Algorithm for Computing Moment-based Markov Equilibrium [*]

Xiyi Zhuang

April 21, 2025

Contents

[*] This file record my notes for the replication code for Ifrach and Weintraub (2017) 's paper, in which they proposed an algorithm written in Java code that employs the equilibrium concept of Moment-based Markov Equilibrium (MME) to solve dynamic oligopoly models.

1 Roadmap

Algorithm Moment-based Markov Equilibrium (MME) solver

1: Define parameters;
2: Construct firm's state and industry state space (moment, shock, dominant state);
3: Calculate **static competition equilibrium**:
4: **if** `model == quantity competition` **then**
5:     **do** Algorithm 1: Static quantity competition equilibrium solver; ▷ Example 6.2 in paper
6: **else if** `model == price competition` **then**
7:     **do** Algorithm 2: Static price competition equilibrium solver; ▷ Example 6.1 in paper
8: **end if**
9: **return** Equilibrium profit vector: `profitFringe`, `profitDominant`;
10: Initialize value matrix: $W(x, \hat{s}) =$ `profitFringe`$/(1\text{-}\beta)$, `profitDominant`$/(1\text{-}\beta)$;
11: Calculate **dynamic equilibrium** using Real-Time Dynamic Programming (RTDP):
12: **while** weighted Euclidean norm $(||\mu - \mu'||_h, ||\lambda - \lambda'||_h) <$ `strategyTOL` **do**
13:         **do** Algorithm 3: Real-Time Dynamic Programming;
14:         **do** Algorithm 12: Calculating the weighted Euclidean norm;
15: **end while**

Notes:

1. **Line 8**: `profitFringe` will be a $\mathcal{S} \times \mathbb{N}^{|\mathcal{X}_{\mho}|}$ matrix and `profitDominant` will be a $\mathcal{S} \times \mathbb{N}^{|\mathcal{X}|}$ matrix, where $\mathcal{S} = \mathcal{S}_\theta \times \mathcal{S}_d \times \mathcal{Z}$, where $|\mathcal{X}_f|$ and $|\mathcal{X}_d|$ are number of states (both include on inactive state) for fringe firms and dominant firms respectively.

2. **Line 10**: This initialization is used only once at the very beginning of the whole algorithm (before the RTDP iteration starts). Function W is the expected continuation value function.

2 Table

2.1 State Parameters

Code	Paper	Definition
`xBar = 6`	/	number of grid points for firm's states (5 physical states + one inactive state)[1]
`nDominant = 3`	$\bar{D}$	maximum number of dominant firms active in the industry
`xDom = 4`	$\bar{x}$	the auxiliary state for fringe firms to consider tier transitions
`barXd = 3`	$\mathbb{N}^{\|\mathcal{X}_d\|}$	number of grid points in dominant states
`barXf = 5`	$\mathbb{N}^{\|\mathcal{X}_f\|}$	number of grid points in fringe states
`legend`	$\|\mathcal{X}\|$	$\{\text{inactive state}, x_1, x_2, x_3, x_4, x_5\}$ [2]
`legendDominant`	$\|\mathcal{X}_d\|$	$\{\text{inactive state}, x_4, x_5\}$
`legendFringe`	$\|\mathcal{X}_f\|$	$\{\text{inactive state}, x_1, x_2, x_3, x_4\}$
`fringeToDominant=1`	/	when $f \to d$, new dominant firm's index in the $\|\mathcal{X}_d\|$ vector becomes 1 (x_4)
`dominantToFringe=3`	/	when $d \to f$, new fringe firm's index in the $\|\mathcal{X}_f\|$ vector becomes 3 (x_3)
`M = 5`	N	maximum of total number of firms that the industry can accommodate

2.2 Moment Discretization Parameters

Code	Paper	Definition
`thMin = 0`	/	starting interval (set to smallest fringe state x smallest number of fringe firms)
`thMax = 0`	/	$\bar{\theta}$, gets modified as required through out the algorithm
`thGrid = 0`	/	step size in grid, for small number of firms make it equal to increment in firm state
`thMinGlobal = 0`	/	grids should remain within [`thMinGlobal, thMaxGlobal`]
`thMaxGlobal = 50`	/	grids should remain within [`thMinGlobal, thMaxGlobal`]

2.3 Profit Function Parameters

2.3.1 Quantity Competition

Code	Paper	Definition
`a = 2`	$\bar{q}_s$	ay+b is a map from state y to capacity for fringe states
`b = 0`	$\bar{q}_i$	/
`c = 2`	$\bar{q}_s$	cy+d is a map from state y to capacity for dominant states
`d = 0`	$\bar{q}_i$	/
`e = 60`	e	demand parameters : $Q = m(e - f \cdot p)$
`f = 10`	f	demand parameters : $Q = m(e - f \cdot p)$

[1] Note that Java index starts from zero.

[2] Example 6.1 in the paper : " each firm's state variable x_{it} is a natural number that represents the quality of its product. "; Example 6.2 in the paper : " We assume that the capacity grows linearly in state, that is $q(x) = \bar{q}_i + \bar{q}_s x$, for some constant $\bar{q}_i, \bar{q}_s > 0$ ".

Code	Paper	Definition
`ms = 1`	m	market size m in demand function : $Q = m(e - f \cdot p)$

2.3.2 Price Competition

Code	Paper	Definition
`rho = .3`	ρ	depreciation of goodwill (Appendix G in paper)
`x0 = 1`	/	seed of goodwill (see paper for def. of parameters)
`m = 30.0`	m	market size (in millions), m in equation (7) in paper
`gamma1Fringe = 1.00`	α_1	α_1 in the utility function in example 6.1
`gamma2Fringe = 0.5`	α_2	α_2 in the utility function in example 6.1
`Y`	Y	Y, income in the utility function in example 6.1

3 Investment Parameters

Code	Paper	Definition
`dFringe = 0.5`	d_f	constant investment cost per unit of INV
`dDominant = 0.5`	d_d	constant investment cost per unit of INV

3.1 DP Parameters

Code	Paper	Definition
`K = 1000000`	K	Number of simulated periods in one RTDP iteration
`VITOL = 0.001`	/	tolerance of value iteration
`NewtonsyTOL = 0.05`	/	tolerance in Newton's method
`strategyTOL = 0.05`	/	tolerance of convergence

4 Algorithm

Algorithm 1 Static quantity competition equilibrium solver

1: **for** moment-based industry state $\hat{s}_t = (\theta_t, d_t, z_t) \in \hat{\mathcal{S}}$ **do**
2: ▷ $\theta_t = \sum_{x \in \mathcal{X}_f} \bar{q}(x)$, total installed capacity among fringe firms
3: Solve the optimal production level $q^*(x, s_t)$ for each firm
4: For **fringe** firms:
5: $q^* = \bar{q}$, produce at full capacity
6: For **dominant** firms:
7: Residual demand faced by dominant firms: $R = e \cdot m - \theta$
8: Compute initial cournot equilibrium production for each dominant firm: $q_d = \frac{R}{1+\bar{D}}$
9: count:=0; Set $q_d = q$ for $d \in D$;
10: **while** $\{\ \exists\ x_d \in \mathcal{X}_d,\ \text{s.t. } q_d > \bar{q}(x)\ \}$ **do**
11: **for** $d \in \bar{D}$ **do**
12: **if** $q_d > \bar{q}_d(x)$ **then** ▷ check if the calculated production exceeds capacity
13: Set $q_d = \bar{q}_d(x)$; count:=count+1;
14: Subtract this firm's capacity from the residual demand: $R' = R - \bar{q}(x)$; $R = R'$;
15: **end if**
16: Recalculate the production for remaining dominant firms: $q'_d = \frac{R'}{1+\bar{D}-\text{count}}$; $q_d = q'_d$;
17: **end for**
18: **end while**
19: **return** `productionDoiminantVector[i]` ▷ Equilibrium production profile for dominant firms
20: Equilibrium industry price: $P^* = e/f - (\sum q^*)/f \cdot m$ (same across all firms)
21: Firm's profits: $\pi_d(x) = \pi_f(x) = q^*(x) \cdot P^*$ (marginal costs = 0)
22: Calculate equilibrium consumer/producer surplus, market share vector
23: Consumer surplus $= \frac{(Q^*)^2}{2f \cdot e}$
24: Producer surplus $= P^* Q^*$ ▷ zero margianl cost
25: **end for**

Notes:

1. **Line 2**: In the capacity competition model, firm's state variable determines its production capacity $\bar{q}(x)$ and it grows linearly in state $\bar{q}(x) = \bar{q}_i + \bar{q}_s x$. The moment is the total installed capacity for all fringe firms.
2. **Line 9**: First, set the production level for all dominant firms to be the cournot competition outcome of $\bar{D}$ identical firms, ignoring firms' heterogeneous capacity for now.
3. **Line 19**: The solution of the equilibrium production profile do not need any calculations of FOC of firm's profit maximization problem. In sum, the equilibrium q^* profile for dominant firms is solved by iteratively comparing firm's capacity and the average residual demand faced by dominant firms.
4. **Line 23**: The consumer surplus is the triangle area under the demand curve.

Algorithm 2 Static price competition equilibrium solver

1: **for** moment-based industry state $\hat{s}_t = (\tilde{\theta}_t, d_t, z_t) \in \hat{\mathcal{S}}$ **do**

2: $\triangleright$ $\tilde{\theta}_t = \sum_{y \in \mathcal{X}_f} y^{\alpha_1} f_t(y)$, total attraction by fringe firms

3: Solve the optimal price level $p^*(x, s_t)$ for each firm

4: For **fringe** firms: [Deduction 4.2.1]

5: $p_f^* = (Y + c\alpha_2)/(1 + \alpha_2)$;

6: `momentAttraction = moment*Math.pow((Y-priceFringe), gamma2Fringe)` $= \tilde{\theta}_t \cdot (Y - p_f^*)^{\alpha_2}$

7: For **dominant** firms:

8: **while** $\text{sqrt}\{\sum_{x \in \mathcal{X}_d} (\frac{\partial\ \pi(x,p)}{\partial\ p})^2\}/\text{sqrt}\ (\bar{D}) <$ `tolerance` **do** $\triangleright$ Euclidean distance

9: Derive the expression of $\frac{\partial\ \pi(x,p)}{\partial\ (Y-p)}$ (FOC); [Deduction 4.2.2]

10: Calculate the hessian matrix;

11: Use Newton's method to update p^*; [Deduction 4.2.3]

12: **for** $i \in D$ **do**

13: `update` $=$ `update` $+ \frac{\partial}{\partial p} \cdot \sum_{j \in D} H^{-1}(i, j)$;

14: $p' = p -$ `update`;

15: **if** $p < 0.01$ **then**

16: Set $p = 0.01$ $\triangleright$ Restrict the lower bound of price

17: **end if**

18: **end for**

19: **end while**

20: **return** the equilibrium price vector $\mathbf{p} = \{p_f^*, p_d^* = [p_1^*, ..., p_D^*]\}$

21: Calculate equilibrium consumer/producer surplus, market share vector

22: Firm's profit: $\pi^* = m(p^* - c) \cdot z_t \cdot$ (Attraction/Total Attraction) $- F$

23: Consumer surplus $= m \cdot \left(1 + \text{momentAttraction} + \sum_{j \in D_t} x^{\alpha_1} z^{\alpha_2}\right)$

24: Producer surplus $= \pi^* - F$ $\triangleright$ $F =$ fixed production costs

25: **end for**

Notes:

1. **Line 2**: In the replication code, the authors do not explicitly model the function form of y and $f_t(y)$, instead, the $\tilde{\theta}$ was directly given some discrete, scalar values to represents the summation and combination of y and $f_t(y)$.

2. **Line 8**: Line 8 in this algorithm use average squared distance to measure how close is the FOC to zero. If the Euclidean distance is not significantly different from zero (i.e., smaller than the tolerance level `tolerance`), than the price level at this iteration is set to be the equilibrium price level.

3. **Line 9**: Since the variable p appears multiple times in the profit function in the form of $(Y - p)$, in the code, p^* is solved by taking the derivative of $Y - p$ as a whole and updating it accordingly. The equilibrium price level is then solved by $p^* = Y - z^*$.

Algorithm 3 Real-Time Dynamic Programming

1: Initiate industry state (f_0, d_0) and the perceived industry state $\hat{\mathcal{S}} = \{\theta(f_0), d_0\}$;
2: `histogram` $= [\ 0, 0, M - D, 0, 0\ ]$ ▷ Records number of fringe firms at each state, $\{x, s\}$
3: Initialize value matrix: $W(x, \hat{s})$ = `profitFringe`/(1-β), `profitDominant`/(1-β);
4: k:=1; ▷ the index for iteration rounds
5: **while** iteration $< K$ **do** ▷ $K = 10^6$, number of simulations
6: Givem the industry state $s = \hat{\mathcal{S}}$ ▷ this is an kick-off point to start RTDP
7: **if** $k > 100$ **then**
8: $K = \frac{K}{2} \cdot (2 + \frac{k}{100} - 1)$ ▷ for higher iteration, reduce simulation times
9: **end if**
10: Draw a random shock used for this time of simulation;
11: Calculate the weights for each state:
12: **if** $\theta(f)$ is off grid **then**
13: Floor off-grid points to the nearest lower grid: $\theta \to \lfloor \theta \rfloor$;
14: Weights $= \frac{\theta - \lfloor \theta \rfloor}{\lfloor \theta \rfloor}$ ▷ how close the off-grid point is to its nearest lower grid point $\lfloor \theta \rfloor$
15: New on-grid industry state: $\lfloor s \rfloor = \{\lfloor \theta \rfloor, d, z\}$
16: **else** Weights $= 1$ ▷ Points on grid do not need to be weighted
17: **end if**
18: Compute the Bellman equation and strategies for fringe firms:
19: **do** Algorithm 4: Bellman equation solver for fringe firms
20: **return** $V_f(s, x)$, $EV_f(s, x)$; ▷ Value function, continuation value function
21: **return** ι^*, $p^{exit}(s, x)$; ▷ Investment, exit decisions
22: **do** Algorithm 6: Simulate fringe firms state transitions;
23: **return** `histogramNewPreEntry`, `countNumberExitingFringe`
24: **return** `histogramNewPreTierTrans`, `countNumberEnteringFringe`
25: Compute the Bellman equation and strategies for dominant firms:
26: **do** Algorithm 5: Bellman equation solver for dominant firms
27: **return** $V_d(s, x)$, ι^*;
28: **do** Algorithm 7: Dominant firms' strategies and tier transitions
29: **return** `histogramNew`, `dominantNew` (See Figure 5 for description)
30: Draw a new random aggregated shock z'
31: Update the new industry state: $s = \hat{\mathcal{S}}' = \{\theta(f'), d', z'\}$; ▷ evolve to a new industry state
32: Run Line 8-13 and convert the new industry state to be on-grid;
33: **return** $\lfloor s' \rfloor$, Weights$'$;
34: **end while**
35: Update perceived transition kernel: **do** Algorithm 8: Update transition kernel
36: Update the W function: **do** Algorithm 9: Update continuation value function
37: k:=k+1;
38: **if** $k > 2$ **then**
39: **do** Algorithm 9: Value function iteration
40: **end if**

Notes:

1. **Line 1**: Note that the random aggregated shock is not initialized at the very beginning like what we do for f and d, the aggregated shock is randomly draw every time when a new round of simulation k starts (see Line 11).

2. **Line 2**: f_0 is initialized to be `histogram` $= [\ 0, 0, M - D, 0, 0\ ]$, where M is the maximum number of total firms that the industry can accommodate, D is the number of dominant firms (constant, always). All fringe firms are initialized at the medium state, and the moment θ_f is then calculated under the specified static competition model.

3. **Line 3**: See equation (21) in paper for the definition of $W(x, \hat{s})$. Note for the difference between function W and function V (mentioned later), function W is the expected continuation value which is a summation over V function averaged by transition probability.

4. **Line 12-19**: Note that in the RTDP algorithm, the weights used to compute strategies and value functions are not transition probabilities but weights to measure how far an off-grid point is to its cloest on-grid point.

5. **Line 9**: Note that only the computed fringe moment, $\theta(f)$, can take some off-grid values since the grid points for moments can not incorporate all of the possible values that the detailed fringe state might lead to (i.e. different f might lead to same θ, and since we have large number of fringe firms, it's computationally infeasible to exactly map any changes in f to θ.) For d and z, we have already taken all of the possible dominant firms distributions into state space, so they will not take off-grid values.

6. **Line 19**: Only investment strategies are needed to be computed for dominant firms and investments decisions are solved in the computation of Bellman equation.

7. **Line 20**: `histogramNew` $= f'$, `dominantNew` $= d'$.

Algorithm 4 Bellman equation solver for fringe firms

1: Given the current on-grid industry state $\lfloor s \rfloor$, the value matrix W_f and weights:

2: **for** $x \in \mathcal{X}_f$ **do** ▷ x is the index corresponding to a state in the fringe state vector

3: $\quad W_f^{\text{mid}} = W_f(\lfloor s \rfloor, x) \times \text{Weight};$ ▷ Stay

4: $\quad$ **if** $x > 1$ **then**

5: $\quad\quad W_f^{\text{low}} = W_f(\lfloor s \rfloor, x-1) \times \text{Weight};$ ▷ move down

6: $\quad$ **else if** $x < |\mathcal{X}_f| - 1$ **then**

7: $\quad\quad W_f^{\text{high}} = W_f(\lfloor s \rfloor, x+1) \times \text{Weight};$ ▷ move up

8: $\quad$ **end if**

9: $\quad$ Solve for firm's **investment decisions** $\iota^* = \iota^*(\lfloor s \rfloor, x)$; [Deduction 4.3]

10: $\quad$ Calculate state transition probabilities under optimal investment ι^*:

11: $\quad p^{\text{high}} = \frac{\delta a \iota^*}{1+a\iota^*}$; $p^{\text{mid}} = \frac{1-\delta'+(1-\delta)a\iota^*}{1+a\iota^*}$; $p^{\text{low}} = \frac{\delta'}{1+a\iota^*}$;

12: $\quad$ **return** $EV(\lfloor s \rfloor, x) = -d\iota^* + \beta \cdot (p^{\text{low}} W^{\text{low}} + p^{\text{mid}} W^{\text{mid}} + p^{\text{high}} W^{\text{high}})$ ▷ `continuationValue`

13: $\quad$ **return** $p^{\text{exit}}(\lfloor s \rfloor, x) = \exp\big(-\lambda_1 EV(\lfloor s \rfloor, x)\big)$ ▷ `probExit`

14: $\quad$ **return** $E(\phi \mid \phi > EV) = EV(\lfloor s \rfloor, x) + \frac{1}{\lambda_1}$ ▷ `condSellOff`

15: $\quad$ **return** $V(\lfloor s \rfloor, x) = \pi_f(\lfloor s \rfloor, x) + (1-p^{\text{exit}})EV + p^{\text{exit}} \cdot E(\phi \mid \phi > EV)$ ▷ `V`

16: **end for**

Notes:

1. **Line 2**: This is an ergodic process over the fringe states [inactive, x_1, x_2, x_3, x_4], see Figure 1 for description.
2. **Line 3**: Make sure that state degrading won't directly lead to the inactive state.
3. **Line 4**: Make sure that state upgrading won't directly lead to the auxiliary state (to become dominant).
4. **Line 8**: For fringe firms with state x_1 (index 1), and firms with state x_4 (the auxiliary transition state), Line 17 should be modified to be:

$$EV(\lfloor s \rfloor, \underline{x} = x_1) = d\iota^* + \beta \cdot (p^{\text{low}} W^{\text{mid}} + p^{\text{mid}} W^{\text{mid}} + p^{\text{high}} W^{\text{high}})$$

$$EV(\lfloor s \rfloor, \bar{x} = x_4) = d\iota^* + \beta \cdot (p^{\text{low}} W^{\text{mid}} + p^{\text{low}} W^{\text{mid}} + p^{\text{high}} W^{\text{mid}})$$

   Since the transition under investment strategies should directly lead firms with lowest fringe state to further move down to the inactive state (state transition should not directly lead to exit). It also cannot directly further move fringe firms with highest state to posses a higher state that only dominant firms are allowed to have.

5. **Line 12**: Since the marginal return on investment is decreasing, A represents the return from investing one unit. If the return from investing one unit is negative when costs are not considered, then there will definitely be no investment.
6. **Line 12**: This is the continuation value. Note that the investment costs $-d\iota^*$ is part of the **continuation value** in Line 12. That is, although is does not affect the calculation of V in Line 15, cannot express $V(\lfloor s \rfloor, x) = \pi - d\iota^* + (1-p)EV + pEV(|\cdot), EV = \beta(\sum pW)$ since firms will only decide to invest if they decide not to exit, meaning that the investment cost must be subtracted when calculating continuation value.
7. **Line 13**: This is the probability of exit, $p^{\text{exit}} = F(\phi > EV) = \exp(\lambda_1 \cdot EV)$, where λ_1 is the exponential distribution parameter for sell-off value ϕ.
8. **Line 14**: This is the conditional sell-off value. It's conditional because the sell-off value is privately observed by firms before exit decisions are made.

Algorithm 5 Bellman equation solver for dominant firms

```
1: Given the current on-grid industry state ⌊s⌋, the value matrix W_d and weights:
2: for x ∈ 𝒳_d do                ▷ x is the index corresponding to a state in the dominant state vector
3:     if x > 1 then
4:         W_d^low = W_d(⌊s⌋, x_0) × Weight;                                   ▷ inactive
5:     else if x < |𝒳_d| − 1 then
6:         W_d^high = W_d(⌊s⌋, x_6) × Weight;                                  ▷ state x_6
7:     else
8:         W_d^mid = W_d(⌊s⌋, x_5) × Weight;                                   ▷ state x_5
9:     end if
10:    Solve optimal investment decisions and store it into vector strategyDominant;
11:    return ι*(⌊s⌋, x)
12:    return V(⌊s⌋, x) = π_f(⌊s⌋, x) − dι* + β · (p^low W^low + p^mid W^mid + p^high W^high)
```

Notes:

1. **Line 1**: Remember that dominant firms would not directly exit the market before downgrading into a fringe firms. That is, only investment strategies are needed to extract from dominant firms. Dominant firms will first have to downgrade into a fringe firm and then exit the market as a fringe firm if it wants. In this sense, the "inactive state" for dominant firms can be more sensibly understood as a "loss" state, which corresponds to the -1 value assigned to the W matrix in Figure 4.

2. **Line 3-9**: Since there's only 2 physical states and one inactive state for dominant firms (see Figure 1 for description), it's more convenient to just simply specify three W correspond to inactive state, x_4 and x_5, respectively.

3. **Line 10**: This is the same as Line 9-11 in Algorithm 4.

4. **Line 12**: This is the continuation value and the Note for Line 8 in Algorithm 4 apply the same to here too.

Algorithm 6 Simulate fringe firms state transitions

```
Set maximalFractionInactiveFringe        ▷ This is the upper bound for fraction of inactive firms
histogramNewPreEntry = histogram;
for $x \in \mathcal{X}_f$ do        ▷ This is an iteration over all states
    if histogramNewPreEntry[x] $> 0$ then        ▷ Jump to the next state if no firms at state $x$
        $i = 1$;
        while $i <$ histogramNewPreEntry[x] do        ▷ This is an iteration over all firms in state $x$
            Draw an random number RAND within $[0, 1)$;        ▷ Monte-Carlo Simulation
            Solve incumbent fringe firm's exit decisions:
            if RAND $< p^{\text{exit}}(\lfloor s \rfloor, x)$ then
                histogramNewPreEntry[0]++;        ▷ exit, [1,0,M-D,0,0]
            end if
            Solve incumbent fringe firm's transitions under $\iota^*$:
            if RAND $< p^{\text{low}}(\lfloor s \rfloor, x)$ then
                histogramNewPreEntry[x-1]++        ▷ move down, [0,1,M-D,0,0]
            else if RAND $< p^{\text{low}}(\lfloor s \rfloor, x) + p^{\text{mid}}(\lfloor s \rfloor, x)$ then
                histogramNewPreEntry[x]++        ▷ stay, [0,0,M-D+1,0,0]
            else if RAND $< p^{\text{high}}(\lfloor s \rfloor, x)$ then
                histogramNewPreEntry[x+1]++        ▷ move up, [0,0,M-D,1,0]
            end if
            i++
        end while
    end if
end for
return histogramNewPreEntry = histogramNewPreEntry - [1,1,1,1,1]
Number of fringe firms that will exit next period: histogramNewPreEntry[0] - historgram[0]
histogramNewPreTierTrans = histogramNewPreEntry
Solve for potential entrants' entry decisions; Compute $V^e = W_f(\lfloor s \rfloor, x^e) \times$ Weight
while a = histogram[0] $> 0$ do        ▷ iteration over all inactive fringe firms (potential entrants)
    Draw a entry cost from the specified entry cost distribution: $\lambda = -\frac{\ln(\text{RAND})}{\lambda_1}$;
    if $V^e - \lambda > 0$ then
        Draw an random number RAND within $[0, 1)$;
        if RAND $< p^{\text{low}}(\lfloor s \rfloor, x^e)$ then
            histogramNewPreTierTrans[x-1]++        ▷ down,+[0,1,0,0,0]
        else if RAND $< p^{\text{low}}(\lfloor s \rfloor, x^e) + p^{\text{mid}}(\lfloor s \rfloor, x^e)$ then
            histogramNewPreTierTrans[x]++        ▷ stay, +[0,1,0,0,0]
        else if RAND $< p^{\text{high}}(\lfloor s \rfloor, x^e)$ then
            histogramNewPreTierTrans[x+1]++        ▷ up, +[0,0,1,0,0]
        end if
        histogramNewPreEntry[0] = histogramNewPreEntry[0] - 1
    end if
    a = a - 1
end while
return histogramNewPreTierTrans
# potential entrants that will enter next period: histogramNewPreEntry[0] - histogramNewPreTierTrans[0] =0
```

Notes:

1. **Line 1**: Use the initialized distributional histogram `histogram = [0,0,M-D,0,0]` as an example to illustrate the idea of this algorithm, where each element in `histogram` represents number of fringe firms on this state. The joint outcome of incumbent firms' exit and investment decisions will determine next period's fringe firms' profile (`histogramNew`). Line 4-11 is the outcome driven by incumbents' exit decisions, Line 12-19 is the outcome driven by incumbents' state transitions which are further driven by their investment decisions.

2. **Line 4**: This is an erogdic process across all physical fringe states (excluding inactive state) and all firms in each state (Line 7).

3. **Line 7**: Note that this random number will be drawn each time for each firm at each state (different firms will have different `RAND` to be compared to).

4. **Line 12**: Note that this is the outcome cause by firm's optimal investment choices which are already solved in the bellman function.

5. **Line 13**: Think more about the order of the `If-ElseIf-Else` loop here.

6. **Line 14**: If $x - 1 = 0$ then set $x - 1 = 0$ (transition should not directly leads to exit).

7. **Line 24**: This is the new distribution for fringe firms at next period after the exiting firm exited, incumbent fringe firms invested and transited. Remember the timing assumptions specify that the exit and transition will be realized at the beginning of next period.

8. **Line 32-38**: Footnote 4 in paper: "It is straight forward to generalize the model by assuming that *entrants can also invest to improve their initial state.*"

9. **Line 39**: This is the distribution histogram after taking account of state transitions and entrants entry decision but do not take into account of tier transitions.

10. **Line 43**: This is the distribution histogram for fringe firms at the next period after the exiting firm exited, incumbent fringe firms invested and transited, and entrants entered and settled at a new state after initial investment. The only left transitions to be considered is the tier transitions that are simulated in Algorithm 7.

$$\text{Incumbents} \xrightarrow{\text{exit ?}} \begin{cases} \text{YES}, \quad \ell^* = 0 & \Rightarrow \text{potential entrants} + 1 \\ \text{NO}, \quad \ell^* > 0 & \Rightarrow \text{state transitions} \end{cases}$$

$$\text{Time } t \quad \longrightarrow \quad \text{Time } t+1$$

Algorithm 7 Dominant firms' strategies and transitions

1: Give dominant firms' vector `dominant` and $\iota^*(\lfloor s \rfloor, x)$;
2: $p^{\text{high}} = \frac{\delta a \iota^*}{1+a\iota^*}$; $p^{\text{mid}} = \frac{1-\delta' + (1-\delta)a\iota^*}{1+a\iota^*}$; $p^{\text{low}} = \frac{\delta'}{1+a\iota^*}$;
3: **for** $d \in$ `dominant` **do** ▷ `dominant[d]`$= x_d$
4:     Draw an random number `RAND` within $[0, 1)$; ▷ Monte-Carlo Simulation
5:     **if** `dominant[d]` > 0 **then**
6:         **if** `RAND` $< p^{\text{low}}(\lfloor s \rfloor, x_d)$ **then**
7:             `dominantNewPreTierTrans[x-1] = max(0, dominant[d] - 1)` ▷ down
8:         **else if** `RAND` $< p^{\text{low}}(\lfloor s \rfloor, x_d) + p^{\text{mid}}(\lfloor s \rfloor, x_d)$ **then**
9:             `dominantNewPreTierTrans[x] = dominant[d]` ▷ stay
10:         **else if** `RAND` $< p^{\text{high}}(\lfloor s \rfloor, x_d)$ **then**
11:             `dominantNewPreTierTrans[x+1] = min(barXd-1, dominant[d] + 1)` ▷ up
12:         **end if**
13:     **end if**
14: **end for**
15: **return** `dominantNewPreTierTrans`
16: Now simulate firms' tier transitions:
17: `histogramNew = histogramNewPreTierTrans`; `dominantNew = dominantNewPreTierTrans`;
18: `aux` $= \bar{D} - 1$;
19: **while** `histogramNewPreTierTrans[nGridFringe-1]` > 0 **do**
20:     **if** `aux` > 0 **then** ▷ conditions for tier transitions are satisfied
21:         `dominantNew = [1,1,2]` ▷ $x_{f \to d} = x_4$
22:         `histogramNew = [0,1,4,0,0]` ▷ see Figure 5
23:         `aux = aux - 1`;
24:     **else** ▷ if cannot accommodate new dominant firms
25:         `histogramNew = [0,1,4,1,0]`
26:     **end if**
27: **end while**
28: **return** `histogramNew` (f); `dominantNew` (d);
29: Construct the new industry state;

Notes:

1. **Line 1**: `dominant` is a detailed vector that store each dominate firm's state, use `dominant = [0,1,2]`, `histogramNewPreTierTrans = [0,1,4,0,1]` as an example to illustrate this algorithm.
2. **Line 15**: This is the dominant firm's detailed state vector after transitions are realized under the transition probabilities driven by investment choices, but before tier transitions are considered.
3. **Line 17-18**: `histogramNewPreTierTrans[nGridFringe-1]` > 0 and `aux` is the condition that there exists some positive number of fringe firms located at the auxiliary state (x_4, the highest state for fringe firms) so that tier transition ($f \to d$) is possible, and a non-negative `aux` make sure that there's some room to accomodate new dominant firms (after new dominant firms enter, number of active dominant firms won't go beyond $\bar{D}$)
4. **Line 25**: If the industry cannot accommodate more active dominant firms (number of active dominant firms already reach the upper bar $\bar{D}$), then fringe firms at the auxuliary transition state will move down at the next period.

Algorithm 8 Update transition kernel

1: Calculate how many fringe firms at auxiliary transition state successfully become dominant;

2: $p(f \to d) =$ `numberEnter / histogramNewPreTierTrans[nGridFringe-1]`

3: `count[industryCode-1]++;`

4: `countMomentState[stateSpace.momentsEncode(moments)]++;`

Notes:

1. **Line 1**: This is then used as the probability for fringe firms at x_4 to become dominant in Algorithm 9.
2. **Line 2**: `histogramNewPreTierTrans[nGridFringe-1]` is the pool that records the number of fringe firms at the auxiliary transition state (the highest state for fringe firms), only firms at this state level can become dominant. So that the difficulty for fringe firms to become dominant is determined by the room to accommodate more dominant firms ($\bar{D} -$ `activeDominant`) and how many peer fringe firms are at the auxiliary transition state.
3. **Line 3-4**: counts visits to industry states and moment states. These are then used as weights to update the W function.

Algorithm 9 Update continuation value function

1: Give the industry state $s = \hat{\mathcal{S}} = \{\theta(f), d, z\}$ and the weights;

2: For **fringe firms**:

3: **for** $x \in \{x_1, x_2, x_3\}$ **do**

4:     **do** Algorithm 4: Bellman equation solver for fringe firms;

5:     **return** $V_f(x, s') = \pi(x, s') + (1 - p^{\text{exit}})W(x, s) + p^{\text{exit}} EV(\phi \mid \phi > V)$;

6:     Update the value for W function at the new industry point:

7:     $W_f^{\text{new}}(x, s') = \alpha(k) \cdot V_f(x, s') + \big(1 - \alpha(k)\big) \cdot W_f(x, s)$; $\alpha(k) = \frac{1}{k+5+\texttt{count[industryCode]}}$;

8: **end for**

9: **for** $x \in \{x_4\}$ **do** ▷ for the auxiliary transition state

10:     **if** `numInactiveDom = 0` **then**

11:         $W_f^{\text{new}}(x_4, s) = W_f(x_3, s)$

12:     **else** ▷ can accommodate new dominant firms

13:         **if** `numEnter > 0` **then**

14:             $p^{f \to d} = \frac{1}{\texttt{count[histogram[nGrid-1]]}}$

15:             $V(x_4, s') = p^{f \to d} \cdot V_d(x_4, s') + (1 - p^{f \to d}) \cdot V_f(x_3, s')$;

16:         **end if**

17:         **if** `numberEnter == 0` **then** ▷ no one enters

18:             $V =$

19:         **end if**

20:     **end if**

21:     $W_f^{\text{new}}(x, s') = \alpha(k) \cdot V(x, s') + \big(1 - \alpha(k)\big) \cdot W_f(x, s)$; $\alpha(k) = \frac{1}{10k+5+10 \cdot \texttt{count[histogram[nGrid-1]]}}$;

22: **end for**

23: For **dominant firms**:

24: **for** $d \in$ `dominant` **do**

25:     **if** $d > 0$ **then** ▷ active dominant firms

26:         $W_d^{\text{new}}(x, s') = \alpha(k) \cdot V_d(x, s') + \big(1 - \alpha(k)\big) \cdot W_d(x, s)$; $\alpha(k) = \frac{1}{k+5+\texttt{count[industryCode]}}$;

27:     **end if**

28:     **if** d **then**

29:

Notes:

1. **Line 1**: This is the original industry state feed into the algorithm, not the new industry state s'. Note that the weights used in this algorithm is the number that measure the distance between on-grid points and off-grid points. This Algorithm 9 is the stochastic algorithm proposed by Pakes and McGuire (2001).
2. **Line 3**: This is the iteration for states in `legendFringe` except for the inactive state and the auxiliary transition state.
3. **Line 5**: Note that this is the V for the new industry state generated by the Monte-Carlo simulation result.
4. **Line 6**: Note that only the point at s' is updated, points not visited in k iteration round will not be updated.
5. **Line 7**: This is same as the equation (8) in Pakes and McGuire (2001):

$$w^{j+1}(\nu; i, s^j) - w^j(\nu; i, s^j) = \alpha(j, s^j) \Big\{ V[i + \nu - \zeta^{j+1}, \hat{s}_i^{j+1} + e(i + \nu - \zeta^{j+1}) : w^j] \Big\} - w^j(\nu; i, s^j)$$

where $\alpha(j, s^j)$ is the inverse of the number of past iterations that s' has been visited. α is introduced because the algorithm only generate one random draw using the Monte-Carlo simulation, therefore only incremental difference is updated.

6. **Line 10**: `numInactiveDom` is the number of zeros in `Dominant` (not in `DominantNew`), if there's no zero in `Dominant` at the k iteration then tier transition won't happen in the next $k+1$ iteration.

Algorithm 10 Value function iteration

1: `VIiteration = 0;` ▷ count for number of value iteration rounds
2: **while** `dif` $>$ `VI-TOL-PRECISION` || `!VIDominant` || `!VIFringe` **do**
3: `Viiteration ++;`
4: `VIFringe = true; VIdominant = true;` ▷ boolean values that determine whether to update
5: **if** `difFringe > difDominant & difFringe > tol` **then**
6: `VIFringe = true; VIdominant = false;`
7: **end if**
8: **if** `difDominant > difFringe & difDominant > tol` **then**
9: `VIFringe = false; VIdominant = true;`
10: **end if**
11: **if** `VIFringe == ture` **then** ▷ iteration for fringe firms
12: **for** $\hat{s}_t = (\theta(f_t), d_t, z_t) \in \hat{\mathcal{S}}$ **do** ▷ iteration over state points in current state space
13: Calculate dominant firms' transitions $d_{t+1} \mid d_t$:
14: Number of active dominant firms = number of non-zero values in d: `activeDominant`;
15: Extract firms' investment strategies from `strategyDominant` store in Algorithm 4;
16: **for** `l=1; l<` $3^{\texttt{activeDominant}}$`; l++` **do** ▷ see Figure 6
17: Construct the l-th `dominant` vector;
18: **if** `activeDominant = D` **then**
19: No fringe firm will become dominant in the next period;
20: Compute the probability that this vector will occur under transition probabilities;
21: **end if**
22: **if** `activeDominant` $<$ `D` **then**
23: Then there's probability that fringe firms will become dominant in the next period;
24: Recalculate the probability vector given that $x_{f\to d}$ replace x_0 in the vector;
25: **end if**
26: **end for**
27: **return** the new probability transition vector `stateTransitions`;
28: **do** Algorithm 10: Update fringe firm's value function
29: **return** $V_f^{i+1} = V_f$ ▷ value function updated
30: **return** `difFringe` $= \max_{i \in \text{iteatiom}} \{V_f^{i+1} - V_f^i\}$
31: **end for**
32: **end if**
33: **if** `VIDominant == true` **then**
34: **do** Algorithm 10: Update dominant firm's value function
35: **return** $V_d^{i+1} = V_d$ ▷ value function updated
36: **return** `difDominant` $= \max_{i \in \text{iteatiom}} \{V_d^{i+1} - V_d^i\}$
37: **end if**
38: `dif = max(difFringe, difDominant)`
39: **end while**

=0

Notes:

1. **Line 5-10**: This make sure that only when the differences in value function for this firm type is larger than the other, and this value is larger than the tolerance level, will this type of value function be updated (this can save computation time compared to use the `If min(difFringe, difDominant) > tol` { `VIFringe=true, VIdominant=true`} statement).
2. **Line 12-18**: Construct the probability (weights) distribution for $\hat{s}_{t+1} \mid \hat{s}_t$. This is important since firms will keep track of the detailed state of dominant firms.

3. **Line 16**: Since each dominant firm has 3 possible direction of movements (up, down, stay), there's $3^{\texttt{activeDominant}}$ number of possible dominant firms' profile after transitions are realized.

Algorithm 11 Update fringe firm's value function

1: Calculate firms' expected continuation value;
2: **for** $x \in \mathcal{X}_f$ **do**
3: $\quad EV^{\text{mid}}(x, s) = \sum_{s' \in \mathcal{S}} V(x' = x, s' \mid s) \cdot p(s' \mid s)$ ▷ stay
4: $\quad$ **if** $x > x_1$ **then**
5: $\quad\quad EV^{\text{low}}(x, s) = \sum_{s' \in \mathcal{S}} V(x' = x - 1, s' \mid s) \cdot p(s' \mid s)$ ▷ down
6: $\quad$ **else**
7: $\quad\quad EV^{\text{low}}(x, s) = EV^{\text{mid}}(x, s)$
8: $\quad$ **end if**
9: $\quad$ **if** $x < x_3$ **then**
10: $\quad\quad EV^{\text{high}}(x, s) = \sum_{s' \in \mathcal{S}} V(x' = x + 1, s' \mid s) \cdot p(s' \mid s)$ ▷ up
11: $\quad$ **else** ▷ for fringe firms at the auxiliary transition state
12: $\quad\quad$ Calculate the new transition probability conditional on fringe firm becoming dominant:
13: $\quad\quad$ New transition probability vector: $p^{f \to d}(s' \mid s)$;
14: $\quad\quad$ Calculate the new fringe moments and the new industry state vector;
15: $\quad\quad$ Taking into account the probability to become dominant:

$$\begin{aligned} EV^{\text{high}}(x_4, s) =& p(f \to d) \cdot \left(\sum_{s' \in \mathcal{S}} V(x' = x_4, s' \mid s) \cdot p^{f \to d}(s' \mid s) \right) + \\ & (1 - p(f \to d)) \cdot \left(\sum_{s' \in \mathcal{S}} V(x' = x_3, s' \mid s) \cdot p(s' \mid s) \right) \end{aligned}$$

16: $\quad$ **end if**
17: **end for**
18: **do** Algorithm 4: Bellman equation solver for fringe firms (Line 10-15).
19: **return** $V_f^{i+1} = V_f(x, s)$ ▷ Value function updated
20: **return** `VIstategyFringe` $= \iota^*(x, s)$; ▷ Strategies updated

Notes:

1. **Line 3**: The industry state transition probability $p(s' \mid s)$ is derived from the calculation in Line 16-26 in Algorithm 9.
2. **Line 14**: Need to substract the fringe firm from moment when becoming dominant.
3. **Line 15**: Note that if fringe firms at the auxiliary transition state do not become dominant at the next period, then they cannot remain at the auxiliary state and have to move down to x_3 in the next period. $p(s' \mid s)$ is the one calculated in Line 2 in Algorithm 8.

5 Mathmetical Proofs

5.1 Capacity-Constrained Quantity Competition

In capacity competition model specified in Example 6.2 in the paper, at each period, firms play a capacity-constrained quantity game. The inverse demand function is given by:

$$P(Q) = \frac{e}{f} - \frac{Q}{mf} \tag{1}$$

where Q is total output, and e, m, f are positive constants.

The Nash equilibrium quantity for a firm in state x is denoted $q^*(x)$. The firm's profit is:

$$\pi(x_{it}, s_t) = P\left(\sum_{x\in\mathcal{X}} s_t(x)q^*(x)\right) q^*(x_{it}). \tag{2}$$

All fringe firms are small and produce at full capacity in equilibrium, thus equation (2) can be rewritten as:

$$\pi(x_{it}, s_t) = P\left(\sum_{y\in\mathcal{X}_f} f_t(y)\bar{q}(y) + \sum_{j\in D_t} q^*(x_{jt})\right) q^*(x_{it}). \tag{3}$$

5.2 Quality-Ladder Price Competition

In quality ladder model, each firm's state variable x_{it} is a natural number that represents the quality level of its product.[3] Firms compete each period à la Bertrand and receives the equilibrium profits.

5.2.1 Deduction 4.2.1: Fringe Firm's Problem

As stated in Example 6.1 in Ifrach and Weintraub (2017), consumer j's utility takes the following logit form:

$$u_{ijt} = \alpha_1 \ln(x_{it}) + \alpha_2 \ln(Y - p_{it}) + v_{ijt},\ \ i \in S_t,\ \ j = 1, ..., m \tag{4}$$

The Gumbel distribution assumption imposed on v_{ijt} yields the following expected profits for each firm:

$$\pi(x_{it}, s_t) = m \cdot (p^*(x_{it}) - c)\frac{K(x_{it}, p^*(x_{it}))}{1 + \sum K(x, p^*(x))},\ \forall\ i \in S_t \tag{5}$$

where m is the market size, c represents for constant marginal cost of production, $K(x, p) = \exp(\alpha_1 \ln(x) + \alpha_2 \ln(Y - p)) = x^{\alpha_1}(Y - p)^{\alpha_2}$.

[3] For clarity, as stated in Example 6.1, do not consider aggregate shocks in this example.

A fringe firm is one of many small firms, so it takes the market environment as given. Fringe firms have no market power. In other words, firm $i \in S_f$ believes it cannot influence the aggregate demand denominator $1 + \sum K$ (the terms involving competitors and the outside option). Practically, this means in equation (5), the fringe firm treats the denominator as approximately constant when choosing its optimal price level. Thus, the static profit maximizing problem can be written as:

$$\max_{p_i} \quad \pi(p_i) = (p_i - c) \cdot (Y - p)^{\alpha_2} \cdot C \tag{6}$$

$C = \frac{m \cdot x^{\alpha_1}}{1+\sum K}$ is the term that do not affect the calculation of p^*. FOC of the problem:

$$\frac{\partial\ \pi(p)}{\partial\ p} = (Y - p_i)^{\alpha_2} + (p_i - c) \cdot \alpha_2 \cdot (Y - p_2)^{\alpha_2 - 1} \tag{7}$$

Thus in equilibrium, fringe firms all set their price level as:

$$p_f^* = \frac{Y + c\alpha_2}{1 + \alpha_2} \tag{8}$$

5.2.2 Deduction 4.2.2: Dominant Firm's Problem

Now move to the problem for dominant firms. Dominant firms compete à la Bertrand-Nash in prices. Dominant firms' expected profits are given by:

$$\pi\big(x_{it}, s_t\big) = m\big(p^*\big(x_{it}\big)\ -\ d\big)\ \frac{K\big(x_{it},\, p^*(x_{it})\big)}{1 + \big(Y - p^*\big)^{\alpha_2} \sum_{y \in \mathcal{X}_f} y^{\alpha_1}\, f_t(y) + \sum_{j \in D_t} K\big(x_{jt},\, p^*(x_{jt})\big)} \tag{9}$$

Note that $\big(Y - p^*\big)^{\alpha_2} \sum_{y \in \mathcal{X}_f} y^{\alpha_1}\, f_t(y)$ in the denominator of equation (9) stands for the fringe moments [4]. For fringe firms, this term is derived from:

$$\sum_{x \in \mathcal{X}_f} K(x, p) = \sum_{x \in \mathcal{X}_f} x^{\alpha_1} (Y - p)^{\alpha_2} = (Y - p)^{\alpha_2} \cdot \underbrace{\sum_{x \in \mathcal{X}_f} x^{\alpha_1}}_{\alpha_1 \text{ -th Fringe state moment}} \tag{10}$$

where $\sum_{x \in \mathcal{X}_f} x^{\alpha_1}$ represents the moments. If we transform x into y by using some linear transformations and weight each state by using some density function f, then this part becomes $\tilde{\theta}_t = \sum_{y \in \mathcal{X}_f} y^{\alpha_1} f_t(y)$ occurred in the paper. In Algorithm 2, equation (10) is represented by the term `momenAttraction`, thus, the fancy equation (9) can be simplified into:

$$\pi = m(p^* - c) \frac{x^{\alpha_1} (Y - p)^{\alpha_2}}{1 + \text{momentAttraction} + \sum_{j \in D_t} x^{\alpha_1} (Y - p)^{\alpha_2}} \tag{11}$$

Abstract away from the term $m \cdot x^{\alpha_1}$ that will not affect calculating the FOC with respect to p:

$$\tilde{\pi} = (p - c) \frac{(Y - p)^{\alpha_2}}{1 + \text{momentAttraction} + \sum_{j \in D_t} x^{\alpha_1} (Y - p)^{\alpha_2}} \tag{12}$$

[4] In code, this term is calculated as: $N(x_j, p_j) = (x_j/\psi + 1)_1^{\gamma} \cdot (Y - p_j)_2^{\gamma}$

Rewrite equation (12) in terms of $Y - p = z$:

$$\tilde{\pi}(z, x) = (Y - c - z) \cdot \frac{z^{\alpha_2}}{1 + \text{momentAttraction} + \sum_{j \in D_t} x^{\alpha_1} z^{\alpha_2}} \tag{13}$$

Take the derivative of equation (13) with respect to z:

$$\begin{aligned}\frac{\partial}{\partial\ z} = &- \frac{z^{\alpha_1}}{1 + \text{momentAttraction} + \sum_{j \in D_t} x^{\alpha_1} z^{\alpha_2}} \\ &+ (Y - c - z) \cdot \frac{\alpha_2 z^{\alpha_2 - 1} \cdot (1 + \text{momentAttraction} + \sum_{j \in D_t} x^{\alpha_1} z^{\alpha_2}) - z^{\alpha_1} \cdot (\partial \sum_{j \in D_t} x^{\alpha_1} z_j^{\alpha_2} / \partial z_j)}{(1 + \text{momentAttraction} + \sum_{j \in D_t} x^{\alpha_1} z^{\alpha_2})^2}\end{aligned} \tag{14}$$

After some simplifications:

$$\Longrightarrow \frac{-z}{(Y - c - z)} + \alpha_2 \cdot (1 + \frac{x^{\alpha_1} z^{\alpha_2}}{(1 + \text{momentAttraction} + \sum_{j \in D_t} x^{\alpha_1} z^{\alpha_2})}) \tag{15}$$

Thus, as explained in Algorithm 2, the value of equation (15) for each dominant firm is then used to construct a Euclidean distance to compare with the tolerance level at each iteration.

5.2.3 Deduction 4.2.3: Hessian Matrix and the Newton's Update Rule

First calculate the hessian matrix. The hessian matrix will be a $\bar{D} \times \bar{D}$ matrix where each element in the matrix stores the second-order partial derivative of the objective function with respect to the corresponding pair of variables.

$$H(\pi(z_1, z_2, \ldots, z_{\bar{D}})) = \begin{pmatrix} \frac{\partial^2 \pi}{\partial z_1^2} & \frac{\partial^2 \pi}{\partial z_1 \partial z_2} & \cdots & \frac{\partial^2 \pi}{\partial z_1 \partial z_{\bar{D}}} \\ \frac{\partial^2 \pi}{\partial z_2 \partial z_1} & \frac{\partial^2 \pi}{\partial z_2^2} & \cdots & \frac{\partial^2 \pi}{\partial z_2 \partial z_{\bar{D}}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 \pi}{\partial z_{\bar{D}} \partial z_1} & \frac{\partial^2 \pi}{\partial z_{\bar{D}} \partial z_2} & \cdots & \frac{\partial^2 \pi}{\partial z_{\bar{D}}^2} \end{pmatrix}$$

where the diagonal elements equals to the simple second-order directives:

$$\begin{aligned} H_{i,i} &= \frac{\partial^2 \pi}{\partial^2 z_i}, \quad i \in D \\ &= -\frac{Y - c}{(Y - c - z)^2} + \alpha_2^2 \cdot \frac{x^{\alpha_1} z^{\alpha_2 - 1}}{(1 + \text{momentAttraction} + \sum_{j \in D_t} x^{\alpha_1} z^{\alpha_2})^2} \end{aligned} \tag{16}$$

and the elements off the diagonal equals to:

$$\begin{aligned} H_{i,j} &= \frac{\partial^2 \pi}{\partial z_i z_j}, \quad i, j \in D, \quad i \neq j \\ &= \alpha_2^2 \cdot \frac{x_i^{\alpha_1} x_j^{\alpha_2} z_i^{\alpha_2} z_j^{\alpha_2 - 1}}{(1 + \text{momentAttraction} + \sum_{j \in D_t} x^{\alpha_1} z^{\alpha_2})^2} \end{aligned} \tag{17}$$

Now, using the Newton update method, the new update for each parameter z_i is given by:

$$z_i^{\text{new}} = z_i^{\text{old}} - (H^{-1} \cdot \nabla \pi(z))_i \tag{18}$$

where: H^{-1} is the inverse of the Hessian matrix, $\nabla\pi(z)$ is the gradient of the objective function, containing the first-order partial derivatives.

This update rule adjusts the value of z_i iteratively by moving in the direction where the function's curvature (as described by the Hessian matrix) and gradient are both considered, which helps converge more quickly to the optimal values for parameters z_i.

5.3 Optimal Investment Decisions

Assume firm's state follows the standard quality ladder transitions:

$$x_{t+1} \mid x_t, \iota = \begin{cases} \min\{x_{t+1}, \bar{x}\}, & \text{w.p. } \frac{a\iota}{1+a\iota} \\ x_t, & \text{w.p. } \frac{1-\delta'+(1-\delta)a\iota}{1+a\iota} \\ \max\{x_{t-1}, \underline{x}\}, & \text{w.p. } \frac{\delta'}{1+a\iota} \end{cases} \tag{19}$$

If firm do not decide to invest ($\iota = 0$), then:

$$W(\iota = 0) = \beta \cdot \left((1-\delta')W^{\text{mid}} + \delta' W^{\text{low}}\right) \tag{20}$$

If firm decide to invest 1 unit ($\iota = 1$), then:

$$W(\iota = 1) = \beta \cdot (\frac{\delta'}{2}W^{\text{low}} + \frac{2-\delta'-\delta}{2}W^{\text{mid}} + \frac{\delta}{2}) \tag{21}$$

Therefore, the marginal benefit of investing in one unit: [5]

$$\Delta = W(\iota = 1) - W(\iota = 0) = \frac{1}{2} \cdot (\delta W^{\text{high}} + (\delta' - \delta)W^{\text{mid}} - \delta' W^{\text{low}}) \tag{22}$$

where $A = 2 \cdot \Delta$. The marginal benefit of investing in ι units:

$$W(\iota = \iota) - W(\iota = 0) = A \cdot \frac{a\iota}{1+a\iota} \tag{23}$$

The optimal investment ι^* occurs at the points where the marginal net benefit of investing in ι^* is maximized:

$$\max_{\iota} \quad \beta A \cdot \frac{a\iota}{1+a\iota} - d \cdot \iota \tag{24}$$

where d is the constant investment cost. Solution:

$$\iota^* = -\frac{1}{a} + \frac{1}{a}\sqrt{\frac{a \cdot \beta A}{d}} \tag{25}$$

[5] A in algorithm 4 equals 2Δ but this cause no difference when comparing to zero.

6 Graphical Explanation

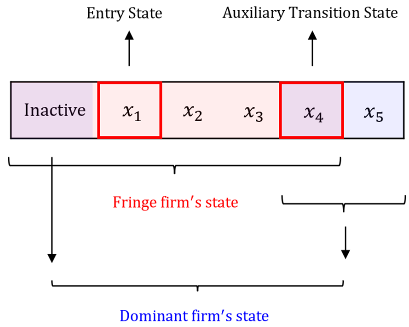

Figure 1. Firm's State.

Notes: Figure 1 displays the state vector for fringe firms and dominant firms. Fringe firms have 4 physical state plus one inactive state while dominant firms have 2 physical state plus also one inactive state. Thus the state vector for fringe firms and dominant firms can be written as: $[\text{inactive}, x_1, x_2, x_3, x_4]$ and $[\text{inactive}, x_4, x_5]$, respectively. The overlapped x_4 state is the auxiliary state in fringe state vector to consider tier transitions, and it's also the state when fringe firms first become dominant. Correspondingly, x_3 is the state if dominant firms become fringe. $x_e = x_1$ is potential entrants' entry state, therefore, all new entrants will be fringe firms.

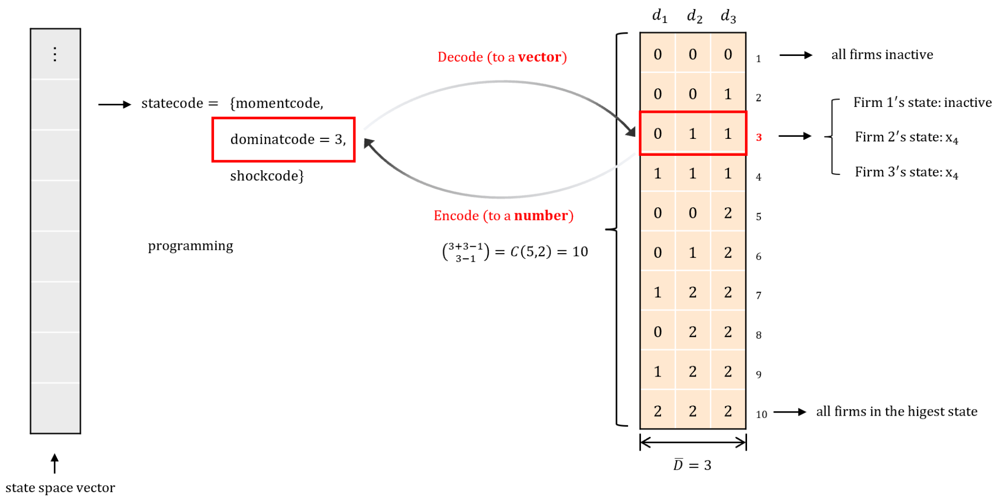

Figure 2. Decoding and Encoding Dominant Firm's State.

Notes: Figure 2 further illustrates the dominant firms' state matrix displayed in Figure 1. Each row in the dominant state matrix should record each dominant firm's state, so this matrix will have a dimension of $\bar{D} \times C(\mathbb{N}^{|\mathcal{X}_d|} + 1 + \bar{D} - 1, \mathbb{N}^{|\mathcal{X}_d|} + 1 - 1) = 3 \times 10$. This is the same combinatorics as the answer to "How many possible ways to put $\bar{D}$ indistinguishable stars (firms) into $\mathbb{N}^{|\mathcal{X}_d|} + 1$ distinguishable boxes (states)?" as in the "Stars and Bars" problem. It's not obvious to conduct the decode and encode process shown in Figure 2 and the coding process is displayed in Code 6.1: Decoding and Encoding the Dominant State.

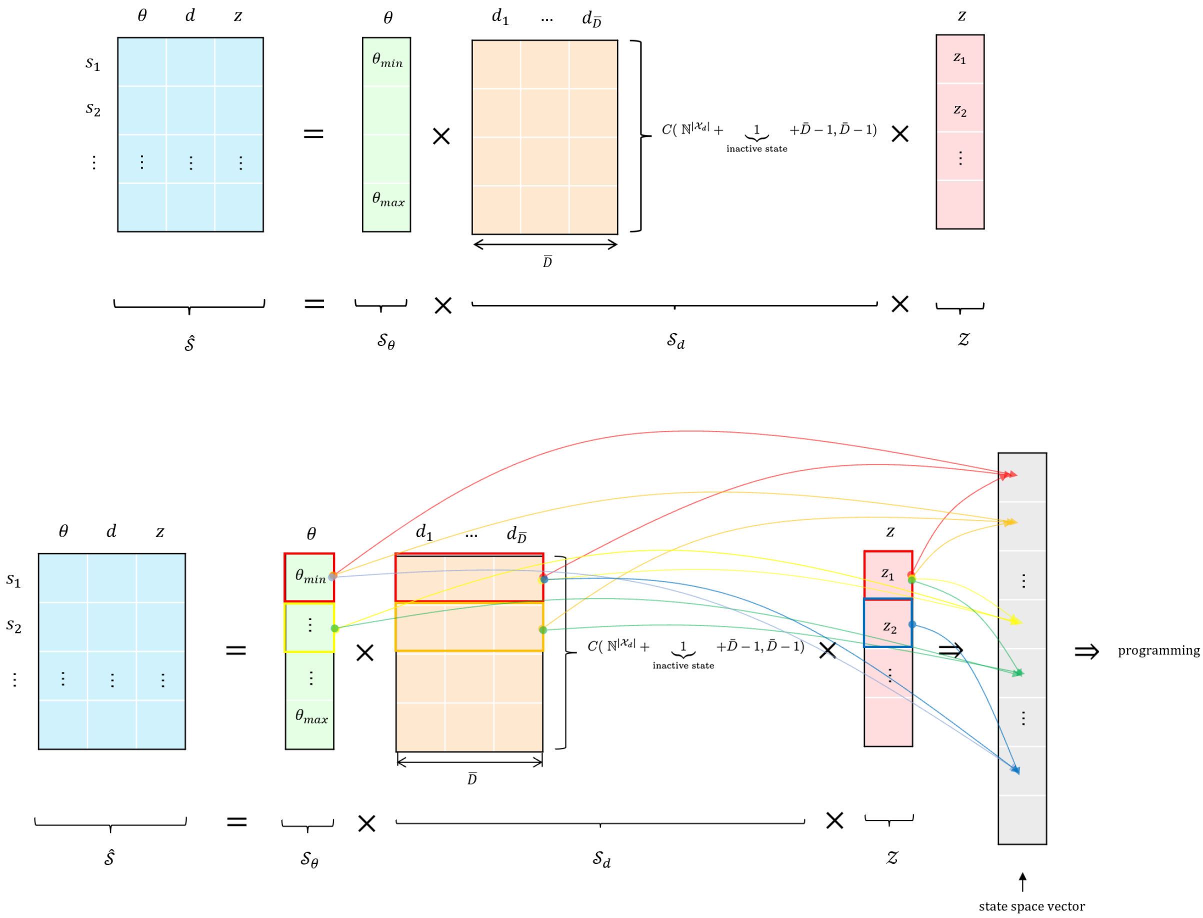

Figure 3. State space decomposition.

Notes: Left-hand side matrix in Figure 1 illustrates the moment-based state space used in the model where each row in the blue matrix represents a slice of a specific composite state $\hat{s}_t = \{\theta_t, d_t, z_t\}$, with θ_t representing the moment, d_t representing the profile/distribution of dominant firms over $\mathcal{X}_d$ and the scalar value of aggregated shock z_t. The algorithm needs a one-dimensional array to handle each state computationally. Hence, we "unfold" every $\hat{s}_t$ into a $S \times 1$ vector where $S = \mathcal{S}_\theta \times \mathcal{S}_d \times \mathcal{Z}$. The $\mathcal{S}_\theta$ is a $M \times 1$ dimension vector where M is the discretized grid points that the value of the fringe moments will possibly take. In other words, the moment θ_t, the dominant-firm distribution d_t (including any inactive states), and the shock z_t are all mapped into a single index space so that we can systematically iterate over these states in the computational routine.

The colored arrows in Figure 1 illustrate how each component from the composite state is connected and combined into this final one-dimensional representation on the right-hand side (in the order of $\succeq$ under A.0 in Ericson and Pakes (1995)). This vectorized format is then fed into the dynamic programming or equilibrium solver with a single industry integer-value index, ensuring a consistent way to track and update the model's state across time.

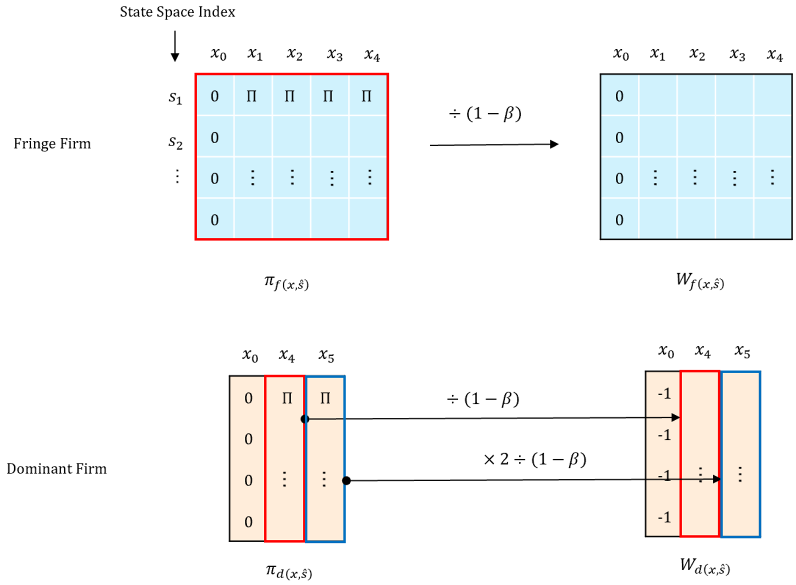

Figure 4. Value Matrix Initialization.

Notes: Figure 4 illustrates how the static equilibrium profit matrix helps to construct and initialize the value matrix W. Fringe firms often change their market position frequently, and differences in their profit levels are smaller. Their future positions are uncertain and less stable, so it's enough to initialize their value simply as current profit divided by $1-\beta$. Dominant firms have stable and powerful market positions. Firms in the highest dominant state don't just have higher profits; they also expect to remain dominant longer into the future, giving them extra long-term advantage. Therefore, the initial value for the top dominant state is multiplied by an extra factor (like 2) to clearly reflect this extra long-term stability and future advantage which also helps to reduce the convergence time.

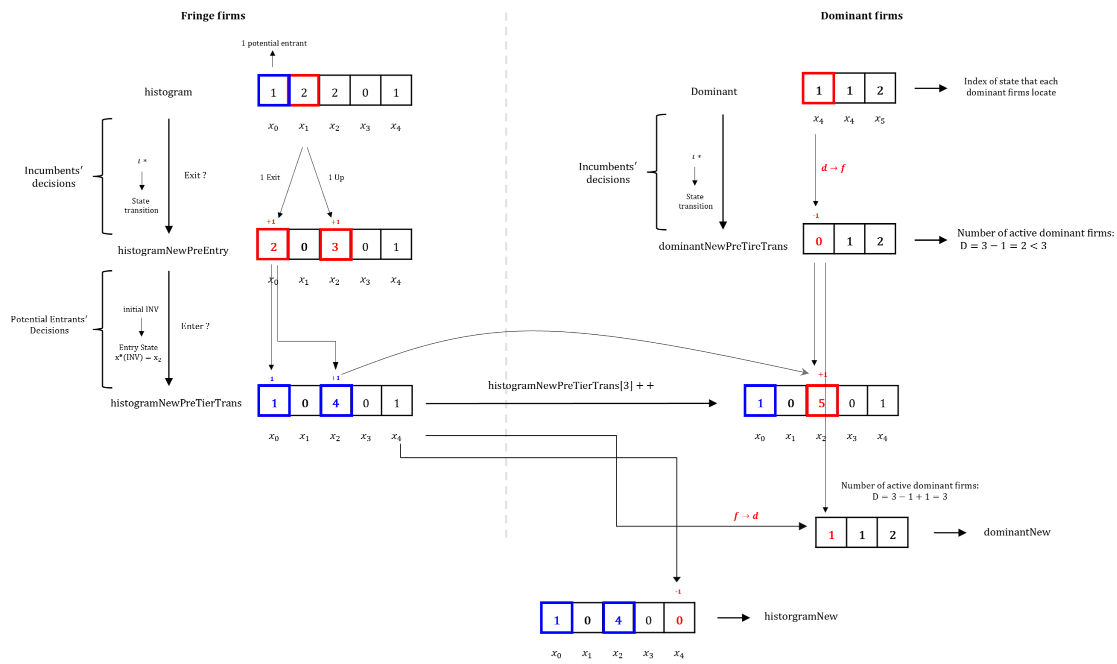

Figure 5. Firm's tier transitions.

Notes: Figure 5 illustrates how firms transit between fringe and dominant states. The detailed descripion of Figure 5 and its relation with the model's timing assumptions are summarized as follows:

1. Incumbent fringe firms' exit and investment decisions

   To assist illustration, assumed that $M = 9$ and $D = 3$, therefore, there are 3 dominant firms (always) active in the industry and the whole industry can support no more that $M = 9$ firms. Assume that the fringe firms' distribution is now:

   ```
   histogram = [1,2,2,0,1]
   ```

   where each element in this `histogram` represents the number of fringe firms at this fringe state. Since `histogram[0]=1`, there's one inactive, potential entrant. After exit and investment decisions are made and realized at the beginning of next period, assume on firm at x_1 exit and one firm also at x_1 move up to x_2 along the ladder. Therefore, the histogram now becomes:

   ```
   histogramNewPreEntry = [2,0,3,0,1]
   ```

2. Potential entrants' entry decisions Potential entrants in the model are inactive fringe firms. Assume that the inactive fringe firm decides to enter and make some initial investment which turns out to successfully move $x^e = x_1$ to $x^e = x_2$. Therefore, after exit and entry decisions, the historgram now becomes:

   ```
   histogramNewPreTierTrans = [1,0,4,0,1]
   ```

3. Dominant firms' transitions Now move to the case for dominant firms. Note that the `histogram` vector records the joint distribution while the `dominant` vector is a $1 \times D$ vector that record each firm's state:

   ```
   dominant = [1,1,2]
   ```

   Assume that after investment decisions are made, dominant firm 1 move from x_4 to x_0, that is, this dominant firm become "inactive" in dominant but transition and become "active" as a fringe firm ($d \to f$). Thus, there's only two dominant firms active in the industry. If there's active fringe firms at the auxiliary transition state, then this fringe firm with become dominant with state x_4 and the number of active dominant firm goes back to three again:

   ```
   histogramNew = [1,0,4,0,0],   dominantNew = [1,1,2]
   ```

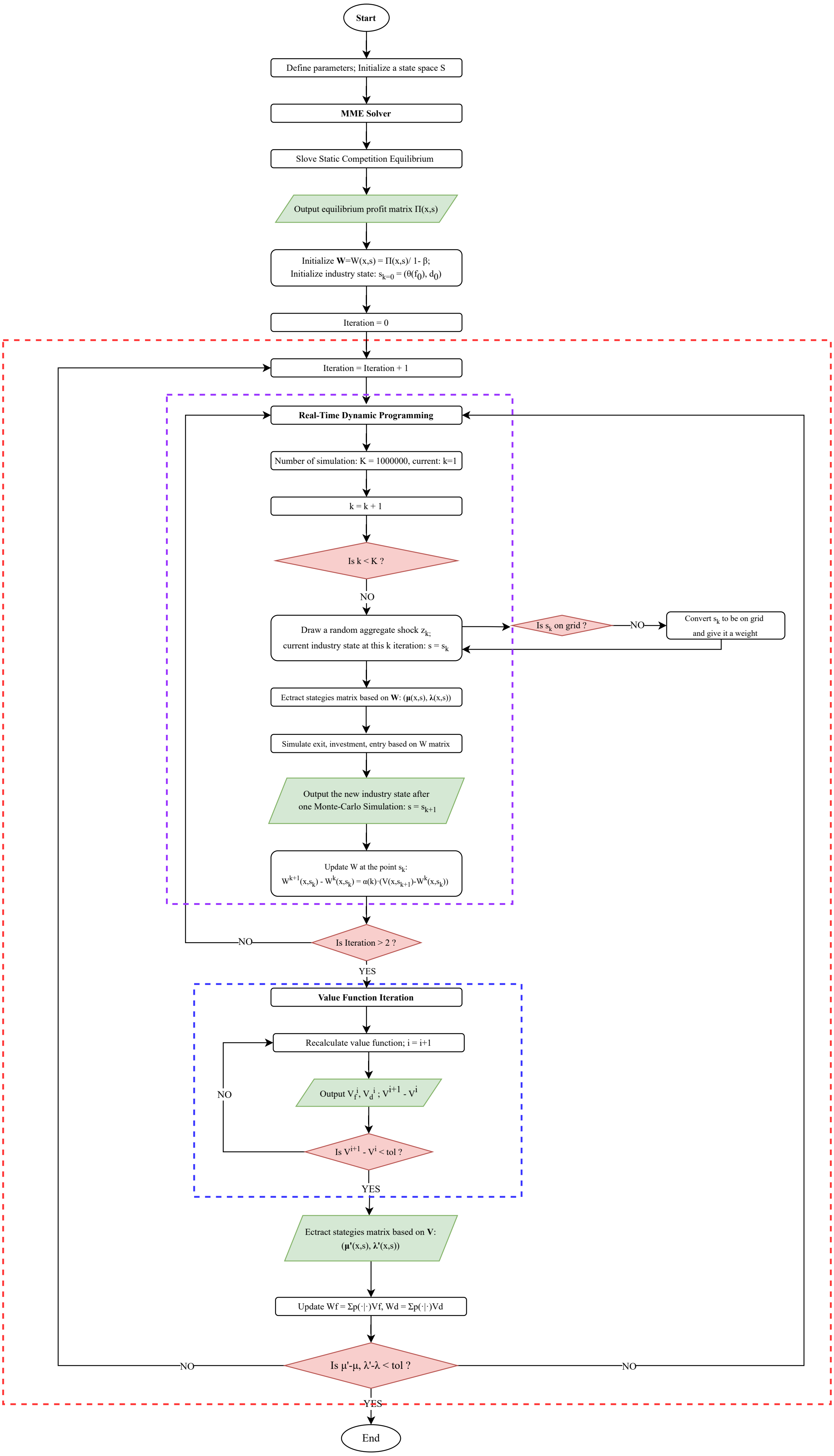
Start
Define parameters; Initialize a state space S
MME Solver
Slove Static Competition Equilibrium
Output equilibrium profit matrix Π(x,s)
Initialize **W**=W(x,s) = Π(x,s)/ 1- β;
Initialize industry state: $s_{k=0}$ = (θ(f_0), d_0)
Iteration = 0
Iteration = Iteration + 1
Real-Time Dynamic Programming
Number of simulation: K = 1000000, current: k=1
k = k + 1
Is k < K ?
NO
Draw a random aggregate shock z_k;
current industry state at this k iteration: s = s_k
Is s_k on grid ?
NO
Convert s_k to be on grid and give it a weight
Ectract stategies matrix based on **W**: (**μ**(x,s), **λ**(x,s))
Simulate exit, investment, entry based on W matrix
Output the new industry state after one Monte-Carlo Simulation: s = s_{k+1}
Update W at the point s_k:
$W^{k+1}(x,s_k) - W^k(x,s_k) = α(k)·(V(x,s_{k+1})-W^k(x,s_k))$
Is Iteration > 2 ?
NO
YES
Value Function Iteration
Recalculate value function; i = i+1
Output V_f^i, V_d^i ; V^{i+1} - V^i
Is V^{i+1} - V^i < tol ?
NO
YES
Ectract stategies matrix based on **V**:
(**μ'**(x,s), **λ'**(x,s))
Update Wf = Σp(·|·)Vf, Wd = Σp(·|·)Vd
Is μ'-μ, λ'-λ < tol ?
NO
NO
YES
End

Figure 1:

Figure 5. Flow chart for the whole algorithm.

7 Coding

7.1 Code 6.1: Decoding and Encoding the Dominant State

```
// ==================================================================

  // State space for dominant firms

  // stateSpaceSizeDominant = Number of distributions of M firms across N dominant states
  //                        = C(N+M-1, M-1)

  void setStateSpaceDominant(){

     // Define notation

     int M = parameters.barXd;    // barXd=3, 1 inactive state + 2 physical states
     int N = parameters.nDominant; // number of dominant firms (always)
     if(N>0){

        // Set up binom - matrix that used for coding and encoding

        int[][] binom = new int[N+M][N+M+1];

        for(int i = 1; i <= N+M; i++){
           binom[i-1][i] = 1;
        }

        int i = 1;

        while(i< N+M){
           for (int j = 1; j<=i; j++){
              binom[i][j] = binom[i-1][j]+binom[i-1][j-1];  // binom[i][j]=C(i,j-1)
           }
           i++;
        }

        stateSpaceSizeDominant = binom[N+M-1][M]; // Note: an int number denoted for
            state size
        this.binom = binom;

     } else{
        stateSpaceSizeDominant = 1;
     }

  }

  // -----------------------------------
  // stateSpaceDominant = C(N+M-1, M-1)
  // -----------------------------------

  // The "Stars and Bars" Problem:
  // Q: How many ways are there to put n indistinguishable balls into k distinguishable
      bins?

  // A: C(n+k-1, k-1)

  // Just think of balls as M firms and bins as states N
  // Same problem as putting M firms into N states
```

```
// See footnote 8: firm are indistinguishable(!) in identity

 // stateSpaceSizeDominant[dominantCode] = { dominant[0], dominant[1], ...,
     dominant[nDominant-1] }

      dominantCode = dominantCode -1;

      for(int n = 0; n<parameters.nDominant; n++){
         dominant[n]=1;
      }

      for(int n = parameters.nDominant+1; n>=2; n=n-1){
         while(binom[dominant[n-1-1]+n-1-1][dominant[n-1-1]+1-1]<=dominantCode){
            dominant[n-1-1]=dominant[n-1-1]+1;
         }
         dominantCode = dominantCode - binom[dominant[n-1-1]+n-2-1][dominant[n-1-1]-1];
      }

      for(int n = 0; n < nDominant; n++){
         dominant[n] = dominant[n]-1;
      }

            int dominantEncode(){

   Arrays.sort(dominant);

   int j = 0;

   for(int n = parameters.nDominant + 1;n >=2; n=n-1){
      j=j+binom[dominant[n-1-1]+n-2][dominant[n-1-1]];
   }

   int dominantCode = 1 +j;

   return dominantCode;
}


// ==================================================================
```

References

Barto, A. G., Bradtke, S. J., & Singh, S. P. (1995). Learning to act using real-time dynamic programming. *Artificial intelligence*, *72*(1-2), 81–138.

Ericson, R., & Pakes, A. (1995). Markov-perfect industry dynamics: A framework for empirical work. *The Review of Economic Studies*, *62*(1), 53–82.

Ifrach, B., & Weintraub, G. Y. (2017). A framework for dynamic oligopoly in concentrated industries. *The Review of Economic Studies*, *84*(3), 1106–1150.

Pakes, A., & McGuire, P. (2001). Stochastic algorithms, symmetric markov perfect equilibrium, and the 'curse'of dimensionality. *Econometrica*, *69*(5), 1261–1281.