

FF_OPTIM_MLSEC_SAVEZRONE Derivative Multisection

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

This is the example vignette for function: `ff_optim_mlsec_savezrone` from the [MEconTools Package](#). This functions solves for optimal savings/borrowing level given an anonymous function that provides the derivative of a intertemporal savings problem. This is a vectorized function solved with multi-section (multiple points bisection concurrently).

The vectorized and looped bisection savings problem rely on this function to solve for optimal savings choices:

- States Grid + Continuous Exact Savings as Share of Cash-on-Hand **Loop**: `ff_vfi_az_bisec_loop`, high precision even with small grid
- States Grid + Continuous Exact Savings as Share of Cash-on-Hand **Vectorized**: `ff_vfi_az_bisec_vec`, precision and speed

Test FF_OPTIM_MLSEC_SAVEZRONE One Individual

Bisection for savings choice at one state:

```
% Generate the state-space and function
[fl_z1, fl_z2, fl_r, fl_beta] = deal(0.4730, 0.6252, 0.0839, 0.7365);
% ffi_intertemporal_max is a function in ff_optim_mlsec_savezrone for testing
fc_der_iwth_uniroot = @(x) ffi_intertemporal_max(x, fl_z1, fl_z2, fl_r, fl_beta);
% Call Function
bl_verbose = false;
bl_timer = true;
% optimally borrowing given the parameters here
mp_mlsec_ctrlinfo = containers.Map('KeyType','char','ValueType','any');
mp_mlsec_ctrlinfo('it_mzoom_jnt_pnts') = 10;
mp_mlsec_ctrlinfo('it_mzoom_max_iter') = 4;
[fl_opti_save_frac, fl_opti_save_level] = ...
    ff_optim_mlsec_savezrone(fc_der_iwth_uniroot, bl_verbose, bl_timer, mp_mlsec_ctrlinfo)
```

```
Elapsed time is 0.011265 seconds.
fl_opti_save_frac = 0.4241
fl_opti_save_level = -0.1316
```

Test FF_OPTIM_MLSEC_SAVEZRONE 5 Individuals 5 Iterations 5 Points Per Iteration

5 grid points per iteration, and 5 iterations.

```
% Generate the state-space and function
rng(123);
it_draws = 6; % must be even number
ar_z1 = exp(rand([it_draws,1])*3-1.5);
ar_z2 = exp(rand([it_draws,1])*3-1.5);
ar_r = (rand(it_draws,1)*10.0);
ar_beta = [rand(round(it_draws/2),1)*0.9+0.1; rand(round(it_draws/2),1)*0.9+1];
fc_der_iwth_uniroot = @(x) ffi_intertemporal_max(x, ar_z1, ar_z2, ar_r, ar_beta);
% Call Function
```

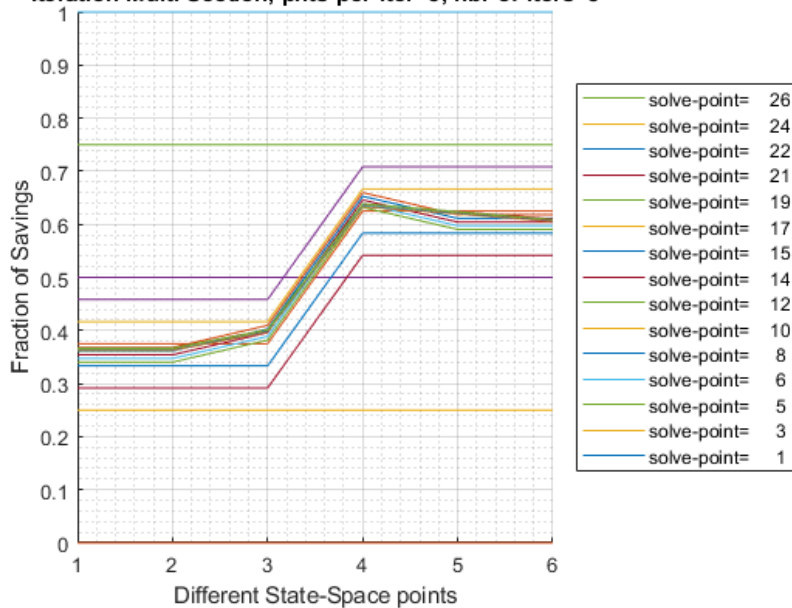
```

bl_verbose = true;
bl_timer = true;
mp_mlsec_ctrlinfo = containers.Map('KeyType','char','ValueType','any');
mp_mlsec_ctrlinfo('it_mlsect_jnt_pnts') = 5;
mp_mlsec_ctrlinfo('it_mlsect_max_iter') = 5;
ff_optim_mlsec_savezrone(fc_der_iwth_uniroot, bl_verbose, bl_timer, mp_mlsec_ctrlinfo);

```

| iter | cl_row_names_a | Var1 | Var2 | Var3 | Var4 | Var5 | Var6 |
|------|----------------|---------|---------|---------|---------|---------|---------|
| 0 | "point=1" | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 |
| 1 | "point=1" | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 |
| 1 | "point=2" | 0.25001 | 0.25001 | 0.25001 | 0.25001 | 0.25001 | 0.25001 |
| 1 | "point=3" | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| 1 | "point=4" | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 | 0.75 |
| 1 | "point=5" | 0.99999 | 0.99999 | 0.99999 | 0.99999 | 0.99999 | 0.99999 |
| 2 | "point=1" | 0.29167 | 0.29167 | 0.29167 | 0.54167 | 0.54167 | 0.54167 |
| 2 | "point=2" | 0.33334 | 0.33334 | 0.33334 | 0.58333 | 0.58333 | 0.58333 |
| 2 | "point=3" | 0.375 | 0.375 | 0.375 | 0.625 | 0.625 | 0.625 |
| 2 | "point=4" | 0.41667 | 0.41667 | 0.41667 | 0.66666 | 0.66666 | 0.66666 |
| 2 | "point=5" | 0.45833 | 0.45833 | 0.45833 | 0.70833 | 0.70833 | 0.70833 |
| 3 | "point=1" | 0.34028 | 0.34028 | 0.38195 | 0.63194 | 0.59028 | 0.59028 |
| 3 | "point=2" | 0.34723 | 0.34723 | 0.38889 | 0.63889 | 0.59722 | 0.59722 |
| 3 | "point=3" | 0.35417 | 0.35417 | 0.39584 | 0.64583 | 0.60416 | 0.60416 |
| 3 | "point=4" | 0.36111 | 0.36111 | 0.40278 | 0.65277 | 0.61111 | 0.61111 |
| 3 | "point=5" | 0.36806 | 0.36806 | 0.40972 | 0.65972 | 0.61805 | 0.61805 |
| 4 | "point=1" | 0.36227 | 0.36227 | 0.39699 | 0.6331 | 0.61921 | 0.60532 |
| 4 | "point=2" | 0.36343 | 0.36343 | 0.39815 | 0.63426 | 0.62037 | 0.60648 |
| 4 | "point=3" | 0.36459 | 0.36459 | 0.39931 | 0.63541 | 0.62153 | 0.60764 |
| 4 | "point=4" | 0.36574 | 0.36574 | 0.40046 | 0.63657 | 0.62268 | 0.60879 |
| 4 | "point=5" | 0.3669 | 0.3669 | 0.40162 | 0.63773 | 0.62384 | 0.60995 |
| 5 | "point=1" | 0.36594 | 0.36594 | 0.40066 | 0.63792 | 0.62288 | 0.60783 |
| 5 | "point=2" | 0.36613 | 0.36613 | 0.40085 | 0.63811 | 0.62307 | 0.60802 |
| 5 | "point=3" | 0.36632 | 0.36632 | 0.40104 | 0.63831 | 0.62326 | 0.60822 |
| 5 | "point=4" | 0.36652 | 0.36652 | 0.40124 | 0.6385 | 0.62345 | 0.60841 |
| 5 | "point=5" | 0.36671 | 0.36671 | 0.40143 | 0.63869 | 0.62365 | 0.6086 |

Iteration Multi-Section, pnts-per-iter=5, nbr-of-its=5



Elapsed time is 0.495996 seconds.

```

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

```

| | i | idx | ndim | numel | rowN | colN | sum | mean | std | co |
|-------------------|---|-----|------|-------|------|------|-------------|-------------|------------|----|
| | — | — | — | — | — | — | — | — | — | — |
| ar_opti_foc_obj | 1 | 1 | 2 | 6 | 6 | 1 | -0.00037648 | -6.2746e-05 | 0.00042601 | -6 |
| ar_opti_save_frac | 2 | 2 | 2 | 6 | 6 | 1 | 3.0037 | 0.50061 | 0.13506 | 0. |

xxx TABLE:ar_opti_foc_obj xxxxxxxxxxxxxxxxxxxx
c1

```

r1  7.0837e-05
r2  -0.0002782
r3  0.00017713
r4  0.00055875
r5  -0.00023392
r6  -0.00067107

```

xxx TABLE:ar_opti_save_frac xxxxxxxxxxxxxxxxxxxx
c1

```

r1  0.36642
r2  0.36661
r3  0.40153
r4  0.63821
r5  0.62297
r6  0.60793

```

Test FF_OPTIM_MLSEC_SAVEZRONE 8 Individuals 3 Iterations 10 Points Per Iteration

10 grid points per iteration, and 3 iterations.

```

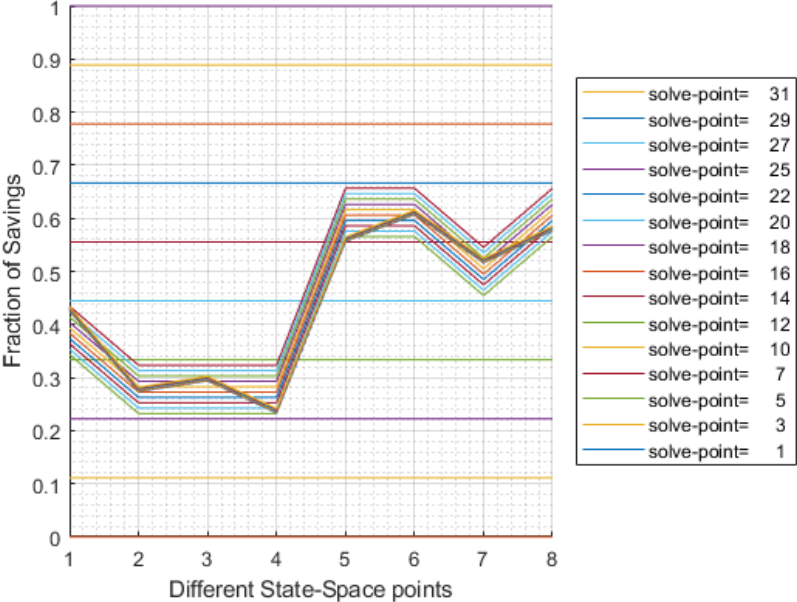
% Generate the state-space and function
rng(123);
it_draws = 8; % must be even number
ar_z1 = exp(rand([it_draws,1])*3-1.5);
ar_z2 = exp(rand([it_draws,1])*3-1.5);
ar_r = (rand(it_draws,1)*10.0);
ar_beta = [rand(round(it_draws/2),1)*0.9+0.1; rand(round(it_draws/2),1)*0.9+1];
fc_deriwth_uniroot = @(x) ffi_intertemporal_max(x, ar_z1, ar_z2, ar_r, ar_beta);
% Call Function
bl_verbose = true;
bl_timer = true;
mp_mlsec_ctrlinfo = containers.Map('KeyType','char','ValueType','any');
mp_mlsec_ctrlinfo('it_mlsect_jnt_pnts') = 10;
mp_mlsec_ctrlinfo('it_mlsect_max_iter') = 3;
ff_optim_mlsec_savezrone(fc_deriwth_uniroot, bl_verbose, bl_timer, mp_mlsec_ctrlinfo);

```

| iter | c1_row_names_a | Var1 | Var2 | Var3 | Var4 | Var5 | Var6 | Var7 | Var8 |
|------|----------------|---------|---------|---------|---------|---------|---------|---------|---------|
| — | — | — | — | — | — | — | — | — | — |
| 0 | "point=1" | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 |
| 1 | "point=1" | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 | 1e-05 |
| 1 | "point=2" | 0.11112 | 0.11112 | 0.11112 | 0.11112 | 0.11112 | 0.11112 | 0.11112 | 0.11112 |
| 1 | "point=3" | 0.22223 | 0.22223 | 0.22223 | 0.22223 | 0.22223 | 0.22223 | 0.22223 | 0.22223 |
| 1 | "point=4" | 0.33334 | 0.33334 | 0.33334 | 0.33334 | 0.33334 | 0.33334 | 0.33334 | 0.33334 |
| 1 | "point=5" | 0.44445 | 0.44445 | 0.44445 | 0.44445 | 0.44445 | 0.44445 | 0.44445 | 0.44445 |
| 1 | "point=6" | 0.55555 | 0.55555 | 0.55555 | 0.55555 | 0.55555 | 0.55555 | 0.55555 | 0.55555 |
| 1 | "point=7" | 0.66666 | 0.66666 | 0.66666 | 0.66666 | 0.66666 | 0.66666 | 0.66666 | 0.66666 |

| | | | | | | | | | |
|---|------------|---------|---------|---------|---------|---------|---------|---------|---------|
| 1 | "point=8" | 0.77777 | 0.77777 | 0.77777 | 0.77777 | 0.77777 | 0.77777 | 0.77777 | 0.77777 |
| 1 | "point=9" | 0.88888 | 0.88888 | 0.88888 | 0.88888 | 0.88888 | 0.88888 | 0.88888 | 0.88888 |
| 1 | "point=10" | 0.99999 | 0.99999 | 0.99999 | 0.99999 | 0.99999 | 0.99999 | 0.99999 | 0.99999 |
| 2 | "point=1" | 0.34344 | 0.23233 | 0.23233 | 0.23233 | 0.56566 | 0.56566 | 0.45455 | 0.56566 |
| 2 | "point=2" | 0.35354 | 0.24243 | 0.24243 | 0.24243 | 0.57576 | 0.57576 | 0.46465 | 0.57576 |
| 2 | "point=3" | 0.36364 | 0.25253 | 0.25253 | 0.25253 | 0.58586 | 0.58586 | 0.47475 | 0.58586 |
| 2 | "point=4" | 0.37374 | 0.26263 | 0.26263 | 0.26263 | 0.59596 | 0.59596 | 0.48485 | 0.59596 |
| 2 | "point=5" | 0.38384 | 0.27273 | 0.27273 | 0.27273 | 0.60606 | 0.60606 | 0.49495 | 0.60606 |
| 2 | "point=6" | 0.39394 | 0.28283 | 0.28283 | 0.28283 | 0.61616 | 0.61616 | 0.50505 | 0.61616 |
| 2 | "point=7" | 0.40404 | 0.29293 | 0.29293 | 0.29293 | 0.62626 | 0.62626 | 0.51515 | 0.62626 |
| 2 | "point=8" | 0.41414 | 0.30303 | 0.30303 | 0.30303 | 0.63636 | 0.63636 | 0.52525 | 0.63636 |
| 2 | "point=9" | 0.42424 | 0.31314 | 0.31314 | 0.31314 | 0.64646 | 0.64646 | 0.53535 | 0.64646 |
| 2 | "point=10" | 0.43434 | 0.32324 | 0.32324 | 0.32324 | 0.65656 | 0.65656 | 0.54545 | 0.65656 |
| 3 | "point=1" | 0.42516 | 0.27365 | 0.29385 | 0.23325 | 0.55647 | 0.60698 | 0.51607 | 0.57667 |
| 3 | "point=2" | 0.42608 | 0.27457 | 0.29477 | 0.23417 | 0.55739 | 0.60789 | 0.51699 | 0.57759 |
| 3 | "point=3" | 0.427 | 0.27549 | 0.29569 | 0.23508 | 0.55831 | 0.60881 | 0.51791 | 0.57851 |
| 3 | "point=4" | 0.42792 | 0.2764 | 0.29661 | 0.236 | 0.55923 | 0.60973 | 0.51882 | 0.57943 |
| 3 | "point=5" | 0.42884 | 0.27732 | 0.29752 | 0.23692 | 0.56015 | 0.61065 | 0.51974 | 0.58035 |
| 3 | "point=6" | 0.42975 | 0.27824 | 0.29844 | 0.23784 | 0.56106 | 0.61157 | 0.52066 | 0.58127 |
| 3 | "point=7" | 0.43067 | 0.27916 | 0.29936 | 0.23876 | 0.56198 | 0.61249 | 0.52158 | 0.58218 |
| 3 | "point=8" | 0.43159 | 0.28008 | 0.30028 | 0.23967 | 0.5629 | 0.6134 | 0.5225 | 0.5831 |
| 3 | "point=9" | 0.43251 | 0.281 | 0.3012 | 0.24059 | 0.56382 | 0.61432 | 0.52342 | 0.58402 |
| 3 | "point=10" | 0.43343 | 0.28191 | 0.30212 | 0.24151 | 0.56474 | 0.61524 | 0.52433 | 0.58494 |

Iteration Multi-Section, pnts-per-iter=10, nbr-of-iters=3



Elapsed time is 0.486844 seconds.

CONTAINER NAME: mp_container_map ND Array (Matrix etc)

| | i | idx | ndim | numel | rowN | colN | sum | mean | std | coefva |
|-------------------|---|-----|------|-------|------|------|-----------|------------|-----------|--------|
| ar_opti_foc_obj | 1 | 1 | 2 | 8 | 8 | 1 | 0.0033175 | 0.00041468 | 0.0029592 | 7.136 |
| ar_opti_save_frac | 2 | 2 | 2 | 8 | 8 | 1 | 3.5124 | 0.43905 | 0.15005 | 0.3417 |

xxx TABLE:ar_opti_foc_obj xxxxxxxxxxxxxxxxxxxx

| | |
|----|------------|
| r1 | 0.00087102 |
| r2 | 0.0033354 |
| r3 | -0.0044871 |
| r4 | 0.001317 |

```

r5      -0.0017862
r6       0.0050249
r7      -0.00058496
r8      -0.00037273

```

```

xxx TABLE:ar_opti_save_frac xxxxxxxxxxxxxxxxxxxx
      c1

```

```

r1      0.42838
r2      0.28054
r3       0.2989
r4      0.23371
r5      0.55877
r6      0.61019
r7      0.5202
r8      0.58172

```

Test FF_OPTIM_MLSEC_SAVEZRONE Speed

Test Speed doing 6.25 million multisections for a savings problem:

```

% Generate the state-space and function
rng(123);
it_draws = 6250000; % must be even number
ar_z1 = exp(rand([it_draws,1])*3-1.5);
ar_z2 = exp(rand([it_draws,1])*3-1.5);
ar_r = (rand(it_draws,1)*10.0);
ar_beta = [rand(round(it_draws/2),1)*0.9+0.1; rand(round(it_draws/2),1)*0.9+1];
% ffi_intertemporal_max is a function in ff_optim_mlsec_savezrone for testing
fc_deriwth_uniroot = @(x) ffi_intertemporal_max(x, ar_z1, ar_z2, ar_r, ar_beta);
% Call Function
bl_verbose = false;
bl_timer = true;
[ar_opti_save_frac, ar_opti_save_level] = ff_optim_mlsec_savezrone(fc_deriwth_uniroot, bl_verbose);

```

Elapsed time is 16.390434 seconds.

```

mp_container_map = containers.Map('KeyType','char','ValueType','any');
mp_container_map('ar_opti_save_frac') = ar_opti_save_frac;
mp_container_map('ar_opti_save_level') = ar_opti_save_level;
mp_container_map('ar_opti_save_frac_notnan') = ar_opti_save_frac(~isnan(ar_opti_save_frac));
ff_container_map_display(mp_container_map);

```

```

-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_container_map ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

```

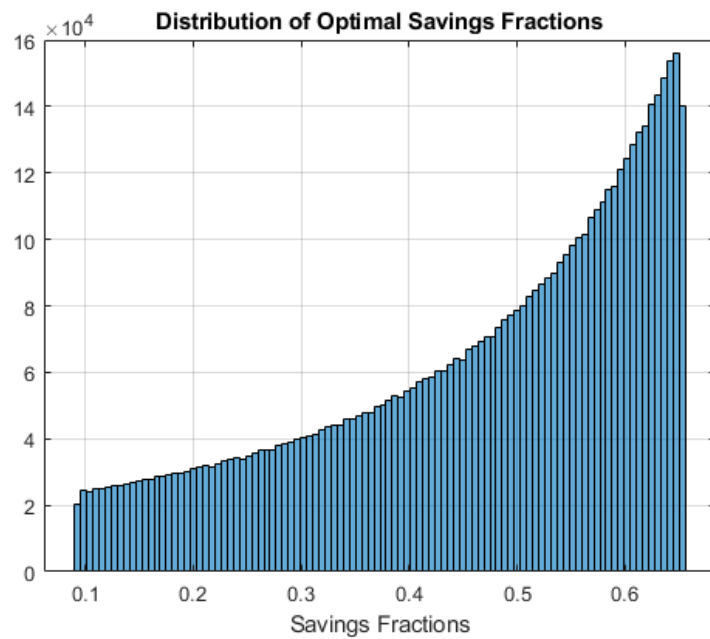
| | i | idx | ndim | numel | rowN | colN | sum | mean | std |
|--------------------------|---|-----|------|----------|----------|------|------------|---------|---------|
| ar_opti_save_frac | 1 | 1 | 2 | 6.25e+06 | 6.25e+06 | 1 | 2.884e+06 | 0.46144 | 0.15306 |
| ar_opti_save_frac_notnan | 2 | 2 | 2 | 6.25e+06 | 6.25e+06 | 1 | 2.884e+06 | 0.46144 | 0.15306 |
| ar_opti_save_level | 3 | 3 | 2 | 6.25e+06 | 6.25e+06 | 1 | 2.9482e+06 | 0.47172 | 0.66667 |

```

figure();
histogram(ar_opti_save_frac(~isnan(ar_opti_save_frac)),100);
title('Distribution of Optimal Savings Fractions');
xlabel('Savings Fractions');

```

```
grid on;
```



Define Two Period Intertemporal FOC Log Utility No Shock

See [Household's Utility Maximization Problem and Two-Period Borrowing and Savings Problem given Endowments](#).

```
function [ar_der_i_zero, ar_saveborr_level] = ffi_intertemporal_max(ar_saveborr_frac, z1, z2, r,  
    ar_saveborr_level = ar_saveborr_frac.*(z1+z2./(1+r)) - z2./(1+r);  
    ar_der_i_zero = 1./(ar_saveborr_level-z1) + (beta.*(r+1))./(z2 + ar_saveborr_level.*(r+1));  
end
```