

FF_VFI_AZ_VEC Savings Vectorized Grid Examples

back to [Fan's Intro Math for Econ](#), [Matlab Examples](#), or [Dynamic Asset Repositories](#)

This is the example vignette for function: `ff_vfi_az_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 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.

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 and relies on First Order Conditions. The `ff_vfi_az_mzoom_vec` also solves the same class of problems with continuous exact percentage asset choices, and does not rely on First Order Conditions, but is slower than `ff_vfi_az_bisec_vec`.

Links to Other Code:

Core Savings/Borrowing Dynamic Programming Solution Functions that are functions in the [MEconTools Package](#). :

- Common Choice and States Grid Loop: [ff_vfi_az_loop](#)
- Common Choice and States Grid Vectorized: [ff_vfi_az_vec](#)
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand, rely on FOC, Loop: [ff_vfi_az_bisec_loop](#)
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand, rely on FOC Vectorized: [ff_vfi_az_bisec_vec](#)
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand, VALUE comparison, Loop: [ff_vfi_az_mzoom_loop](#)
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand, VALUE comparison, Vectorized: [ff_vfi_az_mzoom_vec](#)

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 0.136223 seconds.

```
-----
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

      i      idx      ndim      numel      rowN      colN      sum      mean      std      coefvari      min      max
      -      -      -      -      -      -      -      -      -      -      -      -
```

ap	1	1	2	700	100	7	9855.1	14.079	14.408	1.0234	0	50
xxx TABLE:ap	xxxxxxxxxxxxxxxxxxxx											
	c1	c2	c3	c4	c5	c6	c7					
	<hr/>	<hr/>	<hr/>	<hr/>	<hr/>	<hr/>	<hr/>					
r1	0	0	0	0.045213	0.25576	0.61095	1.0362					
r2	0	0	0	0.045213	0.25576	0.61095	1.0362					
r3	0	0	0	0.045213	0.25576	0.61095	1.0362					
r4	0	0	0	0.06647	0.25576	0.61095	1.0362					
r5	0	0	0	0.06647	0.25576	0.61095	1.164					
r96	43.924	43.924	43.924	43.924	43.924	45.102	45.102					
r97	45.102	45.102	45.102	45.102	45.102	46.298	46.298					
r98	46.298	46.298	46.298	46.298	46.298	47.513	47.513					
r99	47.513	47.513	47.513	47.513	47.513	48.747	48.747					
r100	48.747	48.747	48.747	48.747	48.747	50	50					

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 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_vec(mp_params, mp_support);
```

Elapsed time is 0.025309 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 4.855482 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 12.201130 seconds.

Test FF_VFI_AZ_VEC Control Outputs

Run the function first without any outputs, but only the timer.

```
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') = true;
mp_support('bl_print_params') = false;
mp_support('bl_print_iterinfo') = false;
mp_support('ls_ffcmd') = {};
ff_vfi_az_vec(mp_params, mp_support);

```

Elapsed time is 0.022504 seconds.

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 0.041571 seconds.

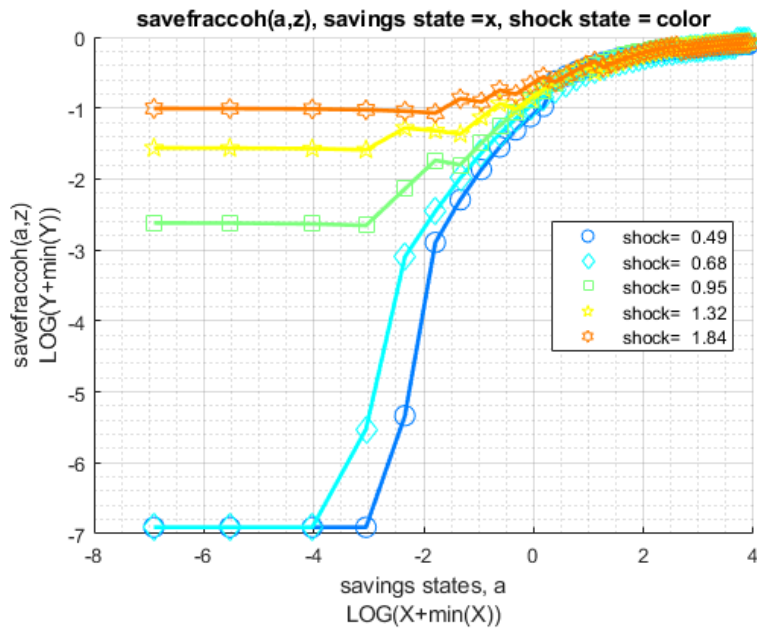
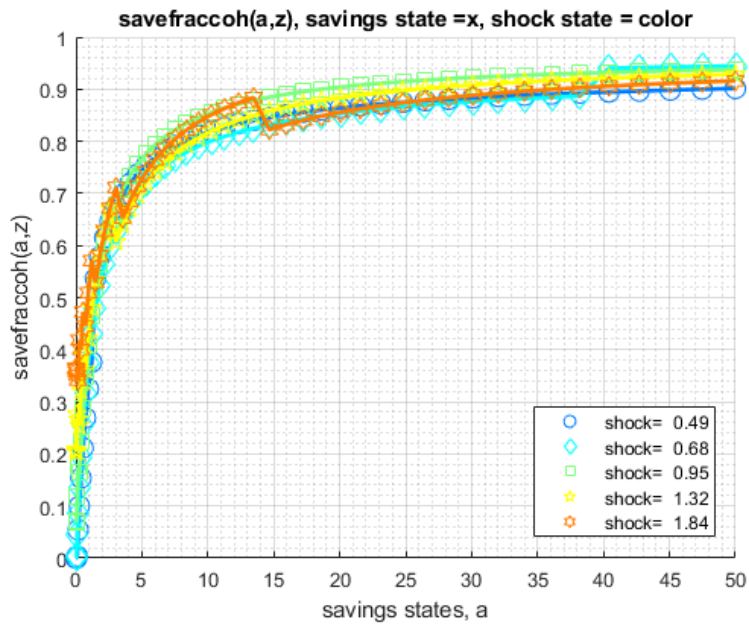
```

xxx ff_vfi_az_vec, outcome=savefraccoh xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

```

group	a	mean_z_0_4858	mean_z_0_67798	mean_z_0_9462	mean_z_1_3205	mean_z_1_8429
1	0	0	0	0.071865	0.20862	0.36462
2	0.002975	0	0	0.071698	0.20827	0.36418
3	0.016829	0	0	0.070928	0.20666	0.36216
4	0.046375	0	0.0029827	0.069341	0.20331	0.35793
5	0.095198	0.0038183	0.044243	0.11681	0.27649	0.35114
6	0.1663	0.054362	0.084837	0.17517	0.26637	0.34171
7	0.26234	0.099899	0.13609	0.16422	0.25383	0.41847
8	0.38568	0.15381	0.19428	0.22348	0.32132	0.40047
9	0.53852	0.21153	0.25554	0.28573	0.39055	0.47258
10	0.72291	0.26934	0.31659	0.34814	0.36175	0.44538
11	0.94076	0.3247	0.37504	0.40848	0.42229	0.50941
12	1.1939	0.37617	0.42941	0.46521	0.4802	0.57087
13	1.484	0.53695	0.47898	0.51743	0.5344	0.5291
14	1.8128	0.57847	0.52356	0.56473	0.58429	0.58056
15	2.1817	0.61468	0.56329	0.6071	0.62958	0.62823
16	2.5924	0.6462	0.5985	0.64475	0.67028	0.67186
17	3.0463	0.67365	0.62963	0.67804	0.60721	0.71141
18	3.5449	0.69762	0.65713	0.70737	0.6404	0.65255
19	4.0894	0.71859	0.68142	0.73318	0.67021	0.68509
20	4.6813	0.73701	0.70293	0.75587	0.6969	0.71446
21	5.3218	0.75325	0.722	0.77584	0.72078	0.74089
22	6.0121	0.76763	0.73895	0.79344	0.74211	0.76461
23	6.7536	0.7804	0.75407	0.80897	0.76119	0.78587
24	7.5474	0.7918	0.76759	0.8227	0.77824	0.80491
25	8.3948	0.80201	0.77972	0.83486	0.79351	0.82194
26	9.2967	0.81119	0.79063	0.84567	0.80719	0.83719
27	10.254	0.81947	0.80049	0.8553	0.81948	0.85083
28	11.269	0.82697	0.80941	0.86389	0.83053	0.86306
29	12.342	0.83379	0.81752	0.87159	0.84048	0.87401
30	13.473	0.84001	0.8249	0.87849	0.84946	0.88384
31	14.665	0.84569	0.83165	0.8847	0.85759	0.82241
32	15.918	0.8509	0.83782	0.8903	0.86495	0.83188

33	17.233	0.8557	0.8435	0.89536	0.87163	0.84053
34	18.611	0.86012	0.84872	0.89995	0.8777	0.84844
35	20.053	0.86421	0.85354	0.90411	0.88324	0.85568
36	21.56	0.86799	0.858	0.9079	0.8883	0.86231
37	23.133	0.87151	0.86214	0.91136	0.89292	0.86841
38	24.773	0.87479	0.86598	0.91452	0.89716	0.87401
39	26.481	0.87784	0.86955	0.91741	0.90105	0.87917
40	28.258	0.8807	0.87289	0.92007	0.90463	0.88393
41	30.104	0.88337	0.87601	0.92251	0.90793	0.88833
42	32.021	0.88588	0.87893	0.92475	0.91097	0.8924
43	34.01	0.88824	0.88166	0.92683	0.91378	0.89617
44	36.07	0.89046	0.88423	0.92874	0.91638	0.89966
45	38.204	0.89256	0.88665	0.93052	0.91879	0.90291
46	40.412	0.89453	0.9403	0.93216	0.92102	0.90592
47	42.695	0.8964	0.94141	0.93368	0.9231	0.90873
48	45.053	0.89817	0.94245	0.9351	0.92504	0.91135
49	47.488	0.89985	0.94341	0.93642	0.92684	0.9138
50	50	0.90144	0.9443	0.93765	0.92853	0.91608



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 0.230510 seconds.

XX

CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)

XX

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coefvari	min
	—	—	—	—	—	—	—	—	—	—	—
ap	1	1	2	900	100	9	12904	14.338	14.524	1.013	0
savefraccoh	2	2	2	900	100	9	619.51	0.68834	0.26953	0.39157	0

xxx TABLE:ap XXXXXXXXXXXXXXXXXXXXXXX

	c1	c2	c3	c4	c5	c6	c7	c8	c9
	—	—	—	—	—	—	—	—	—
r1	0	0	0	0	0.092813	0.25576	0.61095	1.0362	1.6023
r2	0	0	0	0	0.092813	0.25576	0.61095	1.0362	1.6023
r3	0	0	0	0	0.092813	0.25576	0.61095	1.0362	1.6023
r4	0	0	0	0.00051272	0.092813	0.25576	0.61095	1.0362	1.6023
r5	0	0	0	0.0029004	0.092813	0.25576	0.61095	1.0362	1.6023
r96	43.924	43.924	43.924	43.924	43.924	45.102	45.102	45.102	46.298
r97	45.102	45.102	45.102	45.102	45.102	46.298	46.298	46.298	47.513
r98	46.298	46.298	46.298	46.298	46.298	47.513	47.513	47.513	48.747
r99	47.513	47.513	47.513	47.513	47.513	48.747	48.747	48.747	50
r100	48.747	48.747	48.747	48.747	48.747	50	50	50	50

xxx TABLE:savefraccoh XXXXXXXXXXXXXXXXXXXXXXX

	c1	c2	c3	c4	c5	c6	c7	c8	c9
	—	—	—	—	—	—	—	—	—
r1	0	0	0	0	0.070073	0.15255	0.28789	0.38573	0.47121
r2	0	0	0	0	0.070045	0.1525	0.28781	0.38565	0.47114
r3	0	0	0	0	0.069914	0.15228	0.28748	0.3853	0.4708
r4	0	0	0	0.00048613	0.069636	0.1518	0.28676	0.38454	0.47007
r5	0	0	0	0.0027273	0.069182	0.15101	0.28559	0.38329	0.46886
r96	0.92625	0.92358	0.92022	0.916	0.91072	0.92836	0.91992	0.90945	0.92033
r97	0.92676	0.92416	0.92088	0.91677	0.91162	0.92918	0.92095	0.91073	0.92169
r98	0.92727	0.92473	0.92153	0.91752	0.91249	0.92998	0.92194	0.91196	0.923
r99	0.92776	0.92528	0.92216	0.91824	0.91333	0.93076	0.92291	0.91315	0.92426
r100	0.92823	0.92581	0.92277	0.91895	0.91416	0.93151	0.92384	0.91431	0.90252

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') = 100;
mp_params('it_z_n') = 7;
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 0.058079 seconds.

XX

CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)

XX

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coefvari	min	max
savefraccoh	1	1	2	700	100	7	357.49	0.5107	0.2755	0.53945	0	0.8

xxx TABLE:savefraccoh XXXXXXXXXXXXXXXXXXXXXXX

	c1	c2	c3	c4	c5	c6	c7
r1	0	0	0	0	0	0.0002246	0.041573
r2	0	0	0	0	0	0.00022455	0.041566
r3	0	0	0	0	0	0.0012689	0.041533
r4	0	0	0	0	0	0.001266	0.041462
r5	0	0	0	0	0	0.0034759	0.041345
r96	0.78455	0.78145	0.79995	0.79456	0.7876	0.77865	0.76719
r97	0.78669	0.78366	0.77972	0.79679	0.78998	0.78122	0.77001
r98	0.78878	0.78582	0.78197	0.79897	0.79231	0.78374	0.77276
r99	0.79084	0.78794	0.78417	0.77927	0.79459	0.7862	0.77545
r100	0.79285	0.79001	0.78633	0.78154	0.79682	0.7886	0.77808

```

% 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 0.177867 seconds.

XX

CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)

XX

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coefvari	min	max
savefraccoh	1	1	2	700	100	7	479.94	0.68563	0.27152	0.39602	0	0

xxx TABLE:savefraccoh XXXXXXXXXXXXXXXXXXXXXXX

	c1	c2	c3	c4	c5	c6	c7
r1	0	0	0	0.07007	0.17967	0.30874	0.43404
r2	0	0	0	0.070042	0.17961	0.30866	0.43396
r3	0	0	0	0.069911	0.17935	0.30833	0.4336

r4	0	0	0	0.069633	0.17881	0.30762	0.43284
r5	0	0	0.00049972	0.069179	0.17792	0.30645	0.43158
r96	0.92489	0.92134	0.91672	0.91072	0.92717	0.91691	0.92776
r97	0.92544	0.92198	0.91747	0.91162	0.92802	0.91801	0.92895
r98	0.92598	0.9226	0.9182	0.91249	0.92885	0.91908	0.9301
r99	0.9265	0.9232	0.91891	0.91333	0.92965	0.92011	0.93121
r100	0.927	0.92379	0.9196	0.91416	0.93042	0.9211	0.90914

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') = 100;
mp_params('it_z_n') = 7;
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);
```

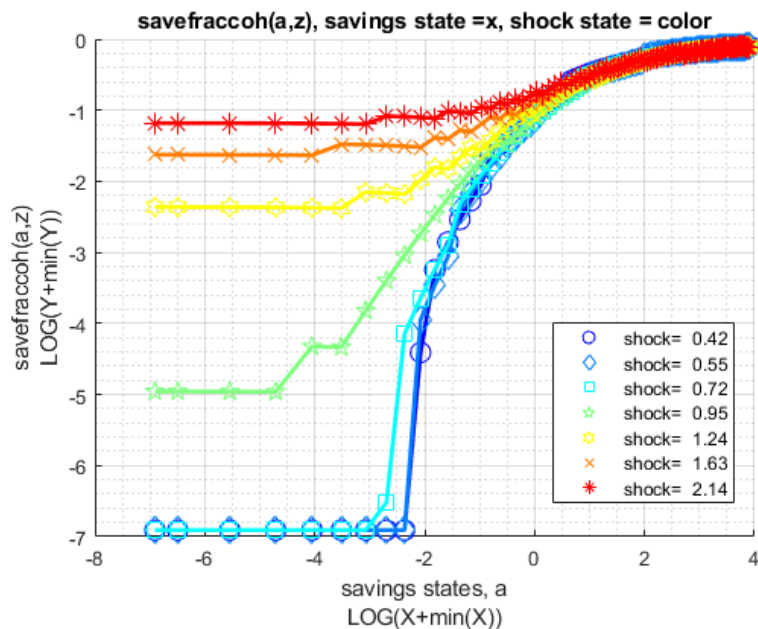
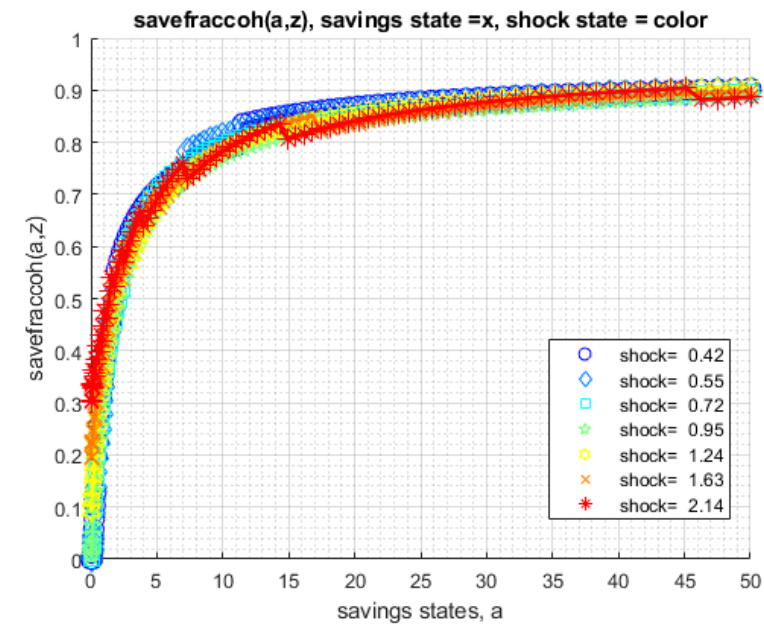
Elapsed time is 0.181638 seconds.

```
-----
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coefvari	min	max
savefraccoh	1	1	2	700	100	7	450.35	0.64336	0.2803	0.43568	0	0.93121

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c6	c7
r1	0	0	0	0.0060341	0.093241	0.19572	0.30604
r2	0	0	0	0.0060316	0.093213	0.19567	0.30599
r3	0	0	0	0.0060204	0.09308	0.19546	0.30574
r4	0	0	0	0.0059964	0.092798	0.19501	0.3052
r5	0	0	0	0.012229	0.092335	0.19427	0.30431
r96	0.90049	0.89703	0.89253	0.88669	0.90296	0.89297	0.90379
r97	0.90128	0.89791	0.89351	0.88781	0.90404	0.89429	0.88181
r98	0.90205	0.89876	0.89447	0.88891	0.9051	0.89557	0.88337
r99	0.9028	0.89959	0.89541	0.88998	0.90612	0.89681	0.88489
r100	0.90354	0.9004	0.89632	0.89101	0.90711	0.89802	0.88636



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 0.152901 seconds.

XX

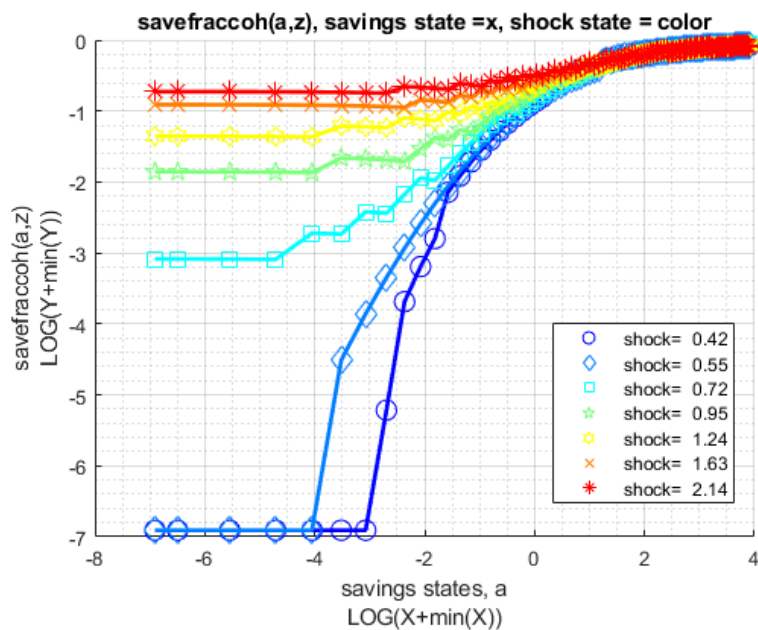
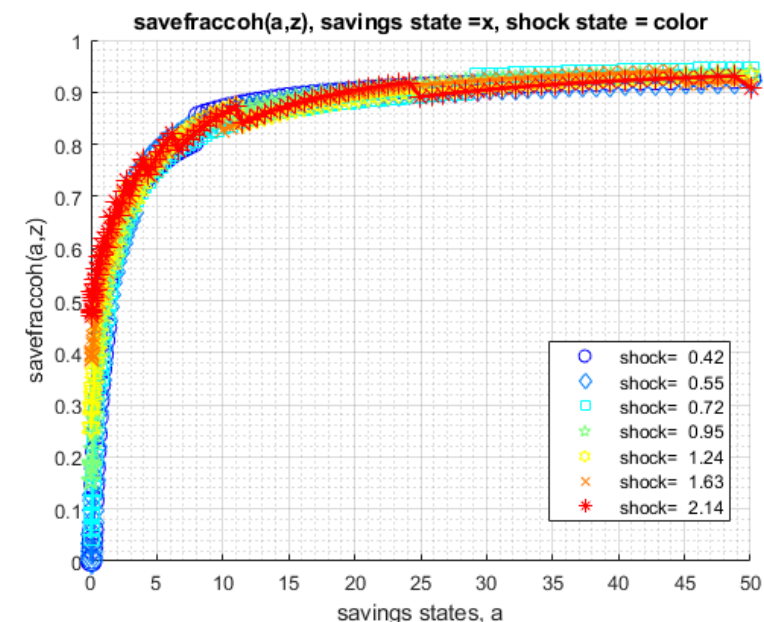
CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)

XX

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coefvari	min	max
savefraccoh	1	1	2	700	100	7	500.59	0.71513	0.25488	0.35641	0	0.9

xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c6	c7
r1	0	0	0.044811	0.15534	0.25694	0.40177	0.48276
r2	0	0	0.044787	0.15528	0.25686	0.40168	0.48268
r3	0	0	0.044678	0.15499	0.2565	0.40124	0.48228
r4	0	0	0.044445	0.15437	0.25572	0.40032	0.48143
r5	0	0	0.064784	0.15337	0.25445	0.39879	0.48003
r96	0.92489	0.92134	0.94129	0.93513	0.92717	0.91691	0.92776
r97	0.92544	0.92198	0.9418	0.9358	0.92802	0.91801	0.92895
r98	0.92598	0.9226	0.9423	0.93644	0.92885	0.91908	0.9301
r99	0.9265	0.9232	0.94278	0.93706	0.92965	0.92011	0.93121
r100	0.927	0.92379	0.94324	0.93765	0.93042	0.9211	0.90914



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') = 150;
mp_params('it_z_n') = 15;
mp_params('fl_a_max') = 50;
mp_params('st_grid_type') = 'grid_powerspace';
% graph color spectrum
mp_params('cl_colors') = 'copper';

```

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 0.544499 seconds.

XX

CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)

XX

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coefvari	min	max
savefraccoh	1	1	2	2250	150	15	1506.3	0.66947	0.28673	0.4283	0	0

xxx TABLE:savefraccoh XXXXXXXXXXXXXXXXXXXXXXX

	c1	c2	c3	c4	c5	c11	c12	c13	c14	c15
r1	0	0	0	0	0	0.14061	0.1891	0.24154	0.2699	0.324
r2	0	0	0	0	0	0.1406	0.18908	0.24152	0.26988	0.324
r3	0	0	0	0	0	0.14053	0.189	0.24142	0.26977	0.324
r4	0	0	0	0	0	0.14038	0.18881	0.2412	0.26956	0.324
r5	0	0	0	0	0	0.14013	0.18851	0.24085	0.2692	0.323
r146	0.93087	0.92957	0.92815	0.92661	0.92492	0.92712	0.92403	0.92069	0.91706	0.913
r147	0.93121	0.92994	0.92854	0.92702	0.92537	0.92768	0.92465	0.92135	0.91778	0.913
r148	0.93156	0.9303	0.92893	0.92743	0.92581	0.92823	0.92525	0.92201	0.91849	0.914
r149	0.93189	0.93065	0.9293	0.92783	0.92623	0.92878	0.92584	0.92264	0.91918	0.915
r150	0.93222	0.931	0.92967	0.92823	0.92665	0.9293	0.92641	0.92327	0.91986	0.916

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 0.515060 seconds.

XX

CONTAINER NAME: mp_ffcmd ND Array (Matrix etc)

XX

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coefvari	min	
	—	—	—	—	—	—	—	—	—	—	—	—
savefraccoh	1	1	2	2250	150	15	1678.8	0.74614	0.22779	0.30529	0	0
xxx TABLE:savefraccoh xxxxxxxxxxxxxxxxxxxxx												
	c1	c2	c3	c4	c5	c11	c12	c13	c14	c15		
	—	—	—	—	—	—	—	—	—	—		
r1	0	0	0	0	0	0.53612	0.59853	0.67884	0.73891	0.776		
r2	0	0	0	0	0	0.53609	0.5985	0.67882	0.73889	0.776		
r3	0	0	0	0	0	0.53594	0.59839	0.67873	0.73883	0.776		
r4	0	0	0	0	0	0.53563	0.59814	0.67853	0.73868	0.776		
r5	0	0	0	0	0	0.53511	0.59774	0.67821	0.73843	0.77		
r146	0.92696	0.9262	0.92513	0.92359	0.92142	0.91653	0.9078	0.88992	0.86057	0.80		
r147	0.92721	0.92647	0.92541	0.9239	0.92176	0.91741	0.90895	0.89144	0.84828	0.793		
r148	0.92746	0.92673	0.92569	0.92421	0.9221	0.91827	0.91007	0.87813	0.83621	0.782		
r149	0.9277	0.92698	0.92596	0.9245	0.92243	0.9191	0.89605	0.86507	0.82436	0.772		
r150	0.92794	0.92724	0.92623	0.9248	0.92276	0.90467	0.88233	0.85227	0.81273	0.762		