JAGS Model

Tom Park

2020-02-08

Basic Model: Ambulance + Overdose

Ambulance Call-outs Model

n_A: sample size
x_A: the total number who confirmed they did call an ambulance
p_A: probability of a person call an ambulance

$$x_A \sim Bin(n_A, p_A)$$

We assume $n_A = 1000, p_A = 0.8$.
Suppose the prior of p_A is noninformative.

$$p(p_A) \sim Beta(1,1)$$

Overdose Model

Now we plug in this values into the overdose model and obtain possible O_t values **assuming we have U_t values.**

Also, we have priors.

$$z_t \sim N(\mu, \sigma^2)$$
$$\lambda_t^{OD} = \exp(z_t)$$
$$O_t \sim Poi(\lambda_t^{OD} N)$$
$$U_t \sim Bin(O_t, p_A)$$

For simplicity we set N =10000 for now. We need to generate reasonable U_t values first. Note that U_t comes from μ, σ following all the way through the overdose model.
$\mu = \log 0.05, \sigma = 1, N = 10000$.
We suppose survey data exists: (n_A, x_A) known.

We set for our prior parameters:

$$\mu \sim U(-10, 0)$$
$$\sigma \sim U(0, 5)$$

```r
# install packages
if (!require(rjags)) install.packages("rjags", dependencies = TRUE)
if (!require(coda)) install.packages("coda", dependencies = TRUE)
if (!require(tidyverse)) install.packages("tidyverse", dependencies = TRUE)
if (!require(tinytex)) install.packages("tinytex", dependencies = TRUE)

library('rjags')
library('coda')
library('tidyverse')
library('tinytex')
```

The data is the same data from pymc3 with Python.
Todo: build a pipeline to connect the python (pymc3) and R (JAGS)

```r
df <- read.csv('./basic_data.csv')
df$X <- NULL
head(df)
```

```
##    o_t  u_t x_a
## 1 2475 1969 799
## 2  262  217 798
## 3  318  253 795
## 4  149  119 816
## 5 1151  934 805
## 6   39   34 794
```

Now we set the model which defines the relations of overdose model and ambulance call model.

The model defined as follows.

```r
cat("model{
## define the priors

p_a ~ dbeta(alpha, beta)
mu_z ~ dunif(mu_a, mu_b)
sigma_z ~ dunif(sigma_a, sigma_b)


 ## ambulance model
for (i in 1:n) {
  #Likelihood
  x_a[i] ~ dbin(p_a, n_a) # each survey result for month

}

# overdose
for (i in 1:n) {

  ## the latent variables
  z_t[i]~ dnorm(mu_z, 1/(sigma_z^2))
  lmb_t[i] <- exp(z_t[i])

  ## overdose model
  o_t[i] ~ dpois(lmb_t[i]*N) # total overdoses per month
  # Note that from pymc3 gamma was used instead of Pois dist
  u_t[i] ~ dbin(p_a, o_t[i]) # ambulanced overdoses per month
}

}", file='basic_model.txt')
```

Pre-set variables.

```r
n_T <- length(df$o_t)
n_a <- 1000
N <- 10000
u_t <- df$u_t
x_a <- df$x_a
```

Define the list providing the values of the variables and the parameters for the priors of the model.

```r
dat <- list(
            # priors for ambulance model
            'alpha' = 1,
            'beta' =  1,

            # priors for overdose model
            'mu_a'=(-10),
            'mu_b'=0,
            'sigma_a'=0,
            'sigma_b'=5,

            # likelihood
            'u_t'=u_t, # giving data
            'x_a'=x_a,   # giving data
            'N'=N, # the population 10000
            'n'= n_T, # total months 12
            'n_a'=n_a # survey size 1000

            )
```

Note: for the list object usually named 'data' or 'dat' in JAGS context, do not use arrow but use equal sign to define elements of the list.

```r
chains=2
# inits = list()
simple.model <- jags.model(file='basic_model.txt',
                           data=dat,
                           n.chains = chains)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 24
##    Unobserved stochastic nodes: 27
##    Total graph size: 88
##
## Initializing model
```

```r
simple.model
```

```
## JAGS model:
##
## model{
## ## define the priors
##
## p_a ~ dbeta(alpha, beta)
## mu_z ~ dunif(mu_a, mu_b)
## sigma_z ~ dunif(sigma_a, sigma_b)
##
##
##  ## ambulance model
## for (i in 1:n) {
##   #Likelihood
##   x_a[i] ~ dbin(p_a, n_a) # each survey result for month
```

```
##
## }
##
## # overdose
## for (i in 1:n) {
##
##   ## the latent variables
##   z_t[i]~ dnorm(mu_z, 1/(sigma_z^2))
##   lmb_t[i] <- exp(z_t[i])
##
##   ## overdose model
##   o_t[i] ~ dpois(lmb_t[i]*N) # total overdoses per month
##   # Note that from pymc3 gamma was used instead of Pois dist
##   u_t[i] ~ dbin(p_a, o_t[i]) # ambulanced overdoses per month
## }
##
## }
## Fully observed variables:
##  N alpha beta mu_a mu_b n n_a sigma_a sigma_b u_t x_a
```

O_t

```
params= c('o_t','p_a')
samples <- coda.samples(simple.model, params, n.iter = 1000)
# guess it's getting posterior samples ?
```

```
interations = 1000
burnin= floor(interations/2)
summary(window(samples), start=burnin)
```

```
##
## Iterations = 1001:2000
## Thinning interval = 1
## Number of chains = 2
## Sample size per chain = 1000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##               Mean        SD  Naive SE Time-series SE
## o_t[1]  2477.4060 27.802746 6.217e-01      0.9911877
## o_t[2]   273.1270  8.638934 1.932e-01      0.2534807
## o_t[3]   317.9915  9.023976 2.018e-01      0.2493259
## o_t[4]   150.2440  6.326648 1.415e-01      0.1906376
## o_t[5]  1175.5660 17.962683 4.017e-01      0.5026958
## o_t[6]    42.9835  3.292666 7.363e-02      0.0896641
## o_t[7]  3009.8790 31.378251 7.016e-01      1.2014412
## o_t[8]   246.4995  8.131486 1.818e-01      0.2350357
## o_t[9]   715.8720 13.974271 3.125e-01      0.4215083
## o_t[10]  387.8660 10.265073 2.295e-01      0.2874126
## o_t[11] 2082.5475 25.846562 5.779e-01      0.8717076
## o_t[12]   69.5490  4.254368 9.513e-02      0.1198737
## p_a         0.7945  0.003794 8.483e-05      0.0001644
##
```

```
## 2. Quantiles for each variable:
##
##              2.5%      25%       50%      75%     97.5%
## o_t[1]  2424.0000 2459.000 2478.0000 2495.000 2533.0000
## o_t[2]   257.0000  267.000  273.0000  279.000  290.0000
## o_t[3]   302.0000  312.000  318.0000  324.000  337.0000
## o_t[4]   138.9750  146.000  150.0000  154.000  163.0000
## o_t[5]  1141.0000 1163.000 1176.0000 1188.000 1211.0000
## o_t[6]    37.0000   41.000   43.0000   45.000   50.0000
## o_t[7]  2950.9750 2988.750 3010.0000 3030.000 3076.0250
## o_t[8]   232.0000  241.000  246.0000  252.000  263.0000
## o_t[9]   690.0000  706.000  716.0000  725.000  743.0000
## o_t[10]  368.9750  381.000  387.5000  395.000  409.0000
## o_t[11] 2030.0000 2065.000 2082.0000 2099.000 2134.0250
## o_t[12]   62.0000   67.000   69.0000   72.000   79.0000
## p_a        0.7867    0.792    0.7946    0.797    0.8019
```

q1: what is the equivalent plot that we can see we have enough iteration?

Boxplots of O_t

q2: I see two elements from the samples list. Which one I should use it or should I use both?

```r
pst_mtx = as.matrix(samples)
temp = pst_mtx[,1:12]
p_a <- pst_mtx[,13]
head(temp)
```

```
##      o_t[1] o_t[2] o_t[3] o_t[4] o_t[5] o_t[6] o_t[7] o_t[8] o_t[9]
## [1,]   2442    292    315    157   1194     42   3036    263    702
## [2,]   2525    300    317    151   1194     45   2999    240    720
## [3,]   2487    281    316    155   1168     42   3016    246    729
## [4,]   2561    272    298    168   1184     48   3024    250    718
## [5,]   2497    275    326    155   1155     46   2990    241    708
## [6,]   2517    272    323    163   1146     44   3032    239    722
##      o_t[10] o_t[11] o_t[12]
## [1,]     373    2095      70
## [2,]     371    2125      74
## [3,]     372    2078      79
## [4,]     382    2082      69
## [5,]     396    2140      67
## [6,]     383    2086      64
```

```r
length(p_a)
```

```
## [1] 2000
```

```r
colnames(temp) <- seq(1,12)
df_o_t <- as.data.frame(temp)
head(df_o_t)
```

```
##      1    2    3    4     5  6     7    8    9  10    11 12
```

```
## 1 2442 292 315 157 1194 42 3036 263 702 373 2095 70
## 2 2525 300 317 151 1194 45 2999 240 720 371 2125 74
## 3 2487 281 316 155 1168 42 3016 246 729 372 2078 79
## 4 2561 272 298 168 1184 48 3024 250 718 382 2082 69
## 5 2497 275 326 155 1155 46 2990 241 708 396 2140 67
## 6 2517 272 323 163 1146 44 3032 239 722 383 2086 64
```

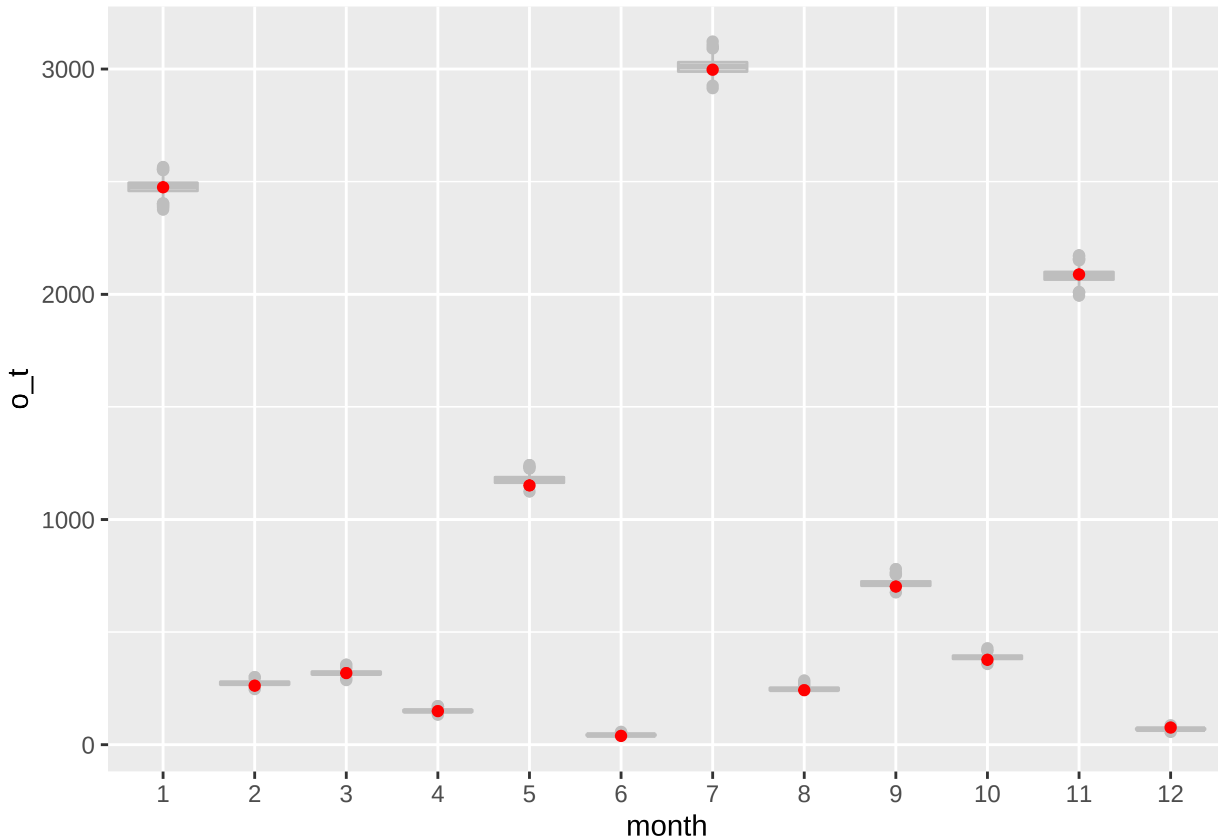
```r
trace_o_t <- gather(df_o_t, key = 'month',value = 'o_t')
trace_o_t$month <- factor(trace_o_t$month,levels = seq(1,12))
str(trace_o_t)
```

```
## 'data.frame':    24000 obs. of  2 variables:
##  $ month: Factor w/ 12 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ o_t  : num  2442 2525 2487 2561 2497 ...
```

```r
head(trace_o_t,n = 24)
```

```
##    month  o_t
## 1      1 2442
## 2      1 2525
## 3      1 2487
## 4      1 2561
## 5      1 2497
## 6      1 2517
## 7      1 2468
## 8      1 2471
## 9      1 2460
## 10     1 2497
## 11     1 2460
## 12     1 2509
## 13     1 2509
## 14     1 2544
## 15     1 2520
## 16     1 2481
## 17     1 2500
## 18     1 2474
## 19     1 2466
## 20     1 2522
## 21     1 2554
## 22     1 2486
## 23     1 2515
## 24     1 2497
```

```r
# real values of the data set from pymc3 samples
o_t_values=data.frame('month'=seq(1,12),'o_t'=df$o_t)
# factorizing the months for box plot visualization
o_t_values$month <- factor(o_t_values$month,levels = seq(1,12))

ggplot()+
  # boxplot from the trace
  geom_boxplot(aes(x=month,y=o_t), color='grey',data = trace_o_t)+
  # real values as red dots
  geom_point(aes(x=o_t_values$month, y=o_t_values$o_t),color='red')
```



Predictive Posterior Checks

```
df_u_t <- data.frame()
m=length(p_a)
for (i in 1:m) {
  obs <- rbinom(n=12,size = as.numeric(df_o_t[i,]),prob =p_a[i])
  df_u_t=rbind(df_u_t,obs)
}
colnames(df_u_t) <- factor(seq(1,12),levels = seq(1,12))
str(df_u_t)
```

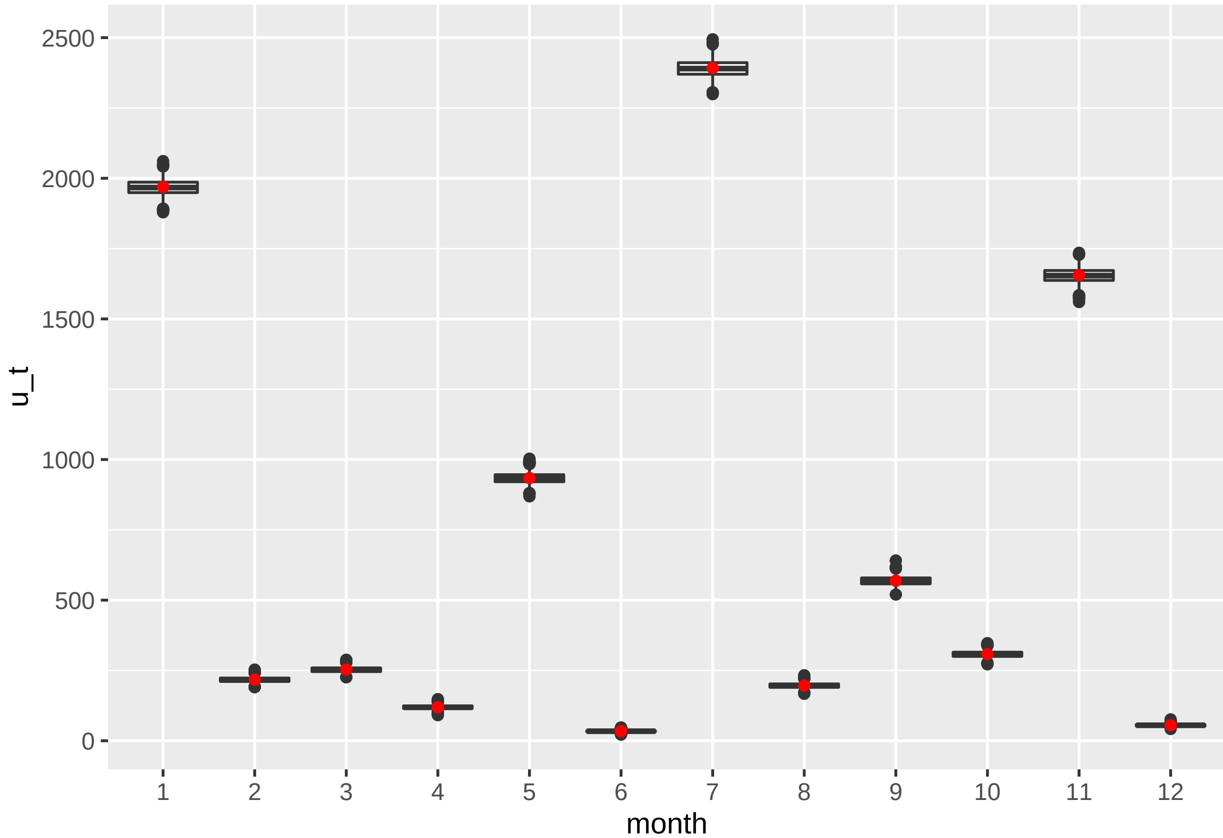
```
## 'data.frame':    2000 obs. of  12 variables:
##  $ 1 : int  1879 2018 1943 2017 1961 1998 1956 1965 1965 1978 ...
##  $ 2 : int  223 239 218 224 218 220 208 213 216 202 ...
##  $ 3 : int  253 254 261 233 256 254 251 258 254 239 ...
##  $ 4 : int  129 120 121 136 122 125 118 119 117 126 ...
##  $ 5 : int  940 937 924 967 924 918 919 947 947 920 ...
##  $ 6 : int  36 33 31 39 35 36 36 35 31 35 ...
##  $ 7 : int  2391 2382 2373 2389 2361 2359 2412 2326 2373 2332 ...
##  $ 8 : int  221 192 190 204 194 181 195 195 198 193 ...
##  $ 9 : int  568 565 603 557 568 569 572 584 574 543 ...
##  $ 10: int  295 297 287 288 312 320 304 310 325 307 ...
##  $ 11: int  1652 1687 1625 1642 1680 1697 1688 1674 1645 1647 ...
##  $ 12: int  55 59 61 56 55 49 49 50 47 54 ...
```

U_t: Predictive Posterior Checks

```
ppc_u_t <- gather(df_u_t, key = 'month',value = 'u_t')
summary(ppc_u_t)
```

```
##     month                u_t
##  Length:24000       Min.   :  22.0
##  Class :character   1st Qu.: 162.2
##  Mode  :character   Median : 279.5
##                     Mean   : 724.8
##                     3rd Qu.:1142.2
##                     Max.   :2494.0
```

```
u_t_values=data.frame('month'=seq(1,12),'u_t'=df$u_t)
# u_t_values$month <- factor(u_t_values$month,levels = seq(1,12))
ppc_u_t$month <- factor(ppc_u_t$month,levels = seq(1,12))

ggplot()+geom_boxplot(aes(x=month,y=u_t),data = ppc_u_t)+geom_point(aes(x=u_t_values$month, y=u_t_values
```



todo: finish this part.

x_A: Predictive Posterior Checks

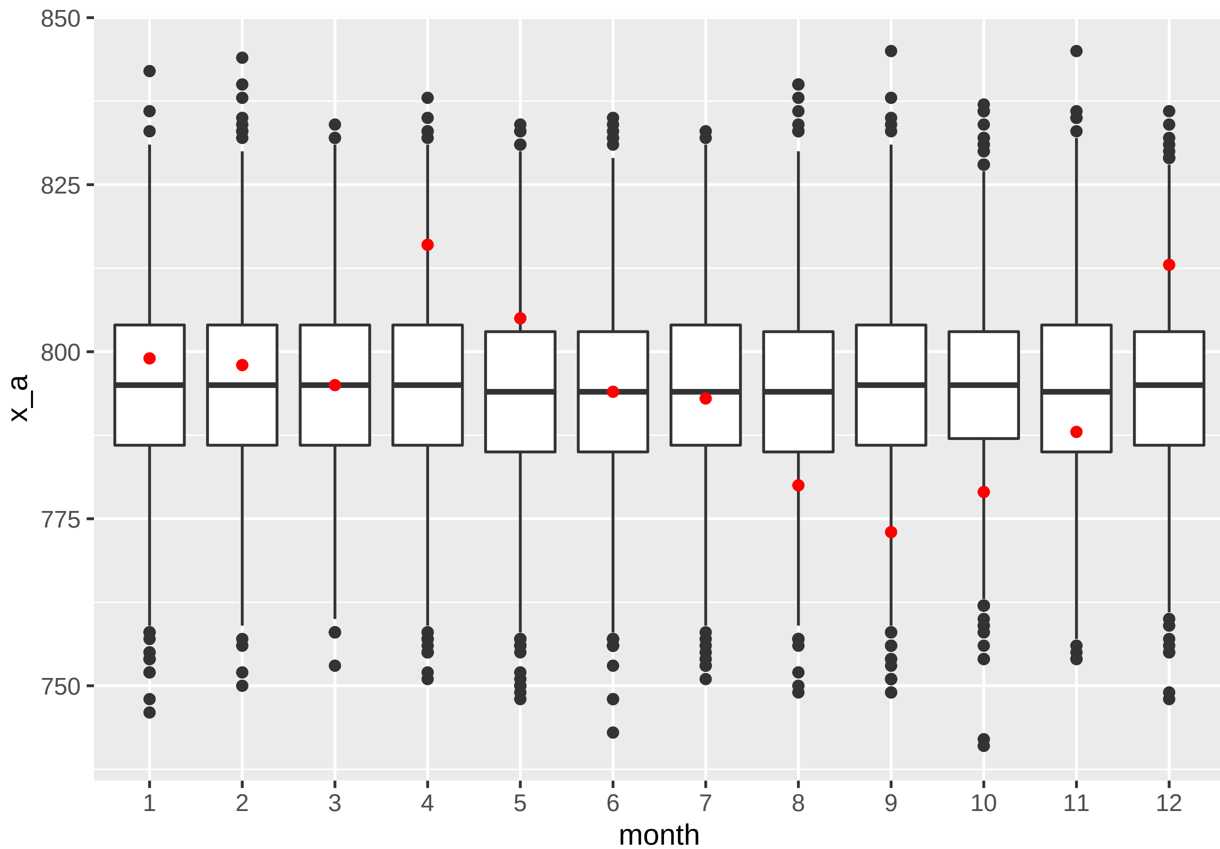
```
df_x_a <- vector()
for (i in 1:n_T) {
  obs <- rbinom(m,n_a,p_a)
  df_x_a <- cbind(df_x_a,obs)
```

```
}

head(df_x_a)
```

```
##      obs obs obs obs obs obs obs obs obs obs obs obs
## [1,] 791 763 797 807 784 783 768 783 779 775 785 773
## [2,] 789 804 789 800 804 799 820 794 793 794 807 789
## [3,] 794 781 779 776 809 803 794 795 817 791 789 789
## [4,] 776 794 769 783 799 778 772 796 792 814 786 777
## [5,] 803 818 789 808 792 796 804 805 793 759 815 785
## [6,] 802 800 809 799 800 808 796 801 826 798 781 801
```

```
colnames(df_x_a) <- seq(1:12)
```

```
df_x_a <- as.data.frame(df_x_a)
ppc_x_a <- gather(df_x_a, key = 'month',value = 'x_a')


ppc_x_a$month <- factor(ppc_x_a$month,levels = seq(1,12))


x_a_values=data.frame('month'=seq(1,12),'x_a'=df$x_a)
x_a_values$month <- factor(x_a_values$month,levels = seq(1,12))

ggplot()+geom_boxplot(aes(x=month,y=x_a),data = ppc_x_a)+geom_point(aes(x=x_a_values$month, y=x_a_value
```



Contamination of p_A

Now, suppose the survey data gives us a wrong (biased) p_A value.

$\text{Bias} = \theta - \hat{\theta} = p_A - \hat{p}_A$
$\hat{p}_A = p_A + bias(p_A)$

Three more data sets are given: unbiased, overestimated, underestimated p_A.

```
## write a function that led to compare o_t, u_t and x_a.
#first we need a fuction that gives us data, model, trace, and ppc.
test_robust <- function(file=file, random=1, N=10000, p_a=0.8, bias = -0.2, n_a=1000, n_T=12) {
  df <- read.csv(file = file)
  df$X <- NULL
  df$month <- seq(1:12)
# obtain the (biased) data
  dat <- list(
            # priors for ambulance model
            'alpha' = 1,
            'beta' =  1,

            # priors for overdose model
            'mu_a'=(-10),
            'mu_b'=0,
            'sigma_a'=0,
            'sigma_b'=5,

            # likelihood
            'u_t'=df$u_t, # giving data
            'x_a'=df$x_a,   # giving data
            'N'=N, # the population 10000
            'n'= n_T, # total months 12
            'n_a'=n_a # survey size 1000

            )
# run the model
  chains=2
  # target 1 to save
  simple.model <- jags.model(file='basic_model.txt',
                           data=dat,
                           n.chains = chains)

# get the samples of O_t, p_a
  params= c('o_t','p_a')
  # target 2 to save
  samples <- coda.samples(simple.model, params, n.iter = 2000)
  pst_mtx = as.matrix(samples)
  # tidy p_a: target 3
  p_a <- pst_mtx[,13]

  ## tidy o_t
  temp = pst_mtx[,1:12]
  colnames(temp) <- seq(1,12)
  df_o_t <- as.data.frame(temp)
  trace_o_t <- gather(df_o_t, key = 'month',value = 'o_t')
```

```r
  trace_o_t$month <- factor(trace_o_t$month,levels = seq(1,12))

  # tidy u_t pp samples
  df_u_t <- data.frame()
  m = length(p_a)
  for (i in 1:m) {
   obs <- rbinom(n=12,size = as.numeric(df_o_t[i,]),prob =p_a[i])
   df_u_t=rbind(df_u_t,obs)
  }
  colnames(df_u_t) <- factor(seq(1,12),levels = seq(1,12))
  ppc_u_t <- gather(df_u_t, key = 'month',value = 'u_t')
  ppc_u_t$month <- factor(ppc_u_t$month,levels = seq(1,12))

  ## tidy x_a
  df_x_a <- vector()
  for (i in 1:n_T) {
    obs <- rbinom(m,n_a,p_a)
    df_x_a <- cbind(df_x_a,obs)
  }
  colnames(df_x_a) <- seq(1:12)
  df_x_a <- as.data.frame(df_x_a)
  ppc_x_a <- gather(df_x_a, key = 'month',value = 'x_a')
  ppc_x_a$month <- factor(ppc_x_a$month,levels = seq(1,12))
  ppc = list('u_t'=ppc_u_t, 'x_a'=ppc_x_a)
  mylist=list('data'=df,'model'=simple.model,'trace'=trace_o_t,'ppc'=ppc)
  return (mylist)
}

# then visualization function which gives us three different types of boxplots.
```

```r
my_list_unbiased = test_robust(file = './basic_data.csv', bias =0)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 24
##    Unobserved stochastic nodes: 27
##    Total graph size: 88
##
## Initializing model
```

```r
my_list_under = test_robust(file = './under_p_a_data.csv',bias=-0.2)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 24
##    Unobserved stochastic nodes: 27
##    Total graph size: 88
##
## Initializing model
```

```r
my_list_over = test_robust(file='./over_p_a_data.csv',bias = +0.1)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 24
##    Unobserved stochastic nodes: 27
##    Total graph size: 88
##
## Initializing model
```

```r
print(my_list_unbiased$data$u_t)
```

```
##  [1] 1969  217  253  119  934   34 2392  196  569  308 1655   55
```

```r
print(my_list_under$data$u_t)
```

```
##  [1] 1969  217  253  119  934   34 2392  196  569  308 1655   55
```

```r
print(my_list_over$data$u_t)
```

```
##  [1] 1969  217  253  119  934   34 2392  196  569  308 1655   55
```

```r
print(my_list_unbiased$data$x_a)
```

```
##  [1] 799 798 795 816 805 794 793 780 773 779 788 813
```

```r
print(my_list_under$data$x_a)
```

```
##  [1] 602 598 622 608 595 593 575 638 586 618 574 582
```

```r
print(my_list_over$data$x_a)
```

```
##  [1] 898 891 910 902 894 921 881 909 879 913 908 914
```

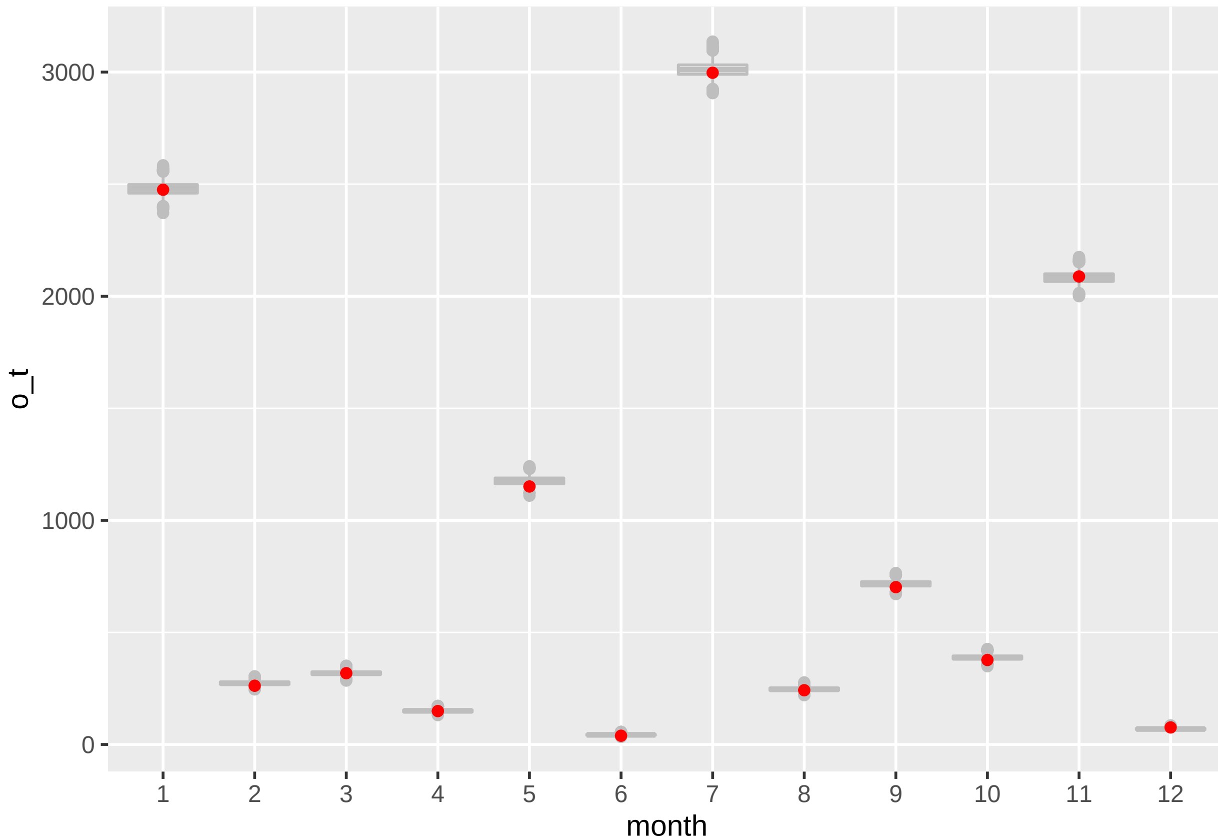
```r
 head(my_list_unbiased$ppc$u_t)
```

```
##   month  u_t
## 1     1 1955
## 2     1 1928
## 3     1 2016
## 4     1 1976
## 5     1 1988
## 6     1 1971
```

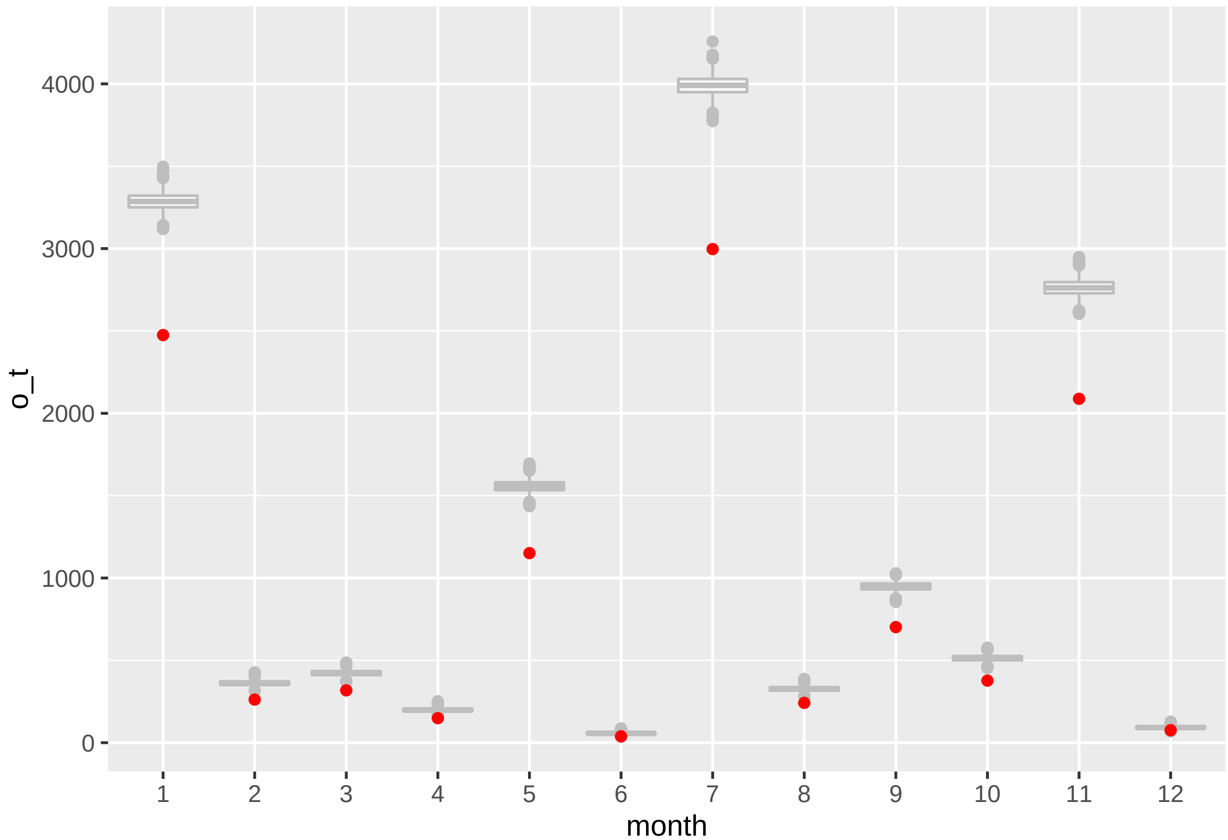
```r
# finish this function
visualization <- function(mylist= None, post= F, u_t = F, x_a = F, string='string') {
  data= mylist$data
  if (post == T) {
    # boxplots of o_t
    trace_o_t = mylist$trace
  p <- ggplot()+
  # boxplot from the trace
  geom_boxplot(aes(x=month,y=o_t), color='grey',data = trace_o_t)+
  # real values as red dots
  geom_point(aes(x=o_t_values$month, y=o_t_values$o_t),color='red')

  }
```

```r
  if (u_t == T) {
    ppc_u_t = mylist$ppc$u_t
    # boxplots of u_t
  p <- ggplot()+geom_boxplot(aes(x=month,y=u_t),data = ppc_u_t)+geom_point(aes(x=data$month, y=data$u_t)

  }

  if (x_a == T) {
    ppc_x_a = mylist$ppc$x_a
    p <- ggplot()+geom_boxplot(aes(x=month,y=x_a),data = ppc_x_a)+geom_point(aes(x=data$month, y=data$x_
  }
  return(p)
}
```

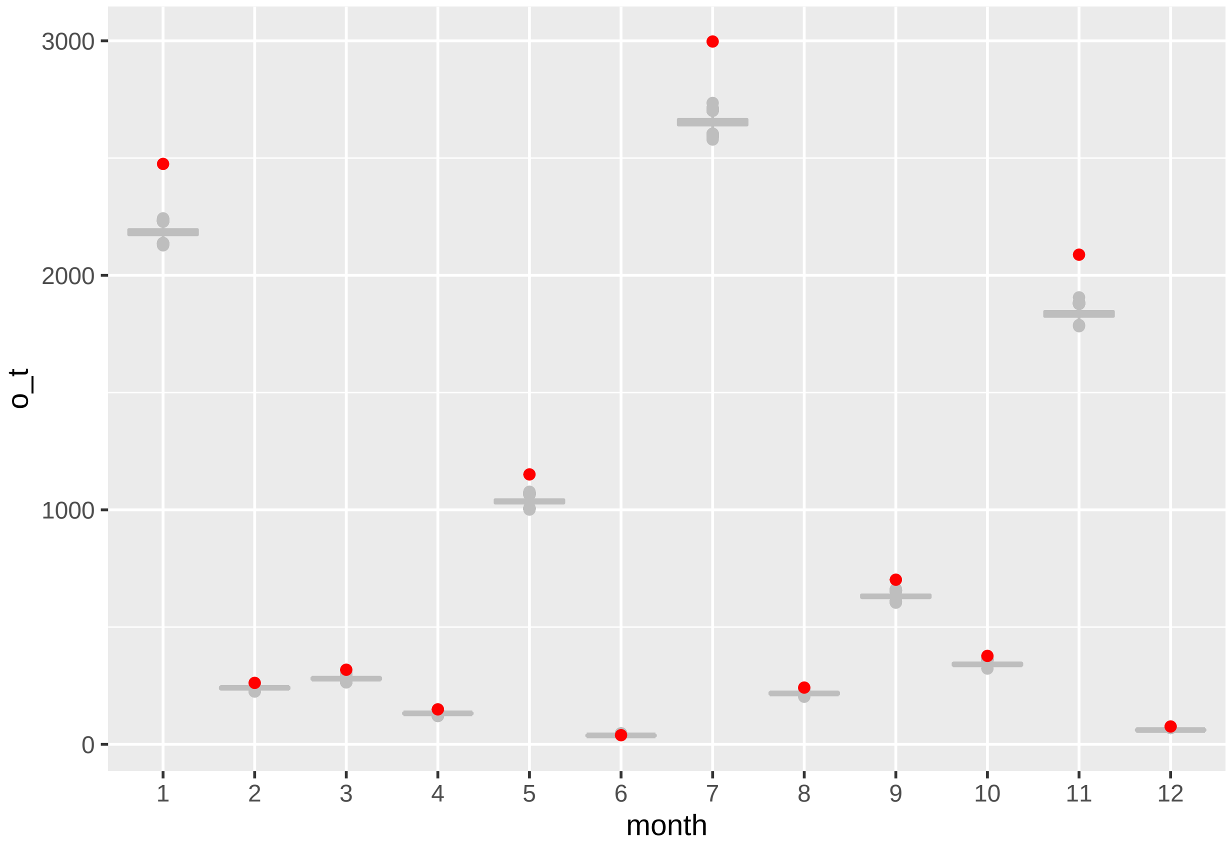
```r
visualization(my_list_unbiased, post=T)
```



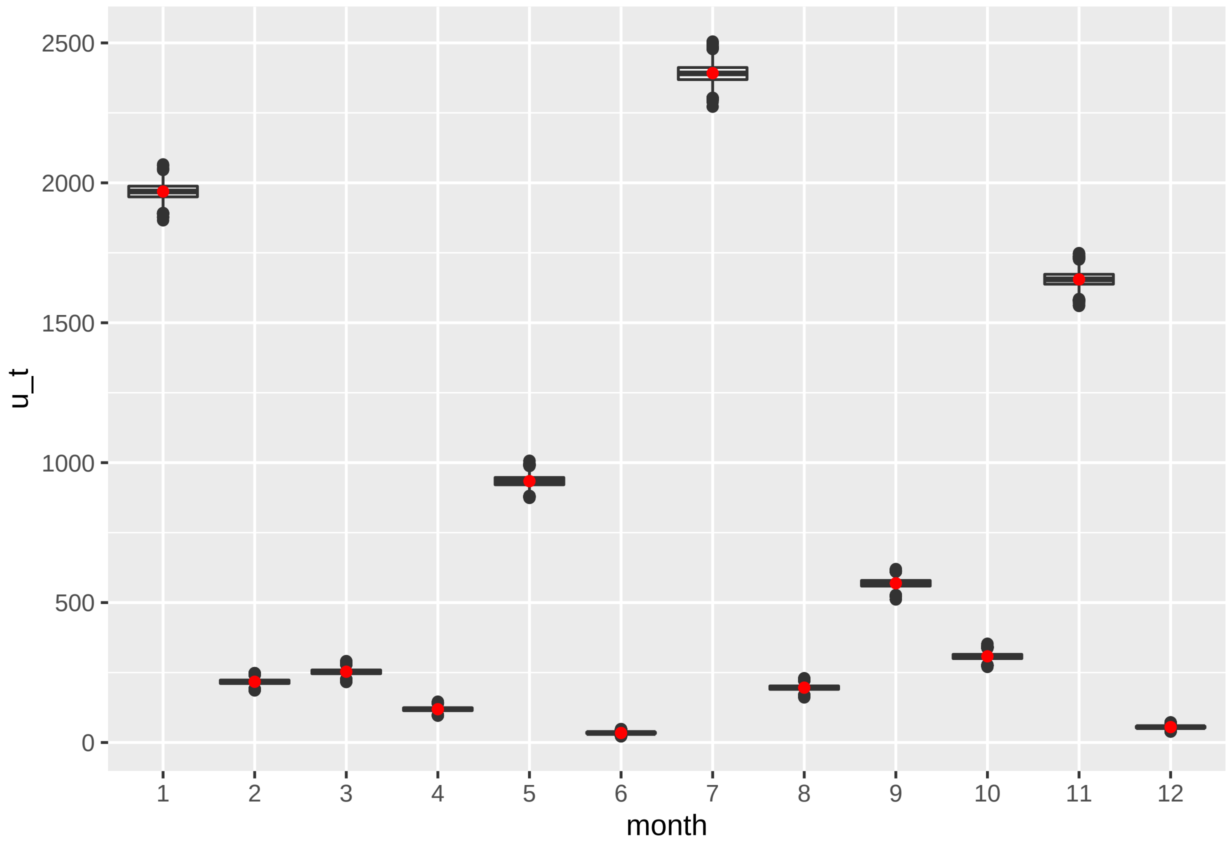

```r
visualization(my_list_under,post= T)
```



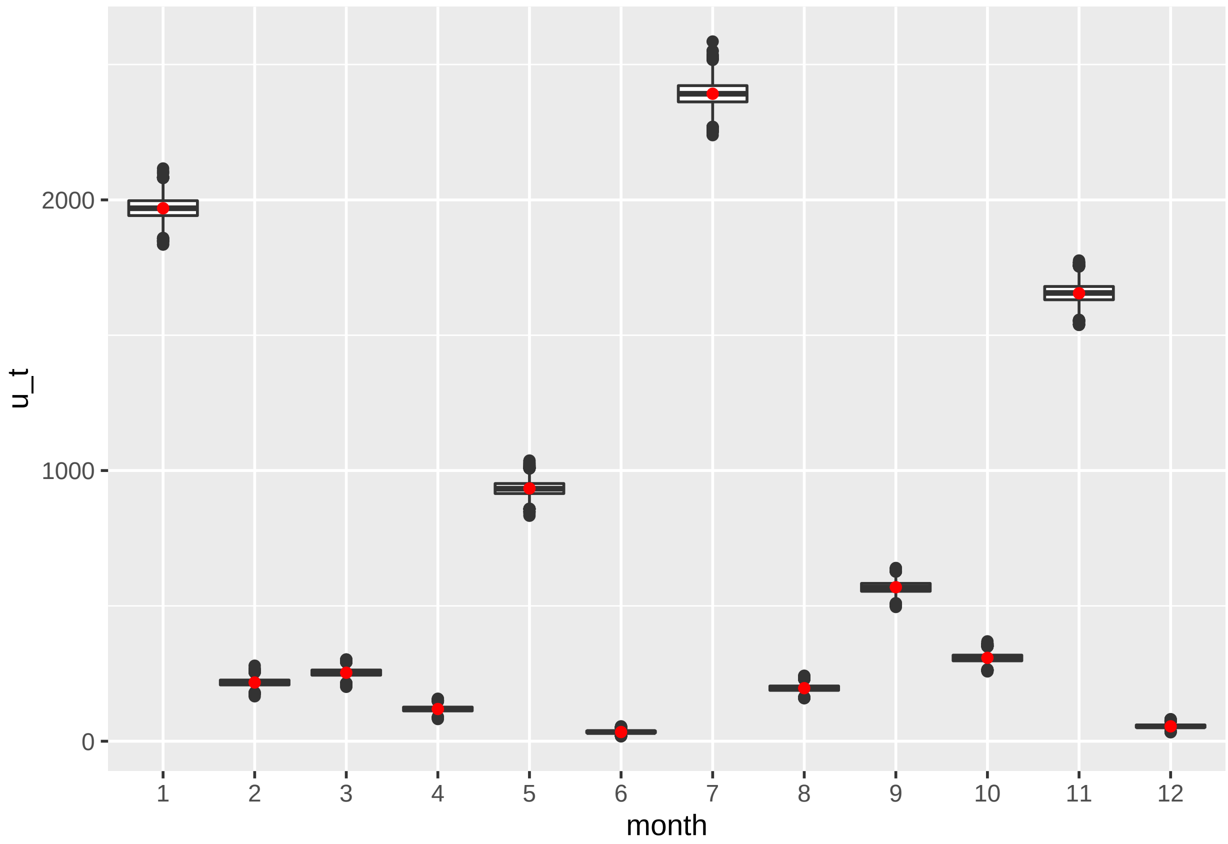

```
visualization(my_list_over,post=T)
```



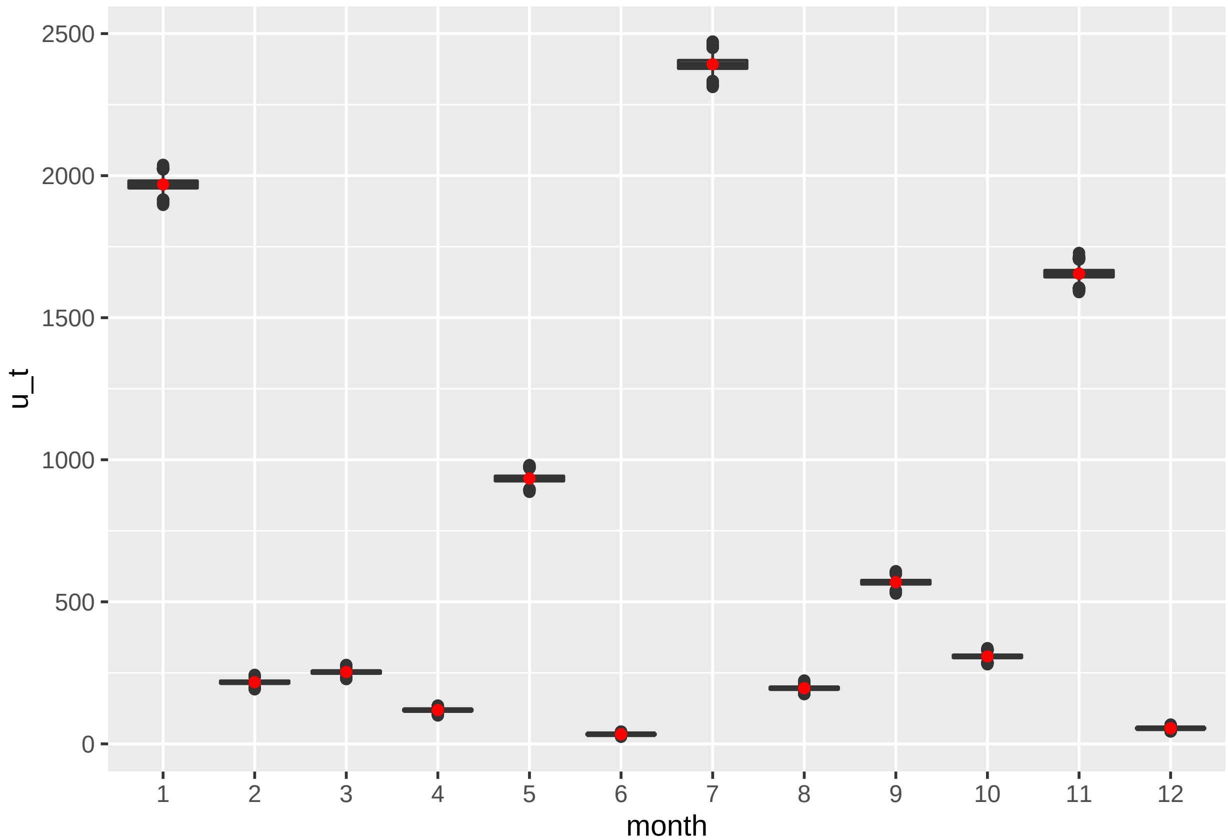

```
visualization(my_list_unbiased, u_t=T)
```



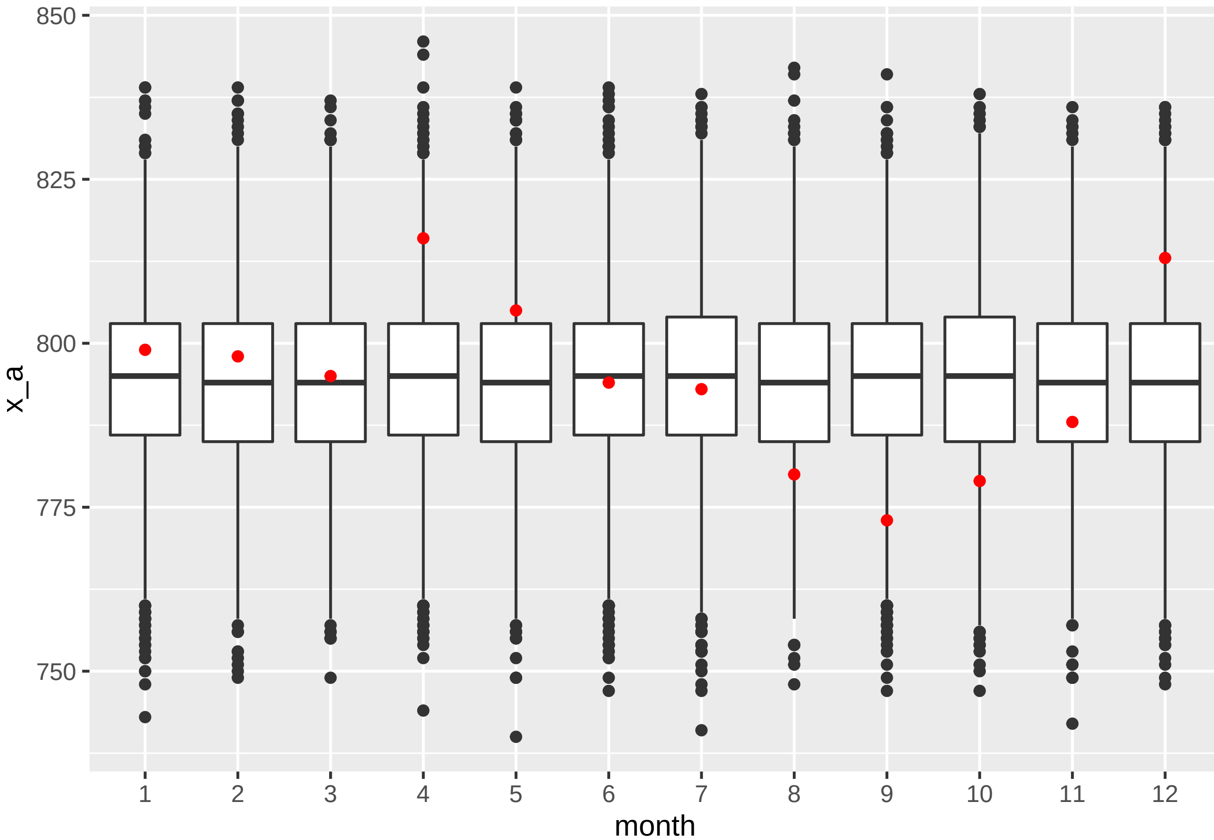

```
visualization(my_list_under,u_t= T)
```



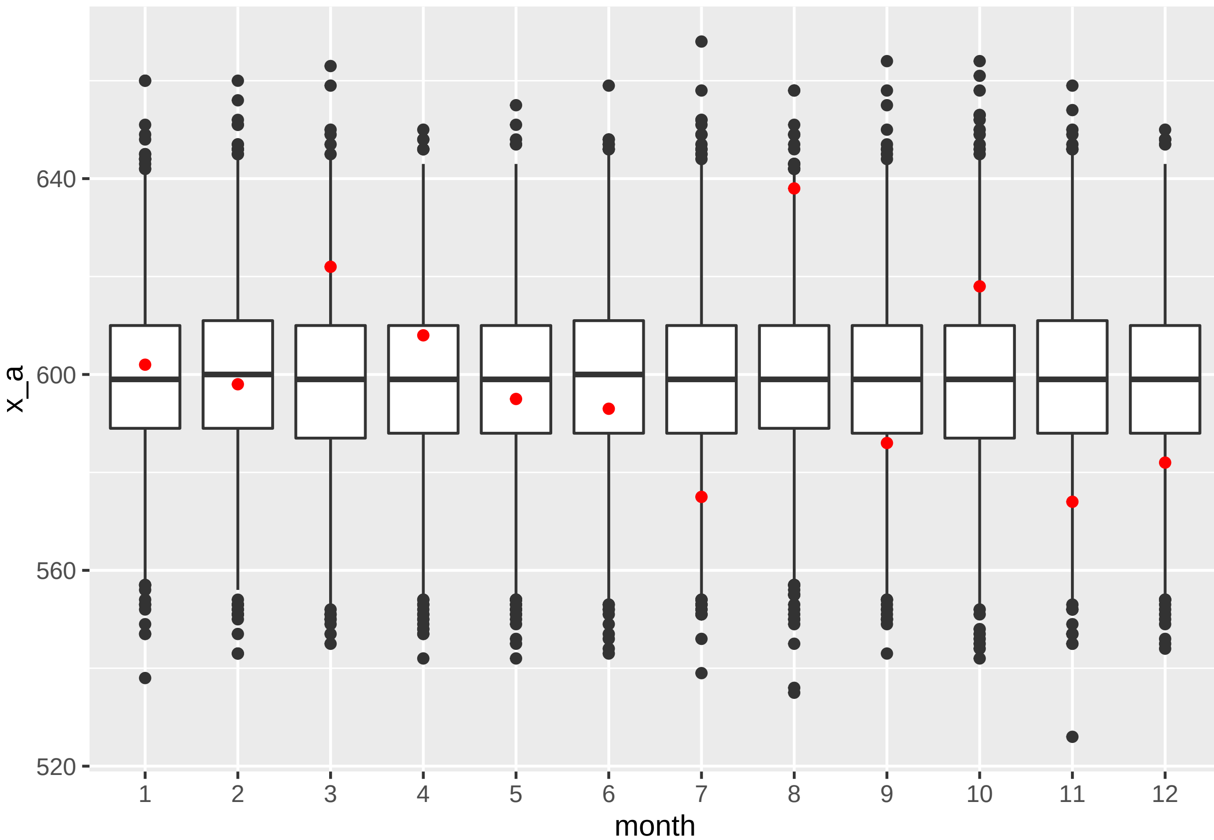

```
visualization(my_list_over,u_t=T)
```



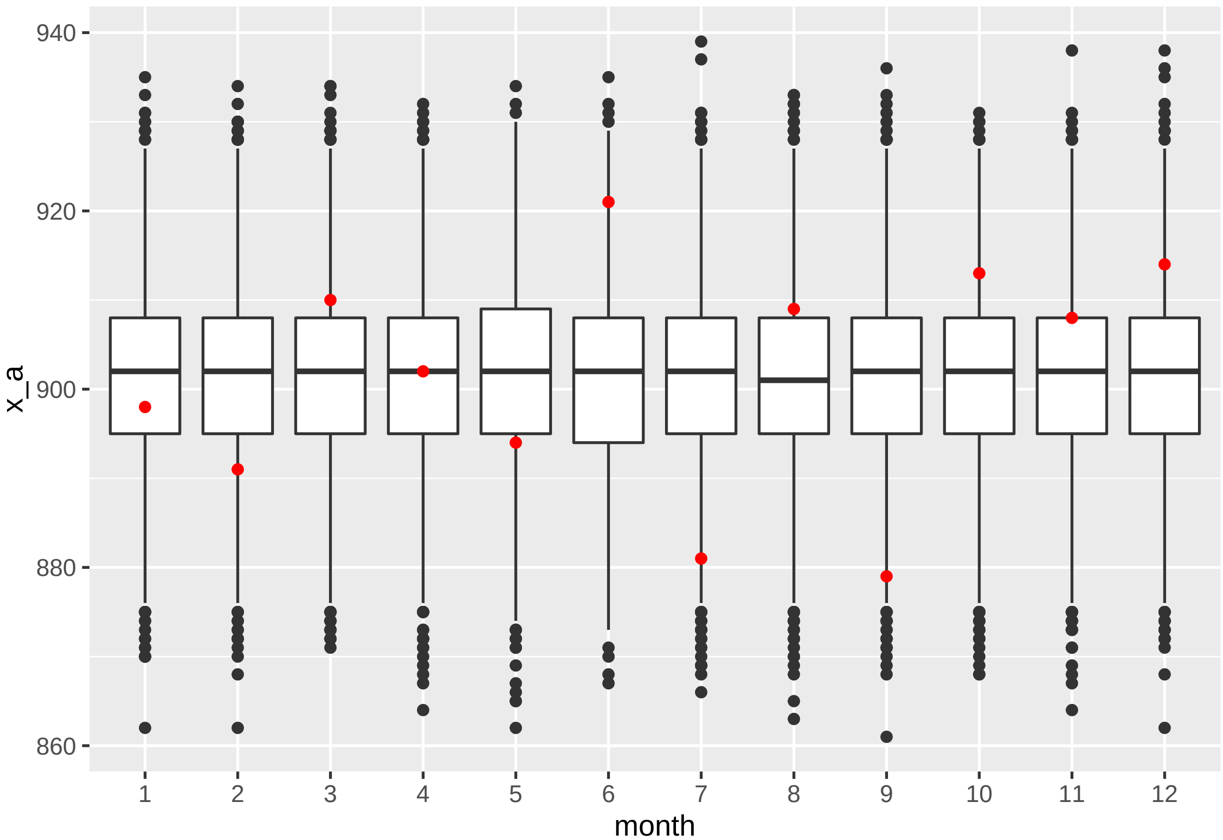

```
visualization(my_list_unbiased, x_a=T)
```




```
visualization(my_list_under,x_a= T)
```




```
visualization(my_list_over,x_a=T)
```



####### Below here is for the purpose of debugging

Trim here and make a function and give us the plots.

```
chains=2
under.model <- jags.model(file='basic_model.txt',
                          data=dat,
                          n.chains = chains)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 24
##    Unobserved stochastic nodes: 27
##    Total graph size: 88
##
## Initializing model
```

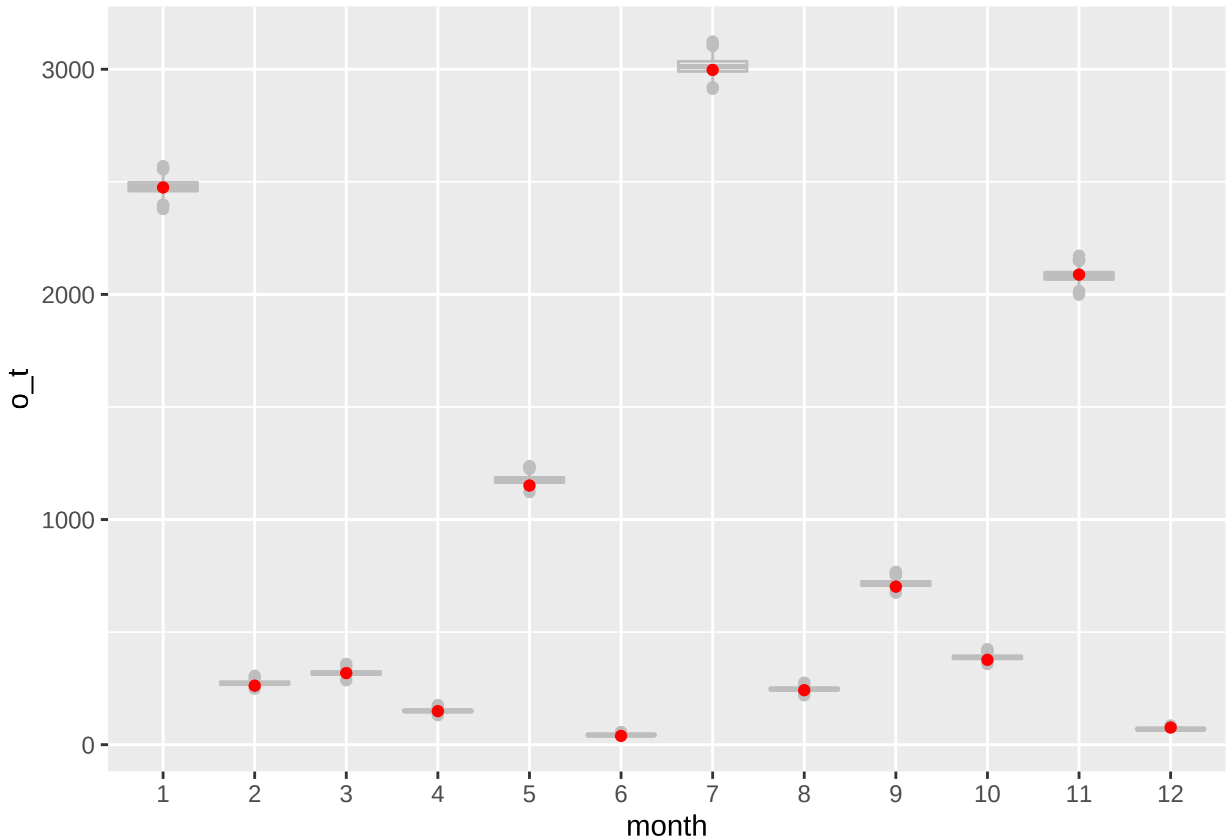
```
# inits = list()


# get the samples of O_t, p_a
params= c('o_t','p_a')
samples <- coda.samples(under.model, params, n.iter = 1000)
pst_mtx = as.matrix(samples)
head(pst_mtx)
```

```
##      o_t[1] o_t[2] o_t[3] o_t[4] o_t[5] o_t[6] o_t[7] o_t[8] o_t[9]
## [1,]   2494    285    313    156   1161     38   3077    242    731
## [2,]   2444    277    317    153   1169     40   3060    248    709
## [3,]   2495    275    321    150   1170     43   3058    241    709
## [4,]   2454    277    325    151   1207     45   3018    231    719
## [5,]   2514    284    321    154   1183     44   3025    232    727
## [6,]   2473    274    322    151   1171     44   3045    238    713
##      o_t[10] o_t[11] o_t[12]       p_a
## [1,]     386    2027      71 0.7977980
## [2,]     393    2056      67 0.7934034
## [3,]     379    2041      68 0.7932773
## [4,]     379    2079      72 0.7912505
## [5,]     378    2083      65 0.7937473
## [6,]     382    2075      70 0.7892883
```

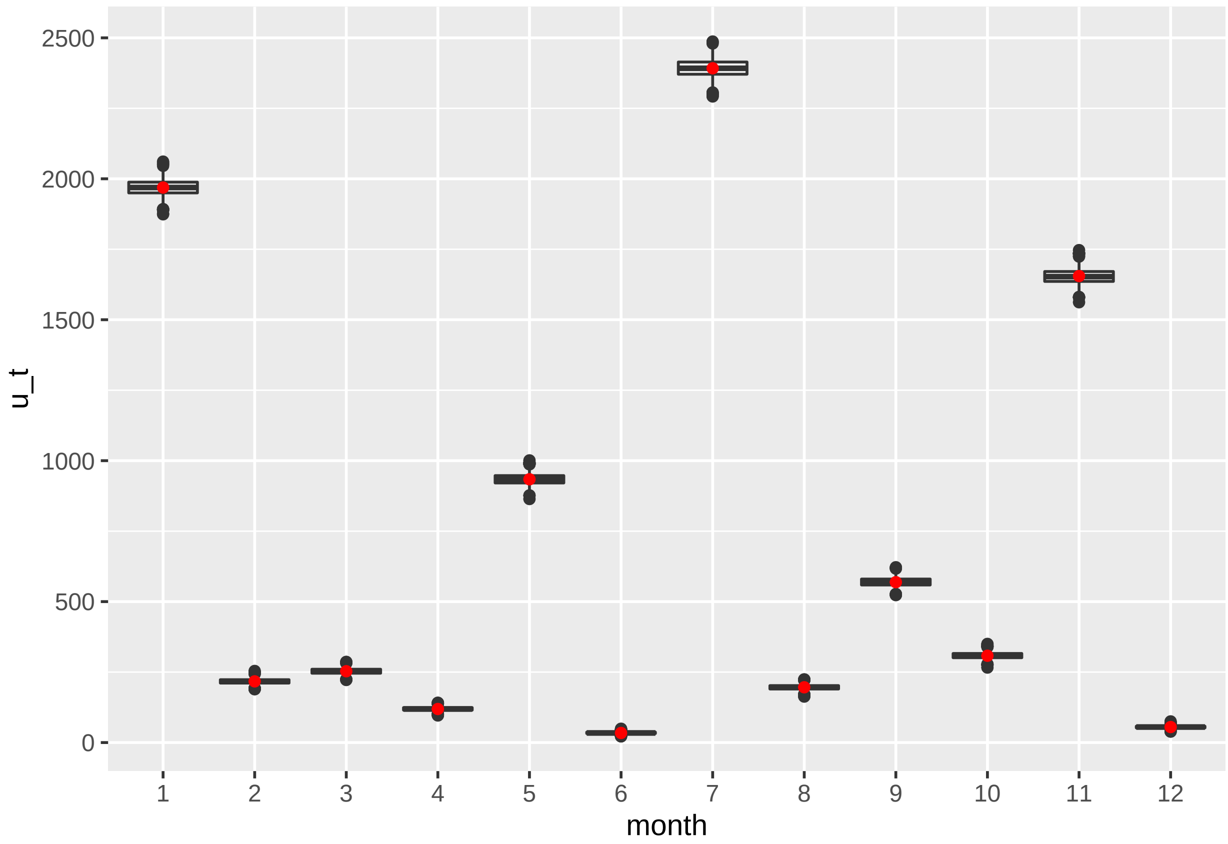
```r
# tidy p_a
p_a <- pst_mtx[,13]
head(p_a)
```

```
## [1] 0.7977980 0.7934034 0.7932773 0.7912505 0.7937473 0.7892883
```

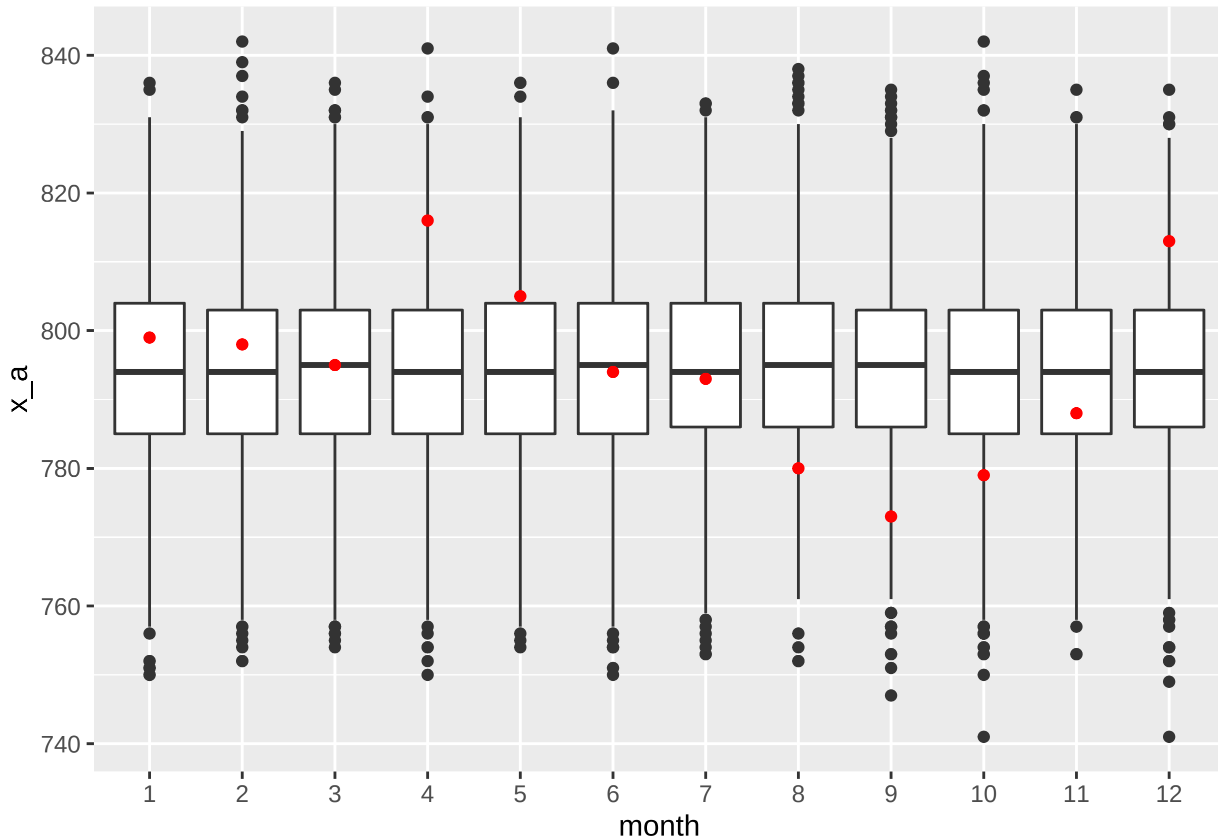
```r
## tidy o_t
pst_mtx = as.matrix(samples)
temp = pst_mtx[,1:12]
colnames(temp) <- seq(1,12)
df_o_t <- as.data.frame(temp)
trace_o_t <- gather(df_o_t, key = 'month',value = 'o_t')
trace_o_t$month <- factor(trace_o_t$month,levels = seq(1,12))
# real values of the data set from pymc3 samples
o_t_values=data.frame('month'=seq(1,12),'o_t'=df$o_t)
# factorizing the months for box plot visualization
o_t_values$month <- factor(o_t_values$month,levels = seq(1,12))
# boxplots of o_t
ggplot()+
  # boxplot from the trace
  geom_boxplot(aes(x=month,y=o_t), color='grey',data = trace_o_t)+
  # real values as red dots
  geom_point(aes(x=o_t_values$month, y=o_t_values$o_t),color='red')
```




```
# tidy u_t pp samples
df_u_t <- data.frame()
for (i in 1:m) {
  obs <- rbinom(n=12,size = as.numeric(df_o_t[i,]),prob =p_a[i])
  df_u_t=rbind(df_u_t,obs)
}
colnames(df_u_t) <- factor(seq(1,12),levels = seq(1,12))
ppc_u_t <- gather(df_u_t, key = 'month',value = 'u_t')
ppc_u_t$month <- factor(ppc_u_t$month,levels = seq(1,12))
u_t_values=data.frame('month'=seq(1,12),'u_t'=df$u_t)
# boxplots of u_t
ggplot()+geom_boxplot(aes(x=month,y=u_t),data = ppc_u_t)+geom_point(aes(x=u_t_values$month, y=u_t_values
```




```r
## tidy x_a
m=length(p_a)
df_x_a <- vector()
for (i in 1:n_T) {
  obs <- rbinom(m,n_a,p_a)
  df_x_a <- cbind(df_x_a,obs)
}

colnames(df_x_a) <- seq(1:12)
df_x_a <- as.data.frame(df_x_a)
ppc_x_a <- gather(df_x_a, key = 'month',value = 'x_a')
ppc_x_a$month <- factor(ppc_x_a$month,levels = seq(1,12))
x_a_values=data.frame('month'=seq(1,12),'x_a'=df$x_a)
x_a_values$month <- factor(x_a_values$month,levels = seq(1,12))


ggplot()+geom_boxplot(aes(x=month,y=x_a),data = ppc_x_a)+geom_point(aes(x=x_a_values$month, y=x_a_value
```



Reference

first tutorial
second tutorial
JAGS manual
error handling guide