

## Module 7: Models with changing variance and frequency models

Andrew Parnell, School of Mathematics and Statistics,  
University College Dublin

## Learning outcomes

- ▶ Understand how to fit ARCH, GARCH and SVM models in JAGS
- ▶ Know how to check assumptions for these methods
- ▶ Understand the basis of seasonal models
- ▶ Know the difference between time and frequency domain models and be able to implement a Fourier model

Relevant JAGS file:

jags\_ARCH.R

jags\_GARCH.R

jags\_SVM.R

jags\_Fourier.R

# General principles of models for changing variance

- ▶ So far we have looked at models where the mean changes but the variance is constant:

$$y_t \sim N(\mu_t, \sigma^2)$$

- ▶ In this module we look at methods where instead:

$$y_t \sim N(\alpha, \sigma_t^2)$$

- ▶ These are:
  - ▶ Autoregressive Conditional Heteroskedasticity (ARCH)
  - ▶ Generalised Autoregressive Conditional Heteroskedasticity (ARCH)
  - ▶ Stochastic Volatility Models (SVM)

## Extension 1: ARCH

- ▶ An ARCH(1) Model has the form:

$$\sigma_t^2 = \gamma_0 + \gamma_1 \epsilon_{t-1}^2$$

where  $\epsilon_t$  is the residual, just like an MA model

- ▶ Note that  $\epsilon_t = y_t - \alpha$  so the above can be re-written as:

$$\sigma_t^2 = \gamma_0 + \gamma_1 (y_{t-1} - \alpha)^2$$

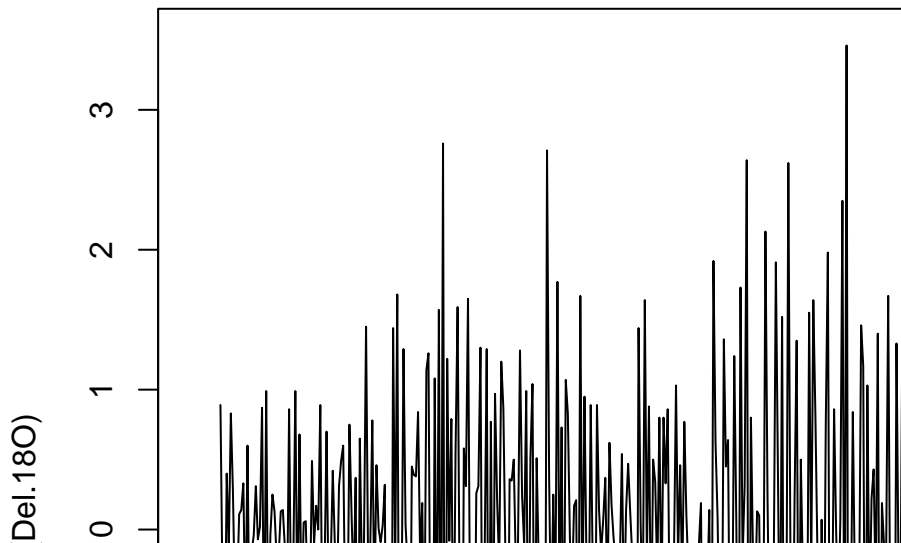
- ▶ The variance at time  $t$  thus depends on the previous value of the series (hence the autoregressive in the name)
- ▶ The residual needs to be squared to keep the variance positive.
- ▶ The parameters  $\gamma_0$  and  $\gamma_1$  also need to be positive, and usually  $\gamma_1 \sim U(0, 1)$

## JAGS code for ARCH models

```
model_code = '  
model  
{  
  # Likelihood  
  for (t in 1:T) {  
    y[t] ~ dnorm(alpha, tau[t])  
    tau[t] <- 1/pow(sigma[t], 2)  
  }  
  sigma[1] ~ dunif(0, 10)  
  for(t in 2:T) {  
    sigma[t] <- sqrt( gamma_1 + gamma_2 * pow(y[t-1] - alpha  
  }  
  
  # Priors  
  alpha ~ dnorm(0.0, 0.01)  
  gamma_1 ~ dunif(0, 10)  
  gamma_2 ~ dunif(0, 1)  
}
```

## Example: ice core data

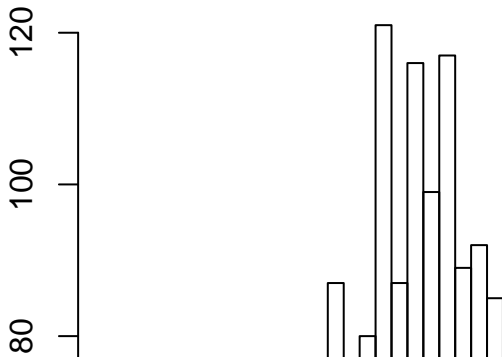
```
with(ice2,plot(Age[-1],diff(Del.180),type='l'))
```



## Example: ice core output

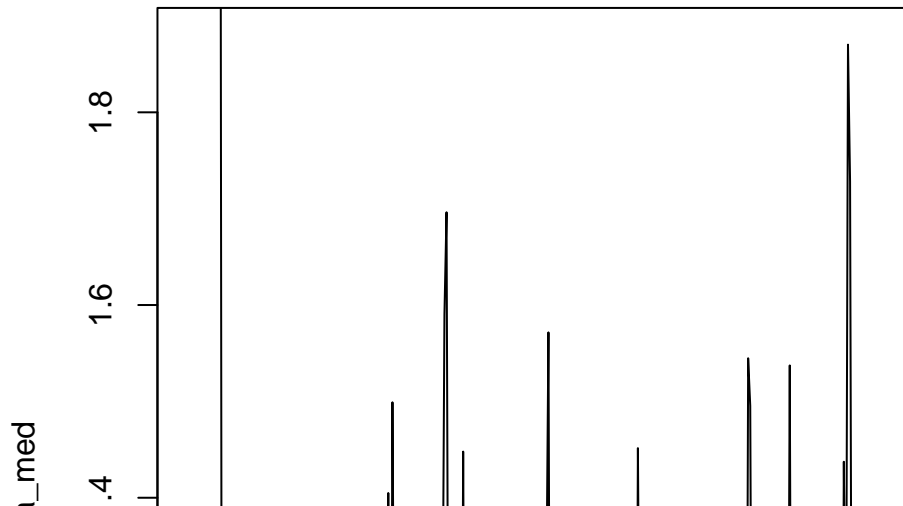
```
par(mfrow=c(1,2))  
hist(real_data_run_1$BUGSoutput$sims.list$gamma_1, breaks=30)  
hist(real_data_run_1$BUGSoutput$sims.list$gamma_2, breaks=30)
```

istogram of real\_data\_run\_1\$BUGSoutput\$sims.list



## Example: posterior standard deviations

```
sigma_med = apply(real_data_run_1$BUGSoutput$sims.list$sign  
plot(ice2$Age[-1],sigma_med,type='l',ylim=range(c(sigma_med
```





## From ARCH to GARCH

- ▶ The Generalised ARCH model works by simply adding the previous value of the variance, as well as the previous value of the observation
- ▶ The GARCH(1,1) model thus has:

$$\sigma_t^2 = \gamma_1 + \gamma_2(y_{t-1} - \alpha)^2 + \gamma_3\sigma_{t-1}^2$$

- ▶ Strictly speaking  $\gamma_1 + \gamma_2 < 1$  though like the stationarity conditions in ARIMA models we can relax this assumption and see if the data support it
- ▶ It's conceptually easy to extend to general GARCH(p,q) models which add in extra previous lags

## Example of using the GARCH(1,1) model

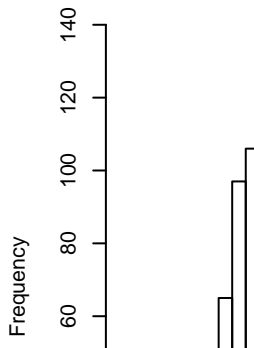
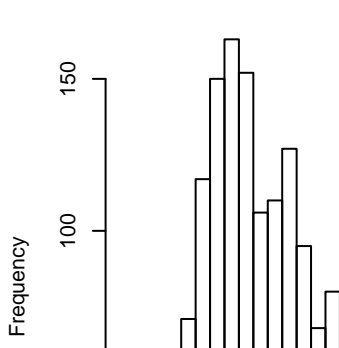
```
model_code = '  
model  
{  
  # Likelihood  
  for (t in 1:T) {  
    y[t] ~ dnorm(alpha, tau[t])  
    tau[t] <- 1/pow(sigma[t], 2)  
  }  
  sigma[1] ~ dunif(0,10)  
  for(t in 2:T) {  
    sigma[t] <- sqrt( gamma_1 + gamma_2 * pow(y[t-1] - alpha  
  }  
  
  # Priors  
  alpha ~ dnorm(0.0, 0.01)  
  gamma_1 ~ dunif(0, 10)  
  gamma_2 ~ dunif(0, 1)  
  gamma_3 ~ dunif(0, 1)
```

## Using the ice core data again

Have a look at the ARCH parameters;

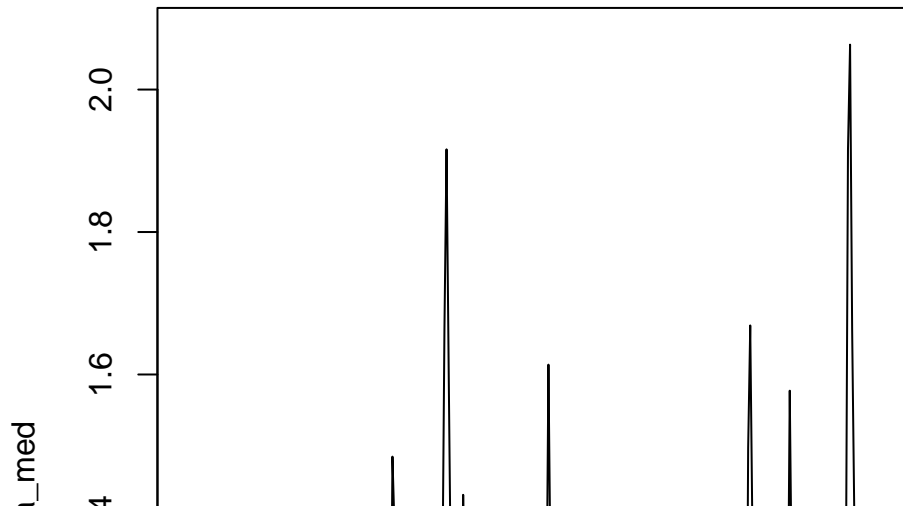
```
par(mfrow=c(1,3))  
hist(real_data_run_2$BUGSoutput$sims.list$gamma_1, breaks=30)  
hist(real_data_run_2$BUGSoutput$sims.list$gamma_2, breaks=30)  
hist(real_data_run_2$BUGSoutput$sims.list$gamma_3, breaks=30)
```

ogram of real\_data\_run\_2\$BUGSoutput\$sims.list\$ogram of real\_data\_r



## Posterior median standard deviation

```
sigma_med = apply(real_data_run_2$BUGSoutput$sims.list$sigma_med, 2, FUN=function(x) {  
  plot(ice2$Age[-1], sigma_med, type='l', ylim=range(c(sigma_med, 1.4)))  
})
```



## Compare with DIC

```
with(r_1, print(c(DIC, pD)))
```

```
## [1] 2152.979029    3.713786
```

```
with(r_2, print(c(DIC, pD)))
```

```
## [1] 2141.418813    7.090526
```

- Suggests full GARCH model is best, despite the extra parameters

# Stochastic Volatility Modelling

- ▶ Both ARCH and GARCH propose a deterministic relationship for the current variance parameter
- ▶ By contrast a Stochastic Volatility Model (SVM) models the variance as its own *stochastic process*
- ▶ SVMs, ARCH and GARCH are all closely linked if you go into the bowels of the theory
- ▶ The general model structure is often written as:

$$y_t \sim N(\alpha, \exp(h_t))$$

$$h_t \sim N(\mu + \phi(h_{t-1} - \mu), \sigma^2)$$

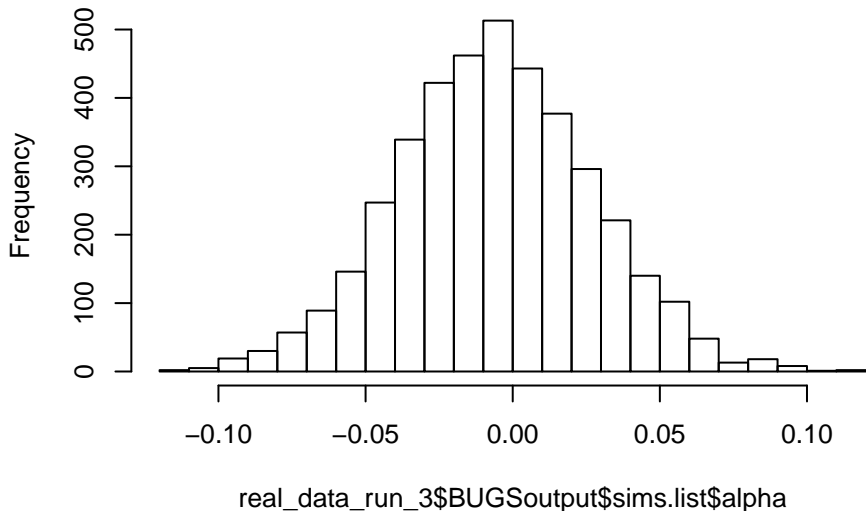
- ▶ You can think of an SVM being like a GLM but with a log link on the variance parameter

## JAGS code for the SVM model

```
model_code = '  
model  
{  
  # Likelihood  
  for (t in 1:T) {  
    y[t] ~ dnorm(alpha, tau_h[t])  
    tau_h[t] <- 1/exp(h[t])  
  }  
  h[1] <- mu  
  for(t in 2:T) {  
    h[t] ~ dnorm(mu + phi * (h[t-1] - mu), tau)  
  }  
  
  # Priors  
  alpha ~ dnorm(0, 0.01)  
  mu ~ dnorm(0, 0.01)  
  phi ~ dunif(-1, 1)  
  tau <- 1/pow(sigma, 2)
```

## Example of SVMs and comparison of DIC

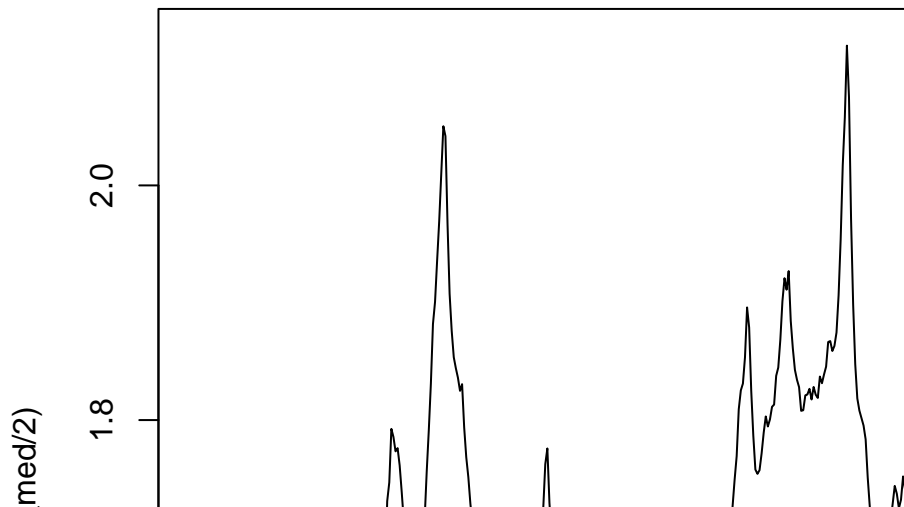
**Histogram of `real_data_run_3$BUGSoutput$sims.list$a`**





## Plot of $h$

```
h_med = exp(apply(real_data_run_3$BUGSoutput$sims.list$h, 2,  
plot(ice2$Age[-1], exp(h_med/2), type='l', ylim=range(c(exp(h_
```



## Comparison with previous models

```
with(r_1, print(c(DIC, pD)))
```

```
## [1] 2152.979029    3.713786
```

```
with(r_2, print(c(DIC, pD)))
```

```
## [1] 2141.418813    7.090526
```

```
with(r_3, print(c(DIC, pD)))
```

```
## [1] 2239.756  164.230
```

Note quite as good, and many extra parameters due to  $h$ !

# Seasonal time series

- ▶ So far we haven't covered how to deal with data that are *seasonal* in nature
- ▶ These data generally fall into two categories:
  1. Data where we know the frequency or frequencies (e.g. monthly data on a yearly cycle, frequency = 12)
  2. Data where we want to estimate the frequencies (e.g. climate time series, animal populations, etc)
- ▶ The former are much simpler, as we can e.g. use month as a covariate in an ARIMAX model, perform a seasonal difference with an ARIMA, or fit a full seasonal ARIMA model (though the JAGS code for this gets complicated)
- ▶ The latter are much more interesting. The ACF and PACF can help, but we can usually do much better by creating a *power spectrum*

# Methods for estimating frequencies

- ▶ The most common way to estimate the frequencies in a time series is to decompose it in a *Fourier Series*
- ▶ We write:

$$y_t = \alpha + \sum_{k=1}^K [\beta_k \sin(2\pi t f_k) + \gamma_k \cos(2\pi t f_k)] + \epsilon_t$$

- ▶ Each one of the terms inside the sum is called a *harmonic*. We decompose the series into a sum of sine and cosine waves rather than with AR and MA components
- ▶ Each sine/cosine pair has its own frequency  $f_k$ . If the corresponding coefficients  $\beta_k$  and  $\gamma_k$  are large we might believe this frequency is important

## Estimating frequencies via a Fourier model

- ▶ It's certainly possible to fit the model in the previous slide in JAGS, as it's just a linear regression model with clever explanatory variables
- ▶ However, it can be quite slow to fit and, if the number of frequencies  $K$  is high, or the frequencies are close together, it can struggle to converge
- ▶ More commonly, people repeatedly fit the simpler model:

$$y_t = \alpha + \beta \sin(2\pi t f_k) + \gamma \cos(2\pi t f_k) + \epsilon_t$$

for lots of different values of  $f_k$ . Then calculate the *power spectrum* as  $P(f_k) = \frac{\beta^2 + \gamma^2}{2}$ . Large values of the power spectrum indicate important frequencies

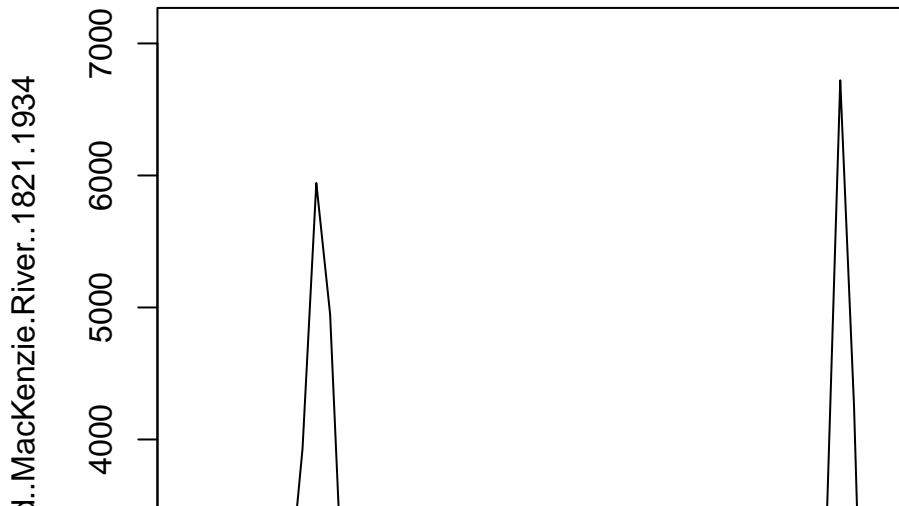
- ▶ It's much faster to do this outside of JAGS, using other methods, but we will stick to JAGS

## JAGS code for a Fourier model

```
model_code = '  
model  
{  
  # Likelihood  
  for (t in 1:T) {  
    y[t] ~ dnorm(mu[t], tau)  
    mu[t] <- alpha + beta * cos( 2 * pi * t * f_k ) + gamma  
  }  
  P = ( pow(beta, 2) + pow(gamma, 2) ) / 2  
  
  # Priors  
  alpha ~ dnorm(0.0,0.01)  
  beta ~ dnorm(0.0,0.01)  
  gamma ~ dnorm(0.0,0.01)  
  tau <- 1/pow(sigma,2) # Turn precision into standard deviation  
  sigma ~ dunif(0.0,100.0)  
}
```

## Example: the Lynx data

```
lynx = as.ts(dmseries('http://data.is/Ky69xY'))  
plot(lynx)
```



## Code to run the JAGS model repeatedly

```
periods = 5:40
K = length(periods)
f = 1/periods
Power = rep(NA,K)
model_parameters = c("P")

for (k in 1:K) {
  curr_model_data = list(y = as.vector(lynx[,1]),
                        T = length(lynx),
                        f_k = f[k],
                        pi = pi)

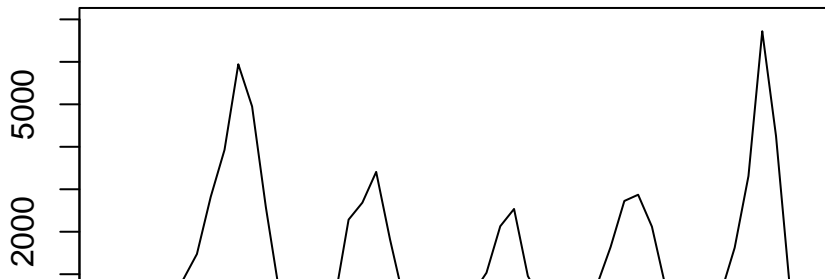
  model_run = jags(data = curr_model_data,
                  parameters.to.save = model_parameters,
                  model.file=textConnection(model_code),
                  n.chains=4,
                  n.iter=1000,
                  n.burnin=200
```



## Plotting the periodogram

```
par(mfrow = c(2, 1))  
plot(lynx)  
plot(f, Power, type='l')  
axis(side = 3, at = f, labels = periods)
```

.trapped..Mackenzie.River..18;



## Bayesian vs traditional frequency analysis

- ▶ For quick and dirty analysis, there is no need to run the full Bayesian model, the R function `periodogram` in the TSA package will do the job
- ▶ However, the big advantage (as always with Bayes) is that we can also plot the uncertainty in the periodogram, or combine the Fourier model with other modelling ideas (e.g. ARIMA)
- ▶ There are much fancier versions of frequency models out there (e.g. Wavelets, or frequency selection models) which can also be fitted in JAGS but require a bit more time and effort
- ▶ These Fourier models work for continuous time series too

# Summary

- ▶ We now know how to fit models with changing variance using a variety of techniques
- ▶ We can combine these new models with all the techniques we have previously learnt
- ▶ With known periodicity we can use seasonal differencing (or seasonal AR terms, not covered here but can be upon request)
- ▶ We've seen a basic Fourier model for estimating frequencies via the Bayesian periodogram