Time Series, and Long Memory in R

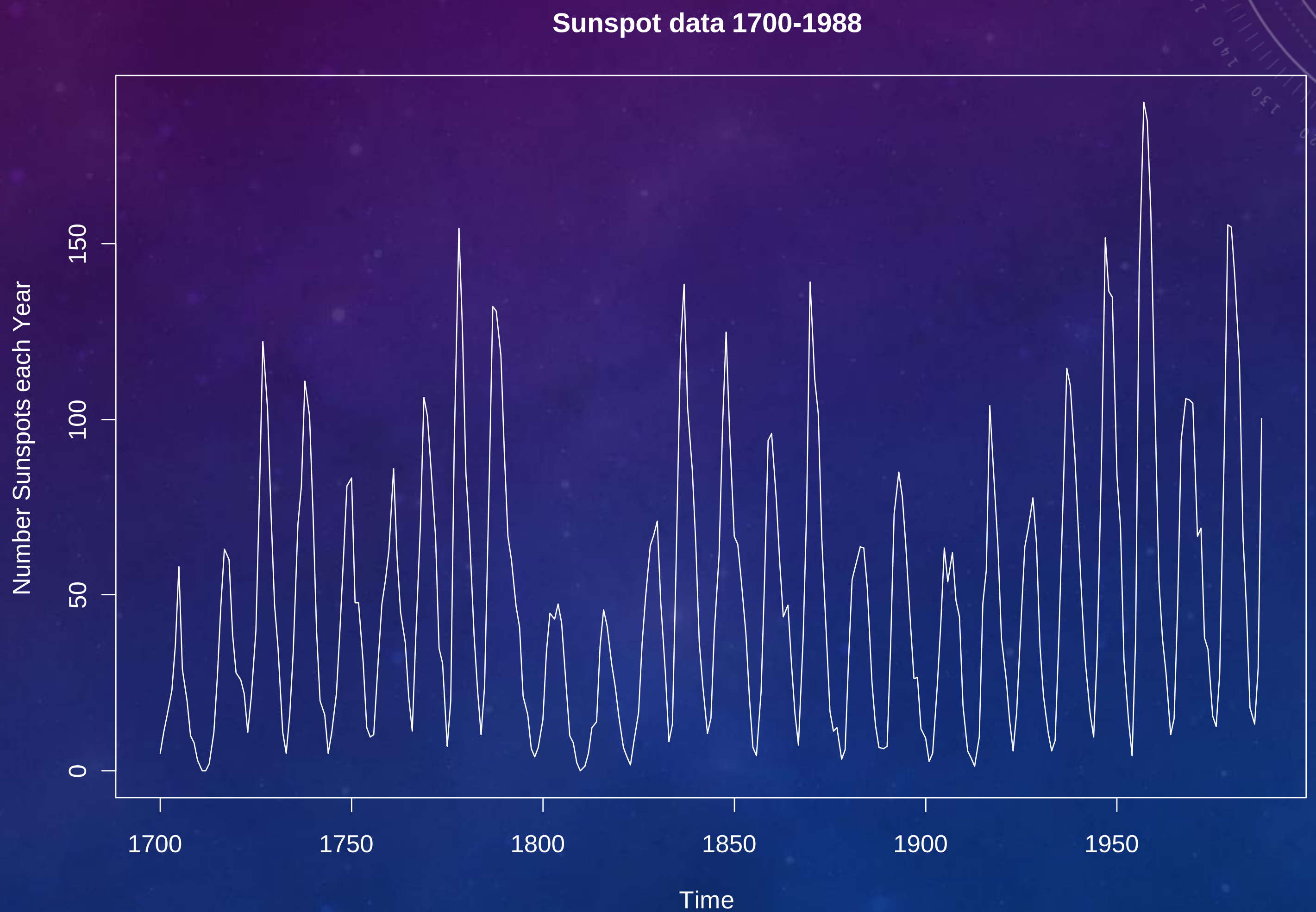
WHAT WE WILL LOOK AT TONIGHT

- Time Series in General
  - A bit of a cookbook… whilst we set things up for the Long Memory part.
  - We look at Trends and differencing. We look at determining the Autoregressive and Moving Average parts of the model.
  - We introduce ARIMA Models, which combine all this together.
  - KEY AIM: Not to show you how to do it, but to give you the tools to find what you need to do it.
- Long Memory
  - How to see if you have it.
  - Cookbook to fit the models.
  - Which R packages are best.
- Some examples
- Little or no algebra!
  - but… you may not thank me! <g>
- Travel Photos

TIME SERIES

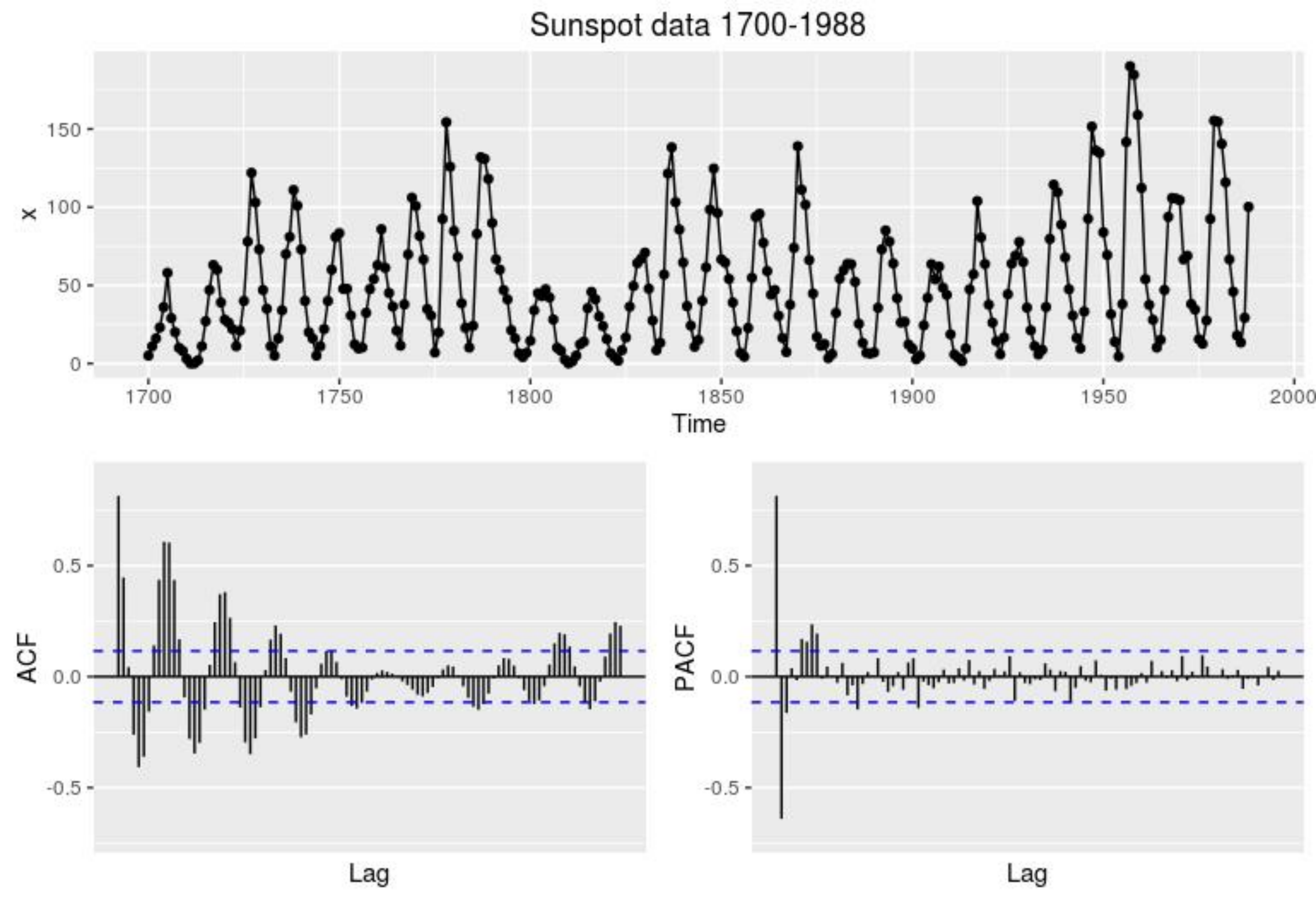
- Financial, Econometrics, Environmental, Physics, Medicine etc
- Next Slides show an example.
  - Note the cyclical nature of the series.
  - Analysis of Time Series (non-Neural Ntwk version!) relies on this.

Annual Sunspot Counts



SUNSPOT COUNTS DATA

```
> library(forecast)
> library(ggplot2)
> data(sunspot.year)
> ggtsdisplay(sunspot.year,lag.max=100, main="Sunspot data 1700-1988")
```



SO "TIME SERIES" HAVE CYCLES

- Cycles which we can model
  - (refer Fourier Theorem!)

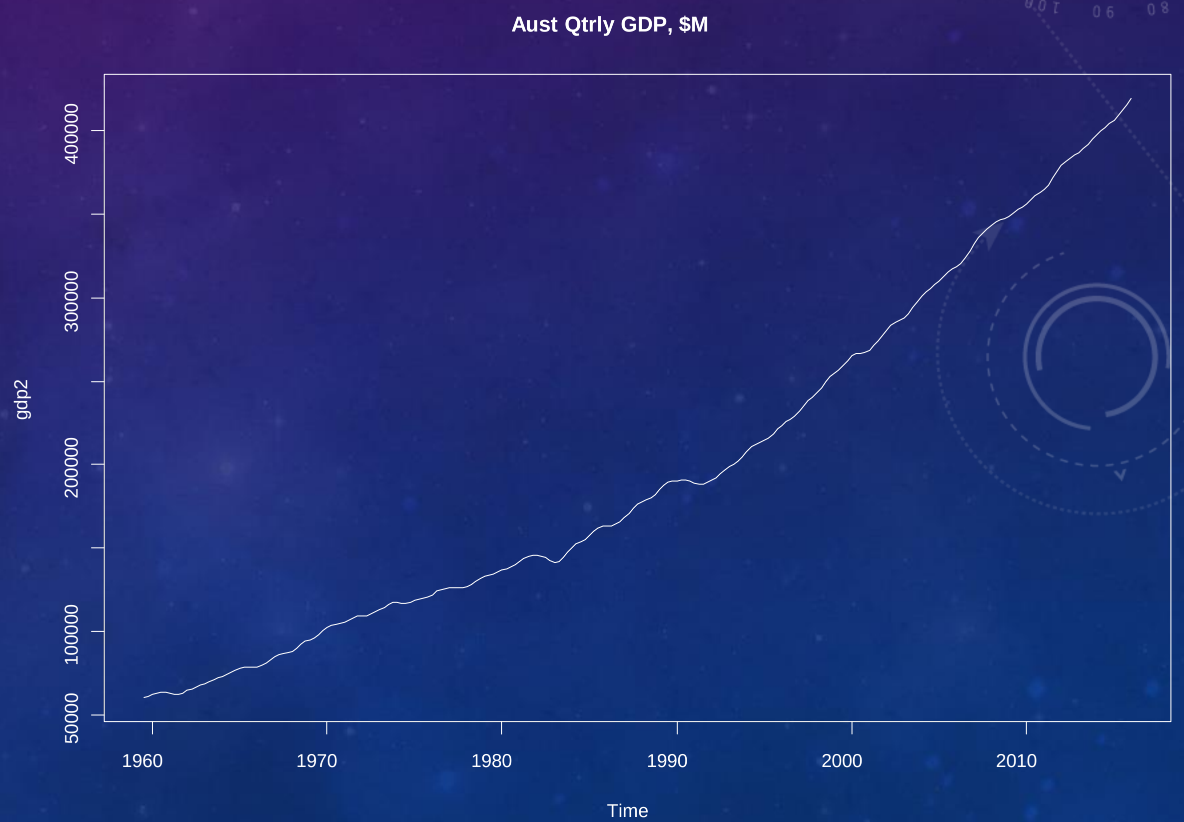
- Key Assumption
  - Average Value doesn't change!
  - "Stationary" Series.
  - If there are trends in the data, in general time series techniques don't work too well.
  - We can difference the series to get a more "stationary" series.

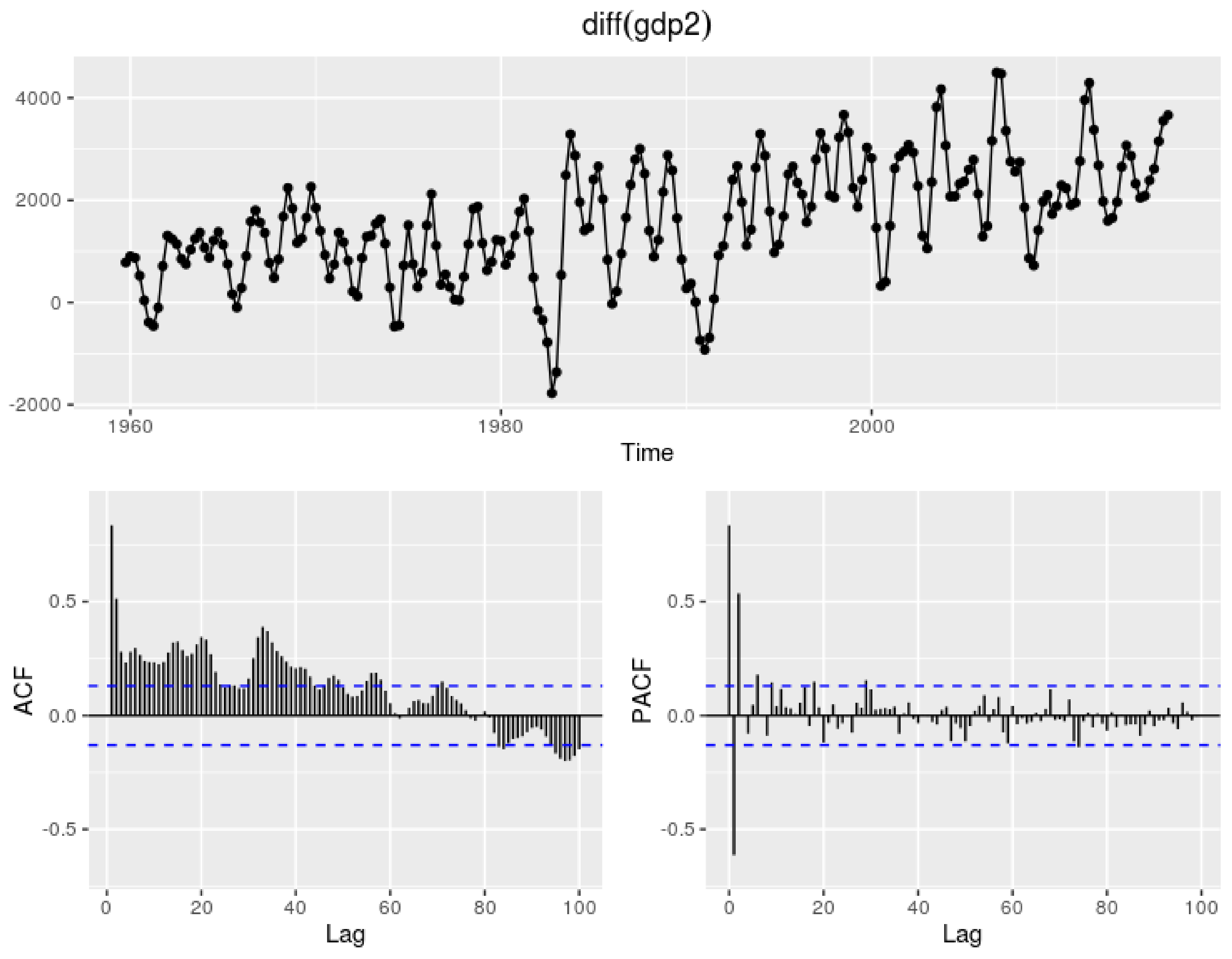
OTHER ASSUMPTIONS

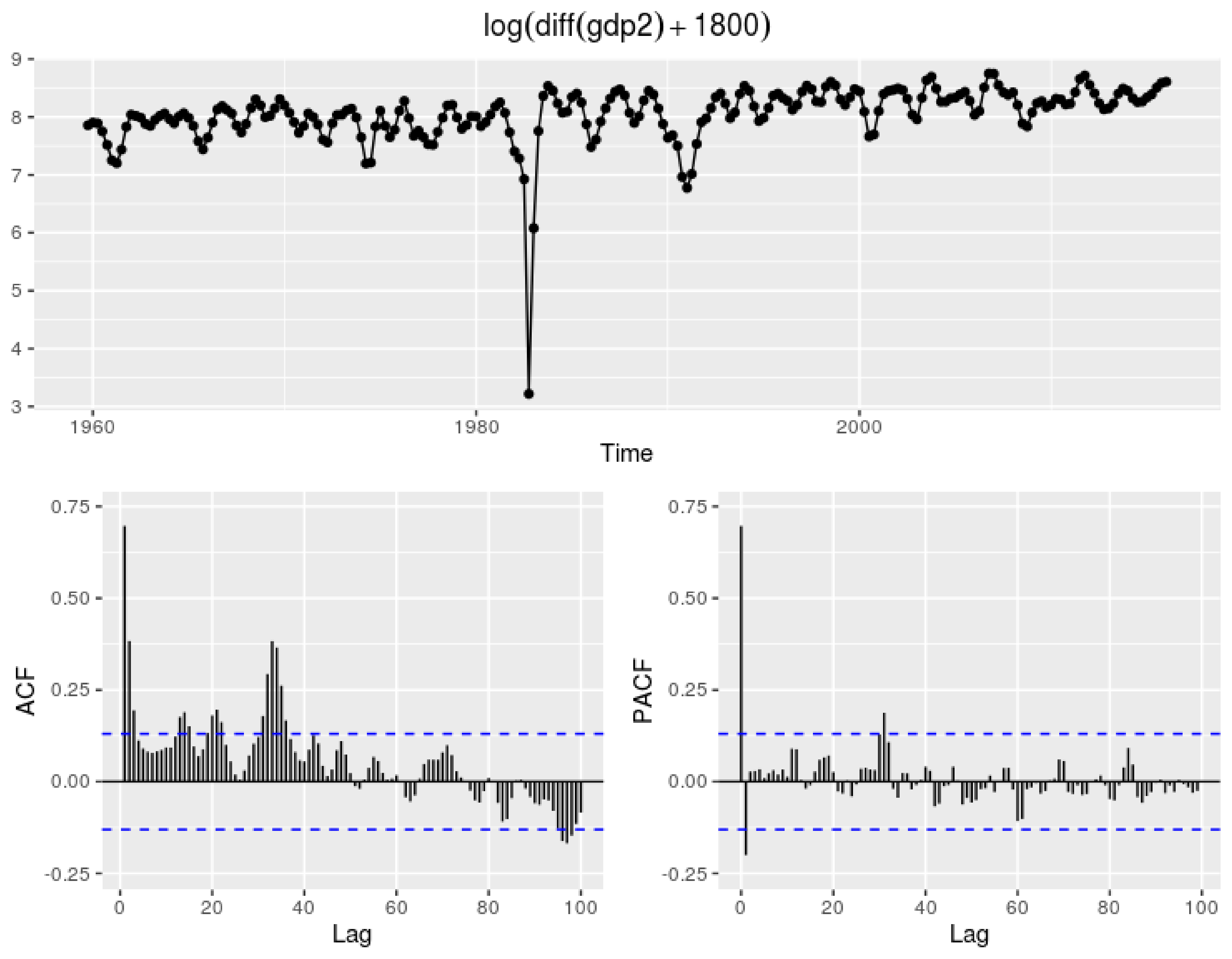
- Variance is approx the same.
  - known as “second order” stationary.
- If not, then GARCH models may be able to help
  - Not further discussed.

REMOVING TRENDS

- Trends are NOT stationary!
- For the below… we don't seem to have anything useful for time series???




diff(gdp2)
4000
2000
0
-2000
1960
1980
2000
Time
ACF
0.5
0.0
-0.5
0
20
40
60
80
100
Lag
PACF
0.5
0.0
-0.5
0
20
40
60
80
100
Lag


log(diff(gdp2) + 1800)
9
8
7
6
5
4
3
1960
1980
2000
Time
ACF
0.75
0.50
0.25
0.00
-0.25
0
20
40
60
80
100
Lag
PACF
0.75
0.50
0.25
0.00
-0.25
0
20
40
60
80
100
Lag

ARIMA – A SET OF USEFUL MODELS!

- ARIMA Models first formalised by Box & Jenkins in 1976.
- ARIMA mixes 3 useful Models.
  - AR = Auto-Regressive
    - This means that we model the series in terms of its past values – ie Y(t) is related to Y(t-1), Y(t-2), Y(t-3), …
  - MA = Moving Average
    - This means that we model the errors (or "disturbances") of the model in terms of the past errors of the process.
  - I – in the middle - "Integrated" - this is the degree of differencing required to remove trends.
  - Thus … AR – I – MA.

SO HOW DO WE FIND THE AR, I, AND MA COMPONENTS?

- First the easy one – I – we need the plot of the series to “look” stationary – ie the average value should not change over time.
- There is a popular test – the “Dicky-Fuller” test – to tell you if you like if taking differences may help (technically looks for unit-roots). Also the kpss test can help (this looks for trends in the data).

```
library(tseries)
adf.test(NileMin)
kpss.test(NileMin)
```

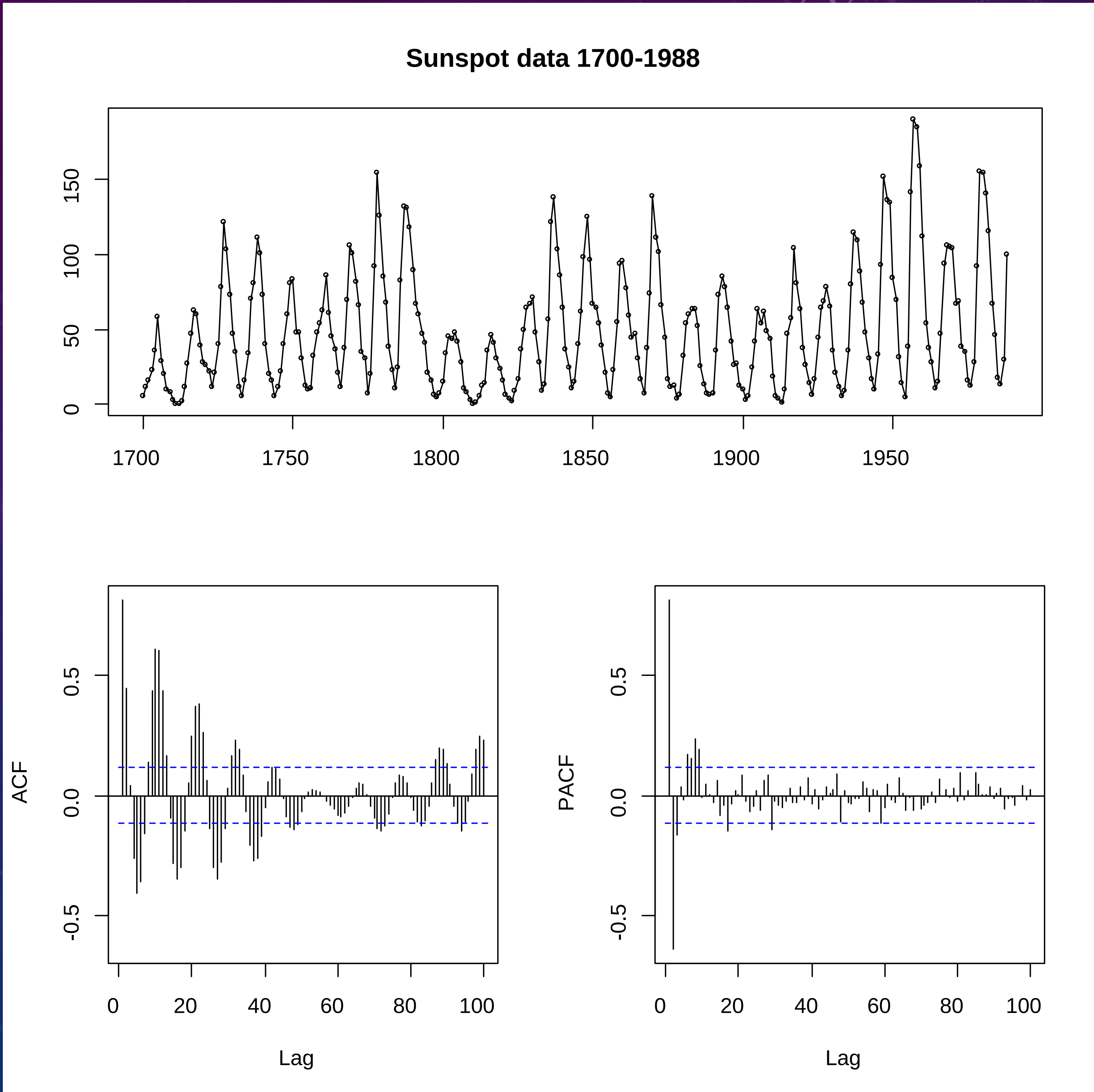
SO HOW DO WE FIND THE AR, I, AND MA COMPONENTS?

- FIRST we need to figure out how many parameters we should give to the model.
- AR – Auto-Regressive – how far back - 1, 2, 3, or more values – is usually called "p".
- Can be determined from "Partial ACF".

- MA – Moving Average – how far back – 1, 2, 3, … is usually called "q".
- Can be determined from the "ACF"

- ACF == "Auto Correlation Function"

GGTSDISPLAY()



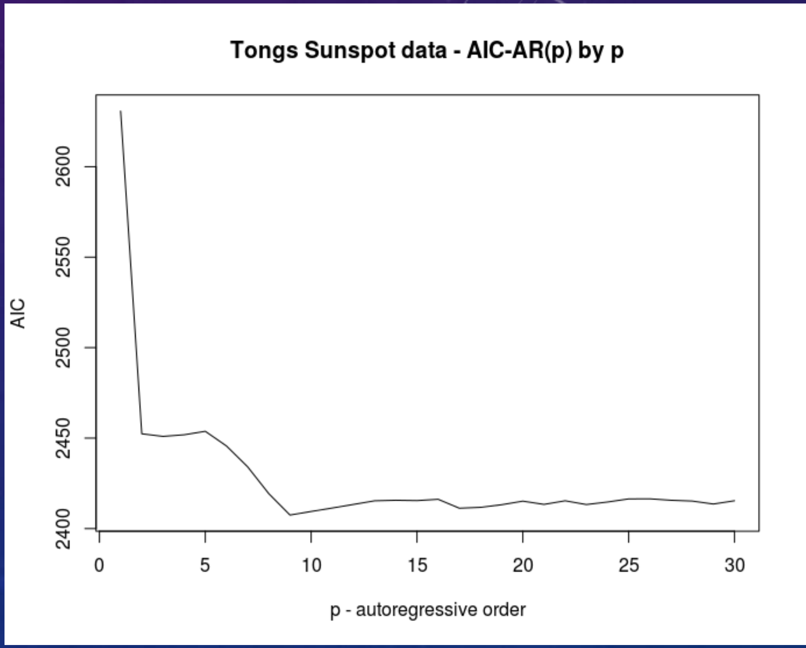
INTERPRETING THE ACF & PACF

- Look for ACF or PACF above the blue lines – these are "significant" correlation.
- The way ARIMA is structured…
  - An "infinite" AR is a finite MA
  - An "infinite" MA is a finite AR
  - The AR and MA components are in a sense "duals" of each other.
  - In the Sunspot data – the ACF components go on as long as the plot – so an "infinite" MA becomes a "finite" AR.
  - Here we can see an AR component (from the PACF) somewhere between 6 and 12 periods (6-12 years).

HOW TO FIND BEST – ARIMA – MODEL? … (BEST SHORT & SWEET VERSION…)

Use the AIC

```
aic_val<-c()
for (p in 1:30)
aic_val=c(aic_val,unlist(arima(sunspot.year,order=c(p,0,0))$aic))
plot(1:30,aic_val,type="l",xlab="p - autoregressive order",ylab="AIC",main="Tongs Sunspot data - AIC-AR(p) by p")
```



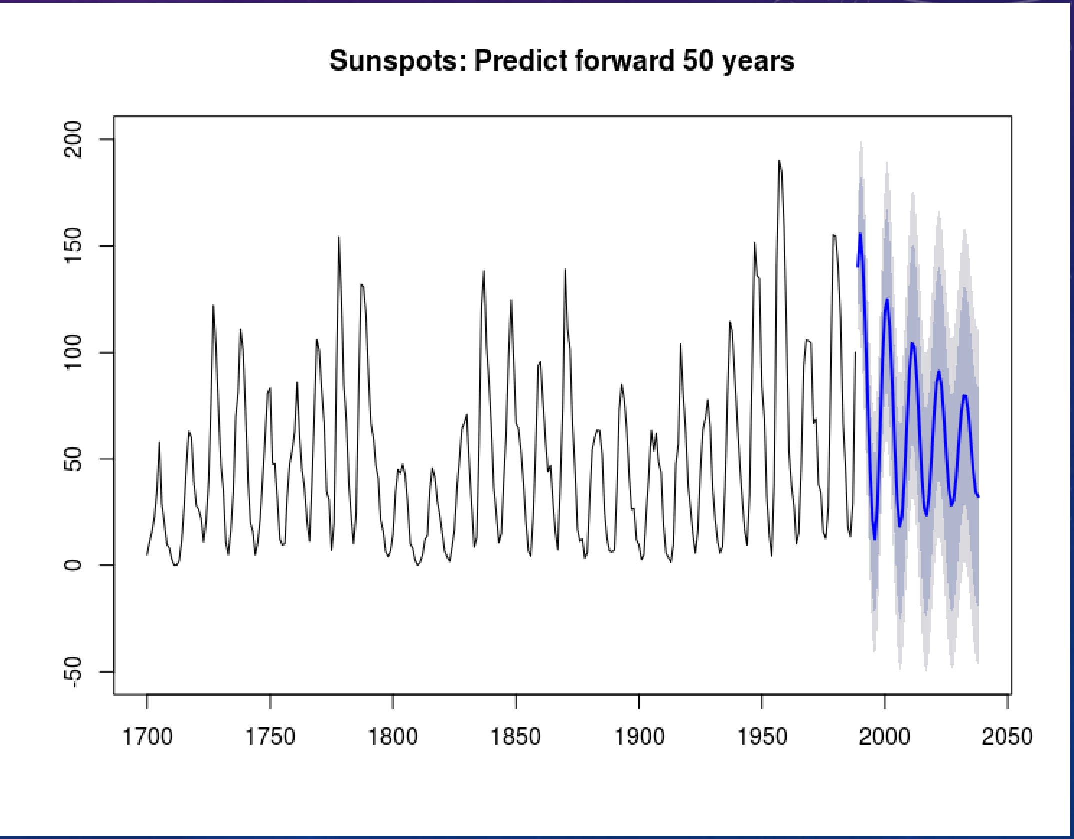
Incredibly Important Note!

AIC only allows you to compare models if you have the same data! If you change the data, then you need to start re-calculating all the AIC information you have.

HOW TO FORECAST?

- Rob Hyndman to the Rescue!

```
> fit<-arima(sunspot.year,order=c(11,0,0))
> plot(forecast(fit,50),main="Sunspots: Predict forward 50 years")
```



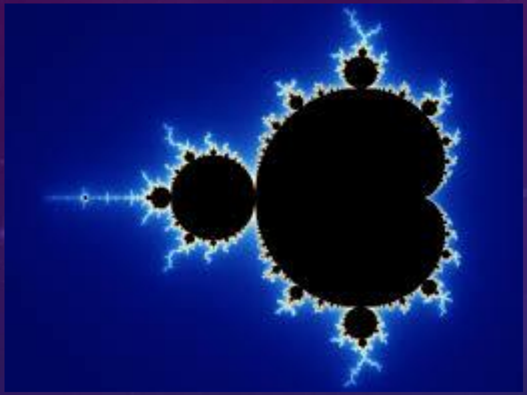
EXTENDING ARIMA – LONG MEMORY

- A Key component of the ARIMA models is that the ACF and PACF should be declining fairly fast, except for “seasonality”.
- But… what if they do not decline so fast?
- This means… points in the series which are far apart from each other are yet highly correlated!!!!!
  - Thus “long memory”.

LONG MEMORY

- Long Memory can be modelled by an extension to ARIMA models.
- Suppose we let the “I” component correspond to a “fractional differencing” parameter.
  - (Defined in terms of an infinite taylor series expansion).
  - Sometimes these models are referred to as ARFIMA models, with the “FI” referring to “Fractionally Integrated”.
  - For Long Memory, generally we refer to the measurement of this by “d”. If d is between -1.0 and +0.5 then process is “stationary”.

FRACTIONAL DIFFERENCING RELATED TO MANDELBROT'S WORK (c 1969)



LONG MEMORY – WHY?

- Key Reason:
- Because observations are correlated when far apart in time, this means that we can forecast ahead in time a lot further than standard ARIMA models.
- Unfortunately though short terms forecasts are generally a bit worse (but not a lot).

EXAMPLES OF LONG MEMORY?

- Harold Hurst published a paper in 1951 identifying Long Memory in measurements from the Nile.
  - These are a famous series – annual low water measurements from the Rhoda gauge near Cairo.
  - Some sample measurements available in “R” are from “Tousson, O. (1925). Mémoire sur l’Histoire du Nil; Mémoire de l’Institut d’Egypte” and cover the period 622-1281 – more than 600 years.

- Hursts’ work was a key component in the building of the Aswan Dam.
  - Hurst was Director of the “Physical Department” of the Egyptian Department of Public Works.
  - Dam level was made considerably higher as a result of Hursts’ work.
  - Nasser later took up the project and had to hunt around for money to build it – causing all sorts of problems in the process.

THE NILOMETER AT RHODA NEAR CAIRO

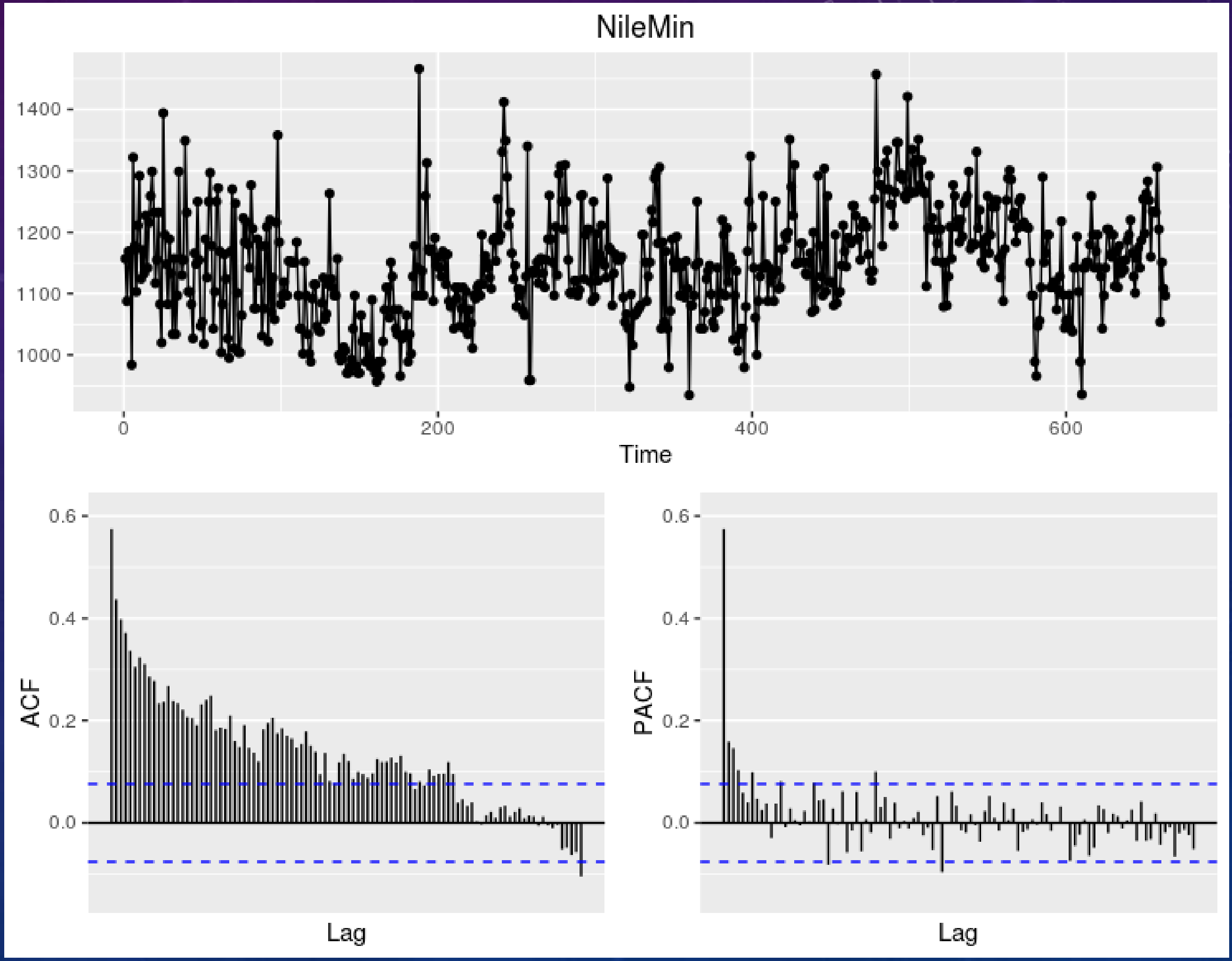
- There are 45 steps down
- The central column has markings on it for each Cubit up to 19 Cubits.

THE ASWAN DAM

Astronaut photograph ISS043-E-101953 was acquired on April 12, 2015, with a Nikon D4 digital camera using an 800 millimeter lens, and is provided by the ISS Crew Earth Observations Facility and the Earth Science and Remote Sensing Unit, Johnson Space Center. The image was taken by a member of the Expedition 43 crew.

NILE MINIMA MEASUREMENTS…

```
> data(NileMin)
> ggtsdisplay(NileMin,lag.max=100,main="Annual Nile Minimum Levels 622-1284")
```



HOW CAN WE ESTIMATE "D"

- d is a fraction – should be between -1 and +½
  - This will ensure "stationarity".
- Lots of methods!
- Lots of R packages!
  - ARFIMA, FRACDIFF, LONGMEMO, WAVESLIM, RUGARCH, FRACTAL

ESTIMATING "D"

- calcLM function at end of slide deck…

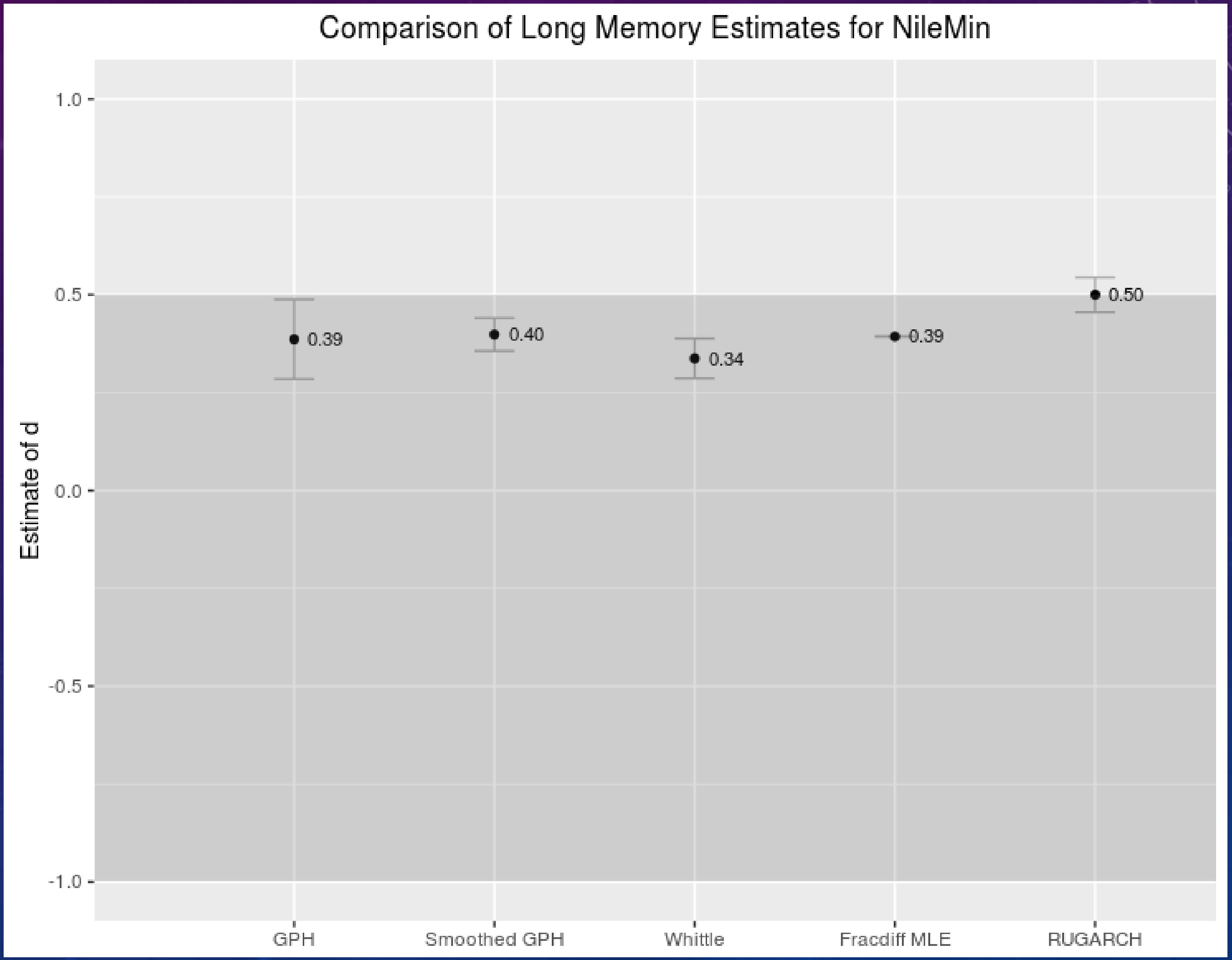
```
> calcLM(NileMin)
Wavelet Estimation failed.

          GPH d=0.3863 se=0.0519
Smoothed GPH d=0.3986 se=0.0215
     Whittle d=0.3374 se=0.0260
Fracdiff MLE d=0.3933 se=0.0000
     RUGARCH d=0.5000 se=0.0227
FD MLE AIC:  7515.52
RUGARCH AIC: 8063.18
```

- "Wavelet Estimation failed" error is common.
  - GPH = Geweke Porter-Hudak method
  - Whittle – has best theoretical results for smallest var.
  - MLE = Maximum Likelihood estimates.

  - Note for GPH – bandw.exp param should always be set to 0.8, never 0.5! (GPH recommended 0.5, but subsequent research showed this too conservative and results in much worse estimates than 0.8. But 0.5 is the default in the R package!). CalcLM function sets this to be 0.8.

calcLM PLOT


Comparison of Long Memory Estimates for NileMin
Estimate of d
1.0
0.5
0.0
-0.5
-1.0
0.39
0.40
0.34
0.39
0.50
GPH
Smoothed GPH
Whittle
Fracdiff MLE
RUGARCH

THINGS TO NOTE...

- On NileMin series... “zero” is not within the confidence intervals.
- Also all the estimates are in the “dark gray” area.
  - This implies there is no “trend” to be concerned about.
- Simulations show RU_GARCH is at least as good as any other method
  - AND its more flexible, and can fit a much wider range of models...

SIMULATION RESULTS (UNPUBLISHED)

- Simulate 1000 Series of length 512 from Gaussian dist. Bias Comparisons:

True d	ARFIMA	FRACDIFF	FDSPERIO	FDGPH5	FDGPH8	WAVELET	WHITTLE	RUGARCH
0.04	0.0145	0.0051	0.0360	0.0054	-0.0014	0.0023	0.0096	-0.0009
0.14	0.0347	0.0094	0.0398	0.0045	0.0022	0.0048	0.0297	0.0020
0.24	0.0547	0.0117	0.0424	0.0063	0.0023	0.0056	0.0497	0.0006
0.34	0.0703	0.0118	0.0382	-0.0096	-0.0046	0.0044	0.0651	-0.0041
0.44	0.0861	0.0169	0.0327	-0.0131	-0.0087	0.0052	0.0807	-0.0132
Avg	**0.0520**	**0.0110**	**0.0378**	**-0.0013**	**-0.0020**	**0.0044**	**0.0470**	**-0.0031**

- ARFIMA – from ARFIMA package
- FRACDIFF – old code but generally not too bad from FRACDIFF package
- fdSPERIO – from FRACDIFF package – a version of smoothed periodogram regression
- FDGPH5 – from FRACDIFF package – func fdGPH() – the original Periodogram regression with default bandwidth of 0.5
- FDGPH8 – fdGPH() with bandwidth of 0.8
- WAVELET – function fdp.mle() from WAVESLIM package
- RUGARCH package – a form of maximum likelihood estimate

MEAN SQUARED ERROR

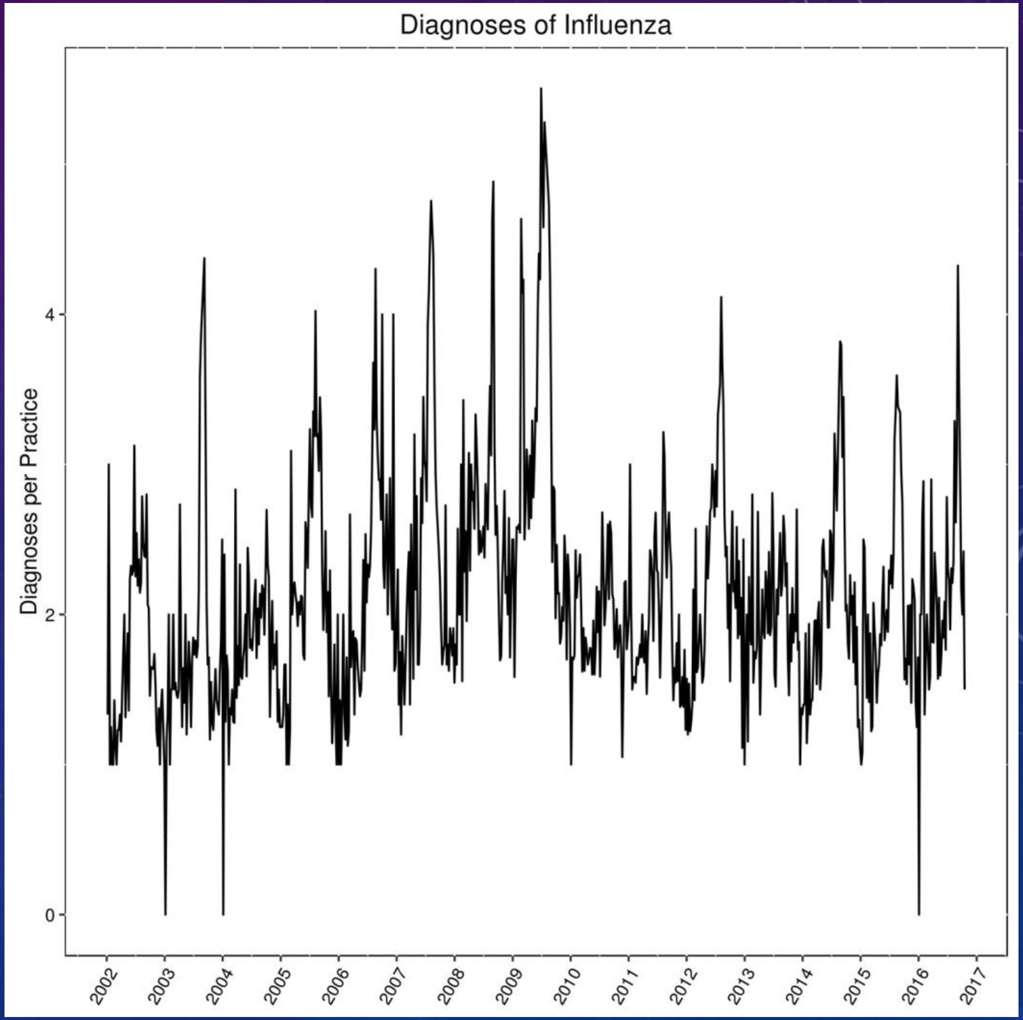
True d	ARFIMA	FRACDIFF	FDSPERIO	FDGPH5	FDGPH8	WAVELET	WHITTLE	RUGARCH
0.04	0.0319	0.0303	0.1390	0.1708	0.0584	0.0349	0.0300	0.0306
0.14	0.0459	0.0383	0.1450	0.1797	0.0596	0.0367	0.0423	0.0367
0.24	0.0621	0.0370	0.1464	0.1767	0.0595	0.0348	0.0577	0.0342
0.34	0.0769	0.0379	0.1487	0.1725	0.0589	0.0354	0.0722	0.0365
0.44	0.0909	0.0348	0.1453	0.1738	0.0573	0.0303	0.0859	0.0368
Avg	**0.0615**	**0.0357**	**0.1449**	**0.1747**	**0.0587**	**0.0344**	**0.0576**	**0.0350**

- Overall – RUGARCH is best – much lower MSE and very low bias.
- RUGARCH fits very wide range of models.
- “sister” package RMGARCH for multivariate models.
- Strongly suggest you read the “vignette” before starting to use it.

AN EXAMPLE – THANKS TO MEDICAL DIRECTOR!

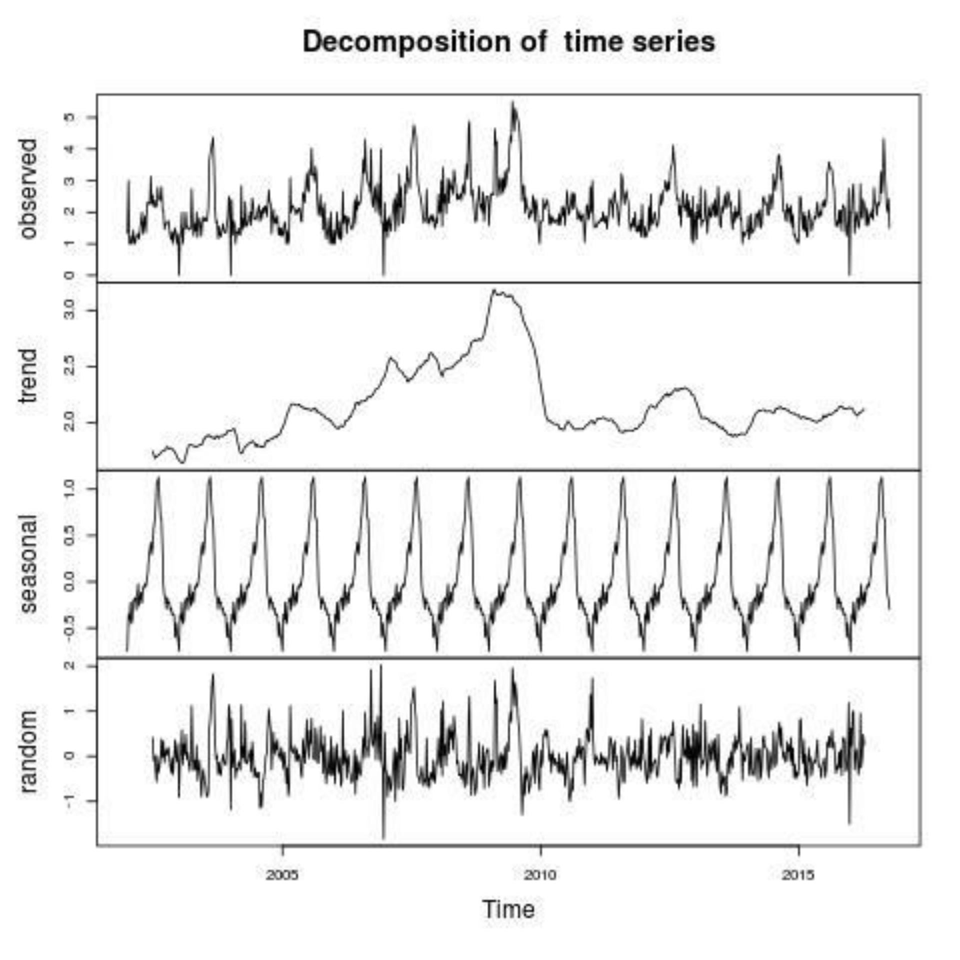
- Medical Director GPRN – anonymously collected GP clinical data.
  - Audited by Privacy Commissioner.
- We can trace cases of the Flu!
  - Weekly totals of Influenza Diagnoses.

GPRN INFLUENZA DIAGNOSES



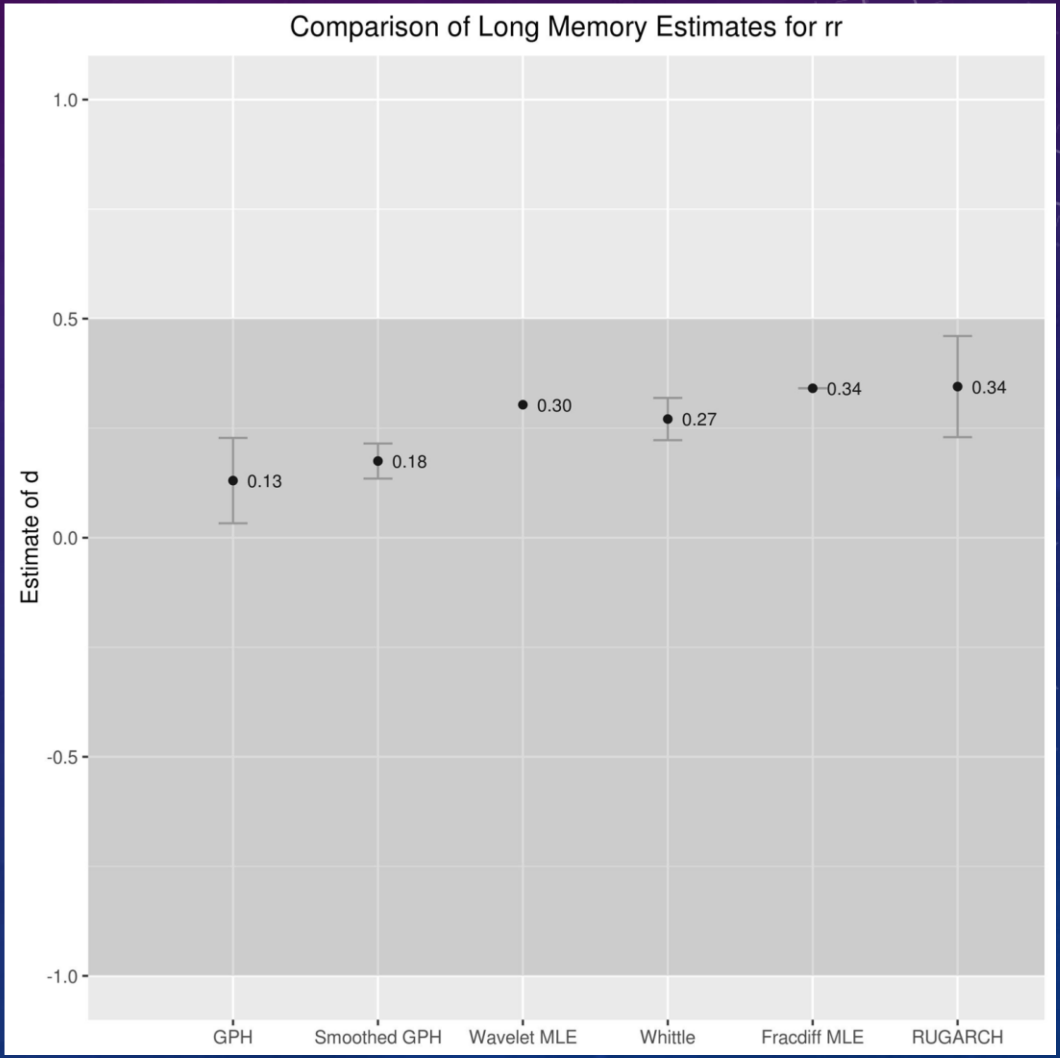
FIRST WE SEPARATE OUT THE TRENDS

```
> d<-decompose(sf$af)
> plot(d)
```



NOW WE CAN ANALYSE THE DE-TRENDED "RANDOM" COMPONENT

```
> rr<-d$random[!is.na(d$random)]
> calcLM(rr)
          GPH d=0.1304 se=0.0497
Smoothed GPH d=0.1750 se=0.0205
 Wavelet MLE d=0.3036 se=NA
     Whittle d=0.2709 se=0.0246
Fracdiff MLE d=0.3411 se=0.0000
     RUGARCH d=0.3450 se=0.0589
FD MLE AIC:   935.06
RUGARCH AIC: 934.30
```

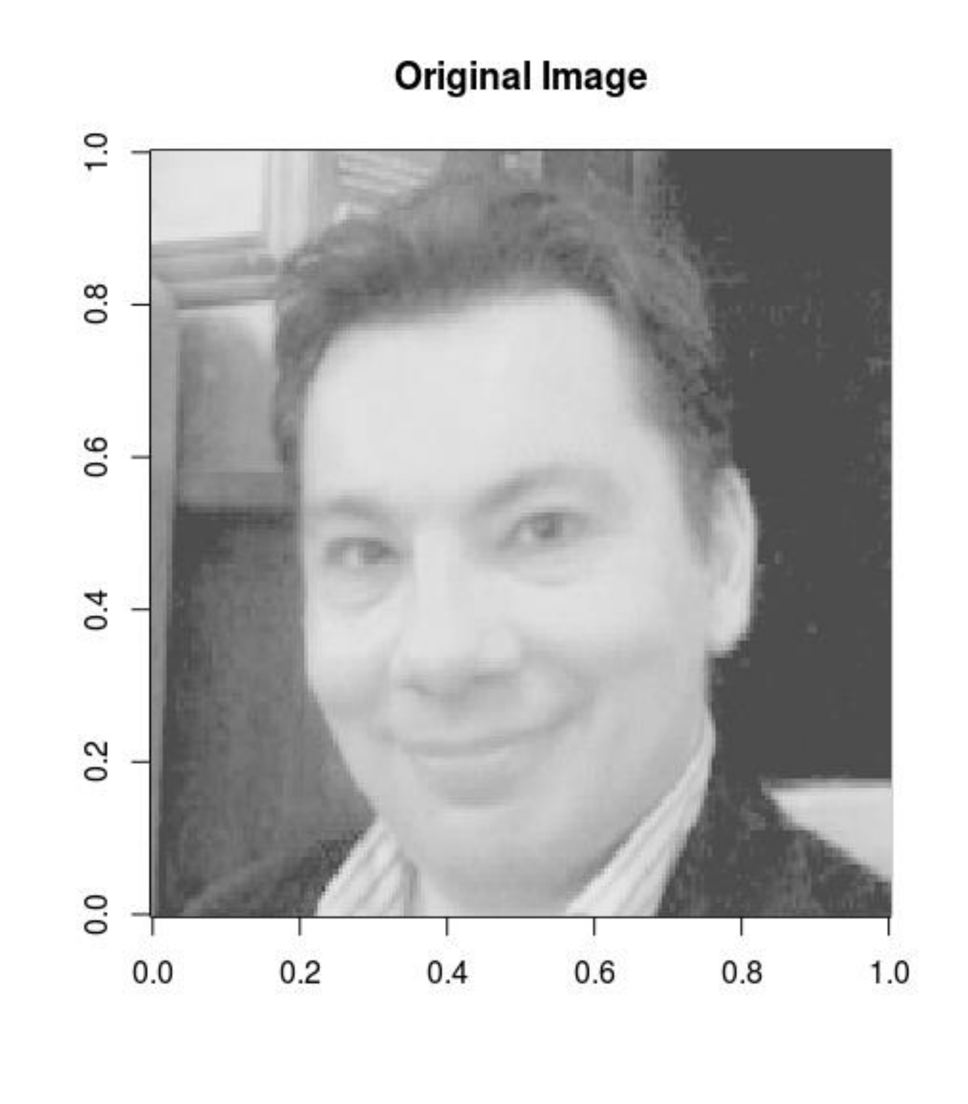


NON-TIME SERIES APPLICATIONS.

- These tools can be used for series where the points might be points in space rather than time.
- Often useful then to work in 2-D!
- Eg Photos

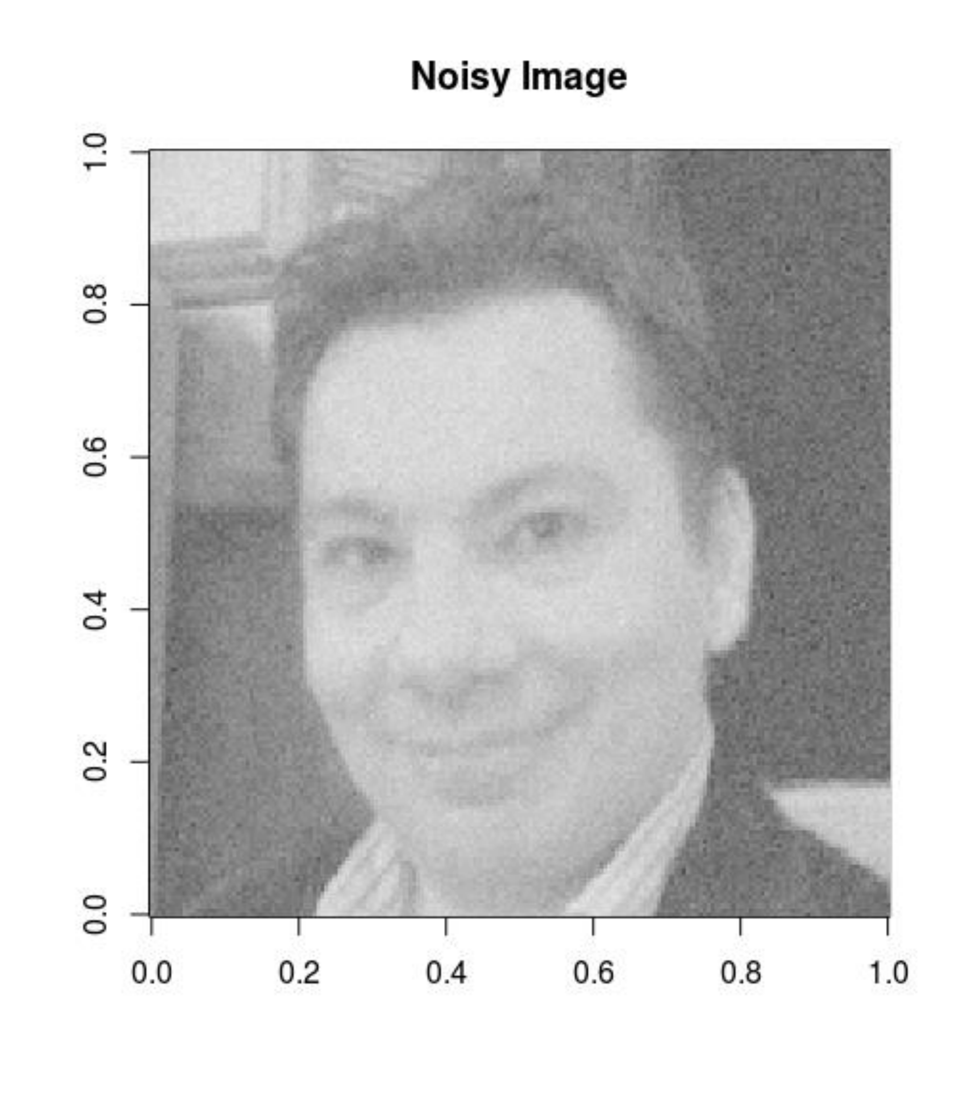
A PHOTO

```
library(jpeg)
ed<-readJPEG("/media/richard/USB DISK/ts/ed.jpg")
ed1<-t(ed[,,1])          # Transpose so image not on side
ed1<-ed1[,nrow(ed1):1]  # not upside down…
# Now draw image.
image(ed1, col=gray.colors(128), main="Original Image")
```



A NOISY PHOTO

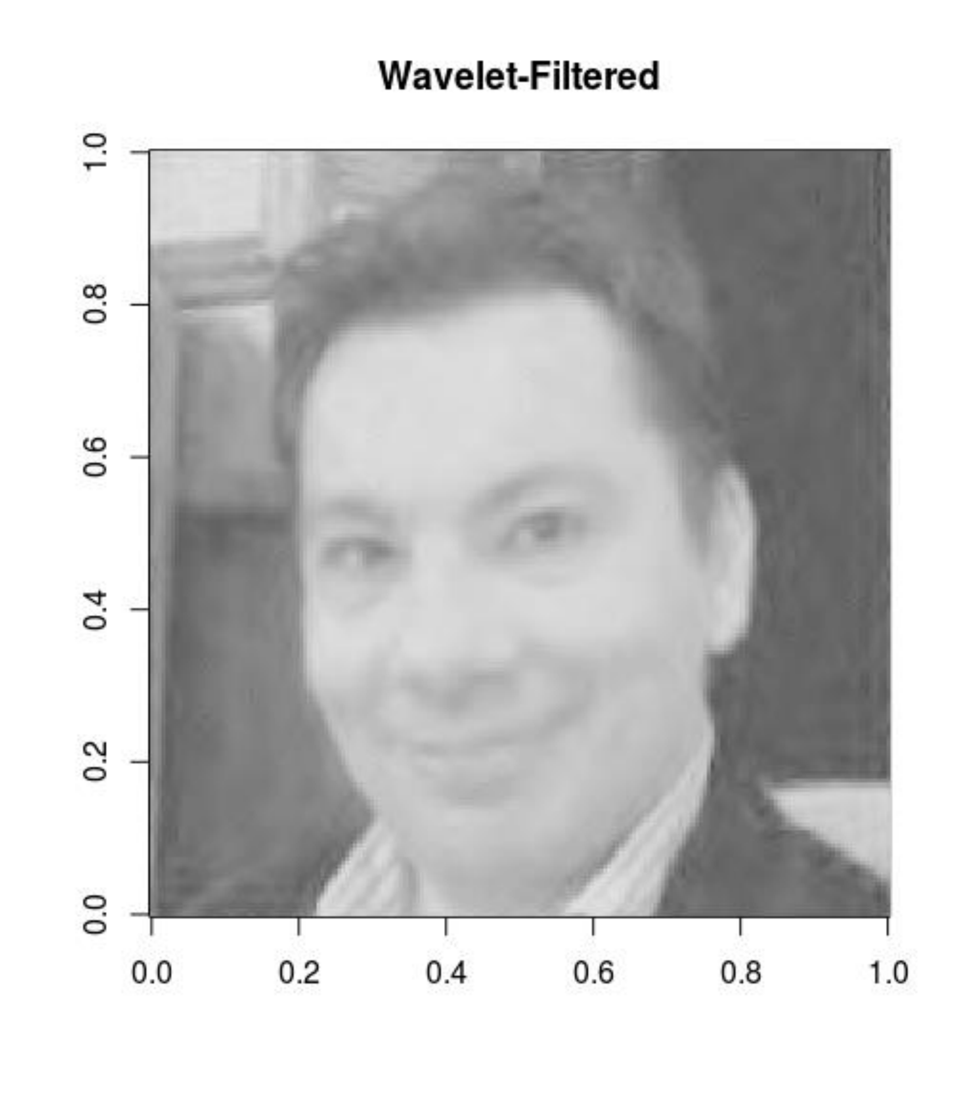
```
> ed.noise <- ed1 + matrix(rnorm(nrow(ed1)*ncol(ed1),
           d=0.05), nrow(ed1), ncol(ed1))
> image(ed.noise, col=gray.colors(128), main="Noisy Image")
```



A DE-NOISED PHOTO

```
> ed.denoise <- denoise.modwt.2d(ed.noise, wf="mb16",
method="long-memory", H=0.1, rule="hard")
> image(ed.denoise, col=gray.colors(128),
main="Wavelet-Filtered")
```

In the photo shown, we use “wavelets” to break down the image, and then look at “long memory” correlations throughout the image to attempt to rebuild the original from the noisy version...



OTHER APPLICATIONS

- Sediment Accumulation, Tree rings
- Acoustics, Diffusion, some astrophysical phenomena
  - Eg active galactic nuclei (AGN) like Cygnus X-1 – Xray patterns
- Econometric Modelling – Esp inflation
- Heart beat period/arrhythmias
- DNA Nucleotide sequences
- Ethernet Traffic
- Speech Recognition
- Climatic phenomena generally

ONE STEP FURTHER…

- Research now into Seasonal/Cyclical Long Memory
  - The ACF cycles through the long memory correlations.
- Also known as Gegenbauer series.
  - Only existing way to estimate in R is the “spp.mle()” in library “waveslim”. Only estimates a single cycle. Forecasting hard.
  - A research interest of mine. I will be looking to build a new R package in the next 12-18 months to support this.

calcLM FUNCTION

```
calcLM<-function(series) {
   seriesName<-deparse(substitute(series))
   gph<-fdGPH(series,bandw.exp=0.8)
   whittle<-WhittleEst(series)
   smoothed<-fdSperio(series, bandw.exp = 0.8, beta = 0.9)
   #fractal package
   fdwhittle<-FDWhittle(series)
   # For Wavelets need to pad series out to power of 2.
   len<-2^(as.integer(log(length(series),2))+1)
   wvlt<-list(par=list(0,0))
   tryCatch(wvlt<-fdp.mle(c(rep(0,len-length(series)),series),"mb8"),error = function(c) cat("Wavelet Estimation failed.\n\n"),
      warning = function(c) {x<-1})
   mle_fd<-fracdiff(series,nar=0,nma=0)
   #now RUGARCH
   aspec<-arfimaspec(mean.model=list(armaOrder=c(0,0), include.mean=FALSE, arfima=TRUE))
   rugarch.fit<-arfimafit(spec=aspec,data=series,solver="hybrid")
   dummy <- capture.output(series.stat<-stationarity(series))
   series.stat.pvals<-attr(series.stat,"pvals")

   if (series.stat.pvals[1]>0.05) cat("\nTests indicate series is stationary.\n\n") else
      cat(sprintf("\nWarning: Tests indicate series may not be stationary (p-val %0.4f)\n\n",series.stat.pvals[1]))

   df<-data.frame(   method=c("GPH","Smoothed GPH", "Wavelet MLE", "Whittle", "FDWhittle", "Fracdiff MLE", "RUGARCH"),
               d.est=c(gph$d, smoothed$d, ifelse(unlist(wvlt$par[1])==0,NA,unlist(wvlt$par[1])),
                      whittle$coefficients[1,1]-0.5, fdwhittle, mle_fd$d, rugarch.fit@fit$robust.matcoef[1,1]),
               se.est=c(gph$sd.as, smoothed$sd.as, NA, whittle$coefficients[1,2], NA, mle_fd$stderror.dpq,
                        rugarch.fit@fit$robust.matcoef[1,2]))
   df$lci<-df$d.est-1.96*df$se.est
   df$uci<-df$d.est+1.96*df$se.est
   df$method<-factor(df$method,levels=c("GPH","Smoothed GPH", "Wavelet MLE", "Whittle", "FDWhittle", "Fracdiff MLE", "RUGARCH"))
   p<-ggplot(df[!is.na(df$d.est),],aes(x=method))+ylim(-1.0,1.0)+
      geom_errorbar(aes(ymin=lci,ymax=uci),width=0.2,colour="darkgray")+
      geom_point(aes(y=d.est),color="black")+
      geom_text(aes(y=d.est,label=sprintf("%0.2f",d.est)),hjust=-0.4,size=3)+
      annotate("rect", xmin=0, xmax=Inf, ymin=-1.0, ymax=0.5, alpha = .2)+
      theme(axis.title.x = element_blank(),plot.title=element_text(color="black"),axis.text.x=element_text(angle=60,hjust=1,colour="black"))+
      labs(y = "Estimate of d")+
      ggtitle(bquote(paste("Comparison of Long Memory Estimates for ",.(seriesName))))
   print(p)
   for (i in 1:nrow(df)) if (!is.na(df$d.est[i])) cat(sprintf("%12s d=%0.4f se=%0.4f\n", df$method[i], df$d.est[i],df$se.est[i]))
   cat(sprintf("FD MLE AIC:  %0.2f\nRUGARCH AIC: %0.2f\n\n",-2.0*mle_fd$log.likelihood, rugarch.fit@fit$LLH*(-2)))
   cat("GPH is fracdiff::fdGPH()\nSmoothedGPH is fracdiff::fdSperio()\nWhittle is longmemo::WhittleEst()\n
FDWhittle is fractal::FDWhittle()\nWavelet MLE is waveslim:fdp.mle()\nFracdiff MLE is fracdiff::fracdiff()\n
RUGARCH is rugarch::arfimafit()\n\n")
}

data(NileMin) #Get the NileMinima data from the LongMemo package
calcLM(NileMin)
```

"R" LIBRARIES

```
library(forecast)
library(longmemo)
library(fracdiff)
library(ggplot2)
library(reshape2)
library(waveslim)
library(rugarch)
library(fractal)
```