

# Predict States

Nathan Shepherd

2022-03-24

```
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)

rand_state_acts <- read_csv("../utils/rand_state_acts.csv")

## Rows: 1047 Columns: 12
## -- Column specification -----
## Delimiter: ","
## dbl (12): timestep, episode, reward, act, obs_ax0, next_ax0, obs_ax1, next_a...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
names(rand_state_acts)

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

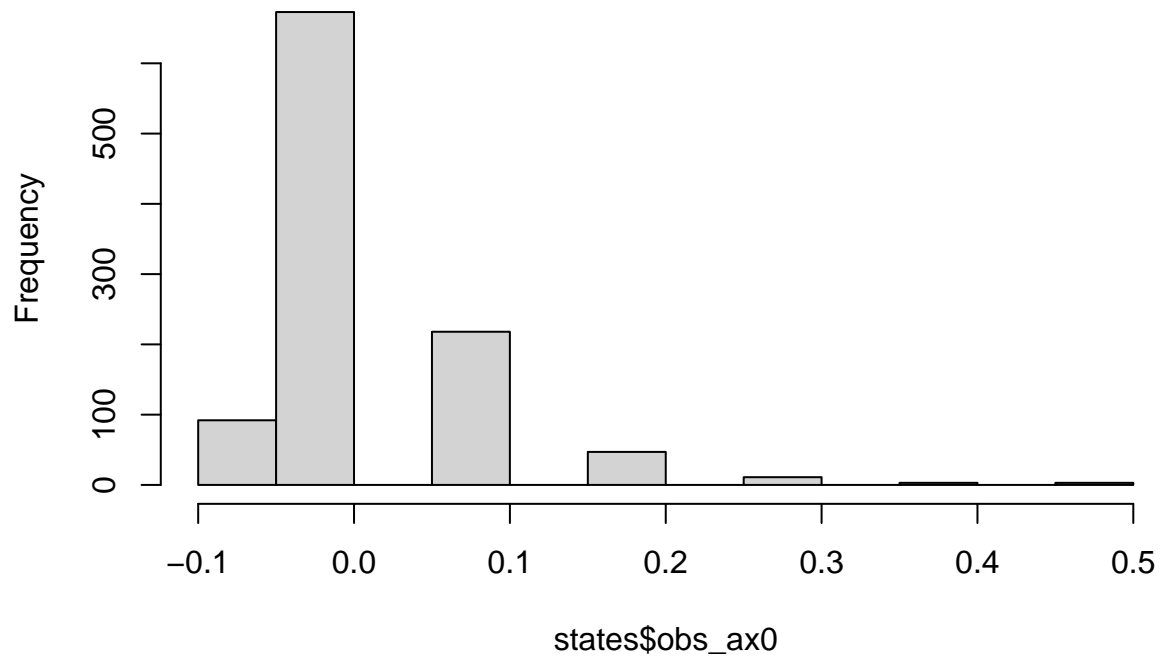
# Roughly 50% of actions should be left
summary(rand_state_acts$act)

##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.0000 0.0000  1.0000  0.5568  1.0000  1.0000

#pairs(rand_state_acts[5:12])

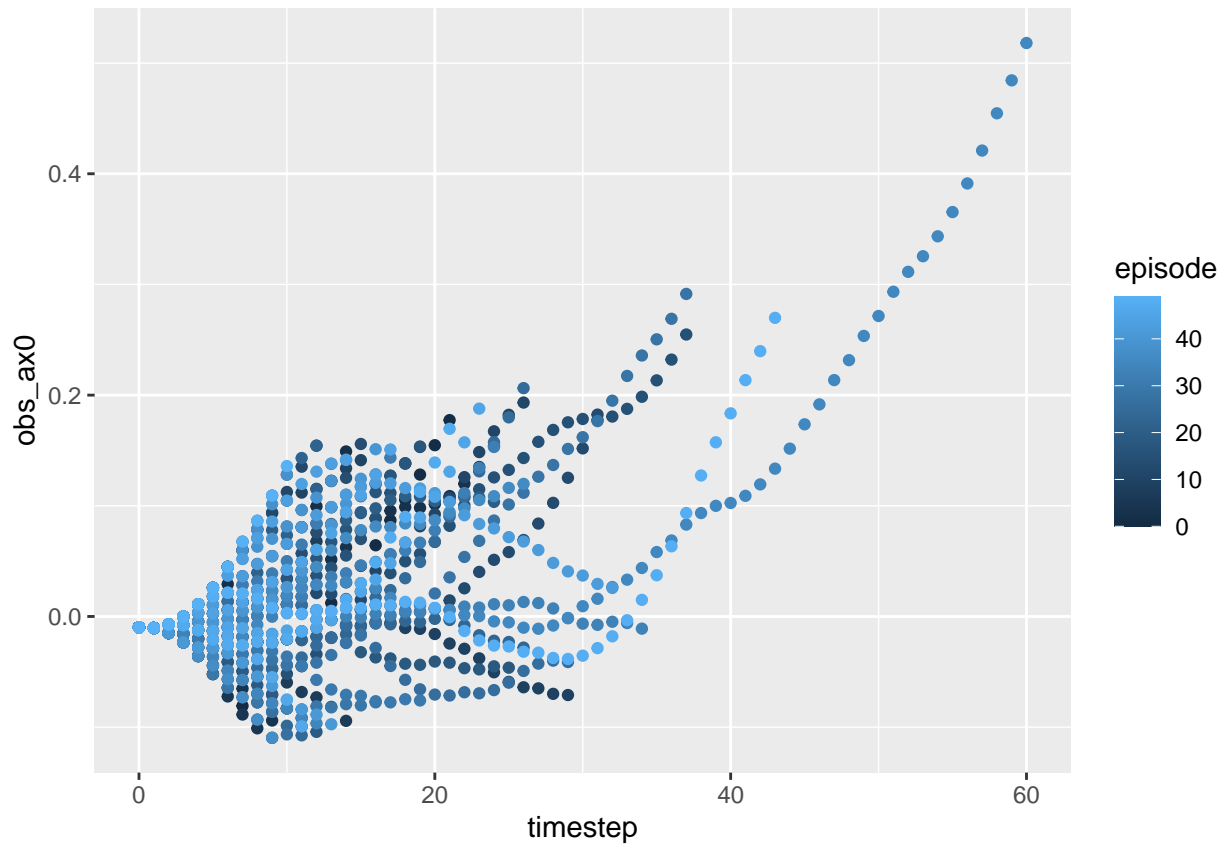
states = rand_state_acts[5:12]
states = round(states, 1)
states$act = rand_state_acts$act
hist(states$obs_ax0)
```

**Histogram of states\$obs\_ax0**

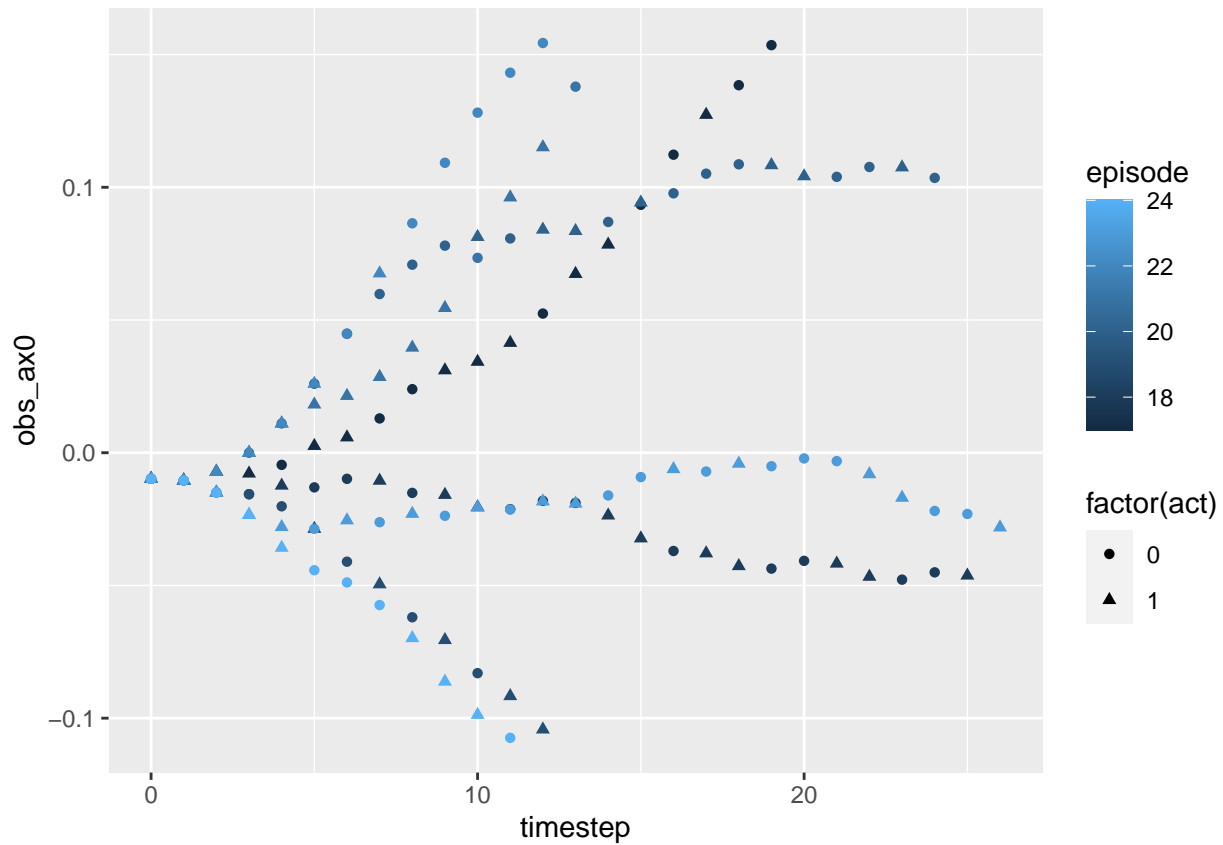


```
right_acts = filter(states, act==1)
#hist(right_acts$obs_ax0)
#pairs(states)

ggplot(rand_state_acts, aes(x=timestep, y=obs_ax0)) +
  geom_point(aes(colour=episode))
```

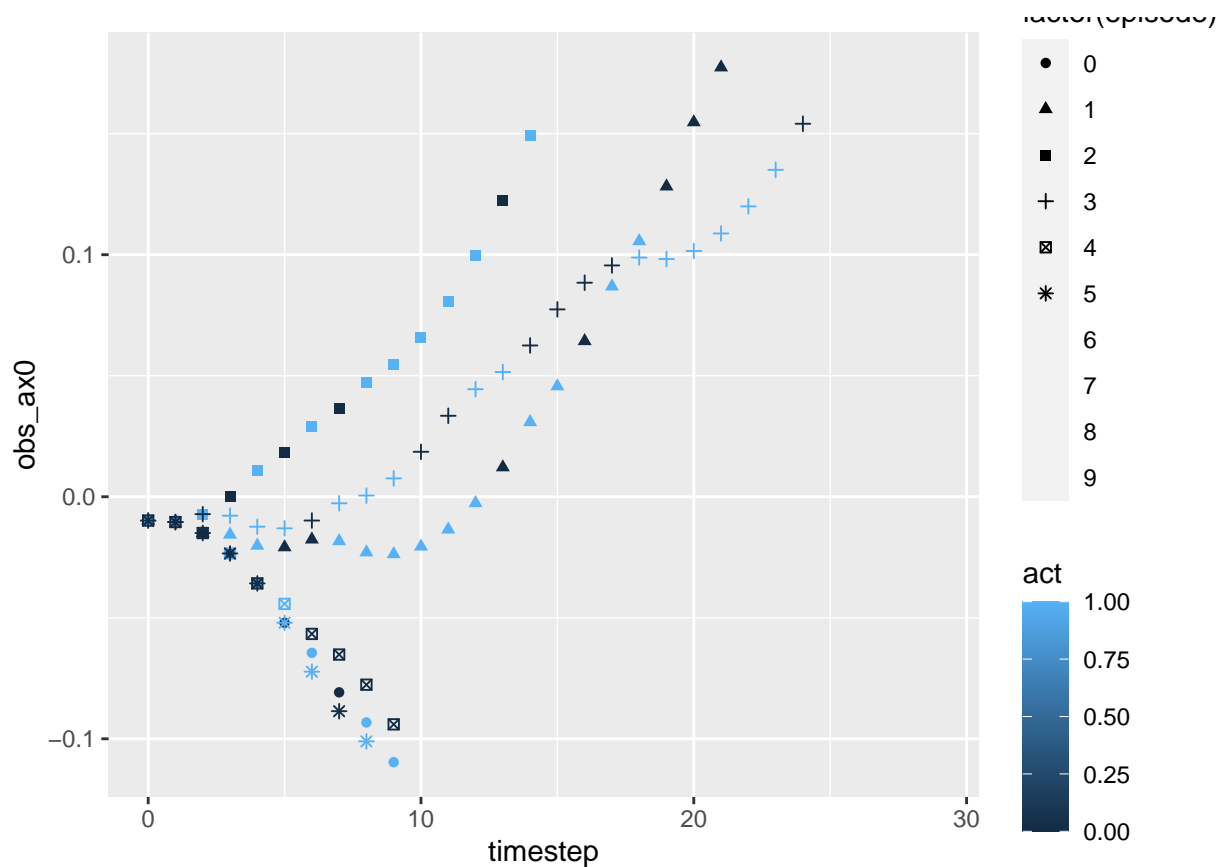


```
ggplot(filter(rand_state_acts, 16 < episode & episode < 25),  
  aes(x=timestep, y=obs_ax0, colour=episode, shape=factor(act))) +  
  geom_point()
```



```
ggplot(filter(rand_state_acts, episode<10),
  aes(x=timestep, y=obs_ax0, colour=act, shape=factor(episode))) +
  geom_point()
```

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 10. Consider
## specifying shapes manually if you must have them.
## Warning: Removed 77 rows containing missing values (geom_point).
```



```
#binaxis = "x", binwidth = .01
```

```
ax0_pred = lm(obs_ax0 ~ next_ax0 + factor(act), data=rand_state_acts)
summary(ax0_pred)
```

```
##
## Call:
## lm(formula = obs_ax0 ~ next_ax0 + factor(act), data = rand_state_acts)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.0197208 -0.0049549  0.0002368  0.0046367  0.0224242
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0017712  0.0003240  -5.467 5.73e-08 ***
## next_ax0      0.9023933  0.0025283 356.922 < 2e-16 ***
## factor(act)1  0.0001679  0.0004258   0.394   0.693
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.00684 on 1044 degrees of freedom
## Multiple R-squared:  0.9919, Adjusted R-squared:  0.9919
## F-statistic: 6.378e+04 on 2 and 1044 DF, p-value: < 2.2e-16
ax1_pred = lm(obs_ax1 ~ next_ax1 + factor(act), data=rand_state_acts)
summary(ax1_pred) # Act is a significant predictor here
```

```
##
## Call:
## lm(formula = obs_ax1 ~ next_ax1 + factor(act), data = rand_state_acts)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.0037869 -0.0003614  0.0000835  0.0004948  0.0026689
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.953e-01  4.637e-05   4211  <2e-16 ***
## next_ax1      9.987e-01  5.793e-05  17242  <2e-16 ***
## factor(act)1 -3.895e-01  6.653e-05  -5854  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0009986 on 1044 degrees of freedom
## Multiple R-squared:  1, Adjusted R-squared:  1
## F-statistic: 1.487e+08 on 2 and 1044 DF, p-value: < 2.2e-16

ax2_pred = lm(obs_ax2 ~ next_ax2 + factor(act), data=rand_state_acts)
summary(ax2_pred)
```

```
##
## Call:
## lm(formula = obs_ax2 ~ next_ax2 + factor(act), data = rand_state_acts)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.033870 -0.006088 -0.000214  0.006681  0.029404
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0058228  0.0004349   13.39  <2e-16 ***
## next_ax2      0.8770086  0.0027661  317.06  <2e-16 ***
## factor(act)1  0.0002209  0.0005816    0.38   0.704
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.009348 on 1044 degrees of freedom
## Multiple R-squared:  0.9897, Adjusted R-squared:  0.9897
## F-statistic: 5.026e+04 on 2 and 1044 DF, p-value: < 2.2e-16

ax3_pred = lm(obs_ax3 ~ next_ax3 + factor(act), data=rand_state_acts)
summary(ax3_pred) # Act is a significant predictor here
```

```
##
## Call:
## lm(formula = obs_ax3 ~ next_ax3 + factor(act), data = rand_state_acts)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.047750 -0.011018 -0.001487  0.008131  0.070536
##
## Coefficients:
```

```
##           Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.2940545  0.0008783  -334.8  <2e-16 ***
## next_ax3     0.9732805  0.0007158  1359.7  <2e-16 ***
## factor(act)1  0.5665373  0.0012500   453.2  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01889 on 1044 degrees of freedom
## Multiple R-squared:  0.9994, Adjusted R-squared:  0.9994
## F-statistic: 9.244e+05 on 2 and 1044 DF,  p-value: < 2.2e-16
plot(rand_state_acts$episode, rand_state_acts$timestep)
```

