ----------

    r1       -0.87434      -3.5432     -1.4628     -0.22368    -0.0035451
    r2       -0.13003      -2.1607    -0.35565     -0.47814    0.00087767
    r3       -0.11248      0.17365    -0.56642    -0.025838     -0.018507
    r4       0.010697     -0.38241    -0.69273      -3.0184       0.17717
    r5     -0.0020165     -0.14618    -0.51584      -3.0371      -0.99056
    r6    -0.00015927    -0.041473    -0.14098      -2.1121       -1.4106

xxx TABLE:mt_x_devi_from_mean xxxxxxxxxxxxxxxxx
            c1          c2          c3          c4          c5
          -------    --------    --------    -------    --------

    r1    -6.8892     -5.8892     -6.8892    -5.8892     -5.8892
    r2    -4.8892     -2.8892     -4.8892    -2.8892     -3.8892
    r3    -1.8892    -0.88919    -0.88919    0.11081    -0.88919
    r4     2.1108      2.1108      3.1108     4.1108      2.1108
    r5     6.1108      5.1108      5.1108     6.1108      5.1108
    r6     8.1108      9.1108      7.1108     9.1108      9.1108

xxx TABLE:mt_x_y_multiply xxxxxxxxxxxxxxxxx
            c1          c2          c3          c4          c5
          -------    -------    -------    --------    -------

    r1    -30.237    -76.225    -64.023     -61.882    -29.792
    r2     -5.396    -23.242     -4.151     -19.842    0.59004
    r3    -14.003     2.3348    -4.4073    -0.40209    -2.4884
    r4     7.9905    -12.854    -7.1868      -35.23     9.5287
    r5    -18.075    -24.568    -14.271     -53.172     -42.62
    r6     -42.83    -87.129    -26.003     -138.66    -121.38

xxx TABLE:mt_y_devi_from_mean xxxxxxxxxxxxxxxxx
            c1          c2          c3          c4          c5
          -------    -------    -------    -------    --------

    r1      4.389      12.943     9.2933      10.508      5.0587
    r2     1.1037      8.0444    0.84902      6.8677    -0.15171
    r3     7.4123    -2.6258      4.9566    -3.6286      2.7985
    r4     3.7855    -6.0898    -2.3103      -8.57      4.5142
    r5    -2.9579    -4.8071    -2.7924    -8.7013     -8.3392
    r6    -5.2806    -9.5633    -3.6568     -15.22     -13.323

fl_cov
  -22.0835

fl_cor
   -0.8133
```

3.4.2 Test FF_DISC_RAND_VAR_MASS2COVCOR Four States-Points

Over some (a,z) states that is 2 by 2, c matrix, and y matrix, find correlation. Positively related.

```
% Set Parameters
```

```
mt_c_of_s = [1,2;3,1];
mt_y_of_s = [2,10;5,1.1];
rng(123);
mt_f_of_s = rand(size(mt_c_of_s));
mt_f_of_s = mt_f_of_s/sum(mt_f_of_s, 'all');
bl_display_drvm2covcor = false;
% Call Function
[fl_cov_xy, fl_cor_xy] = ff_disc_rand_var_mass2covcor(...
    mt_c_of_s, mt_y_of_s, mt_f_of_s, bl_display_drvm2covcor);
display(['cov=' num2str(fl_cov_xy) ',cor=', num2str(fl_cor_xy)]);

cov=1.4446,cor=0.65723
```

Same as before, but now inputs are single column:

```
% Call Function
[fl_cov_xy, fl_cor_xy] = ff_disc_rand_var_mass2covcor(...
    mt_c_of_s(:), mt_y_of_s(:), mt_f_of_s(:), bl_display_drvm2covcor);
display(['cov=' num2str(fl_cov_xy) ',cor=', num2str(fl_cor_xy)]);

cov=1.4446,cor=0.65723
```

3.4.3 Test FF_DISC_RAND_VAR_MASS2COVCOR Two Random Vectors

Generate two random vectors, with random or even mass, correlation should be zero:

```
% Set Parameters
rng(4567);
mt_c_of_s = rand([20,1])*100;
mt_y_of_s = rand([20,1])*100;
mt_f_of_s = rand(size(mt_c_of_s));
mt_f_of_s = mt_f_of_s/sum(mt_f_of_s, 'all');
bl_display_drvm2covcor = false;
% Call Function
[fl_cov_xy, fl_cor_xy] = ff_disc_rand_var_mass2covcor(...
    mt_c_of_s, mt_y_of_s, mt_f_of_s, bl_display_drvm2covcor);
display(['cov=' num2str(fl_cov_xy) ',cor=', num2str(fl_cor_xy)]);

cov=-57.6533,cor=-0.062023
```

3.4.4 Test FF_DISC_RAND_VAR_MASS2COVCOR Provide Mean and SD

Same as above, but now provide means and sd for x andy directly. The results are the same as when mean and sd are calculated inside the function.

```
% Set Parameters
rng(4567);
mt_c_of_s = rand([20,1])*100;
mt_y_of_s = rand([20,1])*100;
mt_f_of_s = rand(size(mt_c_of_s));
mt_f_of_s = mt_f_of_s/sum(mt_f_of_s, 'all');
fl_c_mean = sum(mt_f_of_s.*mt_c_of_s);
fl_c_sd = sqrt(sum(mt_f_of_s.*(mt_c_of_s-fl_c_mean).^2));
fl_y_mean = sum(mt_f_of_s.*mt_y_of_s);
fl_y_sd = sqrt(sum(mt_f_of_s.*(mt_y_of_s-fl_y_mean).^2));
bl_display_drvm2covcor = false;
% Call Function
[fl_cov_xy, fl_cor_xy] = ff_disc_rand_var_mass2covcor(...
```

```
    mt_c_of_s, mt_y_of_s, mt_f_of_s, ...
    fl_c_mean, fl_c_sd, ...
    fl_y_mean, fl_y_sd, bl_display_drvm2covcor);
display(['cov=' num2str(fl_cov_xy) ',cor=', num2str(fl_cor_xy)]);

cov=-57.6533,cor=-0.062023
```

Chapter 4

Graphs

4.1 FF_GRAPH_GRID Examples: X, Y and Color Line Plots

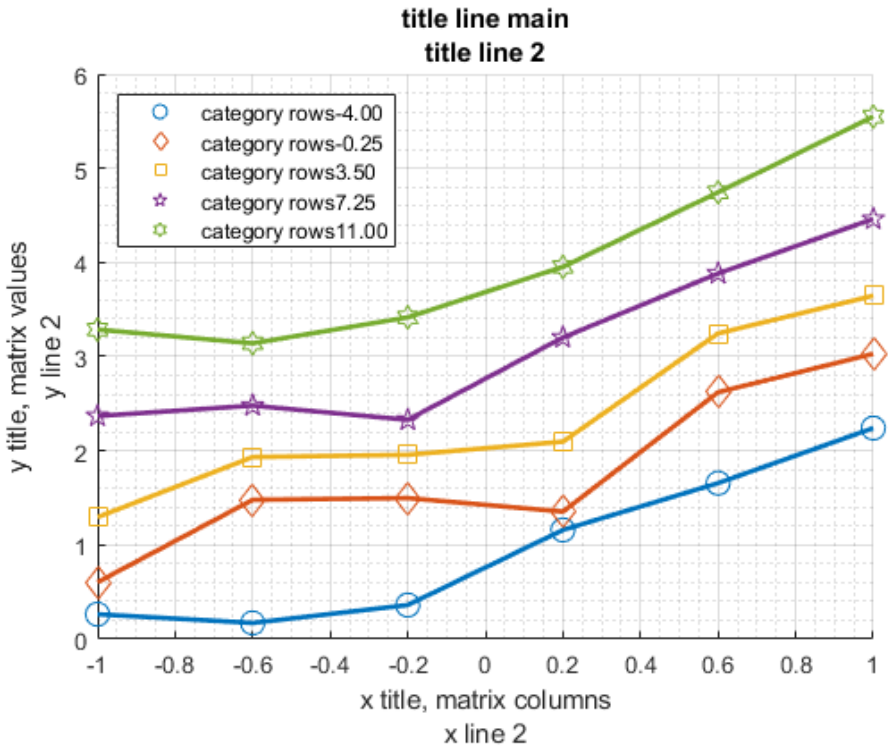
Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

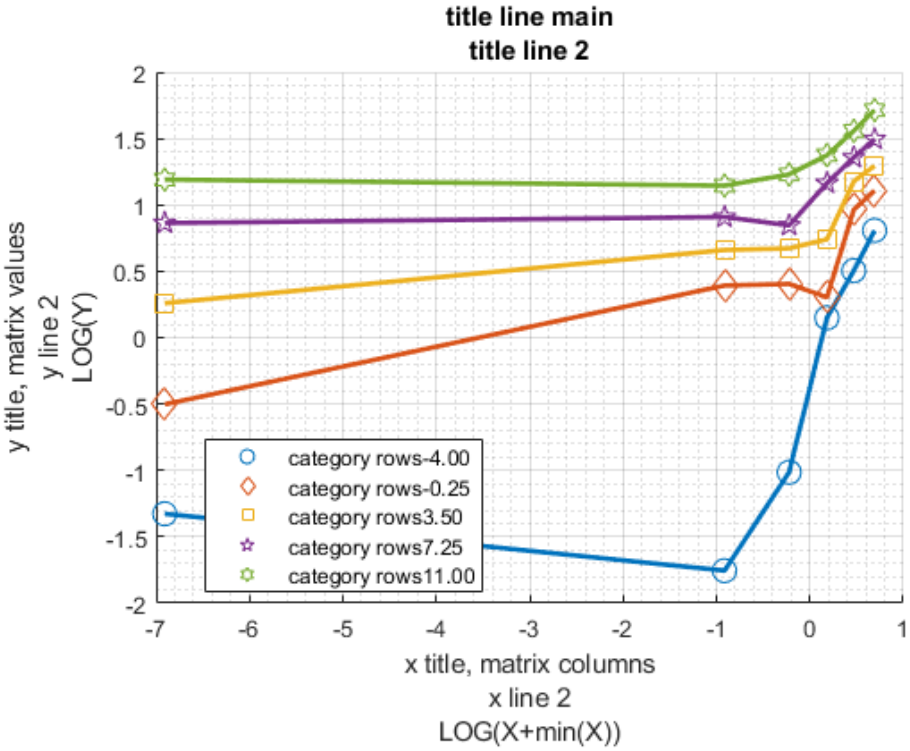
This is the example vignette for function: **ff_graph_grid** from the **MEconTools Package.** This function can graph out value and policy functions given one state vector (x-axis), conditional on other states (line groups). Can handle a few lines (scatter + lines), or many groups (jet spectrum).

4.1.1 Test FF_GRAPH_GRID Defaults

Call the function with defaults.

```
ff_graph_grid();
```

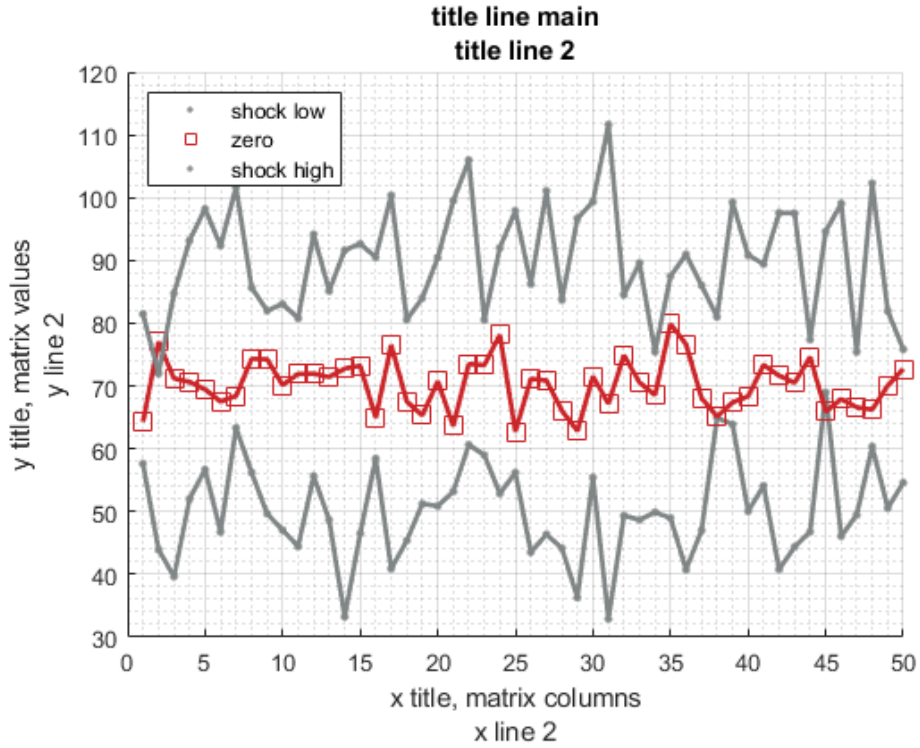


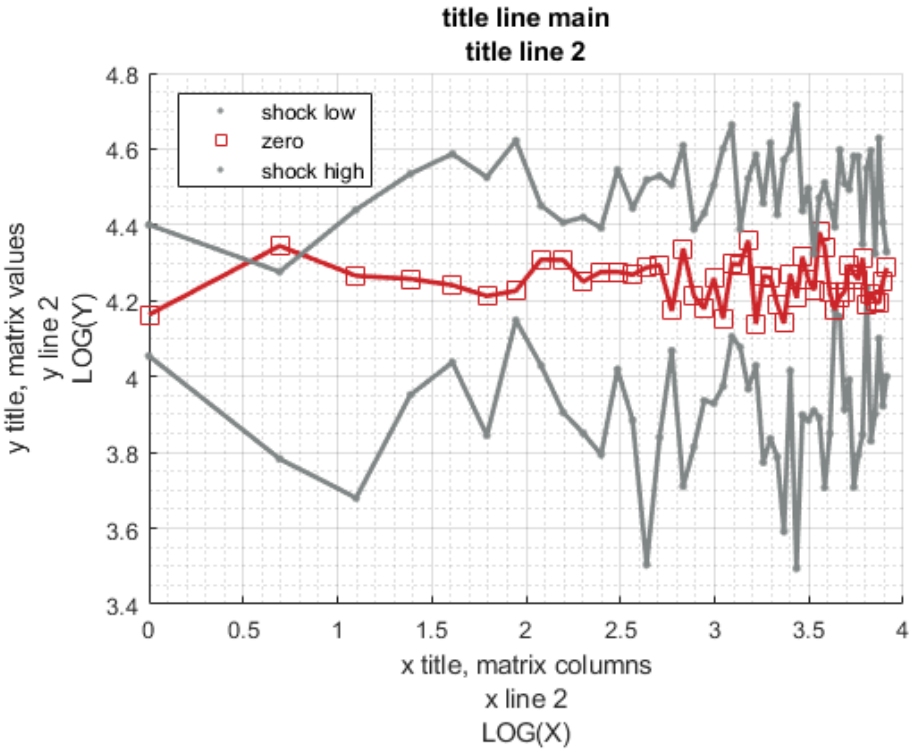


4.1.2 Test FF_GRAPH_GRID Random Matrix Pick Markers and Colors

Call the function with defaults.

```
rng(123);
mt_value = [normrnd(50,10,[1, 50]); ...
            normrnd(70,5,[1, 50]);...
            normrnd(90,10,[1, 50])];
ar_row_grid = ["shock low", "zero", "shock high"];
ar_col_grid = 1:50;
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('cl_scatter_shapes') = { '.', 's' ,'.' };
mp_support_graph('cl_colors') = {'gray', 'red', 'gray'};
ff_graph_grid(mt_value, ar_row_grid, ar_col_grid, mp_support_graph);
```



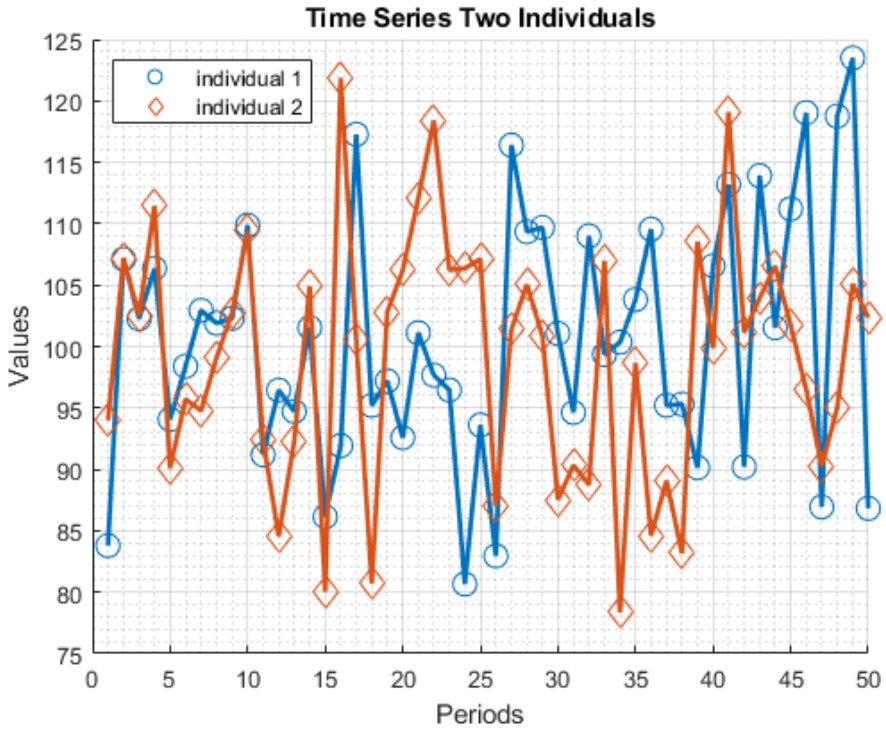


4.1.3 Test FF_GRAPH_GRID Two Random Normal Lines and Labels

There are two autoregressive time series, plot out the time two time series.

```
% Generate the two time series
rng(456);
mt_value = normrnd(100,10,[2, 50]);
ar_row_grid = ["individual 1", "individual 2"];
ar_col_grid = 1:50;
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
```

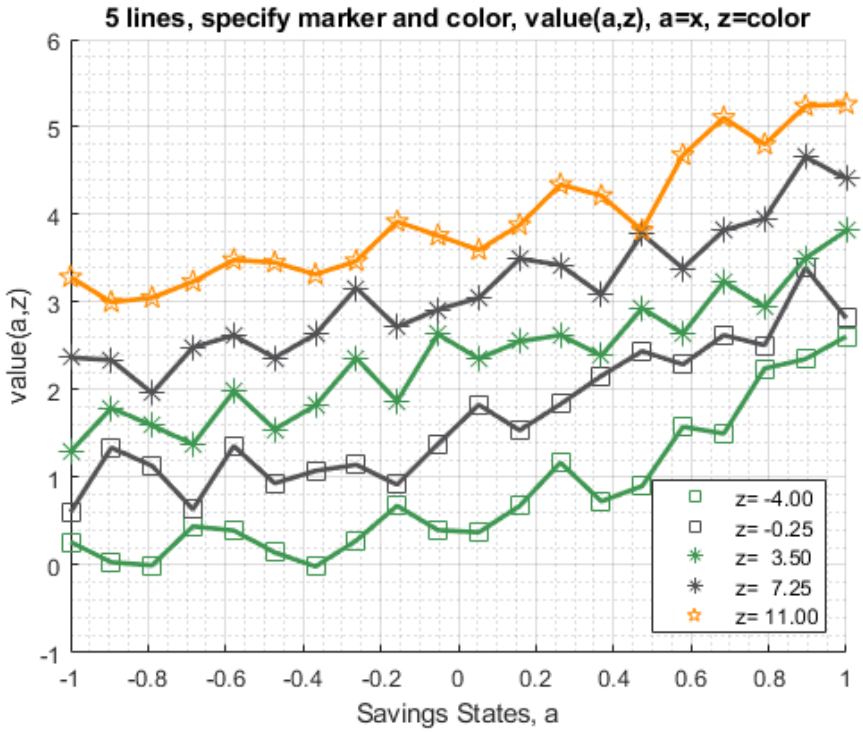
```
mp_support_graph('cl_st_graph_title') = {'Time Series Two Individuals'};
mp_support_graph('cl_st_ytitle') = {'Values'};
mp_support_graph('cl_st_xtitle') = {'Periods'};
mp_support_graph('bl_graph_logy') = false; % do not log
ff_graph_grid(mt_value, ar_row_grid, ar_col_grid, mp_support_graph);
```



4.1.4 Test FF_GRAPH_GRID 6 Lines Pick Marker and Colors

Plot many lines, with auto legend.

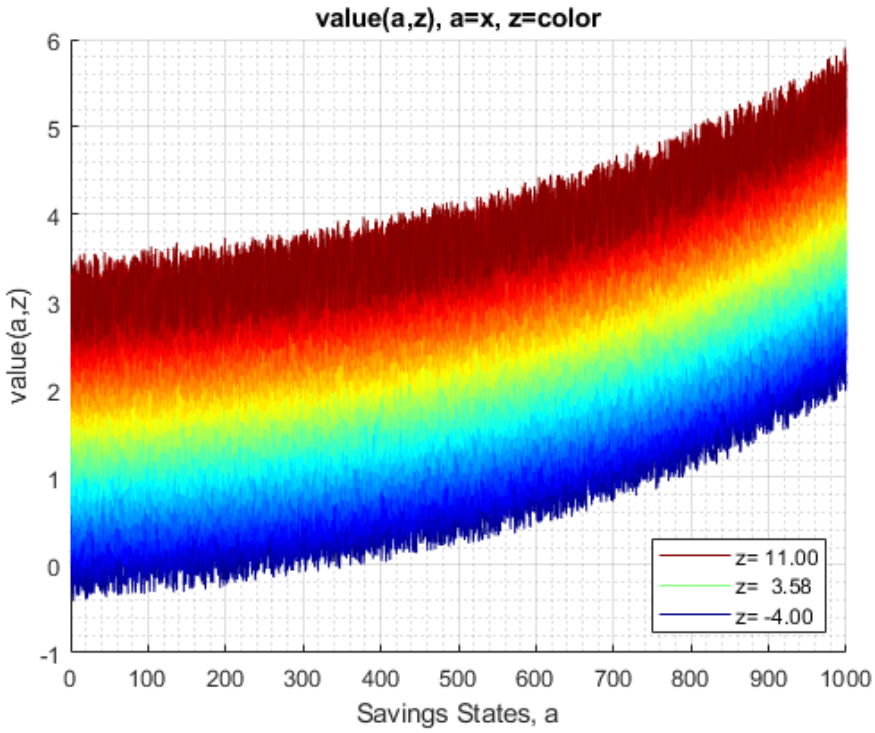
```
% Generate some Data
rng(456);
ar_row_grid = linspace(-4, 11, 5);
ar_col_grid = linspace(-1, 1, 20);
rng(123);
mt_value = 0.2*ar_row_grid' + exp(ar_col_grid) + rand([length(ar_row_grid), length(ar_col_grid)]);
% container map settings
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('cl_st_graph_title') = {'5 lines, specify marker and color, value(a,z), a=x, z=colo
mp_support_graph('cl_st_ytitle') = {'value(a,z)'};
mp_support_graph('cl_st_xtitle') = {'Savings States, a'};
mp_support_graph('st_legend_loc') = 'southeast';
mp_support_graph('bl_graph_logy') = false; % do not log
mp_support_graph('st_rowvar_name') = 'z=';
mp_support_graph('it_legend_select') = 3; % how many shock legends to show
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {'s', 's', '*', '*', 'p'};
mp_support_graph('cl_colors') = {'green', 'black', 'green', 'black', 'orange'};
% Call function
ff_graph_grid(mt_value, ar_row_grid, ar_col_grid, mp_support_graph);
```



4.1.5 Test FF_GRAPH_GRID Many Lines

Plot many lines, with auto legend.

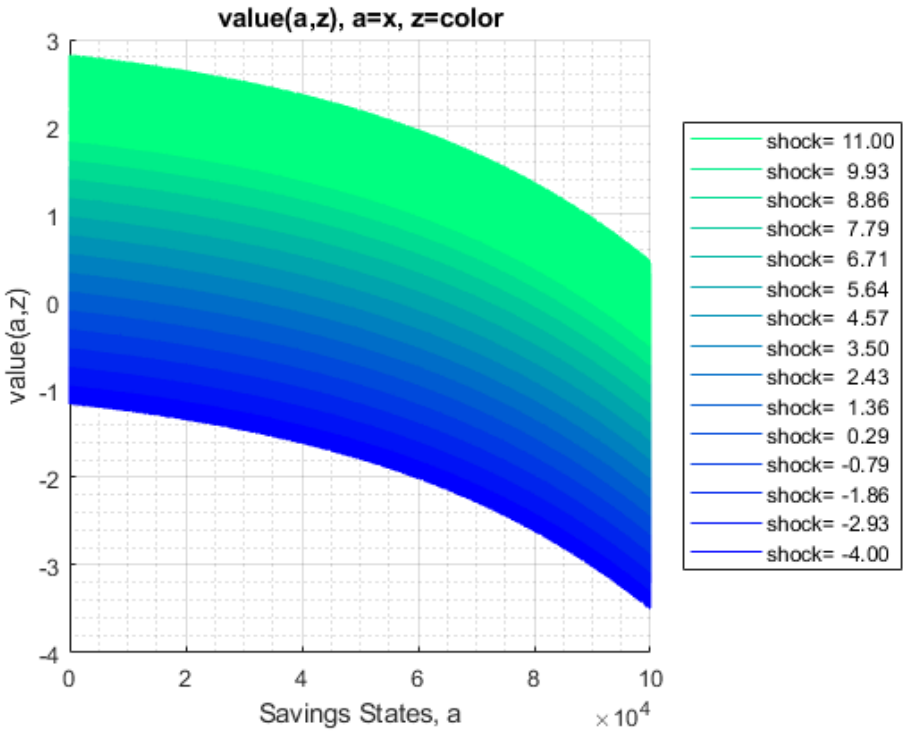
```
% Generate some Data
rng(456);
ar_row_grid = linspace(-4, 11, 100);
ar_col_grid = linspace(-1, 1, 1000);
rng(123);
mt_value = 0.2*ar_row_grid' + exp(ar_col_grid) + rand([length(ar_row_grid), length(ar_col_grid)]);
% container map settings
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('cl_st_graph_title') = {'value(a,z), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'value(a,z)'};
mp_support_graph('cl_st_xtitle') = {'Savings States, a'};
mp_support_graph('st_legend_loc') = 'southeast';
mp_support_graph('bl_graph_logy') = false; % do not log
mp_support_graph('st_rowvar_name') = 'z=';
mp_support_graph('it_legend_select') = 3; % how many shock legends to show
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_colors') = 'jet'; % any predefined matlab colormap
% Call function
ff_graph_grid(mt_value, ar_row_grid, ar_col_grid, mp_support_graph);
```



4.1.6 Test FF_GRAPH_GRID Many Lines Legend Exogenous

Plot many lines, exogenously set legend

```
% Generate the two time series
rng(456);
ar_row_grid = linspace(-4, 11, 15);
ar_col_grid = linspace(-1, 1, 100000);
rng(123);
mt_value = 0.2*ar_row_grid' - exp(ar_col_grid) + rand([length(ar_row_grid), length(ar_col_grid)]);
% setting shock vector name exogenously here
ar_row_grid = string(num2str(ar_row_grid', "shock=%6.2f"));
% container map settings
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('cl_st_graph_title') = {'value(a,z), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'value(a,z)'};
mp_support_graph('cl_st_xtitle') = {'Savings States, a'};
mp_support_graph('st_legend_loc') = 'eastoutside';
mp_support_graph('bl_graph_logy') = false; % do not log
mp_support_graph('it_legend_select') = 15;
mp_support_graph('cl_colors') = 'winter'; % any predefined matlab colormap
% Call function
ff_graph_grid(mt_value, ar_row_grid, ar_col_grid, mp_support_graph);
```


value(a,z), a=x, z=color
value(a,z)
Savings States, a
×10^4
shock= 11.00
shock= 9.93
shock= 8.86
shock= 7.79
shock= 6.71
shock= 5.64
shock= 4.57
shock= 3.50
shock= 2.43
shock= 1.36
shock= 0.29
shock= -0.79
shock= -1.86
shock= -2.93
shock= -4.00

Chapter 5

Data Structures

5.1 FF_SAVEBORR_GRID Example for Generating Asset Grid

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ff_saveborr_grid** from the **MEconTools Package.** This function generates variously spaced savings/borrowing states/choices grid.

5.1.1 Test FF_SAVEBORR_GRID Defaults

Call the function with defaults.

```
ff_saveborr_grid();
```

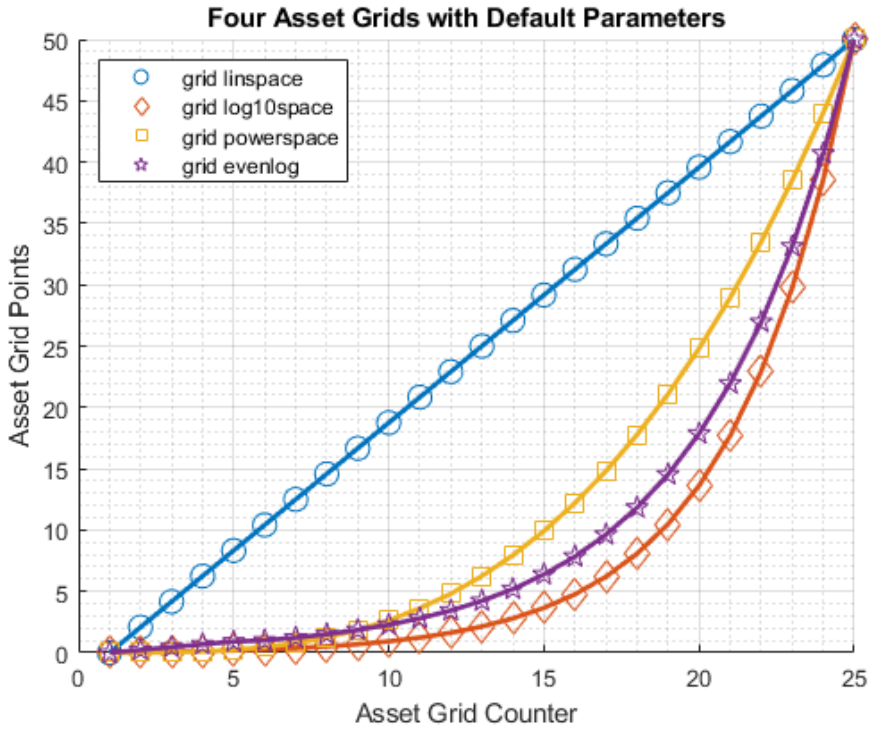
```
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                      i    idx    ndim    numel    rowN    colN     sum     mean      std      coefv
                      -    ---    ----    -----    ----    ----    -----    -----    ------    -----

    ar_fl_saveborr    1     1      2       25       25      1      216.7    8.668    13.363     1.54

xxx TABLE:ar_fl_saveborr xxxxxxxxxxxxxxxxxx
              c1
           --------

    r1           0
    r2     0.029558
    r3     0.067855
    r4      0.11748
    r5      0.18177
    r6      0.26507
    r7      0.37301
    r8      0.51286
    r9      0.69407
    r10     0.92885
    r11      1.2331
    r12      1.6272
    r13      2.1379
    r14      2.7996
```

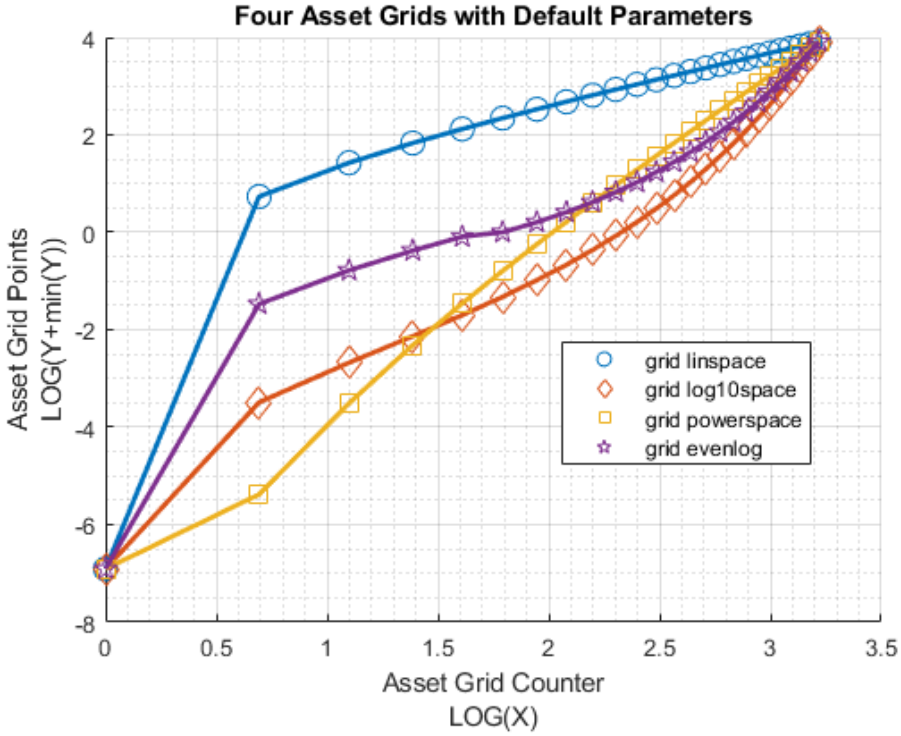
```
    r15       3.657
    r16      4.7679
    r17      6.2072
    r18      8.0722
    r19      10.489
    r20       13.62
    r21      17.676
    r22      22.932
    r23      29.743
    r24      38.567
    r25          50

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                               i    idx    value
                               -    ---    -----

    grid_evenlog_threshold     1     2        1
    grid_log10space_x1         2     3      0.3
    grid_log10space_x2         3     4        3
    grid_powerspace_power      4     5        3
```

5.1.2 Test FF_SAVEBORR_GRID Default Linear Grid, Log Grid, Power Grid, Threshold Grid

Call the function with defaults.

```
% Same min and max and grid points
[fl_a_min, fl_a_max, it_a_points] = deal(0,50,25);
% Four types of grid points
st_grid_type = 'grid_linspace';
[ar_fl_saveborr_linspace] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type);
st_grid_type = 'grid_log10space';
[ar_fl_saveborr_log10space] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type);
st_grid_type = 'grid_powerspace';
[ar_fl_saveborr_powerspace] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type);
st_grid_type = 'grid_evenlog';
[ar_fl_saveborr_evenlog] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type);
% draw four types of lines jointly
mt_value = [ar_fl_saveborr_linspace'; ar_fl_saveborr_log10space'; ...
    ar_fl_saveborr_powerspace'; ar_fl_saveborr_evenlog'];
ar_row_grid = ["grid linspace", "grid log10space", "grid powerspace", "grid evenlog"];
ar_col_grid = 1:it_a_points;
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('cl_st_graph_title') = {'Four Asset Grids with Default Parameters'};
mp_support_graph('cl_st_ytitle') = {'Asset Grid Points'};
mp_support_graph('cl_st_xtitle') = {'Asset Grid Counter'};
mp_support_graph('bl_graph_logy') = true; % do not log
ff_graph_grid(mt_value, ar_row_grid, ar_col_grid, mp_support_graph);
```



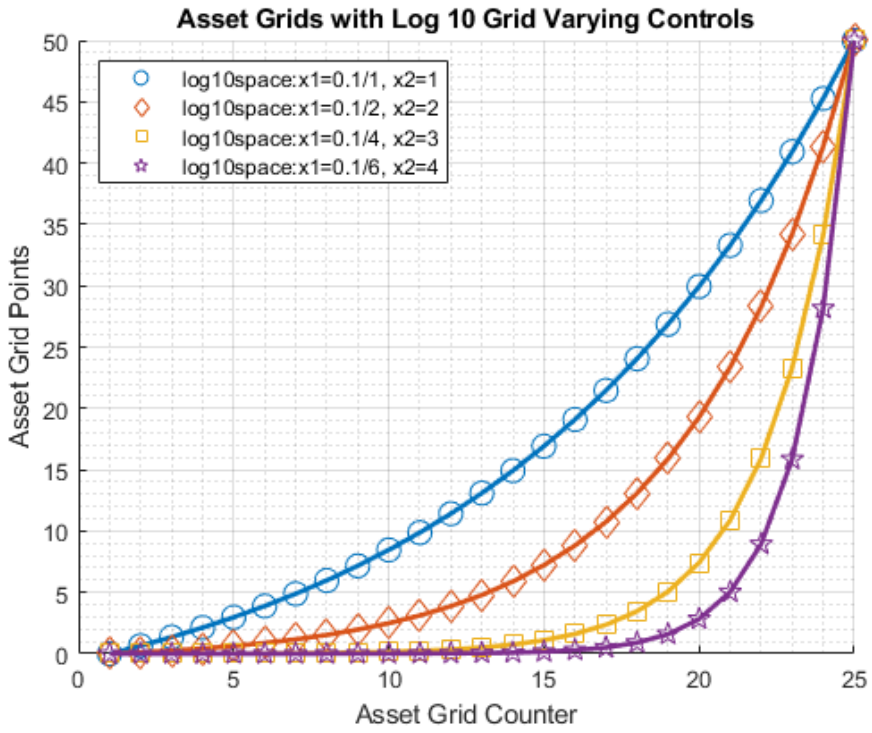


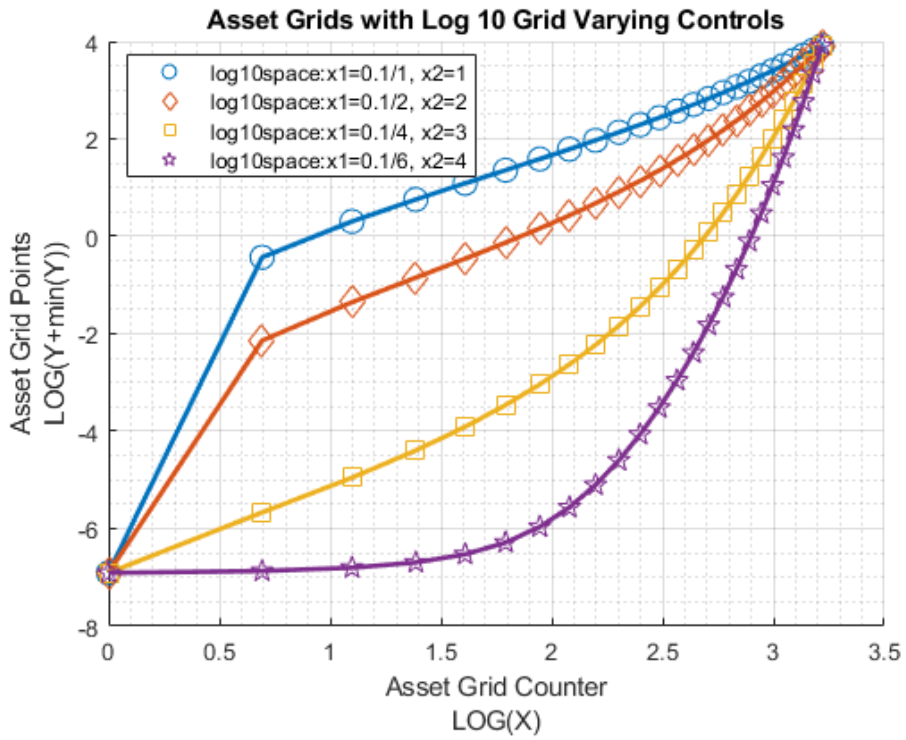
5.1.3 Test FF_SAVEBORR_GRID Log Grid Changing Parameters

Log grid, same min and max, change log X1 and X2 points

```
% Same min and max and grid points
[fl_a_min, fl_a_max, it_a_points] = deal(0,50,25);
st_grid_type = 'grid_log10space';
% Four types of grid points
mp_grid_control = containers.Map('KeyType','char', 'ValueType','any');
mp_grid_control('grid_log10space_x1') = 0.1;
```

```
mp_grid_control('grid_log10space_x2') = 1;
[ar_fl_log10space_a] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_contr
mp_grid_control('grid_log10space_x1') = 0.1/2;
mp_grid_control('grid_log10space_x2') = 1*2;
[ar_fl_log10space_b] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_contr
mp_grid_control('grid_log10space_x1') = 0.1/4;
mp_grid_control('grid_log10space_x2') = 1*4;
[ar_fl_log10space_c] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_contr
mp_grid_control('grid_log10space_x1') = 0.1/6;
mp_grid_control('grid_log10space_x2') = 1*6;
[ar_fl_log10space_d] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_contr
% draw four types of lines jointly
mt_value = [ar_fl_log10space_a'; ar_fl_log10space_b'; ...
    ar_fl_log10space_c'; ar_fl_log10space_d'];
ar_row_grid = [...
    "log10space:x1=0.1/1, x2=1", ...
    "log10space:x1=0.1/2, x2=2", ...
    "log10space:x1=0.1/4, x2=3", ...
    "log10space:x1=0.1/6, x2=4"];
ar_col_grid = 1:it_a_points;
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('cl_st_graph_title') = {'Asset Grids with Log 10 Grid Varying Controls'};
mp_support_graph('cl_st_ytitle') = {'Asset Grid Points'};
mp_support_graph('cl_st_xtitle') = {'Asset Grid Counter'};
mp_support_graph('bl_graph_logy') = true; % do not log
ff_graph_grid(mt_value, ar_row_grid, ar_col_grid, mp_support_graph);
```

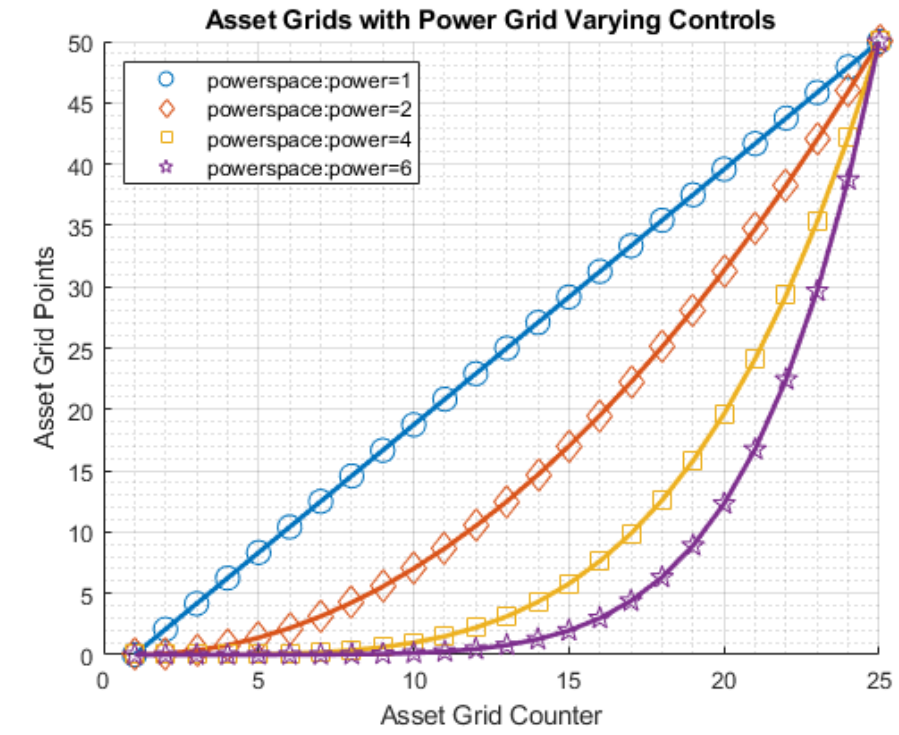


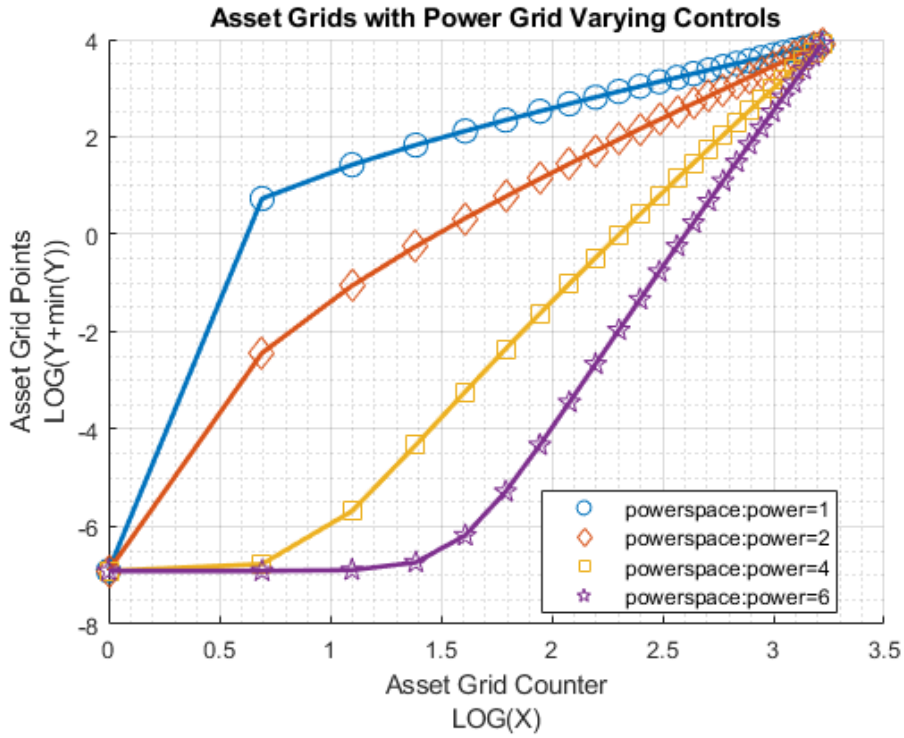


5.1.4 Test FF_SAVEBORR_GRID Power Grid Changing Parameters

Log grid, same min and max, change log X1 and X2 points

```
% Same min and max and grid points
[fl_a_min, fl_a_max, it_a_points] = deal(0,50,25);
st_grid_type = 'grid_powerspace';
% Four types of grid points
mp_grid_control = containers.Map('KeyType','char', 'ValueType','any');
mp_grid_control('grid_powerspace_power') = 1;
[ar_fl_powerspace_a] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_contr
mp_grid_control('grid_powerspace_power') = 2;
[ar_fl_powerspace_b] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_contr
mp_grid_control('grid_powerspace_power') = 4;
[ar_fl_powerspace_c] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_contr
mp_grid_control('grid_powerspace_power') = 6;
[ar_fl_powerspace_d] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_contr
% draw four types of lines jointly
mt_value = [ar_fl_powerspace_a'; ar_fl_powerspace_b'; ...
    ar_fl_powerspace_c'; ar_fl_powerspace_d'];
ar_row_grid = [...
    "powerspace:power=1", ...
    "powerspace:power=2", ...
    "powerspace:power=4", ...
    "powerspace:power=6"];
ar_col_grid = 1:it_a_points;
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('cl_st_graph_title') = {'Asset Grids with Power Grid Varying Controls'};
mp_support_graph('cl_st_ytitle') = {'Asset Grid Points'};
mp_support_graph('cl_st_xtitle') = {'Asset Grid Counter'};
mp_support_graph('bl_graph_logy') = true; % do not log
ff_graph_grid(mt_value, ar_row_grid, ar_col_grid, mp_support_graph);
```



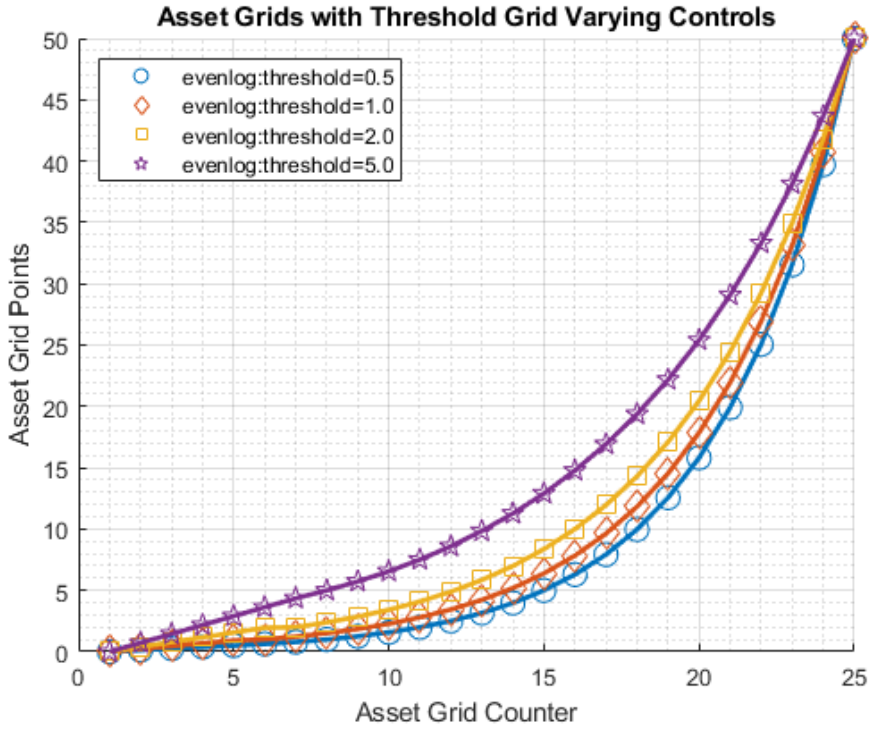


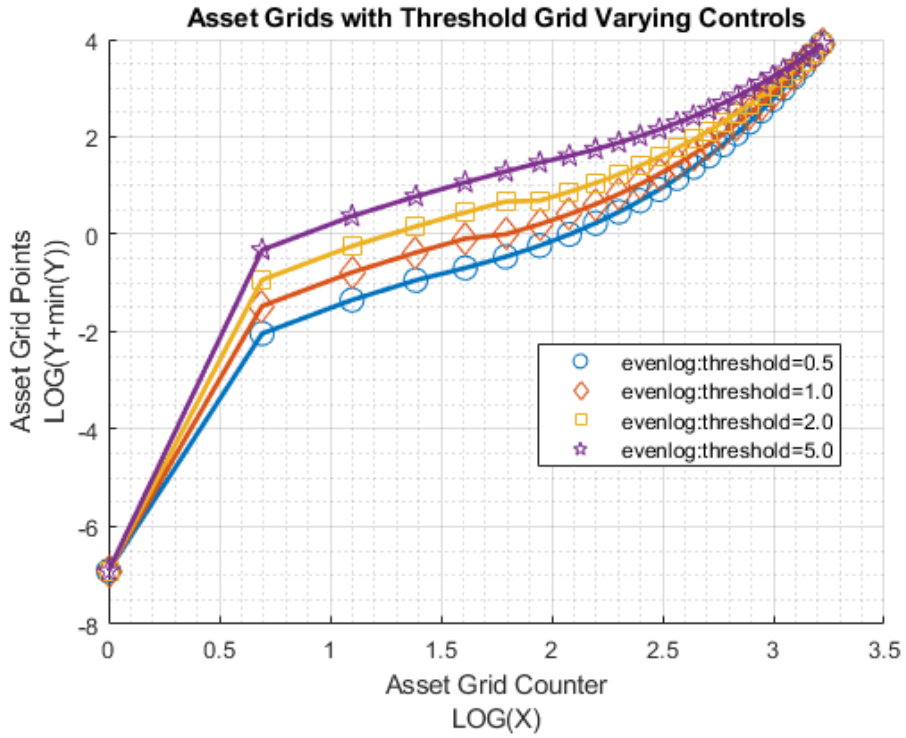
5.1.5 Test FF_SAVEBORR_GRID Threshold Grid Changing Parameters

Threshold Grid, Changing Threshold Levels. Initial segments below threshold are linspace, then logspace.

```
% Same min and max and grid points
[fl_a_min, fl_a_max, it_a_points] = deal(0,50,25);
st_grid_type = 'grid_evenlog';
% Four types of grid points
mp_grid_control = containers.Map('KeyType','char', 'ValueType','any');
mp_grid_control('grid_evenlog_threshold') = 0.50;
```

```
[ar_fl_evenlog_a] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_control)
mp_grid_control('grid_evenlog_threshold') = 1.00;
[ar_fl_evenlog_b] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_control)
mp_grid_control('grid_evenlog_threshold') = 2;
[ar_fl_evenlog_c] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_control)
mp_grid_control('grid_evenlog_threshold') = 5;
[ar_fl_evenlog_d] = ff_saveborr_grid(fl_a_min, fl_a_max, it_a_points, st_grid_type, mp_grid_control)
% draw four types of lines jointly
mt_value = [ar_fl_evenlog_a'; ar_fl_evenlog_b'; ...
    ar_fl_evenlog_c'; ar_fl_evenlog_d'];
ar_row_grid = [...
    "evenlog:threshold=0.5", ...
    "evenlog:threshold=1.0", ...
    "evenlog:threshold=2.0", ...
    "evenlog:threshold=5.0"];
ar_col_grid = 1:it_a_points;
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('cl_st_graph_title') = {'Asset Grids with Threshold Grid Varying Controls'};
mp_support_graph('cl_st_ytitle') = {'Asset Grid Points'};
mp_support_graph('cl_st_xtitle') = {'Asset Grid Counter'};
mp_support_graph('bl_graph_logy') = true; % do not log
ff_graph_grid(mt_value, ar_row_grid, ar_col_grid, mp_support_graph);
```




Asset Grids with Threshold Grid Varying Controls
Asset Grid Points
LOG(Y+min(Y))
Asset Grid Counter
LOG(X)
evenlog:threshold=0.5
evenlog:threshold=1.0
evenlog:threshold=2.0
evenlog:threshold=5.0
4
2
0
-2
-4
-6
-8
0
0.5
1
1.5
2
2.5
3
3.5

Chapter 6

Common Functions

6.1 FFY_TAUCHEN AR1 Shock Discretization Example

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ffy_tauchen** from the **MEconTools Package.** : See also the **ffy_rouwenhorst** function from the **MEconTools Package.** This function discretize a mean zero AR1 process, uses Tauchen (1986). See AR 1 Example for some details on how the AR1 process works. And See Kopecky and Suen (2010).

6.1.1 Test FFY_TAUCHEN Defaults

Call the function with defaults. Default sd bounds arer plus and minus 4. This is used in the following examples, unless otherwise specified as the 5th parameter.

```
ffy_tauchen();
```

```
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                         i    idx    ndim    numel    rowN    colN    sum    mean      std      coef
                         -    ---    ----    -----    ----    ----    ---    ----    -------    ----

    ar_disc_ar1          1     1      2        5       5       1       0       0     0.79057
    mt_disc_ar1_trans    2     6      2       25       5       5       5     0.2     0.27623     1.3

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxxx
           c1
          ----

    r1      -1
    r2    -0.5
    r3       0
    r4     0.5
    r5       1

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxxx
              c1             c2             c3             c4             c5
          ----------     ----------     --------     ----------     ----------

    r1       0.22663        0.73331     0.040048     1.0689e-05     7.3923e-12
    r2      0.012224        0.58648      0.39831      0.0029797      7.605e-08
```

```
    r3    8.8417e-05       0.10556      0.7887       0.10556    8.8417e-05
    r4     7.605e-08     0.0029797     0.39831       0.58648      0.012224
    r5    7.3923e-12    1.0689e-05    0.040048       0.73331       0.22663

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx    value
                          -    ---    -----

    fl_ar1_persistence    1     2      0.6
    fl_ar1_step           2     3      0.5
    fl_shk_std            3     4      0.2
    it_std_bound          4     5        4
```

6.1.2 Test FFY_TAUCHEN Specify Parameters

With a grid of 10 points, the sd bounds on Tauchen and Rouwenhorst are identical. With the not extremely persistent shock process here, the Tauchen and Rouwenhorst Results are very similar.

```
[fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose, it_std_bound] = ...
    deal(0.60, 0.10, 10, true, 3);
ffy_tauchen(fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose, it_std_bound);
```

```
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx    ndim    numel    rowN    colN        sum           mean
                          -    ---    ----    -----    ----    ----    -----------    -----------

    ar_disc_ar1           1     1      2        10      10       1     -7.2164e-16    -7.2164e-17
    mt_disc_ar1_trans     2     6      2       100      10      10              10            0.1

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxx
              c1
           ---------

    r1        -0.375
    r2      -0.29167
    r3      -0.20833
    r4        -0.125
    r5     -0.041667
    r6      0.041667
    r7         0.125
    r8       0.20833
    r9       0.29167
    r10        0.375

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxx
              c1            c2            c3            c4            c5            c6            c7
           ----------    ----------    ----------    ----------    --------    --------    ---------

    r1        0.13933       0.26196       0.31887       0.20154    0.066066    0.011201    0.0009785
    r2       0.056673       0.16995       0.30658       0.28713      0.1396    0.035167     0.004575
    r3        0.01861      0.087039       0.23281       0.32308     0.23281    0.087039      0.01684
    r4      0.0048925      0.035167        0.1396       0.28713     0.30658     0.16995      0.04884
    r5      0.0010235      0.011201      0.066066       0.20154     0.31887     0.26196       0.1116
```

```
    r6     0.00016962     0.0028101       0.02466       0.11169     0.26196     0.31887       0.2015
    r7     2.2197e-05    0.00055483     0.0072547      0.048841     0.16995     0.30658       0.2871
    r8     2.2881e-06    8.6129e-05     0.0016806      0.016841    0.087039     0.23281       0.3230
    r9     1.8543e-07    1.0503e-05    0.00030628     0.0045756    0.035167      0.1396       0.2871
    r10    1.1798e-08    1.0053e-06    4.3874e-05    0.00097859    0.011201    0.066066       0.2015

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx     value
                          -    ---    --------

    fl_ar1_persistence    1     2         0.6
    fl_ar1_step           2     3    0.083333
    fl_shk_std            3     4         0.1
    it_std_bound          4     5           3
```

6.1.3 Test FFY_TAUCHEN High Persistence, Low SD

```
[fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose] = ...
    deal(0.99, 0.01, 7, true);
ffy_tauchen(fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx    ndim    numel    rowN    colN        sum           mean
                          -    ---    ----    -----    ----    ----    -----------    -----------

    ar_disc_ar1           1     1      2        7       7       1      -5.5511e-17    -7.9302e-18
    mt_disc_ar1_trans     2     6      2       49       7       7                7        0.14286

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxx
              c1
          -----------

    r1       -0.28355
    r2       -0.18903
    r3      -0.094517
    r4     -2.7756e-17
    r5       0.094517
    r6        0.18903
    r7        0.28355

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxx
              c1             c2             c3             c4             c5             c6          --
          -----------    -----------    -----------    -----------    ----------    ----------    --

    r1              1     4.4497e-06              0              0             0             0
    r2     4.4412e-07              1     2.8552e-06              0             0             0
    r3      1.632e-46     7.1638e-07              1     1.8164e-06             0             0
    r4    9.6185e-124     6.3021e-46     1.1456e-06              1    1.1456e-06             0
    r5    6.3206e-239    8.9712e-123     2.4121e-45     1.8164e-06             1    7.1638e-07
    r6              0     1.426e-237    8.2932e-122     9.1503e-45    2.8552e-06             1    4.
    r7              0              0    3.1885e-236    7.5984e-121    3.4405e-44    4.4497e-06
```

```
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx     value
                          -    ---    --------

    fl_ar1_persistence    1     2         0.99
    fl_ar1_step           2     3     0.094517
    fl_shk_std            3     4         0.01
    it_std_bound          4     5            4
```

6.1.4 Test FFY_TAUCHEN Low Persistence, Low SD

```
[fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose] = ...
    deal(0.01, 0.01, 7, true);
ffy_tauchen(fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx    ndim    numel    rowN    colN    sum     mean        std
                          -    ---    ----    -----    ----    ----    ---    -------    --------

    ar_disc_ar1           1     1      2        7       7       1       0           0    0.028805
    mt_disc_ar1_trans     2     6      2       49       7       7       7     0.14286     0.17448

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxx
              c1
          ---------

    r1    -0.040002
    r2    -0.026668
    r3    -0.013334
    r4            0
    r5     0.013334
    r6     0.026668
    r7     0.040002

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxx
               c1           c2          c3          c4          c5          c6           c7
          ----------    --------    -------    -------    -------    --------    ----------

    r1    0.00049475    0.024497    0.24044     0.4947    0.21921    0.020299    0.00037109
    r2    0.00047179    0.023751    0.23685    0.49488     0.2227    0.020954    0.00038948
    r3    0.00044982    0.023024    0.23329      0.495    0.22621    0.021626     0.0004087
    r4     0.0004288    0.022316    0.22974    0.49504    0.22974    0.022316     0.0004288
    r5     0.0004087    0.021626    0.22621      0.495    0.23329    0.023024    0.00044982
    r6    0.00038948    0.020954     0.2227    0.49488    0.23685    0.023751    0.00047179
    r7    0.00037109    0.020299    0.21921     0.4947    0.24044    0.024497    0.00049475

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx     value
                          -    ---    --------
```

```
    fl_ar1_persistence    1     2         0.01
    fl_ar1_step           2     3     0.013334
    fl_shk_std            3     4         0.01
    it_std_bound          4     5            4
```

6.1.5 Test FFY_TAUCHEN High Persistence, High SD

```
[fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose] = ...
    deal(0.99, 0.99, 7, true);
ffy_tauchen(fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx    ndim    numel    rowN    colN        sum           mean
                          -    ---    ----    -----    ----    ----    -----------    -----------

    ar_disc_ar1           1     1      2        7       7       1      -3.5527e-15    -5.0753e-16
    mt_disc_ar1_trans     2     6      2       49       7       7                7        0.14286

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxxx
            c1
          -------

    r1    -28.072
    r2    -18.714
    r3    -9.3572
    r4          0
    r5     9.3572
    r6     18.714
    r7     28.072

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxxx
              c1              c2              c3              c4              c5              c6
          -----------    -----------    -----------    -----------    ----------    ----------    --

    r1              1     4.4497e-06              0              0             0             0
    r2     4.4412e-07              1     2.8552e-06              0             0             0
    r3      1.632e-46     7.1638e-07              1     1.8164e-06             0             0
    r4    9.6185e-124     6.3021e-46     1.1456e-06              1    1.1456e-06             0
    r5    6.3206e-239    8.9712e-123     2.4121e-45     1.8164e-06             1    7.1638e-07
    r6              0     1.426e-237    8.2932e-122     9.1503e-45    2.8552e-06             1    4.
    r7              0              0    3.1885e-236    7.5984e-121    3.4405e-44    4.4497e-06

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx    value
                          -    ---    ------

    fl_ar1_persistence    1     2       0.99
    fl_ar1_step           2     3     9.3572
    fl_shk_std            3     4       0.99
    it_std_bound          4     5          4
```

6.1.6 Test FFY_TAUCHEN Low Persistence, Low SD

```
[fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose] = ...
    deal(0.01, 0.01, 7, true);
ffy_tauchen(fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx    ndim    numel    rowN    colN    sum     mean        std
                          -    ---    ----    -----    ----    ----    ---    -------    --------

    ar_disc_ar1           1     1      2        7       7       1       0           0    0.028805
    mt_disc_ar1_trans     2     6      2       49       7       7       7     0.14286     0.17448

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxx
             c1
          ---------

    r1    -0.040002
    r2    -0.026668
    r3    -0.013334
    r4            0
    r5     0.013334
    r6     0.026668
    r7     0.040002

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxx
              c1           c2          c3          c4          c5          c6           c7
          ----------    --------    -------    -------    -------    --------    ----------

    r1    0.00049475    0.024497    0.24044     0.4947    0.21921    0.020299    0.00037109
    r2    0.00047179    0.023751    0.23685    0.49488     0.2227    0.020954    0.00038948
    r3    0.00044982    0.023024    0.23329      0.495    0.22621    0.021626     0.0004087
    r4     0.0004288    0.022316    0.22974    0.49504    0.22974    0.022316     0.0004288
    r5     0.0004087    0.021626    0.22621      0.495    0.23329    0.023024    0.00044982
    r6    0.00038948    0.020954     0.2227    0.49488    0.23685    0.023751    0.00047179
    r7    0.00037109    0.020299    0.21921     0.4947    0.24044    0.024497    0.00049475

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx     value
                          -    ---    --------

    fl_ar1_persistence    1     2        0.01
    fl_ar1_step           2     3    0.013334
    fl_shk_std            3     4        0.01
    it_std_bound          4     5           4
```

6.2 FFY_ROUWENHORST AR1 Shock Discretization Example

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ffy_rouwenhorst** from the **MEconTools Package**. See also **ffy_tauchen** function from the **MEconTools Package**. This function discretize a mean zero AR1 process, uses Rouwenhorst (1995). See AR 1 Example for some details on how the AR1 process works. And See Kopecky and Suen (2010).

6.2.1 Test FFY_ROUWENHORST Defaults

Call the function with defaults.

```
ffy_rouwenhorst();
```

```
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx    ndim    numel    rowN    colN    sum    mean      std       coef
                          -    ---    ----    -----    ----    ----    ---    ----    -------    ----

    ar_disc_ar1           1     1      2        5       5       1       0       0     0.39528
    mt_disc_ar1_trans     2    11      2       25       5       5       5     0.2     0.18246    0.91

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxx
           c1
          -----

    r1     -0.5
    r2    -0.25
    r3        0
    r4     0.25
    r5      0.5

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxx
            c1        c2        c3        c4        c5
          ------    ------    ------    ------    ------

    r1    0.4096    0.4096    0.1536    0.0256    0.0016
    r2    0.1024    0.4864    0.3264    0.0784    0.0064
    r3    0.0256    0.2176    0.5136    0.2176    0.0256
    r4    0.0064    0.0784    0.3264    0.4864    0.1024
    r5    0.0016    0.0256    0.1536    0.4096    0.4096

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx    value
                          -    ---    -----

    fl_ar1_beg            1     2     -0.5
    fl_ar1_end            2     3      0.5
    fl_ar1_persistence    3     4      0.6
    fl_ar1_step           4     5     0.25
    fl_p0                 5     6      0.8
    fl_q0                 6     7      0.8
    fl_shk_std            7     8      0.2
    fl_sig_ar1            8     9     0.25
    it_std_bound          9    10        0
```

6.2.2 Test FFY_ROUWENHORST Specify Parameters

With a grid of 10 points, the Rwouenhorst bounds on standard deviations are equall to Tauchen bounds of 3. With the not extremely persistent shock process here, the Tauchen and Rouwenhorst Results are very similar.

```
[fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose] = ...
    deal(0.60, 0.10, 10, true);
ffy_rouwenhorst(fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                         i    idx    ndim    numel    rowN    colN       sum           mean          --
                         -    ---    ----    -----    ----    ----    ----------    ----------    --

    ar_disc_ar1          1     1      2        10      10       1     5.5511e-17    5.5511e-18     0
    mt_disc_ar1_trans    2    11      2       100      10      10             10           0.1    0.

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxx
              c1
           ---------

    r1        -0.375
    r2       -0.29167
    r3       -0.20833
    r4         -0.125
    r5       -0.041667
    r6        0.041667
    r7           0.125
    r8         0.20833
    r9         0.29167
    r10          0.375

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxx
               c1             c2             c3             c4             c5             c6             c7
           ----------    ----------    ----------    ---------    --------    --------    ---------

    r1        0.13422       0.30199       0.30199      0.17616     0.06606    0.016515    0.0027525
    r2       0.033554       0.20133       0.32716      0.26424     0.12662    0.038535    0.0075694
    r3      0.0083886      0.081789       0.26267      0.32755     0.21401    0.082747     0.019741
    r4      0.0020972      0.028312       0.14038      0.30946     0.30369     0.15877     0.047989
    r5     0.00052429      0.009044      0.061145      0.20246     0.33477     0.25969      0.10585
    r6     0.00013107     0.0027525      0.023642      0.10585     0.25969     0.33477      0.20246
    r7     3.2768e-05    0.00081101     0.0084603     0.047989     0.15877     0.30369      0.30946
    r8      8.192e-06    0.00023347     0.0028677     0.019741    0.082747     0.21401      0.32755
    r9      2.048e-06    6.6048e-05    0.00093389    0.0075694    0.038535     0.12662      0.26424
    r10      5.12e-07    1.8432e-05    0.00029491    0.0027525    0.016515     0.06606      0.17616

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                          i    idx     value
                          -    ---    --------

    fl_ar1_beg            1     2       -0.375
    fl_ar1_end            2     3        0.375
```

```
    fl_ar1_persistence    3     4          0.6
    fl_ar1_step           4     5     0.083333
    fl_p0                 5     6          0.8
    fl_q0                 6     7          0.8
    fl_shk_std            7     8          0.1
    fl_sig_ar1            8     9        0.125
    it_std_bound          9    10            0
```

6.2.3 Test FFY_ROUWENHORST High Persistence, Low SD

```
[fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose] = ...
    deal(0.99, 0.01, 7, true);
ffy_rouwenhorst(fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                         i    idx    ndim    numel    rowN    colN    sum     mean        std      c
                         -    ---    ----    -----    ----    ----    ---    -------    -------    -

    ar_disc_ar1          1     1      2        7       7       1       0           0    0.12503
    mt_disc_ar1_trans    2    11      2       49       7       7       7     0.14286    0.34148

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxx
             c1
          --------

    r1    -0.17364
    r2    -0.11576
    r3    -0.05788
    r4           0
    r5     0.05788
    r6     0.11576
    r7     0.17364

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxx
              c1            c2            c3            c4            c5            c6            c7
          ----------    ----------    ----------    ----------    ----------    ----------    ------

    r1       0.97037      0.029257    0.00036756    2.4627e-06    9.2815e-09    1.8656e-11    1.5625
    r2     0.0048762        0.9705      0.024382    0.00024504    1.2314e-06    3.0938e-09    3.1094
    r3    2.4504e-05      0.009753       0.97057      0.019506    0.00014703    4.9254e-07    6.1877
    r4    1.2313e-07    7.3513e-05       0.01463       0.97059       0.01463    7.3513e-05    1.2313
    r5    6.1877e-10    4.9254e-07    0.00014703      0.019506       0.97057      0.009753    2.4504
    r6    3.1094e-12    3.0938e-09    1.2314e-06    0.00024504      0.024382        0.9705     0.004
    r7    1.5625e-14    1.8656e-11    9.2815e-09    2.4627e-06    0.00036756      0.029257       0.9

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                         i    idx     value
                         -    ---    --------

    fl_ar1_beg           1     2     -0.17364
    fl_ar1_end           2     3      0.17364
    fl_ar1_persistence   3     4         0.99
```

```
    fl_ar1_step           4     5      0.05788
    fl_p0                 5     6       0.995
    fl_q0                 6     7       0.995
    fl_shk_std            7     8        0.01
    fl_sig_ar1            8     9    0.070888
    it_std_bound          9    10           0
```

6.2.4 Test FFY_ROUWENHORST Low Persistence, Low SD

```
[fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose] = ...
    deal(0.01, 0.01, 7, true);
ffy_rouwenhorst(fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                         i    idx    ndim    numel    rowN    colN    sum     mean        std
                         -    ---    ----    -----    ----    ----    ---    -------    --------

    ar_disc_ar1          1     1      2        7       7       1       0           0    0.017639
    mt_disc_ar1_trans    2    11      2       49       7       7       7     0.14286     0.10985

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxx
              c1
          ----------

    r1     -0.024496
    r2     -0.016331
    r3    -0.0081654
    r4             0
    r5     0.0081654
    r6      0.016331
    r7      0.024496

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxx
             c1          c2          c3          c4          c5          c6          c7
          --------    --------    -------    -------    -------    --------    --------

    r1    0.016586    0.097547    0.23904    0.31241    0.22966    0.090047    0.014711
    r2    0.016258    0.096266    0.23749    0.31247    0.23124    0.091266    0.015008
    r3    0.015936    0.094997    0.23594    0.31251    0.23281    0.092497    0.015311
    r4     0.01562    0.093741    0.23438    0.31252    0.23438    0.093741     0.01562
    r5    0.015311    0.092497    0.23281    0.31251    0.23594    0.094997    0.015936
    r6    0.015008    0.091266    0.23124    0.31247    0.23749    0.096266    0.016258
    r7    0.014711    0.090047    0.22966    0.31241    0.23904    0.097547    0.016586

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                         i    idx      value
                         -    ---    ---------

    fl_ar1_beg           1     2     -0.024496
    fl_ar1_end           2     3      0.024496
    fl_ar1_persistence   3     4          0.01
    fl_ar1_step          4     5     0.0081654
```

```
    fl_p0                5     6         0.505
    fl_q0                6     7         0.505
    fl_shk_std           7     8          0.01
    fl_sig_ar1           8     9      0.010001
    it_std_bound         9    10             0
```

6.2.5 Test FFY_ROUWENHORST High Persistence, High SD

```
[fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose] = ...
    deal(0.99, 0.99, 7, true);
ffy_rouwenhorst(fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                         i    idx    ndim    numel    rowN    colN       sum           mean          --

    ar_disc_ar1          1     1      2        7       7       1      3.5527e-15    5.0753e-16     1
    mt_disc_ar1_trans    2    11      2       49       7       7               7       0.14286    0.

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxxx
             c1
          -------

    r1     -17.19
    r2     -11.46
    r3    -5.7301
    r4          0
    r5     5.7301
    r6      11.46
    r7      17.19

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxxx
              c1            c2            c3            c4            c5            c6            c7
          ----------    ----------    ----------    ----------    ----------    ----------    ------

    r1       0.97037      0.029257    0.00036756    2.4627e-06    9.2815e-09    1.8656e-11    1.5625
    r2     0.0048762        0.9705      0.024382    0.00024504    1.2314e-06    3.0938e-09    3.1094
    r3    2.4504e-05      0.009753       0.97057      0.019506    0.00014703    4.9254e-07    6.1877
    r4    1.2313e-07    7.3513e-05       0.01463       0.97059       0.01463    7.3513e-05    1.2313
    r5    6.1877e-10    4.9254e-07    0.00014703      0.019506       0.97057      0.009753    2.4504
    r6    3.1094e-12    3.0938e-09    1.2314e-06    0.00024504      0.024382        0.9705     0.004
    r7    1.5625e-14    1.8656e-11    9.2815e-09    2.4627e-06    0.00036756      0.029257       0.9

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                         i    idx    value
                         -    ---    ------

    fl_ar1_beg           1     2     -17.19
    fl_ar1_end           2     3      17.19
    fl_ar1_persistence   3     4       0.99
    fl_ar1_step          4     5     5.7301
    fl_p0                5     6      0.995
```

```
    fl_q0                6     7      0.995
    fl_shk_std           7     8       0.99
    fl_sig_ar1           8     9     7.0179
    it_std_bound         9    10          0
```

6.2.6 Test FFY_ROUWENHORST Low Persistence, Low SD

```
[fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose] = ...
    deal(0.01, 0.01, 7, true);
ffy_rouwenhorst(fl_ar1_persistence, fl_shk_std, it_disc_points, bl_verbose);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                         i    idx    ndim    numel    rowN    colN    sum     mean        std
                         -    ---    ----    -----    ----    ----    ---    -------    --------

    ar_disc_ar1          1     1      2        7       7       1       0          0    0.017639
    mt_disc_ar1_trans    2    11      2       49       7       7       7    0.14286     0.10985

xxx TABLE:ar_disc_ar1 xxxxxxxxxxxxxxxxx
              c1
          ----------

    r1     -0.024496
    r2     -0.016331
    r3    -0.0081654
    r4             0
    r5     0.0081654
    r6      0.016331
    r7      0.024496

xxx TABLE:mt_disc_ar1_trans xxxxxxxxxxxxxxxxx
              c1          c2          c3          c4          c5          c6          c7
          --------    --------    -------    -------    -------    --------    --------

    r1    0.016586    0.097547    0.23904    0.31241    0.22966    0.090047    0.014711
    r2    0.016258    0.096266    0.23749    0.31247    0.23124    0.091266    0.015008
    r3    0.015936    0.094997    0.23594    0.31251    0.23281    0.092497    0.015311
    r4     0.01562    0.093741    0.23438    0.31252    0.23438    0.093741     0.01562
    r5    0.015311    0.092497    0.23281    0.31251    0.23594    0.094997    0.015936
    r6    0.015008    0.091266    0.23124    0.31247    0.23749    0.096266    0.016258
    r7    0.014711    0.090047    0.22966    0.31241    0.23904    0.097547    0.016586

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                         i    idx      value
                         -    ---    ---------

    fl_ar1_beg           1     2     -0.024496
    fl_ar1_end           2     3      0.024496
    fl_ar1_persistence   3     4          0.01
    fl_ar1_step          4     5     0.0081654
    fl_p0                5     6         0.505
    fl_q0                6     7         0.505
```

```
fl_shk_std            7     8          0.01
fl_sig_ar1            8     9      0.010001
it_std_bound          9    10             0
```

Chapter 7

Support Tools

7.1 FF_CONTAINER_MAP_DISPLAY Examples

Go back to fan's MEconTools Toolbox (bookdown), Matlab Code Examples Repository (bookdown), or Math for Econ with Matlab Repository (bookdown).

This is the example vignette for function: **ff_container_map_display** from the **MEconTools Package.** This function summarizes statistics of matrixes stored in a container map, as well as scalar, string, function and other values stored in container maps.

7.1.1 Test FF_CONTAINER_MAP_DISPLAY Defaults

Call the function with defaults.

```
ff_container_map_display();
```

```
----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	ndim	numel	rowN	colN	sum	mean	std
mat_1	1	7	2	12	3	4	6.5142	0.54285	0.2232
mat_2	2	8	2	2650	50	53	1313.3	0.49559	0.29232
mat_2_boolean	3	9	2	2650	50	53	1361	0.51358	0.49991
mat_3	4	10	2	4	2	2	1.8111	0.45277	0.45111
tensor_1	5	15	3	16	2	8	7.3043	0.45652	0.27787
tensor_2	6	16	3	75	3	25	40.195	0.53593	0.29044
tensor_3	7	17	2	4	1	4	1.6926	0.42315	0.37389
tesseract_1	8	18	4	72	3	24	34.321	0.47669	0.26374
tesseract_2	9	19	4	20	2	10	8.4191	0.42096	0.28981
tesseract_bl_3	10	20	4	10	1	10	3	0.3	0.48305

```
xxx TABLE:mat_1 xxxxxxxxxxxxxxxxxx
```

	c1	c2	c3	c4
r1	0.69647	0.55131	0.98076	0.39212
r2	0.28614	0.71947	0.68483	0.34318
r3	0.22685	0.42311	0.48093	0.72905

```
xxx TABLE:mat_2 xxxxxxxxxxxxxxxxxx
```

	c1	c2	c3	c4	c50	c51	c52	c53

```
          --------    --------    -------    --------    --------    --------    --------    ------

    r1     0.43857      0.6249    0.17108     0.56564    0.072152     0.67855     0.61667     0.540
    r2    0.059678     0.67469    0.82911    0.084904     0.63289     0.27236     0.32528     0.249
    r3     0.39804     0.84234    0.33867     0.58267    0.046367     0.44513    0.075047      0.78
    r4       0.738    0.083195    0.55237     0.81484     0.50561     0.11117     0.59532     0.356
    r5     0.18249     0.76368    0.57855     0.33707     0.10653    0.028681      0.7435     0.918
    r46     0.6813     0.55326    0.88786     0.69983     0.83758     0.16382     0.74191    0.0656
    r47    0.87546     0.85445    0.69631     0.66117     0.97069     0.79092     0.42466     0.787
    r48    0.51042     0.38484    0.44033    0.049097    0.017768     0.33302     0.24401     0.979
    r49    0.66931     0.31679    0.43821      0.7923     0.12979     0.75311     0.79466    0.0790
    r50    0.58594     0.35426     0.7651     0.51872     0.86415     0.58281     0.84795      0.45

xxx TABLE:mat_2_boolean xxxxxxxxxxxxxxxxx
          c1       c2       c3       c4       c50      c51      c52      c53
          -----    -----    -----    -----    -----    -----    -----    -----

    r1    true     false    false    true     true     false    true     true
    r2    true     false    true     true     false    false    true     true
    r3    false    true     false    true     false    true     false    true
    r4    false    true     false    false    false    true     true     true
    r5    true     true     true     false    true     false    false    true
    r46   false    true     true     false    true     true     true     true
    r47   true     true     true     true     true     true     false    false
    r48   true     false    false    false    true     true     false    true
    r49   true     true     false    true     true     true     false    false
    r50   false    false    false    false    false    false    false    false

xxx TABLE:mat_3 xxxxxxxxxxxxxxxxx
             c1          c2
          ----------    -------

    r1    0.00012471    0.13253
    r2       0.88615    0.79226

xxx TABLE:tensor_1 xxxxxxxxxxxxxxxxx
            c1         c2         c3         c4         c5         c6         c7         c8
          --------    -------    -------    -------    -------    -------    ------    -------

    r1    0.019363    0.34271    0.52167    0.53703    0.75756    0.68839    0.8345    0.26597
    r2    0.018091    0.33355    0.11738    0.77857    0.81933    0.28644    0.6157      0.368

xxx TABLE:tensor_2 xxxxxxxxxxxxxxxxx
            c1         c2         c3         c4         c22        c23         c24         c25
          --------    -------    -------    -------    -------    -------    ---------    -------

    r1     0.51866    0.40495    0.48278    0.99731    0.46584    0.62976     0.035924    0.10505
    r2    0.028692    0.37408    0.24149    0.35201    0.66054    0.87243    0.0024293    0.81088
    r3     0.87339    0.19457    0.83212    0.15315    0.77859    0.96663       0.2501     0.8056

xxx TABLE:tensor_3 xxxxxxxxxxxxxxxxx
            c1         c2         c3          c4
          ------    ------    -------    -------

    r1    0.1219    0.5119    0.91553    0.14329

xxx TABLE:tesseract_1 xxxxxxxxxxxxxxxxx
            c1         c2         c3          c4         c21        c22        c23        c24
```

```
          -------    -------    -------    --------    -------    -------    -------    -------

    r1    0.64531    0.59299    0.32115     0.67653    0.90328    0.56911    0.52562    0.12014
    r2    0.74558     0.5007    0.46142     0.21384    0.35564    0.13732      0.155    0.23786
    r3    0.91137    0.46403    0.18118    0.049919    0.46246    0.46842    0.75348    0.64547

xxx TABLE:tesseract_2 xxxxxxxxxxxxxxxxx
             c1          c2          c3          c4          c7          c8          c9          c10
          --------    -------    -------    -------    -------    -------    --------    -------

    r1     0.28898    0.48211    0.44359    0.97146    0.61782    0.65121     0.80715    0.11605
    r2    0.094493    0.34941    0.17595    0.14192    0.16754    0.57097    0.043114    0.70518

xxx TABLE:tesseract_bl_3 xxxxxxxxxxxxxxxxx
           c1       c2       c3       c4       c7       c8       c9       c10
          -----    -----    -----    -----    -----    -----    -----    -----

    r1    false    false    true     true     false    true     false    false

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                      i    idx     value
                      -    ---    -------

    boolean_1         1     1           1
    empty             2     2         NaN
    mat_4             3    11     0.74898
    string_float_1    4    13      1021.1
    string_int_1      5    14        1021

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 String
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                      i     idx            string
                     ---    ----    ---------------------

    list_string_1    "1"    "5"     "col1;col2;col3;col4"
    list_string_2    "2"    "6"     "row1;row2;row3;row4"
    string_1         "3"    "12"    "Table Name"

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 Functions
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
              i     idx      functionString
             ---    ---    -------------------

    func1    "1"    "3"    "@(x)1+2+x"
    func2    "2"    "4"    "@(x,y)x*1+sqrt(y)"
```

7.1.2 Test FF_CONTAINER_MAP_DISPLAY summarize Matrix Only

Three large matrixes, show summaries

```
% Create Container
mp_container_map = containers.Map('KeyType','char', 'ValueType','any');
```

```
rng(123);
mp_container_map('mat_1') = rand(100,100);
mp_container_map('mat_2') = rand(100,100)*2 + 1;
mp_container_map('mat_2_boolean') = (rand(100,100) > 0.5);
% Will only print
ff_container_map_display(mp_container_map);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                     i    idx    ndim    numel    rowN    colN     sum       mean        std      co
                     -    ---    ----    -----    ----    ----    ------    -------    -------    --

    mat_1            1     1      2      10000    100     100     4982.3    0.49823    0.28829    0.
    mat_2            2     2      2      10000    100     100      20029     2.0029    0.57632    0.
    mat_2_boolean    3     3      2      10000    100     100       4995     0.4995    0.50002     1
```

7.1.3 Test FF_CONTAINER_MAP_DISPLAY Show Matrix Subset

A container map with three small matrixes, print only only 2 rows and 3 columns.

```
% Create Container
mp_container_map = containers.Map('KeyType','char', 'ValueType','any');
rng(789);
mp_container_map('mat_1') = rand(3,4);
mp_container_map('mat_2') = rand(50,53);
mp_container_map('mat_2_boolean') = (rand(50,53) > 0.5);
% Will only print
ff_container_map_display(mp_container_map, 2, 3);

----------------------------------------
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                     i    idx    ndim    numel    rowN    colN     sum       mean        std      co
                     -    ---    ----    -----    ----    ----    ------    -------    -------    --

    mat_1            1     1      2        12       3       4     4.9876    0.41564    0.33586    0.
    mat_2            2     2      2      2650      50      53     1324.3    0.49973    0.28834    0.
    mat_2_boolean    3     3      2      2650      50      53       1350    0.50943    0.50001    0.

xxx TABLE:mat_1 xxxxxxxxxxxxxxxxxx
            c1         c2         c3         c4
          -------    -------    -------    -------

    r1    0.32333    0.62442    0.01062    0.53815
    r3    0.79378    0.75889    0.11104    0.55157

xxx TABLE:mat_2 xxxxxxxxxxxxxxxxxx
             c1         c2         c52        c53
           -------    -------    -------    -------

    r1     0.72837    0.20976    0.74583    0.22321
    r50    0.52812      0.545    0.49521    0.29826

xxx TABLE:mat_2_boolean xxxxxxxxxxxxxxxxx
             c1        c2        c52       c53
           -----     -----     -----     -----
```

```
r1     false    true     true     true
r50    true     false    false    true
```

Appendix A

Index and Code Links

A.1 Savings Dynamic Programming links

1. Looped Grid Infinite Horizon Dynamic Savings Problem: **mlx** | **m** | **pdf** | **html**
   - Common grid looped solution
   - Slow, but easy to modify, useful for developing new models
   - Given preferences, some AR(1) shock process, solve the infinite horizon household savings dynamic programming problem. The state-space and choice-space share the same asset grid.
   - **MEconTools**: *ff_vfi_az_loop()*
2. Vectorized Grid Infinite Horizon Dynamic Savings Problem: **mlx** | **m** | **pdf** | **html**
   - Common grid vectorized solution
   - Fast, sufficiently approximate value(a,z), but c(a,z) not precise
   - Given preferences, some AR(1) shock process, solve the infinite horizon household savings dynamic programming problem. The state-space and choice-space share the same asset grid.
   - **MEconTools**: *ff_vfi_az_vec()*
3. Looped Grid Infinite Horizon Dynamic Savings Problem: **mlx** | **m** | **pdf** | **html**
   - Exact choice looped solution
   - Slow, but high precision at low grid size (given value grid, accurate up to 0.001525878 percent of individual-specific cash-on-hand)
   - Given preferences, some AR(1) shock process, solve the infinite horizon household savings dynamic programming problem. The state-space is on a grid. The choice space are continuous percentages of cash-on-hand.
   - **MEconTools**: *ff_vfi_az_bisec_loop()*
4. Vectorized Grid Infinite Horizon Dynamic Savings Problem: **mlx** | **m** | **pdf** | **html**
   - Exact choice vectorized solution
   - Fast, approximates choices with higher precision (given value grid, accurate up to 0.001525878 percent of individual-specific cash-on-hand)
   - Given preferences, some AR(1) shock process, solve the infinite horizon household savings dynamic programming problem. The state-space is on a grid. The choice space are continuous percentages of cash-on-hand.
   - **MEconTools**: *ff_vfi_az_bisec_vec()*

A.2 Summarize Policy and Value links

1. Summarize ND Array Policy and Value Functions: **mlx** | **m** | **pdf** | **html**
   - Given an NDarray matrix with N1, N2, ..., ND dimensions. Generate average and standard deviation for the 3rd dimension, grouping by the other dimensions.
   - For example, show the 5th dimension as the column groups, and the other variables generate combinations shown as rows.
   - The resulting summary statistics table contains mean and standard deviation among other statistics over the policy or value contained in the ND array.
   - **MEconTools**: *ff_summ_nd_array()*

A.3 Distributional Analysis links

1. Gateway Joint Probability Mass Statistics: **mlx** | **m** | **pdf** | **html**
   - Given probability mass function f(s), and information y(s), x(s), z(s) at each element of the state-space, compute statistics for each variable, y, x, z, which are all discrete random variables.
   - Compute their correlation and covariance.
   - **MEconTools**: *ff_simu_stats()*
2. Discrete Random Variable Distributional Statistics: **mlx** | **m** | **pdf** | **html**
   - Model simulation generates discrete random variables, calculate mean, standard deviation, min, max, percentiles, and proportion of outcomes held by x percentiles, etc.
   - **MEconTools**: *ff_disc_rand_var_stats()*
3. Generate Discrete Random Variable: **mlx** | **m** | **pdf** | **html**
   - Given mass at state space points, and y, c, a, z and other outcomes or other information at each corresponding state space points, generate discrete random variable, with unique sorted values, and mass for each unique sorted values.
   - Generate additional joint distributions: if initial distribution is over f(a,z), generate joint distribution of f(y,a) or f(y,z).
   - **MEconTools**: *ff_disc_rand_var_mass2outcomes()*
4. Discrete Random Variable Correlation and Covariance: **mlx** | **m** | **pdf** | **html**
   - Given probability mass function f(s), X(s), and Y(s), compute the covariance and correlation betwen X and Y.
   - **MEconTools**: *ff_disc_rand_var_mass2covcor()*

A.4 Graphs links

1. Multiple Line Graph Function: **mlx** | **m** | **pdf** | **html**
   - Grid based Graph, x-axis one param, color another param, over outcomes.
   - **MEconTools**: *ff_graph_grid()*

A.5 Data Structures links

1. Log and Power Spaced Asset and Choice Grids: **mlx** | **m** | **pdf** | **html**
   - Generate linear, log-space, power-space, or threshold-cut asset or choice grids.
   - **MEconTools**: *ff_saveborr_grid()*

A.6 Common Functions links

1. Discretize AR1 Normal Shock Tauchen (1986): **mlx** | **m** | **pdf** | **html**
   - Mean zero AR(1) shock discretize following Tauchen (1986).
   - **MEconTools**: *ffy_tauchen()*
2. Discretize AR1 Normal Shock Rouwenhorst (1995): **mlx** | **m** | **pdf** | **html**
   - Mean zero AR(1) shock discretize following Rouwenhorst (1995).
   - **MEconTools**: *ffy_rouwenhorst()*

A.7 Support Tools links

1. Organizes and Prints Container Map Key and Values: **mlx** | **m** | **pdf** | **html**
   - Summarizes the contents of a map container by data types. Includes, scalar, array, matrix, string, functions, tensors (3-tuples), tesseracts (4-tuples).
   - **MEconTools**: *ff_container_map_display()*

Bibliography

The MathWorks Inc (2019). *MATLAB*. Matlab package version 2019b.

Xie, Y. (2020). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.18.