Imputation of missing covariates for use in modeling growth rates of juvenile Chinook salmon

Mark Scheuerell[1]
Fish Ecology Division, Northwest Fisheries Science Center, Seattle, WA

Karl Veggerby
Ocean Associates, Seattle, WA

This is version 0.19.06.06.

Background

We would like to examine temperature as a possible covariate in the analyses of the growth data. However, the temperature are not complete for all sites and times, and therefore we have to impute some of the values based upon measurements from two different sources: Team Carcass and Team Tagging. To do so, we will use the **MARSS** package, which is designed to fit autoregressive models to multivariate time series data.

Specifically, we have data from $n = 7$ streams and 1-2 sources for each stream. For temperature measured at site i from source j on day d ($T_{i,j,d}$), we can write

$$\begin{aligned} \mathbf{T}_d &= \mathbf{Z}\mathbf{x}_d + \mathbf{a} + \mathbf{v}_d \\ \mathbf{x}_d &= \mathbf{B}\mathbf{x}_{d-1} + \mathbf{C}\mathbf{c}_{d-h} + \mathbf{w}_d, \end{aligned} \tag{1}$$

where $\mathbf{T}_d$ is an $n \times 1$ vector of the $T_{i,j,d}$ and $\mathbf{x}_d$ is an $m \times 1$ vector of the true, but unobserved temperature in each stream on day d. Here, $m < n$ because we have 2 sources of data from some streams. The vectors of observation ($\mathbf{v}_d$) and process ($\mathbf{w}_d$) errors are both distributed as multivariate normal with means $\mathbf{0}$ and covariance matrices $\mathbf{R}$ and $\mathbf{Q}$, respectively.

The specific forms for $\mathbf{Z}$, $\mathbf{a}$ and $\mathbf{B}$ will be chosen initially based on visually identified shared/different characteristics among the different sites. The values in the matrix $\mathbf{C}$ determine the effect(s) of any potential, and possibly lagged, covariates contained in $\mathbf{c}_{d-h}$. Those will also be chosen after inspection of the data from the different sources

Requirements

```
## for analysis
library(MARSS)
## for plotting
library(viridisLite)
```

[1] Present address: USGS Washington Cooperative Fish and Wildlife Research Unit, School of Aquatic and Fishery Sciences, University of Washington, Seattle, WA

```
## for dir mgmt
library(here)
datadir <- here("data")
analdir <- here("analysis")
```

Data munging

We begin by loading the data file with the temperature summaries by time (rows) and location (cols).

```
## load obs covariates
cobs <- read.csv(file.path(datadir, "daily_mean_temp_2.csv"),
                 stringsAsFactors = FALSE)
## inspect data
head(cobs)
```

```
##   year doy data_source_BVA data_source_CHO data_source_ELK data_source_MAR
## 1 2003 122    team carcass    team carcass    team carcass    achord_gordy
## 2 2003 123    team carcass    team carcass    team carcass    achord_gordy
## 3 2003 124    team carcass    team carcass    team carcass    achord_gordy
## 4 2003 125    team carcass    team carcass    team carcass    achord_gordy
## 5 2003 126    team carcass    team carcass    team carcass    achord_gordy
## 6 2003 127    team carcass    team carcass    team carcass    achord_gordy
##   data_source_LAK data_source_SFS data_source_VAL daily_mean_BVA
## 1    team carcass    achord_gordy    achord_gordy             NA
## 2    team carcass    achord_gordy    achord_gordy             NA
## 3    team carcass    achord_gordy    achord_gordy             NA
## 4    team carcass    achord_gordy    achord_gordy             NA
## 5    team carcass    achord_gordy    achord_gordy             NA
## 6    team carcass    achord_gordy    achord_gordy             NA
##   daily_mean_ELK daily_mean_LAK daily_mean_MAR daily_mean_SFS
## 1             NA             NA       2.964167       3.943333
## 2             NA             NA       2.349583       3.751667
## 3             NA             NA       2.561667       3.743333
## 4             NA             NA       3.106250       3.901250
## 5             NA             NA       3.071667       3.506250
## 6             NA             NA       4.188333       4.332083
##   daily_mean_VAL daily_mean_CHO
## 1       6.193333             NA
## 2       5.201250             NA
## 3       5.589583             NA
## 4       6.169583             NA
## 5       5.454583             NA
## 6       6.882500             NA
```

Let's simplify some names in the file and round the observations to the nearest 0.01.

```
## simplify colnames
colnames(cobs) <- gsub("data_source", replacement = "s", x = colnames(cobs))
colnames(cobs) <- gsub("daily_mean", replacement = "T", x = colnames(cobs))
## site abbrevs
sites <- sort(gsub("T_", replacement = "", x = colnames(cobs)[10:16]))
## re-arrange cols
cn <- c(colnames(cobs)[1:2], paste0("s_",sites), paste0("T_",sites))
cobs <- cobs[,cn]
## simplify sources
cobs[cobs=="team carcass"] <- "car"
cobs[cobs=="achord_gordy"] <- "tag"
## round temps
cobs[,grep("T_", colnames(cobs))] <- round(cobs[,grep("T_", colnames(cobs))], 2)
```

The temperature data come from two different sources:

1. the food web group (Sanderson et al.), and

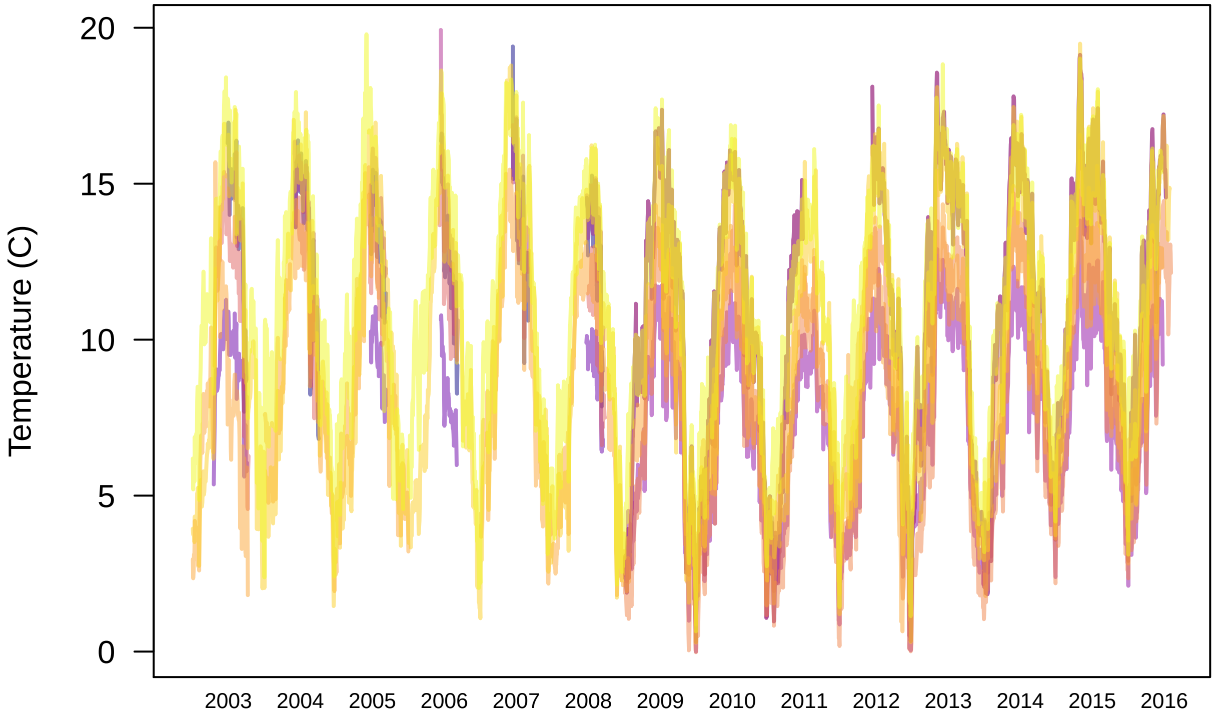
2. the juvenile tagging group (Achord, Axel et al.).

We need to split the data out by those groups.

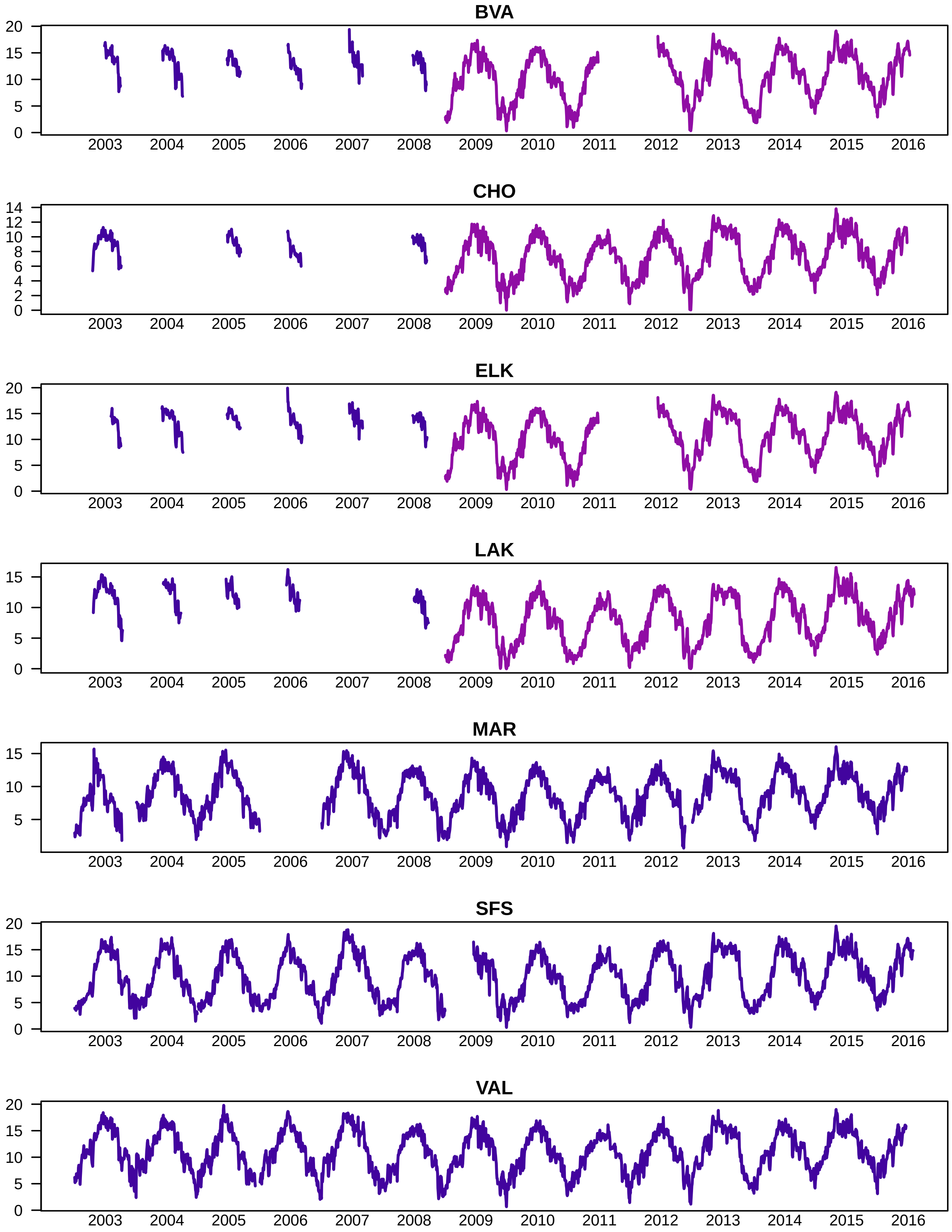
```
## empty data frames
car <- tag <- matrix(NA, nrow(cobs), length(sites),
                     dimnames = list(NULL, sites))
## measurements
vals <- cobs[,-(1:9)]
## indices of data type
i_car <- cobs[,3:9] == "car"
i_tag <- cobs[,3:9] == "tag"
## group-specific data
car[i_car] <- vals[i_car]
tag[i_tag] <- vals[i_tag]
## drop any sites with all NA's
car <- car[,apply(car, 2, function(x) !all(is.na(x)))]
colnames(car) <- paste0(colnames(car), "_car")
tag <- tag[,apply(tag, 2, function(x) !all(is.na(x)))]
colnames(tag) <- paste0(colnames(tag), "_tag")
## regroup by sites & source
cobs_m <- cbind(car, tag)
cobs_m <- cobs_m[,sort(colnames(cobs_m))]
## remove tailing rows with all NA
all_na <- apply(apply(cobs_m, 1, is.na), 2, all)
cobs_m <- cobs_m[!all_na,]
```

Plot the data

All sites together



By site


BVA
20
15
10
5
0
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
CHO
14
12
10
8
6
4
2
0
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
ELK
20
15
10
5
0
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
LAK
15
10
5
0
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
MAR
15
10
5
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
SFS
20
15
10
5
0
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
VAL
20
15
10
5
0
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016

Imputation

Based on the plots above, we will use the following forms for the vectors and matrices in Eqn (1) that relate the states to themselves and the observations to the states. Specifically, it looks as though the seasonal patterns dominate in each stream, with very little evidence for an increasing or decreasing trend. Furthermore, other than subtle changes in the overall level (mean), the streams appear to move up and down in synchrony. Thus, we will set

- $\mathbf{Z}$ equal to an $n \times 1$ column vector of 1's ($[1\ 1 \dots 1]^\top$), such that all of the data are assumed to be observations of a single overall temperature regime;
- $\mathbf{a}$ equal to an $n \times 1$ column vector where the elements are shared for data from the same stream (*e.g.*, both sets of data for Elk would get the same level, but Elk is different from Valley);
- $\mathbf{v}_d \sim \text{MVN}(\mathbf{0}, \mathbf{R})$ with $\mathbf{R}$ equal to an $n \times n$ matrix with the same variance (r) in each element of the diagonal and 0's elsewhere,

$$\mathbf{R} = \begin{bmatrix} r & 0 & \dots & 0 \\ 0 & r & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & r \end{bmatrix};$$

- $\mathbf{B}$ equal to an $m \times m$ matrix with different parameters down the diagonal and 0's elsewhere, which will allow the true temperature in each stream to follow a first-order autoregressive process,

$$\mathbf{B} = \begin{bmatrix} b_1 & 0 & \dots & 0 \\ 0 & b_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & b_m \end{bmatrix};$$

- $\mathbf{C}$ equal to an $m \times 1$ vector with unique parameters to allow for varying effects of the seasonal signal on each stream ($[C_1\ C_2 \dots C_m]^\top$);
- $\mathbf{c}$ equal to a discrete cosine wave with a period of ~189 days, such that $\mathbf{c}_t$ is a scalar; and
- $\mathbf{w}_d \sim \text{MVN}(\mathbf{0}, \mathbf{Q})$ with $\mathbf{Q}$ equal to an $m \times m$ matrix with the same variance (q) in each element of the diagonal and the same covariance (p) elsewhere,

$$\mathbf{Q} = \begin{bmatrix} q & p & \dots & p \\ p & q & \dots & p \\ \vdots & \vdots & \ddots & \vdots \\ p & p & \dots & q \end{bmatrix}.$$

MARSS modeling

Now we can translate our model from Eqn (1) into a form suitable for `MARSS()`.

Observation equation

We begin with the observation equation.

```
## number of data sets
nn <- dim(cobs_m)[2]
## number of streams
mm <- length(sites)
## empty list for model defn
mod_list <- list()
## Z
# ZZ <- matrix(1, nrow = nn, ncol = 1)
ZZ <- matrix(0, nrow = nn, ncol = mm)
for(i in seq(mm)) {
  ZZ[which(gsub("_.*", "", colnames(cobs_m)) %in% sites[i]),i] <- 1
}
mod_list$Z <- ZZ
## a
aa <- matrix(list(0), nrow = nn, ncol = 1)
aa[,] <- gsub("_.*", "", colnames(cobs_m))
mod_list$A <- aa
## R
RR <- matrix(list(0), nrow = nn, ncol = nn)
diag(RR) <- rep("r", nn)
mod_list$R <- RR
```

Process equation

And now the process (state) equation.

```
## B
BB <- matrix(list(0), nrow = mm, ncol = mm)
diag(BB) <- sites
mod_list$B <- BB
## u
mod_list$U <- matrix(0, nrow = mm, ncol = 1)
## Q
QQ <- matrix(list("p"), nrow = mm, ncol = mm)
diag(QQ) <- rep("q", mm)
mod_list$Q <- QQ
```

Fit a base model with no seasonal forcing

```
## fit base model (if not already saved)
if(!file.exists(file.path(analdir, "base_temp_model.rds"))) {
  fit_base <- MARSS(y = t(cobs_m), model = mod_list, control = list(maxit = 2000))
  ## save results to file
  saveRDS(fit_base, file.path(analdir, "base_temp_model.rds"))
}
```

Let's examine the model fits.

```
fit_base <- readRDS(file.path(analdir, "base_temp_model.rds"))
fit_base
```

```
##
## MARSS fit is
## Estimation method: kem
## Convergence test: conv.test.slope.tol = 0.5, abstol = 0.001
## WARNING: Abstol convergence only no log-log convergence.
##  maxit (=2000) reached before log-log convergence.
##  The likelihood and params might not be at the ML values.
##  Try setting control$maxit higher.
## Log-likelihood: -11030.07
## AIC: 22108.14   AICc: 22108.23
##
##        Estimate
## A.BVA    0.0193
## A.CHO    0.0126
## A.ELK    0.0161
## A.LAK   -0.0174
## A.MAR    0.1440
## A.SFS   -0.0274
## A.VAL    0.0962
## R.r      0.0418
## B.BVA    0.9876
## B.CHO    0.9820
## B.ELK    0.9879
## B.LAK    0.9855
## B.MAR    0.9841
## B.SFS    0.9880
## B.VAL    0.9879
## Q.q      0.6478
## Q.p      0.5347
## x0.X1   11.7653
## x0.X2    2.6614
## x0.X3    9.9348
## x0.X4    4.9410
## x0.X5    2.8458
## x0.X6    4.0788
## x0.X7    6.0836
## Initial states (x0) defined at t=0
##
## Standard errors have not been calculated.
## Use MARSSparamCIs to compute CIs and bias estimates.
##
## Convergence warnings
##  Warning: the  A.BVA  parameter value has not converged.
##  Warning: the  A.CHO  parameter value has not converged.
##  Warning: the  A.ELK  parameter value has not converged.
##  Warning: the  A.LAK  parameter value has not converged.
##  Warning: the  A.MAR  parameter value has not converged.
##  Warning: the  A.SFS  parameter value has not converged.
##  Warning: the  A.VAL  parameter value has not converged.
##  Warning: the  logLik  parameter value has not converged.
##  Type MARSSinfo("convergence") for more info on this warning.
```

Fit random-walk models

The estimates of the diagonal values of the matrix **B** from the above model, which control the degree of autocorrelation from one time step to another, are very close to 1. This suggests we should try a subtely different model where we fix **B** equal to the identity matrix **I**.

```
## change B in model defn
mod_list$B <- diag(mm)
## fit model
if(!file.exists(file.path(analdir, "rw_temp_model.rds"))) {
  fit_rw <- MARSS(y = t(cobs_m), model = mod_list, control = list(maxit = 2000))
  ## save results to file
  saveRDS(fit_rw, file.path(analdir, "rw_temp_model.rds"))
}
```

Let's examine the model fits.

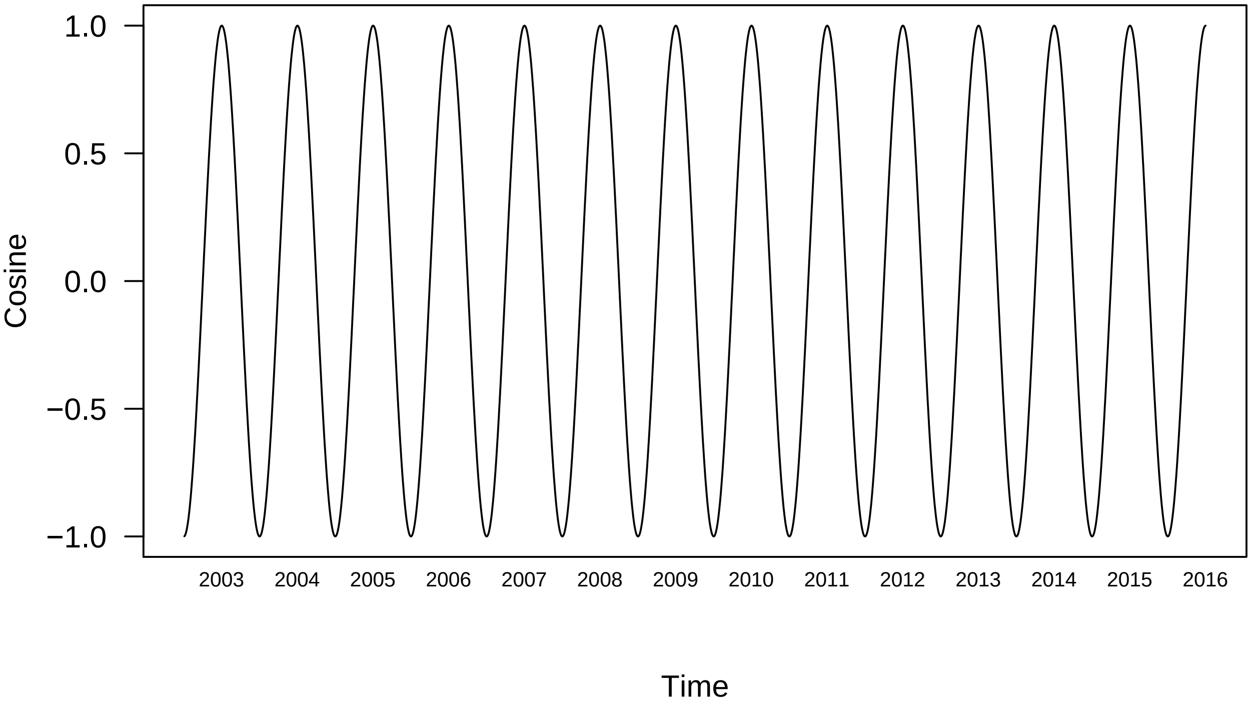
```
fit_rw <- readRDS(file.path(analdir, "rw_temp_model.rds"))
fit_rw
```

```
##
## MARSS fit is
## Estimation method: kem
## Convergence test: conv.test.slope.tol = 0.5, abstol = 0.001
## Estimation converged in 359 iterations.
## Log-likelihood: -11085.94
## AIC: 22205.88   AICc: 22205.93
##
##         Estimate
## A.BVA  5.30e-04
## A.CHO -2.09e-05
## A.ELK  4.33e-04
## A.LAK  2.12e-04
## A.MAR -6.83e-04
## A.SFS -6.44e-04
## A.VAL -7.29e-04
## R.r    4.25e-02
## Q.q    6.34e-01
## Q.p    5.21e-01
## x0.X1  7.12e+00
## x0.X2  2.35e+00
## x0.X3  6.52e+00
## x0.X4  4.04e+00
## x0.X5  2.95e+00
## x0.X6  4.00e+00
## x0.X7  6.10e+00
## Initial states (x0) defined at t=0
##
## Standard errors have not been calculated.
## Use MARSSparamCIs to compute CIs and bias estimates.
```

Interestingly, this model does not fit as well (based on an ΔAIC value of ~90 units).

Base model with seasonal signal

Now let's add a seasonal effect in the form of a discrete cosine wave. By adjusting the period correctly, we can make it reflect the observed seasonal pattern.



```
## C
mod_list$C <- matrix(sites, nrow = mm, ncol = 1)
## c (discrete sine wave)
mod_list$c <- matrix(cc, nrow = 1)
## change B back to diagonal and unequal
mod_list$B <- BB
## fit model
if(!file.exists(file.path(analdir, "seas_temp_model.rds"))) {
  fit_seas <- MARSS(y = t(cobs_m), model = mod_list, control = list(maxit = 2000))
  ## save results to file
  saveRDS(fit_seas, file.path(analdir, "seas_temp_model.rds"))
}
```

Let's examine the model fits.

```
fit_seas <- readRDS(file.path(analdir, "seas_temp_model.rds"))
fit_seas
```

```
##
## MARSS fit is
## Estimation method: kem
## Convergence test: conv.test.slope.tol = 0.5, abstol = 0.001
## WARNING: Abstol convergence only no log-log convergence.
##  maxit (=2000) reached before log-log convergence.
##  The likelihood and params might not be at the ML values.
##  Try setting control$maxit higher.
## Log-likelihood: -11017.84
## AIC: 22097.68   AICc: 22097.83
##
##        Estimate
## A.BVA  0.006249
## A.CHO  0.036693
## A.ELK  0.004676
## A.LAK  0.004775
## A.MAR  0.123677
## A.SFS  0.028047
## A.VAL  0.077150
## R.r    0.042471
## B.BVA  0.987049
## B.CHO  0.979737
## B.ELK  0.987316
## B.LAK  0.983375
## B.MAR  0.983212
## B.SFS  0.986330
## B.VAL  0.987235
## Q.q    0.649048
## Q.p    0.537438
## x0.X1 12.553679
## x0.X2  4.110780
## x0.X3 11.631246
## x0.X4  6.704446
## x0.X5  2.866517
## x0.X6  4.082966
## x0.X7  6.102067
## C.BVA -0.007137
## C.CHO  0.039879
## C.ELK -0.005121
## C.LAK  0.043494
## C.MAR  0.001482
## C.SFS  0.042708
## C.VAL  0.000213
## Initial states (x0) defined at t=0
##
## Standard errors have not been calculated.
## Use MARSSparamCIs to compute CIs and bias estimates.
##
## Convergence warnings
##  Warning: the  A.BVA  parameter value has not converged.
##  Warning: the  A.CHO  parameter value has not converged.
##  Warning: the  A.ELK  parameter value has not converged.
##  Warning: the  A.LAK  parameter value has not converged.
##  Warning: the  A.MAR  parameter value has not converged.
##  Warning: the  A.SFS  parameter value has not converged.
##  Warning: the  A.VAL  parameter value has not converged.
##  Warning: the  logLik  parameter value has not converged.
##  Type MARSSinfo("convergence") for more info on this warning.
```

De-mean the observations

The lack of convergence and low values for **a** (*i.e.*, `A.???` in the `MARSS()` output) suggests we should instead de-mean the observations, fits the model, and then add the mean back to the estimates. We'll also exclude the cosine wave for now.

```
## de-mean obs
dm_obs <- scale(cobs_m, center = TRUE, scale = FALSE)
## set a = 0
mod_list$A <- matrix(0, nrow = nn, ncol = 1)
## fit model
if(!file.exists(file.path(analdir, "dbase_temp_model.rds"))) {
  fit_dbase <- MARSS(y = t(dm_obs), model = mod_list, control = list(maxit = 2000))
  ## save results to file
  saveRDS(fit_dbase, file.path(analdir, "dbase_temp_model.rds"))
}
```

```
fit_dbase <- readRDS(file.path(analdir, "dbase_temp_model.rds"))
fit_dbase
```

```
##
## MARSS fit is
## Estimation method: kem
## Convergence test: conv.test.slope.tol = 0.5, abstol = 0.001
## WARNING: Abstol convergence only no log-log convergence.
##  maxit (=2000) reached before log-log convergence.
##  The likelihood and params might not be at the ML values.
##  Try setting control$maxit higher.
## Log-likelihood: -10844.36
## AIC: 21736.72   AICc: 21736.82
##
##       Estimate
## R.r     0.0301
## B.BVA   0.9544
## B.CHO   0.9110
## B.ELK   0.9543
## B.LAK   0.9490
## B.MAR   0.9146
## B.SFS   0.9360
## B.VAL   0.9219
## Q.q     0.6443
## Q.p     0.5140
## x0.X1  -4.0865
## x0.X2  -8.7688
## x0.X3  -5.3419
## x0.X4 -20.0786
## x0.X5  -6.0825
## x0.X6  -5.9136
## x0.X7  -4.9902
## C.BVA   0.1118
## C.CHO   0.2266
## C.ELK   0.1165
## C.LAK   0.1560
## C.MAR   0.2514
## C.SFS   0.3108
## C.VAL   0.3009
## Initial states (x0) defined at t=0
##
## Standard errors have not been calculated.
## Use MARSSparamCIs to compute CIs and bias estimates.
##
## Convergence warnings
##  Warning: the  x0.X1  parameter value has not converged.
##  Warning: the  x0.X2  parameter value has not converged.
##  Warning: the  x0.X3  parameter value has not converged.
##  Warning: the  x0.X4  parameter value has not converged.
##  Warning: the  logLik  parameter value has not converged.
##  Type MARSSinfo("convergence") for more info on this warning.
```

That didn't work as well as hoped.

Scale the observations

Now we'll try to fully scale the observations, such that times series for each location has zero mean and unit variance. In addition, we will assume all of the locations are observations of a single, regional temperature patter, but scaled accordingly at the localed level (*i.e.*, different min, max, mean).

To make this work within a location, we need to combine the multiple time series into one.

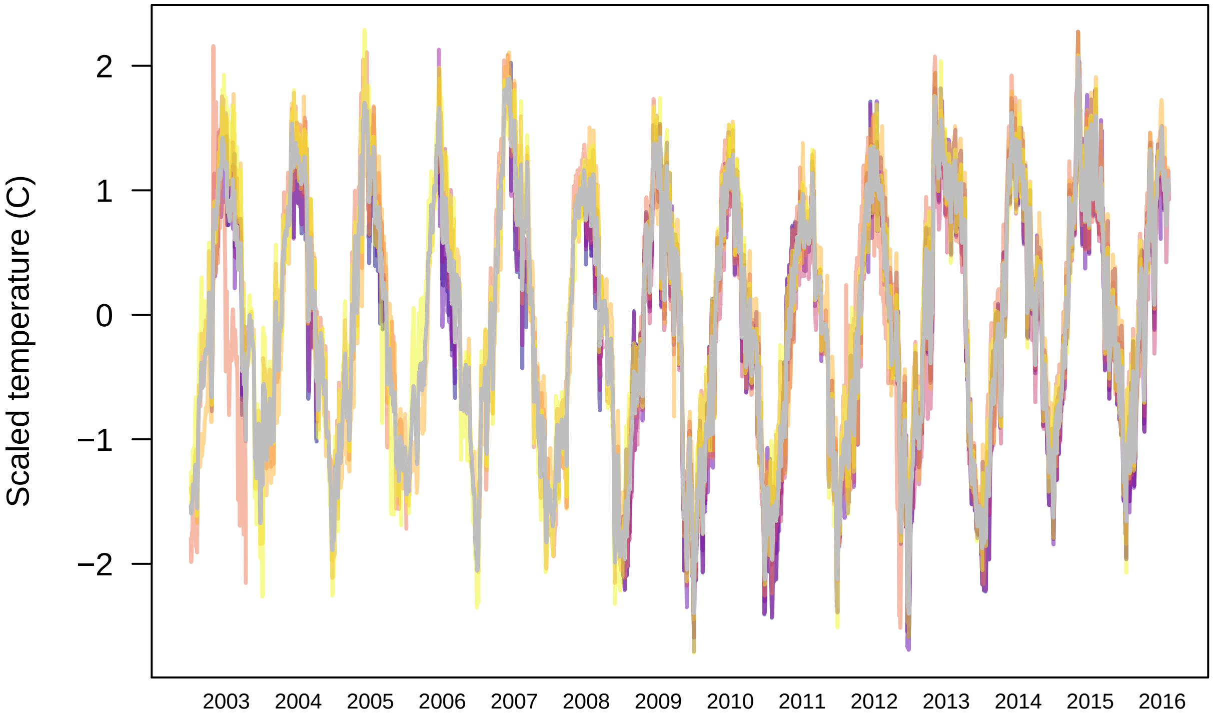
```
## means by location
```

```
scld <- matrix(NA, ncol = length(sites), nrow = dim(cobs_m)[1])
colnames(scld) <- sites
## average fitted values by site
for(i in sites) {
  ## site-specific data
  tmp <- cobs_m[, grep(i, colnames(cobs_m)), drop = FALSE]
  if(dim(tmp)[2] != 1) {
    scld[,i] <- apply(tmp, 1, mean, na.rm = TRUE)
  } else
    scld[,i] <- tmp
}

## de-mean obs
sc_obs <- scale(scld)
## all locations are obs of single state
mod_list$Z <- matrix(1, nrow = length(sites), ncol = 1)
mod_list$A <- matrix(0, nrow = length(sites), ncol = 1)
RR <- matrix(list(0), nrow = length(sites), ncol = length(sites))
diag(RR) <- rep("r", length(sites))
mod_list$R <- RR
## set scalars for state eqn
mod_list$B <- matrix("b")
mod_list$U <- matrix(0)
mod_list$Q <- matrix("q")
mod_list$C <- matrix("C")
## fit model
if(!file.exists(file.path(analdir, "sbase_temp_model.rds"))) {
  fit_sbase <- MARSS(y = t(sc_obs), model = mod_list, control = list(maxit = 2000))
  ## save results to file
  saveRDS(fit_sbase, file.path(analdir, "sbase_temp_model.rds"))
}

fit_sbase <- readRDS(file.path(analdir, "sbase_temp_model.rds"))
fit_sbase
```

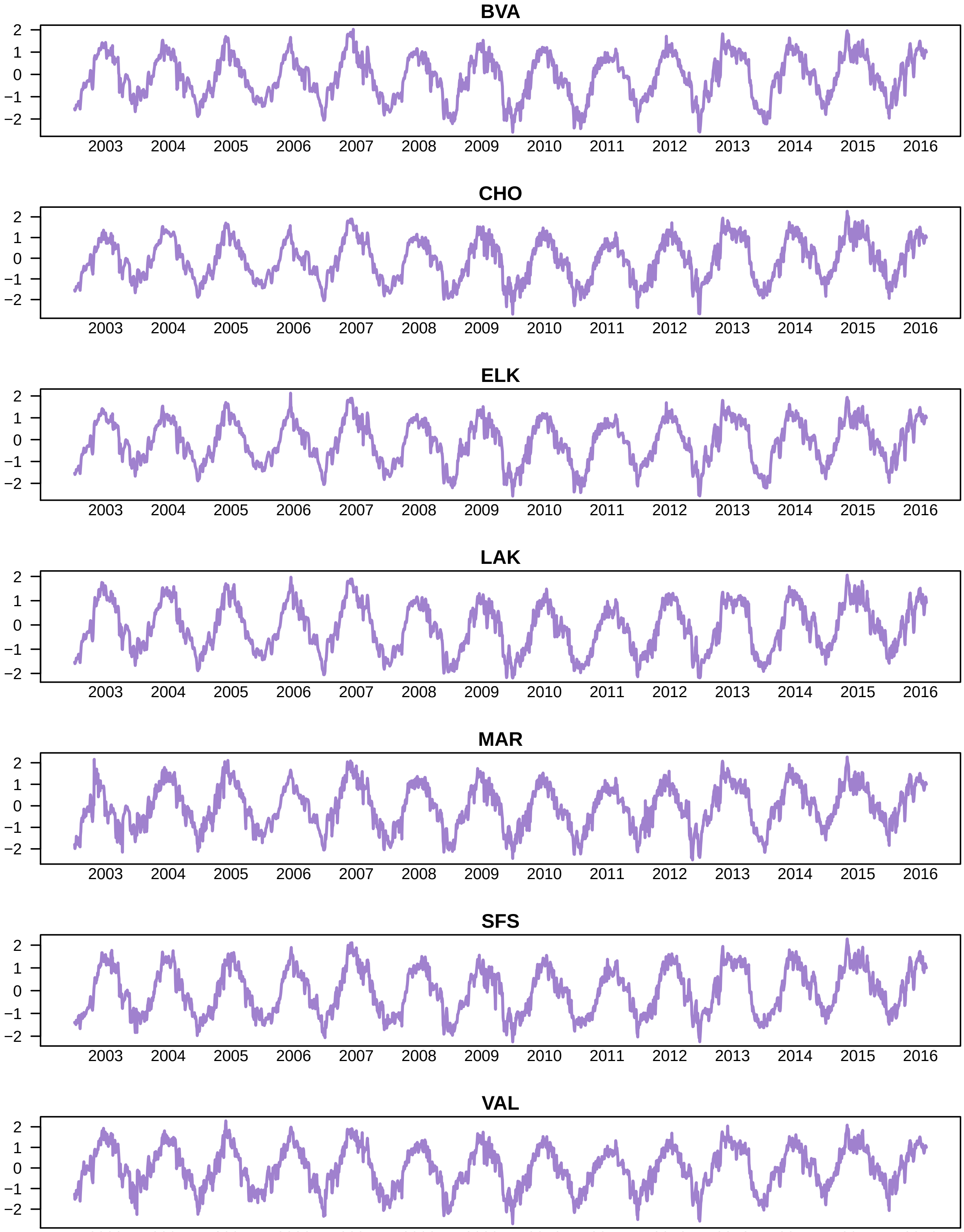
```
##
## MARSS fit is
## Estimation method: kem
## Convergence test: conv.test.slope.tol = 0.5, abstol = 0.001
## Estimation converged in 17 iterations.
## Log-likelihood: -1241.63
## AIC: 2493.261   AICc: 2493.265
##
##       Estimate
## R.r     0.0532
## B.b     0.9676
## Q.q     0.0299
## x0.x0  -1.5588
## C.C     0.0257
## Initial states (x0) defined at t=0
##
## Standard errors have not been calculated.
## Use MARSSparamCIs to compute CIs and bias estimates.
```



Estimated temperature by site

```
y_hat_Z <- t(fit_sbase$ytT)
colnames(y_hat_Z) <- sites

## plot fitted values
par(mfrow = c(length(sites), 1),
    mai = c(0.3,0.4,0.2,0.1),
    omi = c(0,0,0,0))
for(i in sites) {
  ## plot the data
  plot.ts(y_hat_Z[,i], type = "l", lty = "solid",
          lwd = 2, las = 1,
          ylab = "Scaled temperature (C)", xaxt="n", main = i,
          col = plasma(ncol(tmp), alpha = 0.5, begin = 0.1, end = 0.3))
  axis(side = 1,
       at = seq(n_days / 2, by = n_days, length.out = n_yrs),
       labels = years,
       tick = FALSE, line = -1)
}
```



BVA

CHO

ELK

LAK

MAR

SFS

VAL

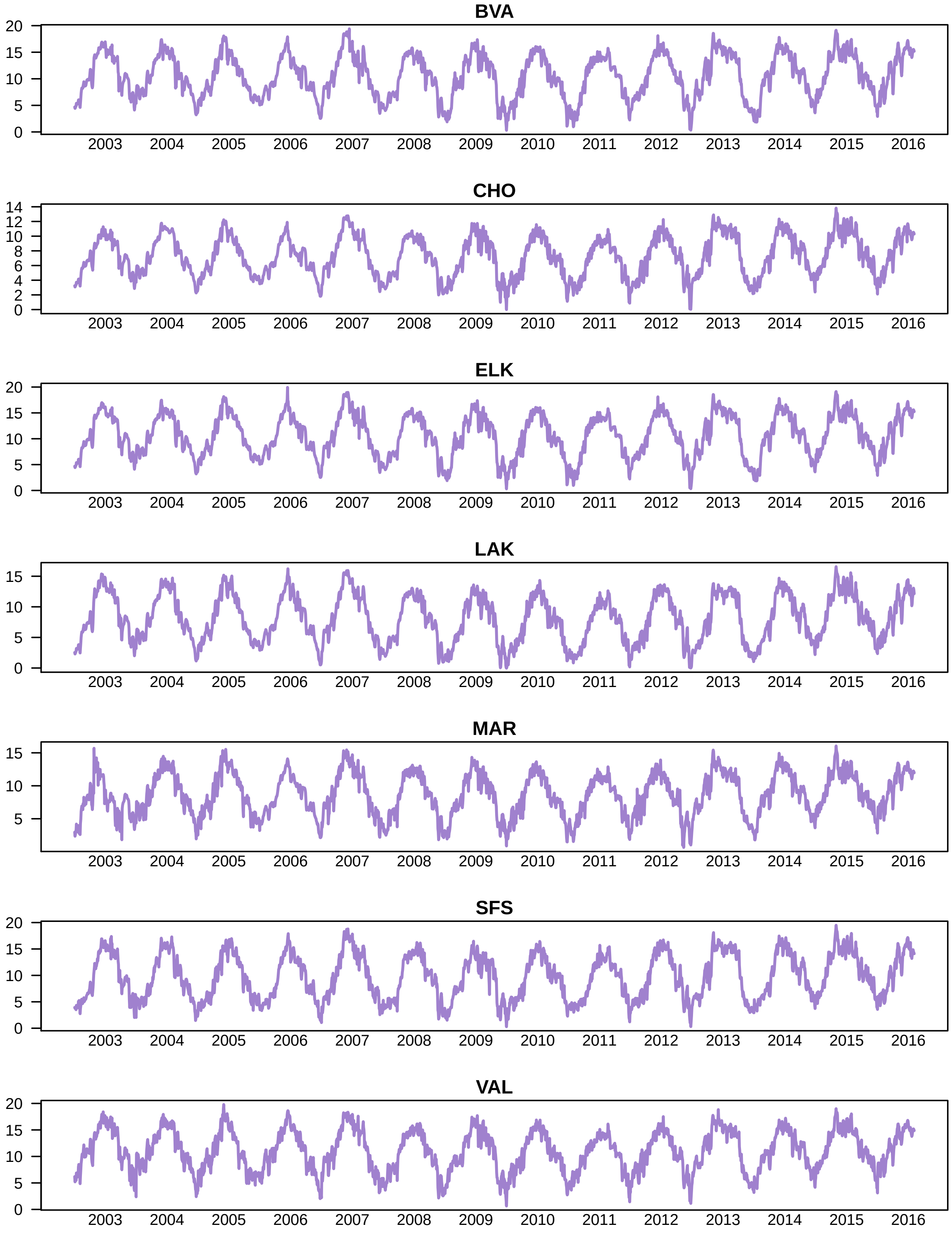
Reconstruction

The last step is to transform these z-scored values back into normal space.

```
## mean and var of orig series
men <- apply(scld, 2, mean, na.rm = TRUE)
std <- apply(scld, 2, sd, na.rm = TRUE)

y_hat <- y_hat_Z
for(i in 1:length(sites)) {
  y_hat[,i] <- y_hat_Z[,i] * std[i] + men[i]
}

## plot fitted values
par(mfrow = c(length(sites), 1),
    mai = c(0.3,0.4,0.2,0.1),
    omi = c(0,0,0,0))
for(i in sites) {
  ## plot the data
  plot.ts(y_hat[,i], type = "l", lty = "solid",
          lwd = 2, las = 1,
          ylab = "Temperature (C)", xaxt="n", main = i,
          col = plasma(ncol(tmp), alpha = 0.5, begin = 0.1, end = 0.3))
  axis(side = 1,
       at = seq(n_days / 2, by = n_days, length.out = n_yrs),
       labels = years,
       tick = FALSE, line = -1)
}
```



These values look reasonable, so we'll round them and save them as an **R** object and a csv.

```
y_hat_rnd <- round(y_hat, 2)
```

```
saveRDS(y_hat_rnd, file.path(analdir, "reconstructed_temps.rds"))
write.csv(y_hat_rnd, file = file.path(datadir, "reconstructed_temps.csv"), row.names = FALSE)
```