**R Codes for paper:**

**Natural spatial pattern–when mutual distances between cities follow Benford's law**

**Katarzyna Kopczewska, Tomasz Kopczewski**

**University of Warsaw, Faculty of Economic Sciences**

[**kkopczewska@wne.uw.edu.pl**](mailto:kkopczewska@wne.uw.edu.pl)

Spis treści

[**# 1. General code – packages, datasets, maps** 3](#_Toc111409111)

[# Option A - reading ready data from files (to skip calculations) 3](#_Toc111409112)

[# Option B - Reading raw data and maps from packages (to run calculations) 4](#_Toc111409113)

[**# 2. Benford, Zipf & Clark-Evans test for countries** 5](#_Toc111409114)

[# Calculations of Benford conformity, Zipf and Clark-Evans test by countries 5](#_Toc111409115)

[# Figure 1 – World map of Benford conformity 11](#_Toc111409116)

[# Figure 3 – World map of Clark-Evans test 12](#_Toc111409117)

[# Figure 4 – World map of Zipf coefficient 13](#_Toc111409118)

[# Figure 7 – Locations of cities in selected countries 14](#_Toc111409119)

[# USA 14](#_Toc111409120)

[# China 15](#_Toc111409121)

[# Poland 16](#_Toc111409122)

[# Italy 17](#_Toc111409123)

[# Figure 2 – Boxplots of Benford conformity vs number of cities 18](#_Toc111409124)

[# Figure 6 – Relations in empirical data 19](#_Toc111409125)

[# Table 1 – Benford conformity by continents 20](#_Toc111409126)

[# Table 2 – Benford conformity vs point patterns in countries 21](#_Toc111409127)

[**# 3. Simulation of point patterns to get a Benford-like mixture** 22](#_Toc111409128)

[# Colours 22](#_Toc111409129)

[# Bounding box with planar coordinates 22](#_Toc111409130)

[# Simulation of pure point patterns and tests for Benford conformity 23](#_Toc111409131)

[# random pattern 23](#_Toc111409132)

[# regular (ordered, uniform) pattern 24](#_Toc111409133)

[# clustered (agglomerated) pattern 25](#_Toc111409134)

[# Figure 8a - mixed point patterns (with clustered pattern centred) 26](#_Toc111409135)

[# Figure 8b – Spatial and Benford distributions in pure patterns 29](#_Toc111409136)

[# mixed point patterns (with clustered pattern shifted/skewed) 29](#_Toc111409137)

[# Figure 8c - density plot of mutual distances in all point-patterns 30](#_Toc111409138)

[# Figure 8d - all Benford distributions together 32](#_Toc111409139)

[# Simulation of a mixture of point patterns 33](#_Toc111409140)

[# Figure 9 – Boxplots of simulated mixtures of point patterns and their Benford conformity 35](#_Toc111409141)

[**# 4. Fitting distributions to population data** 36](#_Toc111409142)

[# Figure 5a - fitting for 0%-90% centiles - Gamma (1, 1) distribution 36](#_Toc111409143)

[# Figure 5b - fitting for 90%-99% centiles - Gamma (1, 1) distribution 38](#_Toc111409144)

[# Figure 5c - fitting for 99%-100% centiles - Gamma (1, 1) distribution 39](#_Toc111409145)

[# combination of three Gamma(1,1) distributions 40](#_Toc111409146)

[# Benford conformity of combination of three Gamma(1,1) distributions - (xi-1) transformation 42](#_Toc111409147)

[# Benford conformity of combination of three Gamma(1,1) distributions - no transformation 43](#_Toc111409148)

[# Benford conformity of simulated population 44](#_Toc111409149)

[**# 5. 3D simulation – spatial patterns, population, values allocation – to best Benford** 45](#_Toc111409150)

[# simulation of complex patterns 45](#_Toc111409151)

[# Figure 10 - relations in simulated data - between MAD statistics of 1D, 2D and 3D distances 48](#_Toc111409152)

# **# 1. General code – packages, datasets, maps**

**# Document below presents all codes necessary to replicate**

**# all analyses and simulations presented in the paper.**

**# Use R software – those calculations were checked in version 4.2.1.**

# required packages for the whole study

library(sp) # for spatial analysis

library(rgdal)

library(maptools)

library(sp)

library(spatstat)

library(SpatialEpi)

library(raster)

library(rgeos)

library(maps) # for spatial data – dataset and maps

library(rworldmap)

library(benford.analysis) # for Benford analysis

library(viridis) # for graphics

library(scales)

library(RColorBrewer)

library(classInt)

library(doBy)

library(GGally)

**# The code below allows for full replication of the study,**

**# starting with a raw dataset on population in cities worldwide.**

**# However, those calculations can be skipped as they were made available**

**# in files at Figshare <https://figshare.com/account/home#/collections/6137520>**

**# The same refers to simulations – simulation results are available in files**

**# at <https://github.com/kkopczewska/spatial-Benford>**

**# For simplicity, the objects – calculated and uploaded – have the same names.**

**# Please choose one of the options:**

**# A - upload in R the data from Figshare**

**# B - replicate all computations on raw data**

## # Option A - reading ready data from files (to skip calculations)

**# 🡪🡪🡪 For analytical part**

**# download data from Figshare**

**# <https://figshare.com/account/home#/collections/6137520>**

**# set your own address of the folder where you store the downloaded files**

**# setting Working Directory**

setwd("C:/mydata") # change into your Working Directory

**# dataset from maps:: package on cities – with population and geo-coordinates**

**# original dataset which was used for all studies below**

**# typically available in maps:: package**

world.cities<-read.table("PART1.txt", sep=" ", dec=".", header=TRUE)

**# calculations of Benford, Zipf and Clark-Evans for cities**

**# analysis for 169 countries**

wynik.df<-read.table("PART2.txt", sep=" ", dec=".", header=TRUE)

**# 🡪🡪🡪 For simulations**

**# download data from Github**

**# <https://github.com/kkopczewska/spatial-Benford>**

**# set your own address of the folder where you store the downloaded files**

**# setting Working Directory**

setwd("C:/mydata") # change into your Working Directory

**# all files below can be self-generated based on codes provided further**

**# simulations might be very time-consuming; importing files is quicker**

**# simulations may differ slightly, as no (set seed) was given**

**# results of simulation on theoretical spatial distributions**

**# dataset includes all three fractions of pattern >0 only**

data.all<-read.table("theor\_spat\_sim.txt", sep=" ", dec=".", header=TRUE)

**# results of simulation of complex 3D model**

**# values assigned to locations logically**

beny.all.logic<-read.table("sim\_model\_all\_2.txt", sep=" ", dec=".", header=TRUE)

**# results of simulation of complex 3D model**

**# values assigned to locations randomly**

beny.all.random<-read.table("sim\_model\_all\_3.txt", sep=" ", dec=".", header=TRUE)

**# contour maps of countries – from rworldmap:: package**

worldMap<-getMap()

mapa<-as.data.frame(worldMap)

head(mapa) # names of countries in maps

## # Option B - Reading raw data and maps from packages (to run calculations)

# list of cities worldwide with geo-coordinates - from maps:: package

head(world.cities) # see the dataset

list.countries<-unique(world.cities$country.etc)

list.countries # names of countries in dataset of cities

# checking the number of cities within each country

world.cities$ones<-rep(1, times=dim(world.cities)[1])

agg<-aggregate(world.cities$ones, by=list(world.cities$country.etc), sum)

agg # number of cities in each country (from the dataset)

# contour maps of countries – from rworldmap:: package

worldMap<-getMap()

mapa<-as.data.frame(worldMap)

head(mapa) # names of countries in maps

# **# 2. Benford, Zipf & Clark-Evans test for countries**

## # Calculations of Benford conformity, Zipf and Clark-Evans test by countries

# code below is quite long, as it includes all individual corrections

# corrections are needed mostly because of two reasons

# first reason: incompatible names of countries in maps and cities datasets

# second reason: corrections of territories (mainly islands)

# due to ongoing political changes, names of countries may change

# if changes in file with cities are incompatible with changes in file with map

# the code may stop

**# NOTE: a result of that analysis can be imported as a file (see reading data)**

wynik<-matrix(0, nrow=169, ncol=17)

continents<-c(rep("Europe", times=41), rep("Asia", times=44), rep("Africa", times=48), rep("America South", times=13), rep("America North", times=16), rep("Australia", times=7))

vec<-c("Spain", "Portugal", "Norway", "Netherlands", "Denmark", "Germany", "Switzerland", "Estonia", "Belgium", "Finland", "Hungary", "Turkey", "Italy", "France", "Ireland", "UK", "Czech Republic", "Romania", "Austria", "Luxembourg", "Cyprus", "Serbia", "Greece", "Croatia", "Bulgaria", "Latvia", "Slovenia", "Sweden", "Lithuania", "Iceland", "Ukraine", "Poland", "Moldova", "North Macedonia", "Belarus", "Malta", "Albania", "Bosnia and Herzegovina", "Slovakia", "Greenland", "Faroe Islands", "Palestine", "Pakistan", "Kuwait", "Iraq", "Saudi Arabia", "Syria", "Israel", "Iran", "Jordan", "Lebanon", "Bahrain", "Afghanistan", "Yemen", "Georgia", "Japan", "Kazakhstan", "Oman", "Indonesia", "Bangladesh", "India", "Armenia", "Qatar", "Philippines", "Russia", "China", "Tajikistan", "Korea South", "Azerbaijan", "East Timor", "Kyrgyzstan", "Turkmenistan", "Uzbekistan", "Myanmar", "Malaysia", "Mongolia", "Nepal", "Sri Lanka", "Thailand", "Korea North", "Bhutan", "Vietnam", "Laos", "Cambodia", "Taiwan", "Somalia", "Ethiopia", "Sudan", "Tunisia", "Algeria", "Mauritania", "Nigeria", "Niger", "Chad", "Ivory Coast", "Egypt", "Benin", "Ghana", "Gambia", "Equatorial Guinea", "Cameroon", "Eritrea", "South Africa", "Uganda", "Morocco", "Kenya", "Libya", "Congo Democratic Republic", "Mauritius", "Sierra Leone", "Central African Republic", "Madagascar", "Togo", "Mozambique", "Guyana", "Namibia", "Mali", "Botswana", "Tanzania", "Cape Verde", "Guinea-Bissau", "Senegal", "Malawi", "Burkina Faso", "Zimbabwe", "Angola", "Guinea", "Swaziland", "Gabon", "Comoros", "Congo", "Burundi", "Zambia", "Brazil", "Paraguay", "Peru", "Bolivia", "Colombia", "French Guiana", "Venezuela", "Uruguay", "Dominican Republic", "Ecuador", "Suriname", "Argentina", "Chile", "Mexico", "Jamaica", "Cuba", "Haiti", "El Salvador", "Guatemala", "Honduras", "Nicaragua", "Panama", "USA", "Puerto Rico", "Costa Rica", "Trinidad and Tobago", "Dominica", "Saint Kitts and Nevis", "Canada", "Australia", "Samoa", "American Samoa", "Guam", "New Zealand", "Papua New Guinea", "New Caledonia")

colnames(wynik)<-c("country", "continent", "name.as.map", "tot.pop", "how.many.cities", "coefvar.popul", "Zipf", "Benford.dist", "Benford.pop.vec", "Benford.pop.dist", "Benford.socgeo.dist", "RofClarkEvans", "ClarkEvans.pvalue", "MAD.dist", "MAD.pop.vec", "MAD.pop.dist", "MAD.socgeo.dist")

wynik[,2]<-continents # column 2 "continent"

################

for(i in 1:169){

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,1]<-vec[i] # column "country"

pop<-world.cities$pop[world.cities$country.etc==vec[i]]

wynik[i,3]<-vec[i] # column "name.as.map"

wynik[i,4]<-sum(pop) # column "tot.pop"

wynik[i,5]<-dim(aaa)[1] # column "how.many.cities"

wynik[i,6]<-round(sd(pop)/mean(pop),4) # column "coef.var.popul"

zero<-which(aaa[,3]==0) # Zipf

aaa[zero,3]<-quantile(aaa[,3], probs=0.1)

aaa<-orderBy(~-pop, data=aaa)

jj<-data.frame(pop=aaa[,3], rank=1:dim(aaa)[1])

jj$log.pop<-log(jj$pop) # column 3

jj$log.rank<-log(jj$rank) # column 4

reg<-lm(jj[,4]~jj[,3]) # Pareto form

a<-summary(reg)

wynik[i,7]<-as.numeric(reg$coefficients[2]) # column "Zipf"

aaa<-world.cities[world.cities$country.etc==vec[i],c(5:4)]

aaa.sp<-aaa

coordinates(aaa.sp)<-c("long", "lat")

dist<-matrix(0, nrow=dim(aaa)[1], ncol=dim(aaa)[1])

for(j in 1:dim(aaa)[1]){

dist[,j]<-spDistsN1(aaa.sp, aaa.sp[j,], longlat=TRUE)}

dist.aaa.v<-as.vector(dist)

bb.dist<-benford(dist.aaa.v)

wynik[i,8]<-bb.dist$MAD.conformity # column "Benford.dist"

wynik[i,14]<-bb.dist$MAD # column "MAD.dist"

bb.pop.vec<-benford(pop)

wynik[i,9]<-bb.pop.vec$MAD.conformity # column "Benford.pop.vec"

wynik[i,15]<-bb.pop.vec$MAD # column "MAD.pop.vec"

dist.pop<-as.vector(as.matrix(dist(as.data.frame(pop))))

bb.pop.dist<-benford(dist.pop)

wynik[i,10]<-bb.pop.dist$MAD.conformity # column "Benford.pop.dist"

wynik[i,16]<-bb.pop.dist$MAD # column "MAD.pop.dist"

aaa<-world.cities[world.cities$country.etc==vec[i],c(3:5)]

dist.socgeo<- as.vector(as.matrix(dist(as.data.frame(aaa))))

bb.socgeo.dist<-benford(dist.socgeo)

wynik[i,11]<- bb.socgeo.dist$MAD.conformity # column "Benford.socgeo.dist"

wynik[i,17]<-bb.socgeo.dist$MAD # column "MAD.socgeo.dist"

if(i==1) {# Spain

mapa1<-worldMap[mapa$NAME\_SORT==vec[i],]

aaa<-world.cities[world.cities$country.etc==vec[i],]

aaa<-aaa[aaa$lat>36,]

elim<-which(aaa$long>1 & aaa$lat<41)

aaa<-aaa[-elim,]

}

else if(i==3){ # Norway

mapa1<-worldMap[mapa$NAME\_SORT==vec[i],]

aaa<-world.cities[world.cities$country.etc==vec[i],]

cc<-mapa1@polygons[[1]]@Polygons[[1]]@coords # korekta mapy

mapa1<-spPolygons(cc)

}

else if(i==16) { # UK

mapa1<-worldMap[mapa$NAME\_SORT=="United Kingdom",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==34){ # "Macedonia"

mapa1<-worldMap[mapa$NAME\_SORT=="Macedonia, The former Yugoslav Republic of",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==40){ # Greenland

mapa1<-worldMap[mapa$NAME\_SORT=="Greenland",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==41){ # Faroe Islands

mapa1<-worldMap[mapa$NAME\_SORT=="Faroe Islands",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==42){ # Palestine / West bank

mapa1<-worldMap[mapa$NAME\_SORT=="West Bank",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

elim<-which(aaa$long<34.6)

aaa<-aaa[-elim,]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==68){ # South Korea

mapa1<-worldMap[mapa$NAME\_SORT=="Korea, South",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==70){ # East Timor

mapa1<-worldMap[mapa$NAME\_SORT=="East Timor (Timor Leste)",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==80){ # North Korea

mapa1<-worldMap[mapa$NAME\_SORT=="Korea, North",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==95){ # Ivory Coast (Ctte d'Ivoire)

mapa1<-worldMap[mapa$NAME\_SORT=="Ivory Coast (Ctte d'Ivoire)",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==108){ # Congo (Kinshasa)

mapa1<-worldMap[mapa$NAME\_SORT=="Congo (Kinshasa)",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}  
else if(i==121){ # Guinea Bissau (Guinea-Bissau)

mapa1<-worldMap[mapa$NAME\_SORT=="Guinea Bissau (Guinea-Bissau)",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==131){ # Congo (Brazzaville)

mapa1<-worldMap[mapa$NAME\_SORT=="Congo (Brazzaville)",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

wynik[i,3]<-as.character(mapa1$NAME)

}

else if(i==156){ # USA

mapa1<-worldMap[mapa$NAME\_SORT=="United States of America",]

wynik[i,3]<-as.character(mapa1$NAME)

aaa<-world.cities[world.cities$country.etc==vec[i],]

cc<-mapa1@polygons[[1]]@Polygons[[1]]@coords # korekta mapy

mapa1<-spPolygons(cc)

elim<-which(aaa$long<(-140))

aaa<-aaa[-elim,]

}

else{

mapa1<-worldMap[mapa$NAME\_SORT==vec[i],]

aaa<-world.cities[world.cities$country.etc==vec[i],]

}

proj4string(mapa1)<-"+proj=longlat +datum=WGS84 +ellps=WGS84"

mapa1.merc<-spTransform(mapa1, CRS("+proj=merc +datum=NAD83"))

box.owin<-as(mapa1.merc, "owin")

points.sp<-SpatialPoints(data.frame(x=aaa[,5], y=aaa[,4]), bbox=bbox(mapa1), proj4string=CRS("+proj=longlat +datum=NAD83"))

points.sp.merc<-spTransform(points.sp, CRS("+proj=merc +datum=NAD83"))

box.ppp<-ppp(x=points.sp.merc@coords[,1], y=points.sp.merc@coords[,2], window=box.owin)

cc<-clarkevans.test(box.ppp)

wynik[i,12]<-round(cc$statistic,4)

wynik[i,13]<-cc$p.value

}

############################ end of loop for the world #################

**# corrections to numerical form of data**

wynik

wynik.df<-as.data.frame(wynik)

wynik.df[,4]<-round(as.numeric(wynik.df[,4]),3)

wynik.df[,5]<-round(as.numeric(wynik.df[,5]),3)

wynik.df[,6]<-round(as.numeric(wynik.df[,6]),3)

wynik.df[,7]<-round(as.numeric(wynik.df[,7]),3)

wynik.df[,12]<-round(as.numeric(wynik.df[,12]),3)

wynik.df[,13]<-round(as.numeric(wynik.df[,13]),3)

wynik.df[,14]<-round(as.numeric(wynik.df[,14]),4)

wynik.df[,15]<-round(as.numeric(wynik.df[,15]),4)

wynik.df[,16]<-round(as.numeric(wynik.df[,16]),4)

wynik.df[,17]<-round(as.numeric(wynik.df[,17]),4)

wynik.df

**# change of names to allow for sorting by labels**

wynik.df$Benford.dist.nr<-ifelse(wynik.df[,8]=="Close conformity", "1.Close conformity", ifelse(wynik.df[,8]=="Acceptable conformity", "2.Acceptable conformity", ifelse(wynik.df[,8]=="Marginally acceptable conformity", "3.Marginally acceptable conformity","4.Nonconformity")))

wynik.df$Benford.pop.vec.nr<-ifelse(wynik.df[,9]=="Close conformity", "1.Close conformity", ifelse(wynik.df[,9]=="Acceptable conformity", "2.Acceptable conformity", ifelse(wynik.df[,9]=="Marginally acceptable conformity", "3.Marginally acceptable conformity","4.Nonconformity")))

wynik.df$Benford.pop.dist.nr<-ifelse(wynik.df[,10]=="Close conformity", "1.Close conformity", ifelse(wynik.df[,10]=="Acceptable conformity", "2.Acceptable conformity", ifelse(wynik.df[,10]=="Marginally acceptable conformity", "3.Marginally acceptable conformity","4.Nonconformity")))

wynik.df$Benford.socgeo.dist.nr<-ifelse(wynik.df[,11]=="Close conformity", "1.Close conformity", ifelse(wynik.df[,11]=="Acceptable conformity", "2.Acceptable conformity", ifelse(wynik.df[,11]=="Marginally acceptable conformity", "3.Marginally acceptable conformity","4.Nonconformity")))

**# labels from Clark-Evans changed into classes**

wynik.df$ClarkEvans.lab<-ifelse(wynik.df$ClarkEvans.pvalue<0.05 & wynik.df$RofClarkEvans<1, "clustering", ifelse(wynik.df$ClarkEvans.pvalue<0.05 & wynik.df$RofClarkEvans>1, "ordering", "random"))

**# saving the results**

write.table(wynik.df, file="wynik\_swiat.txt")

write.table(wynik.df, file="PART2.txt")

**# the code below gives the following result:**

head(wynik.df)

# country continent name.as.map tot.pop how.many.cities coefvar.popul Zipf

#1 Spain Europe Spain 34275359 864 3.404 -1.056

#2 Portugal Europe Portugal 6722908 438 2.036 -1.228

#3 Norway Europe Norway 3403067 714 7.033 -0.789

#4 Netherlands Europe Netherlands 13505251 318 1.595 -1.224

#5 Denmark Europe Denmark 3954225 310 5.047 -1.008

#6 Germany Europe Germany 52973876 998 2.740 -1.310

#

# Benford.dist Benford.pop.vec Benford.pop.dist Benford.socgeo.dist RofClarkEvans

#1 Nonconformity Nonconformity Close conformity Close conformity 0.788

#2 Nonconformity Nonconformity Close conformity Close conformity 0.795

#3 Nonconformity Nonconformity Close conformity Close conformity 0.749

#4 Nonconformity Nonconformity Close conformity Close conformity 1.082

#5 Nonconformity Nonconformity Close conformity Close conformity 1.126

#6 Nonconformity Nonconformity Close conformity Close conformity 1.075

#

# ClarkEvans.pvalue MAD.dist MAD.pop.vec MAD.pop.dist MAD.socgeo.dist Benford.dist.nr

#1 0.000 0.0064 0.0053 0.0009 0.0009 4.Nonconformity

#2 0.000 0.0041 0.0062 0.0008 0.0008 4.Nonconformity

#3 0.000 0.0028 0.0034 0.0010 0.0010 4.Nonconformity

#4 0.006 0.0043 0.0057 0.0012 0.0012 4.Nonconformity

#5 0.000 0.0055 0.0070 0.0011 0.0011 4.Nonconformity

#6 0.000 0.0052 0.0076 0.0008 0.0008 4.Nonconformity

#

# Benford.pop.vec.nr Benford.pop.dist.nr Benford.socgeo.dist.nr ClarkEvans.lab

#1 4.Nonconformity 1.Close conformity 1.Close conformity clustering

#2 4.Nonconformity 1.Close conformity 1.Close conformity clustering

#3 4.Nonconformity 1.Close conformity 1.Close conformity clustering

#4 4.Nonconformity 1.Close conformity 1.Close conformity ordering

#5 4.Nonconformity 1.Close conformity 1.Close conformity ordering

#6 4.Nonconformity 1.Close conformity 1.Close conformity ordering

## # Figure 1 – World map of Benford conformity

**# merging data with map dataset**

wm.df<-data.frame(ID=1:243, NAME=as.data.frame(worldMap)$NAME)

wm.df1<-merge(wm.df, wynik.df, by.x="NAME", by.y="name.as.map", all.x=TRUE, sort=FALSE)

w.data<-orderBy(~ID, data=wm.df1)

**# plotting the map**

x<-w.data$Benford.socgeo.dist.nr

kolory<-rev(brewer.pal(4, "Reds")) # selection of colours

kolory<-c(kolory, "grey50")

xcol<-ifelse(x=="1.Close conformity", kolory[1], ifelse(x=="2.Acceptable conformity", kolory[2], ifelse(x=="3.Marginally acceptable conformity", kolory[3], ifelse(x=="4.Nonconformity", kolory[4], kolory[5]))))

xcol[is.na(x)==TRUE]<-kolory[5]

poziomy<-c(names(table(x)), "NA")

plot(worldMap, col=xcol)

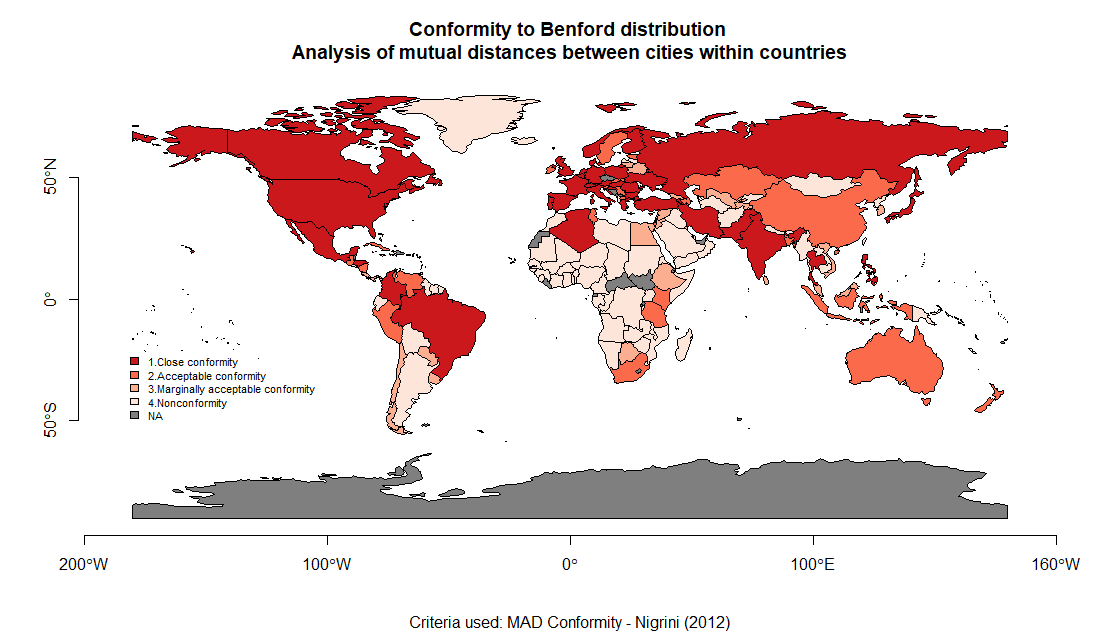
legend(-185, -20, legend=poziomy, fill=kolory, cex=0.7, bty="n")

degAxis(1)

degAxis(2)

title(main="Conformity to Benford distribution

Analysis of 3D socio-geo distances between cities within countries", sub="Criteria used: MAD Conformity - Nigrini (2012)")



## # Figure 3 – World map of Clark-Evans test

x<-w.data$RofClarkEvans

kolory<-brewer.pal(8, "Reds") # selection of colours

kolory<-c("grey50", kolory)

klasy<-classIntervals(x, 9, style="fixed", fixedBreaks=c(0, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2.0, 2.25))

tabela.kolorów<-findColours(klasy, kolory)

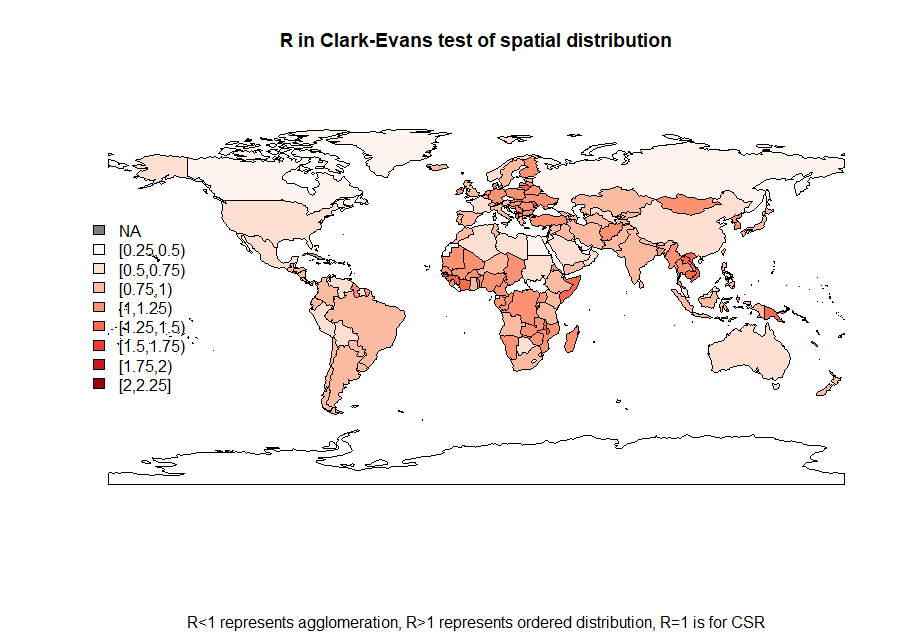
plot(worldMap, col=tabela.kolorów)

napisy<-names(attr(tabela.kolorów, "table"))

napisy[1]<-"NA"

legend("left", legend=napisy, fill=attr(tabela.kolorów, "palette"), cex=1, bty="n")

title(main="R in Clark-Evans test of spatial distribution", sub="R<1 represents agglomeration, R>1 represents ordered distribution, R=1 is for CSR")



## # Figure 4 – World map of Zipf coefficient

# plotting the map – ZIPF

x<-w.data$Zipf

cols<-rev(brewer.pal(6, "Reds")) # selection of colours

cols<-c(cols, "grey50", "white")

xcol<-ifelse(x<(-0.35)&x>(-0.95), cols[3], ifelse(x<(-0.95)& x>(-1.05), cols[1], ifelse(x<(-1.05)& x>(-1.45), cols[5], cols[7])))

poziomy<-c("non-Zipf: from -0.35 to -0.95", "close to Zipf: from -0.95 to -1.05", "non-Zipf: from -1.05 to -1.45", "NA")

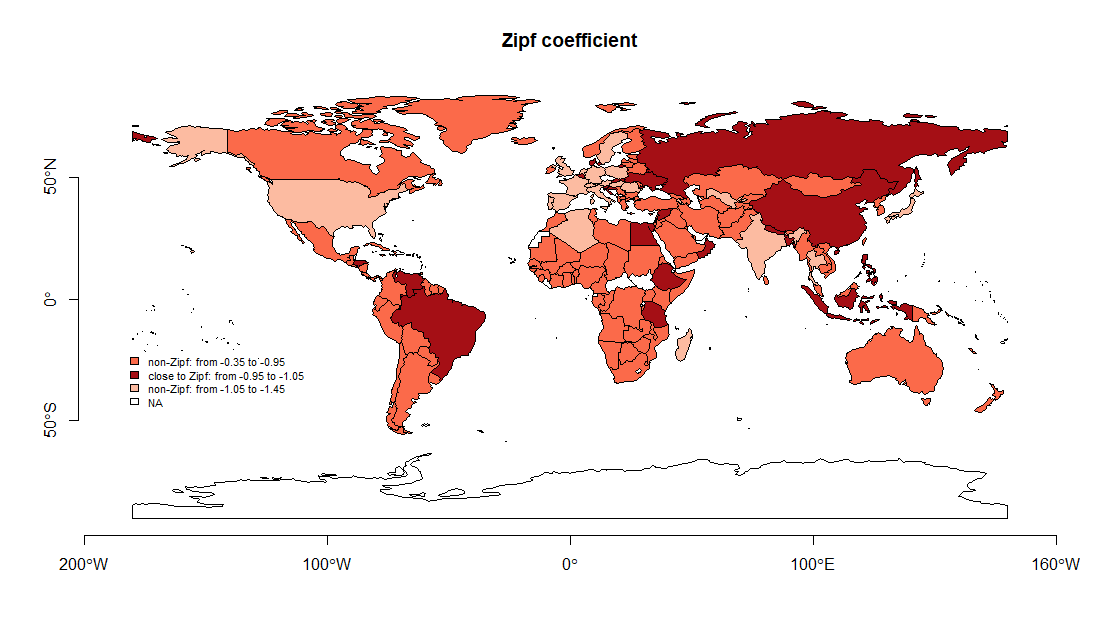
plot(worldMap, col=xcol, xlab="")

legend(-185, -20, legend=poziomy, fill=cols[c(3,1,5,8)], cex=0.7, bty="n")

degAxis(1)

degAxis(2)

title(main="Zipf coefficient")



## # Figure 7 – Locations of cities in selected countries

**# we selected four countries of different shapes and sizes**

**# to illustrate the spatial patterns of cities**

**# please note the location of the legend depends on the size of your screen**

### # USA

i=1

vec<-c("USA")

aaa<-world.cities[world.cities$country.etc==vec[i], 3:5]

dist.aaa<-as.vector(as.matrix(dist(as.data.frame(aaa)))) # distance matrix

bb<-benford(dist.aaa)

mapa1<-worldMap[mapa$NAME\_SORT=="United States of America",]

aaa<-world.cities[world.cities$country.etc==vec[i],]

cc<-mapa1@polygons[[1]]@Polygons[[1]]@coords # correction of the map

mapa1<-spPolygons(cc)

elim<-which(aaa$long<(-140))

aaa<-aaa[-elim,]

proj4string(mapa1)<-"+proj=longlat +datum=WGS84 +ellps=WGS84"

mapa1.merc<-spTransform(mapa1, CRS("+proj=merc +datum=NAD83"))

box.owin<-as(mapa1.merc, "owin")

points.sp<-SpatialPoints(data.frame(x=aaa[,5], y=aaa[,4]), bbox=bbox(mapa1), proj4string=CRS("+proj=longlat +datum=NAD83"))

points.sp.merc<-spTransform(points.sp, CRS("+proj=merc +datum=NAD83"))

box.ppp<-ppp(x=points.sp.merc@coords[,1], y=points.sp.merc@coords[,2], window=box.owin)

cc<-clarkevans.test(box.ppp)

x<-aaa$pop

xm<-x/mean(x)

brks<-c(0, 0.25, 0.50, 0.75, 1, 1.25, 1.50, 2, 8, 100)

size<-brks\*1.2

ss<-5 # legend position element

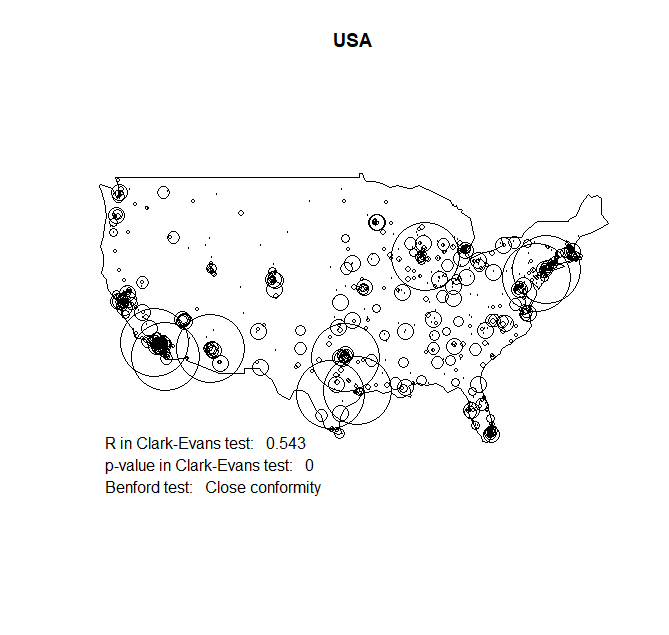
plot(mapa1, main="USA")

points(aaa[,5], aaa[,4], pch=21, cex=size[findInterval(xm, brks)])

text(-125, 20+ss, labels=paste("R in Clark-Evans test: ", round(cc$statistic,3)), pos=4)

text(-125, 18+ss, labels=paste("p-value in Clark-Evans test: ", cc$p.value), pos=4)

text(-125, 16+ss, labels=paste("Benford test: ", bb$MAD.conformity), pos=4)



### # China

i=1

vec<-c("China")

aaa<-world.cities[world.cities$country.etc==vec[i],]

dist.aaa<-as.vector(as.matrix(dist(as.data.frame(aaa))))

bb<-benford(dist.aaa)

mapa1<-worldMap[mapa$NAME\_SORT==vec[i],]

aaa<-world.cities[world.cities$country.etc==vec[i],]

proj4string(mapa1)<-"+proj=longlat +datum=WGS84 +ellps=WGS84"

mapa1.merc<- spTransform(mapa1, CRS("+proj=merc +datum=NAD83"))

box.owin<-as(mapa1.merc, "owin")

points.sp<-SpatialPoints(data.frame(x=aaa[,5], y=aaa[,4]), bbox=bbox(mapa1), proj4string=CRS("+proj=longlat +datum=NAD83"))

points.sp.merc<- spTransform(points.sp, CRS("+proj=merc +datum=NAD83"))

box.ppp<-ppp(x=points.sp.merc@coords[,1], y=points.sp.merc@coords[,2], window=box.owin)

cc<-clarkevans.test(box.ppp)

x<-aaa$pop

xm<-x/mean(x)

brks<-c(0, 0.25, 0.50, 0.75, 1, 1.25, 1.50, 2, 8, 100)

size<-brks\*1.2

plot(mapa1, main=vec[i])

degAxis(1)

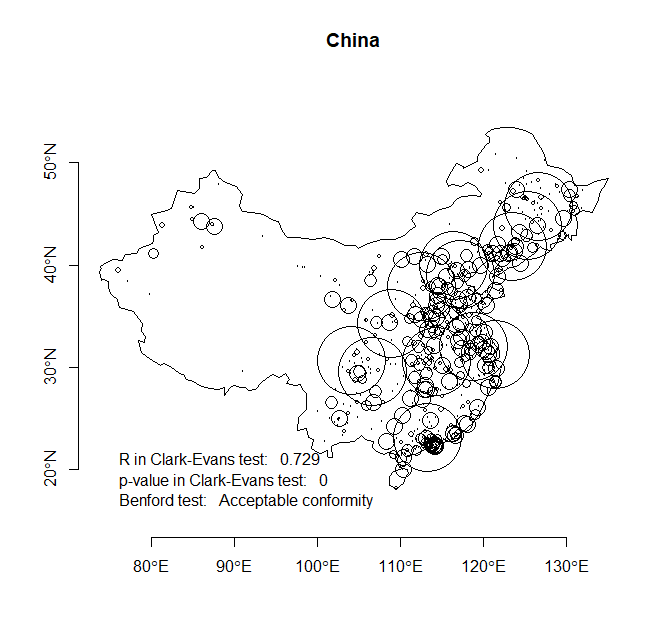
degAxis(2)

points(aaa[,5], aaa[,4], pch=21, cex=size[findInterval(xm, brks)])

text(75, 21, labels=paste("R in Clark-Evans test: ", round(cc$statistic,3)), pos=4)

text(75, 19, labels=paste("p-value in Clark-Evans test: ", cc$p.value), pos=4)

text(75, 17, labels=paste("Benford test: ", bb$MAD.conformity), pos=4)



### # Poland

i=1

vec<-c("Poland")

aaa<-world.cities[world.cities$country.etc==vec[i],]

dist.aaa<-as.vector(as.matrix(dist(as.data.frame(aaa))))

bb<-benford(dist.aaa)

mapa1<-worldMap[mapa$NAME\_SORT==vec[i],]

aaa<-world.cities[world.cities$country.etc==vec[i],]

proj4string(mapa1)<-"+proj=longlat +datum=WGS84 +ellps=WGS84"

mapa1.merc<-spTransform(mapa1, CRS("+proj=merc +datum=NAD83"))

box.owin<-as(mapa1.merc, "owin")

points.sp<-SpatialPoints(data.frame(x=aaa[,5], y=aaa[,4]), bbox=bbox(mapa1), proj4string=CRS("+proj=longlat +datum=NAD83"))

points.sp.merc<-spTransform(points.sp, CRS("+proj=merc +datum=NAD83"))

box.ppp<-ppp(x=points.sp.merc@coords[,1], y=points.sp.merc@coords[,2], window=box.owin)

cc<-clarkevans.test(box.ppp)

x<-aaa$pop

xm<-x/mean(x)

brks<-c(0, 0.25, 0.50, 0.75, 1, 1.25, 1.50, 2, 8, 100)

size<-brks\*1.2

plot(mapa1, main=vec[i])

degAxis(1)

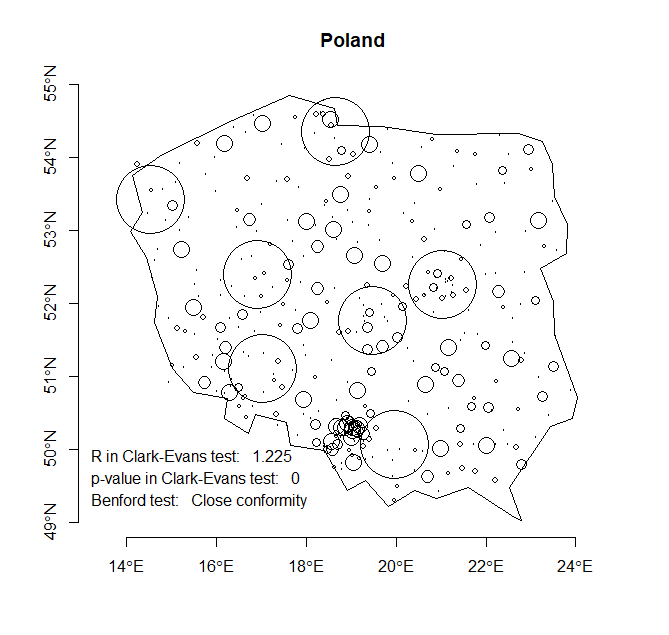
degAxis(2)

points(aaa[,5], aaa[,4], pch=21, cex=size[findInterval(xm, brks)])

text(13, 49.9, labels=paste("R in Clark-Evans test: ", round(cc$statistic,3)), pos=4)

text(13, 49.6, labels=paste("p-value in Clark-Evans test: ", cc$p.value), pos=4)

text(13, 49.3, labels=paste("Benford test: ", bb$MAD.conformity), pos=4)



### # Italy

i=1

vec<-c("Italy")

aaa<-world.cities[world.cities$country.etc==vec[i],]

dist.aaa<-as.vector(as.matrix(dist(as.data.frame(aaa))))

bb<-benford(dist.aaa)

mapa1<-worldMap[mapa$NAME\_SORT==vec[i],]

aaa<-world.cities[world.cities$country.etc==vec[i],]

proj4string(mapa1)<-"+proj=longlat +datum=WGS84 +ellps=WGS84"

mapa1.merc<- spTransform(mapa1, CRS("+proj=merc +datum=NAD83"))

box.owin<-as(mapa1.merc, "owin")

points.sp<-SpatialPoints(data.frame(x=aaa[,5], y=aaa[,4]), bbox=bbox(mapa1), proj4string=CRS("+proj=longlat +datum=NAD83"))

points.sp.merc<- spTransform(points.sp, CRS("+proj=merc +datum=NAD83"))

box.ppp<-ppp(x=points.sp.merc@coords[,1], y=points.sp.merc@coords[,2], window=box.owin)

cc<-clarkevans.test(box.ppp)

x<-aaa$pop

xm<-x/mean(x)

brks<-c(0, 0.25, 0.50, 0.75, 1, 1.25, 1.50, 2, 8, 100)

size<-brks\*1.2

plot(mapa1, main=vec[i])

degAxis(1)

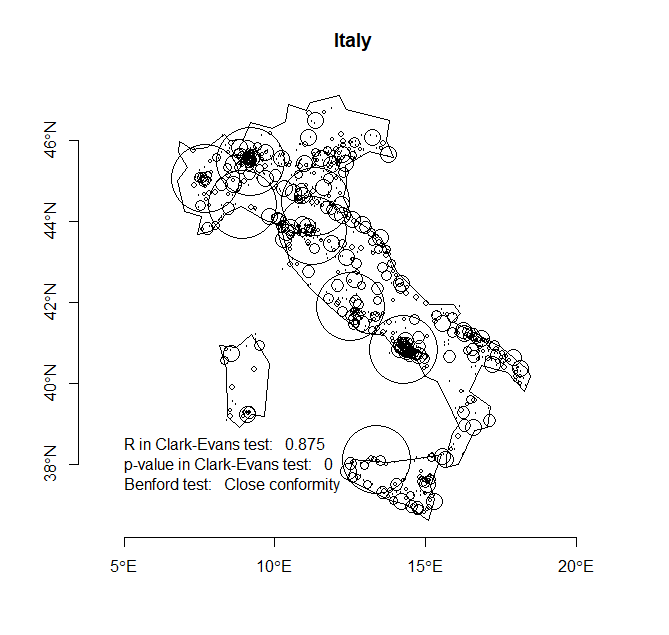
degAxis(2)

points(aaa[,5], aaa[,4], pch=21, cex=size[findInterval(xm, brks)])

text(4.7, 38.5, labels=paste("R in Clark-Evans test: ", round(cc$statistic,3)), pos=4)

text(4.7, 38, labels=paste("p-value in Clark-Evans test: ", round(cc$p.value,3)), pos=4)

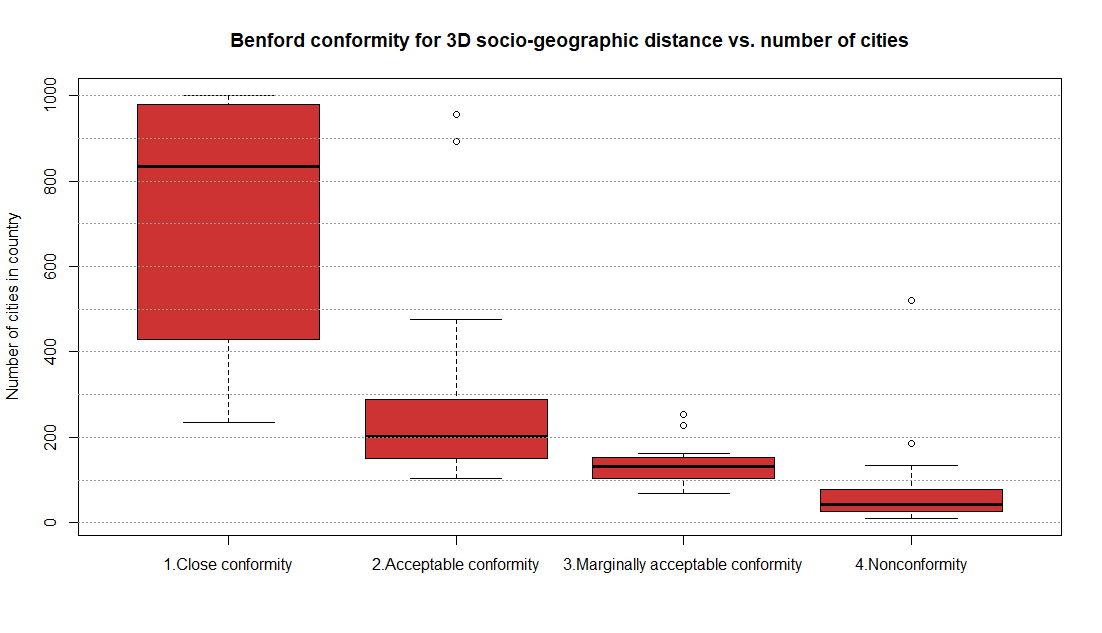
text(4.7, 37.5, labels=paste("Benford test: ", bb$MAD.conformity), pos=4)



## # Figure 2 – Boxplots of Benford conformity vs number of cities

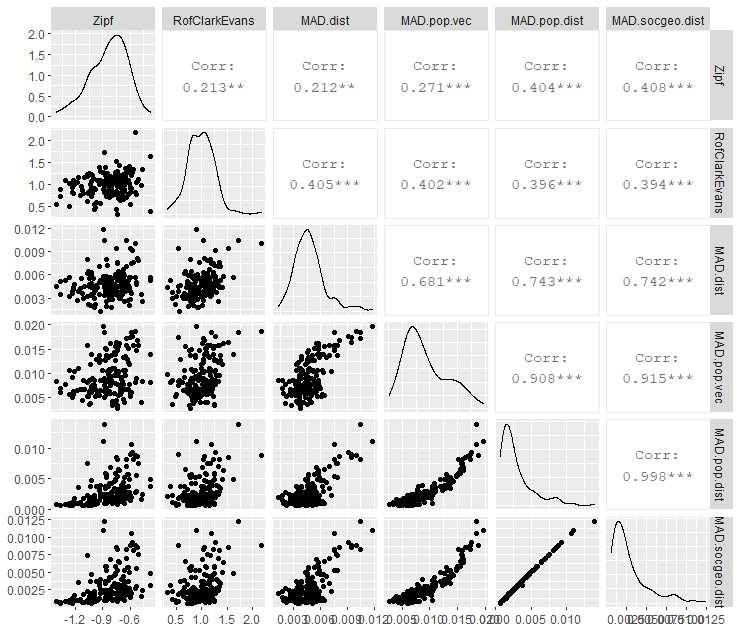
boxplot(wynik.df$how.many.cities~wynik.df$Benford.socgeo.dist.nr, xlab=" ", ylab="Number of cities in country", main="Benford conformity for 3D socio-geographic distance vs. number of cities", col="brown3")

abline(h=(0:10)\*100, lty=3, col="grey60")



## # Figure 6 – Relations in empirical data

ggpairs(wynik.df[,c(7,12,14:17)]) # from GGally::



## # Table 1 – Benford conformity by continents

**# final table was merged from individual outputs**

aggregate(wynik.df$how.many.cities, by=list(wynik.df$continent, wynik.df$Benford.socgeo.dist.nr), length)

Group.1 Group.2 x

1 Africa 1.Close conformity 1

2 America North 1.Close conformity 4

3 America South 1.Close conformity 2

4 Asia 1.Close conformity 8

5 Europe 1.Close conformity 23

6 Africa 2.Acceptable conformity 6

7 America North 2.Acceptable conformity 3

8 America South 2.Acceptable conformity 3

9 Asia 2.Acceptable conformity 7

10 Australia 2.Acceptable conformity 3

11 Europe 2.Acceptable conformity 9

12 Africa 3.Marginally acceptable conformity 4

13 America North 3.Marginally acceptable conformity 2

14 America South 3.Marginally acceptable conformity 3

15 Asia 3.Marginally acceptable conformity 10

16 Europe 3.Marginally acceptable conformity 2

17 Africa 4.Nonconformity 37

18 America North 4.Nonconformity 7

19 America South 4.Nonconformity 5

20 Asia 4.Nonconformity 19

21 Australia 4.Nonconformity 4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Continents** | **Close conformity** | **Acceptable conformity** | **Marginally acceptable conformity** | **Nonconformity** | **Total** |
| **Europe** | 23 | 9 | 2 | 7 | **41** |
| **Asia** | 8 | 7 | 10 | 19 | **44** |
| **North America** | 4 | 3 | 2 | 7 | **16** |
| **South America** | 2 | 3 | 3 | 5 | **13** |
| **Africa** | 1 | 6 | 4 | 37 | **48** |
| **Australia** | 0 | 3 | 0 | 4 | **7** |
| **Total** | **38** | **31** | **21** | **79** | **169** |

## # Table 2 – Benford conformity vs point patterns in countries

**# final table was merged from individual outputs**

# for all countries

table(wynik.df$Benford.socgeo.dist.nr, wynik.df$ClarkEvans.lab)

clustering ordering random

1.Close conformity 23 12 3

2.Acceptable conformity 18 6 7

3.Marginally acceptable conformity 12 3 6

4.Nonconformity 17 28 34

# for countries with 100+ cities

table(wynik.df$Benford.socgeo.dist.nr [wynik.df$how.many.cities>100], wynik.df$ClarkEvans.lab[wynik.df$how.many.cities>100])

clustering ordering random

1.Close conformity 23 12 3

2.Acceptable conformity 18 6 7

3.Marginally acceptable conformity 10 2 5

4.Nonconformity 5 3 5

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **MAD Conformity for Benford test** | **Agglomerated point-pattern** | | **Ordered point pattern** | | **Random**  **point-pattern** | |
| **Whole sample** | **Countries with 100+ cities** | **Whole sample** | **Countries with 100+ cities** | **Whole sample** | **Countries with 100+ cities** |
| **Close conformity** | 23 | 23 | 12 | 12 | 3 | 3 |
| **Acceptable conformity** | 18 | 18 | 6 | 6 | 7 | 7 |
| **Marginally acceptable conformity** | 12 | 10 | 3 | 2 | 6 | 5 |
| **Nonconformity** | 17 | 5 | 28 | 3 | 34 | 5 |

# **# 3. Simulation of point patterns to get a Benford-like mixture**

**# the part below is to be run without uploading any files**

## # Colours

colA<-viridis(15, option="A") # from viridis:: and scales::

colB<-viridis(15, option="B")

colC<-viridis(15, option="C")

colD<-viridis(15, option="D")

colE<-viridis(15, option="E")

colALL<-cbind(colA, colB, colC, colD, colE)

show\_col(colALL) # colour palettes to choose colours

# selection of colours for all plots

# colours: ordered / random / clustered centered / clustered skewed

cols<-c(colB[9], colC[13], colD[6], colD[9])

## # Bounding box with planar coordinates

x1<-c(0,2,2,0)

y1<-c(0,0,2,2)

xy1<-SpatialPointsDataFrame(matrix(c(x1,y1), ncol=2), data.frame(ID=seq(1:length(x1))),

proj4string=CRS("+proj=longlat +ellps=WGS84 +datum=WGS84"))

xy1.m<-matrix(c(x1,y1), ncol=2)

xy1.m<-rbind(xy1.m[dim(xy1.m)[1],],xy1.m) # one extra row

xy1.poly<-spPolygons(xy1.m)

plot(xy1.poly) # in spherical projection

axis(1)

axis(2)

# change from spherical into planar coordinates

proj4string(xy1.poly)<-"+proj=longlat +datum=WGS84 +ellps=WGS84"

xy.poly.merc<- spTransform(xy1.poly, CRS("+proj=merc +datum=NAD83"))

plot(xy.poly.merc) # in planar projection

axis(1)

axis(2)

## # Simulation of pure point patterns and tests for Benford conformity

### # random pattern

b<-spsample(xy.poly.merc, n=1000, "**random**") # random pattern

dist.r<-dist(as.data.frame(b)) # mutual distances between points

dist.r.m<-as.matrix(dist.r) # distances as matrix

dist.r.v<-as.vector(dist.r.m) # distances as vector

# Benford test

bb<-benford(dist.r.v)

bb

plot(bb)

bb.r<-bb # saving Benford output for a random pattern

# switching to spatstat:: - ppp objects

box.owin<-as(xy.poly.merc, "owin")

box.ppp<-ppp(x=b@coords[,1], y=b@coords[,2], window=box.owin)

# Clark-Evans test for random point pattern

cc<-clarkevans.test(box.ppp)

cc

########## 🡪 see the output figure here

# scatterplot of random point-pattern

plot(xy.poly.merc, main="Random point-pattern

n=1.000, x and y drawn randomly")

points(b, pch=21, bg=alpha(cols[2], 0.5), col=alpha(cols[2], 0.5), cex=0.8)

########## 🡪 see the output figure here

# barplot of Benford conformity

tt<-as.data.frame(bb$bfd)

barplot(tt$data.second.order.dist.freq~tt$digits, main="Benford Second-Order Test – Random pattern

all digits, 1.000 points, 1.000.000 mutual distances", xlab="digits", ylab="Second-order frequency distribution")

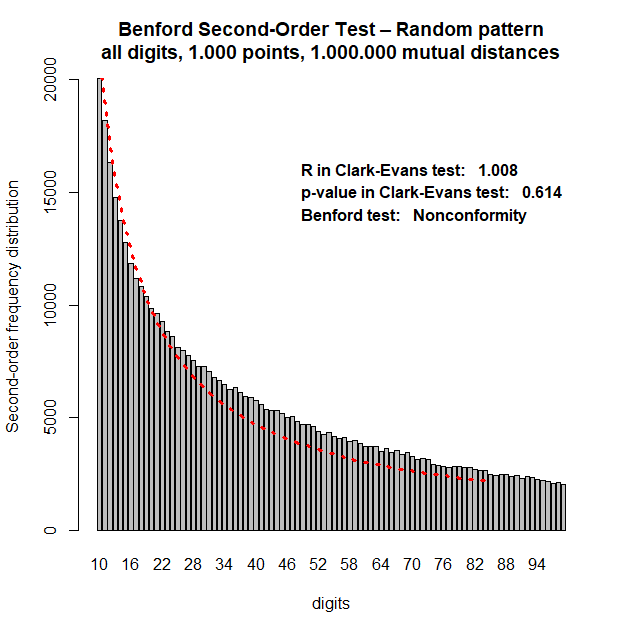
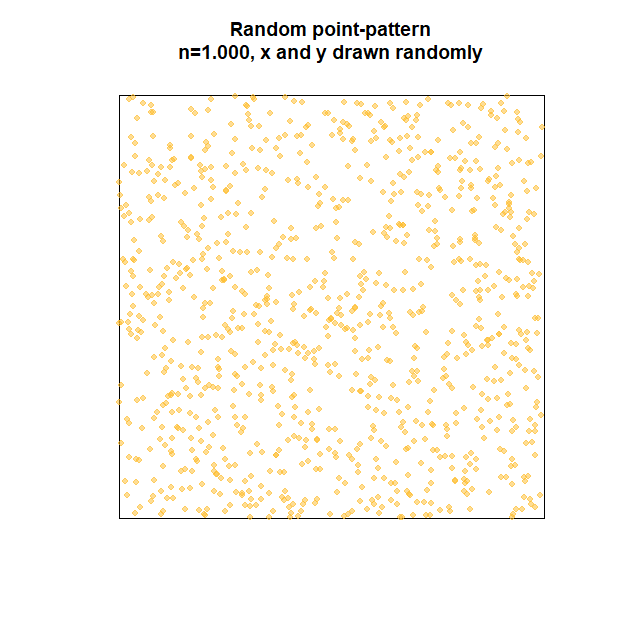
lines(tt$benford.so.dist.freq, lty=3, col="red", lwd=3)

text(45, 16000, labels=paste("R in Clark-Evans test: ", round(cc$statistic,3)), pos=4, font=2)

text(45, 15000, labels=paste("p-value in Clark-Evans test: ", round(cc$p.value,3)), pos=4, font=2)

text(45, 14000, labels=paste("Benford test: ", bb$MAD.conformity), pos=4, font=2)

########## 🡪 see the output figure here



### # regular (ordered, uniform) pattern

b<-spsample(xy.poly.merc, n=1000, "**regular**")

# slight jitter of locations (shifting by an epsilon)

# to reduce computational complexity

bdf<-as.data.frame(b)

epsilonx<-runif(dim(bdf)[1], -1,1)

epsilony<-runif(dim(bdf)[1], -1,1)

bdf[,1]<-bdf[,1]+epsilonx

bdf[,2]<-bdf[,2]+epsilony

dist.u<-dist(bdf) # mutual distances between points

dist.u.m<-as.matrix(dist.u) # distances as matrix

dist.u.v<-as.vector(dist.u.m) # distances as vector

# Benford test

bb<-benford(dist.u.v)

bb

plot(bb)

bb.u<-bb # saving Benford output for uniform pattern

# switching to spatstat:: - ppp objects

box.owin<-as(xy.poly.merc, "owin")

box.ppp<-ppp(x=bdf[,1], y=bdf[,2], window=box.owin)

# Clark-Evans test for random point pattern

cc<-clarkevans.test(box.ppp)

cc

########## 🡪 see the output figure here

# scatterplot of uniform point-pattern

plot(xy.poly.merc, main="Regular / ordered point-pattern

n=1.000, x and y coordinates as grid")

points(b, pch=21, bg=alpha(cols[1], 0.5), col=alpha(cols[1], 0.5), cex=0.8)

########## 🡪 see the output figure here

# barplot of Benford conformity

tt<-as.data.frame(bb$bfd)

barplot(tt$data.second.order.dist.freq~tt$digits, main="Benford Second-Order Test – Regular pattern

all digits, 1.000 points, 1.000.000 mutual distances", xlab="digits", ylab="Second-order frequency distribution")

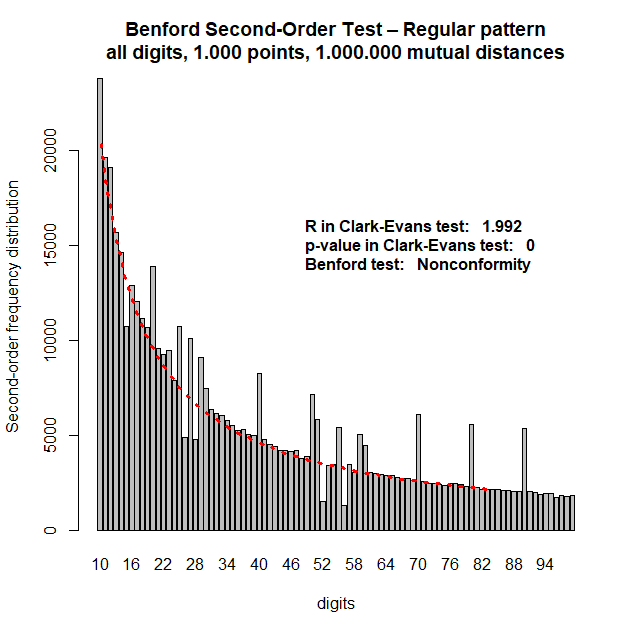
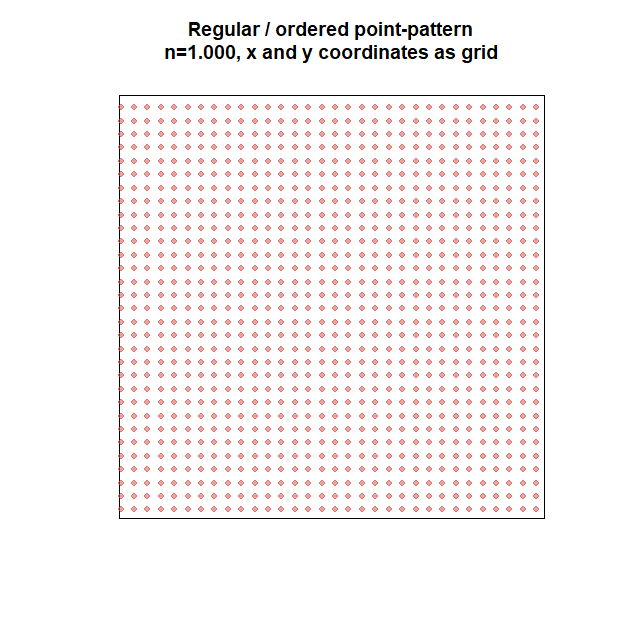
lines(tt$benford.so.dist.freq, lty=3, col="red", lwd=3)

text(45, 16000, labels=paste("R in Clark-Evans test: ", round(cc$statistic,3)), pos=4, font=2)

text(45, 15000, labels=paste("p-value in Clark-Evans test: ", round(cc$p.value,3)), pos=4, font=2)

text(45, 14000, labels=paste("Benford test: ", bb$MAD.conformity), pos=4, font=2)

########## 🡪 see the output figure here



### # clustered (agglomerated) pattern

# generating xy coordinates from a normal distribution

x<-rnorm(1000, 100000, 30000)

y<-rnorm(1000, 100000, 30000)

dist.c<-dist(as.data.frame(x=x, y=y)) # distances between points

dist.c.m<-as.matrix(dist.c) # distances as matrix

dist.c.v<-as.vector(dist.c.m) # distances as vector

# Benford test

bb<-benford(dist.c.v)

bb

plot(bb)

bb.c<-bb # saving Benford output for clustered pattern

# switching to spatstat:: - ppp objects

box.owin<-as(xy.poly.merc, "owin")

points.sp.merc<-SpatialPoints(data.frame(x=x, y=y), bbox=bbox(xy.poly.merc), proj4string=CRS("+proj=merc +datum=NAD83"))

box.ppp<-ppp(x=points.sp.merc@coords[,1], y=points.sp.merc@coords[,2], window=box.owin)

# Clark-Evans test for random point pattern

cc<-clarkevans.test(box.ppp)

cc

########## 🡪 see the output figure here

# scatterplot of clustered point-pattern

plot(xy.poly.merc, main="Clustered point-pattern

n=1.000, x and y coordinates from N(mi, 0.3\*mi)")

points(x,y, pch=21, bg=alpha(cols[3], 0.5), col=alpha(cols[3], 0.5), cex=0.8)

########## 🡪 see the output figure here

# barplot of Benford conformity

tt<-as.data.frame(bb$bfd)

barplot(tt$data.second.order.dist.freq~tt$digits, main="Benford Second-Order Test – Clustered pattern

all digits, 1.000 points, 1.000.000 mutual distances", xlab="digits", ylab="Second-order frequency distribution")

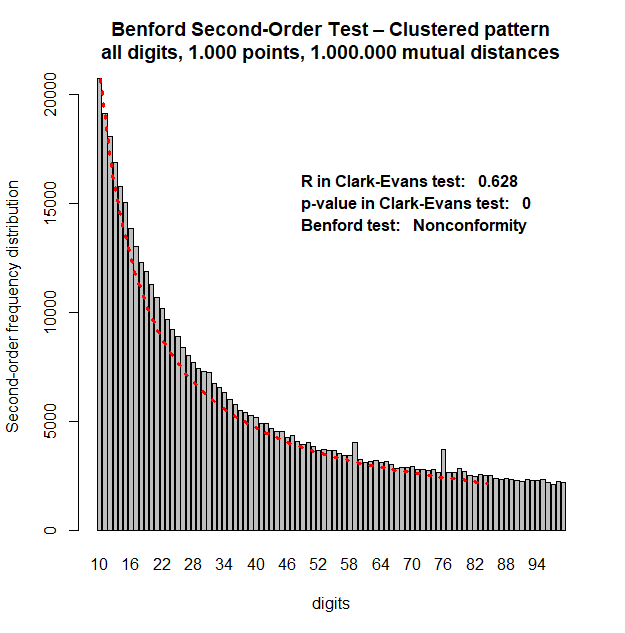
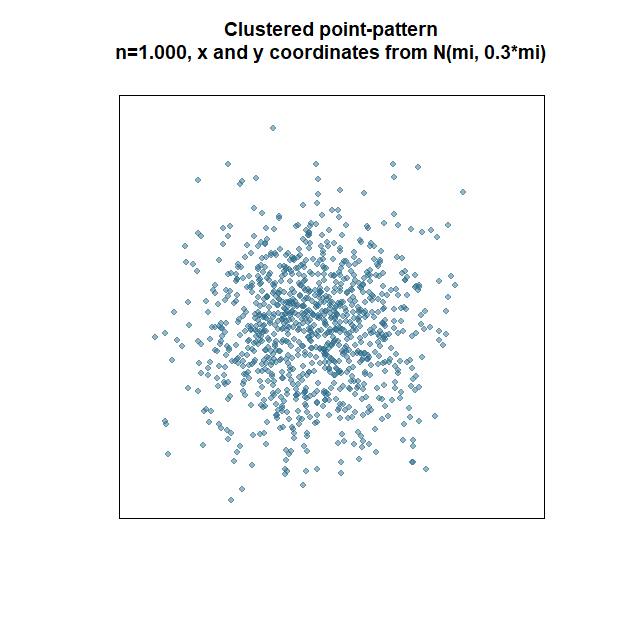
lines(tt$benford.so.dist.freq, lty=3, col="red", lwd=3)

text(45, 16000, labels=paste("R in Clark-Evans test: ", round(cc$statistic,3)), pos=4, font=2)

text(45, 15000, labels=paste("p-value in Clark-Evans test: ", round(cc$p.value,3)), pos=4, font=2)

text(45, 14000, labels=paste("Benford test: ", bb$MAD.conformity), pos=4, font=2)

########## 🡪 see the output figure here



### # Figure 8a - mixed point patterns (with clustered pattern centred)

# generating point-patterns

nn<-c(810, 120, 70) # number of obs. clustered/regular/random point-patterns

x<-rnorm(nn[1], 100000, 30000) # clustered

y<-rnorm(nn[1], 100000, 30000)

bb1<-data.frame(x=x, y=y)

b2<-spsample(xy.poly.merc, n=nn[2], "regular") # regular

bb2<-data.frame(x=b2@coords[,1], y=b2@coords[,2])

b3<-spsample(xy.poly.merc, n=nn[3], "random") # random

bb3<-data.frame(x=b3@coords[,1], y=b3@coords[,2])

bb4<-rbind(bb1, bb2, bb3) # all points in one object

# for spatstat:: owin & ppp objects

box.owin<-as(xy.poly.merc, "owin")

points.sp.merc<-SpatialPoints(data.frame(x=bb4[,1], y=bb4[,2]), bbox=bbox(xy.poly.merc), proj4string=CRS("+proj=merc +datum=NAD83"))

box.ppp<-ppp(x=points.sp.merc@coords[,1], y=points.sp.merc@coords[,2], window=box.owin)

# Clark-Evans test for random point pattern

cc<-clarkevans.test(box.ppp)

cc

# matrix of mutual distances between points

dist.bb4<-dist(as.data.frame(x=bb4[,1], y=bb4[,2])) # distances

dist.bb4.m<-as.matrix(dist.bb4) # distances as matrix

dist.bb4.v<-as.vector(dist.bb4.m) # distances as vector

# Benford test

bb<-benford(dist.bb4.v)

bb.m<-bb

bb

plot(bb)

########## 🡪 see the output figure here

# scatterplot of clustered point-pattern

plot(bb2, pch=21, bg=alpha(cols[1], 0.7), col=alpha(cols[1],0.7)) # ordered

points(bb3, pch=21, bg=cols[2], col=cols[2]) # random

points(bb1, pch=21, cex=0.99, bg=cols[3], col=cols[3]) # clustered

legend("topright", legend=c("ordered", "random", "clustered"), fill=cols, cex=1.2, bty="n")

title(main="Composite point-pattern

with centered core of clustered distribution ")

########## 🡪 see the output figure here

# barplot of Benford conformity

tt<-as.data.frame(bb.m$bfd)

barplot(tt$data.second.order.dist.freq~tt$digits, main="Benford Second-Order Test – Composite pattern

all digits, 1.000 points, 500.000 mutual distances", xlab="digits", ylab="Second-order frequency distribution")

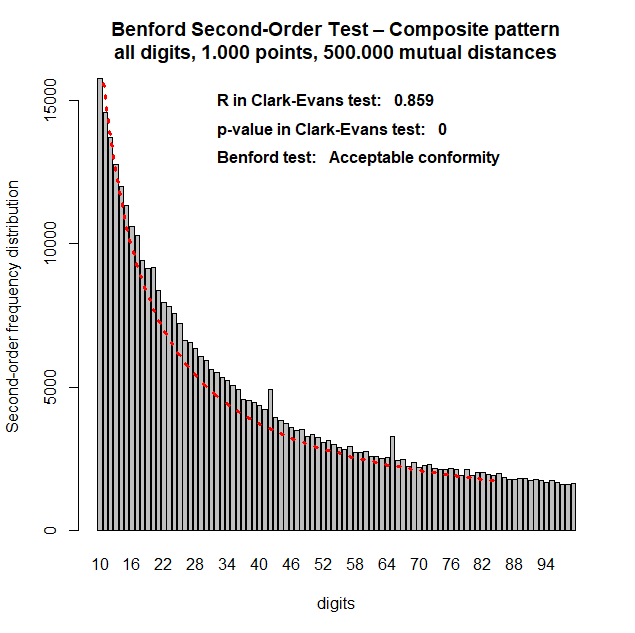
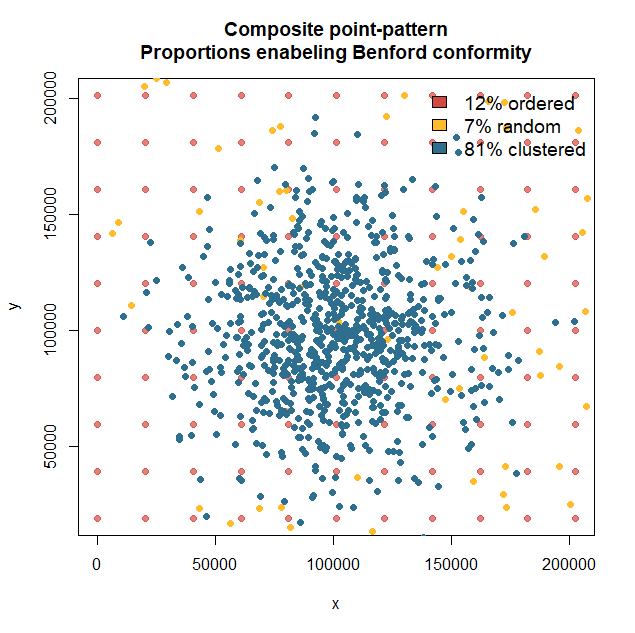
lines(tt$benford.so.dist.freq, lty=3, col="red", lwd=3)

text(25, 15000, labels=paste("R in Clark-Evans test: ", round(cc$statistic,3)), pos=4, font=2)

text(25, 14000, labels=paste("p-value in Clark-Evans test: ", round(cc$p.value,3)), pos=4, font=2)

text(25, 13000, labels=paste("Benford test: ", bb$MAD.conformity), pos=4, font=2)

########## 🡪 see the output figure here



## # Figure 8b – Spatial and Benford distributions in pure patterns

### # mixed point patterns (with clustered pattern shifted/skewed)

# generating point-patterns

nn<-c(810, 120, 70) # number of obs. clustered/regular/random point-patterns

x<-rnorm(nn[1], 50000, 30000) # clustered skewed

y<-rnorm(nn[1], 150000, 30000)

bb1<-data.frame(x=x, y=y)

b2<-spsample(xy.poly.merc, n=nn[2], "regular") # regular

bb2<-data.frame(x=b2@coords[,1], y=b2@coords[,2])

b3<-spsample(xy.poly.merc, n=n[3], "random") # random

bb3<-data.frame(x=b3@coords[,1], y=b3@coords[,2])

bb4<-rbind(bb1, bb2, bb3) # all points in one object

# for spatstat:: owin & ppp objects

box.owin<-as(xy.poly.merc, "owin")

points.sp.merc<-SpatialPoints(data.frame(x=bb4[,1], y=bb4[,2]), bbox=bbox(xy.poly.merc), proj4string=CRS("+proj=merc +datum=NAD83"))

box.ppp<-ppp(x=points.sp.merc@coords[,1], y=points.sp.merc@coords[,2], window=box.owin)

# Clark-Evans test for random point pattern

cc<-clarkevans.test(box.ppp)

cc

# matrix of mutual distances between points

dist.bb4<-dist(as.data.frame(x=bb4[,1], y=bb4[,2])) # distances

dist.bb4.m<-as.matrix(dist.bb4) # distances as matrix

dist.bb4.v.skewed<-as.vector(dist.bb4.m) # distances as vector

# Benford test

bb<-benford(dist.bb4.v.skewed)

bb.m.skew<-bb

bb

plot(bb)

########## 🡪 see the output figure here

# scatterplot of clustered point-pattern

plot(bb2, pch=21, bg=alpha(cols[1], 0.7), col=alpha(cols[1],0.7)) # ordered

points(bb3, pch=21, bg=cols[2], col=cols[2]) # random

points(bb1, pch=21, cex=0.99, bg=cols[3], col=cols[3]) # clustered

legend("topright", legend=c("ordered", "random", "clustered"), fill=cols, cex=1.2, bty="n")

title(main="Composite point-pattern

with shifted core of clustered distribution ")

########## 🡪 see the output figure here

# barplot of Benford conformity

tt<-as.data.frame(bb.m.skew$bfd)

barplot(tt$data.second.order.dist.freq~tt$digits, main="Benford Second-Order Test – Composite pattern

all digits, 1.000 points, 500.000 mutual distances", xlab="digits", ylab="Second-order frequency distribution")

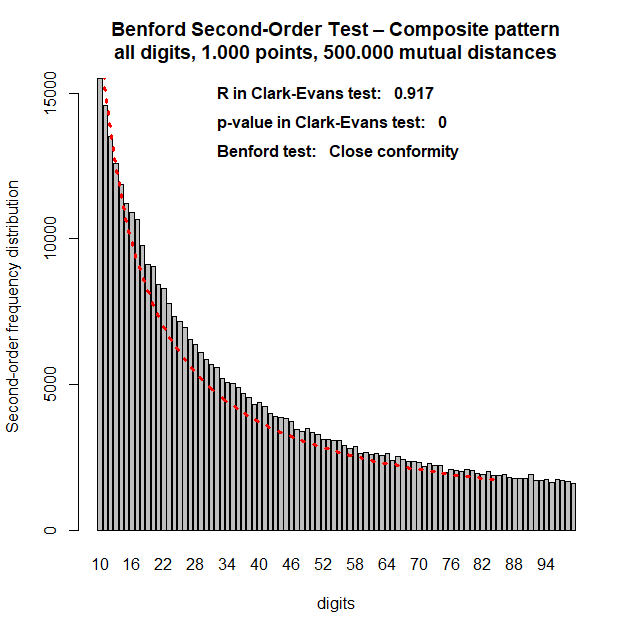
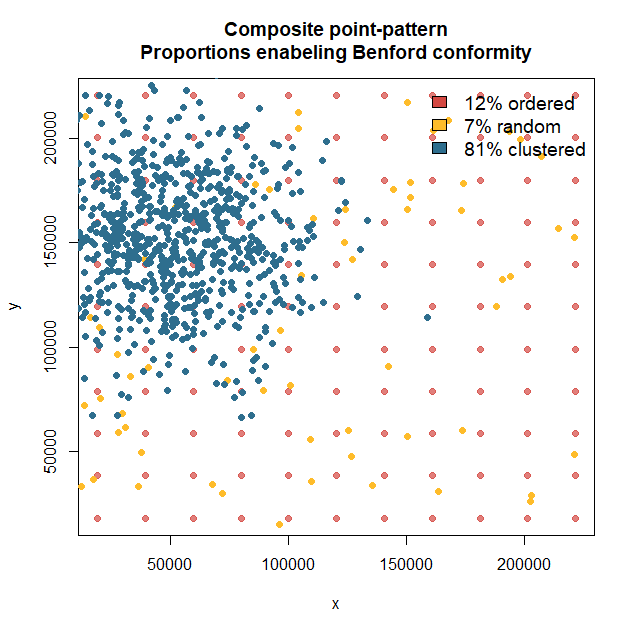
lines(tt$benford.so.dist.freq, lty=3, col="red", lwd=3)

text(25, 15000, labels=paste("R in Clark-Evans test: ", round(cc$statistic,3)), pos=4, font=2)

text(25, 14000, labels=paste("p-value in Clark-Evans test: ", round(cc$p.value,3)), pos=4, font=2)

text(25, 13000, labels=paste("Benford test: ", bb$MAD.conformity), pos=4, font=2)

########## 🡪 see the output figure here



### # Figure 8c - density plot of mutual distances in all point-patterns

plot(density(dist.c.v), col=cols[3], xlim=c(0, 300000), lwd=2, main="Density of mutual distances

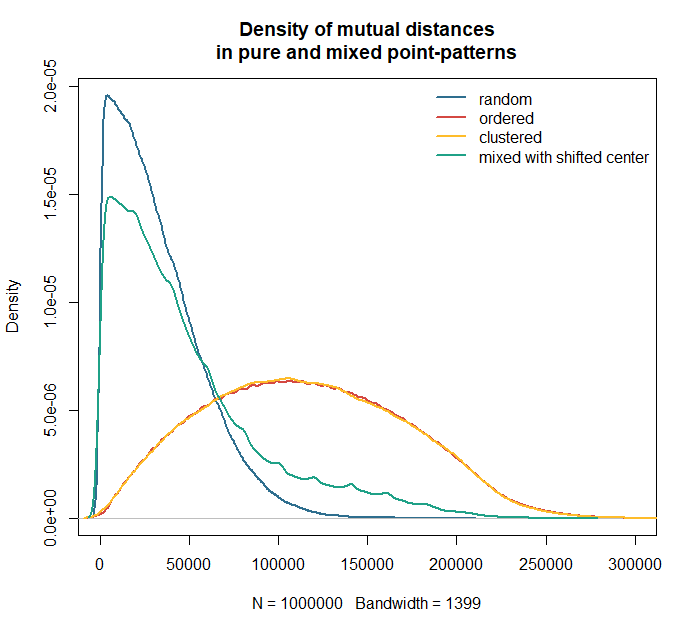
in pure and mixed point-patterns", sub="") # random pattern

lines(density(dist.u.v), col=cols[1], lwd=2) # uniform pattern

lines(density(dist.r.v), col=cols[2], lwd=2) # clustered pattern

lines(density(dist.bb4.v.skewed), col=cols[4], lwd=2) # skewed clustered

legend("topright", legend=c("random", "ordered", "clustered", "mixed with shifted center"), col=c(cols[3], cols[1], cols[2], cols[4]), bty="n", lty=c(1,1,1, 1), lwd=c(2,2,2,2))



### # Figure 8d - all Benford distributions together

# barplot Benford

tt.r<-as.data.frame(bb.r$bfd)

tt.c<-as.data.frame(bb.c$bfd)

tt.u<-as.data.frame(bb.u$bfd)

tt.skew<-as.data.frame(bb.m.skew$bfd)

plot(tt.r$data.second.order.dist.freq~tt.r$digits, type="l", lwd=2, lty=1, main="Benford Second-Order Test

Simulated random/ordered/clustered patterns

1.000 points, 500.000 mutual distances", xlab="digits", ylab="Second-order frequency distribution", axes=FALSE)

lines(tt.c$data.second.order.dist.freq~tt.c$digits, type="l", lwd=1, lty=2)

lines(tt.u$data.second.order.dist.freq~tt.u$digits, type="l", lwd=2, lty=3)

lines(tt.r$benford.so.dist.freq, lwd=1, lty=2, col="red")

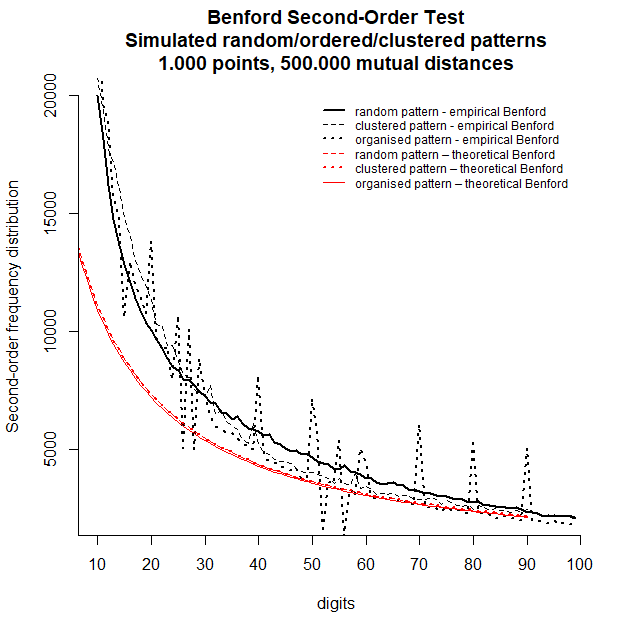
lines(tt.c$benford.so.dist.freq, lwd=2, lty=3, col="red")

lines(tt.u$benford.so.dist.freq, lwd=1, lty=1, col="red")

axis(1, at=(0:10)\*10, labels=(0:10)\*10)

axis(2, at=(0:4)\*5000, labels=(0:4)\*5000)

legend(50, 20000, legend=c("random pattern - empirical Benford", "clustered pattern - empirical Benford", "organised pattern - empirical Benford", "random pattern – theoretical Benford", "clustered pattern – theoretical Benford", "organised pattern – theoretical Benford"), col=c("black", "black","black", "red","red","red"), lwd=c(2,1,2,1,2,1), lty=c(1,2,3,2,3,1), bty="n", cex=0.75)



## # Simulation of a mixture of point patterns

# looking for proportions of point patterns - long-lasting simulation

# to generate many iterations with randomly selected proportions

**# NOTE: a result of that analysis can be imported as a file (see reading data)**

# use the bounding box as given before

# proportions of point-patterns

# decide how long you are to wait: 50000 takes very long to finish

# to check how that goes change into 50

n<-50 # number of generated scenarios (rows)

sampler<-matrix(0, nrow=n, ncol=3)

sampler[,1]<-runif(n, 0,1)

sampler[,2]<-runif(n, 0,1)

sampler[,3]<-1-sampler[,1]-sampler[,2]

elim<-which(sampler[,3]<0)

sampler<-sampler[-elim, ] # eliminating weights < 0

output<-matrix(0, nrow=dim(sampler)[1], ncol=7) # output structure

colnames(output)<-c("frac.agglom", "frac.random", "frac.ordered", "R.ClarkEvans", "pvalue.ClarkEvans", "Benford.1", "Benford.all")

nn=1000 # number of generated points

for(i in 1:dim(sampler)[1]) { # start of the loop

# generating points

x<-rnorm(nn\*sampler[i,1], 50000, 30000) # clustered shifted

y<-rnorm(nn\*sampler[i,1], 150000, 30000)

bb1<-data.frame(x=x, y=y)

b2<-spsample(xy.poly.merc, n=nn\*sampler[i,2], "regular") # regular

bb2<-data.frame(x=b2@coords[,1], y=b2@coords[,2])

b3<-spsample(xy.poly.merc, n=nn\*sampler[i,3], "random") # random

bb3<-data.frame(x=b3@coords[,1], y=b3@coords[,2])

bb4<-rbind(bb1, bb2, bb3) # integration in one object

# owin & ppp objects

box.owin<-as(xy.poly.merc, "owin")

points.sp.merc<-SpatialPoints(data.frame(x=bb4[,1], y=bb4[,2]), bbox=bbox(xy.poly.merc), proj4string=CRS("+proj=merc +datum=NAD83"))

box.ppp<-ppp(x=points.sp.merc@coords[,1], y=points.sp.merc@coords[,2], window=box.owin)

cc<-clarkevans.test(box.ppp) # Clark-Evans test

cc

# mutual distances between points

dist.bb4<-dist(as.data.frame(x=bb4[,1], y=bb4[,2]))

dist.bb4.m<-as.matrix(dist.bb4)

dist.bb4.v<-as.vector(dist.bb4.m)

# Benford tests for mutual distances

bb.1<-benford(dist.bb4.v, number.of.digits=1) # one-digit test

bb.all<-benford(dist.bb4.v) # two-digits test

bb.1

bb.all

output[i,1]<-sampler[i,1] # saving results to output object

output[i,2]<-sampler[i,2]

output[i,3]<-sampler[i,3]

output[i,4]<-cc$statistic

output[i,5]<-cc$p.value

output[i,6]<-bb.1$MAD.conformity

output[i,7]<-bb.all$MAD.conformity

} # end of the loop

output.df<-as.data.frame(output) # formatting of output object

output.df[,1]<-as.numeric(output.df[,1])

output.df[,2]<-as.numeric(output.df[,2])

output.df[,3]<-as.numeric(output.df[,3])

output.df[,4]<-as.numeric(output.df[,4])

output.df[,5]<-as.numeric(output.df[,5])

#x<-which(output.df$frac.agglom==0)

#output.df<-output.df[-x, ]

output.df

write.table(output.df, file="theor\_spat\_sim.txt")# export of output to file

data.all<-output.df # for coherenece of names with next section

head(output.df)

# frac.agglom frac.random frac.ordered R.ClarkEvans pvalue.ClarkEvans

#1 0.06072250 0.5022772 0.43700033 1.1618561 0.000000e+00

#2 0.28729343 0.2301735 0.48253309 0.9836716 3.281331e-01

#3 0.20864137 0.2640278 0.52733085 1.0279099 9.479660e-02

#4 0.82992345 0.1403319 0.02974465 0.9252074 1.150139e-05

#5 0.59123051 0.3003142 0.10845528 1.0984902 6.006316e-09

#6 0.03717054 0.8148784 0.14795108 1.6182755 0.000000e+00

#

# Benford.1 Benford.all

#1 Nonconformity Nonconformity

#2 Nonconformity Nonconformity

#3 Nonconformity Nonconformity

#4 Acceptable conformity Acceptable conformity

#5 Marginally acceptable conformity Marginally acceptable conformity

#6 Nonconformity Nonconformity

## # Figure 9 – Boxplots of simulated mixtures of point patterns and their Benford conformity

# reading the file with output of simulation – results obtained by Authors

# reading the file as in part 1 of codes

data.all<-read.table("theor\_spat\_sim.txt", sep=" ", dec=".", header=TRUE)

dim(data.all)

#[1] 3962 8

# ordering of MAD name – for automatic alphabetical sorting

data.all$Benford.all.nr<-ifelse(data.all$Benford.all=="Close conformity", "1.Close conformity", ifelse(data.all$Benford.all=="Acceptable conformity", "2.Acceptable conformity", ifelse(data.all$Benford.all=="Marginally acceptable conformity", "3.Marginally acceptable conformity","4.Nonconformity")))

# use colours as before

boxplot(data.all$frac.agglom~data.all$Benford.all.nr, boxwex=0.5, col=cols[3], sub="", ylab="", xlab="", main="Composition of point-patterns vs. Benford’s law conformity", ylim=c(0,1))

boxplot(data.all$frac.random~data.all$Benford.all.nr, add=TRUE, col=cols[2], boxwex=0.4)

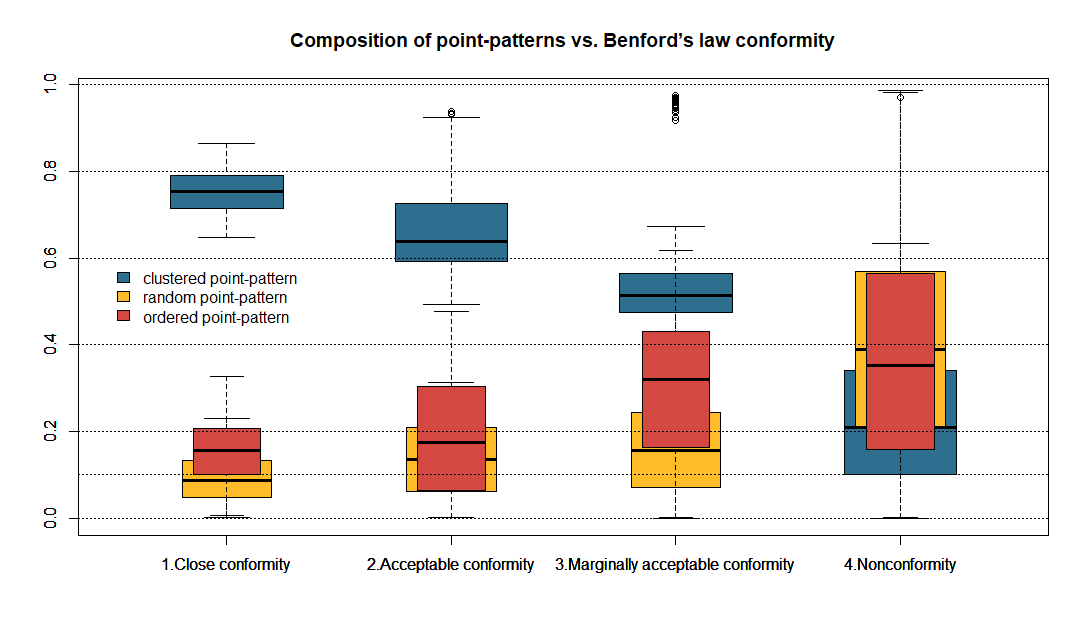
boxplot(data.all$frac.ordered~data.all$Benford.all.nr, add=TRUE, col=cols[1], boxwex=0.3)

abline(h=(0:10)\*2/10, lty=3)

legend(0.45, 0.6, legend=c("clustered point-pattern", "random point-pattern", "ordered point-pattern"), fill=c(cols[3], cols[2], cols[1]), bty="n")

abline(h=(0:10)\*2/10, lty=3)

abline(h=0.1, lty=3)



# **# 4. Fitting distributions to population data**

# quantiles of population data

i=1 # i=1 see data for Spain – see part 2

aaa<-world.cities[world.cities$country.etc==vec[i],3] # population only

quantile(aaa, probs=c(1:10)/10)

quantile(aaa, probs=c(0.9, 0.92, 0.94, 0.96, 0.98, 0.99, 1))

vec<-c("Spain", "Portugal", "Norway", "Netherlands", "Denmark", "Germany", "Switzerland", "Estonia", "Belgium", "Finland", "Hungary", "Turkey", "Italy", "France", "Ireland", "UK", "Czech Republic", "Romania", "Austria", "Luxembourg", "Cyprus", "Serbia", "Greece", "Croatia", "Bulgaria", "Latvia", "Slovenia", "Sweden", "Lithuania", "Iceland", "Ukraine", "Poland", "Moldova", "North Macedonia", "Belarus", "Malta", "Albania", "Bosnia and Herzegovina", "Slovakia", "Greenland", "Faroe Islands", "Palestine", "Pakistan", "Kuwait", "Iraq", "Saudi Arabia", "Syria", "Israel", "Iran", "Jordan", "Lebanon", "Bahrain", "Afghanistan", "Yemen", "Georgia", "Japan", "Kazakhstan", "Oman", "Indonesia", "Bangladesh", "India", "Armenia", "Qatar", "Philippines", "Russia", "China", "Tajikistan", "Korea South", "Azerbaijan", "East Timor", "Kyrgyzstan", "Turkmenistan", "Uzbekistan", "Myanmar", "Malaysia", "Mongolia", "Nepal", "Sri Lanka", "Thailand", "Korea North", "Bhutan", "Vietnam", "Laos", "Cambodia", "Taiwan", "Somalia", "Ethiopia", "Sudan", "Tunisia", "Algeria", "Mauritania", "Nigeria", "Niger", "Chad", "Ivory Coast", "Egypt", "Benin", "Ghana", "Gambia", "Equatorial Guinea", "Cameroon", "Eritrea", "South Africa", "Uganda", "Morocco", "Kenya", "Libya", "Congo Democratic Republic", "Mauritius", "Sierra Leone", "Central African Republic", "Madagascar", "Togo", "Mozambique", "Guyana", "Namibia", "Mali", "Botswana", "Tanzania", "Cape Verde", "Guinea-Bissau", "Senegal", "Malawi", "Burkina Faso", "Zimbabwe", "Angola", "Guinea", "Swaziland", "Gabon", "Comoros", "Congo", "Burundi", "Zambia", "Brazil", "Paraguay", "Peru", "Bolivia", "Colombia", "French Guiana", "Venezuela", "Uruguay", "Dominican Republic", "Ecuador", "Suriname", "Argentina", "Chile", "Mexico", "Jamaica", "Cuba", "Haiti", "El Salvador", "Guatemala", "Honduras", "Nicaragua", "Panama", "USA", "Puerto Rico", "Costa Rica", "Trinidad and Tobago", "Dominica", "Saint Kitts and Nevis", "Canada", "Australia", "Samoa", "American Samoa", "Guam", "New Zealand", "Papua New Guinea", "New Caledonia")

## # Figure 5a - fitting for 0%-90% centiles - Gamma (1, 1) distribution

i=1 # for first country – basic plot

aaa<-world.cities[world.cities$country.etc==vec[i],3]

aaa<-aaa[aaa<quantile(aaa, probs=0.90)]

aaas<-as.numeric(scale(aaa))

plot(density(aaas), ylim=c(0,2), xlim=c(-4, 6), main="0%-90% centiles of populations

all 169 countries", sub="Density of in-sample standardised population of cities")

for(i in 2:169){ # for other countries

aaa<-world.cities[world.cities$country.etc==vec[i],3]

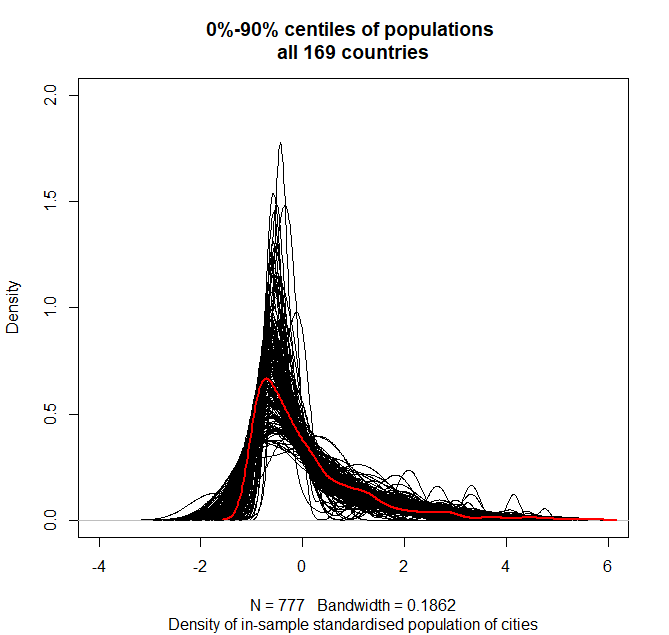
aaa<-aaa[aaa<quantile(aaa, probs=0.90, na.rm=TRUE)]

aaas<-as.numeric(scale(aaa))

lines(density(aaas))

} # end of the loop

lines(density(rgamma(1000, 1, 1)**-1**), col="red", lwd=2) # gamma theoretical



## # Figure 5b - fitting for 90%-99% centiles - Gamma (1, 1) distribution

i=1

aaa<-world.cities[world.cities$country.etc==vec[i],3]

summary(aaa)

aaa<-aaa[aaa>quantile(aaa, probs=0.90) & aaa<quantile(aaa, probs=0.99)]

summary(aaa)

aaas<-as.numeric(scale(aaa))

plot(density(aaas), ylim=c(0,2), xlim=c(-4, 6), main="90%-99% centiles of population

All 169 countries", sub="Density of in-sample standardised population of cities")

for(i in 2:169){

aaa<-world.cities[world.cities$country.etc==vec[i],3]

summary(aaa)

aaa<-aaa[aaa>quantile(aaa, probs=0.90) & aaa<quantile(aaa, probs=0.99)]

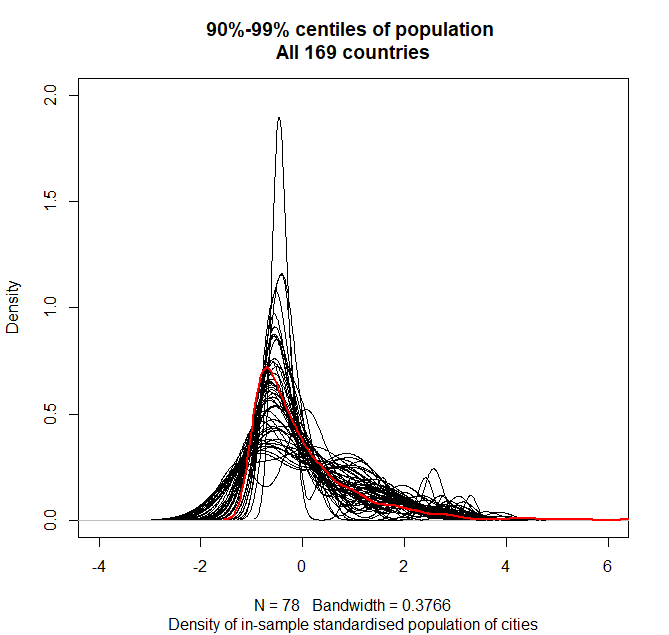
summary(aaa)

aaas<-as.numeric(scale(aaa))

lines(density(aaas))

}

lines(density(rgamma(1000, 1, 1)-1), col="red", lwd=2)



## # Figure 5c - fitting for 99%-100% centiles - Gamma (1, 1) distribution

i=1

aaa<-world.cities[world.cities$country.etc==vec[i],3]

summary(aaa)

aaa<-aaa[aaa>quantile(aaa, probs=0.99) & aaa<quantile(aaa, probs=1)]

summary(aaa)

aaas<-as.numeric(scale(aaa))

plot(density(aaas), ylim=c(0,2), xlim=c(-4, 6), main="99%-100% centiles of population

All 169 countries ", sub="Density of in-sample standardised population of cities")

for(i in 2:169){

aaa<-world.cities[world.cities$country.etc==vec[i],3]

summary(aaa)

aaa<-aaa[aaa>quantile(aaa, probs=0.99) & aaa<quantile(aaa, probs=1)]

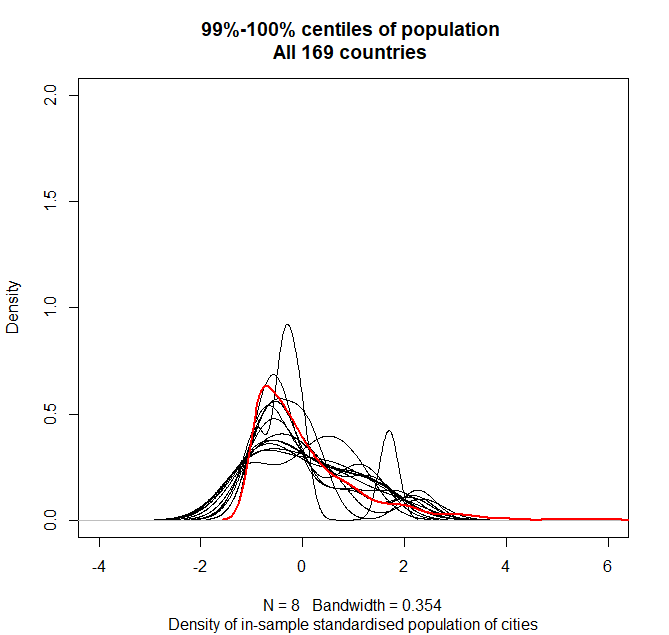
summary(aaa)

aaas<-as.numeric(scale(aaa))

lines(density(aaas))

}

lines(density(rgamma(1000, 1, 1)-1), col="red", lwd=2)



## # combination of three Gamma(1,1) distributions

i=1

aaa<-world.cities[world.cities$country.etc==vec[i],3]

aaa.s<-scale(aaa)

a1<-aaa[aaa<quantile(aaa, probs=0.90)]

a1.s<-scale(a1)

a2<-aaa[aaa>quantile(aaa, probs=0.90) & aaa<quantile(aaa, probs=0.99)]

a2.s<-scale(a2)

a3<-aaa[aaa>quantile(aaa, probs=0.99) & aaa<quantile(aaa, probs=1)]

a3.s<-scale(a3)

param<-matrix(0, ncol=3, nrow=3)

param.s<-matrix(0, ncol=3, nrow=3)

colnames(param)<-c("mean", "sd", "n")

param[1,1]<-mean(a1)

param[1,2]<-sd(a1)

param[1,3]<-length(a1)

param[2,1]<-mean(a2)

param[2,2]<-sd(a2)

param[2,3]<-length(a2)

param[3,1]<-mean(a3)

param[3,2]<-sd(a3)

param[3,3]<-length(a3)

param[3,2]<-ifelse(is.na(param[3,2])==TRUE, 0, param[3,2])

param

x1<-rgamma(param[1,3], shape=1, scale=1)

x1<-(x1-1)\*param[1,2]+param[1,1]

plot(density(a1), col="red", "Empirical and theoretical population

0-90% deciles")

lines(density(x1))

x2<-rgamma(param[2,3], shape=1, scale=1)

x2<-(x2-1)\*param[2,2]+param[2,1]

plot(density(a2), col="red", "Empirical and theoretical population

90-99% deciles")

lines(density(x2))

x3<-rgamma(param[3,3], shape=1, scale=1)

x3<-(x3-1)\*param[3,2]+param[3,1]

plot(density(a3), col="red", "Empirical and theoretical population

99-100% deciles")

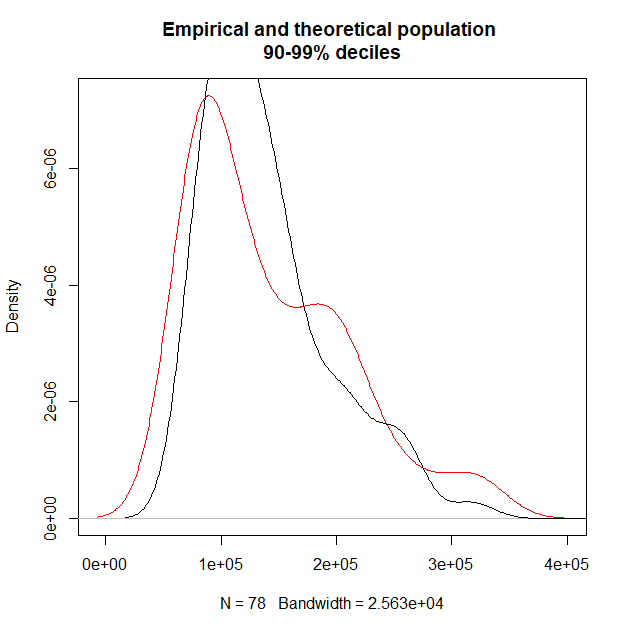
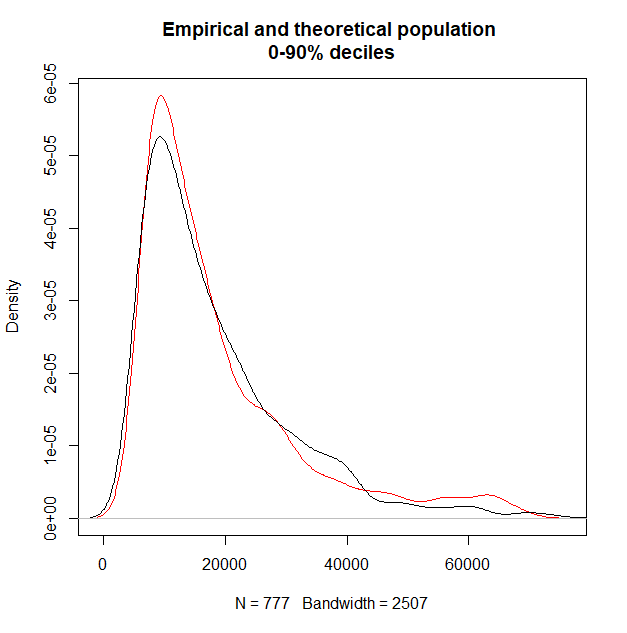
lines(density(x3))

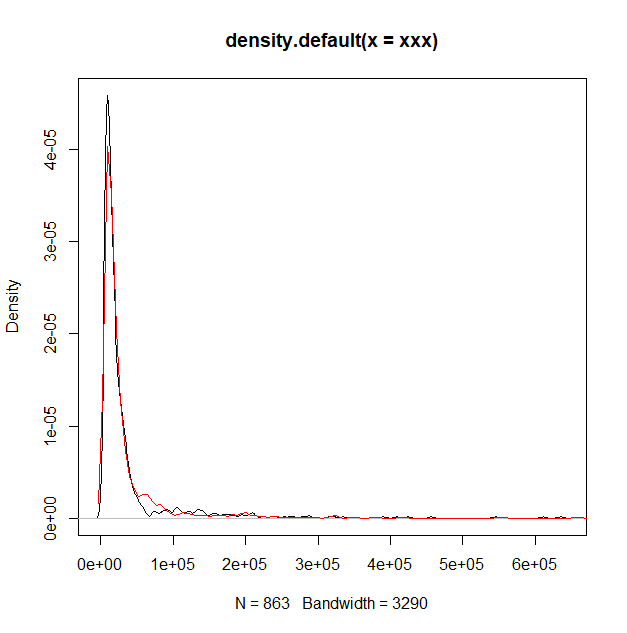
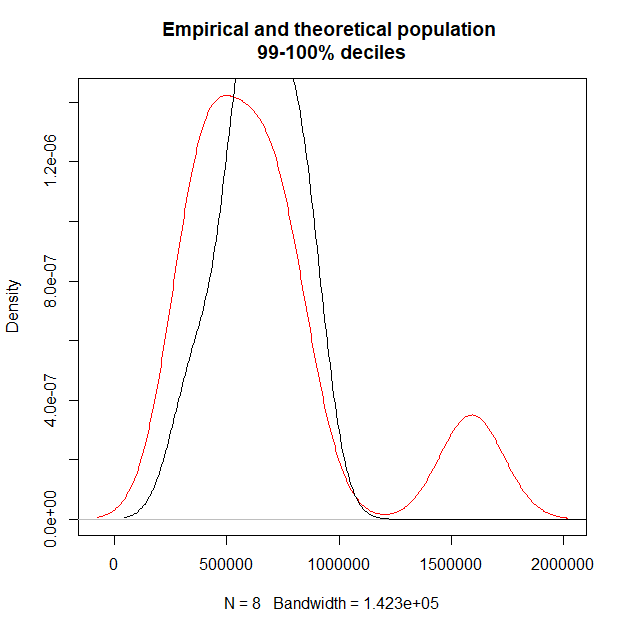
**# merging subsamples in a single dataset**

xxx<-c(x1, x2, x3)

plot(density(xxx))

lines(density(aaa), col="red")





## # Benford conformity of combination of three Gamma(1,1) distributions - (xi-1) transformation

**# generating gamma**

x1<-rgamma(param[1,3], shape=1, scale=1)

x1<-**(x1-1)**\*param[1,2]+param[1,1]

x2<-rgamma(param[2,3], shape=1, scale=1)

x2<-**(x2-1)**\*param[2,2]+param[2,1]

x3<-rgamma(param[3,3], shape=1, scale=1)

x3<-**(x3-1)**\*param[3,2]+param[3,1]

**# merging subsamples in a single dataset**

xxx<-c(x1, x2, x3)

plot(density(xxx))

lines(density(aaa), col="red")

iter=100 # takes long for 100 iter

beny<-matrix(0, nrow=iter, ncol=2)

dist.aaa<-as.vector(as.matrix(dist(as.data.frame(xxx))))

bb.all<-benford(dist.aaa)

bb.all$MAD.conformity

plot(bb.all)

bb.all

Benford object:

Data: dist.aaa

Number of observations used = 743906

Number of obs. for second order = 371932

First digits analysed = 2

Mantissa:

Statistic Value

Mean 0.476

Var 0.088

Ex.Kurtosis -1.221

Skewness 0.168

The 5 largest deviations:

digits absolute.diff

1 12 5834.26

2 11 4920.86

3 19 3380.46

4 18 3238.27

5 10 3145.73

Stats:

Pearson's Chi-squared test

data: dist.aaa

X-squared = 16280, df = 89, p-value < 2.2e-16

Mantissa Arc Test

data: dist.aaa

L2 = 0.0092527, df = 2, p-value < 2.2e-16

Mean Absolute Deviation (MAD): 0.001428648

**MAD Conformity - Nigrini (2012): Acceptable conformity**

Distortion Factor: -3.660113

Remember: Real data will never conform perfectly to Benford's Law. You should not focus on p-values!>

## # Benford conformity of combination of three Gamma(1,1) distributions - no transformation

**# generating gamma**

x1<-rgamma(param[1,3], shape=1, scale=1)

x1<-**(x1)**\*param[1,2]+param[1,1]

x2<-rgamma(param[2,3], shape=1, scale=1)

x2<-**(x2)**\*param[2,2]+param[2,1]

x3<-rgamma(param[3,3], shape=1, scale=1)

x3<-**(x3)**\*param[3,2]+param[3,1]

**# merging subsamples in single dataset**

xxx<-c(x1, x2, x3)

plot(density(xxx))

lines(density(aaa), col="red")

**# Benford conformity**

iter=100 # takes long for 100 iter

beny<-matrix(0, nrow=iter, ncol=2)

dist.aaa<-as.vector(as.matrix(dist(as.data.frame(xxx))))

bb.all<-benford(dist.aaa)

bb.all$MAD.conformity

plot(bb.all)

bb.all

Benford object:

Data: dist.aaa

Number of observations used = 743906

Number of obs. for second order = 371937

First digits analysed = 2

Mantissa:

Statistic Value

Mean 0.440

Var 0.084

Ex.Kurtosis -1.104

Skewness 0.327

The 5 largest deviations:

digits absolute.diff

1 12 15120.26

2 13 14823.62

3 11 9474.86

4 14 6660.18

5 20 4573.15

Stats:

Pearson's Chi-squared test

data: dist.aaa

X-squared = 48523, df = 89, p-value < 2.2e-16

Mantissa Arc Test

data: dist.aaa

L2 = 0.024105, df = 2, p-value < 2.2e-16

Mean Absolute Deviation (MAD): 0.002358284

**MAD Conformity - Nigrini (2012): Nonconformity**

Distortion Factor: -11.48876

Remember: Real data will never conform perfectly to Benford's Law. You should not focus on p-values!>

## # Benford conformity of simulated population

**# Checking Benford for a distance of population – only triple gamma (1,1) transformed as (x-1)**

iter=100 # takes long for 100 iter

beny<-matrix(0, nrow=iter, ncol=1)

**# all mean and sd parameters expressed as ration of mean in group 1**

param<-data.frame(mean=c(1,8,36), sd=c(0.7\*1,0.7\*5,0.7\*30), n=c(0.9, 0.09, 0.01))

param

size=1000 # number of observations, to multiply n

for(i in 1:iter){

x1<-rgamma(param[1,3]\*size, 1, 1)

x1<-(**x1-1**)\*param[1,2]+param[1,1]

x2<-rgamma(param[2,3]\*size, 1, 1)

x2<-(**x2-1**)\*param[2,2]+param[2,1]

x3<-rgamma(param[3,3]\*size, 1, 1)

x3<-as.numeric((**x3-1**)\*param[3,2]+param[3,1])

xxx<-c(x1, x2, x3)

dist.xxx<-as.vector(as.matrix(dist(as.data.frame(xxx))))

bb.xxx<-benford(dist.xxx)

beny[i,1]<-bb.xxx$MAD.conformity

}

table(beny[,1]) # ca. 98% of results are Benford-like

# **# 5. 3D simulation – spatial patterns, population, values allocation – to best Benford**

**# NOTE: a result of that analysis can be imported as a file (see reading data)**

## # simulation of complex patterns

**# the values are logically assigned to locations**

**# inside the simulation, an option to change into randomly assigned values**

iter=200 # number of iterations for each of 5 models, limit to 5-10 to run quickly

beny<-matrix(0, nrow=iter, ncol=7) # output matrix

colnames(beny)<-c("1D.pop.Benf", "1D.pop.MAD", "2D.geo.Benf", "2D.geo.MAD", "3D.socgeo.Benf", "3D.socgeo.MAD", "scenario")

size=1000 # number of observations, to multiply proportions

param<-data.frame(mean=c(1,8,36), sd=c(0.7\*1,0.7\*5,0.7\*30), n=c(0.9\*size, 0.09\*size, 0.01\*size))

for(j in 1:5){ # simulation assumes 5 models

beny[,7]<-rep(j, times=iter) # column with scenario ID

for(i in 1:iter){ # iterations for each model

# generating point-patterns

# bb1 clustered/ bb2 regular/ bb3 random point-patterns

# in nn – number of observations in each point pattern (clustered/regular/random)

if(j==1) { # scenario 1

nn<-c(810, 120, 70) # sum up to 1000 # best Benford ratios

x<-rnorm(nn[1], 50000, 30000) # non-centered cluster (Benford-like)

y<-rnorm(nn[1], 150000, 30000)

}

else if(j==2){ # scenario 2

nn<-c(810, 120, 70) # sum up to 1000 # best Benford ratios

x<-rnorm(nn[1], 100000, 30000) # centered cluster (usually non-Benford)

y<-rnorm(nn[1], 100000, 30000)

}

else if(j==3){ # scenario 3

nn<-c(300, 500, 250) # sum up to 1000 # non-Benford ratios

x<-rnorm(nn[1], 100000, 30000) # centered cluster (usually non-Benford)

y<-rnorm(nn[1], 100000, 30000)

}

else if(j==4){ # scenario 4

nn<-c(810, 120, 70) # sum up to 1000 # best Benford ratios

x<-rnorm(nn[1], 50000, 30000) # non-centered cluster (Benford-like)

y<-rnorm(nn[1], 150000, 30000)

}

else if(j==5){ # scenario 5

nn<-c(250, 200, 250) # sum up to 1000 # non-Benford ratios

x<-rnorm(nn[1], 100000, 30000) # centered cluster (usually non-Benford)

y<-rnorm(nn[1], 100000, 30000)

}

bb1<-data.frame(x=x, y=y) # clustered point-pattern, parameters in if()

bb1<-bb1[1:nn[1],]

b2<-spsample(xy.poly.merc, n=nn[2], "regular") # regular point-pattern

bb2<-data.frame(x=b2@coords[,1], y=b2@coords[,2])

bb2<-bb2[1:nn[2],]

b3<-spsample(xy.poly.merc, n=nn[3], "random") # random point-pattern

bb3<-data.frame(x=b3@coords[,1], y=b3@coords[,2])

bb3<-bb3[1:nn[3],]

bb4<-rbind(bb1, bb2, bb3) # all points in one object

bb4<-bb4[1:1000,] # limiting to 1000 obs.

# generating population data

x1<-rgamma(param[1,3], 1, 1)

x2<-rgamma(param[2,3], 1, 1)

x3<-rgamma(param[3,3], 1, 1)

# rescaling population data

if(j==1) {# Scenario 1

x1<-(x1-1)\*param[1,2]+param[1,1]

x2<-(x2-1)\*param[2,2]+param[2,1]

x3<-as.numeric((x3-1)\*param[3,2]+param[3,1])

}

else if(j==2){ # Scenario 2

x1<-(x1-1)\*param[1,2]+param[1,1]

x2<-(x2-1)\*param[2,2]+param[2,1]

x3<-as.numeric((x3-1)\*param[3,2]+param[3,1])

}

else if(j==3){ # scenario 3

x1<-(x1-1)\*param[1,2]+param[1,1]

x2<-(x2-1)\*param[2,2]+param[2,1]

x3<-as.numeric((x3-1)\*param[3,2]+param[3,1])

}

else if(j==4){ # scenario 4

x1<-(x1+1)\*param[1,2]+param[1,1]

x2<-(x2+1)\*param[2,2]+param[2,1]

x3<-as.numeric((x3+1)\*param[3,2]+param[3,1])

}

else if(j==5){ # scenario 5

x1<-(x1+1)\*param[1,2]+param[1,1]

x2<-(x2+1)\*param[2,2]+param[2,1]

x3<-as.numeric((x3+1)\*param[3,2]+param[3,1])

}

xxx<-c(x1, x2, x3) # merging population in one object

# values assigned logically to locations

# big values go to (random) or (random and regular) locations,

# clustered locations always with small values

bb4$z<-xxx

# values assigned randomly to locations

# please unhash to activate

#xxx<-data.frame(pop=xxx, ID=runif(1000, 0,1))

#xxo<-orderBy(~ID, xxx)

#bb4$z<-xxo$pop

# benford 1D - popul

dist.xxx<-as.vector(as.matrix(dist(as.data.frame(xxx))))

bb.dist.pop<-benford(dist.xxx)

beny[i,1]<-bb.dist.pop$MAD.conformity

beny[i,2]<-bb.dist.pop$MAD

# benford 2d - geo

dist.bb4<-as.vector(as.matrix(dist(as.data.frame(x=bb4[,1], y=bb4[,2]))))# distances

bb.dist.geo<-benford(dist.bb4)

beny[i,3]<-bb.dist.geo$MAD.conformity

beny[i,4]<-bb.dist.geo$MAD

# benford 3d – soc-geo

dist.all<-as.vector(as.matrix(dist(as.data.frame(bb4))))

benf.3d<-benford(dist.all)

beny[i,5]<-benf.3d$MAD.conformity

beny[i,6]<-benf.3d$MAD

}

assign(paste("beny", j, sep=""), beny) # saving simulations into individual objects

} # end of simulation

beny.all<-as.data.frame(rbind(beny1, beny2, beny3, beny4, beny5))

beny.all[,2]<-round(as.numeric(beny.all[,2]),5)

beny.all[,4]<-round(as.numeric(beny.all[,4]),5)

beny.all[,6]<-round(as.numeric(beny.all[,6]),5)

beny.all[,7]<-round(as.numeric(beny.all[,7]),3)

beny.all

**# saving results of simulation into single file**

write.table(beny.all, file="sim\_model\_all\_2.txt") # for logically assigned values

write.table(beny.all, file="sim\_model\_all\_3.txt") # for randomly assigned values

## # Figure 10 - relations in simulated data - between MAD statistics of 1D, 2D and 3D distances

# data can be obtained from the above simulation

# or taken from Author’s file – as below

beny.all.logic<-read.table("sim\_model\_all\_2.txt", sep=" ", dec=".", header=TRUE)

beny.all.random<-read.table("sim\_model\_all\_3.txt", sep=" ", dec=".", header=TRUE)

**# relations between data - for values logically assigned**

ggpairs(beny.all.logic[,c(2,4,6)]) # from GGally::

ggpairs(beny.all.random[,c(2,4,6)]) # from GGally::

