Online Appendix

Heterogeneous Agent Models in Asset Pricing: The Dynamic Programming Approach and the Finite Difference Method

1 The Finite Difference Method

In this appendix, we provide details of the solution method.

1.1 Solution Method

We use the implicit method since it has better convergency properties than the explicit one (e.g., Candler, 2001; Achdou et al., 2022).

1.1.1 The Implicit Method: The Definition

Under the implicit method, the discretized HJB equation for $k = \{1, 2\}$ with the upwind scheme is given by:

$$\begin{aligned}\frac{1}{1-\gamma_k}\frac{A_{k,i}^{n+1}-A_{k,i}^{n}}{\Delta t}+\rho\frac{A_{k,i}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,i}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+\left(\frac{\psi_i^2}{2\gamma_k}+r_i\right)A_{k,i}^{n+1}\\ &+ \tilde{a}_{ki,F}^{n}\left(A_{k,Y}\right)_{i,F}^{n+1}+\tilde{a}_{ki,B}^{n}\left(A_{k,Y}\right)_{i,B}^{n+1}+\frac{(\sigma Y_i)^2}{2(1-\gamma_k)}\left(A_{k,YY}\right)_i^{n+1}\end{aligned} \tag{1}$$

The Eq. (23) deserves two comments. First, one can think to evaluate $\frac{\gamma_k\left(A_{k,i}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}$ at $n+1$ instead of n. The argument for not doing that is that the system would turn out non-linear because of the following term: $\left(A_{k,i}^{n+1}\right)^{1-1/\gamma_k}$. Second, an alternative way is to factorize $A_{k,i}$ from the first two elements of the right side of Eq. (23) and evaluate the resulting coefficient at n and the term $A_{k,i}$ at $n+1$ as follows.

$$\frac{\gamma_k\left(A_{k,i}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+\left(\frac{\psi_i^2}{2\gamma_k}+r_i\right)A_{k,i}^{n+1}\equiv\left(\frac{\gamma_k\left(A_{k,i}^{n}\right)^{-1/\gamma_k}}{1-\gamma_k}+\left(\frac{\psi_i^2}{2\gamma_k}+r_i\right)\right)A_{k,i}^{n+1}$$

1.1.2 The Implicit Method: The Algorithm

In this section, we explain the algorithm of the implicit method to solve the PDE of A_k and S. The steps are the following:

- **Step 1.** Parameters
- **Step 2.** Discretization
- **Step 3.** Preliminaries for iteration of A_k
- **Step 4.** Initial guess of A_k
- **Step 5.** Iteration of A_k
  5.1 Initial point of A_k
  5.2 Finite difference
  5.3 Upwind scheme
  5.4 Discretization of PDE system
  5.5 Update of A_k
  5.6 The optimal A_k

These steps are implemented in the m-file *Main_Ak.m*. We explain these steps carefully in the following paragraphs. We will start solving the HJB for agent 1: A_1

Step 1. Parameters. First, we set up the value of the model's parameters related to preferences and financial assets.

```
%% STEP 1: Parameters
% A. Preferences
rho    = 0.1;   % impatience rate in discount factor e^(-rho*t)
g1 = 0.8;   % g1=gamma1 = RRA of agent 1 (more risk averse)
g2 = 0.5;   % g2=gamma2 = RRA of agent 2 (less risk averse)

gk = g1;    % gk=gammak

% B. Exogeneous State Variable Dynamic (Y)
mu      = 0.05; %E[dY/Y]
sigma   = 0.3;  %Volatility term of dY/Y

% C. Weights in Utility Function of Social Planner (U)
lambda = 1/3; % weight of utility function of agent 1 (gamma_1) in U
a = (1-lambda)/lambda;
```

Step 2. Discretization. We next discretize the state variable Y using a structured grid.

```
% A. State space: structured grid
Ymax = 100;
Ymin = 1;
I = 500;                % N of points in the grid: I + 1
deltaY = (Ymax-Ymin)/I; % the distance between grid points
  Y = Ymin:deltaY:Ymax; %the vector of the state variable (grid)
```

Step 3. Preliminaries for iteration of V. We create a matrix called "Akmatrix" to store the value function Ak for every iteration until it converges.

```
%% STEP 3: Preliminaries for iteration of Ak
% A. Storage of Ak for every iteration
Akmatrix = [];

% B. Variables from Social Planner solution:
% Consumption
c10 = 0.1; %initial point of agent1's consumption
for i=1:I+1
    c1(i) = fsolve(@(c1) c_op(c1,a,g1,g2,Y(i)),c10); %c1
end
c2 = Y - c1; %c2

% Interest Rate (r) and Price of Risk (psi)
a1 = c2*g1 + c1*g2; % aux1
a2 = -c1*(g2^2)*(1+g1) - c2*(g1^2)*(1+g2); % aux2

r   = rho + (mu*Y)*(g1*g2)./a1 + (( (sigma*Y).^2 )/2).*(g1*g2*a2./a1.^3);
psi = (sigma*Y)*(g1*g2)./a1;
```

It is important to mention that the function **c_op(c1,a,g1,g2,Y(i)),c10)** contains the nonlinear equation of c_1 which comes from the Social Planner solution (see Eq. **??**), and the Matlab function "fsolve" is used to solve "c_op" and get the optimal c_1.

Step 4. Initial guess of A_k. we then define the initial guess of the value function *for all the state points.* This will be the initial value of A_k for the first iteration in step 5. So, we need to define

$$A_k^0 = \left(A_{k,1}^0, A_{k,2}^0, ..., A_{k,I+1}^0\right)$$

We suppose that A_k is an increasing function in the state variable, so we use any concave function of Y as the initial value of the value function. So, I assume that

$$A_k^0 = \sqrt{Y},$$

which is implemented in Matlab as

```
%% STEP 4: INITIAL GUESS of Ak (for every point of the state var)
% A. Initial guess of "Ak"
    % Ak = [Ak_1, Ak_2, ..., Ak_I], k={1,2}
    ak0 = Y.^0.5; % Value function of agent k
    ak = ak0;     % row-vector of (I+1) columns
```

Step 5. Iteration of A_k. The PDE of A_k is solved by *iteration* until A_k converges. That means that for every n (iteration), we find A_k^{n+1}. For instance, given A_k^0 and for $n = 0$ we find A_k^1, then given A_k^1 and for $n = 1$ we find A_k^2, and so on. Therefore, in each iteration, we find A_k. The process finishes when A_k of the $n + 1-$th iteration is almost the same as the $n-$th iteration. To implement this step, we first set up the number of iterations, 1000, and the criterion to stop the iteration and get the solution of the PDE system, A_k.

```
%% STEP 5: Iteration of Ak
maxit= 1000;
crit=10^(-6);  %the criterion to stop iteration and
               %get the solution of "Ak"
deltat = 1000; %time length (from Achdou et al (2022))
```

We then construct a loop to solve the discretized PDE system of A_k with the upwind scheme as follows.

```
for n=1:maxit
    Step 5.1
    Step 5.2
    Step 5.3
    Step 5.4
    Step 5.5
    Step 5.6
end
```

Now, we provide details about this loop.

5.1 *Initial point of A_k.*
For the first iteration, we consider that A_k takes the *initial guess of A_k* defined in step 4. we also save A_k for every iteration in the matrix "Akmatrix."

```
%% STEP-5.1: Initial point of Ak
    Ak = ak;
    Akmatrix = [Akmatrix; Ak]; %We save the initial Ak of every iteration
```

5.2 *Finite difference (forward/backward difference approximation).*
We then calculate the forward difference taking into account the boundaries. The state grid has two boundary points: $Y = Y_{min}$ and $Y = Y_{max}$. In these two boundaries, some problems could appear when we calculate the forward, backward, and central differences.

a. *Forward Difference*

– Boundary: Ymin

$$(A_{k,Y})_{1,F}^{n+1} = \frac{A_{k,2}^{n+1} - A_{k,1}^{n+1}}{\Delta Y}, \quad For \quad i = 1 \Rightarrow Ymin = Y_1 \tag{2}$$

- Computational points

$$(A_{k,Y})^{n+1}_{i,F} = \frac{A^{n+1}_{k,i+1} - A^{n+1}_{k,i}}{\Delta Y}, \quad For \quad i = 2, \cdots, I \tag{3}$$

- Boundary: Ymax

$$(A_{k,Y})^{n+1}_{I+1,F} = \frac{A^{n+1}_{k,I+2} - A^{n+1}_{k,I+1}}{\Delta Y}, \quad For \quad i = I+1 \Rightarrow Ymax = Y_{I+1} \tag{4}$$

When we use the forward difference, a problem appears only in the node **Ymax**, where $A^{n+1}_{k,I+2}$ is not known because it is outside of the grid ($i = 1, \cdots, I+1$). The point $I+2$ is typically named the ghost node since it is outside the grid but is needed to calculate difference approximations. In this case, we assume

$$A^{n+1}_{k,I+2} = A^{n+1}_{k,I+1} \tag{5}$$

This implies

$$(A_{k,Y})^{n+1}_{I+1,F} = 0 \tag{6}$$

b. *Backward Difference*

- Boundary: Ymin

$$(A_{k,Y})^{n+1}_{1,B} = \frac{A^{n+1}_{k,1} - A^{n+1}_{k,0}}{\Delta Y}, \quad For \quad i = 1 \Rightarrow Ymin = Y_1 \tag{7}$$

However, this approximation cannot be calculated because $A^{n+1}_{k,0}$ is outside of the grid, and hence it is unknown. Since we do not have a grid point at $i = 0$, this point is also a ghost node. In this case, we assume

$$A^{n+1}_{k,0} = A^{n+1}_{k,1}. \tag{8}$$

This implies

$$(A_{k,Y})^{n+1}_{1,B} = 0 \tag{9}$$

- Computational points

$$(A_{k,Y})^{n+1}_{i,B} = \frac{A^{n+1}_{k,i} - A^{n+1}_{k,i-1}}{\Delta Y}, \quad For \quad i = 2, \cdots, I \tag{10}$$

- Boundary: Ymax

$$(A_{k,Y})^{n+1}_{I+1,B} = \frac{A^{n+1}_{k,I+1} - A^{n+1}_{k,I}}{\Delta Y}, \quad For \quad i = I+1 \Rightarrow Ymax = Y_{I+1} \tag{11}$$

Therefore, for the first derivative approximation, we have two takeaways.

- Use forward approximation at Ymin and use backward approximation at Ymax.
- For computational points, every approximation should be used depending on the rule of the upwind scheme.

c. *Central Difference*

- Boundary: Ymin

$$(A_{k,YY})^{n+1}_{1,c} = \frac{A^{n+1}_{k,2} - 2A^{n+1}_{k,1} + A^{n+1}_{k,0}}{(\Delta Y)^2}, \quad For \quad i = 1 \Rightarrow Ymin = Y_1 \quad (12)$$

A ghost node appears at $i = 0$. Because $A^{n+1}_{k,0}$ is unknown, we assume the following.

$$A^{n+1}_{k,0} = A^{n+1}_{k,1}.$$

Then,

$$(A_{k,YY})^{n+1}_{1,c} = \frac{A^{n+1}_{k,2} - A^{n+1}_{k,1}}{(\Delta Y)^2} \quad (13)$$

- Computational points

$$(A_{k,YY})^{n+1}_{i,c} = \frac{A^{n+1}_{k,i+1} - 2A^{n+1}_{k,i} + A^{n+1}_{k,i-1}}{(\Delta Y)^2}, \quad For \quad i = 2, \cdots, I \quad (14)$$

- Boundary: Ymax

$$(A_{k,YY})^{n+1}_{I+1,c} = \frac{A^{n+1}_{k,I+2} - 2A^{n+1}_{k,I+1} + A^{n+1}_{k,I}}{(\Delta Y)^2}, \quad For \quad i = I + 1 \quad (15)$$

A ghost node also appears at $I + 2$. In this case, we assume

$$A^{n+1}_{k,I+2} = A^{n+1}_{k,I+1}.$$

Therefore,

$$(A_{k,YY})^{n+1}_{I+1,c} = \frac{-A^{n+1}_{k,I+1} + A^{n+1}_{k,I}}{(\Delta Y)^2}, \quad For \quad i = I + 1 \quad (16)$$

We now implement the forward, backward, and central differences with the boundary conditions, as follows.

```
%% STEP-5.2: Finite Difference (Forward/Backward Diff Approx & central)
    % A. Forward and Backward Difference
      % forward difference (A1Y, A2Y, SY)
        dAkf  = [(Ak(2:end) - Ak(1:end-1))/deltaY 0];
        % Boundary nodes (Ymax): dAkf(I+1,:)= 0
                        % it will never be used
                        % because at Ymax we use backward
```

```
                    % ghost node: (i = I+2)

% backward difference (A1Y, A2Y, SY)
  dAkb  = [0 (Ak(2:end) - Ak(1:end-1))/deltaY];
  % Boundary nodes (Ymin): dAkb(1,:)= 0
                    % it will never be used
                    % because at Ymin we use forward
                    % ghost node: (i = 0)

% Central difference (SYY)
  ddAkYY = [(Ak(2) - Ak(1))/deltaY^2,...
          (Ak(3:end) - 2*Ak(2:end-1) + Ak(1:end-2))/deltaY^2,...
          (-Ak(end) + Ak(end-1))/deltaY^2 ];
```

5.3 *Upwind scheme.*

Next, we need a criterion to choose between the backward and forward approximation. In this step, we implement the Eq. (**??**), which is given by

$$\left(A_{k,Y}\right)_i \quad = \quad \left(A_{k,Y}\right)_{i,F} \mathbf{1}_{\tilde{a}_{ki,F}\geq 0} + \left(A_{k,Y}\right)_{i,B} \mathbf{1}_{\tilde{a}_{ki,B}<0} \tag{17}$$

The implementation of these definitions is as follows.

- **The rule.**
  - $\tilde{a}_{ki} > 0 \longrightarrow \tilde{a}_{ki} = \tilde{a}_{ki,F}$ and $\left(A_{k,Y}\right)_{i,F}$

$$\left(A_{k,Y}\right)_i^{n+1} \approx \frac{A_{k,i+1}^{n+1} - A_{k,i}^{n+1}}{\Delta Y} \equiv \left(A_{k,Y}\right)_{i,F}^{n+1} \tag{18}$$

  - $\tilde{a}_{ki} < 0 \longrightarrow \tilde{a}_{ki} = \tilde{a}_{ki,B}$ and $\left(A_{k,Y}\right)_{i,B}$

$$\left(A_{k,Y}\right)_i^{n+1} \approx \frac{A_{k,i}^{n+1} - A_{k,i-1}^{n+1}}{\Delta Y} \equiv \left(A_{k,Y}\right)_{i,B}^{n+1} \tag{19}$$

- **The implementation of the rule.** We implement the previous rule by the following two functions.

$$[K]^+ = \max\{K, 0\} \longrightarrow \left[\tilde{a}_{ki,F}\right]^+ \quad = \quad \max\{\tilde{a}_{ki,F}, 0\} \tag{20}$$

$$[K]^- = \min\{K, 0\} \longrightarrow \left[\tilde{a}_{ki,B}\right]^- \quad = \quad \min\{\tilde{a}_{ki,B}, 0\} \tag{21}$$

We then consider the Upwind scheme (rule) into the PDE of A_k.

$$\begin{aligned}\frac{1}{1-\gamma_k}\frac{A_{k,i}^{n+1} - A_{k,i}^n}{\Delta t} + \rho\frac{A_{k,i}^{n+1}}{1-\gamma_k} \quad &= \quad \frac{\gamma_k\left(A_{k,i}^n\right)^{1-1/\gamma_k}}{1-\gamma_k} + \left(\frac{\psi_i^2}{2\gamma_k} + r_i\right) A_{k,i}^{n+1} \\ &+ \quad \left[\tilde{a}_{ki,F}^n\right]^+ \left(A_{k,Y}\right)_{i,F}^{n+1} + \left[\tilde{a}_{ki,B}^n\right]^- \left(A_{k,Y}\right)_{i,B}^{n+1} \\ &+ \quad \frac{(\sigma Y_i)^2}{2(1-\gamma_k)}\left(A_{k,YY}\right)_i^{n+1}, \qquad (22) \\ & \qquad (23)\end{aligned}$$

with the definition of $\tilde{a}^n_{ki,F}$ and $\tilde{a}^n_{ki,B}$ (see Eqs. **??** and **??**), which are given by.

$$\tilde{a}^n_{ki,F} = \frac{(\sigma Y_i)^2}{2\gamma_k}\frac{(A_{k,Y})^n_{i,F}}{A^n_{k,i}} + \frac{\mu Y_i}{1-\gamma_k} + \sigma Y_i \frac{\psi_i}{\gamma_k} \tag{24}$$

$$\tilde{a}^n_{ki,B} = \frac{(\sigma Y_i)^2}{2\gamma_k}\frac{(A_{k,Y})^n_{i,B}}{A^n_{k,i}} + \frac{\mu Y_i}{1-\gamma_k} + \sigma Y_i \frac{\psi_i}{\gamma_k}. \tag{25}$$

The implementation of the Upwind scheme is done in three steps: (i) coefficient $\tilde{a}_{ki}$, (ii) indicator function, and (iii) the first derivative that includes three cases.

```
%% STEP-5.3: Upwind scheme
 % (A) aki: coefficient
    % Agent k
    akcoef_f = ( ((sigma*Y).^2)/(2*gk) ).*( dAkf./Ak ) + mu*Y/(1-gk) + sigma*Y.*psi/gk;
    akcoef_b = ( ((sigma*Y).^2)/(2*gk) ).*( dAkb./Ak ) + mu*Y/(1-gk) + sigma*Y.*psi/gk;

 % (B) Indicator Functions
    % dAkY_upwind makes a choice of forward or backward differences based on
    % based on the sign of the drift (AkY):
        % Agent k
    Ifak = akcoef_f > 0; %positive drift --> forward difference
    Ibak = akcoef_b < 0; %negative drift --> backward difference:
                  %Ifak is a logic vector: zeros and ones:
                  %1 means "true"
    I0ak = (1-Ifak-Ibak); %when a1=0

    %check: NaN or Inf at "i=1" and "i=I+1"
    check1 = [akcoef_f', akcoef_b', Ifak', Ibak', I0ak'];

 % (C) Boundaries conditions
  % To be sure that in Ymin I will use Forward
    % consistent with X1=0
     %If(1)=1; Ib(1)=0; I0(1)=0;
  % To be sure that in Y(I+1) we will use Backward
    % consistent with Z(I+1)=0
     %If(end)=0; Ib(end)=1; I0(end)=0;
  % Already taken care of automatically

 % (D) The first derivative with Upwind scheme
    AkY_Upwind = dAkf.*Ifak + dAkb.*Ibak + dAkf.*I0ak;
        %AkY_Upwind(I+1) = 0 due to dAkf(I+1)=0
```

5.4 *Discretization of PDE system.*

A. *Coefficients.*
In this step, we introduce the definition of forward, backward, and central approximation into the PDE of A_k. The goal is to obtain a PDE system formed by the PDE evaluated at every grid point. Then, the PDE system would be

$$
\begin{aligned}
\frac{1}{1-\gamma_k}\frac{A_{k,i}^{n+1}-A_{k,i}^{n}}{\Delta t}+\rho\frac{A_{k,i}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,i}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+\left(\frac{\psi_i^2}{2\gamma_k}+r_i\right)A_{k,i}^{n+1} \\
&+ \left[\tilde{a}_{ki,F}^{n}\right]^{+}\left[\frac{A_{k,i+1}^{n+1}-A_{k,i}^{n+1}}{\Delta Y}\right] \\
&+ \left[\tilde{a}_{ki,B}^{n}\right]^{-}\left[\frac{A_{k,i}^{n+1}-A_{k,i-1}^{n+1}}{\Delta Y}\right] \\
&+ \frac{(\sigma Y_i)^2}{2(1-\gamma_k)}\left[\frac{A_{k,i+1}^{n+1}-2A_{k,i}^{n+1}+A_{k,i-1}^{n+1}}{\Delta Y^2}\right]
\end{aligned} \tag{26}
$$

The "variable" in this equation is the updated value function A_k^{n+1}. The other terms are just coefficients. Furthermore, we know A_k^n in the current iteration n, since A_k^n is the initial value function in the iteration "n." After putting in order the terms of Eq. (26), we have.

$$
\begin{aligned}
\frac{1}{1-\gamma_k}\frac{A_{k,i}^{n+1}-A_{k,i}^{n}}{\Delta t}+\rho\frac{A_{k,i}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,i}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k} \\
&+ A_{k,i-1}^{n+1}\left[-\frac{\left[\tilde{a}_{ki,B}^{n}\right]^{-}}{\Delta Y}+\frac{1}{2(1-\gamma_k)}\frac{(\sigma Y_i)^2}{\Delta Y^2}\right] \\
&+ A_{k,i}^{n+1}\left[\alpha_{ki}^{n}-\frac{\left[\tilde{a}_{ki,F}^{n}\right]^{+}}{\Delta Y}+\frac{\left[\tilde{a}_{ki,B}^{n}\right]^{-}}{\Delta Y}-\frac{1}{(1-\gamma_k)}\frac{(\sigma Y_i)^2}{\Delta Y^2}\right] \\
&+ A_{k,i+1}^{n+1}\left[\frac{\left[\tilde{a}_{ki,F}^{n}\right]^{+}}{\Delta Y}+\frac{1}{2(1-\gamma_k)}\frac{(\sigma Y_i)^2}{\Delta Y^2}\right]
\end{aligned} \tag{27}
$$

where

$$
\alpha_{ki}^{n} \equiv \frac{\psi_i^2}{2\gamma_k}+r_i
$$

The expression of price of risk, ψ_t, and interest rate, r_t, come from Eq. (**??**) and (**??**), respectively. The next step is to represent Eq. (27) in a simple form as follows ($\forall i \in [1, I+1]$).

$$
\frac{1}{1-\gamma_k}\frac{A_{k,i}^{n+1}-A_{k,i}^{n}}{\Delta t}+\rho\frac{A_{k,i}^{n+1}}{1-\gamma_k} = \frac{\gamma_k\left(A_{k,i}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+A_{k,i-1}^{n+1}X_{ki}+A_{k,i}^{n+1}H_{ki}+A_{k,i+1}^{n+1}Z_{ki}, \tag{28}
$$

where the coefficients are defined as

$$X_{ki} = \left[-\frac{\left[\tilde{a}^n_{ki,B}\right]^-}{\Delta Y} + \frac{1}{2(1-\gamma_k)} \frac{(\sigma Y_i)^2}{\Delta Y^2} \right] \tag{29}$$

$$H_{ki} = \left[\alpha^n_{ki} - \frac{\left[\tilde{a}^n_{ki,F}\right]^+}{\Delta Y} + \frac{\left[\tilde{a}^n_{ki,B}\right]^-}{\Delta Y} - \frac{1}{(1-\gamma_k)} \frac{(\sigma Y_i)^2}{\Delta Y^2} \right] \tag{30}$$

$$Z_{ki} = \left[\frac{\left[\tilde{a}^n_{ki,F}\right]^+}{\Delta Y} + \frac{1}{2(1-\gamma_k)} \frac{(\sigma Y_i)^2}{\Delta Y^2} \right]. \tag{31}$$

Since $\left[\tilde{a}^n_{ki,F}\right]^+$ and $\left[\tilde{a}^n_{ki,B}\right]^-$ are known for every grid point, the coefficients X_{ki}, H_{ki}, and Z_{ki} are also known. The variables in the Eq. (28) are $A^{n+1}_{k,i-1}$, $A^{n+1}_{k,i}$, and $A^{n+1}_{k,i+1}$. We implement these coefficients as follows.

```
%% STEP-5.4: Discretization of PDE system (Ak)
       % Implicit method
       % A. Coefficients (column vectors)
        alphak = (psi.^2)/(2*gk) + r;

        % Agent k
        Xk = - min(akcoef_b',0)/deltaY + (1/(2*(1-gk)))*( (sigma.*Y').^2 )/deltaY^2;
        Hk =  alphak' - max(akcoef_f',0)/deltaY + min(akcoef_b',0)/deltaY...
               - (1/(1-gk))*((sigma.*Y').^2)/deltaY^2;
        Zk =  max(akcoef_f',0)/deltaY + (1/(2*(1-gk)))*((sigma.*Y').^2)/deltaY^2;
```

B. *Matrix of coefficients.*
We have a system of $(I+1)$ equations because the PDE of A_k is evaluated at every grid point. The next step is to express this system in matrix terms.

The system of A_k. We first evaluate the Eq. (28) for agent k, each point of the grid i, and a general time point, n.

$$\begin{aligned}
\frac{A_{k,1}^{n+1}-A_{k,1}^{n}}{(1-\gamma_k)\Delta t}+\rho\frac{A_{k,1}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,1}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+A_{k,0}^{n+1}X_{k1}+A_{k,1}^{n+1}H_{k1}+A_{k,2}^{n+1}Z_{k1}, \quad i=1\\
\frac{A_{k,2}^{n+1}-A_{k,2}^{n}}{(1-\gamma_k)\Delta t}+\rho\frac{A_{k,2}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,2}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+A_{k,1}^{n+1}X_{k2}+A_{k,2}^{n+1}H_{k2}+A_{k,3}^{n+1}Z_{k2}, \quad i=2\\
\frac{A_{k,3}^{n+1}-A_{k,3}^{n}}{(1-\gamma_k)\Delta t}+\rho\frac{A_{k,3}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,3}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+A_{k,2}^{n+1}X_{k3}+A_{k,3}^{n+1}H_{k3}+A_{k,4}^{n+1}Z_{k3}, \quad i=3\\
\vdots \quad &= \quad \vdots\\
\frac{A_{k,I+1}^{n+1}-A_{k,I+1}^{n}}{(1-\gamma_k)\Delta t}+\rho\frac{A_{k,I+1}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,I+1}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+A_{k,I}^{n+1}X_{kI+1}+A_{k,I+1}^{n+1}H_{kI+1}+A_{k,I+2}^{n+1}Z_{kI+1}
\end{aligned}$$

The last equation is evaluated in $i=I+1$–the last point of the grid.

Boundaries. To solve the PDE of A_k, we usually need boundaries (max and min level of A_k). We evaluate two cases: without boundaries and with them. In both cases, since $A_{k,0}^{n+1}$ and $A_{k,I+2}^{n+1}$ are outside of the grid, we suppose that their coefficients are zero. Therefore,

$$\begin{aligned}
X_{k1} &= 0 \qquad (32)\\
Z_{kI+1} &= 0, \qquad (33)
\end{aligned}$$

(a) Without boundaries. For agent k, the system of equations of A_k^{n+1}, after considering the expression (32) and (33), would be

$$\begin{aligned}
\frac{A_{k,1}^{n+1}-A_{k,1}^{n}}{(1-\gamma_k)\Delta t}+\rho\frac{A_{k,1}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,1}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+A_{k,1}^{n+1}H_{k1}+A_{k,2}^{n+1}Z_{k1}, \quad i=1\\
\frac{A_{k,2}^{n+1}-A_{k,2}^{n}}{(1-\gamma_k)\Delta t}+\rho\frac{A_{k,2}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,2}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+A_{k,1}^{n+1}X_{k2}+A_{k,2}^{n+1}H_{k2}+A_{k,3}^{n+1}Z_{k2}, \quad i=2\\
\frac{A_{k,3}^{n+1}-A_{k,3}^{n}}{(1-\gamma_k)\Delta t}+\rho\frac{A_{k,3}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,3}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+A_{k,2}^{n+1}X_{k3}+A_{k,3}^{n+1}H_{k3}+A_{k,4}^{n+1}Z_{k3}, \quad i=3\\
\vdots \quad &= \quad \vdots\\
\frac{A_{k,I+1}^{n+1}-A_{k,I+1}^{n}}{(1-\gamma_k)\Delta t}+\rho\frac{A_{k,I+1}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k\left(A_{k,I+1}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k}+A_{k,I}^{n+1}X_{kI+1}+A_{k,I+1}^{n+1}H_{kI+1}
\end{aligned}$$

(b) With boundaries. Using the fact that the wealth ratio of agent k, θ_k, should be between zero and one, and the expression of the consumption-wealth ratio when the economy is only populated by one agent ($k \in \{1,2\}$), we have the

following boundaries (see Lemma **??**).

$$
\begin{aligned}
N_1^{-\gamma_1} &\leq A_1 \leq N_2^{-\gamma_1} && (34)\\
N_1^{-\gamma_2} &\leq A_2 \leq N_2^{-\gamma_2} && (35)\\
& && (36)
\end{aligned}
$$

In general form,

$$
N_1^{-\gamma_k} \leq A_k \leq N_2^{-\gamma_k}
$$

For agent k, the system of equations of A_k^{n+1}, after considering the boundaries, would be

$$
\begin{aligned}
0 &= N_1^{-\gamma_k} - A_{k,1}^{n+1}, \quad i=1 \\
\frac{A_{k,2}^{n+1} - A_{k,2}^{n}}{(1-\gamma_k)\Delta t} + \rho \frac{A_{k,2}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k \left(A_{k,2}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k} + A_{k,1}^{n+1} X_{k2} + A_{k,2}^{n+1} H_{k2} + A_{k,3}^{n+1} Z_{k2}, \quad i=2 \\
\frac{A_{k,3}^{n+1} - A_{k,3}^{n}}{(1-\gamma_k)\Delta t} + \rho \frac{A_{k,3}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k \left(A_{k,3}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k} + A_{k,2}^{n+1} X_{k3} + A_{k,3}^{n+1} H_{k3} + A_{k,4}^{n+1} Z_{k3}, \quad i=3 \\
\vdots &= \vdots \\
\frac{A_{k,I}^{n+1} - A_{k,I}^{n}}{(1-\gamma_k)\Delta t} + \rho \frac{A_{k,I}^{n+1}}{1-\gamma_k} &= \frac{\gamma_k \left(A_{k,I}^{n}\right)^{1-1/\gamma_k}}{1-\gamma_k} + A_{k,I}^{n+1} X_{kI} + A_{k,I}^{n+1} H_{kI} + A_{k,I}^{n+1} Z_{kI}, \quad i=I \\
0 &= N_2^{-\gamma_k} - A_{k,I+1}^{n+1}, \quad i=I+1
\end{aligned}
$$

Matrix form (without boundaries). We then write the previous system in matrix form. For agent k, the left side of the system A_k^{n+1} is given by

$$
\text{left side} = P_k \begin{bmatrix} A_{k,1}^{n+1} \\ A_{k,2}^{n+1} \\ A_{k,3}^{n+1} \\ \vdots \\ A_{k,I+1}^{n+1} \end{bmatrix}_{I+1,1} - P_k \begin{bmatrix} A_{k,1}^{n} \\ A_{k,2}^{n} \\ A_{k,3}^{n} \\ \vdots \\ A_{k,I+1}^{n} \end{bmatrix}_{I+1,1} + Q_k \begin{bmatrix} A_{k,1}^{n+1} \\ A_{k,2}^{n+1} \\ A_{k,3}^{n+1} \\ \vdots \\ A_{k,I+1}^{n+1} \end{bmatrix}_{I+1,1} \tag{37}
$$

where,

$$
P_k = \frac{1}{(1-\gamma_k)\Delta t}\mathcal{I}_{I+1} \qquad Q_k = \frac{\rho}{1-\gamma_k}\mathcal{I}_{I+1}
$$

where $\mathcal{I}_{I+1}$ is the identity matrix of $I+1$ size.

$$
\text{left side} = P_k A_k^{n+1} - P_k A_k^{n} + Q_k A_k^{n+1} \tag{38}
$$

On the other hand, the right side would be as follows.

$$\text{right side} = \widetilde{Y}_k + \begin{bmatrix} H_{k1} & Z_{k1} & 0 & \cdots & 0 & 0 & 0 \\ X_{k2} & H_{k2} & Z_{k2} & \cdots & 0 & 0 & 0 \\ 0 & X_{k3} & H_{k3} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & X_{kI} & H_{kI} & Z_{kI} \\ 0 & 0 & 0 & \cdots & 0 & X_{kI+1} & H_{kI+1} \end{bmatrix}_{I+1,I+1} \begin{bmatrix} A_{k,1}^{n+1} \\ A_{k,2}^{n+1} \\ \vdots \\ A_{k,I}^{n+1} \\ A_{k,I+1}^{n+1} \end{bmatrix}_{I+1,1}, \tag{39}$$

where

$$\widetilde{Y}_k = \begin{bmatrix} \frac{\gamma_k \left(A_{k,1}^n\right)^{1-1/\gamma_k}}{1-\gamma_k} \\ \frac{\gamma_k \left(A_{k,2}^n\right)^{1-1/\gamma_k}}{1-\gamma_k} \\ \frac{\gamma_k \left(A_{k,3}^n\right)^{1-1/\gamma_k}}{1-\gamma_k} \\ \vdots \\ \frac{\gamma_k \left(A_{k,I+1}^n\right)^{1-1/\gamma_k}}{1-\gamma_k} \end{bmatrix}$$

The matrix of coefficients is termed M_k matrix as follows.

$$M_k = \begin{bmatrix} H_{k1} & Z_{k1} & 0 & \cdots & 0 & 0 & 0 \\ X_{k2} & H_{k2} & Z_{k2} & \cdots & 0 & 0 & 0 \\ 0 & X_{k3} & H_{k3} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & X_{kI} & H_{kI} & Z_{kI} \\ 0 & 0 & 0 & \cdots & 0 & X_{kI+1} & H_{kI+1} \end{bmatrix}_{I+1,I+1} \tag{40}$$

Then,

$$\text{right side} = \widetilde{Y}_k + M_k A_k^{n+1} \tag{41}$$

Therefore,

$$P_k A_k^{n+1} - P_k A_k^n + Q_k A_k^{n+1} = \widetilde{Y}_k + M_k A_k^{n+1} \tag{42}$$

We then implement these matrices as follows.

```
% B. Matrix of coefficients: "Mk"
 % Up Diagonal (Zk)
     updiagMk = [ 0; Zk(1:end-1)];
 % Central Diagonal (Hk)
     centdiagMk = Hk;
 % Down Diagonal (Xk)
     lowdiagMk = [ Xk(2:end); 0 ];
 % Mk
```

```
    Mk = spdiags([lowdiagMk centdiagMk updiagMk], -1:1, I+1, I+1);

% C. Vectors: Pk, Qk, Yk_tilde
 Pk = 1/((1-gk)*deltat)*eye(I+1);
 Qk = rho/(1-gk)*eye(I+1);
 Yk_tilde = (gk/(1-gk))*Ak.^(1 - 1/gk);
```

Matrix form (with boundaries). We then write the previous system in matrix form. For agent k, the left side of the system A_k^{n+1} is given by

$$\text{left side} = P_k \begin{bmatrix} A_{k,1}^{n+1} \\ A_{k,2}^{n+1} \\ A_{k,3}^{n+1} \\ \vdots \\ A_{k,I+1}^{n+1} \end{bmatrix}_{I+1,1} - P_k \begin{bmatrix} A_{k,1}^{n} \\ A_{k,2}^{n} \\ A_{k,3}^{n} \\ \vdots \\ A_{k,I+1}^{n} \end{bmatrix}_{I+1,1} + Q_k \begin{bmatrix} A_{k,1}^{n+1} \\ A_{k,2}^{n+1} \\ A_{k,3}^{n+1} \\ \vdots \\ A_{k,I+1}^{n+1} \end{bmatrix}_{I+1,1} \tag{43}$$

where,

$$P_k = \frac{1}{(1-\gamma_k)\Delta t}\widetilde{\mathcal{I}}_{I+1} \qquad Q_k = \frac{\rho}{1-\gamma_k}\widetilde{\mathcal{I}}_{I+1}$$

where $\widetilde{\mathcal{I}}_{I+1}$ is a square matrix of $I+1$ size defined as

$$\widetilde{\mathcal{I}} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}$$

$$\text{left side} = P_k A_k^{n+1} - P_k A_k^n + Q_k A_k^{n+1} \tag{44}$$

On the other hand, the right side would be as follows.

$$\text{right side} = \widetilde{Y}_k + \begin{bmatrix} -1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ X_{k2} & H_{k2} & Z_{k2} & \cdots & 0 & 0 & 0 \\ 0 & X_{k3} & H_{k3} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & X_{kI} & H_{kI} & Z_{kI} \\ 0 & 0 & 0 & \cdots & 0 & 0 & -1 \end{bmatrix}_{I+1,I+1} \begin{bmatrix} A_{k,1}^{n+1} \\ A_{k,2}^{n+1} \\ \vdots \\ A_{k,I}^{n+1} \\ A_{k,I+1}^{n+1} \end{bmatrix}_{I+1,1}, \tag{45}$$

where

$$\widetilde{Y}_k = \begin{bmatrix} N_1^{-\gamma_k} \\ \frac{\gamma_k\left(A_{k,2}^n\right)^{1-1/\gamma_k}}{1-\gamma_k} \\ \vdots \\ \frac{\gamma_k\left(A_{k,I}^n\right)^{1-1/\gamma_k}}{1-\gamma_k} \\ N_2^{-\gamma_k} \end{bmatrix}$$

The matrix of coefficients is termed M_k matrix as follows.

$$M_k = \begin{bmatrix} -1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ X_{k2} & H_{k2} & Z_{k2} & \cdots & 0 & 0 & 0 \\ 0 & X_{k3} & H_{k3} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & X_{kI} & H_{kI} & Z_{kI} \\ 0 & 0 & 0 & \cdots & 0 & 0 & -1 \end{bmatrix}_{I+1,I+1} \tag{46}$$

Then,

$$\text{right side} = \widetilde{Y}_k + M_k A_k^{n+1} \tag{47}$$

Therefore,

$$P_k A_k^{n+1} - P_k A_k^n + Q_k A_k^{n+1} = \widetilde{Y}_k + M_k A_k^{n+1} \tag{48}$$

We then implement these matrices as follows:

```
% B. Matrix of coefficients: "Mk"
 % Up Diagonal (Zk)
     updiagMk = [ 0; 0; Zk(2:end-1)];
 % Central Diagonal (Hk)
     centdiagMk = [ -1; Hk(2:end-1); -1];
 % Down Diagonal (Xk)
     lowdiagMk = [ Xk(2:end-1); 0; 0];
 % Mk
     Mk = spdiags([lowdiagMk centdiagMk updiagMk], -1:1, I+1, I+1);

% C. Vectors: Pk, Qk, Yk_tilde
 Itilde = diag([0 ones(1,I-1) 0]);
 Pk = 1/((1-gk)*deltat)*Itilde;
 Qk = rho/(1-gk)*Itilde;
 Yk_tilde = [N1^(-gk); (gk/(1-gk))*Ak(2:I)'.^(1 - 1/gk); N2^(-gk)];
```

C. *Solve the system of equations.* For both cases: without and with boundaries.

Based on the coefficient matrix M_k and the value function vector $A_k^{n+1}{}_{(\mathrm{I}+1)\times 1}$, we can solve the system

$$P_k A_k^{n+1} - P_k A_k^n + Q_k A_k^{n+1} = \widetilde{Y}_k + M_k A_k^{n+1} \tag{49}$$

Ordering the terms such as A_k^{n+1} is on the left side, we have

$$\begin{aligned} \underbrace{[P_k + Q_k - M_k]}_{=B^n} A_k^{n+1} &= \underbrace{\widetilde{Y}_k + P_k A_k^n}_{=b^n} \\ B^n A_k^{n+1} &= b^n, \end{aligned} \tag{50}$$

Since we know B^n and b^n, the variable in this system is A_k^{n+1}. This kind of system can be solved efficiently in, for instance, Matlab using "sparse matrix routines." We implement this step as follows.

```
% D. Left-hand matrix: "Bn"
        Bn = Pk + Qk - Mk;

       % E. Right-hand matrix: "bn" (column vector)
        bn = Yk_tilde + Pk*Ak';

       % F. Solve the system of equations: finding V^n+1
        Aknew = Bn\bn; % column vector
```

5.5 *Update of the price function.*
We know A_k^{n+1} from the previous step. So, we can now calculate the distance between the initial price function (for this iteration, n) A_k^n and the resulting price function, A^{n+1_k}.

$$\text{Akchange} = A_k^{n+1} - A_k^n$$

In our loop, the new value of A_k enters as an initial point in the next iteration "n+1" is A_k^{n+1}. In the current iteration, n, the initial value function ak is A_k^n, and we also have calculated "Akchange." Why do we need the vector *Akchange*? We will use it to calculate the distance between A_k^{n+1} and A_k^n, which is used to evaluate if we got the value function. We implement Akchange and the new initial guess as follows.

```
%% STEP-5.5: Update of the value function
    Akchange = Aknew - Ak';  % since we have "Ak", we calculate "Akchange"
    ak = Aknew';             % the "new initial guess" (row vector)
```

5.6 *The optimal price function.*
To compute the distance between A_k^{n+1} and A_k^n, we consider the absolute value norm of "Akchange." Importantly since the distance is positive, we first calculate the absolute value of *Akchange*, and since it is a column vector, the "max(abs(Akchange))"

will provide the maximum value of that column (in Matlab). We then compare this distance with the criterion to stop the iteration process defined in Step 5. If this distance is low than the criterion, then we find the *optimal value function* A_k–which is "*Aknew*" in the last sentence mentioned in Step 5.5.

```
   %% STEP-5.6: The optimal value function
        %We use the "Absolute-value norm"
        %We can use others: e.g., Euclidean norm
        dist(n) = max(abs(Akchange));
        if dist(n)<crit %crit=10^(-6)
            disp('Value Function Converged, Iteration = ')
            disp(n)
            break
        end

    % To know in what "iteration" we are
    disp(n)
end
```

References

Achdou, Yves, Jiequn Han, Jean-Michel Lasry, Pierre-Louis Lions and Benjamin Moll (2022), 'Income and Wealth Distribution in Macroeconomics: A Continuous-Time Approach', *Review of Economics Studies* **89**(1), 45–86.

Candler, Graham V. (2001), Finite-Difference Methods for Continuous-Time Dynamic Programming, *in* R.Marimon and A.Scott, eds, 'Computational Methods for the Study of Dynamic Economies', Oxford University Press, pp. 172–194.