Web-based material for "A climate of uncertainty: accounting for error in climate variables for species distribution models"

By Jakub Stoklosa, Christopher Daly, Scott D. Foster,

Michael B. Ashcroft and David I. Warton

Web Appendix A: Algorithms and estimation details

A1: Monte Carlo expectation–maximization (MCEM)

We used an MCEM algorithm for estimation of errors-in-variables models with classical error. This is a variation on the traditional EM algorithm, useful for fitting models with unobserved components (random effects, latent variables, predictors measured with error, missing data), in which the expectation integral at the E-step is evaluated using Monte Carlo integration. While the MCEM algorithm (as originally proposed Wei and Tanner (1990)) performs a standard Monte Carlo integration sampling from the posterior distribution of prediction error, we used a variation of the method where we sampled from the prior distribution for prediction error and evaluated the integral using importance sampling. This had the advantages that it is simple and stable, given that the Monte Carlo estimates of prediction error do not need to be updated from one iteration to the next, only their importance weights do (which are readily available since they are proportional to the respective marginal likelihood contributions). An advantage of EM-type algorithms is that the maximisation (M-step) usually can be expressed in a simple form, in our case, as a weighted GLM of "complete data" with importance weights. Our algorithm was as follows:

- We simulate B replicate Monte Carlo values for prediction error U_i from the prior distribution $f(X_i)$, to construct replicates of X_i, using $\tilde{X}_i = w_i - U_i$.
- The weights are the observations proportional to $f(y_i \mid X_i; \beta) f(X_i)$.
- Fit a GLM with weights.

This procedure is repeated until convergence of β is met. In both the simulations and case study example, we set $B = 500$. See Web Appendix C for further details.

Importance sampling works best when the proposal distribution closely matches the posterior. In our case, there was not a lot of information in the data that could be used to estimate prediction error, so the prior seemed to work relatively well as a proposal. This was diagnosed by looking at importance weights, which should be similar in magnitude across MC samples.

Standard errors for $\hat{\beta}$ are obtained by using the Fisher's information function via the `vcov()` function in `R` – i.e. we apply `vcov()` on the final weighted GLM after convergence is met, see Web Appendix C for further details. We use these to calculate the standard errors of the linear predictor (i.e. $\sqrt{X^T Var(\beta) X}$), as plotted in Figure 7.

A2: Simulation–extrapolation (SIMEX)

Consider the set of fixed values $0 = \lambda_1 < \lambda_2 < \cdots < \lambda_K$. We generate new simulated data by adding independent errors from $U_i^* \sim \mathrm{N}(0, \lambda_k \sigma_u^2)$ for $k = 1, \ldots, K$ to the explanatory variables w_i. Note that since we increase λ_k at each succession, we add more error to the contaminated explanatory variables. For each simulated data set we fit a GLM-SDM to obtain the estimates for β. We repeat this simulation/estimation procedure B times for each succession λ_k, and then calculate the mean values for the estimates of β. These means are then plotted against the increasing λ_k values, and polynomial least squares – e.g. a quadratic curve, is fitted to the averaged, error contaminated estimates. Since the total errors-in-variables variance is $\sigma_u^2 + \lambda_k \sigma_u^2 =$

$(1 + \lambda_k)\sigma_u^2$ for the kth data set, we extrapolate to the case of no errors-in-variables (i.e. $\lambda = -1$), which yields the SIMEX estimate.

In both the simulations and case study example, we used the `R`-package `simex` (Lederer and Kuchenhoff, 2006), and the following default values for $\lambda = (0.5, 1, 1.5, 2)$ and $B = 100$, also see Web Appendix C.

Web Appendix B: Extension to spatial models

B1: Spatial autocorrelation in the response

The joint density in equation (2) assumes the response variable is independent across observations, and that errors in the explanatory variables are also independent of each other, leading to "salt-and-pepper" errors for the explanatory variables. To account for spatial autocorrelation in the response, site-specific random effects are employed in the linear predictor. Write $\mu_i = h(X_i^T\beta + \nu_i)$, where $\nu_i \sim \mathrm{N}(0, \Omega)$ are the random effects which are independent of X_i. The covariance matrix $\Omega = R(\psi)$ models the spatial autocorrelation, where $\psi = (\psi_1, \psi_2)$ is a vector of parameters associated with some fixed matrix R – e.g. $R(\psi) = \psi_1 \exp(-d/\psi_2)$, where ψ_1 is the sill, ψ_2 is the scale parameter and d are Euclidean distances, see Li, Tang and Lin (2009) for other spatial autocorrelation structures. Let $\theta = (\beta, \psi)$, then the joint PDF is given by:

$$f(y, w; \theta) = \int\int \left\{ \prod_{i=1}^{n} f(y_i \mid X_i, \nu_i; \beta) f(w_i \mid X_i) f(\nu_i; \Omega) f(X_i) \right\} \mathrm{d}\nu \; \mathrm{d}X.$$

Both ν and X are treated as missing data and we again consider the MCEM-algorithm. We followed the same procedure as given in Section A1 but now additionally simulated replicate Monte Carlo values for the random effects (from the prior distribution, $N(0, \Omega)$). These simulated random effects were then treated as offsets in the GLM fit. The parameters ψ can then be easily estimated outside of the MCEM-algorithm – e.g. using a profile likelihood approach.

In `R`, the exponential covariance structure to model the spatial autocorrelation, and the long./lat. values can be used as coordinates within the `rdist.earth()` function (from the `fields` `R`-package) to calculate the Euclidean distances. The spatial parameters ψ can then be estimated using a profile likelihood approach.

B2: Spatial errors-in-variables models

The assumption that errors in climate variables are independent is potentially reasonable when the dominant source of errors are within cell variations caused by – e.g. complex topography or climate-forcing factors operating at a scale finer that the resolution of the climate grids (Daly, 2006). However, it may be an unreasonable assumption in relatively flat and homogeneous study areas where within cell variability is low and errors are dominated by spatially structured errors in how the climate grids are interpolated or predicted. In the case study, we assume an independent structure for the prediction errors. For completeness, we discuss spatial errors-in-variables models below, which are technically much more difficult to fit.

The only analysis we are aware of which addresses spatially correlated errors-in-variables models is Li et al. (2009) who dealt with a response variable that followed a Gaussian distribution. Including spatial components to the prediction error requires an additional spatial correlation in the true explanatory variable X. Again we consider a mixed model approach – i.e. we now write $X_i = \alpha + \kappa_i + \epsilon_i$, where $\kappa_i \sim \mathrm{N}(0, \Sigma)$ are the random effects with some spatial correlation structure $\Sigma = S(\zeta)$, and $\epsilon_i \sim \mathrm{N}(0, \sigma^2_\epsilon)$ are the independent residuals. As in Section B1, the ζ is a vector of parameters associated with some fixed matrix S, and we must integrate over κ in the joint PDF.

Technically this is a more difficult model to implement, and extending this method to non-normal data, such as using a GLM framework, is a priority for future research.

References

Daly, C. (2006). Guidelines for assessing the suitability of spatial climate data sets. *International Journal of Climatology*, 26, 707-721.

Lederer, W. and Kuchenhoff, H. (2006). A short introduction to the SIMEX and MC-SIMEX. *R News* **v6.4**, pp. 26–31.

Li, Y., Tang, H. and Lin, X. (2009). Spatial linear mixed models with covariate measurement errors. *Statistica Sinica* **19**, pp. 1077–1093.

Wei, G. C. G. and Tanner, M. A. (1990). A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association* **85**, pp. 699–704.

Web Appendix C: R-code

```
################################################################################
################################################################################
## R-code containing functions for the errors-in-variables MCEM method. A
## simulation example is also given, which fit the GLM, SIMEX and MCEM for
## data with errors-in-variables.
##
## Authors: Jakub Stoklosa and David Warton (School of Mathematics and
## Statistics, The University of New South Wales, NSW 2052, Australia).
##
## Please report any problems/suggestions to Jakub Stoklosa at:
## j.stoklosa@unsw.edu.au
##
## This program is meant to be used for non-commercial purposes only.
################################################################################
################################################################################

## Function for implementing the MCEM algorithm for non-Gaussian
## data with errors-in-variables.

MCEMfit<-function(Y,W,beta,sigma.u,sigma.e,n,B,family,epsilon=0.00001)
  {
  set.seed(1000)
  options(warn=-1);
  reps<-0;
  cond<-TRUE;

  ftInit<-glm(Y~W-1,family="binomial");
  muPred<-rep(predict(ftInit,type="response"),B);

  sigma.u1<-sigma.u[1];
  sigma.e1<-sigma.e[1];

  U1_j<-rnorm(n*B,0,sd=sqrt(rep(sigma.u1,B)));
  X1_j<-rep(W[,2],B)-U1_j;
  X<-cbind(rep(1,B*n),X1_j);
```

```
1
2    while(cond)
3      {
4  ## MC and E-step.
5
6      prX<-dnorm(X1_j,0,sd=sqrt(sigma.e1));
7      bigY<-rep(Y,B);
8
9      prY<-dbinom(bigY,1,muPred);
10
11 ## M-step (updates).
12
13     bigW<-matrix(prY*prX,n,B);
14     sumW<-rep(apply(bigW,1,sum),B);
15
16     mod<-glm(bigY~X-1,weights=as.vector(bigW)/sumW,family="binomial");
17     beta.update<-coef(mod);
18     muPred<-predict(mod,type="response");
19     sigma.e.update<-wt.var(X[,2],w=as.vector(bigW)/sumW);
20
21 ## Convergence monitoring.
22
23     beta.norm<-sum((beta-beta.update)^2);
24     diff.sig_e<-abs(sigma.e.update-sigma.e1);
25
26     reps<-reps+1;   # Keeps track of number of iterations.
27
28     if(diff.sig_e<epsilon && beta.norm<epsilon)
29       {
30       cond<-FALSE;
31       print("convergence met");
32       print(reps);
33       break;
34       }
35
36 ## Update parameters.
37
```

```
     beta<-beta.update;
     sigma.e1<-sigma.e.update;
     }

    beta<-beta.update;

   values<-list(beta=beta,beta.se=sqrt(diag(vcov(mod))));
   return(values)
   }

## Simulation starts here:

library("SDMTools");
library("simex");

epsilon<-0.00001;   # Convergence threshold.
B<-500;            # No. of MC replicates.

family<-"binomial";
n<-500; # Sample size.

set.seed(1000)

sigma.e<-1; # True predictor variable variance.
x<-rnorm(n,0,sd=sqrt(sigma.e));
X<-cbind(rep(1,n),x);

sigma.u<-0.5;  # Variance of uncertainty.
w<-x+rnorm(n,0,sd=sqrt(sigma.u)); # Add the uncertainty to explanatory variables.
W<-cbind(rep(1,n),w);

sigma.u<-rep(sigma.u,n);
beta<-c(0.5,1);

mu_Y<-exp(X%*%beta)/(1+exp(X%*%beta));
Y<-rbinom(n,1,prob=mu_Y);

```

```
## GLM-SDM.

mod_naiv1<-glm(Y~w,x=TRUE,family=binomial);
mod_naiv1;
sqrt(diag(vcov(mod_naiv1)));

# Slope coef: 0.578 (0.087).

## MCEM.

start<-Sys.time();
est1<-MCEMfit(Y,W,coef(mod_naiv1),sigma.u,1,n,B,family,epsilon);
end<-Sys.time(); end-start;
est1;

# Slope coef: 0.919 (0.117)

## SIMEX

start<-Sys.time();
est2<-simex(mod_naiv1,SIMEXvariable="w",measurement.error=sqrt(sigma.u));
end<-Sys.time(); end-start;
est2;
sqrt(diag(est2$variance.jackknife));

# Slope coef: 0.809 (0.116)

```