Final Project Working Doc

Matt Thacker

April 13, 2018

Applying methodology from Devred et al. (2009) to attempt to more accurately delineate oceanic ecological provinces in the Northeast Atlantic. Kmeans classification of MODIS chlorophyll a and sea surface temperature along with bathymetry data, physical location, and original ecological province. Does the scaling translate? Does the methodology more generally?

Read data and create necessary raster layers for analysis

```
library(raster)
```

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

```
library(rgdal)
```

```
## rgdal: version: 1.5-8, (SVN revision 990)
## Geospatial Data Abstraction Library extensions to R successfully loaded
## Loaded GDAL runtime: GDAL 3.0.4, released 2020/01/28
## Path to GDAL shared files: C:/Users/matth/Documents/R/win-library/4.0/rgdal/gdal
## GDAL binary built with GEOS: TRUE
## Loaded PROJ runtime: Rel. 6.3.1, February 10th, 2020, [PJ_VERSION: 631]
## Path to PROJ shared files: C:/Users/matth/Documents/R/win-library/4.0/rgdal/proj
## Linking to sp version:1.4-2
## To mute warnings of possible GDAL/OSR exportToProj4() degradation,
## use options("rgdal_show_exportToProj4_warnings"="none") before loading rgdal.
```

```
library(scales)
library(proj4)
```

```
##
## Attaching package: 'proj4'
```

```
## The following object is masked from 'package:rgdal':
##
##     project
```

```
library(ggplot2)
```

```
##read in data
#chlorophyl a
```

```r
chl_a <- raster("Images/Jul2017Mean/Chl_a/chl_subset3.rp.tif")

##optional line commented out which subsets image to remove boreal polar province from longhurst
#chl_a <- crop(chl_a, extent(chl_a, 1, 447, 16, 728))

#sea surface temperature
SST <- raster("Images/Jul2017Mean/SST/sst_subset.rp.tif")
#match proj and extent of modis data
SST <- projectRaster(SST, chl_a)
```

```
## Warning in rgdal::rawTransform(projfrom, projto, nrow(xy), xy[, 1], xy[, : Using
## PROJ not WKT2 strings
```

```
## Warning in rgdal::rawTransform(projto_int, projfrom, nrow(xy), xy[, 1], : Using
## PROJ not WKT2 strings
```

```r
#bathmetry
bath <- raster("Images/Jul2017Mean/Bathymetry/depth_resamp.tif")
#match proj and extent of modis data
bath <- projectRaster(bath, chl_a)
```

```
## Warning in rgdal::rawTransform(projfrom, projto, nrow(xy), xy[, 1], xy[, : Using
## PROJ not WKT2 strings
```

```
## Warning in rgdal::rawTransform(projfrom, projto, nrow(xy), xy[, 1], xy[, : Using
## PROJ not WKT2 strings
```

```r
#longhurst ecological provinces
#vals 1-5, need to divide by 4 to get scaling
prov <- raster("longhurst/rc_provras.tif")
#match proj and extent of modis data
prov<- projectRaster(prov, chl_a, method = "ngb")
```

```
## Warning in rgdal::rawTransform(projfrom, projto, nrow(xy), xy[, 1], xy[, : Using
## PROJ not WKT2 strings
```

```
## Warning in rgdal::rawTransform(projfrom, projto, nrow(xy), xy[, 1], xy[, : Using
## PROJ not WKT2 strings
```

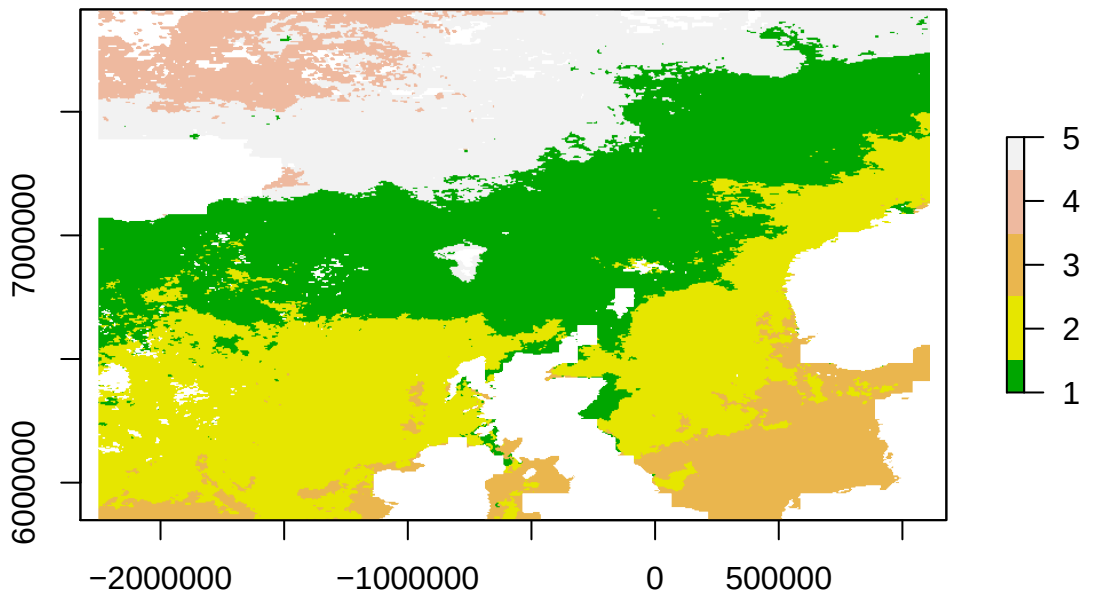
```r
##calculate important variables/clean data
#loop through to set all chl_a values above 10 to 10
chl_a[chl_a > 10] <- 10

#all land set to NA in bathymetry
bath[bath > 0] <- NA
bath <- abs(bath)
bath_log <- log10(bath)

#log chl
chl_log <- log10(chl_a)
```

```
##get lats/longs from projected coordinates
#gcs prj4 string
llprj <-  "+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs +towgs84=0,0,0"

#projected coordinates in radians
Pcoords <- xyFromCell(chl_a, 1:length(chl_a))

#lats/longs
coords <- ptransform(Pcoords, src.proj = crs(chl_a), dst.proj = llprj)

 #degrees
coords <- coords * 180/pi

#aribtratily shift x coords to get rid of negatives for logs
coords[,1] <- coords[,1]+abs(min(coords[,1]))

#drop empty z column, take log
coords_log <- log10(coords[,1:2])
#set NA vals to NA
coords_log[coords_log[,1] == -Inf, 1] <- NA

#create lat and long rasters to stack for NA synchro
Xras <- raster(extent(chl_a), nrow=nrow(chl_a), ncol=ncol(chl_a), crs=crs(chl_a), vals = coords_log[,1]
Yras <- raster(extent(chl_a), nrow=nrow(chl_a), ncol=ncol(chl_a), crs=crs(chl_a), vals = coords_log[,2]
```

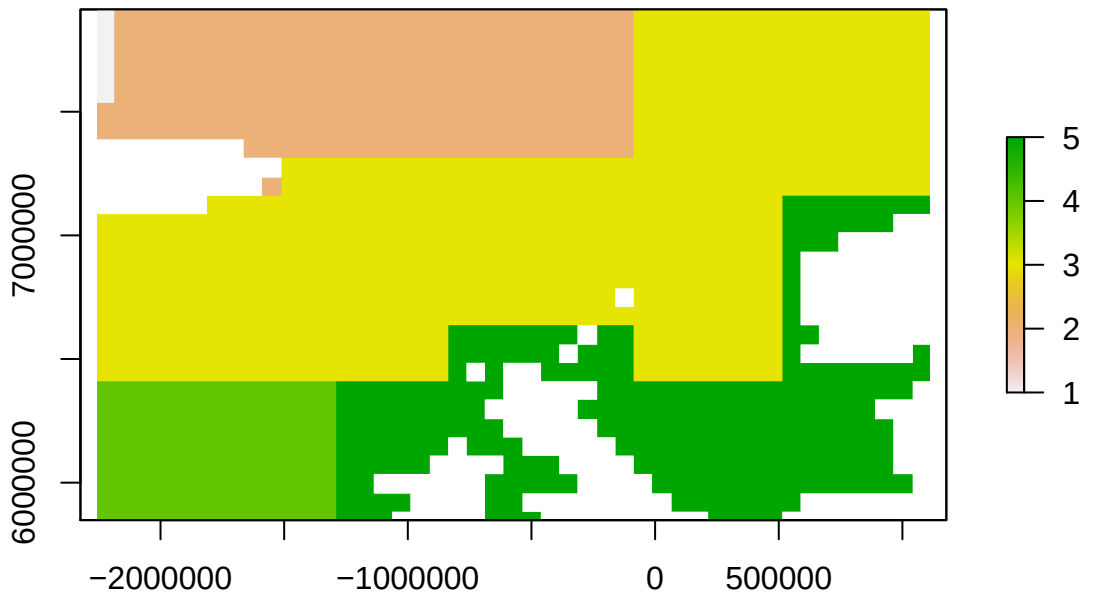
Devred et al. scaling

```
##rescale values using paper weights
#chl_a
chl.vec <- matrix(chl_log)
chl.scale.1 <- rescale(chl.vec, c(0,4))
chl.scale.ras.1 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), va

#sea surface temp
sst.vec<- matrix(SST)
sst.scale.1 <- rescale(sst.vec, c(0,10))
sst.scale.ras.1 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), va


#bathymetry
bath.vec <- matrix(bath_log)
bath.scale.1 <- rescale(bath.vec, c(0,2))
bath.scale.ras.1 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), va

#lats/longs
Xvec <- matrix(Xras)
X.scale.1 <- rescale(Xvec, c(0,1))
X.scale.ras.1 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), vals=

Yvec <- matrix(Yras)
Y.scale.1 <- rescale(Yvec, c(0,1))
Y.scale.ras.1 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), vals=
```

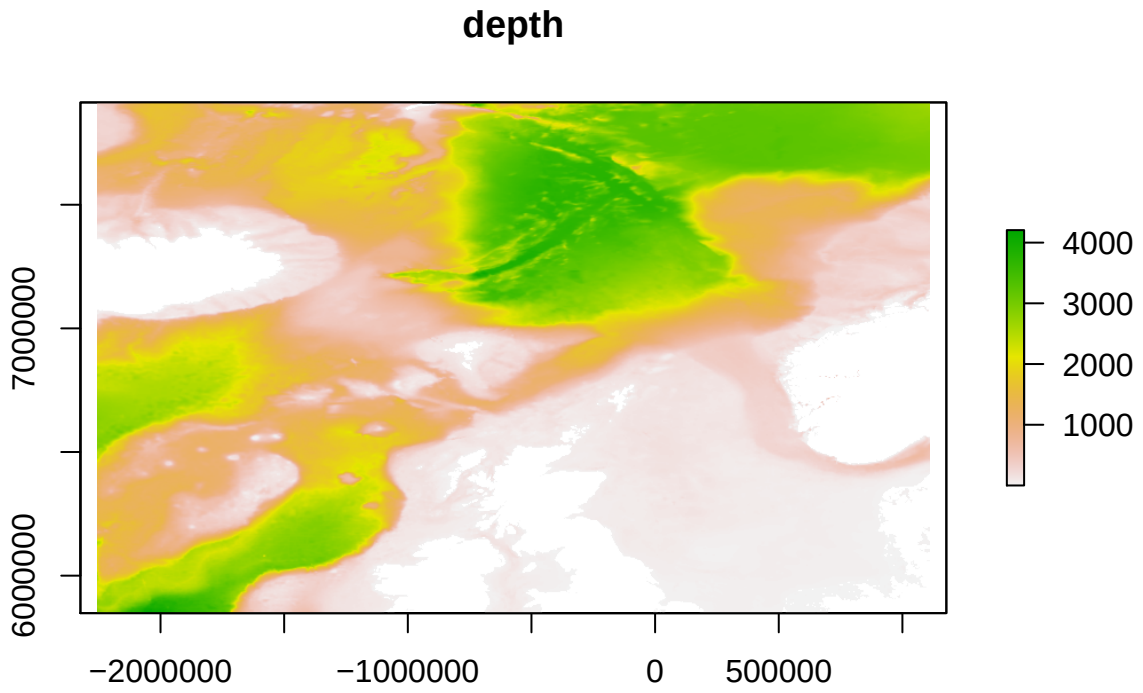
```r
#ecological provinces
prov.vec<- matrix(prov)
prov.scale.1 <- rescale(prov.vec, c(.25, 1))
prov.scale.ras.1 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), va

##create stack to synchronise NA values
#create stack
s <- stack(c(sst.scale.ras.1, chl.scale.ras.1, bath.scale.ras.1, prov.scale.ras.1, X.scale.ras.1, Y.scal

#synchronize the NA values
stack.1 <- mask(s, calc(s,fun = sum))
names(stack.1) <- c("SST", "Chl.log", "Depth.log", "Province", "X.log", "Y.log")


##K means classification
#data.frame from raster stack
Kmat.1 <- as.data.frame(stack.1)

#cluster output, no NAs
cluster.out <- kmeans(na.omit(Kmat.1), 5)

#vector to hold classification results
Kmat.df.factor <- rep(NA, length(Kmat.1[,1]))

#put results into vector
Kmat.df.factor[!is.na(Kmat.1[,1])] <- cluster.out$cluster

## fill the empty raster with the class results
#new raster
output.1 <- raster(stack.1)
#set values from classification
output.1 <- setValues(output.1, Kmat.df.factor)

##visualize results and input vars
plot(output.1, col = terrain.colors(5))
```



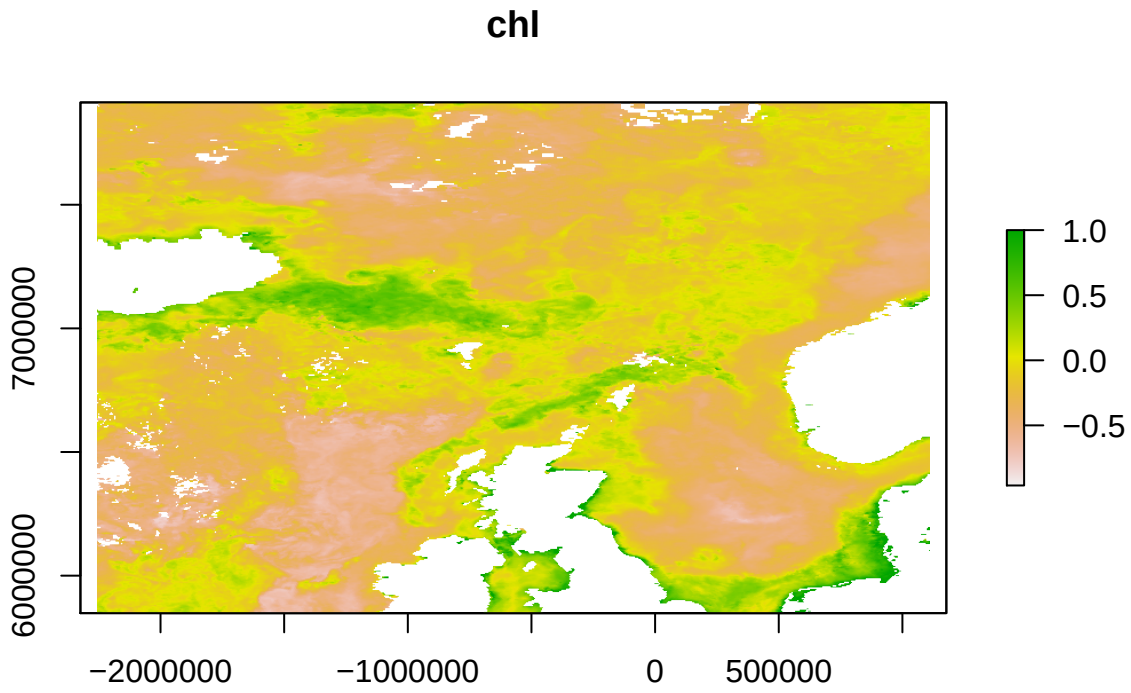

```
plot(prov)
```



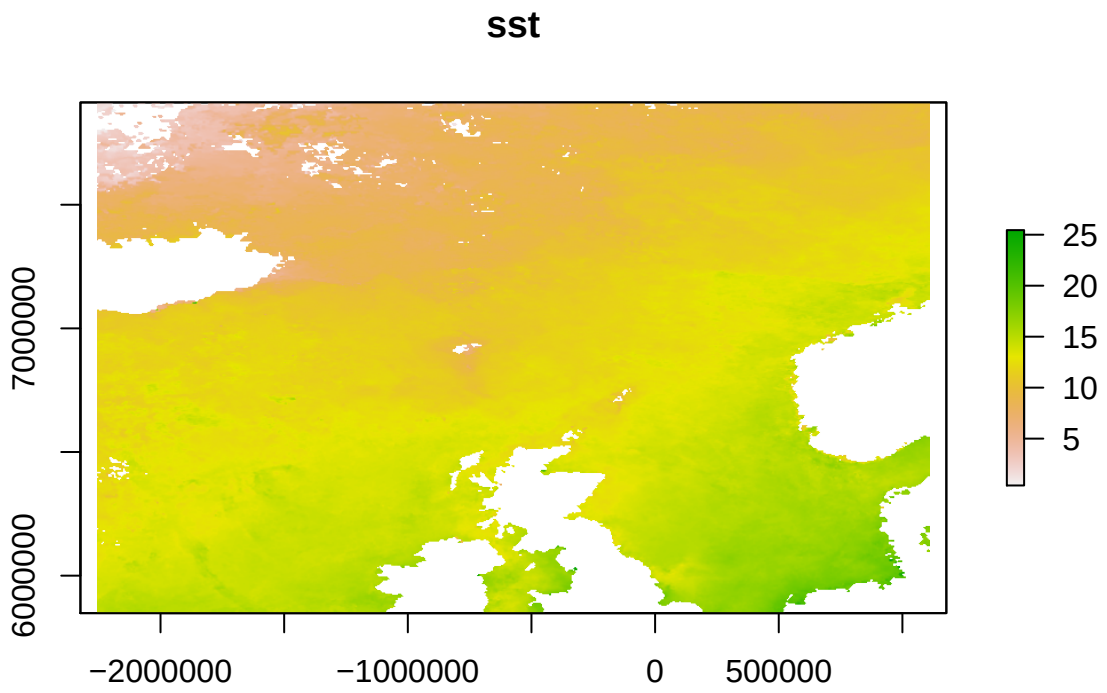

```
plot(bath, main = "depth")
```




```
plot(chl_log, main= "chl")
```



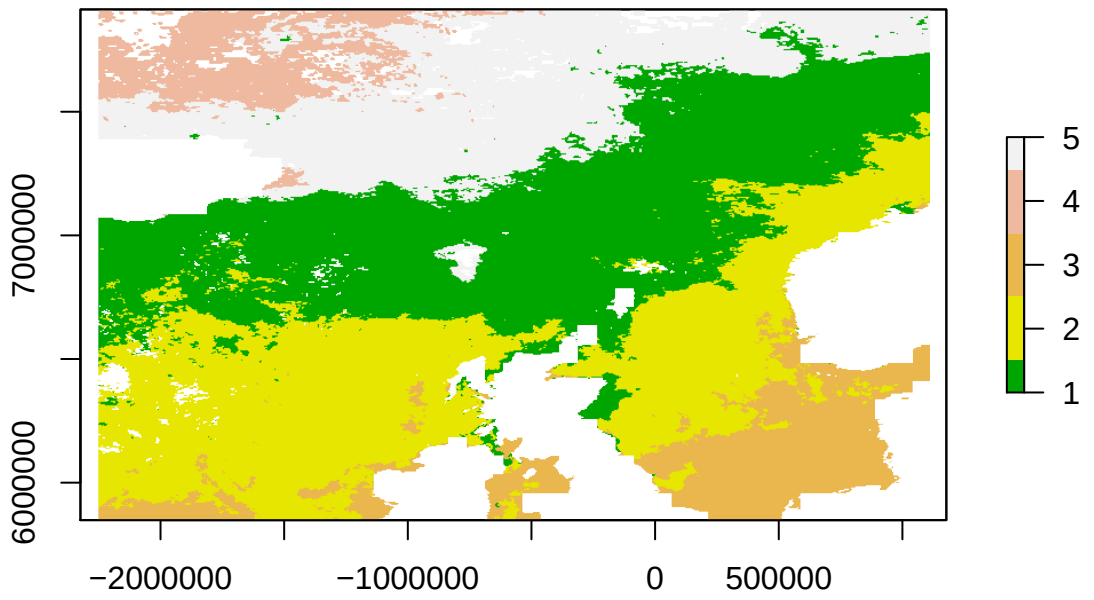

```
plot(SST, main = "sst")
```



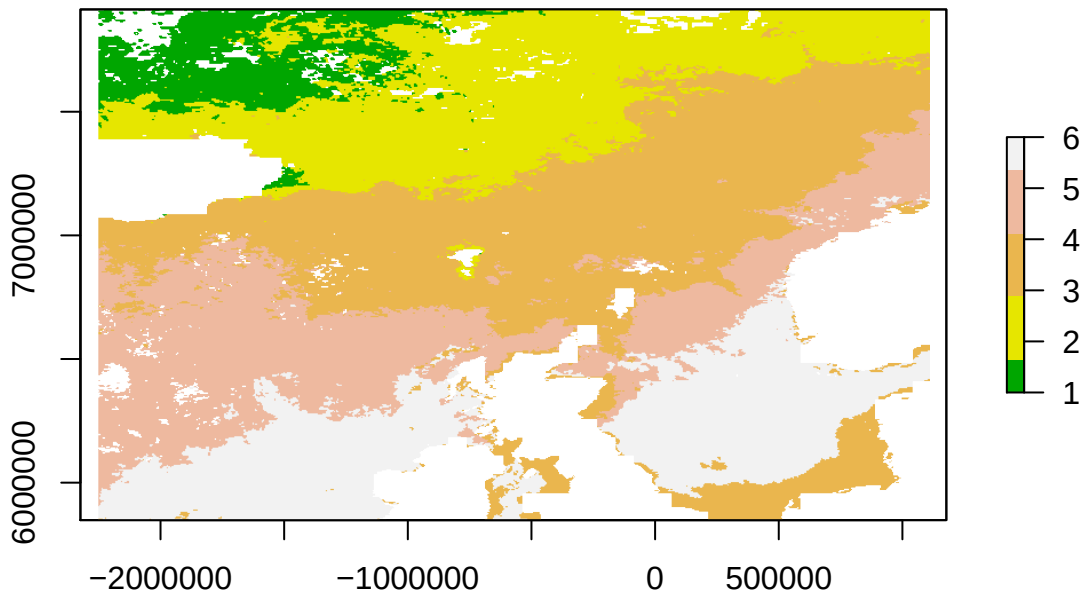
Non optimal scaling, left in for posterity and to show the process

```
##rescale values 2nd try, up bath, up prov, down sst
#chl_a
chl.vec <- matrix(chl_log)
chl.scale.2 <- rescale(chl.vec, c(0,4))
chl.scale.ras.2 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), val

#sea surface temp
sst.vec<- matrix(SST)
sst.scale.2 <- rescale(sst.vec, c(0,8))
sst.scale.ras.2 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), val

#bathymetry
bath.vec <- matrix(bath_log)
bath.scale.2 <- rescale(bath.vec, c(0,2))
bath.scale.ras.2 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), va

#lat/lon
Xvec <- matrix(Xras)
X.scale.2 <- rescale(Xvec, c(0,2))
X.scale.ras.2 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), vals=

Yvec <- matrix(Yras)
Y.scale.2 <- rescale(Yvec, c(0,2))
Y.scale.ras.2 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), vals=
```

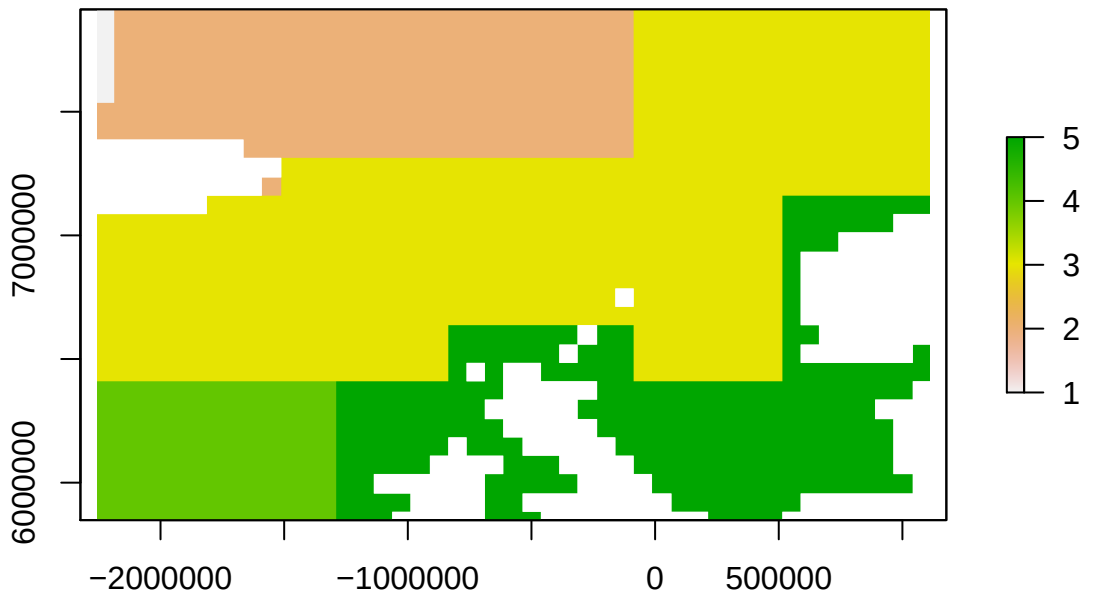
```r
#ecological provinces
prov.vec<- matrix(prov)
prov.scale.2 <- rescale(prov.vec, c(0.25,1))
prov.scale.ras.2 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), va

##create stack to synchronise NA values
#create stack
s <- stack(c(sst.scale.ras.2, chl.scale.ras.2, bath.scale.ras.2, prov.scale.ras.2, X.scale.ras.2, Y.scal

# Synchronize the NA values
stack.2 <- mask(s, calc(s,fun = sum))
names(stack.2) <- c("SST", "Chl.log", "Depth.log", "Province", "X.log", "Y.log")


##Kmeans classification
#data.frame from raster stack
Kmat.2 <- as.data.frame(stack.2)

#cluster output, no NAs
cluster.out <- kmeans(na.omit(Kmat.2), 6)

#vector to hold classification results
Kmat.df.factor <- rep(NA, length(Kmat.2[,1]))

#put results into vector
Kmat.df.factor[!is.na(Kmat.2[,1])] <- cluster.out$cluster

## fill the empty raster with the class results
#new raster
output.2 <- raster(stack.2)
#set values
output.2 <- setValues(output.2, Kmat.df.factor)

##visualize it
plot(output.1, col = sort(terrain.colors(5)))
```




```
plot(output.2, col = sort(terrain.colors(5)))
```



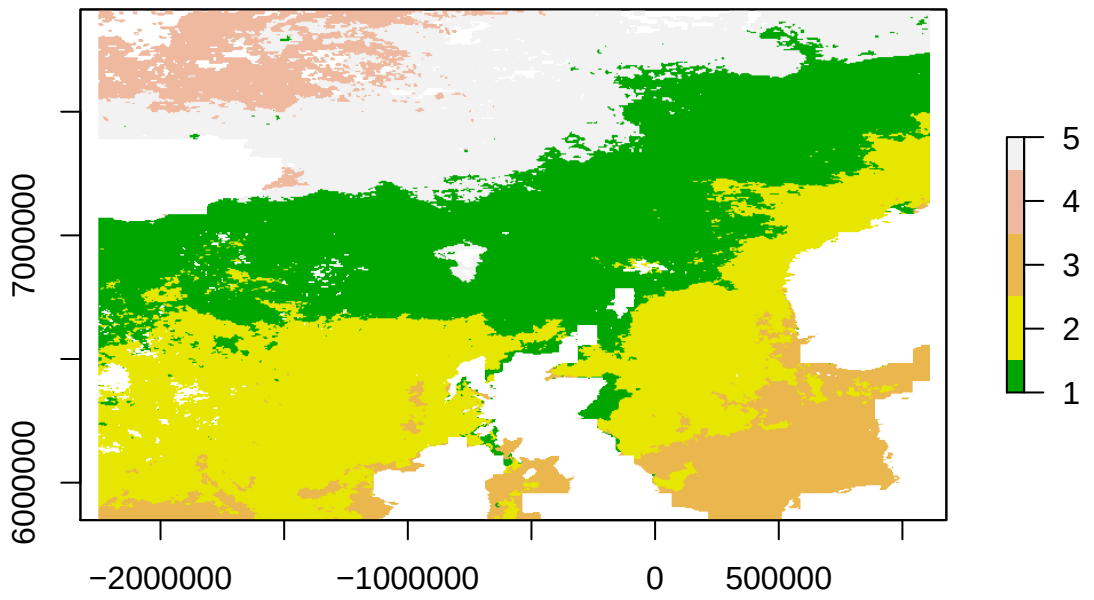

```
plot(prov)
```



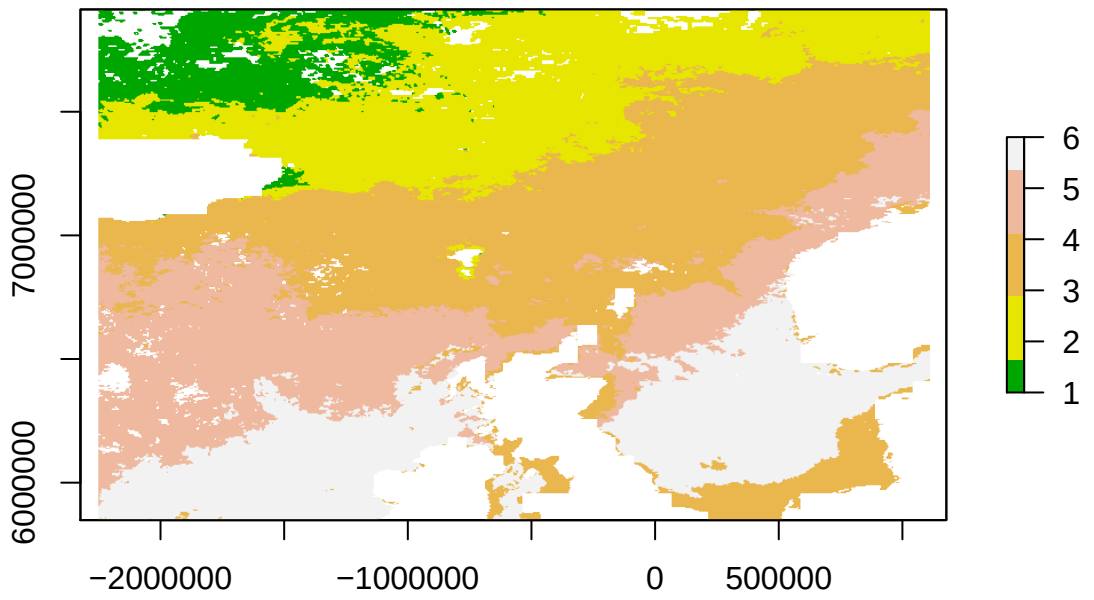
All parameters scaled to same range

```
##rescale values 3rd try, even scaling
#chlorophyll a
chl.vec <- matrix(chl_log)
chl.scale.3 <- rescale(chl.vec, c(0,4))
chl.scale.ras.3 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), val

#sea surface temperature
sst.vec<- matrix(SST)
sst.scale.3 <- rescale(sst.vec, c(0,4))
sst.scale.ras.3 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), val

#bathymetry
bath.vec <- matrix(bath_log)
bath.scale.3 <- rescale(bath.vec, c(0,4))
bath.scale.ras.3 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), va

#lats/longs
Xvec <- matrix(Xras)
X.scale.3 <- rescale(Xvec, c(0,4))
X.scale.ras.3 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), vals=

Yvec <- matrix(Yras)
Y.scale.3 <- rescale(Yvec, c(0,4))
Y.scale.ras.3 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), vals=
```

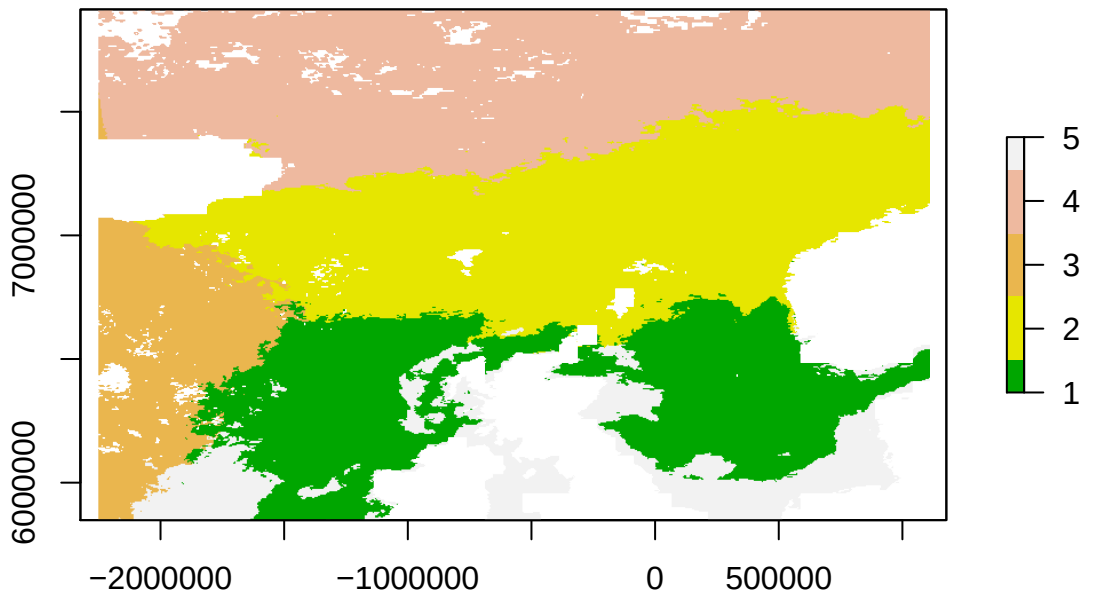
```r
#ecological provinces
prov.vec<- matrix(prov)
prov.scale.3 <- rescale(prov.vec, c(0.25,1))
prov.scale.ras.3 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), va

##create stack to synchronise NA values
#create stack
s <- stack(c(sst.scale.ras.3, chl.scale.ras.3, bath.scale.ras.3, prov.scale.ras.3, X.scale.ras.3, Y.scal

# Synchronize the NA values
stack.3 <- mask(s, calc(s,fun = sum))
names(stack.3) <- c("SST", "Chl.log", "Depth.log", "Province", "X.log", "Y.log")



##Kmeans classification
#data.frame from raster stack
Kmat.3 <- as.data.frame(stack.3)

#cluster output, no NAs
cluster.out <- kmeans(na.omit(Kmat.3), 5)

#vector to hold classification results
Kmat.df.factor <- rep(NA, length(Kmat.3[,1]))

#put results into vector
Kmat.df.factor[!is.na(Kmat.3[,1])] <- cluster.out$cluster

## fill empty raster with the class results
#new raster
output.3 <- raster(stack.3)
#set values to class results
output.3 <- setValues(output.3, Kmat.df.factor)

##visualize it
plot(output.1, col = terrain.colors(5))
```



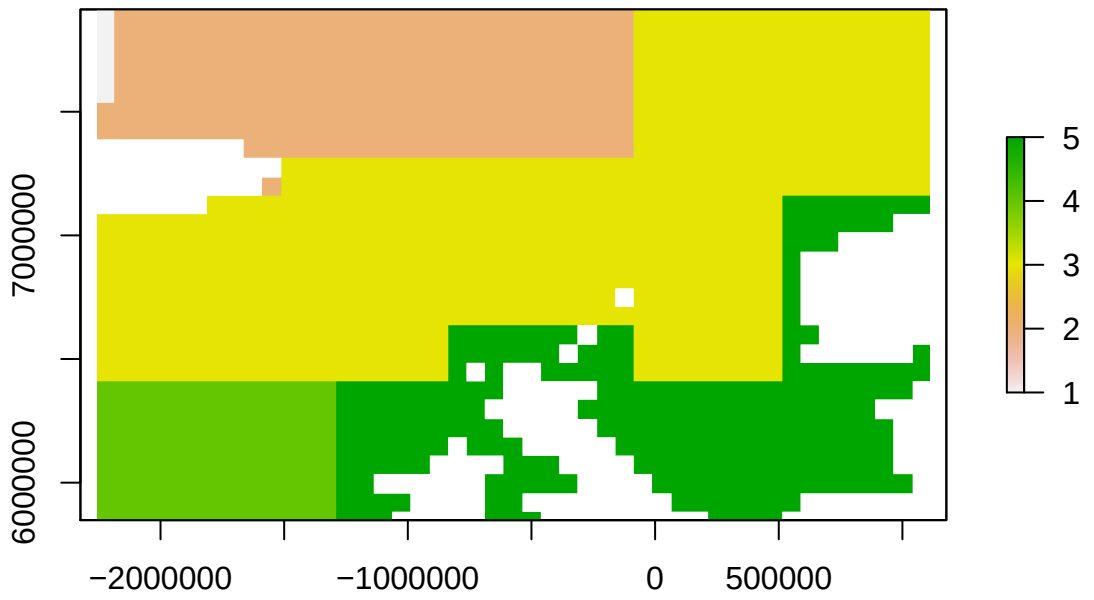

```
plot(output.2, col = terrain.colors(5))
```



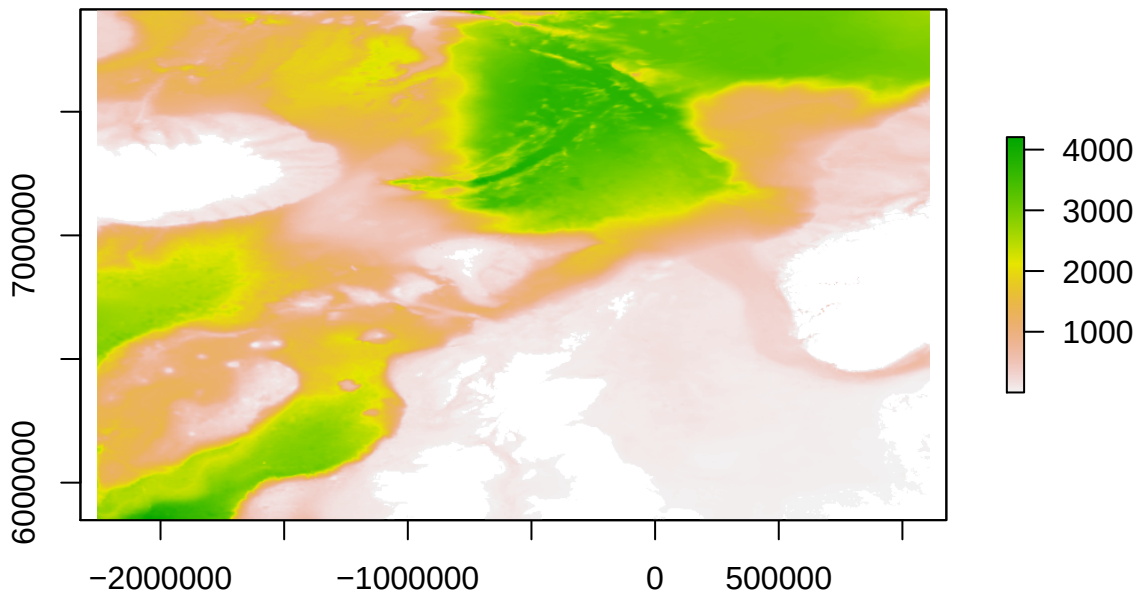

```
plot(output.3, col = terrain.colors(5))
```



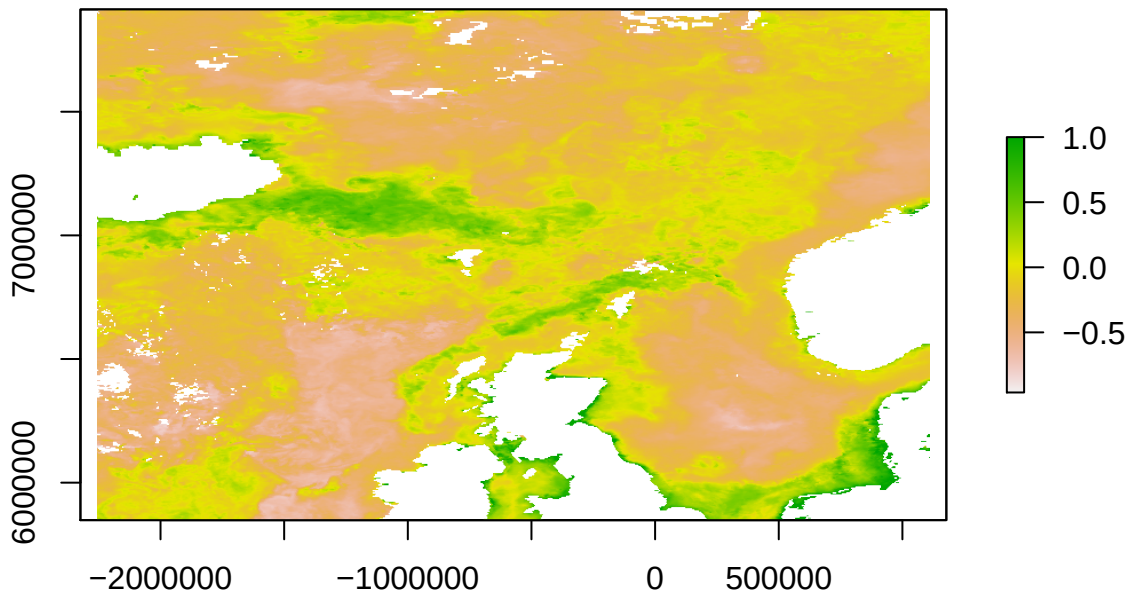

```
plot(prov)
```



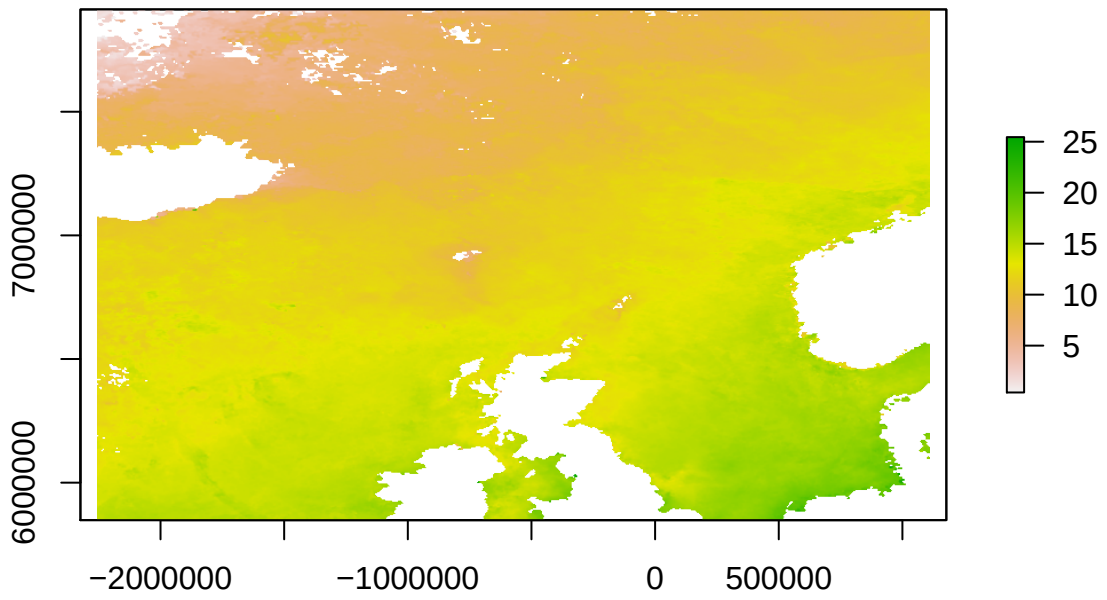

```
plot(bath)
```


7000000
6000000
−2000000
−1000000
0
500000
4000
3000
2000
1000

```
plot(chl_log)
```



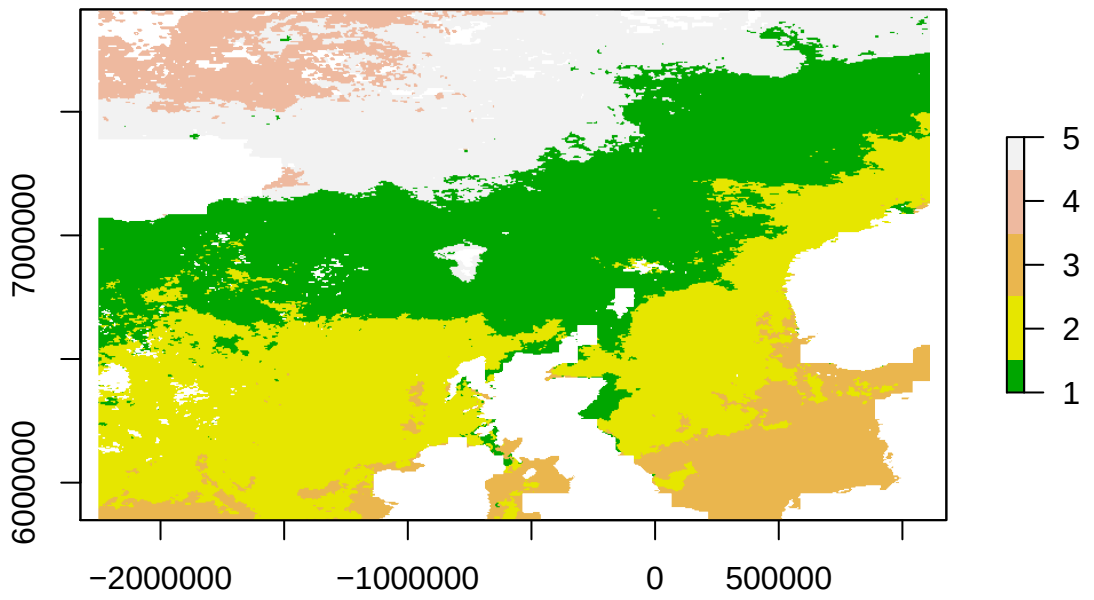

```
plot(SST)
```



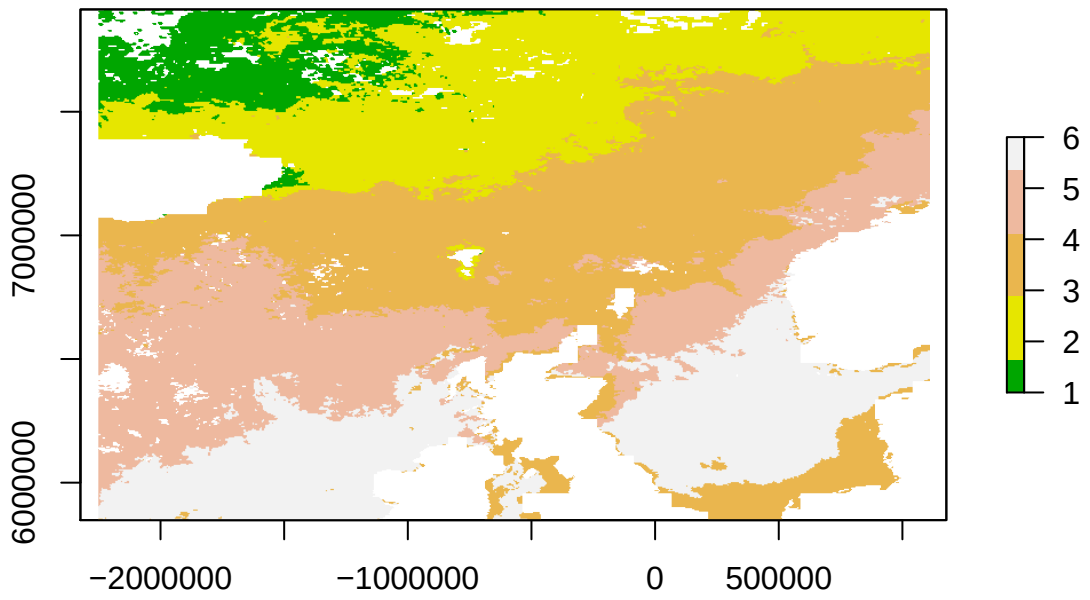
Best alternative scaling identified

```
##rescale values 4th try
#chlorophyll a
chl.vec <- matrix(chl_log)
chl.scale.4 <- rescale(chl.vec, c(0,4))
chl.scale.ras.4 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), val

#sea surface temperature
sst.vec<- matrix(SST)
sst.scale.4 <- rescale(sst.vec, c(0,6))
sst.scale.ras.4 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), val

#bathymetry
bath.vec <- matrix(bath_log)
bath.scale.4 <- rescale(bath.vec, c(0,2))
bath.scale.ras.4 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), va

#lats/longs
Xvec <- matrix(Xras)
X.scale.4 <- rescale(Xvec, c(0,2))
X.scale.ras.4 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), vals=

Yvec <- matrix(Yras)
Y.scale.4 <- rescale(Yvec, c(0,2))
Y.scale.ras.4 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), vals=
```

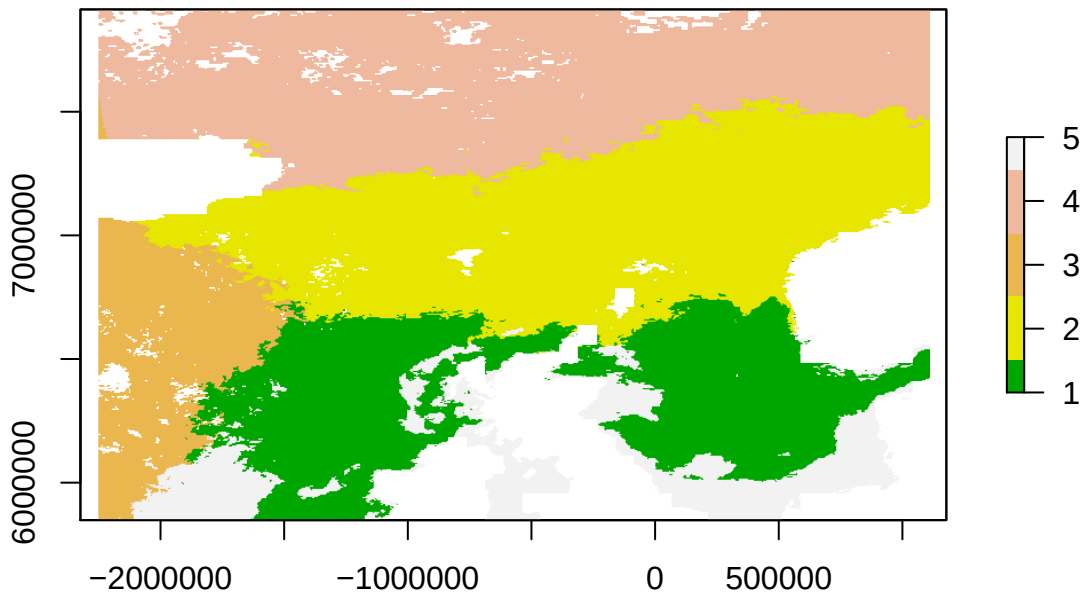
```r
#ecological provinces
prov.vec<- matrix(prov)
prov.scale.4 <- rescale(prov.vec, c(0.25,1.25))
prov.scale.ras.4 <- raster(extent(chl_log), nrow=nrow(chl_log), ncol=ncol(chl_log), crs=crs(chl_log), va

##create stack to synchronise NA values
#create stack
s <- stack(c(sst.scale.ras.4, chl.scale.ras.4, bath.scale.ras.4, prov.scale.ras.4, X.scale.ras.4, Y.scal

# Synchronize the NA values
stack.4 <- mask(s, calc(s,fun = sum))
names(stack.4) <- c("SST", "Chl.log", "Depth.log", "Province", "X.log", "Y.log")



##Kmeans classification
#data.frame from raster stack
Kmat.4 <- as.data.frame(stack.4)

#cluster output, no NAs
cluster.out <- kmeans(na.omit(Kmat.4), 5)

#vector to hold classification results
Kmat.df.factor <- rep(NA, length(Kmat.4[,1]))

#put results into vector
Kmat.df.factor[!is.na(Kmat.4[,1])] <- cluster.out$cluster

## fill empty raster with the class results
#new raster
output.4 <- raster(stack.4)
#set values to class results
output.4 <- setValues(output.4, Kmat.df.factor)

##visualize it
plot(output.1, col = terrain.colors(5))
```



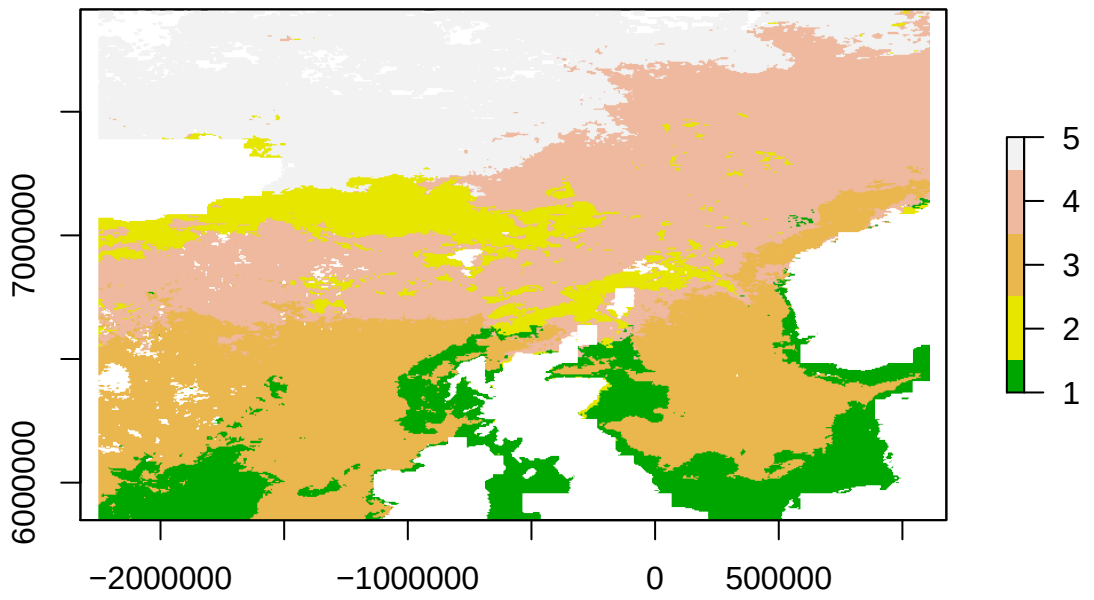

```
plot(output.2, col = terrain.colors(5))
```



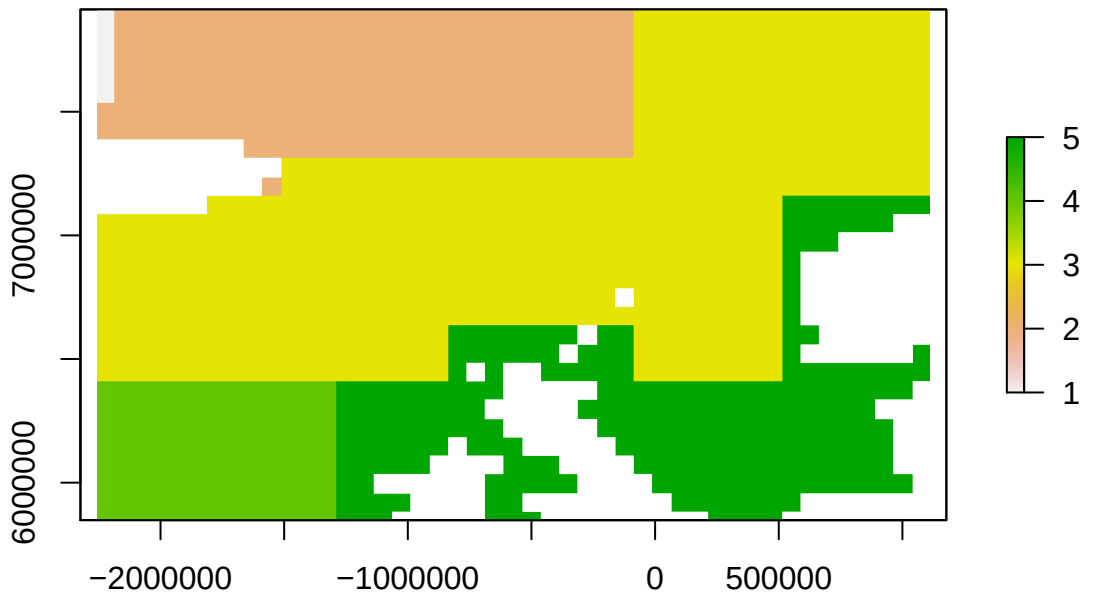

```
plot(output.3, col = terrain.colors(5))
```



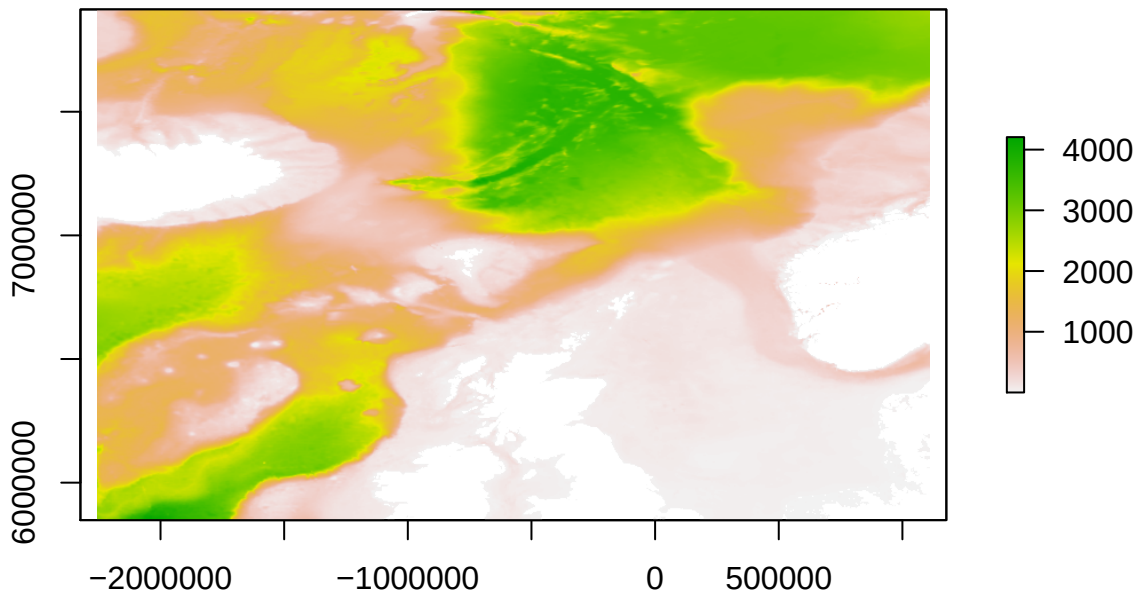

```
plot(output.4, col = terrain.colors(5))
```



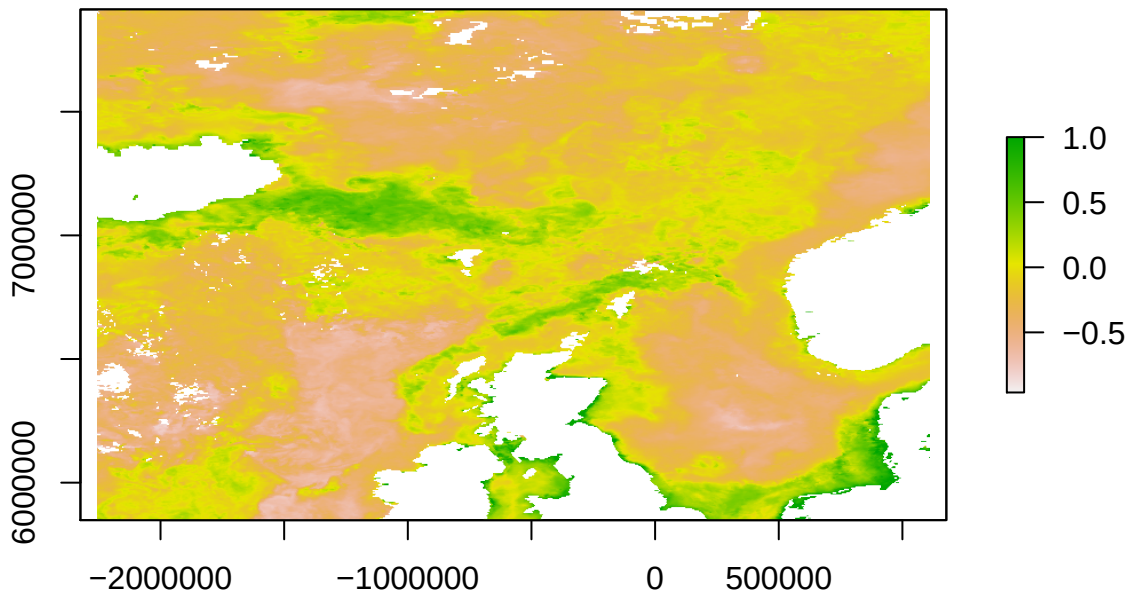

```
plot(prov)
```



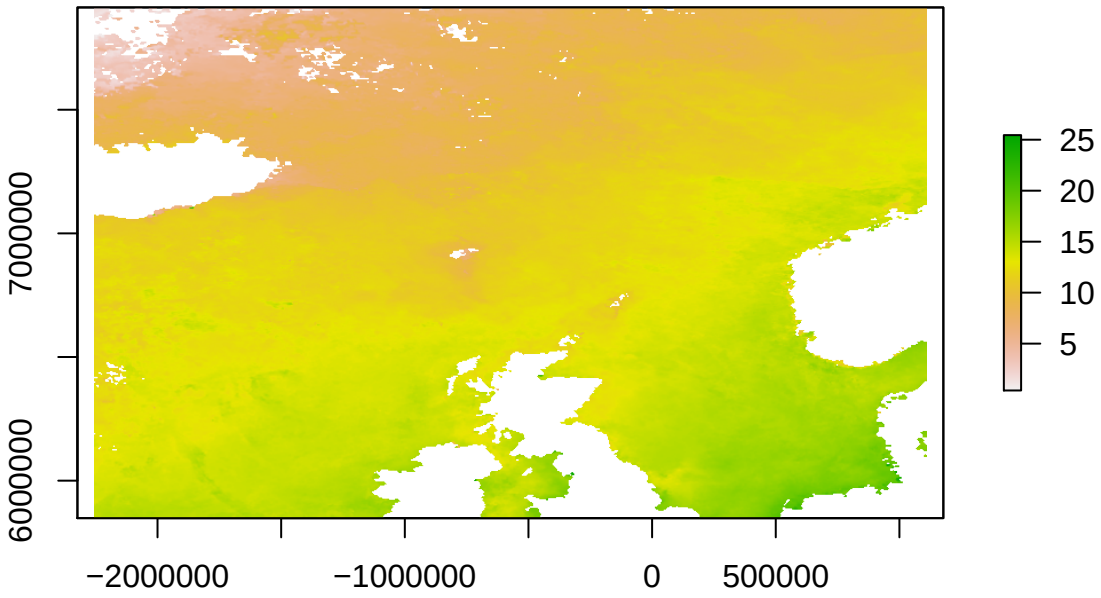

```
plot(bath)
```


7000000
6000000
−2000000
−1000000
0
500000
4000
3000
2000
1000

```
plot(chl_log)
```




```
plot(SST)
```


7000000
6000000
−2000000
−1000000
0
500000
25
20
15
10
5