Hierarchical model with risk level intercept

Laura Balasso

1/3/2021

Hierarchical model for italian regions

```
regions <- unique(data_it$region)
regions
```

```
##  [1] "Abruzzo"               "Basilicata"            "Calabria"
##  [4] "Campania"              "Emilia-Romagna"        "Friuli Venezia Giulia"
##  [7] "Lazio"                 "Liguria"               "Lombardia"
## [10] "Marche"                "Molise"                "P.A. Bolzano"
## [13] "P.A. Trento"           "Piemonte"              "Puglia"
## [16] "Sardegna"              "Sicilia"               "Toscana"
## [19] "Umbria"                "Valle d'Aosta"         "Veneto"
```

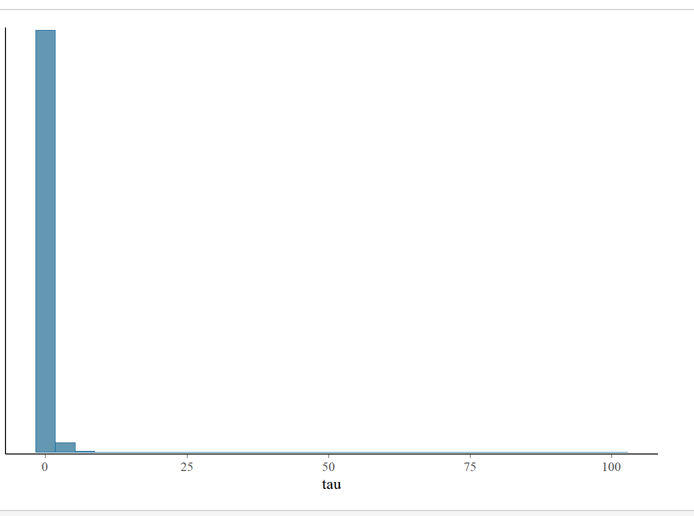
```
hier_data <- get_hier_data(data_it, regions, Y_delay, O_delay, R_delay, initial_date = as.Date('2020-10-15'), fin
al_date = as.Date('2020-12-22') )

risk_idx <- matrix(data = NA, nrow = nrow(hier_data$yellow_dummies), ncol = ncol(hier_data$yellow_dummies))
for(i in 1:nrow(risk_idx)){
  for(j in 1:ncol(risk_idx)){
    if(hier_data$yellow_dummies[i,j] == 1) risk_idx[i, j] <- 2
    else if(hier_data$orange_dummies[i,j] == 1) risk_idx[i,j] <- 3
    else if(hier_data$red_dummies[i,j]==1) risk_idx[i,j] <- 4
  }
}

risk_idx[is.na(risk_idx)] <- 1

p_delay <- get_delay_distribution()


stan_data_hier <- list(J = length(regions),
                       N = nrow(hier_data$exposures),
                       N_nonzero = length(hier_data$nonzero_days),
                       nonzero_days = hier_data$nonzero_days,
                       conv_gt = get_gt_convolution_ln2(nrow(hier_data$exposures)),
                       length_delay = length(p_delay),
                       p_delay = p_delay,
                       exposures = hier_data$exposures,
                       nonzero_positives = hier_data$positives[hier_data$nonzero_days , ],
                       risk_idx = risk_idx[hier_data$nonzero_days , ]
                       )


compiled_hier <- stan_model('../stan/hier_area_intercepts.stan')

fit_hier <- sampling(compiled_hier, data = stan_data_hier, iter= 2000)
```

```
print(fit_hier, pars='tau')
```

```
## Inference for Stan model: hier_area_intercepts.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##     mean se_mean   sd 2.5%  25%  50% 75% 97.5% n_eff Rhat
## tau 0.32    0.03 1.74 0.01 0.03 0.09 0.3  1.77  3772    1
##
## Samples were drawn using NUTS(diag_e) at Wed Jan 06 12:41:26 2021.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
mcmc_hist(fit_hier, pars='tau')
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

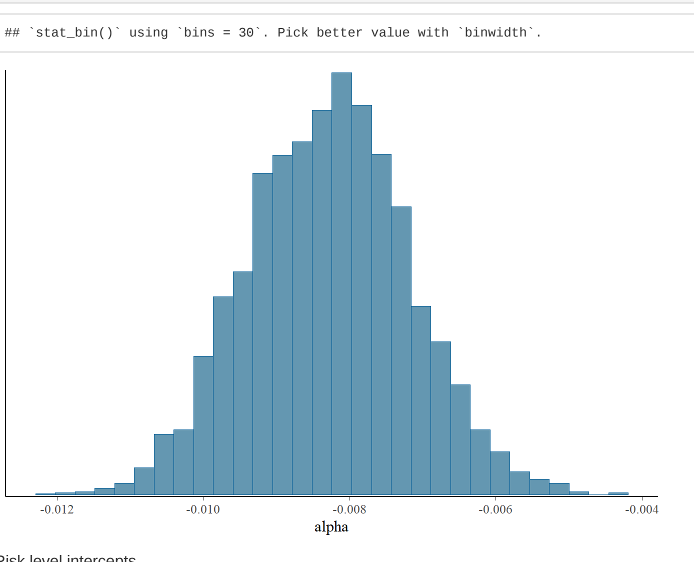



```
print(fit_hier, pars = 'alpha')
```

```
## Inference for Stan model: hier_area_intercepts.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##        mean se_mean sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
## alpha -0.01       0  0 -0.01 -0.01 -0.01 -0.01 -0.01  3139    1
##
## Samples were drawn using NUTS(diag_e) at Wed Jan 06 12:41:26 2021.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
mcmc_hist(fit_hier, pars = 'alpha')
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



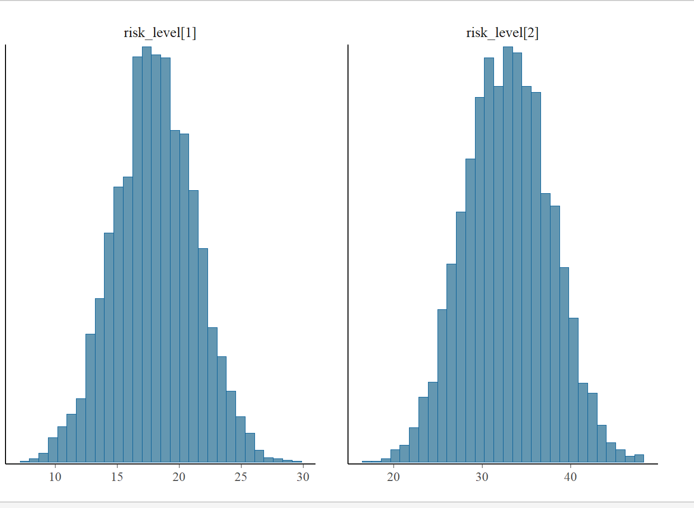
Risk level intercepts

```
print(fit_hier, pars = 'risk_level')
```

```
## Inference for Stan model: hier_area_intercepts.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##                mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
## risk_level[1] 18.03    0.05 3.24 11.59 15.89 18.00 20.24 24.40  4296    1
## risk_level[2] 32.89    0.09 4.79 23.60 29.54 32.89 36.20 42.25  2910    1
## risk_level[3] 29.43    0.08 4.74 20.13 26.12 29.44 32.58 38.87  3322    1
## risk_level[4] 15.61    0.08 3.10  8.17 14.04 16.02 17.70 20.51  1604    1
##
## Samples were drawn using NUTS(diag_e) at Wed Jan 06 12:41:26 2021.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

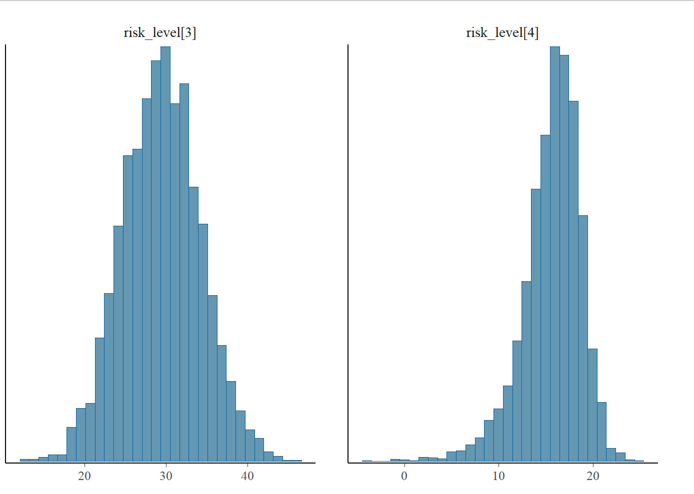
```
mcmc_hist(fit_hier, pars = c('risk_level[1]', 'risk_level[2]'))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
mcmc_hist(fit_hier, pars = c('risk_level[3]', 'risk_level[4]'))
```

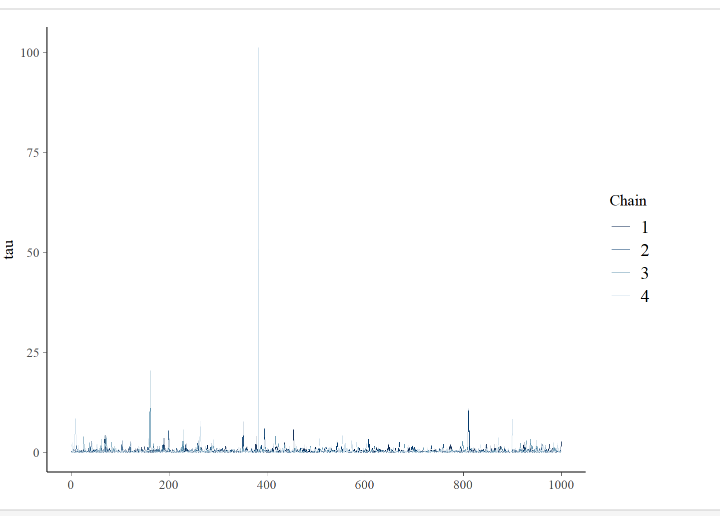
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Trace plots

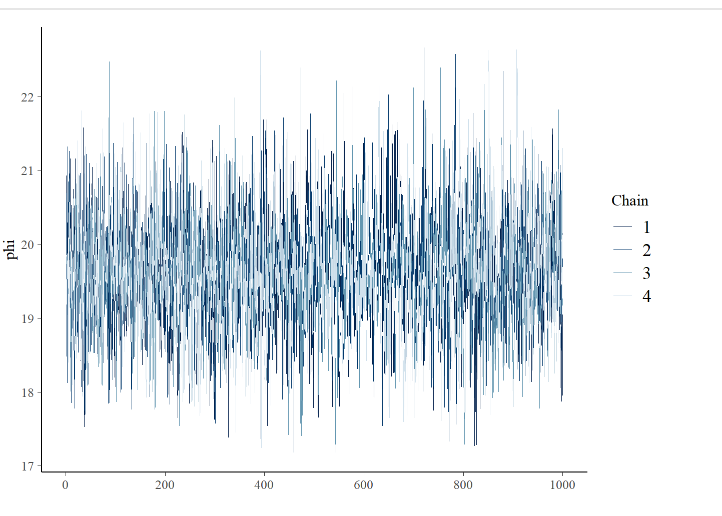
```
mcmc_trace(as.array(fit_hier, pars = c('tau')),
           np = nuts_params(fit_hier)
)
```

```
## No divergences to plot.
```



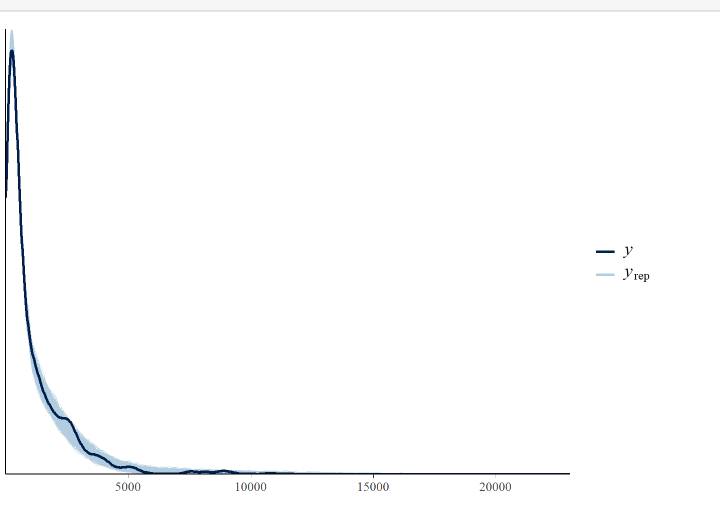

```
mcmc_trace(as.array(fit_hier, pars = c('phi')),
           np = nuts_params(fit_hier)
)
```

```
## No divergences to plot.
```



Posterior predictive check

```
y_rep <- as.matrix(fit_hier, pars = "y_rep")
ppc_dens_overlay(y = as.vector(stan_data_hier$nonzero_positives), y_rep[1:1000, ])
```



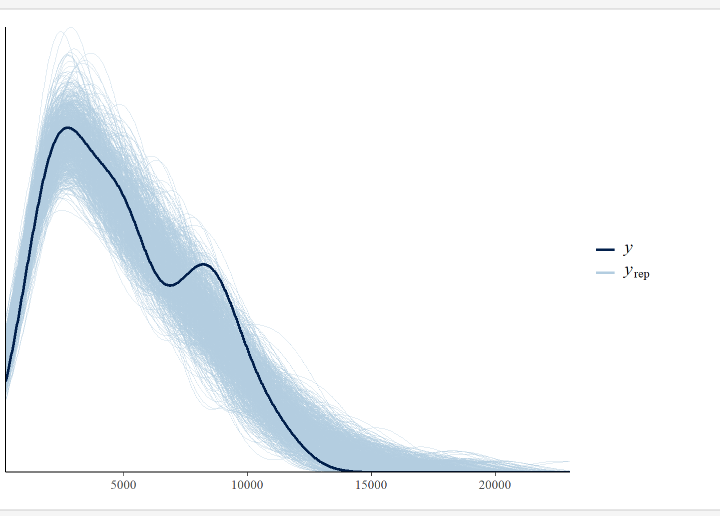
Posterior predictive check by region

```
regional_yrep_idx <- function(region, regions_vector, nonzero_days){
  region_idx <- which(regions_vector == region)
  yrep_idx <- (region_idx-1)* length(nonzero_days) + 1
  range <- yrep_idx : (yrep_idx + length(nonzero_days)-1)
  return(range)
}
```

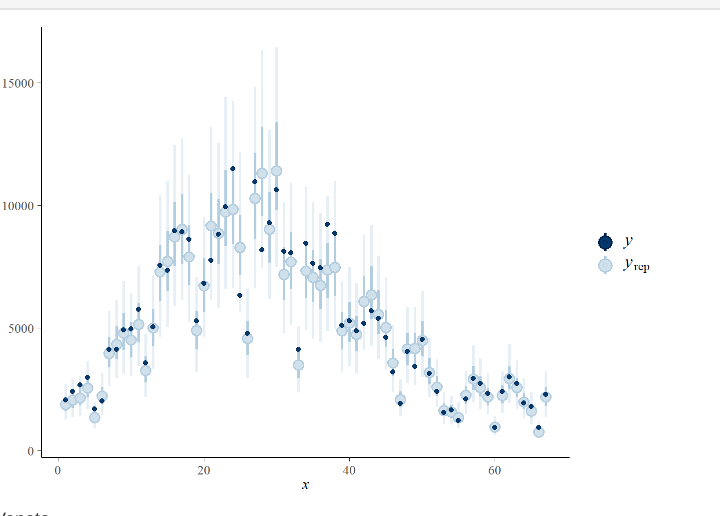
```
groups <- function(regions, nonzero_days){
  group <- rep(regions[1], length(nonzero_days))
  for(r in 2:length(regions))
    group <- c(group, rep(regions[r], length(nonzero_days)))

  return(group)
}
```

Lombardia

```
ppc_dens_overlay(y = stan_data_hier$nonzero_positives[, which(regions == 'Lombardia')], y_rep[1:1000,regional_yre
p_idx('Lombardia', regions, stan_data_hier$nonzero_days)])
```

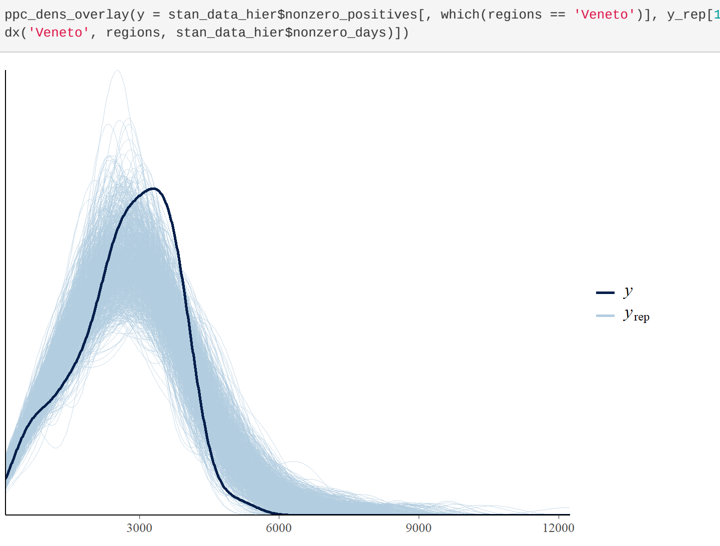



```
ppc_intervals(y = stan_data_hier$nonzero_positives[, which(regions == 'Lombardia')], y_rep[1:1000,regional_yrep_i
dx('Lombardia', regions, stan_data_hier$nonzero_days)])
```

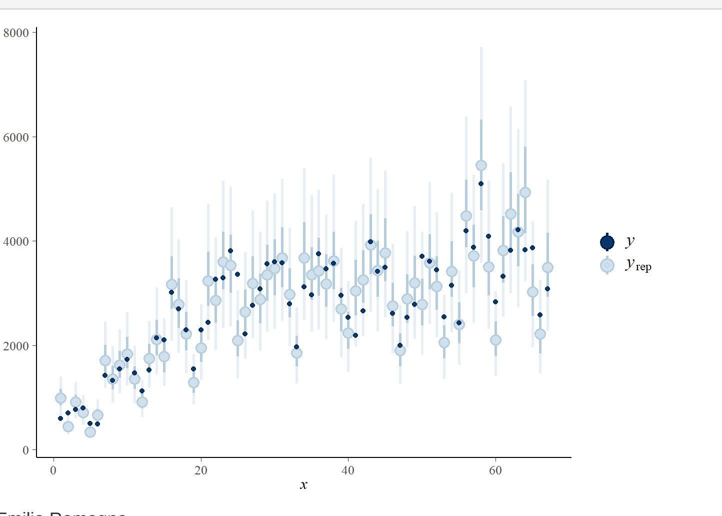


Veneto

```
ppc_dens_overlay(y = stan_data_hier$nonzero_positives[, which(regions == 'Veneto')], y_rep[1:1000,regional_yrep_i
dx('Veneto', regions, stan_data_hier$nonzero_days)])
```

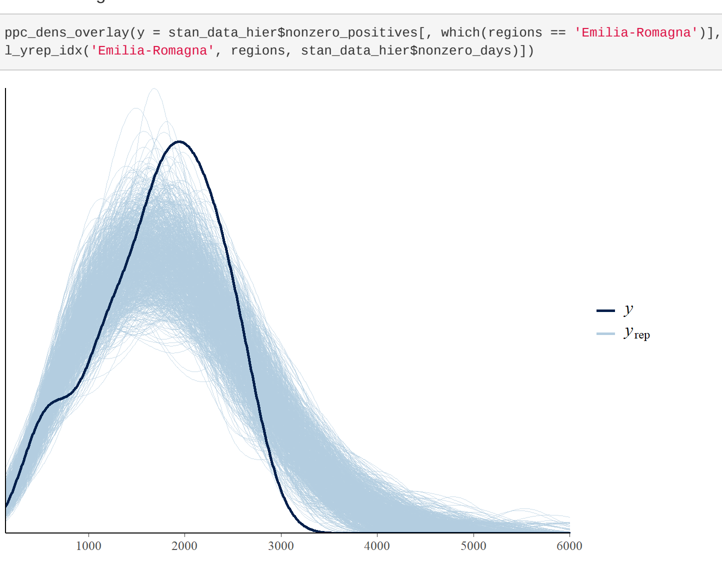



```
ppc_intervals(y = stan_data_hier$nonzero_positives[, which(regions == 'Veneto')], y_rep[1:1000,regional_yrep_idx(
'Veneto', regions, stan_data_hier$nonzero_days)])
```

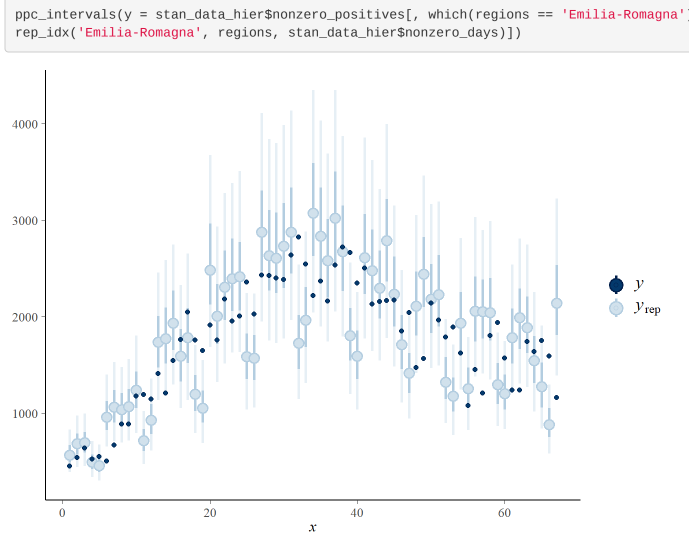


Emilia Romagna

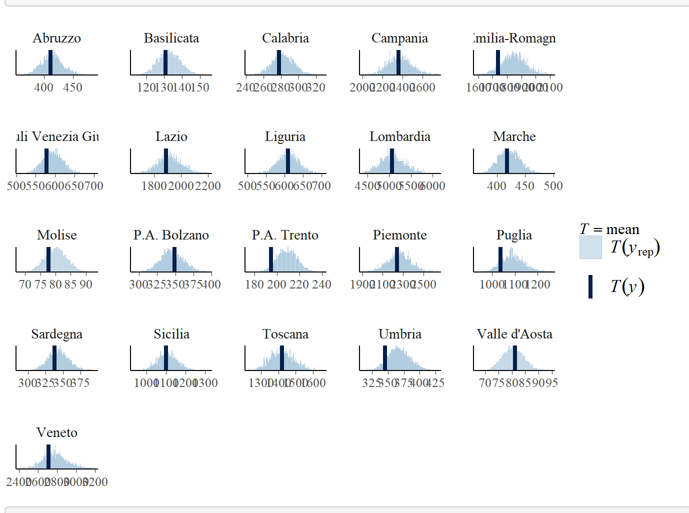
```
ppc_dens_overlay(y = stan_data_hier$nonzero_positives[, which(regions == 'Emilia-Romagna')], y_rep[1:1000,regiona
l_yrep_idx('Emilia-Romagna', regions, stan_data_hier$nonzero_days)])
```



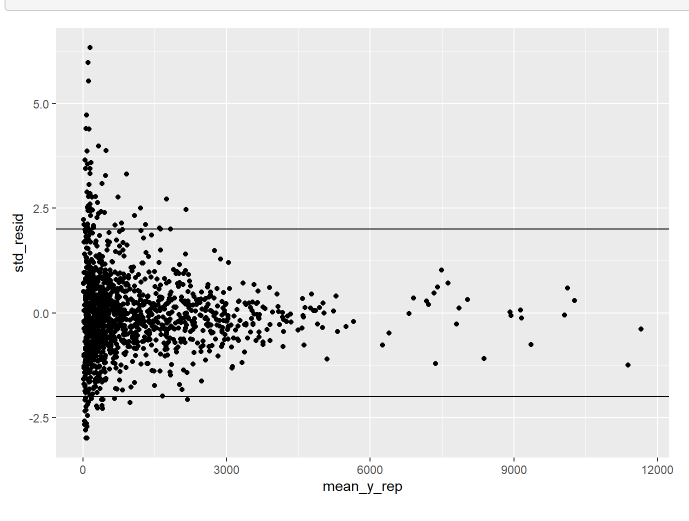

```
ppc_intervals(y = stan_data_hier$nonzero_positives[, which(regions == 'Emilia-Romagna')], y_rep[1:1000,regional_y
rep_idx('Emilia-Romagna', regions, stan_data_hier$nonzero_days)])
```




```
ppc_stat_grouped(y=as.vector(stan_data_hier$nonzero_positives), yrep =y_rep, group = groups(regions, stan_data_hi
er$nonzero_days) ,stat="mean", binwidth=0.5)
```

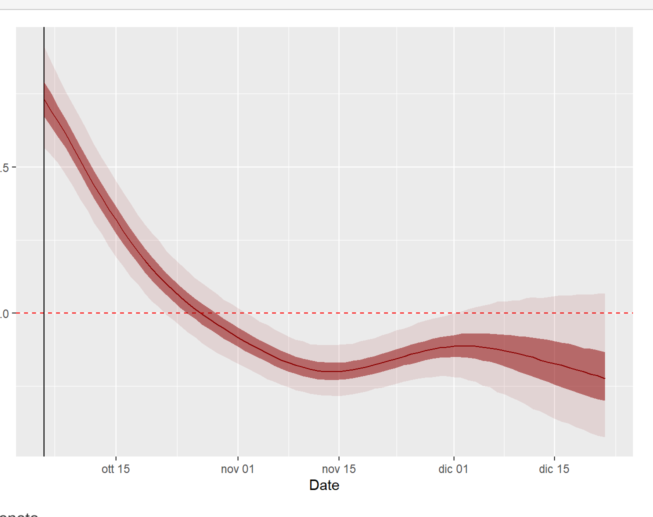



```
mean_inv_phi<-mean(rstan::extract(fit_hier)$inv_phi)
mean_y_rep<-colMeans(y_rep)
std_resid<-(as.vector(stan_data_hier$nonzero_positives)-mean_y_rep)/sqrt(mean_y_rep+mean_y_rep^2*mean_inv_phi)
qplot(mean_y_rep,std_resid)+hline_at(2)+hline_at(-2)
```



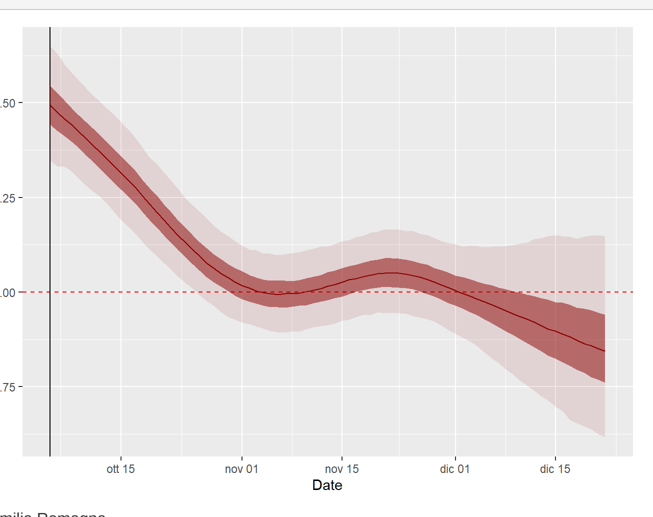
Rt Lombardia

```
plot_rt_hier(hier_data, fit_hier, regions, 'Lombardia')
```



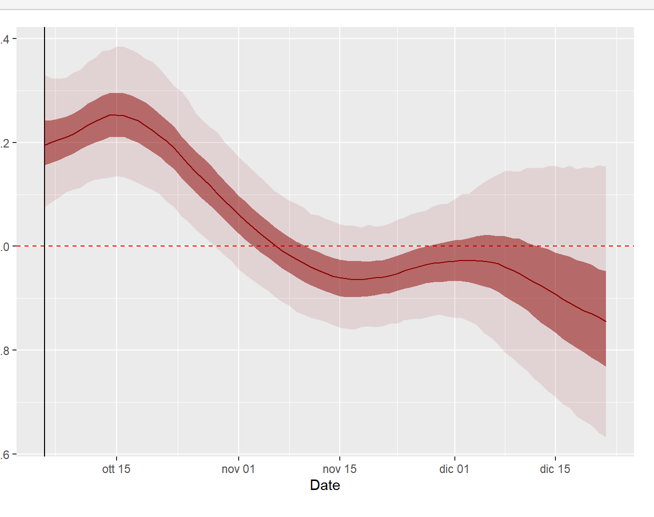
Rt Veneto

```
plot_rt_hier(hier_data, fit_hier, regions, 'Veneto')
```



Rt Emilia Romagna

```
plot_rt_hier(hier_data, fit_hier, regions, 'Emilia-Romagna')
```



Looic and WAIC

```
log_lik <- extract_log_lik(fit_hier)
loo <- loo(log_lik)
waic <- waic(log_lik)
```

```
loo$estimates[3,1]
```

```
## [1] 17800.55
```

```
waic$esestimates[3,1]
```

```
## NULL
```