Covid-19 in Italy

A (reproducible) analysis of the diffusion

Monica Chiogna and Carlo Gaetan

19 March, 2020

```
knitr::opts_chunk$set(tidy.opts=list(width.cutoff=60),tidy=TRUE)
```


Disclaimer

- We want to investigate the evolution of the coronavirus pandemic in Italy from a statistical perspective using aggregated data.
- Our point of view is that of surveillance with the goal of detecting important changes in the underlying (random) process as soon as possible after it has occured.
- We use data provided by Italian Civil Protection Department
- It is possible to select your region or county, but be careful that the estimation algorithm **fails** in the presence of a few observed cases (actually this is good news).
- This document is in a draft mode, and it is continuously updated.
- The layout of the draft must definitely be improved.

The statistical model

Let Y_t the counts (for instance confirmed cases, deaths) at day t.

It is quite common to model these counts with a Poisson distribution

$$Y_t|\mu_t \sim \text{Poisson}(\mu_t), \qquad t = 1, 2, \dots$$

The value $\mu_t > 0$ represents the expected number of counts at day t.

In order to catch the evolution of the viral infection we employ a Dynamic Generalized Linear Model (DGLM) for the Poisson distribution with log-linear link function for the expected counts, μ_t, namely

$$\begin{aligned} Y_t|\mu_t &\sim \text{Poisson}(\mu_t), \qquad t = 1, 2, \dots \\ \log(\mu_t) &= \alpha_t \\ \alpha_t &= \alpha_{t-1} + \beta_{t-1} + e_{1,t} \\ \beta_t &= \beta_{t-1} + e_{2,t} \end{aligned}$$

where $e_t = (e_{1,t}, e_{2,t})^\top \sim N_2(0, Q)$, $\theta_0 = (\alpha_0, \beta_0)^\top \sim N_2(m_0, P_0)$ are random quantities independently of each other. The covariance matrix

$$Q = \begin{pmatrix} \psi_1 & 0 \\ 0 & \psi_2 \end{pmatrix}$$

contains the unknown parameters $\psi_i > 0$, $i = 1, 2$, which need to be estimated.

The model specifies a local linear trend model for $\alpha_t = \log(\mu_t)$ and β_t is the time-varying slope of the local level α_t.

See

- Chiogna, M. and Gaetan, C. (2002), Dynamic generalized linear models with application to environmental epidemiology. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, **51**: 453-468. doi: http://dx.doi.org/10.1111/1467-9876.00280
- Helske, J. (2017). KFAS: Exponential Family State Space Models in R. *Journal of Statistical Software*,**78**: 1 - 39. doi: http://dx.doi.org/10.18637/jss.v078.i10
- West, M., Harrison, P.J. and Migon, H.S. (1985) Dynamic generalized linear models and Bayesian forecasting (with discussion). *Journal of the American Statistical Association* ,**80**: 73-97, doi: http://dx.doi.org/10.1080/01621459.1985.10477131

for more details.

Surveillance

The time-varying parameter β_t measures the slope at time t in the local linear trend model. By monitoring the dynamic of the reconstructed slope $\beta_{t|t-1}$ and the associated pointwise confidence intervals (CI_t), some insight can be gained on increase (CI_t above the line $y = 0$), decrease (CI_t below $y = 0$) or stabilization (CI_t intersects the line $y = 0$) of the diffusion. Here, $\beta_{t|t-1}$ represents the reconstruction of β_t at the day t, given the information on the previous days $I_{t-1} = \{y_{t-1}, \ldots, y_1\}$.

Confidence level $p_{upper} - p_{lower}$, with $0 < p_{lower} < p_{upper} < 1$, can be set by the following `R` code.

```
p.lower <- 0.025
p.upper <- 1 - p.lower
confidence <- p.upper - p.lower
```

In the present version, analyses are carried out at level ***0.95***.

Software

We use the R package KFAS. Model can also be adapted using R-INLA package.

Install packages `dygraphs`,`KFAS` and `xts` if not available

```
checkpackage <- function(package) {
    if (!package %in% installed.packages())
        install.packages(package)
}
checkpackage("dygraphs")
checkpackage("KFAS")
checkpackage("xts")
```

and load them.

```
library(dygraphs)
library(KFAS)
library(xts)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

Source of the data

We download the (raw) data from https://github.com/pcm-dpc/COVID-19/

```
repository <- "https://raw.githubusercontent.com/pcm-dpc/COVID-19/master/"
```

Results

With this code

```
step.ahead <- 3
```

we set the number of days, $s = 3$, for the ahead prediction. Please choose this number carefully, because the model is not suitable for long-term prediction.

Overall estimates

```
overall.dataset <- "dati-andamento-nazionale/dpc-covid19-ita-andamento-nazionale.csv"
overall.filename <- paste(repository, overall.dataset, sep = "")
Italy <- read.csv(overall.filename)
```

Several outcomes can be potentially monitored, that is

```
names(Italy[, -(1:2)])
```

```
##  [1] "ricoverati_con_sintomi"      "terapia_intensiva"
##  [3] "totale_ospedalizzati"        "isolamento_domiciliare"
##  [5] "totale_attualmente_positivi" "nuovi_attualmente_positivi"
##  [7] "dimessi_guariti"             "deceduti"
##  [9] "totale_casi"                 "tamponi"
```

Here is a translation of the meaning.

Italian	English
ricoverati_con_sintomi	Hospitalized with symptoms
terapia_intensiva	Under critical care
totale_ospedalizzati	Hospitalized patients
isolamento_domiciliare	Under isolation at home
totale_attualmente_positivi	Current cases
nuovi_attualmente_positivi	New cases
dimessi_guariti	Released from the hospitals
deceduti	Deaths
totale_casi	Total number of confirmed cases
tamponi	Number of swabs

It is worth noting that some outcomes present negative counts in some regions. It looks like some of these negative counts are redesignations. Outcomes presenting negative values cannot be analyzed using the proposed model.

We consider one outcome.

```
select <- "nuovi_attualmente_positivi"
y <- as.numeric(unlist(subset(Italy, select = select)))
```

We extract the timeseries.

```
myDateTimeStr1 <- Italy$data
myPOSIXct1 <- as.POSIXct(myDateTimeStr1, format = "%Y-%m-%d %H:%M:%S")
days <- as.Date(myPOSIXct1)
days.ahead <- seq(days[length(days)] + 1, days[length(days)] +
    step.ahead, by = 1)
counts <- xts(c(y, rep(NA, step.ahead)), order.by = c(days, days.ahead),
    frequency = 7)
```

Model fitting

```
model_poisson <- SSModel(counts ~ -1 + SSMtrend(2, Q = list(matrix(NA,
    1, 1), matrix(NA, 1, 1))), distribution = "poisson")
```

```
## Warning in `dim<-.zoo`(`*tmp*`, value = length(x)): setting this dimension may
## lead to an invalid zoo object
```

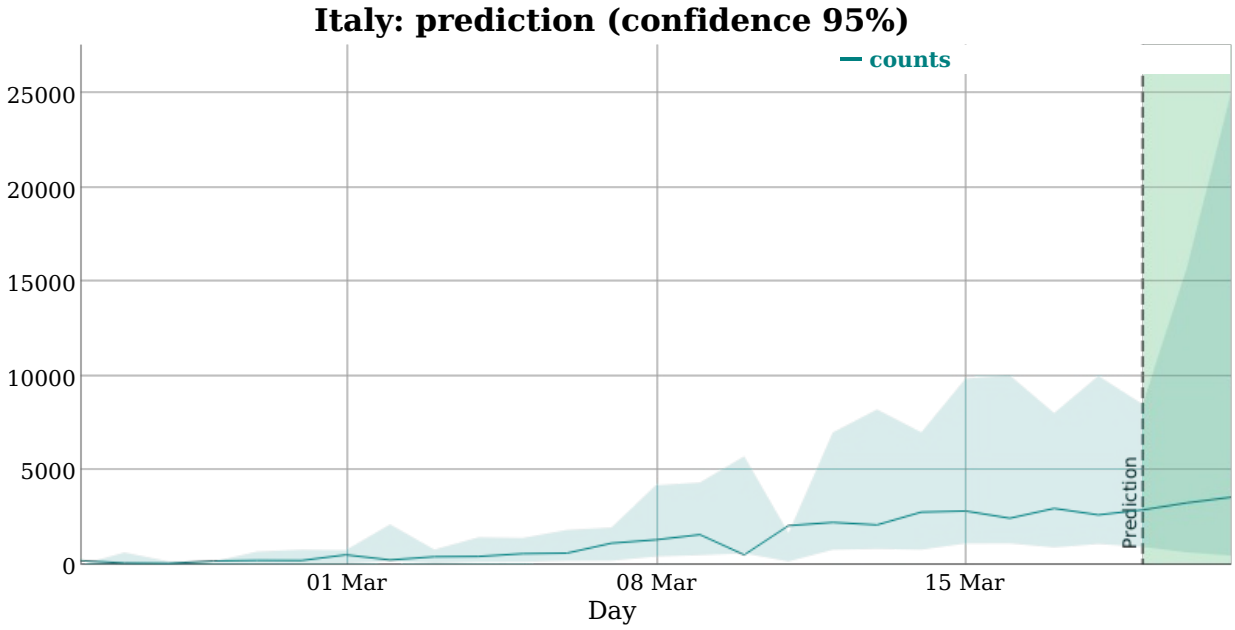
```
## Warning in `dim<-.zoo`(`*tmp*`, value = c(n, p)): setting this dimension may
## lead to an invalid zoo object
```

```
fit_poisson <- fitSSM(model_poisson, inits = c(-2, -2), method = "BFGS")
imp <- importanceSSM(fit_poisson$model, type = "signals", filtered = TRUE,
    nsim = 2000)
lower <- apply(imp$samples, 1, quantile, prob = p.lower)
med <- apply(imp$samples, 1, median)
upper <- apply(imp$samples, 1, quantile, prob = p.upper)
```

The warning messages do not affect the results but we will take care in the future to eliminate them.

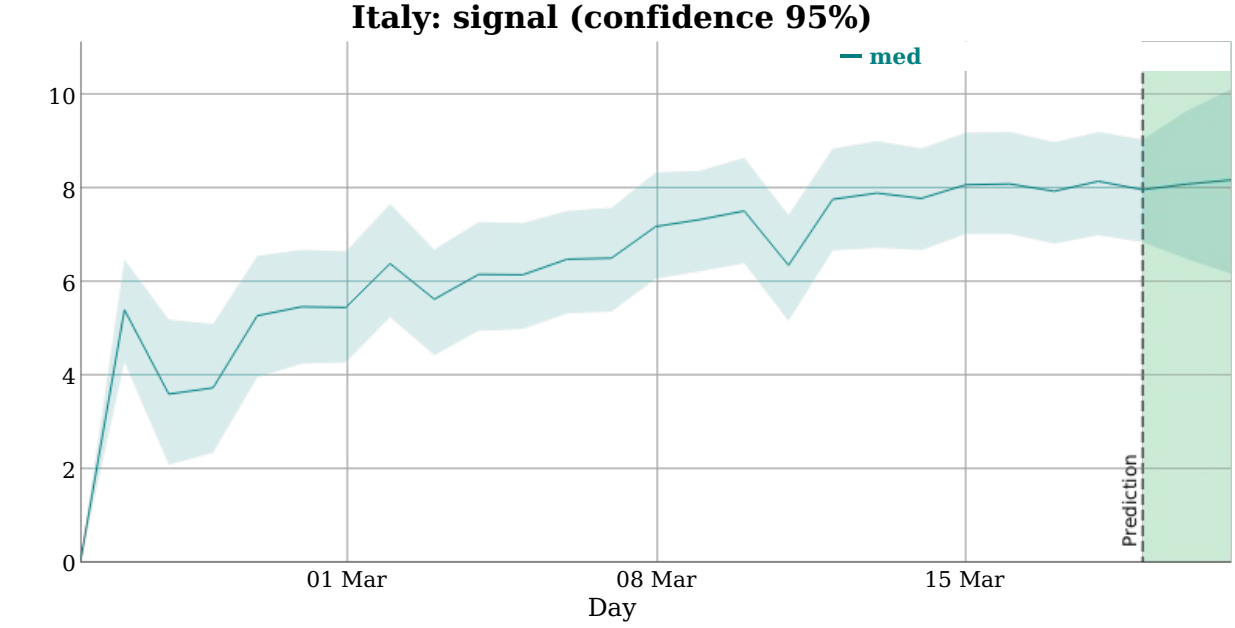
The plot of $\mu_{t+s|t} = \exp(\alpha_{t+s|t})$.

```
mu.lower <- exp(lower)
mu.upper <- exp(upper)
mu.med <- xts(exp(med), order.by = c(days, days.ahead), frequency = 7)
counts[days.ahead] <- mu.med[days.ahead]
mu <- xts(x = as.matrix(cbind(counts, mu.lower, mu.upper)), order.by = c(days,
    days.ahead))
p <- dygraph(mu, main = paste("Italy: prediction (confidence ",
    100 * confidence, "%)", sep = ""), xlab = "Day", height = 400,
    width = 800)
p <- p %>% dySeries(c("mu.lower", "counts", "mu.upper"), label = "counts")
p <- p %>% dyLegend(show = "always", hideOnMouseOut = FALSE)
p <- p %>% dyShading(from = days.ahead[1], to = days.ahead[step.ahead],
    color = "#CCEBD6")
p <- p %>% dyEvent(days.ahead[1], "Prediction", labelLoc = "bottom")
p
```



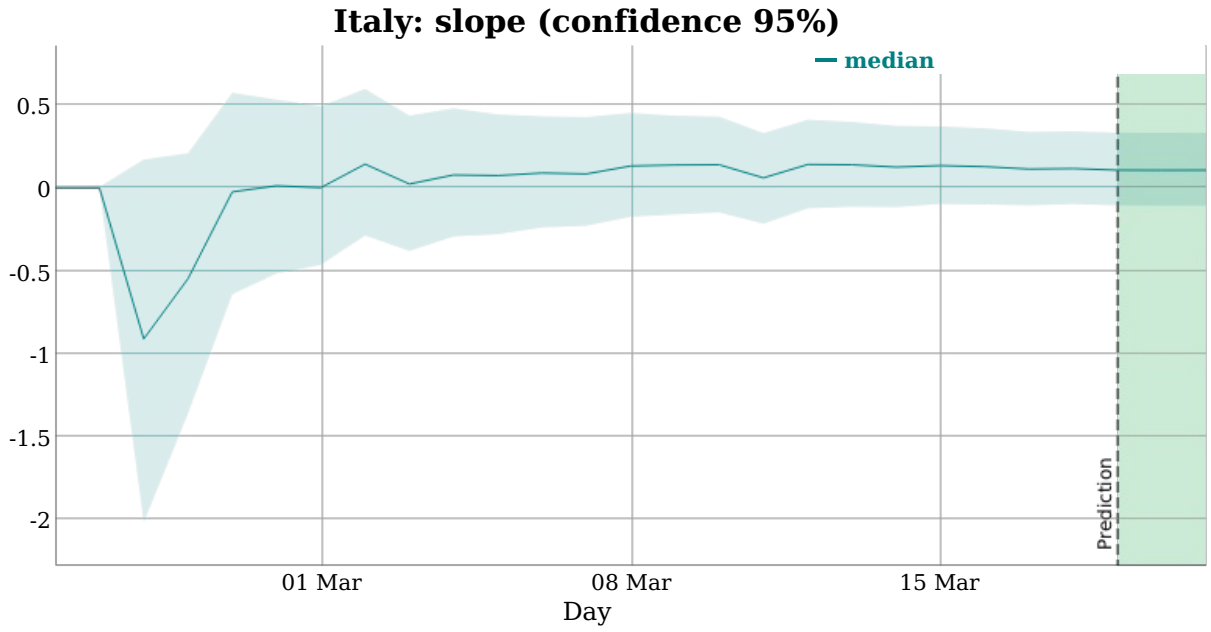
Plot of $\alpha_{t|t-1}$, i.e the signal

```
signal <- xts(cbind(lower, med, upper), order.by = c(days, days.ahead),
    frequency = 7)
p <- dygraph(signal, main = paste("Italy: signal (confidence ",
    100 * confidence, "%)", sep = ""), xlab = "Day", height = 400,
    width = 800)
p <- p %>% dySeries(c("lower", "med", "upper"))
p <- p %>% dyLegend(show = "always", hideOnMouseOut = FALSE)
p <- p %>% dyShading(from = days.ahead[1], to = days.ahead[step.ahead],
    color = "#CCEBD6")
p <- p %>% dyEvent(days.ahead[1], "Prediction", labelLoc = "bottom")
p
```



Plot of the $\beta_{t|t-1}$, i.e the estimated slope

```
imp <- importanceSSM(fit_poisson$model, type = "states", filtered = TRUE,
    nsim = 2000)
lower <- apply(imp$samples[, 2, ], 1, quantile, prob = p.lower)
med <- apply(imp$samples[, 2, ], 1, quantile, prob = 0.5)
upper <- apply(imp$samples[, 2, ], 1, quantile, prob = p.upper)
slope <- xts(cbind(lower, med, upper), order.by = c(days, days.ahead),
    frequency = 7)
p <- dygraph(slope, main = paste("Italy: slope (confidence ",
    100 * confidence, "%)", sep = ""), xlab = "Day", height = 400,
    width = 800) %>% dySeries(c("lower", "med", "upper"), label = "median") %>%
    dyLegend(show = "always", hideOnMouseOut = FALSE)
p <- p %>% dyShading(from = days.ahead[1], to = days.ahead[step.ahead],
    color = "#CCEBD6")
p <- p %>% dyEvent(days.ahead[1], "Prediction", labelLoc = "bottom")
p
```



Estimates by region

Download the regional figures

```
regional.dataset <- "dati-regioni/dpc-covid19-ita-regioni.csv"
regional.filename <- paste(repository, regional.dataset, sep = "")
regions <- read.csv(regional.filename)
```

We select one region

```
myregion <- "Veneto"  # Obviously...
```

In our case Veneto

```
y <- as.numeric(unlist(subset(regions, subset = (denominazione_regione ==
    myregion), select = select)))
myDateTimeStr1 <- regions$data[regions$denominazione_regione ==
    myregion]
```

```
myPOSIXct1 <- as.POSIXct(myDateTimeStr1, format = "%Y-%m-%d %H:%M:%S")
days <- as.Date(myPOSIXct1)
days.ahead <- seq(days[length(days)] + 1, days[length(days)] +
    step.ahead, by = 1)
counts <- xts(c(y, rep(NA, step.ahead)), order.by = c(days, days.ahead),
    frequency = 7)
```

and we fit the DGLM

```
model_poisson <- SSModel(counts ~ -1 + SSMtrend(2, Q = list(matrix(NA,
    1, 1), matrix(NA, 1, 1))), distribution = "poisson")
```

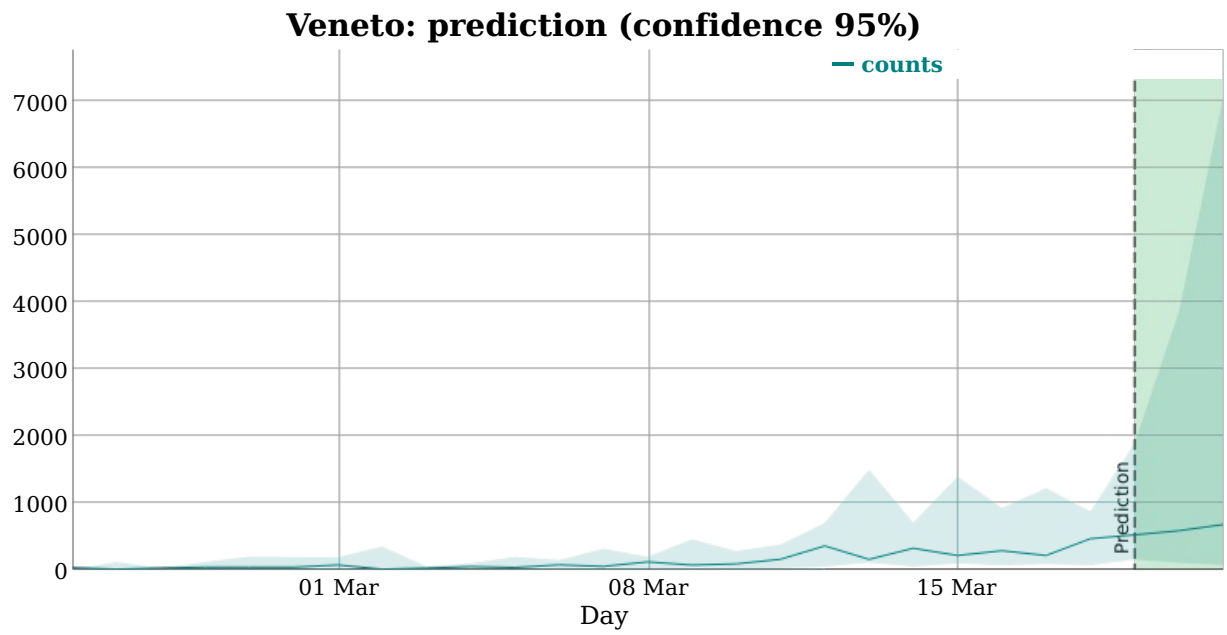
```
## Warning in `dim<-.zoo`(`*tmp*`, value = length(x)): setting this dimension may
## lead to an invalid zoo object
```

```
## Warning in `dim<-.zoo`(`*tmp*`, value = c(n, p)): setting this dimension may
## lead to an invalid zoo object
```

```
fit_poisson <- fitSSM(model_poisson, inits = c(-2, -2), method = "BFGS")
imp <- importanceSSM(fit_poisson$model, type = "signals", filtered = TRUE,
    nsim = 2000)
lower <- apply(imp$samples, 1, quantile, prob = p.lower)
med <- apply(imp$samples, 1, median)
upper <- apply(imp$samples, 1, quantile, prob = p.upper)
```

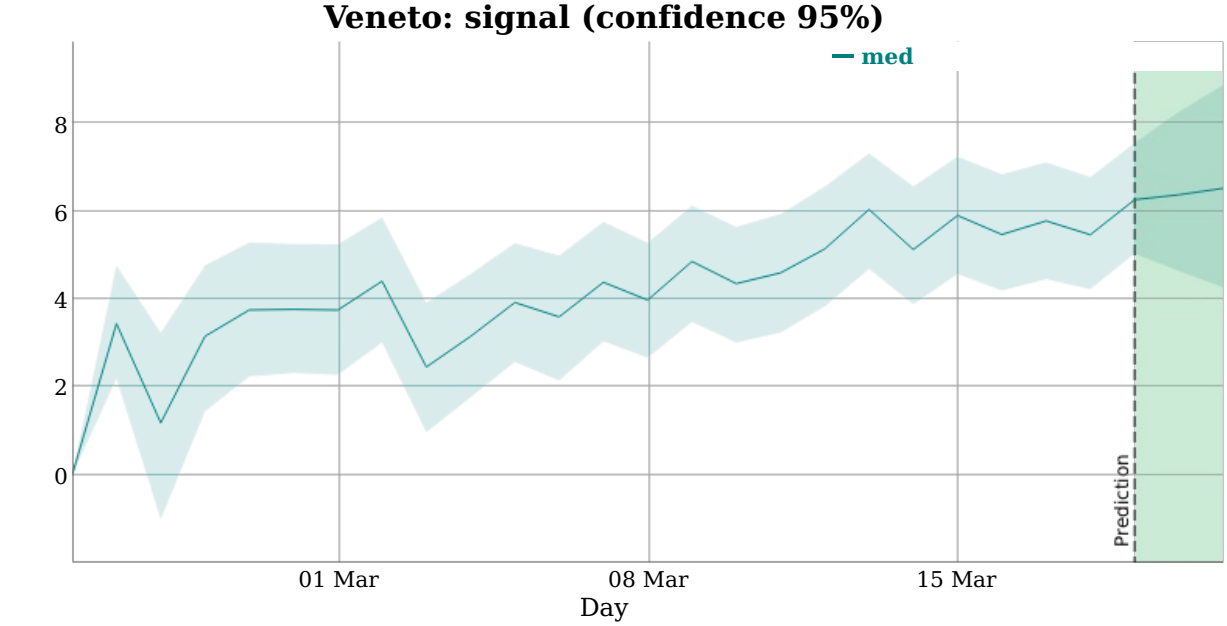
Plot of $\mu_{t+s|t} = \exp \alpha_{t+s|t}$

```
mu.lower <- exp(lower)
mu.upper <- exp(upper)
mu.med <- xts(exp(med), order.by = c(days, days.ahead), frequency = 7)
counts[days.ahead] <- mu.med[days.ahead]
mu <- xts(x = as.matrix(cbind(counts, mu.lower, mu.upper)), order.by = c(days,
    days.ahead))
p <- dygraph(mu, main = paste(myregion, ": prediction (confidence ",
    100 * confidence, "%)", sep = ""), xlab = "Day", height = 400,
    width = 800)
p <- p %>% dySeries(c("mu.lower", "counts", "mu.upper"), label = "counts")
p <- p %>% dyLegend(show = "always", hideOnMouseOut = FALSE)
p <- p %>% dyShading(from = days.ahead[1], to = days.ahead[step.ahead],
    color = "#CCEBD6") %>% dyEvent(days.ahead[1], "Prediction",
    labelLoc = "bottom")
p
```



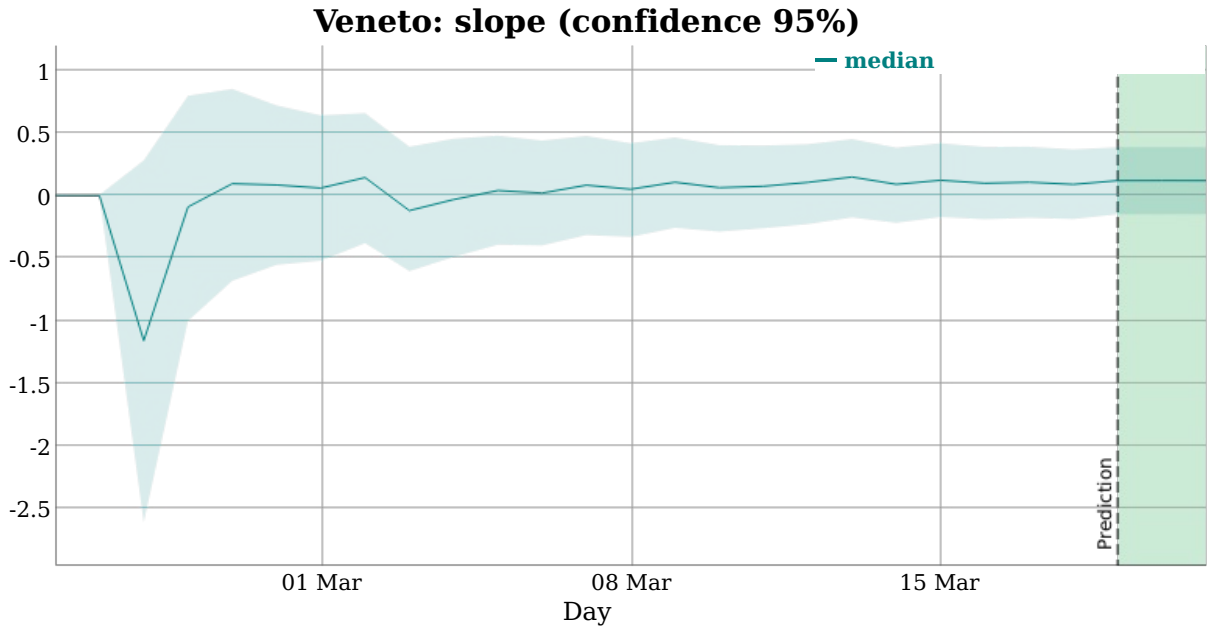
Plot of $\alpha_{t|t-1}$, i.e the signal

```
signal <- xts(cbind(lower, med, upper), order.by = c(days, days.ahead),
    frequency = 7)
p <- dygraph(signal, main = paste(myregion, ": signal (confidence ",
    100 * confidence, "%)", sep = ""), xlab = "Day", height = 400,
    width = 800) %>% dySeries(c("lower", "med", "upper")) %>%
    dyLegend(show = "always", hideOnMouseOut = FALSE)
p <- p %>% dyShading(from = days.ahead[1], to = days.ahead[step.ahead],
    color = "#CCEBD6")
p <- p %>% dyEvent(days.ahead[1], "Prediction", labelLoc = "bottom")
p
```



Plot of the $\beta_{t|t-1}$, i.e the estimated slope

```
imp <- importanceSSM(fit_poisson$model, type = "states", filtered = TRUE,
    nsim = 2000)
lower <- apply(imp$samples[, 2, ], 1, quantile, prob = p.lower)
med <- apply(imp$samples[, 2, ], 1, quantile, prob = 0.5)
upper <- apply(imp$samples[, 2, ], 1, quantile, prob = p.upper)
slope <- xts(cbind(lower, med, upper), order.by = c(days, days.ahead),
    frequency = 7)
p <- dygraph(slope, main = paste(myregion, ": slope (confidence ",
    100 * confidence, "%)", sep = ""), xlab = "Day", height = 400,
    width = 800)
p <- p %>% dySeries(c("lower", "med", "upper"), label = "median")
p <- p %>% dyLegend(show = "always", hideOnMouseOut = FALSE)
p <- p %>% dyShading(from = days.ahead[1], to = days.ahead[step.ahead],
    color = "#CCEBD6")
p <- p %>% dyEvent(days.ahead[1], "Prediction", labelLoc = "bottom")
p
```



Estimates by county

Download the county figures

```
county.dataset <- "dati-province/dpc-covid19-ita-province.csv"
county.filename <- paste(repository, county.dataset, sep = "")
counties <- read.csv(county.filename)
```

We select one county

```
mycounty <- "Treviso"  # Obviously...
```

In our case Treviso

```
select <- "totale_casi"
y <- as.numeric(unlist(subset(counties, subset = (denominazione_provincia ==
    mycounty), select = select)))
myDateTimeStr1 <- regions$data[regions$denominazione_regione ==
```

```
    myregion]
myPOSIXct1 <- as.POSIXct(myDateTimeStr1, format = "%Y-%m-%d %H:%M:%S")
days <- as.Date(myPOSIXct1)
days.ahead <- seq(days[length(days)] + 1, days[length(days)] +
    step.ahead, by = 1)
counts <- xts(c(y, rep(NA, step.ahead)), order.by = c(days, days.ahead),
    frequency = 7)
```

and we fit the DGLM

```
model_poisson <- SSModel(counts ~ -1 + SSMtrend(2, Q = list(matrix(NA,
    1, 1), matrix(NA, 1, 1))), distribution = "poisson")
```

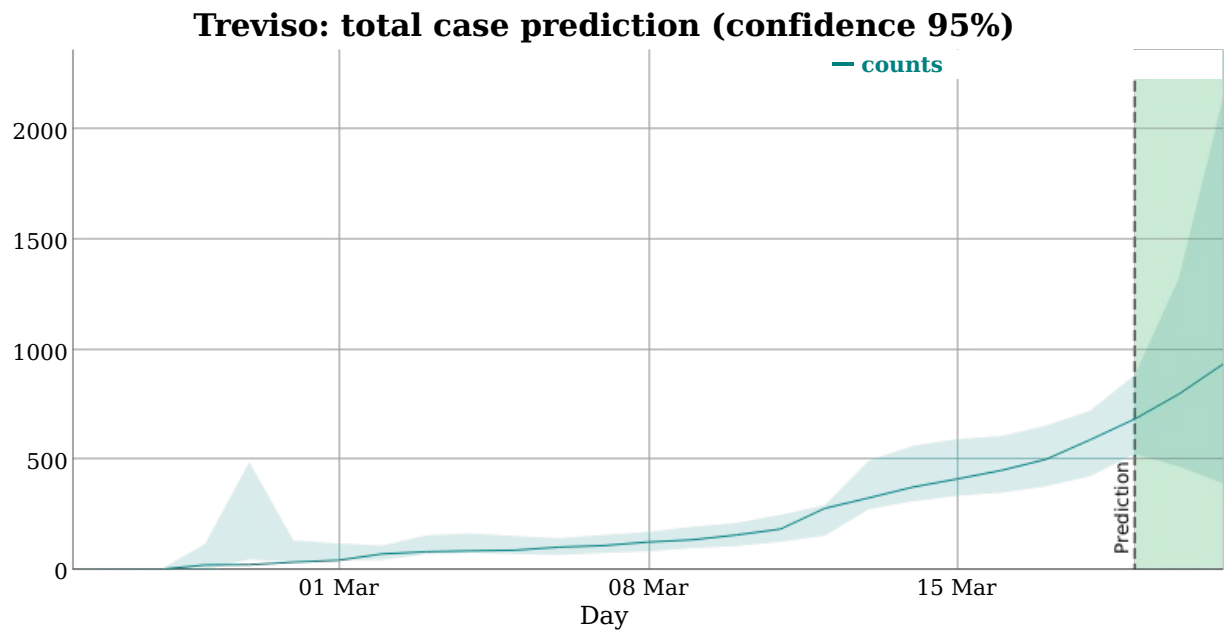
```
## Warning in `dim<-.zoo`(`*tmp*`, value = length(x)): setting this dimension may
## lead to an invalid zoo object
```

```
## Warning in `dim<-.zoo`(`*tmp*`, value = c(n, p)): setting this dimension may
## lead to an invalid zoo object
```

```
fit_poisson <- fitSSM(model_poisson, inits = c(0, 0), method = "BFGS")
imp <- importanceSSM(fit_poisson$model, type = "signals", filtered = TRUE,
    nsim = 2000)
lower <- apply(imp$samples, 1, quantile, prob = p.lower)
med <- apply(imp$samples, 1, median)
upper <- apply(imp$samples, 1, quantile, prob = p.upper)
```

Plot of $\mu_{t+s|t} = \exp \alpha_{t+s|t}$

```
mu.lower <- exp(lower)
mu.upper <- exp(upper)
mu.med <- xts(exp(med), order.by = c(days, days.ahead), frequency = 7)
counts[days.ahead] <- mu.med[days.ahead]
mu <- xts(x = as.matrix(cbind(counts, mu.lower, mu.upper)), order.by = c(days,
    days.ahead))
p <- dygraph(mu, main = paste(mycounty, ": total case prediction (confidence ",
    100 * confidence, "%)", sep = ""), xlab = "Day", height = 400,
    width = 800)
p <- p %>% dySeries(c("mu.lower", "counts", "mu.upper"), label = "counts")
p <- p %>% dyLegend(show = "always", hideOnMouseOut = FALSE) %>%
    dyShading(from = days.ahead[1], to = days.ahead[step.ahead],
        color = "#CCEBD6")
p <- p %>% dyEvent(days.ahead[1], "Prediction", labelLoc = "bottom")
p
```



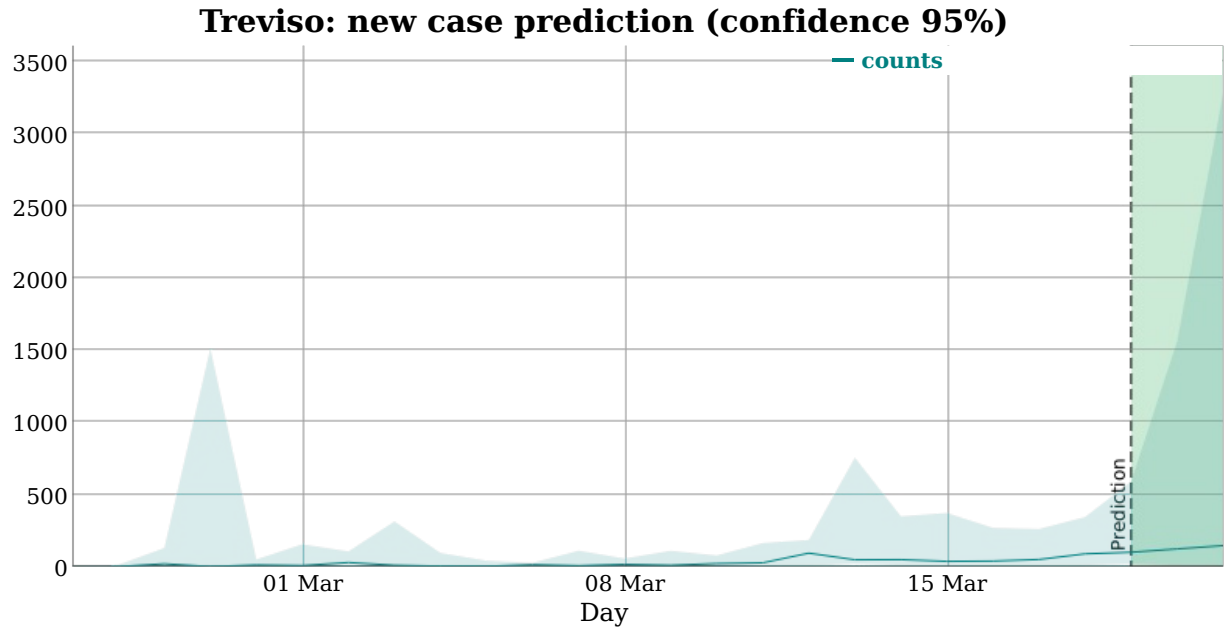

```
y <- as.numeric(unlist(subset(counties, subset = (denominazione_provincia ==
    mycounty), select = select)))
y <- diff(y)
days <- days[-1]
counts <- xts(c(y, rep(NA, step.ahead)), order.by = c(days, days.ahead),
    frequency = 7)
model_poisson <- SSModel(counts ~ -1 + SSMtrend(2, Q = list(matrix(NA,
    1, 1), matrix(NA, 1, 1))), distribution = "poisson")
```

```
## Warning in `dim<-.zoo`(`*tmp*`, value = length(x)): setting this dimension may
## lead to an invalid zoo object
```

```
## Warning in `dim<-.zoo`(`*tmp*`, value = c(n, p)): setting this dimension may
## lead to an invalid zoo object
```

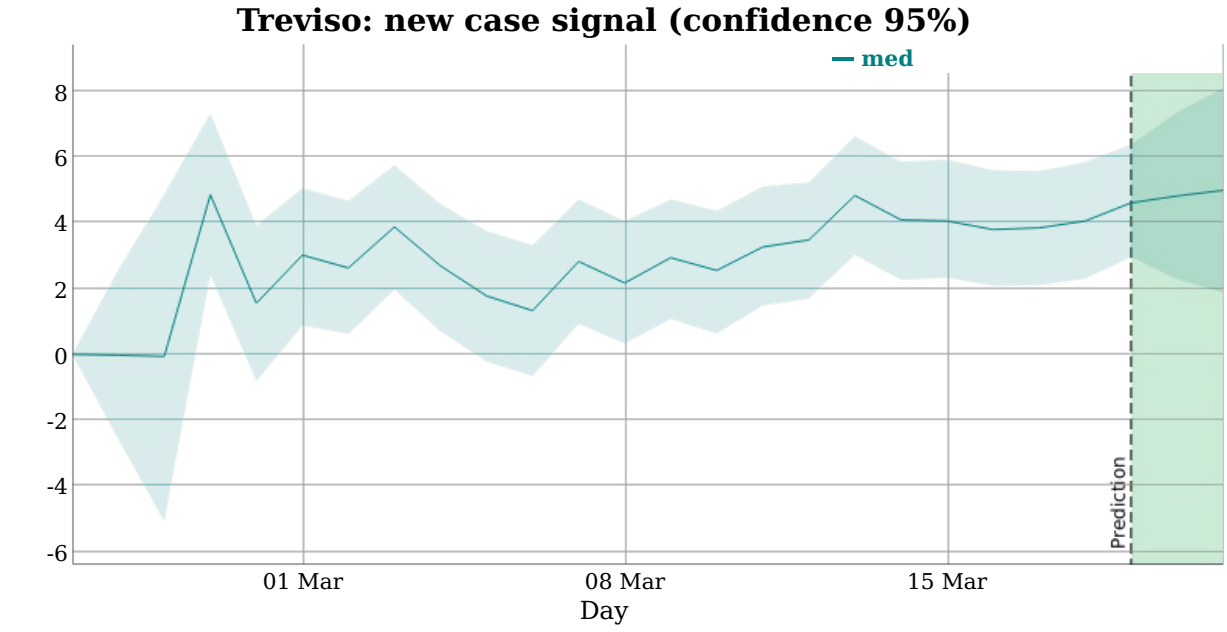
```
fit_poisson <- fitSSM(model_poisson, inits = c(0, 0), method = "BFGS")
imp <- importanceSSM(fit_poisson$model, type = "signals", filtered = TRUE,
    nsim = 2000)
lower <- apply(imp$samples, 1, quantile, prob = p.lower)
med <- apply(imp$samples, 1, median)
upper <- apply(imp$samples, 1, quantile, prob = p.upper)
mu.lower <- exp(lower)
mu.upper <- exp(upper)
mu.med <- xts(exp(med), order.by = c(days, days.ahead), frequency = 7)
counts[days.ahead] <- mu.med[days.ahead]
mu <- xts(x = as.matrix(cbind(counts, mu.lower, mu.upper)), order.by = c(days,
    days.ahead))
p <- dygraph(mu, main = paste(mycounty, ": new case prediction (confidence ",
    100 * confidence, "%)", sep = ""), xlab = "Day", height = 400,
    width = 800)
p <- p %>% dySeries(c("mu.lower", "counts", "mu.upper"), label = "counts")
p <- p %>% dyLegend(show = "always", hideOnMouseOut = FALSE)
p <- p %>% dyShading(from = days.ahead[1], to = days.ahead[step.ahead],
    color = "#CCEBD6")
p <- p %>% dyEvent(days.ahead[1], "Prediction", labelLoc = "bottom")
```

```
p
```



Plot of $\alpha_{t|t-1}$, i.e the signal

```
signal <- xts(cbind(lower, med, upper), order.by = c(days, days.ahead),
    frequency = 7)
p <- dygraph(signal, main = paste(mycounty, ": new case signal (confidence ",
    100 * confidence, "%)", sep = ""), xlab = "Day", height = 400,
    width = 800) %>% dySeries(c("lower", "med", "upper")) %>%
    dyLegend(show = "always", hideOnMouseOut = FALSE)
p <- p %>% dyShading(from = days.ahead[1], to = days.ahead[step.ahead],
    color = "#CCEBD6")
p <- p %>% dyEvent(days.ahead[1], "Prediction", labelLoc = "bottom")
p
```



Plot of the $\beta_{t|t-1}$, i.e the estimated slope

```
imp <- importanceSSM(fit_poisson$model, type = "states", filtered = TRUE,
    nsim = 2000)
lower <- apply(imp$samples[, 2, ], 1, quantile, prob = p.lower)
med <- apply(imp$samples[, 2, ], 1, quantile, prob = 0.5)
upper <- apply(imp$samples[, 2, ], 1, quantile, prob = p.upper)
slope <- xts(cbind(lower, med, upper), order.by = c(days, days.ahead),
    frequency = 7)
p <- dygraph(slope, main = paste(mycounty, ": new case slope (confidence ",
    100 * confidence, "%)", sep = ""), xlab = "Day", height = 400,
    width = 800)
p <- p %>% dySeries(c("lower", "med", "upper"), label = "median")
p <- p %>% dyLegend(show = "always", hideOnMouseOut = FALSE)
p <- p %>% dyShading(from = days.ahead[1], to = days.ahead[step.ahead],
    color = "#CCEBD6")
p <- p %>% dyEvent(days.ahead[1], "Prediction", labelLoc = "bottom")
p
```