Appendices to "Generating realistic assemblages with a Joint Species Distribution Model"

Appendix A: Data

Code for this section can be found in
`mistnet/extras/BBS-analysis/data_extraction/data-extraction.R` in the version control repository at https://github.com/davharris/mistnet/.

Partitioning training and test data:

A set of evenly-spaced coordinates were selected across the Earth's surface using `regularCoordinates(12)` in the `geosphere` R package (Hijmans, Williams, and Vennes 2012). The 1559 routes that were more than 300 km from all of these coordinates were included in the training set, while the 280 routes that were less than 150 km away from one of them were included in the test set.

Criteria for including a species in the data set:

Hybrids and other ambiguous taxa were identified using simple heuristics (e.g. if the listed common or Latin name included the word "or"). Such species were excluded from the analysis; code for the heuristics used is available on the mistnet package's version control repository (`mistnet/extras/BBS-analysis/data_extraction/species-handling.R`). I also omitted any species that were observed along fewer than 10 training routes to ensure that enough data points were available for cross-validation. This left a pool of 368 species for analysis.

Included climate predictors:

The following eight climate predictors were selected out of the 19 Bioclim variables in the Worldclim data set, omitting nearly-collinear variables identified with the `findCorrelation` function in the `caret` package (Kuhn et al. 2012):

bio2:	Mean Diurnal Range
bio3:	Isothermality
bio5:	Max Temperature of Warmest Month
bio8:	Mean Temperature of Wettest Quarter
bio9:	Mean Temperature of Driest Quarter
bio15:	Precipitation Seasonality
bio16:	Precipitation of Wettest Quarter
bio18:	Precipitation of Warmest Quarter

Appendix B: Neural network structure

A feed-forward neural network, such as a mistnet model, performs a series of simple mathematical operations in nodes, called neurons, which are organized into layers. Each layer takes the output of the previous layer as input, with the first layer taking its input directly from environmental measurements or Monte Carlo samples. Each layer contains a matrix of tunable parameters, which project the incoming data vector into a new subspace. Layers typically also perform a pre-determined nonlinear operation on the projected vector, enabling the model to describe nonlinear relationships between its inputs and outputs.

As a *stochastic* network, mistnet models' first layer takes an augmented vector of inputs, $\tilde{x}$, which includes both the observed predictor variables ($\vec{x}$) and Monte Carlo samples of the latent variables ($\vec{z}$). The range of sampled $\vec{z}$ values allows the network to represent the possibility that two locations with similar environmental data ($\vec{x}$) will have systematically different species composition, owing to unmeasured variation. In these analyses, $\vec{z}$ is sampled from isotropic Gaussian noise.

The general structure of the network applied to the BBS data can be found in Figure 3B of the main text, and is described in more detail below (in all cases, "multiplication" refers to matrix multiplication).

The network multiplies $\tilde{x}$ by a tunable coefficient matrix $\mathbf{\Theta_1}$ and passes it through the piecewise linear transformation described in Zeiler et al. (2013), yielding:

$$f(\tilde{x}, \mathbf{\Theta_1}) = \max(0, \tilde{x}\mathbf{\Theta_1}).$$

The second layer takes the vector produced by $f(\tilde{x}, \mathbf{\Theta_1})$ and multiplies it by a low-rank coefficient matrix, $\mathbf{\Theta_2}$, reducing the dimensionality of the network's representation. This layer serves solely for dimensionality reduction, and does not perform any nonlinear operations on vector computed by $f(\tilde{x}, \mathbf{\Theta_1})\mathbf{\Theta_2}$.

Finally, the third layer takes the output of layer 2, multiplies it by $\mathbf{\Theta_3}$, and passes the resulting vector through the logistic function $\text{logistic}(x) = e^x/(1 + e^x)$.

The full model thus performs the following computation:

$$p(\vec{y}|\tilde{x}) = \text{logistic}(f(\tilde{x}, \mathbf{\Theta_1})\mathbf{\Theta_2}\mathbf{\Theta_3}),$$

where $p(\vec{y})$ is a vector of probabilities for observing each species in the data set. The goal of model fitting (Appendix C) is to select well-performing values for the elements of the $\mathbf{\Theta}$ matrices (e.g. values whose likelihoods or posterior probabilities are high).

Objective function (model posterior or penalized likelihood)

Mistnet models' likelihoods are given by $p(\vec{y}|\vec{x}, \mathbf{\Theta})$, where $\vec{y}$ is a presence-absence vector indicating which species were present in the community, $\vec{x}$ is a vector of measured environmental variables, and $\mathbf{\Theta}$ represents the model's coefficient matrices.

In order to calculate $p(\vec{y}|\vec{x}, \mathbf{\Theta})$, one must marginalize over the latent variables $\vec{z}$, usually by Monte Carlo integration:

$$p(\vec{y}|\vec{x}, \mathbf{\Theta}) = \int_{\vec{z}} p(\vec{y}|\vec{x}, \vec{z}, \mathbf{\Theta})\mathrm{d}\vec{z},$$

The log-posterior, which mistnet optimizes, is equal to the log-likelihood described above, plus two terms from the prior (plus an unknown normalization constant which can be ignored).

In the model fitted to BBS data, the prior distributions on elements of $\mathbf{\Theta}$ are each Gaussian (see Appendix D for discussion of how these distributions' parameters were chosen).

A secondary prior was also included on the elements of $p(\vec{y})$ using a Beta distribution, as specified in the text. This secondary prior ensures that the adjustable parameters, $\mathbf{\Theta}$, do not tend to produce extreme predictions (e.g. $p(\vec{y}) < 10^{-6}$).

The full log-posterior is thus composed of three terms

$$\log\big(p(\vec{y}|\vec{x}, \mathbf{\Theta})\big) - \sum_i \frac{(\mathbf{\Theta_i} - \mu_i)^2}{2\sigma_i^2} + \sum_j \log\big(p(\vec{y}_j)^{\alpha-1}(1 - p(\vec{y}_j)^{\beta-1})\big),$$

plus an unknown normalization constant. Here, μ_i and σ_i represent the means and standard deviations of the Gaussian priors noted above.

If one prefers non-Bayesian terminology, one could refer to the latter two terms as regularizing penalties rather than log-priors. Either way, the mistnet package adjusts the elements of Θ to maximize the sum of these three terms.

Appendix C: Model fitting

If the neurons' nonlinear activation functions are differentiable almost everywhere, then one can use the chain rule from calculus to find the gradient of the model's log-likelihood or posterior probability with respect to every coefficient in Θ. The associated algorithm, called backpropagation, allows for local optimization by calculating this gradient using the chain rule from calculus, and climbing it toward higher likelihoods (Murphy 2012).

Mistnet models' predictions, and thus their gradients, depend on unobserved environmental variation, which means that they cannot be calculated exactly without solving difficult integrals. However, one can collect Monte Carlo samples of possible gradients by marginalizing over the latent variables, yielding an approximate gradient (Tang and Salakhutdinov 2013). The mistnet code then adjusts the model parameters in the direction of this estimate.

Tang and Salakhutdinov (2013) show that this procedure is an example of generalized expectation maximization (Neal and Hinton 1998), which means that it will increase a lower bound on the model's log-likelihood until that bound coincides with a local maximimum.

Appendix D: Model configuration

BRT:

All these analyses used the `gbm` package. For each species, I evaluated BRT models with `interaction.depth` of 2, 5, and 8, and with up to 10,000 trees, using the default learning rate of 0.001. The number and depth of the trees was chosen by separate 5-fold cross-validations for each species (Murphy 2012). See `mistnet/extras/BBS-analysis/BBS_evaluation/gbm.R`.

Deterministic neural net:

`nnet`'s hyperparameters were optimized using random search (Bergstra and Bengio 2012). During this search, the number of hidden units was sampled uniformly between 1 and 50 and the weight decay was sampled from an exponential distribution with rate parameter 1. The model was allowed 1,000 BFGS iterations to optimize the log-likelihood in each configuration. The model was fit with 8 different configurations of hidden layer sizes and weight decay values, and the best configuration was selected by five-fold cross-validation. See `mistnet/extras/BBS-analysis/BBS_evaluation/nnet-evaluation.R`.

BayesComm:

I used a development version of BayesComm. This version can be downloaded and installed using the `devtools` package using the following command: `install_github("goldingn/BayesComm", ref = "0d710cda46a6e7427a560ee5b816c8ef5cd03eeb")`.

I used the "full" model type, which models species' responses to both observed and latent environmental factors. BayesComm performed 42,000 rounds of Gibbs sampling, discarding the first 2,000 values as "burn-in" and retaining every 80th sample thereafter. This left 500 samples, which was a small enough number to fit in 8 gigabytes of memory with some room to spare for additional computations. See `mistnet/extras/BBS-analysis/BBS_evaluation/bayescomm.R`

mistnet:

Worldclim variables were standardized to have zero mean and unit variance. The coefficients in each layer were initialized as random samples from a zero-mean Gaussian with a variance of 0.01. Bias (intercept) terms in the first layer were manually initialized at 1; bias terms in the second layer were left initialized at 0; bias terms in the final layer were automatically initialized as $\log(\frac{p}{1-p})$, where p is the proportion of routes where the corresponding species was observed. The variances of the Gaussian priors for the model's coefficients were adjusted in an empirical-Bayesian fashion every 10 iterations, using each prior's `update` method, with a minimum variance of 0.001. The means of the third layer's priors were also adjusted every 10 iterations, using the same function.

I tried 10 different hyperparameter configurations, chosen using random search (Bergstra and Bengio 2012). During this search, I varied the following hyperparameters:

- the number of routes to include in each round of gradient descent ("n.minibatch"), was sampled log-uniformly between 10 and 100. This range is commonly suggested in the neural network literature.
- the number of latent Gaussian variables ("sampler.size") was sampled log-uniformly between 5 and 20. This range was chosen based on my expectations about the likely number of uncorrelated environmental factors that would account for most of the variation in species composition.
- the number of importance samples to collect during each stage of gradient descent ("n.importance.samples") was sampled log-uniformly between 20 and 50, based on Tang and Salakhutdinov (2013)'s suggestion of a range between 20 and 30, combined with some previous experience suggesting that larger values might work better than smaller ones.
- The number of nodes to include in the first hidden layer ("n.layer1") was sampled uniformly between 20 and 50, based on previous experience with neural network models fit to this data set.
- The number of nodes to include in the second hidden layer ("n.layer2") was sampled uniformly between 5 and 20, based on my expectations about the dimensionality of species' environmental responses.
- Coefficient updates were performed using adagrad (Duchi, Hazan, and Singer 2011) and a learning rate of 0.1. Adagrad was chosen because of its relative insensitivity to hyperparameter values. The learning rate was chosen because it was the largest power of ten that did not introduce numerical problems.

These ranges will likely make for good starting points for many JSDM applications, with smaller values for the layer sizes and learning rates being more likely to perform well on smaller data sets. See also the `hyperparameters` vignette in the mistnet package, e.g. via `browseVignettes("mistnet")`.

For each hyperparameter configuration, I performed five-fold cross-validation, with 20 minutes allocated per fold for training. Cross-validation results are reported below, sorted by average route-level log-likelihood. The highest-performing set of hyperparameters (top row) was used to build the model used in the final analyses.

	n.minibatch	sampler.size	n.importance.samples	n.layer1	n.layer2	log.likelihood
1	79	10	28	37	15	-52.20
2	23	6	24	35	12	-52.26
3	15	15	25	43	10	-52.35
4	11	15	27	29	13	-52.37
5	37	13	36	31	15	-52.40
6	42	8	45	39	14	-52.57
7	81	8	22	23	11	-52.66
8	18	6	47	31	16	-53.05
9	46	20	28	22	9	-53.05
10	88	14	20	42	5	-55.29

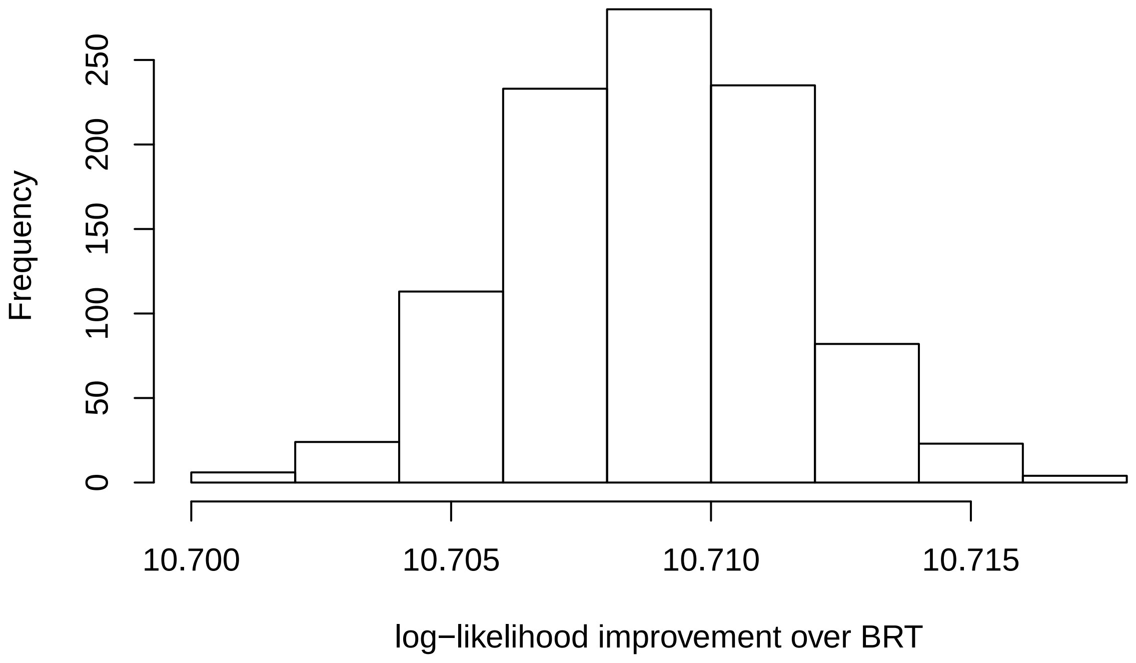
See `mistnet/extras/BBS-analysis/BBS_evaluation/mistnet_cross-validation.R` for the code used in hyperparameter selection.

Appendix E: Sampling error in marginal likelihood estimates

To evaluate the uncertainty in the Monte Carlo estimates of the mistnet model's predictive likelihood, I took 100000 independent predictions for each of the 280 routes in the test set. I then used bootstrap resampling to produce 1000 simulated estimates of model performance.

Among the 280 test-set routes, the median route's bootstrapped standard error was 0.02. Even the largest standard error was 0.27. Both of these values are substantially smaller than the standard error across the 280 test-set routes (0.78).

Across all 1000 bootstrapped samples, the mean route-level improvement over BRT (Figure 7B) was consistently very close to 10.7 log-likelihood units (see graph below). This corresponds to the factor of 45000 described in the main text. While it is possible that the importance sampler missed some very small areas with very high probability densities, these results indicate that the Monte Carlo estimates of model performance are probably not substantial overestimates.



Appendix F: Variance decomposition

The variance decomposition at the beginning of the Results section was performed as follows. For each species, I found the total variance in model predictions across both routes and Monte Carlo samples, as well as the residual variance within routes (i.e. after climate was accounted for). The proportion of non-climate variance for that species was calculated as residual variance divided by total variance.

References

Bergstra, James, and Yoshua Bengio. 2012. "Random Search for Hyper-Parameter Optimization." *J. Mach. Learn. Res.* 13 (March): 281–305. http://dl.acm.org/citation.cfm?id=2188385.2188395.

Duchi, John, Elad Hazan, and Yoram Singer. 2011. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization." *The Journal of Machine Learning Research* 12: 2121–59.

Hijmans, Robert J., Ed Williams, and Chris Vennes. 2012. *geosphere: Spherical Trigonometry*. http://CRAN.R-project.org/package=geosphere.

Kuhn, Max, Jed Wing, Steve Weston, Andre Williams, Chris Keefer, Allan Engelhardt, and others. 2012. "Caret: classification and Regression Training." *R Package Version* 5: 023.

Murphy, Kevin P. 2012. *Machine Learning: A Probabilistic Perspective*. The MIT Press.

Neal, Radford M, and Geoffrey E Hinton. 1998. "A View of the EM Algorithm That Justifies Incremental, Sparse, and Other Variants." In *Learning in Graphical Models*, 355–68. Springer.

Tang, Yichuan, and Ruslan Salakhutdinov. 2013. "Learning Stochastic Feedforward Neural Networks." In *Advances in Neural Information Processing Systems 26*, edited by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, 530–38. http://papers.nips.cc/paper/5026-learning-stochastic-feedforward-neural-networks.pdf.

Zeiler, Matthew D, M Ranzato, Rajat Monga, M Mao, K Yang, Quoc Viet Le, Patrick Nguyen, et al. 2013. "On Rectified Linear Units for Speech Processing." In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 3517–21. IEEE.