Journal of Biogeography

SUPPORTING INFORMATION

A new null model approach to quantify performance and significance for ecological niche models of species distributions

Corentin L. Bohl, Jamie M. Kass, and Robert P. Anderson

Appendix S1 Introduction supplementary details

Model performance: discriminatory ability vs. overfitting:

High discriminatory ability and low overfitting represent two desirable qualities of niche/distribution models (Lobo et al. 2008, Peterson et al. 2011, pp150-176, Warren and Seifert 2011). Discriminatory ability refers to the ability of a model to distinguish suitable from unsuitable areas. Overfitting is the tendency of a model to fit the random error (or any bias in the sample) rather than the true relationship between the calibration records and predictor variables. Hence, overfit models predict calibration data well but perform poorly on independent data sets.

Discrimination and overfitting can be quantified with various measures (Boyce et al., 2002; Peterson et al., 2011, pp 154-181). Discriminatory ability is often measured with the area under the curve of the receiver operating characteristic plot (AUC/ROC) (Peterson et al. 2011, pp 169-176, but see Jiménez-Valverde 2012), a non-parametric rank-based method. High AUC values indicate that presences are predicted more strongly (i.e., ranked higher) than absences (or pseudo-absences, or background pixels), across all possible thresholds of prediction strength. Complementarily, one intuitive measure of overfitting involves the difference between the AUC calculated on calibration (= training) data and that from evaluation (= testing) data (AUC_{DIFF} ;

Warren and Seifert 2011). Overfitting also can be assessed with the false negative rate, or omission error rate (OR henceforth), which indicate the proportion of presences incorrectly classified as falling into unsuitable areas of a binary prediction (Anderson et al. 2003). When compared with a theoretically expected rate for a given threshold, the OR indicates the level of overfitting (Radosavljevic & Anderson, 2013).

Null hypothesis, effect size, and significance:

In addition to assessing whether AUC is high or OR is low, it is also important to determine whether the obtained value is statistically better than expected by chance (i.e., its significance), and how much better it is than a random prediction (i.e., its effect size). Significance indicates the degree of confidence for a departure from a random prediction. Complementarily, effect sizes allow comparison of results across different samples, taxa, and methodologies. To obtain each of these, a researcher must define the random expectation correctly and measure departure from random with an unbiased estimate.

For the OR, binary predictions that cover a large portion of the study region are expected include a higher proportion of presences by chance alone (and thus yield lower ORs) than do more restricted predictions. Specifically, the theoretical random expectation is the inverse of the fractional predicted area corresponding to the threshold chosen (i.e.: if 60% of the area is predicted as suitable, then the OR for a sample of random pixels will be ~40%). Hence, departure from this expectation can be assessed with the p-value obtained from a binomial test (or a chi-square approximation), where the probability of predicting a presence is equal to the fractional predicted area (Anderson et al. 2002, Peterson et al. 2011, p 168).

In contrast, lack of absence data typically precludes easy interpretation of AUC values and assessment of significance. In theory, a random prediction follows the y = x diagonal across the ROC plot, where the true positive rate equals the false positive rate. Such a plot yields an AUC value of 0.5, with departure from that random expectation often judged using arbitrary rules of thumb for values over 0.5 (Swets 1988, Peterson et al. 2011, p 172). However, this expectation

is not valid for AUCs calculated without perfect absence data (Peterson et al. 2011, pp 170-175, Peterson et al. 2008). When pseudo-absence or background pixels are used in lieu of true absences, AUCs are generally calculated by approximating the false positive rate with the proportion of the pixels predicted present at any given threshold across the span of prediction strengths (Anderson et al. 2003, Phillips et al. 2006, Peterson et al. 2008, 2011, p 171). A p-value for AUCs calculated in that manner can be estimated using bootstrap replicates of the calibration or evaluation data (e.g., Peterson et al. 2008).

Unfortunately, however, these commonly used implementations of statistical tests for OR and AUC do not account for spatial patterns in the location of the model successes and failures (Lobo et al., 2008; Pontius & Schneider, 2001). The random expectations of these measures expose a key assumption: for any given threshold or prediction strength, the expected proportion of randomly selected pixels predicted as suitable is given *only* by the fractional predicted area. Unfortunately, this assumption is generally not reasonable due to spatial structure in environmental conditions across the study region (e.g. spatial autocorrelation of environments, and unequal proportions of various environmental conditions available). Because of such spatio-environmental heterogeneity, an appropriate null hypothesis for a given study region should take into account the spatial characteristics of the environment.

Appendix S2 Null-models approaches and their respective null hypothesis

Null-model approach	Null hypothesis
Raes ter Steege (2007)	A model calibrated with real species records is no better at predicting those same records, than a model constructed with geographically random pixels is at predicting those same random pixels. [Originally,

	predictive ability was measured via AUCtrain, but it does not necessarily have to be using AUC.]
Beale et al. (2008)	A model calibrated with a subset of real species records is no better at predicting the remaining (withheld) real records, than a model calibrated and evaluated with different respective sets of random pixels that each have the same degree of spatial autocorrelation as the real species records.
This study	A model calibrated with a subset of real species records is no better at predicting the remaining (withheld) real records, than is a model calibrated with geographically random pixels. [Here, the remaining (withheld) real species records constitute a *spatially independent* subset, but that spatial independence is not necessary].

Appendix S3 Methods supplementary details

Species data:

We gathered 2997 occurrence records of the morphologically distinctive monk parakeet from various sources (GBIF and Species Link databases, records from the primary literature, and personal observations; compiled in April 2011). This parakeet is native to temperate South America and has established several stable and well documented populations worldwide (Muñoz & Real, 2006; Pruett-Jones et al., 2011). We ignored records for the sub-species *M. monachus luchsi* because it is not found in introduced populations, and is geographically, genetically, and behaviorally distinct form the other sub-species (Russello, Avery, & Wright, 2008). To identify outliers unlikely to be part of well-established monk parakeet populations, we examined these records in geographic and environmental space. First, environmental outliers

were identified based on their distance from the multivariate arithmetic mean, using the Mahalanobis distance, and with a 1% alpha level (Calenge, Darmon, Basille, Loison, & Jullien, 2008; Farber & Kadmon, 2003; Rousseeuw & van Zomeren, 1990). Second, geographic outliers were identified based on the criterion that any presence must be part of an existing population and thus potentially connected to other presences via dispersal (Waples & Gaggiotti, 2006). We considered 106 km as the best estimate of dispersal capabilities for this species because it is the maximum reported dispersal distance from population genetic analyses (Gonçalves da Silva, Eberhard, Wright, Avery, & Russello, 2010), and because it corresponds almost exactly to the cutoff of the 1% upper tail of the distribution of the log of nearest-neighbor distances between the records in this study. Outliers identified with these methods were excluded from the subsequent analyses unless they were well documented in the primary literature. In addition, to reduce the degree of spatial-autocorrelation (that likely derives at least in part from biases in sampling effort) and match the resolution of the environmental data, we kept only one record per ~10 km^2 raster cell.

We assigned the resulting 797 records to different native and introduced "populations" worldwide. For the native range, we used the limit of the known distribution (Forshaw & Cooper, 1989; Ridgely et al., 2007). For the invaded regions, we used the aforementioned criterion that occurrences from the same population must be potentially connected via dispersal. We selected four of these populations as calibration sets to create the different models: the native population in Argentina and adjacent countries (AR, n = 121), and the three largest introduced populations (Spain, ES, n = 114; the wider New York City metropolitan area, NY, n = 96; and the southern part of Florida, FL, n = 119). The 347 remaining records, including the populations in Illinois, Texas, Louisiana, Puerto Rico, the northern part of Florida, and a number of smaller populations in the rest of the world, were reserved as a spatially independent evaluation set. We opted for spatially independent evaluation because it offers a more realistic test of performance than randomly selected evaluation records; the latter, being more sensitive to environmental spatial

autocorrelation, typically result in inflated estimates (Hijmans, 2012; Radosavljevic & Anderson, 2013; Veloz, 2009)

Environmental data:

Frostbite resulting from the combination of cold and wet winters is a common cause of mortality for this parakeet and other parrot species in temperate regions (Butler, 2005; Tamara & Arnheim, 1996). This finding is supported by Strubbe and Matthysen (2009), who modeled the distribution of monk parakeets in their native range and Europe and found that the best predictor was the number of frost days per year, closely followed by the log of human population density, and to a lesser extent high mean winter NDVI (an index of plant productivity, hence food availability). These birds tolerate cold temperature but compensate with a higher metabolic rate (Caccamise & Weathers, 1977; Weathers & Caccamise, 1975), and consequently require more food during cold periods. However, introduced populations in northern cities may predominantly depend on seeds from bird feeders during winter months (South & Pruett-Jones, 2000). Thus, human population density may be a better indication of winter food availability than winter NDVI. In fact, this species is associated with human presence in both native and invaded areas. This trend is supported by the study described earlier (Strubbe & Matthysen, 2009), as well as another modeling study where human influence was found to account for almost 64% of the Spanish distribution (Muñoz & Real, 2006). Furthermore, this species prefers open areas over dense forest, and favors cultivated lands (Burger & Gochfeld, 2005).

Therefore, we considered a combination of 24 climatic, anthropogenic, and land cover predictors at 5 arc-minutes resolution, which is approximately equivalent to the daily foraging range of this bird (Forshaw & Cooper, 1989; Spreyer & Bucher, 1998). These predictors are: the 19 WorldClim bioclimatic variables (Hijmans, Cameron, Parra, Jones, & Jarvis, 2005); three land cover variables from the Harmonized World Soil Database: percent forested land, percent grass/scrub land, and percent cultivated land (Fischer et al., 2008); and two anthropogenic predictors: the human population count (Gridded Population of the World, Version 3, 2005) and

the Global Human Influence Index (Last of the Wild Project, Version 2, 2005; Sanderson et al., 2002).

Model evaluation:

We assessed the ability of models to predict the 347 records of the evaluation set. To allow direct comparison of the evaluation statistics across the different scenarios, we calculated the values over the same background area corresponding to a 106 km distance buffer around all the filtered occurrence records (i.e. both calibration and evaluation records). We also tried considering the buffered region around only the 347 records of the evaluation set, but this method made it impossible to calculate meaningful AUC_{DIFF} and resulted in only marginal differences in pattern for the other statistics (not shown).

Furthermore, regarding AUC_{DIFF}, in a few instances the difference between AUC_{TRAIN} and AUC_{TEST} was negative (albeit small. Therefore, for simplicity, we considered the absolute value of this difference when calculating this statistic.

Effect size and significance:

For each combination of Maxent settings and set of calibration records considered in our analyses, we ran one model based on the real species data, and 1,000 corresponding replicate models based on random records to create null distributions of AUC_{TRAIN}, AUC_{TEST}, AUC_{DIFF}, and OR values. Then, for each scenario and each performance measure, we calculated a standardized effect size as a *Z*-transformed score [$Z = (x - \mu)/\sigma$] (Ulrich & Gotelli, 2010). Here, x is the observed value of the given performance measure for the real model, and μ and σ are, respectively, the mean and standard deviation of the 1000 values obtained for each of the null replicates. We also calculated one-tailed *p*-values of the *Z*-transformed scores using a parametric one-tailed *Z*-test. Note that we also considered a non-parametric approach based on the rank of the value obtained for real species data compared with the corresponding null

values. However, both methods produced equivalent results (not shown) and we therefore only reported the *p*-values obtained with the *Z*-test method.

REFERENCES

Anderson, R., Lew, D., & Peterson, A. (2003). Evaluating predictive models of species' distributions: criteria for selecting optimal models. *Ecological Modelling*, *162*, 211–232. Retrieved from http://www.sciencedirect.com/science/article/pii/S0304380002003496

Anderson, R. P., Gomez-Laverde, M., & Peterson, A. T. (2002). Geographical distributions of spiny pocket mice in South America: insights from predictive models. *Global Ecology and Biogeography*, *11*(2), 131–141. https://doi.org/10.1046/j.1466-822X.2002.00275.x

Boyce, M. S., Vernier, P. R., Nielsen, S. E., & Schmiegelow, F. K. A. (2002). Evaluating resource selection functions. *Ecological Modelling*, *157*, 281–300. https://doi.org/https://doi.org/10.1016/S0304-3800(02)00200-4

Burger, J., & Gochfeld, M. (2005). Nesting behavior and nest site selection in monk parakeets (Myiopsitta monachus) in the Pantanal of Brazil. *Acta Ethologica*, *8*(1), 23–34. https://doi.org/10.1007/s10211-005-0106-8

Butler, C. J. (2005). Feral Parrots in the Continental United States and United Kingdom: Past, Present, and Future. *Journal of Avian Medicine and Surgery*, *19*(2), 142–149. https://doi.org/10.1647/183

Caccamise, D. F., & Weathers, W. W. (1977). Winter nest microclimate of Monk Parakeets. *The Wilson Bulletin*, *89*(2), 346–349. Retrieved from http://www.jstor.org/stable/10.2307/4160925

Calenge, C., Darmon, G., Basille, M., Loison, a, & Jullien, J.-M. (2008). The factorial decomposition of the Mahalanobis distances in habitat selection studies. *Ecology*, *89*(2), 555–566. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/18409444

Farber, O., & Kadmon, R. (2003). Assessment of alternative approaches for bioclimatic modeling with special emphasis on the Mahalanobis distance. *Ecological Modelling*, *160*(1–2), 115–130. https://doi.org/10.1016/S0304-3800(02)00327-7

Fischer, G., Nachtergaele, F., Prieler, S., van Velthuizen, H. T., Verelst, L., & Wiberg, D. (2008). *Global Agro-ecological Zones Assessment for Agriculture (GAEZ 2008)*. IIASA, Laxenburg, Austria and FAO, Rome, Italy.

Forshaw, J. M., & Cooper, W. T. (1989). *Parrots of the world* (Third [Rev). Willoughby, Australia: Lansdowne Editions.

Gonçalves da Silva, A., Eberhard, J. R., Wright, T. F., Avery, M. L., & Russello, M. A. (2010). Genetic evidence for high propagule pressure and long-distance dispersal in monk parakeet (Myiopsitta monachus) invasive populations. *Molecular Ecology*, *19*(16), 3336–3350. https://doi.org/10.1111/j.1365-294X.2010.04749.x

Gridded Population of the World, Version 3 (GPWv3): Population Count Grid. (2005). Palisades, NY: NASA Socioeconomic Data and Applications Center (SEDAC). Retrieved from http://sedac.ciesin.columbia.edu/data/set/gpw-v3-population-count.

Hijmans, R. J. (2012). Cross-validation of species distribution models: removing spatial sorting bias and calibration with a null model. *Ecology*, *93*(3), 679–688. https://doi.org/10.1890/11-0826.1

Hijmans, R. J., Cameron, S. E., Parra, J. L., Jones, P. G., & Jarvis, A. (2005). Very high resolution interpolated climate surfaces for global land areas. *International Journal of Climatology*, *25*(15), 1965–1978. https://doi.org/10.1002/joc.1276

Jiménez-Valverde, A. (2012). Insights into the area under the receiver operating characteristic curve (AUC) as a discrimination measure in species distribution modelling. *Global Ecology and Biogeography*, *21*(4), 498–507. https://doi.org/10.1111/j.1466-8238.2011.00683.x

Last of the Wild Project, Version 2, 2005 (LWP-2): Global Human Influence Index (HII) Dataset. (2005). Palisades, NY: NASA Socioeconomic Data and Applications Center (SEDAC). Retrieved from http://sedac.ciesin.columbia.edu/data/set/wildareas-v2-human-influence-index-geographic.

Lobo, J. M., Jiménez-Valverde, A., & Real, R. (2008). AUC: a misleading measure of the performance of predictive distribution models. *Global Ecology and Biogeography*, *17*(2), 145–151. https://doi.org/10.1111/j.1466-8238.2007.00358.x

Muñoz, A.-R., & Real, R. (2006). Assessing the potential range expansion of the exotic monk parakeet in Spain. *Diversity and Distributions*, *12*, 656–665. https://doi.org/10.1111/j.1366-9516.2006.00272.x

Peterson, A. T., Papeş, M., & Soberón, J. (2008). Rethinking receiver operating characteristic analysis applications in ecological niche modeling. *Ecological Modelling*, *213*(1), 63–72. https://doi.org/10.1016/j.ecolmodel.2007.11.008

Peterson, A. T., Soberón, J., Pearson, R. G., Anderson, R. P., Martínez-Meyer, E., Nakamura, M., & Araújo, M. B. (2011). *Ecological Niches and Geographic Distributions*. (S. A. Levin & H. S. Horn, Eds.), *Monographs in Population Biology* (Vol. 49). Princeton University Press. Retrieved from http://press.princeton.edu/titles/9641.html

Phillips, S. J., Anderson, R. P., & Schapire, R. (2006). Maximum entropy modeling of species geographic distributions. *Ecological Modelling*, *190*(3–4), 231–259. https://doi.org/10.1016/j.ecolmodel.2005.03.026

Pontius, R., & Schneider, L. (2001). Land-cover change model validation by an ROC method for the Ipswich watershed, Massachusetts, USA. *Agriculture, Ecosystems & Environment*, *85*, 239–248. Retrieved from http://www.sciencedirect.com/science/article/pii/S0167880901001876

Pruett-Jones, S., Appelt, C. W., Sarfaty, A., Vossen, B., Leibold, M. a., & Minor, E. S. (2011). Urban parakeets in Northern Illinois: A 40-year perspective. *Urban Ecosystems*, *15*(3), 709–719. https://doi.org/10.1007/s11252-011-0222-3

Radosavljevic, A., & Anderson, R. P. (2013). Making better M axent models of species distributions: complexity, overfitting and evaluation. *Journal of Biogeography*, n/a-n/a. https://doi.org/10.1111/jbi.12227

Ridgely, R. S., Allnutt, T. F., Brooks, T., McNicol, D. K., Mehlman, D. W., Young, B. E., & Zook, J. R. (2007). *Digital Distribution Maps of the Birds of the Western Hemisphere, version 3.0.* NatureServe, Arlington, Virginia, USA.

Rousseeuw, P. J., & van Zomeren, B. C. (1990). Unmasking Multivariate Outliers and Leverage Points. *Journal of the American Statistical Association*, *85*(411), 633–639. https://doi.org/10.1080/01621459.1990.10474920

Russello, M. a, Avery, M. L., & Wright, T. F. (2008). Genetic evidence links invasive monk parakeet populations in the United States to the international pet trade. *BMC Evolutionary Biology*, *8*, 217. https://doi.org/10.1186/1471-2148-8-217

Sanderson, E. W., Jaiteh, M., Levy, M. a., Redford, K. H., Wannebo, A. V., & Woolmer, G. (2002). The Human Footprint and the Last of the Wild. *BioScience*, *52*(10), 891. https://doi.org/10.1641/0006-3568(2002)052[0891:THFATL]2.0.CO;2

South, J. M., & Pruett-Jones, S. (2000). Patterns of flock size, diet, and vigilance of naturalized Monk parakeets in Hyde Park, Chicago. *The Condor*, *102*(4), 848–854. Retrieved from http://www.bioone.org/doi/abs/10.1650/0010-5422(2000)102[0848:POFSDA]2.0.CO;2

Spreyer, M. F., & Bucher, E. H. (1998). Monk Parakeet (Myiopsitta monachus). In A. Poole & F. Gill (Eds.), *The birds of North America, No. 322*. Philadelphia, PA.

Strubbe, D., & Matthysen, E. (2009). Establishment success of invasive ring-necked and monk

parakeets in Europe. *Journal of Biogeography*, *36*(12), 2264–2278. https://doi.org/10.1111/j.1365-2699.2009.02177.x

Swets, J. (1988). Measuring the accuracy of diagnostic systems. *Science*, *240*(4857), 1285–1293. https://doi.org/10.1126/science.3287615

Tamara, K., & Arnheim, R. (1996). Perruches à collier (Psittacula krameri) victimes des conditions climatiques en région bruxelloise. *Aves*, *33*(2), 128–129.

Ulrich, W., & Gotelli, N. J. (2010). Null model analysis of species associations using abundance data. *Ecology*, *91*(11), 3384–3397. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/21141199

Veloz, S. D. (2009). Spatially autocorrelated sampling falsely inflates measures of accuracy for presence-only niche models. *Journal of Biogeography*, *36*(12), 2290–2299. https://doi.org/10.1111/j.1365-2699.2009.02174.x

Waples, R. S., & Gaggiotti, O. (2006). What is a population? An empirical evaluation of some genetic methods for identifying the number of gene pools and their degree of connectivity. *Molecular Ecology*, *15*(6), 1419–1439. https://doi.org/10.1111/j.1365-294X.2006.02890.x

Warren, D. L., & Seifert, S. N. (2011). Ecological niche modeling in Maxent: the importance of model complexity and the performance of model selection criteria. *Ecological Applications*, *21*(2), 335–342. https://doi.org/10.1890/10-1171.1

Weathers, W. W., & Caccamise, D. F. (1975). Temperature regulation and water requirements of the monk parakeet, Myiopsitta monachus. *Oecologia*, *18*(4), 329–342. https://doi.org/10.1007/BF00345853

Appendix S4 Supplementary Results

Table S2.1 Significance of 15 scenarios of Maxent models for the monk parakeet created with five different sets of calibration records across three levels of model complexity. Four measures of model performance are evaluated: AUC_{TRAIN} and AUC_{TEST} (AUCs calculated on calibration and evaluation records, respectively), AUC_{DIFF} (= AUC_{TRAIN} - AUC_{TEST}), and OR (the omission error rate with a threshold of 10% of calibration presences). The p-values displayed are calculated using one-tailed Z-tests based null distributions estimated with 1000 replicate of models based on random geographic pixels, except for the last column, which shows p-values calculated using a binomial test for OR.

Calibration Set	Settings	AUC_{TRAIN}	AUC_{TEST}	AUC_{DIFF}	OR	$OR_{BINOMIAL}$
ARESNYFL	Simple	0.000000	0.021692	0.103219	0.234657	0.000000
ARESNYFL	Default	0.000000	0.001269	0.264862	0.209667	0.000000
ARESNYFL	Complex	0.000000	0.000071	0.084333	0.099007	0.000000
AR	Simple	0.223805	0.070292	0.251800	0.182018	0.000000
AR	Default	0.176293	0.000954	0.047187	0.024337	0.000000
AR	Complex	0.415855	0.008607	0.080493	0.055134	0.000000
ES	Simple	0.042012	0.029614	0.244349	0.526343	0.000000
ES	Default	0.044344	0.027030	0.351440	0.828142	0.000000
ES	Complex	0.113760	0.042723	0.338025	0.863281	0.855347
NY	Simple	0.045487	0.120383	0.726645	0.981667	1.000000
NY	Default	0.054621	0.022456	0.379699	0.761669	0.000000
NY	Complex	0.149044	0.013776	0.213984	0.778541	0.000000
FL	Simple	0.070861	0.096993	0.706655	0.982868	0.023735
FL	Default	0.098502	0.017185	0.453077	0.961634	0.620780
FL	Complex	0.165527	0.035468	0.462861	0.888298	0.498190

Appendix S5 R Scripts

This section provides R scripts for the following analyses:

1) The main function to evaluate the performance of a model with AUCtest, AUCdiff, and OR for a specified threshold, and to calculate significance with null models (and the binomial test for OR). The progress is displayed on screen and the results can optionally be saved to a file after each Maxent iteration. This function can be modified to calculate different evaluation statistics and to run different modeling algorithms than Maxent.

2) A script to run in parallel several models with varying levels of complexity (i.e. varying the types of feature classes and the value of the regularization multiplier used in Maxent):

```
## SCRIPTS BEGIN HERE
## Both scripts may require the following libraries, and options to be loaded in R:
Sys.setenv(NOAWT=TRUE) #only on some macs before loading rJava
library(rJava) # required to run Maxent within dismo
#optional, to set the memory allocated to java, hence maxent (before loading dismo)
options(java.parameters = "-Xmx2g" )
library(dismo) # should also load automatically the required packages sp and raster
## 1) The main null model function:
mxt.nulltest(train, bg, n = 9, proj=bg, c.proj=TRUE, args = "addsamplestobackground", threshold = .1, abs.diff=TRUE, test=NULL, group=NULL, bg.group=NULL,save=NULL)
## The inputs for this function are:
## - train : a dataframe of the environmental values for the species calibration presences: one row for each presence, one column for each predictor. Note that categorical variable must be indicated with as.factor() .
## - n : the number of null replicates.
## - bg : the same as “train” but for the background points (the comparisons records for the model).
```

- proj : a dataframe in the same format as “train” and “bg” which corresponds to the pixels over which the statistics will be calculated. By default this is the same as “bg” but it may be useful when the models are projected to a different region.

- c.proj : TRUE, if “proj” includes both the calibration and evaluation records, FALSE, if it includes only the evaluation records (note that when FALSE is selected, AUCdiff will be meaningless).

- args: a list of Maxent arguments.

- abs.diff : TRUE or FALSE, whether the absolute value of AUCdiff should be returned.

- threshold : the percentile of calibration records used to set the threshold to calculate the omission error rate.

- test: the same as “train” but for the evaluation records. If it is not provided, a cross validation test procedure will be used.

- group : (only if test is missing), a vector of length nrow(train) which indicates the partitions for the cross validation procedure. If omitted the presence records are assigned to partitions randomly (with a minimum of 10 records per partitions, and a maximum of 10 partitions).

- bg.group : (only if test is missing) the same as “group” but for the background points. If omitted these are assigned to partitions randomly.

- save: the path for the .csv file where the results should be saved (optional) after each Maxent run (for both the real data and the null replicates).

The function returns an object with the following slots:

@summary : a data.frame with the performance and significance of summary statistics.

@random.reps : an array with the performance statistics for the null replicates.

Note: when a save file path is not provided, the results of this function should be saved to a R object for further analyses.

The function code:

```
mxt.nulltest = function(train, bg, n = 9, proj=bg, c.proj=TRUE, args =
"addsamplestobackground", threshold = .1, abs.diff=TRUE, test=NULL, group=NULL,
bg.group=NULL,save=NULL) {
require(ROCR) # the library 'ROCR' has to be previously installed
# the following sub-function runs the Maxent models and calculates the evaluation statistics
  f.real= function(x, p, args, proj, c.proj, bg, train, test, abs.diff, threshold) {
    tmpdir=paste(tempdir(),runif(1,0,1),sep="/")
    dir.create(tmpdir, showWarnings = TRUE, recursive = FALSE)
    mod=maxent(x,p, args=args,path=tmpdir)
    test.bb = predict(mod,proj)
    if(c.proj==TRUE) train.bb=test.bb else train.bb=predict(mod,bg)
    testpp = predict(mod,test)
    trainpp = predict(mod,train)
    combinedtest = c(testpp, test.bb)
    labeltest = c(rep(1,length(testpp)), rep(0,length(test.bb)))
    combinedtrain = c(trainpp, train.bb)
    labeltrain = c(rep(1,length(trainpp)), rep(0,length(train.bb)))
    predtest =  prediction(combinedtest, labeltest)
    predtrain =  prediction(combinedtrain, labeltrain)
    AUC= function(x) {performance(x, "auc")@y.values[[1]]}
    AUCtest = AUC(predtest)
    AUCtrain = AUC(predtrain)
    AUCdiff= AUCtrain - AUCtest
    if(abs.diff==FALSE) AUCdiff=AUCdiff else AUCdiff=abs(AUCdiff)
    r=length(testpp[testpp<quantile(trainpp,threshold)])
    t=length(testpp)
```

```
    a=length(test.bb[test.bb>=quantile(trainpp,threshold)])/length(test.bb)
    bintest=binom.test(t-r,t,p=a,alternative = "g")
    OR_binom_p=bintest$p.value
    OR= r/t
    stats= c(AUCtrain, AUCtest, AUCdiff, OR, OR_binom_p)
    unlink(tmpdir,recursive=T,force=T)
    return(stats)}
# the rest of the function iterates the sub-function above for each of the real and null models; it
prints on screen and (optionally) saves the results after each iteration.
  if(missing(test)) {
    if(missing(group)) {
      if(nrow(train)<100) {k=trunc(nrow(train)/10)} else {k=10}
      group=kfold(train,k)} else {group=group
                                        k=length(unique(group))}
    if(missing(bg.group)) bg.group=kfold(bg, length(unique(group))) else bg.group=bg.group
    lbl=vector()
    for(i in 1:k) {lbl[[i]]=paste("Real",i,sep="_")}
    a=k-1
    for(i in 1:k) {for(j in 1:n) {lbl[[a+i+j]]=paste("Null",i,j,sep="_")}
                    a=a+n-1}
    res = array(,dim=c(k*(n+1)+5,5), dimnames = list(c(lbl, "Mean_Real", "Mean_Null",
"Std.Dev_Null", "Z_score", "p_value"), c("AUCtrain", "AUCtest", "AUCdiff", "OR",
"OR_binom_p")))
    cat("Replicate",colnames(res),"  % Completion","\n", sep ="\t")
    null = array(,dim=c(n,5,k), dimnames = list(NULL, c("AUCtrain", "AUCtest", "AUCdiff", "OR",
"OR_binom_p"), NULL))
```

```
    a=k-1
    for (i in 1:k) {
      k.train = train[group != i,]
      k.test = train[group == i,]
      k.bg = bg[bg.group != i,]
      x=rbind(k.train, k.bg)
      p=c(rep(1,nrow(k.train)),rep(0,nrow(k.bg)))
      reps.x= list()
      reps =list()
      for (j in 1:n) {
        s = sample(nrow(k.bg),nrow(k.train))
        reps[[j]]=k.bg[s,]
        if(any(args=="noaddsamplestobackground")) {
          reps.x[[j]] = rbind(k.bg[s,], k.bg)
          reps.p = p} else {
            reps.x[[j]] = rbind(k.bg[s,], k.bg[-s,])
            reps.p = c(rep(1,length(s)),rep(0,(nrow(k.bg)-length(s))))}
      }
      res[i,]=f.real(x, p, args, proj, c.proj, bg, train=k.train, test=k.test, abs.diff, threshold)
      cat(rownames(res)[i],res[i,],"  ",round((i*(n+1)-n)/((n+1)*k+1)*100,digits=2),"%","\n", sep
="\t")
      for (j in 1:n) {
        null[j,,i]=f.real(reps.x[[j]],p= reps.p, args, proj, c.proj, bg, train=reps[[j]], test=k.test, abs.diff,
threshold)
        res[a+i+j,]=null[j,,i]
```

```
        cat(rownames(res)[a+i+j],res[a+i+j,],"  ",round((i*(n+1)-
n+j)/((n+1)*k+1)*100,digits=2),"%","\n", sep ="\t")
        if(!missing(save)) write.csv(res,save)
      }
      a=a+n-1}
    res[k*(n+1)+1,]=apply(res[1:k,],2,mean)
    means.k.null = rowMeans(null, dims=2)
    res[k*(n+1)+2,]=colMeans(means.k.null)
    res[k*(n+1)+3,]= apply(means.k.null,2,sd)
    res[k*(n+1)+4,]=(res[k*(n+1)+1,]-res[k*(n+1)+2,])/res[k*(n+1)+3,]
    res[k*(n+1)+5,1:2]=(1-pnorm(res[k*(n+1)+4,1:2]))
    res[k*(n+1)+5,3:5]=(pnorm(res[k*(n+1)+4,3:5]))
    cat(rownames(res)[k*(n+1)+5],res[k*(n+1)+5,],"  100 %","\n", sep ="\t")
    real=res[-((k+1):(k*(n+1))),]
 } else {
    x=rbind(train, bg)
    p=c(rep(1,nrow(train)),rep(0,nrow(bg)))
    reps.x= list()
    reps =list()
    for (i in 1:n) {
      s = sample(nrow(bg),nrow(train))
      reps[[i]]=bg[s,]
      if(any(args=="noaddsamplestobackground")) {
        reps.x[[i]] = rbind(bg[s,], bg)
        reps.p = p} else {
          reps.x[[i]] = rbind(bg[s,], bg[-s,])
```

```
        reps.p = c(rep(1,length(s)),rep(0,(nrow(bg)-length(s))))
      }
    }
    lbl=vector()
    for (i in 1:n) {lbl[[i]]=paste("Null",i,sep="_")}
   res = array(,dim=c(n+5,5), dimnames = list(c("Real", lbl, "Mean_Null", "Std.Dev_Null",
"Z_score", "p_value"), c("AUCtrain", "AUCtest", "AUCdiff", "OR", "OR_binom_p")))
    cat("Replicate",colnames(res),"  % Completion","\n", sep ="\t")
    res[1,]=f.real(x, p, args, proj, c.proj, bg, train=train, test=test, abs.diff, threshold)
    cat(rownames(res)[1],res[1,],"  1 %","\n", sep ="\t")
    for(i in 1:n) {res[i+1,]= f.real(reps.x[[i]], p= reps.p, args, proj, c.proj, bg, train=reps[[i]],
test=test, abs.diff, threshold)
                cat(rownames(res)[i+1],res[i+1,],"  ", round((i+1)/(n+2)*100,digits=2), "%", "\n",
sep ="\t")
                if(!missing(save)) write.csv(res,save)
    }
    null=res[2:(n+1),]
    res[n+2,]=apply(null,2,mean)
    res[n+3,]=apply(null,2,sd)
    res[n+4,]=(res[1,]-res[n+2,])/res[n+3,]
    res[n+5,1:2]=1-pnorm(res[n+4,1:2])
    res[n+5,3:5]=pnorm(res[n+4,3:5])
    cat(rownames(res)[n+5],res[n+5,],"  100 %","\n", sep ="\t")
    real=res[-(2:(n+1)),]
  }
  if(!missing(save)) write.csv(res,save)
```

```
  Results<- setClass("Results", representation(summary = "data.frame", random.reps = "array"))
  res=new("Results")
  res@summary = data.frame(real)
  res@random.reps = null
  return(res)}
```

2) A script to run in parallel several models with varying levels of complexity (i.e. varying the types of feature classes and the value of the regularization multiplier used in Maxent; note: these settings are just provided as an example and do not correspond to the ones described in the main text):

```
# Create a list of maxent arguments corresponding to 42 different settings:
a="addsamplestobackground"
b="noautofeature"
h="nohinge"
t="nothreshold"
p="noproduct"
LQ=lapply(seq(.5,3,.5), function(x) {c(a,b,p,t,h,paste("betamultiplier",x,sep="="))})
LQP=lapply(seq(.5,3,.5), function(x) {c(a,b,t,h,paste("betamultiplier",x,sep="="))})
LQT=lapply(seq(.5,3,.5), function(x) {c(a,b,p,h,paste("betamultiplier",x,sep="="))})
LQH=lapply(seq(.5,3,.5), function(x) {c(a,b,p,t,paste("betamultiplier",x,sep="="))})
LQPT=lapply(seq(.5,3,.5), function(x) {c(a,b,h,paste("betamultiplier",x,sep="="))})
LQPH=lapply(seq(.5,3,.5), function(x) {c(a,b,t,paste("betamultiplier",x,sep="="))})
LQTH=lapply(seq(.5,3,.5), function(x) {c(a,b,t,h,paste("betamultiplier",x,sep="="))})
LQPTH=lapply(seq(.5,3,.5), function(x) {c(a,b,paste("betamultiplier",x,sep="="))})
args=c(LQ,LQP,LQT,LQH,LQPT,LQPH,LQTH,LQPTH)
# Create a list of the files where the results for each combination of settings will be saved
(replace “your_path/your_file” by the desired path and file name):
```

```
LQs=lapply(seq(.5,3,.5),function(x) {paste("your_path/your_file_LQ",x,".csv",sep="")})
LQPs=lapply(seq(.5,3,.5),function(x) {paste("your_path/your_file_LQP",x,".csv",sep="")})
LQTs=lapply(seq(.5,3,.5),function(x) {paste("your_path/your_file_LQT",x,".csv",sep="")})
LQHs=lapply(seq(.5,3,.5),function(x) {paste("your_path/your_file_LQH",x,".csv",sep="")})
LQPTs=lapply(seq(.5,3,.5),function(x) {paste("your_path/your_file_LQPT",x,".csv",sep="")})
LQPHs=lapply(seq(.5,3,.5),function(x) {paste("your_path/your_file_LQPH",x,".csv",sep="")})
LQTHs=lapply(seq(.5,3,.5),function(x) {paste("your_path/your_file_LQTH",x,".csv",sep="")})
LQPTHs=lapply(seq(.5,3,.5),function(x) {paste("your_path/your_file_LQPTH",x,".csv",sep="")})
save=c(LQs,LQPs,LQTs,LQHs,LQPTs,LQPHs,LQTHs,LQPTHs)
# Load the following libraries required for parallel execution:
library(foreach)
library(doSNOW)
# Make a cluster corresponding to how many parallel versions of R should be run (avoid using
every single core in your computer):
cl <- makeCluster(6, "SOCK")
registerDoSNOW(cl)
# Run the foreach loop in parallel:
maxent.runs <- foreach(i = 1:length(args), .packages = c("dismo", "rJava","ROCR")) %dopar% {
mxt.nulltest(train, bg, n = 1000, proj=bg, c.proj=TRUE, args = args[[i]], group=group,
bg.group=bg.group,save=save[[i]])}
stopCluster(cl)
# The results should have been saved to the specified files but they can also be viewed in R, or
as a summary with the following lines of code:
summary=lapply(maxent.runs, function(x) x@summary)
names(summary)=save
##END
```