TECHNICAL REPORT

ESPON

Spatial analysis for quality control

Phase 1: The identification of logical input errors and statistical outliers

Draft: 2

MAIN RESULTS

- Exceptional values can arise from logical input errors and true outlying data.
- The accurate identification of an exceptional value is important as input errors should be treated differently to true outlying data.
- Input errors can usually be identified mathematically or sometimes, statistically. Outliers are identified statistically.
- A weighted evidence approach to statistically identify outliers is presented using a worked example that has been coded with R open source software.

ESPON 2013 DATABASE

MARCH 2011

EUROPEAN UNION
Part-financed by the European Regional Development Fund
INVESTING IN YOUR FUTURE

28 pages

List of authors

Paul Harris, NCG - NUIM
Martin Charlton, NCG - NUIM
Stewart Fotheringham, NCG - NUIM

Contact

Paul.Harris@nuim.ie
Martin.Charlton@nuim.ie
Stewart.Fotheringham@nuim.ie

National Centre for Geocomputation (NCG),
National University of Ireland (Maynooth)
Tel. (+353) 1 708 6455

1 Introduction and aims

The ESPON 2013 Database should be as free from errors as possible. It follows from this that detecting errors is an important activity in both data entry and data checking. This technical report (which accompanies the final report) aims to clarify the results in the Second Interim Report (SIR) on how mathematical, statistical and spatial analysis tools can be applied to the ESPON 2013 Database in order to find ‘logical input errors’ (stage 1 detection) and ‘statistical outliers’ (stage 2 detection).

We first respond to queries or criticisms raised by the ESPON management on the SIR. Secondly (and in some respects, a response to criticisms), we propose a novel approach to the detection of statistical outliers (stage 2 detection), which we have developed since the SIR. As demonstration of this ‘weight of evidence’ approach, it is applied to a real (space-time) ESPON data set. The case study is presented with an R script so that the ‘weight of evidence’ approach is reproducible.

2 Responses to queries or criticisms

We shall simply respond to each point made in turn.

"Please also deliver the R-Scripts as a file." **Response** - This was already done, and the R script for this technical report has been similarly delivered.

"Chapter 4, however, lacks a comparison between the various techniques. Which techniques have shown the best results for these worked examples and are therefore the most promising?" **Response** - For statistical outliers, this is a very difficult task to do with any objectivity or certainty. Outliers in the ESPON database can arise for many different reasons and as such, require different approaches for their detection. Furthermore, the calibration of an outlier detection technique commonly involves many judged, user-specified model decisions, where in the end, it is common to refer to an unusual observation as simply *potentially* outlying (i.e. statistical detection cannot usually say that an observation is outlying with certainty). For these reasons and more, we have adopted a 'weight of evidence' approach to outlier detection, using many of the SIR techniques. The approach is still in its infancy and future work should consider various refinements (see section 3 for details). For the detection of logical input errors (stage 1 detection), we recommend that their methods of detection are simply updated as they arise. It is impossible to foresee the infinite number of ways that a logical input error can occur. We have provided a start to this issue in the SIR and here, the metadata for each variable should provide a helpful basis for this detection.

"For ESPON it is also important to know what exactly the benefit is for ESPON of the research done on improving these techniques and procedures. The benefit in general is clear: the detection of outliers and errors can be done easier, faster and better. However, the report does not show (yet) what has been detected in the datasets that are/will be incorporated in the ESPON 2013 Database. The usefulness of this research for ESPON should be made more clear." **Response** - We demonstrate our 'weight of evidence' approach to a real case study dataset in section 3, where observations are flagged as outlying together with the nature of their 'outlyingness' (i.e. spatial, temporal, etc.). The approach can easily be embedded within the database architecture. Many of the detection techniques from the SIR have already been incorporated in this respect, some of which are key components of the more coherent 'weight of evidence' approach.

"Furthermore, detecting is one step, solving the problem is the next. Are the outliers and errors detected real errors? Can they be solved? And if so, how are they solved. It would be good if this could be shown in the next version of the report." **Response** - Observations can only be flagged as *potentially* outlying, but using a 'weight of evidence' approach allows some observations to have a higher *potential* in this respect, than others. Observations that are flagged as outlying from many detection techniques should be viewed with the strongest suspicion. The decision of what to do with an outlier (e.g. remove, replace, etc.) should ultimately reside with some expert on the given data set (or its provider). Mechanisms can be put in place within the database architecture to do this in an efficient and effective manner.

3 Case study: the 'weight of evidence' approach

In this case study, we apply our 'weight of evidence' approach to detect outliers in an ESPON space-time, *univariate* data set (where an extension to *multivariate* case is simple and direct). This approach applies nine representative and complementary statistical outlier detection techniques, where observations are flagged as outlying according the outcome of each technique. This then builds up a 'weight of evidence' for the likelihood of a given observation being outlying (i.e. statistical evidence is strongest for an observation that is flagged as outlying for all nine techniques).

According to our chosen techniques, we can indicate the nature of the outliers. Outliers can have any combination of aspatial, spatial, temporal or relational characteristics (see Fig. 1). Furthermore, this methodology can be specified such that a different weighting is attached to each technique. For example, we may want to attach more weight to observations that are outlying in a temporal sense. The detection techniques are all non-robust in form. However robust forms are preferred, as outliers can compromise the calibration of a given technique prior to its use as a method of detection. Extensions to robust forms, for each technique, should be the first set of refinements that are considered with this approach.

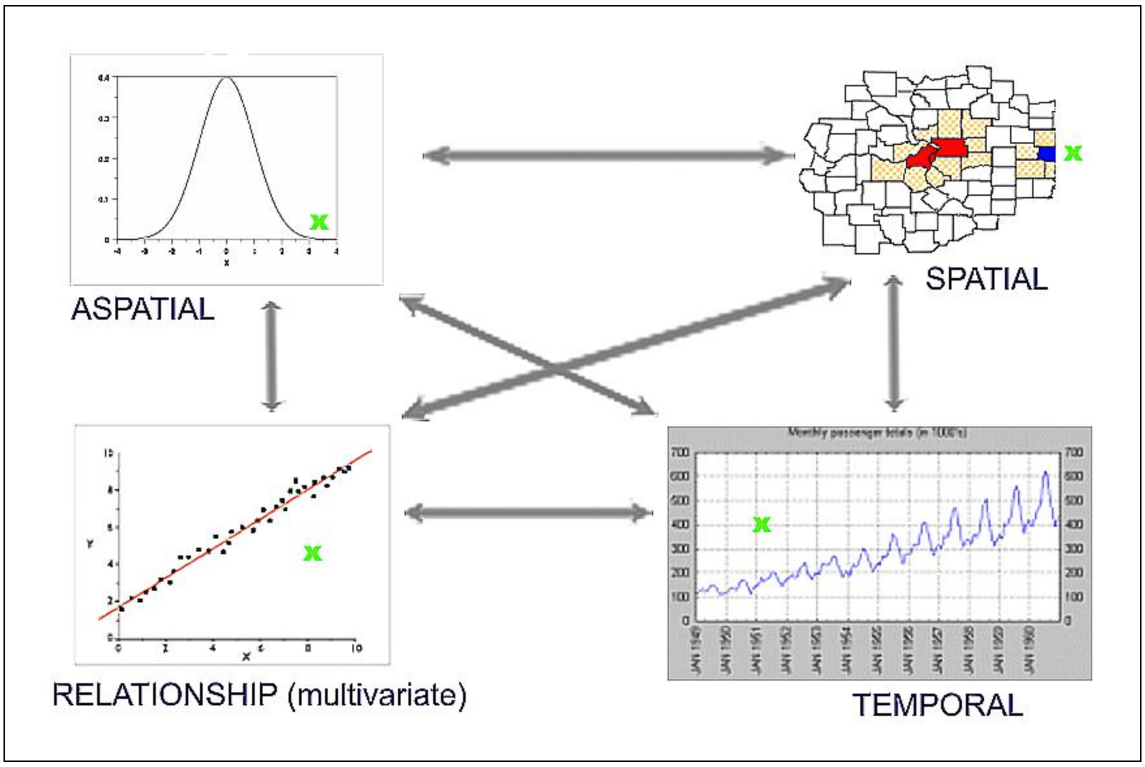
ASPATIAL
SPATIAL
RELATIONSHIP (multivariate)
TEMPORAL
Monthly passenger totals (in 1000's)

Figure 1 Types of outliers detected.

The nine techniques are: (i) boxplot statistics (for aspatial, univariate outliers); (ii) Hawkins' statistic (for spatial, univariate outliers); (iii) time series statistics (for temporal, univariate outliers); (iv) regression (for outliers from an aspatial, linear relationship); (v) locally weighted regression (for outliers from an aspatial, nonlinear relationship); (vi) geographically weighted regression (for outliers from a spatial, nonlinear relationship); (vii) principal component analysis (PCA, for outliers from a 'model-free', aspatial, linear relationship); (viii) locally weighted PCA (for outliers from a 'model-free', aspatial, nonlinear relationship); and (ix) geographically weighted PCA (for outliers from a 'model-free', spatial, nonlinear relationship). Here the PCA methods do not require domains-specific knowledge (i.e. how to choose covariates in a regression) and as such, are termed 'model-free'.

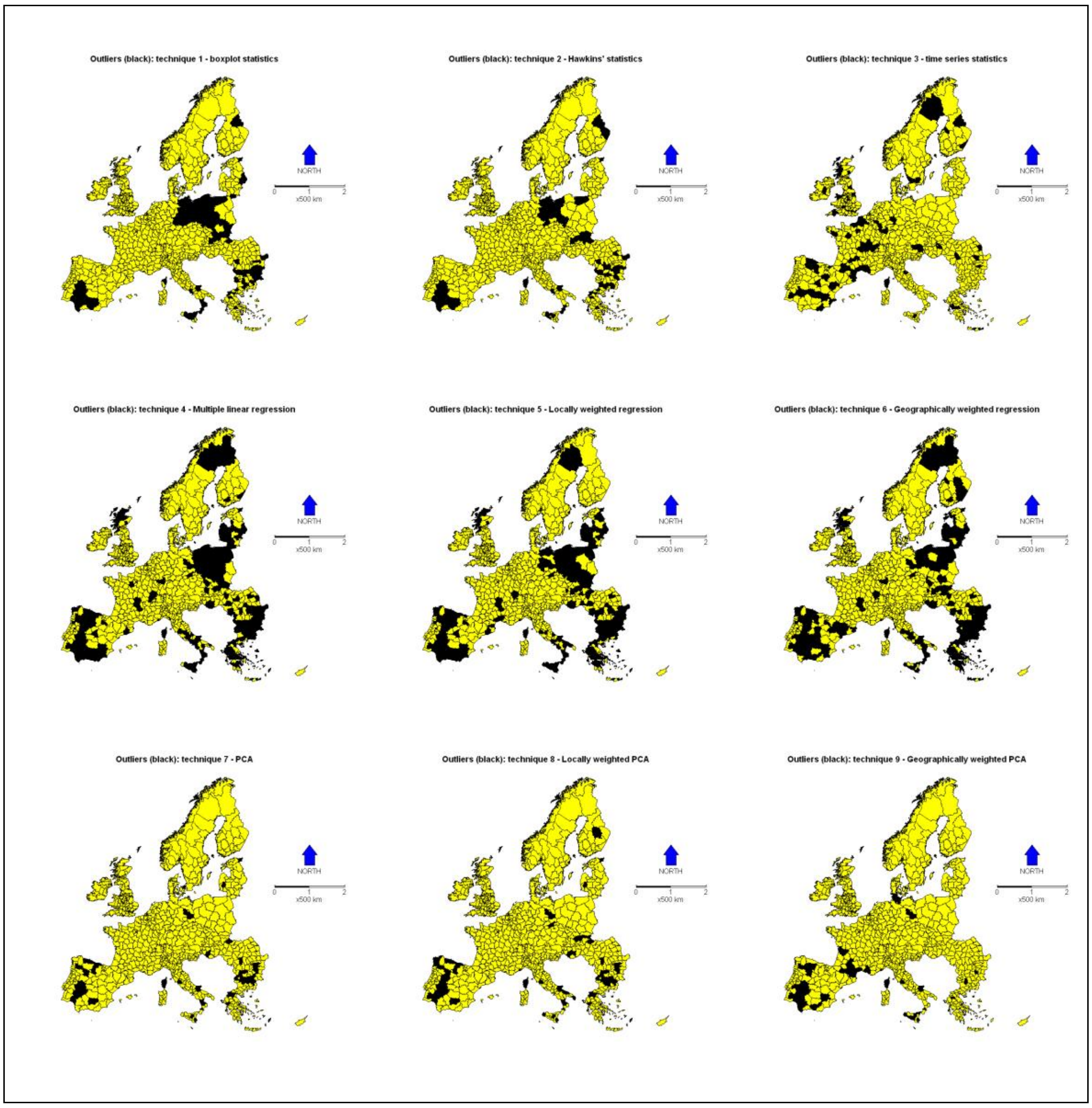
Outliers (black): technique 1 - boxplot statistics
Outliers (black): technique 2 - Hawkins' statistics
Outliers (black): technique 3 - time series statistics
Outliers (black): technique 4 - Multiple linear regression
Outliers (black): technique 5 - Locally weighted regression
Outliers (black): technique 6 - Geographically weighted regression
Outliers (black): technique 7 - PCA
Outliers (black): technique 8 - Locally weighted PCA
Outliers (black): technique 9 - Geographically weighted PCA
NORTH
0 1 2
x500 km

Figure 2 Outliers for each of nine 'weight of evidence' techniques.

We demonstrate our 'weight of evidence' outlier detection approach to ESPON unemployment rate data covering 709 NUTS23 regions for eight consecutive years 2000 to 2007. Here we show that outlier detection can not only be used as a data cleaning or screening exercise, but can also be used to uncover interesting or unusual relationships in the process that has not been considered before. Observe also that the regression methods can also be used for the prediction of missing data.

Full details of the outlier detection approach are given in the associated R script (see Appendix). Fig. 2 presents the results (maps) for each of the nine detection techniques and Fig.3 presents the summary of the results (i.e. the final 'weight of evidence' map). Thus as examples, there is strong evidence of at least one outlying unemployment rate (from the eight years) in a NUTS23 region in SW Spain and a NUTS23 region in N Corsica. Conversely, there is little evidence of at least one outlying unemployment rate (from the eight years) in a NUTS23 region in SW Ireland and all NUTS23 regions in Norway.

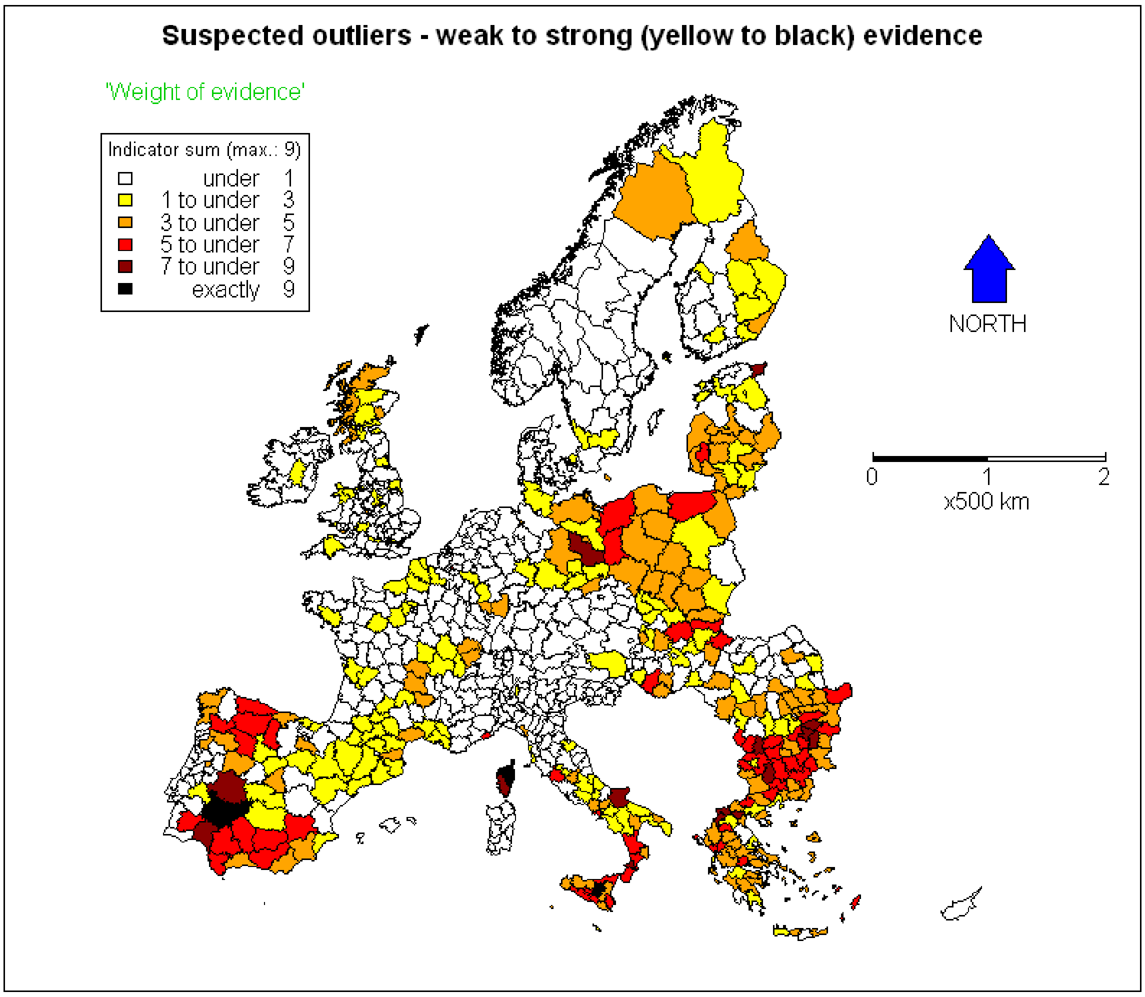

Figure 3 Final 'weight of evidence' map.

Reading

Fotheringham AS, Brunsdon C, Charlton ME (2002) Geographically Weighted Regression - the analysis of spatially varying relationships. Wiley, Chichester.

Ihaka R, Gentleman R (1996) R: A language for data analysis and graphics. Journal of Computational and Graphical Statistics 5, 299-314.

Joliffe IT (2002) Principal Components Analysis, 2nd edition, New York: Springer-Verlag

Loader C (2004) Smoothing: Local Regression Techniques. In Gentle J, Härdle W, Mori Y (eds) Handbook of Computational Statistics. Springer-Verlag, Heidelberg.

Appendix: Worked example of the 'weight of evidence' approach

```
#    This program is free software: you can redistribute it and/or modify
#    it under the terms of the GNU General Public License as published by
#    the Free Software Foundation, either version 3 of the License, or
#    (at your option) any later version.

#    This program is distributed in the hope that it will be useful,
#    but WITHOUT ANY WARRANTY; without even the implied warranty of
#    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
#    GNU General Public License for more details.

#    You should have received a copy of the GNU General Public License
#    along with this program.  If not, see <http://www.gnu.org/licenses/>.


# Challenge 10 - ESPON 2013 database

# P. Harris (NCG), M. Charlton (NCG) & C. Brunsdon (University of Leicester, UK)

# 24/10/10

# 'Weight of evidence' approach for detecting statistical outliers.

# There is no single 'best' outlier detection technique, so our aim is to:
# a. Apply a representative selection of outlier detection techniques.
# b. Flag an observation if it is a likely outlier according to each technique.
# c. Build up a weight of evidence for the likelihood of a given observation being statistically outlying.
# d. Suggest what type of outlier it is likely to be
#        - aspatial, spatial, temporal, relationship, some mixture…
# e. Consult an expert on the data to decide on the appropriate course of action.



# In this case, we employ 9 complementary and representative outlier detection techniques:

# 1. Boxplots
# 2. Hawkins' spatial statistic
# 3. Time series statistics
# 4. MLR residuals
# 5. LWR residuals
# 6. GWR residuals
# 7. PCA 'residuals'
# 8. LWPCA 'residuals'
# 9. GWPCA 'residuals'

# Techniques 1,2,4,5,6 and 7 were all investigated in the SIR.
# This script is for the weight of evidence approach in its most basic form.
# All outlier detection techniques are non-robust in design.
# Further work should replace each technique with a robust version
# to give a robust weight of evidence approach.

# Currently robust forms are readily available in R for:

# 1. Boxplots - robust form - adjusted boxplot - robustbase package
# 2. Hawkins' spatial statistic - robust form needs development/coding
# 3. Time series statistics - robust form needs development/coding
# 4. MLR residuals - robust form - many choices - e.g. robustbase package
# 5. LWR residuals  - robust form - many choices - locfit package
# 6. GWR residuals - robust form needs development/coding
# 7. PCA 'residuals' - robust form - many choices - e.g. mvoutlier package
# 8. LWPCA 'residuals' - robust form needs development/coding
# 9. GWPCA 'residuals' - robust form needs development/coding



# This particular script is also for the univariate space-time case but
# techniques are directly extendable to the multivariate space-time case.

# In 7 cases - outliers are chosen according to standard boxplot statistics.
# We could have specified adjusted boxplots statistics here instead
# but this can be part of any robust weight of evidence approach (future work, see above).

# Furthermore, in all cases - statistical tests could be used instead of boxplots.
# As it stands - this is only done in 2 cases.



# This weight of evidence approach can be performance tested using data that have been infected with outliers.
# However results can be difficult to interpret since:
# 1. It is difficult to guarentee that the infections actually produce outliers.
# 2. The data may already contain outliers.

# Further work should explore the use of simulated data sets
# or more attention should be paid to the theoretical properties of each outlier detection technique.



# Currently - this basic weight of evidence approach gives equal weights to each of the 9 detection techniques.
# There is no reason why these weights cannot vary...

# For example, these weights may work better:
# 1. Boxplots - 1
# 2. Hawkins - 1
# 3. Time series - 1
# 4. MLR - 1/3
# 5. LWR - 1/3
# 6. GWR - 1/3
# 7. PCA - 1/3
# 8. LWPCA - 1/3
# 9. GWPCA - 1/3



# The case study data set is unemployment rate ( = unemployment population/active population)
# at the NUTS 23 level covering 8 years: 2000-2007.
# Thus there are 8 variables at each of 790 regions = 6320 observations in total.

# We assume that the data has been checked for logical input errors (i.e. stage 1 detection is done).

# For brevity, we also assume that we only need at least one of eight time-specific unemployment rates
# in a region to be outlying.
# However in all techniques, execpt PCA/LWPCA/GWPCA, we actually identify outliers by year.


# An earlier version of output from this basic script (with clean and infected data) can be found at:
# http://www.espon.eu/main/Menu_Events/Menu_OpenSeminars/openseminar100609After.html
# This is the presenatation given at the Madrid open seminar 2010.


# Observe that outlier detection can reveal the most interesting results
# i.e. - those not expected.

# Observe also that the regression outlier detection methods can be used for the prediction of missing data too.


# R packages needed.....
# 1. GISTools (version 0.5-4) and all its dependencies (see SIR)
# 2. locfit (version 1.5-4)- for LWR



#################################################################################
# A. Functions needed for PCA, LWPCA & GWPCA ###################################
#################################################################################


# Author: Chris Brunsdon (plus some edits by Paul Harris)

# Details can be found in forthcoming articles

# Sub-functions....

wpca <- function(x,wt,...) {
		local.center <- function(x,wt)
			sweep(x,2,colSums(sweep(x,1,wt,'*'))/sum(wt))
		 svd(sweep(local.center(x,wt),1,sqrt(wt),'*'),...)}

	bw.by.nn <- function(loc,nn,eloc=loc) {
		result <- numeric(nrow(eloc))
		int.part <- floor(nn)
		frac.part <- nn - int.part
		if (frac.part == 0) {
			for (i in 1:nrow(eloc)) {
				d.sqr <- rowSums(sweep(loc,2,eloc[i,])**2)
				result[i] <- sort(d.sqr,partial=nn)[nn]}}
		else {
			for (i in 1:nrow(eloc)) {
				d.sqr <- rowSums(sweep(loc,2,eloc[i,])**2)
				d.tmp <- sort(d.sqr,partial=c(nn,nn+1))[c(nn,nn+1)]
				result[i] <- d.tmp[1]*(1-frac.part) + d.tmp[2]*frac.part}}
		sqrt(result)}

	bw.by.nn.1 <- function(x,nn,ex=x) {
		result <- numeric(nrow(ex))
		int.part <- floor(nn)
		frac.part <- nn - int.part
		if (frac.part == 0) {
			for (i in 1:nrow(ex)) {
				d.sqr <- rowSums(sweep(x,2,ex[i,])**2)
				result[i] <- sort(d.sqr,partial=nn)[nn]}}
		else {
			for (i in 1:nrow(ex)) {
				d.sqr <- rowSums(sweep(x,2,ex[i,])**2)
				d.tmp <- sort(d.sqr,partial=c(nn,nn+1))[c(nn,nn+1)]
				result[i] <- d.tmp[1]*(1-frac.part) + d.tmp[2]*frac.part}}
		sqrt(result)}



	# Basic functions....

	lwpca <- function(x,bw,k=2,ex=x,pcafun=wpca,...) {
		n <- nrow(ex)
		m <- ncol(x)
		w <- array(0,c(n,m,k))
		d <- matrix(0,n,k)
		bw <- bw*bw
		for (i in 1:n) {
			wt <- rowSums(sweep(x,2,ex[i,])**2)/bw
			use <- wt < 1
			if (sum(use) < 5)
				stop(paste('You will need a larger bandwidth: location ',i,
				' has only ',sum(use),'neigbours'))
```

```
                wt <- (1 - wt[use])**2
                temp <- pcafun(x[use,],wt,nu=0,nv=k)
                w[i,,] <- temp$v
                d[i,] <- temp$d[1:k]}
        if (!is.null(rownames(x))) dimnames(w)[[1]] <- rownames(x)
        if (!is.null(colnames(x))) dimnames(w)[[2]] <- colnames(x)
        dimnames(w)[[3]] <- paste("PC",1:k,sep='')
        list(loadings=w,var=d,bw=sqrt(bw))}

gwpca <- function(x,loc,bw,k=2,eloc=loc,pcafun=wpca,...) {
        n <- nrow(eloc)
        m <- ncol(x)
        w <- array(0,c(n,m,k))
        d <- matrix(0,n,k)
        bw <- bw*bw
        for (i in 1:n) {
                wt <- rowSums(sweep(loc,2,eloc[i,])**2)/bw
                use <- wt < 1
                if (sum(use) < 5)
                        stop(paste('You will need a larger bandwidth: location ',i,
                        ' has only ',sum(use),'neigbours'))
                wt <- (1 - wt[use])**2
                temp <- pcafun(x[use,],wt,nu=0,nv=k)
                w[i,,] <- temp$v
                d[i,] <- temp$d[1:k]}
        if (!is.null(rownames(x))) dimnames(w)[[1]] <- rownames(x)
        if (!is.null(colnames(x))) dimnames(w)[[2]] <- colnames(x)
        dimnames(w)[[3]] <- paste("PC",1:k,sep='')
        list(loadings=w,var=d,bw=sqrt(bw))}



# Automatic bandwidth functions...

lwpca.autobw <- function(x,bw.interval,k=2,ex=x,verbose=FALSE,...) {
        bw <- optimise(function(z) lwpca.cv(x,z,k=k,...),bw.interval)
        if (verbose) {
                cat("'Optimal' bandwidth = ",bw$minimum)
                cat(" Score is ",bw$objective)}
        lwpca(x,bw$minimum,k=k,ex=ex,...)}

lwpca.autobw.by.nn <- function(x,nn.interval,k=2,ex=x,verbose=FALSE,...) {
        bw <- optimise(function(z) lwpca.cv(x,bw.by.nn.1(x,z),k=k,...),nn.interval)
        if (verbose) {
                cat("'Optimal' nn = ",bw$minimum)
                cat(" Score is ",bw$objective)}
        lwpca(x,bw.by.nn(x,bw$minimum),k=k,ex=ex,...)}

gwpca.autobw <- function(x,loc,bw.interval,k=2,eloc=loc,verbose=FALSE,...) {
        bw <- optimise(function(z) gwpca.cv(x,loc,z,k=k,...),bw.interval)
```

```
          if (verbose) {
                    cat("'Optimal' bandwidth = ",bw$minimum)
                    cat(" Score is ",bw$objective)}
          gwpca(x,loc,bw$minimum,k=k,eloc=eloc,...)}

gwpca.autobw.by.nn <- function(x,loc,nn.interval,k=2,eloc=loc,verbose=FALSE,...) {
          bw <- optimise(function(z) gwpca.cv(x,loc,bw.by.nn(loc,z),k=k,...),nn.interval)
          if (verbose) {
                    cat("'Optimal' nn = ",bw$minimum)
                    cat(" Score is ",bw$objective)}
          gwpca(x,loc,bw.by.nn(loc,bw$minimum),k=k,eloc=eloc,...)}



# Cross-validation functions...

lwpca.cv <- function(x,bw,k=2,...) {
          n <- nrow(x)
          m <- ncol(x)
          w <- array(0,c(n,m,k))
          bw <- bw*bw
          score <- 0
          for (i in 1:n) {
                    wt <- rowSums(sweep(x,2,x[i,])**2)/bw
                    wt[i] <- Inf
                    use <- wt < 1
                    wt <- (1 - wt[use])**2
                    v <- wpca(x[use,],wt,nu=0,nv=k)$v
                    v <- v %*% t(v)
                    score <- score + sum((x[i,] - x[i,] %*% v))**2}
          score}

gwpca.cv <- function(x,loc,bw,k=2,...) {
          n <- nrow(loc)
          m <- ncol(x)
          w <- array(0,c(n,m,k))
          bw <- bw*bw
          score <- 0
          for (i in 1:n) {
                    wt <- rowSums(sweep(loc,2,loc[i,])**2)/bw
                    wt[i] <- Inf
                    use <- wt < 1
                    wt <- (1 - wt[use])**2
                    v <- wpca(x[use,],wt,nu=0,nv=k)$v
                    v <- v %*% t(v)
                    score <- score + sum((x[i,] - x[i,] %*% v))**2}
          score}
```

```
# Outlier/residual functions...

lwpca.cv.contrib <- function(x,bw,k=2,...) {
        n <- nrow(x)
        m <- ncol(x)
        w <- array(0,c(n,m,k))
        bw <- bw*bw
        score <- numeric(n)
        for (i in 1:n) {
                wt <- rowSums(sweep(x,2,x[i,])**2)/bw
                wt[i] <- Inf
                use <- wt < 1
                wt <- (1 - wt[use])**2
                v <- wpca(x[use,],wt,nu=0,nv=k)$v
                v <- v %*% t(v)
                score[i] <- sum((x[i,] - x[i,] %*% v))**2}
        score}

gwpca.cv.contrib <- function(x,loc,bw,k=2,...) {
        n <- nrow(loc)
        m <- ncol(x)
        w <- array(0,c(n,m,k))
        bw <- bw*bw
        score <- numeric(n)
        for (i in 1:n) {
                wt <- rowSums(sweep(loc,2,loc[i,])**2)/bw
                wt[i] <- Inf
                use <- wt < 1
                wt <- (1 - wt[use])**2
                v <- wpca(x[use,],wt,nu=0,nv=k)$v
                v <- v %*% t(v)
                score[i] <- sum((x[i,] - x[i,] %*% v))**2}
        score}



################################################################################
# B. Data input and exploratory data analysis ##################################
################################################################################



# Read in a data
data0 <- read.table("Worked example data.csv", ",", header=T)
colnames(data0) # Note the first 8 columns relate to an experiment with infected data - ignore this
attach(data0)


# Variables of interest are:
# UNEMP.R1.2000, UNEMP.R1.2001, UNEMP.R1.2002, UNEMP.R1.2003, UNEMP.R1.2004, UNEMP.R1.2005,
# UNEMP.R1.2006 and UNEMP.R1.2007



# Distributions...
X11(width=12,height=7)
par(mfrow=c(2,4))
hist(UNEMP.R1.2000, freq=F)
hist(UNEMP.R1.2001, freq=F)
hist(UNEMP.R1.2002, freq=F)
hist(UNEMP.R1.2003, freq=F)
hist(UNEMP.R1.2004, freq=F)
hist(UNEMP.R1.2005, freq=F)
hist(UNEMP.R1.2006, freq=F)
hist(UNEMP.R1.2007, freq=F)

X11(width=12,height=7)
par(mfrow=c(2,4))
hist(log(UNEMP.R1.2000), freq=F)
hist(log(UNEMP.R1.2001), freq=F)
hist(log(UNEMP.R1.2002), freq=F)
hist(log(UNEMP.R1.2003), freq=F)
hist(log(UNEMP.R1.2004), freq=F)
hist(log(UNEMP.R1.2005), freq=F)
hist(log(UNEMP.R1.2006), freq=F)
hist(log(UNEMP.R1.2007), freq=F)




# Correlations...
cor(data0[9:16])
X11(width=18,height=12)
pairs(data0[9:16],cex=0.5)




# Importing data as a ArcGIS shapefile & using GISTools to do some maps
require(GISTools)
# Ignore all warnings - this code is under development...

# Read in the shapefile...
data1 <- readShapePoly("FinalTS23.shp",
proj4string=CRS("+proj=laea+lat_0=50.0+lon_0=15.0+x_0=0+y_0=0"))
colnames(data1@data)

# Renaming each variable - as they have been truncated in ArcGIS...
colnames(data1@data) <- c("NUTS23x","NUTS23Ax",
"ACTIVE_2000x","ACTIVE_2001x","ACTIVE_2002x","ACTIVE_2003x","ACTIVE_2004x","ACTIVE_2005x","ACTIVE_2006x",
"ACTIVE_2007x",
"UNEMP_2000x","UNEMP_2001x","UNEMP_2002x","UNEMP_2003x","UNEMP_2004x","UNEMP_2005x","UNEMP_2006x",
"UNEMP_2007x",
"Xx","Yx")

# Checking the two data sets match up OK...
length(data0[,1])
length(data1@data[,1])
data.x <- cbind(data1@data$NUTS23x, NUTS23, data1@data$Xx, X, data1@data$Yx, Y)

# Coordinates....
Nuts23Coords <- as.matrix(cbind(X/1000, Y/1000))
colnames(Nuts23Coords) <- c("Easting","Northing")

# Maximum distances of sampled area in main directions...
max(X/1000)-min(X/1000)
max(Y/1000)-min(Y/1000)

# Renaming data...
data1@data <- data0
attach(data1@data)

# Create another data matrix for the analysis
#Data.1.scaled <- scale(as.matrix(data1@data[9:16])) # scaled
Data.1.scaled <- as.matrix(data1@data[9:16]) # not scaled
rownames(Data.1.scaled) <- data1@data[,17]
n1 <- length(Data.1.scaled[,1])

# An example choropleth map...
shades.e = auto.shading(UNEMP.R1.2000,5, cols=brewer.pal(5,'PuBuGn'))
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,UNEMP.R1.2000,shades.e)
title("Unemployment rate: year 2000")
choro.legend(1300000,400000,shades.e,fmt="%4.2f",title='Rate',cex=0.8)
map.scale(1800000,-650000,1100000,"x500 km",2,1)
north.arrow(1800000,1000000,80000, col="blue")



################################################################################
# 1. Standard boxplot outliers ##################################################
################################################################################


# Need to do this for each unemployment variable (i.e. each year) ...

z1 <- data0[,9]
bp <- boxplot.stats(z1, coef=1.5)
indicator.1.1 <-ifelse(z1>bp$stats[1]& z1<bp$stats[5], 0, 1) # i.e. suspected outliers...

z1 <- data0[,10]
bp <- boxplot.stats(z1, coef=1.5)
```

```
indicator.1.2 <-ifelse(z1>bp$stats[1]& z1<bp$stats[5], 0, 1)

z1 <- data0[,11]
bp <- boxplot.stats(z1, coef=1.5)
indicator.1.3 <-ifelse(z1>bp$stats[1]& z1<bp$stats[5], 0, 1)

z1 <- data0[,12]
bp <- boxplot.stats(z1, coef=1.5)
indicator.1.4 <-ifelse(z1>bp$stats[1]& z1<bp$stats[5], 0, 1)

z1 <- data0[,13]
bp <- boxplot.stats(z1, coef=1.5)
indicator.1.5 <-ifelse(z1>bp$stats[1]& z1<bp$stats[5], 0, 1)

z1 <- data0[,14]
bp <- boxplot.stats(z1, coef=1.5)
indicator.1.6 <-ifelse(z1>bp$stats[1]& z1<bp$stats[5], 0, 1)

z1 <- data0[,15]
bp <- boxplot.stats(z1, coef=1.5)
indicator.1.7 <-ifelse(z1>bp$stats[1]& z1<bp$stats[5], 0, 1)

z1 <- data0[,16]
bp <- boxplot.stats(z1, coef=1.5)
indicator.1.8 <-ifelse(z1>bp$stats[1]& z1<bp$stats[5], 0, 1)

# This indictor is one value out of eight is an outlier at a given location
indicator.1 <- indicator.1.1+indicator.1.2+indicator.1.3+indicator.1.4+indicator.1.5+indicator.1.6+indicator.1.7+indicator.1.8



# A choropleth map of the boxplot outliers...
shades.1 = shading(c(0,1,9),c("white","yellow","black")) # i.e. yellow - no & black - yes
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,indicator.1,shades.1)
title("Outliers (black): technique 1 - boxplot statistics")
map.scale(1800000,650000,1100000,"x500 km",2,1)
north.arrow(1800000,1000000,80000, col="blue")



####################################################################################
# 2. Outliers via Hawkins' statistic - using chi-squared ######################
####################################################################################




# GW means and GW variances for Hawkin's statistic found
# using the gw.cov function in spgwr with the default bi-square weighting scheme.
# Bandwdth is user-specifed...
bwd.1 <- 0.1 # 10% of nearby data

# Coordinates....
coordinates(data0) <- c("X", "Y") # note do not need to divide by a 1000

# Critical values of the chi-squared distribution
chi_10 <- 2.70554
chi_5 <- 3.84146
chi_2.5 <- 5.02389
chi_1 <- 6.63490
chi_0.5 <- 7.87944
chi_0.01 <- 10.828

# Hawkins test for all 8 unemloyment variables (or years)

gwss <- gw.cov(data0, vars=9, adapt=bwd.1)
GW.mean <- gwss$SDF$mean.V1
GW.variance <- (gwss$SDF$sd.V1)^2
Hawk.N <- bwd.1*length(X) # number of neighbouring data
Hawk.lm <- GW.mean # the local mean at observation points
Hawk.alv <- mean(GW.variance) # the average local variance with same bandwidth
Hawk.test <- (Hawk.N*(UNEMP.R1.2000-Hawk.lm)^2)/((Hawk.N+1)*Hawk.alv)  # test statistic
indicator.2.1 <-ifelse(Hawk.test <=chi_5, 0, 1) # change critical level accordingly...

gwss <- gw.cov(data0, vars=10, adapt=bwd.1)
GW.mean <- gwss$SDF$mean.V1
GW.variance <- (gwss$SDF$sd.V1)^2
Hawk.N <- bwd.1*length(X) # number of neighbouring data
Hawk.lm <- GW.mean # the local mean at observation points
Hawk.alv <- mean(GW.variance) # the average local variance with same bandwidth
Hawk.test <- (Hawk.N*(UNEMP.R1.2001-Hawk.lm)^2)/((Hawk.N+1)*Hawk.alv)  # test statistic
indicator.2.2 <-ifelse(Hawk.test <=chi_5, 0, 1) # change critical level accordingly...

gwss <- gw.cov(data0, vars=11, adapt=bwd.1)
GW.mean <- gwss$SDF$mean.V1
GW.variance <- (gwss$SDF$sd.V1)^2
Hawk.N <- bwd.1*length(X) # number of neighbouring data
Hawk.lm <- GW.mean # the local mean at observation points
Hawk.alv <- mean(GW.variance) # the average local variance with same bandwidth
Hawk.test <- (Hawk.N*(UNEMP.R1.2002-Hawk.lm)^2)/((Hawk.N+1)*Hawk.alv)  # test statistic
indicator.2.3 <-ifelse(Hawk.test <=chi_5, 0, 1) # change critical level accordingly...

gwss <- gw.cov(data0, vars=12, adapt=bwd.1)
GW.mean <- gwss$SDF$mean.V1
GW.variance <- (gwss$SDF$sd.V1)^2
Hawk.N <- bwd.1*length(X) # number of neighbouring data
Hawk.lm <- GW.mean # the local mean at observation points
Hawk.alv <- mean(GW.variance) # the average local variance with same bandwidth
Hawk.test <- (Hawk.N*(UNEMP.R1.2003-Hawk.lm)^2)/((Hawk.N+1)*Hawk.alv)  # test statistic
```

```
indicator.2.4 <-ifelse(Hawk.test <=chi_5, 0, 1) # change critical level accordingly...

gwss <- gw.cov(data0, vars=13, adapt=bwd.1)
GW.mean <- gwss$SDF$mean.V1
GW.variance <- (gwss$SDF$sd.V1)^2
Hawk.N <- bwd.1*length(X) # number of neighbouring data
Hawk.lm <- GW.mean # the local mean at observation points
Hawk.alv <- mean(GW.variance) # the average local variance with same bandwidth
Hawk.test <- (Hawk.N*(UNEMP.R1.2004-Hawk.lm)^2)/((Hawk.N+1)*Hawk.alv)  # test statistic
indicator.2.5 <-ifelse(Hawk.test <=chi_5, 0, 1) # change critical level accordingly...

gwss <- gw.cov(data0, vars=14, adapt=bwd.1)
GW.mean <- gwss$SDF$mean.V1
GW.variance <- (gwss$SDF$sd.V1)^2
Hawk.N <- bwd.1*length(X) # number of neighbouring data
Hawk.lm <- GW.mean # the local mean at observation points
Hawk.alv <- mean(GW.variance) # the average local variance with same bandwidth
Hawk.test <- (Hawk.N*(UNEMP.R1.2005-Hawk.lm)^2)/((Hawk.N+1)*Hawk.alv)  # test statistic
indicator.2.6 <-ifelse(Hawk.test <=chi_5, 0, 1) # change critical level accordingly...

gwss <- gw.cov(data0, vars=15, adapt=bwd.1)
GW.mean <- gwss$SDF$mean.V1
GW.variance <- (gwss$SDF$sd.V1)^2
Hawk.N <- bwd.1*length(X) # number of neighbouring data
Hawk.lm <- GW.mean # the local mean at observation points
Hawk.alv <- mean(GW.variance) # the average local variance with same bandwidth
Hawk.test <- (Hawk.N*(UNEMP.R1.2006-Hawk.lm)^2)/((Hawk.N+1)*Hawk.alv)  # test statistic
indicator.2.7 <-ifelse(Hawk.test <=chi_5, 0, 1) # change critical level accordingly...

gwss <- gw.cov(data0, vars=16, adapt=bwd.1)
GW.mean <- gwss$SDF$mean.V1
GW.variance <- (gwss$SDF$sd.V1)^2
Hawk.N <- bwd.1*length(X) # number of neighbouring data
Hawk.lm <- GW.mean # the local mean at observation points
Hawk.alv <- mean(GW.variance) # the average local variance with same bandwidth
Hawk.test <- (Hawk.N*(UNEMP.R1.2007-Hawk.lm)^2)/((Hawk.N+1)*Hawk.alv)  # test statistic
indicator.2.8 <-ifelse(Hawk.test <=chi_5, 0, 1) # change critical level accordingly...



# This indictor is one value out of eight is an outlier at a given location
indicator.2 <- indicator.2.1+indicator.2.2+indicator.2.3+indicator.2.4+indicator.2.5+indicator.2.6+indicator.2.7+indicator.2.8



# A choropleth map of Hawkins' test results...
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,indicator.2,shades.1)
title("Outliers (black): technique 2 - Hawkins' statistics")
map.scale(1800000,650000,1100000,"x500 km",2,1)
north.arrow(1800000,1000000,80000, col="blue")
```

```
################################################################################
# 3. Time series outliers - using chi-squared #################################
################################################################################



# Simplified version of Claude Grasland's time series outlier identification
# In this case for one, not multiple spatial units...

Data.1.ts <- data1@data[9:16]
row.means <- apply(Data.1.ts,1,mean)
row.vars <- apply(Data.1.ts,1,var)
row.sds <- row.vars^0.5
op.1 <- sweep(Data.1.ts,1,row.means)
resids.ts <- (op.1)^2/row.vars
# Note the use of variances - thus we do a chi-square test here instead of looking at boxplot statistics
#sort(resids.3)
quant.ts <- quantile(resids.ts, probs = seq(0, 1, 0.01))
threshold.ts <- quant.ts[100] # in this case the 99%tile

# Assume Gaussianity and do a Chi-sqaured test...
threshold.tsx <- 3.84146 # chi-square test at 95% ...
indicator.tsx <-ifelse(resids.ts>0& resids.ts<threshold.tsx, 0, 1) # i.e. suspected outliers...
indicator.3 <- apply(indicator.tsx,1,sum)

# A choropleth map of the time series outliers
shades.2 = shading(c(0,1,2),c("white","yellow","black")) # i.e. yellow - no & black - yes
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,indicator.3,shades.2)
title("Outliers (black): technique 3 - time series statistics")
map.scale(1800000,650000,1100000,"x500 km",2,1)
north.arrow(1800000,1000000,80000, col="blue")


################################################################################
# 4. Outliers via MLR  - via boxplots ##########################################
################################################################################



# In all regressions, we use unemployment from the two closest years to explain unemployment for a given year of interest.

# Note using Boxplot statistics for residuals

# Need to do this eight times...
# one for each unemployment variable & changing the independent variables accordingly
```

```
mlr <- lm(UNEMP.R1.2000 ~ UNEMP.R1.2001+UNEMP.R1.2002)
raw.resids.mlr <- UNEMP.R1.2000-mlr$fitted # Actual minus prediction (raw residuals)
# Identifying & updating outlier information in one file using Boxplot statistics for residuals
bp <- boxplot.stats(raw.resids.mlr, coef=1.5)
indicator.4.1 <-ifelse(raw.resids.mlr>bp$stats[1]& raw.resids.mlr<bp$stats[5], 0, 1)

mlr <- lm(UNEMP.R1.2001 ~ UNEMP.R1.2000+UNEMP.R1.2002)
raw.resids.mlr <- UNEMP.R1.2001-mlr$fitted
bp <- boxplot.stats(raw.resids.mlr, coef=1.5)
indicator.4.2 <-ifelse(raw.resids.mlr>bp$stats[1]& raw.resids.mlr<bp$stats[5], 0, 1)

mlr <- lm(UNEMP.R1.2002 ~ UNEMP.R1.2001+UNEMP.R1.2003)
raw.resids.mlr <- UNEMP.R1.2002-mlr$fitted
bp <- boxplot.stats(raw.resids.mlr, coef=1.5)
indicator.4.3 <-ifelse(raw.resids.mlr>bp$stats[1]& raw.resids.mlr<bp$stats[5], 0, 1)

mlr <- lm(UNEMP.R1.2003 ~ UNEMP.R1.2002+UNEMP.R1.2004)
raw.resids.mlr <- UNEMP.R1.2003-mlr$fitted
bp <- boxplot.stats(raw.resids.mlr, coef=1.5)
indicator.4.4 <-ifelse(raw.resids.mlr>bp$stats[1]& raw.resids.mlr<bp$stats[5], 0, 1)

mlr <- lm(UNEMP.R1.2004 ~ UNEMP.R1.2003+UNEMP.R1.2005)
raw.resids.mlr <- UNEMP.R1.2004-mlr$fitted
bp <- boxplot.stats(raw.resids.mlr, coef=1.5)
indicator.4.5 <-ifelse(raw.resids.mlr>bp$stats[1]& raw.resids.mlr<bp$stats[5], 0, 1)

mlr <- lm(UNEMP.R1.2005 ~ UNEMP.R1.2004+UNEMP.R1.2006)
raw.resids.mlr <- UNEMP.R1.2005-mlr$fitted
bp <- boxplot.stats(raw.resids.mlr, coef=1.5)
indicator.4.6 <-ifelse(raw.resids.mlr>bp$stats[1]& raw.resids.mlr<bp$stats[5], 0, 1)

mlr <- lm(UNEMP.R1.2006 ~ UNEMP.R1.2005+UNEMP.R1.2007)
raw.resids.mlr <- UNEMP.R1.2006-mlr$fitted
bp <- boxplot.stats(raw.resids.mlr, coef=1.5)
indicator.4.7 <-ifelse(raw.resids.mlr>bp$stats[1]& raw.resids.mlr<bp$stats[5], 0, 1)

mlr <- lm(UNEMP.R1.2007 ~ UNEMP.R1.2005+UNEMP.R1.2006)
raw.resids.mlr <- UNEMP.R1.2007-mlr$fitted
bp <- boxplot.stats(raw.resids.mlr, coef=1.5)
indicator.4.8 <-ifelse(raw.resids.mlr>bp$stats[1]& raw.resids.mlr<bp$stats[5], 0, 1)

# This indictor is one value out of eight is an outlier at a given location
indicator.4 <- indicator.4.1+indicator.4.2+indicator.4.3+indicator.4.4+indicator.4.5+indicator.4.6+indicator.4.7+indicator.4.8

# A choropleth map of MLR outliers...
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,indicator.4,shades.1)
title("Outliers (black): technique 4 - Multiple linear regression")
map.scale(1800000,650000,1100000,"x500 km",2,1)
```

```
north.arrow(1800000,1000000,80000, col="blue")


#################################################################################
# 5. Outliers via LWR  - via boxplots #########################################
#################################################################################



# Using locfit...
require(locfit)
# Ignore warning message...

# In this case - we only use the closest year to the dependent variable, as too many variables can create problems...
# Automatic bandwidth selection also was problematic in some cases...

# Automatic bandwidth for a non-robust LR (i.e. not a lowess fit) using generalised cross-validation approach.
#summary(gcvplot(UNEMP.R1.2000 ~ UNEMP.R1.2001+UNEMP.R1.2002,data=data0, scale=F,alpha=seq(0.1,1,by=0.1),
#deg=1,kern="tricube",lfproc=locfit.raw))

# As such - the same user-specified bandwidth is used for all eight fits
bwd.2 <- 0.4

# The eight LWR model fits...

lwr <- locfit(UNEMP.R1.2000 ~ UNEMP.R1.2001,data=data0, scale=F, alpha=bwd.2,
deg=1,kern="tricube",lfproc=locfit.raw)
lwr.p <- fitted.locfit(lwr)
raw.resids.lr <- UNEMP.R1.2000-lwr.p # Actual minus prediction
bp <- boxplot.stats(raw.resids.lr, coef=1.5)
indicator.5.1 <-ifelse(raw.resids.lr>bp$stats[1]& raw.resids.lr<bp$stats[5], 0, 1)

lwr <- locfit(UNEMP.R1.2001 ~ UNEMP.R1.2002,data=data0, scale=F, alpha=bwd.2,
deg=1,kern="tricube",lfproc=locfit.raw)
lwr.p <- fitted.locfit(lwr)
raw.resids.lr <- UNEMP.R1.2001-lwr.p # Actual minus prediction
bp <- boxplot.stats(raw.resids.lr, coef=1.5)
indicator.5.2 <-ifelse(raw.resids.lr>bp$stats[1]& raw.resids.lr<bp$stats[5], 0, 1)

lwr <- locfit(UNEMP.R1.2002 ~ UNEMP.R1.2003,data=data0, scale=F, alpha=bwd.2,
deg=1,kern="tricube",lfproc=locfit.raw)
lwr.p <- fitted.locfit(lwr)
raw.resids.lr <- UNEMP.R1.2002-lwr.p # Actual minus prediction
bp <- boxplot.stats(raw.resids.lr, coef=1.5)
indicator.5.3 <-ifelse(raw.resids.lr>bp$stats[1]& raw.resids.lr<bp$stats[5], 0, 1)

lwr <- locfit(UNEMP.R1.2003 ~ UNEMP.R1.2004,data=data0, scale=F, alpha=bwd.2,
deg=1,kern="tricube",lfproc=locfit.raw)
lwr.p <- fitted.locfit(lwr)
raw.resids.lr <- UNEMP.R1.2003-lwr.p # Actual minus prediction
```

```
bp <- boxplot.stats(raw.resids.lr, coef=1.5)
indicator.5.4 <-ifelse(raw.resids.lr>bp$stats[1]& raw.resids.lr<bp$stats[5], 0, 1)

lwr <- locfit(UNEMP.R1.2004 ~ UNEMP.R1.2005,data=data0, scale=F, alpha=bwd.2,
deg=1,kern="tricube",lfproc=locfit.raw)
lwr.p <- fitted.locfit(lwr)
raw.resids.lr <- UNEMP.R1.2004-lwr.p # Actual minus prediction
bp <- boxplot.stats(raw.resids.lr, coef=1.5)
indicator.5.5 <-ifelse(raw.resids.lr>bp$stats[1]& raw.resids.lr<bp$stats[5], 0, 1)

lwr <- locfit(UNEMP.R1.2005 ~ UNEMP.R1.2006,data=data0, scale=F, alpha=bwd.2,
deg=1,kern="tricube",lfproc=locfit.raw)
lwr.p <- fitted.locfit(lwr)
raw.resids.lr <- UNEMP.R1.2005-lwr.p # Actual minus prediction
bp <- boxplot.stats(raw.resids.lr, coef=1.5)
indicator.5.6 <-ifelse(raw.resids.lr>bp$stats[1]& raw.resids.lr<bp$stats[5], 0, 1)

lwr <- locfit(UNEMP.R1.2006 ~ UNEMP.R1.2007,data=data0, scale=F, alpha=bwd.2,
deg=1,kern="tricube",lfproc=locfit.raw)
lwr.p <- fitted.locfit(lwr)
raw.resids.lr <- UNEMP.R1.2006-lwr.p # Actual minus prediction
bp <- boxplot.stats(raw.resids.lr, coef=1.5)
indicator.5.7 <-ifelse(raw.resids.lr>bp$stats[1]& raw.resids.lr<bp$stats[5], 0, 1)

lwr <- locfit(UNEMP.R1.2007 ~ UNEMP.R1.2006,data=data0, scale=F, alpha=bwd.2,
deg=1,kern="tricube",lfproc=locfit.raw)
lwr.p <- fitted.locfit(lwr)
raw.resids.lr <- UNEMP.R1.2007-lwr.p # Actual minus prediction
bp <- boxplot.stats(raw.resids.lr, coef=1.5)
indicator.5.8 <-ifelse(raw.resids.lr>bp$stats[1]& raw.resids.lr<bp$stats[5], 0, 1)


# This indictor is one value out of eight is an outlier at a given location
indicator.5 <- indicator.5.1+indicator.5.2+indicator.5.3+indicator.5.4+indicator.5.5+indicator.5.6+indicator.5.7+indicator.5.8

# A choropleth map of the LWR outliers...
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,indicator.5,shades.1)
title("Outliers (black): technique 5 - Locally weighted regression")
map.scale(1800000,650000,1100000,"x500 km",2,1)
north.arrow(1800000,1000000,80000, col="blue")


####################################################################################
# 6. Outliers via GWR  - via boxplots ##########################################
####################################################################################

# Using spgwr...
# As with MLR - we again use unemployment from the two closest years to explain unemployment for a given year of interest.
# In this case - all eight GWR fits have an optimally found bandwidth (by cross-validation)

# The eight GWR model fits...

gwr.cv.bwd <-gwr.sel(UNEMP.R1.2000 ~ UNEMP.R1.2001+UNEMP.R1.2002,
data=data0,adapt=TRUE,gweight=gwr.bisquare, method="cv")
gwr.p <- gwr(UNEMP.R1.2000 ~ UNEMP.R1.2001+UNEMP.R1.2002,data=data0,adapt=gwr.cv.bwd[1],
gweight=gwr.bisquare,predictions=T)
raw.resids.gwr <- UNEMP.R1.2000-gwr.p$SDF$pred
bp <- boxplot.stats(raw.resids.gwr, coef=1.5)
indicator.6.1 <-ifelse(raw.resids.gwr>bp$stats[1]& raw.resids.gwr<bp$stats[5], 0, 1)

gwr.cv.bwd <-gwr.sel(UNEMP.R1.2001 ~ UNEMP.R1.2000+UNEMP.R1.2002,
data=data0,adapt=TRUE,gweight=gwr.bisquare, method="cv")
gwr.p <- gwr(UNEMP.R1.2001 ~ UNEMP.R1.2000+UNEMP.R1.2002,data=data0,adapt=gwr.cv.bwd[1],
gweight=gwr.bisquare,predictions=T)
raw.resids.gwr <- UNEMP.R1.2001-gwr.p$SDF$pred
bp <- boxplot.stats(raw.resids.gwr, coef=1.5)
indicator.6.2 <-ifelse(raw.resids.gwr>bp$stats[1]& raw.resids.gwr<bp$stats[5], 0, 1)

gwr.cv.bwd <-gwr.sel(UNEMP.R1.2002 ~ UNEMP.R1.2001+UNEMP.R1.2003,
data=data0,adapt=TRUE,gweight=gwr.bisquare, method="cv")
gwr.p <- gwr(UNEMP.R1.2002 ~ UNEMP.R1.2001+UNEMP.R1.2003,data=data0,adapt=gwr.cv.bwd[1],
gweight=gwr.bisquare,predictions=T)
raw.resids.gwr <- UNEMP.R1.2002-gwr.p$SDF$pred
bp <- boxplot.stats(raw.resids.gwr, coef=1.5)
indicator.6.3 <-ifelse(raw.resids.gwr>bp$stats[1]& raw.resids.gwr<bp$stats[5], 0, 1)

gwr.cv.bwd <-gwr.sel(UNEMP.R1.2003 ~ UNEMP.R1.2002+UNEMP.R1.2004,
data=data0,adapt=TRUE,gweight=gwr.bisquare, method="cv")
gwr.p <- gwr(UNEMP.R1.2003 ~ UNEMP.R1.2002+UNEMP.R1.2004,data=data0,adapt=gwr.cv.bwd[1],
gweight=gwr.bisquare,predictions=T)
raw.resids.gwr <- UNEMP.R1.2003-gwr.p$SDF$pred
bp <- boxplot.stats(raw.resids.gwr, coef=1.5)
indicator.6.4 <-ifelse(raw.resids.gwr>bp$stats[1]& raw.resids.gwr<bp$stats[5], 0, 1)

gwr.cv.bwd <-gwr.sel(UNEMP.R1.2004 ~ UNEMP.R1.2003+UNEMP.R1.2005,
data=data0,adapt=TRUE,gweight=gwr.bisquare, method="cv")
gwr.p <- gwr(UNEMP.R1.2004 ~ UNEMP.R1.2003+UNEMP.R1.2005,data=data0,adapt=gwr.cv.bwd[1],
gweight=gwr.bisquare,predictions=T)
raw.resids.gwr <- UNEMP.R1.2004-gwr.p$SDF$pred
bp <- boxplot.stats(raw.resids.gwr, coef=1.5)
indicator.6.5 <-ifelse(raw.resids.gwr>bp$stats[1]& raw.resids.gwr<bp$stats[5], 0, 1)

gwr.cv.bwd <-gwr.sel(UNEMP.R1.2005 ~ UNEMP.R1.2004+UNEMP.R1.2006,
data=data0,adapt=TRUE,gweight=gwr.bisquare, method="cv")
gwr.p <- gwr(UNEMP.R1.2005 ~ UNEMP.R1.2004+UNEMP.R1.2006,data=data0,adapt=gwr.cv.bwd[1],
gweight=gwr.bisquare,predictions=T)
raw.resids.gwr <- UNEMP.R1.2005-gwr.p$SDF$pred
```

```
bp <- boxplot.stats(raw.resids.gwr, coef=1.5)
indicator.6.6 <-ifelse(raw.resids.gwr>bp$stats[1]& raw.resids.gwr<bp$stats[5], 0, 1)

gwr.cv.bwd <-gwr.sel(UNEMP.R1.2006 ~ UNEMP.R1.2005+UNEMP.R1.2007,
data=data0,adapt=TRUE,gweight=gwr.bisquare, method="cv")
gwr.p <- gwr(UNEMP.R1.2006 ~ UNEMP.R1.2005+UNEMP.R1.2007,data=data0,adapt=gwr.cv.bwd[1],
gweight=gwr.bisquare,predictions=T)
raw.resids.gwr <- UNEMP.R1.2006-gwr.p$SDF$pred
bp <- boxplot.stats(raw.resids.gwr, coef=1.5)
indicator.6.7 <-ifelse(raw.resids.gwr>bp$stats[1]& raw.resids.gwr<bp$stats[5], 0, 1)

gwr.cv.bwd <-gwr.sel(UNEMP.R1.2007 ~ UNEMP.R1.2005+UNEMP.R1.2006,
data=data0,adapt=TRUE,gweight=gwr.bisquare, method="cv")
gwr.p <- gwr(UNEMP.R1.2007 ~ UNEMP.R1.2005+UNEMP.R1.2006,data=data0,adapt=gwr.cv.bwd[1],
gweight=gwr.bisquare,predictions=T)
raw.resids.gwr <- UNEMP.R1.2007-gwr.p$SDF$pred
bp <- boxplot.stats(raw.resids.gwr, coef=1.5)
indicator.6.8 <-ifelse(raw.resids.gwr>bp$stats[1]& raw.resids.gwr<bp$stats[5], 0, 1)



# This indictor is one value out of eight is an outlier at a given location
indicator.6 <- indicator.6.1+indicator.6.2+indicator.6.3+indicator.6.4+indicator.6.5+indicator.6.6+indicator.6.7+indicator.6.8

# A choropleth map of GWR outliers...
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,indicator.6,shades.1)
title("Outliers (black): technique 6 - Geographically weighted regression")
map.scale(1800000,650000,1100000,"x500 km",2,1)
north.arrow(1800000,1000000,80000, col="blue")




#################################################################################
# 7. PCA outliers  - via boxplots #############################################
#################################################################################




# For PCA, LWPCA and GWPCA we need to define k the number of components to 'keep'....
# In this case 2 or 3 seems ideal - can check this by cross-validation...
# This part of the PCA/LWPCA/GWPCA approach to outlier detection needs further work...

# Lets choose k=3...
comp.keep <- 3

# Global PCA result using the gwpca function with a very large fixed bandwidth...
pca.result <- gwpca(Data.1.scaled,Nuts23Coords,bw=100000000000000000,k=comp.keep,verbose=TRUE)

# Identify some outliers
```

```
resids.pca <- sqrt(gwpca.cv.contrib(Data.1.scaled, Nuts23Coords, pca.result$bw,k=comp.keep))
#sort(resids.pca)
quant.pca <- quantile(resids.pca, probs = seq(0, 1, 0.01))
threshold.pca <- quant.pca[96] # in this case the 95%tile

# Standard Boxplot statistics for resids.pca
bp <- boxplot.stats(resids.pca, coef=1.5)
bp$stats
bp$stats[1]
bp$stats[5]
bp$conf
sort(bp$out)
length(bp$out)

# Identifying & updating outlier information in one file
indicator.7 <-ifelse(resids.pca>bp$stats[1]& resids.pca<bp$stats[5], 0, 1) # i.e. suspected outliers...

# A choropleth map of the PCA outliers...
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,indicator.7,shades.2)
title("Outliers (black): technique 7 - PCA")
map.scale(1800000,650000,1100000,"x500 km",2,1)
north.arrow(1800000,1000000,80000, col="blue")


################################################################################
# 8. LWPCA outliers  - via boxplots ########################################
################################################################################



# Automatically finding the optimal adaptive bandwidth by cross-validation...
lwpca.result <- lwpca.autobw.by.nn(Data.1.scaled,c(100,790),k=comp.keep,verbose=TRUE)

# Some output
#lwpca.result$bw
#lwpca.result$var
#lwpca.result$loadings

# Identify some outliers
resids.lwpca <- sqrt(lwpca.cv.contrib(Data.1.scaled, lwpca.result$bw,k=comp.keep))
quant.lwpca <- quantile(resids.lwpca, probs = seq(0, 1, 0.01))
threshold.lwpca <- quant.lwpca[96]

# Standard Boxplot statistics for resids.lwpca
bp <- boxplot.stats(resids.lwpca, coef=1.5)
bp$stats
bp$stats[1]
bp$stats[5]
```

```
bp$conf
sort(bp$out)
length(bp$out)

# Identifying & updating outlier information in one file
indicator.8 <-ifelse(resids.lwpca>bp$stats[1]& resids.lwpca<bp$stats[5], 0, 1) # i.e. suspected outliers...

# A choropleth map of the LWPCA outliers...
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,indicator.8,shades.2)
title("Outliers (black): technique 8 - Locally weighted PCA")
map.scale(1800000,650000,1100000,"x500 km",2,1)
north.arrow(1800000,1000000,80000, col="blue")



##################################################################################
# 9. GWPCA outliers - via boxplots ############################################
##################################################################################



# Automatically finding the optimal adaptive bandwidth by cross-validation...
gwpca.result <- gwpca.autobw.by.nn(Data.1.scaled,Nuts23Coords,c(100,790),k=comp.keep,verbose=TRUE)

# Some output
#gwpca.result$bw
#gwpca.result$var
#gwpca.result$loadings

# Identify some outliers
resids.gwpca <- sqrt(gwpca.cv.contrib(Data.1.scaled, Nuts23Coords, gwpca.result$bw,k=comp.keep))
quant.gwpca <- quantile(resids.gwpca, probs = seq(0, 1, 0.01))
threshold.gwpca <- quant.gwpca[96]

# Standard Boxplot statistics for resids.gwpca
bp <- boxplot.stats(resids.gwpca, coef=1.5)
bp$stats
bp$stats[1]
bp$stats[5]
bp$conf
sort(bp$out)
length(bp$out)

# Identifying & updating outlier information in one file
indicator.9 <-ifelse(resids.gwpca>bp$stats[1]& resids.gwpca<bp$stats[5], 0, 1) # i.e. suspected outliers...


# A choropleth map of the GWPCA outliers...
```

```
shades.4 = shading(c(0,1,2),c("white","yellow","black")) # i.e. yellow - no & black - yes
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,indicator.9,shades.2)
title("Outliers (black): technique 9 - Geographically weighted PCA")
map.scale(1800000,650000,1100000,"x500 km",2,1)
north.arrow(1800000,1000000,80000, col="blue")


#################################################################################
# C. Linking all data together for weight of evidence map #####################
#################################################################################



data.out <- cbind(indicator.1, indicator.2, indicator.3,indicator.4, indicator.5, indicator.6,
indicator.7, indicator.8, indicator.9, data1@data)

# To a text file
#write.table(data.out,"WoE outliers 1.txt", col.names=T,row.names=F)

data.out.1 <- cbind(X, Y, Order_ID, indicator.1, indicator.2, indicator.3,
indicator.4, indicator.5, indicator.6,indicator.7, indicator.8, indicator.9)

# To a text file
#write.table(data.out.1,"WoE outliers indicators 1.txt", col.names=T,row.names=F)

max(indicator.1)
max(indicator.2)
max(indicator.3)
max(indicator.4)
max(indicator.5)
max(indicator.6)
max(indicator.7)
max(indicator.8)
max(indicator.9)

indicator.1a <- indicator.1 >= 1
indicator.2a <- indicator.2 >= 1
indicator.4a <- indicator.4 >= 1
indicator.5a <- indicator.5 >= 1
indicator.6a <- indicator.6 >= 1

max(indicator.1a)
max(indicator.2a)
max(indicator.3)
max(indicator.4a)
max(indicator.5a)
max(indicator.6a)
max(indicator.7)
max(indicator.8)
max(indicator.9)


# Put all indicator data together...
indicator.10 <- indicator.1a+indicator.2a+indicator.3+indicator.4a+indicator.5a+indicator.6a+indicator.7+indicator.8+indicator.9

summary(indicator.10)
# Histogram
X11(width=5.3,height=5.7)
hist(indicator.10,br=c(0,1,2,3,4,5,6,7,8,9), main="Distribution of 'weight of evidence'",
xlab="Indicator sum")

# A choropleth map of suspected outliers...
shades.7 = shading(c(1,3,5,7,9),c("white","yellow","orange","red","dark red","black"))
X11(width=8,height=7)
par(mar=c(0,0,2,0))
choropleth(data1,indicator.10,shades.7)
title("Suspected outliers - weak to strong (yellow to black) evidence")
choro.legend(-2400000,2200000,shades.7,
over="exactly", between="to under",
fmt="%4.0f",title='Indicator sum (max.: 9)',cex=0.8)
map.scale(1800000,650000,1100000,"x500 km",2,1)
north.arrow(1800000,1400000,80000, col="blue")
text(-1900000,2400000, "'Weight of evidence'", cex=1, col=3)
```