

University of Bath

**DEPARTMENT OF MATHEMATICAL SCIENCES
EXAMINATION**

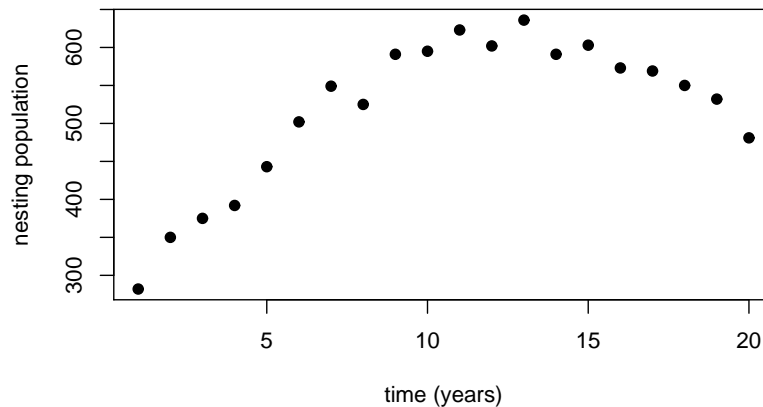
MA40198 Statistics for biological dynamic modelling

2011

Candidates may use university-supplied calculators.

Full marks will be given for correct answers to **THREE** questions.
Only the best three answers will contribute towards the assessment.

1. An annual count is made of the nesting population of breeding puffins on a small offshore island. 20 years of data are shown here:



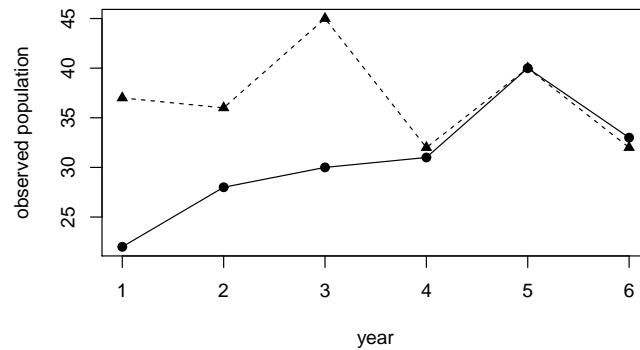
There is some concern about the apparent decline in the nesting population towards the end of the data. In particular there is concern that the puffins are themselves causing soil erosion on the island, which is reducing the number of available nesting burrows over time. The following model is proposed for the size of the breeding population, n , over time.

$$\begin{aligned}\frac{dn}{dt} &= rn\{1 - (n/k)^\alpha\} \\ \frac{dk}{dt} &= -\beta nk\end{aligned}$$

r , α and β are parameters to be estimated, as are initial states n_0 and k_0 . k represents a measure of available nesting space.

- (a) Obtain a discrete time approximation to the model, suitable for solving with a one year time step. Make sure that the model is suitable for use in automatic model fitting: e.g. that it can not produce negative populations. [4]
- (b) Assume that the observed population in year t is a Poisson deviate with mean given by the model population in that year, n_t . Write R code to estimate the 5 model parameters from the data shown (which you can assume are provided in a vector, y). Minor programming errors will not lose marks, provided that the statistical meaning is clear and correct. [6]
- (c) Give the R code for producing 3 model checking plots, and explain what you would be looking for in each. [6]
- (d) Provide R code for computing an approximate 95% confidence interval for β , and give the statistical result on which this is based. [4]

2. Two populations of killer whales have been monitored over a 7 year period after the establishment of a high energy sonar installation at a naval base in the territory of one of the groups. There is concern that the sonar disrupts the social behaviour of the animals, leading to lowered breeding success and higher adult mortality. In the first year of monitoring the populations were studied intensively so that the exact numbers of adults, sub adults and yearlings (young less than a year old) were known in that year. Thereafter, to minimise disturbance, the populations were monitored using short surveys once a year, which estimated the total number of sub adults plus adults only. The data are shown in the following figure. The data from the population near the sonar installation are shown as triangles, joined by dashed lines.



It is believed that each of the two populations can be quite well modelled using a matrix population model of the form

$$\begin{bmatrix} n_0 \\ n_s \\ n_a \end{bmatrix}_{t+1} = \begin{bmatrix} s_0 & 0 & \beta \\ g_0 & s_1 & 0 \\ 0 & g_1 & s_2 \end{bmatrix} \begin{bmatrix} n_0 \\ n_s \\ n_a \end{bmatrix}_t$$

where n_0 , n_s and n_a model the number of yearlings, sub adults and adults, respectively. The quantities in the square matrix are parameters to be estimated: all are positive, and in addition $s_0 + g_0 < 1$, $s_1 + g_1 < 1$ and $s_2 < 1$. The code and output at the end of this question performs an analysis of the killer whale survey data using this model. Examine the code and output and answer the following questions.

- (a) What is the purpose of the R function `whale`? Explain the purpose of the reparameterization scheme used in the function. [4]
- (b) What is the purpose of the R function `whale.ll` and what distributional assumption is it based on? [3]
- (c) What is the purpose of the R function `whale.ll0`? [3]
- (d) Explain the overall aim of the analysis being performed, taking care to relate this purpose to the biological context. What would you conclude? [3]

Question 2 continues on next page ...

Question 2 continued ...

- (e) If it was necessary to come up with more conclusive results within the next year, what further fieldwork would you recommend? [3]
- (f) Explain briefly how you might go about estimating intervals for the model parameters in a way that incorporated prior knowledge about their probable values from studies on other populations? [4]

```
whale <- function(theta) {

  theta <- exp(theta)

  S0 <- theta[1]/(1+theta[1]+theta[2])
  G0 <- theta[2]/(1+theta[1]+theta[2])
  S1 <- theta[3]/(1+theta[3]+theta[4])
  G1 <- theta[4]/(1+theta[3]+theta[4])
  S2 <- theta[5]/(1+theta[5])
  B <- theta[6]
  ## matrix for population 1
  M1 <- matrix(0,3,3)
  diag(M1) <- c(S0,S1,S2)
  M1[2,1] <- G0
  M1[3,2] <- G1;
  M1[1,3] <- B ## birth rate
  ## matrix for population 2
  M2 <- M1
  M2[3,3] <- theta[7]/(1+theta[7])
  M2[1,3] <- theta[8] ## sonar population birth rate

  ## known initial states...
  N1 <- c(0,9,22)
  N2 <- c(3,18,25)

  n <- matrix(0,2,6)
  for (i in 1:6) {
    N1 <- M1%*%N1
    N2 <- M2%*%N2
    n[1,i] <- N1[2]+N1[3]
    n[2,i] <- N2[2]+N2[3]
  }
  n ## return observed immature + adults
}
```

Question 2 continues on next page ...

Question 2 continued ...

```
whale.ll <- function(theta,y) {
  n <- whale(theta)
  -sum(dpois(y,n,log=TRUE))
}

whale.ll0 <- function(theta,y) {
  ## log likelihood with same parameters in both groups
  ## so input vector is size 6
  theta <- c(theta,theta[5],theta[6])
  n <- whale(theta)
  -sum(dpois(y,n,log=TRUE))
}

y ## look at the data
      [,1] [,2] [,3] [,4] [,5] [,6]
[1,]   22  28  30  31  40  33
[2,]   37  36  45  32  40  32

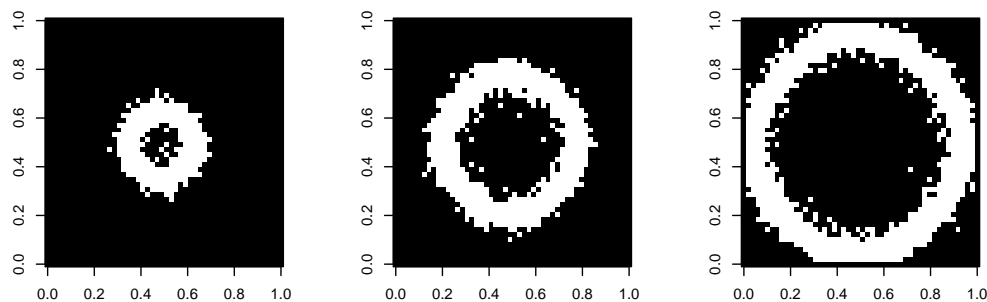
theta <- c(-1,1,1,0,3,-1.5,2.5,-2)

fit <- optim(theta,whale.ll,method="BFGS",hessian=TRUE,y=y)

fit0 <- optim(theta[1:6],whale.ll0,method="BFGS",hessian=TRUE,y=y)

llr <- 2*(fit0$value-fit$value)
pchisq(llr,df=2,lower.tail=FALSE)
[1] 0.5319283
```

3. A laboratory experiment examining the spread of a fungus was conducted on a 50 by 50 grid, where each grid cell contained the same quantity of growing medium (i.e. food for the fungus). The distribution of the fungus was measured on several occasions after the start of the experiment, using an instrument giving a reading of 1 if fungus was detected and 0 otherwise. The instrument was not perfect, but the probability of detecting the fungus increases with fungal density. The experiment resulted in the following data after 100, 200 and 300 hours. Black indicates 0 and white 1.



The fungal spread was modelled using a simple ‘cellular automata’ model, in which fungus in a cell can infect neighbouring cells, survives according to the amount of food available, and depletes the food at some rate. The R code and output at the end of this question is an attempt to fit such a model to the data in the 3 images shown. Carefully examine the code and output and answer the following questions.

- (a) R function `fungus` implements a simple model for the dynamics of the fungal density over the grid. Let $d_{i,j}(k)$ denote the density of fungus in the cell in row i column j , at the k^{th} timestep. Let $f_{i,j}(k)$ be the corresponding food. Write out the model equations for $d_{i,j}(k+1)$ and $f_{i,j}(k+1)$ in terms of the fungal density and food in the grid at step k . [5]
- (b) State precisely what distributional assumption is being made about the observed presence/absence of fungus given the model density of fungus. [3]
- (c) Explain the purpose of the R function `fung.11` and the call to R function `optim`. [4]
- (d) Theory suggests that the parameter θ_1 (`theta[1]`) should be 0.05. Write code to test this theory using a generalized likelihood ratio test. Minor programming errors will not be penalized, provided that the statistical meaning is clear and correct. [5]
- (e) What is the matrix `C` and what is it telling you about the model in this case. [3]

Question 3 continues on next page ...

Question 3 continued ...

```

fungus <- function(theta=c(0.02,0.01,0.1),t=c(100,200,300)) {
## fungal spread model...
  m <- 50 # arena size
  d <- matrix(0,m,m) ## fungus array
  x <- matrix(1:m,m,m,byrow=TRUE) ## array of x locations
  y <- matrix(1:m,m,m) ## array of y locations

  ## initialize fungus and food
  ind <- (x-m/2)^2 + (y-m/2)^2 < 10
  d[ind] <- 1
  f <- matrix(1,m,m) ## food

  dl <- list();k <- 1 ## list for output at times in t
  for (i in 1:max(t)) {
    ## send spores to 4 nearest neighbours...
    ind <- 2:(m-1)
    d[ind,ind] <- d[ind,ind] +
      (d[ind+1,ind] + d[ind-1,ind] + d[ind,ind+1] + d[ind,ind-1])*theta[1]

    ## resource dynamics, deplete resource
    f <- f * exp(-d*theta[2]) ## fungus depletes food
    d <- d * f^theta[3] ## fungus survival depends on food

    if (i %in% t) { dl[[k]] <- d; k <- k + 1 } ## output
  }
  dl
}

fung.ll <- function(theta,y,t) {
## fungal spread model, -ve log likelihood, y is a list
## of binary images taken at times in t.
  theta <- exp(theta)
  d <- fungus(theta,t)
  eps <- 1e-10
  ll <- 0
  for (i in 1:length(t)) {
    D <- d[[i]]
    D[D > 1-eps] <- 1 - eps
    D[D < eps] <- eps
    ll <- ll + sum(dbinom(y[[i]],1,D,log=TRUE))
  }
  -ll
}

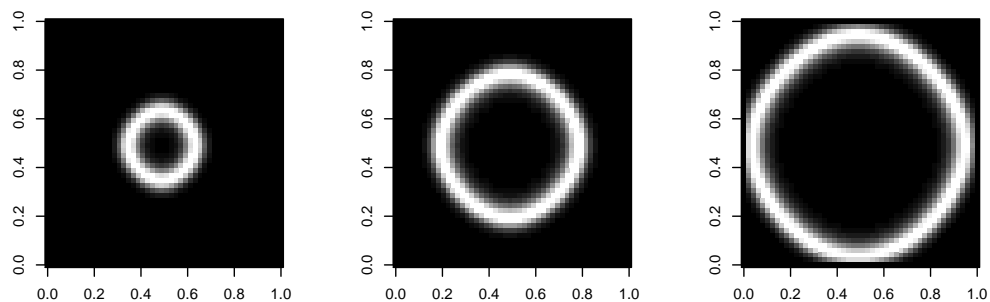
```

Question 3 continues on next page ...

Question 3 continued ...

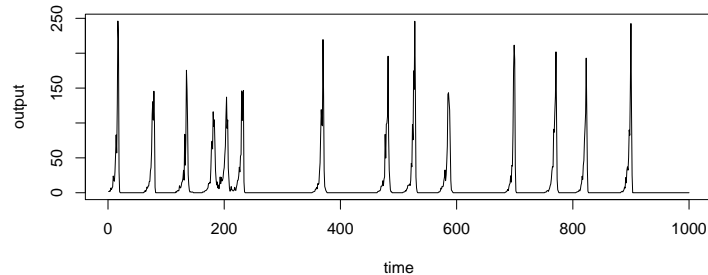
```
## plot raw data, stored as a list of matrices containing binary images
par(mfrow=c(1,3))
for (i in 1:3) image(y[[i]],col=gray(c(0,1)))
## ... figure at start of question.

ltheta0 <- log(c(0.05,0.05,0.2)) ## initial log params
t <- c(100,200,300)
fit <- optim(ltheta0,func.ll,method="BFGS",hessian=TRUE,y=y,t=t)
fv <- fungus(exp(fit$par),t)
## produce greyscale plot of fungus density...
for (i in 1:3) {
  image(fv[[i]],col=gray(c(seq(0,1,length=30))),
    zlim=c(0,2.5)) ## 0 set to black, >= 2.5 to white, greys in between
}
```



```
> V <- solve(fit$hessian)
> S <- diag(diag(V)^-.5)
> C <- S%*%V%*%S
> C
      [,1]      [,2]      [,3]
[1,] 1.000000e+00 -7.586009e-07 -1.863865e-06
[2,] -7.585893e-07 1.000000e+00 -1.000000e+00
[3,] -1.863864e-06 -1.000000e+00 1.000000e+00
```

4. The following figure shows activity of a single grasshopper neuron following an experimental stimulus.



It is believed that this activity can be modelled via a feedback and damping model. If an external stimulus above a known ‘action threshold’ excites the neuron, then there is exponential growth in the neuron output, V , until some further threshold, γ is exceeded. Once γ is exceeded, inhibitor, A , is produced which damps neuron output. The inhibitor decays away with time. If output drops below the original ‘action threshold’ output is damped further to essentially zero. The arrival of external stimuli above the threshold can be modelled as constant rate Poisson process. i.e. there is some small fixed probability of an arrival in any time interval, with the probability proportional to the interval’s length (and fixed constant of proportionality).

The R code and output at the end of this question estimates such a feedback and damping model from the observed data.

- Write out the neuron firing model, coded in the R function `neuron`, as mathematical equations with distributional assumptions. [5]
- Why might the model be difficult to fit by conventional maximum likelihood estimation or MCMC? [3]
- Carefully explain the purpose of the routine `neuron.11`, and the statistical idea underlying it. [3]
- Explain the purpose of the loop labelled `## MCMC loop`. [3]
- Comment on the plots produced at the end. [3]
- Write R code to produce 95% confidence/credible intervals for the parameters `gamma` and `r`. Minor R errors will not lose marks, provided that the meaning and logic are clear. [3]

Question 4 continues on next page ...

Question 4 continued ...

```

neuron <- function(theta,n=1000,n.rep=200) {
## neuron firing model...
  gamma <- theta[1]  ## inhibitor trigger
  lambda <- theta[2] ## external stimulus Poi rate
  beta <- theta[3]   ## inhibitor constant
  alpha <- theta[4]  ## inhibitor decay rate
  r <- theta[5]     ## firing build up rate
  sigma <- theta[6] ## firing noise
  sigma.obs <- 0.3  ## instrument noise (known in this case)
  ep <- matrix(rnorm(n*n.rep)*sigma,n,n.rep) ## firing noise
  eo <- matrix(rnorm(n*n.rep)*sigma.obs,n,n.rep) ## instrument noise
  v <- matrix(0,n,n.rep) ## replicate matrix
  V <- rep(1,n.rep) ## start all with stimulus to match data
  VT <- A <- V*0
  for (i in 1:n) {
    ind <- V > gamma ## index replicates exceeding gamma
    VT[ind] <- V-gamma ## amount, if any, by which threshold exceeded
    VT[!ind] <- 0
    V <- V * exp(r - beta*A +ep[i,]) ## neuron output
    V[V < 1] <- 0
    ind <- runif(n.rep) < lambda ## stimulus or not?
    V[ind] <- V[ind] + 1 ## stimulus
    A <- VT + A*exp(-alpha) ## inhibitor
    v[i,] <- V
  }
  y <- v*exp(eo) ## observation error
}

neuro.trans <- function(y,y.obs=NULL) {
## transform to stats...
  y <- as.matrix(y); n.rep <- ncol(y)
  S <- matrix(NA,14,n.rep)
  if (is.null(y.obs)) y.obs <- (1:nrow(y)-.5)/nrow(y)
  y.obs <- sort(y.obs)
  for (i in 1:n.rep) {
    S[1:7,i] <- as.numeric(acf(y[,i],plot=FALSE)$acf[2:8])
  }
  ## Function 'order.dist' regresses sorted simulated (or real) data on
  ## sorted real data, using a cubic polynomial,
  ## regression coefficients are extracted as statistics.
  S[8:11,] <- order.dist(y,y.obs,np=4,diff=0)
  ## Some more statistics...
  S[12,] <- colSums(abs(diff(sign(diff(y)))))) ## turning point count
  S[13,] <- colMeans(y); S[14,] <- colSums(y==0)
  S
}

```

Question 4 continued ...

```

neuro.ll <- function(theta,y) {
## synthetic likelihood...
  s <- as.numeric(neuro.trans(y,y)) ## observed stats
  #set.seed(0)
  Y <- neuron(theta,n=length(y),n.rep=200)
  S <- neuro.trans(Y,y) ## simulated stats

  mu <- rowMeans(S)
  S <- S - mu
  V <- S%*%t(S)/(ncol(S)-1)
  -t(s-mu)%*%solve(V,s-mu)/2 - as.numeric(determinant(V,logarithm=TRUE)$modulus)/2
}

## now an MCMC loop...
## The original neuron output data are in 'y'.
n.mc <- 25000
n.para <- length(theta)
th <- matrix(0,n.para,n.mc) ## storage for chain results
llr <- rep(NA,n.mc) ## storage for log synthetic likelihood
th[,1] <- log(c(50,0.005,0.01,0.1,0.15,0.05)) ## initial state
prop.sd <- c(0.05,0.05,0.05,0.05,0.05,0.05) ## proposal standard deviations

llr[1] <- neuro.ll(exp(th[,1]),y)
reject <- 0
uni <- runif(n.mc)

for (i in 2:n.mc) { ## MCMC loop
  th[,i] <- th[,i-1] + rnorm(n.para)*prop.sd
  llr[i] <- neuro.ll(exp(th[,i]),y)
  alpha <- min(1,exp(llr[i]-llr[i-1]))
  if (uni[i]>alpha) { ## reject
    th[,i] <- th[,i-1]
    llr[i] <- llr[i-1]
    reject <- reject + 1
  }
} ## end of the MCMC chain

1-reject/n.mc ## acceptance rate
[1] 0.25104

```

Question 4 continues on next page ...

Question 4 continued ...

```
yp <- neuron(exp(th[,n.mc]),n.rep=2)
par(mfrow=c(3,3),mar=c(4,4,1,1))
for (j in 1:nrow(th)) {
  plot(exp(th[j,]),type="l",xlab="i",ylab=pname[j])
}
plot(llr,xlab="i",ylab=expression(l[s](theta)),type="l")
plot(yp[,1],type="l",ylab="output",xlab="time")
plot(yp[,2],type="l",ylab="output",xlab="time")
```

