Small Test Exact Solution Looped Minimizer

This is the example vignette for function: **snw_vfi_main** from the **PrjOptiSNW Package.** This function solves for policy function fully iteratively using matlab minimizer. Small Solution Analysis.This produces the same result as **snw_vfi_main_bisec_vec,** except slower. The purpose of this function is to confirm that the results from **snw_vfi_main_bisec_vec** is correct.

Test SNW_VFI_MAIN Defaults Small

Call the function with defaults parameters.

```
mp_param = snw_mp_param('default_small');
[V_VFI,ap_VFI,cons_VFI,mp_valpol_more] = snw_vfi_main(mp_param);
```

```
SNW_VFI_MAIN: Finished Age Group:18 of 18
SNW_VFI_MAIN: Finished Age Group:17 of 18
SNW_VFI_MAIN: Finished Age Group:16 of 18
SNW_VFI_MAIN: Finished Age Group:15 of 18
SNW_VFI_MAIN: Finished Age Group:14 of 18
SNW_VFI_MAIN: Finished Age Group:13 of 18
SNW_VFI_MAIN: Finished Age Group:12 of 18
SNW_VFI_MAIN: Finished Age Group:11 of 18
SNW_VFI_MAIN: Finished Age Group:10 of 18
SNW_VFI_MAIN: Finished Age Group:9 of 18
SNW_VFI_MAIN: Finished Age Group:8 of 18
SNW_VFI_MAIN: Finished Age Group:7 of 18
SNW_VFI_MAIN: Finished Age Group:6 of 18
SNW_VFI_MAIN: Finished Age Group:5 of 18
SNW_VFI_MAIN: Finished Age Group:4 of 18
SNW_VFI_MAIN: Finished Age Group:3 of 18
SNW_VFI_MAIN: Finished Age Group:2 of 18
SNW_VFI_MAIN: Finished Age Group:1 of 18
Elapsed time is 375.055636 seconds.
Completed SNW_VFI_MAIN;SNW_MP_PARAM=default_small;SNW_MP_CONTROL=default_base
```

Small Param Results Define Frames

Define the matrix dimensions names and dimension vector values. Policy and Value Functions share the same ND dimensional structure.

```
% Grids:
age_grid = [19, 22:5:97, 100];
agrid = mp_param('agrid')';
eta_H_grid = mp_param('eta_H_grid')';
eta_S_grid = mp_param('eta_S_grid')';
ar_st_eta_HS_grid = string(cellstr([num2str(eta_H_grid', 'hz=%3.2f;'), num2str(eta_S_grid', 'wz
edu_grid = [0,1];
marry_grid = [0,1];
kids_grid = (1:1:mp_param('n_kidsgrid'))';
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
cl_mp_datasetdesc = {};
cl_mp_datasetdesc{1} = containers.Map({'name', 'labval'}, {'age', age_grid});
cl_mp_datasetdesc{2} = containers.Map({'name', 'labval'}, {'savings', agrid});
cl_mp_datasetdesc{3} = containers.Map({'name', 'labval'}, {'eta', 1:length(eta_H_grid)});
cl_mp_datasetdesc{4} = containers.Map({'name', 'labval'}, {'edu', edu_grid});
```

```
cl_mp_datasetdesc{5} = containers.Map({'name', 'labval'}, {'marry', marry_grid});
cl_mp_datasetdesc{6} = containers.Map({'name', 'labval'}, {'kids', kids_grid});
```

Analyze Savings and Shocks

First, analyze Savings Levels and Shocks, Aggregate Over All Others, and do various other calculations.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('cl_st_xtitle') = {'Savings States, a'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
```

MEAN(VAL(A,Z)), MEAN(AP(A,Z)), MEAN(C(A,Z))

Tabulate value and policies along savings and shocks:

```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [1,4,5,6,3,2];
% Value Function
tb_az_v = ff_summ_nd_array("MEAN(VAL(A,Z))", V_VFI, true, ["mean"], 4, 1, cl_mp_datasetdesc, ar
```

```
xxx  MEAN(VAL(A,Z))  xxxxxxxxxxxxxxxxxxxxxxxxxxx
    group     savings     mean_eta_1    mean_eta_2    mean_eta_3    mean_eta_4    mean_eta_5
    _____    _________    __________    __________    __________    __________    __________

      1              0      -17.393       -9.1596       -4.4164       -1.5922      -0.05106
      2      0.0097656      -16.967        -9.023       -4.3405       -1.5316     0.0054257
      3       0.078125      -14.925       -8.2554       -3.9177       -1.2071        0.3028
      4        0.26367      -11.699       -6.8681       -3.1808       -0.6913       0.75178
      5          0.625      -8.2751       -5.1669       -2.2786      -0.13884        1.1911
      6         1.2207      -5.3024       -3.4437       -1.3431       0.38361        1.5638
      7         2.1094      -2.9816       -1.9066      -0.47798       0.86411        1.8672
      8         3.3496      -1.2609      -0.64407       0.28611        1.3001        2.1163
      9              5    -0.012548       0.34403        0.9369        1.6782        2.3266
     10         7.1191       0.8875         1.097        1.4725        1.9981        2.5086
     11         9.7656       1.5392         1.665        1.9037        2.2701        2.6684
     12         12.998       2.0158        2.0932        2.2465        2.5004        2.8071
     13         16.875       2.3684        2.4172        2.5172        2.6933        2.9263
     14         21.455       2.6328        2.6644        2.7307        2.8535        3.0288
     15         26.797       2.8339        2.8549        2.8997         2.986        3.1174
     16         32.959        2.989        3.0032         3.034        3.0954        3.1939
     17             40       3.1102          3.12        3.1416        3.1857        3.2598
     18         47.979       3.2059        3.2128        3.2282        3.2603        3.3164
     19         56.953       3.2825        3.2875        3.2986        3.3222        3.3649
     20         66.982       3.3443         3.348        3.3562        3.3738        3.4064
     21         78.125       3.3948        3.3975        3.4036        3.4169        3.4421
     22         90.439       3.4364        3.4384         3.443        3.4532        3.4728
     23         103.98       3.4709        3.4724         3.476        3.4838        3.4991
     24         118.82       3.4998         3.501        3.5037        3.5098        3.5219
     25            135       3.5241        3.5251        3.5272        3.5319        3.5416
```

```
% Aprime Choice
tb_az_ap = ff_summ_nd_array("MEAN(AP(A,Z))", ap_VFI, true, ["mean"], 4, 1, cl_mp_datasetdesc, a
```

```
xxx  MEAN(AP(A,Z))  xxxxxxxxxxxxxxxxxxxxxxxxxxx
    group     savings     mean_eta_1    mean_eta_2    mean_eta_3    mean_eta_4    mean_eta_5
    _____    _________    __________    __________    __________    __________    __________
```

```
      1              0    2.7521e-05    0.0021998      0.046507      0.23828       0.88717
      2      0.0097656    0.00054713    0.0036592      0.049526      0.24213       0.89277
      3       0.078125      0.021674     0.027305      0.079496      0.27478       0.93352
      4        0.26367       0.13129      0.14249       0.19452      0.38205        1.0523
      5          0.625       0.38725       0.4041       0.44789        0.639        1.3005
      6         1.2207       0.83381      0.85545       0.90674       1.0839         1.735
      7         2.1094        1.5206       1.5442        1.6064       1.7452        2.3859
      8         3.3496         2.477       2.5013        2.5629       2.6789        3.3301
      9              5        3.7541       3.7788        3.8405       3.9859        4.5828
     10         7.1191         5.416       5.4412        5.5038       5.6835        6.1821
     11         9.7656        7.4668       7.4912        7.5553       7.7413         8.177
     12         12.998        9.9008       9.9212        9.9832       10.174        10.619
     13         16.875        12.918        12.94        12.995       13.186        13.709
     14         21.455        16.519       16.538        16.594       16.772        17.365
     15         26.797         20.59       20.608        20.657       20.825        21.451
     16         32.959        25.295       25.313        25.358       25.513        26.139
     17             40        30.657        30.68        30.732       30.877        31.477
     18         47.979        36.752       36.772        36.831        36.99        37.553
     19         56.953        43.764       43.786        43.839       44.003        44.551
     20         66.982        51.595       51.618        51.678        51.84        52.393
     21         78.125        59.943       59.966        60.026       60.198        60.755
     22         90.439        69.256        69.28        69.342       69.517        70.086
     23         103.98        79.744       79.765        79.824       79.998        80.576
     24         118.82        91.106        91.13        91.192       91.358        91.933
     25            135        103.46       103.48        103.54       103.71        104.28
```

```
% Consumption Choices
tb_az_c = ff_summ_nd_array("MEAN(C(A,Z))", cons_VFI, true, ["mean"], 4, 1, cl_mp_datasetdesc, a
```

```
xxx  MEAN(C(A,Z))  xxxxxxxxxxxxxxxxxxxxxxxxxxx
    group     savings     mean_eta_1    mean_eta_2    mean_eta_3    mean_eta_4    mean_eta_5
    _____    _________    __________    __________    __________    __________    __________

      1              0      0.3104          0.44       0.69882        1.2297        2.3502
      2      0.0097656      0.3214       0.45001       0.70723        1.2373         2.356
      3       0.078125      0.3809       0.50664       0.75722        1.2844        2.3949
      4        0.26367     0.48992       0.60921       0.85919        1.3936        2.4924
      5          0.625     0.65895       0.77122        1.0281        1.5581        2.6654
      6         1.2207     0.91142        1.0172        1.2649        1.8076        2.9247
      7         2.1094      1.2649        1.3671        1.6019        2.1815        3.3081
      8         3.3496      1.7572        1.8573        2.0907        2.6915        3.8066
      9              5      2.4045         2.503        2.7347        3.3043        4.4728
     10         7.1191      3.2104        3.3074         3.537        4.0708        5.3364
     11         9.7656      4.2385        4.3358        4.5627        5.0889        6.4164
     12         12.998      5.5627        5.6635        5.8917        6.4121        7.7296
     13         16.875      7.0504        7.1499        7.3847         7.904        9.1419
     14         21.455      8.7708        8.8721        9.1059        9.6366        10.804
     15         26.797      10.904        11.007        11.246        11.787        12.921
     16         32.959      13.355        13.457          13.7        14.254        15.388
     17             40      16.168        16.266        16.502        17.066        18.225
     18         47.979      19.337        19.437        19.666        20.215        21.411
     19         56.953      22.744        22.843        23.078        23.621        24.831
     20         66.982      26.557        26.654        26.882        27.427        28.632
     21         78.125      31.144        31.241        31.469        32.005        33.205
     22         90.439      36.126        36.222        36.449        36.981        38.169
     23         103.98      41.361         41.46        41.689        42.223        43.402
     24         118.82      47.219        47.315        47.541        48.083        49.265
     25            135      53.648        53.747        53.978        54.513        55.702
```