2010

Evolutionary Models of Word-Meaning Associations

Daniel Shelby
University of Redlands

Follow this and additional works at: https://inspire.redlands.edu/cas_honors

Part of the Anthropological Linguistics and Sociolinguistics Commons, and the Computational Linguistics Commons

Recommended Citation

This work is licensed under a Creative Commons Attribution-Noncommercial 4.0 License
This material may be protected by copyright law (Title 17 U.S. Code).
This Open Access is brought to you for free and open access by the Theses, Dissertations, and Honors Projects at InSPIRe @ Redlands. It has been accepted for inclusion in Undergraduate Honors Theses by an authorized administrator of InSPIRe @ Redlands. For more information, please contact inspire@redlands.edu.
Evolutionary Models of Word-Meaning Associations

Daniel Shelby
April 24, 2010

1 Introduction

1.1 The Origin of Language

An important subject in linguistics, the study of language, is the origin of language. Questions about how human language arose are studied in the subfield of evolutionary linguistics. As all early and developing languages are long extinct, these questions are difficult to answer and until the 21st century, most linguists regarded them as unanswerable [1]. An exception occurs in a story told by Herodotus in book two of The Histories in which Psammetichos, king of Egypt, decides to find out who the first people were. He orders a shepherd to raise two newborns in a secluded hut but not to speak to them at all. Several years later, the children begin using the word "bekos" to ask for bread. Since "bekos" was the Phrygian word for bread, the king concludes that the Phrygians were the first people and Phrygian was the first language [2].

Due to ethical concerns, linguists have not duplicated Psammetichos' experiment. However, in the 1980's when the Nicaraguan government began a program to educate the country's deaf children, large numbers of children were brought together without a means of communicating, as their teachers did not know any existing sign languages. Very quickly, a sign language emerged. The early system developed by the older children was relatively rudimentary, but was soon developed into a rich language by the younger children. This development is one of the few opportunities linguists have to directly study the origin of language [3].
One method to study the origin of language indirectly is to develop mathematical and computer models to explore the development of different aspects of language. One aspect is the way in which associations between words and meanings developed. We wish to know how linguistic convention was established. That is, how groups of people came to associate the same word with a particular meaning.

1.2 Modeling Word-Meaning Associations

It is assumed that this development was not planned but arose by people acting in an uncoordinated fashion. It is also assumed that our earliest ancestors did not ask themselves, “How can I maximize my communicative success and thus be more likely to survive and reproduce?” That is, they did not choose which words to use intending to be successful communicators, but instead they used words that had been used successfully before. This distinction parallels the difference between regular game theory and evolutionary game theory. In regular game theory, players have a certain amount of knowledge about the game they are playing and the other players, and then choose a strategy that maximizes their expected payoff, or reward. In evolutionary game theory however, players do not choose a strategy at all and are not expected to have any knowledge of the game or the other players. Instead, the success, measured by payoff, of players using a particular strategy determines the degree to which that strategy will be used in the future. In biology, strategies are interpreted as genotypes which are passed on genetically to offspring; payoff determines the number of offspring and hence the prevalence of that genotype in the next generation. In the social sciences, such as linguistics, strategies can represent human behaviors that can be learned or imitated. The more successful a strategy is, the more likely it will be learned or imitated [4].

In the models presented here, we will make several assumptions about language. Firstly, it assumed that the population shares a set of meanings about which to discuss. This presumes for instance, that if colors are being discussed, the entire population distinguishes between colors in the same way. This may not be the case; there are for instance, languages that only distinguish between “dark” and “light,” and although the speakers are able to distinguish between blue and green, they are not different enough to warrant separate words [5]. Next it is assumed that word-meaning associations are arbitrary. With the exception of onomatopoeia, this seems to be the case
since, for instance, the word "robin" has no relation to flying animals other than in the minds of English speakers. Also, it is assumed that there are no genetically programmed associations between meanings and words. Next, with the exception of the model presented by Oliphant, et al, it is assumed that there is a word-meaning association in the mind of each member of the population that determines both the way words are chosen and how words are interpreted. Thus if the association between a particular meaning and word is strong, it is both likely that a speaker will use the word to express the meaning and that a hearer will interpret the word as the meaning. Oliphant, et al assume no such connection in their model. Finally, the number of available words is assumed to be finite and predetermined. This may be justified since early communication would probably have depended on simple and easily distinguished syllables that are limited in number [6].

The strength of a word-meaning association is represented by a real number $0 \leq a \leq 1$ where 1 represents a strong association, and 0 represents a very weak relation between the word and meaning. It is convenient to organize these numbers in $m \times w$ matrices where $m$ is the number of meanings and $w$ is the number of words. In such a matrix $A$, $a_{ij}$ will represent the association between meaning $i$ and word $j$. These matrices will be referred to as lexicons. Technically, a lexicon refers to the set of lexemes in a language. A lexeme is a set of grammatically related words that have the same basic meaning, for instance, *jump*, *jumped*, and *jumping* are forms of the same lexeme. However, in the grammar-free environment of these models, we will assume lexemes correspond to simple words.

2 Lenaerts, et al

Our first model was developed by Tom Lenaerts, Bart Jansen, Karl Tuyls and Bart De Vylder [6,7]. This model assumes that word-meaning associations are developed through communication among peers. Agents alter their lexicons as they communicate and can take the roles of hearer and speaker at different times. This model reflects a horizontal transmission structure where language is learned within a single generation. In the model, each agent is described by its lexicon. At each time step, two members of the population are randomly chosen to play a "naming game" and then these agents alter their lexicons as a result of the outcome of the game. For the naming game, a meaning $i_1$ is randomly chosen for the speaker to communicate. The column
j in the lexicon of the speaker that contains the largest entry of row \(i_1\) is chosen as the word used by the speaker. For convenience later, the rows of the speaking matrix will be normalized as will the columns of the hearing matrix. Then the row \(i_2\) of the hearer’s lexicon that contains the largest entry of column \(j\) is selected as the meaning that the hearer uses to interpret the speaker’s chosen word. If \(i_1 = i_2\), then the communication is successful, otherwise it fails. If communication is successful, the used word-meaning association is augmented by a factor \(\delta_s\) in both the hearer’s and speaker’s lexicons. Thus if \(p_{i_1j}\) is the \((i_1,j)\)th entry of the speaker’s matrix, \(P\), and \(q_{i_2j}\) is the equivalent entry in the hearer’s matrix, \(Q\), then their values at the next time step will be:

\[
    p_{i_1j}(t + 1) = p_{i_1j}(t) + \delta_s(1 - p_{i_1j}(t)) \\
    q_{i_2j}(t + 1) = q_{i_2j}(t) + \delta_s(1 - q_{i_2j}(t))
\]

The correspondence between meaning \(i_1\) and all of the unused words is weakened in the speaker’s matrix and unused meanings for the word \(j\) are weakened in the hearer’s matrix. Thus for \(h \neq i_1\) and \(k \neq j\),

\[
    p_{i_1k}(t + 1) = p_{i_1k}(t) - \delta_s p_{i_1k}(t) \\
    q_{hj}(t + 1) = q_{hj}(t) - \delta_s q_{hj}(t)
\]

All other word-meaning associations remain unchanged:

\[
    p_{hk}(t + 1) = p_{hk}(t) \quad \forall h \neq i_1 \\
    q_{hk}(t + 1) = q_{hk}(t) \quad \forall k \neq j
\]

If, however, communication fails and \(i_1 \neq i_2\), then the word-meaning associations used by the speaker and the hearer are weakened as determined by a parameter \(\delta_f\):

\[
    p_{i_1j}(t + 1) = p_{i_1j}(t) - \delta_f p_{i_1j}(t) \\
    q_{i_2j}(t + 1) = q_{i_2j}(t) - \delta_f q_{i_2j}(t)
\]

Then the association between meaning \(i_1\) and all of the unused words is strengthened in the speaker’s matrix and similarly for word \(j\) and all unused meanings. Thus for \(h \neq i_2\) and \(k \neq j\),

\[
    p_{i_1k}(t + 1) = p_{i_1k}(t) + \delta_f((w - 1)^{-1} - p_{i_1k}(t))
\]
\[ q_{hj}(t + 1) = q_{hj}(t) + \delta_f((m - 1)^{-1} - q_{hj}(t)) \]

All other word meaning-associations remain unchanged.

Their results, computed with a program in Maple using \( \delta_s = \delta_f = .1 \), (Figure 1) shows twenty runs of the program with ten agents, \( m = w = 7 \), and 10000 timesteps. Figure 2 [6] shows a run under similar conditions from Lenaerts, et al.

![Graph showing successful communication over time](image)

Figure 1: Percent successful communication over time

The resulting lexicons are nearly identical for the entire population and indicate that everybody is speaking in the same way. In addition, each row has an entry very close to one and all others very close to zero. This means that there are no synonyms, different words for the same meaning, in the language. If \( m \leq w \), each column will have an entry very close to one and all others very close to zero. If this is the case, then the language has no homonyms, the same word with different meanings.

The model can be related to an asymmetrical evolutionary game theory model [6]. First, we must reinterpret the entries of the speaking and hearing matrices as probabilities, which is possible since these matrices are already appropriately normalized. Then the rows of the speakers’ lexicons correspond to sets of strategies since a speaker chooses to use word \( j \) with probability \( p_{ij} \).
Similarly, the columns in the hearers' lexicons correspond to sets of strategies. Letting $\beta(t)$ equal 0 if communication is a success and 1 if communication is a failure we can combine the success and failure update equations above:

$$
\begin{align*}
  p_{ij}(t+1) &= p_{ij}(t) + \delta_s(1 - \beta(t))(1 - p_{ij}(t)) - \delta_f \beta(t)p_{ij}(t) \\
  q_{ij}(t+1) &= q_{ij}(t) + \delta_s(1 - \beta(t))(1 - q_{ij}(t)) - \delta_f \beta(t)q_{ij}(t) \\
  p_{ik}(t+1) &= p_{ik}(t) - \delta_s(1 - \beta(t))p_{ik}(t) + \delta_f ((w - 1)^{-1} - p_{ik}(t)) \\
  q_{kj}(t+1) &= q_{kj}(t) - \delta_s(1 - \beta(t))q_{kj}(t) + \delta_f ((m - 1)^{-1} - q_{kj}(t))
\end{align*}
$$

Then given that the meaning $k$ is chosen in the naming game, the expected value of the change in $p_{kl}$, $E[\Delta p_{kl}]$ will be $p_{kl}(t)$ multiplied by the first equation minus $p_{kl}(t)$ plus $\sum_{h \neq k} p_{kh}(t)$ times the third equation minus $p_{kl}(t)$.

$$
E[\Delta p_{kl}] = p_{kl}(t)(\delta_s(1 - \beta(t))(1 - p_{kl}(t)) - \delta_f \beta(t)p_{kl}(t)) \\
+ \sum_{h \neq k} p_{kh}(t)(\delta_s(1 - \beta(t))p_{ik}(t) + \delta_f ((w - 1)^{-1} - p_{ik}(t)))
$$

Since the columns of the hearing matrix are normalized, $\sum_{r=1}^{m} q_{rl}(t) = 1$ and so we can multiply the equation by this factor and distribute.

$$
E[\Delta p_{kl}] = p_{kl}(t)(\sum_{r=1}^{m} q_{rl}(t))(\delta_s(1 - \beta(t))(1 - p_{kl}(t)) - \delta_f \beta(t)p_{kl}(t))
$$
\[ + \sum_{h' \neq l} p_{kh'}(t) \left( \sum_{r=1}^{m} q_{rl}(t) \right) \left( -\delta_s(1 - \beta(t)) p_{i \cdot k}(t) + \delta_f((w - 1)^{-1} - p_{i \cdot k}(t)) \right) \]

Combining terms gives the next equation:

\[
E[\Delta p_{kl}] = p_{kl}(t)\left( \sum_{r=1}^{m} \delta_s(1 - \beta(t)) q_{rl}(t) - \sum_{h=1}^{w} p_{kh}(t) \sum_{r=1}^{m} \delta_s(1 - \beta(t)) q_{rl}(t) \right)
\]

\[
-\delta_f \beta(t) p_{kl}(t) \sum_{h=1}^{w} p_{kh}(t) \sum_{r=1}^{m} q_{rl}(t) + \frac{\delta_f \beta(t)}{w - 1} \sum_{h' \neq l} p_{kh'}(t) \sum_{r=1}^{d} q_{rl}(t)
\]

Let \( A \) be a \( m \times w \) matrix with the \( k \)th row consisting of all entries equal to \( \delta_s(1 - \beta(t)) \) and all other rows full of zeros. Then let \( \mu = \delta_f \beta(t) \). Substituting \( A \) and \( \mu \) into the equation and using matrix multiplication we get:

\[
E[\Delta p_{kl}] = p_{kl}(t)(e_k A q_l(t) - p_k(t) A q_l(t)) - \mu p_{kd}(t) + \frac{\mu}{w - 1}(1 - p_{kd}(t))
\]

where \( e_k \) is a row vector with a “1” in the \( k \)th position and zeros elsewhere, \( p_k(t) \) is the \( k \)th row of the speakers matrix, and \( q_l(t) \) is the \( l \)th column of the hearing matrix. This is in the same form as a replicator equation from evolutionary game theory with \( A \) replacing a payoff matrix and \( \mu \) representing a mutation rate as would be used in biology. A similar formula can be derived for the expected change of \( q_{kl} \) \[7\].

3 Nowak, et al

Three models developed by Martin A. Nowak, Joshua B. Plotkin, and David C. Krakauer, assume a vertical transmission structure in which each new generation develops word-meaning associations based on communication in the previous generation \[8\]. The previous generation is then entirely replaced by the new generation. In each model, each agent is described by a lexicon, \( A \). This matrix is transformed into the agent’s speaking matrix, \( P \), by normalizing the rows and the agent’s hearing matrix, \( Q \), by normalizing the columns. An entry \( p_{ij} \) of \( P \) represents the probability that the agent will use word \( j \) to signal meaning \( i \). Similarly an entry \( q_{ij} \) of \( Q \) represents the probability that an agent will interpret the word \( j \) as the meaning \( i \). Each time step represents a period in which all of the agents are trying to communicate with each other and gaining fitness from successful communication. If an agent
with language $L_1$ is communicating to an agent with language $L_2$ and wants to signal meaning $i$, then the probability of communicative success, that is, the probability that $L_2$ will interpret the word $L_1$ chooses as $i$, is:

$$\sum_j^w p^{(1)}_{ij} \times q^{(2)}_{ij}$$

were $w$ is the number of words, $p^{(1)}_{ij}$ is an entry of $L_1$'s speaking matrix, and $q^{(2)}_{ij}$ is an entry of $L_2$'s hearing matrix.

The success of $L_1$ communicating with $L_2$ is determined by the sum of the probabilities of success over all possible meanings:

$$\sum_i^m \sum_j^w p^{(1)}_{ij} \times q^{(2)}_{ij}$$

The payoff to an agent with language $L_1$ communicating with an agent with language $L_2$ is the average of the success of $L_1$ speaking to $L_2$ and $L_2$ speaking to $L_1$:

$$F(L_1, L_2) = \frac{1}{2} \sum_i^m \sum_j^w (p^{(1)}_{ij} \times q^{(2)}_{ij} + p^{(2)}_{ij} \times q^{(1)}_{ij})$$

The fitness of an agent is determined by the average success of communication with the rest of the population. Thus if $p$ is the size of the population, the fitness is

$$F_a = \frac{1}{p} \sum_{b \neq a} F(L_a, L_b)$$

3.1 Parental Learning

In the parental learning model, each agent in a new generation “selects” a parent from the previous generation [8]. The probability that an agent will be selected is equal to the agent’s fitness divided by the total fitness of the population. Once a parent has been selected, the offspring’s $A$ matrix is based on $k$ samples of words for each meaning from the parent. That is, for each meaning $i$, a word $j$ is selected based on the probabilities located in row $i$ of the parental speaking matrix $P$ and a 1 is added to the $a_{ij}$ entry of the offspring matrix $A$ (which is initially the zero matrix). This is then repeated $k$ times for each meaning. Figure 3 shows average fitness over time for 20 runs of a Maple simulation of the parental learning model. Runs that flatten out
at integer fitness levels indicate stable, universally adopted lexicons. The end lexicons contain a single “1” in each row and zero’s elsewhere. This means that the model eliminates synonyms. In the case where \( m \leq w \), fitness corresponds to the number of words being used. Then the maximum fitness is \( m \), and in this case, each meaning has its own word and there are no homonyms. Less than optimal fitness indicates languages with homonyms. The fact that the model generates more homonyms than synonyms corresponds to the observation that in natural languages, homonyms are plentiful but synonyms are rare and most synonyms have slightly different interpretations [6, ch 3].

![Figure 3: 100 agents, 5 words, 5 meanings](image)

### 3.2 Role-Model Learning

The parental learning model can be generalized by the Role Model Learning model [8]. Here agents select \( K \) agents from the previous generation where the probability that an agent is selected is again their fitness divided by the total fitness. These \( K \) agents represent adult role models that the new agent is learning from. The new agent then takes \( k \) samples for each meaning from each of its role models and adds 1’s to the corresponding entries in
its $A$ matrix. When $K = 1$, this model reduces to the parental learning model. Figure 4 shows 20 runs of the model were each agent learns from 4 role-models.

![Figure 4](image)

**Figure 4:**

### 3.3 Random Learning

The final model, the Random Learning model, assumes children learn language from random adults [8]. Thus, for each new agent, $K$ role models are chosen randomly from the previous generation and $k$ samples are taken from each to form the new $A$ matrix. Figure 5 shows 20 runs of the model with $m = 5$, and $w = 6$. Note that the lexicons stabilize but generally have lower fitness than in the previous learning models.

The only stable states for the Random Learning model are states where the entire population uses the same lexicon with a single “1” in each row (one word for each meaning). These states are stable since with a single “1” in each row, offspring will be identical to their parents or role models, and since all the parents have the same lexicons, all the offspring will have the same lexicons. However, so long as there are non-equal lexicons, the distribution of
these lexicons will change randomly over time and, so long as there is a lexicon with multiple non-zero entries in a row, there is a chance for the offspring to have different lexicons from their parent. Thus, like the other learning models, populations come to adopt the same language. However, unlike the other learning models, there is no incentive to make higher-payoff languages. Instead, the fitness attained by random learning is more determined by the number of one-“1”-in-each-row matrices that give a certain fitness.

Assuming that \( m \leq w \), the maximum payoff of a lexicon will be \( m \) and the minimum payoff will be 1 (everybody uses the same word for everything). Let \( Z_s \) be the number of one-“1”-in-each-row matrices with dimensions \( m \times w \) that yield a payoff of \( s \). \( Z_m = \frac{w!}{(w-m)!} \) since there are \( w \) choices of columns to place the 1st “1”, \( w-1 \) to place the second, and so on. To compute \( Z_{m-k} \) for some \( 1 \leq k \leq m-1 \), we must consider the partitions of \( k \). To produce a suboptimal lexicon, \( k \) “1’s” must not have columns of their own but must share columns with other ones. Then we must count the number of ways these \( k \) “1’s” can be distributed amongst the columns. This is equivalent to the number of partitions \( t_1, t_2, \ldots, t_w \) such that:

\[
t_i \geq 0,
\]

(1)
\[ t_1 + t_2 + \cdots + t_w = k, \quad t_i \leq m - 1, \quad k + a \leq m \]  

where \( a \) is the number of \( t_i \) with \( t_i > 0 \). The last two inequalities are to avoid too many “1’s” in a single column, for instance having a column with six “1’s” in it in a matrix with only five rows. This leads to the formula [8]:

\[
Z_{m-k} = \sum_{t_1, t_2, \ldots, t_w} \left( \begin{array}{c} m \\ t_1 + 1 \end{array} \right) \left( m - (t_1 + 1) \right) \cdots \left( m - \sum_{i=1}^{w-1} (t_i + 1) \right) \frac{(w-a)!}{(w-m+k)!} 
\]

\[
= \frac{m!}{(w-m+k)!} \sum_{t_1, t_2, \ldots, t_w} \frac{(w-a)!}{(t_1 + 1)! (t_2 + 1)! \cdots (t_w + 1)! (m-k-a)!} 
\]

Here we are taking the sum over all partitions meeting restrictions (1) through (4), excluding the \( t_i \) that equal 0. \( \left( \begin{array}{c} m \\ t_1 + 1 \end{array} \right) \) is the number of ways to distribute \( t_1 + 1 \) “1’s” in a column, \( \left( \begin{array}{c} m - (t_1 + 1) \\ t_2 + 1 \end{array} \right) \) is the number of ways to distribute \( t_2 + 1 \) “1’s” in the next column, given that \( t_1 + 1 \) rows already have “1’s” in them, and so on. \( \frac{(w-a)!}{(w-m+k)!} \) represents the number of ways to add in the remaining “1’s” that have entire columns to themselves. The following table gives the stable fitness level achieved by 400 runs of the random learning model, all with five agents and five meanings, the first 100 with \( w = 5 \), the next with \( w = 6 \), \( w = 8 \), and \( w = 10 \).

<table>
<thead>
<tr>
<th>Fitness</th>
<th>( w=5 )</th>
<th>( w=6 )</th>
<th>( w=8 )</th>
<th>( w=10 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>28</td>
<td>33</td>
<td>23</td>
<td>24</td>
</tr>
<tr>
<td>3</td>
<td>54</td>
<td>46</td>
<td>54</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>18</td>
<td>19</td>
<td>22</td>
<td>22</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

The next table shows the percentage of lexicons with certain fitnesses as calculated with the formula for \( Z_{m-k} \) where \( m = 5 \) and \( w \in \{5, 6, 8, 10\} \).

<table>
<thead>
<tr>
<th>Fitness</th>
<th>( w=5 )</th>
<th>( w=6 )</th>
<th>( w=8 )</th>
<th>( w=10 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.16%</td>
<td>.077%</td>
<td>.024%</td>
<td>.01%</td>
</tr>
<tr>
<td>2</td>
<td>9.6%</td>
<td>5.79%</td>
<td>2.56%</td>
<td>1.33%</td>
</tr>
<tr>
<td>3</td>
<td>48%</td>
<td>38.55%</td>
<td>25.63%</td>
<td>18.0%</td>
</tr>
<tr>
<td>4</td>
<td>38.4%</td>
<td>46.3%</td>
<td>51.27%</td>
<td>50.41%</td>
</tr>
<tr>
<td>5</td>
<td>3.84%</td>
<td>9.26%</td>
<td>20.51%</td>
<td>30.25%</td>
</tr>
</tbody>
</table>
The first table shows a slight shift towards lexicons of higher fitnesses as $w$ increases, which is likely due to the increased number of lexicons with high fitness at higher values of $w$, as shown in the second table.

3.4 Error

Just as mutations are included in biological interpretations of evolutionary game theory, error can be added to models of the evolution of word-meaning associations. Nowak, et al [8] add the possibility of learning error into their models. When sampling words from a role model’s speaking matrix there is a probability $p$ that the offspring agent will record a 1 in a randomly chosen column. Figure 6 shows 20 runs of the program with a probability of error of .001. Note that some of the runs that have stabilized sometimes deviate above or below their normal fitness level. This can allow runs to shift from one fitness level to another by adding or removing homonyms.

![Figure 6:](image)
4 2-D Nowak et al

4.1 Di Chio

Cecilia Di Chio and Paolo Di Chio have extended Nowak’s model by adding a notion of distance [4]. The agents are placed in cells on a grid on a torus. A torus was probably chosen over a simple rectangular grid to avoid agents being adjacent to a boundary. To simulate the fact that people living closer together, other things being equal, will talk to each other more than those living further away, the fitness gained from the communication between two agents, \( a \) and \( b \), is multiplied by

\[
\rho(a, b) = e^{-d(a,b)}
\]

where \( d(a, b) \) is the distance between \( a \) and \( b \).

The number of offspring for an agent is determined by its fitness relative to the fitness of the agents around it. These agents can then be placed in cells adjacent to the parent, or in the same cell as the parent. In the adjacent cell case, the agents form clusters over time in which the same lexicon is used. In the same cell case, agents pile into fewer and fewer cells in which the same lexicon is used as isolated cells are abandoned.

4.2 Altered Di Chio

I have made an alteration of the Di Chio model in which new agents can not move away from the old agents. This would represent a situation in which people are distributed by geography and populations can not shift based on linguistic similarity. Each agent has one offspring that replaces it in its cell. The new agent then learns by role-model learning and so may not learn anything from its parent. Fitness is computed the same way as in Di Chio’s model (with \( \rho(a, b) \)), and \( \rho \) also modifies the probability that an agent \( b \) will be a role-model to a new agent \( a \). The closer \( b \) is to \( a \), the more likely \( b \) will be a role-model. Figure 7 shows a plot of 100 agents on a 50 \( \times \) 50 grid at timestep 200 of the model. The shape and color of each agent indicates the word they use for meaning one. Note that the agents are clustered in groups in which they use the same word. This is likely due to the fact that neighbors have more influence on an agent than more distant agents. The cluster around (27,33) is particularly stable, possibly because it is isolated from other groups of agents.
Figure 7: 100 agents, 5 meanings, and 5 words

5 Oliphant, *et al*

Michael Oliphant and John Batali present a model in which, unlike the previous models, the agents are interpreted as animals signaling warnings about predators [9]. They are interested in the case where the signals are not innate but must be learned by each animal. Each agent has two matrices, a sending matrix $S$, and a receiving matrix $R$. $s_{ij}$ and $r_{ij}$ again represent the probabilities that word $j$ will be sent for meaning $i$ and that word $j$ will be interpreted as meaning $i$ respectively. The population sending matrix, $SA = \frac{1}{p} \sum_a S_a$ is the average of the sending matrices in the population. Similarly, the population receiving matrix is $RA = \frac{1}{p} \sum_a R_a$. Communicative
accuracy is determined similarly to fitness in the Nowak et al model:

\[ ca(S, R) = \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{w} s_{ij}r_{ij} \]

At each timestep, a member of the population is terminated and a new agent is created to replace them. The new agent develops its sending and receiving matrices based on one of several learning strategies.

The first is called the Imitate-Choose learning procedure [9]. The new agent sets up its sending matrix so that for each meaning, it will send the signal most commonly sent for that meaning in the population. That is, for each meaning \( i \), the \( j \) for which \( sa_{ij} \) is maximum is chosen and \( s_{ij} \) is set to 1. All other entries in row \( i \) are set to zero. Similarly the receiving matrix is set up so that for each word, it will interpret the word the way it is most commonly interpreted. Thus for each word \( j \), the \( i \) for which \( ra_{ij} \) is maximum is chosen and \( r_{ij} \) is set to 1. All other entries in column \( j \) are then set to zero. Figure 8 shows twenty runs of the simulation graphing \( ca(S, R) \) over time. The stabilization of the runs suggests that all of the agents end up with the same behavior even though their communication is not very efficient.

The second learning method is called the Obverter learning strategy [9]. For each meaning \( i \), the \( j \) for which \( ra_{ij} \) is maximum is selected and \( s_{ij} \) is set to 1. All other entries in row \( i \) are set to zero. For each word \( j \), the \( i \) for which \( sa_{ij} \) is maximum is selected and \( r_{ij} \) is set to 1 and all other entries in column \( j \) are set to zero. Figure 9 shows twenty runs of the simulation graphing \( ca(S, R) \) over time. Here not only do the agents adopt the same behavior, but their communication is efficient.

6 Conclusion

Due to the lack of opportunities for direct observation, except possibly for the incident in Nicaragua, models like those presented here are an important tool in understanding the origin of speech. They show how very simple and uncoordinated agents can interact to form a coordinated communication system. Possible extensions of these models include modifying them to model aspects of grammar, the formation of pidgins and creoles (where words from multiple languages are mixed together), or to model language in more structured societies. For instance, the notion of distance in the 2-D Nowak et al models could be modified to represent notions of hierarchy or social ostracism.
7 Bibliography


[7] Lenaerts, Tom; Jansen, Bart; Tuyts, Karl; and De Vylde, Bart, *The


Code for my 2-D model

restart;
Seed:=randomize():
with(linalg,randmatrix):
with(stats[random]):
np:=100:
nw:=5:
nm:=5:
nk:=4:
K:=4:
r:=1:
ntime:=200:
err:=.001:
px:=50:
py:=50:
A:=[ ]:
P:=[ ]:
Q:=[ ]:
F:=[ ]:
H:=[ ]:
M:=[ ]:
WC:=[ ]:
for p from 1 to np do
    A:=[op(A),a[p]]:
P:=[op(P),p[p]]:
Q:=[op(Q),q[p]]:
F:=[op(F),f[p]]:
H:=[op(H),h[p]]:
M:=[op(M),m[p]]:
od:
V:=matrix(nr,ntime):

pf:=proc(x,y):
p:=evalf(exp(-sqrt(x^2+y^2))*1.0):
p:
end proc:
#run loop
for ut from 1 to nr do
    print(ut):
    #placement
    x:=rand(0..nx):
y:=rand(0..ny):
    Dis:=[ ]:
    for p from 1 to np do
        Dis:=[op(Dis),[x(),y()]]:
    od:
    #make matrices
    k:=1:
    while k<(np+1) do
        E:=randmatrix(nm,nw):
        A[k]:=matrix(nm,nw):
        P[k]:=matrix(nm,nw):
        Q[k]:=matrix(nm,nw):
        M[k]:=[seq(w[i],i=l..K)]:
        u:=1:
        v:=1:
        while u<(nm+1) do
            while v<(nw+1) do
                A[k][u,v]:=E[u,v]*.01:
v:=v+1:
            od:
v:=1:
u:=u+1:
od:
i:=1:
j:=1:
while i<(nm+l) do
  while j<(nw+l) do
    if A[k][i,j]<0 then
      f:=A[k][i,j]:
      A[k][i,j]:=-f:
      fi:
    j:=j+1:
  od:
  j:=1:
i:=i+l:
od:
k:=k+l:
od:

#time loop
for t from 1 to ntime do
  print(t):
  wc:=[],
  for p from 1 to np do
    big:=A[p][1,1]:
    ind:=1:
    for j from 2 to nw do
      if A[p][1,j]>big then
        big:=A[p][1,j]:
        ind:=j:
      fi:
    od:
    wc:=[op(wc),ind]:
  od:
  WC:=[op(WC),wc]:

#normalization
for p from 1 to np do
  for i from 1 to nm do
    rowsum:=0:
    for j from 1 to nw do
      rowsum:=rowsum+A[p][i,j]:
    od:
    for j from 1 to nw do
      if not rowsum=0 then:
        P[p][i,j]:=A[p][i,j]/rowsum:
      fi:
    od:
  od:
  for j from 1 to nw do
    colsum:=0:
    for i from 1 to nm do
      colsum:=colsum+A[p][i,j]:
    od:
    for i from 1 to nm do
      if not colsum=0 then
        Q[p][i,j]:=A[p][i,j]/colsum:
      fi:
    od:
  od:
#Fitness Calc
for p from 1 to np do
    F[p]:=0:
    for p2 from 1 to np do
        if not p=p2 then
            f:=0:
            for i from 1 to nm do
                for j from 1 to nl-1 do
                    f:=f+(P[p][i,j]*Q[p2][i,j]+P[p2][i,j]*Q[p][i,j])*0.5:
                od:
            od:
            F[p] :=F[p]+f*pf(abs(Dis[p][1]-Dis[p2][1]),abs(Dis[p][2]-Dis[p2][2]))*1000.0:
        fi:
    od:
    F[p]:=F[p]/(np-1):
od:

#comparing fitness
V[ut,t]:=0:
for p from 1 to np do
    V[ut,t]:=V[ut,t]+F[p]:
od:
V[ut,t]:=V[ut,t]/np:
#choosing role-models
for p from 1 to np do
    sumN:=0:
    for p2 from 1 to np do
        sumN:=sumN+F[p2]*pf(abs(Dis[p][1]-Dis[p2][1]),abs(Dis[p][2]-Dis[p2][2])):
    od:
    run:=0:
    for p2 from 1 to np do
        H[p2]:=run+pf(abs(Dis[p][1]-Dis[p2][1]),abs(Dis[p][2]-Dis[p2][2]))*F[p2]/sumN:
        run:=H[p2]:
    od:
    xrand:=uniform[0,1](K):
    for i from 1 to K do
        tr:=0:
        k:=1:
        while tr=0 do
            if xrand[i]<=H[k] then
                tr:=1:
                M[p][i]:=k:
            else
                k:=k+1:
                fi:
            od:
        od:
    od:
#preparing P
for p from 1 to np do
    for i from 1 to nm do
        run2:=0:
        for j from 1 to nw do
            P[p][i,j]:=run2+P[p][i,j]:
            run2:=P[p][i,j]:
        od:
    od:
#clearing A
for p from 1 to np do
    for i from 1 to nm do
        for j from 1 to nw do
            A[p][i,j]:=0:
        od:
    od:
#reproduction
for p from 1 to np do
    for i from 1 to nm do
        for d from 1 to K do
            yrand:=uniform[0,1](nk+1):
            for k from 1 to nk do
                tr:=0:
                j:=1:
                r:=rand():
                if err<=r/(10.0^12) then
                    while tr=0 do
                        md:=H[p][d]:
                        if yrand[k]<=P[md][i,j] then
                            tr:=1:
                            A[p][i,j]:=A[p][i,j]+1:
                        else:
                            j:=j+1:
                            fi:
                        od:
                    else:
                        r2:=rand(1..nw):
                        b:=r2():
                        A[p][i,b]:=A[p][i,b]+1:
                    fi:
                od:
            od:
        od:
    od:
print("done"):  
> with(plots,textplot,display):  
for j from 1 to nr do  
points[j]:=[seq([i, V[j,i]], i = 1..ntime)]:  
od:
plot1:=plot({points[1],points[2],points[3],points[4],points[5],points[6],points[7],points[8],points[9],points[10],points[11],points[12],points[13],points[14],points[15],points[16],points[17],points[18],points[19],points[20]},style=LINE):  
display(plot1,view=[1..ntime,0..45]);  
> with(plots):  
ts:=1:  
for p from 1 to np do
    if WC[ts][p]=1 then
        plo[p]:=plot([Dis[p]],style=point,color=blue,symbolsize=15, symbol=cross):
    elif WC[ts][p]=2 then
        plo[p]:=plot([Dis[p]],style=point,color=green,symbolsize=15, symbol=circle):
elif \( WC[p] = 3 \) then
  \( plo[p] := \text{plot}([[\text{Dis}[p]]], \text{style} = \text{point}, \text{color} = \text{red}, \text{symbolsize} = 15, \text{symbol} = \text{box}) \):
elif \( WC[p] = 4 \) then
  \( plo[p] := \text{plot}([[\text{Dis}[p]]], \text{style} = \text{point}, \text{color} = \text{black}, \text{symbolsize} = 15, \text{symbol} = \text{diamond}) \):
else
  \( plo[p] := \text{plot}([[\text{Dis}[p]]], \text{style} = \text{point}, \text{color} = \text{brown}, \text{symbolsize} = 15, \text{symbol} = \text{circle}) \):
fi:
oindent od:
display({\( plo[1], plo[2], plo[3], plo[4], plo[5], plo[6], plo[7], plo[8], plo[9], plo[10], plo[11], plo[12], plo[13], plo[14], plo[15], plo[16], plo[17], plo[18], plo[19], plo[20], plo[21], plo[22], plo[23], plo[24], plo[25], plo[26], plo[27], plo[28], plo[29], plo[30], plo[31], plo[32], plo[33], plo[34], plo[35], plo[36], plo[37], plo[38], plo[39], plo[40], plo[41], plo[42], plo[43], plo[44], plo[45], plo[46], plo[47], plo[48], plo[49], plo[50], plo[51], plo[52], plo[53], plo[54], plo[55], plo[56], plo[57], plo[58], plo[59], plo[60], plo[61], plo[62], plo[63], plo[64], plo[65], plo[66], plo[67], plo[68], plo[69], plo[70], plo[71], plo[72], plo[73], plo[74], plo[75], plo[76], plo[77], plo[78], plo[79], plo[80], plo[81], plo[82], plo[83], plo[84], plo[85], plo[86], plo[87], plo[88], plo[89], plo[90], plo[91], plo[92], plo[93], plo[94], plo[95], plo[96], plo[97], plo[98], plo[99], plo[100]}), \text{view} = [0 .. nx, 0 .. ny]);