

Data Exploration

Nathan Shepherd

2022-04-04

```
library(readr)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##   filter, lag

## The following objects are masked from 'package:base':
##
##   intersect, setdiff, setequal, union

library(ggplot2)
library(FactoMineR) # PCA

rand_dat <- read_csv("../utils/rand_state_acts.csv", show_col_types = FALSE)
names(rand_dat)

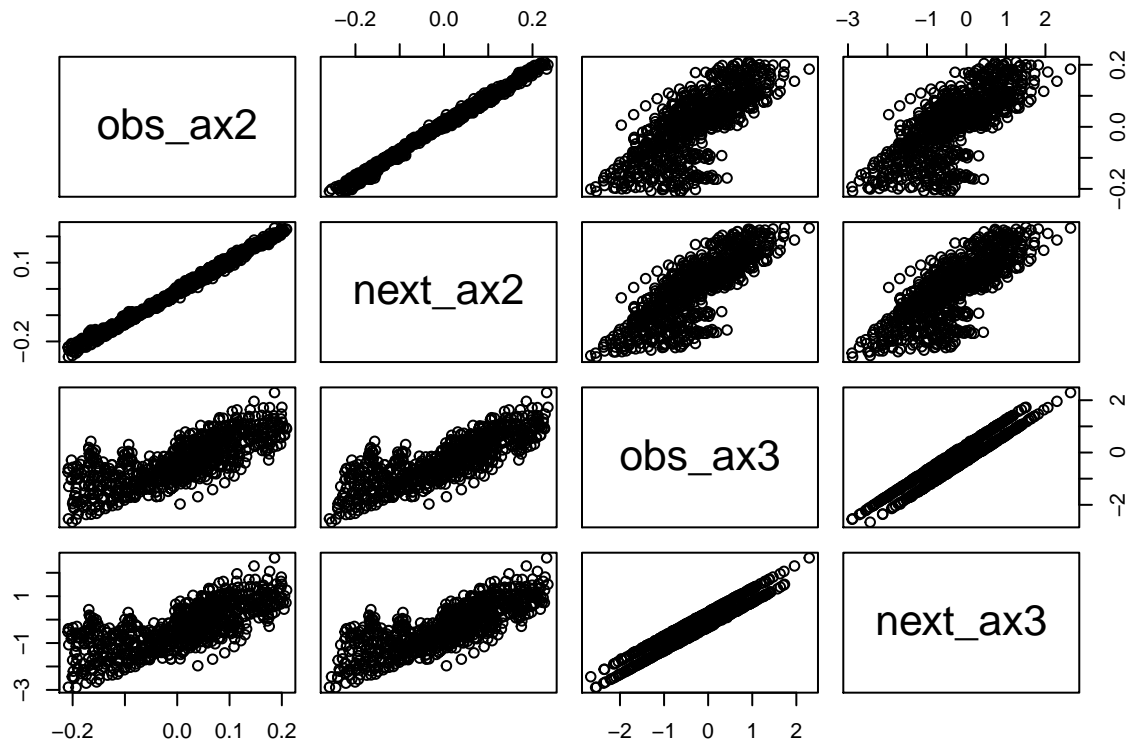
## [1] "timestep" "episode" "reward" "act" "obs_ax0" "next_ax0"
## [7] "obs_ax1" "next_ax1" "obs_ax2" "next_ax2" "obs_ax3" "next_ax3"

summary(rand_dat)

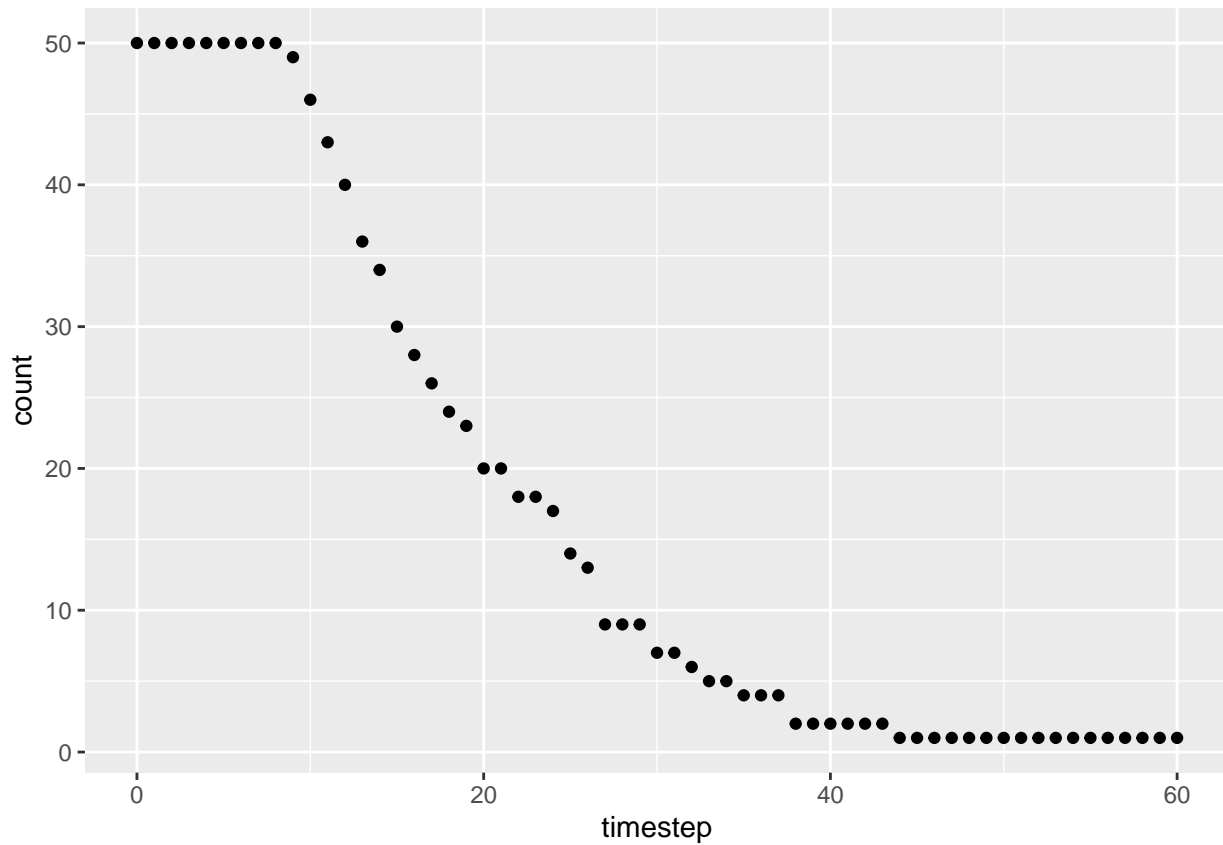
##      timestep      episode      reward      act
## Min.   : 0.00   Min.   : 0.00   Min.   : 9.00   Min.   :0.0000
## 1st Qu.: 5.00   1st Qu.:14.00   1st Qu.:16.00   1st Qu.:0.0000
## Median :10.00   Median :26.00   Median :25.00   Median :1.0000
## Mean   :12.46   Mean   :25.53   Mean   :25.92   Mean   :0.5568
## 3rd Qu.:18.00   3rd Qu.:36.00   3rd Qu.:32.00   3rd Qu.:1.0000
## Max.   :60.00   Max.   :49.00   Max.   :61.00   Max.   :1.0000
##      obs_ax0      next_ax0      obs_ax1      next_ax1
## Min.   :-0.1096541 Min.   :-0.12996 Min.   :-1.40519 Min.   :-1.6015
## 1st Qu.: -0.0156297 1st Qu.: -0.01832 1st Qu.: -0.06116 1st Qu.: -0.2266
## Median : -0.0000219 Median : 0.00432 Median : 0.16127 Median : 0.1625
## Mean   : 0.0242654 Mean   : 0.02875 Mean   : 0.22419 Mean   : 0.2461
## 3rd Qu.: 0.0597766 3rd Qu.: 0.07109 3rd Qu.: 0.55266 3rd Qu.: 0.5563
## Max.   : 0.5181503 Max.   : 0.55580 Max.   : 1.88238 Max.   : 2.0789
##      obs_ax2      next_ax2      obs_ax3      next_ax3
## Min.   :-0.20863 Min.   :-0.25873 Min.   :-2.6632 Min.   :-2.8973
## 1st Qu.: -0.03345 1st Qu.: -0.05214 1st Qu.: -0.7747 1st Qu.: -0.8118
## Median : 0.03682 Median : 0.03486 Median : -0.1940 Median : -0.2275
## Mean   : 0.01503 Mean   : 0.01036 Mean   : -0.2336 Mean   : -0.2620
## 3rd Qu.: 0.06781 3rd Qu.: 0.07017 3rd Qu.: 0.3324 3rd Qu.: 0.3572
```

```
## Max. : 0.20913 Max. : 0.23477 Max. : 2.2920 Max. : 2.6358
```

```
pairs(rand_dat[9:12])
```



```
by_timestep = rand_dat %>% group_by(timestep)
timestep_summ = summarize(by_timestep, count=n(), avg.reward=mean(reward))
ggplot(timestep_summ, aes(x=timestep, y=count)) + geom_point()
```

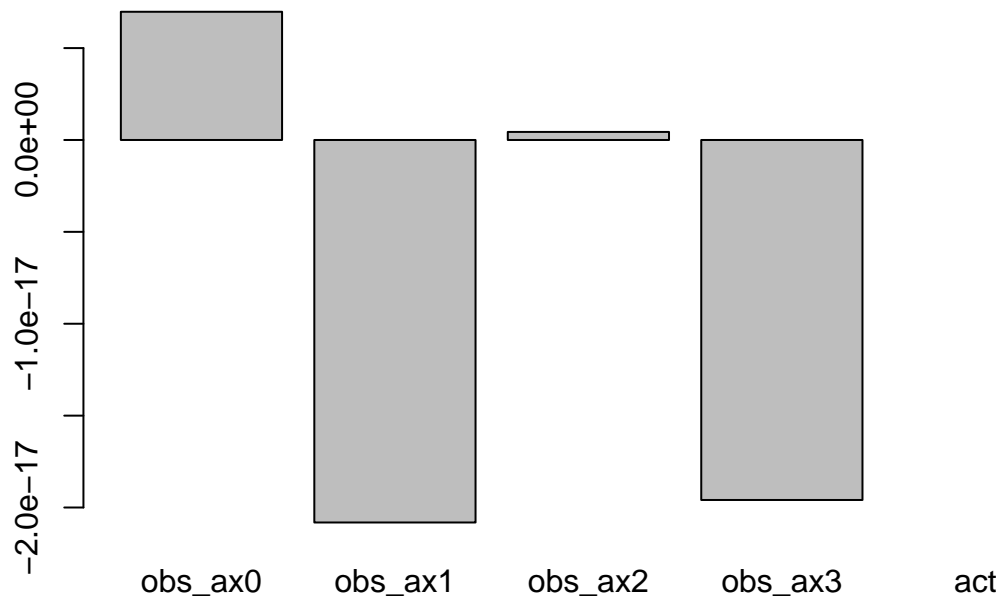


```
# Determine obs_axis with greatest difference for observation given the action
states = rand_dat %>% select(obs_ax0,obs_ax1,obs_ax2,obs_ax3,act)
sact0 = states %>% filter(act==0) %>% scale() %>% colMeans(na.rm = TRUE)
sact1 = states %>% filter(act==1) %>% scale() %>% colMeans(na.rm = TRUE)

sact0 - sact1
```

```
##      obs_ax0      obs_ax1      obs_ax2      obs_ax3      act
## 6.976043e-18 -2.081486e-17  4.376055e-19 -1.958966e-17   NaN
```

```
barplot(sact0 - sact1)
```



Exploratory Factor Analysis

```
# Maximum Likelihood Factor Analysis
# entering raw data and extracting 3 factors,
# with varimax rotation
mydata = states[1:4] # drop act
fit <- factanal(mydata, 1, rotation="varimax")
print(fit, digits=2, cutoff=.3, sort=TRUE)

##
## Call:
## factanal(x = mydata, factors = 1, rotation = "varimax")
##
## Uniquenesses:
## obs_ax0 obs_ax1 obs_ax2 obs_ax3
##    0.52    0.07    0.45    0.00
##
## Loadings:
## [1] -0.70 -0.96  0.74  1.00
##
##               Factor1
## SS loadings      2.96
## Proportion Var   0.74
##
## Test of the hypothesis that 1 factor is sufficient.
## The chi square statistic is 2763.4 on 2 degrees of freedom.
## The p-value is 0

# plot factor 1 by factor 2
#load <- fit$loadings[,1:2]
#plot(load,type="n") # set up plot
#text(load,labels=names(mydata),cex=.7) # add variable names

# fraction of the variable's total variance explained by the factor
apply(fit$loadings^2, 1, sum)
```

```

##   obs_ax0   obs_ax1   obs_ax2   obs_ax3
## 0.4842480 0.9286341 0.5504226 0.9950522

Lambda <- fit$loadings
Psi <- diag(fit$uniquenesses)
S <- fit$correlation
Sigma <- Lambda %*% t(Lambda) + Psi
# residual matrix. Numbers close to 0 indicate that our factor model is a good representation of the un
round(S - Sigma, 6)

##           obs_ax0   obs_ax1   obs_ax2   obs_ax3
## obs_ax0  0.000000  0.035391 -0.295516  0.004840
## obs_ax1  0.035391  0.000000  0.105078 -0.000630
## obs_ax2 -0.295516  0.105078  0.000000  0.005113
## obs_ax3  0.004840 -0.000630  0.005113 -0.000052

reg_fit <- factanal(mydata, factors = 1, scores = "regression")
mean(reg_fit$scores)

## [1] -9.26356e-18

sd(reg_fit$scores)

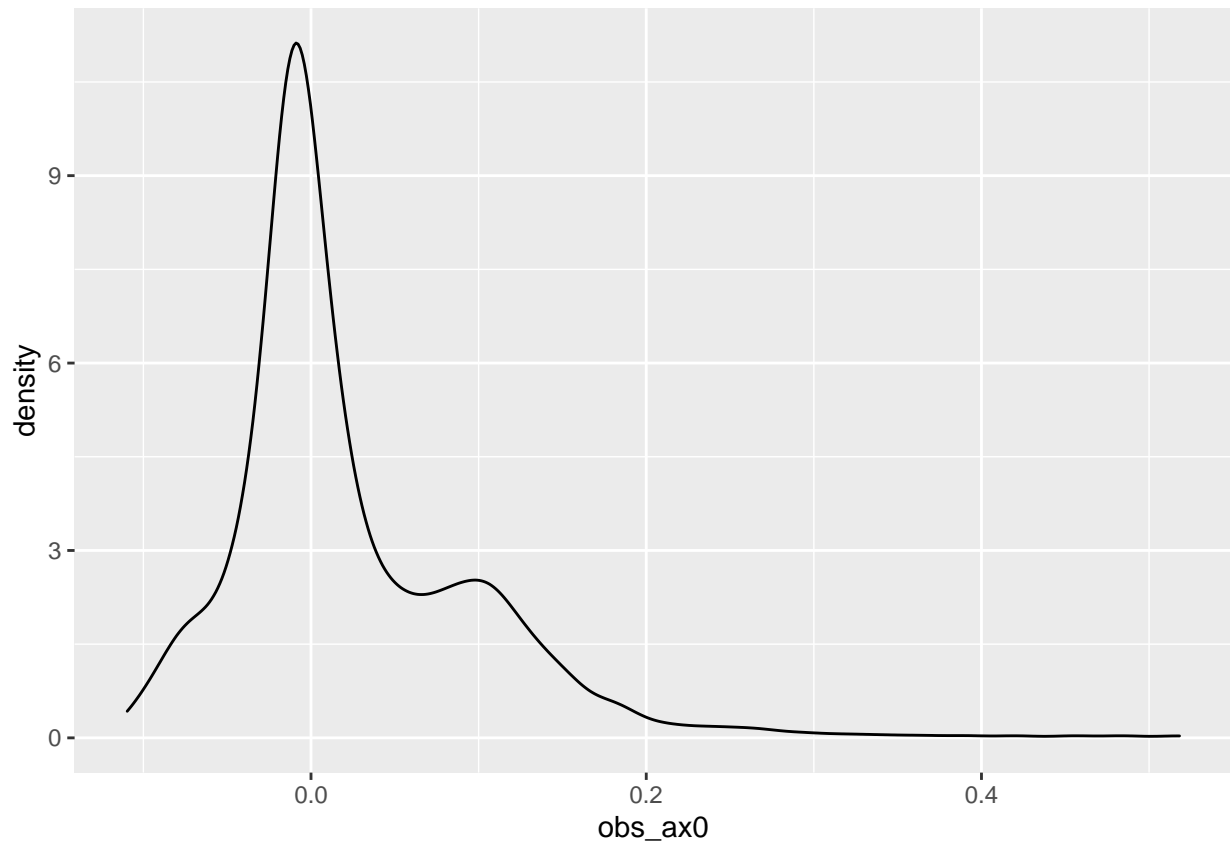
## [1] 0.9976737

reg_fit$loadings

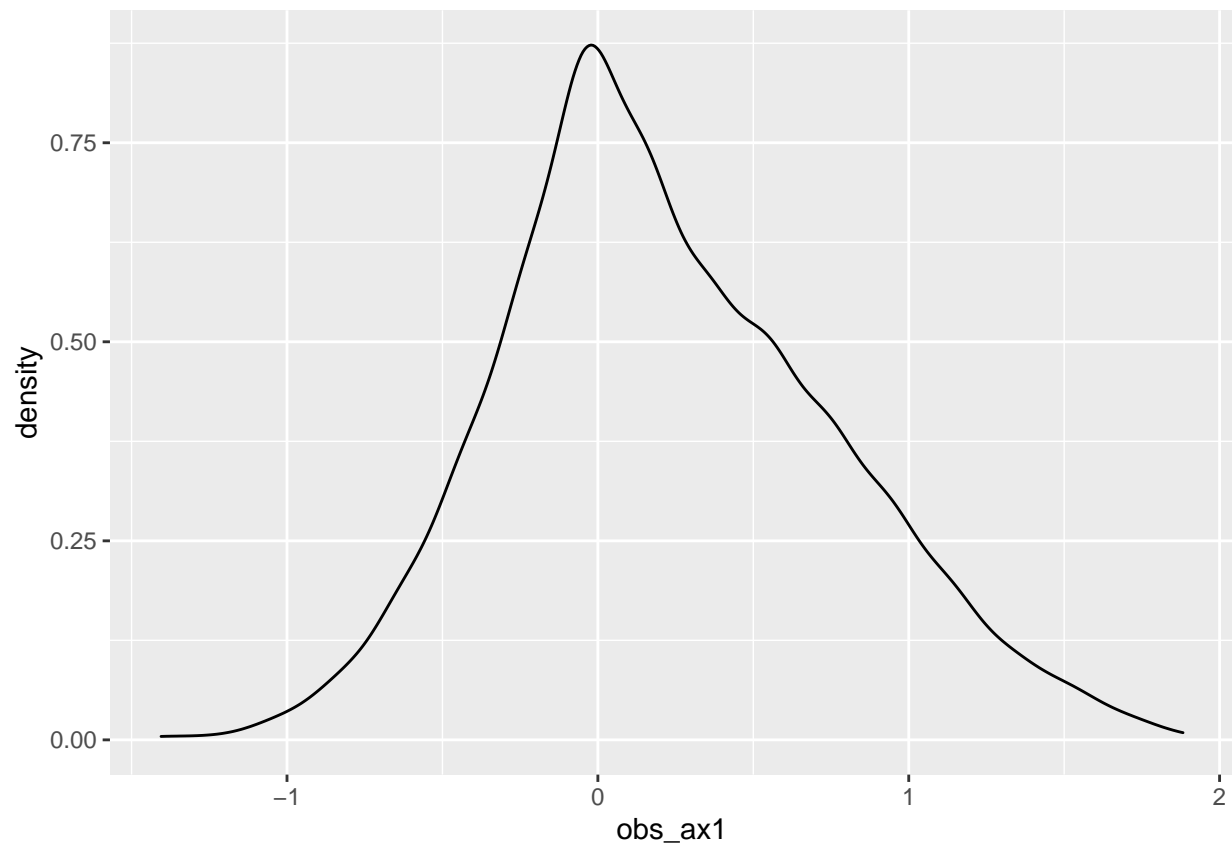
##
## Loadings:
##           Factor1
## obs_ax0 -0.696
## obs_ax1 -0.964
## obs_ax2  0.742
## obs_ax3  0.998
##
##           Factor1
## SS loadings      2.958
## Proportion Var   0.740

ggplot(states, aes(obs_ax0)) + geom_density()

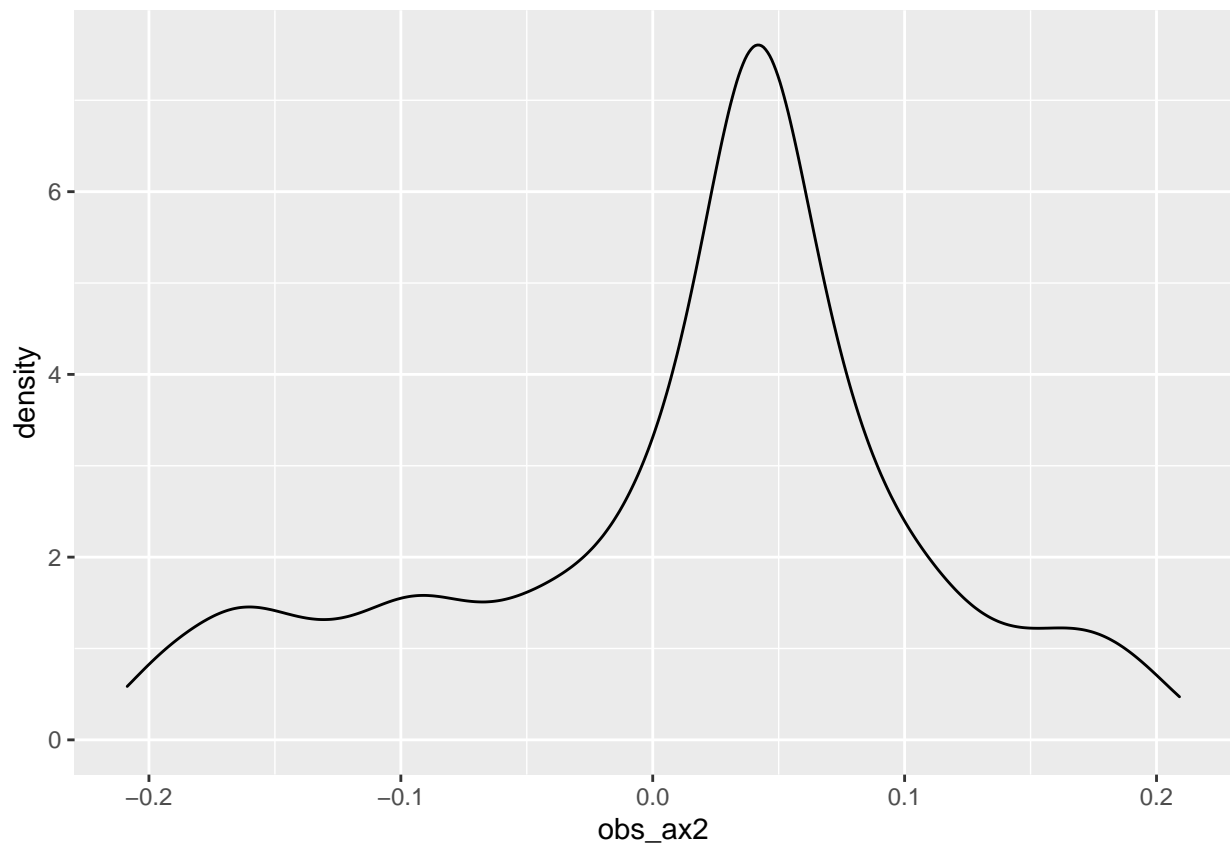
```



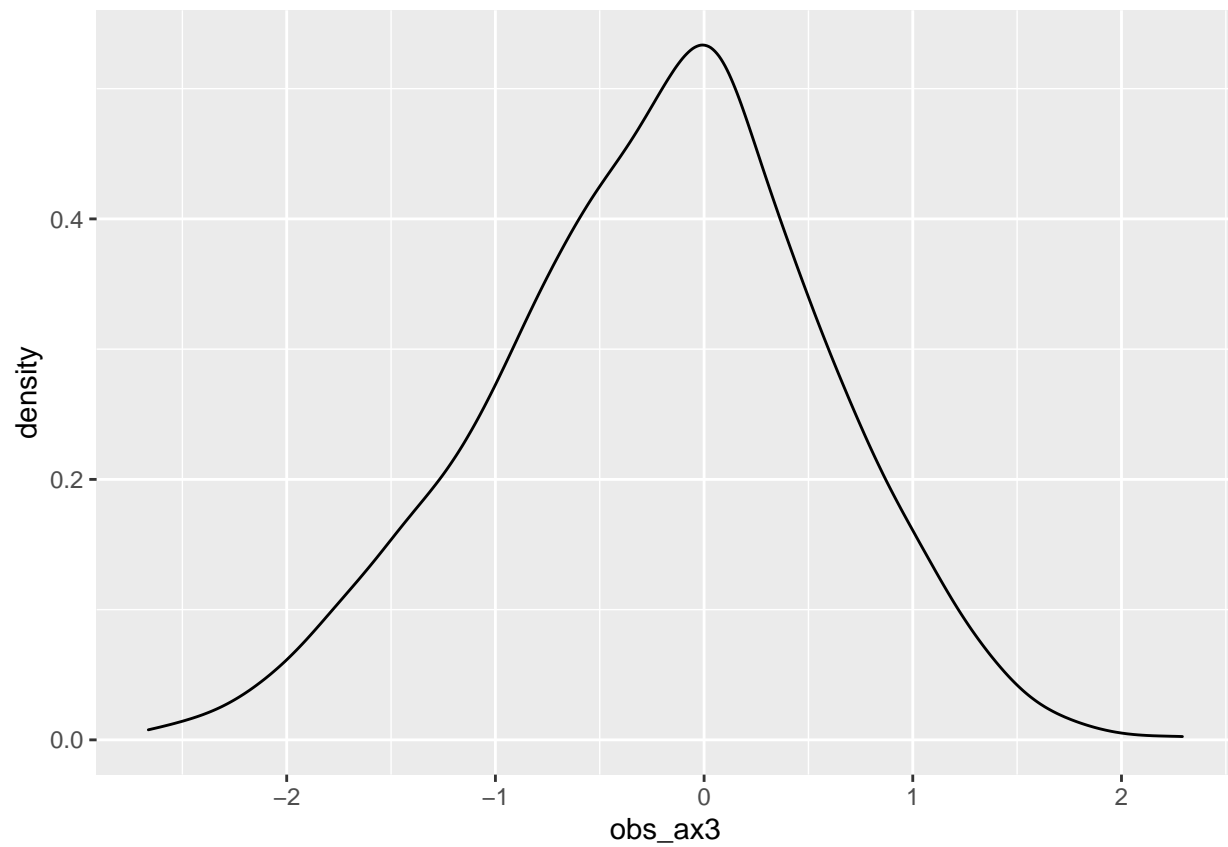
```
ggplot(states, aes(obs_ax1)) + geom_density()
```



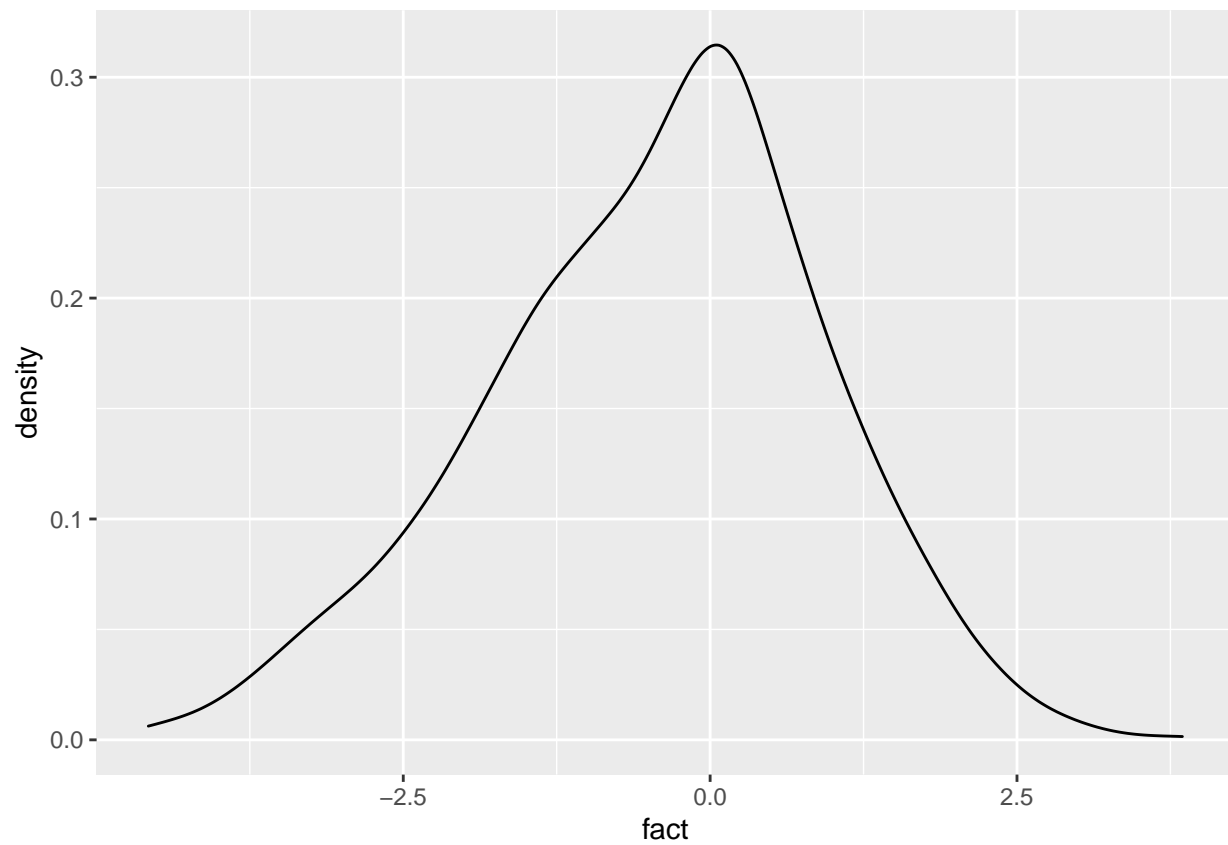
```
ggplot(states, aes(obs_ax2)) + geom_density()
```



```
ggplot(states, aes(obs_ax3)) + geom_density()
```

```
data_factor = data.frame(as.matrix(mydata) %% matrix(fit$loadings))  
names(data_factor) = c("fact")  
ggplot(data_factor, aes(fact)) + geom_density()
```



```
summary(data_factor$fact)
```

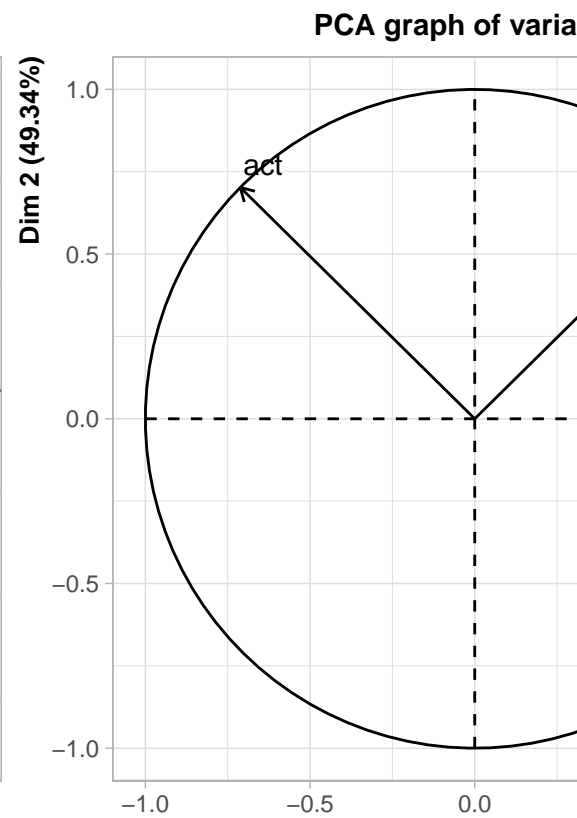
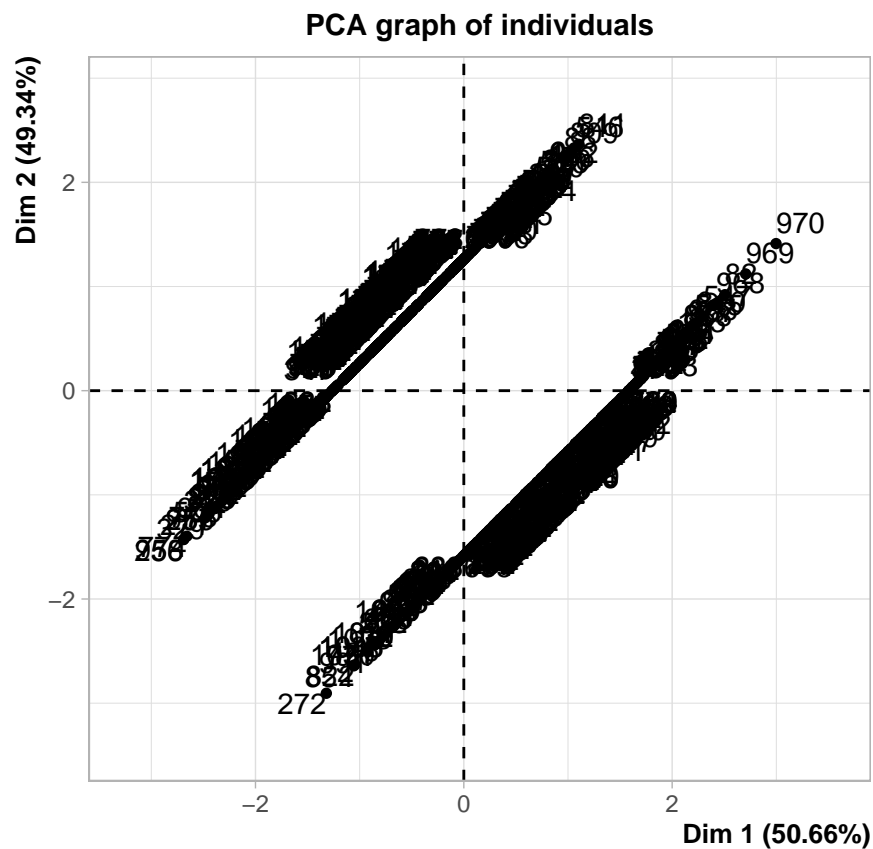
```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -4.5771 -1.3548 -0.3448 -0.4548  0.5709  3.8476
```

```
# PCA Variable Factor Map
```

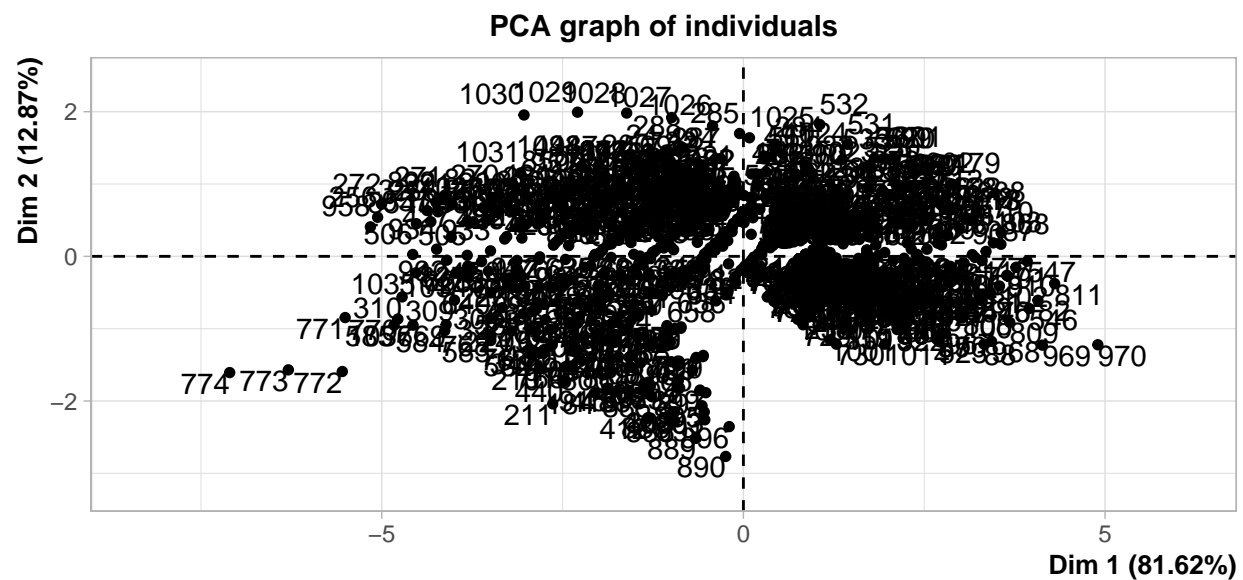
```
# NOTE: scaling has no effect on fit
```

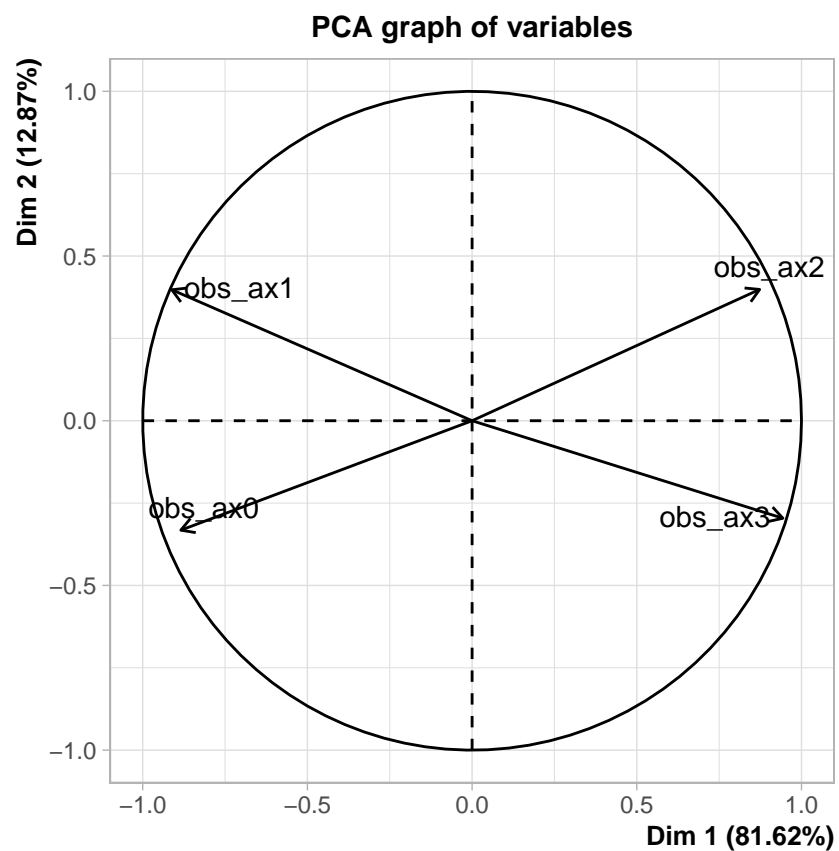
```
data_factor$fact = states$fact
```

```
res <- PCA(data_factor)
```



```
result <- PCA(mydata) # graphs generated automatically
```





```
result <- PCA(states) # graphs generated automatically
```

