STA 360/602: Assignment 3, Spring 2019

Kuei-Yueh Ko

Due at 10:00 AM on Monday, 28 January 2019

Contents


```
suppressWarnings(suppressMessages(library(tidyverse)))
library(RColorBrewer)
```

Lab component

Please refer to module 2 and lab 3 and complete tasks 3—5.

Suppose public health officials in a small city need to decide how much resources to devote toward prevention and treatment of a certain disease, but the fraction θ of infected individuals in the city is unknown.

Suppose they allocate enough resources to accomodate a fraction c of the population. If c is too large, there will be wasted resources, while if it is too small, preventable cases may occur and some individuals may go untreated. After deliberation, they tentatively adopt the following loss function:

$$\ell(\theta, c) = \begin{cases} |\theta - c|, & \text{if } c \geq \theta \\ 10 \times |\theta - c|, & \text{if } c < \theta \end{cases}$$

By considering data from other similar cities, they determine a prior $p(\theta)$. For simplicity, suppose $\theta \sim \text{Beta}(a, b)$ (i.e., $p(\theta) = \text{Beta}(\theta|a, b)$), with $a = 0.05$ and $b = 1$. They conduct a survey assessing the disease status of $n = 30$ individuals, $x_1, \ldots, x_n$. This is modeled as $X_1, \ldots, X_n \overset{iid}{\sim} \text{Bernoulli}(\theta)$, which is reasonable if the individuals are uniformly sampled and the population is large. Suppose all but one are disease-free, i.e., $\sum_{i=1}^{n} x_i = 1$.

Define all the functions

Consider the loss function.

$$\ell(\theta, c) = \begin{cases} |\theta - c|, & \text{if } c \geq \theta \\ 10 \times |\theta - c|, & \text{if } c < \theta \end{cases}$$

- input: cost, theta
- output: loss

```
# compute the loss given theta and c
LOSS_FUN = function(theta, cost){
    if (cost < theta){
        return(10 * abs(theta - cost))
    } else{
        return(abs(theta - cost))
    } # end ifelse
} # end func
```

Posterior risk
a function of cost (c), parameters a_prior and b_prior for the prior distribution of θ, the summation of x_i sum_x, the number of observations n, and also the number of random draws s.

$$\rho(cost, x) = E[\ell(\theta, c)|x]$$

- input: cost, data (sum_x, n), prior parameters
- output: posterior risk values

```
# compute the posterior risk given c (cost)
# s is the number of random draws
posterior_risk = function(cost, sum_x, n, shape_prior, s = 30000){
    ### get posterior
    a_prior = shape_prior[1]
    b_prior = shape_prior[2]
    a_post  = a_prior + sum_x
    b_post  = b_prior + n - sum_x

    ### randow draws from beta distribution
    theta = rbeta(s, a_post, b_post)
    loss  = apply(as.matrix(theta), 1, LOSS_FUN, cost)

    ### average values from the loss function
    risk = mean(loss)
} # end fun
```

cost_min_risk

- input: a vector of costs and a vector of corresponding risks (posterior risk)
- output: the cost corresponding to the minimum risk

```r
cost_min_risk = function(cost, risk){
    ### check dimension
    stopifnot(length(cost) == length(risk))

    ### return cost created minimum risk
    idx = which.min(risk)
    res = c(cost[idx], risk[idx])
    return(res)
} # end fun
```

Procedures

ex: Bayes procedure

$$\text{cost}^* = \underset{c}{\text{argmin}} \rho(c, x)$$

- input: data (sum_x, n), other parameters required for the procedure
- output: the optimal cost (decision)

```r
proc_bayes = function(sum_x, n, shape_prior, s = 100){
    ### initialize a vector of costs
    cost = seq(0, 1.0, by = 0.01)

    ### generate posterior risk
    post_risk = apply(as.matrix(cost), 1, posterior_risk, sum_x, n, shape_prior, s)

    ### get optimal cost
    tmp = cost_min_risk(cost, post_risk)
    cost_opt = tmp[1]
    return(cost_opt)
} # end fun

proc_mean = function(sum_x, n){
    return(sum_x / n)
} # end fun

proc_const = function(sum_x, n, const){
    return(const)
} # end fun
```

Frequentist risk

$$R(\theta, \delta(x)) = E[\ell(\theta, c)|\theta]$$

- input: theta, procedure_function - output: frequentist risk values

```r
frequentist_risk = function(theta, find_optimal_cost, n = 30, ...){
    ### generate different data x
    set.seed(123)
    sum_xs = rbinom(100, n, theta)

    ### for different data, calculate optimal cost
    cost_opt = apply(as.matrix(sum_xs), 1, find_optimal_cost, n, ...)

    ### calculate loss
```

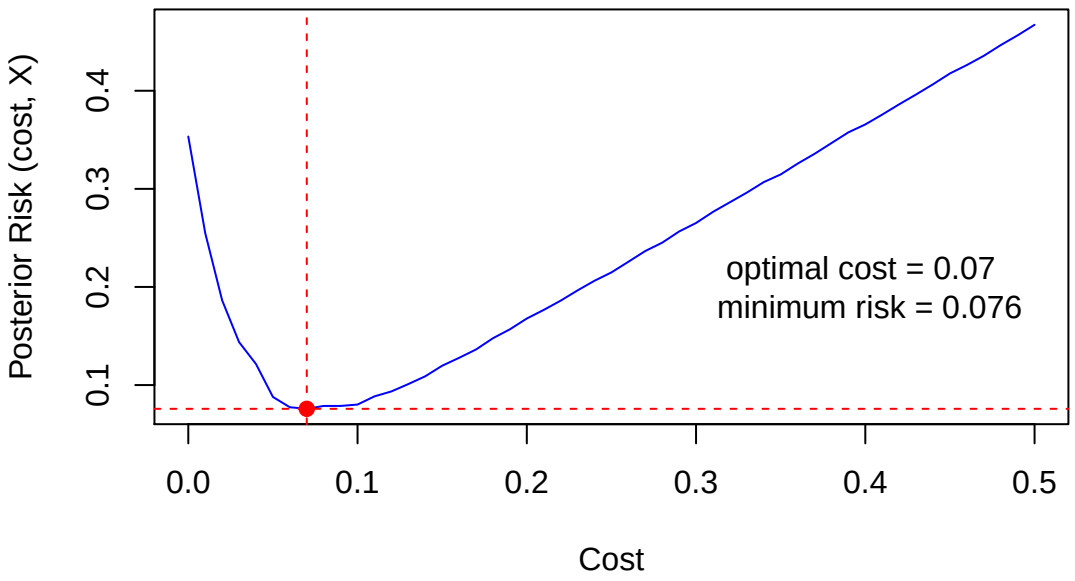
```
    loss = apply(as.matrix(cost_opt), 1, LOSS_FUN, theta = theta)
    freq_risk = mean(loss)
    return(freq_risk)
} # end fun
```

Task 1: Find minimum cost (c) given a, b, n, and $\sum x_i$

We know $p(\theta|x)$ as an updated Beta, so we can numerically compute this integral for each c. Reproduce Figure 1 from lecture, illustrating $\rho(c, x)$ for our example. Also, work through where the minimum occurs numerically ($c \approx 0.08$).

calculate posterior risk and reproduce figure 1 in lecture 2

```
### data
sum_x = 1
n = 30

### prior parameters
a = 0.05; b = 1
shape_prior = c(a, b)
### generate posterior risk
cost = seq(0, 0.5, by = 0.01)
s    = 2000
set.seed(123)
post_risk = apply(as.matrix(cost), 1, posterior_risk, sum_x, n, shape_prior, s)

### find the minimum cost
tmp = cost_min_risk(cost, post_risk)
cost_opt = tmp[1]; risk_min = tmp[2]

### reproduce figure 1
plot(  cost, post_risk,    col = "blue", type = "l", xlab = "Cost", ylab = "Posterior Risk (cost, X)")
points(cost_opt, risk_min, col = "red",  pch = 19)
abline(v = cost_opt, col = "red", lty = 2)
abline(h = risk_min, col = "red", lty = 2)
text(0.4, 0.2,  paste("optimal cost =", cost_opt, "\n",
                      "minimum risk =", round(risk_min, 3)))
```



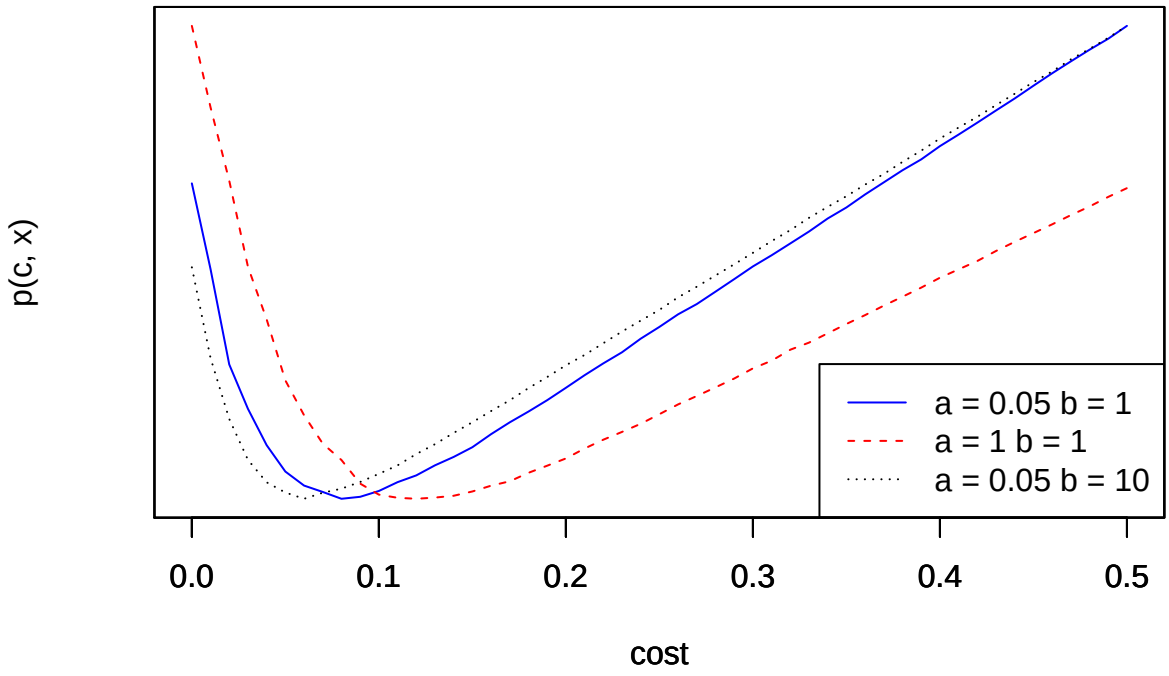
Task 2: Sensitivity Analysis

Now perform a sensitivity analysis for the prior assumption (Beta(a,b)). What do you find?

We now consider task 2. We set $a = 0.05, 1, 0.05$ and $b = 1, 2, 10$. If we have different prior, the posterior risk is minimized at different c values. The optimal c depends on not only the data, but also the prior setting.

```
### initialization
cost = seq(0, 0.5, by = 0.01)
s    = 5000

### set prior
as = c(0.05, 1, 0.05); bs = c(1, 1, 10)
shape_prior = cbind(as, bs)

### for each pair of a and b, compute the posterior risks
post_risk = apply(shape_prior, 1, function(shape){
    apply(as.matrix(cost), 1, posterior_risk, sum_x, n, shape, s)
}) # end apply
post_risk = t(post_risk)

### plot the results
plot(cost, post_risk[1,], type = 'l', col='blue', lty = 1, yaxt = "n", ylab = "p(c, x)")
par(new = T)
plot(cost, post_risk[2,], type = 'l', col='red', lty = 2, yaxt = "n", ylab = "")
par(new = T)
```

```r
plot(cost, post_risk[3,], type = 'l', lty = 3, yaxt = "n", ylab = "")

legend("bottomright",
       lty = c(1,2,3),
       col = c("blue", "red", "black"),
       legend = c("a = 0.05 b = 1", "a = 1 b = 1", "a = 0.05 b = 10"))
```



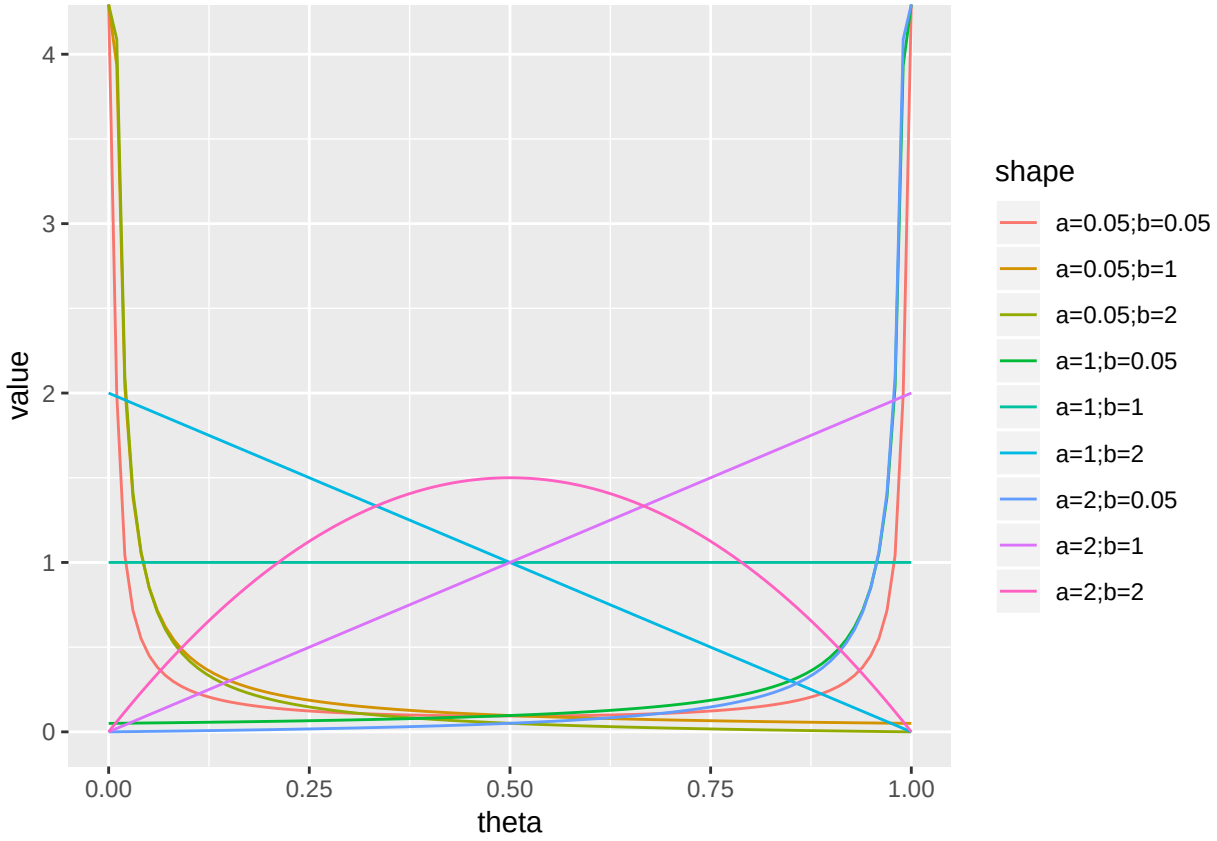
In order to observe how different a and b change the $\rho(cost, x)$, below I performed a more detailed sensitivity analysis.

First, observe how shape change the distribution of theta. Note that the textbook ch03 p39 has mentioned:

- a ≈ "prior number of 1's"
- b ≈ "prior number of 0's"

```r
### define combinations of shapes
as = c(0.05, 1, 2); bs = c(0.05, 1, 2)
shape_prior = expand.grid(as, bs)
cols = brewer.pal(nrow(shape_prior), "Set1")

### get the corresponded distribution
theta = seq(0, 1, length.out = 100)
plot_dbeta = apply(shape_prior, 1, function(shape){dbeta(theta, shape[1], shape[2])})
colnames(plot_dbeta) = apply(shape_prior, 1, function(shape){paste0("a=", shape[1], ";", "b=", shape[2]
plot_dbeta = cbind(theta, plot_dbeta) %>% as.data.frame
```

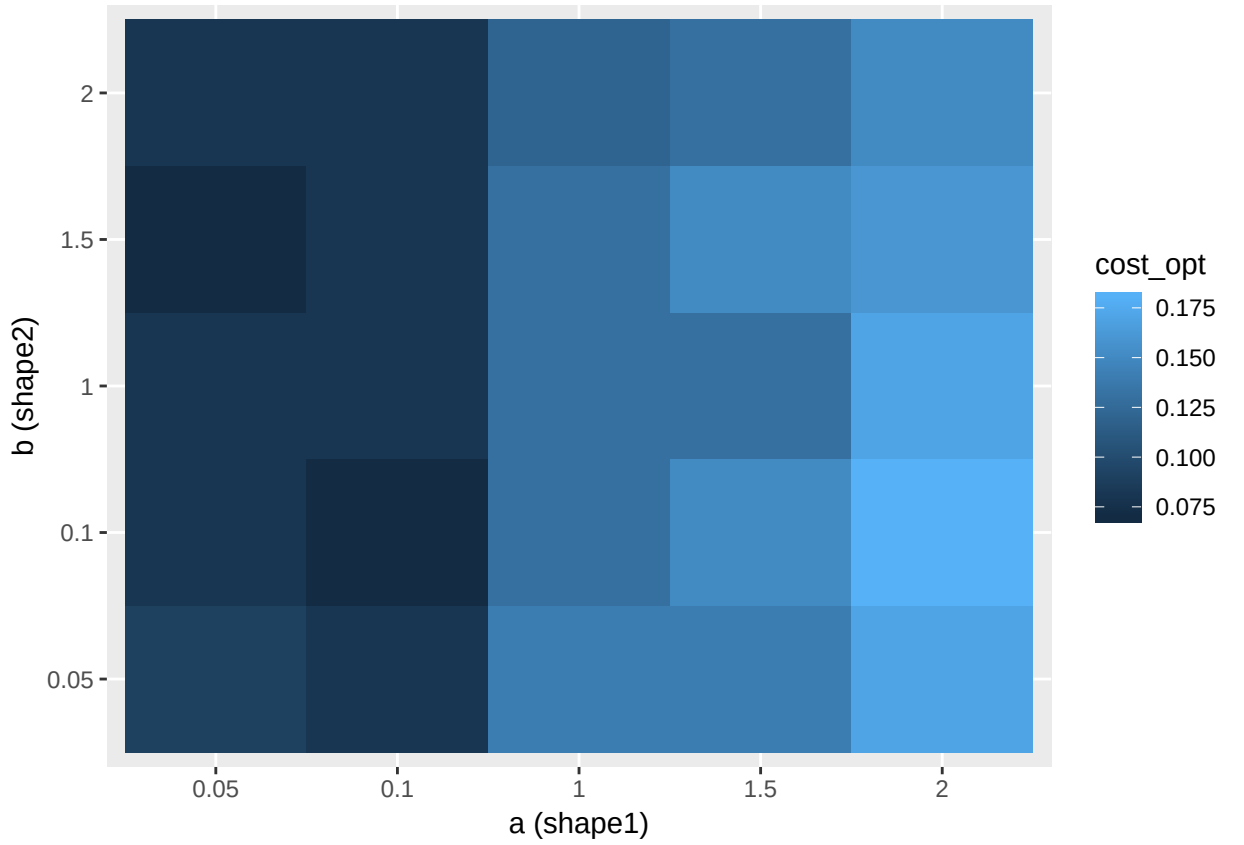
```
### visualization
plot_dbeta %>% gather(shape, value, -theta) %>% ggplot(., aes(x = theta, y = value, color = shape)) + g
```



Below is the optimal cost of Bayesian Procedure for different combination of priors a and b. From the heatmap, we could observe that the level of optimal cost from Bayesian procedure is more dependent on prior a. The highest optimal cost appears to be located at higher a and smaller b. That is, the decision (cost) tends to be higher if we belief there are more desease positive cases.

```
### define combinations of shapes
#as = c(0.05, 1, 2); bs = c(0.05, 1, 2)
as = c(0.05, 0.1, 1, 1.5, 2); bs = c(0.05, 0.1, 1, 1.5, 2)
shape_prior = expand.grid(as, bs)
colnames(shape_prior) = c("a", "b")

### data
sum_x = 1
n     = 30

### proc_bayes = function(sum_x, n, shape_prior, s = 100){
set.seed(123)
cost_opt = apply(shape_prior, 1, function(shape){proc_bayes(sum_x, n, shape, s = 1000)})
res = cbind(shape_prior, cost_opt)

### visualization
gp = ggplot(res, aes(x = as.character(a), y = as.character(b), fill = cost_opt)) +
    geom_tile() +
```

```
    labs(x = "a (shape1)", y = "b (shape2)")
print(gp)
```



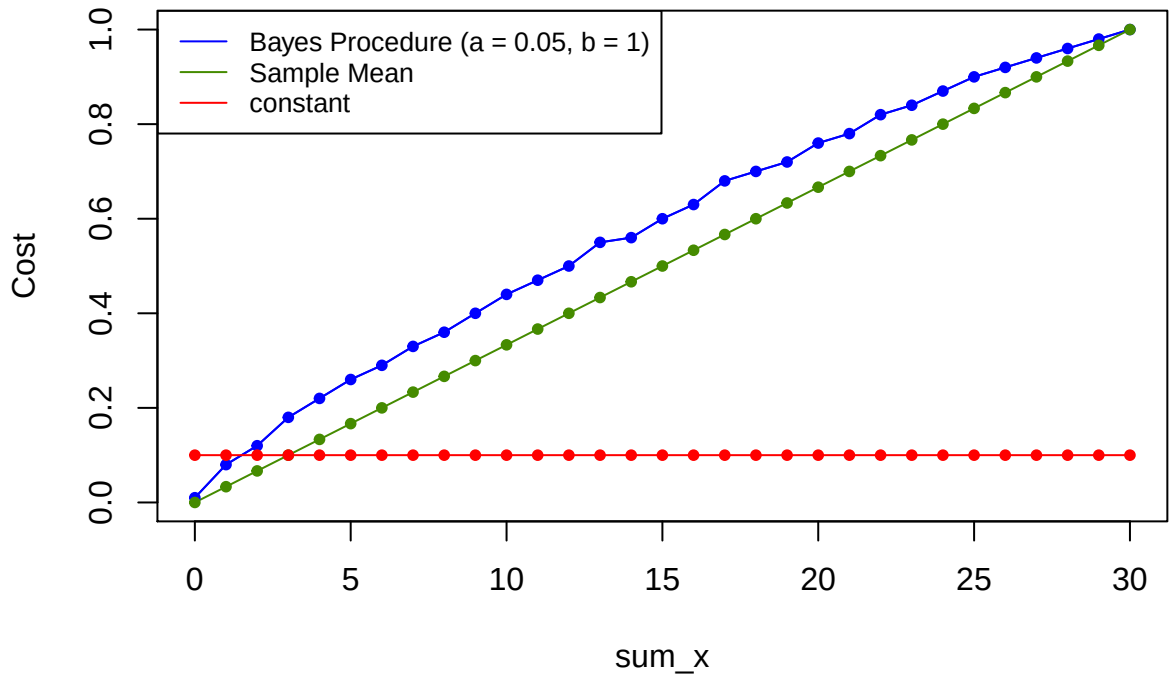
Task 3

Consider the Bayes procedure ($c \approx 0.08$), $c = \bar{x}, c = 0.1$. Reproduce Figure 2. Explain your findings.

```
### data
sum_x = seq(0, 30, by = 1)
n = 30

### prior parameters
a = 0.05; b = 1
shape_prior = c(a, b)

### generate posterior risk
cost = seq(0, 1.0, by = 0.01)
set.seed(123)
cost_opt = sapply(sum_x, function(x){proc_bayes(x, n, shape_prior, s = 2000)})
```

Reproduce figure 2 in the lecture 02. From the plot, we could observe the Bayes Procedure tends to be higher than other procedure such as mean and constant(=0.1)

```r
plot( sum_x, cost_opt, type = "l",     col = "blue", ylim = c(0, 1), ylab = "Cost")
lines(sum_x, cost_opt,                 col = "blue")
lines(sum_x, sum_x / n,                col = "chartreuse4")
lines(sum_x, rep(0.1, length(sum_x)), col = "red")

points(sum_x, cost_opt,                 pch = 20, col = "blue")
points(sum_x, sum_x / n,                pch = 20, col = "chartreuse4")
points(sum_x, rep(0.1, length(sum_x)), pch = 20, col = "red")

legend("topleft",
       lty = c(1,1,1),
       col = c("blue", "chartreuse4", "red"),
       cex = 0.8,
       legend = c("Bayes Procedure (a = 0.05, b = 1)", "Sample Mean", "constant"))
```

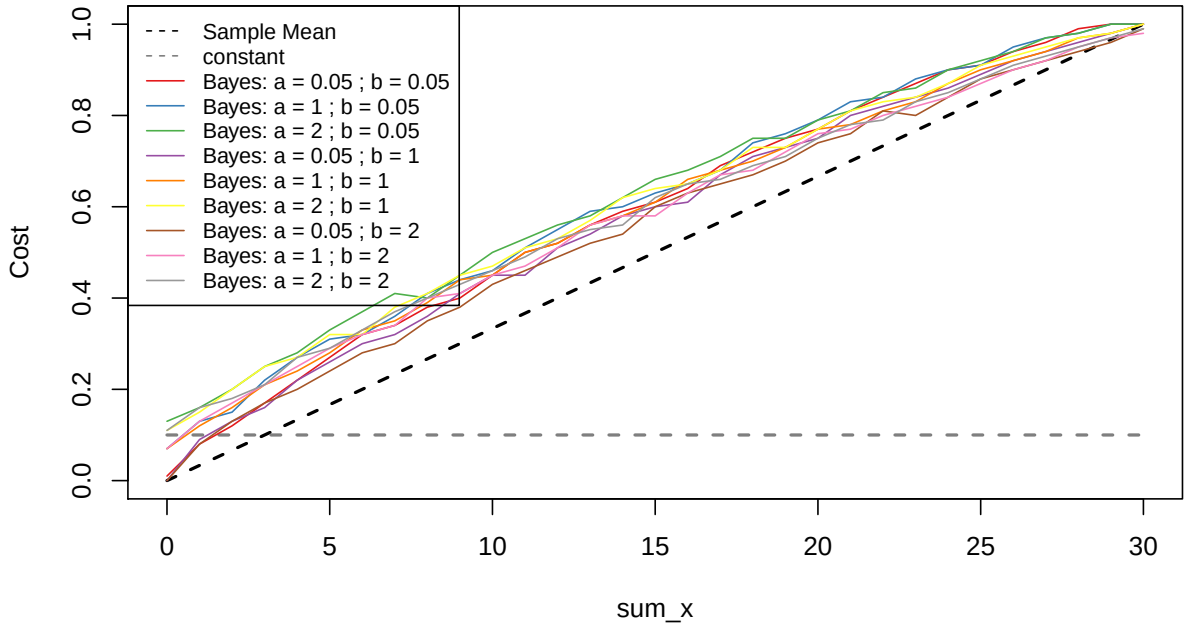



```r
### data
sum_x = seq(0, 30, by = 1)
n = 30

### prior parameters:
as = c(0.05, 1, 2); bs = c(0.05, 1, 2)
shape_prior = expand.grid(as, bs)

### generate cost vs sum_x for each shape
set.seed(123)
```

```
mat_cost_opt = apply(shape_prior, 1, function(shape){
    sapply(sum_x, function(x){
        proc_bayes(x, n, shape, s = 2000)
    }) # end inner apply
}) # end outer apply
mat_cost_opt = t(mat_cost_opt)
```

Visualize how different prior may affect the cost for different data (sum_x). From the results, we could observe that not matter what prior is, the cost decision from Bayes procedure would always be higher than other types of procedure.

```
### set colors
cols = brewer.pal(nrow(shape_prior), "Set1")

### plot mean and constant procedure
plot( sum_x, sum_x / n, lwd = 2, lty = 2, type = "l", col = "black", ylim = c(0, 1), ylab = "Cost")
lines(sum_x, rep(0.1, length(sum_x)), lwd = 2, lty = 2, col = "grey50")

### for each shape parameter, plot cost vs sum_x
for (idx in 1:9){
    lines(sum_x, mat_cost_opt[idx,], col = cols[idx])
} # end loop

### set legend
tmp = apply(shape_prior, 1, function(shape){paste("Bayes:", "a =", shape[1], ";", "b =", shape[2])})
legend("topleft",
       lty = c(2, 2, rep(1, length(cols))),
       col = c("black", "grey50", cols),
       cex = 0.8,
       legend = c("Sample Mean", "constant", tmp))
```



Conclusion of Task 3: The Bayes procedure always picks cost to be a little bigger than $\bar{x}$.

Task 4

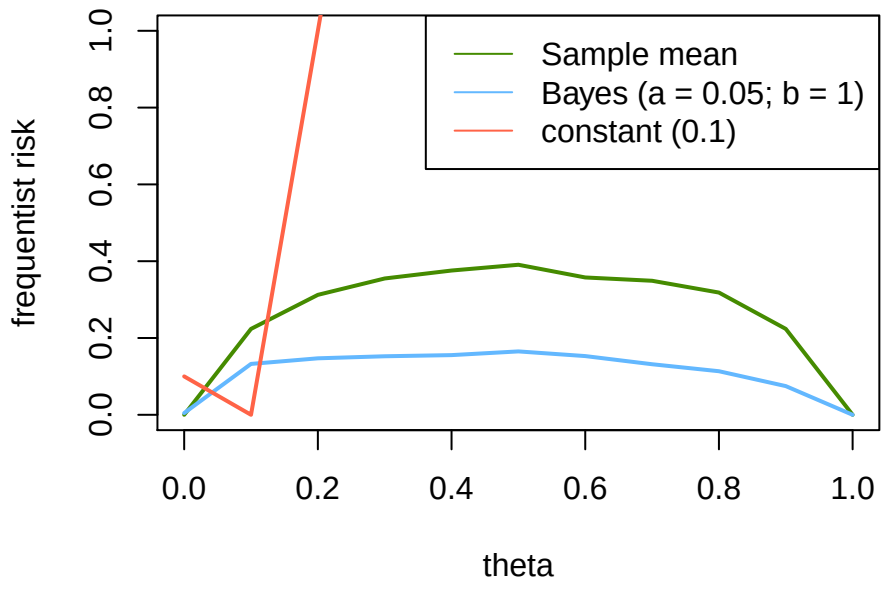
For most of θ, the Bayes procedure has the lower frequentist risk than the sample mean.

```
set.seed(123)

### frequentist_risk = function(theta, find_optimal_cost, n = 30, ...)
thetas = seq(0, 1, by=0.1)

### frequentist risk of Bayes procedure
a = 0.05; b = 1
risk_bayes01 = sapply(thetas, function(theta){
    frequentist_risk(theta, proc_bayes, n = 30, shape_prior = c(a, b), s = 500)
}) # end sapply

a = 1; b = 1
risk_bayes02 = sapply(thetas, function(theta){
    frequentist_risk(theta, proc_bayes, n = 30, shape_prior = c(a, b), s = 500)
}) # end sapply

a = 1; b = 0.05
risk_bayes03 = sapply(thetas, function(theta){
    frequentist_risk(theta, proc_bayes, n = 30, shape_prior = c(a, b), s = 500)
}) # end sapply
```

```r
### frequentist risk of mean procedure
risk_mean = sapply(thetas, function(theta){
    frequentist_risk(theta, proc_mean, n = 30)
}) # end sapply

### frequentist risk of constant procedure
risk_const01 = sapply(thetas, function(theta){
    frequentist_risk(theta, proc_const, n = 30, const = 0.1)
}) # end sapply

risk_const02 = sapply(thetas, function(theta){
    frequentist_risk(theta, proc_const, n = 30, const = 0.5)
}) # end sapply

risk_const03 = sapply(thetas, function(theta){
    frequentist_risk(theta, proc_const, n = 30, const = 0.9)
})
```

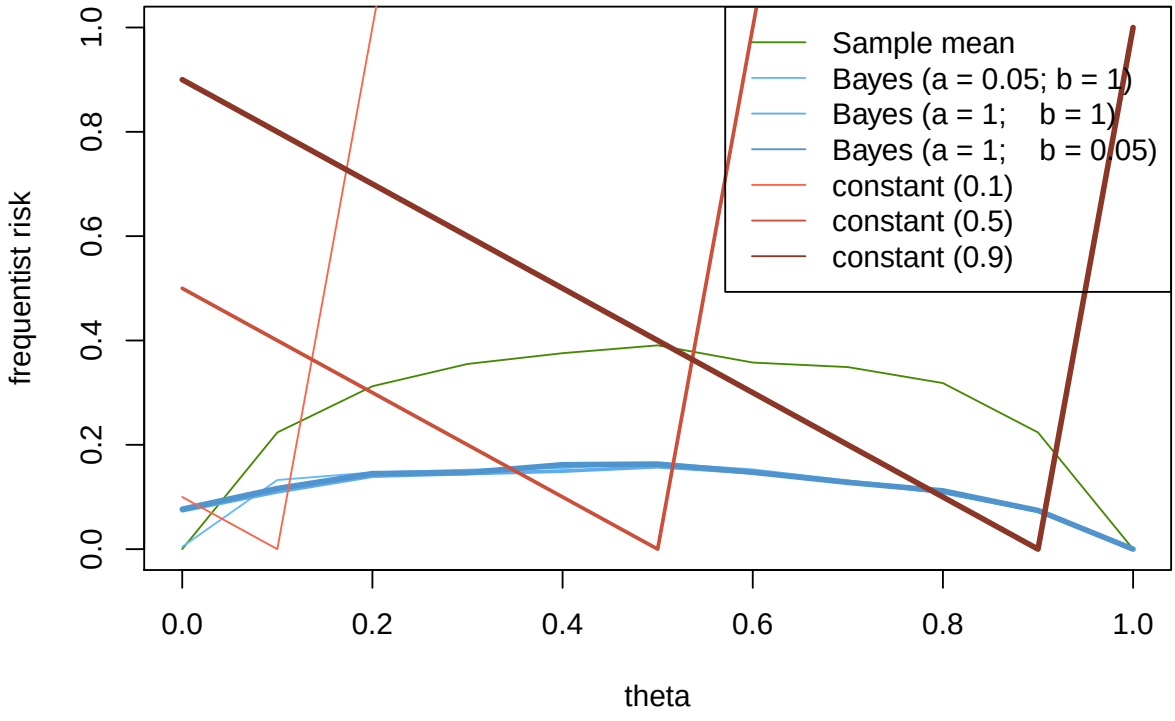
Recreate the plot in lecture. In this plot, the Bayes procedure with prior (0.05, 1.0) has lower risk than the sample mean procedure for all θ.

```r
plot( thetas, risk_mean, lwd = 2,    col ='chartreuse4', type = "l",
     ylab = "frequentist risk", xlab = 'theta', ylim = c(0,1))
lines(thetas, risk_bayes01, lwd = 2, col = 'steelblue1')
lines(thetas, risk_const01, lwd = 2, col = 'tomato1')
legend("topright", lty = c(1,1,1),
       col = c("chartreuse4", "steelblue1", "tomato1"),
       legend = c("Sample mean", "Bayes (a = 0.05; b = 1)", "constant (0.1)"))
```



For a more detail plot, the below plot included the procedure with different parameters. From the plot, we could conclude that for most of the θ, the Bayes procedure tends to have lower risk than other procedures, regardless of the prior.

```
plot( thetas, risk_mean,    lwd = 1,  col ='chartreuse4', type = "l",
     ylab = "frequentist risk", xlab = 'theta', ylim = c(0,1))
lines(thetas, risk_bayes01, lwd = 1,  col = 'steelblue1')
lines(thetas, risk_bayes02, lwd = 2,  col = 'steelblue2')
lines(thetas, risk_bayes03, lwd = 3,  col = 'steelblue3')
lines(thetas, risk_const01, lwd = 1,  col = 'tomato1')
lines(thetas, risk_const02, lwd = 2,  col = 'tomato3')
lines(thetas, risk_const03, lwd = 3,  col = 'tomato4')
legend("topright", lty = c(1,1,1),
       col = c("chartreuse4",
               "steelblue1", "steelblue2", "steelblue3",
               "tomato1",    "tomato3",    "tomato4"),
       legend = c("Sample mean",
                  "Bayes (a = 0.05; b = 1)",
                  "Bayes (a = 1;    b = 1)",
                  "Bayes (a = 1;    b = 0.05)",
                  "constant (0.1)",
                  "constant (0.5)",
                  "constant (0.9)"))
```



Task 5 Admissible & Inadmissible

Based on your plot of the frequentist risk, consider the three estimators—the constant, the mean, and the Bayes estimators. Which estimators are admissible? Be sure to explain why or why they are not admissible.

- The constant procedure is the best procedure for a small region of θ.
- The Bayes procedure works better generally. For example, for all θ, Bayes procedure works better than sample mean procedure. For many simtuations, Bayes procedure is the best among the three procedure.
- By definition, an inadmissible rule is a rule that dominates everywhere. For the three methods: Bayes, sample mean, and constant, none of the procedure is the best for all θ. Therefore, all three methods are admissible. Sample mean is not the best procedure for all θ. Based on the two previous points, the constant procedure and the Bayes procedure are the best for different regions of θ.