Time Series Analysis
Assignment 4

Authors

Daniel Gonzalvez Alfert- s240404
Spiliopoulos Charalampos - s222948
Maria Kokali - s232486
Marios-Dimitrios Lianos - s233558

July 5, 2024

Contents

1 Plot the Data

1.1 Make plots of the data

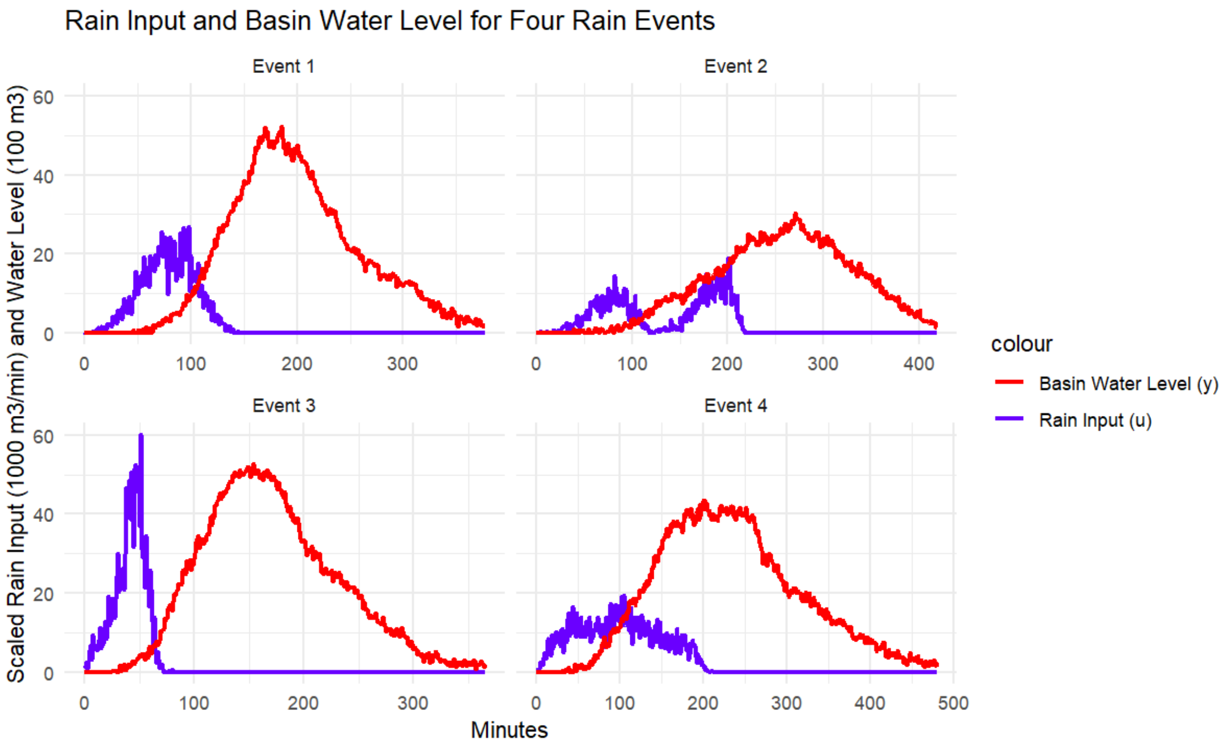

Figure 1: Initial Plots

1.2 Comment on the data. What relations are obvious? dynamics, variance, etc.

There seems to be a dynamic relationship between rain input and basin water level. In general, increases in rain input are followed by rises in water levels, and this response appears to have some delay.

For all events, peaks in rain input precede peaks in basin water level, implying that the basin's response to rain input is not immediate but occurs after a lag period. Each event shows different levels of intensity and duration in both rain input and water levels. Events 1 and 3 show higher peaks in rain input compared to Events 2 and 4.

To be more specific, Event 1 shows a single, smooth peak in both rainfall and water levels, Event 2 features a double-peaked rain input, resulting in a water level response where the basin's reaction to the second peak is subdued, Event 3 presents an intense, sharp rain peak followed by a significant water level rise, indicating the system's sensitivity to sudden heavy rainfall, and, lastly, Event 4 involves prolonged rain, causing a sustained increase in water level before a slow decline.

The variance in rain input and water level readings seems to be event-specific. Event 3, for example, shows a rapid increase and a sharp peak in rain input, which is mirrored by a steep rise in water level with a broader peak.

We can see that the effect previously mentioned holds for events 3 and 4, but not only that. We can see that the variance of the rain and the variance of water level in the basin is also correlated: the sharper the increase and decrease of rain, the sharper the increase and decrease of the basin water level.

2 Simulate using the model

2.1 Write the code to simulate the model and include it in Appendix 1, save both the simulated states and the output in each iteration

The code we used is presented in Appendix 1.

Define Parameters: We started by setting up the parameters (α, σ_1, and σ_2) with the middle values of their ranges.
Then we defined:

- **A**: How the state of the system (basin water level) changes over time without rain.
- **B**: Shows the effect of rain input on the water level.
- **C**: Translates the internal state of the system into what we actually observe as the water level.
- **System Noise Function**: The function G is defined to add variability to the simulation based on the current state.
- **Initial State vector**: The initial state vector (X)

2.2 For the first rain event simulate the Basin water level with the model. Use some values of the three parameters that you find appropriate

α	σ_1	σ_2
0.055	0.5005	2.55

Table 1: Initial values of the three parameters

The values of the three parameters that we have chosen are the middle values for each parameter.

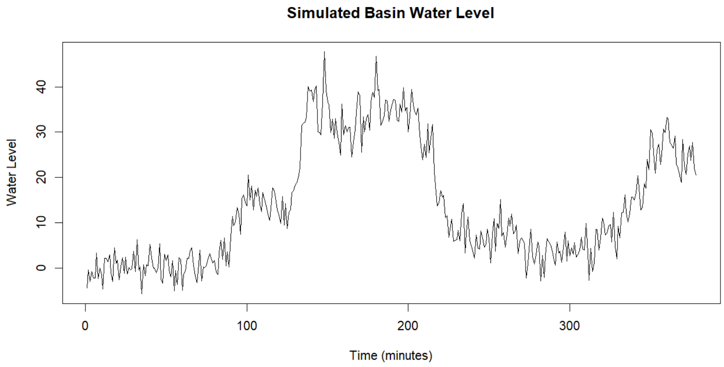

Figure 2: Simulation of event 1 with middle values

2.3 Comment on the plotted simulation explaining how the model works

In the plot, we can observe that the simulated water level starts at a baseline, gradually fluctuating with small amplitude as system noise affects the state. Upon the onset of rainfall (not directly shown in this plot but implied by water level changes), the model responds with an increase in water level, peaking around the middle of the simulation period. This peak corresponds to the culmination of rain impact modeled by matrix B.

After the peak, the water level declines, reflecting either a decrease in rainfall or the basin's mechanisms (like drainage or absorption) taking effect to reduce water levels back towards a baseline. Furthermore, after the primary peak and subsequent decline, the water level exhibits a secondary rise. This might suggest a delayed response or a secondary input of rain. However, it could also just be a peak randomly produced by the noise within the measurement (σ_2) which we have fitted to be quite big.

In addition, the overall variability and jagged appearance of the simulation demonstrate the influence of the noise function G together with both observation and measurement noise, adding realistic randomness that would be expected in natural hydrological processes.

2.4 Vary the three parameters one-by-one, and comment on how the parameter influences the system dynamics and variation

In the simulations we ran earlier, we started with middle values for each parameter as initial guesses. For each parameter, while keeping the other two constant, we tested both the lowest and highest values within their respective ranges.

Regarding the lowest simulation plot where we used two different values for the transition rate α, which controls how quickly the water level responds to changes, we can see that lower values of α (e.g., α=0.01) result in a more gradual increase and smoother curve, indicating a slower response to input changes. Higher values of α (e.g., α=0.1) make the system more sensitive and show a more rapid response, leading to sharper peaks, greater fluctuations and more immediate reactions to input changes.

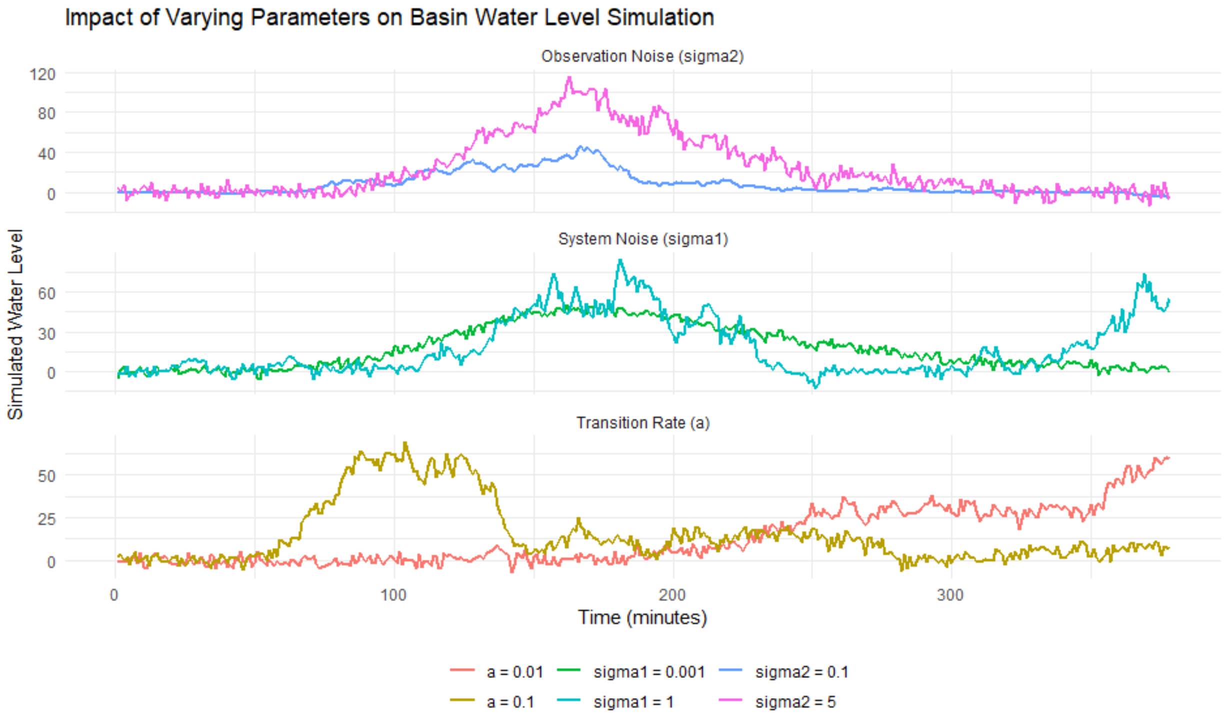

Figure 3: Simulation of event 1

In the middle simulation plot, we used two different values for the system's noise σ_1. Low σ_1 (e.g., σ_1=0.001) results in a relatively smooth simulation curve, indicating less (e.g., σ_1=1) leads to more erratic and jagged fluctuations. The increased σ_1 enhances the unpredictability of the response, making the system appear more dynamic but also less predictable.

Lastly, in the top simulation plot, we applied two different values for the observation noise σ_2. Low σ_2 (e.g., σ_2=0.1) provides a clearer, smoother and more consistent depiction of the water level response by sacrificing the natural variability of the relationship between rain and water level, while high σ_2 (e.g., σ_2=5) results in a plot that is noisier and more spread, which conceals the true water level behavior.

3 Simulate to find best starting values for estimation

3.1 Change the parameters until you find the parameter values (same set for all events) where the simulations seem to capture the dynamics and variation on the bassin water level

In order to try all the possible combinations we would have to look into an immense amount of combination. Thus, for the purpose of this assignment we did 5 constant splits of the parameter ranges and focused on the possible combination between those to find the best solution for all the events.

	1	2	3	4	5
α	0.01	0.0325	0.055	0.0775	0.1
σ_1	0.001	0.25075	0.5015	0.75225	1
σ_2	0.1	1.325	2.55	3.775	5

Table 2: 5-step partition of the 3 parameter ranges

After using suitable code that plots all the possible combinations of the values in Table 2, we concluded that some good parameters for our model are the following:

	α	σ_1	σ_2
Set1	0.0325	0.001	2.55
Set2	0.055	0.001	2.55
Set3	0.0775	0.001	2.55
Set4	0.1	0.001	2.55

Table 3

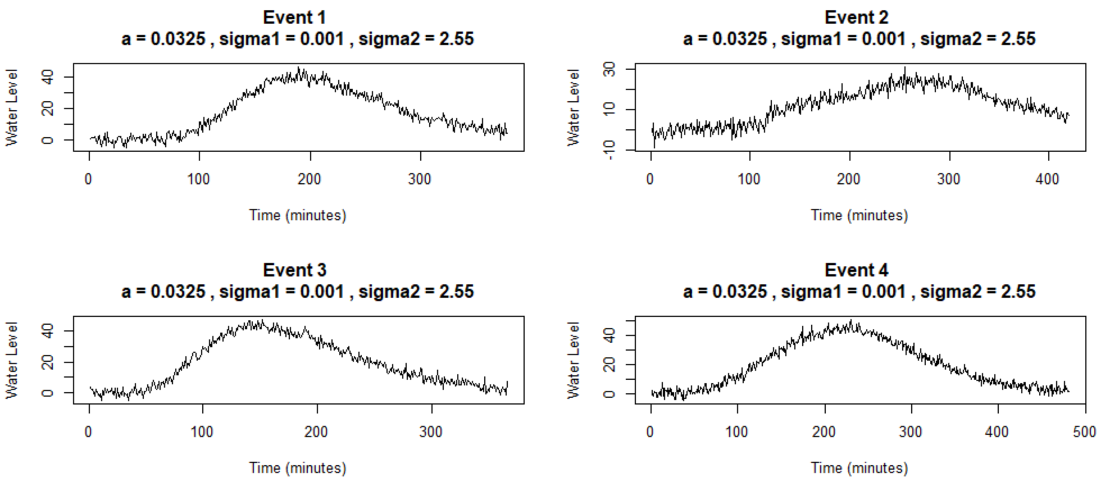

Figure 4: Set 1

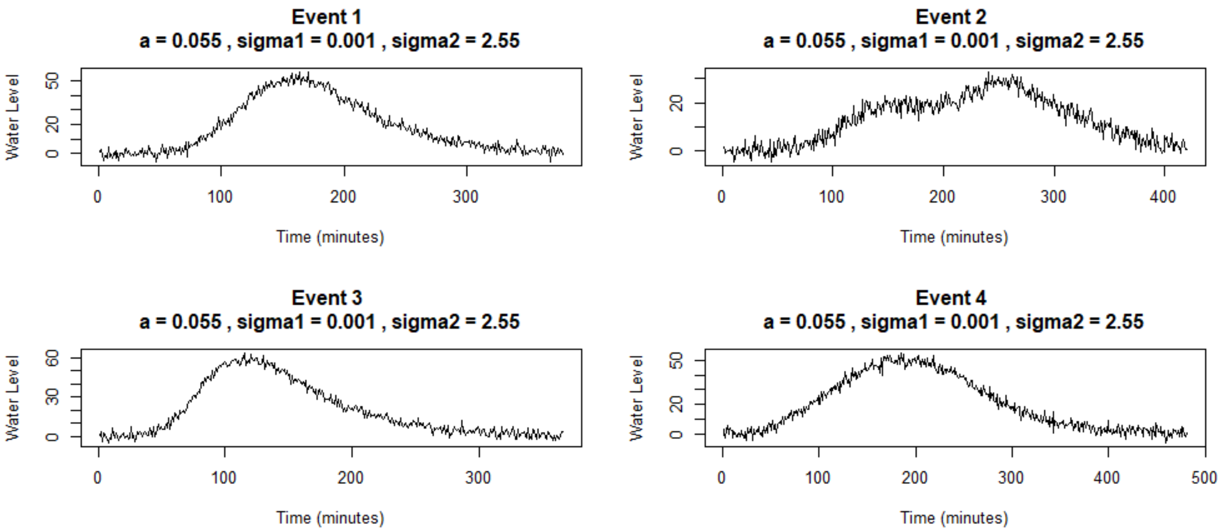

Figure 5: Set 2

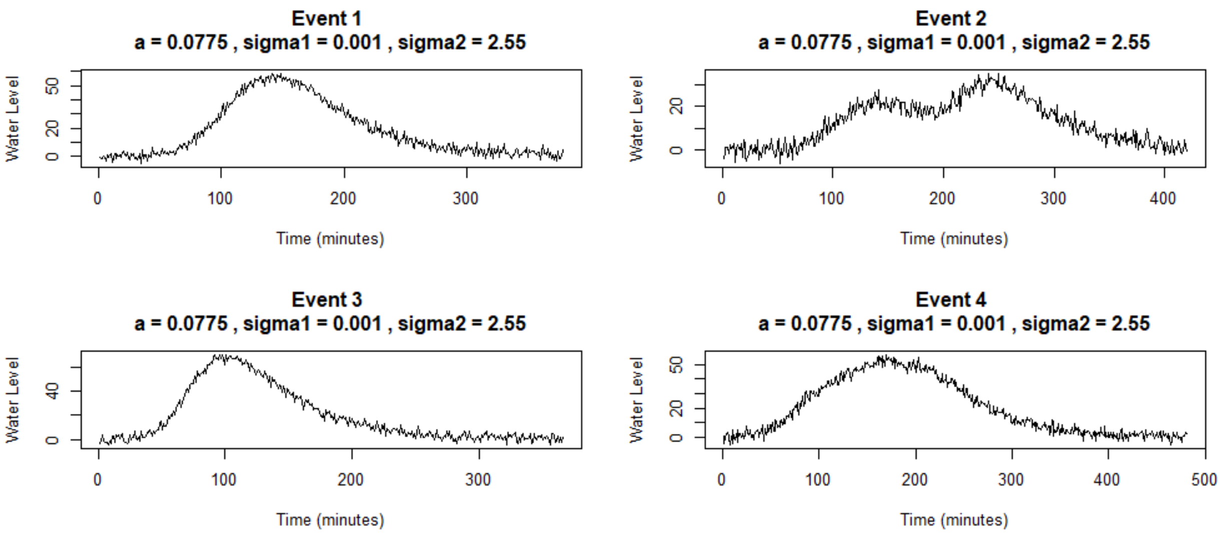

Figure 6: Set 3

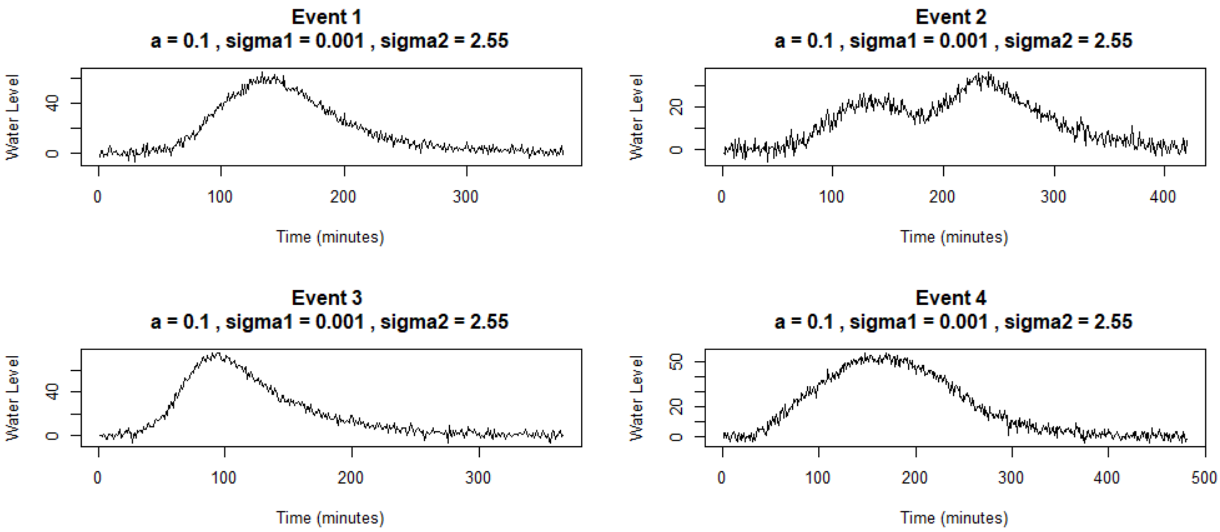

Figure 7: Set 4

Between those the one that seemed to perform the best for all the events it is set 1. In the following section, we are going to compare and showcase the possible candidates for the optimal set of parameters we have chosen.

3.2 Include the plots of the simulations and measurements, and comment on your considerations leading you to decide the parameter values.

We have selected the first set because of several reasons:

- The only change between the sets is the parameter α. This parameter controls the speed of the response of the water level variable. We feel like $\alpha = 0.0325$ represents a realistic speed.
- A system noise of $\sigma_1 = 0.001$ seems adequate for a highly deterministic process which is water being poured into a container.
- An observation noise of $\sigma_2 = 2.55$ seems enough to model the randomness within the measurement without ruining the actual simulation of the process.

4 Maximum likelihood estimation with the Kalman filter

4.1 Model Simulation and Optimization Procedure

1. **Initialization:**
   - Set the number of observations (n) and initial state vector (X_0).
   - Initialize state transition (A), input control (B), and output mapping (C) matrices.
   - Define initial noise variances Σ_1 and Σ_2.
2. **Simulation:**
   - Perform simulations using initial parameters to generate water levels, incorporating input data and noise adjustments.
3. **Log-Likelihood Calculation:**
   - Implement a function to compute the negative log-likelihood based on the deviation of simulated results from actual data.
4. **Parameter Optimization:**
   - Utilize the `optim` function to adjust parameters (a, σ_1, σ_2) aiming to minimize the log-likelihood.
5. **Optimized Simulation:**
   - Update simulation with optimized parameters.

13 , 14

4.2 Calculate the likelihood for the Rain Event 1

Optimal Parameters (KF)	α	σ_1	σ_2
Rain Event 1	0.038548779	0.001041123	0.575587521

Table 4: Optimal Parameters Rain Event 1

Rain Event 1	Optimal	Upper bound	Lower bound
Likelihood	0	3719.275	1377.634

Table 5: Likelihood for Rain Event 1

The analysis of the Kalman Filter implementation for Rain Event 1 reveals that the optimal parameters produced a likelihood of zero, indicating potential underfitting or computational issues, while both the upper and lower parameter bounds yielded significantly higher likelihoods, suggesting that adjustments toward these limits enhance model fit. This variance emphasizes the model's sensitivity to parameter changes.

4.3 Estimate the parameters for each of the rain events

Optimal Parameters (KF)	α	σ_1	σ_2
Rain Event 2	0.03237683	0.00104345	1.77005239
Rain Event 3	0.032707866	0.001267729	0.599172276
Rain Event 4	0.033225360	0.001278316	0.546376584

Table 6: Optimal Parameters Rain Event 2

The estimated parameters for Rain Events 2, 3, and 4 using the Kalman Filter show consistency in the transition rate (α) and system noise (σ_1), suggesting stable hydrological dynamics across these events. The α values are tightly grouped around 0.033, indicating a similar responsiveness of the water levels to rain inputs, which is reasonable given similar environmental settings. Whereas, for Rain Event 1, the parameter estimates reveal a slightly higher transition rate α . σ_1 values are also close in all rain events, reflecting uniform internal system noise. However, observation noise (σ_2) varies significantly, with Event 2 showing much higher noise (1.77005239) compared to Events 1, 3 and 4. This variation might indicate different measurement conditions or environmental disturbances during Event 2.

Overall, the parameter estimates are reasonable, capturing consistent internal dynamics with variable external influences across the events, making them suitable for modeling these hydrological responses accurately.

4.4 Do a model validation by simulating with the estimated parameters for each event

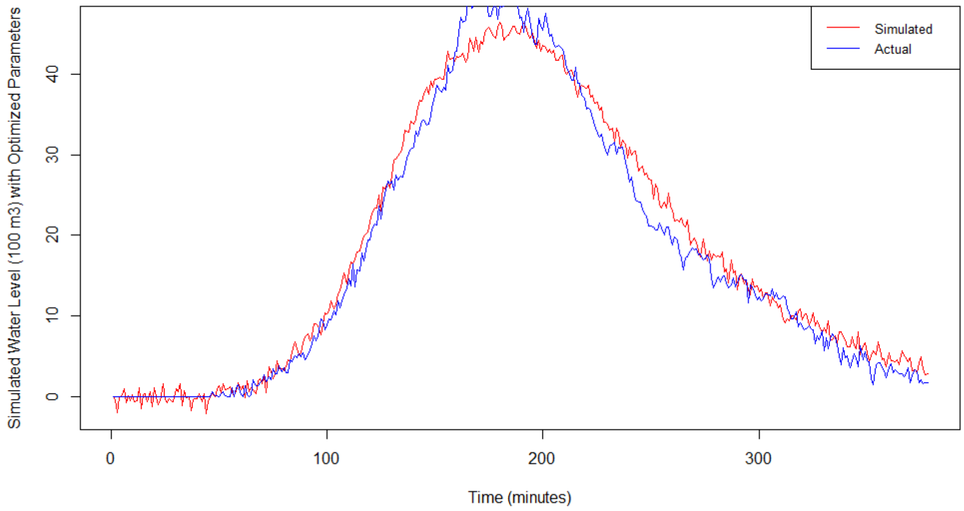

Figure 8: Simulation of Rain Event 1

Alignment and Timing: Both the simulated and actual data reach their peaks around the same time, which suggests that the model effectively captures the timing of the peak water response to the rainfall event. The increase to and decrease from the peak are symmetrical in both datasets, indicating that the model's input and decay dynamics are well-calibrated to this specific event's nature.

Amplitude Accuracy: The simulated data slightly underestimates the peak water level compared to the actual data.

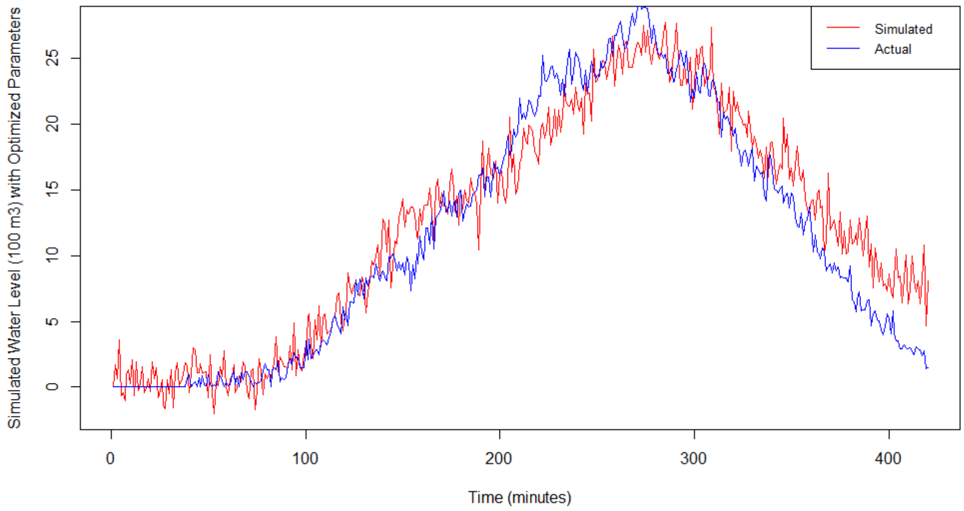

Figure 9: Simulation of Rain Event 2

Alignment and Timing: The simulated water levels rise and peak in very close alignment with the actual data, indicating that the model captures the key dynamics of the water input and output processes accurately. The peak values are nearly identical, suggesting that the maximum water level response to the rainfall input is well-modeled.

Amplitude Accuracy: Both curves decline in sync, but there are minor discrepancies in the rate of decline, with the simulated data occasionally lagging slightly behind the actual data in responsiveness to decreasing water input.

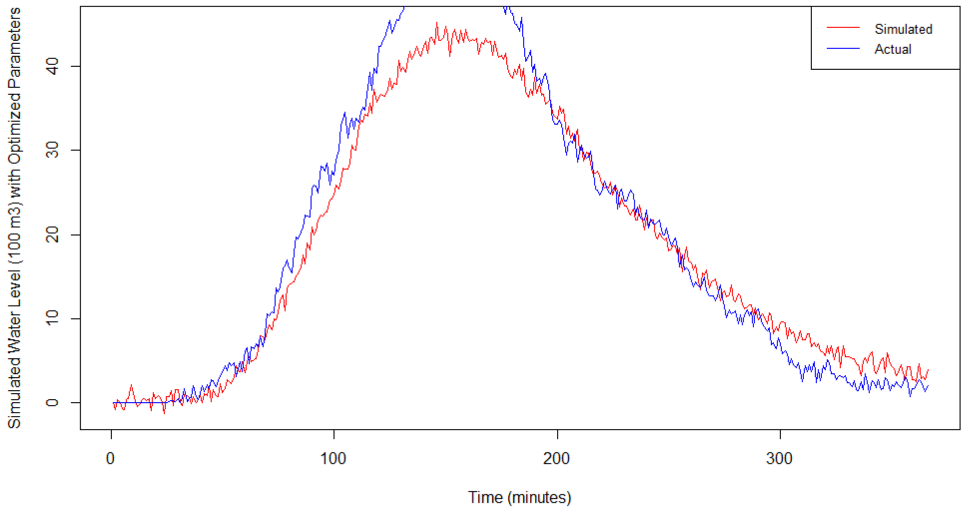

Figure 10: Simulation of Rain Event 3

Alignment and Timing: Both simulated and actual data reach their peak almost concurrently, which indicates that the timing of the peak water response is accurately modeled. The simulated data closely tracks the actual data throughout the event, suggesting that the model's dynamics are well-aligned with the natural behavior of the system.

Amplitude Accuracy: The peak of the simulated water levels is noticeably lower than that of the actual data.

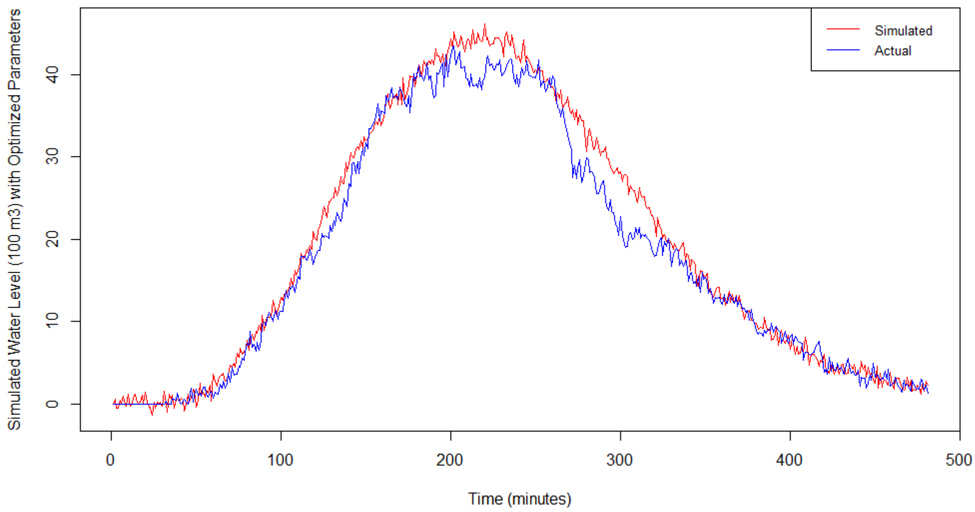

Figure 11: Simulation of Rain Event 4

Alignment and Timing: Both the simulated and actual data reach their peaks around the same time, showcasing that the model's timing for peak water levels is well-calibrated. The simulation captures the prolonged nature of the event effectively.

Amplitude Accuracy: The peak amplitudes are very close, though the simulated data occasionally shows slightly higher peaks. After the peak, both lines exhibit similar fluctuations.

List of Figures

List of Tables

5 Appendix1

```r
## 2) #################################################################################

parameters
a <- 0.05          # Transition rate
sigma1 <- 0.1      # Standard deviation of system noise for each state
sigma2 <- 1        # Standard deviation of observation noise

# Transition matrix A
A <- matrix(c(1-a, 0,   0,   0,
              a, 1-a, 0,   0,
              0,   a, 1-a, 0,
              0,   0,   a, 0.98), byrow=TRUE, nrow=4)

# Input matrix B
B <- matrix(c(1, 0, 0, 0), ncol=1)
# Output matrix C
C <- matrix(c(0, 0, 0, 1), nrow=1)
# State-dependent system noise function
G <- function(X) {
  diag(sapply(X, function(x) sqrt(abs(x))))
}

ut <- data1$u

# Initial state vector
X <- c(0, 0, 0, 0)
# Number of time points
n <- nrow(data1)
# Vectors to save the output and states
y <- numeric(n)

states <- matrix(NA, nrow=n, ncol=4)

# Simulation loop
for (i in 1:n) {
  # System noise
  e1 <- mvrnorm(1, mu = rep(0, 4), Sigma = (sigma1^2) * diag(4))
  # State update
  X <- A %*% X + B * ut[i] + G(X) %*% e1
  # Save states
  states[i,] <- X
  # Observation with noise
  y[i] <- C %*% X + rnorm(1, mean = 0, sd = sigma2)
}

# plot of the results
plot(y, type = 'l', main = "Simulated Basin Water Level", xlab = "Time (minutes)", ylab = "Water Level")
```

Figure 12

6 Appendix2

```r
# Define parameters
n <- nrow(data1)

# Model matrices initialization
A <- matrix(c(1-0.07, 0, 0, 0,
              0.07, 1-0.07, 0, 0,
              0, 0.07, 1-0.07, 0,
              0, 0, 0.07, 0.98), nrow=4, byrow=TRUE)
B <- matrix(c(1, 0, 0, 0), ncol=1)
C <- matrix(c(0, 0, 0, 1), nrow=1)
Sigma1 <- diag(rep(0.001, 4))
Sigma2 <- 1.8^2
X0 <- matrix(c(0, 0, 0, 0), ncol=1)

# Data vectors
u <- data1$u
y <- data1$y

# Simulation with corrected state-dependent noise
X <- X0
y_simulated <- numeric(n)
for (i in 2:n) {
    # State-dependent scaling matrix for the noise
    G <- diag(abs(X[,1]))  # Scaling factor based on the absolute values of the state
    noise <- mvrnorm(1, mu = rep(0, 4), Sigma = G * Sigma1)  # Generate scaled noise
    X <- A %*% X + B * u[i-1] + noise
    y_simulated[i] <- as.numeric(C %*% X + sqrt(Sigma2) * rnorm(1))
}

#  implementation considering the adjusted noise handling
negloglik <- function(prm){
    # Parameters dynamically updated within the function
    A[1, 1] <- A[2, 2] <- A[3, 3] <- 1 - prm["a"]
    A[2, 1] <- A[3, 2] <- A[4, 3] <- prm["a"]
    Sigma1_mod <- diag(rep(prm["sigma1"], 4))  # Adjusted noise level

    X <- X0
    SigmaX <- diag(rep(1000, 4))
    lik <- rep(NA, n)

    for (i in 1:n) {
        G <- diag(abs(X[,1]))  # State-dependent noise scaling
        Sigma1_scaled <- G * Sigma1_mod * G
        X_pred <- A %*% X + B * u[i]
        SigmaX_pred <- A %*% SigmaX %*% t(A) + Sigma1_scaled
        SigmaY <- C %*% SigmaX_pred %*% t(C) + prm["sigma2"]^2
        lik[i] <- dnorm(y[i], mean = as.numeric(C %*% X_pred), sd = sqrt(SigmaY))
        K <- SigmaX_pred %*% t(C) %*% solve(SigmaY)
        X <- X_pred + K %*% (y[i] - C %*% X_pred)
        SigmaX <- SigmaX_pred - K %*% C %*% SigmaX_pred
    }

    return(-sum(log(lik), na.rm = TRUE))
}
```

Figure 13

```r
# Calculate likelihoods at optimized, lower, and upper bound parameters
optimized_params <- c(a = fit$par["a"], sigma1 = fit$par["sigma1"], sigma2 = fit$par["sigma2"])
lower_bound_params <- c(a = 0.01, sigma1 = 0.0001, sigma2 = 0.1)
upper_bound_params <- c(a = 0.1, sigma1 = 1, sigma2 = 5)

likelihood_optimized <- negloglik(optimized_params)
likelihood_lower_bound <- negloglik(lower_bound_params)
likelihood_upper_bound <- negloglik(upper_bound_params)

# Print the calculated likelihoods
print(paste("Likelihood at optimized parameters:", likelihood_optimized))
print(paste("Likelihood at lower bound parameters:", likelihood_lower_bound))
print(paste("Likelihood at upper bound parameters:", likelihood_upper_bound))


# Parameter optimization call
fit <- optim(c(a = 0.07, sigma1 = 0.001, sigma2 = 1.8), negloglik, method = "L-BFGS-B", lower = c(0.01, 0.0001, 0.1), upper = c(0.1, 1, 5))

##plot actual with simulated data
print("Optimized Parameters:")
print(fit$par)

# Update parameters based on optimization results
optimized_a = fit$par["a"]
optimized_sigma1 = fit$par["sigma1"]
optimized_sigma2 = fit$par["sigma2"]

# Update model matrices with optimized parameters
A[1, 1] <- A[2, 2] <- A[3, 3] <- 1 - optimized_a
A[2, 1] <- A[3, 2] <- A[4, 3] <- optimized_a
Sigma1_opt <- diag(rep(optimized_sigma1, 4))

# Rerun simulation with optimized parameters
X <- X0
y_optimized <- numeric(n)
for (i in 2:n) {
    G <- diag(abs(X[,1]))  # State-dependent scaling factor
    noise <- mvrnorm(1, mu = rep(0, 4), Sigma = G * Sigma1_opt)  # Generate scaled noise
    X <- A %*% X + B * u[i-1] + noise
    y_optimized[i] <- as.numeric(C %*% X + sqrt(optimized_sigma2) * rnorm(1))
}

# Plot the simulated water levels with optimized parameters
plot(y_optimized, type = "l", col = "red", xlab = "Time (minutes)", ylab = "Simulated Water Level (100 m3) with Optimized Parameters")
lines(y, col = "blue")  # Optionally plot the actual data for comparison
legend("topright", legend=c("Simulated", "Actual"), col=c("red", "blue"), lty=1, cex=0.8)
```

Figure 14