MATH 4070: Regression with Time Series Errors, Unit Root Test, Spurious Correlation and Prewhitening

Whitney Huang, Clemson University

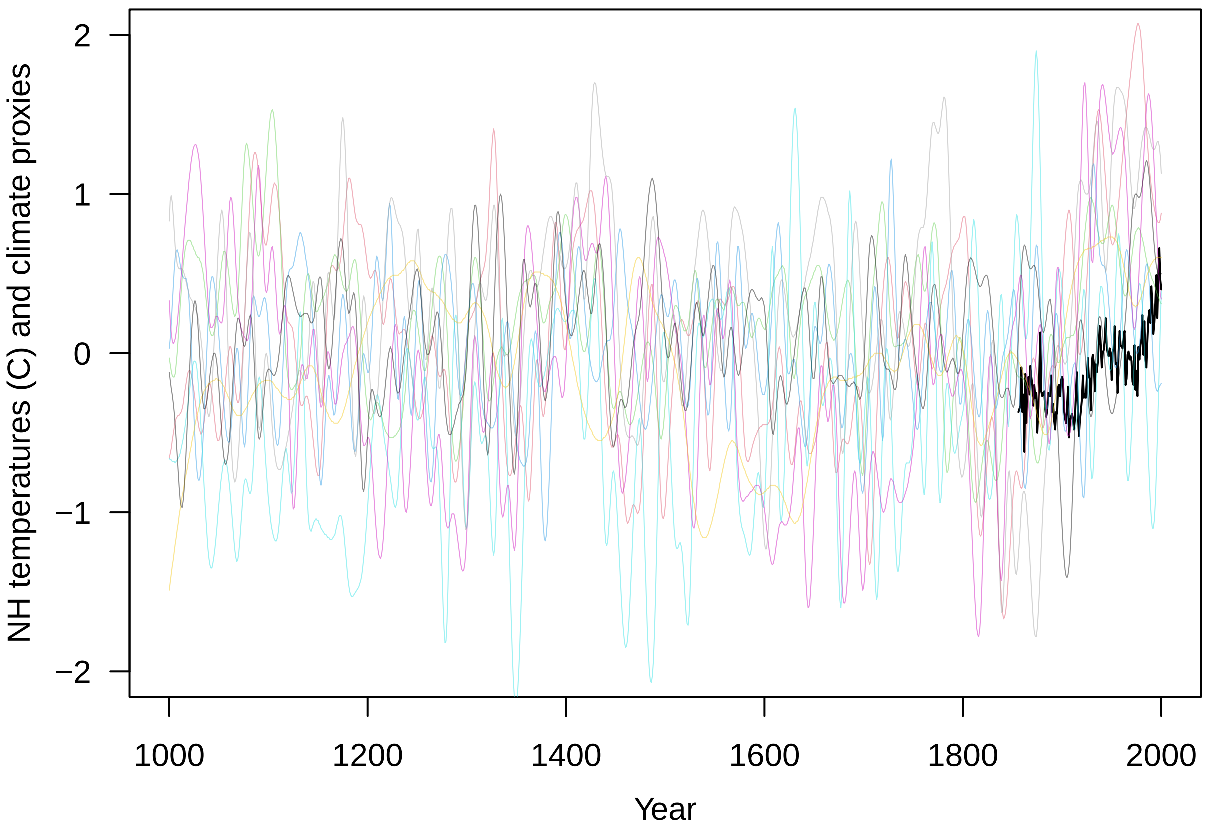
Contents

Regression with Time Series Errors

Let us present a brief data analysis of Northern Hemisphere temperatures and tree ring proxies (Jones and Mann (2004)) to illustrate regression with time series errors.

Plot the data

```
library(faraway)
data(globwarm)
par(las = 1, mgp = c(2.4, 1, 0), mar = c(3.6, 4, 1, 0.6))
plot(globwarm$year, globwarm$nhtemp, type = "l", ylim = c(-2, 2),
     xlab = "Year", ylab = "NH temperatures (C) and climate proxies")
library(scales)
for (i in 2:9) lines(globwarm$year, globwarm[, i], col = alpha(i, 0.45), lwd = 0.5)
```



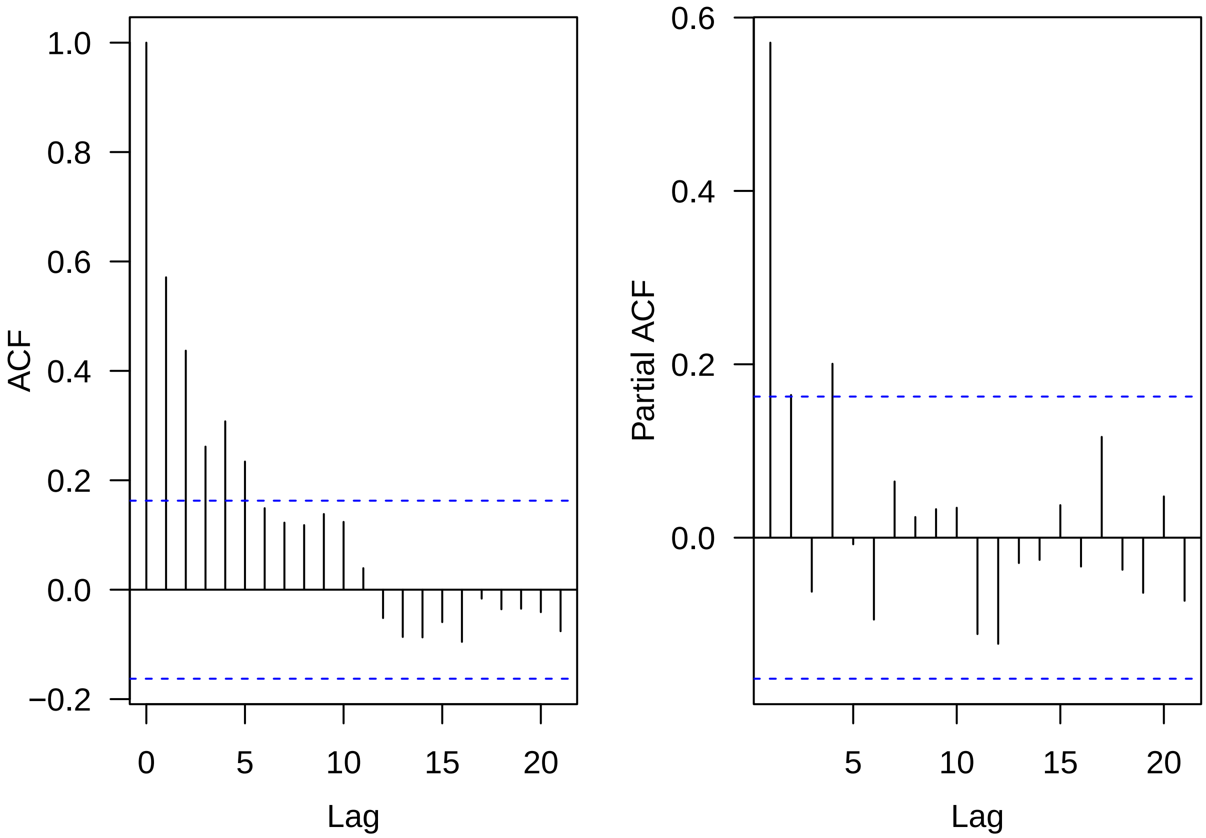
Fit an OLS an examine the residuals

```
lmod <- lm(nhtemp ~ wusa + jasper + westgreen + chesapeake
           + tornetrask + urals + mongolia + tasman, globwarm)
summary(lmod)
```

```
##
## Call:
```

```
## lm(formula = nhtemp ~ wusa + jasper + westgreen + chesapeake +
##     tornetrask + urals + mongolia + tasman, data = globwarm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.43668 -0.11170  0.00698  0.10176  0.65352
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.242555   0.027011  -8.980 1.97e-15 ***
## wusa         0.077384   0.042927   1.803 0.073647 .
## jasper      -0.228795   0.078107  -2.929 0.003986 **
## westgreen    0.009584   0.041840   0.229 0.819168
## chesapeake  -0.032112   0.034052  -0.943 0.347346
## tornetrask   0.092668   0.045053   2.057 0.041611 *
## urals        0.185369   0.091428   2.027 0.044567 *
## mongolia     0.041973   0.045794   0.917 0.360996
## tasman       0.115453   0.030111   3.834 0.000192 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1758 on 136 degrees of freedom
##   (856 observations deleted due to missingness)
## Multiple R-squared:  0.4764, Adjusted R-squared:  0.4456
## F-statistic: 15.47 on 8 and 136 DF,  p-value: 5.028e-16
```

```r
par(las = 1, mgp = c(2.4, 1, 0), mar = c(3.6, 4, 1, 0.6), mfrow = c(1, 2))
acf(lmod$residuals)
pacf(lmod$residuals)
```



Fit a GLS with an AR(1) error structure

```
library(nlme)
glmod <- gls(nhtemp ~ wusa + jasper + westgreen + chesapeake + tornetrask + urals +
               mongolia + tasman, correlation = corAR1(form = ~ year), data = na.omit(globwarm))
summary(glmod)
```

```
## Generalized least squares fit by REML
##   Model: nhtemp ~ wusa + jasper + westgreen + chesapeake + tornetrask +      urals + mongolia + tasma
##   Data: na.omit(globwarm)
##         AIC       BIC   logLik
##   -108.2074 -76.16822 65.10371
##
## Correlation Structure: AR(1)
##  Formula: ~year
##  Parameter estimate(s):
##       Phi
## 0.7109922
##
## Coefficients:
##                   Value  Std.Error   t-value p-value
## (Intercept) -0.23010624 0.06702406 -3.433188  0.0008
## wusa         0.06673819 0.09877211  0.675678  0.5004
## jasper      -0.20244335 0.18802773 -1.076668  0.2835
## westgreen   -0.00440299 0.08985321 -0.049002  0.9610
```

```
## chesapeake  -0.00735289 0.07349791 -0.100042  0.9205
## tornetrask   0.03835169 0.09482515  0.404446  0.6865
## urals        0.24142199 0.22871028  1.055580  0.2930
## mongolia     0.05694978 0.10489786  0.542907  0.5881
## tasman       0.12034918 0.07456983  1.613913  0.1089
##
##  Correlation:
##            (Intr) wusa   jasper wstgrn chespk trntrs urals  mongol
## wusa       -0.517
## jasper     -0.058 -0.299
## westgreen   0.330 -0.533  0.121
## chesapeake  0.090 -0.314  0.230  0.147
## tornetrask -0.430  0.499 -0.197 -0.328 -0.441
## urals      -0.110 -0.142 -0.265  0.075 -0.064 -0.346
## mongolia    0.459 -0.437 -0.205  0.217  0.449 -0.343 -0.371
## tasman      0.037 -0.322  0.065  0.134  0.116 -0.434  0.416 -0.017
##
## Standardized residuals:
##         Min          Q1         Med          Q3         Max
## -2.31122523 -0.53484054  0.02342908  0.50015642  2.97224724
##
## Residual standard error: 0.204572
## Degrees of freedom: 145 total; 136 residual
```

```r
intervals(glmod, which = "var-cov")
```

```
## Approximate 95% confidence intervals
##
##  Correlation structure:
##         lower      est.     upper
## Phi 0.5099757 0.7109922 0.8383747
##
##  Residual standard error:
##     lower      est.     upper
## 0.1540712 0.2045720 0.2716258
```

Comparison of Two-Step and One-Step Estimation Procedures

A Simulated Example

$$y_t = 3 + 0.5x_t + \eta_t,$$

$\eta_t = 0.8\eta_{t-1} + Z_t - 0.4Z_{t-1}$, $Z_t \sim \mathrm{N}(0, 1)$.

Simulated time series data

```r
set.seed(1234)
N = 500; n = 200
x <- rnorm(n, 10, 2)
true_beta <- c(3, 0.5)
mean <- true_beta[1] + true_beta[2] * x
```

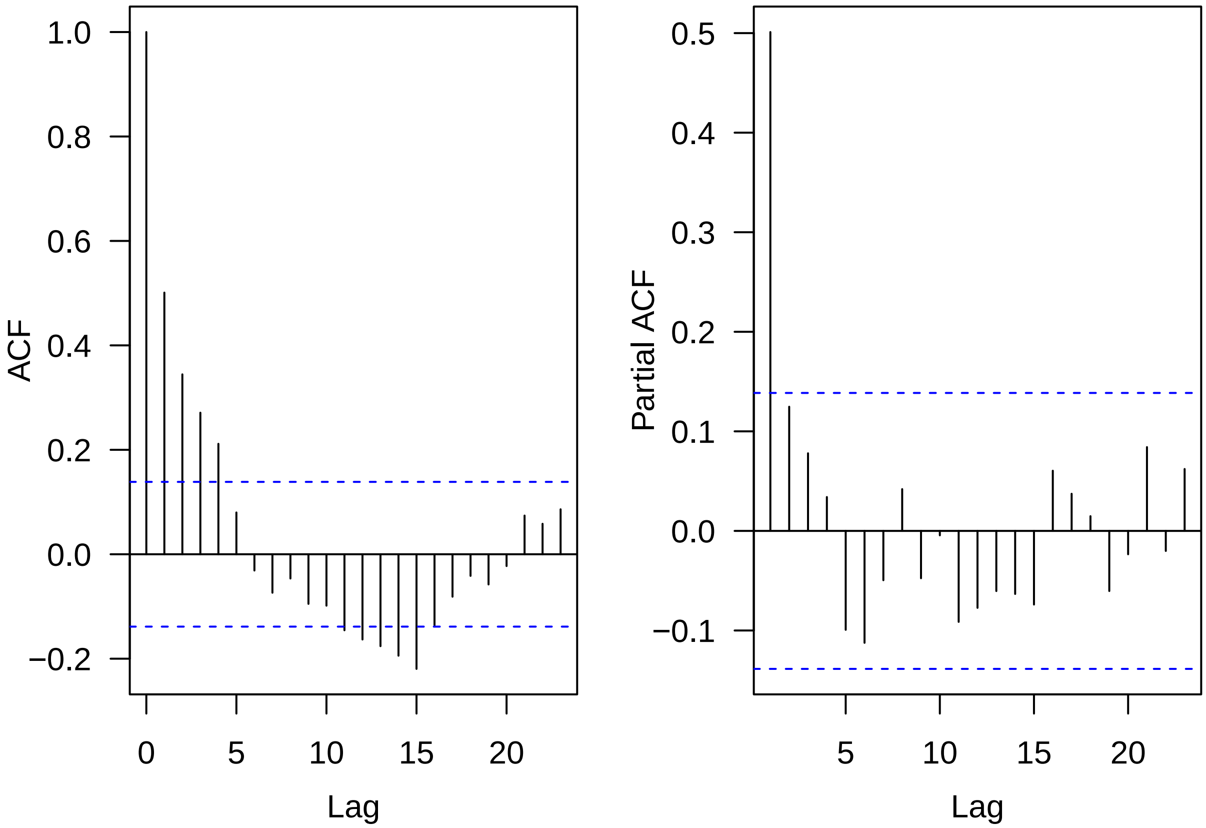
```r
err <- replicate(N, arima.sim(n = n, model = list(ar = 0.8, ma = -0.4), sd = 1))
y <- mean + err
```

Step 1: Perform OLS regression

```r
ols_fit <- apply(y, 2, function(z) lm(z ~ x))
summary(ols_fit[[1]])
```

```
##
## Call:
## lm(formula = z ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5722 -0.7243  0.0514  0.8700  2.9476
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.33404    0.40767   8.178 3.41e-14 ***
## x            0.46400    0.04039  11.487  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.163 on 198 degrees of freedom
## Multiple R-squared:  0.3999, Adjusted R-squared:  0.3969
## F-statistic: 131.9 on 1 and 198 DF,  p-value: < 2.2e-16
```

```r
# Extract residuals
res_ols <- lapply(ols_fit, residuals)
par(las = 1, mgp = c(2.4, 1, 0), mar = c(3.6, 4, 1, 0.6), mfrow = c(1, 2))
acf(res_ols[[1]])
pacf(res_ols[[1]])
```



Step 2: Fit ARMA model to OLS residuals

```
library(forecast) # For ARMA model fitting
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
##
## Attaching package: 'forecast'
```

```
## The following object is masked from 'package:nlme':
##
##     getResponse
```

```
arma_fit <- lapply(res_ols, arima, order = c(1, 0, 1), include.mean = F)
# Extract AR and MA coefficients
phi <- sapply(arma_fit, function(x) x$coef["ar1"])
theta <- sapply(arma_fit, function(x) x$coef["ma1"])
```

Step 3: GLS regression using correlation structure from ARMA model

```r
gls_fit <- list()

for (i in 1:N){
  cor_struct <- corARMA(value = c(phi[i], theta[i]), p = 1, q = 1, form = ~ 1)
  y_each <- y[, i]
  gls_fit[[i]] <- gls(y_each ~ x, correlation = cor_struct, method = "ML")
}

summary(gls_fit[[1]])
```

```
## Generalized least squares fit by maximum likelihood
##   Model: y_each ~ x
##   Data: NULL
##        AIC     BIC    logLik
##   569.2704 585.762 -279.6352
##
## Correlation Structure: ARMA(1,1)
##  Formula: ~1
##  Parameter estimate(s):
##       Phi1     Theta1
##  0.7207913 -0.2723007
##
## Coefficients:
##                 Value Std.Error   t-value p-value
## (Intercept) 2.6285742 0.3652922  7.195813       0
## x           0.5365693 0.0323548 16.583903       0
##
##  Correlation:
##   (Intr)
## x -0.871
##
## Standardized residuals:
##          Min           Q1          Med           Q3          Max
## -3.203649147 -0.566930022  0.006812426  0.699590375  2.632330357
##
## Residual standard error: 1.165508
## Degrees of freedom: 200 total; 198 residual
```

```r
intervals(gls_fit[[1]])
```

```
## Approximate 95% confidence intervals
##
##  Coefficients:
##                lower      est.     upper
## (Intercept) 1.908212 2.6285742 3.3489367
## x           0.472765 0.5365693 0.6003736
##
##  Correlation structure:
##             lower       est.       upper
## Phi1    0.5035786  0.7207913  0.85229748
## Theta1 -0.4895336 -0.2723007 -0.02324316
##
```

```
##  Residual standard error:
##    lower     est.    upper
## 1.009177 1.165508 1.346056
```

One-step MLE

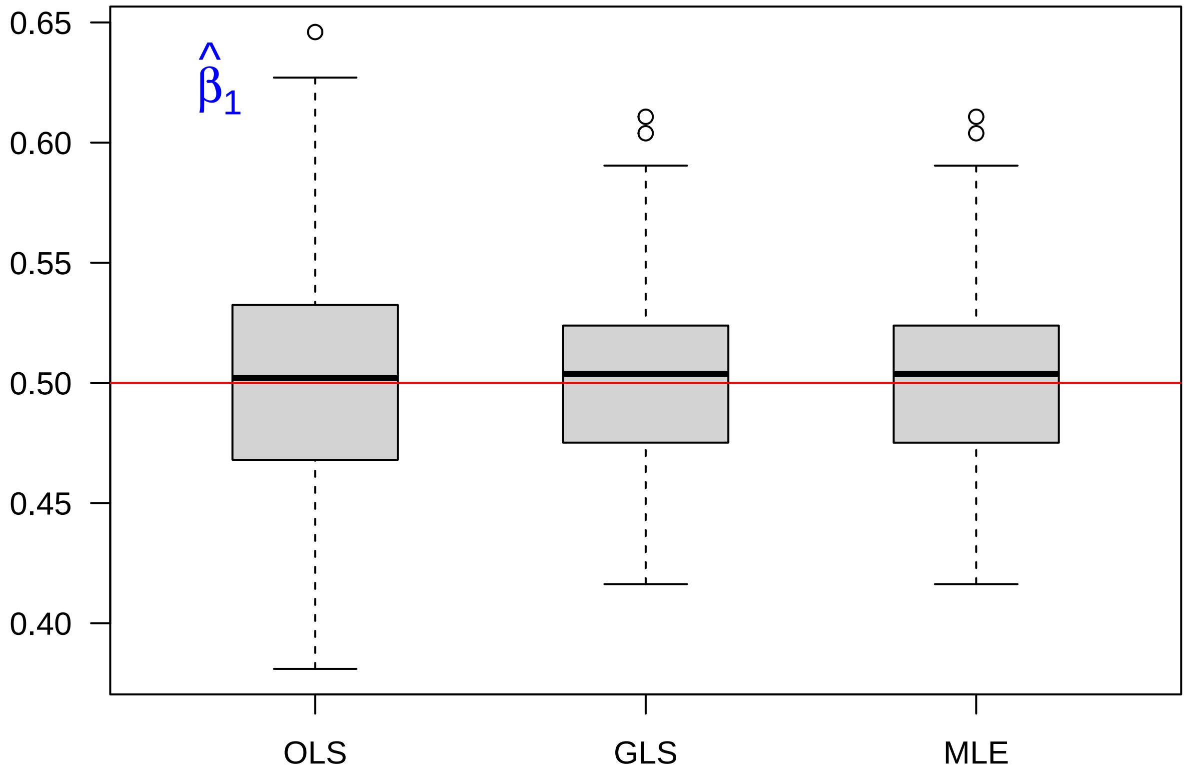
```r
mle <- apply(y, 2, arima, order = c(1, 0, 1), xreg = x)
confint(mle[[1]])
```

```
##                2.5 %      97.5 %
## ar1        0.5505461  0.89103901
## ma1       -0.5088234 -0.03577565
## intercept  1.9116985  3.34531698
## x          0.4728902  0.60025025
```

Summarize the simulation results

```r
beta_ols <- t(sapply(ols_fit, function(x) x$coefficients))
beta_gls <- t(sapply(gls_fit, function(x) x$coefficients))
beta_mle <- t(sapply(mle, function(x) x$coef[3:4]))
```

```r
par(las = 1, mgp = c(2.4, 1, 0), mar = c(3.6, 4, 1, 0.6))
boxplot(beta_ols[, 2], beta_gls[, 2], beta_mle[, 2], xaxt = "n", boxwex = 0.5)
axis(1, 1:3, labels = c("OLS", "GLS", "MLE"))
abline(h = 0.5, col = "red")
legend("topleft", legend = expression(hat(beta)[1]), bty = "n", text.col = "blue",
       cex = 1.5)
```




```
(bias <- c(mean(beta_ols[, 2]), mean(beta_gls[, 2]), mean(beta_mle[, 2])) - 0.5)
```

```
## [1] -0.0004185194  0.0008904009  0.0008902119
```

```
(sd <- c(sd(beta_ols[, 2]), sd(beta_gls[, 2]), sd(beta_mle[, 2])))
```

```
## [1] 0.04647728 0.03465424 0.03465444
```

```
CI_beta_ols <- t(sapply(ols_fit, function(x) confint(x)[2,]))
sum(apply(CI_beta_ols - 0.5, 1, prod) < 0)
```

```
## [1] 454
```

```
mean(apply(CI_beta_ols, 1, diff))
```

```
## [1] 0.1615202
```

```
CI_beta_gls <- t(sapply(gls_fit, function(x) confint(x)[2,]))
sum(apply(CI_beta_gls - 0.5, 1, prod) < 0)
```

```
## [1] 468
```

```
mean(apply(CI_beta_gls, 1, diff))
```

```
## [1] 0.128903
```

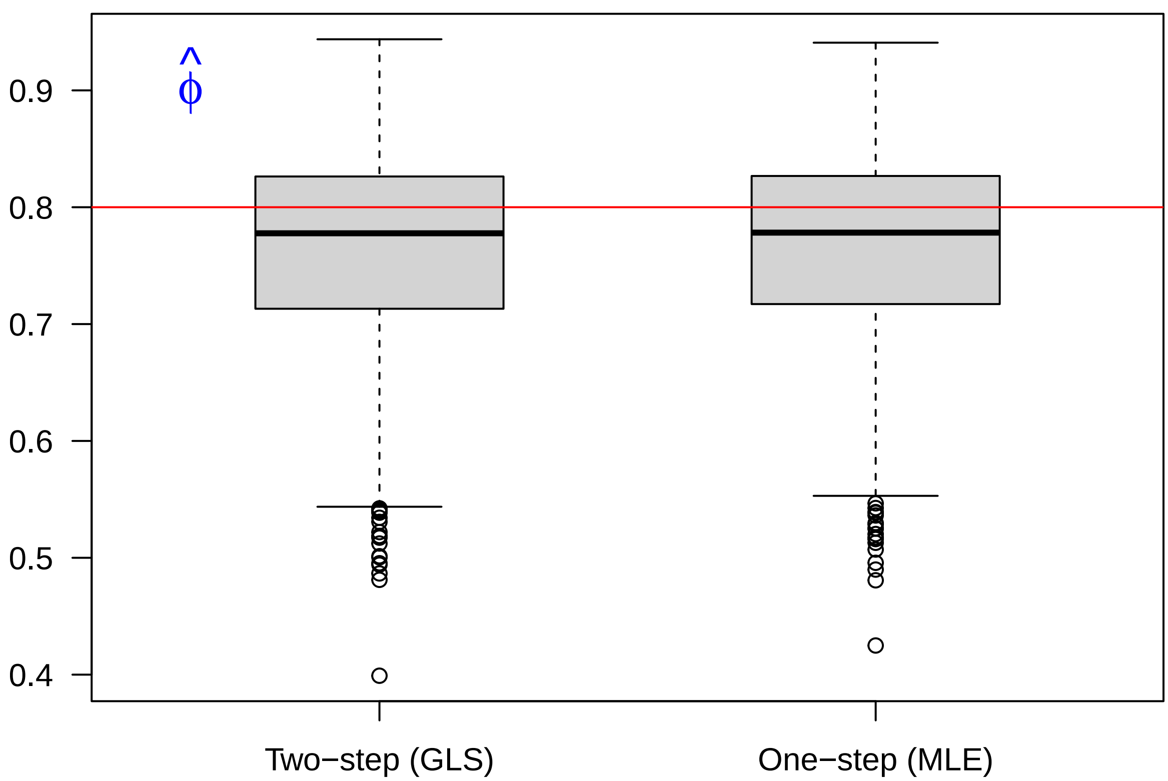
```r
CI_beta_mle <- t(sapply(mle, function(x) confint(x)[4,]))
sum(apply(CI_beta_mle - 0.5, 1, prod) < 0)
```

```
## [1] 468
```

```r
mean(apply(CI_beta_mle, 1, diff))
```

```
## [1] 0.1290107
```

```r
arma_2step <- cbind(phi, theta)
arma_1step <- t(sapply(mle, function(x) x$coef[1:2]))

par(las = 1, mgp = c(2.4, 1, 0), mar = c(3.6, 4, 1, 0.6))
boxplot(arma_2step[, 1], arma_1step[, 1], xaxt = "n", boxwex = 0.5)
axis(1, 1:2, labels = c("Two-step (GLS)", "One-step (MLE)"))
abline(h = 0.8, col = "red")
legend("topleft", legend = expression(hat(phi)), bty = "n", text.col = "blue",
       cex = 1.5)
```




```r
(bias <- c(mean(arma_2step[, 1]), mean(arma_1step[, 1])) - 0.8)
```

```
## [1] -0.03764122 -0.03597188
```

```r
(sd <- c(sd(arma_2step[, 1]), sd(arma_1step[, 1])))
```

```
## [1] 0.08958435 0.08890410
```

```r
CI_phi_mle <- t(sapply(mle, function(x) confint(x)[1,]))
sum(apply(CI_phi_mle - 0.8, 1, prod) < 0)
```

```
## [1] 481
```

```r
mean(apply(CI_phi_mle, 1, diff))
```

```
## [1] 0.3275391
```

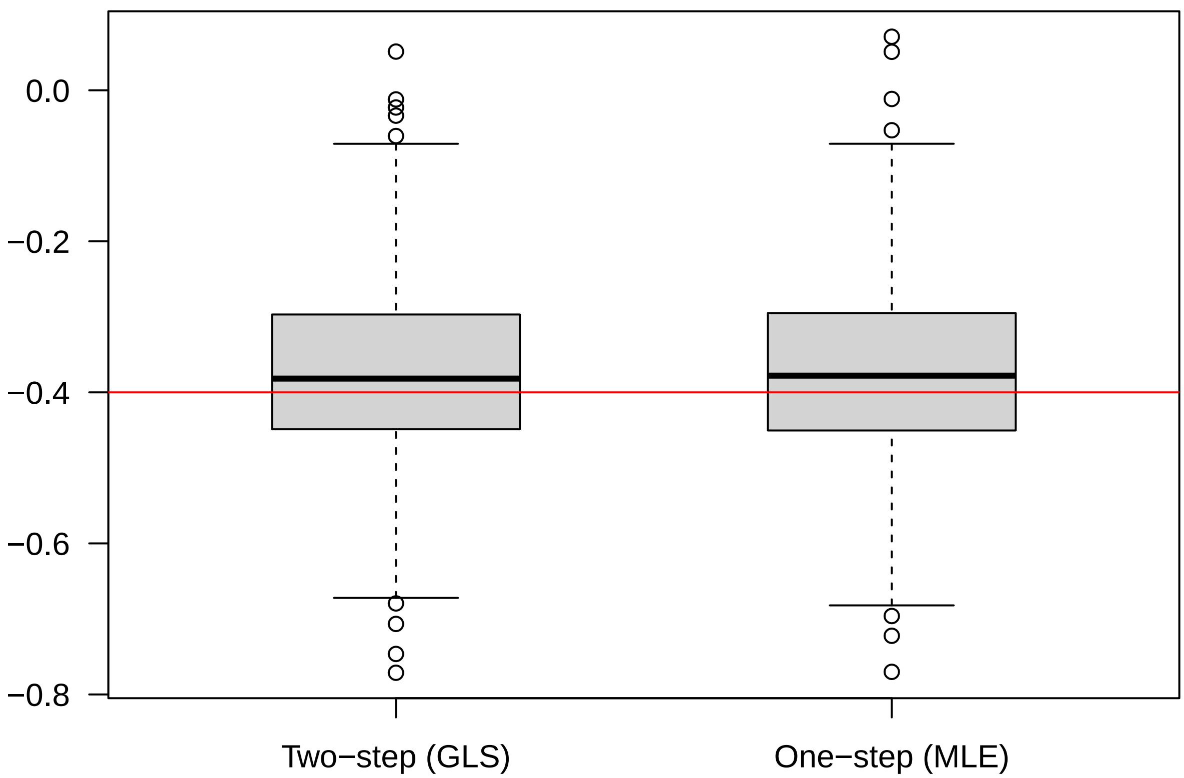
```r
CI_phi_gls <- t(sapply(arma_fit, function(x) confint(x)[1,]))
sum(apply(CI_phi_gls - 0.8, 1, prod) < 0)
```

```
## [1] 483
```

```r
mean(apply(CI_phi_gls, 1, diff))
```

```
## [1] 0.3304249
```

```r
boxplot(arma_2step[, 2], arma_1step[, 2], xaxt = "n", boxwex = 0.5)
axis(1, 1:2, labels = c("Two-step (GLS)", "One-step (MLE)"))
abline(h = -0.4, col = "red")
```




```r
(bias <- c(mean(arma_2step[, 2]), mean(arma_1step[, 2])) - -0.4)
```

```
## [1] 0.02843072 0.03109514
```

```r
(sd <- c(sd(arma_2step[, 2]), sd(arma_1step[, 2])))
```

```
## [1] 0.1223613 0.1222449
```

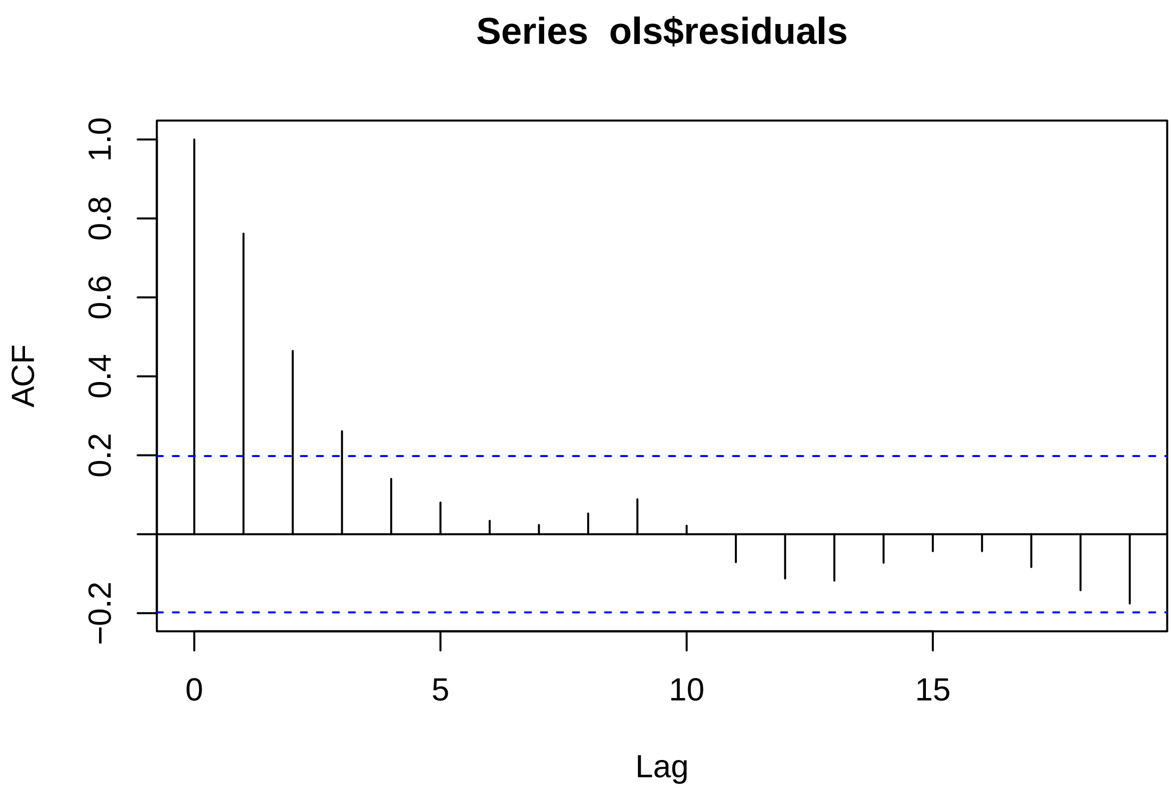
Lake Huron Example

A two-step fit

```r
data(LakeHuron)
years <- time(LakeHuron)
ols<- lm(LakeHuron ~ years)
ols$coefficients
```

```
##  (Intercept)        years
## 625.55491791  -0.02420111
```

```r
acf(ols$residuals)
```




```r
(arma1 <- arima(ols$residuals, order = c(2, 0, 0), include.mean = FALSE))
```

```
##
## Call:
## arima(x = ols$residuals, order = c(2, 0, 0), include.mean = FALSE)
##
```

```
## Coefficients:
##          ar1      ar2
##       1.0050  -0.2925
## s.e.  0.0976   0.1002
##
## sigma^2 estimated as 0.4572:  log likelihood = -101.26,  aic = 208.51
```

One-step MLE fit

```r
yr <- as.numeric(years)
mle <- arima(LakeHuron, order = c(2, 0, 0), xreg = yr, include.mean = T)
```

Comparing CIs

```r
confint(ols)
```

```
##                    2.5 %      97.5 %
## (Intercept) 610.14291793 640.9669179
## years        -0.03221272  -0.0161895
```

```r
confint(arma1)
```

```
##          2.5 %      97.5 %
## ar1  0.8137180  1.19630830
## ar2 -0.4888881 -0.09606208
```

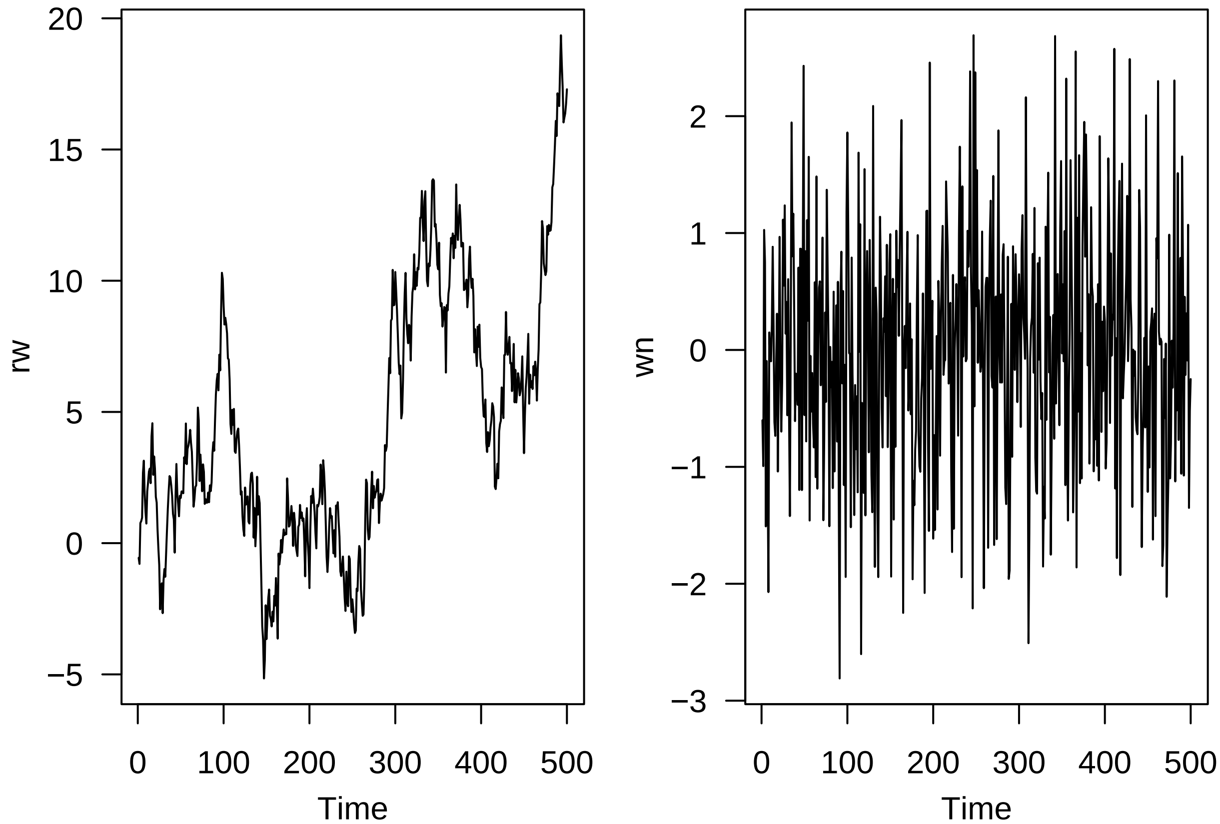
```r
confint(mle)
```

```
##                  2.5 %        97.5 %
## ar1         0.81348340   1.196124084
## ar2        -0.48806617  -0.094573470
## intercept 589.98096094 651.042029064
## yr         -0.03744267  -0.005694985
```

Unit root test examples

OLS

```r
set.seed(123)
rw <- cumsum(rnorm(500))
wn <- rnorm(500)
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6), mfrow = c(1, 2))
ts.plot(rw)
ts.plot(wn)
```



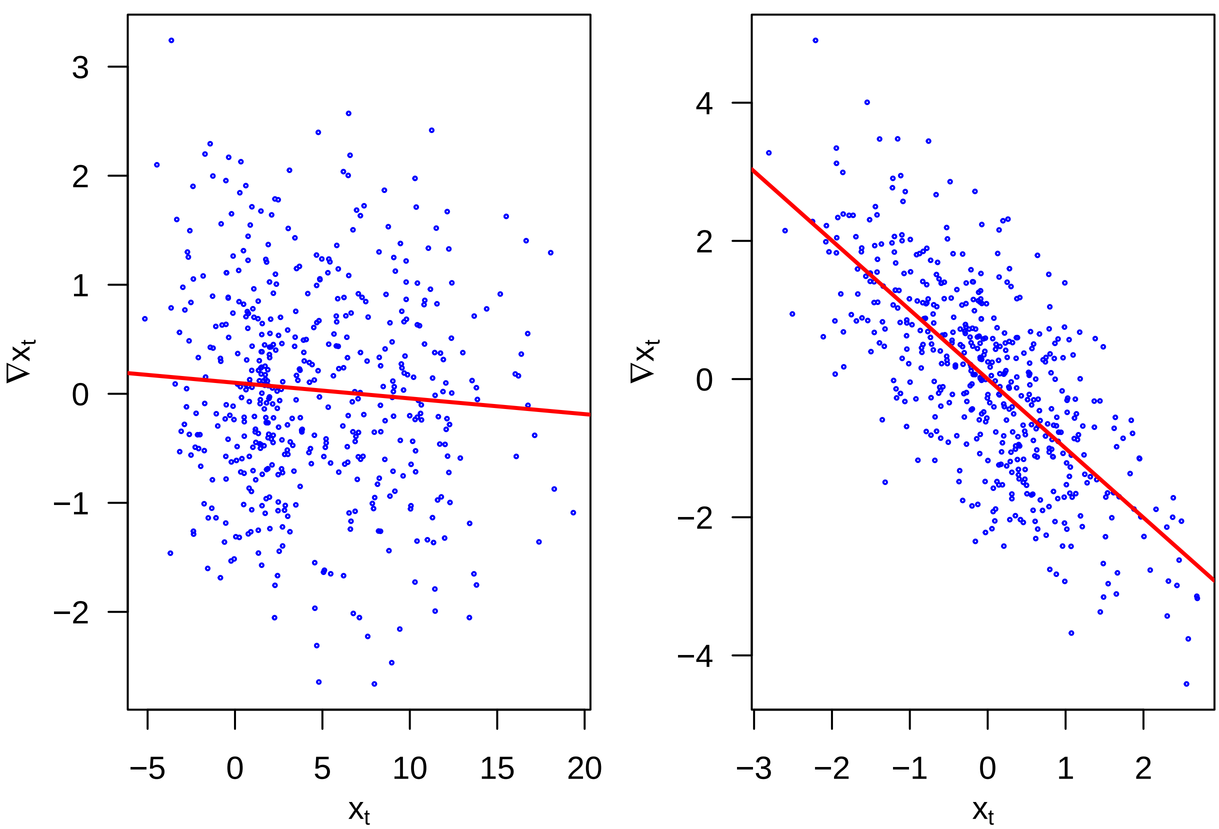

```r
diff.rw <- diff(rw); n <- length(rw)
ys <- diff.rw; xs <- rw[1:(n-1)]
ols.rw <- lm(ys ~ xs); summary(ols.rw)
```

```
##
## Call:
## lm(formula = ys ~ xs)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.67541 -0.62862 -0.01118  0.63805  3.08747
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.10125    0.05973   1.695   0.0906 .
## xs          -0.01438    0.00899  -1.600   0.1102
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9719 on 497 degrees of freedom
## Multiple R-squared:  0.005124,   Adjusted R-squared:  0.003123
## F-statistic:  2.56 on 1 and 497 DF,  p-value: 0.1102
```

```r
diff.wn <- diff(wn)
ys <- diff.wn; xs <- wn[1:(n-1)]
ols.wn <- lm(ys ~ xs); summary(ols.wn)
```

```
##
## Call:
## lm(formula = ys ~ xs)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.81182 -0.69065  0.00075  0.64461  2.68750
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.001138   0.045329  -0.025     0.98
## xs          -1.002420   0.044843 -22.354   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.013 on 497 degrees of freedom
## Multiple R-squared:  0.5014, Adjusted R-squared:  0.5004
## F-statistic: 499.7 on 1 and 497 DF,  p-value: < 2.2e-16
```

```r
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6), mfrow = c(1, 2))
plot(rw[1:length(diff.rw)], diff.rw, xlab = expression(x[t]),
     ylab = expression(paste(nabla, x[t])), cex = 0.25, col = "blue")
abline(ols.rw, col = "red", lwd = 2)
plot(wn[1:length(diff.wn)], diff.wn, xlab = expression(x[t]),
     ylab = expression(paste(nabla, x[t])), cex = 0.25, col = "blue")
abline(ols.wn, col = "red", lwd = 2)
```



ADF

```r
library(tseries)
adf.test(rw)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  rw
## Dickey-Fuller = -1.9203, Lag order = 7, p-value = 0.612
## alternative hypothesis: stationary
```

```r
adf.test(wn)
```

```
## Warning in adf.test(wn): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  wn
## Dickey-Fuller = -7.8953, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

Spurious Correlation and Prewhitening

$$Y_t = X_{t-2} + \varepsilon_t,$$

where $X_t \overset{i.i.d}{\sim} \mathrm{N}(0,1)$, $\varepsilon_t \overset{i.i.d}{\sim} \mathrm{N}(0,0.25)$, and X's and ε's are independent to each other.

```r
library(TSA)
```

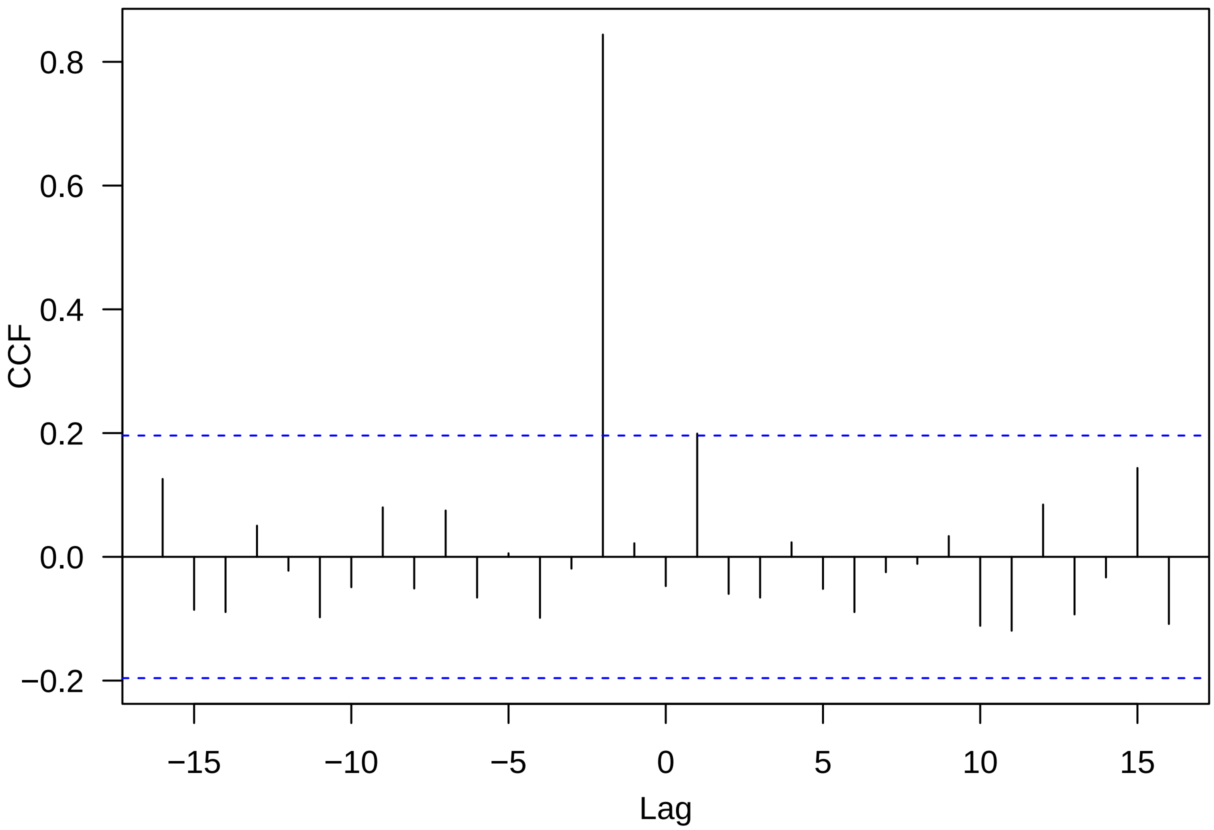
```
## Registered S3 methods overwritten by 'TSA':
##   method       from
##   fitted.Arima forecast
##   plot.Arima   forecast
```

```
##
## Attaching package: 'TSA'
```

```
## The following objects are masked from 'package:stats':
##
##     acf, arima
```

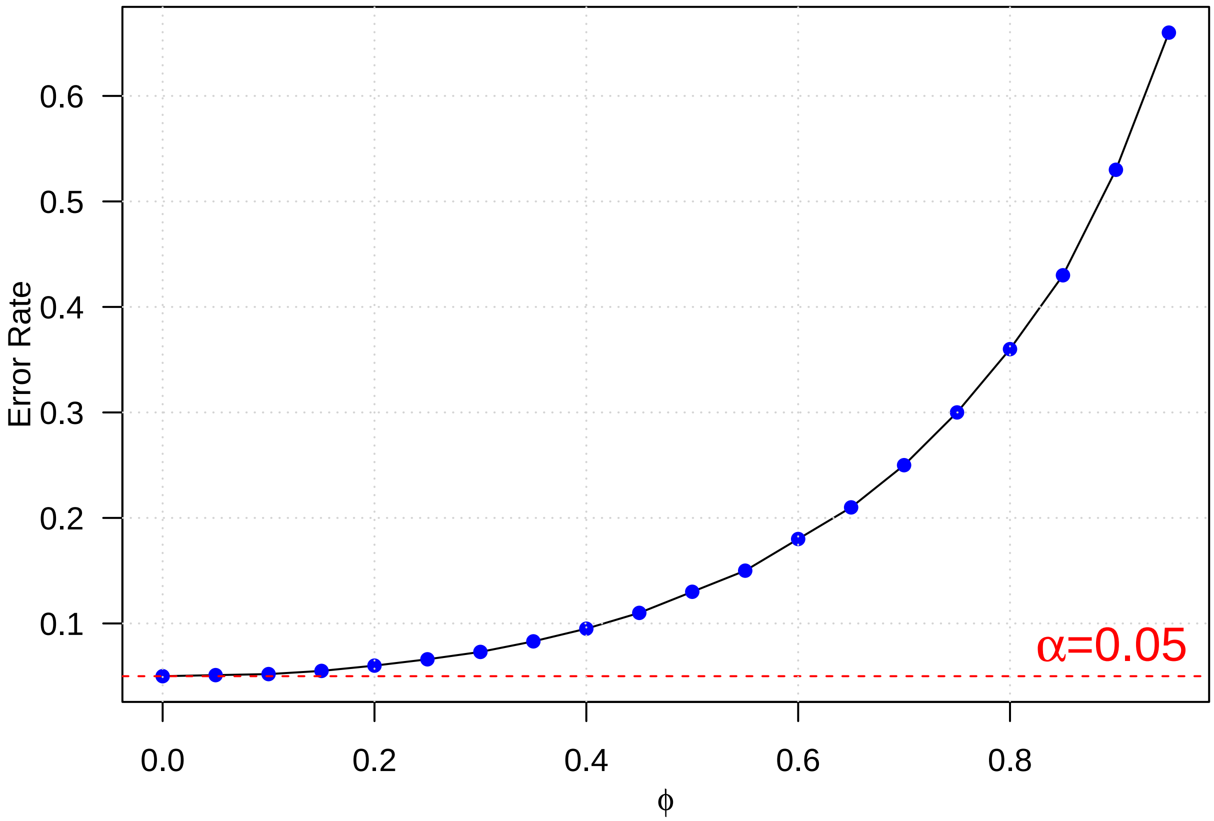
```
## The following object is masked from 'package:utils':
##
##     tar
```

```r
set.seed(123)
n = 105
X <- rnorm(n); Y <- zlag(X, 2) + .5 * rnorm(n)
X = ts(X[-(1:5)], start = 1, freq = 1)
Y = ts(Y[-(1:5)], start = 1, freq = 1)

par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
ccf(X, Y, ylab = 'CCF')
```



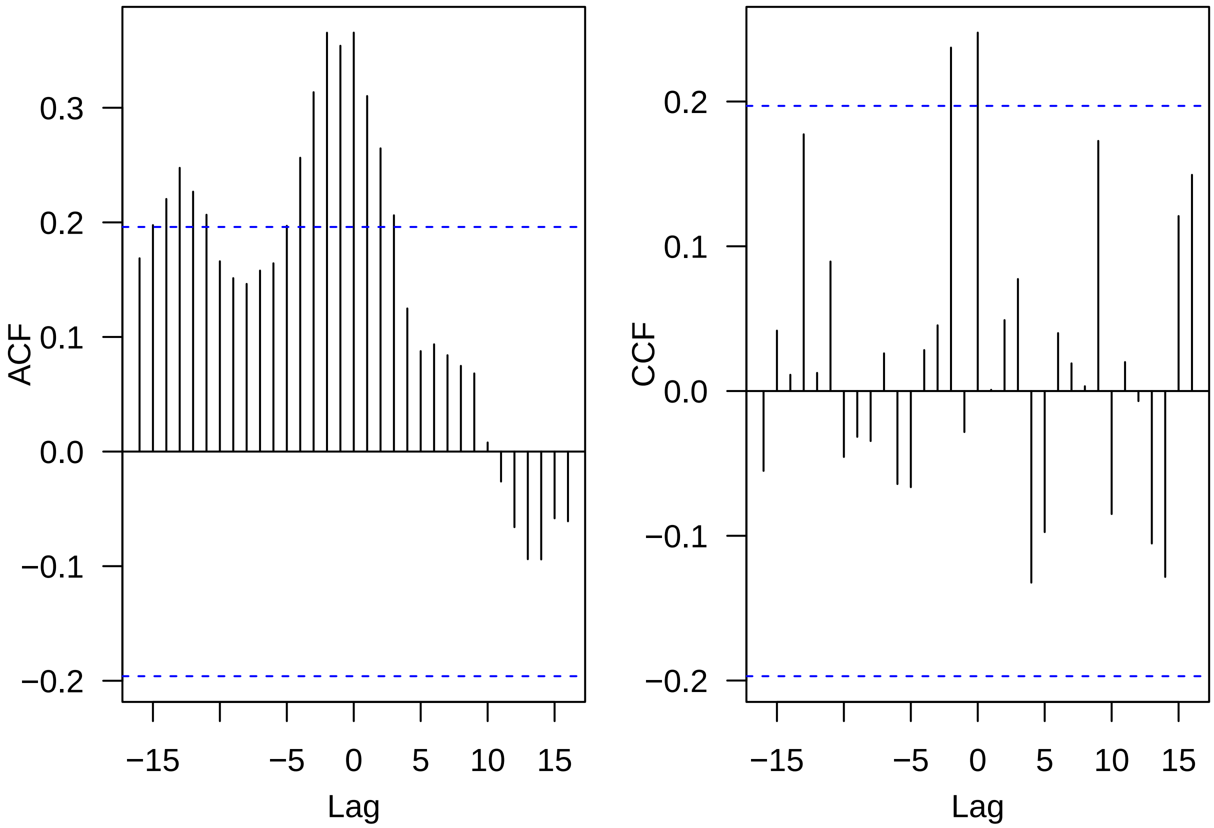
Spurious Correlations: Inflating Type I error rate

```r
phi <- seq(0, .95, .05)
rejection = 2 * (1 - pnorm(qnorm(0.975) * sqrt((1 - phi^2) / (1 + phi^2))))
M = signif(rbind(phi, rejection), 2)
rownames(M) = c('phi', 'Error Rate')
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
plot(M[1,], M[2,], type = "l", xlab = expression(phi), ylab = "Error Rate")
points(M[1,], M[2,], pch = 16, col = "blue")
abline(h = 0.05, lty = 2, col = "red")
legend("bottomright", legend = expression(paste(alpha, "=", 0.05)), bty = "n",
       text.col = "red", cex = 1.5)
grid()
```



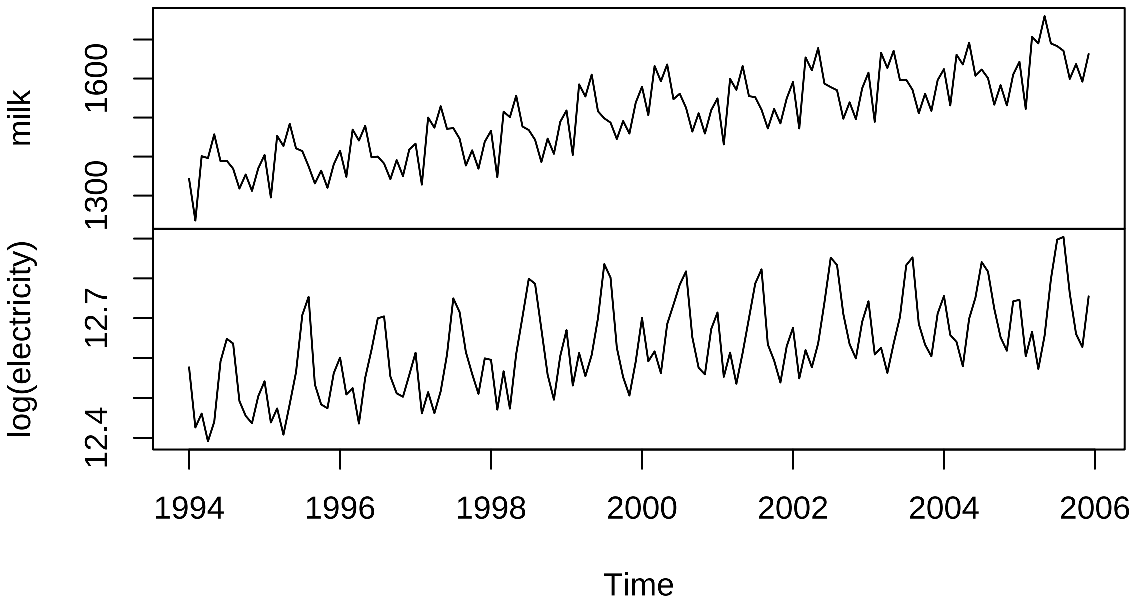
Spurious Correlations: Example I

```
x <- arima.sim(n = 100, list(ar = 0.9))
y <- arima.sim(n = 100, list(ar = 0.9))
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6), mfrow = c(1, 2))
ccf(x, y)
prewhiten(x, y)
```

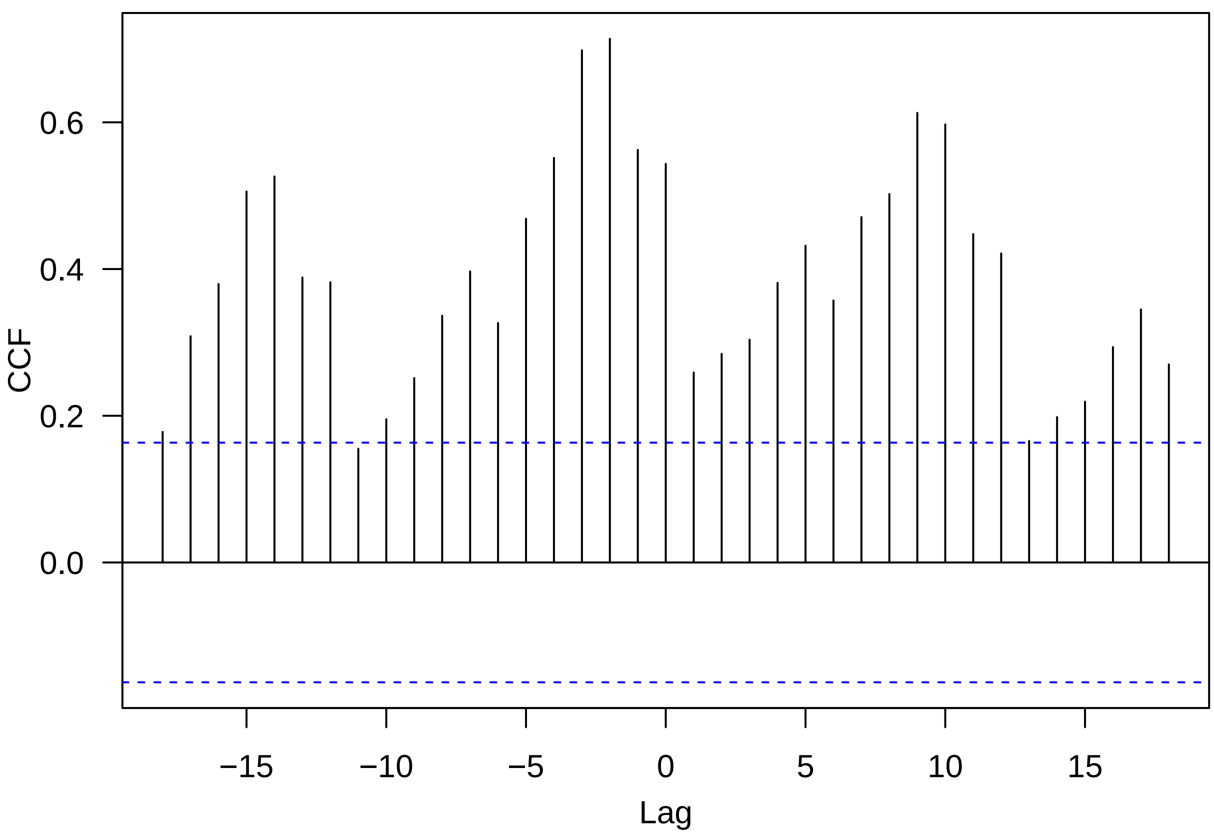


Spurious Correlations: An Example with Milk and Electricity Data

```r
data(milk)
data(electricity)
milk.electricity <- ts.intersect(milk, log(electricity))

par(mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
plot(milk.electricity, main = "")
```

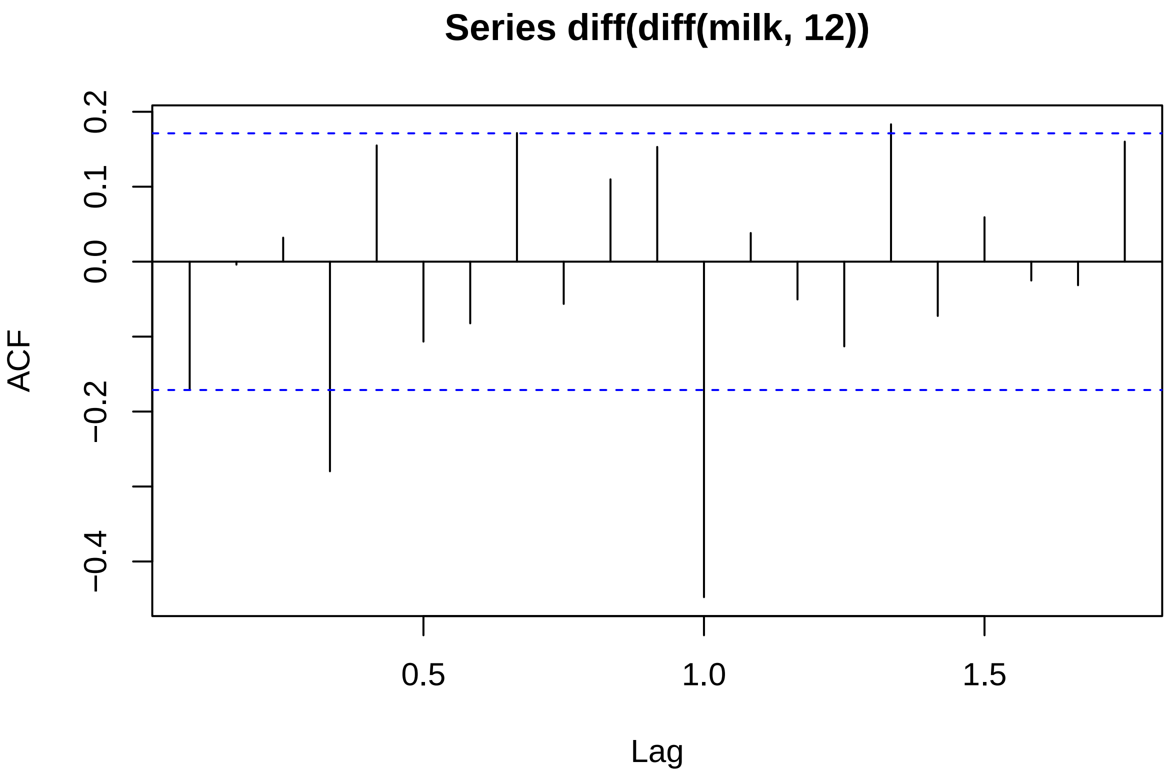



```r
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
ccf(as.vector(milk.electricity[, 1]),
as.vector(milk.electricity[, 2]), ylab = 'CCF', main = "")
```

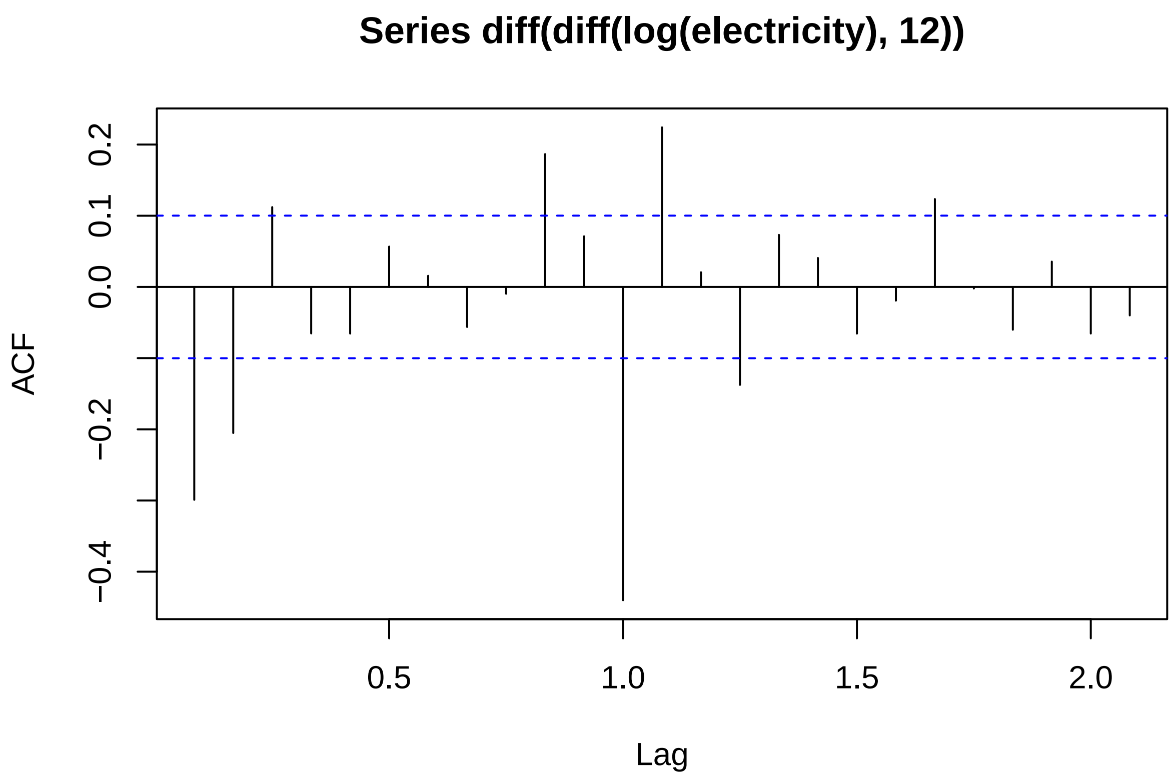


Detrend and remove seasonality by differencing and applying prewhitening

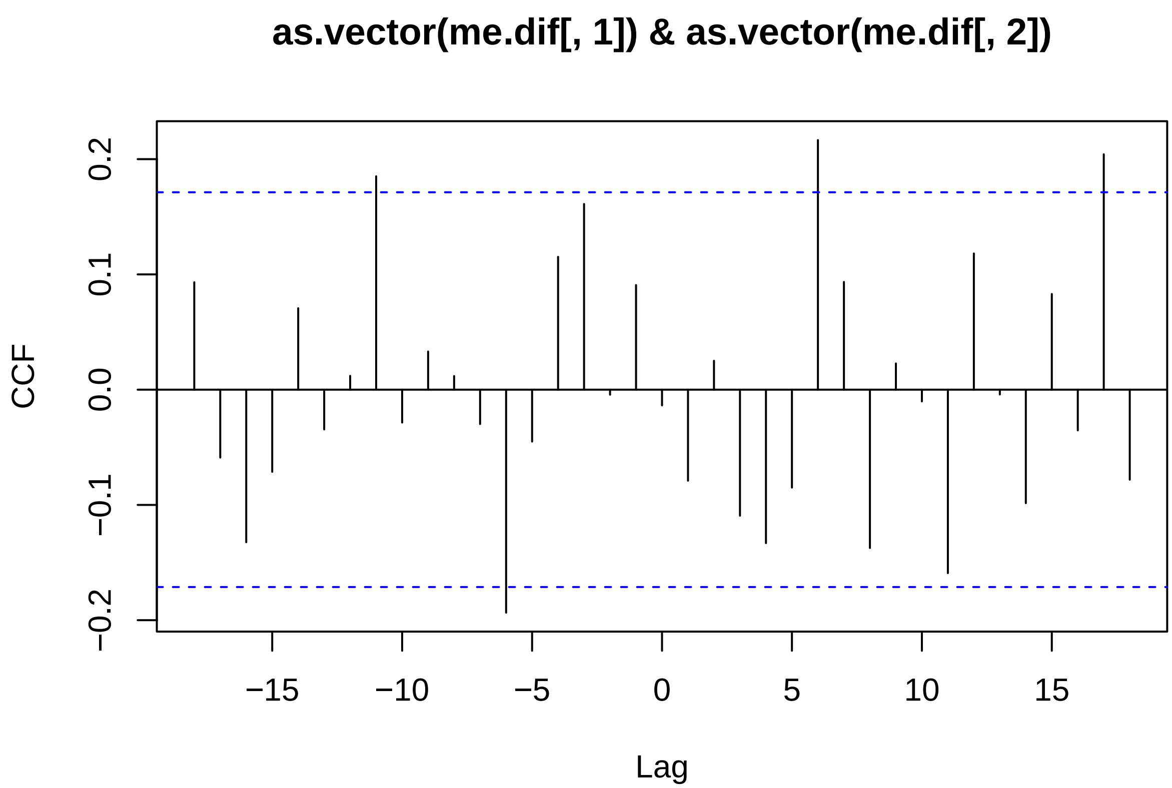
```
me.dif = ts.intersect(diff(diff(milk, 12)),
diff(diff(log(electricity), 12)))

acf(diff(diff(milk, 12)))
```



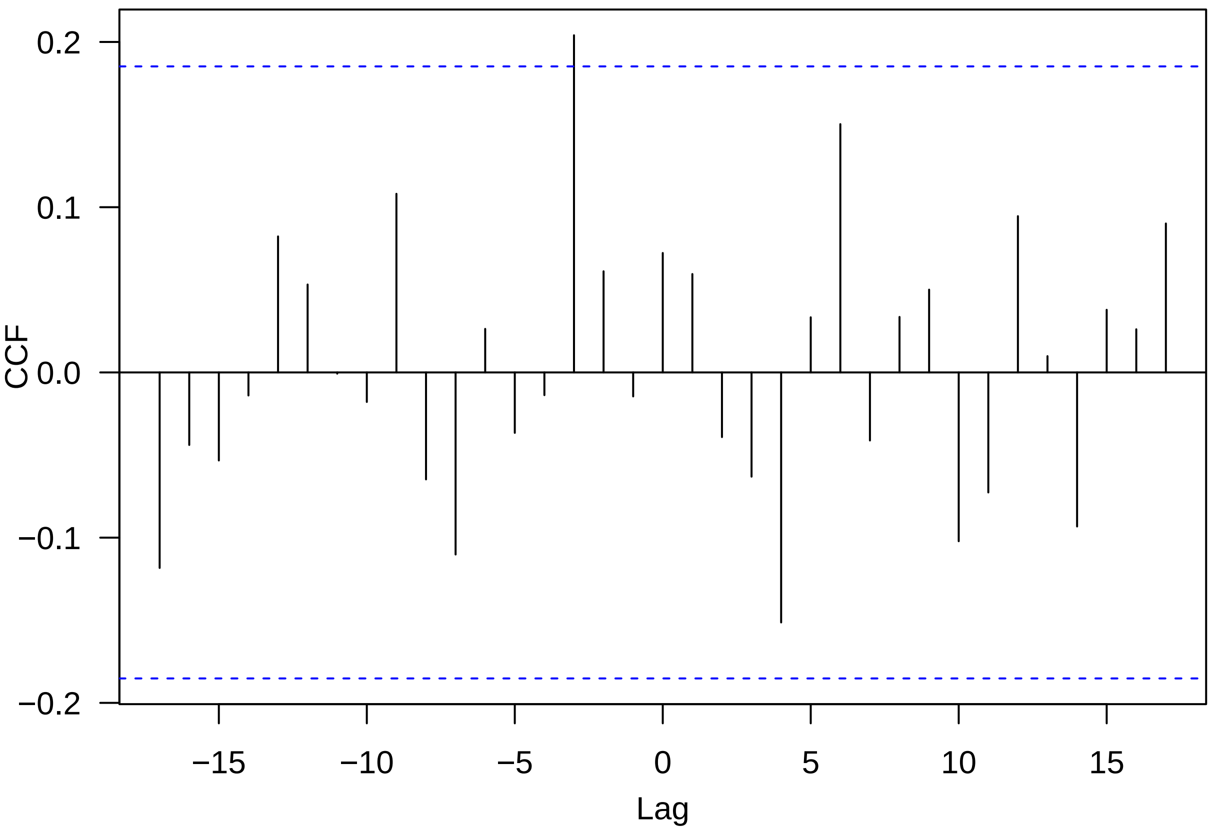

```
acf(diff(diff(log(electricity), 12)))
```




```
ccf(as.vector(me.dif[, 1]), as.vector(me.dif[, 2]), ylab = 'CCF')
```



```r
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
prewhiten(as.vector(me.dif[, 1]), as.vector(me.dif[, 2]), ylab = 'CCF')
```



References

Jones, Philip D, and Michael E Mann. 2004. "Climate over Past Millennia." *Reviews of Geophysics* 42 (2).