ndvi3

niamh

2024-03-14

```r
library(knitr)
require(kableExtra)
```

```
## Loading required package: kableExtra
```

```r
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.0     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter()     masks stats::filter()
## ✖ dplyr::group_rows() masks kableExtra::group_rows()
## ✖ dplyr::lag()        masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
options(knitr.table.format = "html")
```

```r
# Set the working directory (replace with your own file path)
setwd("R:/GEOG493_593_25793_Winter2024/Student_Data/niamhh/R/data/CC-spatial-master/")

# Load packages

# Install packages if needed
# Example:install.packages("sp")

library(sp)
library(raster)
```

```
##
## Attaching package: 'raster'
```

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

```r
library(ggplot2)
library(viridis)
```

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

```r
library(rasterVis)
```

```
## Loading required package: lattice
```

```r
# Load data
tay <- raster('taycrop.tif')

# Get properties of the Tay raster
tay
```

```
## class      : RasterLayer
## band       : 1  (of  12  bands)
## dimensions : 507, 848, 429936  (nrow, ncol, ncell)
## resolution : 9.217891e-05, 9.217891e-05  (x, y)
## extent     : -4.320218, -4.242051, 56.45366, 56.50039  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : taycrop.tif
## names      : taycrop_1
```

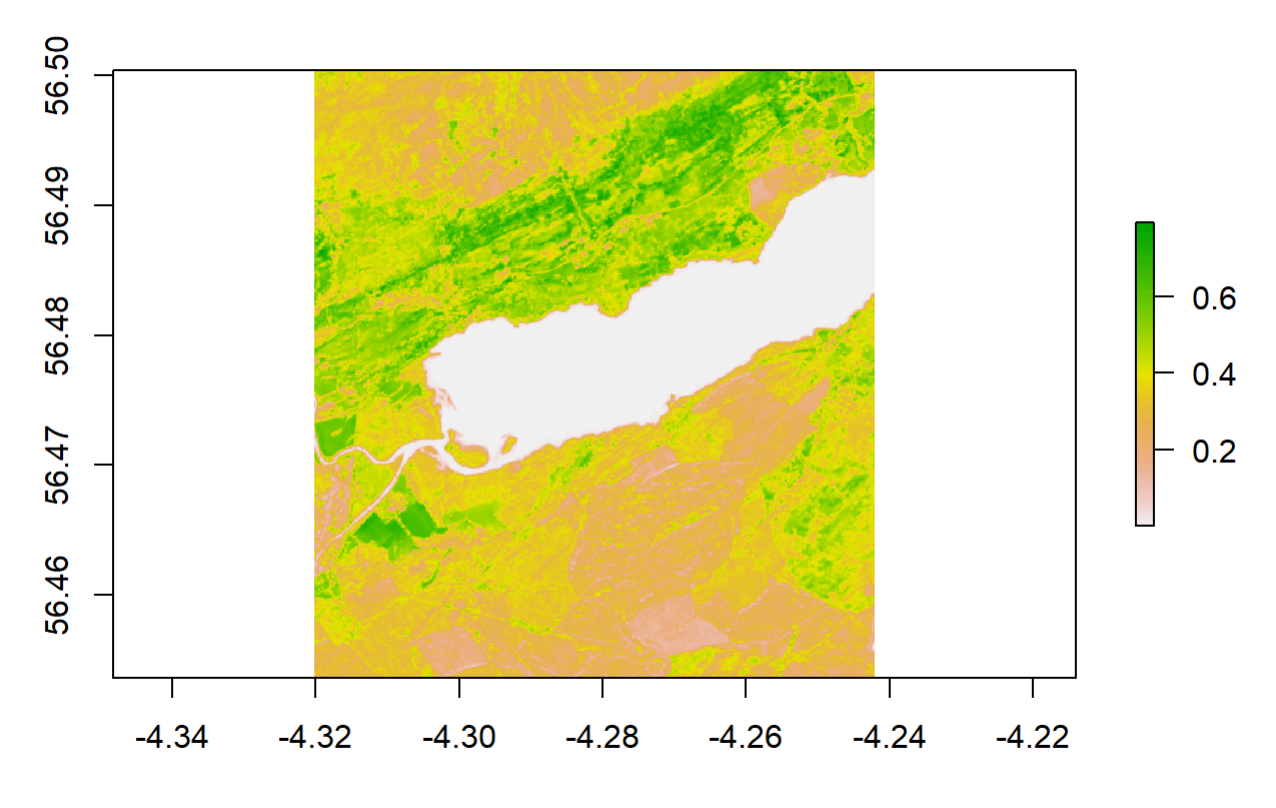
```r
b1 <- raster('taycrop.tif', band=1)
b2 <- raster('taycrop.tif', band=2)
b3 <- raster('taycrop.tif', band=3)
b4 <- raster('taycrop.tif', band=4)
b5 <- raster('taycrop.tif', band=5)
b6 <- raster('taycrop.tif', band=6)
b7 <- raster('taycrop.tif', band=7)
b8 <- raster('taycrop.tif', band=8)
b9 <- raster('taycrop.tif', band=9)
b10 <- raster('taycrop.tif', band=10)
b11 <- raster('taycrop.tif', band=11)
b12 <- raster('taycrop.tif', band=12)
```

```r
compareRaster(b2, b3)
```

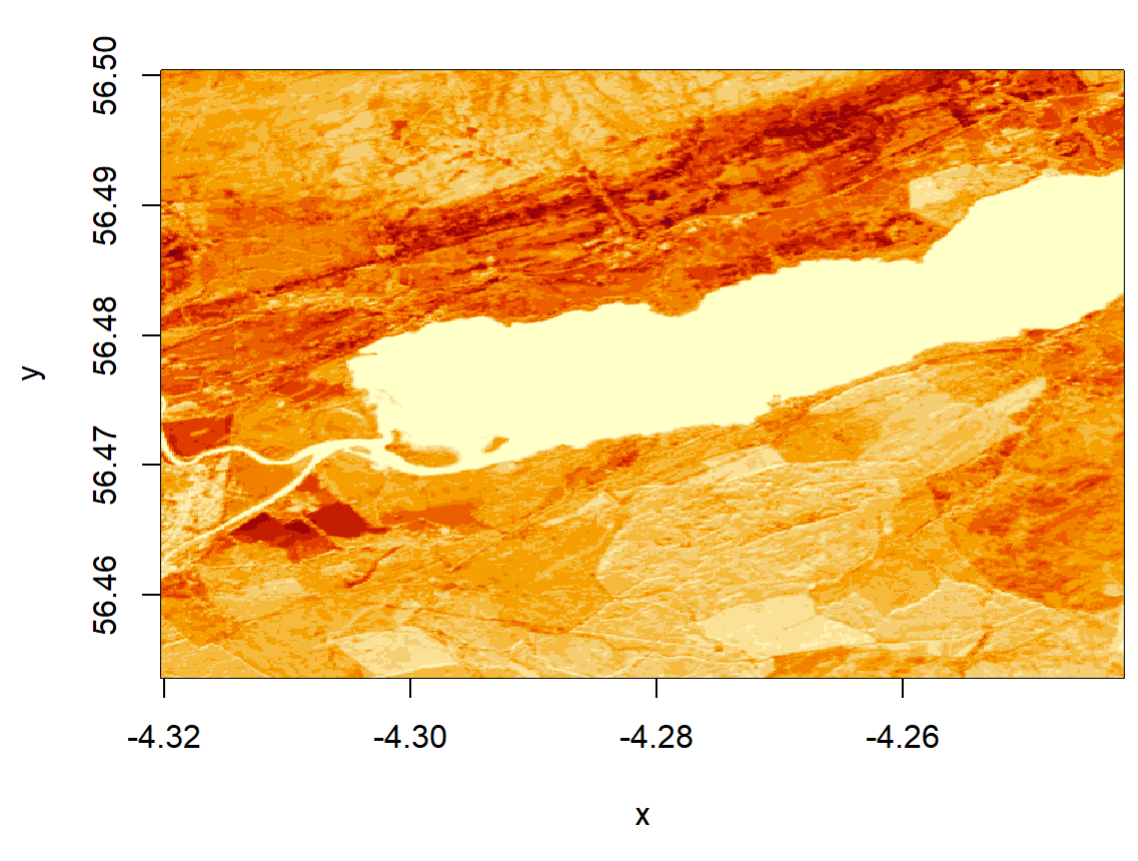
```
## [1] TRUE
```

```r
# TRUE
```

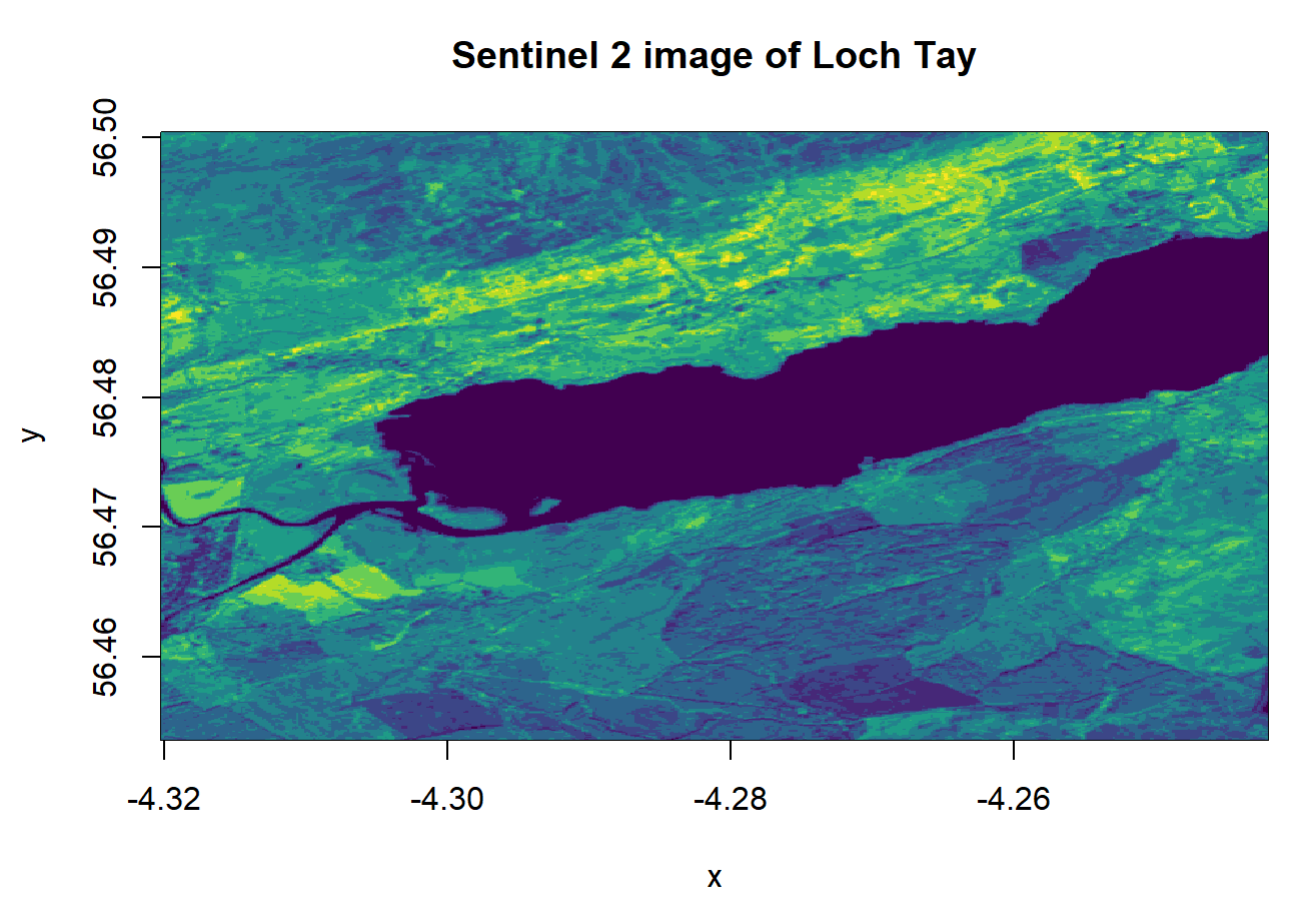
```r
plot(b8)
```



```r
image(b8)
```



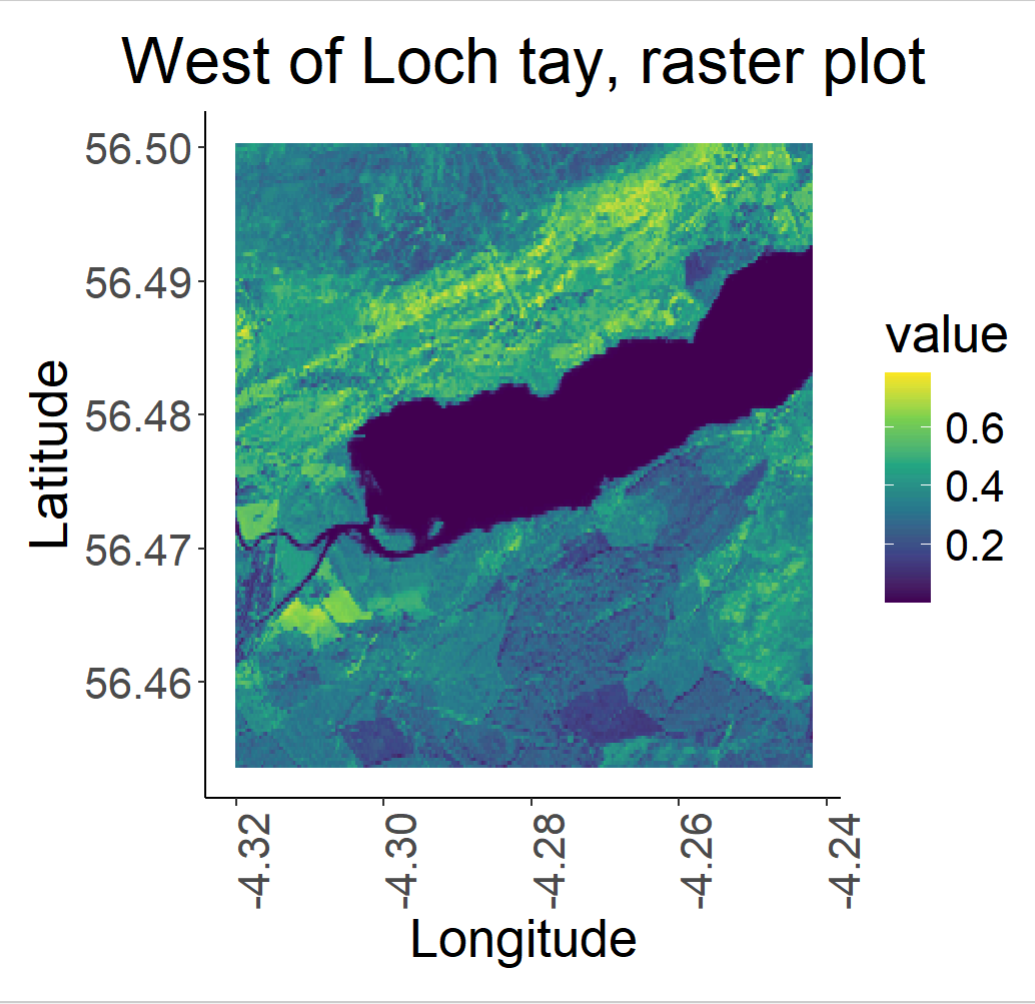
```r
image(b8, col= viridis_pal(option="D")(10), main="Sentinel 2 image of Loch Tay")
```



```r
# this code specifies how we want to save the plot
png('RGB.png', width = 5, height = 4, units = "in", res = 300)
tayRGB <- stack(list(b4, b3, b2))              # creates raster stack
plotRGB(tayRGB, axes = TRUE, stretch = "lin", main = "Sentinel RGB colour composite")
dev.off()
```

```
## png
##   2
```

```r
gplot(b8) +
  geom_raster(aes(x = x, y = y, fill = value)) +
  # value is the specific value (of reflectance) each pixel is associated with
  scale_fill_viridis_c() +
  coord_quickmap() +
  ggtitle("West of Loch tay, raster plot") +
  xlab("Longitude") +
  ylab("Latitude") +
  theme_classic() +                         # removes defalut grey background
  theme(plot.title = element_text(hjust = 0.5),             # centres plot title
        text = element_text(size=20),                # font size
        axis.text.x = element_text(angle = 90, hjust = 1))  # rotates x axis text
```

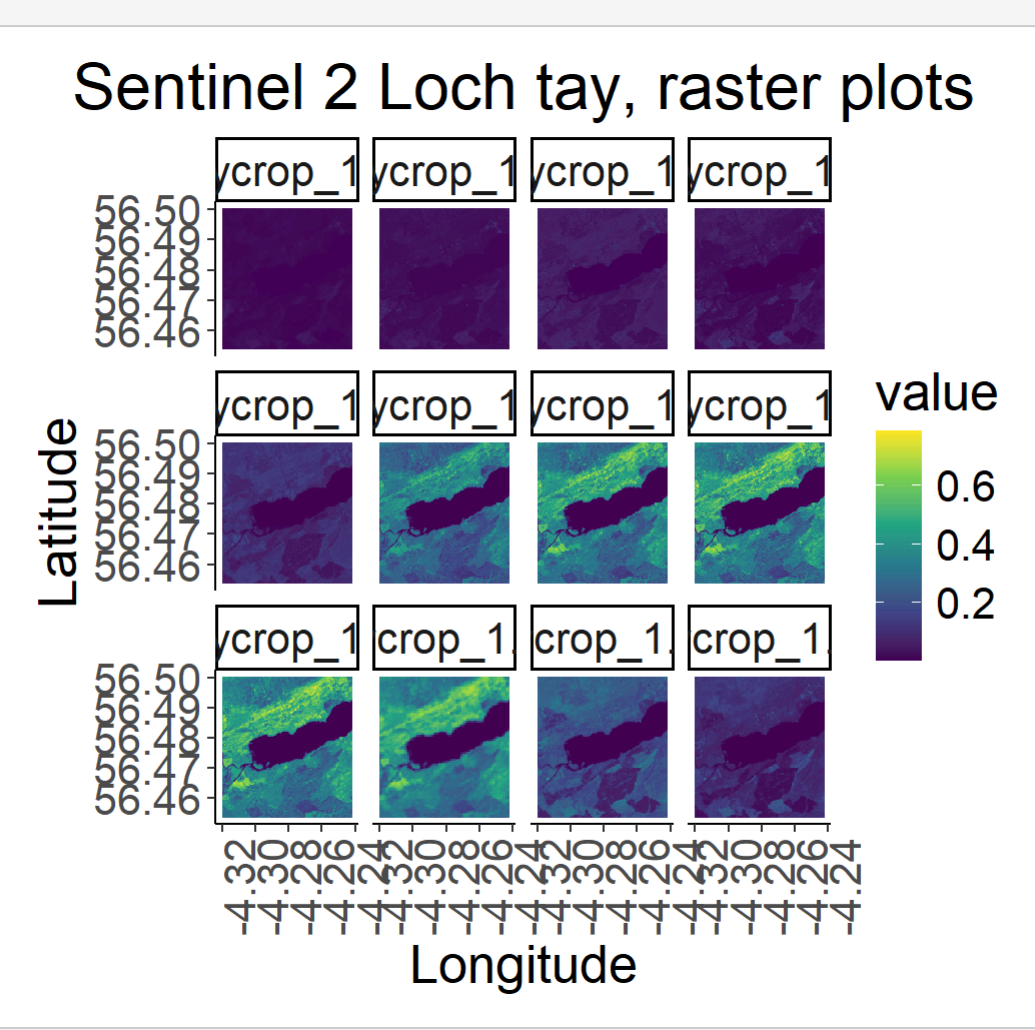


```r
ggsave("ggtay.png", scale = 1.5, dpi = 300)         # to save plot
```

```
## Saving 10.5 x 7.5 in image
```

```r
t <- stack(b1,b2, b3, b4, b5, b6, b7, b8, b9, b10, b11, b12)
```

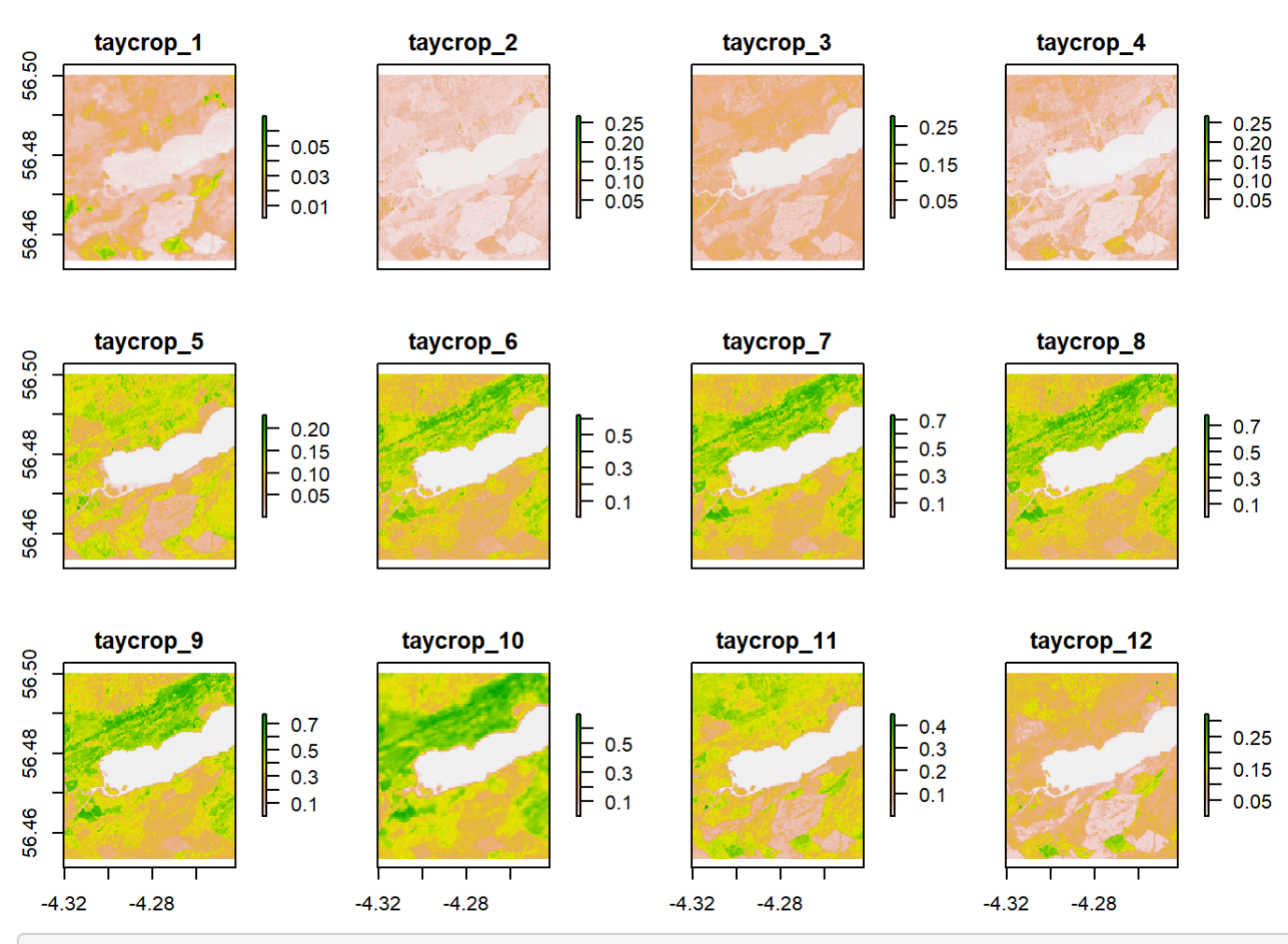
```r
gplot(t) +
  geom_raster(aes(x = x, y = y, fill = value))+
  scale_fill_viridis_c() +
  facet_wrap(~variable) +
  coord_quickmap()+
  ggtitle("Sentinel 2 Loch tay, raster plots") +
  xlab("Longitude") +
  ylab("Latitude") +
  theme_classic() +
  theme(text = element_text(size=20),
        axis.text.x = element_text(angle = 90, hjust = 1)) +
  theme(plot.title = element_text(hjust = 0.5))
```



```r
ggsave("allbands.png", scale = 1.5, dpi = 300) # to save plot
```

```
## Saving 10.5 x 7.5 in image
```

```r
s_tay <- brick('taycrop.tif')
plot(s_tay)
```



```r
# NDVI

# Created a VI function (vegetation index)
VI <- function(img, k, i) {
  bk <- img[[k]]
  bi <- img[[i]]
  vi <- (bk - bi) / (bk + bi)
  return(vi)
}

# For Sentinel 2, the relevant bands to use are:
# NIR = 8, red = 4
```

```r
ndvi <- VI(s_tay, 8, 4)
# 8 and 4 refer to the bands we'll use

png('ndviplot.png', width = 4, height = 4, units = "in", res = 300)
plot(ndvi, col = rev(terrain.colors(10)), main = 'Sentinel 2, Loch Tay-NDVI')
dev.off()
```

```
## png
##   2
```

```r
# Create histogram of NDVI data

png('ndvihist.png', width = 4, height = 4, units = "in", res = 300)
hist(ndvi,
     main = "Distribution of NDVI values",
     xlab = "NDVI",
     ylab= "Frequency",
     col = "aquamarine3",
     xlim = c(-0.5, 1),
     breaks = 30,
     xaxt = 'n')
axis(side = 1, at = seq(-0.5,1, 0.05), labels = seq(-0.5,1, 0.05))
dev.off()
```

```
## png
##   2
```

```r
# Mask cells that have NDVI of less than 0.4 (less likely to be vegetation)

png('ndvimask.png', width = 4, height = 4, units = "in", res = 300)

veg <- reclassify(ndvi, cbind(-Inf, 0.4, NA))
# We are reclassifying our object and making all values between
# negative infinity and 0.4 be NAs

plot(veg, main = 'Veg cover')
dev.off()
```

```
## png
##   2
```

```r
writeRaster(x = ndvi, filename="R:/GEOG493_593_25793_Winter2024/Student_Data/niamhh/R/tay_ndvi_2018.tif", format
= "GTiff", datatype = 'INT2S', overwrite=TRUE)
```

```r
# convert the raster to vector/matrix ('getValues' converts the RasterLAyer to array) )

nr <-getValues(ndvi)
str(nr)
```

```
##  num [1:429936] 0.791 0.791 0.785 0.783 0.783 ...
```

```r
# important to set the seed generator because `kmeans` initiates the centres in random locations
# the seed generator just generates random numbers

set.seed(99)

# create 10 clusters, allow 500 iterations, start with 5 random sets using 'Lloyd' method

kmncluster <- kmeans(na.omit(nr), centers = 10, iter.max = 500,
                     nstart = 5, algorithm = "Lloyd")

# kmeans returns an object of class 'kmeans'

str(kmncluster)
```

```
## List of 9
##  $ cluster     : int [1:429936] 7 7 7 7 7 7 7 4 4 7 ...
##  $ centers     : num [1:10, 1] -0.525 0.421 0.846 0.696 0.233 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:10] "1" "2" "3" "4" ...
##   .. ..$ : NULL
##  $ totss       : num 156503
##  $ withinss    : num [1:10] 67.4 23.6 28.1 28.5 32.4 ...
##  $ tot.withinss: num 484
##  $ betweenss   : num 156020
##  $ size        : int [1:10] 4650 8155 71872 45902 10754 58153 58643 18762 4109 148936
##  $ iter        : int 287
##  $ ifault      : NULL
##  - attr(*, "class")= chr "kmeans"
```

```r
# First create a copy of the ndvi layer
knr <- ndvi

# Now replace raster cell values with kmncluster$cluster
# array
knr[] <- kmncluster$cluster

# Alternative way to achieve the same result
values(knr) <- kmncluster$cluster
knr
```

```
## class      : RasterLayer
## dimensions : 507, 848, 429936  (nrow, ncol, ncell)
## resolution : 9.217891e-05, 9.217891e-05  (x, y)
## extent     : -4.320218, -4.242051, 56.45366, 56.50039  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## source     : memory
## names      : layer
## values     : 1, 10  (min, max)
```

```r
par(mfrow = c(1, 2))
plot(ndvi, col = rev(terrain.colors(10)), main = "NDVI")
plot(knr, main = "Kmeans", col = viridis_pal(option = "D")(10))
```