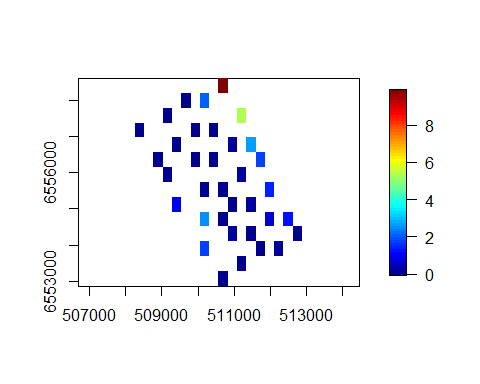
Fall of Warness

Lindesay Scott-Hayward

26 August 2016

# Real Data: Fall of Warness

require(fields)  
quilt.plot(dat$x.pos, dat$y.pos, dat$response, nrow=18, ncol=14, asp=1)



Plot of the data with average counts per grid cell.

### Fit Initial Model

init\_glm<-glm(response ~ as.factor(TideState) + WindStrength + SeaState + SimpPrecipitation + CloudCover + Depth + offset(log(areatime)), data=dat, family=quasipoisson)  
  
init\_glm<-make.gamMRSea(init\_glm, panelid=dat$newbidNum)

### Runs Test Check

Data generated under the null hypothesis of the runs test; independence and with no change induced at this stage.

nsim<-500  
d<-as.numeric(summary(init\_glm)$dispersion)  
newdat<-generateNoise(nsim, fitted(init\_glm), family='poisson', d=d)

500 sets of noisy data are simulated from the model using an overdispersed Poisson distribution where $ = $ 23.

empdistribution<-getRunsCritVals(n.sim = nsim, simData=newdat,   
 model = init\_glm, data = dat, plot=TRUE,   
 returnDist = TRUE, dots=FALSE)

Figure showing the distribution of test statistics from a runs test. The red lines show the lower 2.5% and upper 97.5% critical values of the empirical distribution and the blue lines are from the Normal () distribution.

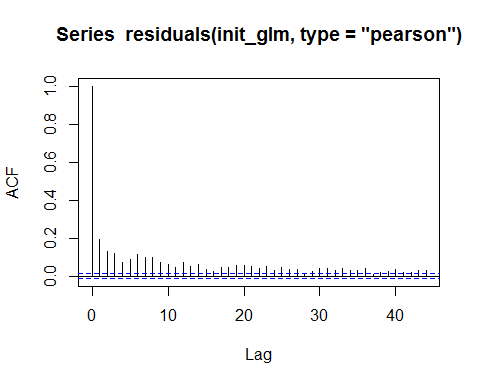
Figure showing the distribution of test statistics from a runs test. The red lines show the lower 2.5% and upper 97.5% critical values of the empirical distribution and the blue lines are from the Normal () distribution.

Evaluate the runs test using the empirical distribution to determine if the data are independent

runs.test(residuals(init\_glm, type='pearson'), critvals = empdistribution)

Runs Test - Two sided; Empirical Distribution  
  
data: residuals(init\_glm, type = "pearson")  
Standardized Runs Statistic = -88.903, p-value = 0.06

acf(residuals(init\_glm, type='pearson'))



The residuals are not considered independent so for the power analysis, the data will be generated as correlated.

### Data Generation

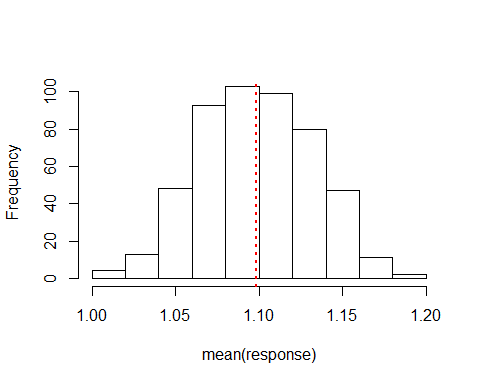
Data are generated with no change but with the same dispersion and correlation as the original data so that the data generation process can be checked.

corrs<-getCorrelationMat(dat$newbidNum, dat$response, dots=FALSE)  
newdatcor<-generateIC(data = dat, corrs = corrs, panels = 'blockid', newdata = newdat, nsim = nsim, dots = FALSE)

Are the generated data consistent with the original data?

#### Mean

hist(apply(newdatcor, 2, mean), main='', xlab='mean(response)')  
abline(v=mean(dat$response), col='red', lwd=2, lty=3)

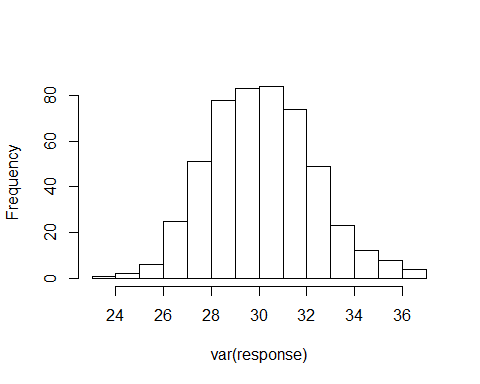


Histogram of the mean of the simulated data, with the red line representing the mean of the original data.

In this case the mean of the original data sits inside the distribution of simulated means.

#### Variance

hist(apply(newdatcor, 2, var), main='', xlab='var(response)')  
abline(v=var(dat$response), col='red', lwd=2, lty=3)



Histogram of the variance of the simulated data, with the red line representing the variance of the original data.

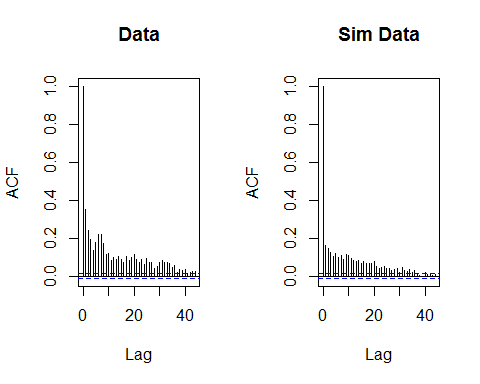
var(dat$response)

[1] 76.3464

Unfortunately the variance of the simulated data is lower than that of the original data ($^2 = $ 76.346). This could be due to the overly simplistic model being unable to return the variability seen in the data. This model is then used to simulate from.

#### Correlation

par(mfrow=c(1,2))  
acf(dat$response, main='Data')  
acf(newdatcor[,50], main='Sim Data')



The correlation seen in the data matches well with the example shown from the simulated data. This indicates the IC method is working well.

#### Check the dispersion parameter

fowsim\_glm<-update(init\_glm, newdatcor[,1] ~ .)  
summary(fowsim\_glm)$dispersion

[1] 25.40465

#### Data spatial distribution

require(fields)  
par(mfrow=c(2,2))  
quilt.plot(dat$x.pos, dat$y.pos, dat$response, main='Original Data', nrow=18, ncol=14, asp=1)  
quilt.plot(dat$x.pos, dat$y.pos, fitted(init\_glm), main='Fitted Values', nrow=18, ncol=14, asp=1)  
quilt.plot(dat$x.pos, dat$y.pos, newdatcor[,1], main='Simulated Data A', nrow=18, ncol=14, asp=1)  
quilt.plot(dat$x.pos, dat$y.pos, newdatcor[,10], main='Simulated Data B', nrow=18, ncol=14, asp=1)

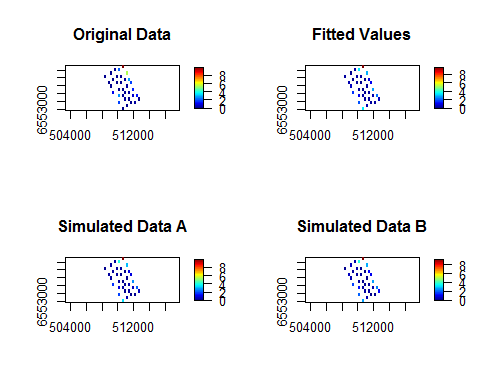


Figure showing the spatial distribution of the original data, the fitted values from the model and two examples of the simulated data.

These plots show that the spatial distribution of the data is well represented by the simulated data.

### Power Analysis

Now that we know we can generate data that represents the original fairly well, we now generate data for the power analysis with a change imposed of **25%**; a 25% decline in animal numbers post event.

First the change is imposed on the data, then the noise is added (in-line with the dispersion seen in the initial model) and finally the correlation.

truebeta<-log(0.9)  
nsim=500  
impdata<-genOverallchangeData(truebeta, init\_glm, data = dat, panels = 'newbidNum')  
newdat<-generateNoise(nsim, impdata$truth, family='poisson', d=d)  
newdatcor<-generateIC(data = impdata, corrs = rbind(corrs, corrs), panels = 'panels', newdata = newdat, nsim = nsim, dots = FALSE)

Update the initial model to include the eventphase term, indicating overall change. This model forms the foundation of the power analysis to follow.

fowsim\_glm<-update(fowsim\_glm, .~. + eventphase, data=impdata)  
fowsim\_glm<-make.gamMRSea(fowsim\_glm, panelid=impdata$panels)

The new values for the empirical distribution of the runs test are estimated using the noisy but un-correlated simulated data. This is differenct to the distribution generated earlier as we have another term in the model (eventphase).

# make sure that the independent data is used to get the null distribution  
empdistpower<-getRunsCritVals(n.sim = nsim, simData=newdat,   
 model = fowsim\_glm, data = impdata, plot=TRUE,   
 returnDist = TRUE, dots=FALSE)

If an offset is present in the model, the area of each gridcell should be provided when setting up the prediction grid so that an accurate estimate of the abundance pre and post event can be made at the outputs stage.

data("fowshco.grid")  
predictdata<-rbind(data.frame(fowshco.grid, TideState=1, WindStrength=0, SeaState=0, SimpPrecipitation='SHOWERS', CloudCover=0, areatime=fowshco.grid$Area, eventphase=0), data.frame(fowshco.grid, TideState=1, WindStrength=0, SeaState=0, SimpPrecipitation='SHOWERS', CloudCover=0,areatime=fowshco.grid$Area, eventphase=1))

nsim=100  
powerout<-powerSimOverallChange(newdatcor, fowsim\_glm, empdistpower, nsim=nsim, powercoefid=length(coef(fowsim\_glm)), predictionGrid=predictdata, g2k=NULL, splineParams=NULL, sigdif=TRUE, n.boot=500, impact.loc=c(510700, 6555700))

# Power Outputs:

To calculate the summary, the values for certain parameters are given under the null hypothesis of 'no change'. If these parameters are not provided, then the summary will not show the comparisons (see below)

### Summary output

summary(powerout, truebeta=truebeta)

++++ Summary of Power Analysis Output ++++  
  
Number of power simulations = 100  
Number of no change simulations = NA   
  
Power to select 'change' term:  
  
 Under Change (true parameter = -0.1053605) = 73%  
 Under no change = Null distribution not specified  
  
Coverage for 'change' coefficient:  
  
 Under change = 97%  
 Under no change = Null distribution not specified  
  
Overall Abundance Summary with 95% Confidence Intervals:  
  
 Abundance LowerCI UpperCI  
Before 5715 4519 6909  
After 5143 3998 6083  
  
Note: These calculations assume the correct area has been given for each prediction grid cell.

In order to assess the summary calculations, we estimate the null parameters to provide the full summary.

null.output<-pval.coverage.null(newdat.ind = newdat, newdat.corr = newdatcor, model = fowsim\_glm, nsim = 500, powercoefid = length(coef(fowsim\_glm)))

summary(powerout, null.output, truebeta=truebeta)

++++ Summary of Power Analysis Output ++++  
  
Number of power simulations = 100  
Number of no change simulations = 250   
  
Power to select 'change' term:  
  
 Under Change (true parameter = -0.1053605) = 73%  
 Under no change (true parameter = 0) = 6.4%  
  
Coverage for 'change' coefficient:  
  
 Under model = 97%  
 Under no change = 92.4%  
  
Overall Abundance Summary with 95% Confidence Intervals:  
  
 Abundance LowerCI UpperCI  
Before 5715 4519 6909  
After 5143 3998 6083  
  
Note: These calculations assume the correct area has been given for each prediction grid cell.

We expect to see the null power to be 5%, indicating a 5% error rate for inclusion of the eventphase term when the data do not contain any change. We also expect that the coverage for the null coefficient is 95%. This indicates that 5% of the time, the confidence intervals for the eventphase coefficient did not include the truth (0; no change).

### How does power change with the error rate?

powerPlot(powerout)

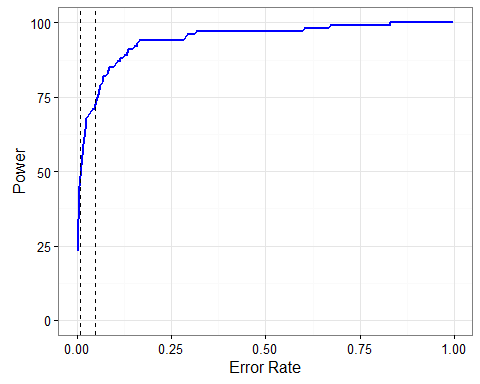


Figure showing how the power changes if the error rate is changed. The dotted lines represent error rates of 0.01 and 0.05.

### Proportion of significant differences:

plotdata<-plot.sigdiff(powerout, predictdata[predictdata$eventphase==0,c('x.pos', 'y.pos')], tailed='two', error.rate = 0.05, family=FALSE)  
plotdata

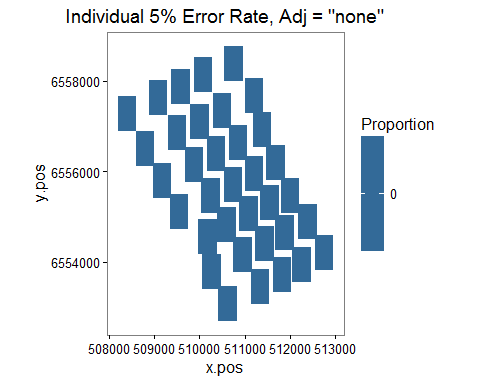


Figure showing the proportion of simulations which showed a significant difference (post event - pre event) in a given grid cell. Significance at the 5% level was determined by comparing the estimated difference in a given cell to the distribution of differences in that cell under the null hypothesis of no-change.

plotdata<-plot.sigdiff(powerout, predictdata[predictdata$eventphase==0,c('x.pos', 'y.pos')], tailed='two', error.rate = 0.05, family=TRUE)  
plotdata

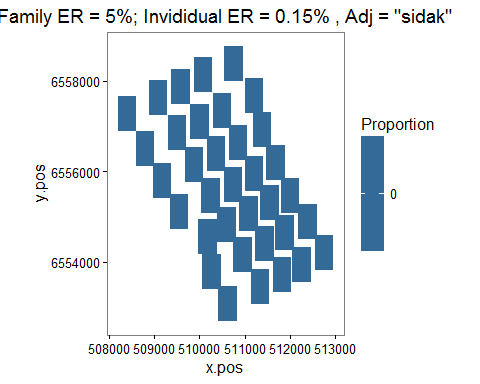


Figure showing the proportion of simulations which showed a significant difference (post event - pre event) in a given grid cell. Significance at the family 5% level was determined by using the Sidak adjustment.

### Distance to event site plot

d2impPlot<-plot.d2imp(powerout, pct.diff=FALSE)  
d2impPlot

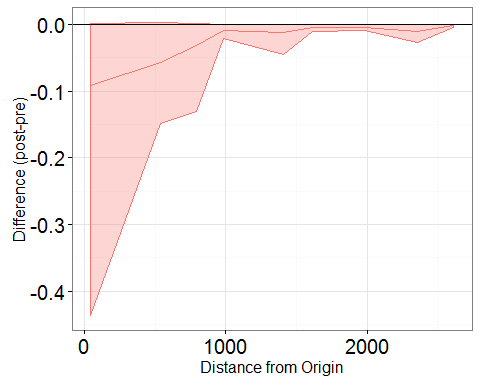


Figure showing the effect of the event on animal numbers with distance from the event site; post event - pre event. The confidence intervals are 95% bootstrap intervals.

d2impPlot<-plot.d2imp(powerout, pct.diff = TRUE)  
d2impPlot

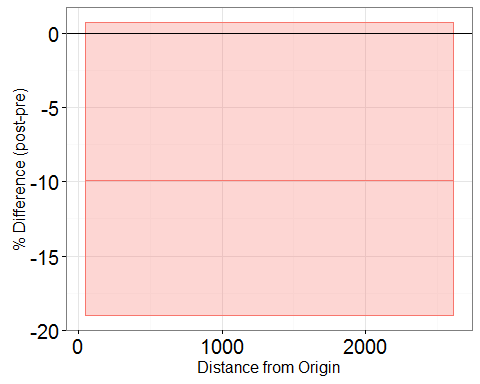


Figure showing the effect of the event on animal numbers with distance from the event site; post event - pre event. The confidence intervals are 95% bootstrap intervals.