Econometrics Assignment

Marcell Granát

2020 10 20

Introduction

This short paper focuses on the comparison of the Akaike information criterion (AIC) and the Bayesian information criterion (BIC) for estimating ARMA models. The setup of these models are based on the Box-Jenkins method, which were publicated in 1970. The conditions to accept an ARMA model is that the data generetic process should be stationer (in this study I restricted the research question to the ARMA models, so the DGP is time-invariant in all case, and I ignored this issue) and the remaining part must be white noise (traditionally tested with Ljung-Box-test, but I also did not focus on this now). In some cases, when we face with a pure autoregressive (AR) or with a pure moving average (MA) model we can read out the the order of model from the collerogram (ACF and PACF). Although, if the DGP is a mix of these previous two, then the ACF and the PACF become messy and the estimation is complex.

In these situation the methodology is that we estimate a set of possible models and calculate one of the information criteria to them, finally we choose the one with smallest error. But these criterias are complex functions of the error term and the number of the parameters. It is an opened question that which one we should choose. In this paper I aim to answer this question relying on simulations, while controlling for number of observations and existence of noise.

Pure AR(1) and MA(1)

Figure 1 presents that although model identification based on AIC outperforms the identification based on BIC in the case when the parameter (θ or ϕ) is low, in other cases, BIC gives a better estimation for the DGP. The reverse U-shape of accuracy is interesting. It occurs because at a low level of θ or ϕ the DGP is too close to white noise (Since AIC is one of the information criteria, suggesting the more complex model most frequently, its advantage here is understandable), while at a high level taking the difference of the series looks to be the right decision.

Estimating in the presence of noise

Although the model identification may be correct at a high rate with some conditions (the parameter is not too small or too high), in the real world we can observe the DGPs only with noise. In an empirical study, noise means a random effect, with zero expected value and some variance. To implement this into my investigation I generate random series from a normal distribution and add to the previously simulated ARMA models. In the following, I will refer to the standard deviation of the noise as the size of the noise. The method is visualized in figure 2.

Beside ARMA parameters and noise, the sample size is a relevant input to control. Hence, in the following analysis I will fix θ and ϕ at 0.5 (since the rate of correct model identification was the highest close to these values), and performed the previous simulation at different sample size and different noise variance. Figure 3 shows the correct identification rate based on AIC in the case of pure models with several conditions, while

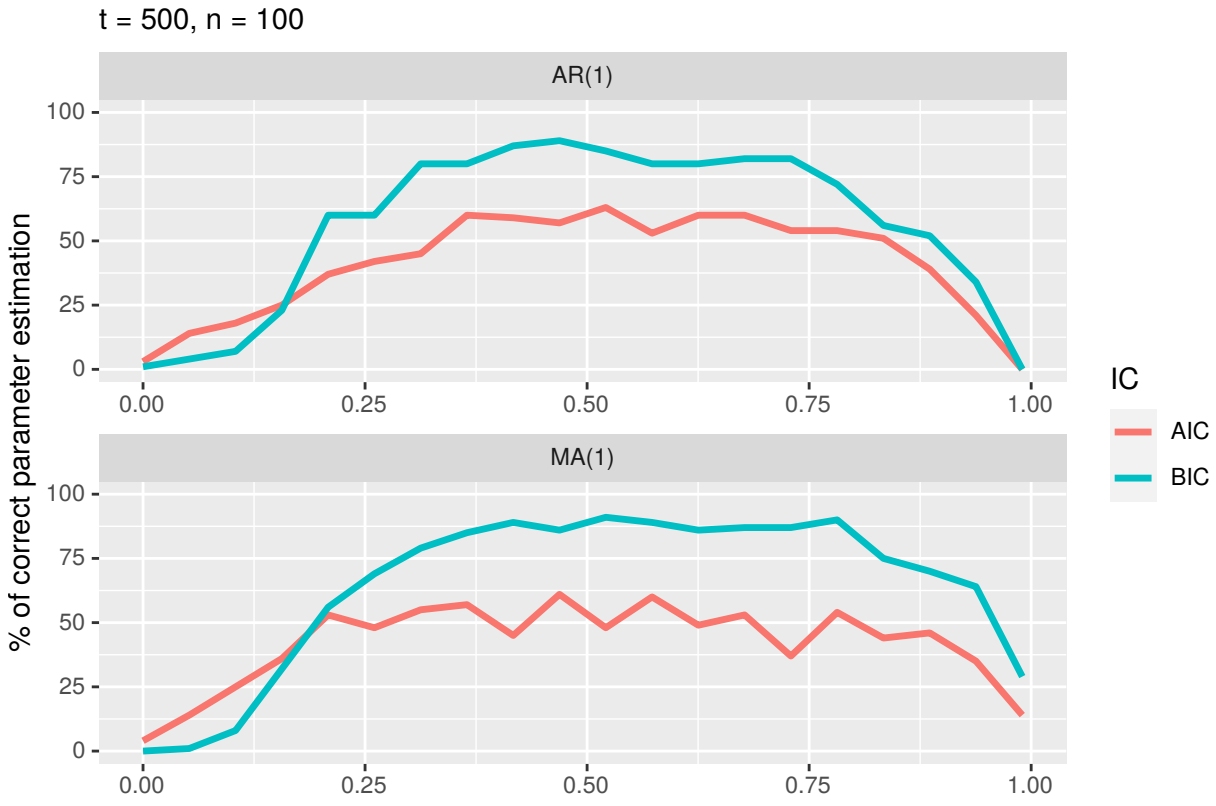

Figure 1: Simulation and estimation of pure models with different parameters

figure 4 presents the rate based on AIC minus the rate based on BIC. As a conclusion, the advantage of AIC is positively correlated with the size of noise, and negatively with the length of the series.

Estimating complex models

In the previous chapter, I wrote about pure models, here I focus on complex models, where not only one AR or MA parameter exist. Since the simulation is highly computation consuming, I restrict the analysis to the following four models: ARMA (3, 0), ARMA (0, 3), ARMA (1, 1), ARMA (3, 3). Where 3 parameters are given, the values are chosen to 0.3, 0.2, 0.1. The result is shown in figure 5 and 6, similarly like above: correct identification rate based on AIC (Figure 5) and the advantage of AIC estimation to BIC (Figure 6). The problem of this analysis is that the values to parameters (0.3, 0.2, 0.1) are too low, and the number of correct identifications is close to zero. A repeat of the simulation with higher values is necessary. However, ARMA(1, 1) model is still useful and confirms the conclusion of the analysis of pure models: **the advantage of AIC to BIC is positively correlated with the size of noise, and negatively with the length of the series.**

Appendix: All code for this report

```
ts.sim <- function(t = 200, n = 10, theta = NULL, phi = c(0.5), noise = 0) {
  dat <- matrix(nrow = t, ncol = n)
  for (i in 1:n) {
    dat[, i] <- arima.sim(list(ma = theta, ar = phi), n = t) # arima simulation
```

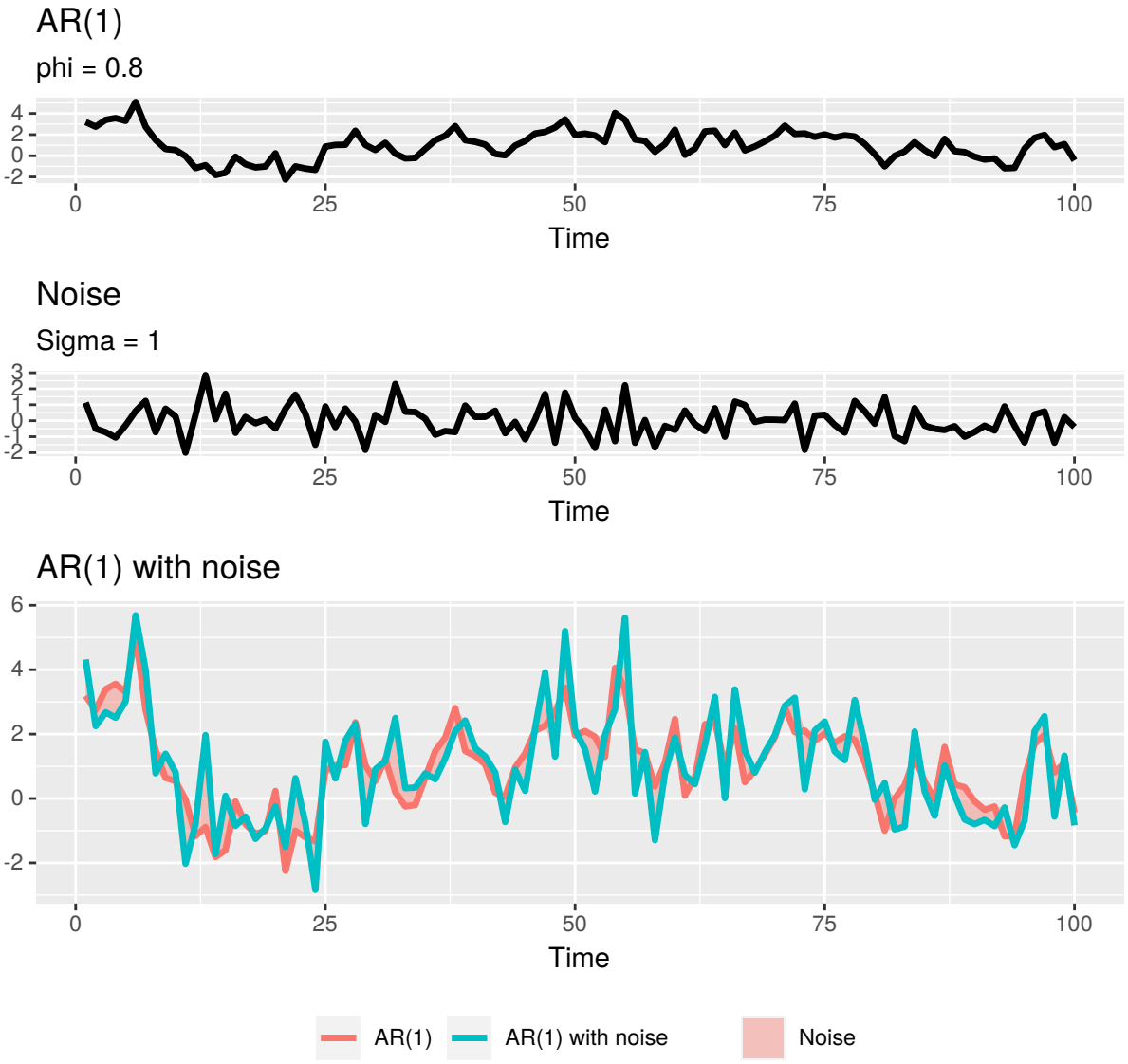

Figure 2: Methodology of ARMA simulation with noise

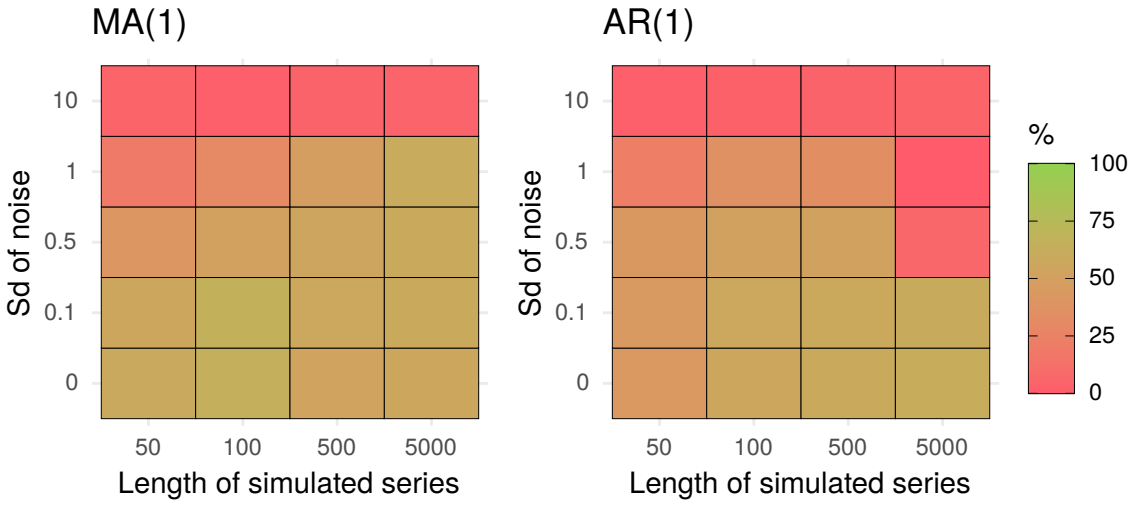

Figure 3: Simulation and estimation pure models with different sample size and presence of noise based on AIC

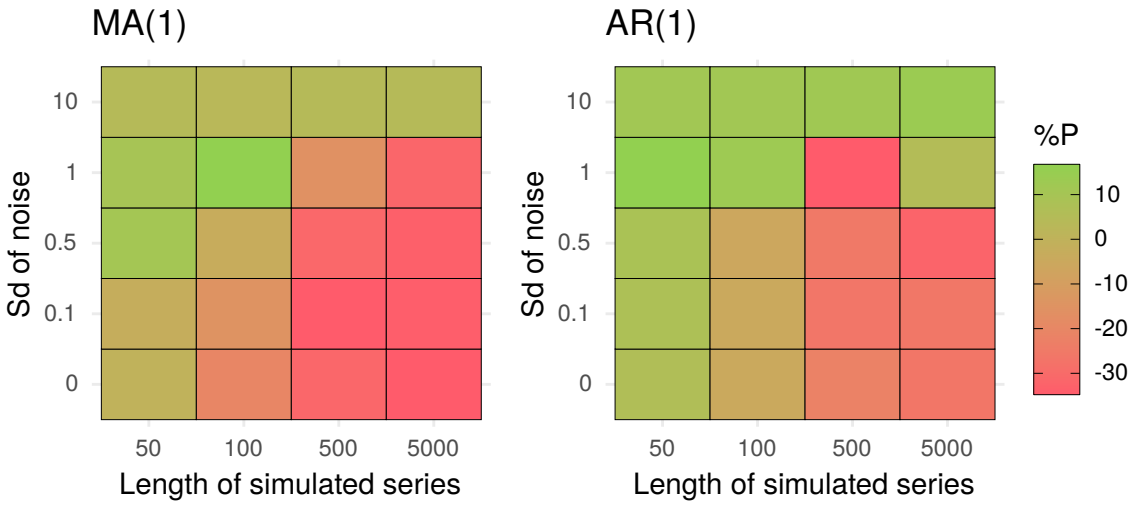

Figure 4: Difference between the accuracy of estimation with AIC and BIC

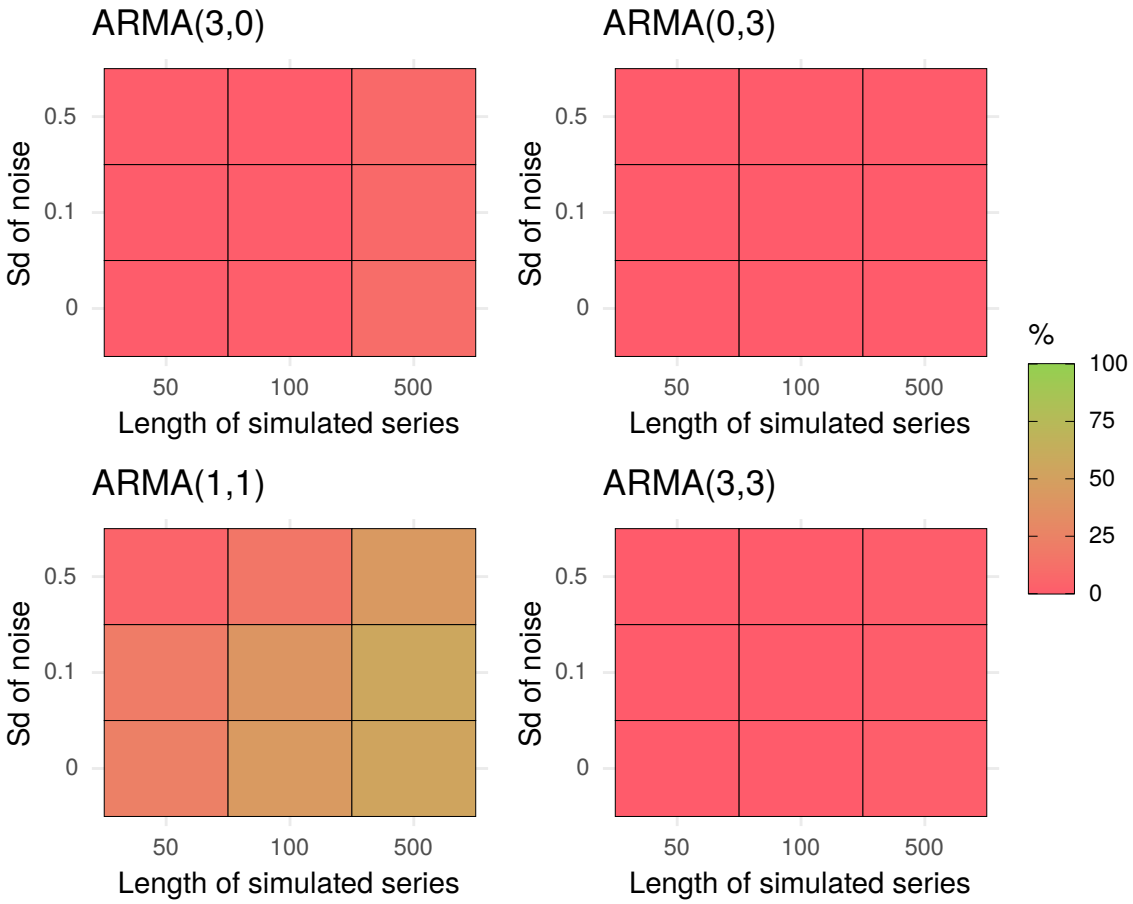

Figure 5: Simulation and estimation complex DGPs based on AIC

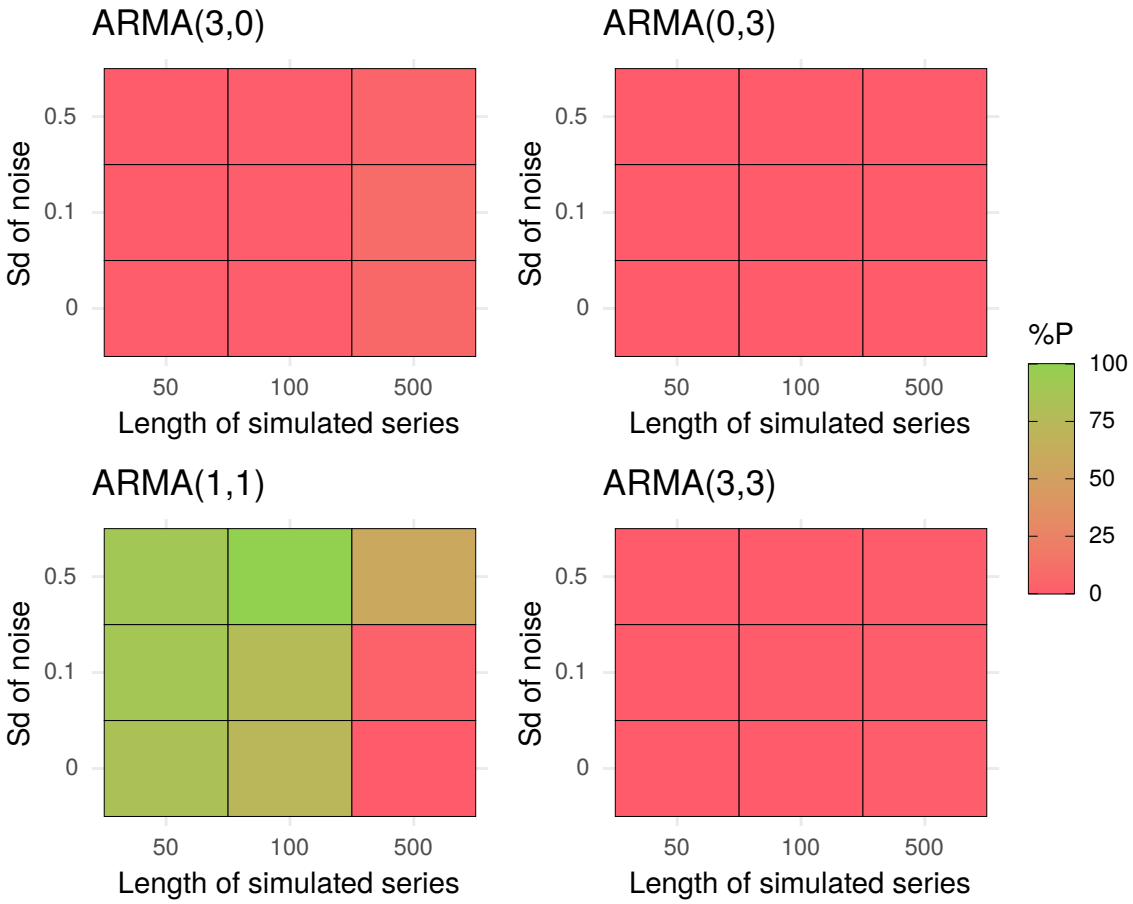

Figure 6: Difference between the accuracy of estimation with AIC and BIC in the case of complex DGPs

```
  }
  dat <- dat + matrix(rnorm(sd = noise, n = t * n), ncol = n) # add noise
  return(dat)
}

ACER <- function(t = 200, n = 10, theta = c(0.8), phi = NULL, noise = 0, ic = "aic") {
  # ACER = ARMA Correct Estimation Rate
  dat <- ts.sim(t = t, n = n, theta = theta, phi = phi, noise = noise) # DGP simulation
  corrects <- vector() # collector vector: 1 if correct, 0 if false
  for (i in seq(ncol(dat))) {
    model <- as.vector(dat[, i]) %>%
      ts() %>%
      auto.arima(trace = F, ic = ic) # model estimation
    corrects[i] <- ifelse(length(model$model$phi) == length(phi) &
    length(model$model$theta) == length(theta) & length(model$model$Delta) == 0, 1, 0)
  }
  mean(corrects) * 100
}

ACER.table <- function(t = c(100, 1000, 5000), n = 10, theta = c(0.8), phi = NULL,
                       noise = c(0, 1, 5, 10, 100), ic = c("aic")) {
  df <- expand.grid(t, noise) %>% set_names(c("t", "noise"))
        # all combination of the vectors
  ACERs <- vector()
  for (i in seq(nrow(df))) {
    ACERs[i] <- ACER(t = df$t[i], n = n, theta = theta, phi = phi, noise = df$noise[i],
                     ic = ic[1])
  }
  if (length(ic) == 2) { # the difference of accuracy if more than 1 ic is given
    for (i in seq(nrow(df))) {
      ACERs[i] <- ACERs[i] - ACER(t = df$t[i], n = n, theta = theta, phi = phi,
                                  noise = df$noise[i], ic = ic[2])
    }
  }
  df$value <- ACERs
  return(df)
}

ACER.plot <- function(t = c(100, 1000, 5000), n = 10, theta = c(0.8), phi = NULL,
                      noise = c(0, 1, 5, 10, 100), ic = c("aic")) {
  df <- ACER.table(t = t, n = n, theta = theta, phi = phi, noise = noise, ic = ic)
      # same as ACER.table, but plot
  p <- df %>%
    mutate(
      t = factor(t),
      noise = factor(noise)
    ) %>%
    ggplot(aes(x = t, y = noise, fill = value)) +
    geom_tile(color = "black") +
    guides(fill = guide_colourbar(ticks.colour = "black", frame.colour = "black",
                                  raster = FALSE)) +
    labs(y = "Sd of noise", x = "Length of simulated series",
         fill = ifelse(length(ic) == 1, "%", "%P")) +
```

```
    theme_minimal()

  if (sum(df$value < 0) == 0) {
    p <- p + scale_fill_gradient(low = "#FF5B6B", high = "#92D050", limits = c(0, 100))
  } else {
    p <- p + scale_fill_gradient(low = "#FF5B6B", high = "#92D050")
  }
  return(p)
}

ARs.aic <- vector() # collector vector
ARs.bic <- vector() # collector vector
MAs.aic <- vector() # collector vector
MAs.bic <- vector() # collector vector
x <- seq(from = 0, to = 0.99, length.out = 20)
for (i in seq_along(x)) {
  ARs.aic[i] <- ACER(t = 500, n = 100, theta = NULL, phi = x[i], noise = 0, ic = "aic")
  ARs.bic[i] <- ACER(t = 500, n = 100, theta = NULL, phi = x[i], noise = 0, ic = "bic")
  MAs.aic[i] <- ACER(t = 500, n = 100, theta = x[i], phi = NULL, noise = 0, ic = "aic")
  MAs.bic[i] <- ACER(t = 500, n = 100, theta = x[i], phi = NULL, noise = 0, ic = "bic")
}
data.frame(x, ARs.aic, ARs.bic, MAs.aic, MAs.bic) %>%
  pivot_longer(-1) %>%
  mutate(
    model = ifelse(str_detect(string = name, pattern = "AR"), "AR(1)", "MA(1)"),
    ic = ifelse(str_detect(string = name, pattern = "aic"), "AIC", "BIC")
  ) %>%
  ggplot(aes(x = x, y = value, color = ic)) +
  geom_line(size = 1.2) +
  scale_y_continuous(limits = c(0, 100)) +
  facet_wrap(~model, nrow = 2, scales = "free") +
  labs(
    color = "IC",
    subtitle = "t = 500, n = 100",
    x = "",
    y = "% of correct parameter estimation"
  )

df <- ts.sim(n = 1, t = 100, theta = 0, phi = 0.8) %>%
  cbind(
    matrix(rnorm(sd = 1, n = 100), ncol = 1)
  ) %>%
  data.frame() %>%
  mutate(X3 = X1 + X2)

ggpubr::ggarrange(
  ggplot(df, aes(x = 1:nrow(df), y = X1)) +
    geom_line(size = 1.2) +
    labs(x = "Time", y = "", title = "AR(1)", subtitle = "phi = 0.8"),

  ggplot(df, aes(x = 1:nrow(df), y = X2)) +
    geom_line(size = 1.2) +
    labs(x = "Time", y = "", title = "Noise", subtitle = "Sigma = 1"),
```

```
  ggplot(df) +
    geom_ribbon(aes(
      x = 1:nrow(df), ymin = ifelse(df$X1 < df$X3, df$X1, df$X3),
      ymax = ifelse(df$X1 > df$X3, df$X1, df$X3), fill = "Noise"
    ), alpha = 0.4) +
    geom_line(aes(x = 1:nrow(df), y = X1, color = "AR(1)"), size = 1.2) +
    geom_line(aes(x = 1:nrow(df), y = X3, color = "AR(1) with noise"), size = 1.2) +
    labs(x = "Time", y = "", title = "AR(1) with noise", color = "", fill = "") +
    theme(legend.position = "bottom"),
  ncol = 1, heights = c(1, 1, 2)
)

ggpubr::ggarrange(
  ACER.plot(t = c(50, 100, 500, 5000), n = 1000, theta = c(0.5), phi = NULL,
            noise = c(0, 0.1, 0.5, 1, 10), ic = c("aic")) + ggtitle("MA(1)"),
  ACER.plot(t = c(50, 100, 500, 5000), n = 1000, theta = NULL, phi = c(0.5),
            noise = c(0, 0.1, 0.5, 1, 10), ic = c("aic")) + ggtitle("AR(1)"),
  ncol = 2, common.legend = T, legend = "right"
)

ggpubr::ggarrange(
  ACER.plot(t = c(50, 100, 500, 5000), n = 1000, theta = c(0.5), phi = NULL,
            noise = c(0, 0.1, 0.5, 1, 10), ic = c("aic", "bic")) + ggtitle("MA(1)"),
  ACER.plot(t = c(50, 100, 500, 5000), n = 1000, theta = NULL, phi = c(0.5),
            noise = c(0, 0.1, 0.5, 1, 10), ic = c("aic", "bic")) + ggtitle("AR(1)"),
  ncol = 2, common.legend = T, legend = "right"
)

ggpubr::ggarrange(
  ACER.plot(t = c(50, 100, 500), n = 1000, theta = c(0.3, 0.2, 0.1), phi = NULL,
            noise = c(0, 0.1, 0.5), ic = c("aic")) + ggtitle("ARMA(3,0)"),
  ACER.plot(t = c(50, 100, 500), n = 1000, theta = NULL, phi = c(0.3, 0.2, 0.1),
            noise = c(0, 0.1, 0.5), ic = c("aic")) + ggtitle("ARMA(0,3)"),
  ACER.plot(t = c(50, 100, 500), n = 1000, theta = c(0.5), phi = c(0.5),
            noise = c(0, 0.1, 0.5), ic = c("aic")) + ggtitle("ARMA(1,1)"),
  ACER.plot(t = c(50, 100, 500), n = 1000, theta = c(0.3, 0.2, 0.1), phi = c(0.3, 0.2, 0.1),
            noise = c(0, 0.1, 0.5), ic = c("aic")) + ggtitle("ARMA(3,3)"),
  ncol = 2, nrow = 2, common.legend = T, legend = "right"
)

ggpubr::ggarrange(
  ACER.plot(t = c(50, 100, 500), n = 1000, theta = c(0.3, 0.2, 0.1), phi = NULL,
            noise = c(0, 0.1, 0.5), ic = c("aic", "bic")) + ggtitle("ARMA(3,0)"),
  ACER.plot(t = c(50, 100, 500), n = 1000, theta = NULL, phi = c(0.3, 0.2, 0.1),
            noise = c(0, 0.1, 0.5), ic = c("aic", "bic")) + ggtitle("ARMA(0,3)"),
  ACER.plot(t = c(50, 100, 500), n = 1000, theta = c(0.5), phi = c(0.5),
            noise = c(0, 0.1, 0.5), ic = c("aic", "bic")) + ggtitle("ARMA(1,1)"),
  ACER.plot(t = c(50, 100, 500), n = 1000, theta = c(0.3, 0.2, 0.1), phi = c(0.3, 0.2, 0.1),
            noise = c(0, 0.1, 0.5), ic = c("aic", "bic")) + ggtitle("ARMA(3,3)"),
  ncol = 2, nrow = 2, common.legend = T, legend = "right"
)
```