Supplemental Information

Nicolas Gauthier

Last knit on: 28 January, 2019

Contents


```
library(raster)
library(tidyverse)
library(sf)
library(broom)
library(wql)
library(furrr)
```

Climate Analysis: Drought Patterns

First we estimate present and past climate patterns.

Study Area

First we define a boundary box covering much of the western United States, ranging between W124.5 deg and W−107 deg and N31 deg and N37.5 deg, which we'll use to constrain all subsequent climate analyses. This study area is significantly larger than the one we'll define in the next section for the network analysis. This allows us to sample a much wider range of climatic variability, while still remaining within the broader western US climate zone. This ensures that both A) our statistical analyses will be more robust to sampling error and B) the results will be less sensitive to the exact location and dimensions of our study area.

```
bbox_wus <- extent(c(-124.5, -102, 30, 42.1))
states <- maps::map('state', regions = c('arizona', 'new mexico',
                                         'colorado', 'california',
                                         'utah', 'nevada'),
                    fill = TRUE, plot = FALSE) %>%
  st_as_sf
bbox <- c(xmin = -124.5, xmax = -102, ymin = 30, ymax = 42.1) %>%
```

```
  st_bbox(crs = st_crs(4326)) %>%
  st_as_sfc
bbox_swsn <- c(xmin = -113, xmax = -107, ymin = 31, ymax = 37.5)
  st_bbox(crs = st_crs(4326)) %>%
  st_as_sfc
```

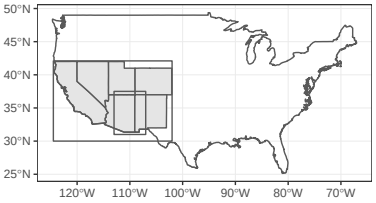


Figure 1: Locations of 2 bounding boxes.

Climate Data

Import high resolution SPEI maps generated from PRISM data. Calculate the average summertime (JJA) SPEI value for each year.

```
# calculate average JJA SPEI
spei_obs <- ((brick('data/spei12_6_PRISM.nc') +
              brick('data/spei12_7_PRISM.nc') +
              brick('data/spei12_8_PRISM.nc')) / 3) %>%
  crop(bbox_wus) %>% # crop to the study area bounding box
  `names<-`(1895:2017) %>% # add year names
  .[[-1]] # SPEI calculated on 12 month lag, so drop 1st year
```

```
## Loading required namespace: ncdf4
```

Import the reconstructed SPEI fields from PHYDA. PHYDA uses a novel off-line data assimilation approach, using simulated SPEI from the CESM LME experiments as physically-consistent model priors and a network of tree rings, ice cores, and corals as assimilated observations. The ensemble Kalman filter uses this information, as well as the spatial covariances from the climate-model prior (e.g. teleconnections), to "spread out" information from the point-based proxies to generate physically optimal extrapolations beyond the proxy locations.

```
spei_recon <- brick('data/da_hydro_JunAug_r.1-2000_d.05-Jan-2018.nc',
                    varname = 'spei_mn') %>%
   .[[1100:1999]] %>% # extract years of interest
   rotate %>% # rotate longitudes to -180 to 180
   crop(bbox_wus, snap = 'near')
```

Let's compare the reconstructions for the summer of 1985.

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

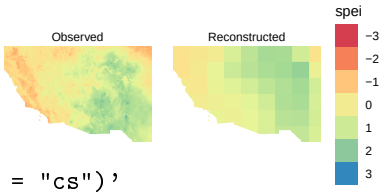


Figure 2: Observed and reconstructed SPEI for summer 1985.

Empirical Orthogonal Functions

We calculate the leading Empirical Orthogonal Functions of a ~100 year series of the summertime average Standardized Precipitation-Evapotranspiration Index. These patterns then undergo a varimax

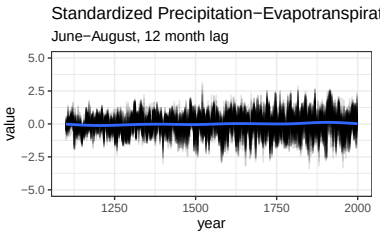


Figure 3: Plotting out the time series of the reconstruction reveals no clear trend in the mean. It does, however, show a pattern of increasing variance over time, which is entirely a function of the changing number of proxies used in the data assimilation approach.

rotation to highlight more physically meaningful spatial patterns. We compare these patterns to those derived from a long term (~2ka) reconstruction based on data assimilation and CESM LME. We confirm that the spatial patterns detected for the past 100 years have been robust over time and explain up to 85% of the variance in a full 1,000 year sequence of reconstructed drought dynamics.

Although the particular choices of variables, resolution, domain size, truncation level, were informed by theory and tested to as to minimize sensitivity to particular choices, different researchers could still generate equally reasonable results given different input parameters.

This is a function to reweight our observations based on the latitude, to acount for the areal distortion of each cell as latitude changes.

```
area_weight <- function(x){
  names_x <- names(x)
  x %>%
    init('y') %>% # get a map of latitudes
    `*`(pi/180) %>% # convert to radians
    cos %>%
    sqrt %>%
    `*`(x) %>%
    `names<-`(names_x)
}
```

This function calculates the effective observations in an autocorrelated time series of rasters.

```
n_effective <- function(x){
  n <- nlayers(x)
  x %>%
    area_weight %>%
    as.data.frame(na.rm = TRUE) %>%
    t %>%
    as_tibble %>%
    gather(cell, value) %>%
    nest(value) %>%
    mutate(rho = map_dbl(data, ~cor(.$value, lag(.$value), use = 'comp')),
           effective_n = n * (1 - rho^2) / (1 + rho^2)) %>% #bretherton et al 1999

    summarise(mean(effective_n)) %>%
    pull
}
```

Principal components analysis of observation and recon data

```
# these are adpated from wql and sinkr
obs_pca <- spei_obs %>%
  area_weight %>%
  as.data.frame(na.rm = TRUE) %>%
  t %>%
  prcomp(scale. = FALSE) # use the covariance matrix

recon_pca <- spei_recon %>%
  area_weight %>%
  as.data.frame(na.rm = TRUE) %>%
  t %>%
  prcomp(scale. = FALSE)
```

Now we calculate the rotated empirical orthogonal functions for both the observed and reconstructed drought maps. For the observations, we see that the leading 5 eofs explain 85% of the variance in the series, so we'll retain those for rotation.

we want to look for separation in the error bars. eofs that are not well separate can be considered effective multiplets, and hsould not be split in truncation.

```
obs_eigs <- obs_pca %>%
  tidy(matrix = 'pcs') %>%
  mutate(eigenvalues = std.dev ^ 2,
         error = sqrt(2 / n_effective(spei_obs)),
         low =  eigenvalues * (1 - error) * 100 / sum(eigenvalues),
         hi = eigenvalues * (1 + error) * 100 / sum(eigenvalues),
         cumvar_line = hi + 0.02 * max(hi))

recon_eigs <- recon_pca %>%
  tidy(matrix = 'pcs') %>%
  mutate(eigenvalues = std.dev ^ 2,
         error = sqrt(2 / n_effective(spei_recon)),
         low =  eigenvalues * (1 - error) * 100 / sum(eigenvalues),
         hi = eigenvalues * (1 + error) * 100 / sum(eigenvalues),
         cumvar_line = hi + 0.02 * max(hi))
```

the overlaps mean these are effective multiplets. the real eof can be some linear combination of these. This won't in practice impact our successive results, as long as we don't truncate within these multplets, but only between them. using the log linear test, keep 7 of recon and 6 for obs Look at the variances for the eofs of each field, to inform truncation. Let's retain the leading four modes from PHYDA

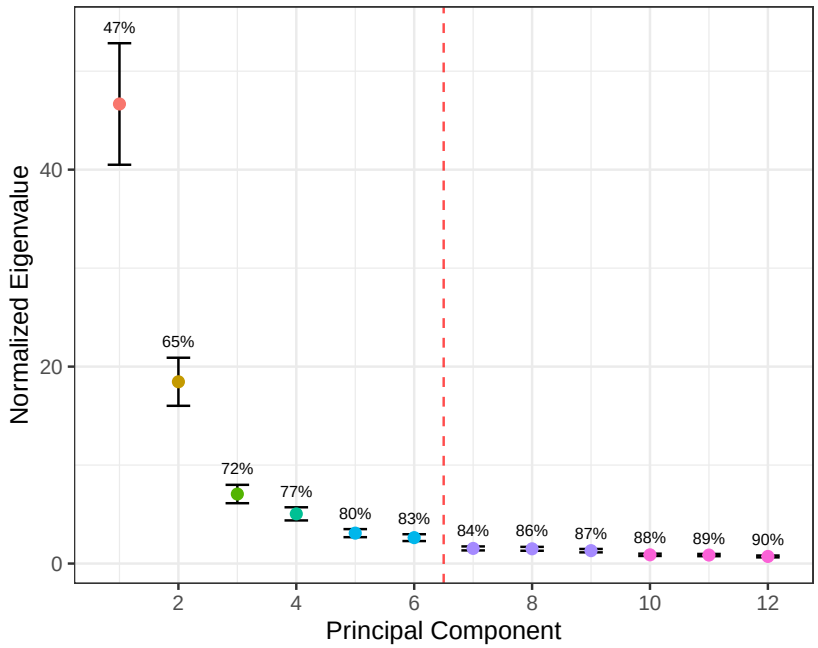


Figure 4: Variance explained

Repeat for the prism observations. Again, we'll retain the leading four modes.

It looks like there is some autocorrelation in phyda but not in the observations, will have to adjust in the future.

Now, calculate the eofs for both fields, retaining the 4 leading components in each case for rotation

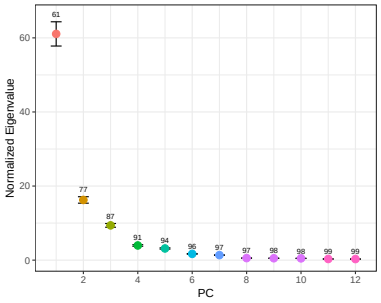


Figure 5: Variance explained

```
# Decide how many modes to retain
n_modes <- 6

obs_reof <- spei_obs %>%
  area_weight %>%
  as.data.frame(na.rm = TRUE) %>%
  t %>% # transpose space and time
  eof(n_modes, scale. = FALSE) # don't rescale (ie we use the covariance matrix, because spei is alread

recon_reof <- spei_recon %>%
  area_weight %>%
  as.data.frame(na.rm = TRUE) %>%
  t %>% # transpose space and time
  eof(n_modes, scale. = FALSE)

plan(multicore)
get_EOFs <- function(pc_object, eigs, rast, n_modes){
```

```
  pc_object %>%
    tidy(matrix = 'variables') %>%
    filter(PC <= n_modes) %>%
    group_by(PC) %>%
    nest %>%
    left_join(eigs[1:2]) %>%
    mutate(data = future_map2(data, std.dev, ~mutate(.x, value = value * .y))) %>%
    unnest %>%
    bind_cols(as.data.frame(rast[[10]], xy = T, na.rm = T)[1:2] %>% slice(rep(1:n(), times = n_modes)))
}

eofs_obs <- get_EOFs(obs_pca, obs_eigs, spei_obs, n_modes)

## Joining, by = "PC"

eofs_recon <- get_EOFs(recon_pca, recon_eigs, spei_recon, n_modes)

## Joining, by = "PC"
```

We see that the leading 3 eigenvectors from the observations and reconstructions are a good match, including their ordering. 4-6 look good too

```
reof_raster <- spei_obs[[1]] %>%
  as.data.frame(xy = TRUE, na.rm = TRUE) %>%
  select(x:y) %>%
  cbind(obs_reof$REOF) %>%
  rasterFromXYZ %>%
  `crs<-`('+proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0') %>%
  `names<-`(paste0('EOF', 1:n_modes)) %>%
  crop(states) %>%
  mask(states)
writeRaster(reof_raster, 'output/reofs.tif', overwrite = TRUE)
```

Let's map out the spatial and temporal patterns in the REOFs. First we'll look at the spatial patterns. It looks like the observed and reconstructed datasets reveal very similar spatial patterns for the 4 leading modes.

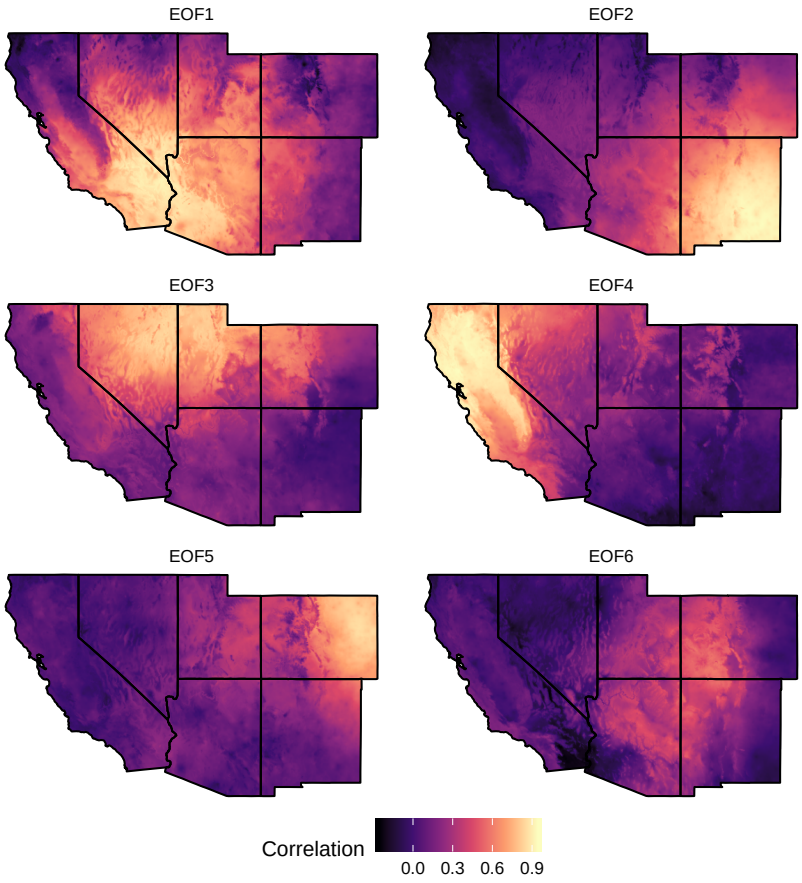


Figure 6: Observed drought REOFs

The colors here are eigenvectors weighted by the sqare root of the associated eigenvalue, so these loadings represent the covariance between each grid cell and each amplitude (principal component). A key assumption here is that the REOFs calculated from both the observations and reconstructions correspond to the same physical phenomena. That way we can just use the observed patterns as the high resolution patterns, and infer their temporal evolution from the reconstructions. This means we don't have to downscale the reconstructed reofs to use them, as otherwise we'd have to deal with issues of spatial represntativeness and non-overlap. Now we can be ensured that these are the same signals.

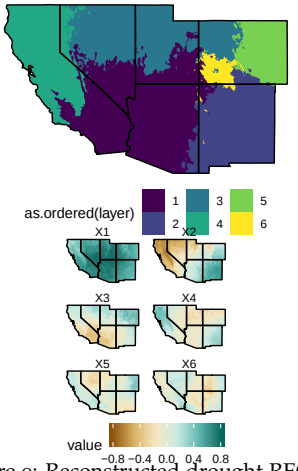
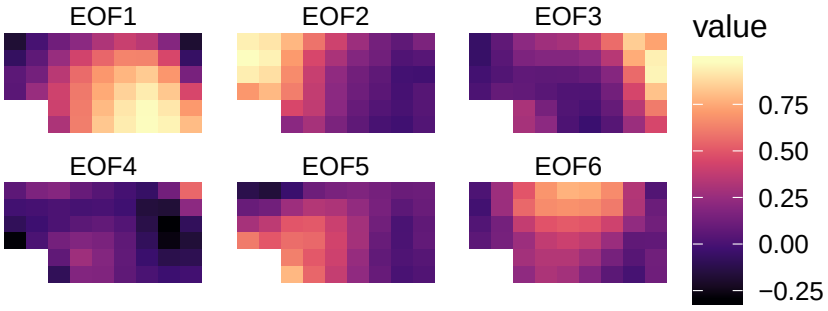


Figure 9: Reconstructed drought REOFs
Figure 7: Observed and reconstructed Empirical Orthogonal Functions before rotation.



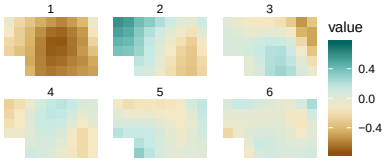


Figure 8: Observed and reconstructed Empirical Orthogonal Functions before rotation.

To confirm this, let's plot the time series (PCs) of each observed and reconstructed REOF against each other, to confirm that they correspond to the same patterns. Look at the correlations between the modes.

Drought Amplitudes

These are the amplitudes of the EOFs, not the REOFs

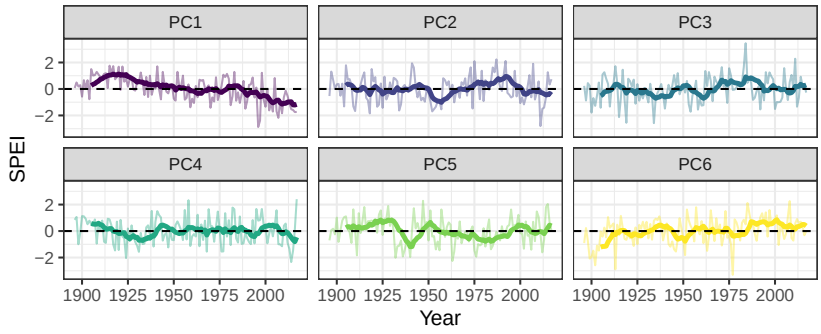
```
#this is the temporary workaround while I still use wql
obs_reof_amp <- obs_reof$amplitude %>%
  as_tibble(rownames = 'year') %>%
  mutate(year = parse_number(year)) %>%
  gather(PC, amplitude, -year) %>%
  mutate(PC = case_when(PC == 'EOF1' ~ 'PC1',
                        PC == 'EOF2' ~ 'PC2',
```

```
                          PC == 'EOF3' ~ 'PC3',
                          PC == 'EOF4' ~ 'PC4',
                          PC == 'EOF5' ~ 'PC5',
                          PC == 'EOF6' ~ 'PC6'))

recon_reof_amp <- recon_reof$amplitude %>%
  as_tibble(rownames = 'year') %>%
  mutate(year = parse_number(year)) %>%
  gather(PC, amplitude, -year) %>%
  mutate(PC = case_when(PC == 'EOF1' ~ 'PC2',
                          PC == 'EOF2' ~ 'PC4',
                           PC == 'EOF3' ~ 'PC5',
                           PC == 'EOF4' ~ 'PC6',
                           PC == 'EOF5' ~ 'PC1',
                           PC == 'EOF6' ~ 'PC3')) %>%
  arrange(PC)

obs_reof_amp %>%
  group_by(PC) %>%
    mutate(amplitude = if_else(PC == 'PC6', amplitude * -1, amplitude)) %>%
  mutate(amp_smooth = zoo::rollmeanr(amplitude, k = 10, fill = NA)) %>%
ggplot(aes(year, amplitude, color = PC)) +
   # geom_vline(xintercept = c(2000), linetype = 2, alpha= .5) +
    geom_line(alpha = .4) +
  geom_line(aes(y = amp_smooth), size = 1.2) +
  facet_wrap(~PC) +
     geom_hline(yintercept = 0, linetype = 2, alpha= 1, color = 'black') +
  scale_color_viridis_d(guide = FALSE) +
  theme_bw() +
  labs(x = 'Year', y = 'SPEI')

## Warning: Removed 54 rows containing missing
## values (geom_path).
```

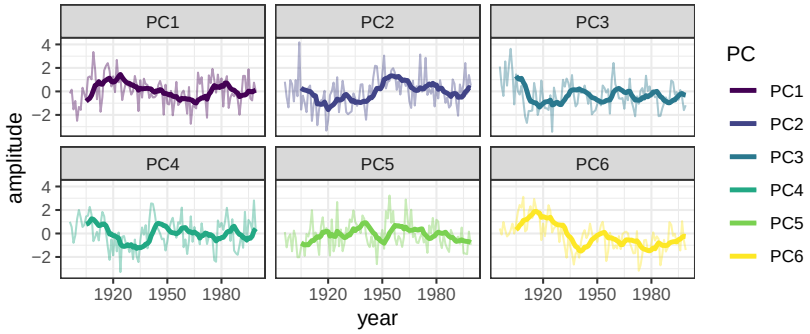



```
ggsave('figures/pc_obs.pdf', width = 6, height = 4)
```

```
## Warning: Removed 54 rows containing missing
## values (geom_path).
```

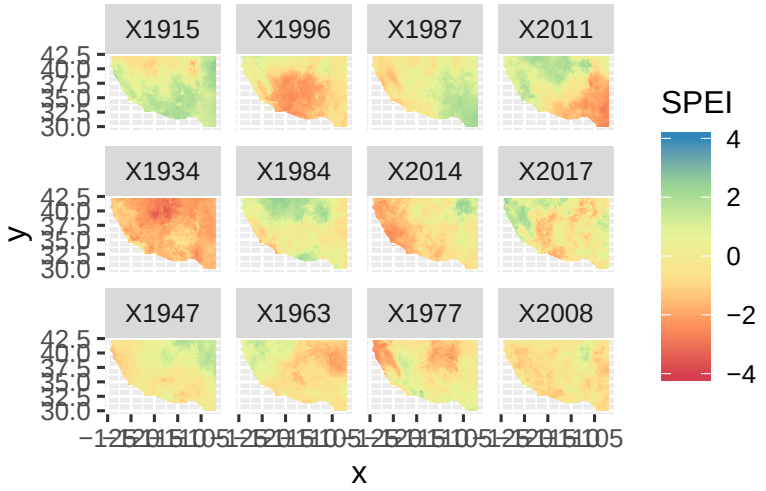
```
recon_reof_amp %>%
  group_by(PC) %>%
  mutate(amplitude = if_else(PC == 'REOF2', amplitude * -1, amplitude)) %>%
  filter(year >= 1896) %>%
  mutate(amp_smooth = zoo::rollmeanr(amplitude, k = 10, fill = NA)) %>%
ggplot(aes(year, amplitude, color = PC)) +
  #  geom_vline(xintercept = c(1977), linetype = 2, alpha= .5) +
  geom_line(alpha = .4) +
  geom_line(aes(y = amp_smooth), size = 1.2) +
  facet_wrap(~PC) +
  scale_color_viridis_d() +
  theme_bw()
```

```
## Warning: Removed 54 rows containing missing
## values (geom_path).
```



So from this plot, we can tell that all but PC6 is of the right sign (high pc amplitude == high positive SPEI)

```
obs_reof_amp %>%
  group_by(PC) %>%
  filter(amplitude == max(amplitude) | amplitude == min(amplitude)) %>%
  pull(year) %>%
  `-`(1895) %>%
  spei_obs[[.]] %>%
  as.data.frame(xy = TRUE, na.rm = TRUE) %>%
  gather(year, SPEI, X1915:X2008) %>%
  mutate(year = factor(year, levels = paste0('X', c(1915, 1996 ,1987, 2011, 1934, 1984, 2014, 2017, 194
  ggplot(aes(x, y, fill = SPEI)) +
  geom_raster() +
  facet_wrap(~year) +
  coord_quickmap() +
  scale_fill_distiller(palette = 'Spectral', limits = c(NA, 4), direction = 1)
```



To cite use of dataset: Boyin Huang, Peter W. Thorne, Viva F. Banzon, Tim Boyer, Gennady Chepurin, Jay H. Lawrimore, Matthew J. Menne, Thomas M. Smith, Russell S. Vose, and Huai-Min Zhang (2017): NOAA Extended Reconstructed Sea Surface Temperature (ERSST), Version 5. [indicate subset used]. NOAA National Centers for Environmental Information. `doi:10.7289/V5T72FNM` [access date]. Please note: If you acquire NOAA_ERSST_V5 data products from PSD, we ask that you acknowledge us in your use of the data. This may be done by including text such as NOAA_ERSST_V5 data provided by the NOAA/OAR/ESRL PSD, Boulder, Colorado, USA, from their Web site at `https://www.esrl.noaa.gov/psd/` in any documents or publications using these data. We would also appreciate receiving a copy of the relevant publications. This will help PSD to justify keeping the NOAA_ERSST_V5 data set freely available online in the future. Thank you!

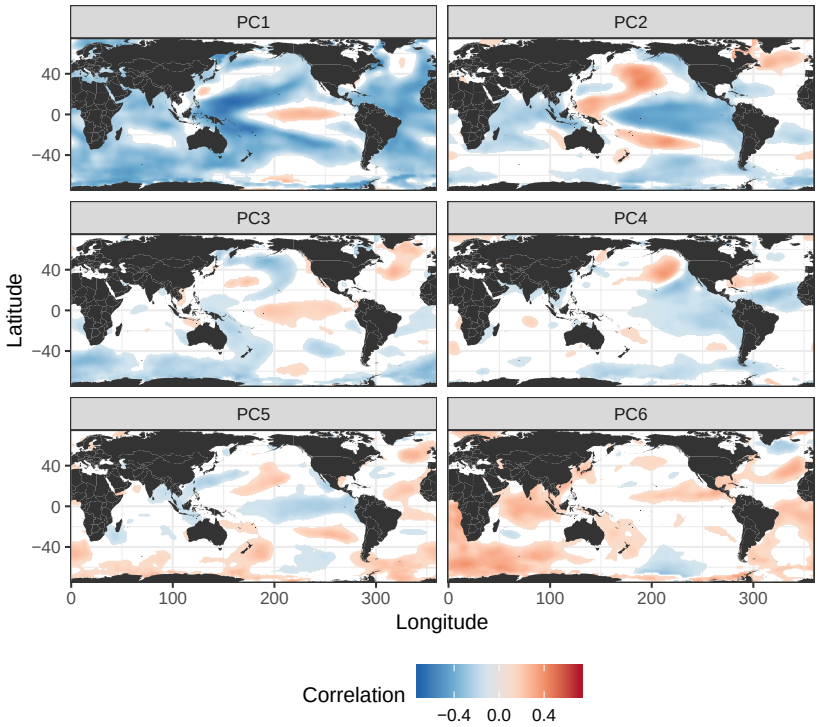
```
sst_mean <- brick('data/sst.mon.ltm.1981-2010.nc')

sst_jan <- brick('data/sst.mnmean.nc') %>%
  .[[505:1966]] %>%
  .[[seq(1, 1462, 12)]] %>%
  `-`(sst_mean[[1]]) %>% crop(extent(c(-1, 359, -75, 75))) %>%
  disaggregate(fact = 2, method = 'bilinear')

reof_cor <- brick(
  calc(sst_jan, fun=function(x) cor(c(x),c(obs_reof$amplitude[,1]))),
  calc(sst_jan, fun=function(x) cor(c(x),c(obs_reof$amplitude[,2]) * -1)),
  calc(sst_jan, fun=function(x) cor(c(x),c(obs_reof$amplitude[,3]) * -1)),
  calc(sst_jan, fun=function(x) cor(c(x),c(obs_reof$amplitude[,4]) * -1)),
  calc(sst_jan, fun=function(x) cor(c(x),c(obs_reof$amplitude[,5]) * -1)),
```

```
  calc(sst_jan, fun=function(x) cor(c(x),c(obs_reof$amplitude[,6]) * -1))) %>%
  `names<-`(paste0('PC', 1:n_modes))

world <- maps::map('world',wrap=c(0,360),fill = TRUE, plot = FALSE)
reof_cor %>%
  as.data.frame(xy = T, na.rm = T) %>%
  gather(eof, value, PC1:PC6) %>%
    filter(value > .1 | value < -.1) %>%
  ggplot(aes(x, y)) +
  geom_raster(aes(fill = value)) +
  facet_wrap(~eof, nrow = 3) +
  scale_fill_distiller(name = 'Correlation', palette = 'RdBu', direction = -1, limits = c(-.7,.7)) +
  coord_quickmap(ylim = c(-75,75), expand = FALSE) +
  geom_polygon(data = world, aes(x = long, y = lat, group = group)) +
  labs(#title = 'Drought teleconnections',
       #subtitle = 'PC correlation to winter sea surface temperatures',
       x = 'Longitude', y = 'Latitude') +
  theme_bw() +
  theme(legend.position = "bottom")
```




```
ggsave('figures/sst_correlation.png', width = 6, height = 5.5)
```

Now let's fit these models in practice. First create a data frame of past amplitudes.

```
amplitudes_modern <- obs_reof_amp %>%
  inner_join(recon_reof_amp, by = c('year', 'PC'),
             suffix = c('_obs', '_recon'))
amplitudes_past <- recon_reof_amp %>%
  rename(amplitude_recon = amplitude) %>%
  group_by(PC) %>%
  nest(.key = 'recons')
```

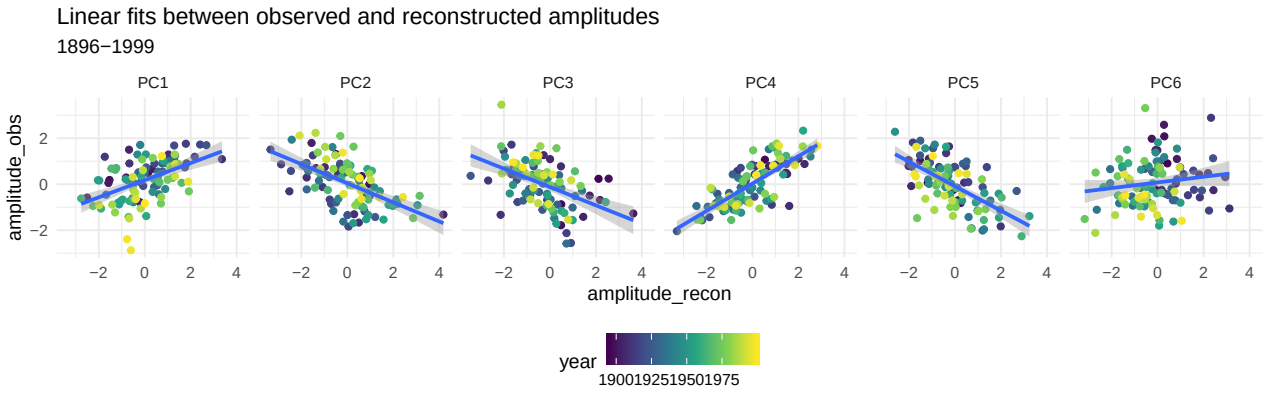


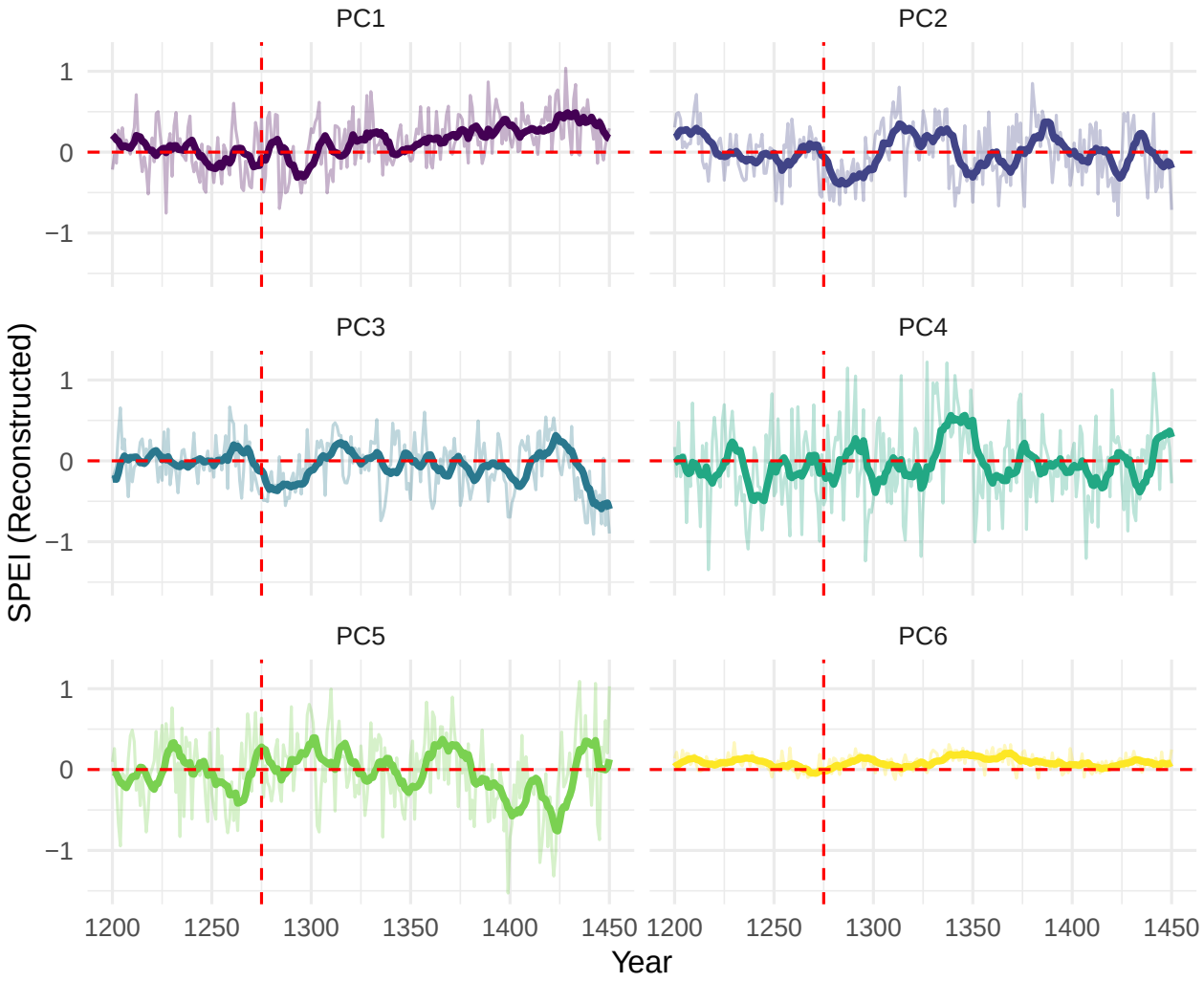
Figure 10: Strong linear correlation between the observed and reconstructed EOFs

We can fit gams between the observed and reconstructed PC time series, and then use these to predict the eof amplitudes back in time. Adding in this regression step essentialily bias-corrects the reconstructed amplitudes.

```
recon_amplitudes <- amplitudes_modern %>%
  group_by(PC) %>%
  nest %>%
  mutate(mod = purrr::map(data, ~lm(amplitude_obs ~ amplitude_recon,
                                    data = .))) %>%
  left_join(amplitudes_past) %>%
  mutate(predictions = purrr::map2(mod, recons,
                                   ~predict(.x, .y, type = 'response')),
         predictions = purrr::map(predictions,
                                  ~tibble(recons = .,
                                          amp_smooth = zoo::rollmeanr(recons, k = 10, fill = NA)))) %>%
  select(PC, recons, predictions) %>%
  unnest %>%
  select(-amplitude_recon) %>%
  mutate(period = floor((year/50)) * 50)
```

```
## Joining, by = "PC"
```

How have the amplitudes of these patterns changed over time? Let's plot the reconstructed time series for each, along with an estimated trend line.



These results point to REOFs 2 and 3 as having major shocks at around 1300 AD.

```
recon_amplitudes %>%
  filter(between(period, 1200, 1400)) %>%
  ggplot(aes(factor(period), abs(recons))) +
  geom_boxplot() +
  geom_hline(yintercept = 0, color = 'red', linetype = 2) +
  scale_color_brewer(palette = 'Spectral') +
```

```
  facet_wrap(~PC) +
  theme_minimal()

recon_amplitudes %>%
  mutate(PC = paste0('reof',parse_number(PC))) %>%
  rename(time = period) %>%
  filter(between(time, 1200, 1400)) %>%
  group_by(PC, time) %>%
  summarise(amplitude = mean(abs(recons))) %>%
  mutate(time = as.factor(time)) %>%
  write_csv('output/amplitudes.csv')

recon_amplitudes %>%
  filter(between(period, 1200, 1400)) %>%
  ggplot(aes((recons), as.factor(period),fill = ..x.., height = ..density..)) +
    ggridges::geom_density_ridges_gradient(stat = 'density', scale = 1.5)+
  facet_wrap(~PC) +
  scale_fill_viridis_c(guide = F) +
  geom_vline(xintercept= 0, linetype = 2) +
  ggridges::theme_ridges(grid = FALSE, center_axis_labels = TRUE)

knitr::knit_exit()
```