Response-Adaptive Randomization in Clinical Trials

Chuyao Xu, Thomas Lumley, Alain Vandal

RARtrials is designed for simulating some popular response-adaptive randomization methods in the literature with comparisons of each treatment group to a control group under no delay and delayed (time between treatment and outcome availability) scenarios. All the designs are based on one-sided tests with a choice from values of 'upper' and 'lower'. The general assumption is that binary outcomes follow Binomial distributions, while continuous outcomes follow normal distributions. Additionally, the number of patients accrued in the population follows a Poisson process and users can specify the enrollment rate of patients enrolled in the trial. The methods included in this R package are as follows:

- The Randomized Play-the-Winner rule for binary outcomes in two-armed trials (Wei and Durham 1978);
- Doubly adaptive biased coin design targeting Neyman allocation and RSIHR allocation using minimal variance strategy for binary outcomes in trials with up to five arms (Biswas and Mandal 2004; Atkinson and Biswas 2013);
- Doubly adaptive biased coin design targeting Neyman allocation and RSIHR allocation using maximal power strategy for binary outcomes in trials with up to five arms and up to three arms respectively (Tymofyeyev, Rosenberger, and Hu 2007; Jeon and Feifang 2010; Bello and Sabo 2016);
- Neyman allocation (A_a-optimal allocation and A-optimal allocation) and generalised RSIHR allocation subject to constraints for continuous outcomes with known and unknown variances in trials with up to five arms (Sverdlov and Rosenberger 2013);
- Bayesian response-adaptive randomization using the Thall & Wathen method for binary outcomes and continuous outcomes with known and unknown variances in trials with up to five arms (Thall and Wathen 2007);
- Forward-looking Gittins Index and controlled forward-looking Gittins Index for binary outcomes and continuous outcomes with known and unknown variances in trials with up to five arms (Villar, Wason, and Bowden 2015; Williamson and Villar 2019).

The Randomized Play-the-Winner Rule for Binary Outcomes

The famous Randomized Play-the-Winner rule, proposed by Wei and Durham (1978), has been well studied over the past decades. Some properties can be summarized as follows:

1.$n_A/n \to q_B/(q_A + q_B)$ almost surely, with n_A representing the random variable for the number of participants on treatment A, n representing the total sample size, q_A and q_B representing the probabilities of failure under treatments A and B, respectively;

2. The maximum likelihood estimators following the Randomized Play-the-Winner rule satisfy

$$\sqrt{n}\left(\begin{bmatrix}\hat{p}_A\\ \hat{p}_B\end{bmatrix} - \begin{bmatrix}p_A\\ p_B\end{bmatrix}\right) \xrightarrow{d} N\left(0, \begin{bmatrix}\frac{p_A q_A (q_A+q_B)}{q_B} & 0\\ 0 & \frac{p_B q_B (q_A+q_B)}{q_A}\end{bmatrix}\right)$$

(convergence in distribution), with p_A and p_B the probabilities of success under treatments A and B, respectively;

3. Inference following the Randomized Play-the-Winner rule can be done using standard asymptotic tests. For a difference of two proportions, we can use a Z^2 test statistic, given by

$$Z^2 = \frac{(\hat{p}_A - \hat{p}_B)^2}{\frac{\hat{p}_A\hat{q}_A}{n_A} + \frac{\hat{p}_B\hat{q}_B}{n_B}} \Bigg| n_A, n_B \xrightarrow{d} \chi^2_1, \quad \text{with } n_A + n_B = n,\ n \to \infty.$$

When the sample size is large enough,the chi-squared distribution will approximate the distribution of the test statistic, allowing the selection of a cut-off value to control the type I error and calculate the power.

```
library(RARtrials)
```

```
## Loading required package: pins
```

```
#Example: RPTW(1,1) with the first 1 represents the initial number of balls for
#each treatment group in the urn and the second 1 represents the number of balls
#added to the urn when result of each participant becomes available.
#The function call below selects the cut-off value to control the type I error.
set.seed(12345)
sim1a<-lapply(1:5000, function(x){
  sim_RPTW(Pats=10,nMax=50000,TimeToOutcome=0,enrollrate=1,na0=1,nb0=1,na1=1,nb1=1,
      h=c(0.5,0.5),N2=192,side='upper')})
sum(sapply(sim1a, "[[", 2)>1.988,na.rm=T)/5000 #0.025
#Select a cut-off value to attain type I error 0.025 using the sample size 192

sim1b<-lapply(1:5000, function(x){
  sim_RPTW(Pats=10,nMax=50000,TimeToOutcome=0,enrollrate=1,na0=1,nb0=1,na1=1,nb1=1,
      h=c(0.5,0.7),N2=192,side='upper',Z=1.988)})
sum(sapply(sim1b, "[[", 1)==1)/5000
#Using the selected cut-off value 1.988, we obtain power 0.7938, which is close to 0.8

#Example: RPTW(1,1) with the first 1 represents the initial number of balls for
#each treatment group in the urn and the second 1 represents the number of balls
#added to the urn when result of each participant becomes available.
#Directly using asymptotic chi-square test which is equivalent to Z test statistics
set.seed(12345)
sim1<-lapply(1:5000, function(x){
  sim_RPTW(Pats=10,nMax=50000,TimeToOutcome=0,enrollrate=1,na0=1,nb0=1,na1=1,nb1=1,
      h=c(0.5,0.7),N2=192,side='upper')})
sum(sapply(sim1, "[[", 1)==1)/5000
#Using standard Z test statistics from normal distribution, we obtain power of 0.8038
```

Doubly Adaptive Biased Coin Design for Binary Outcomes

The doubly adaptive biased coin design is a parametric response-adaptive randomization method from a frequentist perspective (Rosenberger and Lachin 2015), which can attain any desired allocation ratio R^*. Hu and Zhang (2004) proposed the formula below with a tuning parameter $\gamma > 0$ (with a larger γ reducing the adaptiveness) to calculate allocation probabilities in two-armed trials, in which $\rho(\hat{p}_{A,t-1}, \hat{p}_{B,t-1})$ is the probabilty of allocation to arm A at time t given the estimated probabilities of success in each arm at time $t-1$:

$$f\left(0, \rho(\hat{p}_{A,t-1}, \hat{p}_{B,t-1})\right) = 1,$$

$$f\left(1, \rho(\hat{p}_{A,t-1}, \hat{p}_{B,t-1})\right) = 0,$$

$$f\left(\frac{n_{A,(t-1)}}{n_{(t-1)}}, \rho(\hat{p}_{A,t-1}, \hat{p}_{B,t-1})\right)$$

$$= \frac{\rho(\hat{p}_{A,t-1}, \hat{p}_{B,t-1})\left(\frac{\rho(\hat{p}_{A,t-1}, \hat{p}_{B,t-1})}{\frac{n_{A,(t-1)}}{n_{(t-1)}}}\right)^{\gamma}}{\rho(\hat{p}_{A,t-1}, \hat{p}_{B,t-1})\left(\frac{\rho(\hat{p}_{A,t-1}, \hat{p}_{B,t-1})}{\frac{n_{A,(t-1)}}{n_{(t-1)}}}\right)^{\gamma} + (1 - \rho(\hat{p}_{A,t-1}, \hat{p}_{B,t-1}))\left(\frac{1-\rho(\hat{p}_{A,t-1}, \hat{p}_{B,t-1})}{1-\frac{n_{A,(t-1)}}{n_{(t-1)}}}\right)^{\gamma}},$$

with the generalization to K-armed trials given below

$$\frac{\hat{\rho}^{*}_{k,t-1}\left(\frac{\hat{\rho}^{*}_{k,t-1}}{\frac{n_{k,t-1}}{t-1}}\right)^{\gamma}}{\sum_{i=1}^{K}\hat{\rho}^{*}_{i,t-1}\left(\frac{\hat{\rho}^{*}_{i,t-1}}{\frac{n_{i,t-1}}{t-1}}\right)^{\gamma}}.$$

In Hu & Zhang's formula, $\hat{\rho}^{*}_{k,t-1}$ is the optimal allocation ratio of arm k up to time t, which can apply the generalized Neyman allocation to minimize the total sample size given as $\sqrt{p_i q_i}/\sum_{i=1}^{K}\sqrt{p_i q_i}$ (Biswas and Mandal 2004; Atkinson and Biswas 2013) or the generalized RSIHR[1] allocation to minimize the expected number of failures given as $\sqrt{p_k}/\sum_{i=1}^{K}\sqrt{p_i}$ (Rosenberger and Lachin 2015). Those formulas can be generalized to K-armed trials from two-armed trials by employing a minimization variance strategy, which aims to minimize the sum of the variances of the test statistics for all comparisons to the control group. This package contains functions that return the allocation probabilities given data accumulated at each specified time point, as well as functions to simulate an actual trial data set.

```
#Example: Doubly adaptive biased coin design with five arms targeting
#RSIHR allocation using minimal variance strategy with return of allocation
#probabilities before applying Hu \& Zhang's formula.
dabcd_min_var(NN=c(20,23,18,25,27),Ntotal1=c(54,65,72,60,80),armn=5,type='RSIHR',
              dabcd=FALSE,gamma=2)
```

```
## [1] 0.3014955 0.1942672 0.0960814 0.2887962 0.1193597
```

```
#The function call return values:0.3014955 0.1942672 0.0960814 0.2887962 0.1193597

#Doubly adaptive biased coin design with five arms targeting RSIHR
#allocation using minimal variance strategy with return of allocation
#probabilities after applying Hu \& Zhang's formula.
dabcd_min_var(NN=c(20,23,18,25,27),Ntotal1=c(54,65,72,60,80),armn=5,type='RSIHR',
              dabcd=TRUE,gamma=2)
```

```
## [1] 0.2076180 0.2029166 0.1717949 0.2195535 0.1981169
```

```
#The function call return values:0.2076180 0.2029166 0.1717949 0.2195535 0.1981169
```

The other methods targeting generalized Neyman and RSIHR allocations are given in Tymofyeyev, Rosenberger, and Hu (2007) and Jeon and Feifang (2010), with modifications in Bello and Sabo (2016), employing a maximal power strategy, which maximizes the marginal power of a single comparison selected from amongst all comparisons to the control group. All those methods are implemented without smoothing, contrary to the original papers referenced in this package, because the smoothing technique does not function as indicated.

[1]Each letter of RSIHR represents the first character of the names of the individuals who first proposed this rule.

```
#Example: Doubly adaptive biased coin design with three arms targeting
#RSIHR allocation using maximal power strategy with return of allocation
#probabilities before applying Hu \& Zhang's formula.
dabcd_max_power(NN=c(20,60,60),Ntotal1=c(86,90,90),armn=3,BB=0.1, type='Neyman',dabcd=FALSE)
```

```
## [1] 0.4741802 0.2629099 0.2629099
```

```
#The function call return values:0.4741802 0.2629099 0.2629099
dabcd_max_power(NN=c(20,33,34),Ntotal1=c(86,78,90),armn=3,BB=0.1, type='Neyman',dabcd=FALSE)
```

```
## [1] 0.4433424 0.4566576 0.1000000
```

```
#The function call return values:0.4433424 0.4566576 0.1000000

#Doubly adaptive biased coin design with three arms targeting RSIHR
#allocation using maximal power strategy with return of allocation
#probabilities after applying Hu \& Zhang's formula.
dabcd_max_power(NN=c(20,60,60),Ntotal1=c(86,90,90),armn=3,BB=0.1, type='Neyman',dabcd=TRUE)
```

```
## [1] 0.7626214 0.1186893 0.1186893
```

```
#The function call return values:0.7626214 0.1186893 0.1186893
dabcd_max_power(NN=c(20,33,34),Ntotal1=c(86,78,90),armn=3,BB=0.1, type='Neyman',dabcd=TRUE)
```

```
## [1] 0.427536837 0.567983270 0.004479893
```

```
#The function call return values:0.427536837 0.567983270 0.004479893
```

A-optimal Allocation, A_a-optimal Allocation and Generalized RSIHR Allocation for Continuous Outcomes

A-optimal allocation (Sverdlov and Rosenberger 2013) is Neyman allocation for continuous outcomes, which minimizes the overall variance, and the formula for the probability of allocation to arm k is

$$\rho_k = \frac{\sigma_k}{\sum_{i=1}^{K} \sigma_i}, \text{ with } k = 1, .., K.$$

A modification to A-optimal allocation with higher allocation probabilities to the control group is known as A_a-optimal allocation (Sverdlov and Rosenberger 2013), with formula

$$\rho_1 = \frac{\sigma_1\sqrt{K-1}}{\sigma_1\sqrt{K-1} + \sum_{i=2}^{K} \sigma_k}, \quad \rho_k = \frac{\sigma_k}{\sigma_1\sqrt{K-1} + \sum_{i=2}^{K} \sigma_k}, \text{ with } k = 2, .., K.$$

The generalized RSIHR allocation for continuous outcomes minimizes

$$\sum_{i=1}^{K} n_i \Psi_i$$

with constraints

$$\frac{\sigma_1^2}{n_1} + \frac{\sigma_k^2}{n_k} \leq C, \text{ where } k = 2, ..., K.$$

In this formula, Ψ_i indicates a measure of treatment effectiveness of the i-th treatment, K stands for the total number of arms and C is some fixed value. The optimal allocation probabilities are given by

$$\rho_1 = \frac{\sqrt{\frac{w_1}{\Psi_1}}\sqrt{\sum_{i=2}^{k} \Psi_i w_i}}{\sqrt{\frac{w_1}{\Psi_1}}\sqrt{\sum_{i=2}^{k} \Psi_i w_i} + w_2 + ... + w_k},$$

$$\rho_k = \frac{w_k}{\sqrt{\frac{w_1}{\Psi_1}}\sqrt{\sum_{i=2}^{k} \Psi_i w_i} + w_2 + ... + w_k}, \text{ with } k = 2, ..., K.$$

This package is built on $\Psi_i = P(X_i \leq c) = \Phi[(\mu_A - c)/\sigma_A]$ and $w_k = \sigma_k$ (Atkinson and Biswas 2013). All of those methods can be applied to scenarios with either known and unknown variances. In cases where variances are unknown, the unknown parameters are estimated from the available data.

```
#Example: Generalized RSIHR optimal allocation with unknown variances in three-armed trials
set.seed(12345)
#Under the null hypothesis
sim2a<-lapply(1:5000,function(x){sim_RSIHR_optimal_known_var(Pats=10,nMax=50000,
TimeToOutcome=expression(rnorm( length( vStartTime ),30,3)), enrollrate=0.1,N1=12,N2=132,
armn=3,mean=c(9.1/100,9.1/100,9.1/100),sd=c(0.009,0.009,0.009),alphaa=0.025,
armlabel = c(1,2,3),cc=mean(c(9.1/100,9.1/100,9.1/100)),side='lower')})
h0decision<-t(sapply(sim2a, "[[", 1))
sum(h0decision[,1]==1|h0decision[,2]==1)/5000
#Attain lower one-sided type I error of 0.0218 with 5000 simulations

#Under the alternative hypothesis
sim2b<-lapply(1:5000,function(x){sim_RSIHR_optimal_known_var(Pats=10,nMax=50000,
TimeToOutcome=expression(rnorm( length( vStartTime ),30,3)), enrollrate=0.1,N1=12,N2=132,
armn=3,mean=c(9.1/100,8.47/100,8.47/100),sd=c(0.009,0.009,0.009),alphaa=0.025,
armlabel = c(1,2,3),cc=mean(c(9.1/100,8.47/100,8.47/100)),side='lower')})
h1decision<-t(sapply(sim2b, "[[", 1))
sum(h1decision[,1]==1)/5000
sum(h1decision[,2]==1)/5000
sum(h1decision[,1]==1|h1decision[,2]==1)/5000
#Marginal power of rejecting H02 is 0.8472
#Marginal power of rejecting H03 is 0.8432
#Overall power of rejecting H02 or H03 is 0.947
```

Bayesian Response-Adaptive Randomisation with a Control Group

In this package, Bayesian response-adaptive randomization refers to the Thall & Wathen method with a control group (Thall and Wathen 2007; Wathen and Thall 2017). This method can be applied not only to two-armed trials but also to multi-armed trials. Suppose the number of treatment groups is K and the posterior probabilities are $p_1, ..., p_K$ considering data available after recruitment of participant $t-1$. The generalized formula of allocation probability for arm k at the recruitment of participant t is given as

$$p_{k,t} = \frac{Pr(p_k = max\{p_1, ..., p_K\}|data_{t-1})^\gamma}{\sum_{i=1}^{K} Pr(p_i = max\{p_1, ..., p_K\}|data_{t-1})^\gamma}.$$

In this formula, γ is a tuning parameter (with a larger γ increasing the adaptiveness) with commonly used values $\gamma = 1, 0.5,$ and $n/2N$. The posterior probabilities $p_1, ..., p_K$ are often restricted to $[e, 1-e]$ for $0 < e < 1/2$. The decision rules of this method include three parts (Wathen and Thall 2017):

1. Stop arm k for futility, if $Pr(p_k > p_{control} + \delta | data_{t-1}) < 0.01$. It is recommended to choose δ by simulation to satisfy the futility stopping criterion; δ can take negative values;

2. Randomize the remaining participants at time t only to arms that have not been dropped due to futility;

3. Select the effective arm k at the end of the trial if $Pr(p_k > p_{control} + \delta_1 | data_{t-1}) > a_U$.

It should be noted

$*$ At the beginning, an initialization period is planned with equal randomization, followed by response-adaptive randomization.

$*$ The trial stops when all arms, except the control, are dropped.

$*$ δ_1 is the minimal expected effect to be observed at the end of the trial;

$*$ There is no early stopping for efficacy;

$*$ a_U controls the overall Type I error, which is recommended to be obtained by simulations;

$*$ More than one arm can be selected as effective at the final stage.

$*$ The formula to calculate allocation probability for arm k at the recruitment of participant t is based on p standing for the success probability. If p stands for the failure probability, the formula can be updated to

$$p_{k,t} = \frac{Pr(p_k = min\{p_1, ..., p_K\} | data_{t-1})^\gamma}{\sum_{i=1}^{K} Pr(p_i = min\{p_1, ..., p_K\} | data_{t-1})^\gamma}.$$

Similarly, the formula in decision rules can be updated to $Pr(p_{control} > p_k + \delta | data_{t-1}) < 0.01$ and $Pr(p_{control} > p_k + \delta_1 | data_{t-1}) > a_U$ respectively.

$*$ All of those formulas above can be generalized to continuous outcomes with p substituted by μ (representing the mean value).

To take the best advantage of Bayesian adaptive randomization, we can evaluate those rules after every participant enrolls. But the decision is made only based on participants with available data. In this package, the prior distributions for different groups can be specified individually, and the corresponding δ, δ_1 and a_U can be selected for each group to control the one-sided overall type I error or the nominal alpha at $\alpha/(K-1)$ for each comparison to the control group.

R code

For binary outcomes, a Beta prior denoted by $Beta(\alpha, \beta)$ is employed, which is conjugate to the likelihood of binomial distribution written as $B(n, p)$, resulting in the posterior Beta distribution $Beta(\alpha+np, \beta+n-np)$. There are two ways to calculate $Pr(p_k > p_{control} + \delta | data_{t-1})$: one is through simulations using Beta distributions, and the other is through direct integration. The formula for direct integration is given below:

$$P(X > Y + \delta) = \int_0^1 P(X > y + \delta)\, dF_Y(y) = \int_0^1 (1 - F_X(y + \delta)) f_Y(y)\, dy,$$

where X and Y are posterior distributions for success probabilities. Similarly, the calculation of $Pr(p_k = max\{p_1, ..., p_K\} | data_{t-1})$ can use the formula

$$P(X \leq Y) = \int_0^1 P(X \leq y)\, dF_Y(y) = \int_0^1 F_{X_1}(y) F_{X_2}(y) ... F_{X_K}(y) f_Y(y)\, dy.$$

An example of a three-armed trial is given below with some pre-specified parameters from the Beta distribution. Both methods yield similar results, but the latter is quicker.

```r
#### which.is.max is adapt from 'nnet' package
#### which.is.min is a rewrite from which.is.max
which.is.max <- function(x)
{
  y <- seq_along(x)[x == max(x)]
  if(length(y) > 1L) sample(y, 1L) else y
}

which.is.min <- function(x)
{
  y <- seq_along(x)[x == min(x)]
  if(length(y) > 1L) sample(y, 1L) else y
}

#Example: sample code using simulations
options(scipen=999)
set.seed(12345)
#Pre-specified parameters from the Beta distribution for each arm
alpha=c(30,41,35)
beta=c(30,20,27)
#Total number of treatment groups
armn<-3
#Number of treatment groups left at current stage
armleft<-c(1,2,3)
#Store simulation results for each arm
set.seed(12345)
result<-vector("list",length(armleft))
for (j in 1:length(armleft)) {
  result[[j]]<- as.data.frame(rbeta(1000000,alpha[armleft[j]],
                beta[armleft[j]]))
  colnames(result[[j]])<-sprintf("r%s",armleft[j])
}
#Expect the treatment group to have a larger treatment effect compared to the control group
#Combine results into a data frame and select the maximal value of each row
result1<-as.data.frame(do.call(cbind,result))
result1$max<-apply(result1, 1, which.is.max)
#Store results for Pr(p_k>p_{control}+\delta|data_{t-1})
theta1<-vector("list",armn)
#Store results for Pr(p_k=max{p_1,...,p_K}|data_{t-1})
pi<-vector("list",armn)
for (j in 1:length(armleft)) {
  theta1[[armleft[j]]]<-sum(result1[,j]>(result1[,1]+0.1))/1000000
  pi[[armleft[j]]]<-(sum(result1[,length(armleft)+1]==j)/1000000)
}
do.call(cbind,theta1)
```

```
##      [,1]     [,2]     [,3]
## [1,]    0 0.794355 0.347715
```

```r
#Return results: 0 0.794355 0.347715
do.call(cbind,pi)
```

```
##          [,1]     [,2]     [,3]
```

```
## [1,] 0.018097 0.879338 0.102565
```

```
#Return results: 0.018097 0.879338 0.102565

#Expect the treatment group to have a smaller treatment effect compared to the control group
#Combine results into a data frame and select the minimal value of each row
result1<-as.data.frame(do.call(cbind,result))
result1$max<-apply(result1, 1, which.is.min)
#Store results for Pr(p_{control}>p_k+\delta|data_{t-1})
theta1<-vector("list",armn)
#Store results for Pr(p_k=min{p_1,...,p_K}|data_{t-1})
pi<-vector("list",armn)
for (j in 1:length(armleft)) {
  theta1[[armleft[j]]]<-sum(result1[,j]<(result1[,1]-0.1))/1000000
  pi[[armleft[j]]]<-(sum(result1[,length(armleft)+1]==j)/1000000)
}
do.call(cbind,theta1)
```

```
##      [,1]     [,2]   [,3]
## [1,]    0 0.001049 0.0335
```

```
#Return results: 0 0.001049 0.0335
do.call(cbind,pi)
```

```
##          [,1]    [,2]     [,3]
## [1,] 0.755607 0.01215 0.232243
```

```
#Return results: 0.755607 0.01215 0.232243
```

```
#Example: Sample code using Integration
#Expect the treatment group to have a larger treatment effect compared to the control group
#Calculate results of Pr(p_k>p_{control}+\delta|data_{t-1})
pgreater_beta(a1=alpha[1],b1=beta[1],a2=alpha[2], b2=beta[2],delta=0.1,side='upper')
```

```
## [1] 0.7951487
```

```
pgreater_beta(a1=alpha[1],b1=beta[1],a2=alpha[3], b2=beta[3],delta=0.1,side='upper')
```

```
## [1] 0.3477606
```

```
#Return results:  0.7951487 0.3477606

#Calculate results of Pr(p_k=max\{p_1,...,p_K\}|data_{t-1})
pmax_beta(armn=3,a2=alpha[3],b2=beta[3],a3=alpha[2],b3=beta[2],a1=alpha[1],
     b1=beta[1],side='upper')
```

```
## [1] 0.01796526
```

```
pmax_beta(armn=3,a2=alpha[1],b2=beta[1],a3=alpha[3],b3=beta[3],a1=alpha[2],
     b1=beta[2],side='upper')
```

```
## [1] 0.8788907
```

```
pmax_beta(armn=3,a2=alpha[1],b2=beta[1],a3=alpha[2],b3=beta[2],a1=alpha[3],
     b1=beta[3],side='upper')
```

```
## [1] 0.1031441
```

```
#Return results: 0.01796526  0.8788907  0.1031441

#Expect the treatment group to have a smaller treatment effect compared to the control group
#Calculate results of Pr(p_{control}>p_k+\delta|data_{t-1})
pgreater_beta(a1=alpha[1],b1=beta[1],a2=alpha[2],b2=beta[2],delta=-0.1,side='lower')
```

```
## [1] 0.001093548
```

```
pgreater_beta(a1=alpha[1],b1=beta[1],a2=alpha[3],b2=beta[3],delta=-0.1,side='lower')
```

```
## [1] 0.03348547
```

```
#Return results:  0.001093548  0.03348547

#Calculate results of Pr(p_k=min\{p_1,...,p_K\}|data_{t-1})
pmax_beta(armn=3,a2=alpha[3],b2=beta[3],a3=alpha[2],b3=beta[2],a1=alpha[1],
     b1=beta[1],side='lower')
```

```
## [1] 0.7560864
```

```
pmax_beta(armn=3,a2=alpha[1],b2=beta[1],a3=alpha[3],b3=beta[3],a1=alpha[2],
     b1=beta[2],side='lower')
```

```
## [1] 0.01230027
```

```
pmax_beta(armn=3,a2=alpha[1],b2=beta[1],a3=alpha[2],b3=beta[2],a1=alpha[3],
     b1=beta[3],side='lower')
```

```
## [1] 0.2316133
```

```
#Return results: 0.7560864  0.01230027  0.2316133
```

The selection of a_U can be made under the intermediate hypothesis, halfway between the null and alternative hypotheses. This option arises because the test statistics is $Pr(p_k > p_{control} + \delta | data_t)$, which depends on the success probability of each treatment group. So, it makes more sense to use the intermediate hypothesis to avoid excessively large or small power. Suppose the alternative hypothesis is $c(0.2, 0.4)$, then the intermediate hypothesis use to select a_U is $c(0.3, 0.3)$.

```
set.seed(12345)
#Example: Select a_U by calling brar_au_binary 2000 times
simnull3<-lapply(1:2000,function(x){
  set.seed(x)
  brar_select_au_binary(Pats=10,nMax=50000,TimeToOutcome=0,enrollrate=0.9,N1=24,armn=2,
        h=c(0.3,0.3),N2=224,tp=1,armlabel=c(1, 2),blocksize=4,alpha1=1,beta1=1,
        alpha2=1,beta2=1,minstart=24,deltaa=-0.07,tpp=0,deltaa1=0.1,side='upper')
})

#Obtain the data set of test statistics
simf<-list()
for (xx in 1:2000){
  if (any(simnull3[[xx]][24:223,2]<0.01)){
    simf[[xx]]<-NA
  }  else{
    simf[[xx]]<-simnull3[[xx]][224,2]
  }
}
simf<-do.call(rbind,simf)

#Ensure that around 1% of the trials stop for futility
sum(is.na(simf)) #20
#Select a_U to make sure that an overall type I error is around 0.025
sum(simf>0.7591,na.rm=T)/2000 #0.025
#The selected a_U is 0.7591.
```

For continuous outcomes with known variance σ, a normal prior for the mean is commonly used, noted as $\theta \sim N(\mu, \sigma/n_0)$, where n_0 is the implicit sample size in the prior. This prior conjugates to the likelihood of normal distribution, written as $y \sim N(\theta, \sigma/n_1)$ with y represents the outcomes of observations and n_1 stands for the number of observations, resulting in the posterior normal distribution $N(\frac{n_0\mu+n_1y}{n_0+n_1}, \frac{\sigma^2}{n_0+n_1})$.

For continuous outcomes with unknown variance, a Normal-Inverse-Gamma prior is employed with $\theta \sim N(m_0, V_0\sigma_0^2)$ and $\sigma_0^2 \sim IG(a_0, b_0)$. The prior distribution can be written as $NIG(mean = m_0, variance = V_0 \times \sigma_0^2, shape = a_0, rate = b_0)$, which conjugates to the likelihood of normal distribution $y \sim N(\theta, \sigma^2)$, resulting in the posterior Normal-Inverse-Gamma distribution $NIG(\mu_n, \sigma_n, a_n, b_n) = NIG((m_0/V_0 + \sum_{i=1}^n y_i)V_n, 1/(1/V_0 + n), a_0 + n/2, b_0 + [m_0^2/V_0 + \sum_{i=1}^n y_i^2 - m_n^2/V_n]/2)$ with y_i represents the outcome of each observation and n stands for the number of observations (Murphy 2007). Following the integration formulas as before, we should use the marginal posterior distribution of μ which is a t distribution in this case. According to Murphy (2007), we first convert hyper-parameters of Normal-Inverse-Gamma to Normal-Inverse-Chi-Squared distribution. Then, we use the marginal posterior of μ given in section 5.3.2 of Murphy (2007). An example of a three-armed trial with unknown variances is given below with pre-specified prior distributions.

```
#Example: Sample code using integration
options(scipen=999)
set.seed(12345)
#Pre-specified hyper-parameters in prior distributions assumed to be the same across all
#three groups.
para<-list(V=1/2,a=0.5,m=9.1/100,b=0.00002)
#Update hyper-parameters from the Normal-Inverse-Gamma distribution to the
#Normal-Inverse-Chi-Squared distribution.
par<-convert_gamma_to_chisq(para)
#Update hyper-parameters with some data
set.seed(123451)
```

```r
y1<-rnorm(100,0.091,0.009)
par1<-update_par_nichisq(y1, par)
set.seed(123452)
y2<-rnorm(90,0.09,0.009)
par2<-update_par_nichisq(y2, par)
set.seed(123453)
y3<-rnorm(110,0.0892,0.009)
par3<-update_par_nichisq(y3, par)

#Calculate results of Pr(p_{control}>p_k+\delta|data_{t-1}) with delta=0
pgreater_NIX(par1,par2,side='lower')
```

```
## [1] 0.1959142
```

```r
pgreater_NIX(par1,par3,side='lower')
```

```
## [1] 0.8115975
```

```r
#Return results: 0.1959142  0.8115975
#Calculate results for Pr(p_k=min{p_1,...,p_K}|data_{t-1})
pmax_NIX(armn=3,par1=par1,par2=par2,par3=par3,side='lower')
```

```
## [1] 0.1801636
```

```r
pmax_NIX(armn=3,par1=par2,par2=par1,par3=par3,side='lower')
```

```
## [1] 0.02758085
```

```r
pmax_NIX(armn=3,par1=par3,par2=par2,par3=par1,side='lower')
```

```
## [1] 0.7922556
```

```r
#Return results: 0.1801636  0.02758085  0.7922556

#Calculate results of Pr(p_k>p_{control}+\delta|data_{t-1}) with delta=0
pgreater_NIX(par1,par2,side='upper')
```

```
## [1] 0.8040858
```

```r
pgreater_NIX(par1,par3,side='upper')
```

```
## [1] 0.1884025
```

```r
#Return results: 0.8040858  0.1884025
#Calculate results for Pr(p_k=max\{p_1,...,p_K\}|data_{t-1})
pmax_NIX(armn=3,par1=par1,par2=par2,par3=par3,side='upper')
```

```
## [1] 0.1876753
```

```r
pmax_NIX(armn=3,par1=par2,par2=par1,par3=par3,side='upper')
```

```
## [1] 0.7873393
```

```r
pmax_NIX(armn=3,par1=par3,par2=par2,par3=par1,side='upper')
```

```
## [1] 0.02498539
```

```r
#Return results: 0.1876753  0.7873393  0.02498539
```

```r
#Example: sample code using simulations
#Convert hyper-parameters to Normal-Inverse-Gamma distribution
convert_chisq_to_gamma<-function(cpar){
  list(
    m=cpar$mu,
    V=1/cpar$kappa,
    a=cpar$nu/2,
    b=cpar$nu*cpar$sigsq/2
  )
}
rnigamma<-function(nsim,par){
  sigma2<-1/rgamma(nsim, shape=par$a,rate=par$b)
  mu<-rnorm(nsim, par$m, sqrt(sigma2*par$V))
  cbind(mu,sigma2)
}
set.seed(12341)
NIG_par1<-rnigamma(1e5,convert_chisq_to_gamma(par1))
set.seed(12342)
NIG_par2<-rnigamma(1e5,convert_chisq_to_gamma(par2))
set.seed(12343)
NIG_par3<-rnigamma(1e5,convert_chisq_to_gamma(par3))

#Calculate results of Pr(p_{control}>p_k+\delta|data_{t-1}) with delta=0
#Calculate results for Pr(p_k=min{p_1,...,p_K}|data_{t-1})
dat<-matrix(NA,1e5,5)
for (i in 1:1e5){
  dat1<-rnorm(1e5,NIG_par1[i,1],NIG_par1[i,2])
  dat2<-rnorm(1e5,NIG_par2[i,1],NIG_par2[i,2])
  dat3<-rnorm(1e5,NIG_par3[i,1],NIG_par3[i,2])
  dat[i,1]<-sum(dat1>dat2)/1e5
  dat[i,2]<-sum(dat1>dat3)/1e5
  minimal<-base::pmin(dat1,dat2,dat3)
  dat[i,3]<-sum(minimal==dat1)/1e5
  dat[i,4]<-sum(minimal==dat2)/1e5
  dat[i,5]<-sum(minimal==dat3)/1e5
}
mean(dat[,1])
mean(dat[,2])
#Return results: 0.1994863 0.8113217
mean(dat[,3])
mean(dat[,4])
mean(dat[,5])
```

```r
#Return results: 0.1802267 0.0285785 0.7911948

#Calculate results of Pr(p_k>p_{control}+\delta|data_{t-1}) with delta=0
#Calculate results for Pr(p_k=max{p_1,...,p_K}|data_{t-1})
dat<-matrix(NA,1e5,5)
for (i in 1:1e5){
  dat1<-rnorm(1e5,NIG_par1[i,1],NIG_par1[i,2])
  dat2<-rnorm(1e5,NIG_par2[i,1],NIG_par2[i,2])
  dat3<-rnorm(1e5,NIG_par3[i,1],NIG_par3[i,2])
  dat[i,1]<-sum(dat1<dat2)/1e5
  dat[i,2]<-sum(dat1<dat3)/1e5
  maximal<-base::pmax(dat1,dat2,dat3)
  dat[i,3]<-sum(maximal==dat1)/1e5
  dat[i,4]<-sum(maximal==dat2)/1e5
  dat[i,5]<-sum(maximal==dat3)/1e5
}
mean(dat[,1])
mean(dat[,2])
#Return results: 0.8005126  0.1886812
mean(dat[,3])
mean(dat[,4])
mean(dat[,5])
#Return results:  0.1910363 0.7838454 0.02511826
```

```r
#Example: Select a_U by calling brar_au_binary 2000 times
set.seed(12345)
simnull4<-lapply(1:2000,function(x){
  brar_select_au_unknown_var(Pats=10,nMax=50000,TimeToOutcome=expression(
  rnorm(length( vStartTime ),30, 3)),enrollrate=0.1, N1=192,armn=3,
  N2=1920,tp=1,armlabel=c(1,2,3),blocksize=6,
  mean=c((9.1/100+8.92/100+8.92/100)/3,(9.1/100+8.92/100+8.92/100)/3,
         (9.1/100+8.92/100+8.92/100)/3),sd=c(0.009,0.009,0.009),minstart=192,
         deltaa=c(0.00077,0.00077),tpp=1,deltaa1=c(0,0),V01=1/2,a01=0.3,m01=9/100,
         b01=0.00001,side='lower')
  })

#Obtain the data set of test statistics
simf<-list()
for (xx in 1:2000){
  if (any(simnull4[[xx]][192:1919,2]<0.01)){
    simf[[xx]]<-NA
  }  else{
    simf[[xx]]<-simnull4[[xx]][1920,2]
  }
}
simf4a<-do.call(rbind,simf)

simf<-list()
for (xx in 1:2000){
  if (any(simnull4[[xx]][192:1919,3]<0.01)){
    simf[[xx]]<-NA
  }  else{
    simf[[xx]]<-simnull4[[xx]][1920,3]
```

```
  }
}
simf4b<-do.call(rbind,simf)

#Ensure that around 1% of the trials stop for futility
sum(is.na(simf4a)) #21
sum(is.na(simf4b)) #22
#Select a_U to make sure that the overall type I error is around 0.025
sum((simf4a[,1]>0.9836| simf4b[,1]>0.9836),na.rm=T )/2000#0.025
#The selected a_U is 0.9836.

#Example to obtain the power
sim4<-lapply(1:1000,function(x) {
    sim_brar_unknown_var(Pats=10,nMax=50000,TimeToOutcome=expression(rnorm(
      length(vStartTime ),30, 3)),enrollrate=0.1,N1=192,armn=3,
            a_U=c(0.9836,0.9836),N2=1920,tp=1,armlabel=c(1,2,3),blocksize=6,
            mean=c(9.1/100,8.92/100,8.92/100),sd=c(0.009,0.009,0.009),
            minstart=192,deltaa=c(0.00077,0.00077),tpp=1,deltaa1=c(0,0),
            V01=1/2,a01=0.3,m01=9/100,b01=0.00001,side='lower')

  })

decision<-t(sapply(sim4, "[[", 1))
sum(decision[,2]=='Superiorityfinal')/1000 #0.882
#The simulated power from 1000 simulations is 0.882.
```

Forward-Looking Gittins Index

In response-adaptive randomisation, there is a trade-off between providing the best treatment to the future participants and providing it to the next participant, known as the learn-versus-earn trade-off (Villar, Wason, and Bowden 2015). This trade-off involves considering the benefits to participants to be enrolled in the future and to those currently enrolled. Learning involves exploring long-term value to obtain sufficient benefits, while earning entails exploiting short-term rewards to identify the best investment options. For historical reason, this problem is also called the "multi-armed bandit problem". One of the optimal solution to this question is provided by the Gittins Index, the basic principle of which is to find the balance between the learn-versus-earn trade-off, maximizing benefits through Bayesian decision theory. Suppose the t-th participant is assigned to treatment k with $k = 1, ..., K$ and $k = 1$ stands for the control group. The mathematical representation of this problem using a binary endpoint is to determine the allocation sequence

$$\pi_{\text{opt}}(\tilde{x}_0) = \arg\max_{\pi\in\Pi} E^{\pi}[(\sum_{t=0}^{T-1}\sum_{k=1}^{K} d^t P_k a_{k,t})|\tilde{x}_0].$$

In this formula, P_k is the posterior mean,

$$\Pi = \{a_{k,t}|1 \le k \le K, \sum_{k=1}^{K} a_{k,t} = 1\}$$

is a family of allocation rules,

$$\tilde{x}_0 = (x_{k,0})_{k=1}^{K}$$

is the initial state considering the prior information, $d \in (0, 1)$ is a specified discount factor and $E^{\pi}[.]$ is the average responses of participants attained with prior information. Our objective is to maximize the expected discounted number of successes as t ranges from 0 to $T - 1$.

The Gittins Index is the optimal solution to the optimization question above, given by

$$\mathcal{G}(X_{k,t}) = \sup_{\tau \geq 1} \frac{E[(\sum_{i=0}^{\tau-1} P_k d^i)|X_{k,t}]}{\sum_{i=0}^{\tau-1} d^i}.$$

The numerator is the expected total discounted reward from t to $t+\tau-1$, and the denominator is the expected total discounted cost from t to $t+\tau-1$. So $\mathcal{G}(X_{k,t})$ means the maximal total benefits based on the choice of treatment k at stage t.

The motivation for the Gittins Index comes from clinical trials, and there is much more awareness of its potential practical applications than there are actual implementations (Gittins and Jones, 1979b). Villar, Wason, and Bowden (2015) proposed the forward-looking Gittins Index algorithm, which allows participants to enroll in trials with block size m over J stages. The allocation probability of each participant in block j being assigned to treatment k is given by:

$$\pi_{k,j} = \frac{1}{m} \sum_{t=(j-1)m+1}^{jm} \Big[\sum_{\tilde{x}_{t-1} \in \Omega_{t-1}} Pr(a_{k,t}^{GI} = 1 | \tilde{X}_{t-1} = \tilde{x}_{t-1}) \times Pr(\tilde{X}_{t-1} = \tilde{x}_{t-1} | \tilde{X}_{(j-1)m} = \tilde{x}_{(j-1)m}) \Big].$$

The allocation probability of participants to treatment k depends on the data observed up to the $(j-1)$ block with $(j-1) \times m$ participants, written as $\tilde{x}_{(j-1)m}$, and the current information at block j with participant $t-1$. Ω_{t-1} is a set of possible values for $\tilde{X}_{t-1}$ given $\tilde{x}_{(j-1)m}$ for participants in block j using Gittins Index with notation $a_{k,t}^{GI}$.

- When $j = 1$ and all treatment groups have the same prior distributions, $p_{k,j} = 1/(K+1)$;
- When $j \geq 2$, the allocation of the first participant in block j will be determined by the highest Gittins Index with information from the $(j-1) \times m$ participants;
- When $j \geq 2$, the allocation for the second to the last participants in block j will depend on the posterior distribution of future data given $\tilde{x}_{(j-1)m}$.

To maintain a certain allocation probability to the control group, a slight modification to this rule is introduced with $p_{1,j} = 1/K$ and the allocation probabilities to other treatment groups are calculated using their corresponding proportions and normalized to sum to 1, this process is known as the controlled forward-looking Gittins Index rule (Villar, Wason, and Bowden 2015).

An adaptation of the procedure for binary outcome to continuous outcome is provided in Williamson and Villar (2019). All the theory and rules remain similar, with some updates to the calculation of Gittins indices. For continuous outcomes with known variances and normal priors, the Gittins Index can be calculated by

$$G(\tilde{m}_{k,t}, \sigma_k, n_{k,t}) = \tilde{m}_{k,t} + \sigma_k G(0, 1, {n_k}^0 + n_{k,t}, d);$$

For continuous outcomes with unknown variances and Normal-Inverse-Gamma priors, the Gittins Index can be calculated by

$$G(\tilde{m}_{k,t}, \tilde{\sigma}_{k,t}, n_{k,t}) = \tilde{m}_{k,t} + \tilde{\sigma}_{k,t} G(0, 1, {n_k}^0 + n_{k,t}, d),$$

where $\tilde{m}_{k,t}$ and $\tilde{\sigma}_{k,t}$ are the posterior mean and posterior standard deviation of treatment group k at time t (Gittins, Glazebrook, and Weber 2011).

R code

To avoid the inflation of the type I error probability, it is necessary to select the critical cut-off value under the null hypothesis by simulation (Smith and Villar 2017). After the cut-off value is selected, call `sim_flgi_binary()`, `sim_flgi_known_var()` or `sim_flgi_unknown_var()` multiple times to simulate trials and obtain results of interest. This packages is built to obtain type I error and power based on the frequentist Z test statistics and T test statistics. Due to the heavy computational burden involved with large sample sizes, it is recommended to run the code on high-performance platforms. An example is provided below for continuous outcomes with known variances.

```
#Example: The forward-looking Gittins Index rule applied in continuous outcomes with known
#variances
#Select cut-off values under null hypothesis for continuous outcomes with known variances.
#The delayed responses follow a normal distribution with a mean of 30 days, sd of 3 days
#and enrollment rate of 1.
set.seed(123452)
stopbound5<-lapply(1:1000,function(x){flgi_cut_off_known_var(
Gittinstype='KV',df=0.995,Pats=10,nMax=50000,TimeToOutcome=0,enrollrate=1,K=3,
noRuns2=100,Tsize=120,block=8,rule='FLGI PM',prior_n=rep(1,3),prior_mean=rep(0,3),
mean=c(-0.05,-0.05,-0.05),sd=c(0.346,0.346,0.346),side='upper')})
stopbound5a<-do.call(rbind,stopbound5)
sum(stopbound5a[,1]>2.074 | stopbound5a[,2]> 2.074)/1000 #0.05
#The selected cut-off value is 2.074 with an overall upper one-sided type I error of 0.05.
#It is recommended to run more simulations to obtain a more accurate cut-off value.

#Calculate power based on 1000 simulations
set.seed(123452)
sim5<-lapply(1:1000,function(x){sim_flgi_known_var(Gittinstype='KV',
df=0.995,Pats=10,nMax=50000,TimeToOutcome=0,enrollrate=1,K=3,
noRuns2=100,Tsize=120,block=8,rule='FLGI PM', prior_n=rep(1,3),
prior_mean=rep(0,3), mean=c(-0.05,0.07,0.13),sd=c(0.346,0.346,0.346),
stopbound =2.074,side='upper')})
h1decision<-t(sapply(sim5, "[[", 1))
sum(h1decision[,1]==1)/1000 #0.074
sum(h1decision[,2]==1)/1000 #0.244
sum(h1decision[,1]==1 | h1decision[,2]==1)/1000 #0.267
#Marginal power of rejecting H02 is 0.074
#Marginal power of rejecting H03 is 0.244
#Power of rejecting H02 or H03 is 0.267
```

References

Atkinson, Anthony, and Atanu Biswas. 2013. *Randomised Response-Adaptive Designs in Clinical Trials*. London: Routledge. https://doi.org/10.1201/b16101.

Bello, Ghalib, and Roy Sabo. 2016. "Outcome-Adaptive Allocation with Natural Lead-in for Three-Group Trials with Binary Outcomes." *Journal of Statistical Computation and Simulation* 86 (12): 2441–49.

Biswas, A., and S. Mandal. 2004. "Optimal Adaptive Designs in Phase III Clinical Trials for Continuous Responses with Covariates." In *mODa 7 — Advances in Model-Oriented Design and Analysis*.

Gittins, John, Kevin Glazebrook, and Richard Weber. 2011. *Multi-Armed Bandit Allocation Indices, 2nd Edition*. Vol. 33. https://doi.org/10.1002/9780470980033.ch8.

Hu, Feifang, and Li-Xin Zhang. 2004. "Asymptotic Properties of Doubly Adaptive Biased Coin Designs for Multitreatment Clinical Trials." *The Annals of Statistics* 32 (1): 268–301.

Jeon, Youngsook, and Hu Feifang. 2010. "Optimal Adaptive Designs for Binary Response Trials with Three Treatments." *Statistics in Biopharmaceutical Research* 2 (3): 310–18. https://doi.org/10.1198/sbr.2009.0056.

Murphy, Kevin. 2007. "Conjugate Bayesian Analysis of the Gaussian Distribution." The University of British Columbia. https://www.cs.ubc.ca//~murphyk/Papers/bayesGauss.pdf.

Rosenberger, W. F., and J. M. Lachin. 2015. *Randomization in Clinical Trials: Theory and Practice*. John Wiley & Sons.

Smith, Adam, and Sofia Villar. 2017. "Bayesian Adaptive Bandit-Based Designs Using the Gittins Index for Multi-Armed Trials with Normally Distributed Endpoints." *Journal of Applied Statistics* 45 (6). https://doi.org/10.1080/02664763.2017.1342780.

Sverdlov, Oleksandr, and William Rosenberger. 2013. "On Recent Advances in Optimal Allocation Designs in Clinical Trials." *Journal of Statistical Theory and Practice* 7 (4). https://doi.org/10.1080/15598608.2013.783726.

Thall, Peter, and J Wathen. 2007. "Practical Bayesian Adaptive Randomization in Clinical Trials." *European Journal of Cancer* 43 (5). https://doi.org/10.1016/j.ejca.2007.01.006.

Tymofyeyev, Yevgen, William F Rosenberger, and Feifang Hu. 2007. "Implementing Optimal Allocation in Sequential Binary Response Experiments." *Journal of the American Statistical Association* 102 (477). https://doi.org/10.1198/016214506000000906.

Villar, Sofia, James Wason, and Jack Bowden. 2015. "Response-Adaptive Randomization for Multi-Arm Clinical Trials Using the Forward Looking Gittins Index Rule." *Biometrics* 71 (4): 969–78. https://doi.org/10.1111/biom.12337.

Wathen, J, and Peter Thall. 2017. "A Simulation Study of Outcome Adaptive Randomization in Multi-Arm Clinical Trials." *Clinical Trials* 14 (5): 432–40. https://doi.org/10.1177/1740774517692302.

Wei, L. J., and S. Durham. 1978. "The Randomized Play-the-Winner Rule in Medical Trials." *Journal of the American Statistical Association* 73 (364): 840–43.

Williamson, S. Faye, and Sofia Villar. 2019. "A Response-adaptive Randomization Procedure for Multi-armed Clinical Trials with Normally Distributed Outcomes." *Biometrics* 76 (1): 197–209. https://doi.org/10.1111/biom.13119.