Advanced Time Series Analysis: Computer Exercise 2

Anders Launer Bæk (s160159)

11 Oktober 2017

Sparring partners:

- Anja Liljedahl Christensen (s162876)
- Marie Mørk (s112770)

Part 1

There are simulated $n = 3000$ where $\epsilon_t \sim \mathcal{N}(0,\, 1)$. ϵ_t is used as noise input for all simulations in part one.

The equation below shows the used parameters in the SETAR(2,1,1). Let us call eq. 1 and eq. 2 parameter set one (par_1).

$$a_0 = [0.125, -0.125] \tag{1}$$

$$a_1 = [0.6, -0.4] \tag{2}$$

Simulation of the SETAR(2,1,1)

The Self-Exciting Threshold AR (SETAR) model is given by eq. 3.

$$X_t = a_0^{(J_t)} + \sum_{i=1}^{k_{(J_t)}} a_i^{(J_t)} X_{t-i} + \epsilon^{(J_t)} \tag{3}$$

where J_t are regime processes. The complete model are defined in eq. 4.

$$X_t = \begin{Bmatrix} a_{0,1} + a_{1,1} X_{t-1} + \epsilon_t & for & X_{t-1} \leq 0 \\ a_{0,2} + a_{1,2} X_{t-1} + \epsilon_t & for & X_{t-1} > 0 \end{Bmatrix} \tag{4}$$

The model X_t (eq. 4) has been simulated with par_1. Its simulation is plotted in fig. **??**.

Estimate the parameters using conditional least squares

```
Setar <- function(par, model) {
    #
    e_mean <- rep(NA, length(model))
    #
    for (t in 2:length(model)) {
        if (model[t - 1] <= 0) {
            e_mean[t] <- par[1] + par[2] * model[t - 1]
        } else {
            e_mean[t] <- par[3] + par[4] * model[t - 1]
        }
    }
```

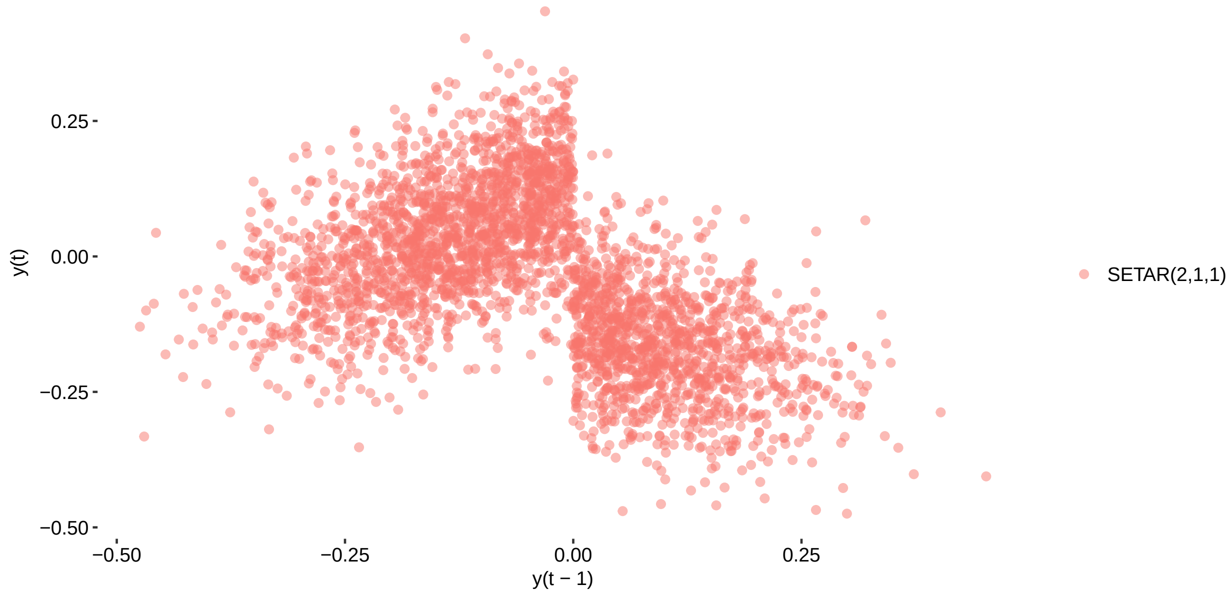
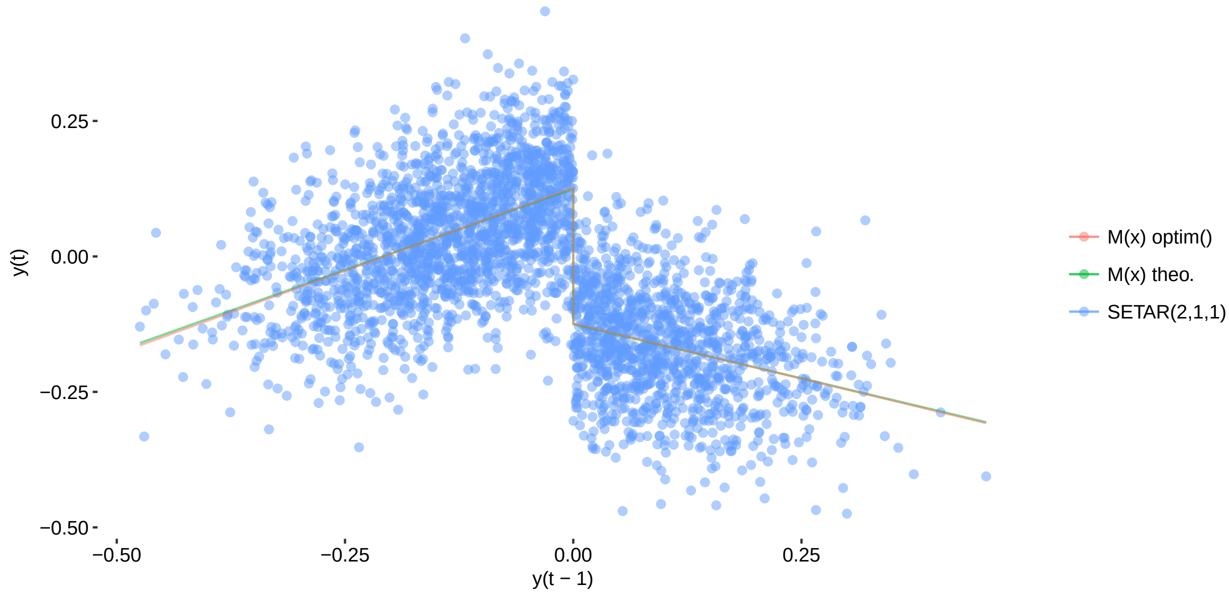


Figure 1: Two simulated SETAR(2,1,1) models using par_1 and par_2.

```
    #
    return(e_mean)
}


RSSSetar <- function(par, model) {
    # conditional mean
    e_mean <- Setar(par, model)

    ## Calculate and return the residuals
    return((model - e_mean)^2)
}


PESetar <- function(par, model) {
    # conditional mean
    e_mean <- Setar(par, model)

    ## Calculate and return the objective function value
    return(sum((model - e_mean)^2, na.rm = TRUE))
}
```



Den lodrette linje findes selfølgelig ikke!!

comment !!!

Part 2

resolution 50 max_change_p 0.1

only change the slope par[2] and par [4]

N = 1:3000

2

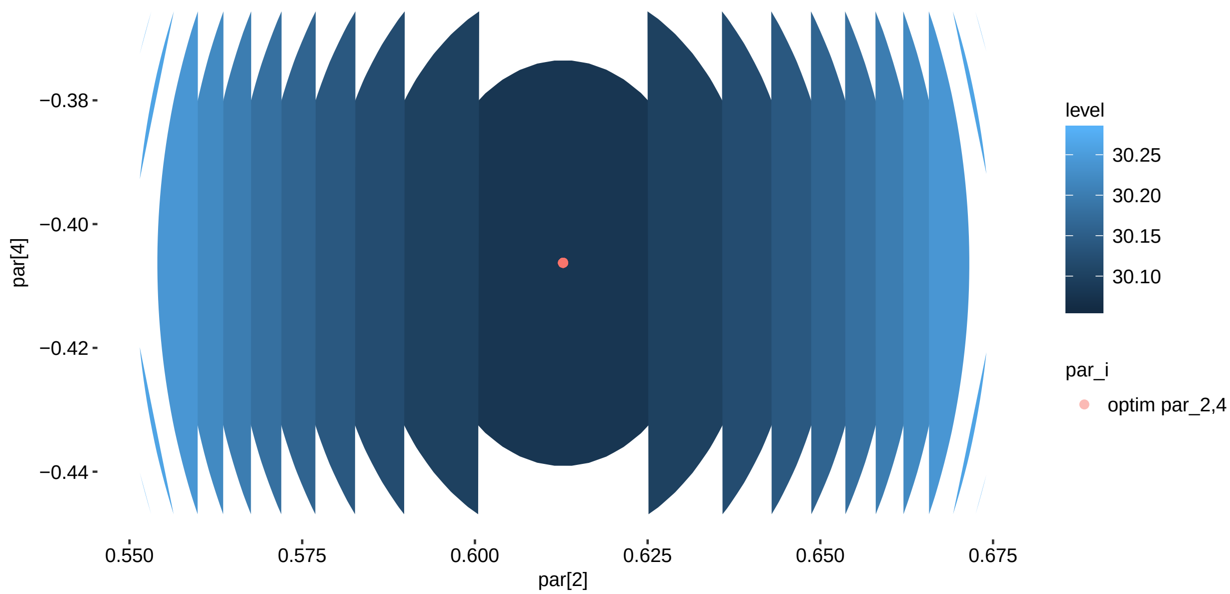
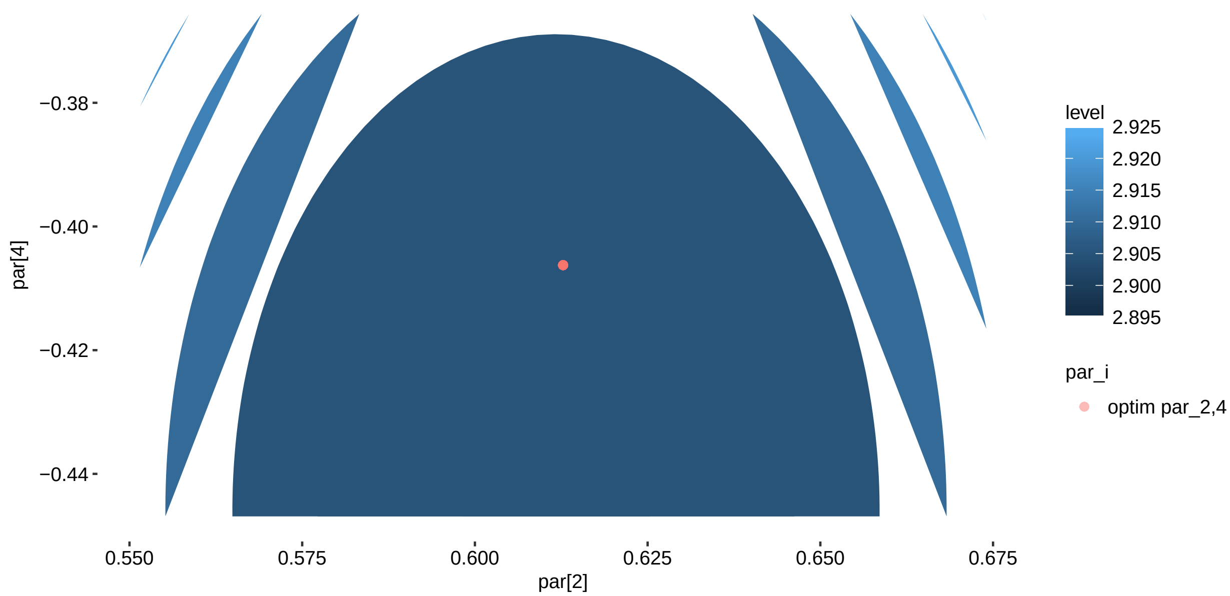
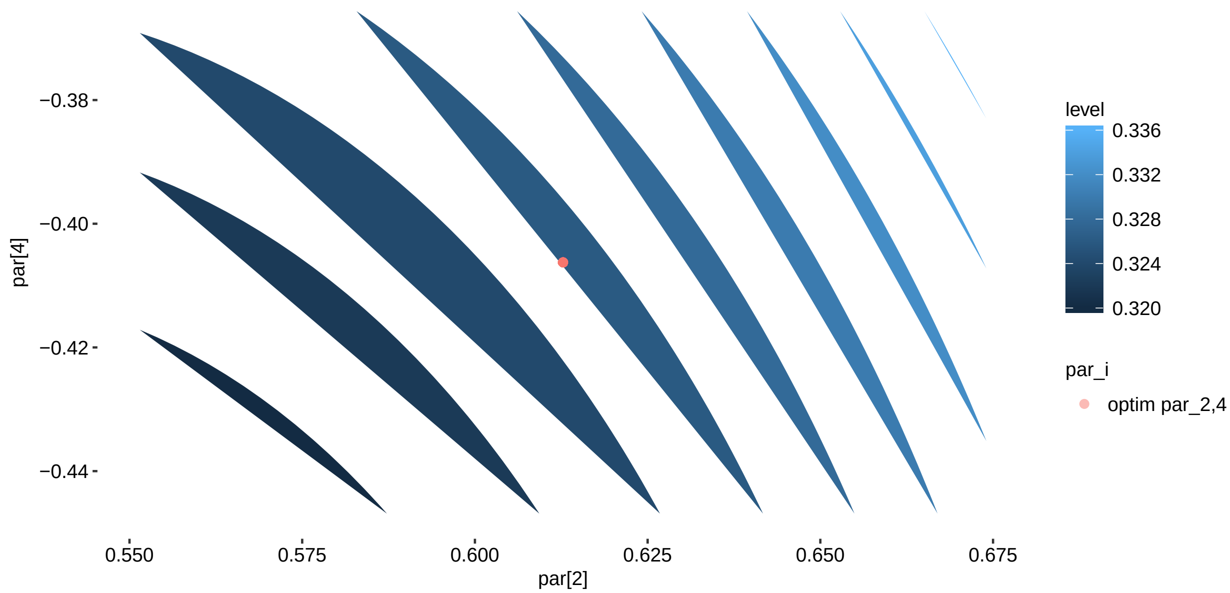


Figure 2: Contour plot of the conditional parametric model approach.

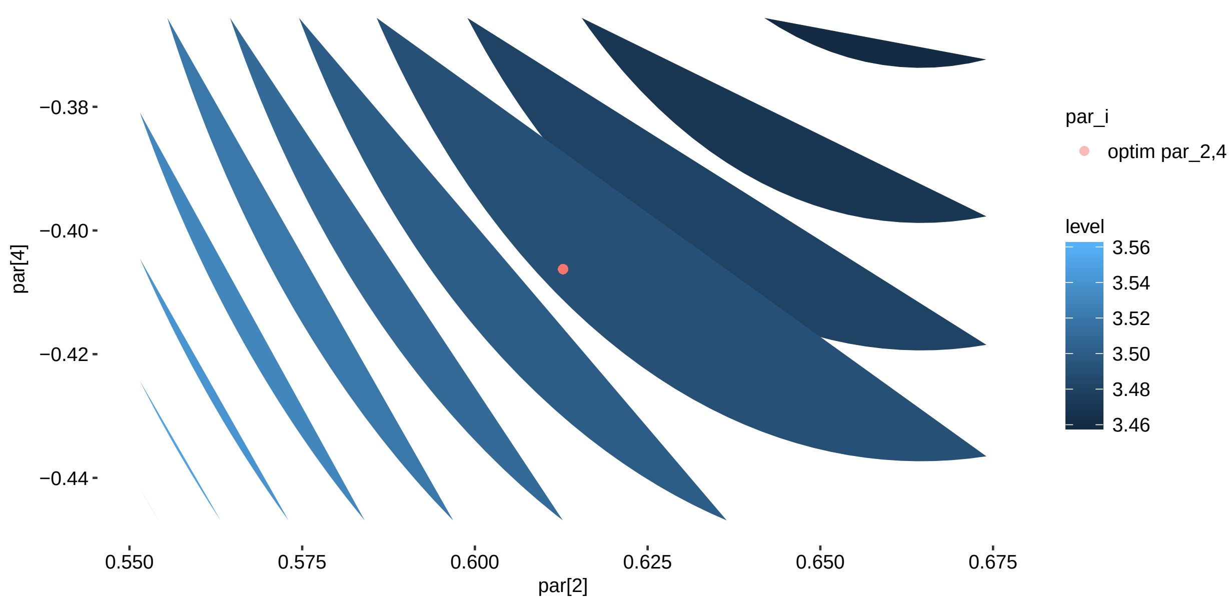
N = 1:300



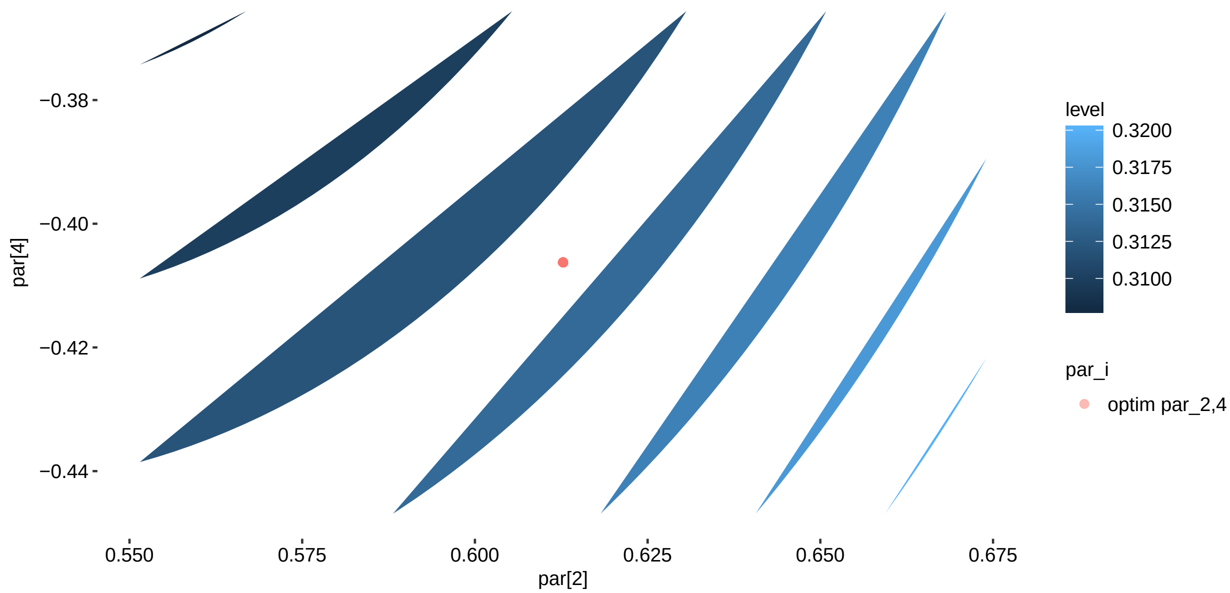
N = 1:30



N = 1001:1300



N = 1001:1030



Discuss my findings

Part 3

AUTO regression outside one -> keep growing

I will consider the following AR(2)-AR(4) non-linear doubly stochastic model, eq. 5.

$$
\begin{aligned}
Y_t &= \sum_{k=1}^{2} \left(\Phi_{t-(1-k)} Y_{t-k}\right) + \epsilon_t \\
\Phi_t - \mu &= \sum_{n=1}^{4} \left(\phi_n \left(\Phi_{t-n} - \mu\right)\right) + \zeta_t \\
\Phi_t &= \sum_{n=1}^{4} \left(\phi_n \left(\Phi_{t-n} - \mu\right)\right) + \zeta_t + \underbrace{\mu \left(1 - \sum_{n=1}^{4} \left(\phi_n\right)\right)}_{\delta}
\end{aligned}
\tag{5}
$$

state space

$$
\begin{aligned}
\begin{pmatrix} \Phi_t \\ \Phi_{t-1} \\ \Phi_{t-2} \\ \Phi_{t-3} \\ \delta_t \end{pmatrix} &= \begin{pmatrix} \phi_1 & \phi_2 & \phi_3 & \phi_4 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \Phi_{t-1} \\ \Phi_{t-2} \\ \Phi_{t-3} \\ \Phi_{t-4} \\ \delta_{t-1} \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \delta_t \\
Y_t &= \begin{pmatrix} Y_{t-1} & Y_{t-1} & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \Phi_t \\ \Phi_{t-1} \\ \Phi_{t-2} \\ \Phi_{t-3} \\ \delta_t \end{pmatrix} + e_t
\end{aligned}
\tag{6}
$$

delta som state i stedet for constant -> estimate

Tænk hvor havd der sker i den underlæggende proces ?

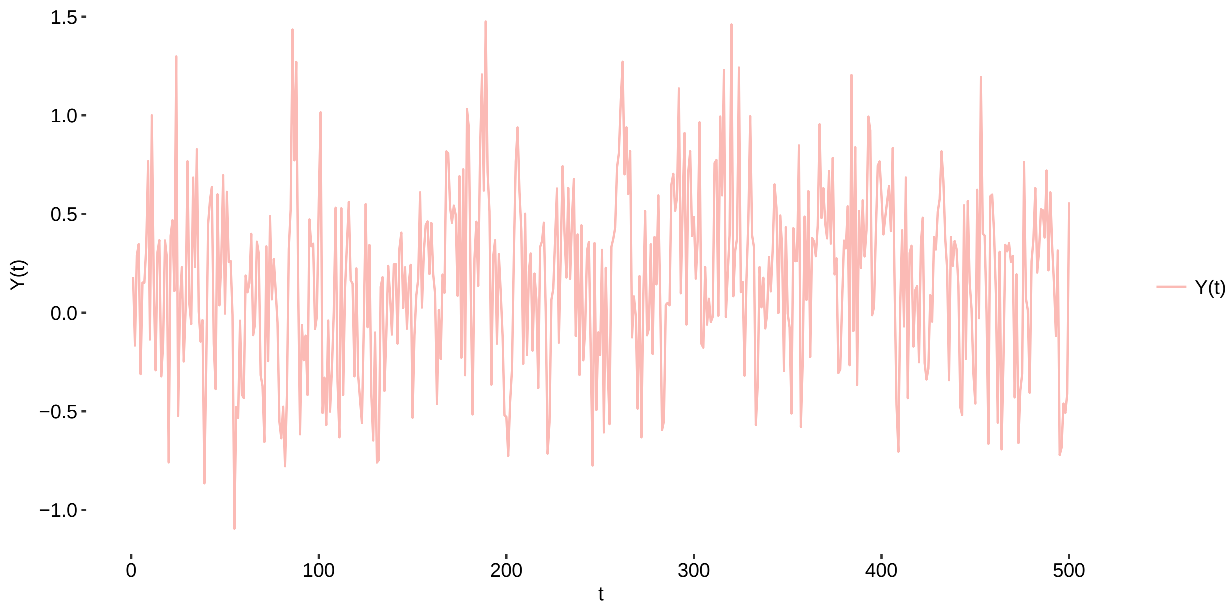
hvordan er stationary conditions?

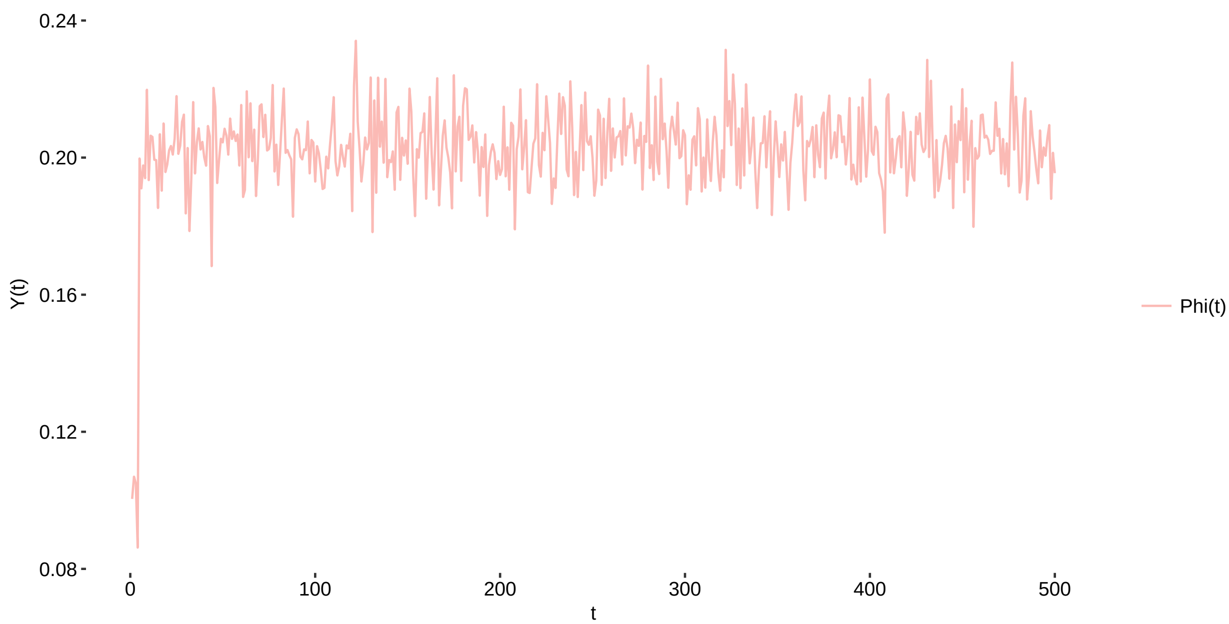
Fordele ved at se delta som et state..

Tjek for stabilitet

Simulate

```
## [1] -0.0003183665
```





Comment

Part 4

Following simple state space model is given, eq. 7.

$$x_{t+1} = ax_t + v_t y_t \quad = x_t + e_t \tag{7}$$

where a is an unknown parameter and v_t and e_t are mutually uncorrelated white noise processes with their variences σ_v^2 and σ_e^2.

Part 4a

The model from eq. 7 is on state space form in eq. 8

$$\begin{aligned} x_{t+1} &= ax_t + v_t \\ y_t &= x_t + e_t \end{aligned} \tag{8}$$

Simulate

Simulate X time series where $a = 0.4$, $\sigma_v^2 = \sigma_e^2 = 1$ with zero mean

Rewrite

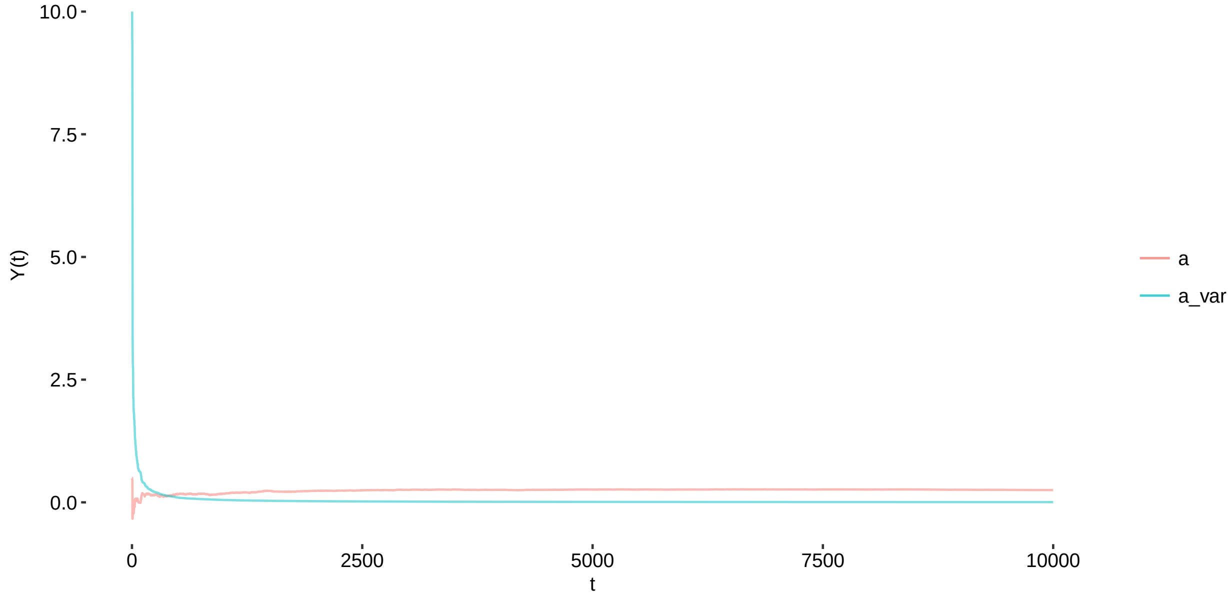
$$\begin{aligned} \begin{pmatrix} x_{t+1} \\ a_{t+1} \end{pmatrix} &= \begin{pmatrix} a_t & 0 \\ 0 & a_t \end{pmatrix} \begin{pmatrix} x_t \\ 1 \end{pmatrix} + \begin{pmatrix} v_t \\ 0 \end{pmatrix} \\ y_t &= \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_t \\ 1 \end{pmatrix} + e_t \end{aligned} \tag{9}$$

Part 4b

```
##----------------------------------------------------------------
## EKF algorithm for use in Part 4 of computer exercise 2 in Advanced Time
## Series Analysis
##----------------------------------------------------------------

ext_kalman <- function(y, aInit = 0.5, aVarInit = 1, sigma.v = 1) {
    ## aInit : The starting guess of the AR coefficient estimate aVarInit : The
    ## initial variance for estimation of the AR coefficient sigma.v : Standard
    ## deviation of the system noise of x in the filter

    # Initialize---- Init the state vector estimate
    zt <- c(0, aInit)
    # Init the variance matrices
    Rv <- matrix(c(sigma.v^2, 0, 0, 0), ncol = 2)
    # sigma.e : Standard deviation of the measurement noise in the filter
    Re <- 1

    # Init the P matrix, that is the estimate of the state variance
```

```
    Pt <- matrix(c(Re, 0, 0, aVarInit), nrow = 2, ncol = 2)
    # The state is [X a] so the differentiated observation function is
    Ht <- t(c(1, 0))
    # Init a vector for keeping the parameter a variance estimates
    aVar <- rep(NA, length(y))
    # and keeping the states
    Z <- matrix(NA, nrow = length(y), ncol = 2)
    Z[1, ] <- zt

    ## The Kalman filtering----
    for (t in 1:(length(y) - 1)) {
        # Derivatives (Jacobians)
        Ft <- matrix(c(zt[2], 0, zt[1], 1), ncol = 2)  # F_t-1
        # Ht does not change

        ## Prediction step
        zt = c(zt[2] * zt[1], zt[2])  #z_t|t-1 f(z_t-1|t-1)
        Pt = Ft %*% Pt %*% t(Ft) + Rv  #P_t|t-1

        ## Update step
        res = y[t] - zt[1]  # the residual at time t
        St = Ht %*% Pt %*% t(Ht) + Re  # innovation covariance
        Kt = Pt %*% t(Ht) %*% St^-1  # Kalman gain
        zt = zt + Kt * res  # z_t|t
        Pt = (diag(2) - Kt %*% Ht) %*% Pt  #P_t|t

        ## Keep the state estimate
        Z[t + 1, ] <- zt
        ## Keep the P[2,2], which is the variance of the estimate of a
        aVar[t + 1] <- Pt[2, 2]
    }
    return(list(zt = zt, Pt = Pt, Rv = Rv, aVar = aVar, Z = Z))
}
```

Check for converges in worst case



a = 0.5

state	sigma_v^2	sigma_a	a mean	a sd	a_var mean	a_var sd
1	10	1	0.2227370	0.0204999	0.0152562	0.0004328
2	1	1	0.3955557	0.0292888	0.0008427	0.0000681
3	10	10	0.2188578	0.0208357	0.0154694	0.0004450
4	1	10	0.3955225	0.0292323	0.0008432	0.0000693

a = -0.5

state	sigma_v^2	sigma_a	a mean	a sd	a_var mean	a_var sd
1	10	1	0.2227370	0.0204999	0.0152562	0.0004328
2	1	1	0.3955557	0.0292888	0.0008427	0.0000681
3	10	10	0.2188578	0.0208357	0.0154694	0.0004450
4	1	10	0.3955225	0.0292323	0.0008432	0.0000693

hvordan påvirker størrelsen ad sigma_v2 og hvordan påvirkes

variance of the system?

Improvements

do regulizing of the sigma vector.. add some to the diagonal