Table of Contents

Go Back

Policy Diffusion in the Beer Game

Jim Duggan,

Department of Information Technology,

National University of Ireland, Galway,

Galway,

Ireland.

Phone: 353-91-524411 Ext. 3336

Email: jim.duggan@nuigalway.ie

Abstract

This research studies the classic beer game simulation model from a new perspective. It does so by providing each agent with two ordering policies, and creating a set of rules that allow an agent to change its policy. Such a change is triggered based on an agent's confidence in their own performance, and on the relative confidence of their nearest neighbour. The overall effect is that policy diffusion can occur, where, under certain circumstances, an agent will mimic the behaviour of its neighbour, if it believes that its neighbour is performing better. The motivation behind this research is to provide an experimental base upon which the decision making strategies of business agents can be studied.

1. The Beer Game Simulation Model

Sterman (2000), Senge (1990) describe how the beer game is a role-playing simulation originally developed by Jay W. Forrester to introduce students to the concepts of system dynamics and simulation. The game is now widely used in management schools as a means to convey to students the causal relationships between their decision making and the behaviour of supply chains. When the game is played, it typically produces results that are counterintuitive, because large oscillations sweep back through the supply chain based on a small increase in customer demand.

Sterman's (1989) influential and widely-cited[1] study of people's performance and decision making heuristics in the beer game, pinpointed that a key reason for what is termed the Forrester or Bullwhip effect, is that decision makers do not use all the information available to them in order to make a decision. In effect, their rationality is bounded by only taking current inventory levels and expected demand into account. Because of this they ignore the supply line, and, as they proceed through the game, this omission of key information leads to amplification in inventory levels.

Sterman (2000) presents a stock and flow model – based on the generic stock management structure - that mimic decision making in the beer game (see Figure 1).

[1] See http://www.informs.org/manscitop50/list.htm

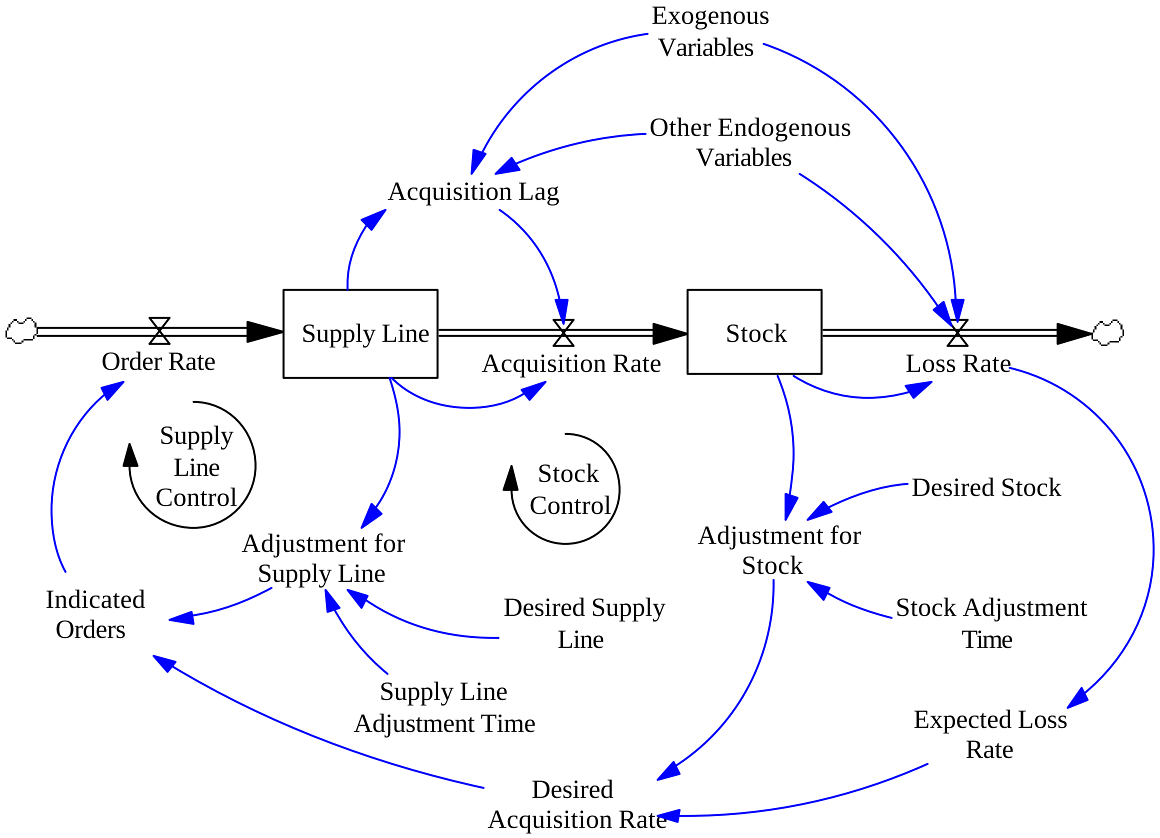


Figure 1 The Generic Stock Management Structure (Sterman 2000)

This model incorporates the stock and flow structure of the supply chain node, along with the decision rule. The total number of orders represent the final decision as to how much stock to order (see equation (a)). The desired acquisition rate (b) is based on calculating a number of values:

- First, the amount of stock that is needed to replenish the expected losses from the next time period. This is the expected loss rate, and this is formulated based on an exponential smoothing of the actual loss rate (i.e. customer demand).
- Second, the amount of stock required to close the gap between the desired stock and the current stock level. This is a goal seeking equation, where the speed of change towards the goal is controlled by the stock adjustment time (see equation (c)).

Indicated Orders = Desired Acquisition Rate + Adjustment for Supply Line	(a)
Desired Acquisition Rate = MAX(0, Expected Loss Rate + Adjustment for Stock)	(b)
Adjustment for Stock = (Desired Stock – Stock)/Stock Adjustment Time	(c)

The third key factor used to calculate the order level is the adjustment for the supply line. In experimental studies of beer game behaviour, Sterman (1989) has found that many decision makers ignore this information cue when making their decisions, and, within the game itself, this is one of the main causal factors in supply chain instability.

Adjustment for Supply Line = (Desired Supply Line – Supply Line) / Supply Line Adjustment Time	(d)
Desired Supply Line = Expected Loss Rate * Acquisition Lag	(e)

Equations (d) and (e) describe how this decision rule is structured. The rationale for this rule is that it models a decision maker with a good memory of what has already been ordered. It is a goal seeking equation, where the target is to always have sufficient goods in the supply line to meet requirements. For example, if our steady state demand is 100 units, and the delay time is 3, then, in steady state our desired supply line would be 300 units. If, however, our actual supply line is above the value, then the adjustment for the supply line will be negative, and hence our overall orders (a) will be proportionately reduced.

If a decision maker ignores the supply line as part of their decision making, equation (a) takes the form of (a′) below, and the use of this decision rule within the beer game gives rise to oscillation in inventory levels, with repeated levels of undershooting and overshooting around the target.

Indicated Orders = Desired Acquisition Rate	(a′)

Why decision makers would be attracted to (a′) rather than (a) is an interesting research question. Sterman argues that it is mainly due to the effect time delays have on our problem solving ability, and in cases where there is a significant time delay between the cause and the effect, our ability to deal with this is limited.

Our research builds upon this beer game model by taking the equations (a) and (a′) as two policy options within the beer game, and proposes a new set of equations that model how a decision maker could switch policy depending on the state of their confidence, and the confidence of the other players in the game.

2. The Policy Diffusion Model

The model augments the classic beer game with an additional policy control stock and flow structure. As the structures are exactly the same for all agents, the sets of equations for the Retailer are only presented, although the full VENSIM model is provided as an attachment. Figure 2 extends the model presented earlier by adding a stock for the queue of orders in the system. The decision rules can be easily configured to either ignore the supply line or they can make use of the supply line variable, which is in this case is the better decision rule.

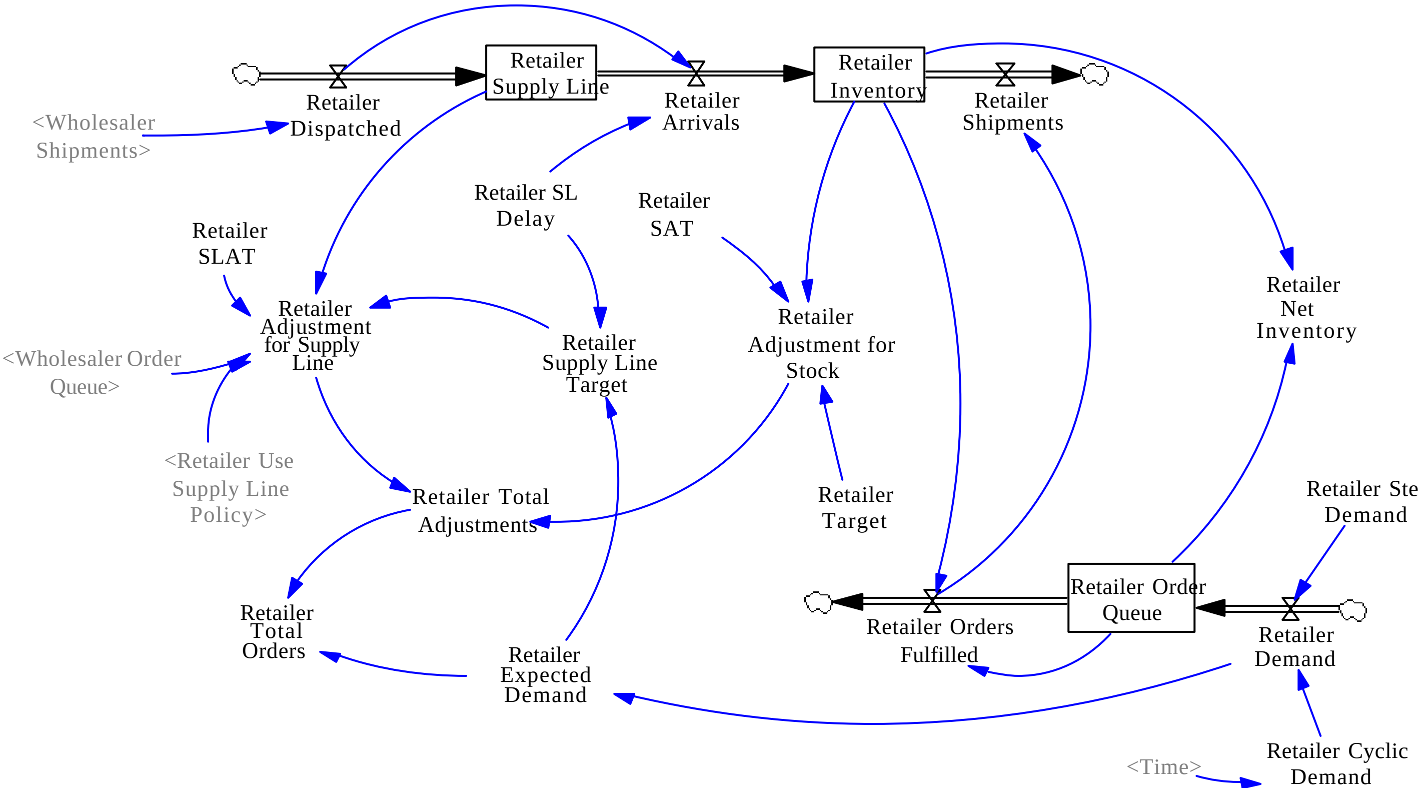


Figure 2 Extension of Sterman's Model to include Backlogs and Policy Switching

From Figure 2, equation (1) generates retailer demand, which is an addition of a cyclic demand pattern (2) with a step demand pattern (3). In equation (1), ones and zeros are uses as multiplier factors to effectively “switch on” one demand pattern, and, in this case, the step demand pattern is active. The step demand inserts the once-off spike that doubles demand and so introduces the required variability into the system. The cyclic demand pattern varies based on a sinwave with a period of twenty five time units, and an amplitude of 100 around a mean of 200.

In this model, because all orders are eventually fulfilled, a queue of orders (4) is maintained, and this queue gets depleted once sufficient inventory is present (5).

Retailer Demand = (Retailer Cyclic Demand * 0) + (Retailer Step Demand * 1)	(1)
Retailer Cyclic Demand = 200+ (100 * SIN (2 * 3.14159 * Time / 25))	(2)
Retailer Step Demand = 100 + step (100, 4)	(3)
Retailer Order Queue = INTEG(Retailer Demand – Retailer Orders Fulfilled , 100)	(4)
Retailer Orders Fulfilled = MIN (Retailer Inventory , Retailer Order Queue)	(5)

The stock of inventory (6) – initially set to the target value - is filled by the arrival of goods (7), and depleted by shipments to the customer (8).

Retailer Inventory = INTEG(Retailer Arrivals - Retailer Shipments , Retailer Target)	(6)
Retailer Arrivals = DELAY FIXED (Retailer Dispatched , Retailer SL Delay , 100)	(7)
Retailer Shipments = Retailer Orders Fulfilled	(8)

The remaining equations model the decision making process of each agent. The total orders (9) are formulated based on the stock management structure replacement heuristic (Sterman 2000), which is based on the total adjustments (10) and on the expected losses over the next time period (11).

Retailer Total Orders = MAX (0, Retailer Expected Demand + Retailer	(9)

Total Adjustments)	
Retailer Total Adjustments = Retailer Adjustment for Stock + Retailer Adjustment for Supply Line	(10)
Retailer Expected Demand = SMOOTHI (Retailer Demand , 3, 100)	(11)

The total adjustments comprise the retailer adjustment for stock (12) and the retailer adjustment for the supply line (13). Both of these equations are goal seeking, as they both seek to close the gap between the desired state of the system, and the current system state. The Retailer Target (14) is set at 400, and the stock adjustment time (15) is fixed at 1.

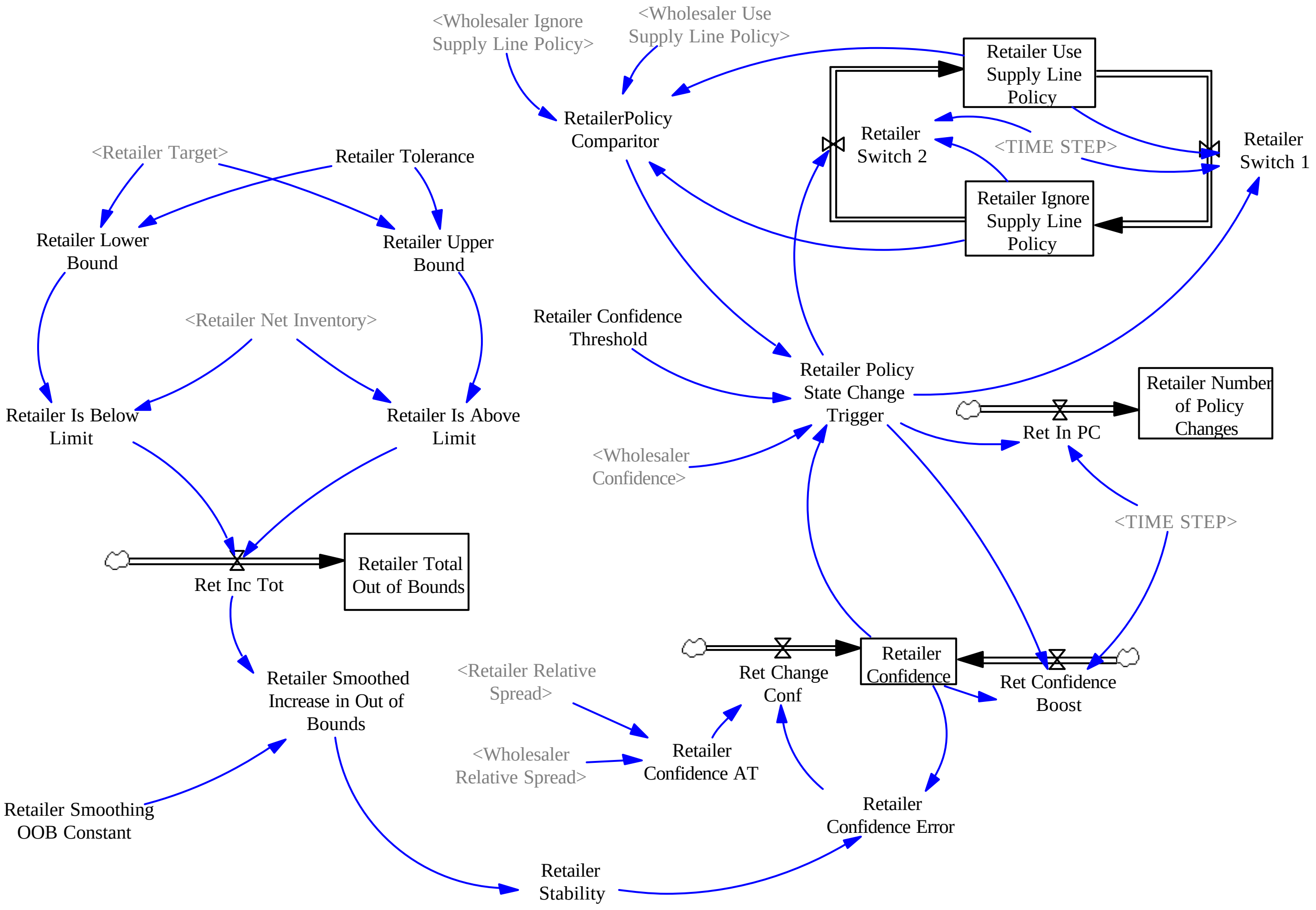
Retailer Adjustment for Stock = (Retailer Target – Retailer Inventory) / Retailer SAT	(12)
Retailer Adjustment for Supply Line = ((Retailer Supply Line Target - (Wholesaler Order Queue + Retailer Supply Line)) / Retailer SLAT) * Retailer Use Supply Line Policy	(13)
Retailer Target = 400	(14)
Retailer SAT = 1	(15)

The Retailer Supply Line Target (16) is the ideal inventory level to have in transit, and is a product of the expected demand (11) and the transportation delay (17), which is made up of a queuing delay of one time unit, and a shipping delay of three time units. If the supply line policy is active, the variable Retailer Use Supply Line Policy (42) has a value of one, otherwise this variable is zero. This allows the model to accommodate the use of either heuristic in the simulation (i.e. the supply line is either taken into account, or it is ignored).

Retailer Supply Line Target = Retailer Expected Demand * Retailer SL Delay	(16)
Retailer SL Delay = 3 + 1	(17)
Retailer Supply Line = INTEG(Retailer Dispatched – Retailer Arrivals, 300)	(18)

Retailer Dispatched = Wholesaler Shipments	(19)
Retailer Net Inventory = Retailer Inventory - Retailer Order Queue	(20)

Finally, the supply line (18) is represented as a stock that is increased by goods dispatched upstream from the wholesaler (19), and decreased, after a pipeline delay, by goods arriving (7). Also, the retailer's net inventory (20) is stored, as this is an important measure that is monitored to influence the policy control equations, which are now described.



<Wholesaler Ignore Supply Line Policy>
<Wholesaler Use Supply Line Policy>
Retailer Use Supply Line Policy
RetailerPolicy Comparitor
Retailer Switch 2
<TIME STEP>
Retailer Switch 1
<Retailer Target>
Retailer Tolerance
Retailer Ignore Supply Line Policy
Retailer Lower Bound
Retailer Upper Bound
<Retailer Net Inventory>
Retailer Confidence Threshold
Retailer Policy State Change Trigger
Retailer Number of Policy Changes
Retailer Is Below Limit
Retailer Is Above Limit
Ret In PC
<Wholesaler Confidence>
<TIME STEP>
Ret Inc Tot
Retailer Total Out of Bounds
Ret Change Conf
Retailer Confidence
Ret Confidence Boost
Retailer Smoothed Increase in Out of Bounds
<Retailer Relative Spread>
<Wholesaler Relative Spread>
Retailer Confidence AT
Retailer Confidence Error
Retailer Smoothing OOB Constant
Retailer Stability

Figure 3 The Stock and Flow Model for Policy Control

The role of the second part of the model is to control the policy being used – this is shown in Figure 3. As with most control strategies, a measurable quantity is needed in order to provide the necessary information to the decision point. For our model, the retailer's confidence is a key factor in deciding whether or not to switch policy, and this confidence is mainly based on the stability of their inventory, which in turn is derived from an out of bounds measure. The out of bounds measures the number of times the stock will exceed tolerable limits, and in cases where significant oscillation occurs, this will be high.

The net inventory value is compared against acceptable lower and upper thresholds. The retailer tolerance (21) is used to specify the upper and lowers bounds of acceptability. In this case, the upper bound (22) is 1.75 times the target, and the lower bound (23) is one quarter of the target. These values are then used to calculate whether, for a particular point in time, the retailer is outside of these limits (equations (24) and (25)). These values are combined (26) so that at any point in the simulation, a record of whether the retailer is out of bounds is kept, and an accumulation of these values is also stored (27).

Retailer Tolerance = 0.75	(21)
Retailer Upper Bound = Retailer Target * (1 + Retailer Tolerance)	(22)
Retailer Lower Bound = Retailer Target * (1 - Retailer Tolerance)	(23)
Retailer Is Above Limit = IF THEN ELSE (Retailer Net Inventory > Retailer Upper Bound, 1, 0)	(24)
Retailer Is Below Limit = IF THEN ELSE (Retailer Net Inventory < Retailer Lower Bound, 1, 0)	(25)
Ret Inc Tot = Retailer Is Above Limit + Retailer Is Below Limit	(26)
Retailer Total Out of Bounds = INTEG(Ret Inc Tot , 0)	(27)

Equation (26) is important, as it captures whether or not the retailer is out of bounds at any point in the simulation. Based on this, we evaluate the exponential moving average of this value (28), so that the most recent value is given highest weighting. This value will be normalised somewhere between [0..1], where a high value indicates that the agent is nearly always out of bounds, whereas a value that is close to zero is an indication that the control system is performing within expectations. The value chosen for the smoothing constant (29) would reflect how reactive the decision maker is to the most recent values.

Retailer Smoothed Increase in Out of Bounds = SMOOTH3I (Ret Inc Tot , Retailer Smoothing OOB Constant , 0)	(28)
Retailer Smoothing OOB Constant = 10	(29)

Equation (28) now has a crucial role to play, because based on this normalised value, we can arrive at a measure for the stability of the retailer's control system. Equation (30) defines this, and transforms the value in (28) to a scale of [0..100]. A high value for (28) translates to a low stability value, and vice-versa (in this case, the relationship is linear with a negative slope).

Retailer Stability = 100 - (Retailer Smoothed Increase in Out of Bounds * 100)	(30)

The retailers confidence (31) is directly related to this stability measure. The mechanism used to model this is *adaptive expectations,* as the confidence continually tracks the stability value in a goal seeking manner (32, 33).

Retailer Confidence = INTEG(Ret Change Conf + Ret Confidence Boost , 100)	(31)
Retailer Confidence Error = Retailer Stability - Retailer Confidence	(32)
Ret Change Conf = Retailer Confidence Error / Retailer Confidence AT	(33)

The smoothing constant (34) has a key role to play. It is based on a constant component and a variable component. The variable component multiplies 5 by the difference of the *wholesaler relative spread* (35) and the *retailer relative spread* (36).

Retailer Confidence AT = 5 + (5 * (Wholesaler Relative Spread – Retailer Relative Spread))	(34)
Retailer Relative Spread = ZIDZ (Retailer Stand Deviation , SumRetailerWholesaler)	(35)
Wholesaler Relative Spread = ZIDZ (Wholesaler Stand Deviation , SumRetailerWholesaler)	(36)

The logic behind equations (35) and (36) is as follows. The relative spread values is the ratio of each agent's inventory standard deviation, divided by the sum of their standard deviations. Let's say, for example, that at a certain point in the simulation, the standard deviation of the retailer's inventory is 200, while the standard deviation of the wholesaler's is 300. In this scenario, the equations have the following values:

(35) Retailer Relative Spread = 200/500 = 0.40
(36) Wholesaler Relative Spread = 300/500 = 0.60
(35) Retailer Confidence AT = 5 + 5*(0.60 – 0.40) = 6

On the other hand, the Wholesaler Confidence AT is formulated as (34a):

Wholesaler Confidence AT = 5 + (5 * (Retailer Relative Spread – Wholesaler Relative Spread))	(34a)

Therefore, under this scenario, equation (34a) evaluates to:

Wholesaler Confidence AT = 5 + 5*(0.40-0.60) = 4

With a lower adjustment time, the Wholesaler Confidence is not as robust as the Retailer Confidence, because the Wholesaler has a higher variability. This means that, under this scenario, the retailer's confidence value will not slide as quickly as the wholesaler, and this has implications for the policy change trigger which we will explore shortly.

If, at some stage, the retailer changes strategy, they receive a confidence boost (37), whereby their confidence levels shoot up. This is based on a variant of the saying that "the faraway hills are always greener", and that when decision makers change a course of action "for the better", initially they are infused with a sense of optimism and confidence. Note, that the confidence boost only occurs if a change in policy has taken place.

Ret Confidence Boost = ((100 - Retailer Confidence) / TIME STEP) * Retailer Policy State Change Trigger	(37)

A policy change (38) will only happen if each of the following conditions are true. First, the retailer's neighbourly agent (wholesaler) must be using a different policy. Second, the retailer's confidence must be less than confidence threshold. Third, the confidence level of your neighbourly agent (wholesaler) is higher than this agent (retailer). Equation (39) is calculated based on the logic described in Table 1, and is only true (i.e. evaluates to 1) when the comparing agents are using different strategies.

Retailer Policy State Change Trigger = IF THEN ELSE (Retailer Confidence < Retailer Confidence Threshold :AND: Retailer Confidence < Wholesaler Confidence, 1 * RetailerPolicyComparitor , 0)	(38)
RetailerPolicyComparitor = IF THEN ELSE (((Retailer Ignore Supply Line Policy * Wholesaler Ignore Supply Line Policy) + (Retailer Use Supply Line Policy * Wholesaler Use Supply Line Policy)) = 1, 0, 1)	(39)

Retailer Use Supply Line Policy	Retailer Ignore Supply Line Policy	Wholesaler Use Supply Line Policy	Wholesaler Ignore Supply Line Policy	Retailer Policy Comparitor
1	0	1	0	0
0	1	1	0	1
1	0	0	1	1
0	1	0	1	0

Table 1 Comparison Logic (Truth Table) for Policy Comparitor

The number of policy changes (40, 41) are also recorded, as the assumption is that the more frequently an agent changes their policy, their confidence surge will suffer, and the effect of the confidence boost will be diminished. In equation (41) this value is divided by the time step (DT), because equation (34) behaves similar to a PULSE, and is only true for one time slice of the simulation.

Retailer Number of Policy Changes = INTEG(Ret In PC , 0)	(40)

Ret In PC = Retailer Policy State Change Trigger / TIME STEP	(41)

Finally, the two policies (42) and (43) are represented as having one of two different states (one or zero), and they flip from one state to the other. If a policy state is zero, it is switched off, and if it is one, the policy is active. In effect, the diagram showing this can be thought of as a simple state machine, controlled by two switches (44) and (45). In this example, the default policy is to use the supply line when making decisions, but this initial state can easily be changed.

Retailer Use Supply Line Policy = INTEG(Retailer Switch 2 - Retailer Switch 1 , 1)	(42)
Retailer Ignore Supply Line Policy = INTEG(Retailer Switch 1 - Retailer Switch 2 , 0)	(43)
Retailer Switch 1 = IF THEN ELSE (Retailer Policy State Change Trigger = 1 :AND: Retailer Use Supply Line Policy = 1, 1 / TIME STEP , 0)	(44)
Retailer Switch 2 = IF THEN ELSE (Retailer Policy State Change Trigger = 1 :AND: Retailer Ignore Supply Line Policy = 1, 1 / TIME STEP , 0)	(45)

The switching (44, 45) will only occur if the policy state change trigger (38) is true, *and* the opposite policy is active. Because this switch occurs over a single time slice and we need to preserve the different stocks at values of one and zero, the increase and decrease amounts are divided by DT.

In summary, equations (1) through (45) extend the classical beer game simulation so that policy diffusion can be experimented with. For our model, those equations were extended to model the remaining actors in the beer game. In the next section, we proceed to conduct a set of experiments to explore the conditions under which policy diffusion can occur.

3. Policy Diffusion Experiments

The model has been constructed so that an agent can only compare itself against one of its neighbours. Because the goal of the experiment is to explore whether the good policy heuristic can diffuse throughout the entire supply chain, two initial states are used. First, if the retailer has the good policy; and all the others do not, then, in theory, the policy can diffuse in the following sequence (R-W-D-F). Second, if the wholesaler has the good policy, then this policy can diffuse in two directions (W-R) and (W-D-F).

Agent	**Can Copy Policy From Agent**
Retailer	Wholesaler
Wholesaler	Retailer
Distributor	Wholesaler
Factory	Distributor

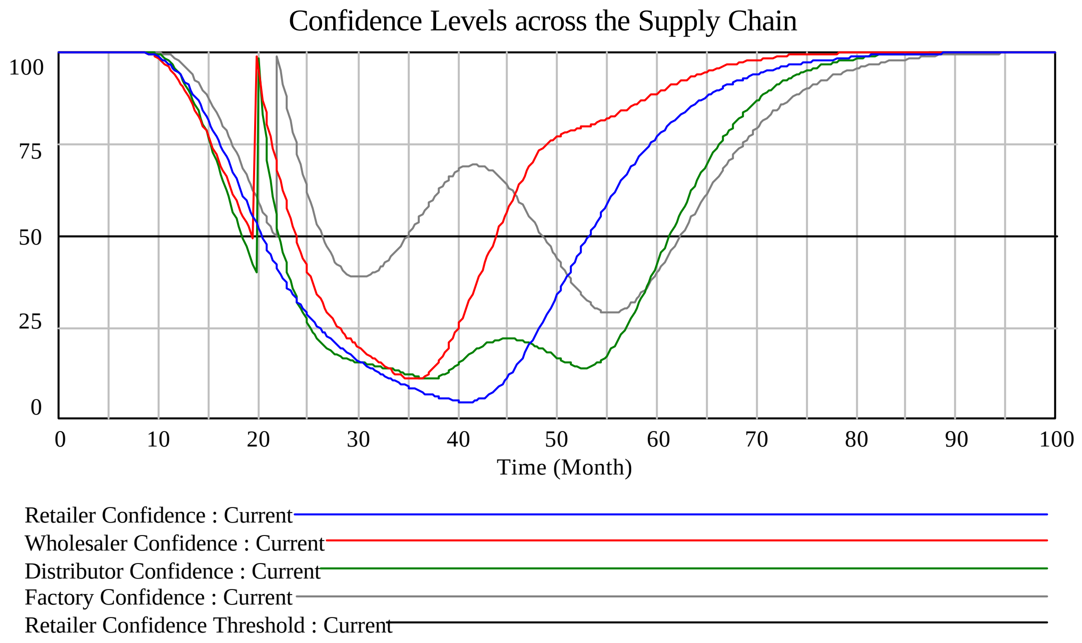
Table 2 Policy Diffusion Structure for Experimentation

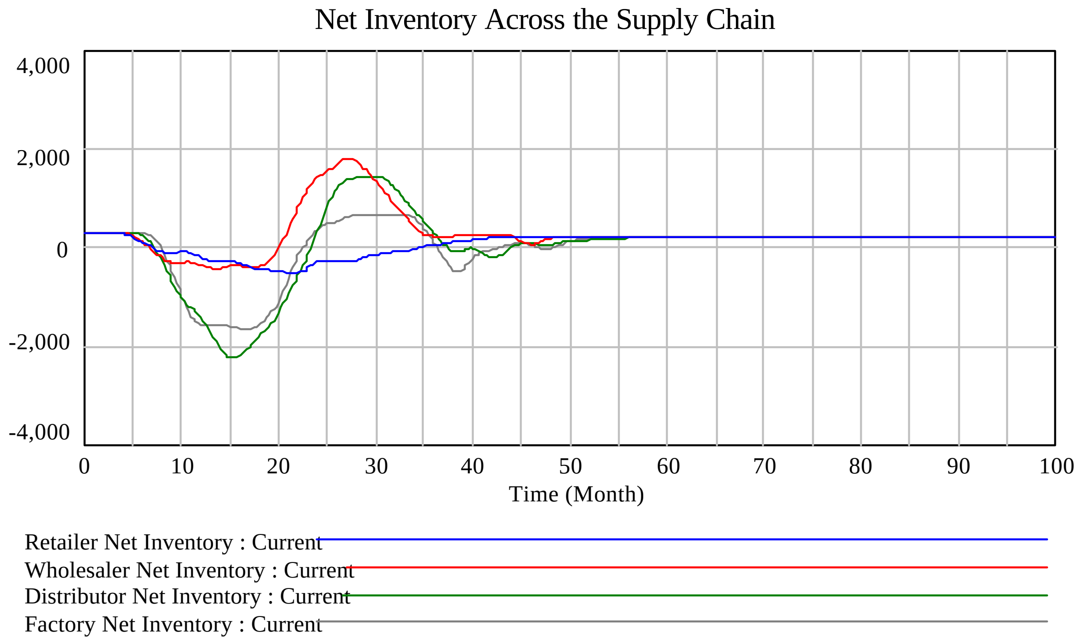
In total, four initial experiments are identified. Two demand patterns are used (for comparison purposes) to drive retailer demand, these are the classic step function (that is used in the beer game), and a cyclic demand pattern.

Experiment Number	**Demand Pattern**	**Retailer SL Policy**	**Wholesaler SL Policy**	**Distributor SL Policy**	**Factory SL Policy**
1	Step	On	Off	Off	Off
2	Cyclic	On	Off	Off	Off
3	Step	Off	On	Off	Off
4	Cyclic	Off	On	Off	Off

Table 3 Initial Conditions for Each Experimental Run

Experiment 1: Retailer SL Policy On, All others Off (Step Demand Pattern)



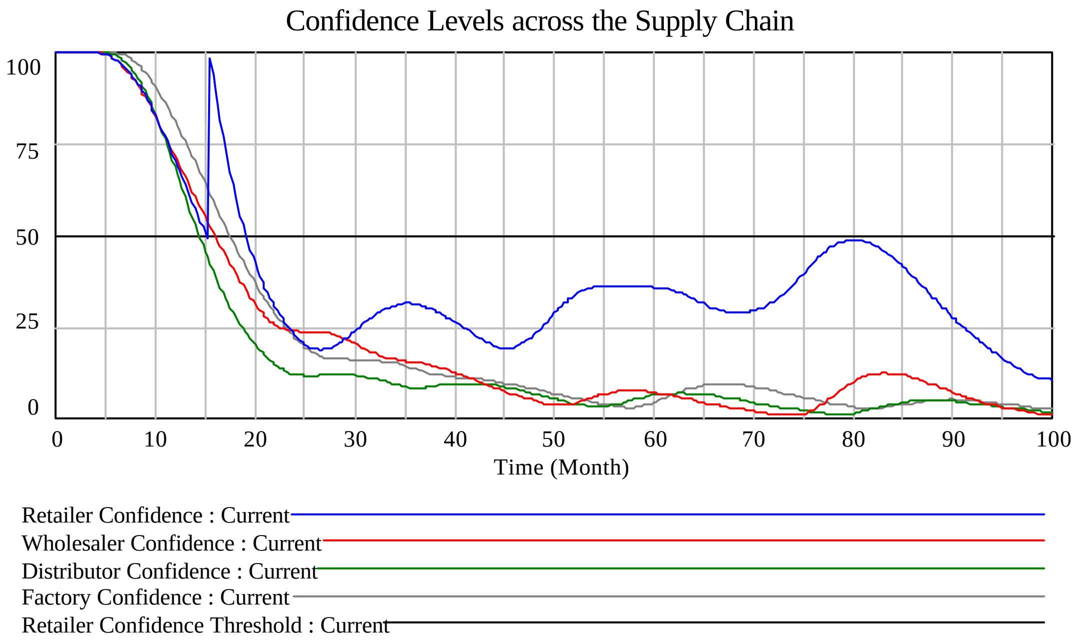


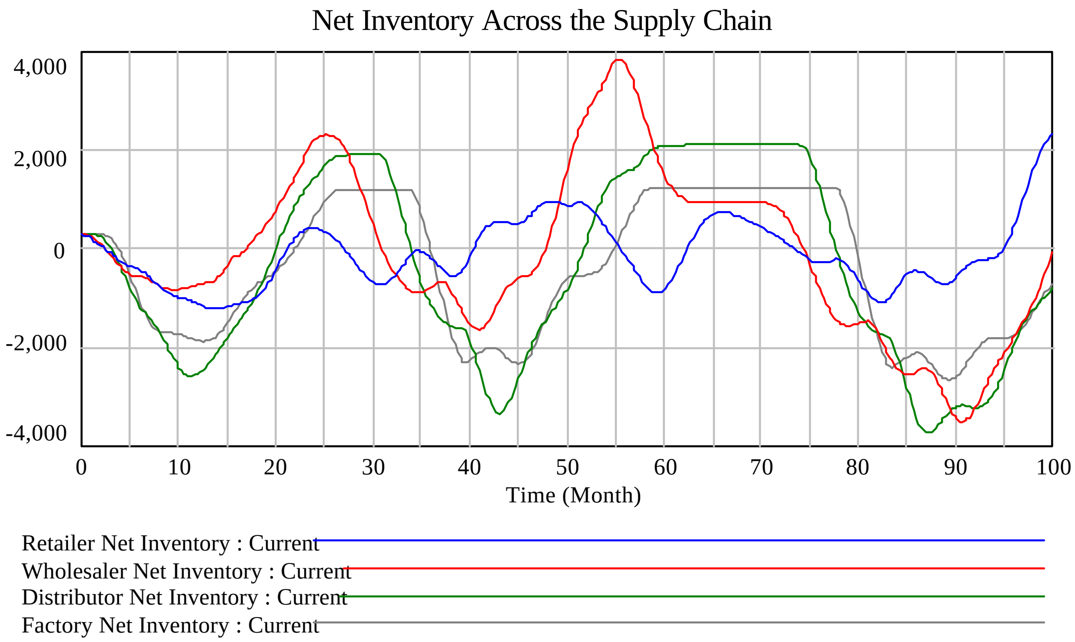
Comments on Experiment 1

- Around time 20, the wholesaler changes policy, and the distributor and factory quickly follow (a policy change triggers a confidence boost).

- From that point onwards, as all policies take account of the supply line, the overall supply chain behaviour stabilises, and oscillations and damped (i.e. the system reaches equilibrium), and confidence levels soar.

Experiment 2: Retailer SL Policy On, All others Off (Cyclic Demand Pattern)



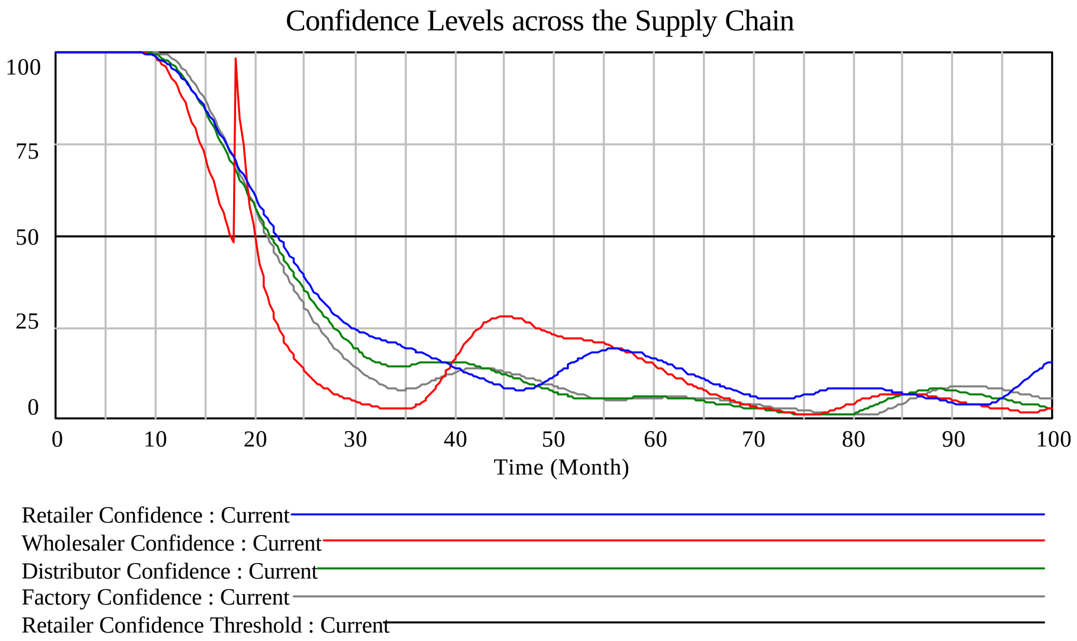


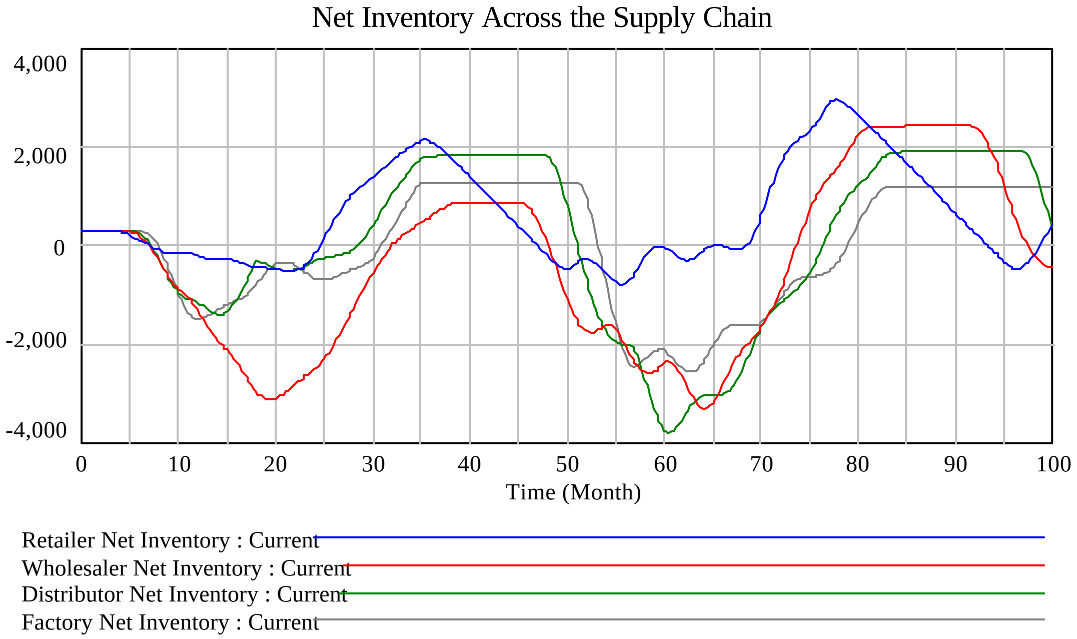
Comments on Experiment 2

- Around time 15, the retailer changes from the good policy to the poor policy. This seems counterintuitive, but it seems that the cyclic demand causes greater oscillation for the retailer (despite having the better heuristic).

- With the retailer switched to the poorer policy, the overall behaviour of the supply chain degenerates into significant oscillation.

Experiment 3: Wholesaler SL Policy On, All others Off (Step Demand Pattern)

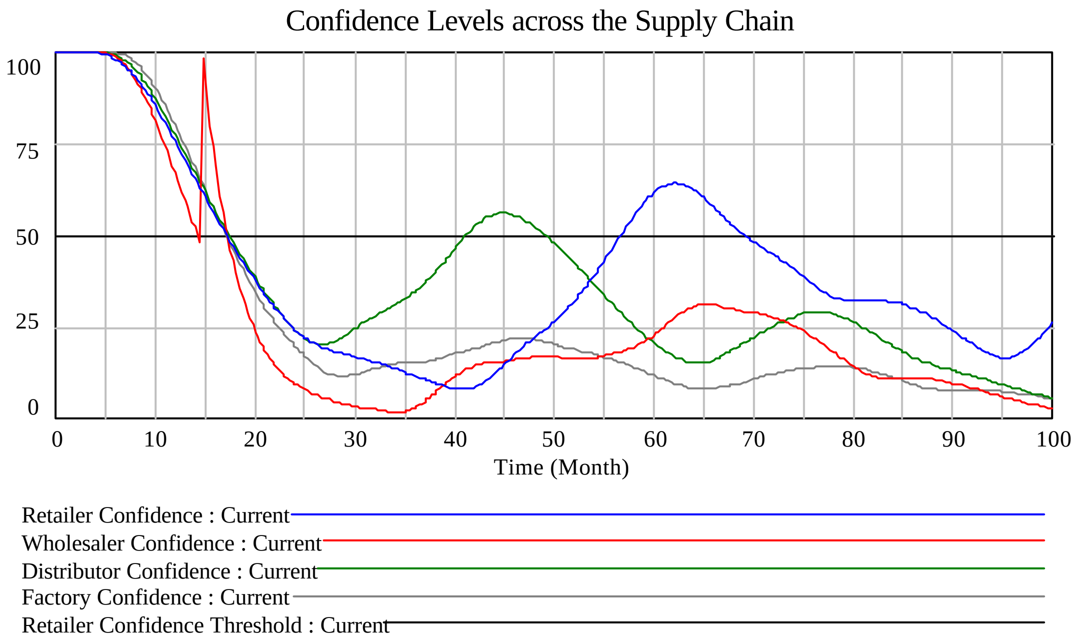


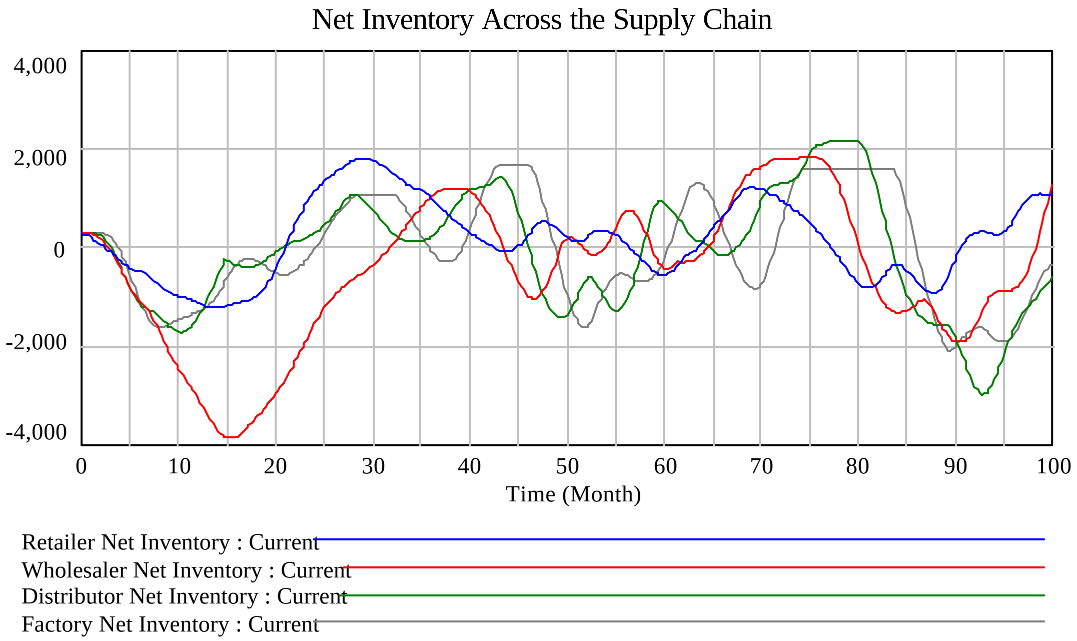


Comments on Experiment 3

- Around time 18, the wholesaler switches policy from using the supply line to ignoring the supply line. The temporary boost in confidence does not last long, and the overall supply chain performance is characterised by oscillation of all inventory levels.

Experiment 4: Wholesaler SL Policy On, All others Off (Cyclic Demand Pattern)





Comments on Experiment 4

- Policy change occurs around time 14, but, as with Experiments 2, 3 and 4, the change seems counterintuitive (i.e. the wholesaler is switching from a good policy to one that ignores that supply line).

The model contains over two hundred variables, so the four experiments presented only account for a small subset of the possible behaviours. The main findings of these experiments are summarised in Table 4.

Experiment	Initial Conditions	Behaviour
1	Step demand. Retailer has the best policy.	Policy diffusion occurs all the way through the supply chain. Oscillation is removed.
2	Cyclic Demand. Retailer has the best policy	Policy diffusion occurs, but in the opposite direction to experiment 1. This means that the Wholesaler performs better than the retailer, even though the wholesaler is employing the poorer strategy.
3	Step demand. Wholesaler has the best policy.	Policy diffusion occurs, but, as with experiment 2, the wholesaler "loses" their best strategy, because the retailer performance is better at the early stage of the model.
4	Cyclic Demand. Wholesaler has the best policy	Policy diffusion occurs, but, as with experiment 2 and , the wholesaler once again "loses" their best strategy.

Table 4 Summary of Experimental Results

Therefore, in three out of the four cases, the unexpected behaviour has surfaced, in that "best practice" has lost out. There are a number of factors at play here:

- First, the metric used to calculate the confidence of each agent is based on the out of bounds figure, and a comparison of the relative standard deviations of the two agents. There may be other metrics that can be used to assess the performance of an agent, ones that would more accurately pinpoint the strengths of the supply line policies, and help identify where the best performing agents are located.

- Second, the beer game is an excellent example of dynamic complexity. Perhaps the initial results are an indication that where significant oscillation occurs, it may be very difficult to pinpoint which agent is the primary cause of this oscillation, and so the good decision makers are indistinguishable from the poor decision makers.

Further research is needed to investigate which of these factors (or indeed what other factors) are the causes of these behaviours. A tabular set of results showing key simulation values is now presented in Figure 4.

Experiment	**Retailer Mean Inventory**	**Wholesaler Mean Inventory**	**Distributor Mean Inventory**	**Factory Mean Inventory**
1	70	271	33	6
2	-116	-118	-417	-441
3	566	-384	56	-67
4	165	-496	-104	-38

Experiment	**Retailer Inventory Stand. Deviation**	**Wholesaler Stand. Deviation**	**Distributor Inventory Stand. Deviation**	**Factory Inventory Stand. Deviation**
1	221	448	732	567
2	673	1651	1895	1372
3	1009	1755	1664	1220
4	793	1441	1185	1111

Experiment	**Retailer Out of Bounds**	**Wholesaler Out of Bounds**	**Distributor Out of Bounds**	**Factory Out of Bounds**
1	32	27	38	29
2	76	92	93	92
3	84	86	89	88
4	71	86	78	87

Figure 4 Summary of Model Statistics Across All Experiments

Finally, in reviewing our results it is important to mention that the policy control component contains decision rules expressed in the form of *if-then-else* statements. Such statements add to the complexity of the model and make it difficult to understand and maintain, although Sterman (2000) mentions that "conditionals can also be useful in representing switches to select among different policies or scenarios for model testing." Also, the policy state change mechanism, while it seems to work well in this situation, would be difficult to scale, as the number of connections needed between the policy states increases in a non-linear way (see Table 5).

Number of Different Policy States	**Total Number of Possible State Changes**
2	2
3	6
5	20
N	N(N-1)

Table 5 Calculation of Possible Policy State Changes

4. Conclusion

In conclusion, the model provides a basis to perform experiments on policy diffusion in the beer game. While the initial results are promising, future research needs to focus on:

- Identifying what metrics are most appropriate to measuring the success or otherwise of performance in the beer game.

- Exploring ways of increasing the sophistication and number of business rules, so that the model can deal with experimentation with a larger number of heuristics.

5. References

1. Senge, P. 1990. *The Fifth Discipline: The Art and Practice of the Learning Organisation.* New York: Doubleday.
2. Sterman, J. 1989. Modeling Management Behaviour: Misperceptions of Feedback in a dynamic decision making experiment. *Management Science.* 35(3), 321-339.
3. Sterman, J. 2000. *Business Dynamics: Systems Thinking and Modeling for a Complex World.* McGraw-Hill Higher Education.

Back to the Top