Principal Component Analysis

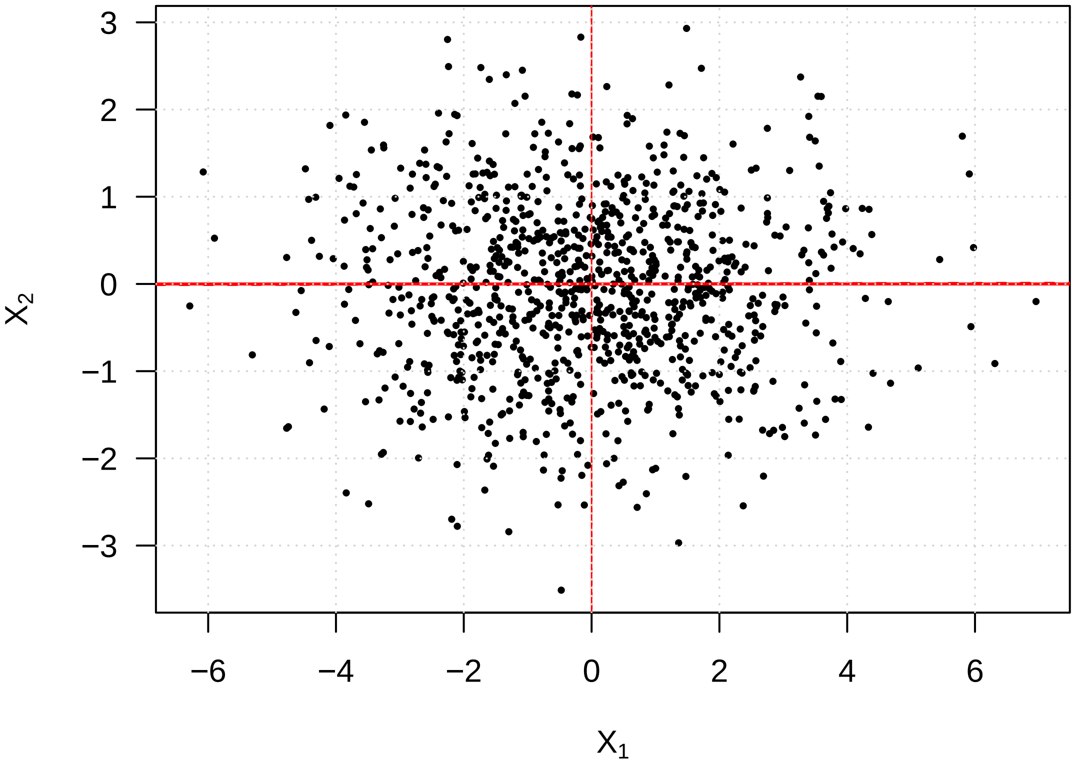
Blake Pappas

December 17, 2023

Toy Examples

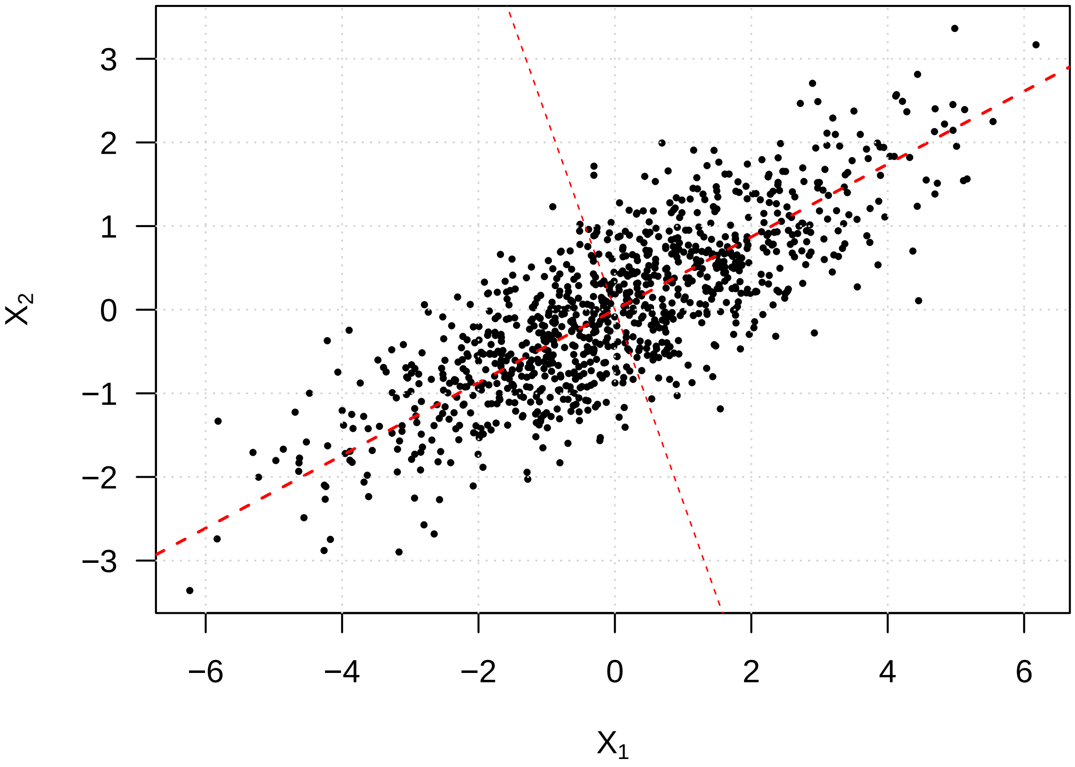
Example 1

```r
library(MASS)
sim1 <- mvrnorm(n = 1000, mu = c(0, 0), Sigma = matrix(c(4, 0, 0, 1), 2, 2))
plot(sim1, pch = 16, cex = 0.5, las = 1,
xlab = expression(X[1]),
ylab = expression(X[2]))

# Population PCs
abline(h = 0, col = "red", lwd = 1.5)
abline(v = 0, col = "red", lwd = 0.75)

# Sample PCs
pca.sim1 <- prcomp(sim1)
abline(0, pca.sim1$rotation[2, 1] / pca.sim1$rotation[1, 1],
       col = "red", lwd = 1.5, lty = 2)
abline(0, pca.sim1$rotation[2, 2] / pca.sim1$rotation[1, 2],
       col = "red", lwd = 0.75, lty = 2)
grid()
```



Example 2

```
sim2 <- mvrnorm(n = 1000, mu = c(0, 0), Sigma = matrix(c(4, 1.6, 1.6, 1), 2, 2))
plot(sim2, pch = 16, cex = 0.5, las = 1,
xlab = expression(X[1]),
ylab = expression(X[2]))

# Sample PCs
pca.sim2 <- prcomp(sim2)
abline(0, pca.sim2$rotation[2, 1] / pca.sim2$rotation[1, 1],
       col = "red", lwd = 1.5, lty = 2)
abline(0, pca.sim2$rotation[2, 2] / pca.sim2$rotation[1, 2],
       col = "red", lwd = 0.75, lty = 2)
grid()
```



Men’s 100k Road Race Example

Read the Data

```
URL <- "http://homepage.divms.uiowa.edu/~dzimmer/applied-multivariate/race100k.dt"
race <- read.table(URL)
head(race)
```

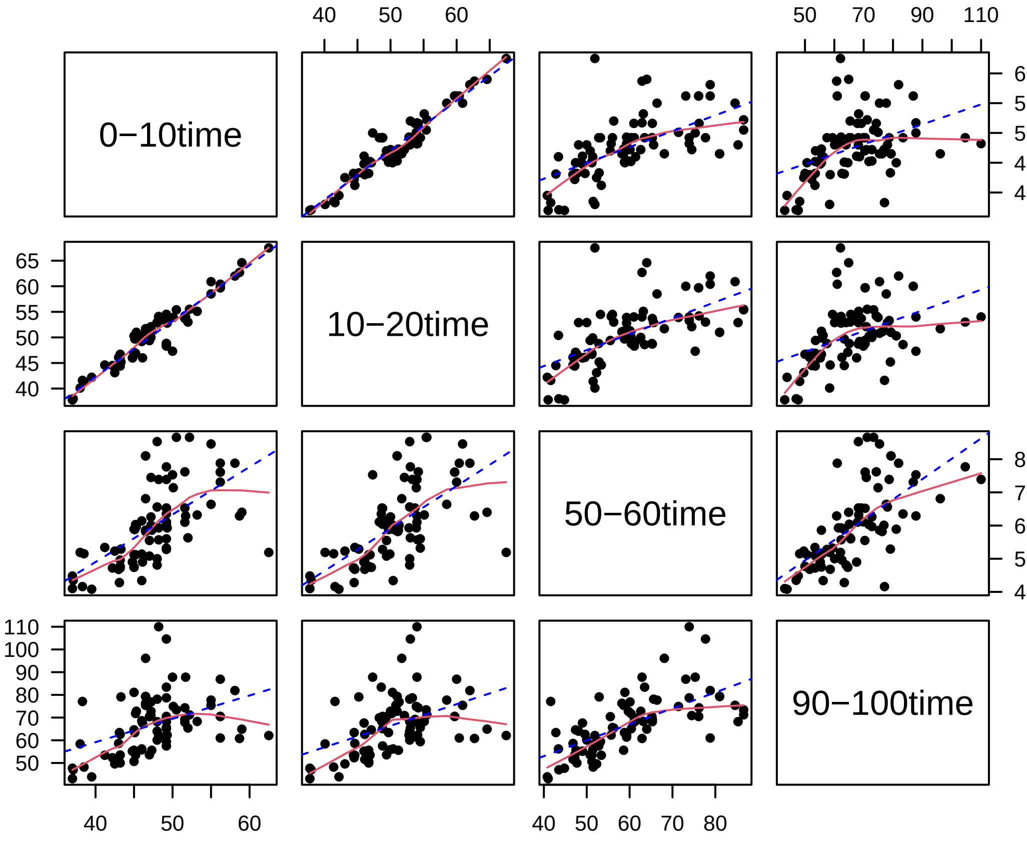
```
##   V1   V2   V3   V4   V5   V6   V7   V8   V9  V10  V11 V12
## 1  1 37.0 37.8 36.6 39.6 41.0 41.0 41.3 45.7 45.1 43.1  39
## 2  2 39.5 42.2 40.0 42.3 40.6 40.8 42.0 43.7 41.0 43.9  39
## 3  4 37.1 38.0 37.7 42.4 41.6 43.5 48.7 49.7 44.8 47.0  -1
## 4  3 37.0 37.8 36.6 39.6 41.0 44.8 44.5 49.4 44.6 47.7  36
## 5  5 42.2 44.5 41.9 43.4 43.0 47.2 49.1 49.9 46.8 52.3  34
## 6  6 43.0 44.6 41.2 42.1 42.5 46.8 47.5 55.8 56.6 58.6  46
```

```
# Remove the Subject ID
race <- race[, -1]
names(race)[1:10] <- paste(seq(0, 90, by = 10), "-", seq(10, 100, by = 10), "time", sep = "")
names(race)[11] <- "Age"
str(race)
```

```
## ’data.frame’:    80 obs. of  11 variables:
##  $ 0-10time   : num  37 39.5 37.1 37 42.2 43 43.2 43.2 38.5 42.5 ...
##  $ 10-20time  : num  37.8 42.2 38 37.8 44.5 44.6 44.4 46.7 41.4 43.1 ...
##  $ 20-30time  : num  36.6 40 37.7 36.6 41.9 41.2 41 44.8 40.1 40.6 ...
##  $ 30-40time  : num  39.6 42.3 42.4 39.6 43.4 42.1 43.4 47.5 43.2 44.5 ...
##  $ 40-50time  : num  41 40.6 41.6 41 43 42.5 43 47.4 43.2 45.4 ...
##  $ 50-60time  : num  41 40.8 43.5 44.8 47.2 46.8 47.2 47.7 51.5 52.3 ...
##  $ 60-70time  : num  41.3 42 48.7 44.5 49.1 47.5 52.4 49.9 56.7 59.7 ...
##  $ 70-80time  : num  45.7 43.7 49.7 49.4 49.9 55.8 57.3 52.1 71.5 59.3 ...
##  $ 80-90time  : num  45.1 41 44.8 44.6 46.8 56.6 54.4 50.7 56.2 55 ...
##  $ 90-100time: num  43.1 43.9 47 47.7 52.3 58.6 53.5 50 48.2 49.6 ...
##  $ Age        : int  39 39 -1 36 34 46 35 47 30 -1 ...
```

Pair Plots

```
par(pch = 16, las = 1, mgp = c(2, 1, 0), mar = c(3, 3, 1, 0.6))
choose <- c(1, 2, 6, 10)
pairs(race[, choose], panel = function(x, y){panel.smooth(x, y)
  abline(lsfit(x, y), lty = 2, col = "blue")})
```



Covariance PCA

```
# Use prcomp
race.pc <- prcomp(race[, -11]) # Conducts a PCA excluding the last variable in race
race.pc$sdev # Standard deviation
```

```
##  [1] 27.124989  9.923061  7.298289  6.107511  5.102671  4.153828  2.834363
##  [8]  2.061318  1.548295  1.136055
```

```
race.pc$rotation # Rotation
```

```
##                   PC1          PC2        PC3          PC4          PC5
## 0-10time   0.1287968 -0.210414172  0.3616819 -0.033912697  0.146863043
## 10-20time  0.1519804 -0.248882958  0.4168698 -0.071459244  0.223499877
## 20-30time  0.1991457 -0.314222348  0.3411230 -0.055307785  0.247236551
## 30-40time  0.2395504 -0.330105834  0.2023603 -0.008025876  0.004951718
## 40-50time  0.3144130 -0.302199147 -0.1348327  0.110995564 -0.355216915
## 50-60time  0.4223103 -0.214642986 -0.2221994 -0.085385615 -0.372760666
## 60-70time  0.3358313  0.049592968 -0.1940861 -0.600075660 -0.192885326
## 70-80time  0.4065846  0.008219549 -0.5382054  0.127699488  0.719979635
## 80-90time  0.3992873  0.267171103  0.1503976  0.718454162 -0.208792667
## 90-100time 0.3854145  0.689038639  0.3472533 -0.280062639  0.055038598
##                    PC6         PC7         PC8          PC9         PC10
## 0-10time   -0.20662226  0.43142669 -0.28050959  0.040956253  0.690231730
## 10-20time  -0.13208196  0.32548306 -0.22949390  0.045957055 -0.712710219
## 20-30time   0.05239710 -0.34262499  0.45728241 -0.587229171  0.083503905
## 30-40time   0.14550042 -0.44807746  0.10556173  0.744773474  0.069516361
## 40-50time   0.28606304 -0.24465020 -0.64643943 -0.305851747 -0.005119664
## 50-60time   0.29103710  0.54002862  0.44912216  0.037473464 -0.022805790
## 60-70time  -0.64366176 -0.18515038 -0.02190932 -0.019251548 -0.019004215
## 70-80time   0.03296114  0.02932515 -0.08159154  0.036786064  0.017934724
## 80-90time  -0.41250620 -0.04787933  0.11299688 -0.002275953 -0.039645274
## 90-100time  0.40479975 -0.02913417 -0.09432018 -0.002518047  0.031906092
```

```
Sigma <- var(race.pc$x)
Eigenvalue <- diag(Sigma)
(Proportion <- round(Eigenvalue / sum(Eigenvalue), 3)) # Calculates the Proportion of the variation exp
```

```
##   PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10
## 0.748 0.100 0.054 0.038 0.026 0.018 0.008 0.004 0.002 0.001
```

```
(Cumulative <- round(cumsum(Eigenvalue) / sum(Eigenvalue), 3)) # Calculates the cumulative proportion o
```

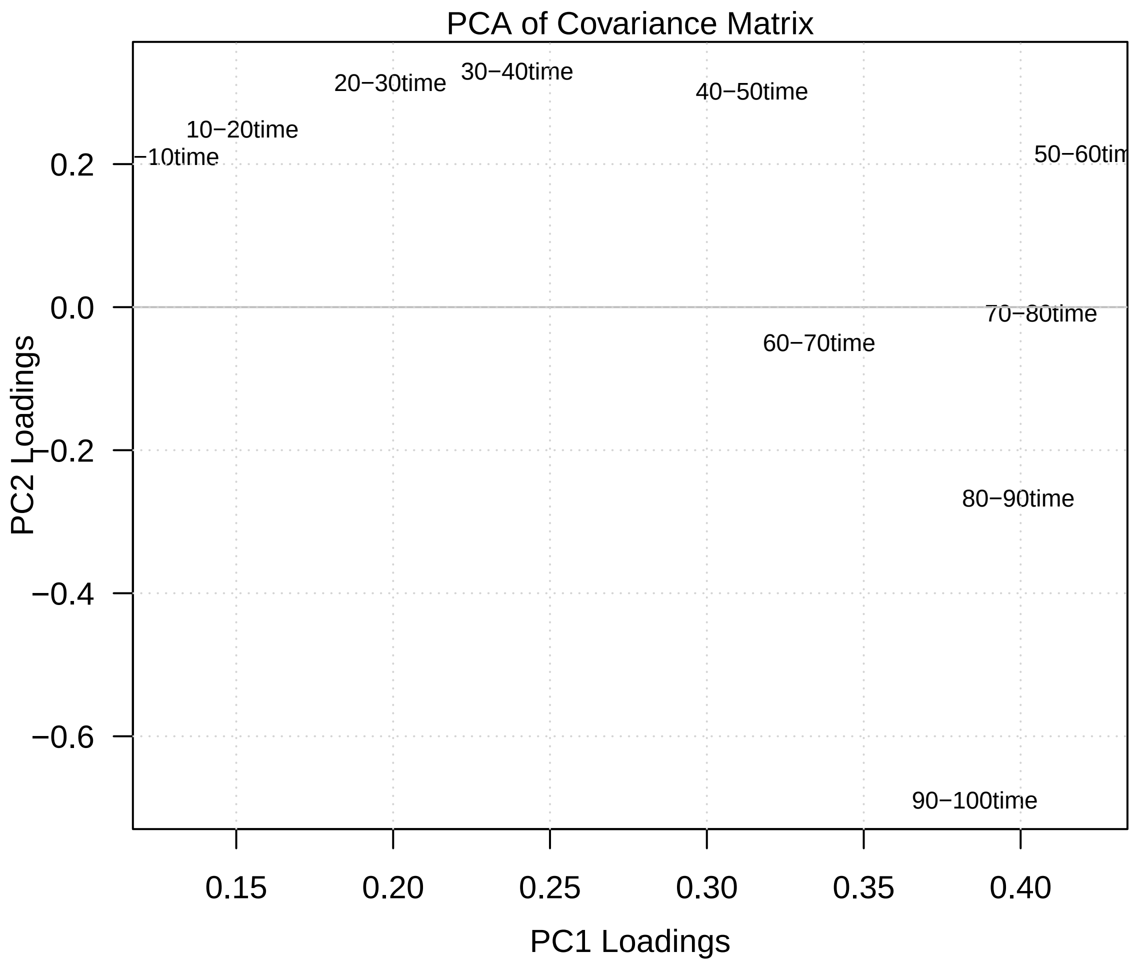
```
##   PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10
## 0.748 0.848 0.902 0.940 0.966 0.984 0.992 0.996 0.999 1.000
```

```
# Use princomp
pcaCOV <- princomp(race[, -11])
str(pcaCOV)
```

```
## List of 7
##  $ sdev    : Named num [1:10] 26.95 9.86 7.25 6.07 5.07 ...
##   ..- attr(*, "names")= chr [1:10] "Comp.1" "Comp.2" "Comp.3" "Comp.4" ...
##  $ loadings: 'loadings' num [1:10, 1:10] 0.129 0.152 0.199 0.24 0.314 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:10] "0-10time" "10-20time" "20-30time" "30-40time" ...
##   .. ..$ : chr [1:10] "Comp.1" "Comp.2" "Comp.3" "Comp.4" ...
##  $ center  : Named num [1:10] 47.6 50.6 49.4 53 54.5 ...
##   ..- attr(*, "names")= chr [1:10] "0-10time" "10-20time" "20-30time" "30-40time" ...
##  $ scale   : Named num [1:10] 1 1 1 1 1 1 1 1 1 1
##   ..- attr(*, "names")= chr [1:10] "0-10time" "10-20time" "20-30time" "30-40time" ...
##  $ n.obs   : int 80
##  $ scores  : num [1:80, 1:10] -56.4 -56.2 -48.7 -50.7 -41 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:10] "Comp.1" "Comp.2" "Comp.3" "Comp.4" ...
##  $ call    : language princomp(x = race[, -11])
##  - attr(*, "class")= chr "princomp"
```

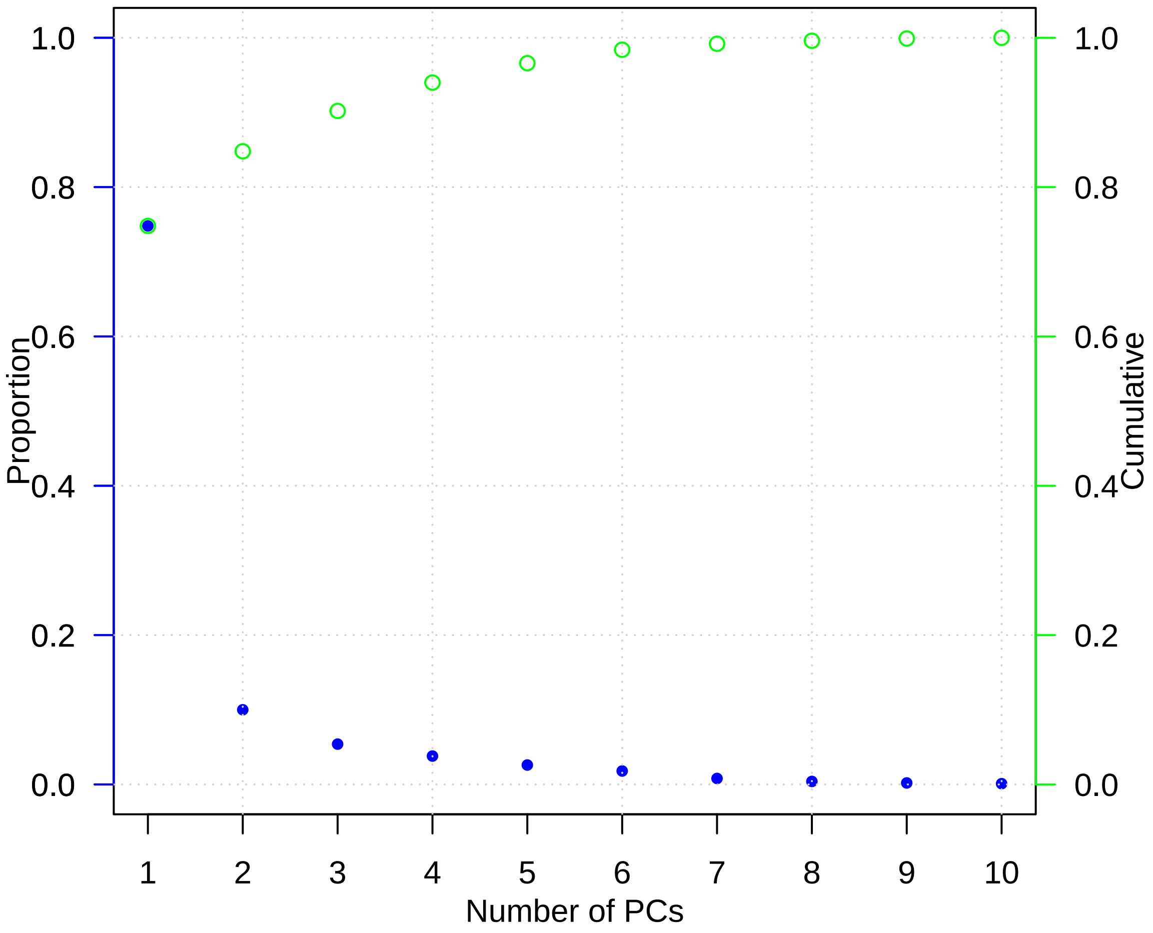
Plot the First 2 PCs and the Loadings

```
par(las = 1, mgp = c(2.4, 1, 0), mar = c(3.5, 3.5, 1, 0.6))
plot(pcaCOV$loadings[, 1:2], xlab = "PC1 Loadings",
     ylab = "PC2 Loadings", type = "n", main = "")
mtext("PCA of Covariance Matrix")
text(pcaCOV$loadings[, 1:2], labels = colnames(race[, -11]),
     cex = 0.75)
abline(h = 0, col = "gray")
grid()
```



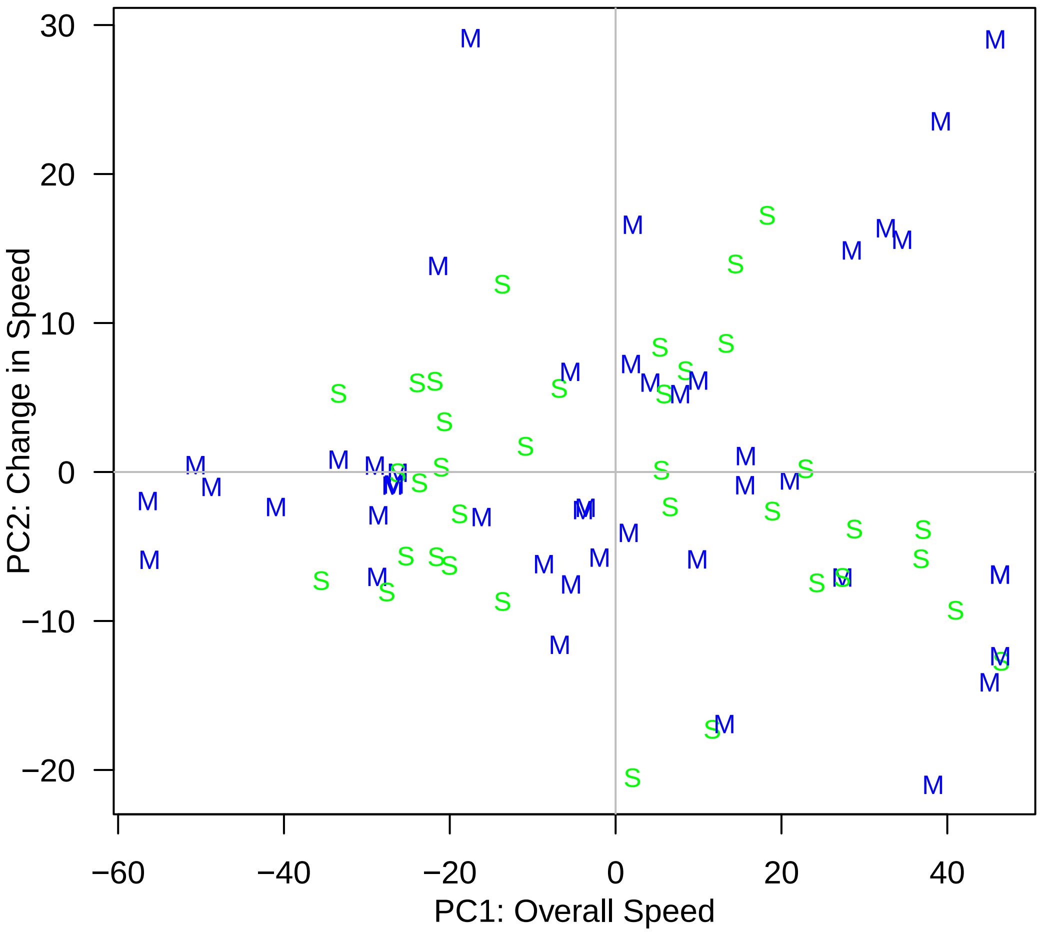
Screen Plot

```
p <- 10
par(las = 1, mgp = c(2, 1, 0), mar = c(3, 3, 1, 3))
plot(1:p, Proportion, xlab = "Number of PCs", ylim = c(0, 1),
ylab = "Proportion", pch = 16, cex = 0.8, xaxt = "n", col = "blue")
axis(1, at = 1:p)
mtext("Cumulative", 4, las = 0, line = 2)
axis(4, col = "green"); axis(2, col = "blue")
grid()
points(1:p, Cumulative, cex = 1, col = "green")
```



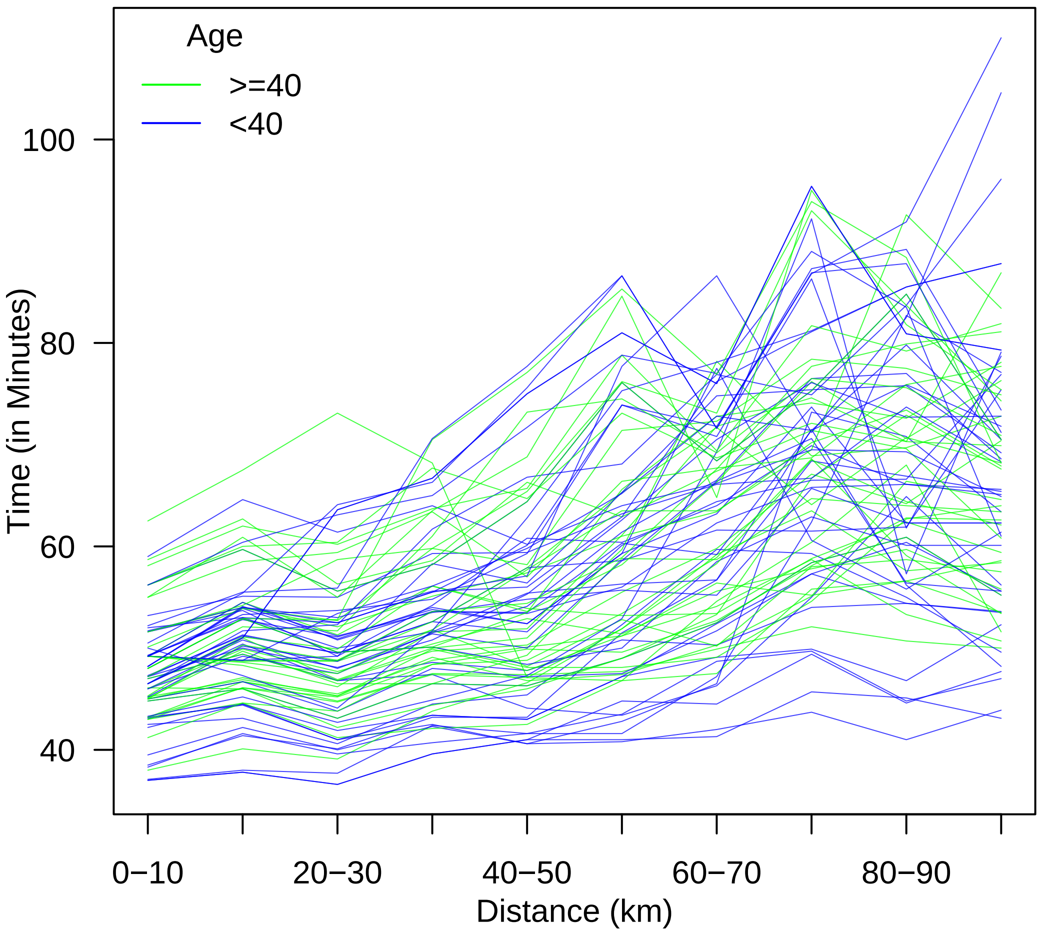
Component Scores by Type

```
par(las = 1, mgp = c(2, 1, 0), mar = c(3, 3, 1, 3))
race.type <- ifelse(race[, 11] >= 40, "S", "M")
col <- ifelse(race.type == "M", "blue", "green")
plot(race.pc$x[, 1], race.pc$x[, 2], type = "n",
xlab = "PC1: Overall Speed", ylab = "PC2: Change in Speed")
text(race.pc$x[, 1], race.pc$x[, 2], labels = race.type, cex = 0.8,
     col = col)
abline(h = 0, col = "gray"); abline(v = 0, col = "gray")
```



Profile Plot

```
library(scales)
par(las = 1, mgp = c(2, 1, 0), mar = c(3, 3, 1, 3))
plot(1:10, race[1, 1:10], type = "l", col = alpha(col[1], 0.75), lwd = 0.5,
     ylim = range(race[, 1:10]), xaxt = "n",
     xlab = "Distance (km)", ylab = "Time (in Minutes)")
for (i in 2:80) lines(1:10, race[i, 1:10], col = alpha(col[i], 0.75),
                  lwd = 0.5)
axis(1, 1:10, paste(seq(0, 90, by = 10), "-", seq(10, 100, by = 10), sep = ""))
legend("topleft", legend = c(">=40", "<40"),
       title = "Age", lty = 1, col = c("green", "blue"),
       bty = "n")
```



Correlation PCA

```
race.std <- scale(race, center = T, scale = T)
races.pc <- prcomp(race.std[, -11])
Sigma.std <- var(races.pc$x)
Eigenvalue.std <- diag(Sigma.std)
(Proportion.std <- round(Eigenvalue.std / sum(Eigenvalue.std), 3))
```

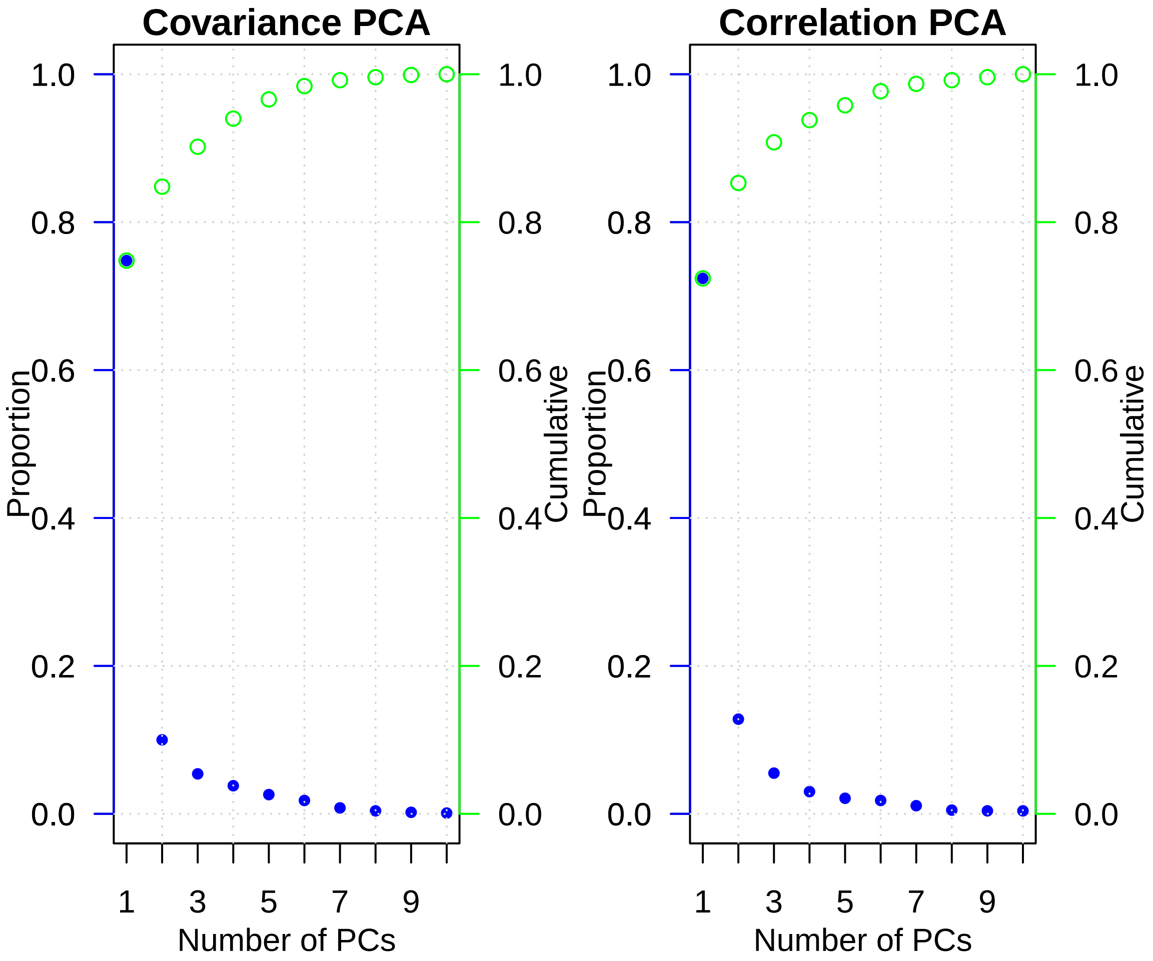
```
##   PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10
## 0.724 0.128 0.055 0.030 0.021 0.018 0.011 0.005 0.004 0.004
```

```
(Cumulative.std <- round(cumsum(Eigenvalue.std) / sum(Eigenvalue.std), 3))
```

```
##   PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10
## 0.724 0.853 0.908 0.938 0.958 0.977 0.987 0.992 0.996 1.000
```

```
par(las = 1, mgp = c(2, 1, 0), mar = c(3, 3, 1.2, 3), mfrow = c(1, 2))
plot(1:p, Proportion, xlab = "Number of PCs", ylim = c(0, 1),
ylab = "Proportion", pch = 16, cex = 0.8, xaxt = "n", col = "blue", main = "Covariance PCA")
```

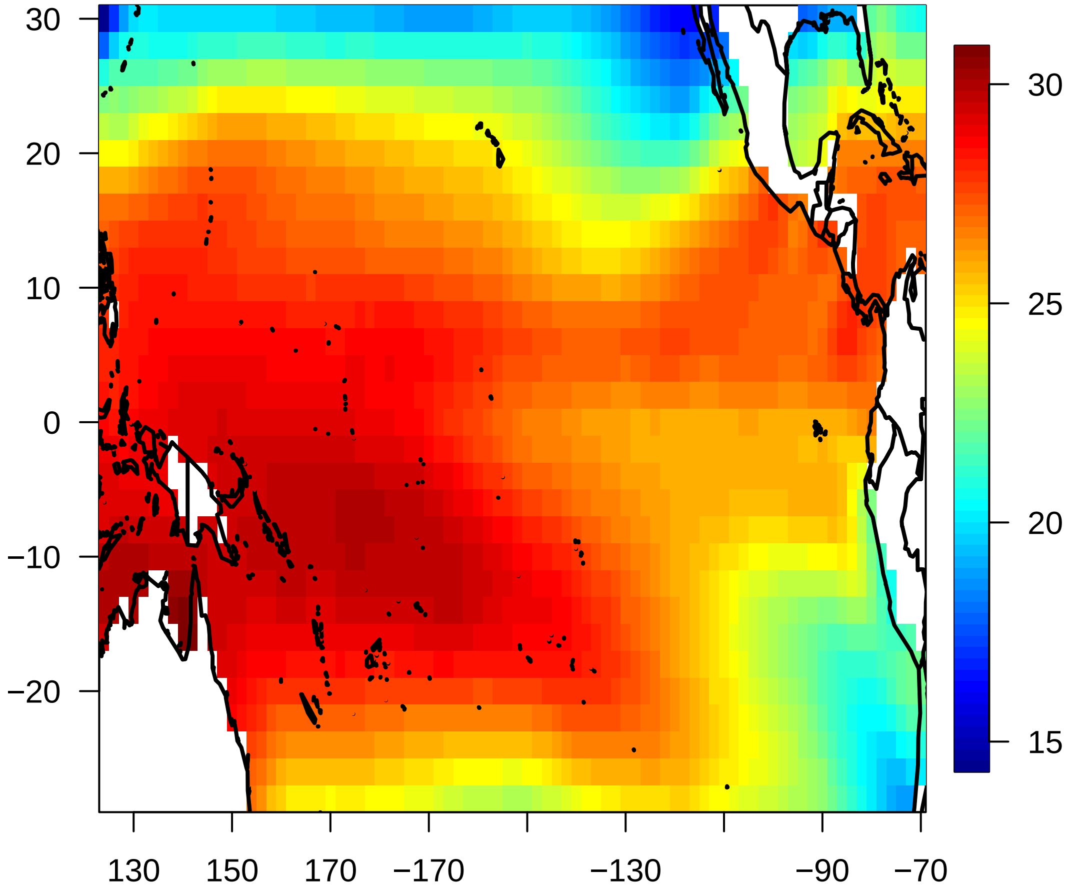
```r
axis(1, at = 1:p)
mtext("Cumulative", 4, las = 0, line = 2)
axis(4, col = "green"); axis(2, col = "blue")
grid()
points(1:p, Cumulative, cex = 1, col = "green")

plot(1:p, Proportion.std, xlab = "Number of PCs", ylim = c(0, 1),
ylab = "Proportion", pch = 16, cex = 0.8, xaxt = "n", col = "blue", main = "Correlation PCA")
axis(1, at = 1:p)
mtext("Cumulative", 4, las = 0, line = 2)
axis(4, col = "green"); axis(2, col = "blue")
grid()
points(1:p, Cumulative.std, cex = 1, col = "green")
```



PCA: SST Example

Load and Visualize the Data

```r
load("SST1.rda")
library(fields)
library(maps)

par(las = 1, mar = c(3, 3, 1, 1))
image.plot(lon1, lat1, SST1[, , 1], xaxt = "n", xlab = "", ylab = "")
lon <- ifelse(lon1 <= 180, lon1, lon1 - 360)
axis(1, at = lon1[seq(4, 84, 10)], lon[seq(4, 84, 10)])
map("world2", add = TRUE, lwd = 2)
```



Compute the SST Anomalies by Subtracting Means

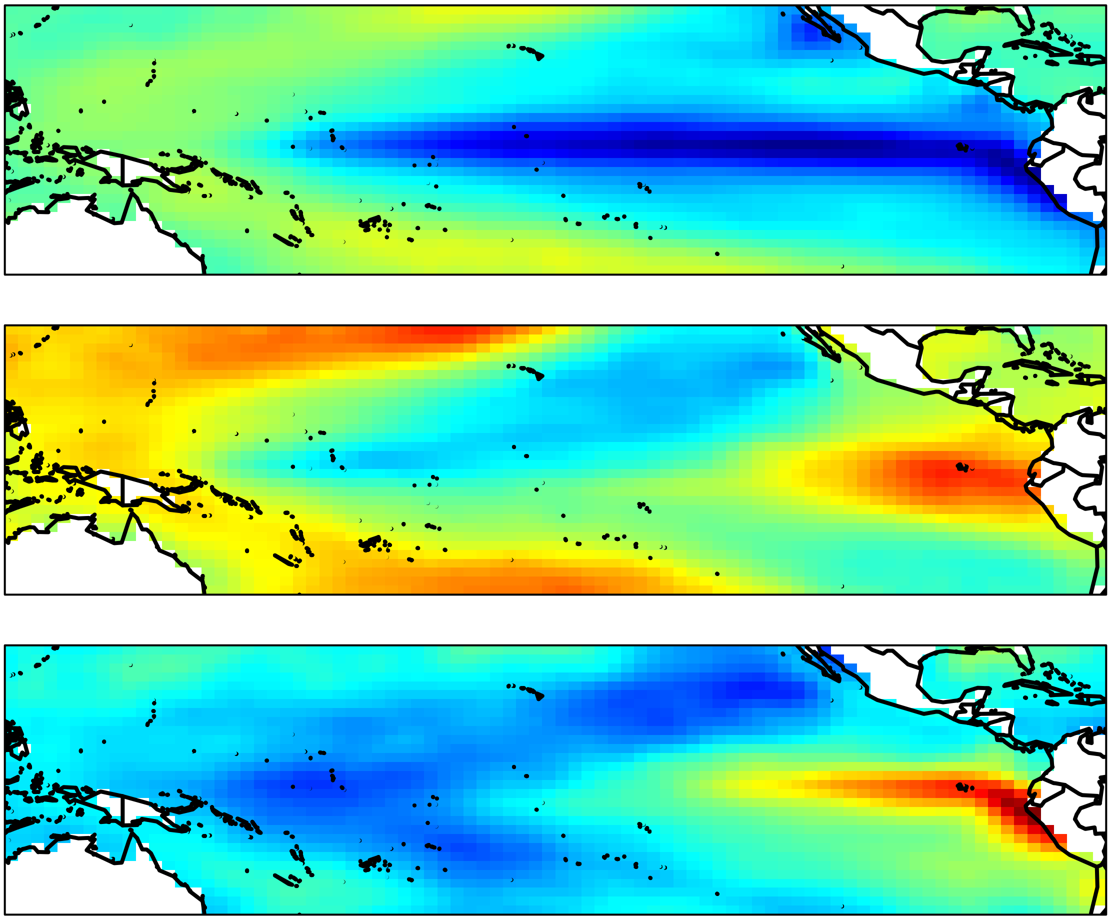
```
t <- array(SST1, dim = c(84, 30, 12, 46))
SST_temp <- apply(t, 1:3, function(x) x - mean(x, na.rm = T))

# Change the Data Into Longitude-Latitude-Month Format
SST_anomalies <- array(dim = c(84, 30, 552))
for (i in 1:84) {
  for (j in 1:30) {
    SST_anomalies[i, j, ] <- c(t(SST_temp[, i, j, ]))
  }
}
```

Empirical Orthogonal Functions (EOFs)

```
# Extracting First Three EOFs Via Singular Value Decomposition
temp <- array(SST_anomalies, c(84 * 30, 552))
ind <- is.na(temp[, 1])
temp <- temp[!ind, ]
temp2 <- svd(temp)
U1 <- matrix(NA, 84 * 30)
U1[!ind] <- temp2$u[, 1]; U1 <- matrix(U1, 84, 30)
U2 <- matrix(NA, 84 * 30)
U2[!ind] <- temp2$u[, 2]; U2 <- matrix(U2, 84, 30)
U3 <- matrix(NA, 84 * 30)
U3[!ind] <- temp2$u[, 3]; U3 <- matrix(U3, 84, 30)
zr <- range(c(U1, U2, U3), na.rm = TRUE)

set.panel(3, 1)
```

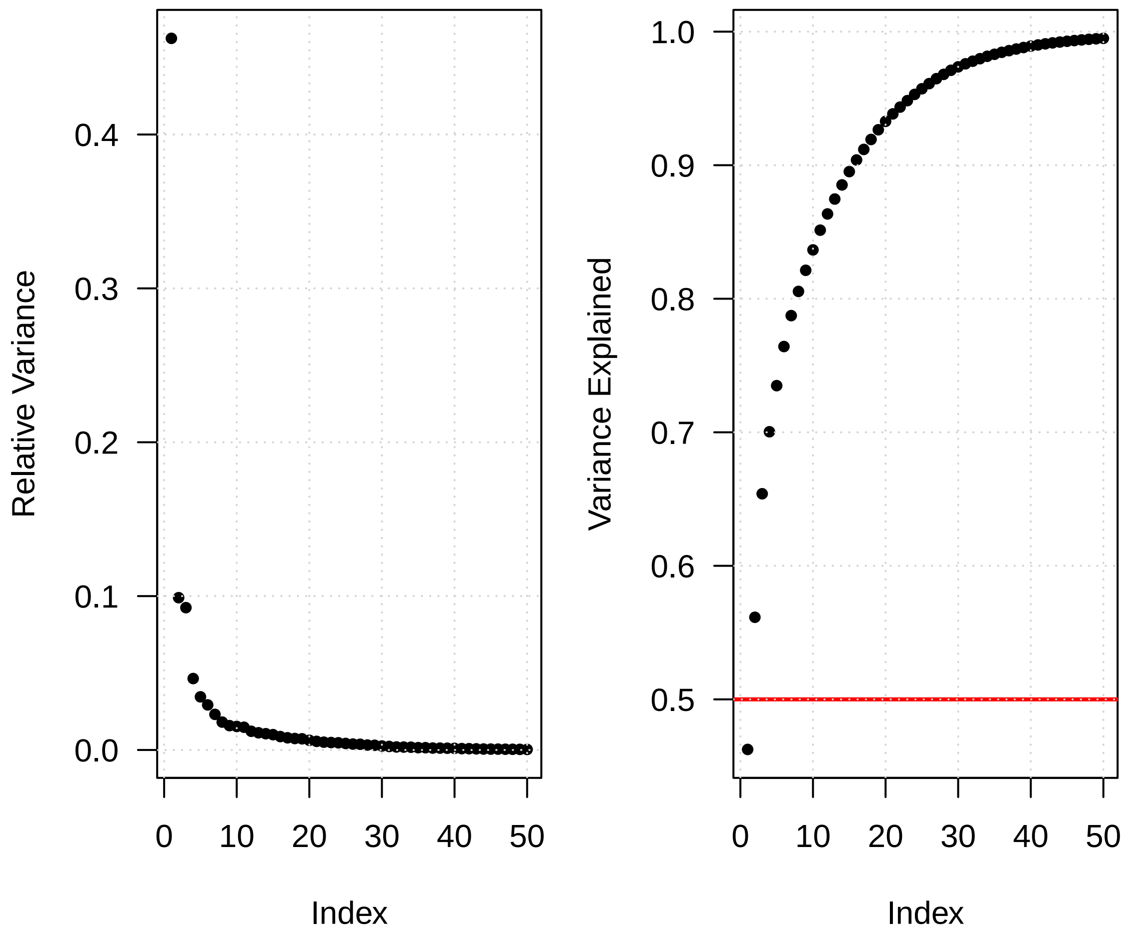
```
## plot window will lay out plots in a 3 by 1 matrix
```

```
par(oma = c(0, 0, 0, 0))
ct <- tim.colors(256)
par(mar = c(1, 1, 1, 1))
image(lon1, lat1, U1, axes = FALSE, xlab = "", ylab = "", zlim = zr, col = ct)
map("world2", add = TRUE, lwd = 2)
box()
image(lon1, lat1, U2, axes = FALSE, xlab = "", ylab = "", zlim = zr, col = ct)
map("world2", add = TRUE, lwd = 2)
box()
image(lon1, lat1, U3, axes = FALSE, xlab = "", ylab = "", zlim = zr, col = ct)
map("world2", add = TRUE, lwd = 2)
box()
```



Screen plot

```
par(mar = c(4, 4, 1, 1), mfrow = c(1, 2), las = 1)
dt <- ((temp2$d^2) / sum(temp2$d^2))
plot(1:50, dt[1:50], xlab = "Index", ylab = "Relative Variance",
     pch = 16, cex = 0.8)
grid()
dt <- (cumsum(temp2$d^2) / sum(temp2$d^2))
plot(1:50, dt[1:50], xlab = "Index", ylab = "Variance Explained", pch = 16, cex = 0.8)
yline(0.5, col = "red", lwd = 2)
grid()
```



1998 Jan El Ni~no Event

```
V <- temp2$v %*% diag(temp2$d)
J <- 337 # The index for January 1998
zr <- range(SST_anomalies, na.rm = TRUE)
set.panel(2, 2)
```

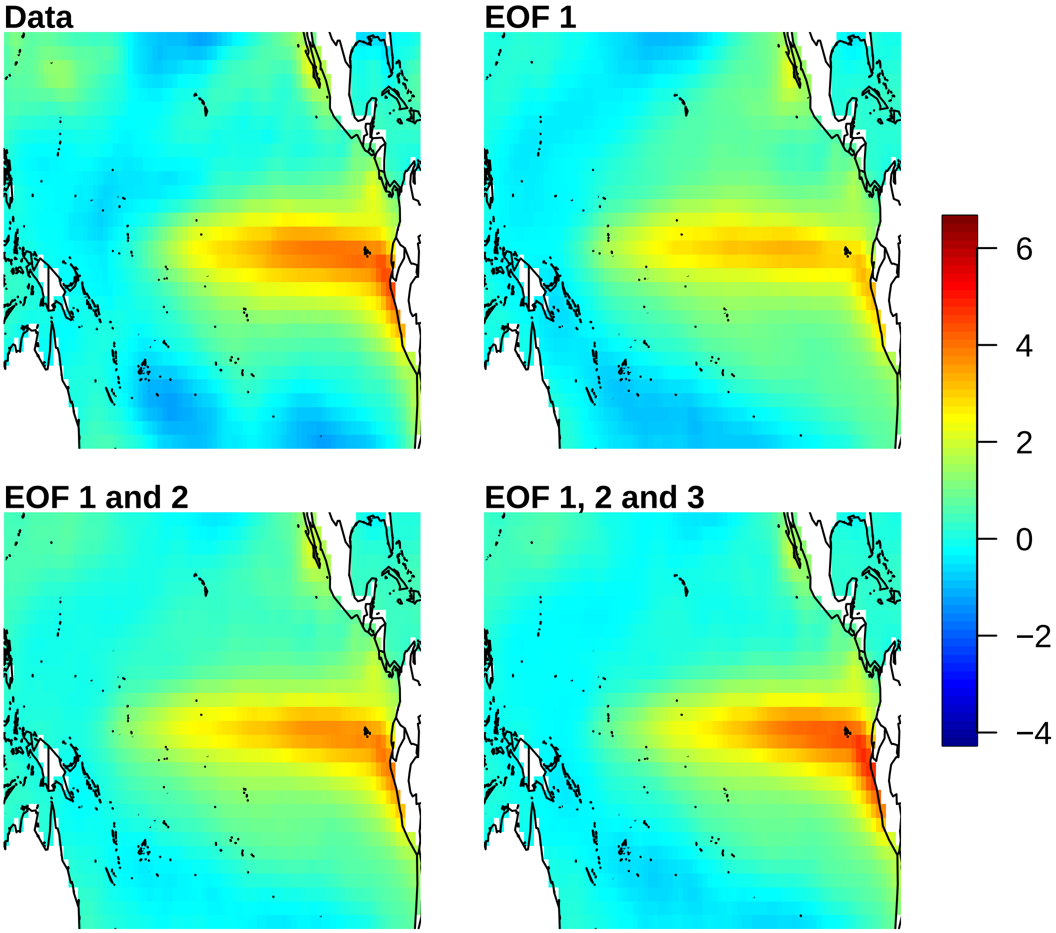
```
## plot window will lay out plots in a 2 by 2 matrix
```

```
par(mar = c(1, 1, 1, 1), oma = c(0, 0, 0, 6))
image(lon1, lat1, SST_anomalies[, , J], axes = FALSE, xlab = "", ylab = "",
      col = tim.colors(256), zlim = zr)
map("world2", add = TRUE)
title("Data", adj = 0)
image(lon1, lat1, V[J, 1] * U1, axes = FALSE, xlab = "", ylab = "",
      col = tim.colors(256), zlim = zr)
map("world2", add = TRUE)
title("EOF 1", adj = 0)
image(lon1, lat1, V[J, 1] * U1 + V[J, 2] * U2, axes = FALSE,
      xlab = "", ylab = "", col = tim.colors(256), zlim = zr)
```

```
map("world2", add = TRUE)
title("EOF 1 and 2", adj = 0)
image(lon1, lat1, V[J, 1] * U1 + V[J, 2] * U2 + V[J, 3] * U3,
      axes = FALSE, xlab = "", ylab = "", col = tim.colors(256),
      zlim = zr)
map("world2", add = TRUE)
title("EOF 1, 2 and 3", adj = 0)
set.panel()
```

```
## plot window will lay out plots in a 1 by 1 matrix
```

```
par(oma = c(0, 0, 0, 0))
image.plot(legend.only = TRUE, zlim = zr, horizontal = FALSE, legend.shrink = 0.6)
```


Data
EOF 1
EOF 1 and 2
EOF 1, 2 and 3
6
4
2
0
−2
−4