Exercise Solutions

Exercise: Binomial (Bernoulli) GLM - dolphin behavioural plasticity

1. The data for this exercise were collected by the Cromarty Lighthouse team between 2010 and 2016, using underwater sound recorders (CPOD) to continuously monitor the pattern of presence and foraging behaviour of bottlenose dolphins at key sites in the Moray Firth. Variables:

- `X` index of the observations
- `presence`: 0 for absence, 1 for presence
- `year`
- `julianday`: day of the year
- `tideangle_deg`: tidal state‘
- `mh`: hour of the day (integer)
- `mon`: month (integer)
- `Per2`: Bin year into two periods (May+June vs rest of year)
- `Per4`: 3 periods of 20 days from early May to end of June vs rest of the year
- `Time6`: Bin time of day into 6 4h periods (first centered on midnight)
- `Tide4`: Bin tide angle into 4 quadrants with peaks in middle of respective bin

It has been suggested that the patterns of use of coastal foraging sites by this dolphin population is quite variable over time. For example, **sightings at Sutors are thought to be more frequent around May-June, although the odds of detecting dolphins may depend on various factors, including tidal state and time of day**. The goal of this exercise is to describe variation in dolphin probability of presence in relation to factors like tidal state, time of day and season (particularly May/June vs. rest of the year), with possible interactions between them.

The data have been aggregated as presence/absence at a 1h resolution. You will focus on one of the sites, "Sutors", a subset which will leave you with just under 5000 presence/absence records to play with. Note that "absence" refers to the absence of a detection, not to the absence of dolphins. We can ignore this in the analysis, but we should keep it in mind when interpreting the results.

Background on the data and the study can be found here, courtesy of Paul Thompson. The exercise can be done entirely without consulting this. I recommend you watch this or any companion material (the referenced paper) outside the synchronous session, to make the most of the time you have with demonstrators to progress on the exercises.

As in previous exercises, either create a new R script (perhaps call it GLM_PresAbs) or continue with your previous R script in your RStudio Project. Again, make sure you include any metadata you feel is appropriate (title, description of task, date of creation etc) and don't forget to comment out your metadata with a `#` at the beginning of the line.

2. Import the data file 'dolphin.csv' into R (a "small" 5000 records-long subset of the original data set) by running the following chunk of code (please unfold and copy/paste). The code for converting the original publicly available data into the 'dolphin.csv' file is given as an appendix code chunk at the end of the practical, for your info. It includes converting the original numeric variables into categories that you can specify (binning). Binning is done easily using the `cut()` function (examples in the code chunk at the end, if you want to create your own categories).

```
dat<- read.csv("./data/dolphin.csv", stringsAsFactors= TRUE)

# re-ordering factor levels for convenience:
dat$Per2<- factor(dat$Per2, levels= c("RestOfYear", "MayJun"))
# (making "RestOfYear" the reference level)
dat$Per4<- factor(dat$Per4, levels= c("RestOfYear", "MayJun1", "MayJun2", "MayJun3"))
dat$Time6<- factor(dat$Time6, levels= c("MNight", "AM1", "AM2", "MDay", "PM1", "PM2"))
# reordering chronologically

str(dat)
```

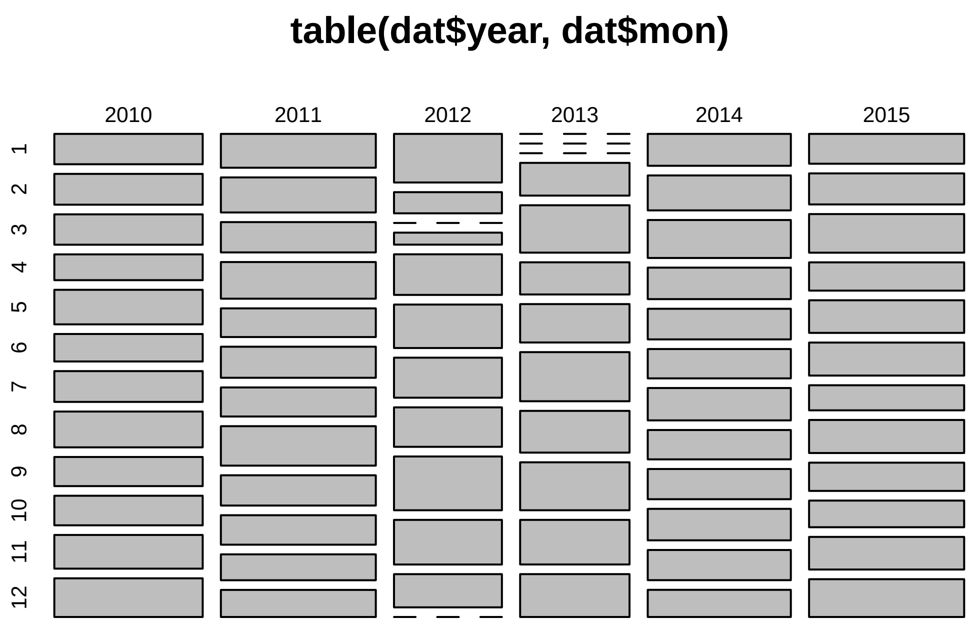
```
## 'data.frame':    5000 obs. of  11 variables:
##  $ X            : int  31458 14027 40551 40456 15894 13109 23797 6053 23445 34584 ...
##  $ presence     : int  0 1 0 0 1 0 0 0 0 0 ...
##  $ year         : int  2014 2011 2015 2015 2011 2011 2013 2010 2012 2014 ...
##  $ julianday    : int  59 226 80 76 312 188 102 256 327 192 ...
##  $ tideangle_deg: int  247 356 176 299 127 75 44 73 180 103 ...
##  $ mh           : int  8 13 7 8 3 7 3 6 14 15 ...
##  $ mon          : int  2 8 3 3 11 7 4 9 11 7 ...
##  $ Per2         : Factor w/ 2 levels "RestOfYear","MayJun": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Per4         : Factor w/ 4 levels "RestOfYear","MayJun1",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Time6        : Factor w/ 6 levels "MNight","AM1",..: 3 4 3 3 2 3 2 2 4 5 ...
##  $ Tide4        : int  4 1 3 4 2 2 1 2 3 2 ...
```

3. Take a look at the structure of this dataframe. Start with an initial data exploration to look at any imbalance between the predictors, and factors affecting presence of dolphins. Which ones are continuous or categorical? Which ones would your intuition guide you to use for data exploration? For modelling? Hints:

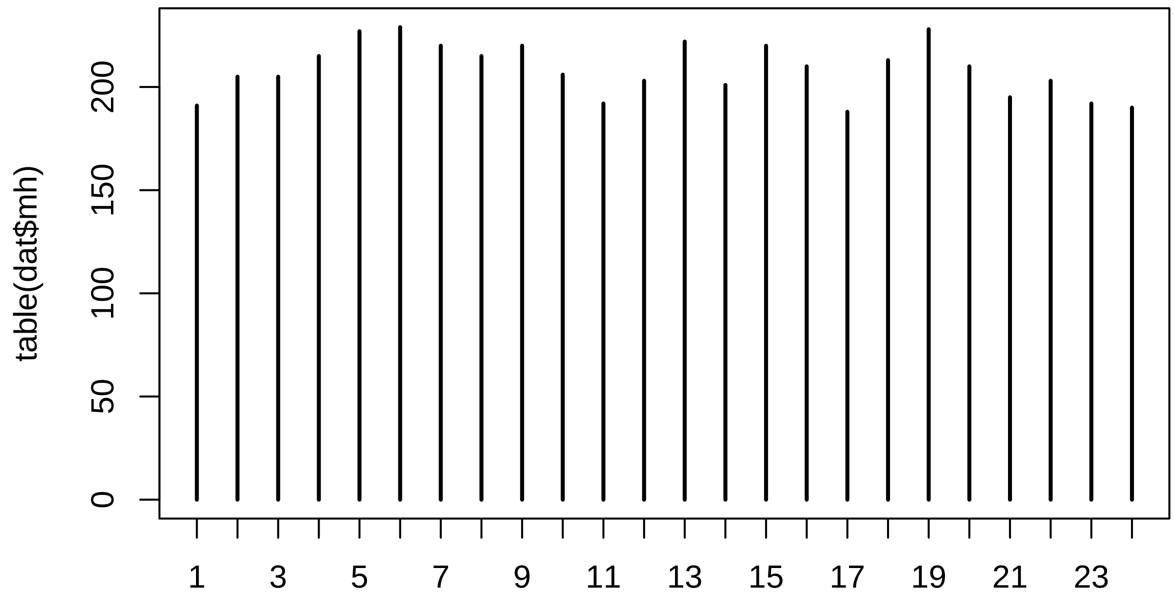
- Presence/absence data (Bernoulli) are more difficult than most to explore.
- One approach is to count observations per categories of interest.
- `table()` is a useful way to count the number of observations per category or combinations of categories, e.g. `ObsPerMonthYear<- table(dat$year, dat$mon)`

- `plot(ObsPerMonthYear)` returns a "mosaic plot" where the area of each rectangle is proportional to the count.
- For proportion of time present, you could calculate mean presence per category `bla<- tapply(dat$presence, list(dat$GroupOfInterest), mean)`
- and plot this using `plot(bla, type= "b", ylim= c(0, 1), xlab= "GroupOfInterest", ylab= "presence")`
- In more than one dimension, `matplot(tapply(dat$presence, list(dat$Group1, dat$Group2), mean), type= "l", ylim= c(0, 1), xlab= "Group1", ylab= "presence", lty= 1)` produces one line per Group2.

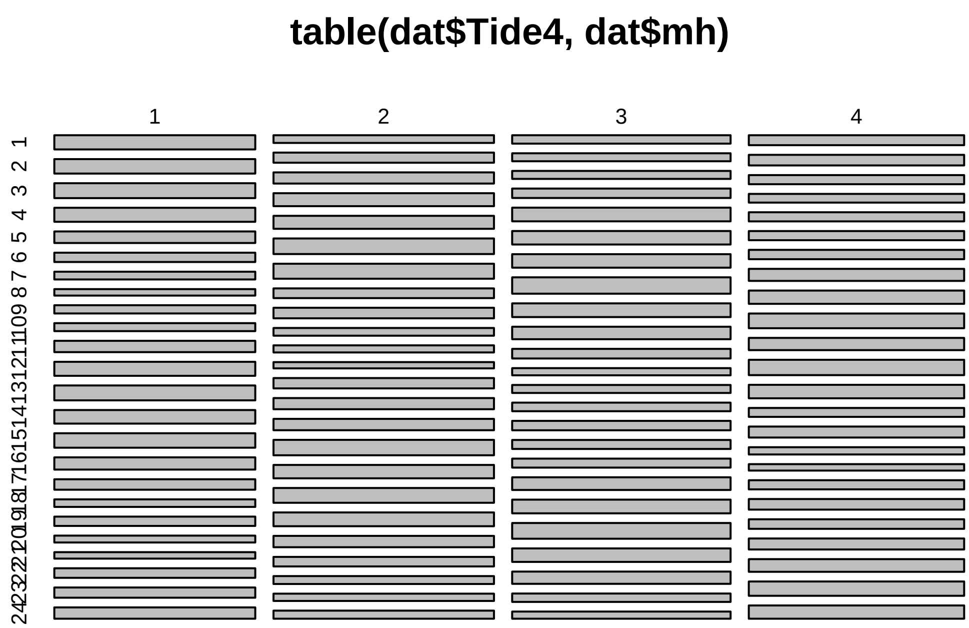
```r
# count observations per year/month combination and represent as mosaicplot
plot(table(dat$year, dat$mon))
```



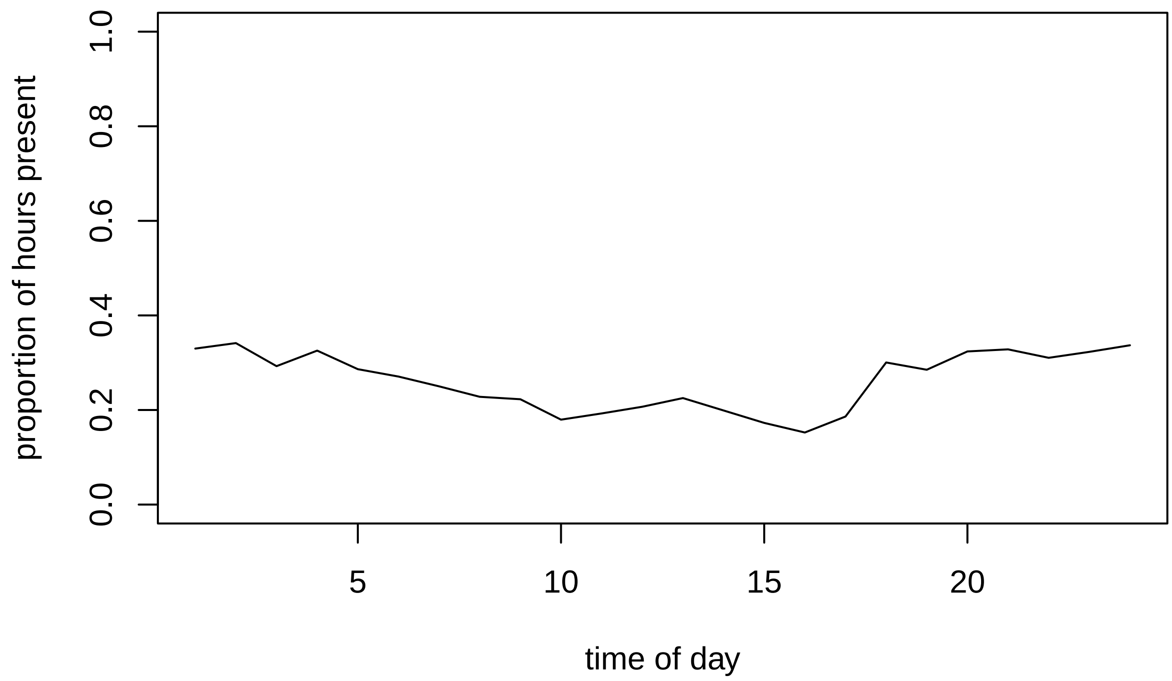

```r
# CPOD failure in Feb-April 2012 and Dec 2012-March 2013

plot(table(dat$mh))
```



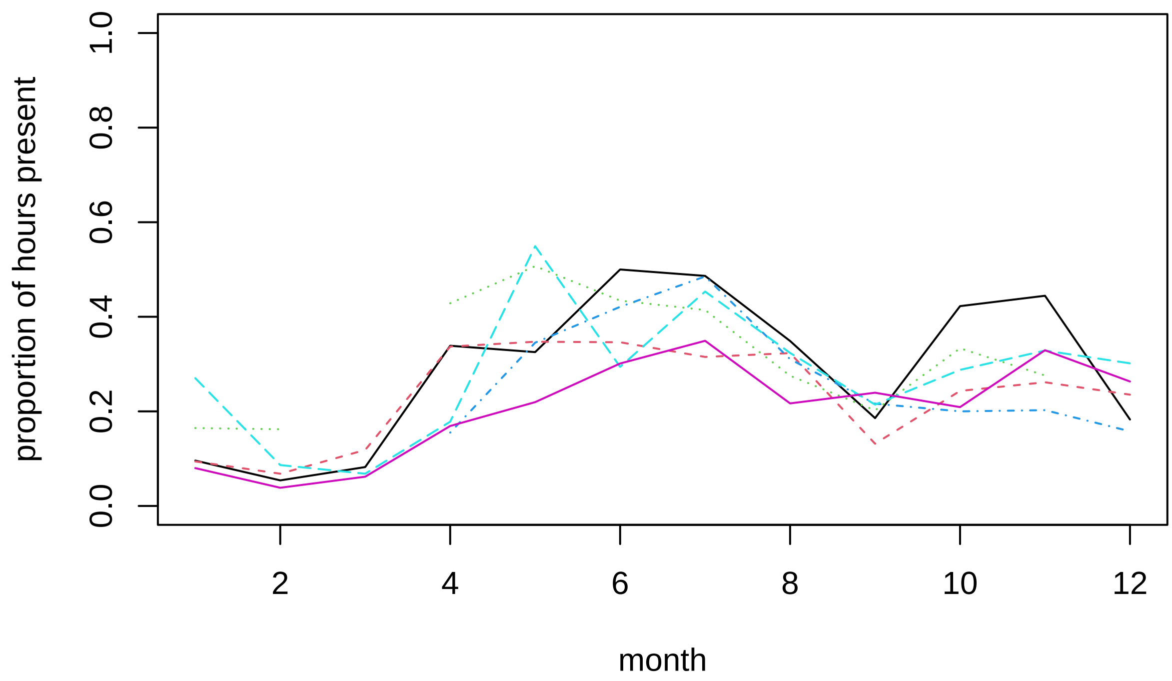

```
# fairly even representation of hours
# (that's on the random sample; Almost perfectly balanced on the full dataset)

plot(table(dat$Tide4, dat$mh))
```



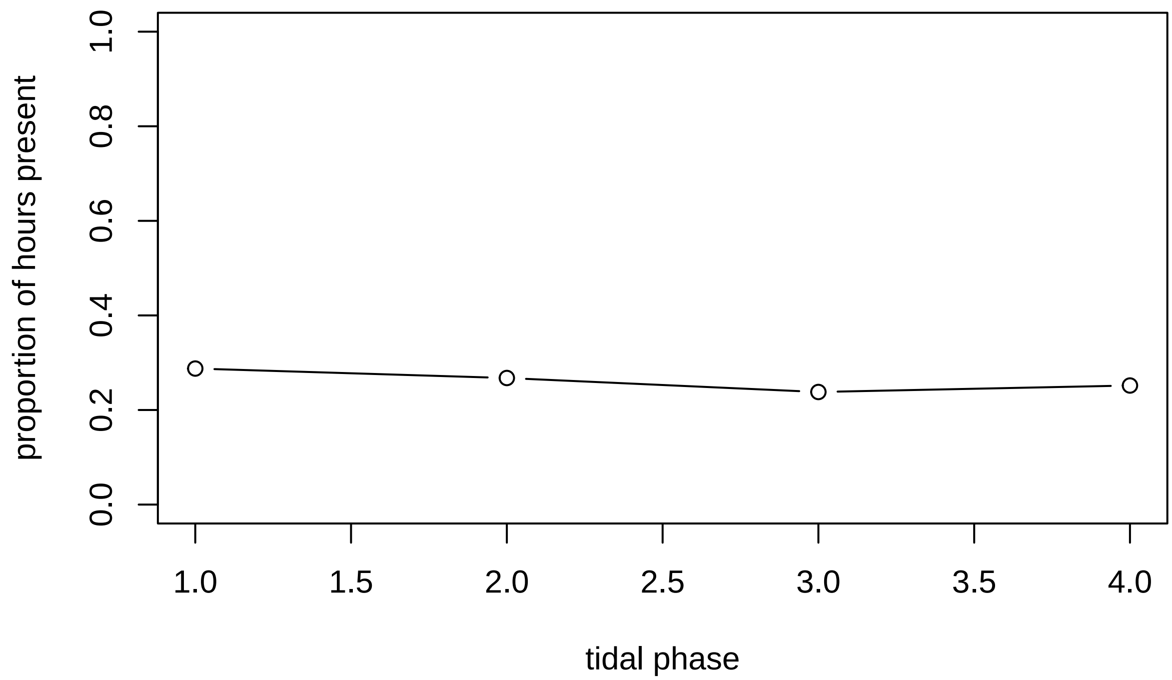

```
# even representation of tides
# time of day and tidal phase not independent (but not a linear correlation)

# presence in relation to time of day
plot(tapply(dat$presence, list(dat$mh), mean), type= "l", ylim= c(0, 1),
         xlab= "time of day", ylab= "proportion of hours present")
```



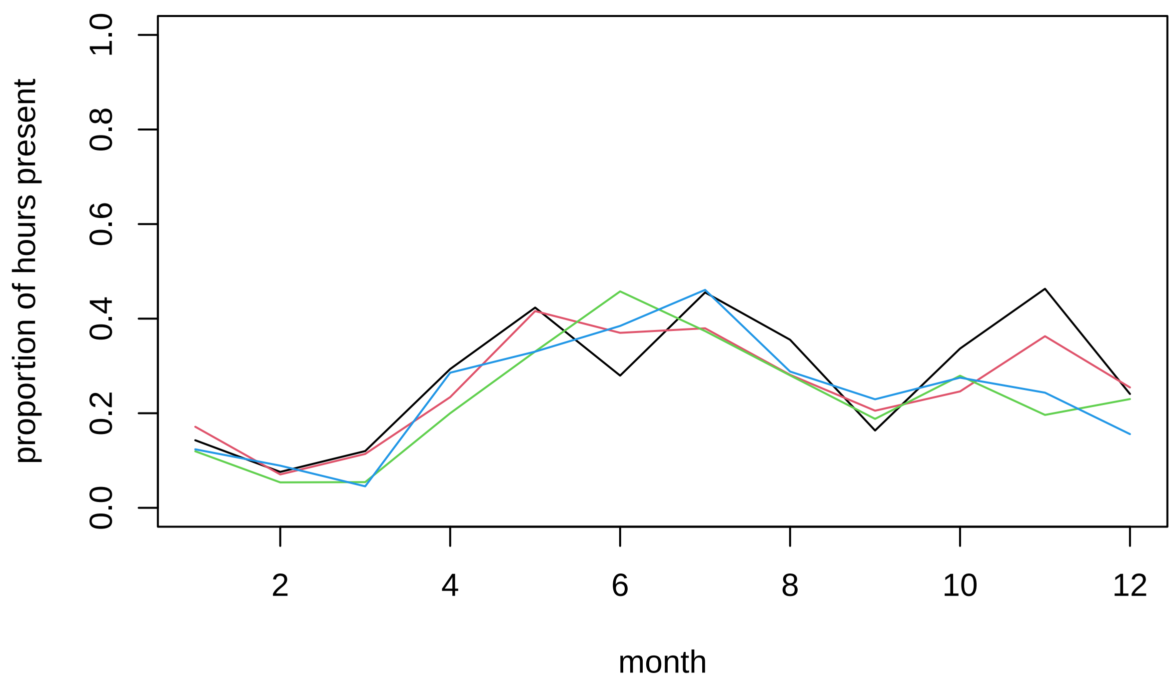

```
# are seasonal patterns similar between years?
matplot(tapply(dat$presence, list(dat$mon, dat$year), mean), type= "l",
                ylim= c(0, 1), xlab= "month", ylab= "proportion of hours present")
```



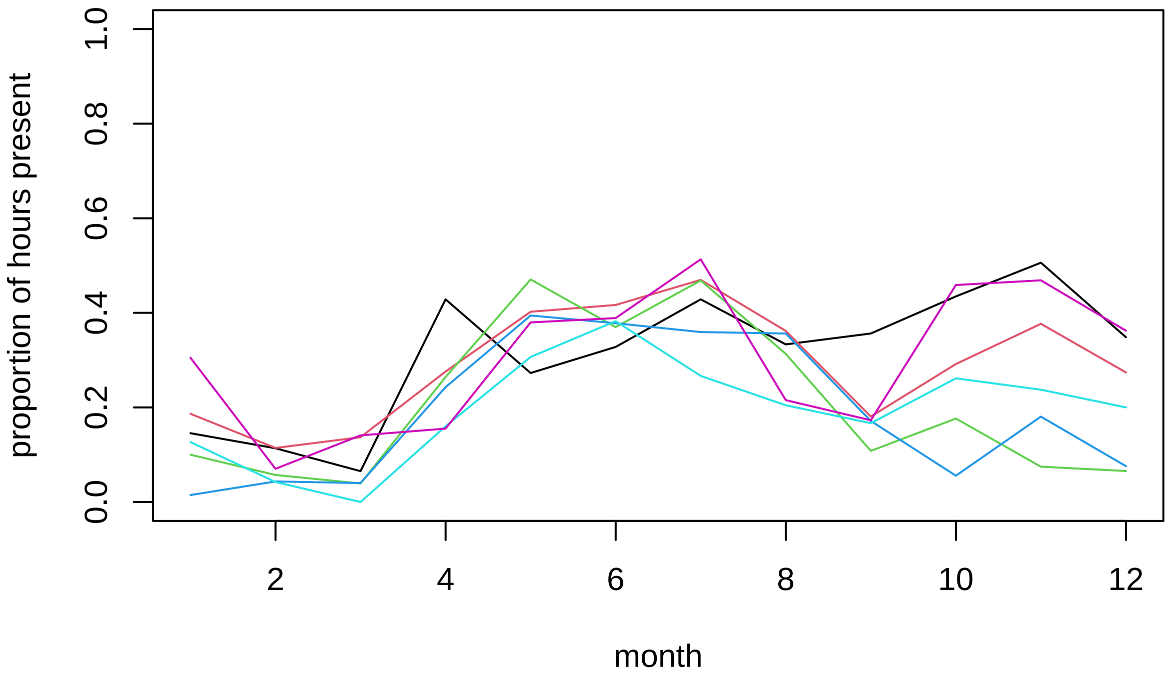

```
# Presence in relation to tide
plot(tapply(dat$presence, list(dat$Tide4), mean), type= "b", ylim= c(0, 1),
     xlab= "tidal phase",
         ylab= "proportion of hours present")
```



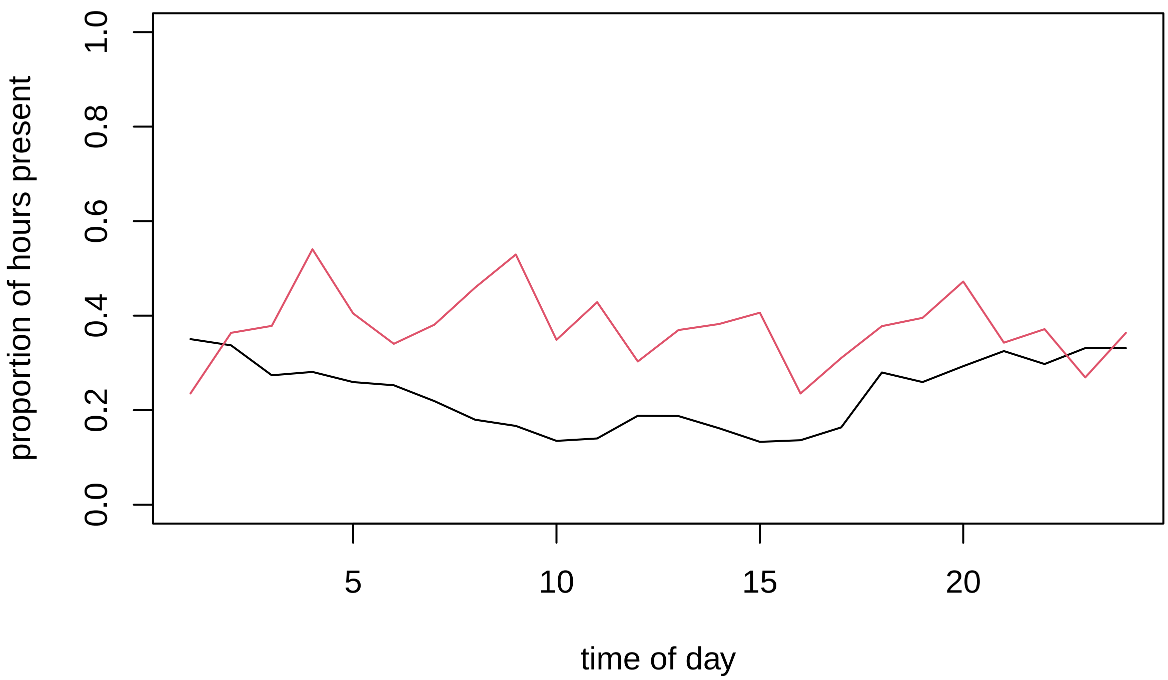

```
matplot(tapply(dat$presence, list(dat$mon, dat$Tide4), mean), type= "l",
       ylim= c(0, 1),
             xlab= "month", ylab= "proportion of hours present", lty= 1)
```



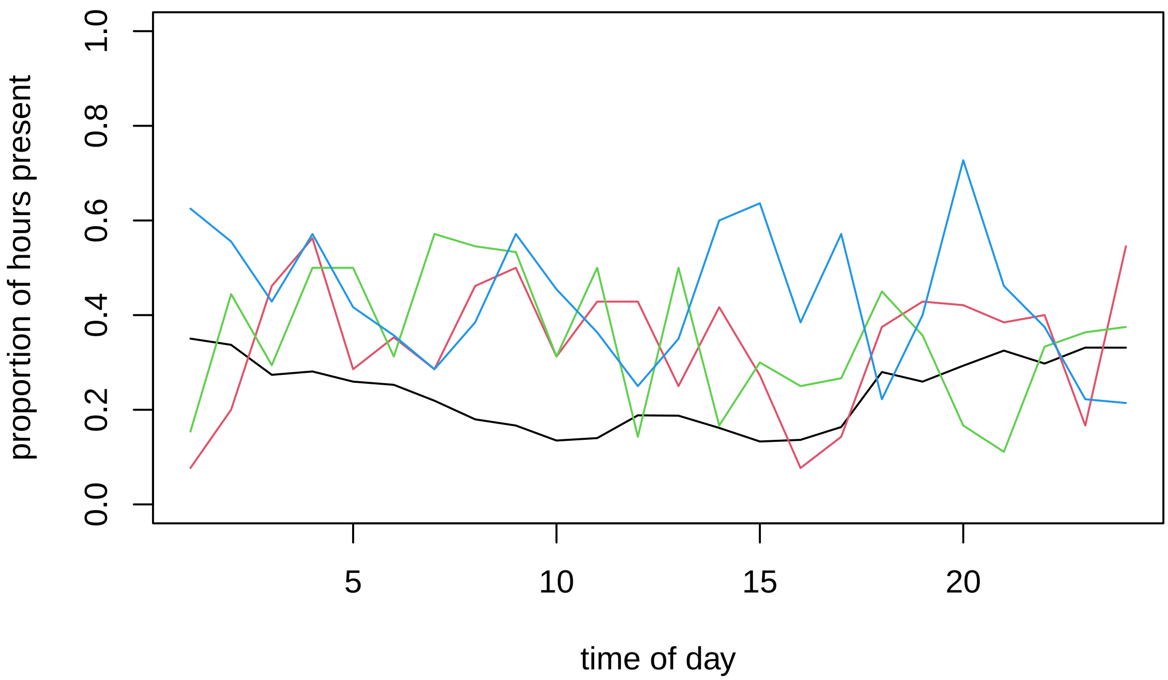

```
# no change in pattern of tide use across seasons

# Seasonal variation in diel pattern
matplot(tapply(dat$presence, list(dat$mon, dat$Time6), mean), type= "l",
        ylim= c(0, 1),
                xlab= "month", ylab= "proportion of hours present", lty= 1)
```




```r
# stronger diel pattern in later part of the year

# Variation in diel pattern between May-June (red) and the rest of the year
matplot(tapply(dat$presence, list(dat$mh, dat$Per2), mean), type= "l",
        ylim= c(0, 1),
                xlab= "time of day", ylab= "proportion of hours present", lty= 1)
```



```
# less nocturnal in spring?

# with more categories in spring
matplot(tapply(dat$presence, list(dat$mh, dat$Per4), mean), type= "l",
        ylim= c(0, 1),
                xlab= "time of day", ylab= "proportion of hours present", lty= 1)
```




```
# no obvious systematic difference between the 3 portions of May-June
```

4. Let's start with a Binomial (Bernoulli) GLM (using `glm()` and the appropriate `family` argument) with all interactions between numerical time of day, tide angle and day of the year as predictors: `tideangle_deg + mh + julianday + tideangle_deg:mh + mh:julianday + tideangle_deg:julianday`. Which individual interactions are implied in this formula? (Hint: if unsure, the summary of the model at the next question will list them).

```
PA1<- glm(presence ~ tideangle_deg + mh + julianday + tideangle_deg:mh +
                    mh:julianday + tideangle_deg:julianday, family= binomial, data= dat)
```

5. Obtain summaries of the model output using the `summary()` function. Make sure you understand the mathematical and biological interpretation of the model, by writing down the complete model on paper (with distribution and link function). What biological hypothesis does each term imply, qualitatively?

```
summary(PA1)
##
## Call:
## glm(formula = presence ~ tideangle_deg + mh + julianday + tideangle_deg:mh +
##     mh:julianday + tideangle_deg:julianday, family = binomial,
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9943  -0.8137  -0.7222   1.4591   1.9201
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -1.043e+00  2.069e-01  -5.041 4.62e-07 ***
## tideangle_deg           -1.748e-03  9.006e-04  -1.941   0.0523 .
## mh                      -2.807e-02  1.332e-02  -2.107   0.0351 *
## julianday                1.461e-03  8.238e-04   1.774   0.0761 .
## tideangle_deg:mh         1.029e-04  4.598e-05   2.239   0.0252 *
## mh:julianday             5.484e-05  4.660e-05   1.177   0.2393
## tideangle_deg:julianday  3.627e-07  3.111e-06   0.117   0.9072
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5738.9  on 4999  degrees of freedom
## Residual deviance: 5682.0  on 4993  degrees of freedom
## AIC: 5696
##
## Number of Fisher Scoring iterations: 4

# Model description:
# presence ~ Bernoulli(p)  or presence ~ Binomial(N= 1, p)
# log(p / (1-p)) =
#       -1.04*(Intercept) + -0.0018*tideangle_deg - 0.0028*mh
#       + 0.0015*julianday - 0.00010*tideangle_deg*mh
#       + 3.63e-07*tideangle_deg*julianday + 5.48e-05*mh*julianday

# "(Intercept)" general intercept
# "tideangle_deg" main effect of tide angle
# "mh" main effect of time of day
# "julianday" main effect of day of year
# "tideangle_deg:mh" does effect of tide angle change with time of day?
# "tideangle_deg:julianday" does effect of tide angle change with day of year?
# "mh:julianday" does effect of time of day change with day of year?
```

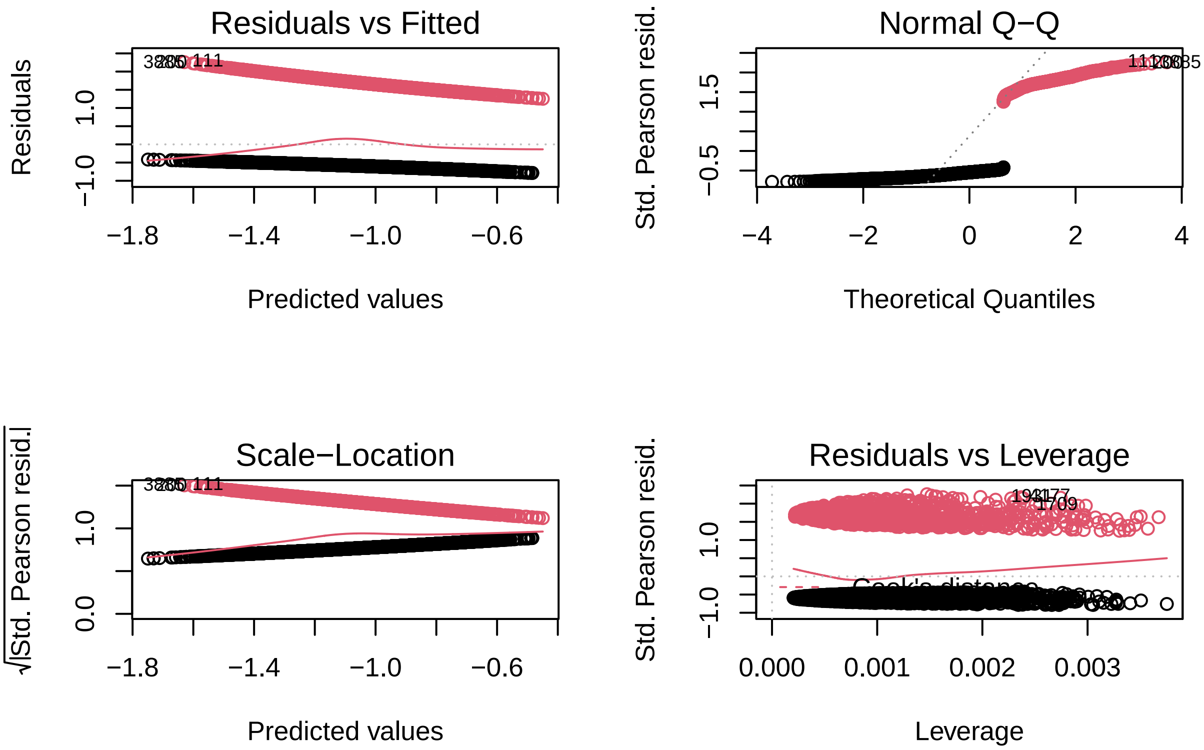
6. Are all the terms significant? if not, simplify the model. Remember to choose the correct ANOVA method (sequential or not), and the appropriate test.

```
drop1(PA1, test= "Chisq")
## Single term deletions
##
## Model:
## presence ~ tideangle_deg + mh + julianday + tideangle_deg:mh +
##     mh:julianday + tideangle_deg:julianday
##                         Df Deviance    AIC    LRT Pr(>Chi)
## <none>                       5682.0 5696.0
## tideangle_deg:mh         1   5687.0 5699.0 5.0173  0.02509 *
## mh:julianday             1   5683.3 5695.3 1.3857  0.23913
## tideangle_deg:julianday  1   5682.0 5694.0 0.0136  0.90719
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# drop tideangle_deg:julianday

PA2<- update(PA1, . ~ . - tideangle_deg:julianday)
drop1(PA2, test= "Chisq")
## Single term deletions
##
## Model:
## presence ~ tideangle_deg + mh + julianday + tideangle_deg:mh +
##     mh:julianday
##                  Df Deviance    AIC    LRT Pr(>Chi)
## <none>                5682.0 5694.0
## tideangle_deg:mh  1   5687.0 5697.0 5.0106  0.02519 *
## mh:julianday      1   5683.4 5693.4 1.4203  0.23335
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# drop mh:julianday

PA3<- update(PA2, . ~ . - mh:julianday)
drop1(PA3, test= "Chisq")
## Single term deletions
##
## Model:
## presence ~ tideangle_deg + mh + julianday + tideangle_deg:mh
##                  Df Deviance    AIC    LRT Pr(>Chi)
## <none>                5683.4 5693.4
## julianday         1   5732.4 5740.4 49.007 2.55e-12 ***
## tideangle_deg:mh  1   5688.4 5696.4  4.996  0.02541 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# nothing left to drop.

# writing final model in full for clarity:
PA4<- glm(presence ~ tideangle_deg + mh + julianday + tideangle_deg:mh,
          family= binomial, data= dat)

summary(PA4)
##
## Call:
## glm(formula = presence ~ tideangle_deg + mh + julianday + tideangle_deg:mh,
##     family = binomial, data = dat)
```

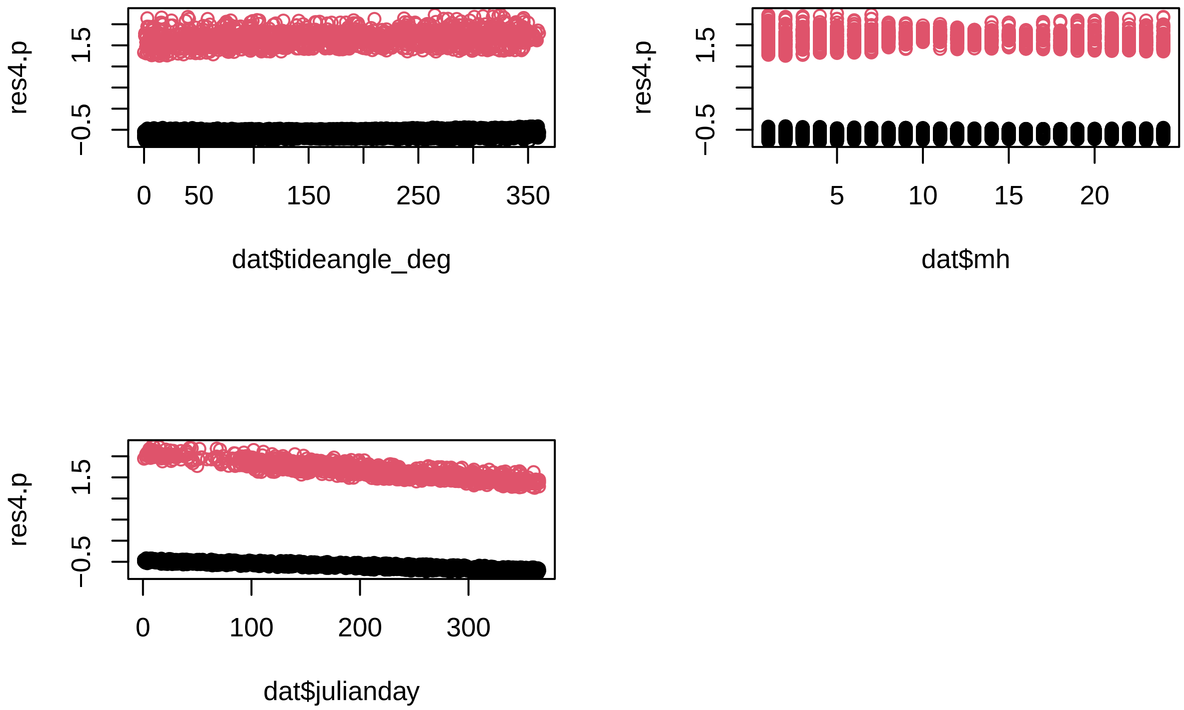
```
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9800  -0.8147  -0.7204   1.4647   1.9019
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -1.191e+00  1.416e-01  -8.413  < 2e-16 ***
## tideangle_deg    -1.684e-03  6.412e-04  -2.626  0.00864 **
## mh               -1.696e-02  9.406e-03  -1.803  0.07136 .
## julianday         2.206e-03  3.177e-04   6.944 3.81e-12 ***
## tideangle_deg:mh  1.026e-04  4.594e-05   2.234  0.02548 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5738.9  on 4999  degrees of freedom
## Residual deviance: 5683.4  on 4995  degrees of freedom
## AIC: 5693.4
##
## Number of Fisher Scoring iterations: 4
```

7. Let's now validate the model, using deviance residuals. The easiest tool is the `binnedplot()` in the `arm` package, if you can. If you can install the arm package and access its binnedplot, use the "DIY" alternative code chunk further down.

```
library(car)
vif(PA4)
##    tideangle_deg               mh        julianday tideangle_deg:mh
##         4.092964         3.973871         1.001209         7.528421
# High for terms involved in the interaction, as expected. No concern.

par(mfrow= c(2, 2))
plot(PA4, col= dat$presence + 1) # red is presence, black is absence
```




```
# Not very useful statistical art. Not worth framing either.

# plot against predictors:
res4.p<- resid(PA4, type= "pearson")

par(mfrow= c(2, 2))
plot(res4.p ~ dat$tideangle_deg, col= dat$presence + 1)

plot(res4.p ~ dat$mh, col= dat$presence + 1)

plot(res4.p ~ dat$julianday, col= dat$presence + 1)

# Can't see anything useful.

# Use arm if you can:
library(arm)
par(mfrow= c(2, 2))
```

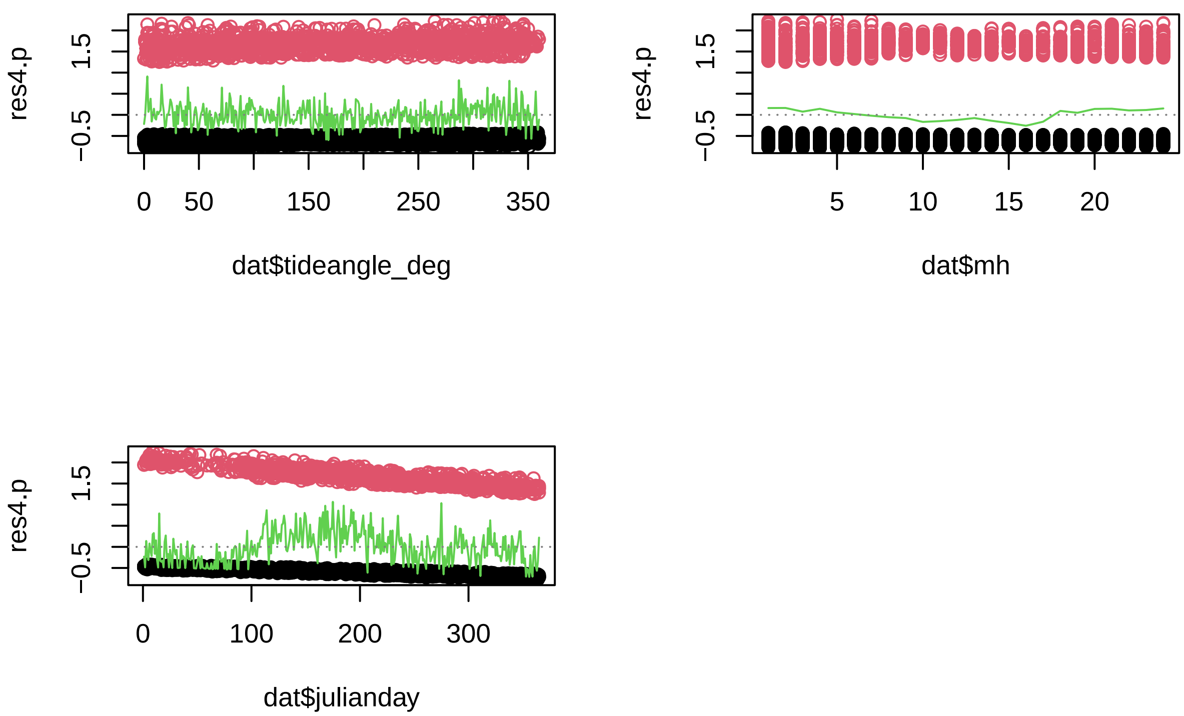



```
binnedplot(x= dat$tideangle_deg, y= res4.p, xlab= "Tide angle", nclass= 100)
binnedplot(x= dat$mh, y= res4.p, xlab= "hour")
binnedplot(x= dat$julianday, y= res4.p, xlab= "Day of the year", nclass= 100)
```

```
# clearly some unwanted patterns, especially in mh and julianday
# but possibly in tide angle, too
# all pointing at non-linear effects of the predictors on the response
```

In case you can't use `binnedplot`, here is a home-made version:

```
par(mfrow= c(2, 2))
plot(res4.p ~ dat$tideangle_deg, col= dat$presence + 1)
tide.means<- tapply(res4.p, list(dat$tideangle_deg), mean)
tide.vals<- as.numeric(names(tide.means))
lines(tide.means ~ tide.vals, col= 3)
abline(h= 0, lty= 3, col= grey(0.5))

plot(res4.p ~ dat$mh, col= dat$presence + 1)
hour.means<- tapply(res4.p, list(dat$mh), mean)
lines(hour.means ~ as.numeric(names(hour.means)), col= 3)
abline(h= 0, lty= 3, col= grey(0.5))

plot(res4.p ~ dat$julianday, col= dat$presence + 1)
day.means<- tapply(res4.p, list(dat$julianday), mean)
lines(day.means ~ as.numeric(names(day.means)), col= 3)
abline(h= 0, lty= 3, col= grey(0.5))

# Same story.
```



8. Are you happy with the diagnostic plots? Is there something you could do to improve the model while addressing the initial question(s)? Spend some time looking at the available predictors, and working out a solution, before unfolding the hints in the code chunk below. If you have relevant biological information, or insight from your data exploration that suggests a better approach than what is indicated below, feel free to try it for comparison.

```
# there are several ways the non-linearity could be addressed.
# one of the most straightforward with glm() is to discretize
# continuous predictors into bins and to treat them as factors.

# Each of the predictors we started with already has one or more
# categorical counterpart in the data set.
# I suggest you try fTide4 + Per2 + Time6 + fTide4:Per2 + fTide4:Time6 + Per2:Time6, with fTide4 being
# You can choose something else or cut your own predictors, too.
```

9. Apply the usual model selection approach for the new version of the model. What are the main sources of variation in the data? What is the proportion of deviance explained?

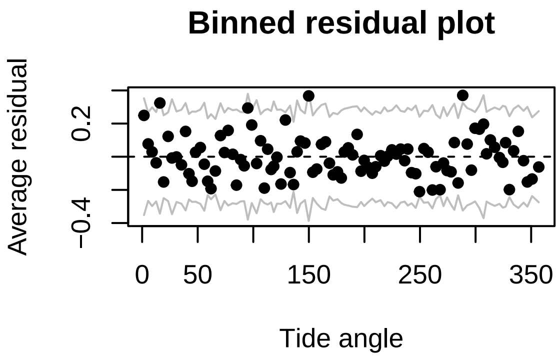
```
# convert numerically coded categorical variables into factors:
dat$fTide4<- factor(dat$Tide4)

PA10<- glm(presence ~ fTide4 + Per2 + Time6 + fTide4:Per2 + fTide4:Time6 +
                        Per2:Time6, family= binomial, data= dat)

drop1(PA10, test= "Chisq")
## Single term deletions
##
## Model:
## presence ~ fTide4 + Per2 + Time6 + fTide4:Per2 + fTide4:Time6 +
##     Per2:Time6
##              Df Deviance    AIC     LRT  Pr(>Chi)
## <none>            5540.9 5606.9
## fTide4:Per2   3   5543.5 5603.5  2.5905    0.4592
## fTide4:Time6 15   5551.9 5587.9 11.0235    0.7509
## Per2:Time6    5   5568.5 5624.5 27.6558 4.249e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# drop fTide4:Time6

PA11<- glm(presence ~ fTide4 + Per2 + Time6 + fTide4:Per2 + Per2:Time6,
                     family= binomial, data= dat)

drop1(PA11, test= "Chisq")
## Single term deletions
##
## Model:
## presence ~ fTide4 + Per2 + Time6 + fTide4:Per2 + Per2:Time6
```

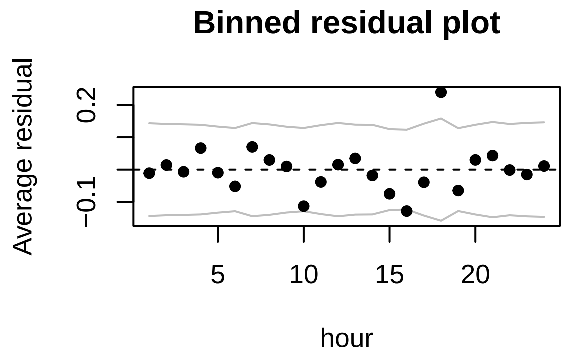
```
##             Df Deviance    AIC     LRT  Pr(>Chi)
## <none>           5551.9 5587.9
## fTide4:Per2  3   5554.8 5584.8  2.9408    0.4008
## Per2:Time6   5   5581.0 5607.0 29.0986 2.218e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# drop fTide4:Per2

PA12<- glm(presence ~ fTide4 + Per2 + Time6 + Per2:Time6,
                     family= binomial, data= dat)

drop1(PA12, test= "Chisq")
## Single term deletions
##
## Model:
## presence ~ fTide4 + Per2 + Time6 + Per2:Time6
##            Df Deviance    AIC    LRT  Pr(>Chi)
## <none>          5554.8 5584.8
## fTide4      3   5565.3 5589.3 10.511   0.01469 *
## Per2:Time6  5   5585.4 5605.4 30.561 1.144e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# nothing else to drop

summary(PA12)
##
## Call:
## glm(formula = presence ~ fTide4 + Per2 + Time6 + Per2:Time6,
##     family = binomial, data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1214  -0.8341  -0.6417   1.3101   1.9492
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -0.53630    0.09887  -5.424 5.82e-08 ***
## fTide42                -0.09815    0.09235  -1.063 0.287880
## fTide43                -0.28709    0.09476  -3.030 0.002448 **
## fTide44                -0.20352    0.09372  -2.172 0.029890 *
## Per2MayJun             -0.13670    0.21003  -0.651 0.515139
## Time6AM1               -0.34058    0.11909  -2.860 0.004239 **
## Time6AM2               -0.84117    0.12957  -6.492 8.47e-11 ***
## Time6MDay              -0.91427    0.13217  -6.917 4.61e-12 ***
## Time6PM1               -0.86865    0.13039  -6.662 2.70e-11 ***
## Time6PM2               -0.17860    0.11860  -1.506 0.132082
## Per2MayJun:Time6AM1     0.80507    0.27719   2.904 0.003680 **
## Per2MayJun:Time6AM2     1.38095    0.28335   4.874 1.10e-06 ***
## Per2MayJun:Time6MDay    1.19832    0.28939   4.141 3.46e-05 ***
## Per2MayJun:Time6PM1     1.00271    0.29379   3.413 0.000642 ***
## Per2MayJun:Time6PM2     0.59002    0.28160   2.095 0.036150 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

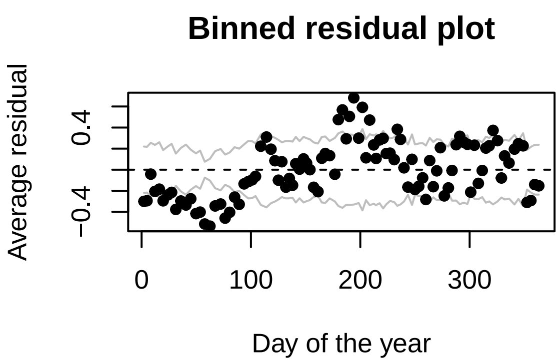
```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5738.9  on 4999  degrees of freedom
## Residual deviance: 5554.8  on 4985  degrees of freedom
## AIC: 5584.8
##
## Number of Fisher Scoring iterations: 4
anova(PA12, test= "Chisq")
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: presence
##
## Terms added sequentially (first to last)
##
##
##            Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                        4999     5738.9
## fTide4      3    8.645      4996     5730.3    0.0344 *
## Per2        1   71.873      4995     5658.4 < 2.2e-16 ***
## Time6       5   73.013      4990     5585.4 2.416e-14 ***
## Per2:Time6  5   30.561      4985     5554.8 1.144e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# fTide4 contributes minimally

# Total proportion of deviance explained is
(PA12$null.deviance - PA12$deviance) / PA12$null.deviance # 3%
## [1] 0.03207773
```

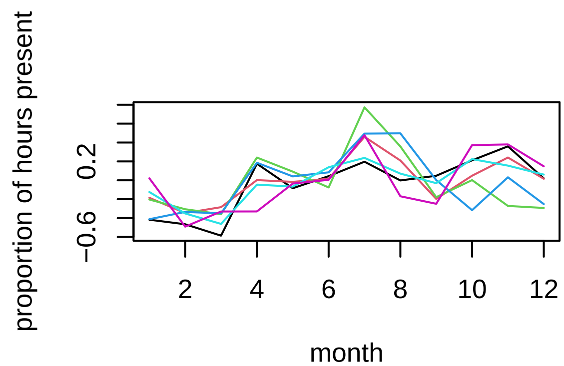
10. Do the model validation for the minimal adequate model. Is everything looking good?

```
# plot against predictors:
 res12.p<- resid(PA12, type= "pearson")

library(arm)
par(mfrow= c(2, 2))
binnedplot(x= dat$tideangle_deg, y=  res12.p, xlab= "Tide angle", nclass= 100)
# okay
binnedplot(x= dat$mh, y=  res12.p, xlab= "hour")
# okay
binnedplot(x= dat$julianday, y=  res12.p, xlab= "Day of the year", nclass= 100)
# less than good (model only allows for difference between May-June
# and rest of the year)

# Check seasonal variation in diel pattern again ("time by season" interaction):
matplot(tapply(res12.p, list(dat$mon, dat$Time6), mean), type= "l",
        xlab= "month", ylab= "proportion of hours present", lty= 1)
```










```
# still residual variation in diel pattern in later part of the year
# (not surprising as there is nothing in the model aiming at capturing this)

# residuals suggest a finer binning of time of the year
# is required for the predictors. Or an approach that circumvents the issues
# with binning (see reference at the end for an alternative approach).
```

11. Assuming that the model is fine as it is, let's plot the predictions with their confidence intervals for the probability of presence in relation to time of day, in both the May/June period and the rest of the year. For tide, assume it is fixed at a level of your choice, e.g. "1". Suggested approach:

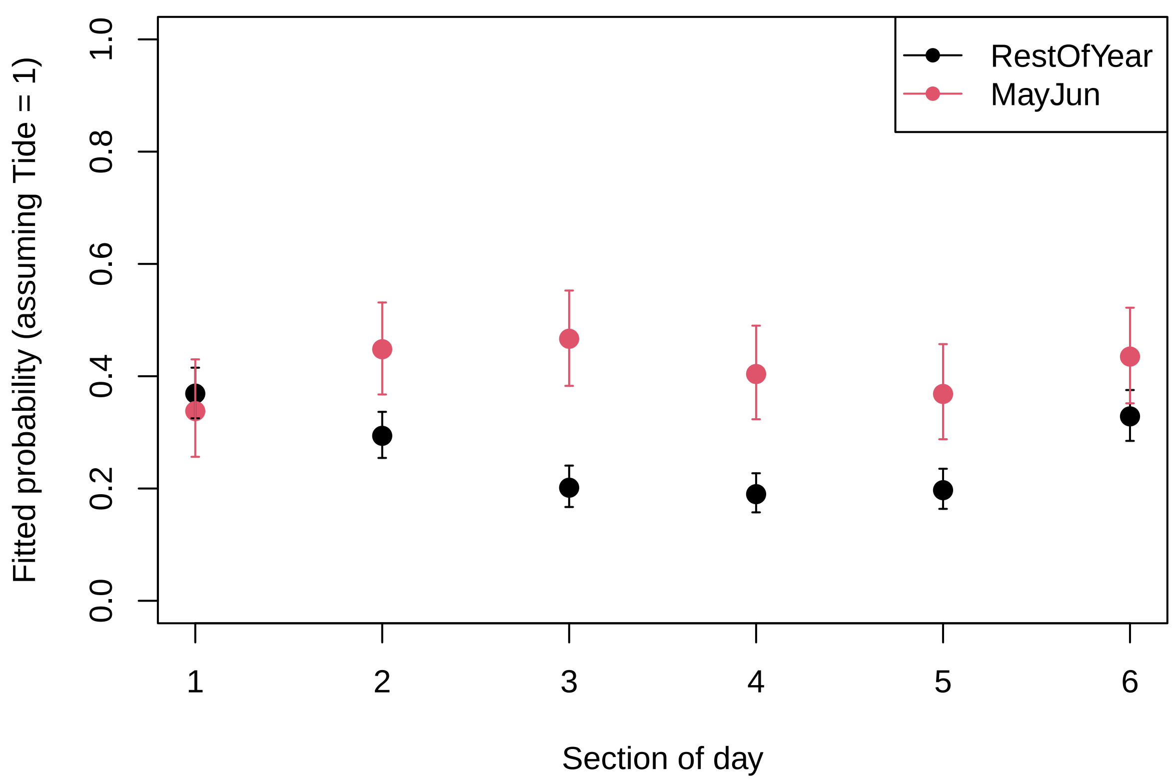
- create a `data.frame` called `X` containing the data to predict for. This can be done by hand following previous examples or using the function `expand.grid` for creating all the combinations of the variables of interest: `expand.grid(NameOfVar1 = levels(data$NameOfVar1), NameOfVar2= levels(data$NameOfVar2), NameOfVar3= "1")`
- use `predict()` with the appropriate options to obtain the fitted values on the link scale and for being able to calculate the confidence intervals later. Store in object `Z`.
- plot fitted values, extracted using `Z$fit`, against the appropriate column of X (you can use different symbols or colours for groups).
- in X, add columns for the fitted values and their confidence intervals, on the response scale (to be calculated).
- use the function `segments` or `arrows` to add confidence intervals to the fitted values (see the help page for the respective function).

The code is available below for you to unfold if you don't want to try yourself.

```r
PA12.dat4pred<- expand.grid(Time6= levels(dat$Time6),
                            Per2= levels(dat$Per2),
                            fTide4= "1")
```

```r
PA12.pred<- predict(PA12, PA12.dat4pred, type= "link", se.fit= T)

PA12.dat4pred$fit.resp<- exp(PA12.pred$fit)/(1+exp(PA12.pred$fit))
# or plogis(PA12.pred$fit)

# lower 95% CI
PA12.dat4pred$LCI<- plogis(PA12.pred$fit - 1.96*PA12.pred$se.fit)
# upper 95% CI
PA12.dat4pred$UCI<- plogis(PA12.pred$fit + 1.96*PA12.pred$se.fit)
```

```r
par(mfrow= c(1, 1))
plot(as.numeric(PA12.dat4pred$Time6), PA12.dat4pred$fit.resp, pch= 16, cex= 1.4,
      col= PA12.dat4pred$Per2, xlab= "Section of day",
          ylab= "Fitted probability (assuming Tide = 1)", ylim= c(0, 1))

arrows(x0= as.numeric(PA12.dat4pred$Time6), x1= as.numeric(PA12.dat4pred$Time6),
          y0= PA12.dat4pred$LCI, y1= PA12.dat4pred$UCI,
          col= PA12.dat4pred$Per2, length= 0.02, angle= 90, code= 3)

legend(x= "topright", legend= c("RestOfYear", "MayJun"), col= c(1, 2), lty= 1, pch= 16)
```

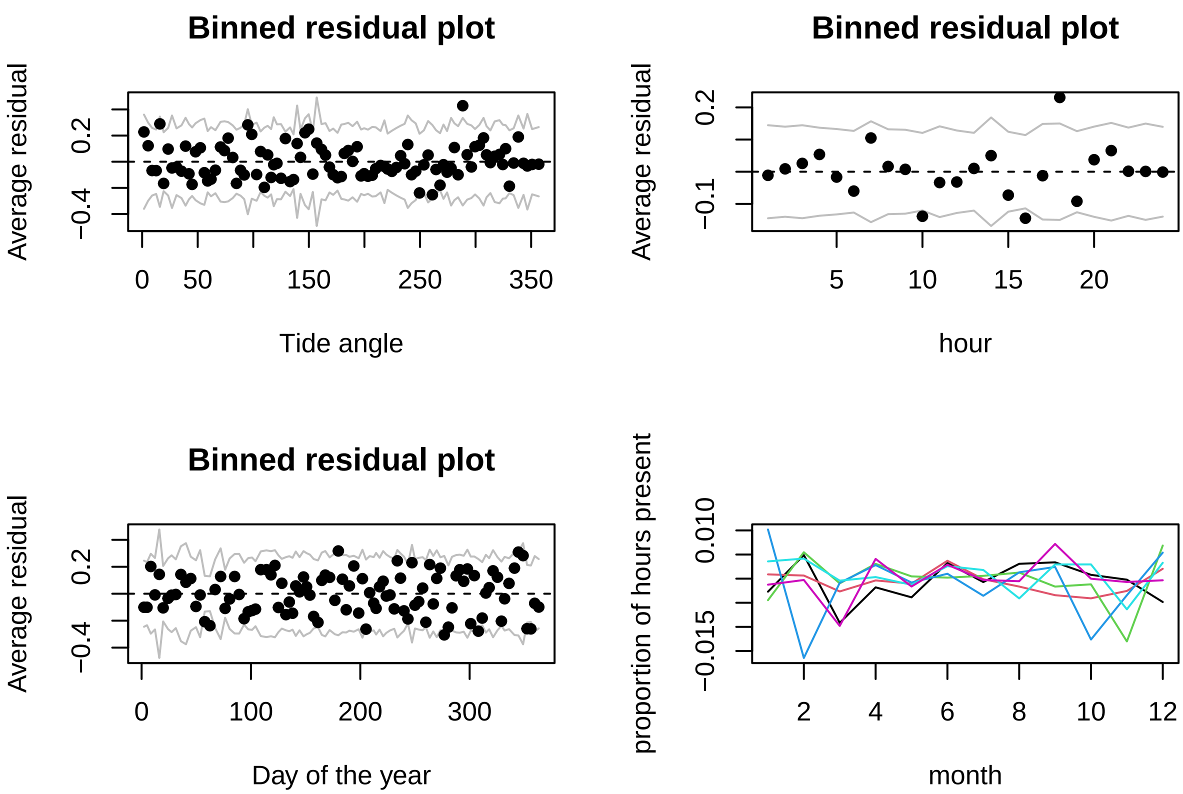


12. Repeat the model selection this time using AIC, with `step()`. Do you obtain the same minimal adequate model? Then replace `Per2` by month `mon` (as a factor) for a finer seasonal resolution, and apply a model selection with `step()` again. Is the same model structure preferred? Which of the `Per2` or `mon` models is favoured by AIC? Do the residuals look better?

```
PA10.MAM.stepAIC<- step(PA10)
## Start:  AIC=5606.86
## presence ~ fTide4 + Per2 + Time6 + fTide4:Per2 + fTide4:Time6 +
##     Per2:Time6
##
##                Df Deviance    AIC
## - fTide4:Time6 15   5551.9 5587.9
## - fTide4:Per2   3   5543.5 5603.5
## <none>              5540.9 5606.9
## - Per2:Time6    5   5568.5 5624.5
##
## Step:  AIC=5587.89
## presence ~ fTide4 + Per2 + Time6 + fTide4:Per2 + Per2:Time6
##
##               Df Deviance    AIC
## - fTide4:Per2  3   5554.8 5584.8
## <none>             5551.9 5587.9
## - Per2:Time6   5   5581.0 5607.0
##
## Step:  AIC=5584.83
## presence ~ fTide4 + Per2 + Time6 + Per2:Time6
##
##              Df Deviance    AIC
## <none>            5554.8 5584.8
## - fTide4      3   5565.3 5589.3
## - Per2:Time6  5   5585.4 5605.4

anova(PA10.MAM.stepAIC, test= "Chisq")
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: presence
##
## Terms added sequentially (first to last)
##
##
##            Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                        4999     5738.9
## fTide4      3    8.645      4996     5730.3    0.0344 *
## Per2        1   71.873      4995     5658.4 < 2.2e-16 ***
## Time6       5   73.013      4990     5585.4 2.416e-14 ***
## Per2:Time6  5   30.561      4985     5554.8 1.144e-05 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# same model as PA13

# convert 'mon' into factor
dat$fMonth<- factor(dat$mon)
# fit new model
PA20<- glm(presence ~ fTide4 + fMonth + Time6 + fTide4:fMonth + fTide4:Time6 +
                        fMonth:Time6, family= binomial, data= dat)

PA20.MAM.stepAIC<- step(PA20)
## Start:  AIC=5343.08
## presence ~ fTide4 + fMonth + Time6 + fTide4:fMonth + fTide4:Time6 +
##     fMonth:Time6
##
##                  Df Deviance    AIC
## - fTide4:Time6   15   5110.6 5326.6
## - fTide4:fMonth 33   5148.2 5328.2
## <none>               5097.1 5343.1
## - fMonth:Time6  55   5269.7 5405.7
##
## Step:  AIC=5326.56
## presence ~ fTide4 + fMonth + Time6 + fTide4:fMonth + fMonth:Time6
##
##                  Df Deviance    AIC
## - fTide4:fMonth 33   5160.1 5310.1
## <none>               5110.6 5326.6
## - fMonth:Time6  55   5285.2 5391.2
##
## Step:  AIC=5310.09
## presence ~ fTide4 + fMonth + Time6 + fMonth:Time6
##
##                 Df Deviance    AIC
## <none>              5160.1 5310.1
## - fTide4        3   5171.4 5315.4
## - fMonth:Time6 55   5331.9 5371.9

anova(PA20.MAM.stepAIC, test= "Chisq")
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: presence
##
## Terms added sequentially (first to last)
##
##
##              Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                          4999     5738.9
## fTide4        3     8.65      4996     5730.3    0.0344 *
## fMonth       11   318.55      4985     5411.7 < 2.2e-16 ***
## Time6         5    79.83      4980     5331.9 9.113e-16 ***
## fMonth:Time6 55   171.81      4925     5160.1 6.042e-14 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# Same structure selected: tide, plus season (month) by time of day
AIC(PA10.MAM.stepAIC)
## [1] 5584.829
AIC(PA20.MAM.stepAIC)
## [1] 5310.091
# Monthly model vastly favoured despite the 60 extra parameters

anova(PA10.MAM.stepAIC, PA20.MAM.stepAIC, test= "Chisq")
## Analysis of Deviance Table
##
## Model 1: presence ~ fTide4 + Per2 + Time6 + Per2:Time6
## Model 2: presence ~ fTide4 + fMonth + Time6 + fMonth:Time6
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1      4985     5554.8
## 2      4925     5160.1 60   394.74 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# also clearly favoured by likelihood ratio test

# residual analysis:
res20.d<- resid(PA20.MAM.stepAIC, type= "pearson")
library(arm)
par(mfrow= c(2, 2))
binnedplot(x= dat$tideangle_deg, y=   res20.d, xlab= "Tide angle", nclass= 100)
binnedplot(x= dat$mh, y=   res20.d, xlab= "hour")
binnedplot(x= dat$julianday, y=   res20.d, xlab= "Day of the year", nclass= 100)
# Check seasonal variation in diel pattern again ("time by season" interaction):
matplot(tapply( res20.d, list(dat$mon, dat$Time6), mean), type= "l",
        xlab= "month",
                ylab= "proportion of hours present", lty= 1)
```

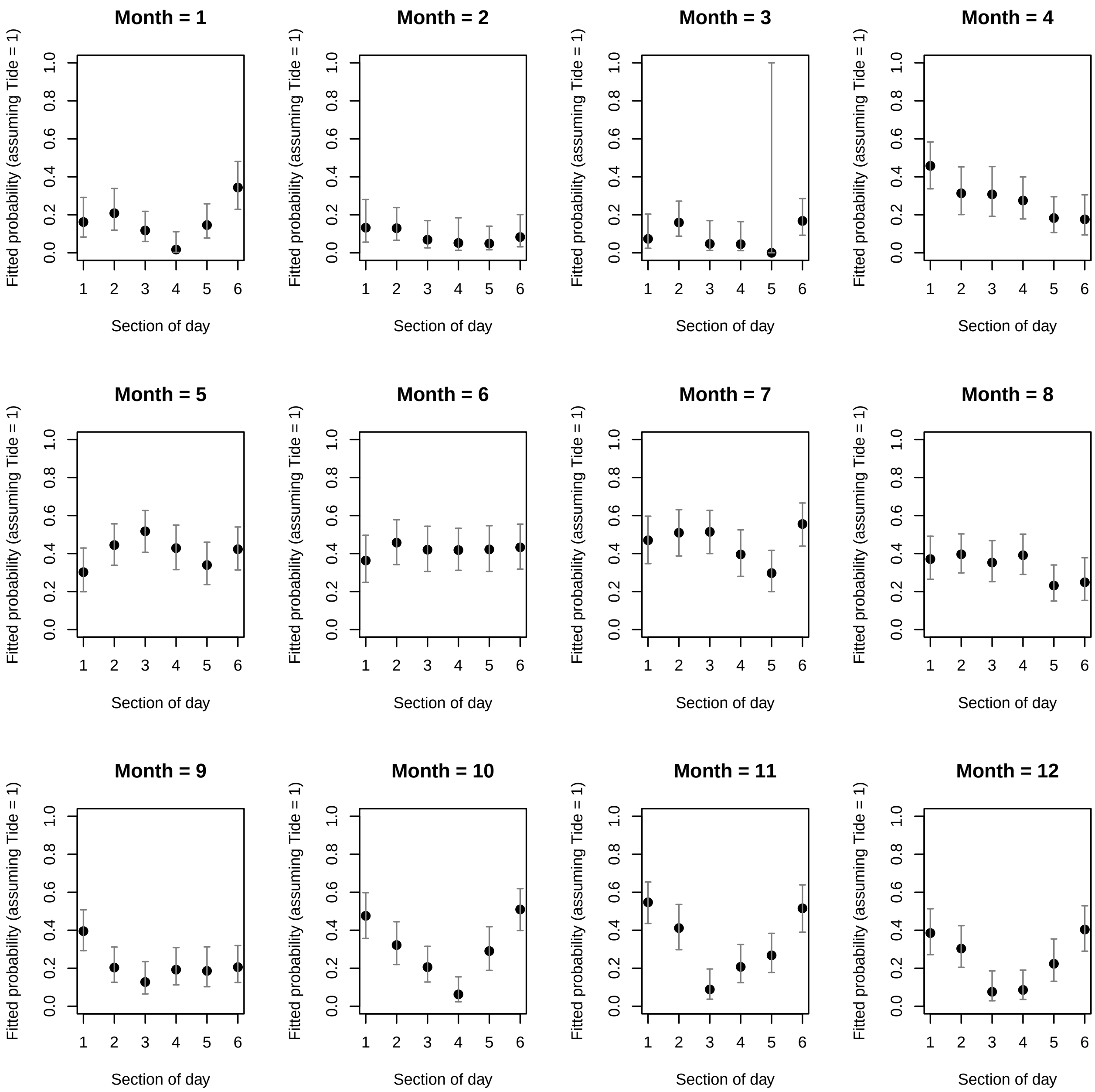



```
# much improved
```

13. How satisfied are you with the model, and with all the assumptions being met? What have you learned from it, with respect to the initial aims of the study? Are there areas of improvement? **Optional** The publication here offers a different approach to analysing these data, using slightly fancier GLMs with smooth terms (called GAMs, for Generalized Additive Models), and a few additional refinements: [https://www.nature.com/articles/s41598-019-38900-4]. What assumptions differ between this and your approach?

```
# dolphins have a weak but apparently stable preference for certain tidal states in Sutors.

# According to model PA13, they are more likely to be seen during the day in
# May/June than in other months where they are more nocturnal (more detail
# could be obtained by plotting the predicted hour of day effect per month
# from PA20.MAM.stepAIC), like this:

par(mfrow= c(3, 4))
for(month in 1:12){
    dat4pred<- expand.grid(Time6= levels(dat$Time6),
                                 fMonth= as.character(month),
                                 fTide4= "1")
    PA20.pred<- predict(PA20.MAM.stepAIC, dat4pred, type= "link", se.fit= T)
    dat4pred$fit.resp<- plogis(PA20.pred$fit)
```

```
    dat4pred$LCI<- plogis(PA20.pred$fit - 1.96*PA20.pred$se.fit)
    dat4pred$UCI<- plogis(PA20.pred$fit + 1.96*PA20.pred$se.fit)
    plot(as.numeric(dat4pred$Time6), dat4pred$fit.resp, pch= 16,
          cex= 1.4, main= paste("Month =", month),
          col= 1, xlab= "Section of day",
          ylab= "Fitted probability (assuming Tide = 1)", ylim= c(0, 1))
    arrows(x0= as.numeric(dat4pred$Time6), x1= as.numeric(dat4pred$Time6),
          y0= dat4pred$LCI, y1= dat4pred$UCI,
          col= grey(0.5), length= 0.02, angle= 90, code= 3)
}
```




```
# According to the better supported model, they tend to use the site more
#  in May, June and
```

```
# July day and night, visit mostly by night from October to December,
#  and seldom from Jan to March.

# There are few assumptions for the Bernoulli distribution other than
# observations being zeros and ones.
# Some assumptions valid for all models still apply here, such as: model
# correctly specified; independent
# residuals. The latter is violated in this data set due to consecutive
# measurements in time. This issue
# is explored in the linked paper, using mixed models for non-independent data
# (covered in course BI5302).
# The paper also uses GAMs for avoiding the discretization of
# continuous variables, and
# accounting for the cyclicity of the preditors (estimates at each end should
# match, e.g. 31st Dec-1st Jan, or 23:59 - 00:00)

# Of note is the low proportion of deviance explained by the model,
# despite its complexity (75 parameters):
(PA20.MAM.stepAIC$null.deviance - PA20.MAM.stepAIC$deviance) /
    PA20.MAM.stepAIC$null.deviance
## [1] 0.1008604
# 10%. This is quite normal with Bernoulli data.
```

End of the Binomial (Bernoulli) GLM - dolphin behavioural plasticity exercise

Appendix. Code for converting the original publicly available data (10 Mb) [https://datadryad.org/stash/dataset/doi:10.5061/dryad.k378542] into the 'dolphin.csv' file. Includes converting numeric variables into categories that you can define to suit your needs (binning), including making more bins if you wish. Binning is done easily using the `cut()` function. For example, creating 5 regular bins is done using `cut(MyVector, breaks= 5)`. Note here that `cut` is used in a non-standard way to make the beginning and end of a cyclic variable belong to the same bin, which may be more biologically meaningful (you can decide, you are the expert!).

```
fulldat<- read.delim("./data/FineScale_Dataset_GAMM_OFB2019.txt")

str(fulldat)

dat<- fulldat[fulldat$site == "Sutors", c("presence", "year", "julianday", "tideangle_deg", "mh")]

dat$mon<- as.numeric(cut(dat$julianday, seq(1, 370, by= 30.5)))

dat$tideangle_deg<- round(dat$tideangle_deg)
# count number of data per year/month combination and represent as mosaicplot
plot(table(dat$year, dat$mon))

# remove 2016
dat<- dat[dat$year != 2016, ]

# Bin year into two periods (May+June vs rest of year)
```

```r
dat$Per2<- cut(dat$julianday, breaks= c(-1, 120, 180, 400),
            labels= c("RestOfYear", "MayJun", "RestOfYear"))
dat$Per2<- factor(dat$Per2, levels= c("RestOfYear", "MayJun"))
# (making "RestOfYear" the reference level)

# check this is working as intended:
plot(as.numeric(dat$Per2) ~ dat$julianday)

# Bin year into 4 periods:
# 3 periods of 20 days from early May to end of June vs rest of the year
dat$Per4<- cut(dat$julianday, breaks= c(0, 120, 140, 160, 180, 400),
            labels= c("RestOfYear", "MayJun1", "MayJun2", "MayJun3", "RestOfYear"))
dat$Per4<- factor(dat$Per4, levels= c("RestOfYear", "MayJun1", "MayJun2", "MayJun3"))
# (reordering levels)

# check this is working as intended:
plot(as.numeric(dat$Per4) ~ dat$julianday)

# Bin time of day into 6 4h periods (first centered on midnight)
dat$Time6<- cut(dat$mh, breaks= c(-1, seq(2, 22, by= 4), 24),
            labels= c("MNight", "AM1", "AM2", "MDay", "PM1", "PM2", "MNight"))
dat$Time6<- factor(dat$Time6, levels= c("MNight", "AM1", "AM2", "MDay", "PM1", "PM2"))
# reordering chronologically

# check this is working as intended:
table(dat$Time6, dat$mh)

# Bin tide angle into 4 quadrants with peaks in middle of respective bin
dat$Tide4<- cut(dat$tideangle_deg, breaks= c(-1, 45, 135, 225, 315, 360),
            labels= c(1:4, 1))

# check this is working as intended:
plot(as.numeric(dat$Tide4) ~ dat$tideangle_deg)

# unless you desperately want to test the performance of your computer,
# play safe and reduce the size of the data set from 50000 to 5000:
set.seed(74) # makes the random sampling reproducible
# This means you will get the same random sample as the solutions to
# the exercises and the same results.
dat<- dat[sample(1:nrow(dat), size= 5000), ] # random subset or rows

write.csv(dat, "dolphin.csv")
```