spacetime: Spatio-Temporal Data in R

```
#install.packages(c("diveMove", "trip", "adehabitatLT", "plm", "cshapes"))

###################################################
options(prompt = "R> ", continue = "+  ", width = 70, useFancyQuotes = FALSE)
set.seed(1331)


###################################################
library("foreign")
read.dbf(system.file("shapes/sids.dbf", package="maptools"))[1:5,c(5,9:14)]
```

```
##          NAME BIR74 SID74 NWBIR74 BIR79 SID79 NWBIR79
## 1        Ashe  1091     1      10  1364     0      19
## 2   Alleghany   487     0      10   542     3      12
## 3       Surry  3188     5     208  3616     6     260
## 4   Currituck   508     1     123   830     2     145
## 5 Northampton  1421     9    1066  1606     3    1197
```

```
###################################################
data("wind", package = "gstat")
wind[1:6,1:12]
```

```
##   year month day   RPT   VAL   ROS   KIL   SHA  BIR   DUB   CLA   MUL
## 1   61     1   1 15.04 14.96 13.17  9.29 13.96 9.87 13.67 10.25 10.83
## 2   61     1   2 14.71 16.88 10.83  6.50 12.62 7.67 11.50 10.04  9.79
## 3   61     1   3 18.50 16.88 12.33 10.13 11.17 6.17 11.25  8.04  8.50
## 4   61     1   4 10.58  6.63 11.75  4.58  4.54 2.88  8.63  1.79  5.83
## 5   61     1   5 13.33 13.25 11.42  6.17 10.71 8.21 11.92  6.54 10.92
## 6   61     1   6 13.21  8.12  9.96  6.67  5.37 4.50 10.67  4.42  7.17
```

```
###################################################
library("plm")
```

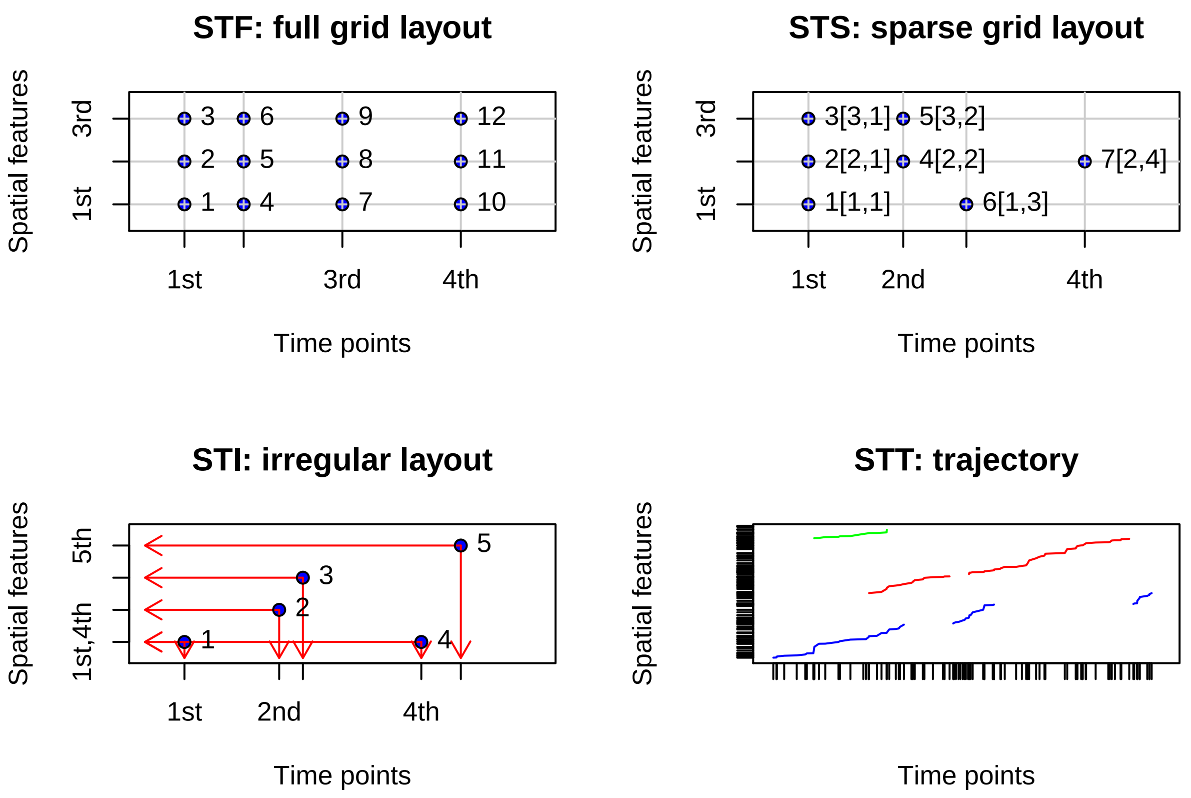
```
## Loading required package: Formula
```

```
data("Produc", package = "plm")
Produc[1:5,1:9]
```

```
##     state year region     pcap     hwy   water    util       pc   gsp
## 1 ALABAMA 1970      6 15032.67 7325.80 1655.68 6051.20 35793.80 28418
## 2 ALABAMA 1971      6 15501.94 7525.94 1721.02 6254.98 37299.91 29375
## 3 ALABAMA 1972      6 15972.41 7765.42 1764.75 6442.23 38670.30 31303
## 4 ALABAMA 1973      6 16406.26 7907.66 1742.41 6756.19 40084.01 33430
## 5 ALABAMA 1974      6 16762.67 8025.52 1734.85 7002.29 42057.31 33749
```

```
###################################################
opar = par()
par(mfrow=c(2,2))
# 1:
s = 1:3
t = c(1, 1.75, 3, 4.5)
g = data.frame(rep(t, each=3), rep(s,4))
col = 'blue'
```

```r
pch = 16
plot(g, xaxt = 'n', yaxt = 'n', xlab = "Time points",
    ylab = "Spatial features", xlim = c(.5,5.5), ylim = c(.5,3.5),
  pch = pch, col = col)
abline(h=s, col = grey(.8))
abline(v=t, col = grey(.8))
points(g)
axis(1, at = t, labels = c("1st", "2nd", "3rd", "4th"))
axis(2, at = s, labels = c("1st", "2nd", "3rd"))
text(g, labels = 1:12, pos=4)
title("STF: full grid layout")
# 2:
s = 1:3
t = c(1, 2.2, 3, 4.5)
g = data.frame(rep(t, each=3), rep(s,4))
sel = c(1,2,3,5,6,7,11)
plot(g[sel,], xaxt = 'n', yaxt = 'n', xlab = "Time points",
    ylab = "Spatial features", xlim = c(.5,5.5), ylim = c(.5,3.5),
  pch = pch, col = col)
abline(h=s, col = grey(.8))
abline(v=t, col = grey(.8))
points(g[sel,])
axis(1, at = t, labels = c("1st", "2nd", "3rd", "4th"))
axis(2, at = s, labels = c("1st", "2nd", "3rd"))
text(g[sel,], labels = paste(1:length(sel), "[",c(1,2,3,2,3,1,2),",",c(1,1,1,2, 2,3,4),"]", sep=""), pos
title("STS: sparse grid layout")
# 3:
s = c(1,2,3,1,4)
t = c(1, 2.2, 2.5, 4, 4.5)
g = data.frame(t,s)
plot(g, xaxt = 'n', yaxt = 'n', xlab = "Time points",
    ylab = "Spatial features", xlim = c(.5,5.5), ylim = c(.5,4.5),
  pch = pch, col = col)
#abline(h=s, col = grey(.8))
#abline(v=t, col = grey(.8))
arrows(t,s,0.5,s,.1,col='red')
arrows(t,s,t,0.5,.1,col='red')
points(g)
axis(1, at = sort(unique(t)), labels = c("1st", "2nd", "3rd", "4th", "5th"))
axis(2, at = sort(unique(s)), labels = c("1st,4th", "2nd", "3rd", "5th"))
text(g, labels = 1:5, pos=4)
title("STI: irregular layout")
# 4: traj
ns = 400
nt = 100
s = sort(runif(ns))
t = sort(runif(nt))
g = data.frame(t[1:30],s[1:30])
plot(g, xaxt = 'n', yaxt = 'n', xlab = "Time points",
    ylab = "Spatial features",
  type='l', col = 'blue', xlim = c(0,1), ylim = c(0,s[136]))
lines(data.frame(t[41:60],s[31:50]), col = 'blue')
lines(data.frame(t[91:100],s[51:60]), col = 'blue')
lines(data.frame(t[21:40],s[61:80]), col = 'red')
lines(data.frame(t[51:90],s[81:120]), col = 'red')
lines(data.frame(t[11:25],s[121:135]), col = 'green')
#abline(h=s, col = grey(.8))
#abline(v=t, col = grey(.8))
#arrows(t,s,0.5,s,.1,col='red')
#arrows(t,s,t,0.5,.1,col='red')
axis(1, at = sort(unique(t)), labels = rep("", length(t)))
axis(2, at = sort(unique(s)), labels = rep("", length(s)))
#text(g, labels = 1:5, pos=4)
title("STT: trajectory")
```



```r
opar$cin = opar$cra = opar$csi = opar$cxy = opar$din = NULL
par(opar)
```

```
## Warning in par(opar): graphical parameter "page" cannot be set
```

```r
#####################################################
sp = cbind(x = c(0,0,1), y = c(0,1,1))
row.names(sp) = paste("point", 1:nrow(sp), sep="")
library("sp")
sp = SpatialPoints(sp)


#####################################################
time = as.POSIXct("2010-08-05", tz = "GMT")+3600*(10:13)
```

```
###################################################
m = c(10,20,30) # means for each of the 3 point locations
values = rnorm(length(sp)*length(time), mean = rep(m, 4))
IDs = paste("ID",1:length(values), sep = "_")
mydata = data.frame(values = signif(values, 3), ID=IDs)


###################################################
library("spacetime")
stfdf = STFDF(sp, time, time+60, data = mydata)


###################################################
#air_quality[2:3, 1:10, "PM10"]


###################################################
#air_quality[Germany, "2008::2009", "PM10"]


###################################################
xs1 = as(stfdf, "Spatial")
class(xs1)
```

```
## [1] "SpatialPointsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
xs1
```

```
##        coordinates X2010.08.05.10.00.00 X2010.08.05.11.00.00
## point1      (0, 0)                 9.66                 9.64
## point2      (0, 1)                21.20                19.50
## point3      (1, 1)                29.90                32.10
##        X2010.08.05.12.00.00 X2010.08.05.13.00.00
## point1                 8.98                 11.4
## point2                19.60                 19.8
## point3                30.20                 29.8
```

```
###################################################
attr(xs1, "time")
```

```
## [1] "2010-08-05 10:00:00 GMT" "2010-08-05 11:00:00 GMT"
## [3] "2010-08-05 12:00:00 GMT" "2010-08-05 13:00:00 GMT"
```

```
###################################################
x = as(stfdf, "STIDF")
xs2 = as(x, "Spatial")
class(xs2)
```

```
## [1] "SpatialPointsDataFrame"
## attr(,"package")
## [1] "sp"
```

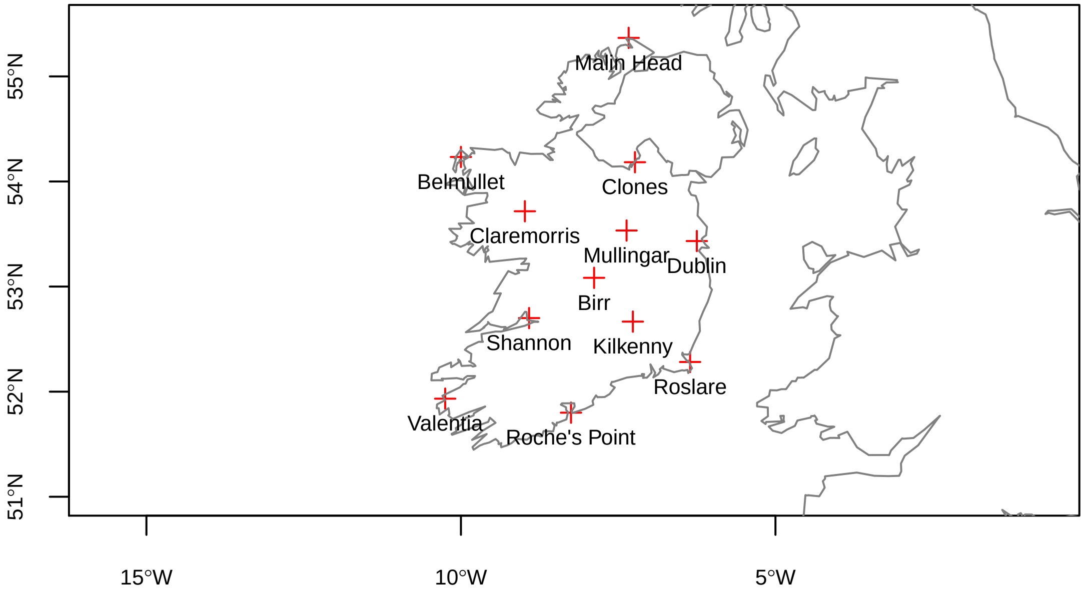
```
xs2[1:4,]
```

```
##   coordinates values   ID                time
## 1      (0, 0)   9.66 ID_1 2010-08-05 10:00:00
## 2      (0, 1)  21.20 ID_2 2010-08-05 10:00:00
## 3      (1, 1)  29.90 ID_3 2010-08-05 10:00:00
## 4      (0, 0)   9.64 ID_4 2010-08-05 11:00:00
```

```
####################################################
library("gstat")
data("wind")
wind.loc$y = as.numeric(char2dms(as.character(wind.loc[["Latitude"]])))
wind.loc$x = as.numeric(char2dms(as.character(wind.loc[["Longitude"]])))
coordinates(wind.loc) = ~x+y
proj4string(wind.loc) = "+proj=longlat +datum=WGS84"


####################################################
library("mapdata")
```

```
## Loading required package: maps
```

```
plot(wind.loc, xlim = c(-11,-5.4), ylim = c(51,55.5), axes=T, col="red",
     cex.axis =.7)
map("worldHires", add=TRUE, col = grey(.5))
text(coordinates(wind.loc), pos=1, label=wind.loc$Station, cex=.7)
```



```
####################################################
wind[1:3,]
```

```
##     year month day    RPT    VAL    ROS    KIL    SHA   BIR    DUB    CLA    MUL
## 1   61     1   1 15.04 14.96 13.17  9.29 13.96 9.87 13.67 10.25 10.83
## 2   61     1   2 14.71 16.88 10.83  6.50 12.62 7.67 11.50 10.04  9.79
## 3   61     1   3 18.50 16.88 12.33 10.13 11.17 6.17 11.25  8.04  8.50
##     CLO   BEL   MAL
## 1 12.58 18.50 15.04
## 2  9.67 17.54 13.83
## 3  7.67 12.75 12.71
```

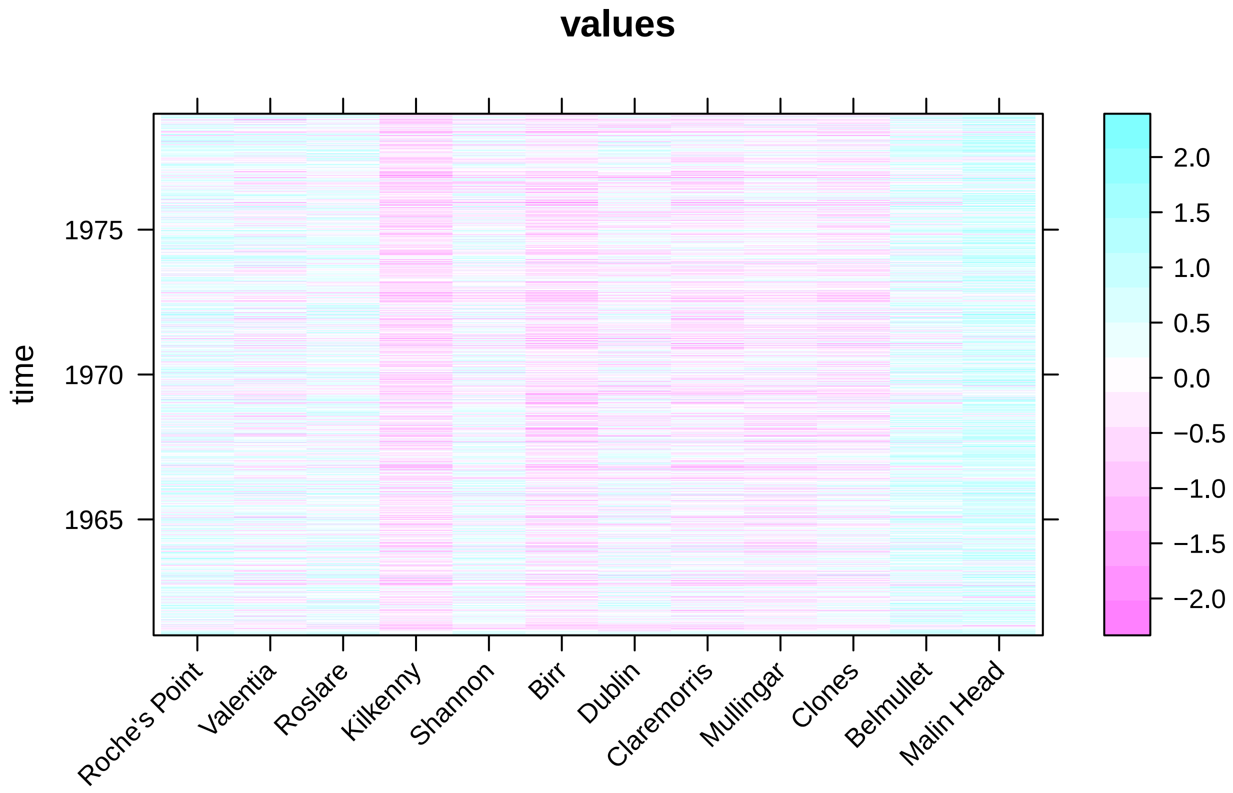
```
###################################################
wind$time = ISOdate(wind$year+1900, wind$month, wind$day)
wind$jday = as.numeric(format(wind$time, '%j'))


###################################################
stations = 4:15
windsqrt = sqrt(0.5148 * as.matrix(wind[stations])) # knots -> m/s
Jday = 1:366
windsqrt = windsqrt - mean(windsqrt)
daymeans = sapply(split(windsqrt, wind$jday), mean)
meanwind = lowess(daymeans ~ Jday, f = 0.1)$y[wind$jday]
velocities = apply(windsqrt, 2, function(x) { x - meanwind })


###################################################
wind.loc = wind.loc[match(names(wind[4:15]), wind.loc$Code),]
pts = coordinates(wind.loc[match(names(wind[4:15]), wind.loc$Code),])
rownames(pts) = wind.loc$Station
pts = SpatialPoints(pts, CRS("+proj=longlat +datum=WGS84"))


###################################################
library("rgdal")
```

```
## rgdal: version: 1.2-16, (SVN revision 701)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.1.2, released 2016/10/24
##  Path to GDAL shared files: /usr/share/gdal/2.1
##  GDAL binary built with GEOS: TRUE
##  Loaded PROJ.4 runtime: Rel. 4.9.3, 15 August 2016, [PJ_VERSION: 493]
##  Path to PROJ.4 shared files: (autodetected)
##  Linking to sp version: 1.2-5
```

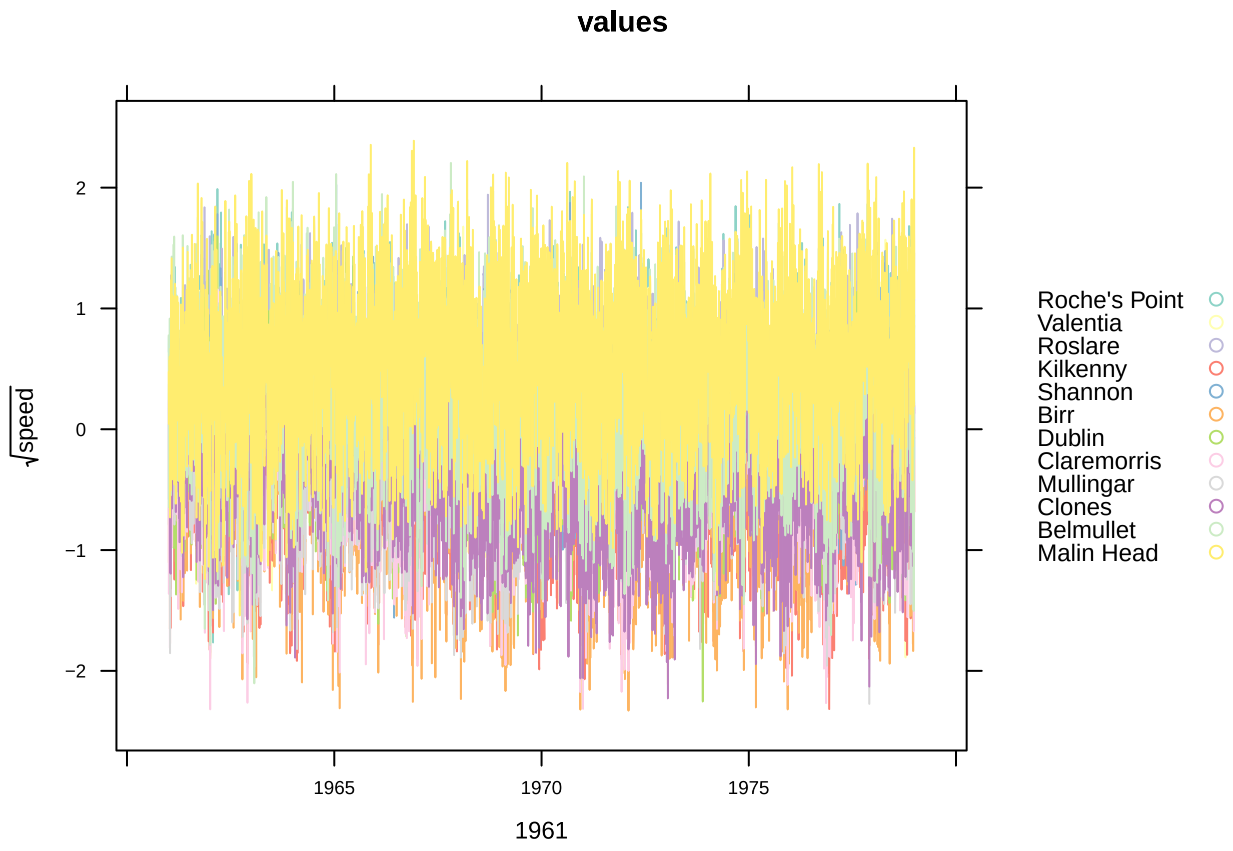
```
utm29 = CRS("+proj=utm +zone=29 +datum=WGS84")
pts = spTransform(pts, utm29)


###################################################
wind.data = stConstruct(velocities, space = list(values = 1:ncol(velocities)),
                        time = wind$time, SpatialObj = pts, interval = TRUE)
class(wind.data)
```

```
## [1] "STFDF"
## attr(,"package")
## [1] "spacetime"
```

```
####################################################
scales=list(x=list(rot = 45))
stplot(wind.data, mode = "xt", scales = scales, xlab = NULL)
```



```
####################################################
# code to create figure 5.
library("lattice")
library("RColorBrewer")
b = brewer.pal(12, "Set3")
par.settings = list(superpose.symbol = list(col = b, fill = b),
  superpose.line = list(col = b),
  fontsize = list(text=9))
stplot(wind.data, mode = "ts",  auto.key=list(space="right"),
  xlab = "1961", ylab = expression(sqrt(speed)),
  par.settings = par.settings)
```



```
####################################################
library(xts)
```

```
## Loading required package: zoo
##
## Attaching package: 'zoo'
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
.parseISO8601('2010-05')
```

```
## $first.time
## [1] "2010-05-01 UTC"
##
## $last.time
## [1] "2010-05-31 23:59:59 UTC"
```

```
####################################################
.parseISO8601('2010-05-01T13:30/2010-05-01T13:39')
```

```
## $first.time
## [1] "2010-05-01 13:30:00 UTC"
##
## $last.time
## [1] "2010-05-01 13:39:59 UTC"
```

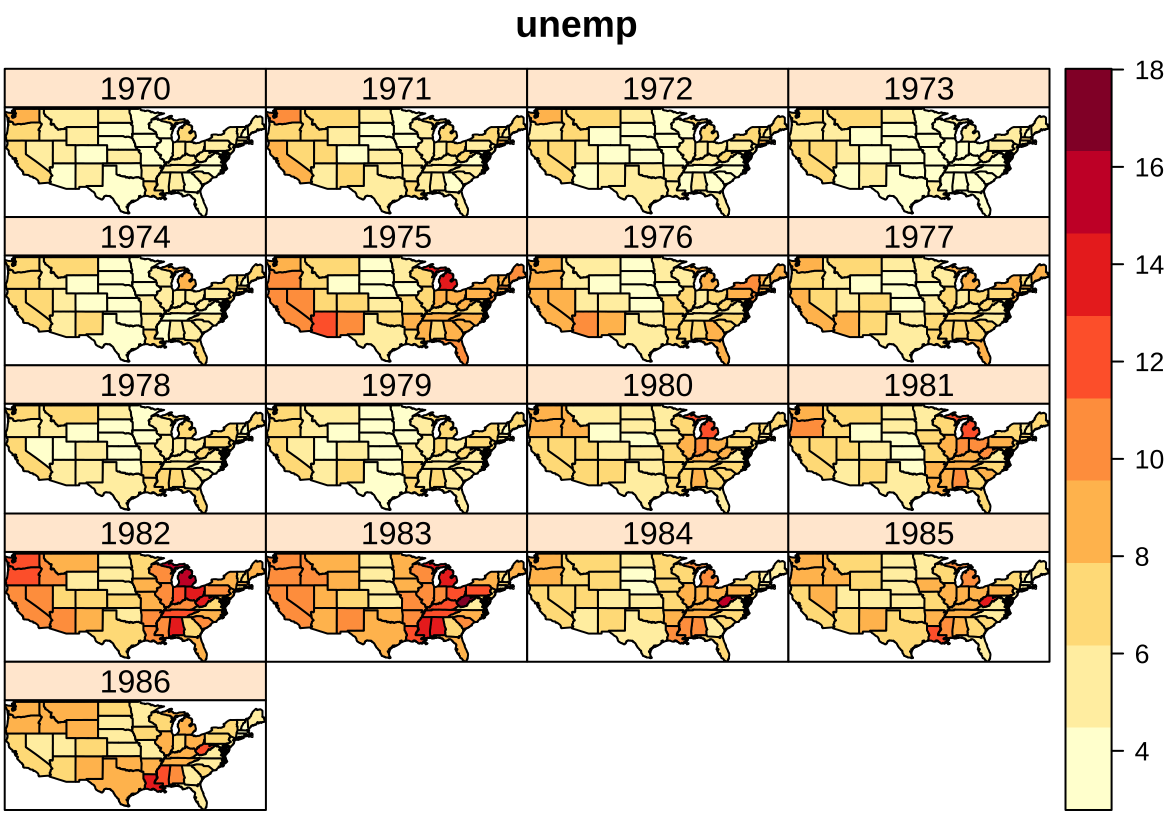
```
###################################################
library("maptools")
```

```
## Checking rgeos availability: TRUE
```

```
fname = system.file("shapes/sids.shp", package="maptools")[1]
nc = readShapePoly(fname, proj4string=CRS("+proj=longlat +datum=NAD27"))
```

```
## Warning: use rgdal::readOGR or sf::st_read
```

```
###################################################
time = as.POSIXct(strptime(c("1974-07-01", "1979-07-01"), "%Y-%m-%d"),
  tz = "GMT")
endTime = as.POSIXct(strptime(c("1978-06-30", "1984-06-30"), "%Y-%m-%d"),
  tz = "GMT")


###################################################
data = data.frame(
  BIR = c(nc$BIR74, nc$BIR79),
  NWBIR = c(nc$NWBIR74, nc$NWBIR79),
  SID = c(nc$SID74, nc$SID79))


###################################################
nct = STFDF(sp = as(nc, "SpatialPolygons"), time, data, endTime)


###################################################
library("maps")
states.m = map('state', plot=FALSE, fill=TRUE)
IDs <- sapply(strsplit(states.m$names, ":"), function(x) x[1])
library("maptools")
states = map2SpatialPolygons(states.m, IDs=IDs)


###################################################
yrs = 1970:1986
time = as.POSIXct(paste(yrs, "-01-01", sep=""), tz = "GMT")


###################################################
data("Produc")


###################################################
# deselect District of Columbia, polygon 8, which is not present in Produc:
Produc.st = STFDF(states[-8], time, Produc[order(Produc[2], Produc[1]),])


###################################################
library("RColorBrewer")
stplot(Produc.st[,,"unemp"], yrs, col.regions = brewer.pal(9, "YlOrRd"),cuts=9)
```



```
####################################################
zz <- plm(log(gsp) ~ log(pcap) + log(pc) + log(emp) + unemp,
  data = as.data.frame(Produc.st), index = c("state", "year"))
```

```
####################################################
library("maptools")
m = map2SpatialLines(
  map("worldHires", xlim = c(-11,-5.4), ylim = c(51,55.5), plot=F))
proj4string(m) = "+proj=longlat +datum=WGS84"
m = spTransform(m, utm29)
```

```
####################################################
grd = SpatialPixels(SpatialPoints(makegrid(m, n = 300)),
  proj4string = proj4string(m))
```

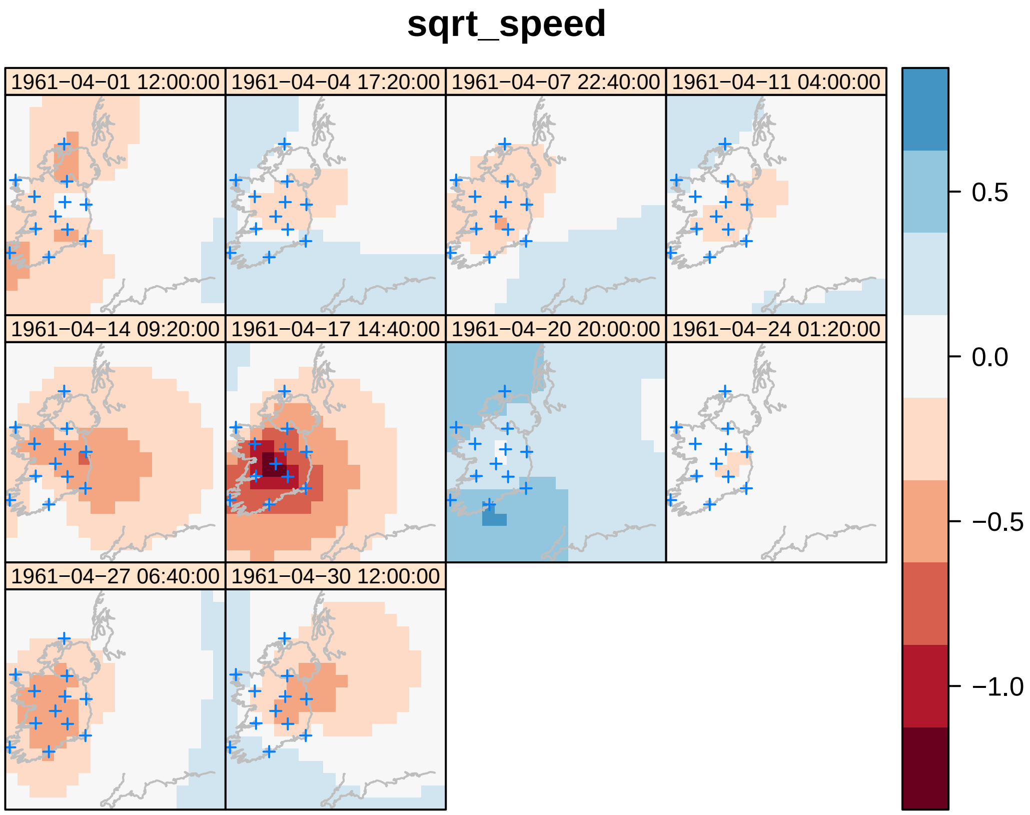
```
####################################################
wind.data = wind.data[, "1961-04"]
```

```
####################################################
```

```
n = 10
tgrd = xts(1:n, seq(min(index(wind.data)), max(index(wind.data)), length=n))
pred.grd = STF(grd, tgrd)


#####################################################
v = vgmST("separable", space = vgm(0.6, "Sph", 750000), time = vgm(1, "Sph", 1.0 * 3600 * 24), sill=0.6
wind.ST = krigeST(values ~ 1, wind.data, pred.grd, v)
```

```
## Warning in krigeST(values ~ 1, wind.data, pred.grd, v): The spatio-
## temporal variogram model does not carry a time unit attribute:
## krisgeST cannot check whether the temporal distance metrics coincide.
```

```
colnames(wind.ST@data) <- "sqrt_speed"

#####################################################
layout = list(list("sp.lines", m, col='grey'),
  list("sp.points", pts, first=F, cex=.5))
stplot(wind.ST, col.regions=brewer.pal(11, "RdBu")[-c(10,11)],
  at=seq(-1.375,1,by=.25),
  par.strip.text = list(cex=.7), sp.layout = layout)
```



```
#####################################################
pdf("wind.pdf", height=4.5)
layout = list(list("sp.lines", m, col='grey'),
  list("sp.points", pts, first=F, cex=.5))
print(stplot(wind.ST, col.regions=brewer.pal(11, "RdBu")[-c(10,11)],
  at=seq(-1.375,1,by=.25),
  par.strip.text = list(cex=.7), sp.layout = layout))
dev.off()
```

```
## pdf
##   2
```

```
####################################################
pdf("windts.pdf", height = 4)
library("lattice")
library("RColorBrewer")
b = brewer.pal(12,"Set3")
par.settings = list(superpose.symbol = list(col = b, fill = b),
  superpose.line = list(col = b),
  fontsize = list(text=9))
print(stplot(wind.data, mode = "ts",  auto.key=list(space="right"),
  xlab = "1961", ylab = expression(sqrt(speed)),
  par.settings = par.settings))
dev.off()
```

```
## pdf
##   2
```

```
####################################################
pdf("hov.pdf")
scales=list(x=list(rot=45))
stplot(wind.data, mode = "xt", scales = scales, xlab = NULL,
  col.regions=brewer.pal(11, "RdBu"),at = seq(-1.625,1.125,by=.25))
dev.off()
```

```
## pdf
##   2
```

```
####################################################
eof.data = EOF(wind.data)
```

```
## Warning: 'EOF' is deprecated.
## Use 'eof' instead.
## See help("Deprecated")
```

```
####################################################
eof.int = EOF(wind.ST)
```

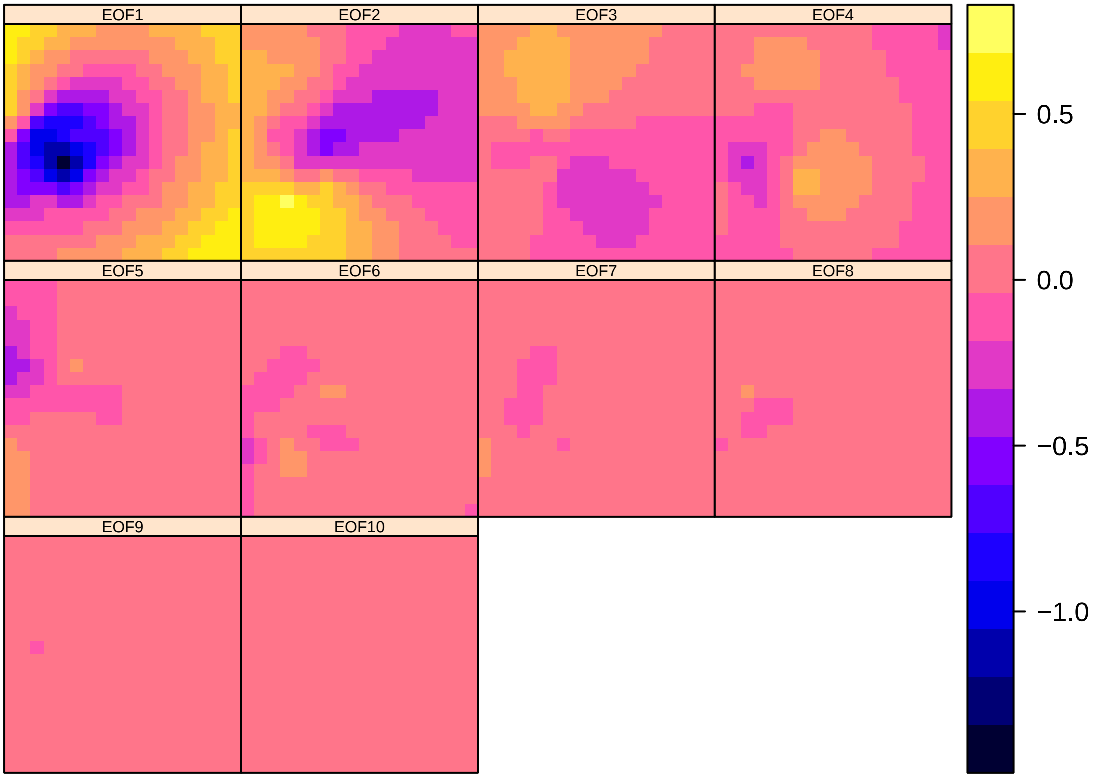
```
## Warning: 'EOF' is deprecated.
## Use 'eof' instead.
## See help("Deprecated")
```

```
####################################################
eof.xts = EOF(wind.ST, "temporal")
```

```
## Warning: 'EOF' is deprecated.
## Use 'eof' instead.
## See help("Deprecated")
```

```
####################################################
print(spplot(EOF(wind.ST), col.regions=bpy.colors(),
  par.strip.text = list(cex=.5), as.table = TRUE))
```

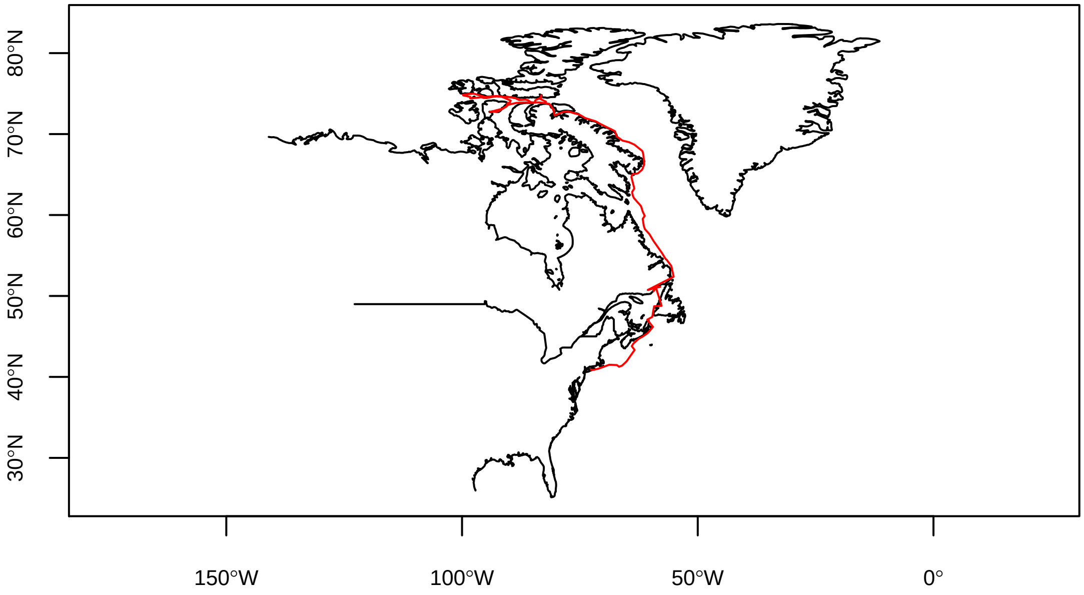
```
## Warning: 'EOF' is deprecated.
## Use 'eof' instead.
## See help("Deprecated")
```



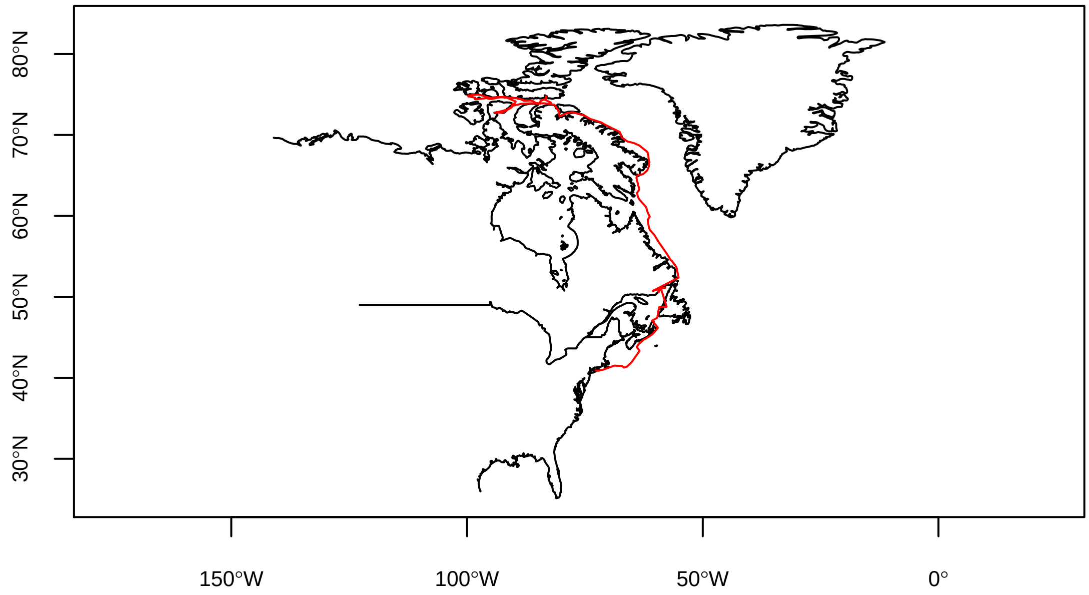
```
#####################################################

library("diveMove")
```

```
## Loading required package: stats4
## This is diveMove 1.4.3. For overview type vignette("diveMove")
```

```
library("trip")
data(sealLocs, package="diveMove")
sealLocs$time = as.POSIXct(sealLocs$time)
ringy = subset(sealLocs, id == "ringy" & !is.na(lon) & !is.na(lat))
coordinates(ringy) = ringy[c("lon", "lat")]
tr = trip(ringy, c("time", "id"))


#####################################################
setAs("trip", "STTDF",
  function(from) {
    from$burst = from[[from@TOR.columns[2]]]
    time = from[[from@TOR.columns[1]]]
    rt = range(time)
    #timeIsInterval(rt) = timeIsInterval(time) = FALSE
    # TODO: take care of endTime?
    #from = from[order(time),]
    STIbox = STI(SpatialPoints(t(bbox(from))), rt)
    STT = new("STT", STIbox, traj = list(STI(geometry(from), time)))
    new("STTDF", STT, data = from@data)
  }
)
x = as(tr, "STTDF")
m = map2SpatialLines(map("world",
  xlim = c(-100,-50), ylim = c(40,77), plot=F))
proj4string(m) = "+proj=longlat +datum=WGS84"
plot(m, axes=TRUE, cex.axis =.7)
lines(x, col = "red")
```



```
####################################################
plot(m, axes=TRUE, cex.axis =.7)
lines(x, col = "red")
```



```
##################################################
library("adehabitatLT")
```

```
## Loading required package: ade4

## Loading required package: adehabitatMA

## Loading required package: CircStats

## Loading required package: MASS

## Loading required package: boot

##
## Attaching package: 'boot'

## The following object is masked from 'package:diveMove':
##
##     logit

## The following object is masked from 'package:lattice':
##
##     melanoma

##
## Attaching package: 'adehabitatLT'

## The following object is masked from 'package:zoo':
##
##     is.regular
```

```
data("puechabonsp")
locs = puechabonsp$relocs
xy = coordinates(locs)
da = as.character(locs$Date)
da = as.POSIXct(strptime(as.character(locs$Date),"%y%m%d"), tz = "GMT")
ltr = as.ltraj(xy, da, id = locs$Name)
#foo = function(dt) dt > 100*3600*24
#l2 = cutltraj(ltr, "foo(dt)", nextr = TRUE)


###################################################
#sttdf = as(l2, "STTDF")
#stplot(sttdf, by="time*id")


###################################################
#sttdf = as(l2, "STTDF")
#print(stplot(sttdf, by="time*id"))


###################################################
library("cshapes")
```

```
## Loading required package: plyr
```

```
##
## Attaching package: 'plyr'
```

```
## The following object is masked from 'package:adehabitatLT':
##
##     id
```

```
## The following object is masked from 'package:adehabitatMA':
##
##     join
```

```
## The following object is masked from 'package:maps':
##
##     ozone
```

```
cs = cshp()
```

```
## Warning: use rgdal::readOGR or sf::st_read
```

```
names(cs)
```

```
##  [1] "CNTRY_NAME" "AREA"       "CAPNAME"    "CAPLONG"    "CAPLAT"
##  [6] "FEATUREID"  "COWCODE"    "COWSYEAR"   "COWSMONTH"  "COWSDAY"
## [11] "COWEYEAR"   "COWEMONTH"  "COWEDAY"    "GWCODE"     "GWSYEAR"
## [16] "GWSMONTH"   "GWSDAY"     "GWEYEAR"    "GWEMONTH"   "GWEDAY"
## [21] "ISONAME"    "ISO1NUM"    "ISO1AL2"    "ISO1AL3"
```

```
row.names(cs) = paste(as.character(cs$CNTRY_NAME), 1:244)


###################################################
begin = as.POSIXct(strptime(paste(cs$COWSYEAR,
  cs$COWSMONTH,cs$COWSDAY, sep="-"), "%Y-%m-%d"), tz = "GMT")
```

```
end = as.POSIXct(strptime(paste(cs$COWEYEAR,
  cs$COWEMONTH,cs$COWEDAY, sep="-"), "%Y-%m-%d"), tz = "GMT")


###################################################
remove <- c(NA)
begin = begin[! begin %in% remove]
end = end[! end %in% remove]
st = STIDF(geometry(cs), begin, as.data.frame(cs), end)


###################################################
pt = SpatialPoints(cbind(7, 52), CRS(proj4string(cs)))
as.data.frame(st[pt,,1:5])
```

```
##         V1       V2                        sp.ID       time
## 1  9.41437 50.57623 Germany Federal Republic 188 1955-05-05
## 2 10.38084 51.09070                  Germany 187 1990-10-03
##      endTime timeIndex               CNTRY_NAME     AREA CAPNAME
## 1 1990-10-02       188 Germany Federal Republic 247366.4    Bonn
## 2 2016-06-30       187                  Germany 356448.2  Berlin
##   CAPLONG   CAPLAT
## 1     7.1 50.73333
## 2    13.4 52.51667
```