Simulations on "Bayesian hierarchical weighting adjustment and survey inference" by Trangucci et al.

Fall 2018

D.Bonnéry

1 Model description

1.1 Trangucci et al. model0

We describe the model given in (**?**, Sec. 2).

- Population made of H strata $U_1,\ldots,U_J$.
- Stratum j size: N_j, Total size: $N=\sum_j N_j$
- Sample size in stratum j: n_j
- I vector: sample indicator
- y vector: study variable
- θ_j: average of y in stratum j. $\theta_j=\sum_{i\in U_j} y_i$. In the paper there is an ambiguity: First, θ is called the "population estimand of interest[...] the overall or domain mean". Ambiguity comes from the term "mean". Is $\theta=N^{-1}\sum_i y_i$ or Is $\theta=N^{-1}\sum_k E[y_i]$?
- X : Design variables. In the paper $X^1,\ldots,X^J$ are badly defined. The idea is that X variables are Q categorical variables. The strata correspond to the cells obtained from these categorical variables.

- denote by $1,\dots K_q$ the categories for variable X_q.
- denote by $j[i]$ the stratim of unit i
- Denote by $k[q,j]$ the category for variable X_q in stratum j

Consider the following hierarchical model:

$$y_i \sim \mathcal{N}\left(\theta^{\star}_{j[i]}, \sigma_y^2\right) \tag{1.1}$$

$$\theta^{\star}_j = \alpha_0 + \sum_{\ell=1}^{Q}\left(\sum_{q_1<\dots<q_\ell\in\{1,\dots,Q\}} \alpha_j^{(q_1,\dots,q_\ell)}\right) \tag{1.2}$$

$$\alpha_j^{(q_1,\dots,q_\ell)} \sim \mathcal{N}(0, (\lambda_j^{(q_1,\dots,q_\ell)}\sigma)^2) \tag{1.3}$$

$$\lambda_j^{(q_1,\dots,q_\ell)} = \delta^{(\ell)}\prod_{l=1}^{\ell}\lambda_{0,k[q_l,j]}^{(q_l)} \tag{1.4}$$

$$\sigma \sim \text{Cauchy}_+(0,1) \tag{1.5}$$

$$\lambda_{0,k}^{(q)} \sim \mathcal{N}_+(0,1) \tag{1.6}$$

$$\delta^{(\ell)} \sim \mathcal{N}_+(0,1) \tag{1.7}$$

$$\sigma_y \sim \text{Cauchy}_+(0,5) \tag{1.8}$$

$$\alpha_0 \sim \text{N}(0,10) \tag{1.9}$$

Note I made up the prior on α_0 as I did not find in the paper.

1.2 Competing model

We describe the model given in (**?**, Sec. 2).

Consider the following hierarchical model:

$$y_i \sim \mathcal{N}\left(\theta^{☾}_{j[i]}, \sigma_y^2\right) \tag{1.10}$$

$$\theta^{\star}_j = \alpha_0 + \sum_{\ell=1}^{Q}\left(\sum_{q_1<\dots<q_\ell\in\{1,\dots,Q\}} \alpha_j^{(q_1,\dots,q_\ell)}\right) \tag{1.11}$$

$$\alpha_j^{(q_1,\dots,q_\ell)} \sim \mathcal{N}(0, (\lambda_j^{(q_1,\dots,q_\ell)}\sigma)^2) \tag{1.12}$$

$$\lambda_j^{(q_1,\dots,q_\ell)} = \delta^{(\ell)}\prod_{l=1}^{\ell}\lambda_{0,k[q_l,j]}^{(q_l)} \tag{1.13}$$

$$\sigma \sim \text{Cauchy}_+(0,1) \tag{1.14}$$

$$\lambda_{0,k}^{(q)} \sim \mathcal{N}_+(0,1) \tag{1.15}$$

$$\delta^{(\ell)} \sim \mathcal{N}_+(0,1) \tag{1.16}$$

$$\sigma_y \sim \text{Cauchy}_+(0,5) \tag{1.17}$$

$$\alpha_0 \sim \text{N}(0,10) \tag{1.18}$$

Note I made up the prior on α_0 as I did not find in the paper.

2 Data Generation

Step 1

Create procedures that can

1. generate a population of size N and with Q random categorical variables $X_1,\dots,X_Q$. X_q is generated by drawing with replacement in $\{1,\dots,K_q\}$ where $K_q > 1$, K_q can be generated by drawing from $\mathcal{U}_{1,\dots,p}$.
2. compute the corresponding stratum $j = 1,\dots,J$ and the correspondances $(k[q,j])_{j=1,\dots,J,q=1,\dots,Q}$.
3. generate, for such a population, the hyper parameters σ_y, α_0, $(\delta^\ell)_{\ell=1,\dots,Q}$,$\lambda_k^l$,$\sigma$.
4. compute the $\lambda_{k_1,\dots,k_\ell}^{(q_1,\dots,q_\ell)}$, for possible values of $k_1,\dots,k_\ell{}^{(q_1,\dots,q_\ell)}$.
5. generate the $\alpha_{j,(k_1,\dots,k_\ell)}^{(q_1,\dots,q_\ell)}$.
6. compute $\theta^\star$.
7. generate a number r of realisations of y for such hyperparameters and such strata.
8. Set seed.

9. Create a population with $N = 10000$,$Q = 2$, $p = 5$. Display J, N_j, $\theta^\star$.

For 1-8, see R code below.
9.
9. This are tables we obtained with different seeds.

Table 2.1:

	X1	X2	Strata	N_j	thetastar
S1	1	1	S1	161	-1.839
S2	1	2	S2	176	0.116
S3	2	1	S3	169	2.321
S4	2	2	S4	163	1.172
S5	3	1	S5	154	-0.536
S6	3	2	S6	177	-0.938

Table 2.2:

	X1	X2	Strata	N_j	thetastar
S1	1	1	S1	76	-2.848
S2	1	2	S2	73	-2.935
S3	1	3	S3	60	-3.077
S4	2	1	S4	68	-3.135
S5	2	2	S5	75	-3.151
S6	2	3	S6	66	-3.051
S7	3	1	S7	64	-3.103
S8	3	2	S8	65	-2.832
S9	3	3	S9	59	-3.017
S10	4	1	S10	59	-2.695
S11	4	2	S11	70	-2.842
S12	4	3	S12	64	-3.270
S13	5	1	S13	74	-2.923
S14	5	2	S14	57	-2.834
S15	5	3	S15	70	-3.419

```
#' Generate design variables
#' @details
#' Generates X, Stratum indicator, computes N_j's and model matrix.
#' @param N population size
#' @param Q Number of design variables
#' @param p maximum number of levels for design variables, >=2.
#' @examples
#' N=1000;Q=2;p=5
Gen_design_variables<-function(N,Q,p,K_q=sample(2:p,Q,replace=T)){
  vars<-paste0("X",1:Q,"_")
  X<-aperm(
    plyr::aaply(K_q,1,function(x){sample(x,N,replace=T)}),2:1)
  X<-X[do.call(order,as.data.frame(X)),]
```

```
    dimnames(X)<-list(k=1:N,Variable=vars)
    Strata<-unique.array(X,margin=1)
    J<-dim(Strata)[1]
    Strata=cbind(Strata,Strata=paste0("S",1:J))
    rownames(Strata)<-Strata[,"Strata"]
    merge(X,Strata, by=colnames(X))->Xd
    Xd[1:Q]<-plyr::llply(Xd[1:Q],as.factor)
    K_q2<-plyr::laply(Xd[1:Q],nlevels)
    names(K_q2)<-vars
    names(Xd$Strata)<-NULL
    list(Xd=Xd,
         vars=vars,
         Q=Q,
         K_q=K_q,
         K_q2=K_q2,
         N_j=plyr::daply(Xd,~Strata,nrow),
         Strata=Strata,
         J=J,
         X.model.matrix<-do.call(
           cbind,plyr::alply(
             c(vars,"Strata"),1,
             function(x){model.matrix(as.formula(paste0("~0+",x)),Xd)})),
         k<-plyr::maply(expand.grid(q=1:Q,j=1:J),function(q,j){Strata[j,q]}))
 }
#' Generate lambda1s
#' @argument XX output from \code[Gen_design_variables]
#'
lambda1f<-function(XX){
    plyr::alply(XX$vars,1,function(q){
      x=abs(rnorm(XX$K_q2[q]))
      names(x)<-as.character(1:XX$K_q2[q])
      x})}
#' Create the lambda_{k1...kell}^{q1...qell}
#'
#' @param XX an output from [Gen_design_variables]
#' @param lambda1 an object with same structure than [lambda1f(XX)]
#' @param delta an object with same structure than [Gen_hyper_parameters(XX)$delta]
#' @examples

lambdaf<-function(XX,lambda1,delta){
      plyr::alply(1:XX$Q,1,function(ell){
      comb<-combn(XX$Q,ell)
      dimnames(comb)<-list(q_l=paste0("q_",1:ell),q1...qell=plyr::aaply(comb,2,paste,
          collapse="."))
      #concernedk1...kell<-unique(XX$Strata[,q1...qell])
      list(comb=comb,
           lambda.q1...qell=plyr::aaply(comb,2,function(q1...qell){
        lambdas=plyr::aaply(XX$Strata[,q1...qell],1,
          .fun=function(x){
          prod(plyr::aaply(1:ell,1,function(i){lambda1[[q1...qell[i]]][x[i]]}))*delta[ell
              ]})
      }))
    })}
#' Create the alpha_{k1...kell}^{q1...qell}
#'
#' @param lamda an object with same structure than [lambda1f(XX)]
#' @param delta an object with same structure than
#' @examples
```

```
alphaf<-function(lambda,sigma){
   plyr::llply(lambda,function(x){
     y<-x
     dime<-if(!is.null(dim(x$lambda.q1...qell))){1:length(dim(x$lambda.q1...qell))}else
         {1}
     y[2]<-list(plyr::aaply(x$lambda.q1...qell,dime,
                                         function(xx){rnorm(1,sd=sigma*xx)}))
     names(y)[2]="alpha.q1...ql"
     y})
   }
thetastarf<-function(alpha,alpha0){
     alphas<-do.call(rbind,
     plyr::llply(alpha,function(x){x$alpha.q1...ql}))
     dimnames(alphas)[[1]]<-1:(dim(alphas)[1])
     thetastar<-alpha0+plyr::aaply(alphas,2,sum)}
yf<-function(XX,thetastar,sigma_y,nrep){
   Strata<-data.frame(XX$Strata,
                 N_j=XX$N_j,
                 thetastar=thetastar)
   do.call(rbind,plyr::alply(1:nrow(Strata),1,function(x){matrix(rnorm(Strata$N_j[x]*
      nrep,mean=Strata$thetastar[x],sd=sigma_y),Strata$N_j[x],nrep)}))
 }
#' Create model hyper-parameters and parameters procedure
#'
#' @param XX an output from [Gen_design_variables]
#' @examples
#' N=1000;Q=2;p=3
#' XX<-Gen_design_variables(N,Q,p)

Gen_hyper_parameters<-function(XX){
 #hyper parametres
   sigma_y<-abs(5*rt(1,1))
   delta<-abs(rnorm(XX$J))
   lambda1<-lambda1f(XX)
   sigma<-abs(rt(1,1))
   alpha0<-rnorm(1,sd=sqrt(10))
     list(sigma_y=sigma_y,
          delta=delta,
          lambda1=lambda1,
          sigma=sigma,
          alpha0=alpha0)
   }
Generate_all<-function(N=NULL,Q=NULL,p=NULL,
                        K_q=sample(2:p,Q,replace=T),
                        XX=Gen_design_variables(N,Q,p,K_q),
                        hyper=Gen_hyper_parameters(XX),
                        nrep=3){
 #depending parameters
   lambda<-lambdaf(XX,hyper$lambda1,hyper$delta)
   alpha<-alphaf(lambda,hyper$sigma)
   thetastar<-thetastarf(alpha,hyper$alpha0)
   y<-yf(XX,thetastar,sigma_y,nrep)

   list(N=N,Q=Q,p=p,K_q=K_q,hyper=hyper,XX=XX,nrep=nrep,lambda=lambda,alpha=alpha,
      thetastar=thetastar,y=y)}
set.seed(11)
GG<-Generate_all(N=1000,Q=2,p=5)
Strata1<-data.frame(GG$XX$Strata,
```

```
                   N_j=GG$XX$N_j,
                   thetastar=GG$thetastar)
set.seed(7)
GG<-Generate_all(N=1000,Q=2,p=5)
Strata2<-data.frame(GG$XX$Strata,
                   N_j=GG$XX$N_j,
                   thetastar=GG$thetastar)
```

3 Bayesian computations

Step 1

Using Jags,

1. Draw posterior distribution of $\theta_j^\star$ for the largest value of $\theta_j^\star$, obtained from the observation of $y^{(1)}$ only. Add real value of θ_1 and prediction to the plot.

2. Draw distribution of predictions of $\theta_j^\star$. Add real value of θ_j to the plot.

3. Draw real values of all $\theta_j^\star$ vs $r = 30$ predictions.

Some functions to generate the jags model file:
library(SimuTrangucci)

Step 2

same thing with Stan

Step 3

1. Design a sampling scheme that favors some cells

2. Compute Trangucci estimator for θ for the $r = 30$ realisations.

Step 4

1. Use another model to generate the population, a model that does not fit the current one, to fail the product structure.

2. Look at predictions for pop total.

1. Assume the following model:

2.

4 Bayesian computations