Proceedings of the 2015 Winter Simulation Conference
L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, eds.

APPLICATION OF BAYESIAN SIMULATION FRAMEWORK IN QUANTITATIVELY MEASURING PRESENCE OF COMPETITION IN LIVING SPECIES

Sabyasachi Guharay
KC Chang

Department of Systems Engineering & Operations Research
George Mason University
4400 University Drive
Fairfax, VA 22030, USA

ABSTRACT

This article uses Bayesian simulation algorithms in a checkerboard matrix framework in order to study whether competition can be statistically detected among living species. We study an exhaustive set of binary co-occurrence matrices for habitation data. We categorize the living species into five distinct groups: (1) Mammals; (2) Plants; (3) Birds; (4) Marine Life; and (5) Reptiles. We implement the Holding-swap and Metropolis-swap simulation algorithms to statistically detect the presence of competition for habitation. We find that for ~50% of our dataset, there is statistically significant presence of competition. We observe the following ranking for percentage of dataset with significant level of competition: (1) 90% of birds show competition; (2) 50% of the dataset of reptiles show competition; (3) 40% of mammals and plants; and (4) 20% of the marine life exhibit statistically significant presence of competition. We conclude that birds value habitation more strongly than marine life.

1 INTRODUCTION

There has been considerable recent interests in quantitatively studying aspects of the classic science of evolutionary theory. One of the key tenets of this science is the idea of the "survival of the fittest" (Darwin 1979). The fundamental notion here is that "competition" among species decides which one survives and which one does not. There has been interest in using quantitative computational methods to statistically quantify the *presence or absence* of competition among habitation.

There has been some conflicting academic literature regarding the question or absence of competition for habitation. Initially, ecologists came up with a set of several "assembly rules" to model the competition for "survival of the fittest" (Diamond 1975). In this model, it was stated that competition among species was a non-random process and can be modeled deterministically. Specifically, he argued that constraints on species composition are due to resource competition as it related to breadth and the minimization of non-utilized resources. An economic analogy would be that resource consumption demand curves will match resource production supply curves. However, there has been a direct contradiction to Diamond's "assembly rules" of competition (Connor and Simberloff 1979). Here it is argued that "... every assembly rule is either a tautological consequence of the definitions employed, a trivial logical deduction from stated circumstances, or a pattern which would largely be expected were species distributed randomly...." (Connor and Simberloff 1979). In addition, “Fill algorithms” (Sanderson 2000) and meta-analysis (Gotelli and McCabe 2002) have attempted to take further steps to answer the above disagreement. Sanderson reported his conclusion as the following: “I have shown, for example,

that bird species on the islands ... are not randomly distributed.... But my work has also shown that judgments about perceived patterns must be reserved ... as Connor and Simberloff first argued." (Sanderson 2000)

However recent work by statisticians (Chen et al. 2005) have argued the presence of a bias when using "Fill algorithms". The authors argue that Sanderson's method introduces statistically biased results. Specifically they showed that using a specific case study where there are five total possibilities, Sanderson's method did not generate each configuration with a 1/5 chance (i.e. generate uniformly). Thus, they argue that since one of the key assumptions of Sanderson's method is the ability to uniformly generate tables, the statistical conclusions are questionable. The authors primary contribution is to mathematically demonstrate the use of Sequential Importance Sampling for studying 0-1 contingency tables. A notable point is that a very limited data set is studied in these aforementioned articles.

The goal of this paper is to perform an exhaustive study of species habitation (across different types of mutually exclusive living species) and statistically determine in an unbiased manner the presence/absence of competition. We implement both the Metropolis-swap and Holding algorithm as Bayesian simulation methods to test for the competition. We hope to shed light on the debate in the ecological community as to whether living species distribute themselves randomly or whether there exists competition which naturally selects species' habitat.

2 BRIEF ECOLOGICAL BACKGROUND AND DATASETS

To study the presence of competition for habitation among species, ecologists collect binary co-occurrence matrices. We show an example of a co- occurrence matrix (Connor and Simberloff 1979) for the distribution of finches in the Galapagos islands in Table 1 below.

Table 1: Sample binary co-occurrence matrix of the 13 species of finches (shown in the rows) which reside in the 17 islands of Galapagos.

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
Species 1	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
Species 2	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	0	0
Species 3	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0
Species 4	0	0	1	1	1	0	0	1	0	1	0	1	1	0	1	1	1
Species 5	1	1	1	0	1	1	1	1	1	1	0	1	0	1	1	0	0
Species 6	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0
Species 7	0	0	1	1	1	1	1	1	1	0	0	1	0	1	1	0	0
Species 8	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Species 9	0	0	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0
Species 10	0	0	1	1	1	1	1	1	1	1	0	1	0	1	1	0	0
Species 11	0	0	1	1	1	0	1	1	0	1	0	0	0	0	0	0	0
Species 12	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Species 13	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

In Table 1, the value of "0" in the (i, j) element corresponds to the absence of the i^{th} species on the j^{th} island while a "1" corresponds to the presence of that particular species on the specific island. The ecological question of interest is the following: Given the knowledge of the binary co-occurrence matrices, how can one detect the presence or absence of competition for habitation?

From the binary co-occurrence matrix such as that of Table 1, ecologists define several notions of competition. The first is the idea of a checkerboard unit (Diamond 1975). An island pair and a species pair form a checkerboard unit if each species appears exactly on only one of the two islands and each of

the islands has the presence of only one of the two species. In mathematical terms corresponding to the co-occurrence matrices, the checkerboard unit C_u, can be written as either of the following 2 × 2 matrices:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \text{ and } \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

where the rows represent the species and the columns the geographical location.

An alternative approach for testing habitat competition is the use of the test statistic, $\bar{S}^2$, for testing the degree of competition (Roberts and Stone 1990):

$$\bar{S}^2 = \frac{1}{m(m-1)} \sum_{i \neq j} s_{ij}^2 \quad (1)$$

where m is the number of species, $S = AA^T$ and $A = (a_{ij})$ is co-occurrence matrix (as shown in Table 1). The higher the value of $\bar{S}^2$, the stronger the evidence of intra-specific competition for habitation.

In the ecological community there is general concession that one can use either the checkerboard units or the $\bar{S}^2$ test statistic as a measure of competition. One can now formulate the problem as a statistical hypothesis testing problem. The null hypothesis, H_0, is that the distribution of the particular living species on the islands of habitation is a matter of chance rather than any specific force, namely competition. The chance expectations are defined based on two assumptions (Connor and Simberloff 1979). The first assumption is for every island there is a fixed number of species (this is determined by the column sum for the co-occurrence matrix). The second is that there is a fixed number of observations for each species (this is determined by the row sum) for every island.

Thus, the alternative hypothesis, H_A, is that the species do not distribute themselves randomly, but rather due to competitive forces. In otherwords, statistically speaking the null hypothesis refers to the fact that the observed 0-1 table is a "representative sample" drawn uniformly from the set of all possible tables with the observed row and column marginal sums. Therefore, the ecological problem can now be formulated as a statistical problem (Chen and Cobb 2003): Given the knowledge of a binary co-occurrence matrix, what is the probability that a matrix randomly generated with the same row and column marginal sum as the original matrix will show a level of competition at least as high as the original co-occurrence matrix? For example, for the case shown in Table 1, there are 333 checkerboard units in all out of $\binom{13}{2}\binom{17}{2}$ = 10,608 submatrices. So in principle to answer the above, one can list all matrices with the correct marginal totals of equal weight, and find the proportion that have at least the same number of checkerboards as found in the ecological dataset. For the classical Darwin's finches example in Table 1, it has been estimated that there are approximately more than 6.71×10^{16} matrices with the same marginal totals. Thus this becomes a problem of *computational* feasibility since many other co-occurrence matrices are even larger (e.g. typically of the magnitude of 30 x 28, the Darwin's finches is a well-studied table and is thus shown as an example).

We use the approach which involves uniformly generating random matrices. Here we employ the law of large numbers to estimate the probability, $\hat{p}$ as the following (this can also be thought of as the statistical p-value):

$$\hat{p} = \frac{\text{Total Number of Random Matrices exhibiting competition}}{\text{Total Number of Random Matrices generated}} \quad (2)$$

We obtain an exhaustive population (as available) of non-overlapping binary co-occurrence matrices for various specimens. We divide the dataset into five distinct categories: (1) Plants; (2) Birds; (3) Reptiles; (4) Marine Life; and (5) Mammals (not including Birds and Marine Life).

3 SIMULATION METHDOLOGY

We study this problem using both the Holding and Metropolis-swap algorithms from Bayesian simulation. We use the notion of checkerboard units as the measure of competition to calculate the $\hat{p}$ as defined in equation (2). The detailed statistical theory can be found by the previous work (Chen et al. 2005). We summarize the main aspects below.

3.1 Holding Algorithm

The Holding algorithm is derived from the basic "swap" algorithm. To motivate the Holding algorithm, we start by describing the "swap" algorithm for ecological binary co-occurrence matrices.

Checkerboard unit Swap Algorithm:

I. Choose two rows and two columns uniformly at random without replacement.
II. Is the resulting 2x2 matrix a checkerboard unit?
   A. If yes, swap the resulting 2x2 matrix with its counterpart checkerboard unit (i.e. if the 2x2 matrix is the identity, change it to the other checkerboard unit matrix and vice versa).
   B. If no, go to step (I) and do not count this as a step of iteration.

The idea is to iterate the above algorithm for a large number of steps to obtain a "random" matrix. The first question to naturally ask is why is the swap algorithm an instance of MCMC? The simplest way to answer this question is to illustrate it with an example.

Let the matrix α represent a specific binary co-occurrence matrix with the pre-defined row sums of (3, 2, 1) and column sums of (2, 1, 1, 1) (Chen and Cobb 2003). Let *A(r, c)* be the set of all co-occurrence matrices with the aforementioned prescribed row sums r and column sums c. The generic representation of α is shown in Table 2 below.

Table 2: Co-occurrence matrix with row sums (3,1,1) and column sums of (1,1,1,2).

*	*	*	*	3
*	*	*	*	1
*	*	*	*	1
1	1	1	2	

For any given matrix α, there are six other co-occurrence matrices with the *same* row and column sum. All the possible cases are shown in Table 3 below.

Table 3: A(r, c) for the prescribed row and column sums as shown in matrix α from Table 2.

$$a=\begin{bmatrix}1&1&1&0\\0&0&0&1\\0&0&0&1\end{bmatrix}\quad b=\begin{bmatrix}1&1&0&1\\0&0&0&1\\0&0&1&0\end{bmatrix}\quad c=\begin{bmatrix}1&0&1&1\\0&0&0&1\\0&1&0&0\end{bmatrix}\quad d=\begin{bmatrix}0&1&1&1\\1&0&0&0\\0&0&0&1\end{bmatrix}$$

$$e=\begin{bmatrix}1&1&0&1\\0&0&1&0\\0&0&0&1\end{bmatrix}\quad f=\begin{bmatrix}1&0&1&1\\0&1&0&0\\0&0&0&1\end{bmatrix}\quad g=\begin{bmatrix}0&1&1&1\\0&0&0&1\\1&0&0&0\end{bmatrix}$$

So, we begin with matrix a (as shown in Table 3). We illustrate one sample step of the swap algorithm below in Figure 1. The idea is to iterate many steps like the one shown in Figure 1. Notice that

the matrix a_2 (shown in Figure 1) is the same as matrix *e* in Table 3. To show how this process can be represented as a Markov chain, we form a graphical representation for all possible cases when one starts from the matrix a. The set *A(r, c)* and all the possible 2x2 checkerboard units form a graph. The vertex set of the graph is represented by the matrices of *A(r, c)* and the edge set is represented by the checkerboard unit swaps as shown in Figure 2.

From Figure 2, it is clear that starting from matrix *a*, one can go directly to the following set of matrices: {*b, c, d, e, f , g*}. However, if one starts from matrix *b*, the only possible choices for the next matrix are the following: {*a, c, e, g*}. The random walk of the state X_t is defined as the 2x2 checkerboard unit swaps in the swap algorithm. When one is in state X_t, the transition probability of going from $X_t \rightarrow X_{t+1}$ is solely dependent on X_t and *not* on any previous time. Therefore, we have the first condition of a Markov chain, i.e. the property that at state t, the probability of moving to state t+1 does not dependent on any of the past history except that of state t.

Next in Figure 3, we show both the adjacency matrix and the corresponding transition matrix. The adjacency matrix is defined as $a_{ij} = 1$ if and only if there is an edge from vertex i to vertex j. If we normalize the adjacency matrix by dividing each element of row i, by the row sum, we can obtain a transition matrix. Thus, we now have a Markov chain representation for the 2x2 checkerboard unit swap algorithm.

$$a_1 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$ **1)** randomly choose *say* rows (1,2) and columns (3,4)

↓

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \rightarrow$$ **2)** Is this a checkerboard? Yes, swap it.

↓

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \rightarrow$$ **3)** Change the original matrix rows (1,2) & columns (3,4)

↓

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = a_2$$ **4)** End of 1-step of swap algorithm

Figure 1: One sample iteration of the swap algorithm.

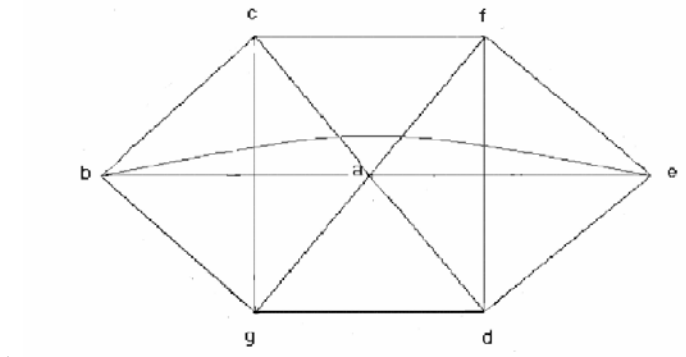

Figure 2: Graphical representation of the set.

A)

	a	*b*	*c*	*d*	*e*	*f*	*g*
a	0	1	1	1	1	1	1
b	1	0	1	0	1	0	1
c	1	1	0	0	0	1	1
d	1	0	0	0	1	1	1
e	1	1	0	1	0	1	0
f	1	0	1	1	1	0	0
g	1	1	1	1	0	0	0

B)

	a	*b*	*c*	*d*	*e*	*f*	*g*
a	0	1 / 6	1 / 6	1 / 6	1 / 6	1 / 6	1 / 6
b	1 / 4	0	1 / 4	0	1 / 4	0	1 / 4
c	1 / 4	1 / 4	0	0	0	1 / 4	1 / 4
d	1 / 4	0	0	0	1 / 4	1 / 4	1 / 4
e	1 / 4	1 / 4	0	1 / 4	0	1 / 4	0
f	1 / 4	0	1 / 4	1 / 4	1 / 4	0	0
g	1 / 4	1 / 4	1 / 4	1 / 4	0	0	0

Figure 3: (A) Adjacency matrix for the graph representation in Fig. 2; (B) Transition matrix for the graph representation in Fig. 2.

In order to test whether this algorithm will give truly "random" matrices, we have to analyze the limiting (stationary) distribution. Following the work of (Chen and Cobb 2003), we proceed with a further mathematical explanation. Suppose $\mathbf{p}^{(n)}$ be a vector of probabilities whose i^{th} element represents the chance that the chain is in state *i* after *n* steps, which gives the probability distribution over the states of the chain at time *n*. Under certain regularity conditions (Chen and Cobb 2003), $\mathbf{p}^{(n)}$ converges to a unique limiting/stationary distribution, say π (regardless of where the chain starts, i.e. $p^{(0)}$). Since we want a "random" matrix, the stationary distribution must converge to a uniform distribution. To test this, one must use the Chapman-Kolmogorov equations of $\mathbf{p}^{(n+1)} = \mathbf{p}^{(n)}\mathbf{T}$, where **T** is the transition matrix. So for the limiting distribution, we can find stationary vector such that $\pi\mathbf{P} = \pi$. Now for any graph this can be found by letting d_i be the degree of vertex *i* (where degrees of a vertex of a graph is the number of edges incident to the vertex, while loops are counted twice), and let $a_{i,j} = 1$ if $\{i, j\}$ is an edge and 0 otherwise. Then the probability of going from vertex *i* to *j*, $p_{i,j} = a_{ij}/d_i$, and also $\sum_i a_{ij} = d_j$. Now if we set $\pi_i = d_i/\sum d_i$, one can easily show that $\pi\mathbf{P} = \pi$. This shows that in the limit, the visiting rate of a vertex is directly proportional to the number of neighbors. Thus, using our example from above, the value of π is (6,4,4,4,4,4,4)/30. This illustrates the bias in the results. The following algorithm corrects this bias.

Checkerboard unit Holding Algorithm:

I. Choose two rows and two columns randomly without replacement.
II. Is the resulting 2x2 matrix a checkerboard unit?
    A. If yes, swap the resulting 2x2 matrix with its counterpart checkerboard unit.
    B. If no, go to step (I) and count this as a step of iteration.

Now we have allowed the graph to self-loop (i.e. go back to itself after one iteration). Thus with this correction, the Holding algorithm will generate a uniform limiting distribution.

3.2 Metropolis-Swap Algorithm

While the Holding algorithm is a valid MCMC method, it tends to be computationally inefficient. This is mainly due to the configuration of the ecological dataset. Since we have been using a checkerboard unit as the measure of competition, the probability of the Holding algorithm moving forward depends entirely on the number of checkerboard units presents. In the dataset as shown in Table 1, for example, only 3% of

the 2x2 sub-matrices form a checkerboard unit. This implies that it will take roughly 30 iterations (on the average) before one can make a swap (for the finch data set). In general, the percentage of checkerboard units tend to be small for most ecological habitation data sets. Therefore, it would take a large number of iterations to properly sample the entire dataset.

To eliminate some of this inefficiency, the Metropolis-swap algorithm has been proposed (Chen et al. 2005). It has been shown that swap walks on co-occurrence matrices are reversible, and thus the Metropolis algorithm can be used. The Metropolis-swap algorithm is defined as the following:

Checkerboard unit Metropolis-Swap Algorithm:

I. Choose two rows and two columns randomly without replacement.
II. Is the resulting 2x2 matrix a checkerboard unit?
    A. If yes, swap the resulting 2x2 matrix with its counterpart checkerboard unit.
    B. If no, go to step (I) and count this as a step of iteration.
III. Denote Λ_j as the sub-matrix obtained by swapping matrix Λ_i with its other checkerboard unit counterpart. Let δ_j and δ_i represent the number of checkerboard units in sub-matrix Λ_j and Λ_i, respectively. Define $\beta = \min(1, \delta_i/\delta_j)$ and generate a uniform random number u in [0, 1].
    A. If $u \leq \beta$, then accept; (this moves from matrix $\Lambda_i \rightarrow \Lambda_j$)
    B. Else reject; (remains at Λ_i)

The main advantage in using the Metropolis-swap algorithm is that it takes into account the *reversibility* property of Markov Chains. In other words, it allows the case of going from $\Lambda_i \rightarrow \Lambda_j$ as well as $\Lambda_j \rightarrow \Lambda_i$. Since it has been shown that swap walks on co-occurrence matrices are reversible (Chen et al. 2005), it is a major computational benefit to take advantage of the reversibility property of the Metropolis algorithm. We implement both algorithms for our dataset to calculate the value of $\hat{p}$.

4 RESULTS AND DISCUSSIONS

We begin by showing some Bayesian Simulation diagnostics before proceeding to the ecological data set results.

4.1 Simulation Diagnostics

Since we are dealing with simulations of the MCMC type, it is important to perform several diagnostics tests before running on real-life data (ecological data). We first start with generating a known example where the truth is known regarding whether there exists competition or not. We randomly choose a 4x4 binary co-occurrence matrix, X, as given below:

$$\begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

It can be easily shown that there are a total of 27 matrices with the same row sum of (2,2,2,3) and column sum of (3,2,3,1) as the above matrix X. For all of these matrices, we computed the number of checkerboard units in each of the other twenty-six cases besides the original matrix X. For the original matrix X as shown above, there were 8 checkerboard units. The $\hat{p}$ (p-value) is determined as the proportion of matrices which has at least eight checkerboard units. The actual p-value in this case equals to 21/27 or approximately 77.8%.

Both Holding and the Metropolis-swap algorithms are tested for several different iteration values using matrix X as our initial state and the results are shown in Table 4 below.

Table 4: $\hat{p}$ values from running the Holding and Metropolis-swap algorithms for several different iteration values using as input X; note that the true p-value here is 21/27.

Iteration #	10	100	1000	10000	100000
Holding	0.924 ± 0.17	0.760 ± 0.13	0.782 ± 0.04	0.771 ± 0.02	0.779 ± 0.002
Metropolis	0.895 ± 0.24	0.776 ± 0.12	0.783 ± 0.05	0.772 ± 0.01	0.776 ± 0.006

As shown in the table above, both the Holding and the Metropolis-swap algorithms converge rapidly after one hundred iterations to near the true p-value. To extend the test, we apply the Holding and the Metropolis to the finch dataset given in Table 1 and results are show in Table 5.

Table 5: $\hat{p}$ values from running the Holding and Metropolis-swap algorithms for several different iteration values using finch data set in Table 1; note that the true p-value here is ≈ 0.01.

Iteration #	10	100	1000	10000	100000
Holding	0.86 ± 0.31	0.33 ± 0.29	0.038 ± 0.03	0.003 ± 0.003	0.005 ± 0.005
Metropolis	0.83 ± 0.29	0.38 ± 0.32	0.03 ± 0.03	0.003 ± 0.005	0.004 ± 0.003

Note that the finch data has been extensively studied (Chen and Cobb 2003, Chen et al. 2005) and the authors tend to agree that the p-value should be around 0.005. From the results in Tables 4-5, it is clear that after around 10,000 iterations, both algorithms are converging to close to the "correct" p-value.

4.2 Simulation Burn-in Estimates

In the above cases, we did not analyze the “burn-in” period for the MCMC simulations (both Holding and the Metropolis-swap). There are several ways to estimate the “burn-in” times (Brooks et al. 2011) and to provide a detailed mathematical analysis on estimating the convergence rates based on eigenvalue analysis of the transition matrix (Chen and Cobb 2003). Since it is impractical to compute the transition matrix for large ecological datasets, the analysis presented before has far more theoretical implications (Chen and Cobb 2003) than its practicality. There is statistical literature on ways to determine a diagnostic program to estimate the number of iterations needed based on a pilot sample run of an MCMC simulation (Raftery and Lewis 1992). This methodology assumes a Bayesian framework, while the methodology used here is primarily that of a frequentist. In this model, we have little or no information about the “prior” distribution. Thus, since the Raftery-Lewis diagnostic is derived from a Bayesian framework and is also an estimate, we do not solely rely on this to estimate the burn-in period.

We simulated multiple pilot runs for both algorithms as shown next.

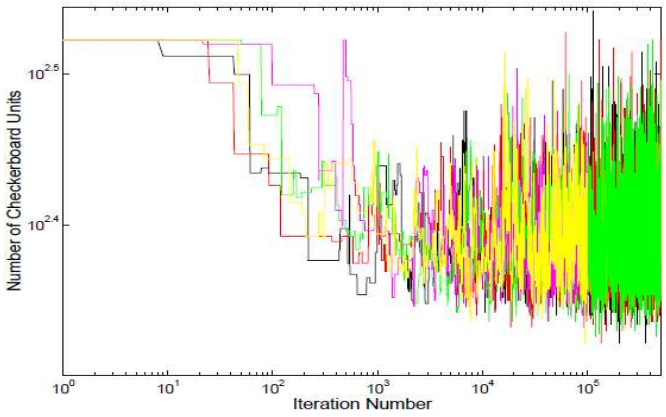

Figure 4: 5 pilot runs each consisting of 5 million iterations of Holding algorithm using the finch data set in Table 1 as a sample; notice after approximately 7,000 iterations, the MCMC simulations "stabilize".

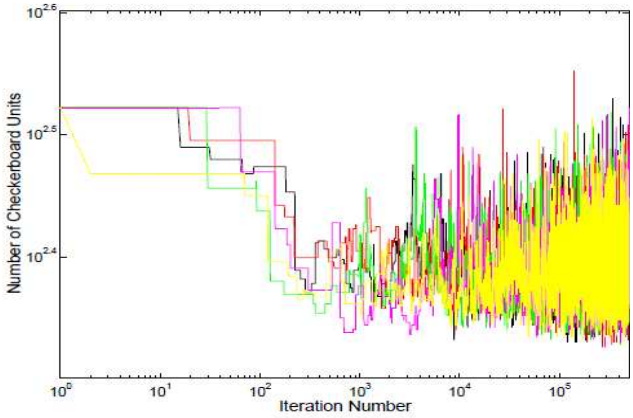

Figure 5: 5 pilot runs each consisting of 5 million iterations of Metropolis-swap algorithm using the finch data set in Table 1; notice after approximately 3,000 iterations, the MCMC simulations "stabilize".

Notice in both Figures 4 and 5, all of the MCMC simulation chains converge in the long-run. We expected that the Metropolis-swap will converge faster and indeed Figure 5 shows evidence of this phenomenon. All five curves in Figure 5 seem to stabilize after approximately 3,000 iterations. In the case of the Holding algorithm, it seems that after around 7,000 iterations or so, the chain has stabilized (as shown in Fig. 4). Since these pictorial representations are an approximation, we use a conservative figure of 10,000 iterations as the "burn-in" period.

4.3 Mixing of the MCMC Simulations

Besides the burn in period, the other pertinent question related to MCMC methods is how well does the MCMC chain in question mix? To test this, we computed the auto-correlation function (acf) for the data after the estimated "burn-in" period.

It is very clear from Figure 6 below that both the Holding and the Metropolis-swap have a long-mixing period. The auto-correlation function decays slowly to zero for both cases. For the Holding algorithm, the acf decays to zero after roughly 3000 lags while for the Metropolis swap, the acf decays to 0 by around 2000 lags. It's interesting to note is that the Metropolis-swap mixes faster than the Holding (as expected from the theory). We computed the p-values for the finch data set when one incorporates the mixing property. In other words, we computed the p-value by sampling every 2000 iterations (for the Metropolis-swap algorithm) and by sampling every 3000 iterations for the Holding algorithm. We obtain similar $\hat{p}$ values (to the third decimal place) with respect to that of not incorporating the mixing property. This gives us confidence in that the slow-mixing of these MCMC simulations will not greatly affect the overall outcomes of the $\hat{p}$ results. Also previous work has argued that if a pilot run for the MCMC simulations (Brooks et al. 2011) is made long enough (for both Holding and Metropolis), the burn-in and the mixing effects will not affect the final results in the long-run. With this in mind, we simulate both the Holding and the Metropolis-swap on the entire population dataset using the first 10,000 iterations as the burn-in and iterating for a minimum of 100,000 simulations (higher for larger matrices). We then computed the final $\hat{p}$ values (p-values) for each ecological dataset.

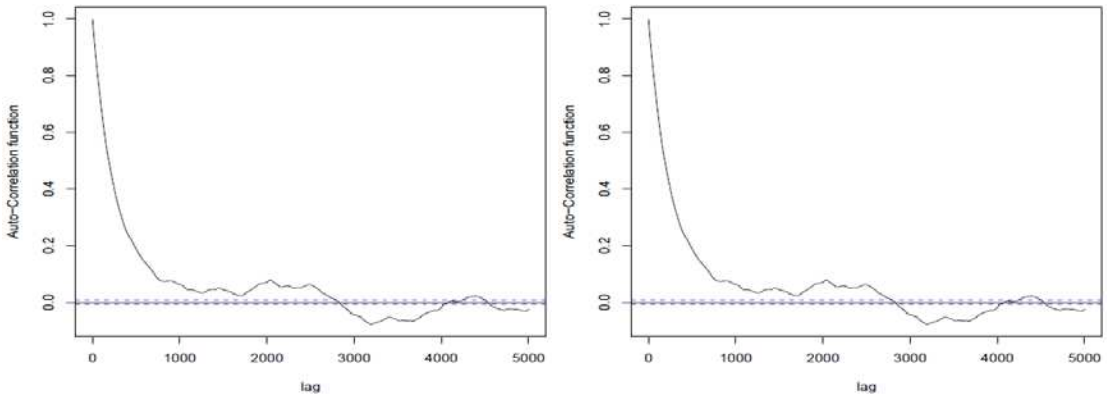

Figure 6: Left shows the Auto-Correlation function (ACF) for the Holding algorithm using the finch dataset in Table 1; Right shows the ACF for the Metropolis-swap algorithm using finch dataset in Table 1

4.4 Simulation Results for Ecological Datasets

We first show a histogram of the p-values for all the species studied below in Figure 7. It seems that the histograms of estimated p-values for both the Metropolis-swap and the Holding simulations follow an exponential distribution. Since the histograms in Figure 7 do not come from continuous data, we cannot use the Kolmogorov-Smirnov goodness of fit test to assess whether the histogram follows some type of an exponential distribution.

For the dataset we studied, we found that approximately 50% of the data has p-values less than 10% (for both the Holding and Metropolis-swap simulation analysis). Thus, based on the checkerboard unit definition of the competition, we find about half of the cases in which we can safely reject the null hypothesis that the species distribute themselves randomly. However, since we have about half of the cases where we fail to detect competition, we break down the cases based on the category in which the species is classified (i.e. marine life, or plant, etc.).

In Figure 8 below, we show the average p-values for each group. This plot was generated using the Holding simulation output. The results are the same (they differ in the third decimal place of the p-value) for the Metropolis-swap algorithm. It is interesting to note that birds show the strongest sign of competition, while mammals show the least sign. Also, notice in Figure 9 that again birds dominate in the percentage of statistically significant p-values (p-value less than 10%). One can hypothesize from these results that birds value their habitation highly and are willing to "fight" for it more than other species.

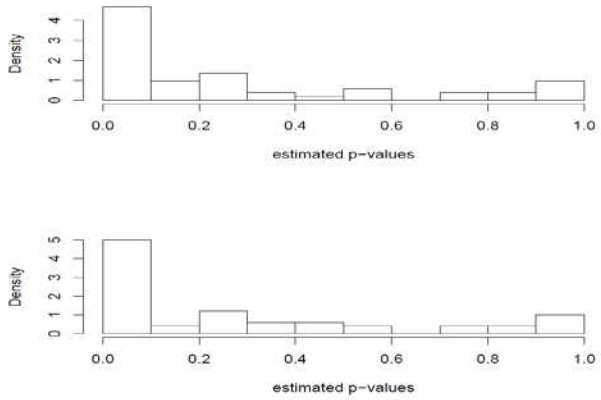

Figure 7: Histogram of the p-values for all of the species studied; top figure shows for the Holding simulations and the bottom figure is for the Metropolis-swap simulations.

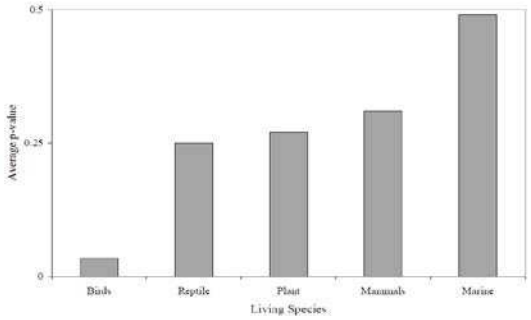

Figure 8: Plot of the average p-values for all of the species classified in each group.

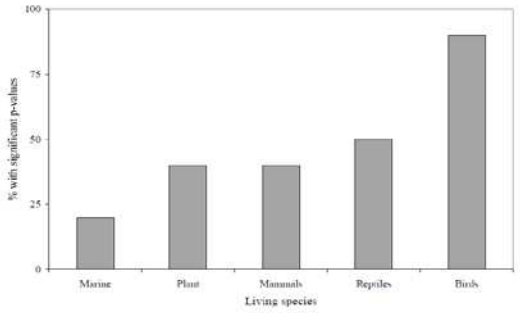

Figure 9: Plot of the percentage of statistically significant p-values ($p \leq 10\%$) for each group of species.

5 CONCLUSIONS & FUTURE WORK

In this article, we implement two instances of MCMC simulation applications to a problem in ecology. We test the "burn-in" times along with the mixing rate through graphical and auto-correlation function analysis, respectively. Through an exhaustive study, we have found evidence for competition among 50% of the living species studied. We find that among the living groups the following order of competition exists (ranked from highest level of competition to least): Birds, Reptiles, Plants, Mammals and finally Marine life. Also we have found the following ranking for percentage of dataset with significant competition (p-value $\leq$ 10%): 90% for birds, 50% for reptiles, 40% for mammals, 40% for plants and finally 20% for marine life. We argue from these results that birds value their habitation very highly and thus compete heavily amongst themselves for a "suitable" habitation. Mammals, reptiles, plants and marine life on the other hand, may compete in general, but perhaps value habitation to a far lesser extent than birds. In the case of marine life, since there is an "abundance" of space for habitation it seems reasonable that the marine life category shows a lesser degree of habitat competition than say those of the birds who always migrate and keep looking for new habitats.

The results for the Holding algorithm versus the Metropolis swap algorithm matches very precisely (up to the third decimal place). This gives further evidence in our overall results in that these two different methods yield the same ecological conclusions. We also validated both the Holding and the Metropolis-swap algorithms by running them on previously established and theoretically-known results.

For future work, we plan to develop a regression model for competition. However, we believe that a more sophisticated regression model can measure the level of competition and provide inference. Also, it is worth exploring whether one can represent the co-occurrence matrices as cooperation vs. competition framework. For this idea, we plan on implementing hybrid MCMC methods.

ACKNOWLEDGMENTS

We would like to acknowledge N. Gotelli from University of Vermont for providing the datasets. In addition, we would like to thank J. Sanderson for providing some of the background literature and survey materials which aided in our understanding of the scientific importance of this problem.

REFERENCES

Brooks, Steve, A. Gelman, G. Jones, and X.-L. Meng. 2011. *Handbook of Markov Chain Monte Carlo*. Boca Raton, Florida: Chapman & Hall.

Chen, Yuguo, Persi Diaconis, Susan P. Holmes, and Jun S. Liu. 2005. "Sequential Monte Carlo Methods for Statistical Analysis of Tables." *Journal of the American Statistical Association* 100(469): 109-120.

Cobb, George W., and Yung-Pin Chen. 2003. "An Application of Markov Chain Monte Carlo to Community Ecology." *American Mathematical Monthly* 110(4): 265-288.

Connor, Edward F., and Daniel Simberloff. 1979. "The Assembly of Species Communities: Chance or Competition?" *Ecology* 60(6): 1132-1140.

Darwin, C. 1979. *Origin of the Species*. New York: Random House.

Diamond, J. M. 1975. *Ecology and Evolution of Communities*. Cambridge, Massachusetts: Harvard University Press.

Gotelli, Nicholas J., and Declan J. McCabe. 2002. "Species Co-Occurrence: A Meta-Analysis of J.M. Diamond's Assembly Rules Model." *Ecology* 83(8): 2091-2096.

Raftery, Adrian E., and Steven Lewis. 1992. "How Many Iterations in the Gibbs Sampler." *Bayesian statistics* 4(2): 763-773.

Roberts, Alan, and Lewis Stone. 1990. "Island-Sharing by Archipelago Species." *Oecologia* 83(4): 560-567.

Sanderson, James. 2000. "Testing Ecological Patterns: A Well-Known Algorithm from Computer Science Aids the Evaluation of Species Distributions." *American Scientist* 88(4): 332-339.

AUTHOR BIOGRAPHIES

SABYASACHI GUHARAY is a Doctoral Student in Systems Engineering & Operations Research (SEOR) at George Mason University (GMU) along with being a Senior Operations Research Analyst at the US Internal Revenue Service. His research interests involve applications of simulation in financial engineering and quantitative risk management. His email address is sguhara2@masonlive.gmu.edu.

KC CHANG is a professor in SEOR at GMU. His research interests are in data fusion, Bayesian probabilistic reasoning, and their applications. His email address is kchang@gmu.edu.