STATS 331

Introduction to Bayesian Statistics
Semester 2, 2016

Introduction to JAGS

What is JAGS?

What is it for?

- JAGS is a general purpose computer program for doing MCMC sampling
- You tell it the prior, the likelihood and the data and it will do everything for you!
- It uses Metropolis and some other stuff (Gibbs sampling, slice sampling, ...)

Alternatives to JAGS

- WinBUGS/OpenBUGS
- Most models will work in both
- WinBUGS is older, but has a GUI. OpenBUGS also only works in Windows

- Stan is becoming popular. Its language is a bit different.

A Binomial Experiment

- An honours student in statistics ran some code $N = 100$ times. Success occurred on $x = 90$ of the runs. Modelling this as a binomial experiment, what was the success probability θ?

- Yes, we've done this before. SO many times

JAGS Syntax

```
# Define a model
model
{
	# Uniform prior for theta
	# between 0 and 1
	theta ~ dunif(0, 1)

	# Likelihood
	x ~ dbin(theta, N)
}
```

Running JAGS

- We will run JAGS **from R** using the R package “rjags”.

- On Canvas there is an R script `use_jags.R` which you can use as a template.

Data Format

- The entire JAGS model code goes in a single R string.
- The data needs to be put into an **R list**.
- `use_jags.R` puts the output (parameter samples) into an R list called `results`.

What is an **R** list?

- In case anyone here has programmed in another language before:

- An **R** list is like a C/C++ **struct**
- An **R** list is like a Python **dictionary**

What is an **R** list?

- An **R** list is similar to a **vector** except the elements are **named** instead of numbered and can have different types/modes.

An Example of an **R** List

```
> me = list(name='Brendon', age=30 +
  10*runif(1))

> me

$name

[1] "Brendon"


$age

[1] 37.00963
```

Accessing List Elements

- In **R**, list elements can be accessed using the `$` operator.

```
> me$name
[1] "Brendon"
```

- In most other languages, it's . instead of $

e.g. `me.name` in C structs, C++ structs/classes, or Python classes

`me["name"]` for Python dictionaries

Uses for **R** Lists

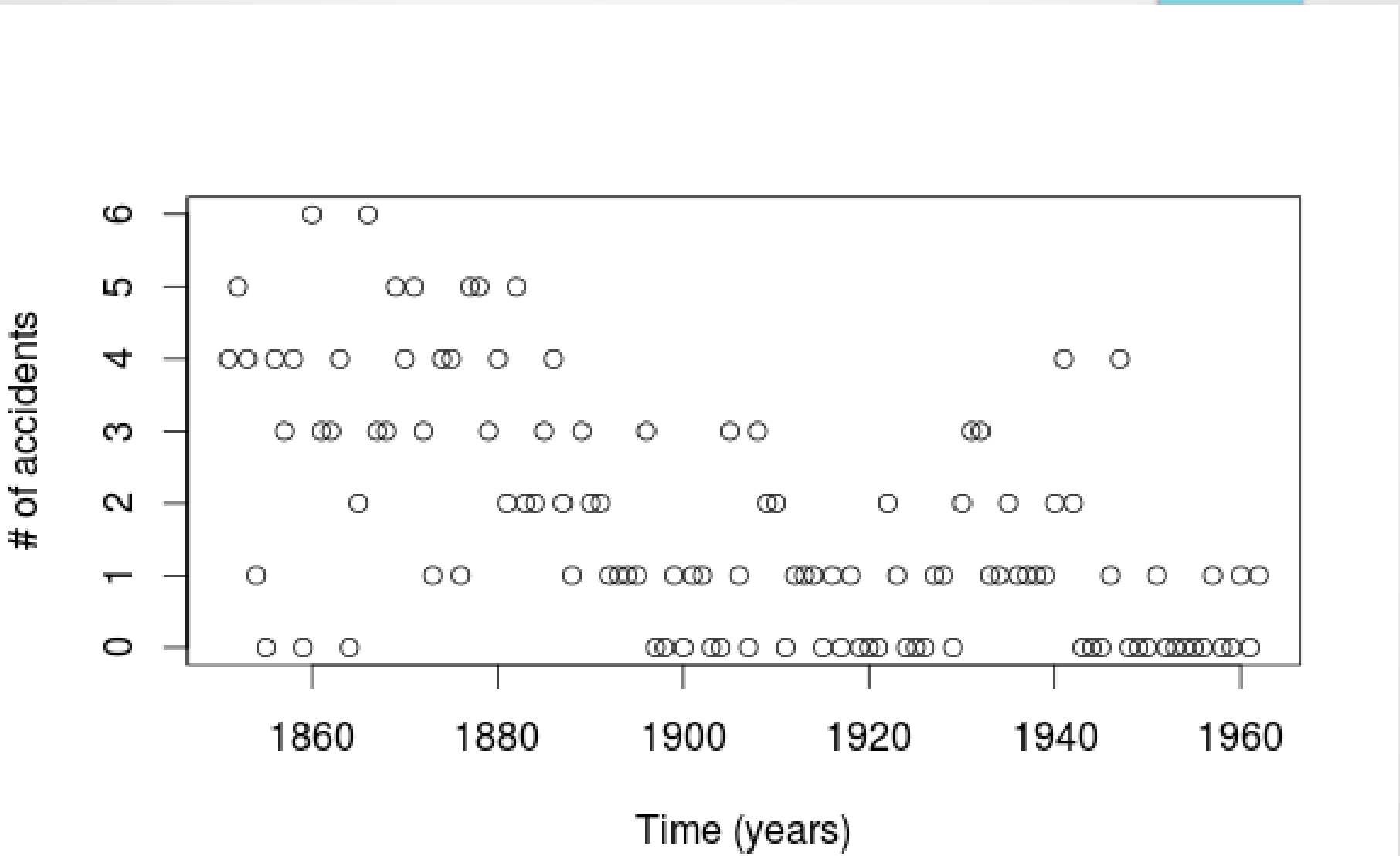
- R Lists are useful for organising data/variables that belong together into a single “object”
- Also, for being able to return **more than one result** from a function!
- Just pack it all into a list before the `return` statement.

Binomial Data results

- We can plot the trace plot and make summaries easily

```
plot(results$theta, type="l")
hist(results$theta, breaks=100)
mean(results$theta)
sd(results$theta)
```

British Coal Mining Accidents



Coal Mining Accidents

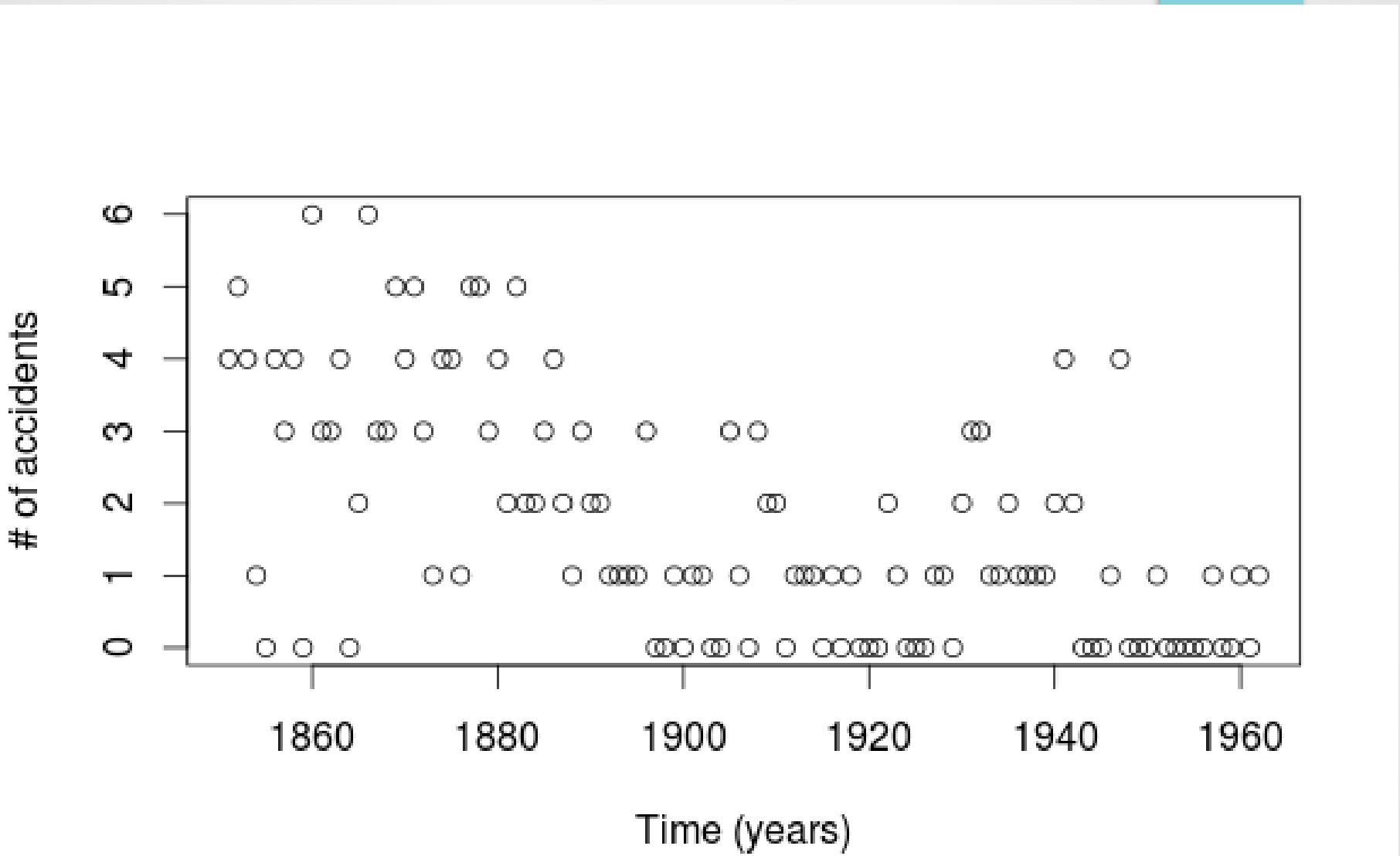
- Counts of rare events → Poisson model

- Does the rate change?
- Like crime example from earlier lab

Nuisance Parameters

- Not interested!
- But would need to know their values in order to predict data
- Need to infer them as well and marginalise out – happens automagically in MCMC

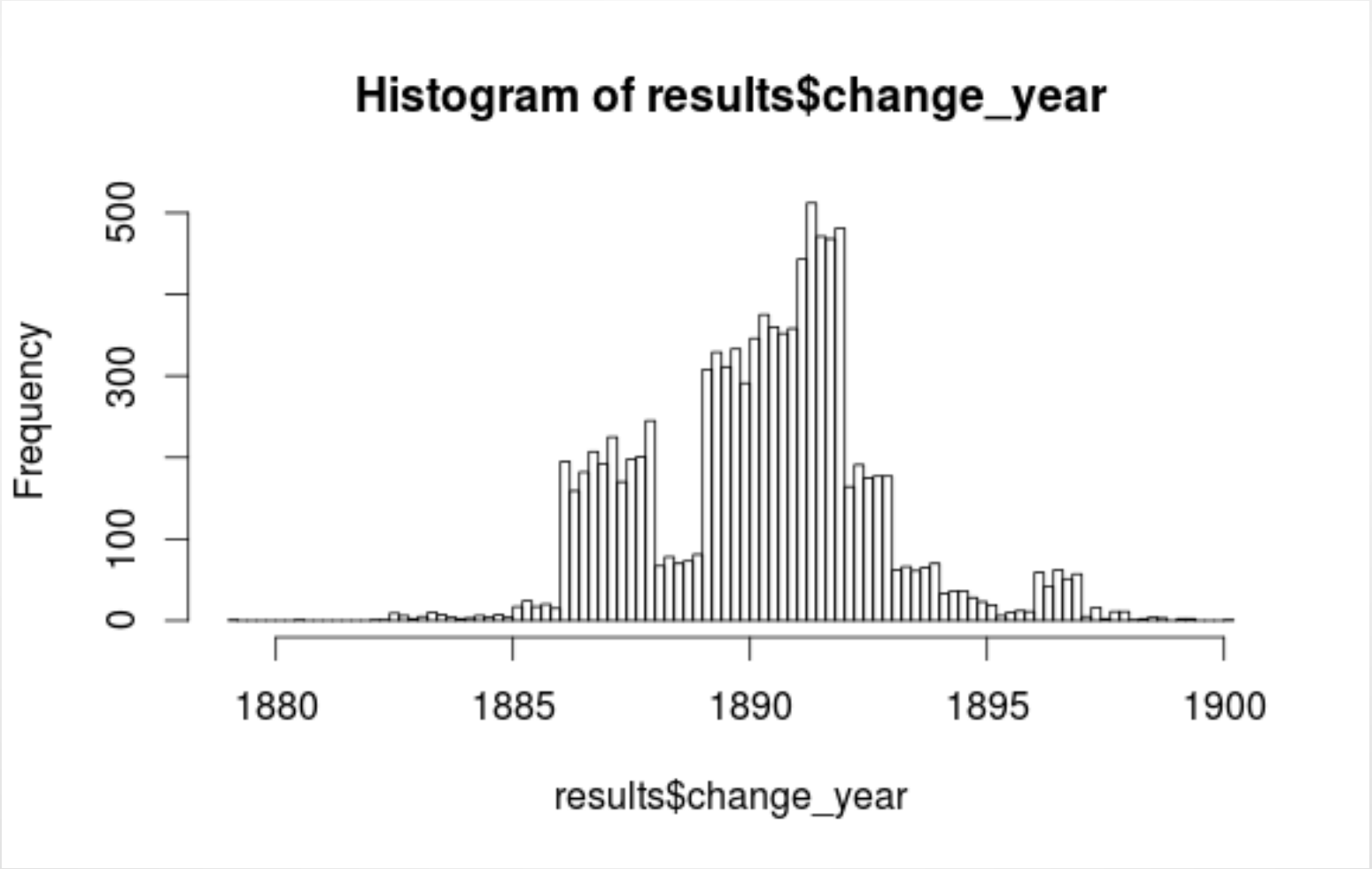
British Coal Mining Accidents



Let's make a change-point model in JAGS

Results

- Main result: marginal posterior for change year



Those “Nuisance Parameters”

- `lambda1` and `lambda2` are still available if we decide they're interesting