State Space model: focus on EM

Donelli Francesco, Esposito Alessandro, Esposito Francesco

Contents:

Application of the Expectation-Maximization Algorithm in State Space Models: A Comprehensive Exploration

*Abstract***:** State space models are versatile mathematical and statistical tools with applications spanning various scientific disciplines. These models facilitate the modeling of dynamic processes by incorporating both observable and latent state variables, making them invaluable in scenarios with incomplete or noisy data. This paper delves into the utilization of the Expectation-Maximization (EM) algorithm within the context of state space models. The EM algorithm plays a crucial role in parameter estimation and handling latent variables in such models, contributing significantly to interdisciplinary applications. The paper provides a thorough overview of state space models and the EM algorithm, elucidating their context and significance. It also delves into the technical intricacies of implementing EM in state space models, encompassing the Expectation and Maximization steps. Furthermore, the paper outlines the practical implementation of EM in state space modeling. Limitations of the methodology are identified, and avenues for future research are suggested. The paper concludes by summarizing key findings, contributions, and implications.

1. Introduction

State space models constitute a class of mathematical and statistical models employed across diverse scientific domains, including systems engineering, economics, finance, biology, and the social sciences. These models

provide a flexible framework for capturing the dynamics of complex systems by accommodating both observed variables and latent states that remain unobservable. In the realm of time series analysis, state space models offer a powerful tool for unraveling the underlying structure of time-varying data.

To elucidate the concept, consider the scenario of tracking a robot's movement within a room. In this context, state space models act as mathematical allies, aiding in precisely determining the robot's location, even when confronted with incomplete or slightly erroneous data due to factors like low battery levels or sensor malfunctions. These models leverage available information, however imperfect, to estimate the robot's exact position and even predict its future movements. State space models prove especially valuable when data are tainted by noise or influenced by latent variables that impact observable data.

This paper primarily centers on the Expectation-Maximization (EM) algorithm within the context of state space models. It aims to illuminate how the EM algorithm facilitates parameter estimation and latent variable handling, bearing immense significance in interdisciplinary applications. The following sections provide an extensive overview of state space models and the EM algorithm, shedding light on their context and relevance. The technical nuances of employing EM in state space models, including the Expectation and Maximization steps, are explored. Practical implementation details are also offered, culminating in the identification of methodological limitations and prospects for future research.

2. Background

2.1 Definition of State Space Models

A state space model typically comprises two fundamental equations:

- *State Equation:* the state equation describes the evolution of latent variables or "states" over time. These states represent unobservable quantities that influence the observed data.
- *Observation Equation:* the observation equation establishes the connection between latent states and observed data. It accounts for potential discrepancies between the true state and the measured data, often due to noise or measurement errors.

3. Mathematical Foundations of State Space Models

3.1 Observation or Measurement Equation

The observation equation bridges the gap between observed data and underlying, unobservable states. In essence, it quantifies how the measured data relate to the true state, accounting for potential noise in the observations.

For a simple linear model, the observation equation can be expressed as:

$$y_t = H_t x_t + v_t$$

Where:
- y_t is the observed data at time t.
- H_t is the observation matrix that maps the state to the observation (e.g. would contain details on how a sensor's readings relate to the states you're trying to estimate).
- x_t is the state vector at time t.
- v_t is the observation noise at time t, typically assumed to be Gaussian with mean zero and covariance R_t, i.e., $v_t \sim N(0, R_t)$.

This mathematical formulation captures situations where observed data deviate from true states due to measurement imprecisions or inaccuracies.

3.2 State or Transition Equation

The state equation governs the evolution of unobservable states over time. It encapsulates the idea that the state at time t is some function of the state at time $t-1$, possibly subject to process noise.

$$x_t = F_t x_{t-1} + w_t$$

Where:
- x_t is the state vector at time t.
- F_t is the state transition matrix, describing the relationship between the state at time $t-1$ and the state at time t in absence of control inputs (e.g. might encode details like acceleration and velocity relation between successive time points).
- w_t is the process noise at time t, typically assumed to be Gaussian with mean zero and covariance Q_t, i.e., $w_t \sim N(0, Q_t)$.

This equation encapsulates scenarios where the underlying state evolves systematically based on past states, with occasional unexpected changes.

In summary, state space models provide a versatile framework for modeling dynamic systems with both observable and latent components. The subsequent sections of this paper delve into the technicalities of employing the Expectation-Maximization (EM) algorithm within this framework, elucidating its role in parameter estimation and latent variable handling.

4. Expectation-Maximization Algorithm in State Space Models

Parameter estimation serves as a cornerstone in state space models, acting as a bridge that aligns the abstract modeling constructs with real, observed data. While the landscape of parameter estimation is populated with methodologies like Maximum Likelihood Estimation (MLE), Bayesian inference, and various optimization techniques, the Expectation-Maximization (EM) algorithm rises prominently within the state space modeling realm.

The EM algorithm's introduction into state space models is not merely incidental. The model's inherent nature, which sometimes grapples with incomplete observed data or latent variables, demands a strategy that can robustly handle these nuances. Enter the EM algorithm.

4.1 The Expectation Step (E-Step):

- **Definition & Mathematical Insight**: The E-Step sets the stage for the EM algorithm's iterative process. At its core lies the Q function, represented as:

  $$Q(\theta|\theta^{(t)}) = \mathbb{E}[\log L(\theta|Y, Z)|Y, \theta^{(t)}]$$

  where:

  - $Q(\theta|\theta^{(t)})$ signifies the expected complete-data log-likelihood conditioned on the current parameter estimates, $\theta^{(t)}$.
  - $\mathbb{E}[]$ stands for the expected value, encapsulating the essence of this step: computing the expected log-likelihood of observed data.
  - The complete-data likelihood, given by $\log L(\theta|Y, Z)$, is a holistic representation accounting for both the observed data Y and latent variables Z. This equation thus merges the visible with the invisible—the observed with the latent.

- **Why it Matters**: Beyond its mathematical elegance, the E-Step's significance is profound. By venturing into the realm of latent variables and integrating their probable values, this step effectively bridges the gaps in the dataset. In simpler terms, it fills in the missing pieces of the puzzle, ensuring a comprehensive understanding of the data landscape. Such an approach isn't just about adding numbers; it's about adding clarity in the face of latent variable-induced uncertainty.

- **Application in State Space Models**: When funneling the E-Step into the conduit of state space models, its utility becomes even more palpable. The step ensures that parameter estimates aren't based on mere observed data but embrace the hidden narratives woven by the latent variables. This enriched perspective is fundamental to the successful convergence of the EM algorithm. Additionally, it molds pivotal pillars of state space models, specifically offering estimates for the observation matrix H_t. In doing so, the E-Step doesn't just optimize; it illuminates, guiding the algorithm through the mists of missing information towards the light of a more accurate model.

4.2 The Maximization Step (M-Step):

- **Definition & Purpose**: If the E-Step is about understanding and assessing the data landscape, the M-Step is about acting upon it. While the former computes the expected log-likelihood by embracing both observed and latent data, the M-Step embarks on a mission to finetune the parameters based on this computed expectation. It's the algorithm's beacon for optimization, laser-focused on the goal of maximizing the expected log-likelihood.
- **Mathematical Deep Dive**: The soul of the M-Step is captured succinctly by the equation:
  $\theta^{(t+1)} = \text{argmax}_\theta Q(\theta|\theta^{(t)})$

  At a glance, it might appear as a confluence of Greek letters and mathematical symbols. But, delve deeper, and the story unfurls. This equation steers the iterative journey of parameters, moving from one estimate ($\theta^{(t)}$) to a more refined one ($\theta^{(t+1)}$). The operative word here is *maximization*, denoted by *argmax*, underscoring the algorithm's drive to unearth the optimal parameter values that maximize our Q function—our expected log-likelihood. However, it's not a one-size-fits-all procedure. Depending on the nuances of the state space model, the optimization journey might harness numerical techniques or navigate the realms of closed-form solutions.
- **Application in State Space Models**: Contextualizing the M-Step within the canvas of state space models adds more color to its significance. It isn't just an algorithmic step but a journey where parameters undergo rigorous refinement. This step persistently tweaks and nudges these parameters, ensuring they align better with the observed data's likelihood. And the fruits of this labor? Enhanced and optimized versions of pivotal components, including but not limited to the observation matrix H_t, the state transition matrix F_t, and the noise covariances R_t and Q_t. In essence, the M-Step is the bridge from understanding the data landscape in the E-Step to adapting and optimizing the parameters to best fit that understanding.

4.3 Convergence and Stopping Criteria

Convergence is determined by monitoring changes in parameter estimates between iterations or when the log-likelihood reaches a plateau. The criteria for stopping are chosen to balance precision with computational efficiency, tailored to the model's specific characteristics and the available computational resources.

Mathematically, the convergence check can be expressed as:

$$\Delta\theta^{(t)} = \theta^{(t+1)} - \theta^{(t)}$$

Where:
- $\Delta\theta^{(t)}$ denotes the change in parameter estimates between iterations t and $t+1$.

The stopping criterion can be formulated as:

$$\text{Stop when } |\Delta\theta^{(t)}| < \epsilon$$

Where:
- ϵ represents a predefined threshold indicating the desired precision level.

Stopping criteria are essential to prevent overfitting and to strike a balance between precision and computational efficiency. The choice of stopping criteria in practice is often a matter of balancing the wish for more precise parameter estimates with the need for a manageable computational workload.

5 Filtering and Smoothing

Filtering and smoothing represent indispensable techniques in state space models for estimating latent or hidden states based on observed data. These methods play critical roles in generating real-time estimates (filtering) and enhancing past state estimates (smoothing) by incorporating future observations.

5.1 Filtering

Filtering furnishes an estimate of the current hidden state by considering all available observations up to the present moment. The Kalman filter, particularly effective in linear state space models, is a widely-used method for this purpose. It employs a recursive algorithm to predict the current state and subsequently updates this prediction when new observations arrive. The Kalman filter excels in real-time applications, delivering accurate state estimates as new data streams in.

How the Kalman Filter Works? The Kalman filter operates in two fundamental phases: *prediction* and *update*.The aim is to estimate the state x_t of a discrete-time dynamic system from observations y_t through an iterative cycle that includes these two phases.

- *Prediction*: Given an estimate of the state x_{t-1} and its uncertainty P_{t-1} at time $t-1$, the Kalman filter predicts the state at time t and its associated uncertainty, using the state transition model and the process noise covariance matrix. The prediction is made using the following equations:

$$x_{t|t-1} = Ax_{t-1|t-1} + Bu_t \tag{1}$$
$$P_{t|t-1} = AP_{t-1|t-1}A^T + Q \tag{2}$$

Where:
- A is the transition matrix
- B is the control matrix, outlining how the control vector u_t influences the system state's evolution, quantifying the effect of external factors on the state.
- u_t is the control vector, representing the control forces or input signals applied to the system at time t.
- Q is the process noise covariance matrix, describing the uncertainty associated with the state transition mod el.

- *Update*: Once a new measurement y_t is obtained, the filter updates the state estimate and its uncertainty based on the estimation error, using the observation model and the observation noise covariance matrix. The update equations are:

$$K_t = P_{t|t-1}C^T(CP_{t|t-1}C^T + R)^{-1}, \tag{3}$$
$$x_{t|t} = x_{t|t-1} + K_t(y_t - Cx_{t|t-1}), \tag{4}$$
$$P_{t|t} = (I - K_tC)P_{t|t-1}. \tag{5}$$

Where:
- C is the observation matrix, linking the internal state x_t to the observations y_t and serves to generate a measurement prediction that is then compared to the actual measurement.
- R is the measurement noise covariance matrix v_t.
- K_t is the Kalman gain, determining how much weight is given to the error between the actual measurement and the predictio n.

5.2 Smoothing

Smoothing techniques refine past state estimates by considering future observations. These methods are valuable when a more accurate representation of the entire state trajectory is needed. The Kalman smoother, an extension of the Kalman filter, is a common choice for smoothing. It improves state estimates once all data points have been observed and essentially works 'backward,' revising prior state estimates using future observations.

The smoothing process unfolds in two stages:

Forward Pass: During this phase, analogous to the Kalman filtering stage, the Kalman filter is executed forward in in time from $t = 1$ to $t = T$ to generate "filtered" state estimates. These estimates are predictions of the state based on past and current observations. The filter covariances and Kalman gains for each time step are also recorded during this pass.

Backward Pass: The actual smoothing occurs in this stage. Starting at $t = T$ and moving back to $t = 1$, the forward-filtered states and future observations are employed to correct state estimates, resulting in "smoothed" state estimates. The standard equations for the backward pass in the Kalman smoother are as follows:

1. The first equation: $x_{t|T} = x_{t|t} + J_t(x_{t+1|T} - x_{t+1|t})$
2. The second equation: $J_t = P_{t|t}A^T P^{-1}_{t+1|t}$

Where:
- $x_{t|T}$ is the smoothed state estimate at time t, which incorporates all observations from $t = 1$ to T.
- $x_{t|t}$ is the filtered state estimate at time t, obtained from the Kalman filter's forward pass.
- $x_{t+1|T}$ and $x_{t+1|t}$ are the smoothed and filtered state estimates at time $t + 1$, respectively.
- J_t is the smoother gain.
- $P_{t|t}$ and $P_{t+1|t}$ are the filtered covariances at time t and $t + 1$, respectively.
- A is the transition matrix used in the forward pass of the Kalman fi lter.

6 Applications and Significance

Both filtering and smoothing techniques play pivotal roles in state space models, finding applications in a multitude of disciplines, ranging from economics to robotics. These methods are instrumental for tasks such as tracking, prediction, and anomaly detection. They are indispensable in scenarios where real-time state estimates are imperative or retrospective analysis is required.

7 Implementation and Results

In this section, we present the practical implementation of the state space model discussed earlier using the Expectation-Maximization (EM) algorithm.

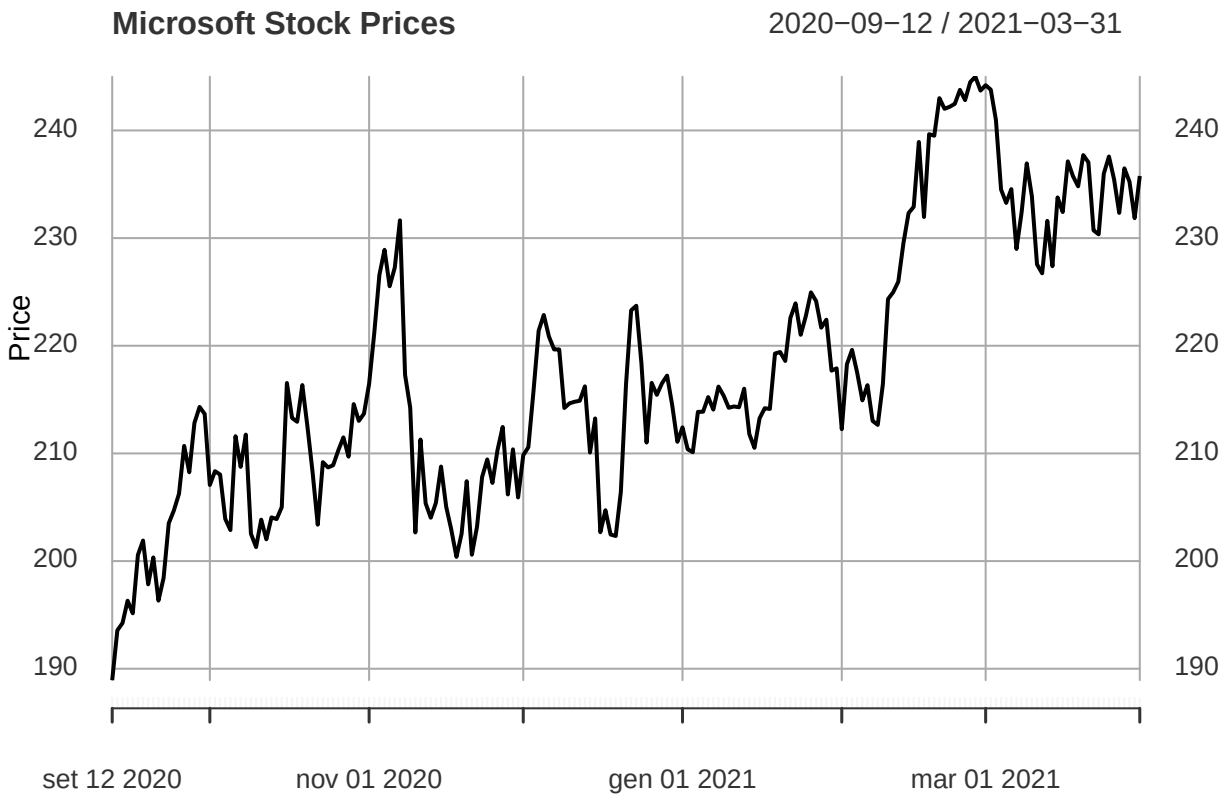
For the implementation, we chose the last 200 observations of the price trend of Microsoft stocks up to 21-03-2021, considering the column 'close' referred to the closed value of the stocks for each day.

To begin with, let's look at the behavior of the observations over time graphically.

```
## Caricamento del pacchetto richiesto: zoo

##
## Caricamento pacchetto: 'zoo'

## I seguenti oggetti sono mascherati da 'package:base':
##
##     as.Date, as.Date.numeric
```



The implemented algorithm is a combination of the Kalman filter and the EM algorithm.

The role of the Kalman filter is to correct predictions based on observed measurements. This correction is carried out using the Kalman gain, which determines how much weight should be given to the observed measurement versus the model-based prediction. Once the estimates are updated, the filter makes a prediction about the next state of the system.

When dealing with stock price time series, it's crucial to separate the "signal" (i.e., the real price movements of the stock) from the "noise" (random fluctuations that don't pertain to real price movements). The EM method used here seeks to do just that: separate the signal from the noise and provide an accurate estimate of the daily returns and their variance. In doing so, we can get a clearer idea of the direction in which Microsoft's stock price is moving and how much we can trust this estimate.

In this algorithm, `taut` represents the estimate of the system's latent state based on observations and the initialized model's parameters (`F`, `H`, `Q`, `R`). In our context, it would represent an estimate of the expected daily returns given the observations (prices). It is updated in each iteration of the EM algorithm and tends to converge towards a more accurate estimate with each iteration. Practically, `taut` represents the current estimate of the latent signal (i.e., the unknown daily returns) that is being extracted from the observed data.

On the other hand, the EM, based on the current parameters, estimates the distribution of latent variables. In the M-step, it will update the model parameters by maximizing the conditional logarithmic likelihood of the observations given the estimates of the latent variables obtained in the E-step

First of all let's try to describe how the latent variable can be affected based on the variation of the initialized values. Furthermore, it must be said that initialization is not a trivial step. There would be in-depth research and profound analyzes to be able to arrive at an "a priori" knowledge such as to be able to best initialize the matrices.

In our case we will base ourselves on elementary values, having in mind the objective of how they affect the results.

1. **F (Transition Matrix)**:
   - The matrix F determines how the latent state evolves over time. If F is close to 1, it implies the state remains relatively constant over time. This can contribute to making the estimate smoother.
2. **Q (Process Noise Covariance)**:
   - Q represents the variance or uncertainty associated with the process model (i.e., how we expect the state to evolve over time). A smaller value of Q suggests that we expect minor changes in the latent state from one time point to the next, leading to smoother estimates. Conversely, a larger value for Q suggests that we expect significant changes in the state, which can make the estimates more volatile.
3. **H (Observation Matrix)**:
   - H links the latent states to the observations. Typically, if we are working with a single-state model, H might simply be a scalar value. Adjusting it might not have a direct impact on the "smoothing" effect.
4. **R (Observation Noise Covariance)**:
   - R represents the variance or uncertainty associated with the observed measurements. A larger value of R indicates that we have less trust in the observations, leading the Kalman filter to give more weight to model-based forecasts, which can result in smoother estimates. A smaller R suggests that the observations are considered highly reliable, which might lead the filter to respond more rapidly to observed changes.

Therefore the most affectable (and easy to handle) ones seem to be R and Q. Let's see how the estimates of the latent variable behave, plotting them with different values of the two matrices.

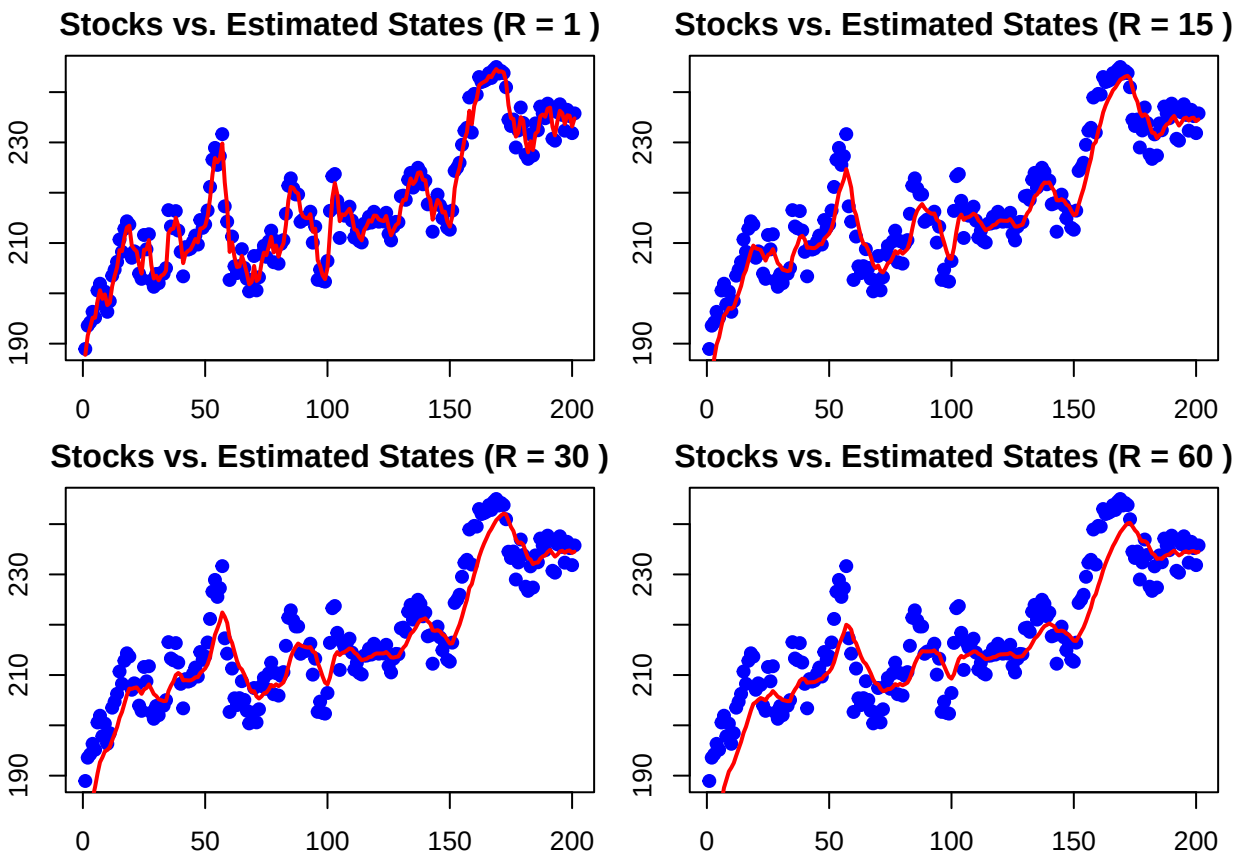
Let's start with R:

```
F <- 1
Q <- 1
H <- 1
R_values <- c(1, 15, 30, 60)

par(mfrow=c(2,2), mar=c(2, 2, 2, 1))

for (R in R_values) {
  results <- run_EM_with_Kalman(stocks, F, H, Q, R)

  plot(stocks, type = "p", pch = 19, col = "blue", xlab = "Time", ylab = "Value",
       main = paste("Stocks vs. Estimated States (R =", R, ")"))
  lines(results$taut, type = "l", lwd = 2, col = "red")
}
```




```
par(mfrow=c(1,1))
```

Increasing R, the algorithm gives less weight to the observations compared to the model predictions. As we expected, the line becomes flatter.

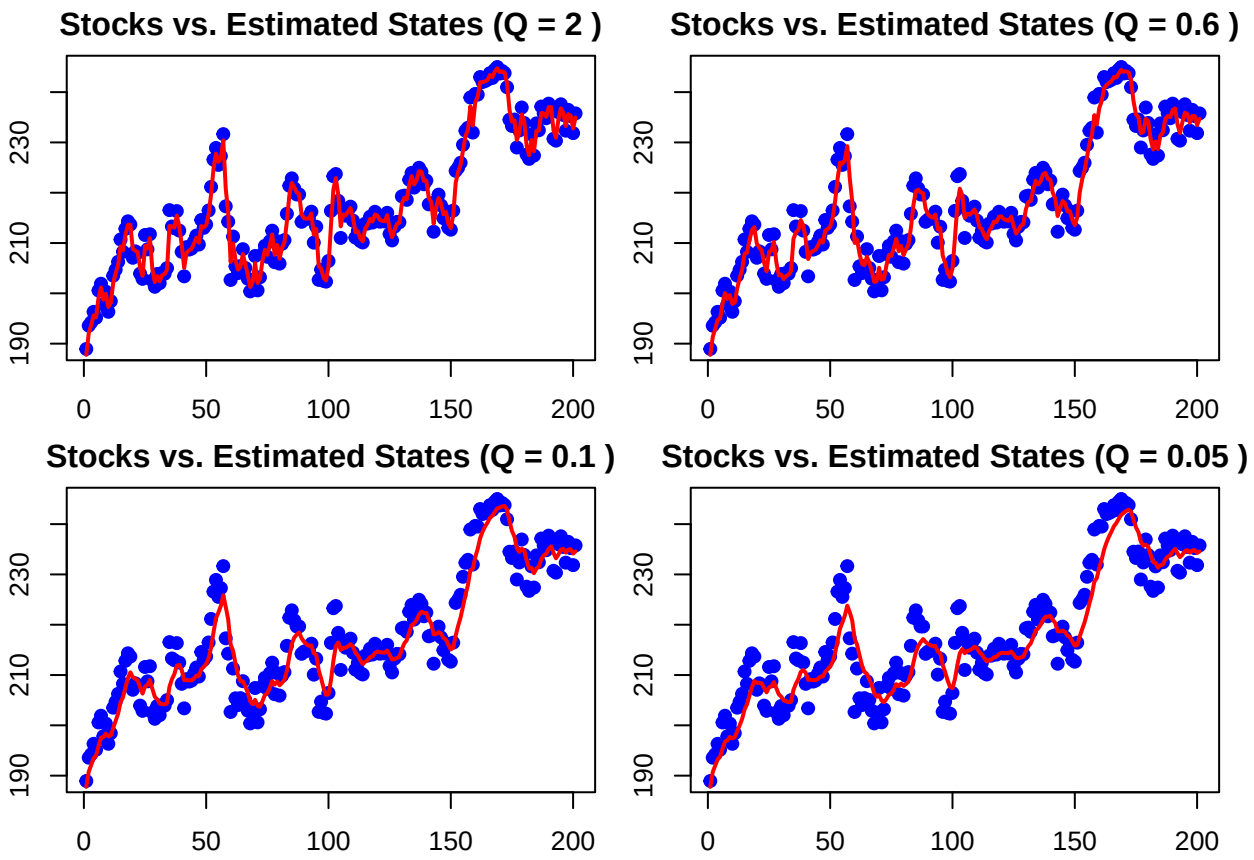
Let's see Q now:

```
F <- 1
H <- 1
R <- 1
Q_values <- c(2, 0.6, 0.1, 0.05)

par(mfrow=c(2,2), mar=c(2, 2, 2, 1))

for (Q in Q_values) {
  results <- run_EM_with_Kalman(stocks, F, H, Q, R)

  plot(stocks, type = "p", pch = 19, col = "blue", xlab = "Time", ylab = "Value",
       main = paste("Stocks vs. Estimated States (Q =", Q, ")"))
  lines(results$taut, type = "l", lwd = 2, col = "red")
}
```




```
par(mfrow=c(1,1))
```

Decreasing Q indicates that we expect the state to evolve slowly over time. Also this behavior can be observed in the plots.

Our assumptions are confirmed.

8 Conclusion

In conclusion, as we have seen, in the application of the EM model to the State Space Model (SSM) on Microsoft share prices, the function matrices that characterize the model have a central role.

We also tried to give an interpretation to the estimates of the latent variable that the model has, highlighting how much the 'noises' affecting the available observations can actuate, making themselves useful for understanding past phenomena and predicting future ones.

Thus, the state space models and the Expectation-Maximization (EM) algorithm constitute powerful tools for modeling dynamic processes with latent variables. These models have applications in diverse fields, making them versatile and invaluable.

Understanding the fundamental principles of state space modeling, parameter estimation using the EM algorithm, and techniques like filtering and smoothing equips researchers and practitioners with the ability to extract meaningful insights from complex and noisy data.

References:

- Statistical Modeling and Computation - Dirk P. Kroese · Joshua C.C. Chan, chapter 11

- TimeSeries Analysis and Its Applications -Robert H. Shumway David S. Stoffer, chapter 6

- Model Estimation by Example - Demonstrations with R - Michael Clark GitHub repository, https://m-clark.github.io/