PRACTICAL 4

Aim of this practical:

1. Work with different types of spatial data types including: Areal, Geostatistical and Spatial Point process data.
2. Read and visualize shapefiles into R
3. Manipulate and visualize `sf` spatial object in R
4. Manipulate and visualize raster data in R.

In this practical we will:

- Explore tools for areal spatial data wrangling and visualization.

1 Areal (lattice) data

Areal data our measurements are summarised across a set of discrete, non-overlapping spatial units such as postcode areas, health board or pixels on a satellite image. In consequence, the spatial domain is a countable collection of (regular or irregular) areal units at which variables are observed. Many public health studies use data aggregated over groups rather than data on individuals - often this is for privacy reasons, but it may also be for convenience.

In the next example we are going to explore data on respiratory hospitalisations for Greater Glasgow and Clyde between 2007 and 2011. The data are available from the `CARBayesdata` R Package:

```
library(CARBayesdata)

data(pollutionhealthdata)
data(GGHB.IZ)
```

The `pollutionhealthdata` contains the spatiotemporal data on respiratory hospitalisations, air pollution concentrations and socio-economic deprivation covariates for the 271 Intermediate Zones (IZ) that make up the Greater Glasgow and Clyde health board in Scotland. Data are provided by the Scottish Government and the available variables are:

- `IZ`: unique identifier for each IZ.
- `year`: the year were the measruments were taken
- `observed`: observed numbers of hospitalisations due to respiratory disease.
- `expected`: expected numbers of hospitalisations due to respiratory disease computed using indirect standardisation from Scotland-wide respiratory hospitalisation rates.
- `pm10`: Average particulate matter (less than 10 microns) concentrations.
- `jsa`: The percentage of working age people who are in receipt of Job Seekers Allowance
- `price`: Average property price (divided by 100,000).

The `GGHB.IZ` data is a Simple Features (`sf`) object containing the spatial polygon information for the set of 271 Intermediate Zones (IZ), that make up of the Greater Glasgow and Clyde health board in Scotland (Figure 1).

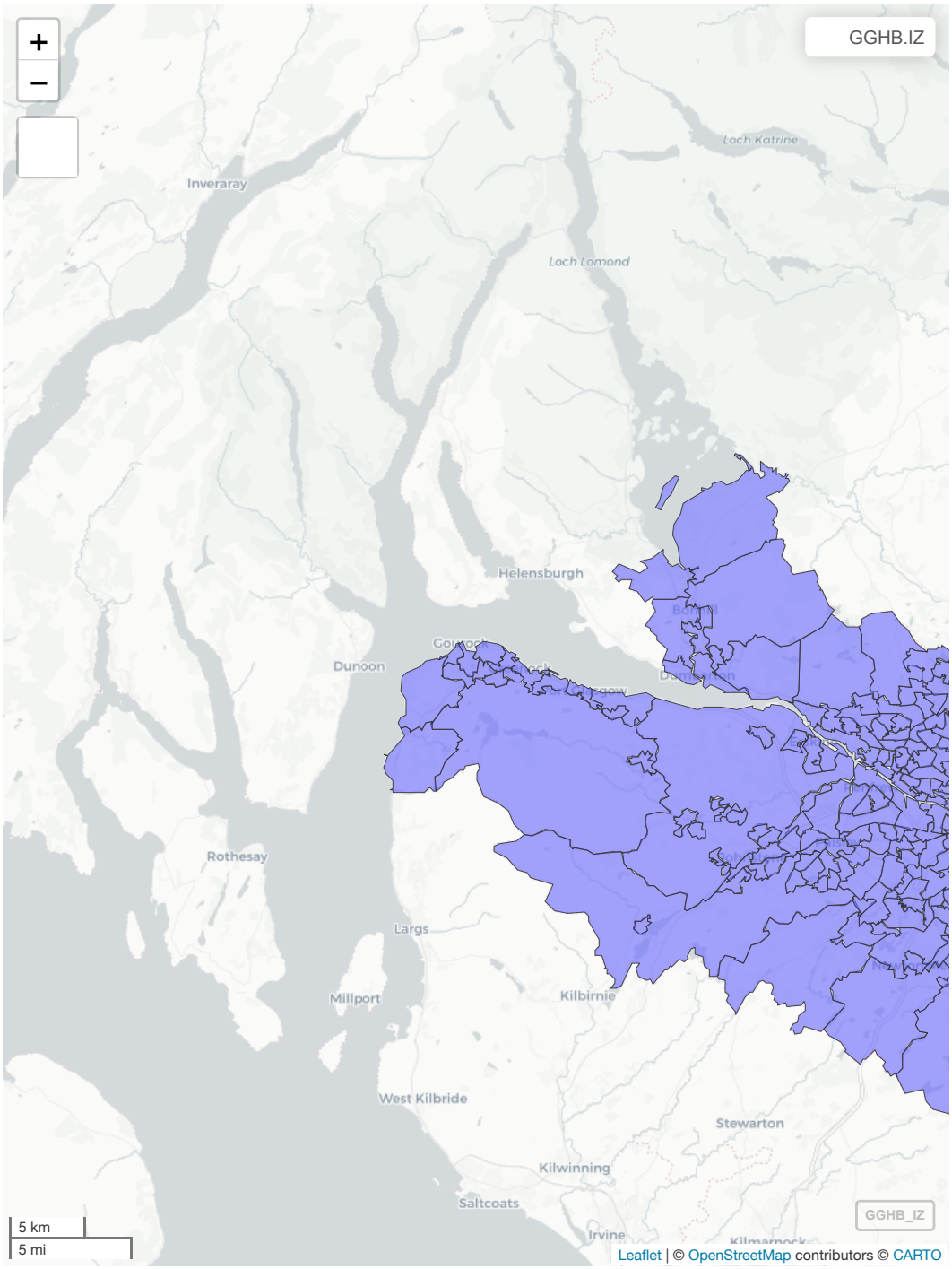


Figure 1: Greater Glasgow and Clyde health board represented by 271 Intermediate Zones

Let's start by loading useful libraries:

```
library(sf)
library(ggplot2)
library(scico)
```

The `sf` package allows us to work with vector data which is used to represent points, lines, and polygons. It can also be used to read vector data stored as a shapefiles.

First, lets combine both data sets based on the Intermediate Zones (IZ) variable using the `merge` function from `base` R:

```
resp_cases <- merge(GGHB.IZ, pollutionhealthdata, by = "IZ")
```

In epidemiology, disease risk is usually estimated using Standardized Mortality Ratios (SMR). The SMR for a given spatial areal unit i is defined as the ratio between the observed (Y_i) and expected (E_i) number of cases:

$$SMR_i = \frac{Y_i}{E_i}$$

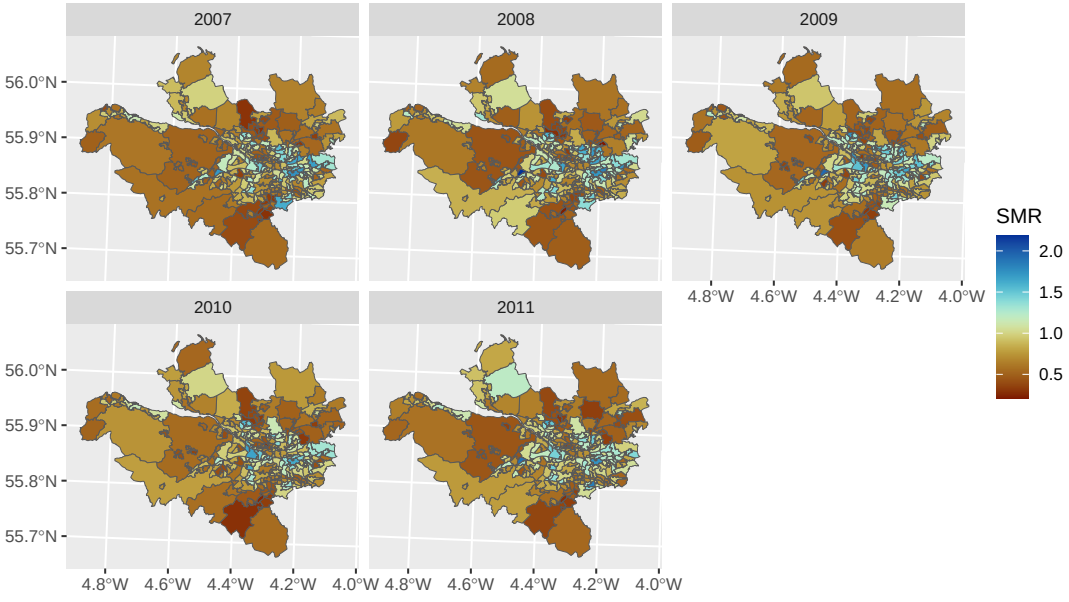
A value $SMR > 1$ indicates that there are more observed cases than expected which corresponds to a high risk area. On the other hand, if $SMR < 1$ then there are fewer observed cases than expected, suggesting a low risk area.

We can manipulate `sf` objects the same way we manipulate standard data frame objects via the `dplyr` package. Lets use the pipeline command %>% and the `mutate` function to calculate the yearly SMR values for each IZ:

```
library(dplyr)
resp_cases <- resp_cases %>%
  mutate(SMR = observed/expected, .by = year )
```

Now we use `ggplot` to visualize our data by adding a `geom_sf` layer and coloring it according to our variable of interest (i.e., SMR). We can further use `facet_wrap` to create a layer per year and chose an appropriate color palette using the `scale_fill_scico` from the `scico` package:

```
ggplot()+
  geom_sf(data=resp_cases,aes(fill=SMR))+
  facet_wrap(~year)+scale_fill_scico(palette = "roma")
```



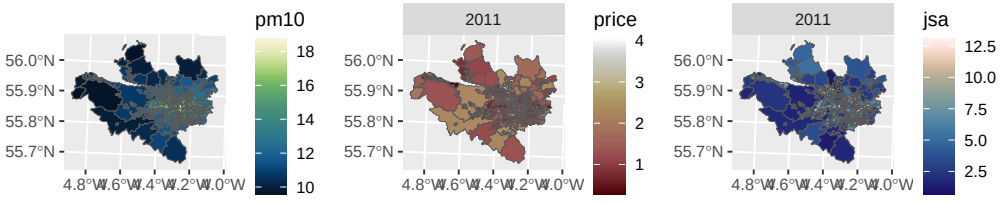
Task

Produce a map that shows the spatial distribution of each of the following variables for the year 2011:

- Average particulate matter `pm10`
- Average property price `price`
- Percentage of working age people who are in receipt of Job Seekers Allowance `jsa`

hint
You can use the `filter` function from `dplyr` to subset the data according to the year of interest.
Click here to see the solution

```
# Library for plotting multiple maps together

library(patchwork)

# subset data set for 2011

resp_cases_2011 <- resp_cases %>% filter(year ==2011)

# pm10 plot

pm10_plot <- ggplot()+
  geom_sf(data=resp_cases_2011,aes(fill=pm10))+
  scale_fill_scico(palette = "navia")

# property price

price_plot <- ggplot()+
  geom_sf(data=resp_cases_2011,aes(fill=price))+
  facet_wrap(~year)+scale_fill_scico(palette = "bilbao")

#  percentage jsa

jsa_plot <- ggplot()+
  geom_sf(data=resp_cases_2011,aes(fill=jsa))+
  facet_wrap(~year)+scale_fill_scico(palette = "lapaz")

# plot maps together

pm10_plot + price_plot + jsa_plot + plot_layout(ncol=3)
```



In this practical we will:

- Explore tools for geostatistical spatial data wrangling and visualization.

First, lets load some useful libraries for data wrangling and visualization

```
# For plotting
library(mapview)
library(ggplot2)
library(scico) # for colouring palettes

# Data manipulation
library(dplyr)
```

2 Georeferenced data

Tobler's first law of geography states that:

"*Everything is related to everything else, but near things are more related than distant things*"

Spatial patterns are fundamental in environmental and ecological data. In many ecological and environmental settings, measurements from fixed sampling units, aiming to quantify spatial variation and interpolate values at unobserved sites.

Georefernced data are the most common form of spatial data found in environmental setting. In these data we regularly take measurements of a spatial ecological or environmental process at a set of fixed locations. This could be data from transects (e.g, where the height of trees is recorded), samples taken across a region (e.g., water depth in a lake) or from monitoring stations as part of a network (e.g., air pollution). In each of these cases, our goal is to estimate the value of our variable across the entire space.

Let D be our two-dimensional region of interest. In principle, there is aninfinite number of locations within D, each of which can be represented by mathematical coordinates (e.g., latitude and longitude). We then can identify any individual location as $s_i = (x_i, y_i)$, where x_i and y_i are their coordinates.

We can treat our variable of interest as a random variable, Z which can be observed at any location as $Z(\mathbf{s}_i)$.

Our geostatistical process can therefore be written as:

$$\{Z(\mathbf{s}); \mathbf{s} \in D\}$$

In practice, our data are observed in a finite number of locations, m, and can be denoted as:

$$z = \{z(\mathbf{s}_1), \dots z(\mathbf{s}_m)\}$$

In the next example, we will explore data on the Pacific Cod (*Gadus macrocephalus*) from a trawl survey in Queen Charlotte Sound. The `pcod` dataset is available from the `sdmTMB` package and contains the presence/absence records of the Pacific Cod during each surveys along with the biomass density of Pacific cod in the area swept (kg/Km^2). The `qcs_grid` data contain the depth values stored as 2×2 km grid for Queen Charlotte Sound.

```
library(sdmTMB)

pcod_df = sdmTMB::pcod
```

```
qcs_grid = sdmTMB::qcs_grid
```

2 Georeferrenced data

Let's create an initial `sf` spatial object using the standard geographic coordinate system (`EPSG:4326`). This correctly defines the point locations based on latitude and longitude.

```
library(sf)
pcod_sf =   st_as_sf(pcod_df, coords = c("lon","lat"), crs = 4326)
```

Now we can transform to the standard UTM Zone 9N projection (EPSG:32609) which uses meters:

```
pcod_sf_proj <- st_transform(pcod_sf, crs = 32609)
st_crs(pcod_sf_proj)$units
```

```
[1] "m"
```

We can change the spatial units to *km* to better reflect the scale of our ecological study and to make resulting distance/area values more intuitive to interpret:

```
pcod_sf_proj = st_transform(pcod_sf_proj,
                            gsub("units=m","units=km",
                                 st_crs(pcod_sf_proj)$proj4string))
st_crs(pcod_sf_proj)$units
```

```
[1] "km"
```

Instead of first setting an EPSG code and then transforming, we can define the target Coordinate Reference System (CRS) directly using a `proj4string`. This allows us to customize non-standard parameters in a single step, in this case, explicitly setting the projection units to kilometers (`+units=km`).

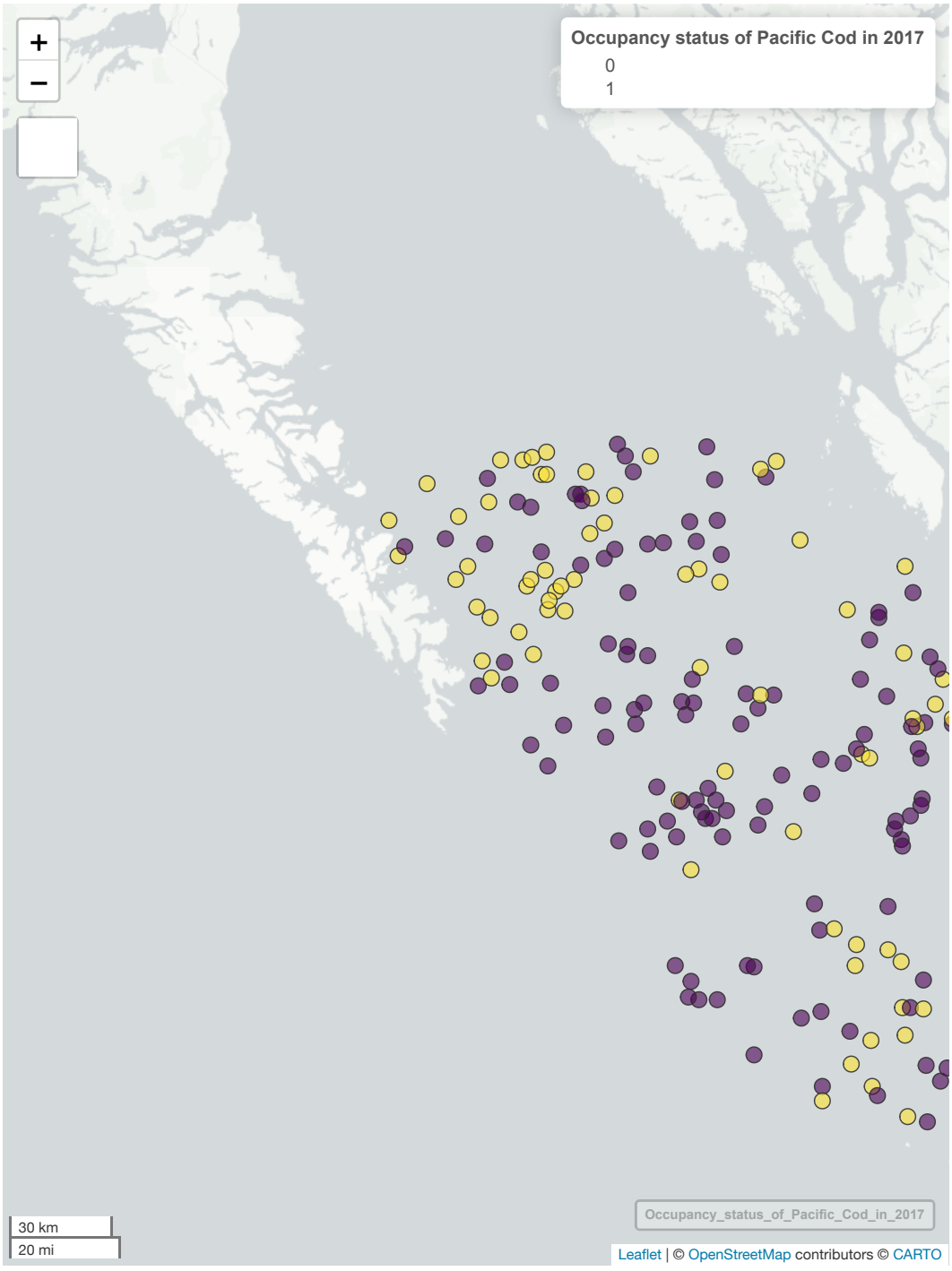
```
pcod_sf = st_transform(pcod_sf,
                       crs = "+proj=utm +zone=9 +datum=WGS84 +no_defs +type=crs +units=km" )
st_crs(pcod_sf)$units
```

```
[1] "km"
```

Spatial `sf` objects can be manipulated the same way we manipulate standard data frame objects via the `dplyr` package. For example, you can select a specific year using the `filter` function from `dplyr`. Let's map the present/absence of the Pacific Cod in 2017 using the `mapview` function:

```
pcod_sf %>%
  filter(year== 2017) %>%
  mutate(present = as.factor(present)) %>%
mapview(zcol = "present",
```

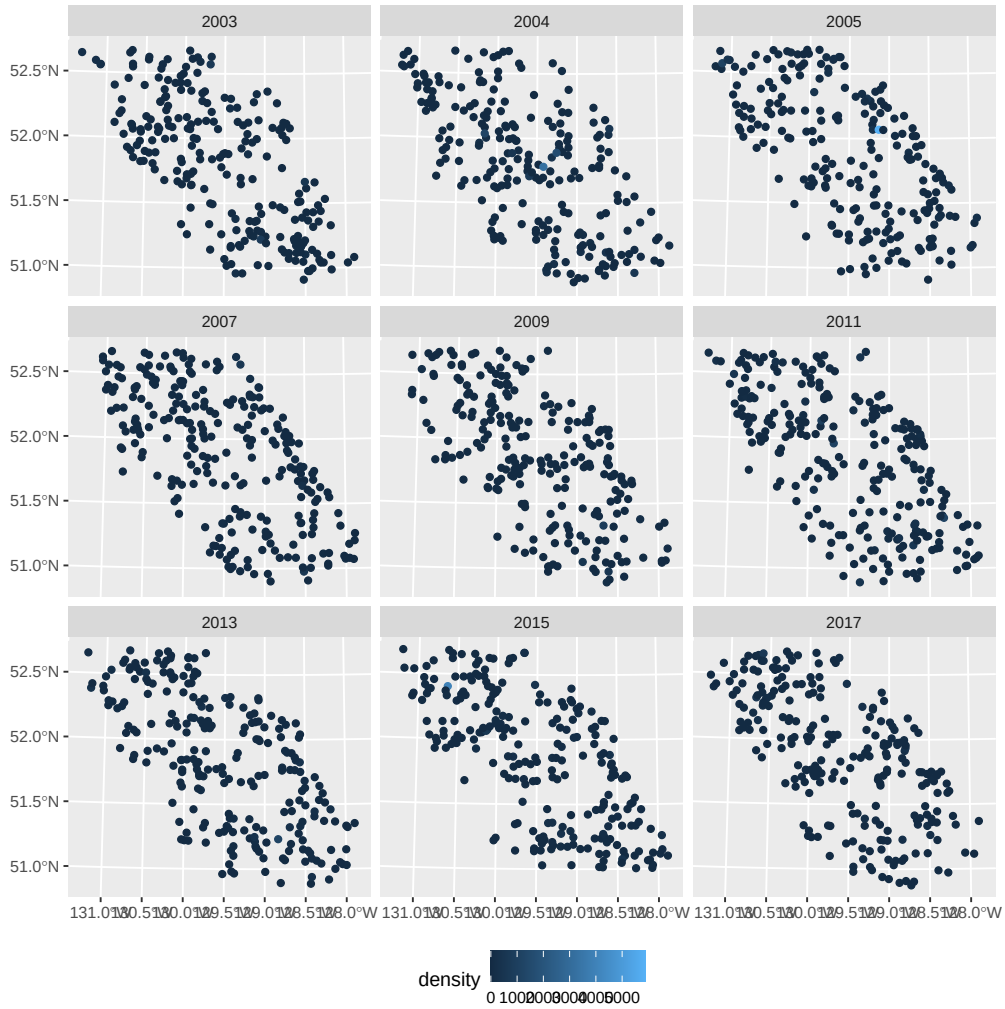
```
        layer.name = "Occupancy status of Pacific Cod in 2017")
```



Task

Use `ggplot` and the `sf` library to map the biomass density of the pacific cod across years.
hint
You can plot an`sf` object by adding a `geom_sf` layer to a ggplot object. You can also use the `facet_wrap` argument to plot an arrange of plots according to a grouping variable.
Click here to see the solution

```
ggplot()+
  geom_sf(data=pcod_sf,aes(color=density))+
  facet_wrap(~year)+
  theme(legend.position = "bottom")
```



2 Raster Data

Environmental data are typically stored in raster format, which represents spatially continuous phenomena by dividing a region into a grid of equally-sized cells, each storing a value for the variable of interest. In R, the `terra` package is a modern and powerful tool for efficiently working with raster data. The function `rast()`, can be used both to read raster files from standard formats (e.g., `.tif` or `.tiff`) and to create a new raster object from a data frame. For instance, the following code creates a raster from the `qcs_grid` grid data for Queen Charlotte Sound.

```
library(terra)
depth_r <- rast(qcs_grid, type = "xyz")
depth_r
```

```
class       : SpatRaster
size        : 102, 121, 3  (nrow, ncol, nlyr)
```

```
resolution  : 2, 2  (x, y)
extent      : 341, 583, 5635, 5839  (xmin, xmax, ymin, ymax)
coord. ref. :
source(s)   : memory
names       :    depth, depth_scaled, depth_scaled2
min values  :  12.0120,    -6.000040,  4.892624e-08
max values  : 805.7514,     3.453937,  3.600048e+01
```

The raster object contains three layers corresponding to the (i) depth values, (ii) the scaled depth values and (iii) the squared depth values.

Notice that there are no CRS associated with the raster. Thus, we can assign appropriate CRS using the `crs` function. Additionally, we also want the raster CRS to match the CRS in the survey data (recall that we have previously reprojected our data to utm coordinates). We can assign an appropiate CRS that matches the CRS of the `sf` object as follows:

```
crs(depth_r) <- crs(pcod_sf)
```

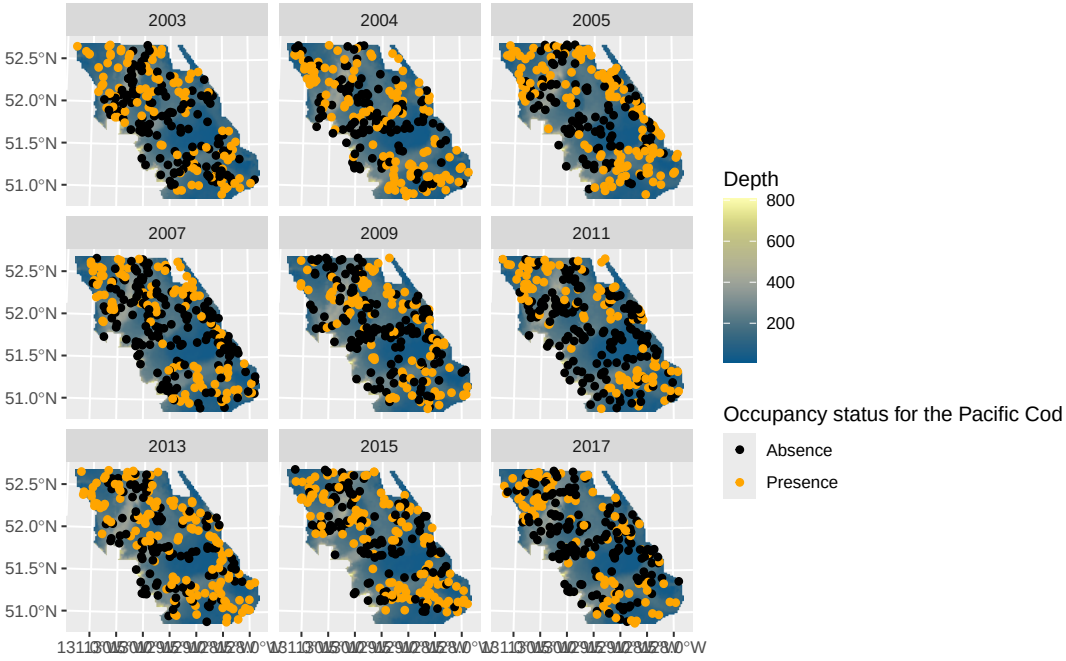
We can use the `tidyterra` package to plot raster data using `ggplot` by adding a `geom_spatraster` function and then select an appropriate `fill` and `color` palettes:

```
library(tidyterra)
```

```
Attaching package: 'tidyterra'
```

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

    filter
```

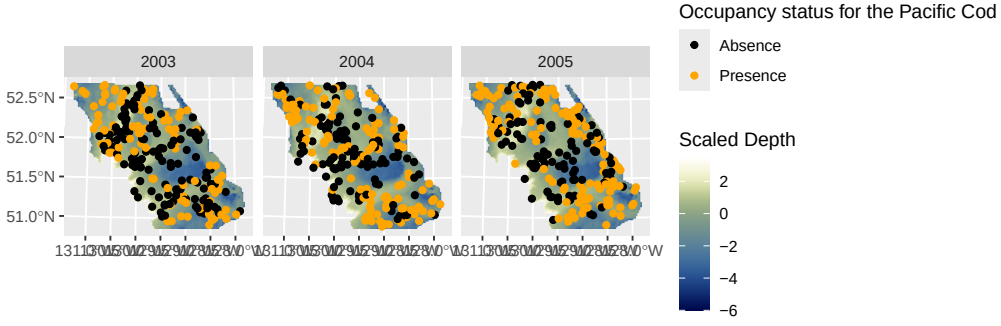
```
ggplot()+
  geom_spatraster(data=depth_r$depth)+
  geom_sf(data=pcod_sf,aes(color=factor(present))) +
  facet_wrap(~year)+
    scale_color_manual(name="Occupancy status for the Pacific Cod",
                       values = c("black","orange"),
                       labels= c("Absence","Presence"))+
  scale_fill_scico(name = "Depth",
                   palette = "nuuk",
                   na.value = "transparent" )
```



Task

Map the scaled depth and the presence/absence records of the Pacific cod for 2003 to 2005 only.
hint
The different layers of a raster can be accessed using the $ symbol.
Click here to see the solution

```
ggplot()+
  geom_spatraster(data=depth_r$depth_scaled)+
  geom_sf(data=pcod_sf %>% filter(year %in% 2003:2005),
          aes(color=factor(present)))+
  facet_wrap(~year)+
  scale_color_manual(name="Occupancy status for the Pacific Cod",
                     values = c("black","orange"),
                     labels= c("Absence","Presence"))+
    scale_fill_scico(name = "Scaled Depth",
                     palette = "davos",
                     na.value = "transparent" )
```



2

In this practical we will:

- Explore tools for spatial point pattern data wrangling and visualization.

First, lets load some useful libraries:

```
# For plotting
library(ggplot2)
library(scico) # for colouring palettes

# Data manipulation
library(dplyr)
```

3 Spatial Point processes data

In point processes we measure the locations where events occur and the coordinates of such occurrences are our data.

Point process models are probabilistic models that describe the likelihood of *patterns* of points that represent the random location of some event. A spatial point process is a set of locations that have been generated by some form of stochastic (random) mechanism. In other words, the point process is a random variable operating in continuous space, and we observe realisations of this variable as point patterns across space (and/or time).

Consider a fixed geographical region A. The set of locations at which events occur are denoted $\mathbf{s} = s_1, \dots, s_n$. We let $N(A)$ be the random variable which represents the number of events in region A.

Our primary interest is in measuring where events occur, so the locations are our data. We typically assume that a spatial point pattern is generated by an unique point process over the whole study area. This means that the delimitation of the study area will affect the observed point patters.

The observed distribution of points can be described based on the intensity of points within a delimited region. We can define the (first order) intensity of a point process as the expected number of events per unit area. This can also be thought of as a measure of the density of our points. In some cases, the intensity will be constant over space (homogeneous), while in other cases it can vary by location (inhomogeneous or heterogenous).

In the next example, we will explore tools for visualizing spatial point patterns. Specifically, we will map the spatial distribution of the Ringlet butterfly in Scotland's Cairngorms National Park (CNP).

3 BNM citizen science program

Citizen science initiatives have become an important source of information in ecological research, offering large volumes of species distribution data collected by volunters for multiple taxonomic groups across wide spatial and temporal scales

Butterflies for the New Millennium (BNM) is a large-scale monitoring scheme launched in the earlies 70's to keep track of butterflies' populations in the UK. With over 12 million butterflies sighting and more than 10,000 volunteers, this recording scheme has proven to be a successful program that has been used to assess long-term changes in the distributions of UK butterfly species.

Here we will focus on the distribution of the Ringlet butterfly species, which holds particular significance in environmental studies as one of the *Habitat specialists* species (UK Government, 2024). The data set consists of Ringlet butterfly presence-only records collected by volunteers in Scotland's Cairngorms National Park (CNP).

Reading shapefiles into R

First, we load the geographical region of interest which can be downloaded here (i.e., CNP boundaries). We can use thre `st_read` function from the `sf` library to load the `.shp` file by specifying the directory where you downloaded the files:

```
library(sf)
shp_SGC <-  st_read("datasets/SG_CairngormsNationalPark/SG_CairngormsNationalPark_2010.shp",
```

Then, we can use appropriate CRS for the UK (i.e., EPSG code: 27700) :

```
shp_SGC <- shp_SGC %>% st_transform(crs = 27700)
st_crs(shp_SGC)$units
```

```
[1] "m"
```

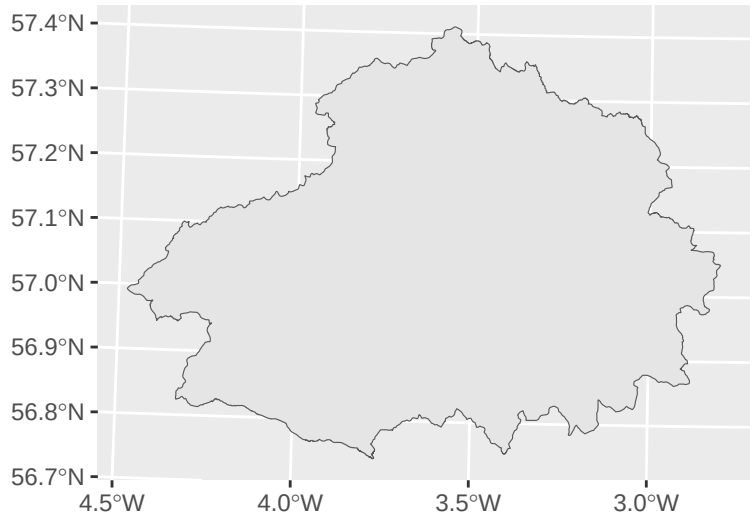
Notice that the spatial resolution is in meters. Let's change the spatial units to *km* to make resulting distance/area values more intuitive to interpret:

```
shp_SGC <- st_transform(shp_SGC,gsub("units=m","units=km",st_crs(shp_SGC)$proj4string))
st_crs(shp_SGC)$units
```

```
[1] "km"
```

We can then plot the CNP boundary as follows:

```
ggplot()+
  geom_sf(data=shp_SGC)
```



Creating sf spatial objects in R

Now we will read the Ringlet butterfly records which can be downloaded below:

```
ringlett <- read.csv("datasets/bnm_ringlett.csv")
head(ringlett)
```

```
         y         x
1 57.58752 -2.712498
2 54.97742 -3.274879
3 54.89929 -3.771451
4 55.40323 -5.737059
5 54.91438 -3.959336
6 55.87255 -4.167174
```

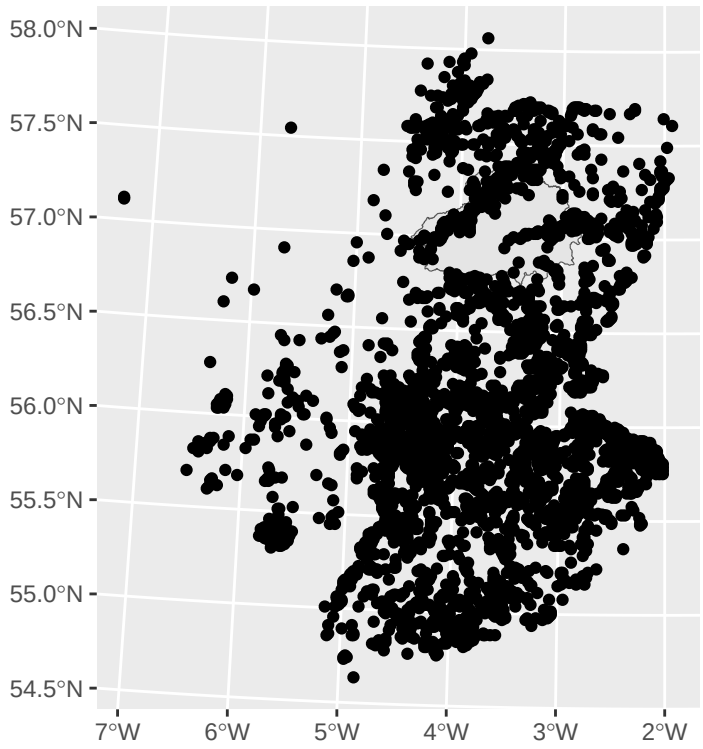
The data set contains the longitude latitude where an observation was made. We can convert this into a spatial `sf` object using the `st_as_sf` function by declaring the columns in our data that contain the spatial coordinates:

```
ringlett_sf <- ringlett %>% st_as_sf(coords = c("x","y"),crs = "+proj=longlat +datum=WGS84")
```

Task

We have set standard WGS84 coordinates for the Ringlet butterfly occurrence records. Set the CRS to match the CRS used in shapefile. Then, produce a map of the CNP region with the projected observations overlayed.
Take hint
You can use the `st_transform()` function to change the coordinates of an `sf` object (type `?st_transform` for more details)/
Click here to see the solution

```
ringlett_sf <- ringlett_sf %>%
  st_transform(st_crs(shp_SGC))

ggplot()+
  geom_sf(data=shp_SGC)+
  geom_sf(data=ringlett_sf)
```

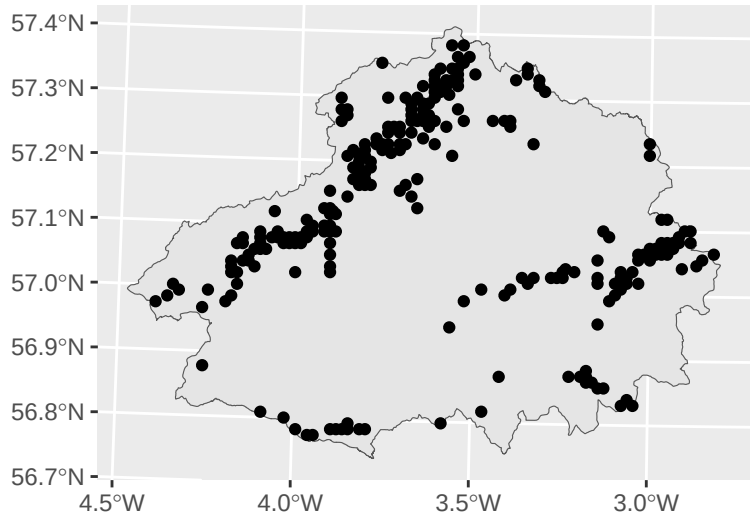


We can subset two `sf` objects with the **same** CRS in the same way as we subset a data frame in R. For example, if we want to subset the Ringlet butterfly occurrence records to those contained only within the CNP, we can type the following:

```
ringlett_CNP <- ringlett_sf[shp_SGC,] # crop to mainland
```

If we plot the `ringlett_CNP` object along with the CNP boundary, we should then obtain a map of the occurrence records within the park:

```
ggplot()+
  geom_sf(data=shp_SGC)+
  geom_sf(data=ringlett_CNP)
```

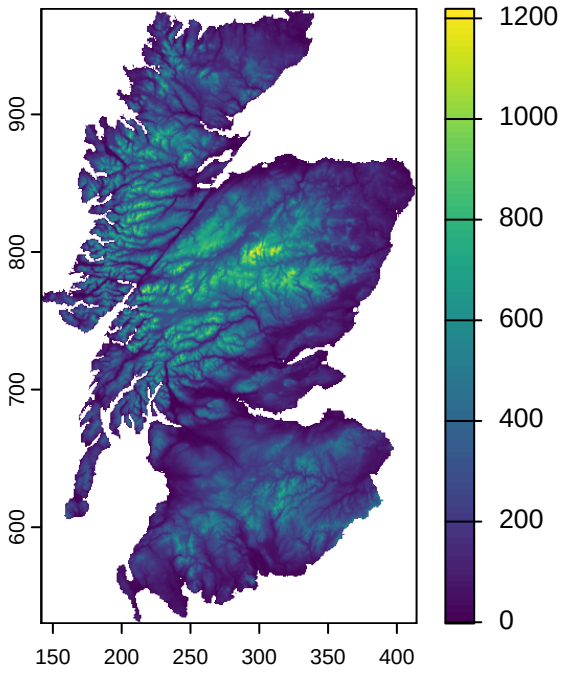


Reading Raster Data

We can use the `terra` R package to read raster files. The `Scotland_elev.tiff` raster contains the output of a digital elevation model for Scotland:

Once you download the raster you can read it using the `rast` function after specifying the path where the file has been stored. Then, we assign the same CRS as our data.

```
library(terra)
elevation_r <- rast("datasets/Scotland_elev.tiff")
crs(elevation_r) = crs(shp_SGC)
plot(elevation_r)
```

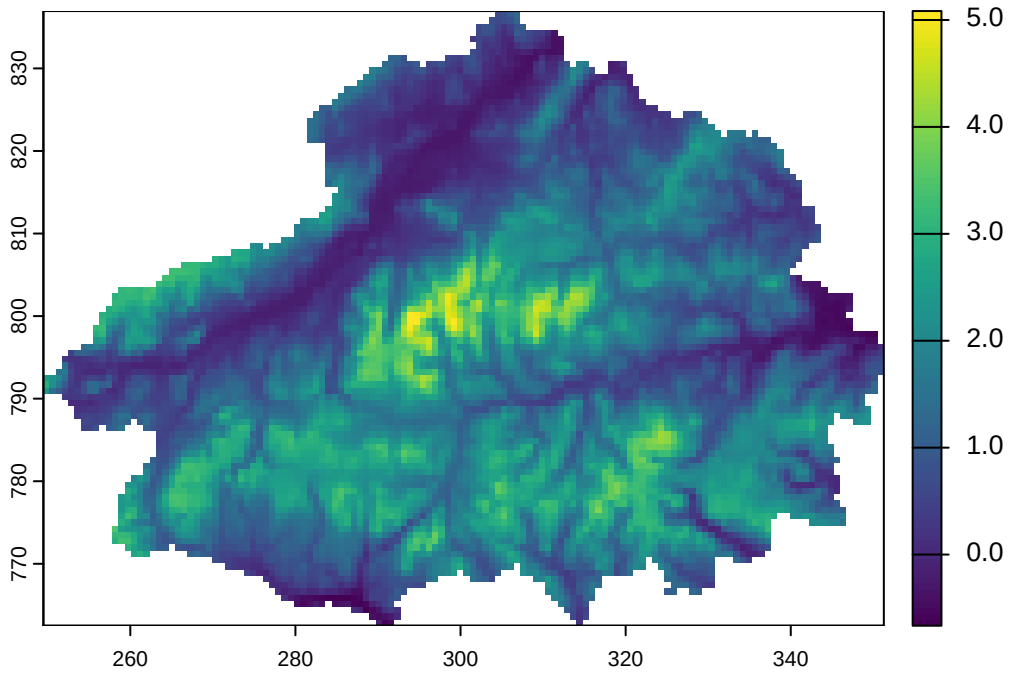


We can apply different R functions to our rasters. For example, we can scale the elevation values as follows:

```
elevation_r <- elevation_r %>% scale()
```

Lastly, we can crop the raster to the boundaries of our region of interest. Let's crop the elevation raster to the CNP area using the `crop` function:

```
elev_CNP <- terra::crop(elevation_r,shp_SGC,mask=T)
plot(elev_CNP)
```



Task

Using `tidyterra` and `ggplot`, produce a map of the elevation profile in the CNP and overlay the spatial point pattern of the Ringlet butterfly occurrence records. Use an appropriate colouring scheme for the elevation values. Do you see any pattern?

Take hint

You can use the `geom_spatraster()` to add a raster layer to a ggplot object. Furthermore the `scico` library contains a nice range of coloring palettes you can choose, type `scico_palette_show()` to see the color palettes that are available.

Click here to see the solution

```
library(scico)

ggplot()+
  tidyterra::geom_spatraster(data=elev_CNP)+
  geom_sf(data=ringlett_CNP)+
  scale_fill_scico(name = "Elevation scaled",
                   palette = "devon",
                   na.value = "transparent" )
```