Estimate Correlation

Kristen Hunter

2/12/2022

Correlation estimation strategy

- For $s \in 1, \ldots, S$:
  - Input DGP parameters, including assumed correlation ρ.
  - Generate full simulated data.
  - Generate treatment assignments and observed data.
  - Calculate t statistics from regressions for each outcome. So for M outcomes, we have a M-vector of test statistics.
- Returns a matrix of test statistics $S \times M$.

Repetition:

- Calculate correlation of matrix.
- Take upper triangle of matrix and take average.
- Gives an estimated correlation.

The following function returns the $S \times M$ matrix of test statistics:

```
get_rawt <- function(d_m, model.params.list, Tbar, n.sims = 100)
{
  M <- model.params.list$M
  rawt.all <- matrix(NA, nrow = n.sims, ncol = M)
  dgp.params.list <- PUMP::convert_params(model.params.list)

  # number of simulations
  for(s in 1:n.sims)
  {
    if (s %% 20 == 0){ message(paste0("Now processing simulation ", s, " of ", n.sims)) }

    # generate simulated data
    samp.full <- PUMP::gen_full_data(dgp.params.list)
    S.id <- samp.full$ID$S.id
    D.id  <- samp.full$ID$D.id

    # generate treatment assignments
    T.x <- PUMP::gen_T.x(d_m = d_m,
                         S.id = S.id, D.id = D.id,
                         nbar = dgp.params.list$nbar,
                         Tbar = 0.5)

    # convert full data to observed data
    samp.obs <- samp.full
    samp.obs$Yobs <- PUMP::gen_Yobs(samp.full, T.x)
```

```
    # calculate t statistics
    dat.all <- makelist_samp(samp.obs, T.x)
    rawpt.out <- get_rawpt(dat.all, d_m = d_m, model.params.list = model.params.list)
    rawt <- sapply(rawpt.out[['rawpt']], function(s){ return(s[['tstat']])})
    rawt.all[s,] <- rawt
  }

  return(rawt.all)
}
```

The following function takes in the matrix of test statistics and returns the estimated correlation.

```
get_cor <- function(rawt.all)
{

  # calculate correlation
  cor.tstat <- cor(rawt.all)
  est.cor <- cor.tstat[lower.tri(cor.tstat)]
  mean.est.cor <- mean(est.cor)

  return(mean.est.cor)
}
```

Simulation results

The following sets up the design and DGP parameters.

```
d_m <- 'd3.3_m3rc2rc'
model.params.list <- list(
  M = 3                           # number of outcomes
  , J = 30                        # number of schools
  , K = 10                        # number of districts
  # (for two-level model, set K = 1)
  , nbar = 50                     # number of individuals per school
  , rho.default = 0.5             # default rho value (optional)
  ################################################## impact
  , MDES = 0.125                  # minimum detectable effect size
  ################################################## level 3: districts
  , numCovar.3 = 1                # number of district covariates
  , R2.3 = 0.1                    # percent of district variation
  # explained by district covariates
  , ICC.3 = 0.2                   # district intraclass correlation
  , omega.3 = 0.1                 # ratio of district effect size variability
  # to random effects variability
  ################################################## level 2: schools
  , numCovar.2 = 1                # number of school covariates
  , R2.2 = 0.1                    # percent of school variation
  # explained by school covariates
  , ICC.2 = 0.2                   # school intraclass correlation
  , omega.2 = 0.1                 # ratio of school effect size variability
  # to random effects variability
  ################################################## level 1: individuals
  , numCovar.1 = 1                # number of individual covariates
```

```
  , R2.1 = 0.1                        # percent of indiv variation explained
  # by indiv covariates
)
Tbar <- 0.5
```

```
n.sims <- 10
out.data <- NULL

rawt.all <- get_rawt(
  d_m = d_m, model.params.list = model.params.list, Tbar = Tbar, n.sims = n.sims
)
```

```
## Using default rho for all rho matrices,
##                 overriding any user-input rho matrices.
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.00220016 (tol = 0.002, component 1)
```

```
est.cor <- get_cor(rawt.all)

cor.data <- data.frame(
  d_m = d_m,
  input.rho = model.params.list$rho.default,
  output.rho = est.cor,
  n.sims = n.sims
)
out.data <- rbind(out.data, cor.data)
print(out.data)
```

```
##            d_m input.rho output.rho n.sims
## 1 d3.3_m3rc2rc       0.5  0.4239692     10
```