

NSW Dynamic Life Cycle and Welfare Checks Code Companion

Vegard M. Nygård, Bent E. Sørensen, and Fan Wang

2020-11-11

Contents

Preface	5
1 Introduction	7
1.1 Household Problem and Distributions	7
1.2 Values of Checks Conditional on 2019 Information	9
1.3 The Welfare Check Planning Problem	10
2 Parameters	13
2.1 Model Parameters	13
2.2 Model Controls	38
3 Solving the Dynamic Life Cycle Problem	41
3.1 Life Cycle Dynamic Programming with Marital Status, Children and Savings	41
4 Alternative Value Function Solution Testing	57
4.1 Small Test Exact Solution Looped Minimizer	57
4.2 Small Test Grid Search Solution	60
4.3 Small Test Exact Solution Vectorized Bisection	74
4.4 Small Test Exact Solution Spousal Shocks	88
5 Solution with Unemployment	103
5.1 Life Cycle Dynamic Programming under Unemployment Shock	103
6 Household Life Cycle Distribution	123
6.1 Distribution Exact Savings Choices	123
6.2 Distribution Exact Savings Choices Vectorized	134
6.3 Distribution Grid Search	140
7 Value of Each Check	153
7.1 2020 V and C without Unemployment	153
7.2 2020 V and C with Unemployment	165
8 2020 Outcomes Full State Space with Savings, Shocks and Education	179
8.1 2020 Full States EV and EC of One Check	179
8.2 2019 Full States EV and EC of One Check	198
9 2019 Expectations Given Income, Age, Kids and Marital Status	221
9.1 2019 Age, Income, Kids, Marry EV and EC of One Check	221
9.2 2019 Age, Income, Kids, Marry EV and EC All Checks	228
10 Taxes	233
10.1 Compute for Equilibrium Tax	233
11 Calibration	239
11.1 Model Calibration	239
12 Summary Statistics	243
12.1 2019 Full States MPC and Distributional Statistics by Marital, Kids, and Income Groups.	243

A Index and Code Links	285
A.1 Introduction links	285
A.2 Parameters links	285
A.3 Solving the Dynamic Life Cycle Problem links	285
A.4 Alternative Value Function Solution Testing links	285
A.5 Solution with Unemployment links	286
A.6 Household Life Cycle Distribution links	286
A.7 Value of Each Check links	287
A.8 Outcomes Full State Space with Savings, Shocks and Education links	287
A.9 Expectations Given Income, Age, Kids and Marital Status links	287
A.10 Taxes links	287
A.11 Calibration links	288
A.12 Summary Statistics links	288

Preface

This is a work-in-progress Matlab package consisting of functions that solve the dynamic life cycle model in Nygård, Sørensen and Wang (2020) (Nygård et al., 2020). The code companion presents solutions to the dynamic life-cycle problem, and methods for evaluating the marginal gains from allocating additional welfare checks. Tested with Matlab 2019a (The MathWorks Inc, 2019).

All functions are parts of a matlab toolbox that can be installed:

Download and install the Matlab toolbox: [PrjOptiSNW.mltbx](#)

The Code Companion can also be accessed via the bookdown site and PDF linked below:

[bookdown pdf](#), [MathWorks File Exchange](#)

This bookdown file is a collection of mlx based vignettes for functions that are available from [PrjOptiSNW](#). Each Vignette file contains various examples for invoking each function.

The package relies on [MEconTools](#), which needs to be installed first. The package does not include allocation functions, only simulation code to generate the value of each welfare check increments for households. Allocation functions rely the R optimal allocation package [PrjOptiAlloc](#).

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

Please contact [FanWangEcon](#) for issues or problems.

Chapter 1

Introduction

1.1 Household Problem and Distributions

In Nygaard, Sorensen and Wang (2020), we study the optimal allocation of COVID-19 welfare checks. Congress spent \$250 billion sending checks to individuals in March 2020 to provide economic stimulus. Could the same amount of stimulus have been achieved for less money? Using a life-cycle consumption-saving model with heterogeneous consumers, we calculate the consumption responses to cash transfers for, e.g., couples and singles with different levels of income and number of children. We calculate the aggregate consumption response for all feasible allocations of \$250 billion and, using a new algorithm that allows for the ranking of an arbitrarily large number of allocations, we find the optimal allocation under alternative constraints. The optimal policy allocates more toward low-income and younger consumers and can achieve the same stimulus effect at almost half the cost.

This Matlab based programming guide, package, and associated vignettes, provide examples and instructions on how the dynamic programming problem in Nygaard, Sorensen and Wang (2020) is solved. The R optimal allocation package [PrjOptiAlloc](#) takes inputs from the dynamic programming problems and solves for optimal allocations given varying planner objectives and constraints.

1.1.1 Flat Script and Code Package

There are two broad versions of the code. A number of files are included in the [zflat](#) folder, including the operation gateway file [main](#). Files in the zflat folder provides a linear, easier to understand illustration/demonstration of the overall code structure. It is useful to review the overall algorithm design. However, it should not be called to implement the programs. Programs in the folder were written to help test out algorithm ideas.

The rest of the files inside [PrjOptiSNW](#) form a matlab [package](#) that can be downloaded and installed. Each component of the overall code program is programmed up separately with its own testing vignette and default parameter structure. Various solution algorithms are provided at each step, with the final checks problem relying on efficient and precise solution methods.

1.1.2 Dynamic Programming Solution Structure COVIDless World

First we solve for the optimal consumption/savings problem in the COVID-less world:

- **83**: 2020 age groups, age 18 to 100 age groups
- **65**: grid of savings state-space grid, and exact continuous optimal savings choices using the [FF_VFI_AZ_BISEC_VEC](#) function from [MEconTools](#).
- **6650 shocks**: 1330 productivity shocks for household head and spouse and 5 kids transition count shocks
- **2** permanent education states
- **2** permanent marital states

The state-space has: $2^2 * 6650 * 65 * 83 = 143,507,000$ elements. The choice-space is continuous. Two important things to note:

1. The large number of shocks are needed to obtain accurate group-specific marginal propensity effects for small income bins that define the choice-set of the allocation problem.
2. While a choice-grid-based solution algorithm might sufficiently approximate the value function, but its policy function zig-zags. For the welfare checks problem, where welfare checks come in small increments, the zig-zags lead to fluctuating (negative and positive) marginal propensities to consume as resource availability increases for very small amounts of check increments. To deal with this challenge, we rely on the **FF_VFI_AZ_BISEC_VEC** function from **MEconTools** to provide efficient exact savings choices.

Solving this dynamic life-cycle programming problem requires approximately 10 to 20 minutes on a home-pc depending on computer speed. There are no processor requirements. Memory requirement is approximately 20GB. There are two core associated functions vignettes that solve the dynamic programming problem to obtain value/policy and distributions induced by exogenous processes and the policy function:

- Core dynamic programming code: [snwx_vfi_bisec_vec](#)
- Core distribution code: [snwx_ds_bisec_vec](#)

Small testing vignettes of alternative solution algorithms for policy/value:

- Small test using matlab minimizer (very slow but identical results as core program): [snwx_vfi_test](#)
- Small test using grid-search-based solution algorithm (insufficiently precise for welfare checks): [snwx_vfi_test_grid_search](#)
- Small test of core dynamic programming code: [snwx_vfi_test_bisec_vec](#)
- Small test of core dynamic programming code with spousal shock: [snwx_vfi_test_bisec_vec_spousalshock](#)

Testing vignettes for alternative solution algorithm for distribution:

- Grid search distributional code (insufficiently precise): [snwx_ds_grid_search](#)
- Core solution distribution code (vectorized for policy/value, looped for dist): [snwx_ds_bisec_vec_loop](#)
- Core solution distribution code (vectorized fully): [snwx_ds_bisec_vec](#)

1.1.3 Dynamic Programming Solution Structure during COVID Year

During the COVID year, we use the value function from the COVID-less world as the continuation value, and solve for consumption-savings policy/value functions during the COVID year. We solve once for households facing realized COVID surprise unemployment shocks, one more time for households who do not experience COVID unemployment shocks.

We solve for the marginal consumption differences and value given 244 increments of checks (\$100) each check. This is done again by using the **FF_VFI_AZ_BISEC_VEC** function from **MEconTools**. While checks could be viewed as an additional state variable, we evaluate the marginal effects of check by solving for the equivalent household-specific variation in savings state that has the same effect as a welfare check transfer. The process takes into account the nonlinear tax-schedule that households face as well as return on savings.

Overall:

- 286 million: Solve 143 million state-space points twice under COVID unemployment and COVID employment world
- 70 billion: Solve at the 143 million state-space elements $244 + 1$ times for all possible check levels (244 checks + no check value/consumption) to arrive at 70 billion marginal propensity to consume for households with heterogeneities in education level, marital status, children below 18 count (0 to 4), age, savings levels, household head and spouse shocks.

Associated functions vignettes: the core dynamic programming code: `snwx_vfi_bisec_vec`, has a third input which is the existing future value function. When this is provided, the dynamical programming problems solves for one period given already computed future value, and so the dynamic programming solution solves forward. When it is not provided, solves for value/policy backwards.

- `snwx_vfi_unemp_bisec_vec` provides the vignette given unemployment shock.
- `snwx_a4chk_wrk_bisec_vec` computes the marginal impacts of a particular welfare check increment for those without unemployment shock in COVID year.
- `snwx_a4chk_unemp_bisec_vec` computes the marginal impacts of a particular welfare check increment for those with unemployment shock in COVID year.
- `snwx_evuvw20_jaeemk` considers probabilities for getting hit with the COVID shock and considers the expected value conditional on age, savings level, shocks, educational status, kids count and marital status in 2020.

1.2 Values of Checks Conditional on 2019 Information

Eligibility for the stimulus checks was tied to each household's income and family size in the year prior to COVID-19. Consistent with this, we focus on the optimal allocation of stimulus checks given household characteristics in 2019.

1.2.1 Expected Outcomes given 2019 Information

In the actual welfare checks allocation policy setting, 2020 realized individual level COVID shocks and other productivity shocks were not used by the IRS to determine allocation eligibility. Instead, the IRS used information available from 2019. Given the persistence of productivity shocks, as well as the correlation between surprised COVID shock and household income and age from 2019, 2019 information are good predictors of household status in 2020.

We compute the expected outcomes from 2019 perspective conditional on household attributes that are observed to the IRS in 2020 given information they gathered in 2019. The planner does not observe the full state-space so we intergrate from 2019 perspective given 2019 to 2020 COVID transition probabilities (conditional on state-space) and kids and shock transitions.

- 82 age groups: Age 18 to 99
- 5 kids groups: Children 0 to 4
- 2 marital groups: Marital Status 0 or 1
- 509 Income Groups: 476 bins below max actual phaseout: solved at \$500 intervals between \$0 and \$238,000, and 33 bins after max actual phaseout: solved at \$5000 interval after \$238,000, where the 33rd final bin is between \$401,130 and Maximum.

Together, these are: $5*2*83*509 = 422,470$ groups/bins in 2019. We have (244+1) marginal average consumption gain, and value gain for each of the groups, so from 2019 planner perspective, we have 103.5 million expected MPCs (and expected value changes): $422470*(244+1)=103,505,150$.

Each income group is composed of individuals of with different 2019 productivity shocks and savings levels. Given the transition probabilities, policy functions, and covid-less distributions across household types, we can compute the joint distribution of education type, shocks, and savings levels condition on income bins, martial status, kids count and age. This joint distribution is only well approximated when sufficient number of shocks were used when solving for value/policy in the covid-less world and covid-year world.

- `snwx_evuvw19_jaeemk` provides the expected outcomes conditional on 2019 age, savings levels, shocks, educational status, marital status and kids under 18 count, given the transition probabilities that incorporate the surprise COVID shock. Households do not optimize in 2019 given COVID shock probabilities, the shock is a surprise. But expected value in 2019 given 2019 state-space for consumption and value depends on COVID shock and the non-covid households dynamic consumption/savings choices in 2019.

- `snwx_evuvw19_jmky` provides the expected value in 2019 given not the full state-space, but the state-space that is potentially known to the IRS in 2019: age, marital status, kids count and 2019 household income. Household income is a function of savings, shock and educational status.
- `snwx_evuvw19_jmky_allchecks` solves for the marginal incremental effects of a vector of checks for all household types, and stores the results to file. This operation is fully parallelizable.

Given the solution, `snwx_evuvw19_jaeemk_mky` shows key overall distributional statistics, and distributional statistics by kids, marital and income bins.

1.2.2 Dynamic Programming Timing

Overall time requirements for solving the checks problem on a standard (\$1000) dollar desktop with 4 cores:

- 30 minutes to 1 hour: solving the dynamic programming problem 3 times for covid-less world, for covid-world with unemployment shock and without.
- 3 hours: derive the distributions induced by policy functions and shock processes.
- 650 seconds: the time it takes to compute the marginal effects of one check. This step is fully scalable. With a cloud computer with larger memory and cores, the problem over all checks could be solved fully parallelly.

Due to the large state-space, especially the large shock-grid, the memory requirement for storing the various multi-dimensional matrixes is high. Solving the problem requires 20GB of memory.

On a workstation with 12 cores with 190 GB of memory (12 workers same time), the computer is able to fully solve the problem from start to finish for 244 check increments in less than 24 hours. On a computer with 4 to 6 cores and 36 GB of memory, the computer is able to fully solve the problem from start to finish for 244 check increments in about 48 hours.

1.3 The Welfare Check Planning Problem

The planner chooses the amount of stimulus checks for each group, where groups are defined by marital status, number of children, income, and age in 2019.

1.3.1 2019 Information Planning Problem

Given the expected outcomes we computed conditional on 2019 information, we can solve the planning problem. We have a number of different planning problems that we solve given different individual level constraints and what the planner can condition allocations on.

For FEASIBLE allocation, there are **970=5*2*97** types/cells of households:

- 5 children groups
- 2 spousal groups
- 97 income bins: the allocation planner sees approximately \$2500 income bins between \$0 and \$238,800, and 1 bin after \$238,800. There are 97 bins

for OPTIMAL G4 (4 age groups 18 to 64) allocation, there are **3880=5*2*97*4** types/cells of households:

- 5 children groups
- 2 spousal groups
- 4 age groups
- 97 income bins

for OPTIMAL G47 (47 age groups) allocation, there are **45590=5*2*97*47** types/cells of households:

- 5 children groups

- 2 spousal groups
- 47 age groups
- 97 income bins

Optimal G4 has a + 1 version where we allocate for a fifth age group of individuals older than 64 years of age. Optimal G47 has a + 35 version where optimal allocation for all age groups are determined.

1.3.2 Allocation Functions

Functions in the [AllocateR/alloc_discrete_fun_R](#) folder of the project repository page is responsible for feeding the dynamic programming results into the allocation functions. The functions in this folder call the [ffp_snw_process_inputs](#) function to solve the allocation problems and compute REV, and call the [ffp_snw_graph_feasible](#) function to generate allocation graphs. These two functions are a part of the [PrjOptiAlloc](#) package.

Chapter 2

Parameters

2.1 Model Parameters

This is the example vignette for function: `snw_mp_param` from the [PrjOptiSNW Package](#). This function sets and gets different parameters.

2.1.1 Parameters Used for Test Simulation

Rather than solving for all ages between 18 to 100, this solves for age groups, and has limited shocks and asset levels. Used for testing.

```
mp_params = snw_mp_param('default_small', true, 100, 6);
```

```
-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_preftechpricegov Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      i      idx      value
      --      ---      -----
Bequests          1        1        0
a0                2        2      0.258
a1                3        3      0.768
a2                4        4     1.5286
a2_covidyr        5        5       NaN
a2_covidyr_manna_heaven 6        6     1.5286
a2_covidyr_tax_fully_pay 7        7    12.718
bequests_option   8        8        1
beta              9        9    0.86389
cons_allocation_rule 10      10        2
g_cons            11      11    0.17576
g_n               12      12    0.05101
gamma             13      13        2
jret              14      14        13
r                 15      15    0.21665
theta             16      16    0.56523
throw_in_ocean    17      17        1
```

```
-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_intlen Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      i      idx      value
```

```

      -  ---  -----
n_agrid      1    1    25
n_educgrid   2    2    2
n_eta_H_grid 3    3    5
n_eta_S_grid 4    4    1
n_etagrid    5    5    5
n_jgrid      6    6    18
n_kidsgrid   7    7    3
n_marriedgrid 8    8    2

-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_covid_unemploy ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      i    idx   ndim  numel   rowN   colN   sum   mean   std   coefv
      -    ---   ----  -----  ----  ----  -----  -----  -----  -----
inc_grid     1    3    2    201    201    1    578.5  2.8781  1.8836  0.654
pi_unemp    2    6    2    240     48    5    9.5319  0.039716 0.019674  0.495

xxx TABLE:inc_grid xxxxxxxxxxxxxxxxx
      c1
      -----
r1          0
r2      0.026667
r3      0.053333
r4          0.08
r5      0.10667
r6      0.13333
r7          0.16
r8      0.18667
r9      0.21333
r10         0.24
r11         0.26667
r12         0.29333
r13         0.32
r14         0.34667
r15         0.37333
r16         0.4
r17         0.42667
r18         0.45333
r19         0.48
r20         0.50667
r21         0.53333
r22         0.56
r23         0.58667
r24         0.61333
r25         0.64
r26         0.66667
r27         0.69333
r28         0.72
r29         0.74667
r30         0.77333
r31         0.8
r32         0.82667
r33         0.85333

```

r34	0.88
r35	0.90667
r36	0.93333
r37	0.96
r38	0.98667
r39	1.0133
r40	1.04
r41	1.0667
r42	1.0933
r43	1.12
r44	1.1467
r45	1.1733
r46	1.2
r47	1.2267
r48	1.2533
r49	1.28
r50	1.3067
r152	4.06
r153	4.12
r154	4.18
r155	4.24
r156	4.3
r157	4.36
r158	4.42
r159	4.48
r160	4.54
r161	4.6
r162	4.66
r163	4.72
r164	4.78
r165	4.84
r166	4.9
r167	4.96
r168	5.02
r169	5.08
r170	5.14
r171	5.2
r172	5.26
r173	5.32
r174	5.38
r175	5.44
r176	5.5
r177	5.56
r178	5.62
r179	5.68
r180	5.74
r181	5.8
r182	5.86
r183	5.92
r184	5.98
r185	6.04
r186	6.1
r187	6.16
r188	6.22
r189	6.28
r190	6.34
r191	6.4
r192	6.46

r193	6.52
r194	6.58
r195	6.64
r196	6.7
r197	6.76
r198	6.82
r199	6.88
r200	6.94
r201	7

xxx TABLE:pi_unemp xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5
r1	0.080278	0.051706	0.041502	0.03538	0.025176
r2	0.080278	0.051706	0.041502	0.03538	0.025176
r3	0.080278	0.051706	0.041502	0.03538	0.025176
r4	0.080278	0.051706	0.041502	0.03538	0.025176
r5	0.080278	0.051706	0.041502	0.03538	0.025176
r6	0.080278	0.051706	0.041502	0.03538	0.025176
r7	0.080278	0.051706	0.041502	0.03538	0.025176
r8	0.080278	0.051706	0.041502	0.03538	0.025176
r9	0.080278	0.051706	0.041502	0.03538	0.025176
r10	0.080278	0.051706	0.041502	0.03538	0.025176
r11	0.080278	0.051706	0.041502	0.03538	0.025176
r12	0.080278	0.051706	0.041502	0.03538	0.025176
r13	0.080278	0.051706	0.041502	0.03538	0.025176
r14	0.070703	0.042132	0.031928	0.025805	0.015601
r15	0.070703	0.042132	0.031928	0.025805	0.015601
r16	0.070703	0.042132	0.031928	0.025805	0.015601
r17	0.070703	0.042132	0.031928	0.025805	0.015601
r18	0.070703	0.042132	0.031928	0.025805	0.015601
r19	0.070703	0.042132	0.031928	0.025805	0.015601
r20	0.070703	0.042132	0.031928	0.025805	0.015601
r21	0.070703	0.042132	0.031928	0.025805	0.015601
r22	0.070703	0.042132	0.031928	0.025805	0.015601
r23	0.070703	0.042132	0.031928	0.025805	0.015601
r24	0.067512	0.038941	0.028736	0.022614	0.01241
r25	0.067512	0.038941	0.028736	0.022614	0.01241
r26	0.067512	0.038941	0.028736	0.022614	0.01241
r27	0.067512	0.038941	0.028736	0.022614	0.01241
r28	0.067512	0.038941	0.028736	0.022614	0.01241
r29	0.067512	0.038941	0.028736	0.022614	0.01241
r30	0.067512	0.038941	0.028736	0.022614	0.01241
r31	0.067512	0.038941	0.028736	0.022614	0.01241
r32	0.067512	0.038941	0.028736	0.022614	0.01241
r33	0.067512	0.038941	0.028736	0.022614	0.01241
r34	0.068576	0.040004	0.0298	0.023678	0.013474
r35	0.068576	0.040004	0.0298	0.023678	0.013474
r36	0.068576	0.040004	0.0298	0.023678	0.013474
r37	0.068576	0.040004	0.0298	0.023678	0.013474
r38	0.068576	0.040004	0.0298	0.023678	0.013474
r39	0.068576	0.040004	0.0298	0.023678	0.013474
r40	0.068576	0.040004	0.0298	0.023678	0.013474
r41	0.068576	0.040004	0.0298	0.023678	0.013474
r42	0.068576	0.040004	0.0298	0.023678	0.013474
r43	0.068576	0.040004	0.0298	0.023678	0.013474
r44	0.080278	0.051706	0.041502	0.03538	0.025176

```

r45 0.080278 0.051706 0.041502 0.03538 0.025176
r46 0.080278 0.051706 0.041502 0.03538 0.025176
r47 0.080278 0.051706 0.041502 0.03538 0.025176
r48 0.080278 0.051706 0.041502 0.03538 0.025176

-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_covid_unemploy Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

      i     idx    value
      -     ---   -----
TR          1     1    0.0017225
b           2     2     1
n_incgrid   3     4    201
n_welfchecksgrid 4     5     45
scaleconvertor 5     7   58056
xi          6     8     0.75

-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_statesgrid ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

      i     idx   ndim  numel   rowN   colN    sum   mean   std
      -     ---   ----  -----  -----  -----  -----  -----  -----
agrid        1     1     2     25     25     1    878.91  35.156  41.372
eta_H_grid   2     2     2      5      5     1 -2.2204e-16 -4.4409e-17 1.4543
eta_S_grid   3     3     2      5      5     1       0       0       0

xxx TABLE:agrid xxxxxxxxxxxxxxxxx
c1
-----

r1      0
r2  0.0097656
r3  0.078125
r4  0.26367
r5  0.625
r6  1.2207
r7  2.1094
r8  3.3496
r9  5
r10 7.1191
r11 9.7656
r12 12.998
r13 16.875
r14 21.455
r15 26.797
r16 32.959
r17 40
r18 47.979
r19 56.953
r20 66.982
r21 78.125
r22 90.439
r23 103.98
r24 118.82

```

```
r25          135

xxx TABLE:eta_H_grid xxxxxxxxxxxxxxxxxxxx
      c1
      -----
r1    -1.8395
r2    -0.91976
r3      0
r4    0.91976
r5    1.8395

xxx TABLE:eta_S_grid xxxxxxxxxxxxxxxxxxxx
      c1
      --
r1    0
r2    0
r3    0
r4    0
r5    0

-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_exotrans ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

      i   idx  ndim  numel  rowN  colN  sum  mean  std
      -   ---  ----  -----  ----  ----  ----  -----  -----
cl_mt_pi_jem_kidseta  1     2     2      1     1     1     0     0
pi_H_eta               2     3     2     25     5     5     5     0.2   0.3851
pi_eta                 3     5     2     25     5     5     5     0.2   0.3851
pi_kids                4     6     5     648    3    216   216   0.33333 0.3561
psi                     5     7     2     18     18     1  14.251  0.79171 0.3125

xxx TABLE:cl_mt_pi_jem_kidseta xxxxxxxxxxxxxxxxxxxx
      c1
      --
r1    0

xxx TABLE:pi_H_eta xxxxxxxxxxxxxxxxxxxx
      c1        c2        c3        c4        c5
      -----  -----  -----  -----  -----
r1    0.925    0.075001  4.8068e-10  0         0
r2    0.0026569  0.96788   0.029459  2.602e-11  0
r3    1.1558e-12  0.0096913  0.98062   0.0096913  1.1559e-12
r4    1.28e-29   2.602e-11  0.029459  0.96788   0.0026569
r5    2.8504e-54  1.8802e-27  4.8068e-10  0.075001  0.925

xxx TABLE:pi_eta xxxxxxxxxxxxxxxxxxxx
      c1        c2        c3        c4        c5
      -----  -----  -----  -----  -----
r1    0.925    0.075001  4.8068e-10  0         0
r2    0.0026569  0.96788   0.029459  2.602e-11  0
r3    1.1558e-12  0.0096913  0.98062   0.0096913  1.1559e-12
```

r4	1.28e-29	2.602e-11	0.029459	0.96788	0.0026569
r5	2.8504e-54	1.8802e-27	4.8068e-10	0.075001	0.925

xxx TABLE:pi_kids xxxxxxxxxxxxxxxxxxxxxxxx

c1	c2	c3	c214	c215	c216
r1	0.88584	0.11137	0.0027905	1	0
r2	0.051343	0.66234	0.28632	1	0
r3	0.0015025	0.063309	0.93519	1	0

xxx TABLE:psi xxxxxxxxxxxxxxxxxxxxxxxx

c1	
r1	0.99935
r2	0.99623
r3	0.99635
r4	0.99537
r5	0.99299
r6	0.98956
r7	0.98547
r8	0.98022
r9	0.96914
r10	0.95071
r11	0.92082
r12	0.87772
r13	0.81394
r14	0.70638
r15	0.54032
r16	0.34767
r17	0.18848
r18	0

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

CONTAINER NAME: mp_params_exotrans Scalars

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

i	idx	value
-	---	-----
bl_store_shock_trans	1	1
pi_S_eta	2	4

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

CONTAINER NAME: mp_params_typelife ND Array (Matrix etc)

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

i	idx	ndim	numel	rowN	colN	sum	mean	std	coefvar
-	---	---	-----	----	----	-----	-----	-----	-----
SS	1	1	2	36	18	2	3.2218	0.089493	0.12913
epsilon	2	2	2	36	18	2	39.526	1.0979	0.85451

xxx TABLE:SS xxxxxxxxxxxxxxxx

c1	c2
-	-----

r1	0	0
r2	0	0
r3	0	0
r4	0	0
r5	0	0
r6	0	0
r7	0	0
r8	0	0
r9	0	0
r10	0	0
r11	0	0
r12	0	0
r13	0.24433	0.29263
r14	0.24433	0.29263
r15	0.24433	0.29263
r16	0.24433	0.29263
r17	0.24433	0.29263
r18	0.24433	0.29263

xxx TABLE:epsilon xxxxxxxxxxxxxxxxxxxxxxxx

	c1	c2
	-----	-----
r1	1	1
r2	1.0778	1.1836
r3	1.2546	1.6124
r4	1.397	1.9418
r5	1.5022	2.1452
r6	1.5712	2.2394
r7	1.6064	2.2588
r8	1.6097	2.2341
r9	1.5815	2.182
r10	1.5204	2.1034
r11	1.4243	1.9846
r12	1.2917	1.8041
r13	0	0
r14	0	0
r15	0	0
r16	0	0
r17	0	0
r18	0	0

xxx

CONTAINER NAME: mp_params_stat ND Array (Matrix etc)

xxx

	i	idx	ndim	numel	rowN	colN	sum	mean	std
	-	---	----	-----	----	----	-----	-----	-----
Pop	1	1	2	18	18	1	9.8945	0.54969	0.31889
stat_distr_educ	2	3	2	2	1	2	1	0.5	0.2786
stat_distr_eta	3	4	2	5	1	5	1	0.2	0.24003
stat_distr_kids	4	5	3	12	2	6	4	0.33333	0.33166
stat_distr_married	5	6	2	4	2	2	2	0.5	0.073381

xxx TABLE:Pop xxxxxxxxxxxxxxxxxxxxxxxx

c1

```

r1          1
r2      0.95085
r3      0.90129
r4      0.85442
r5      0.80919
r6      0.76452
r7      0.71982
r8      0.67493
r9      0.62947
r10     0.58044
r11     0.52505
r12     0.46001
r13     0.38416
r14     0.29751
r15     0.19995
r16     0.1028
r17     0.034004
r18     0.006098

xxx TABLE:stat_distr_educ xxxxxxxxxxxxxxxxxxxxxxxx
      c1      c2
      -----  -----
r1    0.697    0.303

xxx TABLE:stat_distr_eta xxxxxxxxxxxxxxxxxxxxxxxx
      c1      c2      c3      c4      c5
      -----  -----  -----  -----  -----
r1    0.0069316   0.19567   0.59479   0.19567   0.0069316

xxx TABLE:stat_distr_kids xxxxxxxxxxxxxxxxxxxxxxxx
      c1      c2      c3      c4      c5      c6
      -----  -----  -----  -----  -----  -----
r1    0.75801    0.44877    0.1564    0.32041    0.08559    0.23083
r2    0.97627    0.7604    0.023626    0.2173    0.00010011   0.022305

xxx TABLE:stat_distr_married xxxxxxxxxxxxxxxxxxxxxxxx
      c1      c2
      -----  -----
r1    0.5635    0.4365
r2    0.4364    0.5636

-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_stat String
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      i      idx                      string
      ---  ---  -----
st_old_age_depend    "1"    "2"    "Old-age dependency ratio (ratio of 65+/(18-64))=0.1155"

```

2.1.2 Documentation Run Parameters Docdense

Parameters used for documentation vign. "docdense" uses less shocks than the version of the model used to implement the allocation problems in the [Nygaard, Sorensen and Wang \(2020\)](#).

```
mp_params = snw_mp_param('default_docdense', true, 100, 6);
```

```
-----
```

```
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_preftechpricegov Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	value
	--	--	-----
Bequests	1	1	0
a0	2	2	0.258
a1	3	3	0.768
a2	4	4	1.5286
a2_covidyr	5	5	NaN
a2_covidyr_manna_heaven	6	6	1.5286
a2_covidyr_tax_fully_pay	7	7	12.718
bequests_option	8	8	1
beta	9	9	0.97116
cons_allocation_rule	10	10	2
g_cons	11	11	0.17576
g_n	12	12	0.01
gamma	13	13	2
jret	14	14	48
r	15	15	0.04
theta	16	16	0.56523
throw_in_ocean	17	17	1

```
-----
```

```
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_intlen Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	value
	-	--	-----
n_agrid	1	1	65
n_educgrid	2	2	2
n_eta_H_grid	3	3	81
n_eta_S_grid	4	4	5
n_etagrid	5	5	405
n_jgrid	6	6	83
n_kidsgrid	7	7	5
n_marriedgrid	8	8	2

```
-----
```

```
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_covid_unemploy ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	ndim	numel	rowN	colN	sum	mean	std	coefv
	-	--	----	-----	----	----	-----	-----	-----	-----
inc_grid	1	3	2	201	201	1	578.5	2.8781	1.8836	0.654
pi_unemp	2	6	2	415	83	5	9.5319	0.022968	0.024679	1.07

```
xxx TABLE:inc_grid xxxxxxxxxxxxxxxxx
```

c1	
r1	0
r2	0.026667
r3	0.053333
r4	0.08
r5	0.10667
r6	0.13333
r7	0.16
r8	0.18667
r9	0.21333
r10	0.24
r11	0.26667
r12	0.29333
r13	0.32
r14	0.34667
r15	0.37333
r16	0.4
r17	0.42667
r18	0.45333
r19	0.48
r20	0.50667
r21	0.53333
r22	0.56
r23	0.58667
r24	0.61333
r25	0.64
r26	0.66667
r27	0.69333
r28	0.72
r29	0.74667
r30	0.77333
r31	0.8
r32	0.82667
r33	0.85333
r34	0.88
r35	0.90667
r36	0.93333
r37	0.96
r38	0.98667
r39	1.0133
r40	1.04
r41	1.0667
r42	1.0933
r43	1.12
r44	1.1467
r45	1.1733
r46	1.2
r47	1.2267
r48	1.2533
r49	1.28
r50	1.3067
r152	4.06
r153	4.12
r154	4.18
r155	4.24
r156	4.3

r157	4.36
r158	4.42
r159	4.48
r160	4.54
r161	4.6
r162	4.66
r163	4.72
r164	4.78
r165	4.84
r166	4.9
r167	4.96
r168	5.02
r169	5.08
r170	5.14
r171	5.2
r172	5.26
r173	5.32
r174	5.38
r175	5.44
r176	5.5
r177	5.56
r178	5.62
r179	5.68
r180	5.74
r181	5.8
r182	5.86
r183	5.92
r184	5.98
r185	6.04
r186	6.1
r187	6.16
r188	6.22
r189	6.28
r190	6.34
r191	6.4
r192	6.46
r193	6.52
r194	6.58
r195	6.64
r196	6.7
r197	6.76
r198	6.82
r199	6.88
r200	6.94
r201	7

xxx TABLE:pi_unemp xxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5
	-----	-----	-----	-----	-----
r1	0.080278	0.051706	0.041502	0.03538	0.025176
r2	0.080278	0.051706	0.041502	0.03538	0.025176
r3	0.080278	0.051706	0.041502	0.03538	0.025176
r4	0.080278	0.051706	0.041502	0.03538	0.025176
r5	0.080278	0.051706	0.041502	0.03538	0.025176
r6	0.080278	0.051706	0.041502	0.03538	0.025176
r7	0.080278	0.051706	0.041502	0.03538	0.025176
r8	0.080278	0.051706	0.041502	0.03538	0.025176

r9	0.080278	0.051706	0.041502	0.03538	0.025176
r10	0.080278	0.051706	0.041502	0.03538	0.025176
r11	0.080278	0.051706	0.041502	0.03538	0.025176
r12	0.080278	0.051706	0.041502	0.03538	0.025176
r13	0.080278	0.051706	0.041502	0.03538	0.025176
r14	0.070703	0.042132	0.031928	0.025805	0.015601
r15	0.070703	0.042132	0.031928	0.025805	0.015601
r16	0.070703	0.042132	0.031928	0.025805	0.015601
r17	0.070703	0.042132	0.031928	0.025805	0.015601
r18	0.070703	0.042132	0.031928	0.025805	0.015601
r19	0.070703	0.042132	0.031928	0.025805	0.015601
r20	0.070703	0.042132	0.031928	0.025805	0.015601
r21	0.070703	0.042132	0.031928	0.025805	0.015601
r22	0.070703	0.042132	0.031928	0.025805	0.015601
r23	0.070703	0.042132	0.031928	0.025805	0.015601
r24	0.067512	0.038941	0.028736	0.022614	0.01241
r25	0.067512	0.038941	0.028736	0.022614	0.01241
r26	0.067512	0.038941	0.028736	0.022614	0.01241
r27	0.067512	0.038941	0.028736	0.022614	0.01241
r28	0.067512	0.038941	0.028736	0.022614	0.01241
r29	0.067512	0.038941	0.028736	0.022614	0.01241
r30	0.067512	0.038941	0.028736	0.022614	0.01241
r31	0.067512	0.038941	0.028736	0.022614	0.01241
r32	0.067512	0.038941	0.028736	0.022614	0.01241
r33	0.067512	0.038941	0.028736	0.022614	0.01241
r34	0.068576	0.040004	0.0298	0.023678	0.013474
r35	0.068576	0.040004	0.0298	0.023678	0.013474
r36	0.068576	0.040004	0.0298	0.023678	0.013474
r37	0.068576	0.040004	0.0298	0.023678	0.013474
r38	0.068576	0.040004	0.0298	0.023678	0.013474
r39	0.068576	0.040004	0.0298	0.023678	0.013474
r40	0.068576	0.040004	0.0298	0.023678	0.013474
r41	0.068576	0.040004	0.0298	0.023678	0.013474
r42	0.068576	0.040004	0.0298	0.023678	0.013474
r43	0.068576	0.040004	0.0298	0.023678	0.013474
r44	0.080278	0.051706	0.041502	0.03538	0.025176
r45	0.080278	0.051706	0.041502	0.03538	0.025176
r46	0.080278	0.051706	0.041502	0.03538	0.025176
r47	0.080278	0.051706	0.041502	0.03538	0.025176
r48	0.080278	0.051706	0.041502	0.03538	0.025176
r49	0	0	0	0	0
r50	0	0	0	0	0
r51	0	0	0	0	0
r52	0	0	0	0	0
r53	0	0	0	0	0
r54	0	0	0	0	0
r55	0	0	0	0	0
r56	0	0	0	0	0
r57	0	0	0	0	0
r58	0	0	0	0	0
r59	0	0	0	0	0
r60	0	0	0	0	0
r61	0	0	0	0	0
r62	0	0	0	0	0
r63	0	0	0	0	0
r64	0	0	0	0	0
r65	0	0	0	0	0
r66	0	0	0	0	0

r67	0	0	0	0	0
r68	0	0	0	0	0
r69	0	0	0	0	0
r70	0	0	0	0	0
r71	0	0	0	0	0
r72	0	0	0	0	0
r73	0	0	0	0	0
r74	0	0	0	0	0
r75	0	0	0	0	0
r76	0	0	0	0	0
r77	0	0	0	0	0
r78	0	0	0	0	0
r79	0	0	0	0	0
r80	0	0	0	0	0
r81	0	0	0	0	0
r82	0	0	0	0	0
r83	0	0	0	0	0

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_covid_unemploy Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

	i	idx	value
	-	---	-----
TR	1	1	0.0017225
b	2	2	1
n_incgrid	3	4	201
n_welfchecksgrid	4	5	45
scaleconvertor	5	7	58056
xi	6	8	0.75

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_statesgrid ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

	i	idx	ndim	numel	rowN	colN	sum	mean	std
	-	---	----	-----	-----	-----	-----	-----	-----
agrid	1	1	2	65	65	1	2228	34.277	39.432
eta_H_grid	2	2	2	405	405	1	1.3234e-13	3.2676e-16	1.5783
eta_S_grid	3	3	2	405	405	1	-1.7764e-14	-4.3861e-17	2.2103

xxx TABLE:agrid xxxxxxxxxxxxxxxxx
c1

r1	0
r2	0.00051498
r3	0.0041199
r4	0.013905
r5	0.032959
r6	0.064373
r7	0.11124
r8	0.17664
r9	0.26367
r10	0.37542
r11	0.51498

r12	0.68544
r13	0.88989
r14	1.1314
r15	1.4131
r16	1.7381
r17	2.1094
r18	2.5301
r19	3.0034
r20	3.5323
r21	4.1199
r22	4.7693
r23	5.4836
r24	6.2658
r25	7.1191
r26	8.0466
r27	9.0514
r28	10.136
r29	11.305
r30	12.56
r31	13.905
r32	15.342
r33	16.875
r34	18.507
r35	20.241
r36	22.08
r37	24.027
r38	26.085
r39	28.258
r40	30.548
r41	32.959
r42	35.493
r43	38.154
r44	40.945
r45	43.868
r46	46.928
r47	50.126
r48	53.467
r49	56.953
r50	60.587
r51	64.373
r52	68.313
r53	72.411
r54	76.669
r55	81.091
r56	85.68
r57	90.439
r58	95.371
r59	100.48
r60	105.77
r61	111.24
r62	116.89
r63	122.74
r64	128.77
r65	135

xxx TABLE:eta_H_grid xxxxxxxxxxxxxxxxx
c1

r1	-2.6968
r2	-2.6294
r3	-2.562
r4	-2.4945
r5	-2.4271
r6	-2.3597
r7	-2.2923
r8	-2.2249
r9	-2.1574
r10	-2.09
r11	-2.0226
r12	-1.9552
r13	-1.8878
r14	-1.8203
r15	-1.7529
r16	-1.6855
r17	-1.6181
r18	-1.5507
r19	-1.4832
r20	-1.4158
r21	-1.3484
r22	-1.281
r23	-1.2136
r24	-1.1461
r25	-1.0787
r26	-1.0113
r27	-0.94388
r28	-0.87646
r29	-0.80904
r30	-0.74162
r31	-0.6742
r32	-0.60678
r33	-0.53936
r34	-0.47194
r35	-0.40452
r36	-0.3371
r37	-0.26968
r38	-0.20226
r39	-0.13484
r40	-0.06742
r41	2.2204e-16
r42	0.06742
r43	0.13484
r44	0.20226
r45	0.26968
r46	0.3371
r47	0.40452
r48	0.47194
r49	0.53936
r50	0.60678
r356	-0.60678
r357	-0.53936
r358	-0.47194
r359	-0.40452
r360	-0.3371
r361	-0.26968
r362	-0.20226

r363	-0.13484
r364	-0.06742
r365	2.2204e-16
r366	0.06742
r367	0.13484
r368	0.20226
r369	0.26968
r370	0.3371
r371	0.40452
r372	0.47194
r373	0.53936
r374	0.60678
r375	0.6742
r376	0.74162
r377	0.80904
r378	0.87646
r379	0.94388
r380	1.0113
r381	1.0787
r382	1.1461
r383	1.2136
r384	1.281
r385	1.3484
r386	1.4158
r387	1.4832
r388	1.5507
r389	1.6181
r390	1.6855
r391	1.7529
r392	1.8203
r393	1.8878
r394	1.9552
r395	2.0226
r396	2.09
r397	2.1574
r398	2.2249
r399	2.2923
r400	2.3597
r401	2.4271
r402	2.4945
r403	2.562
r404	2.6294
r405	2.6968

xxx TABLE:eta_S_grid xxxxxxxxxxxxxxxxxxxx
c1

r1	-3.122
r2	-3.122
r3	-3.122
r4	-3.122
r5	-3.122
r6	-3.122
r7	-3.122
r8	-3.122
r9	-3.122
r10	-3.122

r11	-3.122
r12	-3.122
r13	-3.122
r14	-3.122
r15	-3.122
r16	-3.122
r17	-3.122
r18	-3.122
r19	-3.122
r20	-3.122
r21	-3.122
r22	-3.122
r23	-3.122
r24	-3.122
r25	-3.122
r26	-3.122
r27	-3.122
r28	-3.122
r29	-3.122
r30	-3.122
r31	-3.122
r32	-3.122
r33	-3.122
r34	-3.122
r35	-3.122
r36	-3.122
r37	-3.122
r38	-3.122
r39	-3.122
r40	-3.122
r41	-3.122
r42	-3.122
r43	-3.122
r44	-3.122
r45	-3.122
r46	-3.122
r47	-3.122
r48	-3.122
r49	-3.122
r50	-3.122
r356	3.122
r357	3.122
r358	3.122
r359	3.122
r360	3.122
r361	3.122
r362	3.122
r363	3.122
r364	3.122
r365	3.122
r366	3.122
r367	3.122
r368	3.122
r369	3.122
r370	3.122
r371	3.122
r372	3.122
r373	3.122

```
r374      3.122
r375      3.122
r376      3.122
r377      3.122
r378      3.122
r379      3.122
r380      3.122
r381      3.122
r382      3.122
r383      3.122
r384      3.122
r385      3.122
r386      3.122
r387      3.122
r388      3.122
r389      3.122
r390      3.122
r391      3.122
r392      3.122
r393      3.122
r394      3.122
r395      3.122
r396      3.122
r397      3.122
r398      3.122
r399      3.122
r400      3.122
r401      3.122
r402      3.122
r403      3.122
r404      3.122
r405      3.122
```

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

CONTAINER NAME: mp_params_exotrans ND Array (Matrix etc)

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

	i	idx	ndim	numel	rowN	colN	sum	mean
	-	---	----	-----	----	----	----	-----
cl_mt_pi_jem_kidseta	1	2	2	1	1	1	0	0
pi_H_eta	2	3	2	6561	81	81	81	0.012346
pi_S_eta	3	4	2	25	5	5	5	0.2
pi_eta	4	5	2	1.6403e+05	405	405	405	0.0024691
pi_kids	5	6	5	8300	5	1660	1660	0.2
psi	6	7	2	83	83	1	78.16	0.94169

xxx TABLE:cl_mt_pi_jem_kidseta xxxxxxxxxxxxxxxxx

c1

--

r1 0

xxx TABLE:pi_H_eta xxxxxxxxxxxxxxxxx

c1	c2	c3	c79	c80	c81
-----	-----	-----	-----	-----	-----
r1	0.44008	0.19741	0.16603	0	0
					0

r2	0.26004	0.18401	0.1972	0	0	0
r3	0.12804	0.13527	0.18471	0	0	0
r4	0.051745	0.078413	0.13644	0	0	0
r5	0.016976	0.035843	0.079479	0	0	0
r6	0.0044863	0.012918	0.036507	0	0	0
r7	0.00094957	0.0036704	0.013221	0	0	0
r8	0.00016032	0.00082204	0.0037748	0	0	0
r9	2.1522e-05	0.0001451	0.00084955	0	0	0
r10	2.2921e-06	2.0182e-05	0.00015069	0	0	0
r11	1.933e-07	2.2115e-06	2.1061e-05	0	0	0
r12	1.2891e-08	1.9089e-07	2.3192e-06	0	0	0
r13	6.7901e-10	1.2976e-08	2.0116e-07	0	0	0
r14	2.8225e-11	6.9453e-10	1.3741e-08	0	0	0
r15	9.2521e-13	2.9264e-11	7.3906e-10	0	0	0
r16	2.3901e-14	9.7051e-13	3.1293e-11	0	0	0
r17	4.8636e-16	2.5328e-14	1.0429e-12	0	0	0
r18	7.7924e-18	5.2007e-16	2.735e-14	0	0	0
r19	9.8265e-20	8.4004e-18	5.6434e-16	0	0	0
r20	9.7502e-22	1.0672e-19	9.1603e-18	0	0	0
r21	7.6101e-24	1.0662e-21	1.1695e-19	0	0	0
r22	4.6713e-26	8.3759e-24	1.1741e-21	0	0	0
r23	2.2546e-28	5.1729e-26	9.269e-24	0	0	0
r24	8.5548e-31	2.5114e-28	5.7527e-26	0	0	0
r25	2.5514e-33	9.583e-31	2.8066e-28	0	0	0
r26	5.9805e-36	2.8738e-33	1.0762e-30	0	0	0
r27	1.1016e-38	6.7725e-36	3.2434e-33	0	0	0
r28	1.5943e-41	1.2541e-38	7.6811e-36	0	0	0
r29	1.8129e-44	1.8245e-41	1.4293e-38	0	0	0
r30	1.6194e-47	2.0853e-44	2.0897e-41	0	0	0
r31	1.1364e-50	1.8723e-47	2.4002e-44	0	0	0
r32	6.2635e-54	1.3205e-50	2.1657e-47	0	0	0
r33	2.7115e-57	7.3149e-54	1.535e-50	0	0	0
r34	9.2192e-61	3.1826e-57	8.5451e-54	0	0	0
r35	2.4617e-64	1.0875e-60	3.7362e-57	0	0	0
r36	5.1617e-68	2.9183e-64	1.283e-60	0	0	0
r37	8.4992e-72	6.1497e-68	3.4599e-64	0	0	0
r38	1.0989e-75	1.0176e-71	7.327e-68	0	0	0
r39	1.1156e-79	1.3223e-75	1.2185e-71	0	0	0
r40	8.8927e-84	1.3491e-79	1.5911e-75	0	0	0
r41	5.5655e-88	1.0807e-83	1.6313e-79	0	0	0
r42	2.7347e-92	6.7971e-88	1.3133e-83	0	0	0
r43	1.055e-96	3.3564e-92	8.3007e-88	0	0	0
r44	3.1951e-101	1.3012e-96	4.1192e-92	0	0	0
r45	7.5967e-106	3.9605e-101	1.6049e-96	0	0	0
r46	1.418e-110	9.4631e-106	4.9088e-101	0	0	0
r47	2.0777e-115	1.7751e-110	1.1787e-105	0	0	0
r48	2.3898e-120	2.6138e-115	2.2219e-110	0	0	0
r49	2.1579e-125	3.0215e-120	3.2881e-115	0	0	0
r50	1.5294e-130	2.7417e-125	3.8196e-120	0	0	0
r51	8.5093e-136	1.9529e-130	3.4831e-125	0	0	0
r52	3.7162e-141	1.0919e-135	2.4933e-130	0	0	0
r53	1.2739e-146	4.7921e-141	1.401e-135	0	0	0
r54	3.4277e-152	1.6509e-146	6.179e-141	0	0	0
r55	7.2393e-158	4.4641e-152	2.1392e-146	0	0	0
r56	1.2001e-163	9.4748e-158	5.8132e-152	0	0	0
r57	1.5615e-169	1.5784e-163	1.2399e-157	0	0	0
r58	1.5947e-175	2.064e-169	2.0759e-163	0	0	0
r59	1.2782e-181	2.1183e-175	2.7279e-169	0	0	0

r60	8.0416e-188	1.7064e-181	2.8135e-175	0	0	0
r61	3.9708e-194	1.0788e-187	2.2776e-181	0	0	0
r62	1.5389e-200	5.3534e-194	1.4472e-187	0	0	0
r63	4.6807e-207	2.085e-200	7.2168e-194	5.5511e-16	0	0
r64	1.1174e-213	6.3733e-207	2.8246e-200	2.7311e-14	5.5511e-16	0
r65	2.0936e-220	1.529e-213	8.677e-207	1.0428e-12	2.5424e-14	4.4409e-16
r66	3.0785e-227	2.8789e-220	2.092e-213	3.1293e-11	9.7056e-13	2.387e-14
r67	3.5527e-234	4.2543e-227	3.9585e-220	7.3906e-10	2.9264e-11	9.2526e-13
r68	3.2178e-241	4.934e-234	5.8786e-227	1.3741e-08	6.9453e-10	2.8225e-11
r69	2.2873e-248	4.491e-241	6.8517e-234	2.0116e-07	1.2976e-08	6.7901e-10
r70	1.276e-255	3.2082e-248	6.2674e-241	2.3192e-06	1.9089e-07	1.2891e-08
r71	5.5866e-263	1.7986e-255	4.4993e-248	2.1061e-05	2.2115e-06	1.933e-07
r72	1.9196e-270	7.9137e-263	2.535e-255	0.00015069	2.0182e-05	2.2921e-06
r73	5.1762e-278	2.7326e-270	1.1209e-262	0.00084955	0.0001451	2.1522e-05
r74	1.0954e-285	7.4052e-278	3.8897e-270	0.0037748	0.00082204	0.00016032
r75	1.8193e-293	1.5749e-285	1.0593e-277	0.013221	0.0036704	0.00094957
r76	2.3712e-301	2.6286e-293	2.264e-285	0.036507	0.012918	0.0044863
r77	2.4254e-309	3.443e-301	3.7975e-293	0.079479	0.035843	0.016976
r78	1.9469e-317	3.5392e-309	4.9987e-301	0.13644	0.078413	0.051745
r79	0	2.8551e-317	5.1639e-309	0.18471	0.13527	0.12804
r80	0	0	4.1864e-317	0.1972	0.18401	0.26004
r81	0	0	0	0.16603	0.19741	0.44008

xxx TABLE:pi_S_eta xxxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5
r1	0.012224	0.2144	0.54675	0.2144	0.012224
r2	0.012224	0.2144	0.54675	0.2144	0.012224
r3	0.012224	0.2144	0.54675	0.2144	0.012224
r4	0.012224	0.2144	0.54675	0.2144	0.012224
r5	0.012224	0.2144	0.54675	0.2144	0.012224

xxx TABLE:pi_eta xxxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c403	c404	c405
r1	0.0053798	0.0024132	0.0020297	0	0	0
r2	0.0031788	0.0022495	0.0024107	0	0	0
r3	0.0015653	0.0016536	0.002258	0	0	0
r4	0.00063256	0.00095856	0.0016679	0	0	0
r5	0.00020753	0.00043816	0.00097159	0	0	0
r6	5.4842e-05	0.00015792	0.00044628	0	0	0
r7	1.1608e-05	4.4868e-05	0.00016162	0	0	0
r8	1.9598e-06	1.0049e-05	4.6145e-05	0	0	0
r9	2.6309e-07	1.7738e-06	1.0385e-05	0	0	0
r10	2.8019e-08	2.4671e-07	1.8421e-06	0	0	0
r11	2.363e-09	2.7035e-08	2.5746e-07	0	0	0
r12	1.5758e-10	2.3335e-09	2.835e-08	0	0	0
r13	8.3005e-12	1.5863e-10	2.459e-09	0	0	0
r14	3.4504e-13	8.4903e-12	1.6797e-10	0	0	0
r15	1.131e-14	3.5774e-13	9.0346e-12	0	0	0
r16	2.9218e-16	1.1864e-14	3.8254e-13	0	0	0
r17	5.9455e-18	3.0962e-16	1.2749e-14	0	0	0
r18	9.5258e-20	6.3575e-18	3.3434e-16	0	0	0
r19	1.2012e-21	1.0269e-19	6.8987e-18	0	0	0
r20	1.1919e-23	1.3046e-21	1.1198e-19	0	0	0
r21	9.303e-26	1.3034e-23	1.4296e-21	0	0	0

r22	5.7105e-28	1.0239e-25	1.4353e-23	0	0	0
r23	2.7561e-30	6.3237e-28	1.1331e-25	0	0	0
r24	1.0458e-32	3.07e-30	7.0324e-28	0	0	0
r25	3.119e-35	1.1715e-32	3.4309e-30	0	0	0
r26	7.3108e-38	3.5131e-35	1.3156e-32	0	0	0
r27	1.3466e-40	8.279e-38	3.9649e-35	0	0	0
r28	1.949e-43	1.533e-40	9.3897e-38	0	0	0
r29	2.2162e-46	2.2303e-43	1.7473e-40	0	0	0
r30	1.9797e-49	2.5492e-46	2.5546e-43	0	0	0
r31	1.3892e-52	2.2888e-49	2.9342e-46	0	0	0
r32	7.6568e-56	1.6142e-52	2.6475e-49	0	0	0
r33	3.3147e-59	8.9421e-56	1.8764e-52	0	0	0
r34	1.127e-62	3.8906e-59	1.0446e-55	0	0	0
r35	3.0092e-66	1.3294e-62	4.5673e-59	0	0	0
r36	6.3099e-70	3.5674e-66	1.5684e-62	0	0	0
r37	1.039e-73	7.5177e-70	4.2295e-66	0	0	0
r38	1.3433e-77	1.244e-73	8.9569e-70	0	0	0
r39	1.3638e-81	1.6164e-77	1.4895e-73	0	0	0
r40	1.0871e-85	1.6492e-81	1.945e-77	0	0	0
r41	6.8035e-90	1.3211e-85	1.9942e-81	0	0	0
r42	3.343e-94	8.3091e-90	1.6054e-85	0	0	0
r43	1.2896e-98	4.1031e-94	1.0147e-89	0	0	0
r44	3.9058e-103	1.5907e-98	5.0355e-94	0	0	0
r45	9.2866e-108	4.8414e-103	1.9619e-98	0	0	0
r46	1.7334e-112	1.1568e-107	6.0007e-103	0	0	0
r47	2.5399e-117	2.1699e-112	1.4409e-107	0	0	0
r48	2.9214e-122	3.1953e-117	2.7162e-112	0	0	0
r49	2.6379e-127	3.6936e-122	4.0195e-117	0	0	0
r50	1.8697e-132	3.3516e-127	4.6693e-122	0	0	0
r356	7.6568e-56	1.6142e-52	2.6475e-49	0	0	0
r357	3.3147e-59	8.9421e-56	1.8764e-52	0	0	0
r358	1.127e-62	3.8906e-59	1.0446e-55	0	0	0
r359	3.0092e-66	1.3294e-62	4.5673e-59	0	0	0
r360	6.3099e-70	3.5674e-66	1.5684e-62	0	0	0
r361	1.039e-73	7.5177e-70	4.2295e-66	0	0	0
r362	1.3433e-77	1.244e-73	8.9569e-70	0	0	0
r363	1.3638e-81	1.6164e-77	1.4895e-73	0	0	0
r364	1.0871e-85	1.6492e-81	1.945e-77	0	0	0
r365	6.8035e-90	1.3211e-85	1.9942e-81	0	0	0
r366	3.343e-94	8.3091e-90	1.6054e-85	0	0	0
r367	1.2896e-98	4.1031e-94	1.0147e-89	0	0	0
r368	3.9058e-103	1.5907e-98	5.0355e-94	0	0	0
r369	9.2866e-108	4.8414e-103	1.9619e-98	0	0	0
r370	1.7334e-112	1.1568e-107	6.0007e-103	0	0	0
r371	2.5399e-117	2.1699e-112	1.4409e-107	0	0	0
r372	2.9214e-122	3.1953e-117	2.7162e-112	0	0	0
r373	2.6379e-127	3.6936e-122	4.0195e-117	0	0	0
r374	1.8697e-132	3.3516e-127	4.6693e-122	0	0	0
r375	1.0402e-137	2.3873e-132	4.2579e-127	0	0	0
r376	4.5428e-143	1.3348e-137	3.0479e-132	0	0	0
r377	1.5573e-148	5.8581e-143	1.7126e-137	0	0	0
r378	4.1902e-154	2.0181e-148	7.5536e-143	0	0	0
r379	8.8497e-160	5.4571e-154	2.6151e-148	0	0	0
r380	1.467e-165	1.1582e-159	7.1063e-154	0	0	0
r381	1.9088e-171	1.9296e-165	1.5158e-159	0	0	0
r382	1.9494e-177	2.5231e-171	2.5377e-165	0	0	0
r383	1.5626e-183	2.5895e-177	3.3347e-171	0	0	0
r384	9.8305e-190	2.0859e-183	3.4394e-177	0	0	0

r385	4.8541e-196	1.3188e-189	2.7843e-183	0	0	0
r386	1.8812e-202	6.5443e-196	1.7691e-189	0	0	0
r387	5.7219e-209	2.5488e-202	8.8221e-196	6.7859e-18	0	0
r388	1.366e-215	7.791e-209	3.453e-202	3.3387e-16	6.7859e-18	0
r389	2.5593e-222	1.8691e-215	1.0607e-208	1.2748e-14	3.108e-16	5.4288e-18
r390	3.7633e-229	3.5193e-222	2.5573e-215	3.8254e-13	1.1865e-14	2.918e-16
r391	4.343e-236	5.2006e-229	4.839e-222	9.0346e-12	3.5774e-13	1.1311e-14
r392	3.9336e-243	6.0316e-236	7.1863e-229	1.6797e-10	8.4903e-12	3.4504e-13
r393	2.7961e-250	5.49e-243	8.3758e-236	2.459e-09	1.5863e-10	8.3005e-12
r394	1.5599e-257	3.9218e-250	7.6616e-243	2.835e-08	2.3335e-09	1.5758e-10
r395	6.8293e-265	2.1987e-257	5.5002e-250	2.5746e-07	2.7035e-08	2.363e-09
r396	2.3466e-272	9.674e-265	3.0989e-257	1.8421e-06	2.4671e-07	2.8019e-08
r397	6.3276e-280	3.3405e-272	1.3702e-264	1.0385e-05	1.7738e-06	2.6309e-07
r398	1.3391e-287	9.0525e-280	4.7549e-272	4.6145e-05	1.0049e-05	1.9598e-06
r399	2.224e-295	1.9252e-287	1.2949e-279	0.00016162	4.4868e-05	1.1608e-05
r400	2.8987e-303	3.2133e-295	2.7676e-287	0.00044628	0.00015792	5.4842e-05
r401	2.9649e-311	4.2089e-303	4.6422e-295	0.00097159	0.00043816	0.00020753
r402	2.38e-319	4.3265e-311	6.1107e-303	0.0016679	0.00095856	0.00063256
r403	0	3.4902e-319	6.3126e-311	0.002258	0.0016536	0.0015653
r404	0	0	5.1176e-319	0.0024107	0.0022495	0.0031788
r405	0	0	0	0.0020297	0.0024132	0.0053798

xxx TABLE:pi_kids xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c1658	c1659	c1660
r1	0.88581	0.11137	0.0027904	0	0	0
r2	0.051118	0.65943	0.28506	0	0	0
r3	0.0012655	0.05332	0.78763	0	0	0
r4	1.808e-05	0.00080512	0.069952	0	0	0
r5	1.8185e-07	8.1047e-06	0.00075722	0	0	0

xxx TABLE:psi xxxxxxxxxxxxxxxxxxxxxxx

	c1
r1	0.99937
r2	0.99933
r3	0.99929
r4	0.99925
r5	0.99923
r6	0.99923
r7	0.99924
r8	0.99927
r9	0.99928
r10	0.99929
r11	0.99927
r12	0.99924
r13	0.9992
r14	0.99915
r15	0.99909
r16	0.99901
r17	0.99892
r18	0.99882
r19	0.99872
r20	0.9986
r21	0.99848
r22	0.99834

r23	0.9982
r24	0.99806
r25	0.99791
r26	0.99775
r27	0.9976
r28	0.99742
r29	0.99725
r30	0.99706
r31	0.99691
r32	0.99674
r33	0.99657
r34	0.99635
r35	0.99607
r36	0.99574
r37	0.99533
r38	0.99487
r39	0.99436
r40	0.9938
r41	0.99319
r42	0.99253
r43	0.9918
r44	0.99097
r45	0.99005
r46	0.98902
r47	0.98787
r48	0.98659
r49	0.98519
r50	0.98372
r51	0.98217
r52	0.98052
r53	0.97863
r54	0.97651
r55	0.97431
r56	0.9721
r57	0.96973
r58	0.96693
r59	0.96365
r60	0.96005
r61	0.95611
r62	0.95165
r63	0.9465
r64	0.94063
r65	0.93385
r66	0.9261
r67	0.9174
r68	0.90767
r69	0.89701
r70	0.88535
r71	0.87267
r72	0.85894
r73	0.84407
r74	0.82817
r75	0.81104
r76	0.79282
r77	0.77349
r78	0.75398
r79	0.7345
r80	0.71587

```

r81      0.69772
r82      0.68139
r83      0

-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_exotrans Scalars
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
          i     idx    value
          -     ---   -----
bl_store_shock_trans   1      1      0

-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_params_typelife ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
          i     idx    ndim   numel   rowN   colN    sum    mean    std   coefvari
          -     ---   ----   -----   ----   ----   -----   -----   -----   -----
SS        1      1      2      166     83      2     19.331  0.11645  0.13441  1.1542
epsilon   2      2      2      166     83      2     159.11   0.95847  0.89457  0.93333

xxx TABLE:SS xxxxxxxxxxxxxxxxxxxxxxxx
          c1       c2
          -----  -----
r1        0       0
r2        0       0
r3        0       0
r4        0       0
r5        0       0
r6        0       0
r7        0       0
r8        0       0
r9        0       0
r10       0       0
r11       0       0
r12       0       0
r13       0       0
r14       0       0
r15       0       0
r16       0       0
r17       0       0
r18       0       0
r19       0       0
r20       0       0
r21       0       0
r22       0       0
r23       0       0
r24       0       0
r25       0       0
r26       0       0
r27       0       0
r28       0       0
r29       0       0
r30       0       0
r31       0       0

```

```
r32      0      0
r33      0      0
r34      0      0
r35      0      0
r36      0      0
r37      0      0
r38      0      0
r39      0      0
r40      0      0
r41      0      0
r42      0      0
r43      0      0
r44      0      0
r45      0      0
r46      0      0
r47      0      0
r48    0.24433  0.29263
r49    0.24433  0.29263
r50    0.24433  0.29263
r51    0.24433  0.29263
r52    0.24433  0.29263
r53    0.24433  0.29263
r54    0.24433  0.29263
r55    0.24433  0.29263
r56    0.24433  0.29263
r57    0.24433  0.29263
r58    0.24433  0.29263
r59    0.24433  0.29263
r60    0.24433  0.29263
r61    0.24433  0.29263
r62    0.24433  0.29263
r63    0.24433  0.29263
r64    0.24433  0.29263
r65    0.2443...
```

2.1.3 Parameters Used for Paper Simulations

Full version of parameters used in [Nygaard, Sorensen and Wang \(2020\)](#). This is not printed to save space.

```
% mp_params = snw_mp_param('default_moredense_a65zh266zs5_e2m2', true, 100, 6);
```

2.2 Model Controls

This is the example vignette for function: `snw_mp_control` from the [PrjOptiSNW Package](#). This function sets and gets different control parameters.

2.2.1 Test SNW_MP_CONTROLS Defaults

Call the function with defaults.

```
mp_controls = snw_mp_control('default_base', true);

pos = 35 ; key = options
fmincon options:

Options used by current Algorithm ('interior-point'):
(Other available algorithms: 'active-set', 'sqp', 'sqp-legacy', 'trust-region-reflective')
```

```

Set properties:
    Display: 'off'

Default properties:
    Algorithm: 'interior-point'
    CheckGradients: 0
    ConstraintTolerance: 1.0000e-06
    FiniteDifferenceStepSize: 'sqrt(eps)'
    FiniteDifferenceType: 'forward'
    HessianApproximation: 'bfgs'
        HessianFcn: []
        HessianMultiplyFcn: []
        HonorBounds: 1
    MaxFunctionEvaluations: 3000
        MaxIterations: 1000
        ObjectiveLimit: -1.0000e+20
    OptimalityTolerance: 1.0000e-06
        OutputFcn: []
        PlotFcn: []
        ScaleProblem: 0
    SpecifyConstraintGradient: 0
    SpecifyObjectiveGradient: 0
        StepTolerance: 1.0000e-10
    SubproblemAlgorithm: 'factorization'
        TypicalX: 'ones(numberOfVariables,1)'
    UseParallel: 0

Show options not used by current Algorithm ('interior-point')

pos = 36 ; key = options2
fsolve options:

Options used by current Algorithm ('trust-region-dogleg'):
(Other available algorithms: 'levenberg-marquardt', 'trust-region')

Set properties:
    Display: 'off'

Default properties:
    Algorithm: 'trust-region-dogleg'
    CheckGradients: 0
    FiniteDifferenceStepSize: 'sqrt(eps)'
        FiniteDifferenceType: 'forward'
        FunctionTolerance: 1.0000e-06
    MaxFunctionEvaluations: '100*numberOfVariables'
        MaxIterations: 400
    OptimalityTolerance: 1.0000e-06
        OutputFcn: []
        PlotFcn: []
    SpecifyObjectiveGradient: 0
        StepTolerance: 1.0000e-06
        TypicalX: 'ones(numberOfVariables,1)'
    UseParallel: 0

Show options not used by current Algorithm ('trust-region-dogleg')

-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

```

CONTAINER NAME: mp_controls Scalars
 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

	i	idx	value
	--	---	-----
A_aux	1	1	NaN
Aeq	2	2	NaN
B_aux	3	3	NaN
Beq	4	4	NaN
bl_compute_drv_stats	5	5	1
bl_print_a4chk	6	6	1
bl_print_a4chk_verbose	7	7	0
bl_print_ds	8	8	1
bl_print_ds_verbose	9	9	0
bl_print_evuvw19_jaeemk	10	10	1
bl_print_evuvw19_jaeemk_verbose	11	11	0
bl_print_evuvw19_jmky	12	12	1
bl_print_evuvw19_jmky_allchecks	13	13	1
bl_print_evuvw19_jmky_allchecks_verbose	14	14	0
bl_print_evuvw19_jmky_mass	15	15	1
bl_print_evuvw19_jmky_mass_verbose	16	16	0
bl_print_evuvw19_jmky_verbose	17	17	0
bl_print_evuvw20_jaeemk	18	18	1
bl_print_evuvw20_jaeemk_verbose	19	19	0
bl_print_find_tax_rate	20	20	1
bl_print_find_tax_rate_verbose	21	21	0
bl_print_precompute	22	22	1
bl_print_precompute_verbose	23	23	0
bl_print_v_planner	24	24	1
bl_print_v_planner_verbose	25	25	0
bl_print_vfi	26	26	1
bl_print_vfi_verbose	27	27	0
bl_print_vu_vw	28	28	1
bl_print_vu_vw_verbose	29	29	0
bl_timer	30	30	1
err	31	31	1
fl_max_trchk_perc_increase	32	32	1.5
nonlcon	33	34	NaN
tol	34	37	0.005

 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

CONTAINER NAME: mp_controls String
 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

	i	idx	string
	--	---	-----
mp_params_name	"1"	"33"	"default_base"

Chapter 3

Solving the Dynamic Life Cycle Problem

3.1 Life Cycle Dynamic Programming with Marital Status, Children and Savings

This is the example vignette for function: `snw_vfi_main_bisec_vec` from the [PrjOptiSNW Package](#). This function solves for policy function with vectorized bisection. Value function during COVIDless year.

3.1.1 Test SNW_VFI_MAIN_BISECT_VEC Defaults

Call the function with defaults.

```
mp_param = snw_mp_param('default_docdense');
[V_VFI,ap_VFI,cons_VFI] = snw_vfi_main_bisec_vec(mp_param);

SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:83 of 82, time-this-age:4.8333
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:82 of 82, time-this-age:3.5522
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:81 of 82, time-this-age:3.562
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:80 of 82, time-this-age:3.5482
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:79 of 82, time-this-age:3.603
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:78 of 82, time-this-age:3.566
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:77 of 82, time-this-age:3.587
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:76 of 82, time-this-age:3.5763
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:75 of 82, time-this-age:3.5899
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:74 of 82, time-this-age:3.5638
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:73 of 82, time-this-age:3.5816
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:72 of 82, time-this-age:3.5953
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:71 of 82, time-this-age:3.5487
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:70 of 82, time-this-age:3.5608
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:69 of 82, time-this-age:3.5752
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:68 of 82, time-this-age:3.5654
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:67 of 82, time-this-age:3.5772
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:66 of 82, time-this-age:3.559
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:65 of 82, time-this-age:3.5818
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:64 of 82, time-this-age:3.5676
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:63 of 82, time-this-age:3.5591
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:62 of 82, time-this-age:3.5857
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:61 of 82, time-this-age:3.5505
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:60 of 82, time-this-age:3.599
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:59 of 82, time-this-age:3.5761
```

SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:58 of 82, time-this-age:3.5793
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:57 of 82, time-this-age:3.5776
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:56 of 82, time-this-age:3.543
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:55 of 82, time-this-age:3.6012
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:54 of 82, time-this-age:3.5452
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:53 of 82, time-this-age:3.5613
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:52 of 82, time-this-age:3.5813
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:51 of 82, time-this-age:3.5609
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:50 of 82, time-this-age:3.536
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:49 of 82, time-this-age:3.5886
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:48 of 82, time-this-age:3.566
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:47 of 82, time-this-age:3.7808
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:46 of 82, time-this-age:3.7661
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:45 of 82, time-this-age:3.7932
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:44 of 82, time-this-age:3.8033
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:43 of 82, time-this-age:3.7945
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:42 of 82, time-this-age:3.7732
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:41 of 82, time-this-age:3.9226
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:40 of 82, time-this-age:3.8366
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:39 of 82, time-this-age:3.8564
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:38 of 82, time-this-age:3.8009
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:37 of 82, time-this-age:3.7941
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:36 of 82, time-this-age:4.5804
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:35 of 82, time-this-age:3.8404
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:34 of 82, time-this-age:4.7106
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:33 of 82, time-this-age:3.8021
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:32 of 82, time-this-age:3.8057
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:31 of 82, time-this-age:3.7965
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:30 of 82, time-this-age:3.8032
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:29 of 82, time-this-age:3.7654
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:28 of 82, time-this-age:3.7559
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:27 of 82, time-this-age:3.7947
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:26 of 82, time-this-age:3.7678
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:25 of 82, time-this-age:3.7811
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:24 of 82, time-this-age:3.7676
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:23 of 82, time-this-age:3.7898
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:22 of 82, time-this-age:3.7599
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:21 of 82, time-this-age:3.8708
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:20 of 82, time-this-age:4.153
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:19 of 82, time-this-age:3.8598
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:18 of 82, time-this-age:3.7762
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:17 of 82, time-this-age:3.7936
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:16 of 82, time-this-age:3.8078
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:15 of 82, time-this-age:3.8384
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:14 of 82, time-this-age:3.8247
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:13 of 82, time-this-age:3.8612
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:12 of 82, time-this-age:3.7894
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:11 of 82, time-this-age:3.8057
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:10 of 82, time-this-age:3.7815
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:9 of 82, time-this-age:3.7954
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:8 of 82, time-this-age:3.8034
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:7 of 82, time-this-age:3.82
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:6 of 82, time-this-age:4.665
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:5 of 82, time-this-age:3.8131
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:4 of 82, time-this-age:3.7714
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:3 of 82, time-this-age:3.7764
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:2 of 82, time-this-age:3.8035
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:1 of 82, time-this-age:3.7918

```
Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_base;time=311.
```

3.1.2 Define Parameters

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

```
% Grids:
age_grid = 18:100;
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=%3.2f'));
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});
```

3.1.3 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') = 'eastoutside';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('it_legend_select') = 21; % how many shock legends to show
mp_support_graph('cl_colors') = 'jet';

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

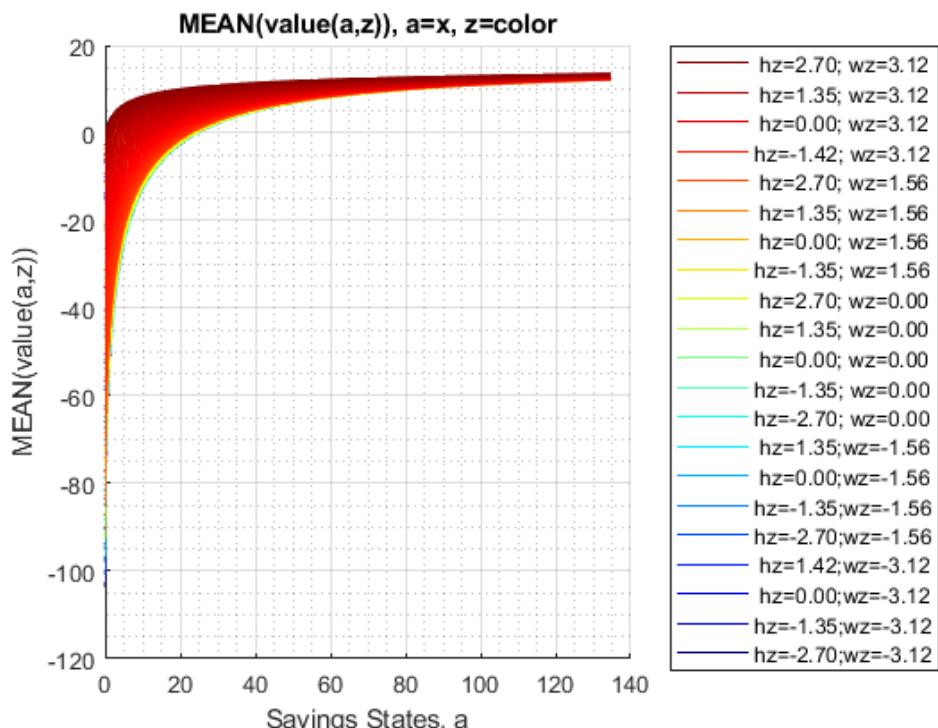
Tabulate value and policies along savings and shocks:

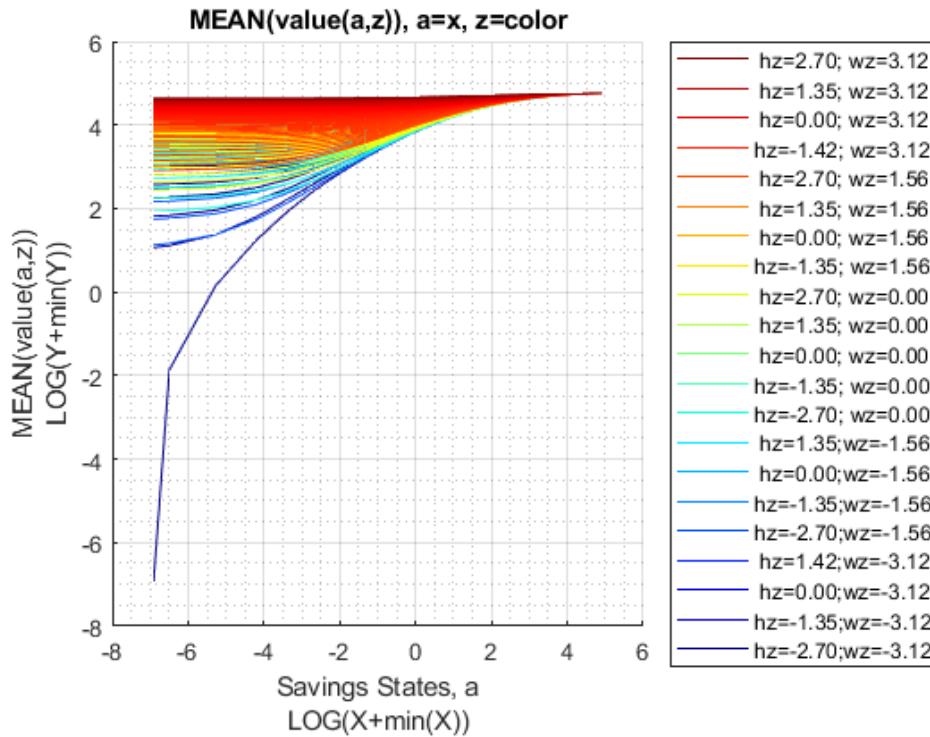
```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [1,4,5,6,3,2];
% Value Function
tb_az_v = ff_summ_nd_array("MEAN(VAL(A,Z))", V_VFI, true, ["mean"], 4, 1, cl_mp_datasetdesc, ar_permute);

xxx MEAN(VAL(A,Z)) xxxxxxxxxxxxxxxxxxxxxxxxx
group      savings      mean_eta_1      mean_eta_2      mean_eta_3      mean_eta_4      mean_eta_5      mean_eta_6
-----  -----
1           0          -103.74       -100.83       -97.586       -94.14        -90.628       -87.112
```

xxx	MEAN(AP(A,Z))	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx							
group	savings	mean_eta_1	mean_eta_2	mean_eta_3	mean_eta_4	mean_eta_5	mean_eta_6	mean_eta_7	mean_eta_8
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----
1	0	0	0	0	0	0	0	0	0

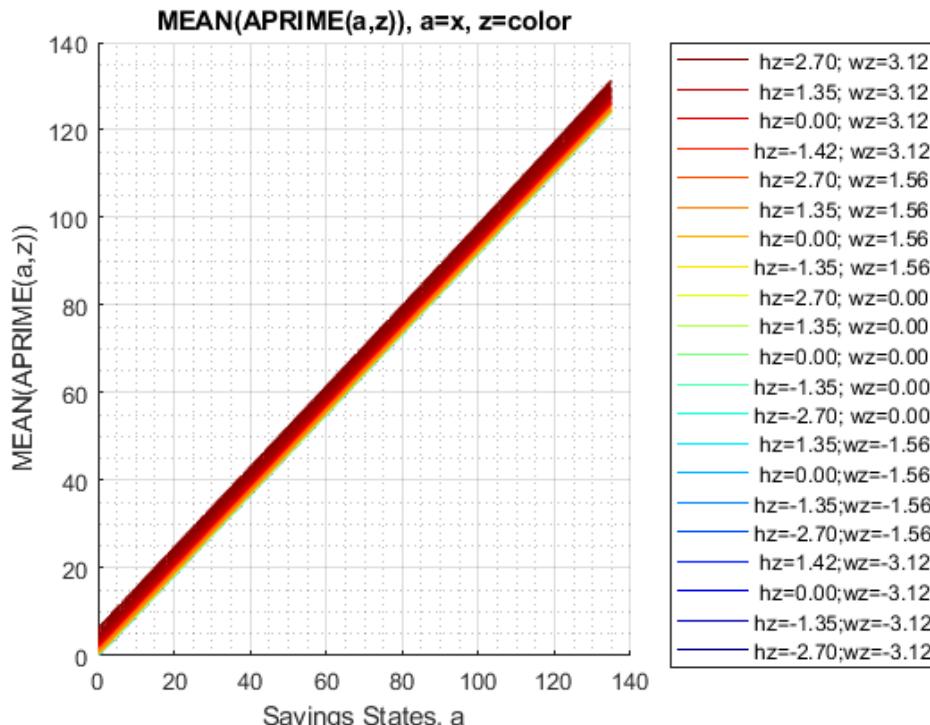
```
mp_support_graph('cl_st_graph_title') = {'MEAN(value(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(a,z))'};
ff_graph_grid((tb_az_v{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);
```

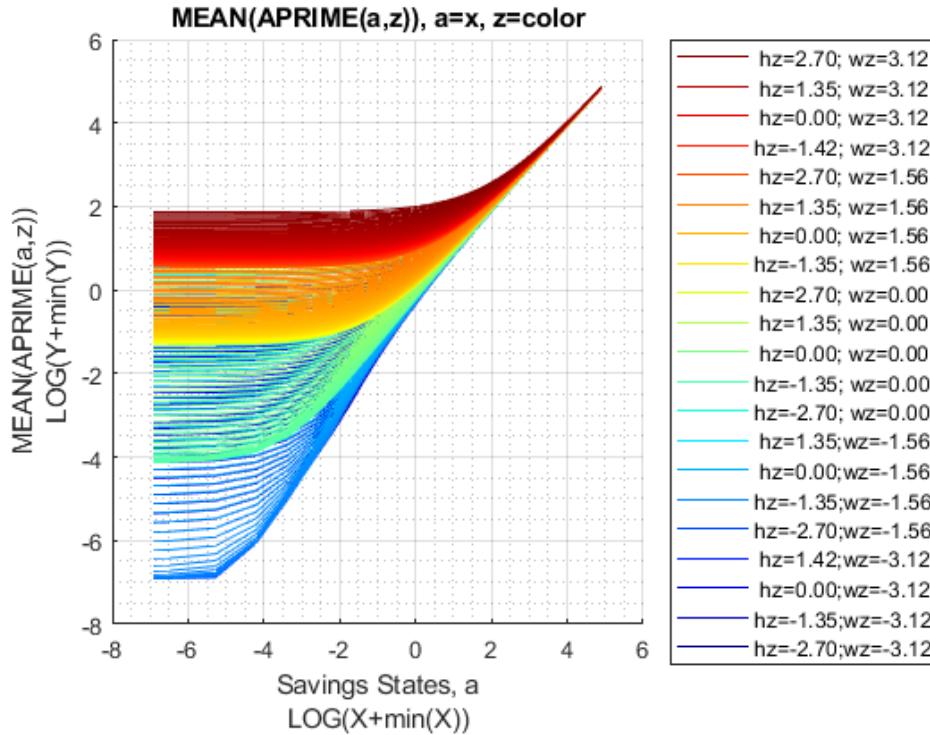




Graph Mean Savings Choices:

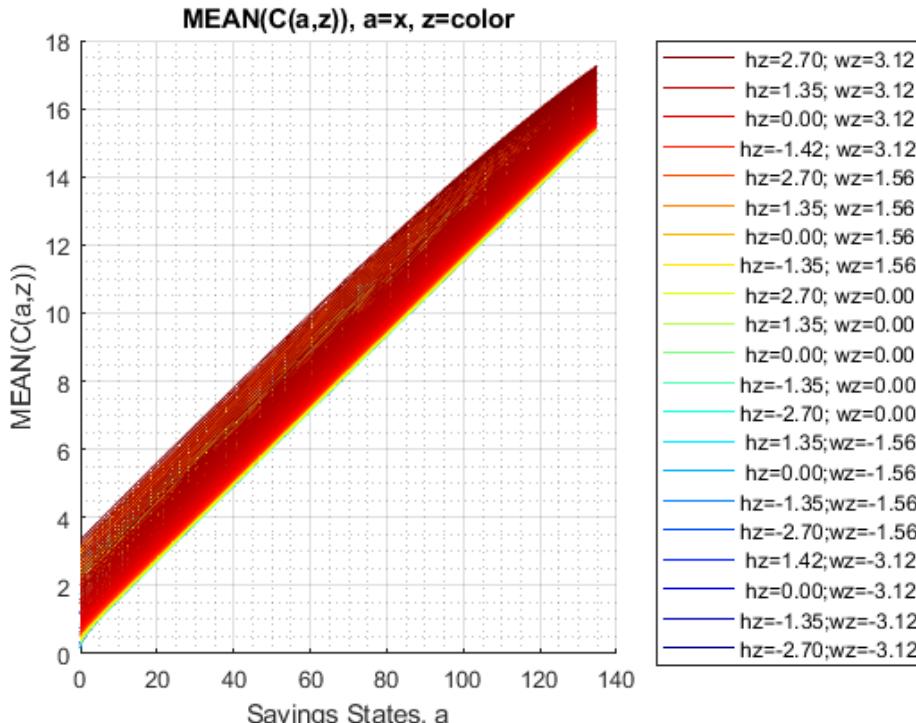
```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(a,z))'};
ff_graph_grid((tb_az_ap{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```

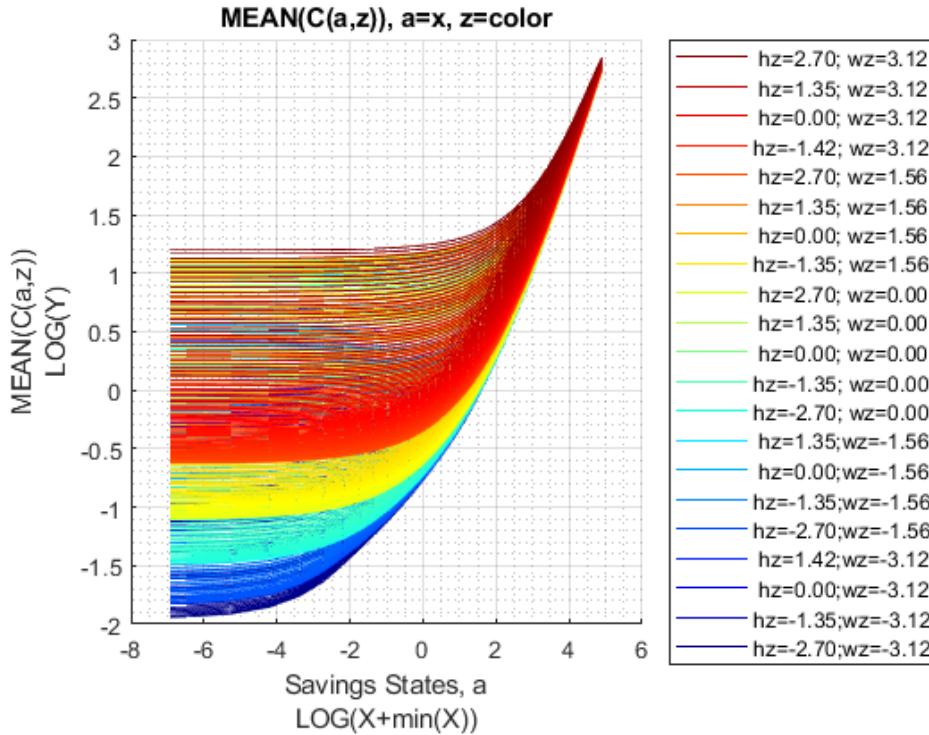




Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(a,z))'};
ff_graph_grid((tb_az_c{1:end, 3:end}'), ar_st_eta_HS_grid,agrid, mp_support_graph);
```





3.1.4 Analyze Kids and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = [...
    "k0M0", "K1M0", "K2M0", "K3M0", "K4M0", ...
    "k0M1", "K1M1", "K2M1", "K3M1", "K4M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {...
    'o', 'd', 's', 'x', '*', ...
    'o', 'd', 's', 'x', '*'};
mp_support_graph('cl_colors') = {...
    'red', 'red', 'red', 'red', 'red',...
    'blue', 'blue', 'blue', 'blue', 'blue'};
```

MEAN(VAL(KM,J)), MEAN(AP(KM,J)), MEAN(C(KM,J))

Tabulate value and policies:

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

xxx MEAN(VAL(KM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx		group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
---	---	---	---	---	---	---	---	---	---
1	1	0	-9.6123	-8.574	-7.5952	-6.6749	-5.8609		
2	2	0	-17.183	-15.851	-14.558	-13.309	-12.171		

3	3	0	-20.909	-19.563	-18.242	-16.949	-15.768
4	4	0	-24.758	-23.406	-22.06	-20.727	-19.5
5	5	0	-27.561	-26.288	-25.009	-23.73	-22.552
6	1	1	2.1559	3.0013	3.7773	4.4944	5.1268
7	2	1	-2.4375	-1.4691	-0.55596	0.31118	1.0968
8	3	1	-4.6483	-3.672	-2.7454	-1.8583	-1.0517
9	4	1	-7.2434	-6.2806	-5.3574	-4.4633	-3.6454
10	5	1	-9.2948	-8.3935	-7.5263	-6.6822	-5.9134

```
% Aprime Choice
```

```
tb_az_ap = ff_summ_nd_array("MEAN(AP(KM,J))", ap_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_pe)
```

xxx MEAN(AP(KM,J))		xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx					
group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	1	0	34.494	34.456	34.416	34.452	34.489
2	2	0	34.3	34.256	34.21	34.238	34.268
3	3	0	34.146	34.101	34.055	34.082	34.11
4	4	0	34.053	34.01	33.964	33.991	34.02
5	5	0	33.97	33.929	33.885	33.915	33.946
6	1	1	35.208	35.246	35.285	35.413	35.545
7	2	1	34.951	34.976	35	35.11	35.222
8	3	1	34.708	34.724	34.739	34.838	34.939
9	4	1	34.506	34.516	34.523	34.613	34.704
10	5	1	34.221	34.218	34.212	34.286	34.363

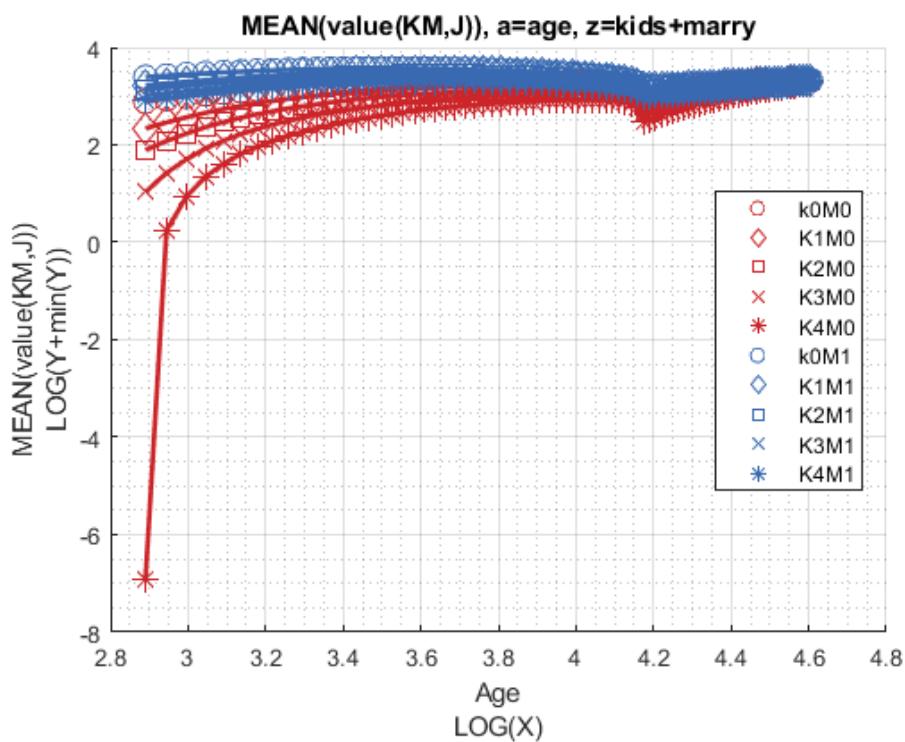
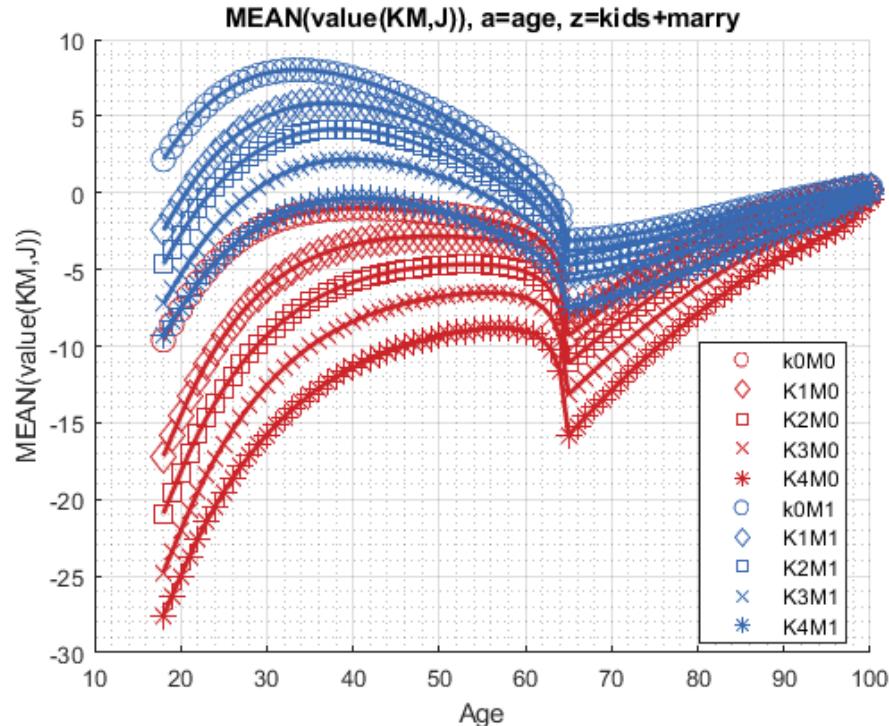
```
% Consumption Choices
```

```
tb_az_c = ff_summ_nd_array("MEAN(C(KM,J))", cons_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_pe)
```

xxx MEAN(C(KM,J))		xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx					
group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	1	0	2.0632	2.102	2.1418	2.184	2.2244
2	2	0	2.2579	2.3019	2.348	2.3975	2.4457
3	3	0	2.4119	2.4563	2.503	2.5537	2.6031
4	4	0	2.5046	2.5481	2.594	2.6445	2.6938
5	5	0	2.5877	2.6287	2.6724	2.7207	2.7678
6	1	1	2.6183	2.6787	2.7402	2.8051	2.8674
7	2	1	2.681	2.7395	2.8002	2.8656	2.9293
8	3	1	2.7896	2.8462	2.9054	2.9698	3.0325
9	4	1	2.8528	2.9056	2.9612	3.0222	3.0816
10	5	1	2.9174	2.966	3.0172	3.0737	3.1281

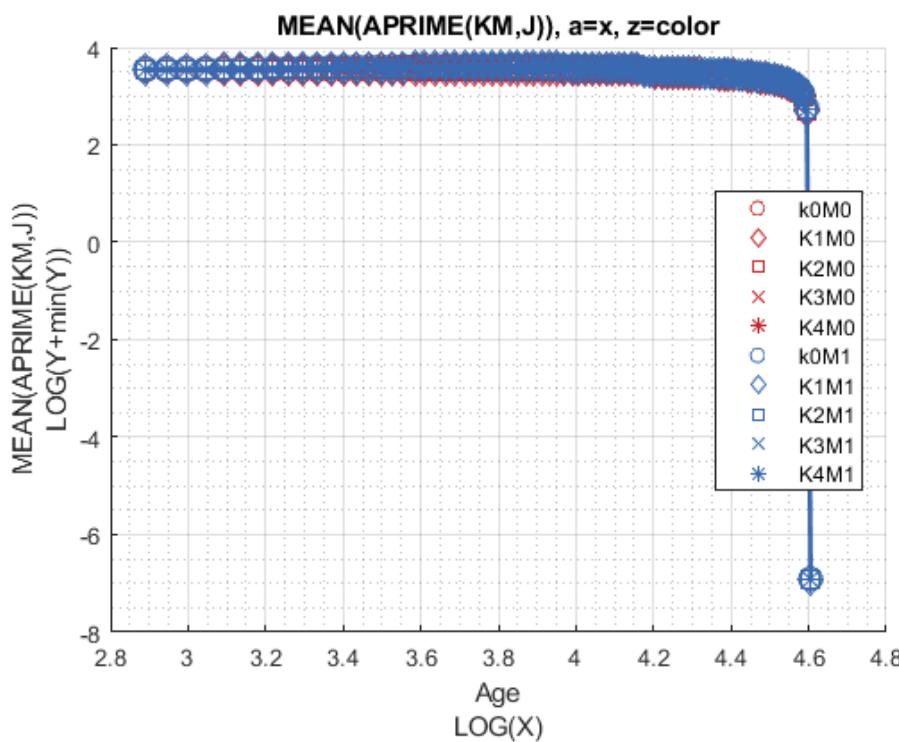
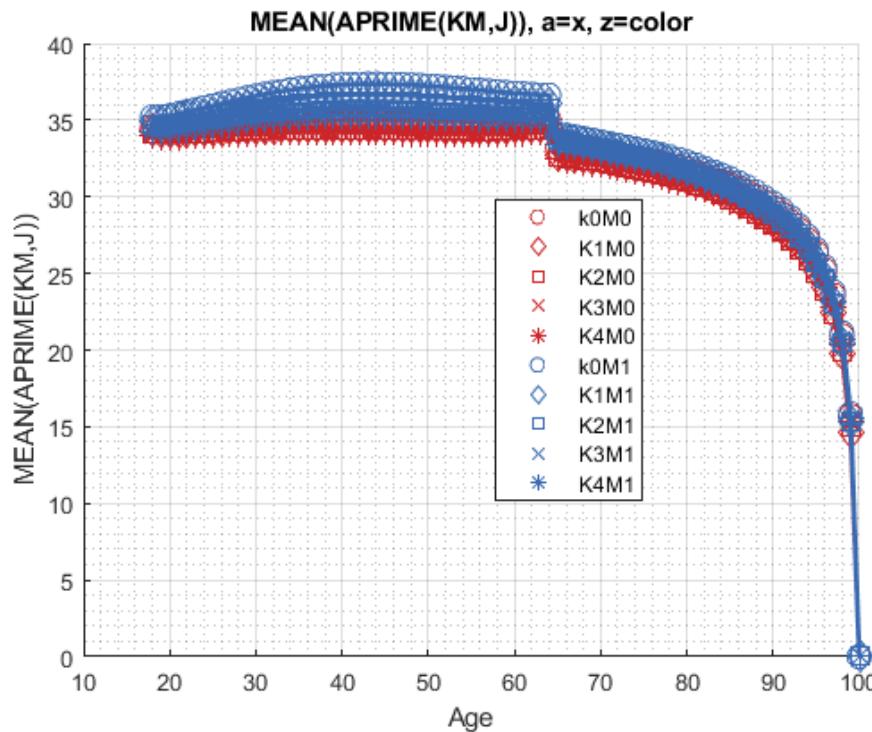
Graph Mean Values:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(value(KM,J)), a=age, z=kids+marry'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(KM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



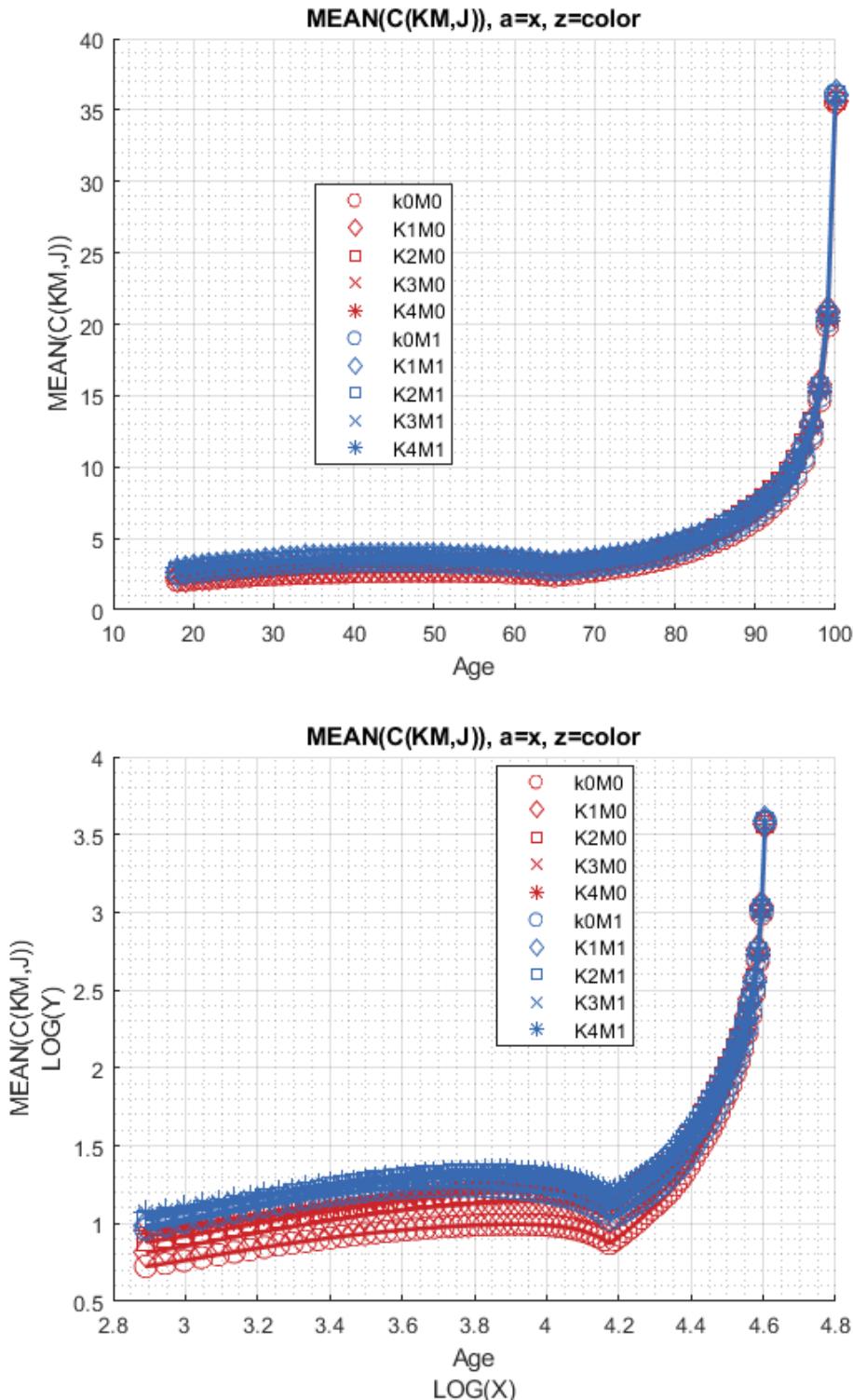
Graph Mean Savings Choices:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(KM,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(KM,J))'};
ff_graph_grid((tb_az_ap{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(KM,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(KM,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



3.1.5 Analyze Education and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["E0M0", "E1M0", "E0M1", "E1M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
```

```
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {'*', 'p', '*', 'p'};
mp_support_graph('cl_colors') = {'red', 'red', 'blue', 'blue'};
```

MEAN(VAL(EKM,J)), MEAN(AP(EKM,J)), MEAN(C(EKM,J))

Tabulate value and policies:

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

xxx MEAN(VAL(EKM,J)) xxxxxxxxxxxxxxxxxxxxxxxx
group edu marry mean_age_18 mean_age_19 mean_age_20 mean_age_21 mean_age_22
---- --- ---- -----
1 0 0 -23.27 -22.094 -20.941 -19.811 -18.761
2 1 0 -16.739 -15.379 -14.045 -12.745 -11.58
3 0 1 -6.6189 -5.6779 -4.7885 -3.9435 -3.1707
4 1 1 -1.9684 -1.0477 -0.17465 0.66417 1.4159

% Aprime Choice
tb_az_ap = ff_summ_nd_array("MEAN(AP(EKM,J))", ap_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_p

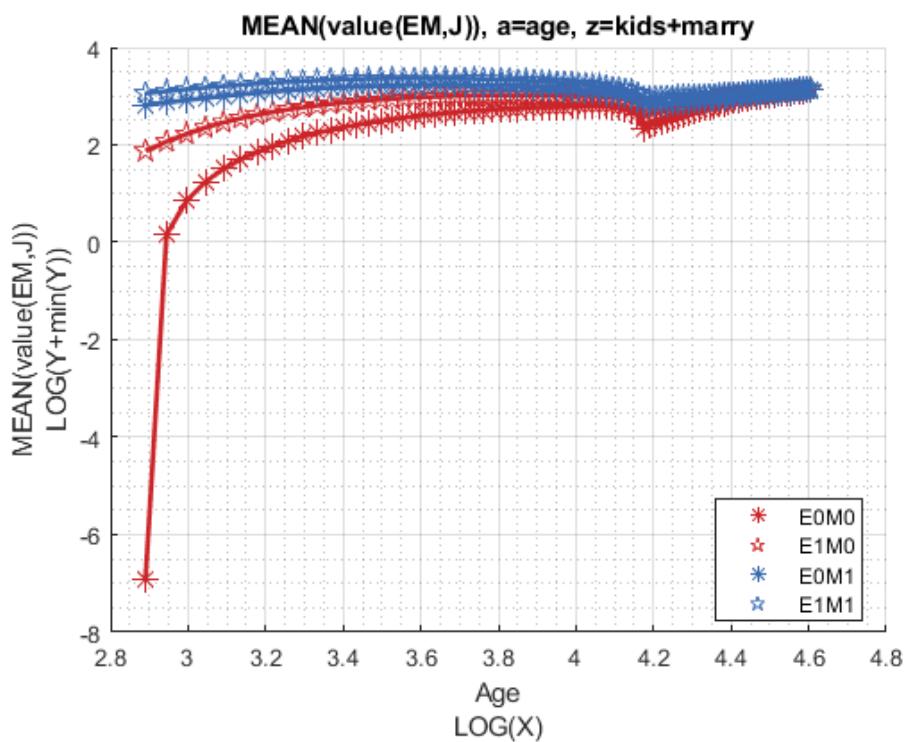
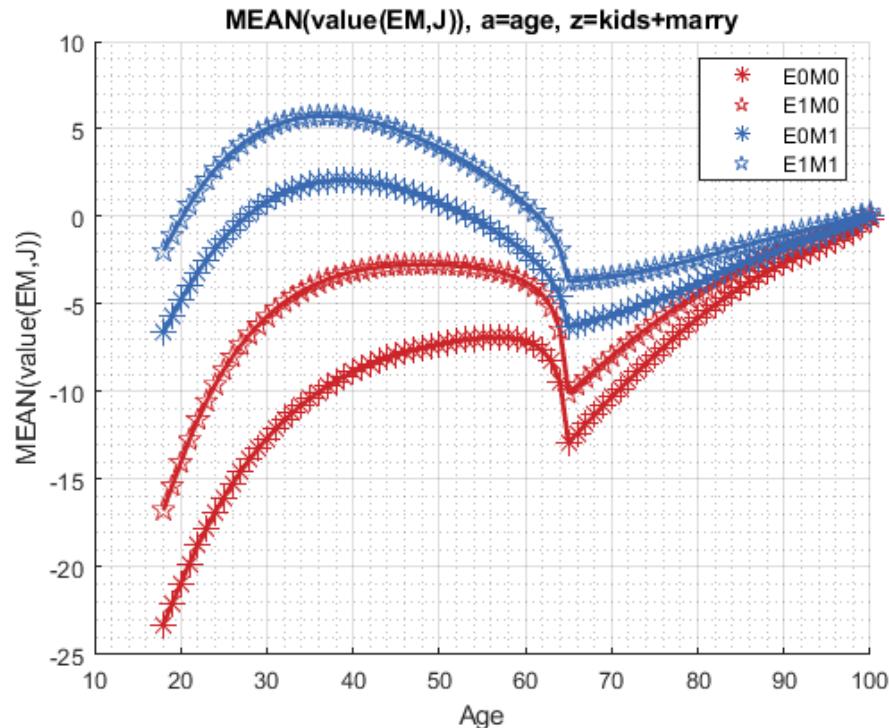
xxx MEAN(AP(EKM,J)) xxxxxxxxxxxxxxxxxxxxxxxx
group edu marry mean_age_18 mean_age_19 mean_age_20 mean_age_21 mean_age_22
---- --- ---- -----
1 0 0 34.294 34.261 34.226 34.237 34.247
2 1 0 34.091 34.04 33.986 34.035 34.087
3 0 1 34.769 34.789 34.809 34.88 34.951
4 1 1 34.669 34.683 34.695 34.824 34.958

% Consumption Choices
tb_az_c = ff_summ_nd_array("MEAN(C(EKM,J))", cons_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_p

xxx MEAN(C(EKM,J)) xxxxxxxxxxxxxxxxxxxxxxxx
group edu marry mean_age_18 mean_age_19 mean_age_20 mean_age_21 mean_age_22
---- --- ---- -----
1 0 0 2.2635 2.2969 2.3317 2.3683 2.4043
2 1 0 2.4666 2.5178 2.572 2.6319 2.6896
3 0 1 2.6261 2.6712 2.7175 2.7661 2.8135
4 1 1 2.9175 2.9832 3.0522 3.1285 3.202
```

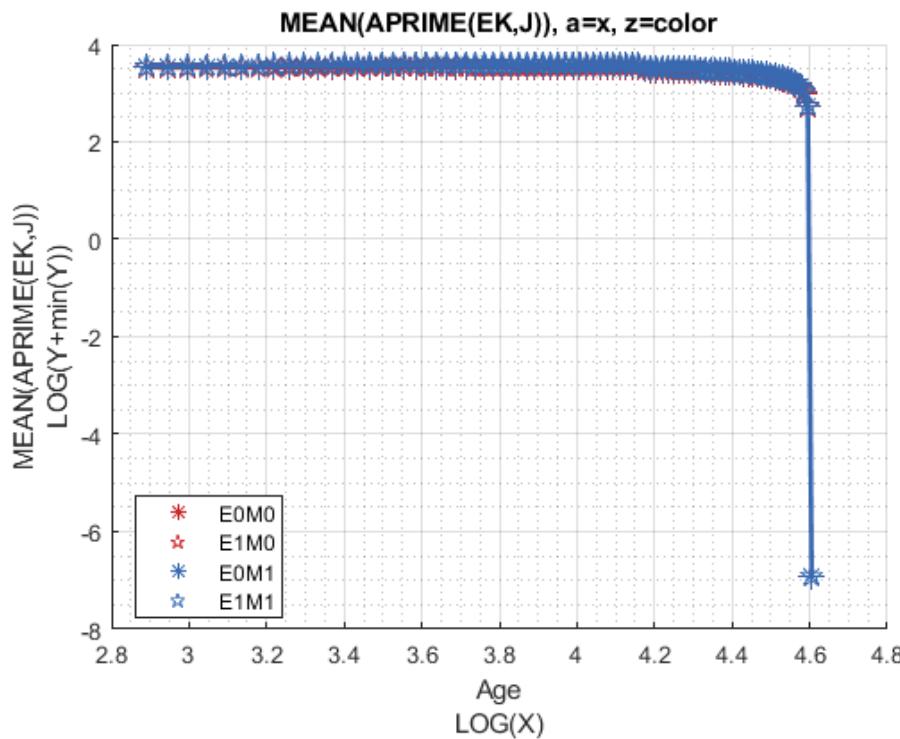
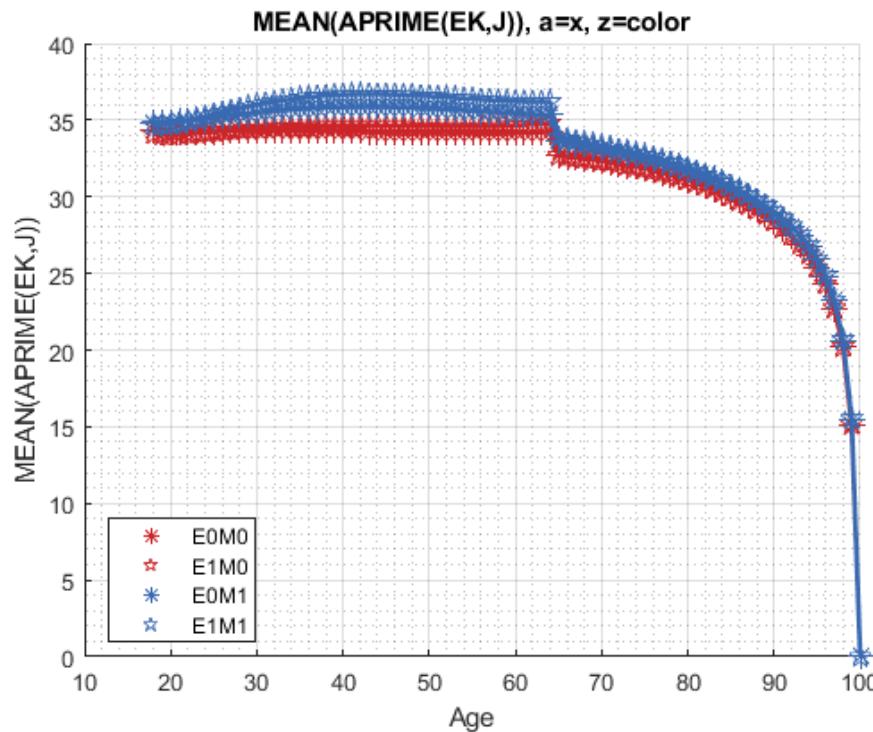
Graph Mean Values:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(value(EM,J)), a=age, z=kids+marry'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(EM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



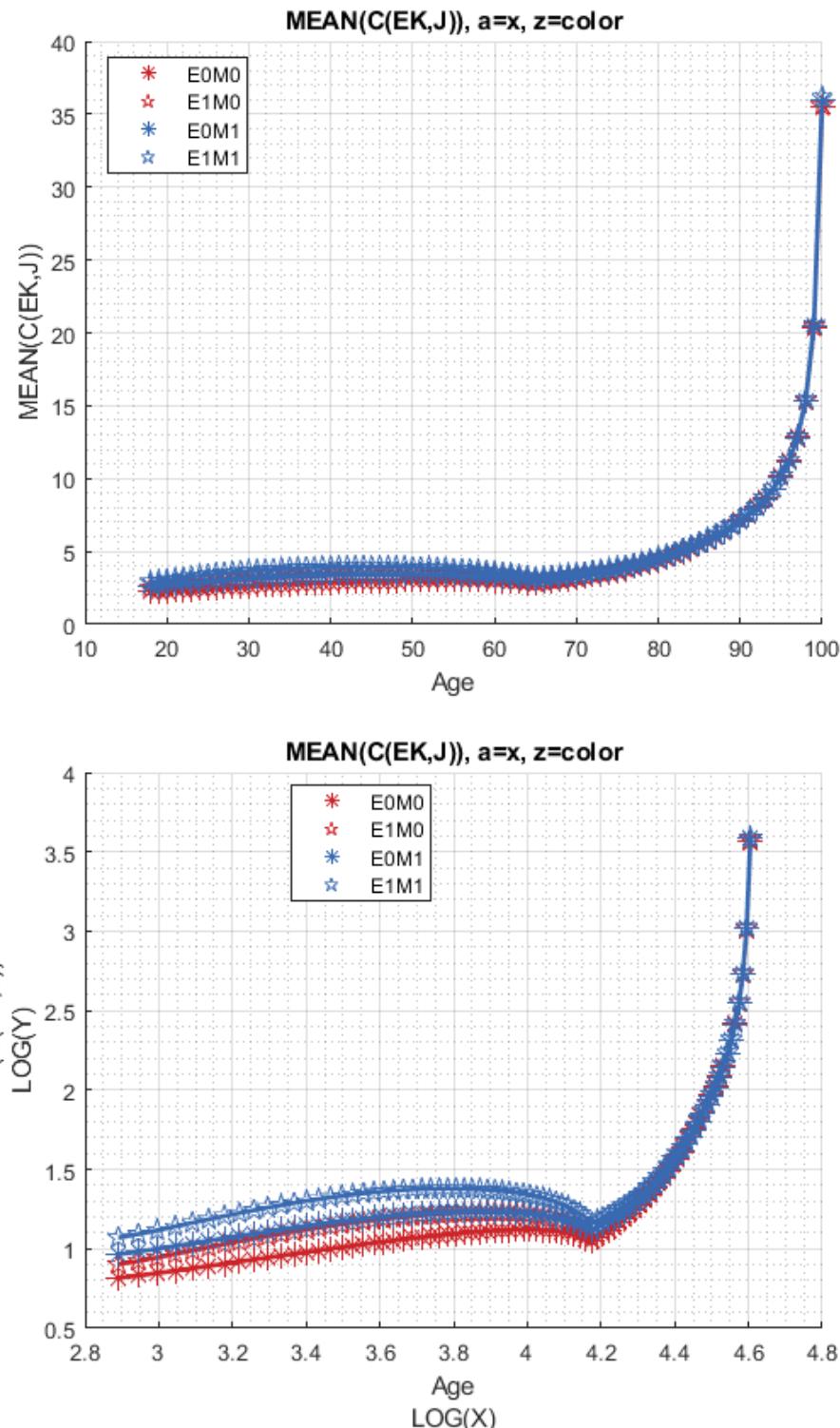
Graph Mean Savings Choices:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(EK,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(EK,J))'};
ff_graph_grid((tb_az_ap{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(EK,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(EK,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Chapter 4

Alternative Value Function Solution Testing

4.1 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.

4.1.1 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
```

4.1.2 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=%3.2f'));
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});
```

4.1.3 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_permute);
```

xxx	MEAN(VAL(A,Z))	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	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, ar_per

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, ar_per

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

4.2 Small Test Grid Search Solution

This is the example vignette for function: `snw_vfi_main_grid_search` from the [PrjOptiSNW Package](#). This function solves for policy function using grid search. Small Solution Analysis. Small Solution Analysis, husband 5 shocks, wife 1 shocks.

4.2.1 Test SNW_VFI_MAIN_GRID_SEARCH 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_grid_search(mp_param);

SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:18 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:17 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:16 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:15 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:14 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:13 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:12 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:11 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:10 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:9 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:8 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:7 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:6 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:5 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:4 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:3 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:2 of 18
SNW_VFI_MAIN_GRID_SEARCH: Finished Age Group:1 of 18
Elapsed time is 5.586732 seconds.
Completed SNW_VFI_MAIN_GRID_SEARCH;SNW_MP_PARAM=default_small;SNW_MP_CONTROL=default_base
```

4.2.2 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=%3.2f;
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'}, {'Hshock', 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});

```

4.2.3 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_permute);

```

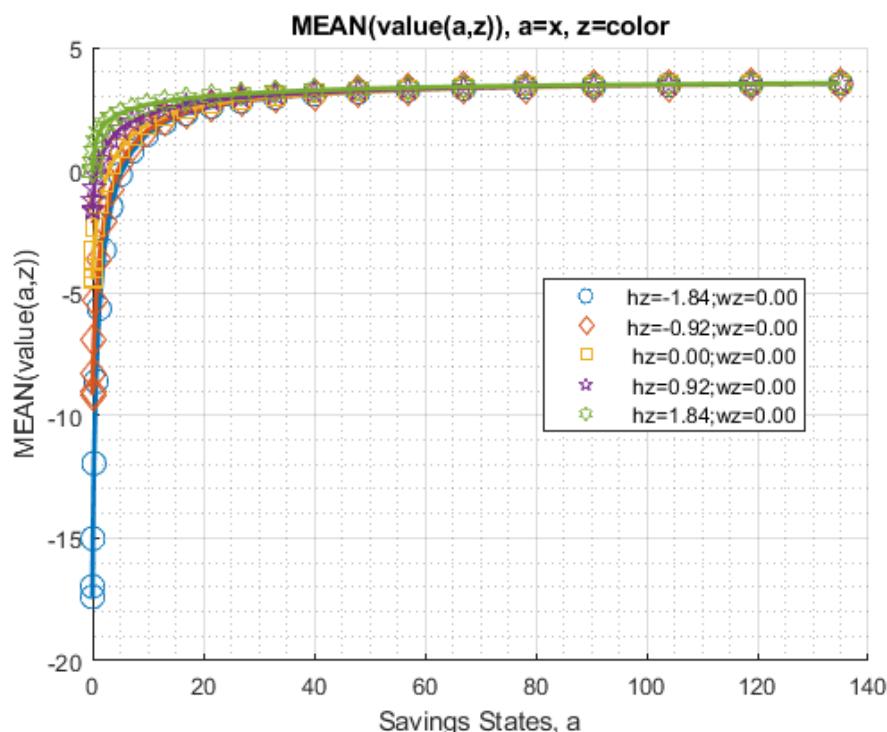
group	savings	mean_Hshock__1_8395	mean_Hshock__0_91976	mean_Hshock_0	mean_Hshock_1
1	0	-17.394	-9.166	-4.4582	-1.6
2	0.0097656	-16.968	-9.0297	-4.383	-1.5
3	0.078125	-15.017	-8.2656	-3.9672	-1.2
4	0.26367	-11.958	-6.9235	-3.2427	-0.73
5	0.625	-8.614	-5.2917	-2.3144	-0.18
6	1.2207	-5.6438	-3.6124	-1.3711	0.33
7	2.1094	-3.2727	-2.0767	-0.51202	0.8
8	3.3496	-1.4899	-0.79383	0.23904	1.2
9	5	-0.18672	0.21807	0.87882	1.6
10	7.1191	0.75696	0.99324	1.4131	1.9
11	9.7656	1.4411	1.5836	1.8494	2.2
12	12.998	1.9409	2.0281	2.1992	2.4
13	16.875	2.3126	2.3665	2.4779	2.6
14	21.455	2.5903	2.6255	2.6981	2.8
15	26.797	2.8009	2.8241	2.8737	2.
16	32.959	2.9638	2.9792	3.0129	3.0
17	40	3.0907	3.1014	3.1247	3.1
18	47.979	3.1906	3.1981	3.2147	3.2
19	56.953	3.2703	3.2756	3.2877	3.3
20	66.982	3.3347	3.3386	3.3473	3.3
21	78.125	3.3872	3.39	3.3965	3.4
22	90.439	3.4302	3.4324	3.4373	3.

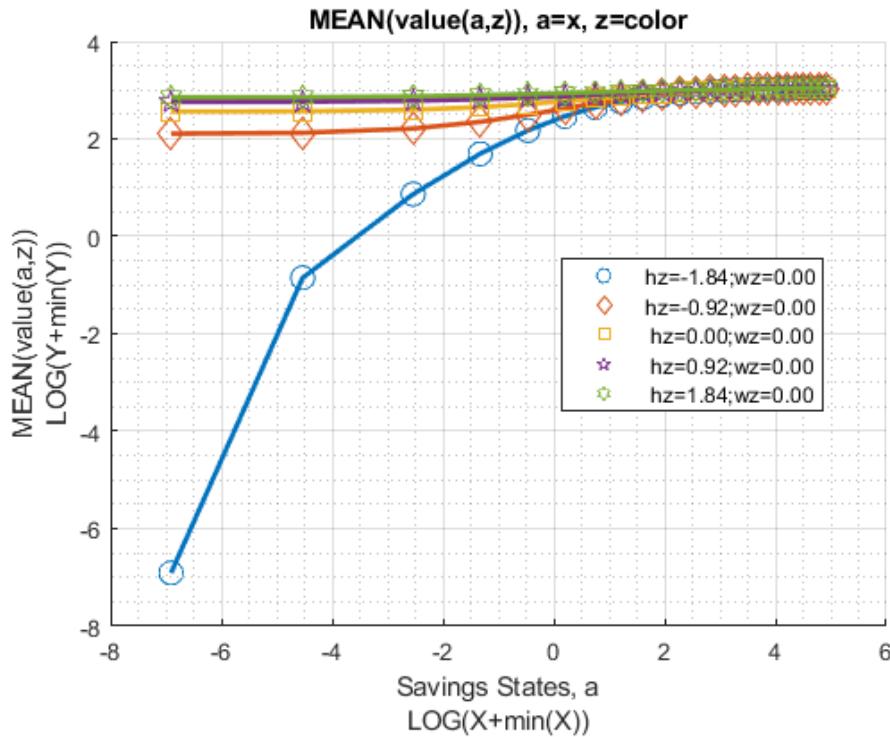
23	103.98	3.4659	3.4675	3.4712	3.4
24	118.82	3.4957	3.497	3.4998	3.5
25	135	3.5208	3.5218	3.524	3.
% Aprime Choice					
tb_az_ap = ff_summ_nd_array("MEAN(AP(A,Z))", ap_VFI, true, ["mean"], 4, 1, cl_mp_datasetdesc, ar_per					
xxx MEAN(AP(A,Z)) xxxxxxxxxxxxxxxxxxxxxxxxx					
group	savings	mean_Hshock__1_8395	mean_Hshock__0_91976	mean_Hshock_0	mean_Hshoc
1	0	1	1.1111	1.5694	2.59
2	0.0097656	1.0463	1.1852	1.6343	2.60
3	0.078125	1.7917	1.9815	2.1806	2.85
4	0.26367	2.9306	3.0231	3.2083	3.60
5	0.625	4.0509	4.1296	4.2454	4.51
6	1.2207	5.1296	5.2176	5.2639	5.38
7	2.1094	6.1065	6.1852	6.2361	6.24
8	3.3496	7.0324	7.0648	7.1574	7.14
9	5	7.9259	7.963	8.037	8.06
10	7.1191	8.8519	8.875	8.9306	9.00
11	9.7656	9.7824	9.7963	9.8472	9.92
12	12.998	10.593	10.625	10.639	10.7
13	16.875	11.481	11.491	11.537	11.5
14	21.455	12.407	12.407	12.426	12.4
15	26.797	13.282	13.296	13.306	13.3
16	32.959	14.116	14.12	14.153	14.
17	40	14.981	14.981	14.991	15.0
18	47.979	15.88	15.88	15.884	15.9
19	56.953	16.75	16.769	16.782	16.7
20	66.982	17.681	17.685	17.699	17.7
21	78.125	18.495	18.5	18.509	18.5
22	90.439	19.338	19.338	19.347	19.
23	103.98	20.25	20.264	20.269	20.2
24	118.82	21.097	21.097	21.13	21.1
25	135	21.963	21.968	21.977	21.9
% Consumption Choices					
tb_az_c = ff_summ_nd_array("MEAN(C(A,Z))", cons_VFI, true, ["mean"], 4, 1, cl_mp_datasetdesc, ar_per					
xxx MEAN(C(A,Z)) xxxxxxxxxxxxxxxxxxxxxxxxx					
group	savings	mean_Hshock__1_8395	mean_Hshock__0_91976	mean_Hshock_0	mean_Hshoc
1	0	0.31042	0.44057	0.71427	1.25
2	0.0097656	0.3215	0.4505	0.72262	1.26
3	0.078125	0.38861	0.50889	0.7788	1.3
4	0.26367	0.51067	0.62506	0.88538	1.43
5	0.625	0.686	0.78667	1.0455	1.60
6	1.2207	0.9128	0.98784	1.2592	1.86
7	2.1094	1.2523	1.3082	1.5599	2.26
8	3.3496	1.7189	1.8031	1.9833	2.71
9	5	2.3724	2.4345	2.6057	3.27
10	7.1191	3.1536	3.2269	3.4012	3.9
11	9.7656	4.0911	4.176	4.3322	4.83
12	12.998	5.4598	5.4763	5.7216	6.16
13	16.875	6.9683	7.0533	7.1634	7.64
14	21.455	8.5994	8.7201	8.9245	9.35

15	26.797	10.632	10.678	10.918	11.3
16	32.959	13.22	13.312	13.401	13.8
17	40	16.041	16.161	16.385	16.7
18	47.979	18.978	19.099	19.35	19.8
19	56.953	22.58	22.534	22.697	23.2
20	66.982	26.096	26.175	26.329	26.8
21	78.125	30.85	30.924	31.108	31.3
22	90.439	35.936	36.056	36.235	36.6
23	103.98	40.993	40.925	41.151	41.7
24	118.82	47.079	47.199	47.025	47.5
25	135	53.5	53.545	53.689	54.1

Graph Mean Values:

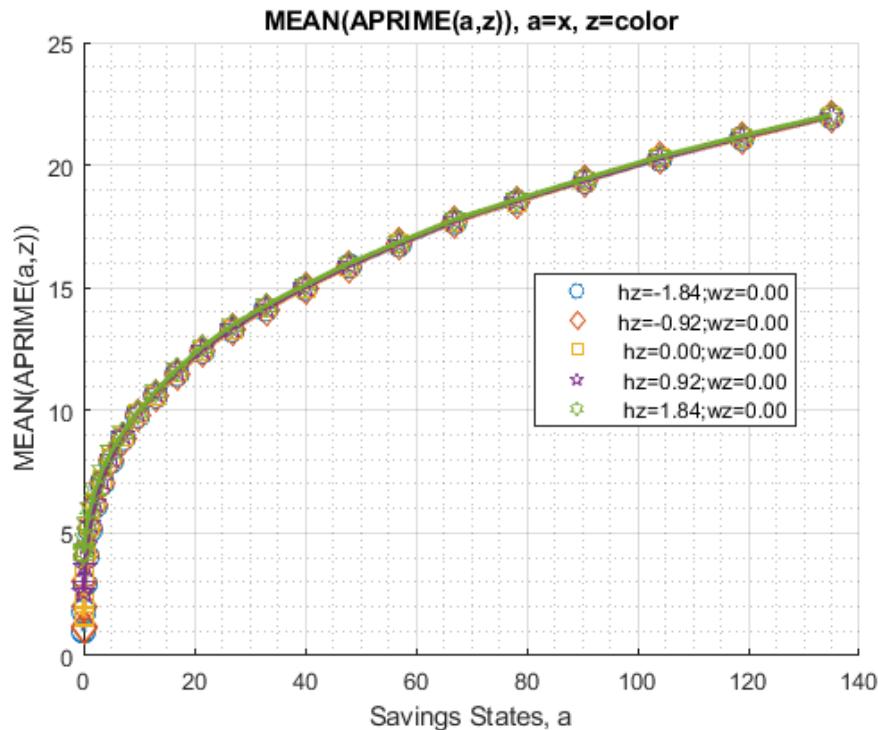
```
mp_support_graph('cl_st_graph_title') = {'MEAN(value(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(a,z))'};
ff_graph_grid((tb_az_v{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);
```

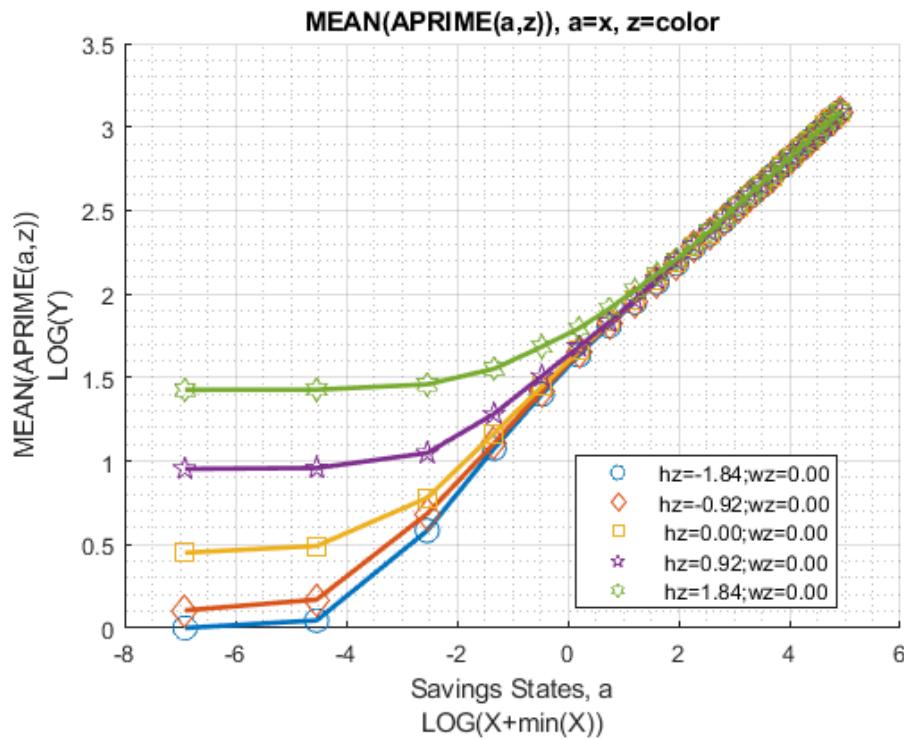




Graph Mean Savings Choices:

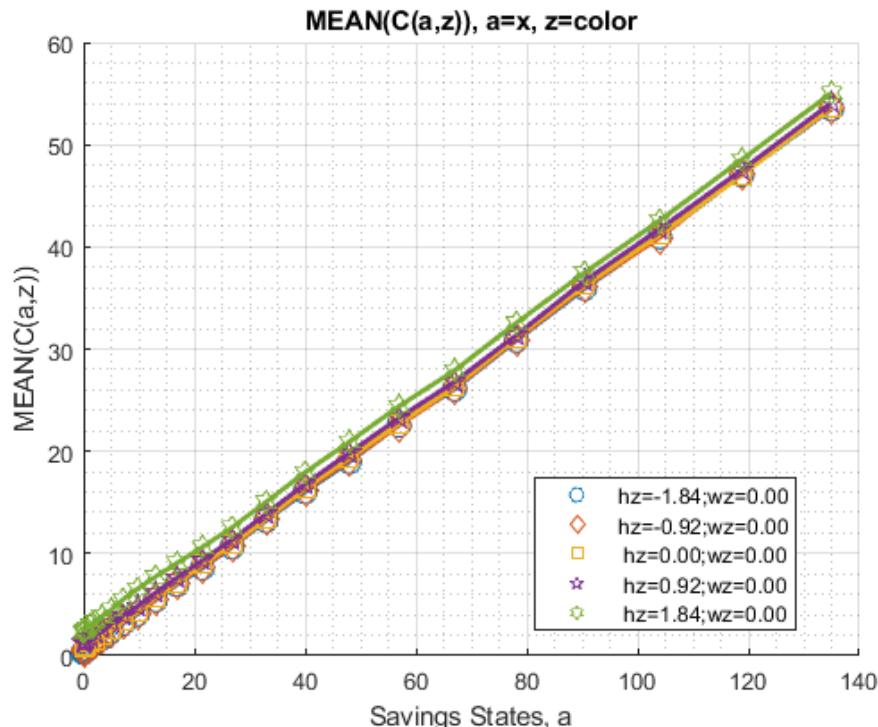
```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(a,z))'};
ff_graph_grid((tb_az_ap{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```

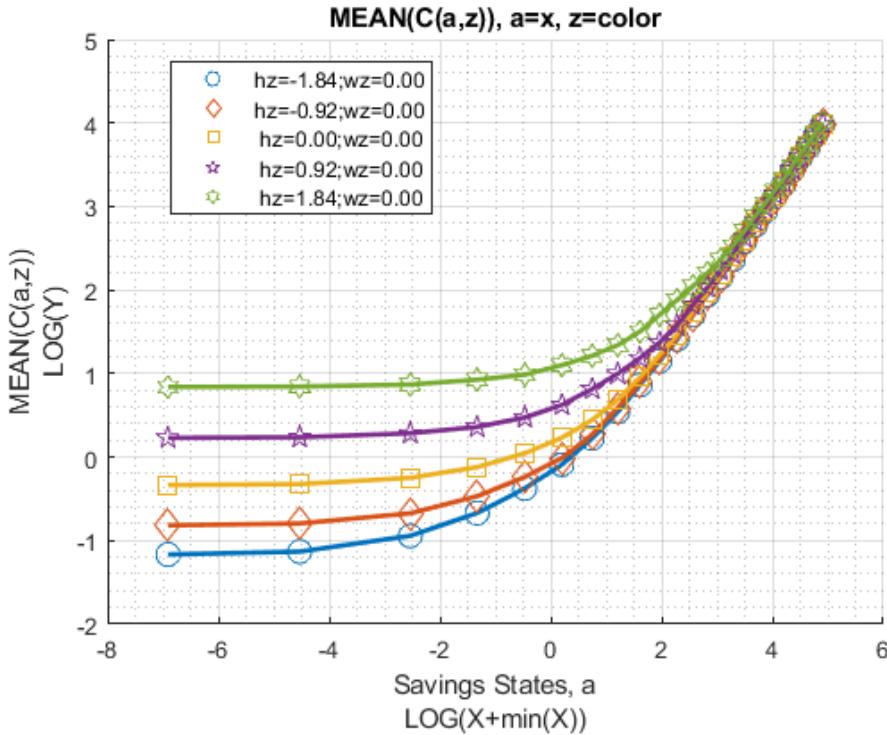




Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(a,z))'};
ff_graph_grid((tb_az_c{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```





4.2.4 Analyze Kids and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

% Generate some Data

```
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["k0M0", "K1M0", "K2M0", "k0M1", "K1M1", "K2M1"];
```

```
mp_support_graph('cl_st_xtitle') = {'Age'};
```

```
mp_support_graph('st_legend_loc') = 'best';
```

```
mp_support_graph('bl_graph_logy') = true; % do not log
```

```
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
```

```
mp_support_graph('cl_scatter_shapes') = { 'o', 'd' , 's', 'o', 'd', 's'};
```

```
mp_support_graph('cl_colors') = {'red', 'red', 'red', 'blue', 'blue', 'blue'};
```

```
MEAN(VAL(KM,J)), MEAN(AP(KM,J)), MEAN(C(KM,J))
```

Tabulate value and policies:

% Set

```
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
```

```
ar_permute = [2,3,4,1,6,5];
```

% Value Function

```
tb_az_v = ff_summ_nd_array("MEAN(VAL(KM,J))", V_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_per
```

xxx	MEAN(VAL(KM,J))	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx						
group	kids	marry	mean_age_19	mean_age_22	mean_age_27	mean_age_32	mean_age_37	
---	---	---	-----	-----	-----	-----	-----	-----
1	1	0	1.4134	1.6987	1.8877	1.9428	1.9141	
2	2	0	-0.11224	0.38086	0.75969	0.96426	1.0617	
3	3	0	-0.88391	-0.40356	-0.0148	0.20487	0.31925	
4	1	1	1.9721	2.188	2.3283	2.3713	2.3479	
5	2	1	0.97335	1.2928	1.5422	1.6825	1.7486	
6	3	1	0.52474	0.81914	1.0571	1.1945	1.2619	

% Aprime Choice

```

tb_az_ap = ff_summ_nd_array("MEAN(AP(KM,J))", ap_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_pe)

xxx MEAN(AP(KM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry mean_age_19 mean_age_22 mean_age_27 mean_age_32 mean_age_36
----- ----- ----- ----- ----- ----- ----- -----
1 1 0 12.948 12.92 13.052 13.152 13.22
2 2 0 12.924 12.88 13.004 13.092 13.156
3 3 0 12.856 12.848 12.972 13.08 13.104
4 1 1 12.86 12.856 12.972 13.072 13.132
5 2 1 12.876 12.82 12.956 13.028 13.096
6 3 1 12.8 12.784 12.912 12.984 13.056

% Consumption Choices
tb_az_c = ff_summ_nd_array("MEAN(C(KM,J))", cons_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_pe)

xxx MEAN(C(KM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry mean_age_19 mean_age_22 mean_age_27 mean_age_32 mean_age_36
----- ----- ----- ----- ----- ----- -----
1 1 0 6.6347 6.7448 6.9773 7.1425 7.2321
2 2 0 6.6476 6.7581 6.9907 7.1658 7.2726
3 3 0 6.6714 6.7696 7.0001 7.1702 7.8471
4 1 1 6.885 7.0096 7.2673 7.4592 7.5807
5 2 1 6.856 6.987 7.2319 7.4245 7.5495
6 3 1 6.8708 6.9855 7.2175 7.4148 7.5369

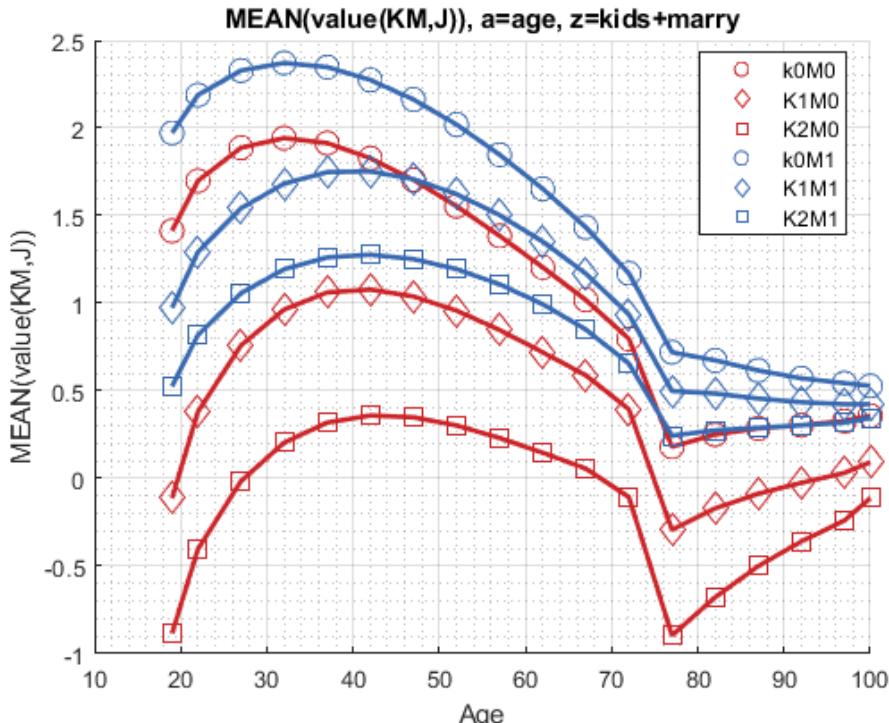
```

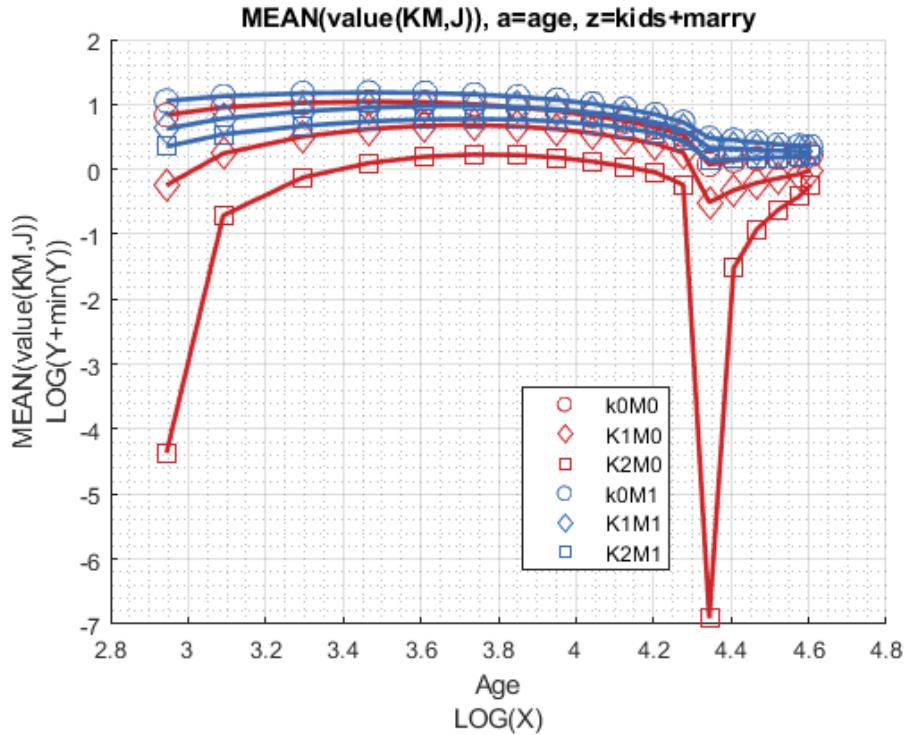
Graph Mean Values:

```

mp_support_graph('cl_st_graph_title') = {'MEAN(value(KM,J)), a=age, z=kids+marry'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(KM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);

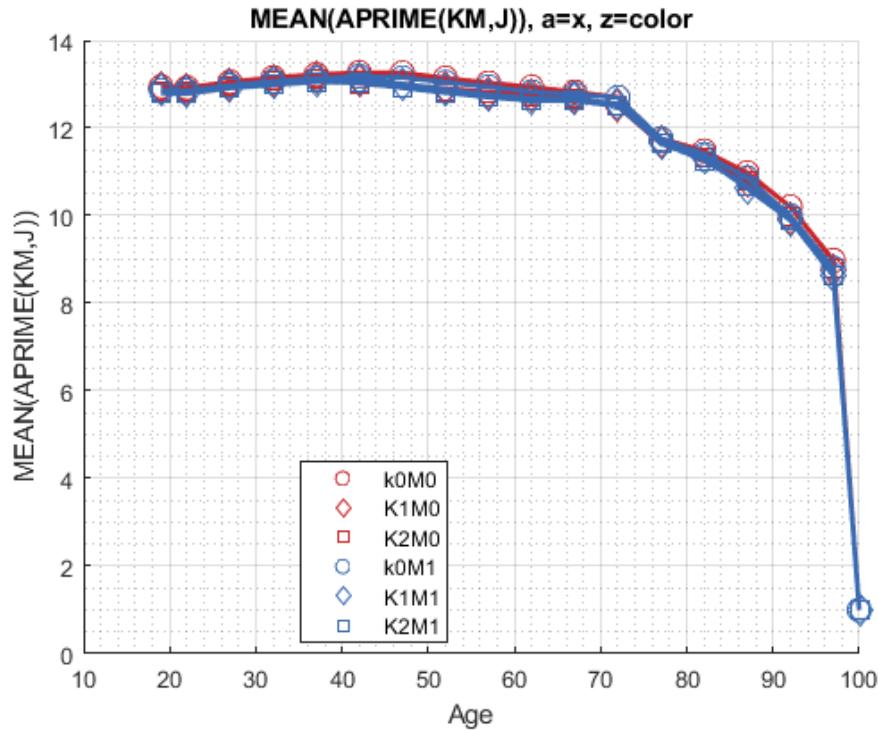
```

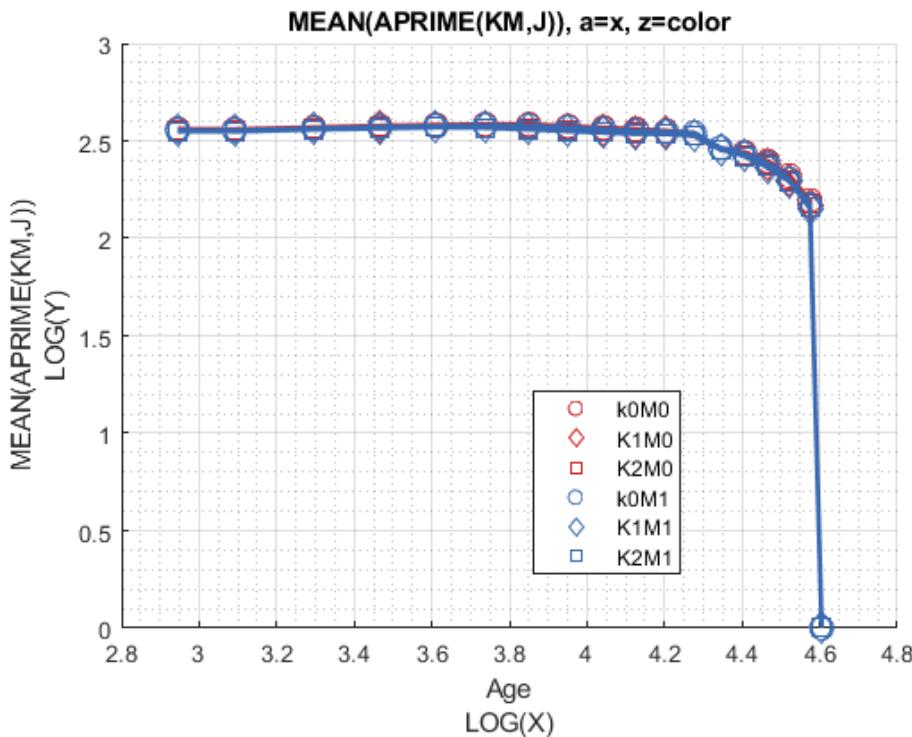




Graph Mean Savings Choices:

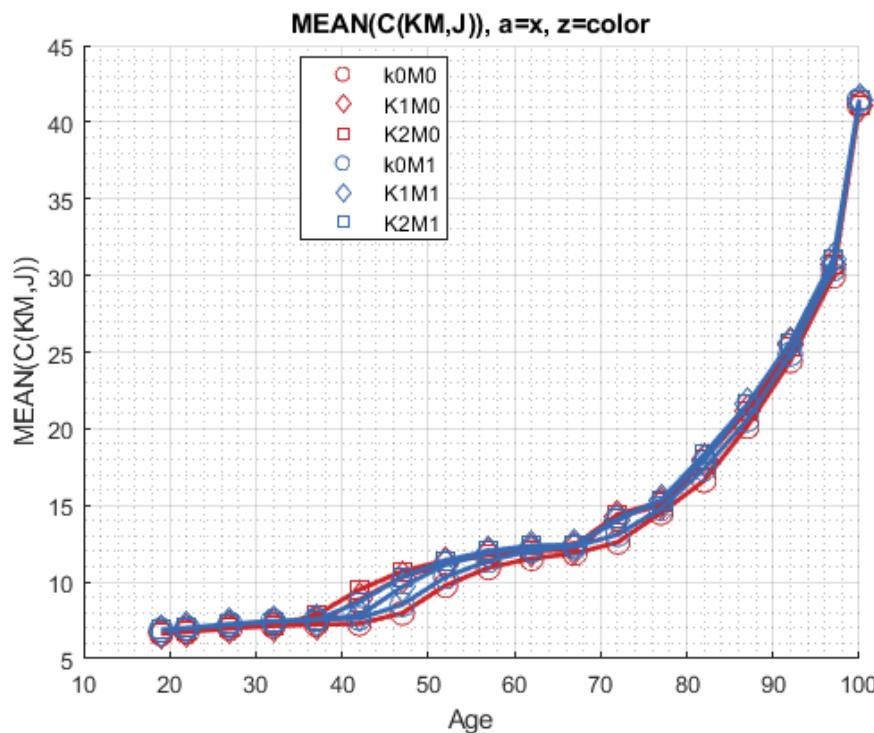
```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(KM,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(KM,J))'};
ff_graph_grid((tb_az_ap{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```

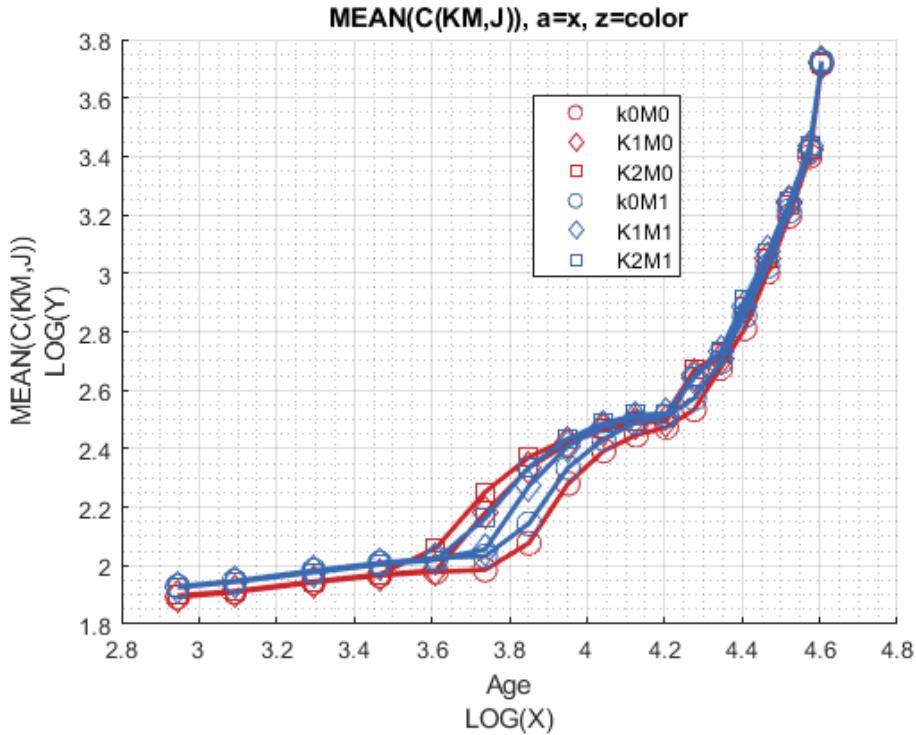




Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(KM,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(KM,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```





4.2.5 Analyze Education and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["E0M0", "E1M0", "E0M1", "E1M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {'*', 'p', '*', 'p'};
mp_support_graph('cl_colors') = {'red', 'red', 'blue', 'blue'};

MEAN(VAL(EKM,J)), MEAN(AP(EKM,J)), MEAN(C(EKM,J))
```

Tabulate value and policies:

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

xxx	MEAN(VAL(EKM,J))	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	group	edu	marry	mean_age_19	mean_age_22	mean_age_27	mean_age_32	mean_age_37
---	---	-----	---	---	---	-----	-----	-----	-----	-----
1	0	0	1	0	0	-0.27576	0.0889	0.38392	0.55759	0.6492
2	1	0	2	1	0	0.55395	1.0284	1.3712	1.5171	1.5475
3	0	1	3	0	1	0.78157	1.0452	1.254	1.3788	1.4422
4	1	1	4	1	1	1.5319	1.8215	2.0311	2.12	2.1301

% Aprime Choice

```
tb_az_ap = ff_summ_nd_array("MEAN(AP(EKM,J))", ap_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_p
```

```

xxx MEAN(AP(EKM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx
group   edu    marry   mean_age_19   mean_age_22   mean_age_27   mean_age_32   mean_age_37
-----  ---  -----  -----  -----  -----  -----  -----
1       0      0       12.989     12.976     13.032     13.091     13.125
2       1      0       12.829     12.789     12.987     13.125     13.195
3       0      1       12.933     12.923     12.976     13.021     13.067
4       1      1       12.757     12.717     12.917     13.035     13.123

% Consumption Choices
tb_az_c = ff_summ_nd_array("MEAN(C(EKM,J))", cons_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_p

xxx MEAN(C(EKM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx
group   edu    marry   mean_age_19   mean_age_22   mean_age_27   mean_age_32   mean_age_37
-----  ---  -----  -----  -----  -----  -----  -----
1       0      0       6.6262     6.6905     6.8287     6.9345     7.2519
2       1      0       6.6762     6.8246     7.1501     7.3846     7.6493
3       0      1       6.8114     6.8929     7.0479     7.1732     7.262
4       1      1       6.9297     7.0952     7.4299     7.6925     7.8494

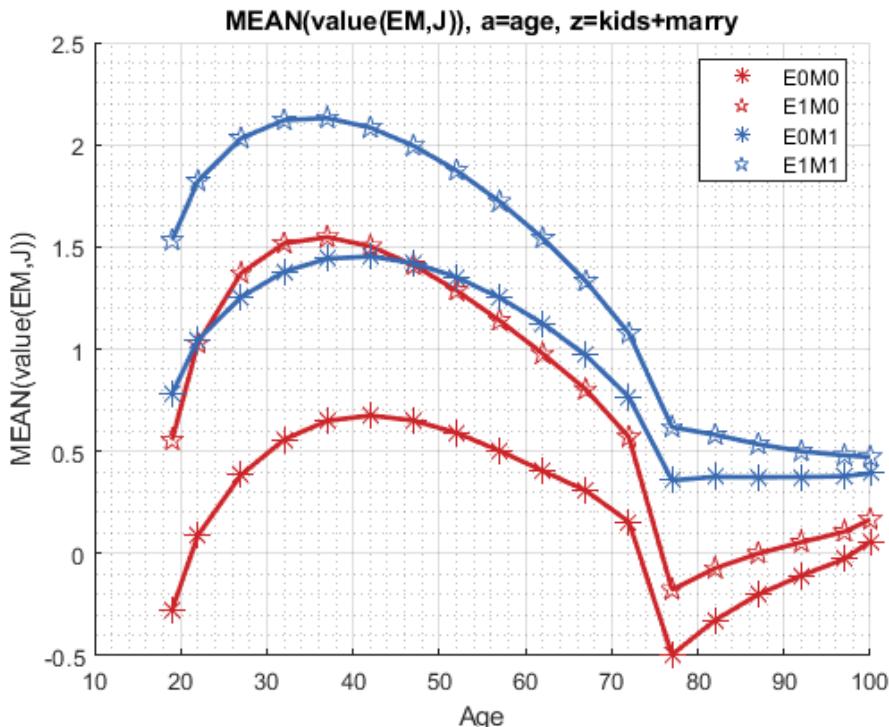
```

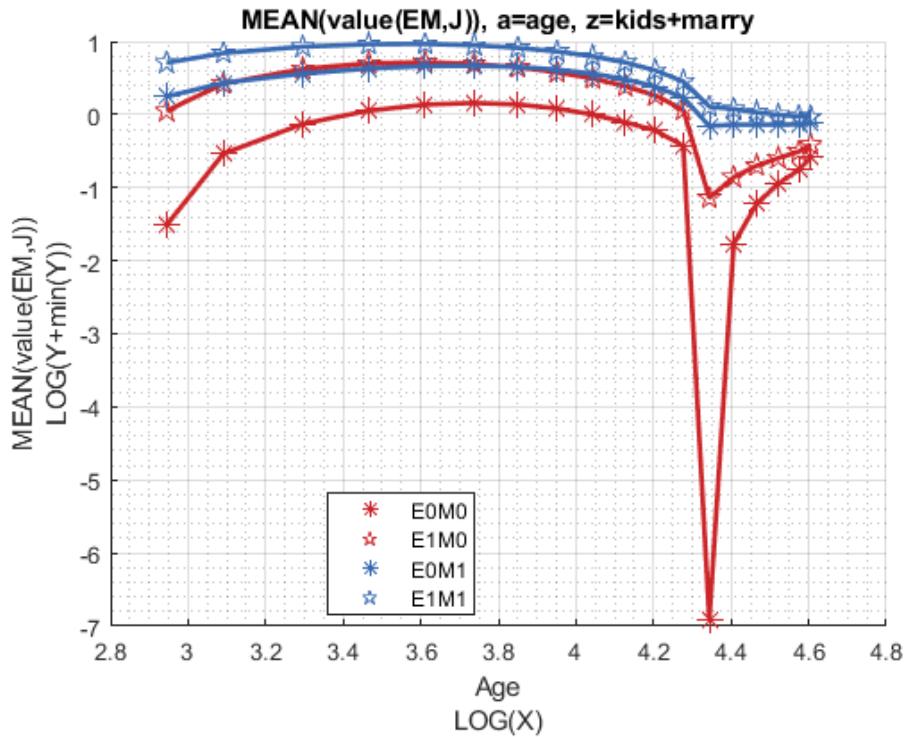
Graph Mean Values:

```

mp_support_graph('cl_st_graph_title') = {'MEAN(value(EM,J)), a=age, z=kids+marry'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(EM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);

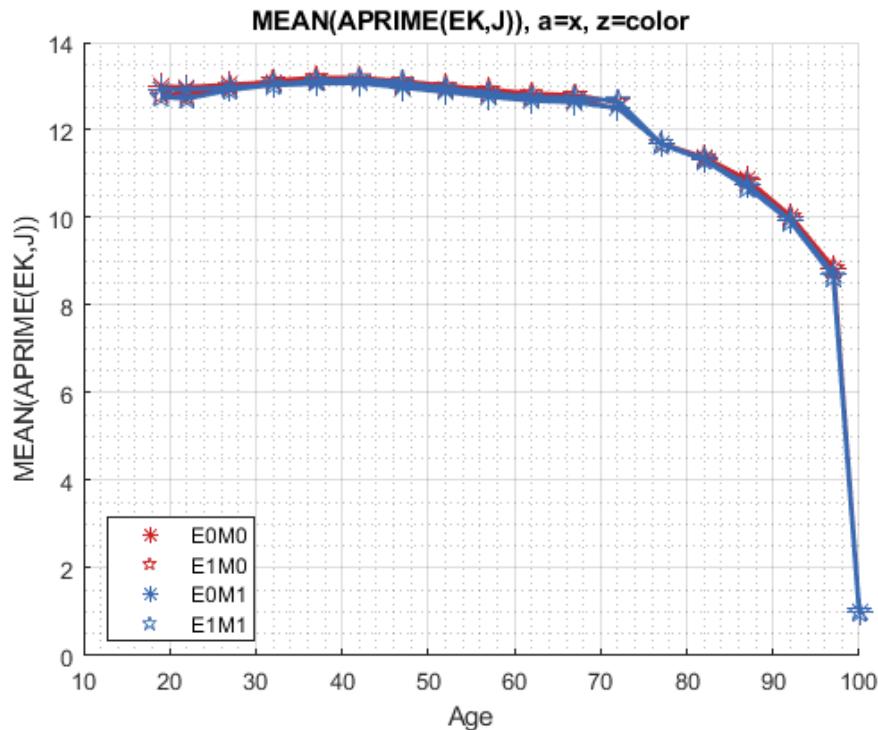
```

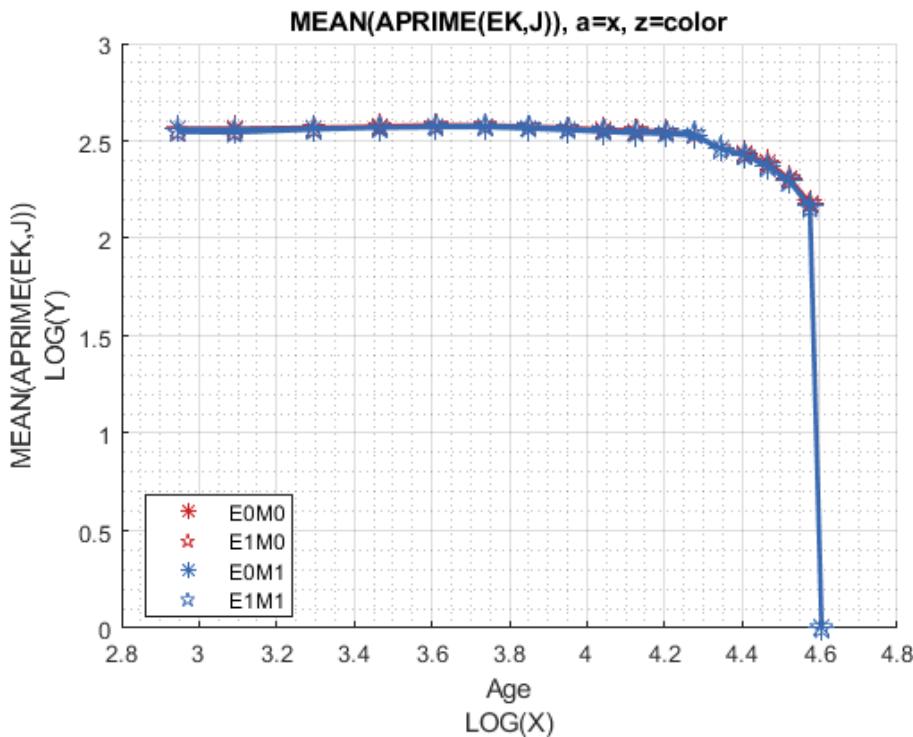




Graph Mean Savings Choices:

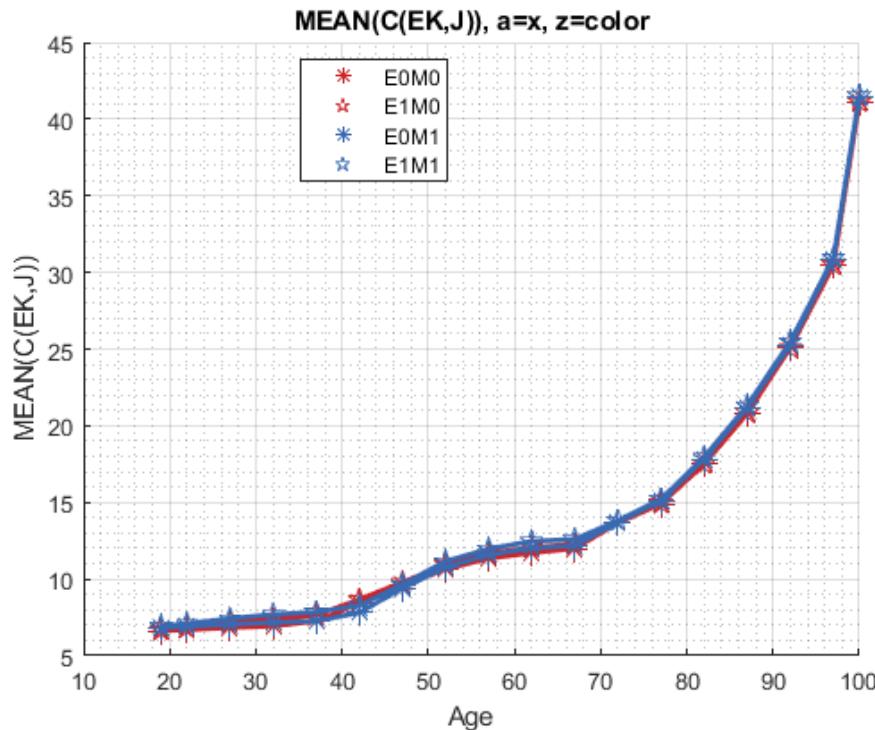
```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(EK,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(EK,J))'};
ff_graph_grid((tb_az_ap{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```

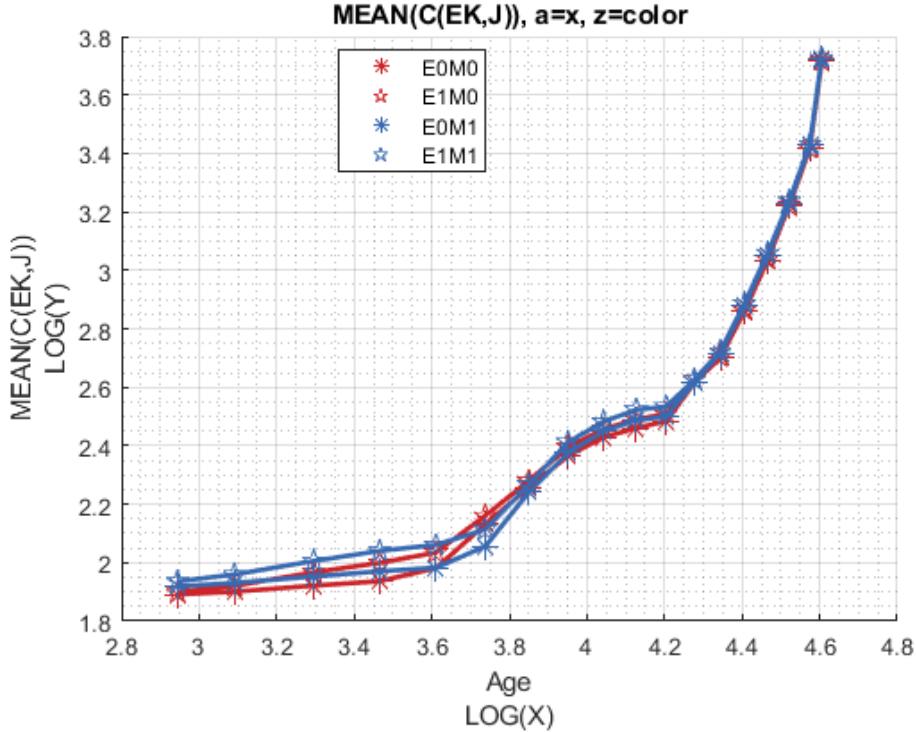




Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(EK,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(EK,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```





4.3 Small Test Exact Solution Vectorized Bisection

This is the example vignette for function: `snw_vfi_main_bisec_vec` from the [PrjOptiSNW Package](#). This function solves for policy function with vectorized bisection. Small Solution Analysis. Small Solution Analysis, husband 5 shocks, wife 1 shocks.

4.3.1 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_bisec_vec(mp_param);
```

```
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:18 of 17, time-this-age:0.019266
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:17 of 17, time-this-age:0.020088
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:16 of 17, time-this-age:0.019484
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:15 of 17, time-this-age:0.020422
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:14 of 17, time-this-age:0.018799
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:13 of 17, time-this-age:0.019274
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:12 of 17, time-this-age:0.019728
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:11 of 17, time-this-age:0.020101
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:10 of 17, time-this-age:0.019674
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:9 of 17, time-this-age:0.019648
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:8 of 17, time-this-age:0.019126
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:7 of 17, time-this-age:0.018838
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:6 of 17, time-this-age:0.01893
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:5 of 17, time-this-age:0.018729
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:4 of 17, time-this-age:0.018653
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:3 of 17, time-this-age:0.018577
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:2 of 17, time-this-age:0.018797
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:1 of 17, time-this-age:0.019341
Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_small;SNW_MP_CONTROL=default_base;time=0.35835
```

4.3.2 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=%3.2f'));
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'}, {'Hshock', 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});
```

4.3.3 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_permute);

xxx MEAN(VAL(A,Z)) xxxxxxxxxxxxxxxxxxxxxxxxx
group      savings      mean_Hshock__1_8395      mean_Hshock__0_91976      mean_Hshock_0      mean_Hshock_1
-----      -----
1            0           -17.393          -9.1596          -4.4164         -1.5
2          0.0097656        -16.967          -9.023          -4.3405         -1.5
3          0.078125         -14.925          -8.2554          -3.9177         -1.2
4          0.26367          -11.699          -6.8681          -3.1808         -0.6
5            0.625          -8.2751          -5.1669          -2.2785         -0.13
6            1.2207          -5.3024          -3.4437          -1.3431         0.38
7            2.1094          -2.9816          -1.9066          -0.47797        0.86
8            3.3496          -1.2609          -0.64407         0.28612        1.3
9              5           -0.012543          0.34403          0.9369        1.6
10           7.1191           0.88751           1.097          1.4725        1.9
11           9.7656           1.5392           1.665          1.9037        2.2
12          12.998            2.0158           2.0932          2.2465        2.5
13          16.875            2.3684           2.4172          2.5172        2.6
14          21.455            2.6328           2.6644          2.7307        2.8
```

15	26.797	2.8339	2.8549	2.8997	2.
16	32.959	2.989	3.0032	3.034	3.0
17	40	3.1102	3.12	3.1416	3.1
18	47.979	3.2059	3.2128	3.2282	3.2
19	56.953	3.2825	3.2875	3.2986	3.3
20	66.982	3.3443	3.348	3.3562	3.3
21	78.125	3.3948	3.3975	3.4036	3.4
22	90.439	3.4364	3.4384	3.443	3.4
23	103.98	3.4709	3.4724	3.476	3.4
24	118.82	3.4998	3.501	3.5037	3.5
25	135	3.5241	3.5251	3.5272	3.5

% Aprime Choice

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

xxx MEAN(AP(A,Z)) xxxxxxxxxxxxxxxxxxxxxxxxx					
group	savings	mean_Hshock__1_8395	mean_Hshock__0_91976	mean_Hshock_0	mean_Hshoc
---	-----	-----	-----	-----	-----
1	0	2.7511e-05	0.0021997	0.046353	0.23
2	0.0097656	0.00054711	0.0036547	0.049525	0.24
3	0.078125	0.021674	0.027305	0.079481	0.27
4	0.26367	0.13129	0.14249	0.19451	0.38
5	0.625	0.38703	0.404	0.44756	0.63
6	1.2207	0.83381	0.85545	0.90672	1.0
7	2.1094	1.5206	1.5442	1.6064	1.7
8	3.3496	2.477	2.5013	2.5629	2.6
9	5	3.7541	3.7788	3.8405	3.9
10	7.1191	5.416	5.4412	5.5038	5.6
11	9.7656	7.4668	7.4912	7.5553	7.7
12	12.998	9.9008	9.9211	9.9832	10.
13	16.875	12.918	12.94	12.995	13.
14	21.455	16.519	16.538	16.594	16.
15	26.797	20.59	20.608	20.657	20.
16	32.959	25.295	25.313	25.358	25.
17	40	30.657	30.68	30.732	30.
18	47.979	36.751	36.772	36.831	36
19	56.953	43.764	43.786	43.839	44.
20	66.982	51.594	51.617	51.677	51
21	78.125	59.942	59.965	60.024	60.
22	90.439	69.254	69.278	69.34	69.
23	103.98	79.741	79.762	79.821	79.
24	118.82	91.103	91.126	91.188	91.
25	135	103.46	103.48	103.53	103

% Consumption Choices

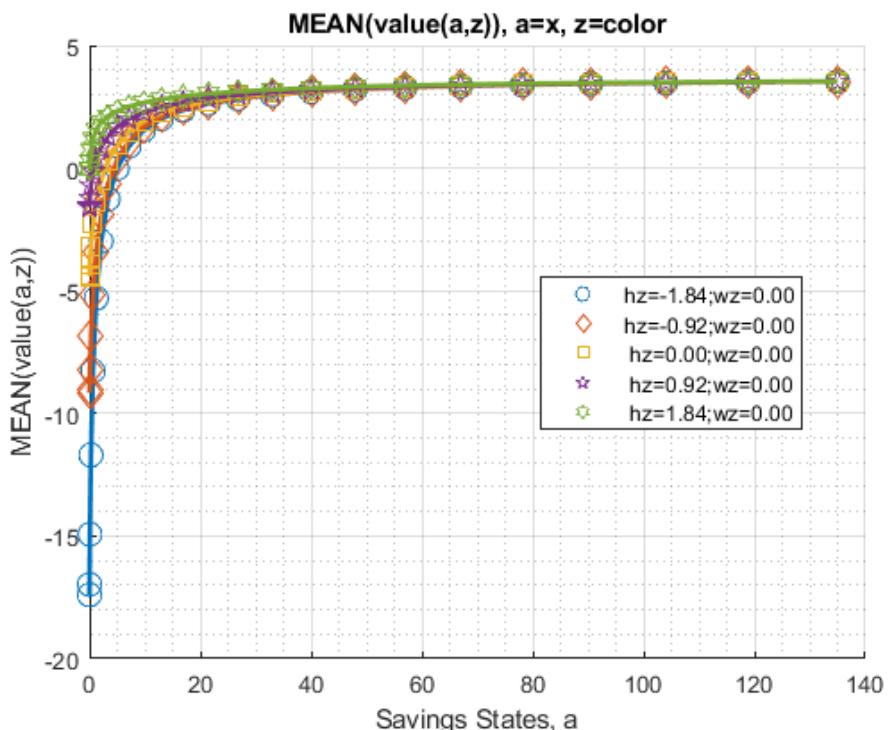
```
tb_az_c = ff_summ_nd_array("MEAN(C(A,Z))", cons_VFI, true, ["mean"], 4, 1, cl_mp_datasetdesc, ar_per
```

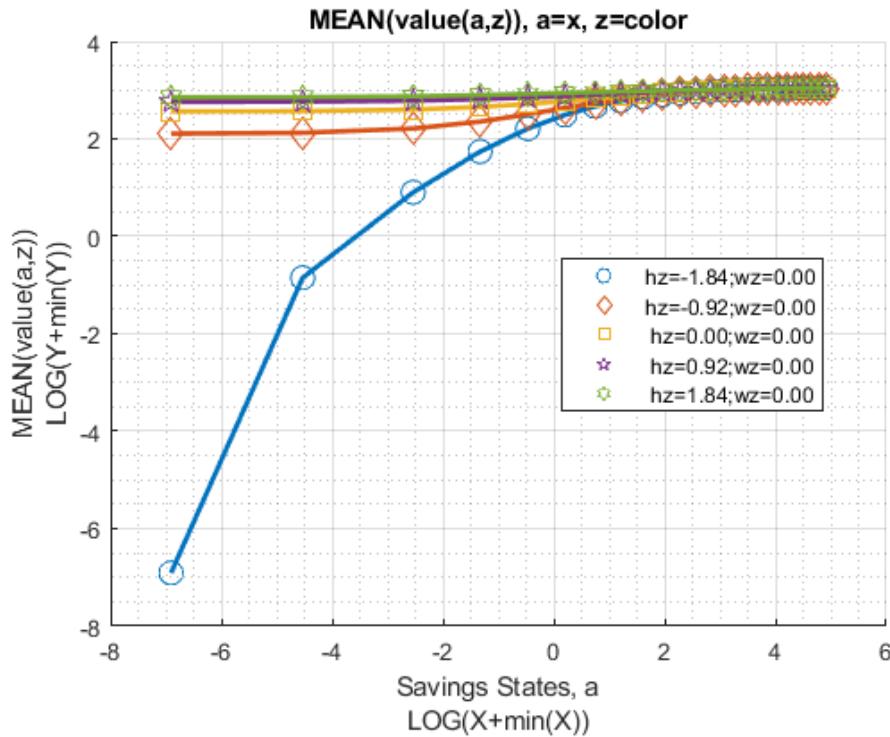
xxx MEAN(C(A,Z)) xxxxxxxxxxxxxxxxxxxxxxxxx					
group	savings	mean_Hshock__1_8395	mean_Hshock__0_91976	mean_Hshock_0	mean_Hshoc
---	-----	-----	-----	-----	-----
1	0	0.3104	0.44	0.69897	1.22
2	0.0097656	0.3214	0.45001	0.70723	1.23
3	0.078125	0.3809	0.50664	0.75724	1.28
4	0.26367	0.48992	0.60921	0.8592	1.39
5	0.625	0.65917	0.77131	1.0284	1.55
6	1.2207	0.91141	1.0172	1.2649	1.80

7	2.1094	1.2649	1.3671	1.6019	2.18
8	3.3496	1.7572	1.8573	2.0907	2.69
9	5	2.4045	2.503	2.7347	3.30
10	7.1191	3.2104	3.3074	3.537	4.07
11	9.7656	4.2385	4.3358	4.5627	5.08
12	12.998	5.5627	5.6635	5.8917	6.41
13	16.875	7.0504	7.1499	7.3847	7.90
14	21.455	8.7708	8.8721	9.1059	9.63
15	26.797	10.904	11.007	11.247	11.7
16	32.959	13.355	13.457	13.7	14.2
17	40	16.168	16.266	16.502	17.0
18	47.979	19.337	19.437	19.666	20.2
19	56.953	22.744	22.843	23.078	23.6
20	66.982	26.557	26.654	26.883	27.4
21	78.125	31.145	31.242	31.47	32.0
22	90.439	36.128	36.224	36.451	36.9
23	103.98	41.364	41.464	41.692	42.2
24	118.82	47.222	47.319	47.545	48.0
25	135	53.652	53.751	53.983	54.5

Graph Mean Values:

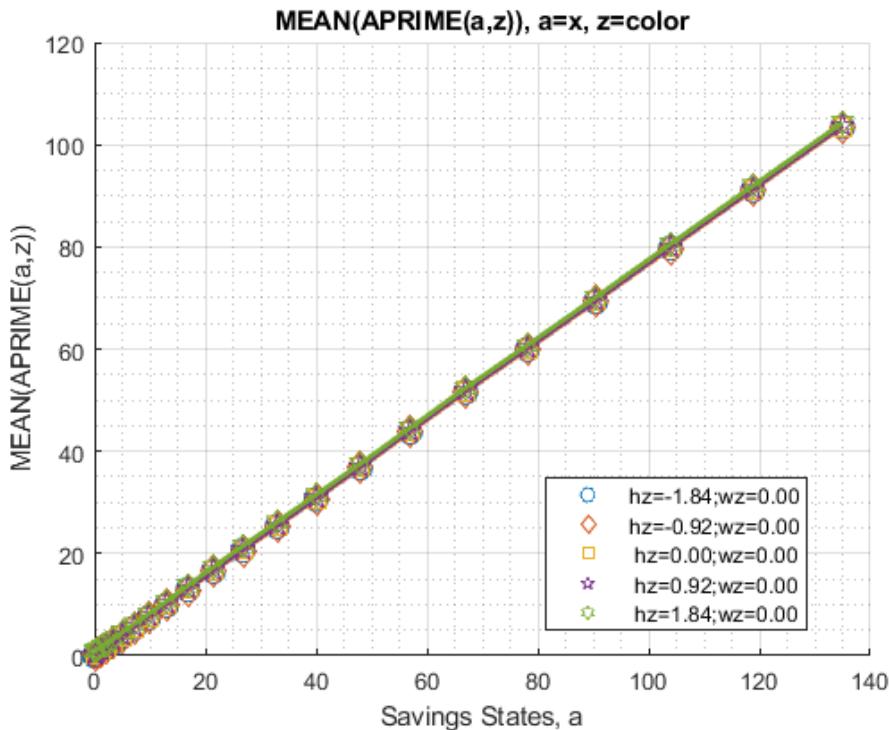
```
mp_support_graph('cl_st_graph_title') = {'MEAN(value(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(a,z))'};
ff_graph_grid((tb_az_v{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```

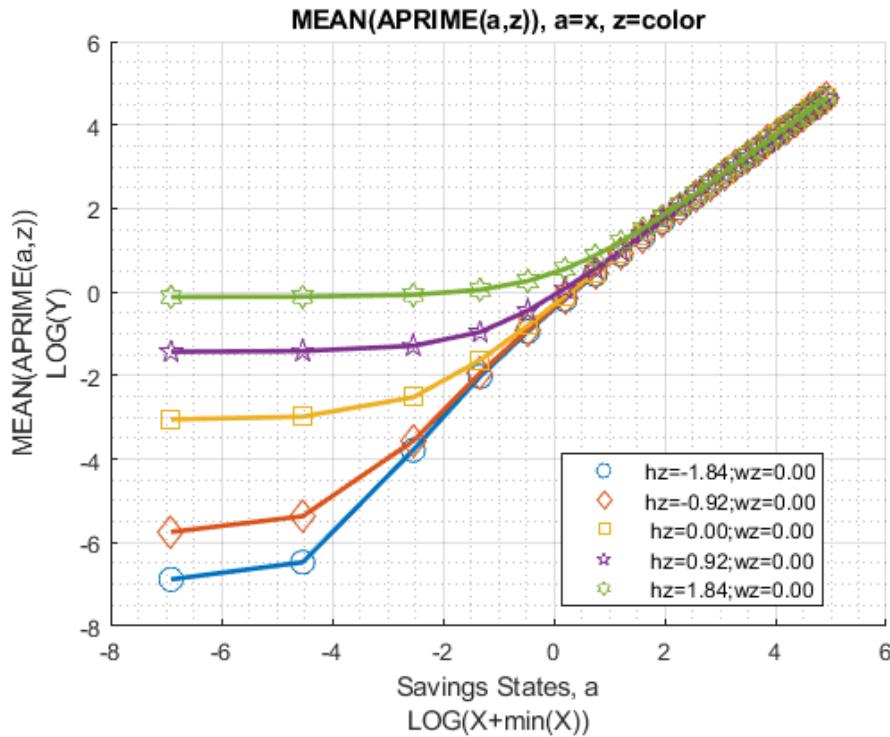




Graph Mean Savings Choices:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(a,z))'};
ff_graph_grid((tb_az_ap{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```



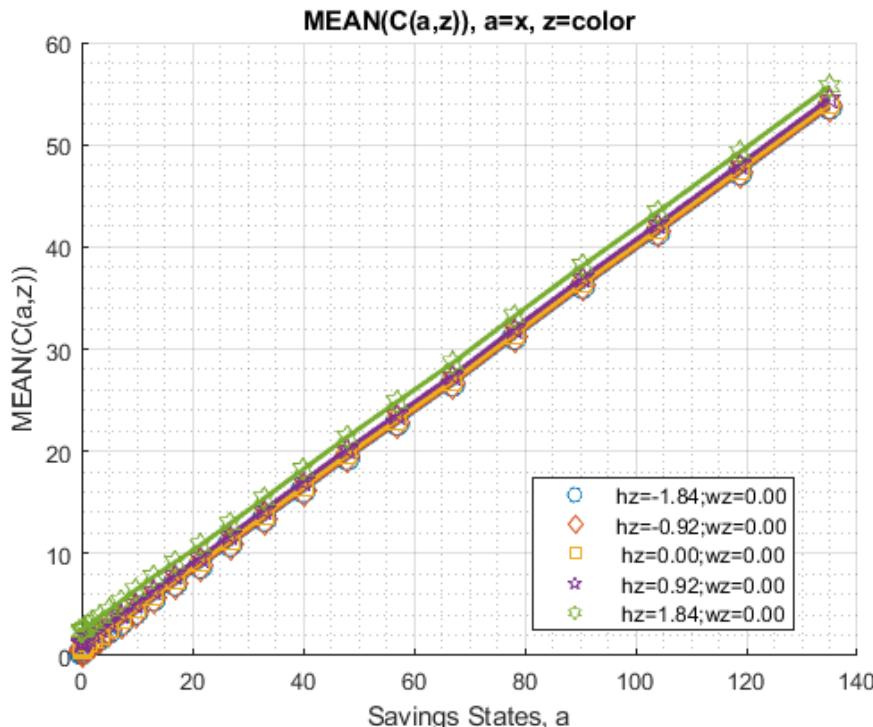


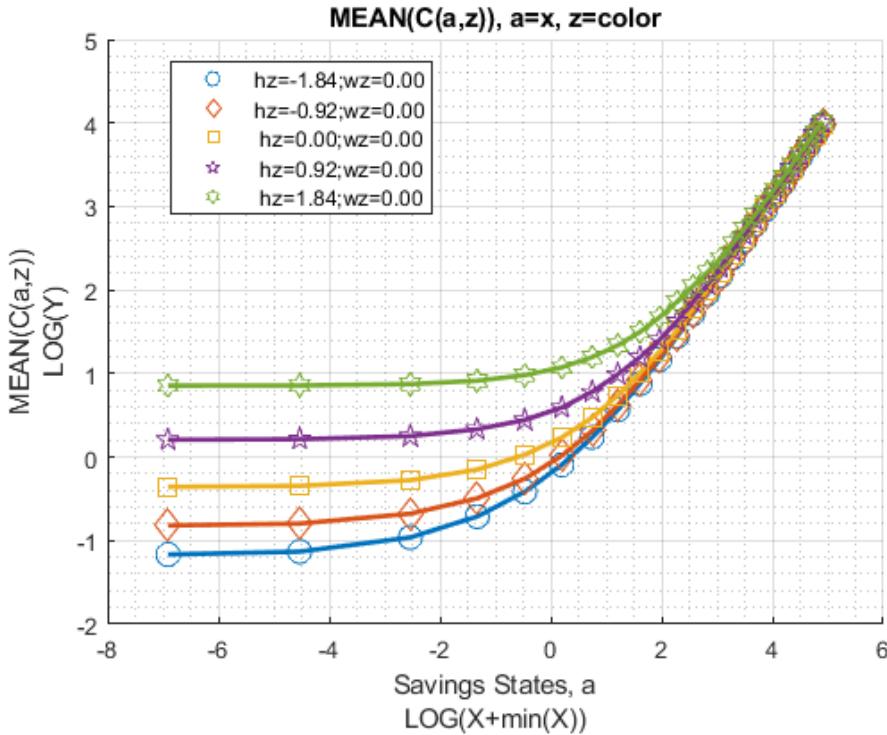
Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(a,z)), a=x, z=color'};  

mp_support_graph('cl_st_ytitle') = {'MEAN(C(a,z))'};  

ff_graph_grid((tb_az_c{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```





4.3.4 Analyze Kids and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["k0M0", "K1M0", "K2M0", "k0M1", "K1M1", "K2M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = { 'o', 'd', 's', 'o', 'd', 's' };
mp_support_graph('cl_colors') = { 'red', 'red', 'red', 'blue', 'blue', 'blue' };
MEAN(VAL(KM,J)), MEAN(AP(KM,J)), MEAN(C(KM,J))

Tabulate value and policies:
```

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

xxx	MEAN(VAL(KM,J))	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	mean_age_19	mean_age_22	mean_age_27	mean_age_32	mean_age_38
group	kids	marry	-----	-----	-----	-----	-----
1	1	0	1.4699	1.7485	1.9344	1.9907	1.9652
2	2	0	-0.020723	0.46111	0.83504	1.0389	1.1397
3	3	0	-0.77111	-0.30145	0.081934	0.30157	0.41928
4	1	1	2.0205	2.2326	2.3705	2.4138	2.3913
5	2	1	1.0463	1.3598	1.6057	1.745	1.8111
6	3	1	0.61068	0.90045	1.1354	1.2721	1.3395

```
% Aprime Choice
```

```

tb_az_ap = ff_summ_nd_array("MEAN(AP(KM,J))", ap_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_pe)

xxx MEAN(AP(KM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry mean_age_19 mean_age_22 mean_age_27 mean_age_32 mean_age_36
----- ----- ----- ----- ----- ----- -----
1 1 0 34.929 34.724 34.662 34.55 34.357
2 2 0 34.6 34.331 34.195 33.99 33.687
3 3 0 34.185 33.965 33.873 33.7 33.421
4 1 1 34.819 34.614 34.562 34.453 34.262
5 2 1 34.667 34.448 34.36 34.201 33.945
6 3 1 34.3 34.115 34.061 33.932 33.7

% Consumption Choices
tb_az_c = ff_summ_nd_array("MEAN(C(KM,J))", cons_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_pe)

xxx MEAN(C(KM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry mean_age_19 mean_age_22 mean_age_27 mean_age_32 mean_age_36
----- ----- ----- ----- ----- -----
1 1 0 6.8551 7.1756 7.502 7.8205 8.1483
2 2 0 7.1843 7.5683 7.9695 8.3802 8.8184
3 3 0 7.5997 7.934 8.2911 8.6703 9.0841
4 1 1 7.1871 7.5271 7.8696 8.209 8.5573
5 2 1 7.3044 7.6564 8.0306 8.4165 8.826
6 3 1 7.6479 7.9629 8.3009 8.6543 9.0382

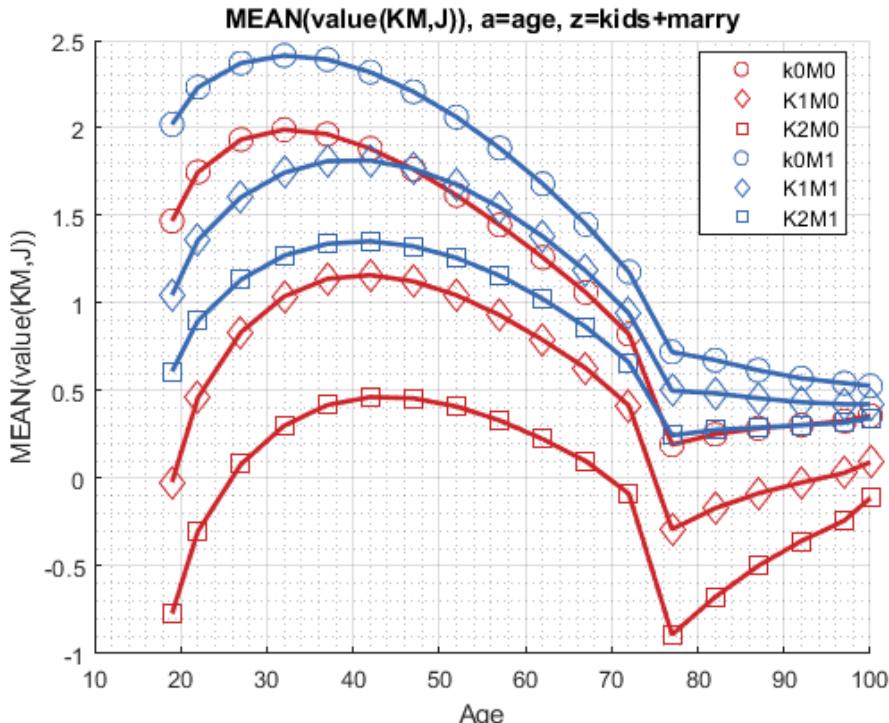
```

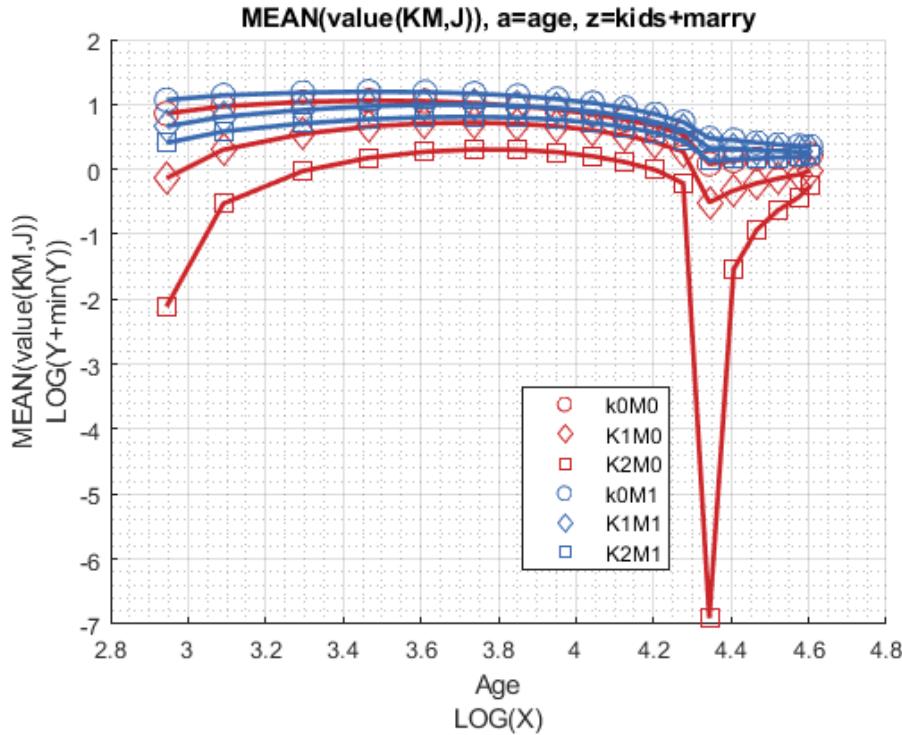
Graph Mean Values:

```

mp_support_graph('cl_st_graph_title') = {'MEAN(value(KM,J)), a=age, z=kids+marry'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(KM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);

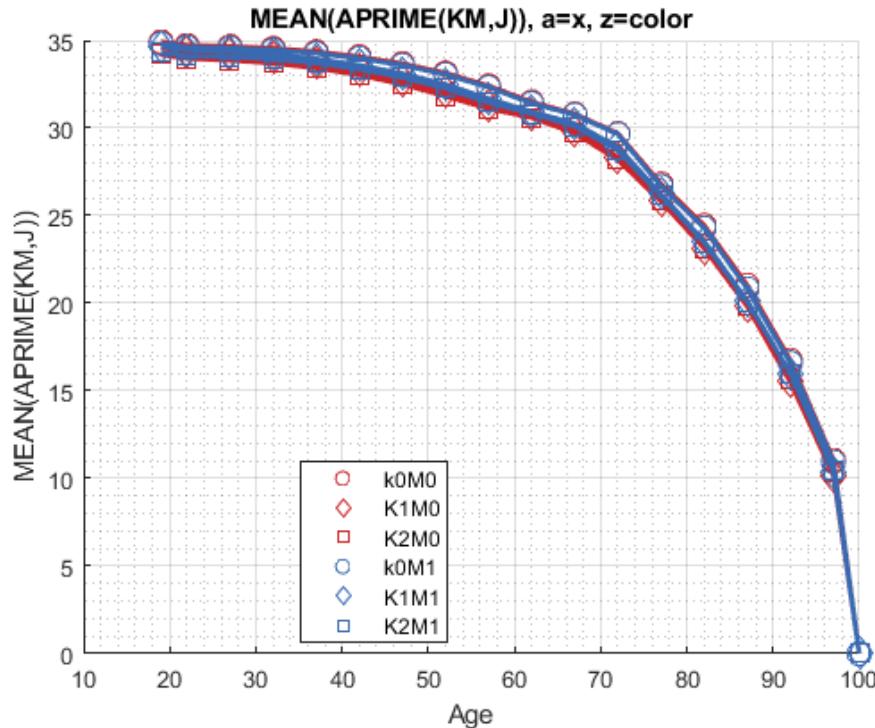
```

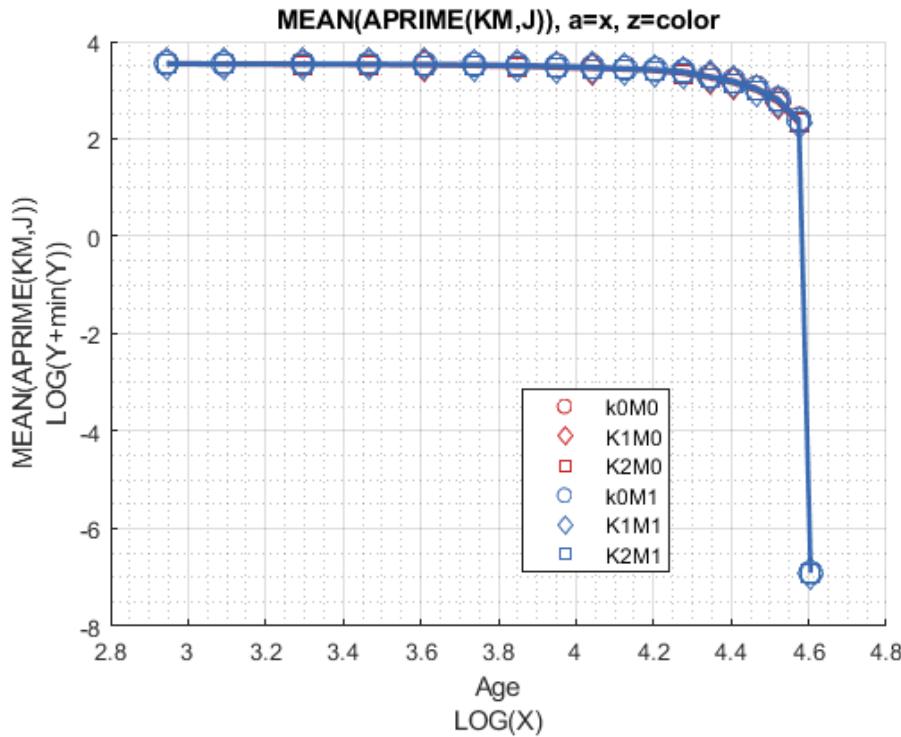




Graph Mean Savings Choices:

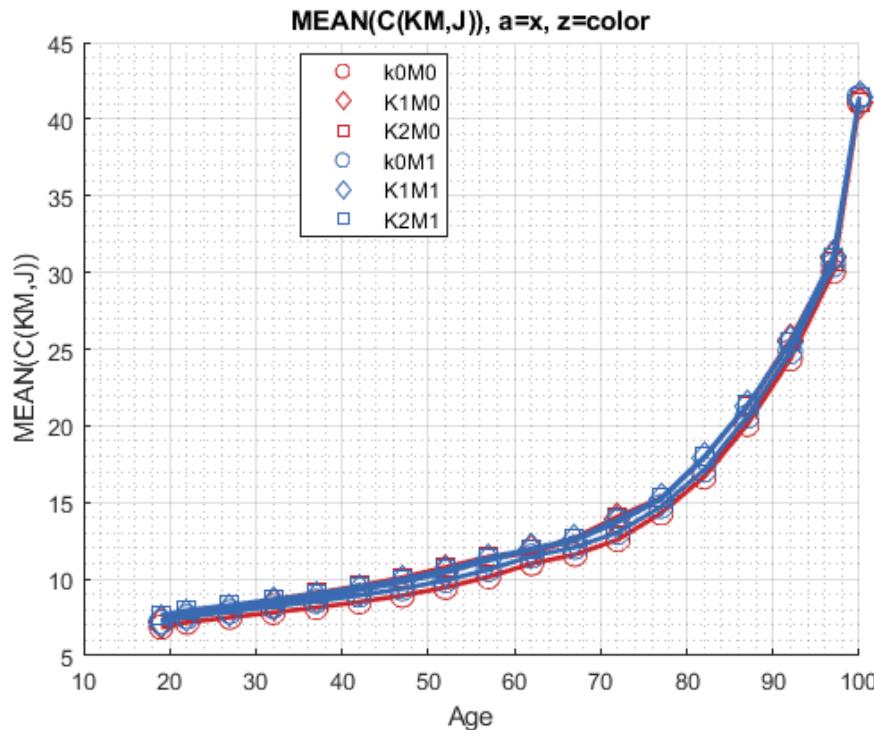
```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(KM,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(KM,J))'};
ff_graph_grid((tb_az_ap{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```

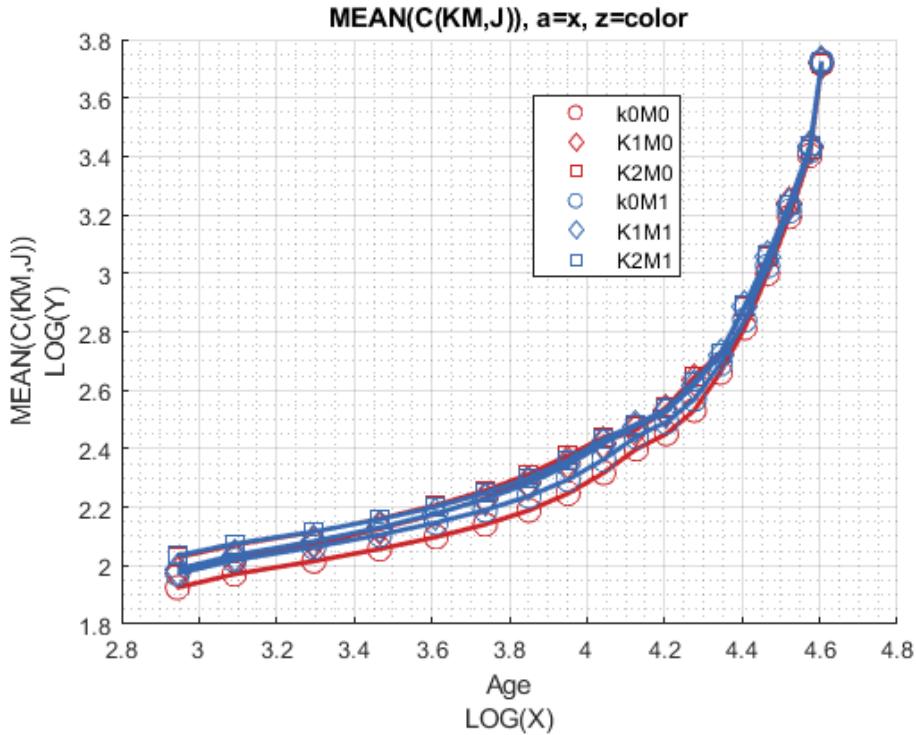




Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(KM,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(KM,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```





4.3.5 Analyze Education and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["E0M0", "E1M0", "E0M1", "E1M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {'*', 'p', '*', 'p'};
mp_support_graph('cl_colors') = {'red', 'red', 'blue', 'blue'};

MEAN(VAL(EKM,J)), MEAN(AP(EKM,J)), MEAN(C(EKM,J))

Tabulate value and policies:
```

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

xxx	MEAN(VAL(EKM,J))	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	group	edu	marry	mean_age_19	mean_age_22	mean_age_27	mean_age_32	mean_age_37
---	---	-----	---	---	---	-----	-----	-----	-----	-----
1	0	0		-0.19018		0.16944		0.46325		0.63924
2	1	0		0.64221		1.1027		1.4377		1.5815
3	0	1		0.85396		1.1146		1.3219		1.4469
4	1	1		1.5977		1.8806		2.0859		2.1737

```
% Aprime Choice
tb_az_ap = ff_summ_nd_array("MEAN(AP(EKM,J))", ap_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_p
```

```

xxx MEAN(AP(EKM,J)) xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
group   edu    marry   mean_age_19   mean_age_22   mean_age_27   mean_age_32   mean_age_37
-----  ---    -----  -----  -----  -----  -----  -----
1       0       0       34.68      34.441     34.268     34.044     33.748
2       1       0       34.463     34.238     34.218     34.116     33.895
3       0       1       34.723     34.511     34.368     34.173     33.909
4       1       1       34.468     34.274     34.287     34.218     34.029

% Consumption Choices
tb_az_c = ff_summ_nd_array("MEAN(C(EKM,J))", cons_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_p

xxx MEAN(C(EKM,J)) xxxxxxxxxxxxxxxxxxxxxxxx
group   edu    marry   mean_age_19   mean_age_22   mean_age_27   mean_age_32   mean_age_37
-----  ---    -----  -----  -----  -----  -----
1       0       0       7.1043     7.4114     7.7391     8.0887     8.4765
2       1       0       7.3218     7.7071     8.1025     8.492      8.8907
3       0       1       7.2329     7.5281     7.8428     8.1801     8.5525
4       1       1       7.5267     7.9028     8.2913     8.6732     9.0619

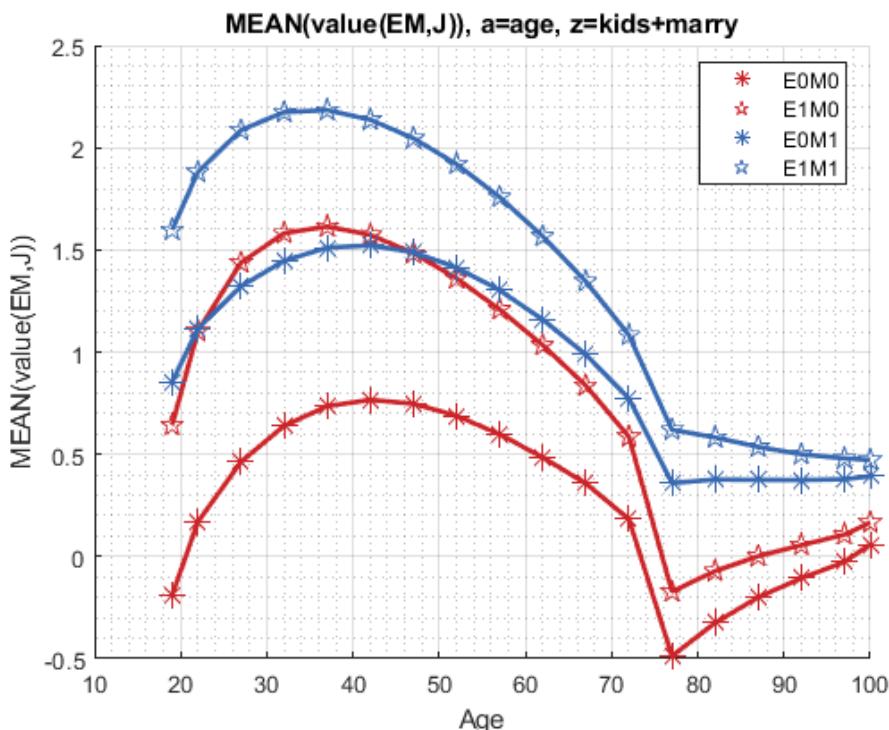
```

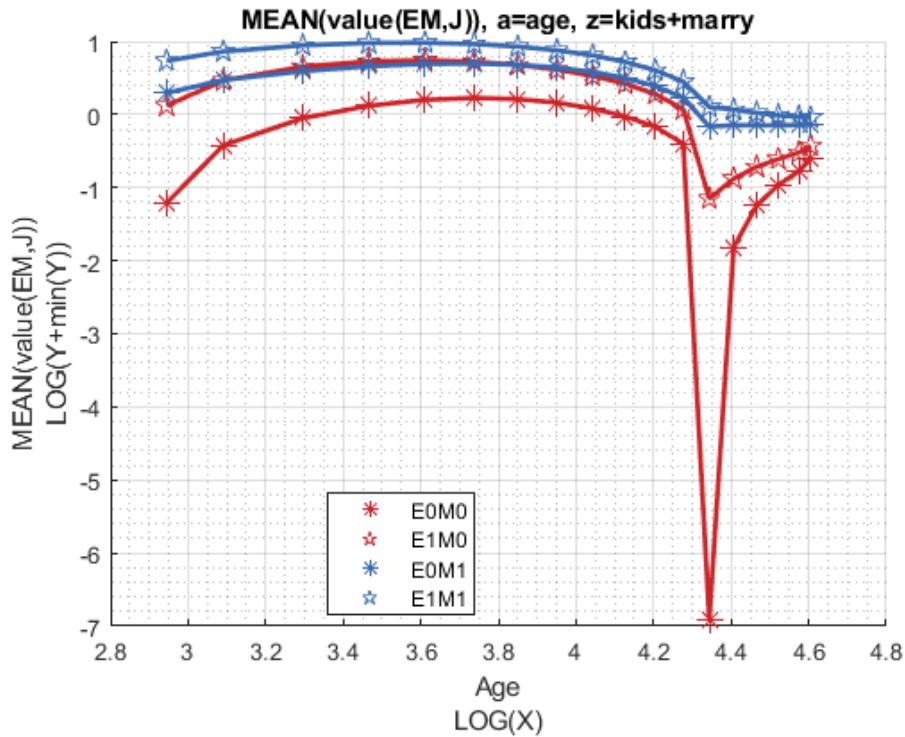
Graph Mean Values:

```

mp_support_graph('cl_st_graph_title') = {'MEAN(value(EM,J)), a=age, z=kids+marry'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(EM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);

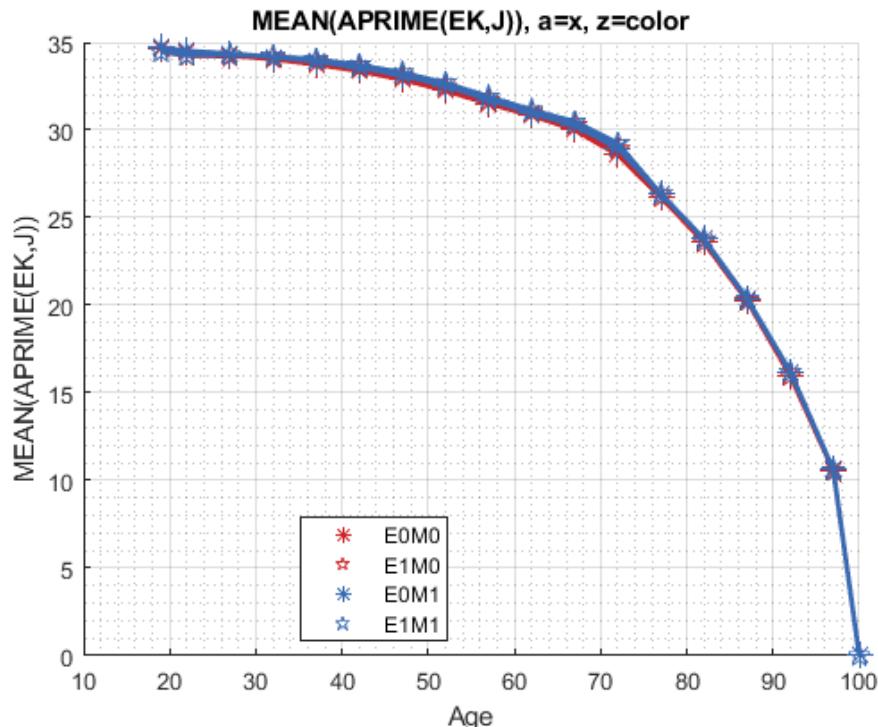
```

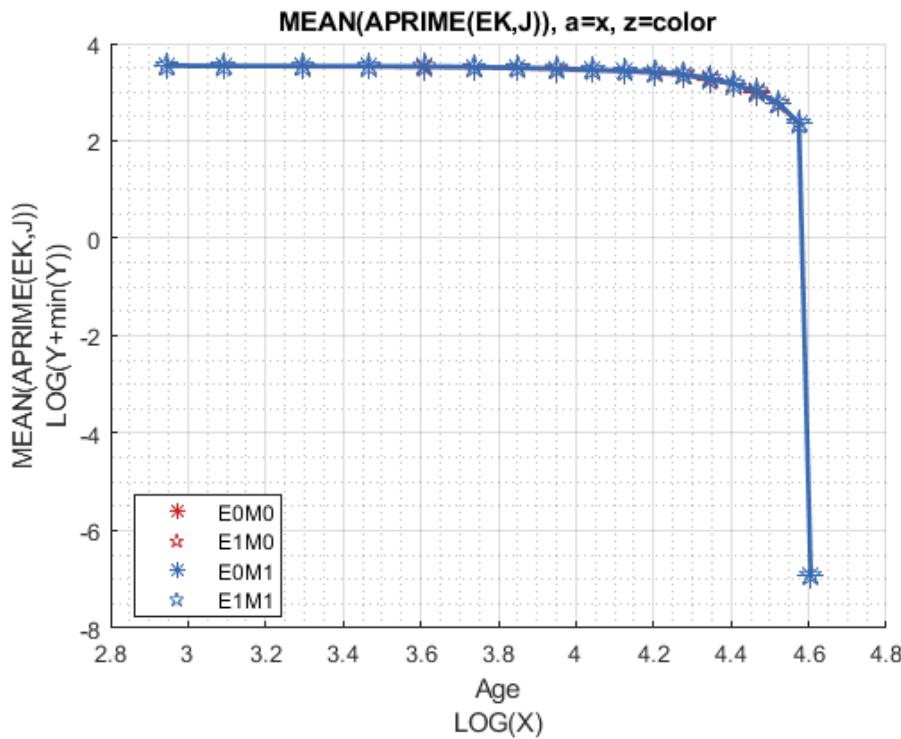




Graph Mean Savings Choices:

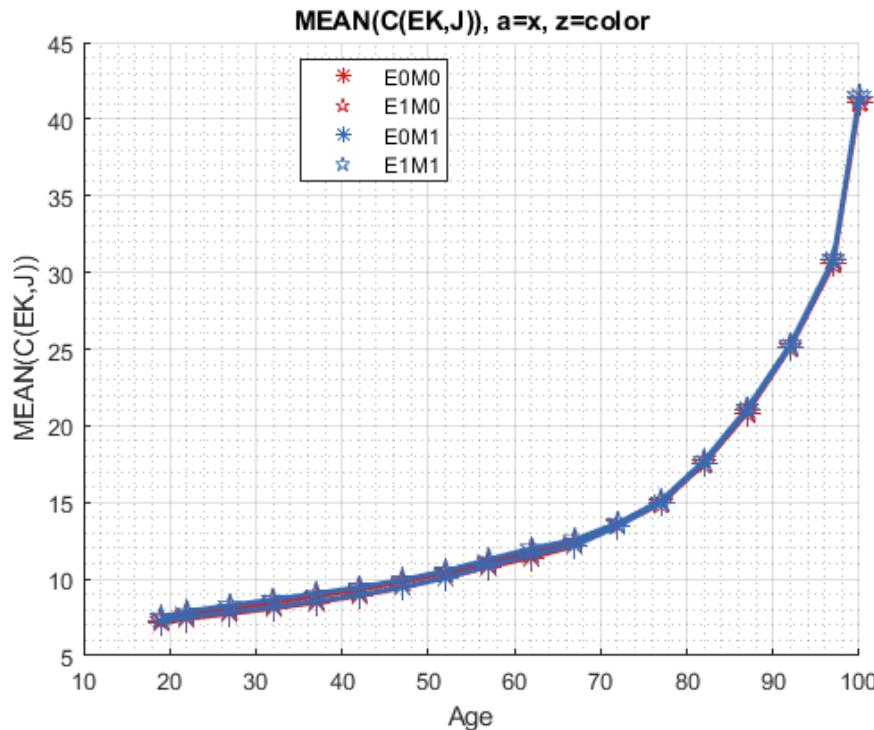
```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(EK,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(EK,J))'};
ff_graph_grid((tb_az_ap{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```

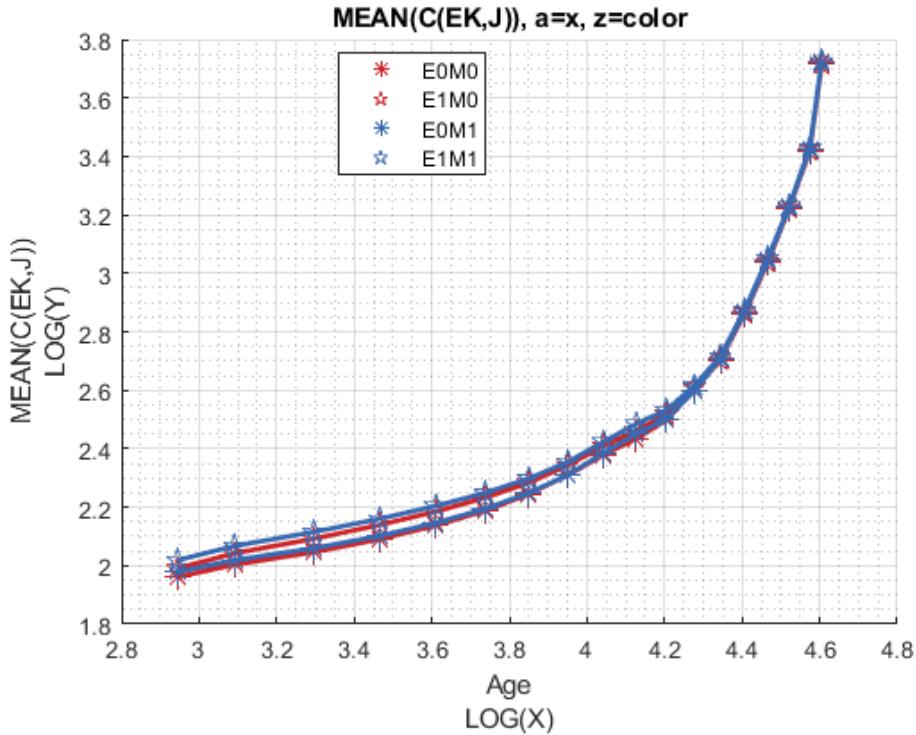




Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(EK,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(EK,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```





4.4 Small Test Exact Solution Spousal Shocks

This is the example vignette for function: `snw_vfi_main_bisec_vec` from the [PrjOptiSNW Package](#). This function solves for policy function with vectorized bisection. Small Solution Analysis, husband 5 shocks, wife 3 shocks.

4.4.1 Test SNW_VFI_MAIN Defaults Small

Call the function with default parameters.

```
mp_param = snw_mp_param('default_small153');
[V_VFI,ap_VFI,cons_VFI,mp_valpol_more] = snw_vfi_main_bisec_vec(mp_param);
```

```
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:18 of 17, time-this-age:0.058874
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:17 of 17, time-this-age:0.059673
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:16 of 17, time-this-age:0.059922
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:15 of 17, time-this-age:0.062184
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:14 of 17, time-this-age:0.057978
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:13 of 17, time-this-age:0.059396
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:12 of 17, time-this-age:0.061417
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:11 of 17, time-this-age:0.058985
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:10 of 17, time-this-age:0.05982
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:9 of 17, time-this-age:0.07028
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:8 of 17, time-this-age:0.06746
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:7 of 17, time-this-age:0.059236
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:6 of 17, time-this-age:0.069577
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:5 of 17, time-this-age:0.068207
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:4 of 17, time-this-age:0.060701
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:3 of 17, time-this-age:0.059637
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:2 of 17, time-this-age:0.062462
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:1 of 17, time-this-age:0.0598
Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_small153;SNW_MP_CONTROL=default_base;time=1.153
```

4.4.2 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=%3.2f'));
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});
```

4.4.3 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
mp_support_graph('it_legend_select') = 9; % how many shock legends to show
mp_support_graph('cl_colors') = 'jet';

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_permute);
```

group	savings	mean_eta_1	mean_eta_2	mean_eta_3	mean_eta_4	mean_eta_5	mean
1	0	-20.307	-10.285	-5.0527	-2.1772	-0.66137	-17
2	0.0097656	-19.558	-10.063	-4.9312	-2.0759	-0.5645	-
3	0.078125	-16.259	-8.8768	-4.2806	-1.5512	-0.069704	-14
4	0.26367	-12.127	-7.062	-3.289	-0.8157	0.59457	-11
5	0.625	-8.3166	-5.145	-2.2609	-0.1528	1.1414	-7.
6	1.2207	-5.2004	-3.3395	-1.2735	0.417	1.5609	-4.
7	2.1094	-2.8448	-1.7849	-0.39262	0.91448	1.8837	-2.
8	3.3496	-1.1351	-0.53317	0.368	1.3497	2.1394	-1.
9	5	0.088433	0.43451	1.0071	1.7212	2.3505	0.
10	7.1191	0.96365	1.1669	1.5292	2.0348	2.5311	0.9
11	9.7656	1.5949	1.7173	1.948	2.3007	2.6878	1.
12	12.998	2.0558	2.1316	2.2803	2.5253	2.8229	2.

13	16.875	2.397	2.4453	2.5427	2.7131	2.939	2.
14	21.455	2.6533	2.6848	2.7497	2.869	3.0391	2.
15	26.797	2.8488	2.8698	2.9139	2.998	3.1258	2.
16	32.959	2.9999	3.0142	3.0447	3.1047	3.2007	3.
17	40	3.1182	3.1282	3.1496	3.1929	3.2653	3.
18	47.979	3.2119	3.219	3.2343	3.2659	3.3208	3.
19	56.953	3.287	3.2921	3.3033	3.3266	3.3685	3.
20	66.982	3.3477	3.3515	3.3597	3.3772	3.4093	3.
21	78.125	3.3974	3.4002	3.4064	3.4196	3.4444	3.
22	90.439	3.4384	3.4405	3.4452	3.4553	3.4746	3.
23	103.98	3.4724	3.4741	3.4776	3.4854	3.5006	3.
24	118.82	3.501	3.5022	3.505	3.5111	3.5231	3.
25	135	3.5251	3.5261	3.5282	3.533	3.5426	3.

% Aprime Choice

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

xxx MEAN(AP(A,Z)) xxxxxxxxxxxxxxxxxxxxxxxxx

group	savings	mean_eta_1	mean_eta_2	mean_eta_3	mean_eta_4	mean_eta_5	mean
1	0	2.7511e-05	0.0015443	0.029727	0.16652	0.75086	0.00
2	0.0097656	0.00054711	0.0027834	0.031634	0.16984	0.75642	0.00
3	0.078125	0.015731	0.018652	0.049638	0.19667	0.79532	0.00
4	0.26367	0.093357	0.0908	0.12387	0.2854	0.9063	0.
5	0.625	0.31381	0.31997	0.35088	0.51766	1.1457	0.
6	1.2207	0.74541	0.7447	0.78537	0.95128	1.5671	0.
7	2.1094	1.4161	1.4196	1.4616	1.6183	2.2194	1.
8	3.3496	2.3637	2.3696	2.4109	2.5645	3.1433	2.
9	5	3.6292	3.6363	3.678	3.8404	4.3795	3.
10	7.1191	5.2766	5.2846	5.326	5.4907	5.9774	5.
11	9.7656	7.3022	7.3101	7.3505	7.5158	7.9941	7.
12	12.998	9.7443	9.7504	9.7888	9.9552	10.482	
13	16.875	12.756	12.762	12.797	12.958	13.553	1.
14	21.455	16.326	16.33	16.365	16.512	17.127	1.
15	26.797	20.39	20.392	20.419	20.557	21.172	2.
16	32.959	25.075	25.082	25.112	25.235	25.829	2.
17	40	30.452	30.46	30.499	30.623	31.182	
18	47.979	36.549	36.557	36.599	36.745	37.265	3.
19	56.953	43.56	43.567	43.602	43.748	44.27	4.
20	66.982	51.366	51.375	51.418	51.556	52.091	5.
21	78.125	59.653	59.661	59.707	59.864	60.396	5.
22	90.439	69.009	69.015	69.057	69.216	69.764	6.
23	103.98	79.499	79.505	79.547	79.696	80.26	7.
24	118.82	90.869	90.876	90.918	91.063	91.614	9.
25	135	103.22	103.22	103.26	103.41	103.95	

% Consumption Choices

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

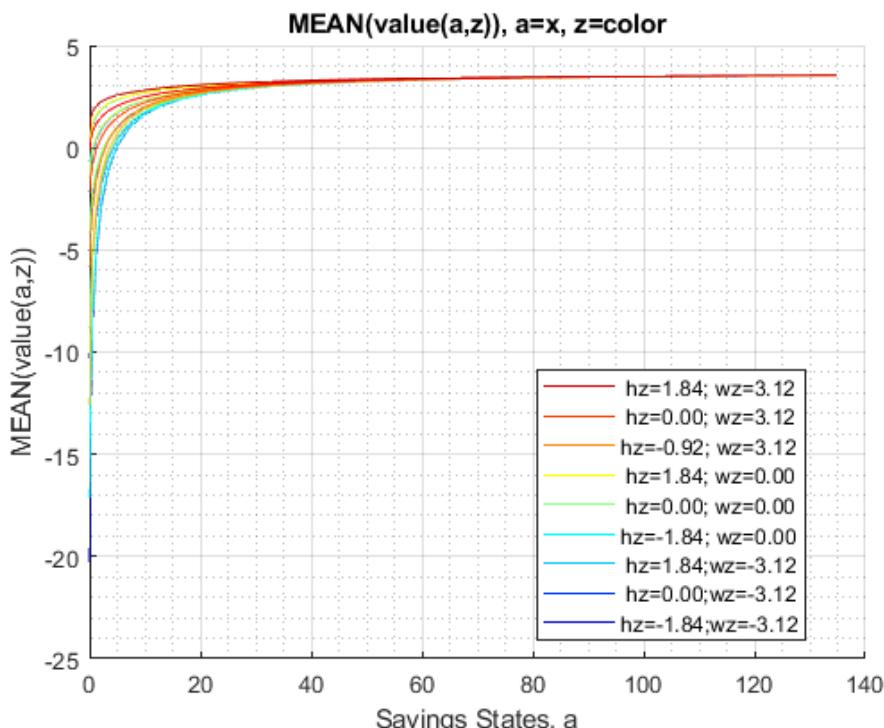
xxx MEAN(C(A,Z)) xxxxxxxxxxxxxxxxxxxxxxxxx

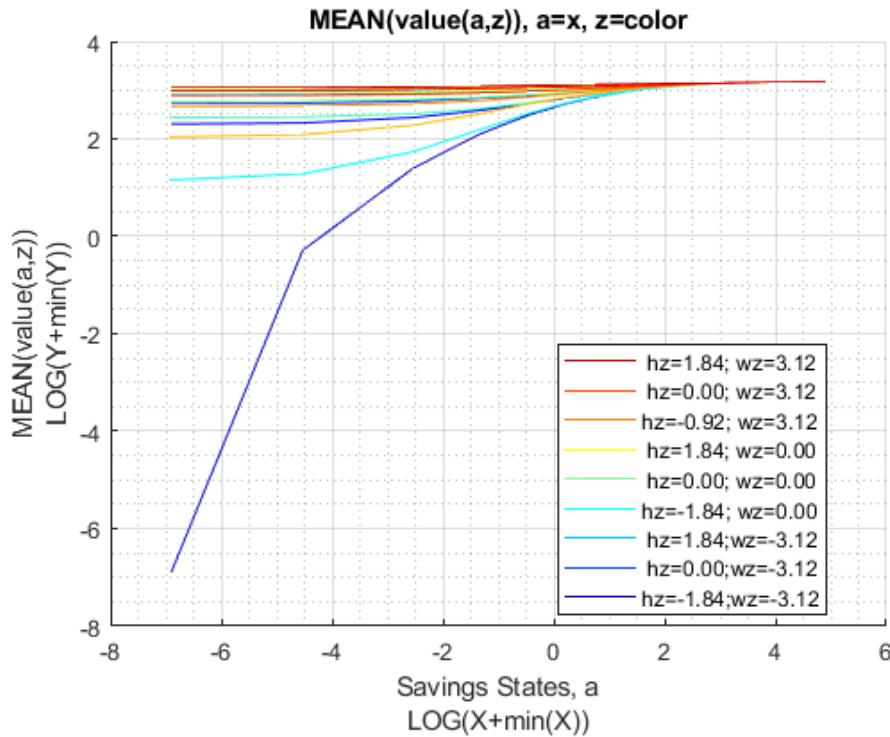
group	savings	mean_eta_1	mean_eta_2	mean_eta_3	mean_eta_4	mean_eta_5	mean
1	0	0.17596	0.2993	0.5664	1.1423	2.3148	0.3
2	0.0097656	0.18702	0.30957	0.57594	1.1504	2.3206	0.3
3	0.078125	0.25285	0.37423	0.63806	1.2035	2.3616	0.3
4	0.26367	0.39479	0.52047	0.78122	1.3316	2.4672	0.4

5	0.625	0.60083	0.71594	0.97718	1.5213	2.6493	0.6
6	1.2207	0.87005	0.9899	1.2392	1.7827	2.9223	0.9
7	2.1094	1.2414	1.355	1.6005	2.1515	3.305	1.
8	3.3496	1.7441	1.8535	2.0975	2.6496	3.8243	1.
9	5	2.4043	2.5112	2.7528	3.2942	4.5075	2.
10	7.1191	3.2256	3.3306	3.571	4.1085	5.3729	3.
11	9.7656	4.2795	4.3841	4.6243	5.1596	6.4314	4.
12	12.998	5.596	5.7019	5.9433	6.4763	7.6989	5.
13	16.875	7.0899	7.1954	7.4401	7.977	9.1305	7.
14	21.455	8.8406	8.9481	9.1919	9.7431	10.875	8.
15	26.797	10.982	11.091	11.342	11.901	13.033	11.
16	32.959	13.452	13.557	13.805	14.378	15.531	13.
17	40	16.251	16.354	16.593	17.165	18.352	16.
18	47.979	19.418	19.521	19.756	20.307	21.532	19.
19	56.953	22.826	22.93	23.172	23.723	24.945	22.
20	66.982	26.663	26.765	27	27.557	28.767	26.
21	78.125	31.312	31.414	31.646	32.184	33.397	3.
22	90.439	36.251	36.355	36.591	37.128	38.324	36.
23	103.98	41.485	41.589	41.825	42.372	43.552	41.
24	118.82	47.334	47.438	47.674	48.224	49.417	4.
25	135	53.768	53.874	54.112	54.659	55.86	

Graph Mean Values:

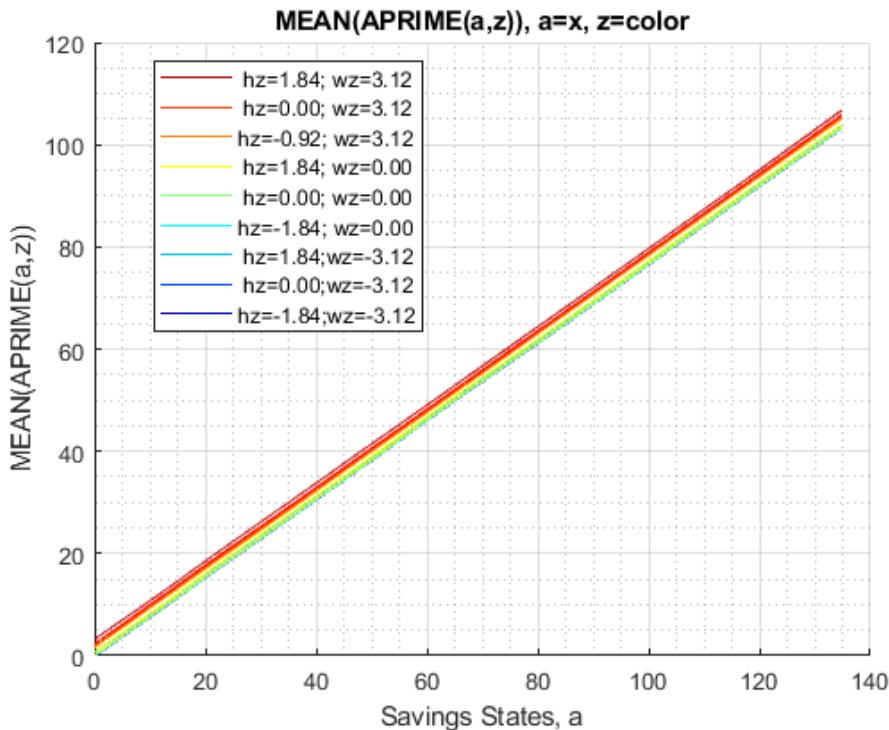
```
mp_support_graph('cl_st_graph_title') = {'MEAN(value(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(a,z))'};
ff_graph_grid((tb_az_v{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);
```

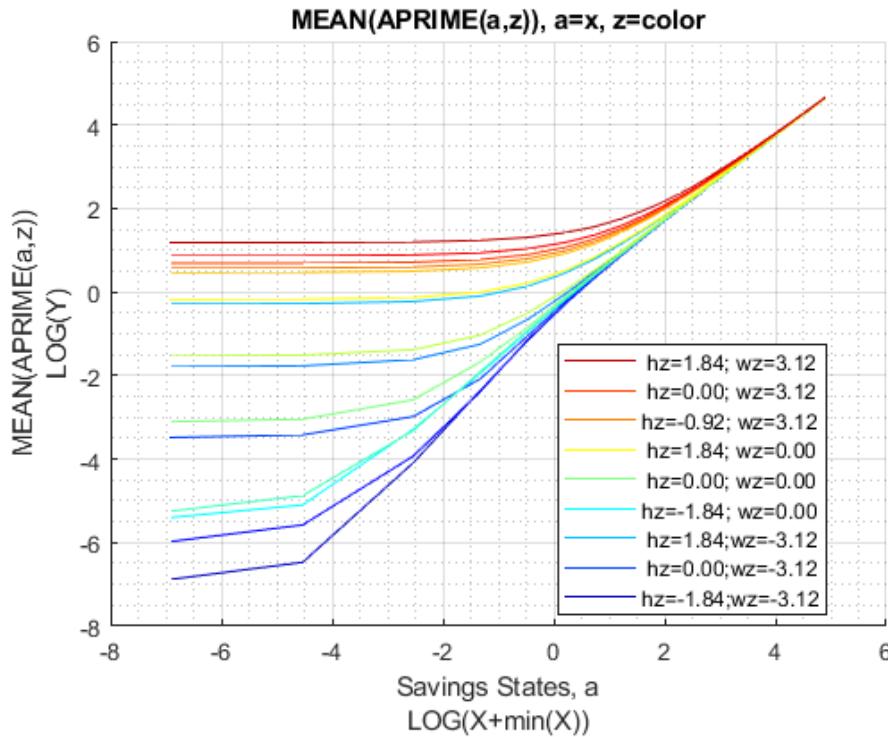




Graph Mean Savings Choices:

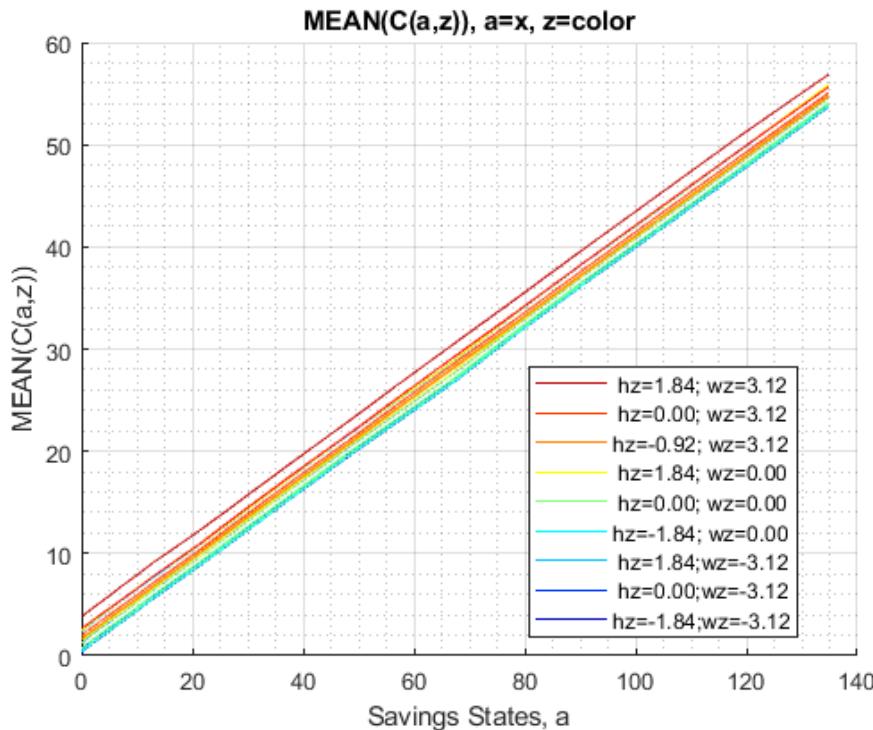
```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(a,z))'};
ff_graph_grid((tb_az_ap{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```

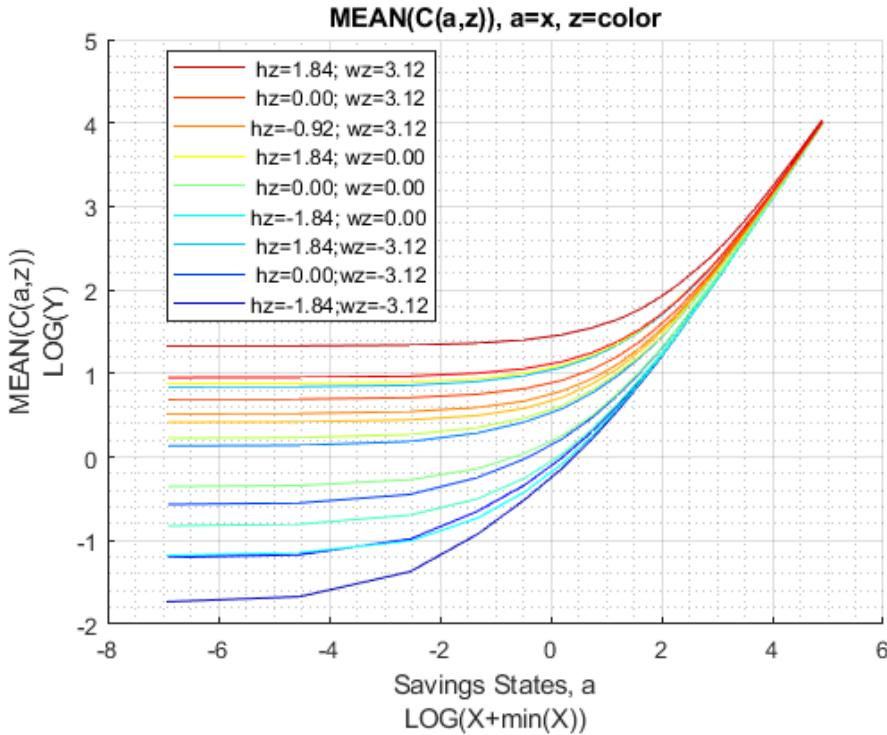




Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(a,z)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(a,z))'};
ff_graph_grid((tb_az_c{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```





4.4.4 Analyze Kids and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["k0M0", "K1M0", "K2M0", "k0M1", "K1M1", "K2M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = { 'o', 'd', 's', 'o', 'd', 's' };
mp_support_graph('cl_colors') = {'red', 'red', 'red', 'blue', 'blue', 'blue'};

MEAN(VAL(KM,J)), MEAN(AP(KM,J)), MEAN(C(KM,J))

Tabulate value and policies:
```

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

xxx	MEAN(VAL(KM,J))	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx						
group	kids	marry	mean_age_19	mean_age_22	mean_age_27	mean_age_32	mean_age_37	
---	---	---	-----	-----	-----	-----	-----	-----
1	1	0	1.4699	1.7485	1.9344	1.9907	1.9652	
2	2	0	-0.020723	0.46111	0.83504	1.0389	1.1397	
3	3	0	-0.77111	-0.30145	0.081934	0.30157	0.41928	
4	1	1	2.7247	2.8812	2.9832	2.9923	2.9362	
5	2	1	1.8762	2.1212	2.3182	2.4103	2.4302	
6	3	1	1.4732	1.7023	1.8951	1.9893	2.0142	

```
% Aprime Choice
```

```

tb_az_ap = ff_summ_nd_array("MEAN(AP(KM,J))", ap_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_pe)

xxx MEAN(AP(KM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry mean_age_19 mean_age_22 mean_age_27 mean_age_32 mean_age_36
----- ----- ----- ----- ----- ----- ----- -----
1 1 0 34.929 34.724 34.662 34.55 34.357
2 2 0 34.6 34.331 34.195 33.99 33.687
3 3 0 34.185 33.965 33.873 33.7 33.421
4 1 1 35.711 35.608 35.696 35.722 35.654
5 2 1 35.365 35.243 35.28 35.238 35.095
6 3 1 34.9 34.807 34.856 34.829 34.694

% Consumption Choices
tb_az_c = ff_summ_nd_array("MEAN(C(KM,J))", cons_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_pe)

xxx MEAN(C(KM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry mean_age_19 mean_age_22 mean_age_27 mean_age_32 mean_age_36
----- ----- ----- ----- ----- ----- -----
1 1 0 6.8551 7.1756 7.502 7.8205 8.1483
2 2 0 7.1843 7.5683 7.9695 8.3802 8.8184
3 3 0 7.5997 7.934 8.2911 8.6703 9.0841
4 1 1 7.8017 8.1851 8.5662 8.9367 9.3167
5 2 1 7.8815 8.2584 8.6593 9.0691 9.4965
6 3 1 8.1632 8.4941 8.8603 9.2345 9.6357

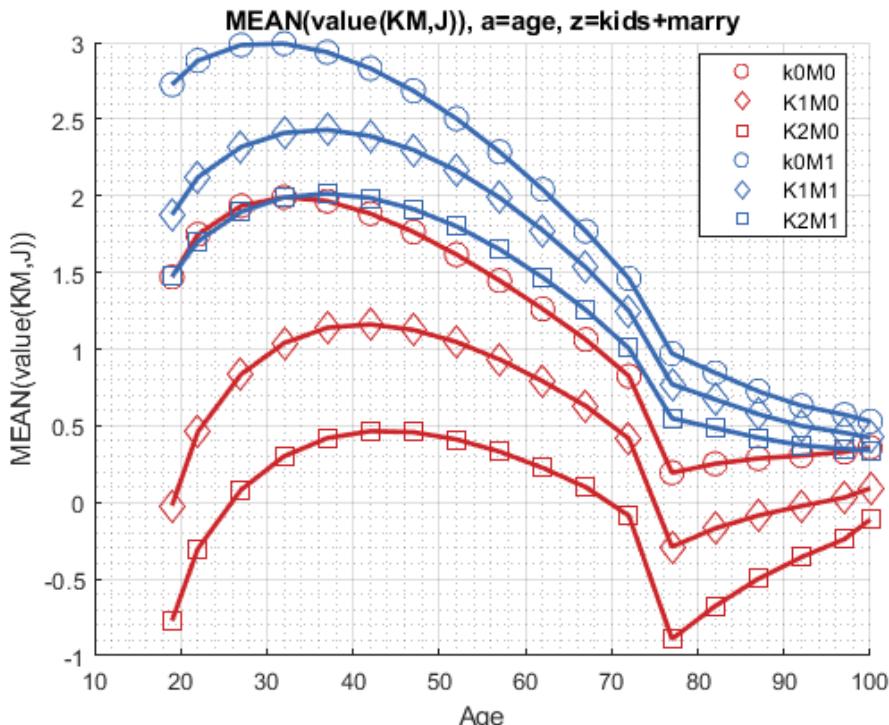
```

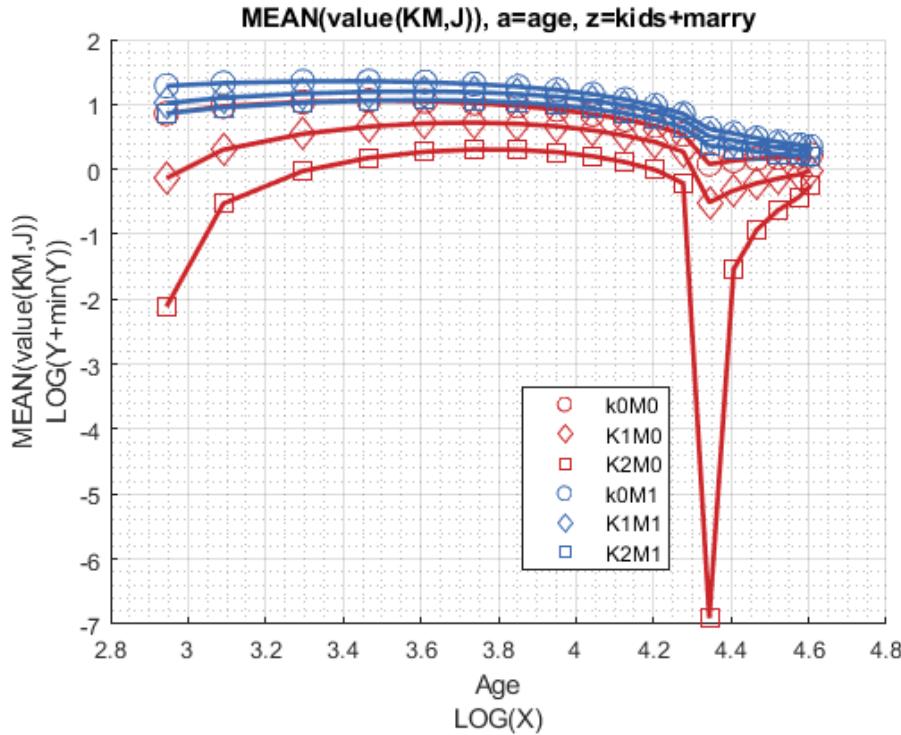
Graph Mean Values:

```

mp_support_graph('cl_st_graph_title') = {'MEAN(value(KM,J)), a=age, z=kids+marry'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(KM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);

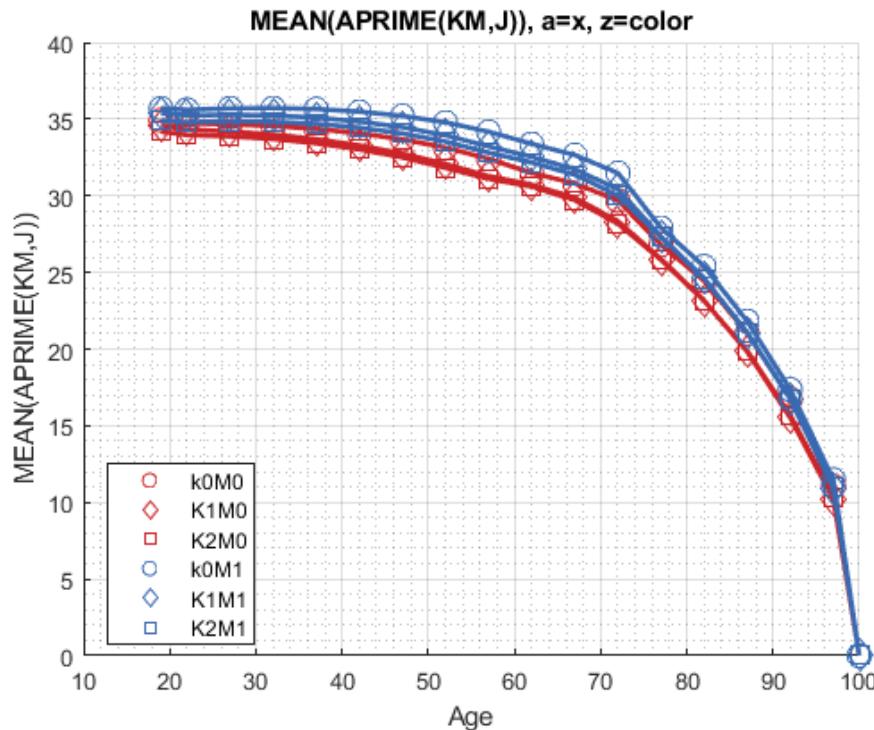
```

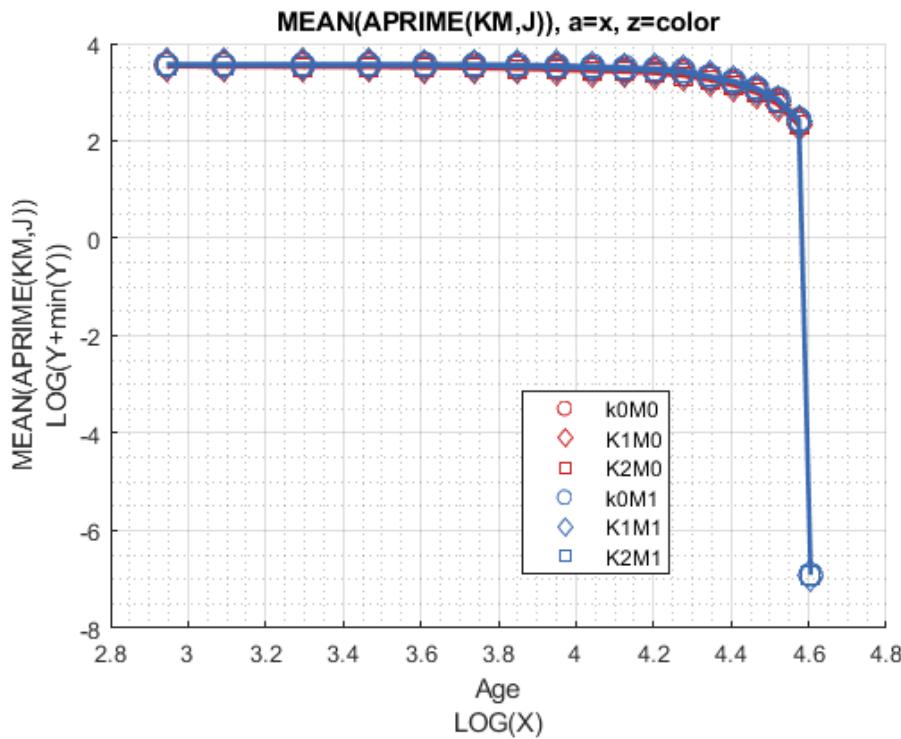




Graph Mean Savings Choices:

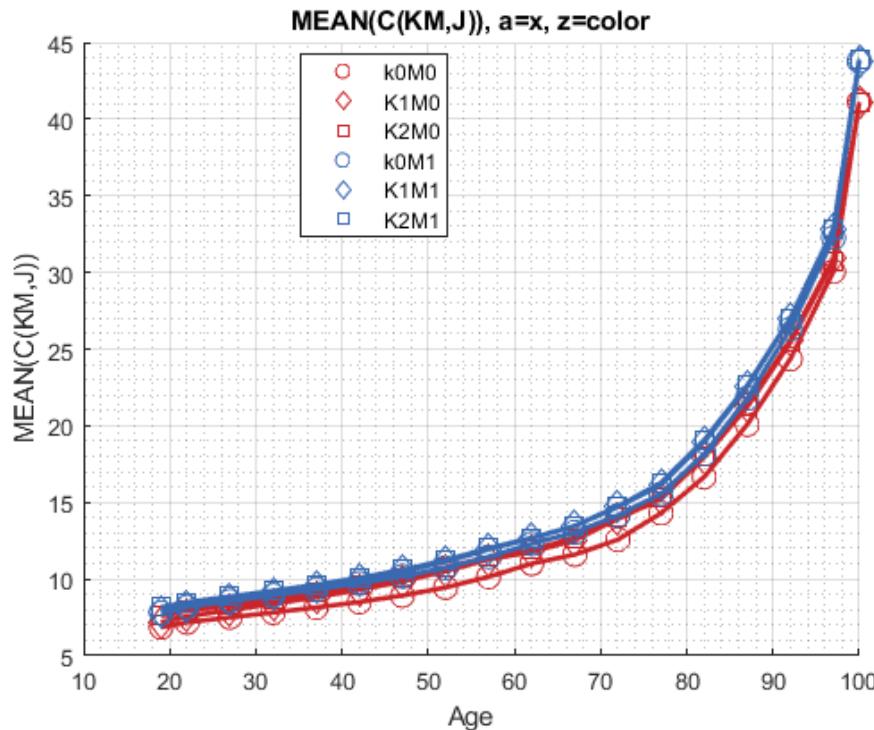
```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(KM,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(KM,J))'};
ff_graph_grid((tb_az_ap{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```

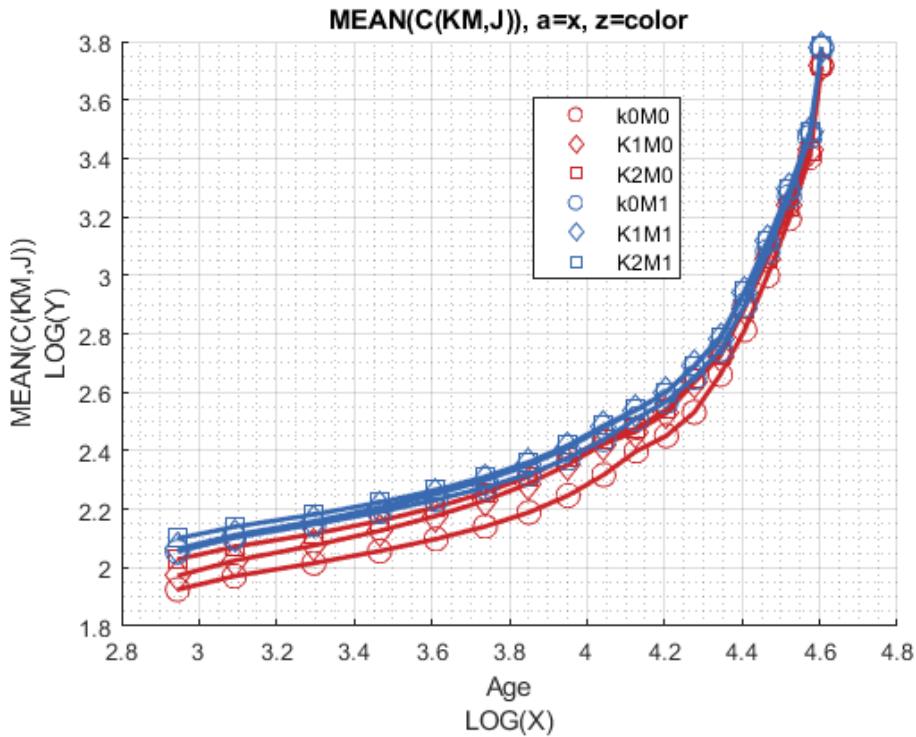




Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(KM,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(KM,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```





4.4.5 Analyze Education and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["E0M0", "E1M0", "E0M1", "E1M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {'*', 'p', '*', 'p'};
mp_support_graph('cl_colors') = {'red', 'red', 'blue', 'blue'};

MEAN(VAL(EKM,J)), MEAN(AP(EKM,J)), MEAN(C(EKM,J))

Tabulate value and policies:
```

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

xxx	MEAN(VAL(EKM,J))	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	group	edu	marry	mean_age_19	mean_age_22	mean_age_27	mean_age_32	mean_age_37
---	---	-----	---	---	---	-----	-----	-----	-----	-----
1	0	0		-0.19018		0.16944		0.46325		0.63924
2	1	0		0.64221		1.1027		1.4377		1.5815
3	0	1		1.7253		1.9221		2.0804		2.1589
4	1	1		2.3242		2.5478		2.7173		2.7432

```
% Aprime Choice
tb_az_ap = ff_summ_nd_array("MEAN(AP(EKM,J))", ap_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_p
```

```

xxx MEAN(AP(EKM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx
group   edu    marry   mean_age_19   mean_age_22   mean_age_27   mean_age_32   mean_age_37
-----  ---    -----  -----  -----  -----  -----  -----
1       0       0       34.68      34.441     34.268     34.044     33.748
2       1       0       34.463     34.238     34.218     34.116     33.895
3       0       1       35.361     35.231     35.189     35.094     34.928
4       1       1       35.29      35.207     35.366     35.432     35.368

% Consumption Choices
tb_az_c = ff_summ_nd_array("MEAN(C(EKM,J))", cons_VFI, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_p

xxx MEAN(C(EKM,J)) xxxxxxxxxxxxxxxxxxxxxxxxx
group   edu    marry   mean_age_19   mean_age_22   mean_age_27   mean_age_32   mean_age_37
-----  ---    -----  -----  -----  -----  -----
1       0       0       7.1043     7.4114     7.7391     8.0887     8.4765
2       1       0       7.3218     7.7071     8.1025     8.492      8.8907
3       0       1       7.761      8.0772     8.4119     8.7662     9.1545
4       1       1       8.1365     8.5478     8.9787     9.394      9.8115

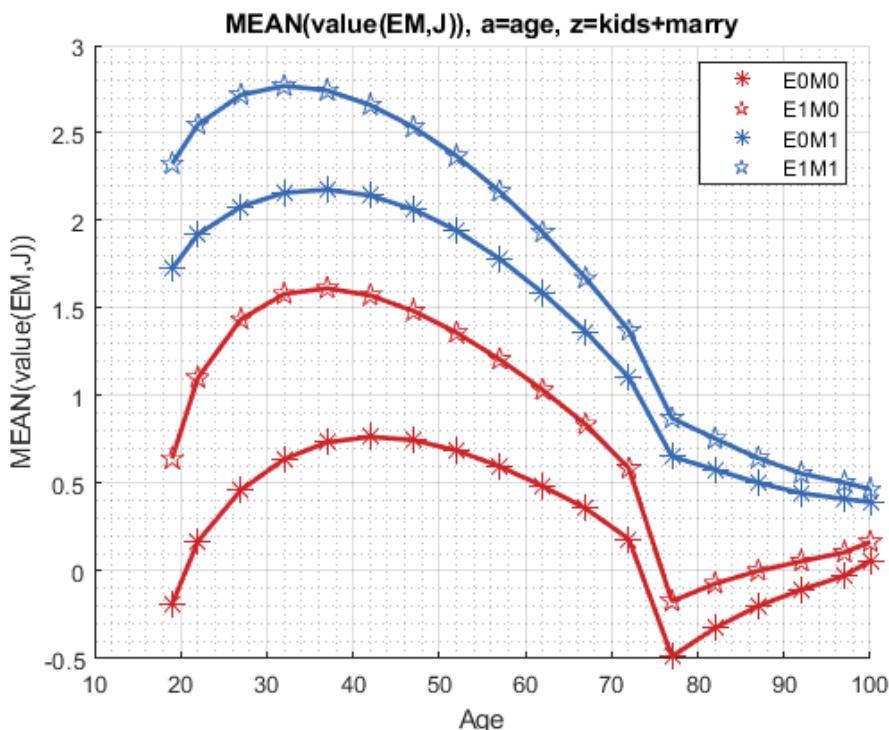
```

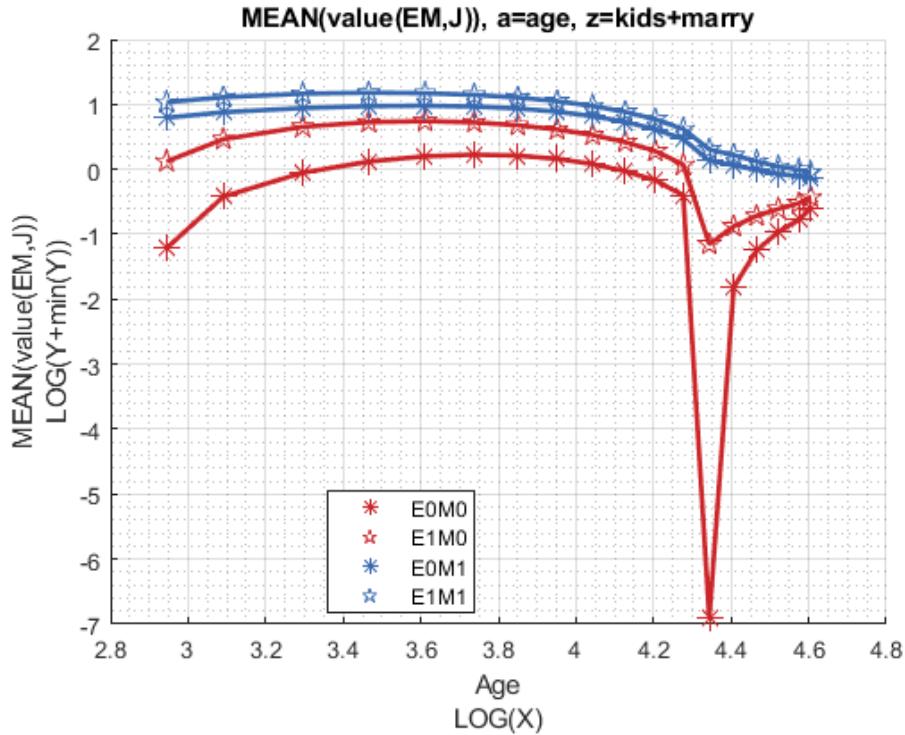
Graph Mean Values:

```

mp_support_graph('cl_st_graph_title') = {'MEAN(value(EM,J)), a=age, z=kids+marry'};
mp_support_graph('cl_st_ytitle') = {'MEAN(value(EM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);

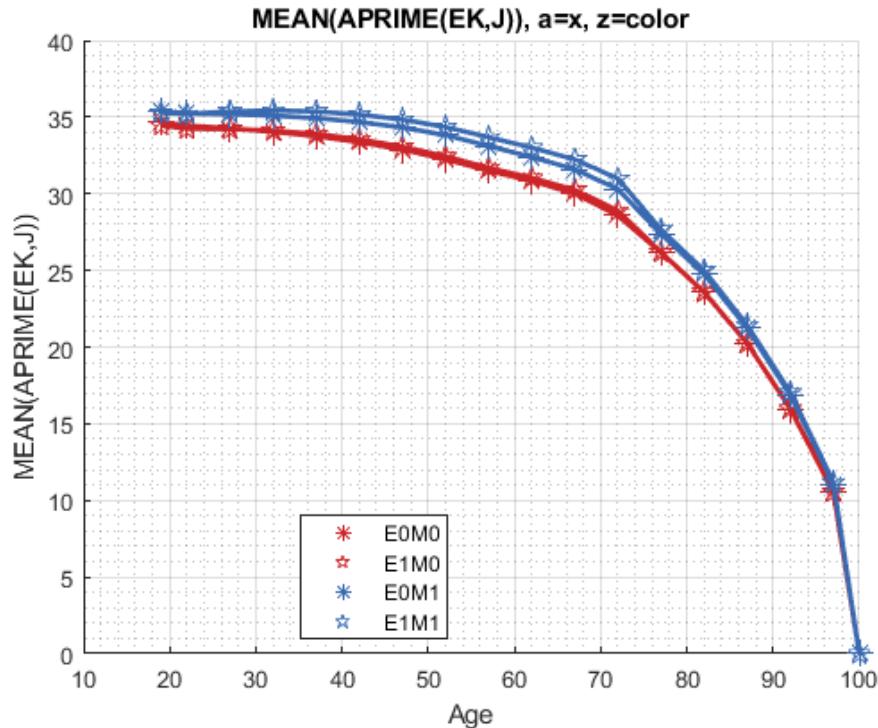
```

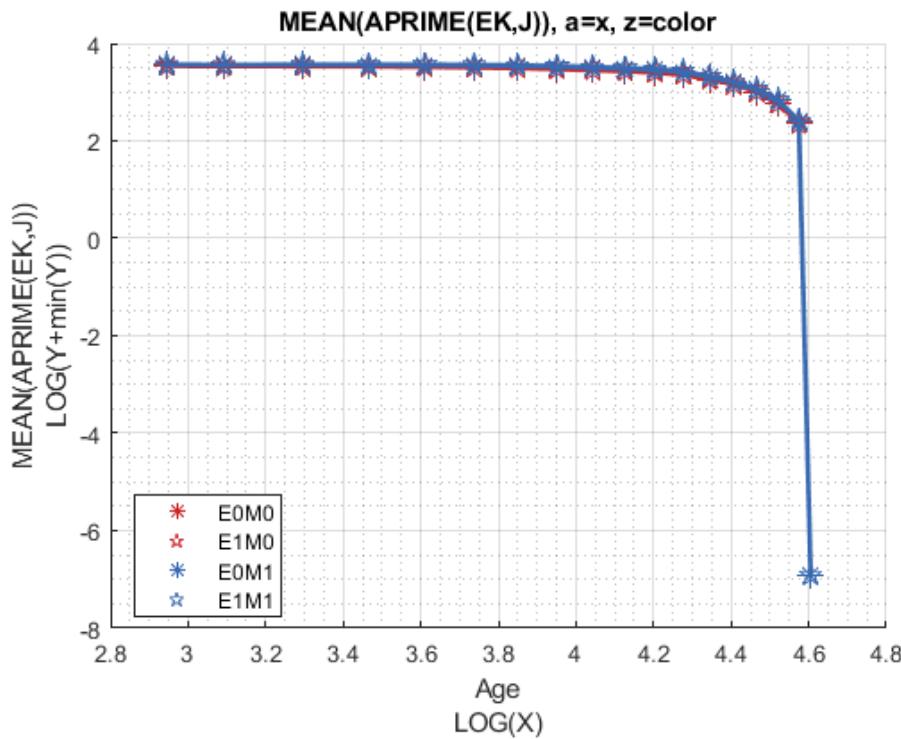




Graph Mean Savings Choices:

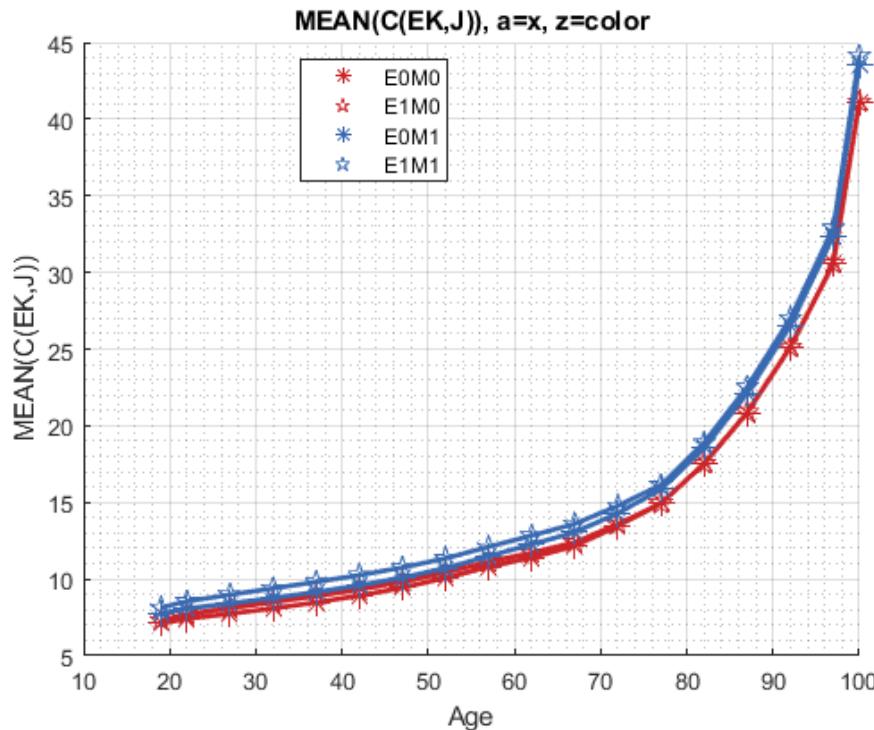
```
mp_support_graph('cl_st_graph_title') = {'MEAN(APRIME(EK,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(APRIME(EK,J))'};
ff_graph_grid((tb_az_ap{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```

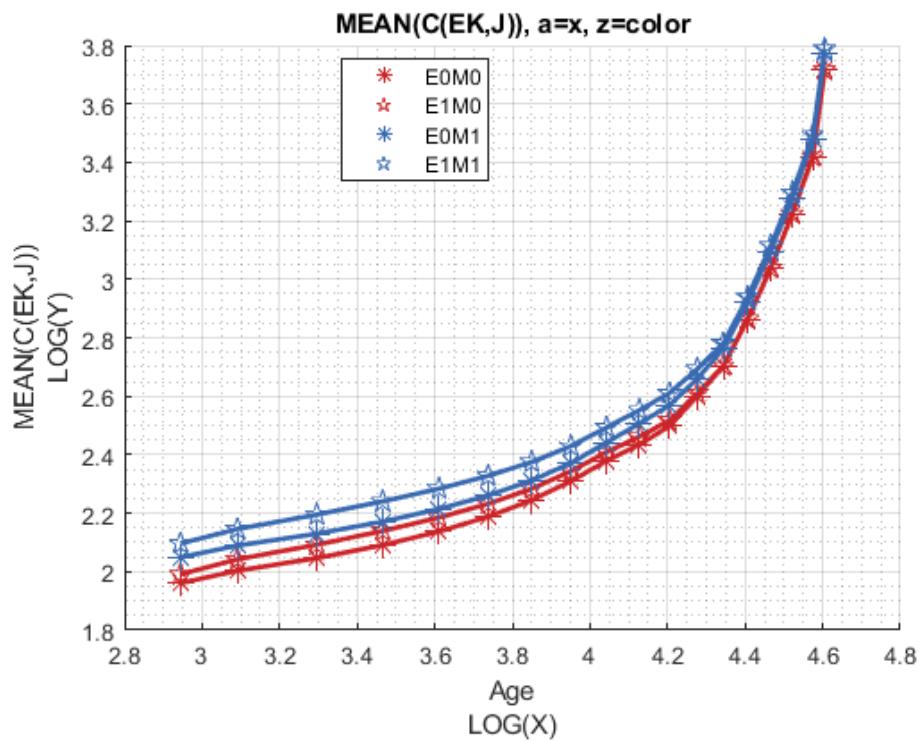




Graph Mean Consumption:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(EK,J)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(EK,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```





Chapter 5

Solution with Unemployment

5.1 Life Cycle Dynamic Programming under Unemployment Shock

This is the example vignette for function: `snw_vfi_main_bisec_vec` from the [PrjOptiSNW Package](#). This function solves for policy function using Exact Vectorized Solution. Value in 2020 with surprise COVID unemployment Shock, with non-covid year Value as the continuation function. The file focuses on the change in value function, asset choice, and consumption choice given a one period unemployment shock (that does not reappear in the future again).

5.1.1 Test SNW_VFI_UNEMP

Solve the Regular Value and Also the Unemployment Value.

First, solve for value without unemployment issue (use the vectorized code that was previously tested):

```
mp_params = snw_mp_param('default_docdense');
mp_controls = snw_mp_control('default_test');
[V_VFI_ss,ap_VFI_ss,cons_VFI_ss,mp_valpol_more_ss] = ...
    snw_vfi_main_bisec_vec(mp_params, mp_controls);

SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:83 of 82, time-this-age:7.0273
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:82 of 82, time-this-age:5.7228
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:81 of 82, time-this-age:5.7083
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:80 of 82, time-this-age:5.7427
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:79 of 82, time-this-age:5.6962
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:78 of 82, time-this-age:5.7223
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:77 of 82, time-this-age:5.7035
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:76 of 82, time-this-age:5.8126
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:75 of 82, time-this-age:5.7166
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:74 of 82, time-this-age:5.6915
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:73 of 82, time-this-age:5.6971
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:72 of 82, time-this-age:5.7128
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:71 of 82, time-this-age:5.6819
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:70 of 82, time-this-age:5.7212
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:69 of 82, time-this-age:5.6814
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:68 of 82, time-this-age:5.6867
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:67 of 82, time-this-age:5.6787
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:66 of 82, time-this-age:5.7367
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:65 of 82, time-this-age:5.7251
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:64 of 82, time-this-age:5.7155
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:63 of 82, time-this-age:5.6715
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:62 of 82, time-this-age:6.3362
```

SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:61 of 82, time-this-age:5.6611
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:60 of 82, time-this-age:5.6816
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:59 of 82, time-this-age:5.6858
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:58 of 82, time-this-age:5.6664
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:57 of 82, time-this-age:5.7059
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:56 of 82, time-this-age:5.6912
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:55 of 82, time-this-age:5.7074
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:54 of 82, time-this-age:5.7189
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:53 of 82, time-this-age:5.8154
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:52 of 82, time-this-age:6.2071
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:51 of 82, time-this-age:5.7038
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:50 of 82, time-this-age:5.7383
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:49 of 82, time-this-age:5.7201
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:48 of 82, time-this-age:5.7148
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:47 of 82, time-this-age:5.8791
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:46 of 82, time-this-age:5.8699
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:45 of 82, time-this-age:5.9276
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:44 of 82, time-this-age:5.8993
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:43 of 82, time-this-age:5.9322
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:42 of 82, time-this-age:5.8956
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:41 of 82, time-this-age:5.892
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:40 of 82, time-this-age:5.941
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:39 of 82, time-this-age:5.9249
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:38 of 82, time-this-age:5.9262
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:37 of 82, time-this-age:5.9554
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:36 of 82, time-this-age:5.9082
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:35 of 82, time-this-age:5.8864
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:34 of 82, time-this-age:5.846
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:33 of 82, time-this-age:5.9126
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:32 of 82, time-this-age:5.918
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:31 of 82, time-this-age:5.873
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:30 of 82, time-this-age:5.9253
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:29 of 82, time-this-age:5.8694
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:28 of 82, time-this-age:5.9057
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:27 of 82, time-this-age:5.9302
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:26 of 82, time-this-age:5.9329
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:25 of 82, time-this-age:6.1905
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:24 of 82, time-this-age:5.9237
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:23 of 82, time-this-age:5.9634
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:22 of 82, time-this-age:5.8632
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:21 of 82, time-this-age:6.2308
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:20 of 82, time-this-age:5.9032
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:19 of 82, time-this-age:5.916
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:18 of 82, time-this-age:5.9014
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:17 of 82, time-this-age:5.852
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:16 of 82, time-this-age:5.8767
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:15 of 82, time-this-age:5.9424
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:14 of 82, time-this-age:5.8916
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:13 of 82, time-this-age:5.9423
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:12 of 82, time-this-age:5.8851
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:11 of 82, time-this-age:5.8739
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:10 of 82, time-this-age:5.8917
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:9 of 82, time-this-age:5.9436
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:8 of 82, time-this-age:5.9345
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:7 of 82, time-this-age:5.9111
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:6 of 82, time-this-age:5.9251
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:5 of 82, time-this-age:5.9407
SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:4 of 82, time-this-age:5.9242

SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:3 of 82, time-this-age:5.9242

SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:2 of 82, time-this-age:5.8728

SNW_VFI_MAIN_BISEC_VEC: Finished Age Group:1 of 82, time-this-age:5.9204

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=486.

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

CONTAINER NAME: mp_outcomes ND Array (Matrix etc)

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

	i	idx	ndim	numel	rowN	colN	sum	mean	std
	-	---	----	-----	---	-----	-----	-----	-----
V_VFI	1	1	6	4.37e+07	83	5.265e+05	-1.5339e+08	-3.5101	26.11
ap_VFI	2	2	6	4.37e+07	83	5.265e+05	1.4159e+09	32.402	36.79
cons_VFI	3	3	6	4.37e+07	83	5.265e+05	2.1402e+08	4.8975	8.329

xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c
	-----	-----	-----	-----	-----	-----	-----	-----	-----
r1	-346.51	-346.12	-343.63	-337.86	-328.51	21.702	21.852	22.003	
r2	-334.38	-333.99	-331.51	-325.83	-316.83	21.724	21.869	22.015	
r3	-322.45	-322.06	-319.6	-314.14	-305.6	21.745	21.885	22.027	
r4	-310.63	-310.27	-307.99	-302.88	-294.87	21.767	21.903	22.041	
r5	-299.94	-299.6	-297.46	-292.67	-285.12	21.775	21.907	22.042	
r79	-9.9437	-9.9325	-9.8557	-9.6597	-9.3232	2.5394	2.5501	2.5602	
r80	-8.9023	-8.8911	-8.8143	-8.6183	-8.2818	2.3039	2.3121	2.3198	
r81	-7.6363	-7.6251	-7.5484	-7.3524	-7.0159	2.0068	2.0124	2.0176	
r82	-5.9673	-5.9561	-5.8793	-5.6833	-5.3468	1.5958	1.5989	1.6018	
r83	-3.5892	-3.578	-3.5012	-3.3052	-2.9687	0.97904	0.98004	0.98097	0

xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c5264
	--	--	-----	-----	-----	-----	-----	-----	-----
r1	0	0	0.0005656	0.0075134	0.022901	114.75	120.41	126.27	132.3
r2	0	0	0.00051498	0.0065334	0.021549	114.86	120.53	126.41	132.5
r3	0	0	0.00051498	0.0049294	0.019875	114.97	120.65	126.56	132.
r4	0	0	0.00051498	0.0047937	0.019672	115.73	121.42	127.34	133.5
r5	0	0	0.00048517	0.0046683	0.019484	116.5	122.21	128.15	134.3
r79	0	0	0	0	0	81.091	85.68	90.335	94.37
r80	0	0	0	0	0	76.669	80.563	84.304	88.0
r81	0	0	0	0	0	68.313	71.534	74.475	77.83
r82	0	0	0	0	0	50.126	53.467	56.953	58.74
r83	0	0	0	0	0	0	0	0	0

xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498
	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.036717	0.037251	0.040426	0.04363	0.048012	9.6491	9.817	9.9649
r2	0.036717	0.037251	0.040477	0.04461	0.049364	9.8118	9.9685	10.101
r3	0.036717	0.037251	0.040477	0.046214	0.051039	9.9779	10.12	10.234
r4	0.038144	0.038678	0.041903	0.047776	0.052666	10.131	10.258	10.354
r5	0.039534	0.040068	0.043323	0.04929	0.054241	10.272	10.384	10.463
r79	0.2179	0.21844	0.22216	0.23228	0.25197	35.858	37.092	38.455
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.253	42.183	44.459
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.587	51.19	54.266
r82	0.2179	0.21844	0.22216	0.23228	0.25197	66.755	69.238	71.77

r83	0.2179	0.21844	0.22216	0.23228	0.25197	116.87	122.69	128.71
-----	--------	---------	---------	---------	---------	--------	--------	--------

Second, solve for the unemployment value, use the exact-bisec result code, call the snw_vfi_main_bisec_vec.m function with a third input of existing value. xi is the share of income lost during covid year given surprise covid shock, b is the share of income loss that is covered by unemployment insurance. xi=0.5 and b=0 means will lose 50 percent of income given COVID shocks, and the loss will not be covered at all by unemployment insurance.

```
mp_params('xi') = 0.5;
mp_params('b') = 0;
mp_params('a2_covidyr') = mp_params('a2_covidyr_manna_heaven');
[V_VFI_unemp,ap_VFI_unemp,cons_VFI_unemp,mp_valpol_more_unemp] = ...
    snw_vfi_main_bisec_vec(mp_params, mp_controls, V_VFI_ss);
```

```
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 1 of 82, time-this-age:5.8952
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 2 of 82, time-this-age:5.8156
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 3 of 82, time-this-age:5.8749
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 4 of 82, time-this-age:6.0917
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 5 of 82, time-this-age:5.8328
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 6 of 82, time-this-age:5.8616
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 7 of 82, time-this-age:5.8821
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 8 of 82, time-this-age:5.8446
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 9 of 82, time-this-age:5.8487
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 10 of 82, time-this-age:5.8319
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 11 of 82, time-this-age:5.8572
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 12 of 82, time-this-age:5.884
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 13 of 82, time-this-age:5.84
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 14 of 82, time-this-age:5.8824
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 15 of 82, time-this-age:5.8512
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 16 of 82, time-this-age:5.8691
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 17 of 82, time-this-age:5.8793
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 18 of 82, time-this-age:5.8426
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 19 of 82, time-this-age:5.8459
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 20 of 82, time-this-age:5.8282
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 21 of 82, time-this-age:5.833
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 22 of 82, time-this-age:5.825
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 23 of 82, time-this-age:5.8161
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 24 of 82, time-this-age:5.816
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 25 of 82, time-this-age:5.8359
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 26 of 82, time-this-age:5.865
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 27 of 82, time-this-age:5.8444
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 28 of 82, time-this-age:5.8535
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 29 of 82, time-this-age:5.8373
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 30 of 82, time-this-age:5.839
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 31 of 82, time-this-age:7.5134
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 32 of 82, time-this-age:5.8543
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 33 of 82, time-this-age:5.9343
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 34 of 82, time-this-age:5.8427
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 35 of 82, time-this-age:5.8876
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 36 of 82, time-this-age:5.866
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 37 of 82, time-this-age:5.8107
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 38 of 82, time-this-age:6.3797
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 39 of 82, time-this-age:5.8577
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 40 of 82, time-this-age:5.8751
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 41 of 82, time-this-age:7.5158
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 42 of 82, time-this-age:5.8538
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 43 of 82, time-this-age:5.8376
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 44 of 82, time-this-age:5.8281
SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 45 of 82, time-this-age:5.8513
```

SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 46 of 82, time-this-age:5.7947
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 47 of 82, time-this-age:5.8408
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 48 of 82, time-this-age:6.7877
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 49 of 82, time-this-age:5.5985
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 50 of 82, time-this-age:5.5925
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 51 of 82, time-this-age:5.634
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 52 of 82, time-this-age:5.6392
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 53 of 82, time-this-age:5.6292
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 54 of 82, time-this-age:5.7258
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 55 of 82, time-this-age:5.6309
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 56 of 82, time-this-age:5.6201
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 57 of 82, time-this-age:5.664
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 58 of 82, time-this-age:5.6269
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 59 of 82, time-this-age:5.6471
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 60 of 82, time-this-age:5.6017
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 61 of 82, time-this-age:5.6667
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 62 of 82, time-this-age:5.6161
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 63 of 82, time-this-age:5.6465
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 64 of 82, time-this-age:5.7106
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 65 of 82, time-this-age:5.6156
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 66 of 82, time-this-age:5.5758
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 67 of 82, time-this-age:5.6609
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 68 of 82, time-this-age:5.6021
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 69 of 82, time-this-age:5.6702
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 70 of 82, time-this-age:5.6106
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 71 of 82, time-this-age:5.6488
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 72 of 82, time-this-age:5.6466
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 73 of 82, time-this-age:5.6387
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 74 of 82, time-this-age:5.6332
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 75 of 82, time-this-age:5.6365
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 76 of 82, time-this-age:5.6274
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 77 of 82, time-this-age:5.6614
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 78 of 82, time-this-age:5.6463
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 79 of 82, time-this-age:5.6609
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 80 of 82, time-this-age:5.6158
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 81 of 82, time-this-age:5.6678
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 82 of 82, time-this-age:5.6513
 SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock: Age 83 of 82, time-this-age:6.9657
 Completed SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=d

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

CONTAINER NAME: mp_outcomes ND Array (Matrix etc)

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

	i	idx	ndim	numel	rowN	colN	sum	mean	std
	-	---	----	-----	---	-----	-----	-----	-----
V_VFI	1	1	6	4.37e+07	83	5.265e+05	-1.7805e+08	-4.0743	27.11
ap_VFI	2	2	6	4.37e+07	83	5.265e+05	1.3789e+09	31.553	36.67
cons_VFI	3	3	6	4.37e+07	83	5.265e+05	2.1097e+08	4.8277	8.328

xxx TABLE:V_VFI xxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c
	-----	-----	-----	-----	-----	-----	-----	-----	-----
r1	-372.97	-371.47	-362.94	-349.52	-336.96	21.573	21.728	21.882	
r2	-360.84	-359.34	-350.81	-337.39	-324.98	21.595	21.745	21.894	
r3	-348.91	-347.41	-338.88	-325.46	-313.34	21.617	21.762	21.906	
r4	-336.09	-334.7	-326.73	-314.01	-302.44	21.633	21.772	21.913	

r5	-324.48	-323.18	-315.72	-303.62	-292.54	21.634	21.77	21.907
r79	-9.9437	-9.9325	-9.8557	-9.6597	-9.3232	2.5374	2.5482	2.5584
r80	-8.9023	-8.8911	-8.8143	-8.6183	-8.2818	2.3024	2.3107	2.3185
r81	-7.6363	-7.6251	-7.5484	-7.3524	-7.0159	2.0057	2.0114	2.0168
r82	-5.9673	-5.9561	-5.8793	-5.6833	-5.3468	1.5952	1.5984	1.6014
r83	-3.5892	-3.578	-3.5012	-3.3052	-2.9687	0.97886	0.97987	0.98082
								0

xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499	c526500
	--	--	--	--	-----	-----	-----	-----	-----	-----
r1	0	0	0	0	0.0092181	110.06	115.71	121.55	127.62	133.93
r2	0	0	0	0	0.008238	110.03	115.68	121.54	127.62	133.95
r3	0	0	0	0	0.0066341	109.99	115.65	121.53	127.63	133.97
r4	0	0	0	0	0.0058019	110.28	115.95	121.84	127.96	134.33
r5	0	0	0	0	0.004998	110.58	116.27	122.17	128.31	134.69
r79	0	0	0	0	0	81.091	85.229	89.297	93.341	97.382
r80	0	0	0	0	0	75.865	79.539	83.28	87.016	90.669
r81	0	0	0	0	0	67.781	70.521	73.462	76.819	81.091
r82	0	0	0	0	0	50.126	53.467	56.108	57.742	60.587
r83	0	0	0	0	0	0	0	0	0	0

xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498
	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.018623	0.019158	0.022901	0.033062	0.04363	9.4708	9.6491	9.817
r2	0.018623	0.019158	0.022901	0.033062	0.04461	9.6414	9.8118	9.9685
r3	0.018623	0.019158	0.022901	0.033062	0.046214	9.8179	9.9779	10.12
r4	0.019354	0.019888	0.023632	0.033792	0.047776	9.9825	10.131	10.258
r5	0.020066	0.020601	0.024344	0.034504	0.04929	10.135	10.272	10.384
r79	0.2179	0.21844	0.22216	0.23228	0.25197	34.82	36.506	38.455
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.033	42.183	44.459
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.106	51.19	54.266
r82	0.2179	0.21844	0.22216	0.23228	0.25197	65.751	68.234	71.611
r83	0.2179	0.21844	0.22216	0.23228	0.25197	115.87	121.69	127.71

Difference Between Value and Choices In Unemployment and Future Periods

```
V_VFI_unemp_drop = V_VFI_ss - V_VFI_unemp;
ap_VFI_unemp_drop = ap_VFI_ss - ap_VFI_unemp;
cons_VFI_unemp_drop = cons_VFI_ss - cons_VFI_unemp;
```

5.1.2 Define Parameter Frames

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

```
% Grids:
age_grid = 18:100;
agrid = mp_params('agrid');
eta_H_grid = mp_params('eta_H_grid');
eta_S_grid = mp_params('eta_S_grid');
ar_st_eta_HS_grid = string(cellstr([num2str(eta_H_grid', 'hz=%3.2f;'), num2str(eta_S_grid', 'wz=%3.2f;'), ...
edu_grid = [0,1];
marry_grid = [0,1];
kids_grid = (1:1:mp_params('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});

```

5.1.3 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_xttitle') = {'Savings States, a'};
mp_support_graph('st_legend_loc') = 'eastoutside';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('it_legend_select') = 15; % how many shock legends to show
mp_support_graph('cl_colors') = 'jet';

MEAN(VAL(A,Z) - VAL(A,Z|unemp)), MEAN(AP(A,Z) - AP(A,Z|unemp)), MEAN(C(A,Z) - C(A,Z|unemp))

Tabulate value and policies along savings and shocks:

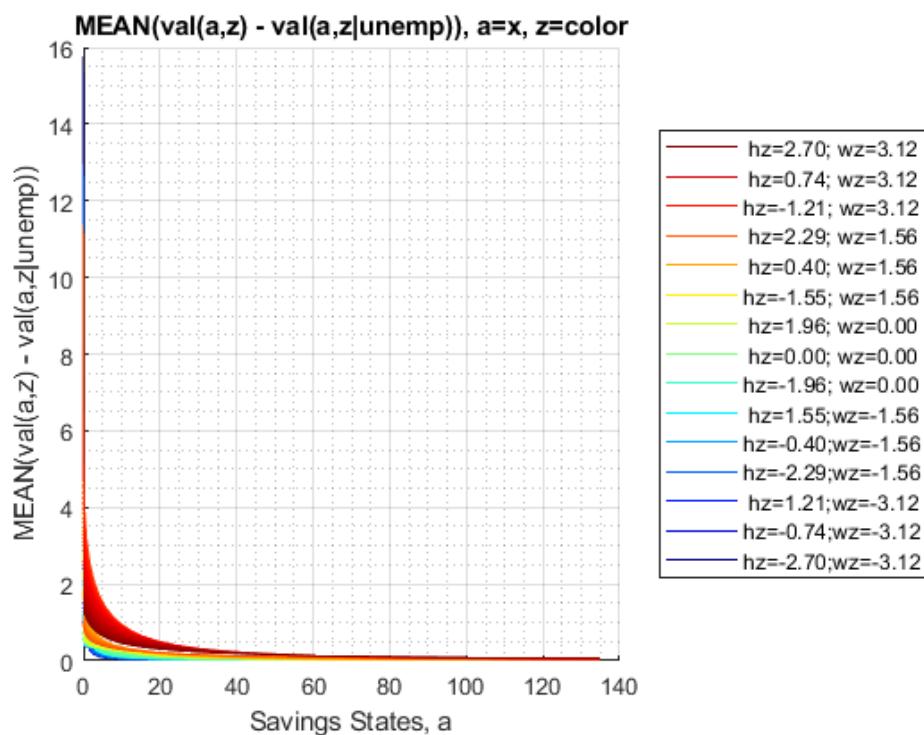
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [1,4,5,6,3,2];
% Value Function
tb_az_v = ff_summ_nd_array("MEAN(v(A,Z) - v(A,Z|unemp))", V_VFI_unemp_drop, true, ["mean"], 4, 1, cl

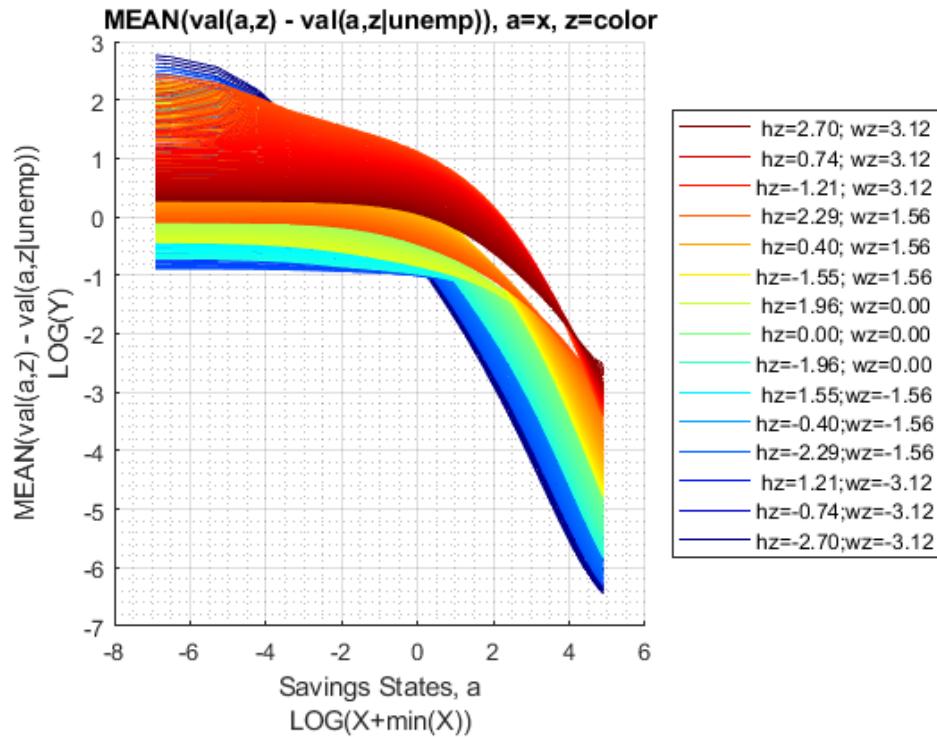
xxx MEAN(v(A,Z) - v(A,Z|unemp)) xxxxxxxxxxxxxxxxxxxxxxxx
group      savings      mean_eta_1      mean_eta_2      mean_eta_3      mean_eta_4      mean_eta_5      mean_eta_6
-----  -----
1           0            15.753        14.805        13.912        13.072        12.281

```

1	0	0.019317	0.020449	0.021654	0.022935	0.024299
---	---	----------	----------	----------	----------	----------

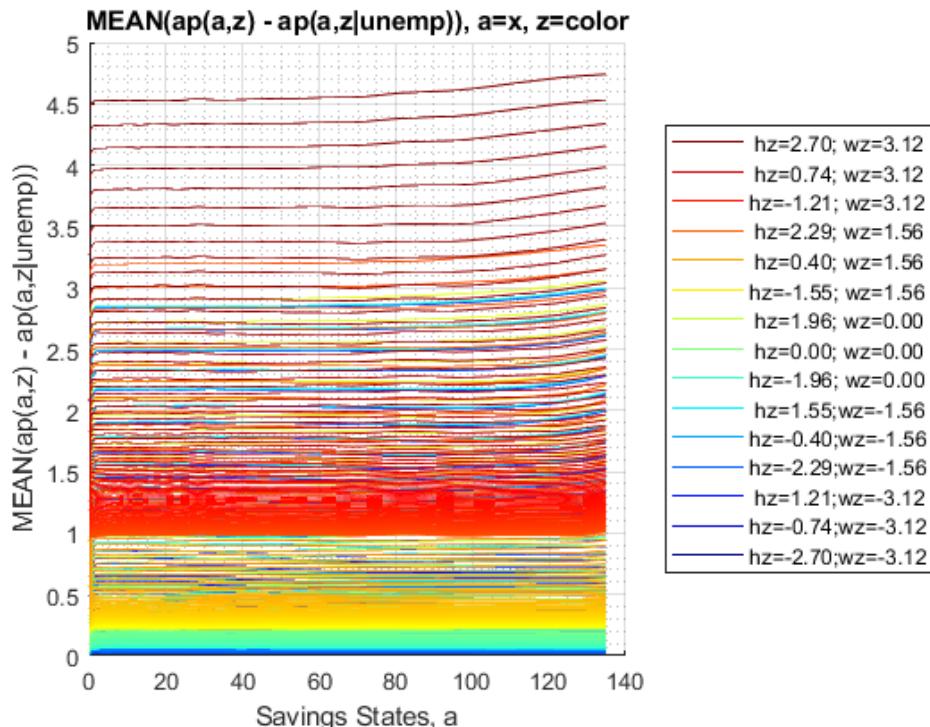
```
mp_support_graph('cl_st_graph_title') = {'MEAN(val(a,z) - val(a,z|unemp)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(val(a,z) - val(a,z|unemp))'};
ff_graph_grid((tb_az_v{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);
```

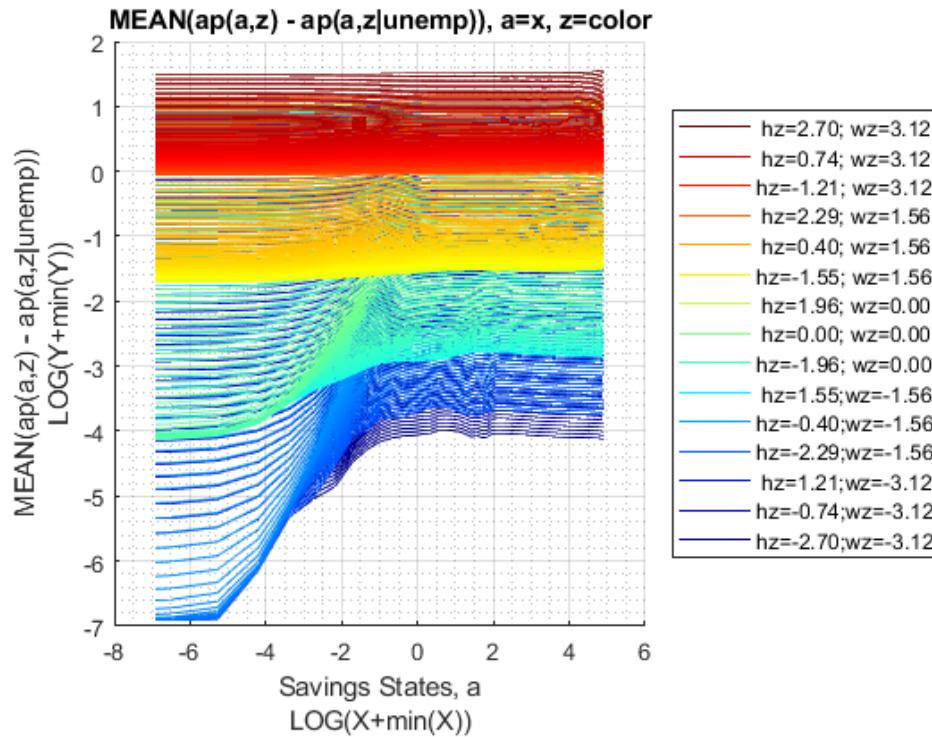




Graph Mean Savings Choices Change:

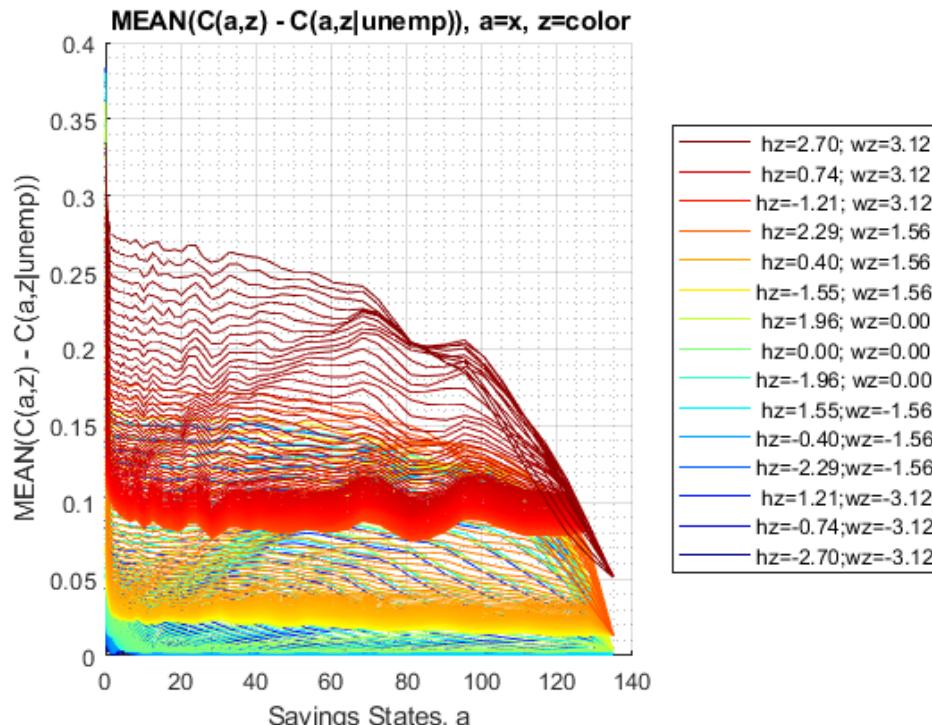
```
mp_support_graph('cl_st_graph_title') = {'MEAN(ap(a,z) - ap(a,z|unemp)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(ap(a,z) - ap(a,z|unemp))'};
ff_graph_grid((tb_az_ap{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```

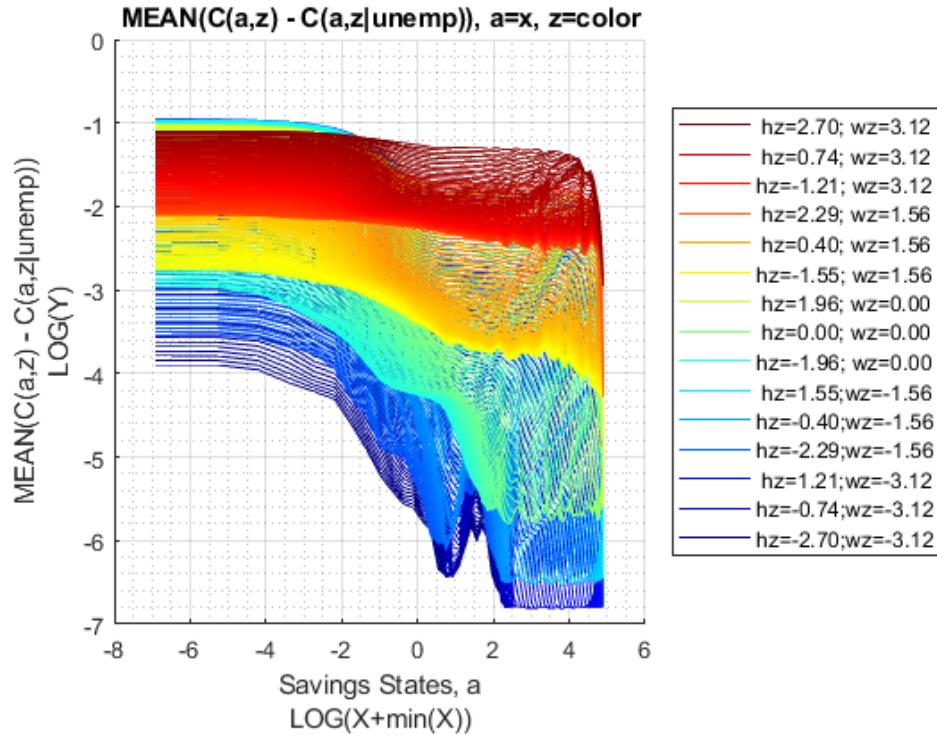




Graph Mean Consumption Change:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(C(a,z) - C(a,z|unemp)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(C(a,z) - C(a,z|unemp))'};
ff_graph_grid((tb_az_c{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);
```





5.1.4 Analyze Kids and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = [...
    "k0M0", "K1M0", "K2M0", "K3M0", "K4M0", ...
    "k0M1", "K1M1", "K2M1", "K3M1", "K4M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {...
    'o', 'd', 's', 'x', '*', ...
    'o', 'd', 's', 'x', '*'};
mp_support_graph('cl_colors') = {...
    'red', 'red', 'red', 'red', 'red',...
    'blue', 'blue', 'blue', 'blue', 'blue'};
```

MEAN($V(KM,J) - V(KM,J | \text{unemp})$), MEAN($ap(KM,J) - ap(KM,J | \text{unemp})$), MEAN($c(KM,J) - c(KM,J | \text{unemp})$)

Tabulate value and policies:

```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,4,1,6,5];
% Value Function
tb_az_v = ff_summ_nd_array("MEAN(V(KM,J) - V(KM,J | unemp))", V_VFI_unemp_drop, true, ["mean"], 3, 1);

xxx MEAN(V(KM,J) - V(KM,J | unemp)) xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry mean_age_18 mean_age_19 mean_age_20 mean_age_21 mean_age_22
----- ----- ----- ----- ----- ----- ----- -----
1 1 0 0.61637 0.59885 0.58106 0.56498 0.55117
```

2	2	0	0.82734	0.80489	0.78136	0.75704	0.73572
3	3	0	0.96755	0.94502	0.92045	0.89136	0.86587
4	4	0	1.0948	1.0713	1.045	1.0118	0.9827
5	5	0	1.2011	1.1779	1.151	1.1149	1.0833
6	1	1	0.76784	0.74924	0.73091	0.71544	0.70238
7	2	1	0.93021	0.90698	0.88323	0.86203	0.84347
8	3	1	1.0185	0.9941	0.96877	0.94495	0.92408
9	4	1	1.1171	1.0915	1.0645	1.0382	1.0151
10	5	1	1.1585	1.1346	1.1083	1.0807	1.0569

% Aprime Choice

```
tb_az_ap = ff_summ_nd_array("MEAN(ap(KM,J) - ap(KM,J | unemp))", ap_VFI_unemp_drop, true, ["mean"], 3)
```

xxx MEAN(ap(KM,J) - ap(KM,J | unemp)) xxxxxxxxxxxxxxxxxxxxxxxxx

group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	1	0	0.54429	0.54157	0.53838	0.57688	0.61527
2	2	0	0.53828	0.53451	0.53011	0.56791	0.60562
3	3	0	0.53173	0.52734	0.52253	0.55991	0.59734
4	4	0	0.5276	0.523	0.51797	0.55513	0.59235
5	5	0	0.52354	0.51894	0.51381	0.55085	0.58805
6	1	1	1.1323	1.1757	1.2198	1.3119	1.4048
7	2	1	1.0396	1.0753	1.1115	1.1942	1.2777
8	3	1	0.97097	1.002	1.0331	1.1097	1.187
9	4	1	0.89591	0.92257	0.94909	1.0212	1.0937
10	5	1	0.78017	0.79798	0.81575	0.87811	0.94079

% Consumption Choices

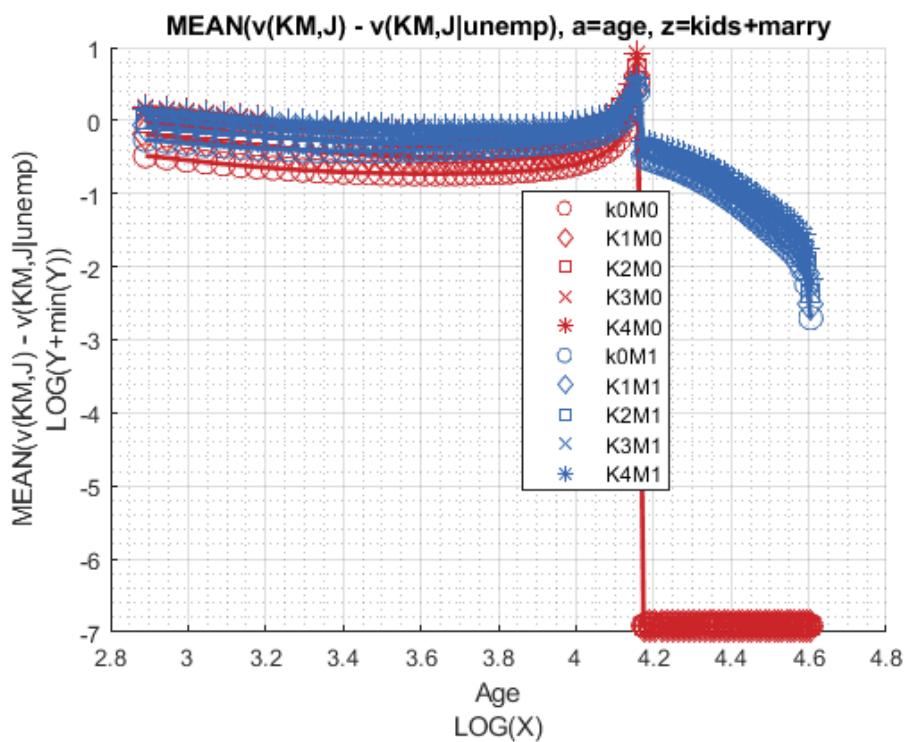
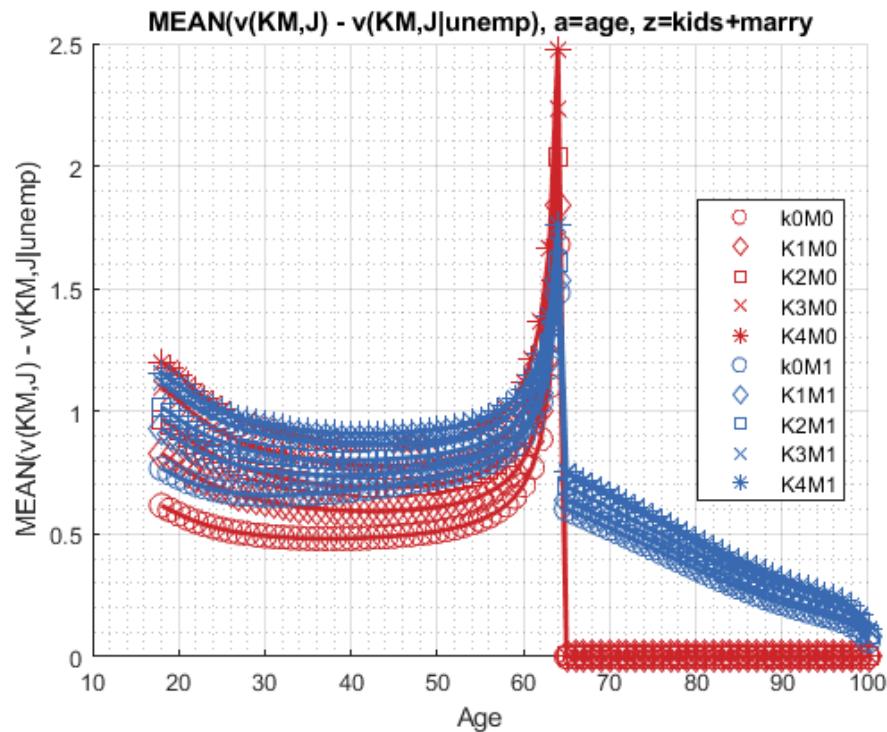
```
tb_az_c = ff_summ_nd_array("MEAN(c(KM,J) - c(KM,J | unemp))", cons_VFI_unemp_drop, true, ["mean"], 3)
```

xxx MEAN(c(KM,J) - c(KM,J | unemp)) xxxxxxxxxxxxxxxxxxxxxxxxx

group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	1	0	0.050084	0.052801	0.055995	0.056344	0.056497
2	2	0	0.056094	0.059866	0.064267	0.065317	0.06615
3	3	0	0.062643	0.067034	0.071841	0.073312	0.074434
4	4	0	0.06677	0.071371	0.076406	0.078097	0.079421
5	5	0	0.07083	0.075431	0.080561	0.082377	0.083719
6	1	1	0.091654	0.09722	0.1029	0.10693	0.11041
7	2	1	0.087426	0.093165	0.099035	0.10362	0.10765
8	3	1	0.089332	0.094467	0.10022	0.10478	0.10884
9	4	1	0.095488	0.099656	0.10451	0.10733	0.10981
10	5	1	0.1018	0.10631	0.11124	0.11381	0.11605

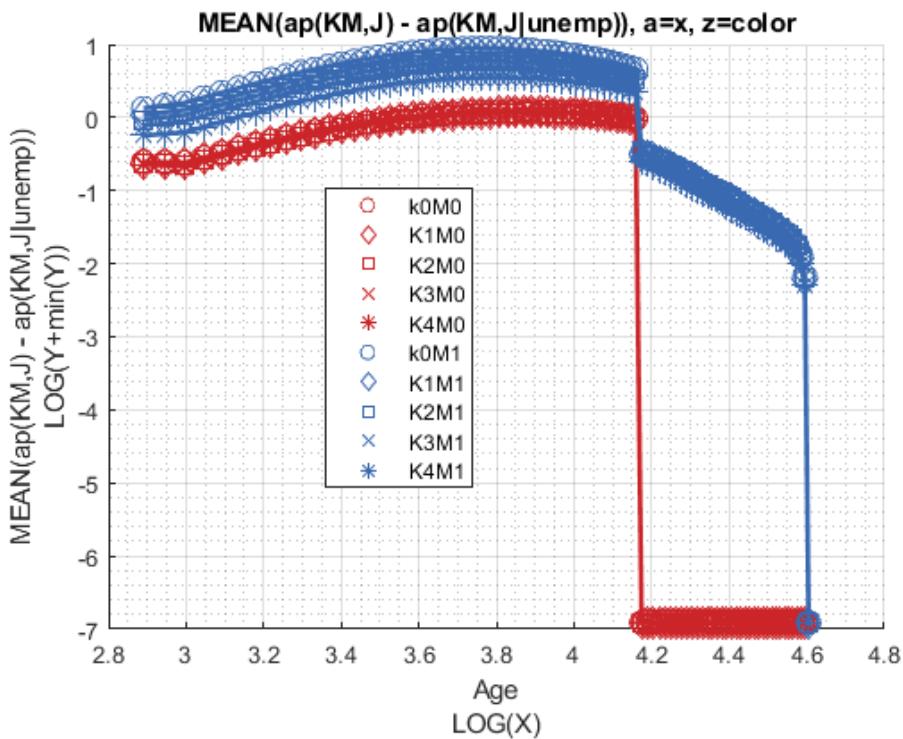
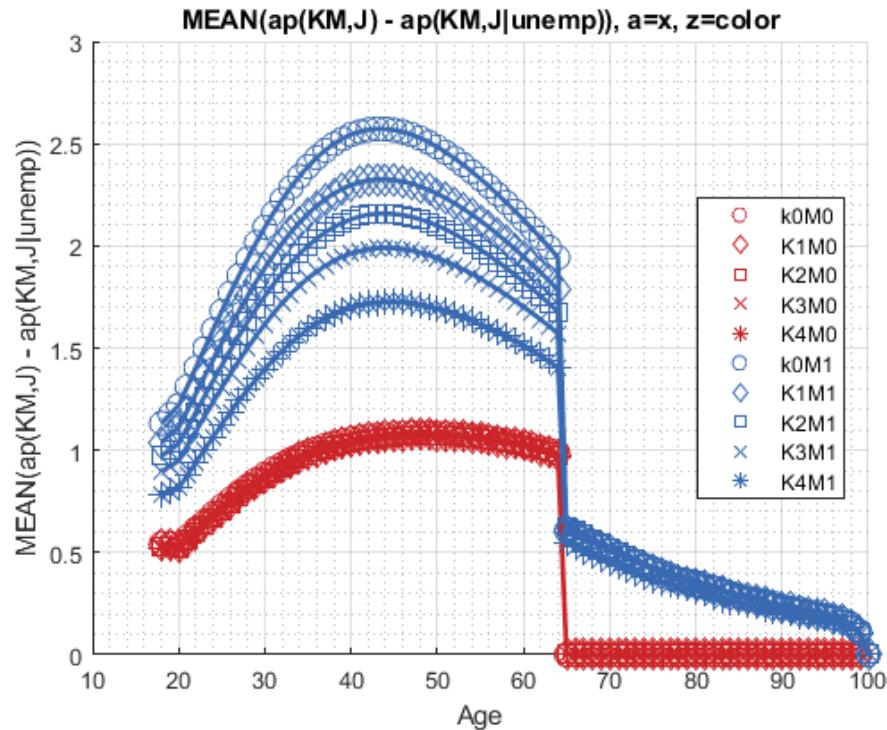
Graph Mean Values Change:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(v(KM,J) - v(KM,J|unemp), a=age, z=kids+marry')};
mp_support_graph('cl_st_ytitle') = {'MEAN(v(KM,J) - v(KM,J|unemp))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



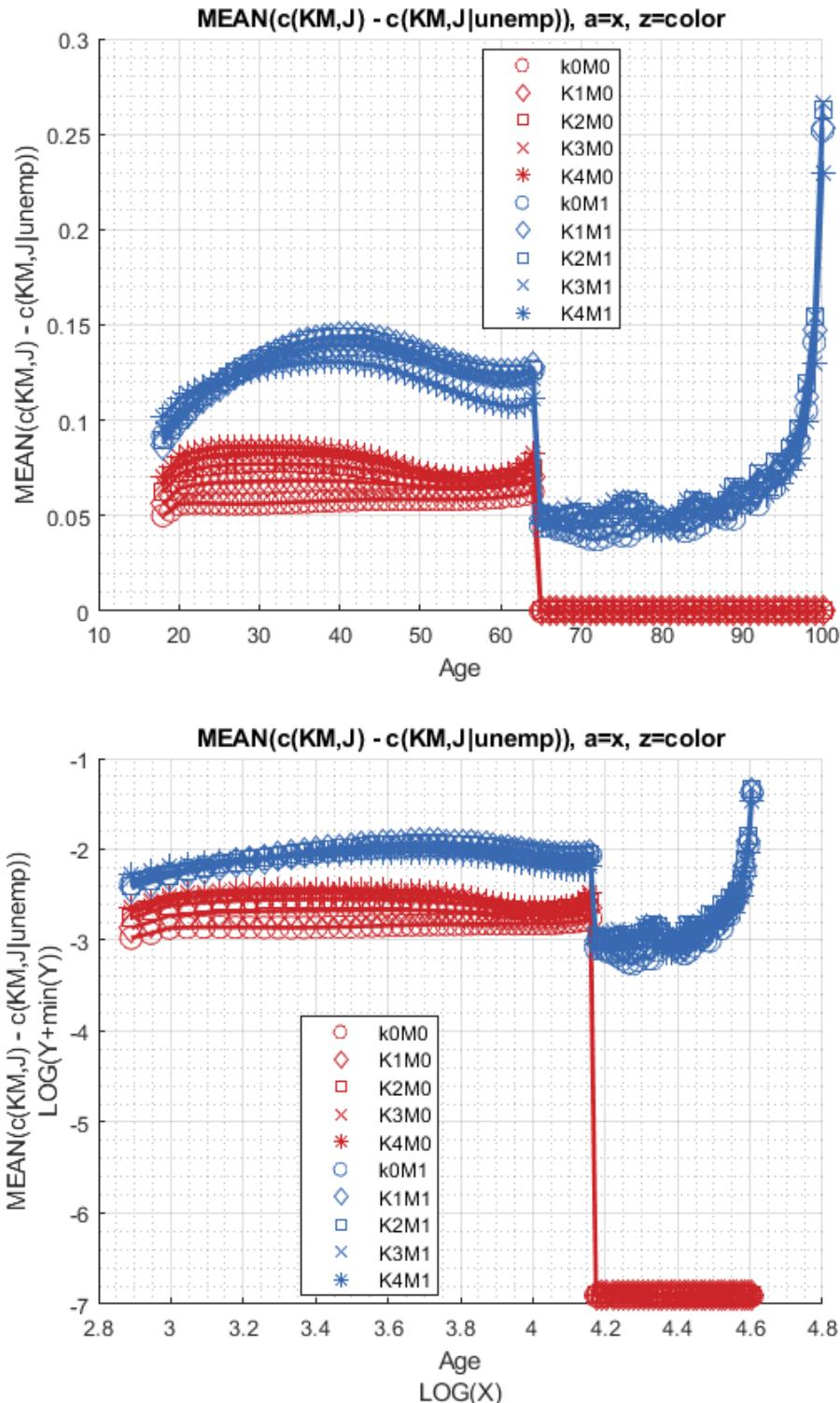
Graph Mean Savings Choices Change:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(ap(KM,J) - ap(KM,J|unemp)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(ap(KM,J) - ap(KM,J|unemp))'};
ff_graph_grid((tb_az_ap{1:end}, 4:end)), ar_row_grid, age_grid, mp_support_graph);
```



Graph Mean Consumption Change:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(c(KM,J) - c(KM,J|unemp)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(c(KM,J) - c(KM,J|unemp))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



5.1.5 Analyze Education and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["E0M0", "E1M0", "E0M1", "E1M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
```

```
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {'*', 'p', '*', 'p'};
mp_support_graph('cl_colors') = {'red', 'red', 'blue', 'blue'};
```

MEAN(v(EKM,J) - v(EKM,J|unemp)), MEAN(ap(EM,J) - ap(EM,J|unemp)), MEAN(c(EM,J) - c(EM,J|unemp))

Tabulate value and policies:

```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,6,1,4,5];
% Value Function
tb_az_v = ff_summ_nd_array("MEAN(v(EM,J) - v(EM,J|unemp))", V_VFI_unemp_drop, true, ["mean"], 3, 1);

xxx MEAN(v(EM,J) - v(EM,J|unemp)) xxxxxxxxxxxxxxxxxxxxxxxx
group   edu   marry   mean_age_18   mean_age_19   mean_age_20   mean_age_21   mean_age_22
-----  ---  -----  -----  -----  -----  -----  -----
1       0      0      0.98303     0.96405     0.94385     0.92458     0.90689
2       1      0      0.89982     0.87513     0.84768     0.81144     0.78062
3       0      1      1.0503      1.0306      1.0104      0.99222     0.97585
4       1      1      0.94657     0.91993     0.89191     0.86431     0.84092

% Aprime Choice
tb_az_ap = ff_summ_nd_array("MEAN(ap(EM,J) - ap(EM,J|unemp))", ap_VFI_unemp_drop, true, ["mean"], 3, 1);

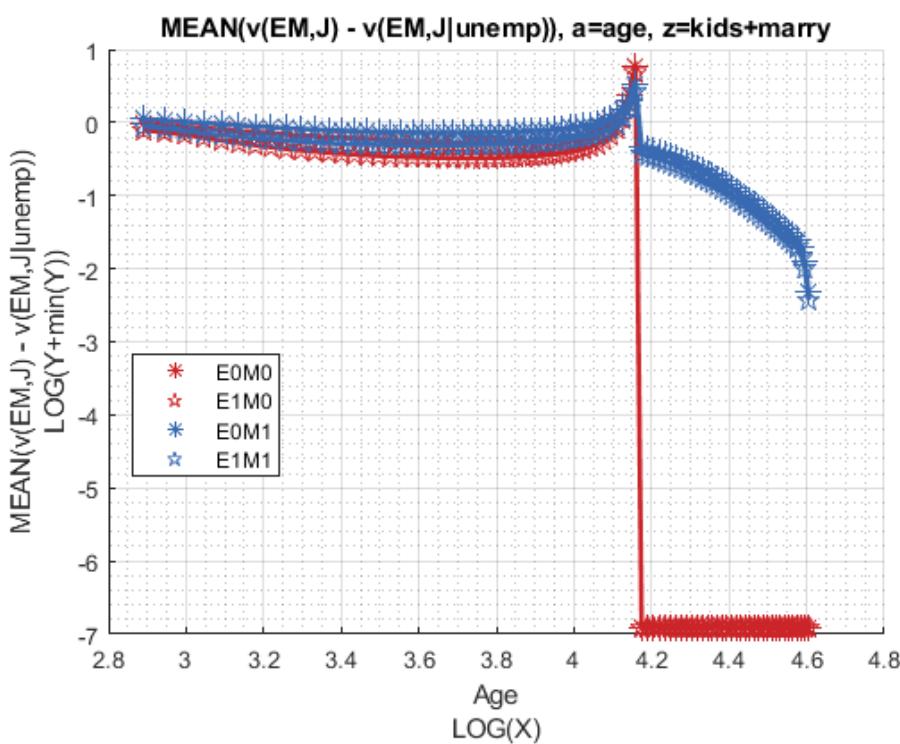
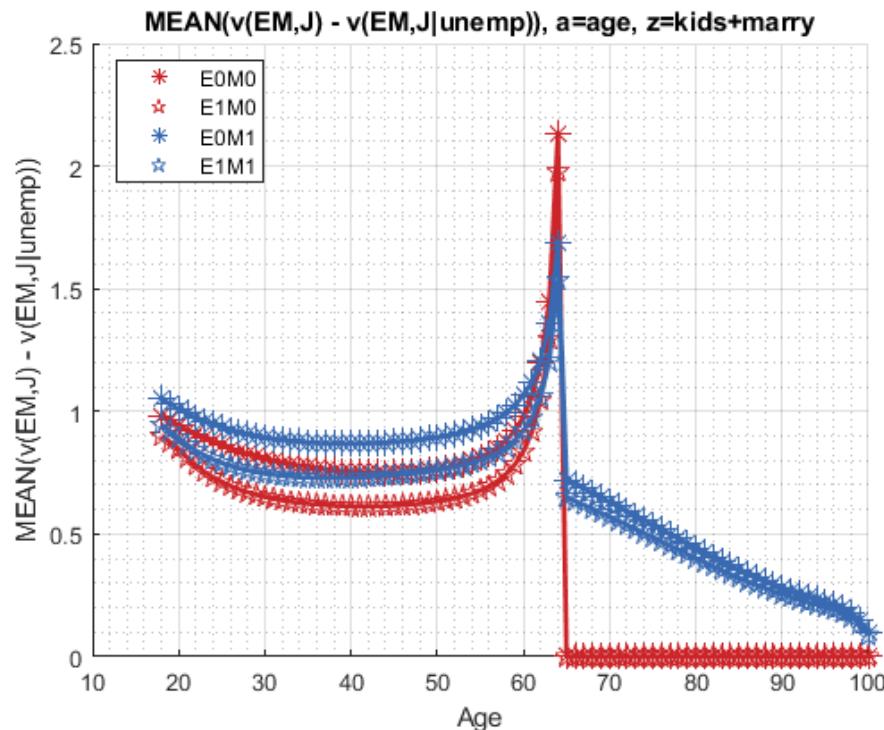
xxx MEAN(ap(EM,J) - ap(EM,J|unemp)) xxxxxxxxxxxxxxxxxxxxxxxx
group   edu   marry   mean_age_18   mean_age_19   mean_age_20   mean_age_21   mean_age_22
-----  ---  -----  -----  -----  -----  -----  -----
1       0      0      0.54395     0.54191     0.53951     0.56214     0.58423
2       1      0      0.52222     0.51623     0.50961     0.56213     0.61523
3       0      1      0.93033     0.95904     0.98801     1.0446      1.1011
4       1      1      0.99726     1.0304      1.0637      1.1614      1.2605

% Consumption Choices
tb_az_c = ff_summ_nd_array("MEAN(c(EM,J) - c(EM,J|unemp))", cons_VFI_unemp_drop, true, ["mean"], 3, 1);

xxx MEAN(c(EM,J) - c(EM,J|unemp)) xxxxxxxxxxxxxxxxxxxxxxxx
group   edu   marry   mean_age_18   mean_age_19   mean_age_20   mean_age_21   mean_age_22
-----  ---  -----  -----  -----  -----  -----  -----
1       0      0      0.05042     0.052463    0.054861    0.055684    0.056488
2       1      0      0.072148    0.078138    0.084767    0.086495    0.0876
3       0      1      0.079245    0.082789    0.086633    0.089336    0.091941
4       1      1      0.10704     0.11354     0.12053     0.12525     0.12917
```

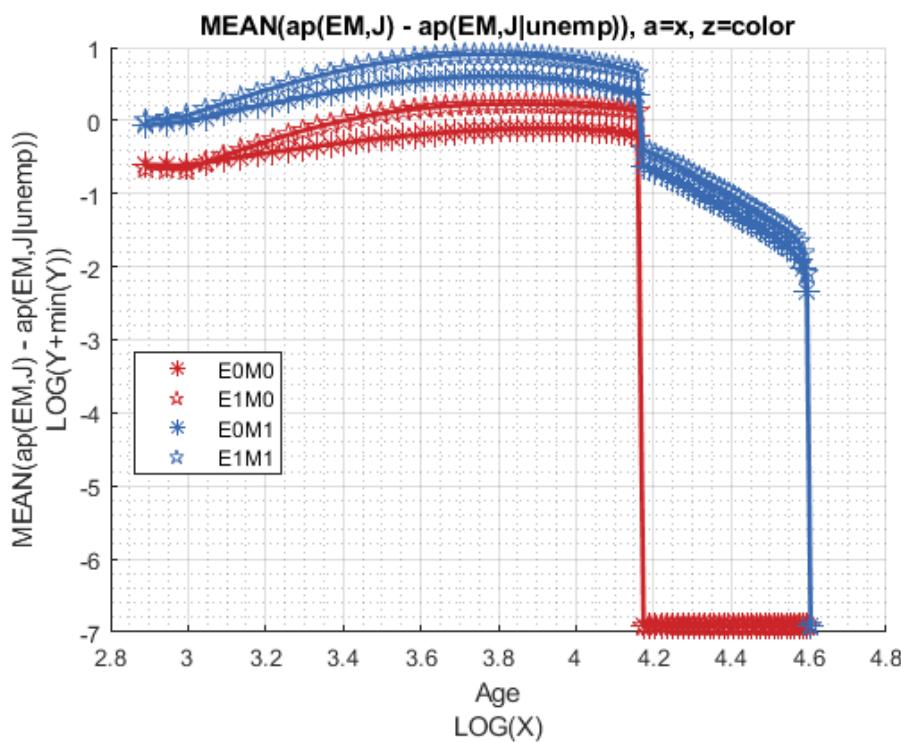
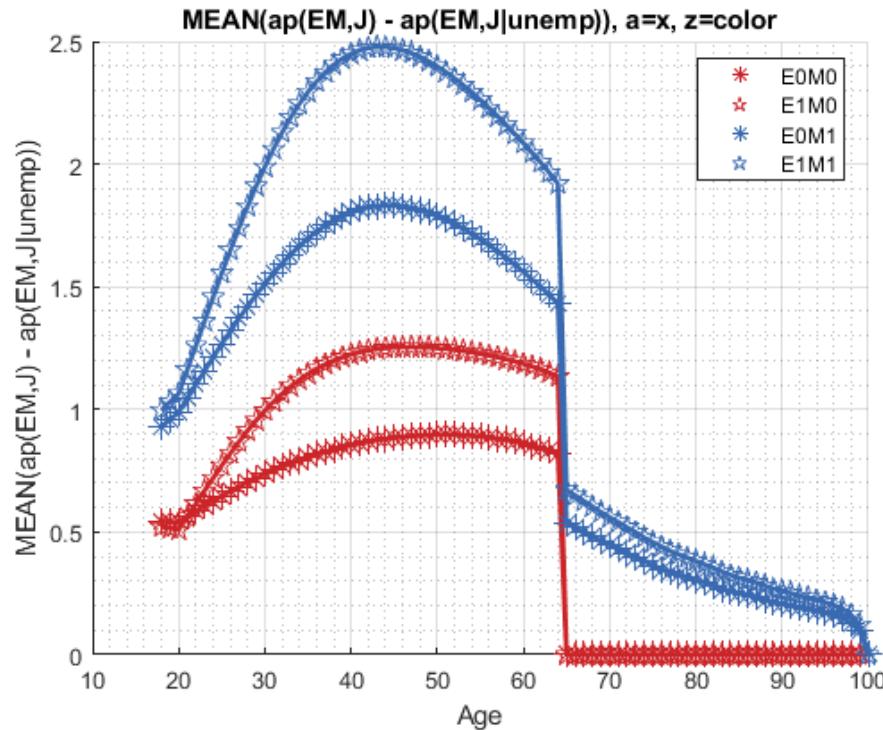
Graph Mean Values Change:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(v(EM,J) - v(EM,J|unemp)), a=age, z=kids+marry'};
mp_support_graph('cl_st_ytitle') = {'MEAN(v(EM,J) - v(EM,J|unemp))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



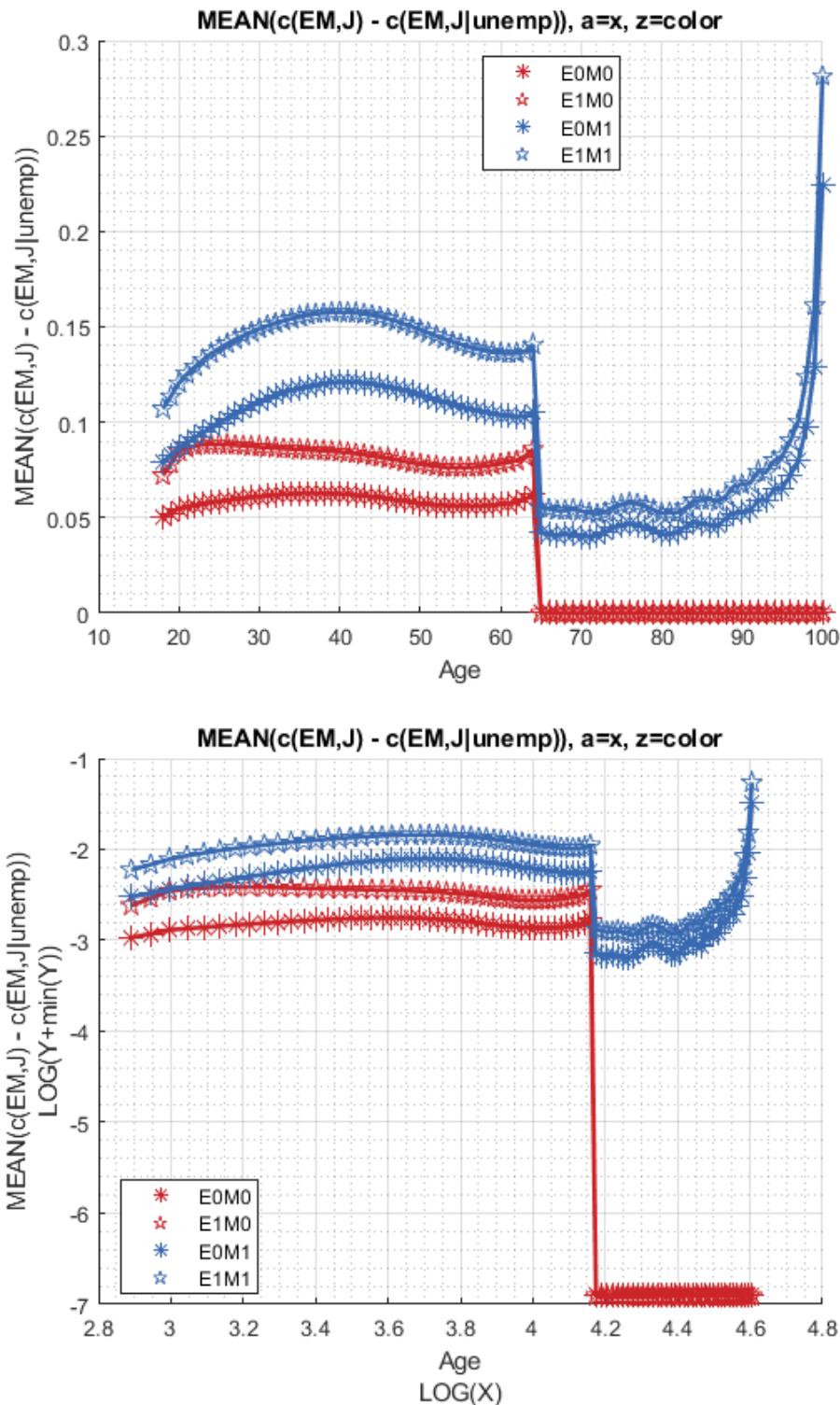
Graph Mean Savings Choices Change:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(ap(EM,J) - ap(EM,J|unemp)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(ap(EM,J) - ap(EM,J|unemp))'};
ff_graph_grid((tb_az_ap{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Graph Mean Consumption Change:

```
mp_support_graph('cl_st_graph_title') = {'MEAN(c(EM,J) - c(EM,J|unemp)), a=x, z=color'};
mp_support_graph('cl_st_ytitle') = {'MEAN(c(EM,J) - c(EM,J|unemp))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Chapter 6

Household Life Cycle Distribution

6.1 Distribution Exact Savings Choices

This is the example vignette for function: `snw_ds_main` from the [PrjOptiSNW Package](#). This function solves for vfi and gets distribution induced by policy functions and exogenous distributions. Looped to get distribution, but uses bisect vec for VFI.

6.1.1 Test SNW_DS_MAIN Defaults

Call the function with testing defaults.

```
mp_params = snw_mp_param('default_docdense');
mp_controls = snw_mp_control('default_test');
mp_controls('bl_print_vfi') = false;
mp_controls('bl_print_ds') = false;
mp_controls('bl_print_ds_verbose') = false;
[Phi_true,Phi_adj,A_agg,Y_inc_agg,it,mp_dsvfi_results] = snw_ds_main(mp_params, mp_controls);

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=496.
-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

      i      idx     ndim    numel      rowN      colN        sum      mean      std
      -      ---     ----  -----      ---  -----  -----  -----
V_VFI      1       1       6   4.37e+07      83  5.265e+05 -1.5339e+08 -3.5101  26.11
ap_VFI     2       2       6   4.37e+07      83  5.265e+05  1.4159e+09 32.402   36.79
cons_VFI   3       3       6   4.37e+07      83  5.265e+05  2.1402e+08 4.8975  8.329

xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxx
      c1      c2      c3      c4      c5      c526496    c526497    c526498    c
      ----  -----  -----  -----  -----  -----
r1 -346.51 -346.12 -343.63 -337.86 -328.51  21.702   21.852   22.003
r2 -334.38 -333.99 -331.51 -325.83 -316.83  21.724   21.869   22.015
r3 -322.45 -322.06 -319.6  -314.14 -305.6   21.745   21.885   22.027
r4 -310.63 -310.27 -307.99 -302.88 -294.87  21.767   21.903   22.041
r5 -299.94 -299.6  -297.46 -292.67 -285.12  21.775   21.907   22.042
r79 -9.9437 -9.9325 -9.8557 -9.6597 -9.3232  2.5394  2.5501  2.5602
r80 -8.9023 -8.8911 -8.8143 -8.6183 -8.2818  2.3039  2.3121  2.3198
r81 -7.6363 -7.6251 -7.5484 -7.3524 -7.0159  2.0068  2.0124  2.0176
r82 -5.9673 -5.9561 -5.8793 -5.6833 -5.3468  1.5958  1.5989  1.6018
```

r83	-3.5892	-3.578	-3.5012	-3.3052	-2.9687	0.97904	0.98004	0.98097	0
xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxxxxxx									
c1	c2	c3	c4	c5	c526496	c526497	c526498	c5264	
--	--	-----	-----	-----	-----	-----	-----	-----	-----
r1	0	0	0.0005656	0.0075134	0.022901	114.75	120.41	126.27	132.3
r2	0	0	0.00051498	0.0065334	0.021549	114.86	120.53	126.41	132.5
r3	0	0	0.00051498	0.0049294	0.019875	114.97	120.65	126.56	132.
r4	0	0	0.00051498	0.0047937	0.019672	115.73	121.42	127.34	133.5
r5	0	0	0.00048517	0.0046683	0.019484	116.5	122.21	128.15	134.3
r79	0	0	0	0	0	81.091	85.68	90.335	94.37
r80	0	0	0	0	0	76.669	80.563	84.304	88.0
r81	0	0	0	0	0	68.313	71.534	74.475	77.83
r82	0	0	0	0	0	50.126	53.467	56.953	58.74
r83	0	0	0	0	0	0	0	0	0
xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxxx									
c1	c2	c3	c4	c5	c526496	c526497	c526498		
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.036717	0.037251	0.040426	0.04363	0.048012	9.6491	9.817	9.9649	
r2	0.036717	0.037251	0.040477	0.04461	0.049364	9.8118	9.9685	10.101	
r3	0.036717	0.037251	0.040477	0.046214	0.051039	9.9779	10.12	10.234	
r4	0.038144	0.038678	0.041903	0.047776	0.052666	10.131	10.258	10.354	
r5	0.039534	0.040068	0.043323	0.04929	0.054241	10.272	10.384	10.463	
r79	0.2179	0.21844	0.22216	0.23228	0.25197	35.858	37.092	38.455	
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.253	42.183	44.459	
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.587	51.19	54.266	
r82	0.2179	0.21844	0.22216	0.23228	0.25197	66.755	69.238	71.77	
r83	0.2179	0.21844	0.22216	0.23228	0.25197	116.87	122.69	128.71	

Completed SNW_DS_MAIN;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=1498.3692

```
% [Phi_true,Phi_adj] = snw_ds_main(mp_params, mp_controls);
Phi_true = Phi_true/sum(Phi_true(:));
```

6.1.2 Show All Info in mp_dsvfi_results More Dense

```
mp_cl_mt_xyz_of_s = mp_dsvfi_results('mp_cl_mt_xyz_of_s');
disp(mp_cl_mt_xyz_of_s('tb_outcomes'))
```

	mean	unweighted_sum	sd	coefofvar	gini	min	
a_ss	4.2486	2228	6.7963	1.5996	0.68054	0	
ap_ss	4.3473	5.3198e+08	6.834	1.572	0.68147	0	
cons_ss	1.0676	5.0976e+07	0.69454	0.65055	0.3385	0.036717	
v_ss	-15.745	-2.1145e+07	21.68	-1.3769	-0.67203	-586.22	
n_ss	2.3554	21	1.4375	0.61029	0.3128	1	
y_all	1.415	8.3532e+07	1.4926	1.0548	0.47801	0	
y_head_inc	1.1087	1.9253e+06	1.0092	0.91029	0.41889	0.038108	
y_head_earn	0.88655	19732	0.92804	1.0468	0.53121	0	
y_spouse_inc	0.35849	4.8273e+05	0.95494	2.6638	0.85255	0	
yshr_interest	0.12214	3.8429e+06	0.16806	1.3759	0.66002	0	
yshr_wage	0.77513	8.8876e+06	0.33759	0.43553	0.2056	0	
yshr_SS	0.10273	30336	0.23637	2.3009	0.91226	0	
yshr_tax	0.17862	2.8339e+06	0.03519	0.19701	0.11226	0.036506	

yshr_nttxss	0.075896	2.8036e+06	0.25563	3.3681	1.3974	-0.89184
-------------	----------	------------	---------	--------	--------	----------

6.1.3 More Dense Param Results Define Frames

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

```
% Grids:
age_grid = 18:100;
agrid = mp_params('agrid');
eta_H_grid = mp_params('eta_H_grid');
eta_S_grid = mp_params('eta_S_grid');
ar_st_eta_HS_grid = string(cellstr([num2str(eta_H_grid), 'hz=%3.2f;'], num2str(eta_S_grid), 'wz=%3.2f'));
edu_grid = [0,1];
marry_grid = [0,1];
kids_grid = (1:1:mp_params('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});
```

6.1.4 Analyze Probability Mass Along Age Dimensions

Where are the mass at? Analyze mass given state space components.

```
% Get the Joint distribution over all states
% Define Graph Inputs
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = false; % do not log
```

Exogenous Permanent States Mass: Life Cycle, Edu and Marraige

Tabulate value and policies along savings and shocks:

```
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,6,1,5,4];
% Value Function
tb_prob_aem = ff_summ_nd_array("P(Age, EDU, MARRY)", Phi_true, true, ["sum"], 3, 1, cl_mp_datasetdesc);

xxx P(Age, EDU, MARRY) xxxxxxxxxxxxxxxxxxxxxxxx
group marry edu sum_age_18 sum_age_19 sum_age_20 sum_age_21 sum_age_22 s
----- ----- --- ----- ----- ----- ----- -----
1 0 0 0.0085768 0.0084866 0.0083969 0.0083078 0.0082194 0
2 1 0 0.0066438 0.0065739 0.0065044 0.0064354 0.0063669 0
3 0 1 0.0028875 0.0028571 0.002827 0.002797 0.0027672 0
4 1 1 0.0037292 0.0036899 0.0036509 0.0036122 0.0035738 0
```

```
mp_support_graph('cl_st_graph_title') = {'Pstationary(Age, Edu, Marry), age=x, marry/edu=color'};
mp_support_graph('cl_st_ytitle') = {'Conditional Aggregate Mass'};
ar_row_grid = ["MOE0", "M1E0", "MOE1", "M1E1"];
mp_support_graph('cl_st_xtitle') = {'Age Groups'};
mp_support_graph('cl_scatter_shapes') = {'*', '*', 'p', 'p' };
mp_support_graph('cl_colors') = {'red', 'blue', 'red', 'blue'};
ff_graph_grid((tb_prob_aem{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



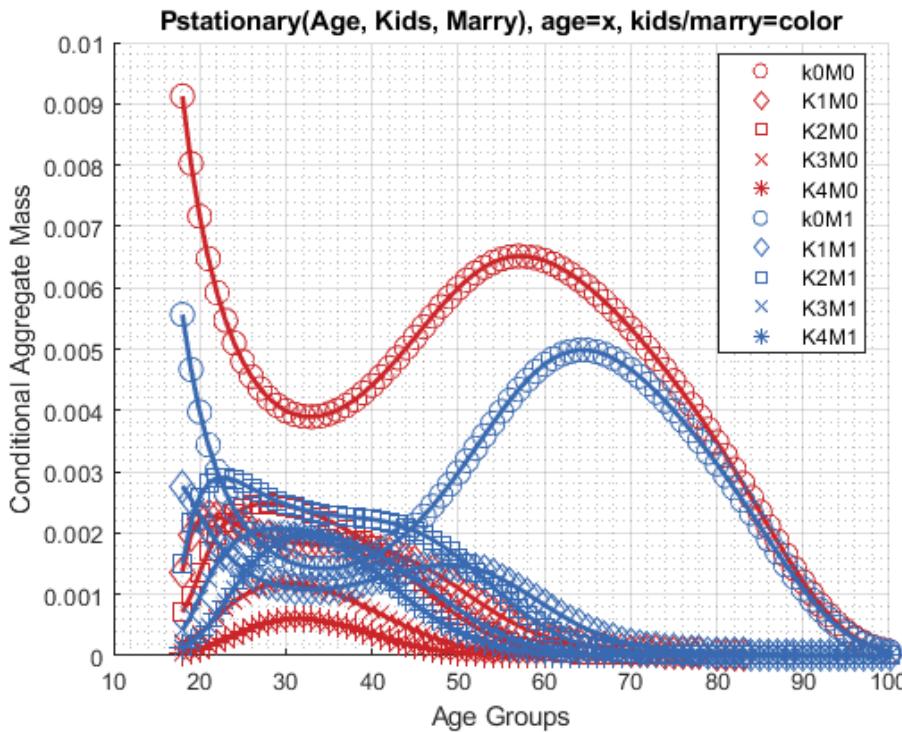
Kids and Marry By Age Mass

```
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,4,1,6,5];
% Value Function
tb_prob_amarrykids = ff_summ_nd_array("P(Age, Kids, Marry)", Phi_true, true, ["sum"], 3, 1, cl_mp_d

xxx P(Age, Kids, Marry) xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry sum_age_18 sum_age_19 sum_age_20 sum_age_21 sum_age_22
----- ----- ----- -----
1 1 0 0.0091249 0.0080278 0.0071652 0.0064765 0.0059205
2 2 0 0.0013699 0.0019743 0.0022187 0.0022858 0.0022687
3 3 0 0.00071266 0.00098425 0.0013537 0.0016929 0.0019639
4 4 0 0.00020622 0.00027865 0.00037326 0.00049476 0.00062818
5 5 0 5.0761e-05 7.8715e-05 0.000113 0.00015485 0.00020534
6 1 1 0.0055624 0.0046679 0.0039774 0.0034368 0.0030088
7 2 1 0.0027682 0.0025539 0.0023005 0.0020611 0.0018525
8 3 1 0.0014982 0.0021823 0.0025943 0.0028096 0.002896
9 4 1 0.00041197 0.00064648 0.00095224 0.0012491 0.0015009
10 5 1 0.00013221 0.0002132 0.00033097 0.00049097 0.00068255

mp_support_graph('cl_st_graph_title') = {'Pstationary(Age, Kids, Marry), age=x, kids/marry=color'};
mp_support_graph('cl_st_ytitle') = {'Conditional Aggregate Mass'};
ar_row_grid = [...
    "k0M0", "K1M0", "K2M0", "K3M0", "K4M0", ...
    "k0M1", "K1M1", "K2M1", "K3M1", "K4M1"];
mp_support_graph('cl_scatter_shapes') = {...
    'o', 'd', 's', 'x', '*', ...
    'o', 'd', 's', 'x', '*'};
mp_support_graph('cl_colors') = {...
    'red', 'red', 'red', 'red', 'red',...
    'blue', 'blue', 'blue', 'blue', 'blue'};
mp_support_graph('cl_st_xtitle') = {'Age Groups'};
```

```
ff_graph_grid((tb_prob_amarrykids{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



6.1.5 Analyze Probability Mass Asset and Shock Dimensions

Where are the mass at?

```
% Define Graph Inputs
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = false; % do not log

Asset and Shock Mass

% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [1,4,5,6,3,2];
% Value Function
tb_prob_az = ff_summ_nd_array("P(A,Z)", Phi_true, true, ["sum"], 4, 1, cl_mp_datasetdesc, ar_permut
```

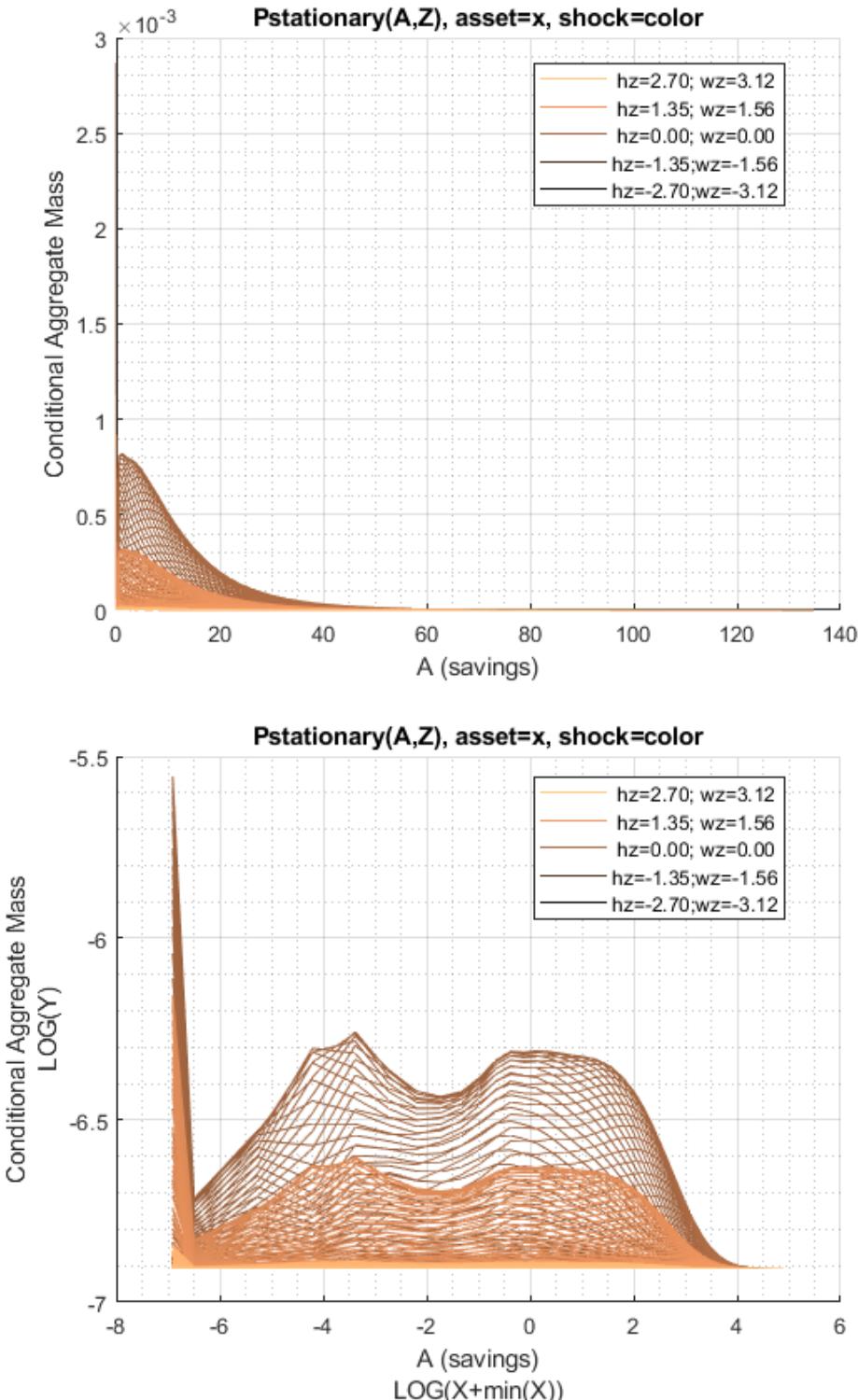
xxx P(A,Z))		xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx						
group	savings	sum_eta_1	sum_eta_2	sum_eta_3	sum_eta_4	sum_eta_5	sum	
1	0	1.6824e-07	1.4406e-07	2.1911e-07	3.1913e-07	4.5491e-07	6.4	
2	0.00051498	3.4279e-10	3.2632e-10	5.6501e-10	1.0203e-09	1.9975e-09	4.1	
3	0.0041199	7.1369e-10	6.2373e-10	9.7246e-10	1.4702e-09	2.2039e-09	3.2	
4	0.013905	1.573e-09	1.3633e-09	2.1044e-09	3.1331e-09	4.6025e-09	6.7	
5	0.032959	5.494e-09	4.7235e-09	7.23e-09	1.0641e-08	1.5401e-08	2.	
6	0.064373	6.5788e-09	5.6779e-09	8.702e-09	1.2804e-08	1.8492e-08	2.6	

```
mp_support_graph('cl_st_graph_title') = {'Pstationary(A,Z), asset=x, shock=color'};
mp_support_graph('cl_st_ytitle') = {'Conditional Aggregate Mass'};
mp_support_graph('cl_st_xtitle') = {'A (savings)'};
mp_support_graph('st_rowvar_name') = 'z=';
mp_support_graph('it_legend_select') = 5;
mp_support_graph('st_rounding') = '6.2f';
```

```

mp_support_graph('bl_graph_logy') = true;
mp_support_graph('cl_colors') = 'copper';
ff_graph_grid((tb_prob_az{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);% Consumption

```



Asset Mass by Age

```

% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [3,4,5,6,1,2];
% Value Function
tb_prob_aage = ff_summ_nd_array("P(A,Z)", Phi_true, true, ["sum"], 4, 1, cl_mp_datasetdesc, ar_permute);

```

xxx	P(A,Z))	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx						
group	savings	sum_age_18	sum_age_19	sum_age_20	sum_age_21	sum_age_22	sum	
1	0	0.021837	0.0023507	0.0017993	0.0039371	0.0058435	0.	
2	0.00051498	0	0.00039608	0.00037932	0.0011301	0.00066626	0.0	
3	0.0041199	0	0.0020816	0.0019888	0.002009	0.00088325	0.0	
4	0.013905	0	0.0038656	0.0031682	0.001688	0.0011334	0.0	
5	0.032959	0	0.0059678	0.0036757	0.0019686	0.0014691	0.	
6	0.064373	0	0.001968	0.0026857	0.0015598	0.0012805	0.	
7	0.11124	0	0.0010155	0.0010772	0.00089495	0.00094737	0.0	
8	0.17664	0	0.00066497	0.00081578	0.0009608	0.0010548	0.	
9	0.26367	0	0.00045021	0.00085579	0.0011593	0.0011712	0.	
10	0.37542	0	0.00053095	0.0011218	0.0012745	0.0011467	0.	
11	0.51498	0	0.00090691	0.0013663	0.0012758	0.0012278	0.	
12	0.68544	0	0.00097523	0.0011111	0.0010957	0.0011325	0.	
13	0.88989	0	0.00023441	0.00050314	0.00074645	0.0009432	0.	
14	1.1314	0	4.5279e-05	0.00027467	0.00049029	0.00060869	0.0	
15	1.4131	0	1.7339e-05	0.00019476	0.00030104	0.00040391	0.0	
16	1.7381	0	8.1464e-06	6.6555e-05	0.00014925	0.00025602	0.0	
17	2.1094	0	6.1188e-06	3.5994e-05	9.5417e-05	0.000162	0.0	
18	2.5301	0	1.3448e-05	3.7101e-05	7.3464e-05	0.00012006	0.0	
19	3.0034	0	2.2537e-05	4.8195e-05	7.7883e-05	0.00011025	0.0	
20	3.5323	0	2.9909e-05	5.5599e-05	8.0928e-05	0.00010452	0.0	
21	4.1199	0	3.0433e-05	5.458e-05	7.2693e-05	9.1664e-05	0.0	
22	4.7693	0	2.0391e-05	3.7793e-05	5.5429e-05	7.2296e-05	8.9	
23	5.4836	0	5.1199e-06	1.8361e-05	3.277e-05	4.8259e-05	6.4	
24	6.2658	0	7.2528e-07	5.2955e-06	1.4093e-05	2.6887e-05	4.	
25	7.1191	0	1.0524e-07	1.2817e-06	4.9228e-06	1.2149e-05	2.2	
26	8.0466	0	1.7628e-08	5.0295e-07	2.0294e-06	5.2782e-06	1.1	
27	9.0514	0	3.0056e-09	3.0395e-07	1.0911e-06	2.7755e-06	5.7	
28	10.136	0	1.1825e-10	1.6421e-07	5.5086e-07	1.5801e-06	3.2	
29	11.305	0	0	4.8037e-08	2.2122e-07	8.0726e-07	1.8	
30	12.56	0	0	9.2865e-09	6.9448e-08	3.1086e-07	1.0	
31	13.905	0	0	1.789e-09	2.077e-08	9.8086e-08	4.7	
32	15.342	0	0	4.0984e-10	6.2012e-09	3.4485e-08	1.8	
33	16.875	0	0	9.8855e-11	1.6718e-09	1.2956e-08	6.	
34	18.507	0	0	2.1171e-11	4.7002e-10	4.2475e-09	2.1	
35	20.241	0	0	8.4937e-13	1.3772e-10	1.2013e-09	8.	
36	22.08	0	0	0	2.9206e-11	3.623e-10	2.7	
37	24.027	0	0	0	3.6378e-12	1.1269e-10	8.3	
38	26.085	0	0	0	7.7367e-13	2.3608e-11	2.7	
39	28.258	0	0	0	1.7753e-13	3.9993e-12	8.0	
40	30.548	0	0	0	8.3602e-15	1.0518e-12	1.7	
41	32.959	0	0	0	0	1.9415e-13	3.6	
42	35.493	0	0	0	0	1.4615e-14	9.1	
43	38.154	0	0	0	0	2.3455e-15	1.4	
44	40.945	0	0	0	0	2.9499e-16	1.7	
45	43.868	0	0	0	0	6.0398e-18	3.2	
46	46.928	0	0	0	0	0	0	3.3
47	50.126	0	0	0	0	0	...	

```

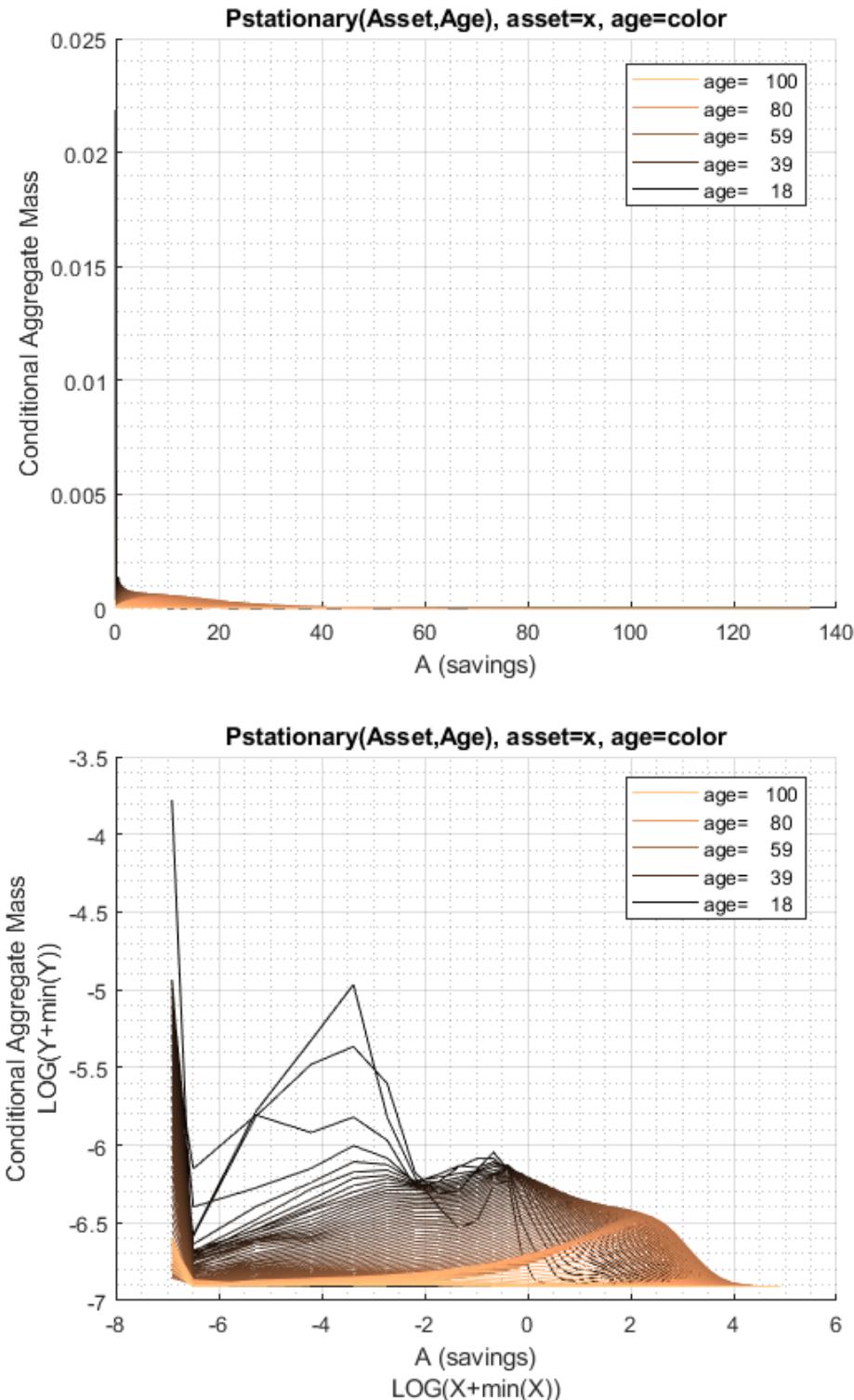
mp_support_graph('cl_st_graph_title') = {'Pstationary(Asset, Age), asset=x, age=color'};
mp_support_graph('cl_st_ytitle') = {'Conditional Aggregate Mass'};
mp_support_graph('cl_st_xtitle') = {'A (savings)'};
mp_support_graph('st_rowvar_name') = 'age=';
mp_support_graph('it_legend_select') = 5;
mp_support_graph('st_rounding') = '6.0f';

```

```

mp_support_graph('bl_graph_logy') = true;
mp_support_graph('cl_colors') = 'copper';
ff_graph_grid((tb_prob_aage{1:end, 3:end})', age_grid, agrid, mp_support_graph);% Consumption Choice

```



6.1.6 Probability Statistics A, C and V Conditional on Ages

Where are the mass at?

```

ap_ss = mp_dsvfi_results('ap_ss');
c_ss = mp_dsvfi_results('cons_ss');

```

```

v_ss = mp_dsvfi_results('v_ss');
n_ss = mp_dsvfi_results('n_ss');

y_head_inc = mp_dsvfi_results('y_head_inc_ss');
y_spouse_inc = mp_dsvfi_results('y_spouse_inc_ss');

yshr_wage = mp_dsvfi_results('yshr_wage_ss');
yshr_SS = mp_dsvfi_results('yshr_SS_ss');
yshr_nttxss = mp_dsvfi_results('yshr_nttxss_ss');

for it_ctr=1:size(ap_ss, 1)
    if (ismember(it_ctr, round(linspace(1, size(ap_ss, 1), 3))))
        display(['age =' num2str(age_grid(it_ctr))]);

        % construct input data
        Phi_true_age = Phi_true(it_ctr, :, :, :, :, :, :);
        ap_ss_age = ap_ss(it_ctr, :, :, :, :, :, :);
        c_ss_age = c_ss(it_ctr, :, :, :, :, :, :);
        v_ss_age = v_ss(it_ctr, :, :, :, :, :, :);
        n_ss_age = n_ss(it_ctr, :, :, :, :, :, :);

        y_head_inc_age = y_head_inc(it_ctr, :, :, :, :, :, :);
        y_spouse_inc_age = y_spouse_inc(it_ctr, :, :, :, :, :, :);
        yshr_wage_age = yshr_wage(it_ctr, :, :, :, :, :, :);
        yshr_SS_age = yshr_SS(it_ctr, :, :, :, :, :, :);
        yshr_nttxss_age = yshr_nttxss(it_ctr, :, :, :, :, :, :);

        mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
        mp_cl_ar_xyz_of_s('ap_ss') = {ap_ss_age(:), zeros(1)};
        mp_cl_ar_xyz_of_s('c_ss') = {c_ss_age(:), zeros(1)};
        mp_cl_ar_xyz_of_s('v_ss') = {v_ss_age(:), zeros(1)};
        mp_cl_ar_xyz_of_s('n_ss') = {n_ss_age(:), zeros(1)};
        mp_cl_ar_xyz_of_s('y_head_inc') = {y_head_inc_age(:), zeros(1)};
        mp_cl_ar_xyz_of_s('y_spouse') = {y_spouse_inc_age(:), zeros(1)};
        mp_cl_ar_xyz_of_s('yshr_wage') = {yshr_wage_age(:), zeros(1)};
        mp_cl_ar_xyz_of_s('yshr_SS') = {yshr_SS_age(:), zeros(1)};
        mp_cl_ar_xyz_of_s('yshr_nttxss') = {yshr_nttxss_age(:), zeros(1)};
        mp_cl_ar_xyz_of_s('ar_st_y_name') = ["ap_ss", "c_ss", "v_ss", "n_ss", ...
            "y_head_inc", "y_spouse", "yshr_wage", "yshr_SS", "yshr_nttxss"];

        % controls
        mp_support = containers.Map('KeyType','char', 'ValueType','any');
        mp_support('ar_fl_percentiles') = [0.01 10 25 50 75 90 99.99];
        mp_support('bl_display_final') = true;
        mp_support('bl_display_detail') = false;
        mp_support('bl_display_drvm2outcomes') = false;
        mp_support('bl_display_drvstats') = false;
        mp_support('bl_display_drvm2covcor') = false;

        % Call Function
        mp_cl_mt_xyz_of_s = ff_simu_stats(Phi_true_age(:)/sum(Phi_true_age,'all'), mp_cl_ar_xyz_of_s
    end
end

age =18
xxx tb_outcomes: all stats xxx
OriginalVariableNames      ap_ss          c_ss          v_ss          n_ss          y_head_inc
-----  -----  -----  -----  -----

```

{'mean'}		0.13166	0.63405	-31.11	1.9854	0.71265
{'unweighted_sum'}		1.0934e+07	8.5358e+05	-2.1835e+06	21	15541
{'sd'}		0.34823	0.37905	29.813	1.0848	0.54567
{'coefofvar'}		2.645	0.59783	-0.95831	0.54639	0.76569
{'gini'}		0.77092	0.31105	-0.47974	0.268	0.36259
{'min'}		0	0.036717	-586.22	1	0.038108
{'max'}		145.07	10.212	24.63	6	13.784
{'pYiso'}		0.10805	0	0	0	0
{'pYls0'}		0	0	0.93414	0	0
{'pYgr0'}		0.89195	1	0.065859	1	1
{'pYisMINY'}		0.10805	1.3288e-05	5.8837e-08	0.41786	2.5312e-05
{'pYisMAXY'}		0	0	0	0.0060544	0
{'p0_01'}		0	0.047727	-322.58	1	0.046651
{'p10'}		0	0.24819	-67.491	1	0.23528
{'p25'}		0.012186	0.36957	-41.871	1	0.35258
{'p50'}		0.032959	0.55272	-24.354	2	0.56523
{'p75'}		0.07477	0.80089	-11.18	3	0.90612
{'p90'}		0.47812	1.1198	-2.6906	4	1.3579
{'p99_99'}		5.4504	3.6593	17.393	6	6.8484
{'fl_cov_ap_ss'}		0.12126	0.055072	2.4507	0.026881	0.05
{'fl_cor_ap_ss'}		1	0.41721	0.23606	0.071158	0.26313
{'fl_cov_c_ss'}		0.055072	0.14368	8.0391	0.07643	0.18689
{'fl_cor_c_ss'}		0.41721	1	0.71138	0.18587	0.90355
{'fl_cov_v_ss'}		2.4507	8.0391	888.8	0.38384	10.004
{'fl_cor_v_ss'}		0.23606	0.71138	1	0.011868	0.61498
{'fl_cov_n_ss'}		0.026881	0.07643	0.38384	1.1768	1.1384e-17
{'fl_cor_n_ss'}		0.071158	0.18587	0.011868	1	1.9231e-17
{'fl_cov_y_head_inc'}		0.05	0.18689	10.004	1.1384e-17	0.29776
{'fl_cor_y_head_inc'}		0.26313	0.90355	0.61498	1.9231e-17	1
{'fl_cov_y_spouse'}		0.18249	0.071644	3.4658	0.13323	0.010455
{'fl_cor_y_spouse'}		0.92021	0.33189	0.20413	0.21565	0.033645
{'fl_cov_yshr_wage'}		1.2236e-32	7.3426e-32	-4.4373e-30	-3.4513e-31	1.4096e-31
{'fl_cor_yshr_wage'}		1.0549e-16	5.8159e-16	-4.4688e-16	-9.5519e-16	7.7559e-16
{'fl_cov_yshr_SS'}		0	0	0	0	0
{'fl_cor_yshr_SS'}		NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}		0.0057457	0.011176	0.85848	0.007516	0.01319
{'fl_cor_yshr_nttxss'}		0.48632	0.86907	0.84874	0.20421	0.71249
{'fracByP0_01'}		0	7.1684e-06	0.0013012	0.21046	7.788e-06
{'fracByP10'}		0	0.030643	0.32088	0.21046	0.027495
{'fracByP25'}		0.0067356	0.10365	0.58193	0.21046	0.092606
{'fracByP50'}		0.04689	0.29058	0.83099	0.53024	0.26377
{'fracByP75'}		0.13162	0.54875	0.97426	0.77109	0.5245
{'fracByP90'}		0.35822	0.76944	1.0077	0.92834	0.74403
{'fracByP99_99'}		0.99575	0.99938	1.0001	1	0.99912
<hr/>						
age =59						
xxx tb_outcomes: all stats xxx						
OriginalVariableNames		ap_ss	c_ss	v_ss	n_ss	y_head_inc
-----	-----	-----	-----	-----	-----	-----
{'mean'}		9.4506	1.2067	-9.9431	1.7239	1.6033
{'unweighted_sum'}		1.1247e+07	1.0819e+06	-3.4419e+05	21	45380
{'sd'}		9.4598	0.76797	14.834	0.90777	1.2742
{'coefofvar'}		1.001	0.63643	-1.4919	0.52659	0.79474
{'gini'}		0.48835	0.32979	-0.78368	0.23461	0.38321
{'min'}		0	0.05663	-208.18	1	0.059541
{'max'}		158.43	12.311	14.965	6	23.47

{'pYis0'}	}	0.0059691	0	0	0	0	0
{'pYls0'}	}	0	0	0.73383	0	0	0
{'pYgro'}	}	0.99403	1	0.26617	1	1	1
{'pYisMINY'}	}	0.0059691	9.8324e-06	2.9687e-09	0.48835	9.8989e-06	
{'pYisMAXY'}	}	9.0457e-09	3.8325e-11	5.2662e-07	0.0036816	1.4683e-06	
{'p0_01'}	}	0	0.07838	-101	1	0.08341	
{'p10'}	}	1.0833	0.41297	-30.14	1	0.49019	
{'p25'}	}	3.0034	0.65765	-16.23	1	0.7717	
{'p50'}	}	6.7818	1.0568	-6.363	2	1.2612	
{'p75'}	}	12.812	1.5534	0.45344	2	2.0256	
{'p90'}	}	20.8	2.1542	4.9139	3	3.0996	
{'p99_99'}	}	112.23	8.4857	13.926	6	15.937	
{'fl_cov_ap_ss'}	}	89.487	6.8831	97.649	0.8159	10.409	
{'fl_cor_ap_ss'}	}	1	0.94746	0.69588	0.095013	0.86354	
{'fl_cov_c_ss'}	}	6.8831	0.58977	8.5503	0.23192	0.85197	
{'fl_cor_c_ss'}	}	0.94746	1	0.75055	0.33267	0.87063	
{'fl_cov_v_ss'}	}	97.649	8.5503	220.04	2.4373	12.623	
{'fl_cor_v_ss'}	}	0.69588	0.75055	1	0.181	0.66782	
{'fl_cov_n_ss'}	}	0.8159	0.23192	2.4373	0.82404	0.055267	
{'fl_cor_n_ss'}	}	0.095013	0.33267	0.181	1	0.04778	
{'fl_cov_y_head_inc'}	}	10.409	0.85197	12.623	0.055267	1.6237	
{'fl_cor_y_head_inc'}	}	0.86354	0.87063	0.66782	0.04778	1	
{'fl_cov_y_spouse'}	}	2.2143	0.24542	3.4887	0.27625	0.116	
{'fl_cor_y_spouse'}	}	0.2103	0.28712	0.21131	0.27342	0.08179	
{'fl_cov_yshr_wage'}	}	-0.54196	-0.036396	-0.86915	0.0011758	-0.038212	
{'fl_cor_yshr_wage'}	}	-0.56735	-0.46933	-0.58024	0.012827	-0.29697	
{'fl_cov_yshr_SS'}	}	0	0	0	0	0	
{'fl_cor_yshr_SS'}	}	NaN	NaN	NaN	NaN	NaN	
{'fl_cov_yshr_nttxss'}	}	0.19452	0.017952	0.42036	0.0075501	0.027003	
{'fl_cor_yshr_nttxss'}	}	0.67266	0.7647	0.92699	0.27208	0.69323	
{'fracByP0_01'}	}	0	6.8812e-06	0.0011212	0.28329	5.8341e-06	
{'fracByP10'}	}	0.004897	0.026408	0.43931	0.28329	0.022426	
{'fracByP25'}	}	0.037048	0.092569	0.77208	0.28329	0.081818	
{'fracByP50'}	}	0.16368	0.27051	1.0414	0.72028	0.23952	
{'fracByP75'}	}	0.41532	0.53706	1.1137	0.72028	0.48823	
{'fracByP90'}	}	0.67288	0.76168	1.075	0.85389	0.72007	
{'fracByP99_99'}	}	0.99866	0.99926	1.0001	1	0.99889	

age =100							
xxx tb_outcomes: all stats xxx							
OriginalVariableNames	ap_ss	c_ss	v_ss	n_ss	y_head_inc	y_s	
-----	-----	-----	-----	-----	-----	-----	-----
{'mean'}	0	0.34868	-3.0033	1.4797	0.2604	0	
{'unweighted_sum'}	0	1.2188e+05	458.94	21	213.14		
{'sd'}	0	0.23392	1.043	0.50567	0.02289	0	
{'coefofvar'}	NaN	0.67088	-0.34728	0.34173	0.087904		
{'gini'}	NaN	0.275	-0.17693	0.12034	0.041151		
{'min'}	0	0.2179	-10.065	1	0.24433		
{'max'}	0	141.66	0.99282	6	5.6926		
{'pYis0'}	1	0	0	0	0	0	
{'pYls0'}	0	0	0.99285	0	0		
{'pYgro'}	0	1	0.0071501	1	1	0	
{'pYisMINY'}	1	0.36483	1.5455e-10	0.5232	0.52813	0	
{'pYisMAXY'}	1	0	0	4.2206e-08	0	1.03	
{'p0_01'}	0	0.2179	-6.3349	1	0.24433		
{'p10'}	0	0.2179	-3.6603	1	0.24433		
{'p25'}	0	0.2179	-3.5892	1	0.24433		

{'p50'}		0	0.25824	-3.5892	1	0.24433	0
{'p75'}		0	0.36458	-2.8095	2	0.29263	0
{'p90'}		0	0.6134	-1.3055	2	0.29279	0
{'p99_99'}		0	2.8989	0.51215	4	0.33789	0
{'fl_cov_ap_ss'}		0	0	0	0	0	0
{'fl_cor_ap_ss'}		NaN	NaN	NaN	NaN	NaN	NaN
{'fl_cov_c_ss'}		0	0.054721	0.19746	0.059476	0.0015551	0
{'fl_cor_c_ss'}		NaN	1	0.80934	0.50281	0.29042	0
{'fl_cov_v_ss'}		0	0.19746	1.0878	0.16711	0.01031	0
{'fl_cor_v_ss'}		NaN	0.80934	1	0.31686	0.43183	0
{'fl_cov_n_ss'}		0	0.059476	0.16711	0.2557	0.0019105	0
{'fl_cor_n_ss'}		NaN	0.50281	0.31686	1	0.16506	0
{'fl_cov_y_head_inc'}		0	0.0015551	0.01031	0.0019105	0.00052397	0.00
{'fl_cor_y_head_inc'}		NaN	0.29042	0.43183	0.16506	1	0
{'fl_cov_y_spouse'}		0	0.05178	0.1649	0.0533	0.00067518	0.
{'fl_cor_y_spouse'}		NaN	0.89356	0.63823	0.4255	0.11907	0
{'fl_cov_yshr_wage'}		0	0.039513	0.15927	0.083913	0.00067571	0.
{'fl_cor_yshr_wage'}		NaN	0.7643	0.69097	0.75087	0.13357	0
{'fl_cov_yshr_SS'}		0	-0.040547	-0.16461	-0.085285	-0.00072523	-0.
{'fl_cor_yshr_SS'}		NaN	-0.77966	-0.70991	-0.75864	-0.14251	-0
{'fl_cov_yshr_nttxss'}		0	0.044511	0.18091	0.091879	0.00087698	0.
{'fl_cor_yshr_nttxss'}		NaN	0.78763	0.71798	0.75212	0.15859	0
{'fracByP0_01'}		NaN	0.22799	0.00053042	0.35357	0.49553	0
{'fracByP10'}		NaN	0.22799	0.22059	0.35357	0.49553	0
{'fracByP25'}		NaN	0.22799	0.6552	0.35357	0.49553	0
{'fracByP50'}		NaN	0.35394	0.6552	0.35357	0.49553	0
{'fracByP75'}		NaN	0.55083	0.87677	0.99419	0.88359	0
{'fracByP90'}		NaN	0.7612	0.97549	0.99419	0.89158	0
{'fracByP99_99'}		NaN	0.99927	1	0.99999	0.99991	0

6.2 Distribution Exact Savings Choices Vectorized

This is the example vignette for function: [snw_ds_main_vec](#) from the [PrjOptiSNW Package](#). This function solves for vfi and gets distribution induced by policy functions and exogenous distributions. Vectorized vfi and distribution methods.

6.2.1 Test SNW_DS_MAIN_VEC

Call the function with testing defaults.

```
mp_params = snw_mp_param('default_docdense');
mp_controls = snw_mp_control('default_test');
mp_controls('bl_print_vfi') = false;
mp_controls('bl_print_ds') = true;
mp_controls('bl_print_ds_verbose') = false;
[Phi_true,Phi_adj,A_agg,Y_inc_agg,it,mp_dsvfi_results] = snw_ds_main_vec(mp_params, mp_controls);
```

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=495.

xx
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

	i	idx	ndim	numel	rowN	colN	sum	mean	std
	-	---	----	-----	----	-----	-----	-----	-----
V_VFI	1	1	6	4.37e+07	83	5.265e+05	-1.5339e+08	-3.5101	26.11
ap_VFI	2	2	6	4.37e+07	83	5.265e+05	1.4159e+09	32.402	36.79
cons_VFI	3	3	6	4.37e+07	83	5.265e+05	2.1402e+08	4.8975	8.329

xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c
	-----	-----	-----	-----	-----	-----	-----	-----	-----
r1	-346.51	-346.12	-343.63	-337.86	-328.51	21.702	21.852	22.003	
r2	-334.38	-333.99	-331.51	-325.83	-316.83	21.724	21.869	22.015	
r3	-322.45	-322.06	-319.6	-314.14	-305.6	21.745	21.885	22.027	
r4	-310.63	-310.27	-307.99	-302.88	-294.87	21.767	21.903	22.041	
r5	-299.94	-299.6	-297.46	-292.67	-285.12	21.775	21.907	22.042	
r79	-9.9437	-9.9325	-9.8557	-9.6597	-9.3232	2.5394	2.5501	2.5602	
r80	-8.9023	-8.8911	-8.8143	-8.6183	-8.2818	2.3039	2.3121	2.3198	
r81	-7.6363	-7.6251	-7.5484	-7.3524	-7.0159	2.0068	2.0124	2.0176	
r82	-5.9673	-5.9561	-5.8793	-5.6833	-5.3468	1.5958	1.5989	1.6018	
r83	-3.5892	-3.578	-3.5012	-3.3052	-2.9687	0.97904	0.98004	0.98097	0

xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c5264
	--	--	-----	-----	-----	-----	-----	-----	-----
r1	0	0	0.0005656	0.0075134	0.022901	114.75	120.41	126.27	132.3
r2	0	0	0.00051498	0.0065334	0.021549	114.86	120.53	126.41	132.5
r3	0	0	0.00051498	0.0049294	0.019875	114.97	120.65	126.56	132.
r4	0	0	0.00051498	0.0047937	0.019672	115.73	121.42	127.34	133.5
r5	0	0	0.00048517	0.0046683	0.019484	116.5	122.21	128.15	134.3
r79	0	0	0	0	0	81.091	85.68	90.335	94.37
r80	0	0	0	0	0	76.669	80.563	84.304	88.0
r81	0	0	0	0	0	68.313	71.534	74.475	77.83
r82	0	0	0	0	0	50.126	53.467	56.953	58.74
r83	0	0	0	0	0	0	0	0	0

xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498
	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.036717	0.037251	0.040426	0.04363	0.048012	9.6491	9.817	9.9649
r2	0.036717	0.037251	0.040477	0.04461	0.049364	9.8118	9.9685	10.101
r3	0.036717	0.037251	0.040477	0.046214	0.051039	9.9779	10.12	10.234
r4	0.038144	0.038678	0.041903	0.047776	0.052666	10.131	10.258	10.354
r5	0.039534	0.040068	0.043323	0.04929	0.054241	10.272	10.384	10.463
r79	0.2179	0.21844	0.22216	0.23228	0.25197	35.858	37.092	38.455
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.253	42.183	44.459
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.587	51.19	54.266
r82	0.2179	0.21844	0.22216	0.23228	0.25197	66.755	69.238	71.77
r83	0.2179	0.21844	0.22216	0.23228	0.25197	116.87	122.69	128.71

SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:1 of 82, time-this-age:0.38533
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:2 of 82, time-this-age:4.8535
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:3 of 82, time-this-age:5.4245
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:4 of 82, time-this-age:5.9426
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:5 of 82, time-this-age:6.1288
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:6 of 82, time-this-age:6.5704
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:7 of 82, time-this-age:6.8969
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:8 of 82, time-this-age:6.9897
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:9 of 82, time-this-age:7.1947
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:10 of 82, time-this-age:8.6388
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:11 of 82, time-this-age:9.0845
 SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:12 of 82, time-this-age:10.3717

SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:13 of 82, time-this-age:10.517
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:14 of 82, time-this-age:10.7579
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:15 of 82, time-this-age:10.8217
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:16 of 82, time-this-age:10.8223
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:17 of 82, time-this-age:10.7043
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:18 of 82, time-this-age:10.7417
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:19 of 82, time-this-age:10.8037
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:20 of 82, time-this-age:10.818
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:21 of 82, time-this-age:10.7822
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:22 of 82, time-this-age:11.9326
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:23 of 82, time-this-age:11.9625
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:24 of 82, time-this-age:11.8969
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:25 of 82, time-this-age:11.7013
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:26 of 82, time-this-age:11.9384
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:27 of 82, time-this-age:12.7336
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:28 of 82, time-this-age:12.6447
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:29 of 82, time-this-age:12.7296
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:30 of 82, time-this-age:12.7955
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:31 of 82, time-this-age:12.6745
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:32 of 82, time-this-age:12.822
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:33 of 82, time-this-age:12.4306
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:34 of 82, time-this-age:12.5625
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:35 of 82, time-this-age:12.4247
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:36 of 82, time-this-age:12.2152
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:37 of 82, time-this-age:12.4463
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:38 of 82, time-this-age:12.409
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:39 of 82, time-this-age:12.2658
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:40 of 82, time-this-age:12.2869
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:41 of 82, time-this-age:12.1216
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:42 of 82, time-this-age:12.1159
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:43 of 82, time-this-age:12.0175
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:44 of 82, time-this-age:11.8754
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:45 of 82, time-this-age:11.8576
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:46 of 82, time-this-age:11.6876
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:47 of 82, time-this-age:11.4529
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:48 of 82, time-this-age:11.2909
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:49 of 82, time-this-age:13.2073
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:50 of 82, time-this-age:14.38
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:51 of 82, time-this-age:14.2335
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:52 of 82, time-this-age:14.3025
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:53 of 82, time-this-age:14.4099
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:54 of 82, time-this-age:14.6607
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:55 of 82, time-this-age:14.3725
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:56 of 82, time-this-age:13.0721
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:57 of 82, time-this-age:14.2288
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:58 of 82, time-this-age:11.5969
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:59 of 82, time-this-age:11.2636
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:60 of 82, time-this-age:11.4644
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:61 of 82, time-this-age:11.153
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:62 of 82, time-this-age:9.8175
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:63 of 82, time-this-age:9.4858
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:64 of 82, time-this-age:8.2039
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:65 of 82, time-this-age:7.8046
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:66 of 82, time-this-age:8.0482
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:67 of 82, time-this-age:7.7332
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:68 of 82, time-this-age:7.8717
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:69 of 82, time-this-age:7.7449
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:70 of 82, time-this-age:7.6917

SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:71 of 82, time-this-age:7.4818
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:72 of 82, time-this-age:7.5373
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:73 of 82, time-this-age:7.477
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:74 of 82, time-this-age:7.0533
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:75 of 82, time-this-age:7.0132
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:76 of 82, time-this-age:6.8244
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:77 of 82, time-this-age:6.5596
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:78 of 82, time-this-age:6.3739
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:79 of 82, time-this-age:6.145
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:80 of 82, time-this-age:5.9883
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:81 of 82, time-this-age:5.6745
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:82 of 82, time-this-age:5.4529
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:1 of 82, time-this-age:0.32995
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:2 of 82, time-this-age:0.046363
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:3 of 82, time-this-age:0.0452
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:4 of 82, time-this-age:0.04461
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:5 of 82, time-this-age:0.046124
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:6 of 82, time-this-age:0.044917
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:7 of 82, time-this-age:0.044963
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:8 of 82, time-this-age:0.045331
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:9 of 82, time-this-age:0.044994
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:10 of 82, time-this-age:0.045092
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:11 of 82, time-this-age:0.045782
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:12 of 82, time-this-age:0.045744
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:13 of 82, time-this-age:0.044755
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:14 of 82, time-this-age:0.045424
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:15 of 82, time-this-age:0.04489
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:16 of 82, time-this-age:0.045122
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:17 of 82, time-this-age:0.044855
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:18 of 82, time-this-age:0.044743
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:19 of 82, time-this-age:0.04457
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:20 of 82, time-this-age:0.042135
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:21 of 82, time-this-age:0.042964
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:22 of 82, time-this-age:0.042523
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:23 of 82, time-this-age:0.041992
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:24 of 82, time-this-age:0.042772
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:25 of 82, time-this-age:0.042253
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:26 of 82, time-this-age:0.042336
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:27 of 82, time-this-age:0.042502
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:28 of 82, time-this-age:0.042069
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:29 of 82, time-this-age:0.042066
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:30 of 82, time-this-age:0.042307
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:31 of 82, time-this-age:0.042301
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:32 of 82, time-this-age:0.042422
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:33 of 82, time-this-age:0.04251
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:34 of 82, time-this-age:0.043235
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:35 of 82, time-this-age:0.041973
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:36 of 82, time-this-age:0.042507
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:37 of 82, time-this-age:0.04212
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:38 of 82, time-this-age:0.041928
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:39 of 82, time-this-age:0.042805
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:40 of 82, time-this-age:0.042282
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:41 of 82, time-this-age:0.042023
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:42 of 82, time-this-age:0.041985
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:43 of 82, time-this-age:0.042461
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:44 of 82, time-this-age:0.042176
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:45 of 82, time-this-age:0.041744
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:46 of 82, time-this-age:0.042401

SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:47 of 82, time-this-age:0.041823
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:48 of 82, time-this-age:0.042776
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:49 of 82, time-this-age:0.042345
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:50 of 82, time-this-age:0.041778
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:51 of 82, time-this-age:0.04279
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:52 of 82, time-this-age:0.042188
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:53 of 82, time-this-age:0.042106
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:54 of 82, time-this-age:0.041468
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:55 of 82, time-this-age:0.04238
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:56 of 82, time-this-age:0.042431
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:57 of 82, time-this-age:0.042291
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:58 of 82, time-this-age:0.042408
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:59 of 82, time-this-age:0.042301
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:60 of 82, time-this-age:0.041831
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:61 of 82, time-this-age:0.041465
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:62 of 82, time-this-age:0.043668
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:63 of 82, time-this-age:0.042077
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:64 of 82, time-this-age:0.041524
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:65 of 82, time-this-age:0.043254
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:66 of 82, time-this-age:0.042202
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:67 of 82, time-this-age:0.041436
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:68 of 82, time-this-age:0.041208
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:69 of 82, time-this-age:0.041526
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:70 of 82, time-this-age:0.042328
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:71 of 82, time-this-age:0.041618
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:72 of 82, time-this-age:0.041941
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:73 of 82, time-this-age:0.04211
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:74 of 82, time-this-age:0.041632
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:75 of 82, time-this-age:0.041663
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:76 of 82, time-this-age:0.042221
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:77 of 82, time-this-age:0.042
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:78 of 82, time-this-age:0.041732
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:79 of 82, time-this-age:0.042972
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:80 of 82, time-this-age:0.04279
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:81 of 82, time-this-age:0.042028
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:82 of 82, time-this-age:0.042981
 SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:83 of 82, time-this-age:0.041914
 SNW_DS_MAIN: Share of population with assets equal to upper bound on asset grid:6.6266e-06
 SNW_DS_MAIN: Accidental bequests are thrown in the ocean
 SNW_DS_MAIN_VEC: Number of a2-adjustments (for taxation) used to balance the government budget= 0
 SNW_DS_MAIN_VEC: Old and updated value of a2=1.5286 1.5286
 SNW_DS_MAIN_VEC: Aggregates: Cons., Gov. cons., Save, Assets, Income, Bequests 48.88966 11.3883
 SNW_DS_MAIN_VEC: Resource constraint: C_t+A_{t+1}+G_t=A_t+Y_t 259.3534 259.3526
 Completed SNW_DS_MAIN_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=869.5447
 xxx tb_outcomes: all stats xxx

OriginalVariableNames	a_ss	ap_ss	cons_ss	n_ss	y_all
{'mean'}	4.2486	4.3473	1.0676	2.3554	1.4672
{'unweighted_sum'}	2228	5.3198e+08	5.0976e+07	21	8.3563e+07
{'sd'}	6.7963	6.834	0.69454	1.4375	1.4636
{'coefofvar'}	1.5996	1.572	0.65055	0.61029	0.99755
{'gini'}	0.68054	0.68147	0.3385	0.3128	0.44353
{'min'}	0	0	0.036717	1	0.038108
{'max'}	135	163.7	141.66	6	50.873
{'pYis0'}	0.1223	0.10225	0	0	0
{'pYls0'}	0	0	0	0	0
{'pYgr0'}	0.8777	0.89775	1	1	1

{'pYisMINY'	}	0.1223	0.10225	8.6094e-07	0.36005	8.6094e-07
{'pYisMAXY'	}	6.6266e-06	2.2031e-12	0	0.041101	2.2031e-12
{'p0_01'	}	0	0	0.066316	1	0.069931
{'p0_1'	}	0	0	0.10404	1	0.11208
{'p1'	}	0	0	0.185	1	0.20291
{'p5'	}	0	0	0.27653	1	0.28
{'p10'	}	0	0	0.35932	1	0.35446
{'p20'	}	0.064373	0.070044	0.49736	1	0.50155
{'p25'	}	0.11124	0.18238	0.56384	1	0.5774
{'p30'	}	0.26367	0.37542	0.63117	1	0.65517
{'p40'	}	0.68544	0.85525	0.77036	2	0.8305
{'p50'	}	1.4131	1.5959	0.92038	2	1.0329
{'p60'	}	2.5301	2.7681	1.086	2	1.2828
{'p70'	}	4.1199	4.5042	1.2802	3	1.6166
{'p75'	}	5.4836	5.7337	1.3961	3	1.8353
{'p80'	}	7.1191	7.2583	1.5324	4	2.1144
{'p90'	}	12.56	12.145	1.9406	5	3.0534
{'p95'	}	16.875	17.551	2.3509	5	4.0423
{'p99'	}	30.548	31.57	3.4062	6	6.9038
{'p99_9'	}	56.953	57.547	5.2994	6	14.807
{'p99_99'	}	90.439	90.439	7.5689	6	21.018
{'fl_cov_a_ss'	}	46.189	46.181	3.4533	-1.4145	4.552
{'fl_cor_a_ss'	}	1	0.99429	0.73158	-0.14479	0.45762
{'fl_cov_ap_ss'	}	46.181	46.704	3.5562	-1.3778	5.3963
{'fl_cor_ap_ss'	}	0.99429	1	0.74922	-0.14025	0.53951
{'fl_cov_cons_ss'	}	3.4533	3.5562	0.48239	0.23887	0.77192
{'fl_cor_cons_ss'	}	0.73158	0.74922	1	0.23926	0.75937
{'fl_cov_n_ss'	}	-1.4145	-1.3778	0.23887	2.0664	0.36008
{'fl_cor_n_ss'	}	-0.14479	-0.14025	0.23926	1	0.17115
{'fl_cov_y_all'	}	4.552	5.3963	0.77192	0.36008	2.1421
{'fl_cor_y_all'	}	0.45762	0.53951	0.75937	0.17115	1
{'fl_cov_y_head_inc'	}	3.9111	4.1925	0.5697	0.092861	1.1244
{'fl_cor_y_head_inc'	}	0.57021	0.60786	0.81274	0.064008	0.7612
{'fl_cov_y_head_earn'	}	1.9006	2.2089	0.43359	0.19345	0.97967
{'fl_cor_y_head_earn'	}	0.30133	0.34828	0.67268	0.14501	0.72126
{'fl_cov_y_spouse_inc'	}	0.64085	1.2037	0.20222	0.26722	1.0177
{'fl_cor_y_spouse_inc'	}	0.098743	0.18445	0.3049	0.19466	0.72817
{'fl_cov_yshr_interest'	}	0.77027	0.72501	0.038411	-0.066855	-0.0092638
{'fl_cor_yshr_interest'	}	0.67438	0.63125	0.32907	-0.27673	-0.037662
{'fl_cov_yshr_wage'	}	-0.77963	-0.69253	-0.0045925	0.17059	0.10778
{'fl_cor_yshr_wage'	}	-0.3398	-0.30018	-0.019587	0.35152	0.21813
{'fl_cov_yshr_SS'	}	0.0093601	-0.032478	-0.033818	-0.10373	-0.098514
{'fl_cor_yshr_SS'	}	0.0058267	-0.020106	-0.206	-0.3053	-0.28477
{'fl_cov_yshr_tax'	}	0.099405	0.11025	0.018741	0.01336	0.038806
{'fl_cor_yshr_tax'	}	0.41564	0.45846	0.76677	0.26411	0.75345
{'fl_cov_yshr_nttxss'	}	0.090044	0.14273	0.052559	0.11709	0.13732
{'fl_cor_yshr_nttxss'	}	0.05183	0.081703	0.29603	0.31865	0.36703
{'fracByP0_01'	}	0	0	5.4315e-06	0.15286	4.2545e-06
{'fracByP0_1'	}	0	0	8.2399e-05	0.15286	6.3218e-05
{'fracByP1'	}	0	0	0.0013761	0.15286	0.0010863
{'fracByP5'	}	0	0	0.01022	0.15286	0.007951
{'fracByP10'	}	0	0	0.025187	0.15286	0.018765
{'fracByP20'	}	0.00074359	0.00061895	0.065443	0.15286	0.04804
{'fracByP25'	}	0.0014042	0.0020497	0.090297	0.15286	0.066483
{'fracByP30'	}	0.0041483	0.0051966	0.11827	0.15286	0.08755
{'fracByP40'	}	0.01665	0.018812	0.18387	0.40183	0.13814
{'fracByP50'	}	0.045047	0.046332	0.26291	0.40183	0.20163
{'fracByP60'	}	0.094899	0.095657	0.35672	0.40183	0.27993

```

{'fracByP70'      }    0.17353    0.17824    0.46729    0.56321    0.37814
{'fracByP75'      }    0.24359    0.23689    0.52992    0.56321    0.43683
{'fracByP80'      }    0.32643    0.31096    0.59841    0.75407    0.50391
{'fracByP90'      }    0.56332    0.5274     0.75898    0.8953     0.67588
{'fracByP95'      }    0.69729    0.69468    0.85836    0.8953     0.79474
{'fracByP99'      }    0.90278    0.90216    0.96058    1          0.93117
{'fracByP99_9'    }    0.98493    0.98362    0.99414    1          0.98801
{'fracByP99_99'   }    0.99793    0.99759    0.99921    1          0.99841

% [Phi_true,Phi_adj] = snw_ds_main(mp_params, mp_controls);
Phi_true = Phi_true/sum(Phi_true(:));

```

6.2.2 Show All Info in mp_dsvfi_results

```

mp_cl_mt_xyz_of_s = mp_dsvfi_results('mp_cl_mt_xyz_of_s');
disp(mp_cl_mt_xyz_of_s('tb_outcomes'))

```

	mean	unweighted_sum	sd	coeofvar	gini	min	
a_ss	4.2486	2228	6.7963	1.5996	0.68054	0	
ap_ss	4.3473	5.3198e+08	6.834	1.572	0.68147	0	
cons_ss	1.0676	5.0976e+07	0.69454	0.65055	0.3385	0.036717	1
n_ss	2.3554	21	1.4375	0.61029	0.3128	1	
y_all	1.4672	8.3563e+07	1.4636	0.99755	0.44353	0.038108	5
y_head_inc	1.1087	1.9253e+06	1.0092	0.91029	0.41889	0.038108	2
y_head_earn	0.88655	19732	0.92804	1.0468	0.53121	0	1
y_spouse_inc	0.35849	4.8273e+05	0.95494	2.6638	0.85255	0	2
yshr_interest	0.12214	3.8429e+06	0.16806	1.3759	0.66002	0	0.
yshr_wage	0.77513	8.8876e+06	0.33759	0.43553	0.2056	0	
yshr_SS	0.10273	30336	0.23637	2.3009	0.91226	0	
yshr_tax	0.17862	2.8339e+06	0.03519	0.19701	0.11226	0.036506	0
yshr_nttxss	0.075896	2.8036e+06	0.25563	3.3681	1.3974	-0.89184	0

6.3 Distribution Grid Search

This is the example vignette for function: `snw_ds_main_grid_search` from the **PrjOptiSNW Package**. This function solves for vfi and gets distribution induced by policy functions and exogenous distributions. Grid Search for VFI and Grid Search also for Distribution. The results are illustrative of the differences between using grid search and exact solution. The grid search solution here is not fully vectorized but loops over the state-space.

6.3.1 Test SNW_DS_MAIN_GRID_SEARCH Defaults More Dense

Rather than solving for "docdense", this solves for "moredense", which has fewer shocks, in order to save time given the relatively slow speed of this algorithm.

```

mp_params = snw_mp_param('default_moredense');
mp_controls = snw_mp_control('default_test');
mp_controls('bl_print_vfi') = false;
mp_controls('bl_print_ds') = false;
mp_controls('bl_print_ds_verbose') = false;
[Phi_true,Phi_adj,A_agg,Y_inc_agg,it,mp_dsvfi_results] = snw_ds_main_grid_search(mp_params, mp_contr

```

Elapsed time is 11762.574665 seconds.

Completed SNW_VFI_MAIN_GRID_SEARCH;SNW_MP_PARAM=default_moredense;SNW_MP_CONTROL=default_test

Elapsed time is 12505.621399 seconds.

Completed SNW_DS_MAIN;SNW_MP_PARAM=;default_moredense;SNW_MP_CONTROL=;default_test

```
Phi_true = Phi_true/sum(Phi_true(:));
```

6.3.2 Show All Info in mp_dsvfi_results More Dense

```
mp_cl_mt_xyz_of_s = mp_dsvfi_results('mp_cl_mt_xyz_of_s');
disp(mp_cl_mt_xyz_of_s('tb_outcomes'))
```

	mean	unweighted_sum	sd	coefofvar	gini	min	-
	-----	-----	-----	-----	-----	-----	-
a_ss	4.1966	5130.2	8.2211	1.959	0.74586	0	
ap_ss	33.417	11476	25.564	0.765	0.44091	1	
cons_ss	1.1837	1.59e+07	1.0186	0.86052	0.40734	0.035637	
v_ss	-19.282	-9.477e+06	35.18	-1.8245	-0.7793	-867.32	
n_ss	2.3554	21	1.4375	0.61029	0.3128	1	
y_all	1.6288	2.398e+07	1.8953	1.1636	0.49934	0.038108	
y_head_inc	1.2693	5.6172e+05	1.541	1.2141	0.50187	0.038108	
y_head_earn	1.0492	2628.2	1.4242	1.3574	0.60462	0	
y_spouse_inc	0.35948	55577	0.96095	2.6732	0.85293	0	
yshr_interest	0.10937	1.0949e+06	0.1698	1.5525	0.711	0	
yshr_wage	0.78519	2.3994e+06	0.34085	0.43409	0.19417	0	
yshr_SS	0.10544	70381	0.24571	2.3303	0.91374	0	
yshr_tax	0.17729	7.7889e+05	0.040058	0.22594	0.12851	0.036506	
yshr_nttxss	0.071855	7.0851e+05	0.26576	3.6986	1.5402	-0.89184	

6.3.3 More Dense Param Results Define Frames

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

```
% Grids:
age_grid = 18:100;
agrid = mp_params('agrid');
eta_H_grid = mp_params('eta_H_grid');
eta_S_grid = mp_params('eta_S_grid');
ar_st_eta_HS_grid = string(cellstr([num2str(eta_H_grid, 'hz=%3.2f;'), num2str(eta_S_grid, 'wz=%3.2f;'), 'nz=%3.2f;']));
edu_grid = [0,1];
marry_grid = [0,1];
kids_grid = (1:1:mp_params('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});
```

6.3.4 Analyze Probability Mass Along Age Dimensions

Where are the mass at? Analyze mass given state space components.

```
% Get the Joint distribution over all states
% Define Graph Inputs
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = false; % do not log
```

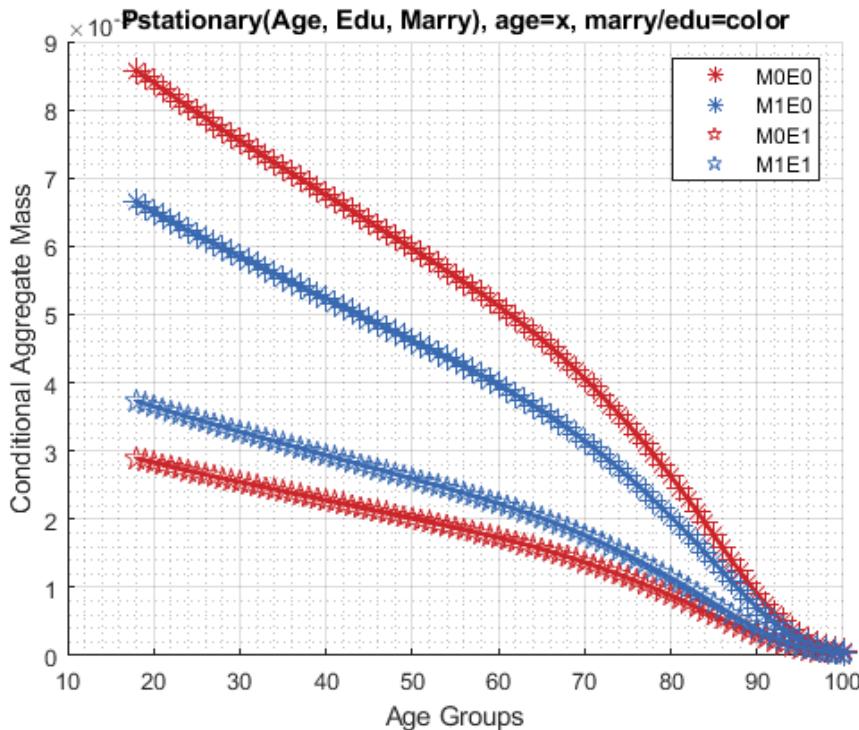
Exogenous Permanent States Mass: Life Cycle, Edu and Marraige

Tabulate value and policies along savings and shocks:

```
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,6,1,5,4];
% Value Function
tb_prob_aem = ff_summ_nd_array("P(Age, EDU, MARRY)", Phi_true, true, ["sum"], 3, 1, cl_mp_datasetde

xxx P(Age, EDU, MARRY)) xxxxxxxxxxxxxxxxxxxxxxxxx
group marry edu sum_age_18 sum_age_19 sum_age_20 sum_age_21 sum_age_22 s
---- ---- --- ----- ----- ----- ----- ----- -----
1 0 0 0.0085768 0.0084866 0.0083969 0.0083078 0.0082194 0
2 1 0 0.0066438 0.0065739 0.0065044 0.0064354 0.0063669 0
3 0 1 0.0028875 0.0028571 0.002827 0.002797 0.0027672 0
4 1 1 0.0037292 0.0036899 0.0036509 0.0036122 0.0035738 0

mp_support_graph('cl_st_graph_title') = {'Pstationary(Age, Edu, Marry), age=x, marry/edu=color'};
mp_support_graph('cl_st_ytitle') = {'Conditional Aggregate Mass'};
ar_row_grid = ["M0E0", "M1E0", "M0E1", "M1E1"];
mp_support_graph('cl_st_xtitle') = {'Age Groups'};
mp_support_graph('cl_scatter_shapes') = {'*', '*', 'p', 'p' };
mp_support_graph('cl_colors') = {'red', 'blue', 'red', 'blue'};
ff_graph_grid((tb_prob_aem{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Kids and Marry By Age Mass

```
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,4,1,6,5];
% Value Function
tb_prob_amarrykids = ff_summ_nd_array("P(Age, Kids, Marry)", Phi_true, true, ["sum"], 3, 1, cl_mp_d

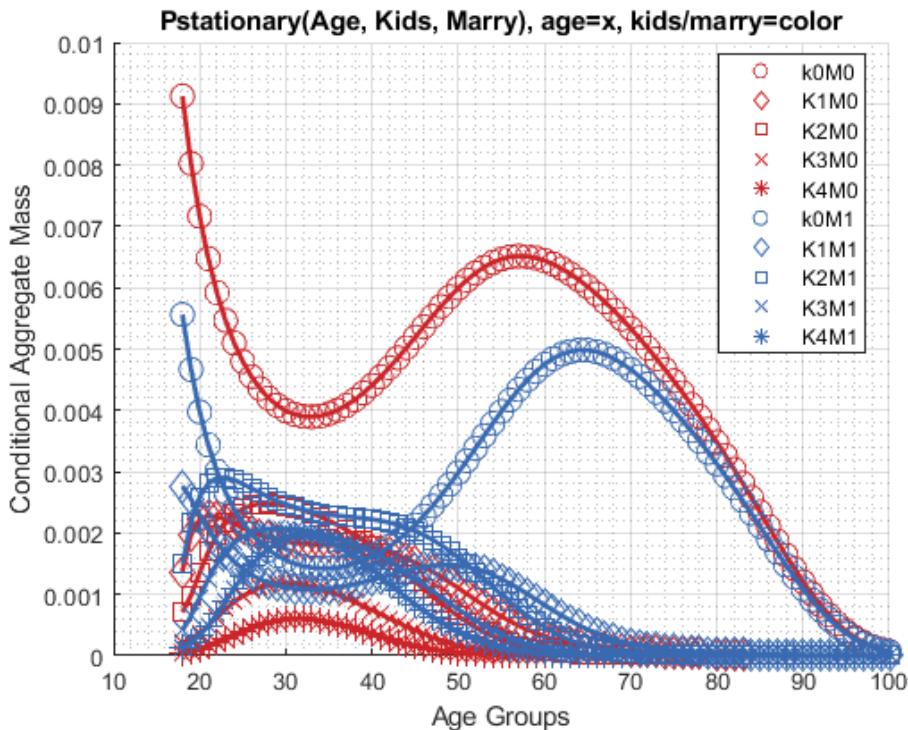
xxx P(Age, Kids, Marry)) xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry sum_age_18 sum_age_19 sum_age_20 sum_age_21 sum_age_22
---- ---- ----- ----- ----- ----- ----- ----- -----
```

1	1	0	0.0091249	0.0080278	0.0071652	0.0064765	0.0059205
2	2	0	0.0013699	0.0019743	0.0022187	0.0022858	0.0022687
3	3	0	0.00071266	0.00098425	0.0013537	0.0016929	0.0019639
4	4	0	0.00020622	0.00027865	0.00037326	0.00049476	0.00062818
5	5	0	5.0761e-05	7.8715e-05	0.000113	0.00015485	0.00020534
6	1	1	0.0055624	0.0046679	0.0039774	0.0034368	0.0030088
7	2	1	0.0027682	0.0025539	0.0023005	0.0020611	0.0018525
8	3	1	0.0014982	0.0021823	0.0025943	0.0028096	0.002896
9	4	1	0.00041197	0.00064648	0.00095224	0.0012491	0.0015009
10	5	1	0.00013221	0.0002132	0.00033097	0.00049097	0.00068255

```

mp_support_graph('cl_st_graph_title') = {'Pstationary(Age, Kids, Marry), age=x, kids/marry=color'};
mp_support_graph('cl_st_ytitle') = {'Conditional Aggregate Mass'};
ar_row_grid = [...
    "k0M0", "K1M0", "K2M0", "K3M0", "K4M0", ...
    "k0M1", "K1M1", "K2M1", "K3M1", "K4M1"];
mp_support_graph('cl_scatter_shapes') = {...
    'o', 'd', 's', 'x', '*', ...
    'o', 'd', 's', 'x', '*'};
mp_support_graph('cl_colors') = {...
    'red', 'red', 'red', 'red', 'red',...
    'blue', 'blue', 'blue', 'blue', 'blue'};;
mp_support_graph('cl_st_xtitle') = {'Age Groups'};
ff_graph_grid((tb_prob_amarrykids{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);

```



6.3.5 Analyze Probability Mass Asset and Shock Dimensions

Where are the mass at?

```
% Define Graph Inputs
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = false; % do not log
```

Asset and Shock Mass

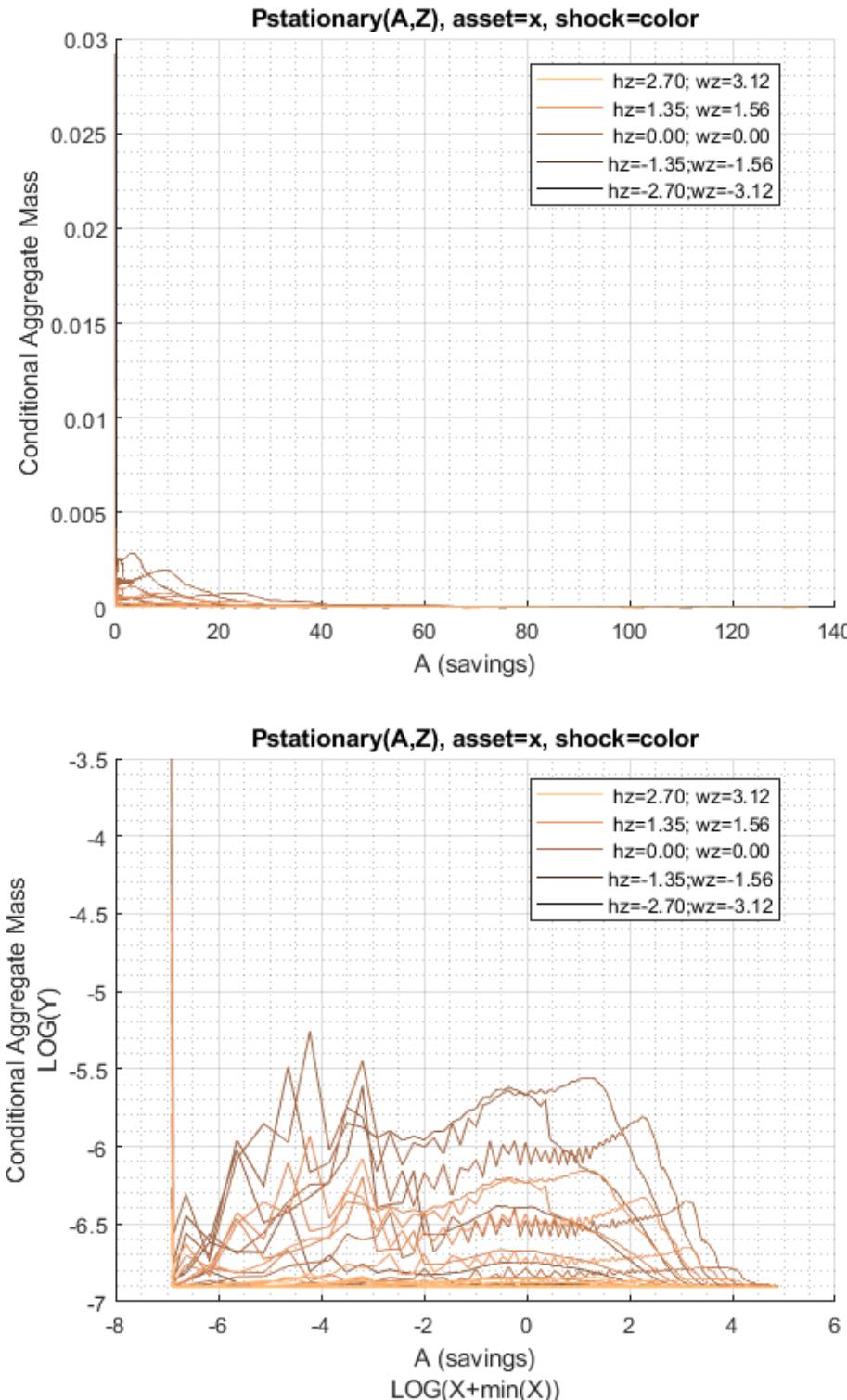
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid); ar_permute = [1,4,5,6,3,2]; % Value Function tb_prob_az = ff_summ_nd_array("P(A,Z)", Phi_true, true, ["sum"], 4, 1, cl_mp_datasetdesc, ar_permut							
xxx	P(A,Z))	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx					
group	savings	sum_eta_1	sum_eta_2	sum_eta_3	sum_eta_4	sum_eta_5	sum_eta
----	-----	-----	-----	-----	-----	-----	-----
1	0	1.7781e-05	0.00011464	0.00040781	0.00065248	0.00059124	0.0003
2	4e-05	2.8722e-07	1.1649e-06	3.9632e-06	1.1727e-06	9.1594e-06	8.6911
3	0.00032	8.5865e-07	2.0949e-06	1.3074e-05	9.3326e-06	1.8355e-05	2.7109
4	0.00108	2.4439e-06	7.4985e-06	7.825e-06	5.1658e-06	4.2511e-06	9.0564
5	0.00256	7.4917e-07	5.7803e-06	3.1919e-05	3.5332e-05	2.8844e-05	5.4161
6	0.005	1.6199e-07	5.684e-06	1.1553e-05	2.0567e-05	4.1715e-05	9.3727
7	0.00864	2.9061e-07	1.562e-05	1.4073e-05	7.0288e-05	3.462e-05	1.6548
8	0.01372	9.5464e-08	2.3479e-06	1.7752e-05	2.4581e-05	9.4236e-05	2.0967
9	0.02048	1.4979e-07	5.1146e-06	2.195e-05	2.7505e-05	3.1649e-05	2.1267
10	0.02916	2.2894e-07	2.3319e-06	2.9711e-05	4.1965e-05	4.8965e-05	4.931
11	0.04	3.76e-07	3.6133e-06	5.9345e-05	4.0368e-05	4.4556e-05	7.3962
12	0.05324	2.756e-07	2.2346e-06	1.4966e-05	3.6227e-05	2.4755e-05	3.695
13	0.06912	3.3888e-07	2.6932e-06	1.5812e-05	3.8986e-05	3.8211e-05	1.648
14	0.08788	3.0263e-07	2.2683e-06	1.6049e-05	3.5262e-05	2.667e-05	2.515
15	0.10976	2.6825e-07	2.1043e-06	2.4986e-05	3.6843e-05	3.4559e-05	1.5405
16	0.135	2.768e-07	1.8377e-06	9.0408e-06	3.5423e-05	3.2867e-05	2.4479
17	0.16384	2.8181e-07	1.9353e-06	9.257e-06	3.8786e-05	3.6177e-05	1.9607
18	0.19652	3.067e-07	2.0467e-06	9.1227e-06	3.8618e-05	3.0376e-05	2.7065
19	0.23328	3.3018e-07	2.1755e-06	1.0247e-05	4.2533e-05	4.2068e-05	1.8014
20	0.27436	3.6009e-07	2.3328e-06	1.0941e-05	4.3919e-05	3.3639e-05	2.5977
21	0.32	4.1186e-07	2.5895e-06	1.1659e-05	4.7371e-05	4.4252e-05	2.4973
22	0.37044	4.4759e-07	2.8965e-06	1.2583e-05	4.8243e-05	4.0516e-05	2.8508
23	0.42592	4.7723e-07	3.3135e-06	1.3374e-05	5.3569e-05	4.9866e-05	2.3551
24	0.48668	5.296e-07	3.4623e-06	1.4309e-05	5.3859e-05	5.1087e-05	3.5022
25	0.55296	5.459e-07	3.6382e-06	1.5329e-05	5.7267e-05	5.3924e-05	2.6534
26	0.625	5.615e-07	3.878e-06	1.5435e-05	5.7034e-05	5.5588e-05	3.3803
27	0.70304	5.616e-07	3.8405e-06	1.5148e-05	5.8804e-05	5.7189e-05	3.0223
28	0.78732	5.8141e-07	3.7688e-06	1.5044e-05	5.7591e-05	5.4784e-05	3.5161
29	0.87808	5.8397e-07	3.8463e-06	1.504e-05	5.6538e-05	5.6164e-05	2.6669
30	0.97556	5.7697e-07	3.9047e-06	1.4901e-05	5.5173e-05	5.4358e-05	3.4721
31	1.08	5.7655e-07	3.8874e-06	1.5177e-05	5.445e-05	5.7049e-05	2.7157
32	1.1916	5.6606e-07	3.778e-06	1.4865e-05	5.185e-05	5.5565e-05	3.2554
33	1.3107	5.5291e-07	3.7261e-06	1.43e-05	4.9523e-05	5.7531e-05	2.8096
34	1.4375	5.3074e-07	3.574e-06	1.3682e-05	5.2445e-05	5.5479e-05	3.1707
35	1.5722	5.0996e-07	3.497e-06	1.3373e-05	3.5566e-05	5.7462e-05	2.7077
36	1.715	5.0049e-07	3.3282e-06	1.3028e-05	3.4521e-05	5.6522e-05	3.2615
37	1.8662	4.7974e-07	3.329e-06	1.2601e-05	3.3434e-05	5.8509e-05	2.6878
38	2.0261	4.596e-07	3.1609e-06	1.2343e-05	3.178e-05	5.9485e-05	3.2101
39	2.1949	4.4954e-07	3.105e-06	1.2095e-05	3.1287e-05	5.9761e-05	2.7128
40	2.3728	4.1729e-07	3.0323e-06	1.186e-05	3.0175e-05	6.1927e-05	3.2379
41	2.56	3.9929e-07	2.924e-06	1.1544e-05	2.9921e-05	6.1827e-05	2.7425
42	2.7568	3.8414e-07	2.7951e-06	1.1251e-05	2.6814e-05	6.3135e-05	3.2763
43	2.9635	3.616e-07	2.7007e-06	1.0868e-05	2.5813e-05	6.3482e-05	2.7626
44	3.1803	3.3481e-07	2.5593e-06	1.0429e-05	2.5595e-05	6.3992e-05	3.3047
45	3.4074	3.131e-07	2.4198e-06	9.99e-06	2.4766e-05	6.3343e-05	2.858
46	3.645	2.9457e-07	2.2754e-06	9.6582e-06	2.4476e-05	6.3967e-05	3.3608
47	3.8934	2.7703e-07	2.1293e-06	9.1931e-06	2.3981e-05	6.2378e-05	3.2136
48	4.1529	2.515e-07	2.018e-06	8.6923e-06	2.3738e-05	6.0398e-05	3.4717
49	4.4237	2.3412e-07	1.8599e-06	8.0926e-06	2.2417e-05	5.8532e-05	3.3219

50	4.706	2.1348e-07	1.7011e-06	7.6231e-06	2.1465e-05	5.6363e-05	3.5656
51	5	1.9593e-07	1.5641e-06	7.1764e-06	2.0854e-05	5.2743e-05	3.4502
52	5.306	1.7768e-07	1.4581e-06	6.7963e-06	2.007e-05	4.821e-05	3.7676
53	5.6243	1.5982e-07	1.3264e-06	6.2348e-06	1.9171e-05	4.3737e-05	3.5788
54	5.9551	1.4334e-07	1.204e-06	5.8483e-06	1.8296e-05	4.106e-05	3.8181
55	6.2986	1.3188e-07	1.1011e-06	5.4121e-06	1.7322e-05	3.901e-05	3.7324
56	6.655	1.1797e-07	9.977e-07	4.9804e-06	1.6132e-05	3.7093e-05	4.0152
57	7.0246	1.0623e-07	9.1605e-07	4.7007e-06	1.5537e-05	3.4804e-05	4.0289
58	7.4077	9.4398e-08	8.3453e-07	4.3022e-06	1.4566e-05	3.2665e-05	4.1753
59	7.8045	8.2422e-08	7.5244e-07	3.9469e-06	1.3777e-05	3.0198e-05	4.183
60	8.2152	7.1784e-08	6.6939e-07	3.6212e-06	1.3091e-05	2.7445e-05	4.309
61	8.64	6.1804e-08	5.8987e-07	3.2784e-06	1.2179e-05	2.5843e-05	4.2847
62	9.0792	5.3502e-08	5.1823e-07	2.9635e-06	1.1462e-05	2.4729e-05	4.4278
63	9.5331	4.5477e-08	4.5311e-07	2.6656e-06	1.0485e-05	2.3755e-05	4.4697
64	10.002	3.8449e-08	3.8904e-07	2.358e-06	9.5066e-06	2.227e-05	4.4247
65	10.486	3.2576e-08	3.3716e-07	2.0726e-06	8.8323e-06	2.1148e-05	4.3231
66	10.985	2.7144e-08	2.8859e-07	1.805e-06	8.1101e-06	1.9267e-05	4.0181
67	11.5	2.234e-08	2.4458e-07	1.5627e-06	7.3458e-06	1.7795e-05	3.8372
68	12.031	1.8426e-08	2.0804e-07	1.3548e-06	6.7189e-06	1.6482e-05	3.5562
69	12.577	1.5109e-08	1.7304e-07	1.1665e-06	6.035e-06	1.5039e-05	3.3252
70	13.14	1.2136e-08	1.4416e-07	9.8771e-07	5.3216e-06	1.3797e-05	2.8982
71	13.72	9.7439e-09	1.1717e-07	8.4655e-07	4.7591e-06	1.2593e-05	2.5621
72	14.316	7.6519e-09	9.5696e-08	7.1116e-07	4.1025e-06	1.1341e-05	2.5829
73	14.93	6.0255e-09	7.71e-08	5.9381e-07	3.5724e-06	1.0484e-05	2.4658
74	15.561	4.7503e-09	6.2213e-08	4.9108e-07	3.1215e-06	9.5178e-06	2.3245
75	16.209	3.7139e-09	4.928e-08	4.0026e-07	2.6199e-06	8.2935e-06	2.1991
76	16.875	2.945e-09	3.8866e-08	3.2717e-07	2.258e-06	7.5498e-06	1.9996
77	17.559	2.3042e-09	3.075e-08	2.6267e-07	1.8986e-06	6.5071e-06	1.9146
78	18.261	1.7888e-09	2.4653e-08	2.1261e-07	1.6083e-06	5.7887e-06	1.8096
79	18.982	1.3465e-09	1.9495e-08	1.7008e-07	1.3129e-06	5.0003e-06	1.5323
80	19.722	9.9583e-10	1.5366e-08	1.3569e-07	1.0922e-06	4.3544e-06	1.2912
81	20.48	7.1218e-10	1.1848e-08	1.0869e-07	9.0647e-07	3.715e-06	1.1136
82	21.258	5.1489e-10	8.9408e-09	8.6675e-08	7.3075e-07	3.2426e-06	1.018
83	22.055	3.7141e-10	6.6969e-09	6.8391e-08	5.9709e-07	2.8121e-06	9.0229
84	22.871	2.7136e-10	4.8427e-09	5.3798e-08	4.7428e-07	2.4448e-06	8.136
85	23.708	1.9274e-10	3.4542e-09	4.184e-08	3.8443e-07	2.0544e-06	7.4782
86	24.565	1.3871e-10	2.4987e-09	3.1083e-08	3.0722e-07	1.6959e-06	6.7652
87	25.442	9.9269e-11	1.8401e-09	2.3394e-08	2.4072e-07	1.4171e-06	6.0563
88	26.34	7.0282e-11	1.338e-09	1.7154e-08	1.9419e-07	1.1731e-06	5.8143

```

mp_support_graph('cl_st_graph_title') = {'Pstationary(A,Z), asset=x, shock=color'};
mp_support_graph('cl_st_ytitle') = {'Conditional Aggregate Mass'};
mp_support_graph('cl_st_xtitle') = {'A (savings)'};
mp_support_graph('st_rowvar_name') = 'z=';
mp_support_graph('it_legend_select') = 5;
mp_support_graph('st_rounding') = '6.2f';
mp_support_graph('bl_graph_logy') = true;
mp_support_graph('cl_colors') = 'copper';
ff_graph_grid((tb_prob_az{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);% Consumption

```



Asset Mass by Age

```
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [3,4,5,6,1,2];
% Value Function
tb_prob_aage = ff_summ_nd_array("P(A,Z)", Phi_true, true, ["sum"], 4, 1, cl_mp_datasetdesc, ar_permute);

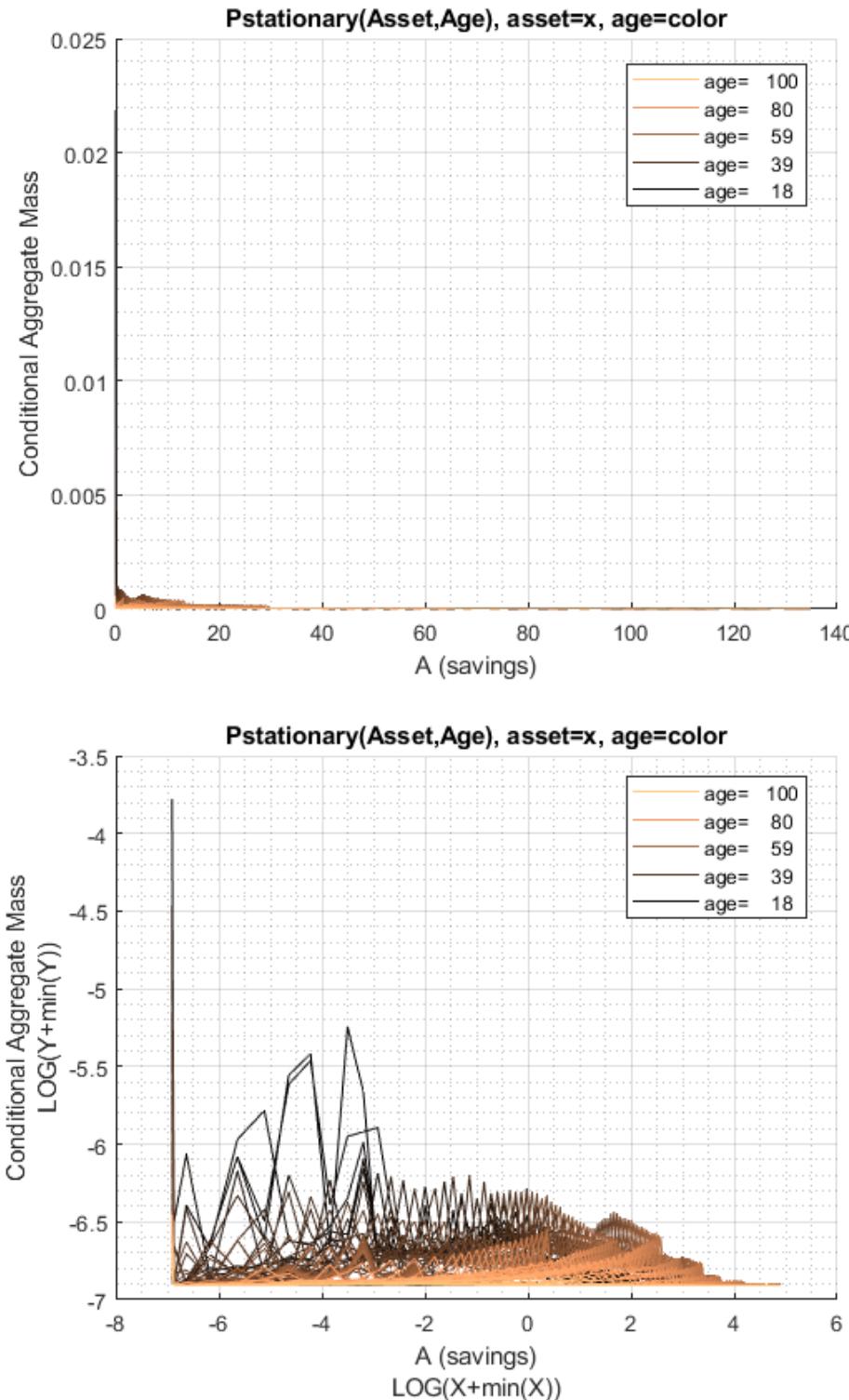
xxx P(A,Z) xxxxxxxxxxxxxxxxxxxxxxxxx
group      savings      sum_age_18      sum_age_19      sum_age_20      sum_age_21      sum_age_22      sum_ag...
```

1	0	0.021837	0.002388	0.0018389	0.006441	0.0087965	0.01
2	4e-05	0	2.3862e-06	2.8257e-06	1.5227e-05	0.0005064	7.0852
3	0.00032	0	3.749e-05	3.8393e-05	0.00067452	0.0013201	3.5849
4	0.00108	0	0.00031485	0.0003134	0.00027518	6.9704e-05	0.0001
5	0.00256	0	0.0012853	0.0012851	0.0015569	8.2105e-05	0.0001
6	0.005	0	0.00034215	0.00051426	0.0020794	0.00015795	0.0001
7	0.00864	0	0.0028722	0.0026464	0.00033471	0.00033022	0.0001
8	0.01372	0	0.003431	0.003249	0.00029554	0.00030632	0.0001
9	0.02048	0	0.00028503	0.00067599	0.00046576	0.00041506	0.0003
10	0.02916	0	0.004274	0.0016076	0.00075151	0.00038657	0.0003
11	0.04	0	0.0024741	0.0016863	0.0015147	0.0012561	0.001
12	0.05324	0	0.00012193	0.0017565	0.00025806	0.00022971	0.0002
13	0.06912	0	0.00044563	0.00062939	0.00029172	0.00029386	0.0002
14	0.08788	0	2.7692e-05	0.00011258	0.00016217	0.00018796	0.0002
15	0.10976	0	6.2377e-05	8.9179e-06	7.302e-05	0.00017801	0.0001
16	0.135	0	0.00067668	0.00016485	0.00010669	0.000221	0.0002
17	0.16384	0	5.8231e-06	5.1096e-05	0.00019395	0.00024128	0.0000
18	0.19652	0	3.2338e-05	4.7486e-05	0.00021219	0.000234	0.0002
19	0.23328	0	2.7827e-05	0.00062962	0.00032249	0.00035572	0.0002
20	0.27436	0	3.3098e-06	0.00012226	0.00073141	0.00035073	0.0003
21	0.32	0	4.0326e-05	0.00029658	0.00038943	0.00026142	0.0003
22	0.37044	0	0.00023294	0.00034328	0.00045557	0.00074931	0.0003
23	0.42592	0	0.00029162	0.00046139	0.0003154	0.00034178	0.0003
24	0.48668	0	0.0002901	0.00049107	0.00051926	0.00039455	0.0006
25	0.55296	0	0.00034886	0.00054566	0.00036044	0.00041009	0.0004
26	0.625	0	0.00050916	0.00043446	0.00028992	0.00033921	0.0003
27	0.70304	0	0.00039586	0.00037772	0.00035949	0.00035245	0.0003
28	0.78732	0	0.00020681	0.00035133	0.00037856	0.00031646	0.0003
29	0.87808	0	1.4297e-05	5.5411e-05	0.00015549	0.0003353	0.0002
30	0.97556	0	1.5592e-05	6.2891e-05	0.00029115	0.00021838	0.0002
31	1.08	0	2.009e-06	8.4112e-05	0.000141	0.00019198	0.0002
32	1.1916	0	2.1045e-05	0.00010104	0.00015492	0.00012578	0.0001
33	1.3107	0	1.4435e-06	6.9531e-05	5.1206e-05	0.0002098	0.0001
34	1.4375	0	5.1689e-07	4.651e-05	7.8499e-05	7.7448e-05	0.0001
35	1.5722	0	4.7793e-07	4.9348e-06	2.019e-05	8.9748e-05	0.0001
36	1.715	0	2.3446e-06	4.4093e-06	2.1355e-05	4.3825e-05	0.0001
37	1.8662	0	2.6545e-07	5.0217e-06	2.7683e-05	3.388e-05	5.2151
38	2.0261	0	5.4286e-07	3.5584e-06	1.9841e-05	3.2561e-05	4.6305
39	2.1949	0	1.5332e-06	2.2585e-05	9.5902e-06	3.1168e-05	4.1488
40	2.3728	0	4.1159e-06	1.2545e-05	1.5104e-05	1.9972e-05	4.8064
41	2.56	0	4.9992e-06	9.9133e-06	2.4176e-05	2.6663e-05	3.4515
42	2.7568	0	7.7981e-06	1.537e-05	2.0404e-05	3.8764e-05	3.1904
43	2.9635	0	1.0694e-05	1.9867e-05	2.6641e-05	3.3667e-05	4.0931
44	3.1803	0	1.3309e-05	1.8778e-05	4.551e-05	3.4837e-05	4.3432
45	3.4074	0	1.3226e-05	2.3e-05	3.5495e-05	3.4352e-05	4.101
46	3.645	0	3.533e-06	2.5708e-05	3.2758e-05	4.0617e-05	3.8932
47	3.8934	0	1.8503e-05	2.4946e-05	2.607e-05	3.9121e-05	3.9913

```

mp_support_graph('cl_st_graph_title') = {'Pstationary(Asset,Age), asset=x, age=color'};
mp_support_graph('cl_st_ytitle') = {'Conditional Aggregate Mass'};
mp_support_graph('cl_st_xtitle') = {'A (savings)'};
mp_support_graph('st_rowvar_name') = 'age=';
mp_support_graph('it_legend_select') = 5;
mp_support_graph('st_rounding') = '6.0f';
mp_support_graph('bl_graph_logy') = true;
mp_support_graph('cl_colors') = 'copper';
ff_graph_grid((tb_prob_aage{1:end, 3:end}'), age_grid, agrid, mp_support_graph);% Consumption Choice

```



6.3.6 Probability Statistics A, C and V Conditional on Ages

Where are the mass at?

```
ap_ss = mp_dsvfi_results('ap_ss');
c_ss = mp_dsvfi_results('cons_ss');
v_ss = mp_dsvfi_results('v_ss');
n_ss = mp_dsvfi_results('n_ss');

y_head_inc = mp_dsvfi_results('y_head_inc_ss');
```

```

y_spouse_inc = mp_dsvfi_results('y_spouse_inc_ss');

yshr_wage = mp_dsvfi_results('yshr_wage_ss');
yshr_SS = mp_dsvfi_results('yshr_SS_ss');
yshr_nttxss = mp_dsvfi_results('yshr_nttxss_ss');

for it_ctr=1:size(ap_ss, 1)
    if (ismember(it_ctr, round(linspace(1, size(ap_ss, 1), 3))))
        display(['age =' num2str(age_grid(it_ctr))]);

    % construct input data
    Phi_true_age = Phi_true(it_ctr, :, :, :, :, :);
    ap_ss_age = ap_ss(it_ctr, :, :, :, :, :);
    c_ss_age = c_ss(it_ctr, :, :, :, :, :);
    v_ss_age = v_ss(it_ctr, :, :, :, :, :);
    n_ss_age = n_ss(it_ctr, :, :, :, :, :);

    y_head_inc_age = y_head_inc(it_ctr, :, :, :, :, :);
    y_spouse_inc_age = y_spouse_inc(it_ctr, :, :, :, :, :);
    yshr_wage_age = yshr_wage(it_ctr, :, :, :, :, :);
    yshr_SS_age = yshr_SS(it_ctr, :, :, :, :, :);
    yshr_nttxss_age = yshr_nttxss(it_ctr, :, :, :, :, :);

    mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
    mp_cl_ar_xyz_of_s('ap_ss') = {ap_ss_age(:), zeros(1)};
    mp_cl_ar_xyz_of_s('c_ss') = {c_ss_age(:), zeros(1)};
    mp_cl_ar_xyz_of_s('v_ss') = {v_ss_age(:), zeros(1)};
    mp_cl_ar_xyz_of_s('n_ss') = {n_ss_age(:), zeros(1)};
    mp_cl_ar_xyz_of_s('y_head_inc') = {y_head_inc_age(:), zeros(1)};
    mp_cl_ar_xyz_of_s('y_spouse') = {y_spouse_inc_age(:), zeros(1)};
    mp_cl_ar_xyz_of_s('yshr_wage') = {yshr_wage_age(:), zeros(1)};
    mp_cl_ar_xyz_of_s('yshr_SS') = {yshr_SS_age(:), zeros(1)};
    mp_cl_ar_xyz_of_s('yshr_nttxss') = {yshr_nttxss_age(:), zeros(1)};
    mp_cl_ar_xyz_of_s('ar_st_y_name') = ["ap_ss", "c_ss", "v_ss", "n_ss", ...
        "y_head_inc", "y_spouse", "yshr_wage", "yshr_SS", "yshr_nttxss"];

    % controls
    mp_support = containers.Map('KeyType','char', 'ValueType','any');
    mp_support('ar_fl_percentiles') = [0.01 10 25 50 75 90 99.99];
    mp_support('bl_display_final') = true;
    mp_support('bl_display_detail') = false;
    mp_support('bl_display_drvm2outcomes') = false;
    mp_support('bl_display_drvstats') = false;
    mp_support('bl_display_drvm2covcor') = false;

    % Call Function
    mp_cl_mt_xyz_of_s = ff_simu_stats(Phi_true_age(:)/sum(Phi_true_age,'all'), mp_cl_ar_xyz_of_s);
end
end

age =18
xxx tb_outcomes: all stats xxx
OriginalVariableNames      ap_ss          c_ss          v_ss          n_ss          y_head_inc
-----  -----  -----  -----  -----
{'mean'}                  }      10.116      0.75737     -37.312      1.9854      0.8434
{'unweighted_sum'}         }      11476      2.4434e+05   -7.8101e+05     21          4422.
{'sd'}                     }      6.9537      0.67774      55.469      1.0848      0.9050

```

{'coefofvar'}	}	0.68742	0.89486	-1.4866	0.54639	1.073
{'gini'}	}	0.32034	0.41117	-0.64451	0.268	0.4135
{'min'}	}	1	0.035637	-867.32	1	0.03810
{'max'}	}	151	18.059	25.519	6	13.78
{'pYiso'}	}	0	0	0	0	0
{'pYls0'}	}	0	0	0.8166	0	
{'pYgr0'}	}	1	1	0.1834	1	
{'pYisMINY'}	}	0.11052	0.0014188	7.8342e-06	0.41786	0.003370
{'pYisMAXY'}	}	0	0	0	0.0060544	
{'p0_01'}	}	1	0.035637	-745.16	1	0.03810
{'p10'}	}	1	0.24578	-86.259	1	0.1467
{'p25'}	}	7	0.3161	-50.56	1	0.2880
{'p50'}	}	9	0.51551	-25.263	2	0.5652
{'p75'}	}	11	0.88958	-5.3994	3	1.109
{'p90'}	}	23	1.5797	6.1229	4	2.176
{'p99_99'}	}	52	6.8857	23.695	6	8.383
{'fl_cov_ap_ss'}	}	48.354	1.9167	115.84	0.29345	1.774
{'fl_cor_ap_ss'}	}	1	0.4067	0.30034	0.038901	0.2819
{'fl_cov_c_ss'}	}	1.9167	0.45934	20.257	0.067217	0.5982
{'fl_cor_c_ss'}	}	0.4067	1	0.53884	0.091423	0.975
{'fl_cov_v_ss'}	}	115.84	20.257	3076.8	2.8057	24.48
{'fl_cor_v_ss'}	}	0.30034	0.53884	1	0.046626	0.4877
{'fl_cov_n_ss'}	}	0.29345	0.067217	2.8057	1.1768	-4.9873e-1
{'fl_cor_n_ss'}	}	0.038901	0.091423	0.046626	1	-5.0797e-1
{'fl_cov_y_head_inc'}	}	1.7747	0.59824	24.488	-4.9873e-18	0.8191
{'fl_cor_y_head_inc'}	}	0.28199	0.9753	0.48778	-5.0797e-18	
{'fl_cov_y_spouse'}	}	3.1074	0.081697	4.9077	0.13364	0.02175
{'fl_cor_y_spouse'}	}	0.77947	0.21026	0.15433	0.21488	0.0419
{'fl_cov_yshr_wage'}	}	-9.6296e-31	-3.1682e-32	5.4234e-31	3.667e-31	3.5697e-3
{'fl_cor_yshr_wage'}	}	-1.5592e-16	-5.2631e-17	1.1008e-17	3.8058e-16	4.4408e-1
{'fl_cov_yshr_SS'}	}	0	0	0	0	
{'fl_cor_yshr_SS'}	}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}	}	0.16611	0.021334	1.8502	0.0077776	0.02521
{'fl_cor_yshr_nttxss'}	}	0.58487	0.77071	0.81669	0.17554	0.6822
{'fracByP0_01'}	}	0.010925	6.6761e-05	0.0030622	0.21046	0.00015222
{'fracByP10'}	}	0.010925	0.050401	0.44077	0.21046	0.01922
{'fracByP25'}	}	0.148	0.072459	0.71224	0.21046	0.09634
{'fracByP50'}	}	0.28531	0.21889	0.94749	0.53024	0.2966
{'fracByP75'}	}	0.60536	0.47077	1.0368	0.77109	0.5936
{'fracByP90'}	}	0.758	0.70215	1.0326	0.92834	0.8450
{'fracByP99_99'}	}	0.99975	0.99993	1	1	

age =59

xxx tb_outcomes: all stats xxx

OriginalVariableNames		ap_ss	c_ss	v_ss	n_ss	y_head_inc
{'mean'}	}	54.878	1.2923	-12.279	1.7239	1.8459
{'unweighted_sum'}	}	11476	2.7092e+05	-80406	21	13268
{'sd'}	}	23.415	1.0959	19.332	0.90777	2.0412
{'coefofvar'}	}	0.42667	0.84801	-1.5745	0.52659	1.1058
{'gini'}	}	0.23612	0.3991	-0.81005	0.23461	0.48077
{'min'}	}	1	0.055882	-229.42	1	0.059541
{'max'}	}	151	32.48	14.764	6	23.47
{'pYiso'}	}	0	0	0	0	0
{'pYls0'}	}	0	0	0.73941	0	0
{'pYgr0'}	}	1	1	0.26059	1	1
{'pYisMINY'}	}	0.0042169	2.9508e-05	3.9539e-07	0.48835	9.9253e-05

{'pYisMAXY'}	}	4.8703e-06	2.3072e-08	0	0.0036816	1.9995e-06
{'p0_01'}	}	1	0.05663	-132.27	1	0.059554
{'p10'}	}	26	0.31762	-39.004	1	0.38493
{'p25'}	}	40	0.59646	-18.282	1	0.63825
{'p50'}	}	54	1.0652	-7.1081	2	1.1351
{'p75'}	}	70	1.6718	0.46981	2	2.1332
{'p90'}	}	85	2.4861	6.4893	3	4.1604
{'p99_99'}	}	146	15.179	14.695	6	22.847
{'fl_cov_ap_ss'}	}	548.26	22.158	403.41	3.0428	38.333
{'fl_cor_ap_ss'}	}	1	0.86352	0.8912	0.14315	0.80205
{'fl_cov_c_ss'}	}	22.158	1.201	13.858	0.23973	2.0792
{'fl_cor_c_ss'}	}	0.86352	1	0.6541	0.24098	0.92951
{'fl_cov_v_ss'}	}	403.41	13.858	373.74	3.5819	22.934
{'fl_cor_v_ss'}	}	0.8912	0.6541	1	0.20411	0.58118
{'fl_cov_n_ss'}	}	3.0428	0.23973	3.5819	0.82404	0.062213
{'fl_cor_n_ss'}	}	0.14315	0.24098	0.20411	1	0.033576
{'fl_cov_y_head_inc'}	}	38.333	2.0792	22.934	0.062213	4.1664
{'fl_cor_y_head_inc'}	}	0.80205	0.92951	0.58118	0.033576	1
{'fl_cov_y_spouse'}	}	6.1095	0.27813	4.5119	0.2771	0.17233
{'fl_cor_y_spouse'}	}	0.23287	0.22651	0.2083	0.27244	0.07535
{'fl_cov_yshr_wage'}	}	-1.3956	-0.043321	-1.0776	-0.0071751	-0.056896
{'fl_cor_yshr_wage'}	}	-0.66407	-0.44044	-0.62107	-0.088065	-0.31056
{'fl_cov_yshr_SS'}	}	0	0	0	0	0
{'fl_cor_yshr_SS'}	}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}	}	0.77952	0.028412	0.68735	0.0085362	0.047811
{'fl_cor_yshr_nttxss'}	}	0.88801	0.69155	0.94837	0.25083	0.62479
{'fracByP0_01'}	}	7.6842e-05	5.431e-06	0.001404	0.28329	4.1671e-06
{'fracByP10'}	}	0.027337	0.019346	0.47531	0.28329	0.013211
{'fracByP25'}	}	0.11727	0.077024	0.79795	0.28329	0.054199
{'fracByP50'}	}	0.33388	0.22863	1.0581	0.72028	0.18178
{'fracByP75'}	}	0.62869	0.48302	1.117	0.72028	0.41537
{'fracByP90'}	}	0.83409	0.72082	1.0748	0.85389	0.64728
{'fracByP99_99'}	}	0.9998	0.99882	1	1	0.99936
<hr/>						
age =100						
xxx tb_outcomes: all stats xxx						
OriginalVariableNames		ap_ss	c_ss	v_ss	n_ss	y_head_inc
-----	-----	-----	-----	-----	-----	-----
{'mean'}	}	1	0.35551	-2.9555	1.4797	0.26067
{'unweighted_sum'}	}	1	2.807e+05	1215	21	491.5
{'sd'}	}	1.7875e-14	0.23928	1.0697	0.50567	0.023035
{'coefofvar'}	}	1.7875e-14	0.67307	-0.36194	0.34173	0.088367
{'gini'}	}	0	0.28119	-0.18783	0.12034	0.041657
{'min'}	}	1	0.2179	-10.065	1	0.24433
{'max'}	}	1	141.66	0.99282	6	5.6926
{'pYis0'}	}	0	0	0	0	0
{'pYls0'}	}	0	0	0.99182	0	0
{'pYgr0'}	}	1	1	0.0081757	1	1
{'pYisMINY'}	}	1	0.35002	1.5074e-10	0.5232	0.50379
{'pYisMAXY'}	}	1	0	0	4.2206e-08	0
{'p0_01'}	}	1	0.2179	-6.3349	1	0.24433
{'p10'}	}	1	0.2179	-3.6603	1	0.24433
{'p25'}	}	1	0.2179	-3.5892	1	0.24433
{'p50'}	}	1	0.25824	-3.5892	1	0.24433
{'p75'}	}	1	0.37165	-2.5873	2	0.29263
{'p90'}	}	1	0.6134	-1.2288	2	0.29283
{'p99_99'}	}	1	2.9509	0.52075	4	0.3403

{'fl_cov_ap_ss'}	}	3.195e-28	6.5284e-30	-2.443e-29	3.5367e-29	2.9775e-31
{'fl_cor_ap_ss'}	}	1	1.5264e-15	-1.2777e-15	3.9129e-15	7.2317e-16
{'fl_cov_c_ss'}	}	6.5284e-30	0.057256	0.20779	0.059046	0.0016896
{'fl_cor_c_ss'}	}	1.5264e-15	1	0.81181	0.488	0.30655
{'fl_cov_v_ss'}	}	-2.443e-29	0.20779	1.1443	0.15982	0.010842
{'fl_cor_v_ss'}	}	-1.2777e-15	0.81181	1	0.29547	0.44002
{'fl_cov_n_ss'}	}	3.5367e-29	0.059046	0.15982	0.2557	0.0018939
{'fl_cor_n_ss'}	}	3.9129e-15	0.488	0.29547	1	0.1626
{'fl_cov_y_head_inc'}	}	2.9775e-31	0.0016896	0.010842	0.0018939	0.00053059
{'fl_cor_y_head_inc'}	}	7.2317e-16	0.30655	0.44002	0.1626	1
{'fl_cov_y_spouse'}	}	6.9736e-31	0.051708	0.16183	0.0533	0.00067244
{'fl_cor_y_spouse'}	}	1.5749e-16	0.87235	0.61072	0.4255	0.11785
{'fl_cov_yshr_wage'}	}	1.4253e-30	0.039337	0.15536	0.083876	0.00066872
{'fl_cor_yshr_wage'}	}	3.6093e-16	0.74409	0.65738	0.75078	0.1314
{'fl_cov_yshr_SS'}	}	1.2113e-29	-0.040637	-0.16221	-0.085115	-0.00073196
{'fl_cor_yshr_SS'}	}	3.0517e-15	-0.76482	-0.68289	-0.75803	-0.1431
{'fl_cov_yshr_nttxss'}	}	-1.3389e-29	0.044612	0.17828	0.091702	0.00088432
{'fl_cor_yshr_nttxss'}	}	-3.1042e-15	0.77263	0.69067	0.75153	0.1591
{'fracByP0_01'}	}	1	0.21454	0.00051608	0.35357	0.47222
{'fracByP10'}	}	1	0.21454	0.21323	0.35357	0.47222
{'fracByP25'}	}	1	0.21454	0.64329	0.35357	0.47222
{'fracByP50'}	}	1	0.32886	0.64329	0.35357	0.47222
{'fracByP75'}	}	1	0.54497	0.88331	0.99419	0.87831
{'fracByP90'}	}	1	0.75075	0.97695	0.99419	0.88528
{'fracByP99_99'}	}	1	0.99925	1	0.99999	0.99987

Chapter 7

Value of Each Check

7.1 2020 V and C without Unemployment

This is the example vignette for function: `snw_a4chk_wrk_bisec_vec` from the [PrjOptiSNW Package](#). This function solves for the V(states, check) for individuals working. Dense solution. Bisection, most time for the test here taken to generate the income matrixes. But these can be generated out of the check loops.

7.1.1 Test SNW_A4CHK_WRK_BISEC_VEC Defaults Dense

Call the function with default parameters. Solve first for non-covid value and policy. Then depending on 2020 taxes, solve for 2020 policy and value.

```
mp_params = snw_mp_param('default_docdense');
mp_controls = snw_mp_control('default_test');
mp_controls('bl_print_vfi') = false;
mp_controls('bl_timer') = true;
[V_ss,~,cons_ss,~] = snw_vfi_main_bisec_vec(mp_params, mp_controls);

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=490.
-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      i     idx    ndim   numel      rowN      colN        sum      mean      std
      -     ---    ----  -----  -----  -----  -----  -----
V_VFI    1      1      6  4.37e+07     83  5.265e+05 -1.5339e+08 -3.5101  26.11
ap_VFI   2      2      6  4.37e+07     83  5.265e+05  1.4159e+09 32.402   36.79
cons_VFI 3      3      6  4.37e+07     83  5.265e+05  2.1402e+08 4.8975  8.329

xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxxx
      c1      c2      c3      c4      c5    c526496    c526497    c526498    c
      ----  -----  -----  -----  -----  -----  -----  -----  -----
r1  -346.51  -346.12  -343.63  -337.86  -328.51  21.702   21.852   22.003
r2  -334.38  -333.99  -331.51  -325.83  -316.83  21.724   21.869   22.015
r3  -322.45  -322.06  -319.6   -314.14  -305.6   21.745   21.885   22.027
r4  -310.63  -310.27  -307.99  -302.88  -294.87  21.767   21.903   22.041
r5  -299.94  -299.6   -297.46  -292.67  -285.12  21.775   21.907   22.042
r79 -9.9437 -9.9325 -9.8557 -9.6597 -9.3232  2.5394  2.5501  2.5602
r80 -8.9023 -8.8911 -8.8143 -8.6183 -8.2818  2.3039  2.3121  2.3198
r81 -7.6363 -7.6251 -7.5484 -7.3524 -7.0159  2.0068  2.0124  2.0176
```

r82	-5.9673	-5.9561	-5.8793	-5.6833	-5.3468	1.5958	1.5989	1.6018
r83	-3.5892	-3.578	-3.5012	-3.3052	-2.9687	0.97904	0.98004	0.98097

xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499
	--	--	-----	-----	-----	-----	-----	-----	-----
r1	0	0	0.0005656	0.0075134	0.022901	114.75	120.41	126.27	132.3
r2	0	0	0.00051498	0.0065334	0.021549	114.86	120.53	126.41	132.5
r3	0	0	0.00051498	0.0049294	0.019875	114.97	120.65	126.56	132.
r4	0	0	0.00051498	0.0047937	0.019672	115.73	121.42	127.34	133.5
r5	0	0	0.00048517	0.0046683	0.019484	116.5	122.21	128.15	134.3
r79	0	0	0	0	0	81.091	85.68	90.335	94.37
r80	0	0	0	0	0	76.669	80.563	84.304	88.0
r81	0	0	0	0	0	68.313	71.534	74.475	77.83
r82	0	0	0	0	0	50.126	53.467	56.953	58.74
r83	0	0	0	0	0	0	0	0	0

xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498
	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.036717	0.037251	0.040426	0.04363	0.048012	9.6491	9.817	9.9649
r2	0.036717	0.037251	0.040477	0.04461	0.049364	9.8118	9.9685	10.101
r3	0.036717	0.037251	0.040477	0.046214	0.051039	9.9779	10.12	10.234
r4	0.038144	0.038678	0.041903	0.047776	0.052666	10.131	10.258	10.354
r5	0.039534	0.040068	0.043323	0.04929	0.054241	10.272	10.384	10.463
r79	0.2179	0.21844	0.22216	0.23228	0.25197	35.858	37.092	38.455
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.253	42.183	44.459
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.587	51.19	54.266
r82	0.2179	0.21844	0.22216	0.23228	0.25197	66.755	69.238	71.77
r83	0.2179	0.21844	0.22216	0.23228	0.25197	116.87	122.69	128.71

```
welf_checks = 2; % 2 checks is $200 dollar of welfare checks
xi=1; % xi=0 full income loss from covid shock, xi=1, no covid income losses
b=1; % when xi=1, b does not matter, no income losses
TR = 100/58056;
mp_params('TR') = TR;
mp_params('xi') = xi;
mp_params('b') = b;
% if = mp_params('a2_covidyr_manna_heaven'), V_emp_2020 same as V_ss if b=1
% or xi=1.
% if = mp_params('a2_covidyr_tax_fully_pay'), V_emp_2020 differ due to 2020
% tax differences
mp_params('a2_covidyr') = mp_params('a2_covidyr_manna_heaven');
% mp_params('a2_covidyr') = mp_params('a2_covidyr_tax_fully_pay');
[V_emp_2020,~,cons_emp_2020,~] = snw_vfi_main_bisec_vec(mp_params, mp_controls, V_ss);
```

Completed SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=d

xxx

CONTAINER NAME: mp_outcomes ND Array (Matrix etc)

xxx

i	idx	ndim	numel	rowN	colN	sum	mean	std
-	---	---	-----	---	-----	-----	-----	-----
V_VFI	1	1	6	4.37e+07	83	5.265e+05	-1.5339e+08	-3.5101
ap_VFI	2	2	6	4.37e+07	83	5.265e+05	1.4159e+09	32.402

cons_VFI	3	3	6	4.37e+07	83	5.265e+05	2.1402e+08	4.8975	8.329
xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxxxxx									
	c1	c2	c3	c4	c5	c526496	c526497	c526498	c
r1	-346.51	-346.12	-343.63	-337.86	-328.51	21.702	21.852	22.003	
r2	-334.38	-333.99	-331.51	-325.83	-316.83	21.724	21.869	22.015	
r3	-322.45	-322.06	-319.6	-314.14	-305.6	21.745	21.885	22.027	
r4	-310.63	-310.27	-307.99	-302.88	-294.87	21.767	21.903	22.041	
r5	-299.94	-299.6	-297.46	-292.67	-285.12	21.775	21.907	22.042	
r79	-9.9437	-9.9325	-9.8557	-9.6597	-9.3232	2.5394	2.5501	2.5602	
r80	-8.9023	-8.8911	-8.8143	-8.6183	-8.2818	2.3039	2.3121	2.3198	
r81	-7.6363	-7.6251	-7.5484	-7.3524	-7.0159	2.0068	2.0124	2.0176	
r82	-5.9673	-5.9561	-5.8793	-5.6833	-5.3468	1.5958	1.5989	1.6018	
r83	-3.5892	-3.578	-3.5012	-3.3052	-2.9687	0.97904	0.98004	0.98097	0
xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxxxxx									
	c1	c2	c3	c4	c5	c526496	c526497	c526498	c5264
r1	0	0	0.0005656	0.0075134	0.022901	114.75	120.41	126.27	132.3
r2	0	0	0.00051498	0.0065334	0.021549	114.86	120.53	126.41	132.5
r3	0	0	0.00051498	0.0049294	0.019875	114.97	120.65	126.56	132.
r4	0	0	0.00051498	0.0047937	0.019672	115.73	121.42	127.34	133.5
r5	0	0	0.00048517	0.0046683	0.019484	116.5	122.21	128.15	134.3
r79	0	0	0	0	0	81.091	85.68	90.335	94.37
r80	0	0	0	0	0	76.669	80.563	84.304	88.0
r81	0	0	0	0	0	68.313	71.534	74.475	77.83
r82	0	0	0	0	0	50.126	53.467	56.953	58.74
r83	0	0	0	0	0	0	0	0	0
xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxx									
	c1	c2	c3	c4	c5	c526496	c526497	c526498	c52649
r1	0.036717	0.037251	0.040426	0.04363	0.048012	9.6491	9.817	9.9649	
r2	0.036717	0.037251	0.040477	0.04461	0.049364	9.8118	9.9685	10.101	
r3	0.036717	0.037251	0.040477	0.046214	0.051039	9.9779	10.12	10.234	
r4	0.038144	0.038678	0.041903	0.047776	0.052666	10.131	10.258	10.354	
r5	0.039534	0.040068	0.043323	0.04929	0.054241	10.272	10.384	10.463	
r79	0.2179	0.21844	0.22216	0.23228	0.25197	35.858	37.092	38.455	
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.253	42.183	44.459	
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.587	51.19	54.266	
r82	0.2179	0.21844	0.22216	0.23228	0.25197	66.755	69.238	71.77	
r83	0.2179	0.21844	0.22216	0.23228	0.25197	116.87	122.69	128.71	
[V_W_2020, C_W_2020] = snw_a4chk_wrk_bisec_vec(welf_checks, V_emp_2020, cons_emp_2020, mp_params, mp									
Completed SNW_A4CHK_WRK_BISEC_VEC; welf_checks=2; TR=0.0017225; SNW_MP_PARAM=default_docdense; SNW_MP_CO									

xx									
CONTAINER NAME: mp_container_map ND Array (Matrix etc)									
xx									
	i	idx	ndim	numel	rowN	colN	sum	mean	
C_W	1	1	6	4.37e+07	83	5.265e+05	2.1404e+08	4.8981	

C_W_minus_C_ss	2	2	6	4.37e+07	83	5.265e+05	23044	0.00052732
V_W	3	3	6	4.37e+07	83	5.265e+05	-1.5281e+08	-3.4969
V_W_minus_V_ss	4	4	6	4.37e+07	83	5.265e+05	5.7996e+05	0.013271
mn_MPC	5	5	6	4.37e+07	83	5.265e+05	6.6891e+06	0.15307

```
mn_V_W_gain_check = V_W_2020 - V_emp_2020;
mn_MPC_W_gain_share_check = (C_W_2020 - cons_emp_2020)./(welf_checks*mp_params('TR'));
```

7.1.2 Dense 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 = 18:100;
agrid = mp_params('agrid');
eta_H_grid = mp_params('eta_H_grid');
eta_S_grid = mp_params('eta_S_grid');
ar_st_eta_HS_grid = string(cellstr([num2str(eta_H_grid), 'hz=%3.2f;'], num2str(eta_S_grid), 'wz=%3.2f'));
edu_grid = [0,1];
marry_grid = [0,1];
kids_grid = (1:1:mp_params('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});
```

7.1.3 Analyze Difference in V and C with Check

The difference between V and V with Check, marginal utility gain given the check.

```
% 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') = 'eastoutside';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('it_legend_select') = 21; % how many shock legends to show
mp_support_graph('cl_colors') = 'jet';

MEAN(MN_V_GAIN_CHECK(A,Z))
```

Tabulate value and policies along savings and shocks:

```
% Set
ar_permute = [1,4,5,6,3,2];
% Value Function
st_title = ['MEAN(MN_V_W_GAIN_CHECK(A,Z)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_params('TR')) ];
tb_az_v = ff_summ_nd_array(st_title, mn_V_W_gain_check, true, ["mean"], 4, 1, cl_mp_datasetdesc, ar_permute);

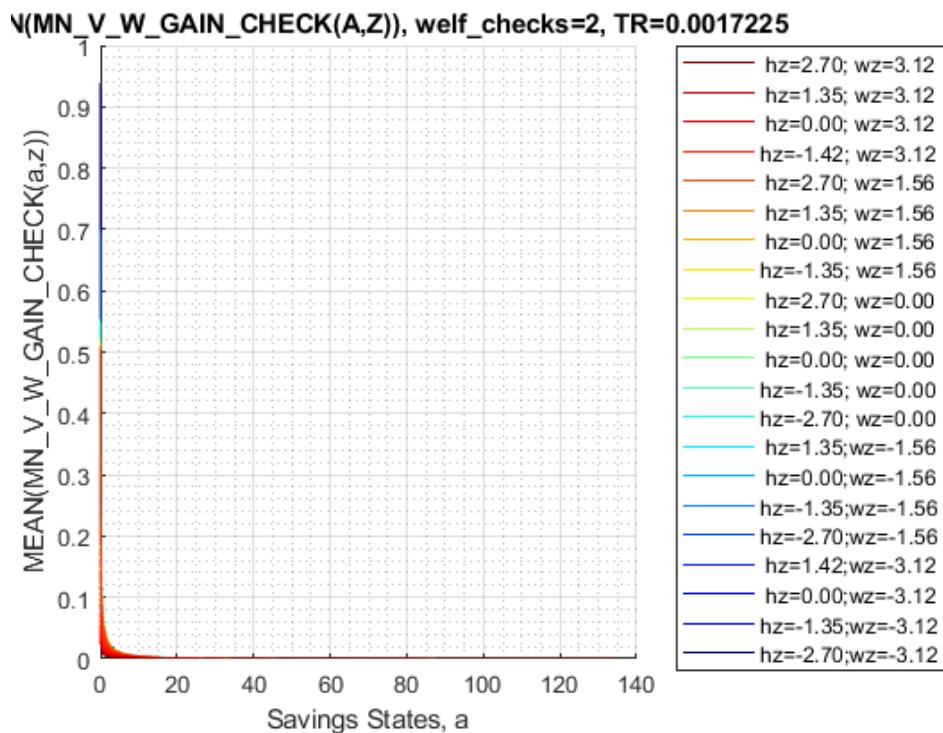
xxx MEAN(MN_V_W_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxx
group      savings      mean_eta_1      mean_eta_2      mean_eta_3      mean_eta_4      mean_eta_5      mean_eta_6
-----  -----  -----  -----  -----  -----  -----  -----
1           0       0.93933     0.8402      0.75179      0.67285      0.60245
```

5	0.032959	0.4333	0.40626	0.37812	0.35006	0.32301
---	----------	--------	---------	---------	---------	---------

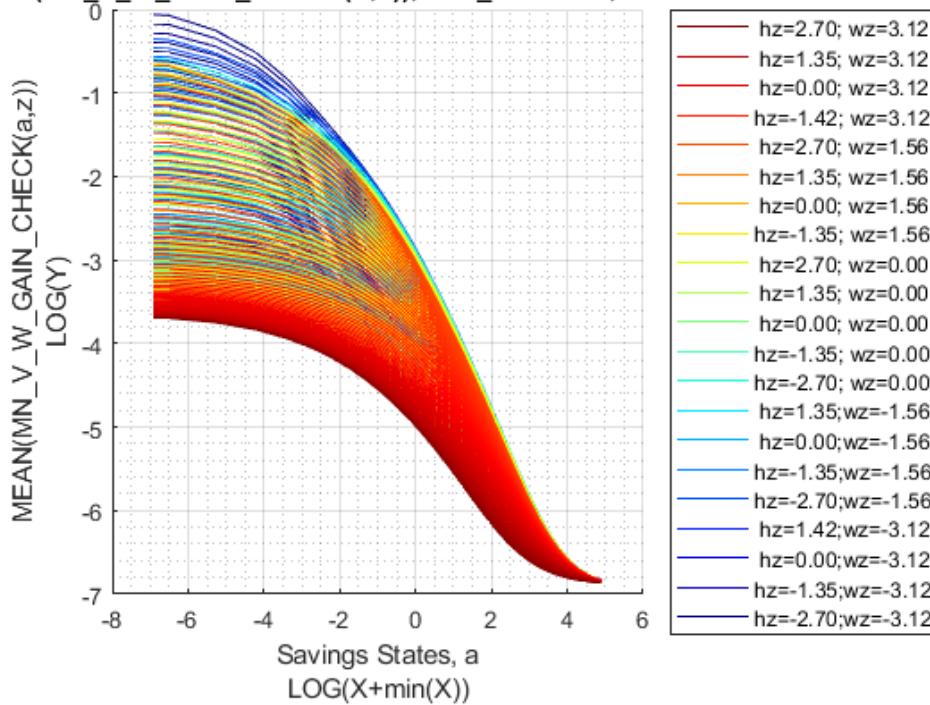
```

st_title = ['MEAN(MN\_V\_W\_GAIN\_CHECK(A,Z)), welf_checks=' num2str(welf_checks) ', TR=' num2str(m
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_V\_W\_GAIN\_CHECK(a,z))'};
ff_graph_grid((tb_az_v{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);

```



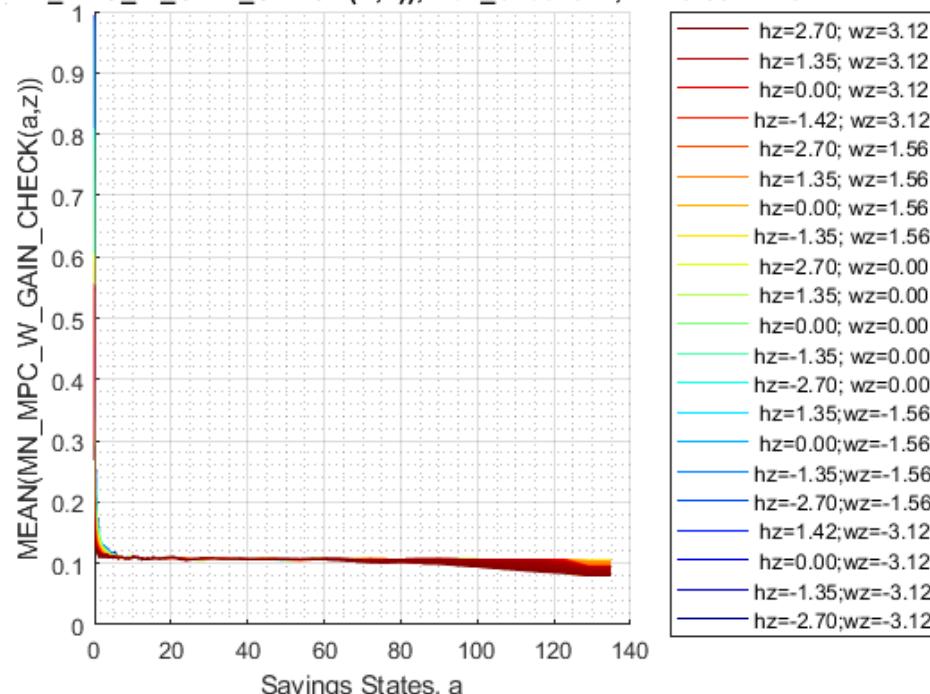
\N(MN_V_W_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225

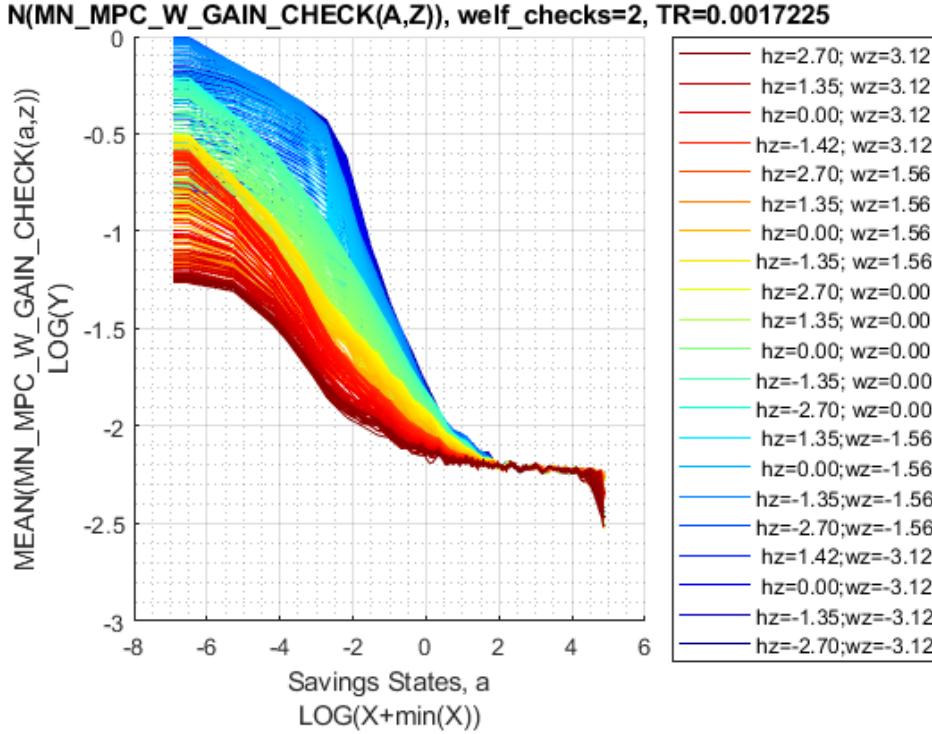


Graph Mean Consumption (*MPC: Share of Check Consumed*):

```
st_title = ['MEAN(MN\_MPC\_W\_GAIN\_CHECK(A,Z)), welf\_checks=' num2str(welf_checks) ', TR=' num2str
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_W\_GAIN\_CHECK(a,z))'};
ff_graph_grid((tb_az_c{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```

MN_MPC_W_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225





7.1.4 Analyze Kids and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = [...
    "k0M0", "K1M0", "K2M0", "K3M0", "K4M0", ...
    "k0M1", "K1M1", "K2M1", "K3M1", "K4M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {...
    'o', 'd', 's', 'x', '*', ...
    'o', 'd', 's', 'x', '*'};
mp_support_graph('cl_colors') = {...
    'red', 'red', 'red', 'red', 'red',...
    'blue', 'blue', 'blue', 'blue', 'blue'};
MEAN(VAL(KM,J)), MEAN(AP(KM,J)), MEAN(C(KM,J))

Tabulate value and policies:
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,4,1,6,5];
% Value Function
st_title = ['MEAN(MN_V_W_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_pa...
tb_az_v = ff_summ_nd_array(st_title, mn_V_W_gain_check, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar...

xxx MEAN(MN_V_W_GAIN_CHECK(KM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry mean_age_18 mean_age_19 mean_age_20 mean_age_21 mean_age_22
----- ----- ----- ----- ----- ----- ----- -----
1 1 0 0.031034 0.029745 0.028239 0.025902 0.02395
```

2	2	0	0.042182	0.040477	0.0384	0.035121	0.032371
3	3	0	0.048861	0.047196	0.045053	0.041229	0.038024
4	4	0	0.055272	0.053518	0.051182	0.046853	0.043227
5	5	0	0.060434	0.0587	0.056288	0.051575	0.047632
6	1	1	0.0088974	0.0084565	0.0080356	0.007287	0.0066592
7	2	1	0.011887	0.011299	0.010733	0.0097237	0.0088734
8	3	1	0.014254	0.013578	0.01292	0.011706	0.010686
9	4	1	0.017048	0.016271	0.015496	0.014054	0.012839
10	5	1	0.020638	0.019777	0.018893	0.017162	0.015705

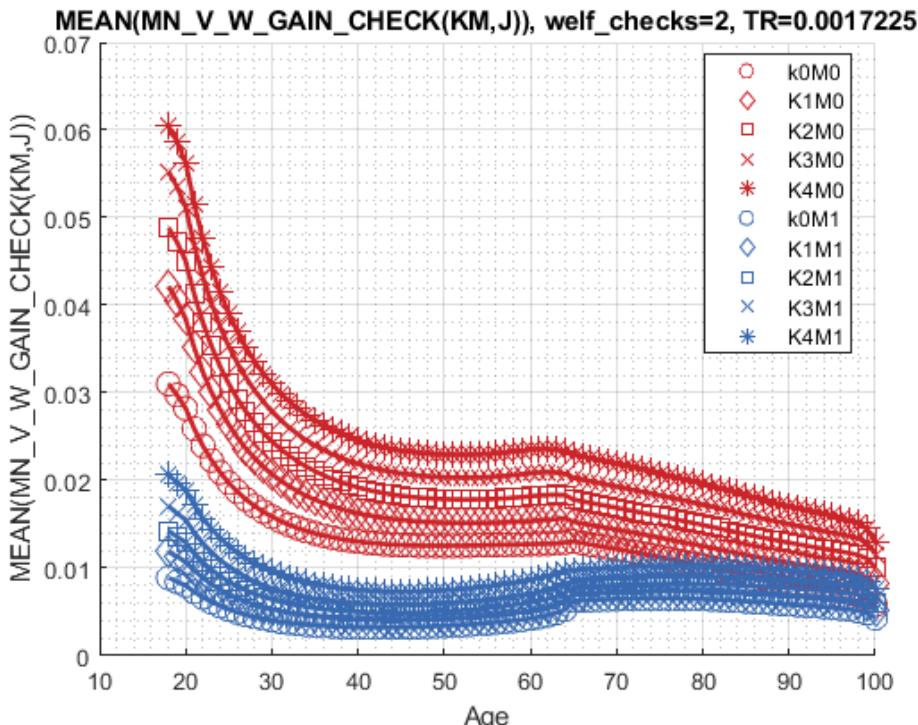
% Consumption Function

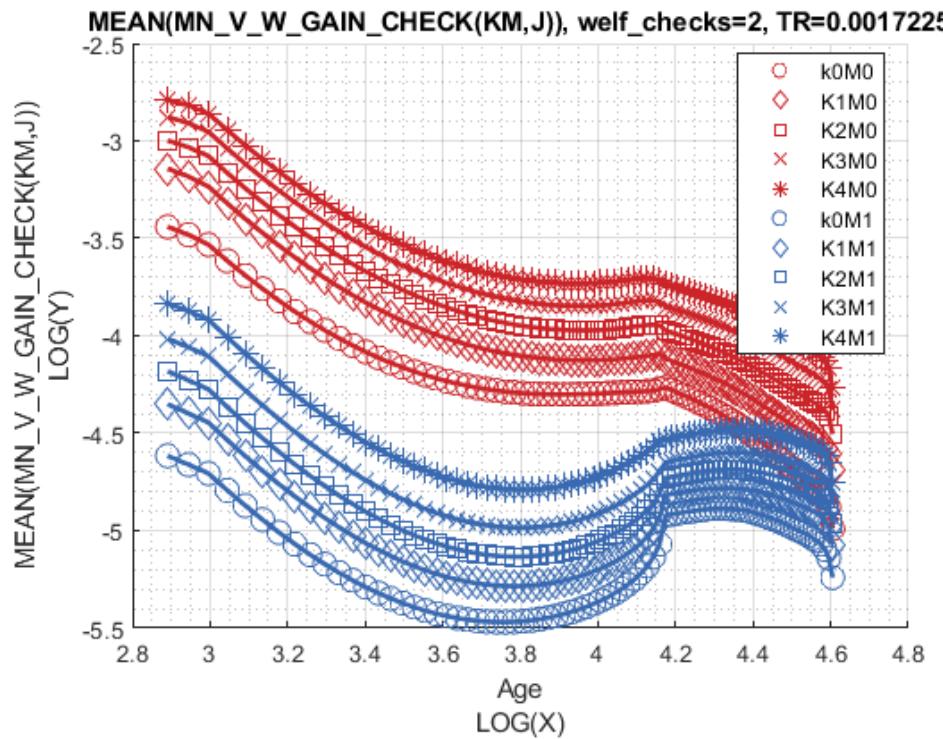
```
st_title = ['MEAN(MN_MPC_W_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
tb_az_c = ff_summ_nd_array(st_title, mn_MPC_W_gain_share_check, true, ["mean"], 3, 1, cl_mp_datasetd
```

group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	1	0	0.050486	0.055015	0.066697	0.065256	0.063327
2	2	0	0.057524	0.062751	0.075955	0.074114	0.073156
3	3	0	0.065349	0.07164	0.08726	0.084785	0.083817
4	4	0	0.069224	0.077159	0.092501	0.089833	0.087737
5	5	0	0.07472	0.082465	0.097494	0.094842	0.091941
6	1	1	0.08249	0.085999	0.088435	0.087541	0.086694
7	2	1	0.084681	0.087712	0.09101	0.089941	0.088666
8	3	1	0.08783	0.092431	0.09692	0.095481	0.094512
9	4	1	0.090426	0.093954	0.098752	0.097968	0.096993
10	5	1	0.095109	0.099933	0.1075	0.10332	0.10194

Graph Mean Values:

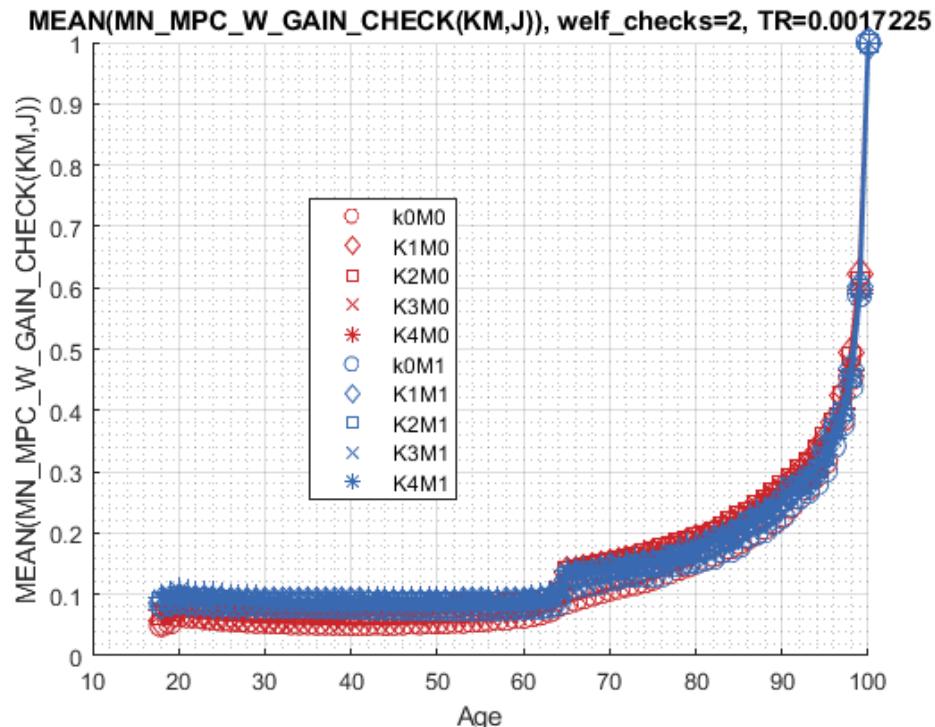
```
st_title = ['MEAN(MN_V_W_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN_V_W_GAIN_CHECK(KM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```

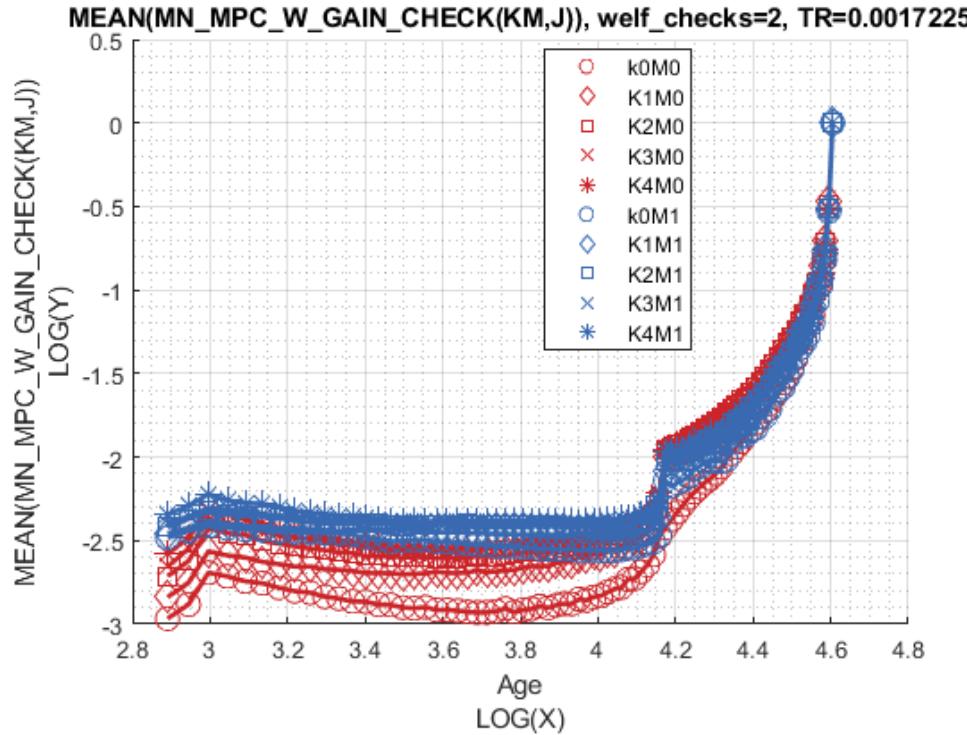




Graph Mean Consumption (*MPC: Share of Check Consumed*):

```
st_title = ['MEAN(MN\_MPC\_W\_GAIN\_CHECK(KM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(TR)];
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_W\_GAIN\_CHECK(KM,J))'};
ff_graph_grid((tb_az_c{1:end}, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```





7.1.5 Analyze Education and Marriage

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["E0M0", "E1M0", "E0M1", "E1M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {'*', 'p', '*', 'p' };
mp_support_graph('cl_colors') = {'red', 'red', 'blue', 'blue'};

MEAN(VAL(EM,J)), MEAN(AP(EM,J)), MEAN(C(EM,J))

Tabulate value and policies:
```

```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,6,1,4,5];
% Value Function
st_title = ['MEAN(MN_V_W_GAIN_CHECK(EM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_pa
tb_az_v = ff_summ_nd_array(st_title, mn_V_W_gain_check, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_
```

group	edu	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	0	0	0.049345	0.048033	0.0464	0.04381	0.041506
2	1	0	0.045768	0.043821	0.041264	0.036462	0.032576
3	0	1	0.015526	0.014854	0.014197	0.013177	0.01228
4	1	1	0.013563	0.012899	0.012234	0.010796	0.0096246

```
% Consumption
st_title = ['MEAN(MN_MPC_W_GAIN_CHECK(EM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
```

```

tb_az_c = ff_summ_nd_array(st_title, mn_MPC_W_gain_share_check, true, ["mean"], 3, 1, cl_mp_datasetd

xxx MEAN(MN_MPC_W_GAIN_CHECK(EM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx
group   edu    marry   mean_age_18   mean_age_19   mean_age_20   mean_age_21   mean_age_22
-----  ---  -----  -----  -----  -----  -----  -----
1       0      0      0.056783  0.060466  0.069707  0.069219  0.069323
2       1      0      0.070138  0.079146  0.098256  0.094312  0.090668
3       0      1      0.08145   0.084246  0.086686  0.086881  0.086662
4       1      1      0.094765  0.099766  0.10636   0.10282   0.10086

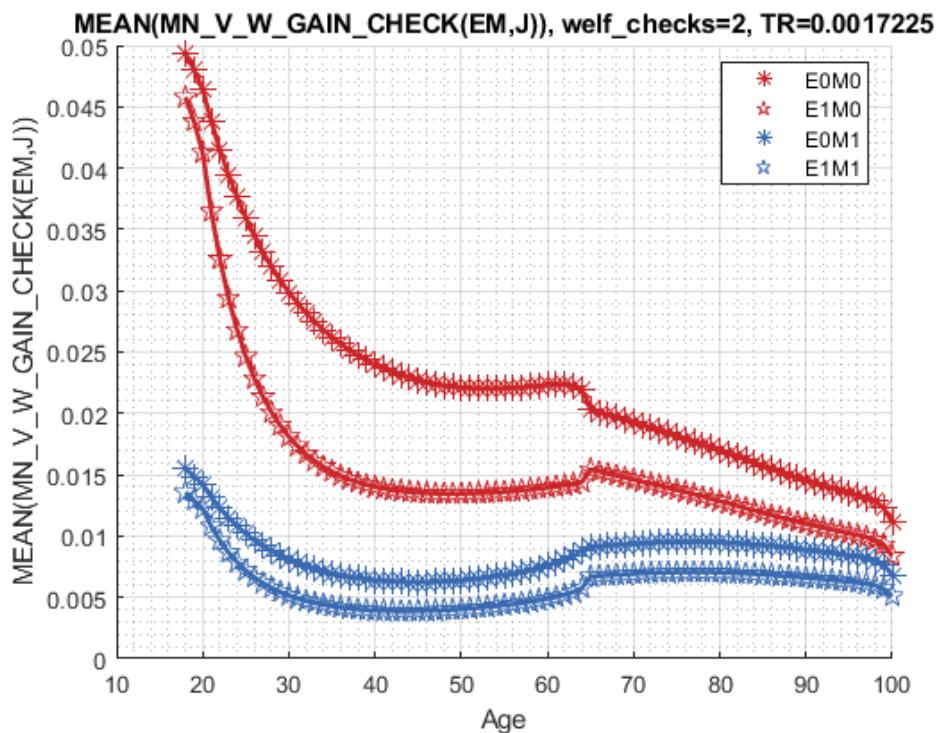
```

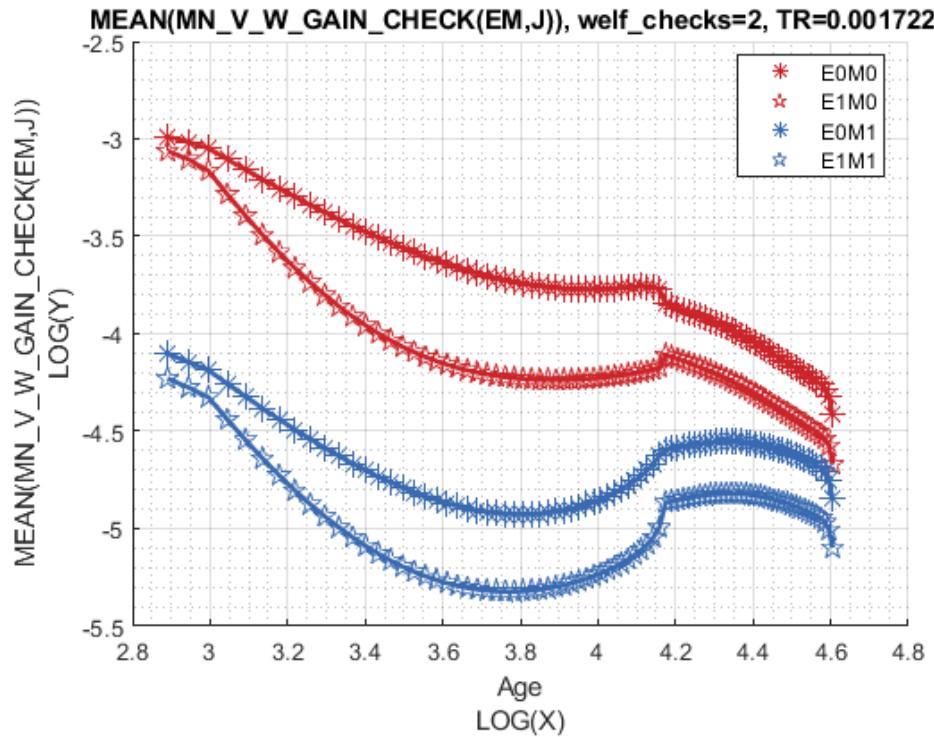
Graph Mean Values:

```

st_title = ['MEAN(MN\_V\_W\_GAIN\_CHECK(EM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_V\_W\_GAIN\_CHECK(EM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);

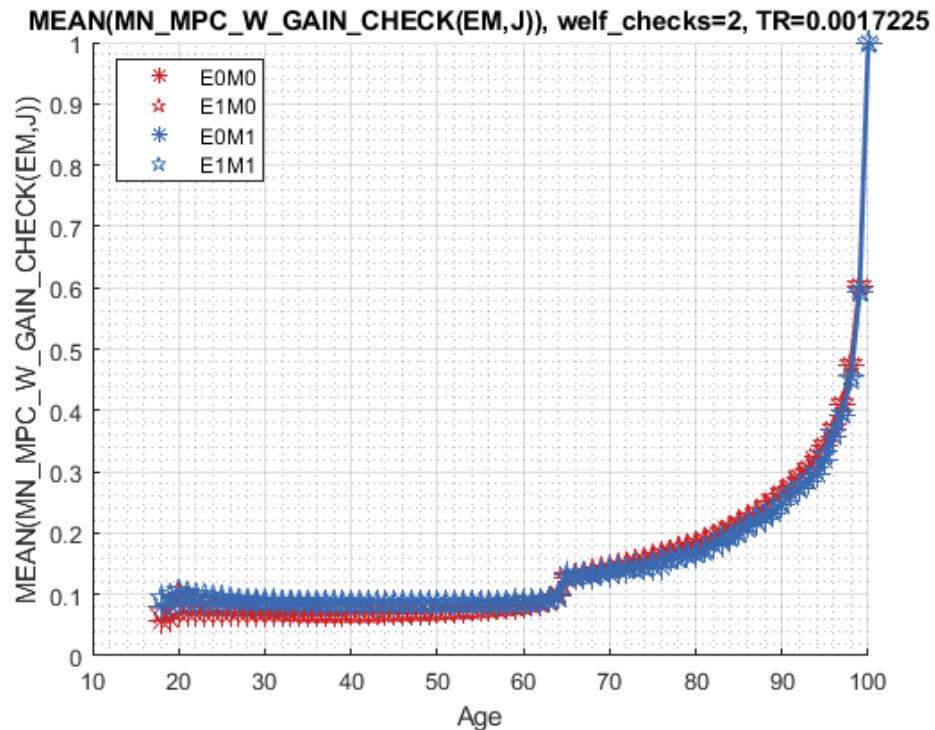
```

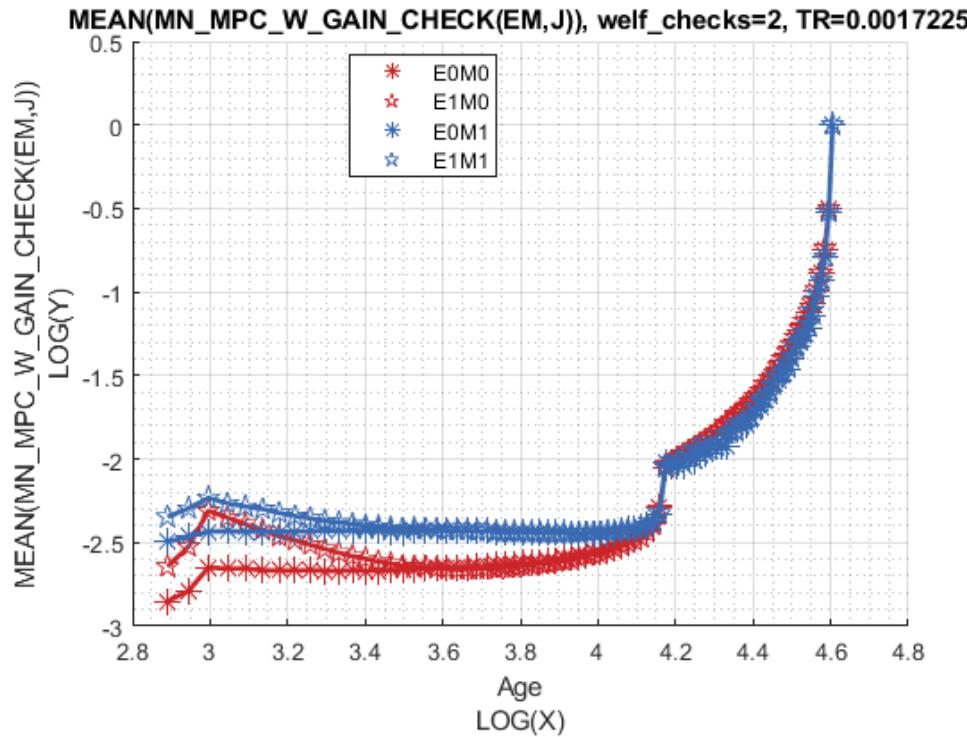




Graph Mean Consumption (*MPC: Share of Check Consumed*):

```
st_title = ['MEAN(MN\_MPC\_W\_GAIN\_CHECK(EM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(TR)];
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_W\_GAIN\_CHECK(EM,J))'};
ff_graph_grid((tb_az_c{1:end}, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```





7.2 2020 V and C with Unemployment

This is the example vignette for function: [snw_a4chk_unemp_bisec_vec](#) from the [PrjOptiSNW Package](#). This function solves for the V(states, check) for individuals working. Dense solution. Bisection, most time for the test here taken to generate the income matrixes. But these can be generated out of the check loops.

7.2.1 Test SNW_A4CHK_UNEMP_BISEC_VEC Defaults

Solve for Value/Policy in non-COVID years, then solve for covid year value/policy given covid shocks. COVID lasts one period.

```
mp_params = snw_mp_param('default_docdense', false, 'tauchen', true);
mp_controls = snw_mp_control('default_test');
mp_controls('bl_print_vfi') = false;
mp_controls('bl_timer') = true;
[V_ss,~,cons_ss,~] = snw_vfi_main_bisec_vec(mp_params, mp_controls);
```

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=494.

```
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	ndim	numel	rowN	colN	sum	mean	std
	-	---	----	-----	----	-----	-----	-----	-----
V_VFI	1	1	6	4.37e+07	83	5.265e+05	-1.5339e+08	-3.5101	26.11
ap_VFI	2	2	6	4.37e+07	83	5.265e+05	1.4159e+09	32.402	36.79
cons_VFI	3	3	6	4.37e+07	83	5.265e+05	2.1402e+08	4.8975	8.329

```
xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxx
      c1          c2          c3          c4          c5      c526496      c526497      c526498      c
-----  -----  -----  -----  -----  -----  -----  -----  -----  -----

```

r1	-346.51	-346.12	-343.63	-337.86	-328.51	21.702	21.852	22.003
r2	-334.38	-333.99	-331.51	-325.83	-316.83	21.724	21.869	22.015
r3	-322.45	-322.06	-319.6	-314.14	-305.6	21.745	21.885	22.027
r4	-310.63	-310.27	-307.99	-302.88	-294.87	21.767	21.903	22.041
r5	-299.94	-299.6	-297.46	-292.67	-285.12	21.775	21.907	22.042
r79	-9.9437	-9.9325	-9.8557	-9.6597	-9.3232	2.5394	2.5501	2.5602
r80	-8.9023	-8.8911	-8.8143	-8.6183	-8.2818	2.3039	2.3121	2.3198
r81	-7.6363	-7.6251	-7.5484	-7.3524	-7.0159	2.0068	2.0124	2.0176
r82	-5.9673	-5.9561	-5.8793	-5.6833	-5.3468	1.5958	1.5989	1.6018
r83	-3.5892	-3.578	-3.5012	-3.3052	-2.9687	0.97904	0.98004	0.98097

xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499
	--	--	-----	-----	-----	-----	-----	-----	-----
r1	0	0	0.0005656	0.0075134	0.022901	114.75	120.41	126.27	132.3
r2	0	0	0.00051498	0.0065334	0.021549	114.86	120.53	126.41	132.5
r3	0	0	0.00051498	0.0049294	0.019875	114.97	120.65	126.56	132.
r4	0	0	0.00051498	0.0047937	0.019672	115.73	121.42	127.34	133.5
r5	0	0	0.00048517	0.0046683	0.019484	116.5	122.21	128.15	134.3
r79	0	0	0	0	0	81.091	85.68	90.335	94.37
r80	0	0	0	0	0	76.669	80.563	84.304	88.0
r81	0	0	0	0	0	68.313	71.534	74.475	77.83
r82	0	0	0	0	0	50.126	53.467	56.953	58.74
r83	0	0	0	0	0	0	0	0	0

xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498
	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.036717	0.037251	0.040426	0.04363	0.048012	9.6491	9.817	9.9649
r2	0.036717	0.037251	0.040477	0.04461	0.049364	9.8118	9.9685	10.101
r3	0.036717	0.037251	0.040477	0.046214	0.051039	9.9779	10.12	10.234
r4	0.038144	0.038678	0.041903	0.047776	0.052666	10.131	10.258	10.354
r5	0.039534	0.040068	0.043323	0.04929	0.054241	10.272	10.384	10.463
r79	0.2179	0.21844	0.22216	0.23228	0.25197	35.858	37.092	38.455
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.253	42.183	44.459
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.587	51.19	54.266
r82	0.2179	0.21844	0.22216	0.23228	0.25197	66.755	69.238	71.77
r83	0.2179	0.21844	0.22216	0.23228	0.25197	116.87	122.69	128.71

welf_checks = 2; % 2 checks is \$200 dollar of welfare checks

xi=0.5; % xi=0 full income loss from covid shock, xi=1, no covid income losses

b=0; % b=0 means no UI benefits compensating COVID, b=1 if full income replacement

TR = 100/58056;

mp_params('TR') = TR;

mp_params('xi') = xi;

mp_params('b') = b;

mp_params('a2_covidyr') = mp_params('a2_covidyr_manna_heaven');

% mp_params('a2_covidyr') = mp_params('a2_covidyr_tax_fully_pay');

[V_unemp_2020,~,cons_unemp_2020,~] = snw_vfi_main_bisec_vec(mp_params, mp_controls, V_ss);

Completed SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=d

xx

CONTAINER NAME: mp_outcomes ND Array (Matrix etc)

xx

i	idx	ndim	numel	rowN	colN	sum	mean	std
---	-----	------	-------	------	------	-----	------	-----

	-	---	----	-----	----	-----	-----	-----	-----	-----
V_VFI	1	1	6	4.37e+07	83	5.265e+05	-1.7805e+08	-4.0743	27.11	
ap_VFI	2	2	6	4.37e+07	83	5.265e+05	1.3789e+09	31.553	36.67	
cons_VFI	3	3	6	4.37e+07	83	5.265e+05	2.1097e+08	4.8277	8.328	
xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxxxxx										
	c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499	c526500
	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----
r1	-372.97	-371.47	-362.94	-349.52	-336.96	21.573	21.728	21.882		
r2	-360.84	-359.34	-350.81	-337.39	-324.98	21.595	21.745	21.894		
r3	-348.91	-347.41	-338.88	-325.46	-313.34	21.617	21.762	21.906		
r4	-336.09	-334.7	-326.73	-314.01	-302.44	21.633	21.772	21.913		
r5	-324.48	-323.18	-315.72	-303.62	-292.54	21.634	21.77	21.907		
r79	-9.9437	-9.9325	-9.8557	-9.6597	-9.3232	2.5374	2.5482	2.5584		
r80	-8.9023	-8.8911	-8.8143	-8.6183	-8.2818	2.3024	2.3107	2.3185		
r81	-7.6363	-7.6251	-7.5484	-7.3524	-7.0159	2.0057	2.0114	2.0168		
r82	-5.9673	-5.9561	-5.8793	-5.6833	-5.3468	1.5952	1.5984	1.6014		
r83	-3.5892	-3.578	-3.5012	-3.3052	-2.9687	0.97886	0.97987	0.98082	0	
xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxxxxx										
	c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499	c526500
	--	--	--	--	-----	-----	-----	-----	-----	-----
r1	0	0	0	0	0.0092181	110.06	115.71	121.55	127.62	133.93
r2	0	0	0	0	0.008238	110.03	115.68	121.54	127.62	133.95
r3	0	0	0	0	0.0066341	109.99	115.65	121.53	127.63	133.97
r4	0	0	0	0	0.0058019	110.28	115.95	121.84	127.96	134.33
r5	0	0	0	0	0.004998	110.58	116.27	122.17	128.31	134.69
r79	0	0	0	0	0	81.091	85.229	89.297	93.341	97.382
r80	0	0	0	0	0	75.865	79.539	83.28	87.016	90.669
r81	0	0	0	0	0	67.781	70.521	73.462	76.819	81.091
r82	0	0	0	0	0	50.126	53.467	56.108	57.742	60.587
r83	0	0	0	0	0	0	0	0	0	0
xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxx										
	c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499	c526500
	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.018623	0.019158	0.022901	0.033062	0.04363	9.4708	9.6491	9.817		
r2	0.018623	0.019158	0.022901	0.033062	0.04461	9.6414	9.8118	9.9685		
r3	0.018623	0.019158	0.022901	0.033062	0.046214	9.8179	9.9779	10.12		
r4	0.019354	0.019888	0.023632	0.033792	0.047776	9.9825	10.131	10.258		
r5	0.020066	0.020601	0.024344	0.034504	0.04929	10.135	10.272	10.384		
r79	0.2179	0.21844	0.22216	0.23228	0.25197	34.82	36.506	38.455		
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.033	42.183	44.459		
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.106	51.19	54.266		
r82	0.2179	0.21844	0.22216	0.23228	0.25197	65.751	68.234	71.611		
r83	0.2179	0.21844	0.22216	0.23228	0.25197	115.87	121.69	127.71		

[V_U_2020, C_U_2020] = snw_a4chk_unemp_bisec_vec(welf_checks, V_unemp_2020, cons_unemp_2020, mp_params)

Completed SNW_A4CHK_UNEMP_BISEC_VEC; welf_checks=2; TR=0.0017225; xi=0.5; b=0; SNW_MP_PARAM=default_docde

xx

CONTAINER NAME: mp_container_map ND Array (Matrix etc)

xx

	i	idx	ndim	numel	rowN	colN	sum	mean
	-	---	----	-----	---	-----	-----	-----
C_U	1	1	6	4.37e+07	83	5.265e+05	2.11e+08	4.828
C_U_minus_C_unemp	2	2	6	4.37e+07	83	5.265e+05	28536	0.00065
V_U	3	3	6	4.37e+07	83	5.265e+05	-1.7705e+08	-4.051
V_U_minus_V_unemp	4	4	6	4.37e+07	83	5.265e+05	9.9227e+05	0.02270
mn_MPC_unemp	5	5	6	4.37e+07	83	5.265e+05	8.2833e+06	0.1895

```

mn_V_U_gain_check = V_U_2020 - V_unemp_2020;
mn_MPC_U_gain_share_check = (C_U_2020 - cons_unemp_2020)./(welf_checks*mp_params('TR'));

```

7.2.2 Dense 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 = 18:100;
agrid = mp_params('agrid');
eta_H_grid = mp_params('eta_H_grid');
eta_S_grid = mp_params('eta_S_grid');
ar_st_eta_HS_grid = string(cellstr([num2str(eta_H_grid), 'hz=%3.2f;'], num2str(eta_S_grid), 'wz=%3.2f;'));
edu_grid = [0,1];
marry_grid = [0,1];
kids_grid = (1:1:mp_params('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});
```

7.2.3 Analyze Difference in V and C with Check

The difference between V and V with Check, marginal utility gain given the check.

```
% 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') = 'eastoutside';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('it_legend_select') = 21; % how many shock legends to show
mp_support_graph('cl_colors') = 'jet';

MEAN(MN_V_GAIN_CHECK(A,Z))
```

Tabulate value and policies along savings and shocks:

```
% Set
ar_permute = [1,4,5,6,3,2];
% Value Function
st_title = ['MEAN(MN_V_U_GAIN_CHECK(A,Z)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_params('TR'))'];
tb_az_v = ff_summ_nd_array(st_title, mn_V_U_gain_check, true, ["mean"], 4, 1, cl_mp_datasetdesc, ar_permute);

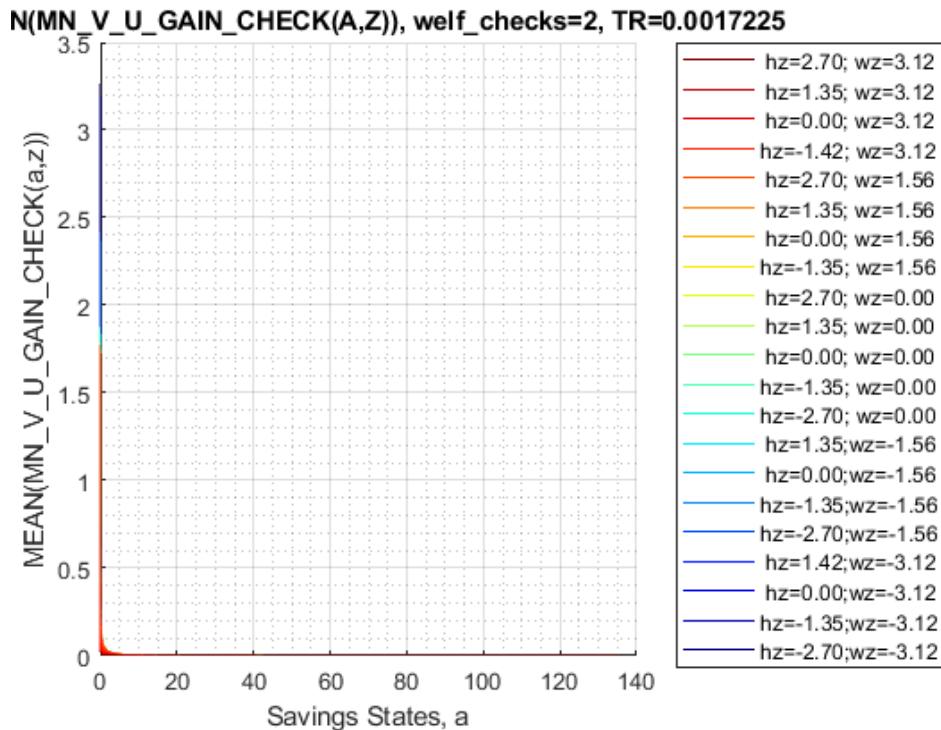
xxx MEAN(MN_V_U_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxx
group      savings      mean_eta_1      mean_eta_2      mean_eta_3      mean_eta_4      mean_eta_5      mean_eta_6
-----  -----  -----  -----  -----  -----  -----  -----
```

1	0	3.2686	2.9159	2.6002	2.318	2.0659
---	---	--------	--------	--------	-------	--------

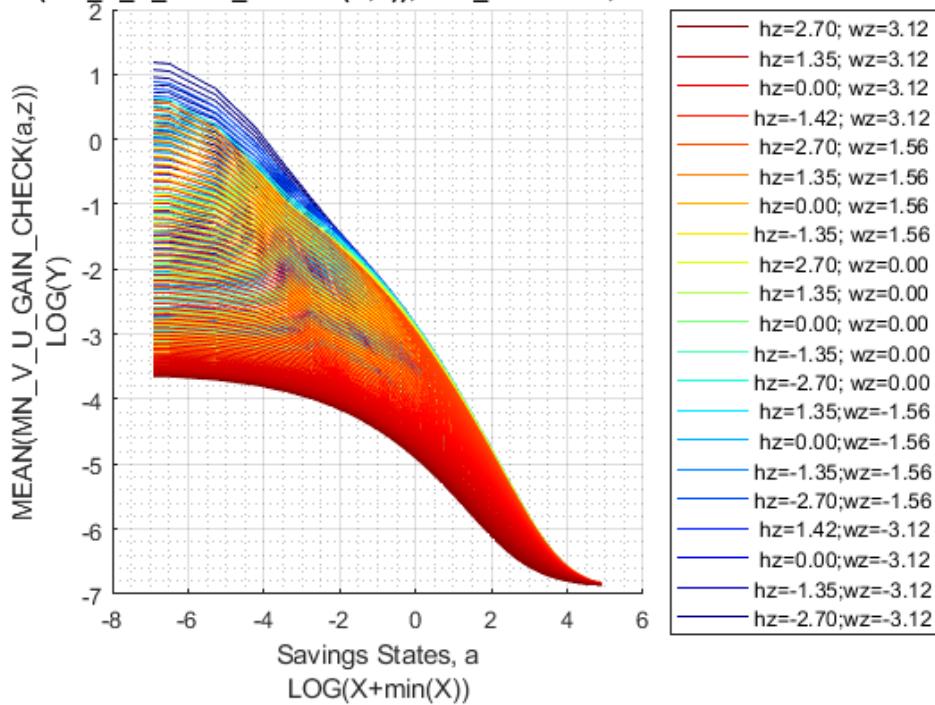
```

st_title = ['MEAN(MN\_V\_U\_GAIN\_CHECK(A,Z)), welf_checks=' num2str(welf_checks) ', TR=' num2str(m
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_V\_U\_GAIN\_CHECK(a,z))'};
ff_graph_grid((tb_az_v{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);

```



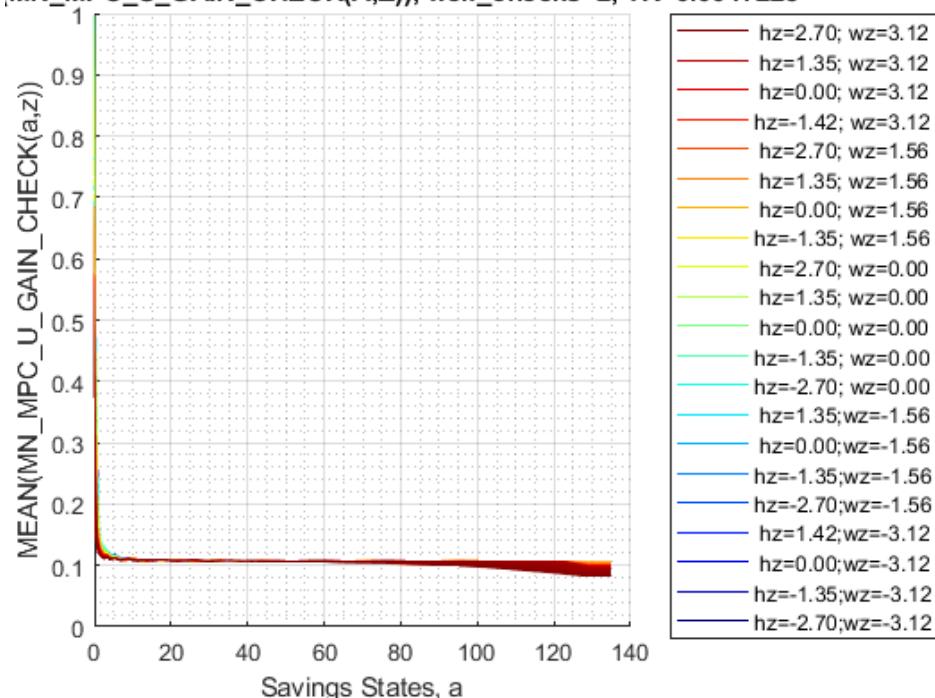
AN(MN_V_U_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225

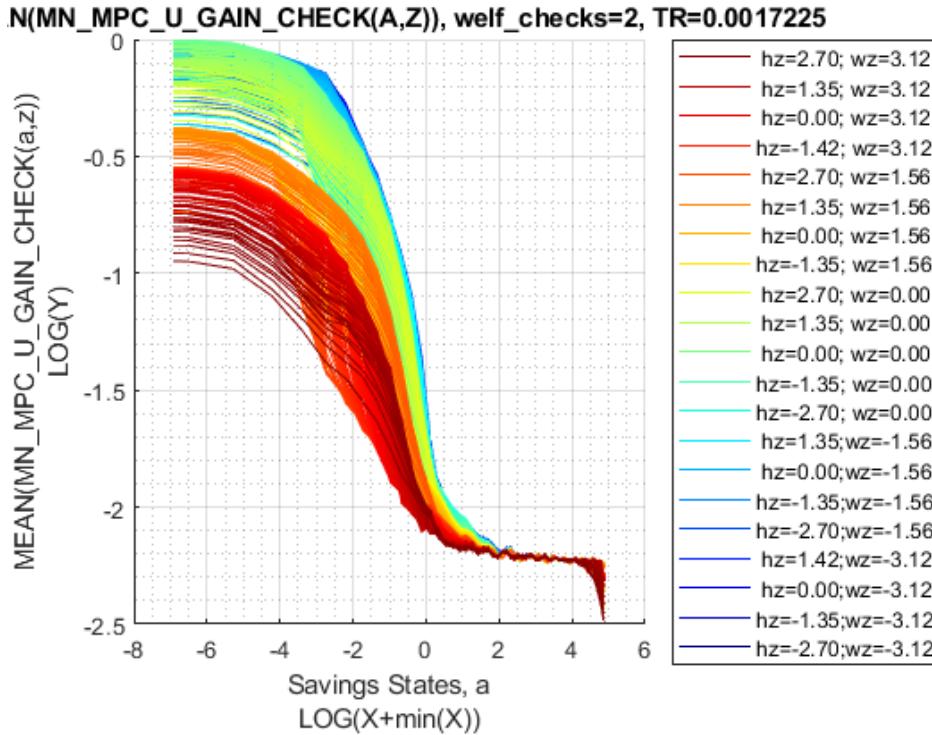


Graph Mean Consumption (*MPC: Share of Check Consumed*):

```
st_title = ['MEAN(MN\_MPC\_U\_GAIN\_CHECK(A,Z)), welf\_checks=' num2str(welf_checks) ', TR=' num2str
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_U\_GAIN\_CHECK(a,z))'};
ff_graph_grid((tb_az_c{1:end, 3:end}'), ar_st_eta_HS_grid, agrid, mp_support_graph);
```

MN_MPC_U_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225





7.2.4 Analyze Kids and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = [...
    "k0M0", "K1M0", "K2M0", "K3M0", "K4M0", ...
    "k0M1", "K1M1", "K2M1", "K3M1", "K4M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {...
    'o', 'd', 's', 'x', '*', ...
    'o', 'd', 's', 'x', '*'};
mp_support_graph('cl_colors') = {...
    'red', 'red', 'red', 'red', 'red',...
    'blue', 'blue', 'blue', 'blue', 'blue'};
```

MEAN(VAL(KM,J)), MEAN(AP(KM,J)), MEAN(C(KM,J))

Tabulate value and policies:

```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,4,1,6,5];
% Value Function
st_title = ['MEAN(MN_V_U_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_pa...
tb_az_v = ff_summ_nd_array(st_title, mn_V_U_gain_check, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_...
xxx MEAN(MN_V_U_GAIN_CHECK(KM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx
group kids marry mean_age_18 mean_age_19 mean_age_20 mean_age_21 mean_age_22
----- ----- ----- ----- ----- ----- ----- -----
1 1 0 0.05876 0.057807 0.056772 0.051707 0.047488
```

2	2	0	0.081525	0.080267	0.078856	0.071726	0.065777
3	3	0	0.097699	0.09639	0.094869	0.0863	0.079154
4	4	0	0.11184	0.11044	0.10878	0.098969	0.090785
5	5	0	0.12393	0.12252	0.1208	0.10994	0.10089
6	1	1	0.020733	0.019926	0.019164	0.01732	0.015771
7	2	1	0.02739	0.02635	0.025367	0.022922	0.020867
8	3	1	0.033087	0.031892	0.030749	0.02779	0.025312
9	4	1	0.039391	0.038018	0.03669	0.033174	0.030229
10	5	1	0.047955	0.046442	0.044963	0.040683	0.037108

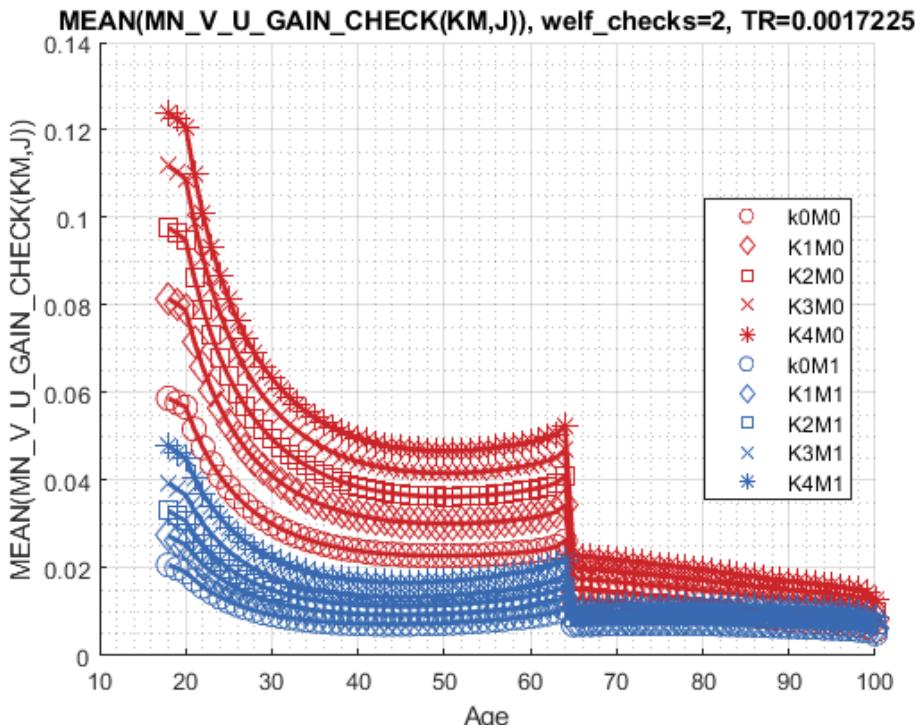
% Consumption Function

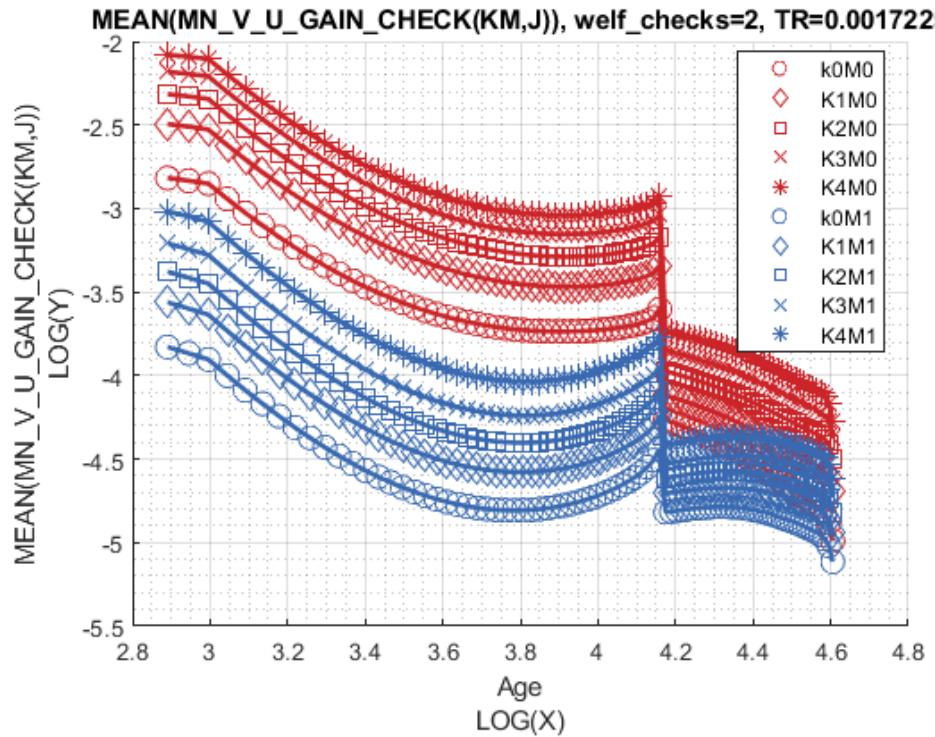
```
st_title = ['MEAN(MN_MPC_U_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
tb_az_c = ff_summ_nd_array(st_title, mn_MPC_U_gain_share_check, true, ["mean"], 3, 1, cl_mp_datasetd
```

group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	1	0	0.13845	0.14295	0.14757	0.14693	0.14633
2	2	0	0.15131	0.15536	0.15963	0.16001	0.16023
3	3	0	0.1601	0.16368	0.16762	0.16817	0.16857
4	4	0	0.16455	0.16795	0.17175	0.17234	0.17279
5	5	0	0.16853	0.17166	0.17526	0.17576	0.17614
6	1	1	0.13988	0.14409	0.14573	0.14586	0.14644
7	2	1	0.1441	0.14787	0.14961	0.15012	0.15078
8	3	1	0.15004	0.15388	0.15527	0.15614	0.15695
9	4	1	0.15447	0.15723	0.15901	0.15929	0.16005
10	5	1	0.16131	0.16321	0.16694	0.16691	0.16454

Graph Mean Values:

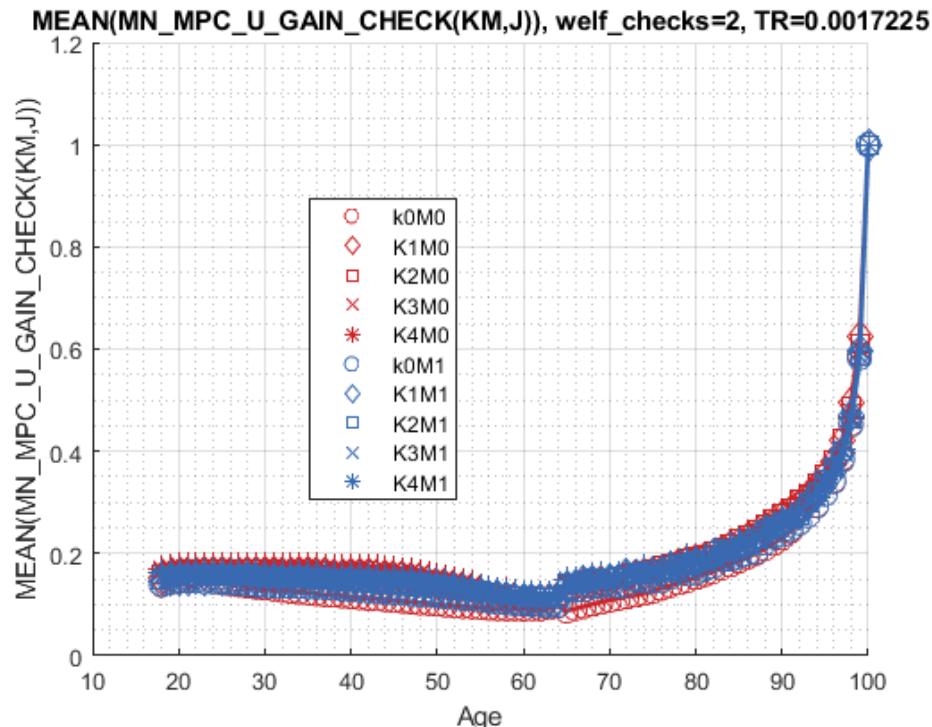
```
st_title = ['MEAN(MN_V_U_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN_V_U_GAIN_CHECK(KM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```

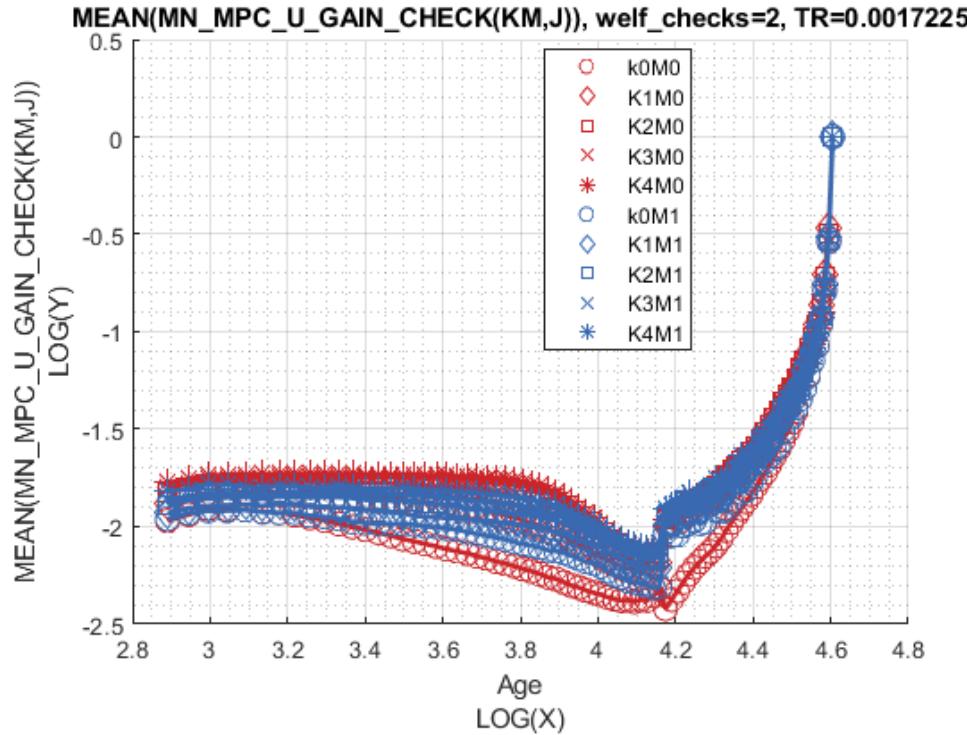




Graph Mean Consumption (*MPC: Share of Check Consumed*):

```
st_title = ['MEAN(MN\_MPC\_U\_GAIN\_CHECK(KM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2st
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_U\_GAIN\_CHECK(KM,J))'};
ff_graph_grid((tb_az_c{1:end}, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```





7.2.5 Analyze Education and Marriage

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["E0M0", "E1M0", "E0M1", "E1M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {'*', 'p', '*', 'p' };
mp_support_graph('cl_colors') = {'red', 'red', 'blue', 'blue'};
```

MEAN(VAL(EM,J)), MEAN(AP(EM,J)), MEAN(C(EM,J))

Tabulate value and policies:

```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,6,1,4,5];
% Value Function
st_title = ['MEAN(MN_V_U_GAIN_CHECK(EM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_pa
tb_az_v = ff_summ_nd_array(st_title, mn_V_U_gain_check, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_
```

group	edu	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	0	0	0.096397	0.095384	0.094239	0.088786	0.083945
2	1	0	0.093102	0.091587	0.089794	0.078671	0.069692
3	0	1	0.035401	0.034205	0.033063	0.030696	0.028627
4	1	1	0.032021	0.030846	0.029711	0.02606	0.023088

% Consumption

```
st_title = ['MEAN(MN_MPC_U_GAIN_CHECK(EM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
```

```

tb_az_c = ff_summ_nd_array(st_title, mn_MPC_U_gain_share_check, true, ["mean"], 3, 1, cl_mp_datasetd

xxx MEAN(MN_MPC_U_GAIN_CHECK(EM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx
group   edu    marry   mean_age_18   mean_age_19   mean_age_20   mean_age_21   mean_age_22
-----  ---  -----  -----  -----  -----  -----  -----
1       0      0      0.14718     0.15045     0.15381     0.15404     0.15431
2       1      0      0.166       0.1702      0.17493     0.17525     0.17532
3       0      1      0.14287     0.14583     0.14746     0.14781     0.14772
4       1      1      0.15705     0.16068     0.16316     0.16351     0.16378

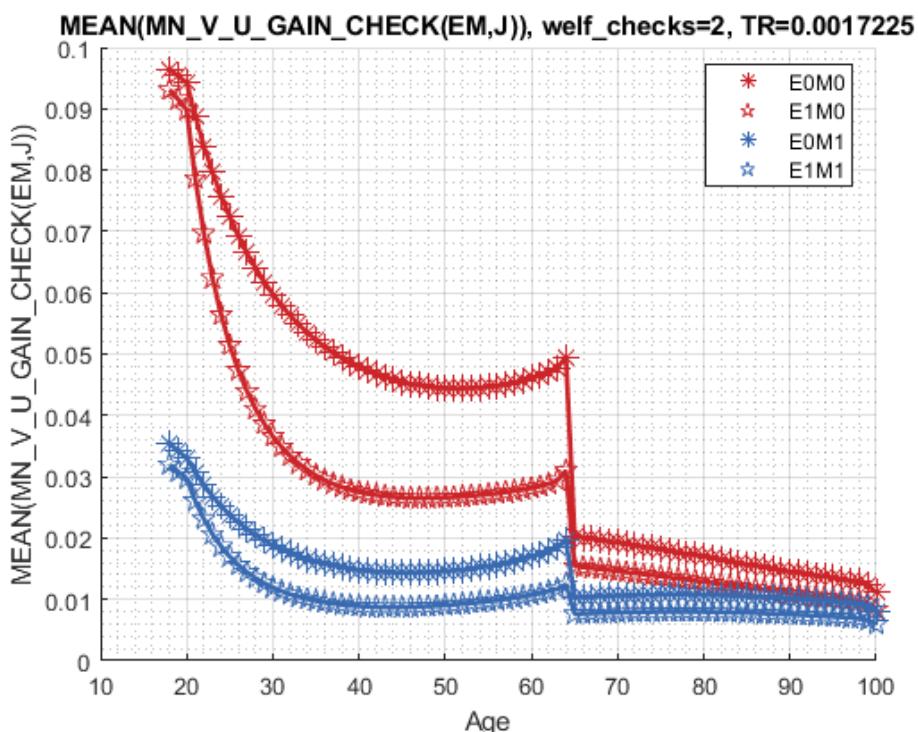
```

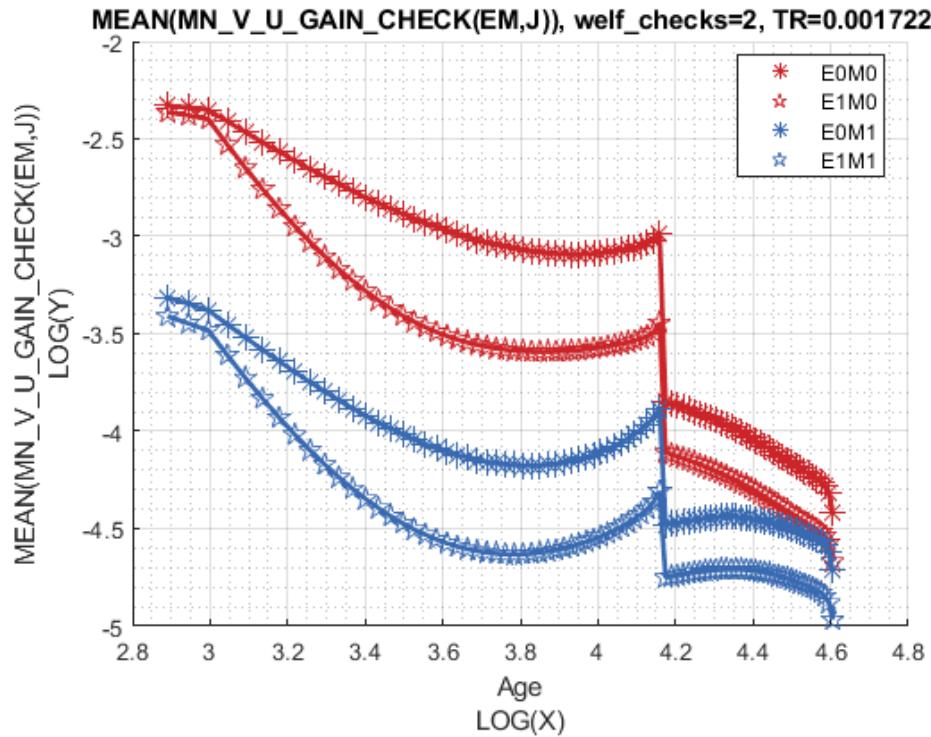
Graph Mean Values:

```

st_title = ['MEAN(MN\_V\_U\_GAIN\_CHECK(EM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_V\_U\_GAIN\_CHECK(EM,J))'};
ff_graph_grid((tb_az_v{1:end}, 4:end}), ar_row_grid, age_grid, mp_support_graph);

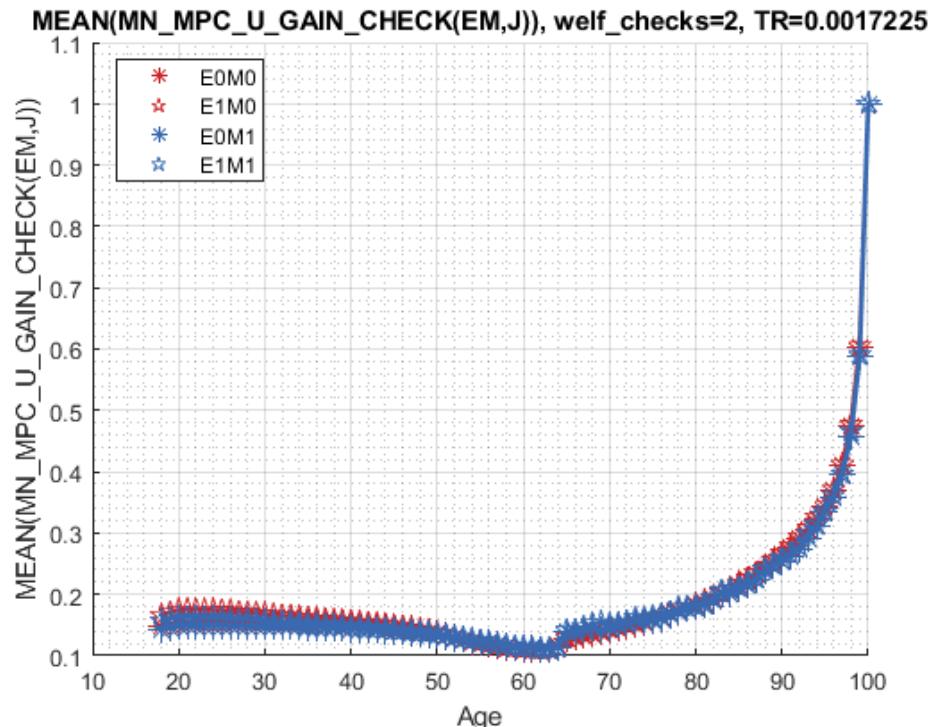
```

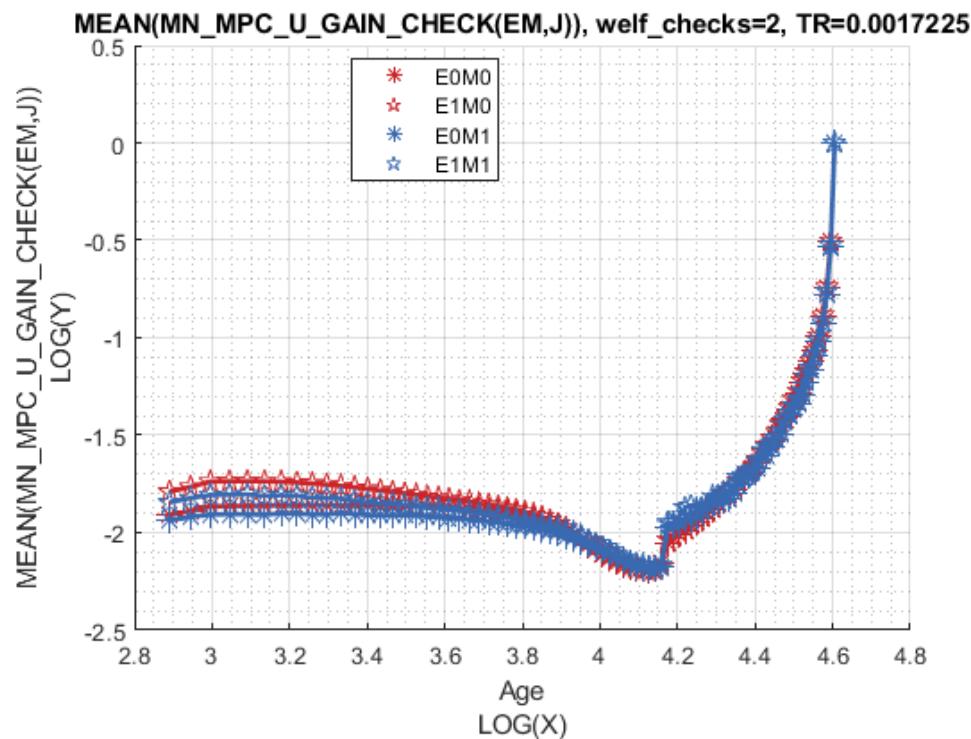




Graph Mean Consumption (*MPC: Share of Check Consumed*):

```
st_title = ['MEAN(MN\_\_MPC\_\_U\_\_GAIN\_\_CHECK(EM,J)), welf\_\_checks=' num2str(welf_checks) ', TR=' num2st
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_\_MPC\_\_U\_\_GAIN\_\_CHECK(EM,J))'};
ff_graph_grid((tb_az_c{1:end}, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```





Chapter 8

2020 Outcomes Full State Space with Savings, Shocks and Education

8.1 2020 Full States EV and EC of One Check

This is the example vignette for function: `snw_evuvw20_jaeemk` from the [PrjOptiSNW Package](#). 2020 integrated over VU and VW. Average C or V given unemployment probabilities.

8.1.1 Test SNW_EVUVW20_JAEEMK Defaults

Call the function with defaults.

```
clear all;
st_solu_type = 'biseq_vec';

% Solve the VFI Problem and get Value Function
mp_params = snw_mp_param('default_docdense');
mp_controls = snw_mp_control('default_test');

% set Unemployment Related Variables
xi=0.5; % Proportional reduction in income due to unemployment (xi=0 refers to 0 labor income; xi=1
b=0; % Unemployment insurance replacement rate (b=0 refers to no UI benefits; b=1 refers to 100 perc
TR=100/58056; % Value of a welfare check (can receive multiple checks). TO DO: Update with alternati

mp_params('xi') = xi;
mp_params('b') = b;
mp_params('TR') = TR;

% Solve for Unemployment Values
mp_controls('bl_print_vfi') = false;
mp_controls('bl_print_ds') = false;
mp_controls('bl_print_ds_verbose') = false;
mp_controls('bl_print_precompute') = false;
mp_controls('bl_print_precompute_verbose') = false;
mp_controls('bl_print_a4chk') = false;
mp_controls('bl_print_a4chk_verbose') = false;
mp_controls('bl_print_evuvw20_jaeemk') = false;
mp_controls('bl_print_evuvw20_jaeemk_verbose') = false;

Solve the model:

%% A. Solve VFI
% 2. Solve VFI and Distributon
% Solve the Model to get V working and unemployed
```

```
% solved with calibrated regular a2
[V_ss,ap_ss,cons_ss,mp_valpol_more_ss] = snw_vfi_main_bisec_vec(mp_params, mp_controls);

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=517.

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

    i      idx     ndim    numel    rowN    colN        sum    mean    std
    -      ---     ----  -----  -----  -----  -----  -----  -----
V_VFI    1       1       6   4.37e+07    83  5.265e+05 -1.5339e+08 -3.5101 26.11
ap_VFI   2       2       6   4.37e+07    83  5.265e+05  1.4159e+09 32.402 36.79
cons_VFI 3       3       6   4.37e+07    83  5.265e+05  2.1402e+08 4.8975 8.329

xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxx
    c1      c2      c3      c4      c5    c526496    c526497    c526498    c
    ----  -----  -----  -----  -----  -----  -----  -----  -----
r1   -346.51  -346.12  -343.63  -337.86  -328.51  21.702   21.852   22.003
r2   -334.38  -333.99  -331.51  -325.83  -316.83  21.724   21.869   22.015
r3   -322.45  -322.06  -319.6   -314.14  -305.6   21.745   21.885   22.027
r4   -310.63  -310.27  -307.99  -302.88  -294.87  21.767   21.903   22.041
r5   -299.94  -299.6   -297.46  -292.67  -285.12  21.775   21.907   22.042
r79  -9.9437 -9.9325 -9.8557 -9.6597 -9.3232  2.5394  2.5501  2.5602
r80  -8.9023 -8.8911 -8.8143 -8.6183 -8.2818  2.3039  2.3121  2.3198
r81  -7.6363 -7.6251 -7.5484 -7.3524 -7.0159  2.0068  2.0124  2.0176
r82  -5.9673 -5.9561 -5.8793 -5.6833 -5.3468  1.5958  1.5989  1.6018
r83  -3.5892 -3.578   -3.5012 -3.3052 -2.9687  0.97904 0.98004 0.98097 0

xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxx
    c1      c2      c3      c4      c5    c526496    c526497    c526498    c5264
    --      --  -----  -----  -----  -----  -----  -----
r1    0       0   0.0005656  0.0075134  0.022901  114.75   120.41   126.27 132.3
r2    0       0   0.00051498 0.0065334  0.021549  114.86   120.53   126.41 132.5
r3    0       0   0.00051498 0.0049294  0.019875  114.97   120.65   126.56 132.
r4    0       0   0.00051498 0.0047937  0.019672  115.73   121.42   127.34 133.5
r5    0       0   0.00048517 0.0046683  0.019484  116.5    122.21   128.15 134.3
r79   0       0       0       0       0   81.091   85.68   90.335 94.37
r80   0       0       0       0       0   76.669   80.563  84.304 88.0
r81   0       0       0       0       0   68.313   71.534  74.475 77.83
r82   0       0       0       0       0   50.126   53.467  56.953 58.74
r83   0       0       0       0       0       0       0       0       0

xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxx
    c1      c2      c3      c4      c5    c526496    c526497    c526498
    ----  -----  -----  -----  -----  -----  -----
r1   0.036717 0.037251 0.040426 0.04363 0.048012 9.6491  9.817  9.9649
r2   0.036717 0.037251 0.040477 0.04461 0.049364 9.8118  9.9685 10.101
r3   0.036717 0.037251 0.040477 0.046214 0.051039 9.9779  10.12  10.234
r4   0.038144 0.038678 0.041903 0.047776 0.052666 10.131  10.258 10.354
r5   0.039534 0.040068 0.043323 0.04929 0.054241 10.272  10.384 10.463
r79  0.2179  0.21844 0.22216 0.23228 0.25197 35.858  37.092 38.455
r80  0.2179  0.21844 0.22216 0.23228 0.25197 40.253  42.183 44.459
r81  0.2179  0.21844 0.22216 0.23228 0.25197 48.587  51.19  54.266
r82  0.2179  0.21844 0.22216 0.23228 0.25197 66.755  69.238 71.77
```

```

r83      0.2179    0.21844   0.22216   0.23228   0.25197   116.87    122.69    128.71

% COVID year tax
mp_params('a2_covidyr') = mp_params('a2_covidyr_manna_heaven');
% 2020 V and C same as V_SS and cons_ss if tax the same
if (mp_params('a2_covidyr') == mp_params('a2'))
    % mana from heaven
    V_ss_2020 = V_ss;
    cons_ss_2020 = cons_ss;
else
    % change xi and b to for people without unemployment shock
    % solving for employed but 2020 tax results
    % a2_covidyr > a2, we increased tax in 2020 to pay for covid and other
    % costs resolve for both employed and unemployed
    xi = mp_params('xi');
    b = mp_params('b');
    mp_params('xi') = 1;
    mp_params('b') = 0;
    [V_ss_2020,~,cons_ss_2020,~] = snw_vfi_main_bisec_vec(mp_params, mp_controls, V_ss);
    mp_params('xi') = xi;
    mp_params('b') = b;
end

% Solve unemployment, with three input parameters, auto will use a2_covidyr
% as tax, similar for employed call above
[V_unemp_2020,~,cons_unemp_2020] = snw_vfi_main_bisec_vec(mp_params, mp_controls, V_ss);

Completed SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=d
-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

      i     idx    ndim    numel    rowN    colN        sum      mean      std
      -     ---    ----    -----    ---    -----    -----    -----    -----
V_VFI    1      1      6    4.37e+07    83    5.265e+05   -1.7805e+08   -4.0743   27.11
ap_VFI   2      2      6    4.37e+07    83    5.265e+05    1.3789e+09   31.553   36.67
cons_VFI 3      3      6    4.37e+07    83    5.265e+05    2.1097e+08   4.8277   8.328

xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxx
      c1      c2      c3      c4      c5    c526496    c526497    c526498    c
      ----  -----  -----  -----  -----  -----  -----  -----  -----
r1    -372.97  -371.47  -362.94  -349.52  -336.96   21.573   21.728   21.882
r2    -360.84  -359.34  -350.81  -337.39  -324.98   21.595   21.745   21.894
r3    -348.91  -347.41  -338.88  -325.46  -313.34   21.617   21.762   21.906
r4    -336.09  -334.7   -326.73  -314.01  -302.44   21.633   21.772   21.913
r5    -324.48  -323.18  -315.72  -303.62  -292.54   21.634   21.77   21.907
r79   -9.9437  -9.9325  -9.8557  -9.6597  -9.3232   2.5374   2.5482   2.5584
r80   -8.9023  -8.8911  -8.8143  -8.6183  -8.2818   2.3024   2.3107   2.3185
r81   -7.6363  -7.6251  -7.5484  -7.3524  -7.0159   2.0057   2.0114   2.0168
r82   -5.9673  -5.9561  -5.8793  -5.6833  -5.3468   1.5952   1.5984   1.6014
r83   -3.5892  -3.578   -3.5012  -3.3052  -2.9687   0.97886  0.97987  0.98082  0

xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxx
      c1      c2      c3      c4      c5    c526496    c526497    c526498    c526499    c526500
      --      --      --      --      --    -----  -----  -----  -----  -----

```

r1	0	0	0	0	0.0092181	110.06	115.71	121.55	127.62	133.93
r2	0	0	0	0	0.008238	110.03	115.68	121.54	127.62	133.95
r3	0	0	0	0	0.0066341	109.99	115.65	121.53	127.63	133.97
r4	0	0	0	0	0.0058019	110.28	115.95	121.84	127.96	134.33
r5	0	0	0	0	0.004998	110.58	116.27	122.17	128.31	134.69
r79	0	0	0	0	0	81.091	85.229	89.297	93.341	97.382
r80	0	0	0	0	0	75.865	79.539	83.28	87.016	90.669
r81	0	0	0	0	0	67.781	70.521	73.462	76.819	81.091
r82	0	0	0	0	0	50.126	53.467	56.108	57.742	60.587
r83	0	0	0	0	0	0	0	0	0	0

xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498
r1	0.018623	0.019158	0.022901	0.033062	0.04363	9.4708	9.6491	9.817
r2	0.018623	0.019158	0.022901	0.033062	0.04461	9.6414	9.8118	9.9685
r3	0.018623	0.019158	0.022901	0.033062	0.046214	9.8179	9.9779	10.12
r4	0.019354	0.019888	0.023632	0.033792	0.047776	9.9825	10.131	10.258
r5	0.020066	0.020601	0.024344	0.034504	0.04929	10.135	10.272	10.384
r79	0.2179	0.21844	0.22216	0.23228	0.25197	34.82	36.506	38.455
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.033	42.183	44.459
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.106	51.19	54.266
r82	0.2179	0.21844	0.22216	0.23228	0.25197	65.751	68.234	71.611
r83	0.2179	0.21844	0.22216	0.23228	0.25197	115.87	121.69	127.71

% B. Solve Dist

[Phi_true] = snw_ds_main_vec(mp_params, mp_controls, ap_ss, cons_ss);

Completed SNW_DS_MAIN_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=876.6781

Previous code

```
% % Solve the Model to get V working and unemployed
% [V_ss,ap_ss,cons_ss,mp_valpol_more_ss] = snw_vfi_main_bisec_vec(mp_params, mp_controls);
% % Solve unemployment
% [V_unemp,~,cons_unemp,~] = snw_vfi_main_bisec_vec(mp_params, mp_controls, V_ss);
% [Phi_true] = snw_ds_main(mp_params, mp_controls, ap_ss, cons_ss, mp_valpol_more_ss);
```

8.1.2 Precompute

```
inc_VFI = mp_valpol_more_ss('inc_VFI');
spouse_inc_VFI = mp_valpol_more_ss('spouse_inc_VFI');
total_inc_VFI = inc_VFI + spouse_inc_VFI;
% Get Matrixes
cl_st_precompute_list = {'a', ...
    'inc', 'inc_unemp', 'spouse_inc', 'spouse_inc_unemp', 'ref_earn_wageind_grid'};
mp_controls('bl_print_precompute_verbose') = false;
[mp_precompute_res] = snw_hh_precompute(mp_params, mp_controls, cl_st_precompute_list, ap_ss, Phi_tr
```

Wage quintile cutoffs=0.4645 0.71528 1.0335 1.5632

Completed SNW_HH_PRECOMPUTE;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time cost=318.

8.1.3 Solve for 2020 Evuvw With 0 and 2 Checks

```
% Call Function
welf_checks = 0;
[ev20_jaeemk_check0, ec20_jaeemk_check0] = snw_evuvw20_jaeemk(...
    welf_checks, st_solu_type, mp_params, mp_controls, ...
```

```

V_ss_2020, cons_ss_2020, V_unemp_2020, cons_unemp_2020, mp_precompute_res);

Completed SNW_A4CHK_WRK_BISEC_VEC;welf_checks=0;TR=0.0017225;SNW_MP_PARAM=default_docdense;SNW_MP_CO
Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=0;TR=0.0017225;xi=0.5;b=0;SNW_MP_PARAM=default_docde
Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;timeEUEC=8.07

% Call Function
welf_checks = 2;
[ev20_jaeemk_check2, ec20_jaeemk_check2] = snw_evuvw20_jaeemk(... ...
welf_checks, st_solu_type, mp_params, mp_controls, ...
V_ss_2020, cons_ss_2020, V_unemp_2020, cons_unemp_2020, mp_precompute_res);

Completed SNW_A4CHK_WRK_BISEC_VEC;welf_checks=2;TR=0.0017225;SNW_MP_PARAM=default_docdense;SNW_MP_CO
Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=2;TR=0.0017225;xi=0.5;b=0;SNW_MP_PARAM=default_docde
Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;timeEUEC=7.84

Differences between Checks in Expected Value and Expected Consumption

mn_V_U_gain_check = ev20_jaeemk_check2 - ev20_jaeemk_check0;
mn_MPC_U_gain_share_check = (ec20_jaeemk_check2 - ec20_jaeemk_check0)./(welf_checks*mp_params('TR'))

```

8.1.4 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 = 18:100;
agrid = mp_params('agrid');
eta_H_grid = mp_params('eta_H_grid');
eta_S_grid = mp_params('eta_S_grid');
ar_st_eta_HS_grid = string(cellstr([num2str(eta_H_grid), 'hz=%3.2f;'], num2str(eta_S_grid), 'wz=%3.2f'));
edu_grid = [0,1];
marry_grid = [0,1];
kids_grid = (1:1:mp_params('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});

```

8.1.5 Analyze Difference in V and C with Check

The difference between V and V with Check, marginal utility gain given the check.

```

% 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') = 'eastoutside';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('it_legend_select') = 21; % how many shock legends to show
mp_support_graph('cl_colors') = 'jet';

MEAN(MN_V_GAIN_CHECK(A,Z))

```

Tabulate value and policies along savings and shocks:

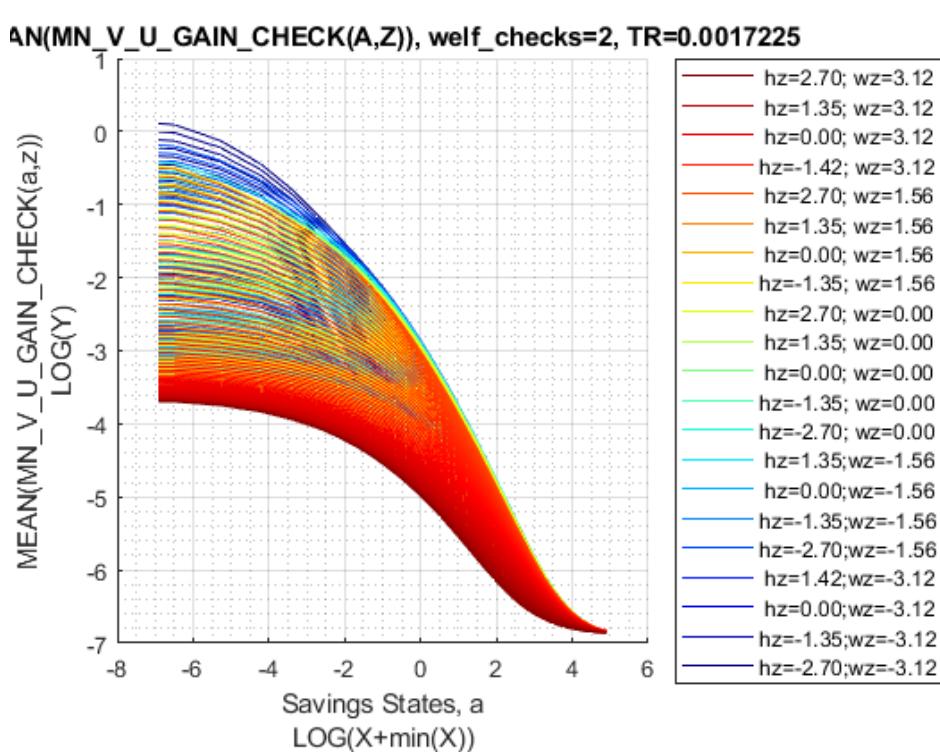
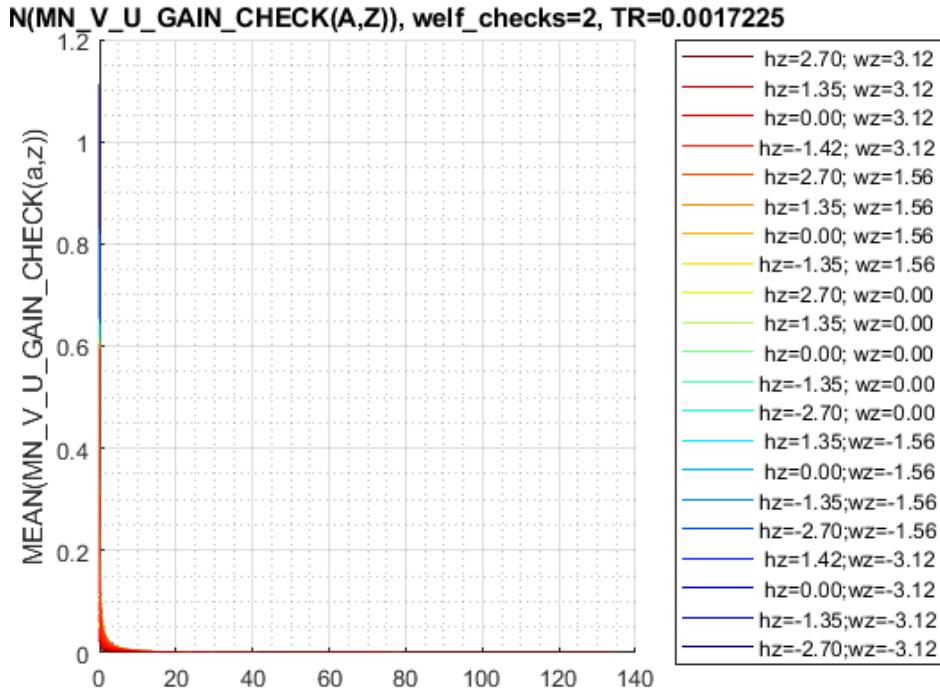
```

% Set
ar_permute = [1,4,5,6,3,2];

```

```
% Value Function
st_title = ['MEAN(MN_V_U_GAIN_CHECK(A,Z)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_par
tb_az_v = ff_summ_nd_array(st_title, mn_V_U_gain_check, true, ["mean"], 4, 1, cl_mp_datasetdesc, ar_
xxx  MEAN(MN_V_U_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx
group      savings      mean_eta_1      mean_eta_2      mean_eta_3      mean_eta_4      mean_eta_5      mea
-----      -----      -----      -----      -----      -----      -----      -----
1           0           1.1134        0.99534        0.88994        0.79581        0.71183
```

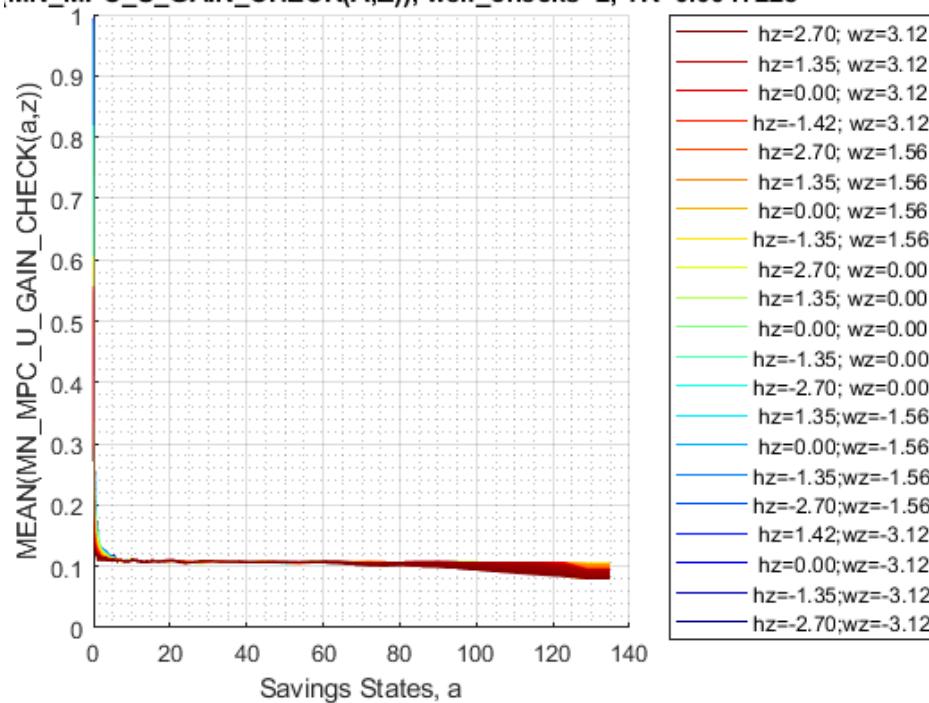
```
st_title = ['MEAN(MN\_\_V\_\_U\_\_GAIN\_\_CHECK(A,Z)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(mp_
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_\_V\_\_U\_\_GAIN\_\_CHECK(a,z))'};
ff_graph_grid((tb_az_v{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);
```



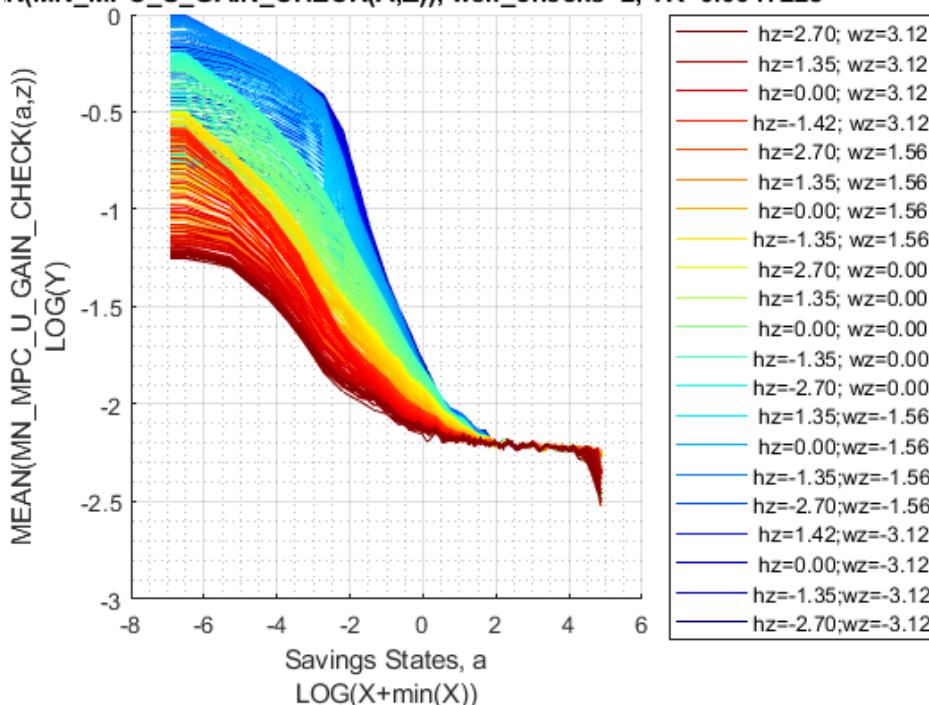
Graph Mean Consumption (MPC: Share of Check Consumed):

```
st_title = ['MEAN(MN\_MPC\_U\_GAIN\_CHECK(A,Z)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_U\_GAIN\_CHECK(a,z))'};
ff_graph_grid((tb_az_c{1:end, 3:end}), ar_st_eta_HS_grid, agrid, mp_support_graph);
```

MN_MPC_U_GAIN_CHECK(A,Z), welf_checks=2, TR=0.0017225



N(MN_MPC_U_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225



8.1.6 Analyze Marginal Value and MPC over Y(a,eta), Conditional On Kids, Marry, Age, Education

Income is generated by savings and shocks, what are the income levels generated by all the shock and savings points conditional on kids, marital status, age and educational levels. Plot on the Y axis MPC, and plot on the X axis income levels, use colors to first distinguish between different a levels, then use colors to distinguish between different eta levels.

Set Up date, Select Age 38, unmarried, no kids, lower education:

```
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
% 38 year old, unmarried, no kids, lower educated
% Only Household Head Shock Matters so select up to 'n_eta_H_grid'
mn_total_inc_jemk = total_inc_VFI(20,:,1:mp_params('n_eta_H_grid'),1,1,1);
mn_V_W_gain_check_use = ev20_jaeemk_check2 - ev20_jaeemk_check0;
mn_C_W_gain_check_use = ec20_jaeemk_check2 - ec20_jaeemk_check0;
```

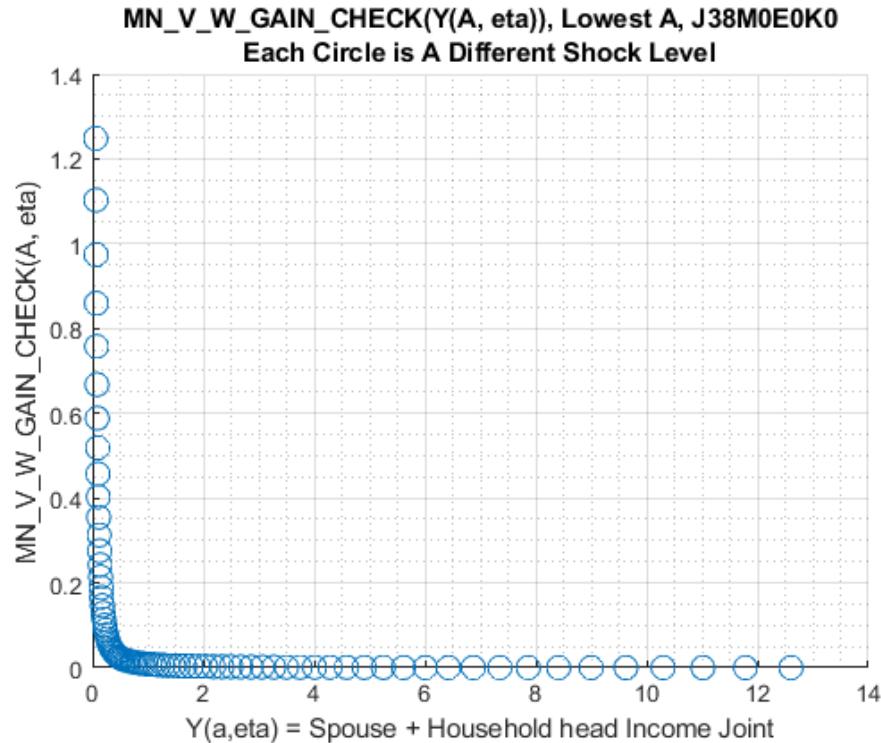
Select Age, Education, Marital, Kids Count:

```
% Selections
it_age = 21; % +18
it_marital = 1; % 1 = unmarried
it_kids = 1; % 1 = kids is zero
it_educ = 1; % 1 = lower education
% Select: NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
mn_C_W_gain_check_jemk = mn_C_W_gain_check_use(it_age, :, 1:mp_params('n_eta_H_grid'), it_educ, it_m
mn_V_W_gain_check_jemk = mn_V_W_gain_check_use(it_age, :, 1:mp_params('n_eta_H_grid'), it_educ, it_m
% Reshape, so shock is the first dim, a is the second
mt_total_inc_jemk = permute(mn_total_inc_jemk,[3,2,1]);
mt_C_W_gain_check_jemk = permute(mn_C_W_gain_check_jemk,[3,2,1]);
mt_C_W_gain_check_jemk(mt_C_W_gain_check_jemk<=1e-10) = 1e-10;
mt_V_W_gain_check_jemk = permute(mn_V_W_gain_check_jemk,[3,2,1]);
mt_V_W_gain_check_jemk(mt_V_W_gain_check_jemk<=1e-10) = 1e-10;
% Generate meshed a and shock grid
[mt_eta_H, mt_a] = ndgrid(eta_H_grid(1:mp_params('n_eta_H_grid')), agrid);
```

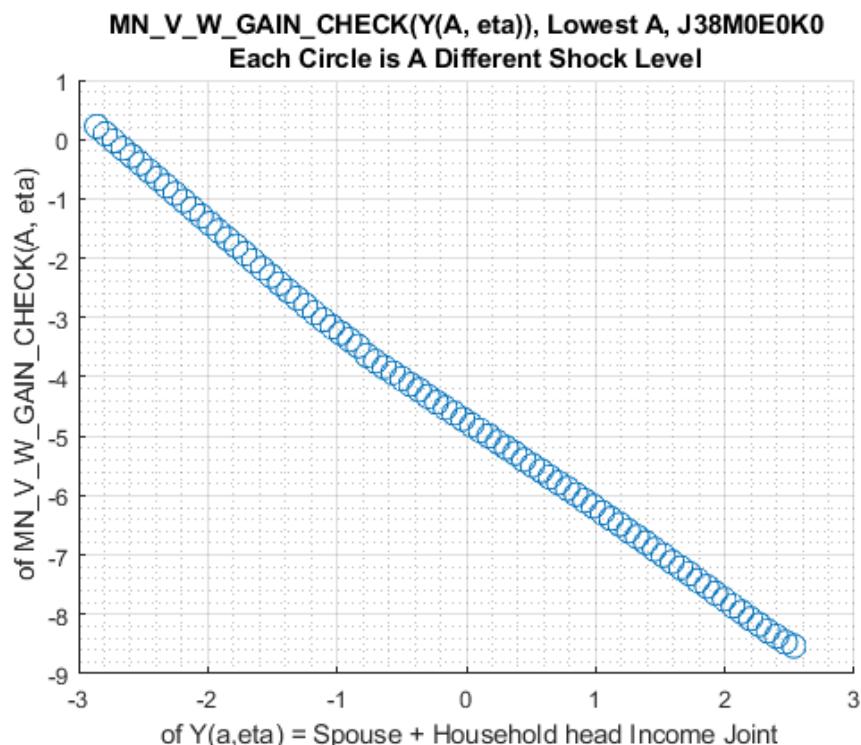
8.1.7 Marginal Value Gains, Color as Shock, Conditional on Age, Marital, Kids, and Education

How do shocks and a impact marginal value. First plot one asset level, variation comes only from increasingly higher shocks:

```
figure();
it_a = 1;
scatter((mt_total_inc_jemk(:,it_a)), (mt_V_W_gain_check_jemk(:,it_a)), 100);
title({'MN\_V\_W\_GAIN\_CHECK(Y(A, eta)), Lowest A, J38M0EOK0', ...
    'Each Circle is A Different Shock Level'});
xlabel('Y(a,eta) = Spouse + Household head Income Joint');
ylabel('MN\_V\_W\_GAIN\_CHECK(A, eta)');
grid on;
grid minor;
```

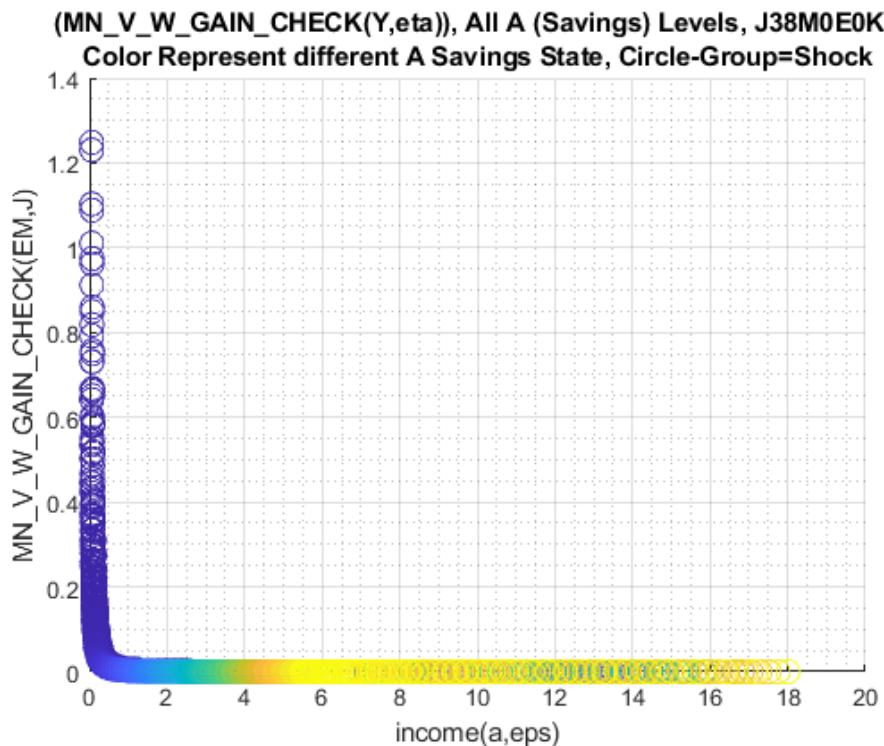


```
figure();
it_shock = 1;
scatter(log(mt_total_inc_jemk(:,it_a)), log(mt_V_W_gain_check_jemk(:,it_a)), 100);
title({'MN\_V\_W\_GAIN\_CHECK(Y(A, eta)), Lowest A, J38M0E0K0', ...
    'Each Circle is A Different Shock Level'});
xlabel(' of Y(a, eta) = Spouse + Household head Income Joint');
ylabel(' of MN\_V\_W\_GAIN\_CHECK(A, eta)');
grid on;
grid minor;
```

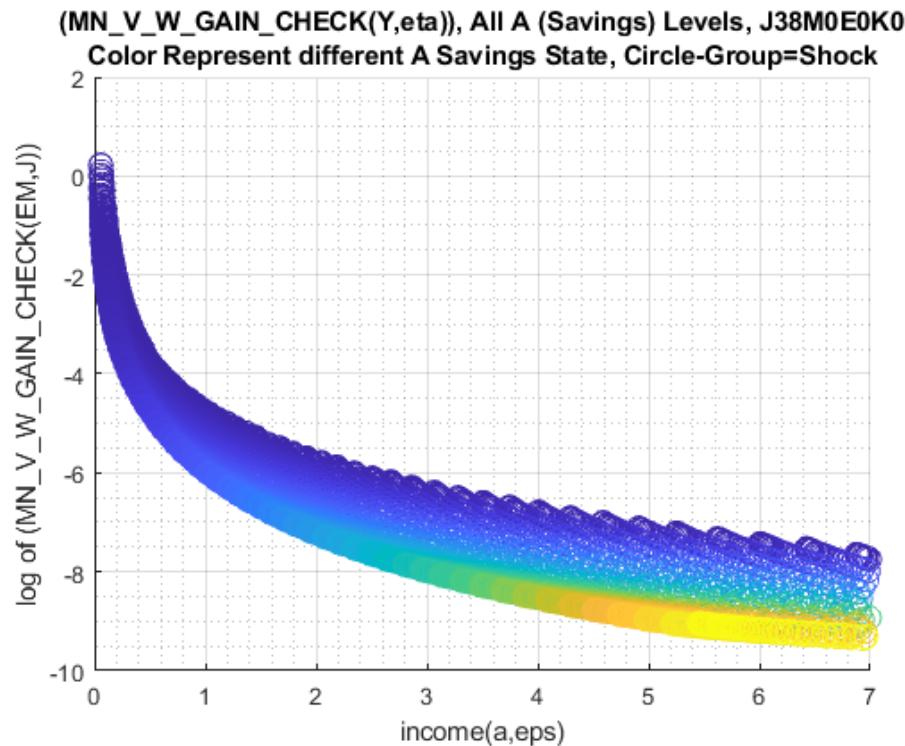


Plot all asset levels:

```
figure();
scatter((mt_total_inc_jemk(:)), (mt_V_W_gain_check_jemk(:)), 100, mt_a(:));
title({'(MN\_V\_W\_GAIN\_CHECK(Y,eta)), All A (Savings) Levels, J38M0E0K0', ...
    'Color Represent different A Savings State, Circle-Group=Shock'});
xlabel('income(a,eps)');
ylabel('MN\_V\_W\_GAIN\_CHECK(EM,J)');
grid on;
grid minor;
```



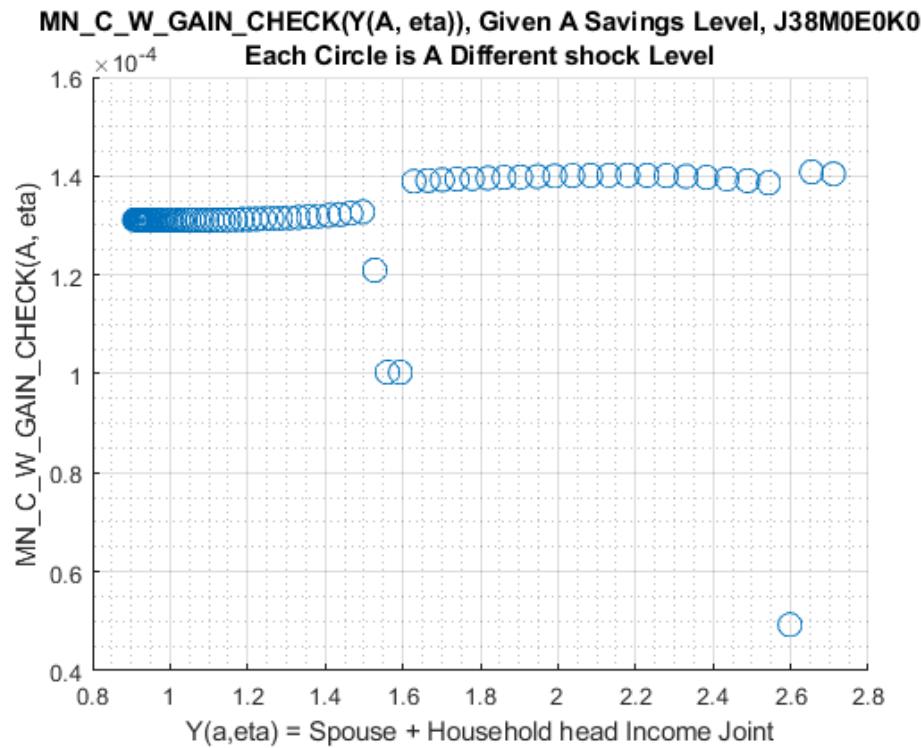
```
figure();
scatter((mt_total_inc_jemk(:)), log(mt_V_W_gain_check_jemk(:)), 100, mt_a(:));
title({'(MN\_V\_W\_GAIN\_CHECK(Y,eta)), All A (Savings) Levels, J38M0E0K0', ...
    'Color Represent different A Savings State, Circle-Group=Shock'});
xlabel('income(a,eps)');
ylabel('log of (MN\_V\_W\_GAIN\_CHECK(EM,J))');
xlim([0,7]);
grid on;
grid minor;
```



8.1.8 Marginal Consumption Gains, Color as Shock, Conditional on Age, Marital, Kids, and Education

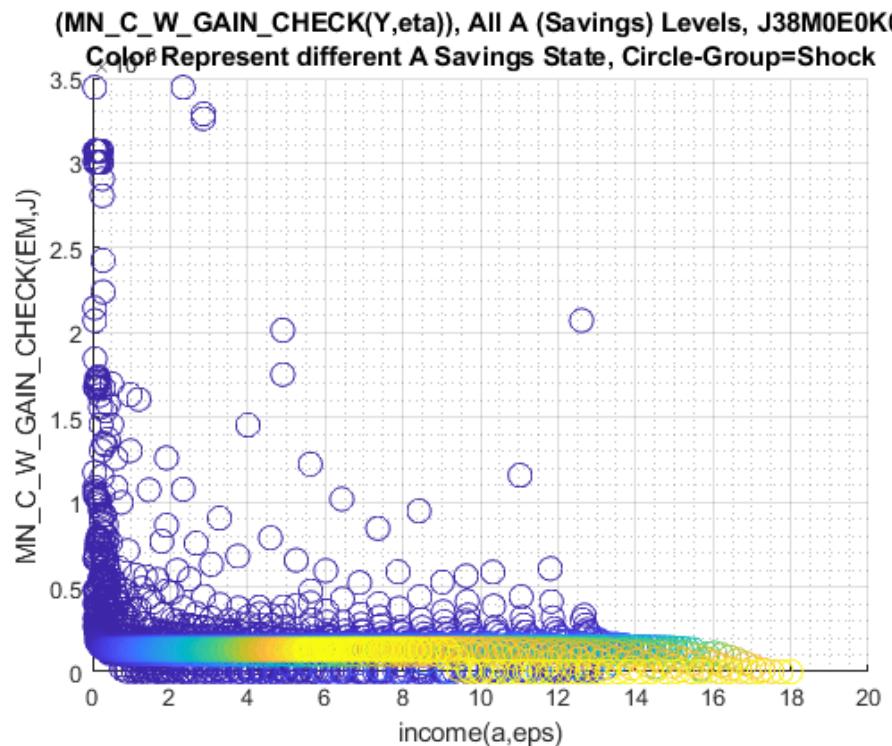
How do shocks and age impact marginal value. First plot one asset level, variation comes only from increasingly higher shocks:

```
figure();
it_a = 50;
scatter(log(mt_total_inc_jemk(:,it_a)), mt_C_W_gain_check_jemk(:,it_a), 100);
title({'MN_C_W_GAIN_CHECK(Y(A, eta)), Given A Savings Level, J38M0E0K0', ...
    'Each Circle is A Different shock Level'});
xlabel('Y(a,eta) = Spouse + Household head Income Joint');
ylabel('MN_C_W_GAIN_CHECK(A, eta)');
grid on;
grid minor;
```

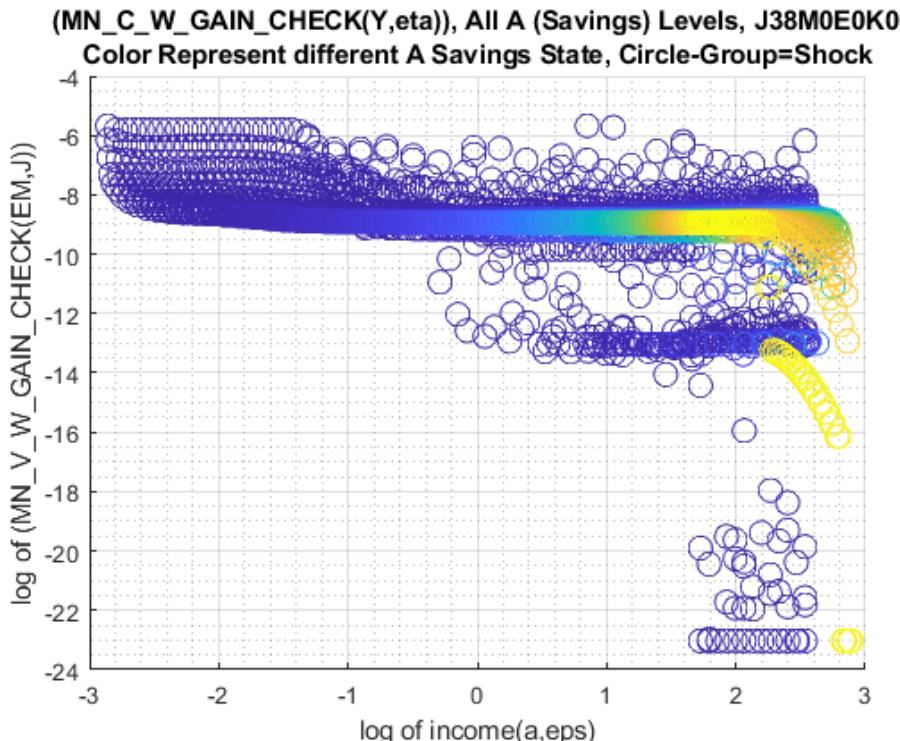


Plot all asset levels:

```
figure();
scatter((mt_total_inc_jemk(:)), (mt_C_W_gain_check_jemk(:)), 100, mt_a(:));
title({'(MN_C_W_GAIN_CHECK(Y,eta)), All A (Savings) Levels, J38M0E0K0', ...
    'Color Represent different A Savings State, Circle-Group=Shock'});
xlabel('income(a,eps)');
ylabel('MN_C_W_GAIN_CHECK(EM,J)');
grid on;
grid minor;
```



```
figure();
scatter(log(mt_total_inc_jemk(:)), log(mt_C_W_gain_check_jemk(:)), 100, mt_a(:));
title({'(MN\ C\ W\ GAIN\ CHECK(Y,eta)), All A (Savings) Levels, J38M0E0K0', ...
    'Color Represent different A Savings State, Circle-Group=Shock'});
xlabel('log of income(a,eps)');
ylabel('log of (MN\ V\ W\ GAIN\ CHECK(EM,J))');
grid on;
grid minor;
```



8.1.9 Analyze Kids and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = [...
    "k0M0", "K1M0", "K2M0", "K3M0", "K4M0", ...
    "k0M1", "K1M1", "K2M1", "K3M1", "K4M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {...
    'o', 'd', 's', 'x', '*', ...
    'o', 'd', 's', 'x', '*'};
mp_support_graph('cl_colors') = {...
    'red', 'red', 'red', 'red', 'red',...
    'blue', 'blue', 'blue', 'blue', 'blue'};
```

MEAN(VAL(KM,J)), MEAN(AP(KM,J)), MEAN(C(KM,J))

Tabulate value and policies:

```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,4,1,6,5];
```

```
% Value Function
```

```
st_title = ['MEAN(MN_V_U_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_pa_
tb_az_v = ff_summ_nd_array(st_title, mn_V_U_gain_check, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_
```

xxx MEAN(MN_V_U_GAIN_CHECK(KM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx							
group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	1	0	0.033245	0.031982	0.030513	0.027957	0.025823
2	2	0	0.045318	0.043648	0.041624	0.038035	0.035028
3	3	0	0.052753	0.051115	0.049022	0.044815	0.041294
4	4	0	0.059779	0.058053	0.055771	0.051	0.047008
5	5	0	0.065493	0.063784	0.061427	0.056219	0.051865
6	1	1	0.0098334	0.0093632	0.008915	0.008078	0.0073763
7	2	1	0.013114	0.012489	0.01189	0.010765	0.0098179
8	3	1	0.015745	0.015027	0.01433	0.012975	0.011838
9	4	1	0.018816	0.017992	0.017173	0.015564	0.014209
10	5	1	0.022802	0.021889	0.020957	0.019021	0.017394

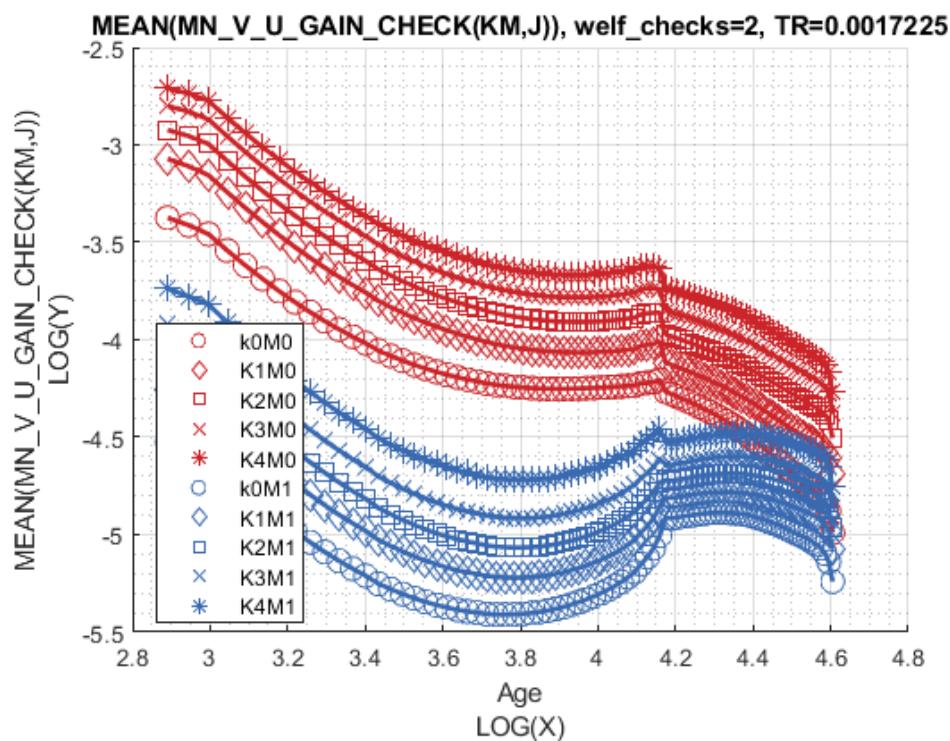
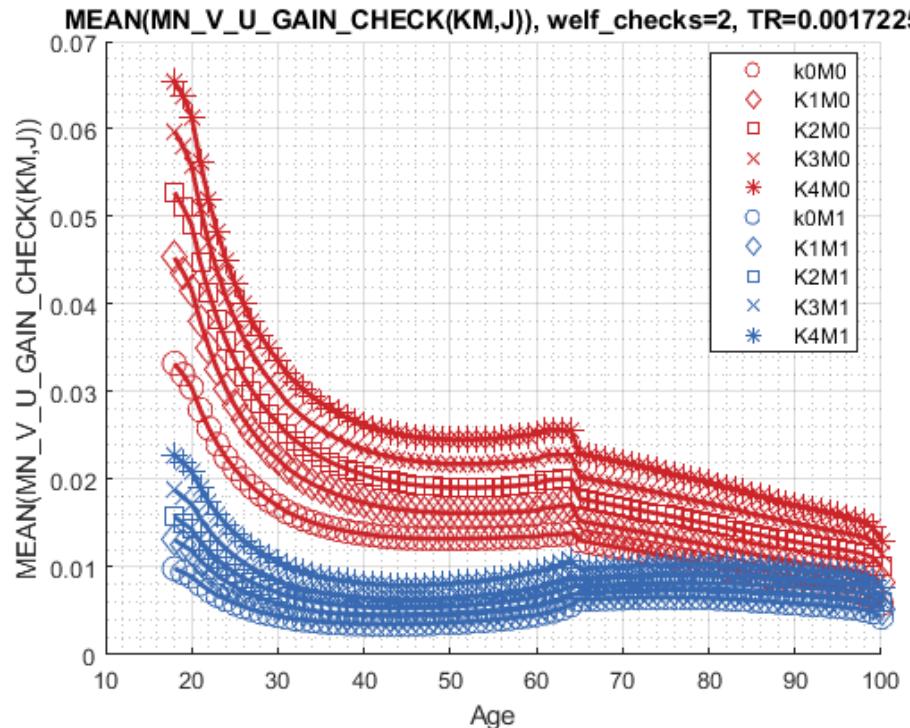
```
% Consumption Function
```

```
st_title = ['MEAN(MN_MPC_U_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
tb_az_c = ff_summ_nd_array(st_title, mn_MPC_U_gain_share_check, true, ["mean"], 3, 1, cl_mp_datasetd
```

xxx MEAN(MN_MPC_U_GAIN_CHECK(KM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx							
group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	1	0	0.054527	0.058931	0.069975	0.068541	0.066643
2	2	0	0.061679	0.066745	0.079243	0.077437	0.076495
3	3	0	0.069419	0.075436	0.090313	0.087902	0.086963
4	4	0	0.073241	0.080862	0.095495	0.092897	0.09086
5	5	0	0.078577	0.086033	0.10041	0.09783	0.095009
6	1	1	0.084627	0.088189	0.090609	0.089711	0.088925
7	2	1	0.086884	0.08995	0.093211	0.092146	0.090954
8	3	1	0.090166	0.09473	0.099076	0.097712	0.096798
9	4	1	0.092841	0.096367	0.10103	0.10024	0.099267
10	5	1	0.097558	0.10223	0.1097	0.10567	0.10418

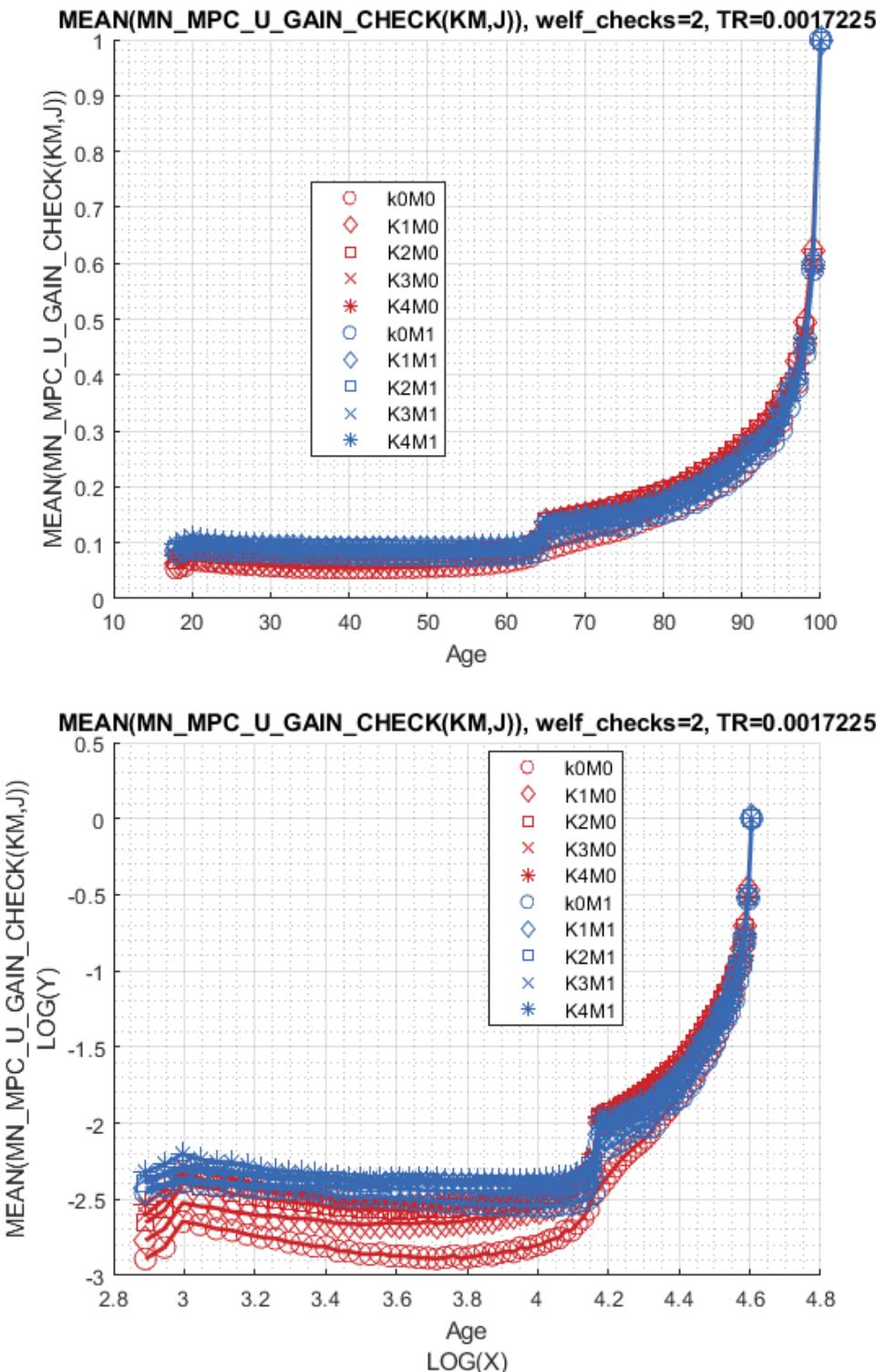
Graph Mean Values:

```
st_title = ['MEAN(MN_V_U_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN_V_U_GAIN_CHECK(KM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Graph Mean Consumption (*MPC: Share of Check Consumed*):

```
st_title = ['MEAN(MN\_MPC\_U\_GAIN\_CHECK(KM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(TR)];
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_U\_GAIN\_CHECK(KM,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



8.1.10 Analyze Education and Marriage

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["E0M0", "E1M0", "E0M1", "E1M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
```

```
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {'*', 'p', '*', 'p'};
mp_support_graph('cl_colors') = {'red', 'red', 'blue', 'blue'};
```

MEAN(VAL(EM,J)), MEAN(AP(EM,J)), MEAN(C(EM,J))

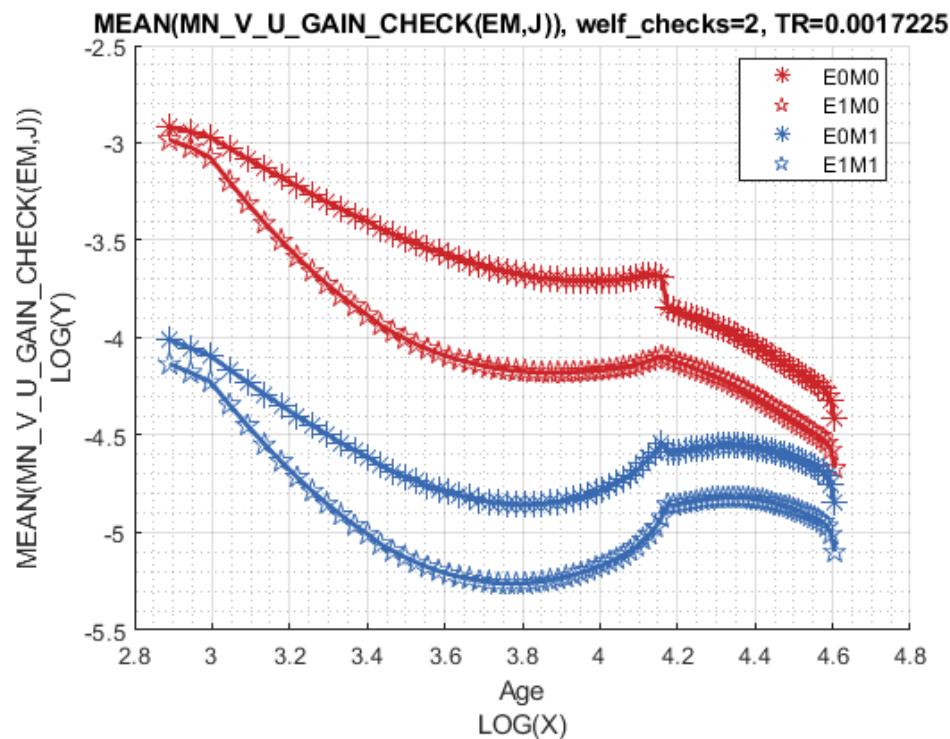
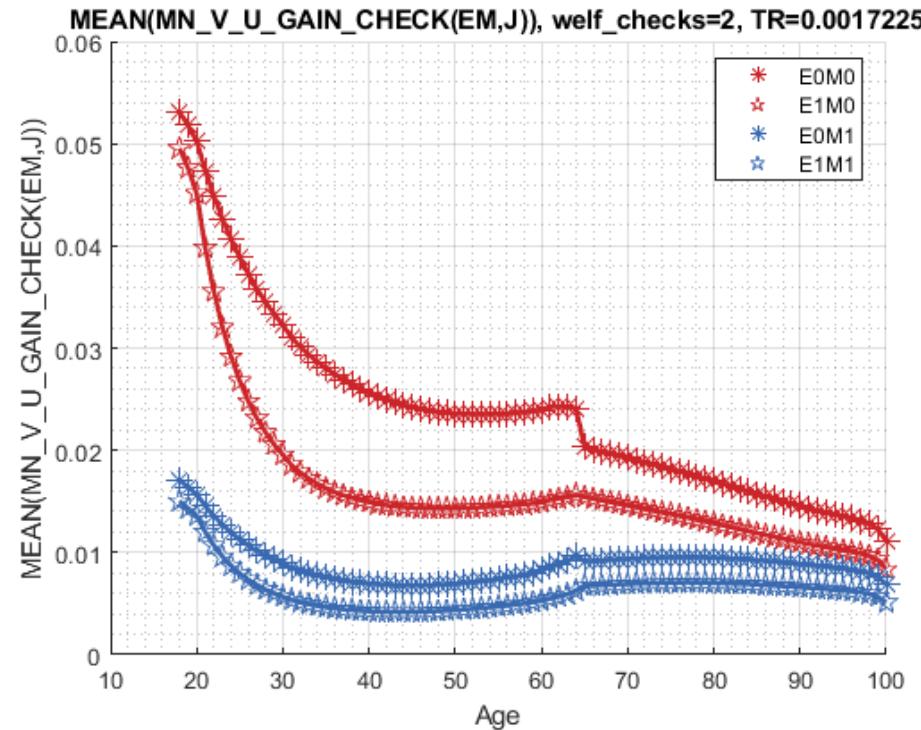
Tabulate value and policies:

```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,6,1,4,5];
% Value Function
st_title = ['MEAN(MN_V_U_GAIN_CHECK(EM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_pa
tb_az_v = ff_summ_nd_array(st_title, mn_V_U_gain_check, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_
xxx MEAN(MN_V_U_GAIN_CHECK(EM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxx
group   edu    marry   mean_age_18   mean_age_19   mean_age_20   mean_age_21   mean_age_22
-----  ---  -----  -----  -----  -----  -----  -----
1       0       0       0.053096   0.051807   0.050213   0.047392   0.044883
2       1       0       0.049539   0.047626   0.04513    0.039818   0.035524
3       0       1       0.0171     0.016386   0.01569    0.014562   0.01357
4       1       1       0.015024   0.014318   0.013616   0.011999   0.010684

% Consumption
st_title = ['MEAN(MN_MPC_U_GAIN_CHECK(EM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
tb_az_c = ff_summ_nd_array(st_title, mn_MPC_U_gain_share_check, true, ["mean"], 3, 1, cl_mp_datasetd
xxx MEAN(MN_MPC_U_GAIN_CHECK(EM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxx
group   edu    marry   mean_age_18   mean_age_19   mean_age_20   mean_age_21   mean_age_22
-----  ---  -----  -----  -----  -----  -----  -----
1       0       0       0.06081    0.064362   0.073095   0.072607   0.072694
2       1       0       0.074167   0.082841   0.10108    0.097236   0.093694
3       0       1       0.083761   0.086559   0.088972   0.089128   0.088901
4       1       1       0.097069   0.10202    0.10848    0.10507    0.10315
```

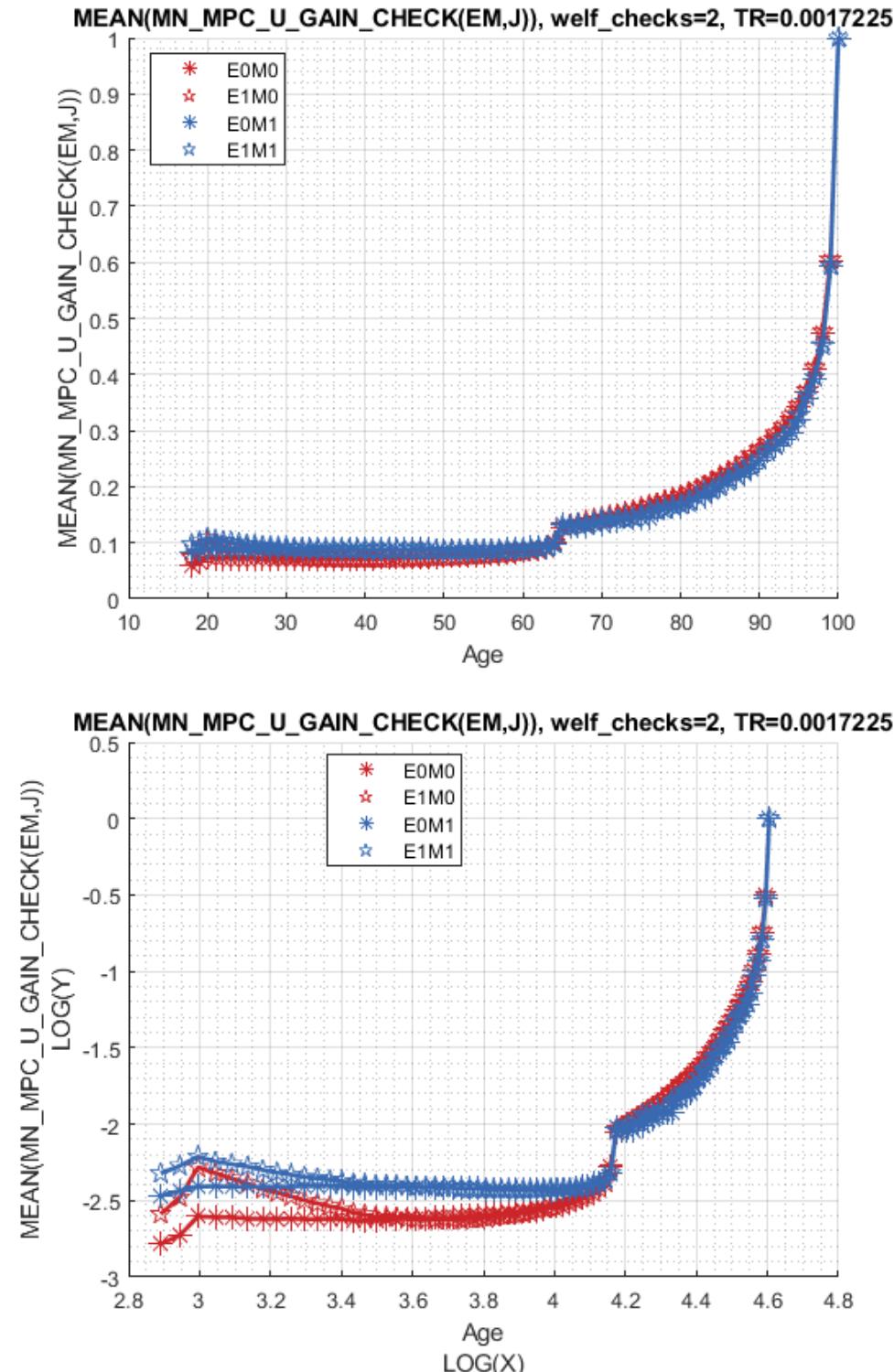
Graph Mean Values:

```
st_title = ['MEAN(MN\_\_V\_\_U\_\_GAIN\_\_CHECK(EM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_\_V\_\_U\_\_GAIN\_\_CHECK(EM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Graph Mean Consumption (*MPC: Share of Check Consumed*):

```
st_title = ['MEAN(MN\_MPC\_U\_GAIN\_CHECK(EM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(TR)];
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_U\_GAIN\_CHECK(EM,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



8.2 2019 Full States EV and EC of One Check

This is the example vignette for function: **snw_evuvw19_jaeemk** from the **PrjOptiSNW Package**. 2019 integrated over VU and VW, given optimal savings choices, unemployment shocks and various expectations.

8.2.1 Test SNW_EVUVW19_JAEEMK Defaults

Call the function with defaults.

```

clear all;
st_solu_type = 'bisec_vec';

% Solve the VFI Problem and get Value Function
mp_params = snw_mp_param('default_docdense');
mp_controls = snw_mp_control('default_test');

% set Unemployment Related Variables
mp_params('a2_covidyr') = mp_params('a2_covidyr_manna_heaven');
% mp_params('a2_covidyr') = mp_params('a2_covidyr_tax_fully_pay');

% Solve for Unemployment Values
mp_controls('bl_print_vfi') = false;
mp_controls('bl_print_vfi_verbose') = true;
mp_controls('bl_print_ds') = false;
mp_controls('bl_print_ds_verbose') = false;
mp_controls('bl_print_precompute') = false;
mp_controls('bl_print_precompute_verbose') = false;
mp_controls('bl_print_a4chk') = false;
mp_controls('bl_print_a4chk_verbose') = false;
mp_controls('bl_print_evuvw20_jaeemk') = false;
mp_controls('bl_print_evuvw20_jaeemk_verbose') = false;

% Solve the Model to get V working and unemployed
[V_ss,ap_ss,cons_ss,mp_valpol_more_ss] = snw_vfi_main_bisec_vec(mp_params, mp_controls);

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=527.

-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

      i    idx   ndim   numel    rowN    colN      sum     mean     std
      -    ---   ----   -----   ----   -----   -----   -----   -----
V_VFI    1      1      6   4.37e+07    83  5.265e+05 -1.5339e+08 -3.5101  26.11
ap_VFI   2      2      6   4.37e+07    83  5.265e+05  1.4159e+09 32.402   36.79
cons_VFI 3      3      6   4.37e+07    83  5.265e+05  2.1402e+08 4.8975  8.329

xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxx
      c1      c2      c3      c4      c5    c526496    c526497    c526498    c
      ----  -----  -----  -----  -----  -----  -----  -----  -----
r1   -346.51  -346.12  -343.63  -337.86  -328.51  21.702   21.852   22.003
r2   -334.38  -333.99  -331.51  -325.83  -316.83  21.724   21.869   22.015
r3   -322.45  -322.06  -319.6   -314.14  -305.6   21.745   21.885   22.027
r4   -310.63  -310.27  -307.99  -302.88  -294.87  21.767   21.903   22.041
r5   -299.94  -299.6   -297.46  -292.67  -285.12  21.775   21.907   22.042
r79  -9.9437 -9.9325 -9.8557 -9.6597 -9.3232  2.5394  2.5501  2.5602
r80  -8.9023 -8.8911 -8.8143 -8.6183 -8.2818  2.3039  2.3121  2.3198
r81  -7.6363 -7.6251 -7.5484 -7.3524 -7.0159  2.0068  2.0124  2.0176
r82  -5.9673 -5.9561 -5.8793 -5.6833 -5.3468  1.5958  1.5989  1.6018
r83  -3.5892 -3.578   -3.5012 -3.3052 -2.9687  0.97904 0.98004 0.98097 0

xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxx
      c1      c2      c3      c4      c5    c526496    c526497    c526498    c526499
      --      --  -----  -----  -----  -----  -----  -----  -----
r1    0       0     0.0005656  0.0075134  0.022901  114.75   120.41   126.27  132.3

```

r2	0	0	0.00051498	0.0065334	0.021549	114.86	120.53	126.41	132.5
r3	0	0	0.00051498	0.0049294	0.019875	114.97	120.65	126.56	132.
r4	0	0	0.00051498	0.0047937	0.019672	115.73	121.42	127.34	133.5
r5	0	0	0.00048517	0.0046683	0.019484	116.5	122.21	128.15	134.3
r79	0	0	0	0	0	81.091	85.68	90.335	94.37
r80	0	0	0	0	0	76.669	80.563	84.304	88.0
r81	0	0	0	0	0	68.313	71.534	74.475	77.83
r82	0	0	0	0	0	50.126	53.467	56.953	58.74
r83	0	0	0	0	0	0	0	0	0

xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498
r1	0.036717	0.037251	0.040426	0.04363	0.048012	9.6491	9.817	9.9649
r2	0.036717	0.037251	0.040477	0.04461	0.049364	9.8118	9.9685	10.101
r3	0.036717	0.037251	0.040477	0.046214	0.051039	9.9779	10.12	10.234
r4	0.038144	0.038678	0.041903	0.047776	0.052666	10.131	10.258	10.354
r5	0.039534	0.040068	0.043323	0.04929	0.054241	10.272	10.384	10.463
r79	0.2179	0.21844	0.22216	0.23228	0.25197	35.858	37.092	38.455
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.253	42.183	44.459
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.587	51.19	54.266
r82	0.2179	0.21844	0.22216	0.23228	0.25197	66.755	69.238	71.77
r83	0.2179	0.21844	0.22216	0.23228	0.25197	116.87	122.69	128.71

```

inc_VFI = mp_valpol_more_ss('inc_VFI');
spouse_inc_VFI = mp_valpol_more_ss('spouse_inc_VFI');
total_inc_VFI = inc_VFI + spouse_inc_VFI;
% Solve employment, same as 2020, except with possible change in tax
mp_params('xi') = 1;
mp_params('b') = 0;
[V_emp_2020,~,cons_emp_2020,~] = snw_vfi_main_bisec_vec(mp_params, mp_controls, V_ss);

```

Completed SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=d

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

CONTAINER NAME: mp_outcomes ND Array (Matrix etc)

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

i	idx	ndim	numel	rowN	colN	sum	mean	std	
-	---	----	-----	----	-----	-----	-----	-----	
V_VFI	1	1	6	4.37e+07	83	5.265e+05	-1.5339e+08	-3.5101	26.11
ap_VFI	2	2	6	4.37e+07	83	5.265e+05	1.4159e+09	32.402	36.79
cons_VFI	3	3	6	4.37e+07	83	5.265e+05	2.1402e+08	4.8975	8.329

xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c
r1	-346.51	-346.12	-343.63	-337.86	-328.51	21.702	21.852	22.003	
r2	-334.38	-333.99	-331.51	-325.83	-316.83	21.724	21.869	22.015	
r3	-322.45	-322.06	-319.6	-314.14	-305.6	21.745	21.885	22.027	
r4	-310.63	-310.27	-307.99	-302.88	-294.87	21.767	21.903	22.041	
r5	-299.94	-299.6	-297.46	-292.67	-285.12	21.775	21.907	22.042	
r79	-9.9437	-9.9325	-9.8557	-9.6597	-9.3232	2.5394	2.5501	2.5602	
r80	-8.9023	-8.8911	-8.8143	-8.6183	-8.2818	2.3039	2.3121	2.3198	
r81	-7.6363	-7.6251	-7.5484	-7.3524	-7.0159	2.0068	2.0124	2.0176	
r82	-5.9673	-5.9561	-5.8793	-5.6833	-5.3468	1.5958	1.5989	1.6018	

r83	-3.5892	-3.578	-3.5012	-3.3052	-2.9687	0.97904	0.98004	0.98097	0
xxx TABLE:ap_VFI xxxxxxxxxxxxxxxxxxxxxxxx									
c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499	c5264
--	--	-----	-----	-----	-----	-----	-----	-----	-----
r1	0	0	0.0005656	0.0075134	0.022901	114.75	120.41	126.27	132.3
r2	0	0	0.00051498	0.0065334	0.021549	114.86	120.53	126.41	132.5
r3	0	0	0.00051498	0.0049294	0.019875	114.97	120.65	126.56	132.
r4	0	0	0.00051498	0.0047937	0.019672	115.73	121.42	127.34	133.5
r5	0	0	0.00048517	0.0046683	0.019484	116.5	122.21	128.15	134.3
r79	0	0	0	0	0	81.091	85.68	90.335	94.37
r80	0	0	0	0	0	76.669	80.563	84.304	88.0
r81	0	0	0	0	0	68.313	71.534	74.475	77.83
r82	0	0	0	0	0	50.126	53.467	56.953	58.74
r83	0	0	0	0	0	0	0	0	0
xxx TABLE:cons_VFI xxxxxxxxxxxxxxxxxxxxxxxx									
c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499	c5264
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.036717	0.037251	0.040426	0.04363	0.048012	9.6491	9.817	9.9649	c
r2	0.036717	0.037251	0.040477	0.04461	0.049364	9.8118	9.9685	10.101	c
r3	0.036717	0.037251	0.040477	0.046214	0.051039	9.9779	10.12	10.234	c
r4	0.038144	0.038678	0.041903	0.047776	0.052666	10.131	10.258	10.354	c
r5	0.039534	0.040068	0.043323	0.04929	0.054241	10.272	10.384	10.463	c
r79	0.2179	0.21844	0.22216	0.23228	0.25197	35.858	37.092	38.455	c
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.253	42.183	44.459	c
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.587	51.19	54.266	c
r82	0.2179	0.21844	0.22216	0.23228	0.25197	66.755	69.238	71.77	c
r83	0.2179	0.21844	0.22216	0.23228	0.25197	116.87	122.69	128.71	c
% Solve unemployment, different income than under ss due to income losses									
mp_params('xi') = 0.50;									
mp_params('b') = 0.50;									
[V_unemp_2020,~,cons_unemp_2020,~] = snw_vfi_main_bisec_vec(mp_params, mp_controls, V_ss);									
Completed SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=d									

xxx									
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)									
xxx									
i	idx	ndim	numel	rowN	colN	sum	mean	std	c
-	--	---	-----	---	-----	-----	-----	-----	-----
V_VFI	1	1	6	4.37e+07	83	5.265e+05	-1.6419e+08	-3.7572	26.50
ap_VFI	2	2	6	4.37e+07	83	5.265e+05	1.3972e+09	31.974	36.73
cons_VFI	3	3	6	4.37e+07	83	5.265e+05	2.1267e+08	4.8667	8.327
xxx TABLE:V_VFI xxxxxxxxxxxxxxxxxxxxxxxx									
c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499	c526500
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----
r1	-355.34	-354.66	-350.52	-342.92	-332.4	21.638	21.791	21.943	c
r2	-343.21	-342.53	-338.39	-330.8	-320.53	21.66	21.807	21.955	c
r3	-331.28	-330.6	-326.46	-318.91	-309.12	21.682	21.824	21.967	c
r4	-319.13	-318.5	-314.65	-307.53	-298.32	21.7	21.838	21.977	c
r5	-308.13	-307.54	-303.94	-297.21	-288.52	21.705	21.839	21.975	c

r79	-9.9437	-9.9325	-9.8557	-9.6597	-9.3232	2.5384	2.5492	2.5593	
r80	-8.9023	-8.8911	-8.8143	-8.6183	-8.2818	2.3032	2.3114	2.3191	
r81	-7.6363	-7.6251	-7.5484	-7.3524	-7.0159	2.0063	2.0119	2.0172	
r82	-5.9673	-5.9561	-5.8793	-5.6833	-5.3468	1.5955	1.5987	1.6016	
r83	-3.5892	-3.578	-3.5012	-3.3052	-2.9687	0.97895	0.97995	0.98089	
xxx TABLE:ap_VFI	xxxxxxxxxxxxxxxxxxxxxx								
c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499	
--	--	--	-----	-----	-----	-----	-----	-----	
r1	0	0	0	0.0017295	0.013921	112.32	117.97	123.84	129.95
r2	0	0	0	0.0014073	0.013905	112.36	118.03	123.91	130.03
r3	0	0	0	0.00051498	0.013905	112.4	118.08	123.99	130.13
r4	0	0	0	0.00051498	0.013905	112.93	118.63	124.55	130.71
r5	0	0	0	0.00051498	0.013905	113.47	119.18	125.12	131.3
r79	0	0	0	0	0	81.091	85.68	89.816	93.86
r80	0	0	0	0	0	76.378	80.051	83.793	87.528
r81	0	0	0	0	0	68.288	71.027	73.968	77.326
r82	0	0	0	0	0	50.126	53.467	56.61	58.244
r83	0	0	0	0	0	0	0	0	0
xxx TABLE:cons_VFI	xxxxxxxxxxxxxxxxxxxxxx								
c1	c2	c3	c4	c5	c526496	c526497	c526498	c526499	
-----	-----	-----	-----	-----	-----	-----	-----	-----	
r1	0.027723	0.028258	0.031999	0.040426	0.048012	9.6491	9.817	9.9649	
r2	0.027723	0.028258	0.031999	0.040748	0.048028	9.8118	9.9685	10.101	
r3	0.027723	0.028258	0.031999	0.041641	0.048028	9.9779	10.12	10.234	
r4	0.028805	0.029339	0.033081	0.042722	0.049108	10.131	10.258	10.354	
r5	0.029859	0.030394	0.034135	0.043775	0.050161	10.272	10.384	10.463	
r79	0.2179	0.21844	0.22216	0.23228	0.25197	35.339	36.573	38.455	
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.033	42.183	44.459	
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.106	51.19	54.266	
r82	0.2179	0.21844	0.22216	0.23228	0.25197	66.254	68.736	71.611	
r83	0.2179	0.21844	0.22216	0.23228	0.25197	116.37	122.19	128.21	

```
[Phi_true] = snw_ds_main(mp_params, mp_controls, ap_ss, cons_emp_2020, mp_valpol_more_ss);

Completed SNW_DS_MAIN;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=1500.619

% Get Matrixes
cl_st_precompute_list = {'a', ...
    'inc', 'inc_unemp', 'spouse_inc', 'spouse_inc_unemp', 'ref_earn_wageind_grid',...
    'ap_idx_lower_ss', 'ap_idx_higher_ss', 'ap_idx_lower_weight_ss'};
mp_controls('bl_print_precompute_verbose') = false;
[mp_precompute_res] = snw_hh_precompute(mp_params, mp_controls, cl_st_precompute_list, ap_ss, Phi_tr

Wage quintile cutoffs=0.4645      0.71528      1.0335      1.5632
Completed SNW_HH_PRECOMPUTE;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time cost=428.
```

8.2.2 Solve for 2019 Evuvw With 0 and 2 Checks

```
% Call Function
welf_checks = 0;
[ev19_jaeemk_check0, ec19_jaeemk_check0, ev20_jaeemk_check0, ec20_jaeemk_check0] = snw_evuvw19_jaeemk...
    welf_checks, st_solu_type, mp_params, mp_controls, ...
    V_emp_2020, cons_emp_2020, V_unemp_2020, cons_unemp_2020, mp_precompute_res);
```

Completed SNW_A4CHK_WRK_BISEC_VEC;welf_checks=0;TR=0.0017225;SNW_MP_PARAM=default_docdense;SNW_MP_CO
 Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=0;TR=0.0017225;xi=0.5;b=0.5;SNW_MP_PARAM=default_doc
 Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;timeEUEC=7.92
 Completed SNW_EVUVW19_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=4834.357

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

CONTAINER NAME: mp_outcomes ND Array (Matrix etc)

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

	i	idx	ndim	numel	rowN	colN	sum	mean
	-	---	----	-----	---	-----	-----	-----
ec19_jaeemk	1	1	6	4.3173e+07	82	5.265e+05	1.8059e+08	4.183
ec20_jaeemk	2	2	6	4.37e+07	83	5.265e+05	2.1399e+08	4.8969
ev19_jaeemk	3	3	6	4.3173e+07	82	5.265e+05	-1.4054e+08	-3.2554
ev20_jaeemk	4	4	6	4.37e+07	83	5.265e+05	-1.5388e+08	-3.5212

xxx TABLE:ec19_jaeemk xxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498
	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.039253	0.039253	0.039822	0.044086	0.049231	9.6484	9.8085	9.9488
r2	0.039253	0.039253	0.039788	0.044409	0.050518	9.7796	9.9379	10.073
r3	0.040776	0.040776	0.041311	0.04504	0.05173	9.994	10.125	10.224
r4	0.042261	0.042261	0.042795	0.046467	0.053274	10.173	10.282	10.354
r5	0.043702	0.043702	0.044205	0.047871	0.054769	10.328	10.416	10.467
r78	0.2179	0.2179	0.2179	0.2179	0.21844	27.794	28.962	29.988
r79	0.2179	0.2179	0.2179	0.2179	0.2179	30.071	31.673	33.01
r80	0.2179	0.2179	0.2179	0.2179	0.2179	33.5	35.375	37.367
r81	0.2179	0.2179	0.2179	0.2179	0.2179	40.296	41.727	43.475
r82	0.2179	0.2179	0.2179	0.2179	0.2179	52.118	55.559	59.15

xxx TABLE:ec20_jaeemk xxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498
	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.035995	0.036529	0.03975	0.043373	0.048012	9.6491	9.817	9.9649
r2	0.035995	0.036529	0.039796	0.0443	0.049257	9.8118	9.9685	10.101
r3	0.035995	0.036529	0.039796	0.045847	0.050797	9.9779	10.12	10.234
r4	0.037394	0.037928	0.041195	0.04737	0.05238	10.131	10.258	10.354
r5	0.038757	0.039291	0.042585	0.048848	0.053914	10.272	10.384	10.463
r79	0.2179	0.21844	0.22216	0.23228	0.25197	35.858	37.092	38.455
r80	0.2179	0.21844	0.22216	0.23228	0.25197	40.253	42.183	44.459
r81	0.2179	0.21844	0.22216	0.23228	0.25197	48.587	51.19	54.266
r82	0.2179	0.21844	0.22216	0.23228	0.25197	66.755	69.238	71.77
r83	0.2179	0.21844	0.22216	0.23228	0.25197	116.87	122.69	128.71

xxx TABLE:ev19_jaeemk xxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498
	-----	-----	-----	-----	-----	-----	-----	-----
r1	-330.68	-330.68	-330.27	-326.02	-318.4	21.591	21.741	21.892
r2	-318.19	-318.19	-317.82	-314.17	-306.96	21.61	21.755	21.902
r3	-305.88	-305.88	-305.54	-302.95	-296.09	21.628	21.769	21.913
r4	-294.75	-294.75	-294.43	-292.07	-285.66	21.648	21.785	21.925
r5	-284.65	-284.65	-284.37	-282.21	-276.18	21.653	21.787	21.924
r78	-9.95	-9.95	-9.95	-9.95	-9.9388	2.4886	2.501	2.5125
r79	-8.9084	-8.9084	-8.9084	-8.9084	-8.9084	2.2539	2.2657	2.2765
r80	-7.6422	-7.6422	-7.6422	-7.6422	-7.6422	1.9631	1.9708	1.9776

```

r81   -5.9728   -5.9728   -5.9728   -5.9728   -5.9728   -5.9728   1.5603   1.5647   1.5684
r82   -3.5937   -3.5937   -3.5937   -3.5937   -3.5937   -3.5937   0.95581  0.95855  0.96107  0

xxx TABLE:ev20_jaeemk xxxxxxxxxxxxxxxxxxxxxxxx
      c1       c2       c3       c4       c5       c526496  c526497  c526498  c
      -----  -----  -----  -----  -----  -----  -----  -----  -----
r1    -347.22   -346.8    -344.19   -338.27   -328.82   21.701   21.851   22.001
r2    -335.09   -334.67   -332.06   -326.23   -317.13   21.722   21.867   22.013
r3    -323.16   -322.74   -320.15   -314.53   -305.89   21.743   21.884   22.026
r4    -311.32   -310.93   -308.52   -303.26   -295.15   21.765   21.902   22.04
r5    -300.59   -300.24   -297.98   -293.04   -285.4    21.773   21.906   22.041
r79   -9.9437   -9.9325   -9.8557   -9.6597   -9.3232   2.5394  2.5501  2.5602
r80   -8.9023   -8.8911   -8.8143   -8.6183   -8.2818   2.3039  2.3121  2.3198
r81   -7.6363   -7.6251   -7.5484   -7.3524   -7.0159   2.0068  2.0124  2.0176
r82   -5.9673   -5.9561   -5.8793   -5.6833   -5.3468   1.5958  1.5989  1.6018
r83   -3.5892   -3.578    -3.5012   -3.3052   -2.9687   0.97904 0.98004 0.98097 0

```

```

% Call Function
welf_checks = 2;
[ev19_jaeemk_check2, ec19_jaeemk_check2, ev20_jaeemk_check2, ec20_jaeemk_check2] = snw_evuvv19_jaeemk(welf_checks, st_solu_type, mp_params, mp_controls, ...
V_emp_2020, cons_emp_2020, V_unemp_2020, cons_unemp_2020, mp_precompute_res);

Completed SNW_A4CHK_WRK_BISEC_VEC;welf_checks=2;TR=0.0017225;SNW_MP_PARAM=default_docdense;SNW_MP_CO
Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=2;TR=0.0017225;xi=0.5;b=0.5;SNW_MP_PARAM=default_doc
Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;timeEUEC=7.86
Completed SNW_EVUVW19_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=4826.021
-----

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      i       idx      ndim      numel      rowN      colN      sum      mean
      --  -----  -----  -----  -----  -----  -----  -----
ec19_jaeemk  1        1        6  4.3173e+07   82  5.265e+05  1.8061e+08  4.1835
ec20_jaeemk  2        2        6  4.37e+07    83  5.265e+05  2.1401e+08  4.8974
ev19_jaeemk  3        3        6  4.3173e+07   82  5.265e+05  -1.4e+08  -3.2427
ev20_jaeemk  4        4        6  4.37e+07    83  5.265e+05  -1.5329e+08 -3.5078

xxx TABLE:ec19_jaeemk xxxxxxxxxxxxxxxxx
      c1       c2       c3       c4       c5       c526496  c526497  c526498
      -----  -----  -----  -----  -----  -----  -----
r1    0.041965  0.041965  0.04239  0.045342  0.049961  9.6485  9.8086  9.9489
r2    0.04233   0.04233   0.042797 0.046025  0.051275  9.7797  9.9379  10.073
r3    0.043853  0.043853  0.04432  0.046854  0.052519  9.9941  10.125   10.224
r4    0.045349  0.045349  0.045817 0.048322  0.054074  10.173  10.282   10.354
r5    0.046815  0.046815  0.047261 0.049758  0.05558   10.328  10.416   10.467
r78   0.22135  0.22135  0.22135  0.22135  0.22188  27.795  28.962  29.988
r79   0.22135  0.22135  0.22135  0.22135  0.22135  30.073  31.674  33.011
r80   0.22135  0.22135  0.22135  0.22135  0.22135  33.501  35.377  37.368
r81   0.22135  0.22135  0.22135  0.22135  0.22135  40.297  41.729  43.477
r82   0.22135  0.22135  0.22135  0.22135  0.22135  52.121  55.563  59.153

xxx TABLE:ec20_jaeemk xxxxxxxxxxxxxxxxx
      c1       c2       c3       c4       c5       c526496  c526497  c526498
      -----  -----  -----  -----  -----  -----

```

r1	0.039035	0.039495	0.040979	0.044181	0.048566	9.6492	9.8171	9.965
r2	0.039071	0.039538	0.041324	0.045164	0.049841	9.8119	9.9686	10.101
r3	0.039071	0.039538	0.041849	0.04671	0.051404	9.978	10.12	10.234
r4	0.04047	0.040937	0.04329	0.048243	0.052999	10.131	10.258	10.354
r5	0.041855	0.042325	0.04471	0.04973	0.054543	10.272	10.384	10.463
r79	0.22135	0.22188	0.22561	0.23572	0.25394	35.858	37.093	38.456
r80	0.22135	0.22188	0.22561	0.23572	0.25394	40.254	42.184	44.46
r81	0.22135	0.22188	0.22561	0.23572	0.25402	48.589	51.192	54.268
r82	0.22135	0.22188	0.22561	0.23572	0.25436	66.757	69.24	71.772
r83	0.22135	0.22188	0.22561	0.23572	0.25541	116.87	122.69	128.71

xxx TABLE:ev19_jaeemk xxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c
r1	-328.44	-328.44	-328.07	-324.3	-317.04	21.591	21.741	21.892	
r2	-315.98	-315.98	-315.64	-312.48	-305.65	21.61	21.755	21.903	
r3	-303.83	-303.83	-303.52	-301.32	-294.84	21.628	21.769	21.913	
r4	-292.84	-292.84	-292.55	-290.54	-284.48	21.648	21.785	21.925	
r5	-282.87	-282.87	-282.61	-280.77	-275.07	21.653	21.787	21.924	
r78	-9.8787	-9.8787	-9.8787	-9.8787	-9.8678	2.4886	2.501	2.5125	
r79	-8.8371	-8.8371	-8.8371	-8.8371	-8.8371	2.2539	2.2657	2.2765	
r80	-7.5709	-7.5709	-7.5709	-7.5709	-7.5709	1.9631	1.9708	1.9776	
r81	-5.9015	-5.9015	-5.9015	-5.9015	-5.9015	1.5603	1.5647	1.5684	
r82	-3.5225	-3.5225	-3.5225	-3.5225	-3.5225	0.95582	0.95855	0.96107	

xxx TABLE:ev20_jaeemk xxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c5	c526496	c526497	c526498	c
r1	-344.77	-344.39	-342.18	-336.62	-327.49	21.701	21.851	22.001	
r2	-332.64	-332.26	-330.08	-324.65	-315.86	21.722	21.868	22.014	
r3	-320.73	-320.36	-318.24	-313.02	-304.68	21.743	21.884	22.026	
r4	-309.06	-308.71	-306.74	-301.84	-294.01	21.765	21.902	22.04	
r5	-298.48	-298.16	-296.31	-291.71	-284.33	21.773	21.906	22.041	
r79	-9.8725	-9.8615	-9.7889	-9.6008	-9.2728	2.5394	2.5501	2.5602	
r80	-8.8311	-8.8201	-8.7475	-8.5594	-8.2319	2.3039	2.3121	2.3198	
r81	-7.5651	-7.5542	-7.4816	-7.2935	-6.9665	2.0068	2.0124	2.0176	
r82	-5.8961	-5.8851	-5.8126	-5.6245	-5.2979	1.5958	1.5989	1.6018	
r83	-3.518	-3.507	-3.4345	-3.2464	-2.9207	0.97904	0.98004	0.98097	

Differences between Checks in Expected Value and Expected Consumption

mn_V_U_gain_check = ev19_jaeemk_check2 - ev19_jaeemk_check0;

mn_MPC_U_gain_share_check = (ec19_jaeemk_check2 - ec19_jaeemk_check0)./(welf_checks*mp_params('TR'))

8.2.3 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 = 18:99;
agrid = mp_params('agrid');
eta_H_grid = mp_params('eta_H_grid');
eta_S_grid = mp_params('eta_S_grid');
ar_st_eta_HS_grid = string(cellstr([num2str(eta_H_grid, 'hz=%3.2f;'), num2str(eta_S_grid, 'wz=%3.2f;')]));
edu_grid = [0,1];
marry_grid = [0,1];
```

```

kids_grid = (1:1:mp_params('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});

```

8.2.4 Analyze Difference in V and C with Check

The difference between V and V with Check, marginal utility gain given the check.

```

% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support_graph('cl_st_xttitle') = {'Savings States, a'};
mp_support_graph('st_legend_loc') = 'eastoutside';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('it_legend_select') = 21; % how many shock legends to show
mp_support_graph('cl_colors') = 'jet';

MEAN(MN_V_GAIN_CHECK(A,Z))

```

Tabulate value and policies along savings and shocks:

```

% Set
ar_permute = [1,4,5,6,3,2];
% Value Function
st_title = ['MEAN(MN_V_U_GAIN_CHECK(A,Z)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_par
tb_az_v = ff_summ_nd_array(st_title, mn_V_U_gain_check, true, ["mean"], 4, 1, cl_mp_datasetdesc, ar_]

xxx MEAN(MN_V_U_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx
group      savings      mean_eta_1      mean_eta_2      mean_eta_3      mean_eta_4      mean_eta_5      mean_
-----      -----      -----      -----      -----      -----      -----      -----
1           0          0.51002       0.47311       0.43235       0.39151       0.35308

```

```

st_title = ['MEAN(MN\_V\_U\_GAIN\_CHECK(A,Z)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(m

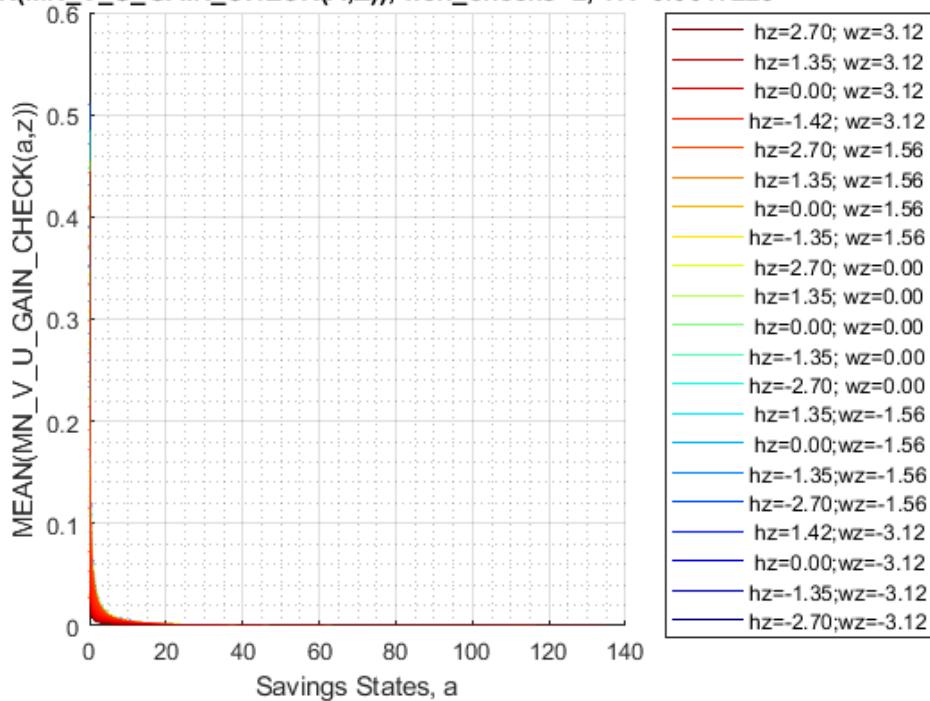
```

```

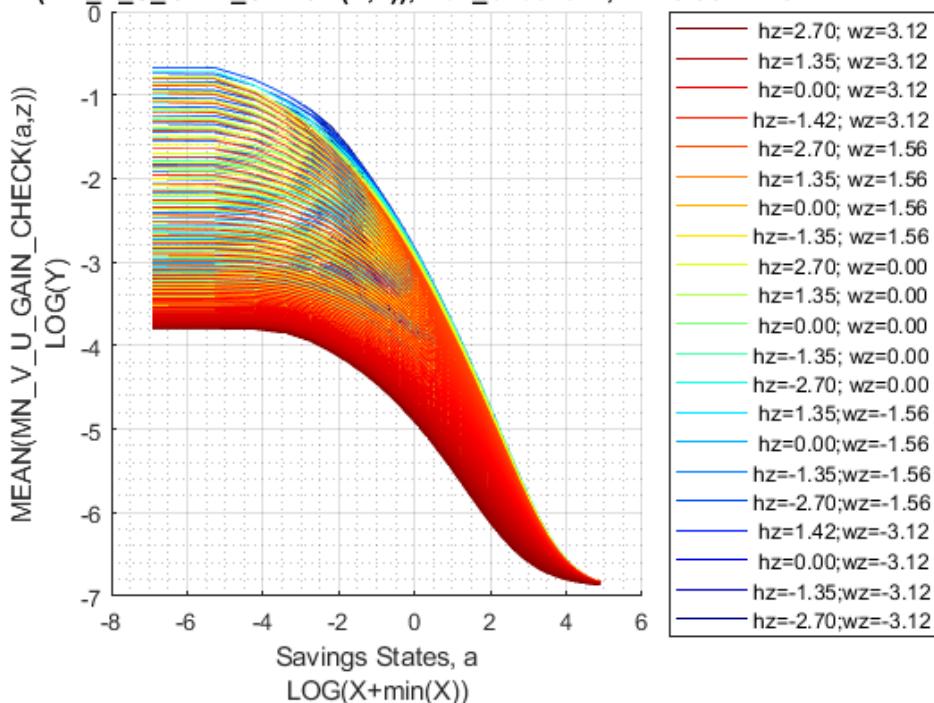
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_V\_U\_GAIN\_CHECK(a,z))'};
ff_graph_grid((tb_az_v{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);

```

N(MN_V_U_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225



AN(MN_V_U_GAIN_CHECK(A,Z)), welf_checks=2, TR=0.0017225

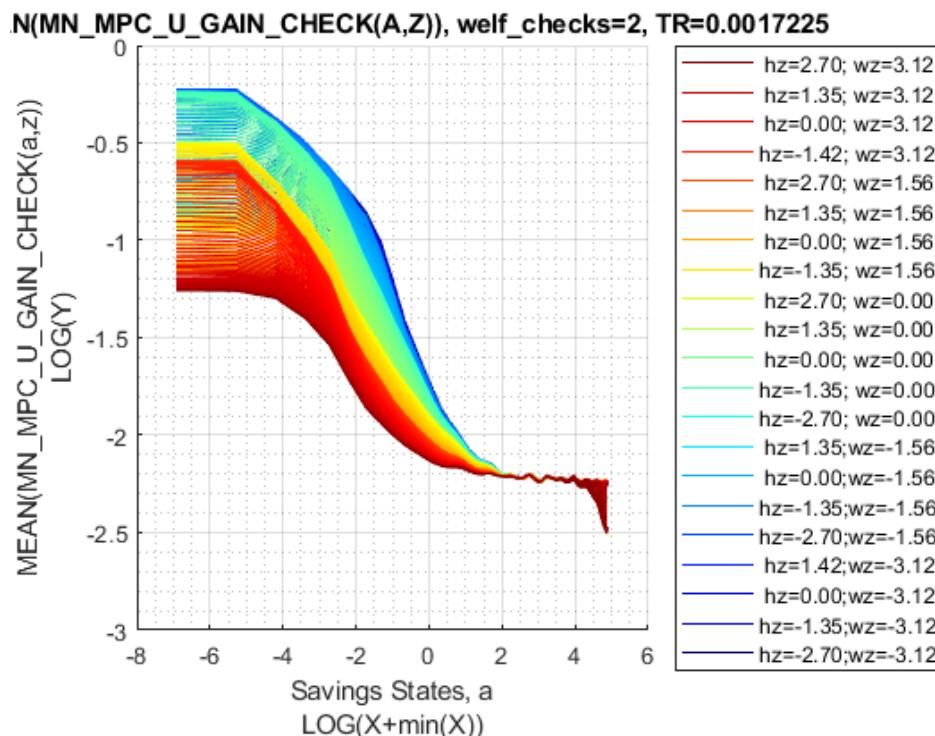
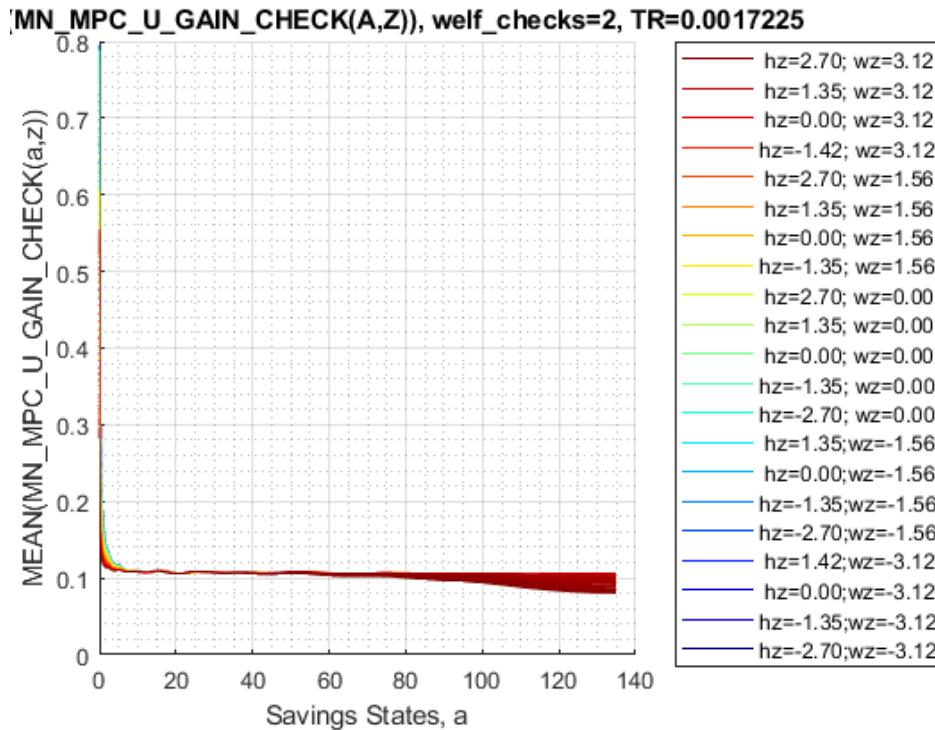


Graph Mean Consumption (*MPC: Share of Check Consumed*):

```

st_title = ['MEAN(MN\_MPC\_U\_GAIN\_CHECK(A,Z))', welf\_checks=' num2str(welf_checks) ', TR=' num2str(
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_U\_GAIN\_CHECK(a,z))'};
ff_graph_grid((tb_az_c{1:end, 3:end})', ar_st_eta_HS_grid, agrid, mp_support_graph);

```



8.2.5 Analyze Marginal Value and MPC over Y(a,eta), Conditional On Kids, Marry, Age, Education

Income is generated by savings and shocks, what are the income levels generated by all the shock and savings points conditional on kids, marital status, age and educational levels. Plot on the Y axis MPC, and plot on the X axis income levels, use colors to first distinguish between different a levels, then use colors to distinguish between different eta levels.

Set Up date, Select Age 37vn

, unmarried, no kids, lower education:

```
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
% 38 year old, unmarried, no kids, lower educated
% Only Household Head Shock Matters so select up to 'n_eta_H_grid'
mn_total_inc_jemk = total_inc_VFI(19,:,1:mp_params('n_eta_H_grid'),1,1,1);
mn_V_W_gain_check_use = ev19_jaeemk_check2 - ev19_jaeemk_check0;
mn_C_W_gain_check_use = ec19_jaeemk_check2 - ec19_jaeemk_check0;
```

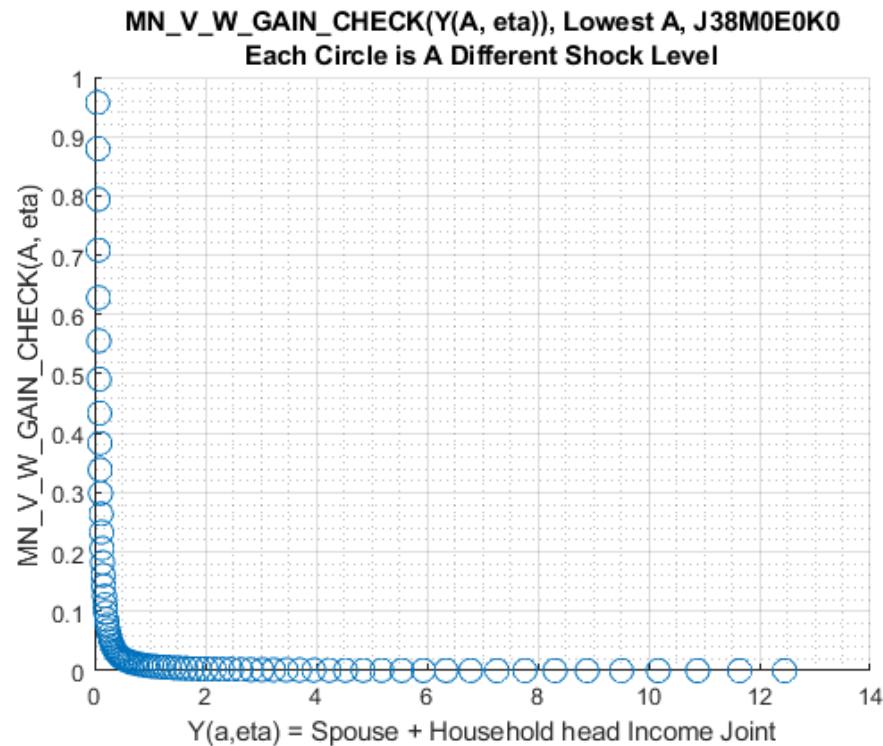
Select Age, Education, Marital, Kids Count:s

```
% Selections
it_age = 21; % +18
it_marital = 1; % 1 = unmarried
it_kids = 1; % 1 = kids is zero
it_educ = 1; % 1 = lower education
% Select: NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
mn_C_W_gain_check_jemk = mn_C_W_gain_check_use(it_age, :, 1:mp_params('n_eta_H_grid'), it_educ, it_m
mn_V_W_gain_check_jemk = mn_V_W_gain_check_use(it_age, :, 1:mp_params('n_eta_H_grid'), it_educ, it_m
% Reshape, so shock is the first dim, a is the second
mt_total_inc_jemk = permute(mn_total_inc_jemk,[3,2,1]);
mt_C_W_gain_check_jemk = permute(mn_C_W_gain_check_jemk,[3,2,1]);
mt_C_W_gain_check_jemk(mt_C_W_gain_check_jemk<=1e-10) = 1e-10;
mt_V_W_gain_check_jemk = permute(mn_V_W_gain_check_jemk,[3,2,1]);
mt_V_W_gain_check_jemk(mt_V_W_gain_check_jemk<=1e-10) = 1e-10;
% Generate meshed a and shock grid
[mt_eta_H, mt_a] = ndgrid(eta_H_grid(1:mp_params('n_eta_H_grid'))), agrid);
```

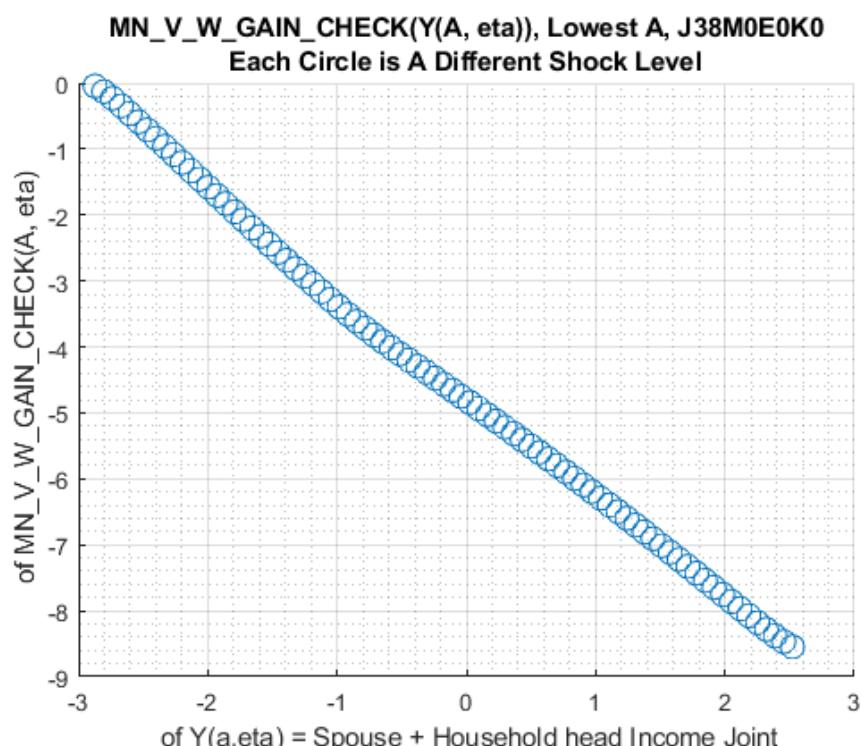
8.2.6 Marginal Value Gains, Color as Shock, Conditional on Age, Marital, Kids, and Education

How do shocks and a impact marginal value. First plot one asset level, variation comes only from increasingly higher shocks:

```
figure();
it_a = 1;
scatter((mt_total_inc_jemk(:,it_a)), (mt_V_W_gain_check_jemk(:,it_a)), 100);
title({'MN\_V\_W\_GAIN\_CHECK(Y(A, eta)), Lowest A, J38M0EOK0', ...
'Each Circle is A Different Shock Level'});
xlabel('Y(a,eta) = Spouse + Household head Income Joint');
ylabel('MN\_V\_W\_GAIN\_CHECK(A, eta)');
grid on;
grid minor;
```

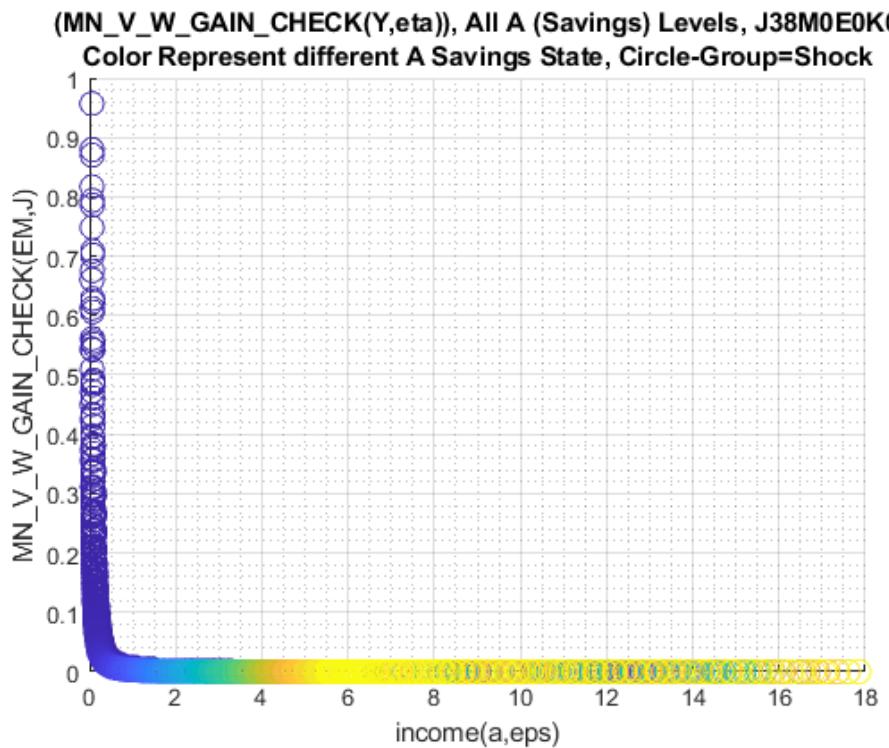


```
figure();
it_shock = 1;
scatter(log(mt_total_inc_jemk(:,it_a)), log(mt_V_W_gain_check_jemk(:,it_a)), 100);
title({'MN\_V\_W\_GAIN\_CHECK(Y(A, eta)), Lowest A, J38M0E0K0', ...
  'Each Circle is A Different Shock Level'});
xlabel(' of Y(a,eta) = Spouse + Household head Income Joint');
ylabel(' of MN\_V\_W\_GAIN\_CHECK(A, eta)');
grid on;
grid minor;
```

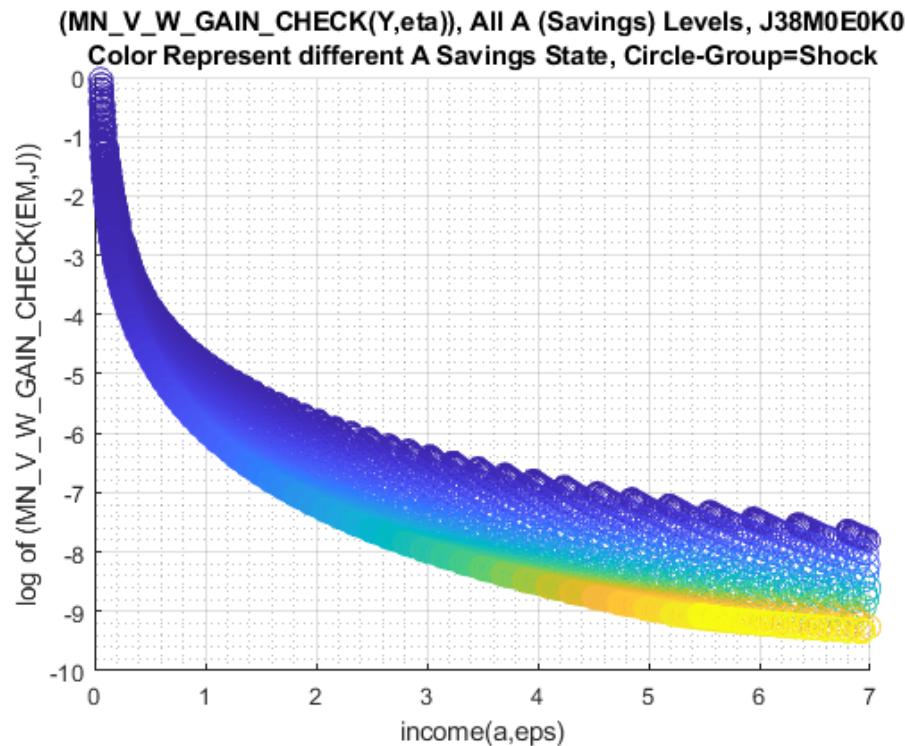


Plot all asset levels:

```
figure();
scatter((mt_total_inc_jemk(:)), (mt_V_W_gain_check_jemk(:)), 100, mt_a(:));
title({'(MN\_V\_W\_GAIN\_CHECK(Y,eta)), All A (Savings) Levels, J38M0E0K0', ...
    'Color Represent different A Savings State, Circle-Group=Shock'});
xlabel('income(a,eps)');
ylabel('MN\_V\_W\_GAIN\_CHECK(EM,J)');
grid on;
grid minor;
```



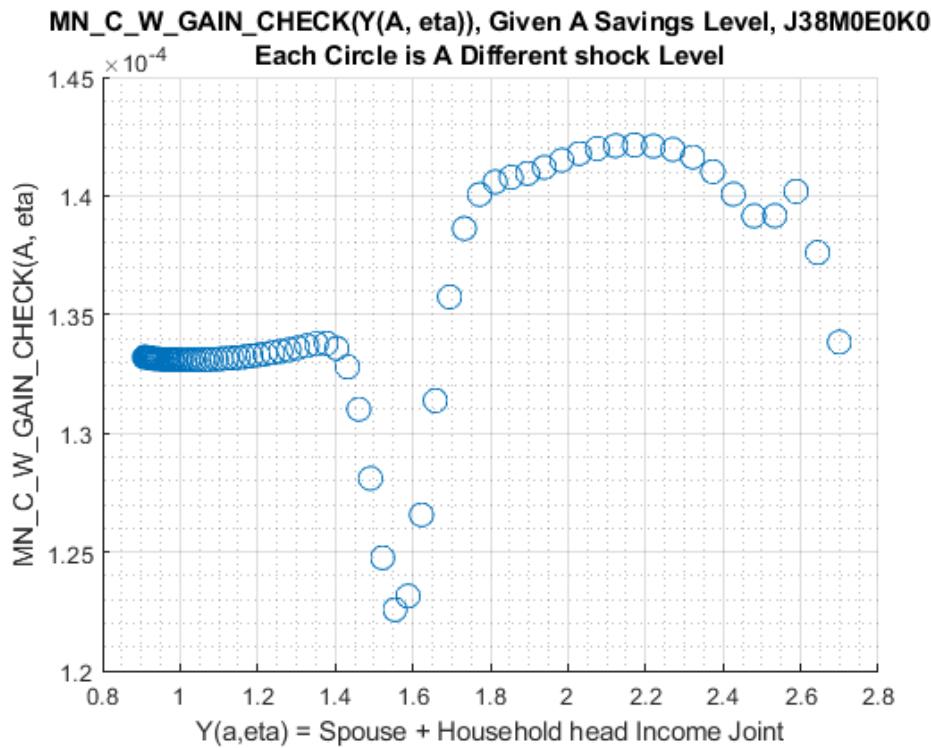
```
figure();
scatter((mt_total_inc_jemk(:)), log(mt_V_W_gain_check_jemk(:)), 100, mt_a(:));
title({'(MN\_V\_W\_GAIN\_CHECK(Y,eta)), All A (Savings) Levels, J38M0E0K0', ...
    'Color Represent different A Savings State, Circle-Group=Shock'});
xlabel('income(a,eps)');
ylabel('log of (MN\_V\_W\_GAIN\_CHECK(EM,J))');
xlim([0,7]);
grid on;
grid minor;
```



8.2.7 Marginal Consumption Gains, Color as Shock, Conditional on Age, Marital, Kids, and Education

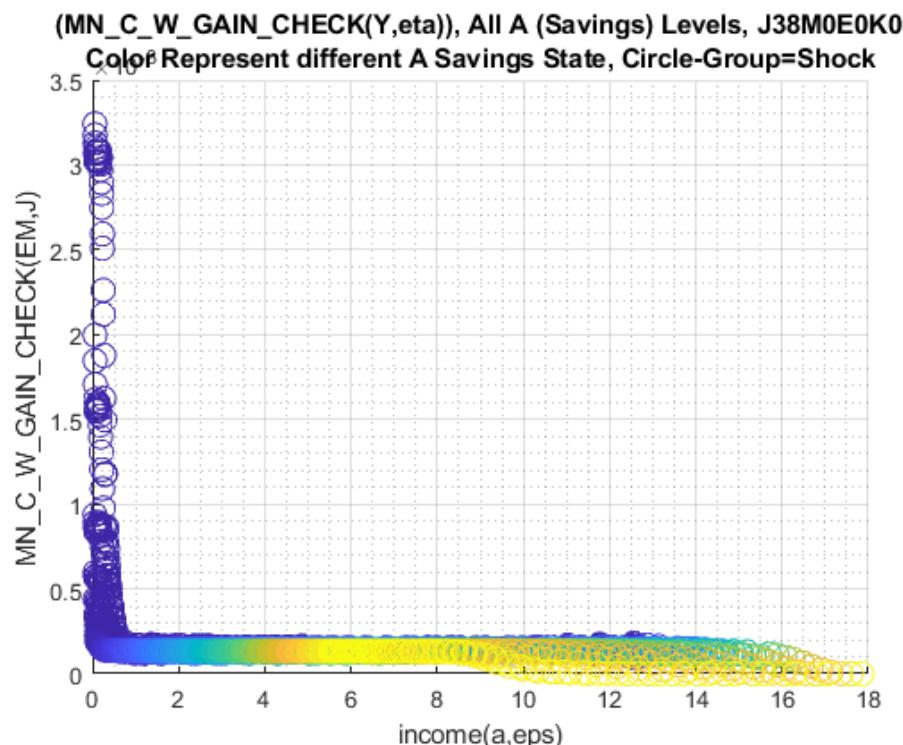
How do shocks and age impact marginal value. First plot one asset level, variation comes only from increasingly higher shocks:

```
figure();
it_a = 50;
scatter(log(mt_total_inc_jemk(:,it_a)), mt_C_W_gain_check_jemk(:,it_a), 100);
title({'MN_C_W_GAIN_CHECK(Y(A, eta)), Given A Savings Level, J38M0E0K0', ...
    'Each Circle is A Different shock Level'});
xlabel('Y(a,eta) = Spouse + Household head Income Joint');
ylabel('MN_C_W_GAIN_CHECK(A, eta)');
grid on;
grid minor;
```

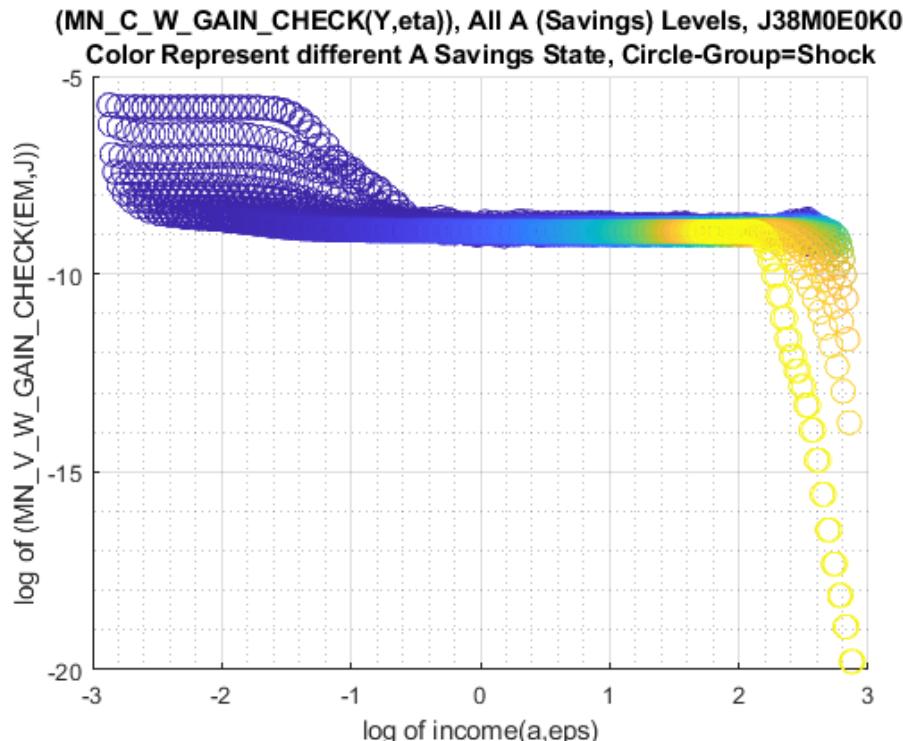


Plot all asset levels:

```
figure();
scatter((mt_total_inc_jemk(:)), (mt_C_W_gain_check_jemk(:)), 100, mt_a(:));
title({'(MN_C_W_GAIN_CHECK(Y, \eta)), All A (Savings) Levels, J38M0E0K0', ...
    'Color Represent different A Savings State, Circle-Group=Shock'});
xlabel('income(a, eps)');
ylabel('MN_C_W_GAIN_CHECK(EM, J)');
grid on;
grid minor;
```



```
figure();
scatter(log(mt_total_inc_jemk(:)), log(mt_C_W_gain_check_jemk(:)), 100, mt_a(:));
title({'(MN\ C\ W\ GAIN\ CHECK(Y,eta)), All A (Savings) Levels, J38M0E0K0', ...
    'Color Represent different A Savings State, Circle-Group=Shock'});
xlabel('log of income(a,eps)');
ylabel('log of (MN\ V\ W\ GAIN\ CHECK(EM,J))');
grid on;
grid minor;
```



8.2.8 Analyze Kids and Marriage and Age

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = [...
    "k0M0", "K1M0", "K2M0", "K3M0", "K4M0", ...
    "k0M1", "K1M1", "K2M1", "K3M1", "K4M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {...
    'o', 'd', 's', 'x', '*', ...
    'o', 'd', 's', 'x', '*'};
mp_support_graph('cl_colors') = {...
    'red', 'red', 'red', 'red', 'red',...
    'blue', 'blue', 'blue', 'blue', 'blue'};
```

MEAN(VAL(KM,J)), MEAN(AP(KM,J)), MEAN(C(KM,J))

Tabulate value and policies:

```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,4,1,6,5];
```

```
% Value Function
```

```
st_title = ['MEAN(MN_V_U_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_pa_
tb_az_v = ff_summ_nd_array(st_title, mn_V_U_gain_check, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_
```

xxx MEAN(MN_V_U_GAIN_CHECK(KM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx							
group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	1	0	0.031641	0.030484	0.02834	0.026038	0.024111
2	2	0	0.043088	0.041562	0.038594	0.035346	0.032616
3	3	0	0.050052	0.048552	0.04484	0.041129	0.038009
4	4	0	0.056653	0.055085	0.050837	0.046658	0.043144
5	5	0	0.061929	0.06035	0.055674	0.051173	0.04739
6	1	1	0.0059451	0.0055031	0.0050109	0.0045637	0.0041817
7	2	1	0.0083276	0.0077158	0.0070125	0.0063646	0.0058204
8	3	1	0.0099952	0.0092796	0.0084495	0.0076771	0.0070251
9	4	1	0.012363	0.0115	0.010491	0.0095402	0.0087368
10	5	1	0.015311	0.014353	0.013136	0.011989	0.011016

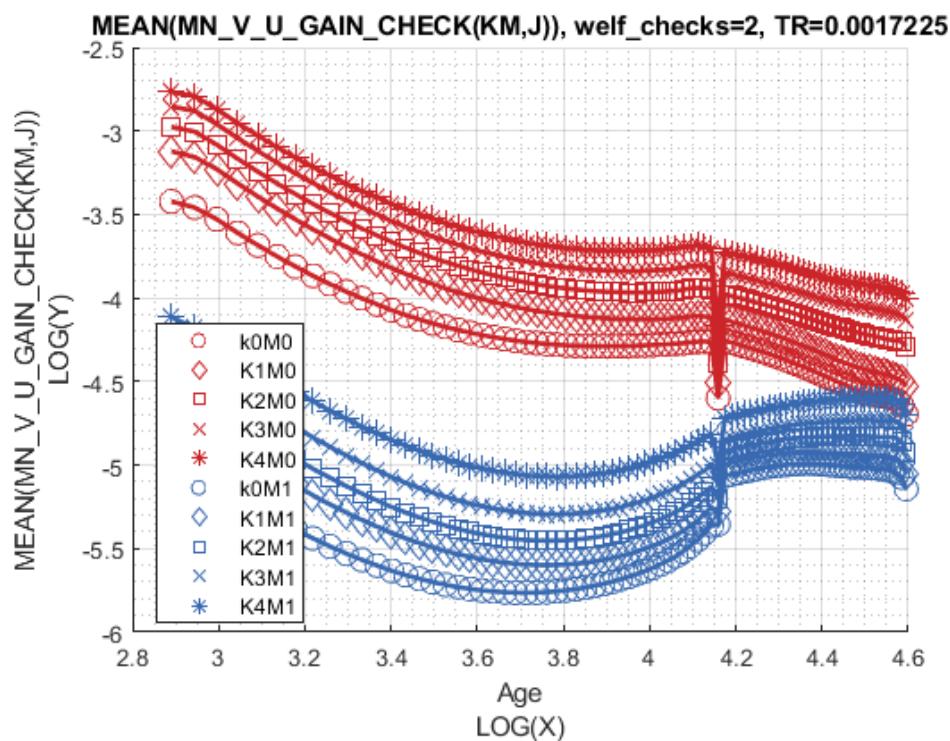
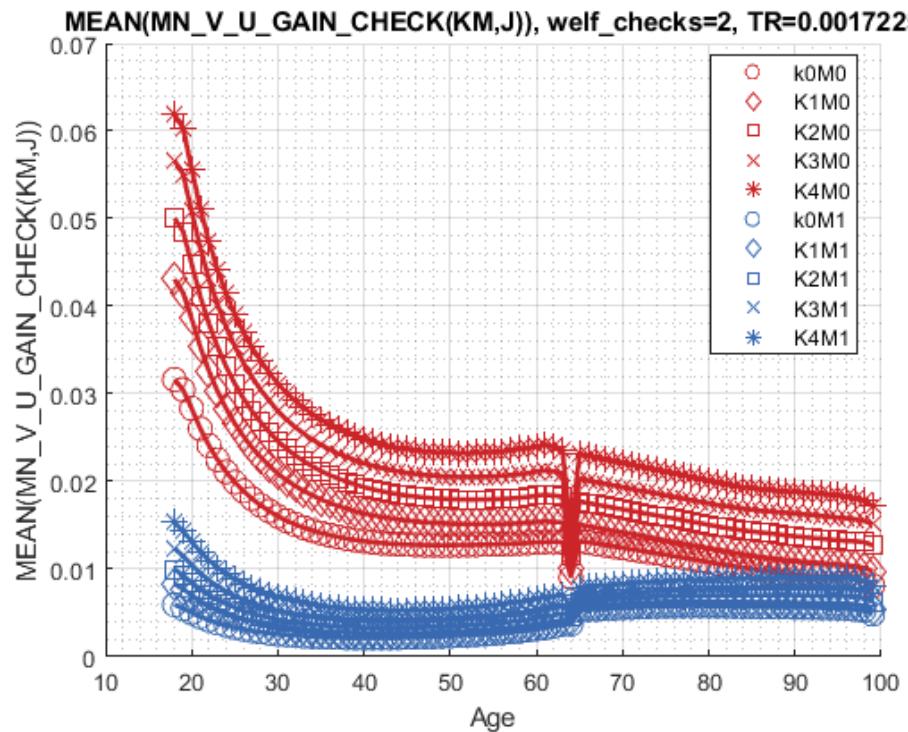
```
% Consumption Function
```

```
st_title = ['MEAN(MN_MPC_U_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
tb_az_c = ff_summ_nd_array(st_title, mn_MPC_U_gain_share_check, true, ["mean"], 3, 1, cl_mp_datasetd
```

xxx MEAN(MN_MPC_U_GAIN_CHECK(KM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx							
group	kids	marry	mean_age_18	mean_age_19	mean_age_20	mean_age_21	mean_age_22
1	1	0	0.056223	0.069417	0.075469	0.073238	0.071407
2	2	0	0.065756	0.079137	0.086795	0.084309	0.082576
3	3	0	0.074976	0.0915	0.097876	0.095212	0.092901
4	4	0	0.080849	0.097766	0.10385	0.10096	0.098266
5	5	0	0.086722	0.10427	0.1095	0.10587	0.10288
6	1	1	0.076254	0.076512	0.076532	0.075066	0.074001
7	2	1	0.078384	0.08099	0.082	0.080237	0.078625
8	3	1	0.081685	0.086418	0.087477	0.086713	0.086059
9	4	1	0.084587	0.091629	0.092233	0.091036	0.089321
10	5	1	0.094144	0.10366	0.10288	0.10024	0.099154

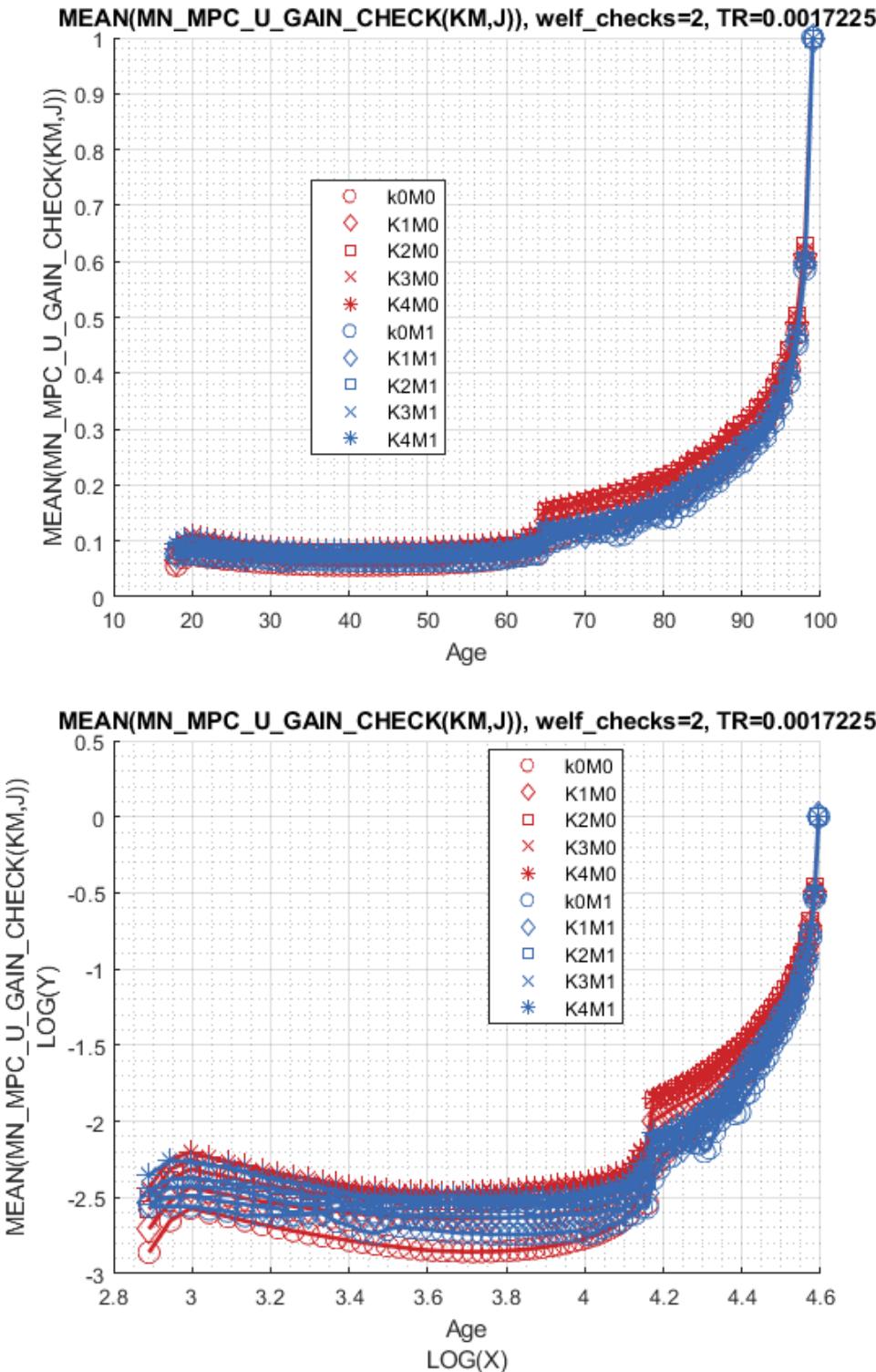
Graph Mean Values:

```
st_title = ['MEAN(MN_V_U_GAIN_CHECK(KM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN_V_U_GAIN_CHECK(KM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Graph Mean Consumption (*MPC: Share of Check Consumed*):

```
st_title = ['MEAN(MN\_MPC\_U\_GAIN\_CHECK(KM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(TR)];
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_U\_GAIN\_CHECK(KM,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



8.2.9 Analyze Education and Marriage

Aggregating over education, savings, and shocks, what are the differential effects of Marriage and Age.

```
% Generate some Data
mp_support_graph = containers.Map('KeyType', 'char', 'ValueType', 'any');
ar_row_grid = ["EOM0", "E1M0", "EOM1", "E1M1"];
mp_support_graph('cl_st_xtitle') = {'Age'};
mp_support_graph('st_legend_loc') = 'best';
mp_support_graph('bl_graph_logy') = true; % do not log
```

```
mp_support_graph('st_rounding') = '6.2f'; % format shock legend
mp_support_graph('cl_scatter_shapes') = {'*', 'p', '*', 'p'};
mp_support_graph('cl_colors') = {'red', 'red', 'blue', 'blue'};
```

MEAN(VAL(EM,J)), MEAN(AP(EM,J)), MEAN(C(EM,J))

Tabulate value and policies:

```
% Set
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
ar_permute = [2,3,6,1,4,5];
% Value Function
st_title = ['MEAN(MN_V_U_GAIN_CHECK(EM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_pa
tb_az_v = ff_summ_nd_array(st_title, mn_V_U_gain_check, true, ["mean"], 3, 1, cl_mp_datasetdesc, ar_]

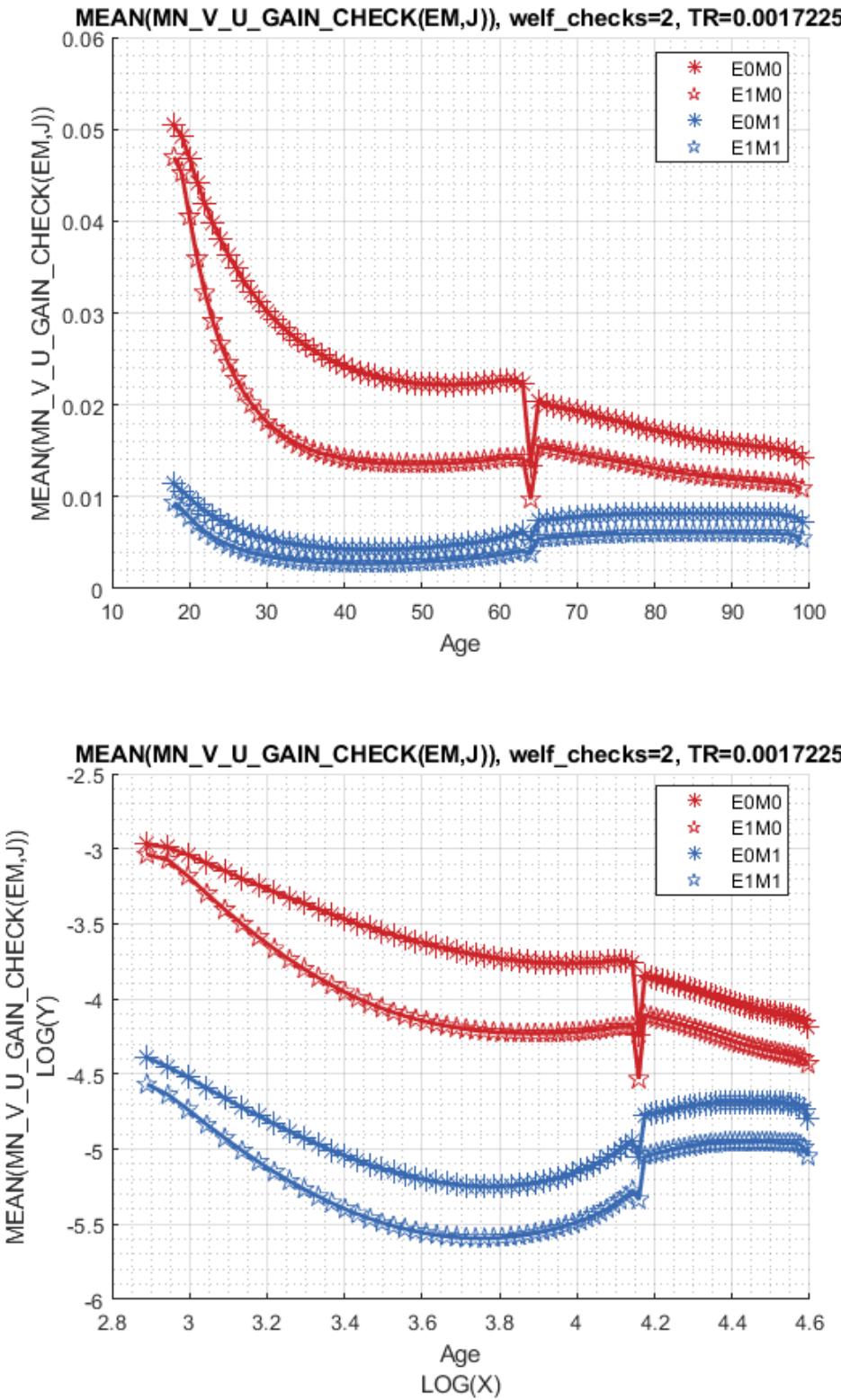
xxx MEAN(MN_V_U_GAIN_CHECK(EM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx
group   edu   marry   mean_age_18   mean_age_19   mean_age_20   mean_age_21   mean_age_22
-----  ---  -----  -----  -----  -----  -----  -----
1       0       0       0.050402   0.049194   0.046825   0.04423    0.041921
2       1       0       0.046943   0.045218   0.040488   0.035907   0.032187
3       0       1       0.011395   0.010664   0.0098666  0.0091158  0.0084651
4       1       1       0.009382   0.0086772  0.0077734  0.0069379  0.0062467

% Consumption
st_title = ['MEAN(MN_MPC_U_GAIN_CHECK(EM,J)), welf_checks=' num2str(welf_checks) ', TR=' num2str(mp_
tb_az_c = ff_summ_nd_array(st_title, mn_MPC_U_gain_share_check, true, ["mean"], 3, 1, cl_mp_datasetd

xxx MEAN(MN_MPC_U_GAIN_CHECK(EM,J)), welf_checks=2, TR=0.0017225 xxxxxxxxxxxxxxxxxxxxxxxxx
group   edu   marry   mean_age_18   mean_age_19   mean_age_20   mean_age_21   mean_age_22
-----  ---  -----  -----  -----  -----  -----  -----
1       0       0       0.063628   0.073179   0.078161   0.077574   0.077065
2       1       0       0.082182   0.10366    0.11123    0.10627    0.10215
3       0       1       0.075603   0.077333   0.078368   0.078269   0.078227
4       1       1       0.090419   0.098351   0.098081   0.09505    0.092637
```

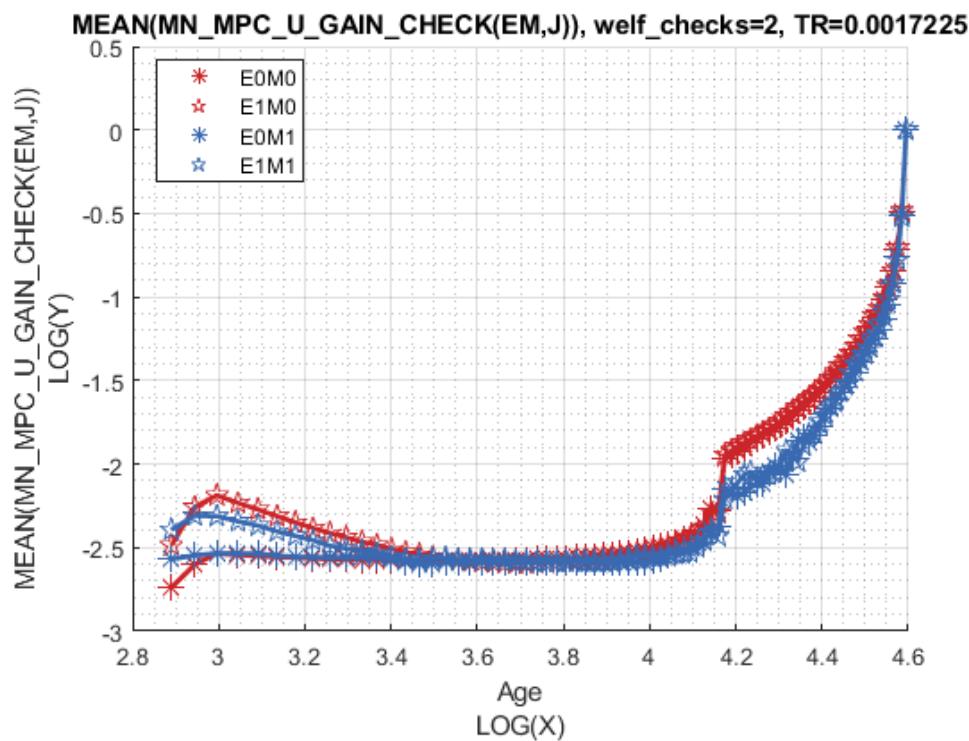
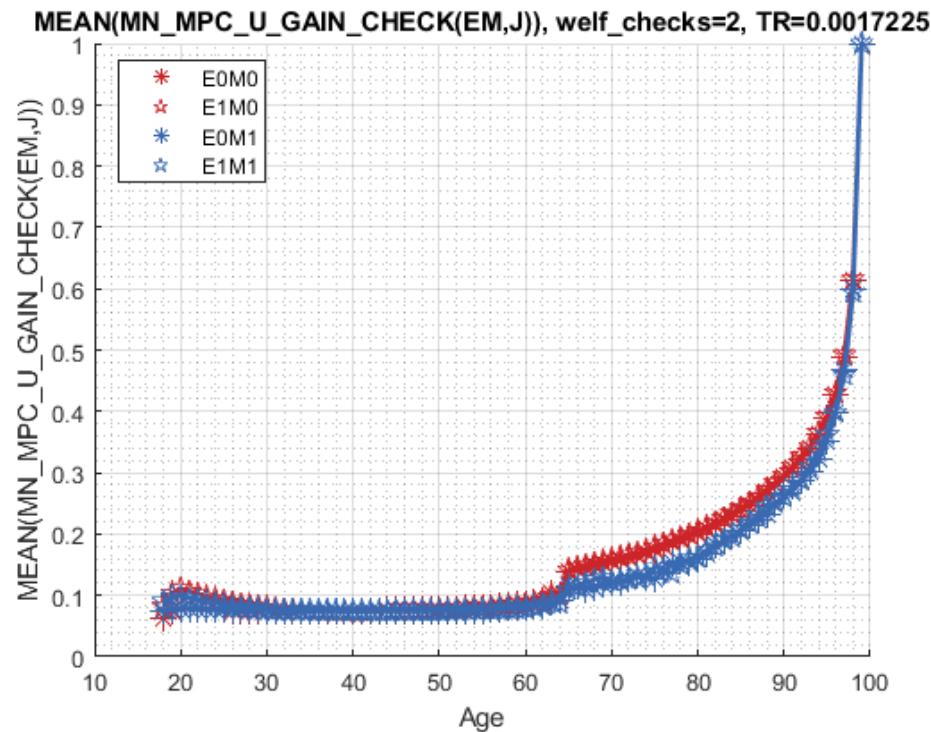
Graph Mean Values:

```
st_title = ['MEAN(MN\_\_V\_\_U\_\_GAIN\_\_CHECK(EM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_\_V\_\_U\_\_GAIN\_\_CHECK(EM,J))'};
ff_graph_grid((tb_az_v{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Graph Mean Consumption (*MPC: Share of Check Consumed*):

```
st_title = ['MEAN(MN\_MPC\_U\_GAIN\_CHECK(EM,J)), welf\_checks=' num2str(welf_checks) ', TR=' num2str(TR)];
mp_support_graph('cl_st_graph_title') = {st_title};
mp_support_graph('cl_st_ytitle') = {'MEAN(MN\_MPC\_U\_GAIN\_CHECK(EM,J))'};
ff_graph_grid((tb_az_c{1:end, 4:end}), ar_row_grid, age_grid, mp_support_graph);
```



Chapter 9

2019 Expectations Given Income, Age, Kids and Marital Status

9.1 2019 Age, Income, Kids, Marry EV and EC of One Check

This is the example vignette for function: [snw_evuvw19_jmky](#) from the [PrjOptiSNW Package](#). 2019 integrated over VU and VW

9.1.1 Test SNW_EVUVW19_JMKY Defaults Dense

Set Parameters

Call the function with defaults.

```
clear all;
st_solu_type = 'bisec_vec';

% Solve the VFI Problem and get Value Function
% mp_params = snw_mp_param('default_tiny');
% mp_params = snw_mp_param('default_dense');
mp_params = snw_mp_param('default_docdense');
mp_controls = snw_mp_control('default_test');

% set Unemployment Related Variables
xi=0.5; % Proportional reduction in income due to unemployment (xi=0 refers to 0 labor income; xi=1
b=0; % Unemployment insurance replacement rate (b=0 refers to no UI benefits; b=1 refers to 100 perc
TR=100/58056; % Value of a welfare check (can receive multiple checks). TO DO: Update with alternati

mp_params('xi') = xi;
mp_params('b') = b;
mp_params('TR') = TR;

% Check Numbers
% n_incgrid=201; % Number of income groups
% n_incgrid_aux=round(0.75*n_incgrid);
% inc_grid1=linspace(0,4,n_incgrid_aux)'; % 4 refers to 4*58056=232224 dollars in 2012USD
% inc_grid=[inc_grid1;linspace(4+((7-4)/(n_incgrid-n_incgrid_aux)),7,n_incgrid-n_incgrid_aux)']; % 7
n_incgrid=201; % Number of income groups
inc_grid=linspace(0,7,n_incgrid)';
mp_params('n_incgrid') = n_incgrid;
mp_params('inc_grid') = inc_grid;

% Solve for Unemployment Values
```

```

mp_controls('bl_print_vfi') = false;
mp_controls('bl_print_vfi_verbose') = false;
mp_controls('bl_print_ds') = false;
mp_controls('bl_print_ds_verbose') = false;
mp_controls('bl_print_recompute') = false;
mp_controls('bl_print_recompute_verbose') = false;
mp_controls('bl_print_a4chk') = false;
mp_controls('bl_print_a4chk_verbose') = false;
mp_controls('bl_print_evuvw20_jaeemk') = false;
mp_controls('bl_print_evuvw20_jaeemk_verbose') = false;
mp_controls('bl_print_evuvw19_jaeemk') = false;
mp_controls('bl_print_evuvw19_jaeemk_verbose') = false;
mp_controls('bl_print_evuvw19_jmky') = false;

```

9.1.2 Solve VFI and Distribution

```

% Solve the Model to get V working and unemployed
[V_ss,ap_ss,cons_ss,mp_valpol_more_ss] = snw_vfi_main_bisec_vec(mp_params, mp_controls);

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=493.

inc_VFI = mp_valpol_more_ss('inc_VFI');
spouse_inc_VFI = mp_valpol_more_ss('spouse_inc_VFI');
total_inc_VFI = inc_VFI + spouse_inc_VFI;

% COVID year tax
mp_params('a2_covidyr') = mp_params('a2_covidyr_manna_heaven');
% 2020 V and C same as V_SS and cons_ss if tax the same
if (mp_params('a2_covidyr') == mp_params('a2'))
    % mana from heaven
    V_ss_2020 = V_ss;
    cons_ss_2020 = cons_ss;
else
    % change xi and b to for people without unemployment shock
    % solving for employed but 2020 tax results
    % a2_covidyr > a2, we increased tax in 2020 to pay for covid and other
    % costs resolve for both employed and unemployed
    xi = mp_params('xi');
    b = mp_params('b');
    mp_params('xi') = 1;
    mp_params('b') = 0;
    [V_ss_2020,~,cons_ss_2020,~] = snw_vfi_main_bisec_vec(mp_params, mp_controls, V_ss);
    mp_params('xi') = xi;
    mp_params('b') = b;
end
% Solve unemployment
[V_unemp_2020,~,cons_unemp_2020] = snw_vfi_main_bisec_vec(mp_params, mp_controls, V_ss);

Completed SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=d

[Phi_true] = snw_ds_main(mp_params, mp_controls, ap_ss, cons_ss, mp_valpol_more_ss);

Completed SNW_DS_MAIN;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=1493.0859

% Get Matrixes
cl_st_recompute_list = {'a', ...
    'inc', 'inc_unemp', 'spouse_inc', 'spouse_inc_unemp', 'ref_earn_wageind_grid', ...
    'ap_idx_lower_ss', 'ap_idx_higher_ss', 'ap_idx_lower_weight_ss', ...}

```

```
'inc_tot_ygroup_grid'};  
mp_controls('bl_print_precompute_verbose') = false;
```

9.1.3 Pre-Compute Matrixes and YMKY Mass

```
% Pre-compute  
[mp_precompute_res] = snw_hh_precompute(mp_params, mp_controls, cl_st_precompute_list, ap_ss, Phi_tr  
  
Wage quintile cutoffs=0.4645      0.71528      1.0335      1.5632  
Completed SNW_HH_PRECOMPUTE;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time cost=457.  
  
inc_tot_ygroup_grid = mp_precompute_res('inc_tot_ygroup_grid');  
% YMKY Mass  
[Phi_true_jmky] = snw_evuvw19_jmky_mass(mp_params, mp_controls, Phi_true, inc_tot_ygroup_grid);  
  
SNW_EVUVW19_JMKY_MASS Start  
Completed SNW_EVUVW19_JMKY_MASS;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=11.30  
-----  
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)  
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
i      idx     ndim    numel      rowN      colN      sum      mean  
-      ---     ----  -----  -----  -----  -----  -----  
Phi_true      1      1      6      4.37e+07      83      5.265e+05      45.793      1.0479e-06  
Phi_true_jmky  2      2      4      1.6482e+05      82          2010      45.787      0.0002778
```

9.1.4 Solve for 2019 Evuvw With 0 and 2 Checks

Zero checks:

```
% Solve ev 19 JAEEMK  
welf_checks = 0;  
[ev19_jaeemk_check0, ec19_jaeemk_check0, ev20_jaeemk_check0, ec20_jaeemk_check0] = ...  
    snw_evuvw19_jaeemk(...  
    welf_checks, st_solu_type, mp_params, mp_controls, ...  
    V_ss_2020, cons_ss_2020, V_unemp_2020, cons_unemp_2020, mp_precompute_res);  
  
Completed SNW_A4CHK_WRK_BISEC_VEC;welf_checks=0;TR=0.0017225;SNW_MP_PARAM=default_docdense;SNW_MP_CO  
Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=0;TR=0.0017225;xi=0.5;b=0;SNW_MP_PARAM=default_docde  
Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;timeEUEC=8.40  
Completed SNW_EVUVW19_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=4628.523  
  
% Solve ev 19 JMKY  
[ev19_jmky_check0, ec19_jmky_check0] = snw_evuvw19_jmky(...  
    mp_params, mp_controls, ...  
    ev19_jaeemk_check0, ec19_jaeemk_check0, ...  
    Phi_true, Phi_true_jmky, inc_tot_ygroup_grid);  
  
Completed SNW_EVUVW19_JMKY;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=20.5467  
-----  
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)  
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
i      idx     ndim    numel      rowN      colN      sum      mean  
-      ---     ----  -----  -----  -----  -----  -----  
Phi_true      1      1      6      4.37e+07      83      5.265e+05      45.793      1.0479e-06  
Phi_true_jmky  2      2      4      1.6482e+05      82          2010      45.787      0.0002778
```

ec19_jaeemk	3	3	6	4.3173e+07	82	5.265e+05	1.8057e+08	4.182
ec19_jmky	4	4	4	1.6482e+05	82	2010	3.4269e+05	2.079
ev19_jaeemk	5	5	6	4.3173e+07	82	5.265e+05	-1.4106e+08	-3.267
ev19_jmky	6	6	4	1.6482e+05	82	2010	-4.0603e+05	-2.463

Two checks:

```
% Solve ev 19 JAEEMK
welf_checks = 1;
[ev19_jaeemk_check2, ec19_jaeemk_check2, ev20_jaeemk_check2, ec20_jaeemk_check2] = ...
    snw_evuvw19_jaeemk(...,
    welf_checks, st_solu_type, mp_params, mp_controls, ...
    V_ss_2020, cons_ss_2020, V_unemp_2020, cons_unemp_2020, mp_precompute_res);

Completed SNW_A4CHK_WRK_BISEC_VEC;welf_checks=1;TR=0.0017225;SNW_MP_PARAM=default_docdense;SNW_MP_CO
Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=1;TR=0.0017225;xi=0.5;b=0;SNW_MP_PARAM=default_docde
Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;timeEUEC=7.59
Completed SNW_EVUVW19_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=4634.111

% Solve ev 19 JMKY
[ev19_jmky_check2, ec19_jmky_check2] = snw_evuvw19_jmky(...,
    mp_params, mp_controls, ...
    ev19_jaeemk_check2, ec19_jaeemk_check2, ...
    Phi_true, Phi_true_jmky, inc_tot_ygroup_grid);

Completed SNW_EVUVW19_JMKY;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=20.4065
-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

```

i	idx	ndim	numel	rowN	colN	sum	mean	
-	---	----	-----	---	-----	-----	-----	
Phi_true	1	1	6	4.37e+07	83	5.265e+05	45.793	1.0479e-0
Phi_true_jmky	2	2	4	1.6482e+05	82	2010	45.787	0.000277
ec19_jaeemk	3	3	6	4.3173e+07	82	5.265e+05	1.8058e+08	4.182
ec19_jmky	4	4	4	1.6482e+05	82	2010	3.4272e+05	2.079
ev19_jaeemk	5	5	6	4.3173e+07	82	5.265e+05	-1.4077e+08	-3.267
ev19_jmky	6	6	4	1.6482e+05	82	2010	-4.0525e+05	-2.458

Differences between Checks in Expected Value and Expected Consumption

```
mn_V_U_gain_check = ev19_jmky_check2 - ev19_jmky_check0;
mn_MPC_U_gain_share_check = (ec19_jmky_check2 - ec19_jmky_check0)./(welf_checks*mp_params('TR'));
```

9.1.5 Dense 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 = 18:99;
marry_grid = [0,1];
kids_grid = (1:1:mp_params('n_kidsgrid'));
inc_grid = mp_params('inc_grid');
cl_mp_datasetdesc = {};
cl_mp_datasetdesc{1} = containers.Map({'name', 'labval'}, {'age', age_grid});
cl_mp_datasetdesc{2} = containers.Map({'name', 'labval'}, {'marry', marry_grid});
cl_mp_datasetdesc{3} = containers.Map({'name', 'labval'}, {'kids', kids_grid});
cl_mp_datasetdesc{4} = containers.Map({'name', 'labval'}, {'ylower', inc_grid});
```

9.1.6 Analyze Marginal Value and MPC over Y(a,eta), Conditional On Kids, Marry, Age, Education

Income is generated by savings and shocks, what are the income levels generated by all the shock and savings points conditional on kids, marital status, age and educational levels. Plot on the Y axis MPC, and plot on the X axis income levels, use colors to first distinguish between different a levels, then use colors to distinguish between different eta levles.

Set Up date, Select Age 37, unmarried, no kids, lower education:

```
% NaN(n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
% 38 year old, unmarried, no kids, lower educated
% Only Household Head Shock Matters so select up to 'n_eta_H_grid'
mn_V_W_gain_check_use = ev19_jmky_check2 - ev19_jmky_check0;
mn_C_W_gain_check_use = ec19_jmky_check2 - ec19_jmky_check0;
```

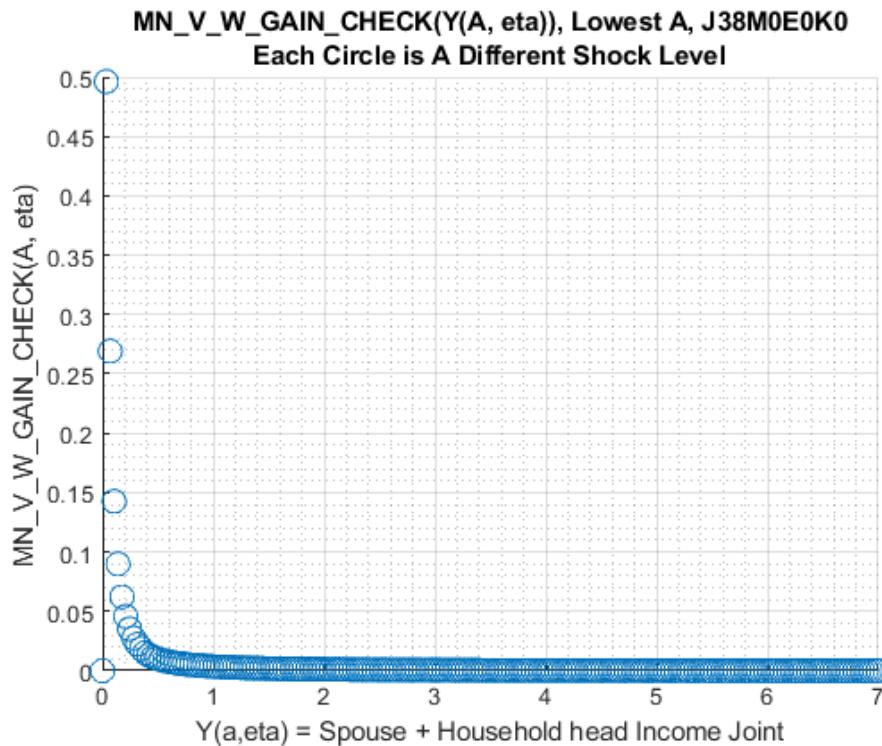
Select Age, Education, Marital, Kids Count:s

```
% Selections
it_age = 21; % +18
it_marital = 1; % 1 = unmarried
it_kids = 1; % 1 = kids is zero
% Select: NaN(n_jgrid-1,n_marriedgrid,n_kidsgrid,n_incgrid);
mn_C_W_gain_check_jemk = mn_C_W_gain_check_use(it_age, it_marital, it_kids, :);
mn_V_W_gain_check_jemk = mn_V_W_gain_check_use(it_age, it_marital, it_kids, :);
% Reshape, so shock is the first dim, a is the second
ar_C_W_gain_check_jemk = mn_C_W_gain_check_jemk(:);
ar_V_W_gain_check_jemk = mn_V_W_gain_check_jemk(:);
```

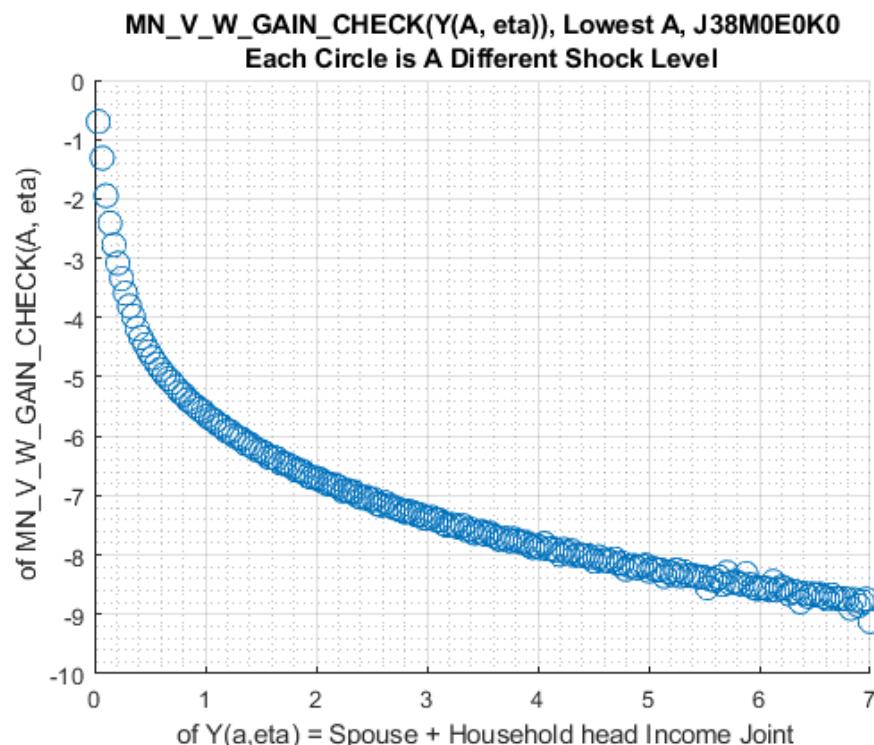
Marginal Value Gains, Color as Shock, Conditional on Age, Marital, Kids, and Education

How do shocks and a impact marginal value. First plot one asset level, variation comes only from increasingly higher shocks:

```
figure();
scatter(inc_grid, ar_V_W_gain_check_jemk, 100);
title({'MN\_V\_W\_GAIN\_CHECK(Y(A, eta)), Lowest A, J38M0EOK0', ...
    'Each Circle is A Different Shock Level'});
xlabel('Y(a,eta) = Spouse + Household head Income Joint');
ylabel('MN\_V\_W\_GAIN\_CHECK(A, eta)');
grid on;
grid minor;
```



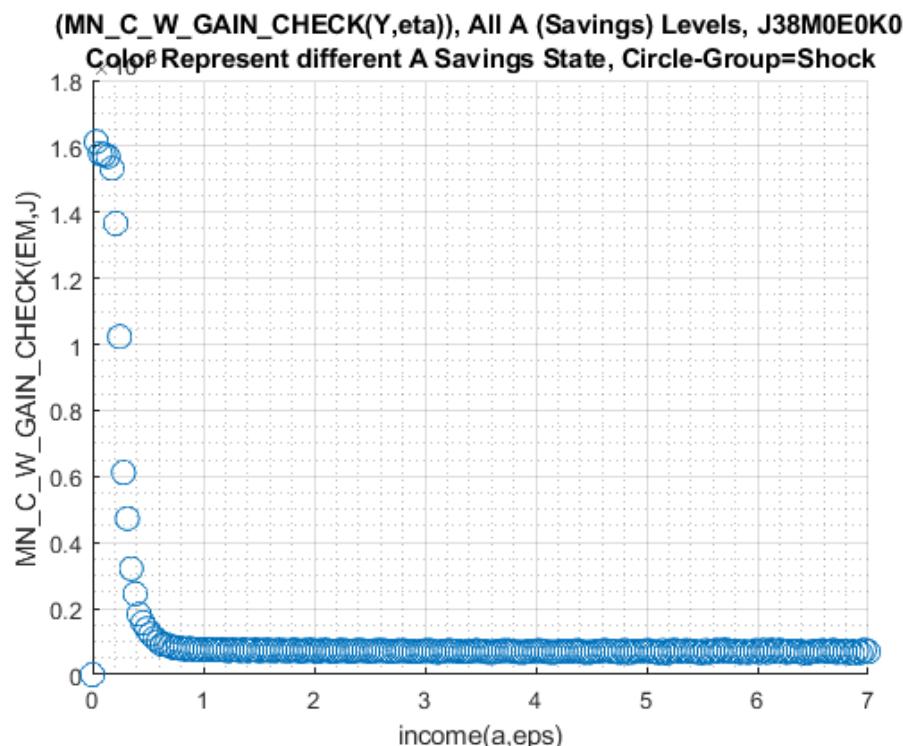
```
figure();
it_shock = 1;
scatter((inc_grid), log(ar_V_W_gain_check_jemk), 100);
title({'MN_V_W_GAIN_CHECK(Y(A, eta)), Lowest A, J38M0E0K0', ...
  'Each Circle is A Different Shock Level'});
xlabel(' of Y(a,eta) = Spouse + Household head Income Joint');
ylabel(' of MN_V_W_GAIN_CHECK(A, eta)');
grid on;
grid minor;
```



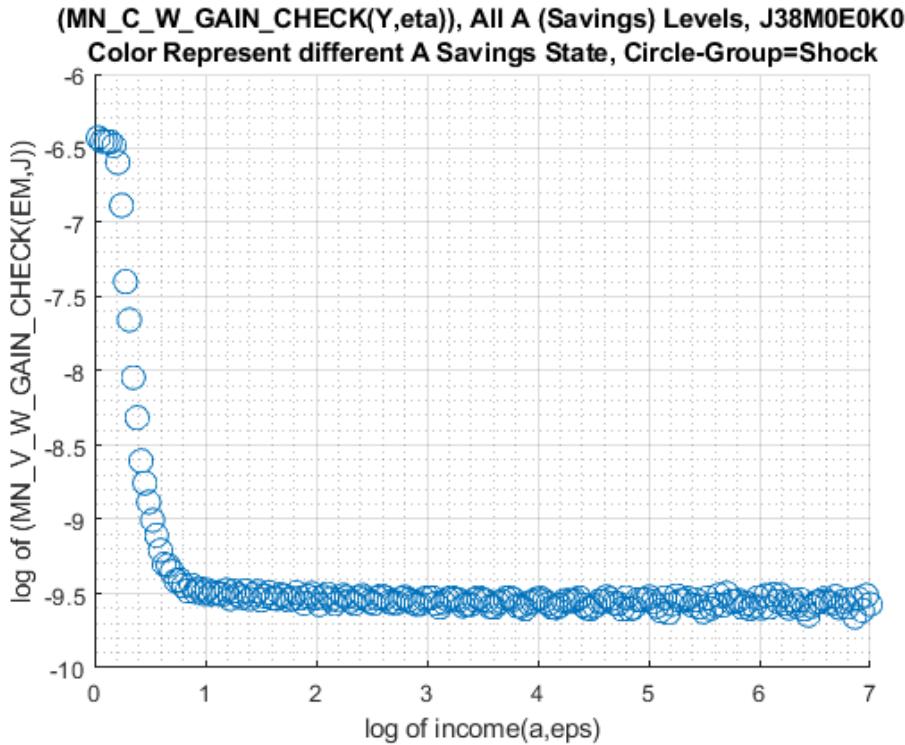
Marginal Consumption Gains, Color as Shock, Conditional on Age, Marital, Kids, and Education

Plot all asset levels:

```
figure();
scatter(inc_grid, ar_C_W_gain_check_jemk, 100);
title({'(MN\_C\_W\_GAIN\_CHECK(Y,eta)), All A (Savings) Levels, J38M0E0K0', ...
    'Color Represent different A Savings State, Circle-Group=Shock'});
xlabel('income(a,eps)');
ylabel('MN\_C\_W\_GAIN\_CHECK(EM,J)');
grid on;
grid minor;
```



```
figure();
scatter((inc_grid), log(ar_C_W_gain_check_jemk), 100);
title({'(MN\_C\_W\_GAIN\_CHECK(Y,eta)), All A (Savings) Levels, J38M0E0K0', ...
    'Color Represent different A Savings State, Circle-Group=Shock'});
xlabel('log of income(a,eps)');
ylabel('log of (MN\_V\_W\_GAIN\_CHECK(EM,J))');
grid on;
grid minor;
```



9.2 2019 Age, Income, Kids, Marry EV and EC All Checks

This is the example vignette for function: [snw_evuvw19_jmky_allchecks](#) from the [PrjOptiSNW Package](#). 2019 integrated over VU and VW

9.2.1 Test SNW_EVUVW19_JMKY_ALLCHECKS Parameters

Save a result that is low in memory cost so that it can be loaded quickly for various allocation tests. Turn off Various Printing Controls. Call function with wide income bins to reduce memory storage and retrievel costs

```
clear all;
% Start mp controls
mp_controls = snw_mp_control('default_test');
% Solve for Unemployment Values
mp_controls('bl_timer') = true;
mp_controls('bl_print_vfi') = false;
mp_controls('bl_print_vfi_verbose') = false;
mp_controls('bl_print_ds') = false;
mp_controls('bl_print_ds_verbose') = true;
mp_controls('bl_print_precompute') = false;
mp_controls('bl_print_precompute_verbose') = false;
mp_controls('bl_print_a4chk') = false;
mp_controls('bl_print_a4chk_verbose') = false;
mp_controls('bl_print_evuvw20_jaeemk') = false;
mp_controls('bl_print_evuvw20_jaeemk_verbose') = false;
mp_controls('bl_print_evuvw19_jaeemk') = false;
mp_controls('bl_print_evuvw19_jaeemk_verbose') = false;
mp_controls('bl_print_evuvw19_jmky') = false;
mp_controls('bl_print_evuvw19_jmky_verbose') = false;
```

Dense default, and unemployment parameters:

```
% default dense load
```

```
% mp_params = snw_mp_param('default_dense');
mp_params = snw_mp_param('default_docdense');
% Unemployment
xi=0.5; % Proportional reduction in income due to unemployment (xi=0 refers to 0 labor income; xi=1
b=0; % Unemployment insurance replacement rate (b=0 refers to no UI benefits; b=1 refers to 100 perc
TR=100/58056; % Value of a wezlfare check (can receive multiple checks). TO DO: Update with alternat
mp_params('xi') = xi;
mp_params('b') = b;
mp_params('TR') = TR;
% Check Count: 89 checks to allow for both the first and the second round
n_welfchecksgrid = 3;
mp_params('n_welfchecksgrid') = n_welfchecksgrid;
mp_params('a2_covidyr') = mp_params('a2_covidyr_manna_heaven');
```

Income bins:

```
% Income Grid
% 4 refers to 4*58056=232224 dollars in 2012USD
% max 7 refers to 7*58056=406392 dollars in 2012USD
% all phase out = (4400/5)*100 + 150000 = 238000
% if 500 dollar interval, need 476 inc groups before 238000
% if have 85 percent of points between 238000,
fl_max_phaseout = 238000;
fl_multiple = 58056;
it_bin_dollar_before_phaseout = 5000;
it_bin_dollar_after_phaseout = 25000;
fl_thres = fl_max_phaseout/fl_multiple;
inc_grid1 = linspace(0,fl_thres,(fl_max_phaseout)/it_bin_dollar_before_phaseout);
inc_grid2 = linspace(fl_thres, 7, (7*fl_multiple-fl_max_phaseout)/it_bin_dollar_after_phaseout);
inc_grid=sort(unique([inc_grid1 inc_grid2]));
mp_params('n_incgrid') = length(inc_grid);
mp_params('inc_grid') = inc_grid;
```

9.2.2 SNW_EVUVW19_JMKY_ALLCHECKS Low Storage Invoke

The simulation here (dense) requires less than 10 GB of memory with 8 workers (8 threads needed), simulating over 88 checks takes with 8 workers

```
st_solu_type = 'biseq_vec';
bl_parfor = false;
it_workers = 1;
bl_export = false;
bl_load_mat = false;
snm_suffix = ['_test_ybin' num2str(it_bin_dollar_before_phaseout)];
[ev19_jmky_allchecks, ec19_jmky_allchecks, output] = ...
    snw_evuvw19_jmky_allchecks(mp_params, mp_controls, st_solu_type, ...
    bl_parfor, it_workers, ...
    bl_export, bl_load_mat, snm_suffix);
```

```
Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=331.
Completed SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=d
Completed SNW_DS_MAIN_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=916.1458
Wage quintile cutoffs=0.4645      0.71528      1.0335      1.5632
Completed SNW_HH_PRECOMPUTE;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time cost=361.
SNW_EVUVW19_JMKY_MASS Start
Completed SNW_EVUVW19_JMKY_MASS;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=5.503
-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_outcomes ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	ndim	numel	rowN	colN	sum	mean	
	-	---	----	-----	----	-----	-----	-----	---
Phi_true	1	1	6	4.37e+07	83	5.265e+05	45.793	1.0479e-06	1.
Phi_true_jmky	2	2	4	42640	82	520	45.787	0.0010738	0.

SNW_EVUVW19_JMKY_ALLCHECKS Start
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```
Completed SNW_A4CHK_WRK_BISEC_VEC;welf_checks=0;TR=0.0017225;SNW_MP_PARAM=default_docdense;SNW_MP_CO  
Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=0;TR=0.0017225;xi=0.5;b=0;SNW_MP_PARAM=default_docde  
Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;timeEUEC=8.73  
Completed SNW_EVUVW19_JAEEMK_FOC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=14.7  
Completed SNW_EVUVW19_JMKY;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=11.4225  
SNW_EVUVW19_JMKY_ALLCHECKS: Finished Check 0 of 2, time=188.6232  
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

Completed SNW_A4CHK_WRK_BISEC_VEC;welf_check

Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=1;TR=0.0017225;xi=0.5;b=0;SNW_MP_PARAM=default_docdense
Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;timeEUEC=8.38
Completed SNW_EVUVW19_JAEEMK_FOC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=14.8
Completed SNW_EVUVW19_JMKY;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=11.4378
SNW_EVUVW19_JMKY_ALLCHECKS: Finished Check 1 of 2, time=189.5593
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Completed SNW A4CHK WBK BTSEC VEC:welf check

Completed SNW_A4CHK_with_BISEC_VEC;welf_checks=2;TR=0.0017225;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=188.6191
Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=2;TR=0.0017225;xi=0.5;b=0;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=8.1211
Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;timeEUEC=8.1211
Completed SNW_EVUVW19_JAEEMK_FOC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=14.8041
Completed SNW_EVUVW19_JMKY;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=10.9041
SNW_EVUVW19_JMKY_ALLCHECKS: Finished Check 2 of 2, time=188.6191
Completed SNW_EVUVW19_JMKY_ALLCHECKS;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=188.6191

Completed SNW_EVUVW19_JMKY_ALLCHECKS; SNW_MP_PARAM=default_docdens

xx

CONTAINER NAME: mp_outcomes ND Array (Matrix etc)

i	idx	ndim	numel	rowN	colN	sum
---	-----	------	-------	------	------	-----

NAME
FATHER
MOTHER
BROTHERS
SISTERS
WIFE
HUSBAND
TOWN
COUNTRY
SURN.

Output 1 1 2 1.0307e+06 1.1452e+05 9 6.804e

ec19_jmky_allchecks 2 2 5 1.2792e+05 3 42640 2.6964e

ec19_imkv_allchecks_posmass 3 3 2 1.1452e+05 1.1452e+05 1 2.6964e-05

ev19_imkv_allchecks 4 4 5 1 2792e+05 3 42640 -8 4257e-01

ev19_jmky_allchecks 4 4 1.2792e+03 3 42640 -8.4257e-01
 ev19_imky_allchecks_noassoc 5 5 1.1452e-05 1 1.1452e-05 1 8.4257e-01

ev19_jmky_allchecks_posmass 5 5 2 1.1452e+05 1.1452e+05 1 -8.4257e

xxx TABLE:Output xxxxxxxxxxxxxxxxxxxx

c1 c2 c3 c4 c6 c7 c8 c9

— — — —

r1	18	0	0	0	2.9349e-05	-0.57722	-204.67	0.069778
r2	18	0	0	1	2.9349e-05	-0.57722	-204.21	0.07068
r3	18	0	0	2	2.9349e-05	-0.57722	-203.76	0.07159
r4	19	0	0	0	2.5821e-05	0.42278	-196.02	0.07008
r5	19	0	0	1	2.5821e-05	0.42278	-195.57	0.071671
r114518	87	1	4	1	1.897e-42	4.2413	3.8911	14.055
r114519	87	1	4	2	1.897e-42	4.2413	3.8911	14.055
r114520	88	1	4	0	1.5306e-60	4.2556	3.7289	15.409
r114521	88	1	4	1	1.5306e-60	4.2556	3.7289	15.409
r114522	88	1	4	2	1.5306e-60	4.2556	3.7289	15.409

xxx TABLE:ec19_jmky_allchecks xxxxxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c42637	c42638	c42639	c42640
	-----	-----	-----	-----	-----	-----	-----	-----
r1	0.069778	0.07008	0.073462	0.07559	0	0	0	0
r2	0.07068	0.071671	0.075078	0.077198	0	0	0	0
r3	0.07159	0.073241	0.076648	0.078753	0	0	0	0

xxx TABLE:ec19_jmky_allchecks_posmass xxxxxxxxxxxxxxxxxxxx

	c1	-----
r1	0.069778	-----
r2	0.07068	-----
r3	0.07159	-----
r4	0.07008	-----
r5	0.071671	-----
r114518	14.055	-----
r114519	14.055	-----
r114520	15.409	-----
r114521	15.409	-----
r114522	15.409	-----

xxx TABLE:ev19_jmky_allchecks_posmass xxxxxxxxxxxxxxxxxxxx

	c1	c2	c3	c4	c42637	c42638	c42639	c42640
	-----	-----	-----	-----	-----	-----	-----	-----
r1	-204.67	-196.02	-187.61	-184.45	0	0	0	0
r2	-204.21	-195.57	-187.2	-184.07	0	0	0	0
r3	-203.76	-195.13	-186.8	-183.69	0	0	0	0

xxx TABLE:ev19_jmky_allchecks_posmass xxxxxxxxxxxxxxxxxxxx

	c1	-----
r1	-204.67	-----
r2	-204.21	-----
r3	-203.76	-----
r4	-196.02	-----
r5	-195.57	-----
r114518	3.8911	-----
r114519	3.8911	-----
r114520	3.7289	-----
r114521	3.7289	-----
r114522	3.7289	-----

Chapter 10

Taxes

10.1 Compute for Equilibrium Tax

Taking advantage of `snw_calibrate_beta_norm_gdp` from the [PrjOptiSNW Package](#), this function solves for equilibrium tax rate.

10.1.1 Parameter Controls

```
clear all;
mp_params = snw_mp_param('default_docdense');
xi=0; % Proportional reduction in income due to unemployment (xi=0 refers to 0 labor income; xi=1 refers to 100%)
b=1; % Unemployment insurance replacement rate (b=0 refers to no UI benefits; b=1 refers to 100%)
mp_params('xi') = xi;
mp_params('b') = b;
mp_controls = snw_mp_control('default_test');
```

Parameters for COVID related Costs:

```
% Average check per household, given COVID actual policy payment schedule
% And given distribution. The number is from averaging over the actual
% allocations given distribution.
Covid_checks_per_capita = 18.7255856*100/58056;
% Covid_checks_per_capita = 0;
% which tax parameter to change a2 is the deafult, a0 shifts max tax rate
bl_adjust_a0 = false;
bl_load_existing = false;
```

Graph Controls etc:

```
mp_controls('bl_timer') = true;
mp_controls('bl_print_vfi') = false;
mp_controls('bl_print_vfi_verbose') = false;
mp_controls('bl_print_ds') = false;
mp_controls('bl_print_ds_verbose') = false;
mp_controls('bl_print_find_tax_rate') = true;
mp_controls('bl_print_find_tax_rate_verbose') = true;
```

10.1.2 Solve for New Tax Rate

Solve for Equilibrium Tax rate that clears government costs and income. In the extreme bounding exercise, we assume the government will pay COVID costs all in one year. This is to test whether an extreme tax scenario will lead to changes in allocation results.

Given the checks that the government hands out and the taxes imposed, individual resources post-tax are different in 2020. Households' savings decisions in 2020 vary with taxes and checks. However, the

policy function post 2020 shifts back to the previous non-COVID world's policy function because the COVID shock is an one period surprise shock.

```
a2 = snw_find_tax_rate(mp_params, mp_controls, Covid_checks_per_capita, bl_adjust_a0, bl_load_existi

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=319.
Completed SNW_DS_MAIN_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=919.8561
Wage quintile cutoffs=0.4645      0.71528      1.0335      1.5632
Y_inc_agg=64.7962
A_agg=194.5563
Y_inc_agg_per_capita_1=1.415
A_per_capita=4.2486
Gov_cons_per_capita=0.24869
Covid_checks_share_of_GDP=0.022795
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:1 of 83, time-this-age:0.41178
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:2 of 83, time-this-age:0.30409
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:3 of 83, time-this-age:0.30899
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:4 of 83, time-this-age:0.30567
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:5 of 83, time-this-age:0.30909
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:6 of 83, time-this-age:0.30687
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:7 of 83, time-this-age:0.3075
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:8 of 83, time-this-age:0.32568
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:9 of 83, time-this-age:0.30706
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:10 of 83, time-this-age:0.30656
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:11 of 83, time-this-age:0.30787
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:12 of 83, time-this-age:0.30388
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:13 of 83, time-this-age:0.30705
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:14 of 83, time-this-age:0.30526
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:15 of 83, time-this-age:0.30613
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:16 of 83, time-this-age:0.30936
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:17 of 83, time-this-age:0.30467
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:18 of 83, time-this-age:0.30324
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:19 of 83, time-this-age:0.30123
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:20 of 83, time-this-age:0.30061
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:21 of 83, time-this-age:0.30007
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:22 of 83, time-this-age:0.30015
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:23 of 83, time-this-age:0.30413
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:24 of 83, time-this-age:0.30042
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:25 of 83, time-this-age:0.3027
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:26 of 83, time-this-age:0.30356
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:27 of 83, time-this-age:0.30119
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:28 of 83, time-this-age:0.30155
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:29 of 83, time-this-age:0.30464
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:30 of 83, time-this-age:0.30107
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:31 of 83, time-this-age:0.30277
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:32 of 83, time-this-age:0.32498
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:33 of 83, time-this-age:0.30004
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:34 of 83, time-this-age:0.29931
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:35 of 83, time-this-age:0.29904
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:36 of 83, time-this-age:0.30101
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:37 of 83, time-this-age:0.30317
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:38 of 83, time-this-age:0.30457
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:39 of 83, time-this-age:0.30586
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:40 of 83, time-this-age:0.30363
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:41 of 83, time-this-age:0.30335
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:42 of 83, time-this-age:0.30586
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:43 of 83, time-this-age:0.30422
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:44 of 83, time-this-age:0.30746
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:45 of 83, time-this-age:0.30742
```

SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:46 of 83, time-this-age:0.30865
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:47 of 83, time-this-age:0.30668
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:48 of 83, time-this-age:0.32596
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:49 of 83, time-this-age:0.31867
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:50 of 83, time-this-age:0.3201
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:51 of 83, time-this-age:0.32131
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:52 of 83, time-this-age:0.31776
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:53 of 83, time-this-age:0.32034
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:54 of 83, time-this-age:0.32022
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:55 of 83, time-this-age:0.31828
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:56 of 83, time-this-age:0.32025
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:57 of 83, time-this-age:0.32119
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:58 of 83, time-this-age:0.34504
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:59 of 83, time-this-age:0.32159
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:60 of 83, time-this-age:0.31967
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:61 of 83, time-this-age:0.3213
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:62 of 83, time-this-age:0.32097
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:63 of 83, time-this-age:0.32492
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:64 of 83, time-this-age:0.32049
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:65 of 83, time-this-age:0.32574
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:66 of 83, time-this-age:0.3202
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:67 of 83, time-this-age:0.32328
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:68 of 83, time-this-age:0.32231
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:69 of 83, time-this-age:0.32269
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:70 of 83, time-this-age:0.34522
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:71 of 83, time-this-age:0.32456
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:72 of 83, time-this-age:0.325
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:73 of 83, time-this-age:0.32367
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:74 of 83, time-this-age:0.31997
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:75 of 83, time-this-age:0.32636
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:76 of 83, time-this-age:0.32233
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:77 of 83, time-this-age:0.32291
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:78 of 83, time-this-age:0.32344
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:79 of 83, time-this-age:0.32237
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:80 of 83, time-this-age:0.32156
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:81 of 83, time-this-age:0.32284
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:82 of 83, time-this-age:0.32193
SNW_FIND_TAX_RATE: Aggregation, Finished Age Group:83 of 83, time-this-age:0.32499
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=1.7855; a0=0.258; err=0.18712
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=2.0367; a0=0.258; err=0.16097
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=2.2819; a0=0.258; err=0.14064
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=2.5211; a0=0.258; err=0.12443
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=2.7542; a0=0.258; err=0.11122
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=2.9813; a0=0.258; err=0.10028
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=3.2027; a0=0.258; err=0.091075
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=3.4184; a0=0.258; err=0.083235
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=3.6286; a0=0.258; err=0.076483
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=3.8334; a0=0.258; err=0.070612
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=4.0331; a0=0.258; err=0.065464
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=4.2278; a0=0.258; err=0.060916
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=4.4176; a0=0.258; err=0.056872
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=4.6026; a0=0.258; err=0.053254
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=4.7832; a0=0.258; err=0.05
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=4.9592; a0=0.258; err=0.047058
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=5.131; a0=0.258; err=0.044388
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=5.2986; a0=0.258; err=0.041953
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=5.4622; a0=0.258; err=0.039726
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=5.6218; a0=0.258; err=0.037681

SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=5.7777; a0=0.258; err=0.035798
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=5.9298; a0=0.258; err=0.034058
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=6.0783; a0=0.258; err=0.032447
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=6.2233; a0=0.258; err=0.030951
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=6.365; a0=0.258; err=0.029558
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=6.5033; a0=0.258; err=0.028259
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=6.6384; a0=0.258; err=0.027045
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=6.7704; a0=0.258; err=0.025908
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=6.8993; a0=0.258; err=0.024841
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=7.0253; a0=0.258; err=0.023838
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=7.1484; a0=0.258; err=0.022894
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=7.2687; a0=0.258; err=0.022004
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=7.3862; a0=0.258; err=0.021164
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=7.5011; a0=0.258; err=0.02037
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=7.6134; a0=0.258; err=0.019618
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=7.7232; a0=0.258; err=0.018905
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=7.8305; a0=0.258; err=0.018229
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=7.9354; a0=0.258; err=0.017586
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.0379; a0=0.258; err=0.016976
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.1382; a0=0.258; err=0.016394
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.2363; a0=0.258; err=0.01584
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.3321; a0=0.258; err=0.015312
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.4259; a0=0.258; err=0.014807
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.5175; a0=0.258; err=0.014325
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.6072; a0=0.258; err=0.013865
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.6949; a0=0.258; err=0.013424
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.7807; a0=0.258; err=0.013002
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.8645; a0=0.258; err=0.012598
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=8.9466; a0=0.258; err=0.01221
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.0269; a0=0.258; err=0.011838
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.1054; a0=0.258; err=0.011481
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.1822; a0=0.258; err=0.011138
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.2573; a0=0.258; err=0.010808
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.3309; a0=0.258; err=0.010491
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.4028; a0=0.258; err=0.010186
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.4732; a0=0.258; err=0.0098926
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.542; a0=0.258; err=0.0096097
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.6094; a0=0.258; err=0.0093372
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.6753; a0=0.258; err=0.0090744
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.7398; a0=0.258; err=0.008821
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.8029; a0=0.258; err=0.0085765
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.8647; a0=0.258; err=0.0083404
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.9252; a0=0.258; err=0.0081125
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=9.9843; a0=0.258; err=0.0078923
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.0422; a0=0.258; err=0.0076795
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.0989; a0=0.258; err=0.0074738
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.1543; a0=0.258; err=0.0072748
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.2086; a0=0.258; err=0.0070823
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.2617; a0=0.258; err=0.0068961
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.3137; a0=0.258; err=0.0067158
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.3646; a0=0.258; err=0.0065412
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.4144; a0=0.258; err=0.006372
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.4632; a0=0.258; err=0.0062082
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.5109; a0=0.258; err=0.0060494
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.5576; a0=0.258; err=0.0058954
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.6033; a0=0.258; err=0.0057461
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.6481; a0=0.258; err=0.0056013
SNW_FIND_TAX_RATE tax a2 or a0 adjustments; a2=10.6919; a0=0.258; err=0.0054608

```
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=10.7348;a0=0.258;err=0.0053245
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=10.7768;a0=0.258;err=0.0051922
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=10.8179;a0=0.258;err=0.0050637
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=10.8582;a0=0.258;err=0.0049389
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=10.8976;a0=0.258;err=0.0048178
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=10.9361;a0=0.258;err=0.0047001
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=10.9739;a0=0.258;err=0.0045857
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.0109;a0=0.258;err=0.0044745
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.0471;a0=0.258;err=0.0043665
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.0825;a0=0.258;err=0.0042614
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.1172;a0=0.258;err=0.0041593
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.1512;a0=0.258;err=0.0040599
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.1844;a0=0.258;err=0.0039633
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.217;a0=0.258;err=0.0038692
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.2489;a0=0.258;err=0.0037777
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.2801;a0=0.258;err=0.0036887
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.3107;a0=0.258;err=0.003602
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.3406;a0=0.258;err=0.0035177
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.3699;a0=0.258;err=0.0034355
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.3986;a0=0.258;err=0.0033555
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.4267;a0=0.258;err=0.0032776
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.4542;a0=0.258;err=0.0032017
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.4812;a0=0.258;err=0.0031278
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.5076;a0=0.258;err=0.0030558
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.5334;a0=0.258;err=0.0029856
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.5587;a0=0.258;err=0.0029172
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.5835;a0=0.258;err=0.0028505
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.6077;a0=0.258;err=0.0027855
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.6315;a0=0.258;err=0.0027222
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.6548;a0=0.258;err=0.0026604
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.6775;a0=0.258;err=0.0026002
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.6998;a0=0.258;err=0.0025415
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.7217;a0=0.258;err=0.0024842
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.7431;a0=0.258;err=0.0024283
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.764;a0=0.258;err=0.0023738
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.7846;a0=0.258;err=0.0023207
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.8046;a0=0.258;err=0.0022688
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.8243;a0=0.258;err=0.0022182
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.8436;a0=0.258;err=0.0021688
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.8625;a0=0.258;err=0.0021206
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.8809;a0=0.258;err=0.0020735
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.899;a0=0.258;err=0.0020276
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.9168;a0=0.258;err=0.0019828
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.9341;a0=0.258;err=0.001939
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.9511;a0=0.258;err=0.0018963
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.9678;a0=0.258;err=0.0018546
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=11.9841;a0=0.258;err=0.0018138
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.0001;a0=0.258;err=0.0017741
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.0157;a0=0.258;err=0.0017352
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.031;a0=0.258;err=0.0016973
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.046;a0=0.258;err=0.0016602
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.0607;a0=0.258;err=0.001624
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.0751;a0=0.258;err=0.0015887
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.0892;a0=0.258;err=0.0015542
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.103;a0=0.258;err=0.0015204
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.1165;a0=0.258;err=0.0014875
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.1298;a0=0.258;err=0.0014552
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.1427;a0=0.258;err=0.0014238
```

```
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.1554;a0=0.258;err=0.001393
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.1679;a0=0.258;err=0.001363
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.1801;a0=0.258;err=0.0013336
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.192;a0=0.258;err=0.0013049
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.2037;a0=0.258;err=0.0012768
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.2151;a0=0.258;err=0.0012494
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.2263;a0=0.258;err=0.0012226
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.2373;a0=0.258;err=0.0011964
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.2481;a0=0.258;err=0.0011708
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.2586;a0=0.258;err=0.0011457
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.2689;a0=0.258;err=0.0011213
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.279;a0=0.258;err=0.0010973
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.2889;a0=0.258;err=0.0010739
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.2986;a0=0.258;err=0.001051
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.3081;a0=0.258;err=0.0010287
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.3174;a0=0.258;err=0.0010068
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.3265;a0=0.258;err=0.0009854
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.3355;a0=0.258;err=0.00096448
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.3442;a0=0.258;err=0.00094402
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.3528;a0=0.258;err=0.00092401
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.3612;a0=0.258;err=0.00090444
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.3694;a0=0.258;err=0.0008853
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.3774;a0=0.258;err=0.00086657
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.3853;a0=0.258;err=0.00084826
SNW_FIND_TAX_RATE tax a2 or a0 adjustments;a2=12.393;a0=0.258;err=0.00083035
```

Chapter 11

Calibration

11.1 Model Calibration

Taking advantage of `snw_calibrate_beta_norm_gdp` from the [PrjOptiSNW Package](#), this function calibrates the discount factor and also solves for the normalizing constant.

11.1.1 Calibrate Parameter Controls for SNW Functions

Set up controls for shock process and tiny/small/dense/densemore

```
clear all;
bl_print_mp_params = false;
% st_shock_method = 'rouwenhorst';
st_shock_method = 'tauchen';
% st_param_group = 'default_tiny';
% st_param_group = 'default_small';
% st_param_group = 'default_base';
% st_param_group = 'default_dense';
% st_param_group = 'default_moredense';
st_param_group = 'default_docdense';
mp_params = snw_mp_param(st_param_group, bl_print_mp_params, st_shock_method);
Pop = mp_params('Pop');
```

Set up print defaults

```
mp_controls = snw_mp_control('default_test');
mp_controls('bl_timer') = true;
mp_controls('bl_print_vfi') = false;
mp_controls('bl_print_vfi_verbose') = false;
mp_controls('bl_print_ds') = false;
mp_controls('bl_print_ds_verbose') = false;
```

11.1.2 Calibrate Routine

Test this for 3 iterations

```
%% Calibration
err=1;
tol=0.005;
it_counter = 1;
while err>tol && it_counter <= 10
    disp('');
    it=1;
    while it>0
```

```
% Solve optimization problem and get the distribution
tm_start_a2 = tic;
a2_old = mp_params('a2');
[Phi_true,~,A_agg,Y_inc_agg,it,mp_dsvfi_results, a2] = snw_ds_main(mp_params, mp_controls);
mp_params('a2') = a2;
tm_end_a2 = toc(tm_start_a2);
disp(['a2_old:' num2str(a2_old) ', a2_new:' num2str(a2) ', tm_end_a2:' num2str(tm_end_a2)])
end

% Get Stats
mp_cl_mt_xyz_of_s = mp_dsvfi_results('mp_cl_mt_xyz_of_s');
tb_outcomes = mp_cl_mt_xyz_of_s('tb_outcomes');
A_agg_alt = tb_outcomes{'a_ss', 'mean'}*sum(Pop);
A_prime_agg_alt = tb_outcomes{'ap_ss', 'mean'}*sum(Pop);
Y_inc_agg_alt = tb_outcomes{'y_all', 'mean'}*sum(Pop);
Y_inc_median = tb_outcomes{'y_all', 'p50'};

% Comparison
name='Median household income (target=1.0)=';
name2=[name,num2str(Y_inc_median)];
disp(name2);
name='Aggregate wealth to aggregate income (target=3.0)=';
name2=[name,num2str(A_agg/Y_inc_agg)];
disp(name2);

err1=abs(Y_inc_median-1.0); % Target: Median household income (normalized to 1 in the model)
err2=abs((A_agg/Y_inc_agg)-3.0); % Target: Annual capital/income ratio of 3

err=max(err1,err2);

% Beta and Theta
theta = mp_params('theta');
beta = mp_params('beta');
param_update=[theta;beta];

if err>tol

    theta=theta*((1.0/Y_inc_median)^0.2); % Normalize theta such that median household income eq
    beta=beta*((3.0/(A_agg/Y_inc_agg))^0.025); % Calibrate beta such that annual capital/income

end
mp_params('theta') = theta;
mp_params('beta') = beta;

param_update=[param_update(1,1),theta;param_update(2,1),beta];

it_counter = it_counter + 1;
name='Old/updated theta:';
st_theta=[name, num2str(param_update(1,:))];
name='Old/updated beta:';
st_beta=[name,num2str(param_update(2,:))];
disp(['counter=' num2str(it_counter) ...
';beta=' num2str(beta) ...
';theta=' num2str(theta)]);
end
```

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=568.

```
Completed SNW_DS_MAIN;SNW_MP_PARAM=default_docdense;SNW_MP_CONTROL=default_test;time=2318.9782
a2_old:1.5286, a2_new:1.5286, tm_end_a2:3102.469
Median household income (target=1.0)=0.99853
Aggregate wealth to aggregate income (target=3.0)=3.0026
counter=2;beta=0.97116;theta=0.56523
```


Chapter 12

Summary Statistics

12.1 2019 Full States MPC and Distributional Statistics by Marital, Kids, and Income Groups.

In the file here, we consider marital, kids and income groups, and summarize various statistics for each bin.

12.1.1 Test SNW_EVUVW19_JAEEMK Defaults Dense

VFI and Distribution

Call the function with defaults.

```
clear all;
st_solu_type = 'bisec_vec';
bl_save_csv = false;

% Solve the VFI Problem and get Value Function
% mp_params = snw_mp_param('default_dense');
% mp_params = snw_mp_param('default_docdense');
mp_params = snw_mp_param('default_moredense_a65zh133zs5_e2m2');
mp_controls = snw_mp_control('default_test');

% set Unemployment Related Variables
xi=0.5; % Proportional reduction in income due to unemployment (xi=0 refers to 0 labor income; xi=1
b=1; % Unemployment insurance replacement rate (b=0 refers to no UI benefits; b=1 refers to 100 perc
TR=100/58056; % Value of a welfare check (can receive multiple checks). TO DO: Update with alternati

mp_params('xi') = xi;
mp_params('b') = b;
mp_params('TR') = TR;

% Solve for Unemployment Values
mp_controls('bl_print_vfi') = false;
mp_controls('bl_print_vfi_verbose') = false;
mp_controls('bl_print_ds') = true;
mp_controls('bl_print_ds_verbose') = true;
mp_controls('bl_print_precompute') = false;
mp_controls('bl_print_precompute_verbose') = false;
mp_controls('bl_print_a4chk') = false;
mp_controls('bl_print_a4chk_verbose') = false;
mp_controls('bl_print_evuvw20_jaeemk') = false;
mp_controls('bl_print_evuvw20_jaeemk_verbose') = false;
```

```

mp_controls('bl_print_evuvvw19_jaeemk') = false;
mp_controls('bl_print_evuvvw19_jaeemk_verbose') = false;

% Solve the Model to get V working and unemployed
[V_ss,ap_ss,cons_ss,mp_valpol_more_ss] = snw_vfi_main_bisec_vec(mp_params, mp_controls);

Completed SNW_VFI_MAIN_BISEC_VEC;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=defa

inc_VFI = mp_valpol_more_ss('inc_VFI');
spouse_inc_VFI = mp_valpol_more_ss('spouse_inc_VFI');
total_inc_VFI = inc_VFI + spouse_inc_VFI;
% tax during covid year
mp_params('a2_covidyr') = mp_params('a2_covidyr_manna_heaven');

% Solve unemployment
[V_unemp,~,cons_unemp,~] = snw_vfi_main_bisec_vec(mp_params, mp_controls, V_ss);

Completed SNW_VFI_MAIN_BISEC_VEC 1 Period Unemp Shock;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2

[Phi_true, Phi_adj, A_agg, Y_inc_agg, ~, mp_dsvfi_results] = snw_ds_main_vec(mp_params, mp_controls, ~);

SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:1 of 82, time-this-age:1.074
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:2 of 82, time-this-age:20.5148
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:3 of 82, time-this-age:23.4908
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:4 of 82, time-this-age:28.525
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:5 of 82, time-this-age:33.2054
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:6 of 82, time-this-age:35.3197
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:7 of 82, time-this-age:37.5611
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:8 of 82, time-this-age:40.226
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:9 of 82, time-this-age:44.3653
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:10 of 82, time-this-age:48.3751
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:11 of 82, time-this-age:49.4182
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:12 of 82, time-this-age:50.6325
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:13 of 82, time-this-age:51.0802
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:14 of 82, time-this-age:52.1717
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:15 of 82, time-this-age:53.2068
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:16 of 82, time-this-age:53.6567
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:17 of 82, time-this-age:53.8811
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:18 of 82, time-this-age:55.0892
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:19 of 82, time-this-age:55.6717
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:20 of 82, time-this-age:56.2143
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:21 of 82, time-this-age:56.5704
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:22 of 82, time-this-age:57.0081
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:23 of 82, time-this-age:57.1682
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:24 of 82, time-this-age:57.3671
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:25 of 82, time-this-age:57.5453
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:26 of 82, time-this-age:57.8356
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:27 of 82, time-this-age:58.0491
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:28 of 82, time-this-age:57.9265
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:29 of 82, time-this-age:57.6332
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:30 of 82, time-this-age:58.1269
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:31 of 82, time-this-age:57.7606
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:32 of 82, time-this-age:57.5816
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:33 of 82, time-this-age:57.3361
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:34 of 82, time-this-age:57.7288
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:35 of 82, time-this-age:56.9154
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:36 of 82, time-this-age:57.2866
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:37 of 82, time-this-age:57.1634
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:38 of 82, time-this-age:57.0388

```

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:39 of 82, time-this-age:56.6859
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:40 of 82, time-this-age:56.7277
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:41 of 82, time-this-age:56.9976
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:42 of 82, time-this-age:56.6711
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:43 of 82, time-this-age:56.7355
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:44 of 82, time-this-age:56.6671
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:45 of 82, time-this-age:56.1114
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:46 of 82, time-this-age:55.9357
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:47 of 82, time-this-age:55.9514
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:48 of 82, time-this-age:55.4533
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:49 of 82, time-this-age:58.5505
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:50 of 82, time-this-age:59.402
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:51 of 82, time-this-age:59.5814
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:52 of 82, time-this-age:59.4987
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:53 of 82, time-this-age:59.3449
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:54 of 82, time-this-age:59.6498
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:55 of 82, time-this-age:59.3396
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:56 of 82, time-this-age:59.4903
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:57 of 82, time-this-age:59.4659
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:58 of 82, time-this-age:59.2382
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:59 of 82, time-this-age:58.2574
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:60 of 82, time-this-age:58.4884
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:61 of 82, time-this-age:58.2825
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:62 of 82, time-this-age:57.4508
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:63 of 82, time-this-age:56.9986
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:64 of 82, time-this-age:56.5337
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:65 of 82, time-this-age:55.94
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:66 of 82, time-this-age:54.1804
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:67 of 82, time-this-age:53.4807
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:68 of 82, time-this-age:52.222
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:69 of 82, time-this-age:51.6643
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:70 of 82, time-this-age:50.7393
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:71 of 82, time-this-age:49.5324
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:72 of 82, time-this-age:47.7517
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:73 of 82, time-this-age:45.9439
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:74 of 82, time-this-age:44.385
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:75 of 82, time-this-age:42.9
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:76 of 82, time-this-age:41.3804
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:77 of 82, time-this-age:35.089
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:78 of 82, time-this-age:33.9143
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:79 of 82, time-this-age:32.9597
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:80 of 82, time-this-age:26.3587
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:81 of 82, time-this-age:25.2198
SNW_DS_MAIN_VEC ACUMU MASS: Finished Age Group:82 of 82, time-this-age:22.8558
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:1 of 82, time-this-age:0.50074
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:2 of 82, time-this-age:0.078102
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:3 of 82, time-this-age:0.077705
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:4 of 82, time-this-age:0.077939
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:5 of 82, time-this-age:0.07796
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:6 of 82, time-this-age:0.078664
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:7 of 82, time-this-age:0.077012
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:8 of 82, time-this-age:0.077566
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:9 of 82, time-this-age:0.076968
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:10 of 82, time-this-age:0.076874
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:11 of 82, time-this-age:0.07674
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:12 of 82, time-this-age:0.07736
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:13 of 82, time-this-age:0.07804
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:14 of 82, time-this-age:0.077614

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

```

SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:73 of 82, time-this-age:0.073002
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:74 of 82, time-this-age:0.073612
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:75 of 82, time-this-age:0.073039
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:76 of 82, time-this-age:0.073474
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:77 of 82, time-this-age:0.073582
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:78 of 82, time-this-age:0.076234
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:79 of 82, time-this-age:0.073668
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:80 of 82, time-this-age:0.073745
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:81 of 82, time-this-age:0.073108
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:82 of 82, time-this-age:0.072892
SNW_DS_MAIN NORMALIZE MASS: Finished Age Group:83 of 82, time-this-age:0.073316
SNW_DS_MAIN: Share of population with assets equal to upper bound on asset grid:6.0111e-06
SNW_DS_MAIN: Accidental bequests are thrown in the ocean
SNW_DS_MAIN_VEC tax and spend;it=1;err=0.0010205
SNW_DS_MAIN_VEC tax and spend;it=2;err=0.0008547
SNW_DS_MAIN_VEC tax and spend;it=3;err=0.0007159
SNW_DS_MAIN_VEC tax and spend;it=4;err=0.00059969
SNW_DS_MAIN_VEC tax and spend;it=5;err=0.00050237
SNW_DS_MAIN_VEC tax and spend;it=6;err=0.00042087
SNW_DS_MAIN_VEC tax and spend;it=7;err=0.00035261
SNW_DS_MAIN_VEC tax and spend;it=8;err=0.00029542
SNW_DS_MAIN_VEC tax and spend;it=9;err=0.00024752
SNW_DS_MAIN_VEC tax and spend;it=10;err=0.0002074
SNW_DS_MAIN_VEC tax and spend;it=11;err=0.00017378
SNW_DS_MAIN_VEC tax and spend;it=12;err=0.00014561
SNW_DS_MAIN_VEC tax and spend;it=13;err=0.00012201
SNW_DS_MAIN_VEC tax and spend;it=14;err=0.00010224
SNW_DS_MAIN_VEC tax and spend;it=15;err=8.567e-05
SNW_DS_MAIN_VEC: Number of a2-adjustments (for taxation) used to balance the government budget= 15
SNW_DS_MAIN_VEC: Old and updated value of a2=1.5286      1.5353
SNW_DS_MAIN_VEC: Aggregates: Cons., Gov. cons., Save, Assets, Income, Bequests 48.78871      11.3586
SNW_DS_MAIN_VEC: Resource constraint: C_t+A_{t+1}+G_t=A_t+Y_t 258.0346      258.0206
Completed SNW_DS_MAIN_VEC;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=default_tes
pos = 19 ; key = mp_controls
    Map with properties:
```

```

        Count: 37
        KeyType: char
        ValueType: any
```

```

pos = 20 ; key = mp_params
    Map with properties:
```

```

        Count: 52
        KeyType: char
        ValueType: any
```

```
-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
CONTAINER NAME: mp_dsvfi_results ND Array (Matrix etc)
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

	i	idx	ndim	numel	rowN	colN	sum	mean
	--	---	----	-----	---	-----	-----	-----
SS_ss	1	11	6	7.1754e+07	83	8.645e+05	8.3556e+06	0.116
a_ss	2	16	6	7.1754e+07	83	8.645e+05	2.4595e+09	34.2
ap_ss	3	17	6	7.1754e+07	83	8.645e+05	2.3245e+09	32.3
cons_ss	4	18	6	7.1754e+07	83	8.645e+05	3.5119e+08	4.89

n_ss	5	21	6	7.1754e+07	83	8.645e+05	2.5114e+08	3
tax_ss	6	22	6	7.1754e+07	83	8.645e+05	6.6049e+07	0.92
y_all_ss	7	23	6	7.1754e+07	83	8.645e+05	2.8219e+08	3.93
y_head_earn_ss	8	24	6	7.1754e+07	83	8.645e+05	1.078e+08	1.50
y_head_inc_ss	9	25	6	7.1754e+07	83	8.645e+05	2.1454e+08	2.
y_spouse_inc_ss	10	26	6	7.1754e+07	83	8.645e+05	6.7646e+07	0.942
yshr_SS_ss	11	27	6	7.1754e+07	83	8.645e+05	1.0586e+07	0.147
yshr_interest_ss	12	28	6	7.1754e+07	83	8.645e+05	3.0079e+07	0.41
yshr_nttxss_ss	13	29	6	7.1754e+07	83	8.645e+05	3.7387e+06	0.0521
yshr_tax_ss	14	30	6	7.1754e+07	83	8.645e+05	1.4324e+07	0.199
yshr_wage_ss	15	31	6	7.1754e+07	83	8.645e+05	3.1088e+07	0.433
<hr/>								
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx								
CONTAINER NAME: mp_dsvfi_results Scalars								
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx								
i	idx		value					
--	--		-----					
A_agg	1	1	193.39					
A_agg_perhh	2	2	4.2232					
Aprime_agg	3	3	197.89					
Aprime_agg_perhh	4	4	4.3213					
Bequests_aux	5	5	2.5593					
Bequests_aux_perhh	6	6	0.055887					
C_agg	7	7	48.789					
C_agg_perhh	8	8	1.0654					
SS_spend	9	9	2.3908					
SS_spend_perhh	10	10	0.052208					
Tax_revenues	11	12	13.735					
Tax_revenues_perhh	12	13	0.29994					
Y_inc_agg	13	14	64.627					
Y_inc_agg_perhh	14	15	1.4113					
<hr/>								
xxx tb_outcomes: all stats xxx								
OriginalVariableNames	a_ss	ap_ss	cons_ss	n_ss	y_all			
-----	-----	-----	-----	-----	-----			
{'mean'}	4.2232	4.3213	1.0654	2.3554	1.4635			
{'unweighted_sum'}	2228	8.7064e+08	8.2948e+07	21	1.3652e+08			
{'sd'}	6.7417	6.779	0.6899	1.4375	1.4563			
{'coefofvar'}	1.5964	1.5687	0.64754	0.61029	0.99508			
{'gini'}	0.68027	0.68124	0.33738	0.3128	0.44246			
{'min'}	0	0	0.036717	1	0.038108			
{'max'}	135	163.7	141.66	6	50.873			
{'pYiso'}	0.12293	0.10299	0	0	0			
{'pYls0'}	0	0	0	0	0			
{'pYgro'}	0.87707	0.89701	1	1	1			
{'pYisMINY'}	0.12293	0.10299	6.7731e-07	0.36005	6.7731e-07			
{'pYisMAXY'}	6.0111e-06	1.6708e-12	0	0.041101	1.6708e-12			
{'p0_01'}	0	0	0.067181	1	0.07102			
{'p0_1'}	0	0	0.10544	1	0.11346			
{'p1'}	0	0	0.18623	1	0.20359			
{'p5'}	0	0	0.27747	1	0.28173			
{'p10'}	0	0	0.36103	1	0.35688			
{'p20'}	0.064373	0.068222	0.49773	1	0.50299			
{'p25'}	0.11124	0.17983	0.56413	1	0.57911			
{'p30'}	0.26367	0.37542	0.63091	1	0.65753			

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

{'p40'}	}	0.68544	0.84816	0.77012	2	0.83048
{'p50'}	}	1.4131	1.5883	0.91942	2	1.0325
{'p60'}	}	2.5301	2.7569	1.0845	2	1.2817
{'p70'}	}	4.1199	4.4885	1.2781	3	1.613
{'p75'}	}	5.4836	5.7144	1.3935	3	1.8306
{'p80'}	}	7.1191	7.2197	1.5293	4	2.1079
{'p90'}	}	12.56	12.096	1.9344	5	3.0419
{'p95'}	}	16.875	17.457	2.3404	5	4.0251
{'p99'}	}	30.548	31.377	3.384	6	6.8588
{'p99_9'}	}	56.953	56.953	5.2437	6	14.778
{'p99_99'}	}	90.439	88.534	7.4817	6	20.971
{'fl_cov_a_ss'}	}	45.451	45.439	3.3942	-1.4049	4.4679
{'fl_cor_a_ss'}	}	1	0.99423	0.72975	-0.14496	0.45507
{'fl_cov_ap_ss'}	}	45.439	45.955	3.4956	-1.3685	5.3067
{'fl_cor_ap_ss'}	}	0.99423	1	0.74743	-0.14043	0.53754
{'fl_cov_cons_ss'}	}	3.3942	3.4956	0.47596	0.23909	0.76142
{'fl_cor_cons_ss'}	}	0.72975	0.74743	1	0.24109	0.75787
{'fl_cov_n_ss'}	}	-1.4049	-1.3685	0.23909	2.0664	0.35987
{'fl_cor_n_ss'}	}	-0.14496	-0.14043	0.24109	1	0.17191
{'fl_cov_y_all'}	}	4.4679	5.3067	0.76142	0.35987	2.1208
{'fl_cor_y_all'}	}	0.45507	0.53754	0.75787	0.17191	1
{'fl_cov_y_head_inc'}	}	3.8282	4.1045	0.55948	0.092667	1.1039
{'fl_cor_y_head_inc'}	}	0.56819	0.60585	0.81146	0.064504	0.75851
{'fl_cov_y_head_earn'}	}	1.8477	2.1508	0.42576	0.19287	0.96246
{'fl_cor_y_head_earn'}	}	0.29785	0.34482	0.67071	0.14582	0.71827
{'fl_cov_y_spouse_inc'}	}	0.63967	1.2022	0.20194	0.2672	1.0169
{'fl_cor_y_spouse_inc'}	}	0.09937	0.18573	0.30656	0.19467	0.73129
{'fl_cov_yshr_interest'}	}	0.76424	0.71927	0.037996	-0.066731	-0.0094215
{'fl_cor_yshr_interest'}	}	0.67572	0.63246	0.3283	-0.27671	-0.038564
{'fl_cov_yshr_wage'}	}	-0.77528	-0.68855	-0.0042957	0.17055	0.10767
{'fl_cor_yshr_wage'}	}	-0.34062	-0.30085	-0.018443	0.35142	0.21899
{'fl_cov_yshr_SS'}	}	0.011037	-0.030725	-0.033701	-0.10382	-0.09825
{'fl_cor_yshr_SS'}	}	0.0069239	-0.019169	-0.2066	-0.30546	-0.28534
{'fl_cov_yshr_tax'}	}	0.098159	0.10896	0.018583	0.01337	0.038535
{'fl_cor_yshr_tax'}	}	0.41485	0.45797	0.76748	0.26501	0.75395
{'fl_cov_yshr_nttxss'}	}	0.087122	0.13969	0.052284	0.11719	0.13679
{'fl_cor_yshr_nttxss'}	}	0.050539	0.080586	0.29639	0.31882	0.36733
{'fracByP0_01'}	}	0	0	5.5188e-06	0.15286	4.2239e-06
{'fracByP0_1'}	}	0	0	8.2593e-05	0.15286	6.444e-05
{'fracByP1'}	}	0	0	0.0013857	0.15286	0.0010994
{'fracByP5'}	}	0	0	0.010292	0.15286	0.0079949
{'fracByP10'}	}	0	0	0.025341	0.15286	0.018888
{'fracByP20'}	}	0.00074832	0.00060951	0.065753	0.15286	0.048269
{'fracByP25'}	}	0.0014123	0.0020285	0.090679	0.15286	0.066791
{'fracByP30'}	}	0.0041719	0.0051595	0.11872	0.15286	0.087944
{'fracByP40'}	}	0.016751	0.01877	0.1844	0.40183	0.13867
{'fracByP50'}	}	0.045326	0.046338	0.26358	0.40183	0.20207
{'fracByP60'}	}	0.095502	0.095716	0.3575	0.40183	0.28072
{'fracByP70'}	}	0.17466	0.17847	0.46813	0.56321	0.37901
{'fracByP75'}	}	0.24517	0.23715	0.53078	0.56321	0.43771
{'fracByP80'}	}	0.32852	0.31134	0.59927	0.75407	0.50477
{'fracByP90'}	}	0.56651	0.52814	0.75975	0.8953	0.67658
{'fracByP95'}	}	0.70071	0.6954	0.85893	0.8953	0.79526
{'fracByP99'}	}	0.90524	0.90259	0.96084	1	0.93132
{'fracByP99_9'}	}	0.98567	0.98372	0.99419	1	0.98801
{'fracByP99_99'}	}	0.99808	0.9976	0.99922	1	0.99841

% Get Matrixes

```

cl_st_recompute_list = {'a', 'ar_z_ctr_amz', ...
    'inc', 'inc_unemp', 'spouse_inc', 'spouse_inc_unemp', 'ref_earn_wageind_grid',...
    'ap_idx_lower_ss', 'ap_idx_higher_ss', 'ap_idx_lower_weight_ss'};
mp_controls('bl_print_recompute_verbose') = false;
[mp_recompute_res] = snw_hh_recompute(mp_params, mp_controls, cl_st_recompute_list, ap_ss, Phi_tr);

Wage quintile cutoffs=0.47017      0.71433      1.0293      1.5654
Completed SNW_HH_PRECOMPUTE;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=default_t

```

12.1.2 Solve for 2019 Evuvw With 0 and 1 Checks

```

% Call Function
welf_checks = 0;
[ev19_jaeemk_check0, ec19_jaeemk_check0, ev20_jaeemk_check0, ec20_jaeemk_check0] = snw_evuvw19_jaeemk...
    welf_checks, st_solu_type, mp_params, mp_controls, ...
    V_ss, ap_ss, cons_ss, V_unemp, cons_unemp, mp_recompute_res);

Completed SNW_A4CHK_WRK_BISEC_VEC;welf_checks=0;TR=0.0017225;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=default_moredense_a65zh133zs5_e2m2
Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=0;TR=0.0017225;xi=0.5;b=1;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=default_moredense_a65zh133zs5_e2m2
Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=default_moredense_a65zh133zs5_e2m2;SNW_MP_FOC=default_moredense_a65zh133zs5_e2m2
Completed SNW_EVUVW19_JAEEMK_FOC;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=default_moredense_a65zh133zs5_e2m2;SNW_MP_FOC=default_moredense_a65zh133zs5_e2m2

% Call Function
welf_checks = 1;
[ev19_jaeemk_check2, ec19_jaeemk_check2, ev20_jaeemk_check2, ec20_jaeemk_check2] = snw_evuvw19_jaeemk...
    welf_checks, st_solu_type, mp_params, mp_controls, ...
    V_ss, ap_ss, cons_ss, V_unemp, cons_unemp, mp_recompute_res);

Completed SNW_A4CHK_WRK_BISEC_VEC;welf_checks=1;TR=0.0017225;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=default_moredense_a65zh133zs5_e2m2
Completed SNW_A4CHK_UNEMP_BISEC_VEC;welf_checks=1;TR=0.0017225;xi=0.5;b=1;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=default_moredense_a65zh133zs5_e2m2
Completed SNW_EVUVW20_JAEEMK;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=default_moredense_a65zh133zs5_e2m2;SNW_MP_FOC=default_moredense_a65zh133zs5_e2m2
Completed SNW_EVUVW19_JAEEMK_FOC;SNW_MP_PARAM=default_moredense_a65zh133zs5_e2m2;SNW_MP_CONTROL=default_moredense_a65zh133zs5_e2m2;SNW_MP_FOC=default_moredense_a65zh133zs5_e2m2

```

Differences between Checks in Expected Value and Expected Consumption

```

mn_V_U_gain_check = ev19_jaeemk_check2 - ev19_jaeemk_check0;
mn_MPC_C_gain_share_check = (ec19_jaeemk_check2 - ec19_jaeemk_check0)./(welf_checks*mp_params('TR'))

```

12.1.3 Additional Variables

Create additional Staet-Spac Arrays

```

% (n_jgrid,n_agrid,n_etagrid,n_educgrid,n_marriedgrid,n_kidsgrid);
% Children Array
ar_kids = (1:mp_params('n_kidsgrid')) - 1;
mn_kids = zeros(1,1,1,1,length(ar_kids));
mn_kids(1,1,1,1,1,:) = ar_kids;
kids_ss = repmat(mn_kids, [mp_params('n_jgrid'), mp_params('n_agrid'), mp_params('n_etagrid'), ...
    mp_params('n_educgrid'), mp_params('n_marriedgrid'), 1]);
% Marital Status Arrays
ar_marital = (1:mp_params('n_marriedgrid')) - 1;
mn_marital = zeros(1,1,1,1,length(ar_marital),1);
mn_marital(1,1,1,1,:,:) = ar_marital;
marital_ss = repmat(mn_marital, [mp_params('n_jgrid'), mp_params('n_agrid'), mp_params('n_etagrid'), ...
    mp_params('n_educgrid'), 1, mp_params('n_kidsgrid')]);
% Educational Status Arrays
ar_educ = (1:mp_params('n_educgrid')) - 1;
mn_educ = zeros(1,1,1,length(ar_educ),1,1);
mn_educ(1,1,1,:,:1) = ar_educ;
educ_ss = repmat(mn_educ, [mp_params('n_jgrid'), mp_params('n_agrid'), mp_params('n_etagrid'), ...
    mp_params('n_educgrid'), 1, mp_params('n_kidsgrid')]);

```

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

```

1, mp_params('n_marriedgrid'), mp_params('n_kidsgrid'))];
% Age Array
ar_age = (1:mp_params('n_jgrid')) + 18;
mn_age = zeros(length(ar_age),1,1,1,1,1);
mn_age(:,1,1,1,1,1) = ar_age;
age_ss = repmat(mn_age, [1, mp_params('n_agrid'), mp_params('n_etagrid'), ...
    mp_params('n_educgrid'), mp_params('n_marriedgrid'), mp_params('n_kidsgrid'))];

```

12.1.4 Adjust to Probability Mass Function

```
Phi_true_1 = Phi_true./sum(Phi_true,'all');
```

12.1.5 Age Bounds

```

% 1 = 18
min_age = 1

min_age = 1

% retirement, 46+18=64, the year prior to retirement year.
max_age = 46;
```

12.1.6 Scale Statistics to Thousands of Dollars

```

a_ss = mp_dsvfi_results('a_ss')*58.056;
ap_ss = mp_dsvfi_results('ap_ss')*58.056;
c_ss = mp_dsvfi_results('cons_ss')*58.056;
n_ss = mp_dsvfi_results('n_ss');
% household head + spousal (realized) income
y_all = mp_dsvfi_results('y_all_ss')*58.056;
y_head_inc = mp_dsvfi_results('y_head_inc_ss')*58.056;
y_spouse_inc = mp_dsvfi_results('y_spouse_inc_ss')*58.056;

yshr_wage = mp_dsvfi_results('yshr_wage_ss');
yshr_SS = mp_dsvfi_results('yshr_SS_ss');
yshr_nttxss = mp_dsvfi_results('yshr_nttxss_ss');
```

12.1.7 Distributional Statistics Overall All Ages

```

% construct input data
marital_grp = marital_ss(min_age:82, :, :, :, :, :, :);
y_all_grp = y_all(min_age:82, :, :, :, :, :, :);
age_ss_grp = age_ss(min_age:82, :, :, :, :, :, :);
educ_ss_grp = educ_ss(min_age:82, :, :, :, :, :, :);
a_ss_grp = a_ss(min_age:82, :, :, :, :, :, :);
ap_ss_grp = ap_ss(min_age:82, :, :, :, :, :, :);
mn_MPC_C_gain_share_check_grp = mn_MPC_C_gain_share_check(min_age:82, :, :, :, :, :, :, :);
Phi_true_grp = Phi_true_1(min_age:82, :, :, :, :, :, :);
c_ss_grp = c_ss(min_age:82, :, :, :, :, :, :);
y_head_inc_grp = y_head_inc(min_age:82, :, :, :, :, :, :);
y_spouse_inc_grp = y_spouse_inc(min_age:82, :, :, :, :, :, :);
yshr_nttxss_grp = yshr_nttxss(min_age:82, :, :, :, :, :, :);

mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
mp_cl_ar_xyz_of_s('married') = {marital_grp(:,), zeros(1)};
mp_cl_ar_xyz_of_s('y_all') = {y_all_grp(:,), zeros(1)};
mp_cl_ar_xyz_of_s('age_ss') = {age_ss_grp(:,), zeros(1)};
mp_cl_ar_xyz_of_s('educ_ss') = {educ_ss_grp(:,), zeros(1)};
mp_cl_ar_xyz_of_s('a_ss') = {a_ss_grp(:,), zeros(1)};
```

```

mp_cl_ar_xyz_of_s('ap_ss') = {ap_ss_grp(:, zeros(1)};
mp_cl_ar_xyz_of_s('MPC') = {mn_MPC_C_gain_share_check_grp(:, zeros(1)};
mp_cl_ar_xyz_of_s('Mass') = {Phi_true_grp(:, zeros(1)};
mp_cl_ar_xyz_of_s('c_ss') = {c_ss_grp(:, zeros(1)};
mp_cl_ar_xyz_of_s('y_head_inc') = {y_head_inc_grp(:, zeros(1)};
mp_cl_ar_xyz_of_s('y_spouse') = {y_spouse_inc_grp(:, zeros(1)};
mp_cl_ar_xyz_of_s('yshr_nttxss') = {yshr_nttxss_grp(:, zeros(1};

mp_cl_ar_xyz_of_s('ar_st_y_name') = ["married", "y_all", "age_ss", "educ_ss", "a_ss", "ap_ss", "MPC"]

% controls
mp_support = containers.Map('KeyType', 'char', 'ValueType', 'any');
mp_support('ar_fl_percentiles') = [0.01 10 25 50 75 90 99.99];
mp_support('bl_display_final') = true;
mp_support('bl_display_detail') = false;
mp_support('bl_display_drvm2outcomes') = false;
mp_support('bl_display_drvstats') = false;
mp_support('bl_display_drvm2covcor') = false;

% Call Function
mp_cl_mt_xyz_of_s = ff_simu_stats(Phi_true_grp(:)/sum(Phi_true_grp,'all'), mp_cl_ar_xyz_of_s, mp_sup

xxx tb_outcomes: all stats xxx


| OriginalVariableNames | married    | y_all      | age_ss     | educ_ss    | a_ss       |
|-----------------------|------------|------------|------------|------------|------------|
| {'mean'}              | 0.47501    | 84.974     | 47.129     | 0.303      | 245.22     |
| {'unweighted_sum'}    | 1          | 7.9255e+09 | 4879       | 1          | 1.2935e+05 |
| {'sd'}                | 0.49938    | 84.549     | 19.231     | 0.45956    | 391.42     |
| {'coefofvar'}         | 1.0513     | 0.995      | 0.40805    | 1.5167     | 1.5962     |
| {'gini'}              | 0.36718    | 0.44243    | 0.23101    | 0.61588    | 0.68023    |
| {'min'}               | 0          | 2.2124     | 19         | 0          | 0          |
| {'max'}               | 1          | 2953.5     | 100        | 1          | 7837.6     |
| {'pYiso'}             | 0.52499    | 0          | 0          | 0.697      | 0.12285    |
| {'pYls0'}             | 0          | 0          | 0          | 0          | 0          |
| {'pYgro'}             | 0.47501    | 1          | 1          | 0.303      | 0.87715    |
| {'pYisMINY'}          | 0.52499    | 6.774e-07  | 0.02184    | 0.697      | 0.12285    |
| {'pYisMAXY'}          | 0.47501    | 1.671e-12  | 0.00020326 | 0.303      | 6.0119e-06 |
| {'p0_01'}             | 0          | 4.1232     | 19         | 0          | 0          |
| {'p10'}               | 0          | 20.726     | 23         | 0          | 0          |
| {'p25'}               | 0          | 33.631     | 31         | 0          | 6.458      |
| {'p50'}               | 0          | 59.948     | 45         | 0          | 82.04      |
| {'p75'}               | 1          | 106.28     | 62         | 1          | 318.35     |
| {'p90'}               | 1          | 176.61     | 75         | 1          | 729.18     |
| {'p99_99'}            | 1          | 1217.5     | 100        | 1          | 5250.6     |
| {'fl_cov_married'}    | 0.24938    | 12.618     | 2.9987e-13 | 0.026842   | 31.201     |
| {'fl_cor_married'}    | 1          | 0.29884    | 3.1225e-14 | 0.11697    | 0.15962    |
| {'fl_cov_y_all'}      | 12.618     | 7148.6     | -105.85    | 6.7259     | 15059      |
| {'fl_cor_y_all'}      | 0.29884    | 1          | -0.065099  | 0.1731     | 0.45504    |
| {'fl_cov_age_ss'}     | 2.9987e-13 | -105.85    | 369.84     | 5.7371e-13 | 2902       |
| {'fl_cor_age_ss'}     | 3.1225e-14 | -0.065099  | 1          | 6.4916e-14 | 0.38553    |
| {'fl_cov_educ_ss'}    | 0.026842   | 6.7259     | 5.7371e-13 | 0.21119    | 20.13      |
| {'fl_cor_educ_ss'}    | 0.11697    | 0.1731     | 6.4916e-14 | 1          | 0.11191    |
| {'fl_cov_a_ss'}       | 31.201     | 15059      | 2902       | 20.13      | 1.5321e+05 |
| {'fl_cor_a_ss'}       | 0.15962    | 0.45504    | 0.38553    | 0.11191    | 1          |
| {'fl_cov_ap_ss'}      | 31.93      | 17886      | 2762.7     | 20.615     | 1.5316e+05 |
| {'fl_cor_ap_ss'}      | 0.16246    | 0.53751    | 0.36501    | 0.11398    | 0.99423    |
| {'fl_cov_MPC'}        | -0.016733  | -6.6507    | -1.2778    | 0.0049583  | -30.154    |


```

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

{'fl_cor_MPC'}	}	-0.13011	-0.30544	-0.258	0.041894	-0.29913
{'fl_cov_Mass'}	}	-5.1035e-07	-7.3196e-05	-2.691e-05	-2.0525e-07	-0.00031586
{'fl_cov_Mass'}	}	-0.19258	-0.16313	-0.26368	-0.084158	-0.15206
{'fl_cov_c_ss'}	}	8.8909	2566.3	57.161	4.6211	11440
{'fl_cov_c_ss'}	}	0.44452	0.75784	0.074211	0.25106	0.72974
{'fl_cov_y_head_inc'}	}	1.6909	3720.9	-73.542	4.2898	12903
{'fl_cov_y_head_inc'}		0.058359	0.75849	-0.065909	0.16088	0.56816
{'fl_cov_y_spouse'}		10.927	3427.7	-32.308	2.436	2155.8
{'fl_cov_y_spouse'}		0.3947	0.73129	-0.030304	0.095619	0.09935
{'fl_cov_yshr_nttxss'}		0.022689	7.935	-3.2573	0.0058708	5.0323
{'fl_cov_yshr_nttxss'}		0.1778	0.36727	-0.66283	0.049993	0.050313
{'fracByP0_01'}		0	4.224e-06	0.0088049	0	0
{'fracByP10'}		0	0.018881	0.047593	0	0
{'fracByP25'}		0	0.066793	0.14054	0	0.0014119
{'fracByP50'}		0	0.20209	0.34194	0	0.045325
{'fracByP75'}		1	0.43774	0.62344	1	0.24517
{'fracByP90'}		1	0.6766	0.82958	1	0.56651
{'fracByP99_99'}		1	0.99841	1	1	0.99808

```
tb_dist_stats_all = mp_cl_mt_xyz_of_s('tb_outcomes');
```

12.1.8 Distributional Statistics Overall 18 to 64

Statistics overall distributionally for 18 to 64 year olds.

```
% construct input data
marital_grp = marital_ss(min_age:max_age, :, :, :, :, :, :);
y_all_grp = y_all(min_age:max_age, :, :, :, :, :, :);
age_ss_grp = age_ss(min_age:max_age, :, :, :, :, :, :);
educ_ss_grp = educ_ss(min_age:max_age, :, :, :, :, :, :);
a_ss_grp = a_ss(min_age:max_age, :, :, :, :, :, :);
ap_ss_grp = ap_ss(min_age:max_age, :, :, :, :, :, :);
mn_MPC_C_gain_share_check_grp = mn_MPC_C_gain_share_check(min_age:max_age, :, :, :, :, :, :);
Phi_true_grp = Phi_true_1(min_age:max_age, :, :, :, :, :, :);
c_ss_grp = c_ss(min_age:max_age, :, :, :, :, :, :);
y_head_inc_grp = y_head_inc(min_age:max_age, :, :, :, :, :, :);
y_spouse_inc_grp = y_spouse_inc(min_age:max_age, :, :, :, :, :, :);
yshr_nttxss_grp = yshr_nttxss(min_age:max_age, :, :, :, :, :, :);

mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
mp_cl_ar_xyz_of_s('married') = {marital_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('y_all') = {y_all_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('age_ss') = {age_ss_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('educ_ss') = {educ_ss_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('a_ss') = {a_ss_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('ap_ss') = {ap_ss_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('MPC') = {mn_MPC_C_gain_share_check_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('Mass') = {Phi_true_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('c_ss') = {c_ss_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('y_head_inc') = {y_head_inc_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('y_spouse') = {y_spouse_inc_grp(:), zeros(1)};
mp_cl_ar_xyz_of_s('yshr_nttxss') = {yshr_nttxss_grp(:), zeros(1});

mp_cl_ar_xyz_of_s('ar_st_y_name') = ["married", "y_all", "age_ss", "educ_ss", "a_ss", "ap_ss", "MPC"]

% controls
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('ar_fl_percentiles') = [0.01 10 25 50 75 90 99.99];
mp_support('bl_display_final') = true;
```

```

mp_support('bl_display_detail') = false;
mp_support('bl_display_drvm2outcomes') = false;
mp_support('bl_display_drvstats') = false;
mp_support('bl_display_drvm2covcor') = false;

% Call Function
mp_cl_mt_xyz_of_s = ff_simu_stats(Phi_true_grp(:)/sum(Phi_true_grp,'all'), mp_cl_ar_xyz_of_s, mp_sup

xxx tb_outcomes: all stats xxx
    OriginalVariableNames      married      y_all      age_ss      educ_ss      a_ss
    -----      -----      -----      -----      -----      -----
{'mean'}      }      0.47501      95.246      39.372      0.303      194.
{'unweighted_sum'}      }      1      7.7487e+09      1909      1      1.2935e+0
{'sd'}      }      0.49938      89.631      13.105      0.45956      344.
{'coefofvar'}      }      1.0513      0.94104      0.33285      1.5167      1.771
{'gini'}      }      0.36718      0.42428      0.18859      0.61588      0.7157
{'min'}      }      0      2.2124      19      0
{'max'}      }      1      2953.5      64      1      7837.
{'pYis0'}      }      0.52499      0      0      0.697      0.1462
{'pYls0'}      }      0      0      0      0
{'pYgr0'}      }      0.47501      1      1      0.303      0.8537
{'pYisMINY'}      }      0.52499      8.6135e-07      0.027771      0.697      0.1462
{'pYisMAXY'}      }      0.47501      2.1248e-12      0.015675      0.303      5.4766e-0
{'p0_01'}      }      0      3.9581      19      0
{'p10'}      }      0      25.069      22      0
{'p25'}      }      0      40.654      28      0      3.737
{'p50'}      }      0      69.57      38      0      51.66
{'p75'}      }      1      119.76      50      1      239.1
{'p90'}      }      1      192.9      58      1      588.4
{'p99_99'}      }      1      1249.3      64      1      4707.
{'fl_cov_married'}      }      0.24938      13.756      2.335e-13      0.026842      25.2
{'fl_cor_married'}      }      1      0.30733      3.5679e-14      0.11697      0.1468
{'fl_cov_y_all'}      }      13.756      8033.6      270.03      7.5617      1785
{'fl_cor_y_all'}      }      0.30733      1      0.22988      0.18358      0.5781
{'fl_cov_age_ss'}      }      2.335e-13      270.03      171.75      4.3386e-15      2241.
{'fl_cor_age_ss'}      }      3.5679e-14      0.22988      1      7.204e-16      0.4964
{'fl_cov_educ_ss'}      }      0.026842      7.5617      4.3386e-15      0.21119      15.47
{'fl_cor_educ_ss'}      }      0.11697      0.18358      7.204e-16      1      0.09776
{'fl_cov_a_ss'}      }      25.27      17852      2241.5      15.478      1.1868e+0
{'fl_cor_a_ss'}      }      0.14689      0.57814      0.49648      0.097766
{'fl_cov_ap_ss'}      }      26.783      20993      2328.9      16.562      1.2238e+0
{'fl_cor_ap_ss'}      }      0.15001      0.65507      0.49704      0.1008      0.9935
{'fl_cov_MPC'}      }      -0.017248      -8.3845      -1.4685      0.0073384      -27.85
{'fl_cor_MPC'}      }      -0.12735      -0.34491      -0.41317      0.058877      -0.2981
{'fl_cov_Mass'}      }      -6.2681e-07      -0.00010581      -2.2759e-05      -2.2235e-07      -0.0003165
{'fl_cor_Mass'}      }      -0.21171      -0.19912      -0.29292      -0.081609      -0.15
{'fl_cov_c_ss'}      }      8.9405      2911.4      117.7      4.6429      9782.
{'fl_cor_c_ss'}      }      0.44676      0.81058      0.22412      0.25211      0.7086
{'fl_cov_y_head_inc'}      }      1.5449      4083.5      215.29      4.8213      1513
{'fl_cor_y_head_inc'}      }      0.050457      0.74307      0.26794      0.17111      0.7164
{'fl_cov_y_spouse'}      }      12.211      3950.1      54.733      2.7405      2719.
{'fl_cor_y_spouse'}      }      0.40608      0.7319      0.069359      0.099033      0.131
{'fl_cov_yshr_nttxss'}      }      0.0064334      2.2345      0.12412      0.0029398      5.703
{'fl_cor_yshr_nttxss'}      }      0.38567      0.74633      0.28352      0.1915      0.4956
{'fracByP0_01'}      }      0      3.6432e-06      0.013402      0
{'fracByP10'}      }      0      0.018969      0.056893      0
{'fracByP25'}      }      0      0.070975      0.15748      0      0.001135

```

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

{'fracByP50'}	}	0	0.21374	0.35932	0	0.03404
{'fracByP75'}	}	1	0.45357	0.64274	1	0.2134
{'fracByP90'}	}	1	0.69054	0.84608	1	0.5149
{'fracByP99_99'}	}	1	0.99855	1	1	0.9971

```
tb_dist_stats_all_18to64 = mp_cl_mt_xyz_of_s('tb_outcomes');
```

12.1.9 Distributional Statistics By Kids Count

Various statistics, including MPC (of the first check) by Children Count

```
it_row_ctr = 0;
for it_ctr=1:mp_params('n_kidsgrid')
    display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);
    display(['kids =' num2str(ar_kids(it_ctr))]);
    display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);

    % construct input data
    marital_grp = marital_ss(min_age:max_age, :, :, :, :, it_ctr);
    y_all_grp = y_all(min_age:max_age, :, :, :, :, it_ctr);
    age_ss_grp = age_ss(min_age:max_age, :, :, :, :, it_ctr);
    educ_ss_grp = educ_ss(min_age:max_age, :, :, :, :, it_ctr);
    a_ss_grp = a_ss(min_age:max_age, :, :, :, :, it_ctr);
    ap_ss_grp = ap_ss(min_age:max_age, :, :, :, :, it_ctr);
    mn_MPC_C_gain_share_check_grp = mn_MPC_C_gain_share_check(min_age:max_age, :, :, :, :, it_ctr);
    Phi_true_grp = Phi_true_1(min_age:max_age, :, :, :, :, it_ctr);
    c_ss_grp = c_ss(min_age:max_age, :, :, :, :, it_ctr);
    y_head_inc_grp = y_head_inc(min_age:max_age, :, :, :, :, it_ctr);
    y_spouse_inc_grp = y_spouse_inc(min_age:max_age, :, :, :, :, it_ctr);
    yshr_nttxss_grp = yshr_nttxss(min_age:max_age, :, :, :, :, it_ctr);

    mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
    mp_cl_ar_xyz_of_s('married') = {marital_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('y_all') = {y_all_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('age_ss') = {age_ss_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('educ_ss') = {educ_ss_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('a_ss') = {a_ss_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('ap_ss') = {ap_ss_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('MPC') = {mn_MPC_C_gain_share_check_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('Mass') = {Phi_true_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('c_ss') = {c_ss_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('y_head_inc') = {y_head_inc_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('y_spouse') = {y_spouse_inc_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('yshr_nttxss') = {yshr_nttxss_grp(:), zeros(1});

    mp_cl_ar_xyz_of_s('ar_st_y_name') = ["married", "y_all", "age_ss", "educ_ss", "a_ss", "ap_ss", "MPC", "Mass", "c_ss", "y_head_inc", "y_spouse", "yshr_nttxss"];

    % controls
    mp_support = containers.Map('KeyType','char', 'ValueType','any');
    mp_support('ar_fl_percentiles') = [0.01 10 25 50 75 90 99.99];
    mp_support('bl_display_final') = true;
    mp_support('bl_display_detail') = false;
    mp_support('bl_display_drvm2outcomes') = false;
    mp_support('bl_display_drvstats') = false;
    mp_support('bl_display_drvm2covcor') = false;

    % Call Function
    mp_cl_mt_xyz_of_s = ff_simu_stats(Phi_true_grp(:)/sum(Phi_true_grp,'all'), mp_cl_ar_xyz_of_s, mp_params('n_kidsgrid'));
```

```

it_kids = ar_kids(it_ctr);

tb_dist_stats = mp_cl_mt_xyz_of_s('tb_outcomes');

fl_married_mean = tb_dist_stats{"married", "mean"};

fl_age_mean = tb_dist_stats{"age_ss", "mean"};
fl_age_p50 = tb_dist_stats{"age_ss", "p50"};

fl_educ_mean = tb_dist_stats{"educ_ss", "mean"};

fl_a_mean = tb_dist_stats{"a_ss", "mean"};
fl_a_p50 = tb_dist_stats{"a_ss", "p50"};

fl_ap_mean = tb_dist_stats{"ap_ss", "mean"};
fl_ap_p50 = tb_dist_stats{"ap_ss", "p50"};

fl_y_all_mean = tb_dist_stats{"y_all", "mean"};
fl_y_all_p50 = tb_dist_stats{"y_all", "p50"};

fl_mpc_mean = tb_dist_stats{"MPC", "mean"};
fl_mpc_p50 = tb_dist_stats{"MPC", "p50"};

fl_mass = tb_dist_stats{"Mass", "unweighted_sum"};

fl_c_ss_mean = tb_dist_stats{"c_ss", "mean"};
fl_c_ss_p50 = tb_dist_stats{"c_ss", "p50"};

fl_y_head_inc_mean = tb_dist_stats{"y_head_inc", "mean"};
fl_y_spouse_mean = tb_dist_stats{"y_spouse", "mean"};

ar_store_stats = [it_kids, fl_married_mean, ...
    fl_age_mean, fl_age_p50, fl_educ_mean, ...
    fl_a_mean, fl_a_p50, fl_ap_mean, fl_ap_p50, ...
    fl_y_all_mean, fl_y_all_p50, ...
    fl_mpc_mean, fl_mpc_p50, ...
    fl_mass, ...
    fl_c_ss_mean, fl_c_ss_p50, ...
    fl_y_head_inc_mean, fl_y_spouse_mean];

it_row_ctr = it_row_ctr + 1;

if (it_row_ctr>1)
    mt_store_stats_by_k = [mt_store_stats_by_k;ar_store_stats];
else
    mt_store_stats_by_k = [ar_store_stats];
end
end

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
kids =0
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxx tb_outcomes: all stats xxx
    OriginalVariableNames      married      y_all      age_ss      educ_ss      a_ss
    -----      -----      -----      -----      -----
{'mean'          }      0.34092      95.696      42.81      0.29837      267.84
{'unweighted_sum'}           1      1.9045e+09      1909           1      1.2935e+05

```

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

{'sd'}		0.47402	93.188	14.55	0.45754	413.66
{'coefofvar'}		1.3904	0.97379	0.33987	1.5335	1.5444
{'gini'}		0.56028	0.43559	0.18997	0.62263	0.66933
{'min'}		0	2.2124	19	0	0
{'max'}		1	2953.5	64	1	7837.6
{'pYis0'}		0.65908	0	0	0.70163	0.10437
{'pYls0'}		0	0	0	0	0
{'pYgr0'}		0.34092	1	1	0.29837	0.89563
{'pYisMINY'}		0.65908	1.2783e-06	0.038791	0.70163	0.10437
{'pYisMAXY'}		0.34092	4.4127e-12	0.029551	0.29837	1.0023e-05
{'p0_01'}		0	3.8399	19	0	0
{'p10'}		0	23.75	21	0	0
{'p25'}		0	39.249	29	0	10.255
{'p50'}		0	68.775	45	0	100.91
{'p75'}		1	119.82	56	1	363.77
{'p90'}		1	198.36	61	1	729.18
{'p99_99'}		1	1317.3	64	1	5250.6
{'fl_cov_married'}		0.22469	15.781	0.41952	0.027901	47.935
{'fl_cor_married'}		1	0.35725	0.060827	0.12864	0.24447
{'fl_cov_y_all'}		15.781	8684	314.15	6.9889	24515
{'fl_cov_y_all'}		0.35725	1	0.23169	0.16391	0.63596
{'fl_cov_age_ss'}		0.41952	314.15	211.7	-0.40705	2895.3
{'fl_cov_age_ss'}		0.060827	0.23169	1	-0.061144	0.48104
{'fl_cov_educ_ss'}		0.027901	6.9889	-0.40705	0.20934	17.081
{'fl_cov_educ_ss'}		0.12864	0.16391	-0.061144	1	0.090246
{'fl_cov_a_ss'}		47.935	24515	2895.3	17.081	1.7111e+05
{'fl_cov_a_ss'}		0.24447	0.63596	0.48104	0.090246	1
{'fl_cov_ap_ss'}		50.8	28037	2996.5	18.304	1.7633e+05
{'fl_cov_ap_ss'}		0.25021	0.70243	0.48082	0.093396	0.9952
{'fl_cov_MPC'}		-0.0040817	-5.3961	-1.3578	0.016362	-22.104
{'fl_cor_MPC'}		-0.040228	-0.27052	-0.43598	0.16707	-0.24964
{'fl_cov_Mass'}		-6.3926e-07	-0.00016151	-4.6365e-05	-2.596e-07	-0.00069632
{'fl_cor_Mass'}		-0.16893	-0.2171	-0.39918	-0.071073	-0.21087
{'fl_cov_c_ss'}		9.1002	3017.9	137.75	4.0645	13300
{'fl_cor_c_ss'}		0.48715	0.82178	0.24023	0.22541	0.81588
{'fl_cov_y_head_inc'}		2.4027	4493.9	256.65	4.356	20579
{'fl_cor_y_head_inc'}		0.078733	0.74906	0.27398	0.14788	0.77274
{'fl_cov_y_spouse'}		13.378	4190.1	57.502	2.6329	3936
{'fl_cor_y_spouse'}		0.45539	0.72551	0.063767	0.092849	0.15353
{'fl_cov_yshr_nttxss'}		0.0065704	2.3727	0.15463	0.0025679	8.1118
{'fl_cor_yshr_nttxss'}		0.40207	0.73855	0.30827	0.1628	0.56882
{'fracByP0_01'}		0	3.5026e-06	0.017216	0	0
{'fracByP10'}		0	0.017915	0.047317	0	0
{'fracByP25'}		0	0.067497	0.14043	0	0.002351
{'fracByP50'}		0	0.20658	0.35779	0	0.050647
{'fracByP75'}		1	0.44409	0.66951	1	0.26381
{'fracByP90'}		1	0.6831	0.86922	1	0.54208
{'fracByP99_99'}		1	0.99848	1	1	0.99785

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

kids =1

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxx tb_outcomes: all stats xxx

OriginalVariableNames	married	y_all	age_ss	educ_ss	a_ss
{'mean'}	0.48303	94.687	37.46	0.31392	163.0
{'unweighted_sum'}	1	1.7814e+09	1909	1	1.2935e+09

{'sd'}		0.49971	90.675	12.413	0.46408	298.8
{'coefofvar'}	}	1.0345	0.95763	0.33137	1.4784	1.833
{'gini'}	}	0.35621	0.42949	0.18779	0.59992	0.7288
{'min'}	}	0	2.2124	19	0	
{'max'}	}	1	2715.2	64	1	7837.
{'pYis0'}	}	0.51697	0	0	0.68608	0.1663
{'pYls0'}	}	0	0	0	0	
{'pYgr0'}	}	0.48303	1	1	0.31392	0.8336
{'pYisMINY'}	}	0.51697	8.7082e-07	0.032554	0.68608	0.1663
{'pYisMAXY'}	}	0.48303	2.5175e-12	0.0061116	0.31392	2.7005e-0
{'p0_01'}	}	0	3.9202	19	0	
{'p10'}	}	0	24.541	21	0	
{'p25'}	}	0	39.852	26	0	1.913
{'p50'}	}	0	68.409	37	0	39.79
{'p75'}	}	1	118.73	47	1	205.0
{'p90'}	}	1	193.79	55	1	467.1
{'p99_99'}	}	1	1212.2	64	1	4203.
{'fl_cov_married'}	}	0.24971	14.899	0.69339	0.029149	35.
{'fl_cor_married'}	}	1	0.32882	0.11178	0.12569	0.2376
{'fl_cov_y_all'}	}	14.899	8221.9	296.2	7.6969	1569
{'fl_cov_y_all'}	}	0.32882	1	0.26316	0.18291	0.5789
{'fl_cov_age_ss'}	}	0.69339	296.2	154.09	0.15644	1774.
{'fl_cov_age_ss'}	}	0.11178	0.26316	1	0.027156	0.4783
{'fl_cov_educ_ss'}	}	0.029149	7.6969	0.15644	0.21537	16.18
{'fl_cov_educ_ss'}	}	0.12569	0.18291	0.027156	1	0.1166
{'fl_cov_a_ss'}	}	35.5	15691	1774.9	16.186	8933
{'fl_cor_a_ss'}	}	0.23769	0.57898	0.47838	0.11669	
{'fl_cov_ap_ss'}	}	38.015	18932	1842.5	17.321	9190
{'fl_cor_ap_ss'}	}	0.24516	0.67285	0.47835	0.12028	0.990
{'fl_cov_MPC'}	}	-0.029849	-9.6826	-1.3781	0.0095259	-25.74
{'fl_cor_MPC'}	}	-0.21465	-0.38374	-0.39896	0.073764	-0.3095
{'fl_cov_Mass'}	}	-1.9886e-07	-6.1897e-05	-1.3655e-05	-1.6868e-07	-0.0001802
{'fl_cor_Mass'}	}	-0.14586	-0.25019	-0.40317	-0.13322	-0.2210
{'fl_cov_c_ss'}	}	8.8069	2953.1	157.35	4.6956	9278.
{'fl_cor_c_ss'}	}	0.43769	0.80882	0.31481	0.25128	0.7709
{'fl_cov_y_head_inc'}	}	2.065	3941	206.32	4.8059	1277
{'fl_cor_y_head_inc'}	}	0.069081	0.72657	0.27785	0.17312	0.7142
{'fl_cov_y_spouse'}	}	12.834	4280.8	89.888	2.891	2920.
{'fl_cor_y_spouse'}	}	0.4103	0.75422	0.11568	0.099519	0.156
{'fl_cov_yshr_nttxss'}		0.0069685	2.2798	0.13506	0.0030572	5.020
{'fl_cor_yshr_nttxss'}		0.41262	0.74395	0.32195	0.19492	0.4970
{'fracByP0_01'}		0	3.7241e-06	0.016512	0	
{'fracByP10'}		0	0.018692	0.055462	0	
{'fracByP25'}		0	0.069801	0.1535	0	0.0006990
{'fracByP50'}		0	0.21065	0.37529	0	0.02806
{'fracByP75'}		1	0.4485	0.6393	1	0.2118
{'fracByP90'}		1	0.68715	0.85347	1	0.4747
{'fracByP99_99'}		1	0.99858	1	1	0.9968
<hr/>						
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx						
kids =2						
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx						
xxx tb_outcomes: all stats xxx						
OriginalVariableNames		married	y_all	age_ss	educ_ss	a_ss
<hr/>						
{'mean'}		0.58436	95.419	35.807	0.30789	124.28
{'unweighted_sum'}		1	1.6966e+09	1909	1	1.2935e+05

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

{'sd'}		0.49283	87.576	10.518	0.46162	238.29
{'coefofvar'}		0.84337	0.91781	0.29375	1.4993	1.9174
{'gini'}		0.22818	0.41656	0.16465	0.60873	0.73697
{'min'}		0	2.2124	19	0	0
{'max'}		1	2551.1	64	1	7837.6
{'pYis0'}		0.41564	0	0	0.69211	0.1963
{'pYls0'}		0	0	0	0	0
{'pYgr0'}		0.58436	1	1	0.30789	0.8037
{'pYisMINY'}		0.41564	4.0938e-07	0.014906	0.69211	0.1963
{'pYisMAXY'}		0.58436	1.0736e-12	0.0019534	0.30789	9.1954e-07
{'p0_01'}		0	4.1232	19	0	0
{'p10'}		0	26.204	23	0	0
{'p25'}		0	41.871	27	0	0.80724
{'p50'}		1	70.257	35	0	29.898
{'p75'}		1	120.84	43	1	146.89
{'p90'}		1	190.32	51	1	363.77
{'p99_99'}		1	1122.5	64	1	3737.2
{'fl_cov_married'}		0.24288	12.863	0.51579	0.025827	25.491
{'fl_cor_married'}		1	0.29802	0.099501	0.11352	0.21706
{'fl_cov_y_all'}		12.863	7669.6	228.36	8.3133	11413
{'fl_cov_y_all'}		0.29802	1	0.2479	0.20564	0.54689
{'fl_cov_age_ss'}		0.51579	228.36	110.63	0.45675	1116.6
{'fl_cov_age_ss'}		0.099501	0.2479	1	0.094068	0.44549
{'fl_cov_educ_ss'}		0.025827	8.3133	0.45675	0.21309	15.009
{'fl_cov_educ_ss'}		0.11352	0.20564	0.094068	1	0.13644
{'fl_cov_a_ss'}		25.491	11413	1116.6	15.009	56783
{'fl_cov_a_ss'}		0.21706	0.54689	0.44549	0.13644	1
{'fl_cov_ap_ss'}		27.147	14327	1160.8	16.023	58304
{'fl_cov_ap_ss'}		0.22214	0.65975	0.44505	0.13997	0.9867
{'fl_cov_MPC'}		-0.055633	-12.184	-1.1929	-0.0029873	-25.836
{'fl_cov_MPC'}		-0.35573	-0.43842	-0.3574	-0.020393	-0.34167
{'fl_cov_Mass'}		-4.6541e-07	-7.3755e-05	-9.0248e-06	-2.4395e-07	-0.0001563
{'fl_cov_Mass'}		-0.32688	-0.29151	-0.29699	-0.18292	-0.22704
{'fl_cov_c_ss'}		8.1321	2864.5	129.17	5.2868	7092.7
{'fl_cov_c_ss'}		0.40653	0.80585	0.30257	0.28216	0.73333
{'fl_cov_y_head_inc'}		1.6658	3681.5	154.09	5.3399	9372.8
{'fl_cov_y_head_inc'}		0.058412	0.72644	0.25315	0.1999	0.67972
{'fl_cov_y_spouse'}		11.197	3988.2	74.272	2.9734	2040.1
{'fl_cov_y_spouse'}		0.37578	0.75322	0.11679	0.10654	0.1416
{'fl_cov_yshr_nttxss'}		0.0064177	2.1485	0.10113	0.0033688	3.5399
{'fl_cov_yshr_nttxss'}		0.39977	0.75315	0.29517	0.22404	0.45606
{'fracByP0_01'}		0	3.7828e-06	0.0079094	0	0
{'fracByP10'}		0	0.019866	0.075143	0	0
{'fracByP25'}		0	0.073594	0.17417	0	0.00024523
{'fracByP50'}		1	0.21851	0.40471	0	0.027182
{'fracByP75'}		1	0.4596	0.65078	1	0.20572
{'fracByP90'}		1	0.69638	0.85987	1	0.47333
{'fracByP99_99'}		1	0.99869	1	1	0.99695

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

kids =3

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxx tb_outcomes: all stats xxx

OriginalVariableNames	married	y_all	age_ss	educ_ss	a_ss
{'mean'}	0.69032	96.012	35.356	0.30365	101.
{'unweighted_sum'}	1	1.6091e+09	1909	1	1.2935e+09

{'sd'}		0.46236	83.53	9.1314	0.45983	196.5
{'coefofvar'}		0.66978	0.86999	0.25827	1.5143	1.940
{'gini'}		0.12198	0.40117	0.14344	0.61493	0.729
{'min'}		0	2.2124	19	0	
{'max'}		1	2381.6	64	1	7837.
{'pYis0'}		0.30968	0	0	0.69635	0.1917
{'pYls0'}		0	0	0	0	
{'pYgr0'}		0.69032	1	1	0.30365	0.8082
{'pYisMINY'}		0.30968	2.133e-07	0.007718	0.69635	0.1917
{'pYisMAXY'}		0.69032	3.4711e-13	0.00070368	0.30365	3.1947e-0
{'p0_01'}		0	4.4187	19	0	
{'p10'}		0	28.136	24	0	
{'p25'}		0	44.054	28	0	0.8072
{'p50'}		1	72.443	34	0	29.89
{'p75'}		1	122.12	42	1	100.9
{'p90'}		1	185.9	48	1	276.8
{'p99_99'}		1	1027.1	64	1	3306.
{'fl_cov_married'}		0.21378	9.9452	0.39867	0.02286	16.46
{'fl_cor_married'}		1	0.25751	0.094427	0.10752	0.1811
{'fl_cov_y_all'}		9.9452	6977.2	176.66	8.4101	8663.
{'fl_cor_y_all'}		0.25751	1	0.23161	0.21896	0.527
{'fl_cov_age_ss'}		0.39867	176.66	83.382	0.55101	713.
{'fl_cor_age_ss'}		0.094427	0.23161	1	0.13123	0.397
{'fl_cov_educ_ss'}		0.02286	8.4101	0.55101	0.21145	12.95
{'fl_cor_educ_ss'}		0.10752	0.21896	0.13123	1	0.1433
{'fl_cov_a_ss'}		16.463	8663.4	713.9	12.958	3864
{'fl_cor_a_ss'}		0.18113	0.5276	0.3977	0.14334	
{'fl_cov_ap_ss'}		17.437	11197	743.03	13.851	3960
{'fl_cor_ap_ss'}		0.184	0.65402	0.397	0.14696	0.9828
{'fl_cov_MPC'}		-0.061242	-11.95	-0.93092	-0.0093462	-21.83
{'fl_cor_MPC'}		-0.42463	-0.45863	-0.32683	-0.065159	-0.3561
{'fl_cov_Mass'}		-2.6557e-07	-3.5455e-05	-3.3149e-06	-1.3715e-07	-6.3012e-0
{'fl_cor_Mass'}		-0.38696	-0.28596	-0.24457	-0.20093	-0.2159
{'fl_cov_c_ss'}		6.6057	2725.3	105	5.4818	5577.
{'fl_cor_c_ss'}		0.35578	0.81251	0.28636	0.29687	0.706
{'fl_cov_y_head_inc'}		1.3302	3539.8	118.35	5.592	7371.
{'fl_cor_y_head_inc'}		0.05051	0.744	0.22755	0.2135	0.6583
{'fl_cov_y_spouse'}		8.6149	3437.4	58.307	2.8181	1291.
{'fl_cor_y_spouse'}		0.3324	0.73415	0.11391	0.10933	0.1172
{'fl_cov_yshr_nttxss'}		0.0052966	1.9936	0.078236	0.0034153	2.582
{'fl_cor_yshr_nttxss'}		0.36767	0.76604	0.27499	0.23838	0.4216
{'fracByP0_01'}		0	4.0037e-06	0.0041476	0	
{'fracByP10'}		0	0.021316	0.072166	0	
{'fracByP25'}		0	0.078153	0.18337	0	0.0003122
{'fracByP50'}		1	0.22773	0.39789	0	0.03805
{'fracByP75'}		1	0.47322	0.69228	1	0.1876
{'fracByP90'}		1	0.7065	0.86	1	0.4594
{'fracByP99_99'}		1	0.9988	1	1	0.9967

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

kids =4

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxx tb_outcomes: all stats xxx

OriginalVariableNames	married	y_all	age_ss	educ_ss	a_ss
{'mean'}	0.78724	91.676	35.383	0.29511	81.61
{'unweighted_sum'}	1	1.4702e+09	1909	1	1.2935e+09

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

{'sd'}	}	0.40926	74.133	7.9178	0.45609	164.
{'coefofvar'}	}	0.51987	0.80864	0.22378	1.5455	2.013
{'gini'}	}	0.054374	0.38168	0.12297	0.62738	0.7274
{'min'}	}	0	2.2124	19	0	
{'max'}	}	1	2113.2	64	1	7837.
{'pYis0'}	}	0.21276	0	0	0.70489	0.1891
{'pYls0'}	}	0	0	0	0	
{'pYgr0'}	}	0.78724	1	1	0.29511	0.8108
{'pYisMINY'}	}	0.21276	9.2536e-08	0.00035072	0.70489	0.1891
{'pYisMAXY'}	}	0.78724	2.0254e-13	0.00027556	0.29511	1.1672e-0
{'p0_01'}	}	0	4.7807	19	0	
{'p10'}	}	0	29.13	26	0	
{'p25'}	}	1	44.24	29	0	0.8072
{'p50'}	}	1	71.8	35	0	29.89
{'p75'}	}	1	115.49	41	1	82.0
{'p90'}	}	1	172.56	46	1	239.1
{'p99_99'}	}	1	888.01	64	1	2910.
{'fl_cov_married'}	}	0.16749	5.9174	0.25239	0.018555	8.620
{'fl_cor_married'}	}	1	0.19504	0.077888	0.099404	0.128
{'fl_cov_y_all'}	}	5.9174	5495.7	126.24	7.9495	6630.
{'fl_cor_y_all'}	}	0.19504	1	0.21507	0.23511	0.5443
{'fl_cov_age_ss'}	}	0.25239	126.24	62.692	0.61699	463.
{'fl_cor_age_ss'}	}	0.077888	0.21507	1	0.17085	0.3564
{'fl_cov_educ_ss'}	}	0.018555	7.9495	0.61699	0.20802	10.80
{'fl_cor_educ_ss'}	}	0.099404	0.23511	0.17085	1	0.1442
{'fl_cov_a_ss'}	}	8.6206	6630.3	463.7	10.809	2699
{'fl_cor_a_ss'}	}	0.1282	0.54435	0.35644	0.14424	
{'fl_cov_ap_ss'}	}	8.7102	8295.7	479.98	11.425	2761
{'fl_cor_ap_ss'}	}	0.12468	0.65556	0.35513	0.14675	0.9845
{'fl_cov_MPC'}	}	-0.04739	-10.199	-0.82379	-0.011367	-17.74
{'fl_cor_MPC'}	}	-0.39132	-0.46494	-0.3516	-0.084227	-0.3649
{'fl_cov_Mass'}	}	-1.0116e-07	-1.7179e-05	-1.6822e-06	-8.6576e-08	-3.0647e-0
{'fl_cor_Mass'}	}	-0.30657	-0.28741	-0.26349	-0.23543	-0.2313
{'fl_cov_c_ss'}	}	4.4325	2483.7	79.686	5.416	4382.
{'fl_cor_c_ss'}	}	0.27632	0.85476	0.25676	0.30296	0.680
{'fl_cov_y_head_inc'}	}	0.92362	3408.6	90.849	5.7717	5998.
{'fl_cor_y_head_inc'}	}	0.03994	0.81373	0.20306	0.22396	0.6461
{'fl_cov_y_spouse'}	}	4.9938	2087.1	35.394	2.1778	631.8
{'fl_cor_y_spouse'}	}	0.28208	0.65082	0.10334	0.11038	0.08889
{'fl_cov_yshr_nttxss'}		0.0035289	1.7418	0.059073	0.0033945	1.929
{'fl_cor_yshr_nttxss'}		0.29013	0.79058	0.25104	0.25043	0.3950
{'fracByP0_01'}	}	0	4.5191e-06	0.0018833	0	
{'fracByP10'}	}	0	0.023539	0.08684	0	
{'fracByP25'}	}	1	0.083914	0.18322	0	0.0003800
{'fracByP50'}	}	1	0.24018	0.44948	0	0.05909
{'fracByP75'}	}	1	0.48959	0.71178	1	0.2009
{'fracByP90'}	}	1	0.71753	0.86506	1	0.4875
{'fracByP99_99'}	}	1	0.9989	1	1	0.9960

12.1.10 Distributional Statistics By Marital Status and Kids Count

Various statistics, including MPC (of the first check) by Marital Status and Kids COunt

```
it_row_ctr = 0;
for it_marry_ctr=1:mp_params('n_marriedgrid')

    display(['']);
    display(['']);
    display(['-----']);
```

```

display(['-----']);
display(['-----']);
display(['-----']);
display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);
display(['Marital =' num2str(ar_marital(it_marry_ctr))]);
display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);
display(['-----']);
display(['-----']);

for it_kids_ctr=1:mp_params('n_kidsgrid')
    display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);
    display(['Marital =' num2str(ar_marital(it_marry_ctr)) ' and kids =' num2str(ar_kids(it_kids_ctr))]);
    display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);

    % construct input data
    y_all_grp = y_all(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    age_ss_grp = age_ss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    educ_ss_grp = educ_ss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    a_ss_grp = a_ss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    ap_ss_grp = ap_ss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    mn_MPC_C_gain_share_check_grp = mn_MPC_C_gain_share_check(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    Phi_true_grp = Phi_true_1(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    c_ss_grp = c_ss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    y_head_inc_grp = y_head_inc(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    y_spouse_inc_grp = y_spouse_inc(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    yshr_nttxss_grp = yshr_nttxss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);

    mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
    mp_cl_ar_xyz_of_s('y_all') = {y_all_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('age_ss') = {age_ss_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('educ_ss') = {educ_ss_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('a_ss') = {a_ss_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('ap_ss') = {ap_ss_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('MPC') = {mn_MPC_C_gain_share_check_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('Mass') = {Phi_true_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('c_ss') = {c_ss_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('y_head_inc') = {y_head_inc_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('y_spouse') = {y_spouse_inc_grp(:), zeros(1)};
    mp_cl_ar_xyz_of_s('yshr_nttxss') = {yshr_nttxss_grp(:), zeros(1});

    mp_cl_ar_xyz_of_s('ar_st_y_name') = ["y_all", "age_ss", "educ_ss", "a_ss", "ap_ss", "MPC", "Mass"];

    % controls
    mp_support = containers.Map('KeyType','char', 'ValueType','any');
    mp_support('ar_fl_percentiles') = [0.01 10 25 50 75 90 99.99];
    mp_support('bl_display_final') = true;
    mp_support('bl_display_detail') = false;
    mp_support('bl_display_drvm2outcomes') = false;
    mp_support('bl_display_drvstats') = false;
    mp_support('bl_display_drvm2covcor') = false;

    % Call Function
    mp_cl_mt_xyz_of_s = ff_simu_stats(Phi_true_grp(:)/sum(Phi_true_grp,'all'), mp_cl_ar_xyz_of_s);

    it_marital = ar_marital(it_marry_ctr);
    it_kids = ar_kids(it_kids_ctr);

    tb_dist_stats = mp_cl_mt_xyz_of_s('tb_outcomes');

```

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

```
fl_age_mean = tb_dist_stats{"age_ss", "mean"};
fl_age_p50 = tb_dist_stats{"age_ss", "p50"};

fl_educ_mean = tb_dist_stats{"educ_ss", "mean"};
fl_a_mean = tb_dist_stats{"a_ss", "mean"};
fl_a_p50 = tb_dist_stats{"a_ss", "p50"};

fl_ap_mean = tb_dist_stats{"ap_ss", "mean"};
fl_ap_p50 = tb_dist_stats{"ap_ss", "p50"};

fl_y_all_mean = tb_dist_stats{"y_all", "mean"};
fl_y_all_p50 = tb_dist_stats{"y_all", "p50"};

fl_mpc_mean = tb_dist_stats{"MPC", "mean"};
fl_mpc_p50 = tb_dist_stats{"MPC", "p50"};

fl_mass = tb_dist_stats{"Mass", "unweighted_sum"};

fl_c_ss_mean = tb_dist_stats{"c_ss", "mean"};
fl_c_ss_p50 = tb_dist_stats{"c_ss", "p50"};

fl_y_head_inc_mean = tb_dist_stats{"y_head_inc", "mean"};
fl_y_spouse_mean = tb_dist_stats{"y_spouse", "mean"};

ar_store_stats = [it_marital, it_kids, ...
    fl_age_mean, fl_age_p50, fl_educ_mean, ...
    fl_a_mean, fl_a_p50, fl_ap_mean, fl_ap_p50, ...
    fl_y_all_mean, fl_y_all_p50, ...
    fl_mpc_mean, fl_mpc_p50, ...
    fl_mass, ...
    fl_c_ss_mean, fl_c_ss_p50, ...
    fl_y_head_inc_mean, fl_y_spouse_mean];

it_row_ctr = it_row_ctr + 1;

if (it_row_ctr>1)
    mt_store_stats_by_mk = [mt_store_stats_by_mk;ar_store_stats];
else
    mt_store_stats_by_mk = [ar_store_stats];
end
end
end

0x0 empty char array

0x0 empty char array

-----
-----
-----
-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Marital =0
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
-----
-----
-----
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Marital =0 and kids =0
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxx tb_outcomes: all stats xxx

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
{'mean'}	71.752	42.174	0.25604	195.11	208.1
{'unweighted_sum'}	1.7831e+08	1909	1	1.2935e+05	1.6068e+0
{'sd'}	62.288	14.196	0.43644	339.77	352.4
{'coefofvar'}	0.8681	0.33661	1.7046	1.7414	1.693
{'gini'}	0.40852	0.1892	0.68372	0.7026	0.6989
{'min'}	2.2124	19	0	0	
{'max'}	1414.1	64	1	7837.6	8386.
{'pYiso'}	0	0	0.74396	0.11911	0.08185
{'pYlso'}	0	0	0	0	
{'pYgro'}	1	1	0.25604	0.88089	0.9181
{'pYisMINY'}	1.9394e-06	0.036566	0.74396	0.11911	0.08185
{'pYisMAXY'}	1.1947e-09	0.024953	0.25604	5.4117e-06	5.9148e-1
{'p0_01'}	3.6063	19	0	0	
{'p10'}	20.129	22	0	0	0.2008
{'p25'}	31.931	29	0	3.7372	6.563
{'p50'}	53.79	44	0	65.686	70.53
{'p75'}	90.494	55	1	239.18	258.9
{'p90'}	143.31	61	1	525.49	583.8
{'p99_99'}	816.36	64	1	4974.3	5033.
{'fl_cov_y_all'}	3879.8	217.85	3.8458	17148	1823
{'fl_cor_y_all'}	1	0.24637	0.14147	0.81024	0.8304
{'fl_cov_age_ss'}	217.85	201.53	-0.25515	2124.1	2205.
{'fl_cor_age_ss'}	0.24637	1	-0.041181	0.44036	0.4407
{'fl_cov_educ_ss'}	3.8458	-0.25515	0.19048	8.5838	9.279
{'fl_cor_educ_ss'}	0.14147	-0.041181	1	0.057885	0.06032
{'fl_cov_a_ss'}	17148	2124.1	8.5838	1.1544e+05	1.1966e+0
{'fl_cor_a_ss'}	0.81024	0.44036	0.057885	1	0.9992
{'fl_cov_ap_ss'}	18233	2205.5	9.2793	1.1966e+05	1.2423e+0
{'fl_cor_ap_ss'}	0.83049	0.44078	0.060323	0.99922	
{'fl_cov_MPC'}	-4.5809	-1.3124	0.020725	-17.446	-18.64
{'fl_cor_MPC'}	-0.30887	-0.38826	0.19943	-0.21564	-0.2221
{'fl_cov_Mass'}	-0.00012497	-5.6686e-05	-3.0201e-07	-0.00063245	-0.0006678
{'fl_cor_Mass'}	-0.21994	-0.43773	-0.075858	-0.20405	-0.2077
{'fl_cov_c_ss'}	1859.9	85.521	2.2319	8778.5	9253.
{'fl_cor_c_ss'}	0.97519	0.19675	0.16702	0.84382	0.857
{'fl_cov_y_head_inc'}	3879.8	217.85	3.8458	17148	1823
{'fl_cor_y_head_inc'}	1	0.24637	0.14147	0.81024	0.8304
{'fl_cov_y_spouse'}	0	0	0	0	
{'fl_cor_y_spouse'}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}	1.7036	0.14382	0.0019345	6.9206	7.370
{'fl_cor_yshr_nttxss'}	0.79998	0.29631	0.12964	0.59576	0.6116
{'fracByP0_01'}	4.4303e-06	0.016474	0	0	
{'fracByP10'}	0.020773	0.059565	0	0	5.8512e-0
{'fracByP25'}	0.075454	0.14379	0	0.0011338	0.001658
{'fracByP50'}	0.22298	0.3659	0	0.043403	0.03929
{'fracByP75'}	0.46643	0.67037	1	0.22326	0.2188
{'fracByP90'}	0.70129	0.88673	1	0.49372	0.5010
{'fracByP99_99'}	0.99869	1	1	0.99759	0.9971

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Marital =0 and kids =1
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

xxx tb_outcomes: all stats xxx

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
{'mean'}	65.867	36.118	0.25753	94.382	101.22
{'unweighted_sum'}	1.7831e+08	1909	1	1.2935e+05	1.5913e+09
{'sd'}	56.932	11.182	0.43728	214.92	223.55
{'coefofvar'}	0.86435	0.3096	1.6979	2.2771	2.2087
{'gini'}	0.40364	0.17438	0.68158	0.79318	0.79099
{'min'}	2.2124	19	0	0	0
{'max'}	1414.1	64	1	7837.6	8291.1
{'pYiso'}	0	0	0.74247	0.23563	0.21309
{'pYls0'}	0	0	0	0	0
{'pYgro'}	1	1	0.25753	0.76437	0.78691
{'pYisMINY'}	1.6845e-06	0.020845	0.74247	0.23563	0.21309
{'pYisMAXY'}	3.4305e-10	0.0031122	0.25753	8.262e-07	1.6379e-10
{'p0_01'}	3.5188	19	0	0	0
{'p10'}	19.292	22	0	0	0
{'p25'}	30.023	26	0	0.029898	0.23918
{'p50'}	49.454	35	0	10.255	14.159
{'p75'}	82.311	45	1	82.04	100.97
{'p90'}	130.49	52	1	276.88	293.79
{'p99_99'}	764.17	64	1	3737.2	3751.1
{'fl_cov_y_all'}	3241.3	152.53	4.1103	9427.3	10125
{'fl_cor_y_all'}	1	0.23959	0.16511	0.77047	0.79555
{'fl_cov_age_ss'}	152.53	125.05	0.19904	967.92	1008
{'fl_cor_age_ss'}	0.23959	1	0.040704	0.40274	0.40323
{'fl_cov_educ_ss'}	4.1103	0.19904	0.19121	5.7853	6.3576
{'fl_cor_educ_ss'}	0.16511	0.040704	1	0.061559	0.065037
{'fl_cov_a_ss'}	9427.3	967.92	5.7853	46190	47995
{'fl_cor_a_ss'}	0.77047	0.40274	0.061559	1	0.99895
{'fl_cov_ap_ss'}	10125	1008	6.3576	47995	49975
{'fl_cor_ap_ss'}	0.79555	0.40323	0.065037	0.99895	1
{'fl_cov_MPC'}	-8.9788	-1.3659	0.021532	-20.591	-22.11
{'fl_cor_MPC'}	-0.45848	-0.35509	0.14315	-0.27853	-0.28753
{'fl_cov_Mass'}	-4.8054e-05	-1.3242e-05	-1.409e-07	-0.00013616	-0.0001451
{'fl_cor_Mass'}	-0.33303	-0.46722	-0.12714	-0.24996	-0.25609
{'fl_cov_c_ss'}	1766.6	76.831	2.5674	5341.5	5695.3
{'fl_cor_c_ss'}	0.98556	0.21823	0.18649	0.7894	0.80918
{'fl_cov_y_head_inc'}	3241.3	152.53	4.1103	9427.3	10125
{'fl_cor_y_head_inc'}	1	0.23959	0.16511	0.77047	0.79555
{'fl_cov_y_spouse'}	0	0	0	0	0
{'fl_cor_y_spouse'}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}	1.5555	0.10317	0.0024166	3.8467	4.1371
{'fl_cor_yshr_nttxss'}	0.80522	0.2719	0.16288	0.5275	0.54542
{'fracByP0_01'}	4.7258e-06	0.010966	0	0	0
{'fracByP10'}	0.021712	0.068419	0	0	0
{'fracByP25'}	0.078097	0.15825	0	6.9352e-06	3.5474e-05
{'fracByP50'}	0.22713	0.37923	0	0.0099825	0.010737
{'fracByP75'}	0.46939	0.67298	1	0.11689	0.12619
{'fracByP90'}	0.70221	0.85638	1	0.39992	0.38891
{'fracByP99_99'}	0.99866	1	1	0.99622	0.99554

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Marital =0 and kids =2

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxx tb_outcomes: all stats xxx

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
-----------------------	-------	--------	---------	------	-------

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
{'mean'}	64.473	34.566	0.24576	62.952	67.62
{'unweighted_sum'}	1.7831e+08	1909	1	1.2935e+05	1.5826e+0
{'sd'}	54.982	9.1574	0.43053	161.65	168.9
{'coefofvar'}	0.85279	0.26492	1.7519	2.5678	2.498
{'gini'}	0.39845	0.14726	0.69833	0.83755	0.8405
{'min'}	2.2124	19	0	0	0
{'max'}	1414.1	64	1	7837.6	822
{'pYis0'}	0	0	0.75424	0.36638	0.3665
{'pYls0'}	0	0	0	0	0
{'pYgr0'}	1	1	0.24576	0.63362	0.6334
{'pYisMINY'}	9.8494e-07	0.01156	0.75424	0.36638	0.3665
{'pYisMAXY'}	6.7773e-11	0.00057855	0.24576	1.4e-07	9.363e-1
{'p0_01'}	3.5915	19	0	0	0
{'p10'}	19.324	23	0	0	0
{'p25'}	29.888	27	0	0	0
{'p50'}	48.811	34	0	1.9135	3.284
{'p75'}	80.448	41	0	51.664	56.74
{'p90'}	126.75	47	1	174.36	199.5
{'p99_99'}	740.25	64	1	2910.1	3038.
{'fl_cov_y_all'}	3023	108.56	4.4029	6789.1	7342.
{'fl_cor_y_all'}	1	0.21562	0.186	0.76387	0.790
{'fl_cov_age_ss'}	108.56	83.858	0.38657	516.18	539.9
{'fl_cor_age_ss'}	0.21562	1	0.09805	0.3487	0.3489
{'fl_cov_educ_ss'}	4.4029	0.38657	0.18536	4.3837	4.932
{'fl_cor_educ_ss'}	0.186	0.09805	1	0.062988	0.06779
{'fl_cov_a_ss'}	6789.1	516.18	4.3837	26131	2728
{'fl_cor_a_ss'}	0.76387	0.3487	0.062988	1	0.9987
{'fl_cov_ap_ss'}	7342.2	539.94	4.9328	27284	2855
{'fl_cor_ap_ss'}	0.7902	0.34891	0.067799	0.99878	
{'fl_cov_MPC'}	-12.648	-1.27	0.0077811	-21.879	-23.53
{'fl_cor_MPC'}	-0.55643	-0.33546	0.043715	-0.32737	-0.3368
{'fl_cov_Mass'}	-7.6259e-05	-1.1065e-05	-2.8216e-07	-0.00014947	-0.0001602
{'fl_cor_Mass'}	-0.36259	-0.31587	-0.17133	-0.24172	-0.2479
{'fl_cov_c_ss'}	1746.8	59.484	2.819	3991.5	4289.
{'fl_cor_c_ss'}	0.98869	0.20214	0.20376	0.76839	0.7899
{'fl_cov_y_head_inc'}	3023	108.56	4.4029	6789.1	7342.
{'fl_cor_y_head_inc'}	1	0.21562	0.186	0.76387	0.790
{'fl_cov_y_spouse'}	0	0	0	0	0
{'fl_cor_y_spouse'}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}	1.4898	0.073937	0.0027409	2.72	2.940
{'fl_cor_yshr_nttxss'}	0.8086	0.24094	0.18997	0.50211	0.5192
{'fracByP0_01'}	4.972e-06	0.0063542	0	0	0
{'fracByP10'}	0.02237	0.067607	0	0	0
{'fracByP25'}	0.079911	0.1793	0	0	0
{'fracByP50'}	0.23057	0.42619	0	0.0018299	0.002042
{'fracByP75'}	0.47337	0.67985	0	0.091755	0.08396
{'fracByP90'}	0.70508	0.85252	1	0.32859	0.3322
{'fracByP99_99'}	0.99867	1	1	0.99463	0.9945
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx					
Marital =0 and kids =3					
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx					
xxx tb_outcomes: all stats xxx					

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

{'mean'}	}	63.898	34.068	0.22983	48.134	51.95
{'unweighted_sum'}	}	1.7831e+08	1909	1	1.2935e+05	1.5774e+0
{'sd'}	}	54.001	7.9772	0.42073	134.32	141.0
{'coefofvar'}	}	0.84511	0.23415	1.8306	2.7906	2.715
{'gini'}	}	0.39521	0.12909	0.72073	0.86678	0.8694
{'min'}	}	2.2124	19	0	0	0
{'max'}	}	1414.1	64	1	7837.6	8183.
{'pYiso'}	}	0	0	0.77017	0.45656	0.4513
{'pYlso'}	}	0	0	0	0	0
{'pYgro'}	}	1	1	0.22983	0.54344	0.5486
{'pYisMINY'}	}	6.8879e-07	0.0083137	0.77017	0.45656	0.4513
{'pYisMAXY'}	}	1.1096e-11	0.00013776	0.22983	2.4752e-08	1.1431e-1
{'p0_01'}	}	3.6125	19	0	0	0
{'p10'}	}	19.479	24	0	0	0
{'p25'}	}	29.928	28	0	0	0
{'p50'}	}	48.607	33	0	0.23918	0.498
{'p75'}	}	79.715	39	0	29.898	37.19
{'p90'}	}	125.03	45	1	146.89	152.5
{'p99_99'}	}	727.36	64	1	2546.8	263
{'fl_cov_y_all'}	}	2916.1	86.208	4.4997	5525	6009.
{'fl_cor_y_all'}	}	1	0.20012	0.19805	0.76171	0.7889
{'fl_cov_age_ss'}	}	86.208	63.635	0.47515	327.14	343.3
{'fl_cor_age_ss'}	}	0.20012	1	0.14157	0.30531	0.3051
{'fl_cov_educ_ss'}	}	4.4997	0.47515	0.17701	3.7005	4.224
{'fl_cor_educ_ss'}	}	0.19805	0.14157	1	0.065482	0.07117
{'fl_cov_a_ss'}	}	5525	327.14	3.7005	18042	1892
{'fl_cor_a_ss'}	}	0.76171	0.30531	0.065482	1	0.9986
{'fl_cov_ap_ss'}	}	6009.3	343.36	4.2241	18921	1989
{'fl_cor_ap_ss'}	}	0.78891	0.30515	0.071177	0.99863	
{'fl_cov_MPC'}	}	-14.294	-1.1063	-0.0020988	-20.435	-22.07
{'fl_cor_MPC'}	}	-0.611	-0.32013	-0.011516	-0.35119	-0.3612
{'fl_cov_Mass'}	}	-4.6648e-05	-5.1607e-06	-2.1548e-07	-7.4065e-05	-7.9832e-0
{'fl_cor_Mass'}	}	-0.3662	-0.27425	-0.21712	-0.23375	-0.2399
{'fl_cov_c_ss'}	}	1734.9	49.892	2.92	3306.4	3575.
{'fl_cor_c_ss'}	}	0.99009	0.19275	0.21389	0.75862	0.7812
{'fl_cov_y_head_inc'}	}	2916.1	86.208	4.4997	5525	6009.
{'fl_cor_y_head_inc'}	}	1	0.20012	0.19805	0.76171	0.7889
{'fl_cov_y_spouse'}	}	0	0	0	0	0
{'fl_cor_y_spouse'}	}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}	}	1.4552	0.059225	0.0028436	2.1663	2.353
{'fl_cor_yshr_nttxss'}	}	0.81051	0.2233	0.20329	0.4851	0.5018
{'fracByP0_01'}	}	5.1241e-06	0.0046366	0	0	
{'fracByP10'}	}	0.022798	0.072031	0	0	
{'fracByP25'}	}	0.081073	0.1985	0	0	
{'fracByP50'}	}	0.23275	0.41302	0	0.00019689	0.0001865
{'fracByP75'}	}	0.47593	0.67505	0	0.056161	0.05684
{'fracByP90'}	}	0.70691	0.86752	1	0.31708	0.2910
{'fracByP99_99'}	}	0.99868	1	1	0.99425	0.9938

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Marital =0 and kids =4

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxx tb_outcomes: all stats xxx

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
{'mean'}	63.864	34.197	0.2079	41.099	44.67
{'unweighted_sum'}	1.7831e+08	1909	1	1.2935e+05	1.5749e+0

{'sd'}		53.609	7.1538	0.4058	120.68	127.2
{'coefofvar'}	}	0.83942	0.2092	1.9519	2.9363	2.849
{'gini'}	}	0.39289	0.11447	0.75112	0.88249	0.8837
{'min'}	}	2.2124	19	0	0	
{'max'}	}	1414.1	64	1	7837.6	8164.
{'pYis0'}	}	0	0	0.7921	0.50485	0.4925
{'pYls0'}	}	0	0	0	0	
{'pYgr0'}	}	1	1	0.2079	0.49515	0.5074
{'pYisMINY'}	}	4.3493e-07	0.0045732	0.7921	0.50485	0.4925
{'pYisMAXY'}	}	1.4837e-12	4.6124e-05	0.2079	5.166e-09	1.5175e-1
{'p0_01'}	}	3.6887	19	0	0	
{'p10'}	}	19.685	25	0	0	
{'p25'}	}	30.153	29	0	0	
{'p50'}	}	48.75	34	0	0	0.02989
{'p75'}	}	79.548	39	0	21.796	27.79
{'p90'}	}	124.62	44	1	122.46	130.0
{'p99_99'}	}	721.01	63	1	2377.1	2423.
{'fl_cov_y_all'}		2873.9	71.941	4.4963	4930.8	539
{'fl_cor_y_all'}		1	0.18759	0.20668	0.76216	0.7900
{'fl_cov_age_ss'}		71.941	51.176	0.5437	238.82	251.8
{'fl_cor_age_ss'}		0.18759	1	0.18729	0.27663	0.2765
{'fl_cov_educ_ss'}		4.4963	0.5437	0.16468	3.6463	4.175
{'fl_cor_educ_ss'}		0.20668	0.18729	1	0.074456	0.08084
{'fl_cov_a_ss'}		4930.8	238.82	3.6463	14564	1533
{'fl_cor_a_ss'}		0.76216	0.27663	0.074456	1	0.9985
{'fl_cov_ap_ss'}		5391	251.85	4.1759	15338	1620
{'fl_cor_ap_ss'}		0.79005	0.27658	0.080846	0.99852	
{'fl_cov_MPC'}		-14.976	-1.0328	-0.010238	-18.883	-20.53
{'fl_cor_MPC'}		-0.63881	-0.33014	-0.057692	-0.35781	-0.3689
{'fl_cov_Mass'}		-2.7333e-05	-2.6204e-06	-1.3997e-07	-3.8707e-05	-4.2009e-0
{'fl_cor_Mass'}		-0.36608	-0.263	-0.24765	-0.2303	-0.2369
{'fl_cov_c_ss'}		1727	42.146	2.9107	2959.6	3218.
{'fl_cor_c_ss'}		0.9908	0.1812	0.22061	0.75427	0.7777
{'fl_cov_y_head_inc'}		2873.9	71.941	4.4963	4930.8	539
{'fl_cor_y_head_inc'}		1	0.18759	0.20668	0.76216	0.7900
{'fl_cov_y_spouse'}		0	0	0	0	
{'fl_cor_y_spouse'}		NaN	NaN	NaN	NaN	Na
{'fl_cov_yshr_nttxss'}		1.437	0.049376	0.0028207	1.8938	2.069
{'fl_cor_yshr_nttxss'}		0.81163	0.20899	0.21047	0.47516	0.4922
{'fracByP0_01'}		5.1325e-06	0.0025409	0	0	
{'fracByP10'}		0.022996	0.072385	0	0	
{'fracByP25'}		0.081963	0.20856	0	0	
{'fracByP50'}		0.23436	0.45949	0	0	4.2228e-0
{'fracByP75'}		0.47776	0.70775	0	0.041779	0.04305
{'fracByP90'}		0.70826	0.87861	1	0.2851	0.266
{'fracByP99_99'}		0.99869	0.99991	1	0.99427	0.9933

0x0 empty char array

0x0 empty char array

xxxxxxxxxxxxxxxxxxxxxxxxxxxx
Marital =1
xxxxxxxxxxxxxxxxxxxxxxxxxxxx

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Marital =1 and kids =0
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxx tb_outcomes: all stats xxx

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
{'mean'}	109.57	44.041	0.38021	408.45	434.2
{'unweighted_sum'}	1.2919e+09	1909	1	1.2935e+05	8.4753e+09
{'sd'}	83.584	15.135	0.48544	498.74	514.1
{'coefofvar'}	0.76282	0.34365	1.2768	1.2211	1.184
{'gini'}	0.3692	0.18905	0.50257	0.57853	0.57485
{'min'}	2.4223	19	0	0	0
{'max'}	2113.2	64	1	7837.6	9503.9
{'pYis0'}	0	0	0.61979	0.07588	0.036618
{'pYls0'}	0	0	0	0	0
{'pYgr0'}	1	1	0.38021	0.92412	0.96338
{'pYisMINY'}	5.1901e-09	0.043093	0.61979	0.07588	0.036618
{'pYisMAXY'}	2.4329e-11	0.038439	0.38021	1.8938e-05	1.2944e-11
{'p0_01'}	6.597	19	0	0	0
{'p10'}	35.716	21	0	1.9135	3.7372
{'p25'}	54.783	29	0	51.664	59.412
{'p50'}	88.064	48	0	239.18	270.71
{'p75'}	137.9	58	1	588.48	634.2
{'p90'}	204.71	62	1	1074.4	1074.4
{'p99_99'}	967.74	64	1	5833.4	5977.2
{'fl_cov_y_all'}	6986.3	361.95	5.7657	27155	31359
{'fl_cor_y_all'}	1	0.28612	0.1421	0.65142	0.72978
{'fl_cov_age_ss'}	361.95	229.06	-0.85351	4123.7	4247.6
{'fl_cor_age_ss'}	0.28612	1	-0.11617	0.5463	0.54592
{'fl_cov_educ_ss'}	5.7657	-0.85351	0.23565	16.048	17.247
{'fl_cor_educ_ss'}	0.1421	-0.11617	1	0.066283	0.069108
{'fl_cov_a_ss'}	27155	4123.7	16.048	2.4874e+05	2.541e+05
{'fl_cor_a_ss'}	0.65142	0.5463	0.066283	1	0.99103
{'fl_cov_ap_ss'}	31359	4247.6	17.247	2.541e+05	2.643e+05
{'fl_cor_ap_ss'}	0.72978	0.54592	0.069108	0.99103	1
{'fl_cov_MPC'}	-4.5371	-1.4233	0.0094147	-28.557	-30.413
{'fl_cor_MPC'}	-0.3463	-0.59994	0.12373	-0.36528	-0.37741
{'fl_cov_Mass'}	-7.4044e-05	-2.2911e-05	5.5246e-08	-0.00041977	-0.00044771
{'fl_cor_Mass'}	-0.19462	-0.33257	0.025002	-0.18491	-0.19132
{'fl_cov_c_ss'}	2941.3	188.87	4.2928	16348	17228
{'fl_cor_c_ss'}	0.86319	0.30611	0.21692	0.80402	0.82201
{'fl_cov_y_head_inc'}	4857.1	318.48	4.4672	25709	27033
{'fl_cor_y_head_inc'}	0.85851	0.31089	0.13595	0.76156	0.77684
{'fl_cov_y_spouse'}	4673.1	95.4	2.8501	3173.5	9494.5
{'fl_cor_y_spouse'}	0.59164	0.066703	0.06213	0.067335	0.19543
{'fl_cov_yshr_nttxss'}	1.6494	0.13956	0.0013995	6.3032	7.2859
{'fl_cor_yshr_nttxss'}	0.7665	0.35816	0.11198	0.49089	0.55047
{'fracByP0_01'}	5.3214e-06	0.018591	0	0	0
{'fracByP10'}	0.024266	0.049706	0	0.00015705	0.00034862
{'fracByP25'}	0.086535	0.1342	0	0.010018	0.010404
{'fracByP50'}	0.24796	0.35975	0	0.097271	0.098709

12.1.11 Distributional Statistics By Marital Status, Kids Count and Income Bins

Various statistics, including MPC (of the first check) by Marital Status and Kids CCount and income bins

```

it_row_ctr = 0;
for it_marry_ctr=1:mp_params('n_marriedgrid')

    display(['']);
    display(['']);
    display(['-----']);
    display(['-----']);
    display(['-----']);
    display(['-----']);
    display(['-----']);
    display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);
    display(['Marital =' num2str(ar_marital(it_marry_ctr))]);
    display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);
    display(['-----']);
    display(['-----']);

for it_kids_ctr=1:mp_params('n_kidsgrid')
    display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);
    display(['Marital =' num2str(ar_marital(it_marry_ctr)) ' and kids =' num2str(ar_kids(it_kids_ctr))]);
    display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);

    % construct input data
    y_all_grp = y_all(min_age:max_age, :, :, :, it_marry_ctr ,it_ctr);
    age_ss_grp = age_ss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    educ_ss_grp = educ_ss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    a_ss_grp = a_ss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    ap_ss_grp = ap_ss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    mn_MPC_C_gain_share_check_grp = mn_MPC_C_gain_share_check(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    Phi_true_grp = Phi_true_1(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    c_ss_grp = c_ss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    y_head_inc_grp = y_head_inc(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    y_spouse_inc_grp = y_spouse_inc(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);
    yshr_nttxss_grp = yshr_nttxss(min_age:max_age, :, :, :, it_marry_ctr, it_kids_ctr);

    % Income Bins
    ar_y_all = y_all_grp(:);
    ar_age_ss = age_ss_grp(:);
    ar_educ_ss = educ_ss_grp(:);
    ar_a_ss = a_ss_grp(:);
    ar_ap_ss = ap_ss_grp(:);
    ar_mn_MPC_C_gain_share_check = mn_MPC_C_gain_share_check_grp(:);
    ar_Phi_true = Phi_true_grp(:);
    ar_c_ss = c_ss_grp(:);
    ar_y_head_inc = y_head_inc_grp(:);
    ar_y_spouse_inc = y_spouse_inc_grp(:);
    ar_yshr_nttxss = yshr_nttxss_grp(:);

    % income bins loop
    for it_y_all_ctr=1:6

        % Current y group index
        % y is in thousands of dollars
        y_all_start = (it_y_all_ctr-1)*20;
        if (it_y_all_ctr == 6)

```

```

        y_all_end = max(ar_y_all);
    else
        y_all_end = it_y_all_ctr*20;
    end

    display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);
    display(['Marital =' num2str(ar_marital(it_marry_ctr)) ', kids =' num2str(ar_kids(it_kid));
    display(['xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx']);

ar_y_idx = (ar_y_all >= y_all_start & ar_y_all <y_all_end);

ar_mky_y_all = ar_y_all(ar_y_idx);
ar_mky_age_ss = ar_age_ss(ar_y_idx);
ar_mky_educ_ss = ar_educ_ss(ar_y_idx);
ar_mky_a_ss = ar_a_ss(ar_y_idx);
ar_mky_ap_ss = ar_ap_ss(ar_y_idx);
ar_mky_mn_MPC_C_gain_share_check = ar_mn_MPC_C_gain_share_check(ar_y_idx);
ar_mky_Phi_true = ar_Phi_true(ar_y_idx);
ar_mky_c_ss = ar_c_ss(ar_y_idx);
ar_mky_y_head_inc = ar_y_head_inc(ar_y_idx);
ar_mky_y_spouse_inc = ar_y_spouse_inc(ar_y_idx);
ar_mky_yshr_nttxss = ar_yshr_nttxss(ar_y_idx);

mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
mp_cl_ar_xyz_of_s('y_all') = {ar_mky_y_all(:), zeros(1)};
mp_cl_ar_xyz_of_s('age_ss') = {ar_mky_age_ss(:), zeros(1)};
mp_cl_ar_xyz_of_s('educ_ss') = {ar_mky_educ_ss(:), zeros(1)};
mp_cl_ar_xyz_of_s('a_ss') = {ar_mky_a_ss(:), zeros(1)};
mp_cl_ar_xyz_of_s('ap_ss') = {ar_mky_ap_ss(:), zeros(1)};
mp_cl_ar_xyz_of_s('MPC') = {ar_mky_mn_MPC_C_gain_share_check(:), zeros(1)};
mp_cl_ar_xyz_of_s('Mass') = {ar_mky_Phi_true(:), zeros(1)};
mp_cl_ar_xyz_of_s('c_ss') = {ar_mky_c_ss(:), zeros(1)};
mp_cl_ar_xyz_of_s('y_head_inc') = {ar_mky_y_head_inc(:), zeros(1)};
mp_cl_ar_xyz_of_s('y_spouse') = {ar_mky_y_spouse_inc(:), zeros(1)};
mp_cl_ar_xyz_of_s('yshr_nttxss') = {ar_mky_yshr_nttxss(:), zeros(1)};
mp_cl_ar_xyz_of_s('ar_st_y_name') = ["y_all", "age_ss", "educ_ss", "a_ss", "ap_ss", "MPC"]

% controls
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('ar_fl_percentiles') = [0.01 10 25 50 75 90 99.99];
mp_support('bl_display_final') = true;
mp_support('bl_display_detail') = false;
mp_support('bl_display_drvm2outcomes') = false;
mp_support('bl_display_drvstats') = false;
mp_support('bl_display_drvm2covcor') = false;

% Call Function
mp_cl_mt_xyz_of_s = ff_simu_stats(ar_mky_Phi_true(:)/sum(ar_mky_Phi_true,'all'), mp_cl_ar_xyz_of_s);

it_marital = ar_marital(it_marry_ctr);
it_kids = ar_kids(it_kids_ctr);
fl_y_all_start = y_all_start;
fl_y_all_end = y_all_end;

tb_dist_stats = mp_cl_mt_xyz_of_s('tb_outcomes');
fl_age_mean = tb_dist_stats{"age_ss", "mean"};
fl_age_p50 = tb_dist_stats{"age_ss", "p50"};

```


12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Marital =0, kids =0, ybin =0 to 20

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxx tb_outcomes: all stats xxx

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
{'mean'}	14.688	34.974	0.18182	2.7226	2.6703
{'unweighted_sum'}	8.4083e+05	1909	1	2690.8	5.2986e+06
{'sd'}	3.6764	14.501	0.3857	10.125	9.7397
{'coeofvar'}	0.25031	0.41462	2.1213	3.7189	3.6474
{'gini'}	0.141	0.23128	0.78641	0.92249	0.92536
{'min'}	2.2124	19	0	0	0
{'max'}	20	64	1	413.31	409.12
{'pYiso'}	0	0	0.81818	0.53967	0.49704
{'pYls0'}	0	0	0	0	0
{'pYgro'}	1	1	0.18182	0.46033	0.50296
{'pYisMINY'}	1.9988e-05	0.084859	0.81818	0.53967	0.49704
{'pYisMAXY'}	4.7568e-12	0.01496	0.18182	1.4916e-11	0
{'p0_01'}	2.6052	19	0	0	0
{'p10'}	9.307	20	0	0	0
{'p25'}	12.172	22	0	0	0
{'p50'}	15.236	30	0	0	0.011132
{'p75'}	17.778	48	0	0.23918	0.48535
{'p90'}	19.14	58	1	6.458	6.0051
{'p99_99'}	19.999	64	1	174.36	166.76
{'fl_cov_y_all'}	13.516	5.6455	0.0023525	6.9774	7.1104
{'fl_cor_y_all'}	1	0.1059	0.001659	0.18744	0.19857
{'fl_cov_age_ss'}	5.6455	210.28	-1.0046	57.763	56.482
{'fl_cor_age_ss'}	0.1059	1	-0.17962	0.39342	0.39991
{'fl_cov_educ_ss'}	0.0023525	-1.0046	0.14876	-0.29328	-0.29618
{'fl_cor_educ_ss'}	0.001659	-0.17962	1	-0.0751	-0.078843
{'fl_cov_a_ss'}	6.9774	57.763	-0.29328	102.52	98.523
{'fl_cor_a_ss'}	0.18744	0.39342	-0.0751	1	0.99907
{'fl_cov_ap_ss'}	7.1104	56.482	-0.29618	98.523	94.862
{'fl_cor_ap_ss'}	0.19857	0.39991	-0.078843	0.99907	1
{'fl_cov_MPC'}	-0.53539	-2.7127	0.035745	-1.3183	-1.3069
{'fl_cor_MPC'}	-0.41675	-0.53535	0.26522	-0.37261	-0.38399
{'fl_cov_Mass'}	6.2968e-06	-7.1893e-05	-3.4162e-07	-1.6308e-05	-1.5421e-05
{'fl_cor_Mass'}	0.18379	-0.53201	-0.095046	-0.17284	-0.1699
{'fl_cov_c_ss'}	11.292	6.0576	0.0049126	9.8651	9.6441
{'fl_cor_c_ss'}	0.98267	0.13365	0.0040749	0.31172	0.31679
{'fl_cov_y_head_inc'}	13.516	5.6455	0.0023525	6.9774	7.1104
{'fl_cor_y_head_inc'}	1	0.1059	0.001659	0.18744	0.19857
{'fl_cov_y_spouse'}	0	0	0	0	0
{'fl_cor_y_spouse'}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}	0.052502	0.022189	1.0001e-05	0.025753	0.026154
{'fl_cor_yshr_nttxss'}	0.99014	0.10609	0.0017978	0.17635	0.18618
{'fracByP0_01'}	1.8388e-05	0.0461	0	0	0
{'fracByP10'}	0.051228	0.088792	0	0	0
{'fracByP25'}	0.16212	0.16288	0	0	0
{'fracByP50'}	0.39701	0.3352	0	0	6.3754e-06
{'fracByP75'}	0.68371	0.60992	0	0.013023	0.018274
{'fracByP90'}	0.86704	0.84015	1	0.16055	0.12661
{'fracByP99_99'}	1	1	1	0.99661	0.99334

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Marital =0, kids =0, ybin =20 to 40

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxx tb_outcomes: all stats xxx

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
{'mean'}	29.952	38.525	0.20638	25.273	26.927
{'unweighted_sum'}	1.9087e+06	1909	1	7355.4	1.8539e+07
{'sd'}	5.6854	14.456	0.4047	44.204	44.499
{'coefofvar'}	0.18982	0.37522	1.961	1.7491	1.6525
{'gini'}	0.10955	0.2128	0.7532	0.72521	0.71433
{'min'}	20	19	0	0	0
{'max'}	40	64	1	890.69	885.93
{'pYiso'}	0	0	0.79362	0.16854	0.10132
{'pYls0'}	0	0	0	0	0
{'pYgro'}	1	1	0.20638	0.83146	0.89868
{'pYisMINY'}	8.0995e-288	0.055295	0.79362	0.16854	0.10132
{'pYisMAXY'}	4.4783e-07	0.018337	0.20638	8.9396e-13	0
{'p0_01'}	20.004	19	0	0	0
{'p10'}	22.011	20	0	0	0
{'p25'}	25.065	25	0	0.80724	1.0692
{'p50'}	30.008	37	0	6.458	6.7423
{'p75'}	34.798	52	0	29.898	33.25
{'p90'}	37.826	59	1	82.04	83.859
{'p99_99'}	39.999	64	1	413.31	407.97
{'fl_cov_y_all'}	32.324	7.6438	0.038871	72.305	78.269
{'fl_cor_y_all'}	1	0.093006	0.016894	0.28771	0.30937
{'fl_cov_age_ss'}	7.6438	208.96	-0.73238	386.27	403.49
{'fl_cor_age_ss'}	0.093006	1	-0.12519	0.60451	0.62727
{'fl_cov_educ_ss'}	0.038871	-0.73238	0.16379	-2.1597	-2.3263
{'fl_cor_educ_ss'}	0.016894	-0.12519	1	-0.12073	-0.12918
{'fl_cov_a_ss'}	72.305	386.27	-2.1597	1954	1964.7
{'fl_cor_a_ss'}	0.28771	0.60451	-0.12073	1	0.99885
{'fl_cov_ap_ss'}	78.269	403.49	-2.3263	1964.7	1980.1
{'fl_cor_ap_ss'}	0.30937	0.62727	-0.12918	0.99885	1
{'fl_cov_MPC'}	-0.30518	-1.6691	0.062231	-3.0143	-3.2123
{'fl_cor_MPC'}	-0.208	-0.4474	0.59583	-0.26423	-0.27972
{'fl_cov_Mass'}	-4.5536e-06	-9.611e-05	3.7003e-08	-0.00013723	-0.00014127
{'fl_cor_Mass'}	-0.059962	-0.49775	0.0068451	-0.23242	-0.23768
{'fl_cov_c_ss'}	19.905	-11.1	0.19764	47.044	47.213
{'fl_cor_c_ss'}	0.88176	-0.19338	0.12299	0.26804	0.26721
{'fl_cov_y_head_inc'}	32.324	7.6438	0.038871	72.305	78.269
{'fl_cor_y_head_inc'}	1	0.093006	0.016894	0.28771	0.30937
{'fl_cov_y_spouse'}	0	0	0	0	0
{'fl_cor_y_spouse'}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}	0.058292	0.013838	6.9651e-05	0.12878	0.13941
{'fl_cor_yshr_nttxss'}	0.99551	0.092947	0.016711	0.28287	0.30419
{'fracByP0_01'}	0.00010275	0.02727	0	0	0
{'fracByP10'}	0.070196	0.052525	0	0	0
{'fracByP25'}	0.18834	0.15502	0	0.0035395	0.0030822
{'fracByP50'}	0.4181	0.33724	0	0.038073	0.032523
{'fracByP75'}	0.68834	0.64594	0	0.19439	0.19244
{'fracByP90'}	0.87021	0.84782	1	0.53295	0.49305
{'fracByP99_99'}	0.99996	1	1	0.99833	0.99827

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Marital =0, kids =0, ybin =40 to 60

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

xxx tb_outcomes: all stats xxx

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
{'mean'}	49.366	41.657	0.23457	81.386	87.44
{'unweighted_sum'}	2.5368e+06	1909	1	13261	2.8595e+0
{'sd'}	5.7595	14.091	0.42373	93.65	93.9
{'coefofvar'}	0.11667	0.33826	1.8064	1.1507	1.074
{'gini'}	0.067274	0.1907	0.71409	0.55942	0.5484
{'min'}	40	19	0	0	
{'max'}	60	64	1	1394.9	1389.
{'pYiso'}	0	0	0.76543	0.074319	0.03470
{'pYls0'}	0	0	0	0	
{'pYgro'}	1	1	0.23457	0.92568	0.9652
{'pYisMINY'}	1.1988e-05	0.035852	0.76543	0.074319	0.03470
{'pYisMAXY'}	2.6918e-19	0.022889	0.23457	1.725e-14	
{'p0_01'}	40.004	19	0	0	
{'p10'}	41.738	22	0	1.9135	4.178
{'p25'}	44.289	28	0	10.255	15.82
{'p50'}	49.163	43	0	51.664	56.52
{'p75'}	54.155	54	0	122.46	130.2
{'p90'}	57.677	60	1	205.07	213.8
{'p99_99'}	59.997	64	1	729.18	718.6
{'fl_cov_y_all'}	33.172	5.7383	0.031749	121.07	129.8
{'fl_cor_y_all'}	1	0.070707	0.013009	0.22446	0.2398
{'fl_cov_age_ss'}	5.7383	198.55	-0.52991	911.38	944.6
{'fl_cor_age_ss'}	0.070707	1	-0.088752	0.69065	0.7133
{'fl_cov_educ_ss'}	0.031749	-0.52991	0.17955	-5.8166	-6.25
{'fl_cor_educ_ss'}	0.013009	-0.088752	1	-0.14658	-0.1569
{'fl_cov_a_ss'}	121.07	911.38	-5.8166	8770.3	8794.
{'fl_cor_a_ss'}	0.22446	0.69065	-0.14658	1	0.9991
{'fl_cov_ap_ss'}	129.84	944.69	-6.252	8794.3	883
{'fl_cor_ap_ss'}	0.23985	0.71331	-0.15698	0.99911	
{'fl_cov_MPC'}	-0.09046	-0.71366	0.029802	-2.9986	-3.239
{'fl_cor_MPC'}	-0.096943	-0.31261	0.43412	-0.19763	-0.2127
{'fl_cov_Mass'}	-4.8663e-06	-5.8353e-05	-1.5517e-07	-0.00023117	-0.0002379
{'fl_cor_Mass'}	-0.088148	-0.43205	-0.038205	-0.25753	-0.264
{'fl_cov_c_ss'}	17.041	-28.838	0.46008	70.248	61.26
{'fl_cor_c_ss'}	0.62733	-0.43394	0.23022	0.15905	0.1382
{'fl_cov_y_head_inc'}	33.172	5.7383	0.031749	121.07	129.8
{'fl_cor_y_head_inc'}	1	0.070707	0.013009	0.22446	0.2398
{'fl_cov_y_spouse'}	0	0	0	0	
{'fl_cor_y_spouse'}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}	0.032789	0.0057148	3.1401e-05	0.11935	0.1280
{'fl_cor_yshr_nttxss'}	0.99787	0.071088	0.012989	0.22339	0.2387
{'fracByP0_01'}	8.1986e-05	0.016353	0	0	
{'fracByP10'}	0.082731	0.059676	0	0.00080736	0.001552
{'fracByP25'}	0.21327	0.13822	0	0.013027	0.01831
{'fracByP50'}	0.44964	0.36454	0	0.12623	0.1133
{'fracByP75'}	0.7111	0.65402	0	0.38755	0.3665
{'fracByP90'}	0.88093	0.85769	1	0.66284	0.654
{'fracByP99_99'}	0.99988	1	1	0.99951	0.9991

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Marital =0, kids =0, ybin =60 to 80

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxx tb_outcomes: all stats xxx

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
-----------------------	-------	--------	---------	------	-------

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
{'mean'}	69.288	43.863	0.26092	158.1	169.5
{'unweighted_sum'}	3.0245e+06	1909	1	20103	3.6714e+0
{'sd'}	5.7381	13.54	0.43913	144.91	144.6
{'coefofvar'}	0.082816	0.30869	1.683	0.91655	0.8531
{'gini'}	0.047737	0.17193	0.67675	0.46696	0.4577
{'min'}	60	19	0	0	0
{'max'}	79.999	64	1	1913.5	1907.
{'pYis0'}	0	0	0.73908	0.036459	0.006283
{'pYls0'}	0	0	0	0	0
{'pYgr0'}	1	1	0.26092	0.96354	0.9937
{'pYisMINY'}	7.0785e-08	0.024362	0.73908	0.036459	0.006283
{'pYisMAXY'}	1.0527e-08	0.027901	0.26092	8.7298e-17	
{'p0_01'}	60.004	19	0	0	0
{'p10'}	61.586	23	0	10.255	18.69
{'p25'}	64.221	32	0	39.794	54.13
{'p50'}	68.93	46	0	122.46	134.9
{'p75'}	74.224	56	1	239.18	250.8
{'p90'}	77.547	61	1	363.77	373.
{'p99_99'}	79.989	64	1	1074.4	1050.
{'fl_cov_y_all'}	32.926	4.1151	0.027108	145.03	154.4
{'fl_cor_y_all'}	1	0.052966	0.010758	0.17442	0.1861
{'fl_cov_age_ss'}	4.1151	183.33	-0.37329	1400.5	143.
{'fl_cor_age_ss'}	0.052966	1	-0.062782	0.71382	0.7342
{'fl_cov_educ_ss'}	0.027108	-0.37329	0.19284	-9.2359	-9.793
{'fl_cor_educ_ss'}	0.010758	-0.062782	1	-0.14514	-0.1541
{'fl_cov_a_ss'}	145.03	1400.5	-9.2359	20999	2094
{'fl_cor_a_ss'}	0.17442	0.71382	-0.14514	1	0.999
{'fl_cov_ap_ss'}	154.46	1438	-9.7931	20944	2091
{'fl_cor_ap_ss'}	0.18612	0.73429	-0.15419	0.9993	
{'fl_cov_MPC'}	-0.032152	-0.1128	0.0061122	-0.71422	-0.7941
{'fl_cor_MPC'}	-0.10557	-0.15696	0.26224	-0.092863	-0.1034
{'fl_cov_Mass'}	-2.7754e-06	-2.3008e-05	-3.5426e-07	-0.00016592	-0.0001677
{'fl_cor_Mass'}	-0.091883	-0.32281	-0.15325	-0.21751	-0.2203
{'fl_cov_c_ss'}	15.819	-34.276	0.57801	165.89	143.
{'fl_cor_c_ss'}	0.46921	-0.43083	0.22402	0.19483	0.1688
{'fl_cov_y_head_inc'}	32.926	4.1151	0.027108	145.03	154.4
{'fl_cor_y_head_inc'}	1	0.052966	0.010758	0.17442	0.1861
{'fl_cov_y_spouse'}	0	0	0	0	0
{'fl_cor_y_spouse'}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}	0.020855	0.0026153	1.697e-05	0.091765	0.0977
{'fl_cor_yshr_nttxss'}	0.99874	0.053079	0.010619	0.17402	0.185
{'fracByP0_01'}	0.00013554	0.010553	0	0	
{'fracByP10'}	0.087755	0.048541	0	0.0025063	0.004513
{'fracByP25'}	0.2241	0.14456	0	0.028727	0.03673
{'fracByP50'}	0.46426	0.38247	0	0.18419	0.1735
{'fracByP75'}	0.72225	0.68941	1	0.48388	0.4489
{'fracByP90'}	0.88703	0.88024	1	0.74634	0.7160
{'fracByP99_99'}	0.99995	1	1	0.99979	0.9993
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx					
Marital =0, kids =0, ybin =80 to 100					
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx					
xxx tb_outcomes: all stats xxx					

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

{'mean'}	}	89.313	45.289	0.28587	244.08	261.1
{'unweighted_sum'}	}	3.4629e+06	1909	1	26756	4.4356e+0
{'sd'}	}	5.7791	13.023	0.45183	194.1	192.8
{'coefofvar'}	}	0.064706	0.28755	1.5805	0.79522	0.7385
{'gini'}	}	0.037295	0.15841	0.6408	0.4125	0.4039
{'min'}	}	80.001	19	0	0	0
{'max'}	}	100	64	1	2377.1	2370.
{'pYiso'}	}	0	0	0.71413	0.020853	1.1851e-1
{'pYlso'}	}	0	0	0	0	0
{'pYgro'}	}	1	1	0.28587	0.97915	
{'pYisMINY'}	}	0	0.018972	0.71413	0.020853	1.1851e-1
{'pYisMAXY'}	}	3.7911e-06	0.02925	0.28587	1.1813e-15	
{'p0_01'}	}	80.012	19	0	0	0.7904
{'p10'}	}	81.54	25	0	29.898	43.52
{'p25'}	}	84.27	35	0	100.91	114.2
{'p50'}	}	88.922	48	0	205.07	224.4
{'p75'}	}	94.198	56	1	363.77	378.8
{'p90'}	}	97.585	61	1	525.49	536.5
{'p99_99'}	}	100	64	1	1281.9	1273.
{'fl_cov_y_all'}	}	33.398	2.1956	0.039297	150.26	159.6
{'fl_cor_y_all'}	}	1	0.029174	0.01505	0.13396	0.1432
{'fl_cov_age_ss'}	}	2.1956	169.59	-0.29823	1813.9	1849.
{'fl_cor_age_ss'}	}	0.029174	1	-0.050684	0.71759	0.7365
{'fl_cov_educ_ss'}	}	0.039297	-0.29823	0.20415	-12.356	-12.92
{'fl_cor_educ_ss'}	}	0.01505	-0.050684	1	-0.14089	-0.148
{'fl_cov_a_ss'}	}	150.26	1813.9	-12.356	37675	3741
{'fl_cor_a_ss'}	}	0.13396	0.71759	-0.14089	1	0.9994
{'fl_cov_ap_ss'}	}	159.68	1849.7	-12.922	37410	3719
{'fl_cor_ap_ss'}	}	0.14327	0.73653	-0.1483	0.99942	
{'fl_cov_MPC'}	}	-0.0031027	0.051957	0.0006815	0.63872	0.6402
{'fl_cor_MPC'}	}	-0.05432	0.40366	0.15261	0.33294	0.3358
{'fl_cov_Mass'}	}	-1.4164e-06	-8.0012e-06	-3.1745e-07	-9.5448e-05	-9.4333e-0
{'fl_cor_Mass'}	}	-0.083209	-0.20859	-0.23854	-0.16695	-0.1660
{'fl_cov_c_ss'}	}	15.988	-34.182	0.59583	378.89	341.3
{'fl_cor_c_ss'}	}	0.39229	-0.3722	0.187	0.27681	0.2509
{'fl_cov_y_head_inc'}	}	33.398	2.1956	0.039297	150.26	159.6
{'fl_cor_y_head_inc'}	}	1	0.029174	0.01505	0.13396	0.1432
{'fl_cov_y_spouse'}	}	0	0	0	0	
{'fl_cor_y_spouse'}	}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}		0.014829	0.0010034	1.7131e-05	0.066952	0.07114
{'fl_cor_yshr_nttxss'}		0.99914	0.030003	0.014763	0.13431	0.1436
{'fracByP0_01'}	}	0.00042007	0.0079591	0	0	5.0103e-0
{'fracByP10'}	}	0.090622	0.05099	0	0.0059303	0.008103
{'fracByP25'}	}	0.22976	0.15254	0	0.060679	0.05259
{'fracByP50'}	}	0.47206	0.40219	0	0.22278	0.212
{'fracByP75'}	}	0.72831	0.67181	1	0.53644	0.4926
{'fracByP90'}	}	0.88939	0.87436	1	0.78432	0.7455
{'fracByP99_99'}	}	1	1	1	0.99942	0.9994

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Marital =0, kids =0, ybin =100 to 1414.0634

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxx tb_outcomes: all stats xxx

OriginalVariableNames	y_all	age_ss	educ_ss	a_ss	ap_ss
{'mean'}	164.25	47.879	0.35462	603.16	641.9
{'unweighted_sum'}	1.6654e+08	1909	1	1.2935e+05	1.4733e+0

{'sd'}		74.664	11.785	0.4784	524.59	535.2
{'coefofvar'}	}	0.45456	0.24615	1.349	0.86973	0.8337
{'gini'}	}	0.20972	0.13265	0.54013	0.41585	0.4099
{'min'}	}	100	19	0	0	2.660
{'max'}	}	1413.7	64	1	7837.6	8386.
{'pYis0'}	}	0	0	0.64538	0.008464	
{'pYls0'}	}	0	0	0	0	
{'pYgr0'}	}	1	1	0.35462	0.99154	
{'pYisMINY'}	}	8.0846e-15	0.0084319	0.64538	0.008464	
{'pYisMAXY'}	}	9.784e-09	0.035671	0.35462	2.5784e-05	2.8187e-0
{'p0_01'}	}	100.01	19	0	0	8.390
{'p10'}	}	105.91	30	0	122.46	146.1
{'p25'}	}	116.38	40	0	239.18	290.4
{'p50'}	}	140.36	50	0	467.15	508.3
{'p75'}	}	184.67	58	1	807.24	835.4
{'p90'}	}	250.3	62	1	1175.1	1271.
{'p99_99'}	}	1005.7	64	1	6140.4	6451.
{'fl_cov_y_all'}	}	5574.6	82.616	3.9383	27029	2868
{'fl_cor_y_all'}	}	1	0.093888	0.11026	0.6901	0.7178
{'fl_cov_age_ss'}	}	82.616	138.9	-0.051378	3187.5	3233.
{'fl_cor_age_ss'}	}	0.093888	1	-0.0091126	0.51557	0.5126
{'fl_cov_educ_ss'}	}	3.9383	-0.051378	0.22886	1.1548	1.862
{'fl_cor_educ_ss'}	}	0.11026	-0.0091126	1	0.0046015	0.007275
{'fl_cov_a_ss'}	}	27029	3187.5	1.1548	2.7519e+05	2.8051e+0
{'fl_cor_a_ss'}	}	0.6901	0.51557	0.0046015	1	0.9990
{'fl_cov_ap_ss'}	}	28687	3233.7	1.8629	2.8051e+05	2.8647e+0
{'fl_cor_ap_ss'}	}	0.71786	0.51265	0.0072754	0.99906	
{'fl_cov_MPC'}	}	-0.0039422	0.067815	-2.0374e-06	1.5699	1.574
{'fl_cor_MPC'}	}	-0.0078548	0.85602	-0.00063355	0.44519	0.4376
{'fl_cov_Mass'}	}	-3.2407e-05	-4.7599e-07	-1.5824e-07	-0.00016835	-0.0001761
{'fl_cor_Mass'}	}	-0.36338	-0.033813	-0.27693	-0.26869	-0.2755
{'fl_cov_c_ss'}	}	2511	15.67	2.2388	14895	1549
{'fl_cor_c_ss'}	}	0.94083	0.037196	0.13092	0.79429	0.8098
{'fl_cov_y_head_inc'}	}	5574.6	82.616	3.9383	27029	2868
{'fl_cor_y_head_inc'}	}	1	0.093888	0.11026	0.6901	0.7178
{'fl_cov_y_spouse'}	}	0	0	0	0	
{'fl_cor_y_spouse'}	}	NaN	NaN	NaN	NaN	NaN
{'fl_cov_yshr_nttxss'}		0.64609	0.011641	0.00050086	3.1347	3.324
{'fl_cor_yshr_nttxss'}		0.90808	0.10365	0.10987	0.62707	0.6518
{'fracByP0_01'}	}	7.5135e-05	0.003346	0	0	1.097e-0
{'fracByP10'}	}	0.062666	0.056801	0	0.013271	0.01293
{'fracByP25'}	}	0.16403	0.16984	0	0.057688	0.06415
{'fracByP50'}	}	0.35792	0.40929	0	0.223	0.2196
{'fracByP75'}	}	0.60086	0.72112	1	0.50267	0.4758
{'fracByP90'}	}	0.79433	0.90509	1	0.70917	0.7129
{'fracByP99_99'}	}	0.99932	1	1	0.99885	0.9988

12.1.12 Store Aggregate To File

Store Several Files:

1. Overall Aggregate Statistics All Distribution
2. Aggregate Statistics Only for 18 to 64 year olds
3. Group Statistics by Kids
4. Group Statistics by Marital + Kids
5. Group Statistics by Marital + Kids + Income Bins

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

```

if (bl_save_csv)
    % All Stats All Ages
    mp_path = snw_mp_path('fan');
    spt_simu_results_csv = mp_path('spt_simu_results_csv');
    writetable(tb_dist_stats_all, [spt_simu_results_csv 'stats_all_allages.csv'], 'WriteRowNames', t
    % All Stats 18 to 64 Year old
    mp_path = snw_mp_path('fan');
    spt_simu_results_csv = mp_path('spt_simu_results_csv');
    writetable(tb_dist_stats_all_18to64, [spt_simu_results_csv 'stats_all_18t64.csv'], 'WriteRowName
    % Group by K: Kids only
    tb_store_stats_by_k = array2table(mt_store_stats_by_k, 'VariableNames', ...
        {'kids', 'married_mean' ...
        'age_mean', 'age_p50', 'educ_mean', ...
        'a_mean', 'a_p50', 'ap_mean', 'ap_p50', ...
        'y_all_mean', 'y_all_p50', ...
        'mpc_mean', 'mpc_p50', ...
        'mass',...
        'c_ss_mean', 'c_ss_p50', ...
        'y_head_inc_mean', 'y_spouse_mean'});
    mp_path = snw_mp_path('fan');
    spt_simu_results_csv = mp_path('spt_simu_results_csv');
    writetable(tb_store_stats_by_k, [spt_simu_results_csv 'stats_by_kids.csv']);
    % Group by MK: marry + kids only
    tb_store_stats_by_mk = array2table(mt_store_stats_by_mk, 'VariableNames', ...
        {'marital', 'kids', ...
        'age_mean', 'age_p50', 'educ_mean', ...
        'a_mean', 'a_p50', 'ap_mean', 'ap_p50', ...
        'y_all_mean', 'y_all_p50', ...
        'mpc_mean', 'mpc_p50', ...
        'mass',...
        'c_ss_mean', 'c_ss_p50', ...
        'y_head_inc_mean', 'y_spouse_mean'});
    mp_path = snw_mp_path('fan');
    spt_simu_results_csv = mp_path('spt_simu_results_csv');
    writetable(tb_store_stats_by_mk, [spt_simu_results_csv 'stats_by_marital_kids.csv']);
    % Group by MKY
    tb_store_stats_by_mky = array2table(mt_store_stats_by_mky, 'VariableNames', ...
        {'marital', 'kids', 'y_all_start', 'y_all_end', ...
        'age_mean', 'age_p50', 'educ_mean', ...
        'a_mean', 'a_p50', 'ap_mean', 'ap_p50', ...
        'y_all_mean', 'y_all_p50', ...
        'mpc_mean', 'mpc_p50', ...
        'mass',...
        'c_ss_mean', 'c_ss_p50', ...
        'y_head_inc_mean', 'y_spouse_mean'});
    mp_path = snw_mp_path('fan');
    spt_simu_results_csv = mp_path('spt_simu_results_csv');
    writetable(tb_store_stats_by_mky, [spt_simu_results_csv 'stats_by_marital_kids_20kincbins.csv']);
end

```

12.1.13 Store Key Stats to Compare to Key US Distributional Statistics

Earning, income and Wealth.

Income = interest earnings + Social Security + labor income + spousal income. This is equal to y_all.

Earnings = labor income + spousal income.

```
% Income Variable
if (min(abs(total_inc_VFI*58.056 - y_all), [], 'all')>0)
```

```

    error('someothing is wrong, total_inc_VFI should be equal to y_all');
end
income = y_all;
% Earning variable
% earn*fl_earn_ratio generated earn_VFI
earning = (mp_valpol_more_ss('earn_VFI') + spouse_inc_VFI)*58.056;
% Wealth Varaible
wealth = a_ss;

```

Generate Key Statistics for these three variables only, distributional Statistics Overall All Ages:

```

% construct input data
income_grp = income(min_age:82, :, :, :, :, :, :);
earning_grp = earning(min_age:82, :, :, :, :, :, :);
wealth_grp = wealth(min_age:82, :, :, :, :, :, :);
Phi_true_grp = Phi_true_1(min_age:82, :, :, :, :, :, :);

mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
mp_cl_ar_xyz_of_s('earning') = {earning_grp(:, zeros(1));
mp_cl_ar_xyz_of_s('income') = {income_grp(:, zeros(1));
mp_cl_ar_xyz_of_s('wealth') = {wealth_grp(:, zeros(1));
mp_cl_ar_xyz_of_s('earninglog') = {log(earning_grp(:)), zeros(1)};
mp_cl_ar_xyz_of_s('incomelog') = {log(income_grp(:)), zeros(1)};
mp_cl_ar_xyz_of_s('wealthlog') = {log(wealth_grp(:)), zeros(1)};
mp_cl_ar_xyz_of_s('ar_st_y_name') = ["earning", "income", "wealth", "earninglog", "incomelog", "wealthlog"];

% controls
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('ar_fl_percentiles') = [20 30 40 60 50 80 90 95 99];
mp_support('bl_display_final') = true;
mp_support('bl_display_detail') = false;
mp_support('bl_display_drvm2outcomes') = false;
mp_support('bl_display_drvstats') = false;
mp_support('bl_display_drvm2covcor') = false;

% Call Function
mp_cl_mt_xyz_of_s = ff_simu_stats(Phi_true_grp(:)/sum(Phi_true_grp,'all'), mp_cl_ar_xyz_of_s, mp_sup

```

xxx tb_outcomes: all stats xxx

OriginalVariableNames	earning	income	wealth	earninglog	incomelog	w
{'mean'}	72.136	84.974	245.22	-Inf	4.1042	
{'unweighted_sum'}	9.5943e+07	7.9255e+09	1.2935e+05	-Inf	1.1455e+08	
{'sd'}	80.749	84.549	391.42	NaN	0.81216	
{'coefofvar'}	1.1194	0.995	1.5962	NaN	0.19789	
{'gini'}	0.51369	0.44243	0.68023	NaN	0.11243	
{'min'}	0	2.2124	0	-Inf	0.79408	
{'max'}	2640	2953.5	7837.6	7.8785	7.9907	
{'pYis0'}	0.10578	0	0.12285	0	0	
{'pYls0'}	0	0	0	0.10695	0	
{'pYgr0'}	0.89422	1	0.87715	0.89305	1	
{'pYisMINY'}	0.10578	6.774e-07	0.12285	0.10578	6.774e-07	
{'pYisMAXY'}	1.5964e-10	1.671e-12	6.0119e-06	1.5964e-10	1.671e-12	
{'p20'}	15.969	29.216	3.7372	2.7707	3.3747	
{'p30'}	29.464	38.184	15.308	3.3832	3.6424	
{'p40'}	40.761	48.225	39.794	3.7077	3.8759	
{'p60'}	65.423	74.426	146.89	4.1809	4.3098	
{'p50'}	52.252	59.948	82.04	3.9561	4.0935	

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

{'p80'}	}	108.96	122.39	413.31	4.691	4.8072
{'p90'}	}	159.7	176.61	729.18	5.0733	5.1739
{'p95'}	}	211.84	233.69	979.69	5.3558	5.454
{'p99'}	}	356.31	398.22	1773.5	5.8758	5.987
{'fl_cov_earning'}	}	6520.5	6671.7	8382.5	NaN	53.875
{'fl_cor_earning'}		1	0.97721	0.26521	NaN	0.82149
{'fl_cov_income'}	}	6671.7	7148.6	15059	NaN	57.878
{'fl_cor_income'}		0.97721	1	0.45504	NaN	0.84286
{'fl_cov_wealth'}	}	8382.5	15059	1.5321e+05	NaN	141.72
{'fl_cor_wealth'}		0.26521	0.45504	1	NaN	0.4458
{'fl_cov_earninglog'}		NaN	NaN	NaN	NaN	NaN
{'fl_cor_earninglog'}		NaN	NaN	NaN	NaN	NaN
{'fl_cov_incomelog'}		53.875	57.878	141.72	NaN	0.65961
{'fl_cor_incomelog'}		0.82149	0.84286	0.4458	NaN	1
{'fl_cov_wealthlog'}		NaN	NaN	NaN	NaN	NaN
{'fl_cor_wealthlog'}		NaN	NaN	NaN	NaN	NaN
{'fracByP20'}		0.012671	0.04827	0.00074821	NaN	0.14532
{'fracByP30'}		0.044498	0.08795	0.0041711	NaN	0.23096
{'fracByP40'}		0.093262	0.13869	0.016749	NaN	0.32262
{'fracByP60'}		0.23895	0.28076	0.095501	NaN	0.52207
{'fracByP50'}		0.15762	0.20209	0.045325	NaN	0.41971
{'fracByP80'}		0.47178	0.50479	0.32852	NaN	0.74357
{'fracByP90'}		0.65353	0.6766	0.56651	NaN	0.86486
{'fracByP95'}		0.78022	0.79527	0.70071	NaN	0.92947
{'fracByP99'}		0.92468	0.93132	0.90524	NaN	0.98459

```

tb_dist_stats_all = mp_cl_mt_xyz_of_s('tb_outcomes');
% Select columns
tb_dist_stats_all_save = tb_dist_stats_all(1:3,:);
ar_st_columns = ["coeofvar", "gini", "varianceoflog", ...
    "p99p50ratio", "p90p50ratio", "meantomedian", "p50p30ratio", ...
    "fracP0toP20", "fracP20toP40", "fracP40toP60", "fracP60toP80", "fracP80toP100", ...
    "fracP90toP95", "fracP95toP99", "fracP99toP100"];

varianceoflog = tb_dist_stats_all{4:6,"sd"}.^2;

p99p50ratio = tb_dist_stats_all_save{:, "p99"}. / tb_dist_stats_all_save{:, "p50"};
p90p50ratio = tb_dist_stats_all_save{:, "p90"}. / tb_dist_stats_all_save{:, "p50"};
meantomedian = tb_dist_stats_all_save{:, "mean"}. / tb_dist_stats_all_save{:, "p50"};
p50p30ratio = tb_dist_stats_all_save{:, "p50"}. / tb_dist_stats_all_save{:, "p30"};
fracP0toP20 = tb_dist_stats_all_save{:, "fracByP20"};
fracP20toP40 = tb_dist_stats_all_save{:, "fracByP40"} - tb_dist_stats_all_save{:, "fracByP20"};
fracP40toP60 = tb_dist_stats_all_save{:, "fracByP60"} - tb_dist_stats_all_save{:, "fracByP40"};
fracP60toP80 = tb_dist_stats_all_save{:, "fracByP80"} - tb_dist_stats_all_save{:, "fracByP60"};
fracP80toP100 = 1 - tb_dist_stats_all_save{:, "fracByP80"};

fracP90toP95 = tb_dist_stats_all_save{:, "fracByP95"} - tb_dist_stats_all_save{:, "fracByP90"};
fracP95toP99 = tb_dist_stats_all_save{:, "fracByP99"} - tb_dist_stats_all_save{:, "fracByP95"};
fracP99toP100 = 1 - tb_dist_stats_all_save{:, "fracByP99"};

tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, varianceoflog, 'Before', 'gini');
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, p99p50ratio);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, p90p50ratio);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, meantomedian);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, p50p30ratio);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP0toP20);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP20toP40);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP40toP60);

```

```

tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP60toP80);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP80toP100);

tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP90toP95);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP95toP99);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP99toP100);
disp(tb_dist_stats_all_save(:, ar_st_columns));



|         | coefofvar | gini    | varianceoflog | p99p50ratio | p90p50ratio | meantomedian |
|---------|-----------|---------|---------------|-------------|-------------|--------------|
|         | -----     | -----   | -----         | -----       | -----       | -----        |
| earning | 1.1194    | 0.51369 | NaN           | 6.819       | 3.0563      | 1.3805       |
| income  | 0.995     | 0.44243 | 0.65961       | 6.6427      | 2.946       | 1.4174       |
| wealth  | 1.5962    | 0.68023 | NaN           | 21.618      | 8.8881      | 2.989        |



% Core Stats Table
if (bl_save_csv)
    mp_path = snw_mp_path('fan');
    spt_simu_results_csv = mp_path('spt_simu_results_csv');
    writetable(tb_dist_stats_all_save(:, ar_st_columns), [spt_simu_results_csv 'stats_all_allages_vr'
end

Statistics overall distributionally for 18 to 64 year olds.

% construct input data
income_grp = income(min_age:max_age, :, :, :, :, :, :);
earning_grp = earning(min_age:max_age, :, :, :, :, :, :);
wealth_grp = wealth(min_age:max_age, :, :, :, :, :, :);
Phi_true_grp = Phi_true_1(min_age:max_age, :, :, :, :, :, :);

mp_cl_ar_xyz_of_s = containers.Map('KeyType','char', 'ValueType','any');
mp_cl_ar_xyz_of_s('income') = {income_grp(:, zeros(1));
mp_cl_ar_xyz_of_s('earning') = {earning_grp(:, zeros(1));
mp_cl_ar_xyz_of_s('wealth') = {wealth_grp(:, zeros(1));
mp_cl_ar_xyz_of_s('earninglog') = {log(earning_grp(:)), zeros(1)};
mp_cl_ar_xyz_of_s('incomelog') = {log(income_grp(:)), zeros(1)};
mp_cl_ar_xyz_of_s('wealthlog') = {log(wealth_grp(:)), zeros(1)};
mp_cl_ar_xyz_of_s('ar_st_y_name') = ["earning", "income", "wealth", "earninglog", "incomelog", "weal

% controls
mp_support = containers.Map('KeyType','char', 'ValueType','any');
mp_support('ar_fl_percentiles') = [20 30 40 60 50 80 90 95 99];
mp_support('bl_display_final') = true;
mp_support('bl_display_detail') = false;
mp_support('bl_display_drvm2outcomes') = false;
mp_support('bl_display_drvstats') = false;
mp_support('bl_display_drvm2covcor') = false;

% Call Function
mp_cl_mt_xyz_of_s = ff_simu_stats(Phi_true_grp(:)/sum(Phi_true_grp,'all'), mp_cl_ar_xyz_of_s, mp_sup

xxx tb_outcomes: all stats xxx


| OriginalVariableNames | earning   | income     | wealth     | earninglog | incomelog | w     |
|-----------------------|-----------|------------|------------|------------|-----------|-------|
|                       | -----     | -----      | -----      | -----      | -----     | ----- |
| {'mean'}              | 87.466    | 95.246     | 194.5      | 4.1711     | 4.2425    |       |
| {'unweighted_sum'}    | 9.394e+07 | 7.7487e+09 | 1.2935e+05 | 1.5445e+06 | 1.116e+08 |       |
| {'sd'}                | 82.434    | 89.631     | 344.5      | 0.76834    | 0.79264   |       |
| {'coefofvar'}         | 0.94247   | 0.94104    | 1.7712     | 0.1842     | 0.18683   |       |


```

12.1. 2019 FULL STATES MPC AND DISTRIBUTIONAL STATISTICS BY MARITAL, KIDS, AND INCOME GROUP

{'gini'}	}	0.417	0.42428	0.71579	0.10382	0.1055
{'min'}	}	2.2124	2.2124	0	0.79408	0.79408
{'max'}	}	2640	2953.5	7837.6	7.8785	7.9907
{'pYiso'}	}	0	0	0.14627	0	0
{'pYls0'}	}	0	0	0	0	0
{'pYgr0'}	}	1	1	0.85373	1	1
{'pYisMINY'}	}	8.617e-07	8.6135e-07	0.14627	8.617e-07	8.6135e-07
{'pYisMAXY'}	}	2.0299e-10	2.1248e-12	5.4766e-06	2.0299e-10	2.1248e-12
{'p20'}	}	34.093	35.624	0.80724	3.5291	3.573
{'p30'}	}	43.249	45.828	6.458	3.767	3.8249
{'p40'}	}	52.993	56.888	29.898	3.9702	4.0411
{'p60'}	}	77.857	85.184	100.91	4.3549	4.4448
{'p50'}	}	64.26	69.57	51.664	4.1629	4.2423
{'p80'}	}	124.43	137.12	318.35	4.8237	4.9209
{'p90'}	}	175.33	192.9	588.48	5.1667	5.2621
{'p95'}	}	227.34	250.34	890.69	5.4265	5.5228
{'p99'}	}	384.15	427.18	1640.6	5.951	6.0572
{'fl_cov_earning'}	}	6795.4	7319.6	13105	53.1	53.884
{'fl_cor_earning'}	}	1	0.99065	0.46144	0.83837	0.82467
{'fl_cov_income'}	}	7319.6	8033.6	17852	58.043	59.852
{'fl_cor_income'}	}	0.99065	1	0.57814	0.84283	0.84246
{'fl_cov_wealth'}	}	13105	17852	1.1868e+05	123.58	149.2
{'fl_cor_wealth'}	}	0.46144	0.57814	1	0.46687	0.5464
{'fl_cov_earninglog'}		53.1	58.043	123.58	0.59034	0.6043
{'fl_cor_earninglog'}		0.83837	0.84283	0.46687	1	0.99226
{'fl_cov_incomelog'}		53.884	59.852	149.2	0.6043	0.62827
{'fl_cor_incomelog'}		0.82467	0.84246	0.5464	0.99226	1
{'fl_cov_wealthlog'}		NaN	NaN	NaN	NaN	NaN
{'fl_cor_wealthlog'}		NaN	NaN	NaN	NaN	NaN
{'fracByP20'}	}	0.053802	0.050961	0.00014055	0.14882	0.14762
{'fracByP30'}	}	0.098055	0.093694	0.0021143	0.23646	0.23488
{'fracByP40'}	}	0.153	0.14753	0.015697	0.3292	0.32764
{'fracByP60'}	}	0.30069	0.29468	0.079605	0.52874	0.52766
{'fracByP50'}	}	0.21981	0.21374	0.034043	0.42667	0.42529
{'fracByP80'}	}	0.52452	0.52079	0.28918	0.74816	0.7478
{'fracByP90'}	}	0.69236	0.69054	0.51495	0.86758	0.86757
{'fracByP95'}	}	0.80576	0.80501	0.69371	0.93096	0.93099
{'fracByP99'}	}	0.93293	0.93437	0.90041	0.98483	0.98492

```

tb_dist_stats_all = mp_cl_mt_xyz_of_s('tb_outcomes');
% Select columns
tb_dist_stats_all_save = tb_dist_stats_all(1:3,:);
ar_st_columns = ["coeofvar", "gini", "varianceoflog", ...
    "p99p50ratio", "p90p50ratio", "meantomedian", "p50p30ratio", ...
    "fracP0toP20", "fracP20toP40", "fracP40toP60", "fracP60toP80", "fracP80toP100", ...
    "fracP90toP95", "fracP95toP99", "fracP99toP100"];

varianceoflog = tb_dist_stats_all{4:6,"sd"}.^2;

p99p50ratio = tb_dist_stats_all_save{:, "p99"}. / tb_dist_stats_all_save{:, "p50"};
p90p50ratio = tb_dist_stats_all_save{:, "p90"}. / tb_dist_stats_all_save{:, "p50"};
meantomedian = tb_dist_stats_all_save{:, "mean"}. / tb_dist_stats_all_save{:, "p50"};
p50p30ratio = tb_dist_stats_all_save{:, "p50"}. / tb_dist_stats_all_save{:, "p30"};
fracP0toP20 = tb_dist_stats_all_save{:, "fracByP20"};
fracP20toP40 = tb_dist_stats_all_save{:, "fracByP40"} - tb_dist_stats_all_save{:, "fracByP20"};
fracP40toP60 = tb_dist_stats_all_save{:, "fracByP60"} - tb_dist_stats_all_save{:, "fracByP40"};
fracP60toP80 = tb_dist_stats_all_save{:, "fracByP80"} - tb_dist_stats_all_save{:, "fracByP60"};
fracP80toP100 = 1 - tb_dist_stats_all_save{:, "fracByP80"};

```

```

fracP90toP95 = tb_dist_stats_all_save{:, "fracByP95"} - tb_dist_stats_all_save{:, "fracByP90"};
fracP95toP99 = tb_dist_stats_all_save{:, "fracByP99"} - tb_dist_stats_all_save{:, "fracByP95"};
fracP99toP100 = 1 - tb_dist_stats_all_save{:, "fracByP99"};

tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, varianceoflog, 'Before', 'gini');
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, p99p50ratio);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, p90p50ratio);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, meantomedian);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, p50p30ratio);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP0toP20);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP20toP40);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP40toP60);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP60toP80);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP80toP100);

tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP90toP95);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP95toP99);
tb_dist_stats_all_save = addvars(tb_dist_stats_all_save, fracP99toP100);
disp(tb_dist_stats_all_save(:, ar_st_columns));

```

	coeofvar	gini	varianceoflog	p99p50ratio	p90p50ratio	meantomedian
	-----	-----	-----	-----	-----	-----
earning	0.94247	0.417	0.59034	5.978	2.7285	1.3611
income	0.94104	0.42428	0.62827	6.1403	2.7727	1.3691
wealth	1.7712	0.71579	NaN	31.755	11.391	3.7648

```

% Core Stats Table
if (bl_save_csv)
    mp_path = snw_mp_path('fan');
    spt_simu_results_csv = mp_path('spt_simu_results_csv');
    writetable(tb_dist_stats_all_save(:, ar_st_columns), [spt_simu_results_csv 'stats_all_18t64_vrrc']);
end

```

Appendix A

Index and Code Links

A.1 Introduction links

1. Household Problem and Distributions: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Summarize the household's dynamic programming problem and the distributions across heterogeneous households groups.
2. Values of Checks Conditional on 2019 Information: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Summarize the computation of expectations relevant for planning objectives given information available in 2019.
3. The Welfare Check Planning Problem: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Summarize several allocation problem that condition allocations on income, marital status, the number of children less than 18, and possibly age.

A.2 Parameters links

1. Model Parameters: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Model parameters, transition matrices, permanent heterogeneities.
 - **PrjOptiSNW:** [snw_mp_param\(\)](#)
2. Model Controls Parameters: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Parameters to control display options etc.
 - **PrjOptiSNW:** [snw_mp_control\(\)](#)

A.3 Solving the Dynamic Life Cycle Problem links

1. Policy and Value Functions Dynamic Life Cycle Vectorized Bisection: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Solving for policy and value functions from 18 to 100 years of age, at 1 year interval.
 - Households face persistent productivity shocks for household heads, stochastic shocks for spousal income, exogenous children under age 17 transition probability, and age-specific household-head survival probabilities.
 - The household can have up to four children under age 17, and has permanent heterogeneity in marital status and education types.
 - Problem solved for exact savings choices using [vectorized bisection](#) from MEconTools.
 - **PrjOptiSNW:** [snwx_vfi_bisec_vec\(\)](#)

A.4 Alternative Value Function Solution Testing links

1. Small Test Looped Minimizer Routine to Solve for Exact Savings Choices: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Solve for the exact savings choices using matlab minimizer in an iterative loop.
 - The code demonstrates the solution structure. We use [snwx_vfi_bisec_vec\(\)](#) with [vectorized bisection](#) for working implementations.
 - Due to speed, only show testing results at small grid without spousal shocks.

- **PrjOptiSNW:** [snw_vfi_main\(\)](#)
2. **Small Test Looped over States Grid Search Solution:** [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - The savings choice grid is the same as the savings states grid. Solve for optimal savings choices using grid-search. Loop over the state space, at each state-space point, vectorized optimization.
 - Our problem requires very high precision to solve for the marginal gains to households from each increment of welfare checks. We rely on the exact solution method from [snwx_vfi_bisec_vec\(\)](#) for the working code.
 - Due to speed, only show testing results at small grid without spousal shocks.
 - **PrjOptiSNW:** [snw_vfi_main_grid_search\(\)](#)
 3. **Small Test Vectorized Bisection Solve for Exact Savings Choices:** [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Vectorized bisection exact solution code tested with small grid to compare to alternative solution methods.
 - Small grid without spousal shocks.
 - **PrjOptiSNW:** [snwx_vfi_bisec_vec\(\)](#)
 4. **Small Test Spousal Shocks Test Vectorized Bisection Solve for Exact Savings Choices:** [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Vectorized bisection exact solution code tested with small grid to compare to alternative solution methods.
 - Small grid with spousal shocks. There are three shocks: persistent household head income shock, i.i.d. spousal income shock, and persistent kids count transition shocks.
 - **PrjOptiSNW:** [snwx_vfi_bisec_vec\(\)](#)

A.5 Solution with Unemployment links

1. **Policy and Value Functions Dynamic Life Cycle if Unemployed:** [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Solving the dynamic programming problem conditional on having an one period unemployment shock.
 - There is an unemployment shock in 2020. We first solve for the policy and value functions without the unemployment shock.
 - Using the value function from the world without the 2020 covid unemployment shock as future values, we solve for optimal choices in 2020 given a COVID unemployment shock.
 - The COVID shock lowers the realization of household's stochastic income process proportionally, but the lost income might be replenished by unemployment benefits up to 100 percent. Unemployment benefits have to be paid for by taxes.
 - **PrjOptiSNW:** [snwx_vfi_bisec_vec\(\)](#)

A.6 Household Life Cycle Distribution links

1. **Assets and Demographic Distributions with Continuous Exact Savings Choices (Loop):** [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Simulate the life cycle distribution of assets, consumptions, and demographic patterns up to age 100, given exogenous initial distributions at age 18. Solves for budget clearing tax rates given distributional results. Uses vectorized bisection to solve for exact savings choices, looped distribution code.
 - **PrjOptiSNW:** [snw_ds_main\(\)](#)
2. **Assets and Demographic Distributions with Continuous Exact Savings Choices (Vectorized):** [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Simulate the life cycle distributions This is the fully vectorized version of [snw_ds_main\(\)](#).
 - **PrjOptiSNW:** [snw_ds_main_vec\(\)](#)
3. **Assets and Demographic Distributions with Grid Search:** [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Grid search solution using grid search for savings choices, the savings state-space grid is the same as the savings choice-grid. Exact choice solution from [snw_ds_main\(\)](#) generates significantly smoother distributions.
 - **PrjOptiSNW:** [snw_ds_main_grid_search\(\)](#)

A.7 Value of Each Check links

1. Marginal Gain Per Check 2020 Employed: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Evaluate the marginal gain per check in 2020 if household head is employed.
 - Solve for the increase in savings that is equivalent to the impact of an additional check on a household's resource available in 2020, given tax and interest rates considerations.
 - PrjOptiSNW: [snw_a4chk_wrk_bisec_vec\(\)](#)
2. Marginal Gain Per Check 2020 Unemployed: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Evaluate the marginal gain per check in 2020 if household head is unemployed.
 - Solve for the increase in savings that is equivalent to the impact of an additional check on a household's resource available in 2020, given tax and interest rates considerations.
 - PrjOptiSNW: [snw_a4chk_unemp_bisec_vec\(\)](#)

A.8 Outcomes Full State Space with Savings, Shocks and Education links

1. Value in 2020 Given Age, Savings, Shocks, Kids, Education and Marriage: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Expected value and expected consumption from 2020 for a household given at a particular age (18-100), with a particular savings level, at a particular combination of household head and spouse income shocks, with 0 to 4 children, high or low Education status, and married or not married.
 - This uses the unemployment probability and generates the average value given the probability of the unemployment state that is dependent on the state-space.
 - PrjOptiSNW: [snw_evuvw20_jaeemk\(\)](#)
2. Expected Value in 2019 Given Age, Savings, Shocks, Kids, Education and Marriage: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Expected value and expected consumption from 2019 for a household at a particular age (18-99), savings level, shocks combinations, kids/education/marriage status, given 2019 optimal savings choices, income shock transition probability as well as household children count transition probabilities.
 - PrjOptiSNW: [snw_evuvw19_jaeemk\(\)](#)

A.9 Expectations Given Income, Age, Kids and Marital Status links

1. Expected Value from 2019 Given Age, Kids, Income and Marriage: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Expected Value from 2019 Given Age, Kids, Income and Marriage.
 - Each 2019 income group consists of individuals with varying productivity shocks, savings, and from lower and higher education groups.
 - PrjOptiSNW: [snw_evuvw19_jmky\(\)](#)
2. Expected Value from 2019 Given Age, Kids, Income and Marriage for All Checks: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Expected Value from 2019 Given Age, Kids, Income and Marriage for All Checks.
 - This is the gateway function that solves policy functions, derive distributions, computes value in 2020 with and without unemployment shocks with varying check levels, derives 2019 planner expected values given household optimization and shocks, and finds the mass of individuals in different income/age/marital-status bins, and saves the simulated value of check results for the planner.
 - PrjOptiSNW: [snw_evuvw19_jmky_allchecks\(\)](#)

A.10 Taxes links

1. Solve for Budget Clearing Tax Rates: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Given welfare checks and unemployment insurance costs, solve for tax rate that clears the budget given household resource availability.
 - PrjOptiSNW: [snw_find_tax_rate\(\)](#), [snw_tax_hh\(\)](#)

A.11 Calibration links

1. Calibrate Discount Factor and Normalize GDP: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - We calibrate the model so that the Asset/Savings/Capital to GDP/Income ratio is 3.
 - We normalize the model so that median household income is equal to 1 in the model.
 - PrjOptiSNW: [*snw_calibrate_beta_norm_gdp\(\)*](#)

A.12 Summary Statistics links

1. Distributional Statistics by Household Structure and Income Groups: [mlx](#) | [m](#) | [pdf](#) | [html](#)
 - Summarize overall model distributional and inequality statistics from covid-less times.
 - Statistics, including first check MPC, by marital status, children count, and income groups.
 - See [*snwx_evuvw19_jmky_mpc_allocated_m*](#) for summarizing function over optimal allocation results.

Bibliography

- Nygård, V., Sørensen, B., and Wang, F. (2020). Optimal allocation of the covid-19 stimulus checks.
- 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.