Appendix S2: Detailed methods on climate simulation and user's guide

Teller et al., Methods in Ecology and Evolution

Preamble

This appendix is divided into two sections:

1. A user's guide for practitioners examining climate signals
2. A detailed description of the simulation approach

In the supplementary R code, there are two subsidiary folders in `Rcode/Climate/`. The subdirectory `AuthenticAnalysis/` provides example code for real data, while `ArtificialAnalysis/` provides the code for artificial data simulation and model performance routines. Some parts of these sections are redundant, so we report them in this order to reduce redundancy. Further, the simulation is inspired by what users might want to know about the performance of the FLM, Lasso, and RF methods, and so this order also makes sense heuristically.

1 Analyzing real data: user's guide

To begin, we start with a real dataset that describes growth of individuals over a series of years. The true data are comprised of 22 years of growth data from individually mapped *Artemisia tripartita* plants (from 1926 to 1957; with and daily temperature and precipitation records for the area from 1925 to 2009).

All analyses from the user's point of view are called from the main script: `/Climate/AuthenticAnalysis/AuthenticAnalysis.R`

- We import data via the script `/Climate/AuthenticAnalysis/utilities/fetchDemoData.R` and lags of the climate data are generated so that they may be used as covariates.
  - When we call the data, for simplicity, we have defined W, the intraspecific competition covariate, using an exponential competition kernel with two parameters ("alpha" and "midRings") that were extracted from the original data. Precipitation is converted from inches to mm, and temperature is converted from degrees F to C. Options are included to detrend the data by subtracting the mean monthly temperature from each reading, or to scale the data to unit variance. Neither of these options usually make a difference in the analyses.
  - The climate data is formatted into lags via two functions nested in the script:
    1. "aggLags" aggregates the data by one of the columns (months or weeks). In this function, the "segment" must be a column of the dataset. So, if one would like to aggregate quarterly, one would need to first generate a column of the data that assigns days of the year to quarters, and then one may proceed to aggregate by this function.

2. "appLags" then applies the climate lags and generates a dataframe by displacing the vector by one unit at a time for a set number of units. For 36 months of lags, this function displaces the monthly means 36 times and appends each displaced vector as a column of the dataframe.

   - Since the demographic data only have one reading per year, and the climate data still have many readings per year, we must subset the lagged dataframe for one of these readings. We have set this to June (month 6), because that is often when censuses were preformed. This means that lag zero is month six of the current year [t+1]. For some individuals in some censuses (if the census occurred in May), these lags may technically occur in the future, and thus are expected to have very little bearing on the demographic response.
   - The demographic data and the lagged climate data are merged by year (the scale at which demographic measurements are taken).
   - The data is formatted so that FLM can read it.

- Implement models and calculate sensitivities

   - Each modeling framework requires data to be handled in its own slightly different way, and so there is some reshaping of the data that occurs before models are run. In particular, FLM needs to know how many knots will be provided for the analysis. When there are plenty of data (more than 30 years, with 50 individuals per year), this should not be a problem and can be implemented as in the simulation (approximately 0.75 the number of lags when climate data are aggregated monthly). If there is a failure at this point, users can open `/Climate/AuthenticAnalyis/Utilities/FuncFLM.R` and browse to line 13 to modify the number of basis functions. Remember to save and re-source the script before continuing with the analyses.

- Results are returned

   - Each model routine returns sensitivities as described in the main text, and coefficients where available (FLM and Lasso). These results can be crudely plotted using the plotting routine at the end of the script.

Interpretation

While it is current practice to aggregate climate signals, there is no reason why the aggregate with less detail (over months or years) should better explain demographic signals. Our reasoning for comparing monthly and weekly signals was to show that FLM lets the data tell you what the right period is (given sufficient data). We argue that researchers should start by using climate data at fine temporal scales rather than aggregating a priori. Then if it seems like aggregation is justified, simpler methods might work as well or even better.

We emphasize that users should analyze these results with caution. As described in the main text, none of these methods preform particularly well when there are too few data (<20-25 years).

When there are "enough data," these analyses should be fairly resilient to changes in lag length (monthly or weekly) and historical period (here we examine 36 months into the past, however interpretation should be consistent if you analyze 24 or 12 months into the past). In addition, users should seek to explain other sources of variation. For alternative methods that seek to resolve the temporal window over which signals occur, see (van de Pol & Cockburn 2011).

2 Artificial analysis: Simulation approach

We designed artificial datasets with which to rigorously compare model performance between FLM, LASSO, and RF. Our goal is to create a simulated dataset that was similar to the data we would like to use, but for which we know the correct answer. In this section, we describe how we generate artificial datasets based on the authentic data described above. This routine is followed in detail by the R code supplement in the folder `/Climate/ArtificialAnalysis/`.

In the Rscripts folder, there are two scripts that are nearly identical in principle:

1. The script `../Climate/ArtificialAnalysis/AnalysisScript.R` runs a single iteration in the simulation for which a number of variables can be user-defined. This script runs on a typical 2015 desktop (8GB RAM) in under three minutes for lags resolved monthly, and in about six minutes for lags resolved weekly (longer, because there are many more covariates).

2. The script `../Climate/ArtificialAnalysis/AnalysisScript_loop.R` loops through a set number of artificial datasets with a user-defined set of parameters, and collects information about performance etc. This script is best implemented in parallel, but still takes a long time to run nevertheless (60 parameter combinations and 10 iterations each takes about 15 hours on a 2015-very-good-machine with 32 GB of RAM over 8 cores). Results from these simulations are stored in `../ArtificialAnalysis/SimOutput`, and results can be plotted (as the manuscript figures) using the script `../ArtificialAnalysis/AnalysisScript_loop_plotting.R`.

The artificial datasets are built to have a number of qualities that can be modified by the user and/or applied in a loop. The table below shows each of the modifiable parameters:

Table AppS2.2: User-defined parameter values

Dataset	Param. name	Symbol	Tested values	definition
Climate	rPT	ρ	[0.0,0.5]	Cross-correlation between Temp. and Precip.
Demography	beta_true	β	a vector	True coefficients for Temp. and Precip.
Demography	varFrac	f_σ	[0.05,0.50]	Relative variance
Both	nFyears	-	[10,20,30,50,100]	Number of artificial years
Both	nUnits	-	[36,156]	Number of months/weeks to estimate β

The generation of artificial dataset (in both the single run and in the loop) is conducted via the script `Climate/ArtificialAnalysis/Utilities/FetchData.R`, which follows the steps:

1. Import the authentic climate and demography data as above
2. Simulate a climate pattern (details below)
3. Simulate a demographic response (details below)
4. Merge climate and demography by year
5. Combine data in format readable to FLM and return

2.1 Simulating climate data

We estimated the real annual trend of temperature by fitting a spline to the midpoint of daily maximum temperature and daily minimum temperature, Tmid, as a function of day-of-year running from 1 to 365. We then overlay onto this an AR(1) process. We chose AR(1) parameters based on the acf of the residuals from the annual trend. The data suggest an AR coefficient of 0.84, which gives about the correct acf at a 14 day lag. After that, the acf of the real data declines more slowly than the acf of the AR model, but the correlation is so low (about 0.1) that we ignore it rather than trying to fit a more complicated model. The standard deviation of the deviations from the trend are specified by the parameter `noiseSD_T`. The value estimated from the data is 4.2.

For precipitation, we fit splines to the annual trends in (a) the daily probability of precipitation occurring, and (b) the log amount of precipitation, when it is positive. A histogram and qq-plot indicate that it's not too inaccurate to model the distribution of log precipitation (on days when precipitation occurs) as Gaussian. The fake data are generated by tossing 365 coins per year to decide when precipitation occur (based on the trend in daily probability), and generating lognormal precipitation amounts with the right mean (from the fitted spline), and user-specified standard deviation specified by `noiseSD_logP`. The value estimated from the data is 1.2.

In both cases, the script ../Utilities/FetchData.R from line 79 takes the real climate data frame (with about 70 years of data), rbinds it to itself to create a data frame with 140 years of data, and substitutes the artificial data for the real data. Longer fake data series could be generated, but 100 years is the maximum duration we consider.

To create correlation between the AR1 series, we generate them using appropriately correlated innovation processes (described below). So what does "appropriately" mean? Let x_t, y_t denote the AR1 series (with mean 0, variance 1) used to generate the artificial Temp and Precip data, so we have

$$\begin{aligned} x(t+1) &= ax(t) + \sigma_1 z_1(t) \\ y(t+1) &= by(t) + \sigma_2 z_2(t) \\ \text{with } E[z_1] = E[z_2] = 0, &Var(z_1) = Var(z_2) = 1, \sigma_1^2 = 1 - a^2, \sigma_2^2 = 1 - b^2. \end{aligned} \tag{AppS2.1}$$

The values of σ_i result in x and y both having stationary variance 1. We assume that the innovation process $Z(t) = (z_1(t), z_2(t))$ is independent and identically distributed with $Cor(z_1(t), z_2(t)) =$

$E[z_1(t)z_2(t)] = \rho$. Then, multiplying the first line of (AppS2.1) by the second, taking expectations and using stationarity,

$$E[xy] = abE[xy] + \sigma_1\sigma_2\rho.$$

$E[xy] = Cor(x, y)$ so we can solve the equation above to get

$$\rho = \frac{1 - ab}{\sqrt{(1 - a^2)(1 - b^2)}} Cor(x, y). \tag{AppS2.2}$$

Equation (AppS2.2) says that if $a = b$ (meaning that Temp and Precip have equal temporal autocorrelations) the correlation between Precip and Temp equals the correlation between their innovation processes, so it can take any value between -1 and 1. However if $a \neq b$, Precip and Temp cannot be perfectly correlated or anti-correlated (as this would imply that they have the same temporal autocorrelation); the maximum absolute cross-correlations occur when $\rho = \pm 1$. The script `FetchData.R` takes $Cor(x, y)$ as an argument named `rPT`, computes the corresponding value of ρ based on the autocorrelation coefficients (computed from the autocorrelation times), generates x and y using AppS2.1 (discarding a 25-year burn-in period to reach stationarity), and returns an error if the value of ρ is outside $[-1, 1]$.

2.2 Simulating growth data with a known response to climate

To generate artificial growth data, we begin by fitting a linear regression with no climate covariates to the real data, and then create artificial sizes at time $t + 1$ by modifying fitted values from the no-climate regression, in two steps:

1. Add a known effect, $\sum_{k=0}^{M} [T(t-k)\beta_T(k) + P(t-k)\beta_P(k)]$ where T is temperature and P is precipitation, and the sum runs up to $M = 36$ months in the past.

2. Add resampled residuals from the no-climate model, possibly scaled down by a constant.

The β_T and β_P have a pre-chosen shape. Their magnitude is adjusted so that the two FLM terms have variances that are specified fractions of growth (f_σ). The code for monthly lags where "varFrac" (f_σ) is specified by the user is depicted below:

```
nlags = ncol(datag$pcovar);
fit.noClimate <- lm(logarea.t1 ~ logarea.t0, data=datag);

# make the betas for precip
beta_year = rep(0,12); beta_year[2:5] = c(0.5,1,1,0.5);
beta_true_p = rep(beta_year,round(nlags/12)+1);
beta_true_p =beta_true_p[1:nlags];
beta_true_p[(nEffect+1):nlags]=0;

# make the betas for temp
```

```
beta_year = rep(0,12); beta_year[6:9] = -c(0.75,1,1,0.75);
beta_true_t = rep(beta_year,round(nlags/12)+1);
beta_true_t = beta_true_t[1:nlags];
beta_true_t[(nEffect+1):nlags]=0;

# make the response
response_p = (datag$pcovar)%*%beta_true_p
response_t = (datag$tcovar)%*%beta_true_t;

# scale betas to get desired target fraction of response variance
targetVar = varFrac*var(datag$logarea.t1-datag$logarea.t0);

fPrecip=0.5 # Fraction of the variance attributed to T or P
targetVar_p = fPrecip*targetVar;
targetVar_t = (1-fPrecip)*targetVar;
beta_true_t = beta_true_t*sqrt(targetVar_t/var(response_t));
beta_true_p = beta_true_p*sqrt(targetVar_p/var(response_p));

response_p = (datag$pcovar)%*%beta_true_p
response_t = (datag$tcovar)%*%beta_true_t;
response = response_p + response_t;

datag_Myclim <- datag;
datag_Myclim$logarea.t1 <- predict(fit.noClimate,data=datag) +
   sigmaFrac*sample(fit.noClimate$residuals) + response
```

2.3 Assessing model performance

We need to determine whether models can detect the true drivers of the demographic signal when the true signal is known (rather than out-of-sample predictive ability). Random Forests does not estimate a parametric model, and so we use the sensitivity of model "predict" functions to changes in the lagged covariates to indicate whether models have identified the true signal. While this sort of analysis is more quickly conducted by simply observing returned coefficients in linear frameworks (FLM and Lasso), for consistency we compare each of the three methods using this analysis. The exact definition is in the main text, but the code for the sensitivities is depicted below:

```
#For preciptiation lags:

p.orig<-as.vector(predict(modelFit, as.matrix(pptSet), type="response"))
sens<- numeric(ncol(pptSet))

for(i in 1:ncol(pptSet)){
```

```
newDat<-pptSet
newDat[,i]<-newDat[,i]+eps;
eps<-.05*mean(apply(pDat,2,sd))
p.up<-as.vector(predict(modelFit, as.matrix(newDat), type="response"))
sens[ii]=(mean(p.up)-p.orig)/eps;
}
```

We expect the sensitivities to be correlated with the true coefficients when the model is performing well. This analysis is done *post hoc* in the plotting script `.../ArtificialAnalysis/AnalysisScript_loop_plotting.R` where `x` is the vector of true coefficients, and `y` is the sensitivity of the model to the lag of interest.

```
newCor <-function(x,y){ifelse(sd(x)*sd(y)>0,cor.test(x,y)$estimate,0)}
newErr <-function(x,y){sqrt(sum((x-y)^2))/sqrt(sum((x)^2))}
```

We also examined the error of coefficient estimation for Lasso and FLM. These results are concordant with sensitivities for the linear methods, and so are not shown explicitly in the manuscript. However, these analyses are available as part of the code provided to the user.

3 References

van de Pol, M. & Cockburn, A. (2011) Identifying the critical climatic time window that affects trait expression. The American Naturalist 177, 698-707.