This is a working paper based on unpublished experimental results. Some of the analysis used proprietary software which is no longer available for this purpose. Someday, this paper will be reworked as an exercise for analyzing experimental results.

Agent-based modeling and experimental economics: exploring the behaviors behind the outcomes

David S. Dixon
Department of Economics
University of New Mexico

December 2004

ABSTRACT

A series of experiments was conducted on a single set of subjects to investigate learning and correlated responses. Altruism was inferred from an ultimatum game, learning was inferred by comparing the results of two successive beauty contest experiments, and iterative thinking was inferred from the beauty contest results. This paper presents a process for numerically modeling the experimental process, then developing an agent-based model (ABM) with a parameter space informed by the numerical analysis. A single candidate ABM is presented for altruism, and another for iterative thinking. Two simple behavior rules were able to reproduce the iterative thinking and altruism results, nearly identically to the experimental results in the case of altruism.

Background

In November 2004, a series of experiments was conducted[1] at University of New Mexico to examine any link between iterative thinking and altruism[1]. Iterative thinking was captured in a *beauty contest*[2], while altruism was captured in an *ultimatum game*[3]. Additionally, the beauty contest was run twice, with a brief illustrative example between them, and the results correlated by subject as a measure of learning.

There were four parts to the experiment:

1. Beauty contest A
2. Example beauty contest
3. Beauty contest B
4. Ultimatum game

The beauty contests were intended to reveal *iterator clusters* – groups of subjects with similar depths of iterative thinking. The instructions were to guess two-thirds of the group mean. A zero-order iterator is one who picks the *anticipated mean* (the experiment revealed two consensus values for the anticipated mean). A first-order iterator picks two thirds of the anticipated mean. An order-two iterator picks two-thirds of that value, and so on.

---

1 The experiment was designed and conducted by Curt Shepherd while at the University of New Mexico Department of Economics. Results used by permission.

Analysis of Experiment Data

The analysis that follows might typically be relegated to an appendix or omitted altogether. In computer modeling, however, defining the problem well is typically the biggest hurdle. The first step, therefore, is thorough analysis of the data. The goal is not rigorous analysis of the experimental result, but rather an exploration of evidence of causal behaviors that could explain the results.

First, the beauty contest data are examined. The initial approach is to assess the combined data from both beauty contests combined, assuming that this is similar to two different data sets.[2] Figure 1 is a histogram of the raw data.

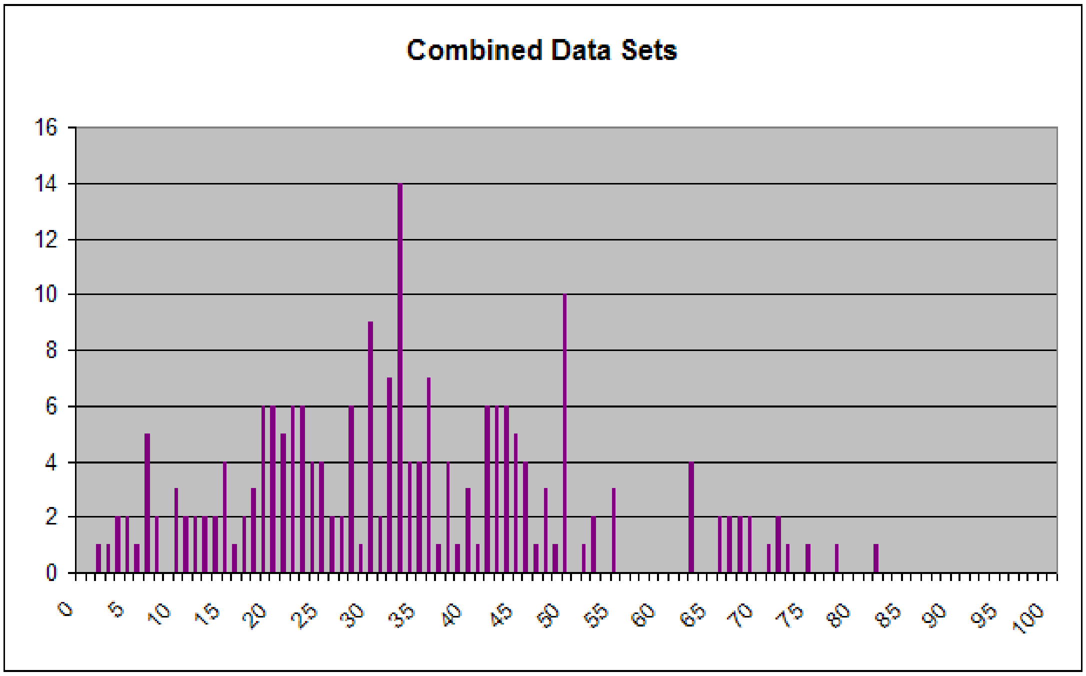

Figure 1 - Raw data

Suppose that the subjects make small numerical errors in estimating the mean or in computing two-thirds of it. Suppose, also, that those errors are normally distributed with constant variance. A normal distribution about each possible guess from 0 to 100 with constant variance is convolved with the data and the variance adjusted to maximize the coefficient of determination, R^2. Figure 2 shows the convolution with standard deviation 0.412, the value that maximizes R^2 at 0.994425.

The residuals from the convolution are shown in Figure 3. The overall mean is -6.8×10^{-5} with a slope of 1.37×10^{-6}. This suggests that the error does not depend on the magnitude of the guess. The small standard deviation indicates very little overlap between choices, implying that the data reflect the subjects' true preference.

Next, the variance of the normal convolution is increased until the chi-squared of the residuals equals the degrees of freedom, 207. This occurs for a standard deviation of 1.1215 and is shown in Figure 4. This is the maximum entropy fit: it minimizes all assumptions beyond the initial normal convolutions. Note that there are distinct peaks at

2. It could be argued that the second group of subjects is simply more self-aware than the first group, at least in terms of iterative thinking.

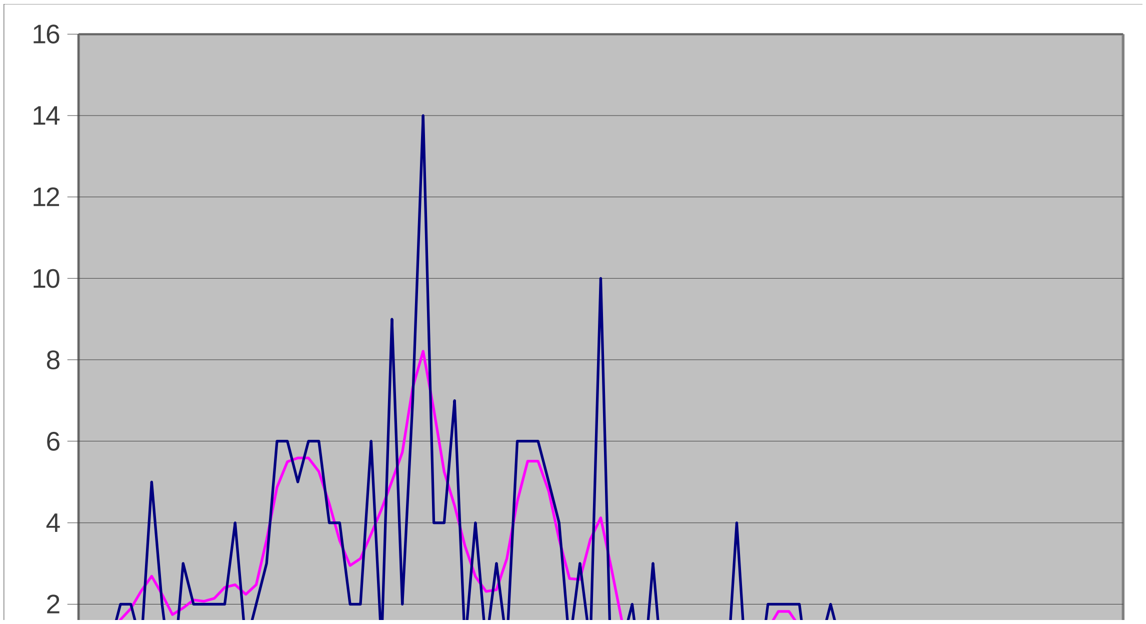

Figure 2 – Convolution with best-fit normal

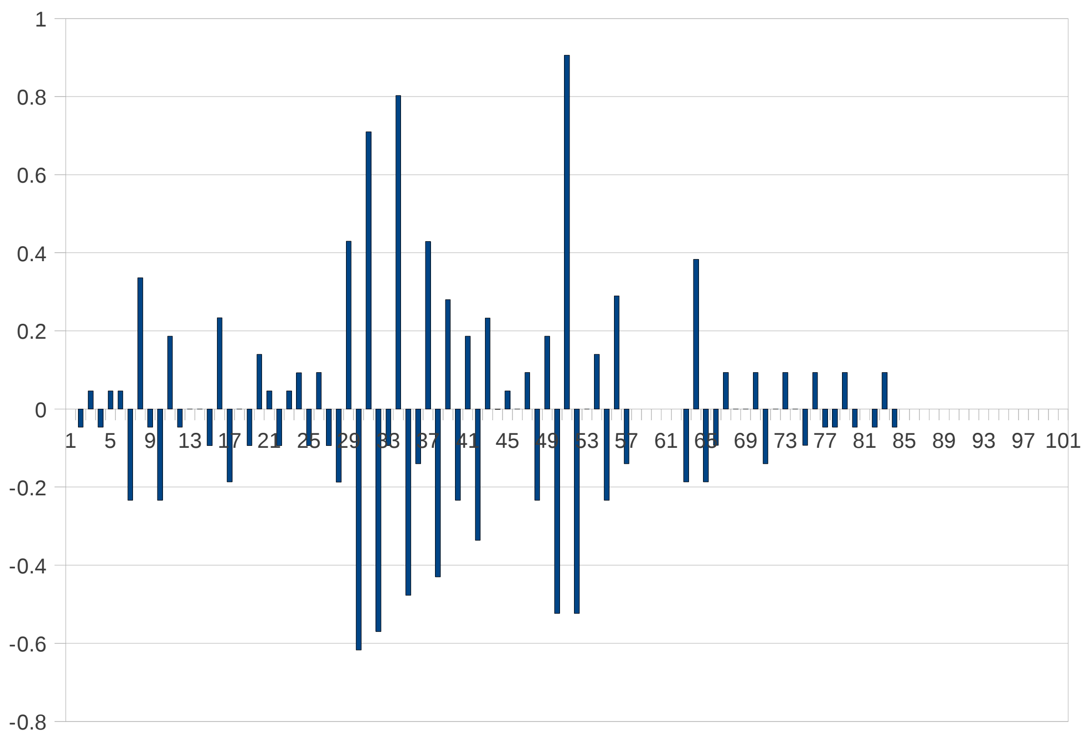

Figure 3 – Distribution of residuals from best-fit normal

8, 22, 34, 45, 51, 64, 69, and 73. These are the maximum entropy iteration clusters.

Another form of convolution is Fourier analysis, in which the data are convolved with sinusoids of various frequencies. Since iterative thinking represents a kind of harmonic, it is possible that Fourier analysis may capture those harmonics. An important first application, however, is to filter out completely random guesses.

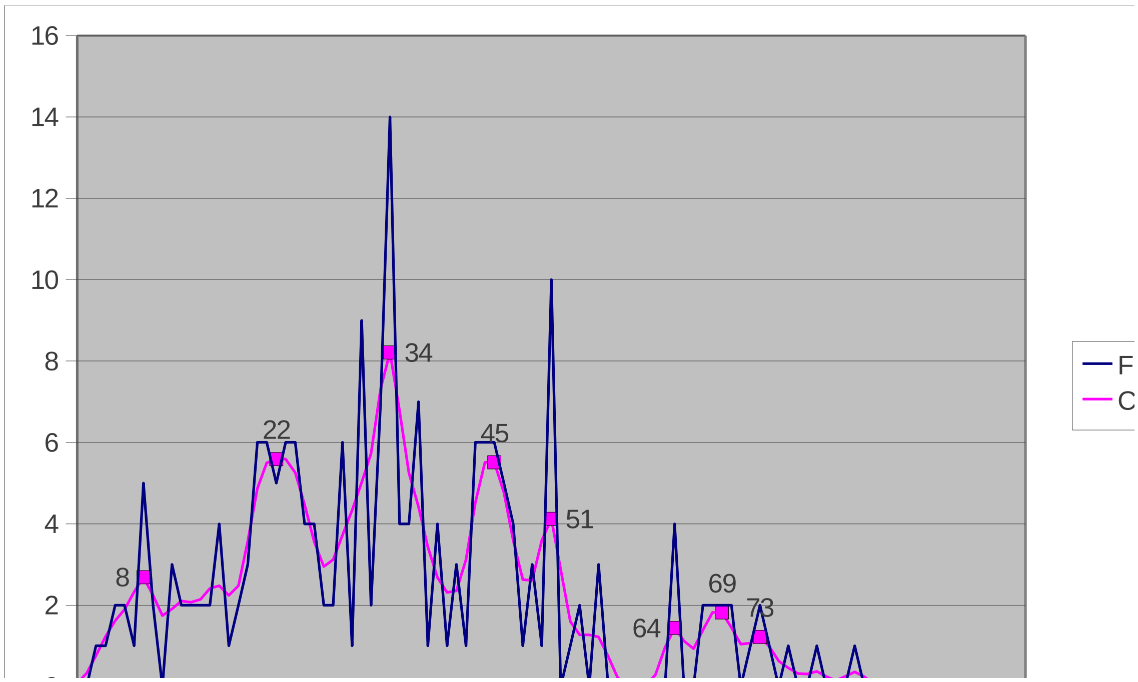

Figure 4 – Maximum entropy normal convolution

The basic idea of Fourier filtering is to transform series data into a frequency spectrum and filter out specific parts of the spectrum. In the case of the beauty contest data, clusters of choices appear to be superimposed over a background of random choices distributed between one and fifty (particularly in the beauty contest B). The goal is to filter out the high frequency (oscillations every one, two, three, four or five intervals) leaving only the large-structure data (clusters about iteration levels, primarily). Fast Fourier transforms (FFT) can only be done for numbers of observations that are powers of two. Since most of the detail occurs for guesses less than about 70, a cutoff of 64 was used and is shown in Figure 5. The original data are shown as *Input*, the high-frequency random guesses are shown as *Noise*. If the high-frequency components are discarded, the the lower frequency components will show only the larger features, as shown as the *Filtered* data.

Note that the peak in the mid-60s is still significant. This peak is problematic because it reflects an unanticipated initial assumption on the part of the subjects. The instructions were to estimate two-thirds of the group mean. If a subject assumes that guesses are distributed uniformly between 0 and 100, then the mean is 50. A non-iterative thinker will guess 50. A one-iteration thinker will guess two-thirds of that, or 33, a two-iteration thinker will guess two-thirds of 33, or 22, and so on. The peaks at 34 and 22 suggest this group.

If, however, a one-iteration thinker begins with the maximum guess of 100 and guesses 66, then a two-iteration thinker will guess 44, and so on. The peaks in the mid-60s and mid-40s are evidence for this latter group.

Thus, the filtered data are convolved with normal distributions about the iteration peaks of 50, 33, 22, 15, 10 and 7 (iterating from anticipated mean of 50) as well as 44, 29 and 19 (iterating from maximum of 100). The center of each peak is adjusted to maximize R^2. The results are shown in figure 6.

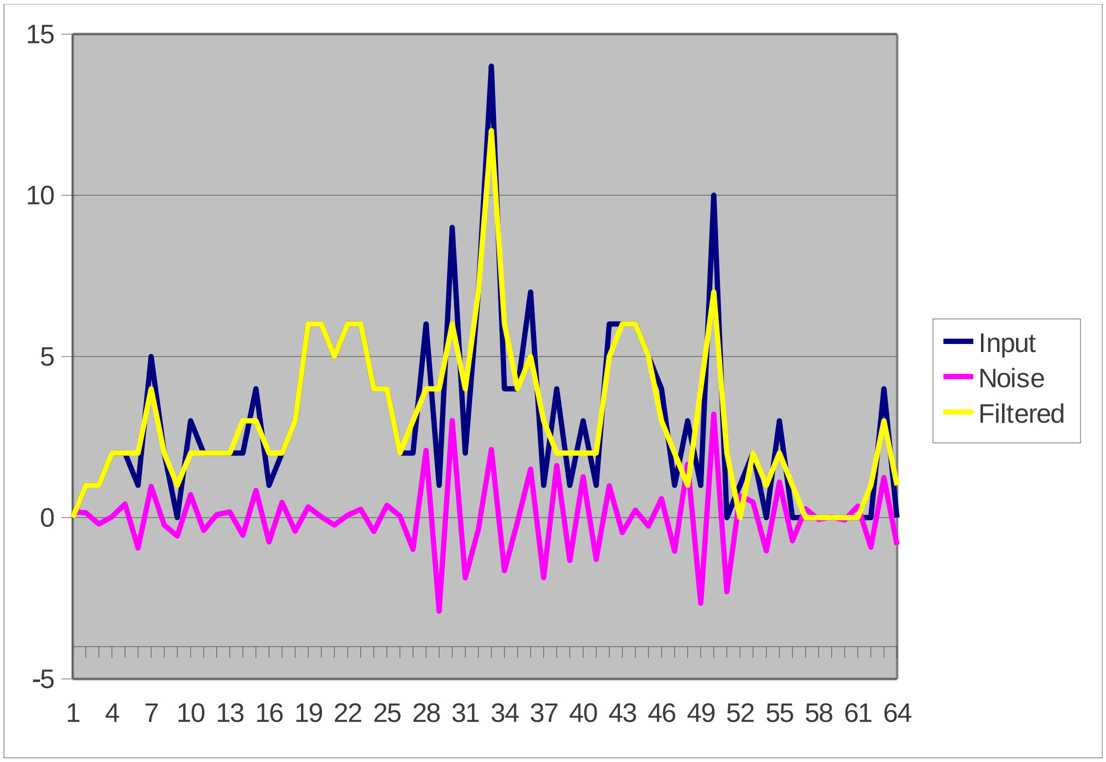

Figure 5 – Fourier noise filtering

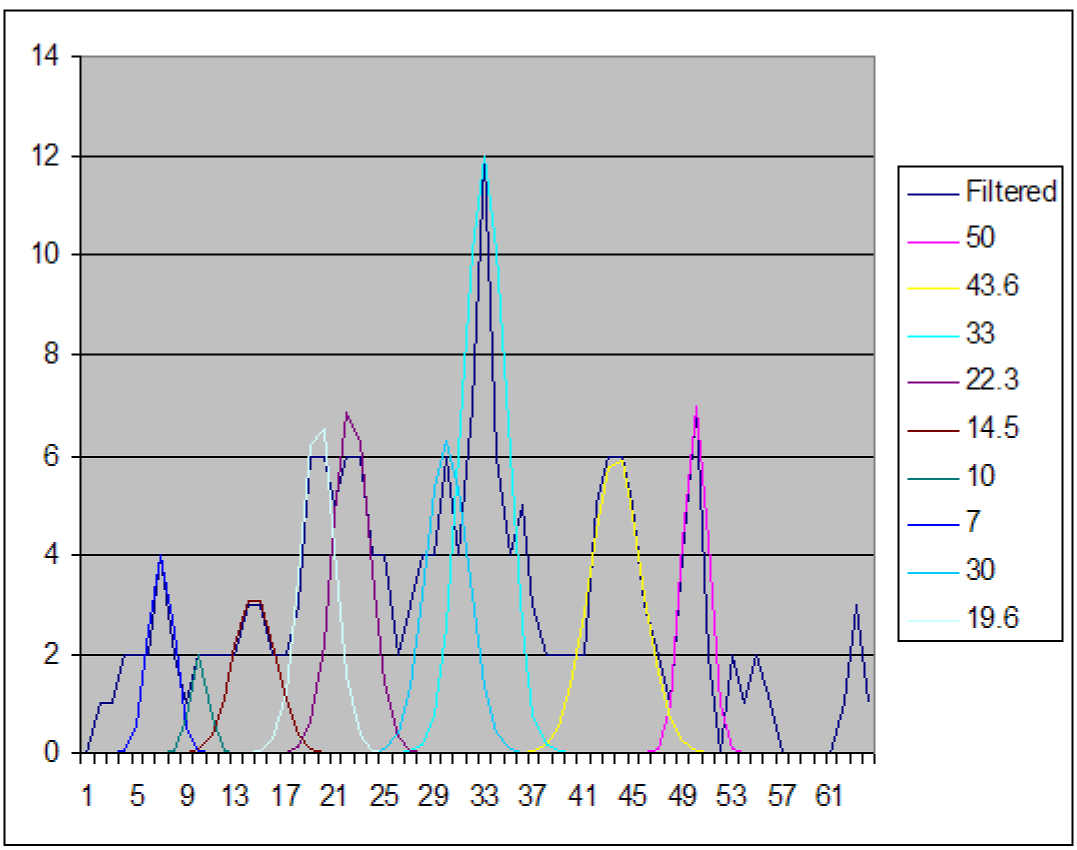

Figure 6 – Clusters fit to the Fourier filtered data

Because Fourier filtering is a harmonic technique, and iterations are a kind of harmonic, there is a possibility that artifacts were created. Another decomposition, using Haar wavelets (decomposition into simple step-up or step-down functions) is shown in Figure 7. This shows not only the same clusters as the Fourier analysis, but the same noise level (RMS 1.17).

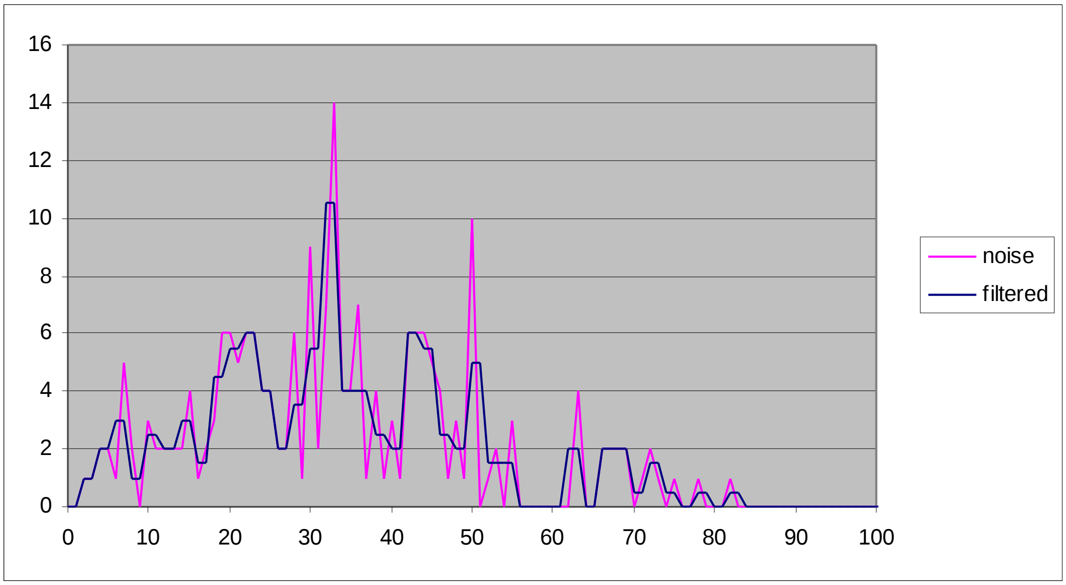

Figure 7 – Haar wavelet filtering

Similar to the preceding Fourier analysis, the Haar wavelet peaks are convolved at each of the iteration values. These results are shown in Figure 8. There is no power-of-two limitation on wavelets.

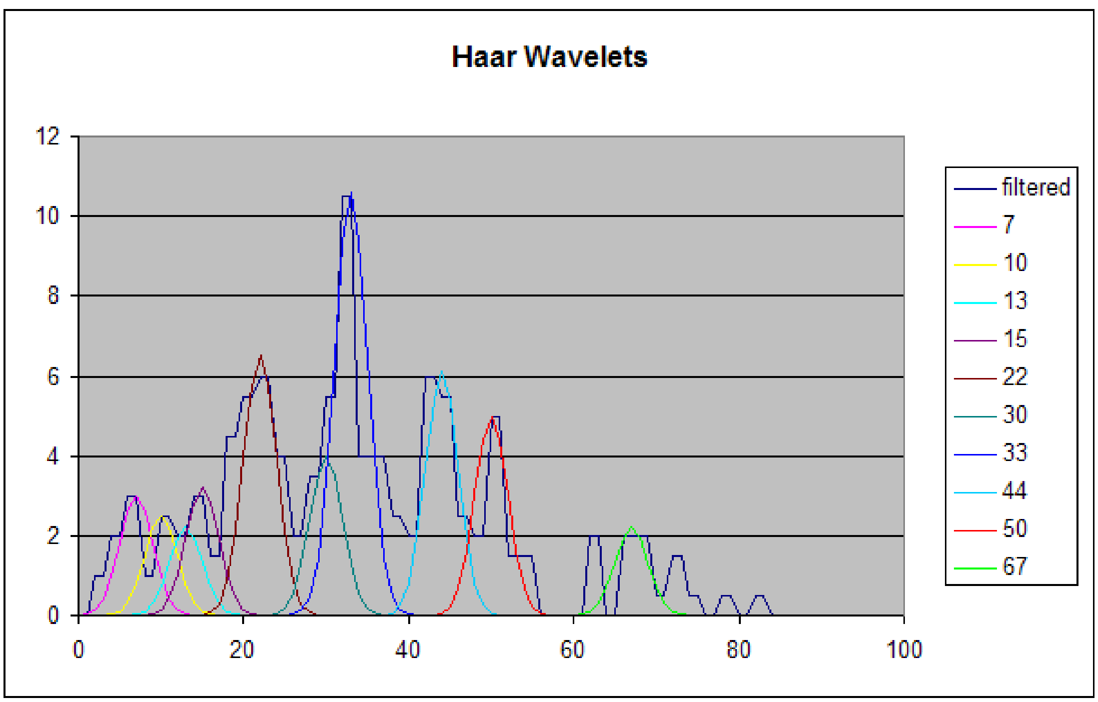

Figure 8 - Clusters fit to wavelet filtered data

Based on the Haar wavelet convolution, the subjects were then assigned to iteration clusters. This made it possible to assess both the distribution of iteration depth, and the distribution of starting points. There were two consensus starting points: the majority mean of 50, and the minority full-range value of 100. This analysis is shown in Tables 1 and 2 and Figure 9. Note that only 65 percent of subjects fell clearly within an iterator cluster. Thus, for 35 percent of subjects, either the experiment lacked salience, the instructions were not understood, or there was another behavior not captured in this study.

Table 1- Mean-based Clusters

Iterations	Center	Width	Count
0	50	2	4
1	33	4	16
2	22	2	4
3	15	2	5
4	10	2	2
5	7	4	5

Table 2 - Range-based Clusters

Iterations	Center	Width	Count
0	100	-	-
1	67	8	1
2	44	4	14
3	30	2	6
4	20	2	8
5	13	2	3

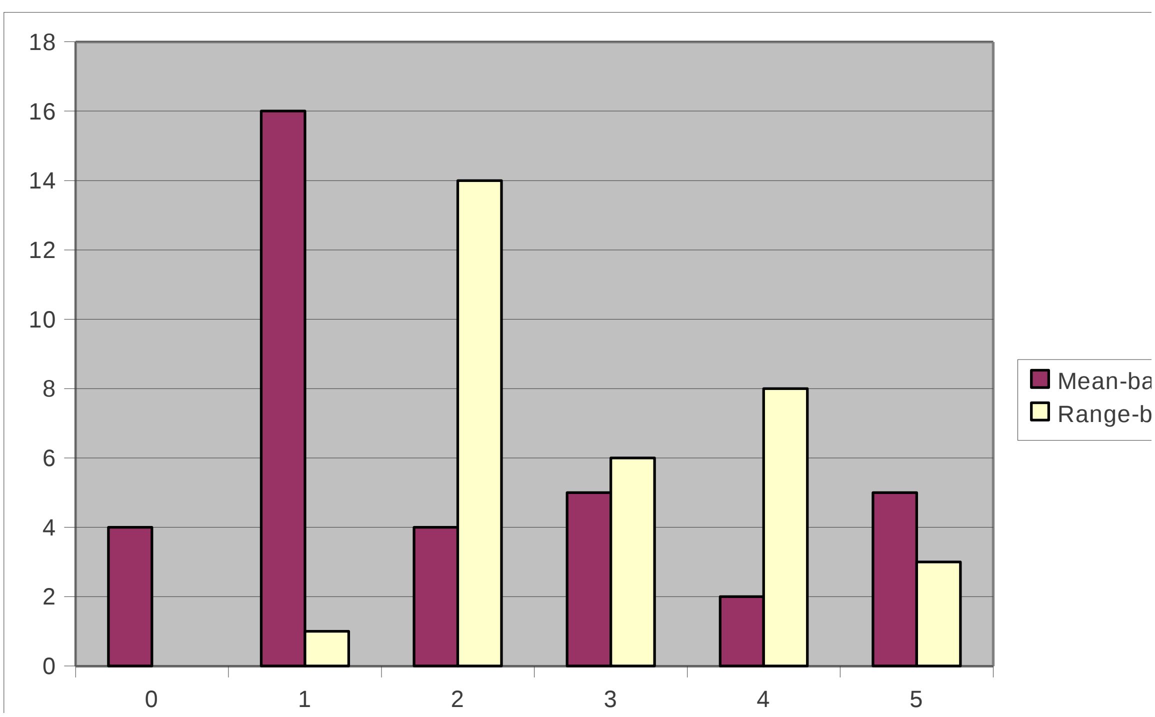

Figure 9 – Distribution of mean-based and range-based iterators.

Although the learning component of the experiment was not used in the initial model, there were some interesting results of the data analysis. Figure 10 shows the in-sample changes between beauty contest A (BCA) and beauty contest B (BCB) choices. The lines originate at the BCA choice, and terminate with a diamond at the BCB choice.

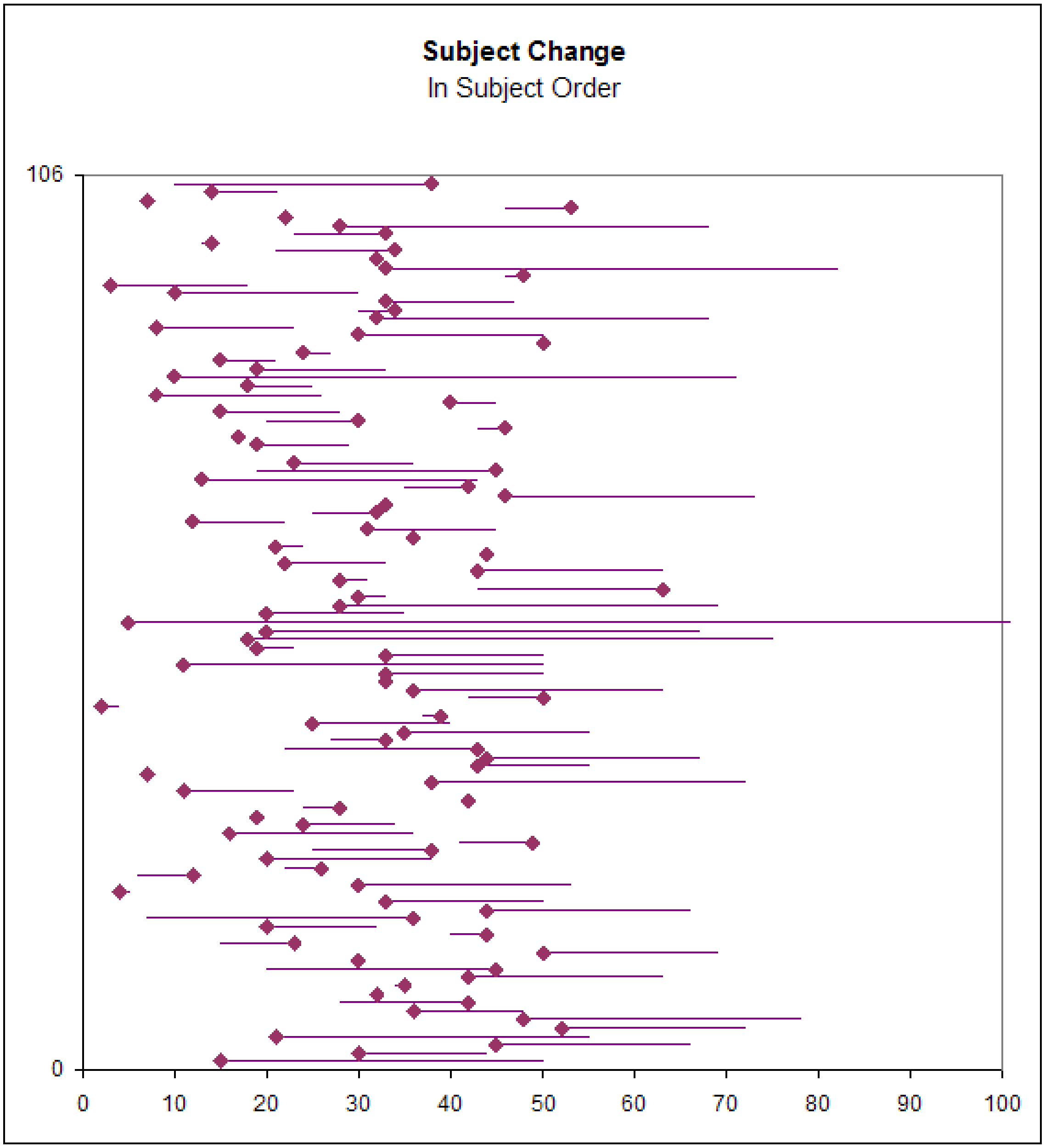

Figure 10 – Beauty contest: learning ordered by subject

Figure 11 shows the same data sorted by BCA choice. This is a good illustration of the corrections by subjects with BCA choices greater than 50.

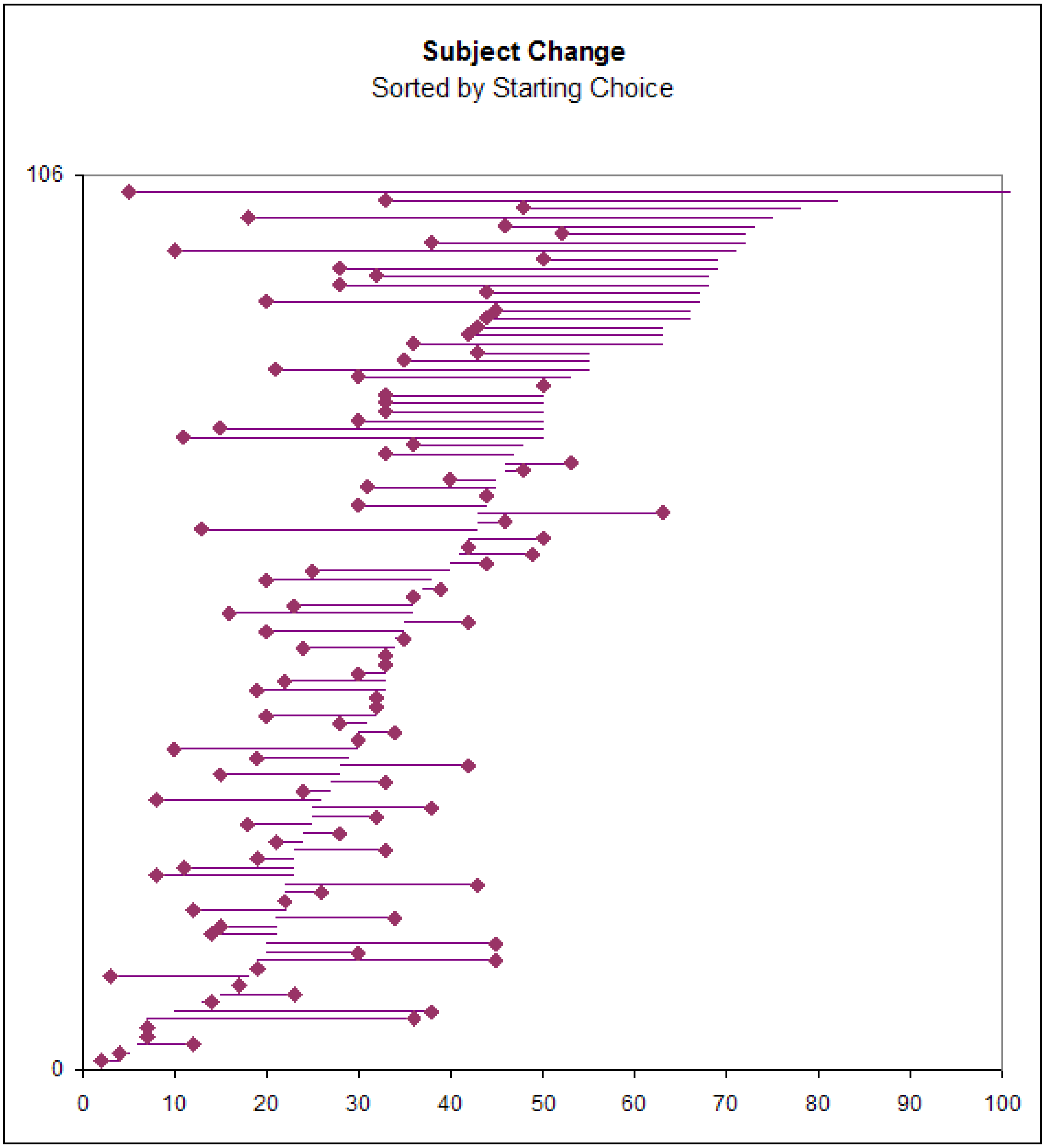

Figure 11 – Beauty contest: learning ordered by first round

Figure 12 shows the same data again, sorted by BCB choice.

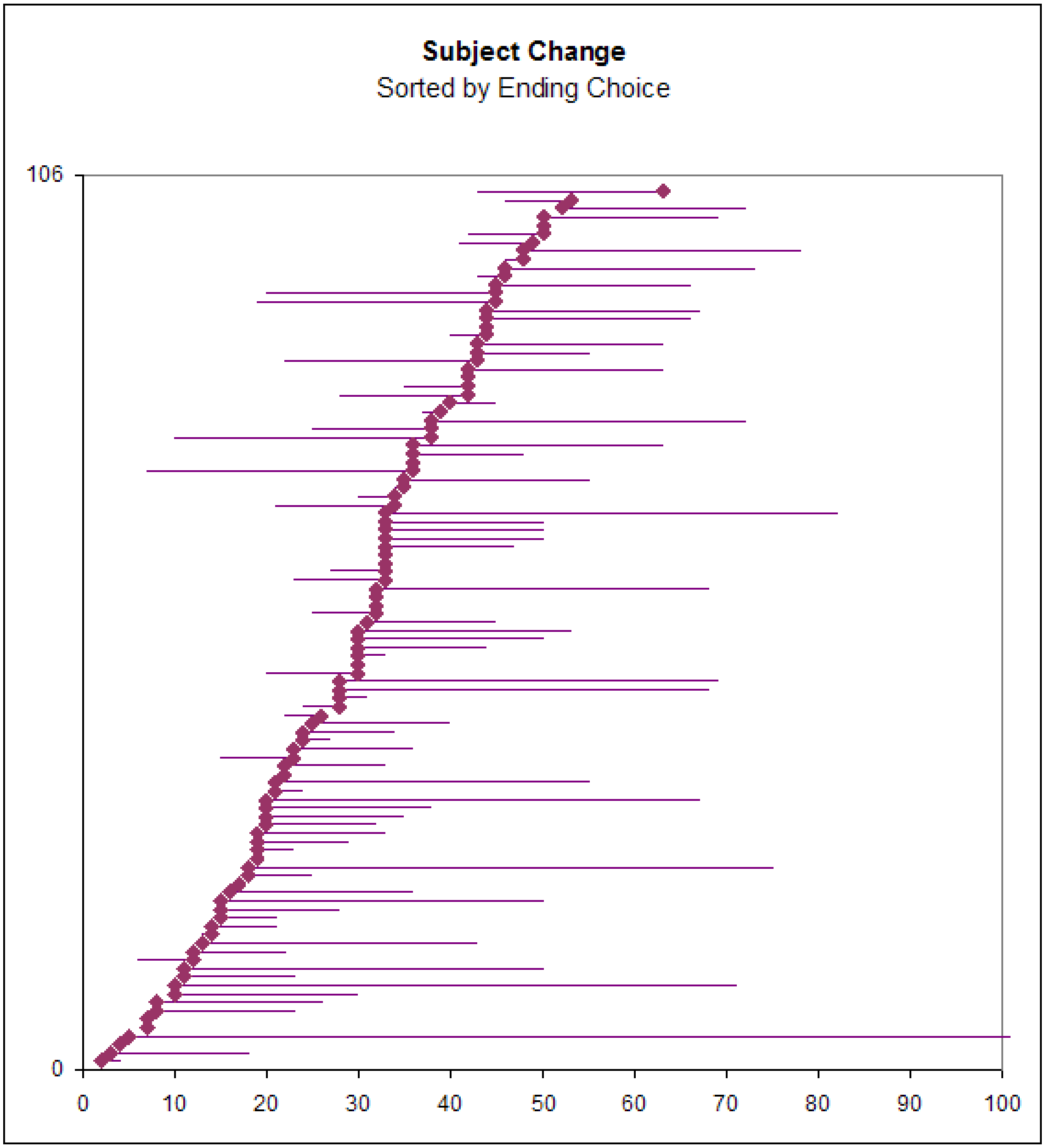

Figure 12 – Beauty contest: learning ordered by second round

Finally, Figure 13 is the same as Figure 12, superimposed on the mean-based clusters.

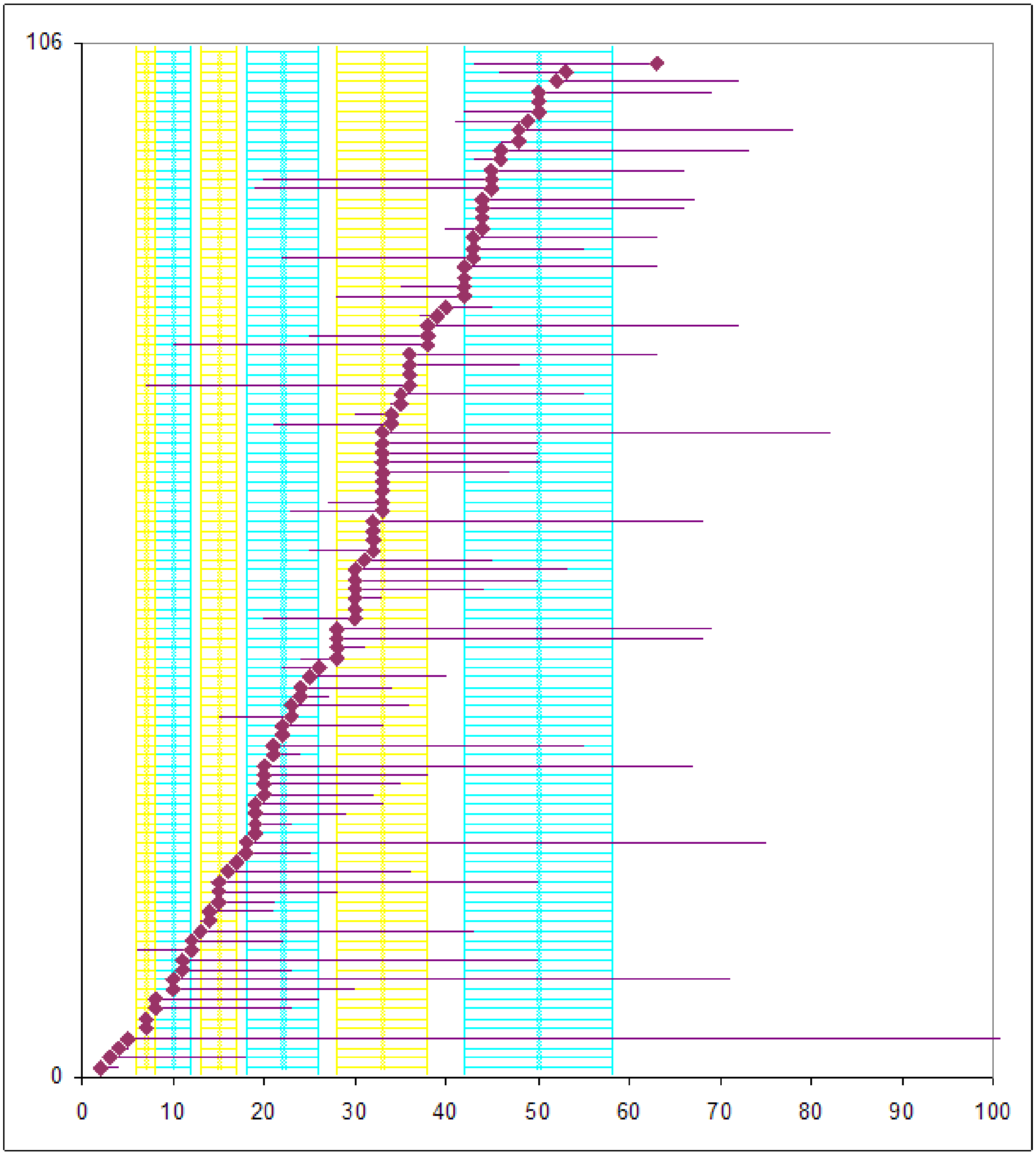

Figure 13 – Beauty contest: learning ordered by second round, showing clusters

In the ultimatum game, the subjects are paired, one subject is given ten dollars, and told to give anywhere from zero to ten dollars to the other subject. If the second subject is satisfied with it, both players keep the money. If, however, the other player is not satisfied, both players lose the money. Theoretically, the second subject should be happy with any amount of money greater than zero. In practice, however, some subjects reject shares of one dollar or even more.

The histogram for the ultimatum game is shown in Figure 14. Based on the four subjects giving more than 5, it was judged that the reward was not salient for 4 out of 105 subjects, for a salience of 96%. This is certainly overstated, since there were probably random answers below 5, as well.

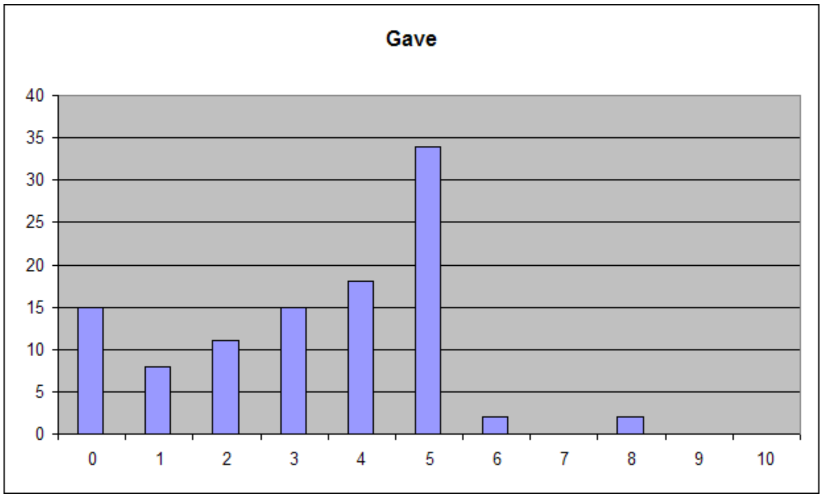

Figure 14 – Ultimatum game results

Agent-based model of the beauty contest results

The agent-based model was implemented in BASP[4] to take advantage of the fuzzy logic built into that system. The model consisted of 100 non-interacting agents. Agents were initialized in the following ways:

1. **Salience** selected the part of the subject population for whom the experiment was salient. This was estimated at 96% based on the experiment. **Salience** had categories **salient** and **nonSalient**. The value for each agent was initialized randomly with P(salient) = 0.96, P(non-salient) = 0.04.

2. **FirstEstimate** selected the part of the subject population which iterated from 50 and the part which iterated from 100. The experiment showed these to be 53% and 47%, respectively. **FirstEstimate** had three categories, **zero**, **fifty** and **hundred**. The value for each agent was initialized to P(fifty) = 0.53 and P(hundred) = 0.47.

3. **Choice** was the choice in the beauty contest. It was initialized based on **Salience**. Choice was used as a numerical variable. In the **salient** case, **Choice** was initialized to **FirstEstimate**. In the **nonSalient** case, it was set to a uniformly random value between 0 and 100.

4. **Iteration** was set based on **FirstEstimate**. **Iteration** had six categories, ranging from **zero** to **five**. If **FirstEstimate** was 50, **Iteration** was initialized to **one** with uniform random probability P(one) = 0.667, otherwise to a uniformly random value between **zero** and **five**. If **FirstEstimate** was 100, **Iteration** was initialized to **two** with uniform random probability P(one) = 0.5, otherwise to a uniformly random value between **one** and **five**.

5. **Altruism** was basis for the ultimatum game. **Altruism** had eight categories: **sociopath**, **misanthrope**, **grouch**, **oblivious**, **liberal**, **contributor**, **volunteer**, **activist**. **Altruism** was initialized with a uniformly random value across the range.

The BASP code for the initialization is shown in Figures 15 and 16.

Once initialized, the simulation was allowed to run for seven time steps. Two behavior rules were in effect during simulation:

1. On the second time step, each agent computed **Give**, the amount offered in the ultimatum game. In the **nonSalient** case, this was a uniformly random value between **zero** and **ten**. In the **salient** case, this value was computed from **Altruism**. If **Altruism** was in the highest category, the agent set **Give** to **five**. If **Altruism** was in one of the highest two categories, the agent set **Give** to a uniformly random value between **four** and **five**. If **Altruism** was in one of the three highest categories, the agent set **Give** to a uniformly random value between **three** and **five**. This continued to the lowest category. Figure 17 is a listing of the BASP code for this rule.
2. At each time step, the time step number was compared with each agent's **Iteration** value. If **Iteration** was greater than or equal to the current time step, the agent multiplied the current value of **Choice** by two-thirds. The BASP code for this is shown in Figure 18, and the algorithm shown in Figure 19.

The simulation was run ten times. The result for the ending value of **Give** for each run is shown in Figure 20. The mean of all ten simulations is shown in Figure 21. This is very close to the experimental data shown in Figure 10. The **Choice** data for all ten simulations are shown in Figure 22, and the mean over all ten simulations are shown in Figure 23.

```
IF my:Salience IS salient AND
    System:time > 0  AND
    System:time <= my:Iteration  THEN
   CHANGE my:Choice BY
         0.02*(your:BeautyContestRule - 1.0)*my:Choice.
   END IF
```

Figure 18 -- Behavior rule for iterative thinking

```
   -- set salience based on distribution
   --   (this may be used in the initialization of
   --    other variables)
   DEFINE SalienceFactor AS INTEGER.
   IF RANDOM(1.0) <= your:Salience THEN
      SET SalienceFactor TO 1.
      SET my:Salience TO salient.
   ELSE
      SET SalienceFactor TO 0.
      SET my:Salience TO notSalient.
   END IF

   -- set beauty contest choice
   --    set first (zero order) estimate (50 or 100)
   --    based on  the experimental distribution
   DEFINE FirstGuess AS INTEGER.
   IF RANDOM(1.0) < your:FirstEstimateDistribution THEN
       SET my:FirstEstimate TO fifty.
       SET FirstGuess TO 1.
   ELSE
       SET my:FirstEstimate TO hundred.
       SET FirstGuess TO 2.
       END IF

   -- define a noise factor
   DEFINE Noise AS REAL.
   SET Noise TO RANDOM(0.05) - 0.025.

   -- for the salient case, this will be FirstEstimate
   --    plus a little noise
   IF SalienceFactor IS 1 THEN
      SET my:Choice TO FirstGuess+Noise.
   ELSE
   -- in the nonsalient case, this will be noise
   --    across the full range
      SET my:Choice TO 0.02*RANDOM(100.0).
       END IF
```

Figure 15 -- Initialization rules

```
-- set level of iteration
DEFINE IterRandom AS REAL.
SET IterRandom TO RANDOM(1.0).
-- Cluster two-thirds of Normal iterators
--   at 'one', distribute the rest randomly
IF FirstGuess IS 1 THEN
   IF IterRandom < 0.4 THEN
      SET my:Iteration TO one.
   ELSE
      IF IterRandom < 0.65 THEN
         SET my:Iteration TO two.
      ELSE
         SET my:Iteration TO RANDOM(5.0).
         END IF
  END IF
-- Cluster half of Defective iterators
--   at 'two', distribute the rest randomly
IF FirstGuess IS 2 THEN
   IF IterRandom < 0.4 THEN
      SET my:Iteration TO two.
   ELSE
      IF IterRandom < 0.6 THEN
         SET my:Iteration TO three.
      ELSE
         SET my:Iteration TO 1+RANDOM(3.0).
         END IF
  END IF

-- set altruism
SET my:Altruism TO RANDOM(8.0).
```

Figure 16 -- Initialization rules (continued)

```
DEFINE GiveAway AS INTEGER.

IF System:time IS 2 THEN
   IF Salience IS salient THEN
      SET GiveAway TO
           Altruism - 5.0 + (11.0 - Altruism)*RANDOM(1.0).
      SET Give TO GiveAway.
   ELSE
      SET Give TO RANDOM(10.0).
      END IF
   END IF
```

Figure 17 – Behavior rule for the ultimatum game

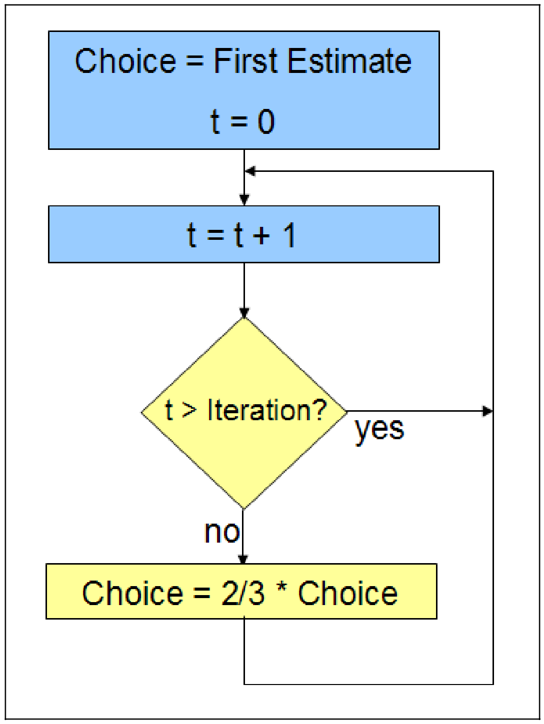

Figure 19 -- Setting the Choice variable

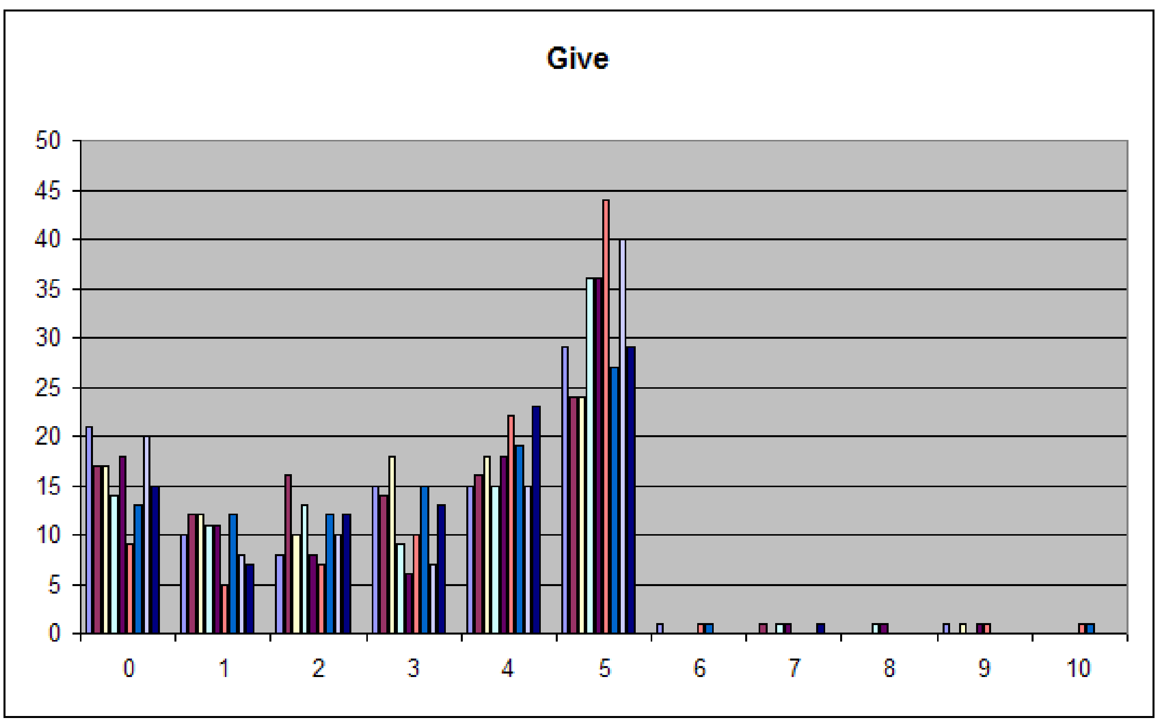

Figure 20 -- Results for Give over ten simulations

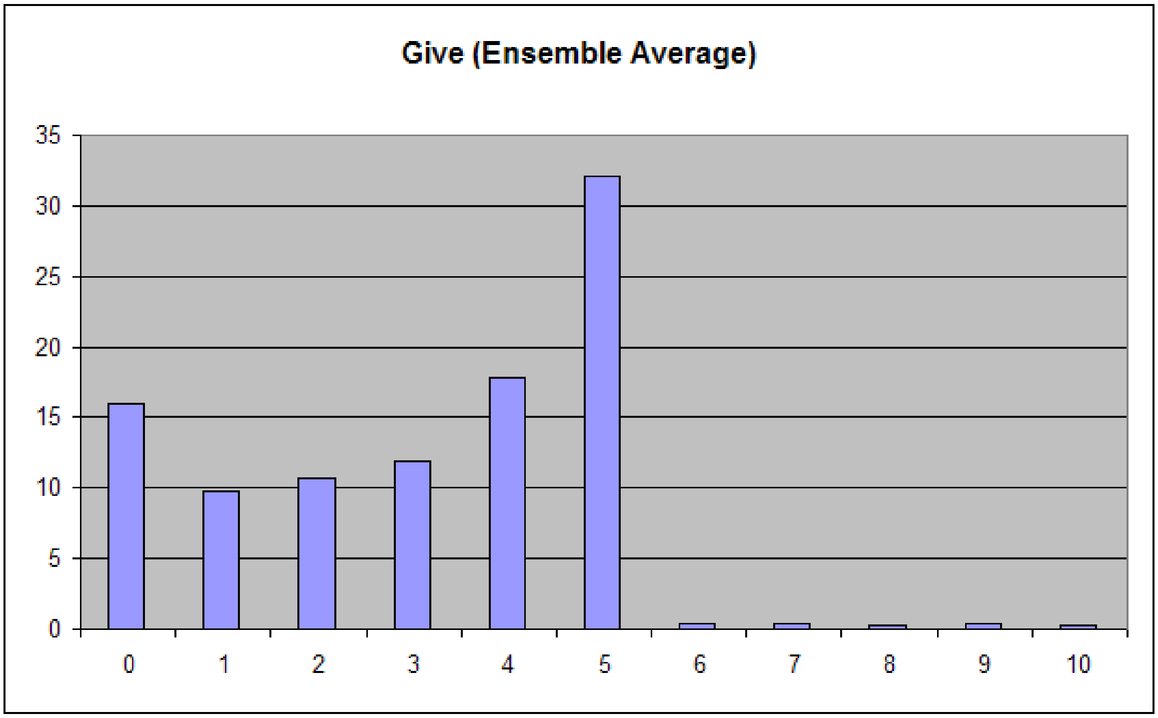

Figure 21 -- Average over ten simulations

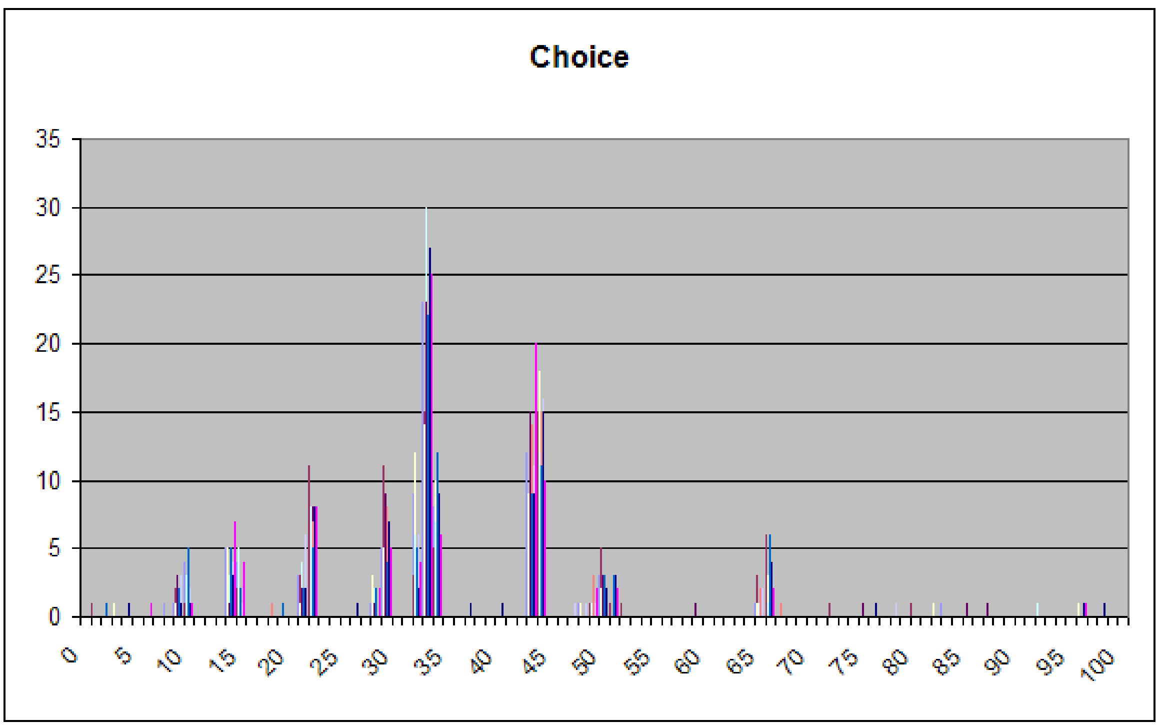

Figure 22 -- Results for Choice over ten simulations

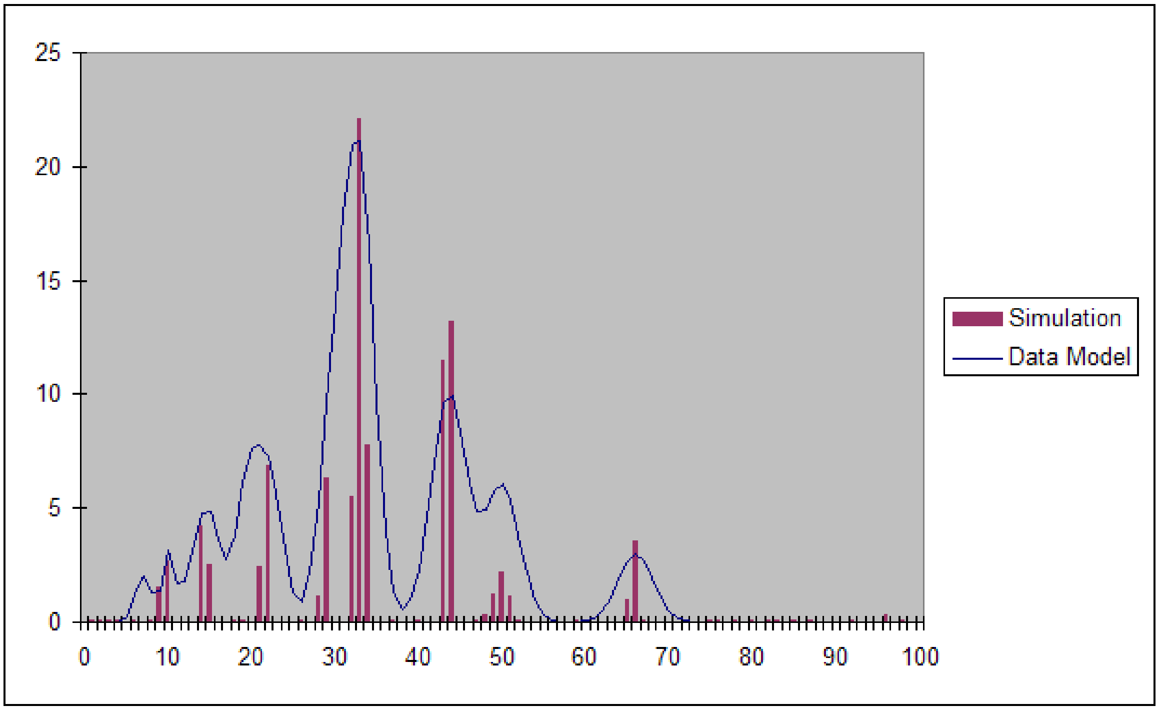

Figure 23 - Average over ten simulations

Conclusion

With minimal initial assumptions and only two simple behavior rules, a computer model closely reproduced the results from the Curt Shepherd Experiment. In the case of the ultimatum game – where the match between computer model and data is very close – this suggests that the underlying behavioral cause is either very simple, like the rule used in the computer model, or a simple composition of a large number of causes. For the beauty contest, the rules are more *ad hoc* and further experimentation and computer modeling are required. Additionally, the learning aspect of the experiment bears further investigation.

Notes

[1]

[2] Nagel, Rosemarie, 1995. Unraveling in guessing games: An experimental study. *American Economic Review*, 85, 1313-26.

[3] Güth, Werner, Rolf Schmittberger, and Bernd Schwarze. 1982. An experimental analysis of ultimatum bargaining. *Journal of Economic Behavior and Organization*, 3, 367-88.

[4] Reynolds, William N. and David S. Dixon, , A General Framework for Representing Behavior in Agent Based Modeling, *Complex Systems and Policy Analysis: New Tools for a New Millennium, Arlington VA, 27 September 2000. RAND Corporation Science & Technology Policy Institute.* http://www.leastsquares.com/papers/rand2000.pdf