Bayesian SAE using Complex Survey Data

Lecture 7B: SAE in R

Richard Li

Department of Statistics
University of Washington

Outline

SAE with BRFSS data

BRFSS data

- These notes were originally prepared with the help of Jessica Godwin, and Laina Mercer, Cici Bauer and Thomas Lumley also worked on the methodology and coding, see Chen et al. (2014) and Mercer et al. (2014) for further methodological details.
- We take as example, the estimation of the prevalence of Type II diabetes in health reporting areas (HRAs) in King County, using BRFSS data.
- These survey data are collected using a complex stratified design.
- The design must be acknowledged in the analysis, but we would like to use spatial smoothing to obtain estimates with more precision.

Outline

We present results from the following analyses:

- Naive (i.e. unweighted, unsmoothed)
- Binomial spatial smoothing model, ignoring weighting
- Weighted (unsmoothed)
  - By hand and using SUMMER package
- Smoothed and weighted
  - By hand and using SUMMER package

Load data

First, we need to read in the King County BRFSS Stata dataset using the `foreign` package.

```
library(foreign)
# kingdata <-
# read.dta(url('http://faculty.washington.edu/jonno/PAA-SAE/ct09
kingdata <- read.dta("../data/ct0913all.dta")
names(kingdata)

##  [1] "age"       "pracex"    "educau"    "zipcode"   "sex"       "
##  [8] "seqno"     "year"      "hispanic" "mracex"    "_ststr"    "
## [15] "rwt_llcp" "genhlth2" "fmd"       "obese"     "smoker1"   "
## [22] "zipout"    "streetx"   "ethn"      "age4"      "ctmiss"
```

Load map

Next, read in the shape files for King County HRAs

```r
# install.packages('maptools')
library(maptools)
f <- "../data/HRA_ShapeFiles/HRA_2010Block_Clip.shp"
kingshape <- readShapePoly(f)
```

```r
# install.packages('rgdal')
library(rgdal)
kingshape <- readOGR("../data/HRA_ShapeFiles",
    layer = "HRA_2010Block_Clip")

## OGR data source with driver: ESRI Shapefile
## Source: "../data/HRA_ShapeFiles", layer: "HRA_2010Block_Clip"
## with 48 features
## It has 9 fields
```

Initial data cleaning

- Our outcome of interest is Type II diabetes and we will drop observations with missing diabetes data.
- Our small area of interest is the HRA. We will also drop observations with missing HRA.

```r
kingdata <- subset(kingdata, !is.na(kingdata$diab2))
kingdata <- subset(kingdata, !is.na(kingdata$hracode))
names(kingdata)[names(kingdata) == "_ststr"] <- "strata"
kingdata$hracode <- as.character(kingdata$hracode)
kingdata[kingdata$hracode == "Fairwood       ",
    "hracode"] <- "Fairwood"
n.area <- length(unique(kingdata$hracode))
```

Naive binomial model

- Let y_i and m_i be the number of individuals flagged as having type II diabetes and the denominators in the $i = 1, \dots, n$ areas.
- We form naive estimates
$$\hat{p}_i = \frac{y_i}{m_i}$$
with associated standard errors
$$\sqrt{\frac{\hat{p}_i(1-\hat{p}_i)}{m_i}}.$$

Naive binomial model

Similar to what we have worked out in Lecture 6

```r
hras <- as.character(kingshape$HRA2010v2_)
props <- matrix(NA, nrow = n.area, ncol = 5)
props <- as.data.frame(props)
colnames(props) <- c("hracode", "p.hat",
    "se.p.hat", "y.i", "n.i")
props[, 1] <- hras
for (i in 1:n.area) {
    props[i, "p.hat"] <- mean(kingdata[kingdata$hracode ==
        props[i, "hracode"], "diab2"])
    props[i, "y.i"] <- sum(kingdata[kingdata$hracode ==
        props[i, "hracode"], "diab2"])
    props[i, "n.i"] <- length(kingdata[kingdata$hracode ==
        props[i, "hracode"], "diab2"])
    naivevar <- props[i, "p.hat"] * (1 -
        props[i, "p.hat"])/props[i, "n.i"]
    props[i, "se.p.hat"] <- sqrt(naivevar)
}
```

Naive binomial model: merge into map

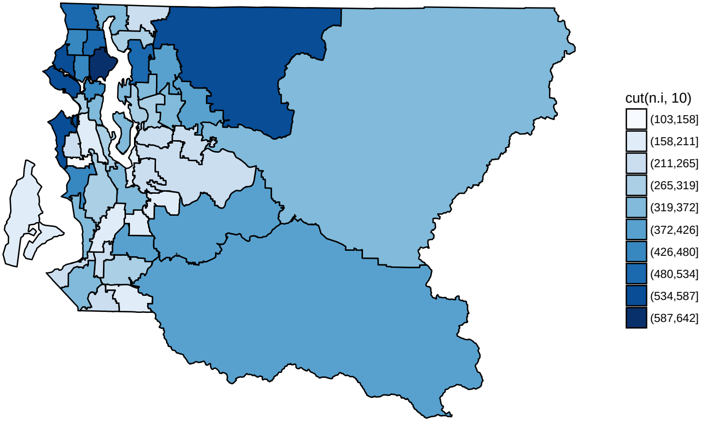
```r
library(ggplot2)
library(RColorBrewer)
geo <- fortify(kingshape, region = "HRA2010v2_")
geo1 <- merge(geo, props, by = "id", by.y = "hracode")
```

Sample size by region

```r
g <- ggplot(geo1)
g <- g + geom_polygon(aes(x = long, y = lat,
    group = group, fill = cut(n.i, 10)),
    color = "black")
g <- g + theme_void()
g <- g + scale_fill_manual(values = colorRampPalette(brewer.pal(9,
    "Blues"))(10))
g
```

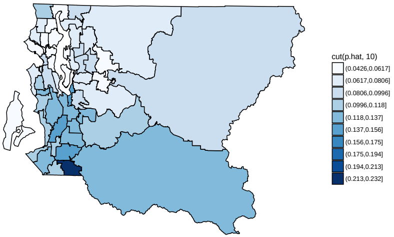
Naive binomial model: merge into map

Sample size by region



Naive binomial model: prevalence

```r
g <- ggplot(geo1)
g <- g + geom_polygon(aes(x = long, y = lat,
    group = group, fill = cut(p.hat, 10)),
    color = "black")
g <- g + theme_void()
g <- g + scale_fill_manual(values = colorRampPalette(brewer.pal(9,
    "Blues"))(10), drop = FALSE)
g
```



‘drop=FALSE’ makes sure all bins show up in the legend.

Binomial smoothing by hand (not weighted)

Binomial smoothing: the model

We use the INLA package to fit the following Bayesian hierarchical model:

$$\begin{aligned} y_i|p_i &\sim \text{Binomial}(N_i, p_i) \\ \theta_i &= \log\left(\frac{p_i}{1-p_i}\right) = \mu + \epsilon_i + s_i, \\ \epsilon_i &\sim N(0, \sigma_\epsilon^2) \\ s_i|s_j, j \in \text{ne}(i) &\sim N\left(\bar{s}_i, \frac{\sigma_s^2}{n_i}\right). \end{aligned}$$

where n_i is the number of neighbors for area i, and

$$\bar{s}_i = \frac{1}{n_i} \sum_{j \in \text{ne}(i)} s_j$$

Priors are put on $\mu, \sigma_\epsilon^2, \sigma_s^2$,

Binomial smoothing: construct adjacency matrix

To perform spatial smoothing using ICAR, we first need to construct an adjacency matrix where each row and column is a region.

- Diagonal elements are 0
- Off-diagonal elements are 1 if the two corresponding regions are adjacent and 0 if otherwise

```
library(spdep)
nb.r <- poly2nb(kingshape, queen=F,
                row.names = kingshape$HRA2010v2_)
mat <- nb2mat(nb.r, style="B",zero.policy=TRUE)
colnames(mat) <- rownames(mat)
mat <- as.matrix(mat[1:dim(mat)[1], 1:dim(mat)[1]])
```

Binomial smoothing: model fitting

Implementation details:

- The index of the areas needs to be the same order as in the adjacency matrix. *It can be easily missed if data has been reordered*
- Multiple random effects each need an index variable (`unstruct` and `struct` below).

```
sum(colnames(mat) != props$region)

## [1] 0

props$unstruct <- props$struct <- 1:n.area
```

Binomial smoothing: model fitting

The following code carries out an unweighted binomial analysis, with global and spatial smoothing, the latter via the ICAR model.

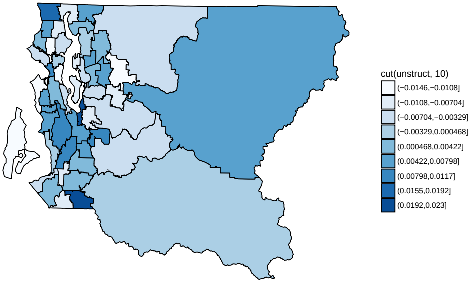
```r
library(INLA)
formula = y.i ~ 1 +
        f(struct,model='besag',
                                adjust.for.con.comp=TRUE,
                                constr=TRUE,graph=mat,
                                scale.model = TRUE,
                                param = c(0.5, 0.0015)) +
        f(unstruct, model='iid',
                                param=c(0.5,0.0015))
fit.naive <- inla(formula,
        family="binomial",
        data=props, Ntrials=n.i,
        control.predictor = list(compute = TRUE))
```

Binomial smoothing: organize output

```r
props.smooth <- props
# posterior median
props.smooth[, "p.hat"] <- fit.naive$summary.fitted.values[,
    "0.5quant"]
# posterior standard deviations
props.smooth[, "se.p.hat"] <- fit.naive$summary.fitted.values[,
    "sd"]
# Post medians of unstructured random
# effects
props.smooth[, "unstruct"] <- fit.naive$summary.random$unstruct[,
    "0.5quant"]
# Post medians of spatial random effects
props.smooth[, "struct"] <- fit.naive$summary.random$struct[,
    "0.5quant"]
```

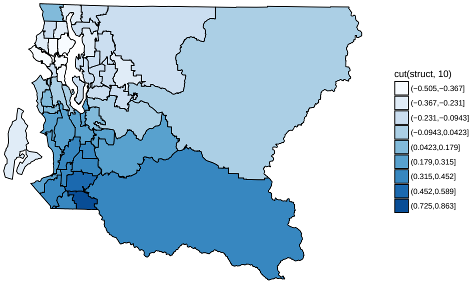
Binomial smoothing: Unstructured random effects

```r
geo2 <- merge(geo, props.smooth, by = "id",
    by.y = "hracode")
g <- ggplot(geo2) + geom_polygon(aes(x = long,
    y = lat, group = group, fill = cut(unstruct,
        10)), color = "black")
g <- g + theme_void()
g <- g + scale_fill_manual(values = colorRampPalette(brewer.pal(9,
    "Blues"))(10))
g
```



Binomial smoothing: Spatial random effects

```
g <- ggplot(geo2) + geom_polygon(aes(x = long,
    y = lat, group = group, fill = cut(struct,
        10)), color = "black")
g <- g + theme_void()
g <- g + scale_fill_manual(values = colorRampPalette(brewer.pal(9,
    "Blues"))(10))
g
```



Binomial smoothing: Proportion of variance (recap)

- It could be interesting to evaluate the proportion of variance explained by the structured spatial component
- However, estimated σ_s^2 and σ_ϵ^2 are not directly comparable
- We alternatively calculates the posterior marginal variance for the structured effect (See Section 6.1.2 of Blangiardo, et.al (2015) for more details.)

```
Sre <- matrix(NA, 1e4, 48)
for (i in 1:48){
  Sre[,i] <- inla.rmarginal(1e4,
        fit.naive$marginals.random$struct[[i]])
}
var.Sre <- apply(Sre,1,var)
var.eps <- inla.rmarginal(1e4, inla.tmarginal(function(x){1/x},
                fit.naive$marginals.hyper$"Precision for unstruct"))
perc.var.Sre <- mean(var.Sre/(var.Sre+var.eps))
perc.var.Sre

## [1] 0.9606218
```

Binomial smoothing: Proportion of variance

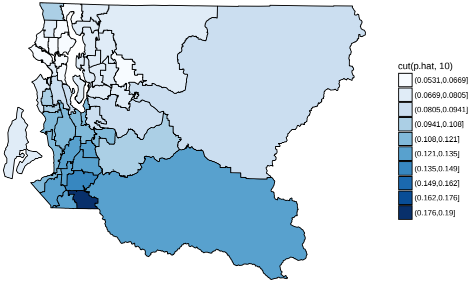
To see there's an difference between σ_s^2 and the posterior marginal variance for the structured effects:

```
var <- matrix(NA, 2, 2)
colnames(var) <- c("S", "Sigma^2")
rownames(var) <- c("median", "mean")
draws1 <- matrix(NA, 10000, 48)
for (i in 1:48) {
    draws1[, i] <- inla.rmarginal(10000,
        fit.naive$marginals.random$struct[[i]])
}
var[1, 1] <- median(apply(draws1, 1, var))
var[2, 1] <- mean(apply(draws1, 1, var))
draws2 <- inla.rmarginal(10000, inla.tmarginal(function(x) 1/x,
    fit.naive$marginals.hyper$"Precision for struct"))
var[1, 2] <- median(draws2)
var[2, 2] <- mean(draws2)
var

##                S    Sigma^2
## median 0.1172726 0.06633293
## mean   0.1179431 0.07036339
```

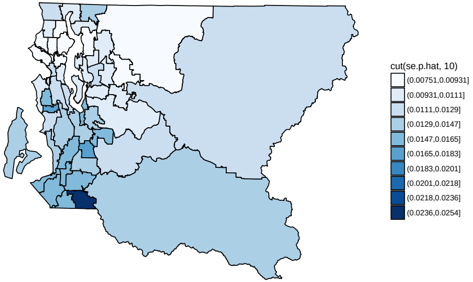
Binomial smoothing: predicted prevalence

```
g <- ggplot(geo2) + geom_polygon(aes(x = long,
    y = lat, group = group, fill = cut(p.hat,
        10)), color = "black")
g <- g + theme_void()
g <- g + scale_fill_manual(values = colorRampPalette(brewer.pal(9,
    "Blues"))(10), drop = F)
g
```



Binomial smoothing: SE of prevalence

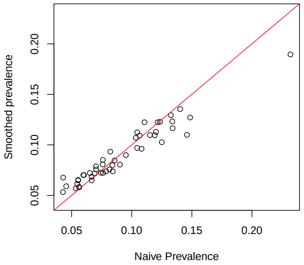
```r
g <- ggplot(geo2) + geom_polygon(aes(x = long,
    y = lat, group = group, fill = cut(se.p.hat,
        10)), color = "black")
g <- g + theme_void()
g <- g + scale_fill_manual(values = colorRampPalette(brewer.pal(9,
    "Blues"))(10), drop = F)
g
```

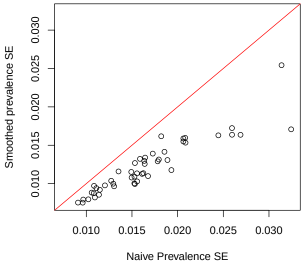


Binomial smoothing: compare with naive approach

```r
par(mfrow = c(1, 2))
lim1 <- range(c(props$p.hat, props.smooth$p.hat))
plot(props$p.hat, props.smooth$p.hat, xlim = lim1,
    ylim = lim1, xlab = "Naive Prevalence",
    ylab = "Smoothed prevalence")
abline(c(0, 1), col = "red")
lim2 <- range(c(props$se.p.hat, props.smooth$se.p.hat))
plot(props$se.p.hat, props.smooth$se.p.hat,
    xlim = lim2, ylim = lim2, xlab = "Naive Prevalence SE",
    ylab = "Smoothed prevalence SE")
abline(c(0, 1), col = "red")
```

Binomial smoothing: compare with naive approach





Accounting for survey designs

Survey weighted estimates: weights

- BRFSS uses a complex survey design. See `http://www.cdc.gov/brfss/annual_data/2013/pdf/Weighting_Data.pdf` for more details of the weighting procedure.
- Raking adjusts for: telephone source (allowing for cell phones), race/ethnicity, education, marital status, age group by gender, gender by race and ethnicity, age group by race and ethnicity, renter/owner status.
- Design weights are

$$\text{STRWT} \times 1/\text{NUMPHON2} \times \text{NUMADULT}.$$

- GEOSTR is the geographical strata (which in general may be the entire state or a geographic subset such as counties, census tracts, etc.). DENSTR is the density of the phone numbers for a given block of numbers as listed or not listed.

Survey weighted estimates: weights

- NRECSTR is the number of available records and NRECSEL is the number of records selected within each geographical strata and density strata.
- Within each GEOSTR $\times$ DENSTR combination, the stratum weight (STRWT) is calculated from the average of the NRECSTR and the sum of all sample records used to produce the NRECSEL. The stratum weight is equal to NRECSTR/NRECSEL, i.e. the reciprocal of the selection probability.
- An adjustment is also made for the mostly cellular telephone dual sampling frame users. Weight trimming also used, prior to trimming.
- The final weight `rwt_llcp` is the raked design weight.

Survey weighted estimates: asymptotic distribution of $\hat{p}_i$

- The survey package will give us survey-weighted estimates of p_i, the proportion of people with Type II diabetes in small area i, and a survey-weighted estimate of the standard error, $\widehat{SE}(\hat{p}_i)$.
- We use the method described in Mercer et al. (2014).
- If we specify $y_i = \log\left(\frac{\hat{p}_i}{1-\hat{p}_i}\right)$ then, by the delta method, the asymptotic (sampling) distribution of y_i is:

$$y_i | p_i \sim N\left(\log\left(\frac{p_i}{1-p_i}\right), \frac{\widehat{\text{var}}(\hat{p}_i)}{\hat{p}_i^{\,2}(1-\hat{p}_i)^2}\right).$$

Survey weighted estimates: calculation

```r
library(survey)
props.w <- props
kingcounty.des <- svydesign(ids = ~1, weights = ~rwt_llcp,
    strata = ~strata, data = kingdata)
weighted <- svyby(~diab2, ~hracode, kingcounty.des,
    svymean)
rows <- match(weighted$hracode, props.w$hracode)
props.w[rows, "p.hat"] <- weighted$diab2
props.w[rows, "se.p.hat"] <- weighted$se
props.w[, "logit.p"] <- log(props.w[, "p.hat"]/(1 -
    props.w[, "p.hat"]))
props.w[, "logit.v"] <- props.w[, "se.p.hat"]^2/(props.w[,
    "p.hat"] * (1 - props.w[, "p.hat"]))^2
props.w[, "logit.prec"] <- 1/props.w[, "logit.v"]
```

Survey weighted estimates: calculation

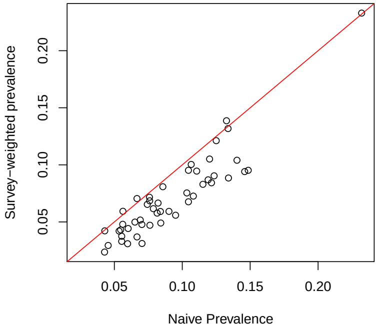
We obtain

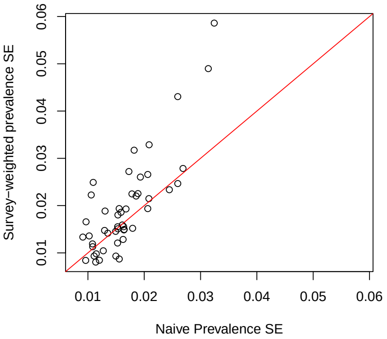
- The weighted estimators of prevalences `p.hat`
- The design standard error of prevalences `se.p.hat`
- The weighted estimators of logits of prevalences `logit.p`
- The design variances of logits of prevalences `logit.v`

Survey weighted estimates: compare with naive approach

```r
par(mfrow = c(1, 2))
lim1 <- range(c(props$p.hat, props.w$p.hat))
plot(props$p.hat, props.w$p.hat, xlim = lim1,
    ylim = lim1, xlab = "Naive Prevalence",
    ylab = "Survey-weighted prevalence")
abline(c(0, 1), col = "red")
lim2 <- range(c(props$se.p.hat, props.w$se.p.hat))
plot(props$se.p.hat, props.w$se.p.hat, xlim = lim2,
    ylim = lim2, xlab = "Naive Prevalence SE",
    ylab = "Survey-weighted prevalence SE")
abline(c(0, 1), col = "red")
```

Survey weighted estimates: compare with naive approach

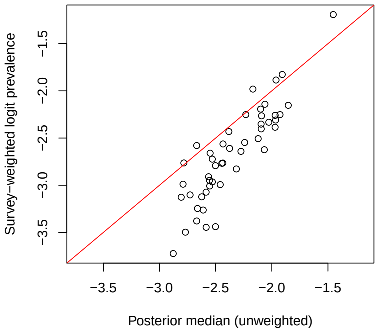


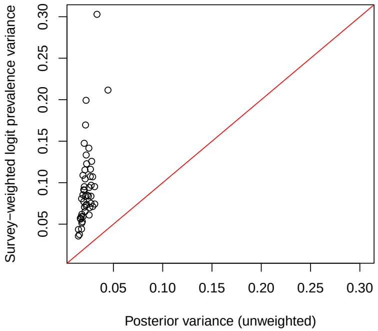


Survey weighted estimates: compare with binomial smoothing

```r
par(mfrow = c(1, 2))
logit.binomial <- fit.naive$summary.linear.predictor[,
    "0.5quant"]
logit.v.binomial <- fit.naive$summary.linear.predictor[,
    "sd"]^2
lim1 <- range(c(logit.binomial, props.w$logit.p))
plot(logit.binomial, props.w$logit.p, xlim = lim1,
    ylim = lim1, xlab = "Posterior median (unweighted)",
    ylab = "Survey-weighted logit prevalence")
abline(c(0, 1), col = "red")
lim2 <- range(c(logit.v.binomial, props.w$logit.v))
plot(logit.v.binomial, props.w$logit.v, xlim = lim2,
    ylim = lim2, xlab = "Posterior variance (unweighted)",
    ylab = "Survey-weighted logit prevalence variance")
abline(c(0, 1), col = "red")
```

Survey weighted estimates: compare with binomial smoothing





Weighted and smoothed model

We use the INLA package to fit the following Bayesian hierarchical model:

$$\begin{aligned} y_i &= \log\left(\frac{\hat{p}_i}{1-\hat{p}_i}\right) \sim N(\theta_i, \hat{V}_i) \\ \theta_i &= \mu + \epsilon_i + s_i, \\ \epsilon_i &\sim N(0, \sigma_\epsilon^2) \\ s_i | s_j, j \in \text{ne}(i) &\sim N\left(\bar{s}_i, \frac{\sigma_s^2}{n_i}\right). \end{aligned}$$

with priors on $\mu, \sigma_\epsilon^2, \sigma_s^2$,
The key here is that the first stage variance $\hat{V}_i$ is assumed known:

$$\hat{V}_i = \frac{\text{var}(\hat{p}_i)}{\hat{p}_i^2(1-\hat{p}_i)^2}.$$

Weighted and smoothed model: model fitting

```r
props.w$unstruct <- props.w$struct <- 1:n.area
formula = logit.p ~ 1 +
        f(struct,model='besag',
          adjust.for.con.comp=TRUE,
          constr=TRUE,graph=mat,
          scale.model = TRUE,
          param = c(0.5, 0.0015))+
        f(unstruct, model='iid', param=c(0.5,0.0015))
fit.weighted <- inla(formula,
        family="gaussian", data=props.w,
        control.predictor = list(compute = TRUE),
        control.family = list(hyper = list(prec = list(
         initial = log(1), fixed=TRUE))),
        scale=props.w$logit.prec)
```

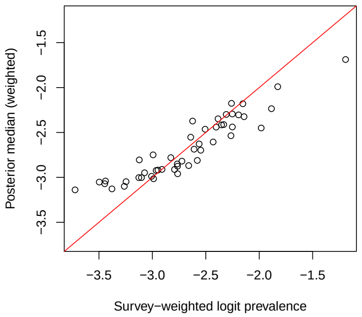
Weighted and smoothed model: organize results

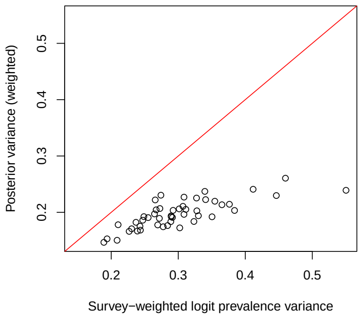
```r
props.wsmooth <- props.w
expit <- function(x) {
    exp(x)/(1 + exp(x))
}
# posterior median
props.wsmooth[, "p.hat"] <- expit(fit.weighted$summary.fitted.values[,
    "0.5quant"])
props.wsmooth[, "logit.p"] <- fit.weighted$summary.fitted.values[,
    "0.5quant"]
# posterior standard deviations
props.wsmooth[, "se.p.hat"] <- NA
props.wsmooth[, "logit.v"] <- fit.weighted$summary.fitted.values[,
    "sd"]^2
props.wsmooth[, "logit.prec"] <- 1/props.wsmooth[,
    "logit.v"]
# Post medians of unstructured random
# effects
props.wsmooth[, "unstruct"] <- fit.weighted$summary.random$unstruct[,
    "0.5quant"]
# Post medians of spatial random effects
props.wsmooth[, "struct"] <- fit.weighted$summary.random$struct[,
    "0.5quant"]
```

Weighted and smoothed model: compare with weighted

```
par(mfrow = c(1, 2))
lim <- range(c(props.w[, "logit.p"], props.wsmooth[,
    "logit.p"]))
plot(props.w[, "logit.p"], props.wsmooth[,
    "logit.p"], xlim = lim, ylim = lim, xlab = "Survey-weighted logit p
    ylab = "Posterior median (weighted)")
abline(c(0, 1), col = "red")
lim <- range(c(props.w[, "logit.v"]^0.5,
    props.wsmooth[, "logit.v"]^0.5))
plot(props.w[, "logit.v"]^0.5, props.wsmooth[,
    "logit.v"]^0.5, xlim = lim, ylim = lim,
    xlab = "Survey-weighted logit prevalence variance",
    ylab = "Posterior variance (weighted)")
abline(c(0, 1), col = "red")
```

Weighted and smoothed model: compare with weighted





Using SUMMER

Weighted and smoothed model: using SUMMER

The SUMMER package has a simple function `fitSpace()` to estimate weighted and smoothed estimates

```r
library(SUMMER)
fit <- fitSpace(data = kingdata, geo = kingshape,
    Amat = mat, family = "binomial", responseVar = "diab2",
    strataVar = "strata", weightVar = "rwt_llcp",
    regionVar = "hracode", clusterVar = "~1",
    hyper = NULL, CI = 0.95)
```

SUMMER: default hyperpriors

For binary variables, the default hyperpriors for precisions is Gamma(0.5, 0.001488), which leads to a 95% prior interval for the residual odds ratio between $[0.5, 2]$, i.e., for

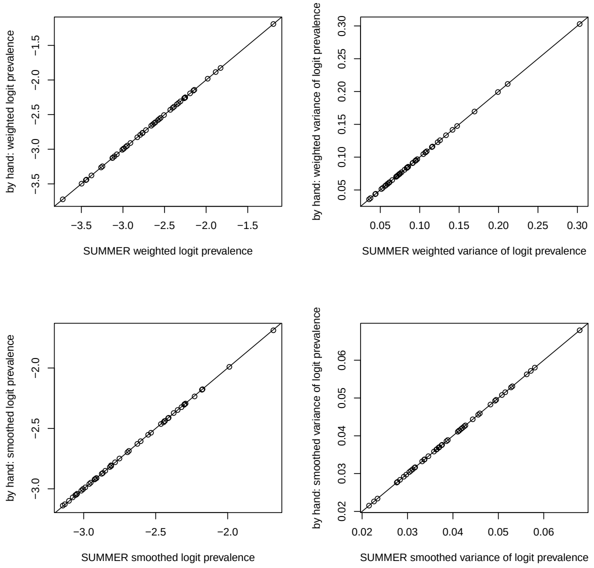
$$u|\tau \sim_{iid} \text{Normal}(0, 1/\tau), \quad \tau \sim \text{Gamma}(0.5, 0.001488)$$

The $[0.025\%, 0.975\%]$ quantiles are roughly $[0.5, 2]$. See Section 9.6.2 of Wakefield (2013) for more details.
The structured effects are all scaled to have unit generalized marginal variance, so that the precision parameter has the similar interpretation. See `https://www.math.ntnu.no/inla/r-inla.org/tutorials/inla/scale.model/scale-model-tutorial.pdf` for more details about the scaled models.

Comparing hand-coded results and SUMMER fit

```
par(mfrow = c(2, 2))
plot(fit$HT$HT.est, props.w$logit.p, xlab = "SUMMER weighted log
    ylab = "by hand: weighted logit prevalence")
abline(c(0, 1))
plot(fit$HT$HT.variance, props.w$logit.v,
    xlab = "SUMMER weighted variance of logit prevalence",
    ylab = "by hand: weighted variance of logit prevalence")
abline(c(0, 1))
plot(fit$smooth$mean, props.wsmooth$logit.p,
    xlab = "SUMMER smoothed logit prevalence",
    ylab = "by hand: smoothed logit prevalence")
abline(c(0, 1))
plot(fit$smooth$sd^2, props.wsmooth$logit.v,
    xlab = "SUMMER smoothed variance of logit prevalence",
    ylab = "by hand: smoothed variance of logit prevalence")
abline(c(0, 1))
```

Comparing hand-coded results and SUMMER fit



SUMMER fit

The fit object from SUMMER package contains

- `HT`
  - `HT.est.original`, `HT.variance.original`: mean, and sd of the direct estimates accounting for survey weights.
  - `HT.est`, `HT.sd`, `HT.variance`, `HT.prec`: mean, and asymptotic sd, variance and precision of the logit transformed direct estimates accounting for survey weights.
- `smooth`
  - `mean.trans`, `sd.trans`, `median.trans`, `lower.trans`, `upper.trans`: mean, sd, median, posterior credible intervals (specified by `CI` argument in function call) of the posterior prediction in the probability scale.
  - `mean`, `sd`, `median`, `lower`, `upper`: mean, sd, median, posterior credible intervals (specified by `CI` argument in function call) of the posterior prediction in logit scale.

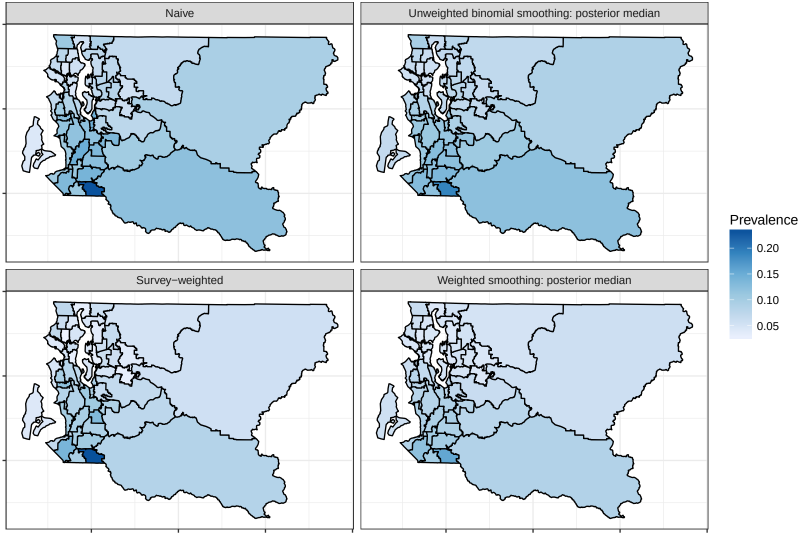
Easier visualization: merge all results

```r
combined <- merge(fit$HT, fit$smooth, by = "region")
combined <- combined[match(props$hracode,
    combined$region), ]
combined$HT.sd.original <- combined$HT.variance.original^0.5
# naive
combined$p.hat.naive <- props$p.hat
combined$se.p.hat.naive <- props$se.p.hat
# binomial smoothing
combined$p.hat.binomial <- props.smooth$p.hat
combined$se.p.hat.binomial <- props.smooth$se.p.hat
```

Easier visualization

```
g <- mapPlot(data = combined, geo = kingshape,
    variables = c("p.hat.naive", "p.hat.binomial",
        "HT.est.original", "median.trans"),
    labels = c("Naive", "Unweighted binomial smoothing: posterio
        "Survey-weighted", "Weighted smoothing: posterior median
    by.data = "region", by.geo = "HRA2010v2_",
    is.long = FALSE)
g <- g + theme_bw() + theme(axis.title = element_blank(),
    axis.text = element_blank())
g <- g + scale_fill_distiller("Prevalence",
    direction = 1)
g
```

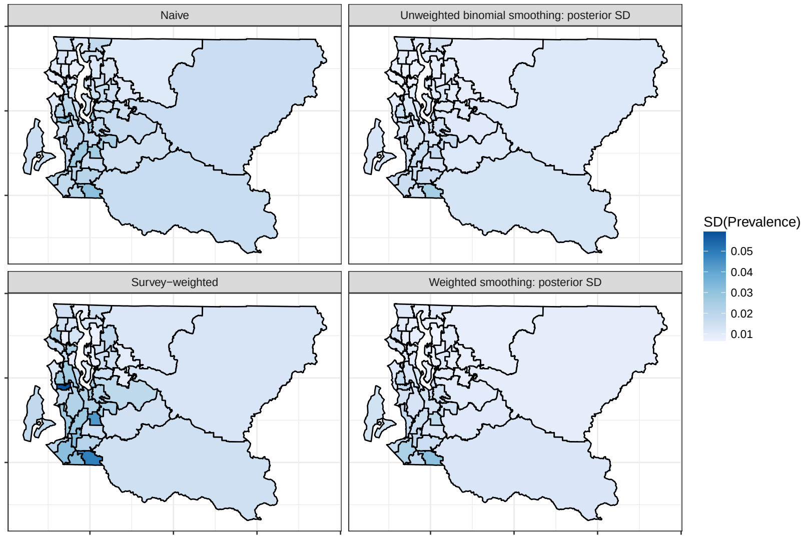
Easier visualization


Naive
Unweighted binomial smoothing: posterior median
Survey−weighted
Weighted smoothing: posterior median
Prevalence
0.20
0.15
0.10
0.05

Easier visualization

```r
g <- mapPlot(data = combined, geo = kingshape,
    variables = c("se.p.hat.naive", "se.p.hat.binomial",
        "HT.sd.original", "sd.trans"), labels = c("Naive",
        "Unweighted binomial smoothing: posterior SD",
        "Survey-weighted", "Weighted smoothing: posterior SD"),
    by.data = "region", by.geo = "HRA2010v2_",
    is.long = FALSE)
g <- g + theme_bw() + theme(axis.title = element_blank(),
    axis.text = element_blank())
g <- g + scale_fill_distiller("SD(Prevalence)",
    direction = 1)
g
```

Easier visualization



Naive
Unweighted binomial smoothing: posterior SD
Survey−weighted
Weighted smoothing: posterior SD
SD(Prevalence)
0.05
0.04
0.03
0.02
0.01

Conclusion

The last two plots illustrate the effect of the Bayesian smoothing model:

- the estimates are shrunk (both globally and locally), this introduces bias,
- the uncertainty is in general reduced, due to the use of all the data.

Overall:

- It is clear we need to consider the weighting
- The smoothing does increase precision, at the expense of a little bias