Bayesian SAE using Complex Survey Data

Lecture 4B: Hierarchical Spatial Bayesian Modeling with INLA

Richard Li

Department of Statistics
University of Washington

Outline

Spatial hierarchical normal model

Hierarchical normal model

Recall the hierarchical normal model

$$\begin{aligned} y_{ij}|\theta_i, \sigma^2 &\sim \text{Normal}\left(\theta_i, \sigma^2\right) \\ \theta_i|\mu, \tau^2 &\sim \text{Normal}\left(\mu, \tau^2\right) \end{aligned}$$

And we use the following 'default' priors on the unknown parameters

$$\begin{aligned} \mu &\sim \text{Normal}\left(\mu_0, \gamma_0^2\right) \\ \sigma^2 &\sim \text{InvGamma}\left(\nu_0/2, \nu_0\sigma_0^2/2\right) \\ \tau^2 &\sim \text{InvGamma}\left(\eta_0/2, \eta_0\tau_0^2/2\right) \end{aligned}$$

The hyperparameters we need to specify are

- μ_0 and τ_0^2 are the prior guess at the μ and the certainty of this guess.
- σ_0 and ν_0 are the prior guess at the within-area variance σ^2 and the certainty of this guess.
- τ_0 and η_0 are the prior guess at the across-area variance τ and the certainty of this guess.

Hierarchical normal model

We change the notation a little by denote $\sigma_\delta = \tau$, and can rewrite the previous model as

$$
\begin{aligned}
y_{ij} &= \mu + \delta_i + \epsilon_{ij} \\
\epsilon_{ij} | \sigma^2_\epsilon &\sim \text{Normal}(0, \sigma^2_\epsilon) \\
\delta_i | \sigma^2_\delta &\sim \text{Normal}(0, \sigma^2_\delta)
\end{aligned}
$$

We can now add in the spatial smoothing effect by letting

$$
\begin{aligned}
y_{ij} &= \mu + \delta_i + s_i + \epsilon_{ij} \\
s_i | s_{i'}, i' \in \text{ne}(i) &\sim \text{Normal}\left(\frac{1}{n_i} \sum_{i' \in \text{ne}(i)} s_{i'}, \frac{\sigma^2_s}{n_i}\right)
\end{aligned}
$$

where s_i is a intrinsic conditional autoregressive (ICAR) random effect in space.

Spatial smoothing: read map

Similar as before, we read in the map files first

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

```
# 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
```

Spatial smoothing: read map

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]])
```

Spatial smoothing: read data

We read in the simulated data from before. Notice:

- When using INLA, 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*
- So we need to recode the area index in the dataset first.

```
load("../data/simKing.rda")
pop$area <- match(pop$areaname, colnames(mat))
```

Spatial smoothing: read data

- We randomly sample $2,000$ observation from the population and calculate the naive area mean.
- Multiple random effects each need an index variable (`unstruct` and `struct` below).

```
set.seed(1)
samp <- pop[sample(1:dim(pop)[1], 2000), ]
samp <- data.frame(unstruct = samp$area, struct = samp$area,
    value = samp$weight)
theta.naive <- aggregate(value ~ unstruct, samp, mean)[,
    2]
```

Recall non-spatial smoothing

```
library(INLA)
newx <- data.frame(value=NA, unstruct = 1:48, struct=1:48)
hyperprior1 <- list(theta=list(prior='loggamma',
        param=c(1, 0.5)))
hyperprior2 <- list(theta=list(prior='loggamma',
        param=c(1, 0.1)))
formula <- value ~ 1+f(unstruct, model="iid",
        hyper = hyperprior2)
fit1 <- inla(formula,
        data=rbind(newx, samp), family = "gaussian",
        control.family = list(hyper =hyperprior1),
        control.predictor = list(compute = TRUE))
```

Spatial smoothing

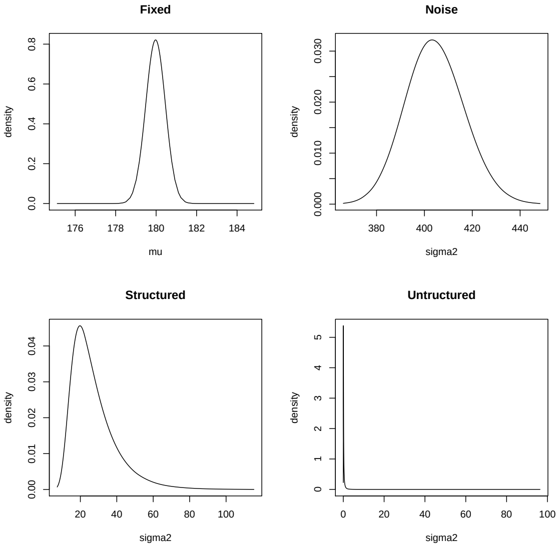
```
formula2 <- value ~ 1+
        f(struct,model='besag',
                                adjust.for.con.comp=TRUE,
                                constr=TRUE,graph=mat,
                                scale.model = TRUE,
                                hyper = hyperprior2) +
        f(unstruct, model="iid",
                                hyper = hyperprior2)
fit2 <- inla(formula2,
        data=rbind(newx, samp), family = "gaussian",
        control.family = list(hyper = hyperprior1),
        control.predictor = list(compute = TRUE))
```

The structured effects are all scaled to have unit generalized marginal variance, so that we can use the same hyperpriors as the independent term. 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.

Spatial smoothing: the posterior

```
par(mfrow = c(2, 2))
plot(fit2$marginals.fixed[[1]], type = "l", xlab = "mu",
    ylab = "density", main = "Fixed")
plot(inla.tmarginal(function(x) 1/x, fit2$marginals.hyperpar[[1]]),
    type = "l", xlab = "sigma2", ylab = "density",
    main = "Noise")
plot(inla.tmarginal(function(x) 1/x, fit2$marginals.hyperpar[[2]]),
    type = "l", xlab = "sigma2", ylab = "density",
    main = "Structured")
plot(inla.tmarginal(function(x) 1/x, fit2$marginals.hyperpar[[3]]),
    type = "l", xlab = "sigma2", ylab = "density",
    main = "Untructured")
```

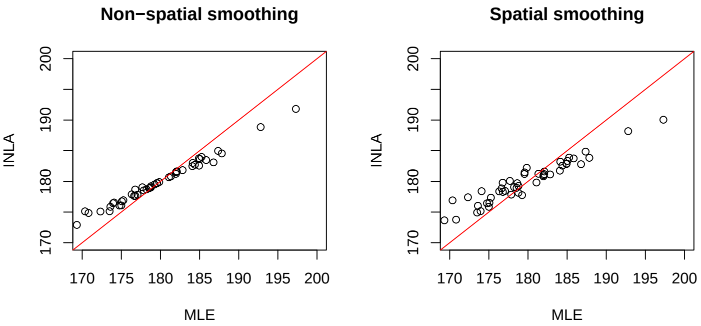
Spatial smoothing: the posterior



Spatial smoothing: compare with non-spatial smoothing

```
par(mfrow = c(1, 2))
theta.median <- fit1$summary.linear.predictor[1:48,
    "0.5quant"]
plot(theta.naive, theta.median, xlim = c(170, 200),
    ylim = c(170, 200), xlab = "MLE", ylab = "INLA",
    main = "Non-spatial smoothing")
abline(c(0, 1), col = "red")
theta.spatial <- fit2$summary.linear.predictor[1:48,
    "0.5quant"]
plot(theta.naive, theta.spatial, xlim = c(170, 200),
    ylim = c(170, 200), xlab = "MLE", ylab = "INLA",
    main = "Spatial smoothing")
abline(c(0, 1), col = "red")
```

Spatial smoothing: compare with non-spatial smoothing



Spatial smoothing: compare with non-spatial smoothing

Now we organize the posterior medians of θ_i, ϵ_i, and s_i into a data frame

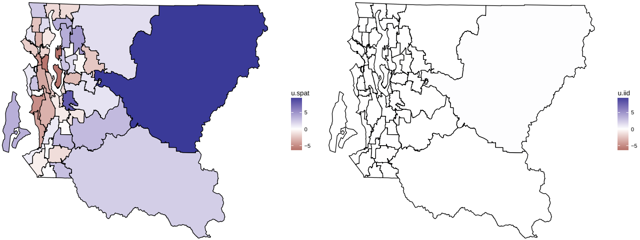
```
samp.aggre <- aggregate(value ~ unstruct, samp, mean)
colnames(samp.aggre)[2] <- "MLE"
samp.aggre$median <- theta.median
samp.aggre$median.spatial <- theta.spatial
samp.aggre$u.spat <- fit2$summary.random$struct[, "0.5quant"]
samp.aggre$u.iid <- fit2$summary.random$unstruct[,
    "0.5quant"]
samp.aggre$name <- colnames(mat)
```

Spatial smoothing: random effects

```
library(ggplot2)
library(gridExtra)
lim <- range(c(samp.aggre$u.iid, samp.aggre$u.spat))
geo <- fortify(kingshape, region = "HRA2010v2_")
geo1 <- merge(geo, samp.aggre, by = "id", by.y = "name")
g1 <- ggplot(geo1)
g1 <- g1 + geom_polygon(aes(x = long, y = lat, group = group,
    fill = u.spat), color = "black")
g1 <- g1 + theme_void() + scale_fill_gradient2(limits = lim,
    midpoint = 0)
```

Spatial smoothing: random effects

```
g2 <- ggplot(geo1)
g2 <- g2 + geom_polygon(aes(x = long, y = lat, group = group,
    fill = u.iid), color = "black")
g2 <- g2 + theme_void() + scale_fill_gradient2(limits = lim,
    midpoint = 0)
grid.arrange(grobs = list(g1, g2), ncol = 2)
```



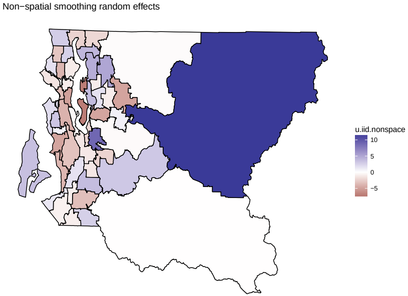
Spatial smoothing: random effects

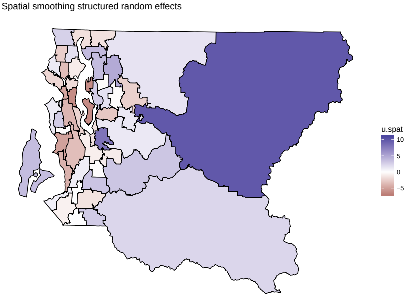
We can also compare the structured random effects from the spatial smoothing model to that form the non-spatial smoothing model

```
samp.aggre$u.iid.nonspace <- fit1$summary.random$unstruct[,
    "0.5quant"]
lim3 <- range(c(samp.aggre$u.iid.nonspace, samp.aggre$u.spat))
geo1 <- merge(geo, samp.aggre, by = "id", by.y = "name")
g0 <- ggplot(geo1)
g0 <- g0 + geom_polygon(aes(x = long, y = lat, group = group,
    fill = u.iid.nonspace), color = "black")
g0 <- g0 + theme_void() + scale_fill_gradient2(limits = lim3,
    midpoint = 0)
g0 <- g0 + ggtitle("Non-spatial smoothing random effects")
g1 <- g1 + ggtitle("Spatial smoothing structured random effects") +
    scale_fill_gradient2(limits = lim3, midpoint = 0)
```

Spatial smoothing: random effects

```r
grid.arrange(grobs = list(g0, g1), ncol = 2)
```





Mean weight without smoothing

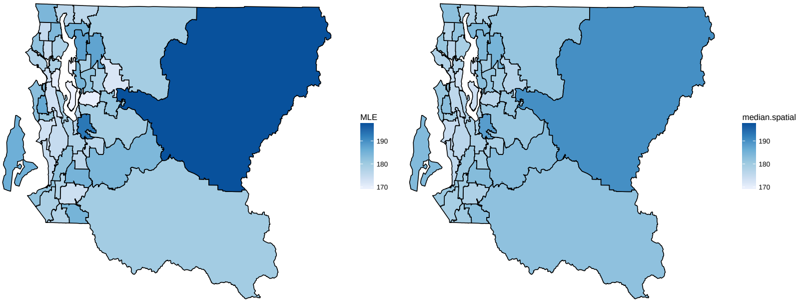
```
g1 <- ggplot(geo1)
lim <- range(c(samp.aggre$MLE, samp.aggre$median, samp.aggre$median.spa
g1 <- g1 + geom_polygon(aes(x = long, y = lat, group = group,
    fill = MLE), color = "black")
g1 <- g1 + theme_void() + scale_fill_distiller(direction = 1,
    limits = lim)

g2 <- ggplot(geo1) + geom_polygon(aes(x = long, y = lat,
    group = group, fill = median), color = "black")
g2 <- g2 + theme_void() + scale_fill_distiller(direction = 1,
    limits = lim)

g3 <- ggplot(geo1)
g3 <- g3 + geom_polygon(aes(x = long, y = lat, group = group,
    fill = median.spatial), color = "black")
g3 <- g3 + theme_void() + scale_fill_distiller(direction = 1,
    limits = lim)
```

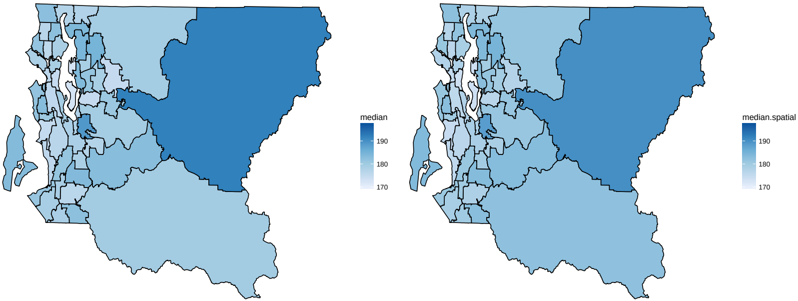
MLE v.s. Spatial smoothing

```
grid.arrange(grobs = list(g1, g3), ncol = 2)
```



Non-spatial v.s. Spatial smoothing

```
grid.arrange(grobs = list(g2, g3), ncol = 2)
```



Spatial Lognormal-binomial model

Simulated binary outcome

```
set.seed(1)
samp <- pop[sample(1:dim(pop)[1], 2000), ]
samp <- data.frame(unstruct = samp$area, value = samp$diabetes)
samp.aggre <- aggregate(value ~ unstruct, samp, sum)
samp.aggre$n <- aggregate(value ~ unstruct, samp, length)[,
    2]
samp.aggre$struct <- samp.aggre$unstruct
```

Non-spatial smoothing of binomial model

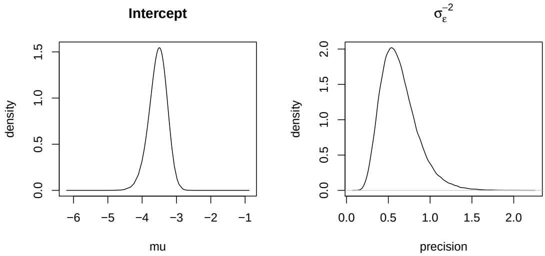
```
formula = value ~ 1 + f(unstruct, model = "iid", param = c(0.5,
    0.0015))
fit3 <- inla(formula, family = "binomial", data = samp.aggre,
    Ntrials = n, control.predictor = list(compute = TRUE))
```

Spatial smoothing of binomial model

```r
formula = value ~ 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))
fit4 <- inla(formula,
        family="binomial",      data=samp.aggre, Ntrials=n,
        control.predictor = list(compute = TRUE))
```

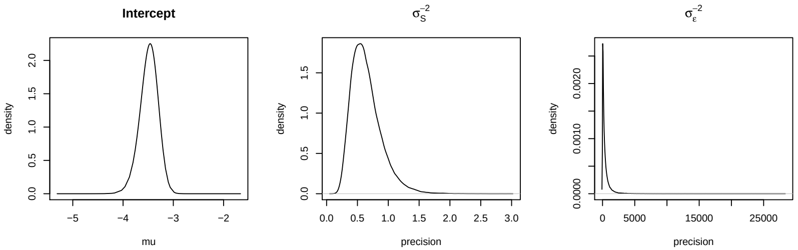
Posteriors: non-spatial smoothing

```
par(mfrow = c(1, 2))
plot(fit3$marginals.fixed[[1]], type = "l", xlab = "mu",
    ylab = "density", main = "Intercept")
plot(density(inla.rmarginal(1e+05, fit3$marginals.hyperpar[[1]])),
    type = "l", xlab = "precision", ylab = "density",
    main = expression(sigma[epsilon]^-2))
```



Posteriors: spatial smoothing

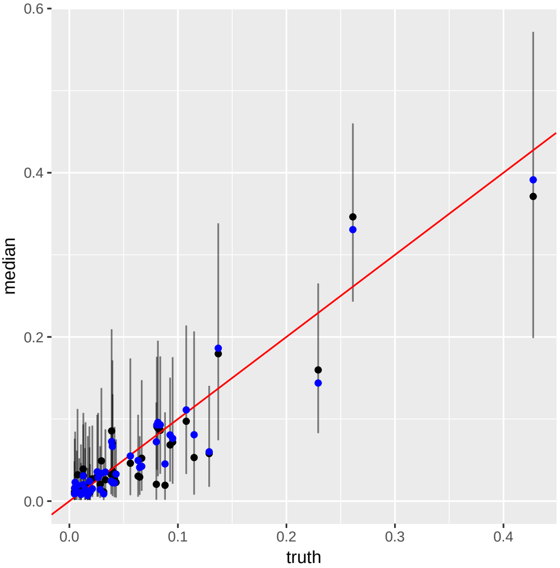
```
par(mfrow = c(1, 3))
plot(fit4$marginals.fixed[[1]], type = "l", xlab = "mu",
    ylab = "density", main = "Intercept")
plot(density(inla.rmarginal(1e+05, fit4$marginals.hyperpar[[1]])),
    type = "l", xlab = "precision", ylab = "density",
    main = expression(sigma[S]^-2))
plot(density(inla.rmarginal(1e+05, fit4$marginals.hyperpar[[2]])),
    type = "l", xlab = "precision", ylab = "density",
    main = expression(sigma[epsilon]^-2))
```



Compare results

```
prev <- data.frame(truth = aggregate(diabetes ~ area,
    pop, mean)[, 2], mle = aggregate(value ~ unstruct,
    samp, mean)[, 2], size = aggregate(value ~ unstruct,
    samp, length)[, 2])
prev <- cbind(prev, fit3$summary.fitted.values, fit4$summary.fit
prev$name <- colnames(mat)
colnames(prev)[6:8] <- c("lower", "median", "upper")
colnames(prev)[11:14] <- c("sd.sp", "lower.sp", "median.sp",
    "upper.sp")
g <- ggplot(prev, aes(x = truth, y = median, ymin = lower,
    ymax = upper))
g <- g + geom_point() + geom_errorbar(alpha = 0.5) +
    geom_abline(color = "red")
g <- g + geom_point(aes(x = truth, y = median.sp),
    color = "blue")
g
```

Compare results

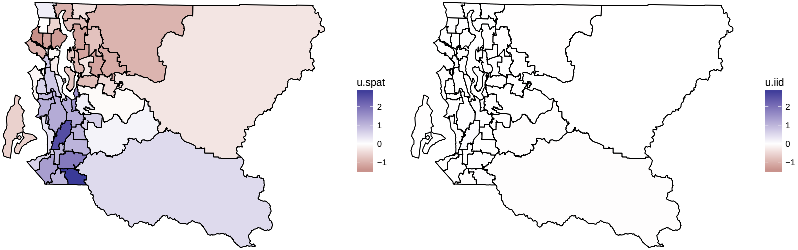


Spatial smoothing: random effects

```r
prev$u.spat <- fit4$summary.random$struct[, "0.5quant"]
prev$u.iid <- fit4$summary.random$unstruct[, "0.5quant"]
geo <- fortify(kingshape, region = "HRA2010v2_")
geo2 <- merge(geo, prev, by = "id", by.y = "name")
lim <- range(c(prev$u.spat, prev$u.iid))
g1 <- ggplot(geo2) + geom_polygon(aes(x = long, y = lat,
    group = group, fill = u.spat), color = "black")
g1 <- g1 + theme_void() + scale_fill_gradient2(limits = lim,
    midpoint = 0)
```

Spatial smoothing: random effects

```
g2 <- ggplot(geo2) + geom_polygon(aes(x = long, y = lat,
    group = group, fill = u.iid), color = "black")
g2 <- g2 + theme_void() + scale_fill_gradient2(limits = lim,
    midpoint = 0)
grid.arrange(grobs = list(g1, g2), ncol = 2)
```



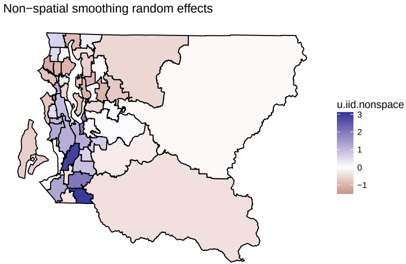
Spatial smoothing: random effects

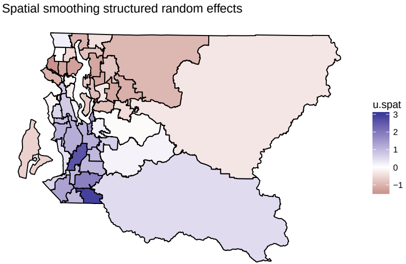
Again we compare the structured random effects to the random effects from the non-spatial smoothing model.

```
prev$u.iid.nonspace <- fit3$summary.random$unstruct[,
    "0.5quant"]
lim2 <- range(c(prev$u.iid.nonspace, prev$u.spat))
geo2 <- merge(geo, prev, by = "id", by.y = "name")
g0 <- ggplot(geo2)
g0 <- g0 + geom_polygon(aes(x = long, y = lat, group = group,
    fill = u.iid.nonspace), color = "black")
g0 <- g0 + theme_void() + scale_fill_gradient2(limits = lim2,
    midpoint = 0)
g0 <- g0 + ggtitle("Non-spatial smoothing random effects")
g1 <- g1 + ggtitle("Spatial smoothing structured random effects"
    scale_fill_gradient2(limits = lim2, midpoint = 0)
```

Spatial smoothing: random effects

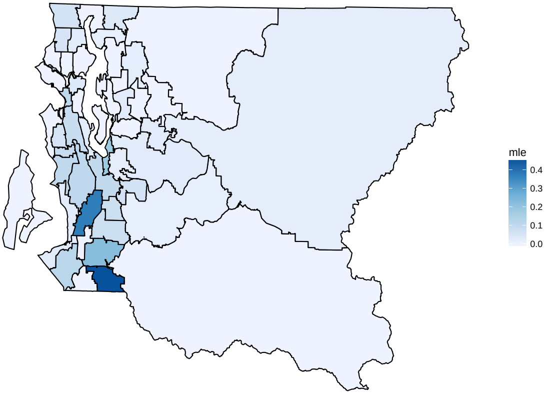
```r
grid.arrange(grobs = list(g0, g1), ncol = 2)
```





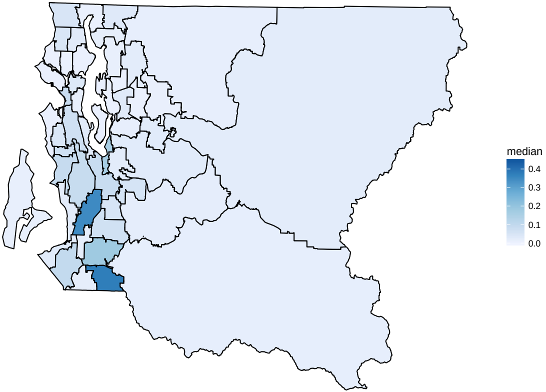
Mean prevalence

```
lim <- range(c(prev$mle, prev$median, prev$median.sp))
g1 <- ggplot(geo2) + geom_polygon(aes(x = long, y = lat,
    group = group, fill = mle), color = "black")
g1 <- g1 + theme_void() + scale_fill_distiller(direction = 1,
    limits = lim)
g1
```



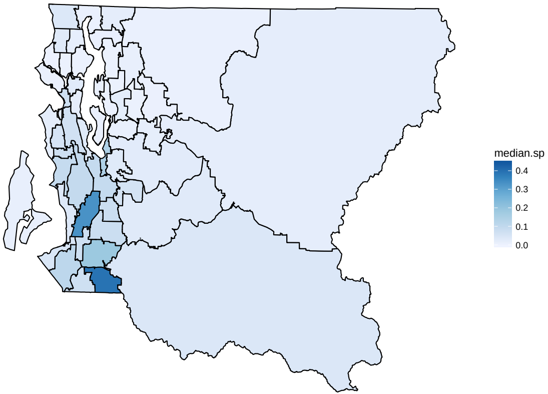
Mean prevalence with non-spatial smoothing

```
g2 <- ggplot(geo2) + geom_polygon(aes(x = long, y = lat,
    group = group, fill = median), color = "black")
g2 <- g2 + theme_void() + scale_fill_distiller(direction = 1,
    limits = lim)
g2
```



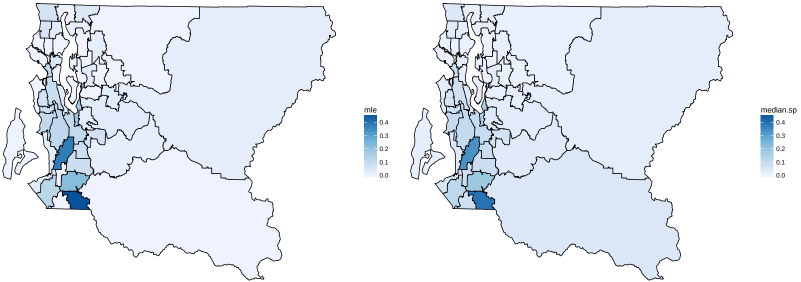
Mean prevalence with spatial smoothing

```
g3 <- ggplot(geo2) + geom_polygon(aes(x = long, y = lat,
    group = group, fill = median.sp), color = "black")
g3 <- g3 + theme_void() + scale_fill_distiller(direction = 1,
    limits = lim)
g3
```



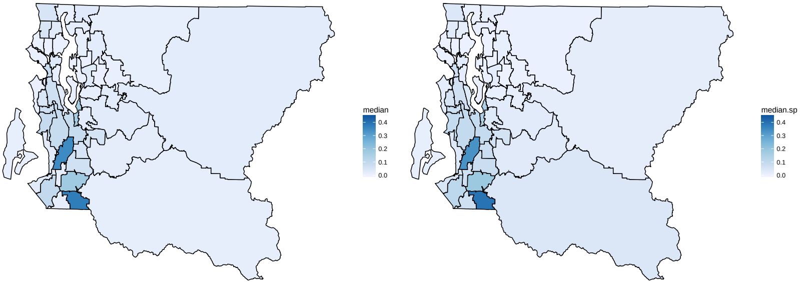
MLE vs Spatial smoothing

```r
grid.arrange(grobs = list(g1, g3), ncol = 2)
```



Non-spatial vs Spatial smoothing

```r
grid.arrange(grobs = list(g2, g3), ncol = 2)
```



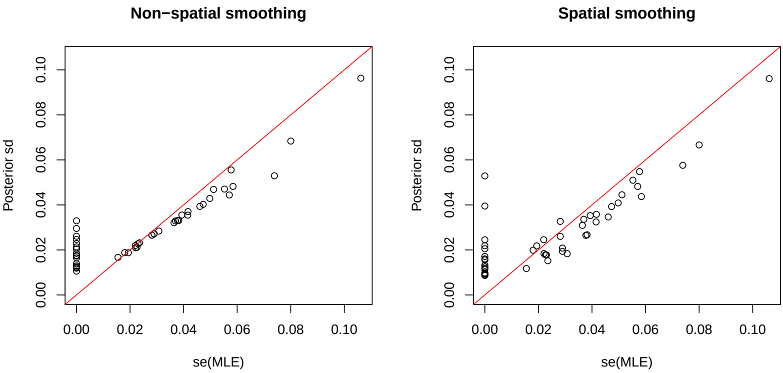
Uncertainty measure

We can visually compare the binomial standard errors and confidence intervals with the posterior summaries from the two smoothing models

```
prev$mle.se <- sqrt(prev$mle * (1 - prev$mle)/prev$size)
prev$mle.lower <- prev$mle - 1.96 * prev$mle.se
prev$mle.upper <- prev$mle + 1.96 * prev$mle.se
lim <- range(c(prev$mle.se, prev$sd, prev$sd.sp))
par(mfrow = c(1, 2))
plot(prev$mle.se, prev$sd, xlab = "se(MLE)", ylab = "Posterior sd",
    xlim = lim, ylim = lim, main = "Non-spatial smoothing")
abline(c(0, 1), col = "red")
plot(prev$mle.se, prev$sd.sp, xlab = "se(MLE)", ylab = "Posterior sd",
    xlim = lim, ylim = lim, main = "Spatial smoothing")
abline(c(0, 1), col = "red")
```

Uncertainty measure

Naive binomial confidence interval versus posterior credible interval from the spatial smoothing mode

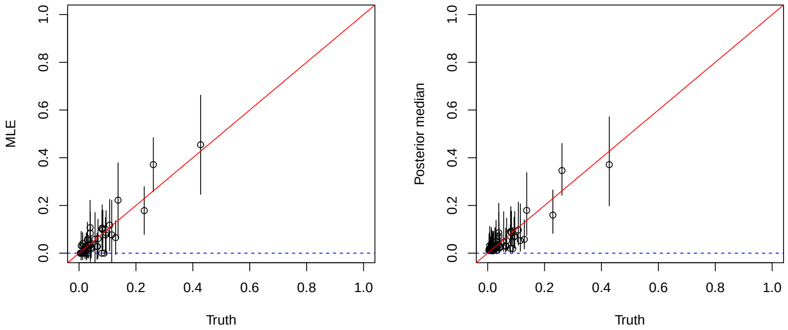


Uncertainty measure

Naive binomial confidence interval versus posterior credible interval from the spatial smoothing mode

```
par(mfrow = c(1, 2))
plot(prev$truth, prev$mle, xlab = "Truth", ylab = "MLE",
    xlim = c(0, 1), ylim = c(0, 1))
segments(x0 = prev$truth, y0 = prev$mle.lower, y1 = prev$mle.upper)
abline(c(0, 1), col = "red")
abline(h = 0, col = "blue", lty = 2)
plot(prev$truth, prev$median, xlab = "Truth", ylab = "Posterior median"
    xlim = c(0, 1), ylim = c(0, 1))
segments(x0 = prev$truth, y0 = prev$lower, y1 = prev$upper)
abline(c(0, 1), col = "red")
abline(h = 0, col = "blue", lty = 2)
```

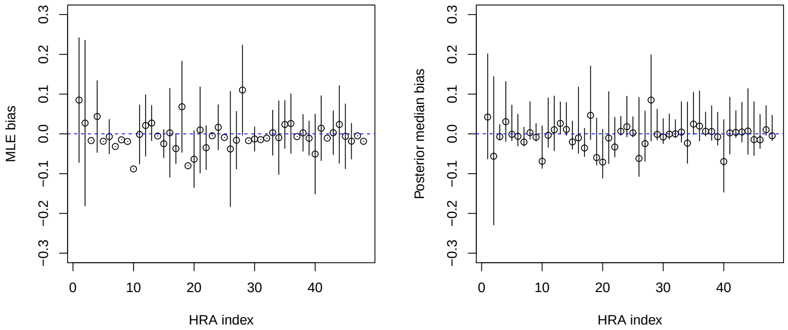
Uncertainty measure



Uncertainty measure

```r
par(mfrow = c(1, 2))
plot(1:48, prev$mle - prev$truth, xlab = "HRA index",
    ylab = "MLE bias", ylim = c(-0.3, 0.3))
segments(x0 = 1:48, y0 = prev$mle.lower - prev$truth,
    y1 = prev$mle.upper - prev$truth)
abline(h = 0, col = "blue", lty = 2)
plot(1:48, prev$median - prev$truth, xlab = "HRA index",
    ylab = "Posterior median bias", ylim = c(-0.3,
        0.3))
segments(x0 = 1:48, y0 = prev$lower - prev$truth, y1 = prev$upper -
    prev$truth)
abline(h = 0, col = "blue", lty = 2)
```

Uncertainty measure



Spatial smoothing: decomposition of variation

- 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

$$
\begin{aligned}
y_{ij} &= \mu + \delta_i + s_i + \epsilon_{ij} \\
\epsilon_{ij}|\sigma_\epsilon^2 &\sim \text{Normal}(0, \sigma_\epsilon^2) \\
\delta_i|\sigma_\delta^2 &\sim \text{Normal}(0, \sigma_\delta^2) \\
s_i|s_{i'}, i' \in \text{ne}(i) &\sim \text{Normal}(\frac{1}{n_i} \sum_{i' \in \text{ne}(i)} s_{i'}, \frac{\sigma_s^2}{n_i})
\end{aligned}
$$

- They are more comparable after setting `scale.model = TRUE` in `f()` function, since the covariance function for $\boldsymbol{s}$ are rescaled.

Spatial smoothing: decomposition of variation

```r
sigma2.spatial <- inla.emarginal(function(x) {
    1/x
}, fit4$marginals.hyper$"Precision for struct")
sigma2.iid <- inla.emarginal(function(x) {
    1/x
}, fit4$marginals.hyper$"Precision for unstruct")
prop <- sigma2.spatial/(sigma2.spatial + sigma2.iid)
c(sigma2.spatial, sigma2.iid, prop)

## [1] 1.7752373 0.0222911 0.9875990
```

About 98.8% of variance are explained by the spatial random effects.

Spatial smoothing: decomposition of variation

- Alternatively, we may also compare the empirical posterior marginal variance instead
- Let $s_u^2 = \sum_{i=1}^{n}(u_i - \bar{u})^2/(n-1)$, for a sample $\boldsymbol{u}$ form the posterior distribution of $\boldsymbol{s}$
- The fraction of variance explained by the spatial random effect vector $\boldsymbol{u}$ is frac $= s_u^2/(s_u^2 + \sigma_\delta^2)$.
- By sampling a large enough number of values from the marginal posterior distribution of the random effects, we can compute the average fraction of variance explained

Spatial smoothing: decomposition of variation

```r
spatial <- matrix(NA, 1e5, 48)
for (i in 1:48){
  spatial[,i] <- inla.rmarginal(1e4,
        fit4$marginals.random$struct[[i]])
}
S.spatial <- apply(spatial,1,var)
S.iid <- inla.rmarginal(1e4,
        inla.tmarginal(function(x){1/x},
        fit4$marginals.hyper$"Precision for unstruct"))
prop2 <- mean(S.spatial/(S.spatial+S.iid))
c(mean(S.spatial), mean(S.iid), prop2)

## [1] 1.55145909 0.02555173 0.98512008
```

About 98.5% of variance are explained by the spatial random effects using the empirical marginal variance.