BIO782P Assessment 2

Joe Parker

11/11/2018

BIO782P Statistics and Bioinformatics Assignment 2018 (2 of 2)

Due: 17:00 Friday 14 December 2018

Part A

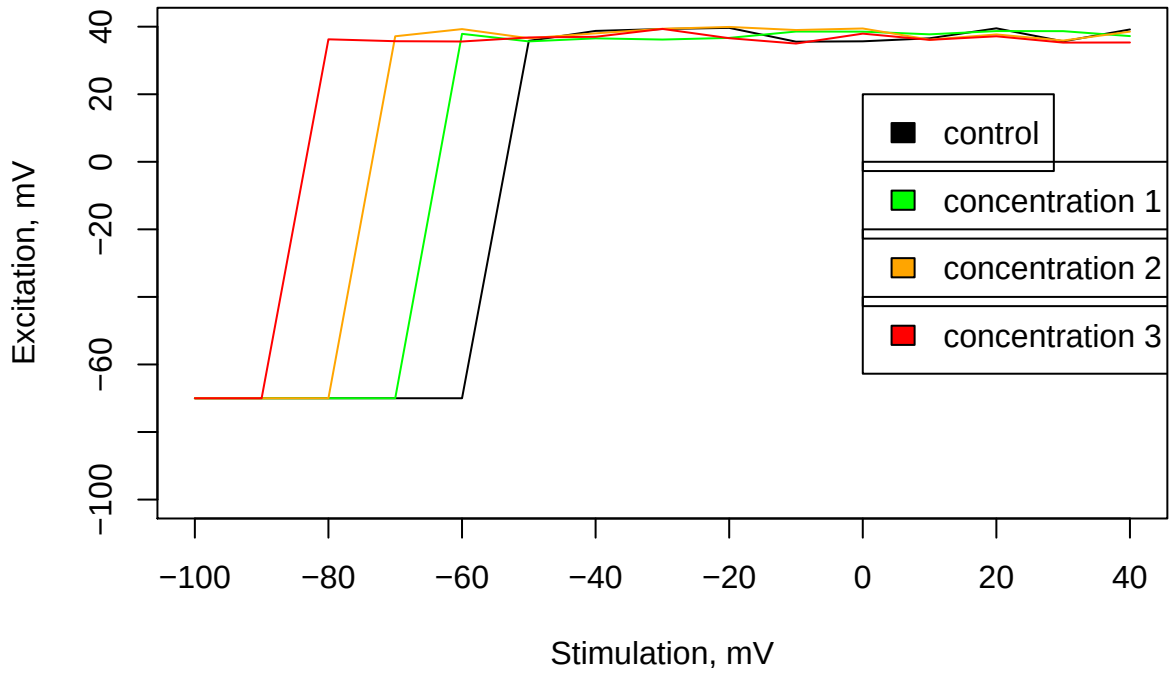
To determine the effect of increasing nicotine concentration on the nervous system, action potentials (in mV) were measured in a series of giant squid axons infused with nicotine at various concentrations. For each series, the axon was stimulated with increasing voltages, and the peak response voltage within the following 3ms recorded. The experiment was repeated with differing concentrations of nicotine.

The input data look like this:

```
stimulation_potential = seq(-100, 40, by=10)
response_voltage_control = c(-70.00000,-70.00000,-70.00000,-70.00000,-70.00000,35.79871,38.75082,39.304
response_voltage_concN_1 = c(-70.00000,-70.00000,-70.00000,-70.00000,37.88144,35.64345,36.54319,36.1877
response_voltage_concN_2 = c(-70.00000,-70.00000,-70.00000,37.16110,39.28471,36.57506,38.03037,39.37853
response_voltage_concN_3 = c(-70.00000,-70.00000,36.25729,35.70035,35.61478,36.81245,37.01299,39.34139,

plot(stimulation_potential,stimulation_potential,type='n',xlab='Stimulation, mV',ylab='Excitation, mV')
lines(stimulation_potential,response_voltage_control)
lines(stimulation_potential,response_voltage_concN_1,col="green")
lines(stimulation_potential,response_voltage_concN_2,col="orange")
lines(stimulation_potential,response_voltage_concN_3,col="red")

legend(0,20,"control",fill="black")
legend(0,00,"concentration 1",fill="green")
legend(0,-20,"concentration 2",fill="orange")
legend(0,-40,"concentration 3",fill="red")
```



A colleague has written some R code to analyse this behaviour by fitting a model of excitation to the data. Unfortunately she has left the lab for another study, and her code is not very well documented.

```
predict = function(activation_threshold,stimulation_voltage){
  activation = rep(-70,length(stimulation_voltage))
  activation[stimulation_voltage>activation_threshold] = 40
  return(activation)
}

calculate_errors = function(predicted, observed){
  total_errors = sum((observed - predicted)^2)
  return(total_errors)
}

fit_threshold = function(input_values_stimulation,input_values_response,threshold){
  predicted_values = predict(threshold,input_values_stimulation)
  fit_errors = calculate_errors(predicted_values,input_values_response)
  return(fit_errors)
}

fitted_threshold = runif(1,-100,40)
error = fit_threshold(stimulation_potential,response_voltage_control,fitted_threshold)

for(i in 1:20){
  new_threshold = runif(1,-100,40)
  new_error     = fit_threshold(stimulation_potential,response_voltage_control,new_threshold)
  if(new_error < error){
    fitted_threshold = new_threshold
    error = new_error
  }
}

fitted_threshold
```

```
error

write.csv(data.frame('fitted value'=fitted_threshold,'errors SS'=error),file = 'excitation.csv',row.nam
```

With reference to the code above, answer the following questions (you may wish to answer separately, in comments to the code, or both):

1) Describe what the inputs, outputs and role for each of the functions `predict`, `calculate_errors` and `fit_threshold` are.
2) How is the data modelled?
3) How many degrees of freedom are present in (a) the model, and (b) the residuals?
4) Describe what happens when the code is executed, in statistical terms.
5) How is the model fit to the data optimised?
6) How could you improve the optimisation?
7) How would you modify the code to:
    a) fit the data given under the three experimental nicotine treatments and
    b) compare the goodness of fit amongst these fitted models to determine whether each nicotine treatment lowers the activation threshold, compared to the control treatment
    c) what is the null hypothesis in this case?

Part B

We have modified your code so that it prints each fit out to a logfile as the optimisation progresses. The logfiles are called `fit_01`, `fit_02`... etc and print out each time optimisation loop runs. The final `fitted_threshold` and `errors` will be written to another .csv, called `final.csv`. We have also modified the code to pass a user-specified random seed to the script for setting the initial value for `fitted_threshold`. Pseudocode for a Docker container to run that code is given:

```
# Docker R base image
FROM r-base

COPY . /usr/local/src/myscripts
WORKDIR /usr/local/src/myscripts

CMD ["Rscript", "activation-thresholds.R"]
```

1) How would we use this code to run our analysis on a grid? Outline what steps we should take.

```
#$ -cwd
#$ -S /bin/bash
#$ -j y
#$ -pe smp 1
#$ -l h_vmem=2G
module load singularity
singularity run activationContainer.img input_data.csv
```

2) How would you rewrite this jobfile to achieve the following:
    - Move the final outputs to a directory (`../output`)?
    - Change random seed (set with the `-p [some integer]` argument)?
    - Run as an array job, with 42 replicates?