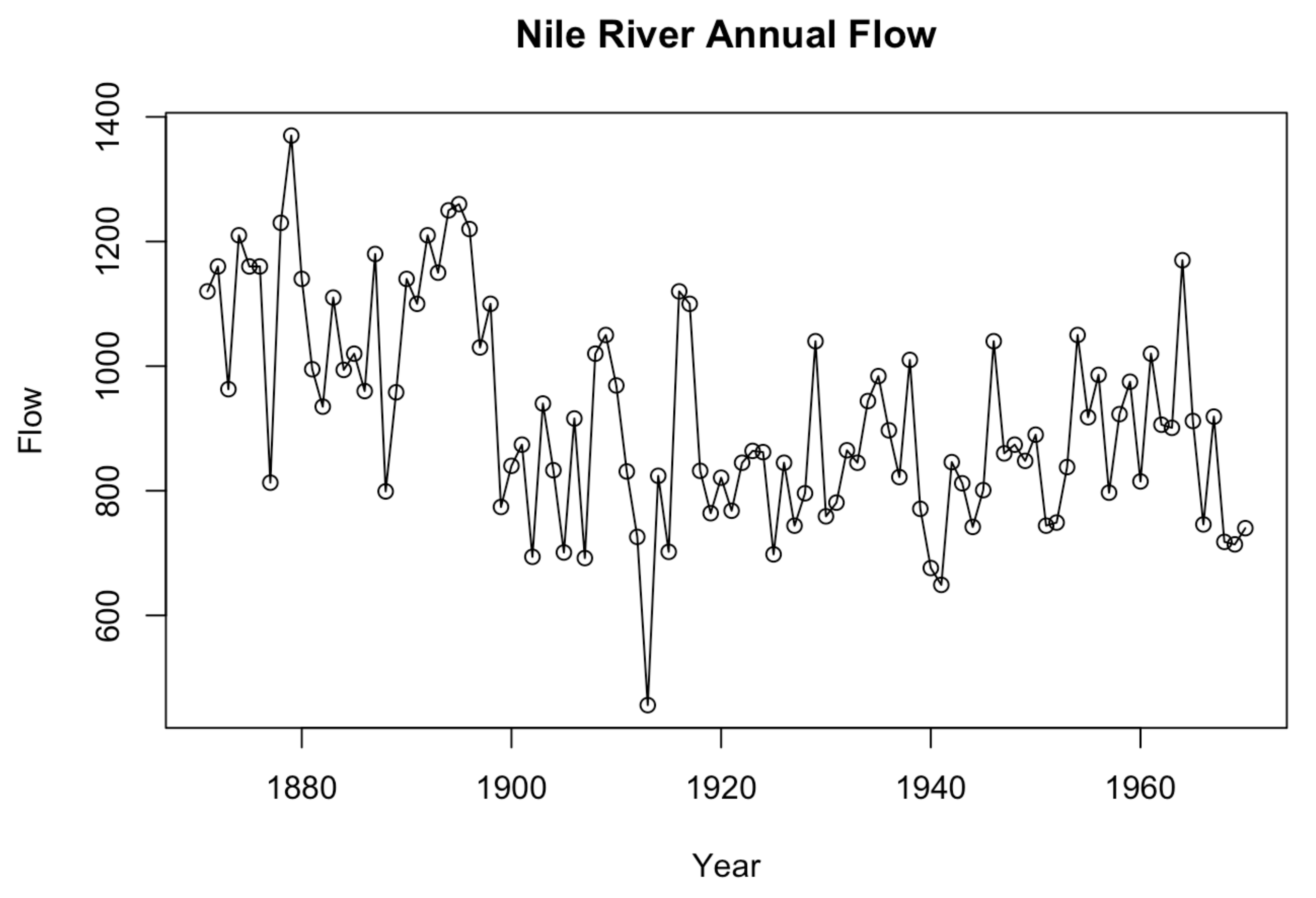
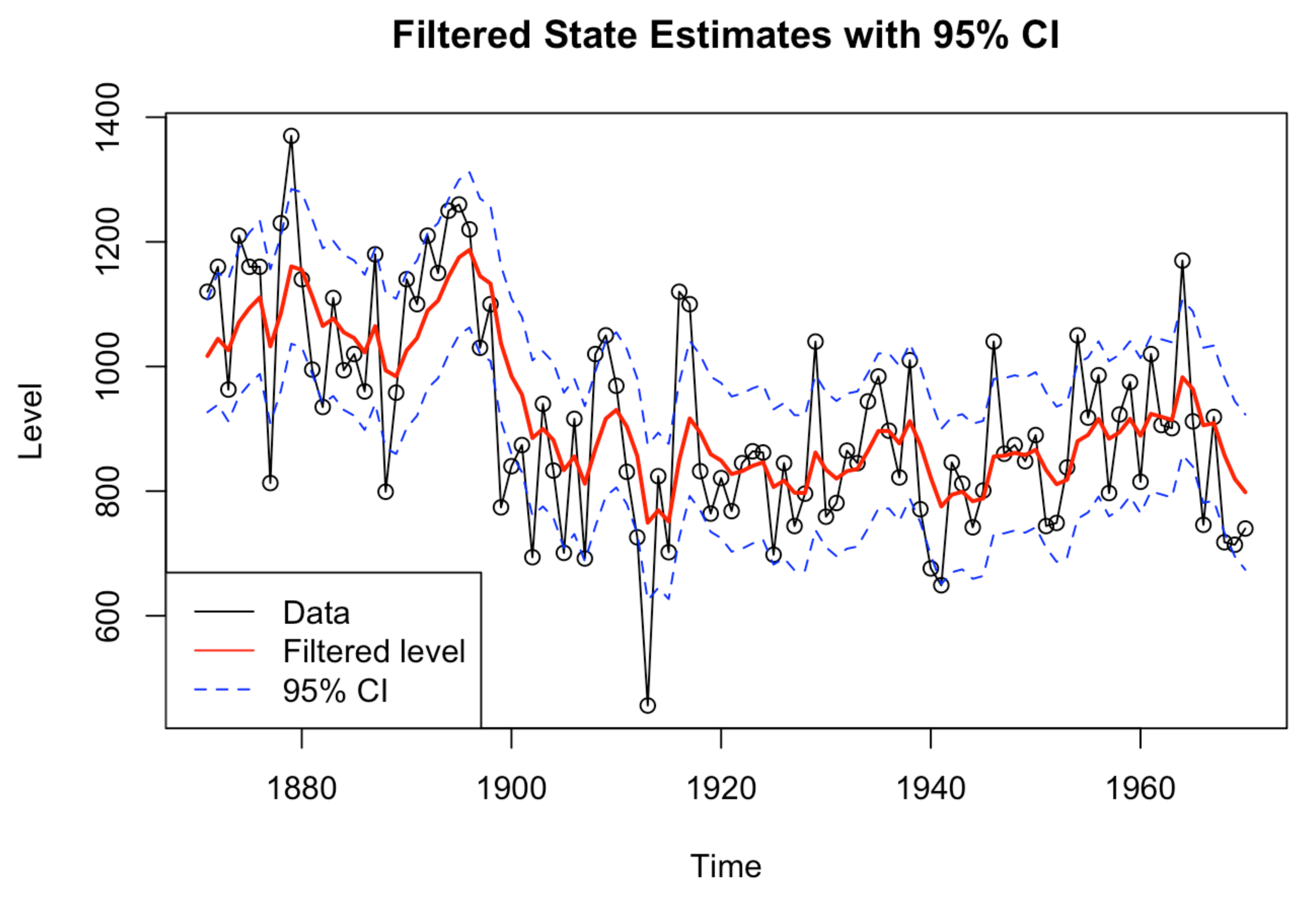
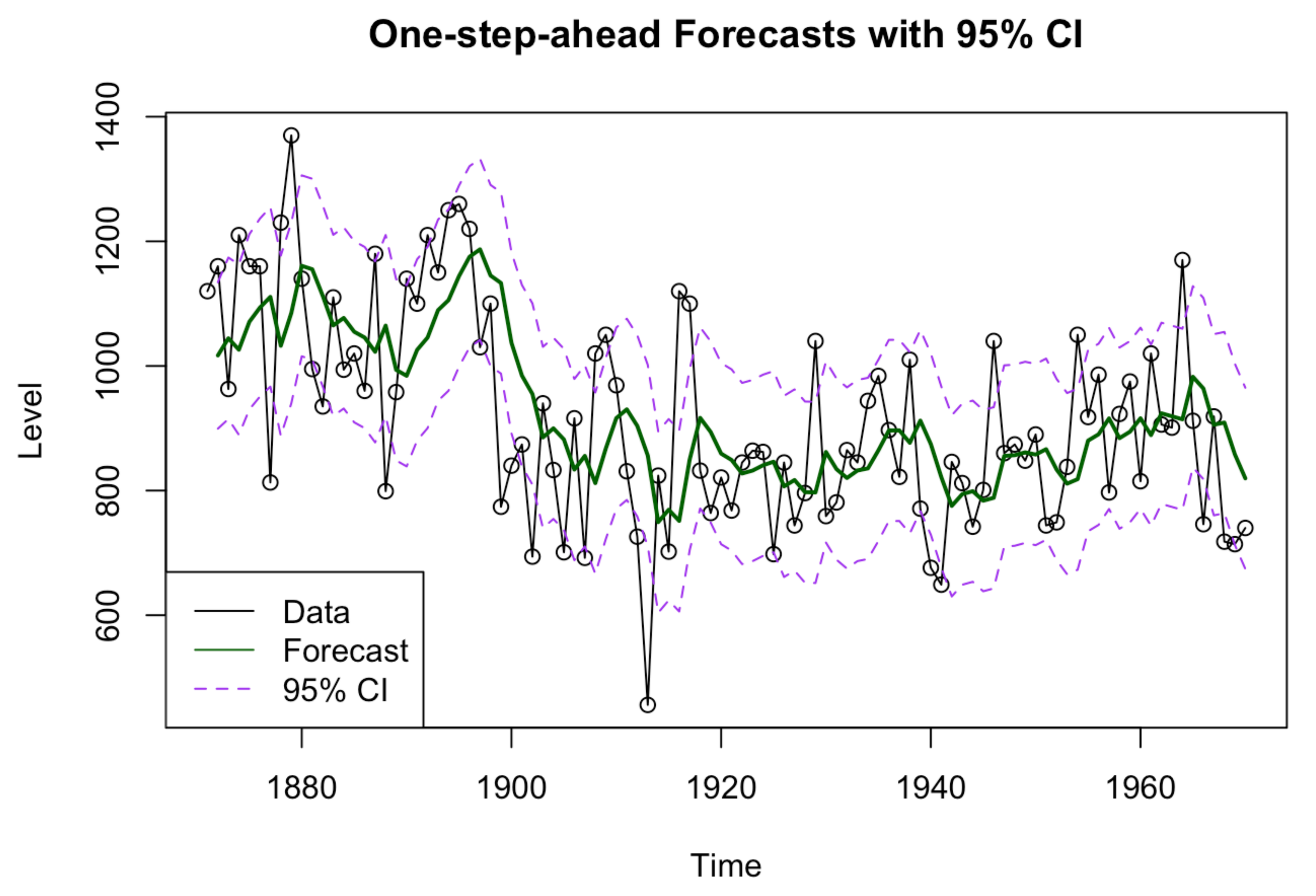
~~~
---
title: "Code TS4"
author: Stefano De Bianchi, Alessandro Caggia, Emanuele Cinti, Edoardo Calleri di
  Sala
date: "2025-04-15"
output: html_document
---



``` r
library(dlm)

data(Nile)
plot(Nile, type = "o", main = "Nile River Annual Flow", ylab = "Flow", xlab = "Year")
~~~



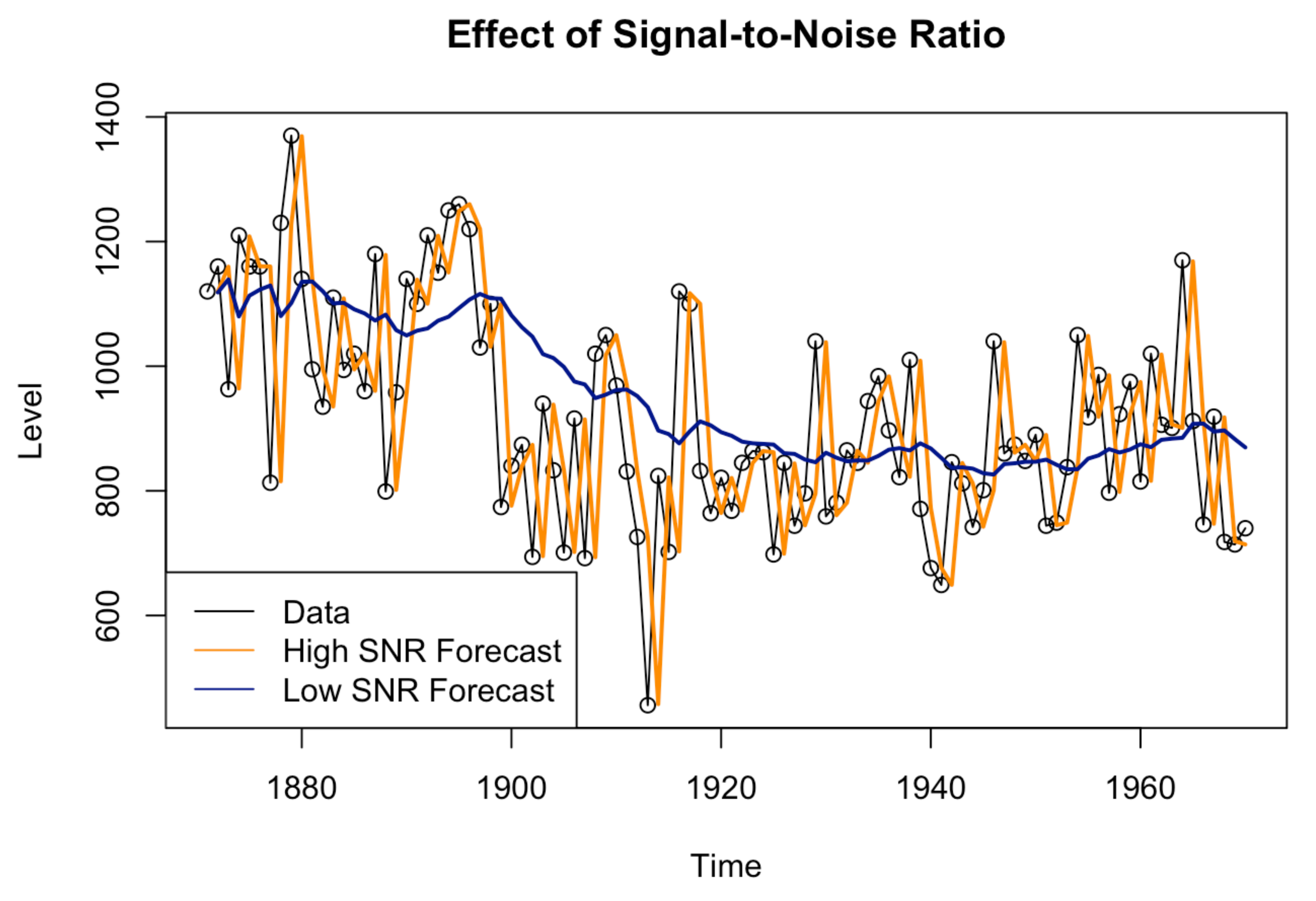

```r
mod <- dlm(m0 = 1000, C0 = 1000, FF = 1, V = 15100, GG = 1, W = 1470)

# 1. Filterring
outFilt <- dlmFilter(Nile, mod)
mt <- dropFirst(outFilt$m)

Ct_list <- dlmSvd2var(outFilt$U.C, outFilt$D.C)
sd_t <- sqrt(unlist(Ct_list))[-1]

years <- time(Nile)
upper <- mt + 1.96 * sd_t
lower <- mt - 1.96 * sd_t

plot(Nile, type = "o", main = "Filtered State Estimates with 95% CI", ylab = "Level")
lines(mt, col = "red", lwd = 2)
lines(upper, col = "blue", lty = 2)
lines(lower, col = "blue", lty = 2)
legend("bottomleft", legend = c("Data", "Filtered level", "95% CI"),
       col = c("black", "red", "blue"), lty = c(1, 1, 2))
```




```r
# 2. Online Forecasting
ft <- dropFirst(outFilt$f)
Q_list <- dlmSvd2var(outFilt$U.R, outFilt$D.R)
sqrtQ <- sqrt(unlist(Q_list))[-1]
upper_f <- ft + 1.96 * sqrtQ
lower_f <- ft - 1.96 * sqrtQ

plot(Nile, type = "o", main = "One-step-ahead Forecasts with 95% CI", ylab = "Level")
lines(ft, col = "darkgreen", lwd = 2)
lines(upper_f, col = "purple", lty = 2)
lines(lower_f, col = "purple", lty = 2)
legend("bottomleft", legend = c("Data", "Forecast", "95% CI"),
       col = c("black", "darkgreen", "purple"), lty = c(1, 1, 2))
```




```r
# 3. Effect of Signal-to-Noise Ratio
mod_high <- dlmModPoly(order = 1, dV = 100, dW = 15000)
filt_high <- dlmFilter(Nile, mod_high)

mod_low <- dlmModPoly(order = 1, dV = 15000, dW = 100)
filt_low <- dlmFilter(Nile, mod_low)

f_high <- dropFirst(filt_high$f)
f_low <- dropFirst(filt_low$f)

plot(Nile, type = "o", main = "Effect of Signal-to-Noise Ratio", ylab = "Level")
lines(f_high, col = "darkorange", lwd = 2)
lines(f_low, col = "darkblue", lwd = 2)
legend("bottomleft", legend = c("Data", "High SNR Forecast", "Low SNR Forecast"),
       col = c("black", "darkorange", "darkblue"), lty = c(1, 1, 1))
```



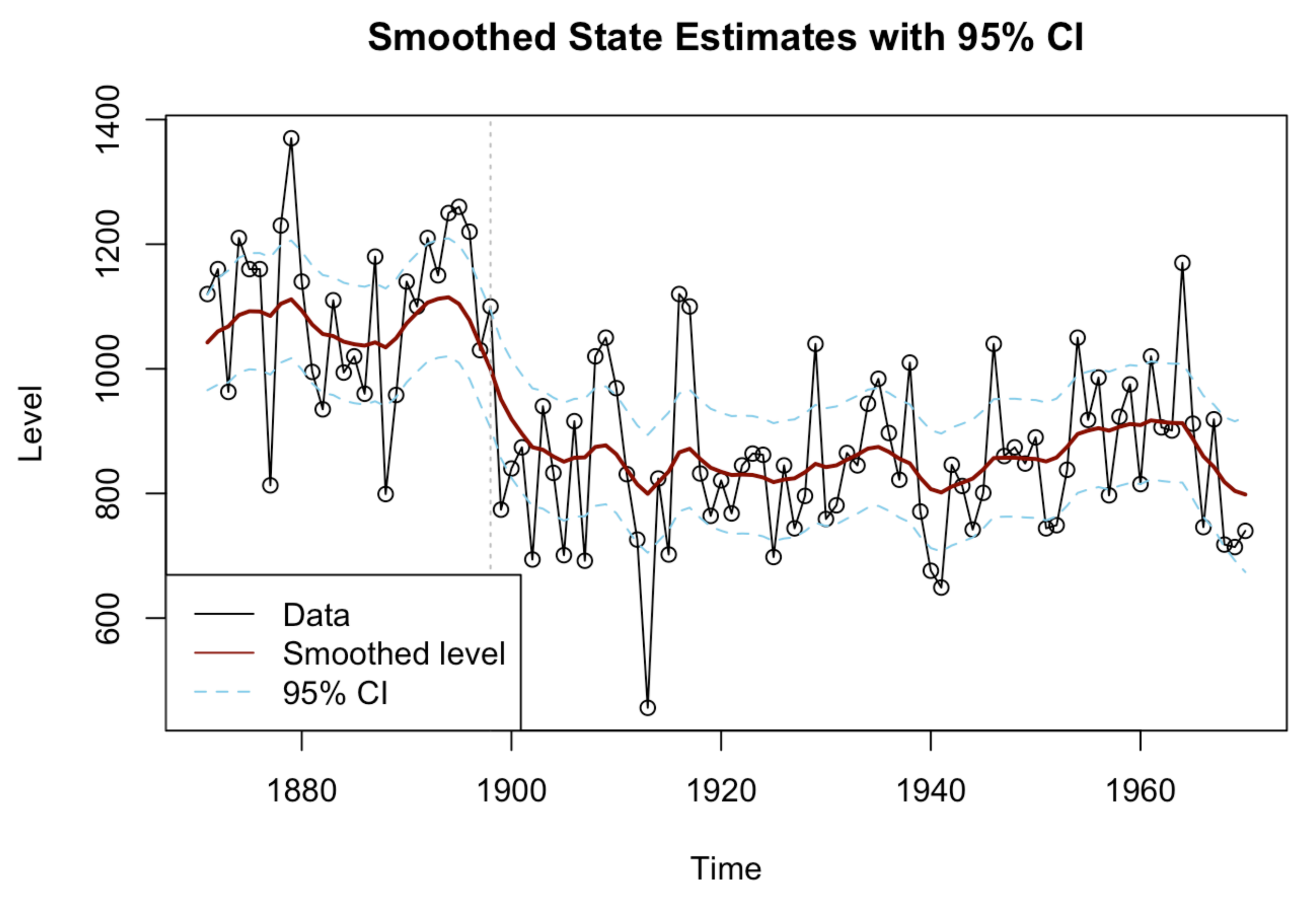

```r
#Answer: the signal to noise ratio vastly affects our one step ahead forecast. Indeed, consider the following two scenarios:
# in one case sigma-w is large and sigma-e is small and in the other it is the exact opposite.
# In the first scenario we have that the variation observed in the time series is mainly determined by the variat
ion of the latent state process.
# In the second scenario instead, the observed variation is mainly determined by measurement error.
# So, we say that in the first case we have a high signal-to-noise ratio while in the second case we have a lower one.
# All of this has crucial implications: in the second scenario, the new information (yt) is not crucially used to determine our forecast,
# as we deem that the variability is caused by noisy and unreliable measurements (high sigma-e).
# In the first scenario instead, we trust the new information (yt) to help estimate the movement in the latent pr
ocess.
# When more "trust" is given to the last information (high SNR), our forecast closely follows the observed data.
# When low "trust" is given to the last information (low SNR), our forecast reacts less to new observations.
# This theoretical reasoning is reflected in the behavior observed in the figure above.

# 4. Smoothing
outSmooth <- dlmSmooth(Nile, mod)
st <- dropFirst(outSmooth$s)

VarSmooth <- dlmSvd2var(outSmooth$U.S, outSmooth$D.S)
sd_smooth <- sqrt(unlist(VarSmooth))[-1]

t_target <- 28
smoothed_estimate_28 <- st[t_target]
smoothed_sd_28 <- sd_smooth[t_target]
ci_lower_28 <- smoothed_estimate_28 - 1.96 * smoothed_sd_28
ci_upper_28 <- smoothed_estimate_28 + 1.96 * smoothed_sd_28

cat("Smoothed estimate at t = 28:", round(smoothed_estimate_28, 2), "\n")
```

```
## Smoothed estimate at t = 28: 999.57
```

```r
cat("95% CI at t = 28: [", round(ci_lower_28, 2), ",", round(ci_upper_28, 2), "]\n")
```

```
## 95% CI at t = 28: [ 905.01 , 1094.13 ]
```

```r
upper_smooth <- st + 1.96 * sd_smooth
lower_smooth <- st - 1.96 * sd_smooth

plot(Nile, type = "o", main = "Smoothed State Estimates with 95% CI", ylab = "Level")
lines(st, col = "darkred", lwd = 2)
lines(upper_smooth, col = "skyblue", lty = 2)
lines(lower_smooth, col = "skyblue", lty = 2)
abline(v = time(Nile)[t_target], col = "gray", lty = 3)
legend("bottomleft", legend = c("Data", "Smoothed level", "95% CI"),
       col = c("black", "darkred", "skyblue"), lty = c(1, 1, 2))
```




```r
#Answer: I computed the smoothed estimate and the required CIs on the full sample.
#  Subsequently, I retrieved the posterior mean, the posterior variance and the CI of the observation at t = 28.
```