SISMID Spatial Statistics in Epidemiology and Public Health
2016 R Notes: Small Area Estimation

Jon Wakefield
Departments of Statistics, University of Washington

2016-07-03

Small Area Estimation (SAE)

These notes were 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.

Overview of analyses

We present results from the following analyses:

- Naive (ie unweighted, unsmoothed)
- Binomial spatial smoothing model, ignoring weighting
- Weighted (unsmoothed)
- Smoothed and weighted

Read in Data

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

```
library(foreign)
library(SpatialEpi)
kingdata <- read.dta("ct0913all.dta")
names(kingdata)
##  [1] "age"       "pracex"    "educau"    "zipcode"   "sex"       "street1"
##  [7] "street2"   "seqno"     "year"      "hispanic"  "mracex"    "_ststr"
## [13] "hracode"   "tract"     "rwt_llcp"  "genhlth2"  "fmd"       "obese"
## [19] "smoker1"   "diab2"     "aceindx2"  "zipout"    "streetx"   "ethn"
## [25] "age4"      "ctmiss"
```

There are 16283 observations, i.e., individuals in the sample. These data were collected over the period 2009-2013.

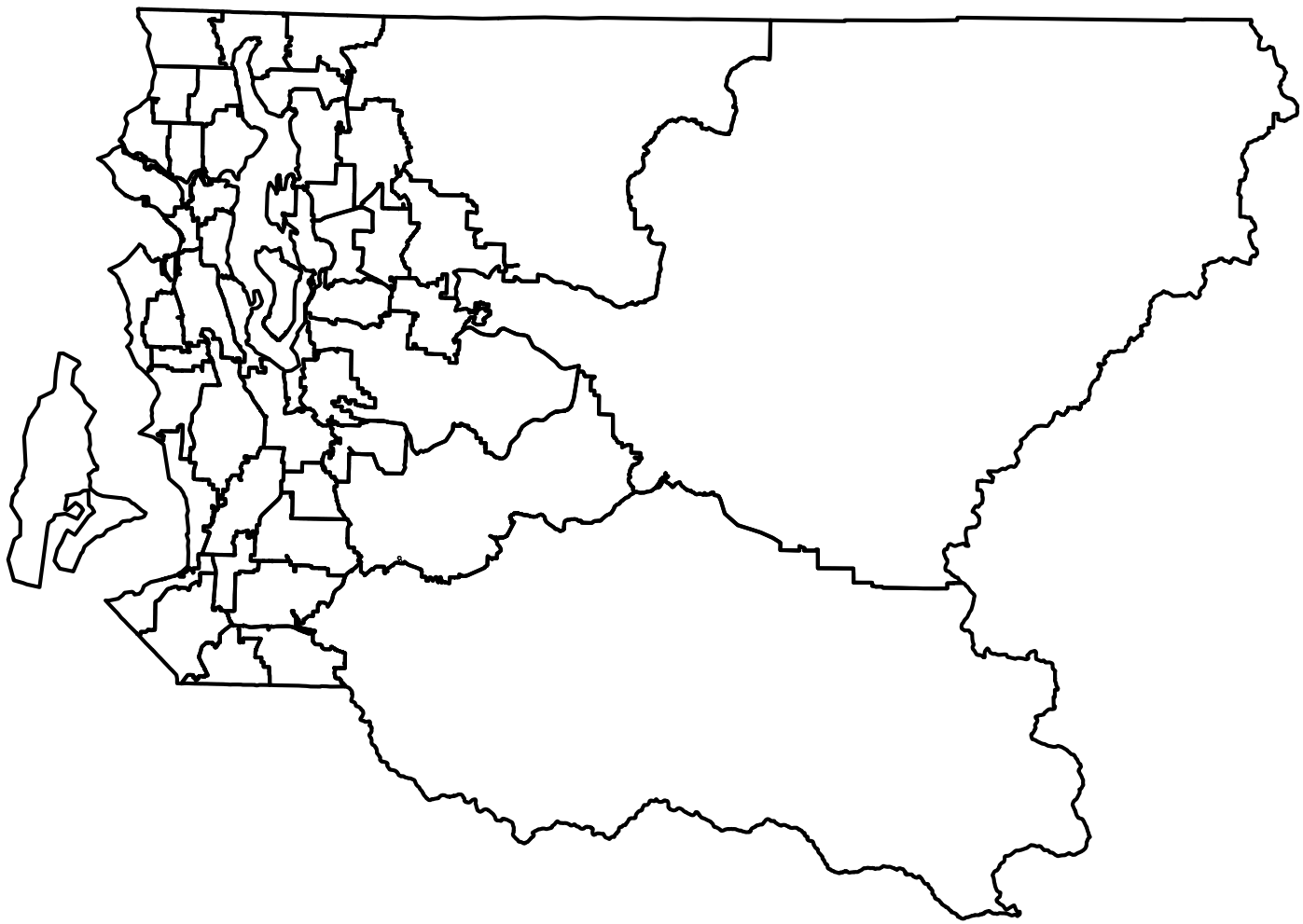
Read in Data

Next, read in the shape files for King County HRAs using the `rgdal` package.

```
library(rgdal)
kingshape <- readOGR("HRA_ShapeFiles", layer = "HRA_2010Block_Cl
## OGR data source with driver: ESRI Shapefile
## Source: "HRA_ShapeFiles", layer: "HRA_2010Block_Clip"
## with 48 features
## It has 9 fields
names(kingshape)
## [1] "FID_HRA_20" "HRA2010v2_" "SUM_plibra" "FID_kc_bor" "COUN
## [6] "CNTYN"      "STATE"      "CNTY"       "FIPS"
```

The study region with HRAs

```
plot(kingshape)
```



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.

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

There are 48 HRAs and we are left with 16124 observations.

Naive estimates

Let y_i and m_i be the number of individuals flagged as having type II diabetes and the denominators in the $i = 1, \ldots, n$ 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 estimates

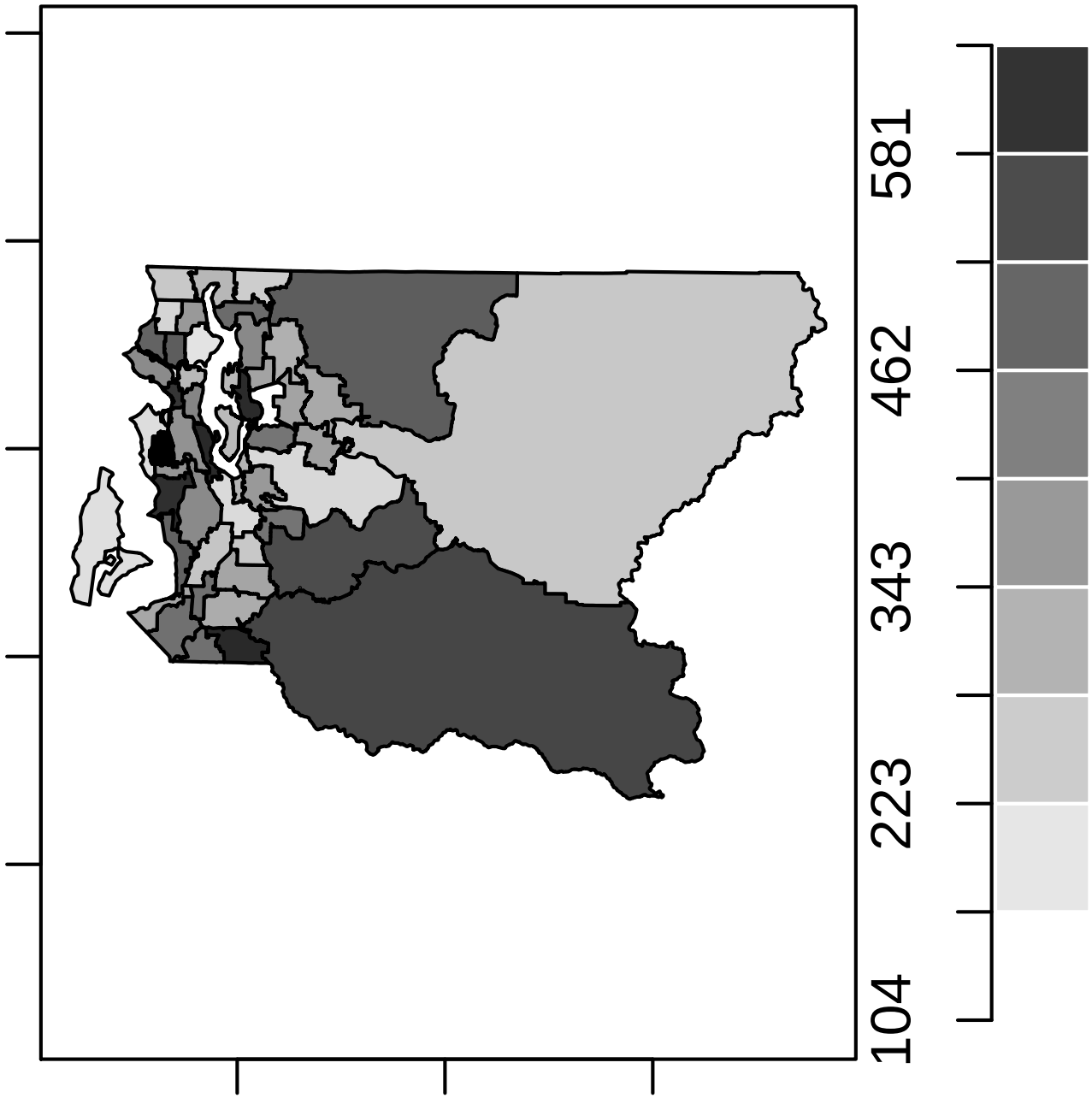
```
hras <- as.character(unique(kingdata$hracode))
props <- matrix(NA, nrow = n.area, ncol = 5)
props <- as.data.frame(props)
colnames(props) <- c("hracode", "p.hat",
    "se.p.hat", "y.i", "n.i")
props[, 1] <- hras
for (i in 1:n.area) {
    props[i, "p.hat"] <- mean(kingdata[kingdata$hracode ==
        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)
}
```

Mapping of sample sizes

We map the number of individuals who answered the diabetes question in each HRA.

```
kingshapepoly <- SpatialPolygons(kingshape@polygons,
    proj4string = kingshape@proj4string)
summary(props[, "n.i"])
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   104.0   244.0   319.0   335.9   420.0   641.0
par(mar = c(1, 1, 1, 1))
mapvariable(props[, "n.i"], kingshapepoly, ncut = 1000,
    nlevels = 10)
```

Mapping of sample sizes

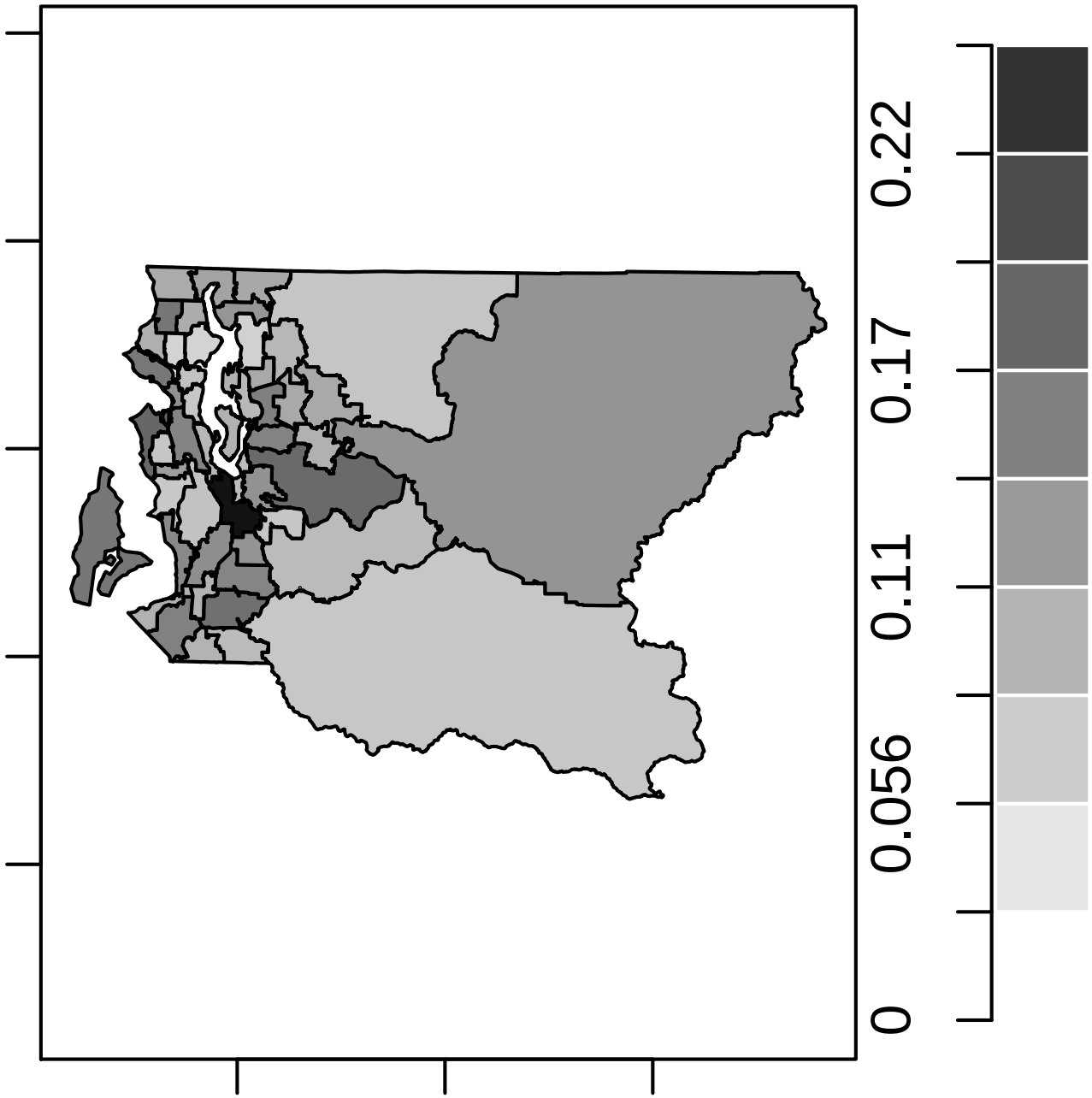


Mapping of naive estimates

Map y_i/n_i, $i = 1, \ldots, 48$.

```
par(mar = c(1, 1, 1, 1))
mapvariable(props[, "p.hat"], kingshapepoly,
    ncut = 1000, nlevels = 10, lower = 0,
    upper = 0.25)
```

Mapping of naive estimates



Naive binomial model

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

$$
\begin{aligned}
y_i | p_i &\sim \text{Binomial}(N_i, p_i) \\
\theta_i &= \log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \epsilon_i + S_i, \\
\epsilon_i &\sim N(0, \sigma_\epsilon^2) \\
S_i | S_j, j \in \text{ne}(i) &\sim N\left(\bar{S}_j, \frac{\sigma_s^2}{m_i}\right).
\end{aligned}
$$

With priors on $\beta_0, \sigma_\epsilon^2, \sigma_s^2$.

Create .graph file for spatial model INLA implementation

```
library(spdep)
king.neigh <- poly2nb(kingshapepoly)
library(INLA)
nb2INLA("HRA_Shapefiles/KingCoNb.graph",
    king.neigh)
```

Naive binomial model

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

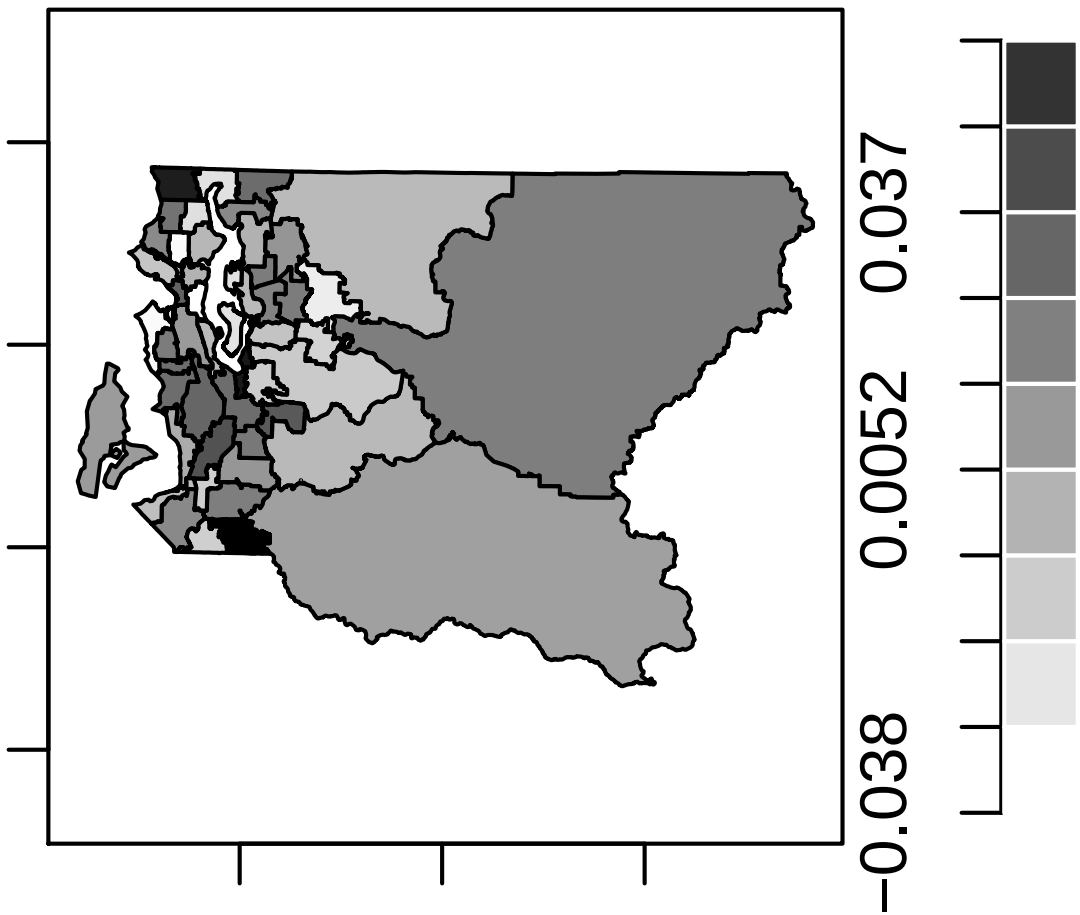
```
props <- props[order(props$hracode), ]
props$unstruct <- props$struct <- 1:n.area
formula = y.i ~ 1 + f(struct, model = "besag",
    adjust.for.con.comp = TRUE, constr = TRUE,
    graph = "HRA_Shapefiles/KingCoNb.graph") +
    f(unstruct, model = "iid", param = c(0.5,
        0.008))
mod.smooth.unweighted <- inla(formula, family = "binomial",
    data = props, Ntrials = n.i, control.predictor = list(comput
```

Naive binomial model

```
# Post medians of prevalences
psmoothunwt <- mod.smooth.unweighted$summary.fitted.values[,
    "0.5quant"]
# Post standard deviations of prevalences
psmoothunwtsd <- mod.smooth.unweighted$summary.fitted.values[,
    "sd"]
# Post medians of unstructured random
# effects
unwtunstruct <- mod.smooth.unweighted$summary.random$unstruct[,
    "0.5quant"]
# Post medians of spatial random effects
unwtstruct <- mod.smooth.unweighted$summary.random$struct[,
    "0.5quant"]
```

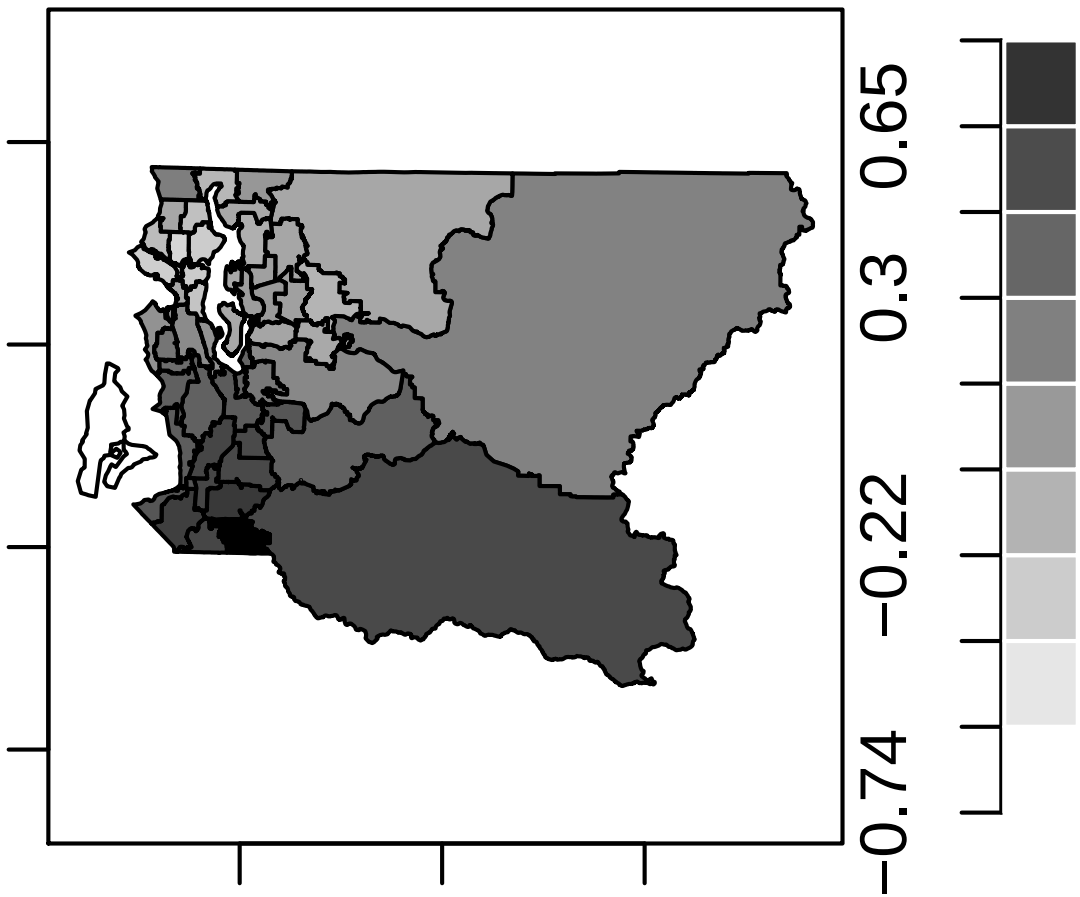
Unstructured random effects

```
par(mar = c(1, 1, 1, 1))
mapvariable(unwtunstruct, kingshapepoly, ncut = 1000,
    nlevels = 10)
```



Structured (spatial) random effects

```
par(mar = c(1, 1, 1, 1))
mapvariable(unwtstruct, kingshapepoly, ncut = 1000,
    nlevels = 10)
```



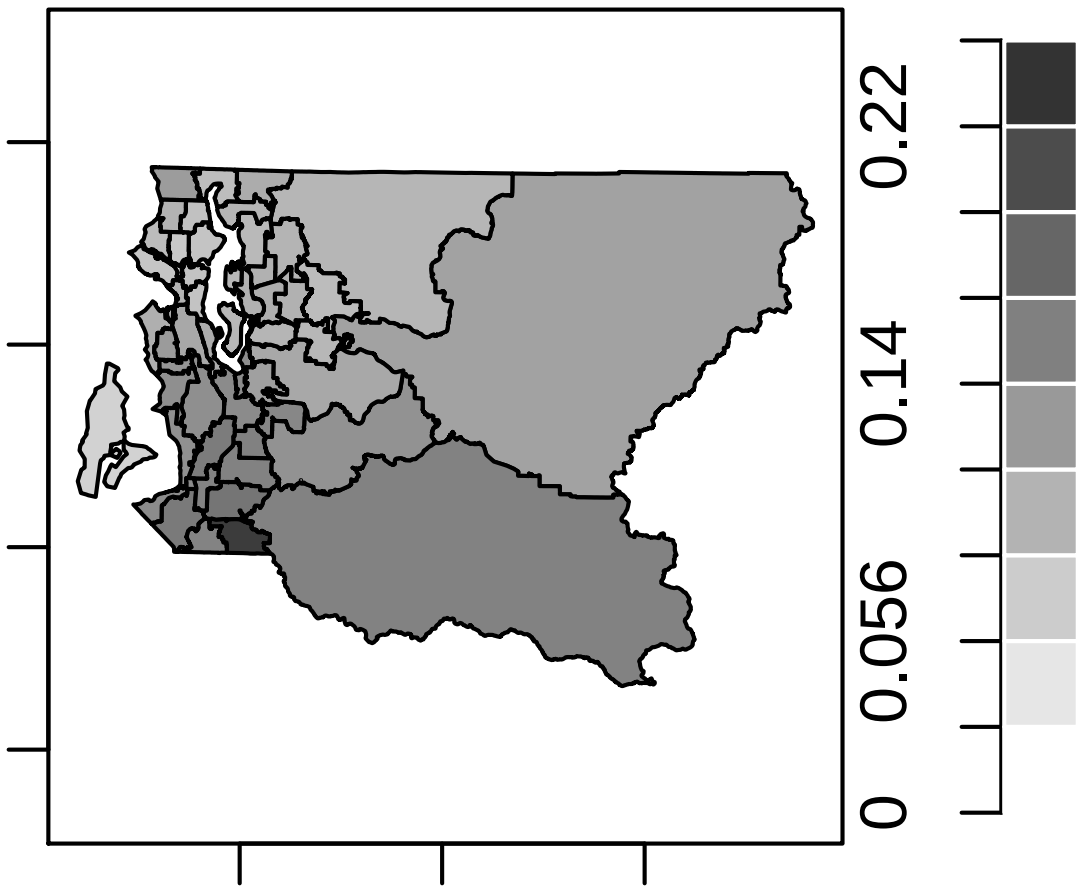
Proportion of variation that is spatial

```
nareas <- 48
mat.marg <- matrix(NA, nrow = nareas, ncol = 1000)
m <- mod.smooth.unweighted$marginals.random$struct
for (i in 1:nareas) {
    Sre <- m[[i]]
    mat.marg[i, ] <- inla.rmarginal(1000, Sre)
}
var.Sre <- apply(mat.marg, 2, var)
var.eps <- inla.rmarginal(1000, inla.tmarginal(function(x) 1/x,
    mod.smooth.unweighted$marginals.hyper$"Precision for unstruc
mean(var.Sre)
## [1] 0.1276691
mean(var.eps)
## [1] 0.01022552
perc.var.Sre <- mean(var.Sre/(var.Sre + var.eps))
```

Percentage variability that is spatial is 93%.

Predicted Prevalence: Rates higher in the South of KC

```
library(SpatialEpi)
par(mar = c(1, 1, 1, 1))
mapvariable(psmoothunwt, kingshapepoly, ncut = 1000,
    nlevels = 10, lower = 0, upper = 0.25)
```



Comparison of estimates

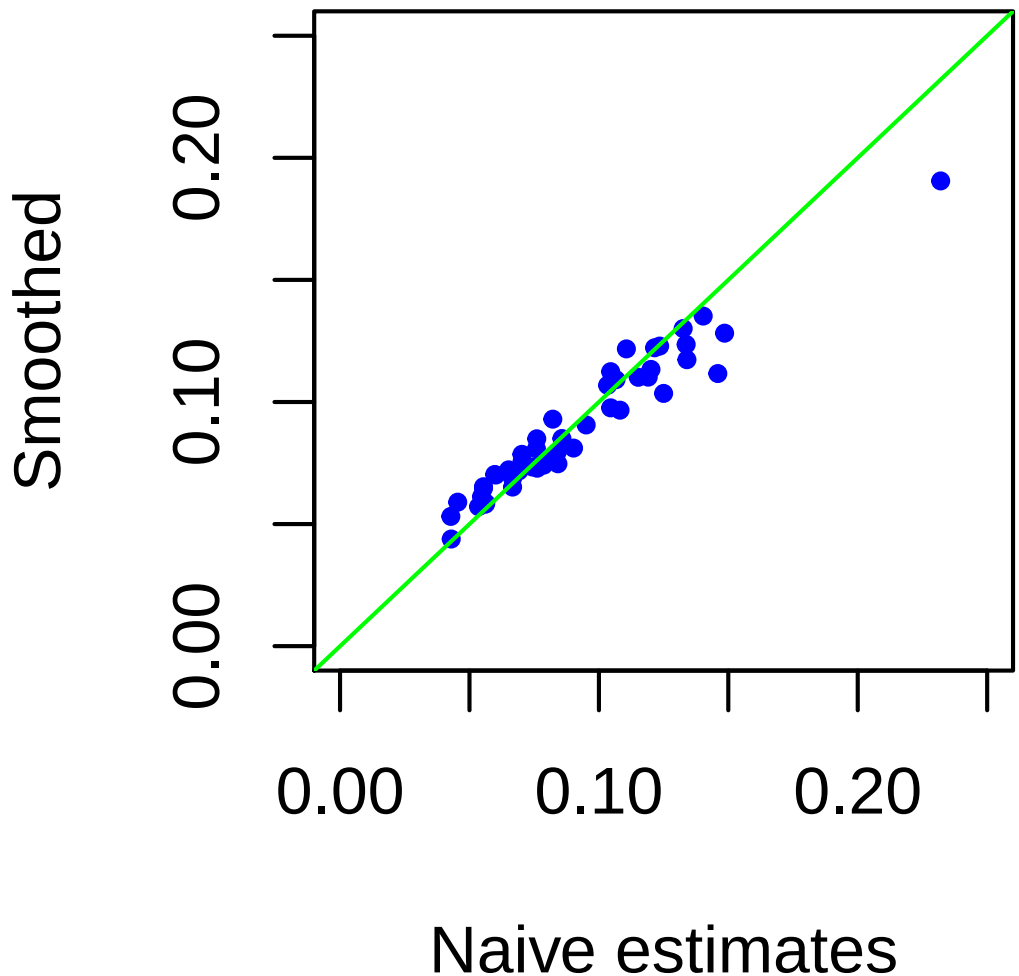
We plot the smoothed estimates versus the naive estimates

There is little smoothing here, as the within HCA sample sizes are relatively large.

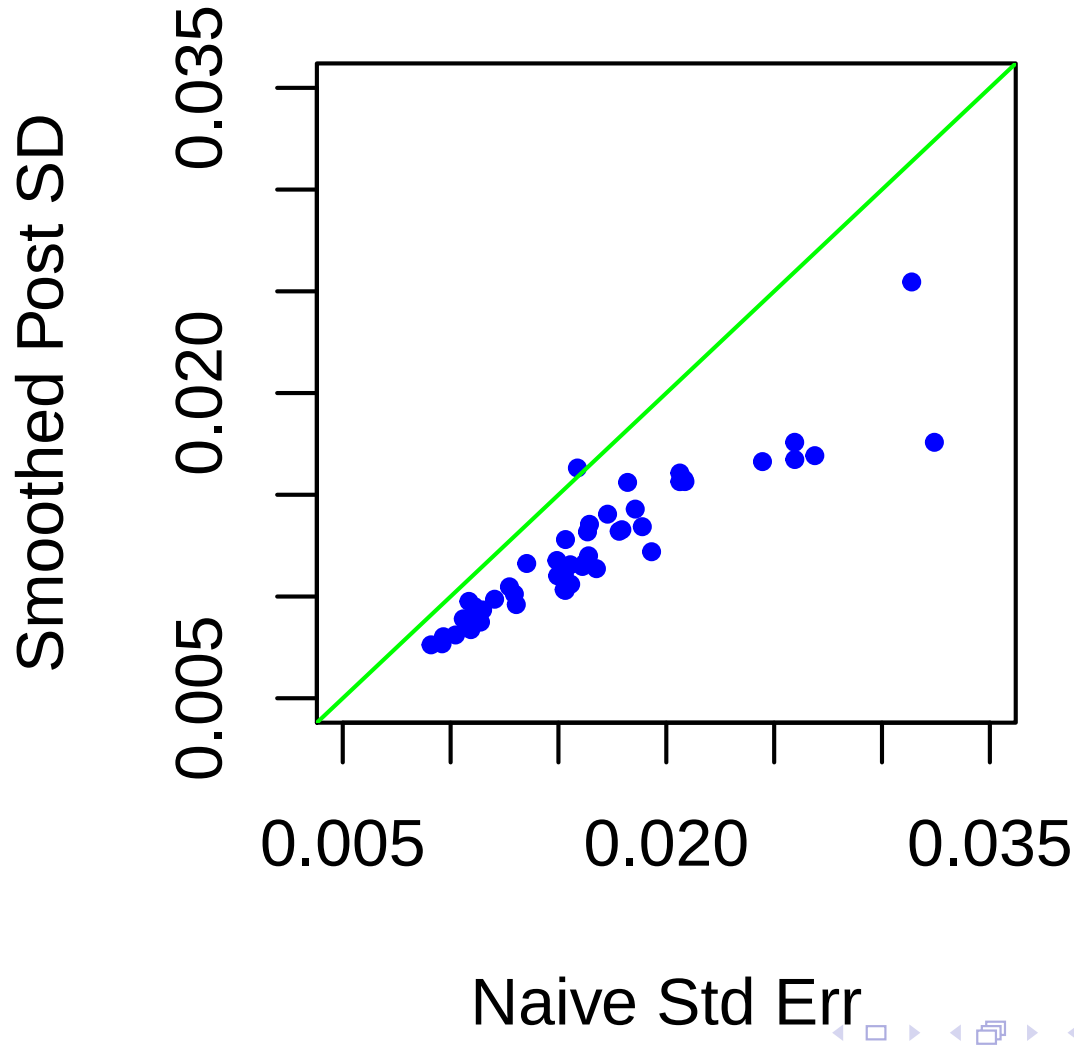
We also plot the posterior standard deviations against the standard errors and see that the former are a little smaller, reflecting the use of all the data.

```
summary(props[, "p.hat"])
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.04279 0.06628 0.08305 0.09175 0.11620 0.23200
summary(psmoothunwt)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.04389 0.07024 0.08097 0.09007 0.11060 0.19050
```

```
plot(psmoothunwt ~ props[, "p.hat"], pch = 19, xlim = c(0,
    0.25), ylim = c(0, 0.25), col = "blue", cex = 0.5,
    xlab = "Naive estimates", ylab = "Smoothed")
abline(0, 1, col = "green")
```



```r
plot(psmoothunwtsd ~ props[, "se.p.hat"], pch = 19,
    xlim = c(0.005, 0.035), ylim = c(0.005, 0.035),
    col = "blue", cex = 0.5, xlab = "Naive Std Err",
    ylab = "Smoothed Post SD")
abline(0, 1, col = "green")
```



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.

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.

Weights

Using the `survey` package, we can get make weighted design-based inference for the proportion in each small area with Type II diabetes.

We need to account for the probability that each person selected in our survey would be selected given the sampling scheme.

`svydesign` will allow us to specify the sampling scheme. The `_ststr` variable we renamed `strata` represents the strata.

The survey weights can be found in the `rwt_llcp` variable. These weights are the products of the design weights and the raking weights.

The function `svyby` allows us to compute the survey-weighted mean of the `diab2` variable for small areas indexed by `hracode`.

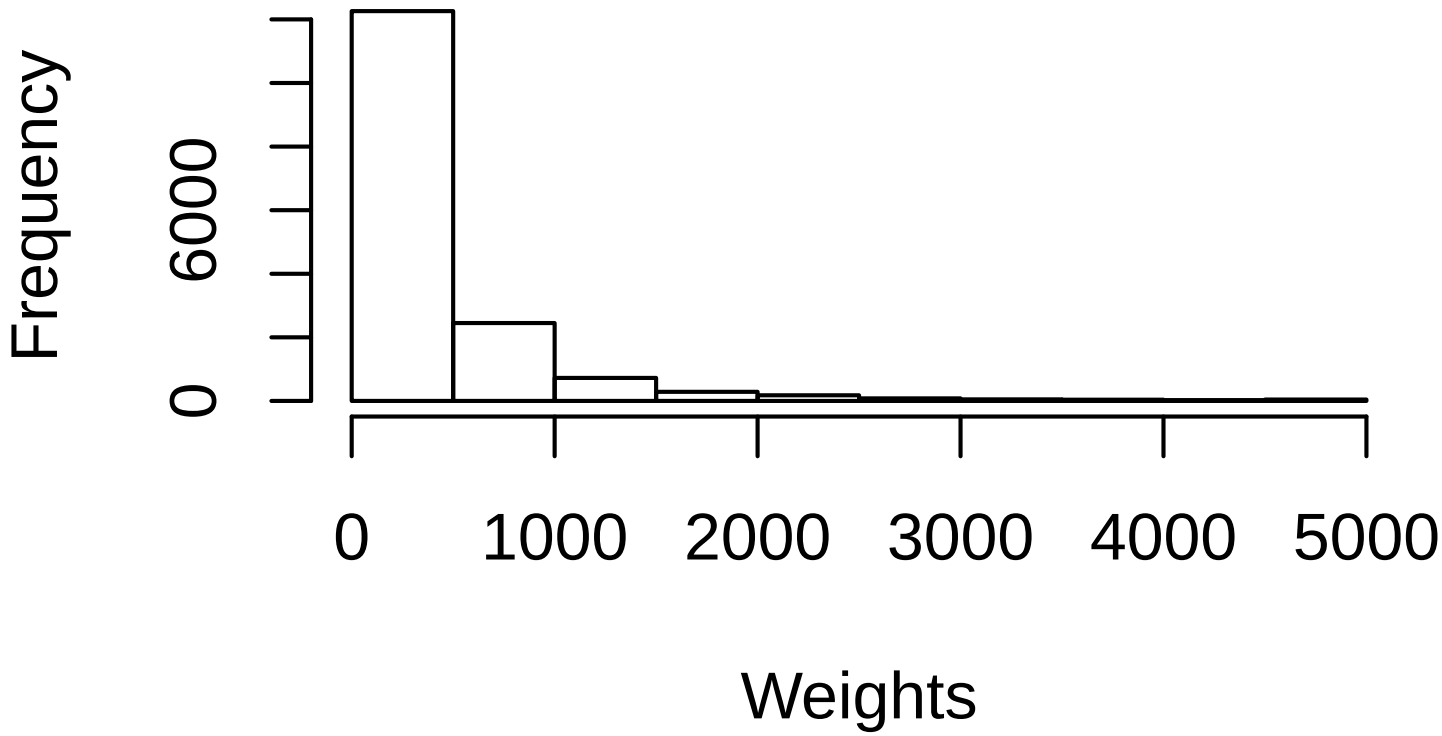
Weights summary

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, $CV^2/(CV^2+1)$ approximates the inefficiency of using the weights

```
# Coefficient of variation of weights
cv <- sqrt(var(kingdata$rwt_llcp, na.rm = T))/mean(kingdata$rwt_
    na.rm = T)
cv^2/(cv^2 + 1)
## [1] 0.6019036
summary(kingdata$rwt_llcp)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    22.8   159.2   261.7   437.4   485.1  5000.0
hist(kingdata$rwt_llcp, xlab = "Weights", main = "")
```

Histogram of weights



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:

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

Calculate weighted means and design-based variances

```
library(survey)
kingcounty.des <- svydesign(ids = ~1, weights = ~rwt_llcp,
    strata = ~strata, data = kingdata)
p.i <- svyby(~diab2, ~hracode, kingcounty.des,
    svymean)$diab2
dv.i <- svyby(~diab2, ~hracode, kingcounty.des,
    svymean)$se^2
logit.pi <- log(p.i/(1 - p.i))
v.i <- dv.i/(p.i^2 * (1 - p.i)^2)
```

We obtain

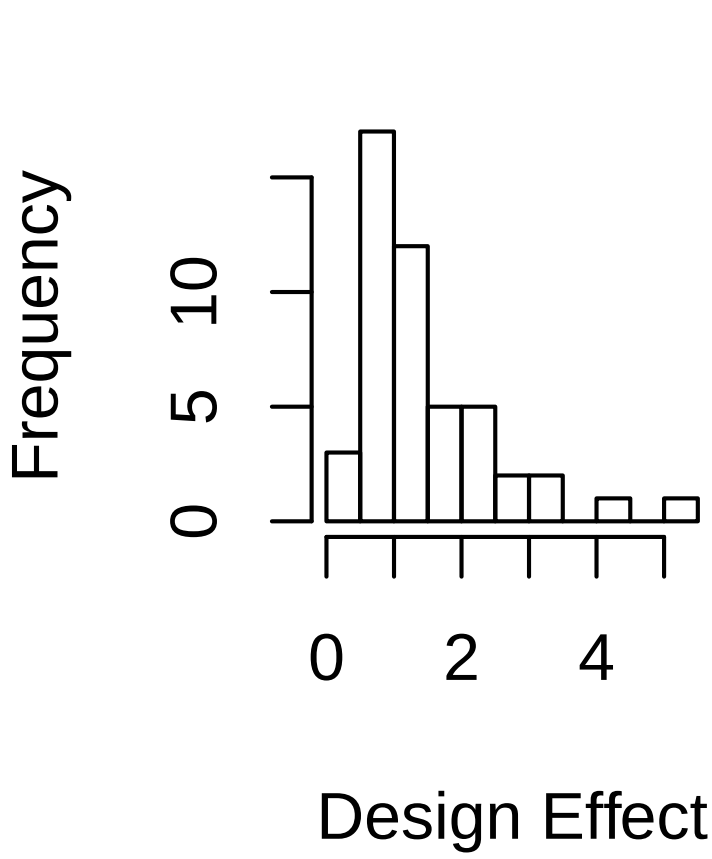
- The weighted estimators of prevalences `p.i`
- The design variances of prevalences `dv.i`
- The weighted estimators of logits of prevalences `logit.p.i`
- The design variances of logits of prevalences `v.i`

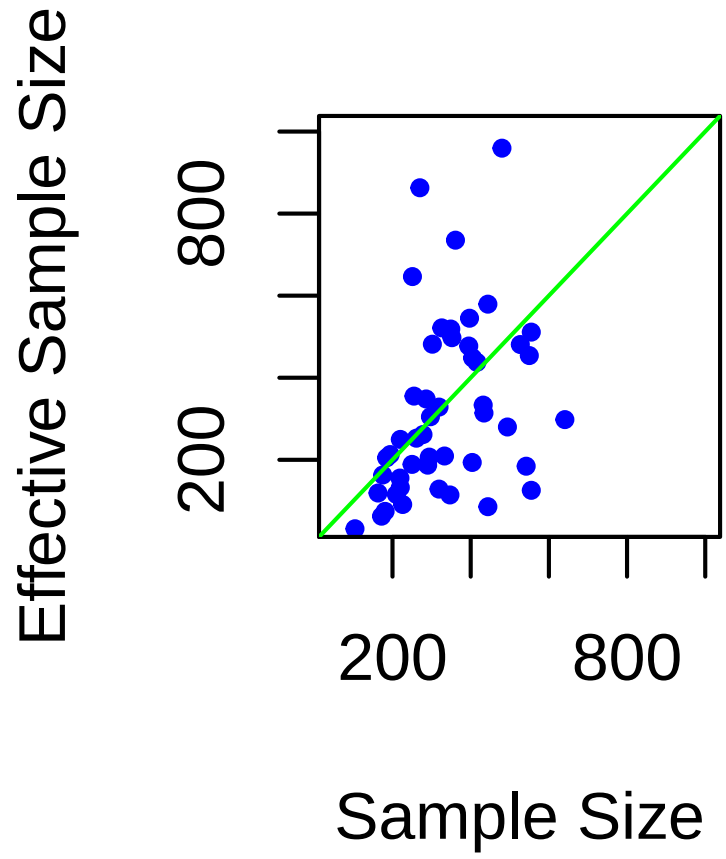
Design effects

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")
plot(effss ~ props[, "n.i"], pch = 19, col = "blue",
    cex = 0.5, xlab = "Sample Size", ylab = "Effective Samp
    xlim = c(50, 1000), ylim = c(50, 1000))
abline(0, 1, col = "green")
```

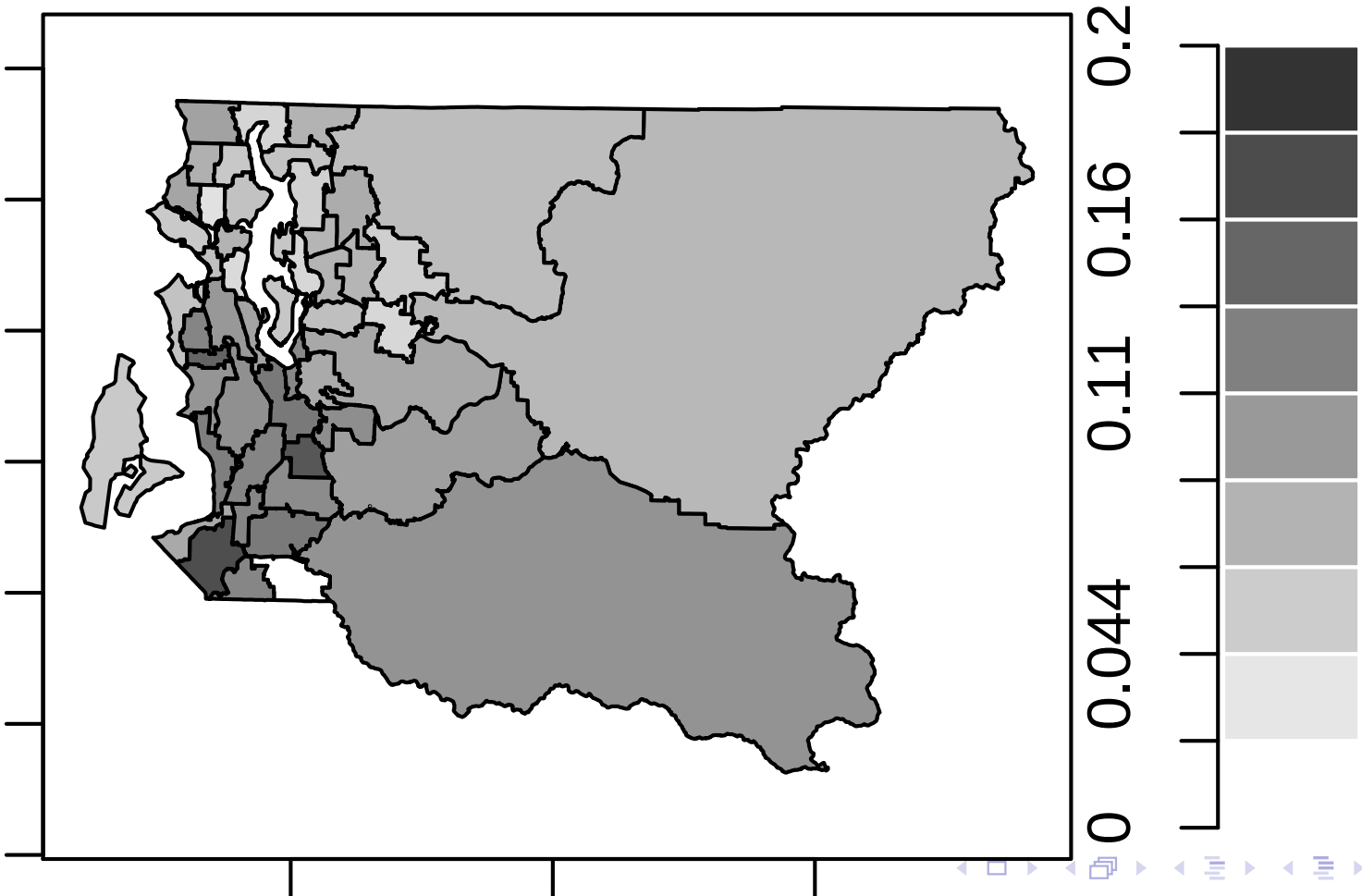

Frequency
0
5
10
0
2
4
Design Effect


Effective Sample Size
200
800
200
800
Sample Size

Weighted estimates

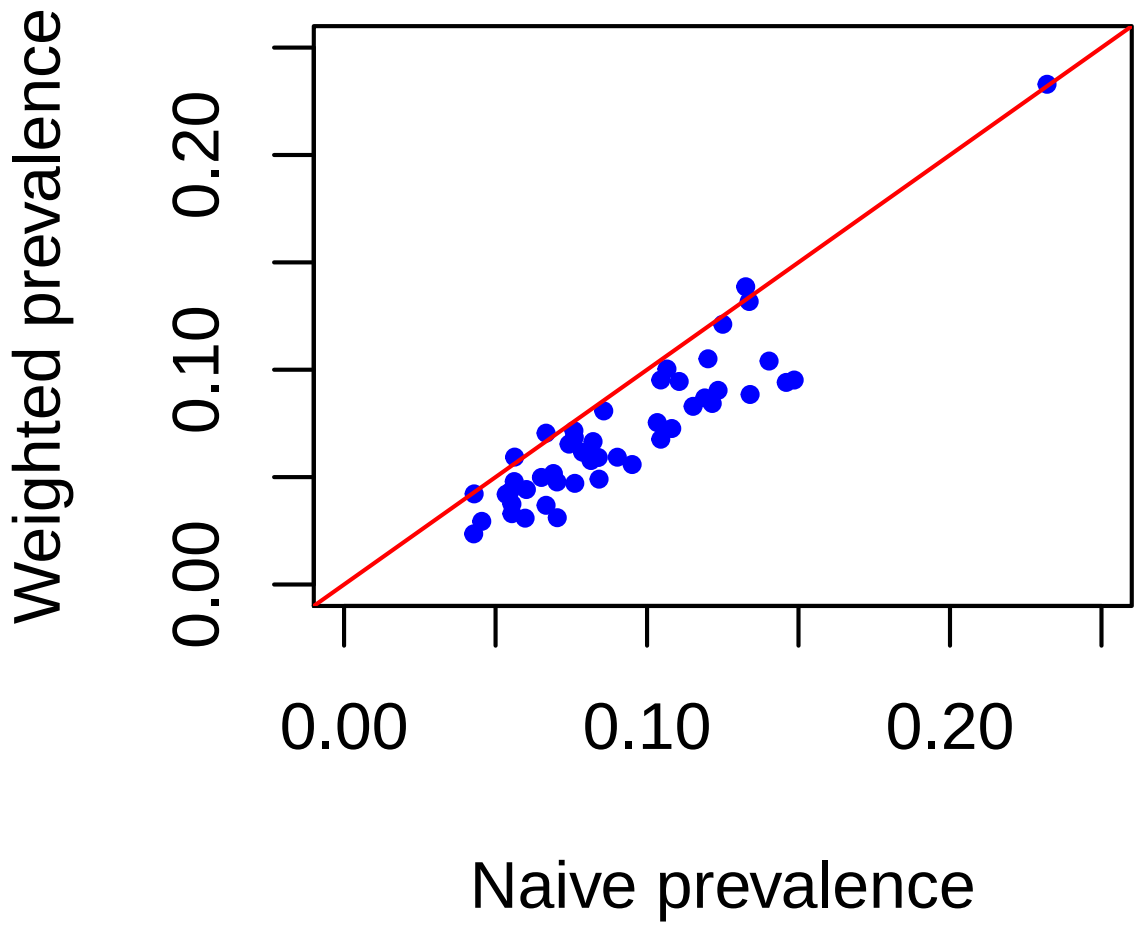
We now map the weighted estimator

```
par(mar = c(1, 1, 1, 1), mfrow = c(1, 1))
mapvariable(p.i, kingshapepoly, ncut = 1000, nlevels = 10,
    lower = 0, upper = 0.2)
```



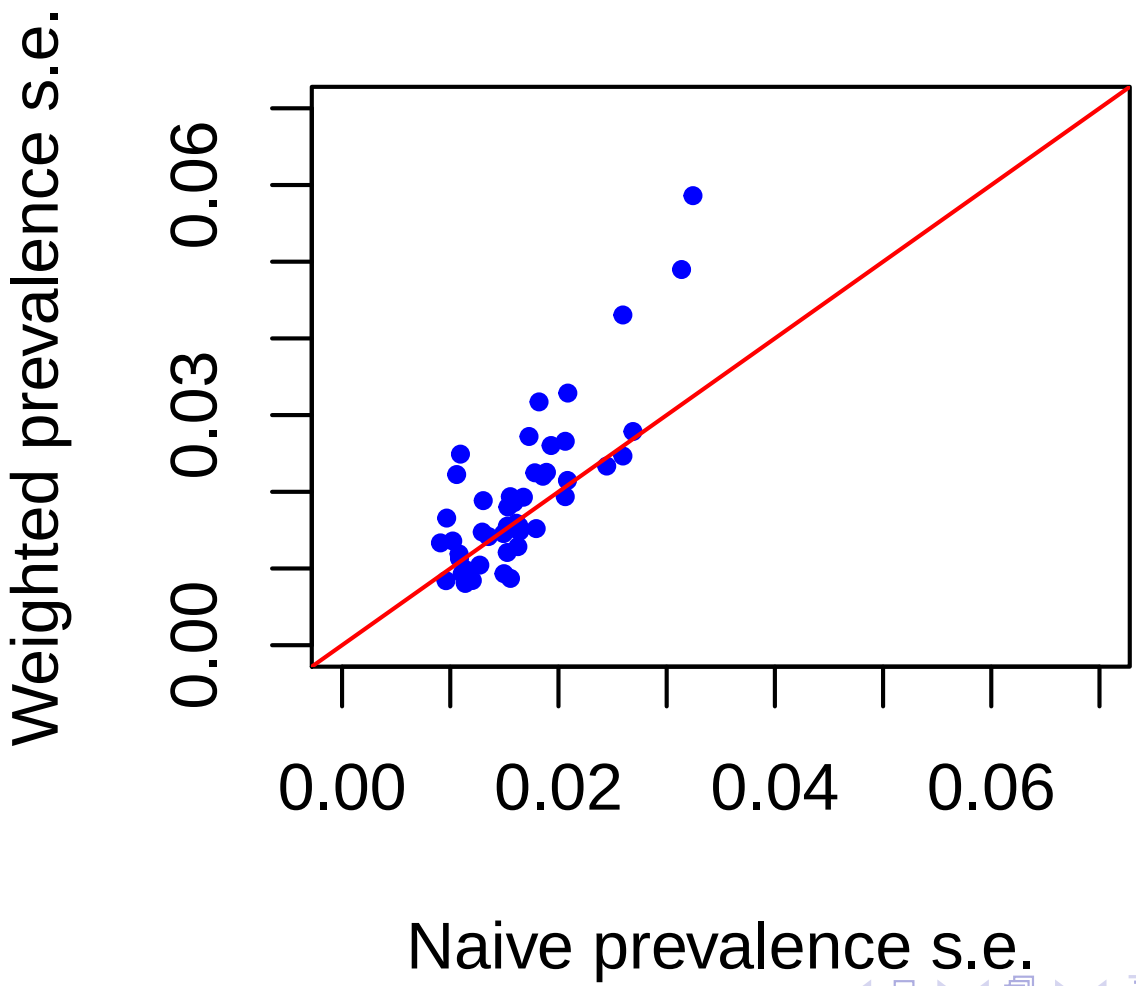
Weighted and naive prevalence estimates

```
plot(p.i ~ props[, "p.hat"], pch = 19, col = "blue",
     cex = 0.5, xlab = "Naive prevalence", ylab = "Weighted preva
     xlim = c(0, 0.25), ylim = c(0, 0.25))
abline(0, 1, col = "red")
```



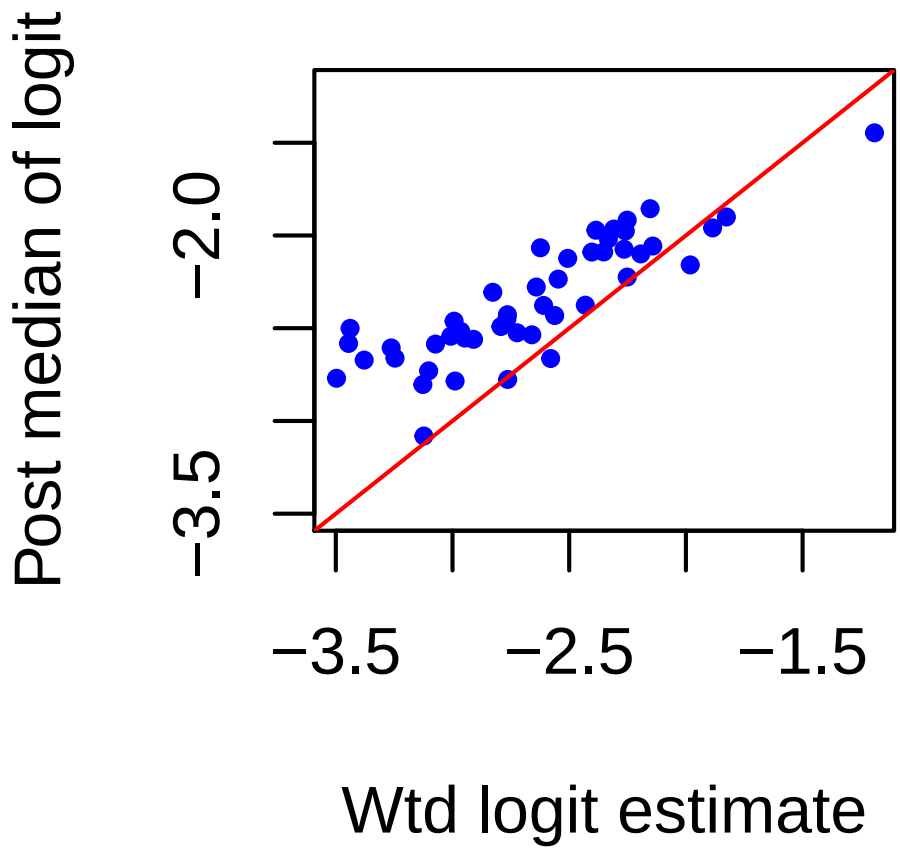
Weighted and naive prevalence standard errors

```
plot(sqrt(dv.i) ~ props[, "se.p.hat"], pch = 19, cex = 0.5,
    col = "blue", xlab = "Naive prevalence s.e.", ylab = "Weight
    xlim = c(0, 0.07), ylim = c(0, 0.07))
abline(0, 1, col = "red")
```



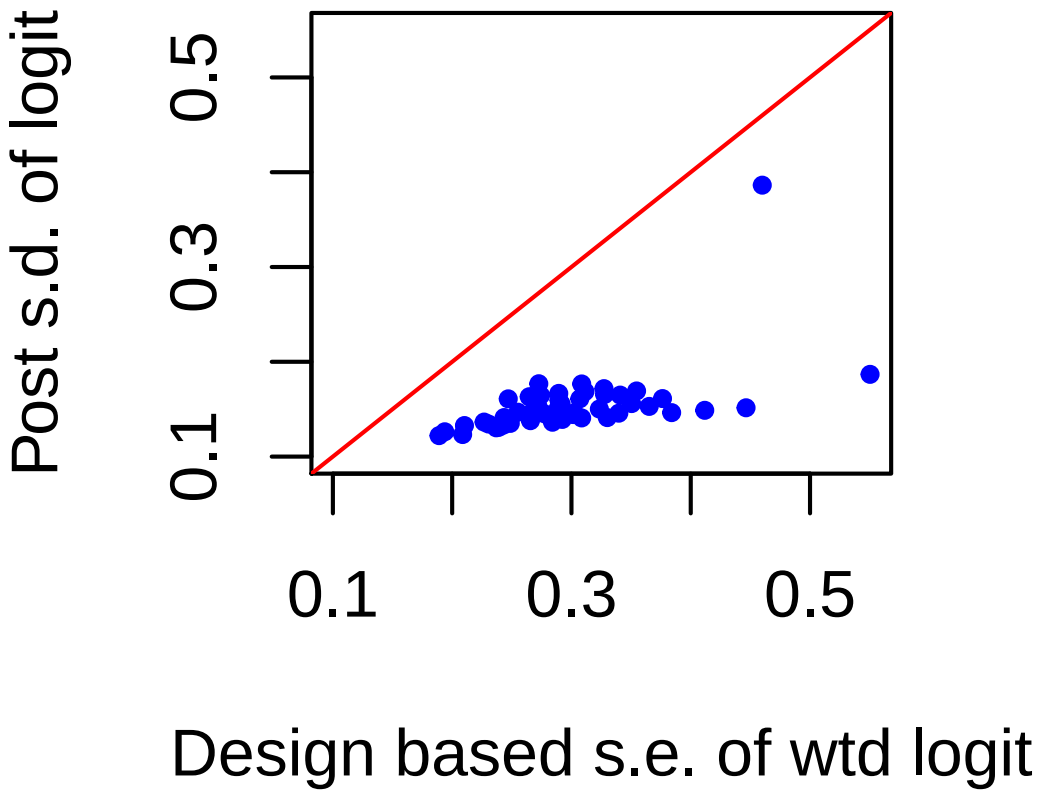
Weighted and naive logits of prevalence estimates

```
plot(mod.smooth.unweighted$summary.linear.predictor[,
    "0.5quant"] ~ logit.pi, pch = 19, col = "blue",
    cex = 0.5, xlab = "Wtd logit estimate", ylab = "Post median of logit",
    xlim = c(-3.5, -1.2), ylim = c(-3.5, -1.2))
abline(a = 0, b = 1, col = "red")
```



Weighted and naive logits of prevalence estimates

```
plot(mod.smooth.unweighted$summary.linear.predictor[,
    "sd"] ~ sqrt(v.i), pch = 19, col = "blue", cex = 0.5,
    xlab = "Design based s.e. of wtd logit", ylab = "Post s.d. of logit",
    xlim = c(0.1, 0.55), ylim = c(0.1, 0.55))
abline(a = 0, b = 1, col = "red")
```



Construct data frame for INLA

```
data <- matrix(NA, nrow = n.area, ncol = 1)
data <- as.data.frame(data)
colnames(data)[1] <- "unstruct"
data$hracode <- props$hracode
data$p.i <- p.i
data$dv.i <- dv.i
data$v.i <- v.i
data$logit.pi <- logit.pi
data$logit.prec <- 1/v.i
data$unstruct <- 1:(n.area)
data$struct <- 1:(n.area)
```

Model Specification

We use the INLA package to fit the following Bayesian hierarchical model (this is an extension of the Fay-Heriott model):

$$
\begin{aligned}
y_i &= \log\left(\frac{\hat{p}_i}{1-\hat{p}_i}\right) \sim N(\theta_i, \hat{V}_i) \\
\theta_i &= \beta + \epsilon_i + S_i, \\
\epsilon_i &\sim N(0, \sigma_\epsilon^2) \\
S_i | S_j, j \in \text{ne}(i) &\sim N\left(\bar{S}_j, \frac{\sigma_s^2}{m_i}\right).
\end{aligned}
$$

With priors on $\beta_0, \sigma_\epsilon^2, \sigma_s^2$.

The key here is that the first stage variance $\hat{V}_i$ is assumed known:

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

Fit global/local spatial smoothing model

```
formula = logit.pi ~ 1 + f(struct, model = "besag",
    adjust.for.con.comp = TRUE, constr = TRUE,
    graph = "HRA_Shapefiles/KingCoNb.graph") +
    f(unstruct, model = "iid", param = c(0.5,
        0.008))
mod.smooth <- inla(formula, family = "gaussian",
    data = data, control.predictor = list(compute = TRUE),
    control.family = list(hyper = list(prec = list(initial = log
        fixed = TRUE))), scale = logit.prec)
```

Results

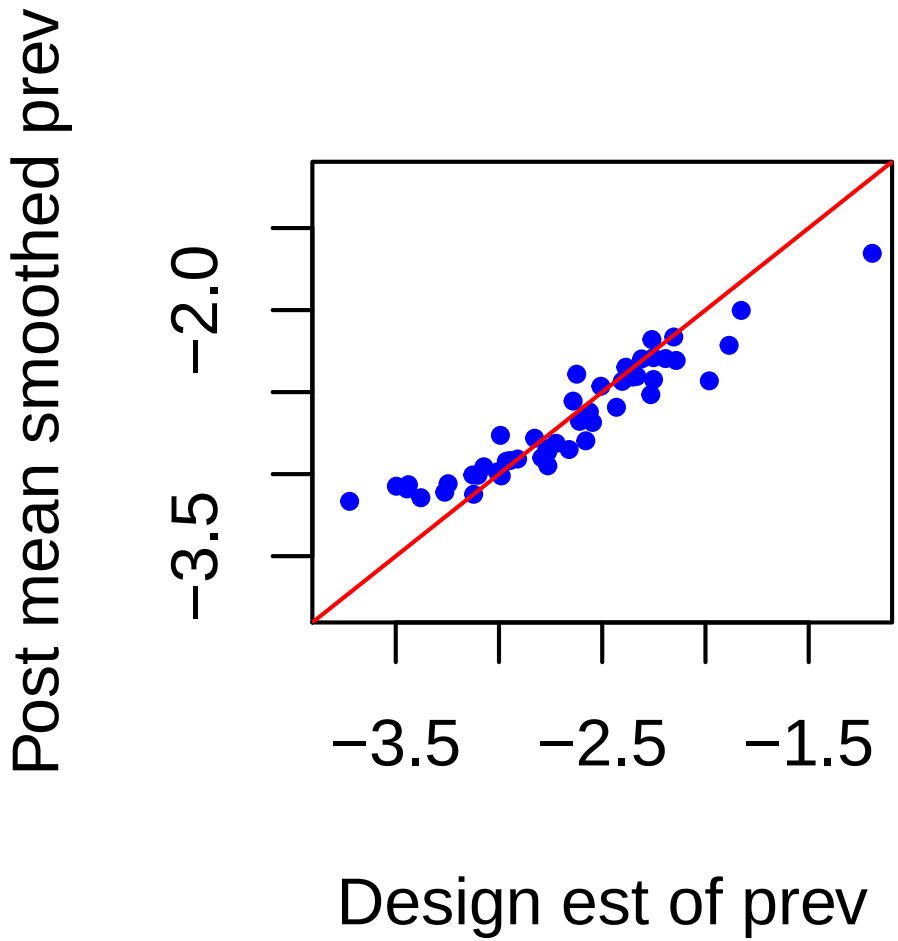
```
mod.smooth$summary.fixed[, c("mean", "0.5quant",
    "sd")]
##                     mean  0.5quant          sd
## (Intercept) -2.666768 -2.666786 0.04576263
mod.smooth$summary.hyperpar[, c("mean", "0.5quant")]
##                                  mean  0.5quant
## Precision for struct     4.619838  4.218685
## Precision for unstruct 125.822003 94.780773
mod.smooth$summary.hyperpar[, c("sd")]
## [1]   2.006049 111.193152
```

Results

```
fixed.med <- rep(mod.smooth$summary.fixed[,
    4], dim(data)[1])
random.iid <- mod.smooth$summary.random$unstruct[,
    5]
random.smooth <- mod.smooth$summary.random$struct[,
    5]
linpred <- mod.smooth$summary.fitted.values[,
    "0.5quant"]
pred <- exp(linpred)/(1 + exp(linpred))
odds <- exp(linpred)
res <- cbind(data, fixed.med, random.iid,
    random.smooth, linpred, pred, odds)
```

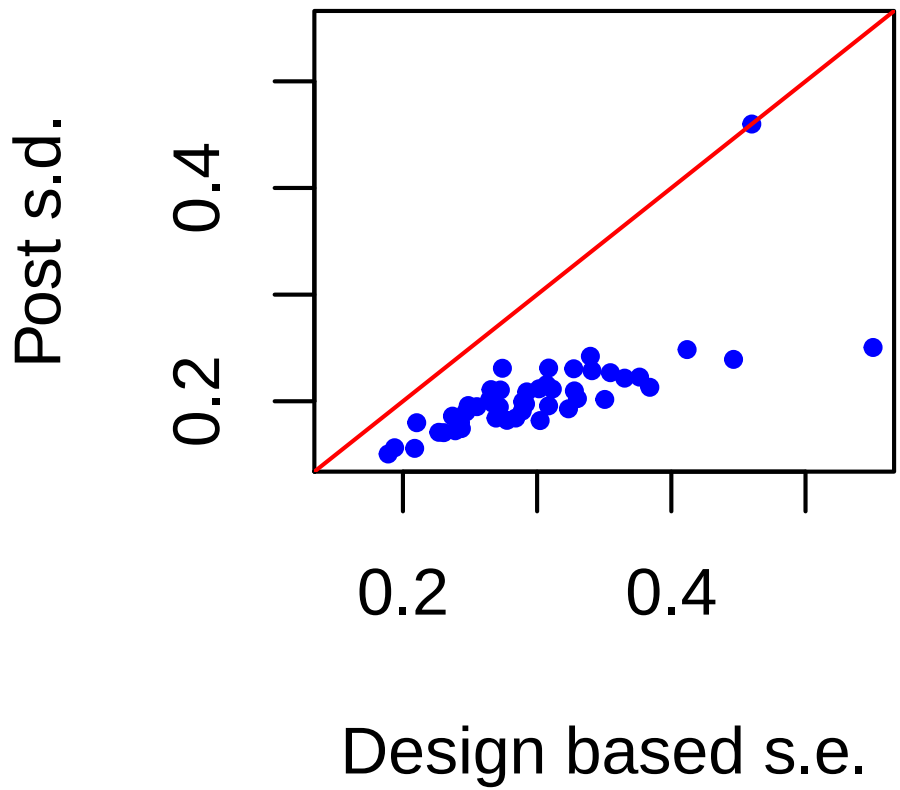
Comparison of estimates on logit scale

```
plot(mod.smooth$summary.fitted.values[, "mean"] ~ logit.pi,
    pch = 19, col = "blue", cex = 0.5, xlab = "Design est of prev",
    ylab = "Post mean smoothed prev", xlim = c(-3.8,
        -1.2), ylim = c(-3.8, -1.2))
abline(a = 0, b = 1, col = "red")
```



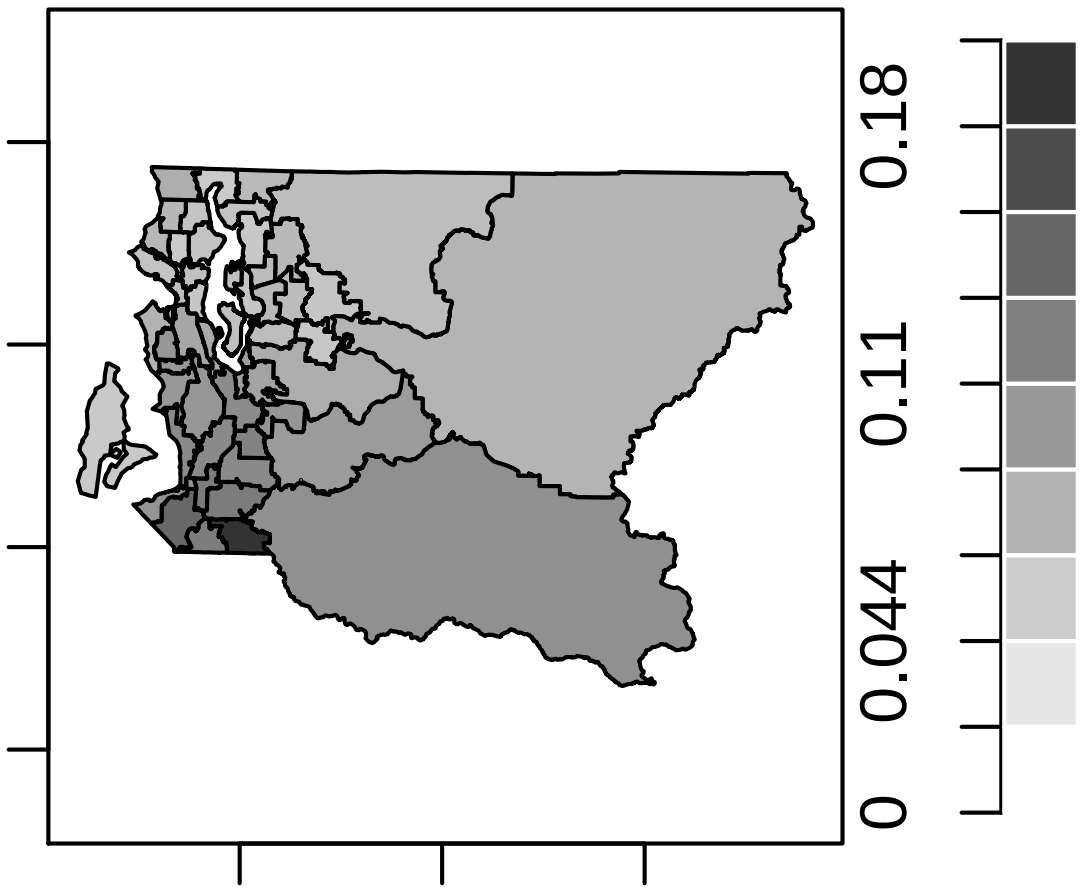
Comparison of uncertainty measures on logit scale

```
plot(mod.smooth$summary.fitted.values[, "sd"] ~ sqrt(v.i),
    pch = 19, col = "blue", cex = 0.5, xlab = "Design based s.e.",
    ylab = "Post s.d.", xlim = c(0.15, 0.55), ylim = c(0.15,
        0.55))
abline(a = 0, b = 1, col = "red")
```



Predicted Prevalence

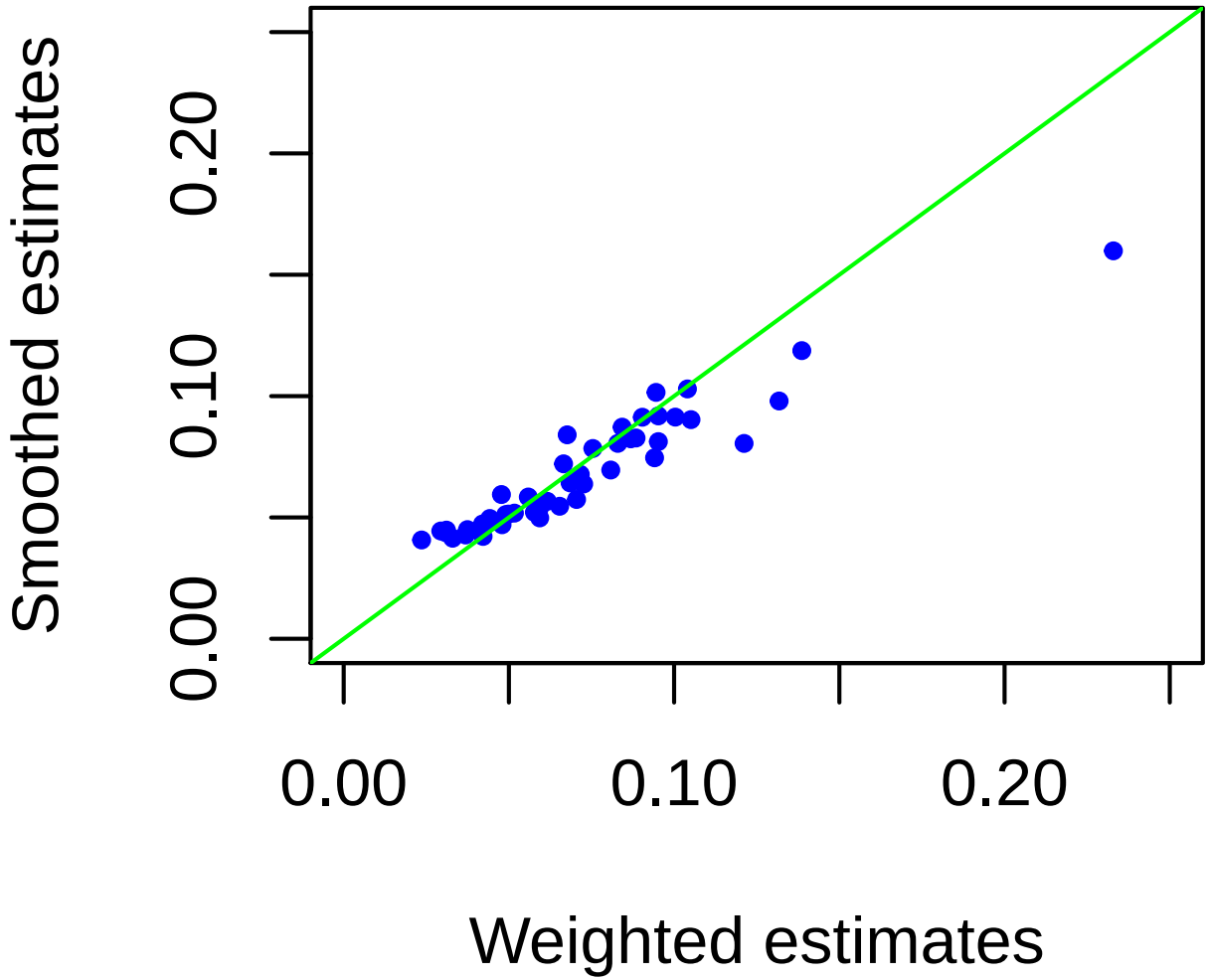
```
par(mar = c(1, 1, 1, 1))
mapvariable(res[, "pred"], kingshapepoly, ncut = 1000,
    nlevels = 10, lower = 0, upper = 0.2)
```



Comparison of estimates on prevalence scale: notice the shrinkage

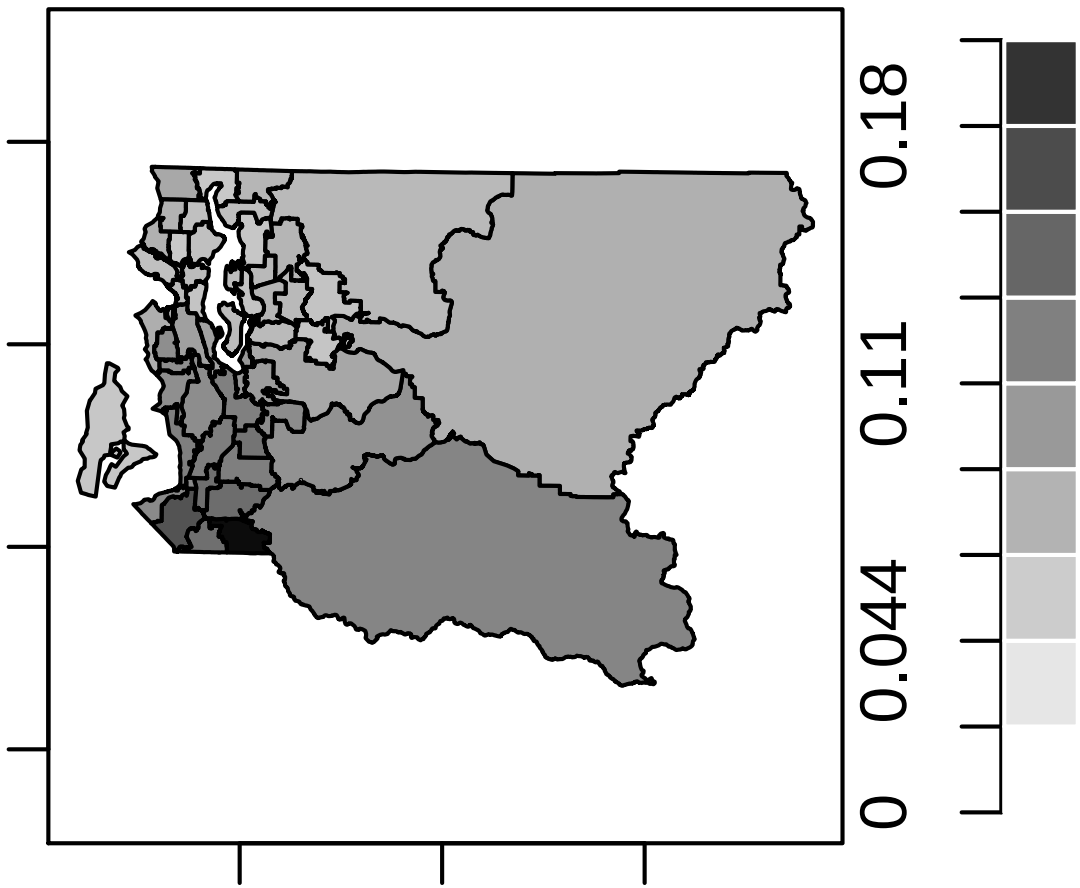
```
summary(p.i)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.02358 0.04757 0.06595 0.07136 0.08896 0.23290
summary(res[, "pred"])
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.04071 0.04921 0.05892 0.06772 0.08248 0.15990
plot(res[, "pred"] ~ p.i, pch = 19, xlim = c(0, 0.25),
    ylim = c(0, 0.25), col = "blue", cex = 0.5, xlab = "Weighted
    ylab = "Smoothed estimates")
abline(0, 1, col = "green")
```

Comparison of estimates on prev scale: notice the shrinkage



Predicted diabetes odds

```
par(mar = c(1, 1, 1, 1))
mapvariable(res[, "odds"], kingshapepoly, ncut = 1000,
    nlevels = 10, lower = 0, upper = 0.2)
```



Post sd of prevalence

We model on the log scale and so to obtain inference on the prevalence scale we need to either simulate from the posterior for the logit and transform, or use numerical integration on the approximation to the marginal distribution.

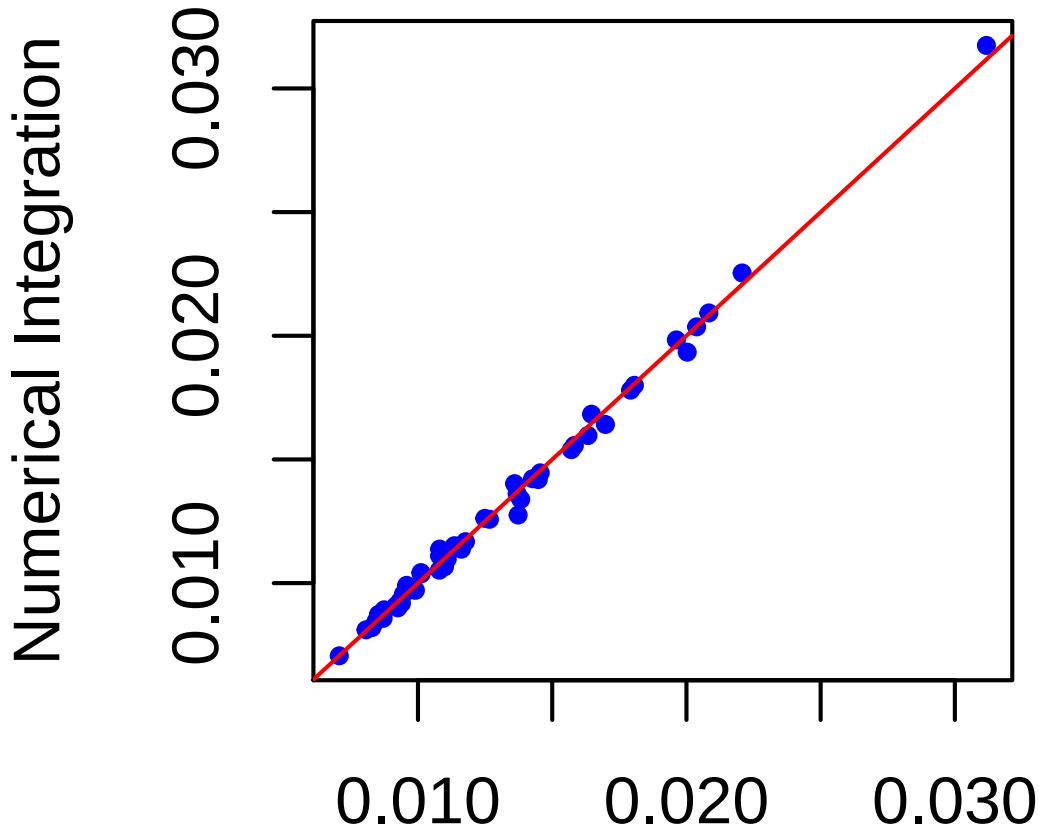
We carry out both and then compare the results.

Post sd of prevalence

```
expit <- function(x) {
    exp(x)/(exp(x) + 1)
}
n.sim <- 1000
test <- matrix(NA, nrow = n.area, ncol = 2)
test <- as.data.frame(test)
colnames(test) <- c("simulated", "e.marginal")
for (i in 1:n.area) {
    test[i, "simulated"] <- sd(expit(inla.rmarginal(n.sim,
        mod.smooth$marginals.linear.predictor[[i]])))
    expectations <- inla.emarginal(function(x) c(expit(x),
        expit(x)^2), mod.smooth$marginals.linear.predictor[[i]])
    test[i, "e.marginal"] <- sqrt(expectations[2] -
        expectations[1]^2)
}
```

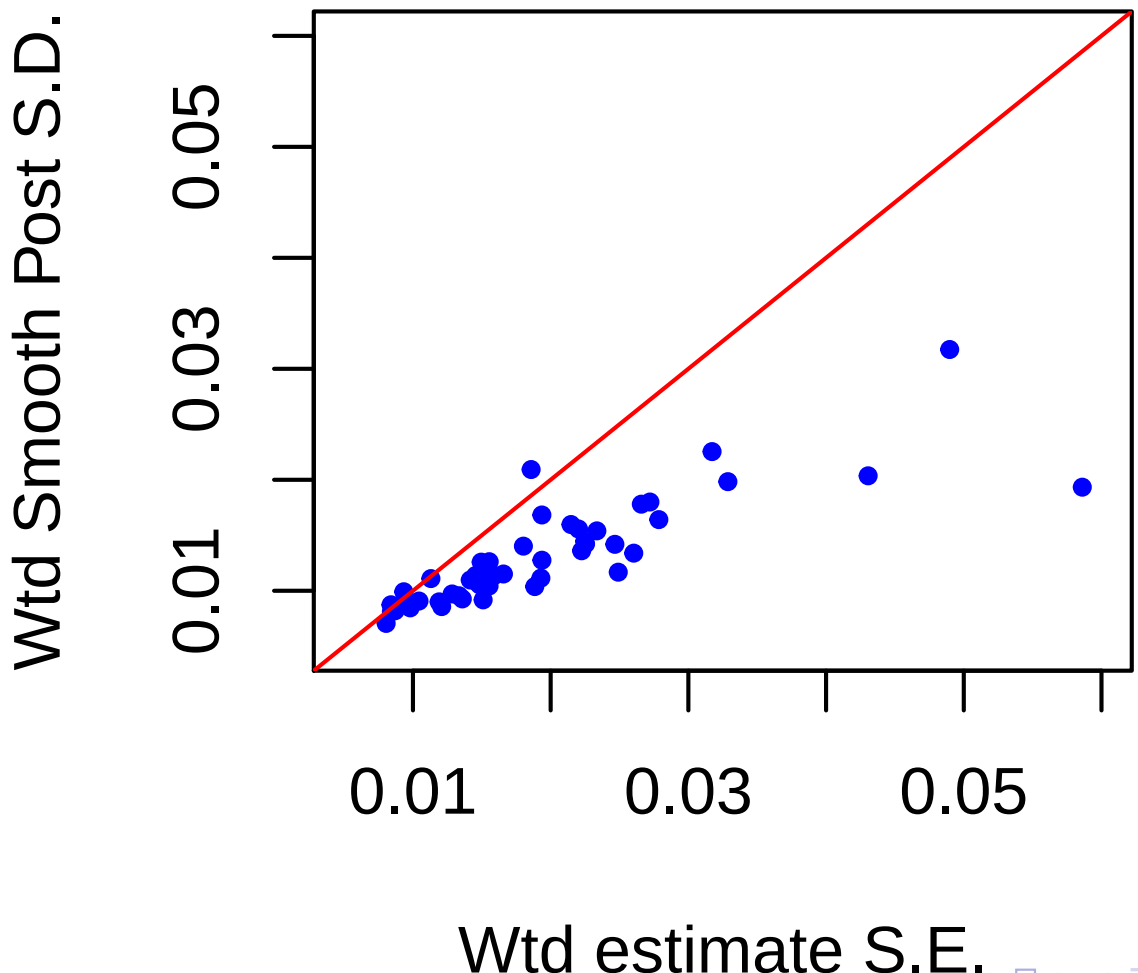
Comparison of approaches: good agreement

```
plot(test$simulated, test$e.marginal, xlab = "Simulation",
     ylab = "Numerical Integration", pch = 19,
     cex = 0.5, col = "blue")
abline(a = 0, b = 1, col = "red")
```



Comparison of standard errors

```
plot(test$e.marginal ~ sqrt(dv.i), pch = 19, cex = 0.5,
    col = "blue", ylab = "Wtd Smooth Post S.D.", xlab = "Wtd est
    xlim = c(0.005, 0.06), ylim = c(0.005, 0.06))
abline(0, 1, col = "red")
```



Conclusions

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