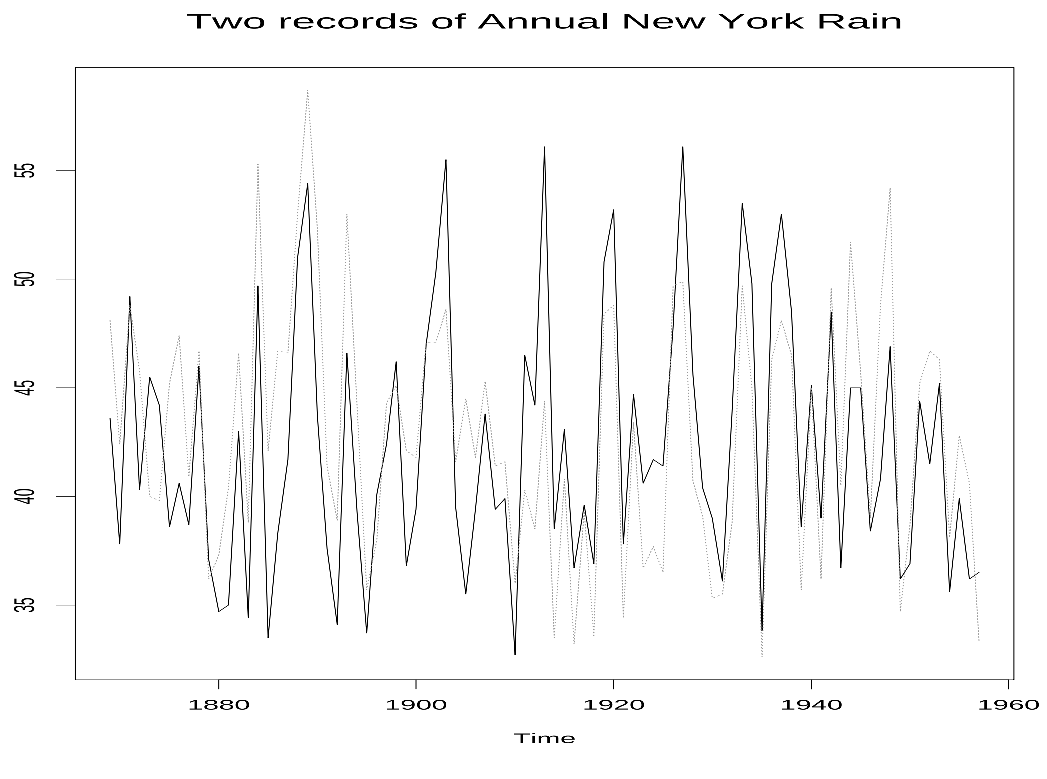
Stat 804: Lecture 10

Model identification for a rainfall difference series

Built into S-Plus are two time series *rain.nyc1* and *rain.nyc2*.

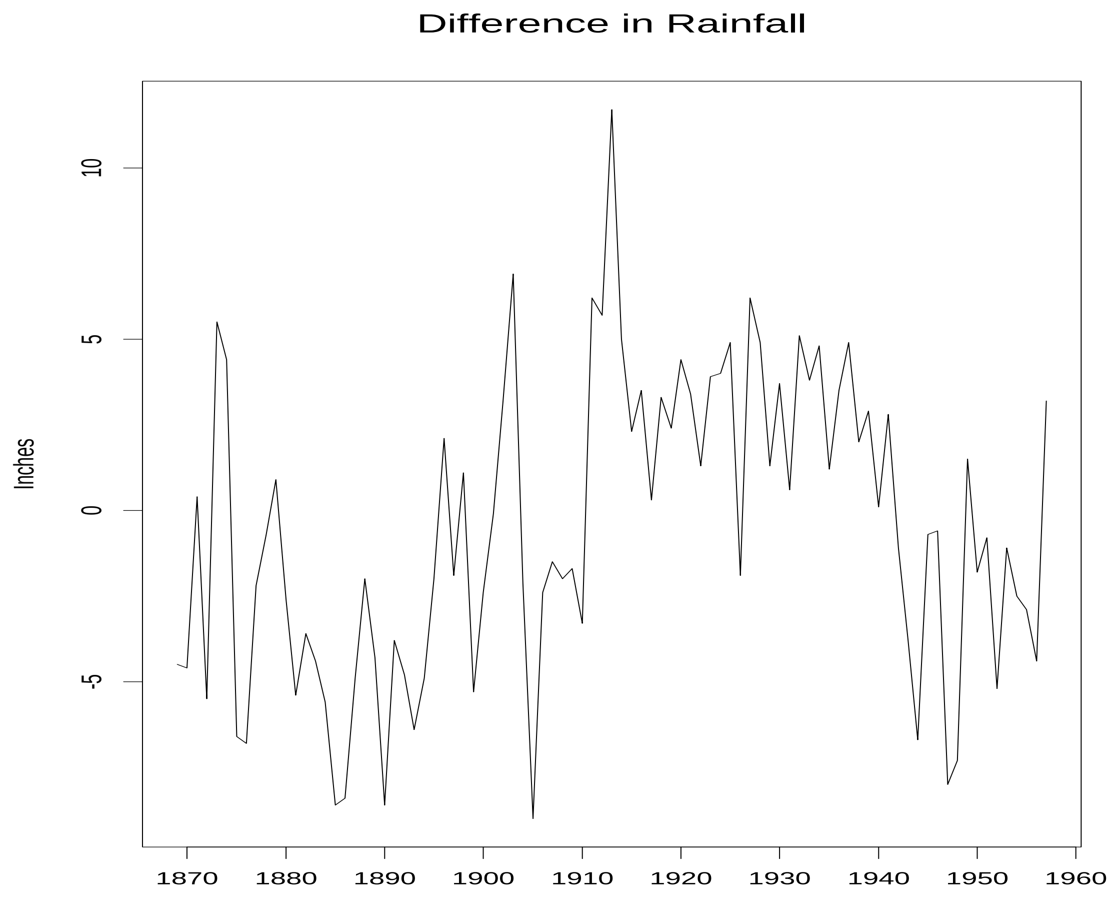
These series are each supposed to be annual rain falls in New York in inches. Here is a plot of the series:



I used S to do plotting, model identification and fitting

```
#
# Time series x is the difference between
#  two rain fall series for New York
#
postscript("raindiff.ps",horizontal=F)
tsplot(x,ylab="Inches",
          main="Difference in Rainfall")
dev.off()
```

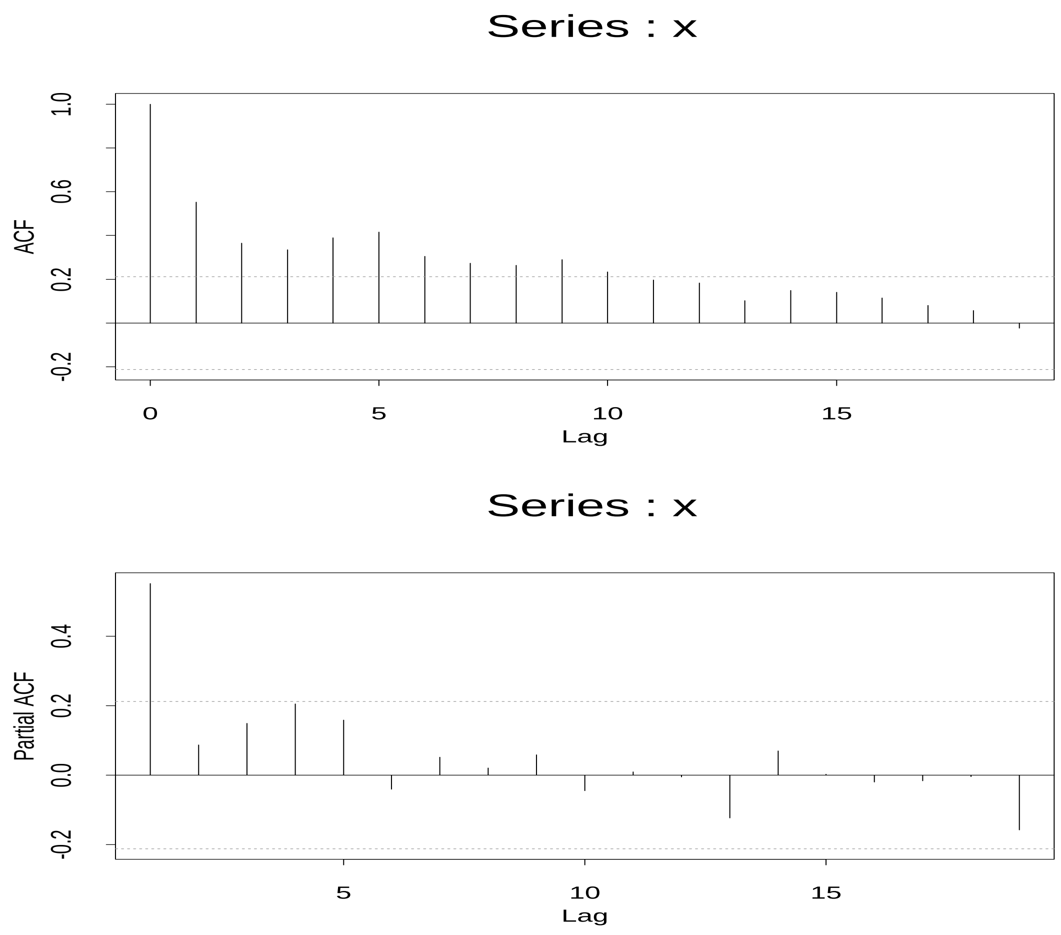
This produces the following plot:



Now I plot the ACF and PACF:

```
postscript("raindiff_acf.ps",horizontal=F)
par(mfrow=c(2,1))
acf(x)
acf(x,type="partial")
dev.off()
```

which gives me:



The PACF is more or less negligible after lag 1 and so I begin with an AR(1) fit

```
> x.arywft <- ar(x,order.max=1)
> x.arywft$ar
      # estimated ar coefficient

, , 1
         [,1]
[1,] 0.552542
> x.arywft$var.pred
      # estimated noise
        (or prediction) variance
         [,1]
[1,] 13.45628
```

The same model can be fitted using the general ARIMA fitting function *arima.mle*:

```
#
#   You must remove a mean from x or put
#    in a regression term explicitly.  The
#    model is specified by a list.  Here
#    I specify an ARIMA(1,0,0) model. Much
#    more general models are possible.
#
> x.armlft <- arima.mle(x-mean(x),
              model=list(order=c(1,0,0)))
#
#  The result is a list with many
#  components including
#
> x.armlft$model
$order:
[1] 1 0 0
$ar:
[1] 0.5572896
$ndiff:
[1] 0
```

```
#
#  You see that the autoregression
#  coefficient is slightly different
#  as is the estimated error variance
#
> x.armlft$sigma2
[1] 13.0793
#
#  Finally the estimated variance of
#   the fitted coefficient is
#
> x.armlft$var.coef
            ar(1)
ar(1) 0.007834412
```

You get the same result from the code

```
> arima.mle(x,xreg=rep(1,length(x)),
          model=list(order=c(1,0,0)))
```

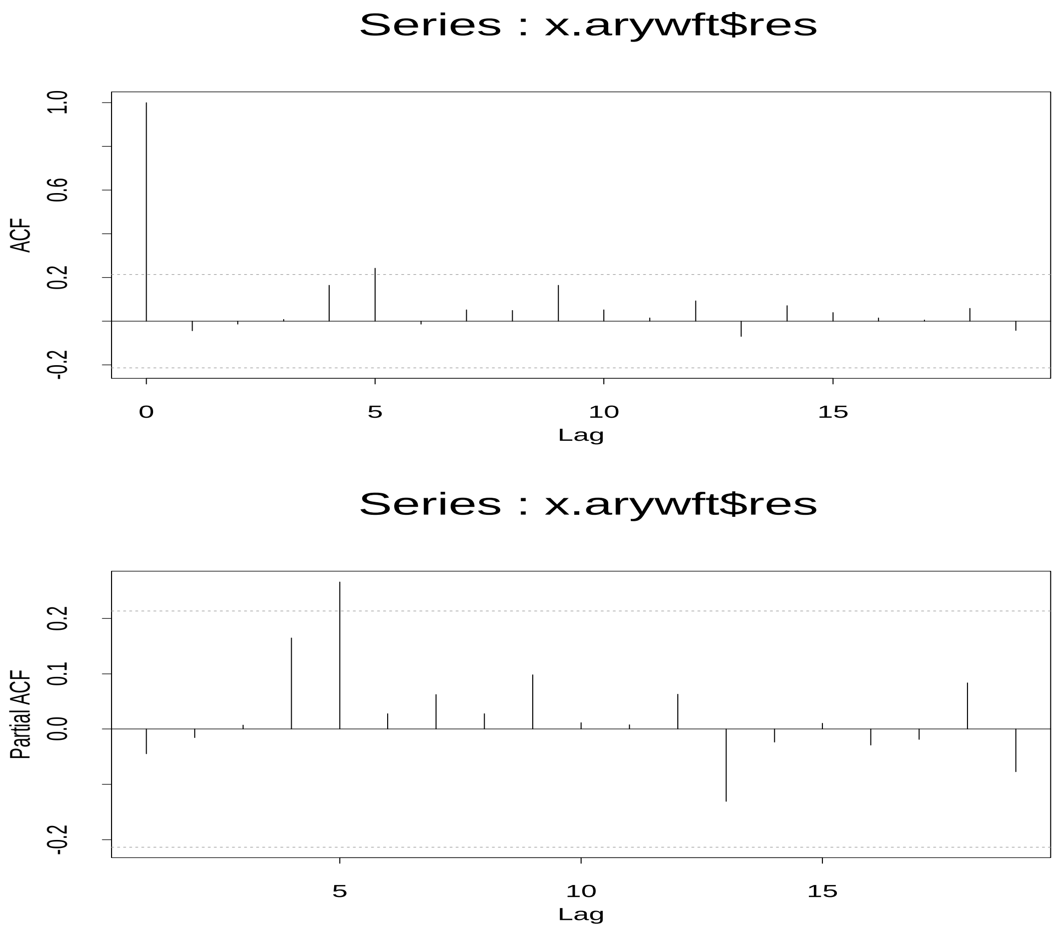
some of whose output is

```
$model$ar:
[1] 0.5574448
$var.coef:
            ar(1)
ar(1) 0.007832446
$sigma2:
[1] 13.07461
```

Next I plotted the autocorrelation function of the residuals which are part of the fit.

```
postscript("raindiff_resid_acf.ps",
           horizontal=F)
par(mfrow=c(2,1))
acf(x.arywft$res)
acf(x.arywft$res,type="partial")
dev.off()
```

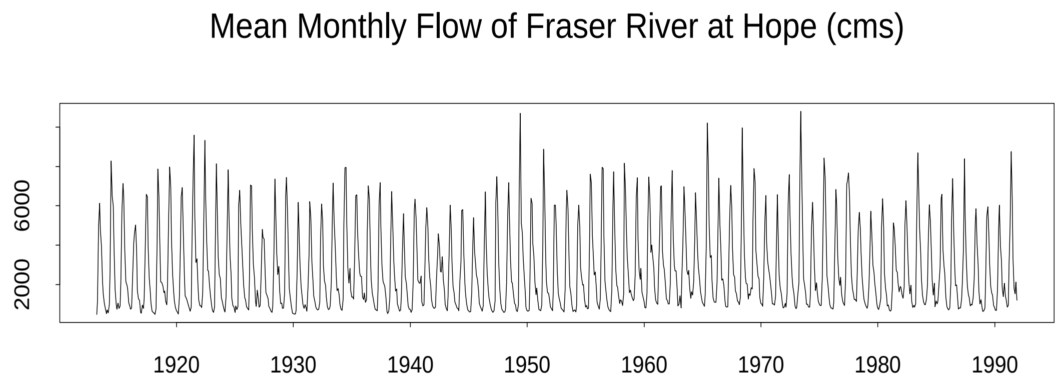
giving the plot:

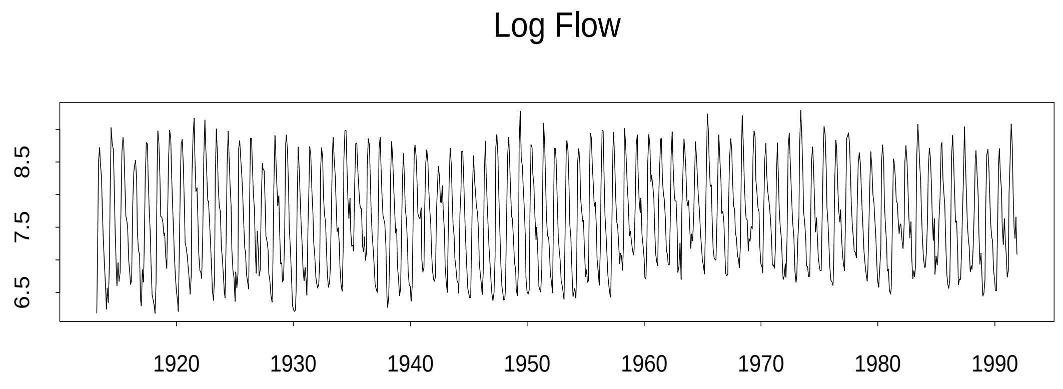


The plots suggest some problem at lag 5. This coefficient is not very significant however, and hard to attach any physical meaning to so I prefer to regard it as sampling variation.

Model identification for Fraser River monthly flows

A data set measuring mean monthly Fraser River flows in cubic meters per second at Hope each month is available through statlib. Here is a plot of the series:

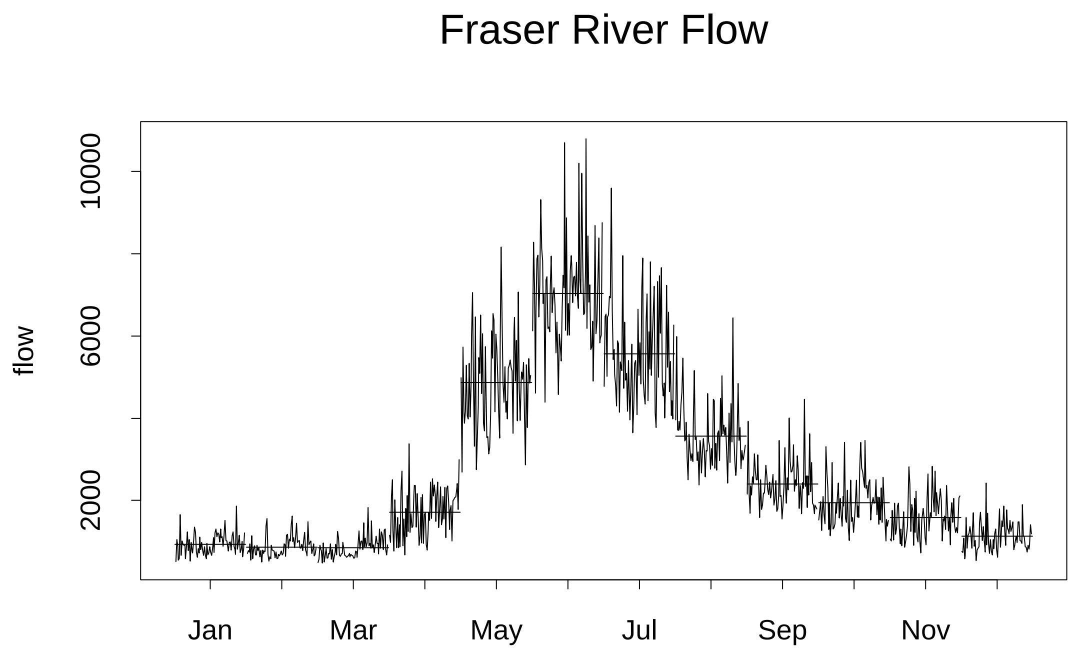


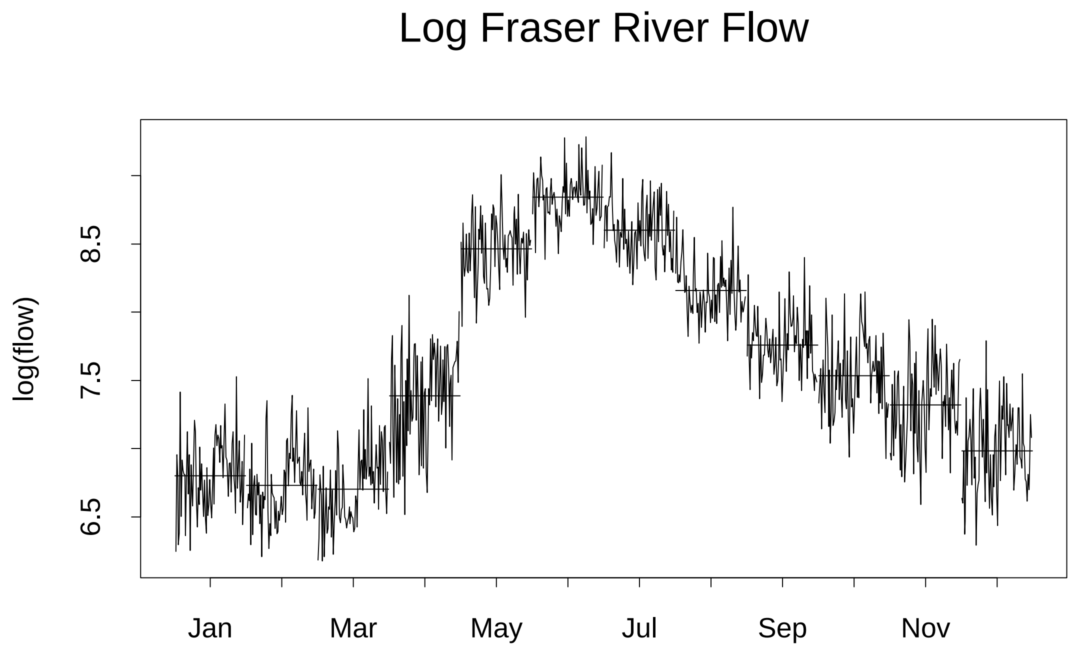


The top of the series appears more volatile than the bottom. To get a good look at this we may use the splus function `monthplot`:

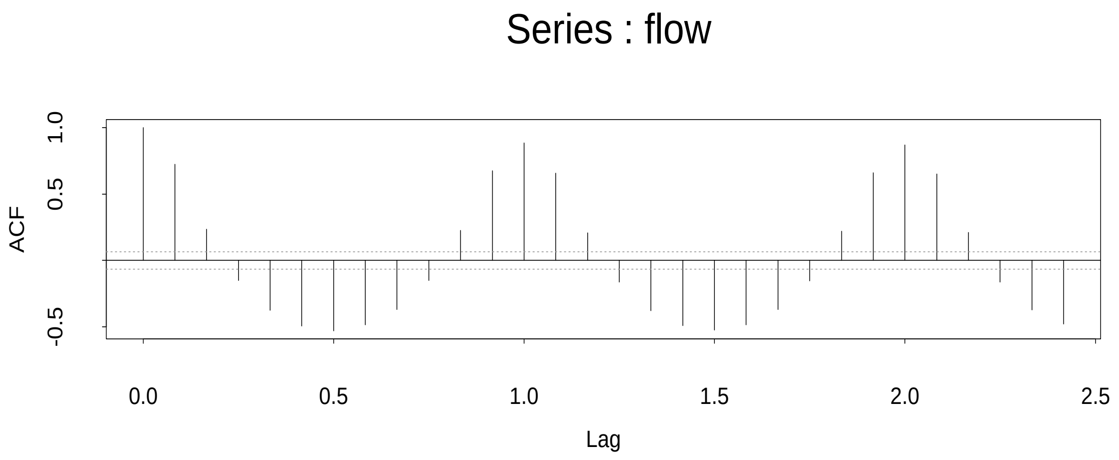
```
postscript(file=
      "fraserriver_monthplot.ps",
       horizontal=F)
par(mfrow=c(2,1))
monthplot(flow,
         main="Fraser River Flow")
monthplot(log(flow),
     main="Log Fraser River Flow")
```

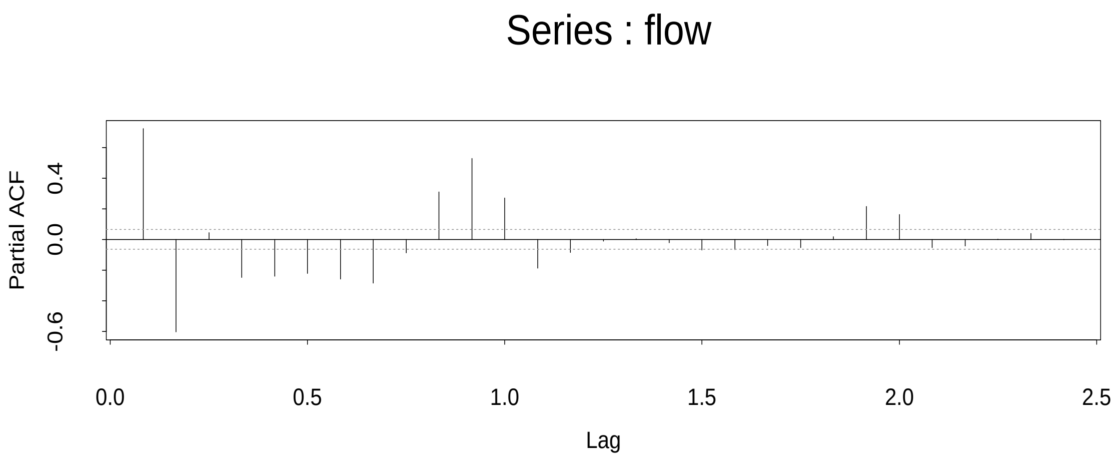
The graphs on the next page suggest transformation is warranted.


Fraser River Flow
flow
2000
6000
10000
Jan
Mar
May
Jul
Sep
Nov


Log Fraser River Flow
log(flow)
6.5
7.5
8.5
Jan
Mar
May
Jul
Sep
Nov

Here are the ACF and PACF:

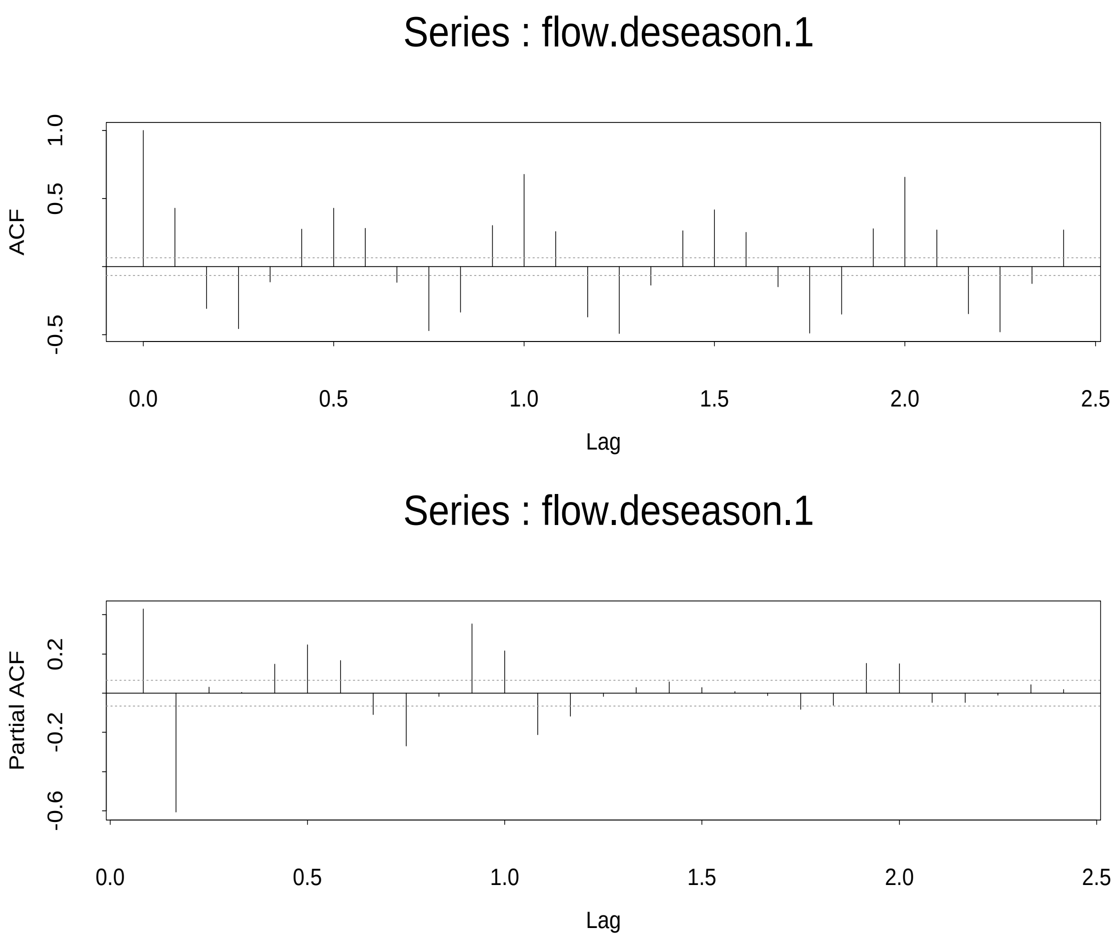




The acf is clearly periodic and I tried a simple sinusoid model:

```
xr <- cbind(rep(1, n),
        cos((2 * pi * (1:n))/12),
        sin((2 * pi * (1:n))/12))
fit.0 <- lm(flow ~ xr -1)
# This fits a linear model with no
#  intercept since I put the column
#  of 1's into  xr.  There is no allowance
#  for the time series structure yet.
flow.deseason.1 <- residuals(fit.0)
# stores the residuals
```

Here are the ACF and PACF of the residuals:



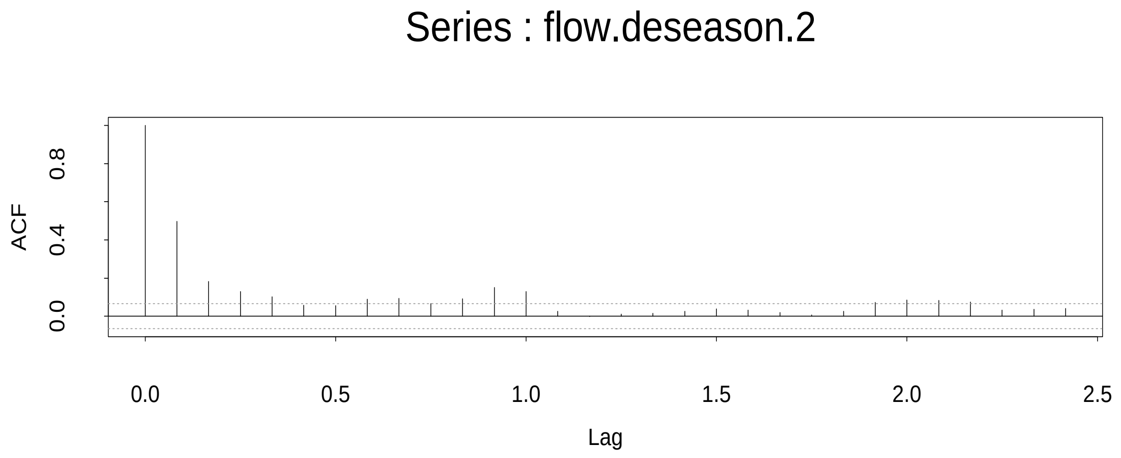
There is still a clear periodic component so the annual cycle is not a simple sinusoid. Instead I built a matrix of indicators for the months

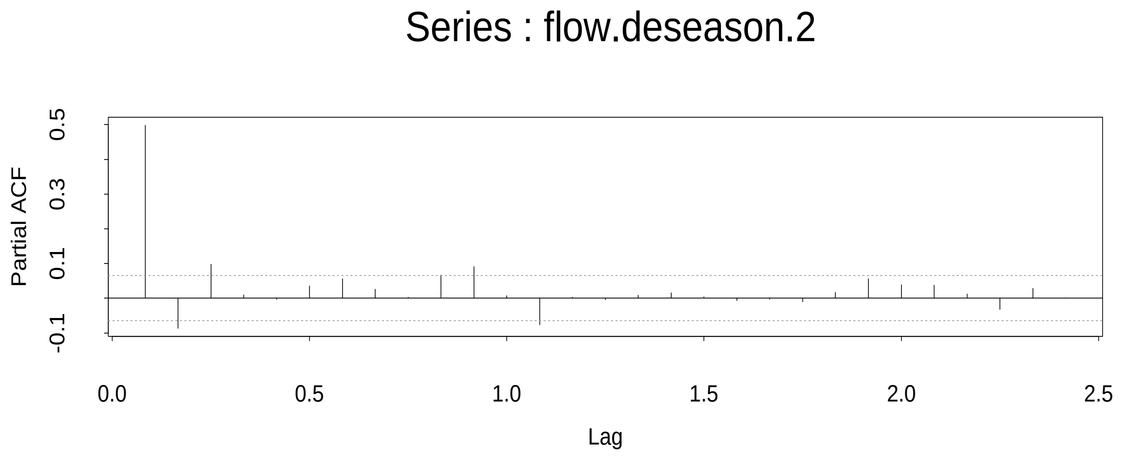
```
xrf <- diag(12)
xrf <- matrix(rep(xrf,78),ncol=12,byrow=T)
xrf <- rbind(xrf[3:12,],xrf)
```

This builds a matrix of 12 columns and 946 rows (there are 946 months of data) with a 1 in column i indicating the time period is month i. My next fit regressed flow on this matrix:

```
fit.1 <- lm(flow ~ xrf - 1)
flow.deseason.2 <- residuals(fit.1)
```

The ACF and PACF of these residuals are:





Plots do not provide a clear simple model but suggest we need something at lags 1, 2 3 and perhaps 12 or so.

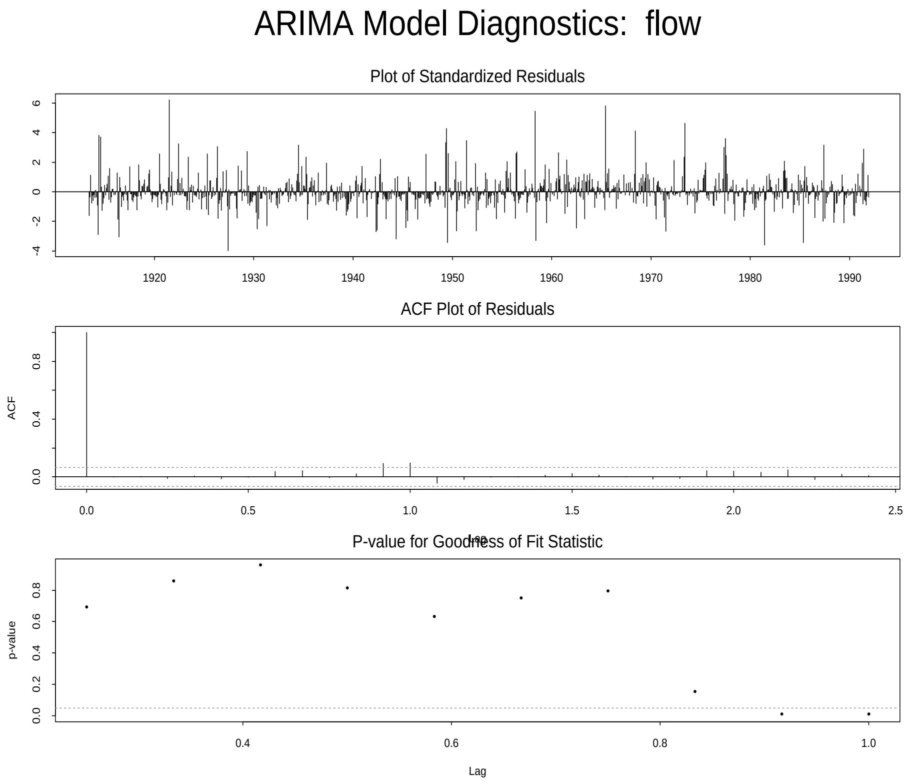
Latter point motivated by idea that there is some effect of the previous years weather.

It is far from clear that such an effect should be summarized in terms of the flow precisely a year ago and you might instead consider coefficients at several lags in the previous year. I tried to begin by ignoring the effects of the previous year and just fit an AR(3).

```
model.2 <- list(order = c(3, 0, 0))
fit.2 <- arima.mle(flow, xreg = xrf,
          model = model.2)
arima.diag(fit.2)
```

Notice that I chose to return to fitting models to the original now permitting the software to make allowance for the correlation structure in fitting the separate means for each month.

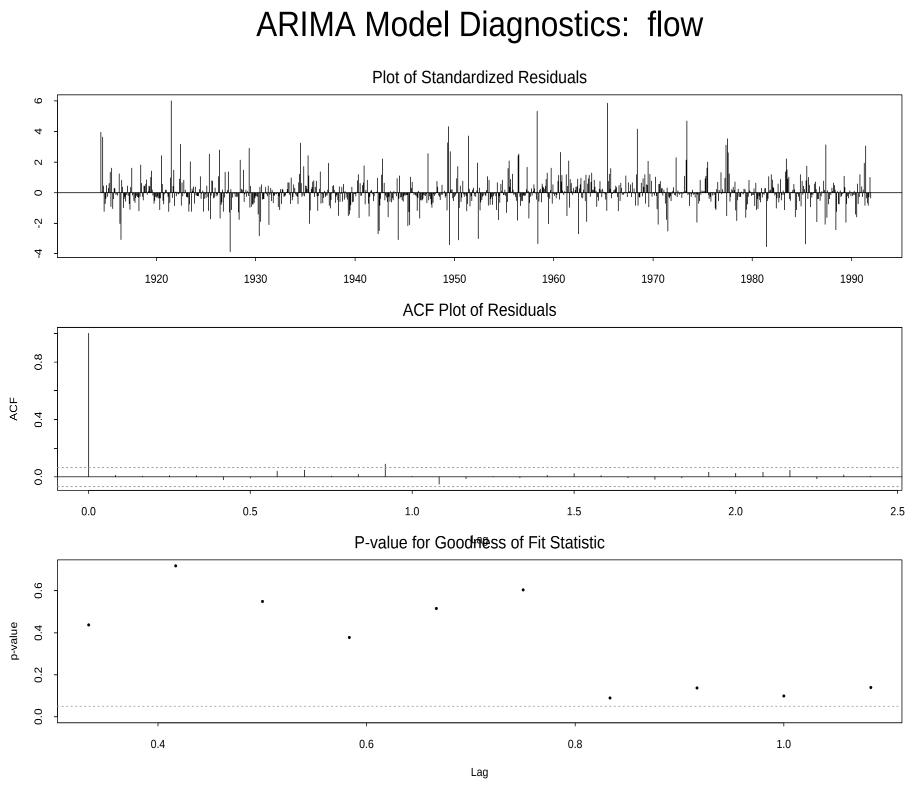
This fit produces the following diagnostic plot



Definite sign of problem near lag of 12 months. I tried adding a seasonal AR component:

```
 model.3 <- list(model.2,
      list(order = c(1, 0, 0), period = 12))
fit.3 <- arima.mle(flow, xreg = xrf,
      model = model.3)
```

which produces the diagnostic plot:



The diagnostics are now satisfactory. The fitted model is described by

```
fit.3
> $model:
$model[[1]]:
$model[[1]]$order:
[1] 3 0 0
$model[[1]]$ar:
[1]  0.54173142 -0.13568443  0.09147659
$model[[1]]$ndiff:
[1] 0
$model[[2]]:
$model[[2]]$order:
[1] 1 0 0
$model[[2]]$period:
[1] 12
$model[[2]]$ar:
[1] 0.09917021
$model[[2]]$ndiff:
[1] 0
```

```
$var.coef:
          ar(1)      ar(2)      ar(3)     ar(12)
 ar(1)  1.06e-03 -5.68e-04  9.25e-05 -6.39e-07
 ar(2) -5.68e-04  1.36e-03 -5.68e-04 -5.29e-08
 ar(3)  9.25e-05 -5.68e-04  1.06e-03 -3.97e-06
ar(12) -6.39e-07 -5.29e-08 -3.97e-06  1.06e-03

$method:
[1] "Maximum Likelihood"

$series:
[1] "flow"

$aic:
[1] 14646.29

$loglik:
[1] 14614.29

$sigma2:
[1] 384435.5

$n.used:
[1] 931

$n.cond:
[1] 15
```

```
$converged:
[1] T

$conv.type:
[1] "relative function convergence"

$reg.coef:
 [1]  938.1983  868.5136  855.3402
 [4] 1737.1811 4904.1479 7064.3307
 [7] 5581.4333 3563.6286 2406.0266
[10] 1952.4763 1592.0973 1133.9484

$reg.series:
[1] "xrf"
```

WARNING: I had to do the following to get the algorithm to converge:

```
fit.3 <- arima.mle(flow,
       xreg = xrf, model = model.3)
fit.3 <- arima.mle(flow,
       xreg=xrf,model=fit.3$model)
```

The use of fit.3$model provides the output of the first fitting effort as starting values for the second fitting effort and the model now converges.

It might actually be better to transform the monthly flow since the variability of the series seems higher when the level is high.

I tried this but the fit was actually a bit worse after taking logs.