Minority Game

Cellerini Claudio, Freni Salvatore, Monaci Luca

January 2024

1 Introduction

Real market dynamics pertain to the functioning and evolution of economic and financial markets over time. It's impossible to have an analytical solution to this problem because it depends on a large number of variables. Therefore, it can be approximated by stochastic models.
Physicists have tried to develop simple interacting agents models that could emulate the key characteristics of real economic markets. One of the simplest is the so called Minority Game (MG).
The aim of this work is to create such a model, validate it by comparing the results obtained by our simulations with those present in the literature, and analyze its merits and discrepancies in comparison to the real-world data features on prices' returns.

2 Minority Game

The original Minority Game [1, 5] is composed by N agents that can operate in the market (N must be an odd integer). At each time step of the game, each of the N agents takes an action $a_i(t)$ with $i \in \{1, ..., N\}$ and he decides either to buy an asset ($a_i(t) = 1$) or to sell it ($a_i(t) = -1$). The agents who take the minority action win, whereas the majority looses. After a round, the total activity is calculated as

$$A(t) = \sum_{i=1}^{N} a_i(t) \tag{1}$$

The way agents choose their action $a_i(t)$ is by inductive reasoning: it is assumed that the players have limited analyzing power and they can only retain the last M winning groups. A strategy s is just a mapping from the sequence of the last M winning groups to the action of the agent, i.e. a table which tells the agent what to do as a function of the input signal (the last M winning groups). Since there are 2^M possible inputs for each strategy, the total number of possible strategies for a given M is 2^{2^M}. At the beginning of the game each agent is given a set of S strategies randomly drawn from the set of all the possible strategies. This assignment is different for each agent and thus, agents may or may not share the same set of strategies. Note that each strategy is determined by a 2^M dimensional vector $\vec{s}_i^{\,j}$ whose components are the output of strategy $\jmath$ of agent i.

input	output
−1 −1 −1	−1
−1 −1 +1	−1
−1 +1 −1	+1
−1 +1 +1	−1
+1 −1 −1	−1
+1 −1 +1	+1
+1 +1 −1	−1
+1 +1 +1	+1

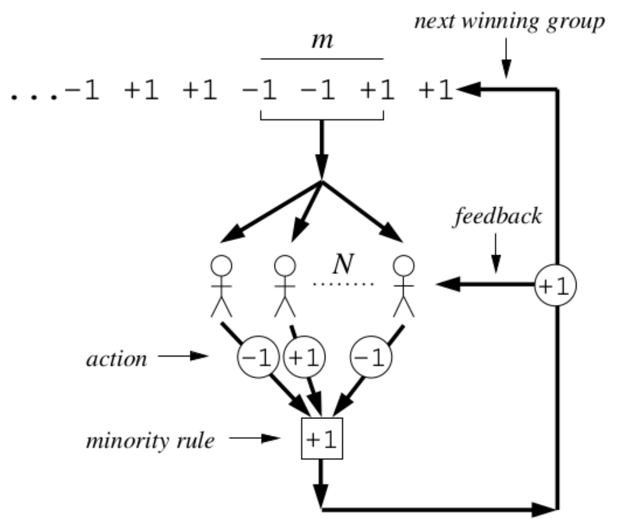


Figure 1: An example of strategy with $M = 3$ from [5]

Thus, at each time step, the prediction of a strategy is given by its $\mu(t)$, which is a number whose binary representation corresponds to the last M winning groups. If we denote with $\vec{I}(t)$ the vector whose components are zero except the $\mu(t)$-th one, which is 1, then the prediction of strategy $\vec{s}_i^{\,j}$ is given by $\vec{s}_i^{\,j} \cdot \vec{I}(t)$.

The minority rule sets the comfort level at $A(t) = 0$, so that agent is given a payoff for the selected strategy

$$\Delta U_{s,i}(t) = -a^{\mu(t)}_{s_i(t),i} g[A(t)] \tag{2}$$

at each time step with g an odd function of A(t). The MG properties are qualitatively independent of the precise analytical form of $g(x)$. (In particular, in our models we used the following definition: $g(x) = sign(x)$).

It is important to note that various possibilities to select the optimal strategy exist. The main approach consists of selecting

$$s_i(t) = \underset{s=\{1,\ldots,S\}}{\text{argmax}}\, U_{s,i}(t) \tag{3}$$

meaning that the chosen strategy is the one with the highest score. Alternatively, based on other possible methods, the agent determines its decision $s_i(t)$ with probability

$$Prob\{s_i(t) = s\} = \frac{exp[\Gamma_i U_{s,i}(t)]}{\sum_{s'} exp[\Gamma_i U_{s',i}(t)]} \tag{4}$$

where $\Gamma_i > 0$ can be considered as an inverse temperature, for this reason, this formulation goes by the name of "Thermal Minority Game". The first method was introduced in the original Minority Game and corresponds to $\Gamma_i = \infty$.

2.1 Volatility

The source of randomness is in the choice of $\mu(0)$ and $s_i(0)$, and this leads $U_{s,i}(t)$ to be fast fluctuating. Now, we define the volatility, i.e. the fluctuations of attendance $A(t)$ as:

$$\sigma^2 = \langle A^2(t) \rangle \tag{5}$$

We decided to use the volatility to validate our model with respect to the results found in literature. As shown in [8, 5], the macroscopic behaviour of the system does not depend on the parameters N and M, but rather relies on the ratio $\alpha = \frac{2^M}{N}$. Plotting the quantity $\frac{\sigma^2}{N}$ against α reveals a phase transition. Going into detail, there is a critical value of $\alpha = \alpha_c$ that divides the plot into two distinct phases: the first called "symmetric phase" (for $\alpha < \alpha_c$), and the other one called "asymmetric phase" (for $\alpha > \alpha_c$). The analytically calculated value of α_c is 0.3374.... (for $S = 2$). Considering the random choice limit (where agents make random decisions at every iteration) corresponding to a value of $\frac{\sigma^2}{N} = 1$, for small values of α the volatility of the game is larger than the random choice limit, and agents are said to act in a *worse-than-random* regime. As α increases agents perform better, entering the *better-than-random* regime. [8, 5],

This analytical result for *alpha* is backed by the work done in [3, 6]

2.2 Price returns

To connect the MG with the financial market, price dynamics must be introduced. Several price formulations can be found in the literature, here we used the following one:

$$r(t) = ln\Big(\frac{P(t)}{P(t-1)}\Big) = \frac{A(t)}{\lambda} \tag{6}$$

where λ is an hyperparameter named liquidity. This brings to

$$P(t) = P(t-1)exp\Big[\frac{A(t)}{\lambda}\Big]. \tag{7}$$

Real-world price returns present leptokurtic distributions, i.e. fat tail behaviour, as extensively discussed in existing literature [4]. Kurtosis is indeed used as a proxy for the "tailedness" of the probability distribution of a real-valued random variable.

$$Kurt[x] = \frac{E[(x-\mu)^4]}{(E[(x-\mu)^2])^2} - 3 \tag{8}$$

A null value corresponds to a normal distribution. Distributions with negative excess kurtosis are said to be platykurtic, which implies that they produce fewer and/or less extreme outliers than the normal distribution. Whereas, distributions with a positive excess kurtosis are said to be leptokurtic and are characterized by thicker tails and a sharper peak compared to a normal distribution.

3 Models

We employed two distinct frameworks to implement our model, each embodying a unique modeling approach. The first utilized was the "Mesa" framework, adhering to an object-oriented paradigm, while the second employed, the ”OsBrain” framework, is better suited for modelling multi-agents systems.

3.1 Mesa

The Mesa framework is an open-source Python toolkit designed for Agent-Based Modeling. In this Mesa-driven model, just two main types of agents are involved: the market agent and inductive agents. As mentioned previously, Inductive agents are designed to gather recent information from the market, decide on strategies, and execute actions based on these strategies. The market agent, on the other hand, calculates the overall activity resulting from the actions of the inductive agents and identifies the minority side of the market (buyers or sellers).

A third kind of object, acting as a super-agent and representing the environment that the Market and the Investors inhabit, regulates the model dynamics, which is structured in a sequential, step-by-step, process. The time is managed by a main *for loop*, and the agents are activated by the environment's scheduler using a multi-step logic.

1. Step 1
   - Agents gather the latest information from the model.
   - Each agent selects a strategy based on this information.
   - Agents take action according to the selected strategy.
2. Step 2
   - The market calculates the aggregate activity based on agents' actions.
   - It determines the minority side (buyers or sellers) of the market.
3. Step 3
   - Agents evaluate the performance of their strategies.
   - Agents update their scores based on this evaluation.
4. Step 4

- The market updates its historical record, based on the minority side of the current round.

This approach models the interaction between individual agent decisions and market dynamics effectively. However, this method has its limitations. Mesa does not support direct communication between agents, leading to a less realistic market dynamics representation. Additionally, the sequential activation of agents and market conducted by the scheduler can create performance bottlenecks, particularly if an agent encounters issues. Despite these drawbacks, Mesa's lightweight agent objects, requiring less computing memory compared to other frameworks, enabled us to simulate a larger number of agents and explore a broader range of values for the model parameter 'memory' M, allowing a more refined measure of the model observables. In addition Mesa's multiprocessing capabilities during simulation runs enhance overall efficiency, making it a valuable tool for complex modeling tasks. The approach offers a clear illustration of the interplay between agents' strategies and market's responses, contributing to a comprehensive understanding of the modeled system.

3.2 OsBrain

OsBrain is a general-purpose multi-agent system module written in Python and developed by OpenSistemas. Agents run independently as system processes and communicate with each other using message passing. It uses ØMQ, that is an open-source messaging library and supports common messaging patterns (pub/sub, request/reply, client/server, ...).

The main perk of using osBrain's communication protocols over the Mesa library is that different agents can compute their calculations simultaneously, speeding up the simulations' needed time. This parallel agent structure can also help the simulation to not crash if an agent gets stuck in its computations.

The main disadvantage of using osBrain's communication protocols over the Mesa library is the heavy toll on the computing memory that each agent requires for its instantiation.

- **Model**

At initialization the model must have specified as hyperparameters the length (number of days) of the simulation, the number of investors, the agents' memory, the starting history (which is usually a randomly generated binary string), the choice criterium selected (classic or thermal MG) and, if needed, the inverse temperature.

Then the needed connections between agents needed for communication are established and the simulation is started.

- **Agents**

For the osBrain implementation three kinds of agents are needed:

Timer agent: this agent regulates the time of the simulation by counting and subdividing into executive time-steps each day. When the day-count reaches its maximum set value the simulation is stopped.

Each time-step turnover is then relayed to the market agent, in order to communicate when it should take its actions.

There are three executive phases in each day: Market Opening, Investments Checkup, and Market Closure.

Market agent: this agent is tasked with publishing to all agents the past history at each Market Opening, waiting to receive their daily investments and finally compile the new activity value, which updates the history as a consequence.

Investor agent: this agent is purely reactive and it is tasked to invest every day (buying or selling assets).

The investor agent i creates a list of strategies s_i to use when initialized.

The investments (or actions) $a^{\mu(t)}_{s_i(t),i}\left(t\right)$ are chosen from their respective strategy table $s_i\left(t\right)$ every day by comparing all its strategies' scores $U_{s,i}\left(t\right)$.

The investor is also tasked with the role of updating everyday its strategies' score.

A more detailed description of each agent's actions during the different daily time steps follows in the appendix.

4 Results

In the Mesa framework, for the classic version of the MG, the results were obtained by averaging the results from 10 simulations for each possible combination of the memory and the agent population with $N \in \{101, 601\}$ and $M \in \{2, ..., 19\}$. For the Thermal version, the number of agents was fixed at $N = 301$. In the OsBrain framework results regarding the volatility were obtained by averaging the results from 25 simulations for each possible combination of M and N, for both the classical and the thermal variants, with $M \in \{3, 5, 7, 9, 11\}$, $N \in \{25, 51, 75\}$, while results regarding the price returns were obtained by using the same values for M and keeping fixed $N = 75$. This was done to ease the computational memory requirements while still allowing for a fairly decent coverage of the parameter α useful range. In both frameworks, for the thermal variant of MG, we used a global inverse temperature instead of individuals, i.e. $\Gamma_i = \Gamma \in \{1, 1000\}\ \forall i$. It is also worth mentioning that the updates for the strategies' scores were done by adding (or subtracting) a constant value to the scores of successful (or unsuccessful) strategies, instead of using the formulas 2. This was done to avoid overflow issues during the calculation of the exponential contained in the definition of 4 All agents, in all simulations, were equipped with only $S = 2$ strategies each.

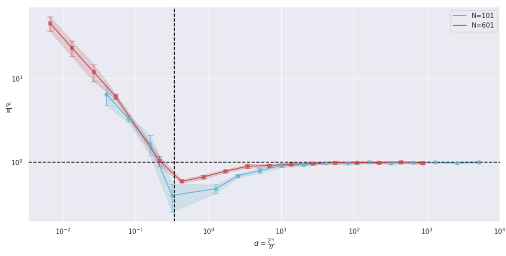
4.1 Discussion

4.1.1 Volatility

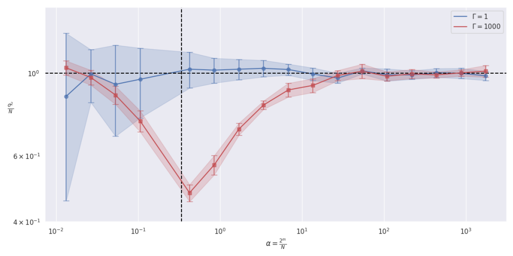
As described above we intend to use volatility's well-known properties to validate our model implementations. The results regarding the high temperature, i.e. ($\Gamma = 1$) are consistent with a constant volatility value, which is to be expected because of the high level of randomness present in the runs. In both frameworks' other results (i.e. classical MG and thermal MG with $\Gamma = 1000$) we can see that the α value for the minimum point, even though it falls in the expected range, is not exactly identified, likely because we considered too few values for the number of agents, leading to a bad graph's resolution.

4.1.2 Price returns' kurtosis

To study the price returns we set the hyperparameter $\lambda = \frac{N}{2}$ (eq. 6) and analyzed their distributions. Our goal is to investigate whether the observed price returns conform to a leptokurtic distribution, which is typical for real markets. For each configuration of the parameters, an average value of kurtosis was obtained over the individual runs to get a proxy of errors of the measurements.



(a) Classic MG



(b) Thermal MG

Figure 2: Volatility of Mesa model

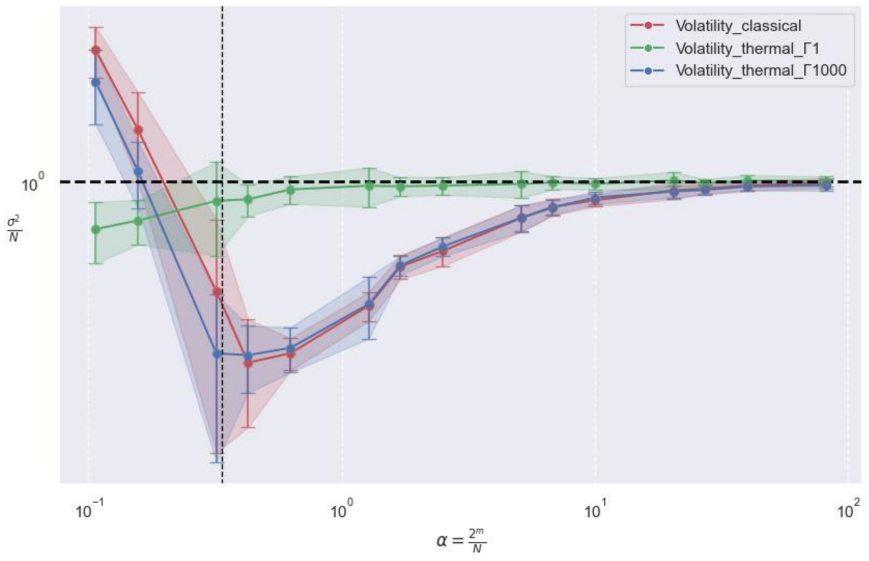
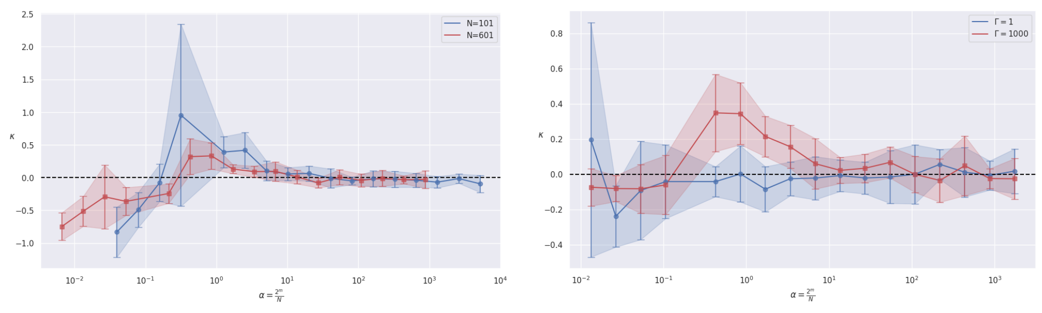


Figure 3: Volatility of osBrain model

The results obtained in the osBrain framework show all data being compatible with the zero value, consequently, they do not exhibit the characteristic behaviour associated with fat tails and real markets, but they are compatible with a gaussian distribution. This is backed by literature [1, 4, 7] and it underlines a problem with the model foundations and not with our realization of it.

However, the results obtained from the Mesa simulations exhibit unusual behaviour. Not all values consistently approach zero, instead, both negative



(a) Classic MG (b) Thermal MG

Figure 4: Kurtosis of Mesa MG

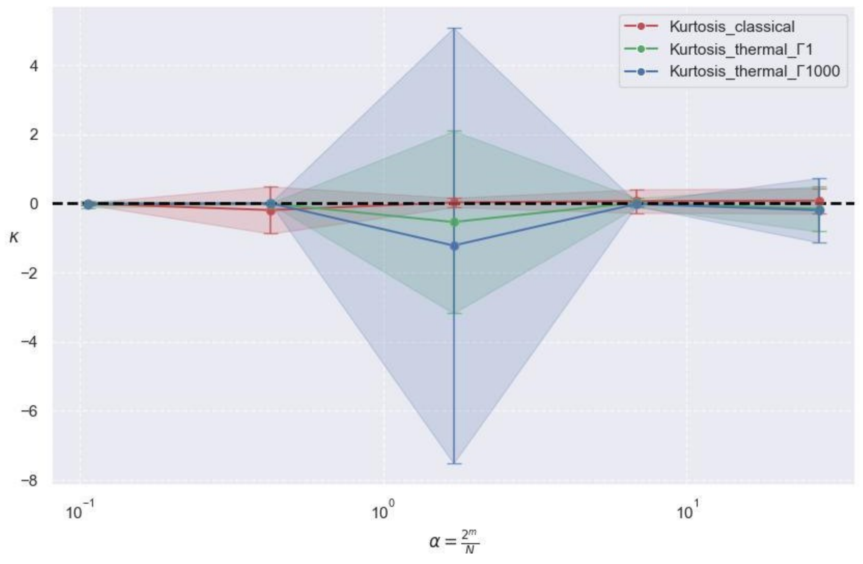


Figure 5: Kurtosis of osBrain model

and positive values emerge, failing to demonstrate any discernible pattern. This behaviour is unlikely to be attributable to the emergence of a genuine market dynamic for specific values of the control parameter *alpha*. Instead, it is more likely that the simulations were terminated prematurely, preventing the system from reaching its steady state.

5 Conclusions

With the mentioned exceptions, the obtained results are consistent with the expectations. While the Classic and Thermal Minority Games have proven invaluable for gaining insights into collective adaptive behaviour, statistical mechanics tools, and non-equilibrium dynamics, their applicability to accurate market dynamics modelling remains questionable. Traditional MG models assume that agents always strive to align with the minority, just by following "carved in stone" rules (like the strategies illustrated), adapting only by inductive reasoning principle, which is unrealistic given the diverse strategies and opportunistic behaviour observed in actual markets.
To address these limitations, a modified MG model, known as the Dollar-game, had been introduced by [7], and others. The Dollar game introduces a novel payoff function

$$\Delta U_{s,i} = a^{\mu(t)}_{s_i(t),i} A(t+1), \tag{9}$$

that encourages agents to shift between minority and majority decisions, allowing for opportunistic behaviours that are more reflective of real-world market dynamics. This modification results in significantly different emergent properties compared to the original MG (e.g. price dynamics). An additional extension involves the introduction of fundamentalist agents (i.e. long position investors), other than speculator agents (already present), allowing population heterogeneity, for the sake of even more realism of market modelling and leading to incredibly precise results in terms of coherence with real price data, as seen in [1].

6 Future Developments

Given that neither the MG nor the DG allow for direct communication between agents, it would be interesting to study a variant of the DG in which agents can communicate, cooperate, and share strategies and decisions, analogously to what happens on social trading platforms, where less experienced agents can copy the trades of more experienced and reliable agents, and understand whether this model can describe the role that herd behaviour can play in the emergence of anomalous price behaviours, such as price bubbles.

References

[1] Frederic D. R. Bonnet and Derek Abbott. Can a minority game follow real market dynamics? *Fluctuation and Noise Letters*, 09(01):107–128, 2010.

[2] Andrea Cavagna, Juan Garrahan, Irene Giardina, and David Sherrington. Thermal model for adaptive competition in a market. *Physical Review Letters*, 83, 03 1999.

[3] Damien Challet and Matteo Marsili. Phase transition and symmetry breaking in the minority game. *Phys. Rev. E*, 60:R6271–R6274, Dec 1999.

[4] Thomas Lux. The socio-economic dynamics of speculative markets: interacting agents, chaos, and the fat tails of return distributions. *Journal of Economic Behavior & Organization*, 33(2):143–165, 1998.

[5] Esteban Moro. The minority game: an introductory guide, 2004.

[6] Robert Savit, Radu Manuca, and Rick Riolo. Adaptive competition, market efficiency, and phase transitions. *Phys. Rev. Lett.*, 82:2203–2206, Mar 1999.

[7] J. Vitting Andersen and D. Sornette. The $-game. *The European Physical Journal B - Condensed Matter*, 31(1):141–145, January 2003.

[8] C. H. Yeung and Y. C. Zhang. Minority games, 2008.

A Tables osBrain

Time-step	Trigger	Timer actions
Market Opening	Internal timer	Sends ”Market Opening” message to the market agent
Investments Checkup	Internal timer	Sends ”Investments Checkup” message to the market agent
Market Closure	Internal timer	Sends ”Market Closure” message to the market agent

Table 1: Timer agent actions

Time-step	Trigger	Market actions
Market Opening	”Market Opening” message received from timer agent	Past history and current day-count published for all investor agents to see
Investments Checkup	”Investments Checkup” message received from timer agent	Sends again the previous message to all investor agents who haven’t yet replied to the opening publishing
Market Closure	”Market Closure” message received from timer agent	The activity value gets computed from all the registered investments, then history is updated and the market is reset for the next day

Table 2: Market agent actions

Time-step	**Trigger**	**Market actions**
Market Opening	First market's message received	Updates its day-count, updates its scores, chooses an action and reports it to the market
Investments Checkup	First market's message received	If its day-count differs from the one in the message: updates its day-count, updates its scores, chooses an action and reports it to the market. If its day-count matches the one in the message it reports again its action to the market.
Market Closure	None	None

Table 3: Investor agents actions