Exercise 9: Model Tuning

Adam B. Smith, Danielle Svehla, & Camilo Sanín

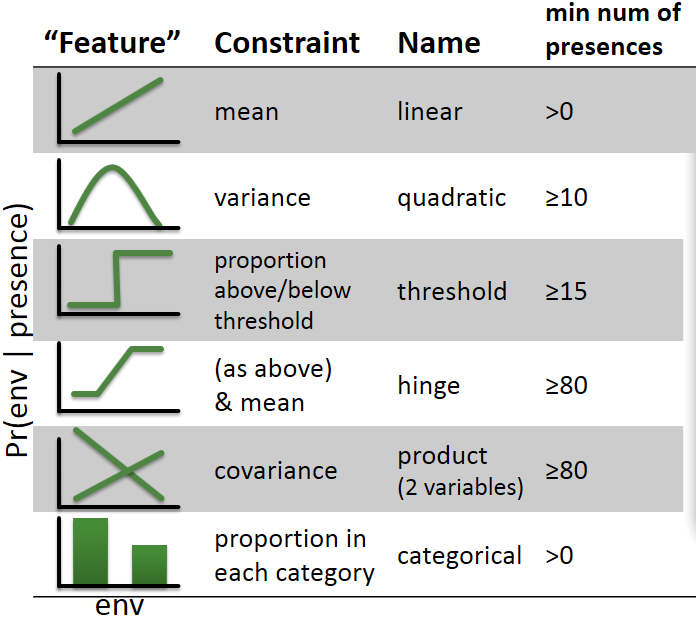
July 28, 2017

Maxent includes its own "tuning" procedures to help alleviate over-fitting to data and account for the "correct" amount of complexity in the modeled response. While generally robust, these procedures were refined using a particular (though large) data set. So the model tuning done behind the scenes by Maxent may fit most circumstances fairly well but aren't guaranteed to work well for any particular species. Here we will explore

1. Tuning Maxent by selection of **feature functions**
2. Tuning Maxent's **regularization parameter** using AICc

# Feature function selection

During the fitting procedure Maxent uses several "feature" function to capture the shape of the species-environment relationship. Features are akin the single terms in a polynomial regression. For example, there are linear features, quadratic features, and two-way interaction features. There are is also a step-function feature and a "hinge" feature which models either a non-changing response followed by a linear increase/decrease or a linear increase/decrease followed by a non-changing response. Finally, there is a categorical feature for categorical variables. By default Maxent will attempt to use all but threshold features (thresholds are special cases of hinges), though the actual mix of functions depends on the number of presences available to the model.



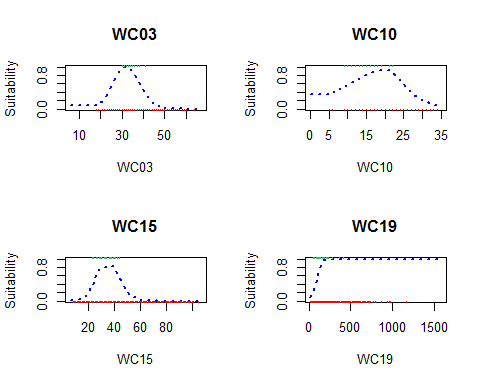
Feature functions

Let's look at the effect these features have on the estimate of environmental suitability. **Response functions** show the change in estimated environmental suitability as the variable of interest is changed while the others are held constant. We'll use the median value across the species' presences as the "constant" value and vary the focal variable from its minimum to its maximum across the study region. We'll be using the target background model from Exercise 8 to make predictions.

# get min/max value of each predictor across study region  
minPred <- minValue(climate)  
maxPred <- maxValue(climate)  
names(minPred) <- names(maxPred) <- names(climate)  
  
# get median value of each predictor across species' thinned presences  
medianPred <- apply(records[ , predictors], 2, median)  
  
# make data frame with median value of each predictor  
env <- as.data.frame(medianPred)  
env <- t(env)  
env <- env[rep(1, 100), ]  
row.names(env) <- 1:nrow(env)  
head(env)

## WC03 WC10 WC15 WC19  
## 1 36 15 36 135  
## 2 36 15 36 135  
## 3 36 15 36 135  
## 4 36 15 36 135  
## 5 36 15 36 135  
## 6 36 15 36 135

## calculate response function for each predictor  
par(mfrow=c(2, 2))  
  
# for each predictor...  
for (pred in predictors) {  
  
 # make copy of data frame  
 thisEnv <- env  
   
 # now vary focal predictor from min to max value...   
 # all other predictors keep median value  
 thisEnv[ , pred] <- seq(minPred[pred], maxPred[pred], length.out=100)  
   
 # make prediction using this data frame  
 prediction <- predict(targetBgModel, thisEnv, type='cloglog')  
   
 # plot  
 plot(x=thisEnv[ , pred], y=prediction, ylim=c(0, 1), xlab=pred,  
 ylab='Suitability', main=pred, type='l', col='blue',  
 lty='dotted', lwd=2)  
   
 # add species' presences (top rug)  
 rug(records[ , pred], side=3, col='mediumseagreen')  
   
 # add background sites (bottom rug)  
 rug(targetBg[ , pred], side=1, col='red')  
   
}



Each plot shows environmental suitability as that particular variable is varied. The red rug at the bottom shows the distribution of target background sites and the green rug at the top the presence sites. Recall that the predicted response is proportional to the ratio of the density of the presences to the background sites at a given environmental condition.

## Reflection

1. Is this what you would expect for a species' response to environmental conditions? Would you have expected smoother or rougher responses?
2. For each variable, do you expect the species to have an overall increasing, decreasing, or unimodal (or multi-modal) response?
3. Why do some of the responses seem asymptotic? Is this what you would expect? What is happening to the data (presences and background) as the asymptote is approached?
4. Of special note, what is happening with the response to WC10 at the low end (mean temperature of the hottest quarter--i.e., summer temperature)? Is this reasonable? How could you remedy this?

As you interpret the responses keep in mind that we kept all other variables at their median value. If there are interactions between variables then the response here may not be indicative of the overall response of the species to that variable.

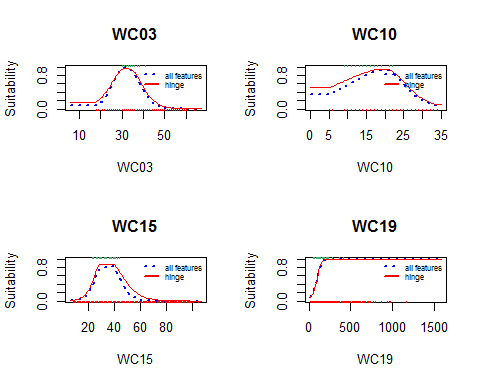
## Feature selection

The easiest way to affect feature selection in Maxent is to turn them off or on. By default they're all on (except threshold features, which are a special case of hinge features) provided sample size is adequate. In our case we have 29 presences, so the model is using linear, quadratic, and hinge features. The effect of the linear and quadratic terms is obvious in some of the response curves, but the hinge effect is somewhat hidden. Let's train a hinge-only model so we can see hinge-type responses better.

# make training data frame with predictors and vector of 1/0 for presence/background  
trainData <- rbind(  
 records[ , predictors],  
 targetBg[ , predictors]  
)  
  
presBg <- c(rep(1, nrow(records)), rep(0, nrow(targetBg)))  
  
# create output directory for model object and rasters  
dir.create('./Models/Model 07 Model Tuning - Feature Selection', recursive=TRUE, showWarnings=FALSE)  
  
# smooth model using only hinge ("h") features  
f <- maxnet.formula(p=as.vector(presBg), data=trainData, classes='h')  
hingeModel <- maxnet(p=presBg, data=trainData, f=f)  
  
# save model  
save(hingeModel,  
 file='./Models/Model 07 Model Tuning - Feature Selection/Model - Hinge Features.Rdata',  
 compress=TRUE)

Now, let's re-examine the response functions of the original target background model (with all features), the linear/quadratic model, and the hinge model.

par(mfrow=c(2, 2))  
  
# for each predictor  
for (pred in predictors) {  
  
 # make copy of data frame  
 thisEnv <- env  
   
 # now vary focal predictor from min to max value... all other predictors keep median value  
 thisEnv[ , pred] <- seq(minPred[pred], maxPred[pred], length.out=100)  
   
 # make prediction using this data frame  
 predictionTargetBg <- predict(targetBgModel, thisEnv, type='cloglog')  
 predictionHinge <- predict(hingeModel, thisEnv, type='cloglog')  
   
 # plot  
 plot(x=thisEnv[ , pred],  
 y=predictionTargetBg,  
 ylim=c(0, 1),  
 xlab=pred,  
 ylab='Suitability',  
 main=pred,  
 type='l',  
 col='blue',  
 lty='dotted',  
 lwd=2  
 )  
   
 lines(x=thisEnv[ , pred], y=predictionHinge, col='red', lwd=1,  
 lty='solid')  
   
 legend('topright',  
 legend=c('all features', 'hinge'),  
 lty=c('dotted', 'solid'),  
 col=c('blue', 'red'),  
 lwd=2,  
 cex=0.6,  
 bty='n'  
 )  
   
 # add species' presences (top rug)  
 rug(records[ , pred], side=3, col='mediumseagreen')  
   
 # add background sites (bottom rug)  
 rug(targetBg[ , pred], side=1, col='red')  
   
}



The effect of hinges is easiest to see in the response to WC15 where there is a slight "cliff" to the right side of the hump. In this case, and in all the other cases, the response is generated by pasting together multiple hinges to simulate a smooth response.

You can experiment with other features to see how they affect the estimated responses to the environment. To do so change the classes='~~~~~' argument in the first line in this code:

f <- maxnet.formula(p=as.vector(presBg), data=trainData, classes='~~~')  
newModel <- maxnet(p=presBg, data=trainData, f=f)

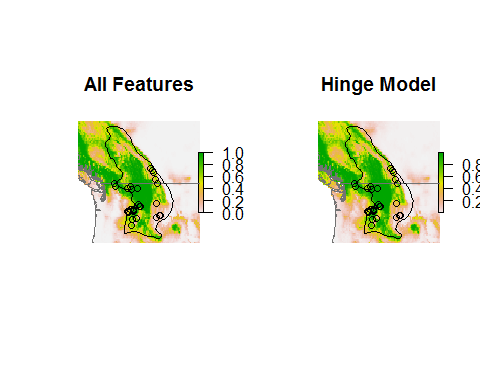
You can replace ~~~ with any combination of: \* "l" (linear) \* "q" (quadratic) \* "p" (product) \* "h" (hinge) \* "t" (threshold) \* "lpqht" (all features) \* "default" (linear, quadratic, product, and hinge features). If you use any combination of "lpqht" maxent will use that feature type even if the number of presences is fewer than the threshold normally required for that feature. Also, if any of your predictors are categorical (factors in R), then maxnet() will use a categorical feature class for these predictors.

## Reflection

Do the "hinge" response curves seem more or less reasonable than the default responses?

How do the models compare? Let's write the output raster.

hingeMap <- predict(  
 climate[[predictors]],  
 hingeModel,  
 filename='./Models/Model 07 Model Tuning - Feature Selection/maxentPrediction1970to2000',   
 format='GTiff',  
 overwrite=TRUE,  
 type='cloglog'  
)  
  
# plot  
par(mfrow=c(1, 2), pty='s')  
  
plot(rangeMap, main='All Features')  
plot(targetBgMap, add=TRUE)  
sp::plot(countries, add=TRUE, border='gray45')  
plot(rangeMap, add=TRUE)  
points(records$longitude, records$latitude)  
  
plot(rangeMap, main='Hinge Model')  
plot(hingeMap, add=TRUE)  
sp::plot(countries, add=TRUE, border='gray45')  
plot(rangeMap, add=TRUE)  
points(records$longitude, records$latitude)



## Reflection

What effect did choosing only hinge features have on the response functions and mapped output?

# *Beta/theta* regularization

Left to its own devices, the numerical solver used by the Maxent software would attempt to fit the training data exactly. This would likely overfit the model and make predictions to data not used for training the model worse (i.e., climate data from 2070). Hence, Maxent uses **regularization** which allows it to fit the data approximately. The overall amount of regularization is controlled by the parameter *beta* (called *theta* in some publications). The larger the value of *beta*, the smoother the model the Maxent software will produce.

## AICc-based tuning

You can calculate the Akaike Information Criterion corrected for sample size (AICc) for a Maxent model. Using AICc, you can vary the *beta* parameter and determine which value best fits the data. By default the master *beta* parameter is set to 1. We will try several values and pick the best. We'll also vary the features included in the model. The function trainMaxNet() cycles through every possible combination of features and *beta* values to find the model that has the lowest AICc.

Note that the code below speeds things along by only examining AICc for *beta* values of 0.5, 1, and 2. Warren et al. (2011) demonstrate this technique using 0.5, 1, 2, 3, ..., 20, but in in our experience most "best" beta values are <10, so to save time you could try 0.5, 1, 2, 3, ..., 10.

The procedure below uses custom code based on [Warren, D.L. and S.N. Siefert. 2011. Ecological niche modeling in Maxent: The importance of model complexity and the performance of model selection criteria. Ecological Applications 21:335-342.](http://dx.doi.org/10.1890/10-1171.1) They have produced stand-alone software, [ENMTools](http://enmtools.blogspot.com), which performs these calculations. Note that that the publication and software assume that you have to calculate AICc across the entire prediction raster, which means you would have to make prediction rasters for every value of beta you want to test. This can take a *long* time. However, Dan Warren was cited in a personal communication to [Amber Wright et al. (2014)](http://onlinelibrary.wiley.com/doi/10.1111/ddi.12257/abstract) saying that if non-random background sites are used then the likelihood can (should) be calculated across just the background sites, not all of the cells of the prediction raster. We'll use target background sites to train our model and compare it to the untuned target background model.

# create output directory for model object and rasters  
dir.create('./Models/Model 08 Model Tuning - Beta Parameter',  
 recursive=TRUE, showWarnings=FALSE)  
  
trainData <- cbind(presBg, trainData)  
  
# note: glmnet causes weird error if run more than once... restart R to run again  
tunedModel <- trainMaxNet(  
 data=trainData,  
 regMult=c(0.5, 1, 2),  
 verbose=TRUE  
)

## Calculating AICc for multipler 0.5 with features: l p q h l q h l p h l h l p q l q l p l   
## Calculating AICc for multipler 1 with features: l p q h l q h l p h l h l p q l q l p l   
## Calculating AICc for multipler 2 with features: l p q h l q h l p h l h l p q l q l p l   
##   
## regMult n classes logLik K AICc deltaAICc relLike  
## 6 0.5 29 lq -200.0549 8 423.3099 0.000000 1.000000e+00  
## 10 1.0 29 lqh -194.2696 11 426.0685 2.758635 2.517503e-01  
## 2 0.5 29 lqh -187.7677 14 433.5354 10.225501 6.019503e-03  
## 9 1.0 29 lpqh -195.3276 12 434.1552 10.845350 4.415320e-03  
## 18 2.0 29 lqh -203.6429 9 434.7595 11.449562 3.264068e-03  
## 14 1.0 29 lq -208.2306 7 435.7946 12.484675 1.945303e-03  
## 20 2.0 29 lh -204.8549 9 437.1835 13.873639 9.713540e-04  
## 17 2.0 29 lpqh -205.5745 9 438.6227 15.312787 4.730102e-04  
## 12 1.0 29 lh -194.9740 13 440.2147 16.904796 2.133881e-04  
## 5 0.5 29 lpq -207.2040 9 441.8817 18.571761 9.272423e-05  
## 4 0.5 29 lh -187.9062 15 442.7355 19.425581 6.050464e-05  
## 11 1.0 29 lph -196.4227 13 443.1121 19.802243 5.011843e-05  
## 1 0.5 29 lpqh -188.4702 15 443.8634 20.553545 3.442345e-05  
## 3 0.5 29 lph -188.7298 15 444.3827 21.072799 2.655217e-05  
## 22 2.0 29 lq -216.7339 5 446.0766 22.766695 1.138348e-05  
## 13 1.0 29 lpq -214.7957 7 448.9248 25.614904 2.740276e-06  
## 19 2.0 29 lph -207.2945 11 452.1183 28.808437 5.550439e-07  
## 21 2.0 29 lpq -228.3075 6 472.4331 49.123253 2.152887e-11  
## 7 0.5 29 lp -233.2353 5 479.0793 55.769415 7.759350e-13  
## 23 2.0 29 lp -237.0498 4 483.7662 60.456302 7.448704e-14  
## 16 1.0 29 l -238.6007 3 484.1613 60.851413 6.113411e-14  
## 24 2.0 29 l -238.6562 3 484.2724 60.962504 5.783096e-14  
## 15 1.0 29 lp -236.5721 5 485.7528 62.442946 2.758586e-14  
## 8 0.5 29 l -238.5844 4 486.8356 63.525670 1.605374e-14  
## aicWeight  
## 6 7.878164e-01  
## 10 1.983330e-01  
## 2 4.742263e-03  
## 9 3.478462e-03  
## 18 2.571486e-03  
## 14 1.532542e-03  
## 20 7.652486e-04  
## 17 3.726452e-04  
## 12 1.681106e-04  
## 5 7.304967e-05  
## 4 4.766655e-05  
## 11 3.948412e-05  
## 1 2.711936e-05  
## 3 2.091823e-05  
## 22 8.968093e-06  
## 13 2.158834e-06  
## 19 4.372727e-07  
## 21 1.696080e-11  
## 7 6.112943e-13  
## 23 5.868211e-14  
## 16 4.816245e-14  
## 24 4.556018e-14  
## 15 2.173259e-14  
## 8 1.264740e-14  
##

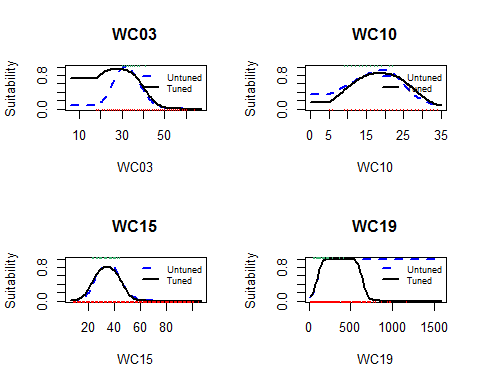
The table shows the value of AICc as *beta* is changed. Lower values of AICc are better and the table sorts rows by AICc, so the best value of *beta* is at the top. K is the number of parameters estimated in the final Maxent model.

The function also automatically returned a model trained with this best value. Let's save it for later use.

# get just model object from output  
tunedModel <- tunedModel$model  
  
save(tunedModel,  
 file='./Models/Model 08 Model Tuning - Beta Parameter/Model.Rdata',  
 compress=TRUE)

Now, let's look at the response functions of our smoothed, untuned target background model and the new smooth and tuned model.

par(mfrow=c(2, 2))  
  
# for each predictor  
for (pred in predictors) {  
  
 # make copy of data frame  
 thisEnv <- env  
   
 # now vary focal predictor from min to max value  
 # all other predictors keep median value  
 thisEnv[ , pred] <- seq(minPred[pred], maxPred[pred], length.out=100)  
   
 # make prediction using this data frame  
 predictionUntuned <- predict(targetBgModel, thisEnv, type='cloglog')  
 predictionTuned <- predict(tunedModel, thisEnv, type='cloglog')  
   
 # plot  
 plot(thisEnv[ , pred],  
 predictionUntuned,  
 ylim=c(0, 1),  
 xlab=pred,  
 ylab='Suitability',  
 main=pred,  
 type='l',  
 col='blue',  
 lty='dashed',  
 lwd=2  
 )  
   
 lines(x=thisEnv[ , pred], y=predictionTuned, col='black', lwd=2)  
   
 legend('topright',  
 legend=c('Untuned', 'Tuned'),  
 lty=c('dashed', 'solid'),  
 col=c('blue', 'black'),  
 lwd=2,  
 cex=0.7,  
 bty='n'  
 )  
   
 # add species' presences (top rug)  
 rug(records[ , pred], side=3, col='mediumseagreen')  
   
 # add background sites (bottom rug)  
 rug(targetBg[ , pred], side=1, col='red')  
   
}



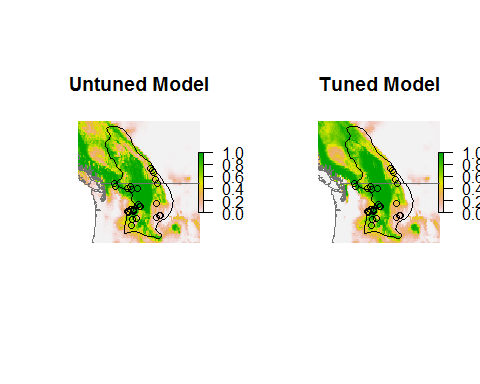
## Reflection

Do the response curves for the tuned or untuned model make more biological sense? In particular, what about the response to lower values of WC03 (isothermality--the range of annual temperature divided by the mean range of monthly temperature)? What about higher values of WC19 (winter precipitation)?

In general using higher values of *beta* causes Maxent not to use hinge and threshold responses (i.e., it creates a smoother model).

How do the responses compare? Let's write the prediction raster.

# predict to raster  
tunedMap <- predict(  
 climate[[predictors]],  
 tunedModel,  
 filename='./Models/Model 08 Model Tuning - Beta Parameter/maxentPrediction1970to2000',  
 format='GTiff', overwrite=TRUE, type='cloglog')  
  
# plot  
par(mfrow=c(1, 2), pty='s')  
  
plot(rangeMap, main='Untuned Model')  
plot(targetBgMap, add=TRUE)  
sp::plot(countries, add=TRUE, border='gray45')  
plot(rangeMap, add=TRUE)  
points(records$longitude, records$latitude)  
  
plot(rangeMap, main='Tuned Model')  
plot(tunedMap, add=TRUE)  
sp::plot(countries, add=TRUE, border='gray45')  
plot(rangeMap, add=TRUE)  
points(records$longitude, records$latitude)



## Reflection

1. How did the AICc-based tuning affect your mapped output?
2. Under what circumstances might you want only increasing/decreasing responses (linear features)? When would you expect unimodal responses?
3. It is possible to get "U"-shaped responses using Maxent. Is this reasonable? When would you expect species to respond to the environment in this way?
4. In general *beta* tuning using AICc is always recommended. Using or not using particular features depends on your situation--do you want (expect) smoother responses or particular types of responses (i.e., linear versus curved)?