Two TMB Models with Matching Parameters: Linear Gaussian SSM and SIR

Overview

We build two likelihoods in **Template Model Builder** (TMB) and use them end-to-end:

1. **Linear Gaussian state-space (LGSS)**: latent AR(1) state with Gaussian observation noise.
2. **SIR epidemic model** (discrete-time with Poisson infection/recovery events and Poisson-thinned reporting).

For both models we (i) write C++ templates, (ii) simulate data in R with *the same parameterization*, (iii) compile and fit via Laplace approximation and MLE, and (iv) plot and summarize results.

Mathematical specification

LGSS model. For $t = 1, \ldots, T$ let the latent state x_t follow an AR(1) with $|\phi| < 1$ and innovation sd $\sigma > 0$, and observations y_t have noise sd $\tau > 0$:

$$x_1 \sim \mathcal{N}\left(0,\ \frac{\sigma^2}{1-\phi^2}\right), \quad x_t \mid x_{t-1} \sim \mathcal{N}(\phi x_{t-1}, \sigma^2), \quad y_t \mid x_t \sim \mathcal{N}(x_t, \tau^2).$$

The negative log-likelihood (NLL) integrates over the random effects $\mathbf{x} = (x_t)$ through Laplace approximation in TMB.

SIR model. We assume a fixed total population N. The model is parameterized by the transmission rate $\beta > 0$, recovery rate $\gamma > 0$, and reporting probability $\rho \in (0, 1)$. For $t = 1, \ldots, T-1$, the dynamics are:

$$\text{newInf}_t \sim \text{Pois}\big(\lambda_t^{(I)}\big), \quad \lambda_t^{(I)} = \beta\, \frac{S_t I_t}{N},$$

$$\text{newRec}_t \sim \text{Pois}\big(\lambda_t^{(R)}\big), \quad \lambda_t^{(R)} = \gamma\, I_t.$$

The state updates follow

$$\begin{aligned} S_{t+1} &= S_t - \text{newInf}_t, \\ I_{t+1} &= I_t + \text{newInf}_t - \text{newRec}_t, \qquad \text{with conservation } S_t + I_t + R_t = N. \\ R_{t+1} &= R_t + \text{newRec}_t, \end{aligned}$$

Finally, the observation model is

$$y_t \,\big|\, \text{newInf}_t \ \sim\ \text{Pois}\big(\rho\, \text{newInf}_t\big),$$

where y_t denotes reported cases.

In TMB, we treat $\mathbf{S} = (S_t)$ and $\mathbf{R} = (R_t)$ as random effects, while $\mathbf{I} = (I_t)$ is implied by the conservation law $I_t = N - S_t - R_t$.

Model 1: Linear Gaussian SSM (simulate, fit, plot)

Simulation with parameters matching the TMB template

We simulate with $\phi = 0.7$, $\sigma = 0.5$, $\tau = 1.0$ and $T = 100$.

```
T <- 100L
phi_true   <- 0.7
sigma_true <- 0.5
tau_true   <- 1.0

x_true <- numeric(T)
y_obs  <- numeric(T)
x_true[1] <- rnorm(1, mean = 0, sd = sigma_true/sqrt(1 - phi_true^2))
for(t in 2:T){
  x_true[t] <- rnorm(1, mean = phi_true * x_true[t-1], sd = sigma_true)
}
y_obs <- rnorm(T, mean = x_true, sd = tau_true)

c(head(x_true,3), "...", tail(x_true,3))

## [1] "0.4346166471439"    "0.322052354856236"  "0.612013890570848"
## [4] "..."                "-1.59052802490664"  "-1.52438420836364"
## [7] "-0.958710899926456"
```

Fit the LGSS likelihood in TMB

Random effects are the latent states $\mathbf{x}$.

```
# Data and initial parameters (near truth)
data_lgss <- list(y = y_obs)
par_lgss <- list(theta_phi = atanh(0.6),  # start near phi=0.6
                 log_sigma = log(0.6),
                 log_tau   = log(1.2),
                 x = rep(0, T))            # initialize states at 0

obj_lgss <- MakeADFun(data = data_lgss, parameters = par_lgss,
                      random = "x", DLL = "lgss")

## Error in MakeADFun(data = data_lgss, parameters = par_lgss, random = "x", : could
not find function "MakeADFun"

opt_lgss <- nlminb(start = obj_lgss$par, objective = obj_lgss$fn, gradient = obj_lgss$gr)
```

```
## Error in setNames(as.double(start), names(start)): object 'obj_lgss' not found
```

```
rep_lgss <- sdreport(obj_lgss)
```

```
## Error in sdreport(obj_lgss): could not find function "sdreport"
```

```
est_lgss <- summary(rep_lgss, "report")
```

```
## Error in summary(rep_lgss, "report"): object 'rep_lgss' not found
```

```
est_lgss
```

```
## Error in eval(expr, envir, enclos): object 'est_lgss' not found
```

LGSS: Parameter table (true vs. estimated)

```
lgss_tab <- data.frame(
  Parameter = c("phi","sigma","tau"),
  True      = c(phi_true, sigma_true, tau_true),
  Estimate  = est_lgss[ c("phi","sigma","tau"), 1],
  SE        = est_lgss[ c("phi","sigma","tau"), 2]
)
```

```
## Error in data.frame(Parameter = c("phi", "sigma", "tau"), True = c(phi_true, : object
'est_lgss' not found
```

```
print(lgss_tab, row.names = FALSE)
```

```
## Error in print(lgss_tab, row.names = FALSE): object 'lgss_tab' not found
```

LGSS: Plot of latent state and observations

```
plot(x_true, type="l", lwd=2, xlab="time t", ylab="value",
     main="LGSS: latent state and observations")
points(y_obs, pch=16, cex=0.6)
legend("topleft", bty="n", lwd=c(2,NA), pch=c(NA,16),
       legend=c("latent x_t","observations y_t"))
```

Model 2: SIR (simulate, fit, plot)

Simulation with matching parameters

We simulate a discrete-time SIR with Poisson events and Poisson-thinned reporting (same form used in the TMB template).

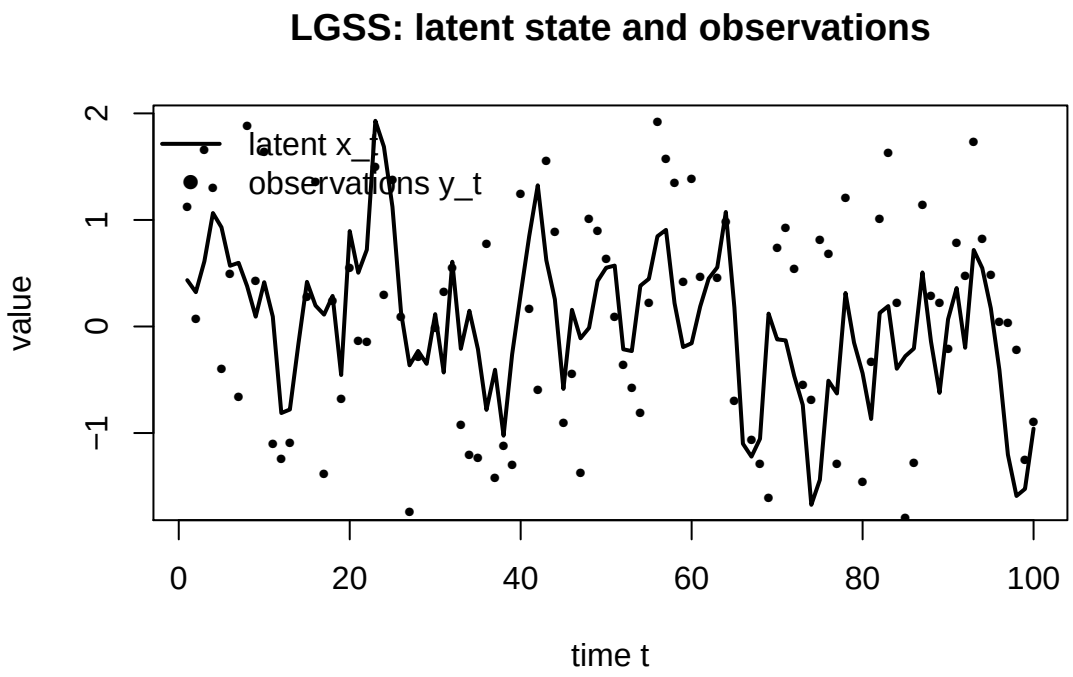

Figure 1: LGSS: latent state $x_t (solid) and observations y_t (points)$.

```r
sir_sim <- function(T, N, beta, gamma, rho, I0, R0=0L){
  S <- numeric(T); I <- numeric(T); R <- numeric(T); y <- numeric(T-1)
  S[1] <- N - I0 - R0; I[1] <- I0; R[1] <- R0
  for(t in 1:(T-1)){
    lambdaI <- beta * S[t] * I[t] / N
    lambdaR <- gamma * I[t]
    newI <- rpois(1, lambdaI)
    newR <- rpois(1, lambdaR)
    newI <- min(newI, S[t])                 # cannot infect more than susceptible
    newR <- min(newR, I[t] + newI)          # cannot recover more than infected next
    S[t+1] <- S[t] - newI
    I[t+1] <- I[t] + newI - newR
    R[t+1] <- R[t] + newR
    y[t]   <- rpois(1, rho * newI)          # reported infections
  }
  list(S=S, I=I, R=R, y=y)
}

# True parameters
T2 <- 100L
N_true <- 20000L
beta_true  <- 0.45
```

```r
gamma_true <- 0.20
rho_true   <- 0.35
I0_true    <- 40L
R0_true    <- 0L

sim <- sir_sim(T=T2, N=N_true, beta=beta_true, gamma=gamma_true,
               rho=rho_true, I0=I0_true, R0=R0_true)

c(head(sim$S,3), "...", tail(sim$S,3))

## [1] "19960" "19943" "19922" "..."   "2482"  "2482"  "2482"
```

Fit the SIR likelihood in TMB

Random effects are $\mathbf{S}, \mathbf{R}$; I_t is implied by $N - S_t - R_t$.

```r
data_sir <- list(y = as.numeric(sim$y), N = as.integer(N_true))
par_sir <- list(
  log_beta   = log(0.4),
  log_gamma  = log(0.25),
  logit_rho  = qlogis(0.3),
  S = as.numeric(sim$S),   # good starting path (speeds convergence)
  R = as.numeric(sim$R)
)

obj_sir <- MakeADFun(data = data_sir, parameters = par_sir,
                     random = c("S","R"), DLL = "sir")

## Error in MakeADFun(data = data_sir, parameters = par_sir, random = c("S", : could
not find function "MakeADFun"

opt_sir <- nlminb(start = obj_sir$par, objective = obj_sir$fn, gradient = obj_sir$gr)

## Error in setNames(as.double(start), names(start)): object 'obj_sir' not found

rep_sir <- sdreport(obj_sir)

## Error in sdreport(obj_sir): could not find function "sdreport"

est_sir <- summary(rep_sir, "report")

## Error in summary(rep_sir, "report"): object 'rep_sir' not found

est_sir

## Error in eval(expr, envir, enclos): object 'est_sir' not found
```

SIR: Parameter table (true vs. estimated)

```
sir_tab <- data.frame(
  Parameter = c("beta","gamma","rho"),
  True      = c(beta_true, gamma_true, rho_true),
  Estimate  = est_sir[ c("beta","gamma","rho"), 1],
  SE        = est_sir[ c("beta","gamma","rho"), 2]
)

## Error in data.frame(Parameter = c("beta", "gamma", "rho"), True = c(beta_true, :
object 'est_sir' not found

print(sir_tab, row.names = FALSE)

## Error in print(sir_tab, row.names = FALSE): object 'sir_tab' not found
```

SIR: Time series of S, I, R and reported cases

```
par(mar=c(4,4,2,1))
plot(sim$I, type="l", lwd=2, col=1, xlab="time t", ylab="count",
     main="SIR: states and reported cases")
lines(sim$S, lwd=1.5, lty=2)
lines(sim$R, lwd=1.5, lty=3)
bar_y <- sim$y
bar_y[bar_y<0] <- 0
barplot(height = bar_y, add=TRUE, border=NA, axes=FALSE)
legend("right", bty="n", lwd=c(2,1.5,1.5,NA), lty=c(1,2,3,NA), pch=c(NA,NA,NA,15),
       legend=c("I (infected)","S (susceptible)","R (removed)","reported cases"),
       pt.cex=1.2)
```

Interpretation

LGSS. The MLE and reported standard errors (delta/Laplace) should be close to the true values for a moderate T. The plot shows y_t noisy around the latent x_t with AR(1) persistence ϕ.

SIR. The likelihood ties reported cases to latent new infections through ρ. The random-effect trajectories (S_t, R_t) are integrated by Laplace, with I_t implied by conservation. Estimates of (β, γ, ρ) recover transmission, recovery, and reporting scales used in simulation.

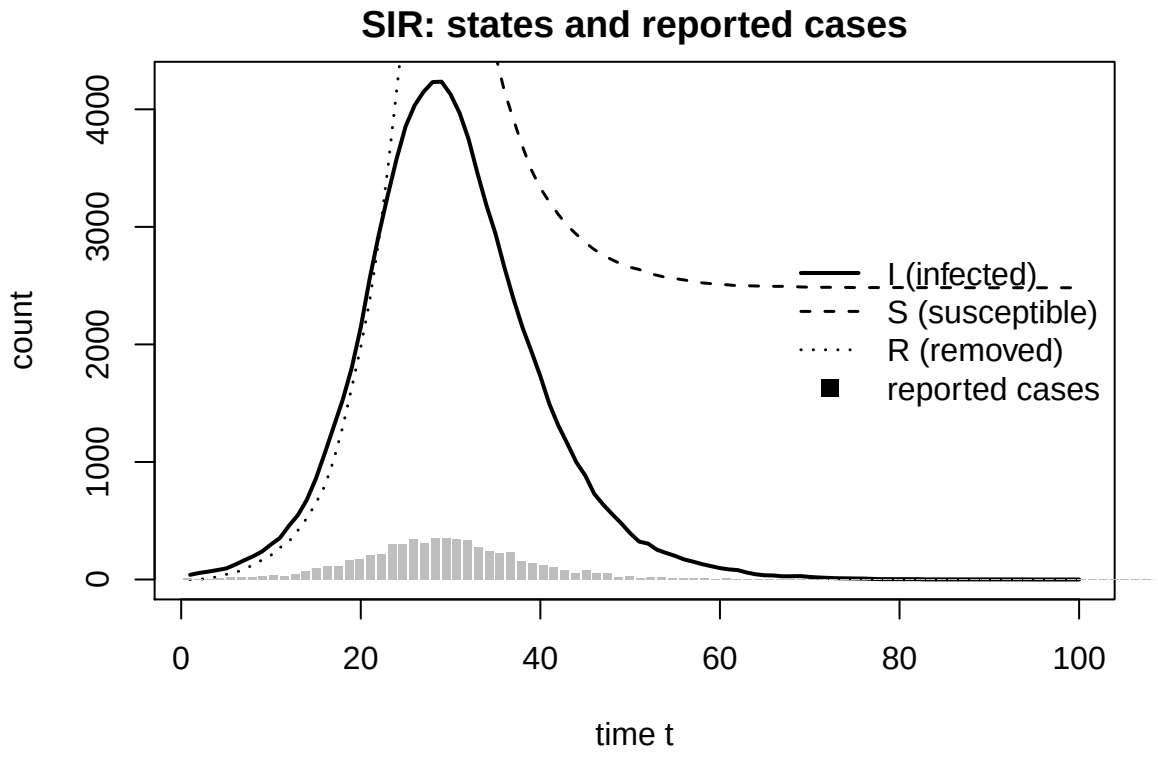

Figure 2: SIR: states and reported cases (bars).