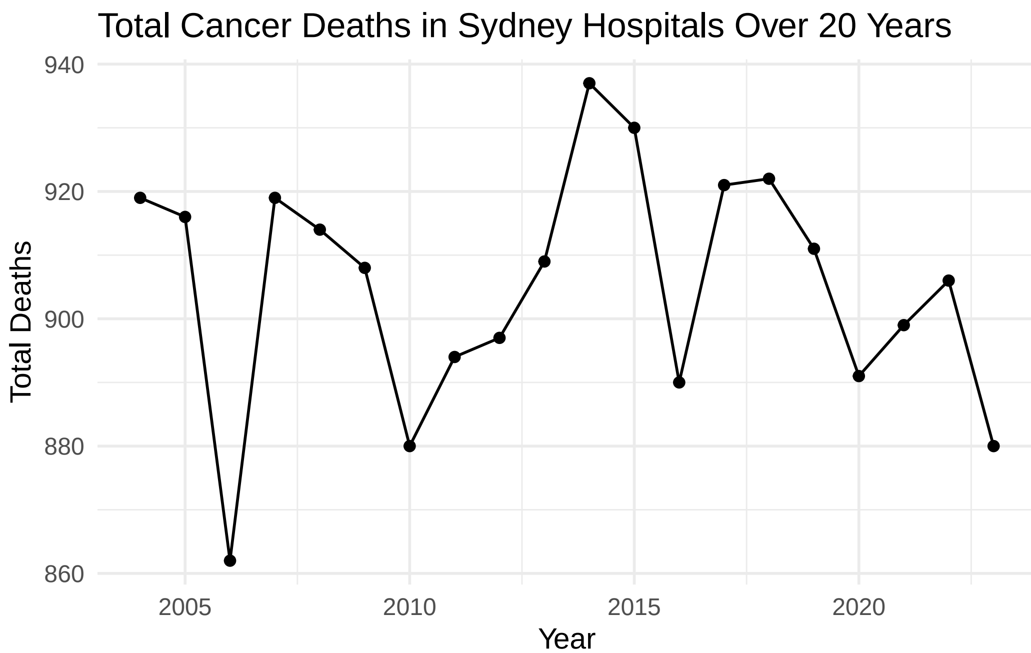
Quiz 12a (Simulate + Explore)

Bernice Bao

Simulate (Question 2)

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  138.0   167.8   181.0   181.1   194.2   225.0
```

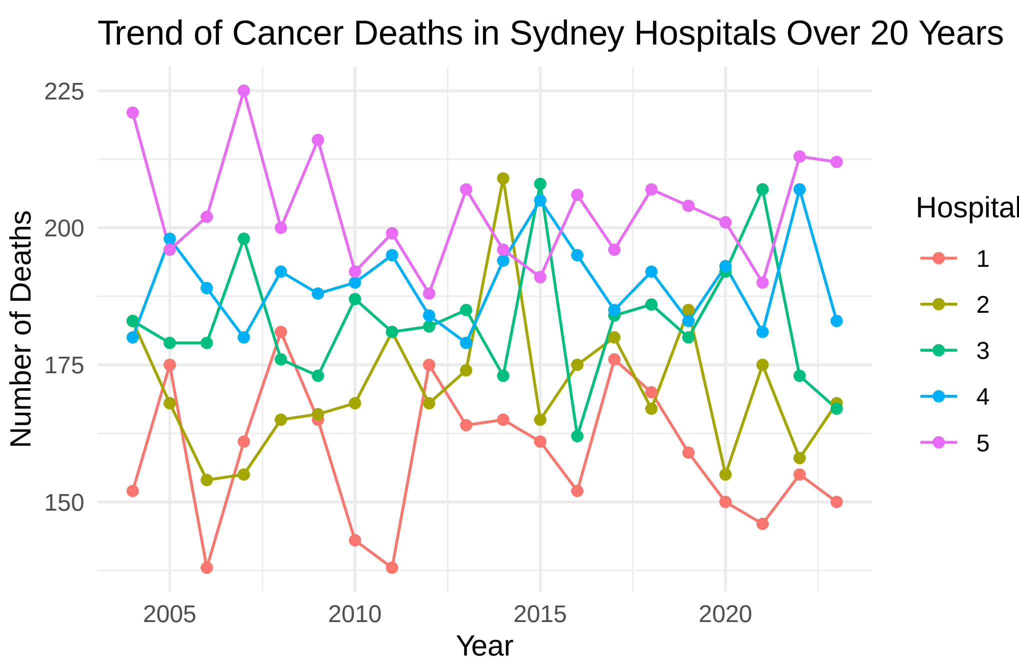



```
  Hospital Deaths
1        1 158.80
2        2 170.95
3        3 182.75
4        4 189.65
5        5 203.10
```

```
  Hospital Deaths
1        1    181
2        2    209
3        3    208
4        4    207
5        5    225
```

```
[1] 2007
```

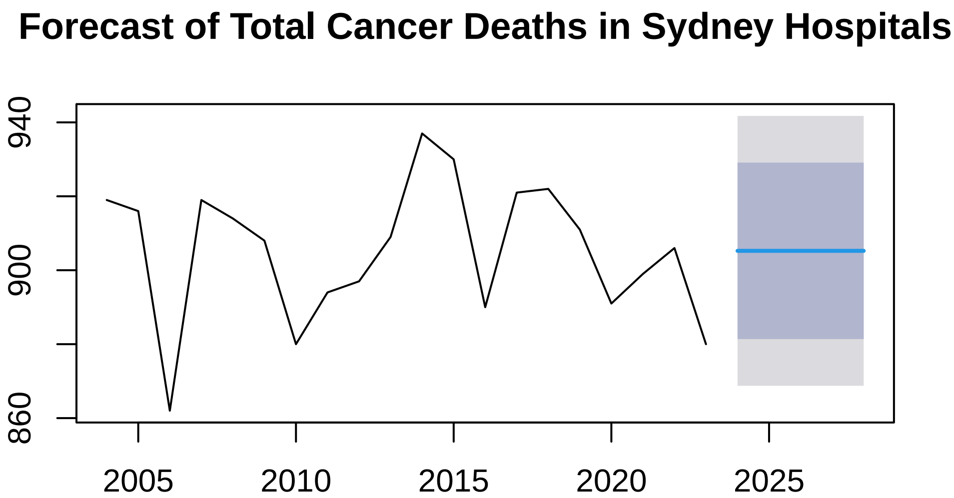
```
[1] 5
```




```
             [,1]        [,2]        [,3]         [,4]        [,5]
[1,]  1.000000000  0.09771522 -0.07541234  0.007501716 -0.08631512
[2,]  0.097715224  1.00000000 -0.21644782 -0.146893103 -0.20572856
[3,] -0.075412343 -0.21644782  1.00000000 -0.108566836 -0.27000566
[4,]  0.007501716 -0.14689310 -0.10856684  1.000000000 -0.25289689
[5,] -0.086315124 -0.20572856 -0.27000566 -0.252896893  1.00000000
```

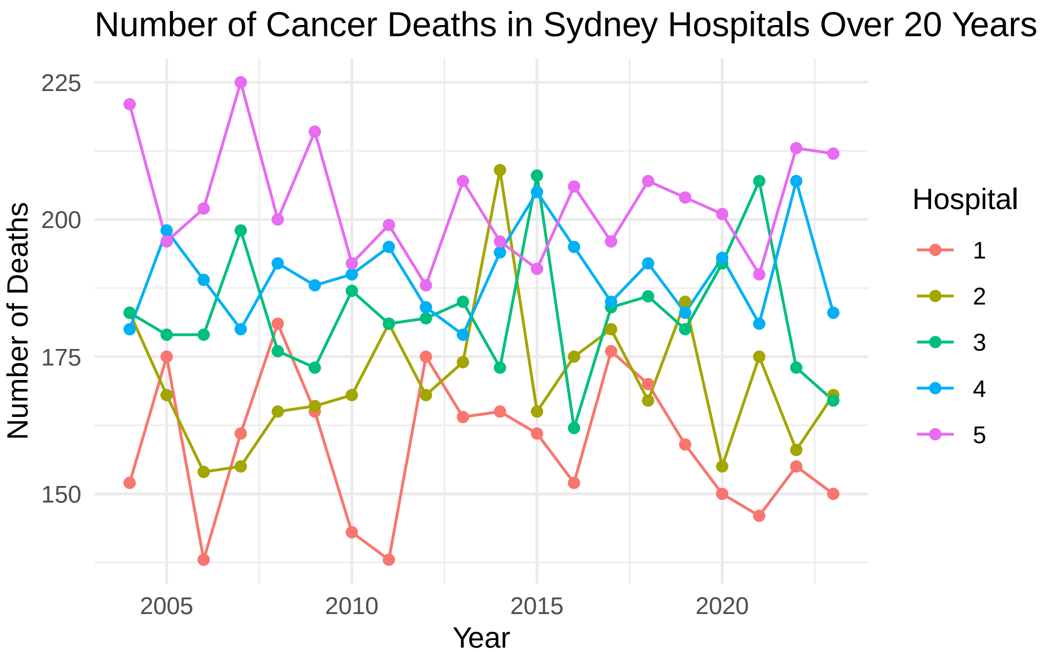
```
                 Df Sum Sq Mean Sq F value Pr(>F)
factor(Hospital)  4  23202    5801   45.52 <2e-16 ***
Residuals        95  12106     127
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```



Explore (Question 4)

```r
# Plot the simulated data
ggplot(df, aes(x = Year, y = Deaths, color = factor(Hospital))) +
  geom_line() +
  geom_point() +
  labs(title = "Number of Cancer Deaths in Sydney Hospitals Over 20 Years",
       x = "Year",
       y = "Number of Deaths",
       color = "Hospital") +
  theme_minimal()
```




```r
# Load necessary libraries
library(rstanarm)
```

```
Loading required package: Rcpp

This is rstanarm version 2.32.1

- See https://mc-stan.org/rstanarm/articles/priors for changes to default priors!

- Default priors may change, so it's safest to specify priors, even if equivalent to the defa

- For execution on a local, multicore CPU with excess RAM we recommend calling

  options(mc.cores = parallel::detectCores())
```

```r
# Fit a simple linear regression model
model <- stan_glm(Deaths ~ Year + factor(Hospital), data = df, family = "poisson")
```

```
SAMPLING FOR MODEL 'count' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 4.4e-05 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.44 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 1:
Chain 1:  Elapsed Time: 0.065 seconds (Warm-up)
Chain 1:                0.064 seconds (Sampling)
Chain 1:                0.129 seconds (Total)
Chain 1:

SAMPLING FOR MODEL 'count' NOW (CHAIN 2).
Chain 2:
Chain 2: Gradient evaluation took 1.1e-05 seconds
Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.11 seconds.
Chain 2: Adjust your expectations accordingly!
Chain 2:
Chain 2:
Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
```

```
Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 2:
Chain 2:  Elapsed Time: 0.067 seconds (Warm-up)
Chain 2:                0.062 seconds (Sampling)
Chain 2:                0.129 seconds (Total)
Chain 2:

SAMPLING FOR MODEL 'count' NOW (CHAIN 3).
Chain 3:
Chain 3: Gradient evaluation took 1e-05 seconds
Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.1 seconds.
Chain 3: Adjust your expectations accordingly!
Chain 3:
Chain 3:
Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 3:
Chain 3:  Elapsed Time: 0.064 seconds (Warm-up)
Chain 3:                0.067 seconds (Sampling)
Chain 3:                0.131 seconds (Total)
Chain 3:

SAMPLING FOR MODEL 'count' NOW (CHAIN 4).
Chain 4:
Chain 4: Gradient evaluation took 1e-05 seconds
Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.1 seconds.
Chain 4: Adjust your expectations accordingly!
Chain 4:
Chain 4:
Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
```

```
Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 4:
Chain 4:  Elapsed Time: 0.063 seconds (Warm-up)
Chain 4:                0.064 seconds (Sampling)
Chain 4:                0.127 seconds (Total)
Chain 4:
```

```
# Print the summary of the model
summary(model)
```

```
Model Info:
 function:     stan_glm
 family:       poisson [log]
 formula:      Deaths ~ Year + factor(Hospital)
 algorithm:    sampling
 sample:       4000 (posterior sample size)
 priors:       see help('prior_summary')
 observations: 100
 predictors:   6

Estimates:
                    mean   sd   10%   50%   90%
(Intercept)        5.4    2.5  2.2   5.4   8.7
Year               0.0    0.0  0.0   0.0   0.0
factor(Hospital)2 0.1    0.0  0.0   0.1   0.1
factor(Hospital)3 0.1    0.0  0.1   0.1   0.2
factor(Hospital)4 0.2    0.0  0.1   0.2   0.2
factor(Hospital)5 0.2    0.0  0.2   0.2   0.3

Fit Diagnostics:
           mean   sd    10%   50%   90%
mean_PPD 181.0    1.9 178.6 181.0 183.5
```

```
The mean_ppd is the sample average posterior predictive distribution of the outcome variable

MCMC diagnostics
                  mcse Rhat n_eff
(Intercept)       0.0  1.0  4836
Year              0.0  1.0  4841
factor(Hospital)2 0.0  1.0  1683
factor(Hospital)3 0.0  1.0  1810
factor(Hospital)4 0.0  1.0  1984
factor(Hospital)5 0.0  1.0  1839
mean_PPD          0.0  1.0  3078
log-posterior     0.0  1.0  1841

For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective
```