Bayesian SAE using Complex Survey Data

Lecture 6B: Introduction to SAE in R

Richard Li

Department of Statistics
University of Washington

Outline

Example: Two-stage stratified sampling

Example: Spatial smoothing with survey designs

Example: Two-stage stratified sampling

Example: DHS model data

- Two-stage or multi-stage stratified sampling is common in DHS.
- Strata is some times available directly (v023)
- Strata is usually defined by region (v024) and urban/rural (v025)
- Two-stage clusters usually defined by
  - First stage: Sample enumeration areas (usually v001)
  - Second stage: Sample households (usually v002)

Example: SAE with DHS model data

We use the simulated and cleaned data from the DHS model dataset, available from the `SUMMER` package. We renamed 'v001', 'v002' into 'clustid' and 'id'. The strata and weights are also defined already.

```
# install.packages('SUMMER')
library(SUMMER)
data(DemoData2)
head(DemoData2)

##   clustid id  region age  weights        strata tobacco.use
## 1       1  1 nairobi  30 1.057703 nairobi.urban           0
## 2       1  3 nairobi  22 1.057703 nairobi.urban           0
## 3       1  4 nairobi  42 1.057703 nairobi.urban           0
## 4       2  4  nyanza  25 1.057703  nyanza.urban           0
## 5       1  5 nairobi  25 1.057703 nairobi.urban           0
## 6       1  6 nairobi  37 1.057703 nairobi.urban           0
```

Weighted mean estimates for small areas

```r
library(survey)
design <- svydesign(ids = ~clustid + id, weights = ~weights,
    strata = ~strata, data = DemoData2)
svyby(~age, by = ~region, design = design, svymean)

##                     region      age        se
## central            central 28.67120 0.3171558
## nairobi            nairobi 28.49361 0.2908435
## eastern            eastern 27.88147 0.5753705
## coast                coast 28.75124 0.4210734
## northeastern northeastern 28.09800 0.3023312
## nyanza              nyanza 28.40967 0.4202241
## western            western 26.82650 0.5830388
## rift valley    rift valley 29.42140 0.5029015
```

Weighted mean estimates for small areas

```
tob <- svyby(~tobacco.use, by = ~region, design = design,
    svymean)
tob

##                    region tobacco.use          se
## central           central  0.04269545 0.007888049
## nairobi           nairobi  0.02748435 0.005712901
## eastern           eastern  0.03453175 0.011373092
## coast               coast  0.07381327 0.008829939
## northeastern northeastern  0.03735942 0.006102720
## nyanza             nyanza  0.02809128 0.006678518
## western           western  0.03462895 0.009799475
## rift valley   rift valley  0.07163454 0.014714351

p.i <- tob$tobacco.use
dv.i <- tob$se^2
```

Naive mean estimates

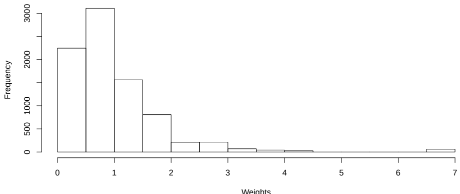
```
n.area <- 8
regions <- as.character(tob[, 1])
props <- matrix(NA, nrow = n.area, ncol = 5)
props <- as.data.frame(props)
colnames(props) <- c("region", "p.hat", "se.p.hat",
    "y.i", "n.i")
props[, 1] <- regions
for (i in 1:n.area) {
    props[i, "p.hat"] <- mean(DemoData2[DemoData2$region ==
        regions[i], "tobacco.use"])
    props[i, "y.i"] <- sum(DemoData2[DemoData2$region ==
        regions[i], "tobacco.use"])
    props[i, "n.i"] <- sum(DemoData2$region == regions[i])
    naivevar <- props[i, "p.hat"] * (1 - props[i, "p.hat"])/props[i,
        "n.i"]
    props[i, "se.p.hat"] <- sqrt(naivevar)
}
```

Comparison: weights

- The weights have high variability.
- The coefficient of variation of the weights is related to the size of the design effect, i.e., to the loss of efficiency compared to simple random sampling. Specifically, $\mathrm{CV}^2/(\mathrm{CV}^2+1)$ approximates the inefficiency of using the weights

```
hist(DemoData2$weights, xlab = "Weights", main = "")
cv <- sqrt(var(DemoData2$weights, na.rm = T))/mean(DemoData2$weights,
    na.rm = T)
cv^2/(cv^2 + 1)

## [1] 0.4069978
```

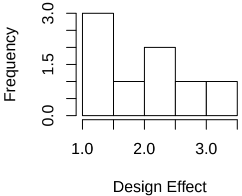


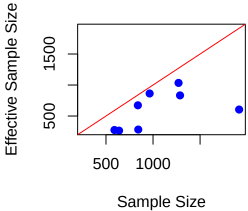
Comnparison: design effect

The design effect for $\hat{p}_i$ is defined as

$$\text{Deff} = \frac{\text{Variance of estimator given complex design}}{\text{Variance of estimator if simple random sampling}}.$$

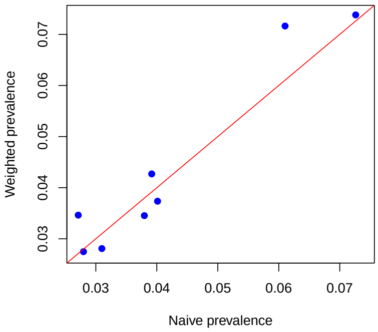
```
unwtvar <- props[, "se.p.hat"]^2
deff <- dv.i/unwtvar
effss <- props[, "n.i"]/deff
par(mfrow = c(1, 2))
hist(deff, main = "", xlab = "Design Effect")
lim <- range(c(effss, props[, "n.i"]))
plot(effss ~ props[, "n.i"], pch = 19, col = "blue",
    xlab = "Sample Size", ylab = "Effective Sample Size",
    xlim = lim, ylim = lim)
abline(0, 1, col = "red")
```





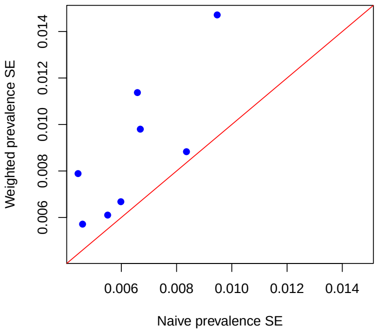
Comparison: prevalence

```
lim <- range(c(p.i, props[, "p.hat"]))
plot(p.i ~ props[, "p.hat"], pch = 19, col = "blue",
    xlab = "Naive prevalence", ylab = "Weighted prevalence",
    xlim = lim, ylim = lim)
abline(0, 1, col = "red")
```



Comparison: SE of prevalence

```
lim <- range(c(sqrt(dv.i), props[, "se.p.hat"]))
plot(sqrt(dv.i) ~ props[, "se.p.hat"], pch = 19, col = "blue",
    xlab = "Naive prevalence SE", ylab = "Weighted prevalence SE
    xlim = lim, ylim = lim)
abline(0, 1, col = "red")
```



Example: Spatial smoothing with survey designs

Data simulation

- We generate some synthetic normally distributed variable for height for each observation.
- Suppose we denote the height of observation k in area i to be x_{ik}, and the associated design weight to be w_{ik}.
- Under the design-based approach to inference, we can calculate the weighted estimator of mean height to be

$$\hat{\mu}_i = \frac{\sum_k w_{ik} x_{ik}}{\sum_k w_{ik}}$$

- The associated variance $\widehat{var}(\hat{\mu}_i)$. We then use INLA to fit the following Bayesian hierarchical model:

$$\begin{aligned}
\hat{\mu}_i &\sim \text{Normal}(\mu_i, \widehat{var}(\hat{\mu}_i)) \\
\mu_i &= \beta + \epsilon_i + \delta_i, \\
\epsilon_i &\sim \text{Normal}(0, \sigma_\epsilon^2) \\
\delta_i &\sim \text{ICAR}(\sigma_\delta^2)
\end{aligned}$$

Data simulation

To simulate from this generative model, we first simulate from the ICAR random fields as follows

```
set.seed(1)
sim.Q <- function(Q) {
    eigenQ <- eigen(Q)
    rankQ <- qr(Q)$rank
    sim <- as.vector(eigenQ$vectors[, 1:rankQ] %*%
        matrix(rnorm(rep(1, rankQ), rep(0, rankQ),
            1/sqrt(eigenQ$values[1:rankQ])), ncol = 1))
    sim
}
Q <- DemoMap2$Amat * -1
diag(Q) <- 0
diag(Q) <- -1 * apply(Q, 2, sum)
struct.error <- sim.Q(Q) * 2
unstruct.error <- rnorm(length(struct.error), sd = 0.3)
```

For details, see Algorithm 3.1 in Rue & Held (2005).

Data simulation

We randomly assign the simulated height variable to observations in `DemoData2`

```
mu <- 70 + struct.error + unstruct.error
regions <- colnames(DemoMap2$Amat)
DemoData2$height <- rnorm(dim(DemoData2)[1], sd = 12) +
    mu[match(DemoData2$region, regions)]
```

Smoothing that takes into account of survey designs

- We can use the ‘fitSpace()‘ function to obtain both the survey-weighted direct estimates and the smoothed estimates from INLA.
- We will discuss more about the details of this function in the next lecture.

```
fit <- fitSpace(data=DemoData2, geo=DemoMap2$geo,
                Amat=DemoMap2$Amat, family="gaussian",
                responseVar="height", strataVar="strata",
                weightVar="weights", regionVar="region",
                clusterVar = "~clustid+id", CI = 0.95,
                hyper = c(0.5, 0.0005))
```

Survey weighted estimates

The survey weighted estimates are

```
fit$HT[, c("HT.est", "HT.variance", "region")]
```

```
##      HT.est HT.variance       region
## 2 69.34625   0.1987104      nairobi
## 1 69.55075   0.1302739      central
## 4 71.80851   0.2705694        coast
## 3 70.27787   0.1022470      eastern
## 6 69.96753   0.2236180       nyanza
## 8 69.26213   0.3126383  rift valley
## 7 68.91053   0.3325231      western
## 5 70.88231   0.1327074 northeastern
```

Smoothed estimates

The smoothed estimates are

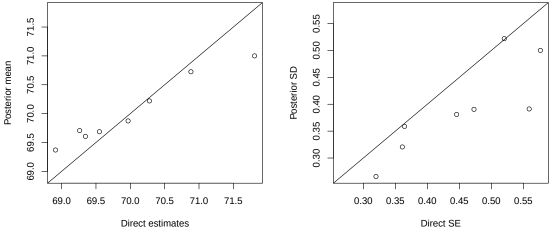
```
fit$smooth[, c("mean", "sd", "median", "lower", "upper",
    "region")]
```

```
##       mean        sd   median    lower    upper       region
## 1 69.60634 0.3809814 69.62256 68.81997 70.29557      nairobi
## 2 69.68676 0.3205864 69.69560 69.03690 70.28269      central
## 3 71.00029 0.5220010 70.98588 70.05018 72.06232        coast
## 4 70.22003 0.2656298 70.21423 69.70971 70.75722      eastern
## 5 69.87365 0.3905091 69.88039 69.09333 70.63879       nyanza
## 6 69.70663 0.3912559 69.73822 68.85466 70.39281  rift valley
## 7 69.36993 0.5001564 69.39092 68.33623 70.26018      western
## 8 70.72597 0.3588411 70.72754 70.03034 71.42891 northeastern
```

Effect of smoothing

To see the effect of smoothing, we plot the smoothed estimates and standard errors agains the direct estimates and their standard errors.

```
par(mfrow = c(1, 2))
lim <- range(c(fit$HT$HT.est, fit$smooth$mean))
plot(fit$HT$HT.est, fit$smooth$mean, xlim = lim, ylim = lim,
    xlab = "Direct estimates", ylab = "Posterior mean")
abline(c(0, 1))
lim <- range(c(fit$HT$HT.sd, fit$smooth$sd))
plot(fit$HT$HT.sd, fit$smooth$sd, xlim = lim, ylim = lim,
    xlab = "Direct SE", ylab = "Posterior SD")
abline(c(0, 1))
```

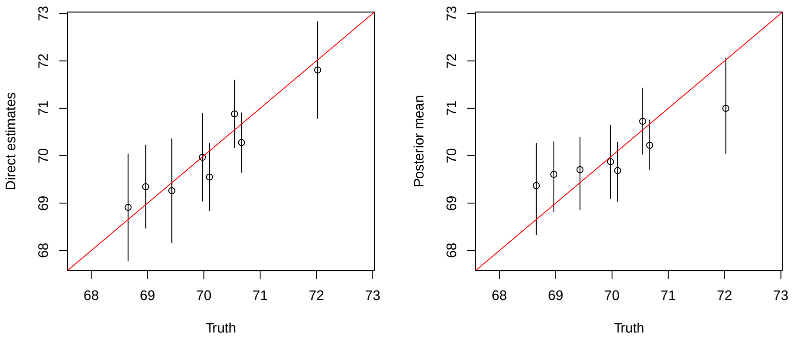


Effect of smoothing

To see how they compare to the true area-specific means

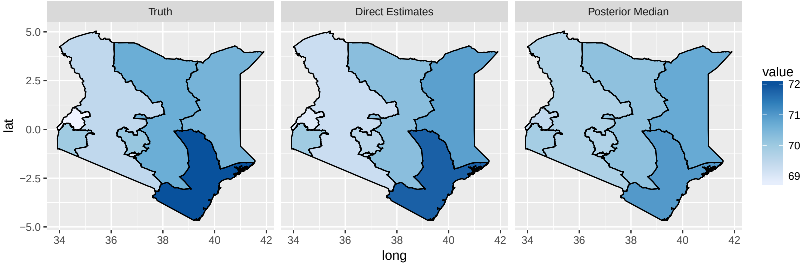
```
par(mfrow = c(1, 2))
truth <- mu[match(fit$HT$region, regions)]
fit$HT$lower <- fit$HT$HT.est - 1.96 * fit$HT$HT.sd
fit$HT$upper <- fit$HT$HT.est + 1.96 * fit$HT$HT.sd
lim <- range(c(fit$HT$HT.est, fit$HT$lower, fit$HT$upper,
    truth))
plot(truth, fit$HT$HT.est, xlim = lim, ylim = lim,
    xlab = "Truth", ylab = "Direct estimates")
segments(x0 = truth, x1 = truth, y0 = fit$HT$lower,
    y1 = fit$HT$upper)
abline(c(0, 1), col = "red")
plot(truth, fit$smooth$mean, xlim = lim, ylim = lim,
    xlab = "Truth", ylab = "Posterior mean")
segments(x0 = truth, x1 = truth, y0 = fit$smooth$lower,
    y1 = fit$smooth$upper)
abline(c(0, 1), col = "red")
```

Effect of smoothing



Effect of smoothing

```
combined <- merge(fit$HT, fit$smooth, by = "region")
combined$truth <- mu[match(combined$region, regions)]
mapPlot(data = combined, geo = DemoMap2$geo,
        variables=c("truth", "HT.est", "median"),
        labels = c("Truth", "Direct Estimates", "Posterior Median"),
        by.data = "region", by.geo = "NAME_final", is.long=FALSE)
```



Effect of smoothing: uncertainty

```
mapPlot(data = combined, geo = DemoMap2$geo, variables = c("HT.sd",
    "sd"), labels = c("SD(direct estimates)", "SD(posterior median)"),
    by.data = "region", by.geo = "NAME_final", is.long = FALSE)
```