Supporting Information Appendix

Diagonal and Off diagonal Impacts (S1)

To aid in showing some of the complex relationships between the diagonal and off diagonal elements of the sampled constrained community matrices we have included measures from the literature that have been used to examine some of these relationships (11, 19, 20). Specifically, we have added the max Gershgorin circle width, which is the radius of the most positive Gershgorin circle (largest center(i) + radius(i)). We have also included a measure of the elliptical bound of the eigenvalues like in (20). Finally, we also include a measure of the average radius of all Gershgorin circles (that is the radius of all the deleted row sums of the sampled community matrix). It is important to state that these measures are all just to show that our results are consistent with other common measures of matrix stability, it in no way is meant to be exhaustive, as this is a complex area of open research and is beyond the scope of this paper. What is important is that we find that these other measures are consistent with our approaches from the main text, and deeper understanding will require further research.

Hit and Run Algorithm (S2)

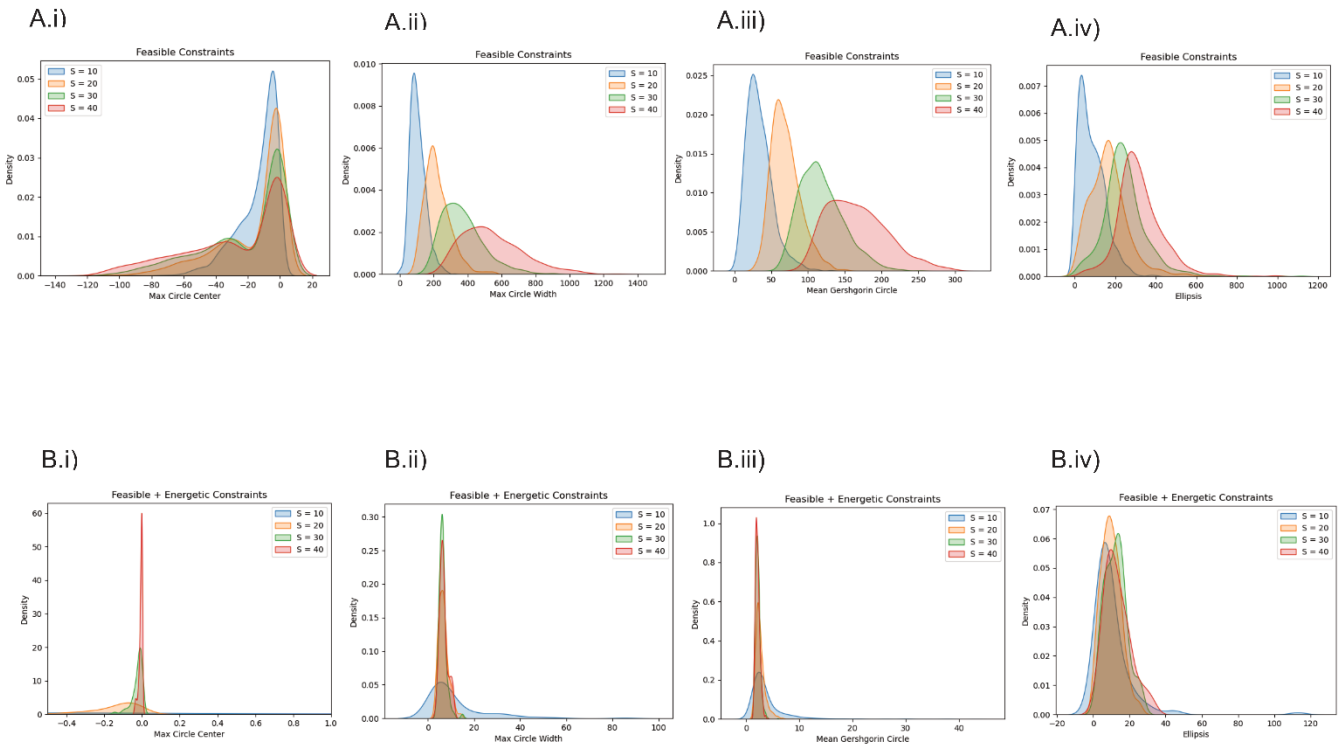
Hit and Run algorithms are a family of Monte Carlo methods that efficiently sample from convex domains (31, 37). Here we briefly illustrate the method to help give intuition with the approach, but it is recommended that the more detailed primarily literature be consulted for the details.

To begin a convex domain is given, an example is shown in Supplementary Figure 2, Step 1). Please note that this will work in general for matrix models, like Generalized Lotka Volterra models (if you bound the parameters like we have done), as linear matrix models like this form convex domains, more complex, nonlinear models may require more complex analysis to assure the convexity. The next step is to try and find an initial starting point that is inside the convex domain. Frequently this is done numerically by first bounding the convex domain in each dimension using linear programming, this is illustrated in Supplementary Figure 2, Step 2. Once you have this rough bounding box you will take the average of all these bounds to find an estimate of the center of the convex domain. This is not guaranteed, and you will need to check if you are inside the bounds numerically before starting the Hit and run sampler (otherwise it will not converge at likely go out to infinity in at least one dimension). For our models this failure never occurred, but the check is in place in the available numerical code.

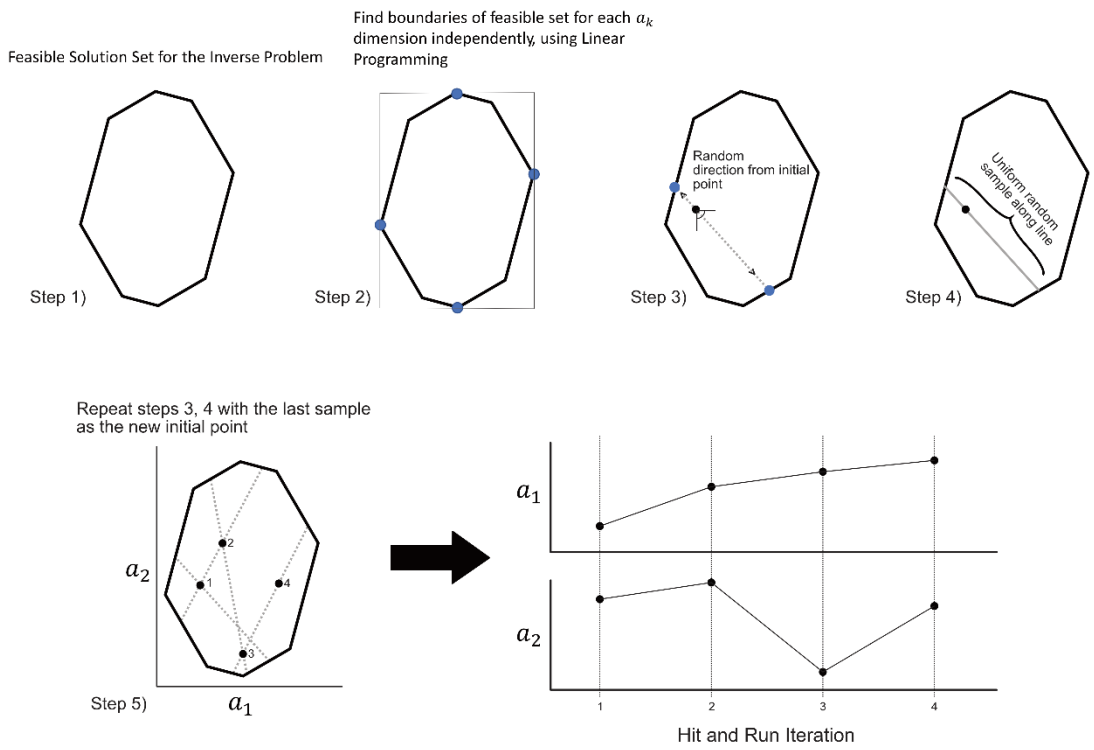
Now that the startup steps are complete the algorithm can start the Monte Carlo sampling procedure. From this initial interior point a random direction is chosen, the line with this direction going through the initial point is projected in both directions and the intersection with the boundary is found (basic linear algebra). With this line segment calculated a new point is found on the line at random (with uniform distribution along the line), this procedure is shown in Supplementary Figure 2, Step 3.

The newly selected point is then made the new “initial point” and the procedure repeats itself, “bouncing” around the interior of the geometric space, generating a random sequence of samples from inside the convex domain, this is illustrated in Supplementary Figure 2, Step 5.

This approach works well, and has been shown to converge to a uniform sample of the domain (37). Developments of the approach largely focus on ways of selecting the random direction between samples. The approach we employed using “artificial centering” which attempts to bias the direction slightly towards the estimated center of the domain to avoid inefficient time in corners (31), but this is just a numerical efficiency and does not change the core principle of the method.



Supplementary Figure 1. Distributional plots of the Gershgorin parameters (circle center of maximum Gershgorin circle as defined above; max gershgorin circle width, and ellipse area). Distributions for 5 to 40 species are colored as per inset. All distribution plots use automatic kernel smoothing with a Guassian kernel. Manual spot checks were used to see if nearby kernels significantly different shapes. We detected no such dependence.



Supplementary Figure 2. A schematic of the hit and run procedure as outlined in (37) for two-dimensional case (i.e., α_1 and α_2). As discussed, the technique first finds boundaries for each α_k using Linear programming (2). This defines a potentially complex space that the algorithm then explores by creating random linear trajectories across the boundaries determined from 2 internal randomly generated points (3). The line that is created from these points is then uniformly sampled (4). Steps 2 and 3 are done repeatedly allowing for the space to be well explored even if complex (5).