Investigation of potential influences to Great Crested Newt (*Triturus cristatus*) presence in ponds, and Metadata Analysis of *Actinopterygii* and *Amphibia* species cohabitation via eDNA samples (using Rstudio)

Student Number: 100204382. School of Biological Sciences BIO-6019Y.

Due Date: 22 May 2020

Contents

---

1 Abstract

The use of environmental DNA (eDNA) as a rapid survey technique for rare or hard to survey freshwater organisms has been increasing in recent years. For species such as the great crested newt (*Triturus cristatus*) in the UK, site developers are required to protect any species that may come to harm through means such as biodiversity offsetting, encouraging the identification of land nearby that contains most biodiversity for least money to conserve. Many of these schemes unfortunately have a 'no net loss' policy that can often be interpreted as maintaining the current rate of decline, doing nothing to help the species survive beyond expected measures. An alternative approach may be the District Liscencing scheme, using a 4:1 ratio of new pond habitats built to pond habitats destroyed ensuring a net gain in *T. cristatus*. Finding new habitats for this scheme is accomplished in part by implementing The Great Crested Newt Habitat Suitability Index (HSI), a system for determining the suitability of a pond in sustaining *T. cristatus* based on ten different environmental factors.

This study tests the influence from the variables: HSI, Nitrate level, Phosphate level, and Cleanliness (combined Nitrate and Phosphate levels), on the presence of *T. cristatus* in ponds surveyed across the UK. This was achieved by analysing data collected from water samples checked for environmental DNA (eDNA) of vertebrate species via high-throughput sequencing (HTS). Data was analysed using Rstudio, learnt throughout the process of this study, with tests being performed including Chi-square (X^2) test for goodness of fit and binomial logistical regression. In addition to this, a Non-metric Multi-Dimensional Scaling (NMDS) ordination, using Kruskal's Stress formula was performed to map out the *Actinopterygii* and *Amphibia* species identified, plotted in accordance to their environmental influences and by extension their ability to cohabitate.

Evidence has been given to support the claim that HSI value impacts the presence of *T. cristatus*, effectively allowing rejection of the null hypothesis. The output of the NMDS have been discussed along with the limitations this study pose, and alterations for future studies have been suggested.

2 Introduction.

Environmental DNA (eDNA) metabarcoding is the detection of multiple species using DNA collected from environments such as ponds, soil, or air, consisting of samples such as faeces, mucus, skin cells, and extracellular DNA (Deiner, *et al.* 2017). Samples of eDNA are simultaneously analysed via high-throughput sequencing (HTS) after PCR amplification and are categorised by taxonomic identification. This is determined via sequence similarity between selected sequences of collected samples with alignment programs such as BLAST, that use the NCBI nucleotide database (or the Barcode of Life Database) and phylogenetic reconstruction (Piper *et al.* 2019).

Identifying what species may exist in any given area is important for protecting biodiversity (Maron, *et al.* 2015). With the current changes in Earth's climate, along with its projected changes for the near future, there are growing concerns for the keystone species for ecosystems to function appropriately, as their habitats change and they are being driven to new locations for survival (in turn affecting other ecosystems) (Walther, *et al.* 2002), and eventually driven to extinction (Harte, *et al.* 2004).

One way to protect species is biodiversity offsetting (Needham, *et al.* 2019), which enforces companies to compensate for habitat damage by developing and managing other habitats for the same disrupted species elsewhere nearby, encouraging the identification of land nearby that contains most biodiversity for least money to conserve. This can mean any number of things including restoring a pre-existing habitat by planting trees or removing threats to a species by giving it a protected status (Maron, *et al.* 2015). Unfortunately, most schemes have a 'no net loss' policy that can often be interpreted as maintaining the current rate of decline, which does nothing to help the species survive beyond expected measures (Robertson. 2000; Quétier, *et al.* 2014).

An example of a protected species throughout England is *Triturus cristatus* (*T. cristatus*), or Great Crested Newts (GCN), but some of the sites they inhabit are also needed for building development and as such workers must wait for proper procedures to be completed before going forward with their construction. These procedures include hiring an ecologist to perform surveys to determine if there is a presence *T. cristatus* eDNA (Deiner, *et al.* 2017), and depending on how many ponds to be tested, may result in high costs (averaging at £250K), especially when there are delays, which site developers are charged for regardless of whether they can build on that site or not (Tew, *et al.* 2018).

With the introduction of District Licencing, building commissions have become a lot easier to apply for by focusing on the conservation of *T. cristatus* populations rather than individuals, using a 4:1 ratio of new pond habitats built to pond habitats destroyed (Tew, *et al.* 2019), ensuring a net gain in *T. cristatus* habitats, as opposed to the common interpretation of 'no net loss'.

Finding new habitats for this scheme is accomplished in part by implementing The Great Crested Newt Habitat Suitability Index (HSI), a system for determining the suitability of a pond in sustaining *T. cristatus*. First developed by US Fish and Wildlife Service (Oldham, *et al.* 2000), HSI compares ten different environmental influences to evaluate any ponds presented in a proposed mitigation scheme and can assist in identifying priorities for habitat management (ARG. 2010), however, there may be other influences on the newts' presence.

Many crop plants require large quantities of nitrogen to produce high yields (Guarda, *et al.* 2004; Bhatty. 1964), so additional nitrogen in the form of fertiliser is applied (Lovatt. 2001). Unfortunately, nitrogen is extremely soluble and can leach into the groundwater, eventually finding its way into watercourses (McKague, *et al.* 2005). This causes a nutrient boost in the environment which can then alter the ecological balance (Camargo and Alonso. 2006). In addition, excess phosphate causes a nutrient boost that can often lead to excessive algae growth, producing toxins that adversely affect the ecosystem by reducing oxygen levels in the water, leading to loss of species and degradation of the waterway (Atkins. 1923; Fried, *et al.* 2003; Adesuyi, *et al.* 2015).

Null hypothesis (H_0): There is no significant influence on *T. cristatus* presence from any or all of the variables: HSI, Nitrate level, Phosphate level, and Cleanliness (combined Nitrate and Phosphate levels).

Alternative hypothesis (H_a): There is a significant influence on *T. cristatus* presence from any or all of the variables: HSI, Nitrate level, Phosphate level, and Cleanliness (combined Nitrate and Phosphate levels).

3 Methods

3.1 Before Beginning the Project

Pond samples were collected and sequenced following a similar method to Biggs, *et al* (2014):

Water samples were collected from ponds using sterile procedures and the use of a Whirl-Pak™ bag. Sample tubes contain 33mL of absolute ethanol and 1.5 ml of sodium acetate 3 M, acting as a DNA preservative. An approximate total of 600 mL of pool water samples were collected (being sure not to disturb the soil to introduce contaminant DNA) from 20 locations to be gently mixed together, allowing an equal spread of eDNA throughout the collected water. These samples were extracted of DNA which was tested for the presence of *T. cristatus* unique sequences via quantitative polymerase chain reaction (qPCR) and sequenced

with an Illumina MiSeq. The eDNA was then tested via metabarcoding to sequence and organise the DNA into groups via taxonomy.

This high-throughput sequence (HTS) data was processed with bioinformatic software testing sequence similarity between reference sequences of collected samples with a compiled library record. If the collected eDNA matches any species within the formulated library, this determines its presence in the sampled ponds. To avoid false positives, strict lab conditions were followed, and positive and negative controls were put in place for the PCR experiment. To avoid false negatives as best as possible, the pond water was collected from multiple (20) areas of the same pond and mixed, however, no guarantee can be made for the presence of eDNA of all species living in and around the pond area. This means that while efforts can be made to avoid false detection of a species, the absence of the species' eDNA is not proof that it does not reside there. In addition to these measures, the code written tests for detecting false positives and negatives in the data.

3.2 The Project.

Initially, basics such as making r blocks for code, installing library packages, setting working directories, and understanding pipelines were taught before handling the dataset. Functions such as `select()`, `filter()` and `mutate()` were then introduced along with principles of graphs such as `ggplot()`. After this introduction, writing the code for analysis begun by setting the working directory, and loading libraries needed (`Chunks 1 and 2`). The .csv containing the data was read and loaded into a data frame (`fhtwild`) in `Chunk 3` where relevant columns were kept and those deemed irrelevant were discarded. The data was then filtered in `Chunk 4` to removes metabarcoding replicates (and as a consequence any Internal Positive Controls) that had *NA* values and any samples with liquid in the bag to mitigate contamination risk. Precalculated HSI values that had *NA* and any collected HSI data that were non-numerical and non-conforming were also filtered out, along with data where *T. cristatus* presence, Phosphate and Nitrate levels, and categorised 'Clean' values were also reported to have an *NA* value.

Some samples were analysed more than once, shown as `IDcount` in `Chunk 4`, and were removed as ID numbers were grouped (`group_by(nm_Kit_ID)`) and counted (`summarise(count = n())`). False-positive and negative results within the data were also isolated in `Chunk 5` by comparing `Status` and `GCN_test_result` with `Amph_Caud_Sala_Tritcris`, where the latter shows detection of *T. cristatus* eDNA when over zero. False negatives were changed into positive status with `mutate()` while false positives were removed from the data set.

The data collected for HSI was converted to numeric values in `Chunk 6` as opposed to the string characters to allow mathematical calculations as the used model was different to the standard model described by the Great Crested Newt Habitat Suitability Index (ARG. 2010) which meant that standard binning was not applicable, and so more binning options were made to factor for this. In HSI 2 (Pond Area) *NA* appears where the area is larger than 2000 m^2, which is omitted from HSI calculation, while in HSI 8 (Pond Count) -Infinity occurred when HSI 8 equalled zero, so given a value of 0.1 instead.

These numbers were then used to calculate the harmonic mean for the HSI value using the equation below (ARG.2010):

$$HSI = (SI_1 * SI_2 * SI_3 * SI_4 * SI_5 * SI_6 * SI_7 * SI_8 * SI_9 * SI_{10})^{1/10}$$

Or when Pond Area is larger than 2000 m^2 (ARG.2010):

$$HSI = (SI_1 * SI_3 * SI_4 * SI_5 * SI_6 * SI_7 * SI_8 * SI_9 * SI_{10})^{1/9}$$

Using these values, a boxplot was made with `ggplot`, `geom_boxplot()`, and `geom_jitter()` to compare *T. cristatus* presence and HSI value in `Chunk 8`, while `Chunk 9` presented Cleanliness concerning Nitrate and Phosphate levels for all samples and those of just Positive results.

A `scatterplotMatrix()` was made in `Chunk 10` to compare HSI value, Nitrate, Phosphate, and Cleanliness with *T. cristatus* presence (`GCN_Status_binary`) to check for any obvious collinearity, which was then

statistically tested for using generalised linear models (`glm()`) as *T. cristatus* presence was converted to binominal values and a Chi-square (X^2) test for goodness of fit was performed.

The equation for X^2 for the goodness of fit is:

$$\chi^2 = \sum \left[\frac{(Oi - Ei)^2}{Ei} \right]$$

Where,χ^2 = Chi-Square goodness of fit test O_i is the observed frequency count for the ith level of the categorical variable, and E_i is the expected frequency count for the ith level of the categorical variable (Howell. 2011).

Any significant variable is then plotted, marking each sample and mapping the probability of *T. cristatus* presence dependent upon the variable(s) predicted value using the equation:

$$y = \{^{1=\beta_0+\beta_1 x+\varepsilon>0}_{0=else}$$

Where β_0 is the intercept, $\beta_1 x$ is the regression coefficient multiplied by a value of the predictor, and ε indicates an exponential function (Allison and Waterman. 2002; Allison. 1996)

In `Chunk 11` the data set was split into two, one for species present (`ID_species`), and another for environmental factors that had been recorded (`ID_Environment`), prepared for a Non-metric Multi-Dimensional Scaling (NMDS) ordination, using Kruskal's Stress formula:

$$Stress = \sqrt{\frac{\Sigma(d_{ij} - \delta_{ij})^2}{\Sigma(d_{ij})^2}}$$

Where Stress is the goodness of fit of the regression, d_{ij} is the ordinated distance between samples i and j, and δ is the distance predicted from the regression *(Holland. 2008)*.

As a result of the shepard stressplot, the *Actinopterygii* and *Amphibia* species identified (Appendix 1) could be plotted in accordance to their environmental influences (Appendix 2) and their ability to cohabitate.

---

4 Results

4.1 Chunk 1:- Setting Working Directory

A working directory was made for the event of loading after a separate project, however, was remained commented out during time working on scripts in case of interference.

```
options(tinytex.verbose = TRUE)

 # setwd("C:\Users\Tobias\Desktop\6019Y_GCN_Student-no_100204382\Code")
 # leave commented out (only necessary when going between projects)
```

4.2 Chunk 2:- Loading Libraries

Needed libraries were installed and loaded to allow the creation of certain graphs and to run tests needed.

```
options(tinytex.verbose = TRUE)

library(readr)

# script-specific libraries
library(sf)
```

```
## Linking to GEOS 3.6.1, GDAL 2.2.3, PROJ 4.9.3
```

```
library(raster)
```

```
## Loading required package: sp
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:raster':
##
##     intersect, select, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(spData)
```

```
## To access larger datasets in this package, install the spDataLarge
## package with: `install.packages('spDataLarge',
## repos='https://nowosad.github.io/drat/', type='source')`
```

```
library(stringdist)
library(vegan)
```

```
## Loading required package: permute

## Loading required package: lattice

## This is vegan 2.5-6
```

```
library(broom)
library(arm)
```

```
## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

## The following objects are masked from 'package:raster':
##
##     area, select

## Loading required package: Matrix

## Loading required package: lme4

##
## Attaching package: 'lme4'

## The following object is masked from 'package:raster':
##
##     getData

##
## arm (Version 1.10-1, built: 2018-4-12)

## Working directory is C:/Users/Tobias/Desktop/6019Y_GCN_Student-no_100204382/Code
```

```
library(ggeffects)
library(car)
```

```
## Loading required package: carData

## Registered S3 methods overwritten by 'car':
##   method                           from
##   influence.merMod                 lme4
##   cooks.distance.influence.merMod lme4
##   dfbeta.influence.merMod          lme4
##   dfbetas.influence.merMod         lme4

##
## Attaching package: 'car'

## The following object is masked from 'package:arm':
##
##     logit

## The following object is masked from 'package:dplyr':
##
##     recode
```

```
library(RColorBrewer)
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(zCompositions)
```

```
## Loading required package: NADA
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'NADA'
```

```
## The following object is masked from 'package:raster':
##
##     flip
```

```
## The following object is masked from 'package:stats':
##
##     cor
```

```
## Loading required package: truncnorm
```

```
# general-use packages
library(here)
```

```
## here() starts at C:/Users/Tobias/Desktop/6019Y_GCN_Student-no_100204382/Code
```

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.3.0     v purrr   0.3.4
## v tibble  3.0.0     v stringr 1.4.0
## v tidyr   1.0.2     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------------------------------------------ tid
## x tidyr::expand()  masks Matrix::expand()
## x tidyr::extract() masks raster::extract()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x tidyr::pack()    masks Matrix::pack()
## x car::recode()    masks dplyr::recode()
## x MASS::select()   masks dplyr::select(), raster::select()
## x purrr::some()    masks car::some()
## x tidyr::unpack()  masks Matrix::unpack()
```

```r
library(readxl)
library(cowplot)
```

```
##
## ************************************************************
```

```
## Note: As of version 1.0.0, cowplot does not change the
```

```
##   default ggplot2 theme anymore. To recover the previous
```

```
##   behavior, execute:
##   theme_set(theme_cowplot())
```

```
## ************************************************************
```

```
##
## Attaching package: 'cowplot'
```

```
## The following object is masked from 'package:ggeffects':
##
##     get_title
```

```r
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
##
## Attaching package: 'GGally'
```

```
## The following object is masked from 'package:dplyr':
##
##     nasa
```

```r
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:cowplot':
##
##     stamp
```

```
## The following objects are masked from 'package:dplyr':
##
##     intersect, setdiff, union
```

```
## The following objects are masked from 'package:raster':
##
##     intersect, union

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(arsenal) # for summary(comparedf())
```

```
##
## Attaching package: 'arsenal'

## The following object is masked from 'package:lubridate':
##
##     is.Date

## The following objects are masked from 'package:Matrix':
##
##     head, tail

## The following objects are masked from 'package:raster':
##
##     head, tail
```

```
library(sjmisc) # for rotate_df()
```

```
## Learn more about sjmisc with 'browseVignettes("sjmisc")'.

##
## Attaching package: 'sjmisc'

## The following object is masked from 'package:arsenal':
##
##     %nin%

## The following object is masked from 'package:purrr':
##
##     is_empty

## The following object is masked from 'package:tidyr':
##
##     replace_na

## The following object is masked from 'package:tibble':
##
##     add_case

## The following object is masked from 'package:raster':
##
##     trim
```

```
library(envDocument)
library(patchwork)
```

```
##
## Attaching package: 'patchwork'

## The following object is masked from 'package:cowplot':
##
##     align_plots

## The following object is masked from 'package:MASS':
##
##     area

## The following object is masked from 'package:raster':
##
##     area
```

```
library(sessioninfo)
library(tmap)    # for static and interactive maps
library(leaflet) # for interactive maps
library(mapview) # for interactive maps
library(ggplot2) # tidyverse data visualization package
library(shiny)   # for web applications
library(conflicted)
  conflict_prefer("mutate", "dplyr")
```

```
## [conflicted] Will prefer dplyr::mutate over any other package
```

```
  conflict_prefer("select", "dplyr")
```

```
## [conflicted] Will prefer dplyr::select over any other package
```

```
  conflict_prefer("summarise", "dplyr")
```

```
## [conflicted] Will prefer dplyr::summarise over any other package
```

```
  conflict_prefer("filter", "dplyr")
```

```
## [conflicted] Will prefer dplyr::filter over any other package
```

```
  conflict_prefer("first", "dplyr")
```

```
## [conflicted] Will prefer dplyr::first over any other package
```

```
  conflict_prefer("here", "here")
```

```
## [conflicted] Will prefer here::here over any other package
```

```
  conflict_prefer("separate", "tidyr")
```

```
## [conflicted] Will prefer tidyr::separate over any other package
```

```
  conflict_prefer("unite", "tidyr")
```

```
## [conflicted] Will prefer tidyr::unite over any other package
```

```
# Provide real numbers, not scientific notation.
options(scipen = 999)

# sessionInfo() # base R method
session_info()
```

```
## - Session info ---------------------------------------------------------------
##  setting  value
##  version  R version 3.6.3 (2020-02-29)
##  os       Windows 10 x64
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  English_United Kingdom.1252
##  ctype    English_United Kingdom.1252
##  tz       Europe/London
##  date     2020-05-20
##
## - Packages -------------------------------------------------------------------
##  package       * version  date       lib source
##  abind           1.4-5    2016-07-21 [1] CRAN (R 3.6.0)
##  arm           * 1.10-1   2018-04-13 [1] CRAN (R 3.6.3)
##  arsenal       * 3.4.0    2020-02-15 [1] CRAN (R 3.6.3)
##  assertthat      0.2.1    2019-03-21 [1] CRAN (R 3.6.3)
##  backports       1.1.6    2020-04-05 [1] CRAN (R 3.6.3)
##  base64enc       0.1-3    2015-07-28 [1] CRAN (R 3.6.0)
##  boot            1.3-24   2019-12-20 [2] CRAN (R 3.6.3)
##  broom         * 0.5.6    2020-04-20 [1] CRAN (R 3.6.3)
##  car           * 3.0-7    2020-03-11 [1] CRAN (R 3.6.3)
##  carData       * 3.0-3    2019-11-16 [1] CRAN (R 3.6.1)
##  cellranger      1.1.0    2016-07-27 [1] CRAN (R 3.6.3)
##  class           7.3-15   2019-01-01 [2] CRAN (R 3.6.3)
##  classInt        0.4-3    2020-04-07 [1] CRAN (R 3.6.3)
##  cli             2.0.2    2020-02-28 [1] CRAN (R 3.6.3)
##  cluster         2.1.0    2019-06-19 [2] CRAN (R 3.6.3)
##  coda            0.19-3   2019-07-05 [1] CRAN (R 3.6.3)
##  codetools       0.2-16   2018-12-24 [2] CRAN (R 3.6.3)
##  colorspace      1.4-1    2019-03-18 [1] CRAN (R 3.6.1)
##  conflicted    * 1.0.4    2019-06-21 [1] CRAN (R 3.6.3)
##  cowplot       * 1.0.0    2019-07-11 [1] CRAN (R 3.6.3)
##  crayon          1.3.4    2017-09-16 [1] CRAN (R 3.6.3)
##  crosstalk       1.1.0.1  2020-03-13 [1] CRAN (R 3.6.3)
##  curl            4.3      2019-12-02 [1] CRAN (R 3.6.3)
##  data.table      1.12.8   2019-12-09 [1] CRAN (R 3.6.3)
```

```
##  DBI             1.1.0    2019-12-15 [1] CRAN (R 3.6.3)
##  dbplyr          1.4.3    2020-04-19 [1] CRAN (R 3.6.3)
##  dichromat       2.0-0    2013-01-24 [1] CRAN (R 3.6.0)
##  digest          0.6.25   2020-02-23 [1] CRAN (R 3.6.3)
##  dplyr         * 0.8.5    2020-03-07 [1] CRAN (R 3.6.3)
##  e1071           1.7-3    2019-11-26 [1] CRAN (R 3.6.3)
##  ellipsis        0.3.0    2019-09-20 [1] CRAN (R 3.6.3)
##  envDocument   * 2.4.1    2019-08-19 [1] CRAN (R 3.6.3)
##  evaluate        0.14     2019-05-28 [1] CRAN (R 3.6.3)
##  fansi           0.4.1    2020-01-08 [1] CRAN (R 3.6.3)
##  fastmap         1.0.1    2019-10-08 [1] CRAN (R 3.6.3)
##  forcats       * 0.5.0    2020-03-01 [1] CRAN (R 3.6.3)
##  foreign         0.8-75   2020-01-20 [2] CRAN (R 3.6.3)
##  fs              1.4.1    2020-04-04 [1] CRAN (R 3.6.3)
##  generics        0.0.2    2018-11-29 [1] CRAN (R 3.6.3)
##  GGally        * 1.5.0    2020-03-25 [1] CRAN (R 3.6.3)
##  ggeffects     * 0.14.3   2020-04-20 [1] CRAN (R 3.6.3)
##  ggplot2       * 3.3.0    2020-03-05 [1] CRAN (R 3.6.3)
##  glue            1.4.0    2020-04-03 [1] CRAN (R 3.6.3)
##  gridExtra       2.3      2017-09-09 [1] CRAN (R 3.6.3)
##  gtable          0.3.0    2019-03-25 [1] CRAN (R 3.6.3)
##  haven           2.2.0    2019-11-08 [1] CRAN (R 3.6.3)
##  here          * 0.1      2017-05-28 [1] CRAN (R 3.6.3)
##  hms             0.5.3    2020-01-08 [1] CRAN (R 3.6.3)
##  htmltools       0.4.0    2019-10-04 [1] CRAN (R 3.6.3)
##  htmlwidgets     1.5.1    2019-10-08 [1] CRAN (R 3.6.3)
##  httpuv          1.5.2    2019-09-11 [1] CRAN (R 3.6.3)
##  httr            1.4.1    2019-08-05 [1] CRAN (R 3.6.3)
##  insight         0.8.3    2020-04-20 [1] CRAN (R 3.6.3)
##  jsonlite        1.6.1    2020-02-02 [1] CRAN (R 3.6.3)
##  KernSmooth      2.23-16  2019-10-15 [2] CRAN (R 3.6.3)
##  knitr           1.28     2020-02-06 [1] CRAN (R 3.6.3)
##  later           1.0.0    2019-10-04 [1] CRAN (R 3.6.3)
##  lattice       * 0.20-38  2018-11-04 [2] CRAN (R 3.6.3)
##  leafem          0.1.1    2020-04-05 [1] CRAN (R 3.6.3)
##  leaflet       * 2.0.3    2019-11-16 [1] CRAN (R 3.6.3)
##  leafsync        0.1.0    2019-03-05 [1] CRAN (R 3.6.3)
##  lifecycle       0.2.0    2020-03-06 [1] CRAN (R 3.6.3)
##  lme4          * 1.1-23   2020-04-07 [1] CRAN (R 3.6.3)
##  lubridate     * 1.7.8    2020-04-06 [1] CRAN (R 3.6.3)
##  lwgeom          0.2-3    2020-04-12 [1] CRAN (R 3.6.3)
##  magrittr        1.5      2014-11-22 [1] CRAN (R 3.6.3)
##  mapview       * 2.7.8    2020-04-07 [1] CRAN (R 3.6.3)
##  MASS          * 7.3-51.5 2019-12-20 [2] CRAN (R 3.6.3)
##  Matrix        * 1.2-18   2019-11-27 [2] CRAN (R 3.6.3)
##  memoise         1.1.0    2017-04-21 [1] CRAN (R 3.6.3)
##  mgcv            1.8-31   2019-11-09 [2] CRAN (R 3.6.3)
##  mime            0.9      2020-02-04 [1] CRAN (R 3.6.2)
##  minqa           1.2.4    2014-10-09 [1] CRAN (R 3.6.3)
##  modelr          0.1.6    2020-02-22 [1] CRAN (R 3.6.3)
##  munsell         0.5.0    2018-06-12 [1] CRAN (R 3.6.3)
##  NADA          * 1.6-1.1  2020-03-22 [1] CRAN (R 3.6.3)
##  nlme            3.1-144  2020-02-06 [2] CRAN (R 3.6.3)
##  nloptr          1.2.2.1  2020-03-11 [1] CRAN (R 3.6.3)
```

```
##  openxlsx        4.1.4    2019-12-06 [1] CRAN (R 3.6.3)
##  patchwork     * 1.0.0    2019-12-01 [1] CRAN (R 3.6.3)
##  permute       * 0.9-5    2019-03-12 [1] CRAN (R 3.6.3)
##  pillar          1.4.3    2019-12-20 [1] CRAN (R 3.6.3)
##  pkgconfig       2.0.3    2019-09-22 [1] CRAN (R 3.6.3)
##  plyr            1.8.6    2020-03-03 [1] CRAN (R 3.6.3)
##  png             0.1-7    2013-12-03 [1] CRAN (R 3.6.0)
##  promises        1.1.0    2019-10-04 [1] CRAN (R 3.6.3)
##  purrr         * 0.3.4    2020-04-17 [1] CRAN (R 3.6.3)
##  R6              2.4.1    2019-11-12 [1] CRAN (R 3.6.3)
##  raster        * 3.1-5    2020-04-19 [1] CRAN (R 3.6.3)
##  RColorBrewer  * 1.1-2    2014-12-07 [1] CRAN (R 3.6.0)
##  Rcpp            1.0.4.6  2020-04-09 [1] CRAN (R 3.6.3)
##  readr         * 1.3.1    2018-12-21 [1] CRAN (R 3.6.3)
##  readxl        * 1.3.1    2019-03-13 [1] CRAN (R 3.6.3)
##  reprex          0.3.0    2019-05-16 [1] CRAN (R 3.6.3)
##  reshape         0.8.8    2018-10-23 [1] CRAN (R 3.6.3)
##  rio             0.5.16   2018-11-26 [1] CRAN (R 3.6.3)
##  rlang           0.4.5    2020-03-01 [1] CRAN (R 3.6.3)
##  rmarkdown       2.1      2020-01-20 [1] CRAN (R 3.6.3)
##  rprojroot       1.3-2    2018-01-03 [1] CRAN (R 3.6.3)
##  rstudioapi      0.11     2020-02-07 [1] CRAN (R 3.6.3)
##  rvest           0.3.5    2019-11-08 [1] CRAN (R 3.6.3)
##  satellite       1.0.2    2019-12-09 [1] CRAN (R 3.6.3)
##  scales          1.1.0    2019-11-18 [1] CRAN (R 3.6.3)
##  sessioninfo   * 1.1.1    2018-11-05 [1] CRAN (R 3.6.3)
##  sf            * 0.9-2    2020-04-14 [1] CRAN (R 3.6.3)
##  shiny         * 1.4.0.2  2020-03-13 [1] CRAN (R 3.6.3)
##  sjlabelled      1.1.3    2020-01-28 [1] CRAN (R 3.6.3)
##  sjmisc        * 2.8.4    2020-04-03 [1] CRAN (R 3.6.3)
##  sp            * 1.4-1    2020-02-28 [1] CRAN (R 3.6.3)
##  spData        * 0.3.5    2020-04-06 [1] CRAN (R 3.6.3)
##  stars           0.4-1    2020-04-07 [1] CRAN (R 3.6.3)
##  statmod         1.4.34   2020-02-17 [1] CRAN (R 3.6.3)
##  stringdist    * 0.9.5.5  2019-10-21 [1] CRAN (R 3.6.1)
##  stringi         1.4.6    2020-02-17 [1] CRAN (R 3.6.2)
##  stringr       * 1.4.0    2019-02-10 [1] CRAN (R 3.6.3)
##  survival      * 3.1-8    2019-12-03 [2] CRAN (R 3.6.3)
##  tibble        * 3.0.0    2020-03-30 [1] CRAN (R 3.6.3)
##  tidyr         * 1.0.2    2020-01-24 [1] CRAN (R 3.6.3)
##  tidyselect      1.0.0    2020-01-27 [1] CRAN (R 3.6.3)
##  tidyverse     * 1.3.0    2019-11-21 [1] CRAN (R 3.6.3)
##  tmap          * 3.0      2020-04-09 [1] CRAN (R 3.6.3)
##  tmaptools       3.0      2020-03-30 [1] CRAN (R 3.6.3)
##  truncnorm     * 1.0-8    2018-02-27 [1] CRAN (R 3.6.3)
##  units           0.6-6    2020-03-16 [1] CRAN (R 3.6.3)
##  vctrs           0.2.4    2020-03-10 [1] CRAN (R 3.6.3)
##  vegan         * 2.5-6    2019-09-01 [1] CRAN (R 3.6.3)
##  viridis       * 0.5.1    2018-03-29 [1] CRAN (R 3.6.3)
##  viridisLite   * 0.3.0    2018-02-01 [1] CRAN (R 3.6.3)
##  webshot         0.5.2    2019-11-22 [1] CRAN (R 3.6.3)
##  withr           2.2.0    2020-04-20 [1] CRAN (R 3.6.3)
##  xfun            0.13     2020-04-13 [1] CRAN (R 3.6.3)
##  XML             3.99-0.3 2020-01-20 [1] CRAN (R 3.6.2)
```

```
##  xml2            1.3.1    2020-04-09 [1] CRAN (R 3.6.3)
##  xtable          1.8-4    2019-04-21 [1] CRAN (R 3.6.3)
##  yaml            2.2.1    2020-02-01 [1] CRAN (R 3.6.2)
##  zCompositions * 1.3.4    2020-03-04 [1] CRAN (R 3.6.3)
##  zip             2.0.4    2019-09-01 [1] CRAN (R 3.6.3)
##
## [1] C:/Users/Tobias/Documents/R/win-library/3.6
## [2] C:/Program Files/R/R-3.6.3/library
```

```
# rm(list=ls())
```

4.3 Chunk 3:- Reading data and selecting relevant columns

The .csv containing the data was read and loaded into a data frame where columns deemed irrelevant, such as sampling dates, lab notes and surveyor name were discarded.

```
options(tinytex.verbose = TRUE)

fhtwild <- read_csv("../data/fhtnmdf_ENV_wildOTUs_20191105.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   nm_Kit_ID = col_character(),
##   nm_Sampling_Date = col_character(),
##   fht_eDNA_survey_date = col_character(),
##   datecomp = col_character(),
##   fht_Site_comb = col_character(),
##   nm_Pond_Name = col_character(),
##   fht_Site_Name = col_character(),
##   Company = col_character(),
##   Sampler_Name = col_character(),
##   Sampling_DateTime = col_character(),
##   Arrival_Date = col_character(),
##   Damaged_tubes = col_character(),
##   Liquid_in_bag = col_character(),
##   Tubes_labelled = col_character(),
##   Tubes_filled = col_character(),
##   Additional_notes = col_character(),
##   Lab_work_Started = col_character(),
##   Status = col_character(),
##   Surveyor = col_character(),
##   `1_km_square` = col_character()
##   # ... with 18 more columns
## )

## See spec(...) for full column specifications.
```

```
fhtwild <- fhtwild %>%
#Removes the following COLS from the data sheet
  select(-nm_Sampling_Date:-Arrival_Date, -Additional_notes, -Lab_work_Started,
         -Site_number_CHECK, -Surveyor, -SITE_FROM_DIMITRIOS)
```

4.4 Chunk 4:- Extended Filtering after studying the data frame

Metabarcoding replicates that had *NA* values are removed along with any samples with liquid in the bag to mitigate contamination risk. Precalculated HSI values that had *NA* and any collected HSI data that were non-numerical and non-conforming were filtered out, along with data where *T. cristatus* presence, Phosphate and Nitrate levels, and categorised 'Clean' values were also reported to have an *NA* value. Samples analysed more than once, were removed, leaving $n = 235$.

```
options(tinytex.verbose = TRUE)


fhtwild <- fhtwild %>%
#removes mb_replicate and IPC N/As and any samples with liquid in bag (contamination risk)
   filter(
          !is.na(mb_replicate) & #removes IPC na as well
          Liquid_in_bag == "No" & Tubes_filled == "Yes" &
#removes N/A DIMITRIOS_HSI and any collected HSI data that didn't conform to suggested fields
          !is.na(DIMITRIOS_HSI)&
  # removes N/As for GCN presence, Phosphate and Nitrate levels, and catogorised 'Clean' values
          !is.na(Status)&
          !is.na(GCN_test_result)&
          !is.na(P)&
          !is.na(N)&
          !is.na(Clean)
)

fhtwild <- fhtwild %>%
   filter(
          HSI5_Shade != "<1%" &
          HSI5_Shade != "<1" &
          HSI8_Pond_count != ">12"&
          HSI8_Pond_count != ">30" &
          HSI8_Pond_count != "12+" &
          HSI8_Pond_count != ">20" &
          HSI10_Macrophytes != "30?"&
          HSI10_Macrophytes != "<1%" &
          HSI10_Macrophytes != "<1" &
          HSI10_Macrophytes != "10-100"&
          HSI10_Macrophytes != "5.00E-03" #convert to number
         )

IDcount <- fhtwild %>%
    group_by(nm_Kit_ID) %>%
    summarise(count = n()) %>%
    filter(count > 1) %>%
    arrange(nm_Kit_ID)
(IDcount)
```

```
## # A tibble: 12 x 2
##    nm_Kit_ID count
##    <chr>     <int>
##  1 FHT363        2
##  2 FHT364        2
##  3 FHT365        2
```

```
##  4 FHT367        2
##  5 FHT368        2
##  6 FHT370        2
##  7 FHT371        2
##  8 FHT429        2
##  9 FHT439        2
## 10 FHT698        2
## 11 FHT859        2
## 12 FHT869        2
```

```
# these ....

fhtwild <- fhtwild %>%
  filter(nm_Kit_ID != "FHT363"&
         nm_Kit_ID != "FHT364"&
         nm_Kit_ID != "FHT365"&
         nm_Kit_ID != "FHT367"&
         nm_Kit_ID != "FHT368"&
         nm_Kit_ID != "FHT370"&
         nm_Kit_ID != "FHT371"&
         nm_Kit_ID != "FHT429"&
         nm_Kit_ID != "FHT439"&
         nm_Kit_ID != "FHT698"&
         nm_Kit_ID != "FHT859"&
         nm_Kit_ID != "FHT869"  )

remove(IDcount)
```

4.5 Chunk 5:- Checking for False Positives/Negatives

False-positive and negative results within the data were isolated by comparing `Status` and `GCN_test_result` with `Amph_Caud_Sala_Tritcris`, where the latter shows detection of *T. cristatus* eDNA when over zero. If detected once from both tests then given Positive status. False negatives were changed into positive status with `mutate()` while false positives were removed from the data set. Any Inconclusive results were left as inconclusive regardless of the other tests outcome.

```
options(tinytex.verbose = TRUE)

False_Tests <- fhtwild %>%

  select(-Nitrate:-Amph_Caud_Sala_Lissvulg,
         -Aves_Acci_Acci_Butebute:-Mamm_Rode_Sciu_Sciucaro)

False_Negatives <- False_Tests %>%
  filter(Status == "Negative" &
         GCN_test_result == "Negative" &
         Amph_Caud_Sala_Tritcris > "0")
count(False_Negatives)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    15
```

```
False_Positives<- False_Tests %>%
  filter(Status == "Positive" &
        GCN_test_result == "Positive" &
           Amph_Caud_Sala_Tritcris == "0")
count(False_Positives)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    52
```

```
fhtwild <- fhtwild %>%
mutate(
    GCN_Binary = case_when(
      Status == "Negative" &
        GCN_test_result == "Negative" &
           Amph_Caud_Sala_Tritcris > "0" ~   "Positive", #False Negatives
      Status == "Positive" &
        GCN_test_result == "Positive" &
           Amph_Caud_Sala_Tritcris > "0" ~   "Positive", #Positive both datasets
      Status == "Negative" &
        GCN_test_result == "Negative" &
           Amph_Caud_Sala_Tritcris == "0" ~  "Negative", #Negative both datasets
       Status == "Negative" &
        GCN_test_result == "Positive" &
           Amph_Caud_Sala_Tritcris > "0" ~   "Positive", #Positive single dataset
       Status == "Positive" &
        GCN_test_result == "Negative" &
           Amph_Caud_Sala_Tritcris > "0" ~   "Positive", #Positive single dataset
      Status == "Inconclusive" &
        GCN_test_result == "Negative" &
           Amph_Caud_Sala_Tritcris == "0" ~   "Inconclusive",
     Status == "Inconclusive" &
        GCN_test_result == "Inconclusive" &
           Amph_Caud_Sala_Tritcris == "0" ~   "Inconclusive"
  ))

fhtwild <- fhtwild %>%
  filter(!is.na(GCN_Binary)) #Filters out remaining (False Positives)

False_Tests <- fhtwild %>%

  select(-Nitrate:-Amph_Caud_Sala_Lissvulg,
         -Aves_Acci_Acci_Butebute:-Mamm_Rode_Sciu_Sciucaro)

False_Positives <- False_Tests %>%
    filter(GCN_Binary == "Positive" &
           Amph_Caud_Sala_Tritcris == "0")
count(False_Positives)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
```

```
## 1     0
```

```
False_Negatives <- False_Tests %>%
    filter(GCN_Binary == "Negative" &
           Amph_Caud_Sala_Tritcris > "0")
count(False_Negatives)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1     0
```

```
remove(False_Tests)
remove(False_Positives)
remove(False_Negatives)
```

4.6 Chunk 6:- Calculating HSI

Values for HSI was calculated with guidance from the Great Crested Newt Habitat Suitability Index (ARG. 2010), and then using the outputs into the equation: $HSI = (SI_1 * SI_2 * SI_3 * SI_4 * SI_5 * SI_6 * SI_7 * SI_8 * SI_9 * SI_{10})^{1/10}$.

```
options(tinytex.verbose = TRUE)

fhtwild <- fhtwild %>%
    mutate(
        HSI3_Pond_drying = as.numeric(HSI3_Pond_drying),
        HSI4_Water_quality = as.numeric(HSI4_Water_quality),
        HSI5_Shade = as.numeric(HSI5_Shade),
        HSI6_Waterfowl = as.numeric(HSI6_Waterfowl),
        HSI7_Fish = as.numeric(HSI7_Fish),
        HSI8_Pond_count = as.numeric(HSI8_Pond_count),
        HSI9_Terrestrial_habitat = as.numeric(HSI9_Terrestrial_habitat),
        HSI10_Macrophytes = as.numeric(HSI10_Macrophytes)
    ) %>%
  mutate(
    pondarea_hsi= case_when(
      HSI2_Pond_area >= 0 & HSI2_Pond_area <= 500 ~
        (HSI2_Pond_area)*0.002, # gradient calculated using y = mx+b
      HSI2_Pond_area > 500 & HSI2_Pond_area <= 700 ~ 1.0,
      HSI2_Pond_area > 700 & HSI2_Pond_area <= 2000 ~
        HSI2_Pond_area*-0.000153846154 +1.1076923076923
                    # gradient calculated using y = mx+b
                # NA appears where area > 2000 (to be omitted from HSI calc)
      )
  ) %>%
  mutate(
    ponddry_hsi= case_when(
      HSI3_Pond_drying == 1 ~ 0.1,
      HSI3_Pond_drying == 1.5 ~ 0.25,
      HSI3_Pond_drying == 2 ~ 0.5,
      HSI3_Pond_drying == 2.5 ~ 0.75,
```

```
      HSI3_Pond_drying == 3 ~ 1.0,
      HSI3_Pond_drying == 3.5 ~ 0.95,
      HSI3_Pond_drying == 4 ~ 0.9
      )
  ) %>%
  mutate(
    waterquality_hsi= case_when(
      HSI4_Water_quality == 1 ~ 0.01,
      HSI4_Water_quality == 1.5 ~ 0.17,
      HSI4_Water_quality == 2 ~ 0.33,
      HSI4_Water_quality == 2.5 ~ 0.5,
      HSI4_Water_quality == 3 ~ 0.67,
      HSI4_Water_quality == 3.5 ~ 0.84,
      HSI4_Water_quality == 4 ~ 1.00
    )
  ) %>%
    filter(!is.na(HSI4_Water_quality)
           #Only one value turned up, DIMITRIOS_HSI has entry
           ) %>%
   mutate(
    HSI5_Shade = as.numeric(HSI5_Shade),
        shade_hsi = ifelse(
            (HSI5_Shade >= 0) & (HSI5_Shade <= 60), 1.0,
            ifelse((HSI5_Shade > 60) & (HSI5_Shade <= 100),
                   HSI5_Shade*-0.02 + 2.2,
                   # gradients calculated using y = mx+b
                   "don't know")
        )
    ) %>%
    mutate(
    waterfowl_hsi= case_when(
      HSI6_Waterfowl == 1 ~ 0.01,
      HSI6_Waterfowl == 2 ~ 0.67,
      HSI6_Waterfowl == 3 ~ 1.00
    )
  ) %>%
  mutate(
    fish_hsi= case_when(
      HSI7_Fish == 1 ~ 0.01,
      HSI7_Fish == 2 ~ 0.33,
      HSI7_Fish == 3 ~ 0.67,
      HSI7_Fish == 4 ~ 1.00
    )
  ) %>%
  mutate(
    pondcount_hsi= case_when(
      HSI8_Pond_count == 0 ~ 0.1,
      HSI8_Pond_count > 0 & HSI8_Pond_count <= 4 ~ 0.225*
        log(HSI8_Pond_count/3.14) + 0.9455339,
      HSI8_Pond_count > 4 ~ 1.0
#Pond count 0.9455339 value ensures  when x = 4, y = 1.
#-Inf occurred when HSI8 == 0 following the equation, so given a numerical value instead
    )
```

```
  ) %>%
  mutate(
    terrestrial_habitat_hsi= case_when(
      HSI9_Terrestrial_habitat == 0 ~ 0.00,
      HSI9_Terrestrial_habitat == 0.5 ~ 0.005,
      HSI9_Terrestrial_habitat == 1 ~ 0.01,
      HSI9_Terrestrial_habitat == 1.5 ~ 0.18,
      HSI9_Terrestrial_habitat == 2 ~ 0.33,
      HSI9_Terrestrial_habitat == 2.5 ~ 0.51,
      HSI9_Terrestrial_habitat == 3 ~ 0.67,
      HSI9_Terrestrial_habitat == 3.5 ~ 0.85,
      HSI9_Terrestrial_habitat == 4 ~ 1.00
  )
  ) %>%
   mutate(
        macrophytes_hsi = ifelse(
          HSI10_Macrophytes >= 0 & HSI10_Macrophytes <= 70, HSI10_Macrophytes*0.01 +0.3,
                                                  # gradient calculated using y = mx+b
          ifelse(HSI10_Macrophytes > 70 & HSI10_Macrophytes <= 80, 1.0,
          ifelse(HSI10_Macrophytes > 80 & HSI10_Macrophytes <= 100,
                 # gradient calculated using y = mx+b
                 HSI10_Macrophytes*-0.01 +1.8, "Don't Know"))
        )
   )
```

```
## Warning: NAs introduced by coercion
```

```
fhtwild <- fhtwild %>%

  mutate(
        pondarea_hsi =  as.numeric(pondarea_hsi),
        ponddry_hsi = as.numeric(ponddry_hsi),
        waterquality_hsi = as.numeric(waterquality_hsi),
        shade_hsi = as.numeric(shade_hsi),
        waterfowl_hsi = as.numeric(waterfowl_hsi),
        fish_hsi = as.numeric(fish_hsi),
        pondcount_hsi = as.numeric(pondcount_hsi),
        terrestrial_habitat_hsi = as.numeric(terrestrial_habitat_hsi),
        macrophytes_hsi = as.numeric(macrophytes_hsi)
  ) %>%
  mutate(
  HSI_val = ifelse(
    !is.na(pondarea_hsi),
       ((pondarea_hsi*ponddry_hsi*waterquality_hsi*shade_hsi*waterfowl_hsi*
           fish_hsi*pondcount_hsi*terrestrial_habitat_hsi*macrophytes_hsi)^0.1),
    ((ponddry_hsi*waterquality_hsi*shade_hsi*waterfowl_hsi*fish_hsi*
        pondcount_hsi*terrestrial_habitat_hsi*macrophytes_hsi)^(1/9))
               )
  )
```

4.7 Chunk 7:- HSI Tests

Tests to make sure that the calculated HSI values fall within the requirements set out in the aforementioned Great Crested Newt Habitat Suitability Index, followed by a test of correlation between pre-calculated HSI (`DIMITRIOS_HSI`) and the calculated values from `Chunk 6` (`HSI_val`).

```
options(tinytex.verbose = TRUE)

HSI_NA <- fhtwild %>%     #Checks there are no NA's in calculated HSI
  filter(is.na(HSI_val))
count(HSI_NA)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1     0
```

```
HSI_over <- fhtwild %>%   #Checks there are no values >1 in calculated HSI
  filter(HSI_val > 1)
count(HSI_over)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1     0
```

```
HSI_under <- fhtwild %>%  #Checks there are no values <0 in calculated HSI
  filter(HSI_val < 0)
count(HSI_under)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1     0
```

```
remove(HSI_NA)
remove(HSI_over)
remove(HSI_under)

HSIf1 <- ggpairs(fhtwild,
        columns = c("HSI_val", "DIMITRIOS_HSI"),
        upper = list(continuous = wrap("cor",
      size = 9)),
        lower = list(continuous = "smooth"))+theme_cowplot()
```

Once the HSI value (`HSI_val`) was calculated, it was compared with the pre-calculated values (`DIMITRIOS_HSI`) to test for correlation (Figure 1.), where a moderately positive relationship between pre-calculated and coded HSI was found (corr = 0.69).

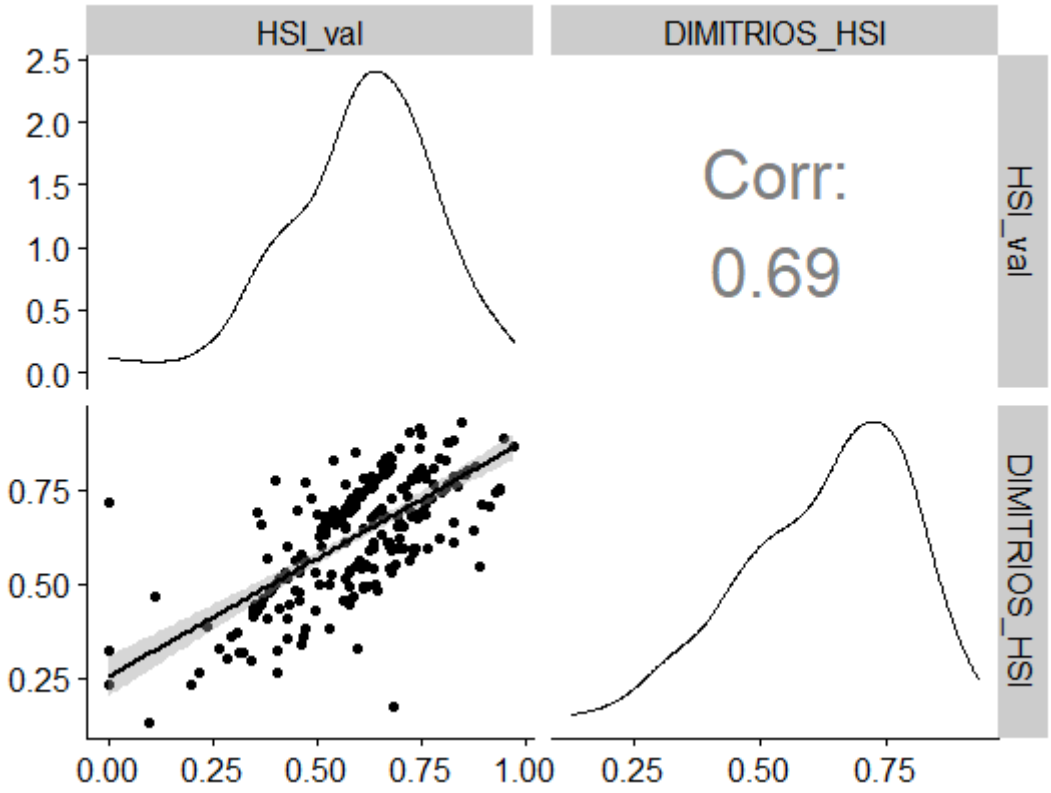


Figure 1: (Produced in Chunk 7). Samples were plotted onto a graph (bottom left) comparing pre-determined Habitat Suitability Index scores (DIMITRIOS_HSI, y-axis) with those calculated in Chunk 6 (HSI_val). Correlation (in top right) = 0.69, n = 235.

4.8 Chunk 8:- GCN Test result and HSI Boxplot

Results of *T. cristatus'* presence were analysed in comparison to the calculated HSI, with mean and standard deviation being used to create a boxplot with `geom_boxplot`.

```
options(tinytex.verbose = TRUE)

n_total_sample<-fhtwild %>% #n of samples used
group_by(Tubes_filled) %>%
tally()
(n_total_sample)     #n = 235
```

```
## # A tibble: 1 x 2
##   Tubes_filled     n
##   <chr>        <int>
## 1 Yes            235
```

```
n_update_group<-fhtwild %>%
  #Group fhtwild by status, tally +ve, -ve, inc.
group_by(GCN_Binary) %>%
  count()
(n_update_group) # Inconclusive = 9, Negative = 171, Positive = 55
```

```
## # A tibble: 3 x 2
## # Groups:   GCN_Binary [3]
##   GCN_Binary       n
##   <chr>        <int>
## 1 Inconclusive     9
## 2 Negative       171
## 3 Positive        55
```

```
HSI_mean<-fhtwild %>%
group_by(GCN_Binary) %>%
  summarise(
        mean_HSI = mean(HSI_val))
(HSI_mean) # Inconclusive = 0.622, Negative = 0.580, Positive = 0.679
```

```
## # A tibble: 3 x 2
##   GCN_Binary   mean_HSI
##   <chr>           <dbl>
## 1 Inconclusive    0.622
## 2 Negative        0.580
## 3 Positive        0.679
```

```
HSI_SD<-fhtwild %>%
group_by(GCN_Binary) %>%
  summarise(
        sd_HSI = sd(HSI_val))
(HSI_SD)  # Inconclusive = 0.127, Negative = 0.190, Positive = 0.121
```

```
## # A tibble: 3 x 2
```

```
##   GCN_Binary   sd_HSI
##   <chr>         <dbl>
## 1 Inconclusive  0.127
## 2 Negative      0.190
## 3 Positive      0.121
```

```
remove(n_total_sample)
remove(n_update_group)
remove(HSI_mean)
remove(HSI_SD)

GCN.HSIboxplot <- ggplot(fhtwild, aes(x = GCN_Binary, y = HSI_val, colour = GCN_Binary)) +
  geom_boxplot() +
  geom_jitter(width = 0.1, height = 0)+
     labs(x = "Results of tests for presence of T. cristatus", y = "HSI value") +
    theme_classic()
```

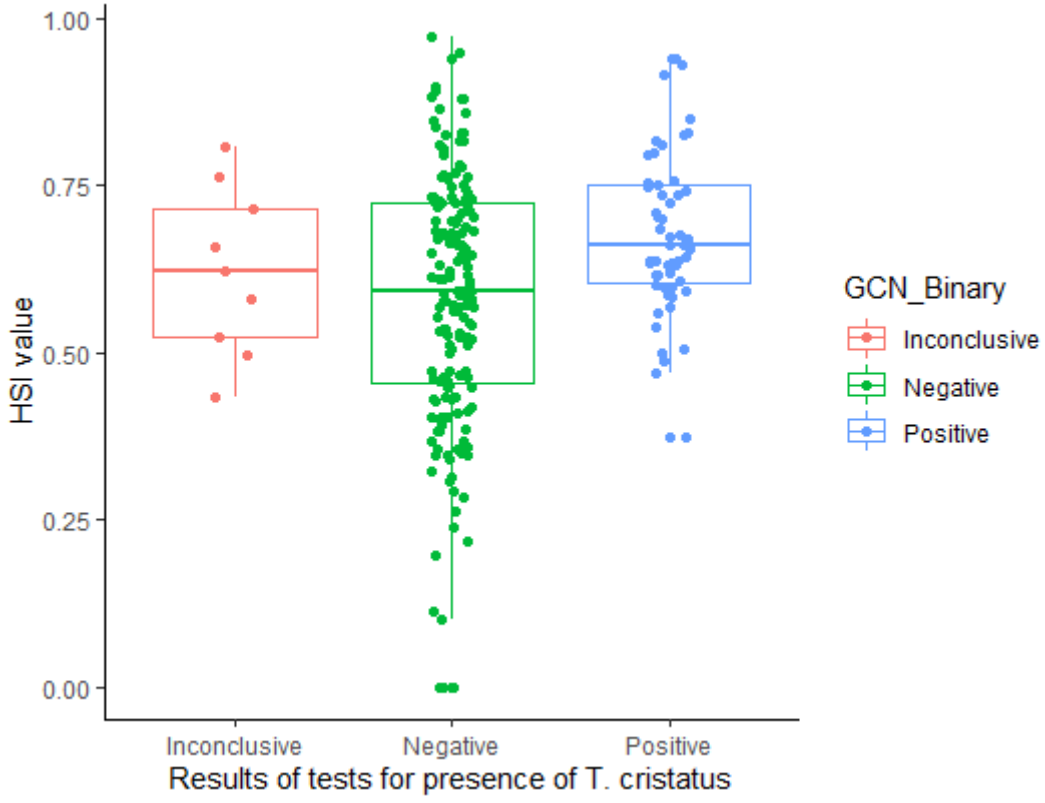


Figure 2: (Produced in Chunk 8). Boxplot of results concerning *T. cristatus* presence compared with calculated HSI value. Inconclusive: n = 9, mean = 0.622, SD = 0.127; Negative: n = 171, mean = 0.580, SD = 0.190; Positive: n = 55, mean = 0.679, SD = 0.121

With a total number of samples at 235, the data was seperated by results of *T. cristatus* presence, and analysed in a boxplot (Figure 2.) showing that; Inconclusive: n = 9, mean = 0.622, SD = 0.127, Negative: n = 171, mean = 0.580, SD = 0.190 Positive: n = 55, mean = 0.679, SD = 0.121.

4.9 Chunk 9:- Nitrate/Phosphate analysis with HSI_val

```
options(tinytex.verbose = TRUE)

NPFILT <- fhtwild %>%
  filter(GCN_Binary == "Positive")


fhtwild <- fhtwild %>%
    mutate(
    P = as.numeric(P),
    N = as.numeric(N)
    ) %>%
   mutate(
        N_level = ifelse(
             (N > 1), 0,
     ifelse ((N >= 0.5) & (N <= 1), 1, 2)
      ))%>%
        mutate(
        P_level = ifelse(
             (P > 0.1), 0,
     ifelse ((P >= 0.05) & (P <= 0.1), 1, 2))
       )

NPFILT <- NPFILT %>%
    mutate(
    P = as.numeric(P),
    N = as.numeric(N)
    ) %>%
   mutate(
        N_level = ifelse(
             (N > 1), 0,
     ifelse ((N >= 0.5) & (N <= 1), 1, 2))
     ) %>%
        mutate(
        P_level = ifelse(
             (P > 0.1), 0,
     ifelse ((P >= 0.05) & (P <= 0.1), 1, 2))
       )


NPFILT.clean.HSI <-
  (ggplot(fhtwild, aes(x = Clean, y = HSI_val, colour = GCN_Binary)) +
   geom_jitter(width = 0.1, height = 0) +
    labs(x = "Pollution Levels (0=Highly Polluted, 1=Some Pollution, 2=Clean)", y = "HSI value") +
      theme_classic())&
  (ggplot(NPFILT, aes(x = Clean, y = HSI_val, colour = GCN_Binary)) +
   geom_jitter(width = 0.1, height = 0) +
    labs(x = "Pollution Levels (0=Highly Polluted, 1=Some Pollution, 2=Clean)", y = "HSI value") +
     theme_classic())


Clean_group.all <-fhtwild %>%
```

```
  #Group fhtwild by Clean, 0,1,2.
group_by(Clean) %>%
  count()
(Clean_group.all)
```

```
## # A tibble: 3 x 2
## # Groups:   Clean [3]
##   Clean     n
##   <dbl> <int>
## 1     0    17
## 2     1   113
## 3     2   105
```

```
Clean_group.pos <-NPFILT %>%
  #Group fhtwild by Clean, 0,1,2.
group_by(Clean) %>%
  count()
(Clean_group.pos)
```

```
## # A tibble: 3 x 2
## # Groups:   Clean [3]
##   Clean     n
##   <dbl> <int>
## 1     0     2
## 2     1    27
## 3     2    26
```

```
NPFILT.N.HSI <-
  (ggplot(fhtwild, aes(x = N, y =HSI_val, colour = GCN_Binary)) +
    geom_jitter(width = 0.1, height = 0) +
       labs(x = bquote ('Recorded Nitrate levels '(mgL^-1)), y = "HSI value")+
        theme_classic()) &
   (ggplot(NPFILT, aes(x = N, y =HSI_val, colour = GCN_Binary)) +
    geom_jitter(width = 0.1, height = 0) +
       labs(x = bquote ('Recorded Nitrate levels '(mgL^-1)), y = "HSI value")+
      theme_classic())


N_group.all <-fhtwild %>%
    group_by(N) %>%
      count()
    (N_group.all)
```

```
## # A tibble: 7 x 2
## # Groups:   N [7]
##       N     n
##   <dbl> <int>
## 1  0.1    191
## 2  0.35    20
## 3  0.75     9
## 4  1.5      4
## 5  3.5      3
```

```
## 6  7.5      2
## 7 11        6
```

```
    N_SD.all <-fhtwild %>%
      group_by(N_level) %>%
        summarise(
        sd_HSI = sd(N))
    (N_SD.all)
```

```
## # A tibble: 3 x 2
##   N_level sd_HSI
##     <dbl>  <dbl>
## 1       0 4.23
## 2       1 0
## 3       2 0.0734
```

```
N_group.pos<-NPFILT %>%
      group_by(N) %>%
       count()
      (N_group.pos)
```

```
## # A tibble: 5 x 2
## # Groups:   N [5]
##       N     n
##   <dbl> <int>
## 1  0.1     49
## 2  0.35     3
## 3  0.75     1
## 4  3.5      1
## 5 11        1
```

```
    N_SD.pos <-NPFILT %>%
      group_by(N_level) %>%
        summarise(
        sd_N = sd(N))
    (N_SD.pos)
```

```
## # A tibble: 3 x 2
##   N_level    sd_N
##     <dbl>   <dbl>
## 1       0  5.30
## 2       1 NA
## 3       2  0.0589
```

```
NPFILT.P.HSI <-
  (ggplot(fhtwild, aes(x = P, y = HSI_val, colour = GCN_Binary)) +
    geom_jitter(width = 0.1, height = 0)+
       labs(x = bquote ('Recorded Phosphate levels '(mgL^-1)), y = "HSI value") +
  theme_classic() &
(ggplot(NPFILT, aes(x = P, y = HSI_val, colour = GCN_Binary)) +
    geom_jitter(width = 0.1, height = 0)+
```

```
       labs(x = bquote ('Recorded Phosphate levels '(mgL^-1)), y = "HSI value") +
         theme_classic()))


P_group.all <-fhtwild %>%
      group_by(P) %>%
      count()
    (P_group.all)
```

```
## # A tibble: 7 x 2
## # Groups:   P [7]
##       P     n
##   <dbl> <int>
## 1 0.01     59
## 2 0.035    59
## 3 0.075    33
## 4 0.15     31
## 5 0.35     25
## 6 0.75     22
## 7 1.1       6
```

```
   P_SD.all <- fhtwild %>%
      group_by(P_level) %>%
        summarise(
        sd_P = sd(P))
      (P_SD.all)
```

```
## # A tibble: 3 x 2
##   P_level   sd_P
##     <dbl>  <dbl>
## 1       0 0.301
## 2       1 0
## 3       2 0.0126
```

```
P_group.pos <-NPFILT %>%
      group_by(P) %>%
      count()
    (P_group.pos)
```

```
## # A tibble: 6 x 2
## # Groups:   P [6]
##       P     n
##   <dbl> <int>
## 1 0.01     14
## 2 0.035    15
## 3 0.075     4
## 4 0.15      7
## 5 0.35      9
## 6 0.75      6
```

```
   P_SD.pos <-NPFILT %>%
      group_by(P_level) %>%
        summarise(
        sd_P = sd(P))
      (P_SD.pos)
```

```
## # A tibble: 3 x 2
##   P_level   sd_P
##     <dbl>  <dbl>
## 1       0 0.239
## 2       1 0
## 3       2 0.0127
```

```
remove(Clean_group.pos)
remove(Clean_group.all)
remove(P_group.all)
remove(P_group.pos)
remove(N_group.all)
remove(N_group.pos)
remove(N_SD.all)
remove(N_SD.pos)
remove(P_SD.all)
remove(P_SD.pos)
remove(NPFILT)
```

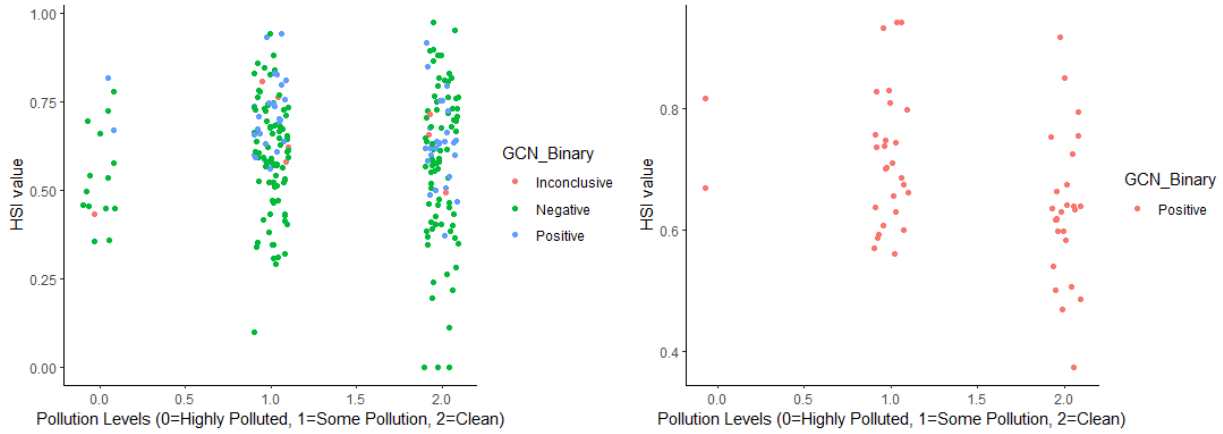


Figure 3: (Produced in Chunk 9). **0** is classified as High or very high levels of pollution where phosphate >0.1 mgL^{-1}, nitrate >1 mgL^{-1}; **1** shows some evidence of pollution where phosphate 0.05-0.1 mgL^{-1}, nitrate 0.5-1 mgL^{-1}; and **2** signifying clean water, with phosphate <0.05 mgL^{-1}, and nitrate <0.5 mgL^{-1}. **All samples** (left) have an n of 0 = 17, 1 = 113, 2 = 105; while **positive results** (right) only show 0 = 2, 1 = 27, 2 = 26.

In the original dataset, clean values were assigned in accordance with the Clean Water for Wildlife Technical Manual (Biggs, *et al.* 2016) where **(0)** is classified as High or very high levels of pollution where phosphate >0.1 mgL^{-1}, nitrate >1 mgL^{-1}; **(1)** shows some evidence of pollution where phosphate 0.05-0.1 mgL^{-1}, nitrate 0.5-1 mgL^{-1}; and **(2)** signifying clean water, with phosphate <0.05 mgL^{-1}, and nitrate <0.5 mgL^{-1}. Totals were counted for each cleanliness level (Figure 3), where, for pooled samples, n for each cleanliness level: 0 = 17, 1 = 113, 2 = 105, while for positive, n for each cleanliness level: 0 = 2, 1 = 27, 2 = 26.

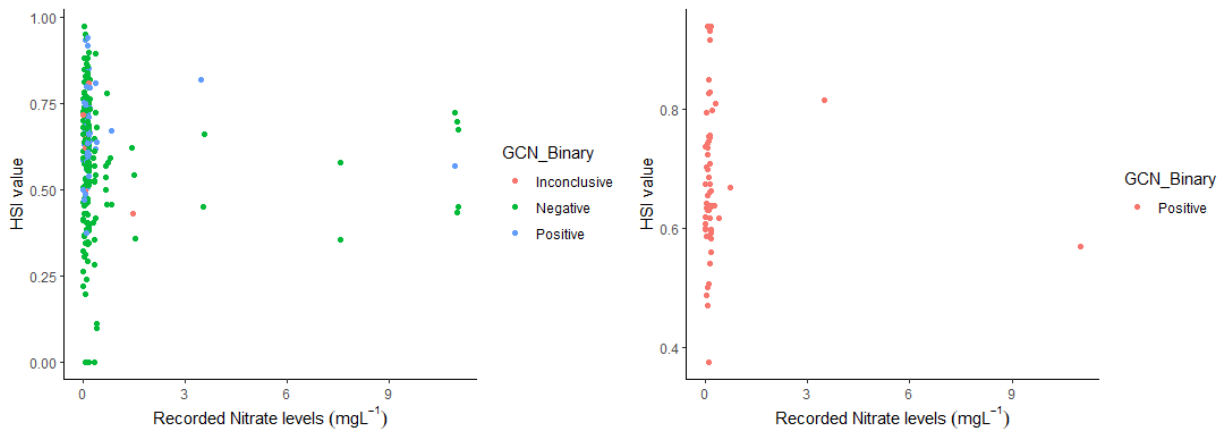


Figure 4: (Produced in Chunk 9). Nitrate levels for **all samples** (n where: <0.5 mgL^{-1} = 211; 0.5-1 mgL^{-1} = 9; >1 mgL^{-1} = 15, and SD where: <0.5 mgL^{-1} = 0.0734; 0.5-1 mgL^{-1} = 0.00 and >1 mgL^{-1} = 4.23) and **Positive samples** (right) (n where: <0.5 mgL^{-1} = 52; 0.5-1 mgL^{-1} = 1; >1 mgL^{-1} = 2, and SD where: <0.05 mgL^{-1} = 0.0589; 0.05-0.1 mgL^{-1} = $N.A.$; and >0.1 mgL^{-1} = 5.30).

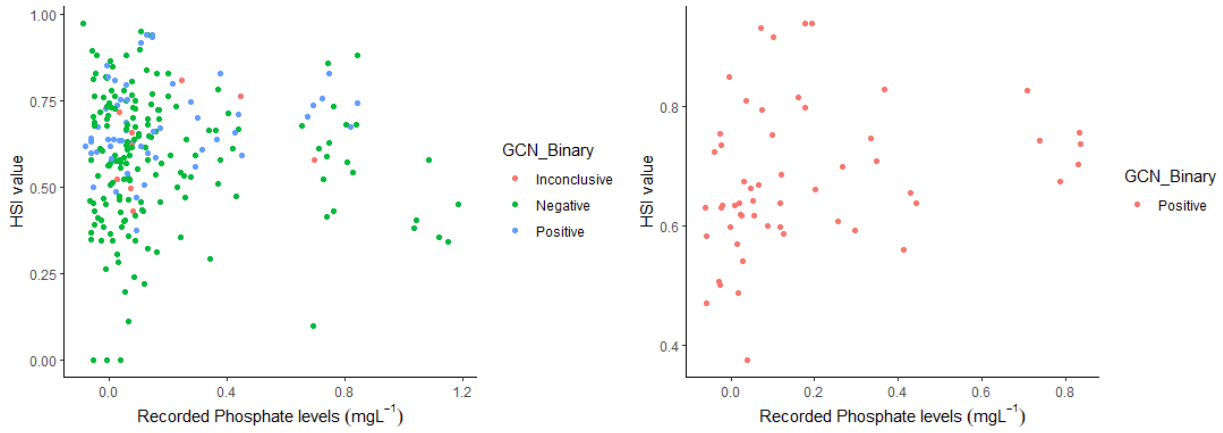


Figure 5: (Produced in Chunk 9). Phosphate levels for: **All samples** (left) (n where: <0.05 mgL^{-1} = 118; 0.05-0.1 mgL^{-1} = 33; >0.1 mgL^{-1} = 84, and SD where: <0.05 mgL^{-1} = 0.0172; 0.05-0.1 mgL^{-1} = 0.00 and >0.1 mgL^{-1} = 0.301) and **Positive samples** (right) (n where: <0.05 mgL^{-1} = 28; 0.05-0.1 mgL^{-1} = 4; >0.1 mgL^{-1} = 22, and SD where: <0.05 mgL^{-1} = 0.0127; 0.05-0.1 mgL^{-1} = 0.00; and >0.1 mgL^{-1} = 0.0239).

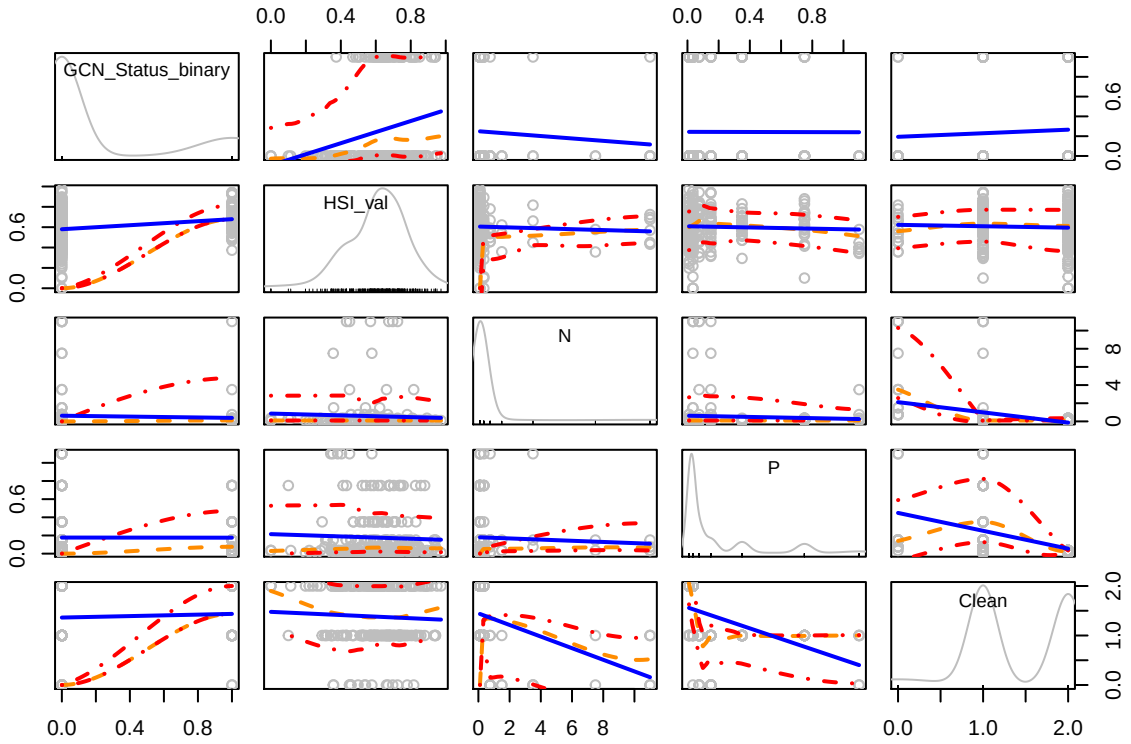
Given the collected data, pollution levels can be further analysed by looking at nitrate and phosphate levels separately. The nitrate levels, shown in Figure 4., show that for all samples, n where samples are: <0.5 mgL^{-1} = 211; 0.5-1 mgL^{-1} = 9; >1 mgL^{-1} = 15, with *SD*'s where: <0.5 mgL^{-1} = 0.0734; 0.5-1 mgL^{-1} = 0.00 and >1 mgL^{-1} = 4.23. For the Positive samples, n where: <0.5 mgL^{-1} = 52; 0.5-1 mgL^{-1} = 1; >1 mgL^{-1} = 2, with *SD*'s of: <0.05 mgL^{-1} = 0.0589; 0.05-0.1 mgL^{-1} = *N.A.* (as only one sample exists in this category); and >0.1 mgL^{-1} = 5.30.

Phosphate levels, shown in Figure 5., show that for all samples, n where samples are: <0.05 mgL^{-1} = 118; 0.05-0.1 mgL^{-1} = 33; >0.1 mgL^{-1} = 84, and *SD* where: <0.05 mgL^{-1} = 0.0126; 0.05-0.1 mgL^{-1} = 0.00 and >0.1 mgL^{-1} = 0.301. Positive samples have phosphate levels where n: <0.05 mgL^{-1} = 29; 0.05-0.1 mgL^{-1} = 4; >0.1 mgL^{-1} = 22, with *SD*'s of: <0.05 mgL^{-1} = 0.0127; 0.05-0.1 mgL^{-1} = 0.00; and >0.1 mgL^{-1} = 0.0239.

4.10 Chunk 10:- Scatterplot/GLM/Chi-square

```
options(tinytex.verbose = TRUE)

fhtnmdf_GCN <- fhtwild %>%
  select(mb_replicate, GCN_Positive_out_of_12, Status, GCN_test_result, HSI_val, N, P,
         Clean, Inflow_present, Outflow_present, nm_Kit_ID, GCN_Binary) %>%
  mutate(
    GCN_Status_binary = case_when(
      GCN_Binary == "Positive" ~ 1,
      GCN_Binary == "Negative" ~ 0,
    )
  ) %>%
  filter(!is.na(GCN_Status_binary) & !is.na(HSI_val))


scatterplotMatrix(~ GCN_Status_binary +HSI_val + N + P + Clean,
              data = fhtnmdf_GCN, regLine = list(col=c("Blue")),
                  smooth=list(col.smooth="dark orange", col.spread="red"), col = c('grey'))
```

```
## Warning in smoother(x[subs], y[subs], col = smoother.args$col[i], log.x =
## FALSE, : could not fit smooth

## Warning in smoother(x[subs], y[subs], col = smoother.args$col[i], log.x =
## FALSE, : could not fit smooth

## Warning in smoother(x[subs], y[subs], col = smoother.args$col[i], log.x =
## FALSE, : could not fit smooth
```




```
# no obvious collinearity between N or P  with HSI

model1 <- glm(GCN_Status_binary ~ N + P + Clean + HSI_val, family = binomial, data = fhtnmdf_GCN)
summary(model1)
```

```
##
## Call:
## glm(formula = GCN_Status_binary ~ N + P + Clean + HSI_val, family = binomial,
##     data = fhtnmdf_GCN)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2702  -0.7946  -0.6027  -0.2701   2.0420
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.91210    0.96465  -4.055  0.00005 ***
## N           -0.02072    0.10944  -0.189 0.849812
## P            0.44412    0.73321   0.606 0.544698
## Clean        0.30291    0.34551   0.877 0.380645
## HSI_val      3.61763    1.04626   3.458 0.000545 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 250.83  on 225  degrees of freedom
## Residual deviance: 235.80  on 221  degrees of freedom
## AIC: 245.8
##
## Number of Fisher Scoring iterations: 4
```

```
# Deviance is the deviance left over after the model has been fit (the residual)
model1$deviance
```

```
## [1] 235.8022
```

```
# df.residual is the number of degrees of freedom leftover after fitting the model
model1$df.residual
```

```
## [1] 221
```

```
# We can check for overdispersion by calculating this ratio:
model1$deviance / model1$df.residual
```

```
## [1] 1.066979
```

```
# this (1.066979) is < 2, so we can go with model selection instead of fitting with quasipoisson

model2 <- glm(GCN_Status_binary ~ N + HSI_val, family = binomial, data = fhtnmdf_GCN)
summary(model2)
```

```
##
## Call:
## glm(formula = GCN_Status_binary ~ N + HSI_val, family = binomial,
##     data = fhtnmdf_GCN)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2190  -0.8078  -0.6240  -0.2627   2.0734
##
## Coefficients:
##             Estimate Std. Error z value   Pr(>|z|)
## (Intercept) -3.34330    0.70707  -4.728 0.00000226 ***
## N           -0.06127    0.10260  -0.597   0.550347
## HSI_val      3.54281    1.04188   3.400   0.000673 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 250.83  on 225  degrees of freedom
## Residual deviance: 236.62  on 223  degrees of freedom
## AIC: 242.62
##
## Number of Fisher Scoring iterations: 4
```

```r
MASS::dropterm(model1, test = "Chi")
```

```
## Single term deletions
##
## Model:
## GCN_Status_binary ~ N + P + Clean + HSI_val
##         Df Deviance    AIC     LRT   Pr(Chi)
## <none>       235.80 245.80
## N        1   235.84 243.84  0.0371 0.8472855
## P        1   236.16 244.16  0.3595 0.5487746
## Clean    1   236.59 244.59  0.7904 0.3739674
## HSI_val  1   249.77 257.77 13.9646 0.0001863 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(model1, model2, test = "Chi") #0.6654
```

```
## Analysis of Deviance Table
##
## Model 1: GCN_Status_binary ~ N + P + Clean + HSI_val
## Model 2: GCN_Status_binary ~ N + HSI_val
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       221     235.80
## 2       223     236.62 -2 -0.81472   0.6654
```

```r
model3 <- glm(GCN_Status_binary ~ HSI_val, family = binomial, data = fhtnmdf_GCN)
summary(model3)
```

```
##
## Call:
## glm(formula = GCN_Status_binary ~ HSI_val, family = binomial,
##     data = fhtnmdf_GCN)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2155  -0.8028  -0.6263  -0.2556   2.0893
##
## Coefficients:
##             Estimate Std. Error z value  Pr(>|z|)
## (Intercept)  -3.4048     0.7035  -4.840 0.0000013 ***
## HSI_val       3.5913     1.0424   3.445  0.000571 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 250.83  on 225  degrees of freedom
## Residual deviance: 237.02  on 224  degrees of freedom
## AIC: 241.02
##
## Number of Fisher Scoring iterations: 4
```

```r
MASS::dropterm(model1, test = "Chi") # P's p = 0.048; N's p = 0.034 (2 s.f.)
```

```
## Single term deletions
##
## Model:
## GCN_Status_binary ~ N + P + Clean + HSI_val
##         Df Deviance    AIC     LRT   Pr(Chi)
## <none>       235.80 245.80
## N        1   235.84 243.84  0.0371 0.8472855
## P        1   236.16 244.16  0.3595 0.5487746
## Clean    1   236.59 244.59  0.7904 0.3739674
## HSI_val  1   249.77 257.77 13.9646 0.0001863 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(model2, model3, test = "Chi") # 0.5239
```

```
## Analysis of Deviance Table
##
## Model 1: GCN_Status_binary ~ N + HSI_val
## Model 2: GCN_Status_binary ~ HSI_val
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       223     236.62
## 2       224     237.02 -1 -0.40614   0.5239
```

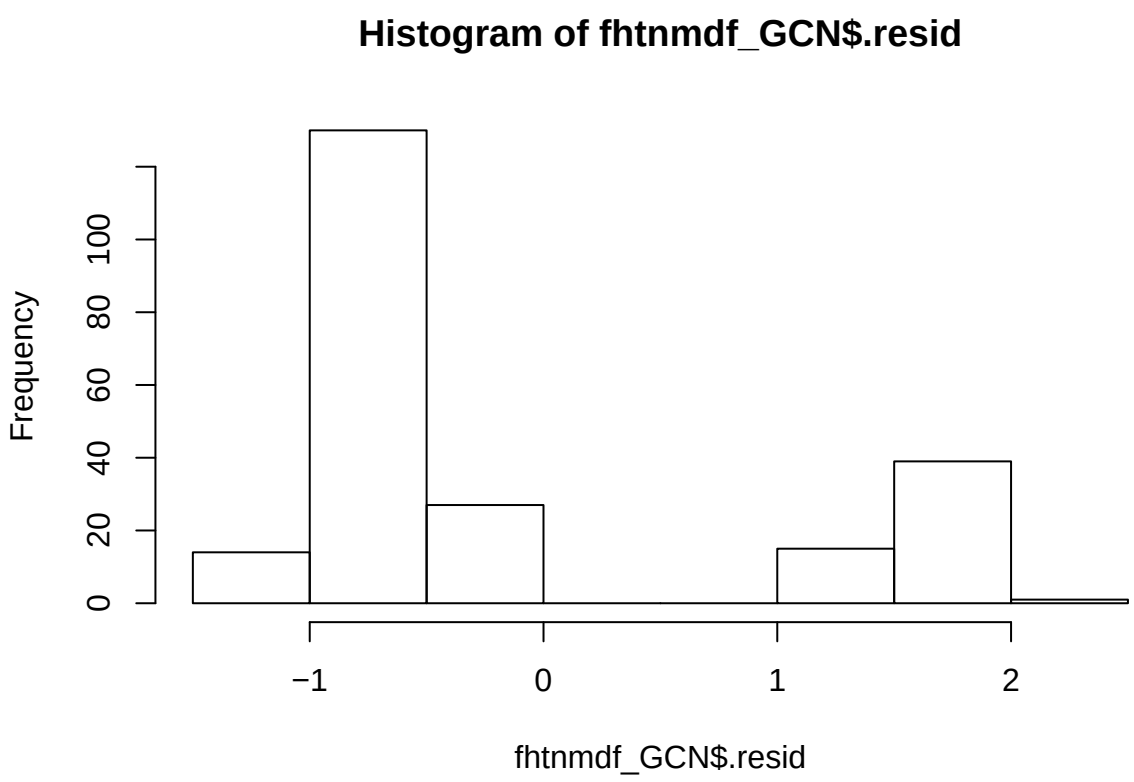
```r
# broom package to extract information from a model
(model3_parms <- tidy(model3)) # model parameters
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic    p.value
##   <chr>          <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)    -3.40     0.704     -4.84 0.00000130
## 2 HSI_val         3.59     1.04       3.45 0.000571
```

```r
(model3_dev <- glance(model3)) # model deviance
```

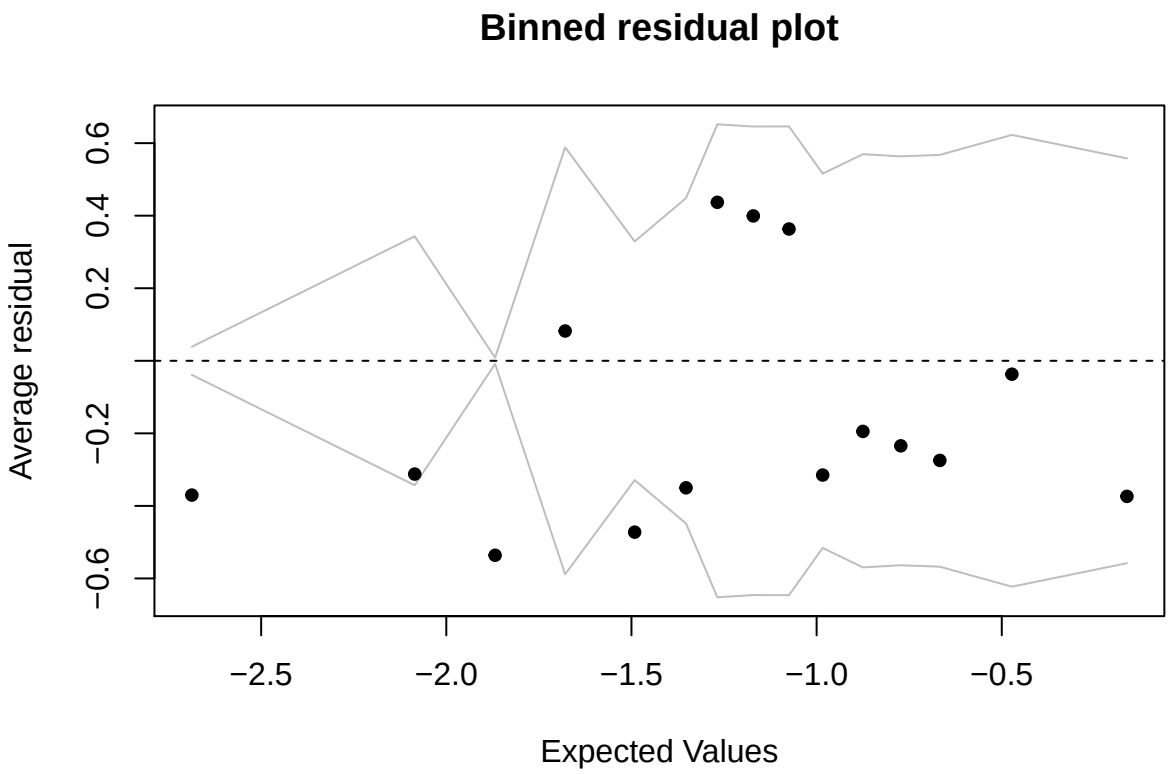
```
## # A tibble: 1 x 7
##   null.deviance df.null logLik   AIC   BIC deviance df.residual
##           <dbl>   <int>  <dbl> <dbl> <dbl>    <dbl>       <int>
## 1          251.     225  -119.  241.  248.     237.         224
```

```r
#using the broom package to add the fitted model estimates to the original dataset
fhtnmdf_GCN <- augment(model3, fhtnmdf_GCN)
(hist(fhtnmdf_GCN$.resid))
```




```
## $breaks
## [1] -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5
##
## $counts
## [1]  14 130  27   0   0  15  39   1
##
## $density
## [1] 0.123893805 1.150442478 0.238938053 0.000000000 0.000000000 0.132743363
## [7] 0.345132743 0.008849558
##
## $mids
## [1] -1.25 -0.75 -0.25  0.25  0.75  1.25  1.75  2.25
##
## $xname
## [1] "fhtnmdf_GCN$.resid"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
(binnedplot(fhtnmdf_GCN$.fitted, fhtnmdf_GCN$.resid))
```




```
## NULL
```

```
# Check that 95% of residuals fall within the grey lines indication +/-2SE
# SE=0.704, 2SE =  1.408
# 13.1408/15 = 89.3

# Null deviance is the total amount of deviance e in the null model)
model3_dev$null.deviance
```

```
## [1] 250.8262
```

```
# Deviance is the deviance left over after the model has been fit (the residual)
model3_dev$deviance
```

```
## [1] 237.0231
```

```
# % deviance explained is thus:
(model3_dev$null.deviance - model3_dev$deviance) / model3_dev$null.deviance
```

```
## [1] 0.05503062
```

```
# ($\frac{(250.8262 - 237.0231)}{250.8262}= 0.0550305351$)
# 5.5%

# fitted model equation
model3pred <- ggpredict(model3, terms = c("HSI_val[all]"))
# [all] produces more points for a smoother predicted line
model3pred
```

```
##
## # Predicted values of GCN_Status_binary
## # x = HSI_val
##
##    x | Predicted |   SE |       95% CI
## -------------------------------------
## 0.00 |      0.03 | 0.70 | [0.01, 0.12]
## 0.38 |      0.12 | 0.33 | [0.07, 0.20]
## 0.47 |      0.15 | 0.25 | [0.10, 0.23]
## 0.57 |      0.20 | 0.18 | [0.15, 0.27]
## 0.62 |      0.23 | 0.17 | [0.18, 0.30]
## 0.67 |      0.27 | 0.16 | [0.21, 0.33]
## 0.73 |      0.31 | 0.18 | [0.24, 0.39]
## 0.97 |      0.52 | 0.37 | [0.35, 0.69]
```

```
(colorvec <- brewer.pal(7, "RdYlBu"))
```

```
## [1] "#D73027" "#FC8D59" "#FEE090" "#FFFFBF" "#E0F3F8" "#91BFDB" "#4575B4"
```

```
GCNBinary.HSI <-
  (ggplot() + # diff geoms use diff datasets
    geom_jitter(data = fhtnmdf_GCN, aes(x = HSI_val, y = GCN_Status_binary), width = .1,
    height = .1, size = 1, shape = 21) +
    labs(x = "Pond HSI value", y = "Probability of GCN presence") +
    theme_cowplot() + # or theme_bw() if you don't have the cowplot package
    geom_line(data = model3pred, aes(x = x, y = predicted)) +
    geom_ribbon(data = model3pred, aes(x = x, ymin = conf.low, ymax = conf.high,
        group = group), alpha=0.05, fill = "green") +
    scale_fill_manual(values = colorvec) +
    labs(fill = "Clean"))

remove(fhtnmdf_GCN)
remove(model1)
remove(model2)
```

Firstly, a scatterplot was made showing no obvious collinearity between Nitrate, Phosphate, or cleanliness levels with HSI, while simultaneously reporting the expected negative correlation between clean water and both Nitrate and Phosphate levels. In addition, no obvious collinearity was shown between *T. cristatus* presence (`GCN_Status_binary`) Phosphate or cleanliness levels, however the impact of HSI and Nitrate warranted further investigation.

The generalised linear model (GLM) is a flexible generalisation of ordinary linear regression that allows for response variables that have error distribution models other than a normal distribution, and as such, is a way of unifying various statistical models (Fox. 2003). Here, it is used in the lead up for a Chi-square test for

goodness of fit, deciding whether there is any difference between the observed (experimental) value and the expected (theoretical) value. As shown in Table 1., only the variable HSI is determined to be a significant influence on *T. cristatus* presence (X^2=251, df= -1, p= 0.5239, dev= 0.055).

Table 1. Reproduced MASS::dropterm table showing Chi-square test for goodness of fit. *T. cristatus* presence was tested against Nitrate, Phosphate, cleanliness, and HSI value, however only HSI is determined to be a significant influence (X^2=251, df= -1, p= 0.5239, dev= 0.055).

	Df	Deviance	AIC	LRT	Pr(Chi)
None		235.80	245.80		
N	1	235.84	243.84	0.0371	0.8472855
P	1	236.16	244.16	0.3595	0.5487746
Clean	1	236.59	244.59	0.7904	0.3739674
HSI_val	1	249.77	257.77	13.9646	0.0001863***

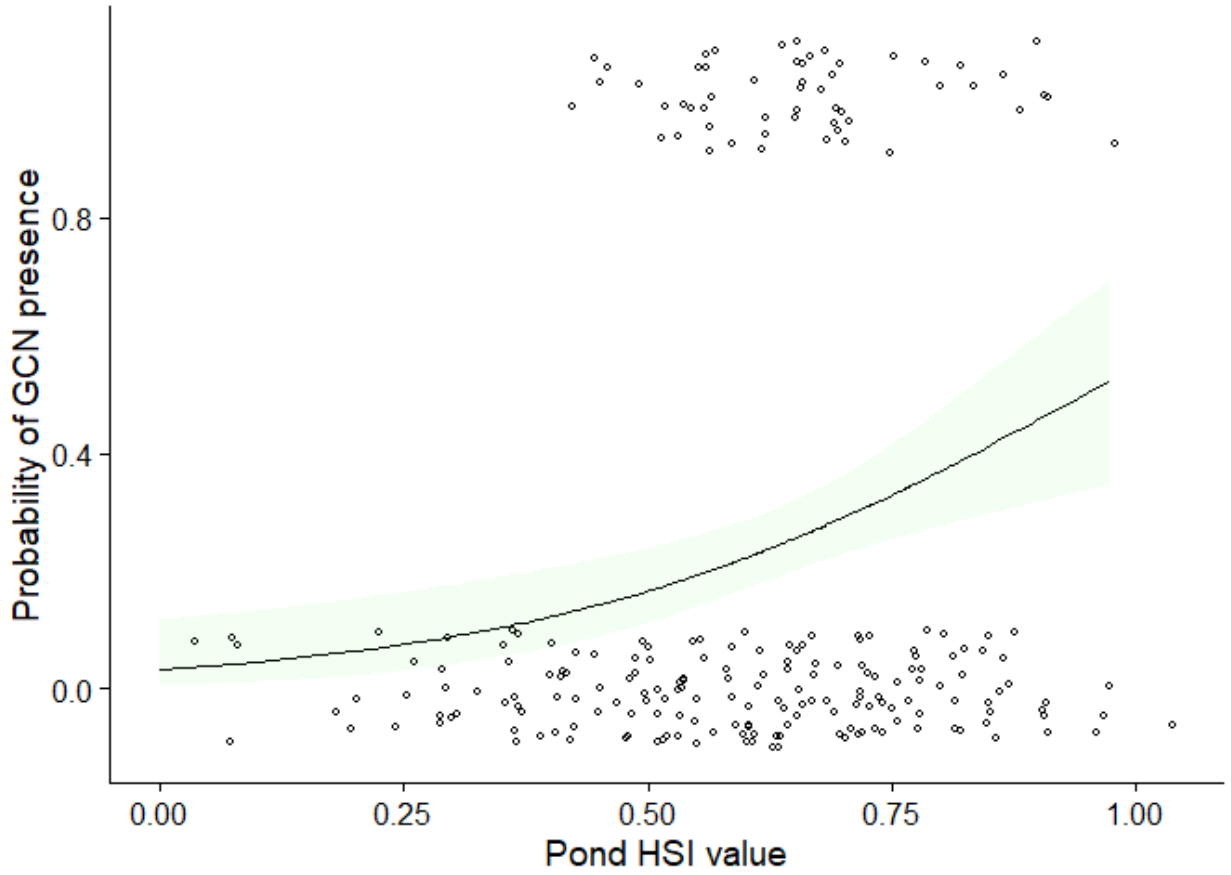


Figure 6: (Produced in Chunk 10).The parametrised equation Logit transformation converts the output of the paramatised equation (shown in-text) to a number between 0 (Absent) and 1 (Present). This number represents a factor of probability where the higher the `HSI_val` value, the more likely *T. cristatus* is present. When the HSI value is at zero, probability of finding *T. cristatus* eDNA is at 0.03 (3%), increasing where at an HSI value of 0.97, the probability is up to ~0.52 (~52%).

To determine the extent of this influence, a plot was made to map the predicted probabilistic outcome of

this model (Figure 6.). The parameterised equation:

$$y = \{^{1=\beta_0+\beta_1 x+\varepsilon>0}_{0=else}$$

Where β_0 is the intercept (-3.40), $\beta_1 x$ is the regression coefficient (3.59) multiplied by a value of the predictor (HSI_val), and ε indicates exponential function. Therefore the final equation is: $y = \{^{1=-3.40+3.59x+\varepsilon>0}_{0=else}$. Logit transformation converts the output of this equation to a number between 0 (Absent) and 1 (Present). This number represents a factor of probability where the higher the `HSI_val` value, the more likely *T. cristatus* is present, represented in the graph of Figure 6.

4.11 Chunk 11:- Shepard Stressplot and NMDS

```
options(tinytex.verbose = TRUE)

FHTfilter <- fhtwild %>%
  filter(nm_Kit_ID != "FHTpc") %>%
  mutate(
  mbsum = select(., Acti_Angu_Angu_Anguangu:Amph_Caud_Sala_Tritcris &
        !(Acti_Cypr_Cypr_Pimeprom)) %>%
                      #_Pimeprom had no metabarcoding data but not filtered out below
          rowSums()
) %>%

  filter(mbsum != 0) %>%  #gets rid of data rows where no metabarcoding data was found

  select(-Aves_Acci_Acci_Butebute:-Mamm_Rode_Sciu_Sciucaro)
     #selects Actinopterygii and Amphibia species only
ID_Environment <- FHTfilter %>%
select(-mb_replicate:-GCN_Negative_Square,
      -Acti_Angu_Angu_Anguangu:-Amph_Caud_Sala_Tritcris)

ID_species <- FHTfilter %>%
select((Acti_Angu_Angu_Anguangu:Amph_Caud_Sala_Tritcris))


n_total_sample<-FHTfilter %>% #n of samples used
group_by(GCN_Binary) %>%
count()
(n_total_sample)     #total n = 171, Inconclusive = 2, Negative = 114, Positive = 55
```

```
## # A tibble: 3 x 2
## # Groups:   GCN_Binary [3]
##   GCN_Binary       n
##   <chr>        <int>
## 1 Inconclusive     2
## 2 Negative       114
## 3 Positive        55
```

```
remove(n_total_sample)

rowSums(ID_species)
```

```
##   [1]    323 289597  25582 259948 166616  23152   3800   1108  75018  94457
##  [11]  25653  23817 185199   4202 319062 480188 757492   3672    719  41057
##  [21] 129406 165233 116906 382059  57918 107221 168798 242215  93965  84713
##  [31]  68002  52259  18376  53764  42708 110198 240007 115274 290244   7720
##  [41] 264342 340093   6200 480571  31705  10345 392235 331314 734607   1385
##  [51]   2278 268924 144813 859562 126143  51328  49499 110430 257053  23581
##  [61]  26242  88612 322766 118278  49018 104922  72339  61759 134952  75532
##  [71]  39406 221828    969  49068  30022   8138    890   4603 113609    518
##  [81] 535562 290439 169364  18807  34213 343870  32349  67619 149021    706
##  [91]  23213 276258  17437 107254  60370 811608 197332 156814  63864   1101
## [101]   8889 158934  21100  22825  26212  54820   7367   1403  77817  10544
## [111]    769  65523   3646  70620  18158 280769  97756  12879 195668 229556
## [121] 293127  70472 302110 398943  44750    343  19575 232327 347878  48347
## [131]  79672 252961 162142  97522  98508    852   8383 391955 306708 182914
## [141] 232268 436830 211935  77181   8067 473426  12251 949763 427183 183887
## [151]  26099  58969  99825  25574   4780   1452  11687  41500  32651  42540
## [161]  47923 250157  86500 122709  38415 387166 586021  22915 136710  77211
## [171]  59463
```

```
colSums(ID_species)
```

```
## Acti_Angu_Angu_Anguangu Acti_Cypr_Cobi_Cobitean Acti_Cypr_Cypr_Abrabram
##                    9309                   24719                  139819
##  Acti_Cypr_Cypr_Albuunk  Acti_Cypr_Cypr_Blicunk Acti_Cypr_Cypr_Caracara
##                     111                    2936                 2154395
## Acti_Cypr_Cypr_Cyprcarp Acti_Cypr_Cypr_Gobigobi Acti_Cypr_Cypr_Leucidus
##                 1956949                   96106                  181925
## Acti_Cypr_Cypr_Phoxphox Acti_Cypr_Cypr_Pimeprom Acti_Cypr_Cypr_Rutiruti
##                   18011                       0                 1982181
## Acti_Cypr_Cypr_Tinctinc Acti_Cypr_Nema_Barbbarb Acti_Esoc_Esoc_Esoxluci
##                  249234                   20058                  152256
## Acti_Gast_Gast_Gastacul Acti_Perc_Perc_Percluci Amph_Anur_Bufo_Bufobufo
##                 2991667                  949128                 3410007
## Amph_Anur_Rani_Ranatemp Amph_Caud_Sala_Lisshelv Amph_Caud_Sala_Lissvulg
##                 2140126                  386007                 4858726
## Amph_Caud_Sala_Tritcris
##                 2848860
```

```
fhtnmdf.jmds <- metaMDS(ID_species, dist= "bray", binary=TRUE, k = 4, try = 40)
```

```
## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.04039964
## Run 1 stress 0.0451876
## Run 2 stress 0.03796135
## ... New best solution
## ... Procrustes: rmse 0.02624255  max resid 0.1109795
```

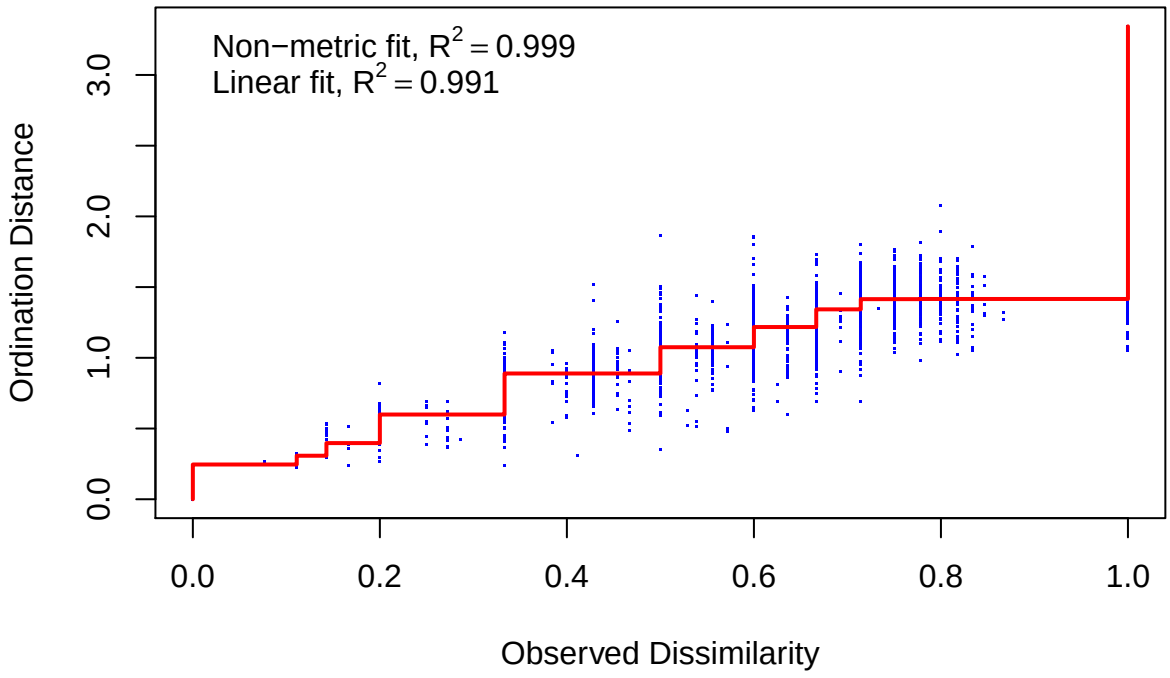
```
## Run 3 stress 0.03757825
## ... New best solution
## ... Procrustes: rmse 0.0116734  max resid 0.05276316
## Run 4 stress 0.04473537
## Run 5 stress 0.03885195
## Run 6 stress 0.03889592
## Run 7 stress 0.04246034
## Run 8 stress 0.04142882
## Run 9 stress 0.0434025
## Run 10 stress 0.04124443
## Run 11 stress 0.0384104
## Run 12 stress 0.04373736
## Run 13 stress 0.03896232
## Run 14 stress 0.04238198
## Run 15 stress 0.04405211
## Run 16 stress 0.04365794
## Run 17 stress 0.04238952
## Run 18 stress 0.04074836
## Run 19 stress 0.04350517
## Run 20 stress 0.04530475
## Run 21 stress 0.0500919
## Run 22 stress 0.04056987
## Run 23 stress 0.04246473
## Run 24 stress 0.04325155
## Run 25 stress 0.04408587
## Run 26 stress 0.04793906
## Run 27 stress 0.03857196
## Run 28 stress 0.04226681
## Run 29 stress 0.04304348
## Run 30 stress 0.03768988
## ... Procrustes: rmse 0.007478581  max resid 0.04474493
## Run 31 stress 0.04219685
## Run 32 stress 0.042609
## Run 33 stress 0.04197619
## Run 34 stress 0.04174668
## Run 35 stress 0.04046751
## Run 36 stress 0.03887807
## Run 37 stress 0.03998864
## Run 38 stress 0.04018996
## Run 39 stress 0.04187958
## Run 40 stress 0.04102298
## *** No convergence -- monoMDS stopping criteria:
##     40: no. of iterations >= maxit
```

```
fhtnmdf.jmds <- metaMDS(ID_species, dist= "bray", binary=TRUE, k = 4, try = 40,
    previous = fhtnmdf.jmds)  #MDS == MultiDimensionalScaling
```

```
## Square root transformation
## Wisconsin double standardization
## Starting from 4-dimensional configuration
## Run 0 stress 0.03757825
## Run 1 stress 0.04160861
## Run 2 stress 0.0391277
## Run 3 stress 0.04716952
```

```
## Run 4 stress 0.03847478
## Run 5 stress 0.04325476
## Run 6 stress 0.04021081
## Run 7 stress 0.04144925
## Run 8 stress 0.0416063
## Run 9 stress 0.03745158
## ... New best solution
## ... Procrustes: rmse 0.005835773  max resid 0.04654811
## Run 10 stress 0.04167568
## Run 11 stress 0.04280869
## Run 12 stress 0.03762507
## ... Procrustes: rmse 0.005197789  max resid 0.04613682
## Run 13 stress 0.04025895
## Run 14 stress 0.04100997
## Run 15 stress 0.03866807
## Run 16 stress 0.04393943
## Run 17 stress 0.04132717
## Run 18 stress 0.04248849
## Run 19 stress 0.0403764
## Run 20 stress 0.04137229
## Run 21 stress 0.04075655
## Run 22 stress 0.0383341
## Run 23 stress 0.04062944
## Run 24 stress 0.04248827
## Run 25 stress 0.03991538
## Run 26 stress 0.0479771
## Run 27 stress 0.04279163
## Run 28 stress 0.03998966
## Run 29 stress 0.04069077
## Run 30 stress 0.03922899
## Run 31 stress 0.03862731
## Run 32 stress 0.04520445
## Run 33 stress 0.04235781
## Run 34 stress 0.03836237
## Run 35 stress 0.04145324
## Run 36 stress 0.04328889
## Run 37 stress 0.04258132
## Run 38 stress 0.04208271
## Run 39 stress 0.03867983
## Run 40 stress 0.04030732
## *** No convergence -- monoMDS stopping criteria:
##     35: no. of iterations >= maxit
##      5: stress ratio > sratmax
```

```
stressplot(fhtnmdf.jmds)  # used to visualise the Shepard stress plot.
```




```
fhtnmdf_HSIrotate.jmds <- with(ID_Environment, MDSrotate(fhtnmdf.jmds, HSI_val))

fhtnmdf.jmds <- fhtnmdf_HSIrotate.jmds

envfit(fhtnmdf.jmds ~ HSI_val + N + P + Clean, data = ID_Environment,
       perm = 999, na.rm = TRUE)
```

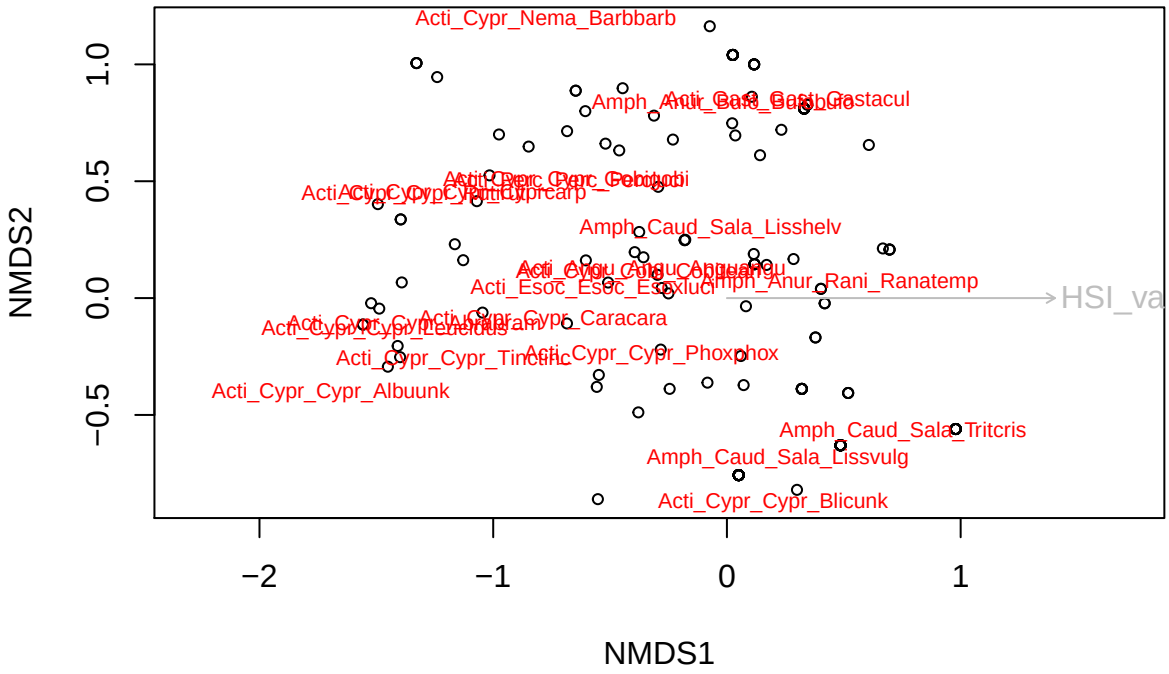
```
##
## ***VECTORS
##
##            NMDS1    NMDS2     r2 Pr(>r)
## HSI_val  1.00000  0.00000 0.3173  0.001 ***
## N        0.99306  0.11757 0.0013  0.899
## P       -0.72110  0.69283 0.0053  0.613
## Clean   -0.79103 -0.61177 0.0055  0.634
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Permutation: free
## Number of permutations: 999
```

```
fhtnmdf.jmds.envfit <- envfit(fhtnmdf.jmds ~ HSI_val, data = ID_Environment,
                              perm = 999, na.rm = TRUE)
fhtnmdf.jmds.envfit
```

```
##
```

```
## ***VECTORS
##
##                             NMDS1                    NMDS2     r2 Pr(>r)
## HSI_val 1.00000000000000000000 0.00000000000000045543 0.3173  0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Permutation: free
## Number of permutations: 999
```

```
plot(fhtnmdf.jmds, display = "site",
   xlab = "NMDS1",
   ylab = "NMDS2")
text(fhtnmdf.jmds, display = "spec", cex = 0.7, col = "red")
plot(fhtnmdf.jmds.envfit, col = "grey")
```



Splitting the data frame into two sets, one for the environmental variables (`ID_Environment`), and another for the *Actinopterygii* and *Amphibia* species present (`ID_species`), and getting rid of data rows in both where no metabarcoding data was found, a meta Multi-Dimensional Scaling (`metaMDS`) function could be performed using 40 random starting points to calculate the Shepard stress plot between Ordination Distance and Observed Dissimilarity (Non-metric fit $R^2 = 0.999$) and develop a non-metric Multi-Dimensional Scaling (NMDS) based on 999 permutations to arrive with the best fitting model, replicated in Figure 7. Full list of detected species can be found in Appendix 1, while variables within `ID_Environment` are listed in Appendix 2.

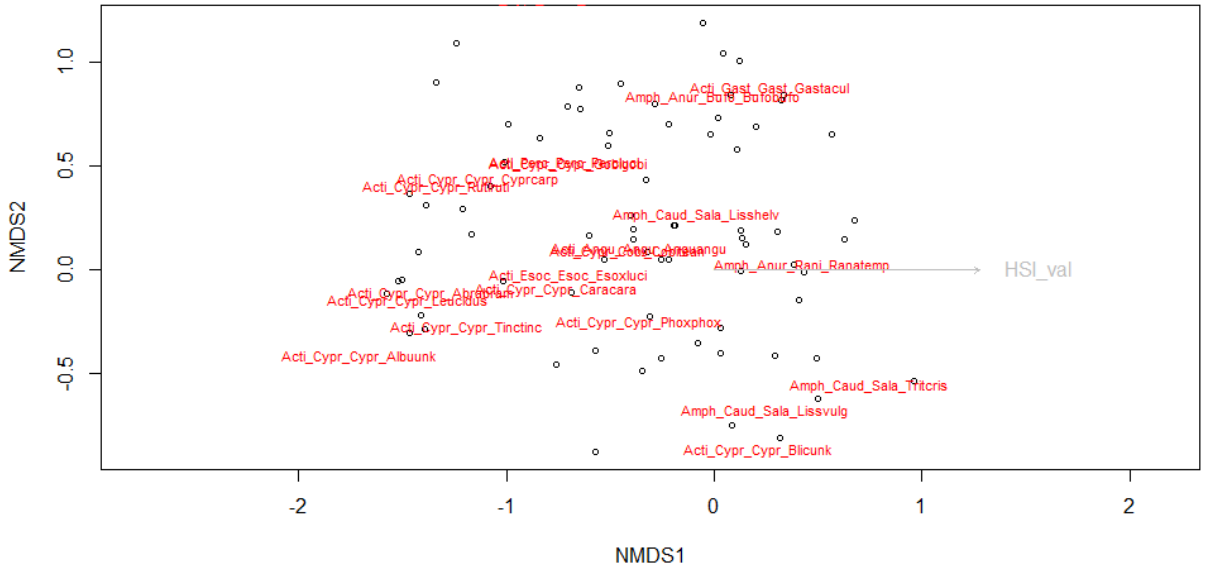


Figure 7: (Produced in Chunk 11). Non-metric Multi-Dimensional Scaling (NMDS), plotting *Actinopterygii* and *Amphibia* species, found within eDNA UK pond samples, in accordance to multivariate co-habitation.

5 Discussion

5.1 Analysis of Results and Implications of the Study

Once the HSI values (`HSI_val`) were calculated, it was compared with the pre-calculated values (`DIMITRIOS_HSI`) to test for correlation (Figure 1.), where a moderately positive relationship between pre-calculated and coded HSI was found. This difference indicates that the HSI was calculated differently both times, however, as there were no indications as to how the `DIMITRIOS_HSI` was calculated from the individual SI's, the coded values were used for the rest of the study. Part of this difference may be explained due to miscalculated values in the `DIMITRIOS_HSI` data set where fields such as Pond area are given a zero value but still resulted in a positive result in `DIMITRIOS_HSI` (ID: FHT736, DIMITRIOS_HSI: 0.718 (3dp); ID: FHT742, DIMITRIOS_HSI: 0.233 (3dp)).

From these calculated HSI values, compared with results concerning *T. cristatus* presence a boxplot was made (Figure 2). The overlapping error bars show that the samples were taken from a wide variety of ponds, and while negative samples were found throughout the surveys, positive and inconclusive results seemed to be limited to anywhere that HSI is >0.4.

To reiterate what was said before, clean values were assigned per the Clean Water for Wildlife Technical Manual (Biggs, *et al.* 2016) where **(0)** is classified as High or very high levels of pollution where phosphate >0.1 mgL^{-1}, nitrate >1 mgL^{-1}; **(1)** shows some evidence of pollution where phosphate 0.05-0.1 mgL^{-1}, nitrate 0.5-1 mgL^{-1}; and **(2)** signifying clean water, with phosphate <0.05 mgL^{-1}, and nitrate <0.5 mgL^{-1}. The totals from Figure 3 show that while many samples were collected from ponds with minor (113) or no (105) pollution, very few samples were collected from highly polluted ponds (17), which may be due from either the general area the samples are from or due to selective processes when collecting data.

The nitrate levels (Figure 4.) show that samples with low levels (<0.5 mgL^{-1}) are far more common than those of medium or high nitrate levels (211 compared to 9 and 15 respectively), which is reflected in those that have positive results (52 compared to 1 and 2 in same respective order). Similarly, phosphate levels

(Figure 5.) show that samples with low levels (<0.05 mgL^{-1}) are more common than those of medium or high nitrate levels, though to a lesser extent (118 compared to 33 and 84 respectively). Positive samples show low (<0.05 mgL^{-1}) phosphate levels (29) at a similar quantity to high (>0.1 mgL^{-1}) levels (22) but much lower quantities (4) at medium phosphate levels (0.05-0.1 mgL^{-1}). For better accuracy in how both nitrate and phosphate levels impact the presence of *T. cristatus*, more samples should be collected for analysis.

Continuing the study with acknowledgement of data limitations, testing Nitrate, Phosphate, Clean, and HSI levels (Table 1.), show no significant detectable difference existing between the tested groups except for HSI_val.

The logit transformation converts the output of the equation $y = \{^{1=-3.40+3.59x+\varepsilon>0}_{0=else}$ to a number between 0 (Absent) and 1 (Present). This number represents a factor of probability where the higher the `HSI_val` value, the more likely *T. cristatus* is present (Figure 6). When the HSI value is at zero, probability of finding *T. cristatus* eDNA is at 0.03 (3%), increasing gradually, where at an HSI value of 0.97, the probability is up to ~0.52 (~52%), displaying how sites with optimal HSI values are significantly more likely to contain *T. cristatus* eDNA, but is not a conclusive guarantee. This backs up initial claims (ARG. 2010) that higher HSI scores are more likely to support these newt habitats but are not sufficient to substitute a newt survey.

The function `metaMDS` tries to find a stable solution using several random starts and standardizes the scaling in the result, so that the configurations are easier to interpret, and adds species scores to the site ordination (Holland. 2008). As a result, a final analysis was made concerning different influences on habitats, as a Non-metric Multi-Dimensional Scaling (NMDS) plot displaying *Actinopterygii* and *Amphibia* species found within pond samples, dependent upon multiple variables (Appendix 2) and the impact on co-habitation (Figure 7). Out of the different factors that would affect the results of this figure, the separation of fish and amphibian species is likely due in-part to the seventh factor of calculating the HSI (`fish_hsi`). This factor is put in place as fish are known to predate on eggs of amphibian species (Hecnar and M'Closkey. 1997; Winandy, *et al.* 2015; Murray, *et al.* 2004). The figure would suggest, for example, that an unknown *Blicca* species can live with *T. cristatus*, given their proximity to each other, compared to other fish species known to predate on *T. cristatus* such as the stickleback species *Gasterosteus aculeatus* (Jarvis. 2010).

5.2 Limitations to study

It is known that fish species such the stone loach (*Barbatula barbatula*) is sensitive to the oxygen level and the stream velocity of the water (something that can be obstructed by influences such as aquatic plants) (Pont *et al.* 2005; Gerritsen. 2011), variables that were not considered during data collection. While it is believed that stone loach population rapidly recovers from these environmental obstructions (Gerritsen. 2011), quantitative studies revealing information on how this effects amphibians such as *T. cristatus* has not been made (Gustafson. 2011; Moya, *et al* 2011). In addition to this data being collected, as pointed out before, more data for various Nitrate and Phosphate levels should be available for a more distributed sample set and more accurate data analysis.

Another limitation may result from an MDS ordination such as the one performed in `Chunk 11`, as it is not a unique solution due to permutations stopping after a pre-specified number of attempts. A subsequent MDS analysis on the same set of data and following the same methodology may result in a somewhat different ordination, however, the vast quantity of attempts used will help to mitigate the impact this has.

When it comes to improvements in the code, within `Chunk 4` certain data is eliminated because of its ambiguous formatting within `HSI8_Pond_count` and `HSI10_Macrophytes` (a total of 43 data rows). Ideally these would be mutated into a value that can be understood as a numerical. Also within `Chunk 4`, any data where `nm_Kit_ID` had two or more inputs were removed from the data set (a total of 24 data rows, or 12 samples). Not only would it be ideal for the last run of each entry to be kept, the elimination of these data rows was done in a way where they would have to be individually removed, rather than an automatic process. This means that the process would have been prone to error without careful diligence.

5.3 Conclusion

Despite the limitations this study has, evidence has been given to support the claim that HSI value impacts the presence of *T. cristatus*, effectively allowing rejection of the null hypothesis. For further studies, more samples from various Nitrate and Phosphate levels should be available for a fairer analysis to be made. Additionally, more variables that could impact *T. cristatus* should be studied, such as the oxygen level and the stream velocity of the water.

---

6 References

Adesuyi, A.A., Nnodu, V.C., Njoku, K.L. and Jolaoso, A., 2015. Nitrate and Phosphate Pollution in Surface Water of Nwaja Creek, Port Harcourt, Niger Delta, Nigeria. International Journal of Geology, Agriculture and Environmental Sciences, 3(5), pp.14-20.

Allison, P.D. and Waterman, R.P., 2002. Fixed–effects negative binomial regression models. Sociological methodology, 32(1), pp.247-265.

Allison, P.D., 1996. Fixed-effects partial likelihood for repeated events. Sociological Methods & Research, 25(2), pp.207-222.

ARG, U., 2010. Great Crested Newt Habitat Suitability Index. ARG UK Advice Note 5.

Atkins, W.R.G., 1923. The phosphate content of fresh and salt waters in its relationship to the growth of the algal plankton. Journal of the Marine Biological Association of the United Kingdom, 13(1), pp.119-150.

Bhatty, R.S., 1964. Influence of nitrogen fertilization on the yield, protein, and oil content of two varieties of rape. Canadian Journal of Plant Science, 44(2), pp.215-217.

Biggs, J., Ewald, N., Valentini, A., Gaboriaud, C., Dejean, T., Griffiths, R.A., Foster, J., Wilkinson, J.W., Arnell, A., Brotherton, P. and Williams, P., 2015. Using eDNA to develop a national citizen science-based monitoring programme for the great crested newt (Triturus cristatus). Biological Conservation, 183, pp.1928.

Biggs, J., McGoff, E., Ewald, N., Williams, P., Dunn, F. and Nicolet, P. 2016. Clean Water for Wildlife technical manual. Evaluating PackTest nitrate and phosphate test kits to find clean water and assess the extent of pollution. Freshwater Habitats Trust, Oxford.

Camargo, J.A. and Alonso, Á., 2006. Ecological and toxicological effects of inorganic nitrogen pollution in aquatic ecosystems: a global assessment. Environment international, 32(6), pp.831-849.

Deiner, K., Bik, H.M., Mächler, E., Seymour, M., Lacoursière-Roussel, A., Altermatt, F., Creer, S., Bista, I., Lodge, D.M., De Vere, N. and Pfrender, M.E., 2017. Environmental DNA metabarcoding: Transforming how we survey animal and plant communities. Molecular ecology, 26(21), pp.5872-5895.

Fox, J., 2003. Effect displays in R for generalised linear models. Journal of statistical software, 8(15), pp.1-27. Fried, S., Mackie, B. and Nothwehr, E., 2003. Nitrate and phosphate levels positively affect the growth of algae species found in Perry Pond. Tillers, 4, pp.21-24.

Grasman, J., van Deventer, W.B. and van Laar, V., 2012. Estimation of parameters in a bertalanffy type of temperature dependent growth model using data on juvenile stone loach (Barbatula barbatula). Acta biotheoretica, 60(4), pp.393-405.

Guarda, G., Padovan, S. and Delogu, G., 2004. Grain yield, nitrogen-use efficiency and baking quality of old and modern Italian bread-wheat cultivars grown at different nitrogen levels. European Journal of Agronomy, 21(2), pp.181-192.

Gustafson, D., 2011. Choosing the best of both worlds (Vol. 2011, No. 87).

Harte, J., Ostling, A., Green, J.L. and Kinzig, A., 2004. Biodiversity conservation: Climate change and extinction risk. Nature, 430(6995), p.34.

Hecnar, S.J. and M'Closkey, R.T., 1997. The effects of predatory fish on amphibian species richness and distribution. Biological conservation, 79(2-3), pp.123-131.

Holland, S.M., 2008. Non-metric multidimensional scaling (MDS). Department of Geology, University of Georgia, Athens, Tech. Rep. GA, pp.30602-2501.

Howell, D.C., 2011. Chi-Square Test: Analysis of Contingency Tables.

Jarvis, L.E., 2010. Non-consumptive effects of predatory three-spined sticklebacks (Gasterosteus aculeatus) on great crested newt (Triturus cristatus) embryos. The Herpetological Journal, 20(4), pp.271-275.

Lovatt, C.J., 2001. Properly Timed Soil-applied Nitrogen Fertilizer Increases Yield and Fruit Size ofHass' Avocado. Journal of the American Society for Horticultural Science, 126(5), pp.555-559.

Maron, M., Gordon, A., Mackey, B.G., Possingham, H.P. and Watson, J.E., 2015. Conservation: stop misuse of biodiversity offsets. Nature News, 523(7561), p.401.

McKague, K., Reid, K. and Simpson, H., 2005. Environmental impacts of nitrogen use in agriculture. Ontario, Ministry of Agriculture, Food and Rural Affairs.

Moya, N., Hughes, R.M., Domínguez, E., Gibon, F.M., Goitia, E. and Oberdorff, T., 2011. Macroinvertebrate-based multimetric predictive models for evaluating the human impact on biotic condition of Bolivian streams. Ecological indicators, 11(3), pp.840-847.

Murray, D.L., Roth, J.D. and Wirsing, A.J., 2004. Predation risk avoidance by terrestrial amphibians: the role of prey experience and vulnerability to native and exotic predators. Ethology, 110(8), pp.635-647.

Needham, K., de Vries, F.P., Armsworth, P.R. and Hanley, N., 2019. Designing markets for biodiversity offsets: Lessons from tradable pollution permits. Journal of Applied Ecology, 56(6), pp.1429-1435.

Oldham, R.S., Keeble, J., Swan, M.J.S. and Jeffcote, M., 2000. Evaluating the suitability of habitat for the great crested newt (Triturus cristatus). Herpetological Journal, 10(4), pp.143-156.

Piper, A.M., Batovska, J., Cogan, N.O., Weiss, J., Cunningham, J.P., Rodoni, B.C. and Blacket, M.J., 2019. Prospects and challenges of implementing DNA metabarcoding for high-throughput insect surveillance. GigaScience, 8(8), p.giz092.

Pont, D., Hugueny, B. and Oberdorff, T., 2005. Modelling habitat requirement of European fishes: do species have similar responses to local and regional environmental constraints?. Canadian Journal of Fisheries and Aquatic Sciences, 62(1), pp.163-173.

Quétier, F., Regnery, B. and Levrel, H., 2014. No net loss of biodiversity or paper offsets? A critical review of the French no net loss policy. Environmental Science & Policy, 38, pp.120-131.

Robertson, M.M., 2000. No net loss: wetland restoration and the incomplete capitalization of nature. Antipode, 32(4), pp.463-493.

Tew, T. and Nicolet, P., 2019. District Licensing for Great Crested Newts – A Successful First Year for the South Midlands Scheme, InPractice 103 pp. 28-32

Tew, T., Biggs, J. and Gent, T., 2018. 'District licensing' for great crested newts – delivering a big idea InPractice, 100, pp. 35-39

Walther, G.R., Post, E., Convey, P., Menzel, A., Parmesan, C., Beebee, T.J., Fromentin, J.M., HoeghGuldberg, O. and Bairlein, F., 2002. Ecological responses to recent climate change. Nature, 416(6879), p.389.

Winandy, L., Darnet, E. and Denoël, M., 2015. Amphibians forgo aquatic life in response to alien fish introduction. Animal Behaviour, 109, pp.209-216.

7 Appendix

7.1 List of Present Vertebrate Species

Data frame entry	Class	Order	Family	Binominal name
Acti_Angu_Angu_Anguangu	Actinopterygii	Anguilliformes	Anguillidae	Anguilla anguilla
Acti_Cypr_Cobi_Cobitean	Actinopterygii	Cypriniformes	Cobitoidea	Cobitis taenia
Acti_Cypr_Cypr_Abrabram	Actinopterygii	Cypriniformes	Cyprinidae	Abramis brama
Acti_Cypr_Cypr_Albuunk	Actinopterygii	Cypriniformes	Cyprinidae	Alburnus unknown
Acti_Cypr_Cypr_Blicunk	Actinopterygii	Cypriniformes	Cyprinidae	Blicca unknown
Acti_Cypr_Cypr_Caracara	Actinopterygii	Cypriniformes	Cyprinidae	Carassius carassius
Acti_Cypr_Cypr_Cyprcarp	Actinopterygii	Cypriniformes	Cyprinidae	Cyprinus carpio
Acti_Cypr_Cypr_Gobigobi	Actinopterygii	Cypriniformes	Cyprinidae	Gobio gobio
Acti_Cypr_Cypr_Leucidus	Actinopterygii	Cypriniformes	Cyprinidae	Leuciscus idus
Acti_Cypr_Cypr_Phoxphox	Actinopterygii	Cypriniformes	Cyprinidae	Phoxinus phoxinus
Acti_Cypr_Cypr_Pimeprom	Actinopterygii	Cypriniformes	Cyprinidae	Pimephales promelas
Acti_Cypr_Cypr_Rutiruti	Actinopterygii	Cypriniformes	Cyprinidae	Rutilus rutilus
Acti_Cypr_Cypr_Tinctinc	Actinopterygii	Cypriniformes	Cyprinidae	Tinca tinca
Acti_Cypr_Nema_Barbbarb	Actinopterygii	Cypriniformes	Nemacheilidae	Barbatula barbatula
Acti_Esoc_Esoc_Esoxluci	Actinopterygii	Esociformes	Esocidae	Esox lucius
Acti_Gast_Gast_Gastacul	Actinopterygii	Gasterosteiformes	Gasterosteidae	Gasterosteus aculeatus
Acti_Perc_Perc_Percluci	Actinopterygii	Percopsiformes	percopsidae	Percopsis (luci?)
Amph_Anur_Bufo_Bufobufo	Amphibia	Anura	Bufonidae	Bufo bufo
Amph_Anur_Rani_Ranatemp	Amphibia	Anura	Ranidae	Rana temporia
Amph_Caud_Sala_Lisshelv	Amphibia	Caudata	Salamandridae	Lissotriton helveticus
Amph_Caud_Sala_Lissvulg	Amphibia	Caudata	Salamandridae	Lissotriton vulgaris
Amph_Caud_Sala_Tritcris	Amphibia	Caudata	Salamandridae	Triturus cristatus
Aves_Acci_Acci_Butebute	Aves	Accipitriformes	Accipitrimorphae	Buteo buteo
Aves_Anse_Anat_Aixgale	Aves	Anseriformes	Anatidae	Aix galericulata
Aves_Anse_Anat_Anastado	Aves	Anseriformes	Anatidae	Anas tadorna (now T. tadorna)
Aves_Anse_Anat_Anatsp	Aves	Anseriformes	Anatidae	Anatidae sp.
Aves_Char_Lari_Larusp	Aves	Charadriiformes	Laridae	Laridae sp.
Aves_Colu_Colu_Colulivi	Aves	Columbiformes	Columbinae	Columba livia
Aves_Colu_Colu_Colusp	Aves	Columbiformes	Columbinae	Columba sp.
Aves_Gall_Phas_Phascolc	Aves	Galliformes	Phasianidae	Phasianus colchicus
Aves_Grui_Rall_Fuliatra	Aves	Gruiformes	Rallidae	Fulicia atra
Aves_Grui_Rall_Gallchlo	Aves	Gruiformes	Rallidae	Gallinula chloropus
Aves_Pass_Acro_Acroscir	Aves	Passeriformes	Acrocephalidae	Acrocephalus scirpaceus
Aves_Pass_Corv_Garrglan	Aves	Passeriformes	Corvoidae	Garrulus glandarius
Aves_Pass_Corv_Picapica	Aves	Passeriformes	Corvoidae	Pica pica

Data frame entry	Class	Order	Family	Binominal name
Aves_Pass_Musc_Eritrube	Aves	Passeriformes	Muscicapidae	Erithicus rubecula
Aves_Pass_Musc_Turdsp	Aves	Passeriformes	Muscicapidae	Turdus sp.
Aves_Pass_Panu_Panubiar	Aves	Passeriformes	Panuridae	Panurus biarmicus
Aves_Pass_Pari_Parumajo	Aves	Passeriformes	Paridae	Parus major
Aves_Pass_Pass_Pass1	Aves	Passeriformes	Passeridae	Passer 1
Aves_Pass_Pass_Pass2	Aves	Passeriformes	Passeridae	Passer 2
Aves_Pass_Prun_Prunmodu	Aves	Passeriformes	Prunellidae	Prunella modularis
Aves_Pass_Stur_Sturvulg	Aves	Passeriformes	Sturnidae	Sturnus vulgaris
Aves_Pass_Sylv_Sylvatri	Aves	Passeriformes	Sylviidae	Sylvia atricapilla
Aves_Pass_Trog_Trogtrog	Aves	Passeriformes	Troglodytidae	Troglodytes troglodytes
Aves_Pele_Arde_Ardecine	Aves	Pelicaniformes	Ardedae	Ardea cinerea
Aves_Pici_Pici_Dendmajo	Aves	Piciformes	Picidae	Dendrocopos major
Aves_Pici_Pici_Picuviri	Aves	Piciformes	Picidae	Picus viridis
Aves_Stri_Stri_Strisp	Aves	Strigiformes	Strigidae	Strix sp.
Mamm_Arti_Cerv_Caprcapr	Mammalia	Artiodactyla	Cervidae	Capreolus capreolus
Mamm_Arti_Cerv_Cervelap	Mammalia	Artiodactyla	Cervidae	Cervini elaphus
Mamm_Arti_Cerv_Damadama	Mammalia	Artiodactyla	Cervidae	Dama dama
Mamm_Arti_Cerv_Hydriner	Mammalia	Artiodactyla	Cervidae	Hydropotes inermis
Mamm_Arti_Cerv_Muntreev	Mammalia	Artiodactyla	Cervidae	Muntiacus Reevisi
Mamm_Carn_Cani_Vulpvulp	Mammalia	Carnivora	Canidae	Vulpes vulpes
Mamm_Carn_Must_Lutrlutr	Mammalia	Carnivora	Mustelidae	Lutra lutra
Mamm_Carn_Must_Melemele	Mammalia	Carnivora	Mustelidae	Meles Meles
Mamm_Chir_Vesp_Myotnatt	Mammalia	Chiroptera	Vespertilionidae	Myotis nattereri
Mamm_Chir_Vesp_Pipiaust	Mammalia	Chiroptera	Vespertilionidae	Pipistrellus (aust?)
Mamm_Euli_Sori_Sorearan	Mammalia	Eulipotyphla	Soricidae	Sorex araneus
Mamm_Euli_Talp_Talpeuro	Mammalia	Eulipotyphla	Talpidae	Talpa europaea
Mamm_Rode_Cric_Arviterr	Mammalia	Rodentia	Cricetidae	Arvicola terrestris
Mamm_Rode_Cric_Micragre	Mammalia	Rodentia	Cricetidae	Microtis agrestis
Mamm_Rode_Cric_Myodglar	Mammalia	Rodentia	Cricetidae	Myodes glareolus
Mamm_Rode_Muri_Apodsylv	Mammalia	Rodentia	Muridae	Apodemus sylvaticus
Mamm_Rode_Muri_Rattnorv	Mammalia	Rodentia	Muridae	Rattus norvegicus
Mamm_Rode_Sciu_Sciucaro	Mammalia	Rodentia	Sciuridae	Sciurus carolinensis

7.2 List of Variables in `ID_Environment`

Variable Name	Function
nm_Kit_ID	Unique nature metrics ID
1_km_square	1 km area where pond is found

Variable Name	Function
Grid_reference	Grid reference to where pond is found
Easting	Easting Coordinates (used with Northing)
Northing	Northing Coordinates (used with Easting)
eDNA_score	Number of samples positive for GCN
GCN_test_result	Number of samples positive for GCN
Positive_no_kit	GCN present?
Kit_negative	GCN present?
Newt_Positive_sites	GCN present?
Newt_Negative_sites	GCN present?
HSI1_Pond_location	HSI calulation
HSI2_Pond_area	HSI calulation
HSI3_Pond_drying	HSI calulation
HSI4_Water_quality	HSI calulation
HSI5_Shade	HSI calulation
HSI6_Waterfowl	HSI calulation
HSI7_Fish	HSI calulation
HSI8_Pond_count	HSI calulation
HSI9_Terrestrial_habitat	HSI calulation
HSI10_Macrophytes	HSI calulation
DIMITRIOS_HSI	Final premade HSI calulation
Inflow_present	Flow disturbance. (Y=inflow present. N=no inflow present. ?/NA= Data not collected)
Outflow_present	Flow disturbance. (Y= Outflow flow present. N=no outflow present. ?/NA= Data not collected)
Nitrate	Rough data (with mistakes in format)
Phosphate	rough data (with mistakes in format)
Clean	Clean data
N	Clean data
P	Clean data
GCN_Binary	GCN Presence
pondarea_hsi	HSI calulation
ponddry_hsi	HSI calulation
waterquality_hsi	HSI calulation
shade_hsi	HSI calulation
waterfowl_hsi	HSI calulation
fish_hsi	HSI calulation
pondcount_hsi	
terrestrial_habitat_hsi	HSI calulation
macrophytes_hsi	HSI calulation
HSI_val	Final HSI calulation
N_level	Clean data in accordance to Freshwater Habitats Trust levels
P_level	Clean data in accordance to Freshwater Habitats Trust levels
mbsum	Sum of metabarcoding data