Short title: FEASIBLE SETS FOR MACROECOLOGICAL PATTERNS

Article title: Efficient algorithms for sampling feasible sets of macroecological patterns

Kenneth J. Locey[‡1] and Daniel J. McGlinn[*1,2]

Department of Biology, Utah State University, Logan, UT, 84322

*Ecology Center, Utah State University, Logan, UT, 84322

‡To whom correspondence should be addressed

1 ken@weecology.org; phone: (435) 764-5070, fax (435) 797-1575

2 daniel.mcglinn@usu.edu;

Statement of authorship: KL derived the integer partitioning algorithms, coded them in Python, conducted the analyses, and served as primary author. DM coded algorithms into R and C, and served as secondary author. Both KL and DM contributed substantially to the organization of the paper, its ideas, and discussion.

Keywords: combinatorics, constraints, feasible set, integer partitions, macroecology, sampling algorithms, species abundance distribution, species spatial abundance distribution

Words in the abstract: 153

Number of cited references: 25

Number of figures: 5

Number of tables: 1

Abstract. Ecological variables such as species richness (*S*) and total abundance (*N*) can strongly influence the forms of macroecological patterns. For example, the majority of variation in the species abundance distribution (SAD) can often be explained by the majority of possible forms having the same *N* and *S*, i.e. the feasible set. The feasible set reveals how variables such as *N* and *S* determine observable variation and whether empirical patterns are exceptional to the majority of possible forms. However, this approach has currently only been applied to the SAD using relatively inefficient random sampling algorithms. We extend the use of the feasible set approach by developing new algorithms to efficiently generate random samples of the feasible set for the SAD and the intraspecific spatial abundance distribution (SSAD). These algorithms are often several orders of magnitude faster than a previous method, which greatly increases the size and diversity of communities that can be examined.

INTRODUCTION

Understanding patterns of abundance, distribution, and diversity is a primary goal in ecology (Brown 1995). These macroecological patterns reveal general characteristics of ecological systems and provide insights into the processes and phenomena that drive ecological structure. It has also been shown that general ecological variables such as total community abundance (*N*) and species richness (*S*) can greatly constrain the forms of ecological patterns (Harte et al. 2008, McGlinn and Hurlbert 2012, Supp et al. 2012, White et al. 2012, Locey & White 2013). In fact, more than 90% of observed variation in the species abundance distribution (SAD), i.e. the vector of abundances of all species in a community (McGill et al 2007), can often be explained by models constrained by *N* and *S* (Harte 2011, White et al. 2012). This is perhaps not surprising given that the majority of all possible forms of the SAD having the same *N* and *S*, i.e. the feasible set, often explain the general form of empirical SADs (Locey and White 2013).

The feasible set reveals how constraint variables such as *N* and *S* determine observable variation and whether empirical patterns are exceptional to or representative of the majority of possible forms having the same constraint values (Haegeman and Loreau 2008, Haegeman and Loreau 2009, Locey and White 2013). Currently only the SAD has been examined using the feasible set approach (but see Storch et al. 2003). Though Locey and White (2013) suggested that other macroecological patterns could also be examined in the context of their feasible sets, there are considerable computational challenges to using a feasible set approach.

Feasible sets can be immense and enumerating them can be untenable. However, small random samples can be used to characterize the center of the feasible set (i.e. average form) as well as the distribution of statistical features (e.g. evenness, diversity) within the feasible set.

Locey and White (2013) took a conceptually simple and unbiased approach to sampling the SAD feasible set, a combinatorial method known as integer partitioning. This approach is based on the fact that there are a limited number of unordered ways that *n* integers can sum to a total *q*, and hence, a limited number of ways that the abundances of *S* unlabeled species can sum to a total abundance of *N*. These unordered configurations of integers are called integer partitions (Bóna 2006). For example, the feasible set for $q = 6$ and $n = 3$ is: {(4, 1, 1), (3, 2, 1), (2, 2, 2)}, where differently ordered configurations having the same integer values (e.g. (4, 1, 1), (1, 1, 4), (1, 4, 1)) represent the same integer partition, i.e. (4, 1, 1).

Use of integer partitioning to randomly sample the feasible set allows the feasible space to be characterized without generating all possible forms. However, all published partitioning algorithms sample the feasible set only with regards to the total, *q*. This means that all partitions of *q*, regardless of the number of elements *n*, have the same probability of being drawn. For example, randomly generating one partition for $q = 1000$ and $n = 10$ requires drawing from a feasible set of nearly 2.4×10^{31} partitions, one of the roughly 8.9×10^{14} having 10 elements; a probability of nearly 3.7×10^{-17}. This means that randomly sampling the feasible set for a given *q* and *n*, requires generating partitions according to *q* and then rejecting those not having *n* elements, often resulting in impractically high rejection rates.

Another challenge in applying integer partitioning to the study of feasible sets is that some macroecological patterns include parts with zero values. One example is the species spatial abundance distribution (SSAD) describing the frequency with which individuals of a single species occupy areas within a landscape (Brown et al. 1995, Harte et al. 2008). In the SSAD, individuals can be absent from a number of areas, meaning that there are some areas with zero individuals. Because integer partitions *per se* do not include zeros, integer partitioning methods need to be modified to examine the SSAD feasible set.

Here, we present algorithms that greatly increase the efficiency of sampling the SAD and SSAD feasible sets. We explain each algorithm in concept and develop Python and R based implementations of them. We test each algorithm for sampling bias and for speed against the method of Locey and White (2013). To reveal the practical gains of these new algorithms, we reanalyze the SAD datasets of Locey and White (2013) wherein it took more than 10000 compute hours to examine only 60% of the available data (9562 of 15950 SADs). Our algorithms should allow us to examine a much larger portion of the data in less time. Finally, we examine general characteristics of the SSAD feasible set including characteristics of the frequency distribution, skewness of the SSAD, and the rank distribution of abundances. Our work expands the feasible set approach to an additional ecological pattern and to values of q and n that were previously untenable.

METHODS

Our goal is to develop fast and unbiased integer partitioning algorithms, to generate random samples of feasible sets for ecological patterns such as the SAD and SSAD that are defined by a total q and composed of n elements. Integer partitioning is a mature field of mathematics and algorithms for generating random partitions of q (e.g. Nijenhuis and Wilf 1978) are often implemented in mathematical environments (e.g. Sage, Mathematica). However, these approaches do not allow the random partitioning of q into exactly n elements. Here, we use two well-established theorems and a partitioning identity to develop a general method for generating a random partition of q having n elements. We then modify our method to allow elements having zero values. We begin by using two theorems to generate a random partition of q (see Appendix 1 for generating functions and recurrence relations).

1) For every integer q there are $p(q)$ partitions having q or less elements.
2) For every integer q there are $p_k(q)$ partitions having k or less as the first element.

For example, there are 5 partitions of 4 and each has 4 or less as the first element; i.e. $p(4) = 5$, {(4), (3, 1), (2, 2), (2, 1, 1), (1, 1, 1, 1)}. Likewise, there are 4 partitions of 4 having 3 or less as the first element; $p_3(4) = 4$, {(3, 1), (2, 2), (2, 1, 1), (1, 1, 1, 1)}. It is possible to sequentially build a random partition element by element by iteratively applying these two theorems (Fig. 1). Specifically, if we choose a random number x from 1 to $p(q)$, we can say there are at least x partitions of q having some value k or less as the first element, i.e. $x \leq p_k(q)$. Likewise, there must also be some value of $k - 1$, for which, there are less than x partitions of q having $k - 1$ or less as the first element, i.e. $p_{k-1}(q) < x$. Putting these statements together, there must be a value k for which $p_{k-1}(q) < x \leq p_k(q)$. In this way, we can find the value of the first element in one of the partitions by finding the value of k that satisfies $p_{k-1}(q) < x \leq p_k(q)$. Having found the value of the first element, we can decrease x by $p_{k-1}(q)$ and q by k, and then find the first element for this combination of smaller values. Repeating this process will sequentially build the partition until $q = 0$ and the sum of the partition is equal to the original value of q (Fig 1).

The above approach is similar to well-established methods of generating random partitions of q (e.g. Nijenhuis and Wilf 1978, Stojmenovic 2008). However, our goal is to generate random partitions of q having n elements. For this, we use a well-known integer partitioning identity to restrict the number of elements in a randomly chosen partition to n. Specifically, the number of partitions of q having n elements equals the number of partitions of q having n as the first element (Bóna 2006). This is because each partition of q having n

elements corresponds to one unique partition of *q* having *n* as the first element (Bóna 2006). For example, consider the partition (3, 1), which can be illustrated with rows of dots, called a Ferrer's diagram (Fig 1). In the Ferrer's diagram for (3, 1) there are two rows, the largest having three dots. Flipping the diagram on its diagonal produces its conjugate (2, 1, 1), which has three rows, the largest row having two dots. So, the conjugate of (3, 1) is (2, 1, 1) and vice versa. Consequently, the first part of an integer partition determines the number of parts in its conjugate (Bóna 2006). This allows us to extend the problem of generating random partitions of *q* to random partitions of *q* having exactly *n* elements. That is, knowing the first element must be *n* so that its conjugate has *n* elements, we can decrease *q* by *n* and then generate a random partition for this decreased value of *q* having *n* or less as the first element. Once the partition is generated, we append *n* to the beginning of the partition and conjugate it to produce a random partition of the original *q* having exactly *n* elements (Fig 1).

The approach outlined above begins with a randomly chosen number *x* between 1 and $p(q)$.. It then finds the value of *k* that satisfies the inequality $p_{k-1}(q) < x \leq p_k(q)$, that value of *k* is the value of the first element of the partition. However, the question remains as to which value of *k* to start with and how to proceed to different values. Indeed, we could start with the smallest possible value of *k* ($k = 1$) and take a 'bottom-up' approach, or the largest possible value ($k = p(q)$) and take a 'top-down' approach, or even choose *k* at random and use a 'divide-and-conquer' method. These approaches differ only in how *k* is chosen and each builds the partition one element at a time. However, in the event that *q* is much larger than *n*, e.g. all trees in the 50 ha Barro-Colorado Island mapped forest plot where $q \approx 200{,}000$ individuals and $n \approx 300$ species, the three algorithms above will still be inefficient. This is because they would first generate a partition of, say, 200000 having 300 as the first element, but having as many as 199701 elements, and then conjugate it to produce a partition of 200000 having 300 elements. Clearly building the partition one element at a time would be inefficient in this case.

An alternative approach is to build a partition using multiples of integers – the 'multiplicity' approach. Instead of finding the value of k corresponding to the first element, appending it and moving on, we can instead ask how many times must k occur, i.e. the partitions having some multiple m of k. We can start with the smallest possible multiple (i.e. $m = 1$) and ask whether x is less than or equal to the number of partitions of $q - k*m$ having less than k as the first part. This is because the set of partitions of q having a number of k's equal to m actually contains the set of partitions of $q - k*m$ having less than k as the first part (Appendix 1). We can increase m by one until $x \leq p_k(q - k*m)$, at which point we will have found the corresponding multiple of k. One drawback to this method is the overhead due to the extra computation.

Random partitions for q and n, with some parts having zero values

The above algorithms address patterns having positive values, e.g. SAD. In contrast, some macroecological patterns include zero values (e.g. absences). One example is the species spatial abundance distribution (SSAD), a frequency distribution that characterizes the number of quadrats, cells, or areas containing a given abundance of a species (Brown et al. 1995). However, only small changes are needed to adapt the above approaches to cases allowing zero-valued parts. For example, let 10 unlabeled individuals occupy a landscape sectioned into quarters. The most aggregated distribution would be for all 10 to occupy the same quarter, [10, 0, 0, 0]. The least aggregated would be for 3 to occupy two quarters while 2 occupy the other two quarters, [3, 3, 2, 2]. In fact, the number of configurations for 10 unlabeled individuals distributed across 4 unlabeled sections equals the number of partitions of 10 having 4 or less parts, i.e. $p_4(10 + 4) =$

23. Consequently, if $n \leq q$, a random partition for q and n allowing for zero-valued parts, is simply a random partition for q having n or less parts, with zeros appended to ensure the final form of the partition has n parts.

On the other hand if $n > q$ a different approach is needed. To see this let 4 unlabeled individuals occupy a landscape sectioned into tenths. The most aggregated distribution would be for all 4 to occupy the same subsection, [4, 0, 0, 0, 0, 0, 0, 0, 0, 0] and the least aggregated configuration would be for 4 sections to have one individual and for 6 sections to have zero, i.e. [1, 1, 1, 1, 0, 0, 0, 0, 0, 0]. In this way, the number of possible configurations for 4 unlabeled individuals distributed across 10 unlabeled sections is $p(q)$. Consequently, if $q < n$, a random partition for q and n, allowing for zero-valued parts, is simply a random partition for q having q or less parts, with zeros appended to ensure the partition has n parts.

Examining for bias and speed

We implemented the above algorithms in Python and R and made them freely available using a public Github repository (https://github.com/klocey/partitions). We are currently developing these algorithms into a Python module to be distributed on the Python Package Index (PyPI; https://pypi.python.org/pypi) and an R package to be distributed on The Comprehensive R Archive Network (CRAN; http://cran.us.r-project.org/). We used kernel density curves to visually compare the results of the above algorithms to full feasible sets and to random samples generated with the function implemented in Sage, which is based on the algorithm of Nijenhuis and Wilf (1978) and is the method used in Locey and White (2013). If our algorithms are unbiased, then their distributions will not differ in any systematic way from full feasible sets and random samples generated using Sage. We compare the computational speed of our algorithms to that of the approach used in Locey and White (2013) (i.e. using Sage to generate random partitions for a given *q* and rejecting those not having *n* elements) across a range of values of *q*, *n*, and *q*-*n* ratios for which the latter method was likely to return random samples within reasonable time (one hour).

Empirical Demonstration of the New Algorithms

Locey and White (2013) analyzed 9562 SADs of trees, birds, mammals, fungi, and prokaryotes using a partitioning algorithm that sampled the feasible set according to total abundance *N* but not with respect to species richness *S* (i.e. the number of elements). Those data consisted, in part, of a subset of previously compiled datasets of site-specific species abundance data (see White et al. 2012), and included four continental-to-global scale surveys, including the Christmas Bird Count (129 sites) (National Audubon Society 2002), North American Breeding Bird Survey (1,586 sites) (Sauer et al. 2011), Gentry's Forest Transect Data Set (182 sites) (Phillips and Miller 2002), Forest Inventory Analysis (7,359 sites) (U.S. Department of Agriculture 2010), and one global-scale data compilation, the Mammal Community Database (42 sites) (Thibault et al. 2011). Locey and White (2013) also compiled abundance data at the species level from five microbial metagenome projects for a total of 264 SADs. Those data were obtained from the metagenomics server MG-RAST (Meyer et al. 2008). Metagenomic data were compiled into datasets representing aquatic prokaryotic communities (48 metagenomes) (Flores et al. 2011, www.catlin.com/en/Responsibility/CatlinArcticSurvey), terrestrial prokaryotic communities (92 metagenomes) (Chu et al. 2010, Fierer et al. 2012), and terrestrial fungal communities (124 metagenomes) (Amend et al. 2010).

The inefficiency of the partitioning method used in Locey and White (2013) restricted their analyses to combinations of *N* and species richness *S*, for which, there was a reasonable probability of generating a random integer partition of *q* with exactly *n* elements (> 10^{-6}). This restriction allowed for only 60% of the available data to be examined despite more than 10000 compute hours worth of effort. We reanalyze those datasets using the algorithms developed here, which should allow for random samples of a greater number of SADs to be produced in less time.

General characteristics of the SSAD feasible set

Brown et al. (1995) revealed evidence that the general form of the SSAD, like that of the SAD, is characterized by a hollow-curve. In the sense of the SSAD, a hollow-curve implies that many areas are occupied by few or no individuals and that relatively few areas are occupied by many individuals. We generated random samples of the feasible set of the SSAD for ecologically realistic combinations of *q* and *n*, and examined their general features.

RESULTS

Statistical properties of entire feasible sets are indistinguishable from random subsets generated with our sampling algorithms, demonstrating that the implementation of our algorithms was unbiased (Fig 2 and Figs 1-2 of Appendix). When generating 300 random partitions, i.e. enough to safely characterize the feasible space (Locey and White 2013), these implementations were, at worst, one-to-two orders of magnitude faster than the method used by Locey and White (2013) and were, at best, 4 to 5 orders of magnitude faster for the combinations of *q* and *n* we tested (Fig 3). These combinations were limited to values for which the algorithm used in Sage could generate random samples in reasonable time. Consequently, the algorithms we developed quickly produce random samples for values of *q* and *n* that are impractical with algorithms that sample only according to *q*. Each algorithm was best suited for particular values of *q* and *n* (Fig 4). For cases where all parts have positive values, the multiplicity algorithm is the fastest for combinations where *q* is partitioned among a relatively small number of elements (Fig 3 Appendix).

The greater efficiency of the algorithms developed here allowed us to generate between 300 and 500 random partitions for 92.7% of the SADs (14786/15950) from the compilation of SAD data used by Locey and White (2013), in less than 1000 compute hours. In contrast, the method used by Locey and White (2013) required more than 10000 compute hours to generate between 300 and 500 random partitions for 60% of the available data (9562/15950 SADs).

Our examination of the SSAD feasible set supports the observation of Brown et al. (1995) that SSADs are characterized by hollow-curves (few cells with many individuals and many cells with few individuals) (Fig 5). The hollow-curve nature of the members of the feasible set increases as the total *q* is distributed across a greater number of elements (e.g. quadrats, areas).

DISCUSSION

The feasible set approach is a general method for understanding how constraints determine observable variation in the forms of macroecological patterns and in distributions of wealth, in general (Locey and White 2013). We used integer partitioning to understand how the feasible set is ordered, to find the size and general features of the feasible set, and to generate unbiased random samples of the feasible set for a given a total (e.g. total community abundance, total species abundance) and number of entities (e.g. species, quadrats). The algorithms we derived greatly increase the practical use of feasible set by decreasing computing time. In addition to examining the SAD, we expanded the feasible set approach to distributions with zero values, such as the SSAD. We also provided the algorithms in two computing languages frequently used by ecologists, R and Python, and have taken steps to ensure our implementations are unbiased.

Integer partitioning is only one way to examine and randomly sample the feasible set of possible SAD and SSAD shapes. Other possibilities include linear programming and iterative random walks, such as that used by Haegeman and Loreau (2008). Those approaches may not require combinatorial problems to be solved, i.e. $p_k(q)$, and so may not suffer from the problem of combinatorial explosion (large increases in the size of the feasible set for small changes in the total q and number of elements), as is the case with our algorithms. However, as stated by Locey and White (2013) one benefit to the integer partitioning approach is that random sampling algorithms can be derived that are inherently unbiased and do not require ‘burn-in’ periods to produce effectively independent samples. The combinatorial approach also reveals properties such as the size of the feasible set, mathematical properties and symmetries within the feasible set (e.g. the theorems and identity used in this study), and a way to relate the feasible sets of different patterns, e.g. SAD and SSAD, in purely quantitative terms. However, we suggest that the combinatorial algorithms developed here should be compared to approaches such as those in Haegeman and Loreau (2008).

Our examination of the SSAD feasible set (Fig. 5) reveals that the central tendency of the set is characterized by a hollow-curve which is consistent with the empirical SSADs found by Brown et al. (1995). In that study, the authors state that the highly ‘clumped’ and hollow-curve nature of the SSAD resembles distributions used to predict the form of the SAD. The authors offer an ecological interpretation for the similarity between the patterns in terms of niche requirements. However, the SSAD feasible set (in purely mathematical terms and without interpreting what the total *q* and number of elements *n* represent) differs from the SAD feasible set only in that zero values are allowed, which also allows for $n > q$. Consequently, the two patterns are not only coupled by ecological mechanisms, but are also coupled by the purely mathematical properties of their feasible sets.

The feasible set approach ignores biological and statistical mechanisms and focuses entirely on observable variation in the shape of empirical patterns. Consistency of empirical patterns with the center of the feasible set suggests that the shapes of those patterns contain little information beyond that encoded by the constraints used to characterize the feasible set (Haegeman and Loreau 2008, Locey and White 2013). However, consistency with the feasible set does not mean that biological processes are not operating but rather that they may indirectly influence empirical patterns through their effects on constraints (Supp et al. 2012, White et al. 2012). Alternatively if empirical patterns occupy an uncommon portion of the feasible set (e.g. in being exceptionally uneven) biological processes or additional constraints beyond those used to characterize the feasible set may be relevant.

Our work provides the opportunity to more fully explore the feasible set approach by greatly decreasing computational time and by defining the feasible set of another macroecological pattern, i.e., the SSAD. Our computational and theoretical advances enabled us to examine combinations of constraint values that were previously out of reach. The algorithms we developed apply to frequency distributions such as the SAD and SSAD. However, many macroecological patterns are also cumulative, describing the rates at which species are encountered with increasing area (species-area relationship), time (species-time relationship) or both area and time (species-time-area relationship). Characterizing and randomly sampling the feasible sets of these and other patterns may require extensive modification of the algorithms we developed, approaches more similar to that of Haegeman and Loreau (2008), or altogether new approaches.

ACKNOWLEDGMENTS

We thank X. Xiao and E. P. White for critical discussions and friendly reviews. We thank the numerous individuals involved in collecting and providing the data used in this paper including the essential citizen scientists who collect the North American Breeding Bird Survey and Christmas Bird Count data, USGS and CWS scientists and managers, researchers who collected and sequenced the microbial metagenomic data, the MG-RAST project, the Ribosome Database Project, the Audubon Society, the U.S. Forest Service, the Missouri Botanical Garden, and Alwyn H. Gentry.

LITERATURE CITED

Amend, A. S., Seifert, K. A., Samson, R., & Bruns, T.D. (2010). Indoor fungal composition is geographically patterned and more diverse in temperate zones than in the tropics. *P. Natl. Acad. Sci. USA.*, 107, 13748-13753.

Bóna, M. 2006. A walk through combinatorics: An introduction to enumeration and graph theory. 2nd Edition. World Scientific Publishing Co. Singapore.

Brown, J. H. 1995. Macroecology. Univ. Chicago Press, Chicago.

Brown, J. H., Mehlman, D. W. and G. C. Stevens. 1995. Spatial variation in abundance. Ecology, 76, 2028-2043.

Chu, H., Fierer, N., Lauber, C.L., Caporaso, J.G., Knight, R. & Grogan, P. (2010). Soil bacterial diversity in the Arctic is not fundamentally different from that found in other biomes. *Environ. Microbiol.*, 12,2998–3006.

Fierer, N., Lauber, C.L., Ramirez, K.S., Zaneveld, J., Bradford, M.A. & Knight, R. (2012). Comparative metagenomic, phylogenetic and physiological analyses of soil microbial communities across nitrogen gradients. *ISME J*., 6, 1007–17.

Flores, G.E., Campbell, J., Kirshtein, J., Meneghin, J., Podar, M., Steinberg, J.I. *et al*. (2011). Microbial community structure of hydrothermal deposits from geochemically different vent fields along the Mid-Atlantic Ridge. *Environ. Microbiol*., 13, 2158-2171.

Haegeman, B. and R. S. Etienne. 2010. Entropy Maximization and the Spatial Distribution of Species. Am. Nat., 175, E74–E90.

Haegeman, B. and M. Loreau. 2008. Limitations of entropy maximization in ecology. Oikos, 117, 1700–1710.

Harte, J., T. Zillio, E. Conlisk and A. B. Smith. 2008. Maximum entropy and the state-variable approach to macroecology. Ecology, 89, 2700–2711.

Locey, K. J. and E. P. White. 2013. How species richness and total abundance constrain the distribution of abundance. Ecology Letters, DOI: 10.1111/ele.12154.

McGlinn, D. J., and A. H. Hurlbert. 2012. Scale dependence in species turnover reflects variance in species occupancy. Ecology, 93, 294–302.

Meyer, F., Paarmann, D., D'Souza, M., Olson, R., Glass, E.M., Kubal, M. *et al*. (2008). The metagenomics RAST server - a public resource for the automatic phylogenetic and functional analysis of metagenomes. *BMC Bioinformatics*, 9, 386.

National Audubon Society. (2002). The Christmas Bird Count historical results. Retrieved from http://www.audubon.org/bird/cbc.

Nijenhuis, A. and H. S. Wilf. 1978. Combinatorial Algorithms for Computers and Calculators. Academic Press, New York.

Sauer, J.R., Hines, J.E., Fallon, J.E., Parkieck, D.J., Ziolkowski, D.J. Jr. & Link, W.A. (2011). *The North American Breeding Bird Survey 1966-2009*. Version 3.23.2011. USGS Patuxent Wildlife Research Center, Laurel, MD.

Stojmenovic, I. 2008. Generating all and random instances of a combinatorial object. In Handbook of Applied Algorithms: Solving scientific, Engineering and Practical Problems. John Wiley & Sons, Inc., New Jersey, pp. 1-38.

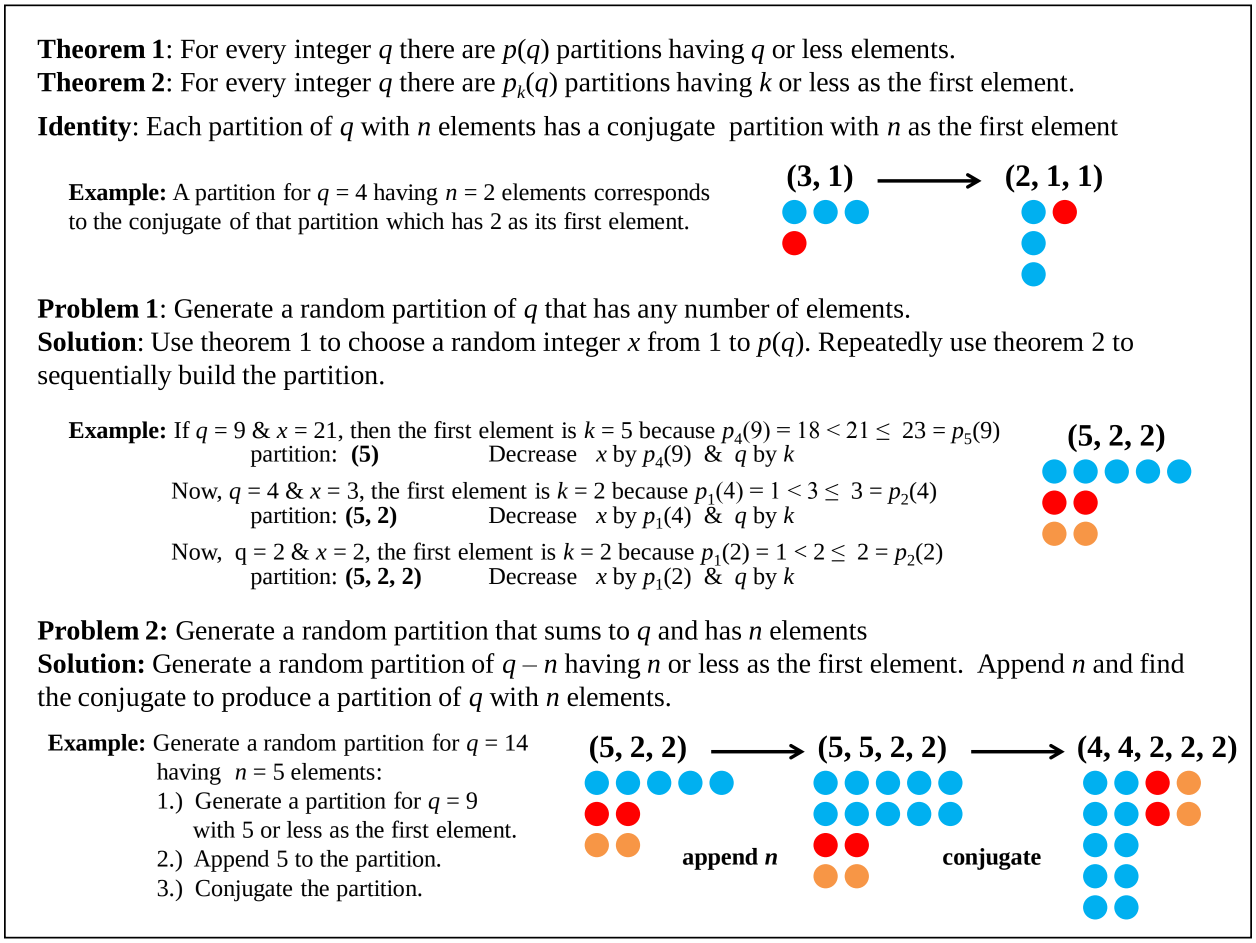
Storch, D., A. L. Šizling, J. Reif, J. Polechová, E. Šizlingová, and K. J. Gaston. 2008. The quest for a null model for macroecological patterns: geometry of species distributions at multiple spatial scales. Ecology Letters, 11,771–784.

Supp, S. R., X. Xiao, S. K. M. Ernest and E. P. White. 2012. An experimental test of the response of macroecological patterns to altered species interactions. Ecology, 93, 2505–2511.

Thibault, K.M., Supp, S.R., Giffin, M., White, E.P. & Ernest, S.K.M. (2011). Species composition and abundance of mammalian communities. *Ecology*, 92, 2316-2316.

U.S. Department of Agriculture, F.S. (2010). Forest inventory and analysis national core field guide (Phase 2 and 3), version 4.0. Washington, DC: U.S. Department of Agriculture Forest Service, Forest Inventory and Analysis.

White, E. P., K. M. Thibault and X. Xiao. 2012. Characterizing species abundance distributions across taxa and ecosystems using a simple maximum entropy model. Ecology, 93, 1772–1778.



Theorem 1: For every integer q there are $p(q)$ partitions having q or less elements.
Theorem 2: For every integer q there are $p_k(q)$ partitions having k or less as the first element.

Identity: Each partition of q with n elements has a conjugate partition with n as the first element

Example: A partition for $q = 4$ having $n = 2$ elements corresponds to the conjugate of that partition which has 2 as its first element.

Problem 1: Generate a random partition of q that has any number of elements.
Solution: Use theorem 1 to choose a random integer x from 1 to $p(q)$. Repeatedly use theorem 2 to sequentially build the partition.

Example: If $q = 9$ & $x = 21$, then the first element is $k = 5$ because $p_4(9) = 18 < 21 \leq 23 = p_5(9)$
partition: **(5)** Decrease x by $p_4(9)$ & q by k

Now, $q = 4$ & $x = 3$, the first element is $k = 2$ because $p_1(4) = 1 < 3 \leq 3 = p_2(4)$
partition: **(5, 2)** Decrease x by $p_1(4)$ & q by k

Now, $q = 2$ & $x = 2$, the first element is $k = 2$ because $p_1(2) = 1 < 2 \leq 2 = p_2(2)$
partition: **(5, 2, 2)** Decrease x by $p_1(2)$ & q by k

Problem 2: Generate a random partition that sums to q and has n elements
Solution: Generate a random partition of $q – n$ having n or less as the first element. Append n and find the conjugate to produce a partition of q with n elements.

Example: Generate a random partition for $q = 14$ having $n = 5$ elements:
1.) Generate a partition for $q = 9$ with 5 or less as the first element.
2.) Append 5 to the partition.
3.) Conjugate the partition.

Figure 1. *Top:* Two theorems and one identity used to generate random integer partitions. *Center:* General method for generating a random partition of q having 1 to q elements using the two theorems. *Bottom:* General method for generating a random partition of q into exactly n elements, using the two theorems and the partitioning identity.

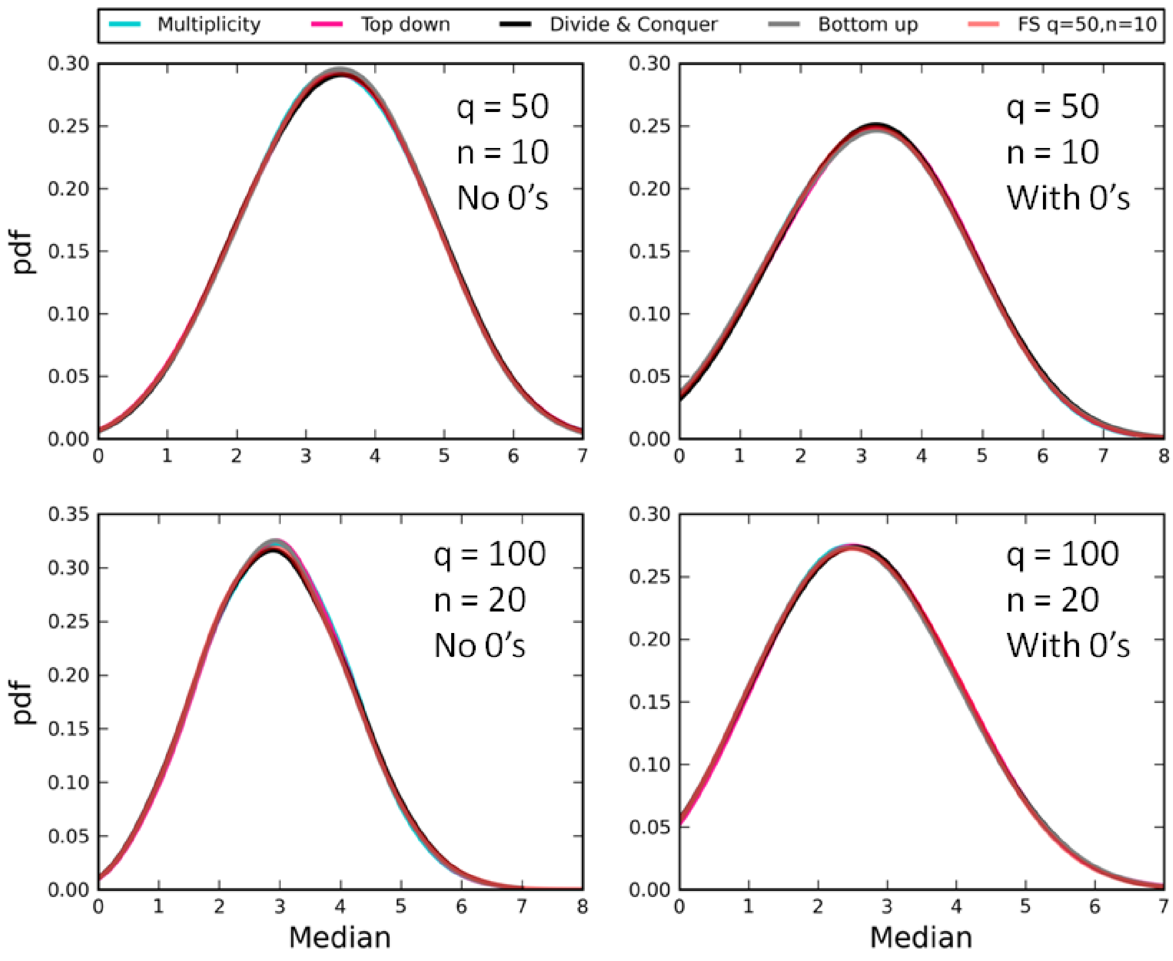


Figure 2. A comparison of the full feasible set and kernel density curves for the median derived from 1000 random samples for different combinations of *q* and *n* using our four new algorithims for parts without zeros (left column) and for parts with zeros (right column). The similarity between the results derived using our algorithms and the full feasible set reveals that the algorithms produce unbiased random samples of the feasible set. We used Sage to generate the entire feasible set for *q* = 50 and *n* = 10 (16928 partitions) and used the random partitioning function in Sage to generate 1000 partitions for *q* = 100 and *n* = 20, which is too large to enumerate in full in reasonable time (10474462 partitions).

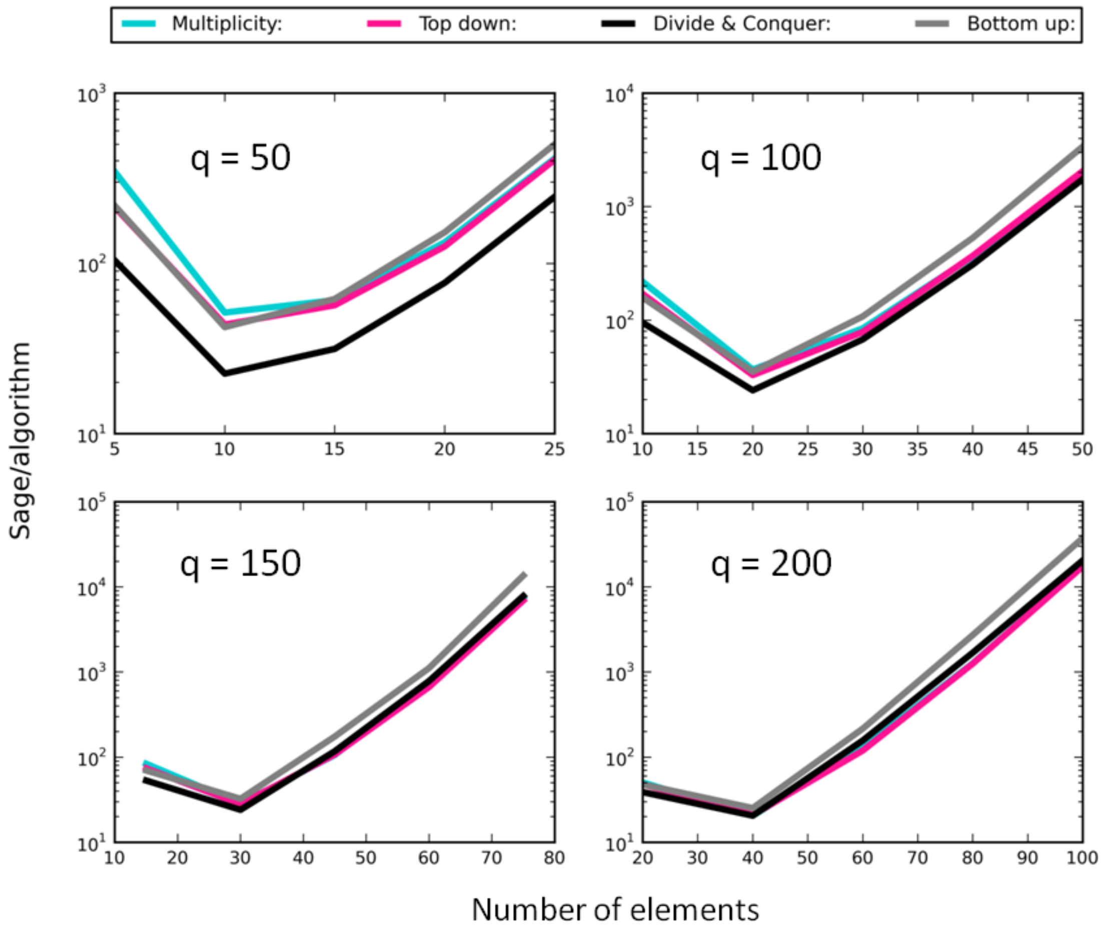


Figure 3. Plots of the ratio of the computational time for Sage to generate 300 random integer partitions (no zeros) to the time taken for the new algorithms ('Multiplicty', 'Top down', 'Divide and Conquer', 'Bottom up') to do the same.

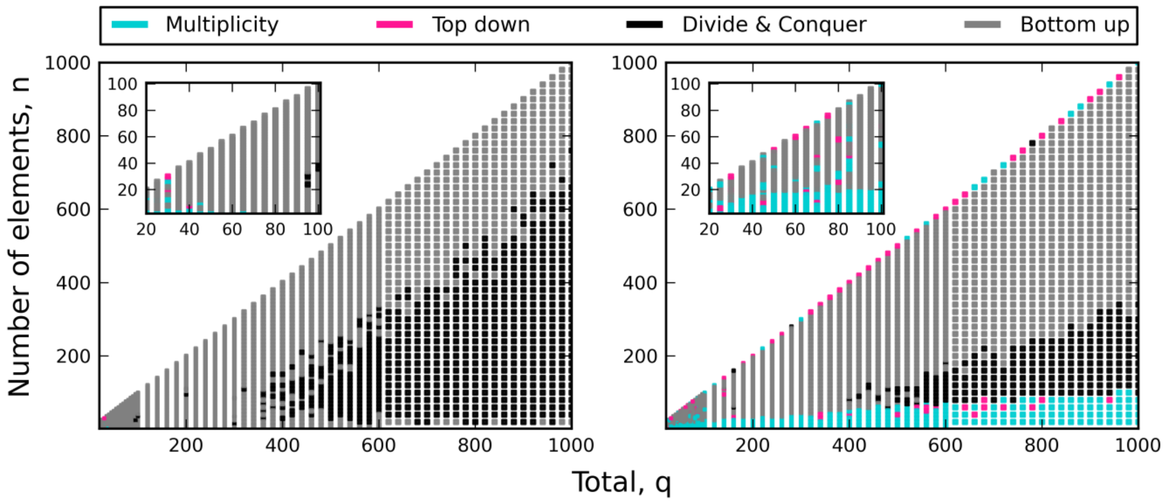


Figure 4. Color map revealing the fastest algorithm for specific combinations of $q \leq 1000$ and $n \leq q$. Comparisons were based on the time taken to generate 300 random partitions for each combination of q and n, both for cases where parts were allowed to have zero values (left) and when parts had positive values only (right). Insets reveal the small corner of the main graph where $q \leq 100$.

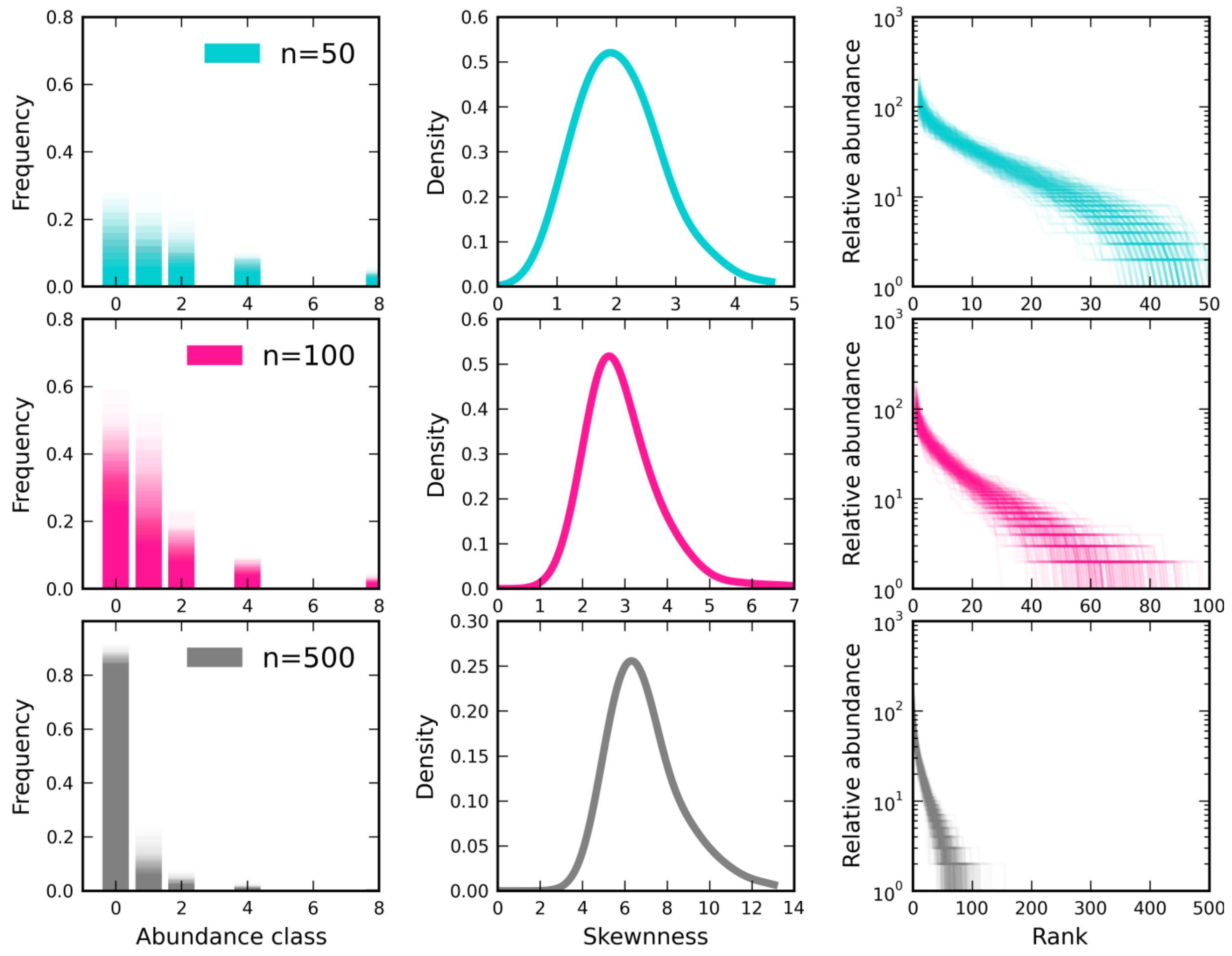


Figure 5. The SSAD feasible sets for *q* = 1000 and *n* = {50, 100, 500} for 500 random samples displayed as frequency distributions (left), kernel density curve of skewness (center), and ranked distributions of abundance (i.e. areas or subplots ranked from most-to-least occupied) (right). Each row applies to one value of *n*. All random samples were generated with the 'Divide and Conquer' algorithm. As *q* becomes distributed across an increasing number of elements, e.g. areas or quadrats, the feasible set in general becomes characterized by increasingly hollow-curves (i.e. few large elements and many small or zero-value elements), suggesting higher aggregation in SSADs as q is distributed across an increasingly large number of areas, quadrats, or subplots. For n = 500, ranks with values of 0 are plotted but difficult to discern.

Dataset	*Number of SADs	Locey & White **~10K hours**	Present Study **~1K hours**
North American Breeding Bird Survey	2769	1586, 57%	2769, 100%
Christmas Bird Count	1992	129, 6.5%	1231, 62%
Gentry's forest transects	222	182, 82%	221, 99.5%
Forest Inventory and Analysis	10356	7359, 71%	10101, 98%
Mammal Community Database	103	42, 41%	103, 100%
Aquatic metagenomes	252	48, 19%	120, 48%
Terrestrial metagenomes	128	92, 72%	113, 88%
Fungi metagenomes	128	124, 97%	128, 100%
Total	15950	9562, **60%**	14786, **92.7%**

* This is the number of SADs in the dataset matching these criteria: one randomly chosen SAD per site (most sites were sampled in multiple years) having 10 or more species recorded.

Table 1. Results of Locey and White (2013) and those from the reanalysis of those data used in that study reveal the practical gains of the algorithms developed here. Note, sampling effort per dataset was not controlled or accounted for.