EMAT30007 Applied Statistics

Lab 10: Generalised Linear Models

Nikolai Bode

In this lab we will look at two common types of Generalised Linear Models (GLMs), Logistic regression and Poisson regression. Recall that GLMs are written in terms of their link function g. For a response Y_i and predictors X_i (we use matrix notation here), a GLM is specified as: $g(\mu_i) = g(E(Y_i)) = X_i\beta$, where $Y_i \sim$ *exponential family distribution*.

(1) Logistic regression

In this part of the lab we will work through an example for logistic regression. To help you get a feel for what the link function does, we will start by fitting the model without using the ready-made model fittting functions in Matlab.

1.1. Data used in logistic regression

As discussed in lectures, logistic regression is used when the response is categorical (e.g. yes/no occurrences, or failure occurrences). Data for logistic regression thus contains information on the outcome of a yes/no-type trial and on variables that we believe influence the probability of the outcome.

In this example, we will look at an illustrative data set on 3d printers. The file `3dprinters.txt` contains three columns. The fitst indicates whether printing process fails (0 - fail, 1 - success), the second includes a measure of the ambient temperature and the third column records the grade of the polymer used for printing (qualitative variable, 3 levels).

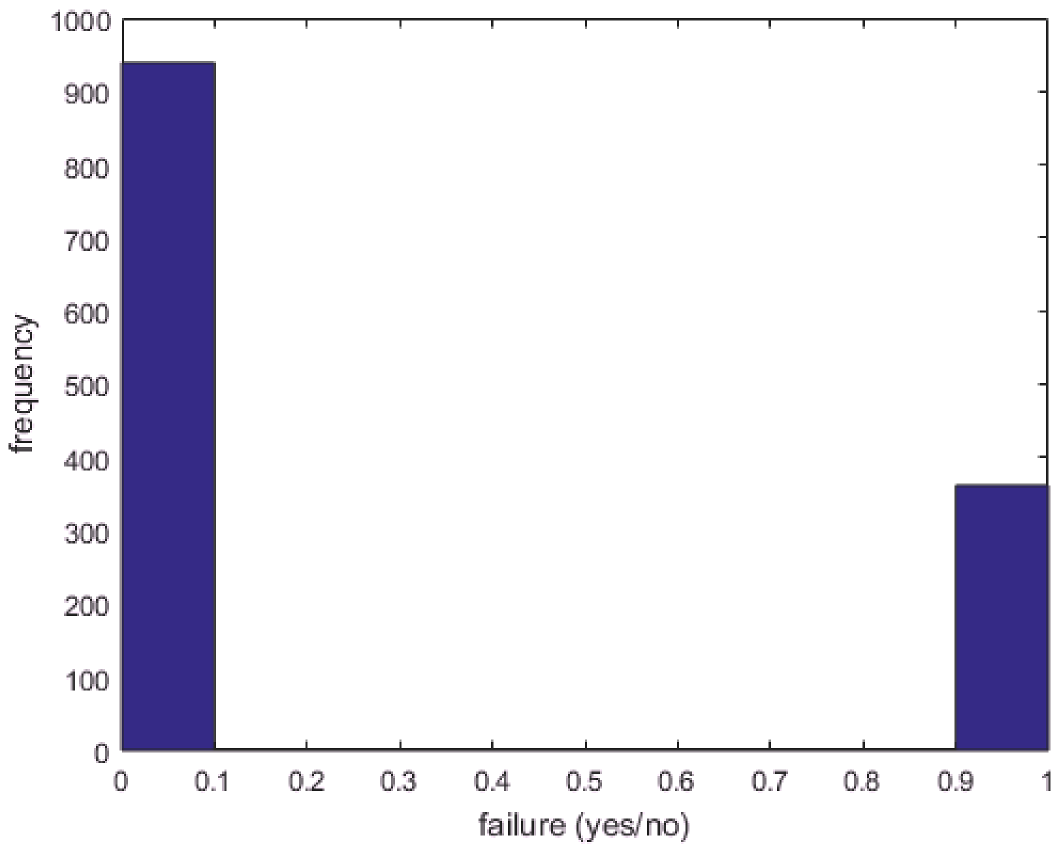
Your tasks is to investigate whether ambient temperature and polymer grade affect the probability that the printing process fails.

Start by reading the data into Matlab. Plot the distribution of succeses and failures, it should take values 0 and 1 (use `hist()`).

```
%% read data:
delimiterIn = ',';
A = importdata('3dprinters.txt',delimiterIn);
A.data(1:5,:)
```

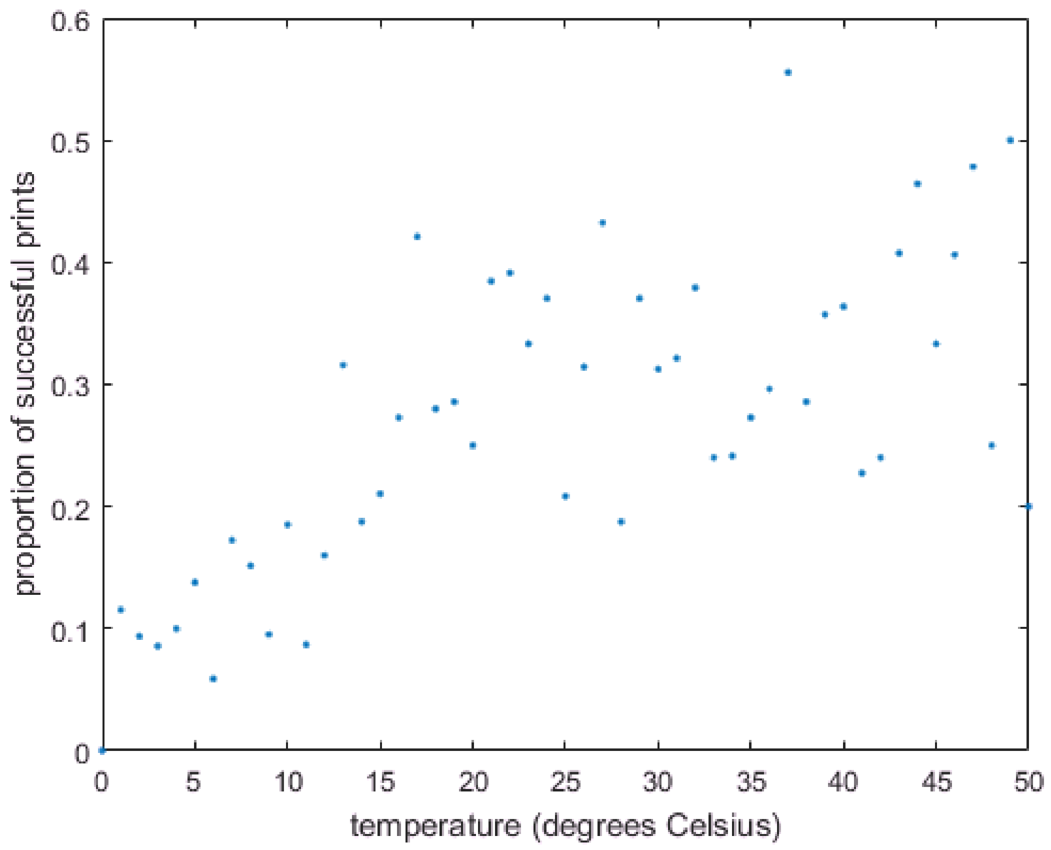
```
ans = 5x3 double

     0    43     0
     0    46     0
     0    27     0
     0    33     0
     0    47     0
```

```
clf
hist(A.data(:,1))
xlabel('failure (yes/no)')
ylabel('frequency')
```



It is always a good idea to explore the raw data before fitting statistical models to it. However, for binary response data, scatterplots are not very usef (try this). To obtain a meaningful plot, we have to work out the proportion of failures for each unique temperature recorded (or for temperature bins, e.g. [0,5] degrees Celsius). Try to prduce a plot like this.

```
% we will look at unique temperatures recorded.
tmp = unique(A.data(:,2));
fraction = zeros(0,length(tmp));
for i = 1:length(tmp)
    idx = find(A.data(:,2)==tmp(i));
    fraction(i) = sum(A.data(idx,1))/length(idx);
end
clf
plot(tmp,fraction,'.')
xlabel('temperature (degrees Celsius)')
ylabel('proportion of successful prints')
```



This plot may look a bit messy, but you should see that the probability for a successful print seems to increase with increasing temperature. Now we will fit a GLM to quantify this trend.

1.2. Fitting a logistic regression model 'by hand'

In lab 6, we used three different approaches to fit a simple Linear regression model. One of these approaches involved writing a Matlab function that evaluated the likelihood for the model given some parameter values and subsequently using an optimisation routine to find the parameter values that maxise the likelihood. We will use the same approach here. You will find the skeleton of the Matlab function you need for the Likelihood in the file `negloglikGLM.m`. You can just alter this function.

For the likelihood function of our logistic regression model, we need to multiply the probabilities for observing our data for all data points. From lecture 10 we know that a GLM for yes/no occurrences uses the binomial distribution. As we only have one outcome per data point, we essentially need to model the probability $P(success)$. Our response is a randome variable Y that tales values 1 and 0 (success and failure). Basic probability theory tells us that $P(success) = E(Y)$. Thus, to find our desired probabilities, we need to apply the inverse of our link function, γ, to our linear predictor $X\beta$ (see above). From lecture 10 we know that a useful link function for our data is the `logit` function, $ln(\frac{\mu}{1-\mu})$. It turns out that the inverse for this is the *logistic function*, $\gamma(a) = \frac{1}{1+e^{-a}}$ (hence the name 'logistic regression').

All we need to do for our likelihood function is to multiply the probabilities predicted by our model, where we replace a in the logistic functio above with the linear predictor $X\beta$. Seeing that we have one quantitative and one qualitative (3 levels) predictor in our data, the link function will look something like $\beta_0 + \beta_1 temperature + \beta_2 dummy1_{polymer} + \beta_3 dummy2_{polymer}$.

A different way to think about the purpose of the logistic function is that it maps an unconstrained predictor that can take values in $(-\infty, \infty)$ to a probability in $[0, 1]$.

Try to use the likelihood function given in `negloglikGLM.m` to fit the logistic regression model. You can use very similar code to what we used in lab 6 for fitting the model (i.e. for finding the maximum likelihood estimates of the parameters). Ask for help if you get stuck!

```
% we start by defining our variables:
response = A.data(:,1);
temperature = A.data(:,2);
% need 2 dummy variables for polymer type. Level 0 is absorbed into the intercept.
dummy1 = zeros(length(response),1);
dummy2 = zeros(length(response),1);
dummy1(find(A.data(:,3)==1)) = 1;
dummy2(find(A.data(:,3)==2)) = 1;


% Now define values for parameters that we use at the start of the optimisation.
% We have 4 parameters:
x0 = [0 0 0 0];

% check that computing the negative log-likelihood
% usning 'negloglikGLM.m' works:
negloglikGLM([1 2 0 1],1,2,0,0) % uses beta=[1 20 0 1].
```

```
ans = 0.0067
```

```
% response=1, temperature=20, dummy1=0, dummy=1

% maximum likelihood fit:
fun = @(s)negloglikGLM(s,response',temperature',...
    dummy1',dummy2');
options = optimset('MaxFunEval',10000000,'MaxIter',10000);
s = fminsearch(fun,x0,options); % here we performe the optimisation
% (i.e. find parameters for which negloglik is minimal)

% The fitted parameters are stored in 's':
disp(s)
```

```
   -5.5517    0.0935    4.7297   -2.2434
```

You should find parametes $(\beta_0, \beta_1, \beta_2, \beta_3) \approx (-5.55, 0.09, 4.73, -2.24)$, suggesting that increasing temperatures increase the size of the linear predictor ($\beta_1 > 0$) which means that the probability of success also increases with temperature (to see this, look at what happens when a in the logistic function, $\gamma(a) = \frac{1}{1+e^{-a}}$, increases).

Fortunately Matlab does all the hard work for us when it comes to fitting GLMs... .

1.3. Fitting a logistic regression model in Matlab

Fitting GLMs in Matlab works in a similar way to fitting LMs. Use the function `fitglm` and specifiy the linear predictor as usual (e.g. `y~x+w`). In addition, it is important to tell Matlab to use the correct exponential family distributio n and link function (if you are not using the default link function for a given exponential family distribution). For logistic regression, the command you need is: `fitglm(data,'y~x+w','Distribution','binomial')`. Remember to tell Matlab if a variable (response or predictor) is qualitative.

```
% create a table for the data:
response = nominal(response);
polymer = A.data(:,3);
polymer = nominal(polymer);
data = table(response,temperature,polymer,...
    'VariableNames',{'success','temperature','polymer'});
data(1:5,:)
```

```
ans =
    success    temperature    polymer
    _______    ___________    _______

    0          43             0
    0          46             0
    0          27             0
    0          33             0
    0          47             0
```

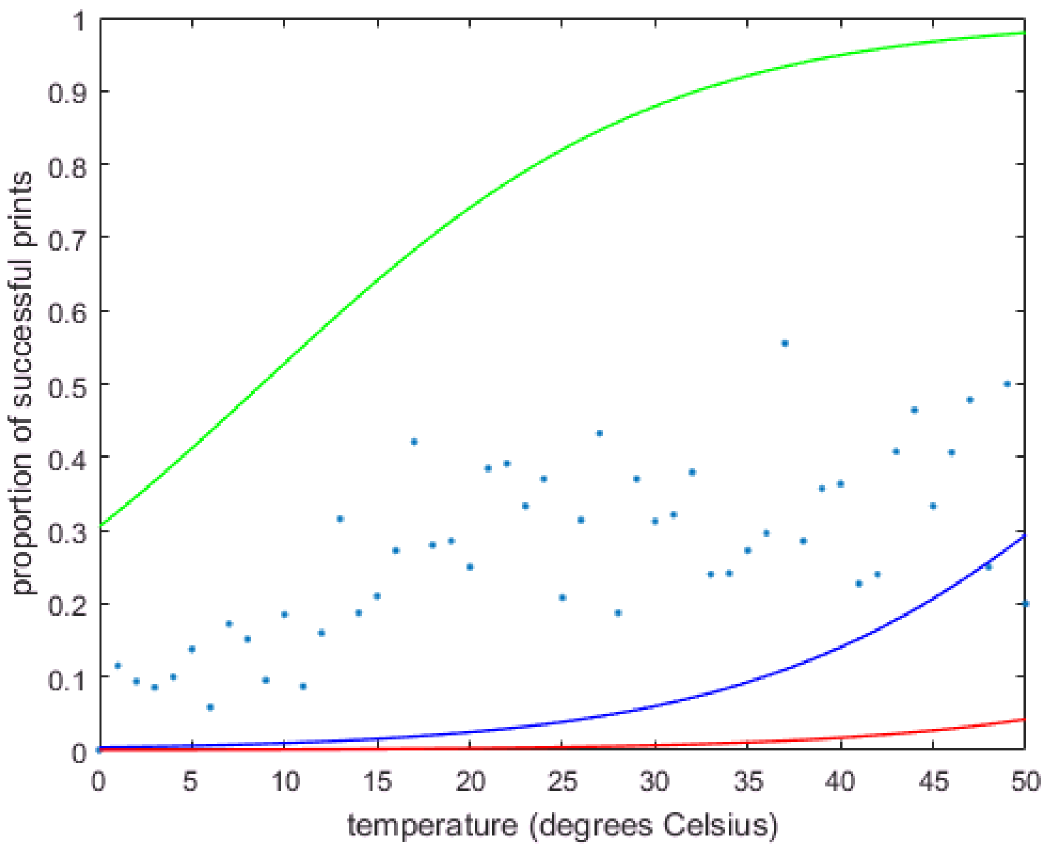
```
m1 = fitglm(data,'success~temperature+polymer',...
    'Distribution','binomial')
```

```
m1 =

Generalized Linear regression model:
    logit(P(success='1')) ~ 1 + temperature + polymer
    Distribution = Binomial

Estimated Coefficients:
                   Estimate       SE         tStat       pValue
                   ________    _________    _______    __________

    (Intercept)     -5.5517       0.4034    -13.762    4.3122e-43
    temperature    0.093534    0.0094093     9.9406    2.7709e-23
    polymer_1        4.7297      0.29997     15.767    5.2562e-56
    polymer_2       -2.2434      0.54204    -4.1389    3.4903e-05


1301 observations, 1297 error degrees of freedom
Dispersion: 1
Chi^2-statistic vs. constant model: 924, p-value = 6.61e-200
```

You should find the same parameter estimates as in section 1.2. In addition, Matlab performs parameter-specific hypothesis tests and a Likelihood-ratio test comparing the full model to a constant model that only includes in intercept.

Because of the link function, we cannot read off the effect different variables have from the parameter estimates directly. Often it is a good idea to plot the fit of the model on data. The code below suggests one way for doing this:

```
% setting up the service duration interval
% we want to plot model fit lines for:
w = linspace(min(temperature),max(temperature));

% plot the data and fit lines:
clf
plot(tmp,fraction,'.')
xlabel('temperature (degrees Celsius)')
ylabel('proportion of successful prints')
line(w,feval(m1,w,'0'),'Color','b','LineWidth',1)
line(w,feval(m1,w,'1'),'Color','g','LineWidth',1)
line(w,feval(m1,w,'2'),'Color','r','LineWidth',1)
```



The model fit lines for the three different polymer grades differ substantially and it is clear to see that averaging the probability of success over polymer grades as we have done when producing the scatterplot is not appropriate.

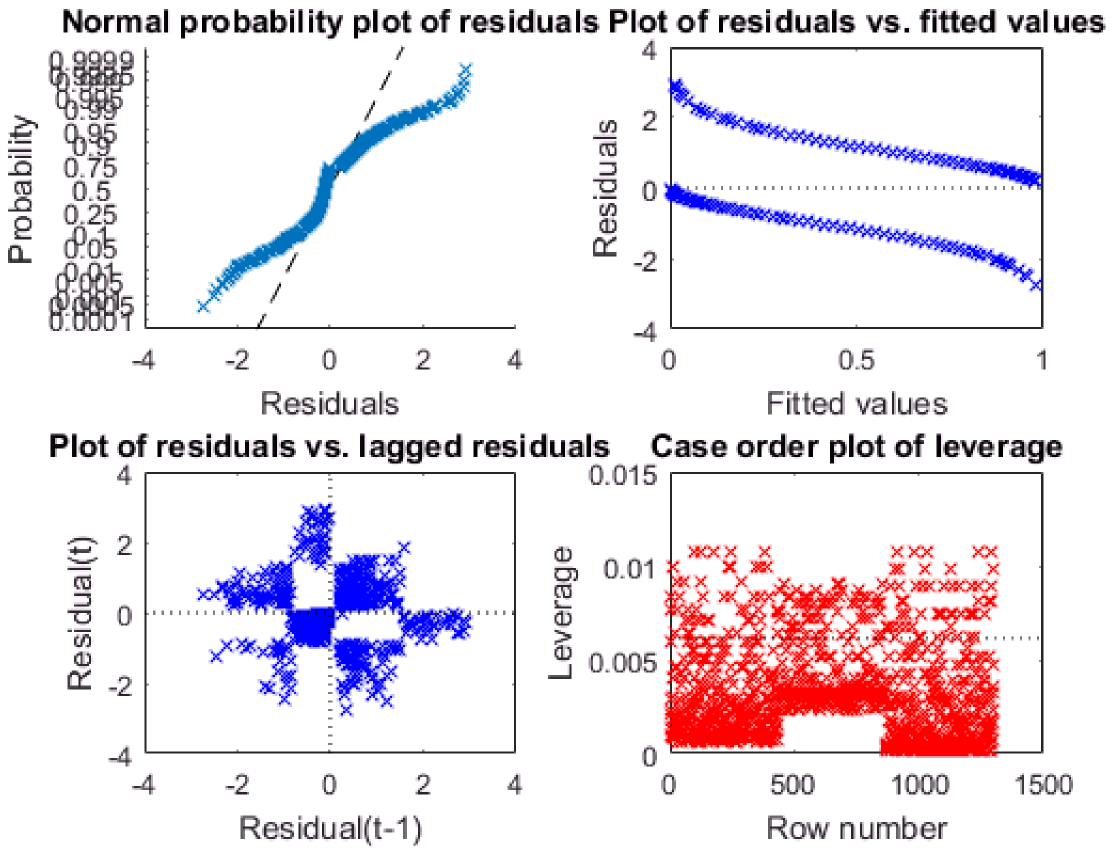
You can use the function `predict` in Matlab to obtain confidence bounds for model fits or predictions.

1.4. Model checking and model selection

Model checking and model selection for GLMs works in a similar way as for LMs.

For example, you can look at residual plots (but remember that you cannot necessarily use them in the same way! see lecture 10):

```
% residual plots
% Q-Q plot to check normality
subplot(2,2,1)
plotResiduals(m1,'probability','ResidualType','Deviance')
% residuals versus fitted values
subplot(2,2,2)
plotResiduals(m1,'fitted','ResidualType','Deviance')
% auto-correlation (via lagged residuals)
subplot(2,2,3)
plotResiduals(m1,'lagged','ResidualType','Deviance')
% leverage plot
subplot(2,2,4)
plotDiagnostics(m1)
```



... these plots might look a bit funny showing almost discrete patterns. This is not too surprising considering that the response is binary.

As a quick example for model selection methods, try to reproduce the Likelihood-ratio test in the summary table of the model you fitted above.

```
% fit a constant model to the data:
m0 = fitglm(data,'success~1',...
    'Distribution','binomial')
```

```
m0 =
Generalized Linear regression model:
    logit(P(success='1')) ~ 1
    Distribution = Binomial

Estimated Coefficients:
                   Estimate       SE        tStat       pValue  
                   ________    ________    _______    __________

    (Intercept)    -0.95317    0.061866    -15.407    1.4677e-53


1301 observations, 1300 error degrees of freedom
Dispersion: 1
```

```
[h pvalue stat] = lratiotest(m1.LogLikelihood,m0.LogLikelihood,3)
```

```
h =
     1

pvalue = 0
stat = 923.6352
```

(2) Poisson reggression

In this part of the lab we will work through an example for poisson regression. We will only fit a model using the read-made model fittting functions in Matlab.

2.1. Data for poisson regression

As discussed in lectures, GLMs using the poisson distribution are appropriate for data where the response represents counts.

We will look at the example we saw plotted in lecture 10 - the number of damage incidents in ships of different types. The file `ship_damage.txt` contains six columns. We will only use the second, fifth and sixth column here. The second column records the ship type (qualitative, 5 levels A-E), the fifth column contains the number of months a ship has been in service and the sixth colum, the response, records the count of damade incidents over the period the ship was in service.

The data is from:

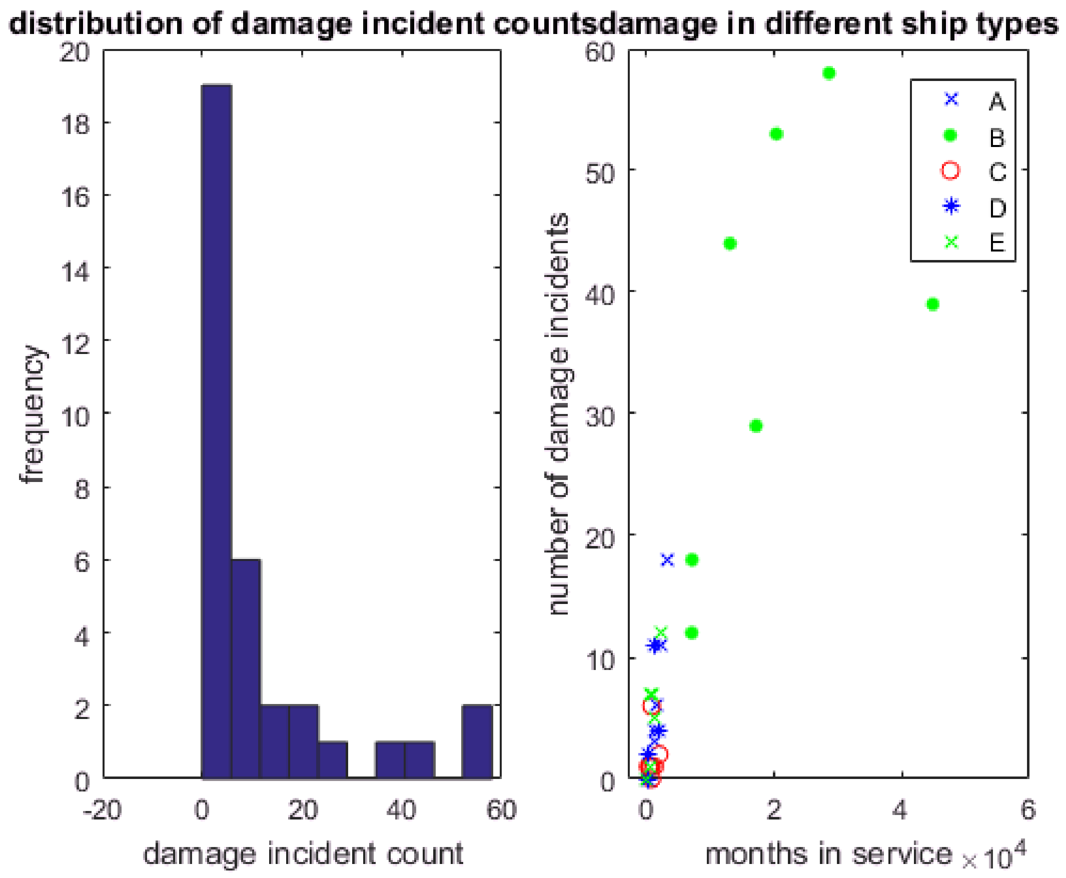
[McCullagh, P., and J. A. Nelder. 1989. *Generalized Linear Models*, 2nd ed. London: Chapman & Hall]

Your tasks is to investigate whether there is a connection between ship type and the number of damage incidents, whilst accounting for service duration of ships (i.e. service duration is also a predictor in the model).

Start by reading the data into Matlab and plotting it. The command `gscatter` is useful for producing a nice plot of the data.

```
%% read data:
delimiterIn = ' ';
A = importdata('ship_damage.txt',delimiterIn);
service = A.data(:,1);
damages = A.data(:,2);
```

```
type = A.textdata(2:35,2);
type = nominal(type);

clf
subplot(1,2,1)
hist(damages)
title('distribution of damage incident counts')
xlabel('damage incident count')
ylabel('frequency')
subplot(1,2,2)
gscatter(A.data(:,1),A.data(:,2),A.textdata(2:35,2),'bgr','x.o*')
xlabel('months in service'); ylabel('number of damage incidents');
title('damage in different ship types')
```



The scatterplot should show some evidence that ship type B serves for longer, but also suffers more damage incidents. Let's investigate this using a GLM.

2.2. Fitting a poisson regression model in Matlab

Now fit a GLM with Poisson distribution to the model. Look at the Matlab help pages to find out what link function is being used. Does ship type B lead to a higher number of damage incidents?

```
data = table(damages,service,type,'VariableNames',...
    {'damage_count','service_duration','ship_type'});
data(1:5,:)
```

```
ans =
    damage_count    service_duration    ship_type
    ____________    ________________    _________

    0                127                A
    0                 63                A
    3               1095                A
    4               1095                A
    6               1512                A
```

```
m1 = fitglm(data,'damage_count~service_duration+ship_type',...
    'Distribution','Poisson')
```

```
m1 =

Generalized Linear regression model:
    log(damage_count) ~ 1 + service_duration + ship_type
    Distribution = Poisson

Estimated Coefficients:
                         Estimate         SE         tStat        pValue
                        __________    __________    ________    __________

    (Intercept)              1.765       0.15443      11.429    3.0078e-30
    service_duration    1.9595e-05    4.6115e-06      4.2492    2.1454e-05
    ship_type_B             1.4035       0.19442       7.219    5.2387e-13
    ship_type_C            -1.2434       0.32733     -3.7984    0.00014561
    ship_type_D            -0.8902       0.28748     -3.0966     0.0019578
    ship_type_E           -0.10784       0.23466    -0.45957       0.64583


34 observations, 28 error degrees of freedom
Dispersion: 1
Chi^2-statistic vs. constant model: 461, p-value = 2.14e-97
```

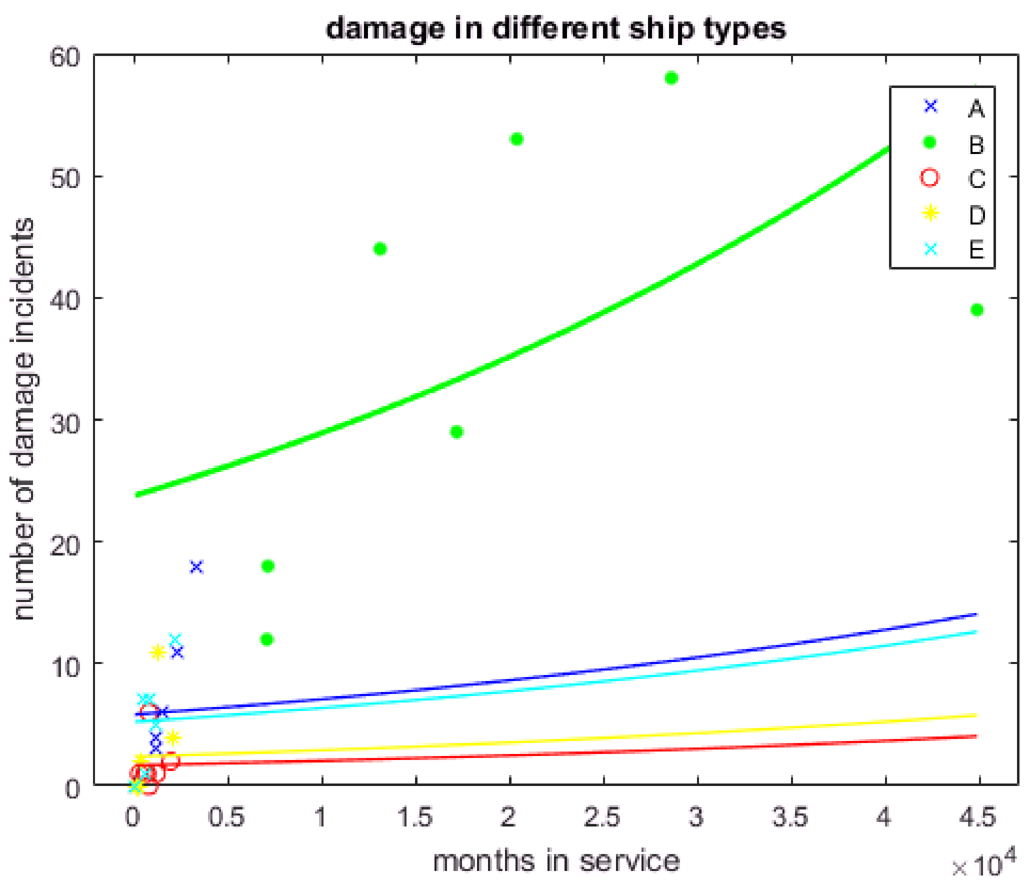
```
% The link function is 'log'.
% This means that we take the exponential of the linear
% predictor to obtain the mean of the response.
% The parameter estimate for type B is positive,
% suggesting an increase in mean damage incidents for
% this ship type.
```

Notice that because of the link function we cannot read off the effect of variables straight from parameter estimates. Matlab contains a number of inbuilt routines we can use to obtain predictions or best fit lines from GLMs (e.g. using `feval`, see below).

Here's an example for how you can plot the fit lines for different ship types:

```
% setting up the service duration interval
% we want to plot model fit lines for:
w = linspace(min(service),max(service));

% plot the data and fit lines:
clf
gscatter(service,damages,A.textdata(2:35,2),'bgryc','x.o*')
xlabel('months in service'); ylabel('number of damage incidents');
title('damage in different ship types')
```

```
line(w,feval(m1,w,'A'),'Color','b','LineWidth',1)
line(w,feval(m1,w,'B'),'Color','g','LineWidth',2)
line(w,feval(m1,w,'C'),'Color','r','LineWidth',1)
line(w,feval(m1,w,'D'),'Color','y','LineWidth',1)
line(w,feval(m1,w,'E'),'Color','c','LineWidth',1)
```

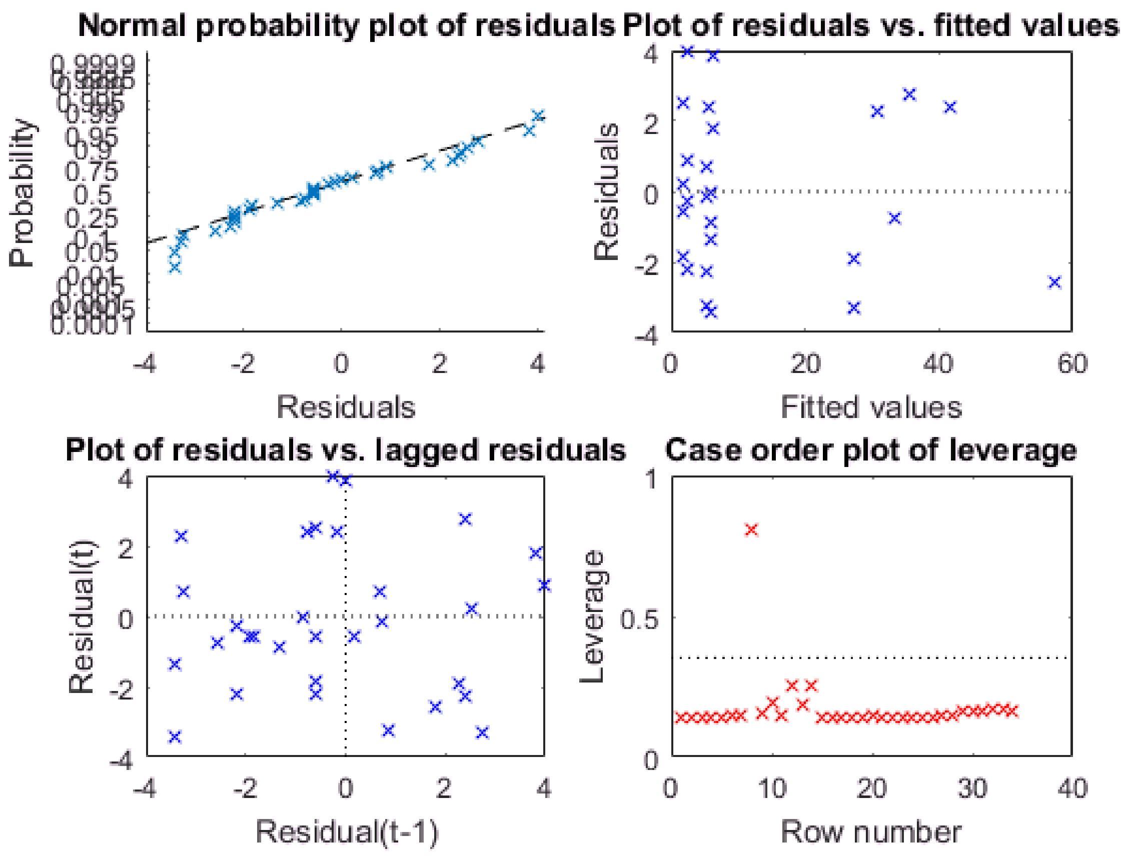


You can use the function `predict` if you want to find confidence bounds for model fits/predictions as well.

Notice how all ship types other than type B cover a much smaller range if service durations. This is an issue with the data and we should take it into consideration when interpreting the model findings.

Have a look at the residual plot.

```
% residual plots
% Q-Q plot to check normality
subplot(2,2,1)
plotResiduals(m1,'probability','ResidualType','Deviance')
% residuals versus fitted values
subplot(2,2,2)
plotResiduals(m1,'fitted','ResidualType','Deviance')
% auto-correlation (via lagged residuals)
subplot(2,2,3)
plotResiduals(m1,'lagged','ResidualType','Deviance')
% leverage plot
subplot(2,2,4)
plotDiagnostics(m1)
```



These look pretty re-assuring. Notice that when the Poisson distribution has a fairly high mean, it can be approximated with the normal distribution which means looking at the normal probability plot of the residuals is a sensible idea. However, there appears to be one data point that stands out from the rest, which may be a cause for concern (see leverage plot).