DTU02417, Time Series Analysis, Assignment 4
State Space Model and Kalman Filter

Edward J. Xu (Jie Xu), s181238, DTU Management, edxu96@outlook.com

May 16th, 2019

Question 4.1

We consider the following system of differential equations to describe the thermal dynamics of a room:

$$\frac{dT_i}{dt} = \frac{1}{R_{im}C_i}(T_m - T_i) + \frac{1}{R_{ia}C_i}(T_a - T_i) + \frac{1}{C_i}\Phi_h + \frac{1}{C_i}G_v \tag{1}$$

$$\frac{dT_m}{dt} = \frac{1}{R_{im}C_m}(T_i - T_m) \tag{2}$$

Consider the indoor air and wall temperature as model states. Outdoor temperature, heating power and solar radiation serve as model inputs. Then this system can be transferred to a discrete time state space model with system equation:

$$\begin{pmatrix} T_i^t \\ T_m^t \end{pmatrix} = A \cdot \begin{pmatrix} T_i^{t-1} \\ T_m^{t-1} \end{pmatrix} + B \cdot \begin{pmatrix} T_a^{t-1} \\ \Phi_h^{t-1} \\ G_v^{t-1} \end{pmatrix} + \begin{pmatrix} \eta_1^t \\ \eta_2^t \end{pmatrix} \tag{3}$$

Continuous system equation can be written as:

$$\begin{pmatrix} T_i' \\ T_m' \end{pmatrix} = \begin{pmatrix} -\frac{(R_{im}+R_{ia})}{R_{im}R_{ia}C_i} & \frac{1}{R_{im}C_i} \\ \frac{1}{R_{im}C_m} & -\frac{1}{R_{im}C_m} \end{pmatrix} \begin{pmatrix} T_i \\ T_m \end{pmatrix} + \begin{pmatrix} -\frac{1}{R_{ia}C_i} & \frac{1}{C_i} & \frac{1}{C_i} \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} T_a^{t-1} \\ \Phi_h^{t-1} \\ G_v^{t-1} \end{pmatrix} + \begin{pmatrix} \eta_1^t \\ \eta_2^t \end{pmatrix} \tag{4}$$

Use a constant time step Δt and approximate the derivatives by:

$$\frac{dT}{dt} = \frac{T^t - T^{t-1}}{\Delta t} \tag{5}$$

We can transfer eq.1 and eq.2 to:

$$\frac{T_i^t - T_i^{t-1}}{\Delta t} = \frac{1}{R_{im}C_i}\left(T_m^{t-1} - T_i^{t-1}\right) + \frac{1}{R_{ia}C_i}\left(T_a^{t-1} - T_i^{t-1}\right) + \frac{1}{C_i}\Phi_h^{t-1} + \frac{1}{C_i}G_v^{t-1} \tag{6}$$

$$\frac{T_m^t - T_m^{t-1}}{\Delta t} = \frac{1}{R_{im}C_m}\left(T_i^{t-1} - T_m^{t-1}\right) \tag{7}$$

Simplify the above equations:

$$T_i^t = \frac{R_{ia}R_{im}C_i - \Delta t R_{ia} - \Delta t R_{im}}{R_{ia}R_{im}C_i} T_i^{t-1} + \frac{\Delta t R_{ia}}{R_{ia}R_{im}C_i} T_m^{t-1} + \frac{\Delta t R_{im}}{R_{ia}R_{im}C_i} T_a^{t-1} + \frac{\Delta t R_{im}R_{ia}}{R_{ia}R_{im}C_i}\Phi_h^{t-1} + \frac{\Delta t R_{im}R_{ia}}{R_{ia}R_{im}C_i} G_v^{t-1} \tag{8}$$

$$T_m^t = \frac{\Delta t}{R_{im}C_m} T_i^{t-1} + \frac{R_{im}C_m - \Delta t}{R_{im}C_m} T_m^{t-1} \tag{9}$$

Then, we can get **A** and **B**:

$$\mathbf{A} = \begin{pmatrix} \frac{R_{ia}R_{im}C_i - \Delta t R_{ia} - \Delta t R_{im}}{R_{ia}R_{im}C_i} & \frac{\Delta t R_{ia}}{R_{ia}R_{im}C_i} \\ \frac{\Delta t}{R_{im}C_m} & \frac{R_{im}C_m - \Delta t}{R_{im}C_m} \end{pmatrix} \tag{10}$$

$$\mathbf{B} = \begin{pmatrix} \frac{\Delta t R_{im}}{R_{ia}R_{im}C_i} & \frac{\Delta t R_{im}R_{ia}}{R_{ia}R_{im}C_i} & \frac{\Delta t R_{im}R_{ia}}{R_{ia}R_{im}C_i} \\ 0 & 0 & 0 \end{pmatrix} \tag{11}$$

Assume we measure the indoor temperature but not the temperature in the wall. Therefore, the measurement equation is given by:

$$Y^t = T_i^t + \varepsilon^t \tag{12}$$

The measurement equation is:

$$Y^t = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} T_i^t \\ T_m^t \end{pmatrix} + \varepsilon^t \tag{13}$$

1 Question 4.2

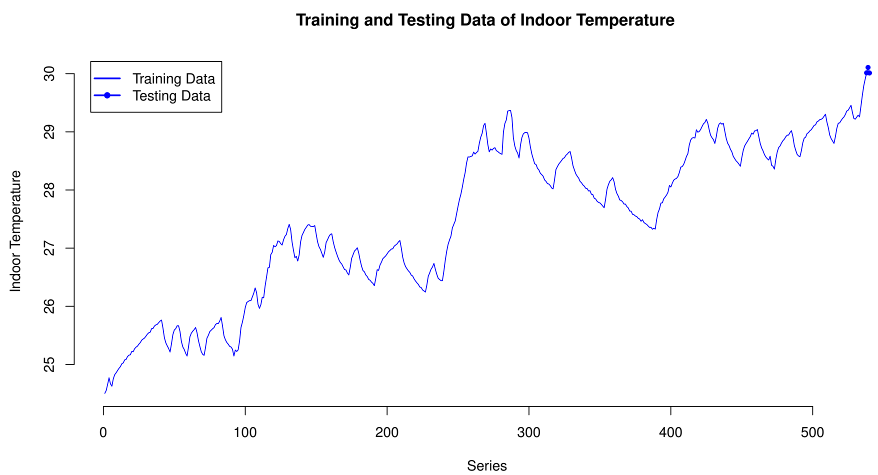

Figure 1. Training and Testing Data of Indoor Temperature. The overall trend is increasing, while there are fluctuations.

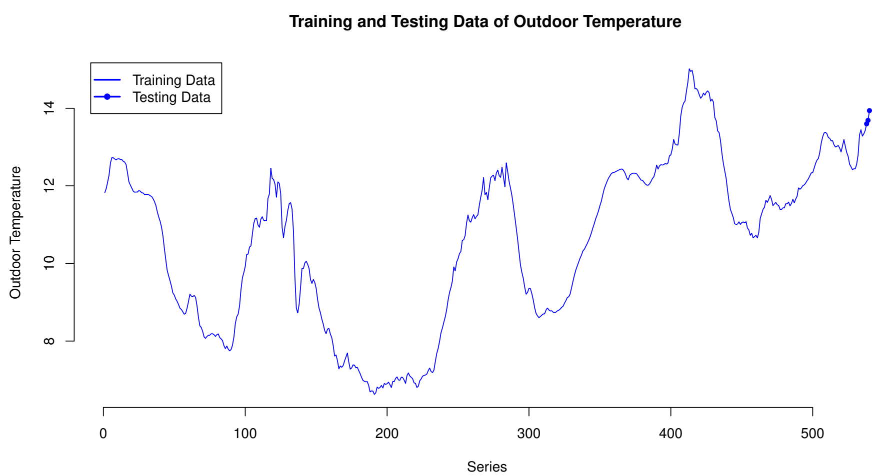

Figure 2. Training and Testing Data of Outdoor Temperature.

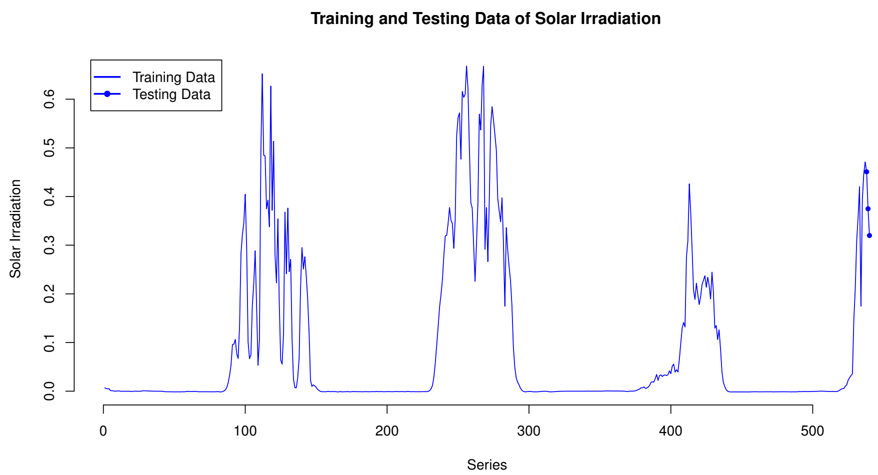

Figure 3. Training and Testing Data of Solar Irradiation. The solar irradiation fluctuates more wildly in the first day. The average solar irradiation is lower in the third day. There is no solar irradiation during night.

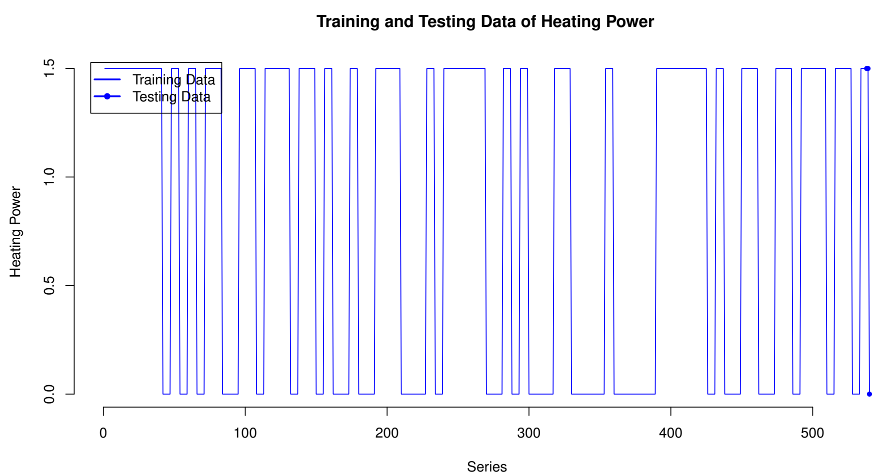

Figure 4. Training and Testing Data of Heating Power. There are only two values. When the heating facility is open, the power is 1.5. There is no obvious pattern of when the heating facility will be open.

2 Question 4.3

The "fkf" function in "FKF" package can be used to reconstruct the state value by means of Kalman Filter. The confidence interval can also be calculated using the standard error from the return value. The one-step predictions and their prediction interval can be calculated. The result are plotted in the following figures.

state 0	uncertainty	reconstruction 0	prediction	standard error
[25, 25]	[10, 10]	[24.52, 25]	$\begin{bmatrix} 30.17309 & 30.80634 & 31.10986 \\ 29.32372 & 29.40866 & 29.54843 \end{bmatrix}$	$\begin{bmatrix} 0.86246 & 1.32928 & 1.74762 \\ 1.929827 & 2.168612 & 2.426792 \end{bmatrix}$

Table 1. Result of Reconstruction and Prediction using Given Parameters

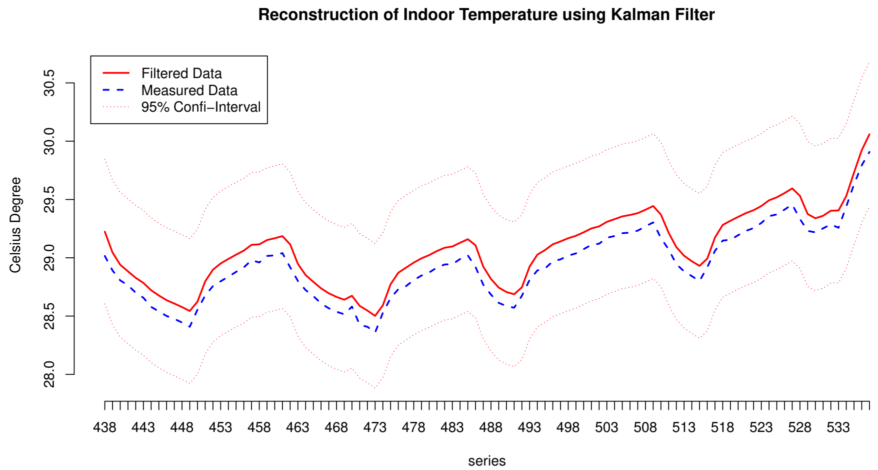

Figure 5. Reconstruction of Indoor Temperature using Kalman Filter (last 100 observations).

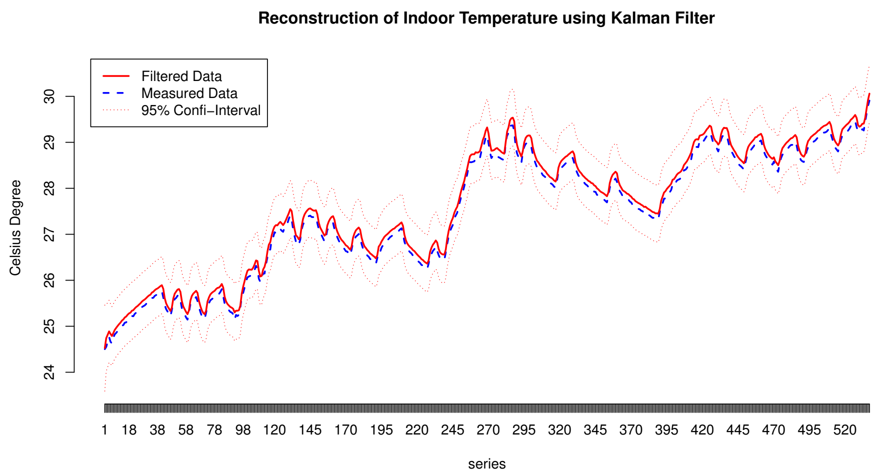

Figure 6. Reconstruction of Indoor Temperature using Kalman Filter (all observations).

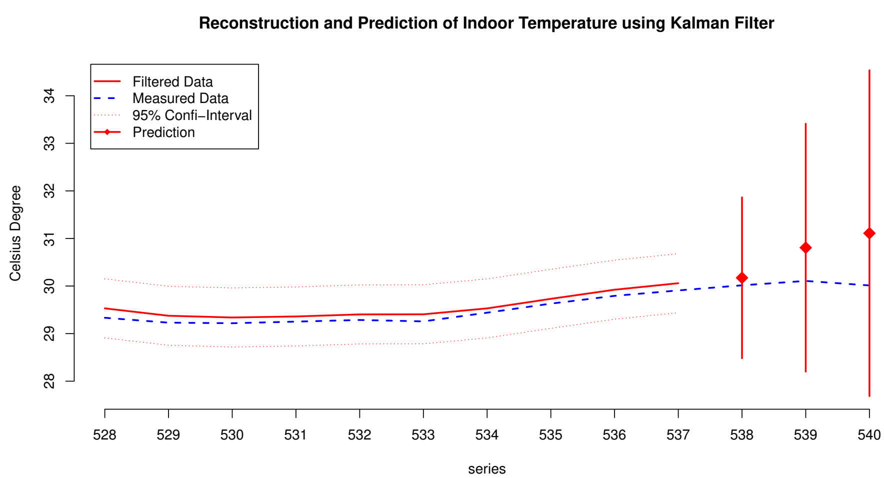

Figure 7. Reconstruction and Prediction of Indoor Temperature using Kalman Filter (last 10 observations and 3 predictions).

3 Question 4.4

To get the maximum log-likelihood of reconstruction result, the parameters of initial values, uncertainty of initial values, and structures of variances are optimized. The upper bound of initial uncertainty is set to 10, while the lower bound of sigma is set to 10e-6.

state 0	uncertainty	reconstruction 0	prediction	standard error
[24.95, 24.73]	[9.95, 9.99]	[10e-6, 0.153]	$\begin{bmatrix} 29.99424 & 30.62545 & 30.92732 \\ 29.14752 & 29.23219 & 29.37152 \end{bmatrix}$	$\begin{bmatrix} 0.00882 & 0.03355 & 0.07363 \\ 0.27702 & 0.38346 & 0.47916 \end{bmatrix}$

Table 2. Result of Reconstruction and Prediction using Optimized Parameters

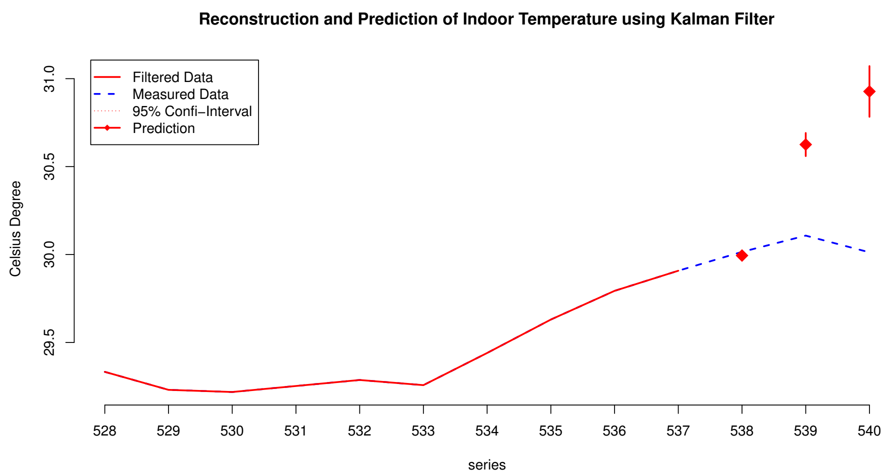

Figure 8. Reconstruction and Prediction of Indoor Temperature using Kalman Filter (last 10 observations and 3 predictions, optimized model). We can see that the reconstructed values are nearly the same as the measured values. However, the predicted result is not very satisfying. It may be caused by the over-fitting of measured value.

Appendix

1. Detailed Simplification of eq.6 and eq.7 in Question 4.1

$$T_i^t = T_i^{t-1} + \frac{\Delta t}{R_{im}C_i}\left(T_m^{t-1} - T_i^{t-1}\right) + \frac{\Delta t}{R_{ia}C_i}\left(T_a^{t-1} - T_i^{t-1}\right) + \frac{\Delta t}{C_i}\Phi_h^{t-1} + \frac{\Delta t}{C_i}G_v^{t-1} \tag{14}$$

$$T_m^t = T_m^{t-1} + \frac{\Delta t}{R_{im}C_m}\left(T_i^{t-1} - T_m^{t-1}\right) \tag{15}$$

$$R_{ia}R_{im}C_iT_i^t = R_{ia}R_{im}C_iT_i^{t-1} + \Delta t R_{ia}\left(T_m^{t-1} - T_i^{t-1}\right) + \Delta t R_{im}\left(T_a^{t-1} - T_i^{t-1}\right) + \Delta t R_{im}R_{ia}\Phi_h^{t-1} + \Delta t R_{im}R_{ia}G_v^{t-1} \tag{16}$$

$$R_{im}C_mT_m^t = R_{im}C_mT_m^{t-1} + \Delta t\left(T_i^{t-1} - T_m^{t-1}\right) \tag{17}$$

$$R_{ia}R_{im}C_iT_i^t = \left(R_{ia}R_{im}C_i - \Delta t R_{ia} - \Delta t R_{im}\right)T_i^{t-1} + \Delta t R_{ia}T_m^{t-1} + \Delta t R_{im}T_a^{t-1} + \Delta t R_{im}R_{ia}\Phi_h^{t-1} + \Delta t R_{im}R_{ia}G_v^{t-1} \tag{18}$$

$$R_{im}C_mT_m^t = \left(R_{im}C_m - \Delta t\right)T_m^{t-1} + \Delta t T_i^{t-1} \tag{19}$$

2. Optimization of Parameters in Kalman Filter

```r
logLikKalmanFilter <- function(par){
    matSigma1 <- matrix(c(par[5], 0, 0, par[6]), nrow = 2)
    matSigma2 <- matrix(par[7])
    result <- fkf(a0 = c(par[1], par[2]), P0 = diag(c(par[3], par[4]), 2), dt = matB %*% matU, ct = 0,
                  Tt = matA, Zt = matC, HHt = matSigma1, GGt = matSigma2, yt = matrix(dat$tempIn[1:numTrain], nrow = 1),
                  check.input = TRUE)
    return(-result$logLik)
```

```
}
resultOptim <- optim(c(25, 25, 10, 10, 0.5, 0.5, 0.5), logLikKalmanFilter, method = "L-BFGS-B", lower = rep(10^-6, 7),
    upper = c(40, 40, 40, 40, 5, 5, 5))
```

3. Prediction using State Space Model

```
predictKalmanFilter <- function(matA, matB, matU.pred, vecX.filt_latest, matVar.filt_latest, matSigma1, numX = 2){
    numU <- dim(matU.pred)[1]
    numPred <- dim(matU.pred)[2]
    matX.pred <- matrix(NA, nrow = numX, ncol = numPred)
    mat3Var.pred <- array(NA, c(numX, numX, numPred))
    #
    vecColX.filt_latest <- matrix(vecX.filt_latest, nrow = numX)
    vecColX.pred <- matrix(matX.pred[,1], nrow = numX)
    vecColU.pred <- matrix(matU.pred[,1], nrow = numU)
    vecColX.pred <- matA %*% vecColX.filt_latest + matB %*% vecColU.pred
    mat3Var.pred[,,1] <- matA %*% matVar.filt_latest %*% t(matA) + matSigma1
    matX.pred[,1] <- vecColX.pred
    if (numPred >= 2) {
        for (i in 2:numPred) {
            vecColX.pred_latest <- vecColX.pred
            vecColX.pred <- matrix(matX.pred[,i], nrow = numX)
            vecColU.pred <- matrix(matU.pred[,i], nrow = numU)
            vecColX.pred <- matA %*% vecColX.pred_latest + matB %*% vecColU.pred
            mat3Var.pred[,,i] <- matA %*% mat3Var.pred[,,(i - 1)] %*% t(matA) + matSigma1
            matX.pred[,i] <- vecColX.pred
        }
    }
    return(list("matX.pred" = matX.pred, "mat3Var.pred" = mat3Var.pred))
}
```