Autoregressive model estimated with maximum likelihood

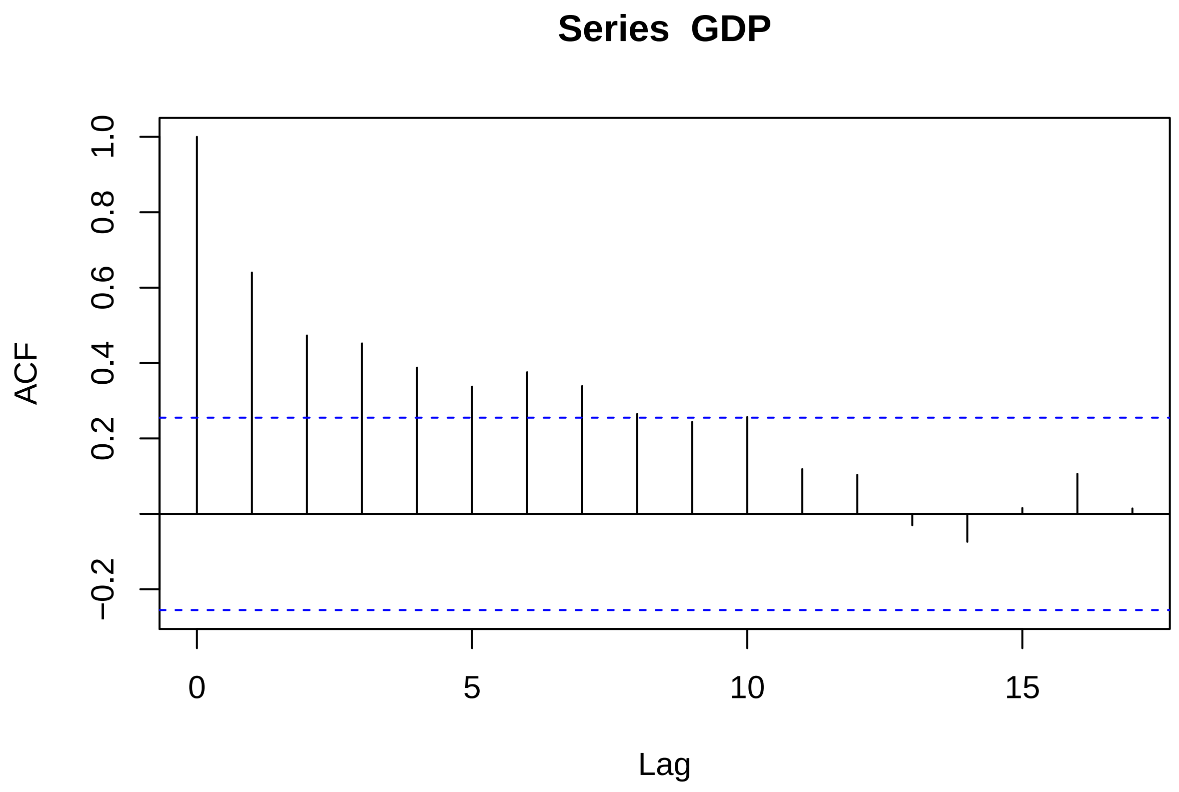
Estimation of AR(p) models

In this short note, we show how to estimate AR(p) models using the OLS estimator and the maximum likelihood approach. In particular, we shall do the following:

1. Estimate an AR(p) model with the OLS method.
2. Estimate an AR(p) model with the maximum likelihood approach.
3. Compare the estimates and their standard deviations.

In this document, we use the GDP growth rate of France to illustrate the two approaches. We first load the data. Then, we will estimate an AR(2) model using both estimation techniques.

```
url = "https://raw.githubusercontent.com/adufays/GDP_expectancy/main/france-gdp-growth-rate.csv"
data = read.csv(url, sep=",")
GDP = data[,2]
autocorr = acf(GDP)
```



```
(autocorr)
```

```
##
## Autocorrelations of series 'GDP', by lag
```

```
##
##      0      1      2      3      4      5      6      7      8      9     10
##  1.000  0.640  0.473  0.452  0.388  0.338  0.376  0.339  0.265  0.244  0.257
##     11     12     13     14     15     16     17
##  0.119  0.104 -0.030 -0.074  0.015  0.106  0.014
```

We observe that the series exhibits some autocorrelation effects.

Estimation of an AR(p) model using OLS

We use the OLS estimator to estimate an AR(2) model.

```
## OLS estimation
T = length(GDP)
nb_p = 2

y = GDP[(nb_p+1):T]
X = array(1,dim=c(T-nb_p,nb_p+1))
for(i in 1:nb_p){
  X[,i] = GDP[(nb_p+1-i):(T-i)]
}

T = length(y)
beta_ols = solve(t(X)%*%X)%*%t(X)%*%y
residuals = y - X%*%beta_ols
estimated_var = sum(residuals^2)/(T-length(beta_ols))
var_beta = solve(t(X)%*%X)*estimated_var
(beta_ols)
```

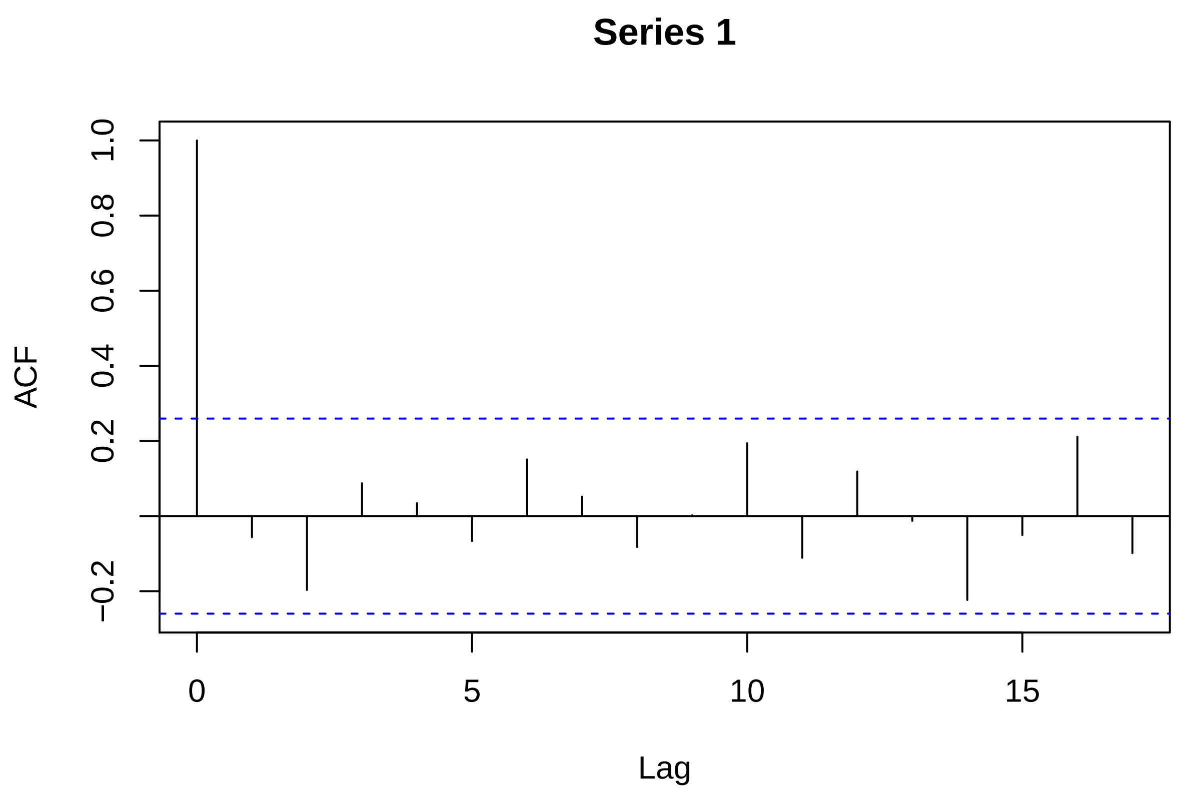
```
##           [,1]
## [1,] 0.5286290
## [2,] 0.1399668
## [3,] 0.8167162
```

```
(var_beta)
```

```
##             [,1]        [,2]        [,3]
## [1,]  0.01720056 -0.01104492 -0.01643626
## [2,] -0.01104492  0.01693727 -0.01727148
## [3,] -0.01643626 -0.01727148  0.13665121
```

Note that for the estimation, we had to delete the first two observations to be able to build the explanatory variables. We now check the residuals for the presence of autocorrelations.

```
## check residuals
acf(residuals)
```



The residuals do not seem to be autocorrelated.

Estimation of an AR(p) model using the maximum likelihood approach

For estimating the AR model using the ML approach, we need to write down the conditional likelihood function (see function *log_likelihood_AR*).

```
## Log likelihood function:
log_likelihood_AR = function(beta_and_sigma_sq,y,X){
  dim_param = length(beta_and_sigma_sq)
  sigma_sq = beta_and_sigma_sq[dim_param]
  beta = beta_and_sigma_sq[1:(dim_param-1)]
  if(sigma_sq<0){
    LL = -10e8
  }else{
    LL = 0
    T = length(y)
    residuals = y - X%*%beta
    for(t in 1:T){
      LL = LL -0.5*log(2*pi*sigma_sq) - (residuals[t]^2)/(2*sigma_sq)
    }
  }
  return(-1*LL)
}
```

Once the function has been coded, we can maximize the log-likelihood function using a optimization routine from R. To do so, we need to fix starting values and then, the optimization method will try to find the global maximum. As most optimization functions in R (and in any other languages) try to find the global minimum of a function, we have multiplied the log-likelihood (LL) function by -1 (see the *return* line in function *log_likelihood_AR*).

```
beta_and_sigma_sq0=c(0.1,0.4,0.2,0.2)
MLE_regression = optim(beta_and_sigma_sq0,log_likelihood_AR,y=y,X=X,method=c('BFGS'),hessian=TRUE)

df_beta_all = data.frame(rbind(c(beta_ols,estimated_var),MLE_regression$par))
colnames(df_beta_all) = c("beta0","beta1","beta2","var")
rownames(df_beta_all) = c("OLS","MLE")
(df_beta_all)
```

```
##         beta0     beta1     beta2      var
## OLS 0.5286290 0.1399668 0.8167162 2.411100
## MLE 0.5286278 0.1399678 0.8167156 2.284212
```

We see that the both estimates for the mean parameters are identical. However, the variance estimate is not the same even though asymptotically both are consistent. In fact, the maximum likelihood estimate of the variance is equal to

$$\hat{\sigma}^2_{\text{MLE}} = \frac{1}{T}\sum_{t=1}^{T}\hat{\epsilon}^2_t.$$

We end this exercise by comparing the standard errors of our estimators. Again, we observe that both techniques provide similar results.

```
## Comparison of the standard errors of the two estimators
inv_hess = solve(MLE_regression$hessian)
std_beta = sqrt(diag(inv_hess))
std_beta_ols = sqrt(diag(var_beta))

df_std_all = data.frame(rbind(std_beta_ols,std_beta[1:length(beta_ols)]))
colnames(df_std_all) = c("beta0","beta1","beta2")
rownames(df_std_all) = c("OLS","MLE")
(df_std_all)
```

```
##         beta0     beta1     beta2
## OLS 0.1311509 0.1301433 0.3696636
## MLE 0.1276532 0.1266725 0.3598050
```