

Report: COVID19 Bayesian Epidemiological Model

Måns Magnusson

2020-05-11

Contents

Summary / Take Away	1
Case study: The Sweden, Finland and Italy	2
Global covariate effects (model 1 and 2)	2
Country specific effects (model 3 and 4)	3
R0 for different countries	5
Infections, cases, deaths and Rt by country and model	7
Stan code	10
Appendix	13
Country specific effects (model 3 to 4)	13
Infections, cases, deaths and Rt by country and model	21

Summary / Take Away

New take-aways

- The hierarchical model with no pooling is much easier to fit with fewer divergences. It is also clear that that model better captures the Rt. Especially in the case of Finland and Sweden.

Previous take-aways

- Using the increased number of covariates seems to be better, model-wise compared to the baseline Imperial model. I think in many situations, our models now perform better than Imperial College Londons baseline.
- Mobility on the transit station seems to be a good covariate to use.
- The regression coefficients are sensitive. They need to be bounded on R^+ .

Current flaws to be fixed

- The infection to fatality ratio (IFR) for Finland now set to the IFR for Sweden.

Potential ways forward

- Introduce a dynamic component in the effect of Rt to smooth out the development over time.
- Use hospitalizations in addition to deaths.
- Focus on Finland and Sweden at the county level, although data may be missing to do this currently.
- Include serological studies to better estimate general prevalence over time.
- Include age-groups in the model.

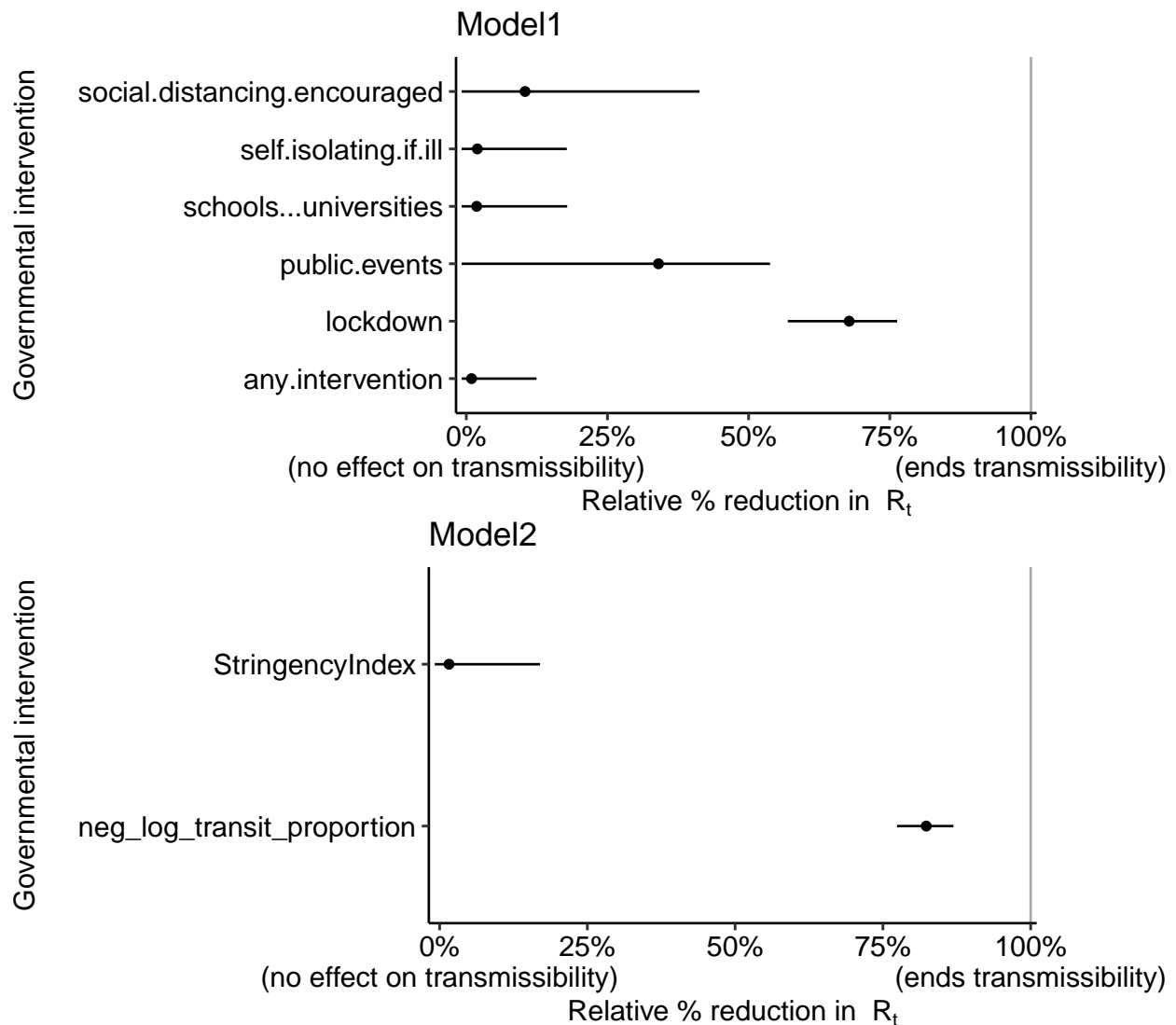
Model descriptions

- Model 1: The Imperial College model v3, with partial pooling of lockdown variable
- Model 2: Google Mobility and Stringency Index. Regression coefficients are assumed common for all countries (pooled alpha).
- Model 3: Google Mobility and Stringency Index. Regression coefficients are hierarchical and inform each other between countries. (hierarchical alpha)
- Model 4: Google Mobility and Stringency Index. Regression coefficients are estimated for each country. (local alpha)

Case study: The Sweden, Finland and Italy

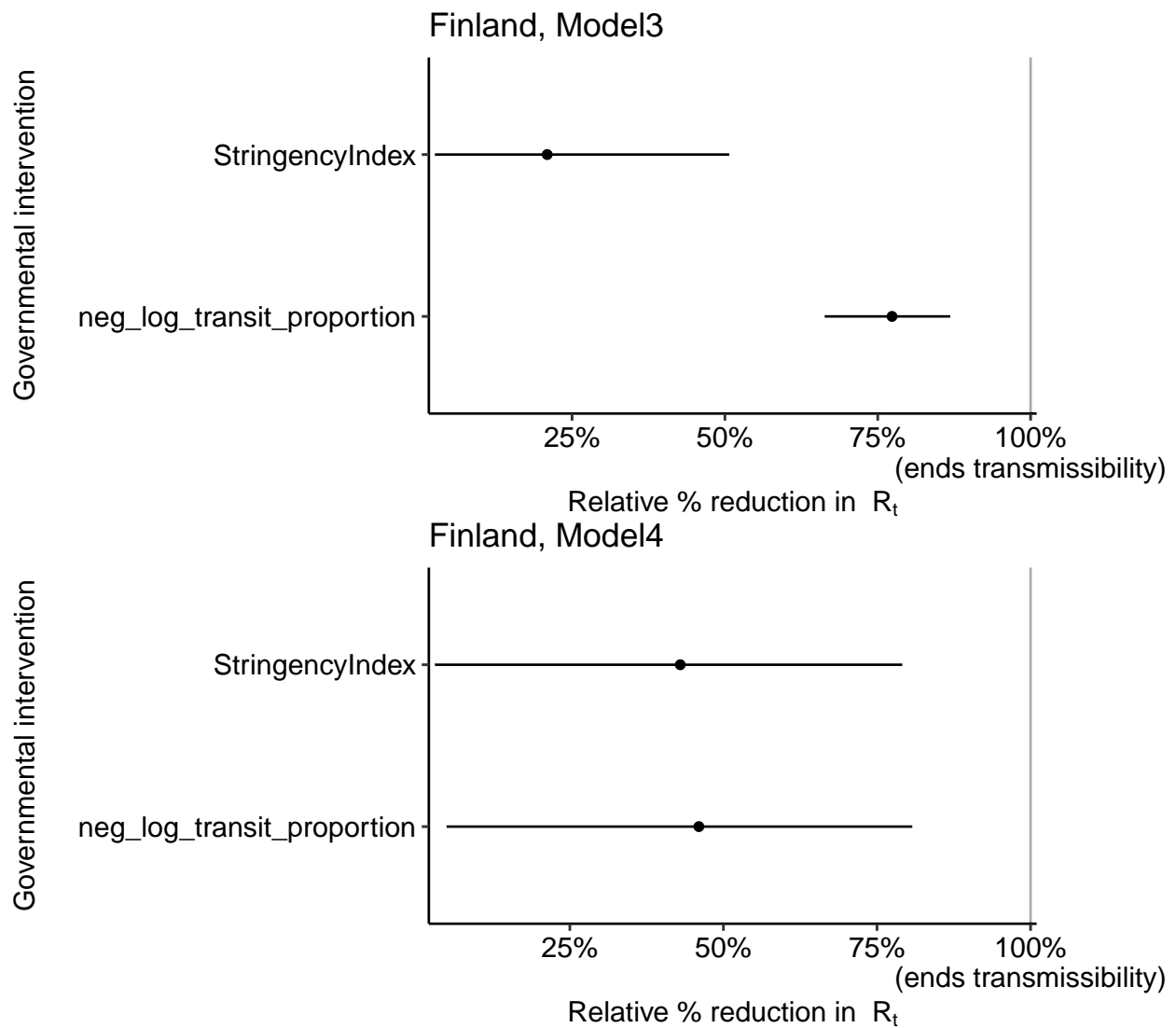
All other countries are included in the Appendix.

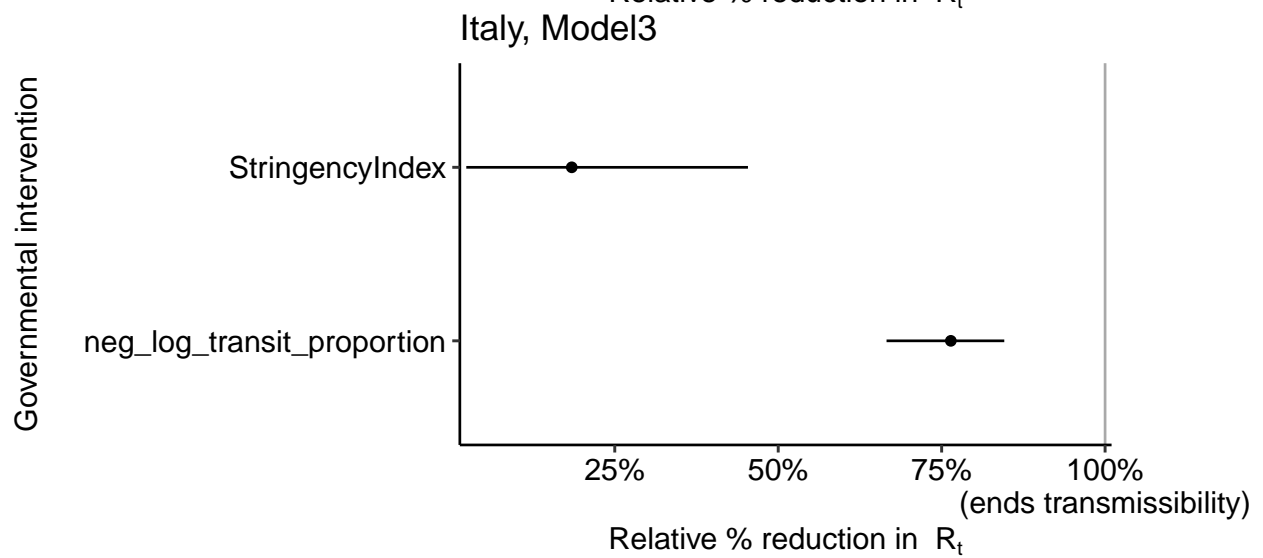
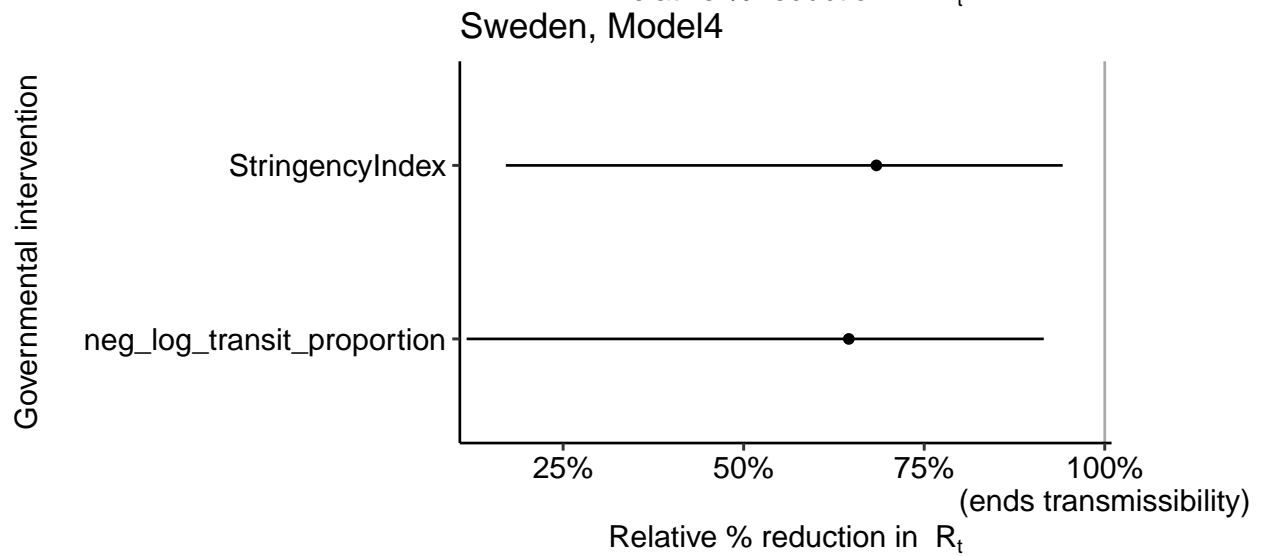
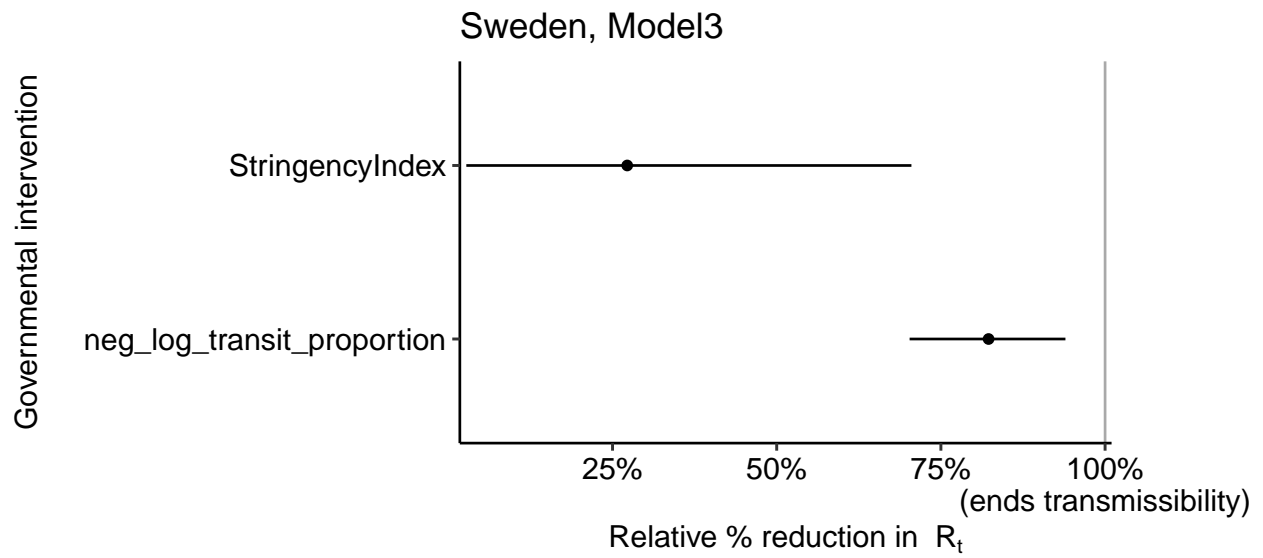
Global covariate effects (model 1 and 2)

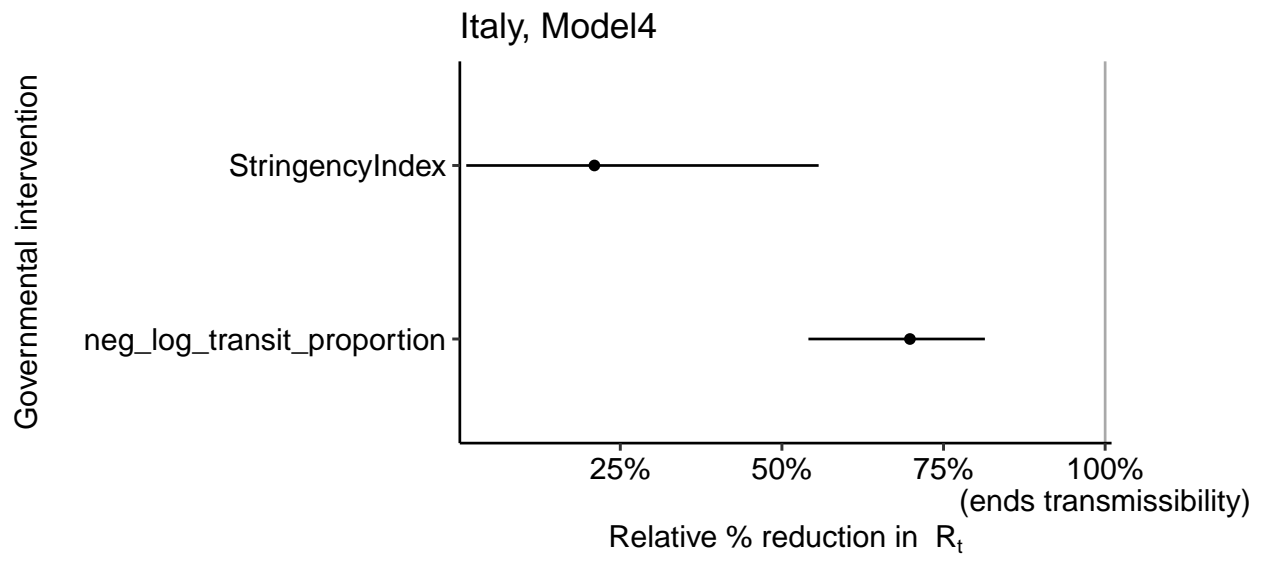


Country specific effects (model 3 and 4)

We see that the hierarchical model has an easier time to capture the general effect that mobility has a larger effect than the stringency index. Hence the actual mobility affects R_t more than the interventions.



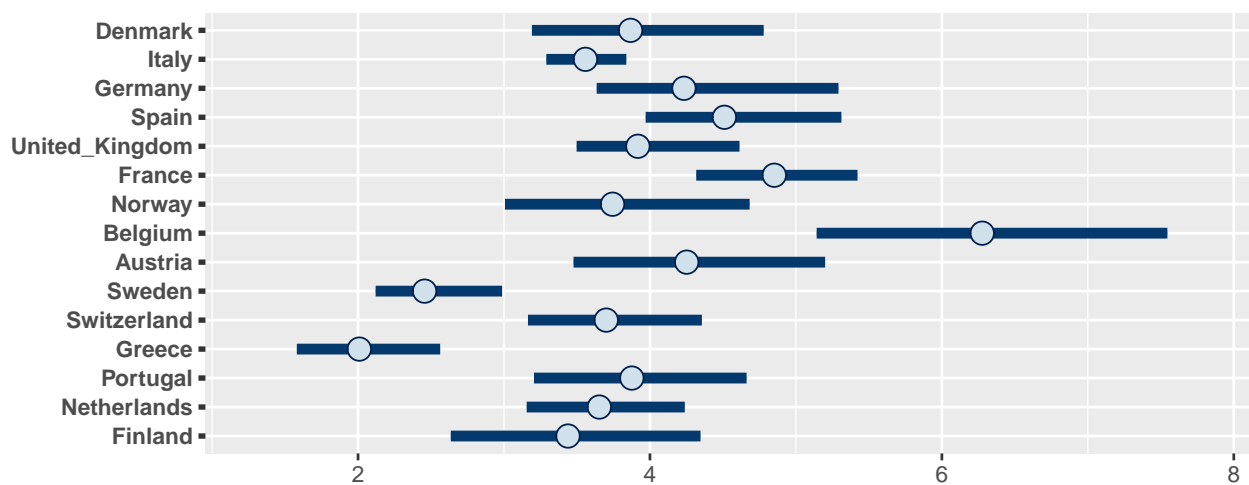




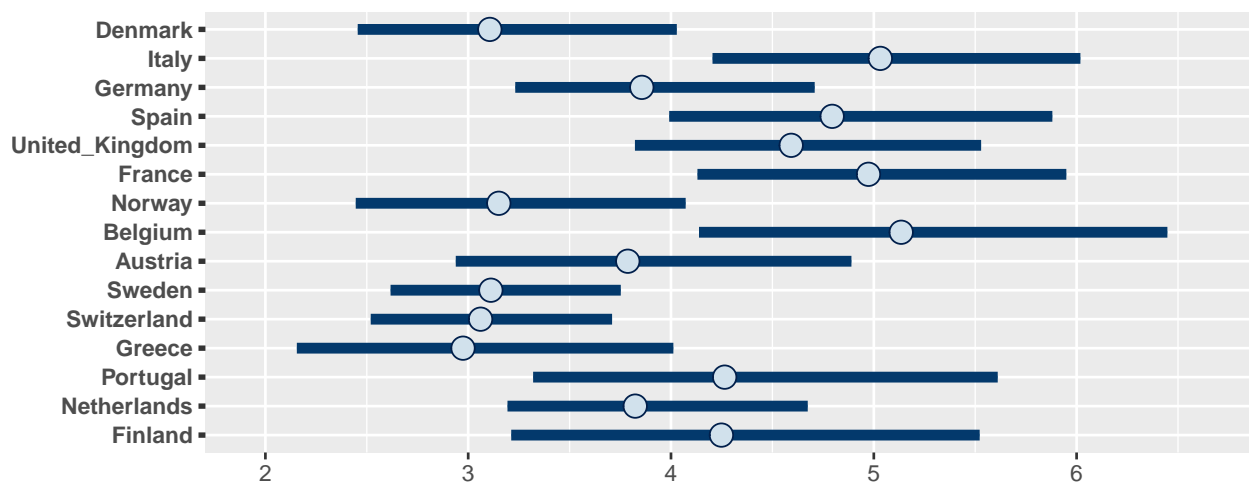
R0 for different countries

We can see that the different values for R_0 vary much more between the countries when we use models with common regression coefficients.

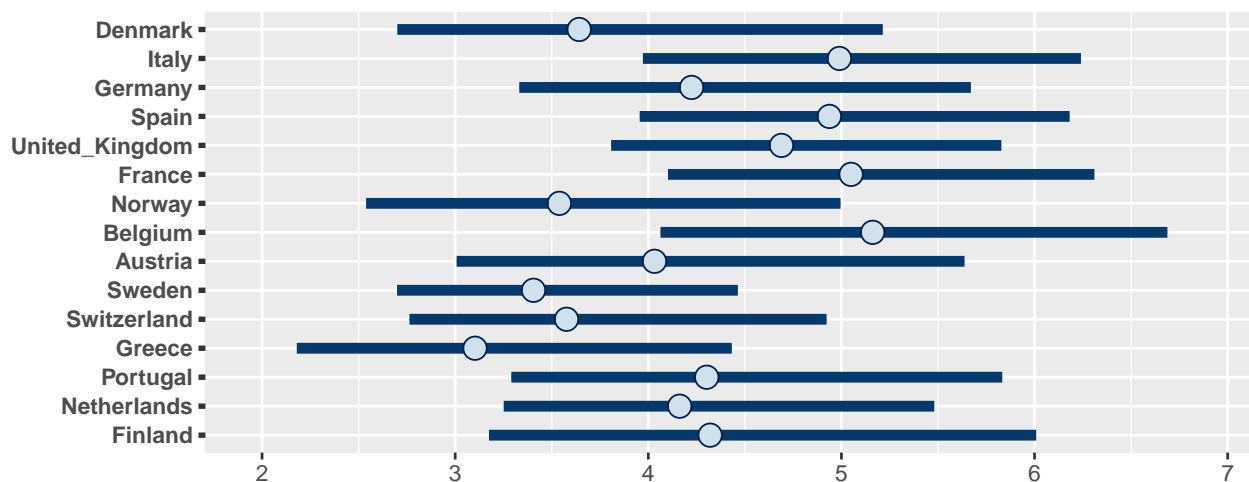
Model1



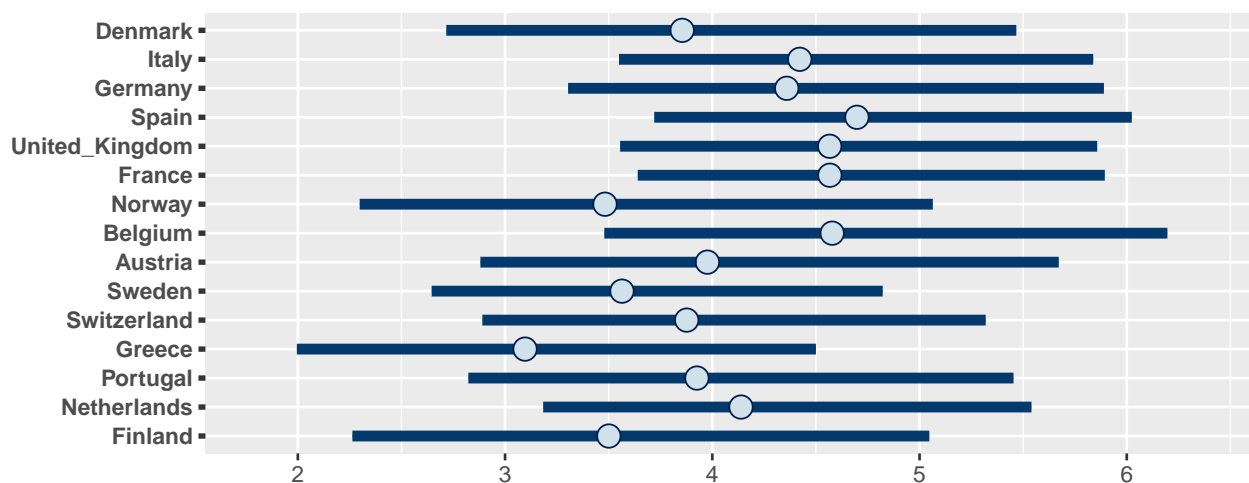
Model2



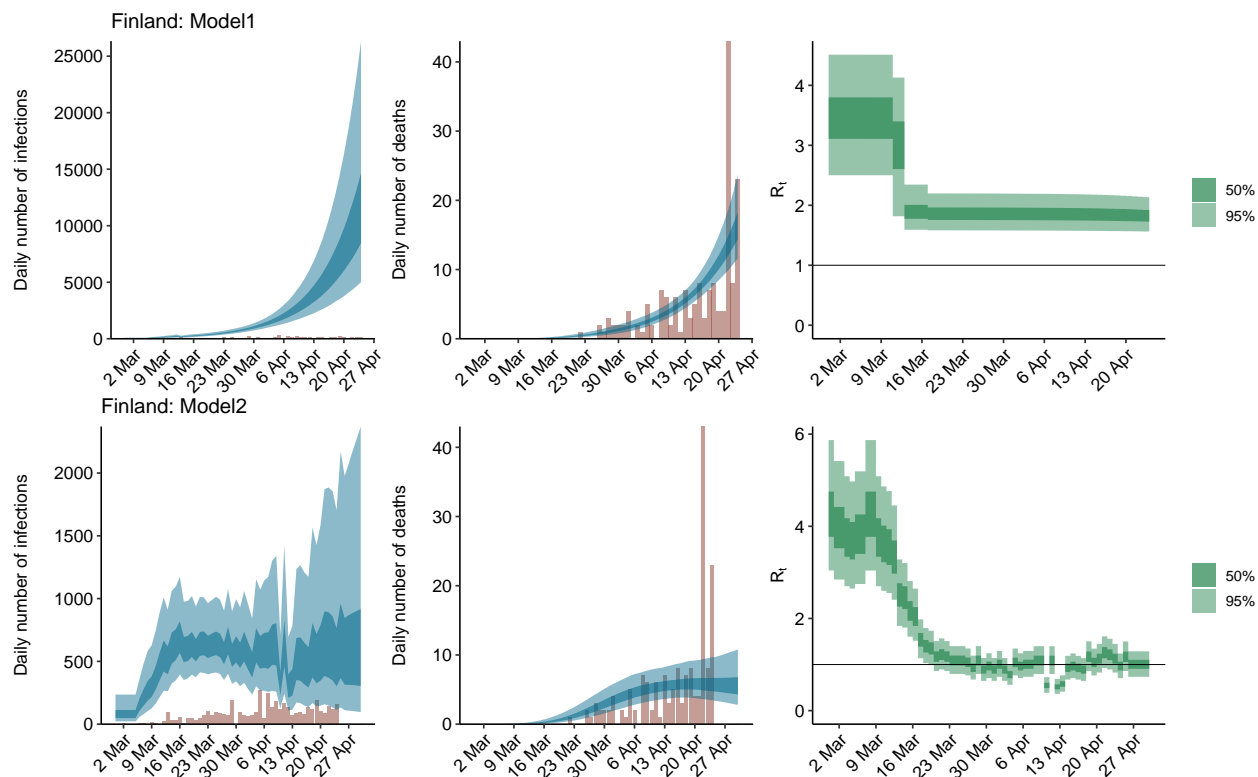
Model3

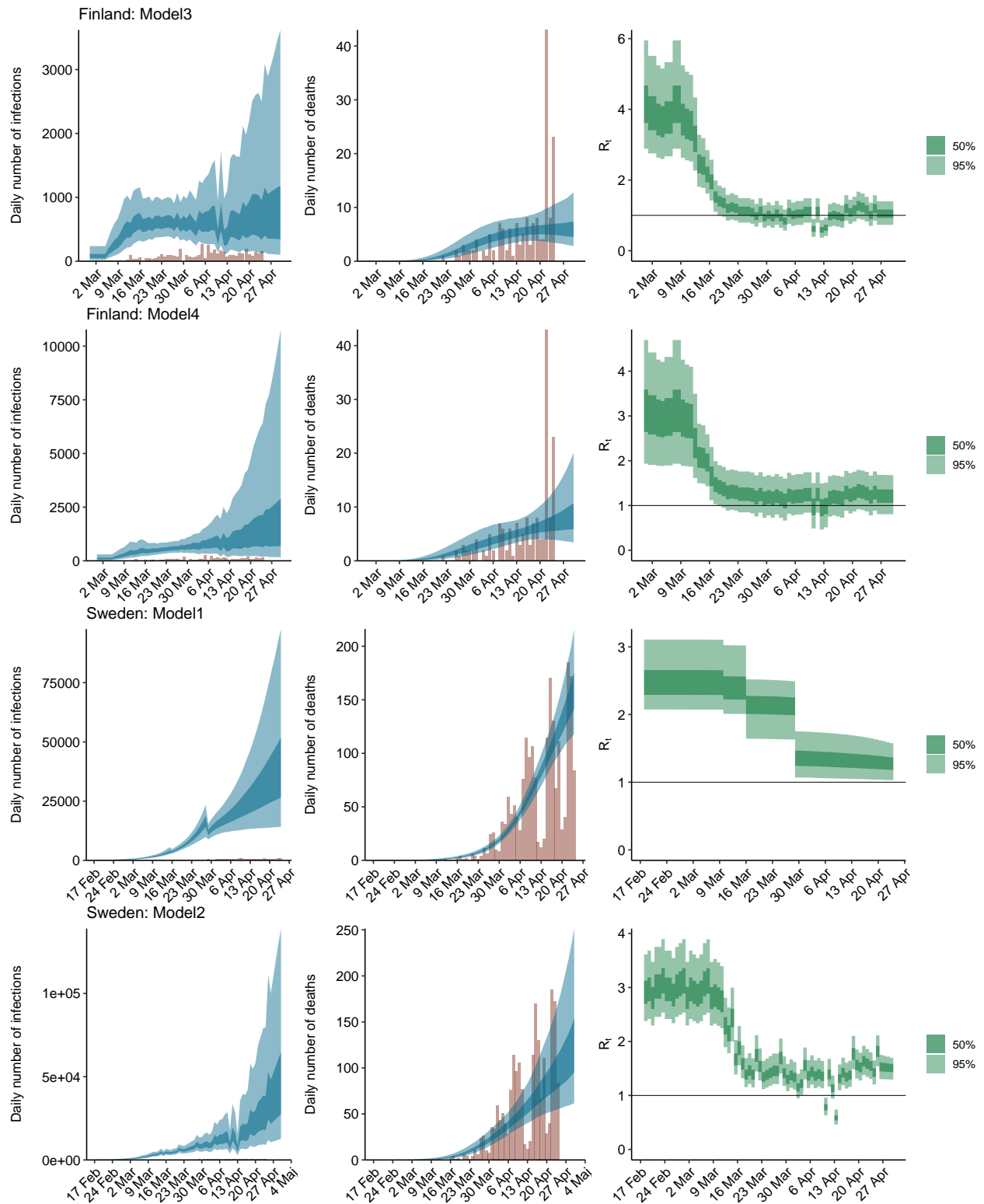


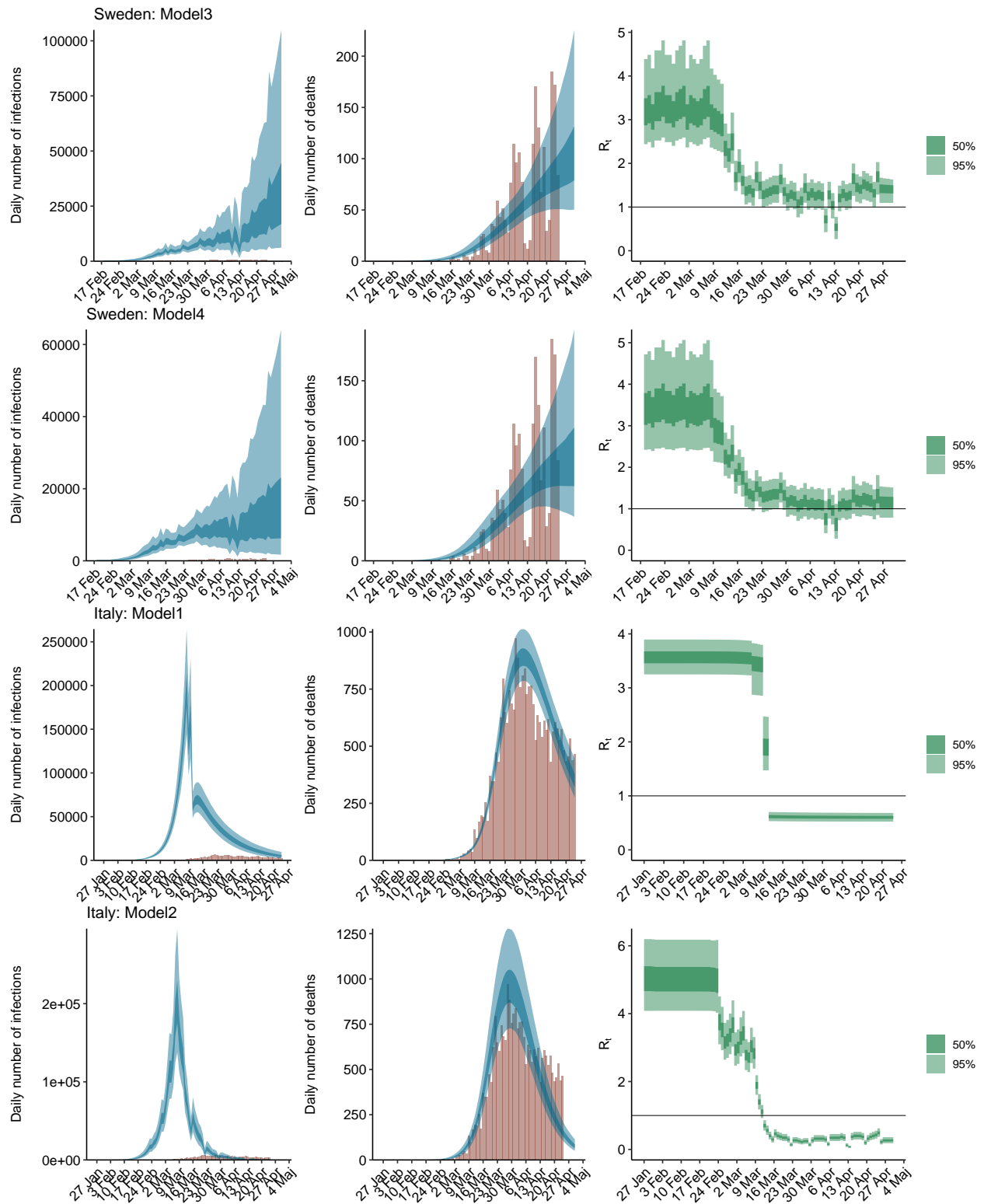
Model4

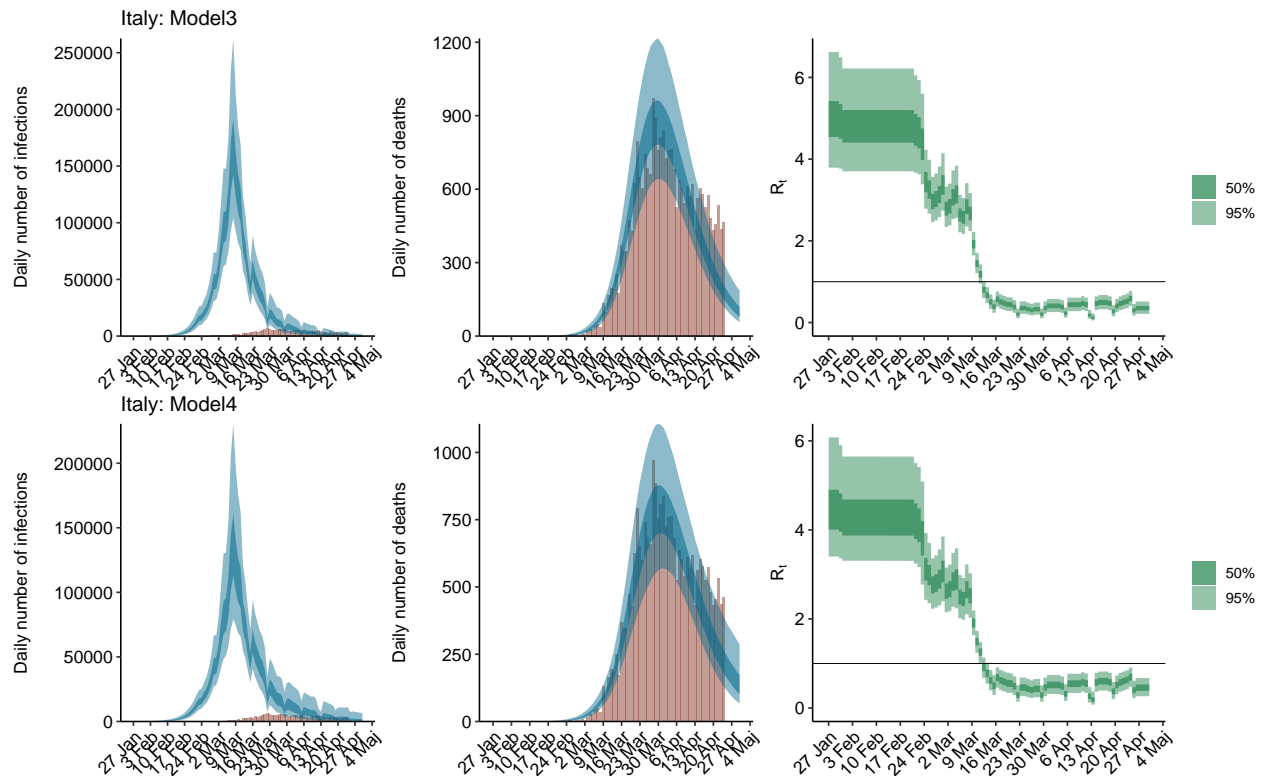


Infections, cases, deaths and Rt by country and model









Stan code

```
## S4 class stanmodel 'base' coded as follows:
## data {
##   int <lower=1> M; // number of countries
##   int <lower=1> P; // number of covariates
##   int <lower=1> N0; // number of days for which to impute infections
##   int<lower=1> N[M]; // days of observed data for country m. each entry must be <= N2
##   int<lower=1> N2; // days of observed data + # of days to forecast
##   int cases[N2,M]; // reported cases
##   int deaths[N2, M]; // reported deaths -- the rows with i > N contain -1 and should be ignored
##   matrix[N2, M] f; // h * s
##   matrix[N2, P] X[M]; // features matrix
##   int EpidemicStart[M];
##   real pop[M];
##   real SI[N2]; // fixed pre-calculated SI using emprical data from Neil
## }
##
## transformed data {
##   vector[N2] SI_rev; // SI in reverse order
##   vector[N2] f_rev[M]; // f in reversed order
##
##   for(i in 1:N2)
##     SI_rev[i] = SI[N2-i+1];
##
##   for(m in 1:M){
##     for(i in 1:N2) {
##       f_rev[m, i] = f[N2-i+1,m];
##     }
##   }
## }
```

```

##   }
## }
## }
##
## parameters {
##   real<lower=0> mu[M]; // intercept for Rt
##   real<lower=0> alpha_hier[P]; // sudo parameter for the hier term for alpha
##   real<lower=0> gamma;
##   vector[M] lockdown;
##   real<lower=0> kappa;
##   real<lower=0> y[M];
##   real<lower=0> phi;
##   real<lower=0> tau;
##   real <lower=0> ifr_noise[M];
## }
##
## transformed parameters {
##   vector[P] alpha;
##   matrix[N2, M] prediction = rep_matrix(0,N2,M);
##   matrix[N2, M] E_deaths = rep_matrix(0,N2,M);
##   matrix[N2, M] Rt = rep_matrix(0,N2,M);
##   matrix[N2, M] Rt_adj = Rt;
##
##   {
##     matrix[N2,M] cumm_sum = rep_matrix(0,N2,M);
##     for(i in 1:P){
##       alpha[i] = alpha_hier[i] - ( log(1.05) / 6.0 );
##     }
##     for (m in 1:M){
##       prediction[1:NO,m] = rep_vector(y[m],NO); // learn the number of cases in the first NO days
##       cumm_sum[2:NO,m] = cumulative_sum(prediction[2:NO,m]);
##
##       Rt[,m] = mu[m] * exp(-X[m] * alpha - X[m][,5] * lockdown[m]);
##       Rt_adj[1:NO,m] = Rt[1:NO,m];
##       for (i in (NO+1):N2) {
##         real convolution = dot_product(sub_col(prediction, 1, m, i-1), tail(SI_rev, i-1));
##         cumm_sum[i,m] = cumm_sum[i-1,m] + prediction[i-1,m];
##         Rt_adj[i,m] = ((pop[m]-cumm_sum[i,m]) / pop[m]) * Rt[i,m];
##         prediction[i, m] = Rt_adj[i,m] * convolution;
##       }
##       E_deaths[1, m]= 1e-15 * prediction[1,m];
##       for (i in 2:N2){
##         E_deaths[i,m] = ifr_noise[m] * dot_product(sub_col(prediction, 1, m, i-1), tail(f_rev[m],
##       }
##     }
##   }
## }
## model {
##   tau ~ exponential(0.03);
##   for (m in 1:M){
##     y[m] ~ exponential(1/tau);
##   }
##   gamma ~ normal(0,.2);

```

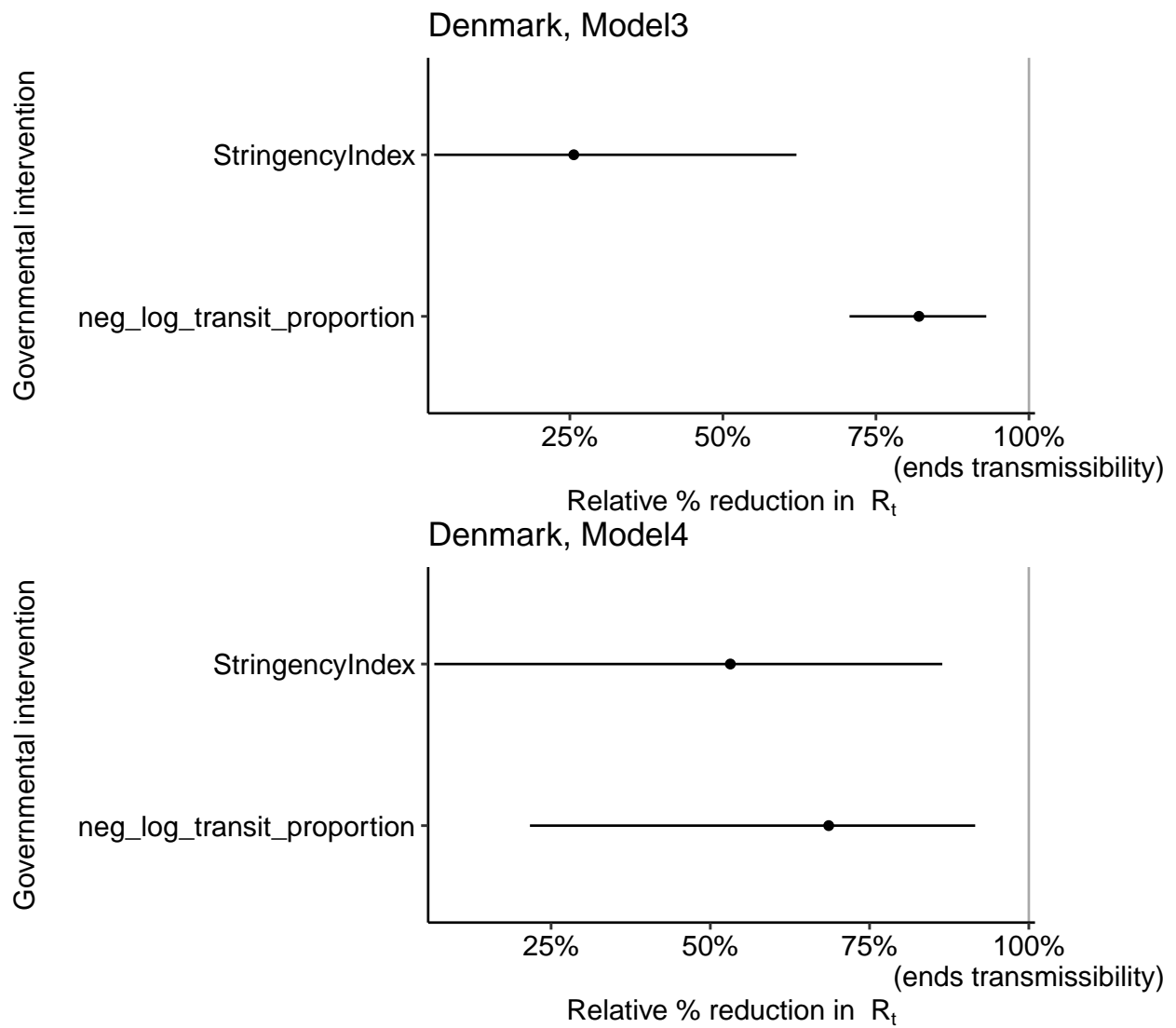
```

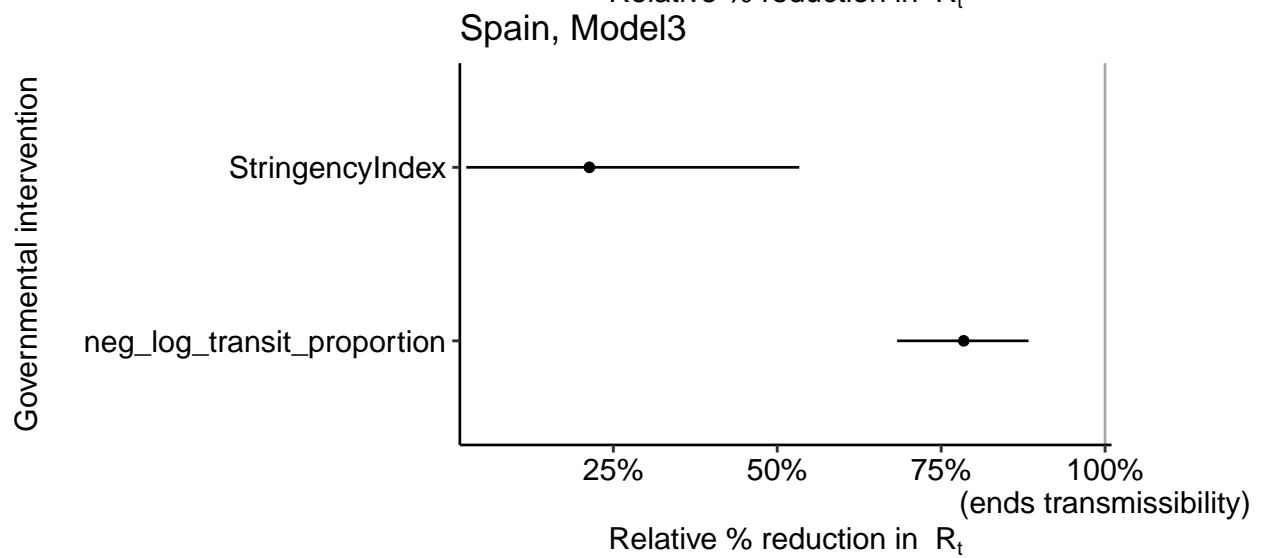
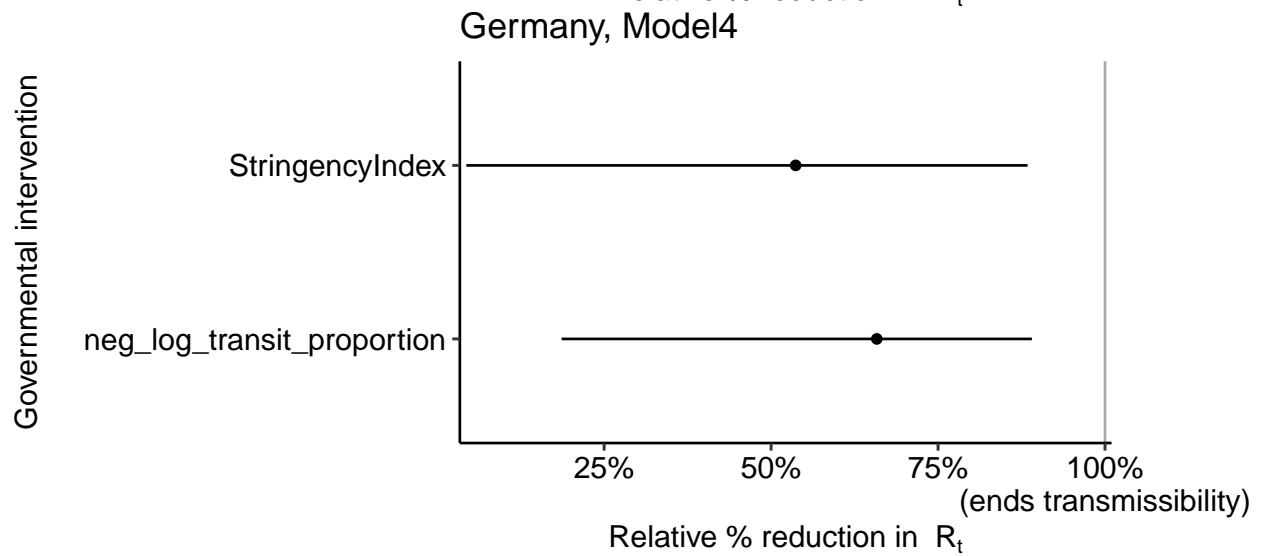
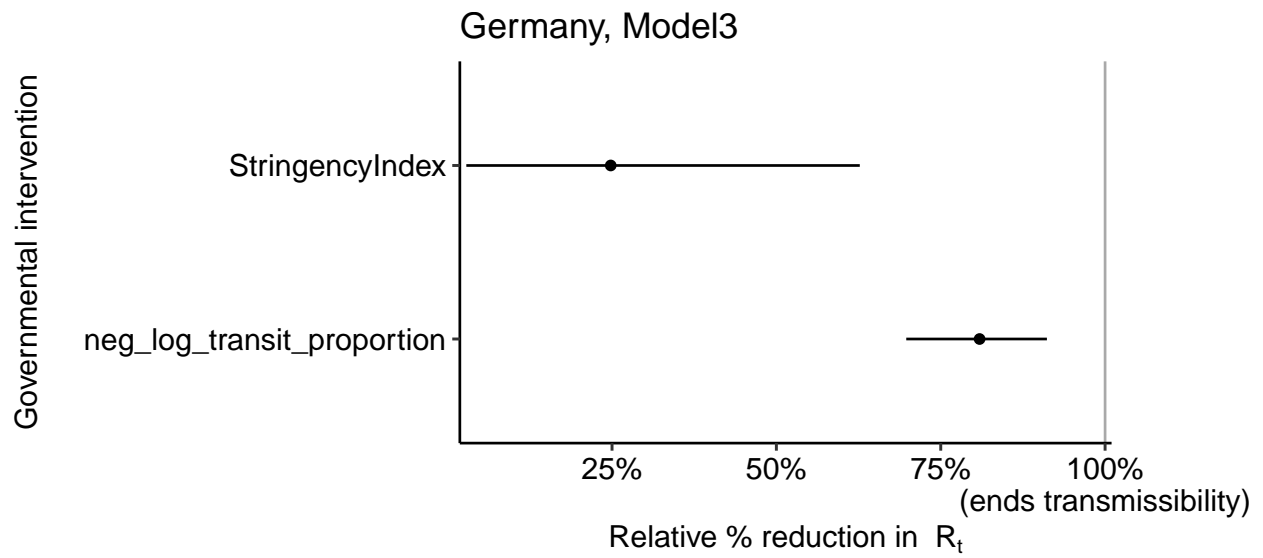
## lockdown ~ normal(0,gamma);
## phi ~ normal(0,5);
## kappa ~ normal(0,0.5);
## mu ~ normal(3.28, kappa); // citation: https://academic.oup.com/jtm/article/27/2/taaa021/5735319
## alpha_hier ~ gamma(.1667,1);
## ifr_noise ~ normal(1,0.1);
## for(m in 1:M){
##   deaths[EpidemicStart[m]:N[m], m] ~ neg_binomial_2(E_deaths[EpidemicStart[m]:N[m], m], phi);
## }
## }
##
## generated quantities {
##   matrix[N2, M] prediction0 = rep_matrix(0,N2,M);
##   matrix[N2, M] E_deaths0 = rep_matrix(0,N2,M);
##
##   {
##     matrix[N2,M] cumm_sum0 = rep_matrix(0,N2,M);
##     for (m in 1:M){
##       for (i in 2:NO){
##         cumm_sum0[i,m] = cumm_sum0[i-1,m] + y[m];
##       }
##       prediction0[1:NO,m] = rep_vector(y[m],NO);
##       for (i in (NO+1):N2) {
##         real convolution0 = dot_product(sub_col(prediction0, 1, m, i-1), tail(SI_rev, i-1));
##         cumm_sum0[i,m] = cumm_sum0[i-1,m] + prediction0[i-1,m];
##         prediction0[i, m] = ((pop[m]-cumm_sum0[i,m]) / pop[m]) * mu[m] * convolution0;
##       }
##       E_deaths0[1, m]= 1e-15 * prediction0[1,m];
##       for (i in 2:N2){
##         E_deaths0[i,m] = ifr_noise[m] * dot_product(sub_col(prediction0, 1, m, i-1), tail(f_rev[m], i-1));
##       }
##     }
##   }
## }
##

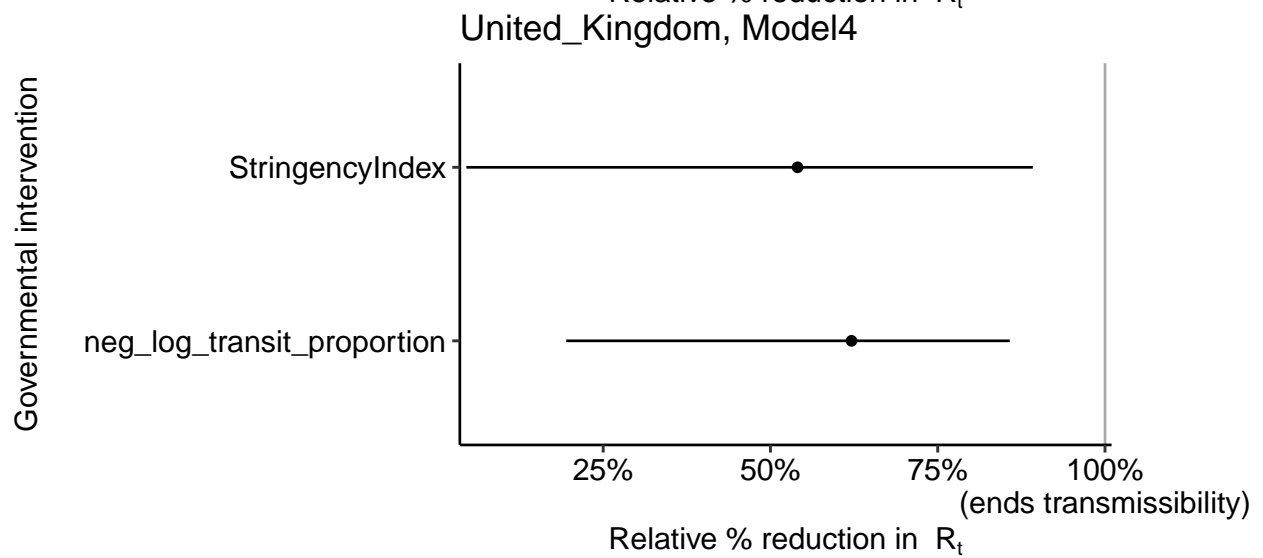
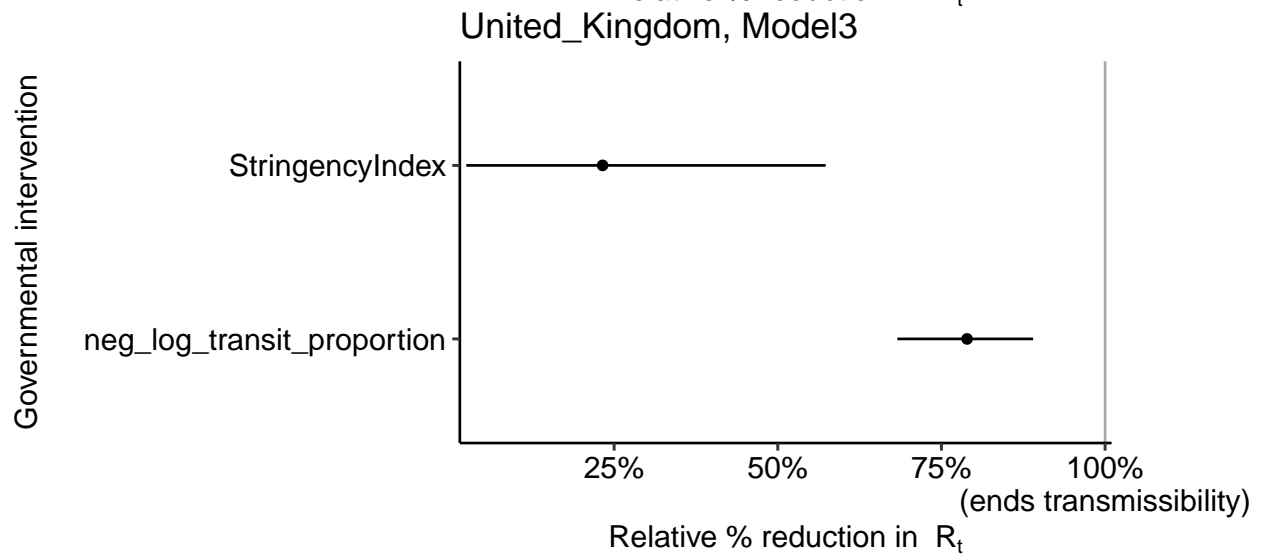
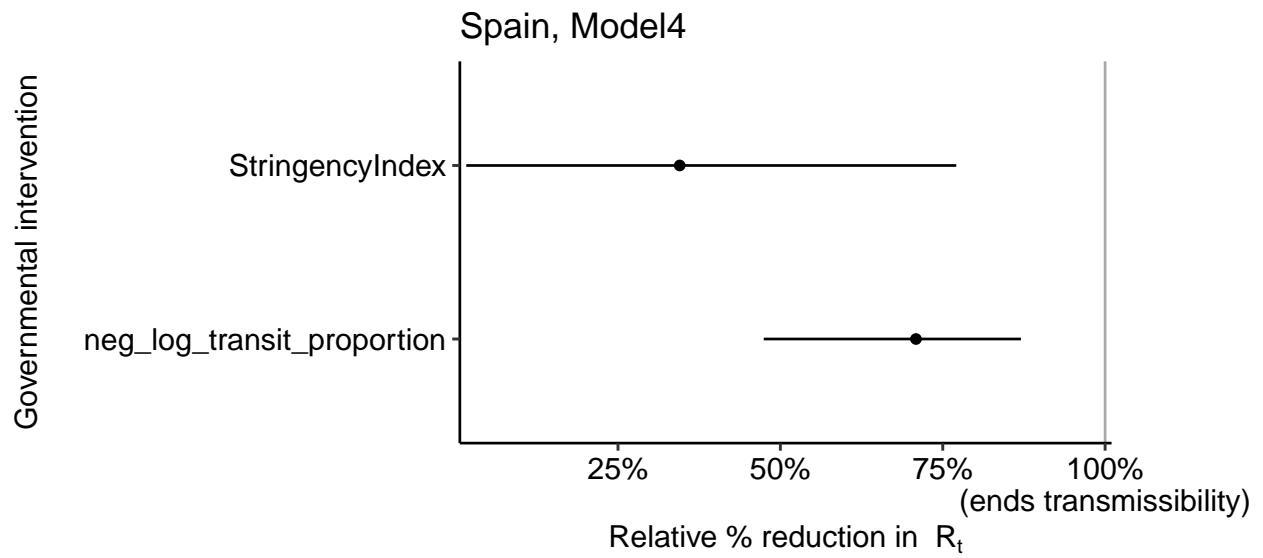
```

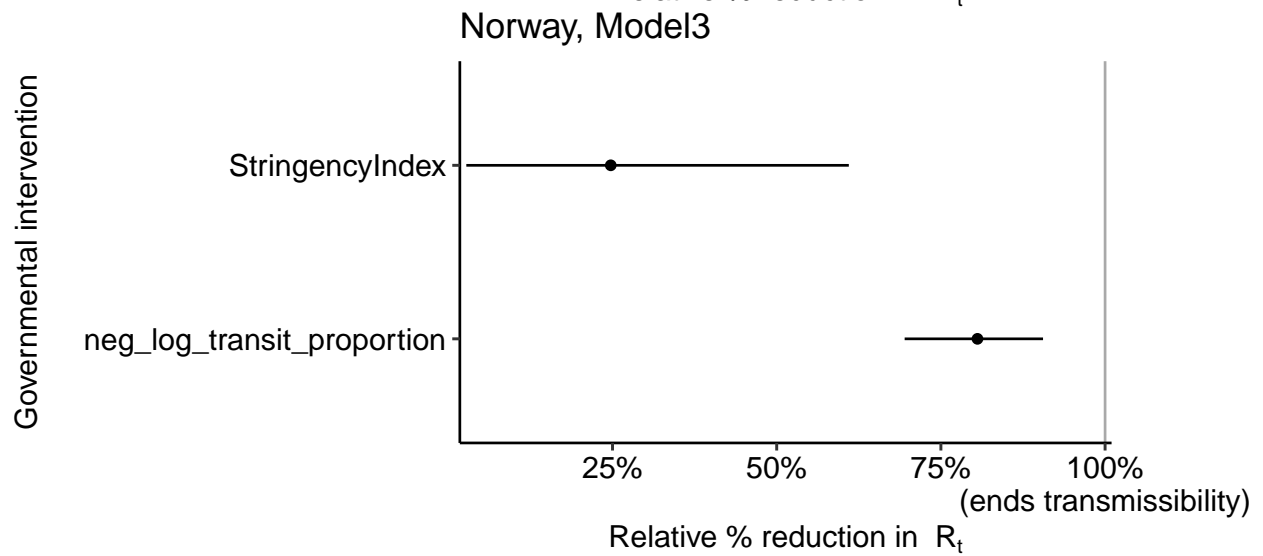
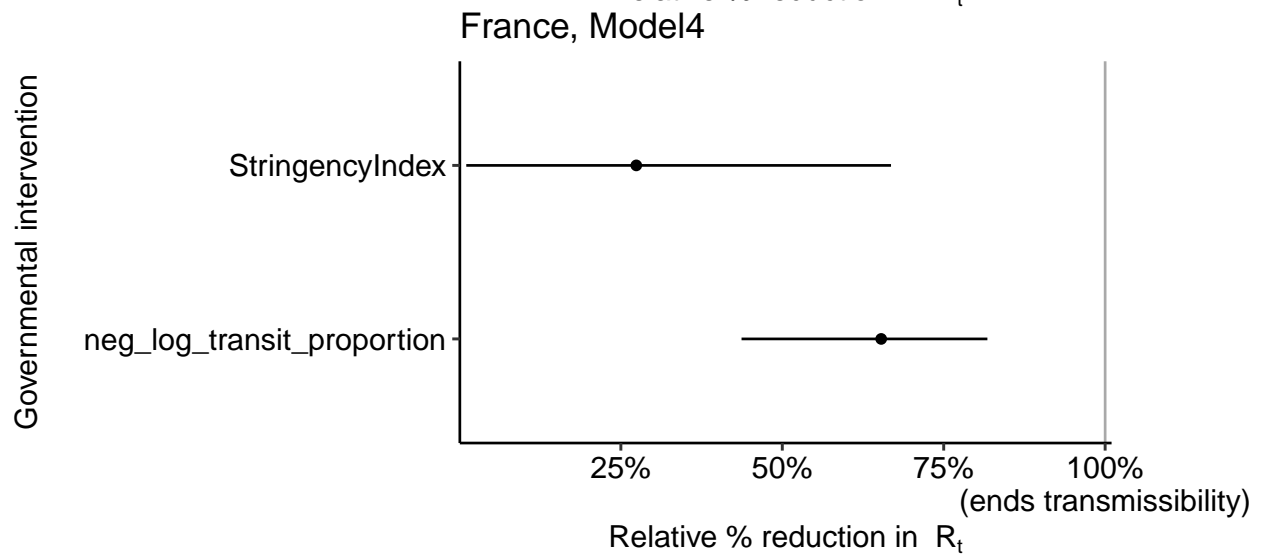
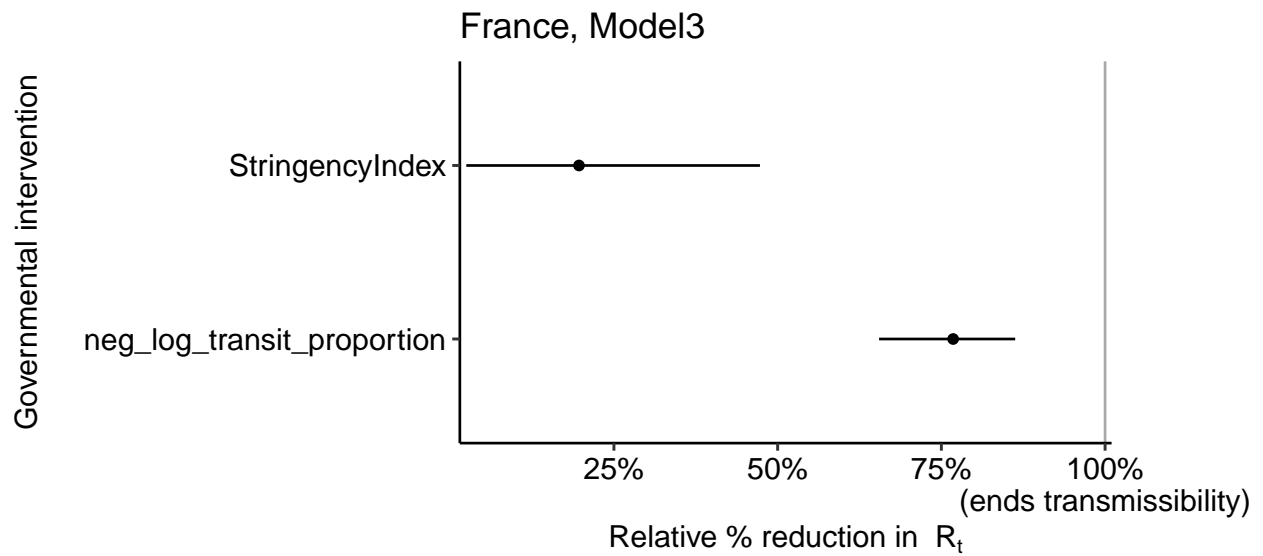
Appendix

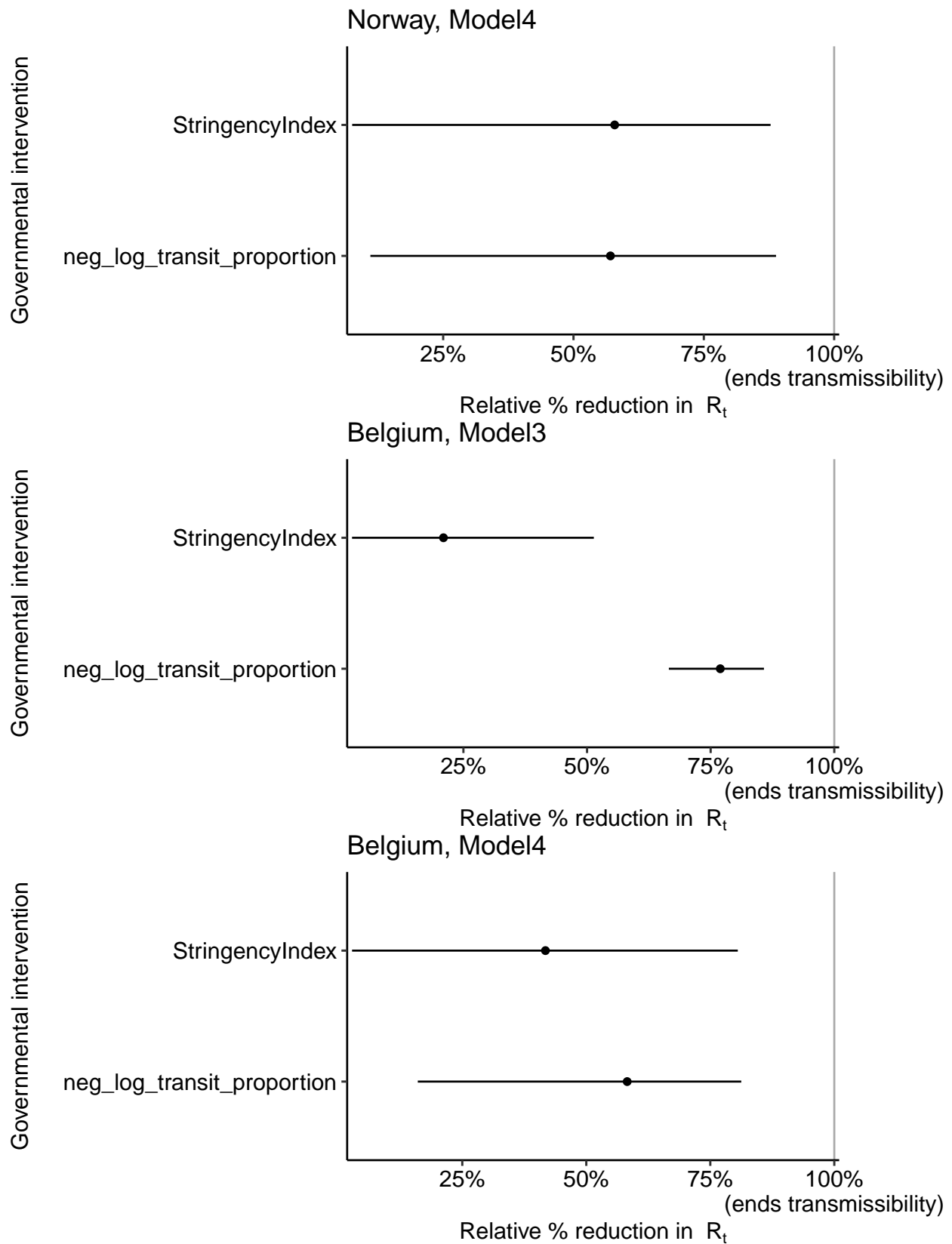
Country specific effects (model 3 to 4)

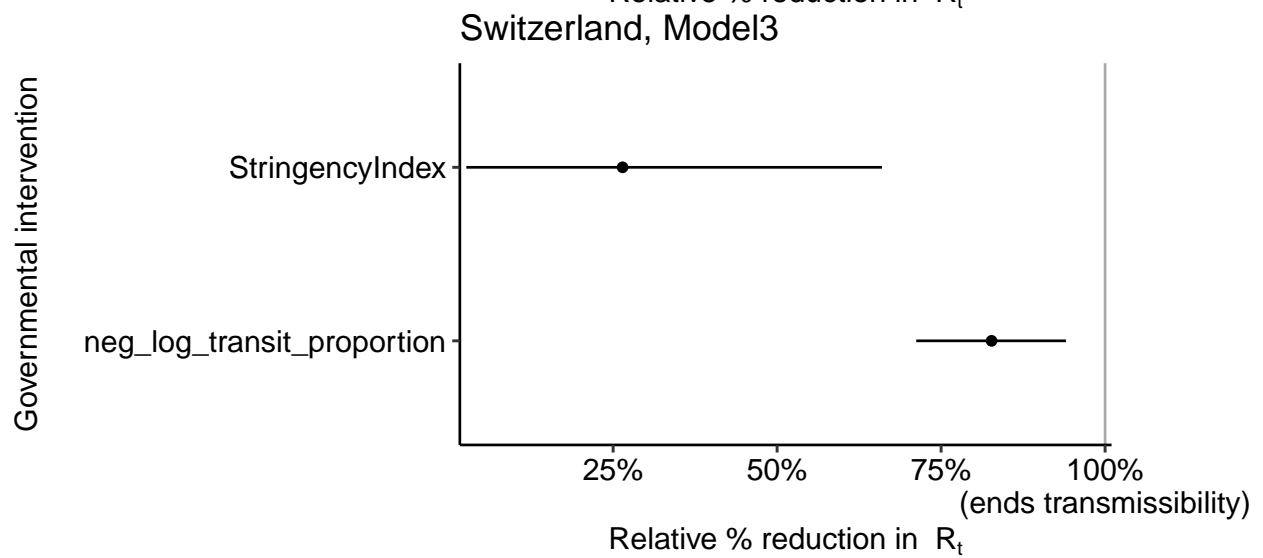
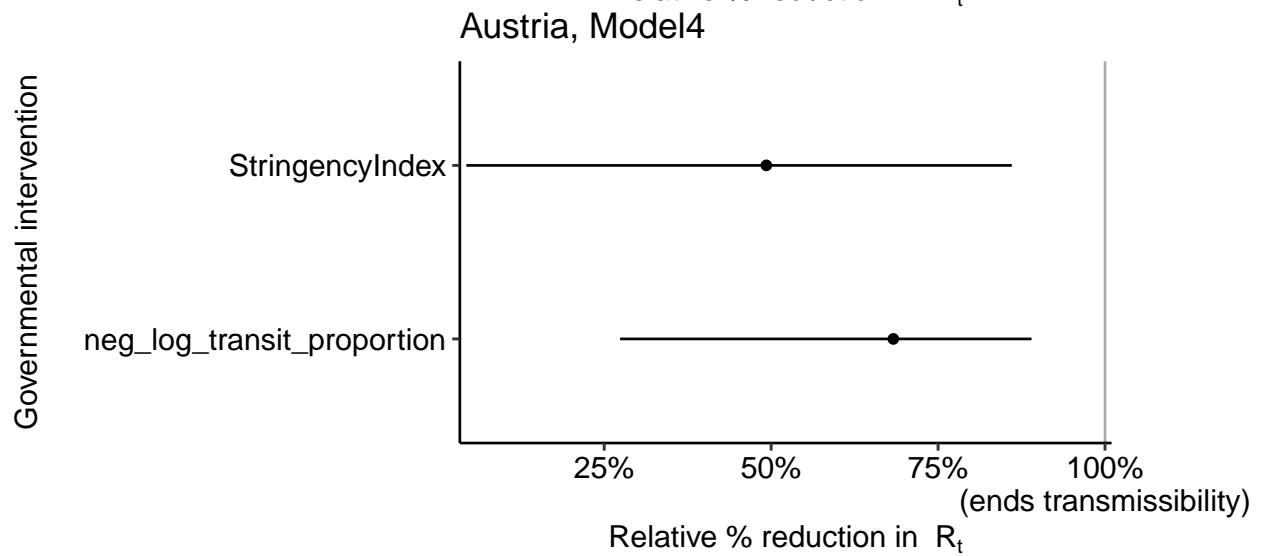
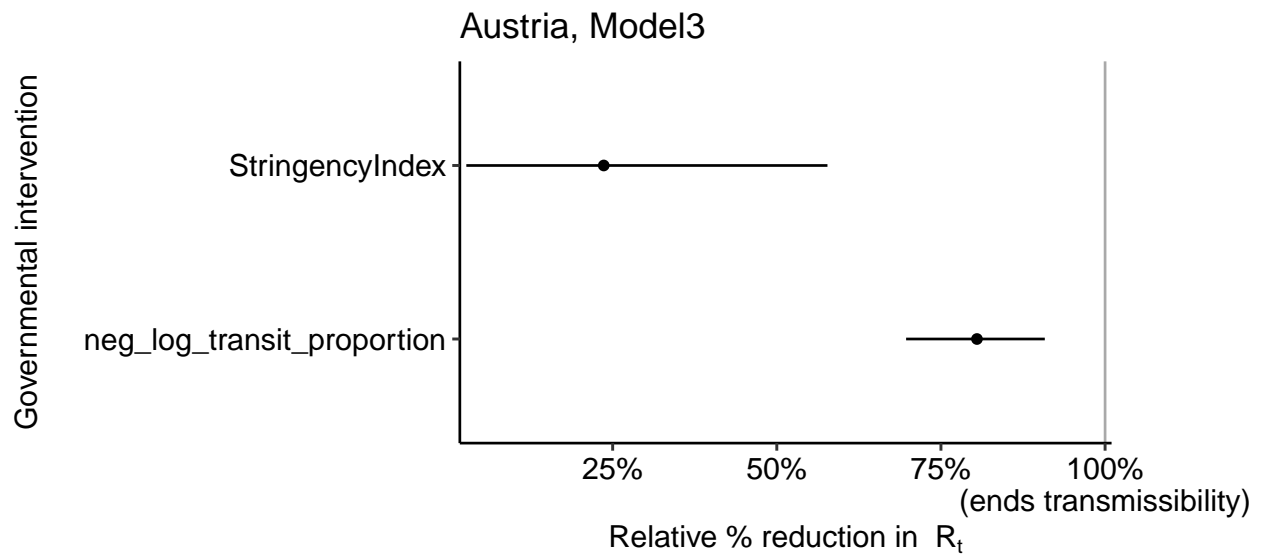


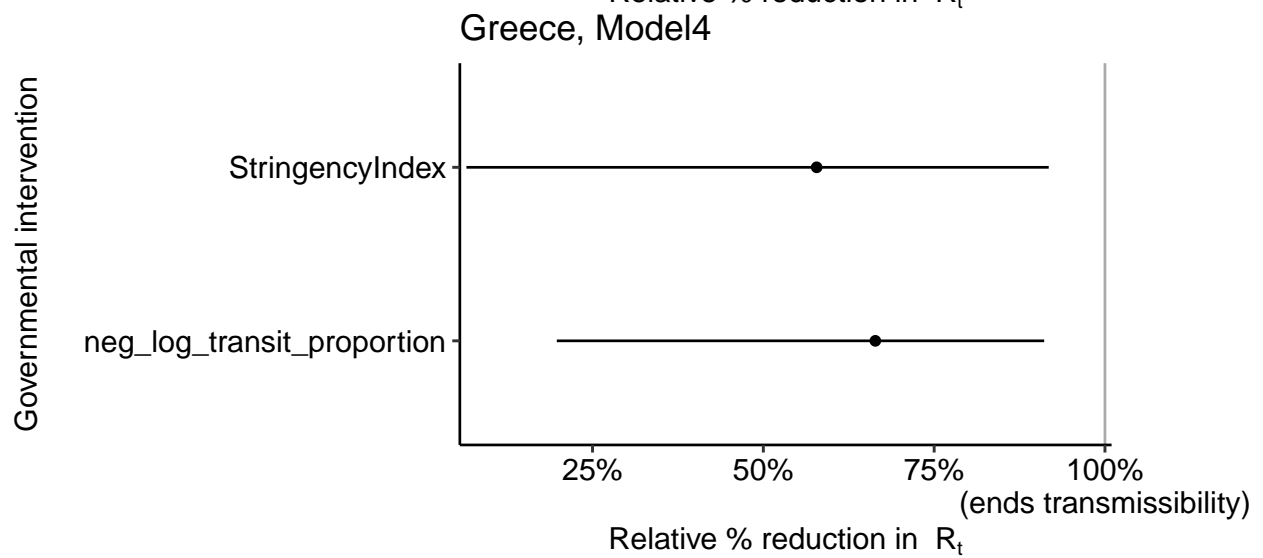
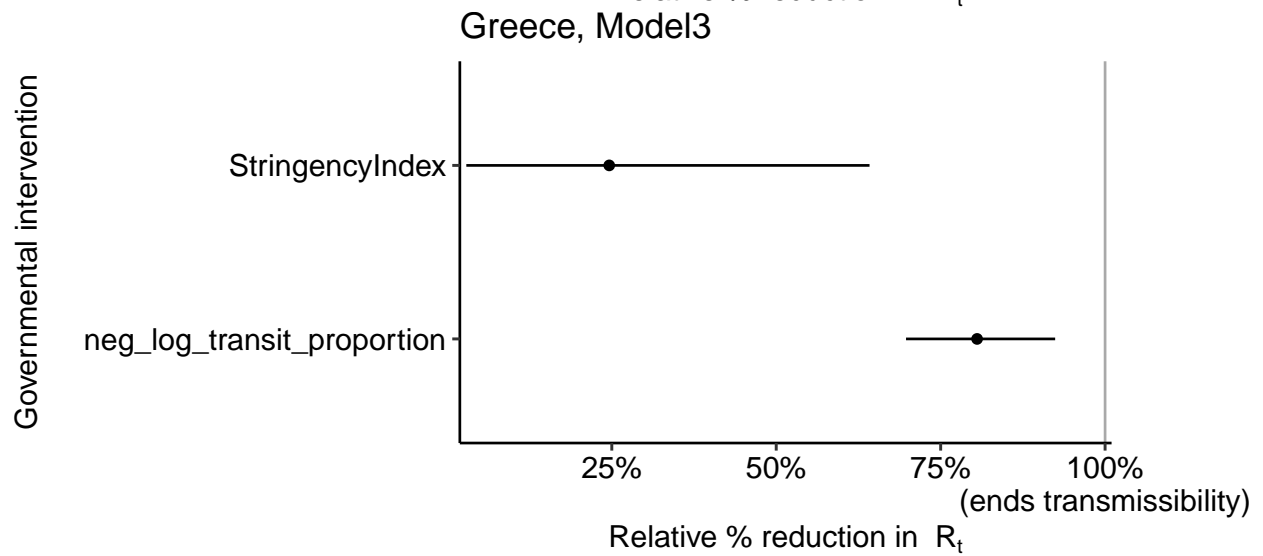
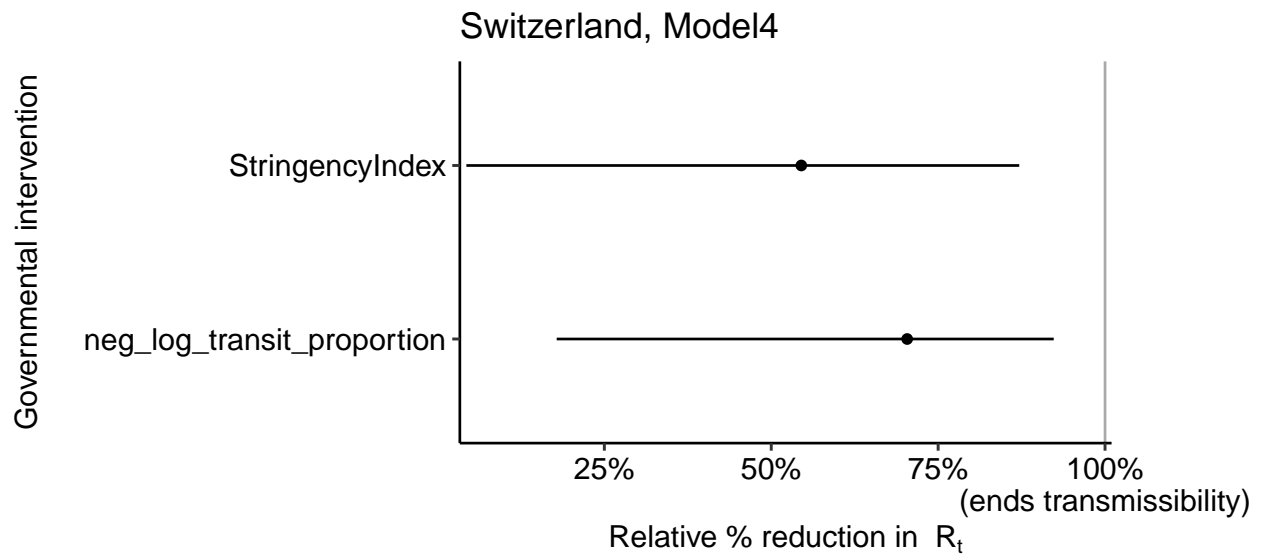


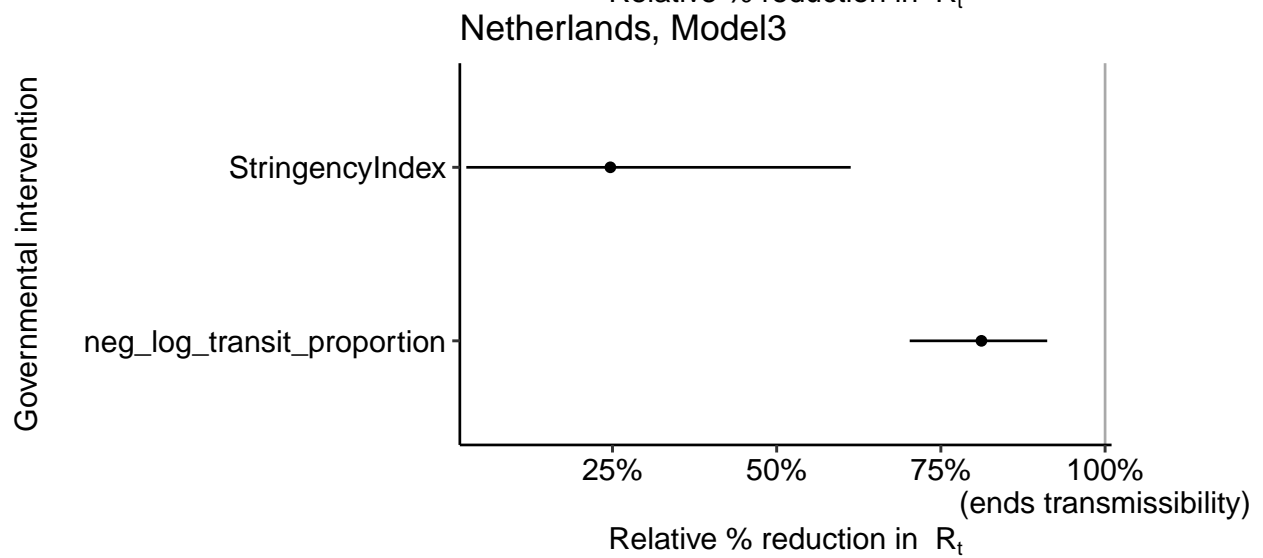
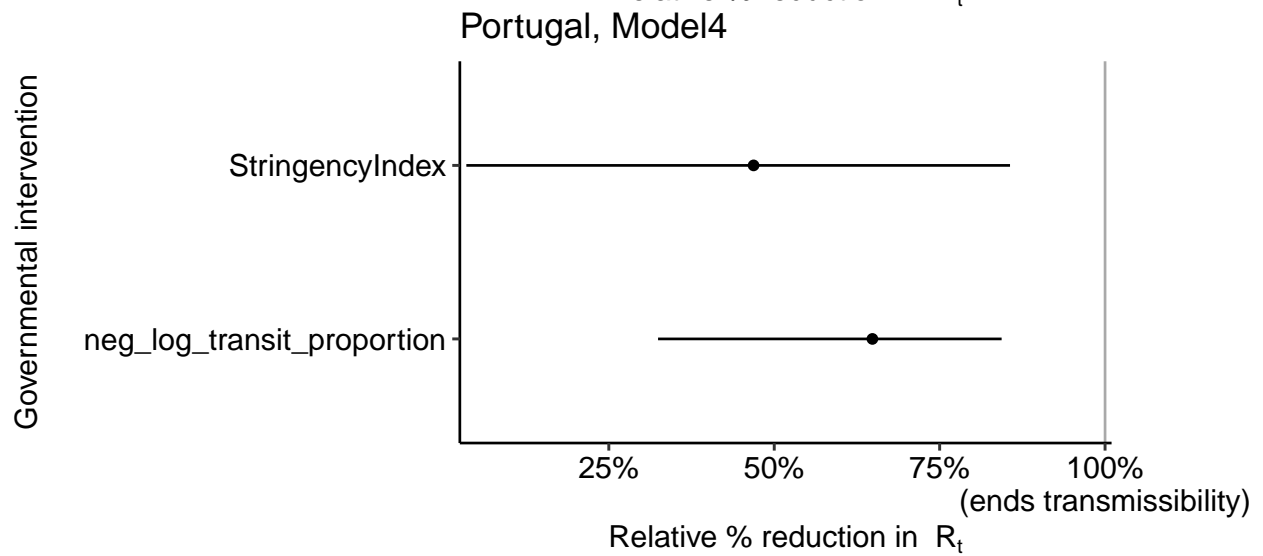
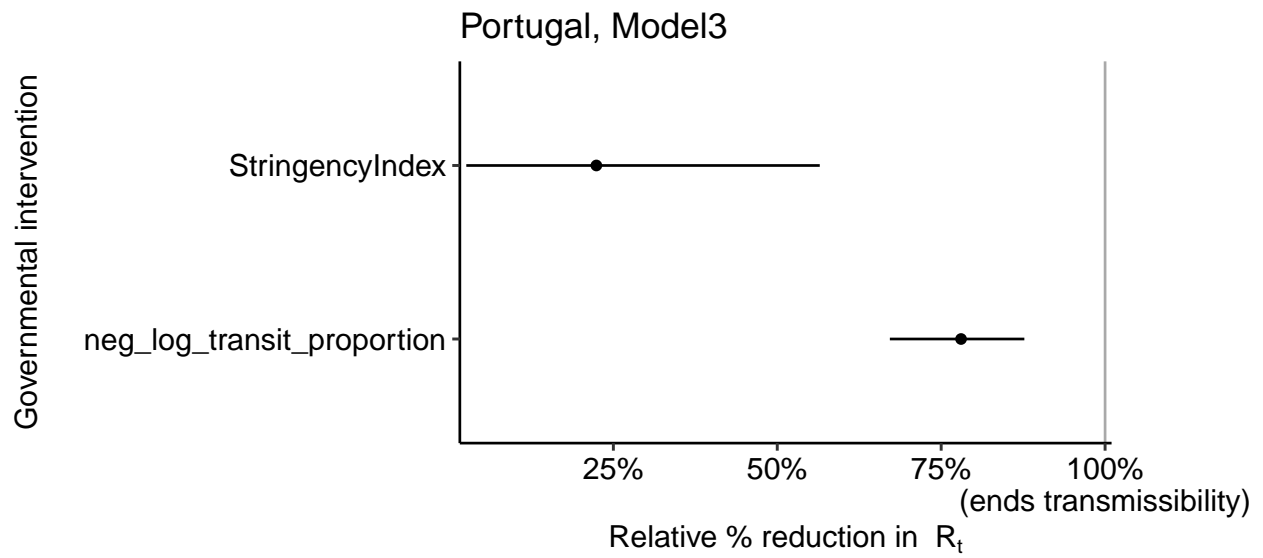


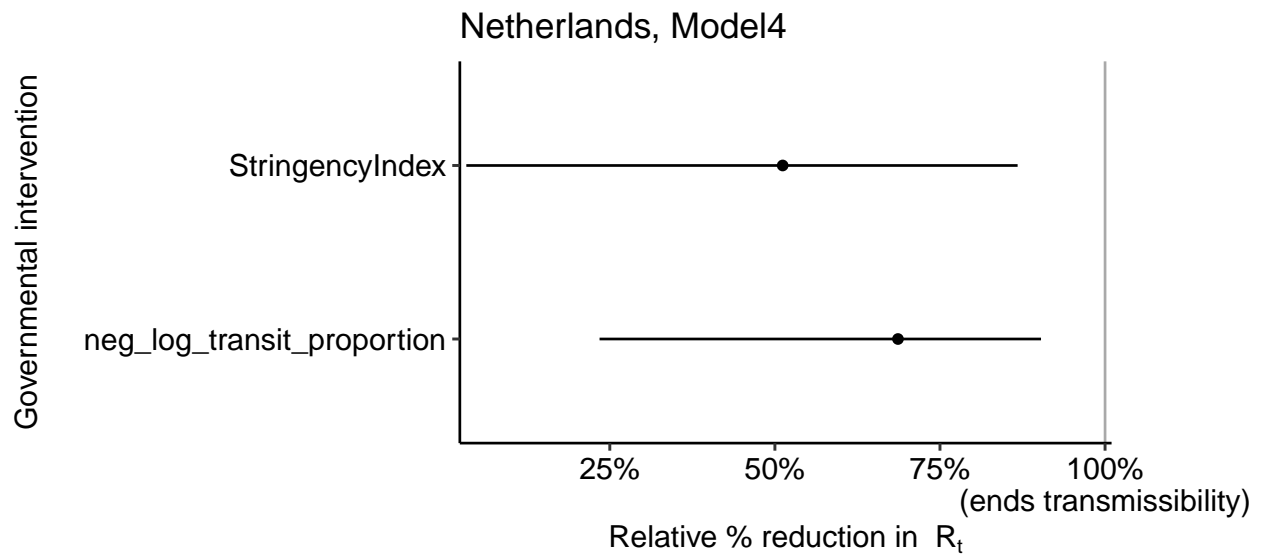




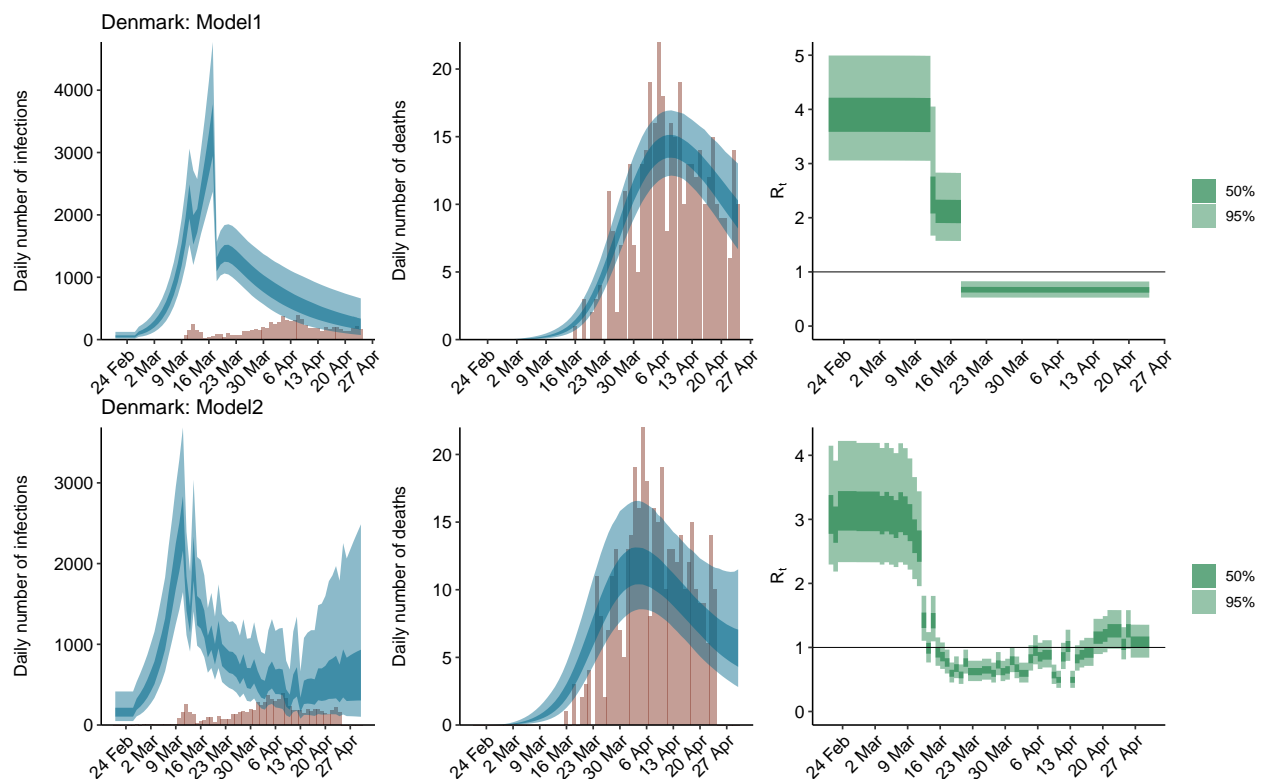


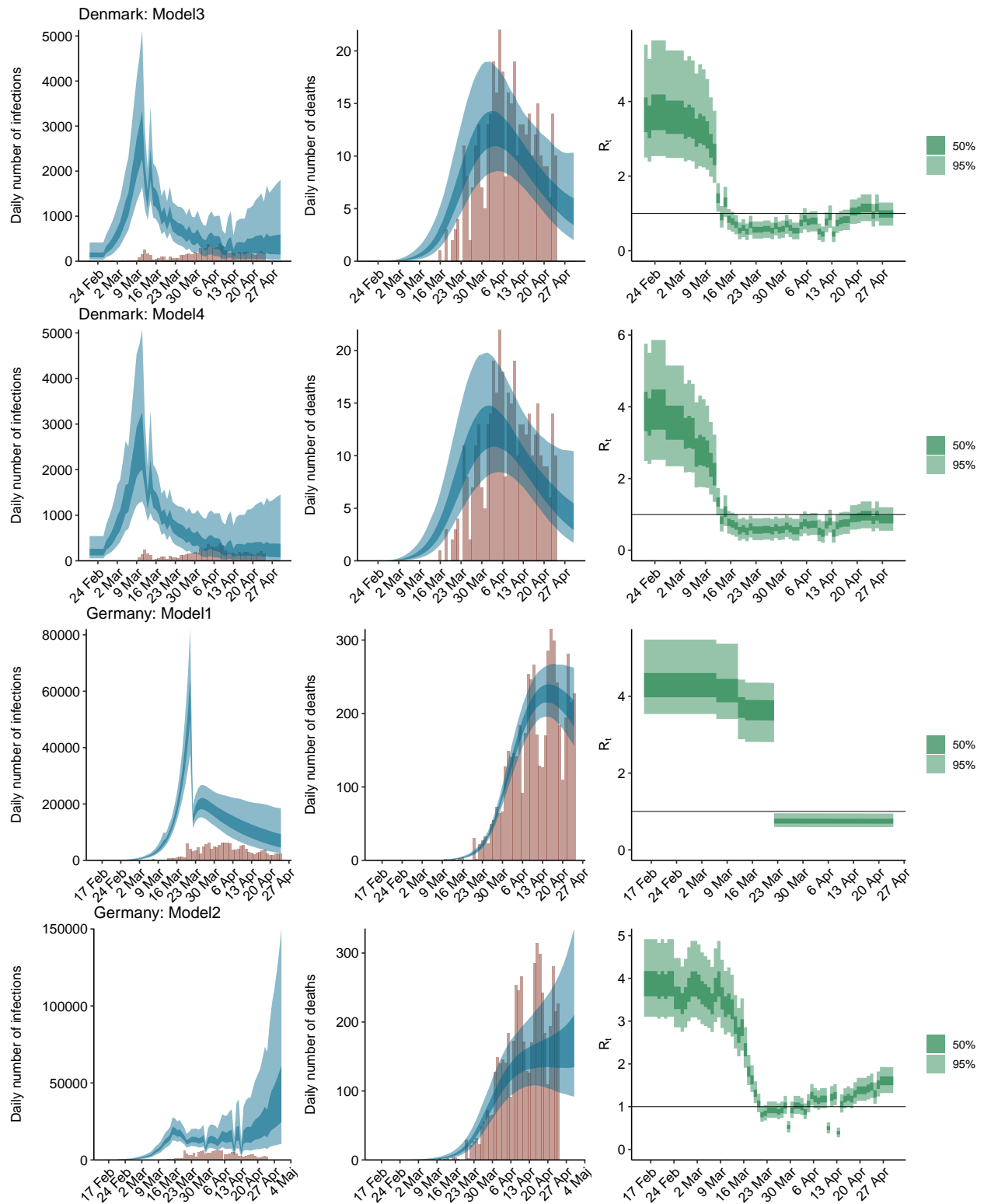


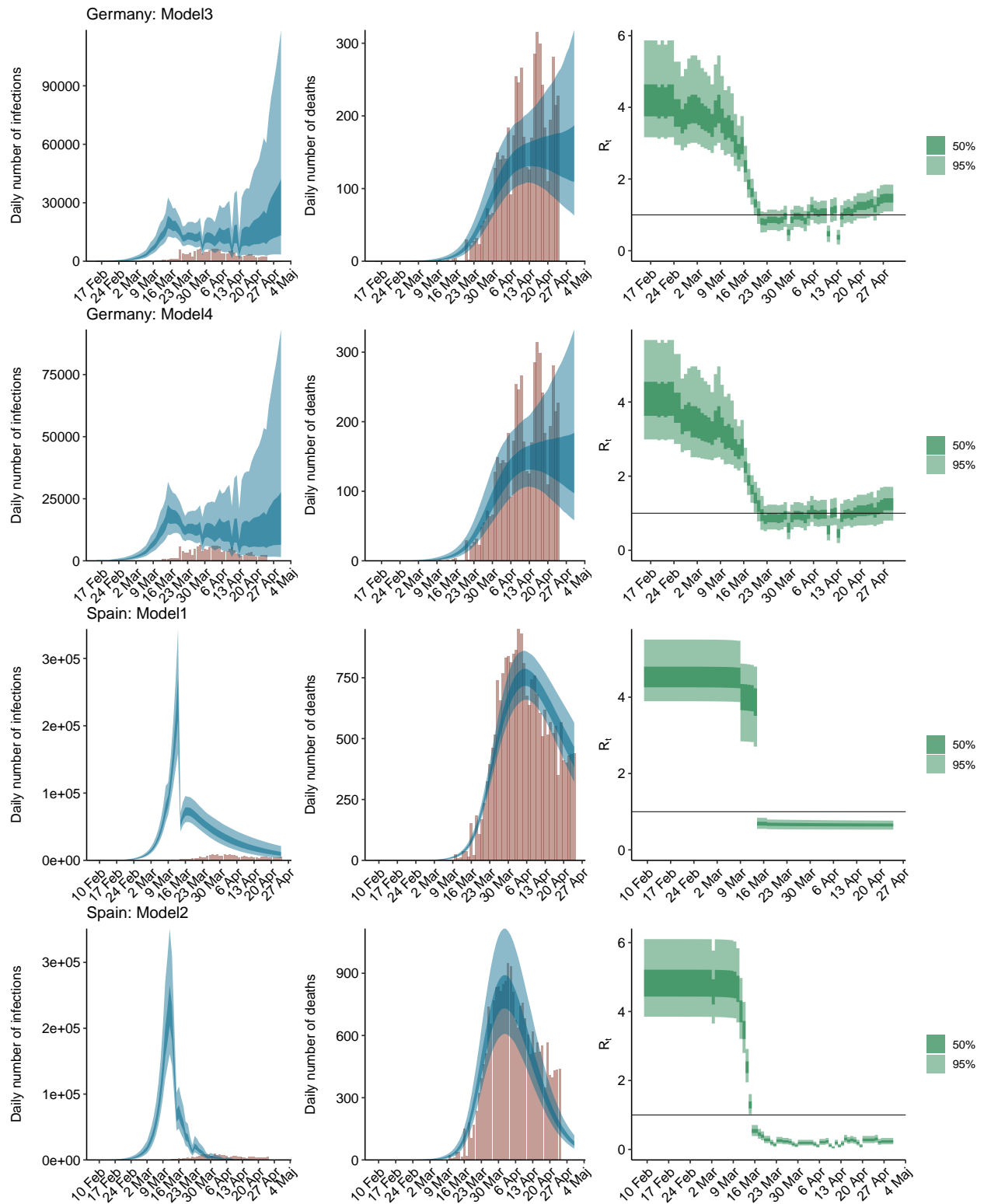


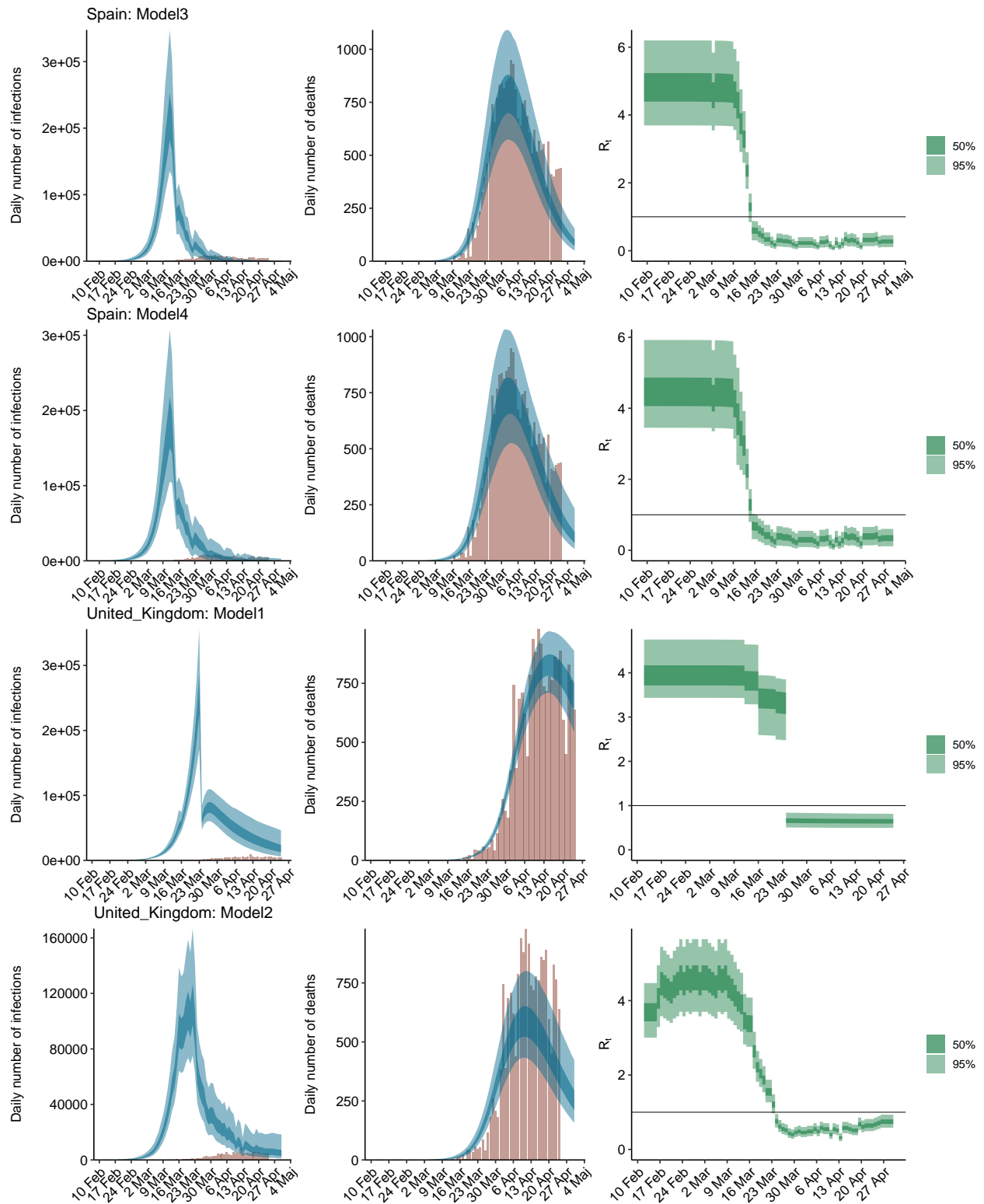


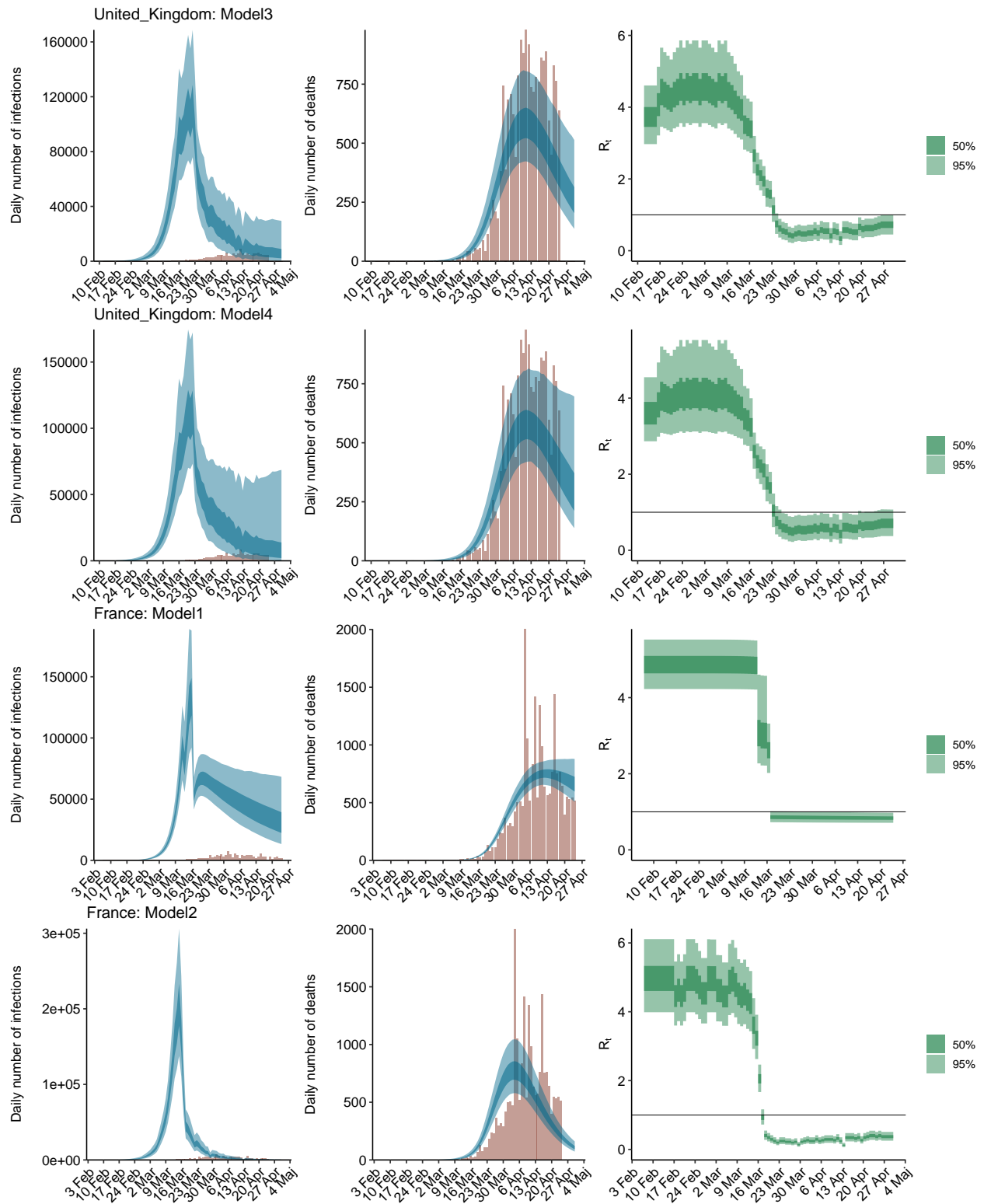
Infections, cases, deaths and R_t by country and model

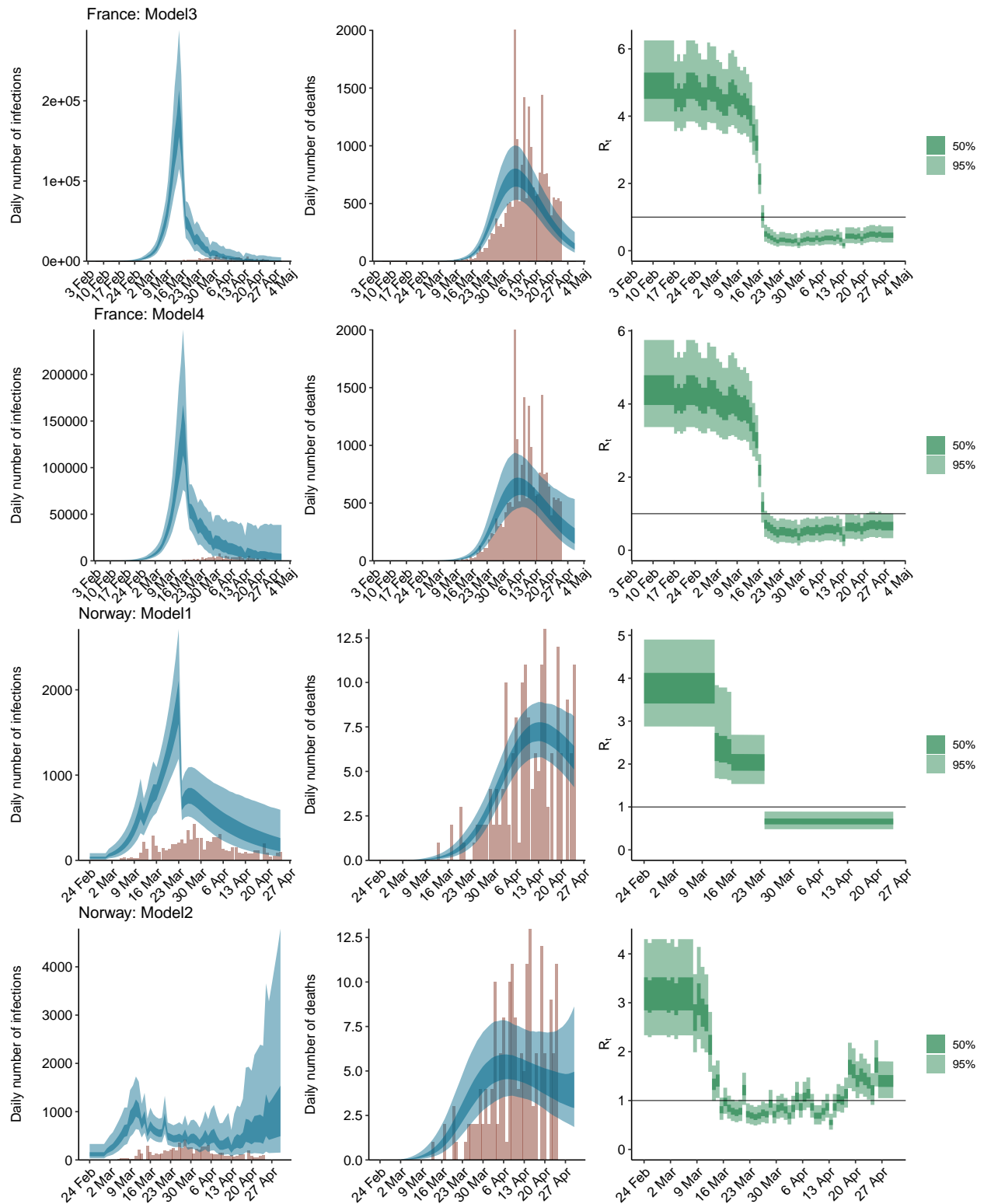


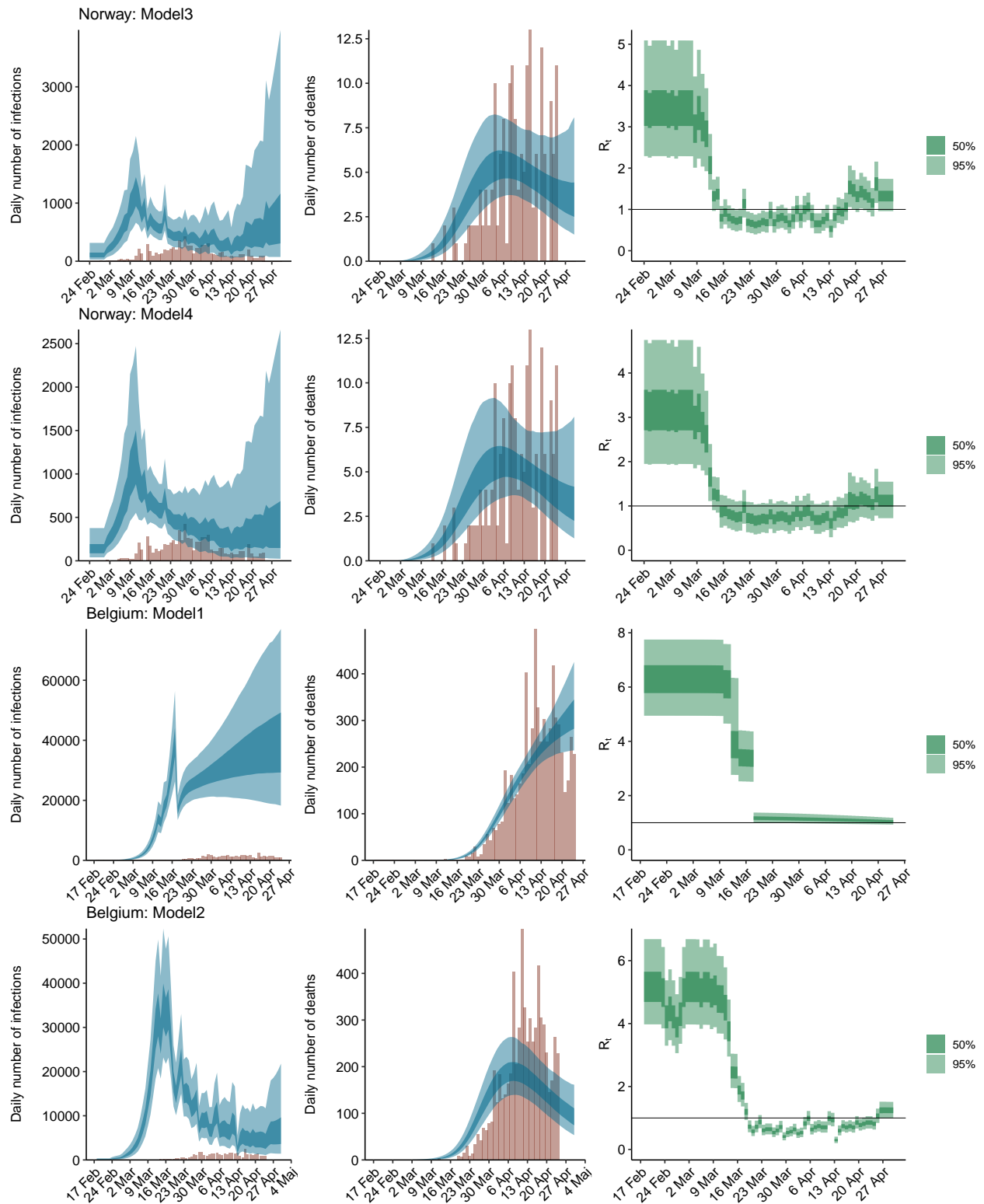


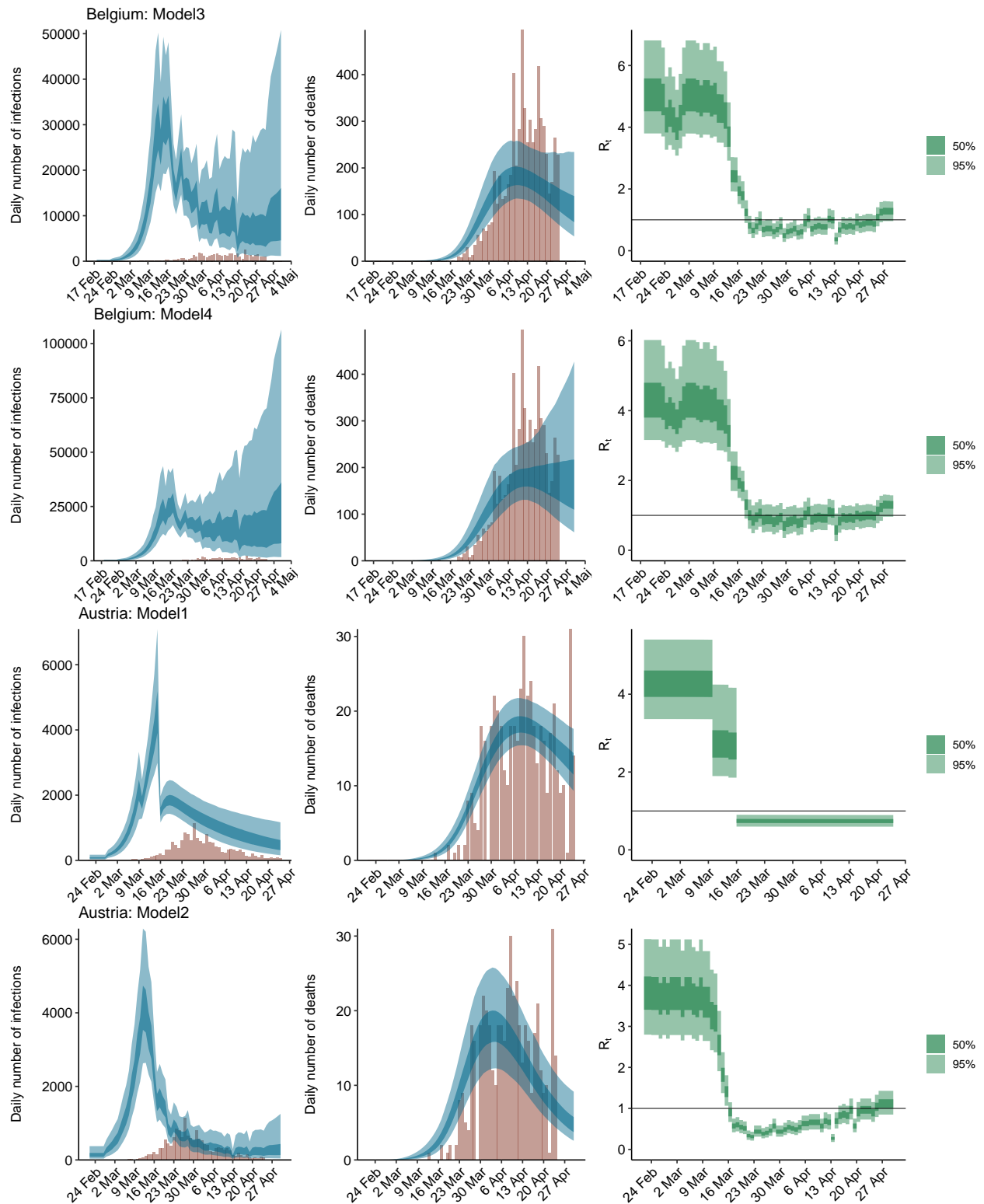


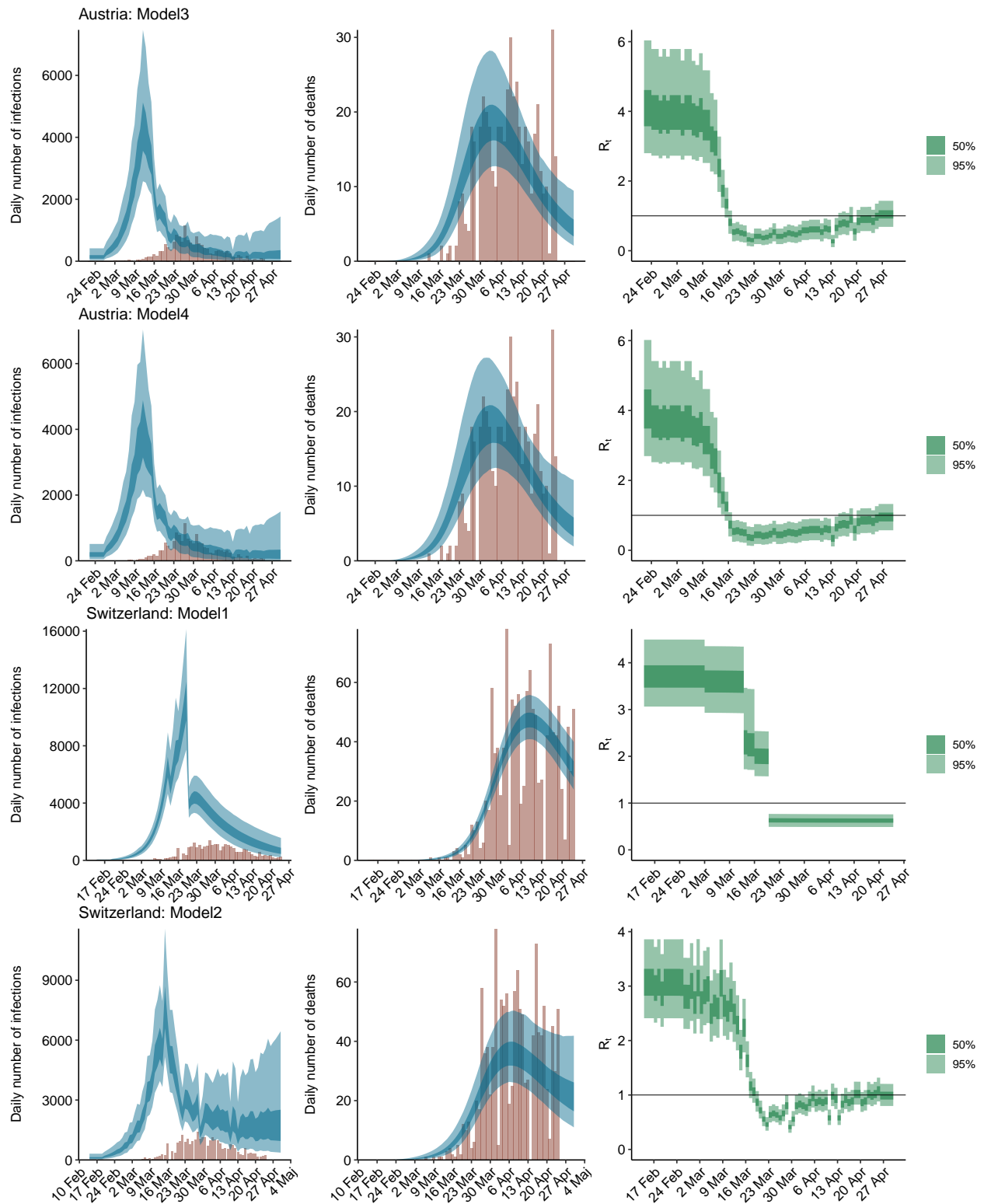


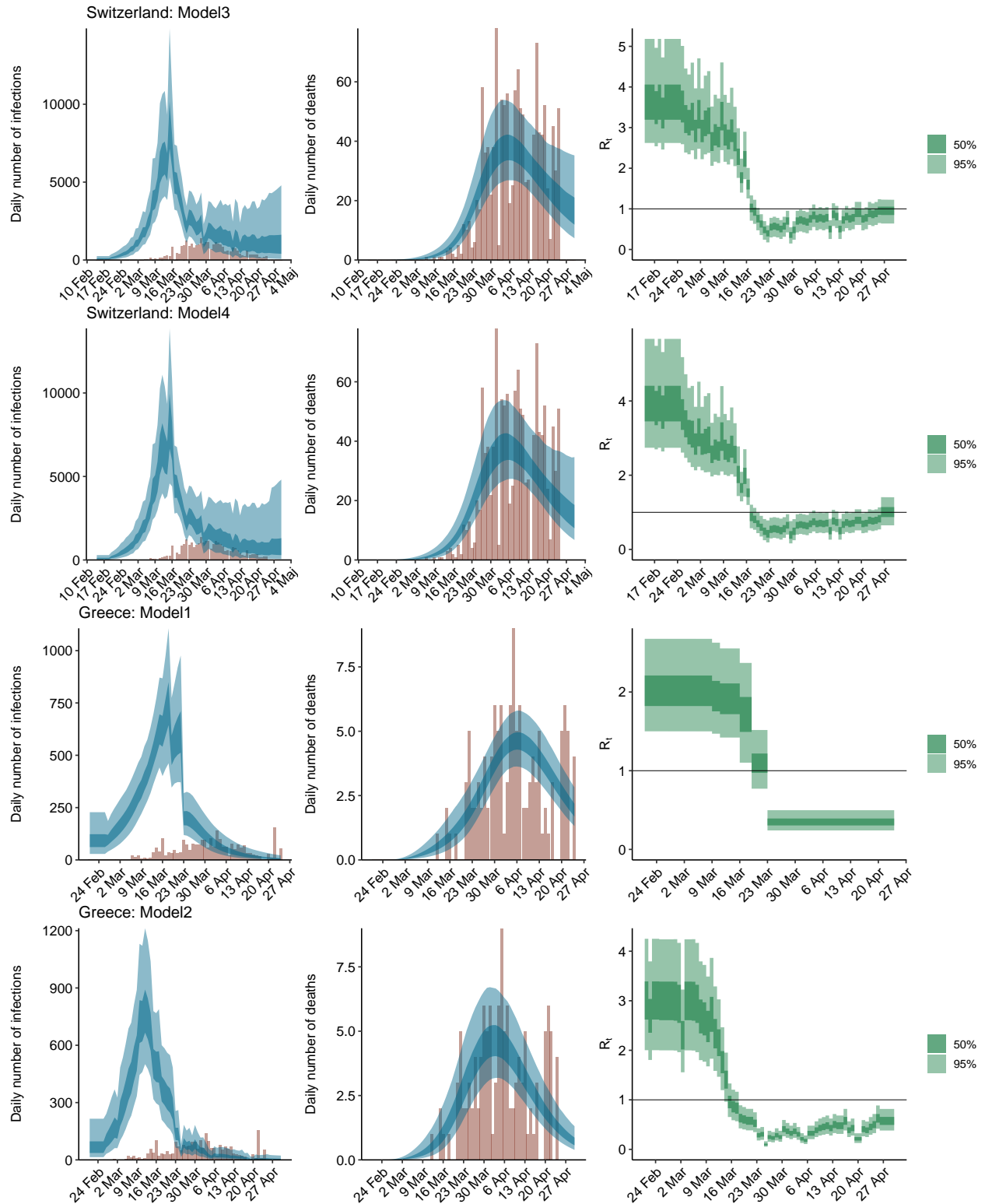


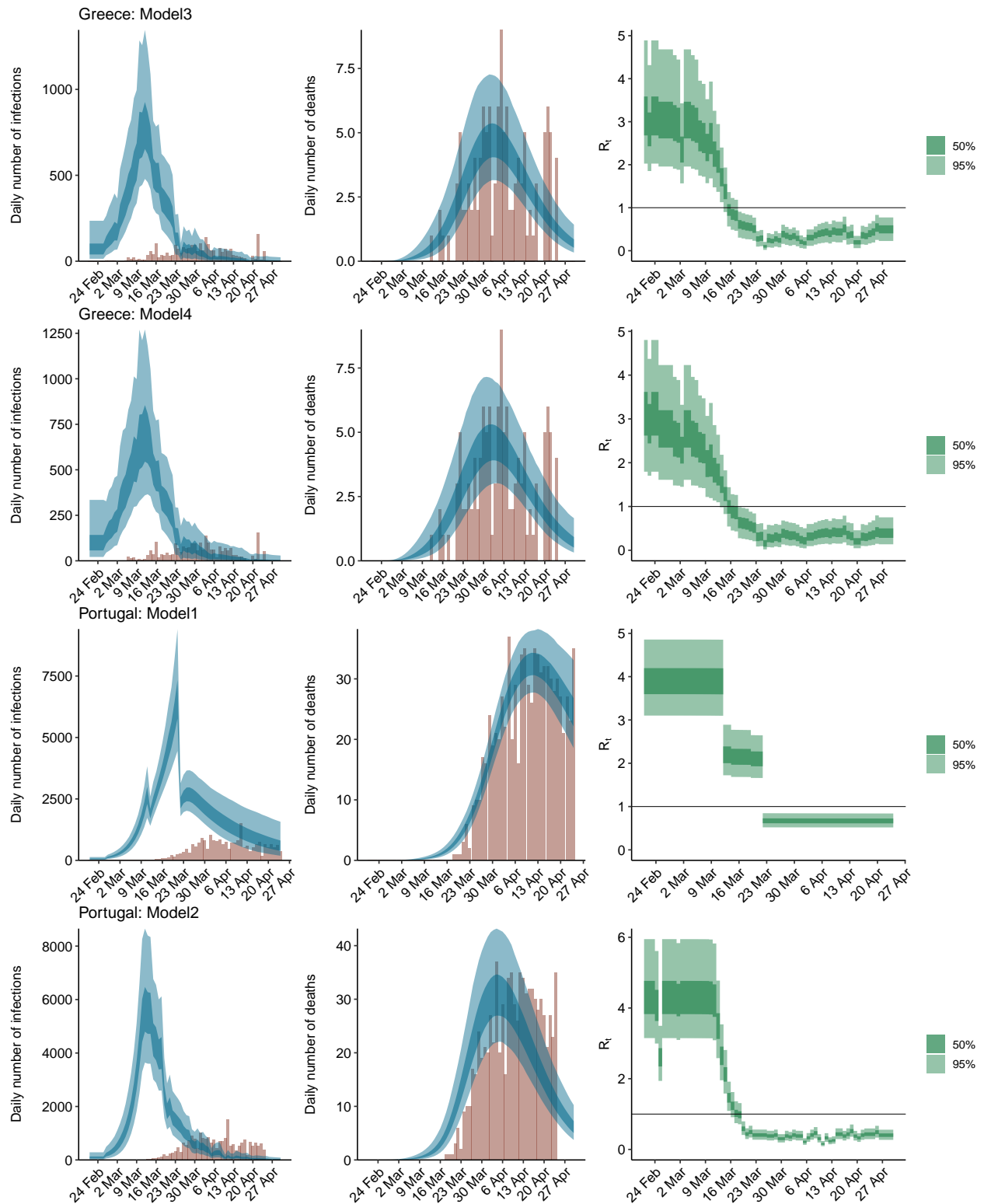


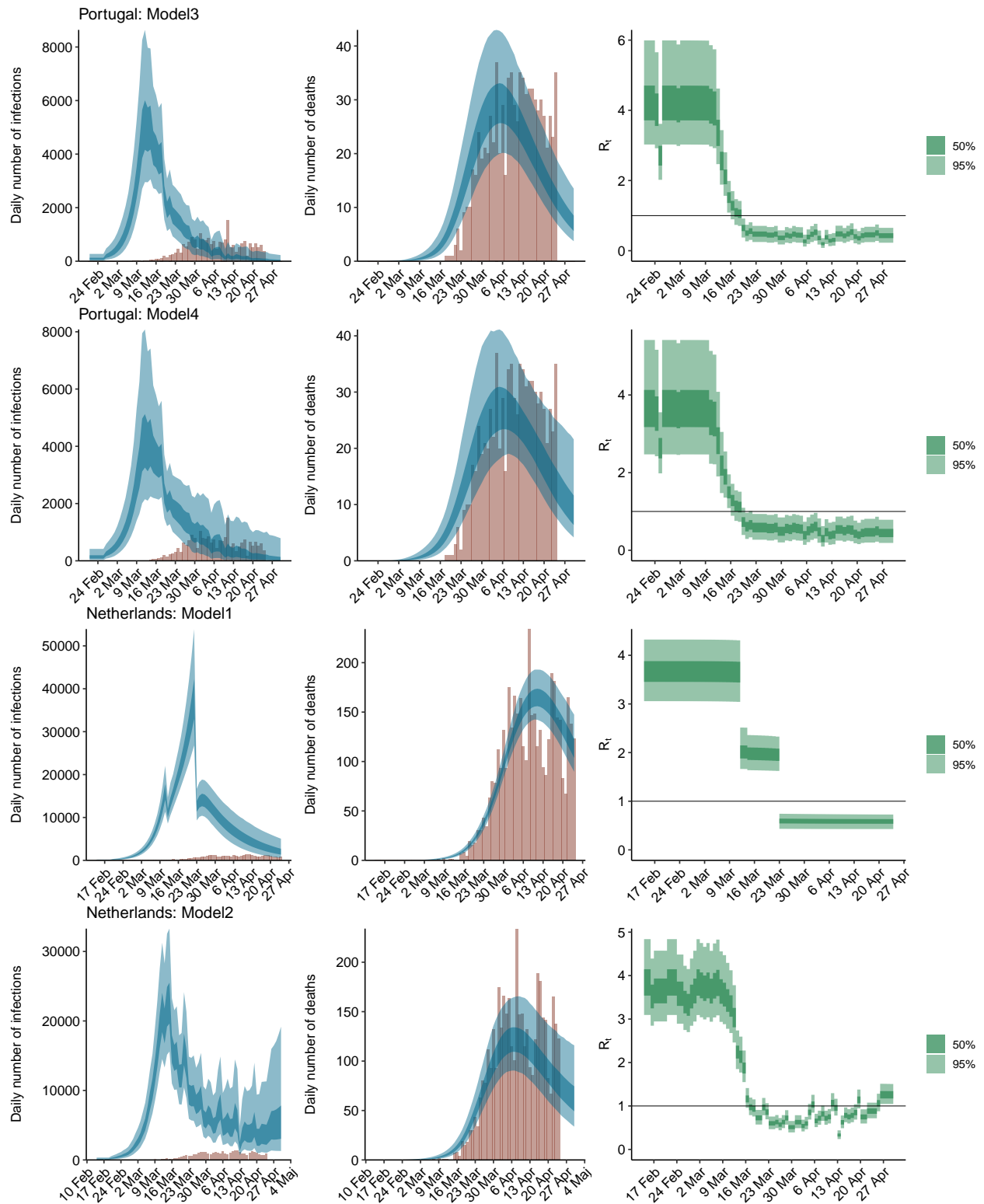


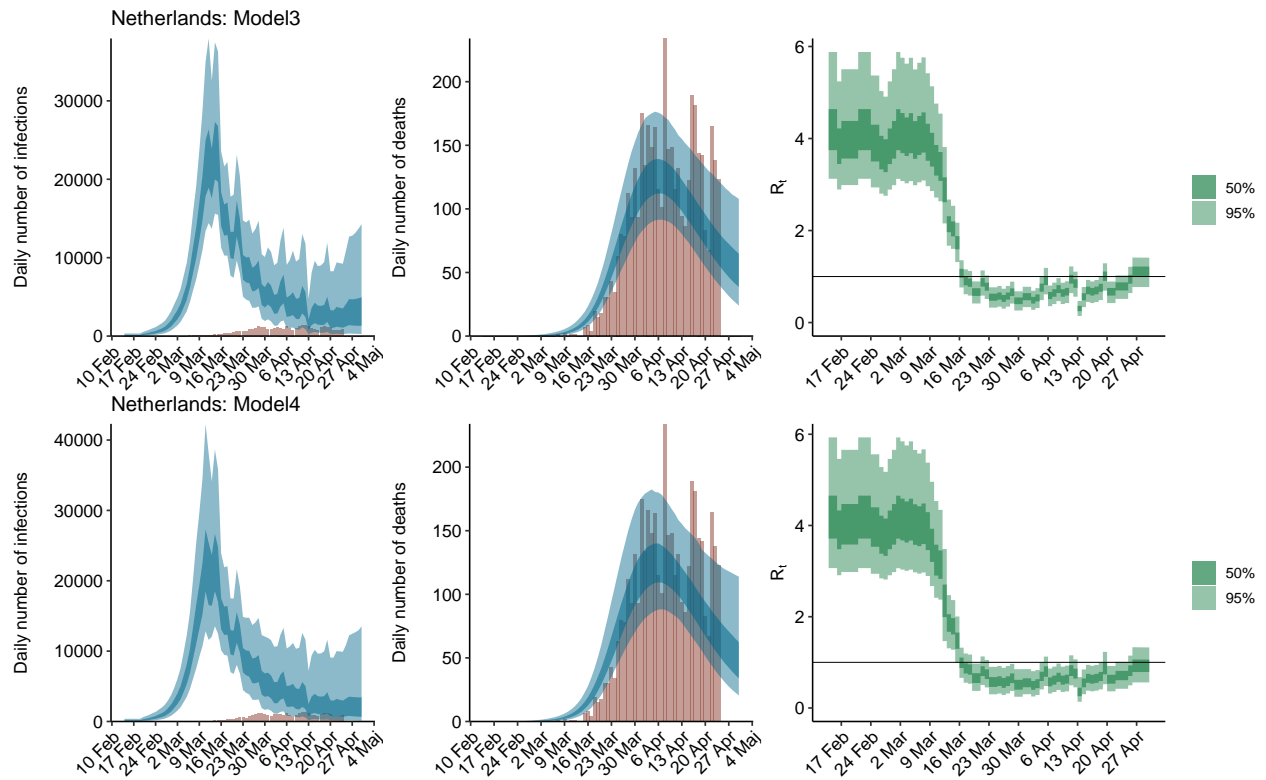












Reproducibility

Report git hash:

4473e0191f3b40756192ec0cf1bac62b027dbc78