# MATH 8090: Spatio-Temporal Models

Whitney Huang, Clemson University

11/24/2025

## Contents

## Ireland Wind Data

This dataset contains daily average wind-speed observations from 1961 to 1978 recorded at 12 synoptic meteorological stations situated across the Republic of Ireland. The stations form a well-maintained national monitoring network, offering consistent, long-term measurements that enable detailed analysis of temporal trends, spatial dependence, and extreme wind events.

### Load the data

```
library(gstat)
data(wind)
head(wind)
```

```
##   year month day   RPT   VAL   ROS   KIL   SHA  BIR   DUB   CLA   MUL   CLO
## 1   61     1   1 15.04 14.96 13.17  9.29 13.96 9.87 13.67 10.25 10.83 12.58
## 2   61     1   2 14.71 16.88 10.83  6.50 12.62 7.67 11.50 10.04  9.79  9.67
## 3   61     1   3 18.50 16.88 12.33 10.13 11.17 6.17 11.25  8.04  8.50  7.67
## 4   61     1   4 10.58  6.63 11.75  4.58  4.54 2.88  8.63  1.79  5.83  5.88
## 5   61     1   5 13.33 13.25 11.42  6.17 10.71 8.21 11.92  6.54 10.92 10.34
## 6   61     1   6 13.21  8.12  9.96  6.67  5.37 4.50 10.67  4.42  7.17  7.50
##     BEL   MAL
## 1 18.50 15.04
## 2 17.54 13.83
## 3 12.75 12.71
## 4  5.46 10.88
## 5 12.92 11.83
## 6  8.12 13.17
```

```
summary(wind)
```

```
##       year          month             day             RPT
##  Min.   :61.0   Min.   : 1.000   Min.   : 1.00   Min.   : 0.67
##  1st Qu.:65.0   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 8.12
##  Median :69.5   Median : 7.000   Median :16.00   Median :11.71
##  Mean   :69.5   Mean   : 6.523   Mean   :15.73   Mean   :12.36
##  3rd Qu.:74.0   3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.:15.92
##  Max.   :78.0   Max.   :12.000   Max.   :31.00   Max.   :35.80
##       VAL             ROS             KIL             SHA
##  Min.   : 0.21   Min.   : 1.50   Min.   : 0.000   Min.   : 0.13
##  1st Qu.: 6.67   1st Qu.: 8.00   1st Qu.: 3.580   1st Qu.: 6.75
##  Median :10.17   Median :10.92   Median : 5.750   Median : 9.96
##  Mean   :10.65   Mean   :11.66   Mean   : 6.306   Mean   :10.46
##  3rd Qu.:14.04   3rd Qu.:14.67   3rd Qu.: 8.420   3rd Qu.:13.54
##  Max.   :33.37   Max.   :33.84   Max.   :28.460   Max.   :37.54
##       BIR             DUB             CLA             MUL
```

```
##  Min.   : 0.000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 4.000   1st Qu.: 6.000   1st Qu.: 5.090   1st Qu.: 5.370
##  Median : 6.830   Median : 9.210   Median : 8.080   Median : 8.170
##  Mean   : 7.092   Mean   : 9.797   Mean   : 8.494   Mean   : 8.496
##  3rd Qu.: 9.670   3rd Qu.:12.960   3rd Qu.:11.420   3rd Qu.:11.210
##  Max.   :26.160   Max.   :30.370   Max.   :31.080   Max.   :25.880
##       CLO              BEL              MAL
##  Min.   : 0.040   Min.   : 0.13   Min.   : 0.67
##  1st Qu.: 5.330   1st Qu.: 8.71   1st Qu.:10.71
##  Median : 8.290   Median :12.50   Median :15.00
##  Mean   : 8.707   Mean   :13.12   Mean   :15.60
##  3rd Qu.:11.630   3rd Qu.:16.88   3rd Qu.:19.83
##  Max.   :28.210   Max.   :42.38   Max.   :42.54
```

```
wind.loc
```

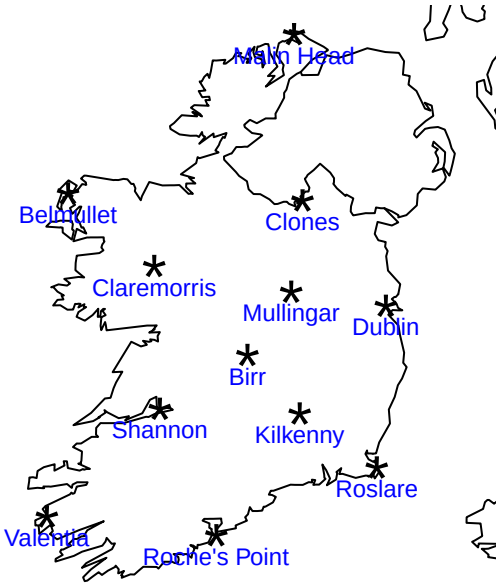
```
##           Station Code       Latitude     Longitude MeanWind
## 1       Valentia  VAL        51d56'N       10d15'W     5.48
## 2      Belmullet  BEL        54d14'N       10d00'W     6.75
## 3    Claremorris  CLA        53d43'N        8d59'W     4.32
## 4        Shannon  SHA        52d42'N        8d55'W     5.38
## 5  Roche's Point  RPT        51d48'N        8d15'W     6.36
## 6           Birr  BIR        53d05'N        7d53'W     3.65
## 7      Mullingar  MUL        53d32'N        7d22'W     4.38
## 8     Malin Head  MAL        55d22'N        7d20'W     8.03
## 9       Kilkenny  KIL        52d40'N        7d16'W     3.25
## 10        Clones  CLO        54d11'N        7d14'W     4.48
## 11        Dublin  DUB        53d26'N        6d15'W     5.05
## 12       Roslare  ROS 52d16'56.791"N 6d21'25.056"W     6.00
```

```
library(sp) # char2dms
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
```

We load the Ireland wind data from `gstat`, explore the wind speeds, convert station latitudes and longitudes from degree–minute–second strings to numeric decimal degrees, and then turn the station table `wind.loc` into a spatial object with coordinates, ready for geostatistical analysis.

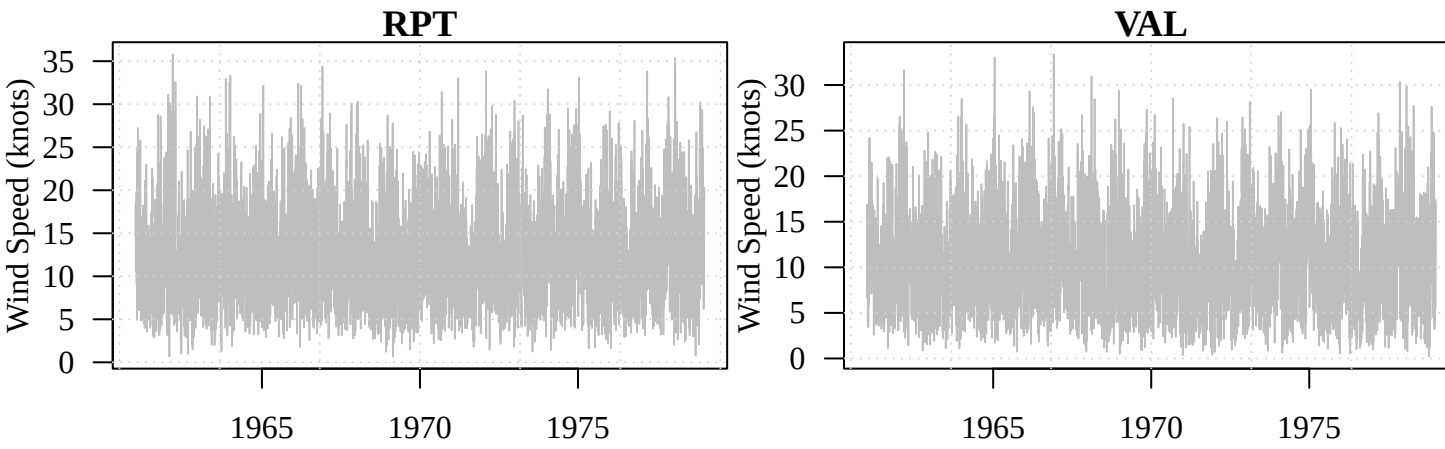
### Spatial locations

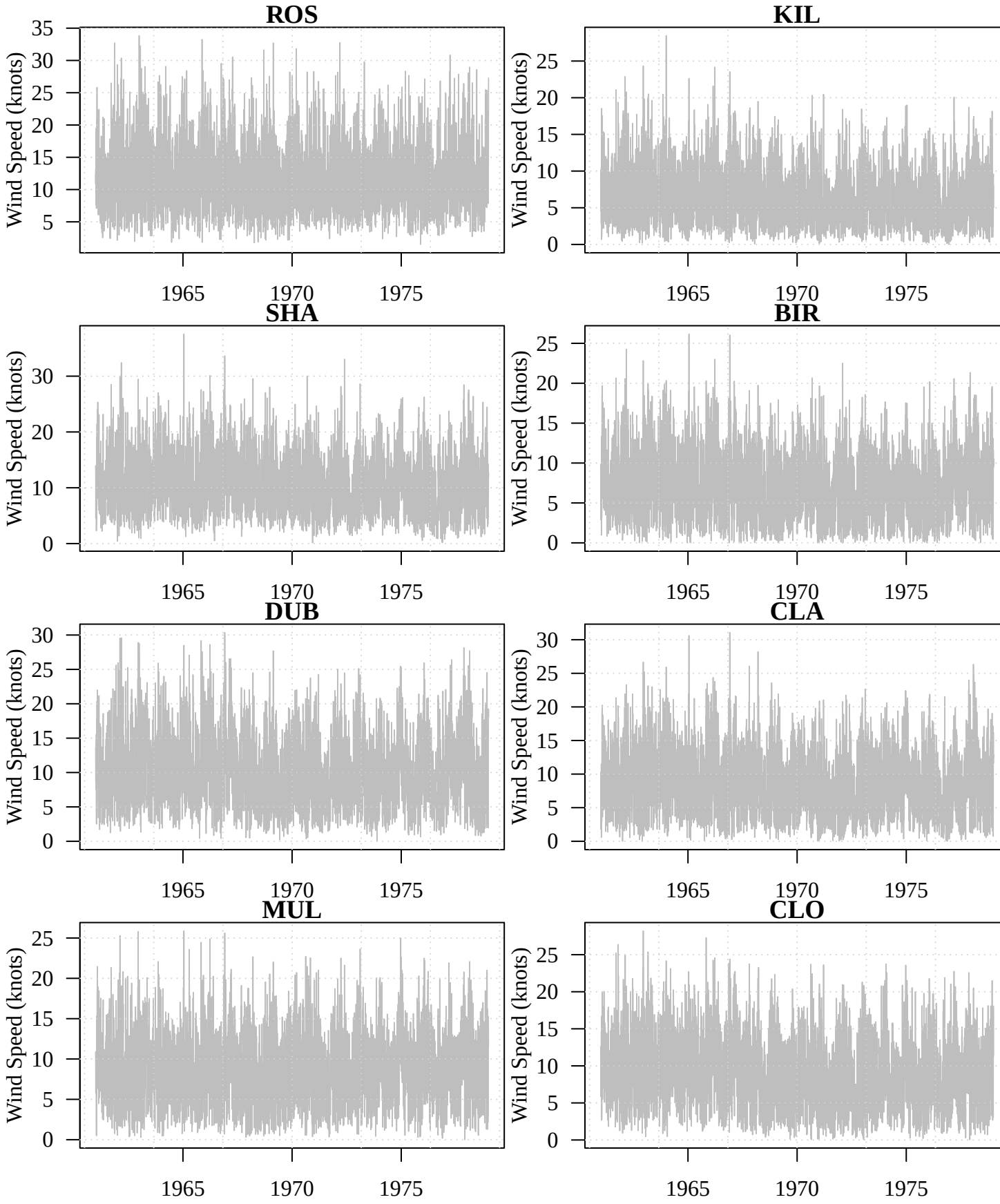
```
library(maps)
library(mapdata)
map("worldHires", xlim = c(-11.5, -5), ylim = c(51, 55.5))
points(wind.loc, pch = "*", cex = 2)
text(coordinates(wind.loc), pos = 1, label = wind.loc$Station, cex = 0.75, col = "blue")
```

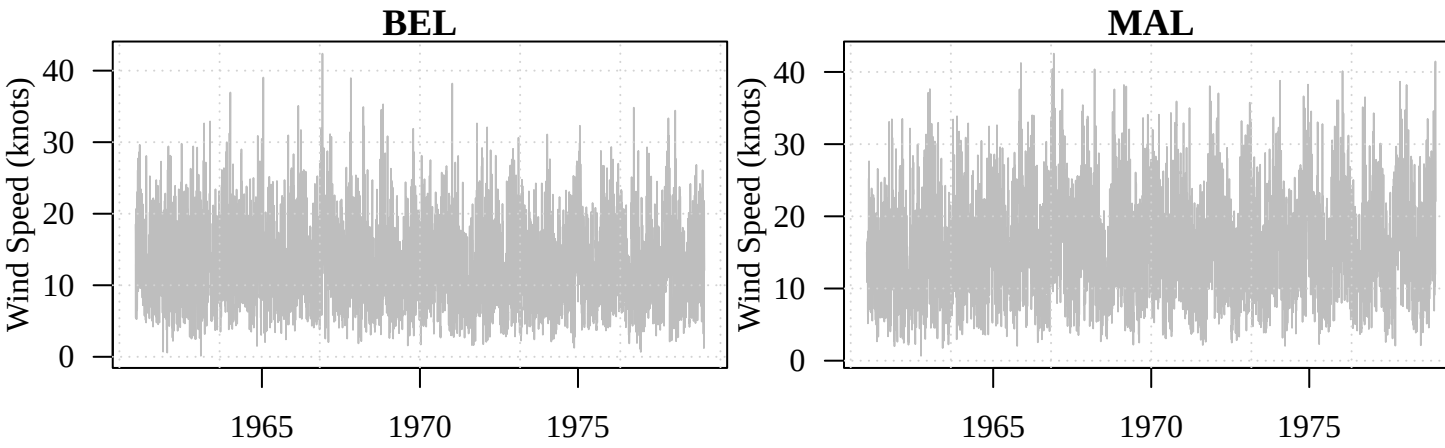



**Time series plot**

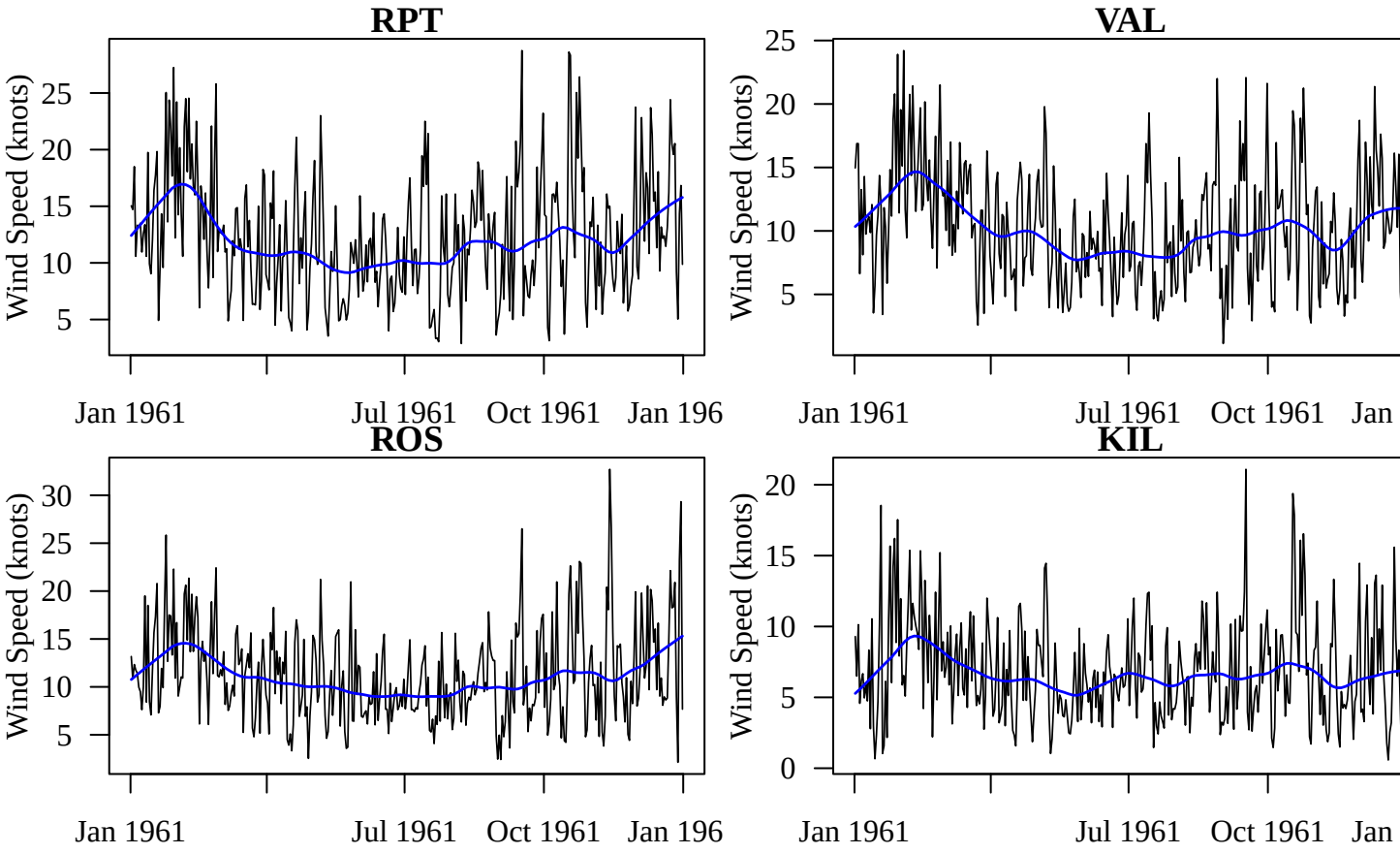
```
par(las = 1, mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1, 0.5), family = "serif")
wind$time = ISOdate(wind$year + 1900, wind$month, wind$day)
for (i in 4:15){
  plot(wind[, i] ~ time, wind, type =  'l', ylab = "Wind Speed (knots)", main = colnames(wind)[i], lwd =
       xlab = "")
  grid()
}
```

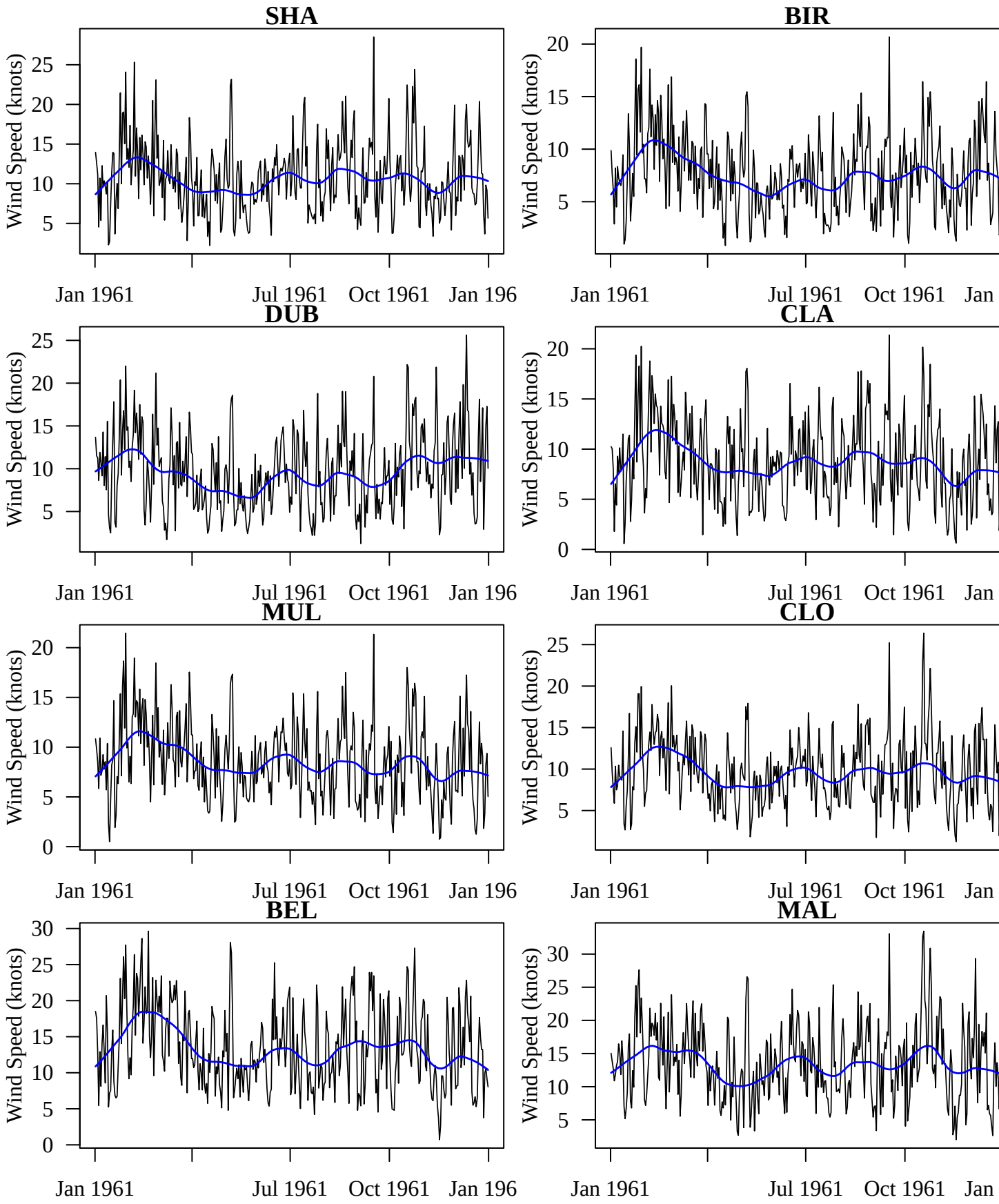



ROS
KIL
SHA
BIR
DUB
CLA
MUL
CLO
Wind Speed (knots)
1965
1970
1975

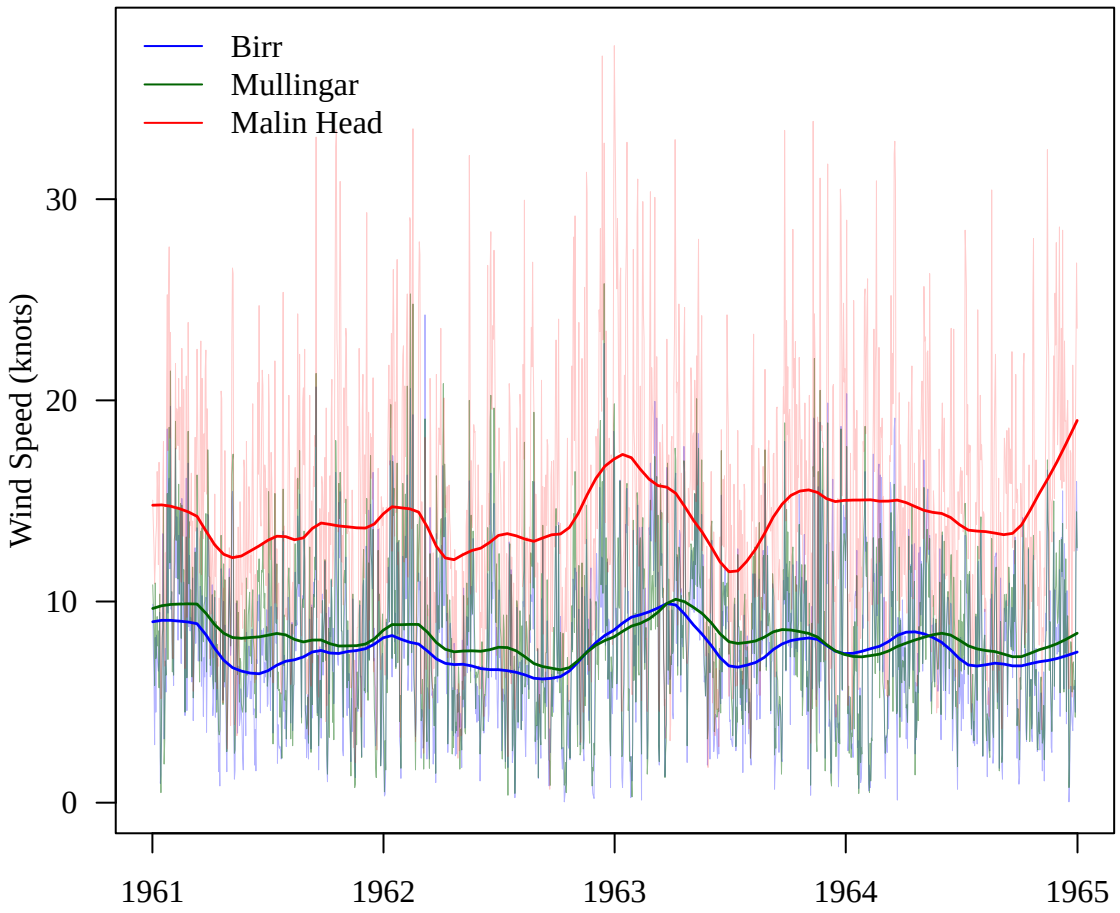



```
for (i in 4:15){
  plot(wind[1:365, i] ~ time[1:365], wind, type =  'l', ylab = "Wind Speed (knots)",
       main = colnames(wind)[i], lwd = 0.85, xlab = "")
  lines(lowess(wind[1:365, i] ~ wind$time[1:365], f = 0.15), col = "blue", lwd = 1.4)
}
```




SHA
BIR
DUB
CLA
MUL
CLO
BEL
MAL
Wind Speed (knots)
Jan 1961
Jul 1961
Oct 1961
Jan 196
Jan

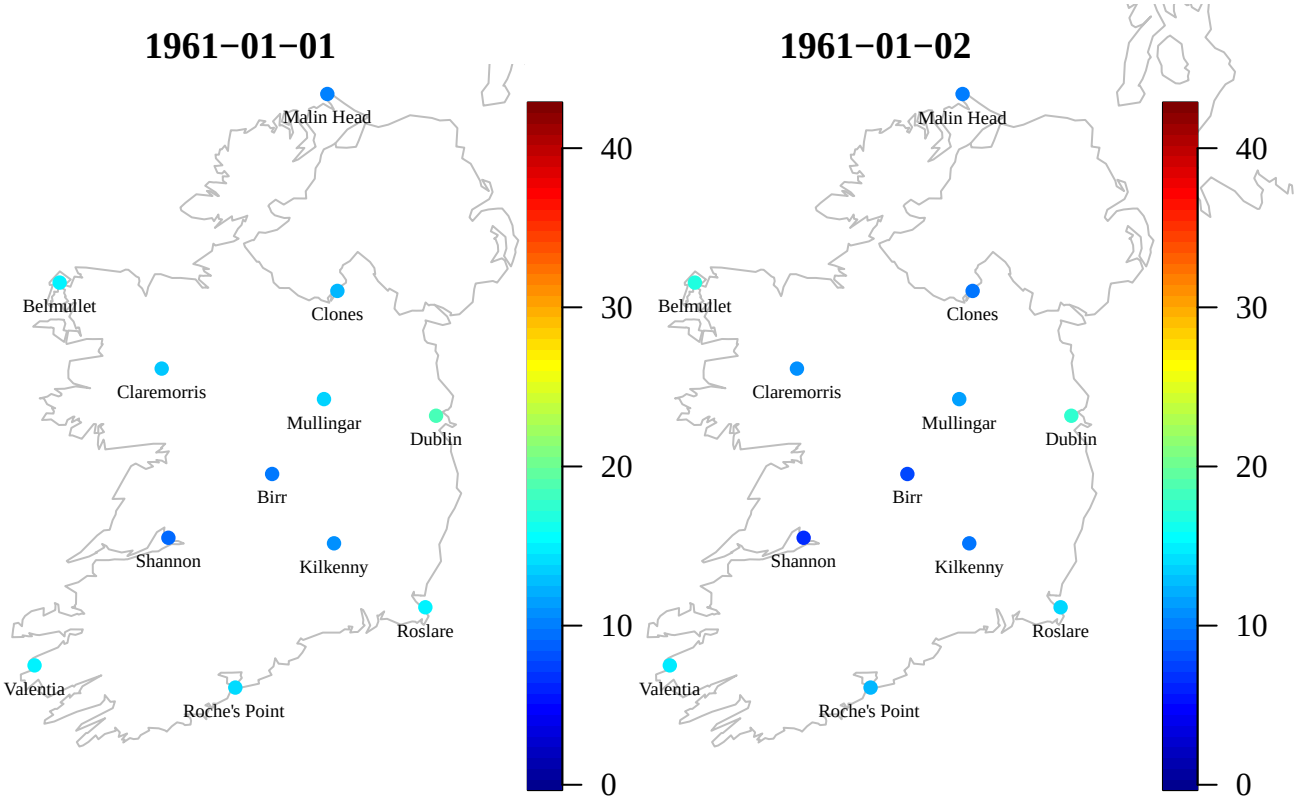
```
library(scales)
par(las = 1, mgp = c(2, 1, 0), mar = c(3, 3.5, 0.5, 0.5), family = "serif")
plot(wind[1:1461, 9] ~ time[1:1461], wind, type =  'l', ylab = "Wind Speed (knots)",
     main = "", lwd = 0.5, xlab = "", col = alpha("blue", 0.3), ylim = c(0, 38))
lines(wind$time[1:1461], wind[1:1461, 12], col = alpha("darkgreen", 0.45), lwd = 0.5)
lines(wind$time[1:1461], wind[1:1461, 15], col = alpha("red", 0.2), lwd = 0.5)

lines(lowess(wind[1:1461, 9] ~ wind$time[1:1461], f = 0.1), col = "blue", lwd = 1.4)
lines(lowess(wind[1:1461, 12] ~ wind$time[1:1461], f = 0.1), col = "darkgreen", lwd = 1.4)
lines(lowess(wind[1:1461, 15] ~ wind$time[1:1461], f = 0.1), col = "red", lwd = 1.4)
legend("topleft", legend = c("Birr", "Mullingar", "Malin Head"),
       col = c("blue", "darkgreen", "red"), lty = 1, bty = "n")
```

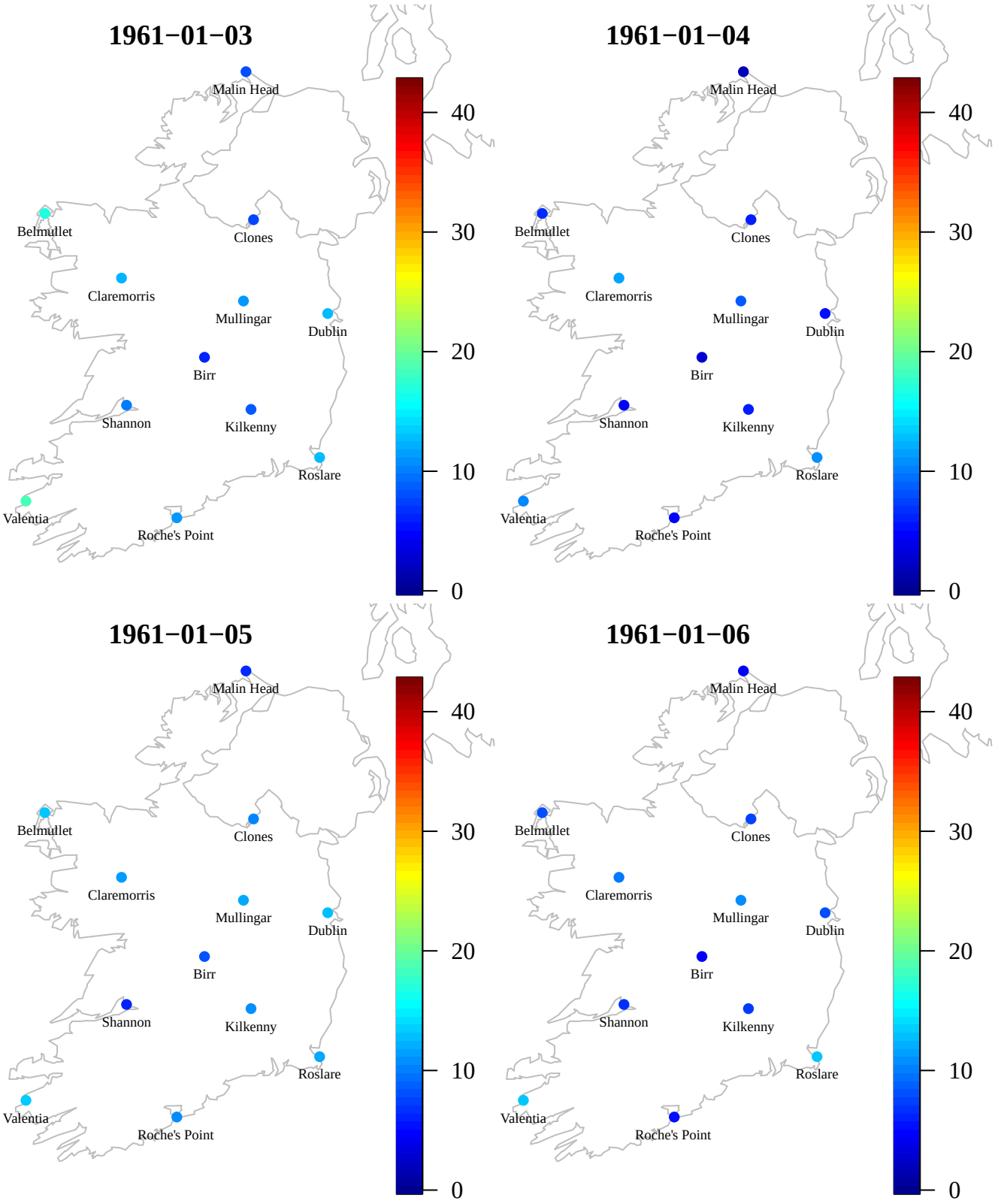


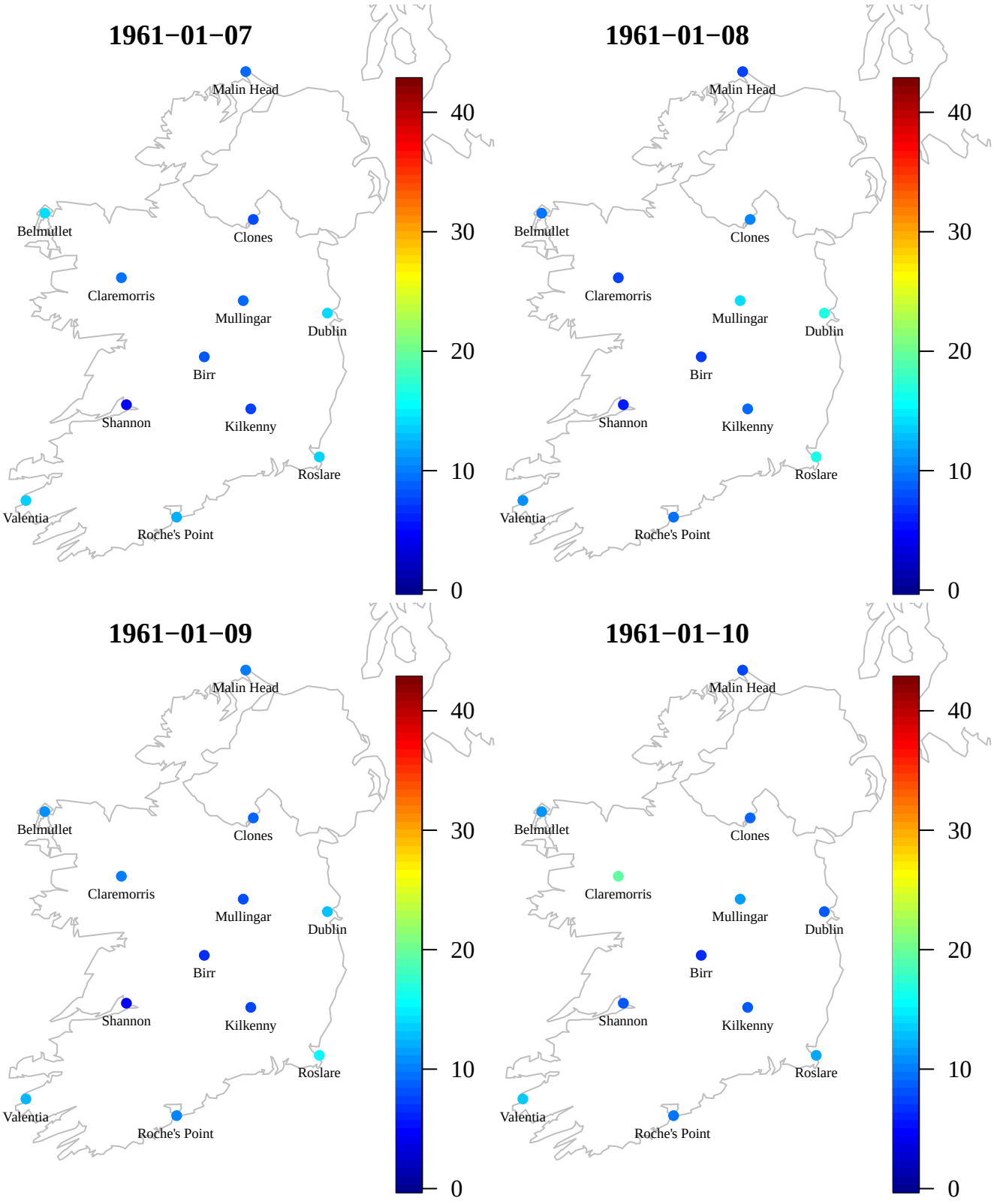

**Spatial snapshots**

```
library(fields)
# Winter
col <- color.scale(unlist(wind[, 4:15]))
par(las = 1, mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1, 0.5), family = "serif")
for (t in 1:10){
  map("worldHires", xlim = c(-11, -5.4), ylim = c(51, 55.5),
      col = "gray")
```

```
  title(main  = as.Date(wind$time[t]))
  points(wind.loc, pch = 16, col = col[seq(t, 11 * 6574 + t, 6574)])
  image.plot(legend.only = T, zlim = range(unlist(wind[, 4:15])))
  text(coordinates(wind.loc), pos = 1, label = wind.loc$Station, cex = 0.6)
}
```

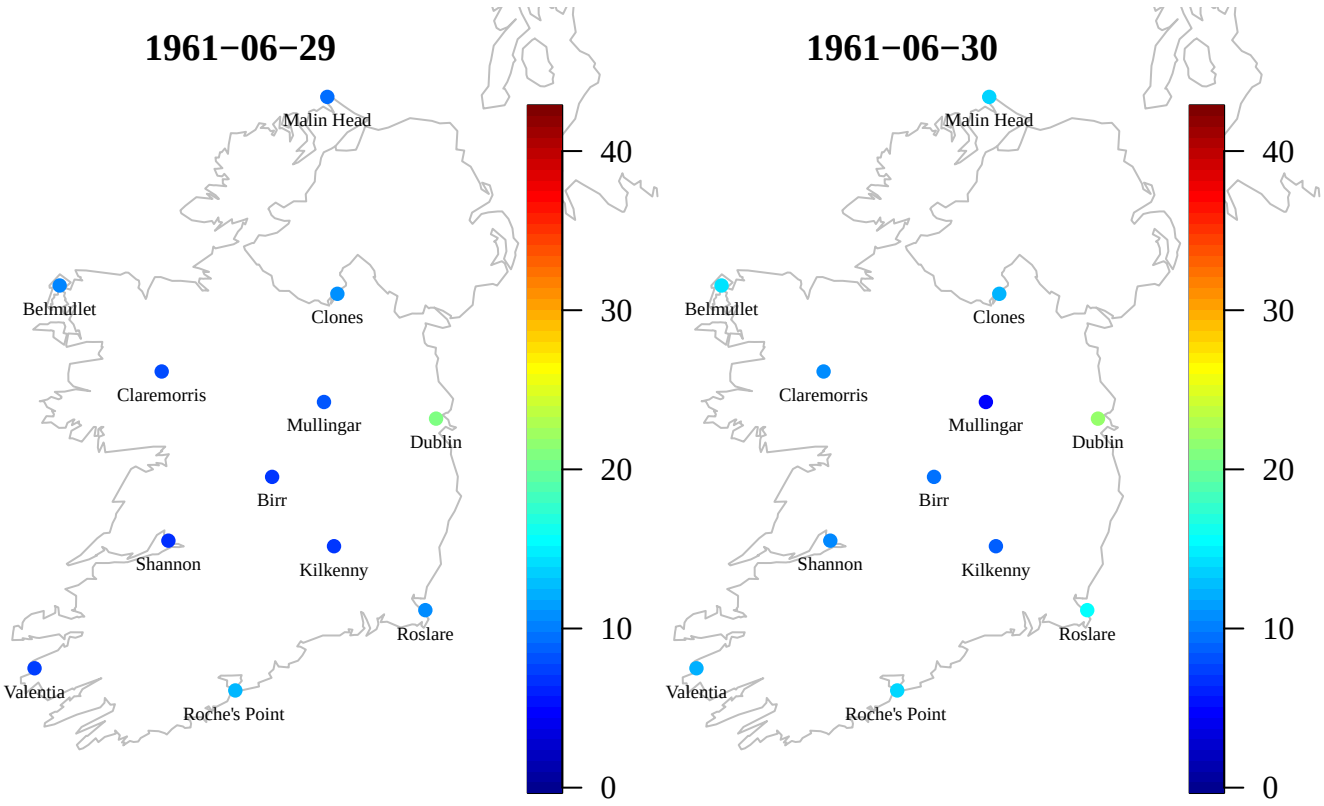


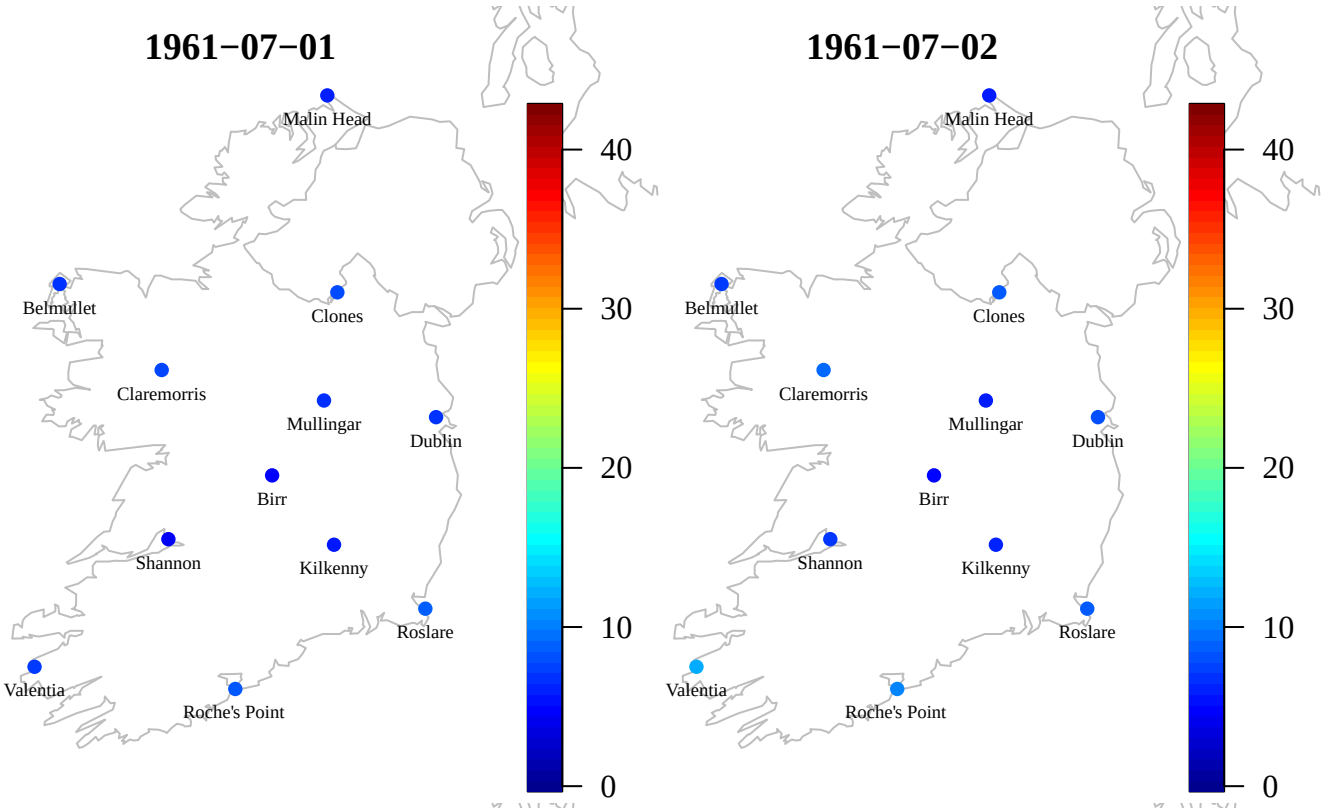

1961−01−03
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0
1961−01−04
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0
1961−01−05
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0
1961−01−06
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0



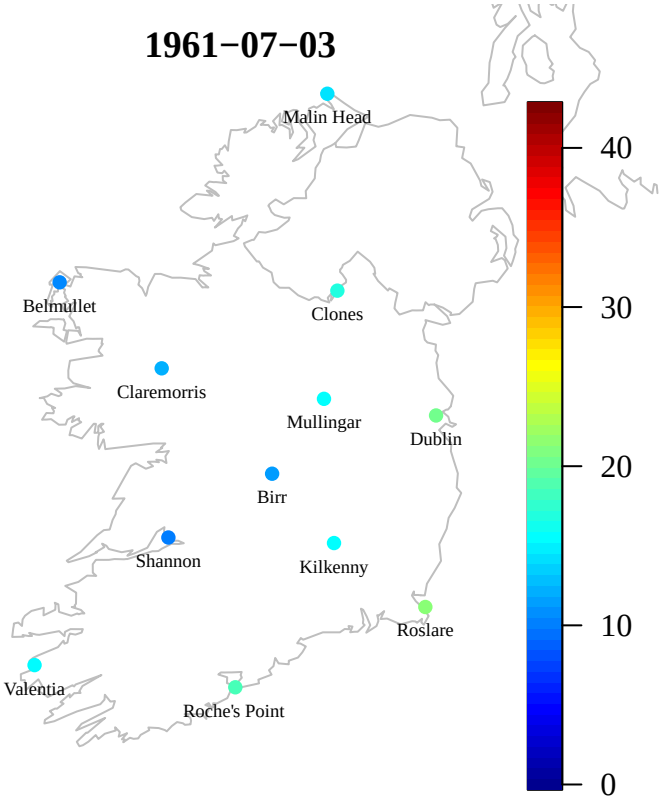

```
#Summer
for (t in 180:189){
  map("worldHires", xlim = c(-11, -5.4), ylim = c(51, 55.5),
```

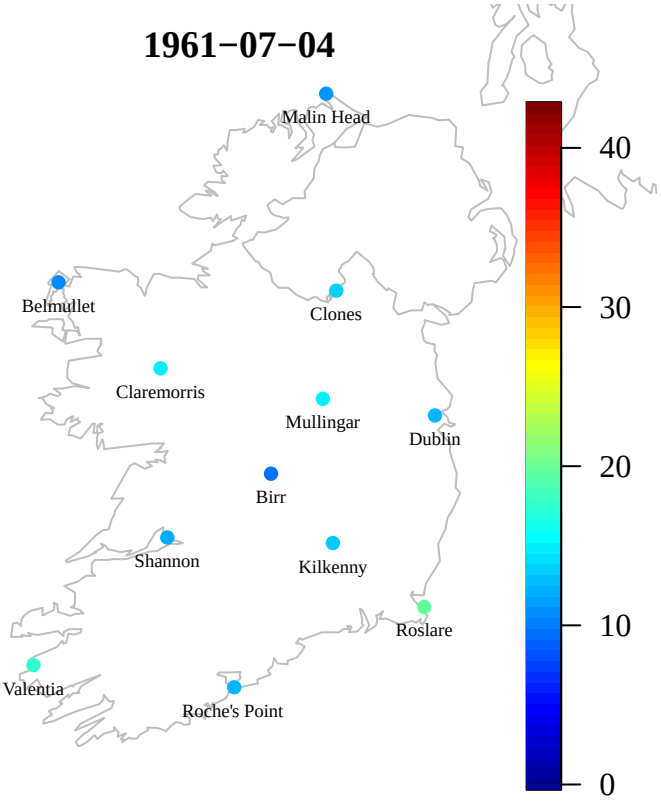
```
      col = "gray")
  title(main  = as.Date(wind$time[t]))
  points(wind.loc, pch = 16, col = col[seq(t, 11 * 6574 + t, 6574)])
  image.plot(legend.only = T, zlim = range(unlist(wind[, 4:15])))
  text(coordinates(wind.loc), pos = 1, label = wind.loc$Station, cex = 0.6)
}
```

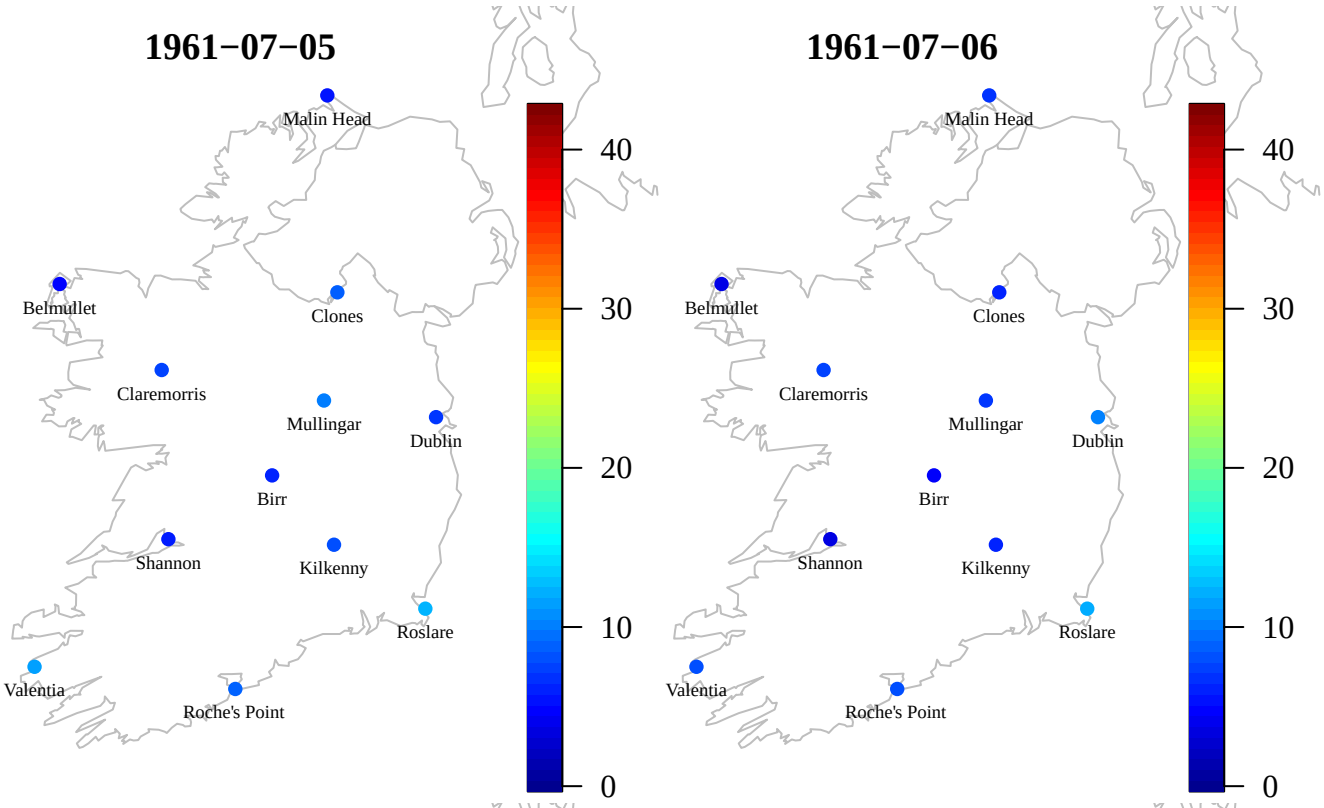



1961−07−01
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0

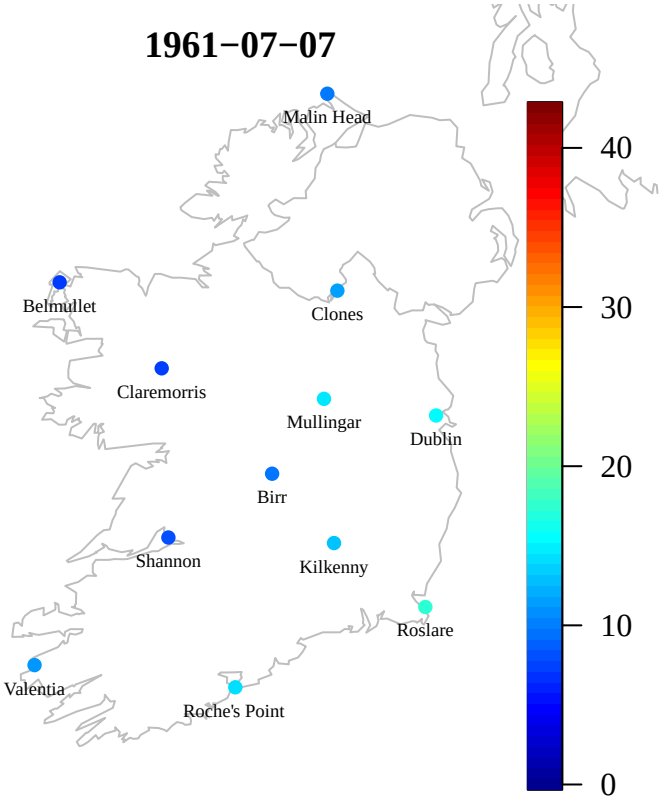

1961−07−02
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0

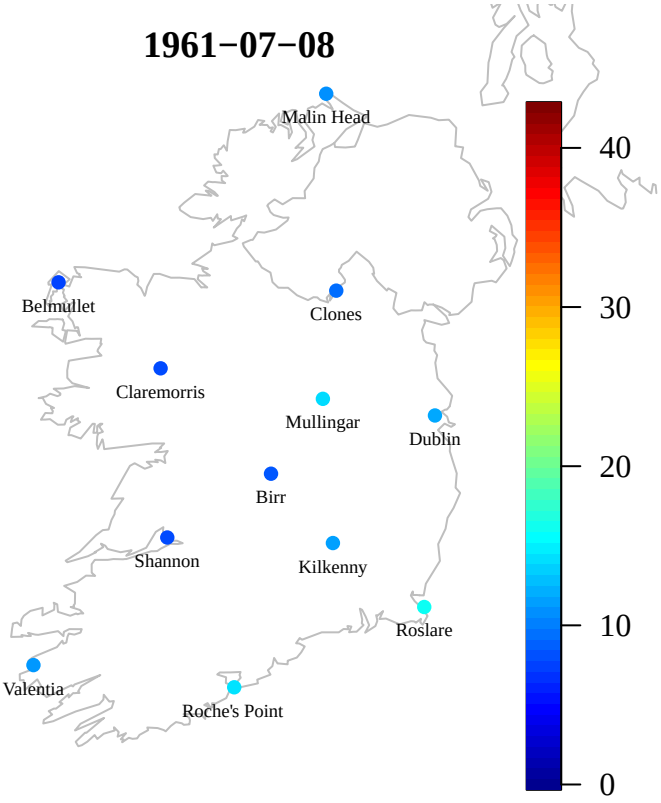

1961−07−03
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0


1961−07−04
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0


1961−07−05
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0

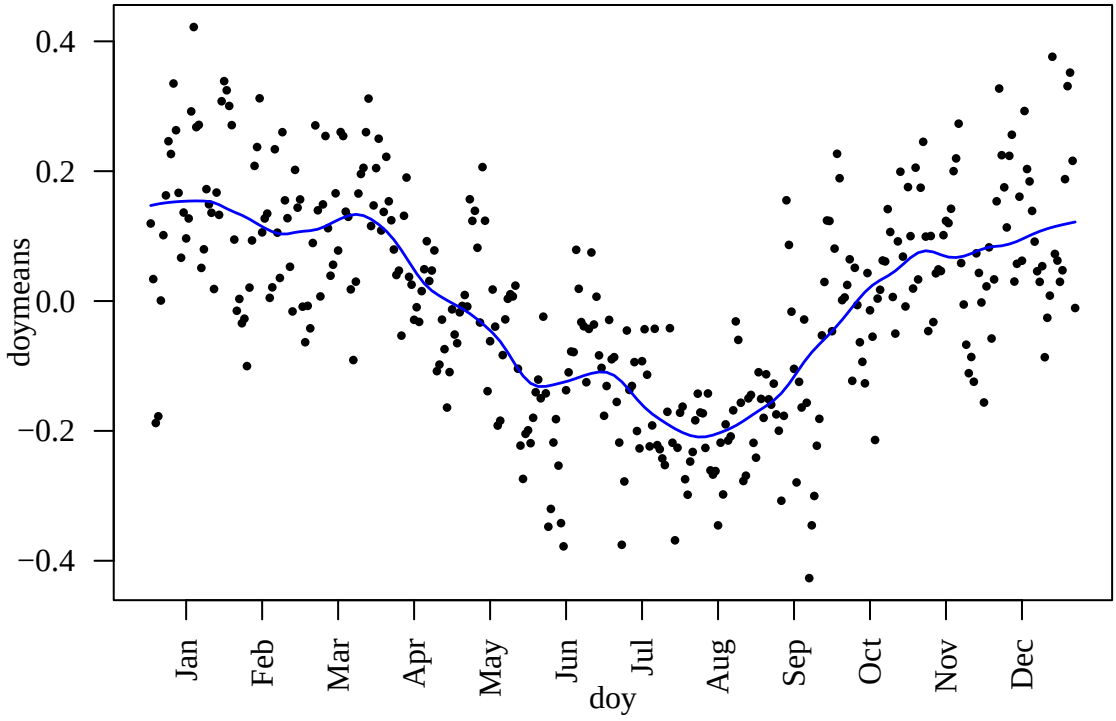

1961−07−06
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0


1961−07−07
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0


1961−07−08
Malin Head
Belmullet
Clones
Claremorris
Mullingar
Dublin
Birr
Shannon
Kilkenny
Roslare
Valentia
Roche's Point
40
30
20
10
0

**Annual cycle**

```
#wind = wind[!(wind$month == 2 & wind$day == 29),]
par(las = 1, mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1, 0.5), family = "serif")
wind$doy = as.numeric(format(wind$time, '%j'))
windsqrt = sqrt(0.5148 * as.matrix(wind[4:15]))
doy = 1:366
windsqrt_res = apply(windsqrt, 2, function(x) x - mean(x))
doymeans = sapply(split(windsqrt_res, wind$doy), mean)
plot(doymeans ~ doy, pch = 16, cex = 0.6, las = 1, xaxt = "n")
axis(1, at = seq(15, 345, 30), labels = month.abb, las = 2)
lines(lowess(doymeans ~ doy, f = 0.15), col = "blue", lwd = 1.4)
```




We begin by converting each date to a day-of-year index, which allows us to pool the same calendar day across multiple years (e.g., all January 15 observations). This transformation is standard in climatology because it enables estimation of an annual cycle by aggregating information across time rather than treating each year in isolation.

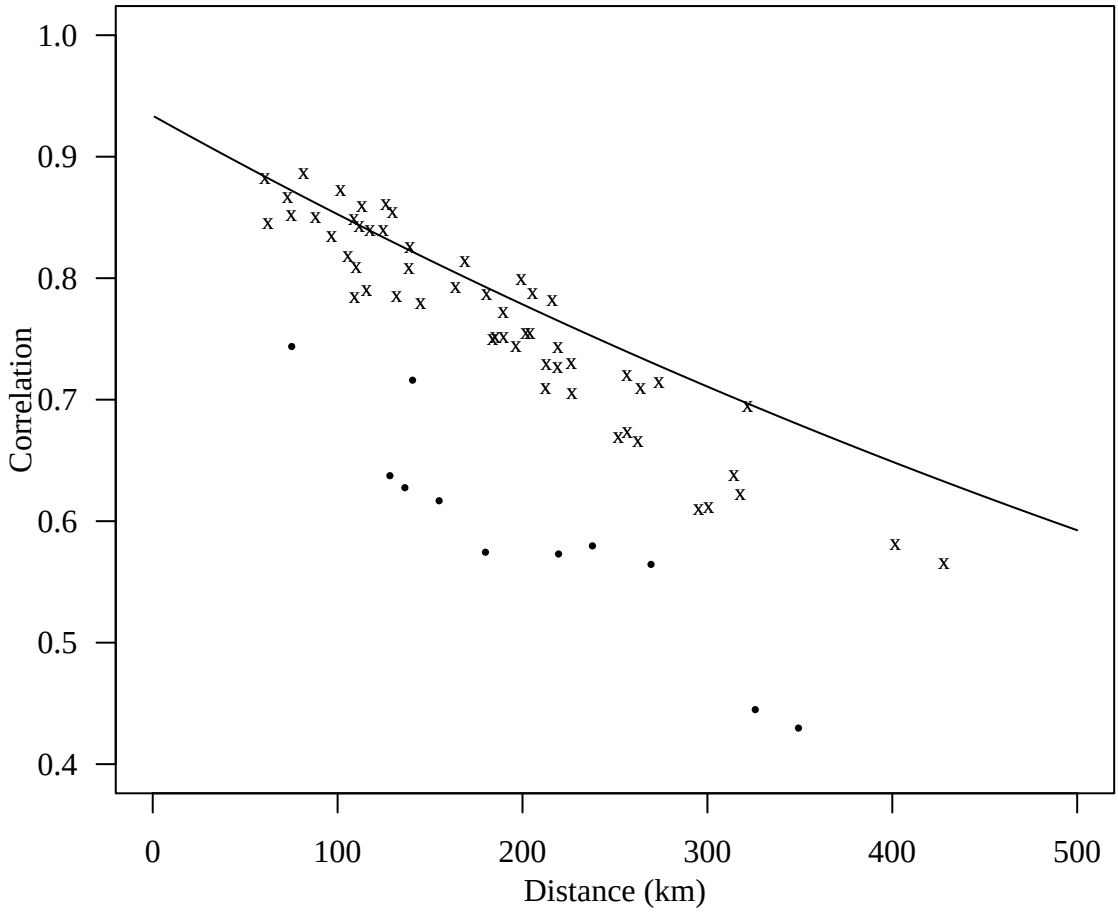
Wind speeds are provided in knots, so we convert them to meters per second using the standard factor 0.5148. A square-root transformation is then applied, which is commonly used for wind data because it reduces skewness and stabilizes variance, facilitating comparison across stations and seasons. Since each station has its own climatological mean wind level, we subtract the station-specific mean from its entire time series, producing detrended anomalies centered near zero. This step removes differences in baseline windiness across locations and allows us to focus on the shared seasonal structure common to all sites.

Next, we compute the average wind anomaly for each day of the year, pooling data from all years in the 1961–1978 period. By aggregating across nearly two decades of observations, we suppress short-term weather noise and reveal the underlying, recurring seasonal pattern. This approach is widely used in atmospheric and environmental sciences to form "climatologies"—long-term averages that characterize typical meteorological behavior.

Finally, the daily averages are visualized using a scatterplot with a LOWESS smoothing curve overlaid. The scatterplot displays the raw daily variability, while the smoothed curve highlights the dominant seasonal cycle without imposing any parametric structure. The resulting figure provides an intuitive and statistically robust depiction of how wind speeds evolve throughout the year, with increased clarity due to the careful preprocessing steps undertaken earlier.

### Spatial correlation vs. distance

```
par(las = 1, mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1, 0.5), family = "serif")
# subtract the seasonality:
meanwind = lowess(doymeans ~ doy, f = 0.1)$y[wind$doy]
velocity = apply(windsqrt, 2, function(x) {x - meanwind})
# match order of columns in wind to Code in wind.loc:
pts = coordinates(wind.loc[match(names(wind[4:15]), wind.loc$Code),])
# calculate
dists = spDists(pts, longlat = TRUE)
corv = cor(velocity)
ros = rownames(corv) == "ROS"
dists.nr = dists[!ros, !ros]
corv.nr = corv[!ros, !ros]
sel = !(as.vector(dists.nr) == 0)
plot(as.vector(corv.nr[sel]) ~ as.vector(dists.nr[sel]), pch = "X",
    xlim = c(0, 500), ylim = c(.4, 1), xlab = "Distance (km)",
    ylab = "Correlation", las = 1, cex = 0.5)
# add outlier:
points(corv[ros, !ros] ~ dists[ros, !ros], pch = 16, cex = .5)
xdiscr = 1:500
# add correlation model:
lm <- lm(as.vector(corv.nr[sel]) ~ as.vector(dists.nr[sel]))
lines(xdiscr, lm$coefficients[1] * exp(lm$coefficients[2] * xdiscr))
```




To investigate how spatial dependence in wind speeds decays with distance, we first remove the dominant seasonal cycle from each station's time series. A smooth LOWESS curve is fitted to the day-of-year averages, and this estimated seasonal component is subtracted from each observation to obtain deseasonalized wind anomalies. These residuals reflect short-term weather variability rather than long-term climatological structure, making them more appropriate for studying spatial correlation. Next, the station locations in `wind.loc` are reordered to match the column structure of the wind-speed data, ensuring that pairwise distances and correlations correspond to the same station pairs. Great-circle distances between all stations are then computed, along with the correlation matrix of the deseasonalized wind anomalies. Because one station (ROS) often behaves atypically, its correlations are treated separately to assess whether it aligns with or deviates from the general spatial pattern. The pairwise correlations for all non-ROS station pairs are plotted against their corresponding distances, producing an empirical view of how dependence weakens as stations become farther apart. The ROS station's correlations are then added to the plot to highlight any outlier behavior. Finally, a simple exponential decay curve is fitted using a linear regression on the correlation–distance relationship and overlaid on the plot, providing a convenient parametric summary of the observed spatial correlation structure.

## Annual Mean PM Data from the EPA AQS Dataset

```
dat <- read.csv("AQS_annual_mean_2005_2014.csv", header = TRUE)
dim(dat)
```

```
## [1] 50428     6
```

```r
head(dat)
```

```
##   Year      Site Longitude Latitude pollutant mean.obs.value
## 1 2005 100010002  -75.5568  38.9867        O3      0.0494486
## 2 2005 100010002  -75.5556  38.9847      PM25     13.1458333
## 3 2005 100010003  -75.5181  39.1550        EC      0.5897667
## 4 2005 100010003  -75.5181  39.1550       NH4      1.9013000
## 5 2005 100010003  -75.5181  39.1550       NO3      1.9445500
## 6 2005 100010003  -75.5181  39.1550        OC      1.6482184
```

```r
# Pull out a subset
keep <- (dat[, 1] == 2014) & (dat[, 5] == "PM25" | dat[, 5] == "PM10")
dat <- dat[keep,]
dim(dat)
```

```
## [1] 1448    6
```

```r
head(dat)
```

```
##       Year     Site Longitude Latitude pollutant mean.obs.value
## 45741 2014 10030010  -87.8814  30.4980      PM25       8.906838
## 45743 2014 10270001  -85.8022  33.2813      PM25       8.903509
## 45744 2014 10331002  -87.6506  34.7588      PM25       9.290517
## 45746 2014 10491003  -85.9683  34.2876      PM25       9.600917
## 45749 2014 10550010  -85.9911  33.9937      PM25       9.335238
## 45751 2014 10690003  -85.3908  31.2264      PM25       8.702703
```

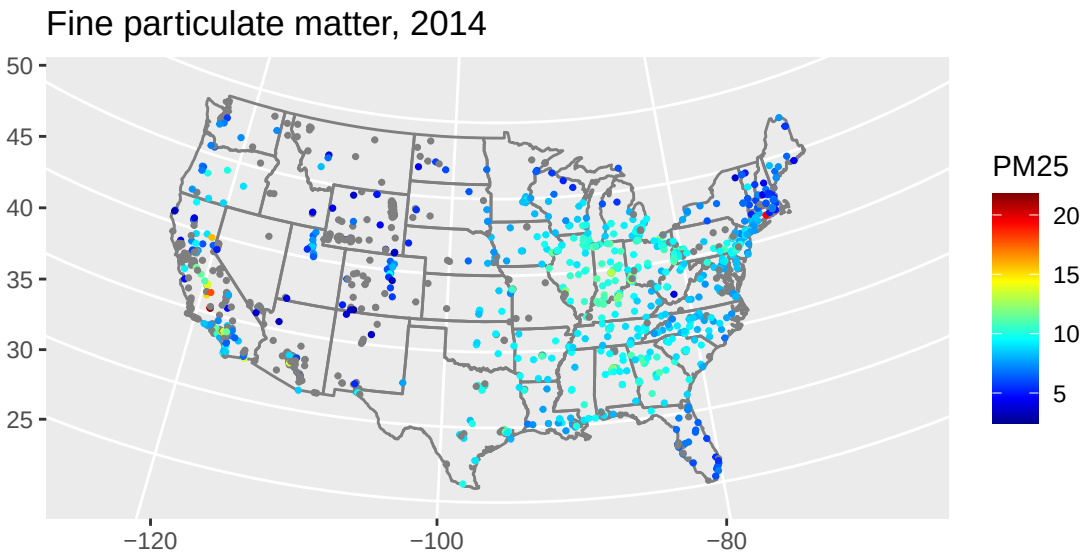
```r
s <- unique(dat[, 3:4])
n <- nrow(s)
dim(s)
```

```
## [1] 1180    2
```

```r
PM25 <- PM10 <- rep(NA, n)
for(i in 1:nrow(dat)){
   j <- which(dat[i, 3] == s[,1] & dat[i, 4] == s[, 2])[1]
   if(dat[i, 5] == "PM25"){
     PM25[j] <- dat[i, 6]
   }
   if(dat[i, 5] == "PM10"){
     PM10[j] <- dat[i, 6]
   }
}

type <- "Both"
type <- ifelse(is.na(PM25), "PM10", type)
type <- ifelse(is.na(PM10), "PM25", type)
table(type)
```

```
## type
## Both PM10 PM25
##  268  425  487
```

```r
library(ggplot2)

par(mar = c(0, 0, 0, 0))
pm <- data.frame(long = s[, 1], lat = s[, 2], PM25 = PM25)
ggplot(pm, aes(long, lat)) +
  borders("state") +
  geom_point(aes(colour = PM25), cex = 0.6) +
  scale_colour_gradientn(colours = tim.colors()) +
  coord_map(projection = "albers", lat0 = 39, lat1 = 45) +
  xlab("") + ylab("") + labs(title = "Fine particulate matter, 2014")
```




## Dynamic Space-Time Modeling of Monthly Temperature in New England

In this example, we analyze *monthly* temperature data (in °C) recorded at monitoring stations across the Northeastern US starting in January 2000.

For each station, we have:

- Geographic coordinates (projected UTM coordinates),
- Elevation (meters),
- Monthly temperature measurements over time.

Our goals are to:

1. Explore the spatial and temporal structure of the data.
2. Examine empirical variograms for selected months.
3. Fit a *dynamic space-time* linear model using `spBayes::spDynLM`.
4. Investigate the posterior behavior of regression and covariance parameters.
5. Assess the in-sample fit and out-of-sample predictive performance.

### Data setup

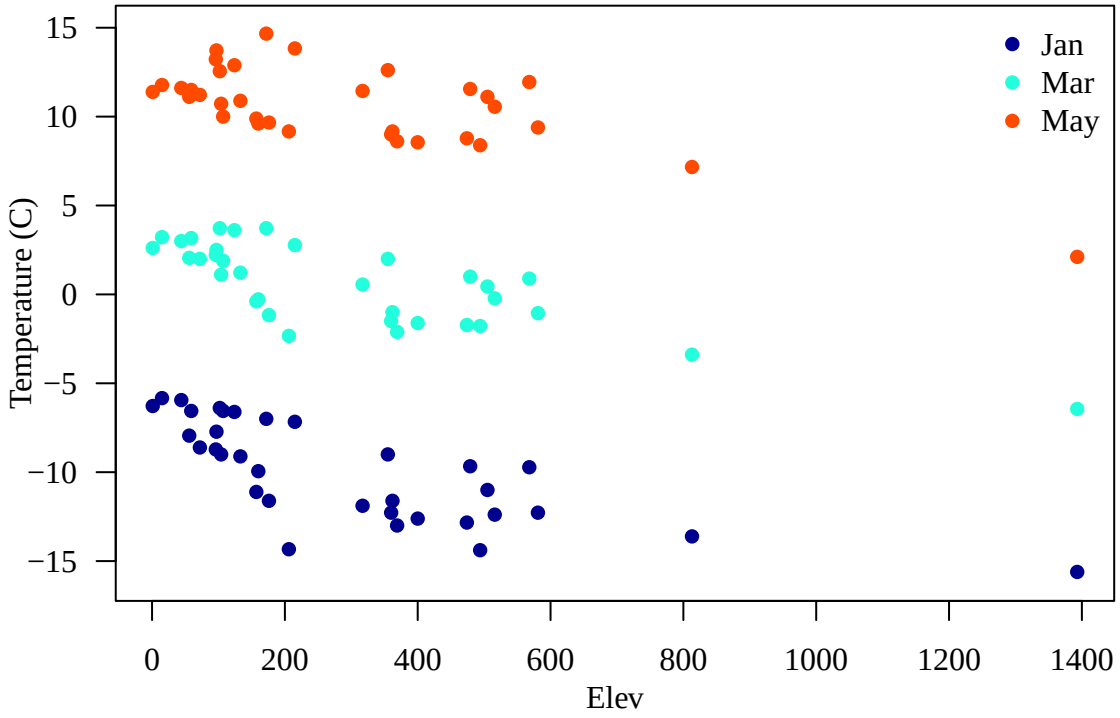
```
library(spBayes)
data("NETemp.dat")
dat <- NETemp.dat
##take a smaller spatial region of New England
dat <- dat[dat[, "UTMX"] > 5500000 & dat[, "UTMY"] > 3000000,]
(dim(dat))
```

```
## [1]  34 132
```

### Exploratory data analysis

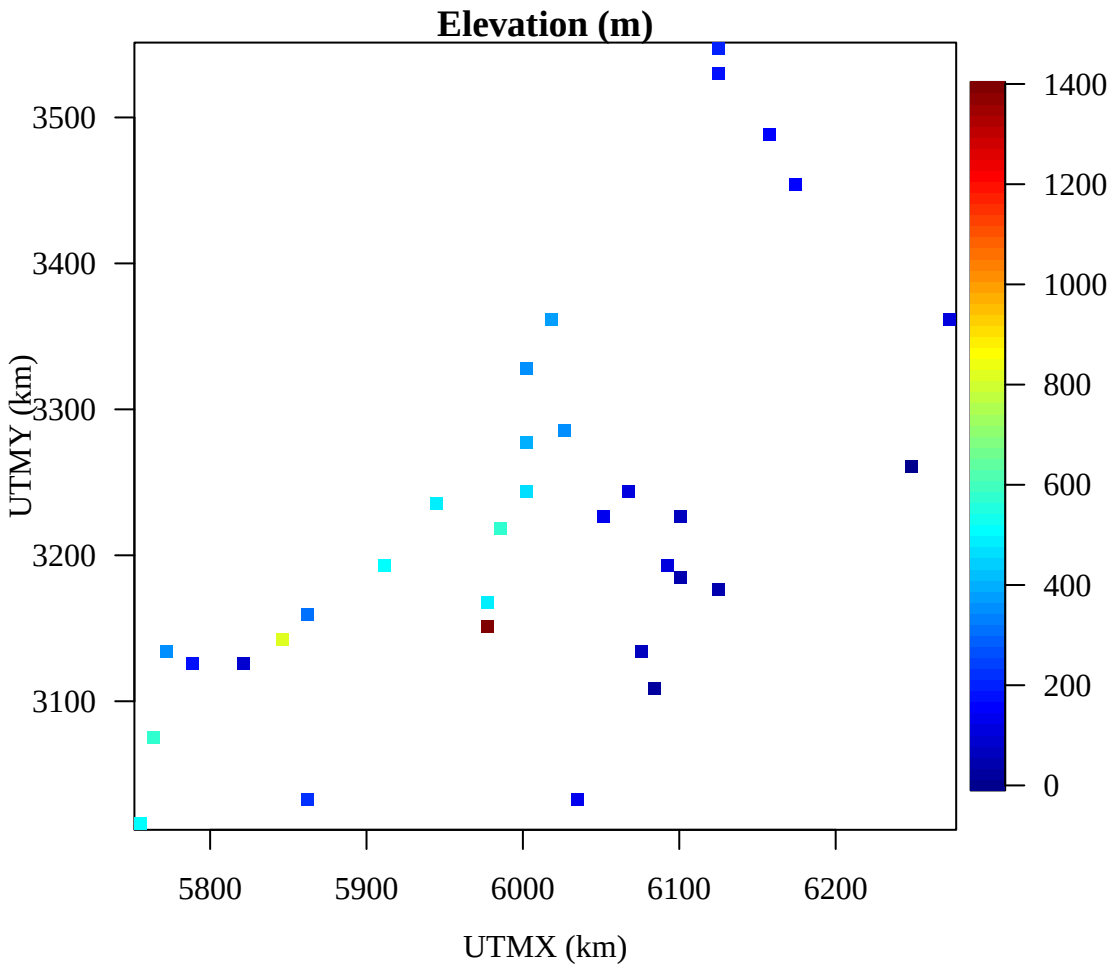
We begin by exploring how temperature relates to elevation, and how both temperature and elevation vary across space.

```
col_mon <- c(tim.colors(6), rev(tim.colors(6)))
par(las = 1, mar =c(3.5, 3.5, 1, 0.5), mgp = c(2, 1, 0), family = "serif")
plot(dat$y.1 ~ dat$elev, pch = 16, col = col_mon[1],
     xlab = "Elev", ylab = "Temperature (C)", ylim = c(-16, 15))
points(dat$y.3 ~ dat$elev, pch = 16, col = col_mon[3])
points(dat$y.5 ~ dat$elev, pch = 16, col = col_mon[5])
legend("topright", legend = month.abb[c(1, 3, 5)],
       col = col_mon[c(1, 3, 5)], pch = 16, bty = "n")
```



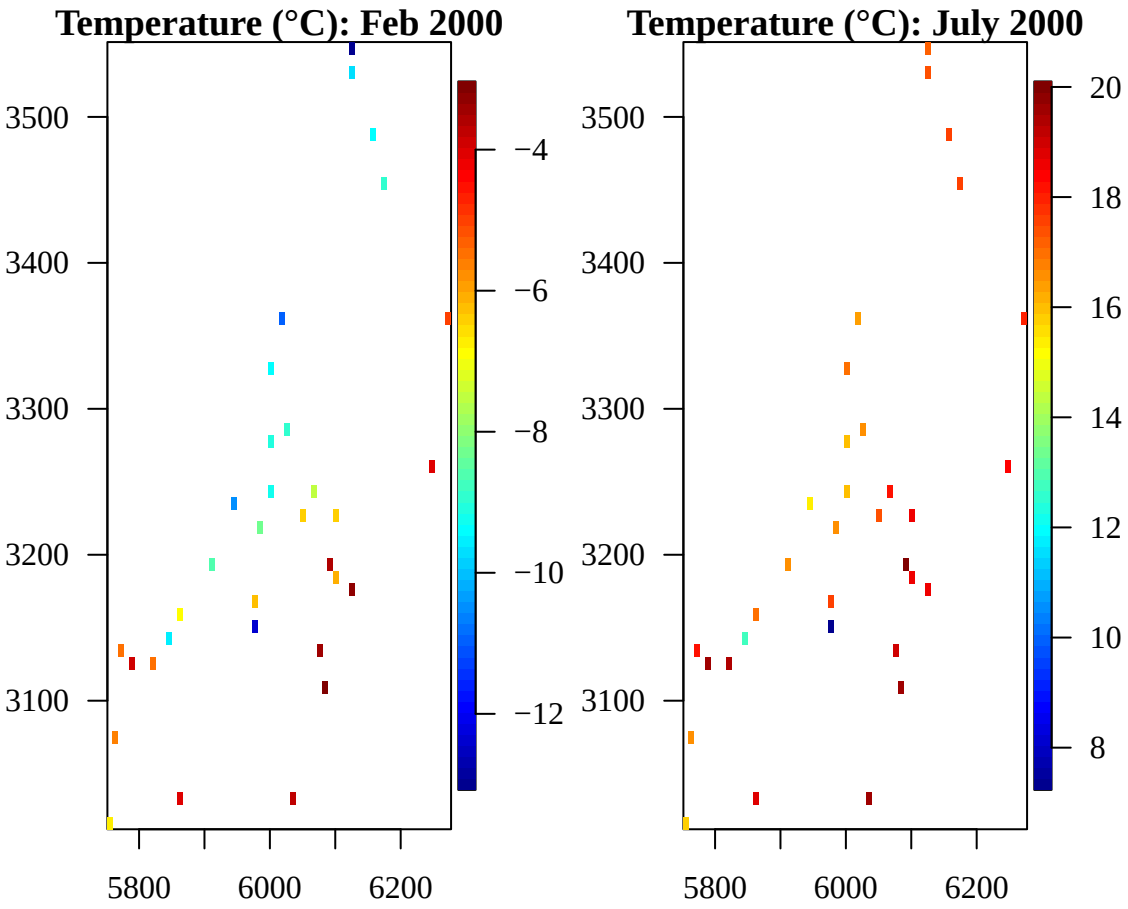

We now plot the elevation field over the study region.

```
par(las = 1, mar =c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
quilt.plot(dat$UTMX / 1000, dat$UTMY / 1000, dat$elev,
           xlab = "UTMX (km)", ylab = "UTMY (km)", main = "Elevation (m)")
```



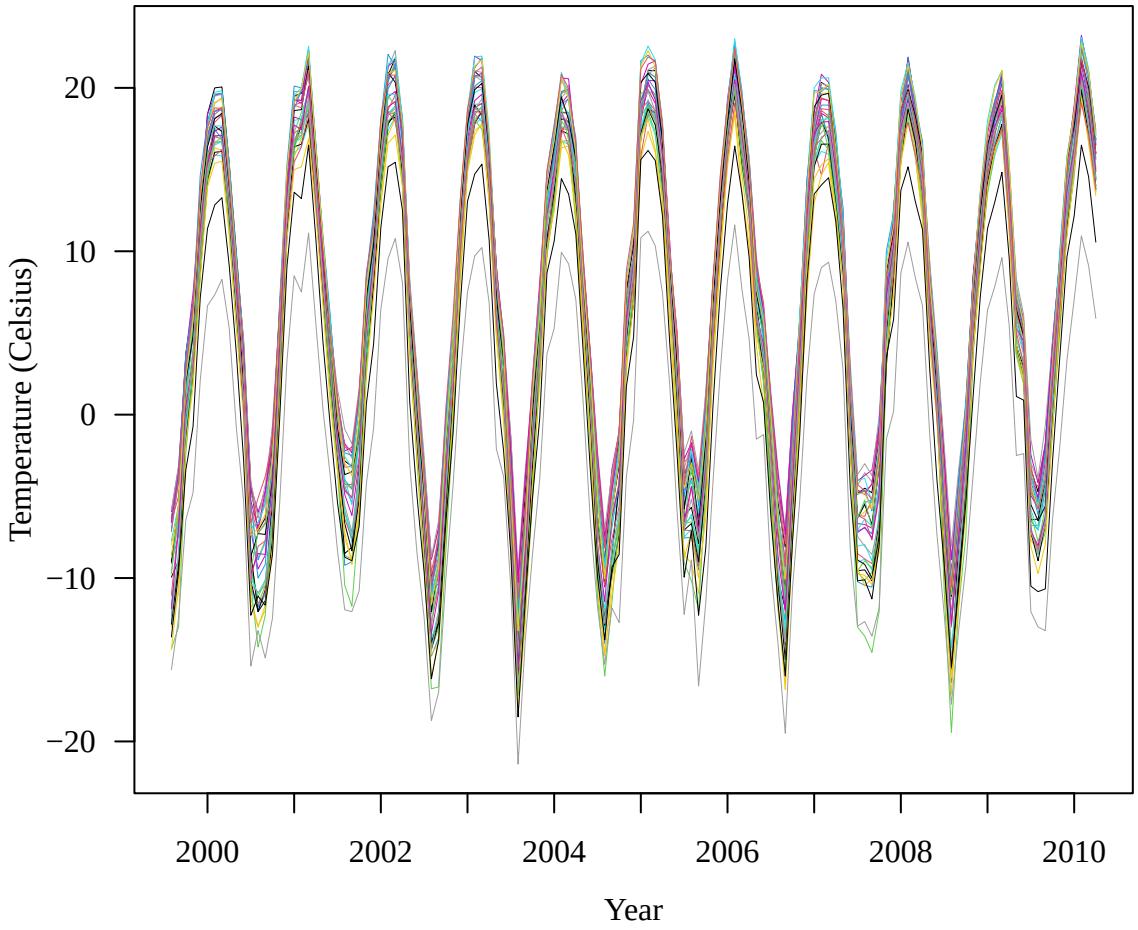

Next, we examine the spatial temperature fields for two contrasting months: February and July 2000.

```
par(mfrow = c(1, 2), las = 1, mar =c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
quilt.plot(dat$UTMX / 1000, dat$UTMY / 1000, dat$y.2, las = 1,
           main = "Temperature (°C): Feb 2000")
quilt.plot(dat$UTMX / 1000, dat$UTMY / 1000, dat$y.7, las = 1,
           main = "Temperature (°C): July 2000")
```




We now look at temperature time series at all stations to get a sense of inter-annual and seasonal variability.

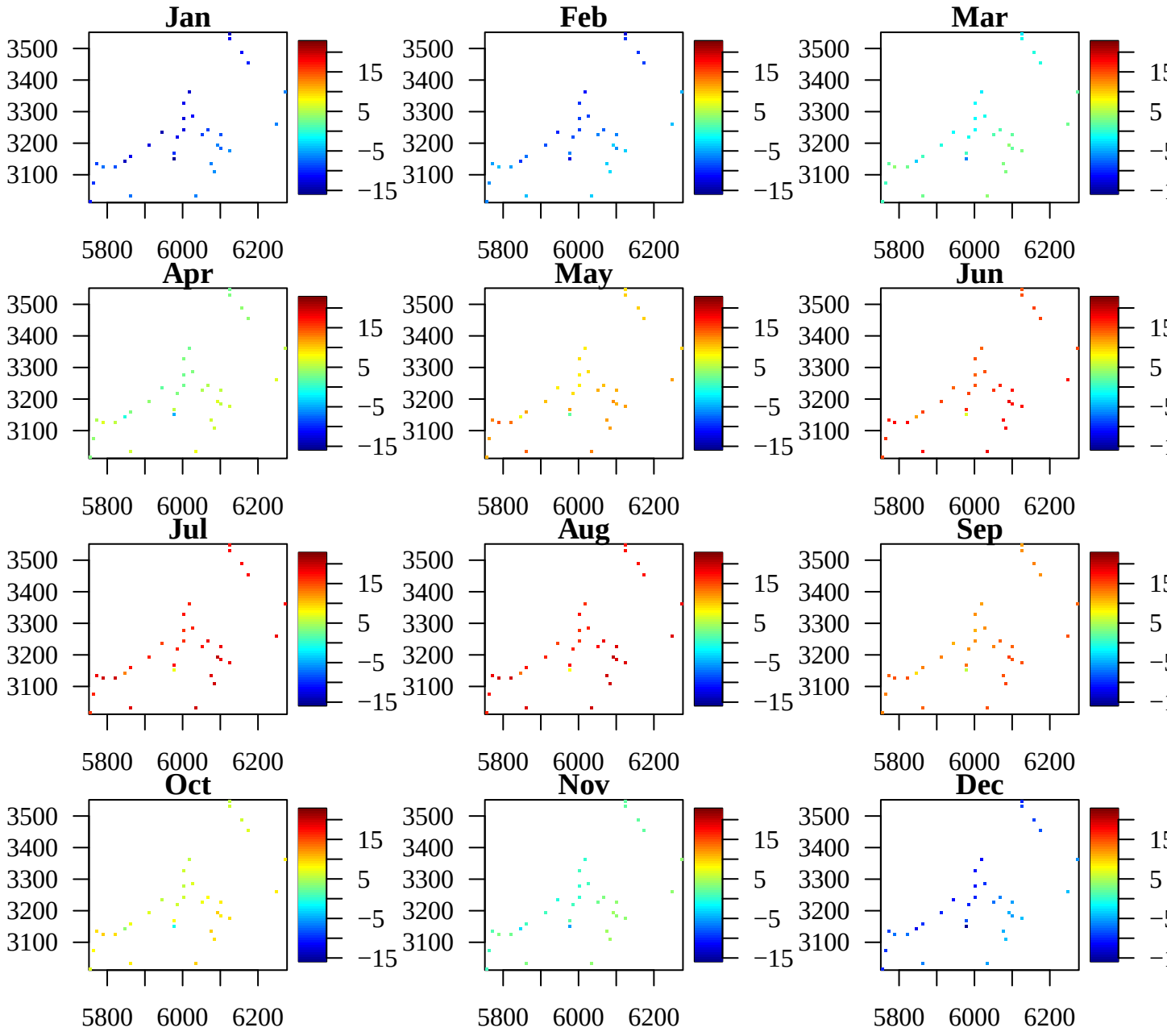
```
par(las = 1, mar =c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(1:129, unlist(dat[1, 4:132]), type = "l", lwd = 0.35, col = 1, las = 1,
     xaxt = "n", ylab = "Temperature (Celsius)", xlab = "Year",
     ylim = range(unlist(dat[, 4:132])))
axis(1, at = seq(6, 126, 12), labels = 2000:2010, xlab = "Year")
for (i in 2:34){
  lines(1:129, unlist(dat[i, 4:132]), lwd = 0.35, col = i)
}
```




**Space–time plots for the first two years**

For illustration, we restrict attention to the first two years (Jan 2000 – Dec 2001), which correspond to 24 monthly observations.

```
##subset first 2 years (Jan 2000 - Dec. 2001)
temp <- dat[, 4:27]
m <- ncol(temp) ##number of months
n <- nrow(temp) ##number of observation per months

par(las = 1, mar =c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
for (i in 1:12){
  quilt.plot(dat$UTMX / 1000, dat$UTMY / 1000, temp[, i], las = 1, zlim = range(temp))
  title(main = month.abb[rep(1:12, 2)][i])
}
```



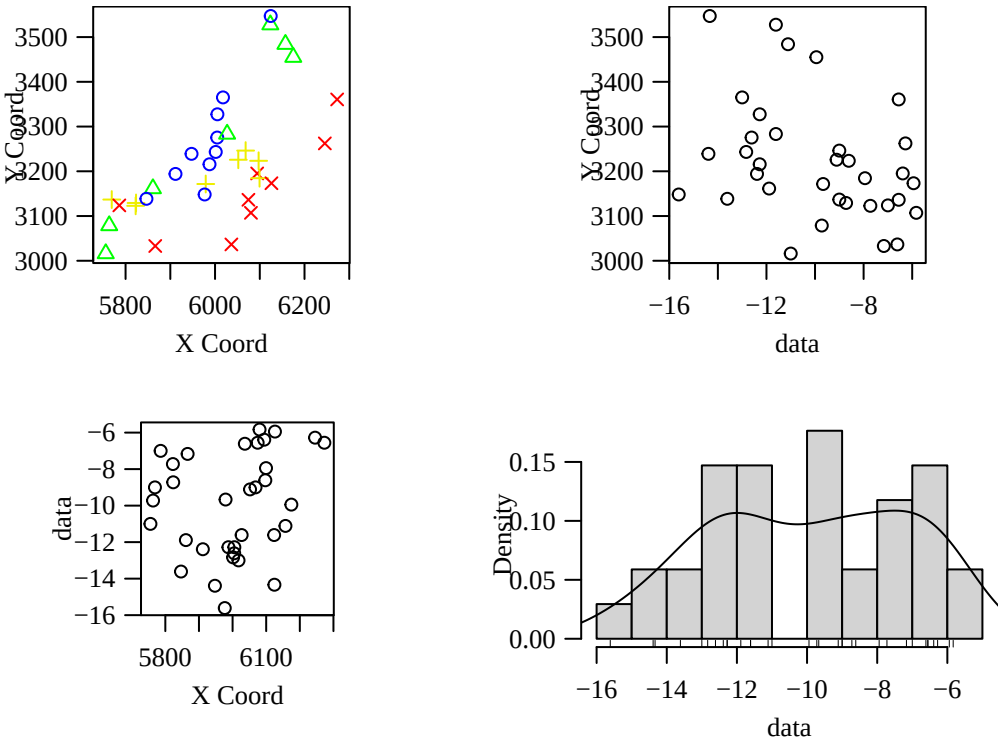

**Empirical variograms**

We now look at empirical variograms for each month in the first two years to explore spatial correlation.

```
library(geoR)
```

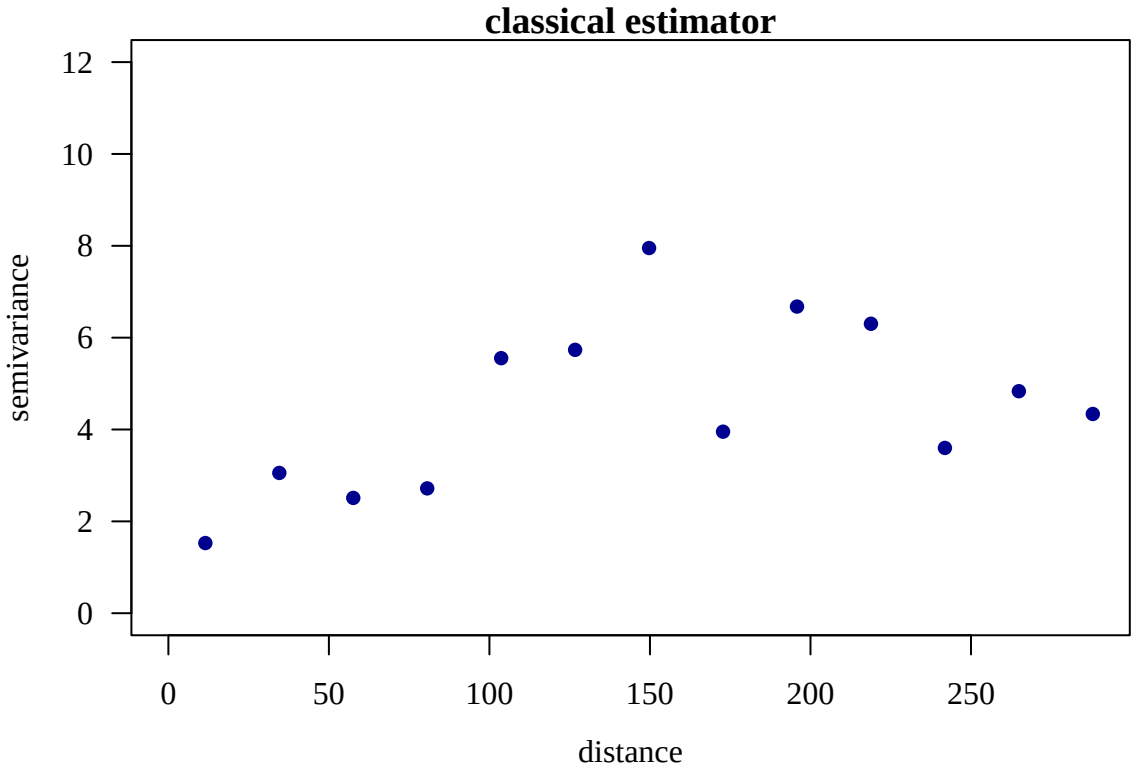
```
## ----------------------------------------------------------
##  Analysis of Geostatistical Data
##  For an Introduction to geoR go to http://www.leg.ufpr.br/geoR
##  geoR version 1.9-5 (built on 2025-04-25) is now loaded
## ----------------------------------------------------------
```

```
y.t.1 <- as.geodata(cbind(dat[, "UTMX"] / 1000, dat[, "UTMY"] / 1000, temp[, 1]))
par(las = 1, mar =c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(y.t.1)
```

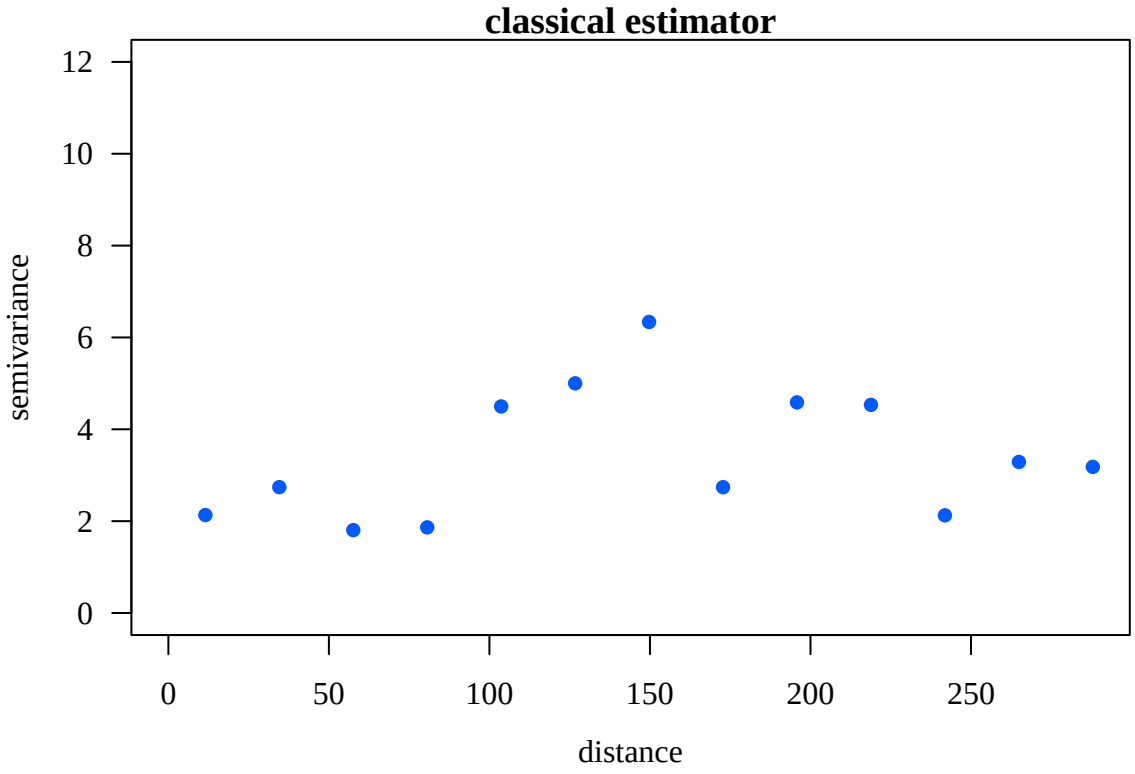



```
## Empirical Variograms

for (i in 1:12){
  y.t <- as.geodata(cbind(dat[, "UTMX"] / 1000, dat[, "UTMY"] / 1000, temp[, i]))
  cloud1 <- variog(y.t, trend = "1st", max.dist = 300)
  plot(cloud1, main = "classical estimator", las = 1, ylim = c(0, 12), pch = 16, col = col_mon[i])
}
```

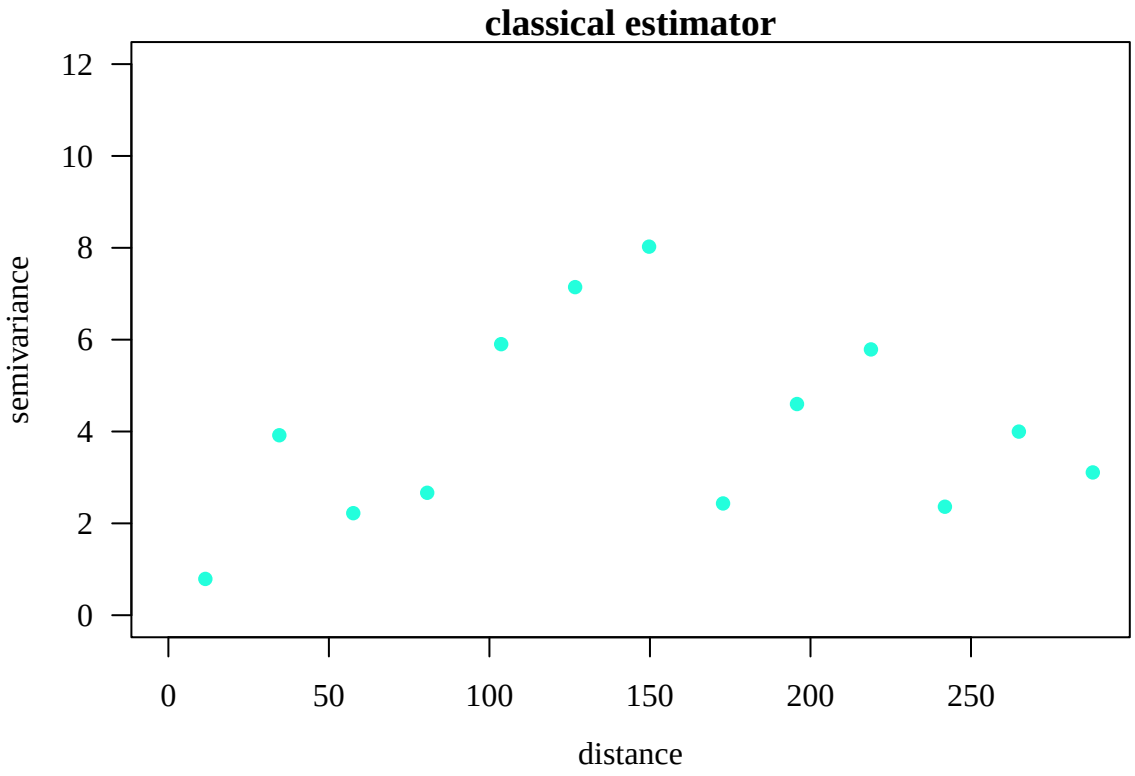
```
## variog: computing omnidirectional variogram
```



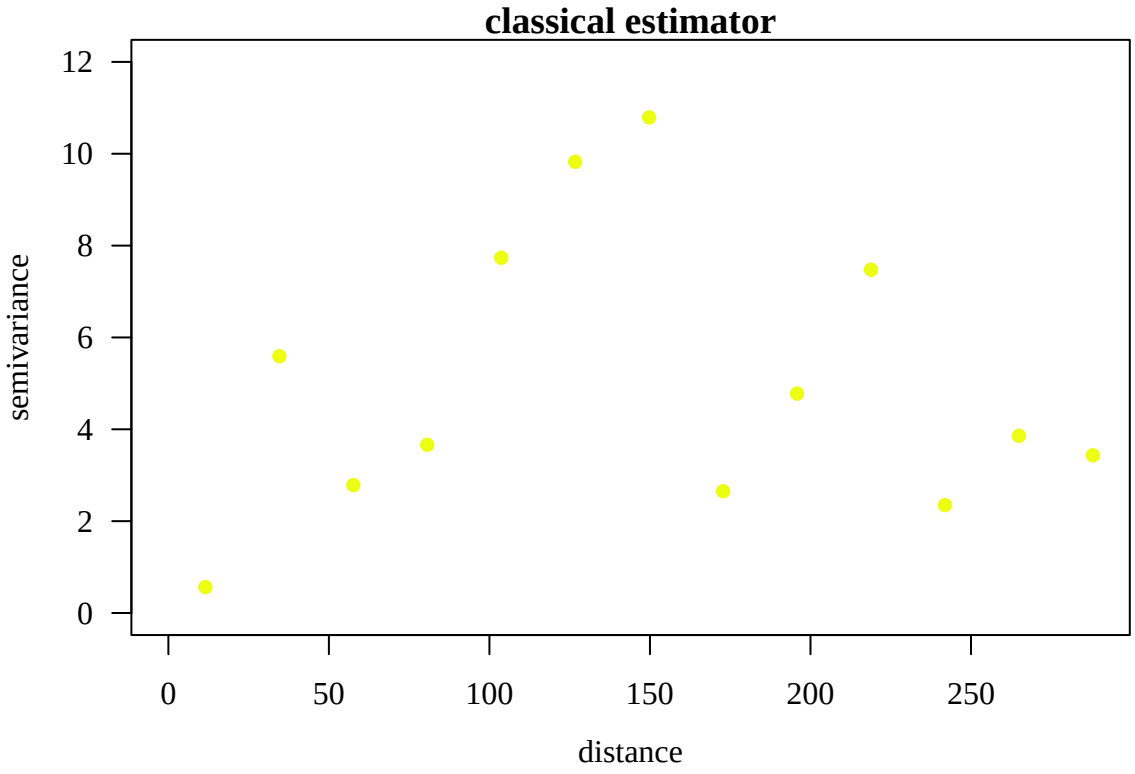

```
## variog: computing omnidirectional variogram
```



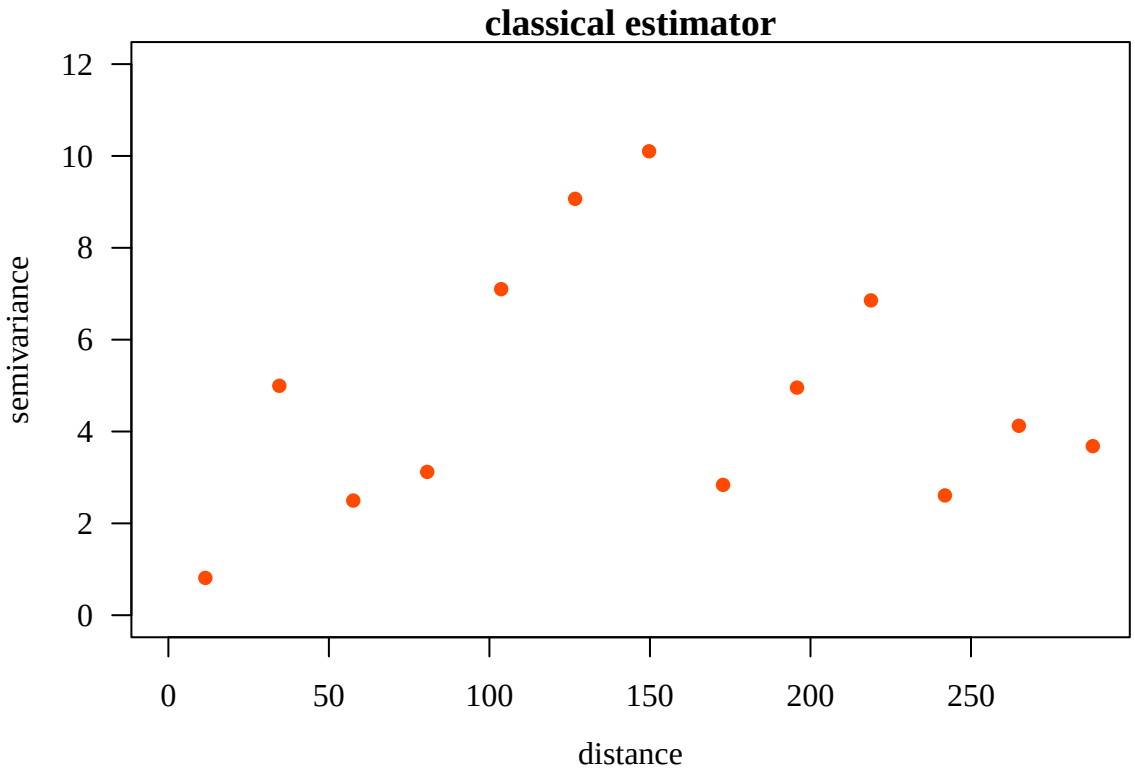

```
## variog: computing omnidirectional variogram
```



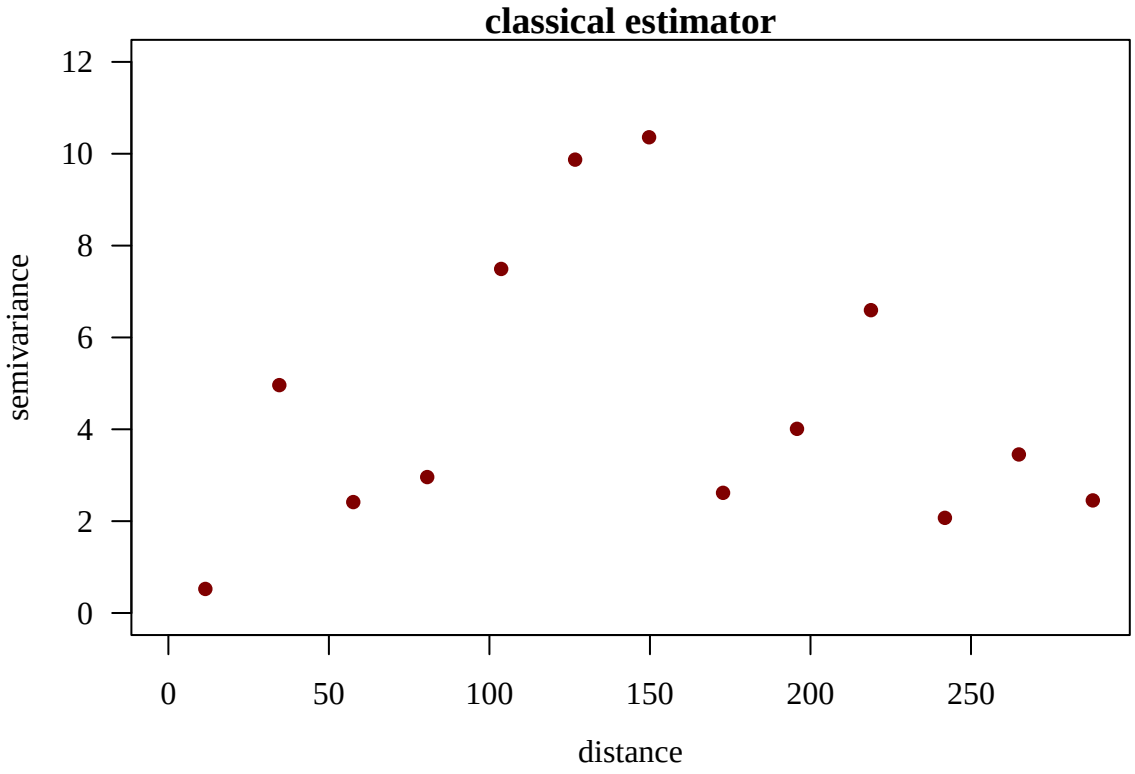

```
## variog: computing omnidirectional variogram
```



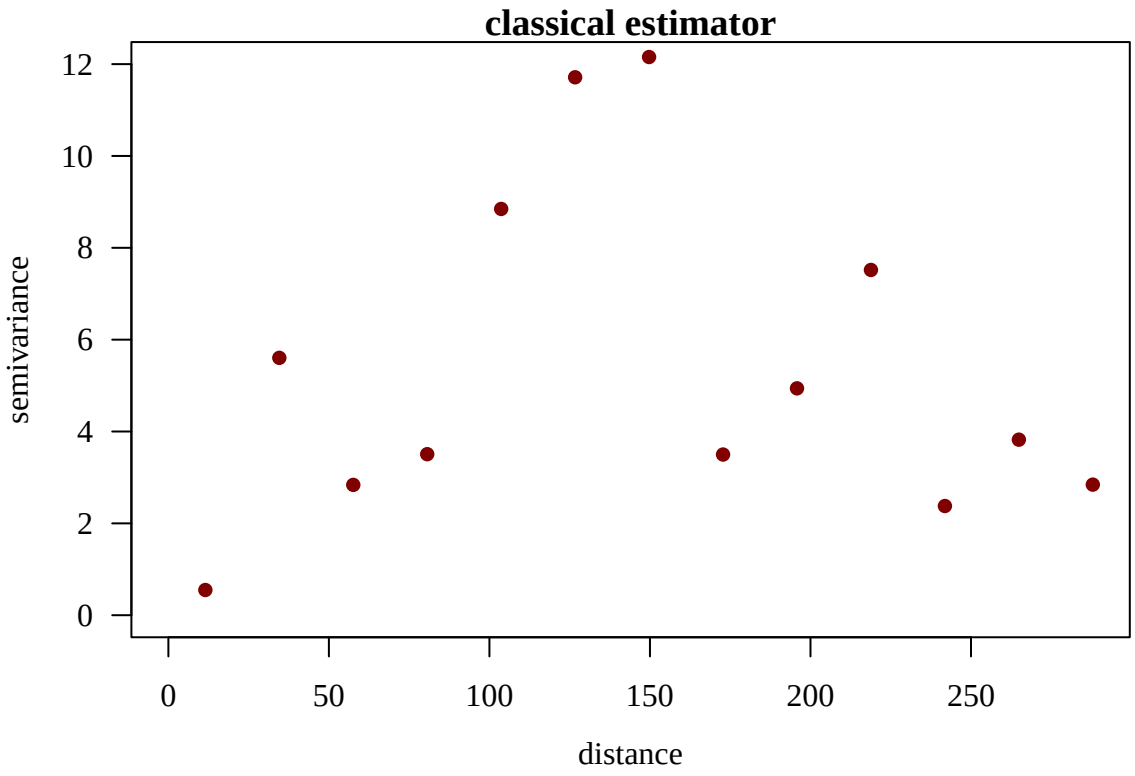

```
## variog: computing omnidirectional variogram
```



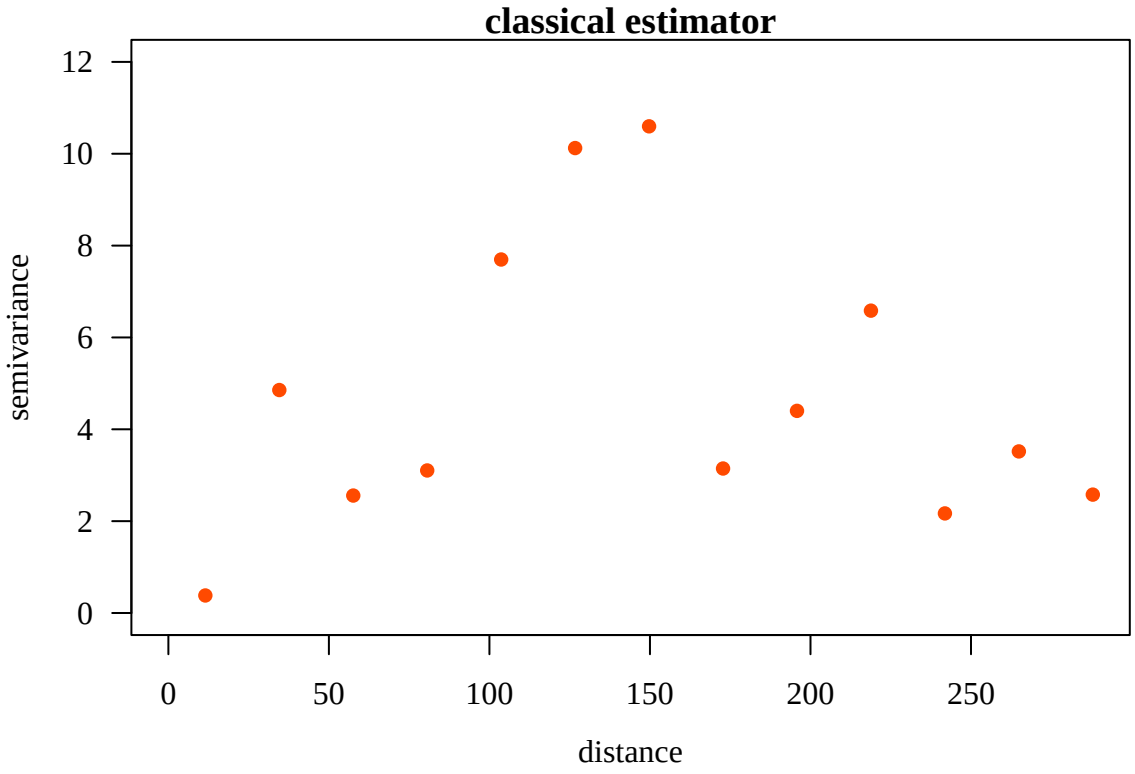

```
## variog: computing omnidirectional variogram
```



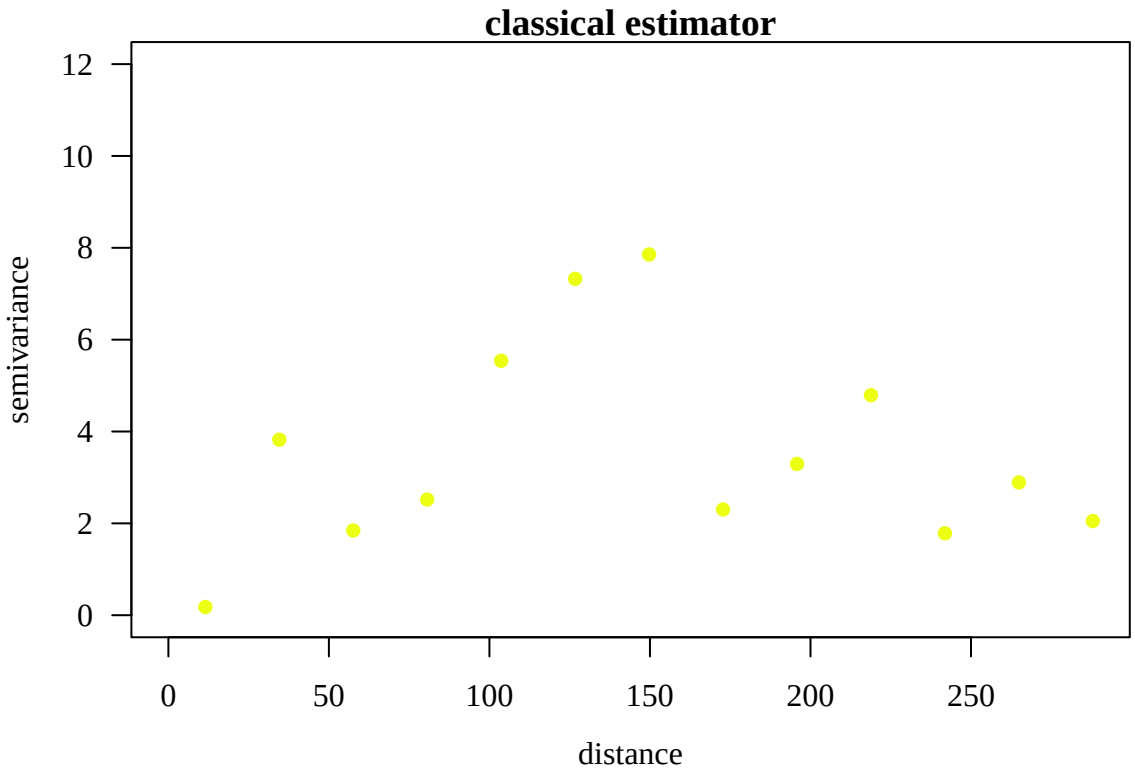

```
## variog: computing omnidirectional variogram
```



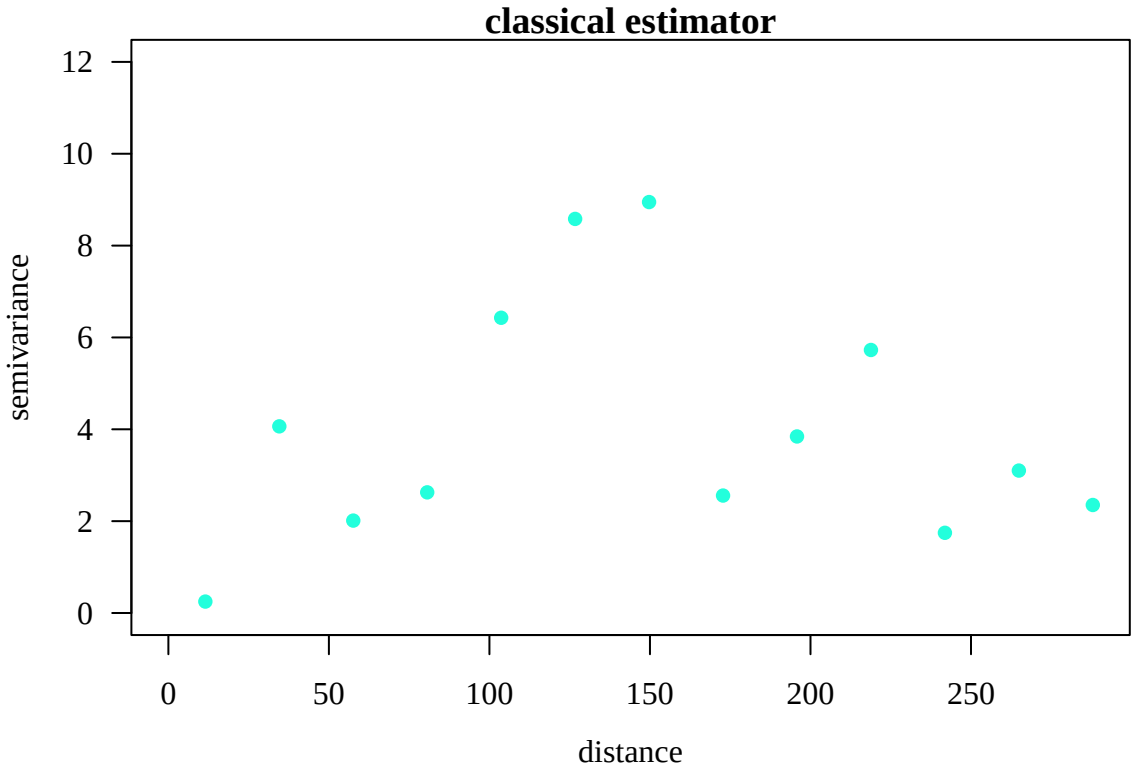

```
## variog: computing omnidirectional variogram
```



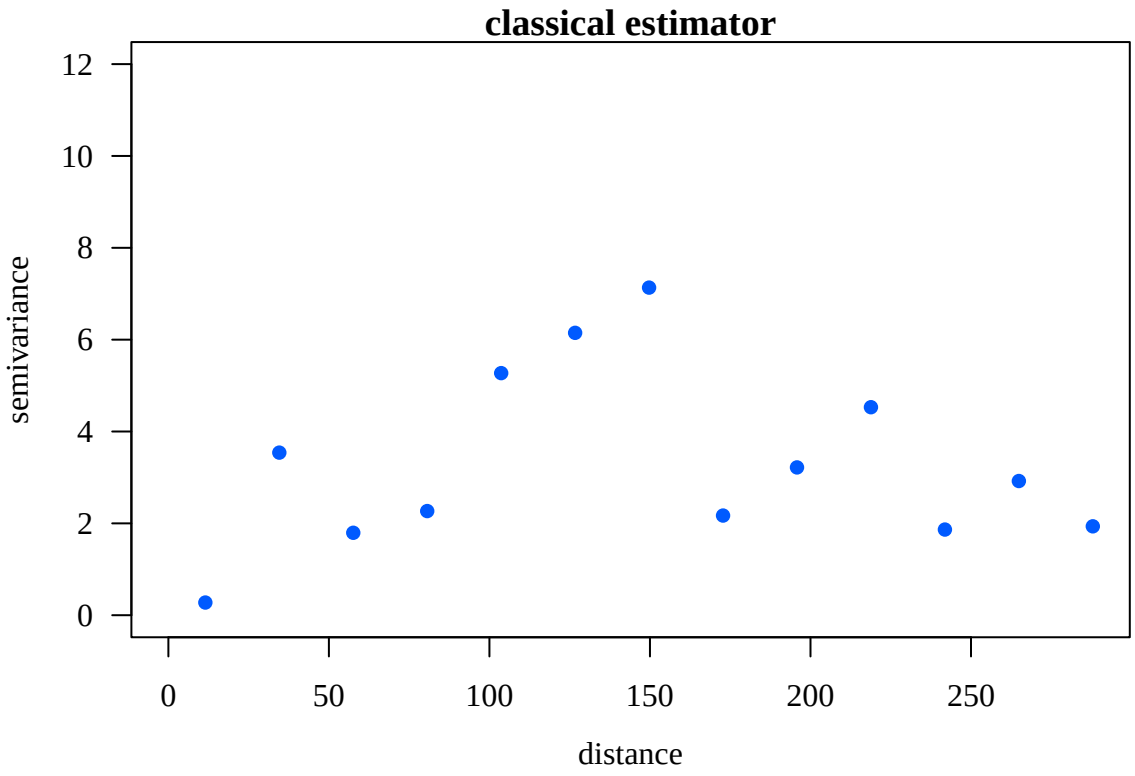

```
## variog: computing omnidirectional variogram
```



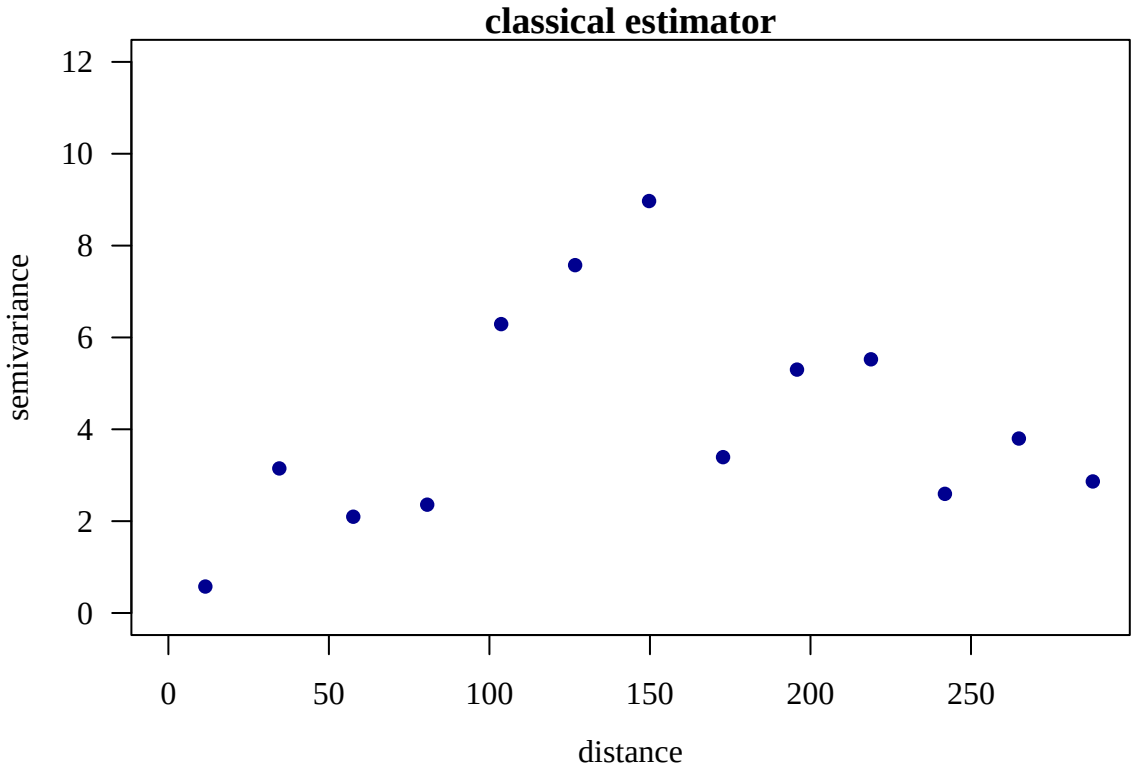

```
## variog: computing omnidirectional variogram
```




```
## variog: computing omnidirectional variogram
```




```
## variog: computing omnidirectional variogram
```



## Prediction setup (hold-out design)

Before fitting the dynamic spatio-temporal model, we set up:

1. Spatial coordinates (in km),
2. A small set of held-out observations for prediction assessment.

```
##scale to km
coords <- as.matrix(dat[, c("UTMX", "UTMY")] / 1000)
max.d <- max(iDist(coords))

miss <- sample(1:m, 10)
holdout.station.id <- 5
y.t.holdout <- temp[holdout.station.id, miss]
temp[holdout.station.id, miss] <- NA
```

### Priors and starting values for the dynamic model

We fit a dynamic linear model with monthly-varying regression parameters and spatial covariance parameters.

Let:

- $p$ = number of regression parameters per month (intercept + elevation),
- $m$ = number of months (here 24).

```
p <- 2 #number of regression parameters in each month
starting <- list(
  "beta" = rep(0, m * p),
  "phi" = rep(3 / (0.5 * max.d), m),
  "sigma.sq" = rep(2, m),
  "tau.sq" = rep(1, m),
  "sigma.eta" = diag(rep(0.01, p)))
tuning <- list("phi" = rep(5, m))
priors <- list(
  "beta.0.Norm" = list(rep(0, p), diag(1000, p)),
  "phi.Unif" = list(rep(3 / (0.9 * max.d), m), rep(3 / (0.05 * max.d), m)),
  "sigma.sq.IG" = list(rep(2, m), rep(10, m)),
  "tau.sq.IG" = list(rep(2, m), rep(5, m)),
  "sigma.eta.IW" = list(2, diag(0.001, p)))
```

### Model fitting with `spDynLM`

We specify a separate regression formula for each month, with temperature as the response and elevation as the covariate.

```
mods <- lapply(paste(colnames(temp), 'elev', sep = '~'), as.formula)
mods
```

```
## [[1]]
## y.1 ~ elev
## <environment: 0x12acf22c0>
##
## [[2]]
## y.2 ~ elev
## <environment: 0x12acf22c0>
##
## [[3]]
## y.3 ~ elev
## <environment: 0x12acf22c0>
##
## [[4]]
## y.4 ~ elev
## <environment: 0x12acf22c0>
##
## [[5]]
## y.5 ~ elev
## <environment: 0x12acf22c0>
##
## [[6]]
## y.6 ~ elev
## <environment: 0x12acf22c0>
##
## [[7]]
## y.7 ~ elev
## <environment: 0x12acf22c0>
##
## [[8]]
## y.8 ~ elev
## <environment: 0x12acf22c0>
##
## [[9]]
## y.9 ~ elev
## <environment: 0x12acf22c0>
##
## [[10]]
## y.10 ~ elev
## <environment: 0x12acf22c0>
##
## [[11]]
## y.11 ~ elev
## <environment: 0x12acf22c0>
##
## [[12]]
## y.12 ~ elev
## <environment: 0x12acf22c0>
##
## [[13]]
## y.13 ~ elev
## <environment: 0x12acf22c0>
##
## [[14]]
## y.14 ~ elev
```

```
## <environment: 0x12acf22c0>
##
## [[15]]
## y.15 ~ elev
## <environment: 0x12acf22c0>
##
## [[16]]
## y.16 ~ elev
## <environment: 0x12acf22c0>
##
## [[17]]
## y.17 ~ elev
## <environment: 0x12acf22c0>
##
## [[18]]
## y.18 ~ elev
## <environment: 0x12acf22c0>
##
## [[19]]
## y.19 ~ elev
## <environment: 0x12acf22c0>
##
## [[20]]
## y.20 ~ elev
## <environment: 0x12acf22c0>
##
## [[21]]
## y.21 ~ elev
## <environment: 0x12acf22c0>
##
## [[22]]
## y.22 ~ elev
## <environment: 0x12acf22c0>
##
## [[23]]
## y.23 ~ elev
## <environment: 0x12acf22c0>
##
## [[24]]
## y.24 ~ elev
## <environment: 0x12acf22c0>
```

```
n.samples <- 5000
m.1 <- spDynLM(mods, data = cbind(temp, dat[, "elev", drop = FALSE]), coords = coords,
starting = starting, tuning = tuning, priors = priors, get.fitted = TRUE,
cov.model = "exponential", n.samples = n.samples, n.report = 500)
```

```
## ----------------------------------------
##  General model description
## ----------------------------------------
## Model fit with 34 observations in 24 time steps.
##
## Number of missing observations 10.
##
```

```
## Number of covariates 2 (including intercept if specified).
##
## Using the exponential spatial correlation model.
##
## Number of MCMC samples 5000.
##
## Priors and hyperpriors:
##  beta normal:
##  m_0:    0.000   0.000
##  Sigma_0:
##  1000.000    0.000
##  0.000   1000.000
##
##  sigma.sq_t=1 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=1 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=1 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=2 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=2 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=2 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=3 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=3 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=3 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=4 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=4 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=4 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=5 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=5 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=5 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=6 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=6 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=6 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=7 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=7 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=7 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=8 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=8 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=8 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=9 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=9 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=9 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=10 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=10 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=10 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=11 IG hyperpriors shape=2.00000 and scale=10.00000
```

```
##  tau.sq_t=11 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=11 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=12 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=12 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=12 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=13 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=13 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=13 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=14 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=14 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=14 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=15 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=15 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=15 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=16 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=16 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=16 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=17 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=17 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=17 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=18 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=18 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=18 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=19 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=19 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=19 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=20 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=20 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=20 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=21 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=21 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=21 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=22 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=22 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=22 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=23 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=23 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=23 Unif hyperpriors a=0.00516 and b=0.09281
##  ---
##  sigma.sq_t=24 IG hyperpriors shape=2.00000 and scale=10.00000
##  tau.sq_t=24 IG hyperpriors shape=2.00000 and scale=5.00000
##  phi_t=24 Unif hyperpriors a=0.00516 and b=0.09281
```

```
##  ---
## ------------------------------------------------
##      Sampling
## ------------------------------------------------
## Sampled: 499 of 5000, 9.98%
## Report interval Mean Metrop. Acceptance rate: 25.81%
## Overall Metrop. Acceptance rate: 25.86%
## ------------------------------------------------
## Sampled: 999 of 5000, 19.98%
## Report interval Mean Metrop. Acceptance rate: 25.39%
## Overall Metrop. Acceptance rate: 25.63%
## ------------------------------------------------
## Sampled: 1499 of 5000, 29.98%
## Report interval Mean Metrop. Acceptance rate: 26.87%
## Overall Metrop. Acceptance rate: 26.04%
## ------------------------------------------------
## Sampled: 1999 of 5000, 39.98%
## Report interval Mean Metrop. Acceptance rate: 26.30%
## Overall Metrop. Acceptance rate: 26.10%
## ------------------------------------------------
## Sampled: 2499 of 5000, 49.98%
## Report interval Mean Metrop. Acceptance rate: 25.48%
## Overall Metrop. Acceptance rate: 25.98%
## ------------------------------------------------
## Sampled: 2999 of 5000, 59.98%
## Report interval Mean Metrop. Acceptance rate: 26.89%
## Overall Metrop. Acceptance rate: 26.13%
## ------------------------------------------------
## Sampled: 3499 of 5000, 69.98%
## Report interval Mean Metrop. Acceptance rate: 26.24%
## Overall Metrop. Acceptance rate: 26.15%
## ------------------------------------------------
## Sampled: 3999 of 5000, 79.98%
## Report interval Mean Metrop. Acceptance rate: 26.32%
## Overall Metrop. Acceptance rate: 26.17%
## ------------------------------------------------
## Sampled: 4499 of 5000, 89.98%
## Report interval Mean Metrop. Acceptance rate: 26.23%
## Overall Metrop. Acceptance rate: 26.18%
## ------------------------------------------------
## Sampled: 4999 of 5000, 99.98%
## Report interval Mean Metrop. Acceptance rate: 26.02%
## Overall Metrop. Acceptance rate: 26.16%
## ------------------------------------------------
```

```
burn.in <- floor(0.75 * n.samples)
```

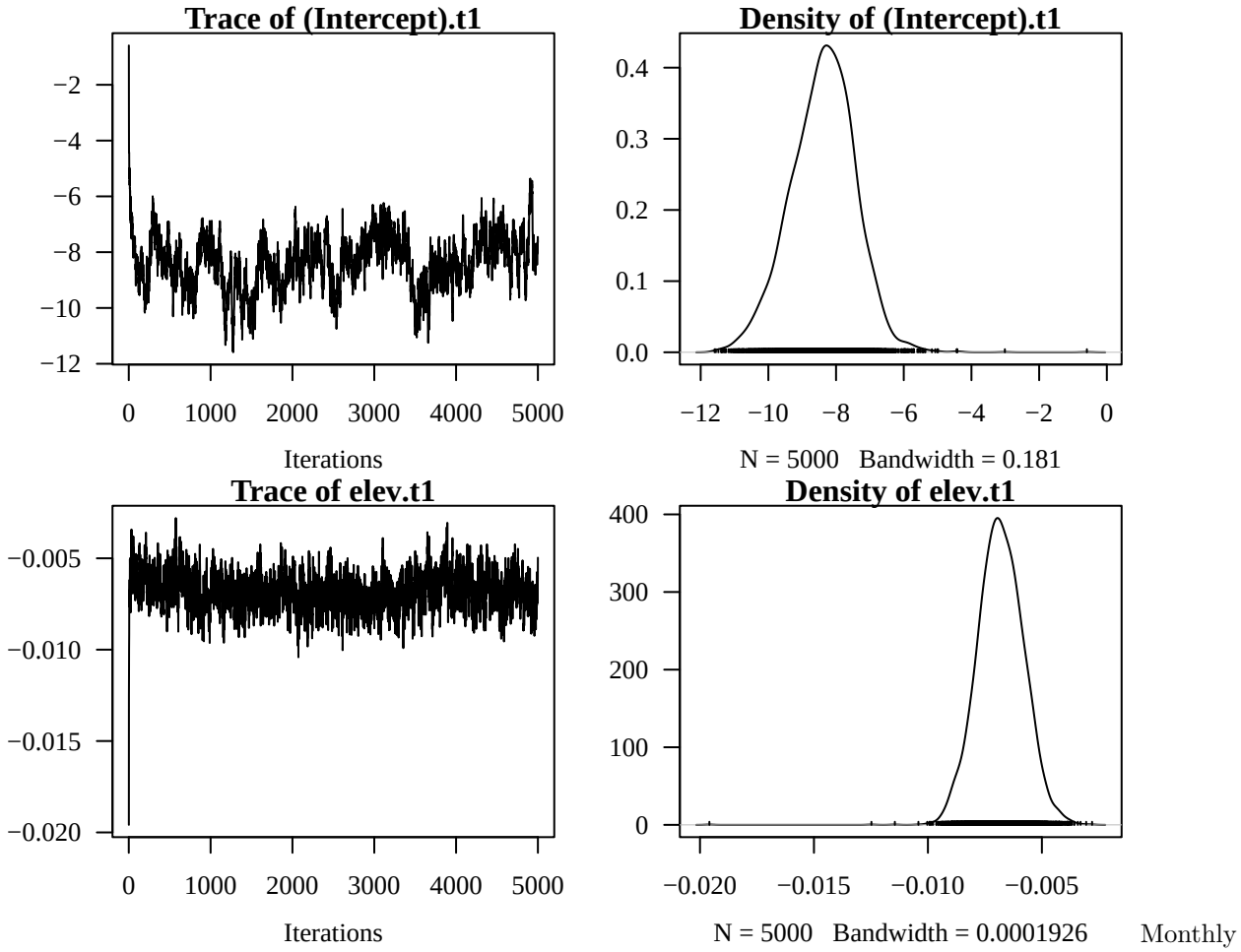
### Posterior summaries for regression parameters

We summarize the posterior distributions of the monthly regression coefficients:

- $\beta_{0,t}$: intercept (baseline temperature),

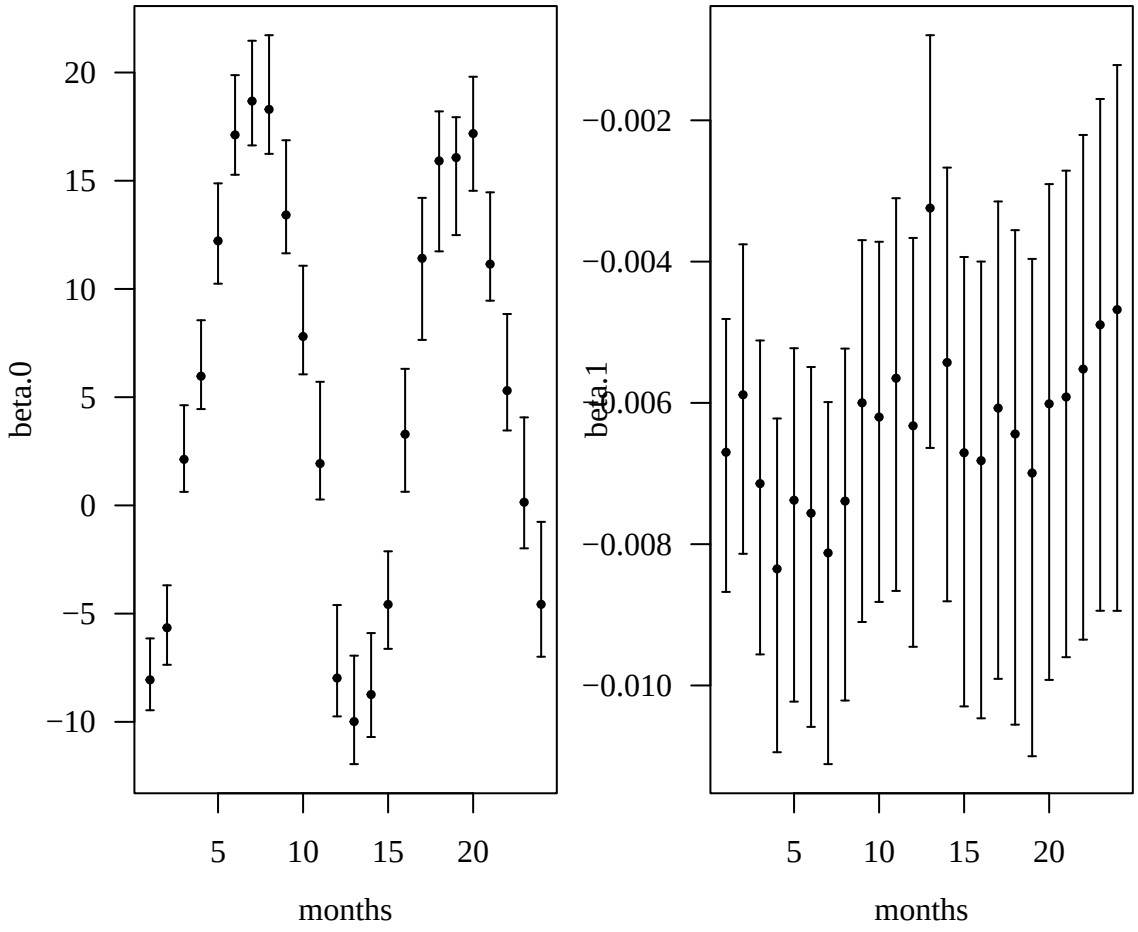
- $\beta_{1,t}$: effect of elevation.

```
quant <- function(x) {quantile(x, prob = c(0.5, 0.025, 0.975))}
beta <- apply(m.1$p.beta.samples[burn.in:n.samples,], 2, quant)
beta.0 <- beta[, grep("Intercept", colnames(beta))]
beta.1 <- beta[, grep("elev", colnames(beta))]

par(las = 1, mar =c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(m.1$p.beta.samples[, 1:2])
```




Monthly credible intervals

```
par(mfrow = c(1, 2), las = 1, mar =c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(1:m, beta.0[1,], pch = 19, cex = 0.5, xlab = "months", ylab = "beta.0",
     ylim = range(beta.0))
arrows(1:m, beta.0[1,], 1:m, beta.0[3,], length = 0.02, angle = 90)
arrows(1:m, beta.0[1,], 1:m, beta.0[2,], length = 0.02, angle = 90)
plot(1:m, beta.1[1,], pch=19, cex = 0.5, xlab = "months", ylab = "beta.1", ylim = range(beta.1))
arrows(1:m, beta.1[1,], 1:m, beta.1[3,], length = 0.02, angle = 90)
arrows(1:m, beta.1[1,], 1:m, beta.1[2,], length = 0.02, angle = 90)
```

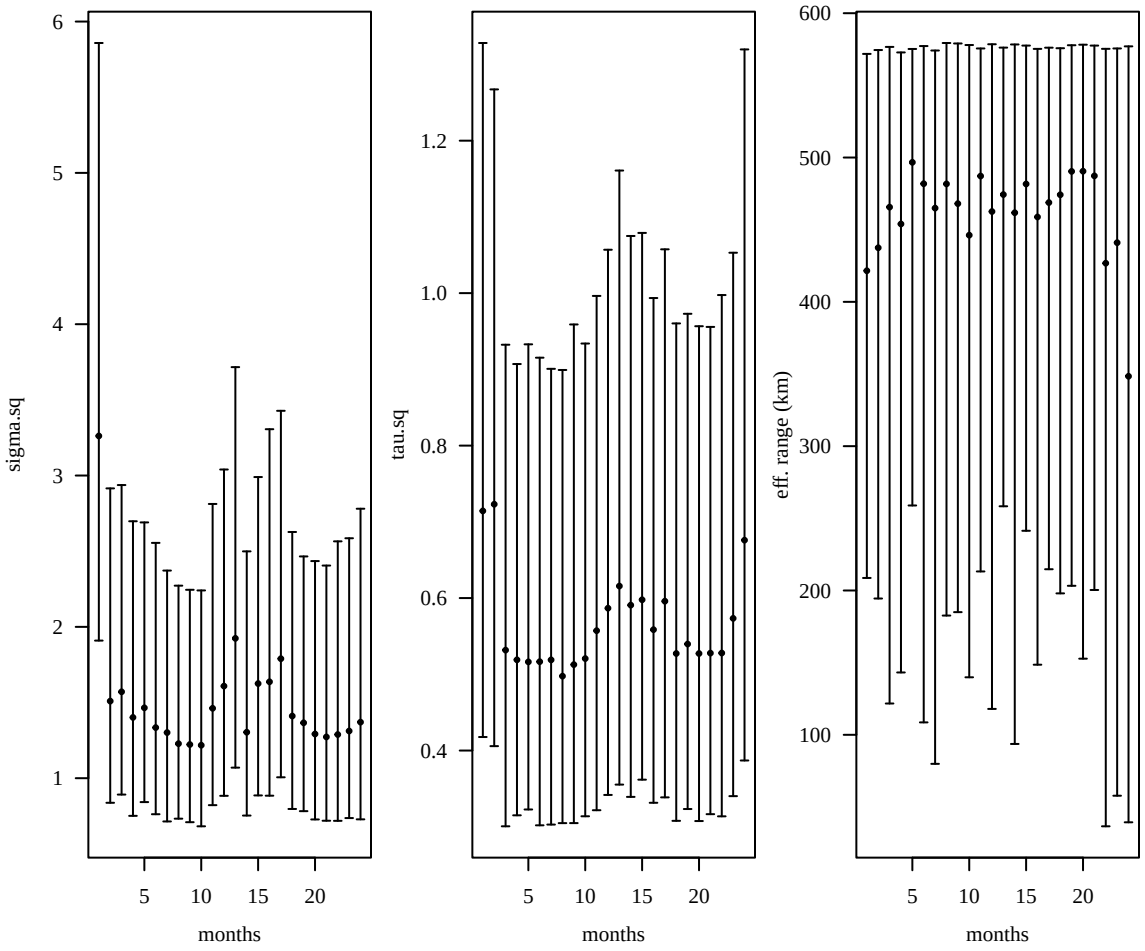



**Posterior summaries for covariance parameters**

We now summarize the spatial variance ($\sigma^2$), nugget ($\tau^2$), and spatial range (expressed as an effective range, ($\frac{3}{\phi}$).

```
theta <- apply(m.1$p.theta.samples[burn.in:n.samples,], 2, quant)
sigma.sq <- theta[, grep("sigma.sq", colnames(theta))]
tau.sq <- theta[, grep("tau.sq", colnames(theta))]
phi <- theta[, grep("phi", colnames(theta))]
```

```
par(mfrow = c(1, 3), las = 1, mar =c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(1:m, sigma.sq[1,], pch = 19, cex = 0.5, xlab = "months", ylab = "sigma.sq", ylim = range(sigma.sq)
arrows(1:m, sigma.sq[1,], 1:m, sigma.sq[3,], length = 0.02, angle = 90)
arrows(1:m, sigma.sq[1,], 1:m, sigma.sq[2,], length = 0.02, angle = 90)

plot(1:m, tau.sq[1,], pch = 19, cex = 0.5, xlab = "months", ylab = "tau.sq", ylim = range(tau.sq))
arrows(1:m, tau.sq[1,], 1:m, tau.sq[3,], length = 0.02, angle = 90)
arrows(1:m, tau.sq[1,], 1:m, tau.sq[2,], length = 0.02, angle = 90)

plot(1:m, 3 / phi[1,], pch = 19, cex = 0.5, xlab = "months", ylab = "eff. range (km)", ylim = range(3 /
arrows(1:m, 3 / phi[1,], 1:m, 3 / phi[3,], length = 0.02, angle = 90)
arrows(1:m, 3 / phi[1,], 1:m, 3 / phi[2,], length = 0.02, angle = 90)
```



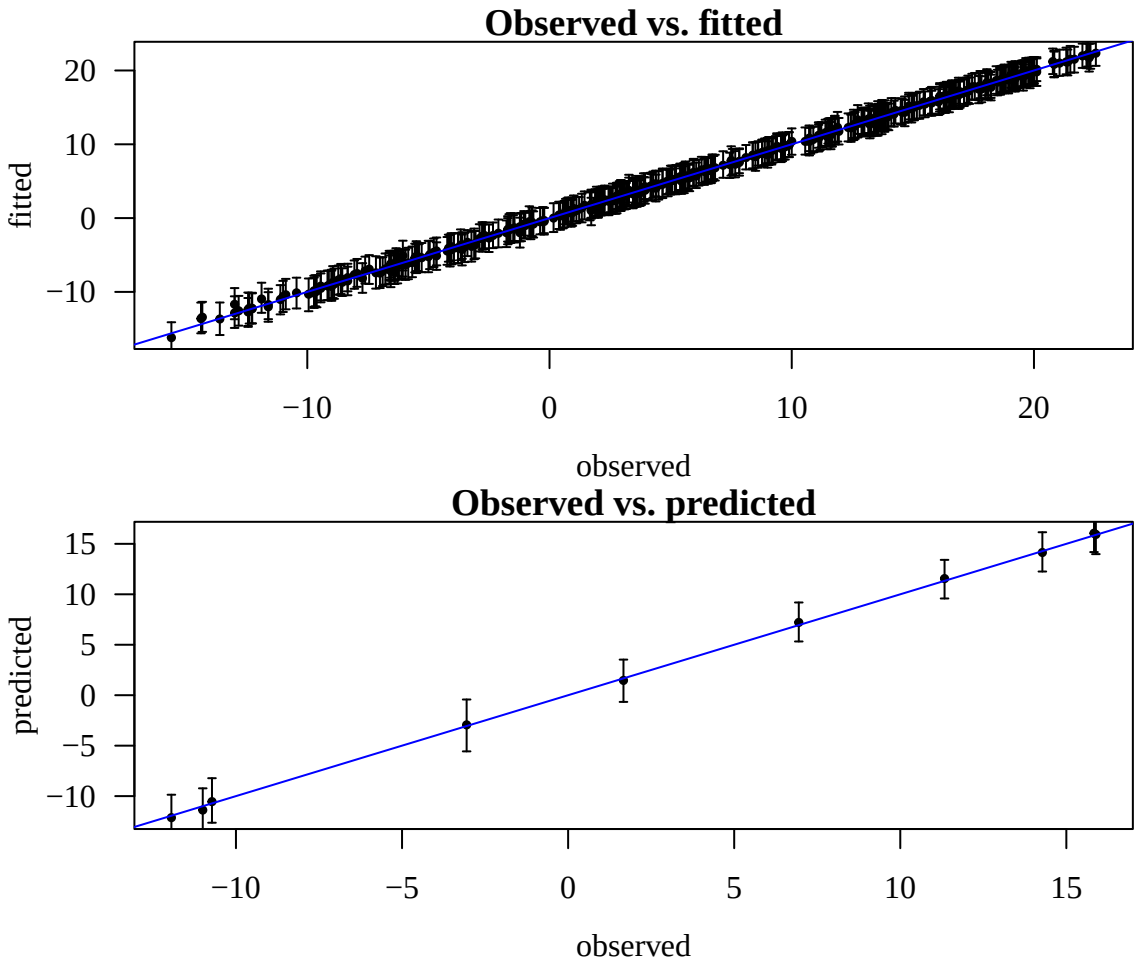

### Fitted values and predictive performance

Finally, we examine how well the model predicts both:

- In-sample (non-held-out) observations, and
- Out-of-sample (held-out station/months).

```
y.hat <- apply(m.1$p.y.samples[,burn.in:n.samples], 1, quant)
y.hat.med <- matrix(y.hat[1,], ncol = m)
y.hat.up <- matrix(y.hat[3,], ncol = m)
y.hat.low <- matrix(y.hat[2,], ncol = m)

y.obs <- as.vector(as.matrix(temp[-holdout.station.id, -miss]))
y.obs.hat.med <- as.vector(y.hat.med[-holdout.station.id, -miss])
y.obs.hat.up <- as.vector(y.hat.up[-holdout.station.id, -miss])
y.obs.hat.low <- as.vector(y.hat.low[-holdout.station.id, -miss])

y.ho <- as.matrix(y.t.holdout)
y.ho.hat.med <- as.vector(y.hat.med[holdout.station.id, miss])
y.ho.hat.up <- as.vector(y.hat.up[holdout.station.id, miss])
y.ho.hat.low <- as.vector(y.hat.low[holdout.station.id, miss])
```

```
par(mfrow = c(2, 1), las = 1, mar =c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(y.obs, y.obs.hat.med, pch = 19, cex = 0.5, xlab = "observed",
ylab = "fitted", main = "Observed vs. fitted")
arrows(y.obs, y.obs.hat.med, y.obs, y.obs.hat.up, length = 0.02, angle = 90)
arrows(y.obs, y.obs.hat.med, y.obs, y.obs.hat.low, length = 0.02, angle = 90)
lines(-50:50, -50:50, col = "blue")

plot(y.ho, y.ho.hat.med, pch = 19, cex = 0.5, xlab = "observed",
ylab = "predicted", main = "Observed vs. predicted")
arrows(y.ho, y.ho.hat.med, y.ho, y.ho.hat.up, length = 0.02, angle = 90)
arrows(y.ho, y.ho.hat.med, y.ho, y.ho.hat.low, length = 0.02, angle = 90)
lines(-50:50, -50:50, col="blue")
```




In this example, we:

1. Performed exploratory analyses to understand spatial and temporal patterns of monthly temperatures and their relationship with elevation.
2. Examined empirical variograms, revealing spatial dependence that varies by month.
3. Fitted a dynamic spatio-temporal model using `spDynLM`, allowing regression coefficients and covariance parameters to evolve over time.
4. Summarized posterior distributions of key parameters, including monthly intercepts, elevation effects, and spatial covariance parameters.

5. Evaluated the model's in-sample fit and out-of-sample predictions using held-out data.

This workflow demonstrates how dynamic space-time models can be used to capture evolving spatial structure in environmental data.

## Empirical Orthogonal Function (EOF) Analysis of ENSO

Here we illustrate **Empirical Orthogonal Function (EOF) analysis** (also known as Principal Component Analysis) using monthly sea-surface temperature (SST) anomalies over the tropical Pacific.

We will:

1. Load and visualize gridded SST data.
2. Compute monthly SST anomalies by removing the seasonal cycle.
3. Extract the leading EOFs and their associated principal components (PCs).
4. Relate the leading PC to the Southern Oscillation Index (SOI).
5. Reconstruct the January 1998 El Niño event using a few leading EOFs.
6. Explore SST trends and covariance patterns.

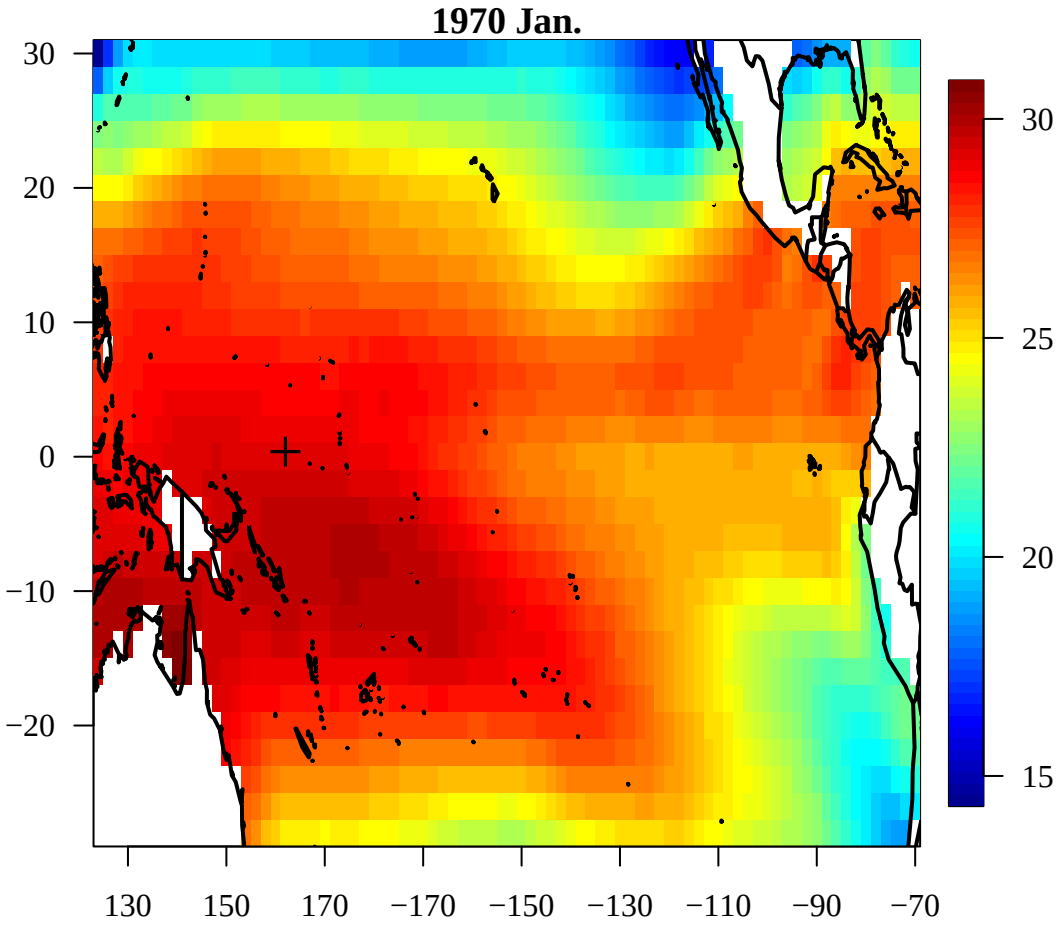
### Data and basic visualization

Load data

```
load("SST1.rda")
```

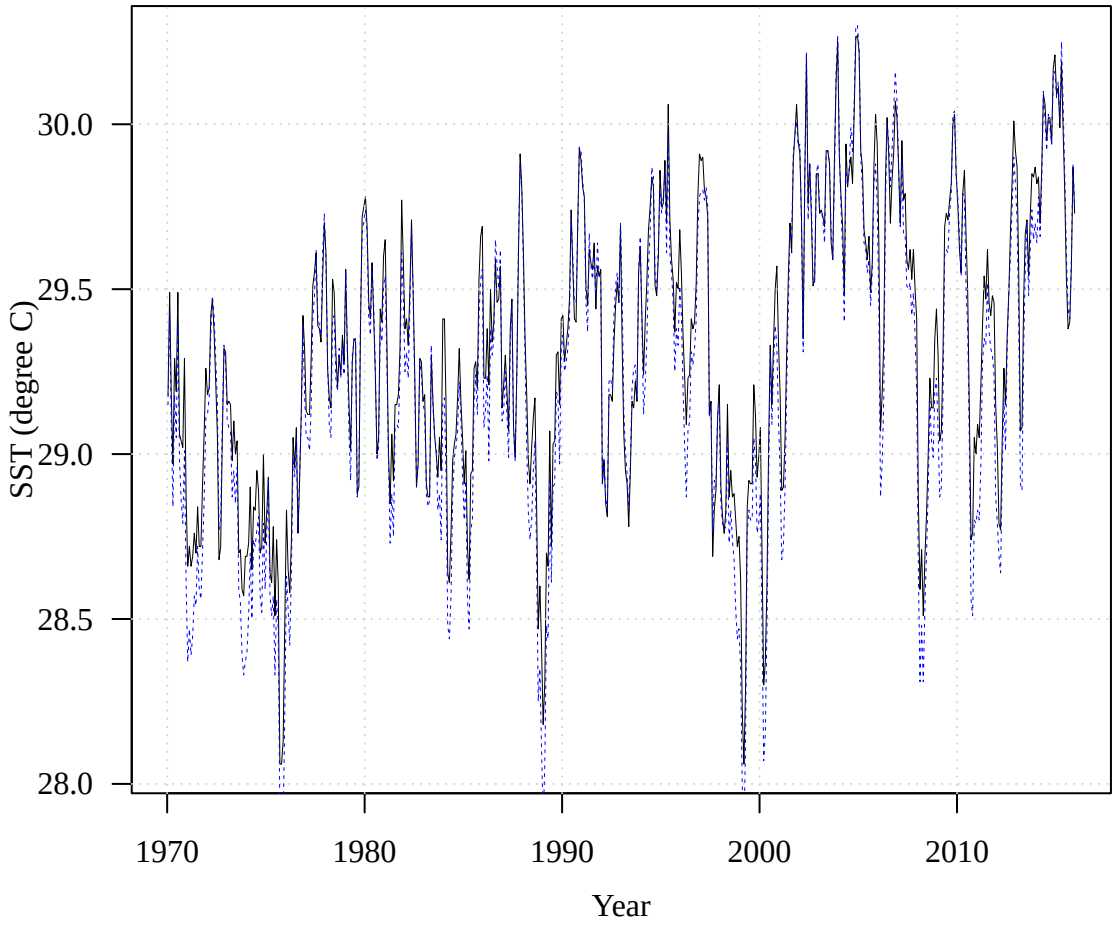
We first look at the SST field for a single month (1970 Jan.) to understand the spatial grid.

```
par(las = 1, mar = c(3, 3, 1, 1), mgp = c(2, 1, 0), family = "serif")
image.plot(lon1, lat1, SST1[,, 1], xaxt = "n", xlab = "", ylab = "",
           main = "1970 Jan.")
lon <- ifelse(lon1 <= 180, lon1, lon1 - 360)
axis(1, at = lon1[seq(4, 84, 10)], lon[seq(4, 84, 10)])
points(lon1[20], lat1[15], pch = "+", cex = 2)
map("world2", add = TRUE, lwd = 2)
```




Next we examine the SST time series at one equatorial grid location and compare it to a neighboring grid box.

```
par(las = 1, mar = c(3.5, 3.5, 1, 1), mgp = c(2.4, 1, 0), family = "serif")
plot(tm1, SST1[20, 15,], type = "l", las = 1, xlab = "Year",
     ylab = "SST (degree C)", lwd = 0.5)
grid()
# Add the time series at a nearby grid box
lines(tm1, SST1[21, 15,], col = "blue", lwd = 0.5, lty = 2)
```




### Animation of monthly SST

This section creates a PDF-based animation of monthly SST fields from 1970–2015. It is set to `eval=FALSE` because it can be slow and requires LaTeX.

```
SST_range <- range(SST1, na.rm = T)
mon <- rep(month.abb, 46)
yr <- rep(1970:2015, each = 12)
library(animation)
saveLatex({
for (i in 1:552){
  par(las = 1, mar = c(3, 3, 1, 1), family = "serif")
  image.plot(lon1, lat1, SST1[,, i], zlim = SST_range,
             xaxt = "n", xlab = "", ylab = "", main = paste(yr[i], mon[i]))
  axis(1, at = lon1[seq(4, 84, 10)], lon[seq(4, 84, 10)])
  map("world2", add = TRUE, lwd = 2)
 }
}, img.name = "SST_1970-2015", ani.opts = "controls,width=0.95\\textwidth",
latex.filename = ifelse(interactive(), "SST_monthly_1970-2015.tex", ""),
nmax = 552, ani.dev = "pdf", ani.type = "pdf", ani.width = 8,
ani.height = 3.6,documentclass = paste("\\documentclass{article}",
"\\usepackage[papersize={8in,3.6in},margin=0.15in]{geometry}",
                                       sep = "\n"))
```

### Computing SST anomalies

EOF analysis is typically done on anomalies: we remove the climatological mean seasonal cycle, so we are focusing on variability rather than the annual cycle.

We first reshape data to separate month and year. The data cube SST1 has dimensions (lon, lat, time). We reshape time into (month, year) and then remove the monthly mean at each location.

```
t <- array(SST1, dim = c(84, 30, 12, 46))
SST_temp <- apply(t, 1:3, function(x) x - mean(x, na.rm = T))
# Change the data into lon-lat-month format
SST_anomalies <- array(dim = c(84, 30, 552))
for (i in 1:84){
  for (j in 1:30){
    SST_anomalies[i, j,] <- c(t(SST_temp[, i, j,]))
  }
}
```

### EOF analysis via singular value decomposition (SVD)

We now perform EOF analysis on the anomaly fields. We will:

1. Reshape the 3D data into a 2D matrix (space $\times$ time).
2. Apply `SVD` to obtain EOFs (spatial patterns) and PCs (time series).
3. Visualize the leading EOFs.

```
# Extracting first three EOFs via singular value decomposition
temp <- array(SST_anomalies, c(84 * 30, 552))
ind <- is.na(temp[, 1])
temp <- temp[!ind, ]
temp2 <- svd(temp)
```

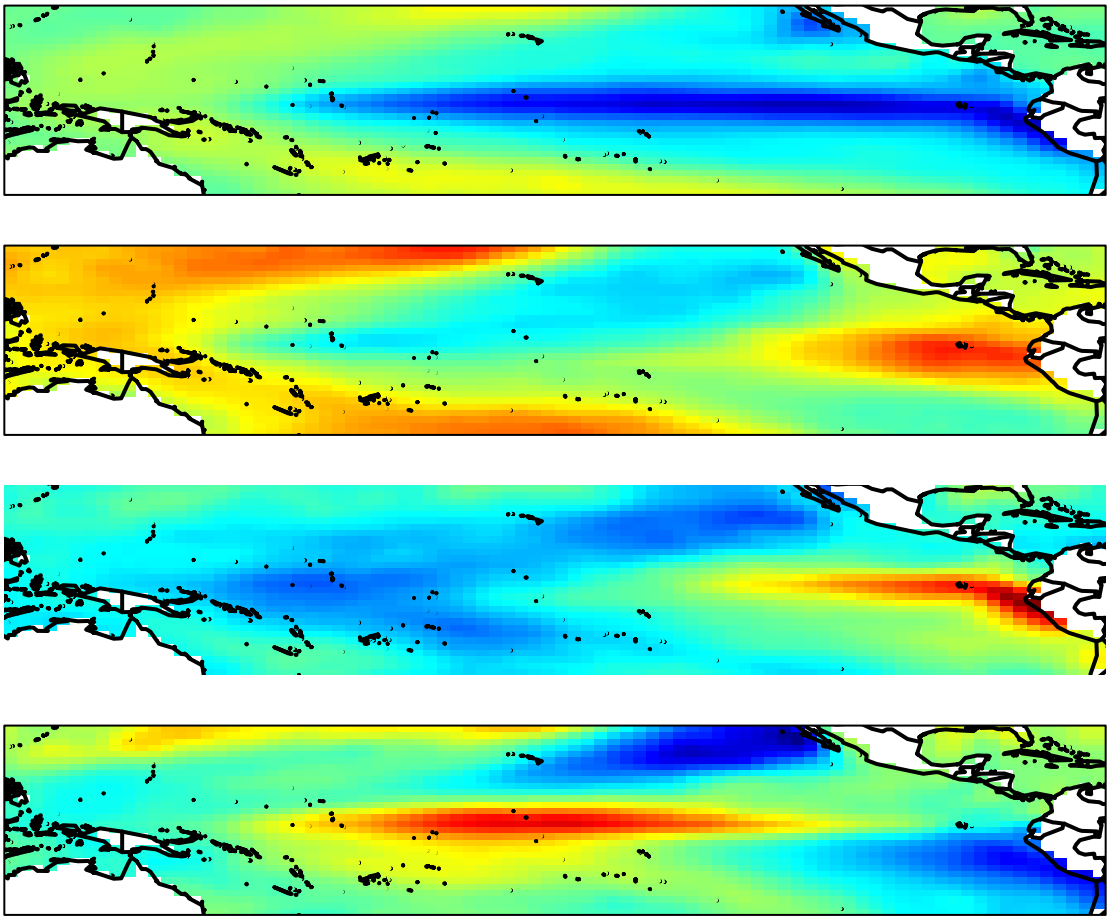
Each EOF is a spatial pattern over the grid.We store EOF 1–4 as $84 \times 30$ matrices.

```
U1 <- matrix(NA, 84 * 30); U1[!ind] <- temp2$u[, 1]; U1 <- matrix(U1, 84, 30)
U2 <- matrix(NA, 84 * 30); U2[!ind] <- temp2$u[, 2]; U2 <- matrix(U2, 84, 30)
U3 <- matrix(NA, 84 * 30); U3[!ind] <- temp2$u[, 3]; U3 <- matrix(U3, 84, 30)
U4 <- matrix(NA, 84 * 30); U4[!ind] <- temp2$u[, 4]; U4 <- matrix(U4, 84, 30)
zr <- range(c(U1, U2, U3, U4), na.rm = TRUE)

set.panel(4, 1)
```

```
## plot window will lay out plots in a 4 by 1 matrix
```

```
par(oma = rep(0, 4), las = 1, family = "serif", mar = rep(1, 4))
ct <- tim.colors(256)
par(mar = c(1, 1, 1, 1))
image(lon1, lat1, U1, axes = FALSE, xlab = "", ylab = "", zlim = zr, col = ct)
map("world2", add = TRUE, lwd = 2)
box()
```

```
image(lon1, lat1, U2, axes = FALSE, xlab = "", ylab = "", zlim = zr, col = ct)
map("world2", add = TRUE, lwd = 2)
box()
image(lon1, lat1, U3, axes = FALSE, xlab = "", ylab = "", zlim = zr, col = ct)
map("world2", add = TRUE, lwd = 2)
image(lon1, lat1, U4, axes = FALSE, xlab = "", ylab = "", zlim = zr, col = ct)
map("world2", add = TRUE, lwd = 2)
box()
```



### Principal component (PC) time series and ENSO

The principal components (PCs) are the temporal coefficients associated with each EOF. We compare the leading PC with the Southern Oscillation Index (SOI) to show that EOF 1 captures ENSO variability.

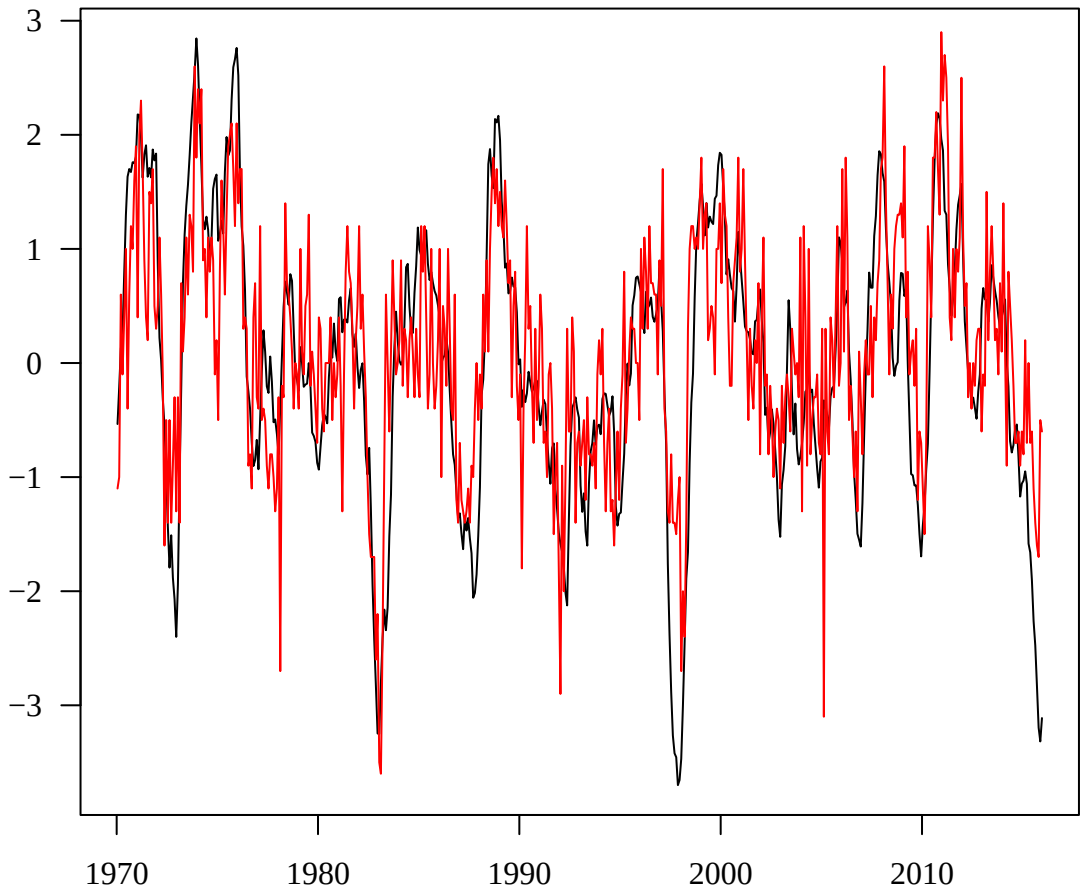
```
V <- temp2$v %*% diag(temp2$d)
par(las = 1, mar = c(3, 3, 1, 1), family = "serif")
plot(tm1, V[, 1] / 15, type = "l", las = 1, xlab = "Year", ylab = "")
library(rsoi)
enso <- download_enso()
order <- order(enso$Date)
SOI <- enso$SOI[order]
(start <- which(enso$Date[order] == "1970-01-01"))
```

```
## [1] 229
```

```
(end <- which(enso$Date[order] == "2015-12-01"))
```

```
## [1] 780
```

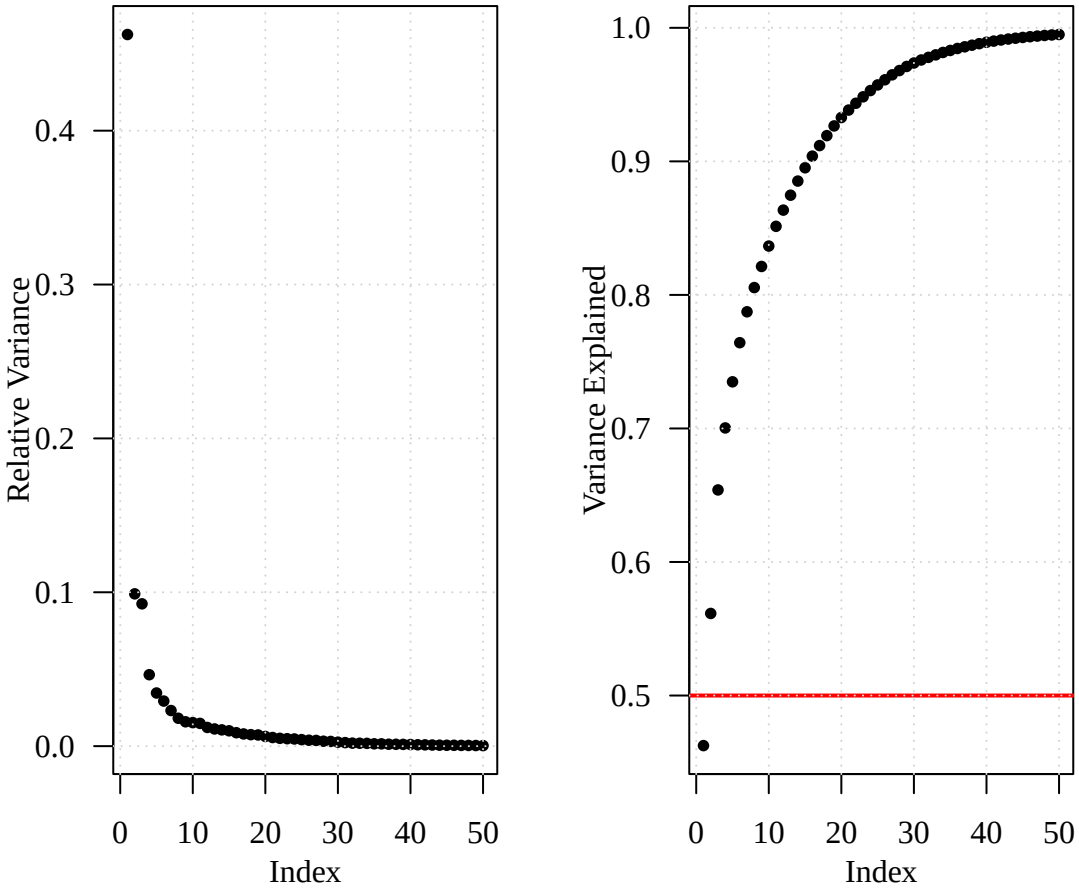
```
lines(tm1, SOI[start:end], col = "red")
```




**Scree plot: variance explained by EOFs**

We now examine how much variance is explained by each EOF and the cumulative variance explained by the leading modes.

```
par(mar = c(4, 4, 1, 1), mfrow = c(1, 2), las = 1, mgp = c(2, 1, 0),
    family = "serif")
dt <- ((temp2$d^2) / sum(temp2$d^2))
plot(1:50, dt[1:50], xlab = "Index", ylab = "Relative Variance",
     pch = 16, cex = 0.8)
grid()
dt <- (cumsum(temp2$d^2) / sum(temp2$d^2))
plot(1:50, dt[1:50], xlab = "Index", ylab = "Variance Explained", pch = 16, cex = 0.8)
yline(0.5, col = "red", lwd = 2)
grid()
```




**January 1998 El Niño event: reconstruction with EOFs**

January 1998 is a strong El Niño month. We reconstruct the SST anomaly field using 1, 2, and 3 leading EOFs and compare with the actual anomaly field.

```
V <- temp2$v %*% diag(temp2$d)
J <- 337 # the index for 1998 Jan.
zr <- range(SST_anomalies, na.rm = TRUE)
set.panel(2, 2)
```

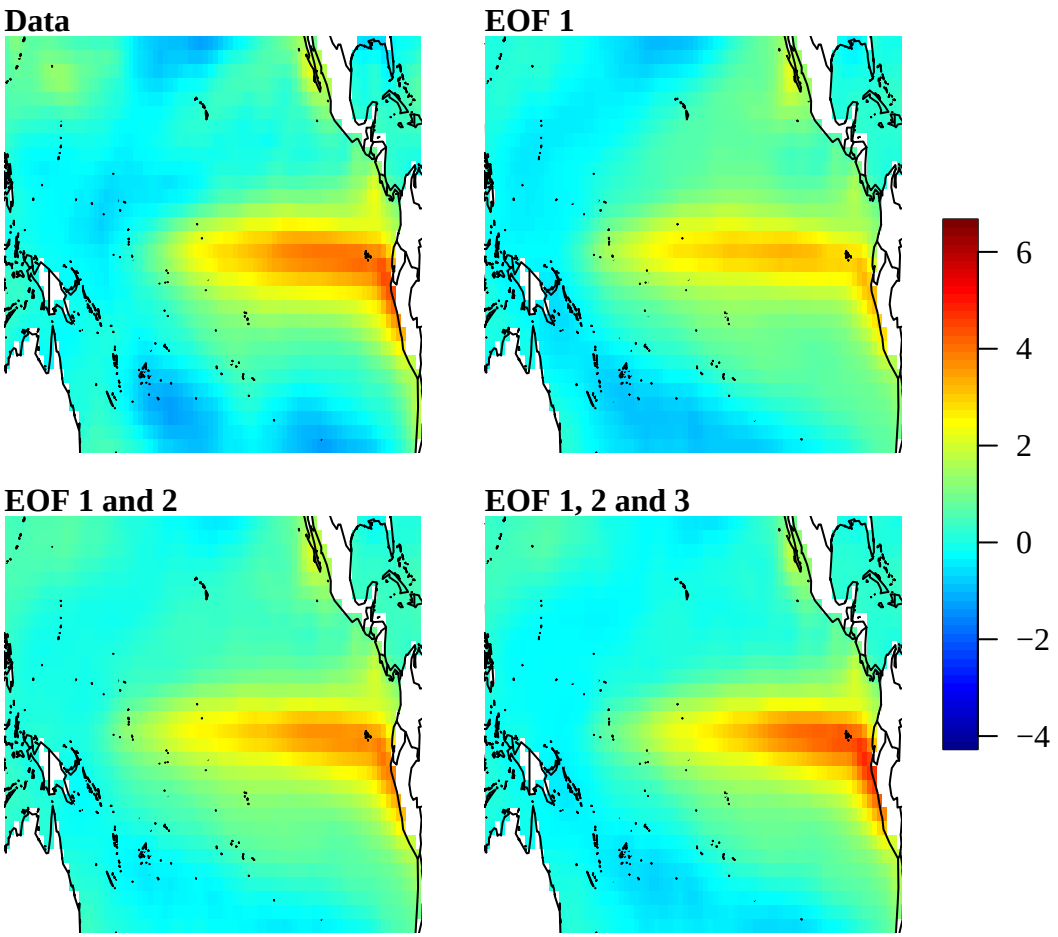
```
## plot window will lay out plots in a 2 by 2 matrix
```

```
par(mar = c(1, 1, 1, 1), oma = c(0, 0, 0, 6), family = "serif")
image(lon1, lat1, SST_anomalies[, , J], axes = FALSE, xlab = "", ylab = "",
      col = tim.colors(256), zlim = zr)
map("world2", add = TRUE)
title("Data", adj = 0)
image(lon1, lat1, V[J, 1] * U1, axes = FALSE, xlab = "", ylab = "",
      col = tim.colors(256), zlim = zr)
map("world2", add = TRUE)
title("EOF 1", adj = 0)
image(lon1, lat1, V[J, 1] * U1 + V[J, 2] * U2, axes = FALSE,
      xlab = "", ylab = "", col = tim.colors(256), zlim = zr)
map("world2", add = TRUE)
```

```
title("EOF 1 and 2", adj = 0)
image(lon1, lat1, V[J, 1] * U1 + V[J, 2] * U2 + V[J, 3] * U3,
      axes = FALSE, xlab = "", ylab = "", col = tim.colors(256),
      zlim = zr)
map("world2", add = TRUE)
title("EOF 1, 2 and 3", adj = 0)
set.panel()
```

```
## plot window will lay out plots in a 1 by 1 matrix
```

```
par(oma = c(0, 0, 0, 0))
image.plot(legend.only = TRUE, zlim = zr, horizontal = FALSE, legend.shrink = 0.6)
```

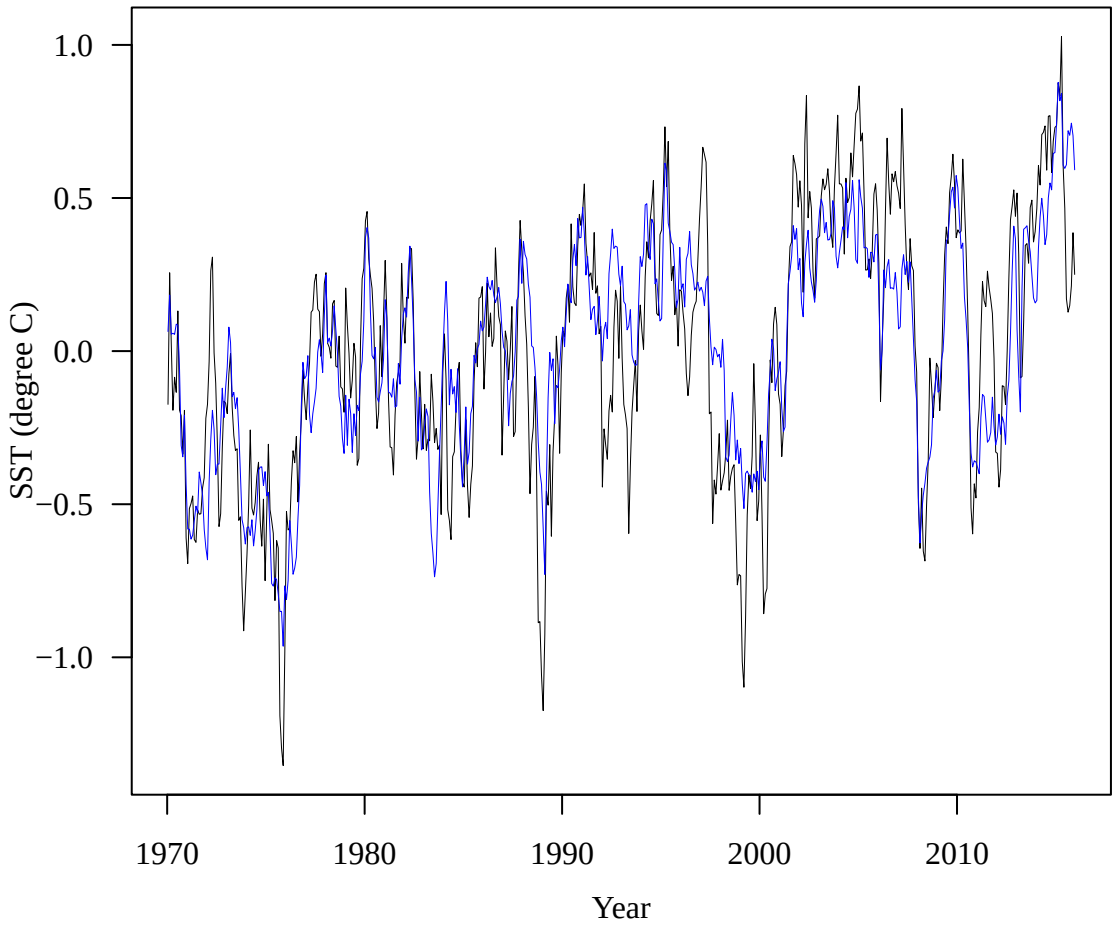



**Local time series reconstruction using leading EOFs**

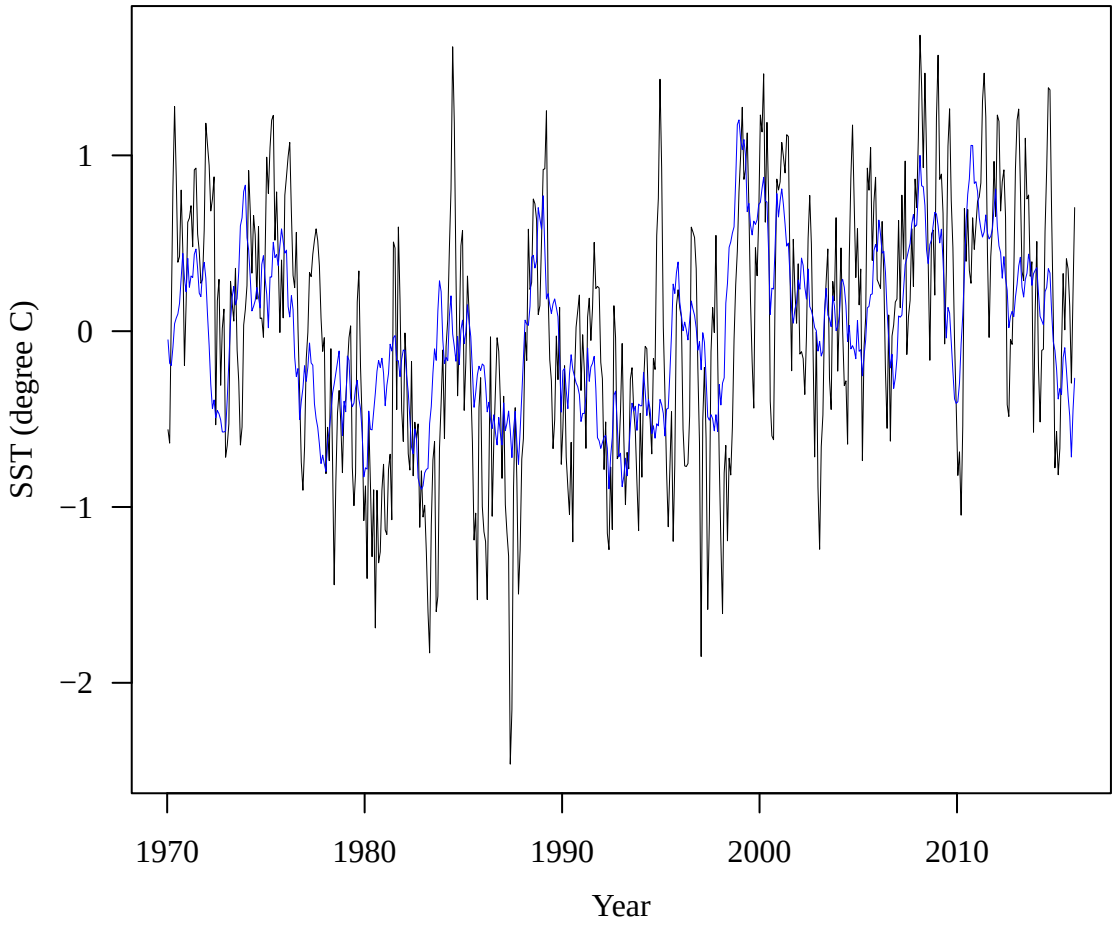
We now look at reconstruction at individual grid boxes, comparing the observed anomaly time series with the reconstruction using the first three EOFs.

```
par(las = 1, mar = c(3.5, 3.5, 1, 1), mgp = c(2.4, 1, 0), family = "serif")
plot(tm1, SST_anomalies[20, 15,], type = "l", xlab = "Year",
     ylab = "SST (degree C)", lwd = 0.5)
```

```
lines(tm1, V[, 1] * U1[20, 15] + V[, 2] * U2[20, 15] + V[, 3] * U3[20, 15],
      lwd = 0.5, col = "blue")
```




```
plot(tm1, SST_anomalies[40, 30,], type = "l", xlab = "Year",
     ylab = "SST (degree C)", lwd = 0.5)
lines(tm1, V[, 1] * U1[40, 30] + V[, 2] * U2[40, 30] + V[, 3] * U3[40, 30],
      lwd = 0.5, col = "blue")
```

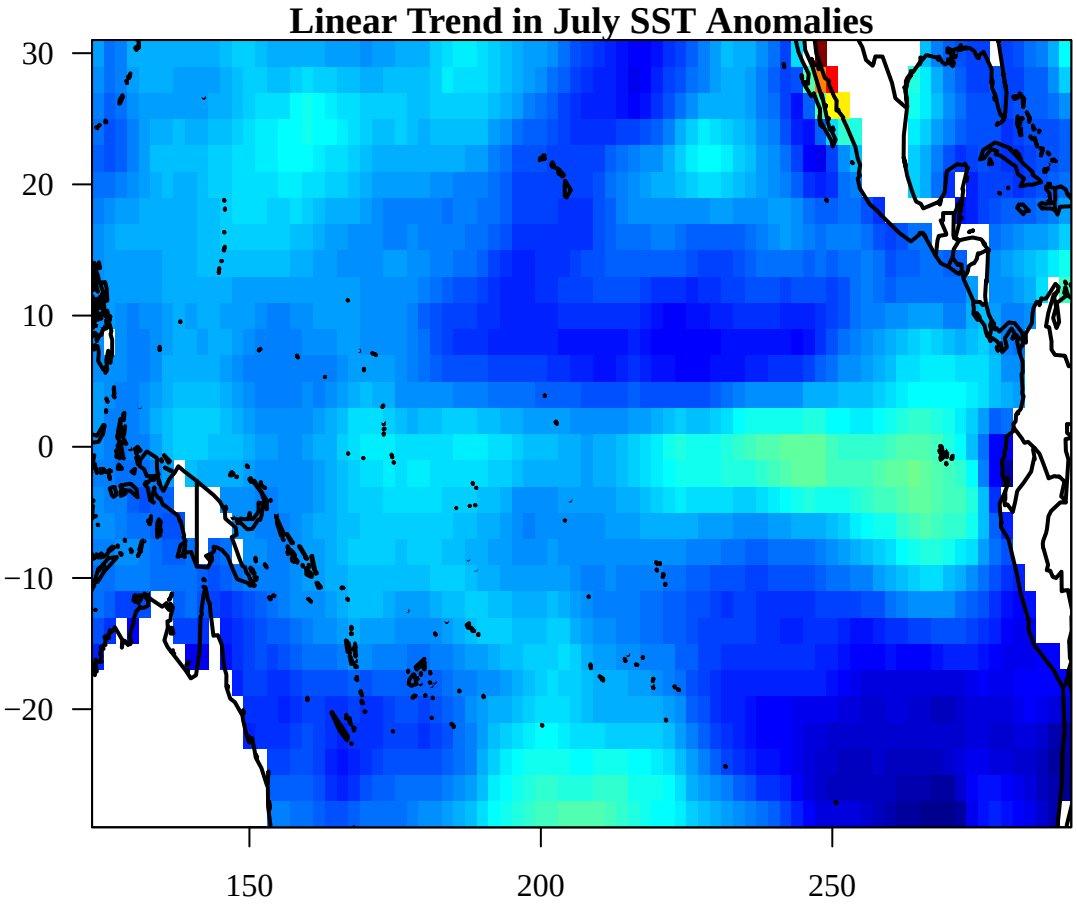



**Trend and covariance patterns (July-only series)**

Finally, we illustrate two additional analyses:

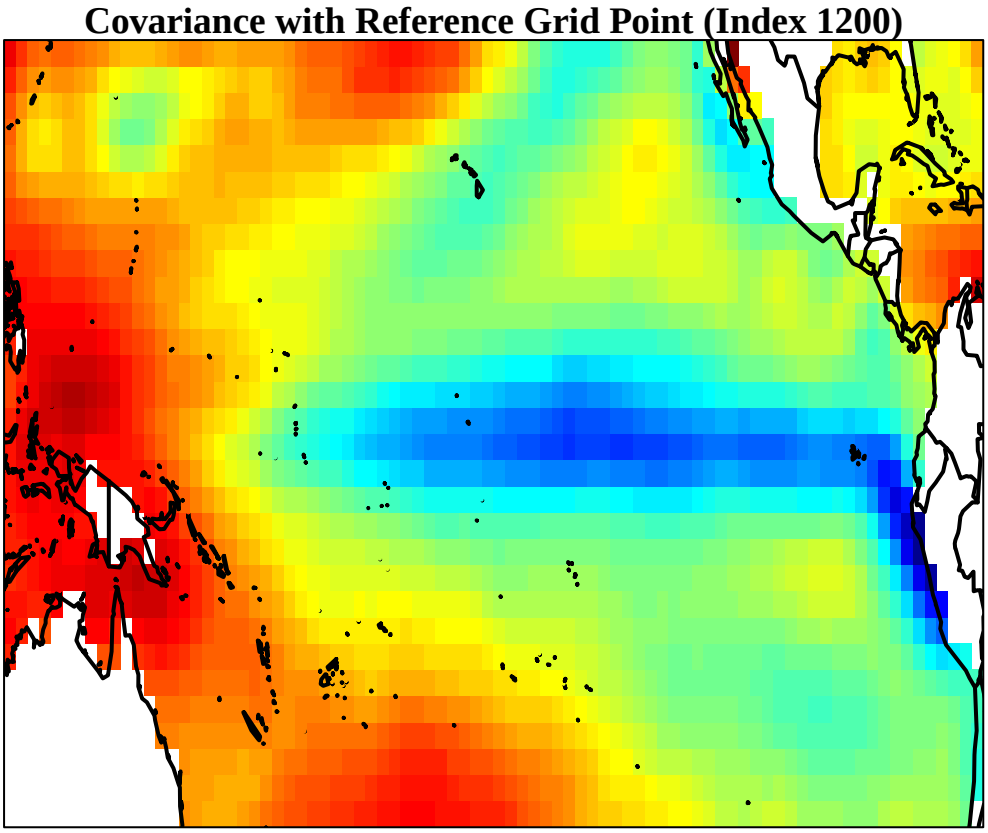
1. Linear trend in July SST anomalies at each grid box.
2. Covariance pattern with respect to a particular grid location.

```
SST_July <- array(SST_anomalies[,, seq(7, 7 + 45 * 12, 12)], dim = c(84 * 30, 46))

SST_July_OK <- SST_July[!ind,]

yr <- 1:46
lm_trend <- apply(SST_July_OK, 1, function(x) lm(x ~ yr)$coefficients)

trend <- matrix(NA, 84 * 30)
trend[!ind] <- lm_trend[2,]; trend <- matrix(trend, 84, 30)

par(las = 1, mar = c(3.5, 3.5, 1, 1), mgp = c(2.4, 1, 0), family = "serif")
image(lon1, lat1, trend, xlab = "", ylab = "", col = tim.colors())
map("world2", add = TRUE, lwd = 2)
box()
title("Linear Trend in July SST Anomalies")
```




We also compute the covariance field between SST at one reference location and all other grid boxes (using only July anomalies).

```
cov <- cov(t(SST_July_OK))
COV <- matrix(NA, 84 * 30)
COV[!ind] <- cov[, 1200]; COV <- matrix(COV, 84, 30)

par(las = 1, mar = c(3.5, 3.5, 1, 1), mgp = c(2.4, 1, 0), family = "serif")
image(lon1, lat1, COV, axes = FALSE, xlab = "", ylab = "", col = tim.colors())
map("world2", add = TRUE, lwd = 2)
box()
title("Covariance with Reference Grid Point (Index 1200)")
```




In this example we:

1. Constructed SST anomalies by removing the climatological seasonal cycle.
2. Applied EOF (PCA in space) to extract dominant SST variability patterns.
3. Demonstrated that the leading EOF/PC pair captures ENSO-like variability by comparing PC1 with the SOI.
4. Reconstructed a strong El Niño event (January 1998) using only a few EOFs.
5. Explored trends and covariance structures in July SST anomalies.

EOF analysis provides a compact way to describe and interpret large geophysical data sets such as global SST fields, and forms a foundation for many advanced climate data analysis methods.