Supplement 01 - R code for the figures and examples in the manuscript "Sedproxy: a forward model for sediment archived climate proxies".

Andrew M. Dolman and Thomas Laepple

16 February, 2018

This file contains the R code to produce the figures and tables in the manuscript "Sedproxy: a forward model for sediment archived climate proxies", submitted to Climate of the Past Discussions.

The code is separated into chunks in an rmarkdown (supplement_01_sedproxy.Rmd) text file. The .Rmd file can be "knit" to form a PDF file (which you may be reading) containing the formatted code and resulting figures and tables (supplement_01_sedproxy.pdf).

To run the code you will need the packages loaded in the first code chunk. They are all available on CRAN, except for *sedproxy* itself, which can be installed directly from Bitbucket (https://bitbucket.org/ecus/sedproxy) with the commented out code. Alternatively, a source package for *sedproxy* is supplied as supplement 2 to the manuscript.

```r
library(dplyr)
library(tidyr)
library(ggplot2)
library(knitr)
library(egg)
library(pander)

# if (!require("devtools")) {
#   install.packages("devtools")
# }
#
# devtools::install_bitbucket("ecus/sedproxy")
library(sedproxy)

opts_chunk$set(echo=TRUE, message = FALSE, warning = FALSE, cache = TRUE,
               dpi = 300, autodep = TRUE, fig.pos="H", tidy = FALSE)
```

Table 1

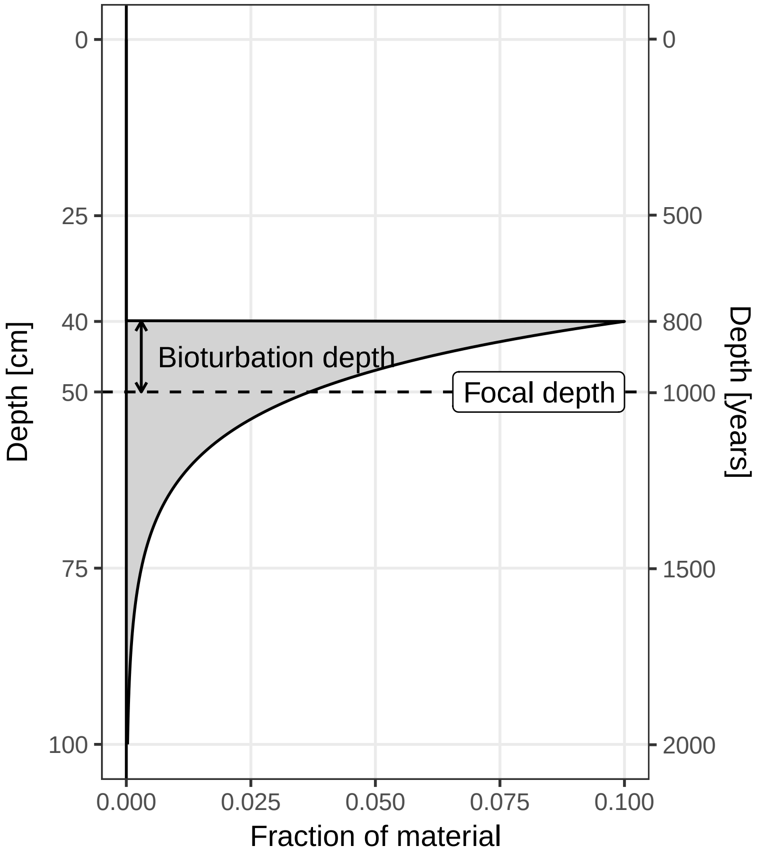
```r
tbl <- sedproxy::param.tab
tbl[is.na(tbl)] <- "..."
pander::pander(tbl, justify = c("left"), style = "multiline",
               split.tables = Inf,
               split.cells = c(23, 40, 30, 10),
               caption = "Required input data and parameters to generate a
               pseudo-proxy record with sedproxy. The final two arguments
               control the experimental design rather than the proxy record
               creation process itself.")
```

Table 1: Required input data and parameters to generate a pseudo-proxy record with sedproxy. The final two arguments control the experimental design rather than the proxy record creation process itself.

Function argument	Description	Possible sources	Default
clim.signal	Input climate signal from which a pseudo-proxy will be forward modelled.	Climate model, instrumental record.	…
timepoints	Timepoints at which to generate pseudo-proxy values.	Arbitrary, or to match an existing proxy record.	…
proxy.prod.weights	Proxy production weights provide information on seasonal and habitat (e.g. depth) differences in the amount of proxy material produced. This allows seasonal and habitat biases in the recorded climate to be modelled.	Sediment trap data or dynamic population/biogeochemical model (Fraile et al., 2008; Uitz et al., 2010)	equal for all
bio.depth	Bioturbation depth in cm, the depth down to which the sediment is mixed by burrowing organisms.	Estimated from radiocarbon or from global distribution (Teal et al., 2010).	10
sed.acc.rate	Sediment accumulation rate in cm ka^{-1}.	Sediment core age model.	50
layer.width	Width of the sediment layer in cm from which samples were taken, e.g. foraminifera were picked or alkenones were extracted.	Core sampling protocol	1
n.samples	No. of e.g. foraminifera sampled per timepoint. A single number or a vector with one value for each timepoint.	Core sampling protocol	30
meas.noise	Standard deviation of white noise added to each pseudo-proxy value.	…	0
meas.bias	Each replicate proxy time-series has a constant bias added drawn from a normal distribution with mean = 0, sd = meas.bias.	…	0
…	…	…	…
n.replicates	Number of replicate pseudo-proxy time-series to simulate from the climate signal.	…	…
smoothed.signal.res	The resolution, in years, of the smoothed (block averaged) version of the input climate signal returned for plotting purposes. If set to NA, no smoothed climate output is generated, this can speed up some simulations.	…	100

Figure 1

```
prs <- tibble(biot.depth = 10) %>%
  mutate(sed.acc.rate = 50,
         rte = 1/biot.depth,
         f.depth = 50)

dat <- tibble(Depth = seq(0, 100, by = 0.1)) %>%
  mutate(
    Prob = dexp(x = Depth-prs$f.depth+prs$biot.depth, rate = prs$rte)
         )

prs <- prs %>%
  mutate(max.prob = max(dat$Prob),
         l.prob = max.prob * 0.15)

brks <- sort(c(seq(0, max(dat$Depth), length.out = 5),
               prs$f.depth - prs$biot.depth))

p.origin <- ggplot(dat, aes(x = Depth, y = Prob)) +
  geom_ribbon(aes(ymin = 0, ymax = Prob), fill = "Lightgrey") +
  geom_line() +
  geom_vline(data = prs, aes(xintercept = f.depth), linetype = 2) +
  geom_hline(yintercept = 0) +
  coord_flip() +
  geom_segment(data = prs, aes(x = f.depth, xend = f.depth - biot.depth,
                               y = l.prob * 0.2, yend = l.prob * 0.2),
               arrow = arrow(length = unit(0.02, "npc"), ends = "both"),
               colour = "Black") +
  geom_text(dat = prs,
            aes(x = f.depth - biot.depth/2, y = l.prob * 0.1, hjust = -0.1),
            label = "Bioturbation depth", colour = "Black") +
  geom_label(dat = prs,
             aes(x = f.depth, y = max.prob, hjust = 1), label = "Focal depth") +
  scale_x_reverse("Depth [cm]", breaks = brks,
                  sec.axis = sec_axis(~./(prs$sed.acc.rate/1000),
                                      name = "Depth [years]",
                                      breaks = brks/(prs$sed.acc.rate/1000),
                                      labels = brks/(prs$sed.acc.rate/1000))) +
  theme_bw() +
  theme(aspect.ratio = sqrt(2),
        panel.grid.minor = element_blank()) +
  labs(y = expression("Fraction of material"))
p.origin
```



Example 1: A foraminiferal Mg/Ca pseudo-proxy record for sediment core MD97-2141

```
library(sedproxy)

N41.proxy.details %>%
  select(Core, Location, Lat, Lon, Proxy, Foram.sp, Reference) %>%
  kable(., caption = "Sediment core MD97-2141", digits = 2)
```

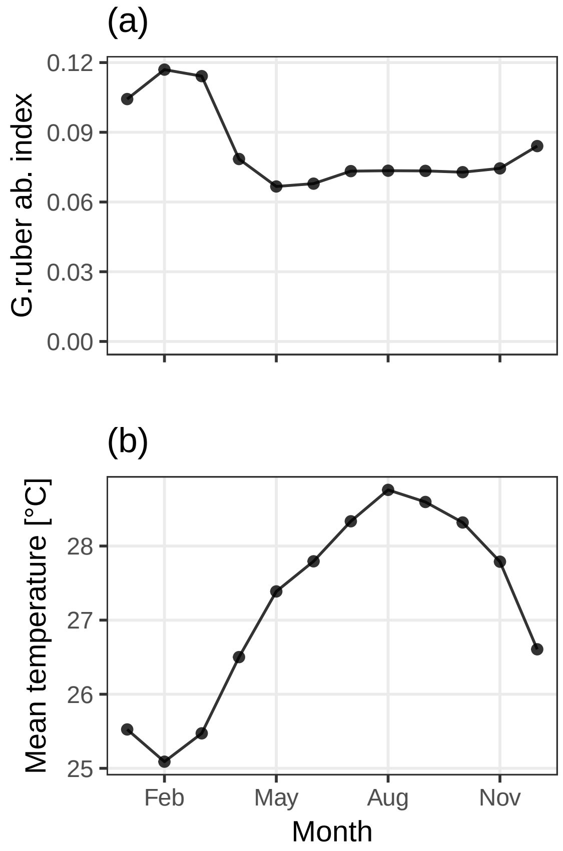
Table 2: Sediment core MD97-2141

Core	Location	Lat	Lon	Proxy	Foram.sp	Reference
MD97-2141	Sulu Sea	8.78	121.28	Mg/Ca	G. ruber	Rosenthal et al., 2003

```
# Seasonality of G.ruber
MD97_2141.ab.index <- tibble(Month = 1:12,
                             MD97_2141.ab.index = N41.G.ruber.seasonality /
                               sum(N41.G.ruber.seasonality))

p.ab <- MD97_2141.ab.index %>%
  ggplot(aes(x = Month, y = MD97_2141.ab.index)) +
  geom_line(colour = "Black", alpha = 0.8) +
  geom_point(colour = "Black", alpha = 0.8) +
  scale_x_continuous("", breaks = seq(2, 12, by = 3)) +
  scale_y_continuous("G.ruber ab. index") +
  expand_limits(y = 0) +
  theme_bw() + labs(title = "(a)") +
  theme(plot.title = element_text(hjust = 0, vjust = 1),
        panel.grid.minor = element_blank(),
        axis.text.x = element_blank())
```

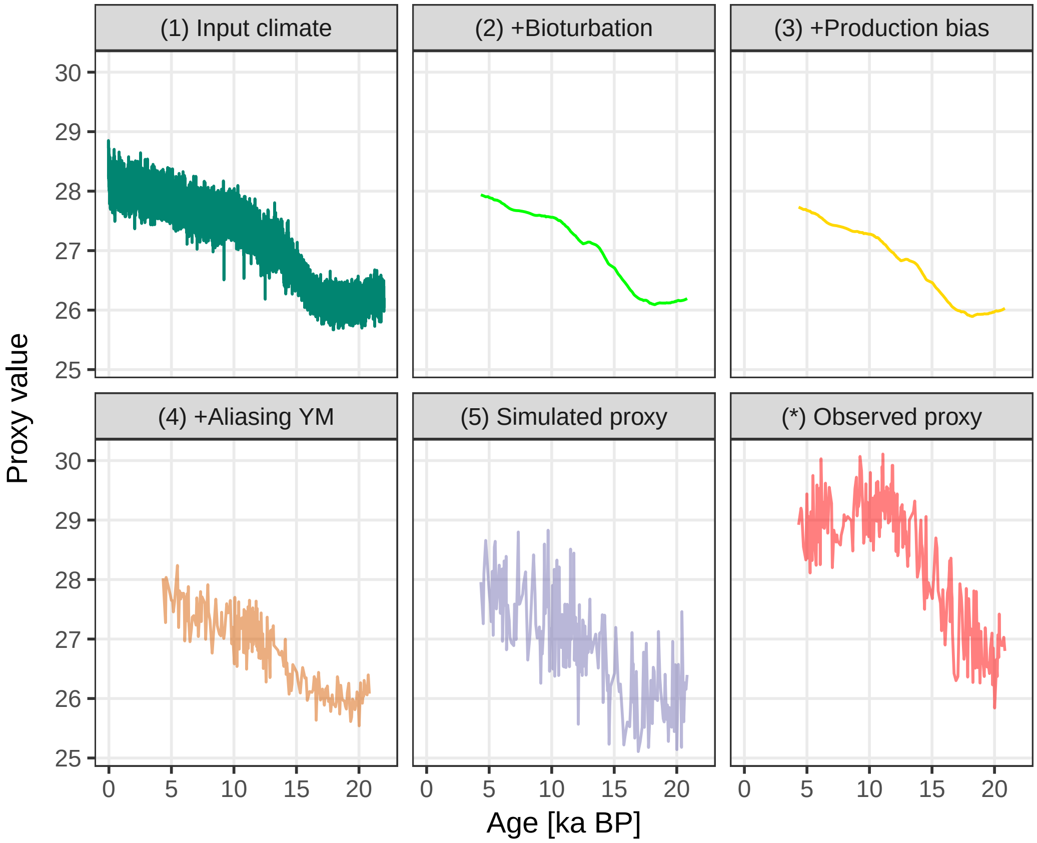
```
# Seasonality of climate
clim.seas.MD97_2141 <- data.frame(
  Month = 1:12,
  Temperature = colMeans(N41.t21k.climate - 273.15)
)

p.t <-  ggplot(clim.seas.MD97_2141, aes(x = Month, y = Temperature)) +
  geom_line(colour = "Black", alpha = 0.8) +
  geom_point(colour = "Black", alpha = 0.8) +
  scale_x_continuous("Month", breaks = seq(2, 12, by = 3),
                     labels = month.abb[seq(2, 12, by = 3)]) +
  scale_y_continuous("Mean temperature [°C]")+
  theme_bw() + labs(title = "(b)") +
  theme(plot.title = element_text(hjust = 0, vjust = 1),
        panel.grid.minor = element_blank())
```

```
# Combine plots
egg::ggarrange(p.ab, p.t)
```



```r
library(sedproxy)

# Reverse matrix so that top row is most recent year,
# also convert from Kelvin to °C
N41.t21k.climate.in <- N41.t21k.climate[nrow(N41.t21k.climate):1, ] - 273.15

# Convert matrix to a ts object and set start to most recent year,
# in this case -39 (1989 in years "before" 1950)
N41.t21k.climate.in <- ts(N41.t21k.climate.in, start = -39)

# Set seed of random number generator so that the results are reproducable.
set.seed(20170824)

# Call the forward model
Mg_Ca.30 <- ClimToProxyClim(
  clim.signal = N41.t21k.climate.in,
  timepoints = N41.proxy$Published.age,
  sed.acc.rate = N41.proxy$Sed.acc.rate.cm.ka,
  smoothed.signal.res = 1,
  proxy.prod.weights = N41.G.ruber.seasonality,
  meas.noise = 0.46,
  n.samples = 30, n.replicates = 1)
```

```r
plot.dat <- Mg_Ca.30$everything

# Rescale timepoints to ka for plotting
plot.dat$timepoints <- plot.dat$timepoints / 1000

# Add observed proxy record
obs.proxy <- data.frame(timepoints = N41.proxy$Published.age / 1000,
                        value = N41.proxy$Published.temperature,
                        stage = "observed.proxy", replicate = 1)
plot.dat <- bind_rows(obs.proxy, plot.dat)

p.MD97_2141 <- PlotPFMs(plot.dat, stage.order = "seq") +
  facet_wrap(~stage, labeller = as_labeller(stage.labels))  +
  scale_x_continuous("Age [ka BP]") + theme(legend.position = "none")
p.MD97_2141
```



```
# Calculate values quoted in the text
med.sed.acc <- median(N41.proxy$Sed.acc.rate.cm.ka)
exp.sd.age <- round(10 / (med.sed.acc /1000), -1)

shfts <- Mg_Ca.30$simulated.proxy$proxy.bt.sb -
  Mg_Ca.30$simulated.proxy$proxy.bt
```

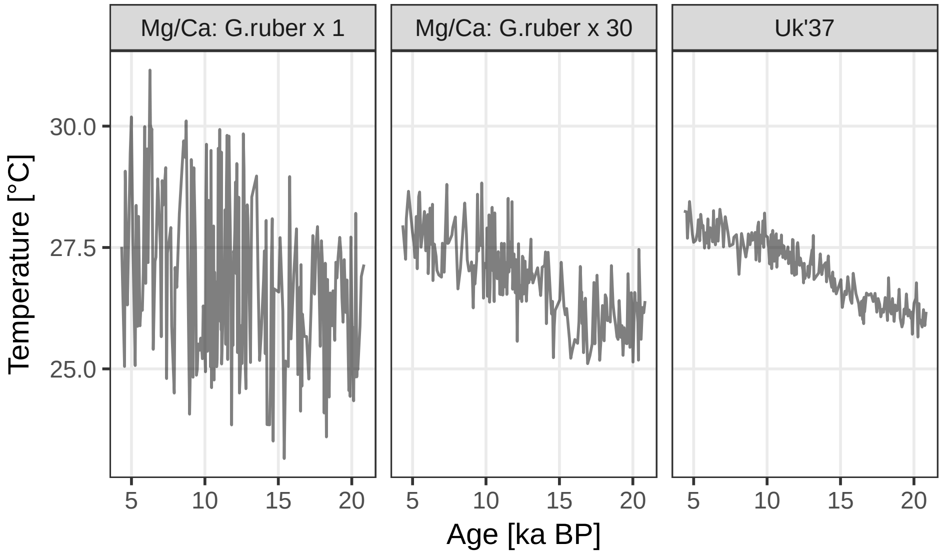
Median sediment accumulation rate = 25.6

SD of ages = 390

Example 2: Influence of the number of foraminifera per sample

```
# Run sedproxy for 1 foram and Uk'37
Mg_Ca.1 <- ClimToProxyClim(
  clim.signal = N41.t21k.climate.in,
  timepoints = N41.proxy$Published.age,
  sed.acc.rate = N41.proxy$Sed.acc.rate.cm.ka,
  proxy.prod.weights = N41.G.ruber.seasonality,
  meas.noise = 0.46, n.samples = 1)

Uk37 <- ClimToProxyClim(
  clim.signal = N41.t21k.climate.in,
  timepoints = N41.proxy$Published.age,
  sed.acc.rate = N41.proxy$Sed.acc.rate.cm.ka,
  meas.noise = 0.25, n.samples = Inf)
```

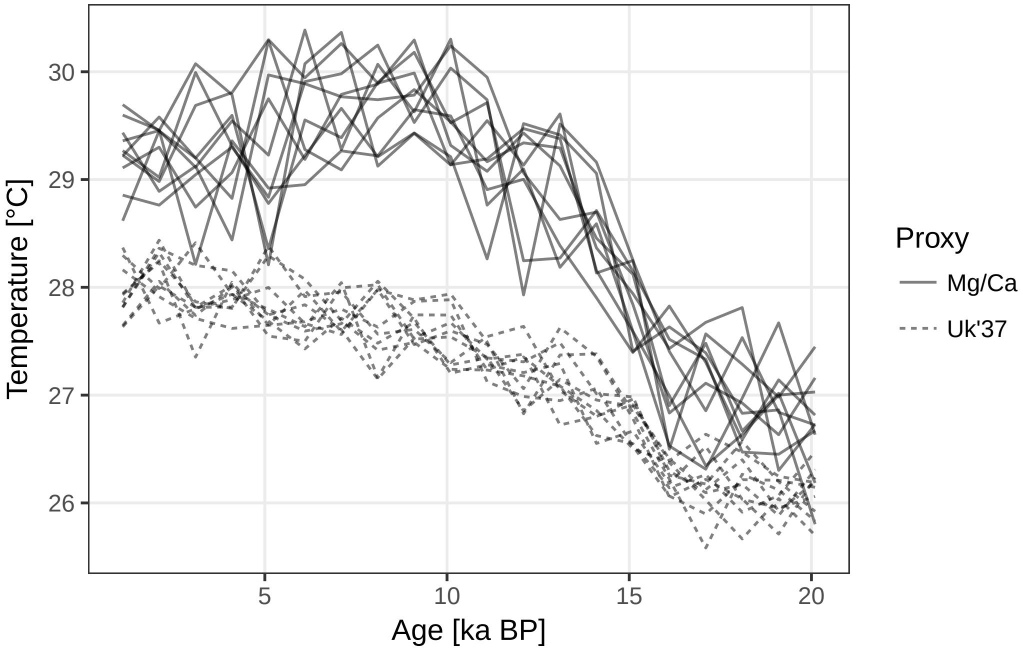
```
all.Pr <- bind_rows(
  "Mg/Ca: G.ruber x 1" = Mg_Ca.1$everything,
  "Mg/Ca: G.ruber x 30" = Mg_Ca.30$everything,
  "Uk'37" = Uk37$everything,  .id = "Type") %>%
  filter(stage == "simulated.proxy") %>%
  mutate(timepoints = timepoints / 1000)


PlotPFMs(all.Pr, alpha.palette = 0.5) +
  facet_wrap( ~ Type) +
  scale_colour_manual(values = "Black") +
  theme(legend.position = "none")  +
  scale_x_continuous("Age [ka BP]") +
  scale_y_continuous("Temperature [°C]")
```



Example 3: Correlation between two proxy types.

```
# 1000 replicates of a hypothetical Uk'37 and Mg/Ca record
Uk37.reps <- ClimToProxyClim(
  clim.signal = N41.t21k.climate.in,
  timepoints = seq(100, 21000, by = 1000),
  sed.acc.rate = 25, proxy.prod.weights = rep(1/12, 12),
  meas.noise = 0.25,
  n.samples = Inf,  n.replicates = 1000)

MgCa.reps <- ClimToProxyClim(
  clim.signal = N41.t21k.climate.in,
  timepoints = seq(100, 21000, by = 1000),
  sed.acc.rate = 25,
  proxy.prod.weights = c(0, 0, 0, 0, 0, 0, 0.2, 0.7, 1, 0.6, 0, 0),
  meas.noise = 0.46,
  n.samples = 30, n.replicates = 1000)

proxies <- bind_rows("Mg/Ca"=MgCa.reps$everything,
                     "Uk'37"=Uk37.reps$everything,
                     .id = "Proxy")

proxies <- filter(proxies, stage == "simulated.proxy")
```

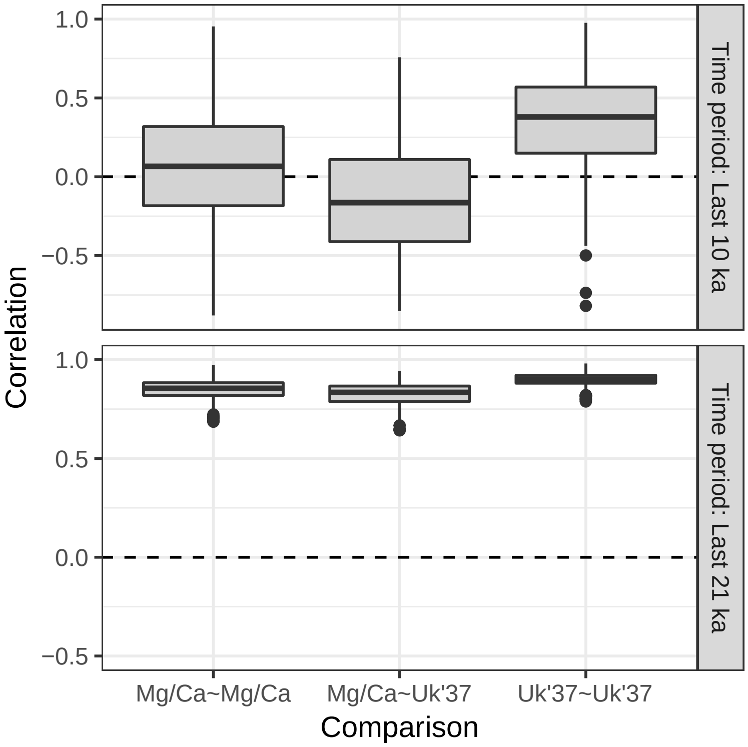
```
# plot the first 10 replicates of each proxy type
proxies.10.reps <- subset(proxies, proxies$replicate < 11)

ggplot(proxies.10.reps, aes(x = timepoints/1000, y = value,
                            linetype = Proxy,
                            group = paste(replicate, Proxy))) +
  geom_line(colour = "Black", alpha = 0.5) +
  scale_x_continuous("Age [ka BP]") +
  scale_y_continuous("Temperature [°C]") +
  theme_bw() +
  theme(panel.grid.minor = element_blank())
```



```
# Calculate correlations between pairs of records
GetCorr <- function(dat){

corr.mg_uk <- dat %>%
  spread(Proxy, value) %>%
  group_by(replicate) %>%
  summarise(Correlation = cor(`Mg/Ca`, `Uk'37`)) %>%
  mutate(Comparison = "Mg/Ca~Uk'37")

rep.1.50 <- dat %>%
  filter(replicate <= (max(replicate)/2))

rep.51.100 <- dat %>%
  filter(replicate > (max(replicate)/2)) %>%
  rename(value.2 = value) %>%
  mutate(replicate = replicate - (max(replicate)/2))
```

```r
corr.same <-
  left_join(rep.1.50, rep.51.100) %>%
  group_by(replicate, Proxy) %>%
  summarise(Correlation = cor(value, value.2)) %>%
  mutate(Comparison = factor(
    ifelse(Proxy == "Uk'37", "Uk'37~Uk'37", "Mg/Ca~Mg/Ca"),
    levels = c("Uk'37~Uk'37", "Mg/Ca~Uk'37", "Mg/Ca~Mg/Ca"),
    ordered = TRUE))

out <- corr.mg_uk %>%
  filter(replicate <= (max(replicate)/2)) %>%
  bind_rows(., corr.same)

return(out)
}

corr.mg_uk.8k <- proxies %>%
  filter(timepoints < 10000) %>%
  GetCorr(.) %>%
  mutate(`Time period` = "Last 10 ka")

corr.mg_uk.21k <- proxies %>%
  GetCorr(.)%>%
  mutate(`Time period` = "Last 21 ka")

corr.mg.uk <- bind_rows(corr.mg_uk.8k, corr.mg_uk.21k)
```

```r
corr.mg.uk %>%
  ggplot(aes(x = Comparison, y = Correlation)) +
  expand_limits(y = c(-0.5, 1))+
  geom_hline(yintercept = 0, linetype = 2) +
  geom_boxplot(fill = "Lightgrey") +
  scale_y_continuous(breaks = seq(-0.5, 1, 0.5)) +
  facet_grid(`Time period`~., scales = "free_y", labeller = "label_both") +
  theme_bw()
```



Example 4: Individual Foraminiferal Analysis

```r
# Parameters for the simulation
duration <- 150000
strt <- 80000
time <- (strt+1):(strt+duration)

var.clim <- 0.15
trans.time <- 133000
trans.speed <- 1000
clim.low <- 1.4
clim.diff <- 1.2
seas.amp <- 0.5
meas.noise <- 0.1
```

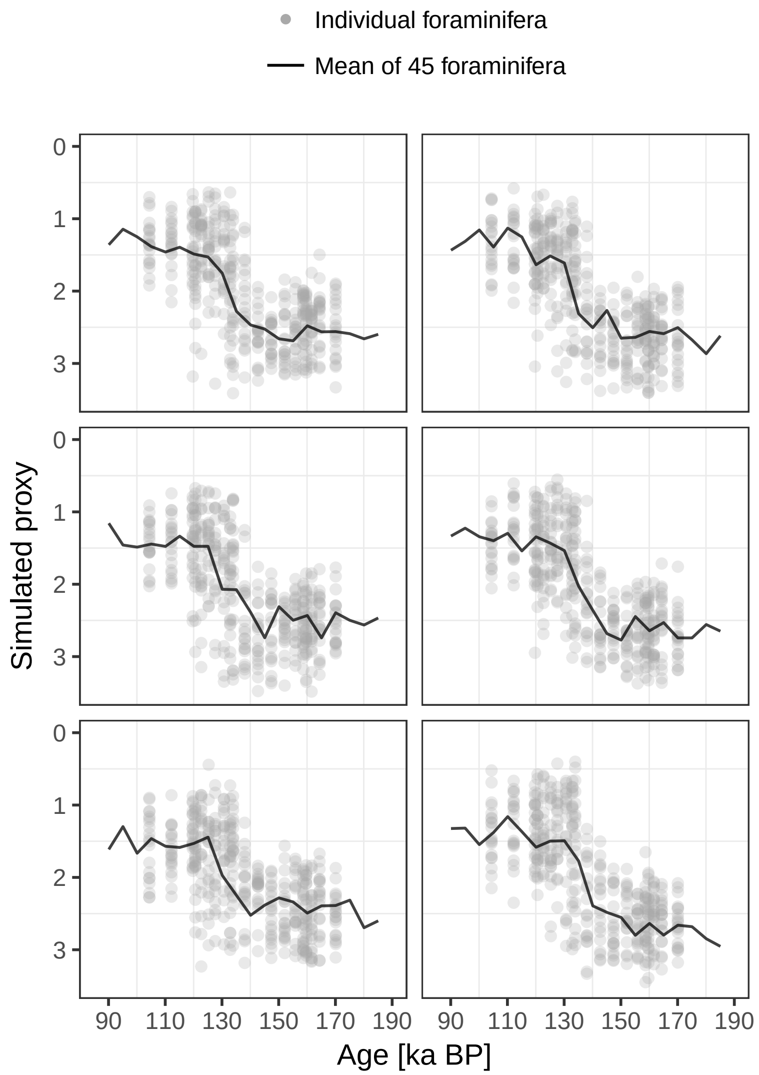
```r
# Estimate mean sediment accumulation rate from
# Table 1 of Scussolini et al. (2013)
sc.dat <- scussolini.tab1 %>%
  filter(Age.ka > 3)

sed.rate <- coef(lm(Depth.cm ~ Age.ka, data = sc.dat))[2]
```

```r
##' Function to simulate a random timeseries with a powerlaw spectrum
##'
##' @title Simulate a random timeseries with a powerlaw spectrum
##' @param beta slope
##' @param N length of timeseries to be generated
##' @return vector containing the timeseries
##' @author Thomas Laepple
##' @export
SimPowerlaw <- function(beta, N)
{
  N2 <- (3^ceiling(log(N, base = 3)))
  df  <- 1 / N2
  f <- seq(from = df, to = 1/2, by = df)
  Filter <- sqrt(1/(f^beta))
  Filter <- c(max(Filter), Filter, rev(Filter))
  x   <- rnorm(N2, 1)
  fx  <- fft(x)
  ffx <- fx * Filter
  result <- Re(fft(ffx, inverse = TRUE))[1:N]
  return(scale(result)[1:N])
}
```

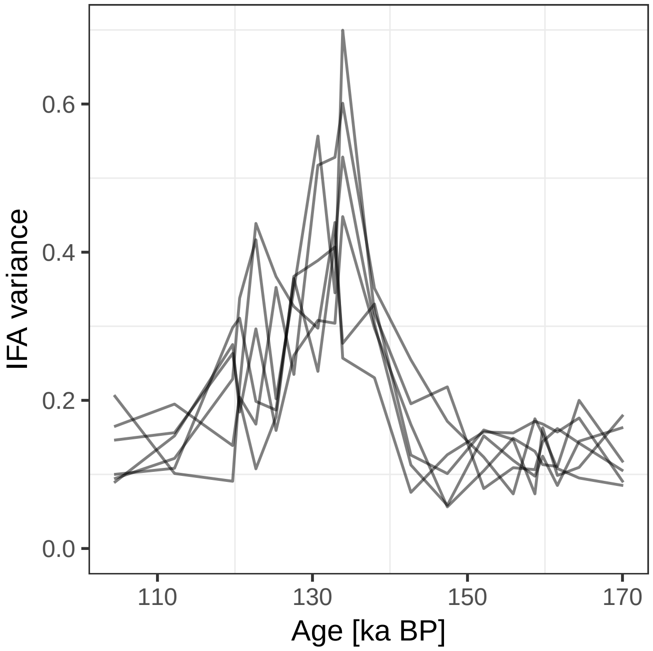
```r
# Run simulation 6 times and return results for plotting
set.seed(20171110)
n.reps <- 6
dat <- replicate(n.reps, {
  clim.noise <- SimPowerlaw(1, duration) * var.clim
  trend <- SSlogis(time, clim.diff, trans.time, trans.speed) + clim.low
  clim.ann <- trend + clim.noise
  clim.seas <- sin(seq(0, 2*pi, length.out = 12)) * seas.amp
  clim <- outer((clim.ann), (clim.seas), "+")
  clim <- ts(clim, start = 1)
# run proxy model for IFA
PFM.IFA <- ClimToProxyClim(clim.signal = clim,
                timepoints = scussolini.tab1$Age.ka[3:22]*1000 - strt,
                meas.noise = meas.noise, n.samples = 1,
                n.replicates = 20,
                sed.acc.rate = sed.rate, bio.depth = 10,
                smoothed.signal.res = 10000)

# run proxy model for bulk samples
PFM.bulk <- ClimToProxyClim(clim.signal = clim,
                timepoints = seq(90, 190*1000, by = 5000),
                meas.noise = meas.noise, n.samples = 45,
                n.replicates = 1, sed.acc.rate = sed.rate, bio.depth = 10,
                smoothed.signal.res = 1000)

plot.dat.bulk <- PFM.bulk$everything %>%
  filter(stage == "simulated.proxy") %>%
  mutate(stage = "Mean of 45 foraminifera")

plot.dat <- PFM.IFA$everything %>%
  filter(stage %in% c("simulated.proxy")) %>%
  mutate(stage = gsub("simulated.proxy", "Individual foraminifera", stage)) %>%
  bind_rows(., plot.dat.bulk)

return(plot.dat)
}, simplify = FALSE)

names(dat) <- paste0("Rep ", 1:n.reps)
```

```
plot.dat <- dplyr::bind_rows(dat, .id = "rep")
```

```
plot.dat %>%
  filter(stage == "Individual foraminifera") %>%
  ggplot(aes(x = (timepoints+strt)/1000, colour = stage)) +
  geom_point(aes(y = value, colour = stage), alpha = 0.25, show.legend = T) +
  # geom_point(data = filter(plot.dat, stage == "45 foraminifera"),
  #            aes(y = value, colour = stage), shape = 1) +
  geom_line(data = filter(plot.dat, stage == "Mean of 45 foraminifera"),
            aes(y = value, colour = stage), alpha = 0.75) +
  scale_x_continuous("Age [ka BP]", breaks = seq(90, 190, by = 20),
                     limits = c(85, 190)) +
  scale_y_continuous("Simulated proxy", trans = "reverse",
                     breaks = seq(0, 3.5, by = 1)) +
  scale_color_manual("", values = c("Mean of 45 foraminifera" = "Black",
                                    "Individual foraminifera"="Darkgrey")) +
  expand_limits(y = c(0, 3.5)) +
  facet_wrap(~rep, ncol = 2) +
  theme_bw() +
  theme(legend.position = "top", legend.direction = "vertical",
        panel.grid.major = element_blank(),
        #panel.grid.minor = element_blank(),
        strip.background = element_blank(),
  strip.text.x = element_blank()) +
  guides(fill = FALSE,
         color = guide_legend(override.aes = list(shape = c(16, NA),
                                                  linetype = c(0, 1),
                                                  alpha = c(1, 1))))
```



```r
# calculate variance between individuals
sd.dat <- plot.dat %>%
  filter(stage == "Individual foraminifera") %>%
  group_by(rep, timepoints, stage) %>%
  summarise(n = n(),
            # include correction for measurement error
            var = var(value) - meas.noise^2)

sd.dat %>%
  ggplot(aes(x = (timepoints+strt)/1000, y = var)) +
  #geom_point(colour = "Black", alpha = 0.5) +
  geom_line(aes(group = rep), colour = "Black", alpha = 0.5) +
  expand_limits(y = c(0, 0.5)) +
  scale_x_continuous("Age [ka BP]", breaks = seq(90, 190, by = 20)) +
  scale_y_continuous("IFA variance") +
  theme_bw() +
  theme(legend.position = "top",
        panel.grid.major = element_blank(),
        #panel.grid.minor = element_blank(),
        strip.background = element_blank(),
      strip.text.x = element_blank())
```


IFA variance
Age [ka BP]