Interpolation spatiale de la température de l'air à l'échelle de la Belgique

1 Préparation des données

1.1 Packages et données

```
library(sp)
library(rgdal)
library(gstat)
library(dplyr)

load("TT_interpolation.Rdata")

# données journalières de 2016 pour 130 stations:
head(data)

##   CODE      DAY MONTH   TX  TN
## 1  101 20160101     1  7.0 1.6
## 2  101 20160102     1 12.0 2.0
## 3  101 20160103     1  7.0 7.0
## 4  101 20160104     1  8.4 6.0
## 5  101 20160105     1  8.0 5.0
## 6  101 20160106     1  9.4 5.0

# metadonnées associées à chacune des stations
head(stations)

##   CODE         NAME      LAT      LON  ALT QUALITY
## 1  101 KNOKKE-HEIST 51.33694 3.326667  4.0       0
## 2  102     KOKSIJDE 51.08806 2.652500  4.7       0
## 3  103  MIDDELKERKE 51.20028 2.887222  3.5       0
## 4  199         ZELE 51.05667 4.042778  5.0       0
## 5  200       DEURNE 51.19222 4.453056 14.0       0
## 6  204     Stabroek 51.32472 4.363889  4.0       0

codes=unique(data$CODE)
length(codes)

## [1] 130
```

TX : température maximale
TN : température minimale
QUALITY : 2 = influence d'une surface d'eau (fleuve, canal, etc.) ; 1 = influence de surfaces bâties ; 0 = pas de problème particulier (voir Figure 2 pour la répartion des stations dans ces 3 classes).

1.2 Modèle de terrain

```
DEM=readGDAL("DEM_GTOPO30.tif")

## DEM_GTOPO30.tif has GDAL driver GTiff
## and has 264 rows and 480 columns

colnames(DEM@data)="ALT_DEM"
DEM=DEM[which(! is.na(DEM$ALT_DEM)),]

stations.sp=stations
coordinates(stations.sp) = ~LON+LAT
proj_LatLon="+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0"
proj4string(stations.sp) =  proj_LatLon

over.out=over(stations.sp, DEM)
stations$ALT_DEM=over.out$ALT_DEM
```

On observe des différences parfois importantes entre l'altitude reprise dans notre base de donnée (colonne ALT) et l'altitude du modèle de terrain (ALT_DEM) :

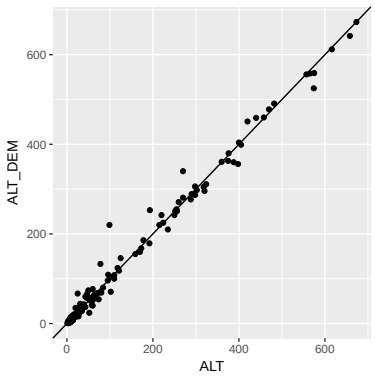

FIGURE 1 – Scatter plot of ALT versus ALT DEM

1.3 Données journalières, mensuelles et annuelles (sous forme d'objets de classe sp)

```
data.d=left_join(data, stations, by='CODE')
coordinates(data.d) = ~LON+LAT
proj4string(data.d) =  proj_LatLon

data.m = data %>% group_by(CODE, MONTH) %>%
         summarize(TX_AVG=mean(TX), TN_AVG=mean(TN)) %>% as.data.frame()
data.m=left_join(data.m, stations, by='CODE')
coordinates(data.m) = ~LON+LAT
proj4string(data.m) =  proj_LatLon

data.y = data %>% group_by(CODE) %>%
         summarize(TX_AVG=mean(TX), TN_AVG=mean(TN)) %>% as.data.frame()
data.y=left_join(data.y, stations, by='CODE')
coordinates(data.y) = ~LON+LAT
proj4string(data.y) =  proj_LatLon
```

2 Interpolation des moyennes annuelles

```
# fixed variogram: exponential, range of 80km, zero nugget
vgm=vgm(1, "Exp", 80, 0)

# ordinary kriging
OK.TX=krige(TX_AVG~1, data.y, DEM, vgm)
OK.TN=krige(TN_AVG~1, data.y, DEM, vgm)

# external drift kriging
KED.TX=krige(TX_AVG~ALT_DEM, data.y, DEM, vgm)
KED.TN=krige(TN_AVG~ALT_DEM, data.y, DEM, vgm)

# measurement error definition:
#   error standard deviation 0.1°C if QUALITY=0
#   error standard deviation 0.5°C if QUALITY=1
#   error standard deviation 1°C if QUALITY=2

error_weight=rep(1/(0.1^2), nrow(data.y))
error_weight[which(data.y$QUALITY==1)]=1/(0.5^2)
error_weight[which(data.y$QUALITY==2)]=1

# ordinary kriging with measurement errors
OK2.TX=krige(TX_AVG~1, data.y, DEM, vgm, weight=error_weight)
OK2.TN=krige(TN_AVG~1, data.y, DEM, vgm, weight=error_weight)

# external drift kriging with measurement errors
KED2.TX=krige(TX_AVG~ALT_DEM, data.y, DEM, vgm, weight=error_weight)
KED2.TN=krige(TN_AVG~ALT_DEM, data.y, DEM, vgm, weight=error_weight)
```

Comparaison des résultats d'interpolation sous forme de cartes

Les cartes des Figures 2 et 3 représentent pour les moyennes annuelles des maximax et minimas :

- les valeurs aux stations avant interpolation,
- la valeurs du paramètre QUALITY associé à chaque station,
- le résultat de l'interpolation par ordinary kriging (OK),
- le résultat de l'interpolation par external drift kriging (KED) avec prise en compte du modèle de terrain,
- le résultat de l'interpolation par ordinary kriging (OK) avec prise en compte d'erreurs de mesure,
- le résultat de l'interpolation par external drift kriging (KED) avec prise en compte du modèle de terrain et d'erreurs de mesure.

Les résultats d'interpolation sont représentés à la résolution du modèle de terrain ($\pm$ 1km $\times$ 1km).

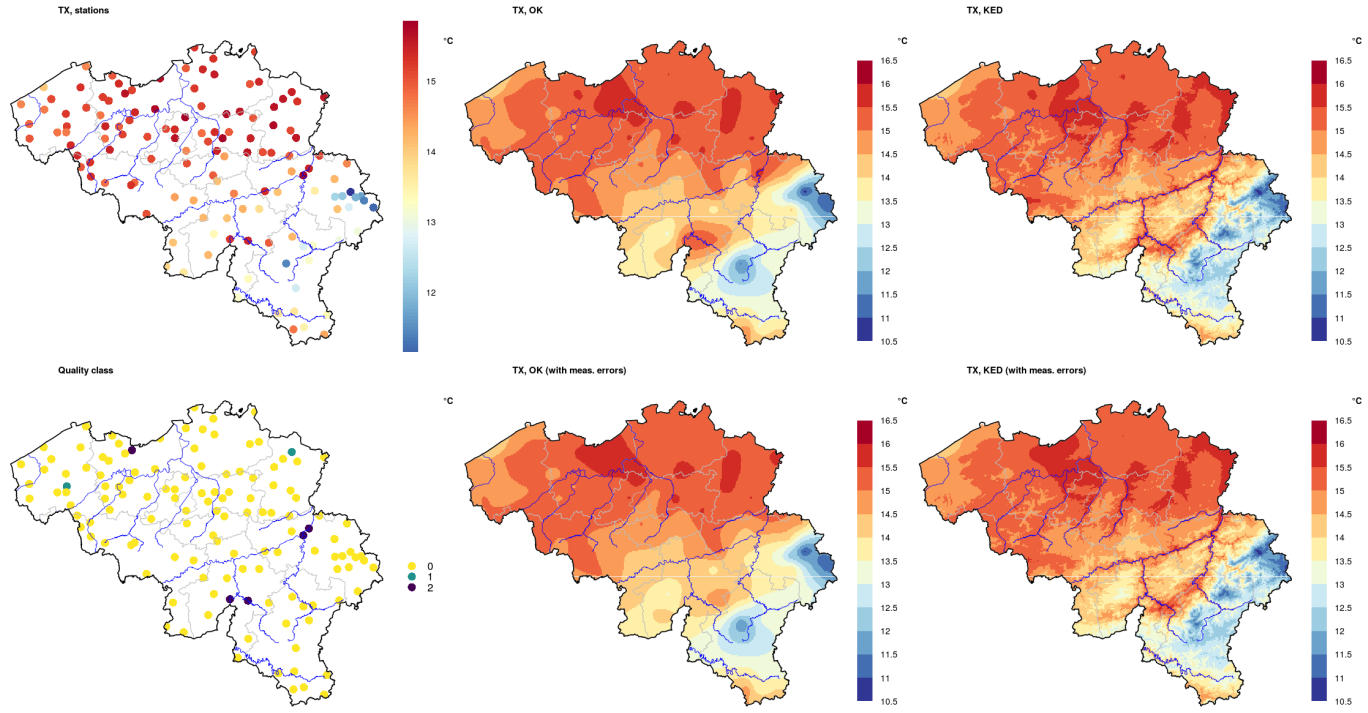

FIGURE 2 – Moyenne annuelle de la température journalière maximale (TX).

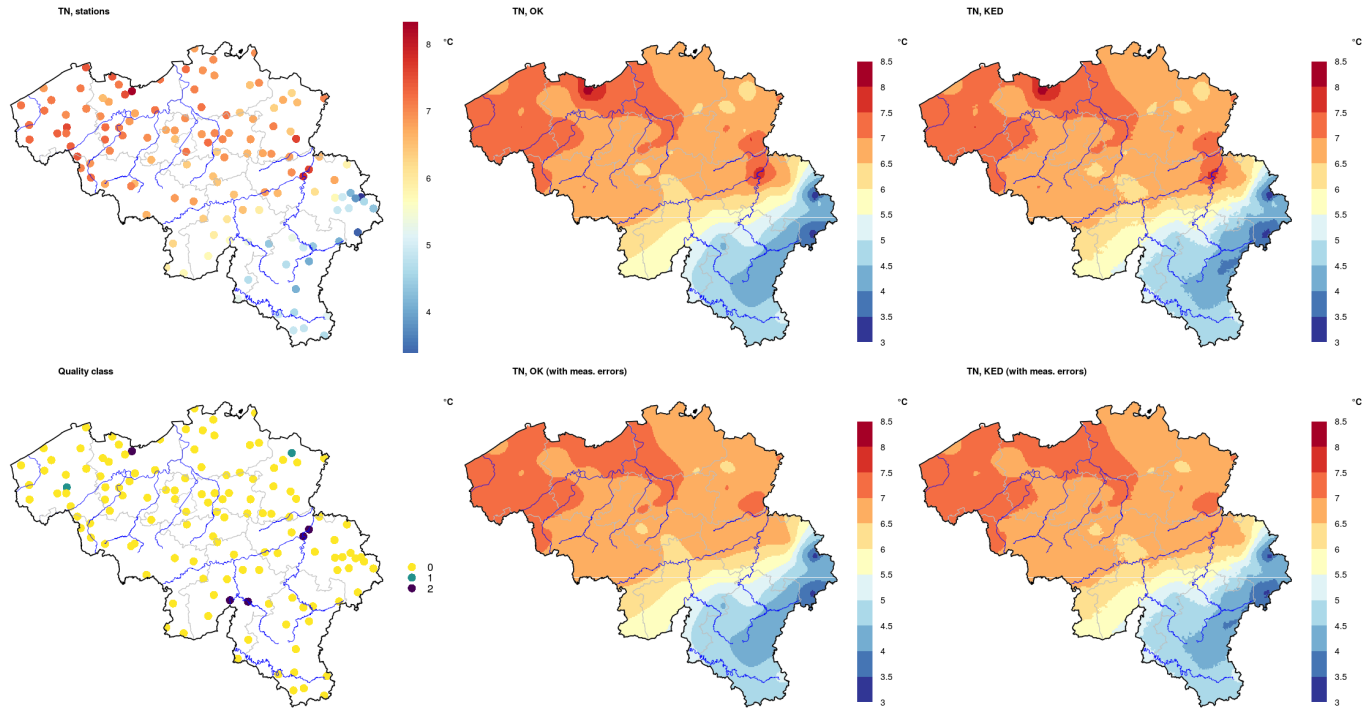

FIGURE 3 – Moyenne annuelle de la température journalière minimale (TN).

Observations.

— Le modèle de terrain est très (peut-être trop...) exploité dans le cas du TX, et nettement moins (trop peu ?) dans le cas du TN.

— La prise en compte d'erreurs de mesure permet de limiter l'influence de stations spatiallement peu représentatives (proximité de surfaces d'eau ou de surfaces bâties).

La méthode d'interpolation actuellement utilisée par l'IRM est le krigeage avec dérive externe sur base du modèle de terrain avec prise en compte d'erreur de mesures. Les résultats à la résolution du modèle de terrain sont moyennés à la résolution 5km $\times$ 5km dans notre base de données.

3 Validation croisée

Les lignes de codes suivantes permettent une validation croisée pour les moyennes annuelles et mensuelles ainsi que les valeurs journalières des maximas et minimas de la température de l'air. Seules les méthodes d'interpolation exacte (OK et KED) sont reprises. Les méthodes qui intègrent les erreurs de mesures ne sont pas considérées ici.

En chaque station et pour chacune des méthodes, une estimation est associée à l'observation disponible. Plusieurs indices constuits sur ce bias (estimation – observation) permettent d'évaluer les méthodes considérées. Ces indices doivent encore être établis et analysés.

3.1 Moyennes annuelles

```
data.y$TX_OK=NA
data.y$TN_OK=NA
data.y$TX_KED=NA
data.y$TN_KED=NA

vgm=vgm(1, "Exp", 80, 0)     # fixed variogram
for (code in codes){
  # observations considered for spatial modeling :
  data.mod=data.y[which(data.y$CODE !=code),]
  # prediction location :
  data.pred=data.y[which(data.y$CODE ==code),]

  OK.TX=krige(TX_AVG ~ 1, data.mod, data.pred, vgm)
  KED.TX=krige(TX_AVG ~ ALT_DEM, data.mod, data.pred, vgm)
  OK.TN=krige(TN_AVG ~ 1, data.mod, data.pred, vgm)
  KED.TN=krige(TN_AVG ~ ALT_DEM, data.mod, data.pred, vgm)

  data.y$TX_OK[which(data.y$CODE==code)]=OK.TX$var1.pred
  data.y$TX_KED[which(data.y$CODE==code)]=KED.TX$var1.pred
  data.y$TN_OK[which(data.y$CODE==code)]=OK.TN$var1.pred
  data.y$TN_KED[which(data.y$CODE==code)]=KED.TN$var1.pred
}
```

Biais moyen et extrêmes pour chaque méthode et paramètre :

```
bias=data.y$TX_OK-data.y$TX_AVG
data.frame(MAX=max(bias), MIN=min(bias), MAE=mean(abs(bias)))

##        MAX       MIN       MAE
## 1 2.083085 -1.418802 0.4246142

bias=data.y$TX_KED-data.y$TX_AVG
data.frame(MAX=max(bias), MIN=min(bias), MAE=mean(abs(bias)))

##         MAX       MIN       MAE
## 1 0.9579745 -1.067923 0.2904451

bias=data.y$TN_OK-data.y$TN_AVG
data.frame(MAX=max(bias), MIN=min(bias), MAE=mean(abs(bias)))

##        MAX       MIN       MAE
## 1 1.311567 -1.132757 0.2900128
```

```
bias=data.y$TN_KED-data.y$TN_AVG
data.frame(MAX=max(bias), MIN=min(bias), MAE=mean(abs(bias)))

##        MAX       MIN       MAE
## 1 1.375542 -1.199389 0.2943896
```

3.2 Moyennes mensuelles

```
months=unique(data.m$MONTH)

data.m$TX_OK=NA
data.m$TN_OK=NA
data.m$TX_KED=NA
data.m$TN_KED=NA

vgm=vgm(1, "Exp", 80, 0)     # fixed variogram
for (month in months){
  print(month)
  for (code in codes){
    # observations considered for spatial modeling :
    data.mod=data.m[which(data.m$CODE !=code & data.m$MONTH==month),]
    # prediction location
    data.pred=data.m[which(data.m$CODE ==code & data.m$MONTH==month),]

    OK.TX=krige(TX_AVG ~ 1, data.mod, data.pred, vgm)
    KED.TX=krige(TX_AVG ~ ALT_DEM, data.mod, data.pred, vgm)
    OK.TN=krige(TN_AVG ~ 1, data.mod, data.pred, vgm)
    KED.TN=krige(TN_AVG ~ ALT_DEM, data.mod, data.pred, vgm)

    data.m$TX_OK[which(data.m$CODE==code & data.m$MONTH==month)]=OK.TX$var1.pred
    data.m$TX_KED[which(data.m$CODE==code & data.m$MONTH==month)]=KED.TX$var1.pred
    data.m$TN_OK[which(data.m$CODE==code & data.m$MONTH==month)]=OK.TN$var1.pred
    data.m$TN_KED[which(data.m$CODE==code & data.m$MONTH==month)]=KED.TN$var1.pred
  }
}
```

3.3 Valeurs journalières

```
days=unique(data.d$DAY)

data.d$TX_COR=NA
data.d$TN_COR=NA
data.d$TX_OK=NA
data.d$TN_OK=NA
data.d$TX_KED=NA
data.d$TN_KED=NA

vgm=vgm(1, "Exp", 80, 0)     # fixed variogram
for (day in days){
   for (code in codes){
    # observations considered for spatial modeling
    data.mod=data.d[which(data.d$CODE !=code & data.d$DAY==day),]
    # prediction location
    data.pred=data.d[which(data.d$CODE ==code & data.d$DAY==day),]
```

```
    # correlation coefficients:
    cor.TX=cor(data.mod$ALT_DEM, data.mod$TX)
    cor.TN=cor(data.mod$ALT_DEM, data.mod$TN)

    OK.TX=krige(TX ~ 1, data.mod, data.pred, vgm)
    KED.TX=krige(TX ~ ALT_DEM, data.mod, data.pred, vgm)
    OK.TN=krige(TN ~ 1, data.mod, data.pred, vgm)
    KED.TN=krige(TN ~ ALT_DEM, data.mod, data.pred, vgm)

    data.d$TX_COR[which(data.d$CODE==code & data.d$DAY==day)]=cor.TX
    data.d$TN_COR[which(data.d$CODE==code & data.d$DAY==day)]=cor.TN
    data.d$TX_OK[which(data.d$CODE==code & data.d$DAY==day)]=OK.TX$var1.pred
    data.d$TX_KED[which(data.d$CODE==code & data.d$DAY==day)]=KED.TX$var1.pred
    data.d$TN_OK[which(data.d$CODE==code & data.d$DAY==day)]=OK.TN$var1.pred
    data.d$TN_KED[which(data.d$CODE==code & data.d$DAY==day)]=KED.TN$var1.pred
  }
}
```