Appendix

Appendix 1: Simulate individual patient data for a non-small-cell lung cancer trial

Denote $\log(T) \sim N(u_T, \sigma_T^2)$, $PTR_{wk8} \sim N(u_{PTR}, \sigma_{PTR}^2)$, the following equations need to be satisfied so that PTR_{wk8} and $\log(T)$ correlate with a correlation coefficient ρ while preserving (1) and (2).

For treatment group,

$$u_T = 5.57 + 0.42 * u_{PTR}$$

$$\sigma_{PTR} = \rho \frac{\sigma_T}{0.42}$$

$$\sigma_{TD}^2 = (1 - \rho^2)\sigma_T^2$$

For placebo group,

$$u_T = 5.42 + 0.38 * u_{PTR}$$

$$\sigma_{PTR} = \rho \frac{\sigma_T}{0.38}$$

$$\sigma_{TD}^2 = (1 - \rho^2)\sigma_T^2$$

To simulate the data, we make assumptions for the primary study endpoint on u_T and follow the below procedures:

(1) Calculate u_{PTR} ,σ_T^2 and σ_{PTR}^2using the above equations

(2) Simulate PTR_{wk8} from $N(u_{PTR}, \sigma_{PTR}^2)$ and $\varepsilon_{TD}$from$N(0, \sigma_{TD}^2)$

(3) Calculate $\log(T)$ using (1) for treatment group and (2) for placebo group.

For example, in our simulation setting with ρ= 0.1, we simulate PTR_{wk8} from $N(-0.6, 0.047)$ and $\varepsilon_{TD}$from $N(0, 0.68)$ and $\log(T) = 5.42 + 0.38 \times PTR_{wk8} + \varepsilon_{TD}$ for placebo data; we simulate PTR_{wk8} from $N(-0.21, 0.041)$ and $\varepsilon_{TD}$from $N(0, 0.72)$ and $\log(T) = 5.57 + 0.42 \times PTR_{wk8} + \varepsilon_{TD}$ for treatment group 1 data; we simulate PTR_{wk8} from $N(0.318, 0.041)$ and $\varepsilon_{TD}$from $N(0, 0.72)$ and $\log(T) = 5.57 + 0.42 \times PTR_{wk8} + \varepsilon_{TD}$ for treatment group 2 data; we simulate PTR_{wk8} from $N(0.753, 0.041)$ and $\varepsilon_{TD}$from $N(0, 0.72)$ and $\log(T) = 5.57 + 0.42 \times PTR_{wk8} + \varepsilon_{TD}$ for treatment group 3 data.

Appendix 2: R code for power simulation of biomarker-informed two-stage winner design with hierarchical model

```
library(mvtnorm)
powerBInfo=function(uCtl, u0y, u0x, rhou, suy, sux, rho, sy, sx, Zalpha, N1, N, nArms, nSims){
uy=rep(0,nArms); ux=rep(0,nArms); probWinners=rep(0,nArms);power=0;
```

```
varcov0=matrix(c(suy^2,rhou*suy*sux, rhou*suy*sux, sux^2),2,2)
varcov=matrix(c(sy^2,rho*sy*sx, rho*sx*sy, sx^2),2,2)
for (i in 1:nSims){
winnerMarker=-Inf
for (j in 1:nArms) {
u=rmvnorm(1, mean=c(u0y[j],u0x[j]), sigma=varcov0)
uy[j]=u[1]; ux[j]=u[2];
dataStg1=rmvnorm(N1, mean=c(uy[j], ux[j]), sigma=varcov);
meanxMarker=mean(dataStg1[,2]);
if (meanxMarker>winnerMarker)
{winner=j; winnerMarker=meanxMarker; winnerY=dataStg1[,1]}
}
for (j in 1:nArms) {if (winner==j) {probWinners[j]=probWinners[j]+1/nSims}}
yStg1=winnerY
yStg2=rnorm(N-N1, mean=uy[winner], sd=sy)
yTrt=c(yStg1+yStg2)
yCtl=rnorm(N, mean=uCtl, sd=sy)
tValue=t.test(yTrt, yCtl)$statistic
if (tValue>=Zalpha) {power=power+1/nSims}
}
return (c(power, probWinners))
}
```

Codes to invoke the R-function are presented in the following:

```
>> R: Invoke R-Function>>
## determine critical value Zalpha for alpha (power)=0.025###
> u0y=c(0,0,0)
> u0x=c(0,0,0)
>
```

>powerBInfo(uCtl=0, u0y, u0x, rhou=1, suy=0, sux=0, rho=1, sy=2, sx=2, Zalpha=2.4, N1=43, N=86, nArms=3, nSims=10000)

##Power simulation##

> u0y=c(5.48,5.7,5.88)

> u0x=c(-0.21,0.32,0.75)

>

>powerBInfo(uCtl=5.42, u0y, u0x, rhou=0.1, suy=4, sux=4, rho=0.2, sy=2, sx=2, Zalpha=2.4, N1=43, N=86, nArms=3, nSims=10000)

<< R<<

Appendix 3: R code for estimation of parameters

```
###################### Constants ######################
# size: size of historical data in treatment group j #
# iteration: number of MCMC iterations #
# burn: number of burn-in iterations #
########################################################


Library (MCMCpack)
library(MASS)

size=500
iteration=4000
burn=2000



###################### simulate trial data ############
# trial with 3 active treatments and 1 control arm #
########################################################


bmkr0=rnorm(size,-0.6, 0.047)
e0=rnorm(size, 0, 0.68)
prmy0=5.42+0.38*bmkr0+e0


bmkr1=rnorm(size,-0.21, 0.041)
e1=rnorm(size, 0, 0.72)
prmy1=5.57+0.42*bmkr1+e1


bmkr2=rnorm(size,0.318, 0.041)
```

```
e2=rnorm(size, 0, 0.72)
prmy2=5.57+0.42*bmkr2+e2

bmkr3=rnorm(size,0.753, 0.041)
e3=rnorm(size, 0, 0.72)
prmy3=5.57+0.42*bmkr3+e3

bmkr=c(bmkr0, bmkr1, bmkr2, bmkr3)
prmy=c(prmy0, prmy1, prmy2, prmy3)



################### Initialization ####################



sigma_ind=matrix(c(var(prmy), cov(prmy, bmkr), cov(prmy, bmkr), var(bmkr)), 2, 2)

u_ind=matrix(c(0,0), 2, 1)
u_m=matrix(c(0,0), 2, 1)
sigma_m=matrix(c(1,0,0, 1), 2, 2)

u0=matrix(c(0,0), 2, 1)
k0=0.001
A0=matrix(c(1,0,0, 1), 2, 2)
v0=3

u_indest=matrix(0,nrow=2,ncol=iteration)
u_mest=matrix(0, nrow=2, ncol=iteration)
sigma_m11est=c()
sigma_m12est=c()
sigma_m22est=c()



#########updating u_m, sigma_m, u_ind###########


for (i in 1:iteration){


un=(k0*u0+u_ind)/(k0+1)
kn=k0+1
vn=v0+1
An=A0+(k0/(k0+1))*(u_ind-u0) %*% t(u_ind-u0)

sigma_m=riwish(vn, An)
u_m=as.matrix(mvrnorm(n=1,un, sigma_m/kn), 2, 1)


sigmaN=solve(solve(sigma_m)+size*solve(sigma_ind))
uN=sigmaN %*% (size*solve(sigma_ind) %*% matrix(c(mean(prmy0), mean(bmkr0)), 2, 1) +
solve(sigma_m) %*% u_m)

u_ind=as.matrix(mvrnorm(n=1,uN, sigmaN),2,1)
```

```
u_indest[,i]=u_ind

u_mest[,i]=u_m

sigma_m11est=c(sigma_m11est, sigma_m[1,1])

sigma_m12est=c(sigma_m12est, sigma_m[1,2])

sigma_m22est=c(sigma_m22est, sigma_m[2,2])

i=i+1

}



################estimator of parameter#######################
sigma_ind_est=sigma_ind
u_ind_est=rowMeans(u_indest[, burn:iteration])
u_m_est=rowMeans(u_mest[, burn:iteration])
sigma_m11_est=mean(sigma_m11est[burn:iteration])
sigma_m12_est=mean(sigma_m12est[burn:iteration])
sigma_m22_est=mean(sigma_m22est[burn:iteration])
rho_est=sigma_m12_est/sqrt(sigma_m11_est*sigma_m22_est)
```