```
    library(sp)
    library(ggplot2)
    library(dplyr)
```

```
##
## Attaching package:  'dplyr'
##
## The following object is masked from 'package:stats':
##
##    filter
##
## The following objects are masked from 'package:base':
##
##    intersect, setdiff, setequal, union
```

```
    devtools::load_all("~/Wollongong/pkgs/FRK",
                       export_all = FALSE)
```

```
## Loading FRK
```

```
    opts_FRK$set("progress",FALSE)
    set.seed(1)

    ## Get data
    data(meuse)
    meuse$fs <- 1
    coordinates(meuse) = ~x+y # change into an sp object

    ## Set up BAUs
    data(meuse.grid)
    gridded(meuse.grid) = ~x + y
    HexPts <- spsample(meuse.grid,
                       type = "hexagonal",
                       cellsize = 50)
    HexPols <- HexPoints2SpatialPolygons(HexPts)
    HexPols_df <- SpatialPolygonsDataFrame(HexPols,
    cbind(over(HexPols,meuse.grid),
    coordinates(HexPts)))
    HexPols_df$fs <- 1
    #HexPols_df <- subset(HexPols_df,!is.na(dist))

    # Generate observations with large spatial support
    HexPts2 <- spsample(meuse.grid,
                        type = "hexagonal",
                        cellsize = 100)
    HexPols2 <- HexPoints2SpatialPolygons(HexPts2)
    HexPols_df2 <- SpatialPolygonsDataFrame(HexPols2,
```

```
    over(HexPols2,meuse) %>%
    select(zinc)) %>%
    subset(!is.na(zinc))

    ## Generate basis functions
    G <- auto_basis(m = plane(),data=meuse,nres = 2,
                    prune=10,type = "Gaussian")
```

```
## Loading required package:  splancs
##
## Spatial Point Pattern Analysis Code in S-Plus
##
##  Version 2 - Spatial and Space-Time analysis
```

```
## [1] "Number of basis at resolution 1 = 6"
## [1] "Number of basis at resolution 2 = 27"
```

```
    ## Setup SRE model
    f <- log(zinc) ~ 1
    S <- SRE(f,list(meuse,HexPols_df2),BAUs = HexPols_df, G)
```

```
## Warning in map_data_to_BAUs(data[[i]], BAUs, av_var = av_var, variogram.formula
= f):  Not accounting for multiple data in the same grid box during
variogram estimation.  Need to see how to do this with gstat
```

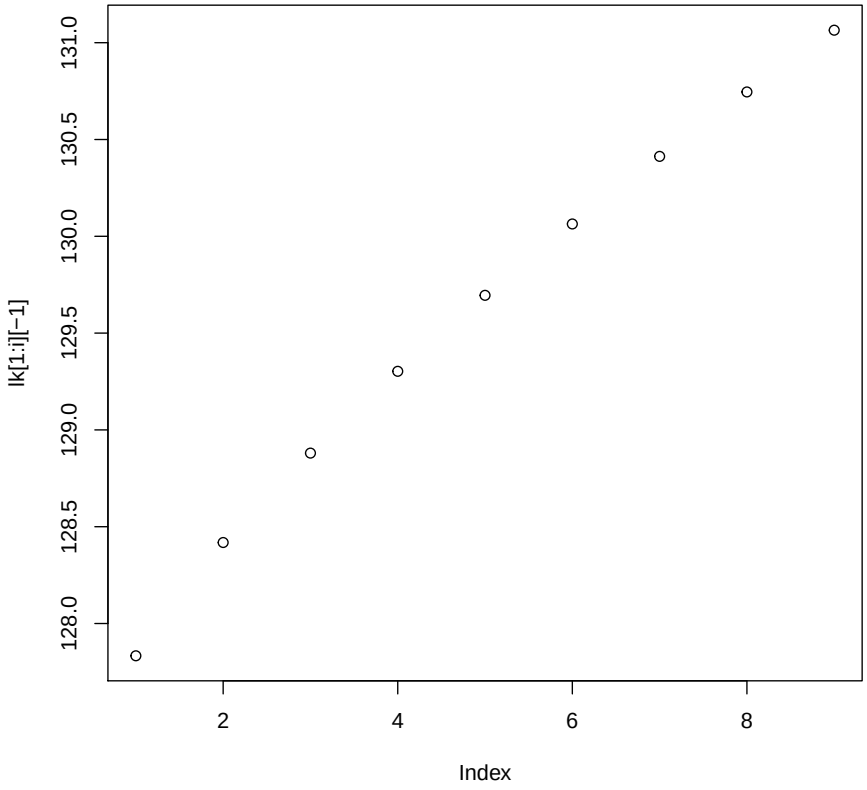
```
## [1] "sigma2e estimate = 0.0152413306239711"
```

```
## Warning in map_data_to_BAUs(data[[i]], BAUs, av_var = av_var, variogram.formula
= f):  Not accounting for multiple data in the same grid box during
variogram estimation.  Need to see how to do this with gstat
```

```
## [1] "sigma2e estimate = 0.00784995366538696"
## [1] "Averaging over polygons"
```

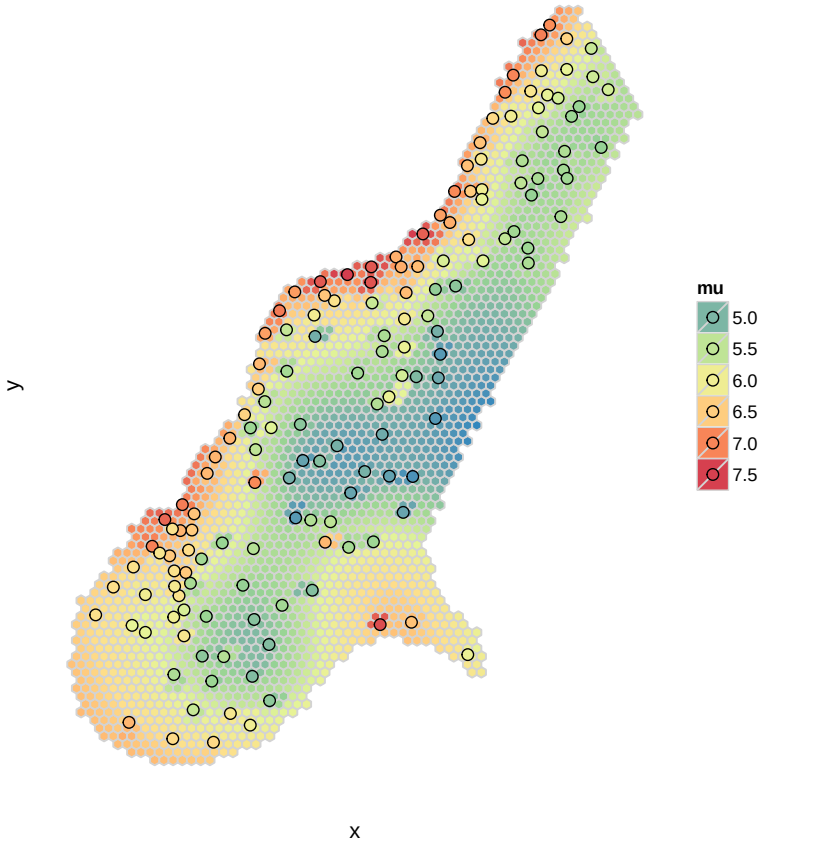
```
    S <- SRE.fit(S,n_EM = 10,print_lik=T)
```

```
## [1] "Maximum EM iterations reached"
```

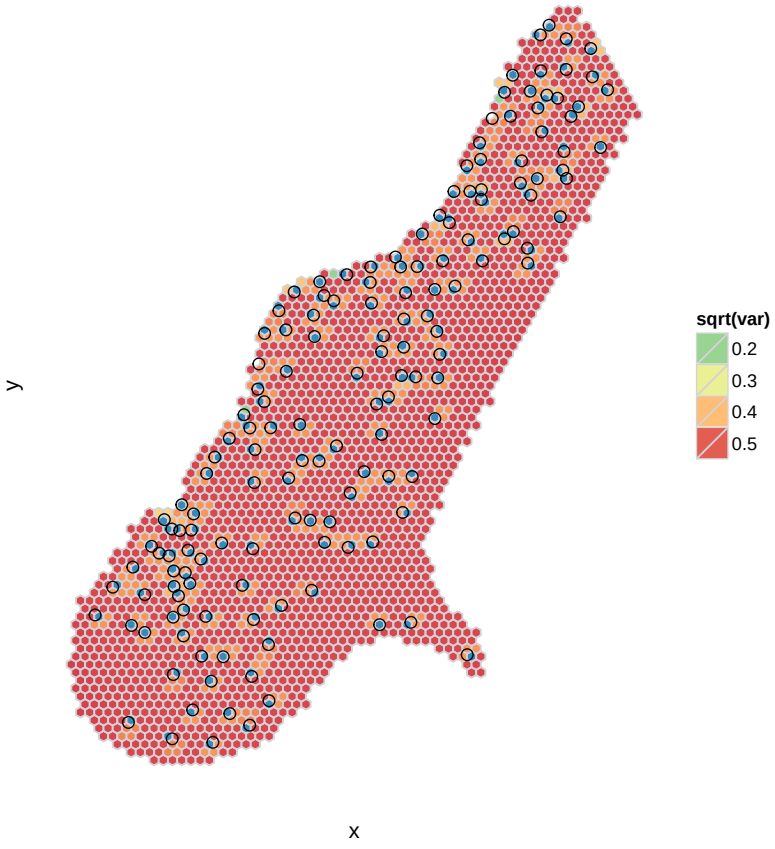



```
    ## Point predict
    HexPols_df <- SRE.predict(S,use_centroid = T)

    X <- SpatialPolygons_to_df(sp_polys = HexPols_df,
                               vars = c("mu","var"))

## Joining by:  "id"

    g1 <- EmptyTheme() +
        geom_polygon(data=X,aes(x,y,fill=mu,group=id),
                     colour="light grey") +
        scale_fill_distiller(palette="Spectral",trans="reverse") +
        geom_point(data=data.frame(meuse),
                   aes(x,y,fill=log(zinc)),
                   colour="black",
                   pch=21, size=3) +
```

```
    coord_fixed()

g2 <- EmptyTheme() +
    geom_polygon(data=X,aes(x,y,fill=sqrt(var),group=id),
                 colour="light grey") +
    scale_fill_distiller(palette="Spectral",trans="reverse") +
    geom_point(data=data.frame(meuse),
               aes(x,y),colour="black",pch=21, size=3) +
    coord_fixed()

print(g1)
```

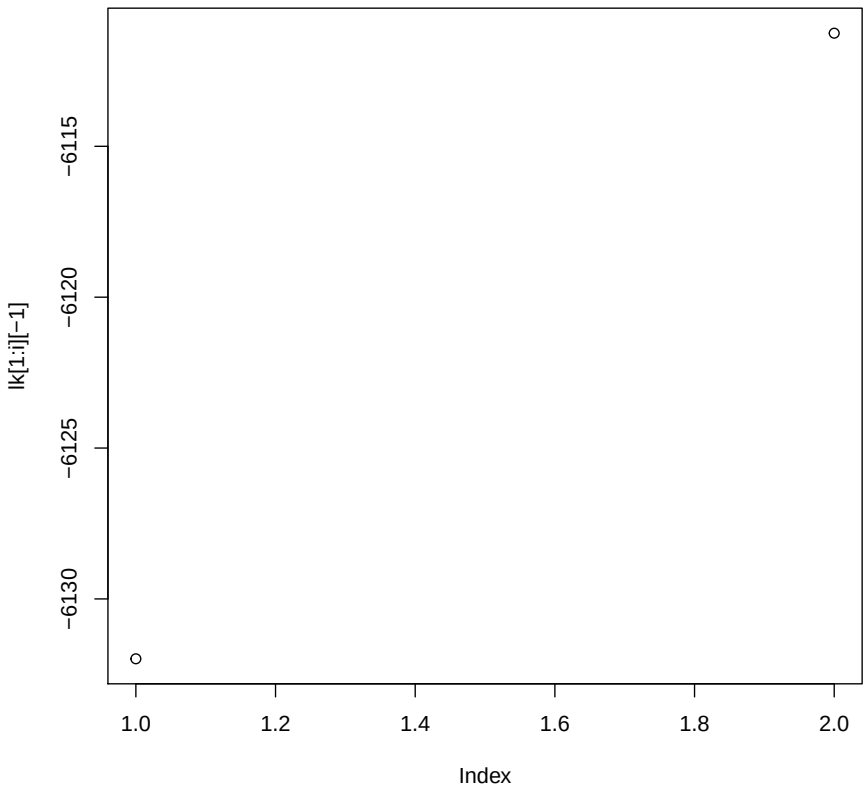


```
    print(g2)
```



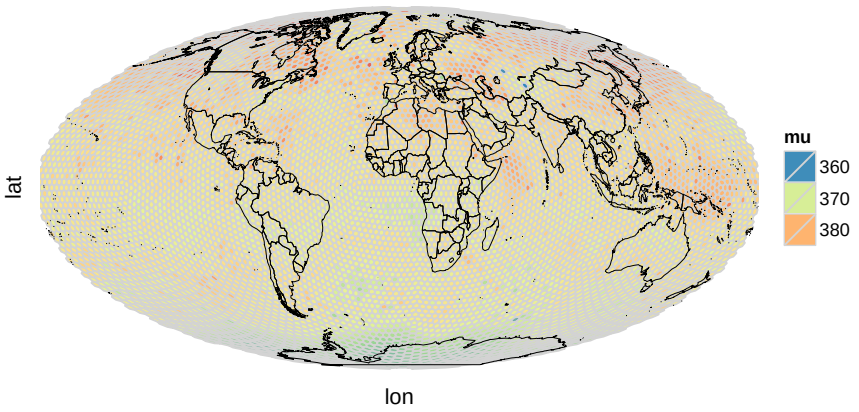

```
## Load data
load(system.file("extdata","AIRS_05_2003.rda", package = "FRK"))
AIRS_05_2003 <- filter(AIRS_05_2003,day %in% 1) %>%
select(lon,lat,co2avgret)
coordinates(AIRS_05_2003) = ~lon+lat # change into an sp object
proj4string(AIRS_05_2003)=CRS("+proj=longlat")

## Prediction (BAU) grid
load(system.file("extdata","isea3h.rda", package = "FRK"))
isea3h_res <- filter(isea3h,res == 6) %>%
arrange(id) %>%
group_by(id) %>%
filter(diff(range(lon)) < 90) %>%
data.frame() %>%
```

```
mutate(fs=1)

isea3h_sp_pol <- df_to_SpatialPolygons(
    df=filter(isea3h_res,centroid==0),
    keys=c("id"),
    coords=c("lon","lat"),
    proj=CRS("+proj=longlat"))

isea3h_sp_poldf <- SpatialPolygonsDataFrame(
    isea3h_sp_pol,
    cbind(data.frame(row.names=names(isea3h_sp_pol)),
          (filter(isea3h_res,centroid==1) %>%
               select(id,lon,lat,fs))))

## Set up SRE model
G <- auto_basis(m = sphere(),data=AIRS_05_2003,
                nres = 3,prune=15,type = "bisquare")

## [1] "Number of basis at resolution 1 = 32"
## [1] "Number of basis at resolution 2 = 90"
## [1] "Number of basis at resolution 3 = 258"

f <- co2avgret ~ lat + 1
S <- SRE(f,list(AIRS_05_2003),G,isea3h_sp_poldf) %>%
SRE.fit(n_EM = 3,print_lik=T)

## Warning in map_data_to_BAUs(data[[i]], BAUs, av_var = av_var, variogram.formula
= f):  Not accounting for multiple data in the same grid box during
variogram estimation.  Need to see how to do this with gstat

## [1] "sigma2e estimate = 3.52279378129999"
## [1] "Maximum EM iterations reached"
```

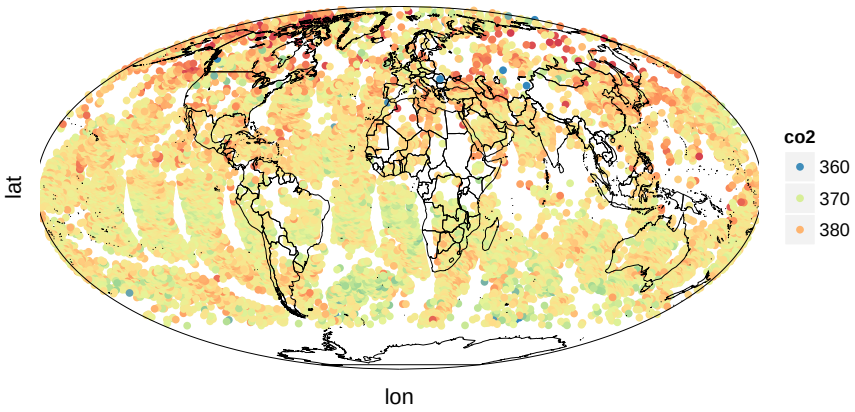



```
isea3h_sp_poldf <- SRE.predict(S,use_centroid = T)

X <- SpatialPolygons_to_df(sp_polys = isea3h_sp_poldf,
                           vars = c("mu","var"))

## Joining by:  "id"

g1 <- (EmptyTheme() +
           geom_polygon(data=X,
                        aes(lon,lat,fill=mu,group=id),
                        colour="light grey") +
           scale_fill_distiller(palette="Spectral",trans="reverse") +
           coord_map("mollweide")) %>%
draw_world(inc_border=FALSE)

mumin <- min(X$mu)
```

```
mumax <- max(X$mu)

g2 <- (EmptyTheme() +
           geom_point(data=data.frame(AIRS_05_2003),
                      aes(lon,lat,
                          colour=pmin(pmax(
                              co2avgret,mumin),
                                      mumax)),
                      size=2) +
           scale_colour_distiller(palette="Spectral",
                                  trans="reverse",
                                  guide_legend(title="co2")
                                  ) +
               coord_map("mollweide")) %>%
draw_world(inc_border=TRUE)

print(g1)
```




```
print(g2)
```


lat
lon
co2
360
370
380