Class 1: Models for continuous time series: Brownian Motion and Ornstien Uhlenbeck processes

Andrew Parnell
andrew.parnell@ucd.ie

Learning outcomes

- Understand the differences between continuous and discrete time series
- Understand the basics of Brownian motion
- Extend the AR model to the Ornstein-Uhlenbeck process
- Ito formulation vs Euler-Maruyama methods
- Forecasting and interpolation for continuous time models

Discrete time vs continuous time

- Almost all of the models we have studied so far assume that time is discrete, e.g. $t = 1, 2, 3, \ldots$, or year $= 1850, 1851, \ldots$.
- Many real world time series do not work on discrete times scales. Instead the time value t might be any integer or non-integer value.
- Continuous time series might occur because:
  1. The data are recorded in irregular time, e.g. the geese isotope data
  2. The data contain many missing values. We could use the `NA` trick but if there are too many this becomes impractical
- Really all time series are recorded in continuous time, we just sometimes approximate them onto a grid. There can be lots of subtle issues when data are aggregated incorrectly

Some models for continuous time we have met

We have already seen three methods which work for data recorded in continuous time:

1. Linear and logistic regression
2. Fourier Methods
3. Seasonal factor models

However for all of these it could be argued that they are not true time series models since they borrow predictive strength from both the future and the past

Brownian motion

- Perhaps the simplest of all continuous time series models is that of *Brownian Motion* (BM)
- As we are now working in continuous time we write the time series as $y(t)$ rather than y_t to allow for y to be a function of continuous t
- The likelihood for BM is:

$$y(t) - y(t-s) \sim N(0, s\sigma^2)$$

where s is any positive value. Note that if s is 1 we have the standard random walk model

- You can also add in a *drift* parameter and re-write the model as:

$$y(t) - y(t-s) \sim N(\alpha s, s\sigma^2)$$

JAGS code for simple Brownian Motion

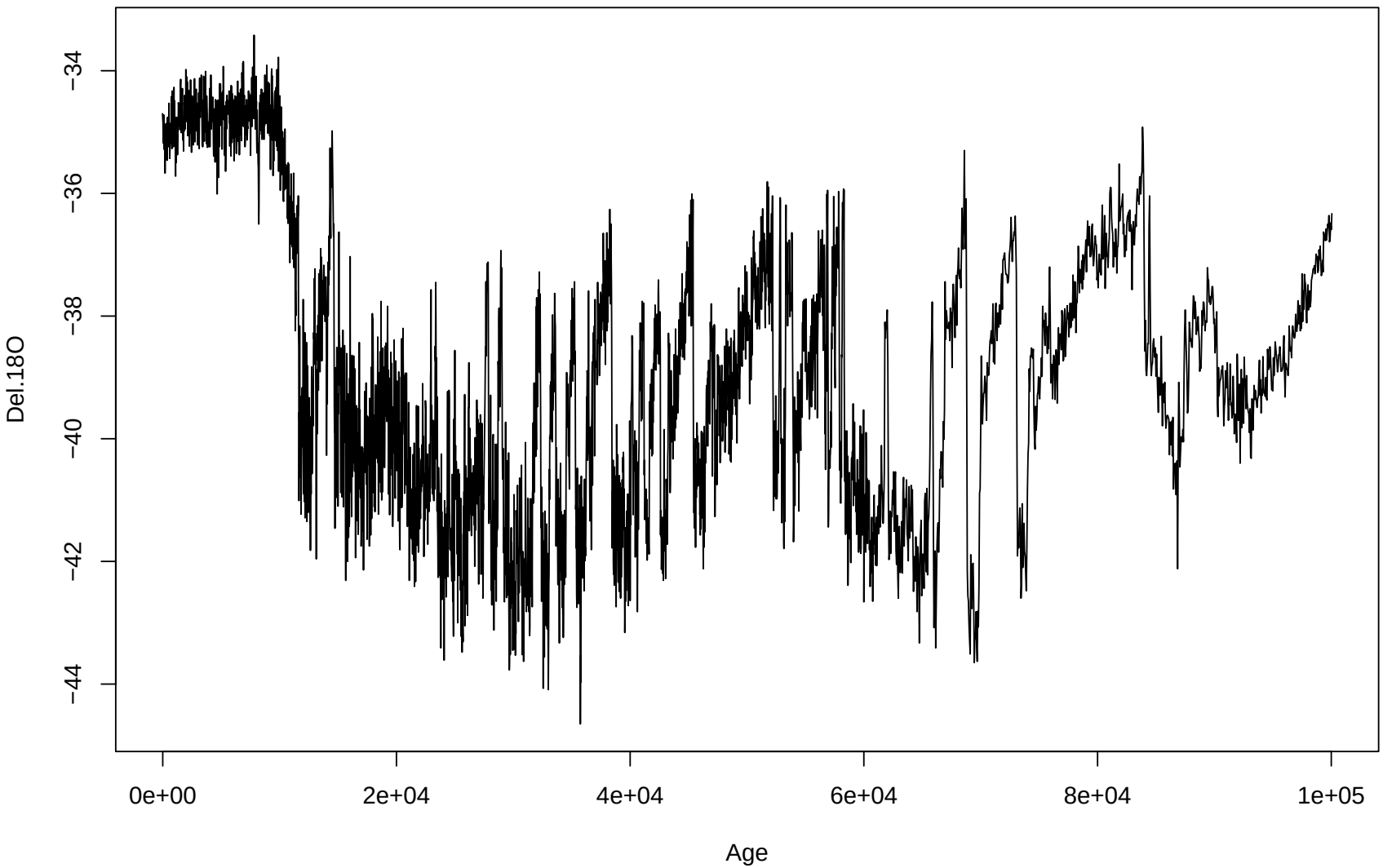
```
model_code = '
model
{
  # Likelihood
  for (i in 2:T) {
  y[i] ~ dnorm( alpha * (t[i] - t[i-1]) + y[i-1],
                sigma_rw[i]^-2 )
  sigma_rw[i] <- sigma * sqrt(t[i] - t[i-1])
  }

  # Priors
  alpha ~ dnorm(0, 10^-2)
  sigma ~ dunif(0, 10)
}
'
```

- You can alternatively write this in terms of differences of y and t

Example: ice core data

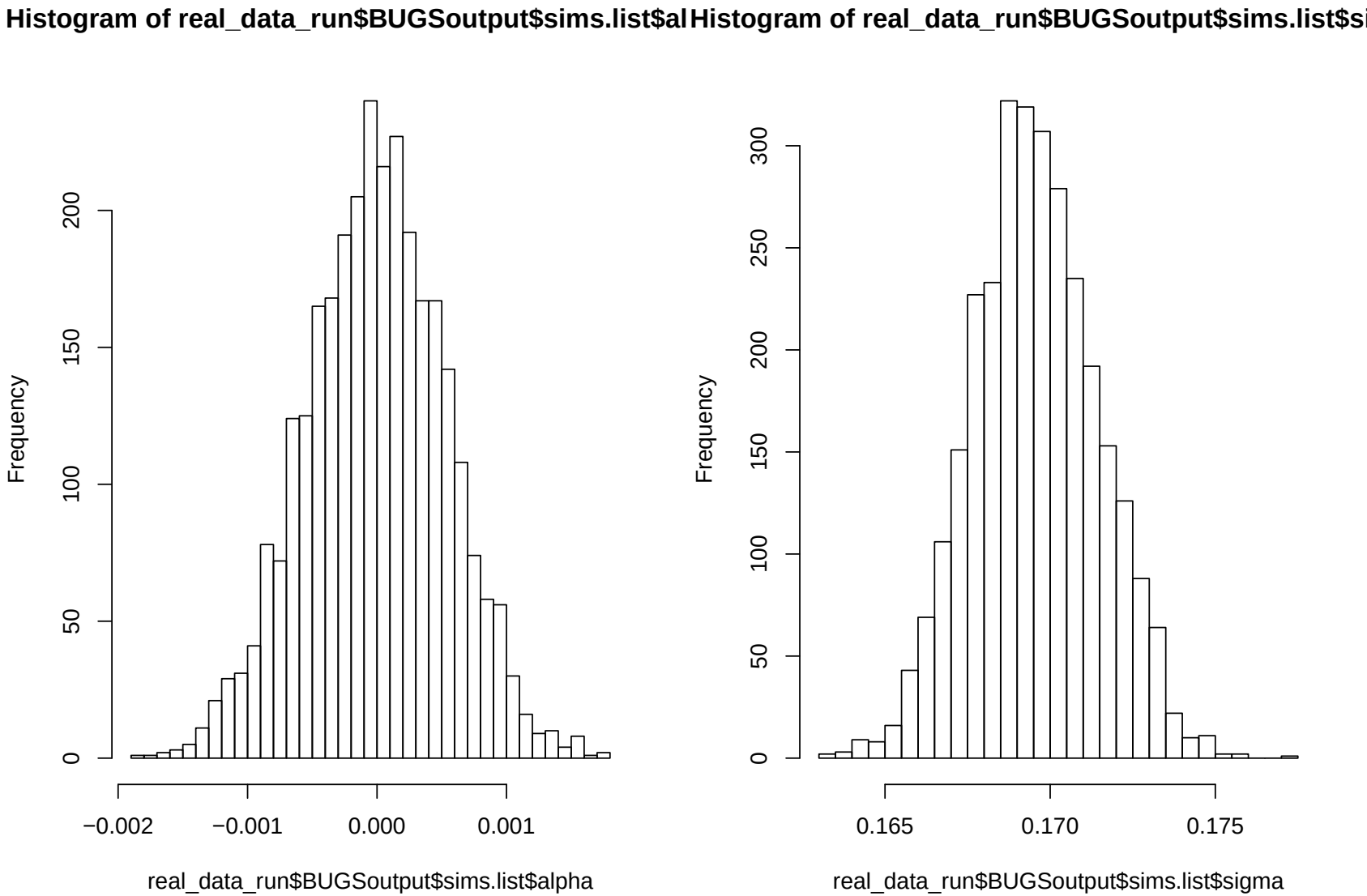
```
ice = read.csv('../../data/GISP2_20yr.csv')
with(ice, plot(Age, Del.18O, type = 'l'))
```



Running the model in JAGS

```
# Set up the data
real_data = with(ice,
                 list(y = Del.18O,
                      T = nrow(ice),
                      t = Age))
# Run the model
real_data_run = jags(data = real_data,
                     parameters.to.save = c("alpha",
                                            "sigma"),
                     model.file =
                       textConnection(model_code))
```

Plot the results



Interpolation

- We can use the NA trick to create a new set of times at which we wish to predict $\delta^{18}O$
- We need to be careful that we don't give JAGS any time values which have 0 differences as this will cause it to crash

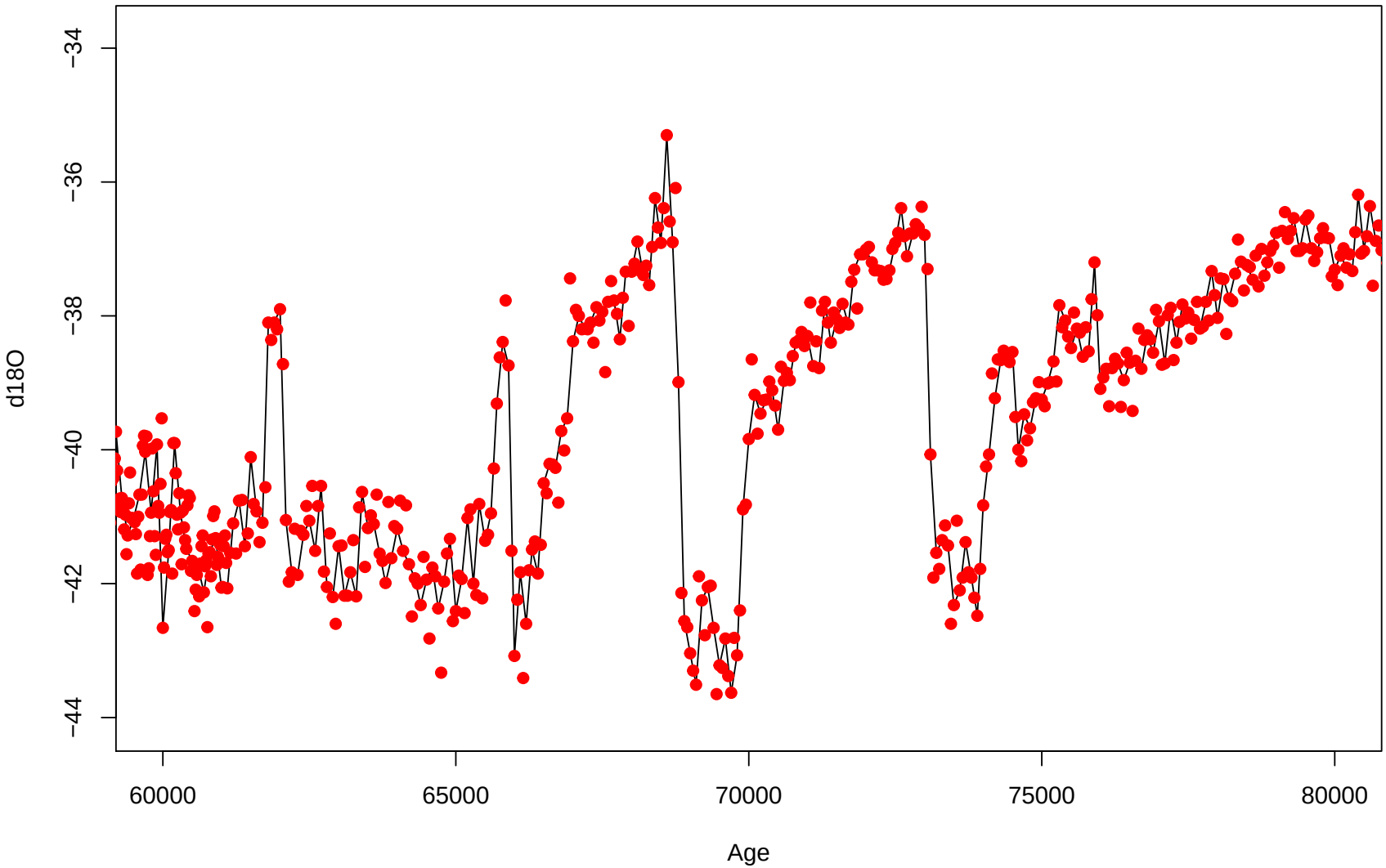
```
t_ideal = seq(0+0.01,max(ice$Age)+0.01, by = 100) # 100 year gri
y_ideal = rep(NA, length(t_ideal))
t_all = c(ice$Age, t_ideal)
y_all = c(ice$Del.18O, y_ideal)
o = order (t_all)
t_all[o][1:10]
```

```
##  [1] -20.00   0.00   0.01  20.00  40.00  60.00
##  [7]  80.00 100.00 100.01 120.00
```

```
y_all[o][1:10]
```

```
##  [1] -34.70 -34.78     NA -34.84 -35.16 -35.18
##  [7] -34.72 -35.28     NA -35.28
```

Interpolation plots



The Ornstein Uhlenbeck process

- One extension of Brownian Motion is called the *Ornstein-Uhlenbeck* (OU) process
- It can also be thought of as the continuous time version of the AR(1) process
- The likelihood is:

$$y(t) - y(t-s) \sim N(\theta(\alpha - y(t-s))s, s\sigma^2)$$

- It looks very much like the BM model but with an extra parameter θ which controls the dependence of $y(t)$ on $y(t-s)$ according to how far away it is
- With a bit of algebra if you set $s = 1$ above you end up with the AR(1) model
- Like the AR(1) model, θ needs to be between -1 and 1 to be stationary, but in practice can go beyond that range

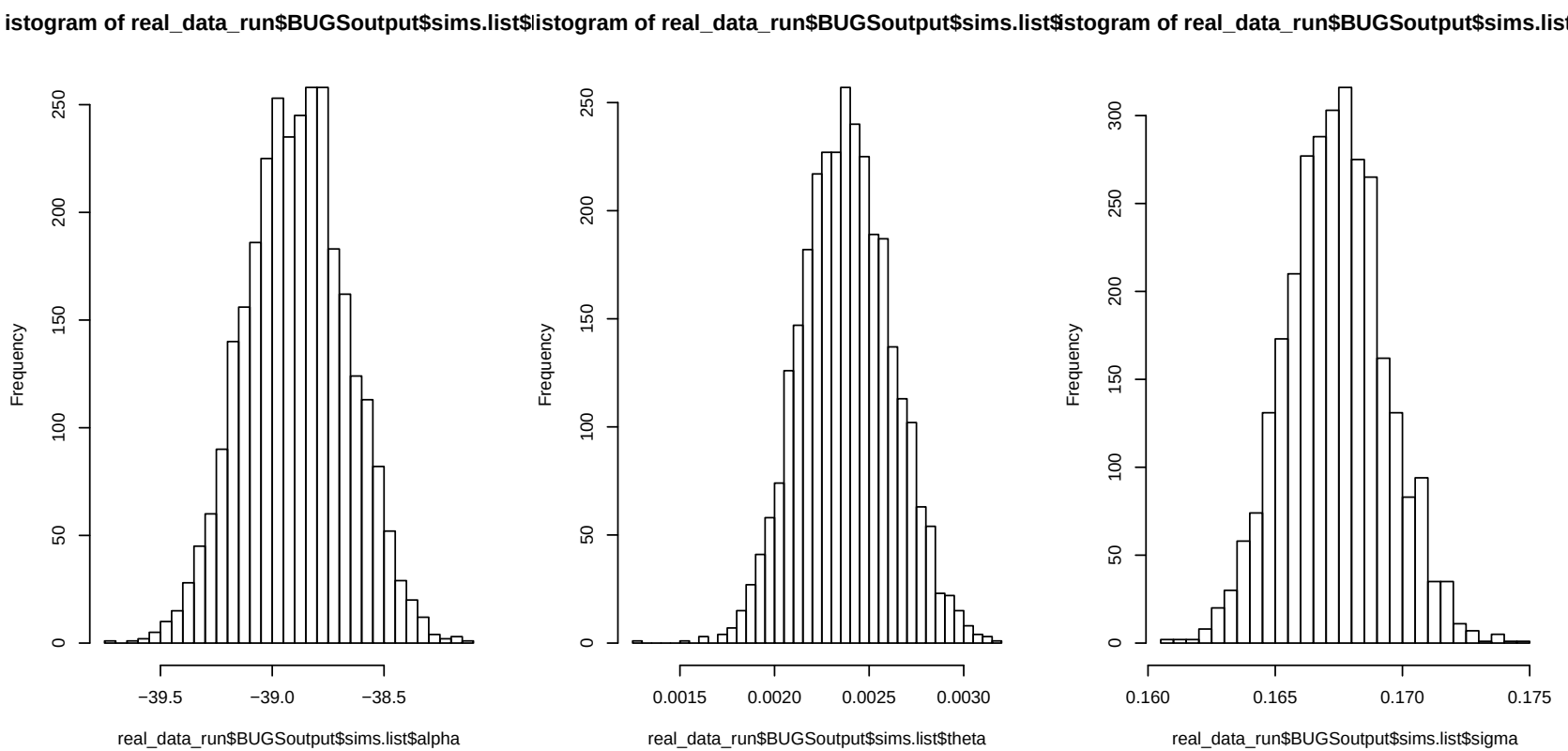
JAGS code for the OU process

```
model_code = '
model
{
  # Likelihood
  for (i in 2:T) {
    y[i] ~ dnorm( theta * (alpha - y[i-1]) *
                  (t[i] - t[i-1]) + y[i-1],
                  sigma_ou[i]^-2 )
    sigma_ou[i] <- sigma * sqrt(t[i] - t[i-1])
  }

  # Priors
  alpha ~ dnorm(0, 100^-2)
  theta ~ dnorm(0, 10^-2)
  sigma ~ dunif(0, 100)
}
'
```

Ice core output

```
par(mfrow=c(1,3))
hist(real_data_run$BUGSoutput$sims.list$alpha, breaks=30)
hist(real_data_run$BUGSoutput$sims.list$theta, breaks=30)
hist(real_data_run$BUGSoutput$sims.list$sigma, breaks=30)
```



Ito vs Euler-Marayuma forms

- You will often see, e.g. the BM model written as:

$$dy = \alpha dt + \sigma dW(t)$$

- This is sometimes called *Ito* form and is a stochastic differential equation with W a standard BM (i.e. with unit variance 1 and no drift)
- An alternative is to discretise the equation in *Euler-Marayama* form:

$$y(t) - y(t-s) = \alpha s + \sigma(W(t) - W(t-s))$$

- Writing it in this form makes it easier to see the likelihood version we used:

$$y(t) - y(t-s) \sim N(\alpha s, s\sigma^2)$$

- The Ito format for the OU process is:

$$dy = \theta(\alpha - y)dt + \sigma dW(t)$$

More complex stochastic equations

- Lots of the harder population dynamic and diffusion models used in ecology can be written in Ito format, discretised using Euler-Marayuma, and then fitted in JAGS or similar
- For example the Lotka-Volterra (predator-prey) model with diffusion (from Arato 2003):

$$dN_1(t) = (b_1 - a_1 N_2(t))N_1(t)dt + \sigma_1 dW(t)$$

$$dN_2(t) = (b_2 - a_2 N_1(t))N_2(t)dt + \sigma_2 dW(t)$$

  where N_1 is the number of prey, N_2 is the number of predators, and a, b, σ are all parameters
- Stan has some special modules that allow specific differential equations (even non-stochastic ones) to be included in the model

Continuous time stochastic volatility models

- Another way of extending these models is to give the standard deviation of the Brownian Motion its own stochastic process, just like the SVMs we met yesterday
- For example:

$$dy = \alpha dt + \sigma(dt)dW(t)$$

- You need to be careful about the choice of stochastic process on σ

- One good choice is the Inverse Gaussian Process which produces an amazingly flexible array of distributional shapes it can accomplish

Summary

- We have covered some methods for continuous time series including Brownian Motion and Ornstein-Uhlenbeck
- These are extensions of some of the discrete time methods we have already met, such as the random walk and the AR(1) process
- We have looked at how these models can be written out in Ito and Euler-Marayama format