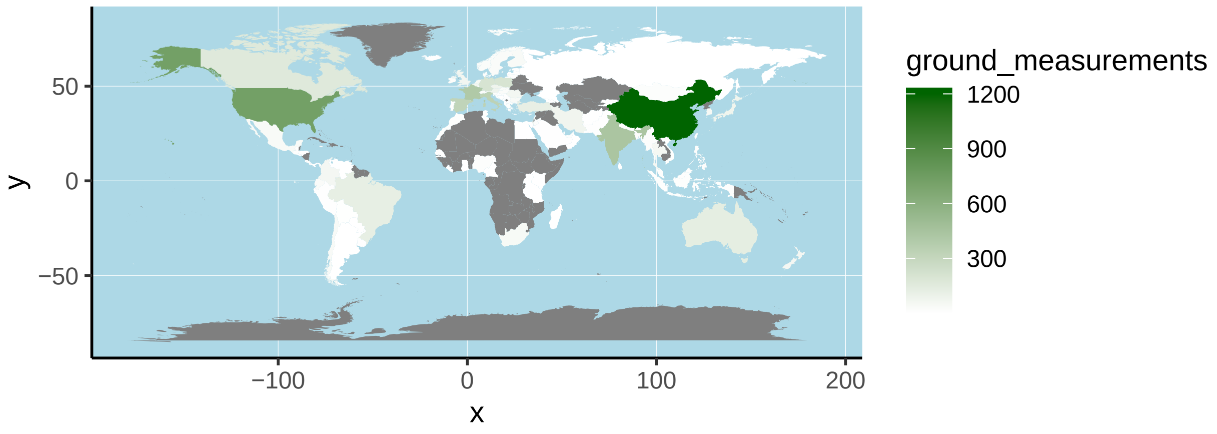
Source Code for Assignment 1

```r
library(dplyr)
library(here)
library(ggplot2)
library(INLA)
library(tidyr)
library(rworldmap)
library(fitdistrplus)
library(knitr)
load(here('data', 'assignment1.RData'))
d = assignment_1
```

PM25 is a measure of fine particle pollution. Exposure to the particles is linked to many health issues so it is important to have accurate global measures. In the US for example, the Environmental Proteciton Agency (EPA) has standards for 24-hour average PM25 rates across the country. How do global public health organizations measure PM25? For starters, there are a number of grouynd measurement stations. The map below displays the number of ground stations in each country.

```r
map.world <- map_data(map="world") %>%
   mutate(region=case_when(
    region=='UK' ~ 'United Kingdom',
    region=='USA' ~ 'United States',
    TRUE ~ region
  ))

d$region = d$country_name_1

ground_measurement_counts=map.world %>%
  group_by(region) %>%
  summarise() %>%
  mutate(region=case_when(
    region=='UK' ~ 'United Kingdom',
    region=='USA' ~ 'United States',
    TRUE ~ region
  )) %>%
  full_join(d %>%
              group_by(region) %>%
              summarise(ground_measurements = n()),
            by='region')
  # mutate(ground_measurements=replace_na(ground_measurements, 0))


ggplot(data=ground_measurement_counts, aes(map_id=region)) +
  geom_map(map=map.world, aes(fill=ground_measurements)) +
  expand_limits(x=map.world$long, y=map.world$lat) +
  coord_equal() +
  scale_fill_gradient(low="#FFFFFF", high="darkgreen") +
  theme_classic()+
  theme(panel.background = element_rect(fill = "lightblue",
                                colour = "lightblue",
                                size = 0.5, linetype = "solid"),
        panel.grid.major = element_line(size = 0.1, linetype = 'solid',
                                colour = "white")) +
  labs(caption='Grey indicates no measurements')
```



Grey indicates no measurements

Clearly, there are large portions of the globe with few, or no, moniitoring resources. Fortunately, we have satelite measurements that can capture measurements anywhere. However, these satelite measures are known to vary from the true ground measurements, likely due to atmospheric interference. In this analysis we will develop a method of calibrating our satelite measurements using ground measures as reference of truth.

It is fair to assume that at least some of the miscalibration of satelite is going to vary based on location on globe that measurement is being taken.

Builing model

We will start with the simplest model, a single regression as calibration of satelite to ground measure, iteratively add complexity to our model and compare to the previous versions. Along the way we will build understanding for how these techniques work.

Simple Linear Model

A simple linear regression across all measurements is modelled in the following way:

$$Y_i \sim N(\beta_0 + \beta_1 X_i, \sigma^2)$$
$$i = 1, ..., n \text{ observations}$$

where X_i are the satelite measurements on log scale and y_i are PM25 measurements from ground measurement stations on log scale.

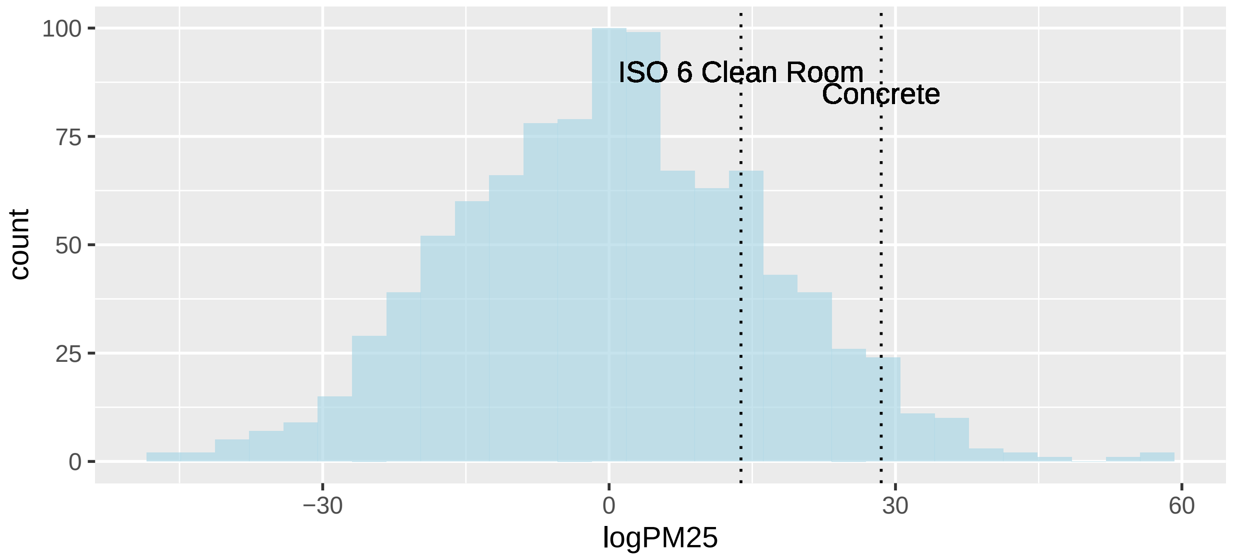
We are approaching this with Bayesian framework (i.e. $\beta|\sigma^2, y, X \sim Normal((X'X)^{-1}X'y, \sigma^2(X'X)^{-1})$).

If satelite measurements, X_i, were well-calibrated, we would expect the intercept to be zero and slope to be 1. We have a limited understanding of satellite miscalibration so it is reasonable to believe that the intercept and slope of well-calibrated scenario described should not be too far off. A vague prior on β (both beta's here) and σ would be something like $\beta \sim N(0, 100)$ and $\sigma \sim InvGamma(1, 100)$.

I believe that we can do a bit better with a weakly informative prior, $\beta \sim N(0, 5)$ and $\sigma \sim N^+(0, 1)$. The prior for σ^2 is based on an evaluation of EPA standards. According to EPA policies, PM25 minimum is 0 and maximum is 500, but anything above 300 is considered "Hazardous" so I think it is fair to assume that the stanard deviation of the distribution is somewhere between 5 (always safe) and 150 (often unsafe). A conservative standard deviation on the log scale for the case when anything above 150 is not likely to happen is approximately 1. See the appendix for EPA standards chart.

Let's simulate the prior predictive distribution using these priors.

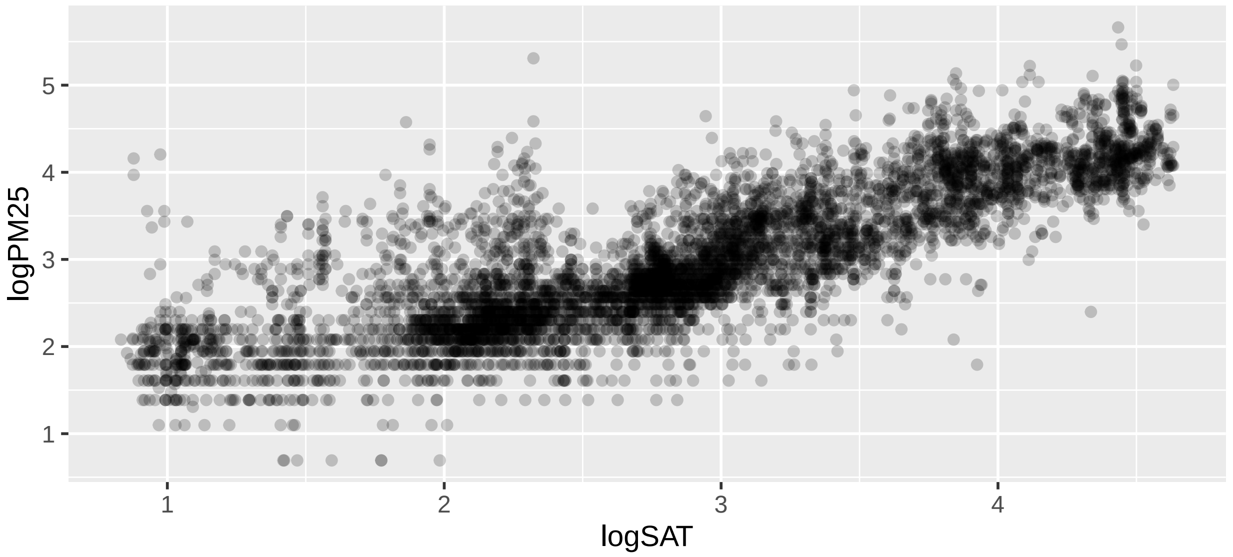
```r
# Concrete is 2,400 kilograms per cubic meter
# PM25 measured in micrograms per cubic meter
# 2400 kg per cubic meter = 2.4e+12 micrograms per cubic meter

concrete_dens = 2.4 * 10^12
log_concrete_dens = log(concrete_dens)
iso_6_clean_room_dens = 1000000
log_iso_6_clean_room_dens = log(iso_6_clean_room_dens)


nsims=1000
sigma <- abs(rnorm(nsims,0,1))
beta0 = rnorm(nsims, mean=0, sd=5)
beta1 = rnorm(nsims, mean=0, sd=5)

dsims = tibble(scaled_logPM25 = d$logPM25)

for (i in 1:nsims){
  mu_i = beta0[i] + beta1[i] * dsims
  dsims[paste0('sim', i)] = rnorm(n=length(mu_i$scaled_logPM25), mean=mu_i$scaled_logPM25, sd=sigma[i])
}

dsims_mean = lapply(dsims, mean)
a = tibble(values = unlist(dsims_mean))

a %>%
  ggplot(aes(x=values)) +
  geom_histogram(bins=30, alpha=0.7, fill='lightblue') +
  labs(x='logPM25') +
  geom_vline(xintercept=log_concrete_dens, linetype='dotted' ) +
  geom_text(x=log_concrete_dens, y=85, label='Concrete') +
  geom_vline(xintercept=log_iso_6_clean_room_dens, linetype='dotted') +
  geom_text(x=log_iso_6_clean_room_dens, y=90, label='ISO 6 Clean Room')
```



We've included the log density of air in an ISO 6 Clean Room and the log density of concrete to show that our prior predictive is reasonable (conserviatve in fact, which is fine). We'll continue with these priors.

Let's take a look at the data we are going to be using to build our model.

```
d %>%
  ggplot(aes(x=logSAT, y=logPM25)) +
  geom_point(alpha=0.2)
```

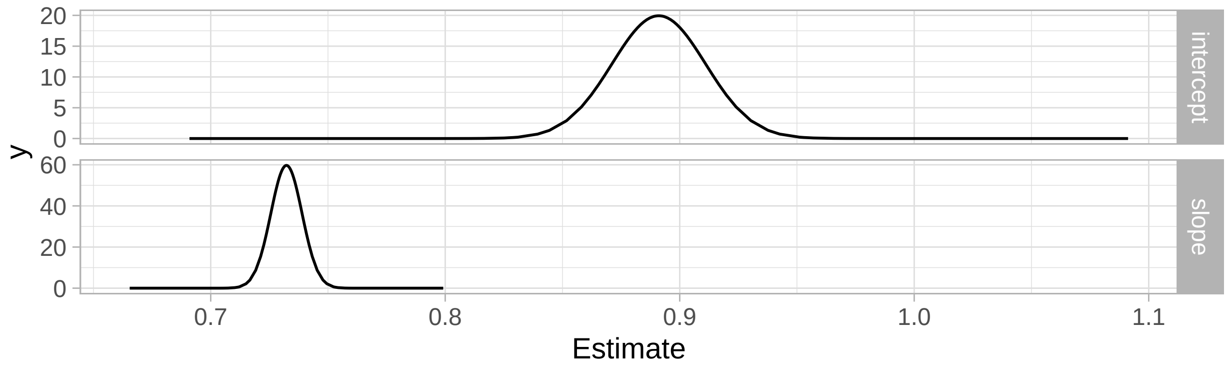


The posteriors of intercept and slope:

```
plain_lm_model = logPM25 ~ 1 + logSAT
result = inla(formula = plain_lm_model,
              data=d,
              family='gaussian',
              control.family=list(hyper=list(
                                  prec=list(prior="normal",param=c(0,1)))),
                                  #could not find half-normal distribution
              control.fixed = list(
                prec.intercept=1/5^2,
                prec = 1/5^2
              ),
              control.compute = list(config=TRUE, cpo = TRUE))

round(result$summary.fixed[,1:5],3)
```

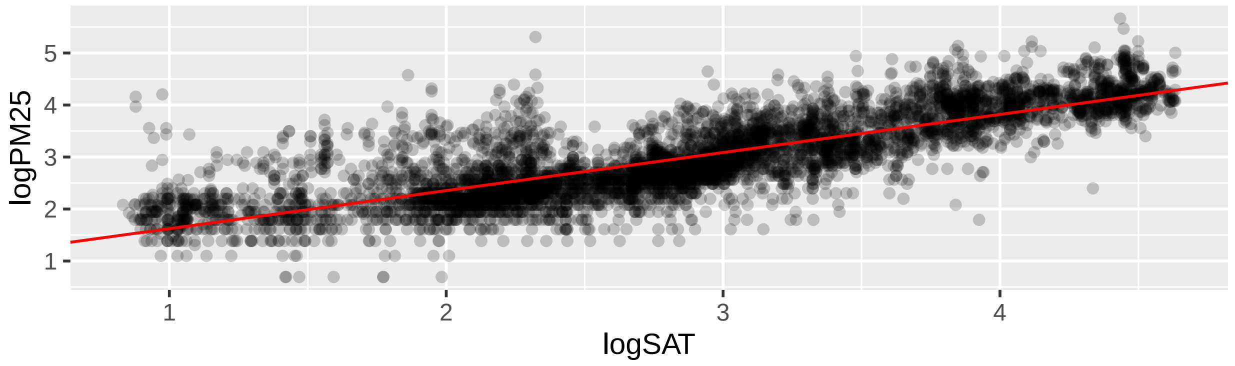
```
##              mean    sd 0.025quant 0.5quant 0.975quant
## (Intercept) 0.891 0.020      0.852    0.891      0.930
## logSAT      0.732 0.007      0.719    0.732      0.745
```

```
result_df = data.frame(param=as.factor('intercept'),
                       x=result$marginals.fixed$`(Intercept)`[,1],
                       y=result$marginals.fixed$`(Intercept)`[,2]) %>%
  bind_rows(data.frame(param=as.factor('slope'),
                       x=result$marginals.fixed$`logSAT`[,1],
                       y=result$marginals.fixed$`logSAT`[,2]))

result_df %>%
  ggplot(aes(x=x, y=y)) +
  geom_line() +
  facet_grid(param~., scales="free_y") +
  theme_light() +
  labs(x='Estimate')
```



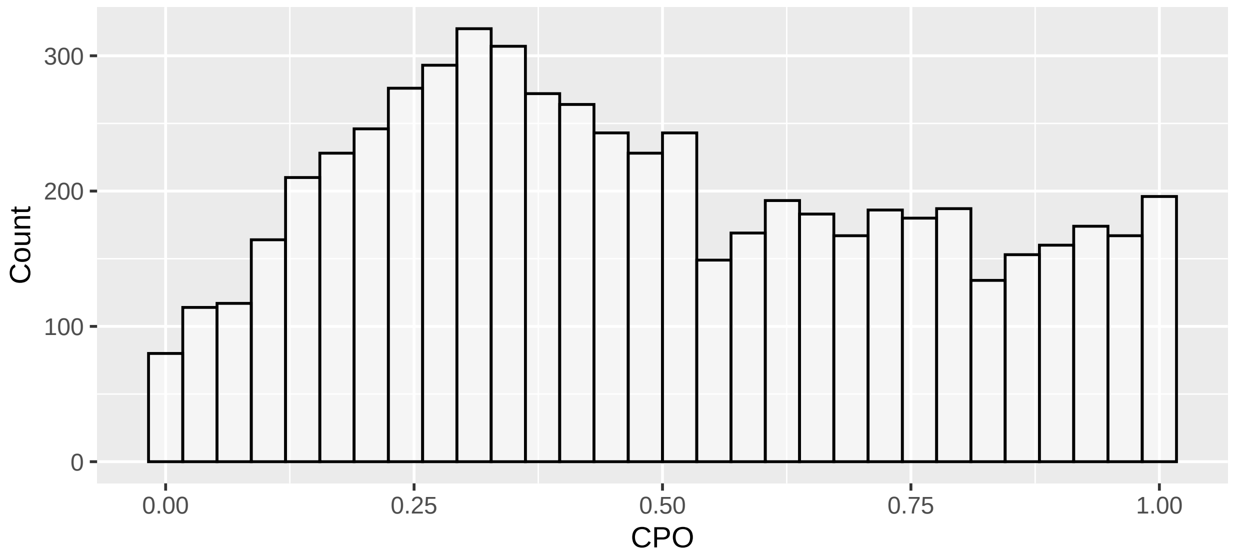
The plot of our regression line shows approximately how our satellite measurements will be callibreated:

```
d %>%
  ggplot(aes(x=logSAT, y=logPM25)) +
  geom_point(alpha=0.2)+
  geom_abline(aes(intercept=result$summary.fixed$mean[1], slope=result$summary.fixed$mean[2]), color='r
```



To understand the performance of this model, in our model specification, we requested a leave one out cross validation to be performed, i.e. $p(y_i|y_{-i})$. The frequency of values for $p(y_i|y_{-i})$ should be $\text{Uniform}(0, 1)$ since $F(Y) \sim \text{Uniform}(0, 1)$.

```
plot_data = data.frame(x=result$cpo$pit)

plot_data %>% ggplot(aes(x=x)) +
  geom_histogram(color='black', fill='white', alpha=0.5, bins=30) +
  labs(x='CPO', y='Count')
```

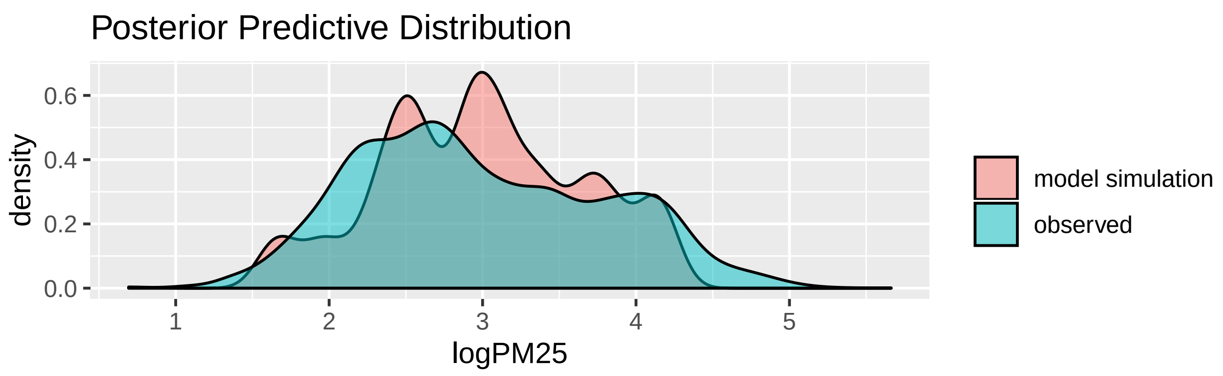


In this case, the variance of our predictive distribution is too small (peaked near center of distribution) and biased to take on smaller values too often (peak is off to the left). We believe that we can do better with improvements to our model.

If we look at our posterior predictive density, we see that this is approximately what the CPO graph is telling us.

```
samples = inla.posterior.sample(n=1, result = result)
samples_df = tibble(name='model simulation', values=samples[[1]]$latent)
samples_df = rbind(samples_df, tibble(name='observed', values=d$logPM25))
samples_df %>%
  ggplot(aes(x=values, fill=name)) +
```

```r
  geom_density(alpha=0.5) +
  labs(title='Posterior Predictive Distribution', fill='', x='logPM25')
```



The Independent and Hierarchical Models

In the previous section, we characterized the distribution of all the data with a single parameter $\theta = (\beta_0, \beta_1)$. That model treated observations as exchangeable, and dismissed further structural information we have about the data. This model is called a *pooled model*. There are natural spatial groupings, country and super-region, of our observations that are not taken advantage of. There are multiple measurements within each super region and further, there are multiple measurements made within each country. It is fair to believe that the calibration in measurement from satelite is going to vary based on the location on the globe that the measurement is made. Consider the hypothesis that atmospheric properties effect satelite measurements and atmospheric properties vary by region, then we can expect different parts of the globe to have varying calibraton needs.

At the other modelling extreme of our analysis so far is a θ for each grouping (e.g. country grouping) independently. We call this, namely, the *independent model*. This could be a great strategy given enough data for each country but as we've seen in the world map, some countries, have limited observations. Ideally, we would have a compromise between considering all observations at once and and a strict grouping by country.

Well, luckily, a compromise exists! Known as a *hierarchical model*, each grouping has its own parameter, like the independent case. So we have a collection of parameters, $\theta = (\theta_1, ..., \theta_n)$, but in the hierarchical model, θ_i come from the same distribution, parameterized by Ψ with its own prior distribution. So, we have an exchange of informatioin between observations.

In the last section, we built a pooled model. In this section, we will build an independent model and hierarchical model allowing the *intercept* to vary by super region to illustrate how the two parameter estimates change.

The independent model looks like the following. We have fixed effects for each super region.

$$\begin{aligned} Y_i &\sim N(\beta_{0k} + \beta_1 X_i, \sigma^2) \\ i = 1, ..., n \text{ observations} k &= 1, ..., K \text{ super regions} \end{aligned}$$

The hierarchical model is similar, but with one difference, the intercept is constructed with a random effect.

$$
\begin{aligned}
Y_i &\sim N(\beta_0 + \beta_1 X_i, \sigma^2)\\
\beta_0 &= b_0 + v_{0,k}\\
v_{0,k} &\sim N(0, \sigma^2_{v_0})\\
i &= 1, ..., n \text{ observations}\\
k &= 1, ..., K \text{ super regions}
\end{aligned}
$$

The computational tool we are using is INLA to perform thsi analysis. This tool requests the log precision of $v_{0,k}$ and a noninformative logGamma prior is assumed, which we will use.

```
d_new = d %>%
  mutate(sr_intercept = Super_region_name_1, sr_slope = Super_region_name_1,
         c_intercept = country_code_1, c_slope = country_code_1)

indep.model = logPM25 ~ 1 + factor(country_code_1) + logSAT

lcs <- inla.make.lincombs(id = diag(107),"(Intercept)" = rep(1,107))

indep_result = inla(formula = indep.model,
              data=d_new,
              family='gaussian',
              control.fixed = list(
                prec.intercept=1/5^2,
                prec = 1/5^2
              ),
              control.compute = list(cpo = TRUE, config = TRUE))
```

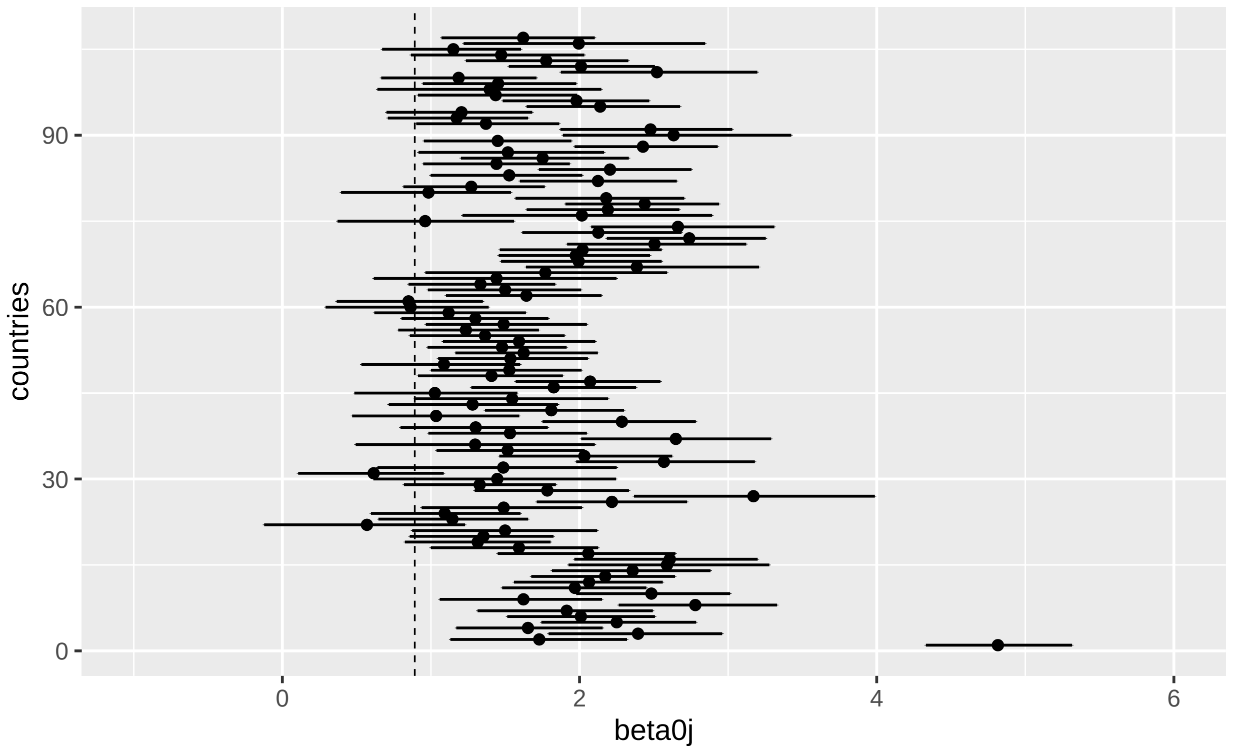
```
hierarchical.model = logPM25 ~ 1 + logSAT +
  #v_{1,k} term
  f(country_code_1, model = "iid", hyper = list(
    prec = list(prior = "pc.prec", param = list(1, 0.05))
  ))

hierarhical_result = inla(formula = hierarchical.model,
              data=d_new,
              family='gaussian',
              control.fixed = list(
                prec.intercept=1/5^2,
                prec = 1/5^2
              ),
              control.compute = list(cpo = TRUE, config = TRUE))
```

```
samples = inla.posterior.sample(n=1, result = indep_result)
samples_df = tibble(name='simulation', values=samples[[1]]$latent)
samples_df = rbind(samples_df, tibble(name='actual', values=d$logPM25))
p1 = samples_df %>%
  ggplot(aes(x=values, fill=name)) +
  geom_density(alpha=0.5) +
  labs(title='Posterior Predictive Distribution', fill='', x='logPM25')
```

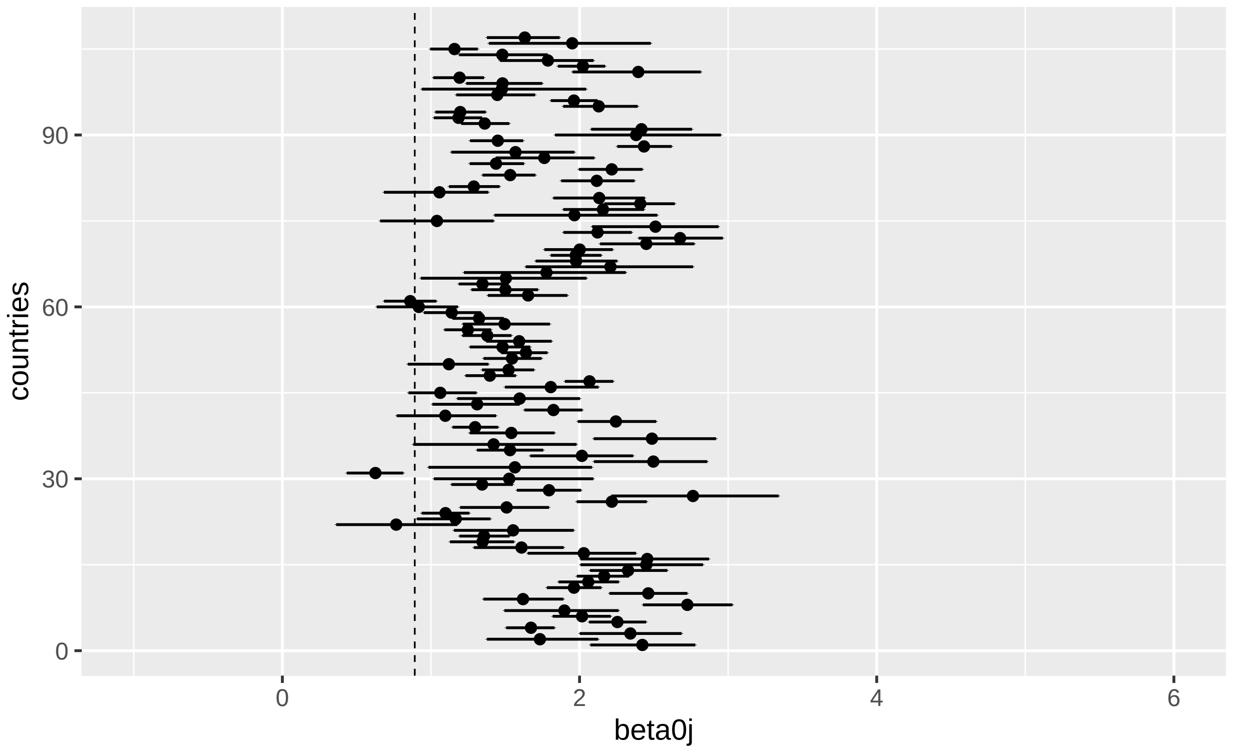
```r
b0 <- inla.rmarginal(1000, marg = hierarhical_result$marginals.fixed$'(Intercept)')
v0 <- matrix(NA,1000,107)
for(i in 1:107){
  v0[,i] <- inla.rmarginal(1000,marg = hierarhical_result$marginals.random$country_code_1[[i]])
}
beta0 <- b0 + v0
#Compute quartiles for beta0
hierarchical_beta0_quartiles <- t(apply(beta0, MARGIN=2,
  function(x) quantile( x, probs= c(0.025,0.5,0.975))))
```

```r
b0 <- inla.rmarginal(1000, marg = indep_result$marginals.fixed$'(Intercept)')
v0 <- matrix(NA,1000,107)
for(i in 1:107){
  v0[,i] <- inla.rmarginal(1000,marg = indep_result$marginals.fixed[[i]])
}
beta0 <- b0 + v0
#Compute quartiles for beta0
indep_beta0_quartiles <- t(apply(beta0, MARGIN=2,
  function(x) quantile( x, probs= c(0.025,0.5,0.975))))
```

```r
indep_intercept = tibble(ind = seq.int(nrow(indep_beta0_quartiles)), lower_ci=indep_beta0_quartiles[,1]

ggplot(data=indep_intercept, mapping=aes(x=ind, y=median)) + geom_point() +
  geom_errorbar(aes(ymin=lower_ci, ymax=upper_ci), width=0.1) +
  labs(y='beta0j', x='countries') +
  ylim(-1,6) +
  geom_hline(yintercept=result$summary.fixed[1,4], linetype='dashed', size=0.3) +
  coord_flip()
```



```
hier_intercept = tibble(ind = seq.int(nrow(hierarchical_beta0_quartiles)), lower_ci=hierarchical_beta0_

ggplot(data=hier_intercept, mapping=aes(x=ind, y=median)) + geom_point() +
  geom_errorbar(aes(ymin=lower_ci, ymax=upper_ci), width=0.1) +
  labs(y='beta0j', x='countries') +
  ylim(-1,6) +
  geom_hline(yintercept=result$summary.fixed[1,4], linetype='dashed', size=0.3) +
  coord_flip()
```



The last two charts show $\beta_0 j$ for the fixed effects of independent model and the random effects for the hierarchical model, respectively, across all 107 countries in our dataset. The median of the fixed intercept for the pooled model (from simple linear model earlier) is denoted by the dashed line. It is quite clear that the estimated parameters for ther hierarchiical model have smaller 95% credibility interrvals. This is an advantage of using the hierarchical model.

Now, to evaluate which model performs better, we can use the sum of CPO scores (described earlier) for the hierarchical model and independent model and take the difference:

```
sum(hierarhical_result$cpo$cpo) - sum(indep_result$cpo$cpo)
```

```
## [1] 3.899498
```

and between the hierarchical model and pooled model:

```
sum(hierarhical_result$cpo$cpo) - sum(result$cpo$cpo)
```

```
## [1] 1736.785
```

By this standard, we find that the hierarchical model outperforms both the independent model and pooled model. This method of sharing information between countries for the intercept parameter seems to be adding value.

Linear Model with IID Random Effects

In the last section we introduce a random effect country on the intercept with success. We would like to try introducing hierarchical parameters to both β_0 and β_1 for both country and super region.

Our new model these IID random effects is defined as,

$$\begin{aligned}
\mu_i &= \beta_0 + \beta_1 X_i \\
\beta_0 &= b_0 + u_{0,j} + v_{1,k} \\
\beta_1 &= b_1 + u_{1,j} + v_{1,k} \\
i &= 1, ..., n \text{ measurements} \\
j &= 1, ..., J \text{ countries} \\
k &= 1, ..., K \text{ super regions}
\end{aligned}$$

where j is index for super region and k is index of country. So, b_0 and b_1 are fixed effects on intercept and slope, $u_{1,j}$ and $v_{1,k}$ are random effects on intercept for super region and country and $u_{2,i}$ and $v_{2,j}$ are random effects on slope for super region and country, respectively.

We need to set priors on the u's and v's. We will use the same priors for β's as in simple linear regression for all intercept and slope parameters. I believe that it is very unlikely that the intercept random effects are unlikely to be greater than 3 since like we said earlier, reasonable levels of PM25 are below 150. On the log scale that means below approximately 2.2. So we will apply a prior on the intercept such that we receive a value greater than 3, 10% of time. For the slope, I don't expect that the calibrations will be off by an extreme amount. Again, perfect calibration would be a slope of 1. I expect it to be unlikely that we get a slope greater than 5. So we will say we get a slope gretater than 5, 10% of the time.

```r
d = d %>%
  mutate(sr_intercept = Super_region_name_1, sr_slope = Super_region_name_1,
         c_intercept = country_code_1, c_slope = country_code_1)

sr_and_c_re_model = logPM25 ~ 1 + logSAT +
  #u_{1,j} term
  f(sr_intercept, model = "iid", hyper = list(
    prec = list(prior = "pc.prec", param = list(3, 0.1))
)) +
  #u_{2,j} term
  f(sr_slope, logSAT, model='iid', hyper=list(
    prec = list(prior = "pc.prec", param = list(3, 0.1))
)) +
  #v_{1,k} term
  f(c_intercept, model = "iid", hyper = list(
    prec = list(prior = "pc.prec", param = list(5, 0.1))
)) +
  #v_{2,k} term
  f(c_slope, logSAT, model='iid', hyper=list(
    prec = list(prior = "pc.prec", param = list(5, 0.1))
))
```

```r
sr_and_c_re_result = inla(formula = sr_and_c_re_model,
              data=d,
              family='gaussian',
              control.fixed = list(
                prec.intercept=1/5^2,
                prec = 1/5^2
              ),
              control.compute = list(cpo = TRUE, config=TRUE))
```

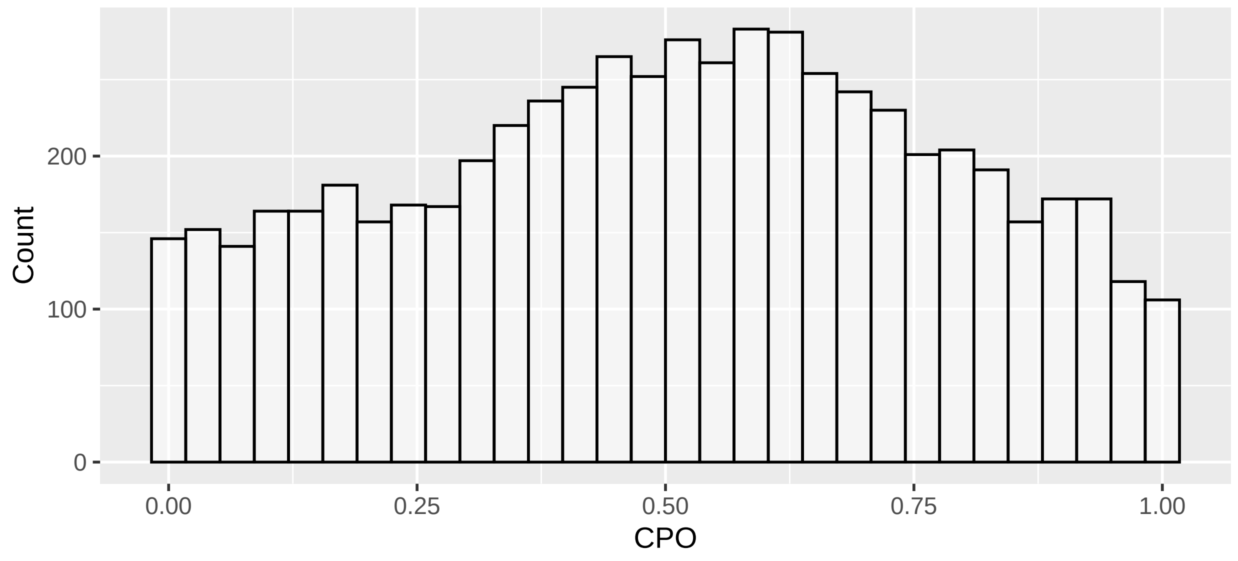
How does this new model compare with a single random effect for country on the intercept (from the last section)? Again, using our CPO measure of fit.

```r
sum(sr_and_c_re_result$cpo$cpo) - sum(hierarhical_result$cpo$cpo)
```

```
## [1] 137.2551
```

We have outperformed the last model with our new IID random effects including slope and random effect for super region on intercetpt.

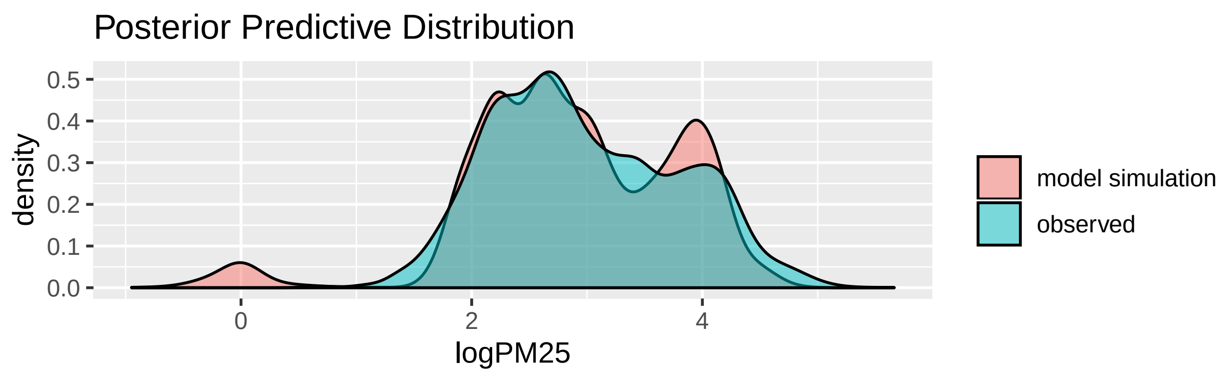
```r
# hist(result$cpo$pit, breaks=30)
plot_df = data.frame(x=sr_and_c_re_result$cpo$pit)

plot_df %>% ggplot(aes(x=x)) +
  geom_histogram(color='black', fill='white', alpha=0.5, bins=30) +
  labs(x='CPO', y='Count')
```



The sum of CPO scores has increased in this new model so we have improve the overall fit of the model. The variance of our prediction is still too small however.

Let us also evaluate the posterior predictive density (PPD) to compare with the PPD from the pooled model early on.

```
samples = inla.posterior.sample(n=1, result = sr_and_c_re_result)
samples_df = tibble(name='model simulation', values=samples[[1]]$latent)
samples_df = rbind(samples_df, tibble(name='observed', values=d$logPM25))
samples_df %>%
  ggplot(aes(x=values, fill=name)) +
  geom_density(alpha=0.5) +
  labs(title='Posterior Predictive Distribution', fill='', x='logPM25')
```



It appears that our observed data is close to the replicated data from our model. It looks better than the replicated data from the pooled model earlier in this analysis.

Linear Model with IID Super Region Random Effects and Country Spatially Correlated Country Random Effects

The model specification from the previous model is similar in this next iteration but we are introducing spatial correlation to the random effect representing hierarchical structure of country measurement, $u_{0,j}$ and $u_{1,j}$.

In the last model, all θ_i were independent of each other. Now, with new spatial dependency, θ_i is independent of all θ_j for countries i given the set of its neighbouors, $\mathcal{N}(i)$. In other words, $\theta_i \perp\!\!\!\perp \theta_{-i}|\theta_{\mathcal{N}(i)}$ where θ_{-i} indicates all elements of θ not includnig θ_i.

In the case of country i having n neighbors, then

$$u_i|u_{-i} \sim \text{Normal}(\mu_i + \sum_{j=1}^{n} r_{i,j}(u_j - \mu_j), s_i^2)$$

This specification is called initrinsic conditional autoregressive (iCAR) and along with the country random effects defined in part 2, is known as the Besag-York-Mollie (BYM) model. In our case, $r_{i,j}$ is 1 when countries are adjacent, 0 otherwise and $s_i^2 = \sigma^2/\mathcal{N}_\rangle$. So, replacing the definition of country IID random effects with the iCAR based on country adjacency, we can define a new model. We can continue to use the priors that we have defined in previous sections.

```
d = d %>%
  mutate(sr_intercept = Super_region_name_1, sr_slope = Super_region_name_1,
         c_intercept = country_code_1, c_slope = country_code_1)

sr_and_c_re_model = logPM25 ~ 1 + logSAT +
  #u_{1,j} term
  f(sr_intercept, model = "iid", hyper = list(
    prec = list(prior = "pc.prec", param = list(3, 0.1))
)) +
  #u_{2,j} term
  f(sr_slope, logSAT, model='iid', hyper=list(
    prec = list(prior = "pc.prec", param = list(3, 0.1))
)) +
  #v_{1,k} term
  f(c_intercept, model = "bym2", graph=here('data', 'world.adj'), hyper = list(
    prec = list(prior = "pc.prec", param = list(5, 0.1))
)) +
  #v_{2,k} term
  f(c_slope, logSAT, model='bym2', graph=here('data', 'world.adj'), hyper=list(
    prec = list(prior = "pc.prec", param = list(5, 0.1))
))

sr_and_c_corr_re_result = inla(formula = sr_and_c_re_model,
              data=d,
              family='gaussian',
              control.fixed = list(
                prec.intercept=1/5^2,
                prec = 1/5^2
              ),
              control.compute = list(cpo = TRUE, config=TRUE))
```

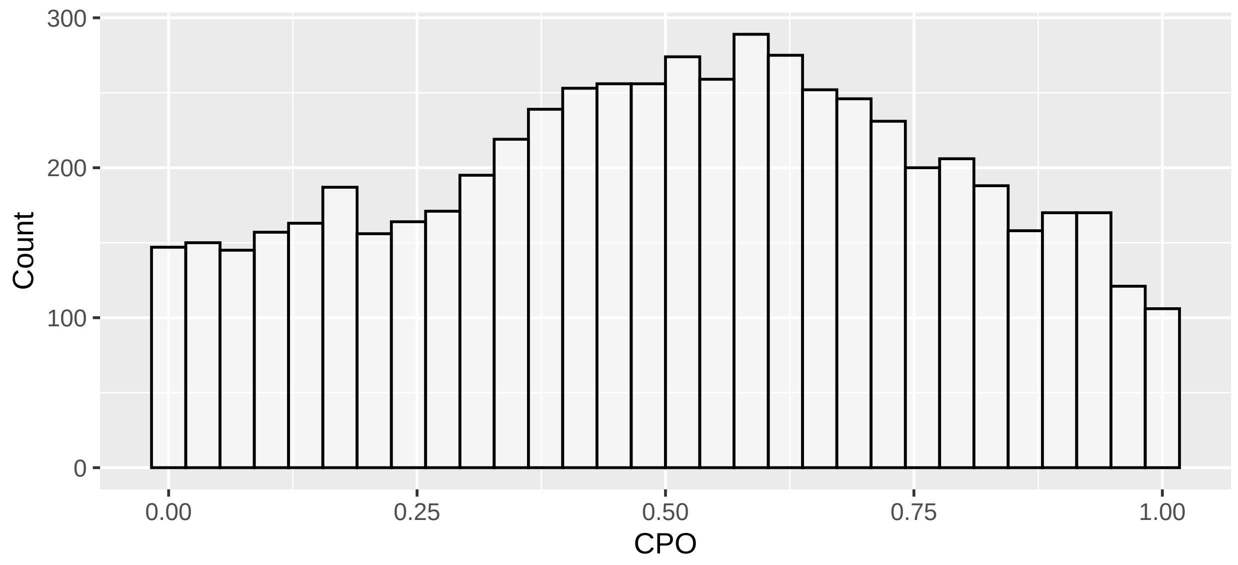
How does this model compare to the last?

```
sum(sr_and_c_corr_re_result$cpo$cpo) - sum(sr_and_c_re_result$cpo$cpo)
```

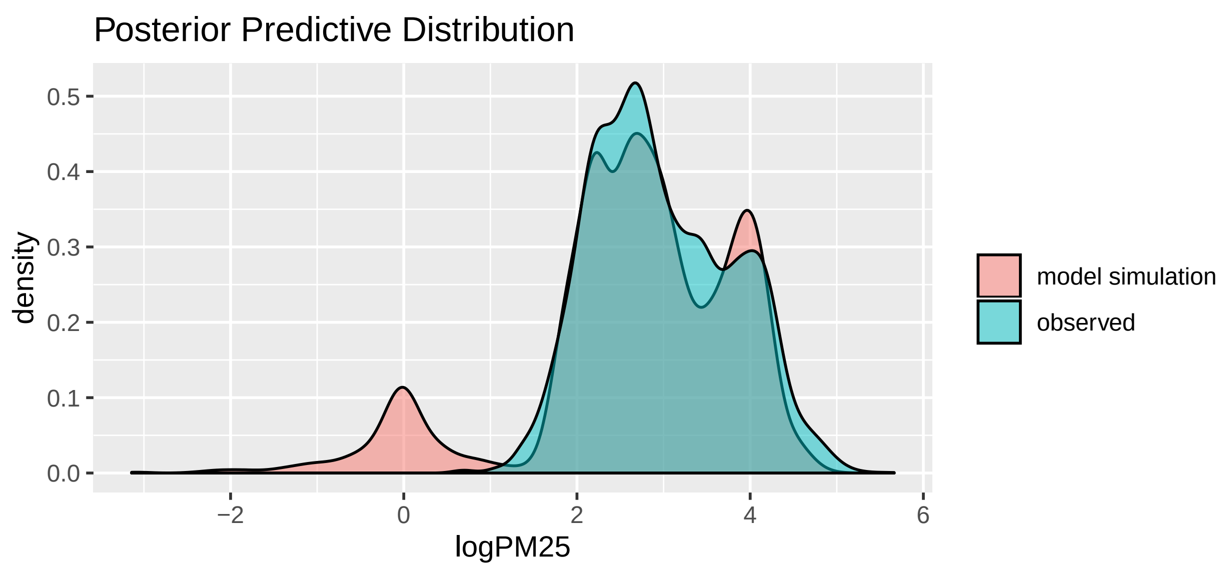
```
## [1] -1.222651
```

The CPO score indicates that it in fact underperformed the previous model, without the spatial correlation.

```
# hist(result$cpo$pit, breaks=30)
plot_df = data.frame(x=sr_and_c_corr_re_result$cpo$pit)

plot_df %>% ggplot(aes(x=x)) +
  geom_histogram(color='black', fill='white', alpha=0.5, bins=30) +
  labs(x='CPO', y='Count')
```



```r
samples = inla.posterior.sample(n=1, result = sr_and_c_corr_re_result)
samples_df = tibble(name='model simulation', values=samples[[1]]$latent)
samples_df = rbind(samples_df, tibble(name='observed', values=d$logPM25))
samples_df %>%
  ggplot(aes(x=values, fill=name)) +
  geom_density(alpha=0.5) +
  labs(title='Posterior Predictive Distribution', fill='', x='logPM25')
```



Limiting the sharing of information between adjacent countries according to CPO suggests that this new version of the model is not performing as well

Model Performance Measure Considerations

So far in our analysis, we have been using LOO cross validation, summing LOO CPO and comparing between models. This sort of cross validation assumes that training and evaluation data are independent.

Here, though, error estimates will be too optimistic since we are dealing with correlated data.

Our data is correlated due to the hierarchical, spatial structure of the data we are working with. We have semi-repeated measurements for each country (not quite strictly repeated since the measurement locations are spread throughout country) and correlation between countries that are nearby to each other.

To deal with this, we would like to remove chunks of data that are correlated for testing. Train on the other data and then test on this held out spatially correlated data. This way, the correlation of nearby data won't artificially increase the CPO as in the LOO method has done thus far.

This is also referred to as *block cross validation*. In this case, I will be creating three different train-test sets where the test set is a super-region that defines a geographic space that are spatially close and training set is all observations excluding those from that super region. There are 6003 observations so to get a test set that is approximately 10% of total observations, we need 600 observations. The following shows the number of observations by super region:

```r
kable(d %>%
  group_by(Super_region_name_1) %>%
  summarise(number_of_observations = n()))
```

Super_region_name_1	number_of_observations
Central Europe, Eastern Europe, Central Asia	518
High income	3051
Latin America and Caribbean	308
North Africa / Middle East	273
South Asia	464
Southeast Asia, East Asia and Oceania	1312
Sub-Saharan Africa	77

We will use: (1) *Central Europe, Eastern Europe, Central Asia*; (2) *South Asia*; (3) *Latin American and Carribean + North Africa / Middle East* as our three test sets for the three train-test sets. The following are root mean squared erors (RMSE) averaged across the three test sets for three of our models constructed above:

```r
train_1 = d %>% filter(Super_region_name_1 != 'Central Europe, Eastern Europe, Central Asia')
test_1 = d %>% filter(Super_region_name_1 == 'Central Europe, Eastern Europe, Central Asia')
train_test_1 = train_1 %>% rbind(test_1 %>% mutate(logPM25 = NA))
train_1_size = dim(train_1)[[1]]

train_2 = d %>% filter(Super_region_name_1 != 'South Asia')
test_2 = d %>% filter(Super_region_name_1 == 'South Asia')
train_test_2 = train_2 %>% rbind(test_2 %>% mutate(logPM25 = NA))
train_2_size = dim(train_2)[[1]]

train_3 = d %>% filter(Super_region_name_1 != 'Latin America and Caribbean' &
                       Super_region_name_1 != 'North Africa / Middle East')
test_3 = d %>% filter(Super_region_name_1 == 'Latin America and Caribbean' |
                      Super_region_name_1 == 'North Africa / Middle East')
train_test_3 = train_3 %>% rbind(test_3 %>% mutate(logPM25 = NA))
train_3_size = dim(train_3)[[1]]
```

Model 1 - Simple Linear Model:

```
rmse1 = 0
for (i in 1:3){
  plain_lm_model = logPM25 ~ 1 + logSAT
  train_test_set = eval(as.name(paste0('train_test_',i)))
  test_set = eval(as.name(paste0('test_',i)))
  result = inla(formula = plain_lm_model,
              data=train_test_set,
              family='gaussian',
              control.family=list(hyper=list(
                                    prec=list(prior="normal",param=c(0,1)))),
                                    #could not find half-normal distribution
              control.fixed = list(
                prec.intercept=1/5^2,
                prec = 1/5^2
              ),
              control.compute = list(config=TRUE, cpo = TRUE))
  rmse1 = rmse1 + sqrt(mean((test_set$logPM25 - result$summary.fitted.values$mean[(eval(as.name(paste0(
}
rmse1 = rmse1 / 3
```

```
rmse1
```

```
## [1] 0.5100959
```

Model 2 - Linear Model with IID Random Effects:

```
rmse2 = 0
sr_and_c_re_model = logPM25 ~ 1 + logSAT +
  #u_{1,j} term
  f(sr_intercept, model = "iid", hyper = list(
    prec = list(prior = "pc.prec", param = list(3, 0.1))
)) +
  #u_{2,j} term
  f(sr_slope, logSAT, model='iid', hyper=list(
    prec = list(prior = "pc.prec", param = list(3, 0.1))
)) +
  #v_{1,k} term
  f(c_intercept, model = "iid", hyper = list(
    prec = list(prior = "pc.prec", param = list(5, 0.1))
)) +
  #v_{2,k} term
  f(c_slope, logSAT, model='iid', hyper=list(
    prec = list(prior = "pc.prec", param = list(5, 0.1))
))

for (i in 1:3){
  train_test_set = eval(as.name(paste0('train_test_',i)))
  test_set = eval(as.name(paste0('test_',i)))
  result = inla(formula = sr_and_c_re_model,
              data=train_test_set,
              family='gaussian',
              control.fixed = list(
                prec.intercept=1/5^2,
                prec = 1/5^2
              ),
              control.compute = list(cpo = TRUE, config=TRUE))
  rmse2 = rmse2 + sqrt(mean((test_set$logPM25 - result$summary.fitted.values$mean[(eval(as.name(paste0(
}
```

```
rmse2
```

```
## [1] 1.440505
```

Model 3 - Linear Model with IID Super Region Random Effects and Country Spatially Correlated Random Effects:

```
rmse3=0
sr_and_c_re_model = logPM25 ~ 1 + logSAT +
  #u_{1,j} term
  f(sr_intercept, model = "iid", hyper = list(
    prec = list(prior = "pc.prec", param = list(3, 0.1))
)) +
  #u_{2,j} term
  f(sr_slope, logSAT, model='iid', hyper=list(
    prec = list(prior = "pc.prec", param = list(3, 0.1))
)) +
  #v_{1,k} term
  f(c_intercept, model = "bym2", graph=here('data', 'world.adj'), hyper = list(
    prec = list(prior = "pc.prec", param = list(5, 0.1))
)) +
  #v_{2,k} term
  f(c_slope, logSAT, model='bym2', graph=here('data', 'world.adj'), hyper=list(
    prec = list(prior = "pc.prec", param = list(5, 0.1))
))

for (i in 1:3){
  train_test_set = eval(as.name(paste0('train_test_',i)))
  test_set = eval(as.name(paste0('test_',i)))
  train_size=eval(as.name(paste0('train_',i, '_size')))
  result = inla(formula = sr_and_c_re_model,
              data=train_test_set,
              family='gaussian',
              control.fixed = list(
                prec.intercept=1/5^2,
                prec = 1/5^2
              ),
              control.compute = list(cpo = TRUE, config=TRUE))
  rmse3 = rmse3 + sqrt(mean((test_set$logPM25 - result$summary.fitted.values$mean[(train_size+1):6003])
}
```

```
rmse3
```

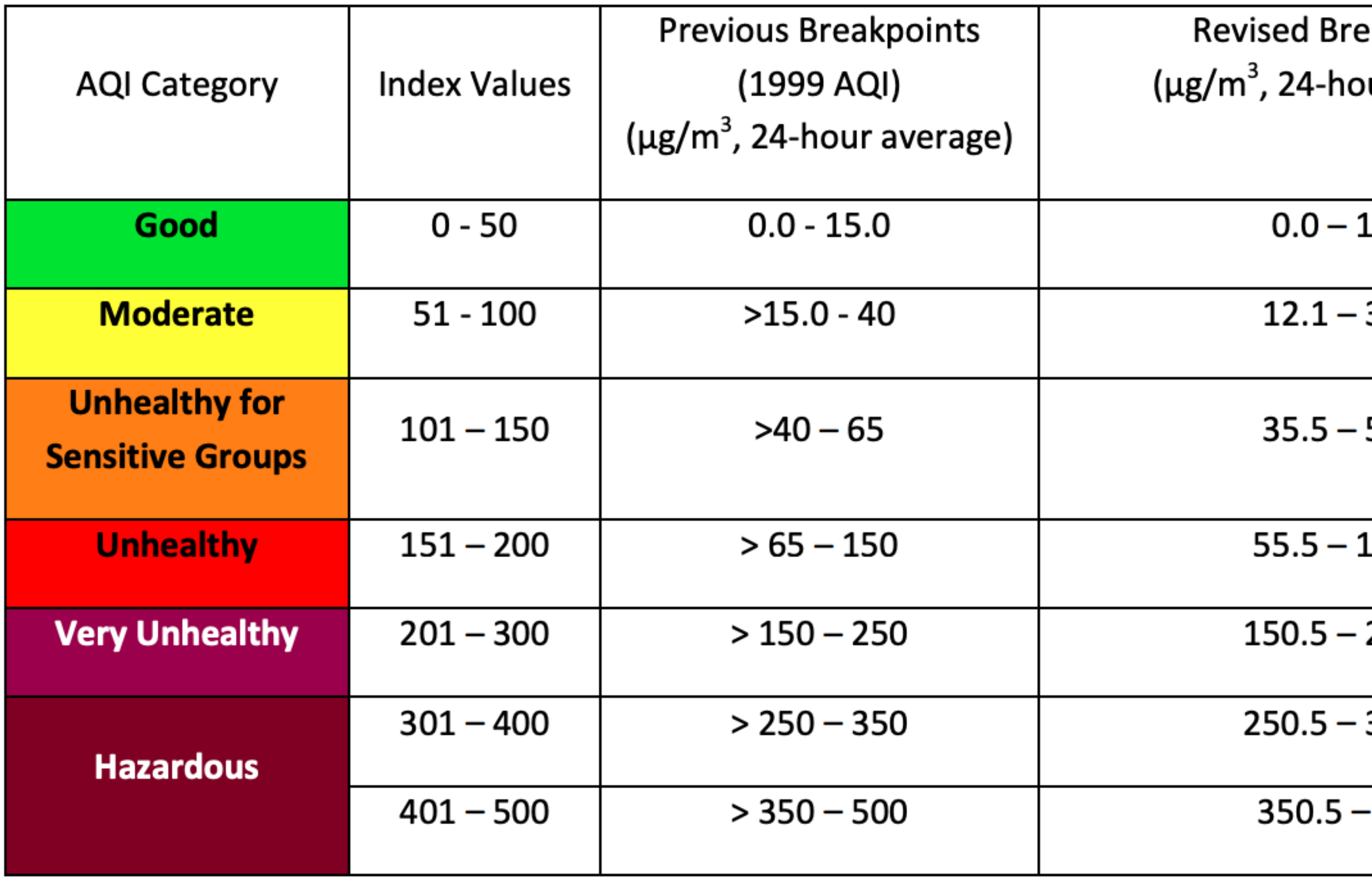
```
## [1] 1.418411
```

There are two key findings here. First, with block cross validation, Model 3 outperforms Model 2 unlike prevsiouly measured using LOO CV. Second, simple linear regression outperforms both of the more complex models. Our earlier hypothesis that model performace as measured with LOO CV will be optimistic in presence of correlated data seems valid. Our complex models are overfitting.

Depending on the use case, the simple linear regressino may be preferred. If we had huge portions of the globe with no measurements, we would prefer to use the simple linear regression since it does not overfit to what we have already seen like we have seen using block cross validaiton. However, we do have quite a bit of information about each part of the globe. I would recommend Model 3 as it captures the correlated measures that are important in trying to calibrate new locations on the globe but also slightly less susceptible to spurious correlations than Model 2 as we've seen here.

Appendix

```r
library(knitr)
include_graphics(here('data/epa_standards.png'))
```



AQI Category	Index Values	Previous Breakpoints (1999 AQI) ($\mu g/m^3$, 24-hour average)	Revised Bre ($\mu g/m^3$, 24-hou
Good	0 - 50	0.0 - 15.0	0.0 – 1
Moderate	51 - 100	>15.0 - 40	12.1 – 3
Unhealthy for Sensitive Groups	101 – 150	>40 – 65	35.5 – 5
Unhealthy	151 – 200	> 65 – 150	55.5 – 1
Very Unhealthy	201 – 300	> 150 – 250	150.5 – 2
Hazardous	301 – 400	> 250 – 350	250.5 – 3
	401 – 500	> 350 – 500	350.5 –

via https://www.epa.gov/sites/production/files/2016-04/documents/2012_aqi_factsheet.pdf