

Computing Assignment 1

Alyssa Vanderbeek

10/7/2019

All R code is shown at the end of the document.

Problem 1

The two-stage adaptive design as described in lecture is as follows: stage 1 enrolls 20 patients. At the interim, we stop the trial and conclude futility if less than 6 subjects respond. Otherwise, we continue onto stage 2 and enroll an additional 51 subjects for a total sample size of 71 subjects. At the end of the trial, we conclude futility (“no-go”) if fewer than 24 subjects respond, and conclude efficacy (“go”) if at least 24 subjects respond.

To mimic this design in the Bayesian formulation, we can set $(\delta, \alpha) = (0.15, 0.15)$, which gives a “go” decision for 24 responses, and a no-go decision for either 5/20 or 23/71 responses.

Problem 2

- (a) Using the selection of design parameters from problem 1, $(\delta, \alpha) = (0.15, 0.15)$, we stop the trial if $Pr(p_E > p_S + 0.15 | \text{data}) \leq 0.15$. The stopping boundaries at each interim analysis are then as follows:
- At 20 subjects, stop the trial if less than 6 subjects respond.
 - At 40 subjects, stop the trial if less than 13 subjects respond.
 - At 60 subjects, stop the trial if less than 20 subjects respond.
 - At the end of the trial (71 subjects), stop the trial if less than 24 subjects respond.
- (b) When the true response rate is 0.25, the stopping rules from part (a) give a ~0.96 probability of stopping for futility (which, in this case, would be the correct conclusion). The distribution of stopping times is given in Table 1.

Table 1: (2b) Probability of stopping for futility at each interim analysis, assuming the trial has passed previous stopping rules and $p=0.25$

Interim analysis	Probability
20	0.616
40	0.235
60	0.085
71	0.026

- (c) When the true response rate is 0.4, the stopping rules from part (a) give a ~0.24 probability of stopping for futility (which, in this case, would be the incorrect conclusion). The distribution of stopping times is given in Table 2.

Table 2: (2c) Probability of stopping for futility at each interim analysis, assuming the trial has passed previous stopping rules and $p=0.4$

Interim analysis	Probability
20	0.125
40	0.064
60	0.035
71	0.020

Problem 3

- (a) Using the posterior probability of declaring futility, where futility is declared when $Pr(S_{71} \geq 24 | S_m = s) < 0.9$, the stopping boundaries at each interim analysis are then as follows:
- At 20 subjects, stop the trial if less than 10 subjects respond.
 - At 40 subjects, stop the trial if less than 16 subjects respond.
 - At 60 subjects, stop the trial if less than 22 subjects respond.
 - At the end of the trial (71 subjects), stop the trial if less than 24 subjects respond.
- (b) When the true response rate is 0.25, the stopping rules from part (a) give a 0.99647 probability of stopping for futility (which, in this case, would be the correct conclusion). The distribution of stopping times is given in Table 3.

Table 3: (3b) Probability of stopping for futility at each interim analysis, assuming the trial has passed previous stopping rules and $p=0.25$

Interim analysis	Probability
20	0.986
40	0.008
60	0.003
71	0.000

- (c) When the true response rate is 0.4, the stopping rules from part (a) give a ~0.78 probability of stopping for futility (which, in this case, would be the incorrect conclusion). The distribution of stopping times is given in Table 4.

Table 4: (3c) Probability of stopping for futility at each interim analysis, assuming the trial has passed previous stopping rules and $p=0.4$

Interim analysis	Probability
20	0.754
40	0.021
60	0.005
71	0.000

R code

Problem 1

```
### Prior distribution of null and alternative response rates
n1 = 20
s1 = 6
ntotal = 71 # total sample size
s2 = 24 # futility interim
N = 10e4

a.s = 25 # prior a for control
b.s = 75 # prior b for control
ps = rbeta(N, a.s, b.s) # simulate control responses from prior

a = 0.5
b = 1.5
a_cond.total = a + s2 # prior a for exp conditional on s
b_cond.total = b + ntotal - s2 # prior b for exp conditional on s
pe.total = rbeta(N, a_cond.total, b_cond.total) # simulate exp responses from prior

#### Find delta, alpha to satisfy decision boundary
d = 0.15 # seq(0, 0.2, 0.01)[-1]
alpha = seq(0, 1, 0.05)[-1]

### 1. For given delta, find alpha s.t. s = 24 gives a "go" decision
t = sum(pe.total > (ps + d)) / N # left side of Bayesian decision rule
# take only those values of alpha for which s=22 lends a "go" decision
alpha = alpha[which((t > alpha) == TRUE)]; alpha

### 2. For alphas that satisfy 1, find the subset that satisfy "go" for s2=24 AND "no-go" for s2=23 AND
## go/no-go decision for s2=24
t.go = sum(pe.total > (ps + d)) / N

## go/no-go decision for s=23
s2.no = s2 - 1
a_cond = a + s2.no
b_cond = b + ntotal - s2.no
pe = rbeta(N, a_cond, b_cond)
t.nogo2 = sum(pe > (ps + d)) / N

## go/no-go decision for s1=5
s1.no = s1 - 1
a_cond = a + s1.no
b_cond = b + n1 - s1.no
pe = rbeta(N, a_cond, b_cond)
t.nogo1 = sum(pe > (ps + d)) / N

## Subset that satisfies all three conditions
index = Reduce(intersect, list(which((t.go > alpha) == TRUE),
                                which((t.nogo2 > alpha) == FALSE),
                                which((t.nogo1 > alpha) == FALSE)))
alpha.test = alpha[index] # new alphas
```

```
c(d, alpha.test)
```

Problem 2

```
### 2a
delta = d
alpha = alpha.test

interim.n = c(20, 40, 60, 71)
stop.rule = c()

## interim 1 at 20 patients
n = interim.n[1]
s = seq(0, n, 1)
dec = unlist(
  lapply(s, function(i){
    a_cond = a + i
    b_cond = b + n - i
    pe = rbeta(N, a_cond, b_cond)
    t = sum(pe > (ps + delta)) / N
    t > alpha
  })
)
stop.rule[1] = s[which(dec == TRUE)[1]]

## interim 2 at 40 patients
n = interim.n[2]
s = seq(0, n, 1)
dec = unlist(
  lapply(s, function(i){
    a_cond = a + i
    b_cond = b + n - i
    pe = rbeta(N, a_cond, b_cond)
    t = sum(pe > (ps + delta)) / N
    t > alpha
  })
)
stop.rule[2] = s[which(dec == TRUE)[1]]

## interim 3 at 60 patients
n = interim.n[3]
s = seq(0, n, 1)
dec = unlist(
  lapply(s, function(i){
    a_cond = a + i
    b_cond = b + n - i
    pe = rbeta(N, a_cond, b_cond)
    t = sum(pe > (ps + delta)) / N
    t > alpha
  })
)
stop.rule[3] = s[which(dec == TRUE)[1]]
```

```

## final look at 71 patients
n = interim.n[4]
s = seq(0, n, 1)
dec = unlist(
  lapply(s, function(i){
    a_cond = a + i
    b_cond = b + n - i
    pe = rbeta(N, a_cond, b_cond)
    t = sum(pe > (ps + delta)) / N
    t > alpha
  })
)
stop.rule[4] = s[which(dec == TRUE)[1]]

#stop.rule

### 2b
pe.true = 0.25

B = list(interim = c(),
         futility = c())

for (i in 1:N) {
  set.seed(i)
  subj = rbinom(ntotal, 1, pe.true) # simulate 71 Bernoulli RV's with prob pe.true

  # stopping rules
  if (sum(subj[1:interim.n[1]]) < stop.rule[1]) {
    B$interim[i] = interim.n[1]
    B$futility[i] = TRUE
  } else if (sum(subj[1:interim.n[2]]) < stop.rule[2]) {
    B$interim[i] = interim.n[2]
    B$futility[i] = TRUE
  } else if (sum(subj[1:interim.n[3]]) < stop.rule[3]) {
    B$interim[i] = interim.n[3]
    B$futility[i] = TRUE
  } else if (sum(subj) < stop.rule[4]) {
    B$interim[i] = interim.n[4]
    B$futility[i] = TRUE
  } else {
    B$interim[i] = NA
    B$futility[i] = FALSE
  }
}

# Table of stopping times
B %>%
  as.data.frame %>%
  group_by(interim) %>%
  filter(!is.na(interim)) %>%
  summarise(n = n(),
            prop = round(n/N, 3)) %>%
  select(-n) %>%

```

```

knitr::kable(col.names = c("Interim analysis", "Probability"),
             caption = "Probability of stopping for futility at each interim analysis, assuming the t

sum(B$futility) / N

### 2c

pe.true = 0.4

B = list(interim = c(),
         futility = c())

for (i in 1:N) {
  set.seed(i)
  subj = rbinom(ntotal, 1, pe.true) # simulate 71 Bernoulli RV's with prob pe.true

  # stopping rules
  if (sum(subj[1:interim.n[1]]) < stop.rule[1]) {
    B$interim[i] = interim.n[1]
    B$futility[i] = TRUE
  } else if (sum(subj[1:interim.n[2]]) < stop.rule[2]) {
    B$interim[i] = interim.n[2]
    B$futility[i] = TRUE
  } else if (sum(subj[1:interim.n[3]]) < stop.rule[3]) {
    B$interim[i] = interim.n[3]
    B$futility[i] = TRUE
  } else if (sum(subj) < stop.rule[4]) {
    B$interim[i] = interim.n[4]
    B$futility[i] = TRUE
  } else {
    B$interim[i] = NA
    B$futility[i] = FALSE
  }
}

# Table of stopping times
B %>%
  as.data.frame %>%
  group_by(interim) %>%
  filter(!is.na(interim)) %>%
  summarise(n = n(),
            prop = round(n/N, 3)) %>%
  select(-n) %>%
  knitr::kable(col.names = c("Interim analysis", "Probability"),
              caption = "Probability of stopping for futility at each interim analysis, assuming the t

sum(B$futility) / N

```

Problem 3

```

delta = 0.9
#alpha = alpha.test

```

```

interim.n = c(20, 40, 60, 71)
stop.rule = c()

## interim 1 at 20 patients
n = interim.n[1]
s = seq(0, n, 1)
dec = unlist(
  lapply(s, function(i){
    a_cond = a + i
    b_cond = b + n - i
    pe = rbeta(N, a_cond, b_cond)
    t = mean(rbinom(N, 71 - n, pe) >= (24 - i))
    t < delta
  })
)
stop.rule[1] = s[which(dec == FALSE)[1]]

## interim 2 at 40 patients
n = interim.n[2]
s = seq(0, n, 1)
dec = unlist(
  lapply(s, function(i){
    a_cond = a + i
    b_cond = b + n - i
    pe = rbeta(N, a_cond, b_cond)
    t = mean(rbinom(N, 71 - n, pe) >= (24 - i))
    t < delta
  })
)
stop.rule[2] = s[which(dec == FALSE)[1]]

## interim 3 at 60 patients
n = interim.n[3]
s = seq(0, n, 1)
dec = unlist(
  lapply(s, function(i){
    a_cond = a + i
    b_cond = b + n - i
    pe = rbeta(N, a_cond, b_cond)
    t = mean(rbinom(N, 71 - n, pe) >= (24 - i))
    t < delta
  })
)
stop.rule[3] = s[which(dec == FALSE)[1]]

## Make sure that boundary is 24 patients for final look at 71 patients
n = interim.n[4]
s = seq(0, n, 1)
dec = unlist(
  lapply(s, function(i){
    a_cond = a + i
    b_cond = b + n - i
    pe = rbeta(N, a_cond, b_cond)

```

```

    t = mean(rbinom(N, 71 - n, pe) >= (24 - i))
    t < delta
  })
)
stop.rule[4] = s[which(dec == FALSE)[1]]

pe.true = 0.25

B = list(interim = c(),
         futility = c())

for (i in 1:N) {
  set.seed(i)
  subj = rbinom(ntotal, 1, pe.true) # simulate 71 Bernoulli RV's with prob pe.true

  # stopping rules
  if (sum(subj[1:interim.n[1]]) < stop.rule[1]) {
    B$interim[i] = interim.n[1]
    B$futility[i] = TRUE
  } else if (sum(subj[1:interim.n[2]]) < stop.rule[2]) {
    B$interim[i] = interim.n[2]
    B$futility[i] = TRUE
  } else if (sum(subj[1:interim.n[3]]) < stop.rule[3]) {
    B$interim[i] = interim.n[3]
    B$futility[i] = TRUE
  } else if (sum(subj) < stop.rule[4]) {
    B$interim[i] = interim.n[4]
    B$futility[i] = TRUE
  } else {
    B$interim[i] = NA
    B$futility[i] = FALSE
  }
}

# Table of stopping times
B %>%
  as.data.frame %>%
  group_by(interim) %>%
  filter(!is.na(interim)) %>%
  summarise(n = n(),
            prop = round(n/N, 3)) %>%
  select(-n) %>%
  knitr::kable(col.names = c("Interim analysis", "Probability"),
              caption = "(3b) Probability of stopping for futility at each interim analysis, assuming

#sum(B$futility) / N

pe.true = 0.4

B = list(interim = c(),
         futility = c())

for (i in 1:N) {
  set.seed(i)

```



```

subj = rbinom(ntotal, 1, pe.true) # simulate 71 Bernoulli RV's with prob pe.true

# stopping rules
if (sum(subj[1:interim.n[1]]) < stop.rule[1]) {
  B$interim[i] = interim.n[1]
  B$futility[i] = TRUE
} else if (sum(subj[1:interim.n[2]]) < stop.rule[2]) {
  B$interim[i] = interim.n[2]
  B$futility[i] = TRUE
} else if (sum(subj[1:interim.n[3]]) < stop.rule[3]) {
  B$interim[i] = interim.n[3]
  B$futility[i] = TRUE
} else if (sum(subj) < stop.rule[4]) {
  B$interim[i] = interim.n[4]
  B$futility[i] = TRUE
} else {
  B$interim[i] = NA
  B$futility[i] = FALSE
}
}

# Table of stopping times
B %>%
  as.data.frame %>%
  group_by(interim) %>%
  filter(!is.na(interim)) %>%
  summarise(n = n(),
            prop = round(n/N, 3)) %>%
  select(-n) %>%
  knitr::kable(col.names = c("Interim analysis", "Probability"),
              caption = "(3c) Probability of stopping for futility at each interim analysis, assuming ")

#sum(B$futility) / N

```