Interpolation_peatlands

MarteFandrem

8 2 2022

#Packages needed

```
library(readxl)
library(writexl)
library(rgdal)
```

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

```
## Please note that rgdal will be retired by the end of 2023,
## plan transition to sf/stars/terra functions using GDAL and PROJ
## at your earliest convenience.
##
## rgdal: version: 1.5-27, (SVN revision 1148)
## Geospatial Data Abstraction Library extensions to R successfully loaded
## Loaded GDAL runtime: GDAL 3.2.1, released 2020/12/29
## Path to GDAL shared files: \\home.ansatt.ntnu.no/martef/Documents/R/win-library/4.1/rgdal/gdal
## GDAL binary built with GEOS: TRUE
## Loaded PROJ runtime: Rel. 7.2.1, January 1st, 2021, [PJ_VERSION: 721]
## Path to PROJ shared files: \\home.ansatt.ntnu.no/martef/Documents/R/win-library/4.1/rgdal/proj
## PROJ CDN enabled: FALSE
## Linking to sp version:1.4-5
## To mute warnings of possible GDAL/OSR exportToProj4() degradation,
## use options("rgdal_show_exportToProj4_warnings"="none") before loading sp or rgdal.
## Overwritten PROJ_LIB was \\home.ansatt.ntnu.no/martef/Documents/R/win-library/4.1/rgdal/proj
```

```
library(raster)
library(ggplot2)
library(gstat)
library(sf)
```

```
## Linking to GEOS 3.9.1, GDAL 3.2.1, PROJ 7.2.1
```

```
library(broom)
library(ggthemes)
library(viridis)
```

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

```
library(sp)
library(spatialEco)
```

```
##
## Attaching package: 'spatialEco'

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

```
library(spm)
library(tmap)
library(Metrics)
library(rlist)
library(dplyr)
```

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

## The following object is masked from 'package:spatialEco':
##
##     combine

## 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(tmaptools)
library(shinyjs)
```

```
##
## Attaching package: 'shinyjs'

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

## The following object is masked from 'package:sp':
##
##     show

## The following objects are masked from 'package:methods':
##
##     removeClass, show
```

```
library(rgeos)
```

```
## rgeos version: 0.5-8, (SVN revision 679)
##  GEOS runtime version: 3.9.1-CAPI-1.14.2
##  Please note that rgeos will be retired by the end of 2023,
## plan transition to sf functions using GEOS at your earliest convenience.
##  GEOS using OverlayNG
##  Linking to sp version: 1.4-5
##  Polygon checking: TRUE
```

```
library(automap)
library(ggbreak)
```

```
## ggbreak v0.0.9
##
## If you use ggbreak in published research, please cite the following
## paper:
##
## S Xu, M Chen, T Feng, L Zhan, L Zhou, G Yu. Use ggbreak to effectively
## utilize plotting space to deal with large datasets and outliers.
## Frontiers in Genetics. 2021, 12:774846. doi: 10.3389/fgene.2021.774846
```

#Import and clean up data

```
#Import
setwd("C:/Users/martef/DokumenterIntern/GitHub/PhDGRAN")
shp <- readOGR(dsn="Data/Geilo", layer="geilo-dybdef")
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "C:\Users\martef\DokumenterIntern\GitHub\PhDGRAN\Data\Geilo", layer: "geilo-dybdef"
## with 13 features
## It has 21 fields
## Integer64 fields read as strings:  POLY_ POLY_ID
```

```
df <- read.csv("Data/Geilo/torvdybder.csv", sep=";")

#Make spatial
dfs <- st_as_sf(x = df,
                        coords = c("x", "y"),
                        crs = "+init=epsg:25832")
```

```
## Warning in CPL_crs_from_input(x): GDAL Message 1: +init=epsg:XXXX syntax is
## deprecated. It might return a CRS with a non-EPSG compliant axis order.
```

```
#Check projections
sf::st_crs(shp)
```

```
## Coordinate Reference System: NA
```

```
sf::st_crs(dfs)
```

```
## Coordinate Reference System:
##   User input: +init=epsg:25832
##   wkt:
## PROJCRS["ETRS89 / UTM zone 32N",
##     BASEGEOGCRS["ETRS89",
##         DATUM["European Terrestrial Reference System 1989",
##             ELLIPSOID["GRS 1980",6378137,298.257222101,
##                 LENGTHUNIT["metre",1]]],
##         PRIMEM["Greenwich",0,
##             ANGLEUNIT["degree",0.0174532925199433]],
##         ID["EPSG",4258]],
##     CONVERSION["UTM zone 32N",
##         METHOD["Transverse Mercator",
##             ID["EPSG",9807]],
##         PARAMETER["Latitude of natural origin",0,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8801]],
##         PARAMETER["Longitude of natural origin",9,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8802]],
##         PARAMETER["Scale factor at natural origin",0.9996,
##             SCALEUNIT["unity",1],
##             ID["EPSG",8805]],
##         PARAMETER["False easting",500000,
##             LENGTHUNIT["metre",1],
##             ID["EPSG",8806]],
##         PARAMETER["False northing",0,
##             LENGTHUNIT["metre",1],
##             ID["EPSG",8807]],
##         ID["EPSG",16032]],
##     CS[Cartesian,2],
##         AXIS["(E)",east,
##             ORDER[1],
##             LENGTHUNIT["metre",1,
##                 ID["EPSG",9001]]],
##         AXIS["(N)",north,
##             ORDER[2],
##             LENGTHUNIT["metre",1,
##                 ID["EPSG",9001]]],
##     USAGE[
##         SCOPE["unknown"],
##         AREA["Europe between 6Â°E and 12Â°E: Austria; Belgium; Denmark - onshore and offshore; Germa
##         BBOX[38.76,6,83.92,12]]]
```

```
#Set same projection on all files
#crs(shp) <- CRS('+init=EPSG:32632') #Use either this or next one

proj4string(shp)<-crs(dfs)


# Make sf file from sp shapefile
```

```
sf_shp <- st_as_sf(shp)

#make spatial file from data frame file
dfsp <- as(dfs, Class="Spatial")


#Repair geometry if needed
#sf_shp <- st_make_valid(sf_shp)
#sf_shp_myr <- st_make_valid(sf_shp_myr)
```

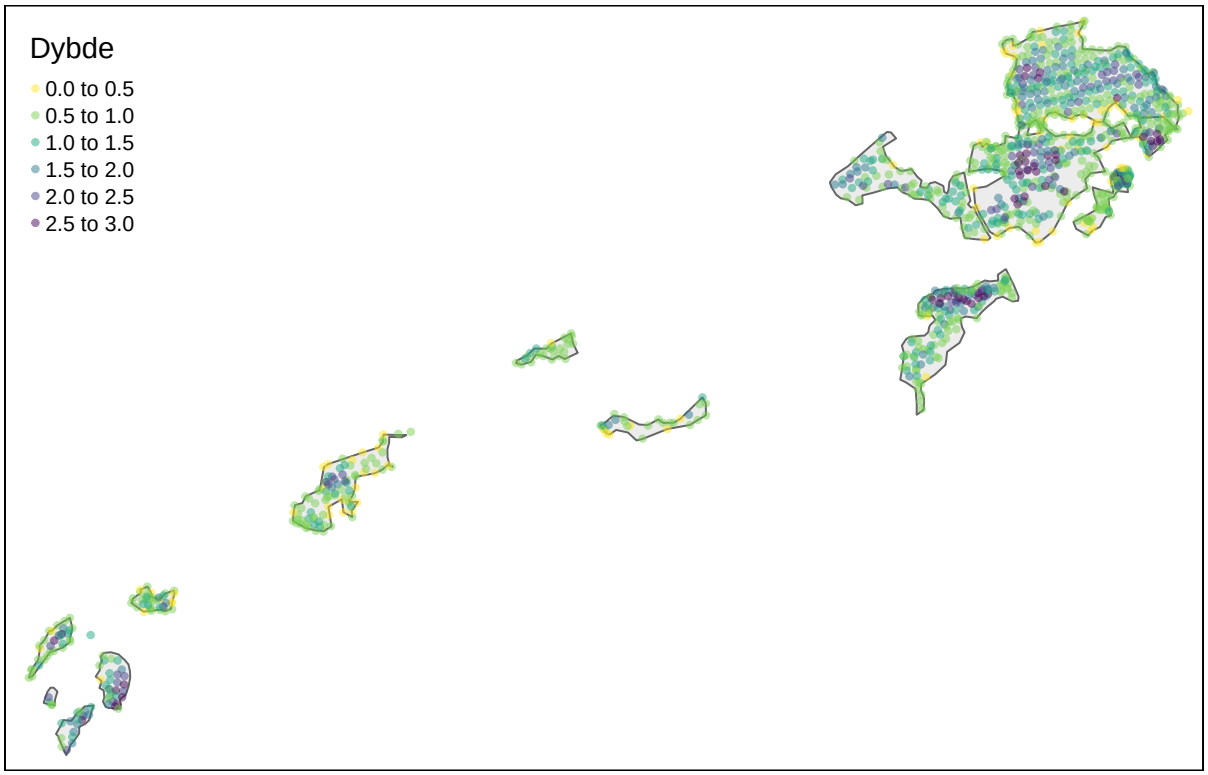
#Visualize the data

```
tmap_mode("plot")
```

```
## tmap mode set to plotting
```

```
   tm_shape(sf_shp)+
     tm_fill(alpha=0.5) +
     tm_polygons() +
   tm_shape(dfsp)+
     tm_dots(col="Dybde", alpha=0.5, palette="-viridis", size=0.05 )
```

```
## Warning: One tm layer group has duplicated layer types, which are omitted. To
## draw multiple layers of the same type, use multiple layer groups (i.e. specify
## tm_shape prior to each of them).
```



#Create grid and adjust to extent of peatland

```
grid <- raster(extent(shp)) #create a raster grid from the extent of the peatland
res(grid) <- 1              #set resolution of the grid to 1x1m
proj4string(grid)<-crs(dfs) #set similar projection to the grid as to the datapoints

grid_sp <-as(grid, "SpatialPixels") #convert the grid from raster to spatialpixels

grid_crop <- grid_sp[shp,]  #crop the grid to only include the peatland
```

#Interpolate volume Run several interpolations to find mean and range of volume of peat

```
neighbors = length(dfsp$Dybde)
power = c(seq(from = 1, to = 4, by = 1))
neigh = c((1), seq(from=2,to=30,by = 2), c(length=(neighbors)))

temp <- data.frame()

for (i in power) {
  for (j in neigh) {

    temp2 <- NULL
    temp3 <- NULL
    temp4 <- NULL

    run = paste(i, j, sep="_")

    temp2 <- idw(Dybde ~ 1, dfsp, grid_crop, nmax=j, idp=i)
    temp3 <- as.data.frame(temp2@data)
    temp4 <- sum(temp3$var1.pred)
    temp5 <- cbind(run, temp4)
    temp  <- rbind(temp, temp5)
  }
}
```

```
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
```

```
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
## [inverse distance weighted interpolation]
```

```
volume <- temp
volume <-dplyr::rename(volume, volume=temp4)
volume <- tidyr::separate(volume,
```

```
                        run,
                        into = c("power", "nn"),
                        sep = "_",
                        remove=F)
volume$power <- as.numeric(volume$power)
volume$nn <- as.numeric(volume$nn)
volume$volume <- as.numeric(volume$volume)
```

#Values for printing (mean, min, max, SD)

```
max <- max(volume$volume)
min <- min(volume$volume)
mean <- mean(volume$volume)
sd <- sd(volume$volume)


Description <- c("mean", "min", "max", "SD")
Results_volume <- data.frame(Description, Results = c(mean, min, max, sd))
```

#Visualize the interpolation

```
idw_map <- tmap_mode("plot") +
             tm_shape(idw(Dybde ~ 1, dfsp, grid_crop, nmax=8, idp=3))+
              tm_raster(title= "Torvdybde (m)", col="var1.pred", palette="-viridis") +
            tm_shape(sf_shp)+
              tm_borders()   +
              tm_compass(type="8star", position = c("right", "bottom"), size = 2) +
              tm_scale_bar(position = c("right", "bottom"), width = 0.3) +
                 tm_layout(inner.margins = c(0.2, 0.1, 0.1, 0.1), legend.show = TRUE, legend.position =
```

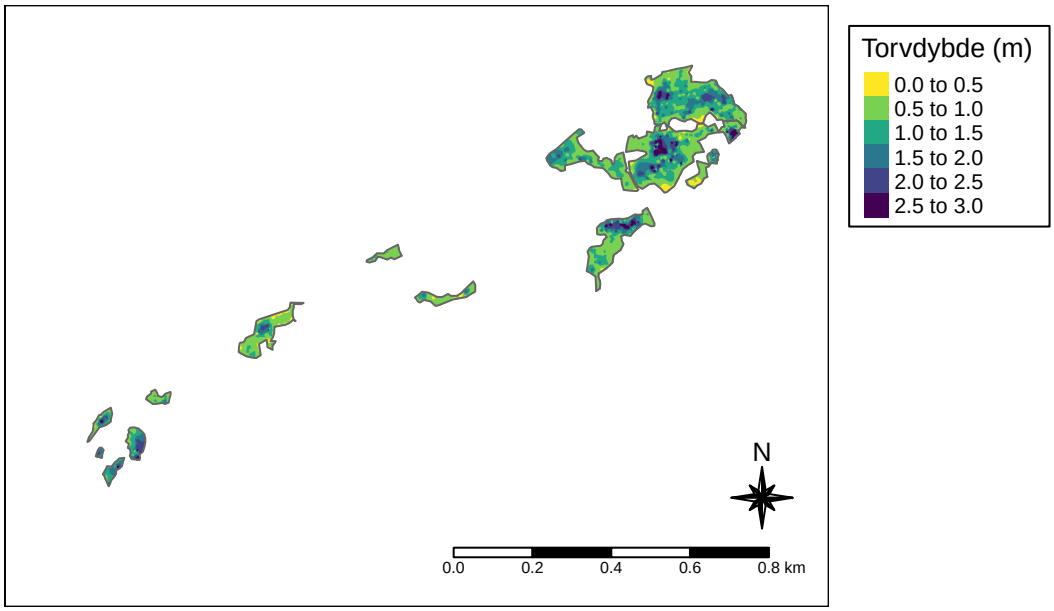
```
## tmap mode set to plotting

## [inverse distance weighted interpolation]
```

```
idw_map
```

```
## Warning in sp::proj4string(obj): CRS object has comment, which is lost in output

## stars object downsampled to 1251 by 799 cells. See tm_shape manual (argument raster.downsample)
```


Torvdybde (m)
0.0 to 0.5
0.5 to 1.0
1.0 to 1.5
1.5 to 2.0
2.0 to 2.5
2.5 to 3.0
N
0.0
0.2
0.4
0.6
0.8 km