Bharat Thakkar
Email address: b.thakkar.22@abdn.ac.uk
Student ID: 52214197

Exploring features and packages of R for Image Simulation, Classification, Sea Level Change Modelling, and 3D Visualisation

1495 UNIVERSITY OF ABERDEEN
OILTHIGH OBAR DHEATHAIN

CONTENTS

ABSTRACT

R is a language for statistical programming and analysis. R also offers a range of tools for image and data processing with its elaborate set of libraries available for spatial analysis and geospatial processing. These libraries are explored in this report by performing spatial analysis tasks such as mapping, plotting, classification, reclassification, digital terrain model processing, sea level change modelling, 3D modelling of spatial data. While performing these tasks, a simplistic yet powerful capabilities of the language are explored for spatial data retrieval, manipulation, processing, and translation of data into an image using base features and spatial libraries. Further, spatial tools are used for performing raster calculation on a DTM to visualise sea level changes and raster manipulation features to change the spatial resolution for 3D plotting are also explored.

INTRODUCTION

Spatial packages when combined with base features of R makes it a powerful language to perform image simulation, classification, sea level change modelling, and 3D Modelling. Using spatial packages and base features of R, two tasks will be performed and discussed in this report.

In the first task, image simulation, mapping, classification, and reclassification will be performed. First, base features of R will be used to create and manipulate vectors and matrices to represent to represent an image. Importantly, matrix created will be mapped and plotted by using the raster library (a spatial library of R). Subsequently, the plotted image will then be classified to represent reduced set of colours. Next, another process for reclassification will be applied to further reduce the number of colours. Finally, the reclassified image will be plotted to show the outcome of the task.

In another task, Findhorn Bay DTM will be accessed and plotted along with sea level change simulation, and 3D modelling of Happisburgh will be also performed. First, raster library will be used to read and plot the DTM. Next, reclassification of the DTM will be performed to identify the elevation range, introduce contour lines, and calculation will be performed to simulate and visualise the sea level change of the Findhorn Bay. Finally, 3D modelling of Happisburgh DTM will be performed to visualise the details of the raster in three dimensions.

METHODOLOGY

In this section, tasks are carried out to explore features of R by simulating a matrix data into an image, mapping, plotting, classification/reclassification of the image as well as accessing and plotting a DTM, simulating sea level changes and 3D modelling of a DTM.

1: Image simulation of a matrix, mapping, classification, and reclassification

Image simulation with matrix:

For image simulation, raw data needs to be created which can be visually displayed with the RGB colour having values ranging up to 255. To achieve this, first a list object is created to contain other objects such as vectors and matrices as shown in below snippet of code. The snippet shows sequence of values generated for X axis, Y axis, and a matrix with random values generated using uniform distribution function for rows and columns to represent an image (An Introduction to R, 2023).

```
first.img=list()
first.img$x=seq(1,by=1,len=255)
first.img$y=seq(1,by=1,len=255)
first.img$z=matrix(runif(255*255),255,255)
```

Next, the matrix is looped through and edited to populate a number at specific interval to create different sections in the image with the code as shown below.

```
for (i in 1:255){
  for (j in 1:255){
    if ((i %% 50 == 0) || (j %% 50 == 0)){
      first.img$z[i,j] <- 1
    }
  }
}
```

Refer the results section to view the output produced from this step.

Mapping the simulated data:

To map the output produced in previous step, the matrix first.img needs to be converted into a raster layer. To do this, raster package is installed and loaded into the memory as shown below (Hijmans, 2012).

```
install.packages("raster")
library(raster)
```

Once raster package is loaded, raster function can then be called with first.img as a parameter to create a raster layer object (Hijmans, 2012), and then plot function is used to plot the raster as shown below.

```
raster1 <- raster(first.img)
plot(raster1, main="First image plotted as Raster from matrix")
```

Refer the results section for the output produced from this step.

Classify the image into 10 classes:

The matrix created in step 1 has more than 65,000 different decimal values ranging from 0.0000 to 1.0000. To classify these values into 10 different classes, all the values of the matrix are rounded to 1 digit after decimal place with the below code.

```
class.img <- first.img
class.img$z <- round(first.img$z, 1)
```

Next, the classified image raster (Hijmans, 2012) is plotted to confirm the change to have reduced number of colours with the below code snippet.

```
r2 <- raster(class.img)
plot(r2, main="Classified image plotted as Raster." ,col=topo.colors(10))
```

Refer the result section for the output of this step.

Reclassify the image:

To reclassify the image further, ten categories of values created earlier are re-categorised again. To achieve this, first the image matrix is looped through, and values beyond certain range are reassigned to different values as shown in below code (An Introduction to R, 2023). For instance, range of values beyond 0.5 are now reassigned to lower values between 0.0 and 0.5 as shown.

```
reclass.img <- class.img
for (i in 1:255){
  for (j in 1:255){
    if (class.img$z[i, j] == 0.6) {
      reclass.img$z[i, j] <- 0.05
    } else if (class.img$z[i,j] == 0.7) {
      reclass.img$z[i,j] <- 0.15
    } else if (class.img$z[i,j] == 0.8) {
      reclass.img$z[i,j] <- 0.25
    } else if (class.img$z[i,j] == 0.9) {
      reclass.img$z[i,j] <- 0.35
    } else if (class.img$z[i,j] == 1.0) {
      reclass.img$z[i,j] <- 0.45
    }
  }
}
```

To confirm, the reclassification image is then plotted again as a raster layer (Hijmans, 2012) using below snippet of code.

```
r3 <- raster(reclass.img)
plot(r3, main="Reclassified image plotted as Raster.", col=colorRampPalette(c("blue",
"green"))(255))
```

Refer the result section for the output of this step.

2: Plotting Findhorn Bay DTM, sea level change simulation and 3D Modelling.

Findhorn Bay:

The Findhorn Bay is geographically located between 57° 37' 48.97" N, 3° 38' 15.52" W and 57° 39' 48.85" N, 3° 35' 30.83" W. The Findhorn Bay is known for ecological and sustainable living (Visit Moray Speyside, 2023). On the west of the Findhorn Bay, Culbin Forest is located, and the Findhorn village is located on the east which is accessible via a road travelling alongside the bay (Visit Forres, 2023). Findhorn bay is a popular tourist attraction where many people visit for bird watching, observe wildlife, explore the nature, and enjoy the beaches of the Findhorn Bay (Visit Forres, 2023). The coastal town of Findhorn has population of over 900 people, and it is popular amongst sailors, sport enthusiasts and nature lovers visiting or living around the bay (Visit Moray Speyside, 2023). In the year 1998, the Findhorn Bay was also declared a nature reserve as the bay is a sandy inlet considered important for migrant birds (Visit Forres, 2023).

For this report, a 50m resolution DTM of Findhorn Bay was downloaded from digimap.edina.ac.uk. Reviewing the DTM, it has been noticed that the Findhorn is situated in very low-lying area with some part of the land happens to be below sea level. Furthermore, part of the land immediately surrounding the bay ranges from -1.6 metres below the sea level to less than 30 metres above the sea level, while certain areas farther from the bay are more than 60 metres above the sea level.

Reading the DTM using R:

The downloaded DTM was in ASC format. ASC format is ESRI Ascii grid data format file which contains height or elevation data for every pixel representing 50m resolution.

To read the ASC file, first “raster” package is required to be installed and loaded into memory (Hijmans, 2012) using following commands.

```
install.packages("raster")
library(raster)
```

Next, the directory is set where the file is located using setwd() command and then raster() function is invoked from the raster package to load the DTM into memory (Hijmans, 2012) as shown in below snippet of code.

```
setwd("G:/GIS/GG5567/Assessment2/Download_findhorn-bay_2268664/terrain-50-dtm_5057321/nj")
r3 = raster("NJ06.asc")
```

Plotting the DTM:

After loading the DTM, plot() function is called to plot the loaded raster as shown in below snippet of code (Hijmans, 2012).

```
plot(r3, main="DTM raster of Findhorn Bay", col=topo.colors(32))
```

Refer the results section to view the DTM plotted.

Reclassification for sea level changes:

To plot sea level change, first range of elevation values from the DTM of Findhorn Bay needs to be extracted. Next, the sea level change expected in metres must also be identified based on the elevation range. Using elevation range, contours need to be plotted on the DTM to visualise the levels of change expected to the terrain. Next, calculate function calc() must be applied to the DTM to reclassify the section of the terrain identified to be below the sea level (Hijmans, 2012). Finally, the newly reclassified raster must be plotted to visualise the changes.

The below line of code extracts range of possible elevation values from the Findhorn Bay DTM.

```
raster_range <- range(getValues(r3))
elevation_range <- seq(0, raster_range[2], by=2)
```

In the above code, first all the values of the raster r3 are retrieved using getValues() function (Hijmans, 2012) and then, the values are supplied to range function which outputs two values, minimum and maximum of all the values retrieved. Next, the maximum value is used to create an elevation range vector using sequence function with values between 0 and the maximum value while intermediate values are being incremented by 2. Refer the results section for the values derived for above two variables.

Next, present and new sea level change values are identified to be 0 and 4 metres respectively with below code.

```
present_sea_level_change <- elevation_range[1]
new_sea_level_change <- elevation_range[3]
```

Further, contour lines are plotted using contour() function on to the DTM using present and new sea levels captured in variables earlier (refer the below code).

```
raster::contour(r3, add=TRUE, col="white", lwd=1, levels=present_sea_level_change,
drawlabels=FALSE)
raster::contour(r3, add=TRUE, col="yellow", lwd=1, levels=new_sea_level_change,
drawlabels=FALSE)
```

Refer the results section for the output.

Finally, the DTM with new sea level change is reclassified by applying calc() function (Hijmans, 2012) which has been attached with a custom mathematical function to compare elevation values with new sea level change value and set the elevation to 0 to show the sea level change as shown in below code. The plot() function is used to visualise the sea level change applied.

```
r4 <- calc(r3, fun=function(x){ x[x<=new_sea_level_change] <- 0; return(x)})
plot(r4, main="Reclassified DTM raster of Findhorn Bay", col=topo.colors(32))
```

Refer the results section for the output.

Adding a box, north arrow, and a title:

When a raster is plotted with plot() function, the box is added with coordinates populated with the extent data retrieved from the raster. A north arrow can be added by invoking north() function from the terra package (Terra, 2023) as shown below.

```
plot(r4, main="Reclassified DTM raster of Findhorn Bay", col=topo.colors(32))
install.package("terra")
library("terra")
north(c(309500, 869000), type=3, col="white")
```

Note that the terra package needs to be installed and loaded in memory before invoking the north() function as shown.

3D plotting the Happisburgh_Survey_dsm.tiff:

Before the Happisburgh DTM raster can be plotted in 3D, initial analysis of the raster is performed by loading the raster using the raster() function (Hijmans, 2012) as shown below.

```
setwd("G:/GIS/GG5567/Assessment2")
happisburgh_raster<-raster("Happisburgh_Survey_dsm.tiff")
```

Refer the result section for raster summary. Based on the initial analysis, it was identified that the dimensions of the DTM are considerably large with 16,026 rows and 17,095 columns resulting in 273,964,470 cells. Also, the resolution of the raster is high with 0.0143 metres per pixel (1.43-centimetre per pixel). These two factors make the raster very heavy for 3D plotting process. Hence, the raster is required to be aggregated using aggregate() function

(Hijmans, 2012) to create a new raster with lower but acceptable resolution for 3D plotting. The commands shown below are used for aggregation of the DTM by factor of 10.

```
aggregated_raster <- raster::aggregate(happisburgh_raster, fact=10)
```

The newly aggregated raster now has the resolution of 0.143 metres per pixel (14.3-centimetre per pixel). Further, the aggregated raster can be plotted in 2D using plot() function, and the raster is plotted in 3D with the use of persp3d() function of rgl library (RGL, 2023) as shown below.

```
plot(aggregated_raster,  main="Aggregated DTM raster of Happisburgh",
col=terrain.colors(32))
```

```
Install.packages("rgl")
library(rgl)
persp3d(x = seq(xmin(aggregated_raster), xmax(aggregated_raster), length.out =
nrow(aggregated_matrix)),
      y = seq(ymin(aggregated_raster), ymax(aggregated_raster), length.out =
ncol(aggregated_matrix)),
      z=aggregated_matrix,  col = terrain.colors(32), xlab = "Longitude", ylab="Latitude",
zlab="Elevation")
```

Refer the result section for 3D model of the Happisburgh DTM.

RESULTS & DISCUSSIONS

This section discusses the results produced for all the tasks and steps detailed in methodology section.

1: Image simulation of a matrix, mapping, classification and reclassification

Image simulation with matrix:

According to the process detailed in the image simulation with matrix step of the methodology section, first.img list created is shown below with x and y variables for dimensions and matrix z having uniformly distributed values ranging between 0 and 1.

```
> first.img
$x
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
28 29 30
……
[241] 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

$y
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
28 29 30
….
[241] 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

$z
          [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]      [,8]      [,9]
 [1,] 0.091320555 0.716214508 0.4582359290 0.261607569 0.291377606 0.044774707
0.1032366904 0.650028118 0.628371568
 [2,] 0.523537665 0.955934605 0.5104629593 0.337473981 0.251454877 0.546249919
0.5584592682 0.038425370 0.767283389
 [3,] 0.108470754 0.001426781 0.7619450900 0.154873882 0.763772370 0.100967022
0.5300013116 0.254138485 0.354467952
 .......
      [,251]     [,252]     [,253]     [,254]     [,255]
 [1,] 0.266609253 0.0214321201 0.714585365 0.172331091 0.360930000
 [2,] 0.299276538 0.7438580443 0.704725230 0.244660815 0.313488496
 [3,] 0.131796407 0.3089746707 0.279972233 0.993697805 0.599004874
 [ reached getOption("max.print") -- omitted 253 rows ]
```

Mapping the simulated data:

In this step, the first.img data is plotted as raster as shown in fig. 1 below with the process described in mapping the simulated data step of the methodology section.

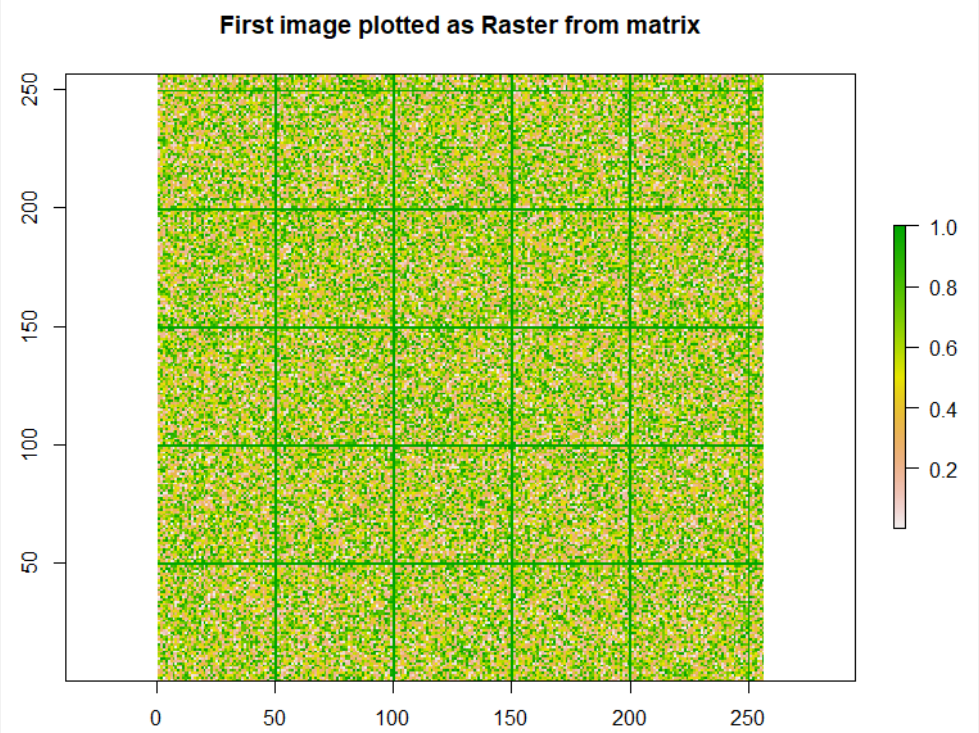

Fig. 1. First image plotted as Raster from matrix.

Classify the image into 10 classes:

In this step, the first.img is classified into class.img having 10 possible values ranging from 0.0 to 1.0 as shown below. Further after classification, the class.img is plotted as shown in fig. 2. The process for classification is detailed in classify the image into 10 classes step.

```
> class.img
$x
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
28 29 30
 ...
[241] 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

$y
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
28 29 30
 ...
[241] 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

$z
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20]
[,21]
 [1,] 0.1 0.7 0.5 0.3 0.3 0.0 0.1 0.7 0.6  0.0  0.6  0.6  0.4  0.6  0.7  0.9  0.0  0.4  0.2  0.6
0.5
...
[,237] [,238] [,239] [,240] [,241] [,242] [,243] [,244] [,245] [,246] [,247] [,248] [,249] [,250] [,251]
[,252] [,253]
 [1,]  0.0  0.4  0.1  0.4  0.5  0.3  0.1  0.9  0.7  0.7  0.1  0.1  0.4    1  0.3  0.0  0.7
 [2,]  0.9  0.5  0.4  0.6  0.4  0.1  0.3  0.8  0.1  0.0  0.3  0.0  0.3    1  0.3  0.7  0.7
 [3,]  0.1  0.3  0.3  0.9  0.8  0.0  0.4  0.6  0.1  0.4  0.9  0.6  0.8    1  0.1  0.3  0.3
     [,254] [,255]
 [1,]  0.2  0.4
 [2,]  0.2  0.3
 [3,]  1.0  0.6
```

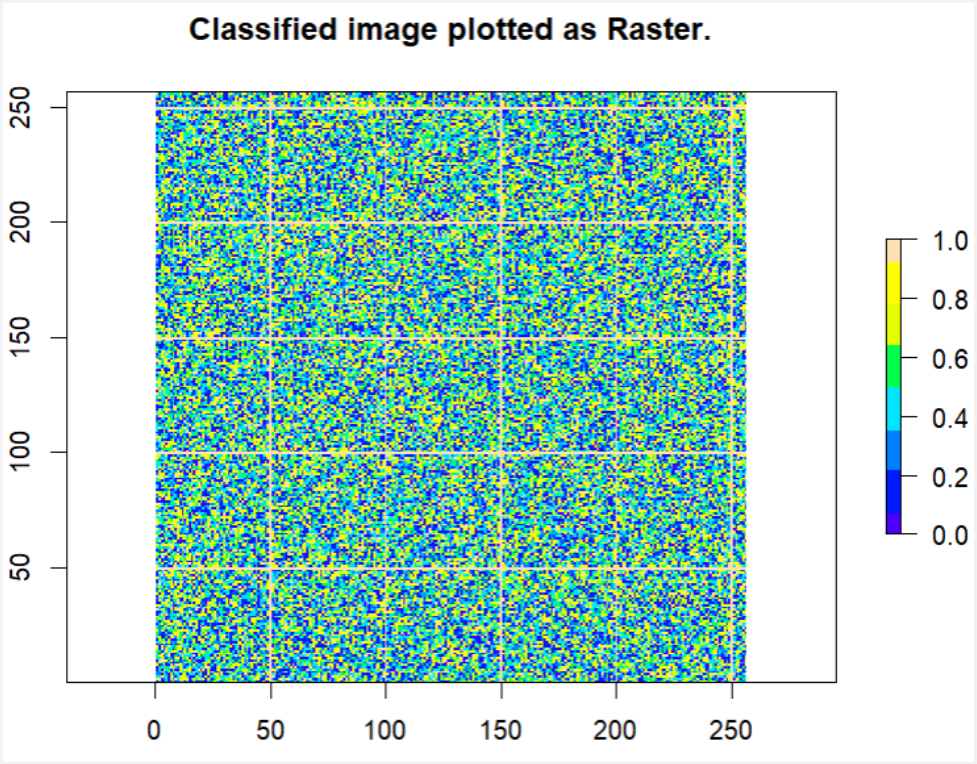

Fig. 2. Classified image plotted as Raster.

Reclassify the image:

The class.img is further reclassified to have values ranging only between 0.0 and 0.5 and plotted as shown in the fig. 3 below. The process of reclassification is detailed in the reclassify the image step of methodology section.

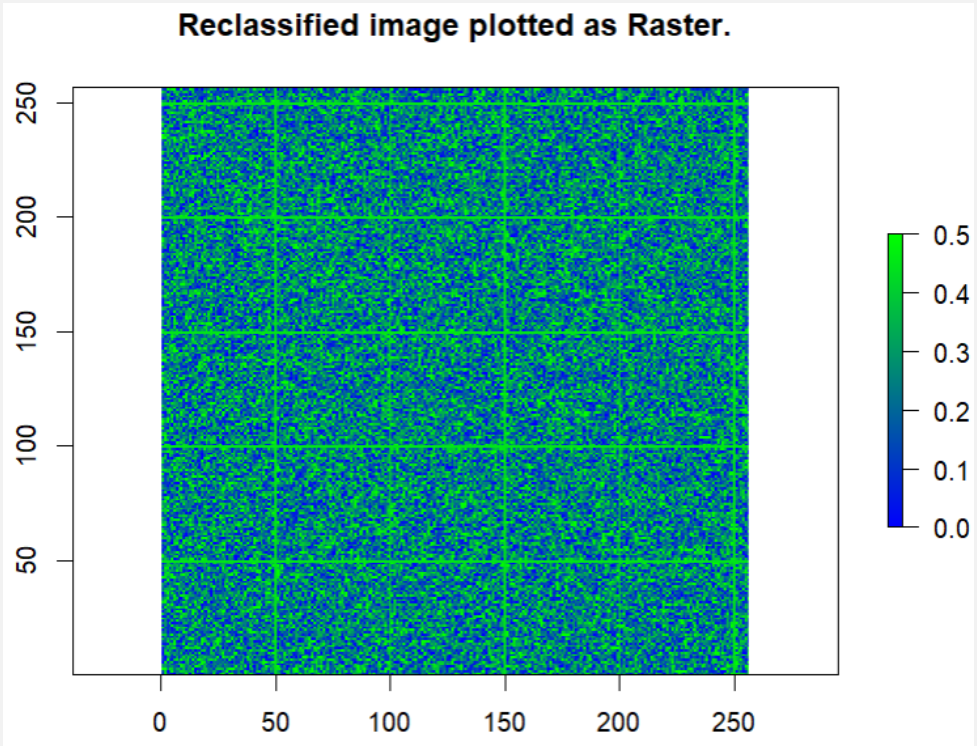

Fig. 3. Reclassified image plotted as raster.

2: Findhorn Bay DTM plotting, sea level change simulation and 3D Modelling.

Findhorn Bay DTM is plotted and processed for sea level change simulation as detailed below. Additionally, Happisburgh DTM is aggregated, and 3D plotted as well.

Reading and plotting the DTM:

The Findhorn Bay DTM raster layer is read and plotted as shown in fig. 4 with the process detailed in Reading the DTM using R and Plotting the DTM steps of the methodology section.

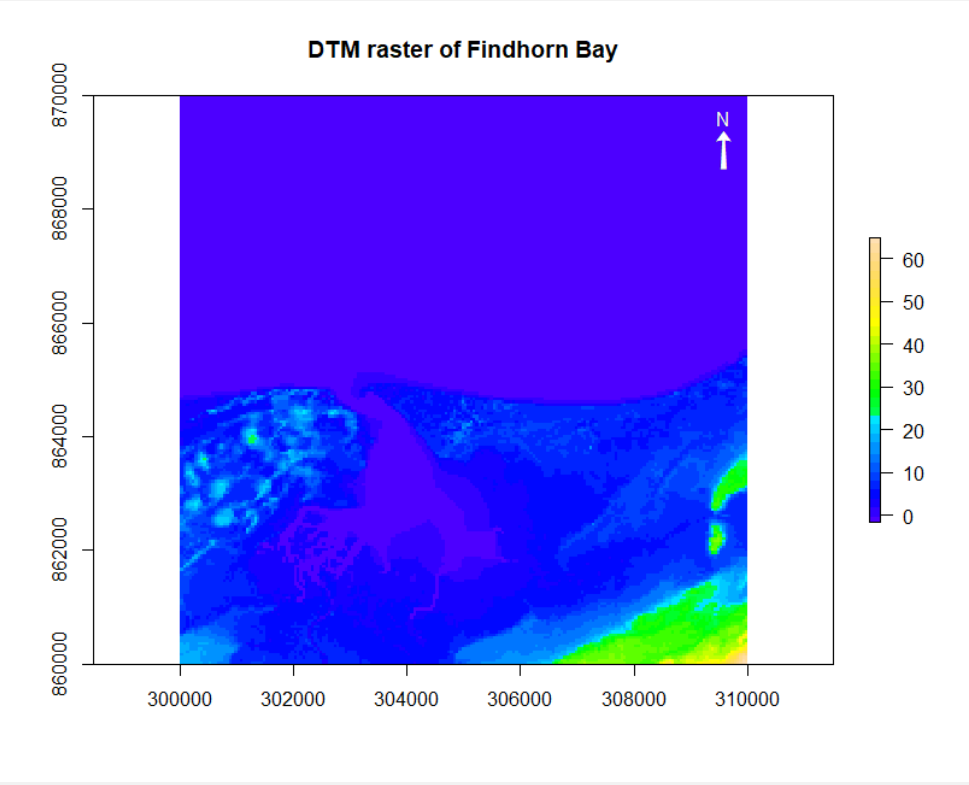

Fig. 4. DTM raster of Findhorn Bay.

Reclassification for sea level changes:

The process of reclassification for sea level change is detailed in the methodology section earlier. As part of the process, variables raster_range and elevation_range are derived as shown below.

```
> raster_range
[1] -1.6 64.8
> elevation_range
 [1]  0  2  4  6  8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60
```

Further, the DTM is added with contour line for present sea level changes as shown in fig. 5 and subsequently, the DTM is added with yellow contour line to identify new sea level change of 4 metres as shown in the fig. 6 below.

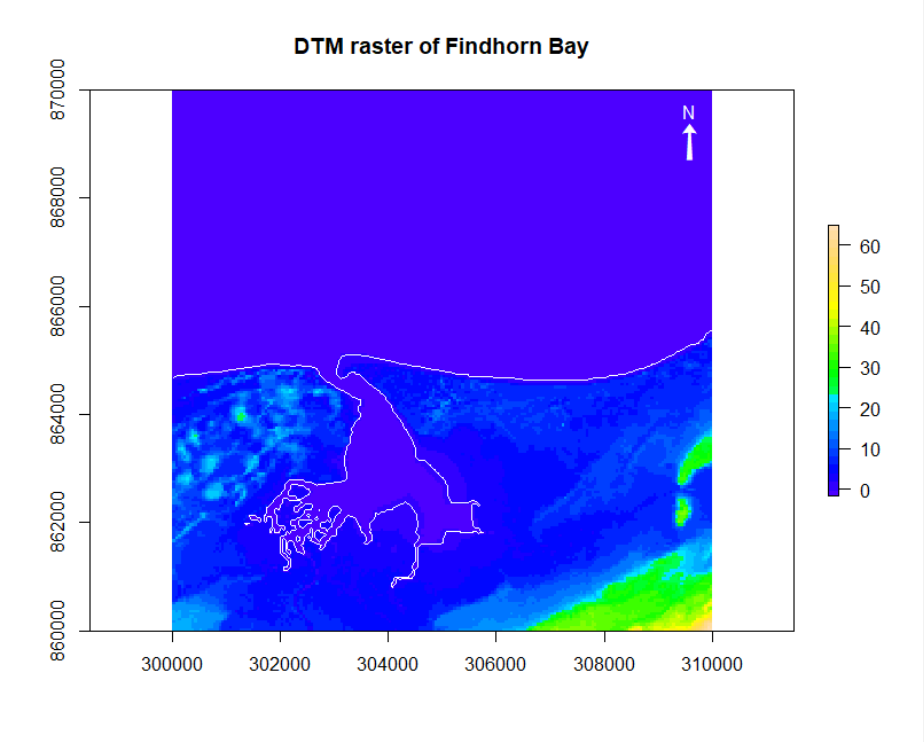

Fig. 5. DTM with present sea level change shown with white contour line.

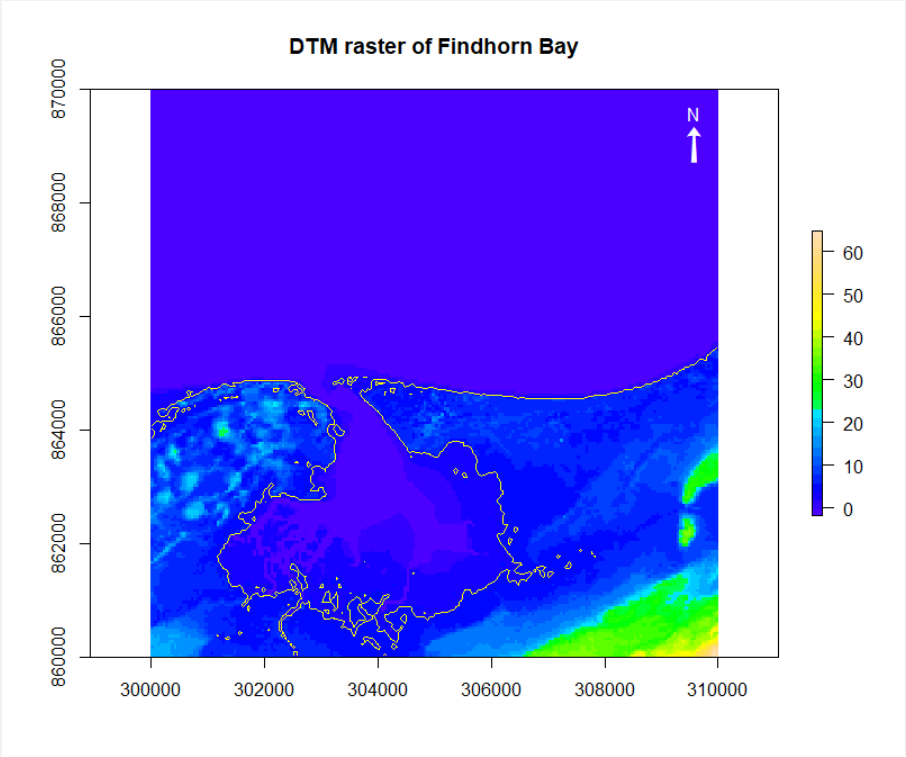

Fig. 6. DTM with new sea level change of 4m shown with yellow contour line.

Furthermore, the DTM of Findhorn Bay is reclassified to show sea level changes of 4 metres as shown below in fig. 7. As per the simulation, the Findhorn Bay will be inundated as shown when sea levels will rise by 4 metres.

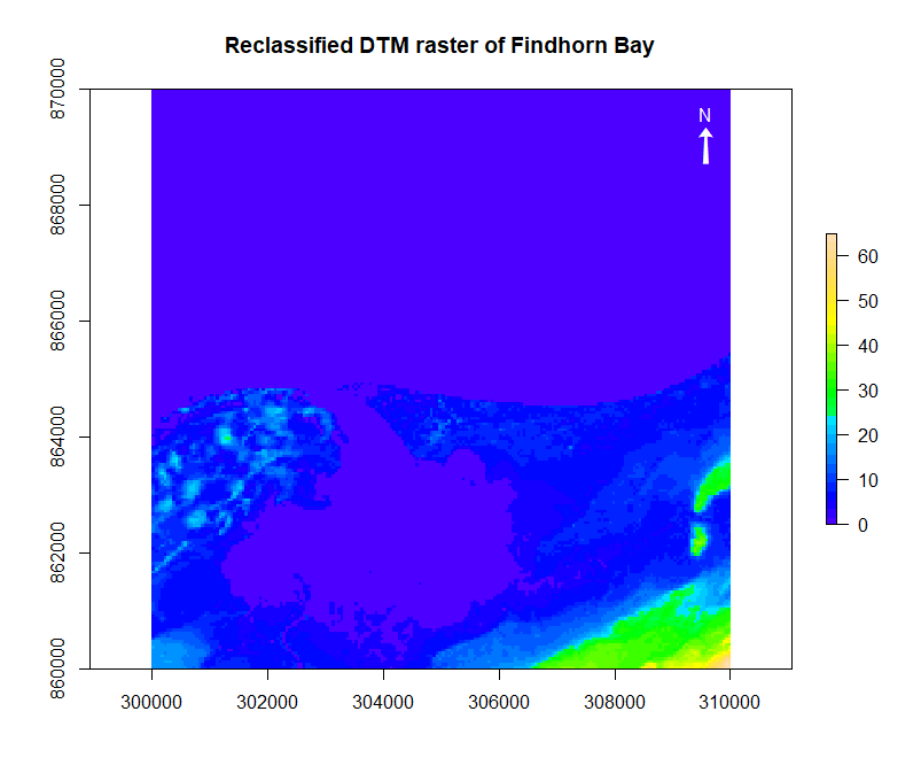

Fig. 7. Reclassified DTM with 4m of sea level change.

3D plotting the Happisburgh_Survey_dsm.tiff:

The 3D plotting process of the Happisburgh_Survey_dsm.tiff raster is detailed in the methodology section. The Happisburgh DTM raster is summarised as shown below.

```
> happisburgh_raster
class      : RasterLayer
dimensions : 16026, 17095, 273964470  (nrow, ncol, ncell)
resolution : 0.0143, 0.0143  (x, y)
extent     : 638508, 638752.5, 330647.9, 330877.1  (xmin, xmax, ymin, ymax)
crs        : +proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000
+ellps=airy +units=m +no_defs
source     : Happisburgh_Survey_dsm.tiff
names      : Happisburgh_Survey_dsm
```

Due to the higher resolution of 0.0143 metres per pixel, 3D plotting of the DTM is highly memory intensive. Hence, the aggregated raster is created as summarised below.

```
> aggregated_raster
class      : RasterLayer
dimensions : 1603, 1710, 2741130  (nrow, ncol, ncell)
resolution : 0.143, 0.143  (x, y)
extent     : 638508, 638752.5, 330647.9, 330877.1  (xmin, xmax, ymin, ymax)
crs        : +proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000
+ellps=airy +units=m +no_defs
source     : r_tmp_2023-05-09_121620_3240_14634.grd
names      : Happisburgh_Survey_dsm
values     : -0.4726772, 14.34385  (min, max)
```

After the raster was aggregated to a manageable resolution of 0.143m, it is first plotted in 2D as shown below in fig. 8.

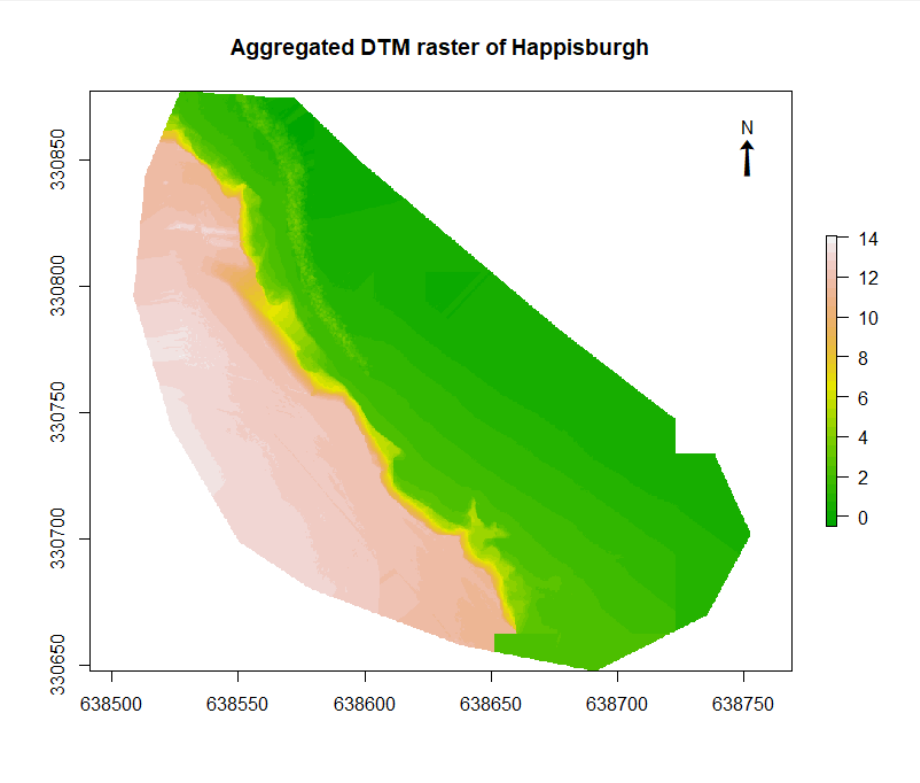

Fig. 8. Aggregated DTM of Happisburgh with 0.143m resolution.

Further, the aggregated Happisburgh DTM is 3D plotted using persp3d() function as shown in fig. 9a and 9b below. Reviewing the 3D plot, it can be noticed that elevation of the DTM ranges between -0.47 and 14.34 metres, and it steeply rises from less than 5 metres to more than 10 metres across the middle section of the DTM.

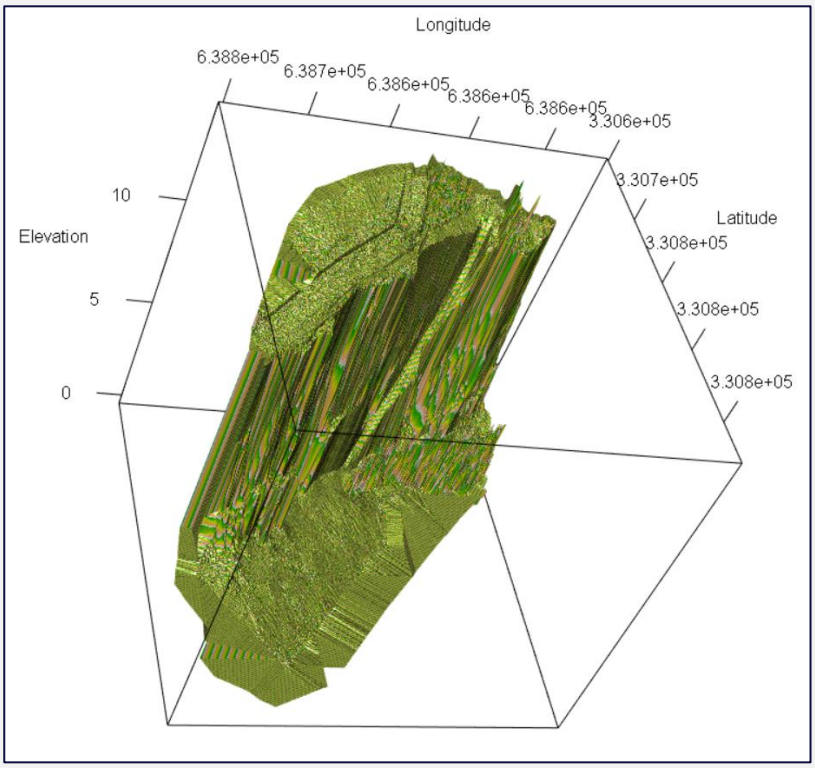

Fig. 9a. A 3D View of aggregated DTM of Happisburgh.

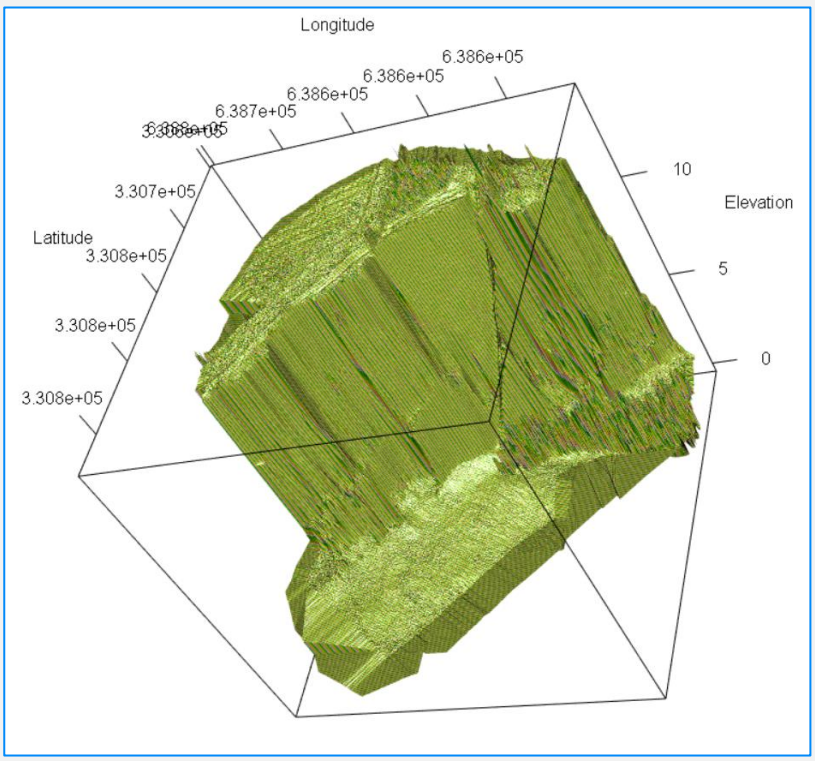

Fig. 9b. Another 3D View of aggregated DTM of Happisburgh.

SUMMARY & CONCLUSION

The language R has powerful spatial packages, and it also has strong base features. When the base features are used in conjunction with spatial packages, they can be extremely useful for geospatial analysis and GIS applications. Although, the spatial tools explored in this report were limited, yet a detailed spatial analysis could still be performed as it can be noticed from the tasks detailed in the methodology section earlier. Not only the tasks accomplished a complex spatial analysis such as image simulation, classification, reclassification, DTM processing, sea level change simulation and 3D modelling with limited set of libraries, but the libraries also produced a visually detailed output for all the steps.

One of the predominantly used spatial packages was the 'raster' package. With the package, not only custom matrices were created and plotted, but data sets such as DTMs stored on the file system in .asc or .tif formats could also be efficiently loaded and plotted with the package. Further, several functions from the package were found to be useful such as getValues() for retrieving values, res() for identifying resolution and dim() for dimensions etc. Additionally, several mathematical operations performed on several rasters are also useful such as applying raster algebra, using calc() function for sea level change reclassification or applying aggregate() function to simply change the resolution of a raster.

Another powerful tool used for the tasks was the 'rgl' package for 3D plotting. Specifically, the persp3d() function from the package was used to efficiently plot a DTM in three dimensions. Specifically, the 3D plotting of Happisburgh DTM highlighted a noticeably important and sudden elevation change of several metres which was not possible by referencing the 2D plot. Additionally, the package also offers other powerful functions such as plot3d(), points3d(), lines3d(), text3d() etc. which could also be useful in performing more complex 3D operations with R.

REFERENCES

- An Introduction to R (2023), R Core Team. Available at: http://www.stats.bris.ac.uk/R/doc/manuals/R-intro.html
- Hijmans R. (2012), Introduction to the 'raster' package, version 2.0-31. Available at: https://mran.microsoft.com/snapshot/2019-02-07/web/packages/raster/vignettes/Raster.pdf
- Visit Forres (2023), Findhorn Bay, Visit Forres, Moray Media CIC. Available at: https://visitforres.scot/listing/findhorn-bay/
- Visit Moray Speyside (2023), Findhorn, Visit Moray Speyside, Moray Speyside. Available at: https://morayspeyside.com/visit/findhorn/
- RGL (2023), Package: rgl, 3D Visualization Using OpenGL, Version: 1.1.3. Available at: https://dmurdoch.github.io/rgl/
- Terra (2023), Package: terra, Spatial Data Analysis, Version: 1.7-18. Available at: https://rspatial.org/terra/

APPENDICES