Systemic Complexity of a Growth Management Game: Comparative Analysis of Decision Heuristics and Experimental Results [1]

Onur Özgün and Yaman Barlas

Boğaziçi University

Department of Industrial Engineering, 34342 Bebek, Istanbul, Turkey

Tel: +90 212 3597343, Fax: +90 212 2651800

E-mail: onur.ozgun@boun.edu.tr, ybarlas@boun.edu.tr

Abstract

In this study, using different versions of a growth management game involving two different complexity factors, we compare performances of heuristic rules with experimental results. We present a method for obtaining a statistical distribution of scores resulting from a given simulated decision heuristic, which can be used to compare against and assess experimental gaming results. The method is based on the idea of generating vast number of scores by stochastically simulating a given decision rule and obtaining the resulting score distribution. We use this method to compare scores from different game versions whose scores are essentially not comparable, and to see how the score distributions change from one game version to another. In simulations, we first use a simple random "decision rule" and then develop a more intelligent hill-climbing heuristic. The results show that when the games involve delay, human subjects do not perform better than the random heuristic —a primitive rule composed of a sequence of random decisions. On the other hand, in nonlinear games, subjects outperform the random heuristic and their scores fit better the score distribution of the hill-climbing heuristic. We also demonstrate how the score distribution from random heuristic can be used as a reference performance measure.

Keywords: growth management, systemic complexity, decision heuristics, simulation games, gaming experiments

1 Introduction

Simulation games are widely used as a research tool in the field of dynamic decision-making because they provide a simple controlled environment for testing hypotheses on human psychology and behavior. Tested hypotheses are often on the effect of a factor on the specified independent variable. These factors can be related to (1) the underlying model such as delay or strength of feedback, (2) the simulator characteristics such as decision interval or transparency, or (3) the player characteristics such as mental model or cognitive style (Rouwette et al., 2004). In most of the studies, the independent variable is the task performance; often measured in terms of total cost (Trees et al., 1996; Diehl and Sterman, 1995), cumulative profit (Größler et al., 2000; Yang, 1996), market growth (Bakken, 1993) or deviation from some benchmark (Paich and Sterman, 1993; Moxnes and Saysel, 2009).

[1]Research supported by Boğaziçi University Research Grant no 09HA301D

With the purpose of assessing the influences of systemic complexity factors on the overall complexity of a simulation game, we designed a growth management game and its different versions involving *nonlinearity*, *delay* and *feedback*. The objective of the experiment was to determine a strength level for each factor where the factor becomes effective on game complexity. One of the measures of game complexity was subject' performances, measured by the amount of growth they achieved. During the analysis, we realized that it is very difficult to find an objective measure of growth that works for all game versions. Although the performance measure we used (*normalized cumulative profit*) was an adequate measure for comparing scores coming from different *levels* of a factor (which was our objective), we wanted to develop a method for comparing scores coming from games involving different *factors*.

To illustrate the difficulty of comparing scores from different game versions, suppose that the possible scores in a game are between 0 and 100. Also suppose that introducing *delay* to the game changes this possible range to 0–500. In that case, the analyst should rescale the scores from these two versions to make them comparable. The rescaling problem may be relatively easy if the analyst can determine some constants in the score distribution, like the minimum and the maximum possible scores. Unfortunately, it is often very difficult, if not impossible, to determine such constant references. In rescaling the scores, instead of constant references, it is possible to use the some reference scores such as do-nothing strategy or a standard heuristic. However, applying a fixed strategy does to guarantee a fixed score in the score distribution. For example, the do-nothing strategy might yield the minimum score in one version whereas it might correspond to an average performance in another version. Considering our earlier example, assume that the scores from the base game are symmetrically distributed in the range of 0–100. A score of 50 would reflect an average performance in this version. Now assume that the scores of the game version involving *delay* are strongly left-skewed, i.e. the relative probability of having low scores is higher than the probability of having high scores. In this case, a score of 250 would correspond to a very good performance despite being located in the mid-point of the range.

We tackle the problem of rescaling the game scores by making use of the statistical distribution of possible scores. If we know the distribution of possible scores obtainable from the game, we can tell the probability of acquiring a particular score by chance. In this way, we can assess a particular performance, regardless of the game structure variations.

To obtain the true distribution of all possible scores, in theory, we can enumerate all possible scores, but it is practically impossible for any realistic simulation game. Fortunately, a random sample of all possible scenarios can be used to obtain a reasonably good estimation of the score distribution. Selecting a random sample from all the possible scenarios is essentially generating a random decision in each decision period of the game and running the game with these decisions. We call this decision rule the *random decision heuristic*.

It is also to possible to utilize other decision heuristics instead of the random decision heuristic. In this case, the resulting distribution would be the distribution of scores obtainable by applying this particular decision heuristic. An important step of testing effectiveness of decision rules is comparing experimental results with the decision rule (Sterman, 1987). While comparing the *behavior* of the rule with the behavior of real subjects is a necessary part of this test (Barlas and Özevin, 2004; Diehl and Sterman, 1995; Paich and Sterman, 1993), one can further assess the effectiveness of the decision rule by comparing the score distribution coming from a decision heuristic with experimental results (see Kampmann, 1992, for an example).

In this paper, we describe the growth management game and its different versions involving complexity factors. We illustrate how we obtain the random score distributions for different game versions. Next, we compare score distributions for different game versions with each other and with the players' performances. Finally, we design a *hill-climbing decision heuristic* and compare its performance with random heuristic and players' performances.

2 A Growth Management Game

We designed a growth management game for testing the effects of complexity factors on the overall complexity of the game. The base version of the growth management game does not have any dynamic complexity element, and *nonlinearity*, *delay* and *feedback* (naturally with a stock) is introduced one by one to the game. The factor levels are changed in the experiments by changing the delay order, delay duration, shape of nonlinear functions and gain of feedback loop in the simulator. We selected several levels for each complexity factor: eight for *delay time* and *feedback strength*; four for *delay order* and *nonlinearity*. A total of 20 subjects is used. There are three player groups composed of eight players: delay group, nonlinearity group and feedback group. The players play the simplest base in the beginning and at the end. In between, they play eight games involving one of the complexity factors in different levels. We use a modified version of Latin square design so that the effect of playing order is minimized.

The details of the experiments and analysis of the results are out of the scope of this paper, they can be found in Özgün and Barlas (2011). In this paper, we focus on the effects of complexity factors on the score distribution, and comparison of experimental results with decision heuristics. Due to space limitations, we exclude the *feedback* factor, however, the method described here is applicable to any factor.

2.1 Base Version of the Growth Management Game

The players play the role of a product manager for a certain brand of lotion of a cosmetics company. The time unit of the model is *weeks* and the time horizon is 40 weeks. The calculation step, dt, is taken to be one week. Players can give two types of decisions: price p, and advertising minutes aired on radio per week a. Sales S, is directly proportional to advertising and inversely proportional to price. The sales amount determines the revenue R, and the revenue determines the weekly profit Π. The players' aim is to increase their cumulative profit as much as possible, in a sustainable way.

The game structure is as shown in Figure 1. The equations of the game are in Appendix A. Since the base game will constitute a reference point, which the results of other games are compared against, it is kept as simple as possible. The base game does not include any stock. Thus, essentially it is not a dynamic game, but a simple trial-and-error task of figuring out the price–advertising combination yielding a high profit.

The game is initially at equilibrium. The players are not allowed to change the initial conditions in the first week. Starting from the second week, players give two decisions: price and advertising. They can see the behavior of their weekly profit and the benchmark behavior. The benchmark behavior is defined such that long-run profit is maximized. For finding the benchmark, the weekly profits corresponding to all price–advertising combinations are

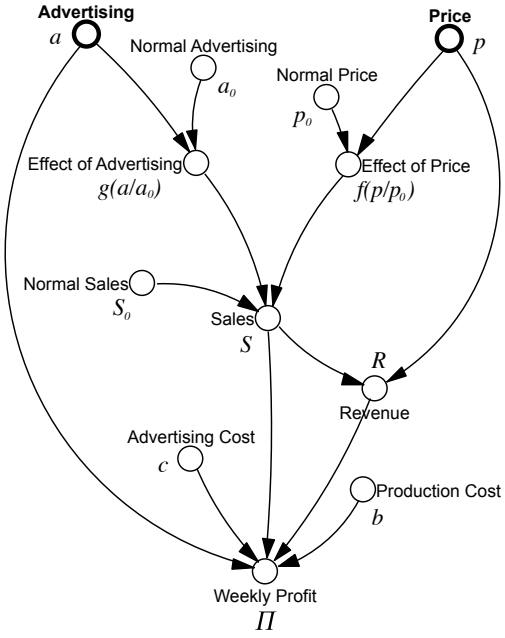


Figure 1: The base structure of the Growth Management Game.

calculated and the combination yielding the maximum weekly profit is applied starting from the second week. In the base game, it is impossible to exceed the weekly benchmark profit, hence, the cumulative benchmark profit. Therefore, the benchmark provides a strict upper-limit.

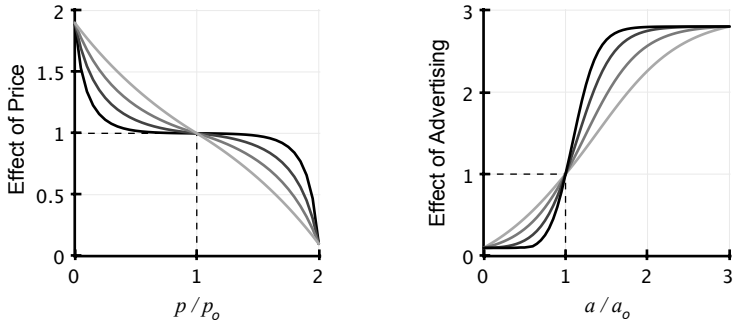
2.2 Nonlinearity Factor

When nonlinearity is added to the base model, effects of price and advertising on sales are made nonlinear, simultaneously, as shown in Figures 2(a) and 2(b). There are four levels of *nonlinearity*: mild (denoted by N1), moderate (N2), high (N3) and extreme (N4). Note that similar to the base game, the nonlinear games do not include any kind of stock. Thus, the players give 40 independent decisions when playing these game versions.

Introducing nonlinearity changes the size and location of the region where the player can obtain high profits (See Figure 3). Similar to the base game, the benchmark behavior for the nonlinear game is the optimum behavior, which is found by the same method applied in the base game.

2.3 Delay Factor

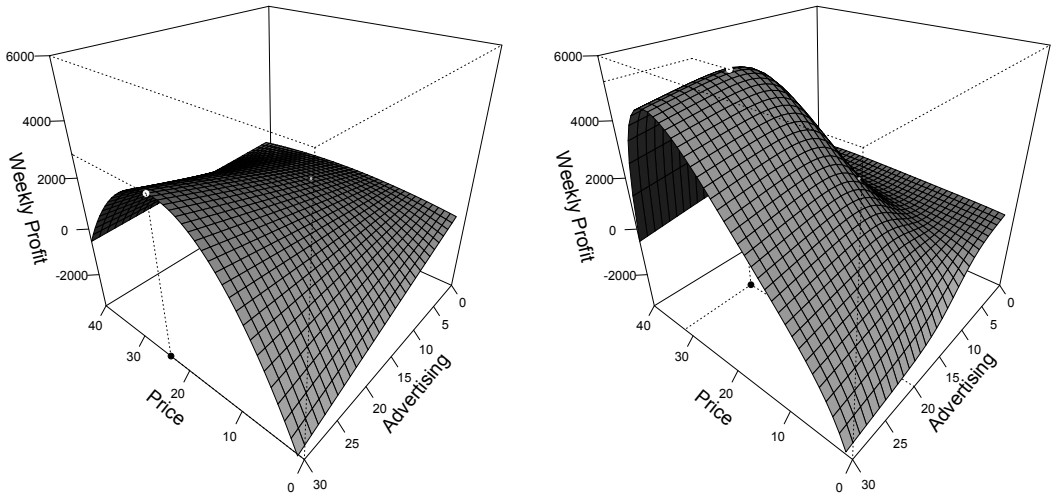
Delay is introduced in the form of information delays between decisions and their effects on sales. Since delay structures include stocks, this version of the game has a dynamic component, unlike the base game and the nonlinear games. The immediate effect of an action on weekly profit is positive if price is increased or advertising is decreased. For example, consider the case of a price increase: since sales will be negatively affected form this price increase after a delay, the player will enjoy a temporary weekly profit rise until



(a) $f(\cdot)$: Effect of price

(b) $g(\cdot)$: Effect of advertising

Figure 2: Levels of nonlinear effect functions. From light to dark: mild, moderate, high and extreme *nonlinearity* versions.



(a) Base game

(b) High *Nonlinearity* setting (N3)

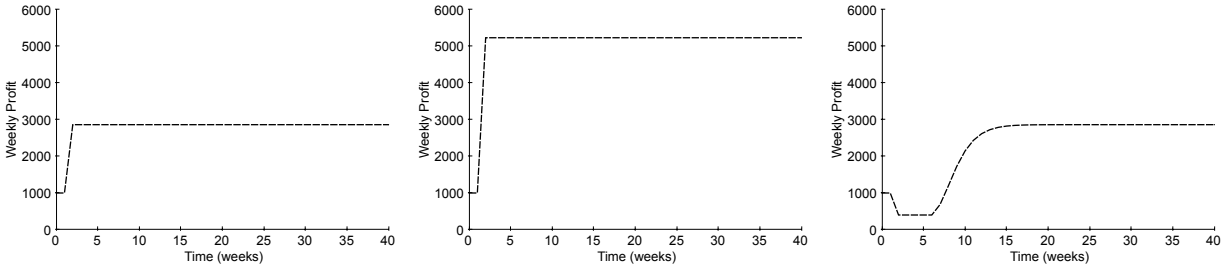
Figure 3: The surface plots showing *weekly profits* resulting from all combinations of *price* and *advertising*.

sales starts to fall. If no other subsequent action is taken, weekly profit will eventually come to the level that it would come if there were not any delay. *Delay* is analyzed in two components: *order of delay* and *delay duration*. *Order of delay* has four levels while *delay duration* has eight levels. There are $4 \times 8 = 32$ possible combinations of all levels of these two variables. The game versions involving *delay* are given in Table 1.

Table 1: The versions of the game involving *delay*.

	Delay Duration							
Delay Order	2 wk	4 wk	6 wk	7 wk	8 wk	9wk	10 wk	11 wk
First Order	O1T2	O1T4	O1T6	O1T7	O1T8	O1T9	O1T10	O1T11
Third Order		O3T4	O3T6	O3T7	O3T8	O3T9	O3T10	O3T11
Fifth Order			O5T6	O5T7	O5T8	O5T9	O5T10	O5T11
Discrete	ODT2	ODT4	ODT6	ODT7	ODT8	ODT9	ODT10	ODT11

The long-term equilibrium point of the profit is not affected by the existence of delay. Thus, the benchmark behavior is found by setting the price and advertising to a combination that maximizes the profit in the long-term, starting from the second week. In the existence of delay, weekly profit first decreases and then gradually reaches a higher level. Figure 4 shows the benchmark behaviors for the base game and games involving nonlinearity and delay.



(a) Base game (b) *Nonlinear* game (setting N3) (c) *Delay* game (setting O5T8)

Figure 4: Benchmark behaviors for the base, *nonlinear* and *delay* game versions.

The benchmark behaviors of game versions involving *delay* maximize the long-term sustainable profit. The *cumulative profit* of a player within the limited time horizon of a *delay* game can be potentially higher than *cumulative benchmark profit*. Therefore, unlike the base game and the nonlinear games, the benchmark behavior of a *delay* game is not the optimum behavior, although it is found by applying the same strategy. Due to the dynamic component of the game involving *delay*, it is not possible to calculate the optimum behavior that yields maximum possible cumulative profit.

3 Obtaining Random Heuristic Score Distributions

For obtaining a distribution of the scores for the random decision heuristic, we generate a sequence random *price* and *advertising* decisions for 40 weeks. Then, we simulate the game with these random decisions and record the resulting *cumulative profit*. This cumulative profit represents the performance score of a simulated player and constitutes one data point. We repeat these steps many times and obtain vast number of scores. The histogram of these scores shows the distribution of possible scores.

In the growth management game, there are 40 periods and roughly $33 \times 28 = 924$ possible decisions at each period. This makes 40^{924} possible decision sequences, each of which yielding a different *cumulative profit*. From this pool of all possible sequences of decisions, we took a sample size of 10^6. Generating the random decisions and running the model for 10^6 simulated players took about two hours on a powerful computer.

Remember that in the base game and the nonlinear games, the results of consecutive decisions are independent (because these games do not have a stock). Since *cumulative profit* is the sum of 40 independent *weekly profits*, the statistical distribution of *cumulative profits* should be normal (Gaussian) by the Central Limit Theorem.

Figure 5 shows the resulting distribution of *cumulative profits* for the base game, obtained by running the random heuristic 10^6 times. As expected, the distribution fits the normal

distribution with a mean of 40,264 and a standard deviation of 5020. The theoretical values of the mean and the standard deviation calculated using the Central Limit Theorem are 40,256 and 5020, respectively. Figure 5 also shows the locations of two fixed strategies: do-nothing strategy and benchmark. The *cumulative profit* of the do-nothing strategy is 40,000. The proximity of do-nothing score and mean of random-heuristic scores is purely coincidental. Benchmark's cumulative profit is 113,634, which is calculated as the sum of weekly profits shown in Figure 4(a). Note that, the benchmark score (which is the maximum possible score in the base game) is far beyond the simulated scores of random heuristic. Indeed, the benchmark is 14.6-σ away from the mean of the distribution!

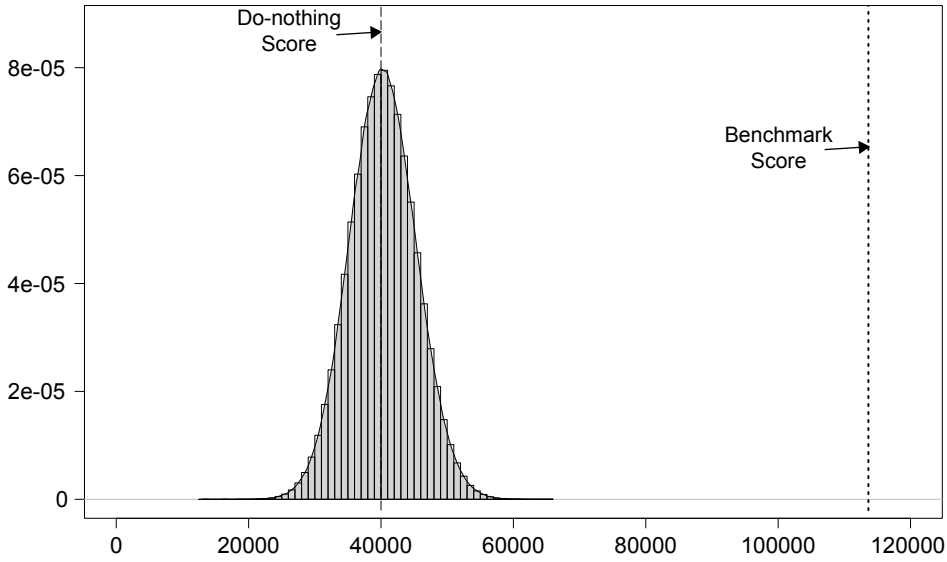


Figure 5: Probability density of random scores (*cumulative profits*) for the base game, estimated by histogram and kernel estimator.

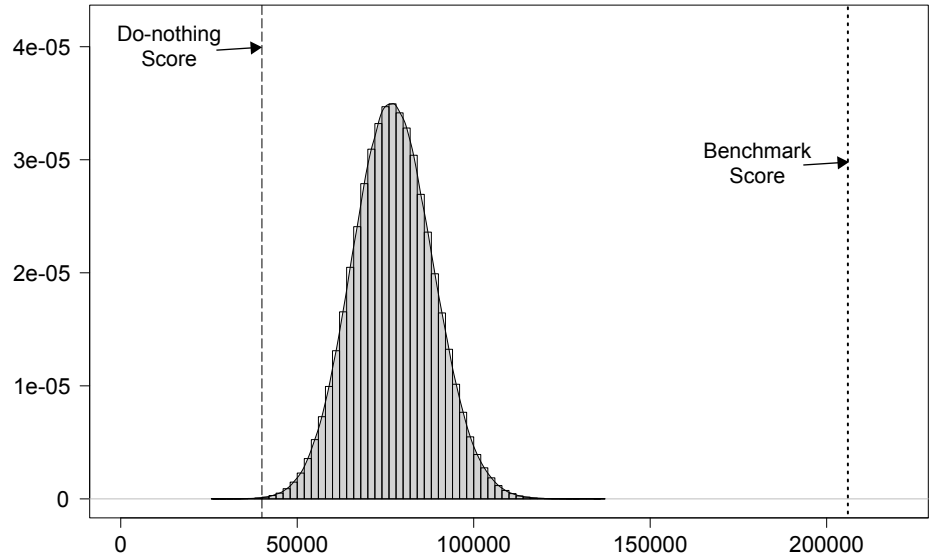


Figure 6: Probability density of random scores (*cumulative profits*) for the game involving high nonlinearity (setting N3).

We repeat the same procedure for the *nonlinear* games. Note that the introduction of nonlinearity increases the maximum possible weekly profit obtainable (see Figure 4(b)). This effect is expected to increase the standard deviation of the cumulative profits. On top of that, introducing *nonlinearity* also changes the shape of the weekly profit surface (see Figure 3(b)). This effect shifts the distribution of the cumulative profits. Figure 6 shows the resulting distribution of the random heuristic scores for the game involving nonlinearity. (Note the scale difference between Figure 5 and Figure 6.) The standard deviation of *nonlinear* games' scores is 2.25 times higher with respect to the base game: from 5020 to 11,319. The mean is shifted to 77,122. Observe that the do-nothing score is now much below the median of the score distribution. Moreover, benchmark score (still the maximum possible score) is now 11.6-σ away from the mean (as compared to 14.6-σ).

Figure 7 shows the cumulative profit distribution for a game version involving *delay* (Setting O5T6: fifth order 6-wk delay). Although the Central Limit Theorem would not apply here, Kolmogorov–Smirnov test for normality yields a very low p-value. The most striking change brought by the introduction of *delay* is the relative lowering of the benchmark cumulative profit. Although the benchmark is calculated by applying same strategy in all games, unlike the base and nonlinear games, the benchmark in the *delay* case is not the maximum possible score. Nonetheless, it is 5.7-σ away from the mean, meaning that it still represents a fairly good strategy.

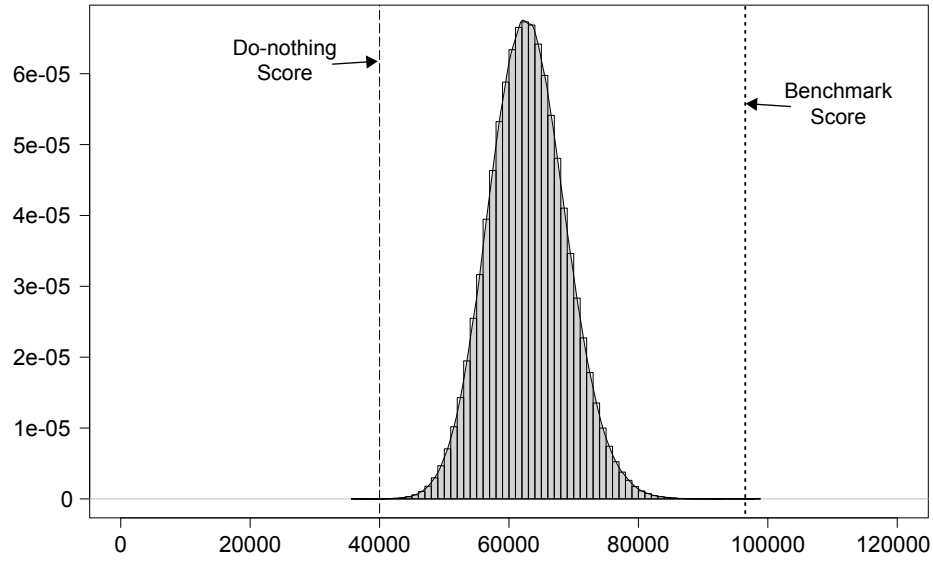


Figure 7: Probability density of random scores (*cumulative profits*) for the game involving 6-wk fifth order delay (setting O5T6).

4 Comparing Random Score Distributions with Experimental Results

First, we compare the random scores for the base game with the players' scores (Figure 8). We have 24 data points for the base game. The players play the base game at the beginning and at the end of the experiment session. In between, they play nine games involving

complexity factors. Overall, the players' scores are much better than the random scores. Most experiment scores are even higher the best random score obtained in the simulations. This is not much surprising because the random heuristic represent a completely random decision process. The intelligent decision-making processes of the players are expected to beat an unintelligent random heuristic.

Figure 9 compares random scores with players' cumulative profits for a game version involving high *nonlinearity*. We have eight experiment data for each nonlinear game version. All 32 of the experimental scores are higher than the maximum of random heuristic scores (See Figure 19 in Appendix B). The densities of nonlinear versions are similar to the base game's density. The results indicate *nonlinearity* does not bring any significant complexity to the game.

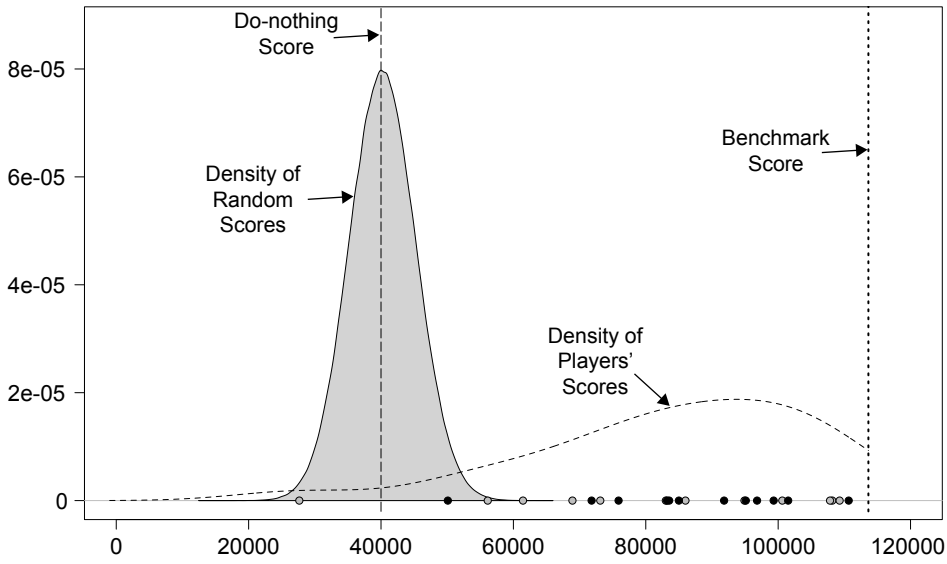


Figure 8: Distribution of the random scores for the base game (grey area) versus players' scores (individual dots) and their density (dashed line). Black dots represent scores from the first trial, the grey dots represent scores from the last trial.

Figure 10 compares random scores with experiment scores for a game involving delay. The figures for other delay games are in Appendix B. Introduction of delay brings a considerable difference. The players' scores are much lower compared to no-delay versions. The distinction between players' scores and simulated scores seems to vanish. This is an important observation because it indicates that *delay* makes is very difficult to carry out a sensible decision making process, giving rise to scores close to the results of a totally unintelligent decision making rule. In the growth management game, delay not only brings a difficulty in controlling the game but it makes difficult to discover a good strategy. Unlike a game in which players control a single variable, in the growth management game players have to find out which strategy would yield a good profit in a two-dimensional price–advertising space. This trial-and-error process takes long time due to delay exists and leads to bad performance scores.

Since now we have the distribution for all game versions, we can compare the scores coming from different game versions with each other. To do this, we define two performance measures.

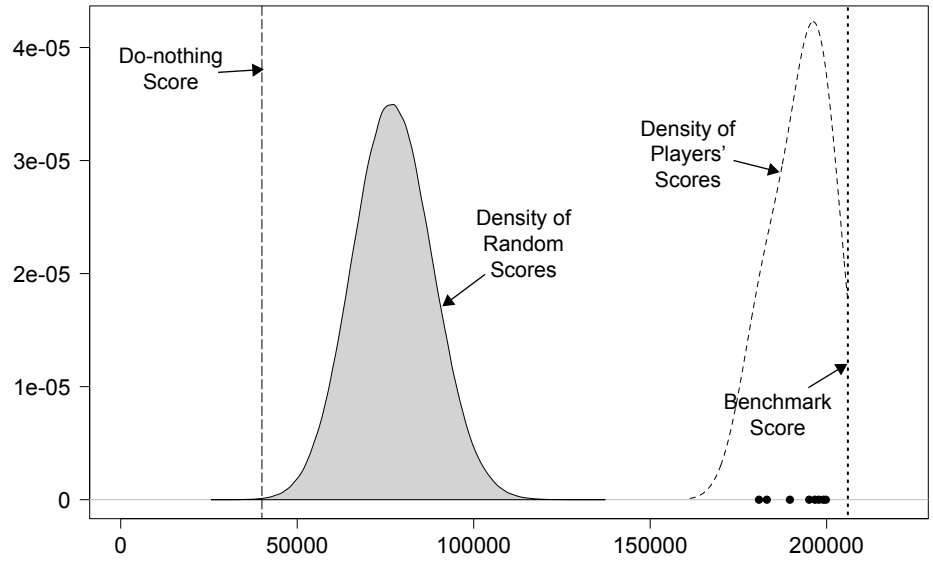


Figure 9: Distribution of the random scores for a nonlinear game – setting N3 (grey area) versus players' scores (individual dots) and their density (dashed line).

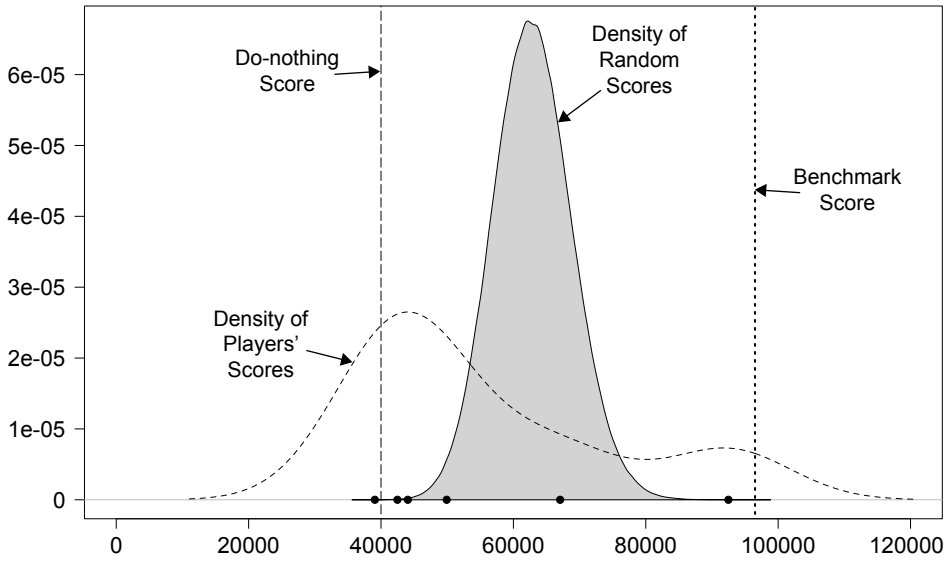


Figure 10: Distribution of the random scores for a delay game – setting O5T6 (grey area) versus players' scores (individual dots) and their density (dashed line).

First performance measure is *standardized cumulative profit*. It standardizes the player scores with respect to means and variances of corresponding game version's random score distributions, as follows:

$$\frac{\textit{Cumulative profit} - \textit{Random heuristic's mean cumulative profit}}{\textit{Standard deviation of random heuristic's cumulative profits}} \tag{1}$$

This performance measure makes use of the fact that the random score distributions follow Normal distribution. It shows how many standard deviations away is a given score from the mean of random scores.

A second performance measure is called *probabilistic score* and is defined as;

$$P\{X \leq x\} \tag{2}$$

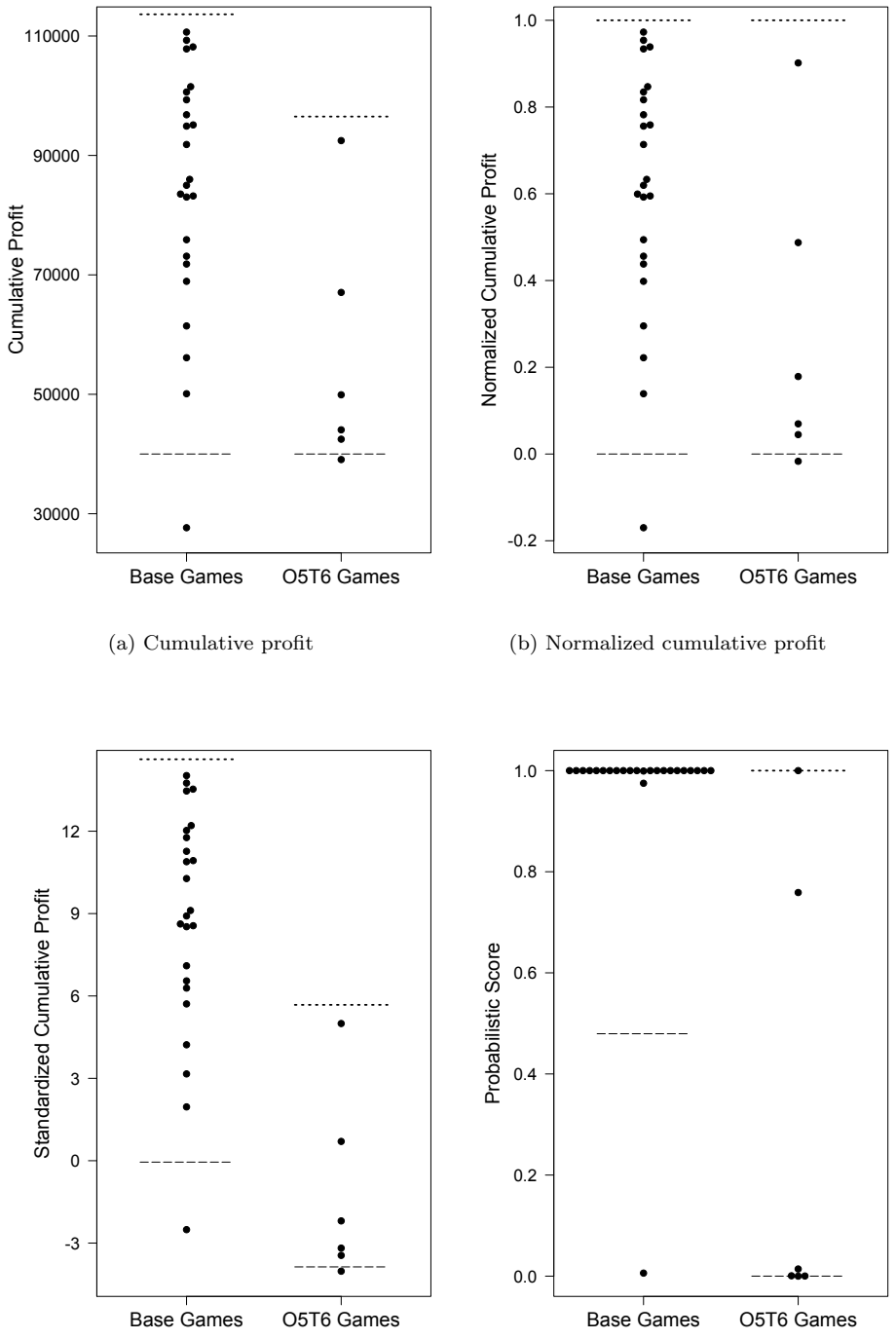
where is x is the score to be evaluated and X is a random variable coming from the density of random scores. This performance measure changes between 0 and 1 and shows the probability of a random score being less than the score x. Note that it does not make any assumption about the distribution of the random scores. In general, we have two options to calculate this probability: either (1) empirically by calculating the frequency of random scores lower than x, or (2) by fitting a known probability distribution to the simulated data and calculating the probability using this distribution. In the growth management game, many experiment scores of the base and nonlinear games are higher than scores coming from random heuristic. Therefore, the first option is not feasible for these game versions. Fortunately, we know that for the base and nonlinear games, the theoretical distribution of cumulative profits is normal. Thus, we can estimate the mean and standard deviation from the sample and use the second option. For the delay version, since real data points fall in the range of random heuristic score distribution, we can use the first option.

Figure 11 compares scores coming experiments of the base game and a delay game using three performance measures. The first performance measure is the usual *cumulative profit* (Figure 11(a)). Since the scales of two games are different, it is not very clear whether a particular data point coming from one version is better than another performance in the other version. The second performance measure is *normalized cumulative profit*. It is defined as:

$$\frac{\textit{Cumulative profit} - \textit{Cumulative profit of do-nothing strategy}}{\textit{Cumulative benchmark profit} - \textit{Cumulative profit of do-nothing strategy}} \tag{3}$$

This performance measure attempts to solve the scale problem by normalizing the scores based on two fixed strategies: do-nothing strategy and benchmark strategy (Figure 11(b)). This is a simple normalization yielding conclusions consistent with the nature of the problem, especially when comparing game versions with similar structures. However, it ignores the possibility that these two reference points may not be at the same difficulty level for different game versions. *Standardized cumulative profit* takes into account the difficulty of obtaining a score by measuring its distance to the mean of random scores in terms of standard deviations. As seen from Figure 11(c), *standardized cumulative profit* reveals the fact that with respect to the random scores, the players performed much better in the base game compared to the delay game. Note that, although *standardized cumulative profit* considers the relative position of player scores within the random score distribution, it ignores the fact that the probabilities drop as we move away from the mean. *Probabilistic score* calculates the probability of obtaining a score by applying the random heuristic. Based on this measure, we see that (Figure 11(d)) most of the scores obtained by players in the base game are almost impossible to obtain by the random heuristic (hence densely located around 1.0) although they seem to be scattered in a wide range in terms of first two performance measures. This measure indicates a stronger difference between the scores of delay games and base games is stronger than indicated by the other measures.

Despite its theoretical appeal, the *probabilistic score* yields results that are difficult to interpret for the growth management game. Data points are mostly at either extremes, which reduces the reliability of the *probabilistic score* as a performance measure for the growth management game.



(a) Cumulative profit

(b) Normalized cumulative profit

(c) Standardized cumulative profit

(d) Probabilistic score

Figure 11: Distribution of scores of for the base game and for a delay game using four performance measures. The upper and lower lines show the benchmark and the do-nothing scores, respectively.

5 Hill-Climbing Heuristics

We saw that the player scores are usually higher than the random scores. In this section, we propose a decision heuristic that is expected to better represent the decision process of

players. We then simulate the heuristic and compare the resulting score distribution with experimental results to assess its performance.

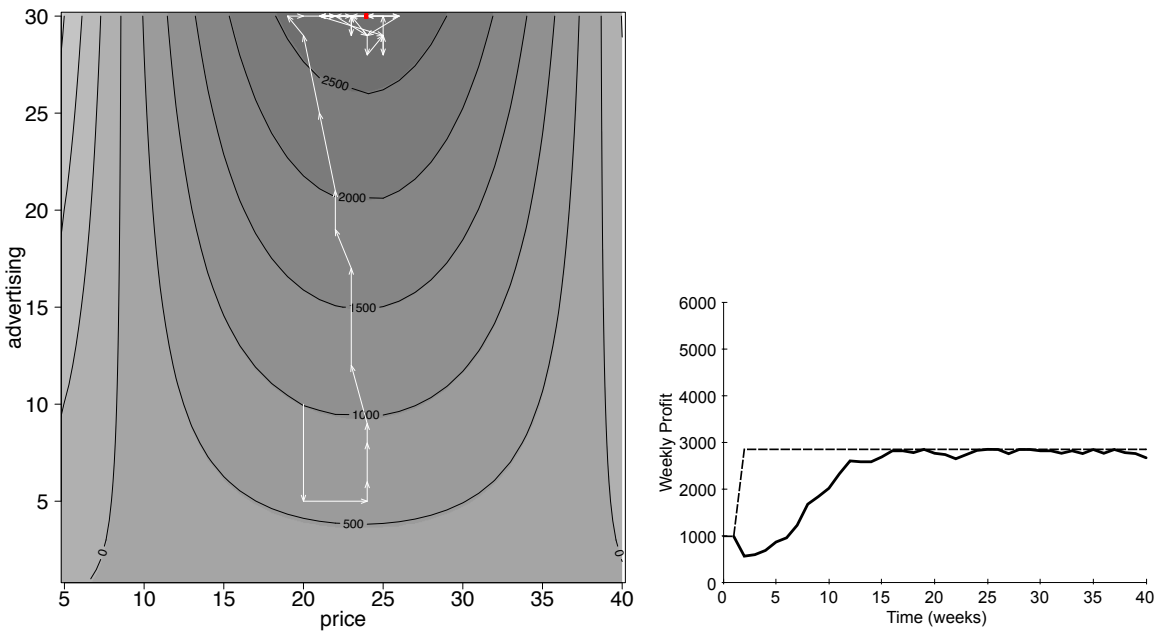
Since this is a growth management task, a hill-climbing heuristic is appropriate. It is widely used as an optimization heuristic (Sterman, 2000). It is based on the idea that if you continue to move in the direction with steepest ascent, you will reach the maximum point, as long as you are not caught in a local optimum. Our algorithm extends the idea of hill-climbing by implementing it on a two-dimensional price–advertising space. Moreover, it only uses the information that is available to a player.

5.1 Hill-Climbing for Base and Nonlinear Game Versions

The hill-climbing algorithm proceeds as follows. First, it takes two initial steps: one for *price*, one for *advertising*, in random order, with arbitrary directions and step sizes. Each of these first two steps involves movement only in one of two variables. In this way, the algorithm determines whether a direction is beneficial and how much profit change it brings. Since there is no delay in these games, the algorithm can immediately assess the outcome of the decisions. Based on the first two steps, the algorithm determines the most promising direction by taking a weighted linear combination of *price* and *advertising* directions with positive profit gains (These are called two base directions: b_1 and b_2). The weights are based on the profit gains per unit change in the variables. Then, it advances in that direction (d^*) with a random step size. To add a further randomness, the movement is not precisely in the best direction but within one-unit neighborhood of the direction. After moving, the algorithm checks whether this direction (d^*) is still beneficial and records the change in the profit. Depending on the direction, either d^* or $-d^*$ becomes one of two base directions (b_2, to be precise), and the newer of former base directions becomes b_1. At each step, the best direction is updated as such, making sure that we always have two linearly independent vectors as the base directions. A detailed pseudocode can be found in Algorithm 1 in Appendix C.

Figure 12 shows a typical output of the hill-climbing algorithm for the base game. In this case, the algorithm first determines a good direction by taking one advertising step downwards and one price step to the right. Then, it quickly approaches the optimum point. The step size is decreased as the slope gets milder. Since the algorithm does not know the location of the benchmark, and the fact that benchmark is optimum, it continues searching for a better profit. Such a pattern is consistent with players' behavior.

Next, we carry out simulations with the hill-climbing algorithm. Figure 13 shows the distribution of 10,000 scores generated by the hill-climbing heuristic for the base game. Based on the *cumulative profits*, it is apparent that the hill-climbing heuristic is more consistent with experiment results than the random heuristic (see Figure 8).



(a) Contour plot

(b) Time-dependent behavior

Figure 12: A typical output of the hill-climbing algorithm for the base game.

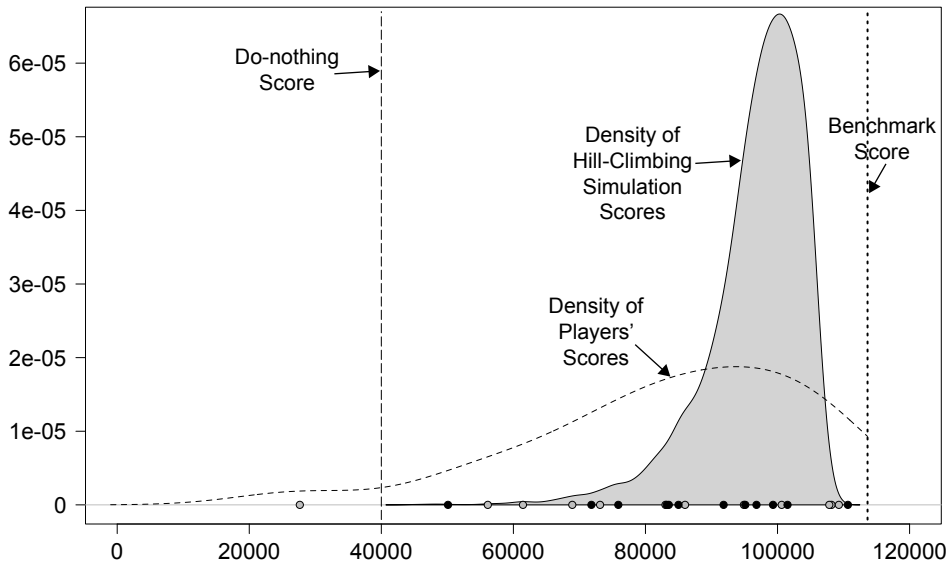
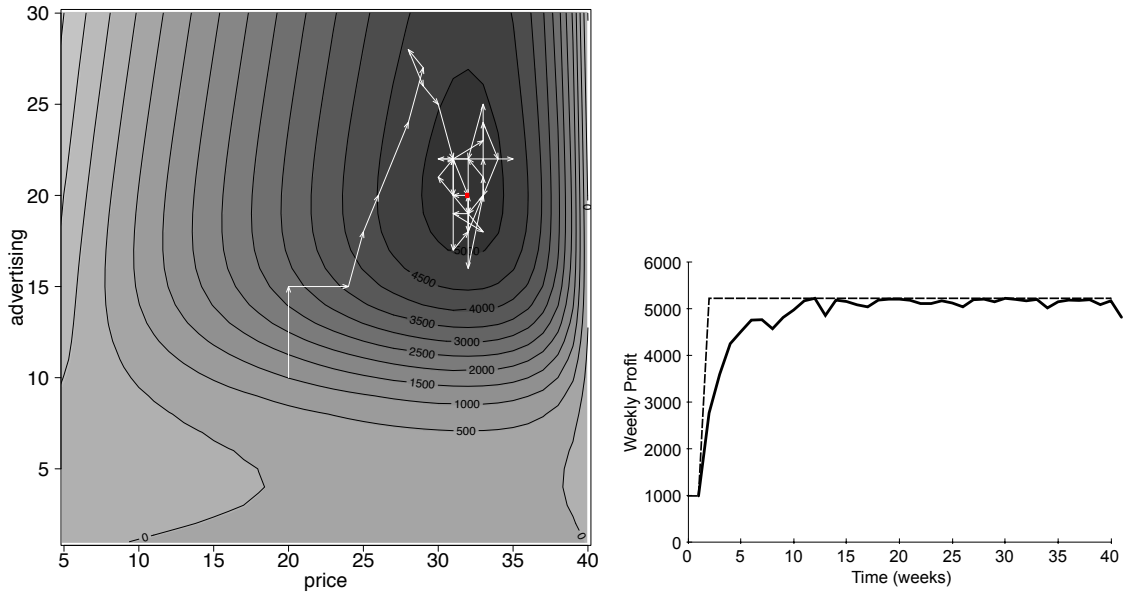


Figure 13: Distribution of the hill-climbing heuristic scores for the base game (grey area) versus players' *cumulative profits* (individual dots) and their density (dashed line).

Figure 14 shows a typical output for a nonlinear version. In the nonlinear versions, the optimum point is different and the profit surface is steeper but the algorithm works in the same fashion. The performance of the algorithm is also similar to the performance in the base game. As seen in Figure 15, the score distribution is close to the benchmark, containing the experiment results. These results indicate that hill-climbing heuristic is consistent with players' performances for the base and nonlinear versions.



(a) Contour plot

(b) Time-dependent behavior

Figure 14: A typical output of the hill-climbing algorithm for a nonlinear game version (N3).

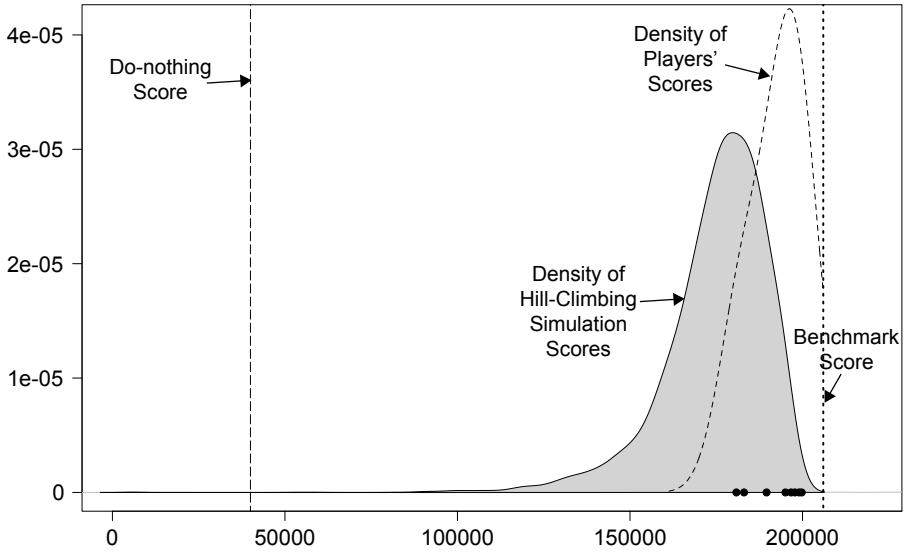


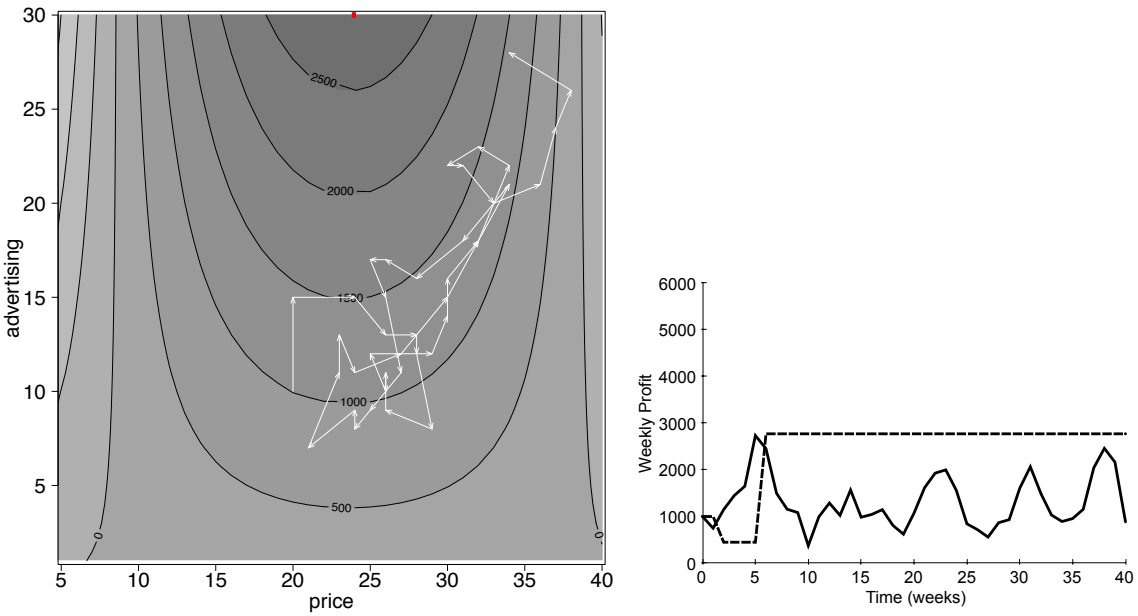
Figure 15: Distribution of the hill-climbing heuristic scores for a nonlinear game – setting N3 (grey area) versus players' *cumulative profits* (individual dots) and their density (dashed line).

5.2 Hill-Climbing for the Delay Game Version

The existence of delay remarkably complicates the growth management game task. First, the results of the actions are delayed. If we want to apply the same principles that we used in the hill-climbing algorithm described above, we have to wait until any action shows its full effect, which is impractical when we have long and high order delays. Second, in the growth management game, each action has two consequences: one immediate and one delayed effect. If the player takes decisions without waiting to see their full effects, the effects of immediate and delayed actions mix. It becomes impossible to understand what portion of a profit change is due to an immediate decision and what portion is due to a delayed decision.

Nevertheless, we adapted the hill-climbing algorithm to accommodate delay. We discriminate between discrete and continuous delay cases. In this paper, we only present the simpler version: discrete-delay case. The algorithm for the delay case makes use of two sets of *base directions*: base directions based on immediate information and delayed information. The base directions based on immediate information are updated like the base directions in the no-delay case. In doing this, we ignore the fact that a profit change is also affected from an earlier decision through delayed effect. In calculating the base direction based on delayed information, the algorithm assumes that T-weeks-earlier decisions are realized in this week. However, in doing this, we ignore the fact the last week's decision is also effective on this week's profit. To compensate for these problems, we use a weighted average of delayed and immediate information to determine the direction of movement. Algorithm 2 in Appendix C presents a pseudocode of the heuristic.

Figure 16 shows a typical behavior for the discrete-delay hill-climbing algorithm when



(a) Contour plot

(b) Time-dependent behavior

Figure 16: A typical output of the hill-climbing algorithm for the discrete delay game version with delay time of 4 weeks and weight of delayed information 0.5.

weights of delayed and immediate information are equal. Note that, the contour plot in the delay case does to show the instantaneous level of profit, but the equilibrium level that will be reached eventually if no further action is taken. For the delayed case, the algorithm is not able to show a clear progress.

Figure 17 shows the distribution of hill-climbing simulation scores versus random heuristic scores. It is clear from the figure that the hill-climbing heuristic is not superior to the simple do-nothing strategy. This conclusion is further verified by experimenting with different delay orders, delay times and weights.

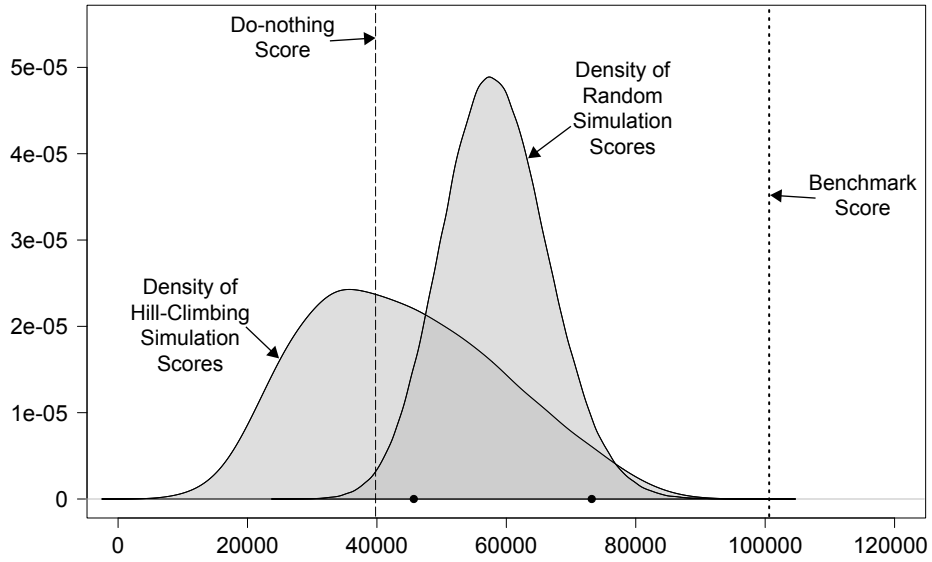


Figure 17: Distributions of hill-climbing scores versus random scores for the discrete delay version (ODT4).

6 Conclusion

In this paper, we use statistical distributions of scores generated by decision heuristics to evaluate the results of gaming experiments with a growth management game involving different complexity factors. We develop a random "decision rule" in which the heuristic takes random decisions in every game step and the score is calculated by feeding these random decisions to the game. We compare the resulting score distribution with real players' scores. We discover that in the experiments of the base game (which does not have any complexity element) and the nonlinear game (which only includes *nonlinearity* between action and its effect), real subjects outperform the random decision heuristic. On the other hand, for the game versions involving *delay*, player results and random heuristic results are close. Indeed, considerable amount of players perform worse than the mean of the random heuristic score distribution. There are even players performing worse than the do-nothing strategy. These results show that for the growth management game, *delay* creates an environment that renders rational decision-making ineffective.

We next propose a hill-climbing decision heuristic for the growth management game. The heuristic addresses the problem of finding steepest ascent in two-dimensional feasible space with limited information. It only uses the information that is available to the player and exhibits a quick progress. The algorithm is stochastic because of the randomness in step

sizes and deliberate minor deviations from the best direction of movement. By running the algorithm numerous times, we obtain a score distribution. The results show that the hill-climbing heuristic is a good representation of players' decision strategies for the base and nonlinear game versions.

The hill-climbing heuristic algorithm is modified to account for delays. However, the nature of delay and some special properties of the growth management game make it very difficult to develop an effective heuristic. The results of the hill-climbing algorithm in this case are not superior to the results of the random heuristics. This outcome is in parallel with our previous observations about the delay game: not only subjects' mental heuristics fall behind random heuristics, but also an "intelligent" heuristic cannot perform well. Despite its poor performance, the hill-climbing heuristic for the delay version can constitute a first step toward a more sophisticated decision rule.

Throughout the paper, we illustrate how the score distributions can be used for different purposes. First, we use the random score distribution as a point of reference to compare different versions of a game, which are difficult to compare due to differences in the game structures. Second, we use score distributions resulting from the hill-climbing heuristic simulations to compare its performance for different versions and under different settings. In addition, it is possible to use the simulation outputs to represent experiment results if the heuristic used is a good-enough representation of the players' decision heuristics. However, such a substitution requires an elaborate experimental study by its own.

References

Bakken, B. E., 1993, *Learning and Transfer of Understanding in Dynamic Decision Environments*, Ph.D. Dissertation, Massachusetts Institute of Technology.

Barlas, Y. and M. G. Özevin, 2004, "Analysis of Stock Management Gaming Experiments and Alternative Ordering Formulations," *Systems Research and Behavioral Science*, Vol. 21, pp. 439–470.

Diehl, E. and J. D. Sterman, 1995, "Effects of Feedback Complexity in Dynamic Decision Making," *Organizational Behavior and Human Decision Processes*, Vol. 62, No. 2, pp. 198–215.

Größler, A., F. H. Maier and P. M. Milling, 2000, "Enhancing Learning Capabilities by Providing Transparency in Business Simulators," *Simulation & Gaming*, Vol. 31, No. 2, pp. 257–278.

Kampmann, C. E., 1992, *Feedback Complexity and Market Adjustment: An Experimental Approach*, Ph.D. Dissertation, Massachusetts Institute of Technology.

Moxnes, E. and A. K. Saysel, 2009, "Misperceptions of global climate change: information policies," *Climatic Change*, Vol. 93, pp. 15–37, 10.1007/s10584-008-9465-2.

Özgün, O. and Y. Barlas, 2011, "Analysis of the Effects of Different Complexity Factors on the Complexity of a Simulation Game," *29th International Conference of the System Dynamics Society*, Seoul, Republic of Korea.

Paich, M. and J. D. Sterman, 1993, "Boom, Bust, and Failures to Learn in Experimental Markets," *Management Science*, Vol. 39, No. 12, pp. 1439–1458.

Rouwette, E. A. J. A., A. Größler and J. A. M. Vennix, 2004, "Exploring Influencing Factors on Rationality: A Literature Review of Dynamic Decision-Making Studies in System Dynamics," *Systems Research and Behavioral Science*, Vol. 21, pp. 351–370.

Sterman, J. D., 1987, "Testing Behavioral Simulation Models by Direct Experiment," *Management Science*, Vol. 33, No. 12, pp. 1572–1592.

Sterman, J. D., 2000, *Business dynamics*, McGraw-Hill New York.

Trees, W. S., J. K. Doyle and M. J. Radzicki, 1996, "Using Cognitive Styles Typology to Explain Differences in Dynamic Decision Making in a Computer Simulation Game Environment," *Proceedings of the 14th International Conference of the System Dynamics Society*, Cambridge, MA, U.S.A.

Yang, J., 1996, "Facilitating Learning through Goal Setting in A Learning Laboratory," *Proceedings of the 1996 International System Dynamics Conference*, Cambridge, MA, U.S.A., The System Dynamics Society.

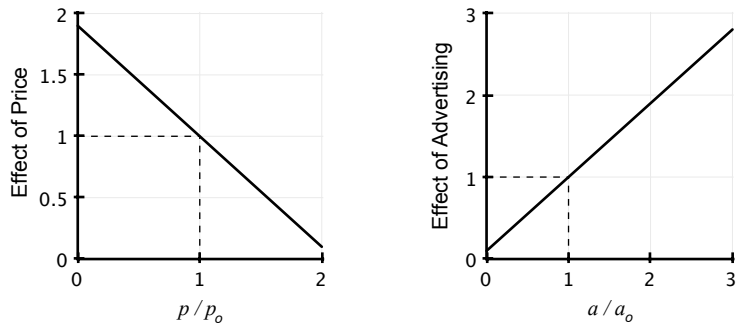
Appendices

A Equations of the Growth Management Game

The sales amount for week t is:

$$S_t = s_0 \; f(p_t) \; g(a_t) \tag{4}$$

where $f(\cdot)$ and $g(\cdot)$ are effect functions of price and advertising, respectively. Their functional forms are shown in Figure 18.



(a) $f(\cdot)$: Effect of price

(b) $g(\cdot)$: Effect of advertising

Figure 18: Linear functions showing the effects of *price* and *advertising* on *sales*.

Weekly revenue and profit are computed as:

$$R_t = p_t \; S_t \tag{5}$$

$$\Pi_t = R_t - b \; S_t - c \; a_t \tag{6}$$

where c is the cost of advertising per minute per week and b is the production cost per item sold.

B Random Score Distributions for Different Game Versions

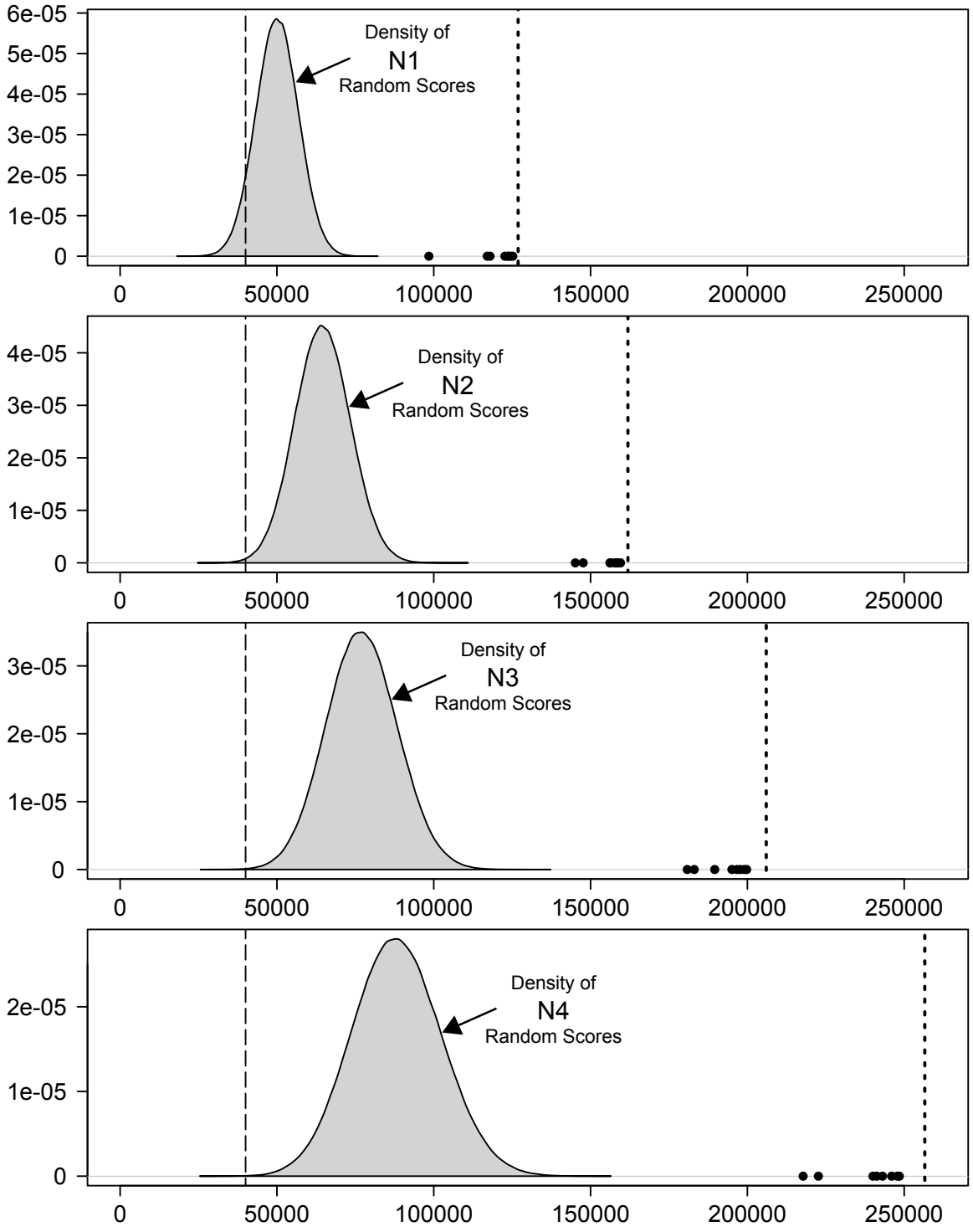


Figure 19: Densities of random scores for nonlinear game versions.

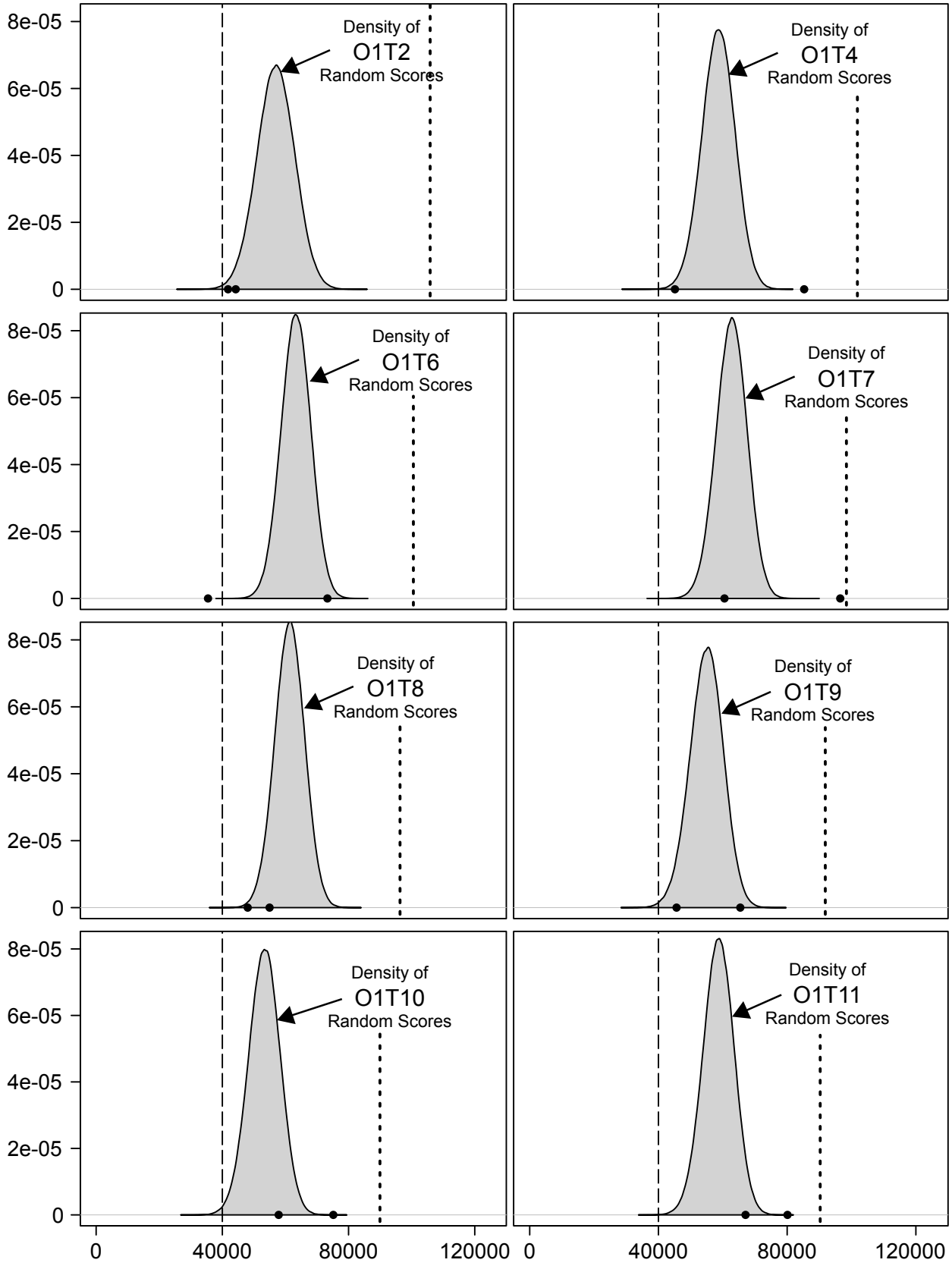


Figure 20: Densities of random scores for game versions involving first order delay.

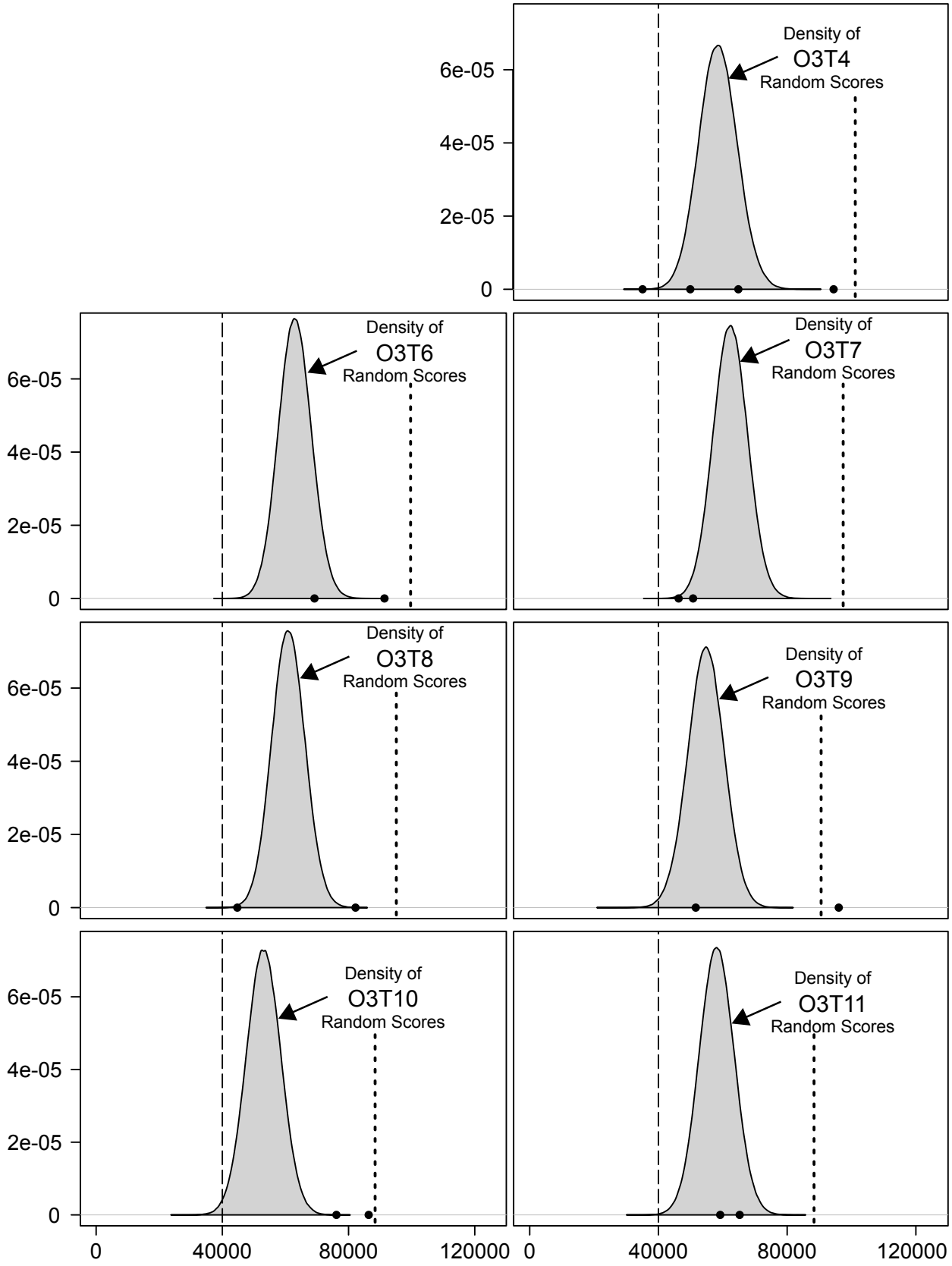


Figure 21: Densities of random scores for game versions involving third order delay.

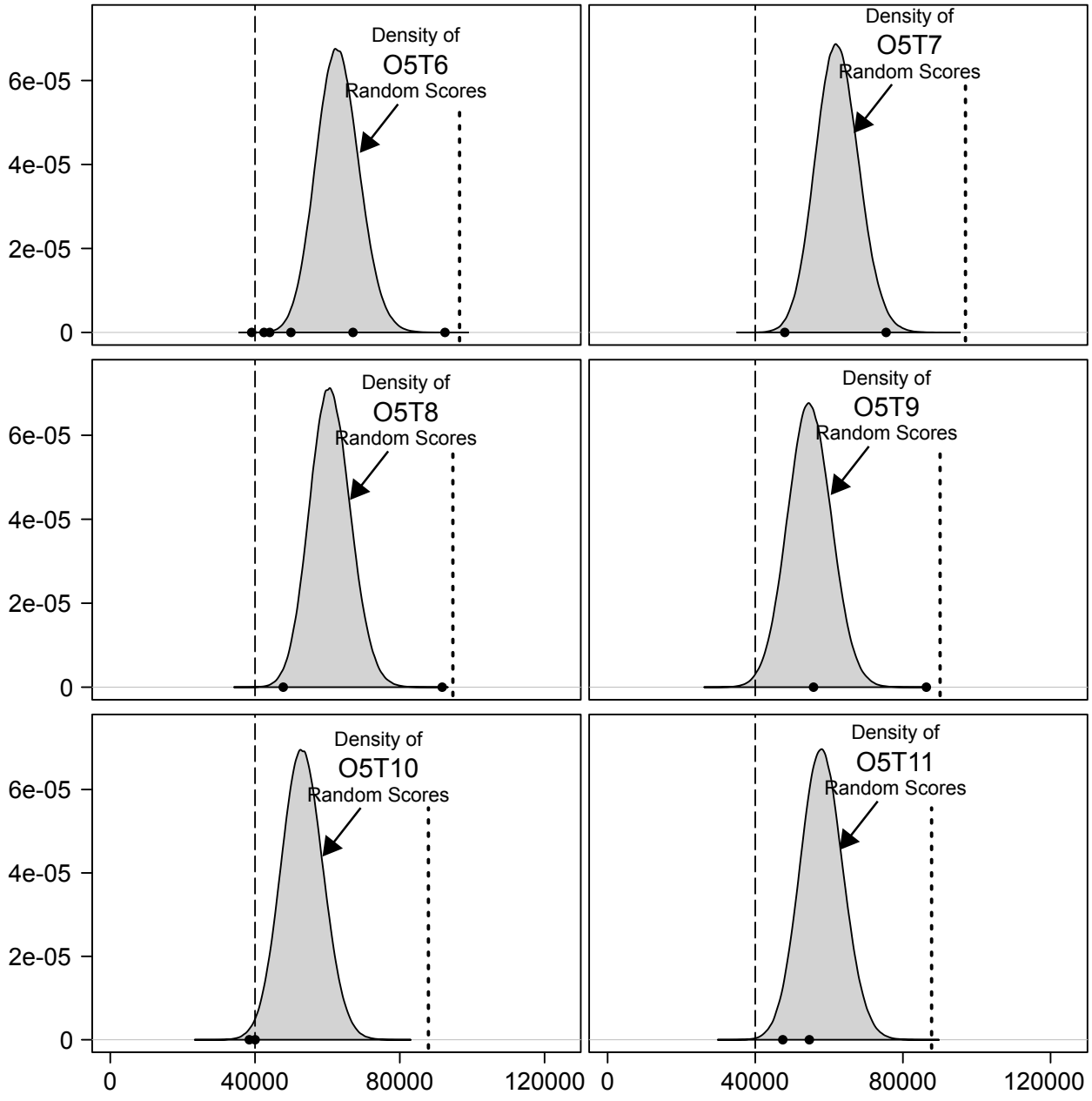


Figure 22: Densities of random scores for game versions involving fifth order delay.

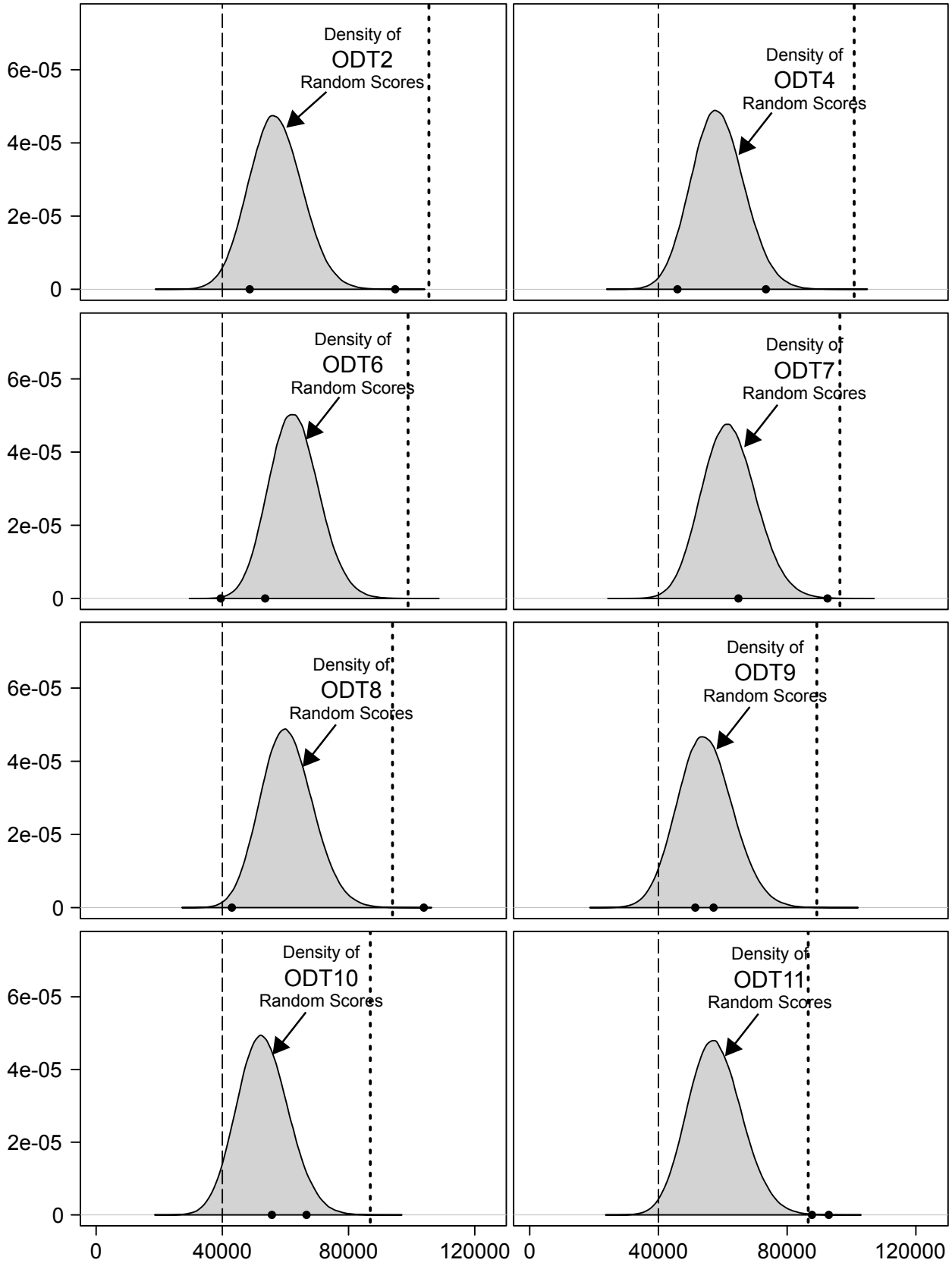


Figure 23: Densities of random scores for game versions involving discrete delay.

C Hill-Climbing Algorithm Pseudocodes

Algorithm 1 hill-climbing heuristic for the base and nonlinear versions

Require: EFFECTPRICE(p_n) ▷ function defining effect of normalized price, p_n on sales
Require: EFFECTADVTS(a_n) ▷ function defining effect of normalized advertising, a_n
Require: UNIFORM(a, b) ▷ a function generating Uniform random variates between a and b
Require: $p_{min}, p_{max}, a_{min}, a_{max}$ ▷ allowed limits for price and advertising in the game

function PROFIT(p, a)
return $100 \cdot (\text{EFFECTADVTS}(a/10) \cdot \text{EFFECTPRICE}(p/20)) \cdot p - 50 \cdot a - 5 \cdot 100 \cdot (\text{EFFECTADVTS}(a/10) \cdot \text{EFFECTPRICE}(p/20))$
end function

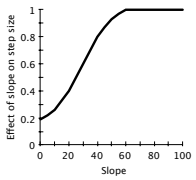
function PICK_DIRECTION() ▷ picks a random direction 1 or -1
if UNIFORM$(0, 1) < 0.5$ **then return** -1
else return 1
end if
end function
function PICK_STEPSIZE_CONT($minss, maxss$) ▷ picks a random real stepsize between $minss$ & $maxss$
return UNIFORM($minss, maxss$)
end function
function PICK_STEPSIZE_DISC($minss, maxss$) ▷ picks a random integer stepsize
return $\lfloor \text{UNIFORM}(minss - 0.5, maxss + 0.5) \rceil$
end function

function FIND_COMBINED_DIRECTION(p_1, a_1, g_1, p_2, a_2, g_2) ▷ determines best direction by taking linear combination of two unit vectors and their gains per unit change
if $\det \begin{vmatrix} p_1 & a_1 \\ p_2 & a_2 \end{vmatrix} \neq 0$ **then**

$$w_1 \leftarrow \frac{g_1}{(g_1 + g_2)},\ w_2 \leftarrow \frac{g_2}{(g_1 + g_2)},\ l_v \leftarrow \sqrt{\left(\begin{pmatrix} p_1 \\ a_1 \end{pmatrix} \cdot w_1 + \begin{pmatrix} p_2 \\ a_2 \end{pmatrix} \cdot w_2\right)^2}$$

return $\dfrac{\begin{pmatrix} p_1 \\ a_1 \end{pmatrix} \cdot w_1 + \begin{pmatrix} p_2 \\ a_2 \end{pmatrix} \cdot w_2}{l_v}$

else
return $\begin{pmatrix} p_2 \\ a_2 \end{pmatrix}$
end if
end function
function EFFECTOFSLOPEONSTEPSIZE(Slope)



end function

Initialize

$\mathbf{b_1} \leftarrow \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, $\mathbf{b_2} \leftarrow \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ ▷ two base directions

$ug_1 \leftarrow 0$, $ug_2 \leftarrow 0$ ▷ base unit gains for each direction

$\mathbf{d}^* \leftarrow \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ ▷ best direction

$t \leftarrow 0$
$a_t \leftarrow 10$, $p_t \leftarrow 20$ ▷ initial conditions of the game
$\Pi_t \leftarrow \text{PROFIT}(p_t, a_t)$

Step 1 ▷ Advance two weeks by taking one price and one advertising decision

$t \leftarrow t + 1$ ▷ $t = 1$

if $\text{UNIFORM}(0, 1) < 0.5$ **then** ▷ randomly pick price or advertising as the starting variable to change
  $picked \leftarrow p$
else
  $picked \leftarrow a$
end if

$p_d \leftarrow \text{PICK\_DIRECTION}()$ ▷ pick a random direction for price
$p_{ss} \leftarrow \text{PICK\_STEPSIZE\_DISC}(2, 5)$ ▷ pick a stepsize between 2 and 5
$a_d \leftarrow \text{PICK\_DIRECTION}()$ ▷ pick a random direction for advertising
$a_{ss} \leftarrow \text{PICK\_STEPSIZE\_DISC}(2, 5)$ ▷ pick a stepsize between 2 and 5

if $picked = p$ **then**
  $p_t \leftarrow \max(p_{min}, \min(p_{max}, p_{t-1} + p_d \cdot p_{ss}))$ ▷ In the first step advance p
  $a_t \leftarrow a_{t-1}$ ▷ Keep a constant
  $p_{t+1} \leftarrow p_t$ ▷ In the second step keep p constant
  $a_{t+1} \leftarrow \max(a_{min}, \min(a_{max}, a_t + a_d \cdot a_{ss}))$ ▷ Advance a
else ▷ If a is picked, do the opposite
  $a_t \leftarrow \max(a_{min}, \min(a_{max}, a_{t-1} + a_d \cdot a_{ss}))$
  $p_t \leftarrow p_{t-1}$
  $a_{t+1} \leftarrow a_t$
  $p_{t+1} \leftarrow \max(p_{min}, \min(p_{max}, p_t + p_d \cdot p_{ss}))$
end if

$\Pi_t \leftarrow \text{PROFIT}(p_t, a_t)$
$\Pi_{t+1} \leftarrow \text{PROFIT}(p_{t+1}, a_{t+1})$

$$\tilde{\mathbf{b}}_\mathbf{1} \leftarrow \begin{pmatrix} (p_t - p_{t-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \\ (a_t - a_{t-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \end{pmatrix}, \quad \mathbf{b_1} \leftarrow \frac{\tilde{\mathbf{b}}_\mathbf{1}}{\| \tilde{\mathbf{b}}_\mathbf{1} \|}$$

$$\tilde{\mathbf{b}}_\mathbf{2} \leftarrow \begin{pmatrix} (p_{t+1} - p_t) \cdot \text{sgn}(\Pi_{t+1} - \Pi_t + \varepsilon) \\ (a_{t+1} - a_t) \cdot \text{sgn}(\Pi_{t+1} - \Pi_t + \varepsilon) \end{pmatrix}, \quad \mathbf{b_2} \leftarrow \frac{\tilde{\mathbf{b}}_\mathbf{2}}{\| \tilde{\mathbf{b}}_\mathbf{2} \|}$$

$$ug_1 \leftarrow \frac{\Pi_t - \Pi_{t-1}}{\| \mathbf{b_1} \|}, \quad ug_2 \leftarrow \frac{\Pi_{t+1} - \Pi_t}{\| \mathbf{b_2} \|}$$

$\mathbf{d}^* \leftarrow \text{FIND\_COMBINED\_DIRECTION}(b_{1,1}, b_{1,2}, ug_1, b_{2,1}, b_{2,2}, ug_2)$ ▷ find the best direction

Step 2

$t \leftarrow t + 2$ ▷ $t = 3$

while t ≤ 40 **do** ▷ Repeat for all remaining weeks
  $ss \leftarrow$ PICK_STEPSIZE_CONT$(1, 4)\cdot$EFFECTOFSLOPEONSTEPSIZE(ug_2) ▷ Pick a step size
  $p_t \leftarrow \min(p_{max}, \max(p_{min}, \lfloor p_{t-1} + d_1^* \cdot ss \rceil + \lfloor$UNIFORM$(-1, 1) \rceil))$ ▷ Calculate next p
  $a_t \leftarrow \min(a_{max}, \max(a_{min}, \lfloor a_{t-1} + d_2^* \cdot ss \rceil + \lfloor$UNIFORM$(-1, 1) \rceil))$ ▷ Calculate next a
  $\Pi_t \leftarrow$ PROFIT(p_t, a_t)
  if $p_t \neq p_{t-1} \vee a_t \neq a_{t-1}$ **then** ▷ Update base directions if moved
    $\mathbf{b_1} \leftarrow \mathbf{b_2}$
    $ug_1 \leftarrow ug_2$
    $\tilde{\mathbf{b}}_\mathbf{2} \leftarrow \begin{pmatrix} (p_t - p_{t-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \\ (a_t - a_{t-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \end{pmatrix}, \quad \mathbf{b_2} \leftarrow \dfrac{\tilde{\mathbf{b}}_\mathbf{2}}{\| \tilde{\mathbf{b}}_\mathbf{2} \|}$
    $ug_2 \leftarrow \dfrac{\Pi_t - \Pi_{t-1}}{\| \mathbf{b_2} \|}$
  end if
  $\mathbf{d}^* \leftarrow$ FIND_COMBINED_DIRECTION$(b_{1,1}, b_{1,2}, ug_1, b_{2,1}, b_{2,2}, ug_2)$
  $t \leftarrow t + 1$
end while

Algorithm 2 Hill-climbing heuristic for the discrete delay version

Require: O_g ▷ order of the delay present in the game
Require: T_g ▷ duration of the delay present in the game
Require: T_h ▷ duration of the delay assumed by the hypothetical player in the heuristic
Require: w_{del} ▷ weight of the delayed information in making decisions
Require: EFFECTPRICE(p_n)
Require: EFFECTADVTS(a_n)
Require: UNIFORM(a, b)
Require: FIND_COMBINED_DIRECTION(p_1, a_1, g_1, p_2, a_2, g_2)
Require: PICK_STEPSIZE_CONT($minss, maxss$)
Require: PICK_STEPSIZE_DISC($minss, maxss$)

function PROFITD(p, a, eop_d, eoa_d) ▷ calculates delayed profit given current price, advertising, delayed effect of price and delayed effect of advertising
  return $100 \cdot (eop_d \cdot eoa_d) \cdot p - 50 \cdot a - 5 \cdot 100 \cdot (eop_d \cdot eoa_d)$
end function

Initialize
$\mathbf{b_1} \leftarrow \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, $\mathbf{b_2} \leftarrow \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ ▷ two base directions

$ug_1 \leftarrow 0$, $ug_2 \leftarrow 0$ ▷ base unit gains for each direction

$\mathbf{d}^* \leftarrow \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ ▷ best direction

$t \leftarrow 0$
$a_t \leftarrow 10$, $p_t \leftarrow 20$ ▷ initial conditions of the game
$\Pi_t \leftarrow$ PROFIT(p_t, a_t)
$eop_d \leftarrow 1$, $eoa_d \leftarrow 1$ ▷ variables keeping delayed effect of price and advertising

Step 1
$t \leftarrow t + 1$
if UNIFORM$(0, 1) < 0.5$ **then** ▷ randomly pick price or advertising as the starting variable to change
  $picked \leftarrow p$
else
  $picked \leftarrow a$
end if
$p_d \leftarrow$ PICK_DIRECTION() ▷ pick a random direction for price
$p_{ss} \leftarrow$ PICK_STEPSIZE_DISC$(2, 5)$ ▷ pick a stepsize between 2 and 5
$a_d \leftarrow$ PICK_DIRECTION() ▷ pick a random direction for advertising
$a_{ss} \leftarrow$ PICK_STEPSIZE_DISC$(2, 5)$ ▷ pick a stepsize between 2 and 5
if $picked = p$ **then**
  $p_t \leftarrow \max(p_{min}, \min(p_{max}, p_{t-1} + p_d \cdot p_{ss}))$
  $a_t \leftarrow a_{t-1}$
  $p_{t+1} \leftarrow p_t$
  $a_{t+1} \leftarrow \max(a_{min}, \min(a_{max}, a_t + a_d \cdot a_{ss}))$
else
  $a_t \leftarrow \max(a_{min}, \min(a_{max}, a_{t-1} + a_d \cdot a_{ss}))$
  $p_t \leftarrow p_{t-1}$
  $a_{t+1} \leftarrow a_t$
  $p_{t+1} \leftarrow \max(p_{min}, \min(p_{max}, p_t + p_d \cdot p_{ss}))$
end if
for $\tau = 2$ to 3 **do** ▷ 4-step routine for updating delay stocks and calculating current profit
  for $\omega = 11$ to 2 **do** ▷ update delay order stocks by using lower order stock's level as input
    $eoadelst_\omega \leftarrow eoadelst_\omega + \dfrac{eoadelst_{\omega-1} - eoadelst_\omega}{T_g/O_g}$
    $eopdelst_\omega \leftarrow eopdelst_\omega + \dfrac{eopdelst_{\omega-1} - eopdelst_\omega}{T_g/O_g}$
  end for

$eoadelst_1 \leftarrow eoadelst_1 + \dfrac{\text{EFFECTADVTS}(a[\tau]/10) - eoadelst_1}{T_g/O_g}$

$eopdelst_1 \leftarrow eopdelst_1 + \dfrac{\text{EFFECTPRICE}(p[\tau]/20) - eopdelst_1}{T_g/O_g}$

▷ update first order stock by using the current functions

$\Pi_\tau \leftarrow \text{PROFITD}(p_\tau, a_\tau, eop_d, eoa_a)$ ▷ calculate the weekly profit by using the last updated value of the delayed effect function for the corresponding delay order

$eoa_d < -eoadelst_{O_g}, \quad eop_d < -eopdelst_{O_g}$ ▷ update the delayed effect functions

end for ▷ end of 4-step routine

▷ initialize two base directions based on immediate information

$$\tilde{\mathbf{b}}_{\mathbf{1}}^{\mathbf{imm}} \leftarrow \begin{pmatrix} (p_t - p_{t-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \\ (a_t - a_{t-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \end{pmatrix}, \quad \mathbf{b}_{\mathbf{1}}^{\mathbf{imm}} \leftarrow \frac{\tilde{\mathbf{b}}_{\mathbf{1}}^{\mathbf{imm}}}{\| \tilde{\mathbf{b}}_{\mathbf{1}}^{\mathbf{imm}} \|}$$

$$\tilde{\mathbf{b}}_{\mathbf{2}}^{\mathbf{imm}} \leftarrow \begin{pmatrix} (p_{t+1} - p_t) \cdot \text{sgn}(\Pi_{t+1} - \Pi_t + \varepsilon) \\ (a_{t+1} - a_t) \cdot \text{sgn}(\Pi_{t+1} - \Pi_t + \varepsilon) \end{pmatrix}, \quad \mathbf{b}_{\mathbf{2}}^{\mathbf{imm}} \leftarrow \frac{\tilde{\mathbf{b}}_{\mathbf{2}}^{\mathbf{imm}}}{\| \tilde{\mathbf{b}}_{\mathbf{2}}^{\mathbf{imm}} \|}$$

$$ug_1^{imm} \leftarrow \frac{\Pi_t - \Pi_{t-1}}{\| \mathbf{b}_{\mathbf{1}}^{\mathbf{imm}} \|}, \quad ug_2^{imm} \leftarrow \frac{\Pi_{t+1} - \Pi_t}{\| \mathbf{b}_{\mathbf{2}}^{\mathbf{imm}} \|}$$

$\mathbf{d}_{\mathbf{imm}}^* \leftarrow \text{FIND\_COMBINED\_DIRECTION}(b_{1,1}^{imm}, b_{1,2}^{imm}, ug_1^{imm}, b_{2,1}^{imm}, b_{2,2}^{imm}, ug_2^{imm})$ ▷ Find the best direction based on immediate information

Step 2

while $t \leq 3$ **do**

if $t > T_h + 1$ **then** ▷ if enough time has passed, initialize directions based on delayed information

if $(p_{t-T_h} \neq p_{t-T_h-1}) \vee (a_{t-T_h} \neq a_{t-T_h-1})$ **then** ▷ if either price or advertising has changed T_h-wk-ago, use new information

if $\| \mathbf{b}_{\mathbf{1}}^{\mathbf{del}} \| = 0$ **then** ▷ if first base direction is still zero, use T_h-wk-old data to initialize it

$$\tilde{\mathbf{b}}_{\mathbf{1}}^{\mathbf{del}} \leftarrow \begin{pmatrix} (p_{t-T_h} - p_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \\ (a_{t-T_h} - a_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \end{pmatrix}, \quad \mathbf{b}_{\mathbf{1}}^{\mathbf{del}} \leftarrow \frac{\tilde{\mathbf{b}}_{\mathbf{1}}^{\mathbf{del}}}{\| \tilde{\mathbf{b}}_{\mathbf{1}}^{\mathbf{del}} \|}$$

$$ug_1^{del} \leftarrow \frac{\Pi_t - \Pi_{t-1}}{\| \mathbf{b}_{\mathbf{1}}^{\mathbf{del}} \|}$$

else if $\| \mathbf{b}_{\mathbf{2}}^{\mathbf{del}} \| = 0$ **then** ▷ if first base direction is nonzero but second base direction is zero, use T_h-wk-old data to initialize second base direction

$$\tilde{\mathbf{b}}_{\mathbf{2}}^{\mathbf{del}} \leftarrow \begin{pmatrix} (p_{t-T_h} - p_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \\ (a_{t-T_h} - a_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \end{pmatrix}, \quad \mathbf{b}_{\mathbf{2}}^{\mathbf{del}} \leftarrow \frac{\tilde{\mathbf{b}}_{\mathbf{2}}^{\mathbf{del}}}{\| \tilde{\mathbf{b}}_{\mathbf{2}}^{\mathbf{del}} \|}$$

$$ug_2^{del} \leftarrow \frac{\Pi_t - \Pi_{t-1}}{\| \mathbf{b}_{\mathbf{1}}^{\mathbf{del}} \|}$$

else ▷ if both base directions are nonzero, use T_h-wk-old data to update the directions

$\mathbf{b}_{\mathbf{1}}^{\mathbf{del}} \leftarrow \mathbf{b}_{\mathbf{2}}^{\mathbf{del}}$

$ug_1^{del} \leftarrow ug_2^{del}$

$$\tilde{\mathbf{b}}_{\mathbf{2}}^{\mathbf{del}} \leftarrow \begin{pmatrix} (p_{t-T_h} - p_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \\ (a_{t-T_h} - a_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \end{pmatrix}, \quad \mathbf{b}_{\mathbf{2}}^{\mathbf{del}} \leftarrow \frac{\tilde{\mathbf{b}}_{\mathbf{2}}^{\mathbf{del}}}{\| \tilde{\mathbf{b}}_{\mathbf{2}}^{\mathbf{del}} \|}$$

$$ug_2^{del} \leftarrow \frac{\Pi_t - \Pi_{t-1}}{\| \mathbf{b}_{\mathbf{1}}^{\mathbf{del}} \|}$$

end if

$\mathbf{d}_{\mathbf{del}}^* \leftarrow \text{FIND\_COMBINED\_DIRECTION}(b_{1,1}^{del}, b_{1,2}^{del}, ug_1^{del}, b_{2,1}^{del}, b_{2,2}^{del}, ug_2^{del})$ ▷ Find the best direction based on delayed information

end if

end if

$t \leftarrow t + 1$

end while

Step 3

while $t \leq 40$ **do**

$\tilde{\mathbf{d}} \leftarrow w_{del}\ \mathbf{d}^*_{\mathbf{del}} + (1 - w_{del})\ \mathbf{d}^*_{\mathbf{imm}}$ ▷ Calculate the direction of movement

$\mathbf{d}^* \leftarrow \dfrac{\tilde{\mathbf{d}}}{\parallel \tilde{\mathbf{d}} \parallel}$ ▷ divide by its length to have a unit vector

$ss \leftarrow$ PICK_STEPSIZE_CONT$(1, 4)$ ▷ Pick a step size

$p_t \leftarrow \min(p_{max}, \max(p_{min}, \lfloor p_{t-1} + d^*_1 \cdot ss \rceil + \lfloor$UNIFORM$(-1, 1) \rceil))$

$a_t \leftarrow \min(a_{max}, \max(a_{min}, \lfloor a_{t-1} + d^*_2 \cdot ss \rceil + \lfloor$UNIFORM$(-1, 1) \rceil))$

▷ 4-step routine for updating delay stocks and calculating current profit

for $\omega = 11$ to 2 **do**

$eoadelst_\omega \leftarrow eoadelst_\omega + \dfrac{eoadelst_{\omega-1} - eoadelst_\omega}{T_g/O_g}$

$eopdelst_\omega \leftarrow eopdelst_\omega + \dfrac{eopdelst_{\omega-1} - eopdelst_\omega}{T_g/O_g}$

end for

$eoadelst_1 \leftarrow eoadelst_1 + \dfrac{\text{EFFECTADVTS}(a[\tau]/10) - eoadelst_1}{T_g/O_g}$

$eopdelst_1 \leftarrow eopdelst_1 + \dfrac{\text{EFFECTPRICE}(p[\tau]/20) - eopdelst_1}{T_g/O_g}$

$\Pi_t \leftarrow$ PROFITD(p_t, a_t, eop_d, eoa_a)

$eoa_d < -eoadelst_{O_g}, \quad eop_d < -eopdelst_{O_g}$ ▷ end of 4-step routine

if $(p_t \neq p_{t-1}) \vee (a_t \neq a_{t-1})$ **then** ▷ Update base immediate directions if moved

$\tilde{\mathbf{b}}^{\mathbf{imm}}_{\mathbf{1}} \leftarrow \tilde{\mathbf{b}}^{\mathbf{imm}}_{\mathbf{2}}$

$ug^{imm}_1 \leftarrow ug^{imm}_2$

$\tilde{\mathbf{b}}^{\mathbf{imm}}_{\mathbf{2}} \leftarrow \begin{pmatrix} (p_t - p_{t-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \\ (a_t - a_{t-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \end{pmatrix}, \quad \mathbf{b}^{\mathbf{imm}}_{\mathbf{2}} \leftarrow \dfrac{\tilde{\mathbf{b}}^{\mathbf{imm}}_{\mathbf{2}}}{\parallel \tilde{\mathbf{b}}^{\mathbf{imm}}_{\mathbf{2}} \parallel}$

$ug^{imm}_2 \leftarrow \dfrac{\Pi_t - \Pi_{t-1}}{\parallel \mathbf{b}^{\mathbf{imm}}_{\mathbf{2}} \parallel}$

end if

$\mathbf{d}^*_{\mathbf{imm}} \leftarrow$ FIND_COMBINED_DIRECTION$(b^{imm}_{1,1}, b^{imm}_{1,2}, ug^{imm}_1, b^{imm}_{2,1}, b^{imm}_{2,2}, ug^{imm}_2)$ ▷ Find the best direction based on immediate information

if $t > T_h + 1$ **then** ▷ if enough time has passed, initialize directions based on delayed information

if $(p_{t-T_h} \neq p_{t-T_h-1}) \vee (a_{t-T_h} \neq a_{t-T_h-1})$ **then** ▷ if either price or advertising has changed T_h-wk-ago, use new information

if $\parallel \mathbf{b}^{\mathbf{del}}_{\mathbf{1}} \parallel = 0$ **then** ▷ if first base direction is still zero, use T_h-wk-old data to initialize it

$\tilde{\mathbf{b}}^{\mathbf{del}}_{\mathbf{1}} \leftarrow \begin{pmatrix} (p_{t-T_h} - p_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \\ (a_{t-T_h} - a_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \end{pmatrix}, \quad \mathbf{b}^{\mathbf{del}}_{\mathbf{1}} \leftarrow \dfrac{\tilde{\mathbf{b}}^{\mathbf{del}}_{\mathbf{1}}}{\parallel \tilde{\mathbf{b}}^{\mathbf{del}}_{\mathbf{1}} \parallel}$

$ug^{del}_1 \leftarrow \dfrac{\Pi_t - \Pi_{t-1}}{\parallel \mathbf{b}^{\mathbf{del}}_{\mathbf{1}} \parallel}$

else if $\parallel \mathbf{b}^{\mathbf{del}}_{\mathbf{2}} \parallel = 0$ **then** ▷ if first base direction is nonzero but second base direction is zero, use T_h-wk-old data to initialize second base direction

$\tilde{\mathbf{b}}^{\mathbf{del}}_{\mathbf{2}} \leftarrow \begin{pmatrix} (p_{t-T_h} - p_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \\ (a_{t-T_h} - a_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \end{pmatrix}, \quad \mathbf{b}^{\mathbf{del}}_{\mathbf{2}} \leftarrow \dfrac{\tilde{\mathbf{b}}^{\mathbf{del}}_{\mathbf{2}}}{\parallel \tilde{\mathbf{b}}^{\mathbf{del}}_{\mathbf{2}} \parallel}$

$ug^{del}_2 \leftarrow \dfrac{\Pi_t - \Pi_{t-1}}{\parallel \mathbf{b}^{\mathbf{del}}_{\mathbf{1}} \parallel}$

else ▷ if both base directions are nonzero, use T_h-wk-old data to update the directions

$\mathbf{b}^{\mathbf{del}}_{\mathbf{1}} \leftarrow \mathbf{b}^{\mathbf{del}}_{\mathbf{2}}$

$ug^{del}_1 \leftarrow ug^{del}_2$

$\tilde{\mathbf{b}}^{\mathbf{del}}_{\mathbf{2}} \leftarrow \begin{pmatrix} (p_{t-T_h} - p_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \\ (a_{t-T_h} - a_{t-T_h-1}) \cdot \text{sgn}(\Pi_t - \Pi_{t-1} + \varepsilon) \end{pmatrix}, \quad \mathbf{b}^{\mathbf{del}}_{\mathbf{2}} \leftarrow \dfrac{\tilde{\mathbf{b}}^{\mathbf{del}}_{\mathbf{2}}}{\parallel \tilde{\mathbf{b}}^{\mathbf{del}}_{\mathbf{2}} \parallel}$

$ug^{del}_2 \leftarrow \dfrac{\Pi_t - \Pi_{t-1}}{\parallel \mathbf{b}^{\mathbf{del}}_{\mathbf{1}} \parallel}$

end if

$\mathbf{d}^*_{\mathbf{del}} \leftarrow$ FIND_COMBINED_DIRECTION$(b^{del}_{1,1}, b^{del}_{1,2}, ug^{del}_1, b^{del}_{2,1}, b^{del}_{2,2}, ug^{del}_2)$ ▷ Find the best direction based on delayed information

end if

end if

$t \leftarrow t + 1$

end while