

747 Additional file 6(a) — R code for simulation

748 All simulations were performed using R[44], Stan[33], and NYU high-performance computing (HPC) clusters[45].
 749 The code here shows how we simulated the data as in Section [Simulation setup - basic model](#), fitted the final *co*
 750 model, implemented HMC algorithms for Bayesian inference, and computed the posterior medians of the key
 751 parameter of interest Δ_{co} . The code for the other simulations is available upon request from the authors.

```

752 1 library(simstudy)
753 2 library(cmdstanr)
754 3 library(data.table)
755 4 library(slurmR)
756 5 library(posterior)
757 6
758 7 mco <- cmdstan_model("./primary_co.stan")
759 8
760 9 #--- generating the study-specific baseline probabilities for each
761     outcome level---#
762 0
763 1 genBaseProbs <- function(n, base, similarity, digits = 2) {
764 2
765 3   n_levels <- length(base)
766 4   x <- gtools::rdirichlet(n, similarity * base)
767 5
768 6   #--- ensure that each vector of probabilities sums exactly to 1---#
769 7
770 8   x <- round(floor(x*1e8)/1e8, digits) # round the generated
771     probabilities
772 9   xpart <- x[, 1:(n_levels-1)]         # delete the base prob of the
773     final level
774 0   partsum <- apply(xpart, 1, sum)       # add the values of levels 1 to
775     K-1
776 1   x[, n_levels] <- 1 - partsum         # the base prob of the level K
777     = 1 - sum(1:[K-1])
778 2
779 3   return(x)
780 4 }
781 5
782 6 nsites <- 9                            # assume 9 RCTs in total
783 7
784 8 basestudy <- genBaseProbs(n = nsites,
785 9                           base = c(0.100, 0.107, 0.095, 0.085, 0.090,
786     0.090, 0.108, 0.100, 0.090, 0.075, 0.060), similarity = 100)
787 0
788 1 # More detailed information: https://www.rdatagen.net/post/generating-
789     probabilities-for-ordinal-categorical-data/
790 2
791 3 #--- data definition code ---#
792 4
793 5 defC <- defDataAdd(varname = "b", formula = 0, variance= .01,
794 6                   dist = "normal")      # RCT specific intercept
795 7 defC <- defDataAdd(defC, varname = "size", formula = "75+75*large",

```

```

796:8         dist = "nonrandom") # sample size; large=1: 150
797         patients, large=0: 75 patients
798:9 defC2 <- defDataAdd(varname="C_rv",formula="C * control",
799:0         dist = "nonrandom") # 0=CP, 1= standard of care,
800         2=non-CP, 3=saline
801:1 defC2 <- defDataAdd(defC2,varname="sex", formula = 0.5,
802:2         dist = "binary")
803:3 defC2 <- defDataAdd(defC2,varname="who_enroll", formula = "1/3;1/3;1/3
804         ",
805:4         dist = "categorical")
806:5 defC2 <- defDataAdd(defC2,varname="age", formula = "0.25;0.25;0.50",
807:6         dist = "categorical")
808:7 defC2 <- defDataAdd(defC2, varname = "ss",
809:8         formula = "0.2;0.2;0.2;0.2;0.2", dist = "
810         categorical")
811:9 defC2 <- defDataAdd(defC2, varname = "z",
812:0         formula = "0.05*(ss-1) + 0.1*sex +
813:1         0.075*(age-1) + 0.06*(who_enroll-1) +
814:2         (0.3 + b ) * (C_rv==1) +
815:3         (0.4 + b ) * (C_rv==2) + (0.5 + b ) * (C_rv==3)",
816:4         dist = "nonrandom")
817:5
818:6 iter <- function(iternum, defC, defC2, basestudy, nsites,mco) {
819:7
820:8     set_cmdstan_path(path = "/gpfs/share/apps/cmdstan/2.25.0")
821:9
822:0     #--- data generation ---#
823:1
824:2     dstudy <- genData(nsites, id = "study") # 9 RCTs
825:3     dstudy <- trtAssign(dstudy, nTrt = 3, grpName = "C") # allocate 3
826         control conditions
827:4     dstudy <- trtAssign(dstudy, nTrt = 2, strata = "C", grpName = "large
828         ", ratio = c(2,1))
829:5     dstudy <- addColumns(defC, dstudy)
830:6
831:7     dind <- genCluster(dstudy, "study", numIndsVar = "size", "id")
832:8     dind <- trtAssign(dind, strata="study", grpName = "control")
833:9     dind <- addColumns(defC2,dind)
834:0
835:1     setkey(dind, "id")
836:2
837:3     dl <- lapply(1:nsites, function(i) {
838:4         b <- basestudy[i,]
839:5         dx <- dind[study == i]
840:6         dx <- genOrdCat(dx, adjVar = "z", b, catVar = "ordY")
841:7         dx[, ordY := factor(ordY, levels = c(1:11))]
842:8         dx[]
843:9     })
844:0
845:1     dind <- rbindlist(dl)
846:2

```

```

8473 #--- model estimation ---#
8484
8495 N = nrow(dind) # number of
850 observations
8516 L <- dind[, length(unique(ordY))] # number of levels of
852 outcome
8537 K <- dind[, length(unique(study))] # number of RCTs
8548 y <- as.numeric(dind$ordY) # individual outcome
8559 kk <- dind$study # RCT for each
856 individual
8570 ctrl <- dind$control # treatment arm for
858 individual
8591 cc <- dind[, .N, keyby = .(study, C)]$C # specific control arm
860 for RCT
8612 x <- model.matrix(ordY ~ factor(who_enroll) + factor(age) + factor(
862 sex) + factor(ss), data = dind)[, -1]
8633 D <- ncol(x)
8644 prior_div <- 8
8655 prior_Delta_sd <- .354
8666 eta <- 0.1
8677 prior_eta_0 <- 0.25
8688 prior_beta_sd <- 2.5
8699
8700 studydata <- list(
8711 N=N, L= L,K=K, y=y, ctrl=ctrl,cc=cc,kk=kk,prior_div=prior_div,
8722 prior_Delta_sd=prior_Delta_sd,eta=eta,
8733 prior_eta_0 = prior_eta_0,x=x,D=D,prior_beta_sd=prior_beta_sd)
8744
8755 fit_co <- mco$sample(
8766 step_size = 0.1,
8777 data = studydata,
8788 chains = 4L,
8799 parallel_chains = 4L,
8800 refresh = 500,
8811 iter_warmup = 500,
8822 iter_sampling = 2500,
8833 adapt_delta=0.8
8844 )
8855
8866 #--- estimate extraction ---#
8877
8888 diagnostics_df <- as_draws_df(fit_co$sampler_diagnostics())
8899 div_num_co <- sum(diagnostics_df[, 'divergent_'])
8900
8911 res_co <- data.table(fit_co$summary(variables = "negDelta"))[, .(
892 median)]
8932 res_co$div_co <- div_num_co
8943 data.table(iternum,res_co)
8954 }
8965
8976 #--- Replication ---#

```

```

898:7 job <- Slurm_lapply(1:2520,
898:8   iter,
900:9   defC=defC,
901:0   defC2 = defC2,
902:1   basestudy=basestudy,
904:3   nsites=nsites,
905:4   mco = mco,
906:5   njobs = 90,
907:6   mc.cores = 4,
908:7   tmp_path = "/gpfs/scratch/...",
909:8   job_name = "sim_1",
910:9   sbatch_opt = list(time = "24:00:00"),
911:0   plan = "wait",
912:1   overwrite=TRUE)
913:2
914:3 site_plasma_all <- Slurm_collect(job)
915:4 site_plasma <- rbindlist(site_plasma)
916:5
917:6 date_stamp <- gsub("-", "", Sys.Date())
918:7 dir.create(file.path("/gpfs/home/.../r/", date_stamp), showWarnings =
919:   FALSE)
920:8 save(site_plasma, file = paste0("/gpfs/home/.../r/", date_stamp, "/"
921:   model_co.rda"))

```

922 Additional file 6(b) — Stan code for the final version of the *co* model

```

923:1 data {
924:2   int<lower=0> N;           // number of observations
925:3   int<lower=2> L;           // number of WHO categories
926:4   int<lower=1> K;           // number of RCTs
927:5   int<lower=1,upper=L> y[N]; // vector of categorical outcomes
928:6   int<lower=1,upper=K> kk[N]; // RCT for individual
929:7   int<lower=0,upper=1> ctrl[N]; // treatment or control
930:8   int<lower=1,upper=3> cc[K]; // specific control for RCT
931:9   int<lower=1> D;           // number of covariates
932:0   row_vector[D] x[N];      // strata indicators N x D matrix
933:1
934:2   real<lower=0> prior_div;   // prior sd of tau
935:3   real<lower=0> prior_Delta_sd; // prior sd of overall control effect
936:4   real<lower=0> eta;         // prior sd of delta
937:5   real<lower=0> prior_beta_sd; // prior sd of beta
938:6   real<lower=0> prior_eta_0; //prior sd of eta_0 (the sd of
939:   delta_k)
940:7 }
941:8
942:9 parameters {
943:0
944:1   real alpha;               // overall intercept for treatment
945:2   ordered[L-1] tau[K];      // cut-points for cumulative odds
946:   matrix[K X [L-1] matrix)

```

```

9423   real<lower=0> eta_0;           // sd of delta_k (around delta)
94824
94925
95026   // non-central parameterization
95127
95228   vector[K] z_ran_rx;
95329   vector[3] z_delta;
95430   vector[D] z_beta;
95531   real z_Delta;
95632 }
95733
95834 transformed parameters{
95935
96036   vector[3] delta;           // control-specific effect
96137   vector[K] delta_k;         // RCT-specific treatment effect
96238   vector[D] beta;           // covariate estimates
96339   real Delta;               // overall control effect
96440   vector[N] yhat;
96541
96642   Delta = prior_Delta_sd * z_Delta;
96743   delta = eta * z_delta + Delta;
96844   beta = prior_beta_sd * z_beta;
96945
97046   for (k in 1:K)
97147     delta_k[k] = eta_0 * z_ran_rx[k] + delta[cc[k]];
97248
97349   for (i in 1:N)
97450     yhat[i] = alpha + ctrl[i] * delta_k[kk[i]] + x[i] * beta;
97551 }
97652
97753 model {
97854
97955   // priors
98056
98157   alpha ~ normal(0,0.1);
98258   z_ran_rx ~ std_normal();
98359   z_delta ~ std_normal();
98460   z_beta ~ std_normal();
98561   z_Delta ~ std_normal();
98662
98763   eta_0 ~ student_t(3,0,prior_eta_0);
98864
98965   for (k in 1:K)
99066     for (l in 1:(L-1))
99167       tau[k, l] ~ student_t(3, 0, prior_div);
99268
99369   // outcome model
99470
99571   for (i in 1:N)
99672     y[i] ~ ordered_logistic(yhat[i], tau[kk[i]]);
99773 }

```

```
998 4
999 5 generated quantities {
1000 6
1001 7     real OR;                // overall CCP effect (odds ratio)
1002 8     real negDelta;         // overall CCP effect on log scale
1003 9
1004 0     OR = exp(-Delta);
1005 1     negDelta=-1*Delta;
1006 2
1007 3 }
```