Web-based Supplementary Materials for "A Utility-Based Design for Randomized Comparative Trials with Ordinal Outcomes and Prognostic Subgroups" by Thomas A. Murray, Ying Yuan, Peter F. Thall, Joan H. Elizondo and Wayne L. Hofstetter

# Re-statement and Proof of Theorem 1

**Theorem 2.** *If $\boldsymbol{\pi}(x,a)$ stochastically dominates $\boldsymbol{\pi}(x,a')$, then $\overline{U}(\boldsymbol{\pi}(x,a)) > \overline{U}(\boldsymbol{\pi}(x,a'))$ for all admissible $U(Y)$ such that $U(Y=0) > U(Y=1) > \cdots > U(Y=4) > U(Y=5)$.*

*Proof.* Note that

$$\begin{aligned}
\overline{U}(\boldsymbol{\pi}(x,a)) - \overline{U}(\boldsymbol{\pi}(x,a')) &= \sum_{y=0}^{5} U(Y=y)\pi_y(x,a) - \sum_{y=0}^{5} U(Y=y)\pi_y(x,a') \\
&= U(Y=0)[\pi_0^+(x,a)] + \sum_{y=1}^{4} U(Y=y)[\pi_y^+(x,a) - \pi_{y-1}^+(x,a)] + U(Y=5)[1-\pi_4^+(x,a)] \\
&\quad - U(Y=0)[\pi_0^+(x,a')] - \sum_{y=1}^{4} U(Y=y)[\pi_y^+(x,a') - \pi_{y-1}^+(x,a')] - U(Y=5)[1-\pi_4^+(x,a')] \\
&= \sum_{y=0}^{4} [U(Y=y) - U(Y=y+1)] \left[\pi_y^+(x,a) - \pi_y^+(x,a')\right].
\end{aligned}$$

Therefore, if for $y=0,\ldots,4$, $U(Y=y) > U(Y=y+1)$, $\pi_y^+(x,a) \geq \pi_y^+(x,a')$, and for some $y=0,\ldots,4$, $\pi_y^+(x,a) > \pi_y^+(x,a')$, then

$$\sum_{y=0}^{4} [U(Y=y) - U(Y=y+1)] \left[\pi_y^+(x,a) - \pi_y^+(x,a')\right] > 0,$$

and thus, $\overline{U}(\boldsymbol{\pi}(x,a)) > \overline{U}(\boldsymbol{\pi}(x,a'))$. □

# R Software

## 7.1 Readme.txt

```
### Author: Thomas A Murray (8tmurray@gmail.com)
### Date: 14 June 2017

This folder contains two files for replicating the simulation study reported in Murray et
     al. (2017).

(i) functions.R, which contains the relevant R functions.
(iv) simulation.R, which implements the simulation study reported in Murray et al. (2017)

Because simulation.R 'sources' functions.R, these files need to be kept in the same
    working directory.

This software requires R 3.2.0 (or newer), which is freely available for download here:
    https://www.r-project.org/
JAGS 4.2.0 (or newer), which is freely available for download here: https://sourceforge.
    net/projects/mcmc-jags/files/

This software also requires the R packages 'R2jags' (version 0.7 or newer) and 'gsDesign'
     (version 3.1 or newer).

This software was written and executed on a 64-bit machine running Windows 7 Enterprise.
functions.R, which is also indicated in the file.
```

## 7.2 functions.R

```
#   This file contains R functions for implementing the Nuprehab Trial
#
#   Requirements, R version 3.3.2 or newer (available at https://www.r-project.org/)
#               JAGS version 4.2.0 or newer (available at https://sourceforge.net/
    projects/mcmc-jags/files/)
#               R packages, "R2jags" version 0.7 or newer
#                           "gsDesign" version 3.1 or newer
#
```

```
#   (Originally written for a machine with 64-Bit Windows 7 Enterprise)


### Function to generate datasets using stratified randomization (this function is
    required for simulations and testing, but not for trial conduct)
# Inputs:
#  n - sample size
#  pis - 2x2x6 array defining POM score probabilities in each patient subgroup and
    treatment arm
#         i.e., pis[sgp, trt, response level]
#  prob.primary - probability that a particular subject is a primary patient, default is
    0.6
#  block.size - block size for stratified randomization, default is 4
#Outputs:
# data.frame with the three variables
#    y - POM Scores
#  sgp - patient subgroup: 1-Primary, 2-Salvage
#  trt - treatment arm: 1-Control, 2-Nutritional Prehabilitation
get.dataset <- function(n,pis,prob.primary=0.6,block.size=4){
sgp = rbinom(n,1,1-prob.primary)+1
trt.prim = as.vector(sapply(1:ceiling(n/block.size),function(x) sample(rep(c(1,2),each=
    block.size/2))))
trt.salv = as.vector(sapply(1:ceiling(n/block.size),function(x) sample(rep(c(1,2),each=
    block.size/2))))
patid.prim = sapply(1:n,function(i) sum(sgp[1:i]==1))
patid.salv = sapply(1:n,function(i) sum(sgp[1:i]==2))
trt = sapply(1:n,function(i) ifelse(sgp[i]==1,trt.prim[patid.prim[i]],trt.salv[patid.salv
    [i]]))
y = sapply(1:n,function(i) sample(0:5,1,prob=pis[sgp[i],trt[i],]))
return(data.frame(y=y,trt=trt,sgp=sgp))
}
```

```
### Function to fit non-proportional odds cumulative logistic regression model in JAGS
library('R2jags')
npo.model <- function(){
### Likelihood ###
# Cumulative Clavien-Dindo Probabilities, i.e., Prob(y <= s | Sgp, Trt), s=0,...,4
for(s in 1:5){
logit(Q[1,1,s]) <- alpha[s] - 0.5*gamma[1,s] - 0.5*gamma[2,s] + 0.25*gamma[3,s] #Primary,
    Control
logit(Q[2,1,s]) <- alpha[s] + 0.5*gamma[1,s] - 0.5*gamma[2,s] - 0.25*gamma[3,s] #Salvage,
    Control
logit(Q[1,2,s]) <- alpha[s] - 0.5*gamma[1,s] + 0.5*gamma[2,s] - 0.25*gamma[3,s] #Primary,
    NuPrehab
logit(Q[2,2,s]) <- alpha[s] + 0.5*gamma[1,s] + 0.5*gamma[2,s] + 0.25*gamma[3,s] #Salvage,
    NuPrehab
}
# Clavien-Dindo Probabilities, i.e., Prob(y = s | Sgp, Trt), s=0,...,5
for(j in 1:2){
for(k in 1:2){
pi[j,k,1] <- Q[j,k,1]
for(s in 2:5){
pi[j,k,s] <- Q[j,k,s]-Q[j,k,s-1]
}
pi[j,k,6] <- 1-Q[j,k,5]
}
}
# Likelihood Contributions
for(i in 1:n){
y.plus.one[i] ~ dcat(pi[sgp[i],trt[i],])
}

### Prior ###
alpha[1] ~ dt(alpha.star[1],0.4,5)
```

```
for(s in 2:5){
alpha[s] ~ dt(alpha.star[s],0.4,5)%_%T(alpha[s-1]+0.5*(abs(gamma[1,s-1]-gamma[1,s])+abs(
    gamma[2,s-1]-gamma[2,s])+0.5*abs(gamma[3,s-1]-gamma[3,s])),)
}
for(m in 1:3){
for(s in 1:5){
gamma[m,s] ~ dnorm(beta[m],tau[m])
}
beta[m] ~ dt(beta.star[m],0.4,5)
tau[m] <- 1/sigma[m]^2
sigma[m] ~ dnorm(0,4)%_%T(0,)
}

### Mean Utilities ###
for(j in 1:2){
for(k in 1:2){
u[j,k] <- inprod(U[1:6],pi[j,k,1:6])
}
}
}
```

```
### Function to fit NPO model in JAGS and return the posterior probabilities for the
    decision rules
# Inputs:
#   dataset - data.frame with the three variables, (y, sgp, trt)
#   U - numerical utilities, default corresponds to the values for the IMPACT trial
#   pi.star - prior POM score probabilities, default corresponds to the values for the
    IMPACT trial
#   nsamps - number of posterior samples to generate, default is 10000
```

```
#   warmup - number of warmup samples, default is 500
# Outputs:
#   vector of the four posterior probabilities,
#     1 - posterior probability that nutritional prehabilitation is superior to the
    control for primary patients
#     2 - posterior probability that nutritional prehabilitation is superior to the
    control for salvage patients
#     3 - posterior probability that nutritional prehabilitation is inferior to the
    control for primary patients
#     4 - posterior probability that nutritional prehabilitation is inferior to the
    control for salvage patients
npo.analysis <- function(dataset,U=c(100,80,65,25,10,0),pi.star=rbind(c
    (0.50,0.20,0.10,0.10,0.05,0.05),c(0.30,0.25,0.10,0.10,0.10,0.15)),nsamps=10000,warmup
    =500){
# Data preparation
n = nrow(dataset); y = dataset$y; sgp = dataset$sgp; trt = dataset$trt; y.plus.one = y+1
dat = list("n","y.plus.one","sgp","trt","alpha.star","beta.star","U"); params = c("u")

# Prior Specification
alpha.star <- colMeans(rbind(sapply(1:5,function(s) log(sum(pi.star[1,1:s])/(1-sum(pi.
    star[1,1:s])))),sapply(1:5,function(s) log(sum(pi.star[2,1:s])/(1-sum(pi.star[2,1:s])
    )))))
beta.star <- c(mean(apply(rbind(sapply(1:5,function(s) log(sum(pi.star[1,1:s])/(1-sum(pi.
    star[1,1:s])))),sapply(1:5,function(s) log(sum(pi.star[2,1:s])/(1-sum(pi.star[2,1:s])
    )))),2,function(x) diff(x))),0,0)

# Fit model with interaction
fit <- jags(dat,inits=NULL, parameters.to.save=params, model.file=npo.model, n.iter=
    nsamps+warmup, n.burnin=warmup, n.chains=1, n.thin=1)

# Calculate Posterior Probabilities
pp.prim.sup = mean(fit$BUGSoutput$sims.list$u[,1,2] > fit$BUGSoutput$sims.list$u[,1,1])
pp.salv.sup = mean(fit$BUGSoutput$sims.list$u[,2,2] > fit$BUGSoutput$sims.list$u[,2,1])
```

```
pp.prim.inf = mean(fit$BUGSoutput$sims.list$u[,1,2] < fit$BUGSoutput$sims.list$u[,1,1])
pp.salv.inf = mean(fit$BUGSoutput$sims.list$u[,2,2] < fit$BUGSoutput$sims.list$u[,2,1])

return(round(c(pp.prim.sup,pp.salv.sup,pp.prim.inf,pp.salv.inf),4))
}



### Function to get the decision for the NPO model-based stratified medicine design at
    the current analysis
# Inputs:
#   dataset - data.frame with the three variables, (y, sgp, trt)
#   analysis - "Interim" or "Final"
#   active.subgroups - active subgroups, i.e,. the subgroups for which no decision has
    been made at a previous analysis
#                     at the interim analysis this will be c("P","S"), i.e., both Primary
     (P) and Salvage (S) subgroups are active
#                     at the final analysis this may be either c("P","S"), c("P"), or c("
    S")
#   U - numerical utilities, default corresponds to the values for the IMPACT trial
#   pi.star - prior POM score probabilities, default corresponds to the values for the
    IMPACT trial
#   nsamps - number of posterior samples to generate, default is 10000
#   warmup - number of warmup samples, default is 500
#   alpha - type I error, default is 0.05
# Outputs: decision
library('gsDesign')
get.decision <- function(dataset,analysis=c("Interim","Final"),active.subgroups=c("P","S
    "),U=c(100,80,65,25,10,0),pi.star=rbind(c(0.50,0.20,0.10,0.10,0.05,0.05),c
    (0.30,0.25,0.10,0.10,0.10,0.15)),nsamps=10000,warmup=500,alpha=0.05){

#Calculate Relevant Posterior Probabilities
post.probs <- npo.analysis(dataset=dataset,U=U,pi.star=pi.star,nsamps=nsamps,warmup=
```

```
    warmup)

#Get Cutoffs for Declaring Superiority/Inferiority at Each Analysis
cutoffs <- pnorm(gsDesign(alpha=alpha/2,k=2,test.type=2,sfu=sfPower,sfupar=3)$upper$bound
    )

#Output what subgroups are still active and whether this is an iterim or final analysis
if(analysis == "Interim"){ cat("This is an Interim Analysis. \n\n"); cut = cutoffs[1]}
if(analysis == "Final"){ cat("This is the Final Analysis. \n\n"); cut = cutoffs[2]}
if("P" %in% active.subgroups) cat("No Decision has previously been made for Primary
    Patients. \n")
if(!("P" %in% active.subgroups)) cat("A Decision was made at an earlier analysis for
    Primary Patients. \n")
if("S" %in% active.subgroups) cat("No Decision has previously been made for Salvage
    Patients. \n\n")
if(!("S" %in% active.subgroups)) cat("A Decision was made at an earlier analysis for
    Salvage Patients. \n\n")

#Output Posterior Probabilities Corresponding to Active Subgroup and the Cut-off
cat(paste("The Posterior Probability Cutoff for Declaring Superiority/Inferiority is",
    round(cut,4),"\n\n",sep=" "))
if("P" %in% active.subgroups){
cat(paste("The Posterior Probability that NuPrehab is Superior to Placebo for Primary
    Patients is",post.probs[1],"\n",sep=" "))
cat(paste("The Posterior Probability that NuPrehab is Inferior to Placebo for Primary
    Patients is",post.probs[3],"\n\n",sep=" "))
}
if("S" %in% active.subgroups){
cat(paste("The Posterior Probability that NuPrehab is Superior to Placebo for Salvage
    Patients is",post.probs[2],"\n",sep=" "))
cat(paste("The Posterior Probability that NuPrehab is Inferior to Placebo for Salvage
    Patients is",post.probs[4],"\n\n",sep=" "))
}
```

```
#Output Decision
cat("Therefore, \n")
#Interim Analysis Decisions
if(analysis == "Interim" & "P" %in% active.subgroups & post.probs[1] > cut) cat("Stop
    Enrolling Primary Patients: NuPrehab is Superior to Placebo for Primary Patients \n")
if(analysis == "Interim" & "P" %in% active.subgroups & post.probs[3] > cut) cat("Stop
    Enrolling Primary Patients: NuPrehab is Inferior to Placebo for Primary Patients \n")
if(analysis == "Interim" & "P" %in% active.subgroups & max(post.probs[c(1,3)]) <= cut)
    cat("Continue Enrolling Primary Patients \n")
if(analysis == "Interim" & "S" %in% active.subgroups & post.probs[2] > cut) cat("Stop
    Enrolling Salvage Patients: NuPrehab is Superior to Placebo for Salvage Patients \n")
if(analysis == "Interim" & "S" %in% active.subgroups & post.probs[4] > cut) cat("Stop
    Enrolling Salvage Patients: NuPrehab is Inferior to Placebo for Salvage Patients \n")
if(analysis == "Interim" & "S" %in% active.subgroups & max(post.probs[c(2,4)]) <= cut)
    cat("Continue Enrolling Salvage Patients \n")
#Final Analysis Decisions
if(analysis == "Final" & "P" %in% active.subgroups & post.probs[1] > cut) cat("NuPrehab
    is Superior to Placebo for Primary Patients \n")
if(analysis == "Final" & "P" %in% active.subgroups & post.probs[3] > cut) cat("NuPrehab
    is Inferior to Placebo for Primary Patients \n")
if(analysis == "Final" & "P" %in% active.subgroups & max(post.probs[c(1,3)]) <= cut) cat
    ("The Trial is Inconclusive for Primary Patients \n")
if(analysis == "Final" & "S" %in% active.subgroups & post.probs[2] > cut) cat("NuPrehab
    is Superior to Placebo for Salvage Patients \n")
if(analysis == "Final" & "S" %in% active.subgroups & post.probs[4] > cut) cat("NuPrehab
    is Inferior to Placebo for Salvage Patients \n")
if(analysis == "Final" & "S" %in% active.subgroups & max(post.probs[c(2,4)]) <= cut) cat
    ("The Trial is Inconclusive for Salvage Patients \n")
}
```

```
### Function to fit proportional odds cumulative logistic regression model in JAGS
library('R2jags')
po.model <- function(){
### Likelihood ###
# Cumulative Clavien-Dindo Probabilities, i.e., Prob(y <= s | Sgp, Trt), s=0,...,4
for(s in 1:5){
logit(Q[1,1,s]) <- alpha[s] - 0.5*beta[1] - 0.5*beta[2] + 0.25*beta[3] #Primary, Control
logit(Q[1,2,s]) <- alpha[s] - 0.5*beta[1] + 0.5*beta[2] - 0.25*beta[3] #Primary, NuPrehab
logit(Q[2,1,s]) <- alpha[s] + 0.5*beta[1] - 0.5*beta[2] - 0.25*beta[3] #Salvage, Control
logit(Q[2,2,s]) <- alpha[s] + 0.5*beta[1] + 0.5*beta[2] + 0.25*beta[3] #Salvage, NuPrehab
}
# Clavien-Dindo Probabilities, i.e., Prob(y = s | Sgp, Trt), s=0,...,5
for(j in 1:2){
for(k in 1:2){
pi[j,k,1] <- Q[j,k,1]
for(s in 2:5){
pi[j,k,s] <- Q[j,k,s]-Q[j,k,s-1]
}
pi[j,k,6] <- 1-Q[j,k,5]
}
}
# Likelihood Contributions
for(i in 1:n){
y.plus.one[i] ~ dcat(pi[sgp[i],trt[i],])
}

### Prior ###
alpha[1] ~ dt(alpha.star[1],0.4,5)
for(s in 2:5){
alpha[s] ~ dt(alpha.star[s],0.4,5)%_%T(alpha[s-1],)
}
for(m in 1:3){
beta[m] ~ dt(beta.star[m],0.4,5)
```

```
}

### Mean Utilities ###
for(j in 1:2){
for(k in 1:2){
u[j,k] <- inprod(U[1:6],pi[j,k,1:6])
}
}
}



### Function to fit PO model in JAGS and return the posterior probabilities for the
    decision rules
# Inputs:
#   dataset - data.frame with the three variables, (y, sgp, trt)
#   U - numerical utilities, default corresponds to the values for the IMPACT trial
#   pi.star - prior POM score probabilities, default corresponds to the values for the
    IMPACT trial
#   nsamps - number of posterior samples to generate, default is 10000
#   warmup - number of warmup samples, default is 500
# Outputs:
#   vector of the four posterior probabilities,
#     1 - posterior probability that nutritional prehabilitation is superior to the
    control for primary patients
#     2 - posterior probability that nutritional prehabilitation is superior to the
    control for salvage patients
#     3 - posterior probability that nutritional prehabilitation is inferior to the
    control for primary patients
#     4 - posterior probability that nutritional prehabilitation is inferior to the
    control for salvage patients
po.analysis <- function(dataset,U=c(100,80,65,25,10,0),pi.star=rbind(c
```

```
    (0.50,0.20,0.10,0.10,0.05,0.05),c(0.30,0.25,0.10,0.10,0.10,0.15)),nsamps=10000,warmup
    =500){
# Data preparation
n = nrow(dataset); y = dataset$y; sgp = dataset$sgp; trt = dataset$trt; y.plus.one = y+1
dat = list("n","y.plus.one","sgp","trt","alpha.star","beta.star","U"); params = c("u")

# Prior Specification
alpha.star <- colMeans(rbind(sapply(1:5,function(s) log(sum(pi.star[1,1:s])/(1-sum(pi.
    star[1,1:s])))),sapply(1:5,function(s) log(sum(pi.star[2,1:s])/(1-sum(pi.star[2,1:s])
    )))))
beta.star <- c(mean(apply(rbind(sapply(1:5,function(s) log(sum(pi.star[1,1:s])/(1-sum(pi.
    star[1,1:s])))),sapply(1:5,function(s) log(sum(pi.star[2,1:s])/(1-sum(pi.star[2,1:s])
    )))),2,function(x) diff(x))),0,0)

# Fit model with interaction
fit <- jags(dat,inits=NULL, parameters.to.save=params, model.file=po.model, n.iter=nsamps
    +warmup, n.burnin=warmup, n.chains=1, n.thin=1)

# Calculate Posterior Probabilities
pp.prim.sup = mean(fit$BUGSoutput$sims.list$u[,1,2] > fit$BUGSoutput$sims.list$u[,1,1])
pp.salv.sup = mean(fit$BUGSoutput$sims.list$u[,2,2] > fit$BUGSoutput$sims.list$u[,2,1])
pp.prim.inf = mean(fit$BUGSoutput$sims.list$u[,1,2] < fit$BUGSoutput$sims.list$u[,1,1])
pp.salv.inf = mean(fit$BUGSoutput$sims.list$u[,2,2] < fit$BUGSoutput$sims.list$u[,2,1])

return(round(c(pp.prim.sup,pp.salv.sup,pp.prim.inf,pp.salv.inf),4))
}




### Function to faciliate fitting the propotional odds logistic regression model in JAGS
    (no input or output)
library('R2jags')
```

```
trad.model <- function(){
### Likelihood ###
# Cumulative Clavien-Dindo Probabilities, i.e., Prob(y <= s | Sgp, Trt), s=0,...,4
for(s in 1:5){
logit(Q[1,1,s]) <- alpha[s] - 0.5*beta[1] - 0.5*beta[2] #Primary, Control
logit(Q[1,2,s]) <- alpha[s] - 0.5*beta[1] + 0.5*beta[2] #Primary, NuPrehab
logit(Q[2,1,s]) <- alpha[s] + 0.5*beta[1] - 0.5*beta[2] #Salvage, Control
logit(Q[2,2,s]) <- alpha[s] + 0.5*beta[1] + 0.5*beta[2] #Salvage, NuPrehab
}
# Clavien-Dindo Probabilities, i.e., Prob(y = s | Sgp, Trt), s=0,...,5
for(j in 1:2){
for(k in 1:2){
pi[j,k,1] <- Q[j,k,1]
for(s in 2:5){
pi[j,k,s] <- Q[j,k,s]-Q[j,k,s-1]
}
pi[j,k,6] <- 1-Q[j,k,5]
}
}
# Likelihood Contributions
for(i in 1:n){
y.plus.one[i] ~ dcat(pi[sgp[i],trt[i],])
}

### Prior ###
alpha[1] ~ dt(alpha.star[1],0.4,5)
for(s in 2:5){
alpha[s] ~ dt(alpha.star[s],0.4,5)%_%T(alpha[s-1],)
}
for(m in 1:2){
beta[m] ~ dt(beta.star[m],0.4,5)
}
}
```

```
### Function to fit traditional PO model in JAGS and return the posterior probabilities
    for the decision rules
# Inputs:
#   dataset - data.frame with the three variables, (y, sgp, trt)
#   pi.star - prior POM score probabilities, default corresponds to the values for the
    IMPACT trial
#   nsamps - number of posterior samples to generate, default is 10000
#   warmup - number of warmup samples, default is 500
# Outputs:
#   vector of the two posterior probabilities,
#      1 - posterior probability that nutritional prehabilitation is superior to the
    control
#      2 - posterior probability that nutritional prehabilitation is inferior to the
    control
trad.analysis <- function(dataset,pi.star=rbind(c(0.50,0.20,0.10,0.10,0.05,0.05),c
    (0.30,0.25,0.10,0.10,0.10,0.15)),nsamps=10000,warmup=500){
# Data preparation
n = nrow(dataset); y = dataset$y; sgp = dataset$sgp; trt = dataset$trt; y.plus.one = y+1
dat = list("n","y.plus.one","sgp","trt","alpha.star","beta.star"); params = c("beta")

# Prior Specification
alpha.star <- colMeans(rbind(sapply(1:5,function(s) log(sum(pi.star[1,1:s])/(1-sum(pi.
    star[1,1:s])))),sapply(1:5,function(s) log(sum(pi.star[2,1:s])/(1-sum(pi.star[2,1:s])
    )))))
beta.star <- c(mean(apply(rbind(sapply(1:5,function(s) log(sum(pi.star[1,1:s])/(1-sum(pi.
    star[1,1:s])))),sapply(1:5,function(s) log(sum(pi.star[2,1:s])/(1-sum(pi.star[2,1:s])
    )))),2,function(x) diff(x))),0)

# Fit model with interaction
```

```
fit <- jags(dat,inits=NULL, parameters.to.save=params, model.file=trad.model, n.iter=
    nsamps+warmup, n.burnin=warmup, n.chains=1, n.thin=1)

# Calculate Posterior Probabilities
pp.sup = mean(fit$BUGSoutput$sims.list$beta[,2] > 0)
pp.inf = mean(fit$BUGSoutput$sims.list$beta[,2] < 0)

return(round(c(pp.sup,pp.inf),4))
}
```

## 7.3 simulation.R

```
### This file is for replicating the simulation study reported in Murray et al. (2017), "
    A Bayesian Utility-Based Stratified Medicine Design for the Effectiveness of
    Nutritional Prehabilitation in Thoracic Surgery"
### Written by Thomas A Murray on 14 June 2017
### E-mail: 8tmurray@gmail.com

# Set working directory
setwd("C:/Users/")

# Load functions
source('functions.R')

# Analysis Sample Sizes, probability of each subject being a primary patient
ns = c(50,100); prob.primary = 0.6

# Elicited Utilities
U = c(100,80,65,25,10,0)

# Prior mean probabilities in the control arm of the two patient subgroups
pi.star = rbind(c(0.50,0.20,0.10,0.10,0.05,0.05), #Primary Patients
c(0.30,0.25,0.10,0.10,0.10,0.15)) #Salvage Patients
```

```
# MCMC Specifications
nsamps=10000; warmup=500

# Probability Thresholds
# for parallel you need to supply the cutoffs by hand since the server doesn't have
    gsDesign package...or you can request to have this added. To do this, just calculate
    the cutoffs on your own machine. right now these are 0.997 and 0.976 at the interim
    and final analyses
alpha = 0.05; cutoffs <- round(pnorm(gsDesign(alpha=alpha/2,k=2,test.type=2,sfu=sfPower,
    sfupar=3)$upper$bound),3)

# Number of trial simulations per scenario
niters = 2 #We use 5000 iterations in the paper. We recommend parallelizing the code in
    the sequel.

# Number of scenarios
scens = 1:7

# Run Simulation
for(scen in scens){
# True POM Score Probabilities in each scenario
pis = array(NA,c(2,2,6)); #Cumulative POM Score Probabilities <0 (= 0.00), <=0, <=1, ...,
     <=4, <=5 (= 1.00)
pis[1,1,] = diff(c(0.00,0.50,0.70,0.80,0.90,0.95,1.00)) #Primary, Control Arm
pis[2,1,] = diff(c(0.00,0.30,0.55,0.65,0.75,0.85,1.00)) #Salvage, Control Arm

if(scen == 1){
pis[1,2,] = diff(c(0.00,0.50,0.70,0.80,0.90,0.95,1.00)) #Primary, Nuprehab Arm
pis[2,2,] = diff(c(0.00,0.30,0.55,0.65,0.75,0.85,1.00)) #Salvage, Nuprehab Arm
}
if(scen == 2){
pis[1,2,] = diff(c(0.00,0.50,0.70,0.80,0.90,0.95,1.00)) #Primary, Nuprehab Arm
```

```
pis[2,2,] = diff(c(0.00,0.71,0.87,0.91,0.94,0.97,1.00)) #Salvage, Nuprehab Arm
}
if(scen == 3){
pis[1,2,] = diff(c(0.00,0.83,0.92,0.95,0.98,0.99,1.00)) #Primary, Nuprehab Arm
pis[2,2,] = diff(c(0.00,0.30,0.55,0.65,0.75,0.85,1.00)) #Salvage, Nuprehab Arm
}
if(scen == 4){
pis[1,2,] = diff(c(0.00,0.83,0.92,0.95,0.98,0.99,1.00)) #Primary, Nuprehab Arm
pis[2,2,] = diff(c(0.00,0.71,0.87,0.91,0.94,0.97,1.00)) #Salvage, Nuprehab Arm
}
if(scen == 5){
pis[1,2,] = diff(c(0.00,0.50,0.70,0.80,0.90,0.95,1.00)) #Primary, Nuprehab Arm
pis[2,2,] = diff(c(0.00,0.53,0.80,0.91,0.94,0.97,1.00)) #Salvage, Nuprehab Arm
}
if(scen == 6){
pis[1,2,] = diff(c(0.00,0.67,0.85,0.95,0.98,0.99,1.00)) #Primary, Nuprehab Arm
pis[2,2,] = diff(c(0.00,0.30,0.55,0.65,0.75,0.85,1.00)) #Salvage, Nuprehab Arm
}
if(scen == 7){
pis[1,2,] = diff(c(0.00,0.67,0.85,0.95,0.98,0.99,1.00)) #Primary, Nuprehab Arm
pis[2,2,] = diff(c(0.00,0.53,0.80,0.91,0.94,0.97,1.00)) #Salvage, Nuprehab Arm
}

# True log-odds ratios compartion Nuprehab to Control at each POM score level in Primary
    then Salvage
#log(cumsum(pis[1,2,1:5])/(1-cumsum(pis[1,2,1:5])))-log(cumsum(pis[1,1,1:5])/(1-cumsum(
    pis[1,1,1:5])))
#log(cumsum(pis[2,2,1:5])/(1-cumsum(pis[2,2,1:5])))-log(cumsum(pis[2,1,1:5])/(1-cumsum(
    pis[2,1,1:5])))

set.seed(1985)
for(iter in 1:niters){
### Create Storage Objects
```

```
pps.npo = pps.po = matrix(NA,nrow=length(ns),ncol=4)
pps.trad = matrix(NA,nrow=length(ns),ncol=2)

### Generate a Dataset
dataset = get.dataset(n=max(ns),pis=pis,prob.primary=prob.primary)

### Run each analysis function at the interim analysis
pps.npo[1,] = npo.analysis(U=U,dataset=dataset[1:ns[1],],pi.star=pi.star,nsamps=nsamps,
    warmup=warmup)
pps.po[1,]  = po.analysis(U=U,dataset=dataset[1:ns[1],],pi.star=pi.star,nsamps=nsamps,
    warmup=warmup)
pps.trad[1,] = trad.analysis(dataset=dataset[1:ns[1],],pi.star=pi.star,nsamps=nsamps,
    warmup=warmup)

### Run each analysis function at the final analysis based on the interim decision
# NPO Model-based Stratified Medicine Design
if((pps.npo[1,1] > cutoffs[1] | pps.npo[1,3] > cutoffs[1]) & (pps.npo[1,2] > cutoffs[1] |
     pps.npo[1,4] > cutoffs[1])) {
pps.npo[2,] = NA
}
if((pps.npo[1,1] > cutoffs[1] | pps.npo[1,3] > cutoffs[1]) & (pps.npo[1,2] <= cutoffs[1]
    & pps.npo[1,4] <= cutoffs[1])) {
pps.npo[2,] = npo.analysis(U=U,dataset=rbind(dataset[1:ns[1],],subset(dataset[(ns[1]+1):
    ns[2],],sgp==2)),pi.star=pi.star,nsamps=nsamps,warmup=warmup)
pps.npo[2,c(1,3)] = NA
}
if((pps.npo[1,1] <= cutoffs[1] & pps.npo[1,3] <= cutoffs[1]) & (pps.npo[1,2] > cutoffs[1]
     | pps.npo[1,4] > cutoffs[1])) {
pps.npo[2,] = npo.analysis(U=U,dataset=rbind(dataset[1:ns[1],],subset(dataset[(ns[1]+1):
    ns[2],],sgp==1)),pi.star=pi.star,nsamps=nsamps,warmup=warmup)
pps.npo[2,c(2,4)] = NA
}
if((pps.npo[1,1] <= cutoffs[1] & pps.npo[1,3] <= cutoffs[1]) & (pps.npo[1,2] <= cutoffs
```

```
    [1] & pps.npo[1,4] <= cutoffs[1])) {
pps.npo[2,] = npo.analysis(U=U,dataset=dataset,pi.star=pi.star,nsamps=nsamps,warmup=
    warmup)
}

# PO Model-based Stratified Medicine Design
if((pps.po[1,1] > cutoffs[1] | pps.po[1,3] > cutoffs[1]) & (pps.po[1,2] > cutoffs[1] |
    pps.po[1,4] > cutoffs[1])) {
pps.po[2,] = NA
}
if((pps.po[1,1] > cutoffs[1] | pps.po[1,3] > cutoffs[1]) & (pps.po[1,2] <= cutoffs[1] &
    pps.po[1,4] <= cutoffs[1])) {
pps.po[2,] = po.analysis(U=U,dataset=rbind(dataset[1:ns[1],],subset(dataset[(ns[1]+1):ns
    [2],],sgp==2)),pi.star=pi.star,nsamps=nsamps,warmup=warmup)
pps.po[2,c(1,3)] = NA
}
if((pps.po[1,1] <= cutoffs[1] & pps.po[1,3] <= cutoffs[1]) & (pps.po[1,2] > cutoffs[1] |
    pps.po[1,4] > cutoffs[1])) {
pps.po[2,] = po.analysis(U=U,dataset=rbind(dataset[1:ns[1],],subset(dataset[(ns[1]+1):ns
    [2],],sgp==1)),pi.star=pi.star,nsamps=nsamps,warmup=warmup)
pps.po[2,c(2,4)] = NA
}
if((pps.po[1,1] <= cutoffs[1] & pps.po[1,3] <= cutoffs[1]) & (pps.po[1,2] <= cutoffs[1] &
     pps.po[1,4] <= cutoffs[1])) {
pps.po[2,] = po.analysis(U=U,dataset=dataset,pi.star=pi.star,nsamps=nsamps,warmup=warmup)
}

# PO Model-based Traditional Design
if(pps.trad[1,1] > cutoffs[1] | pps.trad[1,2] > cutoffs[1]){
pps.trad[2,] = NA
}
if(pps.trad[1,1] <= cutoffs[1] & pps.trad[1,2] <= cutoffs[1]){
pps.trad[2,] = trad.analysis(dataset=dataset,pi.star=pi.star,nsamps=nsamps,warmup=warmup)
```

```
}

### Write posterior probabilities to .txt files
write.table(rbind(as.vector(pps.npo)),file=paste("NPO-Res-Scen",scen,".txt",sep=""),sep
    =",",append=TRUE,row.names=FALSE,col.names=FALSE)
write.table(rbind(as.vector(pps.po)),file=paste("PO-Res-Scen",scen,".txt",sep=""),sep
    =",",append=TRUE,row.names=FALSE,col.names=FALSE)
write.table(rbind(as.vector(pps.trad)),file=paste("Trad-Res-Scen",scen,".txt",sep=""),sep
    =",",append=TRUE,row.names=FALSE,col.names=FALSE)
}
}




### Results
library('gsDesign')
#For alpha = 0.05 type I error, the monitoring Boundaries at interim and final analyses
    are
alpha = 0.05; cutoffs <- round(pnorm(gsDesign(alpha=alpha/2,k=2,test.type=2,sfu=sfPower,
    sfupar=3)$upper$bound),3)

### Load NPO Results
res = read.csv("NPO-Res-Scen1.txt",header=FALSE)

# Calculate Power Figures
mean(res[,1] > cutoffs[1] | (res[,1] <= cutoffs[1] & res[,5] <= cutoffs[1] & res[,2] >
    cutoffs[2])) #N superior for P patients
mean(res[,5] > cutoffs[1] | (res[,1] <= cutoffs[1] & res[,5] <= cutoffs[1] & res[,6] >
    cutoffs[2])) #N inferior for P patients
mean(res[,3] > cutoffs[1] | (res[,3] <= cutoffs[1] & res[,7] <= cutoffs[1] & res[,4] >
    cutoffs[2])) #N superior for S patients
mean(res[,7] > cutoffs[1] | (res[,3] <= cutoffs[1] & res[,7] <= cutoffs[1] & res[,8] >
```

```
    cutoffs[2])) #N inferior for S patients

# Calculate Expected Sample Size
50*mean((res[,1] > cutoffs[1] | res[,5] > cutoffs[1]) & (res[,3] > cutoffs[1] | res[,7] >
     cutoffs[1])) +
70*mean((res[,1] > cutoffs[1] | res[,5] > cutoffs[1]) & (res[,3] <= cutoffs[1] & res[,7]
    <= cutoffs[1])) +
80*mean((res[,1] <= cutoffs[1] & res[,5] <= cutoffs[1]) & (res[,3] > cutoffs[1] | res[,7]
     > cutoffs[1])) +
100*mean((res[,1] <= cutoffs[1] & res[,5] <= cutoffs[1]) & (res[,3] <= cutoffs[1] & res
    [,7] <= cutoffs[1]))



### Load PO Results
res = read.csv("PO-Res-Scen1.txt",header=FALSE)

# Calculate Power Figures
mean(res[,1] > cutoffs[1] | (res[,1] <= cutoffs[1] & res[,5] <= cutoffs[1] & res[,2] >
    cutoffs[2])) #N superior for P patients
mean(res[,5] > cutoffs[1] | (res[,1] <= cutoffs[1] & res[,5] <= cutoffs[1] & res[,6] >
    cutoffs[2])) #N inferior for P patients
mean(res[,3] > cutoffs[1] | (res[,3] <= cutoffs[1] & res[,7] <= cutoffs[1] & res[,4] >
    cutoffs[2])) #N superior for S patients
mean(res[,7] > cutoffs[1] | (res[,3] <= cutoffs[1] & res[,7] <= cutoffs[1] & res[,8] >
    cutoffs[2])) #N inferior for S patients

# Calculate Expected Sample Size
50*mean((res[,1] > cutoffs[1] | res[,5] > cutoffs[1]) & (res[,3] > cutoffs[1] | res[,7] >
     cutoffs[1])) +
70*mean((res[,1] > cutoffs[1] | res[,5] > cutoffs[1]) & (res[,3] <= cutoffs[1] & res[,7]
    <= cutoffs[1])) +
80*mean((res[,1] <= cutoffs[1] & res[,5] <= cutoffs[1]) & (res[,3] > cutoffs[1] | res[,7]
     > cutoffs[1])) +
```

```
100*mean((res[,1] <= cutoffs[1] & res[,5] <= cutoffs[1]) & (res[,3] <= cutoffs[1] & res
    [,7] <= cutoffs[1]))



### Load Trad Results
res = read.csv("Trad-Res-Scen1.txt",header=FALSE)

# Calculate Power Figures
mean(res[,1] > cutoffs[1] | (res[,1] <= cutoffs[1] & res[,3] <= cutoffs[1] & res[,2] >
    cutoffs[2])) #N superior
mean(res[,3] > cutoffs[1] | (res[,1] <= cutoffs[1] & res[,3] <= cutoffs[1] & res[,4] >
    cutoffs[2])) #N inferior

# Calculate Expected Sample Size
50*mean((res[,1] > cutoffs[1] | res[,3] > cutoffs[1])) + 100*mean(res[,1] <= cutoffs[1] &
     res[,3] <= cutoffs[1])
```