

# Geostatystyka

*Jakub Nowosad*

*2016-03-31*



# Contents

<b>1 Wprowadzenie</b>	<b>5</b>
1.1 Wymagania wstępne . . . . .	5
<b>2 R a dane przestrzenne</b>	<b>7</b>
2.1 R a dane przestrzenne . . . . .	7
2.2 Import danych . . . . .	9
2.3 Eksport danych . . . . .	11
2.4 Wizualizacja danych 2D . . . . .	12
2.5 Tworzenie siatek . . . . .	15
<b>3 Eksploracja analiza danych nieprzestrzennych</b>	<b>19</b>
3.1 Eksploracyjna analiza danych   Cel . . . . .	19
3.2 Dane . . . . .	19
3.3 Statystyki opisowe . . . . .	20
3.4 Wykresy . . . . .	22
3.5 Porównanie zmiennych . . . . .	27
3.6 Transformacje danych . . . . .	31
<b>4 Eksploracyjna analiza danych przestrzennych</b>	<b>41</b>
4.1 Mapy . . . . .	41
4.2 Próbkowanie . . . . .	41
4.3 Rozgrupowanie danych . . . . .	45
<b>5 Metody interpolacji</b>	<b>55</b>
5.1 Modele deterministyczne . . . . .	55
5.2 Modele statystyczne . . . . .	62
<b>6 Geostatystyka - prolog</b>	<b>63</b>
6.1 Geostatystyka . . . . .	63
6.2 Przestrzenna kowariancja, korelacja i semiwariancja . . . . .	65
6.3 Anizotropia . . . . .	74

<b>7 Modelowanie matematycznie autokorelacji przestrzennej</b>	<b>79</b>
7.1 Modelowanie matematycznie autokorelacji przestrzennej . . . . .	79
7.2 Modele podstawowe . . . . .	79
7.3 Metody modelowania . . . . .	82
7.4 Modelowanie semiwariogramu . . . . .	82
7.5 Modelowanie izotropowe . . . . .	82
7.6 Modelowanie anizotropowe . . . . .	99
<b>8 Estymacje jednozmienne</b>	<b>101</b>
8.1 Kriging . . . . .	101
8.2 Kriging prosty . . . . .	101
8.3 Kriging zwykły . . . . .	103
8.4 Kriging z trendem . . . . .	104
8.5 Porównanie wyników SK, OK i KZT . . . . .	109
<b>9 Estymacja lokalnego rozkładu prawdopodobieństwa</b>	<b>111</b>
9.1 Kriging danych kodowanych . . . . .	111
9.2 Kriging danych kodowanych   Przykłady . . . . .	112
<b>10 Estymacje wielozmienne</b>	<b>119</b>
10.1 Kokriging (prosty i zwykły, SCK i OCK) . . . . .	119
10.2 Krossemiwariogramy . . . . .	120
10.3 Modelowanie krossemiwariogramów . . . . .	121
10.4 Kokriging . . . . .	122
<b>11 Wykorzystanie do estymacji danych uzupełniających</b>	<b>123</b>
11.1 Kriging stratyfikowany . . . . .	123
11.2 Prosty kriging ze zmiennymi średnimi lokalnymi (LVM) . . . . .	129
11.3 Kriging uniwersalny (ang. <i>Universal kriging</i> ) . . . . .	131
<b>12 Ocena jakości estymacji</b>	<b>139</b>
12.1 Statystyki jakości estymacji . . . . .	139
12.2 Walidacja wyników estymacji . . . . .	142
<b>13 Symulacje</b>	<b>153</b>
13.1 Symulacje geostatystyczne . . . . .	153
13.2 Typy symulacji . . . . .	153
13.3 Symulacje bezwarunkowe . . . . .	154
13.4 Symulacje warunkowe . . . . .	155
13.5 Sekwencyjna symulacja danych kodowanych . . . . .	162

# Chapter 1

## Wprowadzenie

Znajdujesz się właśnie na stronie zawierającej materiały do ćwiczeń z geostatystyki. Składa się ona z kilkunastu rozdziałów pokazujących jak: dodawać i wizualizować dane przestrzenne w R [r-a-dane-przestrzenne], wykonywać wstępna eksplorację danych nieprzestrzennych [eksploracja-analiza-danych-nieprzestrzennych], analizować wstępnie dane przestrzenne [eksploracyjna-analiza-danych-przestrzennych], wykorzystywać deterministyczne metody interpolacji [metody-interpolacji], rozumieć i tworzyć przestrzenne miary podobieństwa i niepodobieństwa [geostatystyka-prolog], modelować semiwariogramy bezkierunkowe i kierunkowe [modelowanie-matematycznie-autokorelacji-przestrzennej], tworzyć estymacje jednozmienne [estymacje-jednozmienne], estymacje danych kodowanych [estymacja-lokalnego-rozkadu-prawdopodobienstwa], estymacje wielozmienne [estymacje-wielozmienne], estymacje wykorzystujące dane uzupełniające [wykorzystanie-do-estymacji-danych-uzupelniajacych], oraz tworzyć symulacje przestrzenne symulacje. Wszystkie zaprezentowane przykłady zawierają również kod w języku R.

### 1.1 Wymagania wstępne

Do odtworzenia przykładów użytych na poniższej stronie konieczna jest podstawowa znajomość R. Aby zainstalować R oraz RStudio można skorzystać z poniższych odnośników:

- R - <https://cloud.r-project.org/>
- RStudio - <https://www.rstudio.com/products/rstudio/download/>

Dane wykorzystywane na tej stronie można pobrać korzystając ze spakowanego archiwum (dla rozdziału drugiego) oraz korzystając z pakietu **geostatbook** (dla kolejnych rozdziałów). Dodatkowo, przy instalacji pakietu **geostatbook** pobierane są wszystkie inne pakiety potrzebne do pełnego korzystania z materiałów zawartych na tej stronie.

- Rozdział drugi
- Kolejne rozdziały:

```
# install.packages("devtools")
devtools::install_github("nowosad/geostatbook")
```



# Chapter 2

## R a dane przestrzenne

### 2.1 R a dane przestrzenne

#### 2.1.1 Pakiety

- GIS - `sp`, `rgdal`, `raster`, `rasterVis`, `rgeos`, `maptools`, `GeoXp`, `deldir`, `pgirmess`
- Geostatystyka - `gstat`, `geoR`, `geoRglm`, `fields`, `spBayes`, `RandomFields`, `vardiag`
- Inne - `ggplot2`, `corrplot`, `caret`

```
pakiety <- c('caret', 'corrplot', 'dismo', 'fields', 'geoR', 'ggplot2', 'gridExtra', 'gstat', 'pgirmess')

install.packages(pakiety)
```

Pakiety R oraz zbiory danych używane w tej książce można również zainstalować poprzez:

```
devtools::install_github("nowosad/geostatbook")
```

#### 2.1.2 Reprezentacja danych nieprzestrzennych

- Wektory (ang. *vector*):
  - liczbowe (ang. *integer*, *numeric*) - `c(1, 2, 3)` i `c(1.21, 3.32, 4.43)`
  - znakowe (ang. *character*) - `c('jeden', 'dwa', 'trzy')`
  - logiczne (ang. *logical*) - `c(TRUE, FALSE)`
  - czynnikowe (ang. *factor*) - `c('jeden', 'dwa', 'trzy', 'jeden')`
- Ramki danych (ang. *data.frame*) - to zbiór zmiennych (kolumn) oraz obserwacji (wierszy) zawierających różne typy danych
- Macierze (ang. *matrix*)
- Listy (ang. *list*)

#### 2.1.3 Reprezentacja danych przestrzennych

- Obiekty klasy `Spatial*` z pakietu `sp` - wszystkie z nich zawierają dwie dodatkowe informacje:
  - bounding box (`bbox`) - obwiednia - określa zasięg danych

- CRS (`proj4string`) - układ współrzędnych
- Najczęściej stosowane obiekty klasy `Spatial*` to `SpatialPointsDataFrame`, `SpatialPolygonsDataFrame` oraz `SpatialGridDataFrame`
- Obiekty klasy `Raster*` z pakietu `raster`, tj. `RasterLayer`, `RasterStack`, `RasterBrick`
- Inne

## 2.1.4 GDAL/OGR

- <http://www.gdal.org/>
- GDAL to biblioteka zawierająca funkcje służące do odczytywania i zapisywania danych w formatach rastrowych
- OGR to biblioteka służąca do odczytywania i zapisywania danych w formatach wektorowych
- Pakiet `rgdal` pozwala na wykorzystanie bibliotek GDAL/OGR w R

## 2.1.5 PROJ.4

- Dane przestrzenne powinny być zawsze powiązane z układem współrzędnych
- PROJ.4 - to biblioteka pozwalająca na identyfikację oraz konwersję pomiędzy różnymi układami współrzędnych
- Strona <http://www.spatialreference.org/> zawiera bazę danych układów współrzędnych

## 2.1.6 EPSG

- Kod EPSG (ang. *European Petroleum Survey Group*) pozwala na łatwe identyfikowanie układów współrzędnych
- Przykładowo, układ PL 1992 może być określony jako:

“+proj=tmerc +lat\_0=0 +lon\_0=19 +k=0.9993 +x\_0=500000 +y\_0=-5300000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m +no\_defs”

- ...lub też za pomocą kodu EPSG:

“+init=epsg:2180”

## 2.1.7 Układ geograficzny

- Proporcje pomiędzy współrzędną oznaczającą długość geograficzną (X) a oznaczającą szerokość geograficzną (Y) nie są równe 1:1
- Wielkość oczka siatki jest zmienna
- Nie pozwala to na proste określanie odległości czy powierzchni
- Jednostka mapy jest abstrakcyjna
- Powyższe cechy układów geograficznych powodują, że do większości algorytmów w geostatystyce wykorzystywane są układy współrzędnych prostokątnych płaskich

## 2.2 Import danych

### 2.2.1 Dane punktowe (format csv)

```
dane_punktowe <- read.csv('dane/punkty.csv')

head(dane_punktowe)

##      srtm  clc      temp      ndvi      savi       x       y
## 1 175.7430    1 13.852222 0.6158061 0.4189449 750298.0 716731.6
## 2 149.8111    1 15.484209 0.5558816 0.3794864 753482.9 717331.4
## 3 272.8583    NA 12.760814 0.6067462 0.3745572 747242.5 720589.0
## 4 187.2777    1 14.324648 0.3756170 0.2386246 755798.9 718828.1
## 5 260.1366    1 15.908549 0.4598393 0.3087599 746963.5 717533.5
## 6 160.1416    2  9.941118 0.5600288 0.3453627 756801.6 720474.1
```

```
coordinates(dane_punktowe) <- ~x+y
summary(dane_punktowe)
```

```
## Object of class SpatialPointsDataFrame
## Coordinates:
##      min     max
## x 745592.5 756967.8
## y 712642.4 721238.7
## Is projected: NA
## proj4string : [NA]
## Number of points: 248
## Data attributes:
##      srtm        clc        temp        ndvi
## Min.   :146.5   Min.   :1.000   Min.   : 7.883   Min.   :0.2024
## 1st Qu.:191.5   1st Qu.:1.000   1st Qu.:12.003   1st Qu.:0.4636
## Median :217.9   Median :1.000   Median :14.941   Median :0.5154
## Mean   :214.9   Mean   :1.481   Mean   :15.273   Mean   :0.5047
## 3rd Qu.:239.5   3rd Qu.:2.000   3rd Qu.:17.630   3rd Qu.:0.5742
## Max.   :278.4   Max.   :4.000   Max.   :24.945   Max.   :0.6597
## NA's   :3       NA's   :5       NA's   :1       NA's   :1
##      savi
## Min.   :0.0824
## 1st Qu.:0.2935
## Median :0.3256
## Mean   :0.3176
## 3rd Qu.:0.3594
## Max.   :0.4404
## NA's   :1
```

```
proj4string(dane_punktowe) <- '+init=epsg:2180'
summary(dane_punktowe)
```

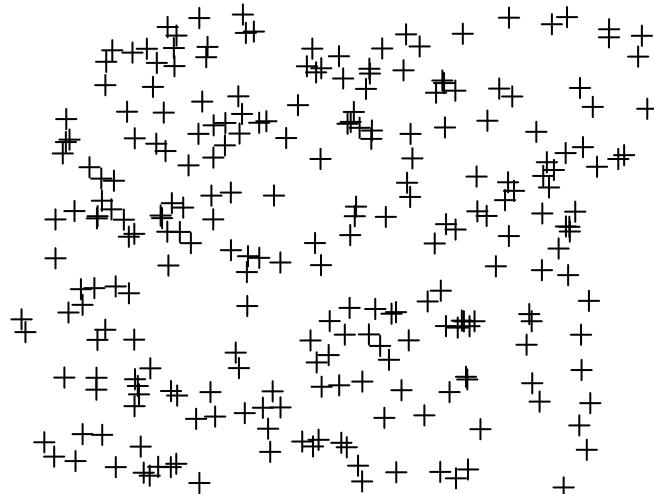
```
## Object of class SpatialPointsDataFrame
## Coordinates:
```

```

##           min         max
## x 745592.5 756967.8
## y 712642.4 721238.7
## Is projected: TRUE
## proj4string :
## [+init=epsg:2180 +proj=tmerc +lat_0=0 +lon_0=19 +k=0.9993
## +x_0=500000 +y_0=-5300000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0
## +units=m +no_defs]
## Number of points: 248
## Data attributes:
##          srtm          clc          temp          ndvi
## Min.   :146.5   Min.   :1.000   Min.   : 7.883   Min.   :0.2024
## 1st Qu.:191.5   1st Qu.:1.000   1st Qu.:12.003   1st Qu.:0.4636
## Median :217.9   Median :1.000   Median :14.941   Median :0.5154
## Mean   :214.9   Mean   :1.481   Mean   :15.273   Mean   :0.5047
## 3rd Qu.:239.5   3rd Qu.:2.000   3rd Qu.:17.630   3rd Qu.:0.5742
## Max.   :278.4   Max.   :4.000   Max.   :24.945   Max.   :0.6597
## NA's    :3       NA's    :5       NA's    :1       NA's    :1
##          savi
## Min.   :0.0824
## 1st Qu.:0.2935
## Median :0.3256
## Mean   :0.3176
## 3rd Qu.:0.3594
## Max.   :0.4404
## NA's    :1

```

```
plot(dane_punktowe)
```

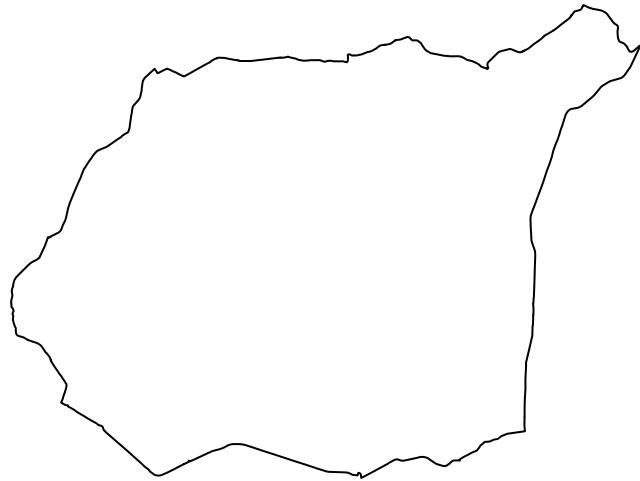


## 2.2.2 Dane poligonalne (formaty gisowe)

```

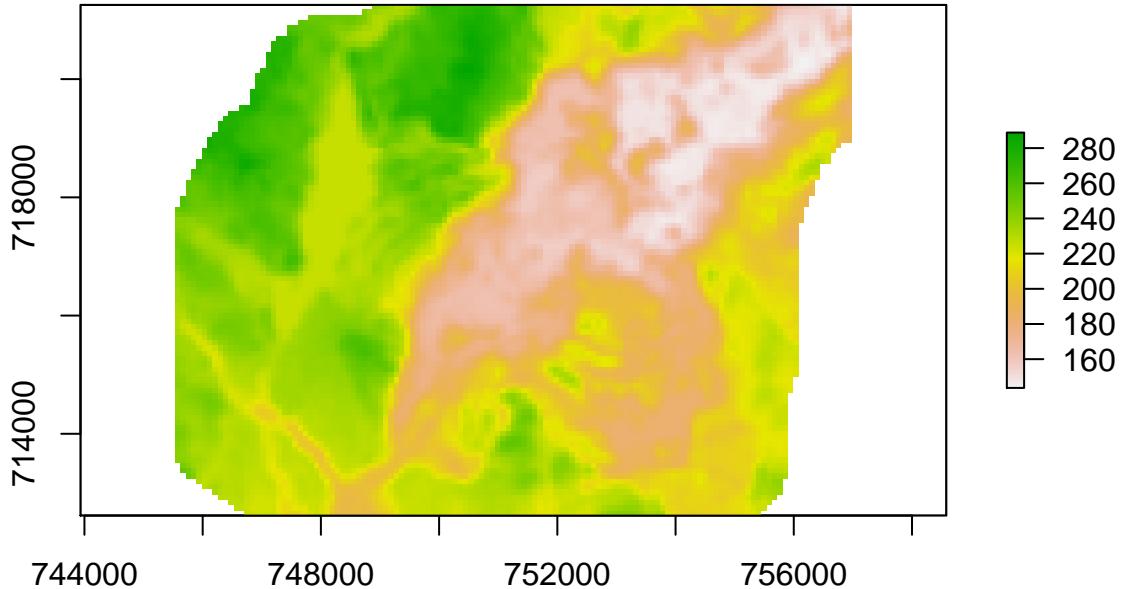
granica <- readOGR(dsn='dane', layer='granica', verbose=FALSE)
plot(granica)

```



### 2.2.3 Rastry

```
siatka_raster <- raster('dane/siatka.tif')
plot(siatka_raster)
```



## 2.3 Eksport danych

### 2.3.1 Zapisywanie danych wektorowych

```
writeOGR(polygon, dsn='nazwa_folderu', layer='nowy_poligon', driver='ESRI Shapefile')
```

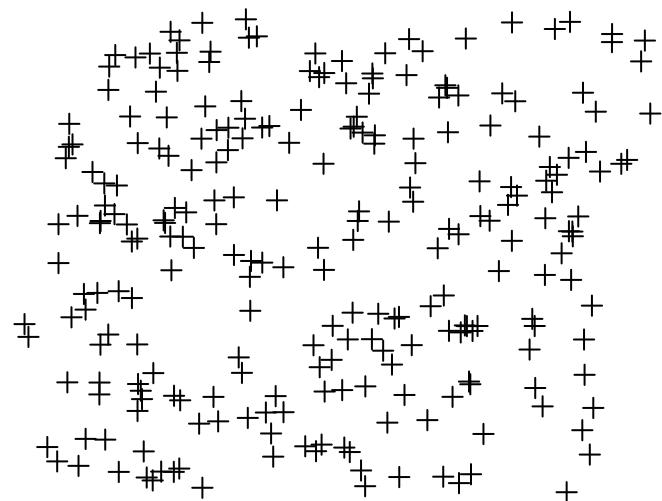
### 2.3.2 Zapisywanie danych rasterowych

```
writeRaster(siatka_raster, filename='nazwa_folderu/nowy_raster.tif')
```

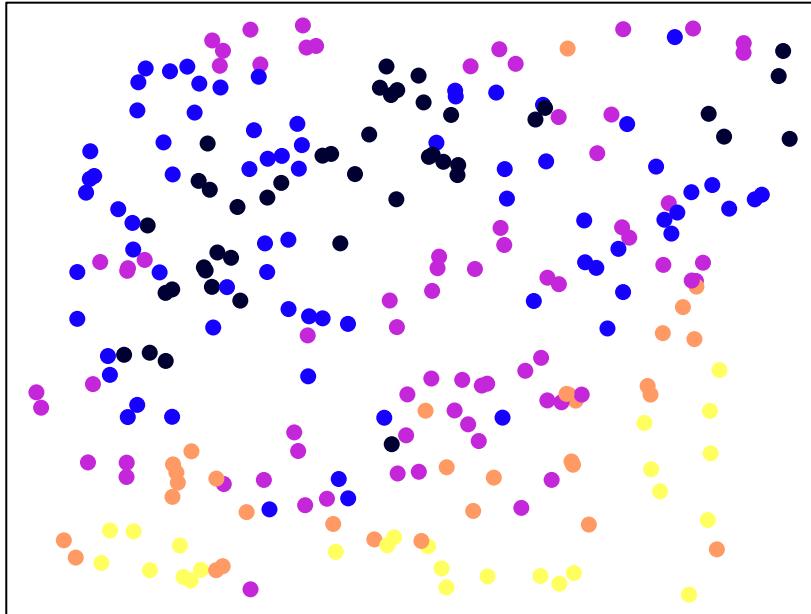
## 2.4 Wizualizacja danych 2D

### 2.4.1 Dane punktowe

```
plot(dane_punktowe)
```

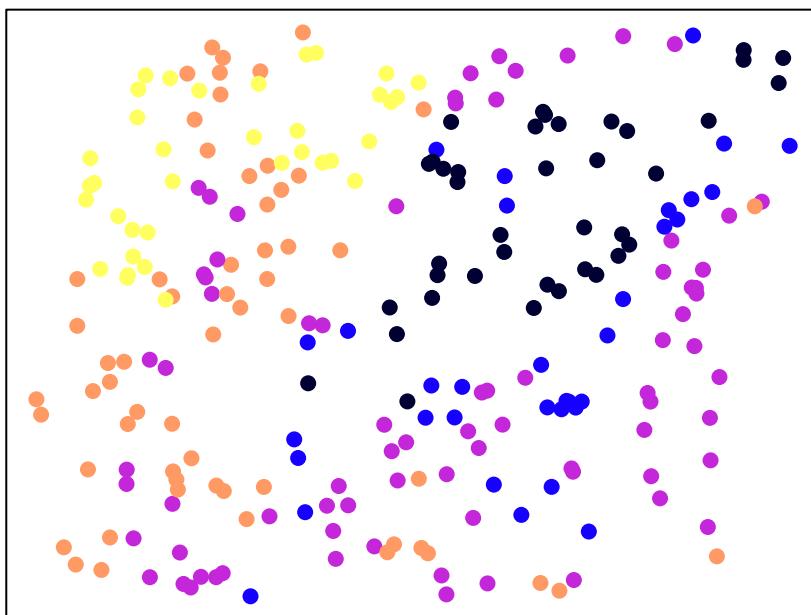


```
spplot(dane_punktowe, 'temp')
```



- $[7.883, 11.3]$
- $(11.3, 14.71]$
- $(14.71, 18.12]$
- $(18.12, 21.53]$
- $(21.53, 24.94]$

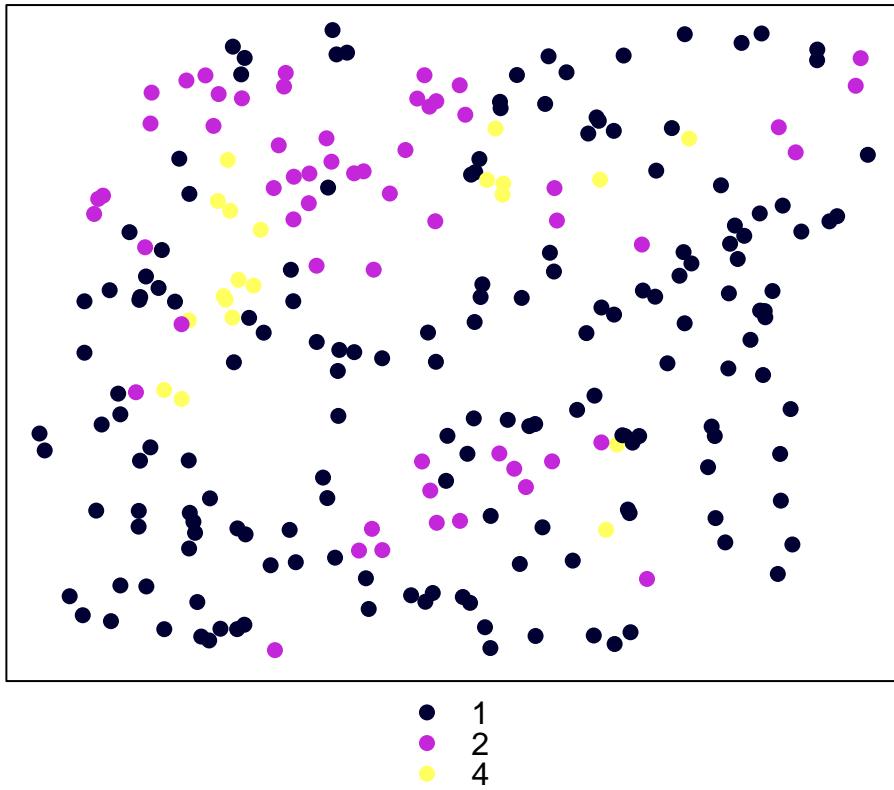
```
spplot(dane_punktowe, 'srtm')
```



- $[146.5, 172.9]$
- $(172.9, 199.3]$
- $(199.3, 225.6]$
- $(225.6, 252]$
- $(252, 278.4]$

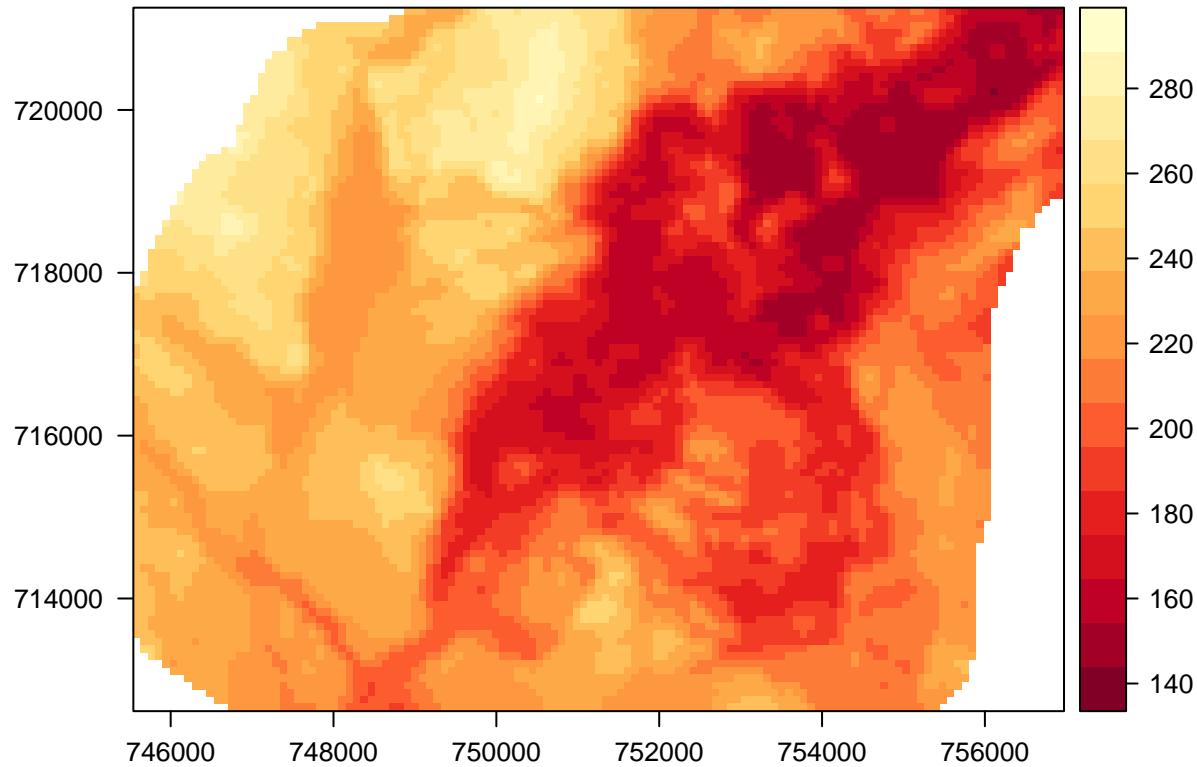
### 2.4.2 Dane punktowe - kategorie

```
dane_punktowe@data$clc <- as.factor(dane_punktowe@data$clc)
spplot(dane_punktowe, 'clc')
```



### 2.4.3 Rastry

```
levelplot(siatka_raster, margin=FALSE)
```



## 2.5 Tworzenie siatek

### 2.5.1 Siatki regularne

```
bbox(dane_punktowe)
```

```
##           min         max
## x 745592.5 756967.8
## y 712642.4 721238.7
```

```
extent(dane_punktowe)
```

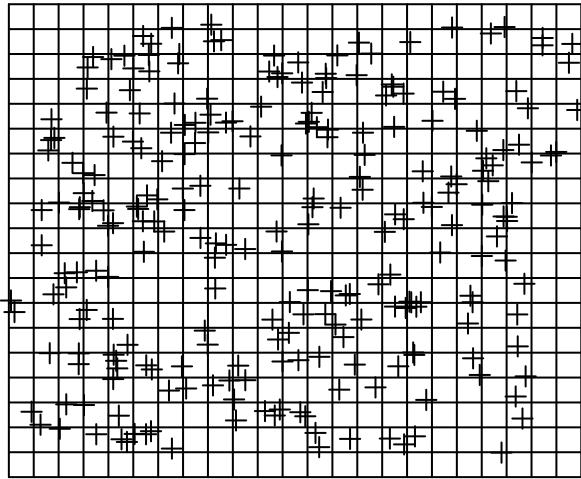
```
## class       : Extent
## xmin       : 745592.5
## xmax       : 756967.8
## ymin       : 712642.4
## ymax       : 721238.7
```

```
siatka <- expand.grid(x = seq(from = 745050, to = 757050, by = 500),
                      y = seq(from = 712650, to = 721650, by = 500))
coordinates(siatka) <- ~x + y
gridded(siatka) <- TRUE
proj4string(siatka) <- proj4string(dane_punktowe)
```

```
siatka <- makegrid(dane_punktowe, cellsize=500)
names(siatka) <- c('x', 'y')
coordinates(siatka) <- ~x + y
gridded(siatka) <- TRUE
proj4string(siatka) <- proj4string(dane_punktowe)
```

## 2.5.2 Siatki regularne

```
plot(siatka)
plot(dane_punktowe, add=TRUE)
```



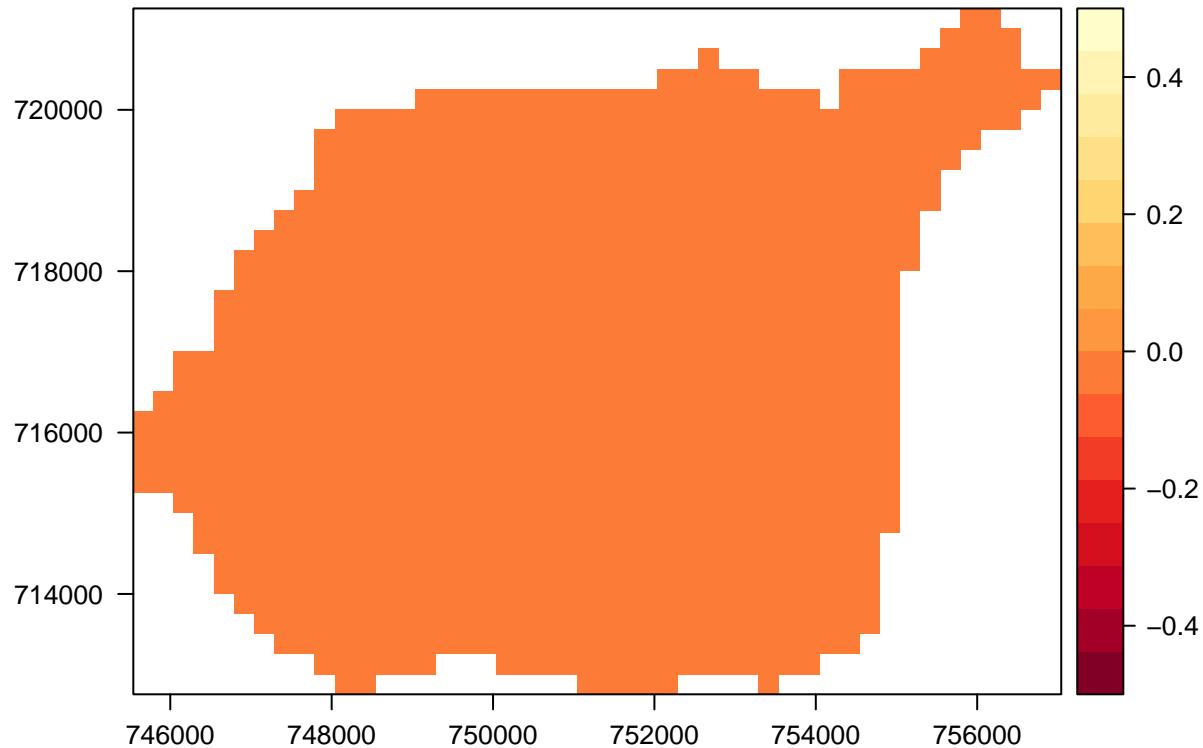
## 2.5.3 Siatki nieregularne - klasa RasterLayer

```
granica <- readOGR(dsn='dane', layer='granica')
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "dane", layer: "granica"
## with 1 features
## It has 3 fields
```

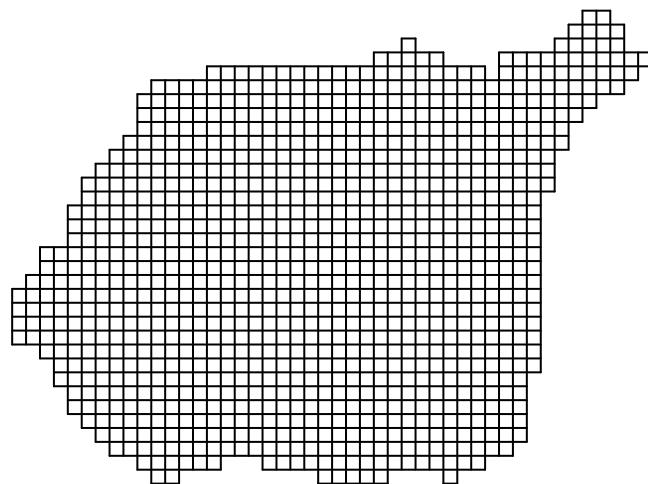
```
siatka_n <- raster(extent(granica))
res(siatka_n) <- c(250, 250)
siatka_n[] <- 0
proj4string(siatka_n) <- proj4string(granica)
siatka_n <- mask(siatka_n, granica)
```

```
levelplot(siatka_n, margin=FALSE)
```



#### 2.5.4 Siatki nieregularne - klasa SpatialPixelsDataFrame

```
siatka_n <- as(siatka_n, 'SpatialPixelsDataFrame')
siatka_n <- siatka_n[!is.na(siatka_n@data$layer), ]
gridded(siatka_n) <- TRUE
plot(siatka_n)
```





# Chapter 3

## Eksploracja analiza danych nieprzestrzennych

### 3.1 Eksploracyjna analiza danych | Cel

- Ogólna charakterystyka danych oraz badanego zjawiska
- Określenie przestrzennego/czasowego typu próbkowania
- Informacja o relacji pomiędzy lokalizacją obserwacji a czynnikami wpływającymi na zmienność przestrzenną cechy

### 3.2 Dane

#### 3.2.1 Dane

```
library('geostatbook')
data(punkty)

str(punkty)

## Formal class 'SpatialPointsDataFrame' [package "sp"] with 5 slots
##   ..@ data      :'data.frame': 250 obs. of  5 variables:
##     ...$ srtm: num [1:250] 231 185 269 212 209 ...
##     ...$ clc : num [1:250] 1 1 1 1 2 2 2 2 1 1 ...
##     ...$ temp: num [1:250] 22.5 16 16.1 17.1 22.3 ...
##     ...$ ndvi: num [1:250] 0.589 0.546 0.509 0.529 0.491 ...
##     ...$ savi: num [1:250] 0.357 0.382 0.342 0.345 0.22 ...
##   ..@ coords.nrs : num(0)
##   ..@ coords    : num [1:250, 1:2] 753638 749498 750131 751764 753676 ...
##   ...- attr(*, "dimnames")=List of 2
##   ...  ...$ : NULL
##   ...  ...$ : chr [1:2] "x" "y"
##   ..@ bbox      : num [1:2, 1:2] 745547 712619 756937 721193
##   ...- attr(*, "dimnames")=List of 2
##   ...  ...$ : chr [1:2] "x" "y"
```

```

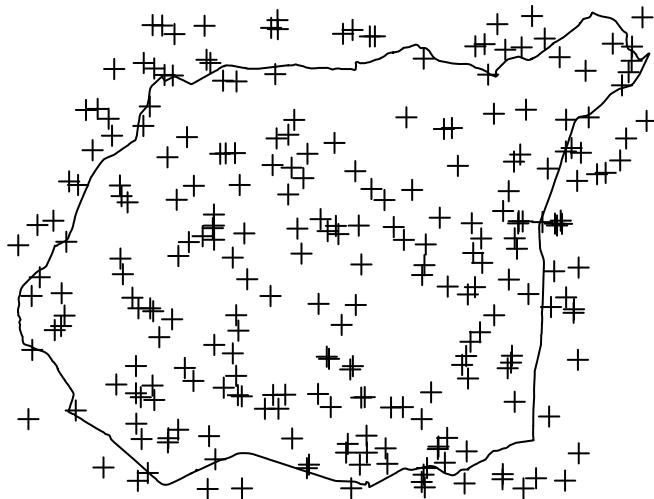
## ... . . . $ : chr [1:2] "min" "max"
## ..@ proj4string:Formal class 'CRS' [package "sp"] with 1 slot
## .....@ projargs: chr "+init=epsg:2180 +proj=tmerc +lat_0=0 +lon_0=19 +k=0.9993 +x_0=500000 +y_0=-530

str(punkty@data)

## 'data.frame':   250 obs. of  5 variables:
## $ srtm: num  231 185 269 212 209 ...
## $ clc : num  1 1 1 1 2 2 2 2 1 1 ...
## $ temp: num  22.5 16 16.1 17.1 22.3 ...
## $ ndvi: num  0.589 0.546 0.509 0.529 0.491 ...
## $ savi: num  0.357 0.382 0.342 0.345 0.22 ...

plot(punkty)
data(granica)
plot(granica, add=TRUE)

```



### 3.3 Statystyki opisowe

#### 3.3.1 Statystyki opisowe

```
summary(punkty@data)
```

```

##      srtm          clc          temp          ndvi
## Min.   :146.2   Min.   :1.000   Min.   :8.706   Min.   :0.2102
## 1st Qu.:191.0   1st Qu.:1.000   1st Qu.:13.284  1st Qu.:0.4567
## Median :218.2   Median :1.000   Median :15.309  Median :0.5037
## Mean   :213.4   Mean   :1.268   Mean   :15.950  Mean   :0.5039
## 3rd Qu.:236.4   3rd Qu.:1.000   3rd Qu.:18.273  3rd Qu.:0.5521
## Max.   :278.8   Max.   :4.000   Max.   :26.139  Max.   :0.6848
##      savi
## Min.   :0.08343

```

```
## 1st Qu.:0.29017
## Median :0.32212
## Mean    :0.31847
## 3rd Qu.:0.35292
## Max.   :0.48354
```

### 3.3.2 Statystyki opisowe | średnia i mediana

```
median(punkty$temp, na.rm=TRUE)
```

```
## [1] 15.30944
```

```
mean(punkty$temp, na.rm=TRUE)
```

```
## [1] 15.95036
```

- W wypadku symetrycznego rozkładu te dwie cechy są równe
- Średnia jest bardziej wrażliwa na wartości odstające
- Mediana jest lepszą miarą środka danych, jeżeli są one skośne

Po co używać średniej?

- Bardziej przydatna w przypadku małych zbiorów danych
- Gdy rozkład danych jest symetryczny
- (Jednak) często warto podawać obie miary

### 3.3.3 Statystyki opisowe | minimum i maksimum

```
min(punkty$temp, na.rm=TRUE)
```

```
## [1] 8.706485
```

```
max(punkty$temp, na.rm=TRUE)
```

```
## [1] 26.13947
```

### 3.3.4 Statystyki opisowe | ochylenie standardowe

```
sd(punkty$temp, na.rm=TRUE)
```

```
## [1] 3.839596
```

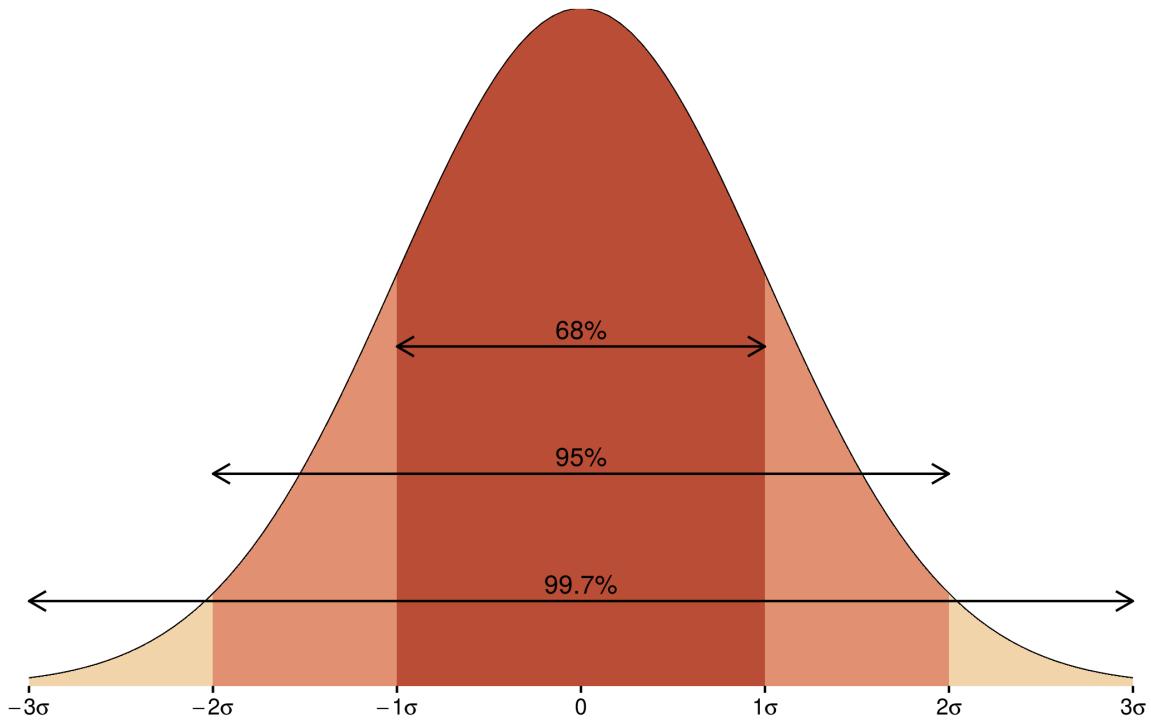
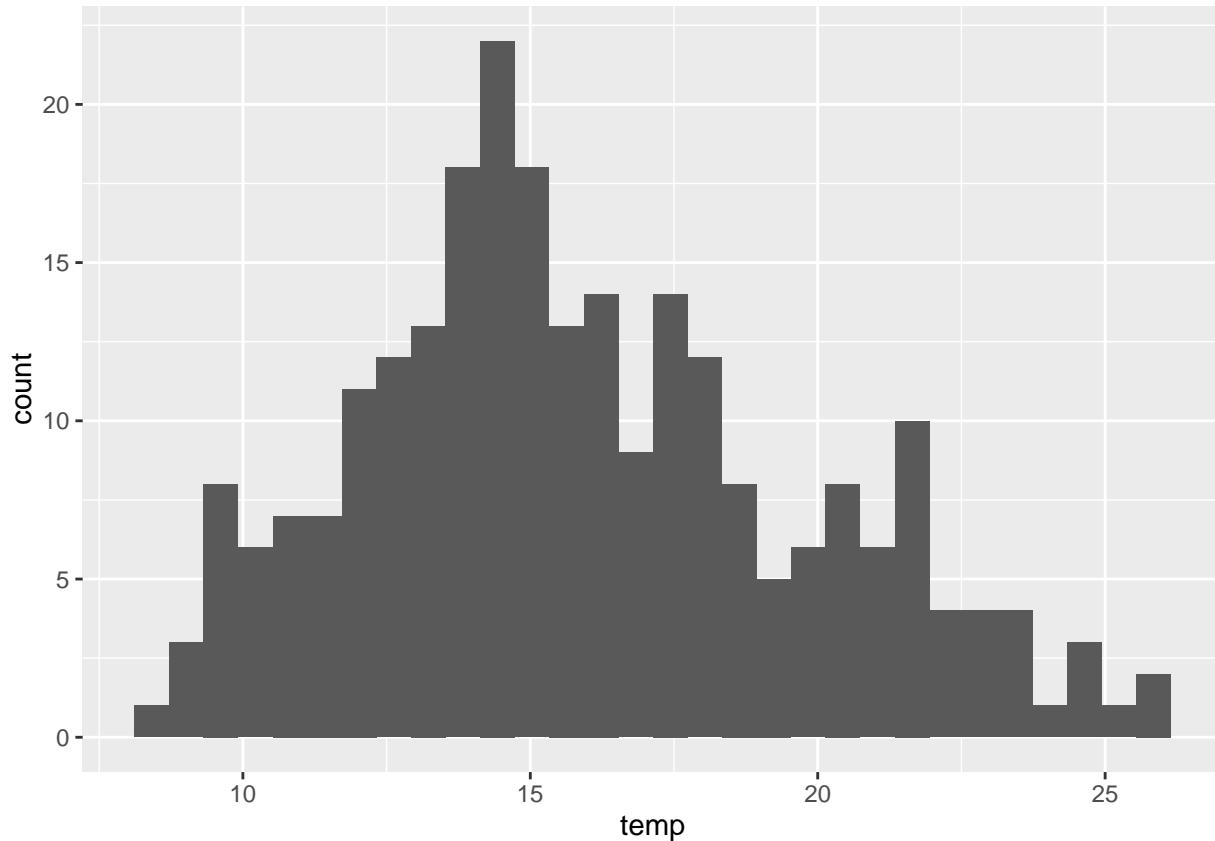


Figure 3.1:

## 3.4 Wykresy

### 3.4.1 Histogram

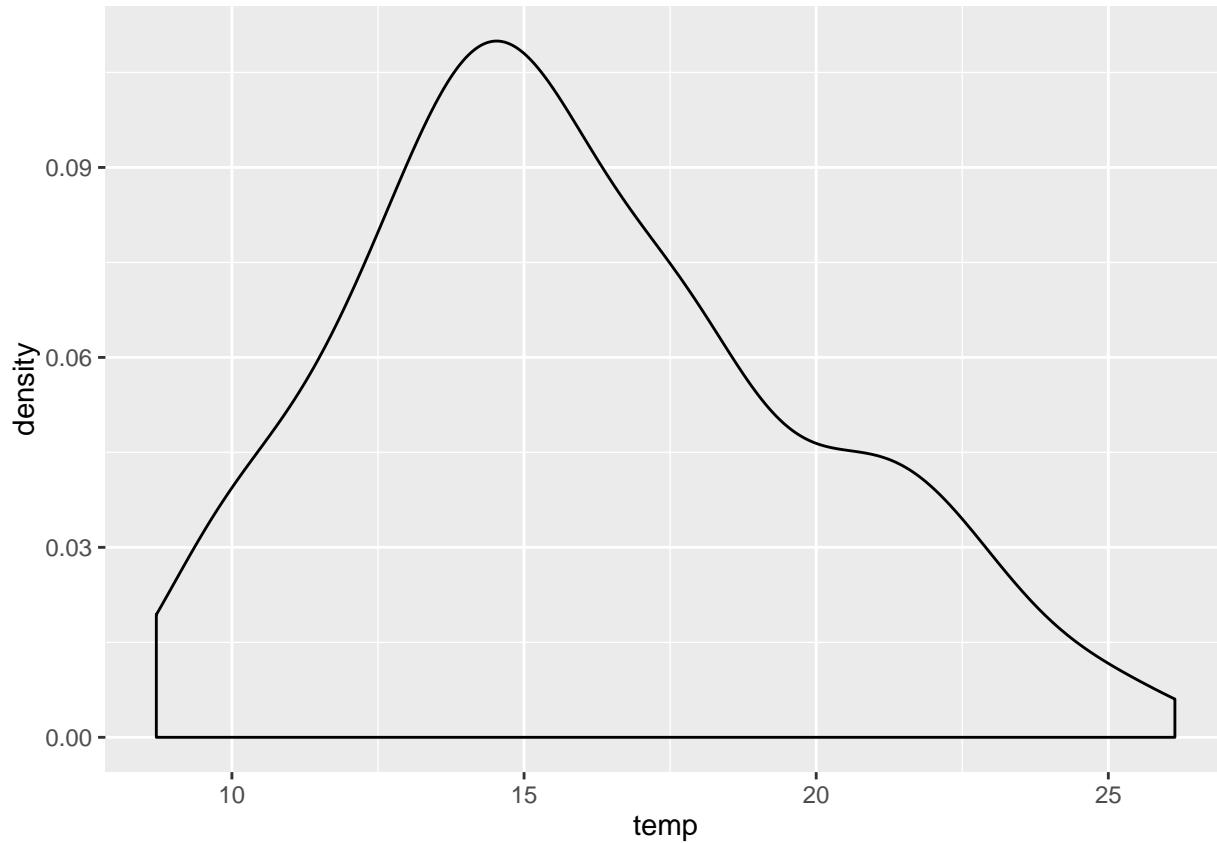
```
ggplot(punkty@data, aes(temp)) + geom_histogram()
```



- Stworzony przez Karla Pearsona
- Jest graficzną reprezentacją rozkładu danych
- Wartości danych są łączone w przedziały (na osi poziomej) a na osi pionowej jest ukazana liczba punktów (obserwacji) w każdym przedziale
- Różny dobór przedziałów może dawać inną informację
- W pakiecie `ggplot2`, domyślnie przedział to zakres/30

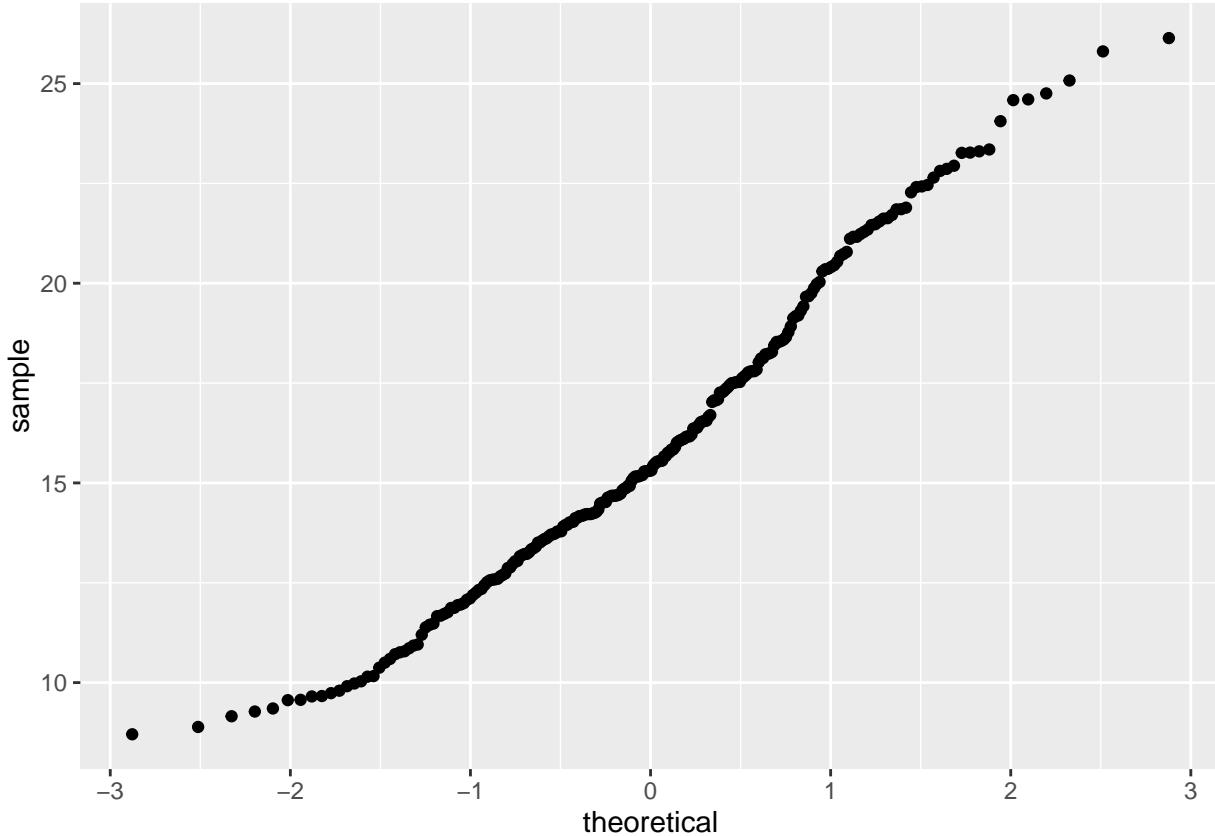
### 3.4.2 Estymator jądrowy gęstości (ang. *kernel density estimation*)

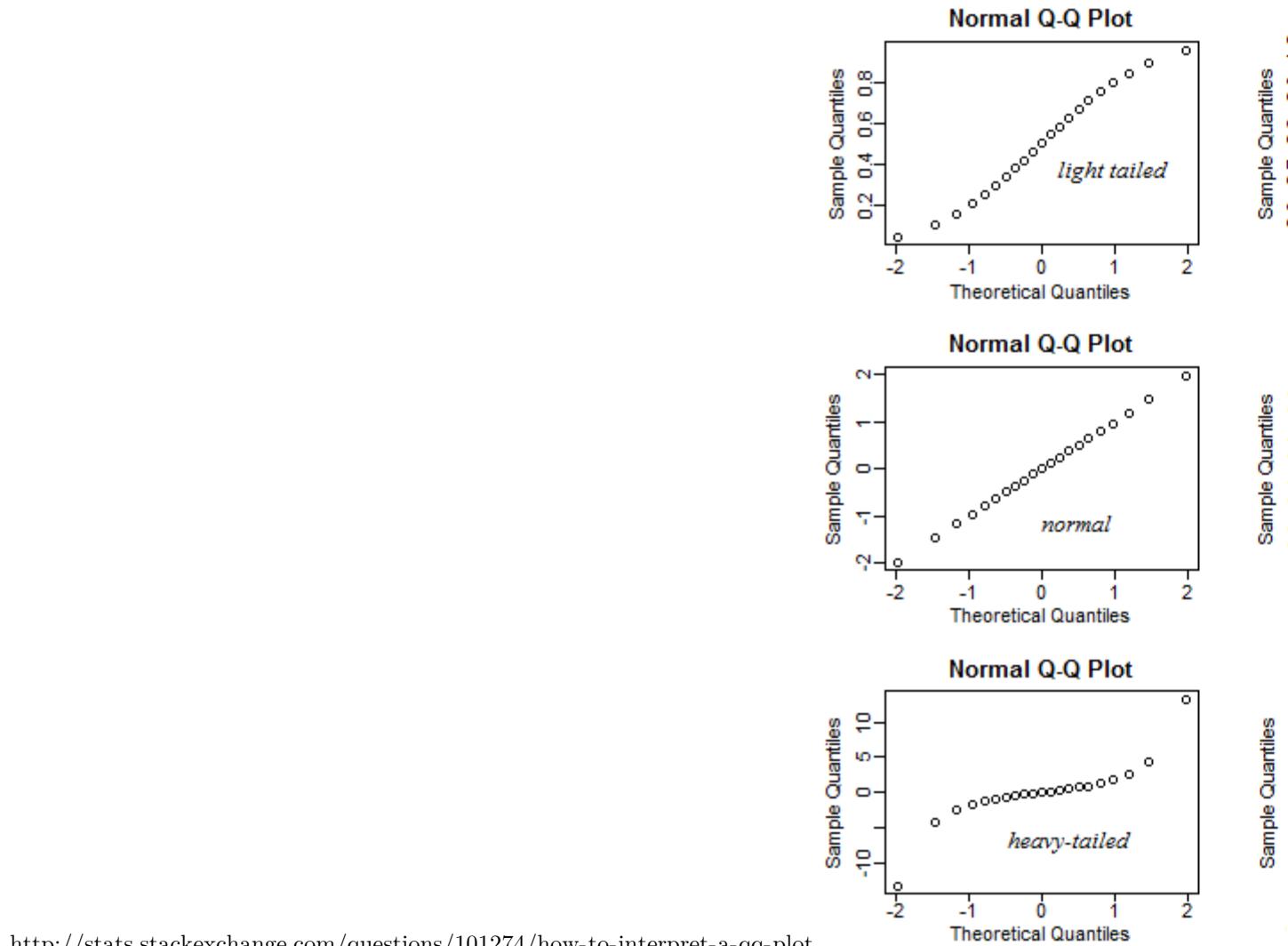
```
ggplot(punkty@data, aes(temp)) + geom_density()
```



### 3.4.3 Wykresy kwantyl-kwantyl (ang. *quantile-quantile*)

```
ggplot(punkty@data, aes(sample=temp)) + stat_qq()
```



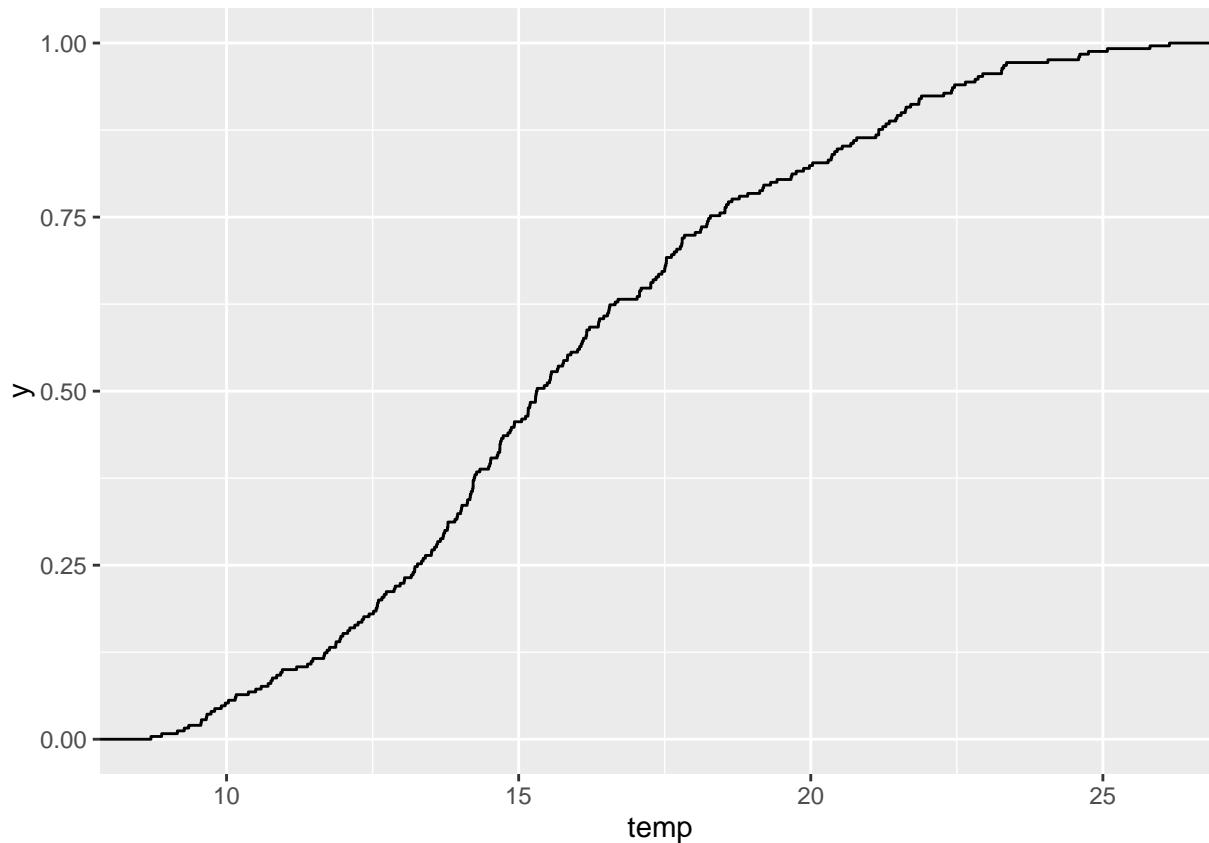


<http://stats.stackexchange.com/questions/101274/how-to-interpret-a-qq-plot>

### 3.4.4 Dystrybuanta (CDF)

- Dystrybuanta (ang. *conditional density function* - CDF) wyświetla prawdopodobieństwo, że wartość zmiennej przewidywanej jest mniejsza lub równa określonej wartości

```
ggplot(punkty@data, aes(temp)) + stat_ecdf()
```



## 3.5 Porównanie zmiennych

### 3.5.1 Kowariancja

- Kowariancja jest nieunormowaną miarą zależności liniowej pomiędzy dwiema zmiennymi
- Kowariancja dwóch zmiennych  $x$  i  $y$  pokazuje jak dwie zmienne są ze sobą liniowo powiązane
- Dodatnia kowariancja wskazuje na pozytywną relację liniową pomiędzy zmiennymi, podczas gdy ujemna kowariancja świadczy o odwrotnej sytuacji
- Jeżeli zmienne nie są ze sobą liniowo powiązane, wartość kowariacji jest bliska零
- Inaczej mówiąc, kowariancja stanowi miarę wspólnej zmienności dwóch zmiennych
- Wielkość samej kowariacji uzależniona jest od przyjętej skali zmiennej (jednostki)
- Inne wyniku uzyskamy (przy tej samej zależności pomiędzy parą zmiennych), gdy będziemy analizować wyniki np. wieku i dochodu w złotówkach a inne dla wieku i dochodu w dolarach

```
cov(punkty$temp, punkty$ndvi, use='complete.obs')
```

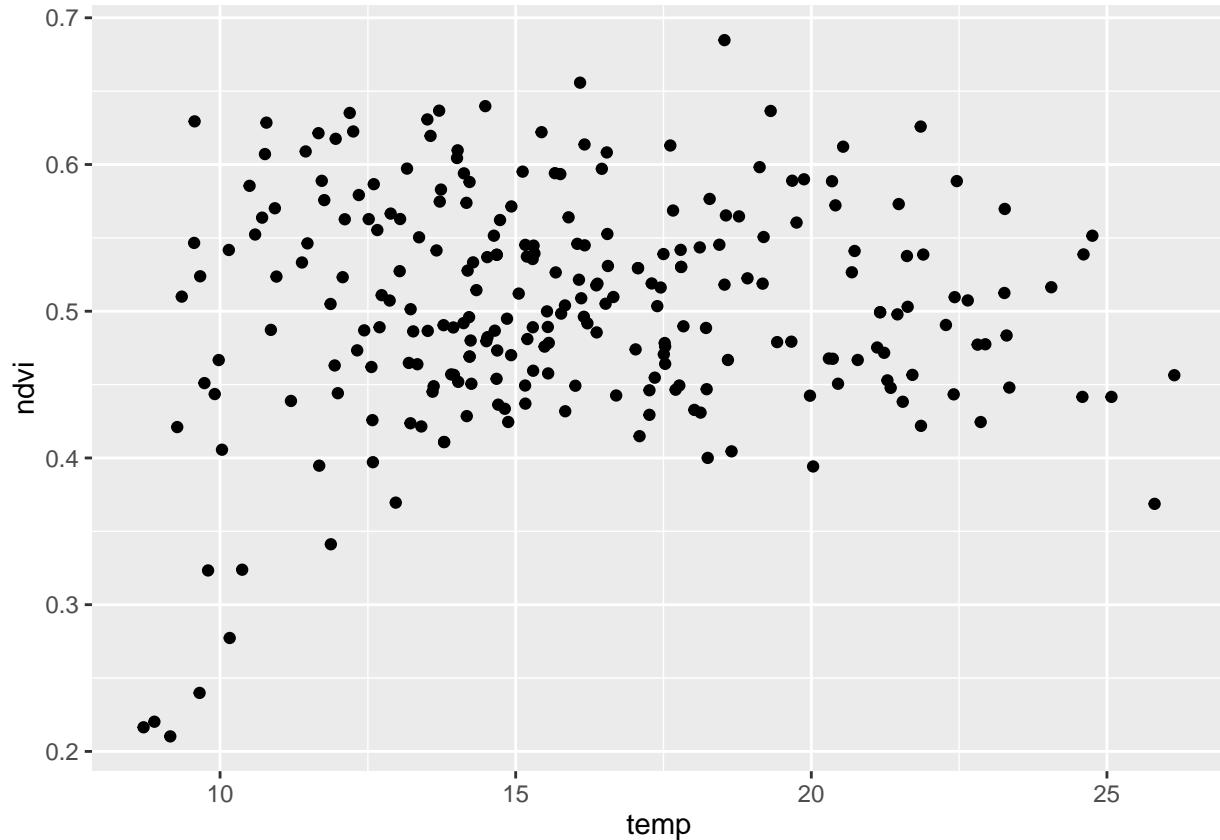
```
## [1] 0.02047509
```

### 3.5.2 Współczynnik korelacji

- Współczynnik korelacji to unormowana miara zależności pomiędzy dwiema zmiennymi, przyjmująca wartości od -1 do 1

- Współczynnik korelacji jest uzyskiwany poprzez podzielenie wartości kowariancji przez odchylenie standardowe wyników
- Z racji unormowania nie jest ona uzależniona od jednostki

```
ggplot(punkty$data, aes(temp, ndvi)) + geom_point()
```



```
cor(punkty$temp, punkty$ndvi, use='complete.obs')
```

```
## [1] 0.07049136
```

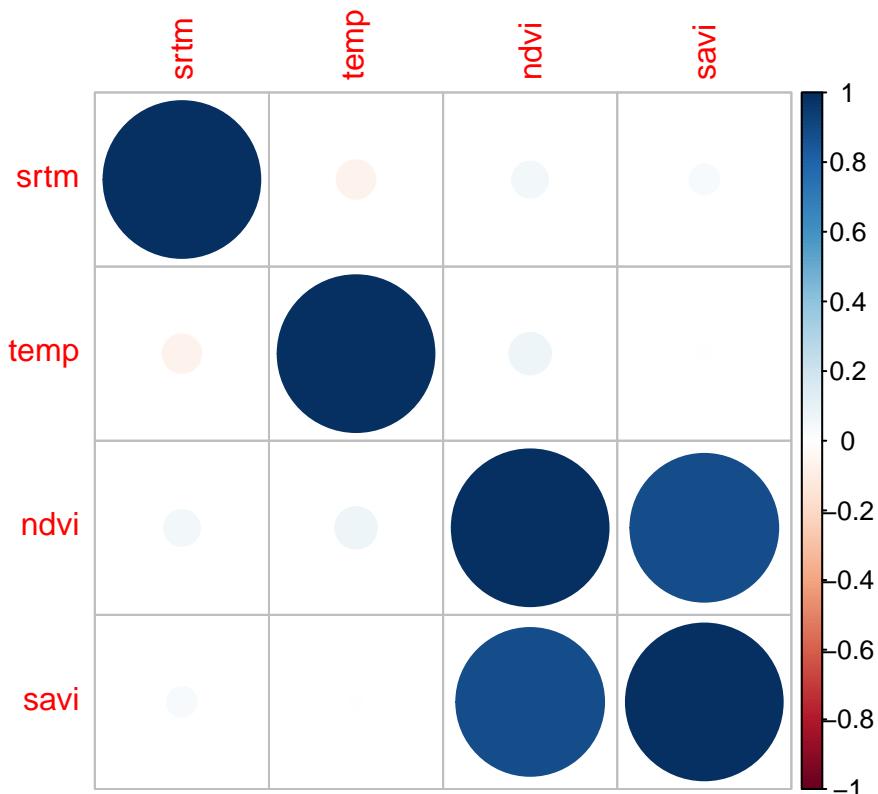
```
cor.test(punkty$temp, punkty$ndvi)
```

```
##
## Pearson's product-moment correlation
##
## data: punkty$temp and punkty$ndvi
## t = 1.1129, df = 248, p-value = 0.2668
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
## -0.05404834 0.19287158
## sample estimates:
##      cor
## 0.07049136
```

```
cor(punkty@data[c(1, 3:5)], use='complete.obs')
```

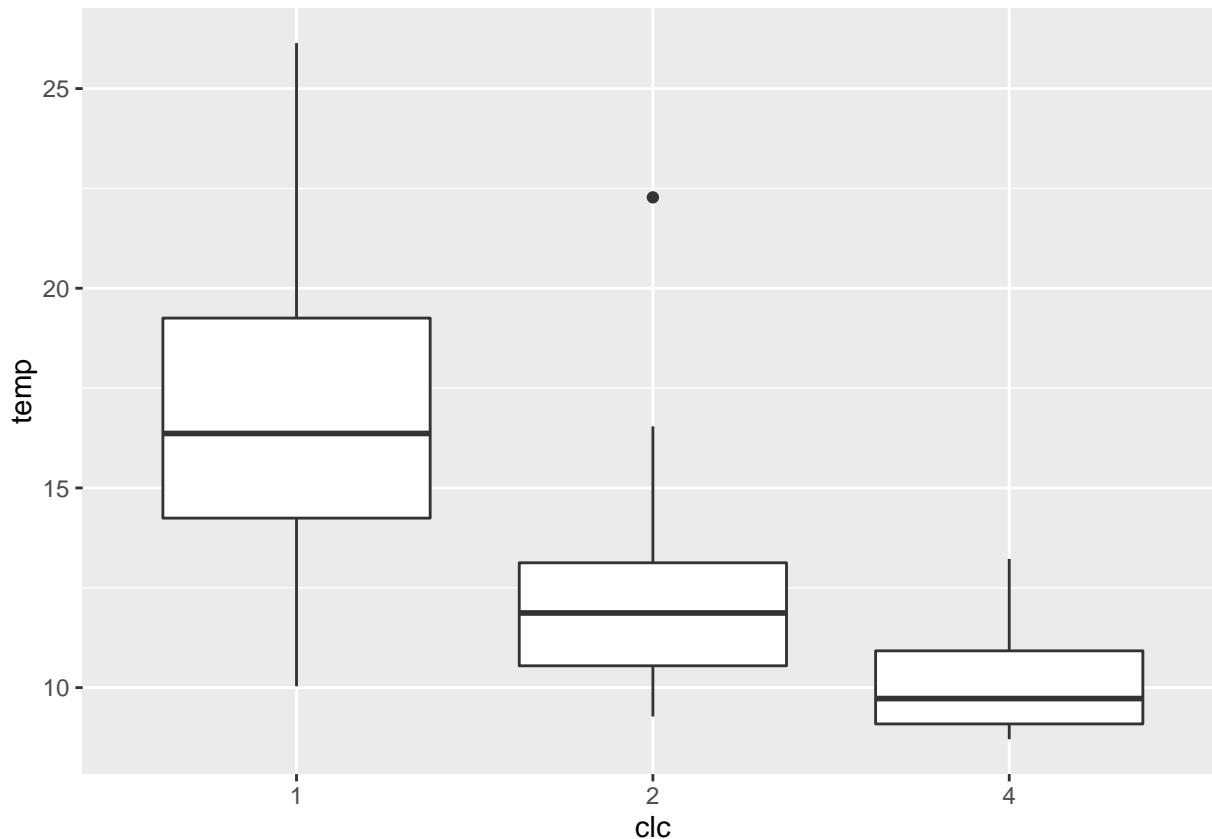
```
##          srtm      temp      ndvi      savi
## srtm  1.00000000 -0.060458939 0.05230326  0.035760652
## temp -0.06045894  1.000000000 0.07049136 -0.007748702
## ndvi  0.05230326  0.070491358 1.00000000  0.889037014
## savi  0.03576065 -0.007748702 0.88903701  1.000000000
```

```
corrplot(cor(punkty@data[c(1, 3:5)], use='complete.obs'))
```



### 3.5.3 Wykresy pudełkowe

```
punkty$clc <- as.factor(punkty$clc)
ggplot(punkty@data, aes(clc, temp)) + geom_boxplot()
```



- Obrazuje pięć podstawowych statystyk opisowych oraz wartości odstające
- Pudełko to zakres międzykwantylowy
- Linie oznaczają najbardziej ekstremalne wartości, ale nie odstające. Górną to  $1,5 \times \text{IQR}$  ponad krawędź pudełka, dolna to  $1,5 \times \text{IQR}$  poniżej wartości dolnej krawędzi pudełka
- Linia środkowa to mediana

### 3.5.4 Testowanie istotności różnic średniej pomiędzy grupami

```

punkty$clc <- as.factor(punkty$clc)
aov_test <- aov(temp~clc, data=punkty)
summary(aov_test)

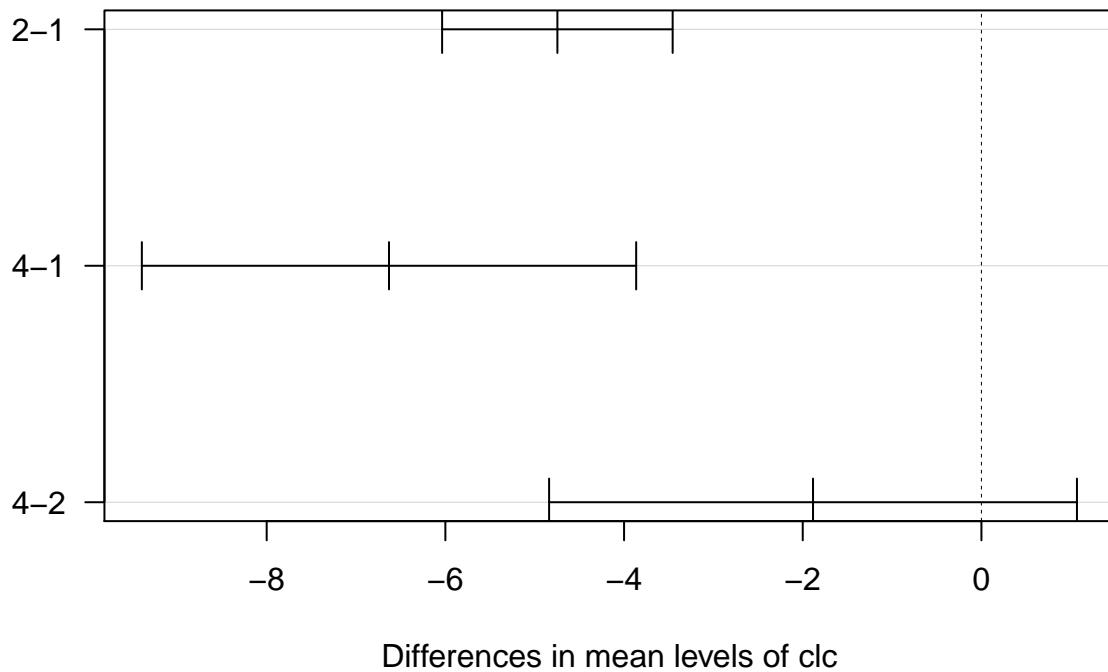
##           Df Sum Sq Mean Sq F value Pr(>F)
## clc        2   1056    528.0   49.87 <2e-16 ***
## Residuals 247   2615     10.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## null device
##               1

tukey <- TukeyHSD(aov_test, 'clc')
plot(tukey, las=1)

```

### 95% family-wise confidence level



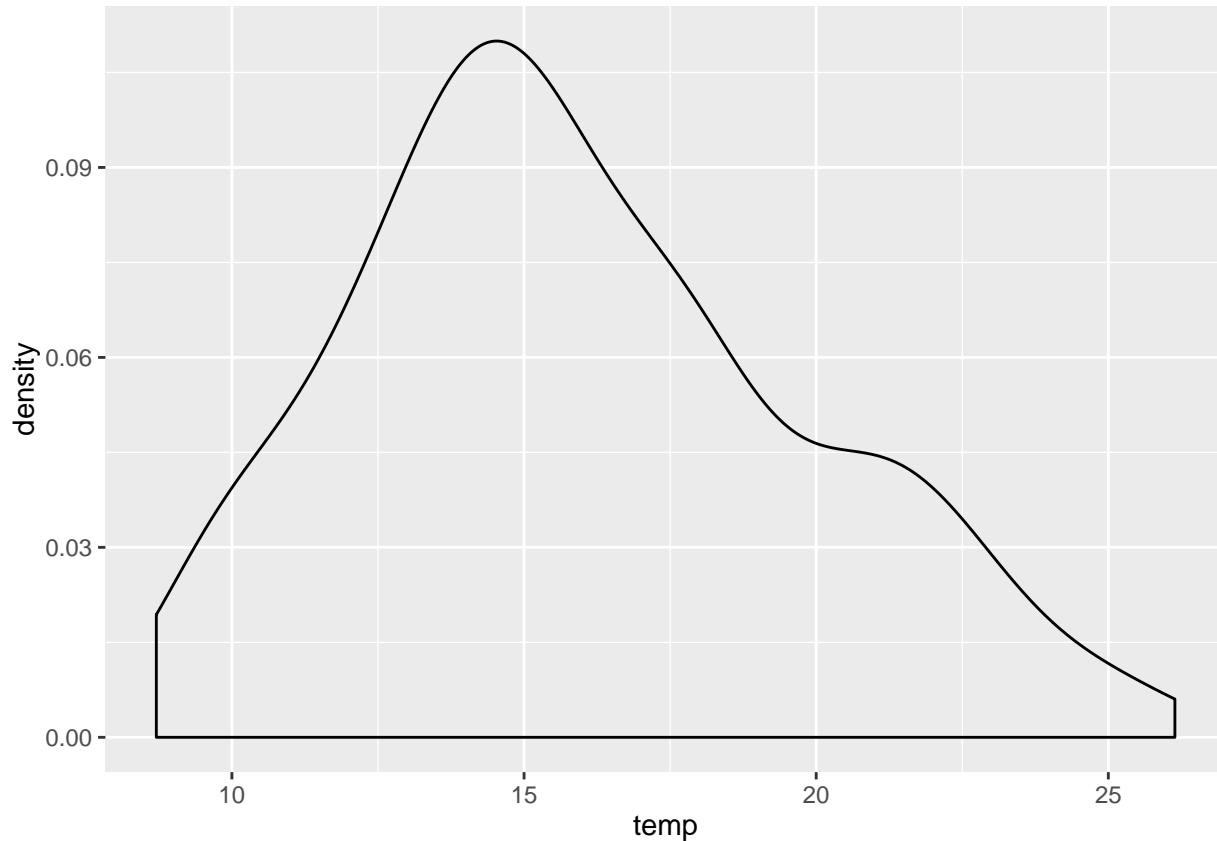
## 3.6 Transformacje danych

### 3.6.1 Transformacje danych

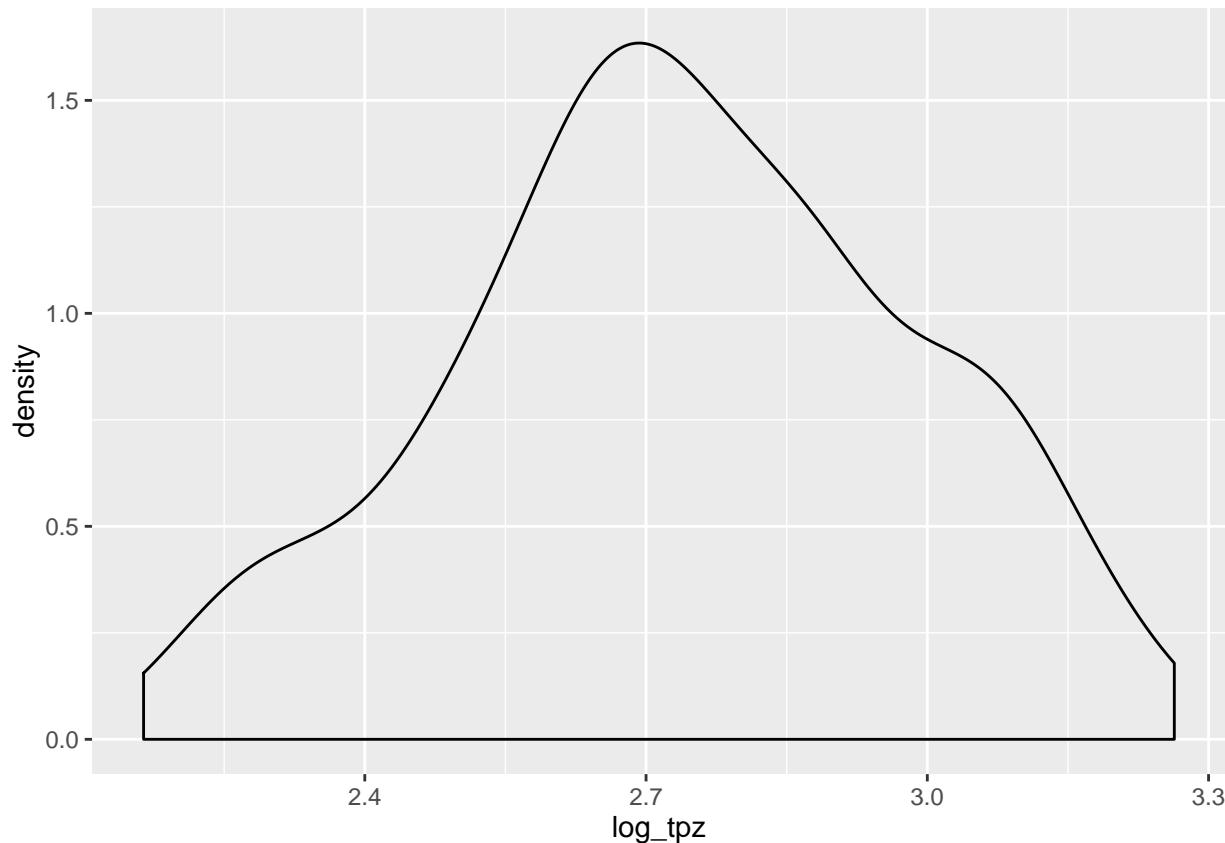
- Transformacja danych może mieć na celu ułatwienie porównywania różnych zmiennych, zniwelowanie skośności rozkładu lub też zmniejszenie wpływu danych odstających
- Centrowanie i skalowanie (standaryzacja):
  - Centrowanie danych - wybierana jest przeciętna wartość predyktora, a następnie od wszystkich wartości predyktorów odejmowana jest wybrana wcześniej wartość
  - Skalowanie danych - dzielenie każdej wartości predyktora przez jego odchylenie standardowe
  - Wadą tego podjęcia jest główne zmniejszenie interpretowalności pojedynczych wartości
- Redukcja skośności:
  - Logarytmizacja
  - Pierwiastkowanie
  - Rodzina transformacji Boxa Coxa
  - Inne

### 3.6.2 Transformacja danych | Logarytmizacja

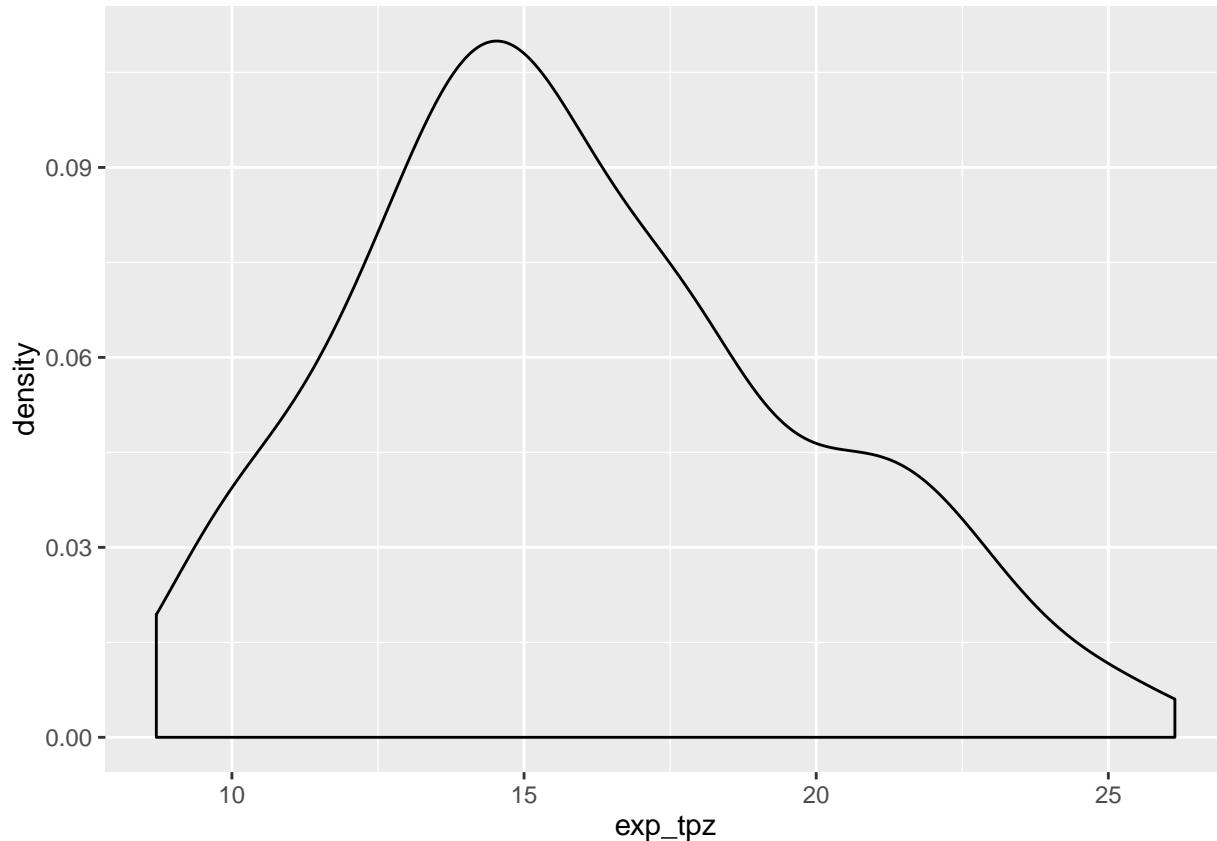
```
ggplot(punkty@data, aes(temp)) + geom_density()
```



```
punkty$log_tpz <- log(punkty$temp)
ggplot(punkty@data, aes(log_tpz)) + geom_density()
```

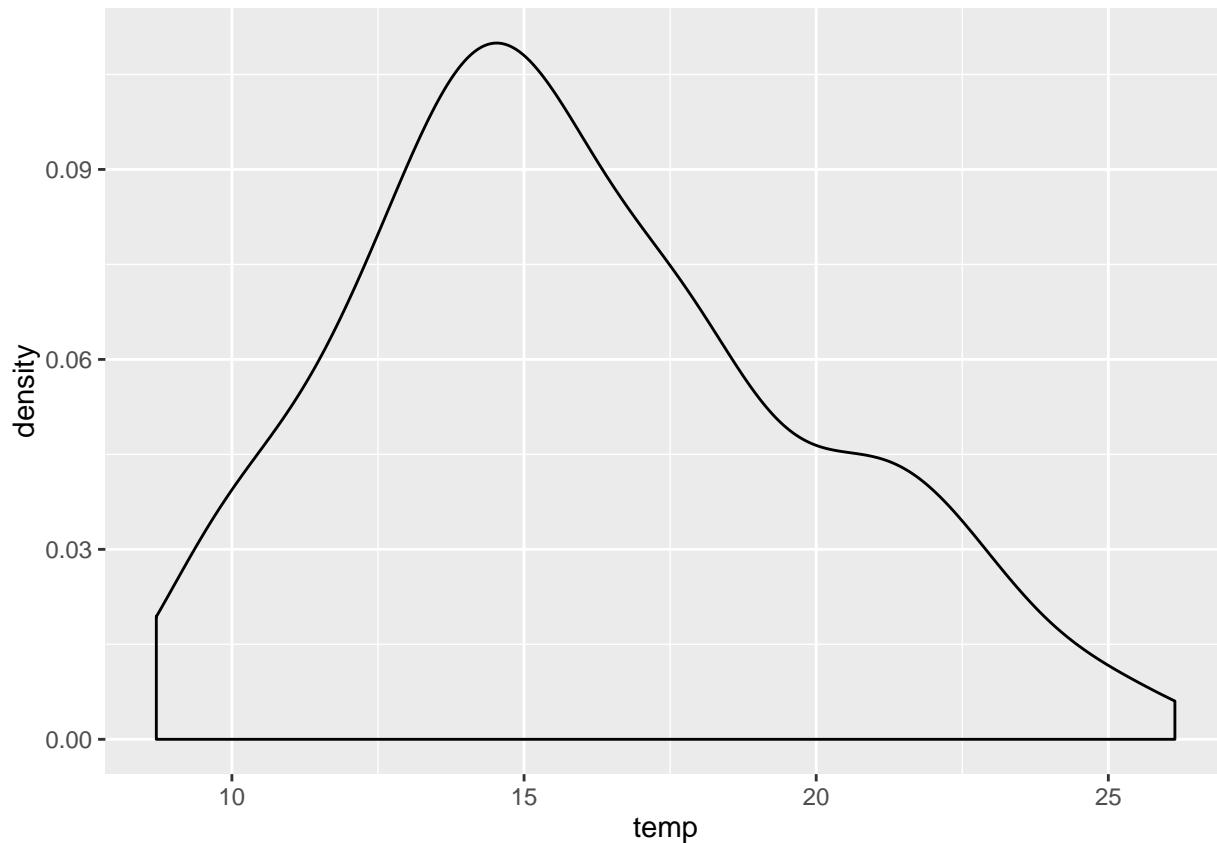


```
punkty$exp_tpz <- exp(punkty$log_tpz)
ggplot(punkty@data, aes(exp_tpz)) + geom_density()
```

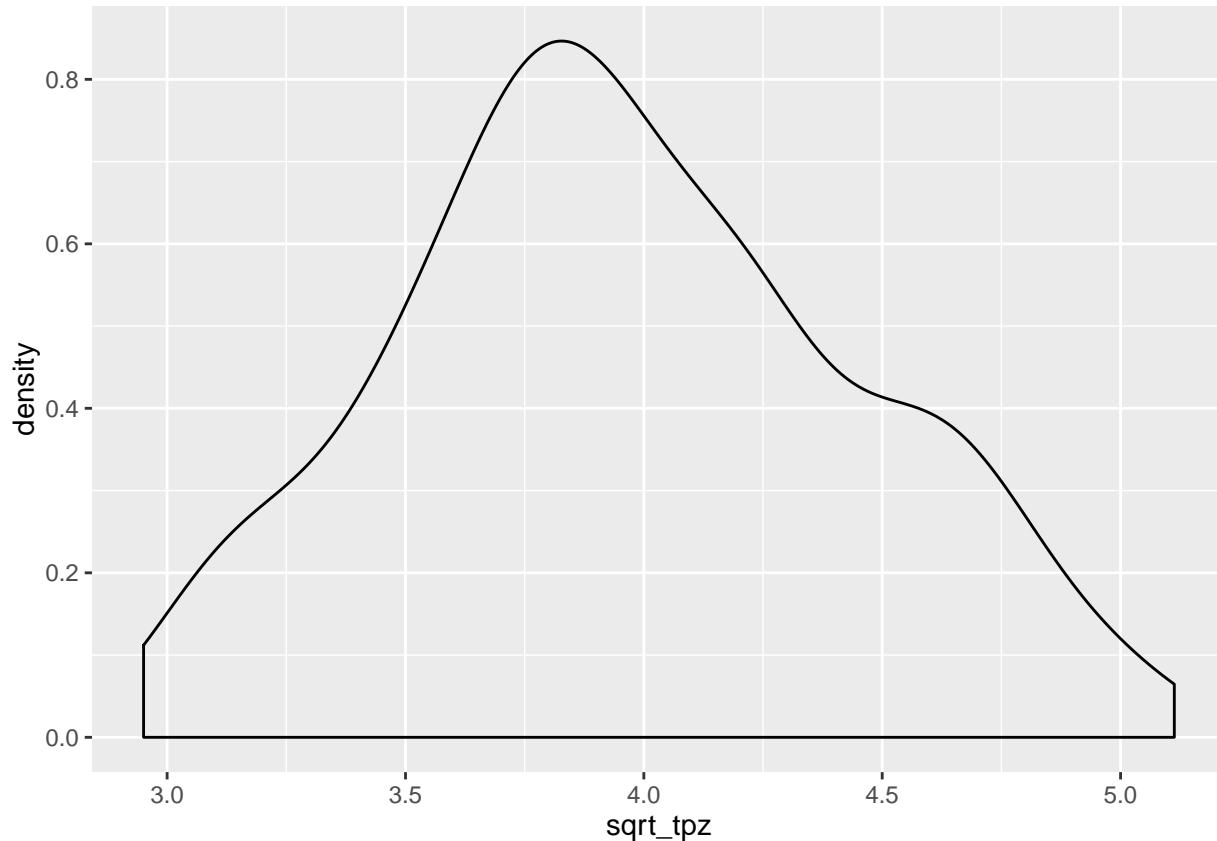


### 3.6.3 Transformacja danych | Pierwiastkowanie

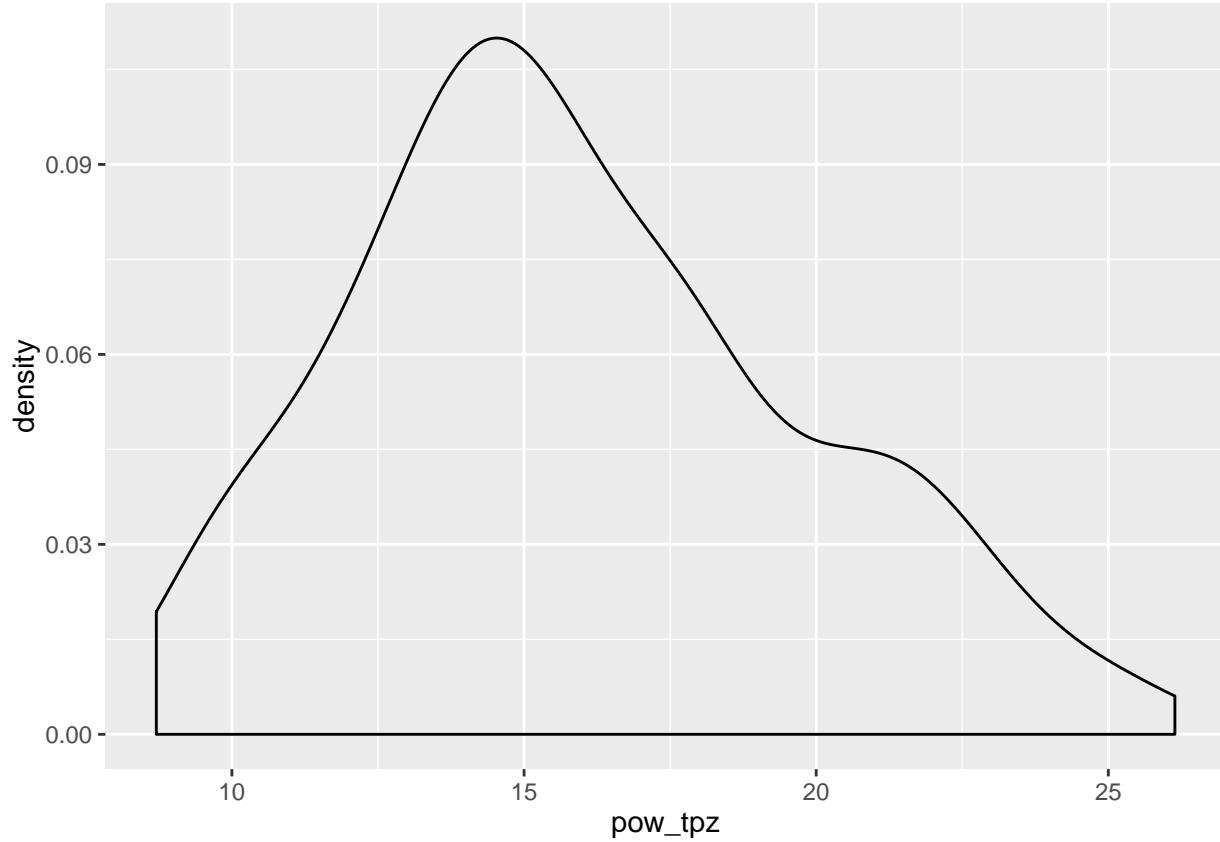
```
ggplot(punkty@data, aes(temp)) + geom_density()
```



```
punkty$sqrt_tpz <- sqrt(punkty$temp)
ggplot(punkty@data, aes(sqrt_tpz)) + geom_density()
```

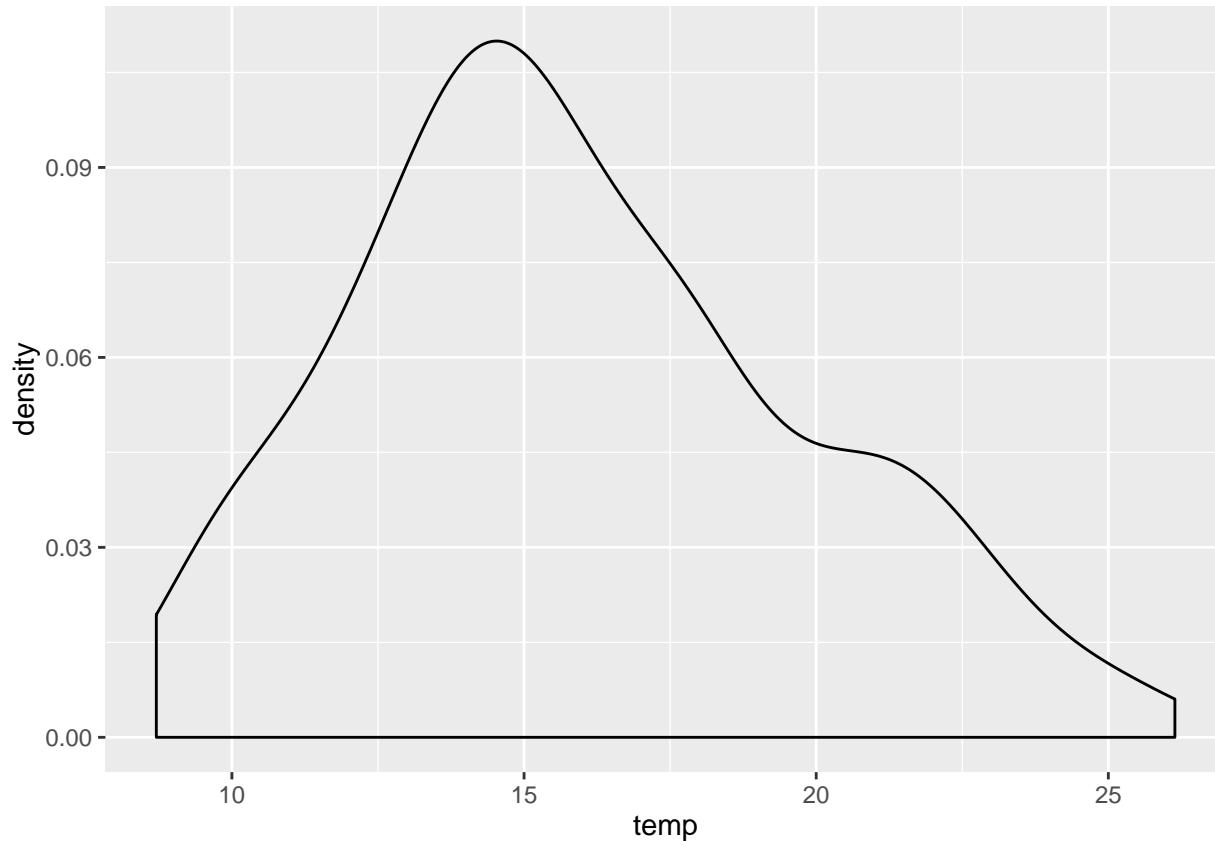


```
punkty$pow_tpz <- punkty$sqrt_tpz^2  
ggplot(punkty@data, aes(pow_tpz)) + geom_density()
```



#### 3.6.4 Transformacja danych | Rodzina transformacji Boxa Coxa

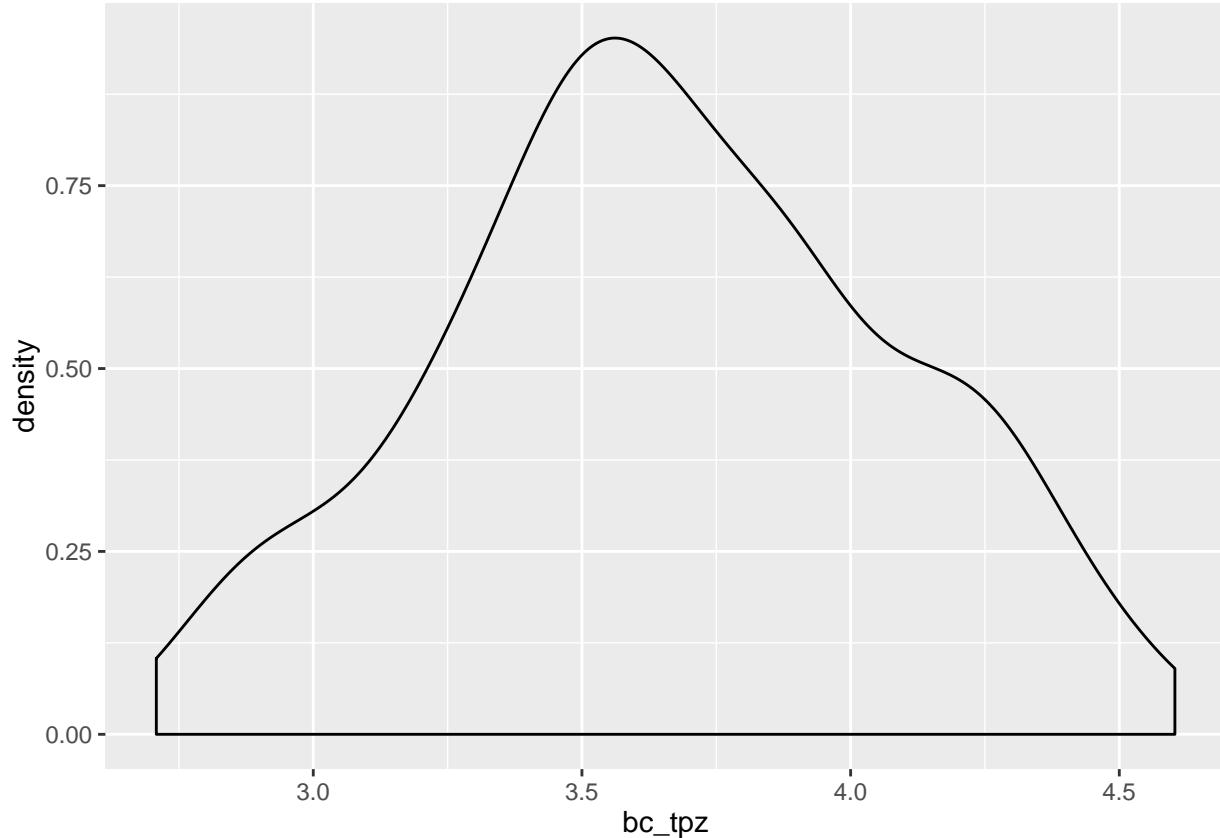
```
ggplot(punkty@data, aes(temp)) + geom_density()
```



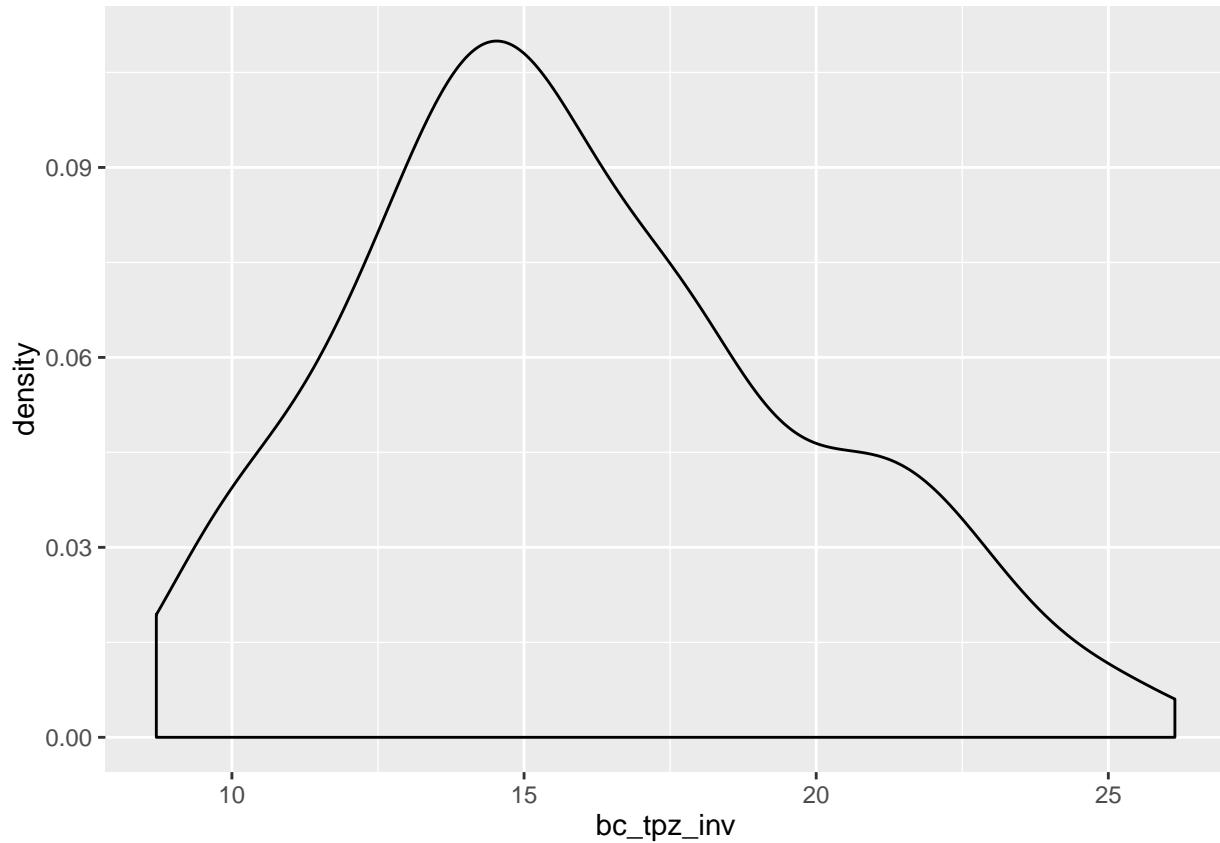
```
transformacja <- BoxCoxTrans(punkty$temp)
transformacja
```

```
## Box-Cox Transformation
##
## 250 data points used to estimate Lambda
##
## Input data summary:
##      Min. 1st Qu. Median      Mean 3rd Qu.      Max.
##     8.706 13.280 15.310 15.950 18.270 26.140
##
## Largest/Smallest: 3
## Sample Skewness: 0.411
##
## Estimated Lambda: 0.2
```

```
punkty$bc_tpz <- predict(transformacja, punkty$temp)
ggplot(punkty@data, aes(bc_tpz)) + geom_density()
```



```
invBoxCox <- function(x, lambda) if (lambda == 0) exp(x) else (lambda*x + 1)^(1/lambda)
punkty$bc_tpz_inv <- invBoxCox(punkty$bc_tpz, lambda=0.2)
ggplot(punkty@data, aes(bc_tpz_inv)) + geom_density()
```



# Chapter 4

## Eksploracyjna analiza danych przestrzennych

```
library('geostatbook')
data(punkty)
data(granica)
```

### 4.1 Mapy

#### 4.1.1 Podstawowe terminy | Kontekst przestrzenny

- Populacja - cały obszar, dla którego chcemy określić wybrane właściwości
- Próba - zbiór obserwacji, dla których mamy informacje. Inaczej, próba to podzbiór populacji. Zazwyczaj niemożliwe (lub bardzo kosztowne) jest zdobycie informacji o całej populacji. Z tego powodu bardzo cenne jest odpowiednie wykorzystanie informacji z próby.

#### 4.1.2 Mapy punktowe | Cel

- Sprawdzenie poprawności współrzędnych
- Wgląd w typ próbkowania
- Sprawdzenie poprawności danych - dane odstające lokalnie
- Identyfikacja głównych cech struktury przestrzennej zjawiska (np. trend)

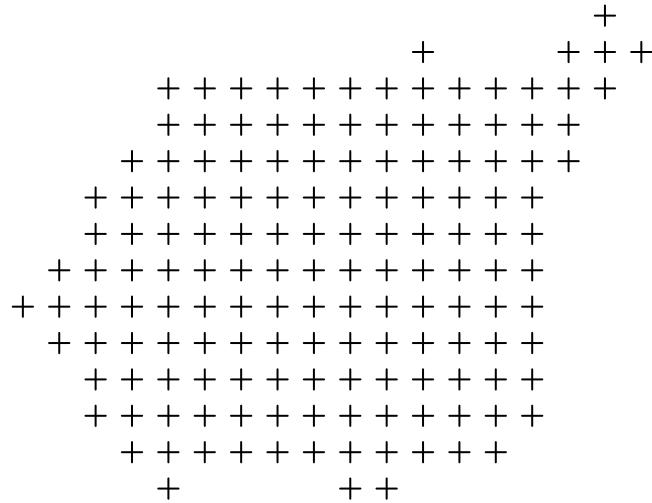
### 4.2 Próbkowanie

#### 4.2.1 Typy próbowania (oferowane przez pakiet **sp**)

- Regularny (ang.*regular*)
- Losowy (ang.*random*)
- Losowy stratyfikowany (ang.*stratified*)
- Preferencyjny (ang.*clustered*)

### 4.2.2 Typ próbowania | Regularny

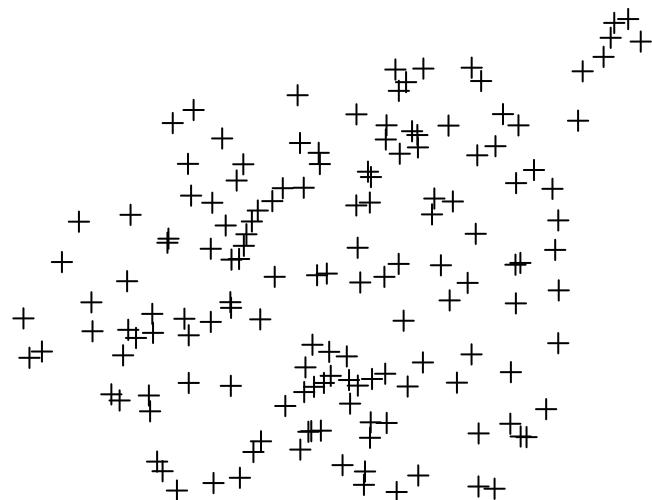
```
set.seed(225)
regularny <- spsample(granica, 150, type = 'regular')
plot(regularny)
```



- Zmienna *offset*

### 4.2.3 Typ próbowania | Losowy

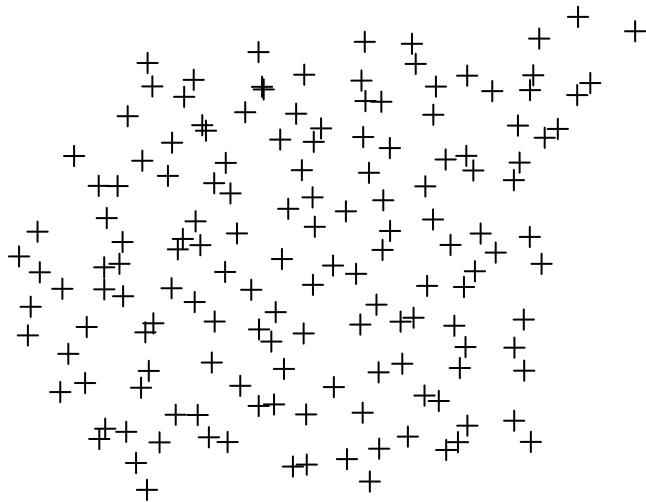
```
set.seed(301)
losowy <- spsample(granica, 150, type = 'random')
plot(losowy)
```



- Każda lokalizacja ma takie samo prawdopodobieństwo wystąpienia
- Każdy punkt jest losowany niezależnie od pozostałych

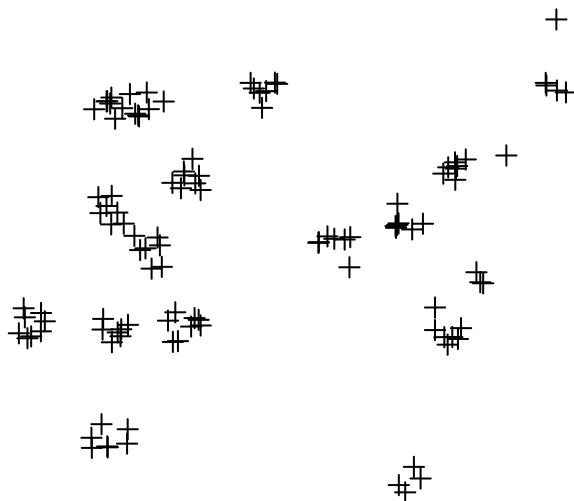
#### 4.2.4 Typ próbowania | Losowy stratyfikowany

```
set.seed(125)
strat <- spsample(granica, 150, type = 'stratified')
plot(strat)
```



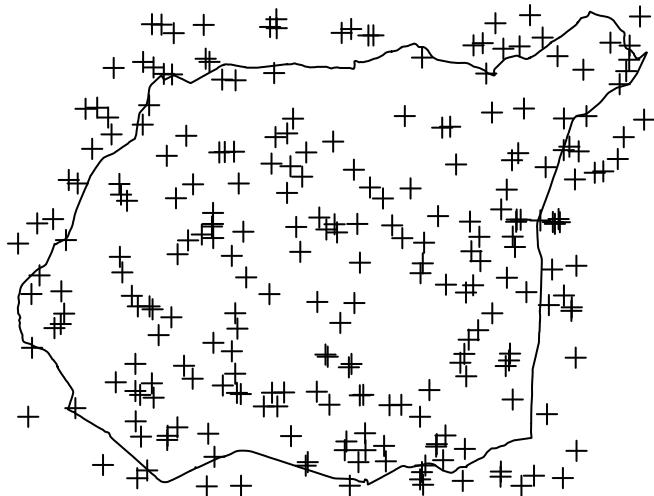
#### 4.2.5 Typ próbowania | Preferencyjny

```
set.seed(425)
pref <- spsample(granica, 150, type = 'clustered', nclusters=30)
plot(pref)
```



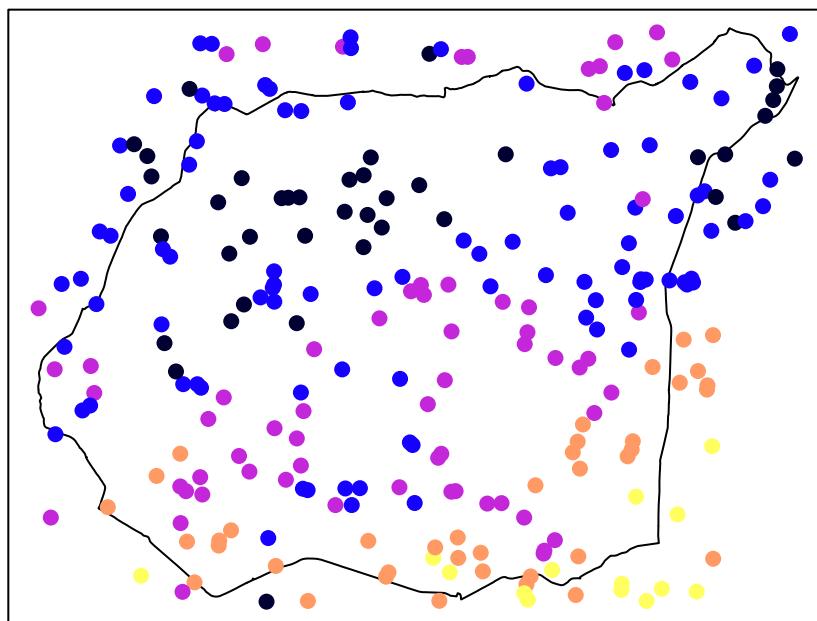
#### 4.2.6 Mapy punktowe i dane lokalnie odstające

```
data(granica)
plot(granica)
plot(punkty, add=TRUE)
```



```
spplot(punkty, 'temp', identify=TRUE)
```

```
spplot(punkty, 'temp', sp.layout = granica)
```



- [8.706, 12.19]
- (12.19, 15.68]
- (15.68, 19.17]
- (19.17, 22.65]
- (22.65, 26.14]

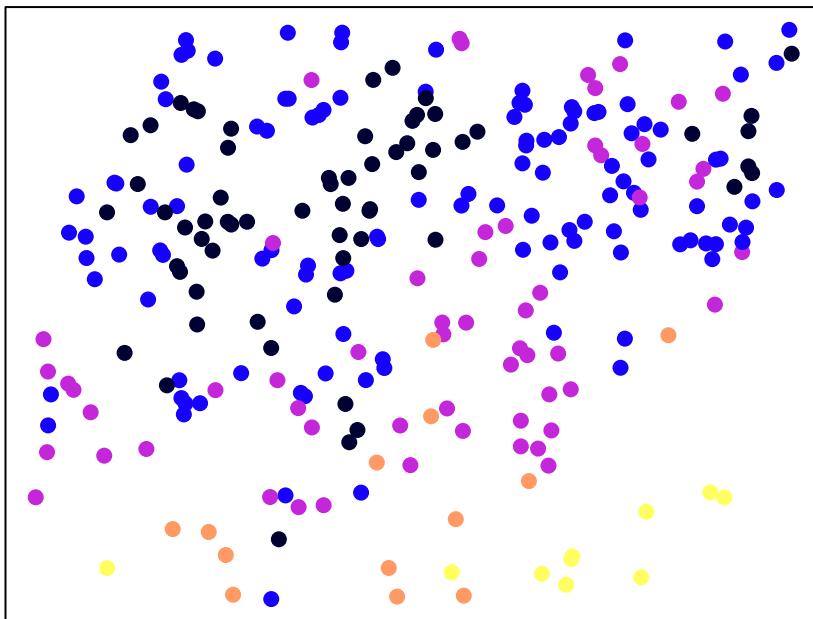
## 4.3 Rozgrupowanie danych

### 4.3.1 Rozgrupowanie danych

- Istnieje szereg metod rozgrupowywania danych, między innymi:
  - Rozgrupowywanie komórkowe
  - Rozgrupowywanie poligonalne
- Celem tych metod jest nadanie wag obserwacjom w celu zapewnienia reprezentatywności przestrzennej danych

### 4.3.2 Rozgrupowanie danych

```
data(punkty_pref)
spplot(punkty_pref, 'temp')
```



- [8.233,11.76]
- (11.76,15.28]
- (15.28,18.8]
- (18.8,22.33]
- (22.33,25.85]

```
summary(punkty_pref$temp)
```

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	8.233	12.070	14.260	14.280	15.540	25.850

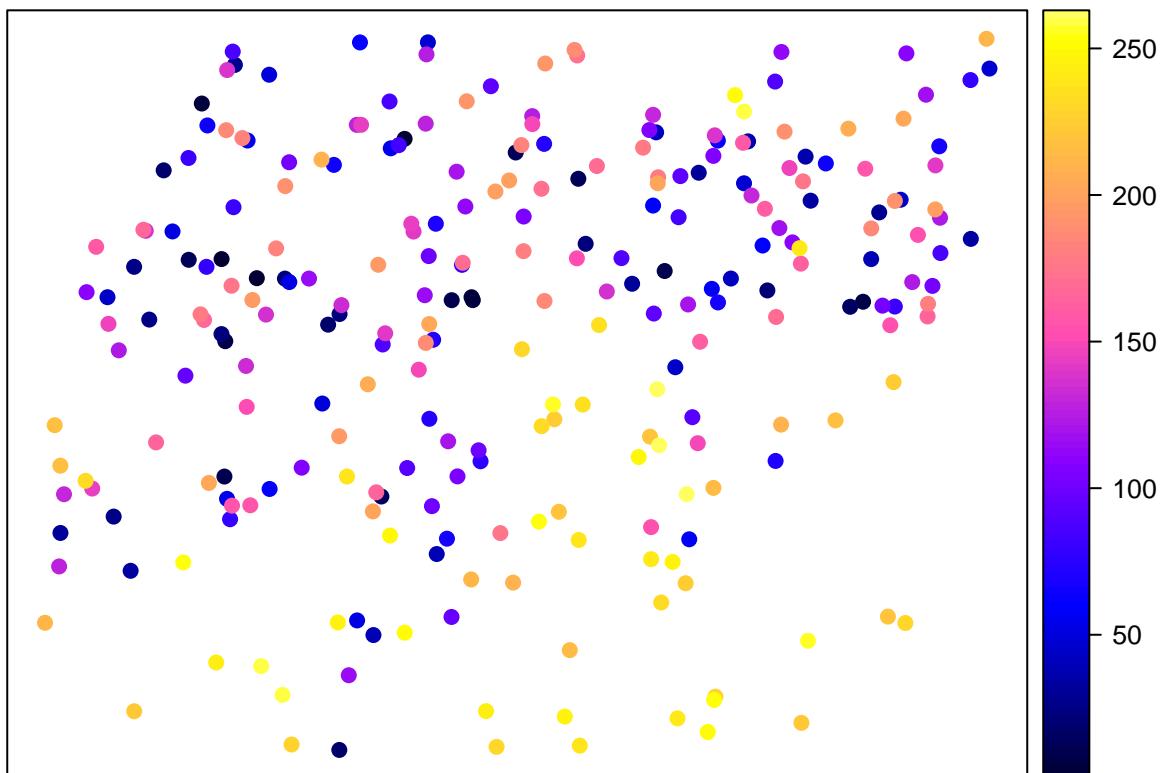
### 4.3.3 Rozgrupowanie komórkowe I | (ang. *cell declustering*)

$$w'_j = \frac{\frac{1}{n_i}}{\text{liczba komórek z danymi}} \cdot n$$

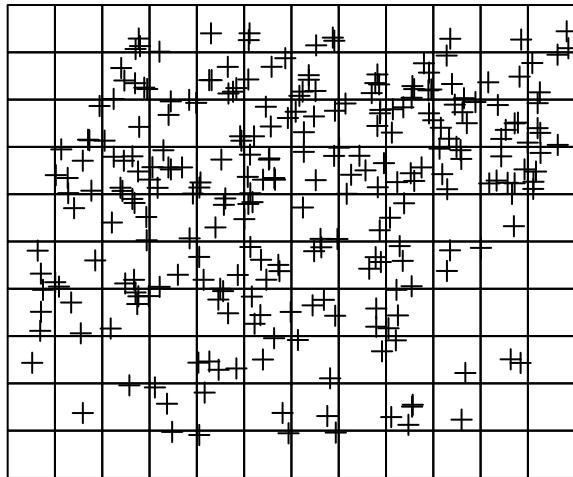
, gdzie  $n_i$  to liczba obserwacji w komórce, a  $n$  to łączna liczba obserwacji

#### 4.3.4 Rozgrupowanie komórkowe I | (ang. *cell declustering*)

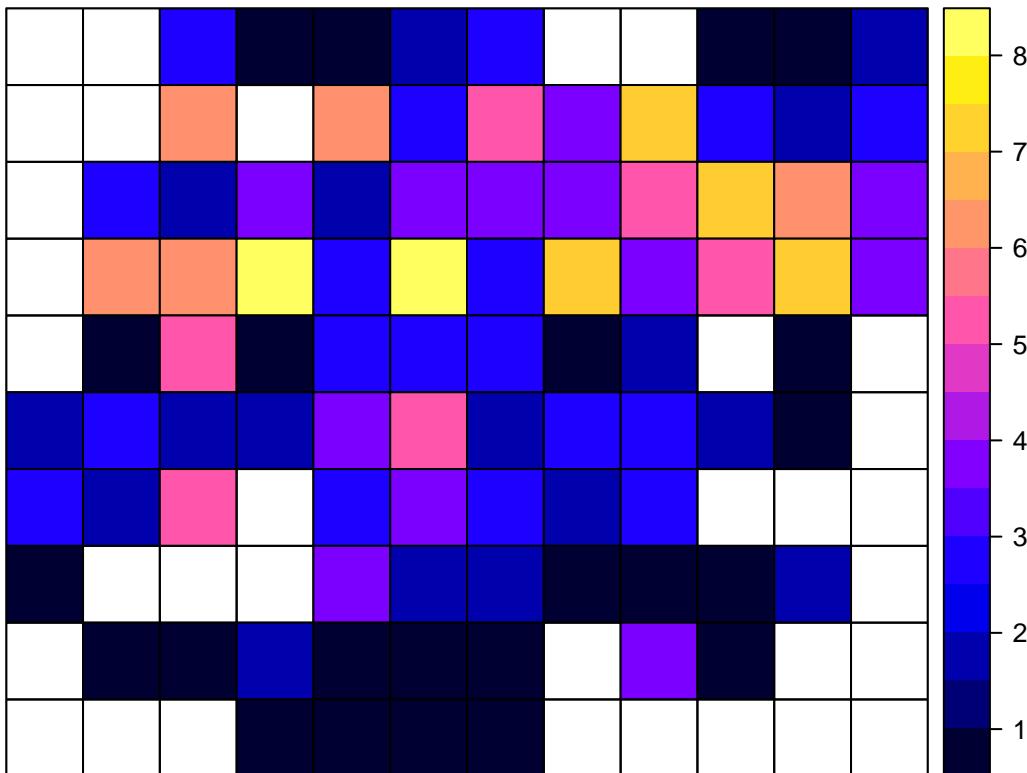
```
data(punkty_pref)
punkty_pref$id <- 1:nrow(punkty_pref)
spplot(punkty_pref, 'id', colorkey=TRUE)
```



```
data(granica)
siatka_n <- raster(extent(gBuffer(granica, width = 500)))
res(siatka_n) <- c(1000, 1000)
siatka_n[] <- 0
proj4string(siatka_n) <- CRS(proj4string(punkty_pref))
siatka_n <- as(siatka_n, 'SpatialPolygonsDataFrame')
siatka_n <- siatka_n[!is.na(siatka_n@data$layer), ]
plot(siatka_n)
plot(punkty_pref, add=TRUE)
```

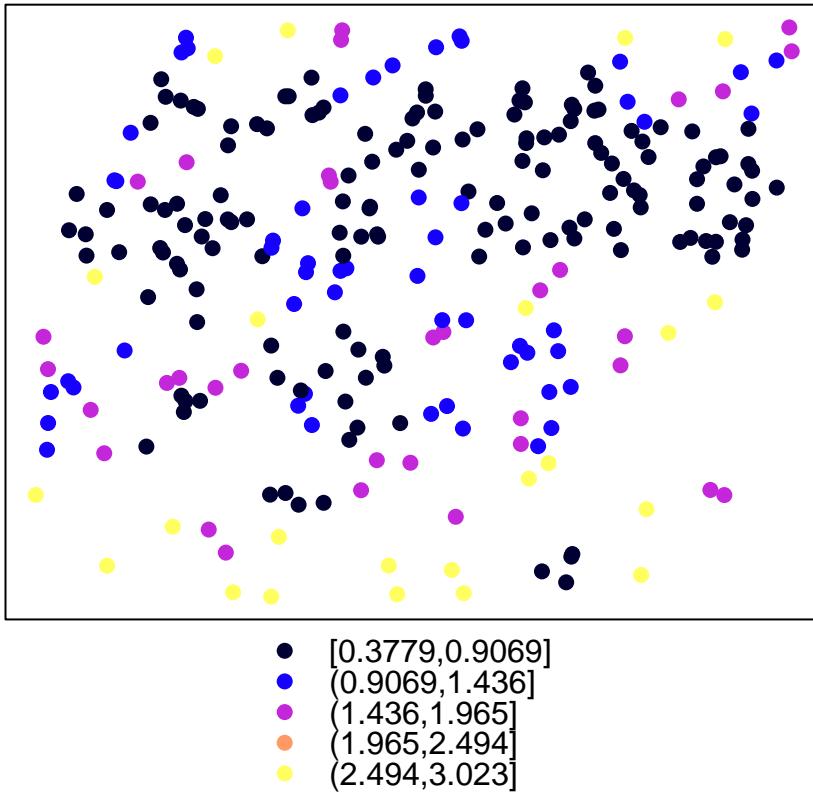


```
punkty_pref$liczebnosc <- rep(0, length(punkty_pref))
siatka_nr <- aggregate(punkty_pref['liczebnosc'], by = siatka_n, FUN = length)
spplot(siatka_nr, 'liczebnosc')
```



```
liczba <- over(punkty_pref, siatka_nr)
punkty_pref$waga <- ((1/liczba$liczebnosc)/sum(!is.na(siatka_nr$liczebnosc))) * length(punkty_pref)

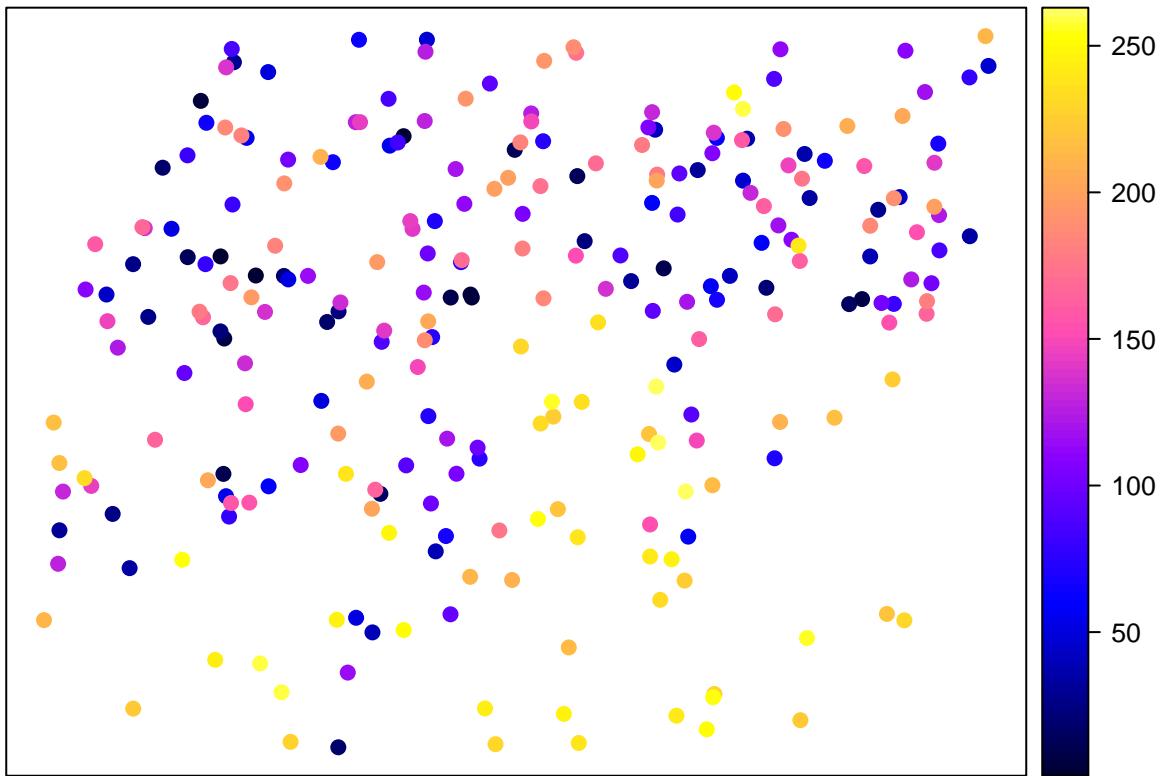
spplot(punkty_pref, 'waga')
```



```
srednia_arytmetyczna <- mean(punkty_pref$temp)
srednia_wazona_c1 <- mean(punkty_pref$temp * punkty_pref$waga, na.rm=TRUE)
```

#### 4.3.5 Rozgrupowanie komórkowe II | (ang. *cell declustering*)

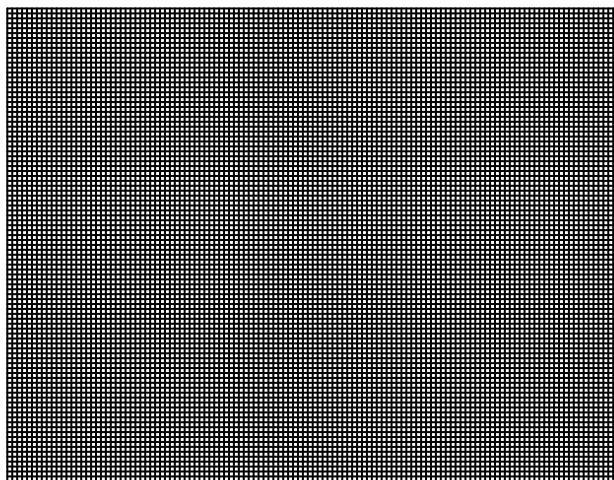
```
data(punkty_pref)
punkty_pref$id <- 1:nrow(punkty_pref)
spplot(punkty_pref, 'id', colorkey=TRUE)
```



```

data(granica)
siatka_n <- raster(extent(gBuffer(granica, width = 500)))
res(siatka_n) <- c(100, 100)
siatka_n[] <- 0
proj4string(siatka_n) <- CRS(proj4string(punkty_pref))
# siatka_n <- mask(siatka_n, granica)
siatka_n <- as(siatka_n, 'SpatialPointsDataFrame')
siatka_n <- siatka_n[!is.na(siatka_n@data$layer), ]
gridded(siatka_n) <- TRUE
plot(siatka_n)

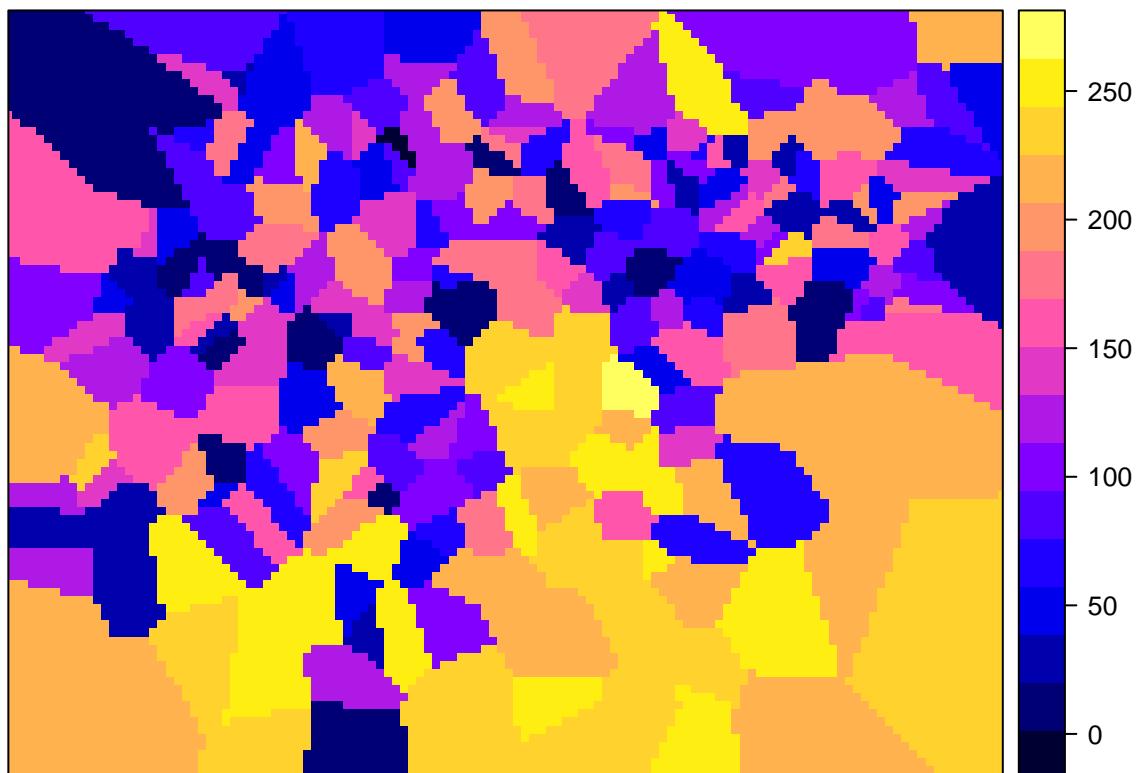
```



```
out <- kriging(id~1, punkty_pref, siatka_n, nmax=1)
```

```
## [inverse distance weighted interpolation]
```

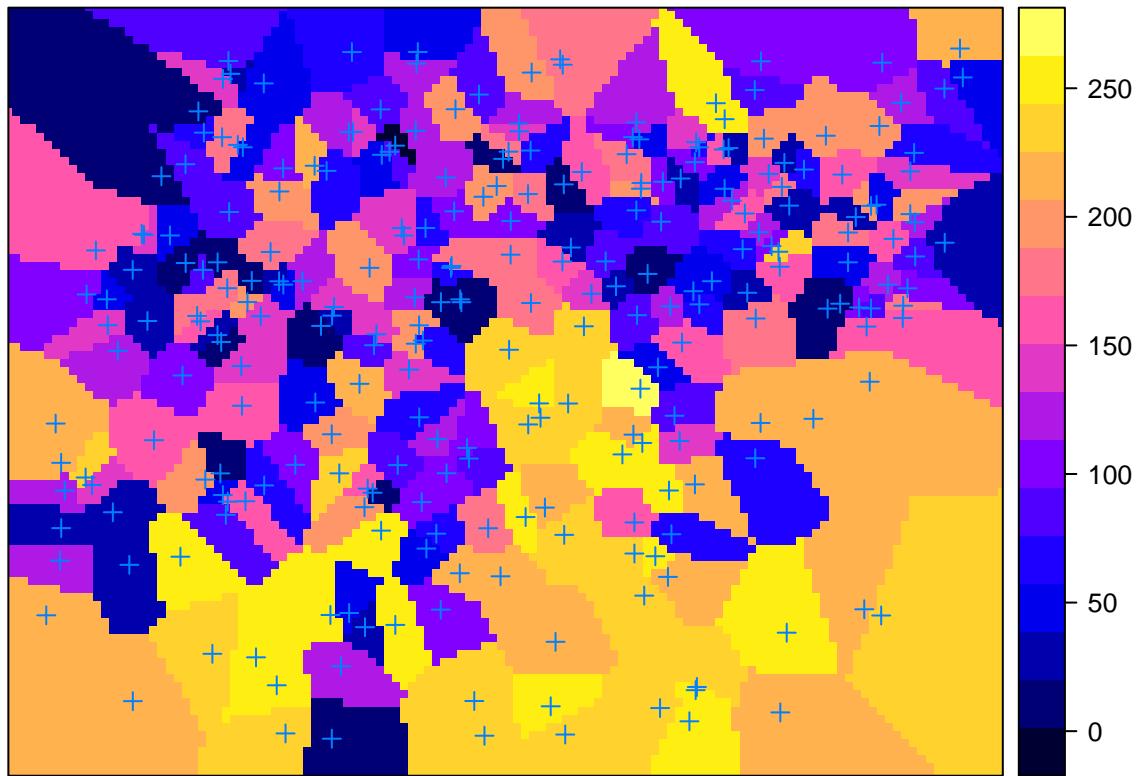
```
spplot(out, 'var1.pred')
```



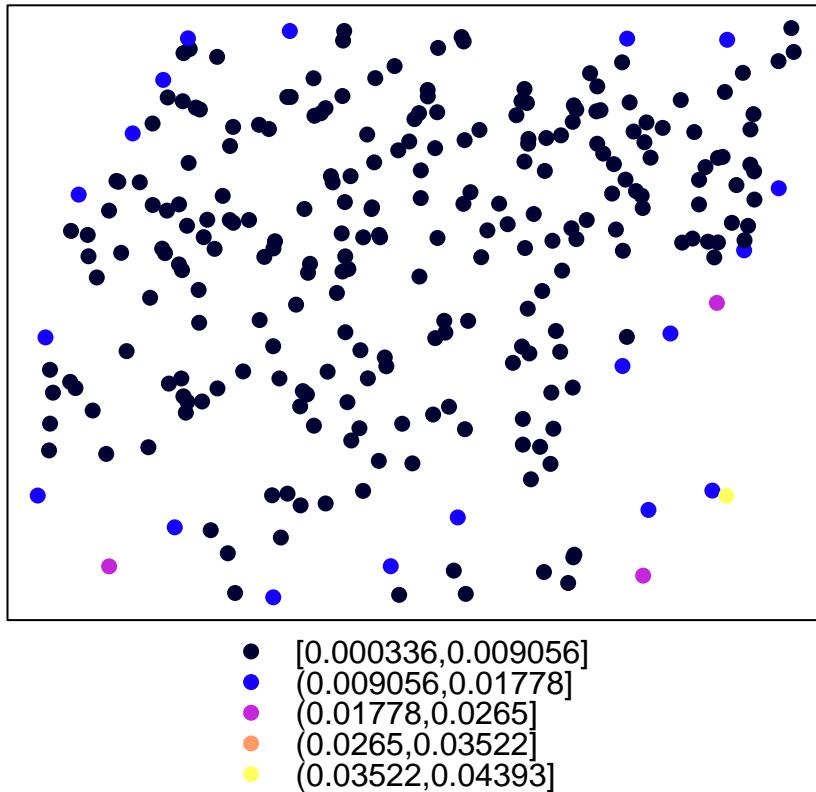
```
df <- as.data.frame(table(out[[1]]))
df$waga <- df$Freq/sum(df$Freq)
punkty_pref <- merge(punkty_pref, df, by.x='id', by.y='Var1')
summary(punkty_pref$waga)
```

```
##      Min. 1st Qu. Median      Mean 3rd Qu.      Max.
## 0.000336 0.001386 0.002520 0.003802 0.004326 0.043930
```

```
spplot(out, 'var1.pred', sp.layout=list('sp.points', punkty_pref))
```



```
spplot(punkty_pref['waga'])
```



```
srednia_arytmetyczna <- mean(punkty_pref$temp)
srednia_wazona_c2 <- sum(punkty_pref$temp * punkty_pref$waga, na.rm=TRUE)
```

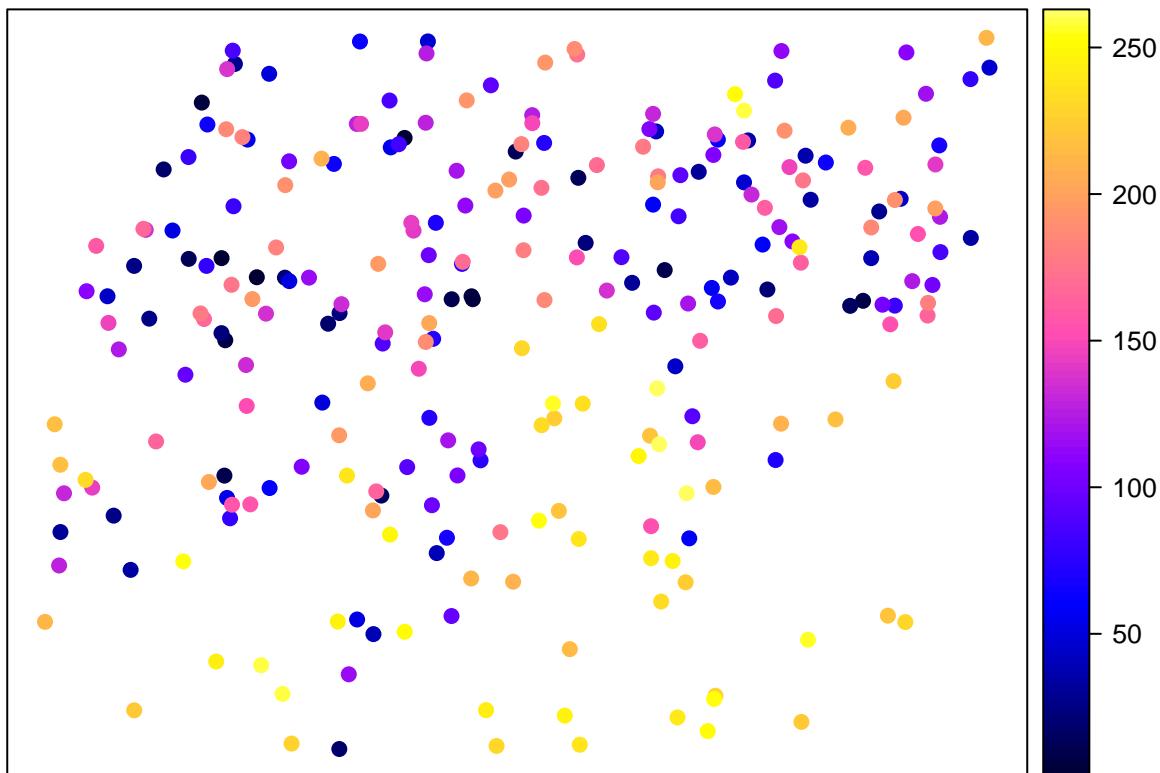
#### 4.3.6 Rozgrupowanie poligonalne | (ang. *polygon declustering*)

$$w'_j = \frac{area_j}{\sum_{j=1}^n area_j} \cdot n$$

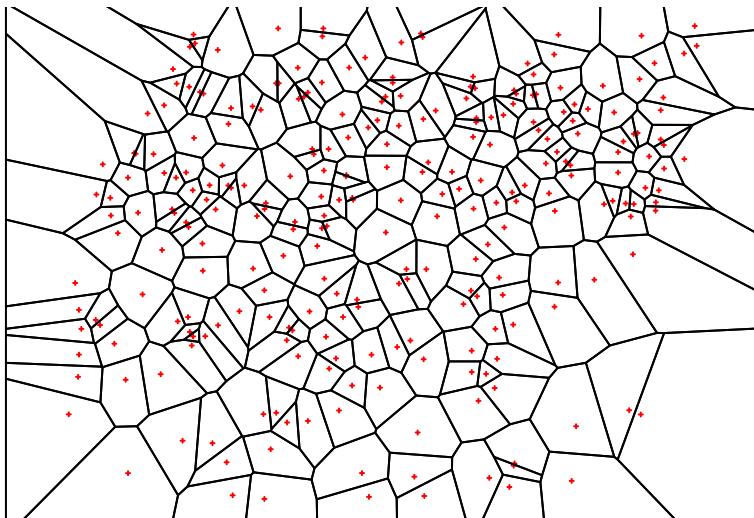
, gdzie  $area_j$  powierzchnia dla wybranej obserwacji, a  $n$  to łączna liczba obserwacji

#### 4.3.7 Rozgrupowanie poligonalne | (ang. *polygon declustering*)

```
data(punkty_pref)
punkty_pref$id <- 1:nrow(punkty_pref)
spplot(punkty_pref, 'id', colorkey=TRUE)
```

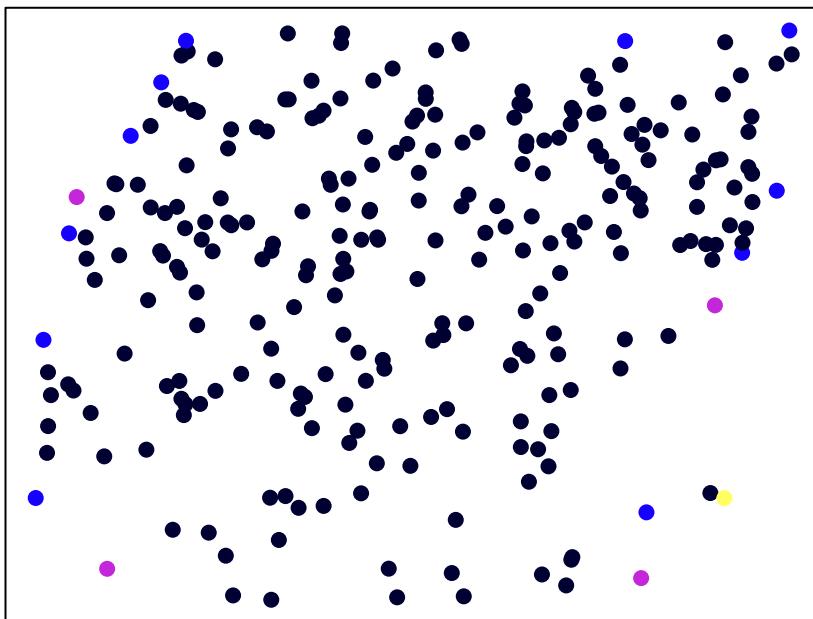


```
v <- voronoi(punkty_pref)
plot(punkty_pref, cex=0.2, col='red')
plot(v, add=TRUE)
```



```
v$pow <- area(v)
v$waga <- v$pow/sum(v$pow) * length(punkty_pref)

punkty_pref <- merge(punkty_pref, v[c('id', 'waga')], by='id')
spplot(punkty_pref, 'waga')
```



- [0.06294,2.992]
- (2.992,5.922]
- (5.922,8.851]
- (8.851,11.78]
- (11.78,14.71]

```
srednia_arytmetyczna <- mean(punkty_pref$temp, na.rm=TRUE)
srednia_wazona_p <- mean(punkty_pref$temp*punkty_pref$waga, na.rm=TRUE)
```

	Średnia arytmetyczna
Populacja	15.59128
Próba	14.28147
Rozgrupowanie komórkowe I	15.37895
Rozgrupowanie komórkowe II	16.07342
Rozgrupowanie poligonalne	16.25110

# Chapter 5

## Metody interpolacji

```
library('geostatbook')
data(punkty)
data(siatka)
data(granica)
```

### 5.1 Modele deterministyczne

#### 5.1.1 Modele deterministyczne

- Parametry tych modeli są zazwyczaj ustalane w oparciu o funkcję odległości lub powierzchni. Brakuje również szacunków na temat oceny błędu modelu. Np:
  - Funkcje wielomianowe (ang. *Polynomials*)
  - Funkcje sklejane (ang. *Splines*)
  - Triangulacje (ang. *Triangulations*)
  - Metoda średniej ważonej odlegością (ang. *Inverse Distance Weighted - IDW*)

#### 5.1.2 Modele deterministyczne

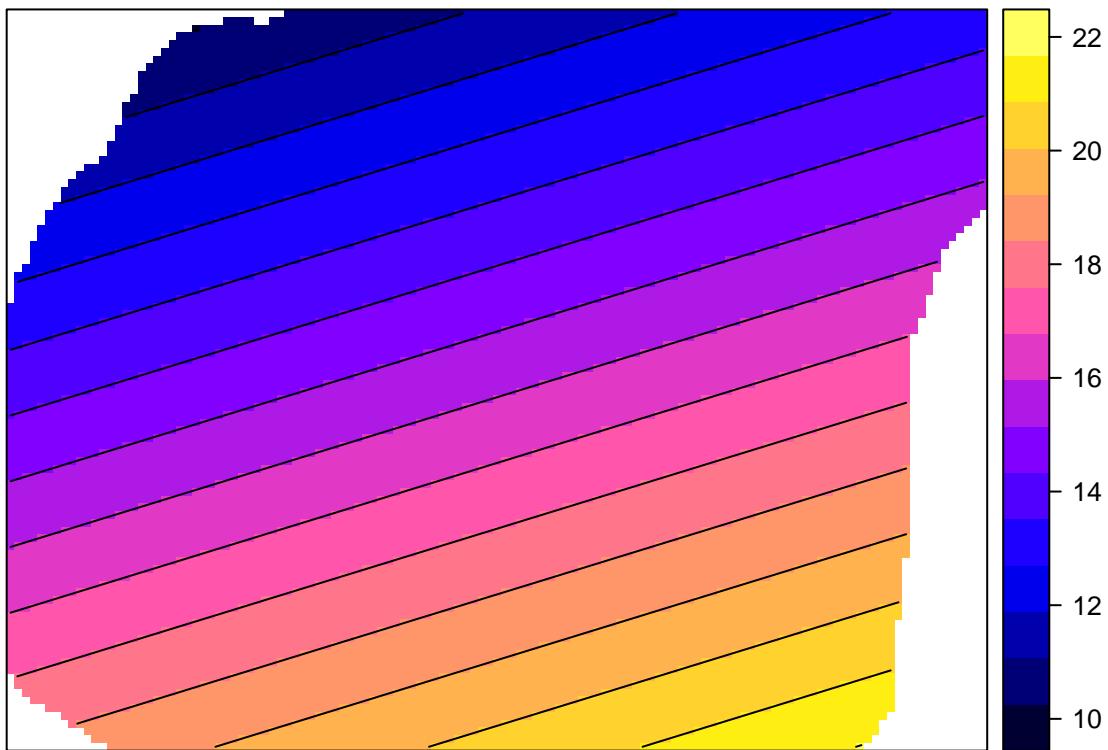
#### 5.1.3 Modele deterministyczne | Funkcje wielomianowe

```
wielomian_1 <- gstat(formula=temp~1, data=punkty, degree=1)
wielomian_1_pred <- predict(wielomian_1, newdata=siatka)
```

```
## [ordinary or weighted least squares prediction]
```

```
spplot(wielomian_1_pred[1], contour=TRUE, main='Powierzchnia trendu - wielomian pierwszego stopnia')
```

## Powierzchnia trendu – wielomian pierwszego stopnia

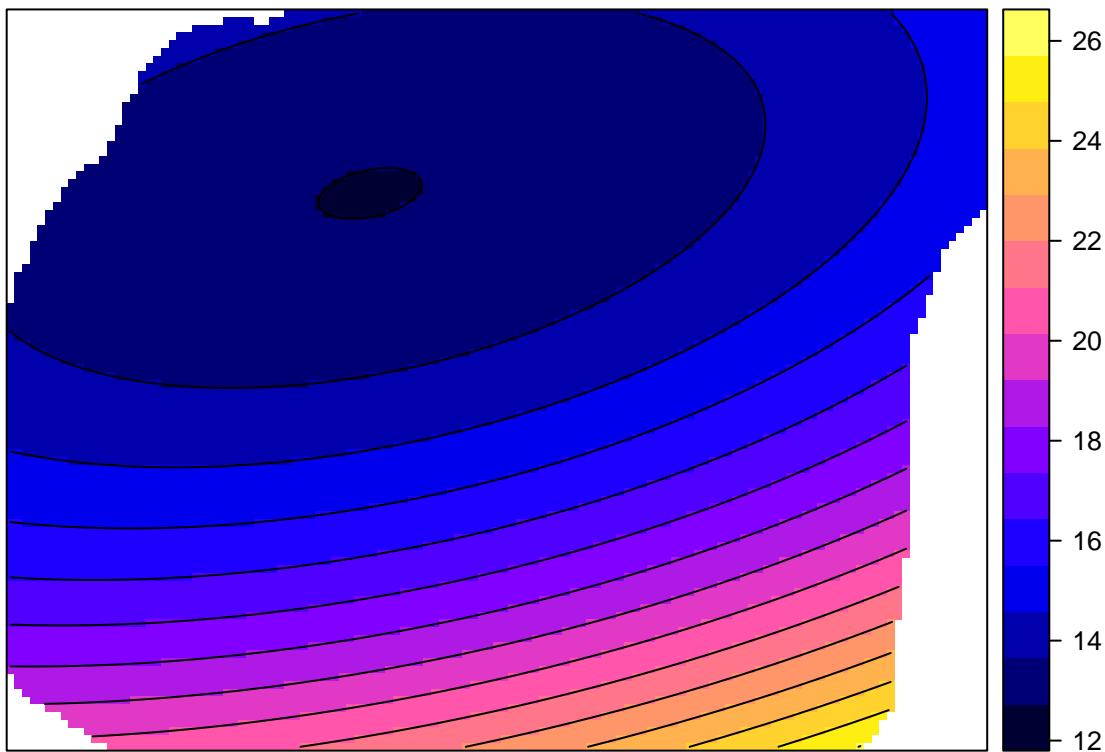


```
wielomian_2 <- gstat(formula=temp~1, data=punkty, degree=2)
wielomian_2_pred <- predict(wielomian_2, newdata=siatka)
```

```
## [ordinary or weighted least squares prediction]
```

```
spplot(wielomian_2_pred[1], contour=TRUE, main='Powierzchnia trendu - wielomian drugiego stopnia')
```

### Powierzchnia trendu – wielomian drugiego stopnia

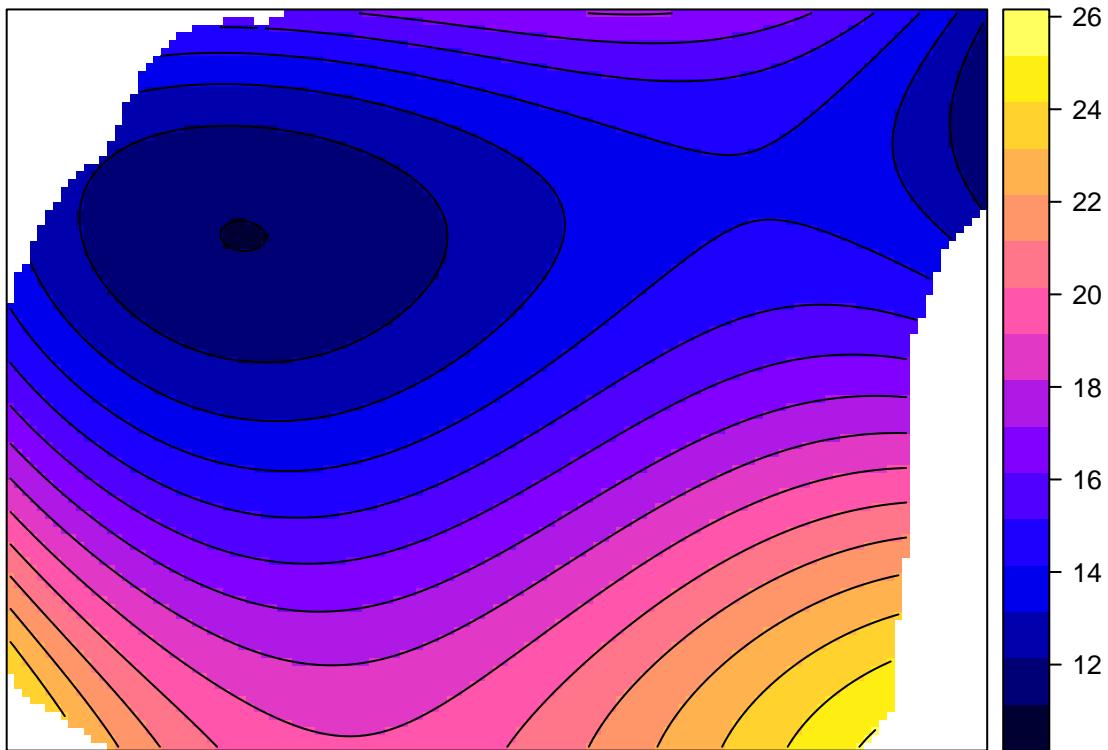


```
wielomian_3 <- gstat(formula=temp~1, data=punkty, degree=3)
wielomian_3_pred <- predict(wielomian_3, newdata=siatka)
```

```
## [ordinary or weighted least squares prediction]
```

```
spplot(wielomian_3_pred[1], contour=TRUE, main='Powierzchnia trendu - wielomian trzeciego stopnia')
```

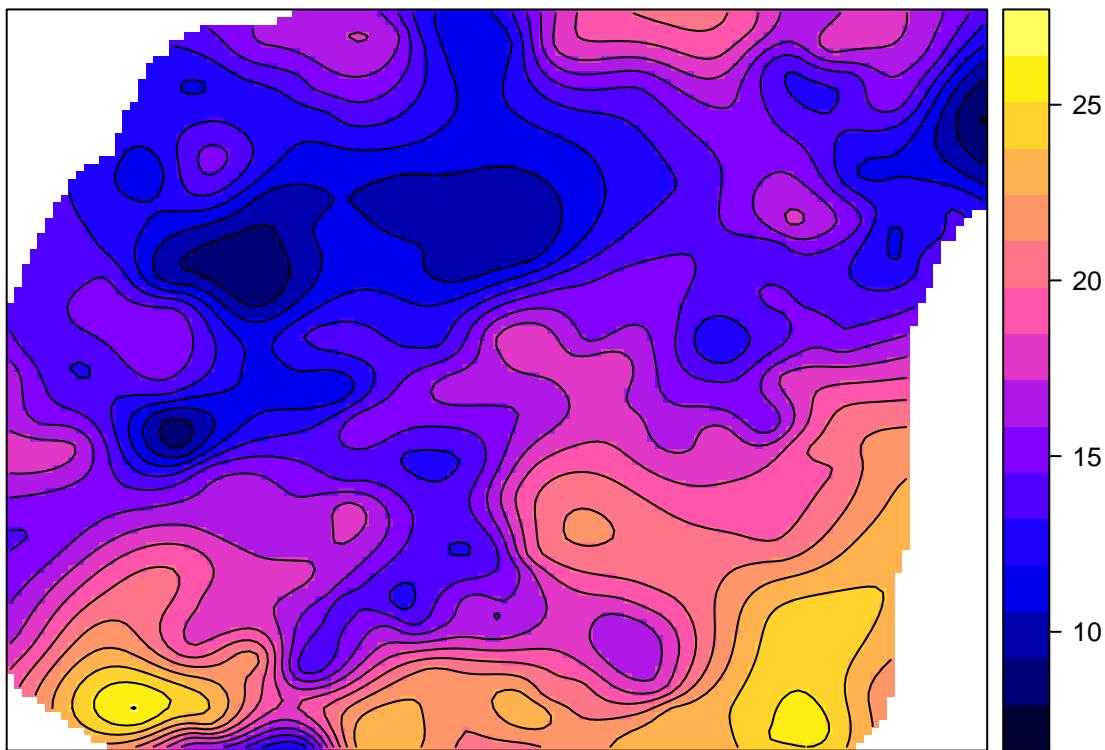
## Powierzchnia trendu – wielomian trzeciego stopnia



### 5.1.4 Modele deterministyczne | Funkcje sklejane

```
tps <- Tps(coordinates(punkty), punkty@data$temp)
ras <- raster(siatka)
spline <- interpolate(ras, tps)
spline <- mask(spline, ras)
spplot(spline, contour=TRUE, main='Funkcje sklejane')
```

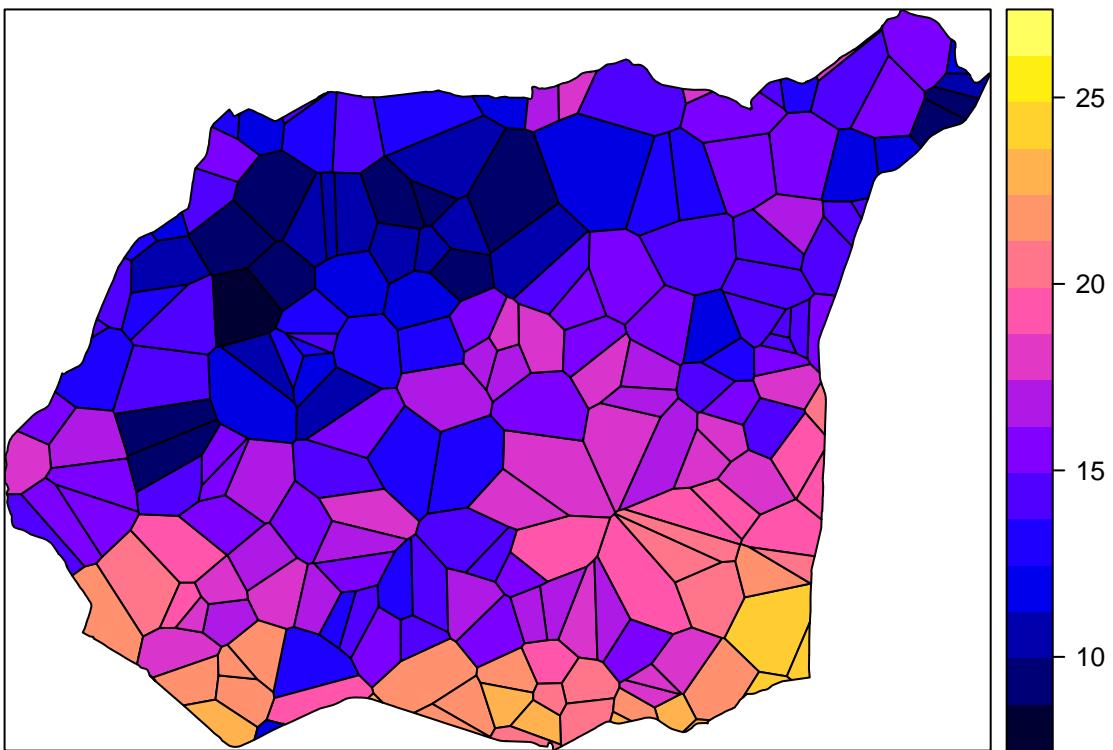
## Funkcje sklejane



### 5.1.5 Modele deterministyczne | Triangulacje (Voronoi)

```
voronoi_interp <- voronoi(punkty)
voronoi_interp <- intersect(granica, voronoi_interp)
spplot(voronoi_interp, 'temp', contour=TRUE, main='Poligony Voronoi')
```

## Poligony Voronoia

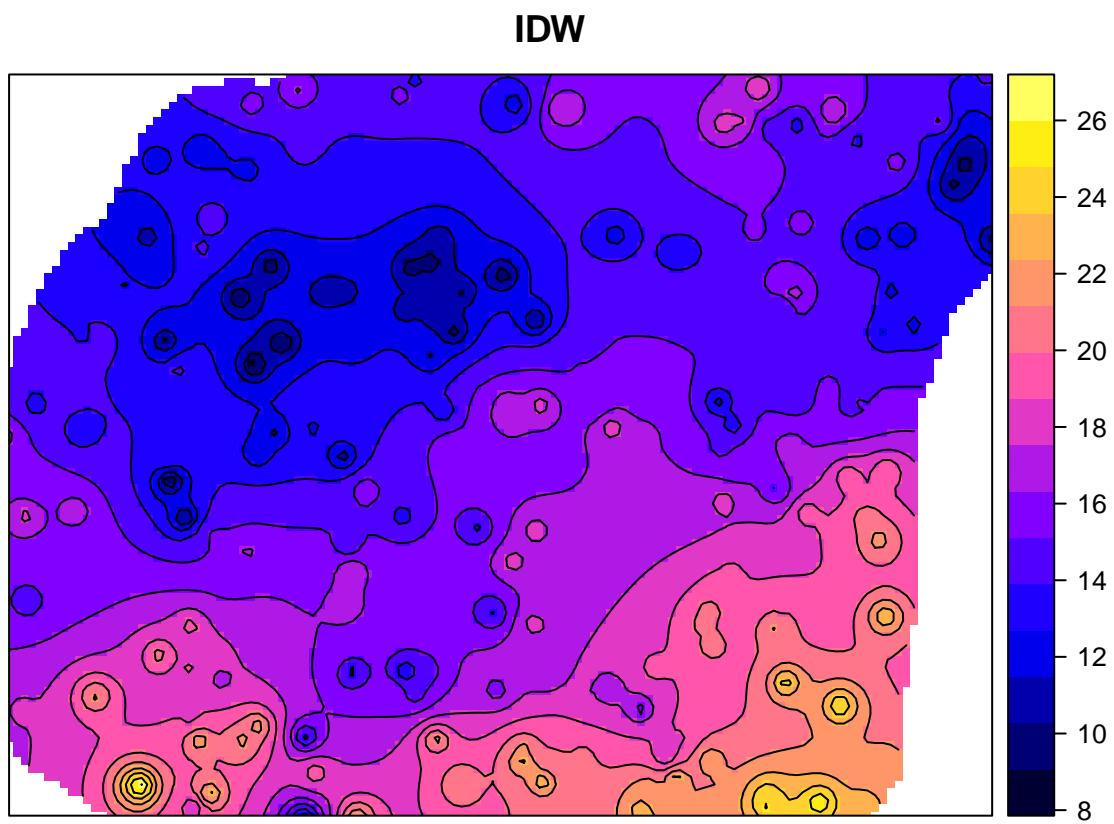


### 5.1.6 Modele deterministyczne | IDW

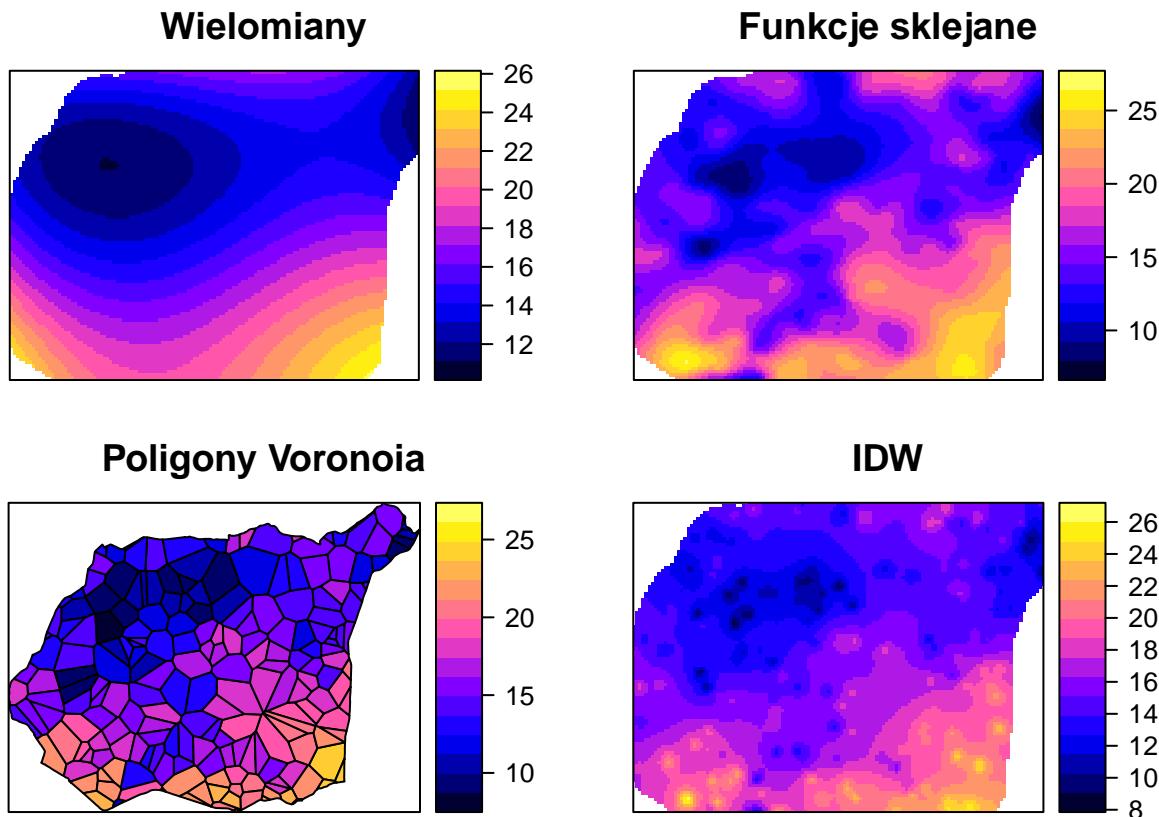
```
idw_wolin <- idw(temp~1, punkty, siatka, idp=2)
```

```
## [inverse distance weighted interpolation]
```

```
spplot(idw_wolin, 'var1.pred', contour=TRUE, main='IDW')
```



### 5.1.7 Modele deterministyczne | Porównanie



## 5.2 Modele statystyczne

- Parametry modeli są określane w oparciu o teorię prawdopodobieństwa. Dodatkowo wynik estymacji zawiera oszacowanie błędu. Np.:
  - Kriging
  - Modele regresyjne
  - Modele bayesowe
  - Modele hybrydowe

# Chapter 6

## Geostatystyka - prolog

```
library('geostatbook')
data(punkty)
data(siatka)
data(granica)
```

### 6.1 Geostatystyka

#### 6.1.1 Geostatystyka

- Zbiór narzędzi statystycznych uwzględniających w analizie danych ich przestrzenną i czasową lokalizację, a opartych o teorię funkcji losowych

#### 6.1.2 Geostatystyka | Funkcje

- Identyfikacja i modelowanie struktury przestrzennej/czasowej zjawiska
- Estymacja - szacowanie wartości badanej zmiennej w nieoprisowanym miejscu i/lub momencie czasu
- Symulacja - generowanie alternatywnych obrazów, które honorują wyniki pomiarów i strukturę przestrzenną/czasową zjawiska
- Optymalizacja próbkowania/sieci pomiarowej

#### 6.1.3 Geostatystyka | Dane wejściowe

1. Wystarczająca duża liczba punktów (minimalnie  $>30$ , ale zazwyczaj więcej niż 100/150)
2. Są reprezentatywne
3. Są niezależne
4. Były stworzone używając stałej metodyki
5. Są wystarczająco dokładne

#### 6.1.4 Geostatystyczna analiza danych

#### 6.1.5 Geostatystyka | Podstawowe etapy

1. Zaprojektowanie sposobu (typu) próbkowania oraz organizacji zadań

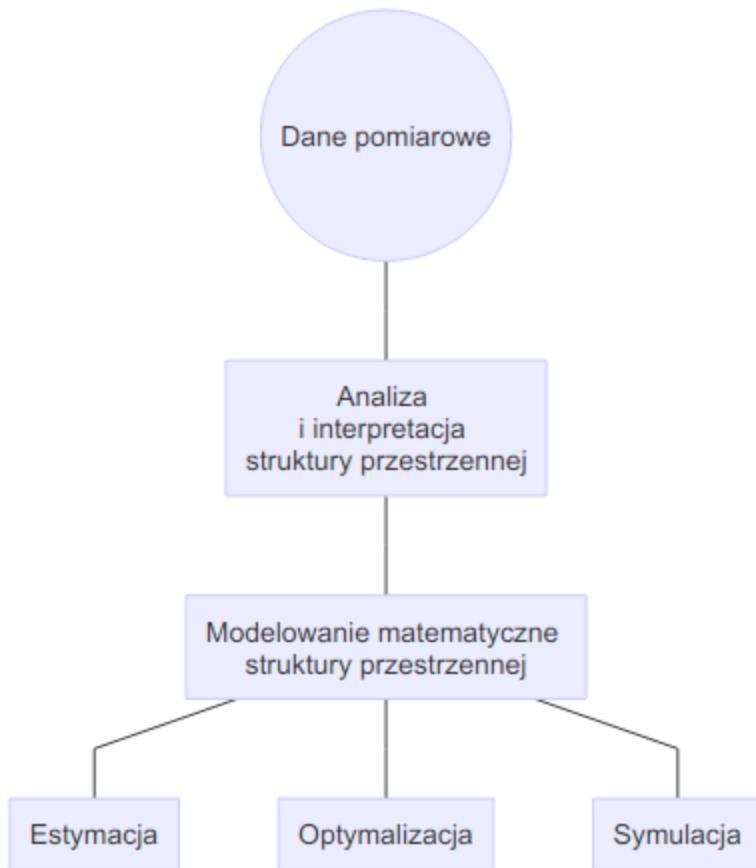


Figure 6.1:

2. Zebranie danych, analiza laboratoryjna
3. Wstępna eksploracja danych, ocena ich jakości
4. Modelowanie semivariogramów na podstawie dostępnych danych
5. Estymacja badanej cechy
6. Porównanie i ocena modeli
7. Stworzenie wynikowego produktu i jego dystrybucja

## 6.2 Przestrzenna kowariancja, korelacja i semiwariancja

### 6.2.1 Przestrzenna kowariancja, korelacja i semiwariancja | Założenia

1. Przestrzennej ciągłości - przestrzenna korelacja między zmienną w dwóch lokalizacjach zależy tylko od ich odległości (oraz czasem kierunku), lecz nie od tego gdzie są one położone
2. Stacjonarności - średnia i wariancja są stałe na całym badanym obszarze

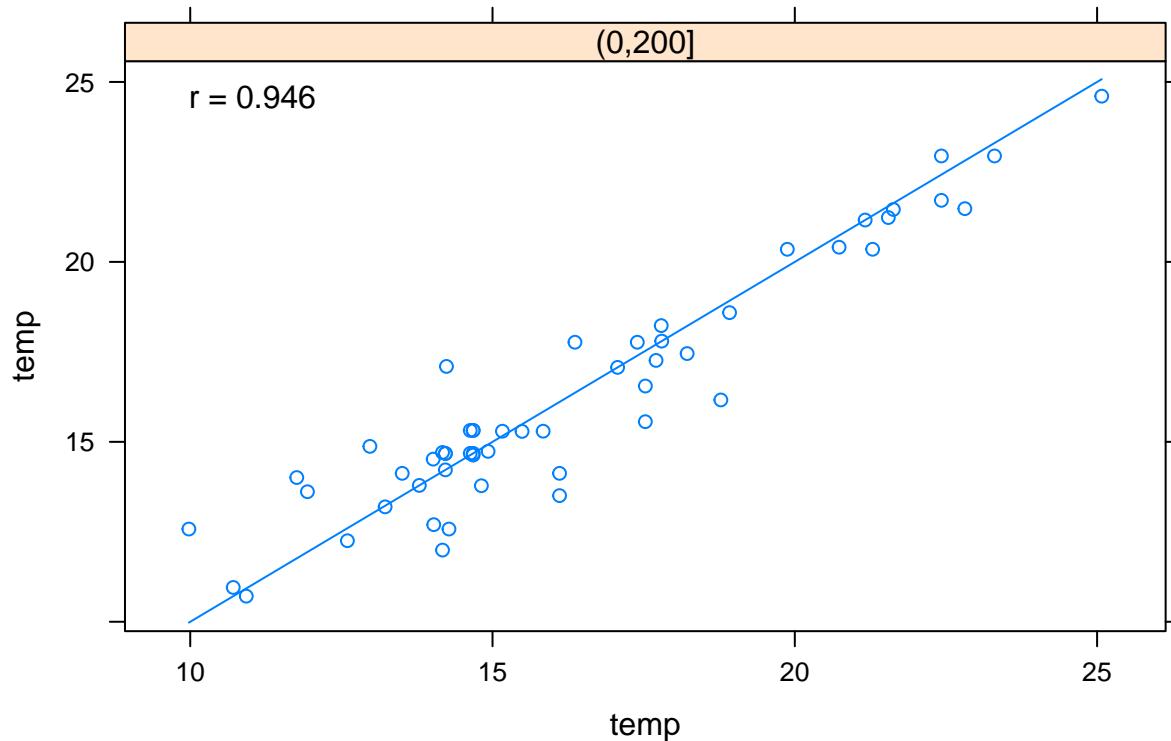
### 6.2.2 Przestrzenna kowariancja, korelacja i semiwariancja

- $u$  - wektor współrzędnych
- $z(u)$  - badana zmienna jako funkcja położenia - inaczej określany jako ogon (ang. *tail*)
- $h$  - lag - odstęp pomiędzy dwoma lokalizacjami
- $z(u + h)$  - wartość badanej zmiennej odległej o odstęp  $h$  - inaczej określany jako głowa (ang. *head*)

### 6.2.3 Przestrzenna kowariancja, korelacja i semiwariancja

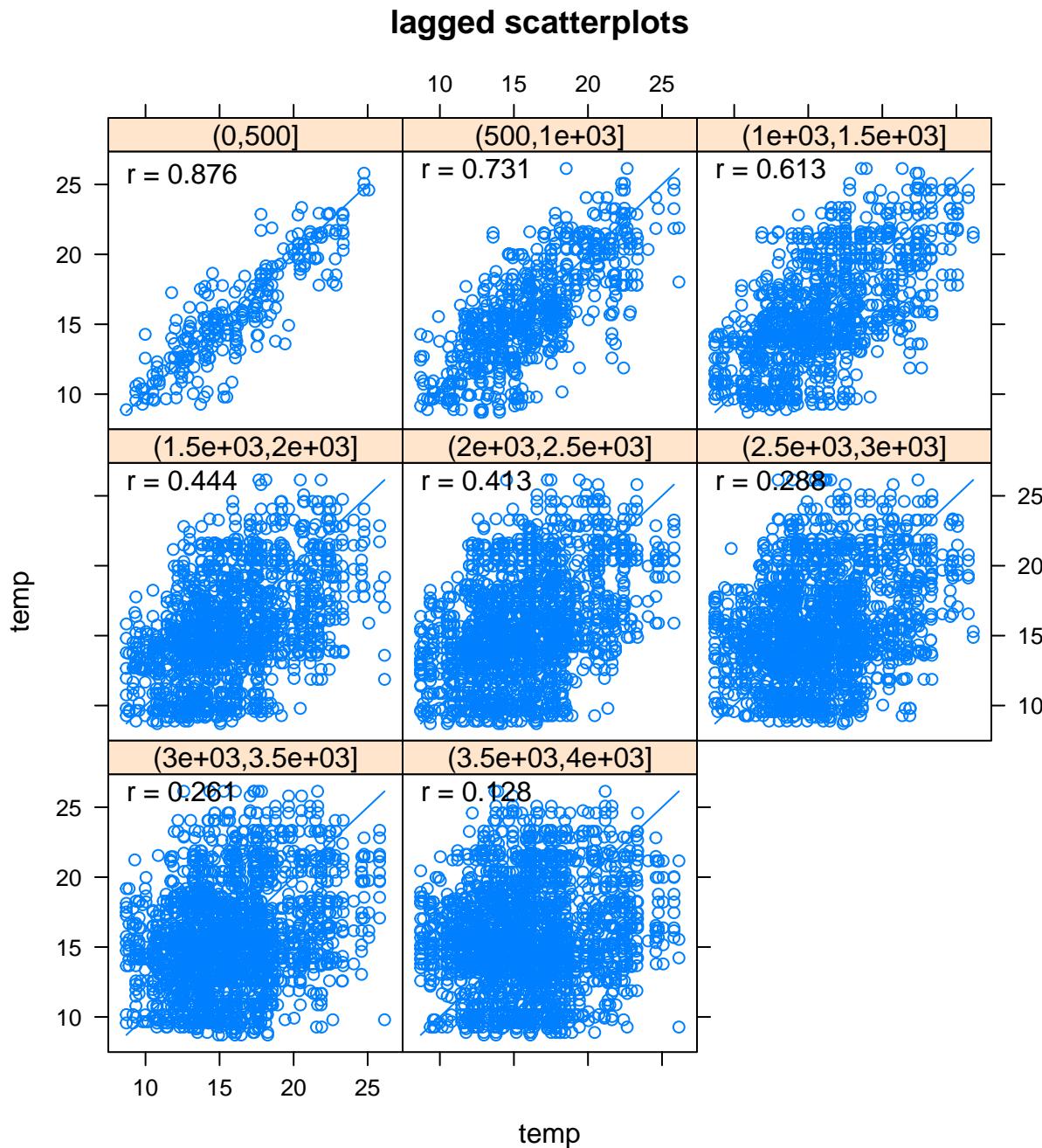
- Kowariancja i korelacja to miary podobieństwa pomiędzy dwoma zmiennymi
- Przenosząc to na aspekt przestrzenny, badamy jedną zmienną ale pomiędzy dwoma punktami odległymi od siebie o pewien dystans (określany jako lag)
- W efekcie otrzymujemy miarę podobieństwa pomiędzy wartością głową i ogona
- Trzecią miarę charakteryzującą relację między obserwacjami odległymi o kolejne odstępy jest semiwariancja
- Z praktycznego punktu widzenia, semivariogram jest preferowaną miarą relacji przestrzennej, ponieważ wykazuje tendencję do lepszego wygładzania danych niż funkcja kowariancji
- Dodatkowo, semivariogram jest mniej wymagający obliczeniowo
- Jednocześnie, dla potrzeb interpretacji relacji kowariancja i korelacja przestrzenna nadaje się nie gorzej niż semiwariancja

## lagged scatterplots



### 6.2.4 Wykres rozrzutu z przesunięciem

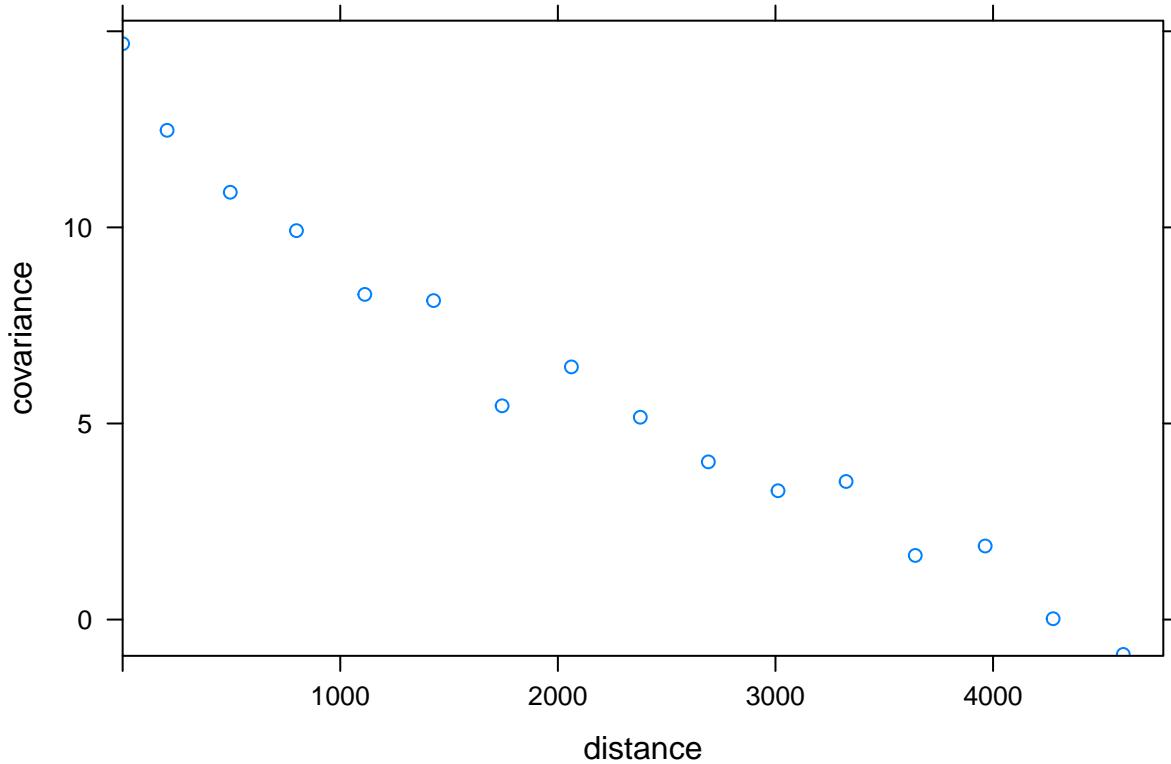
```
hscat(temp~1, punkty, breaks=seq(0, 4000, by=500))
```



### 6.2.5 Autokowariancja

- Autokowariancja pokazuje jak mocno są ze sobą powiązane przestrzennie wartości pary obserwacji odległych od siebie o kolejne przedziały

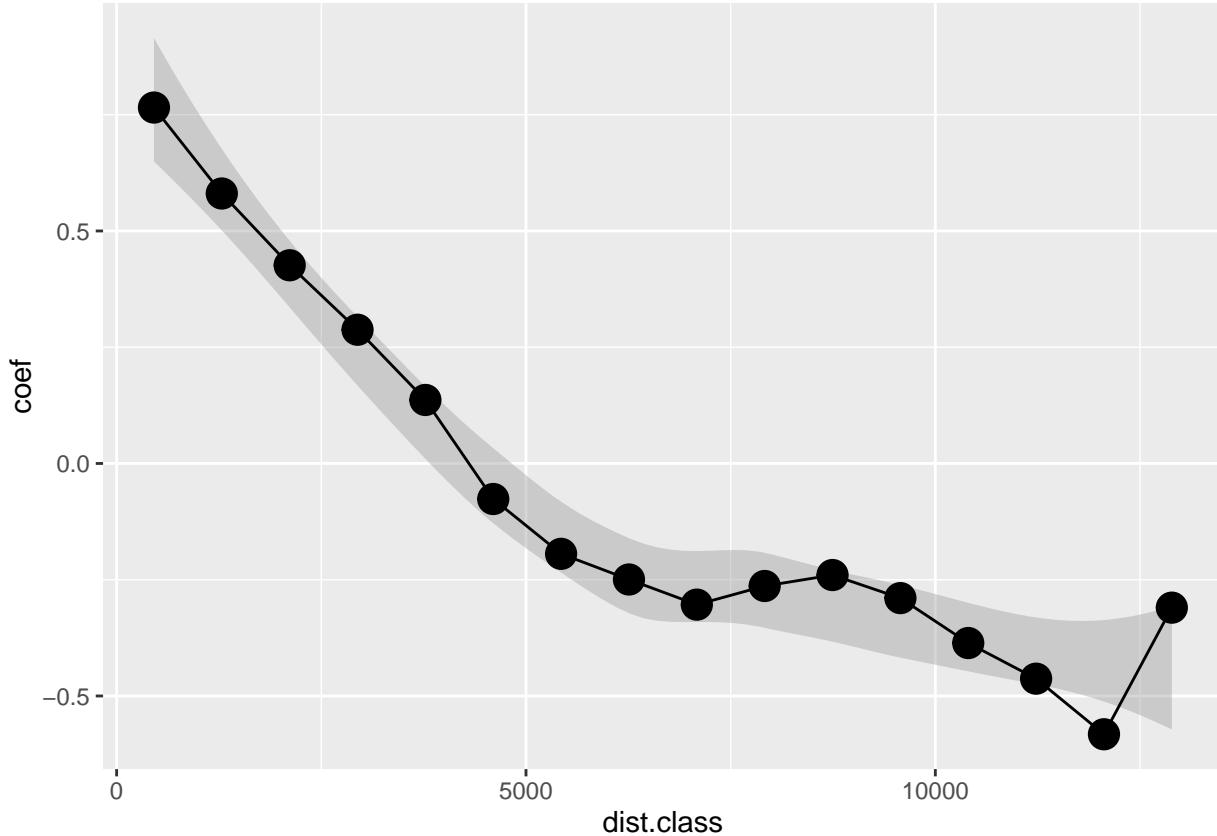
```
kowario <- variogram(temp~1, punkty, covariogram = TRUE)
plot(kowario)
```



### 6.2.6 Autokorelacja

- Autokoreogram jest wykresem pokazującym jedną z miar autokorelacji (np. I Morana lub C Geary'ego) w stosunku do odległości

```
wsp <- coordinates(punkty)
kor <- correlog(wsp, punkty$temp)
kor <- as.data.frame(kor)
ggplot(kor, aes(dist.class, coef)) + geom_smooth(linetype=0) + geom_line() + geom_point(size=5)
```



### 6.2.7 Semiwariancja

- Zmienność przestrzenna może być określona za pomocą semiwariancji. Jest to połowa średniej kwadratu różnicy pomiędzy wartościami badanej zmiennej w dwóch lokalizacjach odległych o wektor  $h$

$$\gamma(h) = \frac{1}{2} E[(z(s) - z(s + h))^2]$$

### 6.2.8 Określenie występowania autokorelacji przestrzennej | Chmura semiwarioramu

- Jeżeli w badanej próbie mamy  $n$  obserwacji oznacza to, że możemy zaobserwować  $\frac{1}{2}n(n - 1)$  par obserwacji
- Każda para obserwacji daje nam informacje o semiwariancji występującej wraz z odległością
- Semiwariancję można zaprezentować na wykresie zwanym chmurą semiwarioramu

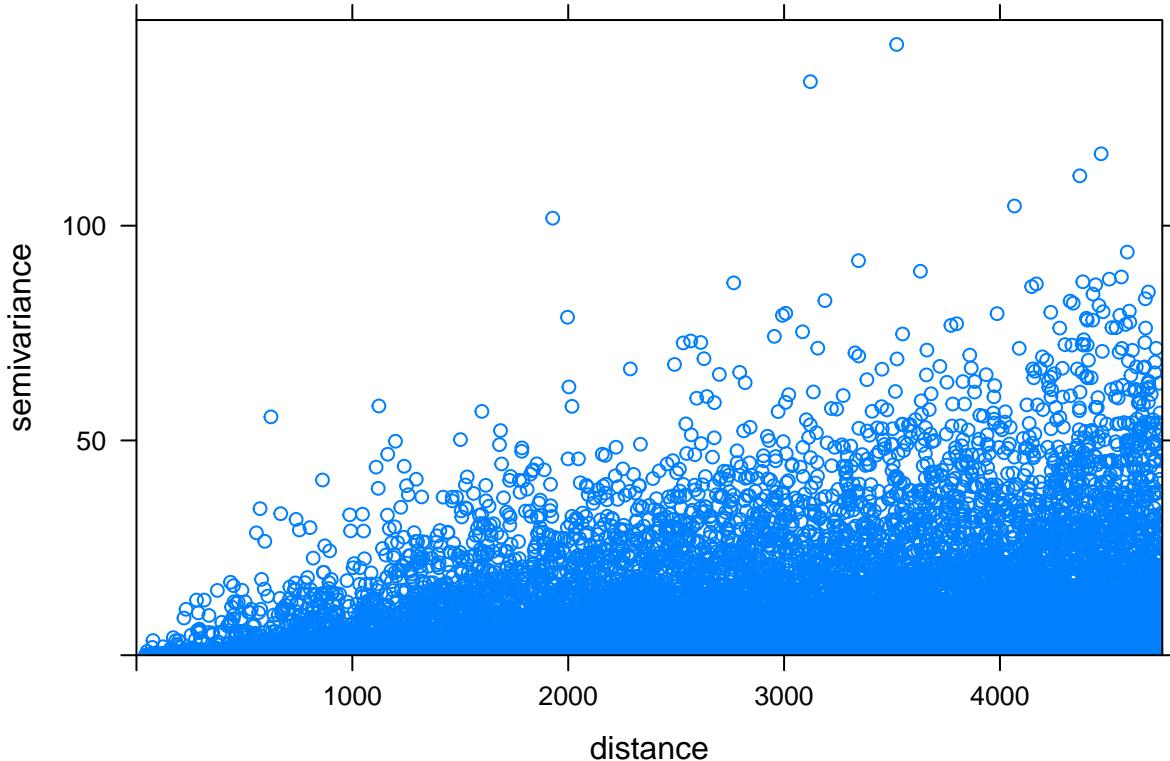
### 6.2.9 Semiwariancja | Przykładowe obliczenia

```
odl <- dist(coordinates(punkty)[c(1, 2), ])
gamma <- 0.5 * (punkty$temp[1] - punkty$temp[2])^2
gamma
```

```
## [1] 20.60122
```

### 6.2.10 Określenie występowania autokorelacji przestrzennej | Chmura semiwariorogramu

```
vario_cloud <- variogram(temp~1, punkty, cloud=TRUE)
plot(vario_cloud)
```



```
vario_cloud_sel <- plot(variogram(temp~1, punkty, cloud=TRUE), digitize=TRUE)
plot(vario_cloud_sel, punkty)
```

### 6.2.11 Semiwariogram | Charakterystyka struktury przestrzennej

- Semiwariogram to wykres pokazujący relację pomiędzy odległością a semiwariancją
- Jest to uśrednieniem semiwariancji dla kolejnych odstępów (lagów)
- W oparciu o semiwariogram empiryczny możemy następnie dopasować do niego model/e

$$\hat{\gamma}(h) = \frac{1}{2N(h)} \sum_{i=1}^{N(h)} (z(s_i) - z(s_i + h))^2$$

gdzie  $N(h)$  oznacza liczbę par punktów w odstępie  $h$

### 6.2.12 Semiwariogram

- **Nugget** - efekt nuggetowy - pozwala na określenie błędu w danych wejściowych oraz zmienności na dystansie krótszym niż pierwszy odstęp
- **Sill** - semiwariancja progowa - oznacza wariancję badanej zmiennej
- **Range** - zasięg - to odległość do której istnieje przestrzenna korelacja

### 6.2.13 Semiwariogram | Rules of thumb

- W każdym odstępie powinno się znaleźć co najmniej 30 par punktów
- Maksymalny zasięg semiwariogramu (ang. *cutoff distance*) to 1/2 pierwiastka z badanej powierzchni (inne źródła mówią o połowie z przekątnej badanego obszaru/jednej trzeciej)
- Liczba odstępów powinna nie być mniejsza niż 10
- Optymalnie maksymalny zasięg semiwariogramu powinien być dłuższy o 10-15% od zasięgu zjawiska
- Optymalnie odstępy powinny być jak najbliżej siebie i jednocześnie nie być chaotyczne
- Warto metodą prób i błędów określić optymalne parametry semiwariogramu
- Należy określić czy zjawisko wykazuje anizotropię przestrzenną

### 6.2.14 Semiwariogram | Obliczenia pomocnicze

- Liczba par obserwacji

```
0.5 * length(punkty) * (length(punkty) - 1)
```

```
## [1] 31125
```

- Połowa pierwiastka powierzchni

```
pow <- area(granica)
as.vector(0.5 * sqrt(pow))
```

```
## [1] 3980.472
```

- Powierzchnia zajmowana przez jedną próbkę

```
pow_pr <- area(granica) / length(punkty)
pow_pr
```

```
##      117
## 253506.6
```

- Średnia odległość między punktami

```
sqrt(pow_pr)
```

```
##      117
## 503.4944
```

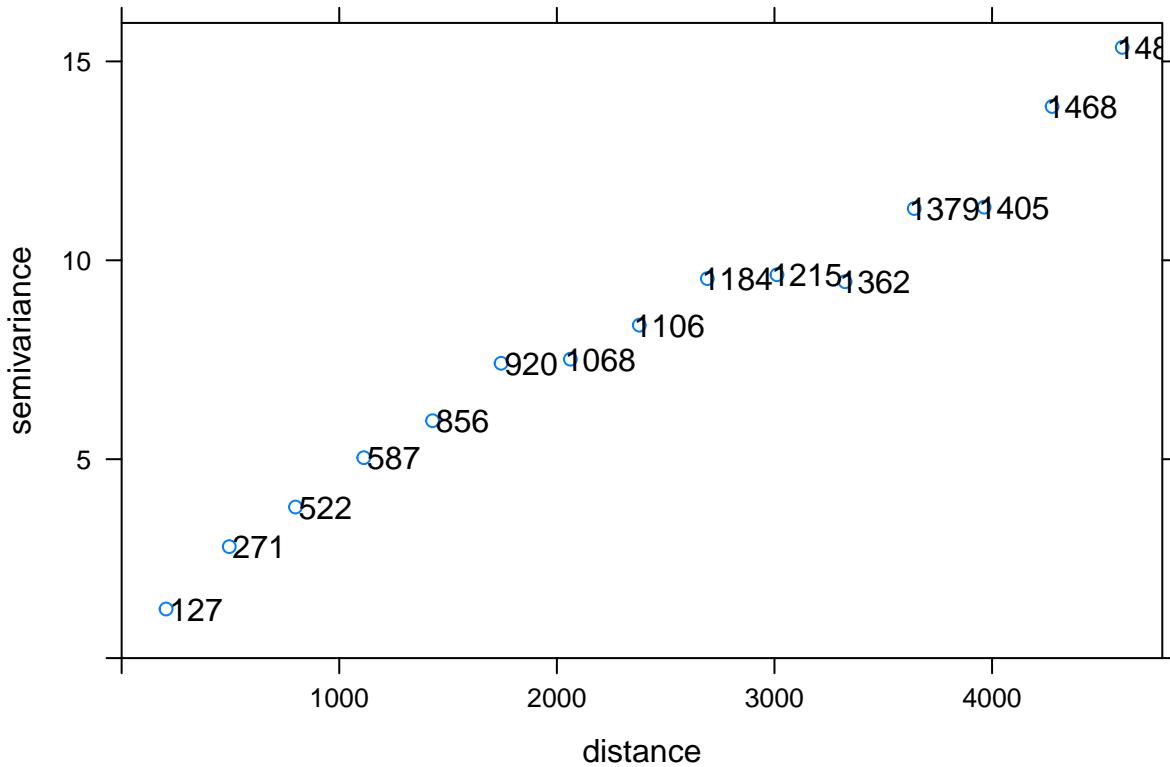
### 6.2.15 Semiwariogram | Maksymalny zasięg semiwariogramu (ang. *Cutoff distance*)

- Maksymalny zasięg semiwariogramu (ang. *Cutoff distance*) jest domyślnie wyliczany w pakiecie `gstat` jako 1/3 z najdłuższej przekątnej badanego obszaru

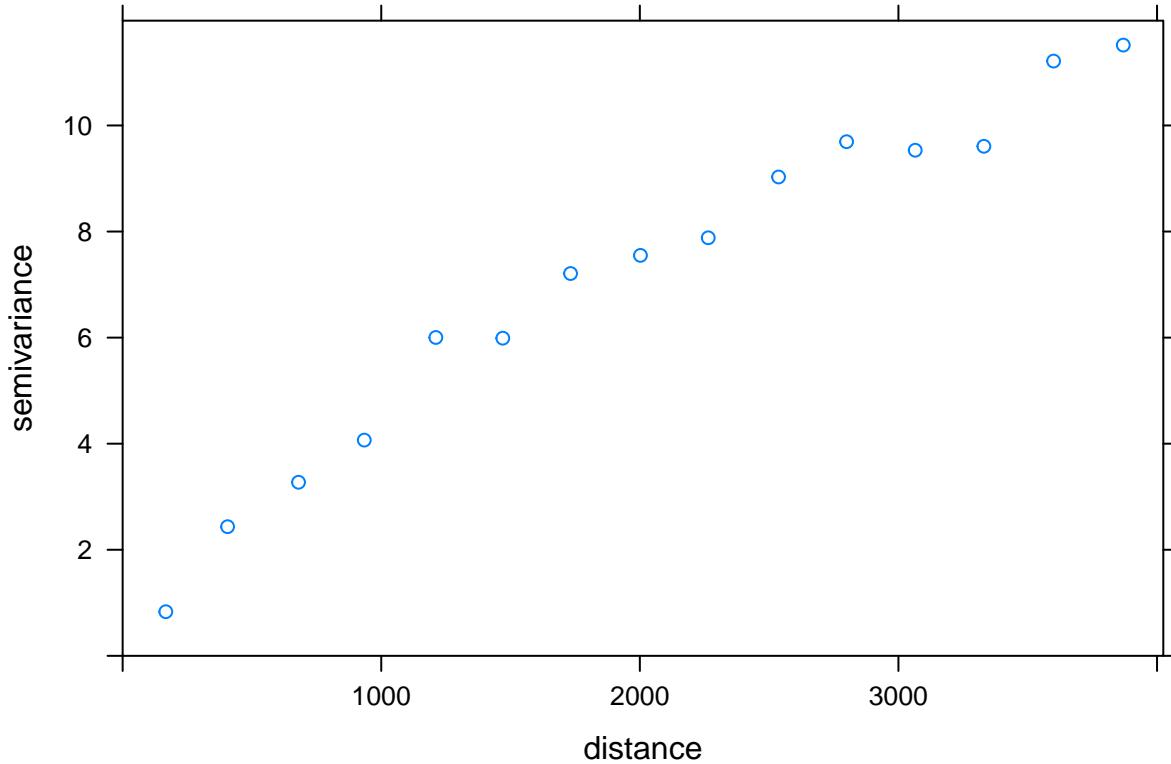
```
vario_par <- variogram(temp~1, punkty)
vario_par
```

```
##      np      dist     gamma dir.hor dir.ver   id
## 1    127  204.2244  1.233999      0      0 var1
## 2    271  494.5572  2.802019      0      0 var1
## 3    522  798.7911  3.797612      0      0 var1
## 4    587 1112.7783  5.037545      0      0 var1
## 5    856 1428.7866  5.967276      0      0 var1
## 6    920 1743.7864  7.411276      0      0 var1
## 7   1068 2062.3041  7.514067      0      0 var1
## 8   1106 2378.9333  8.369087      0      0 var1
## 9   1184 2691.5206  9.539870      0      0 var1
## 10  1215 3011.7729  9.636891      0      0 var1
## 11  1362 3324.6705  9.461429      0      0 var1
## 12  1379 3642.5616 11.301515      0      0 var1
## 13  1405 3963.6776 11.335589      0      0 var1
## 14  1468 4276.0078 13.866888      0      0 var1
## 15  1482 4598.4144 15.353206      0      0 var1
```

```
plot(vario_par, plot.numbers=TRUE)
```



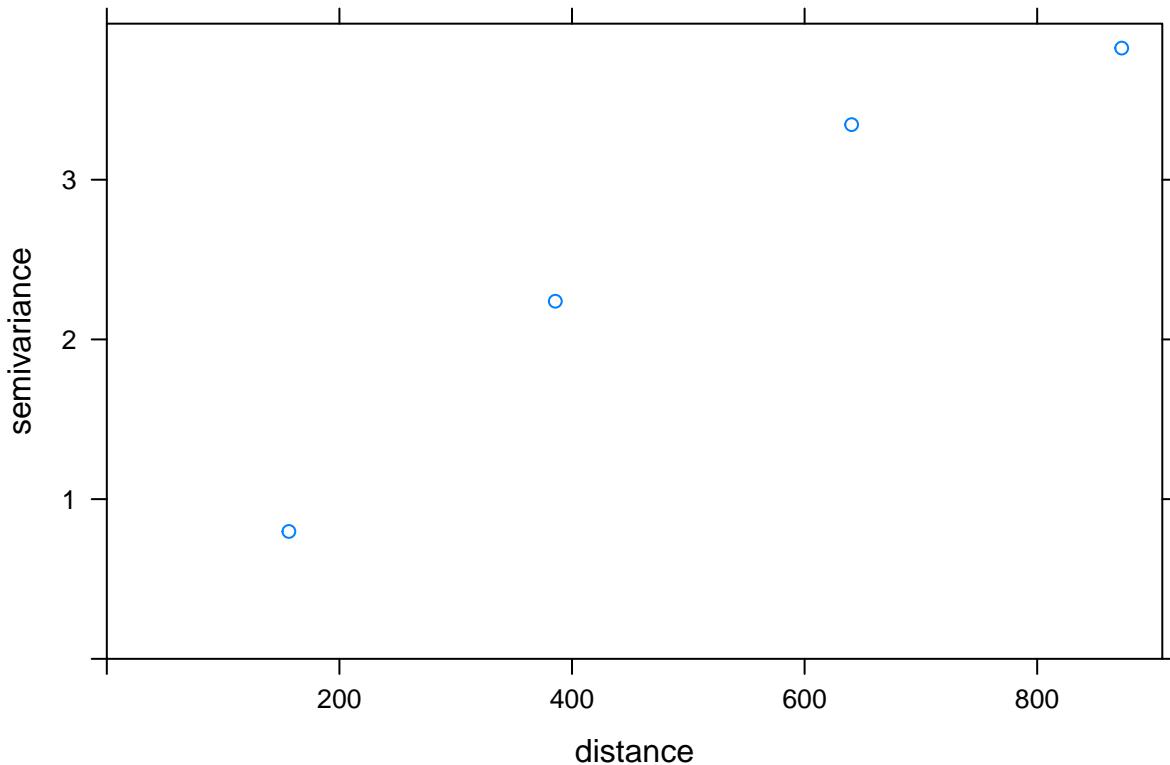
```
vario_par <- variogram(temp~1, punkty, cutoff = 4000)
plot(vario_par)
```



### 6.2.16 Semiwariogram | Odległość między przedziałami (ang. *Interval width*)

- Domyślnie to maksymalny zasięg semiwariogramu podzielony przez 15 dahe odległość między przedziałami (ang. *Interval width*)

```
vario_par <- variogram(temp~1, punkty, cutoff = 1000, width = 250)
plot(vario_par)
```



## 6.3 Anizotropia

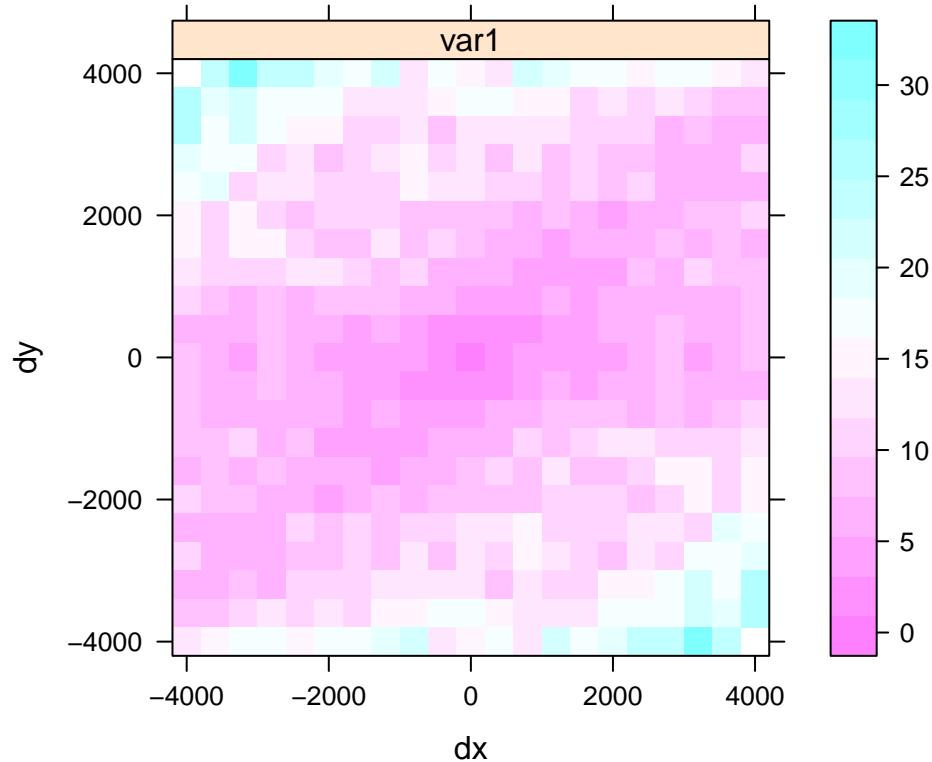
### 6.3.1 Anizotropia struktury przestrzennej

- W wielu rzeczywistych sytuacjach, wartość cechy zależy nie tylko od odległości, ale także od kierunku

### 6.3.2 Mapa semiwariogramu

- Mapa semiwariogramu (powierzchnia semiwariogramu) służy do określenia czy struktura przestrzenna zjawiska posiada anizotropię, a jeżeli tak to w jakim kierunku
- Na podstawie mapy semiwariogramu identyfikuje się parametry potrzebne do zbudowania semiwariogramów kierunkowych

```
vario_map <- variogram(temp~1, punkty, cutoff=4000, width=400, map=TRUE)
plot(vario_map, threshold=30) # co najmniej 30 par punktów
```



### 6.3.3 Mapa semiwariogramu | 3D

```
plot3D(raster(vario_map$map), col=rainbow)
```

### 6.3.4 Semiwariogramy kierunkowe | Kierunki

- W przypadku, gdy zjawisko wykazuje anizotropię przestrzenną, możliwe jest stworzenie semiwariogramów dla różnych wybranych kierunków
- Przykładowo, dla argumentu  $\alpha = c(0, 45, 90, 135)$  cztery główne kierunki to 0, 45, 90 i 135 stopni. Oznacza to, że dla kierunku 45 stopni brane pod uwagę będą wszystkie pary punktów pomiędzy 22,5 a 67,5 stopnia.

```
vario_kier <- variogram(temp~1, punkty, alpha = c(0, 45, 90, 135))
plot(vario_kier)
```

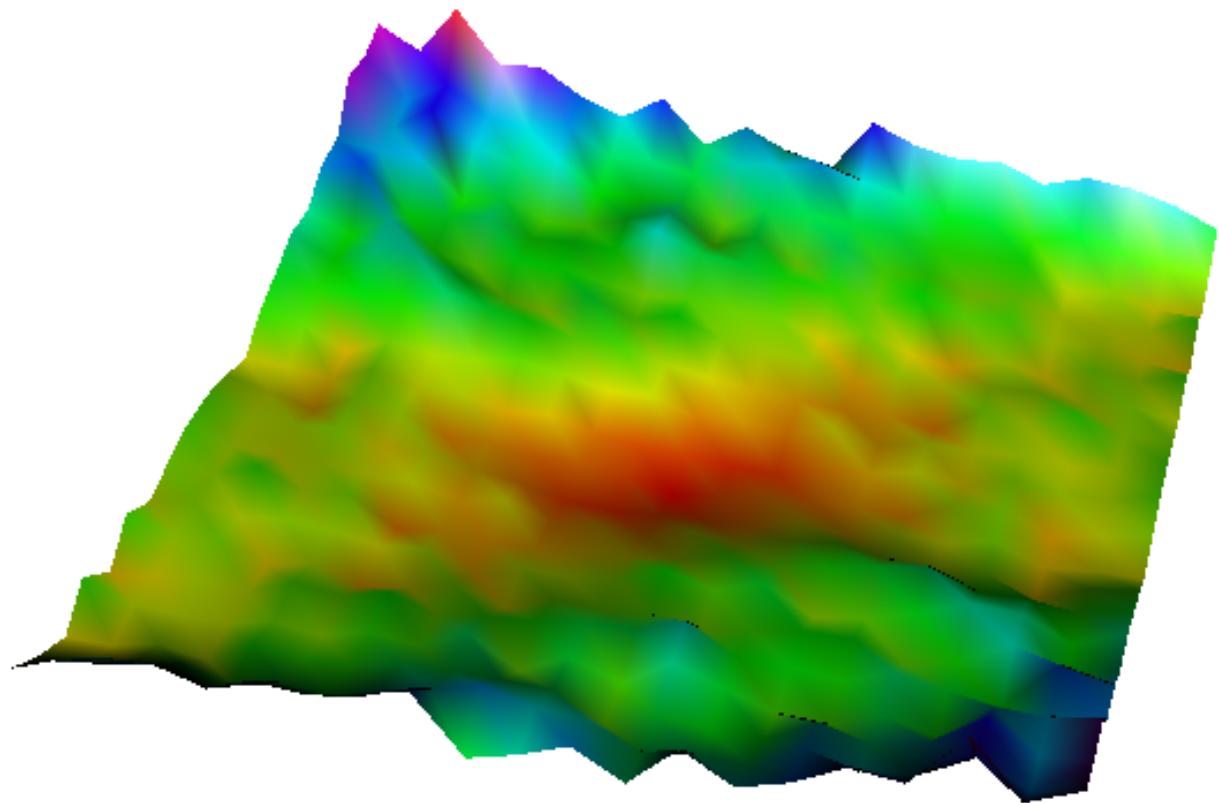
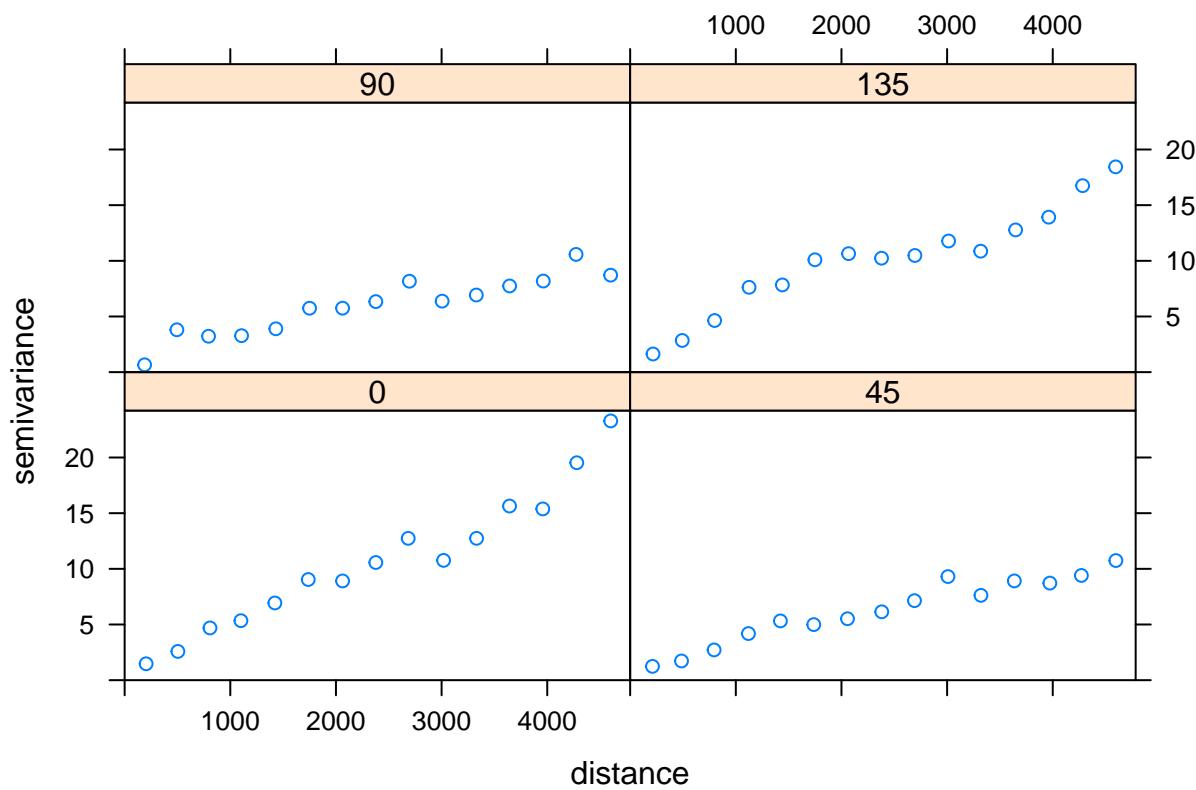


Figure 6.2:





# Chapter 7

## Modelowanie matematycznie autokorelacji przestrzennej

```
library('geostatbook')
data(punkty)
```

### 7.1 Modelowanie matematycznie autokorelacji przestrzennej

#### 7.1.1 Modelowanie matematycznie autokorelacji przestrzennej

- Semiwariogram empiryczny jest:
  - Nieciągły - wartości semiwariancji są średnimi przedziałowymi
  - Chaotyczny - badana próba jest jedynie przybliżeniem rzeczywistości, dodatkowo obciążonym błędami
- Estymacje i symulacje przestrzenne wymagają modelu struktury przestrzennej analizowanej cechy, a nie tylko wartości empirycznych (wyliczonych z danych)
- Dodatkowo, matematycznie modelowanie wygładza chaotyczne fluktuacje danych empirycznych

### 7.2 Modele podstawowe

#### 7.2.1 Modele podstawowe

- Nuggetowy (ang. *Nugget effect model*)
- Sferyczny (ang. *Spherical model*)
- Gaussowski (ang. *Gaussian model*)
- Potęgowy (ang. *Power model*)
- Wykładniczy (ang. *Exponential model*)
- Inne

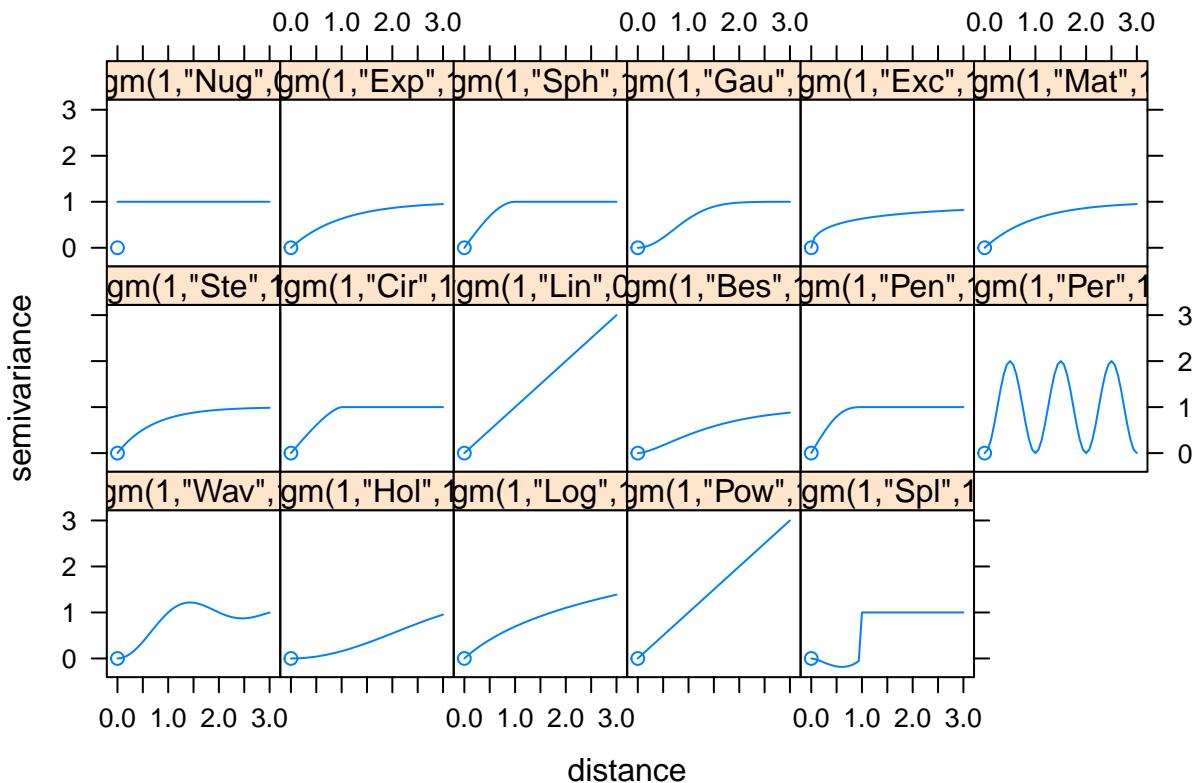
```
vgm()
```

```

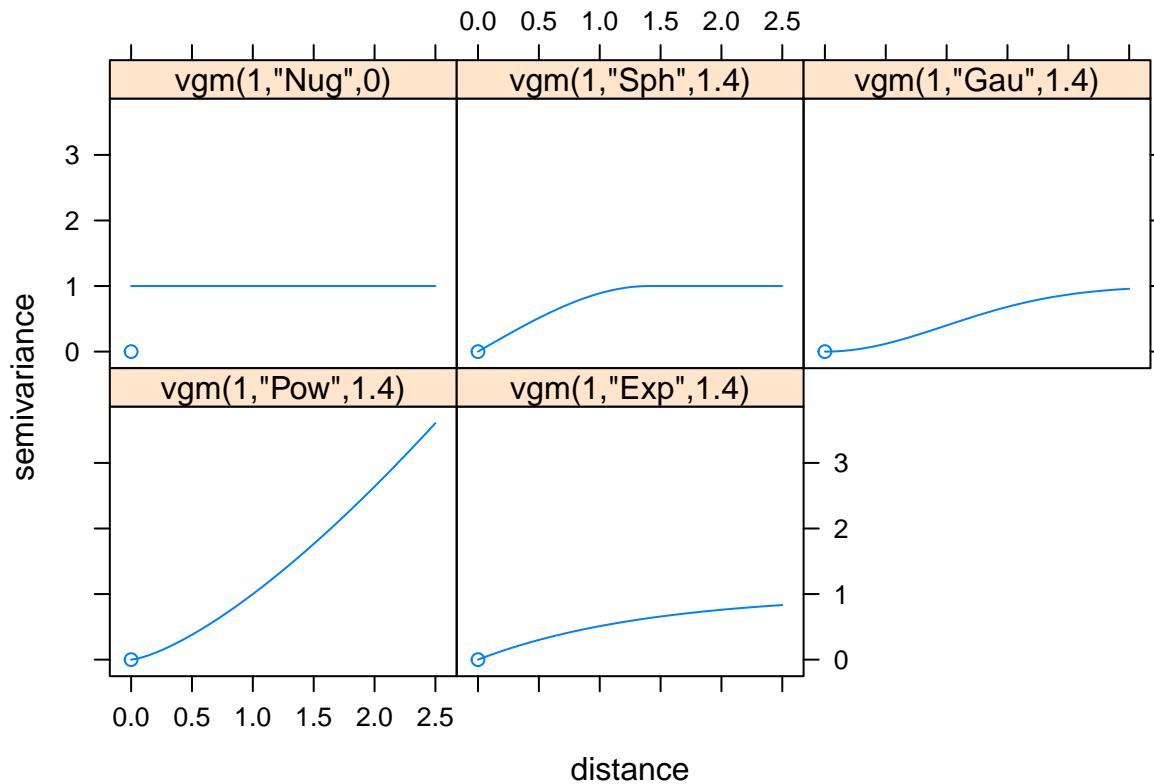
##      short                               long
## 1    Nug                                Nug (nugget)
## 2    Exp                                Exp (exponential)
## 3    Sph                                Sph (spherical)
## 4    Gau                                Gau (gaussian)
## 5    Exc      Exclass (Exponential class/stable)
## 6    Mat                                Mat (Matern)
## 7    Ste Mat (Matern, M. Stein's parameterization)
## 8    Cir                                Cir (circular)
## 9    Lin                                Lin (linear)
## 10   Bes                                Bes (bessel)
## 11   Pen                                Pen (pentaspherical)
## 12   Per                                Per (periodic)
## 13   Wav                                Wav (wave)
## 14   Hol                                Hol (hole)
## 15   Log                                Log (logarithmic)
## 16   Pow                                Pow (power)
## 17   Spl                                Spl (spline)
## 18   Leg                                Leg (Legendre)
## 19   Err      Err (Measurement error)
## 20   Int                                Int (Intercept)

```

```
show.vgms()
```

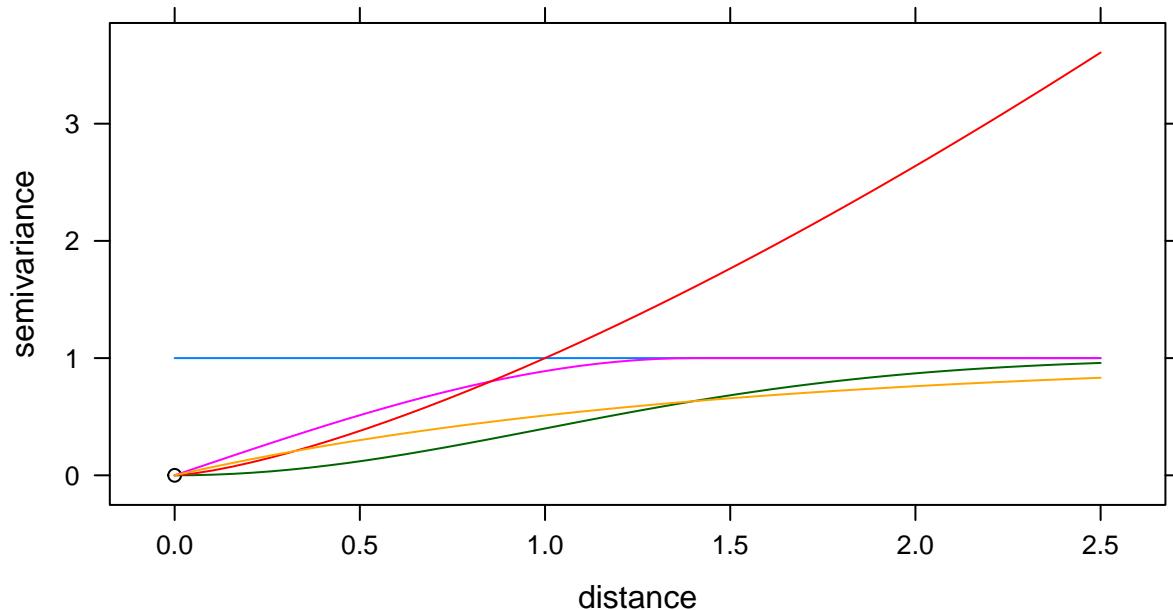


```
show.vgms(models=c('Nug', 'Sph', 'Gau', 'Pow', 'Exp'), range=1.4, max=2.5)
```



```
show.vgms(models=c('Nug', 'Sph', 'Gau', 'Pow', 'Exp'), range=1.4, max=2.5, as.groups = TRUE)
```

vgm(1, "Nug", 0)	○
vgm(1, "Sph", 1.4)	○
vgm(1, "Gau", 1.4)	○
vgm(1, "Pow", 1.4)	○
vgm(1, "Exp", 1.4)	○



## 7.3 Metody modelowania

### 7.3.1 Metody modelowania

- Ustawianie “ręczne” parametrów modelu, np. funkcja `vgm` z pakietu `gstat`
- Ustawianie “wizualne” parametrów modelu, np. funkcja `eyefit` z pakietu `geoR`
- Automatyczny wybór parametrów na podstawie różnych kryterów statystycznych, np. funkcja `fit.variogram` z pakietu `gstat`, `variofit` z pakietu `geoR`, `autofitVariogram` z pakietu `automap`
- Odpowiednie określenie modelu matematycznego często nie jest proste
- Automatyczne metody nie zawsze są w stanie dać lepszy wynik od modelowania “ręcznego”
- Najlepiej, gdy wybór modelu oparty jest o wiedzę na temat zakładanego procesu przestrzennego

### 7.3.2 Metody modelowania | Liniowy model regionalizacji

- W przypadku, gdy analizowane zjawisko jest złożone, odwzorowanie kształtu semiwariogramu empirycznego wymaga połączenia dwóch lub większej liczby modeli podstawowych
- W takiej sytuacji konieczne jest spełnienie dwóch warunków:
  - Wszystkie zastosowane modele muszą być dopuszczalne (`vgm()`)
  - Wariancja progowa każdego podstawowego modelu musi być dodatnia

## 7.4 Modelowanie semiwariogramu

### 7.4.1 Modelowanie semiwariogramu | funkcja `fit.variogram`

- Funkcja `fit.variogram` z pakietu `gstat` dopasowuje zasięg oraz semiwariancję progową w oparciu o ustalone “ręczne” parametry modelu

## 7.5 Modelowanie izotropowe

### 7.5.1 Modelowanie izotropowe | Modelowanie “wizualne”

```
v_eye <- eyefit(variog(as.geodata(punkty, 'temp'))))
ve_fit <- as.vgm.variomodel(v_eye[[1]])
```

### 7.5.2 Modelowanie izotropowe | Model nuggetowy

```
vario <- variogram(temp~1, punkty)
plot(vario)
model_nug <- vgm(10, model = 'Nug', range=0)
model_nug
plot(vario, model=model_nug)
fitted_nug <- fit.variogram(vario, model_nug)
fitted_nug
plot(vario, model=fitted_nug)
```

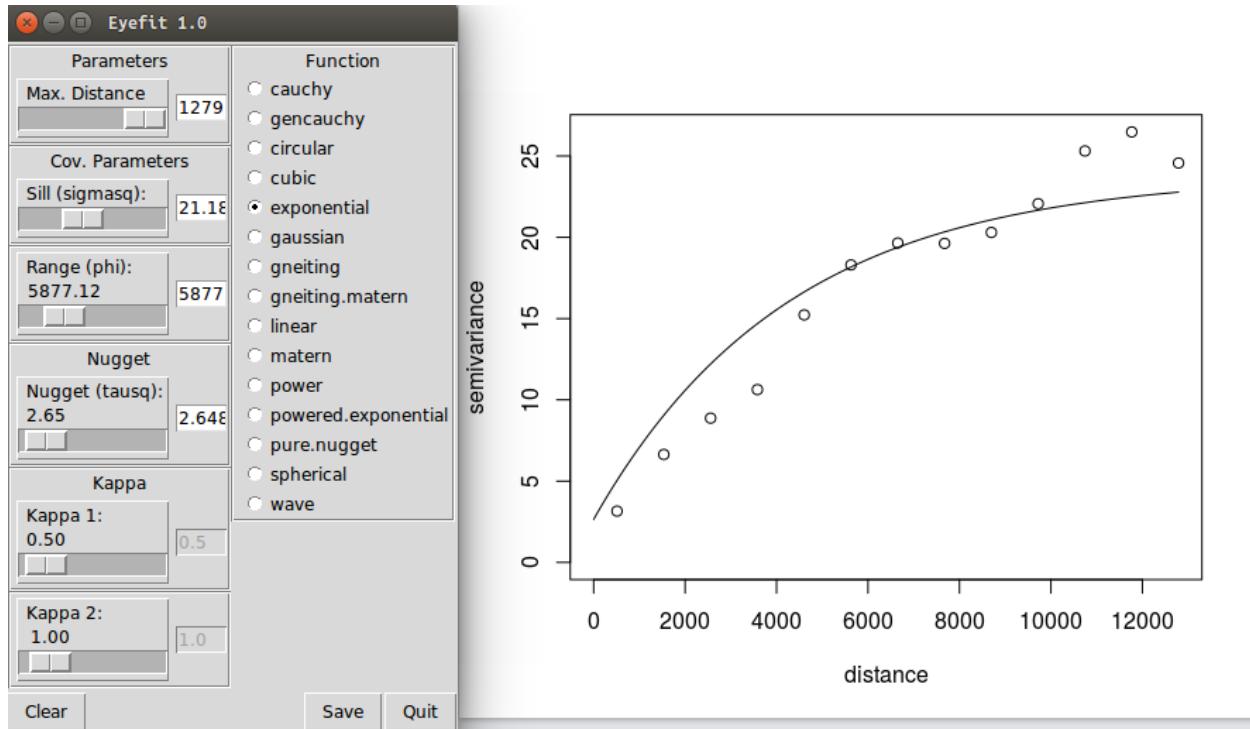
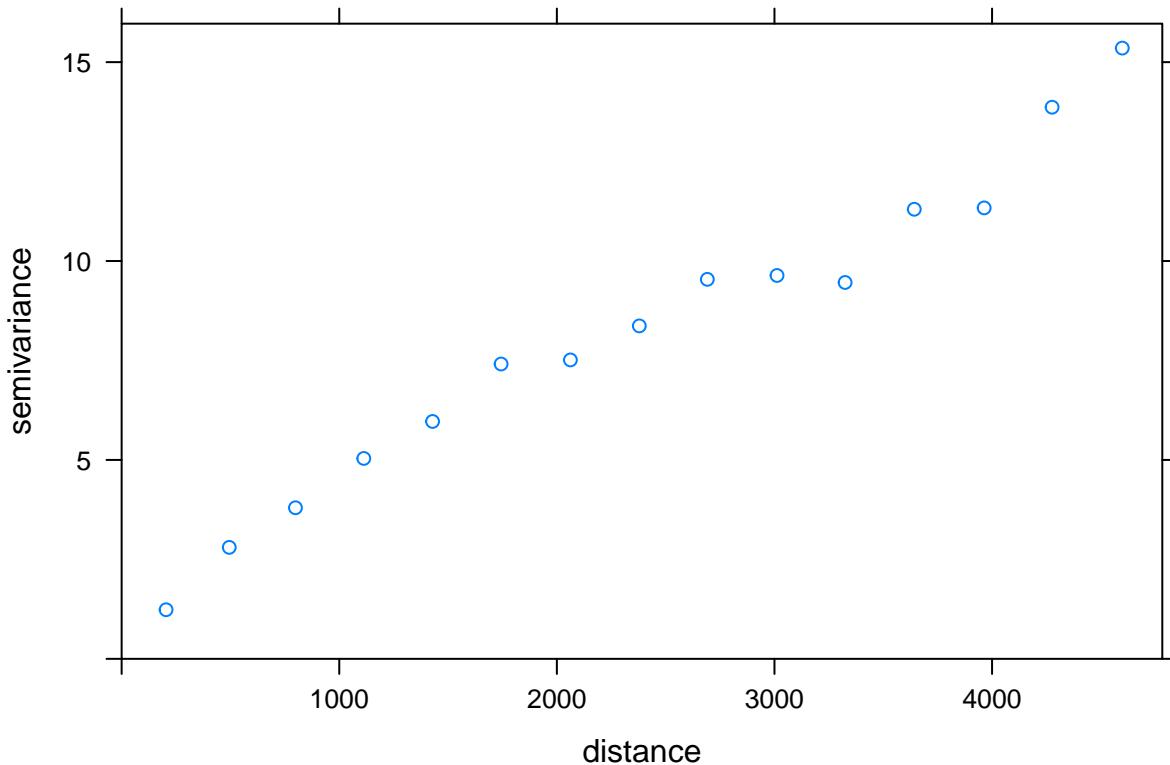


Figure 7.1:

### 7.5.3 Modelowanie izotropowe | Model sferyczny

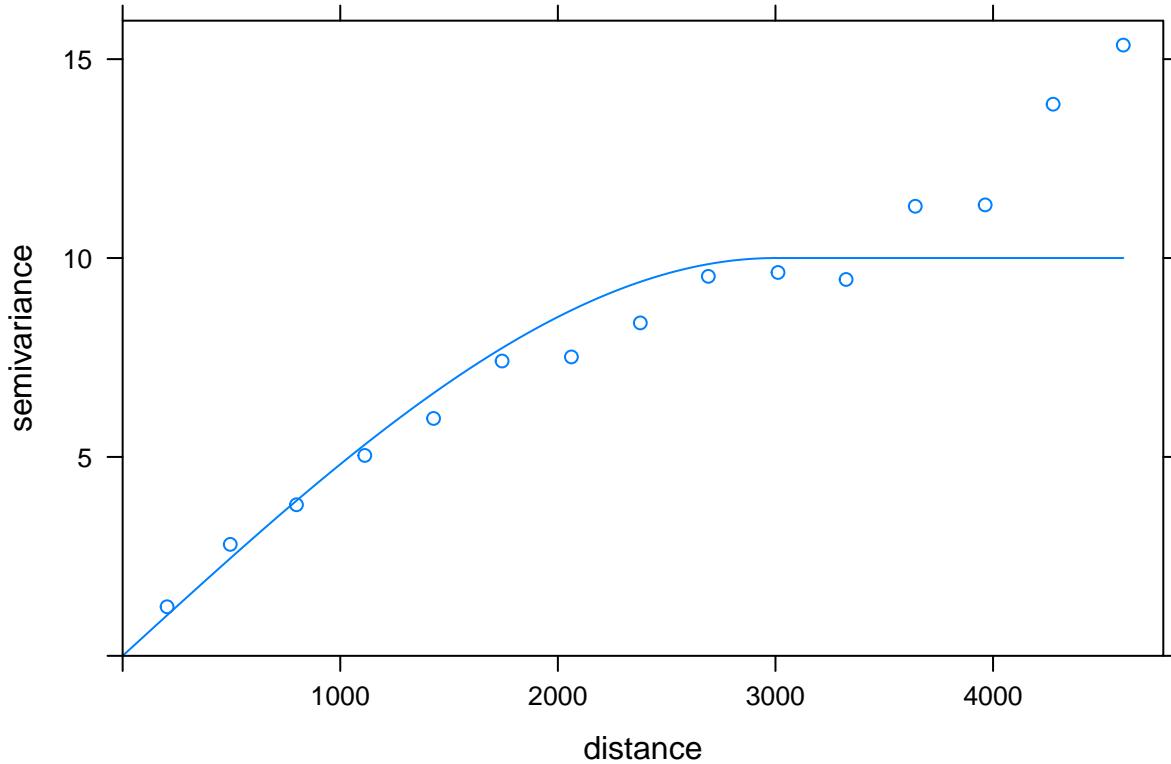
```
vario <- variogram(temp~1, punkty)
plot(vario)
```



```
model_sph <- vgm(psill=10, model = 'Sph', range=3000)
model_sph
```

```
##   model psill range
## 1   Sph    10  3000
```

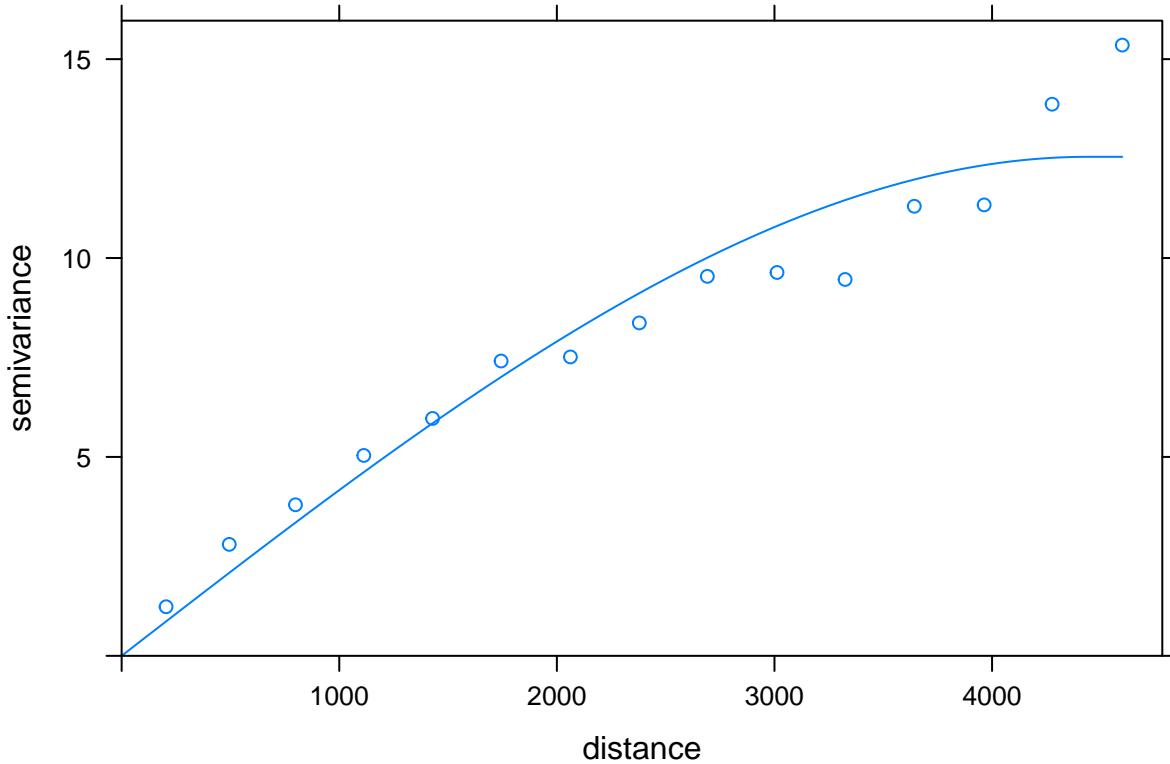
```
plot(vario, model=model_sph)
```



```
fitted_sph <- fit.variogram(vario, model_sph)
fitted_sph
```

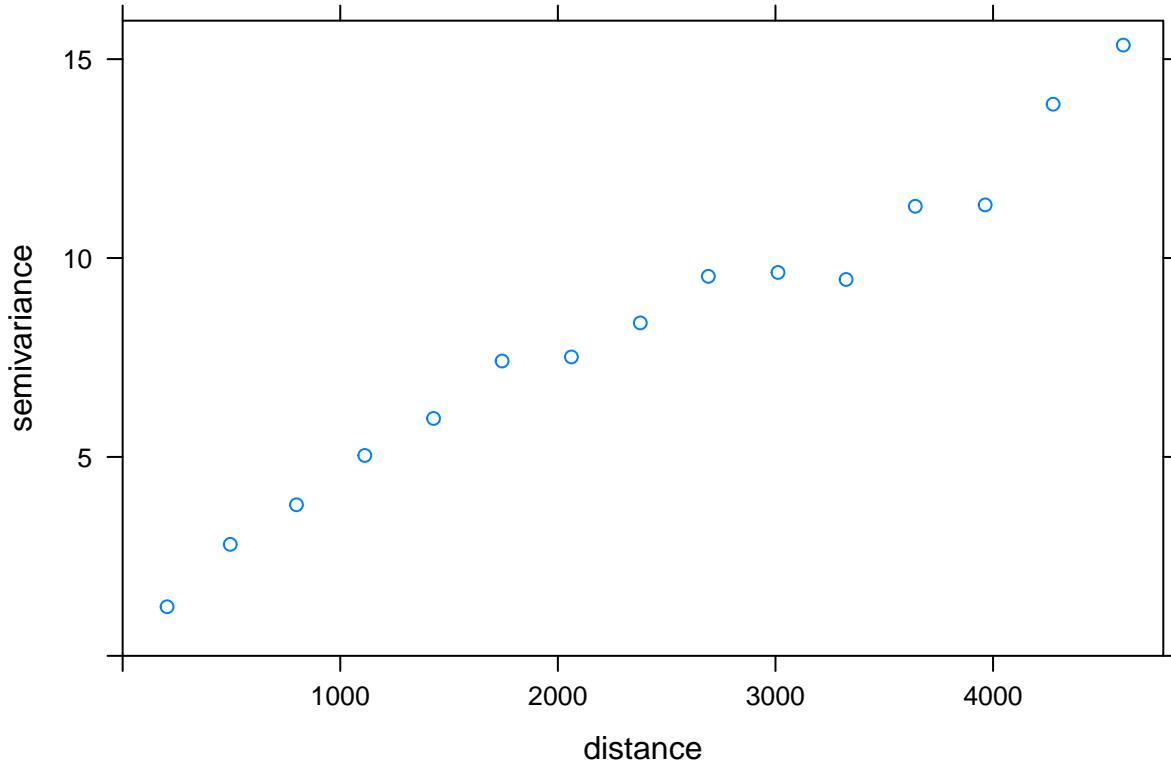
```
##   model   psill     range
## 1   Sph 12.5445 4440.768
```

```
plot(vario, model=fitted_sph)
```



#### 7.5.4 Modelowanie izotropowe | Model Gaussowski

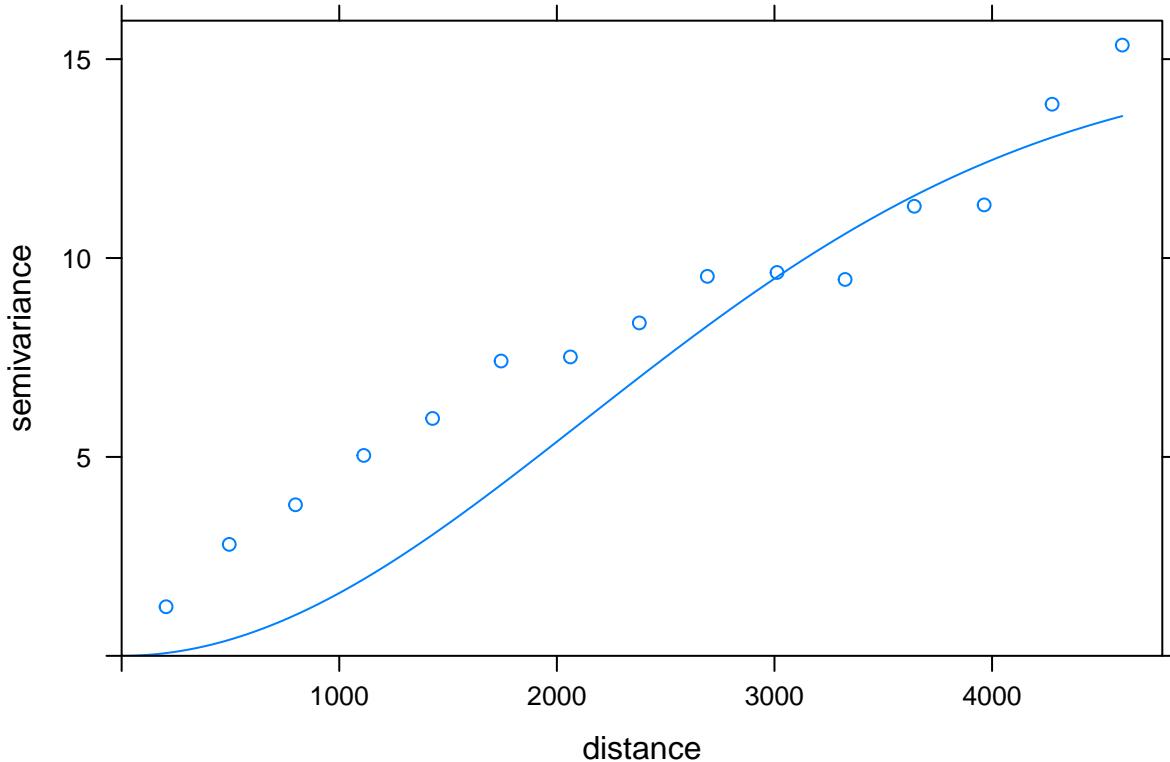
```
vario <- variogram(temp~1, punkty)
plot(vario)
```



```
model_gau <- vgm(psill=15, model = 'Gau', range=3000)
model_gau
```

```
##   model psill range
## 1   Gau    15  3000
```

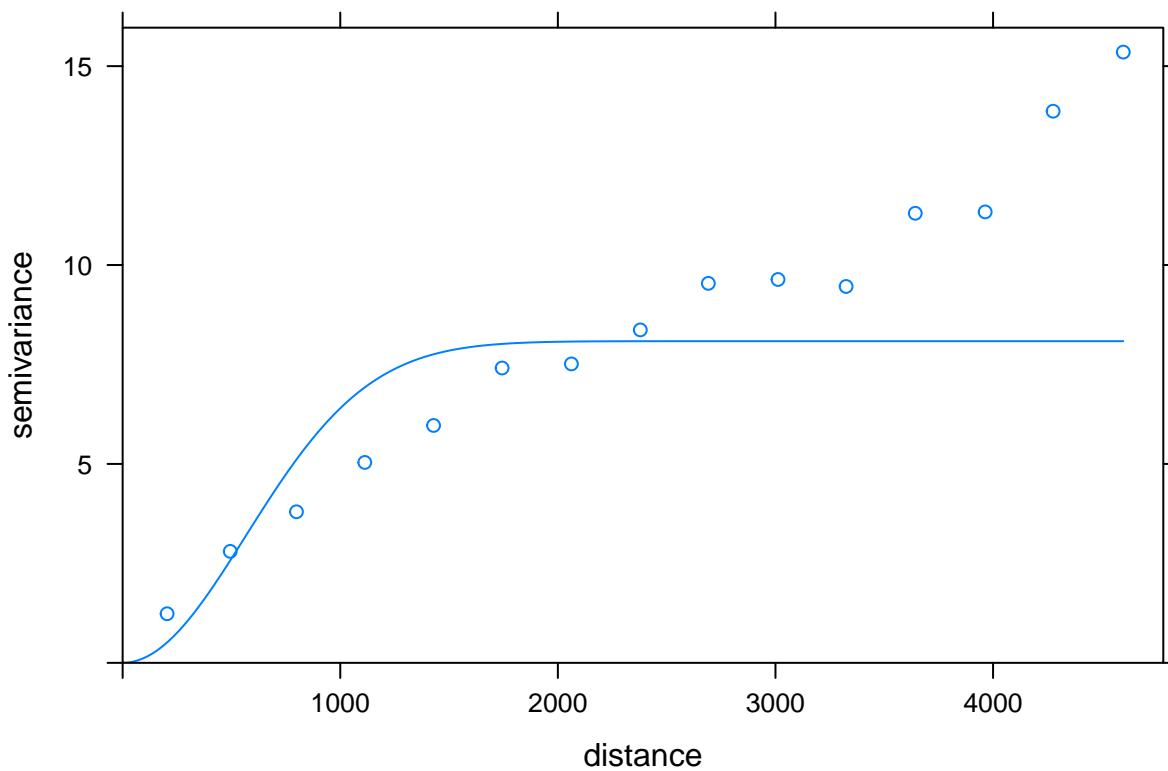
```
plot(vario, model=model_gau)
```



```
fitted_gau <- fit.variogram(vario, model_gau)
fitted_gau
```

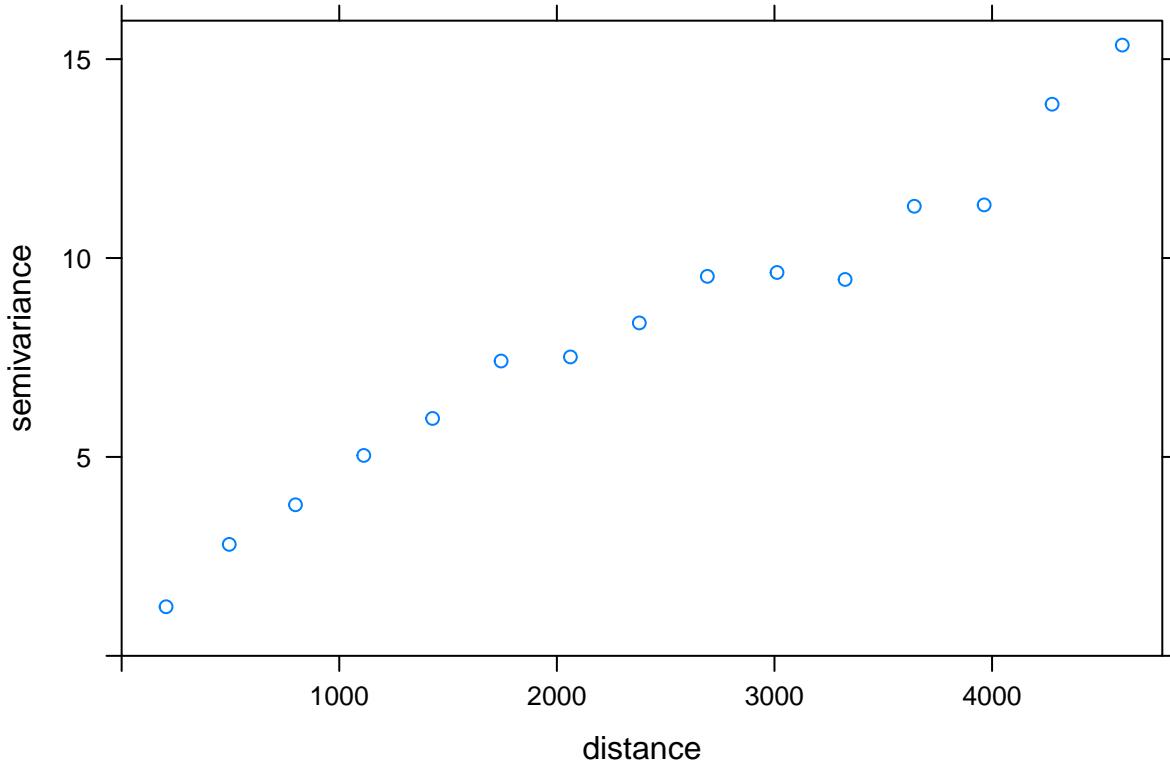
```
##   model    psill     range
## 1   Gau 8.085963 798.7454
```

```
plot(vario, model=fitted_gau)
```



### 7.5.5 Modelowanie izotropowe | Model potęgowy

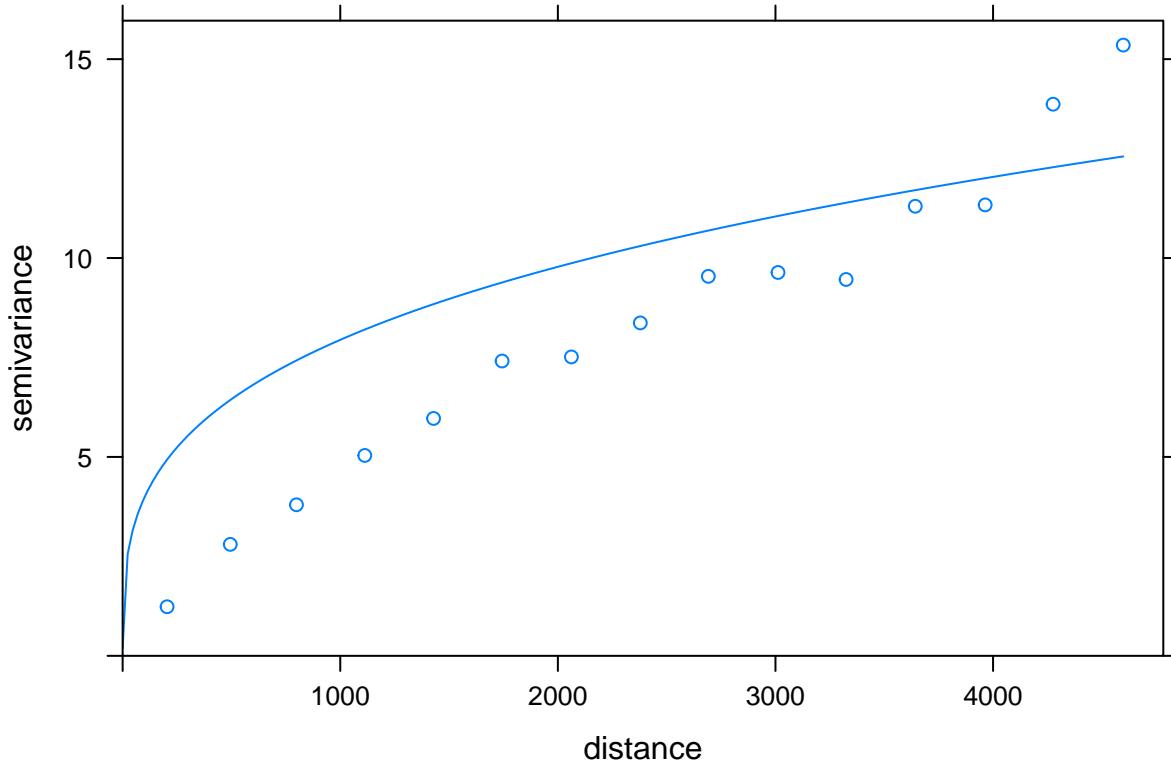
```
vario <- variogram(temp~1, punkty)
plot(vario)
```



```
model_pow <- vgm(psill=1, model = 'Pow', range=0.30)
model_pow
```

```
##   model psill range
## 1   Pow     1    0.3
```

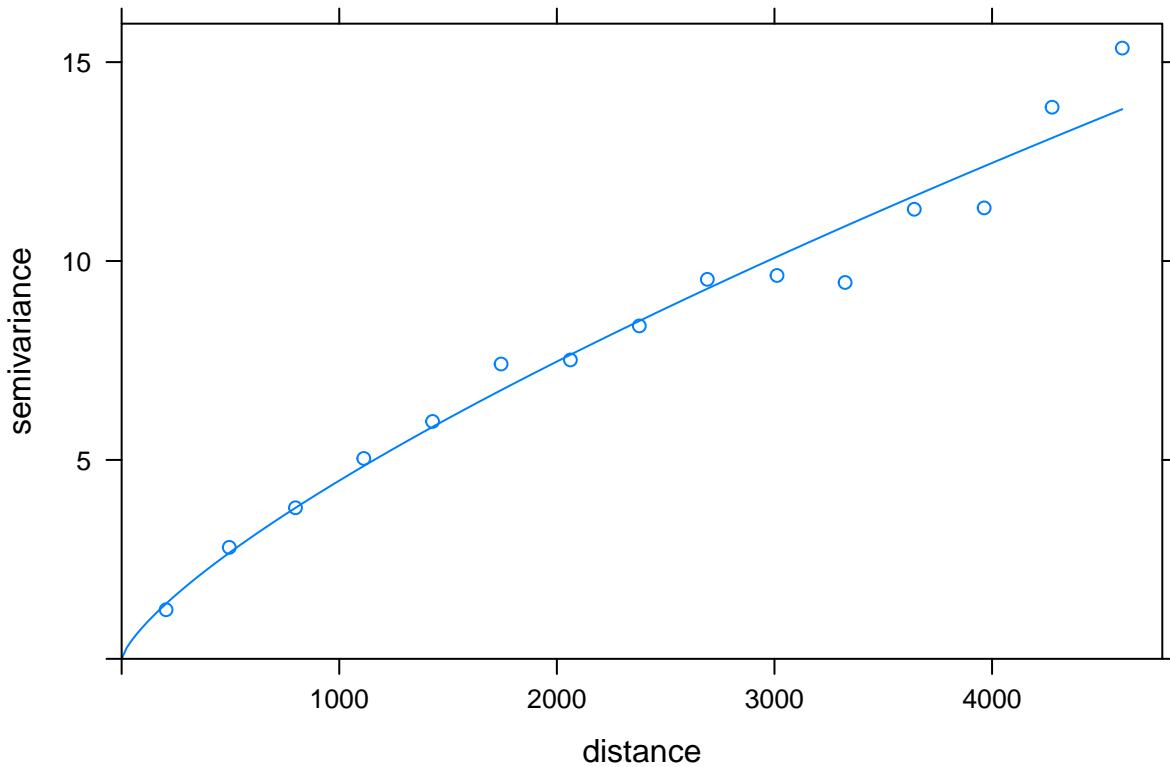
```
plot(vario, model=model_pow)
```



```
fitted_pow <- fit.variogram(vario, model_pow)  
fitted_pow
```

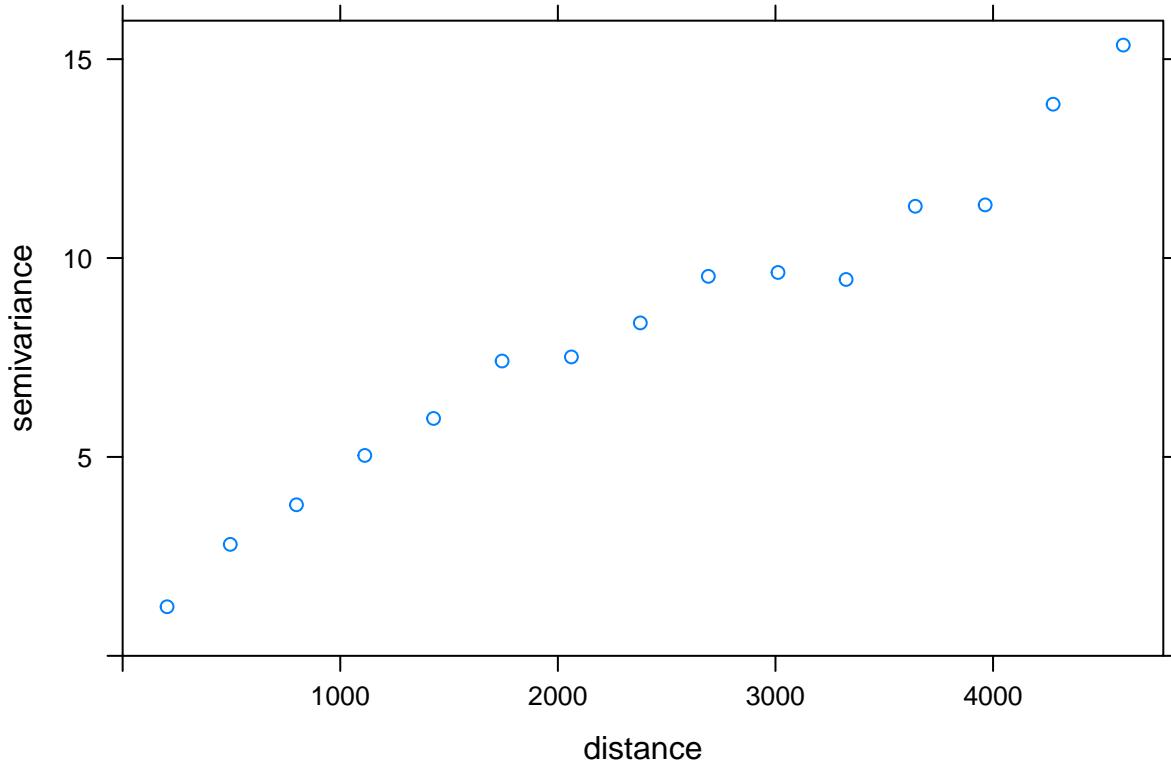
```
##   model      psill      range  
## 1 Pow 0.02732271 0.7382383
```

```
plot(vario, model=fitted_pow)
```



### 7.5.6 Modelowanie izotropowe | Model wykładniczy

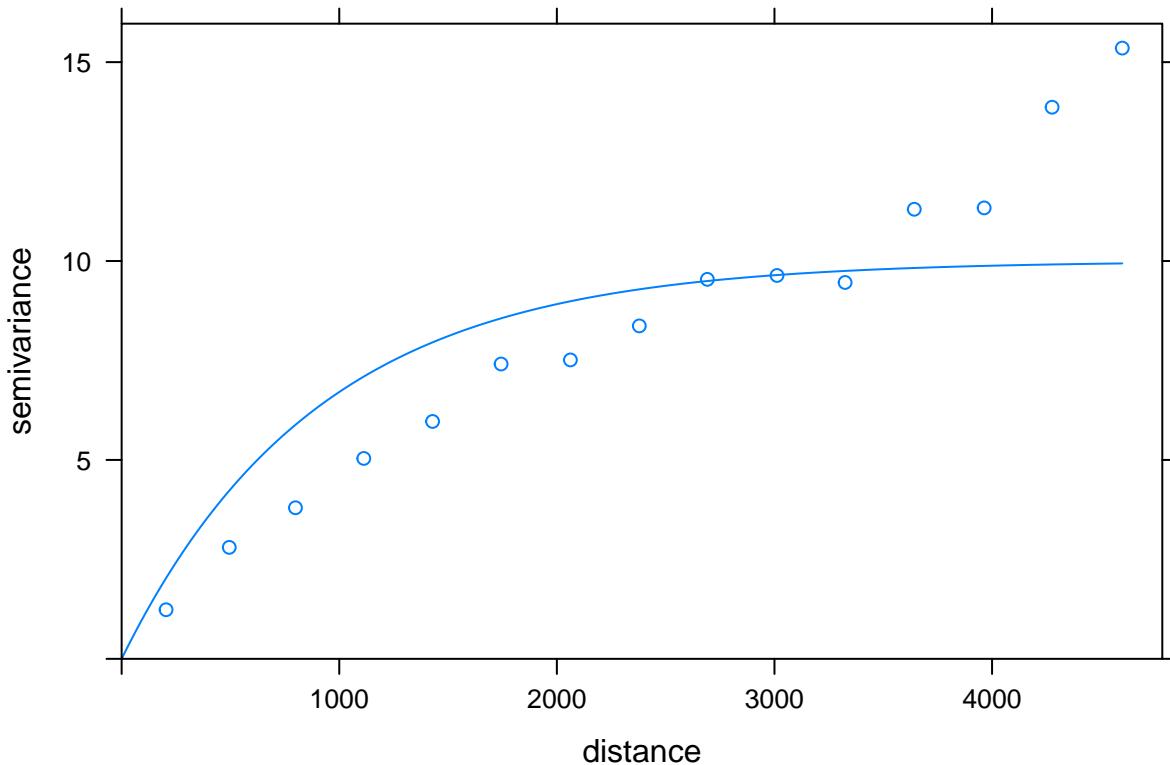
```
vario <- variogram(temp~1, punkty)
plot(vario)
```



```
model_exp <- vgm(psill=10, model = 'Exp', range=900)
model_exp
```

```
##   model psill range
## 1   Exp    10    900
```

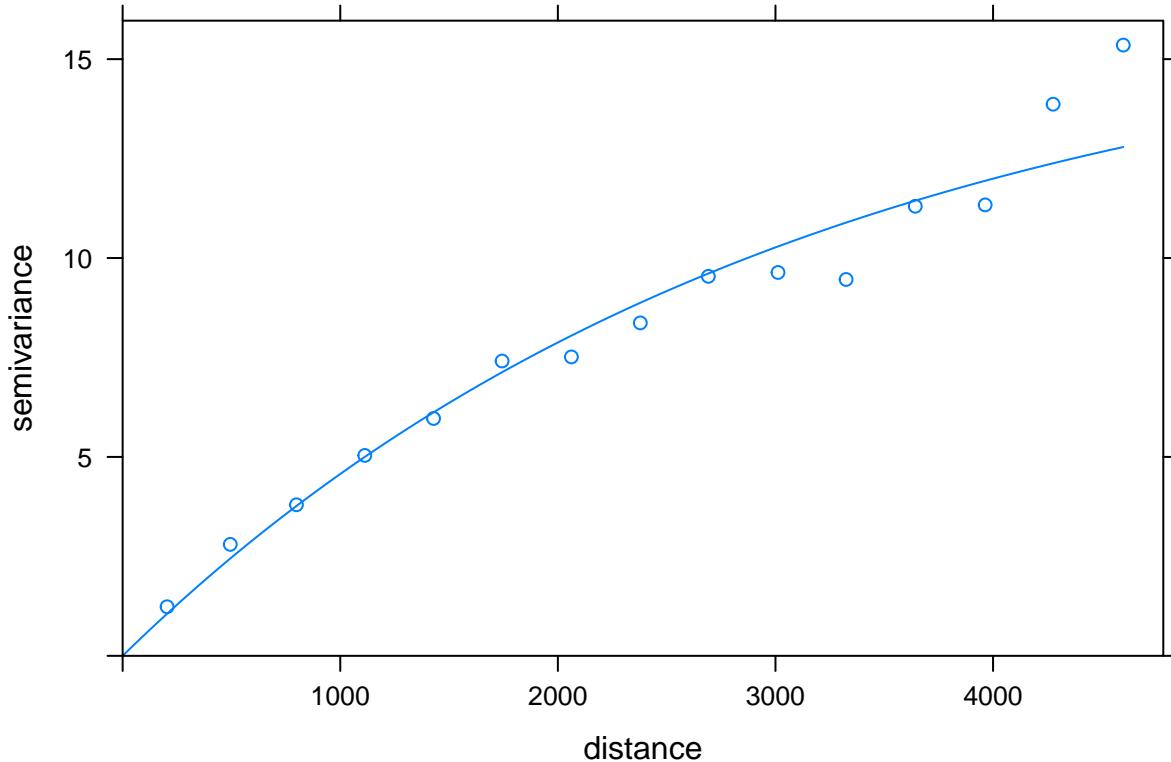
```
plot(vario, model=model_exp)
```



```
fitted_exp <- fit.variogram(vario, model_exp)
fitted_exp
```

```
##   model    psill     range
## 1   Exp 16.50715 3083.864
```

```
plot(vario, model=fitted_exp)
```

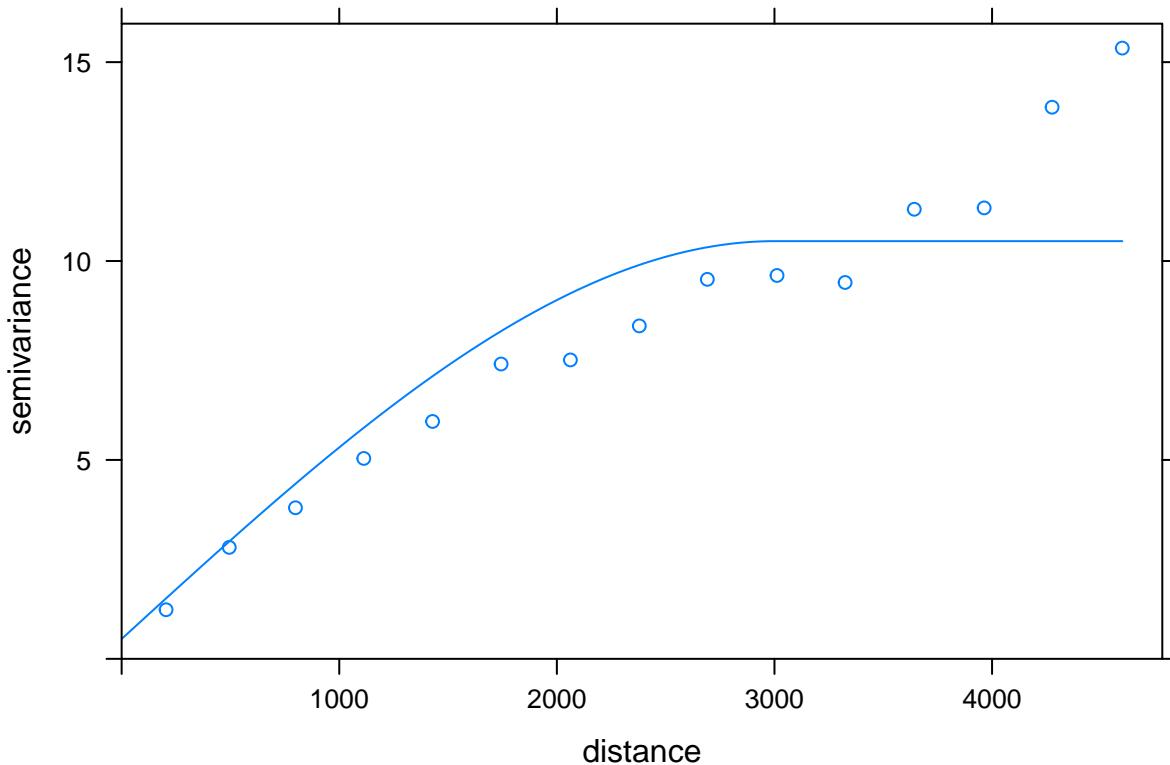


### 7.5.7 Modelowanie izotropowe | Modele złożone I

```
vario <- variogram(temp~1, punkty)
model_zl1 <- vgm(psill=10, model = 'Sph', range = 3000, nugget = 0.5)
model_zl1
```

```
##   model psill range
## 1   Nug    0.5     0
## 2   Sph   10.0   3000
```

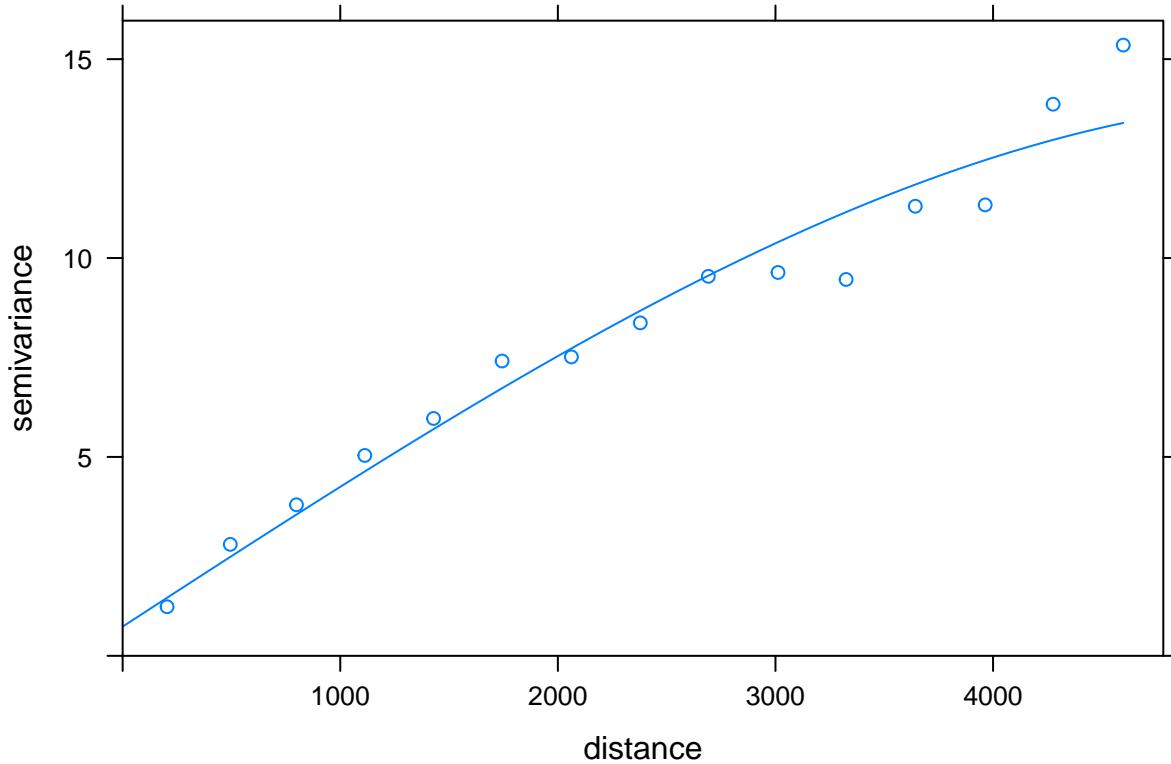
```
plot(vario, model=model_zl1)
```



```
fitted_z11 <- fit.variogram(vario, model_z11)
fitted_z11
```

```
##   model      psill     range
## 1   Nug  0.7346354  0.000
## 2   Sph 13.2621876 5602.027
```

```
plot(vario, model=fitted_z11)
```

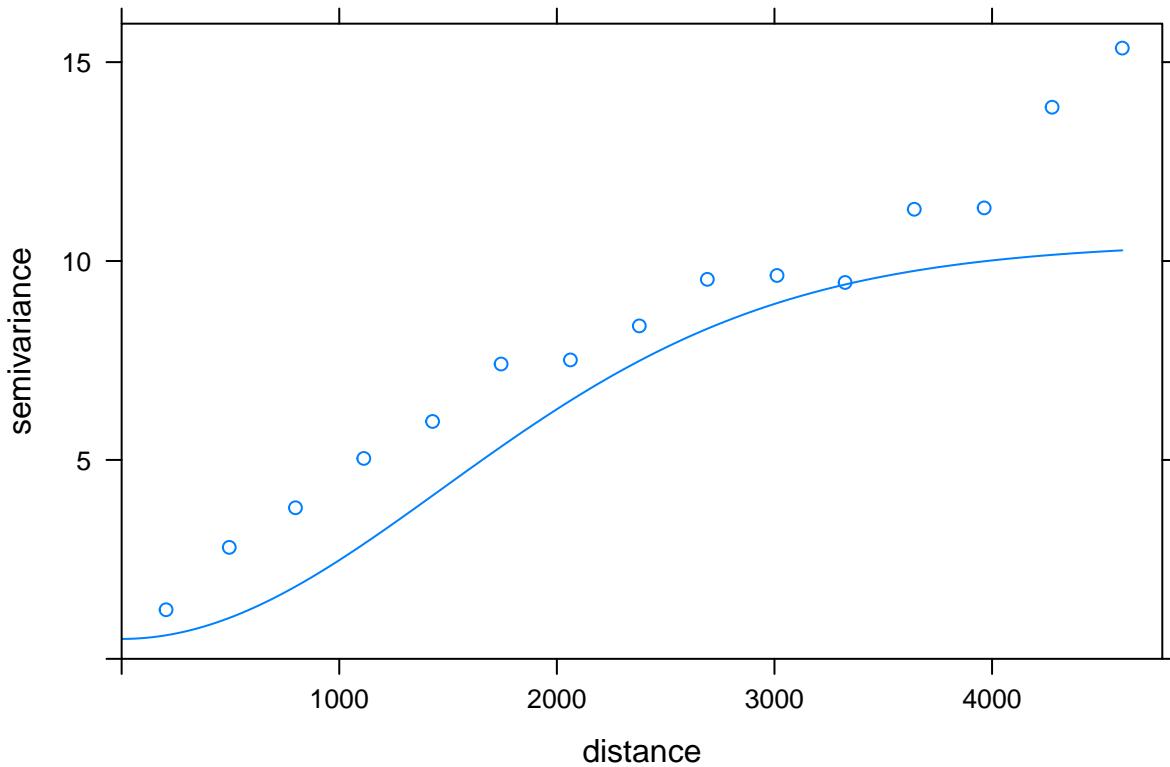


### 7.5.8 Modelowanie izotropowe | Modele złożone II

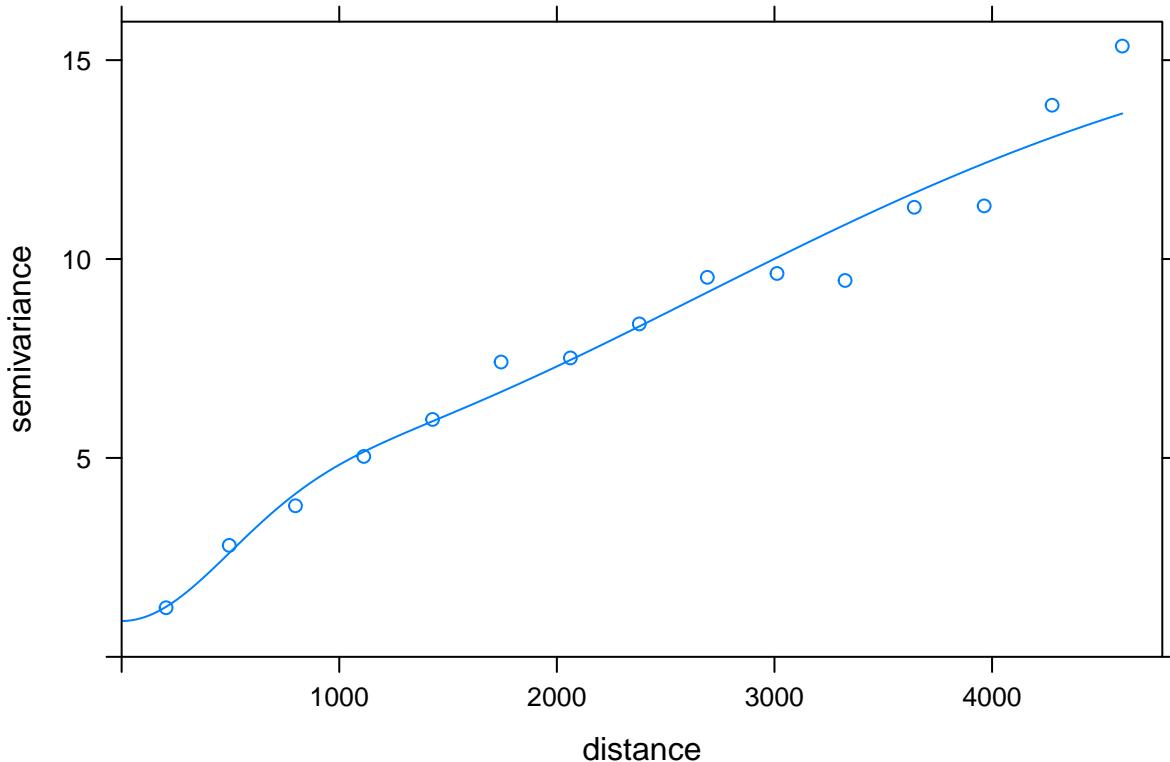
```
vario <- variogram(temp~1, punkty)
model_zl2 <- vgm(2, 'Gau', 3000, add.to = vgm(8, model = 'Gau', range = 2000, nugget = 0.5))
model_zl2

##   model psill range
## 1   Nug    0.5     0
## 2   Gau    8.0   2000
## 3   Gau    2.0   3000
```

```
plot(vario, model=model_zl2)
```



```
fitted_zl2 <- fit.variogram(vario, model_zl2)
plot(vario, model=fitted_zl2)
```

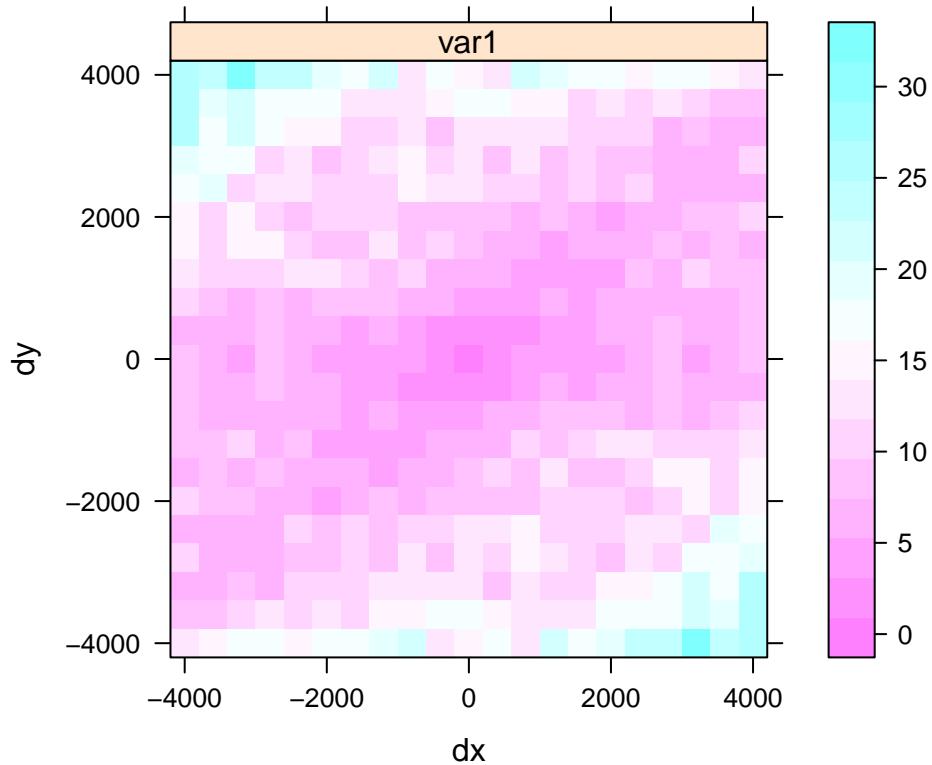


## 7.6 Modelowanie anizotropowe

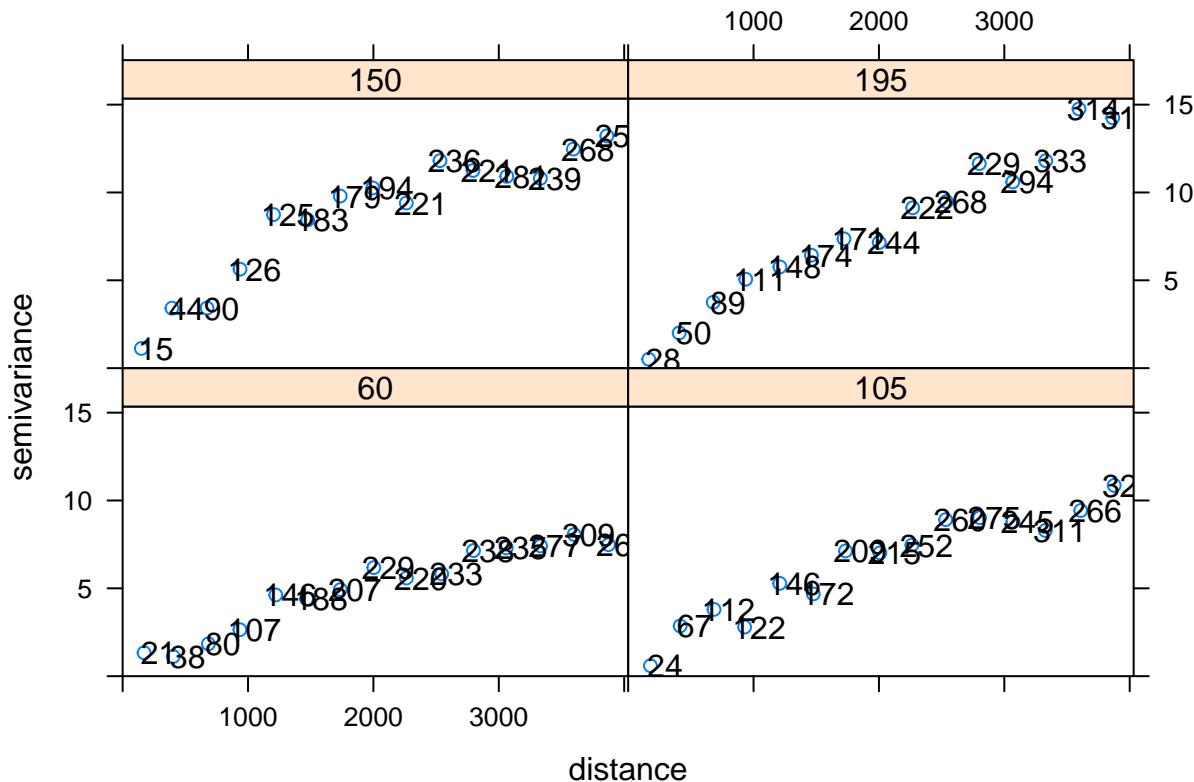
### 7.6.1 Anizotropia

- Uwzględnienie anizotropii wymaga zamiany parametru zasięgu na trzy inne parametry:
- Zasięg w dominującym kierunku
- Kąt określający dominujący kierunek
- Proporcję anizotropii, czyli relację pomiędzy zasięgiem w dominującym kierunku a zasięgiem w przeciwnym kierunku

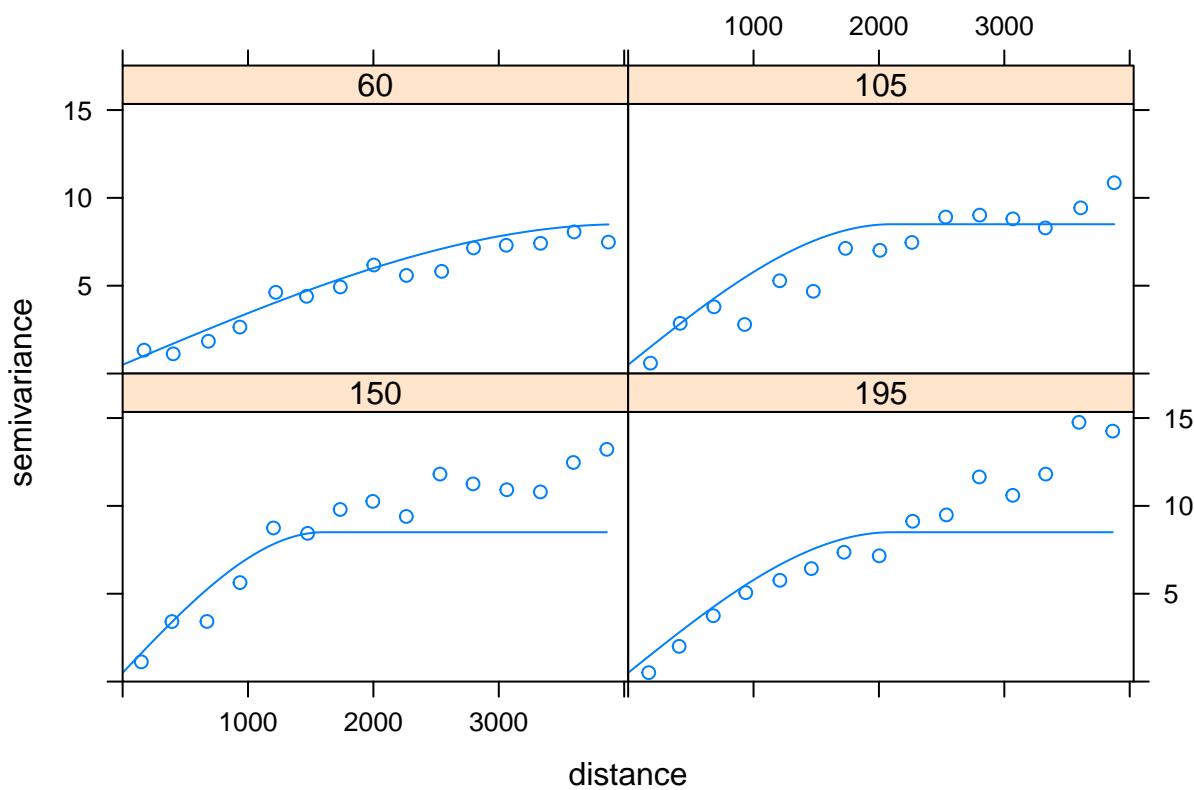
```
vario_map <- variogram(temp~1, punkty, cutoff=4000, width=400, map=TRUE)
plot(vario_map)
```



```
vario_kier <- variogram(temp~1, punkty, alpha = c(60, 105, 150, 195), cutoff=4000)
plot(vario_kier, plot.numbers=TRUE)
```



```
vario_kier_fit <- vgm(psill=8, model='Sph', range=4000, nugget=0.5, anis = c(60, .4))
plot(vario_kier, vario_kier_fit, as.table=TRUE)
```



# Chapter 8

## Estymacje jednozmienne

```
library('geostatbook')
data(punkty)
data(siatka)
```

### 8.1 Kriging

#### 8.1.1 Kriging | Interpolacja geostatystyczna

- Zaproponowana w latach 50. przez Daniela Krige
- Istnieje wiele rodzajów krigingu
- Główna zasada mówi, że prognoza w danej lokalizacji jest kombinacją obokległych obserwacji
- Waga nadawana każdej z obserwacji jest zależna od stopnia (przestrzennej) korelacji - stąd też bierze się istotna rola semiwariogramów

#### 8.1.2 Rodzaje krigingu

- Kriging prosty (ang. *Simple kriging*)
- Kriging zwykły (ang. *Ordinary kriging*)
- Kriging z trendem (ang. *Kriging with a trend*)
- Kriging danych kodowanych (ang. *Indicator kriging*)
- Kriging stratyfikowany (ang. *Kriging within strata – KWS*)
- Kriging prosty ze zmiennymi średnimi lokalnymi (ang. *Simple kriging with varying local means - SKlm*)
- Kriging z zewnętrznym trendem/Uniwersalny kriging (ang. *Kriging with an external trend/Universal kriging*)
- Kokriging (ang. *Co-kriging*)
- Inne

### 8.2 Kriging prosty

#### 8.2.1 Kriging prosty (ang. *Simple kriging*)

- Zakłada, że średnia jest znana i stała na całym obszarze

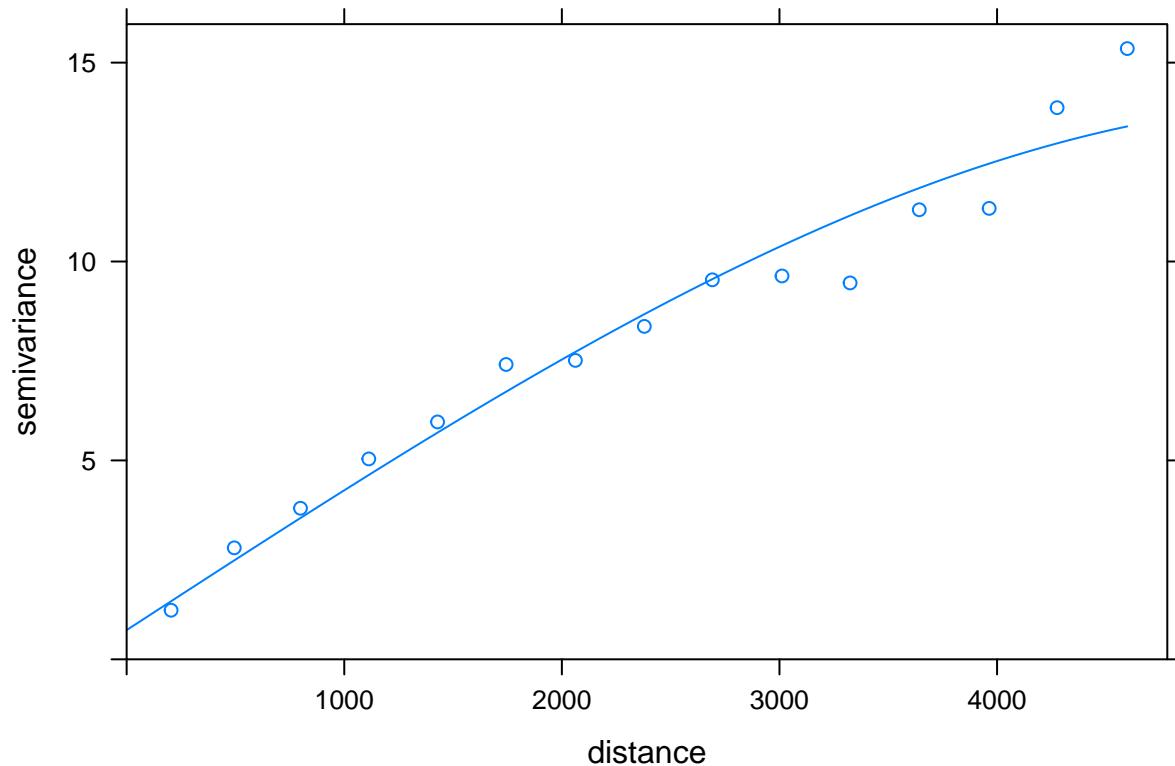
```

vario <- variogram(temp~1, punkty)
model <- vgm(10, model = 'Sph', range = 4000, nugget = 0.5)
model

##   model psill range
## 1   Nug   0.5     0
## 2   Sph  10.0  4000

fitted <- fit.variogram(vario, model)
plot(vario, model=fitted)

```



```
sk <- krige(temp~1, punkty, siatka, model=fitted, beta=15.324)
```

```
## [using simple kriging]
```

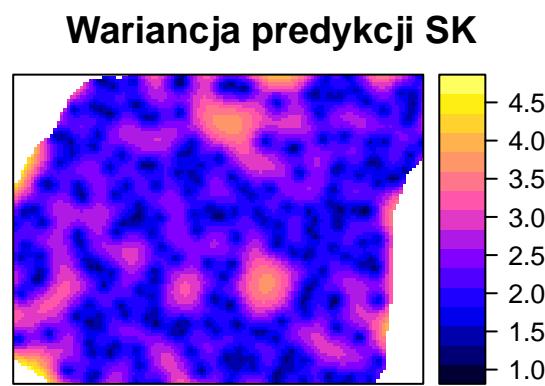
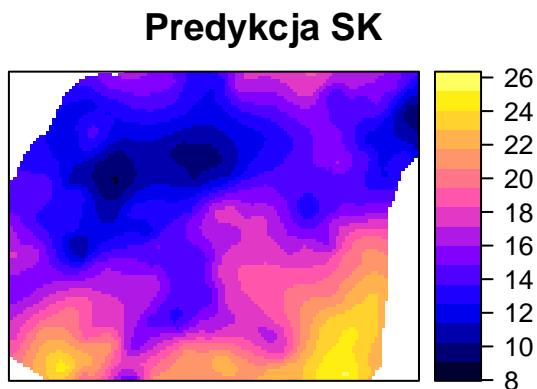
```
summary(sk)
```

```

## Object of class SpatialPixelsDataFrame
## Coordinates:
##      min     max
## x 745586.7 756926.7
## y 712661.2 721211.2
## Is projected: TRUE
## proj4string :
## [+init=epsg:2180 +proj=tmerc +lat_0=0 +lon_0=19 +k=0.9993
## +x_0=500000 +y_0=-5300000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0

```

```
## +units=m +no_defs]  
## Number of points: 10993  
## Grid attributes:  
##   cellcentre.offset cellsize cells.dim  
##   s1           745586.7    90     127  
##   s2           712661.2    90      96  
## Data attributes:  
##   vari1.pred      vari1.var  
##   Min.    : 9.083  Min.    :1.062  
##   1st Qu.:13.208  1st Qu.:1.875  
##   Median  :15.224  Median  :2.183  
##   Mean    :15.862  Mean    :2.253  
##   3rd Qu.:18.081  3rd Qu.:2.574  
##   Max.    :25.218  Max.    :4.615  
  
spplot(sk, 'vari1.pred')  
spplot(sk, 'vari1.var')
```



## 8.3 Kriging zwykły

### 8.3.1 Kriging zwykły (ang. *Ordinary kriging*)

- Średnia traktowana jest jako nieznana
- Uwzględnia lokalne fluktuacje średniej poprzez stosowanie ruchomego okna

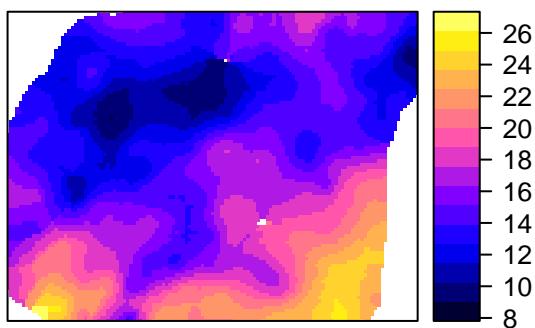
```
ok <- krige(temp~1, punkty, siatka, model=fitted, maxdist=1000)
```

```
## [using ordinary kriging]
```

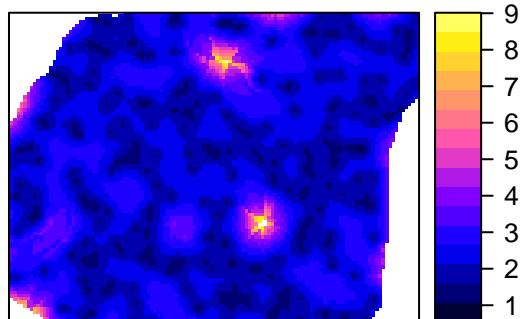
```
# ok <- krige(temp~1, punkty, siatka, model=fitted, nmax=30)
```

```
spplot(ok, 'var1.pred')
spplot(ok, 'var1.var')
```

**Predykcja OK**



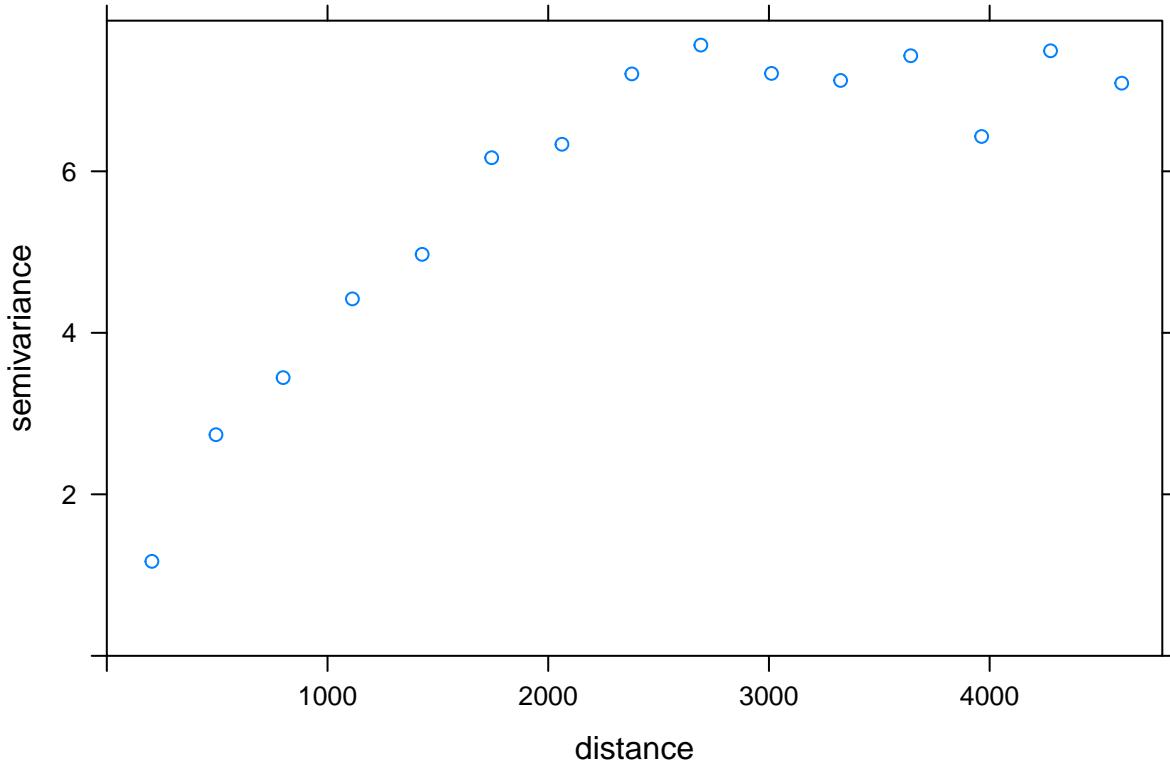
**Wariancja predykcji OK**



## 8.4 Kriging z trendem

### 8.4.1 Kriging z trendem (ang. *Kriging with a trend*)

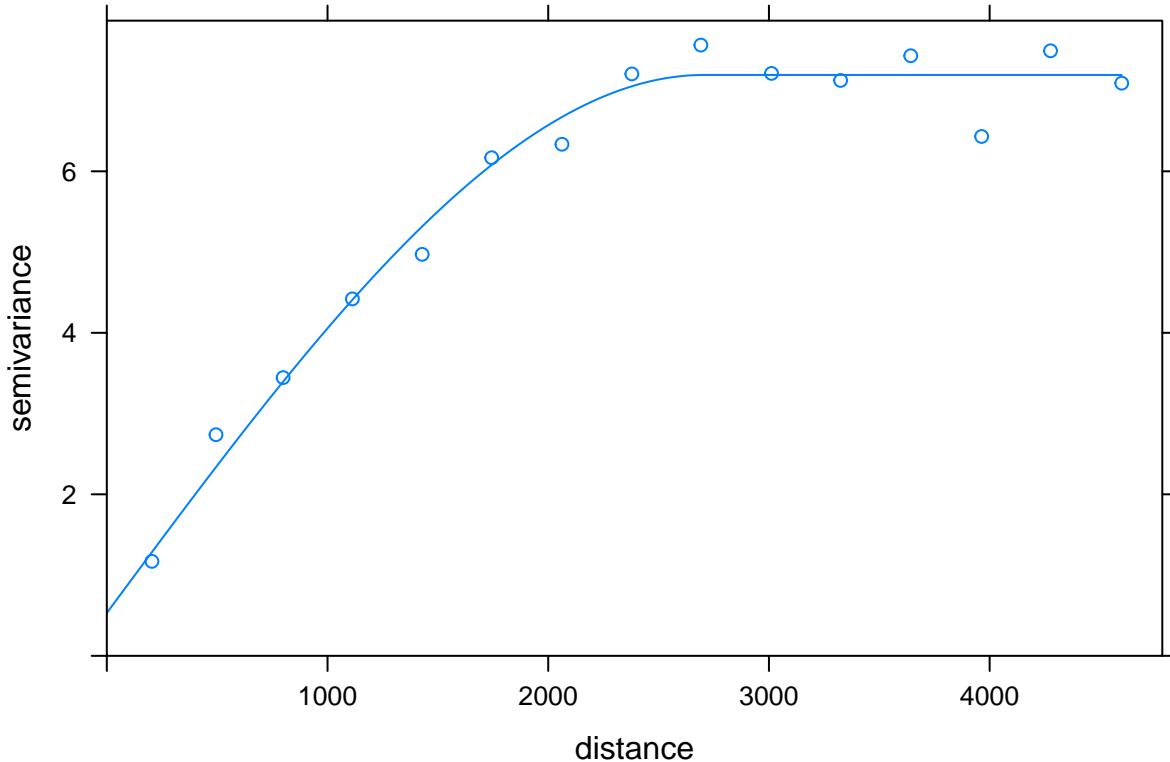
```
vario_kzt <- variogram(temp~x+y, data=punkty)
plot(vario_kzt)
```



```
model_kzt <- vgm(psill = 5, model = 'Sph', range = 2500, nugget = 1)
fitted_kzt <- fit.variogram(vario_kzt, model_kzt)
fitted_kzt
```

```
##   model      psill      range
## 1   Nug 0.5308819    0.000
## 2   Sph 6.6620981 2705.967
```

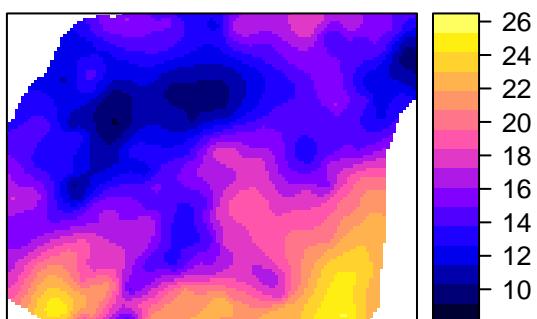
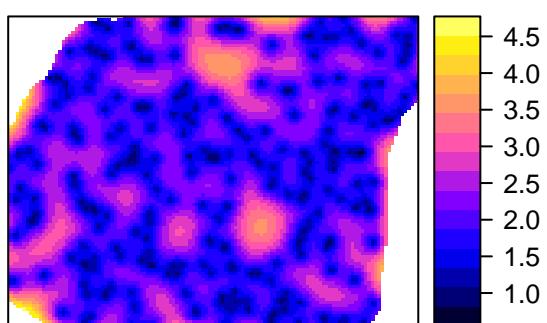
```
plot(vario_kzt, fitted_kzt)
```



```
punkty@data <- cbind(punkty@data, as.data.frame(coordinates(punkty)))
kzt <- krige(temp~x+y, punkty, siatka, model=fitted_kzt)
```

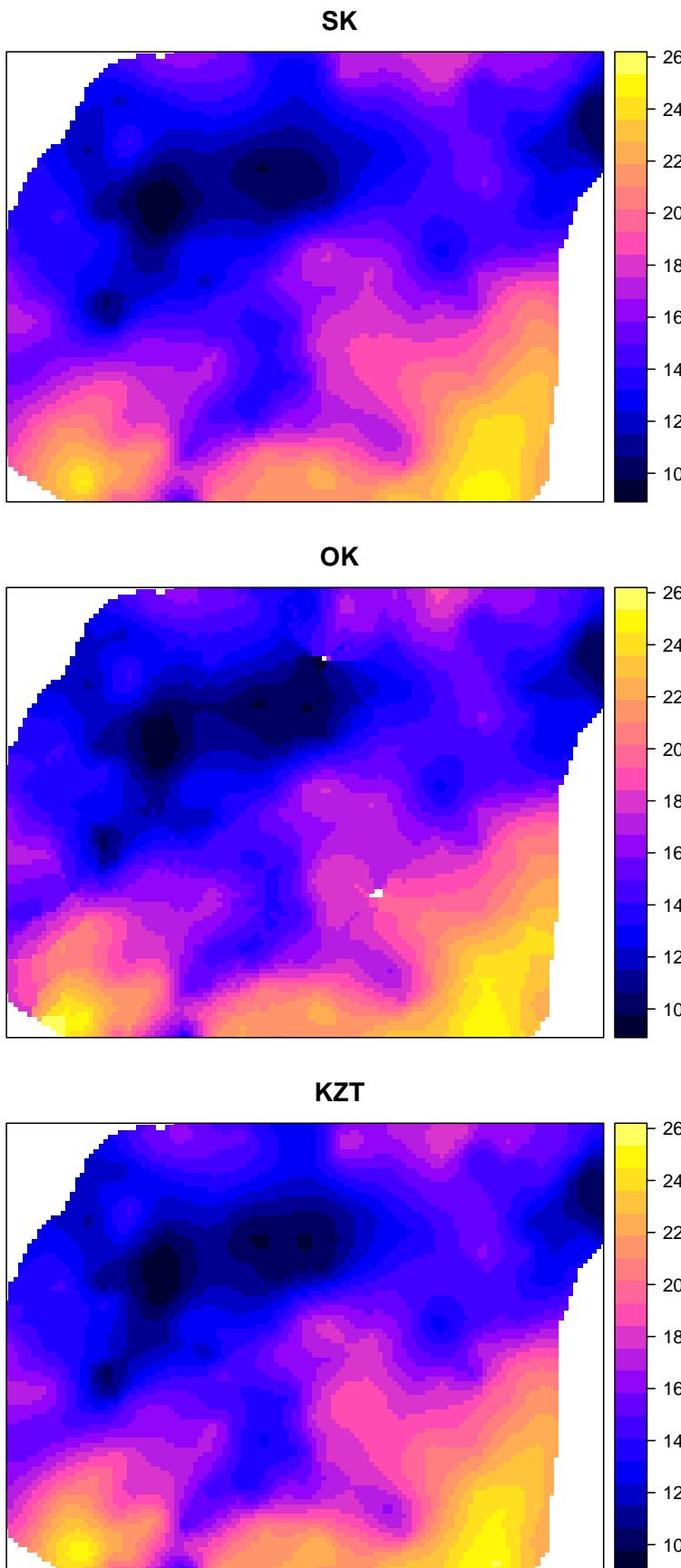
```
## [using universal kriging]
```

```
spplot(kzt, 'var1.pred')
spplot(kzt, 'var1.var')
```

**Preidykcja OK****Wariancja preidykcji OK**



## 8.5 Porównanie wyników SK, OK i KZT





# Chapter 9

## Estymacja lokalnego rozkładu prawdopodobieństwa

```
library('geostatbook')
data(punkty)
data(siatka)
```

### 9.1 Kriging danych kodowanych

#### 9.1.1 Kriging danych kodowanych (ang. *Indicator kriging*)

- Jest to metoda krigingu oparta o dane kategoryzowane lub też dane przetworzone z postaci ciągiej do binarnej
- Zazwyczaj używana jest do oszacowania prawdopodobieństwa przekroczenia zdefiniowanej wartości progowej
- Może być też używana do szacowania wartości z całego rozkładu
- Wartości danych wykorzystywane do krigingu danych kodowanych są określone jako 0 lub 1
- Powyzsze kategorie reprezentują czy wartość danej zmiennej jest powyżej czy poniżej określonego progu

#### 9.1.2 Kriging danych kodowanych (ang. *Indicator kriging*)| Wady i zalety

- Zalety:
  - Możliwość zastosowania, gdy nie interesuje nas konkretna wartość, ale znalezienie obszarów o wartości przekraczającej dany poziom
  - Nie jest istotny kształt rozkładu
- Wady:
  - Potencjalnie trudne do modelowania semiwariogramy (szczególnie skrajnych przedziałów)
  - Czasochłonność/pracochłonność

## 9.2 Kriging danych kodowanych | Przykłady

### 9.2.1 Binaryzacja danych

```
summary(punkty$temp)

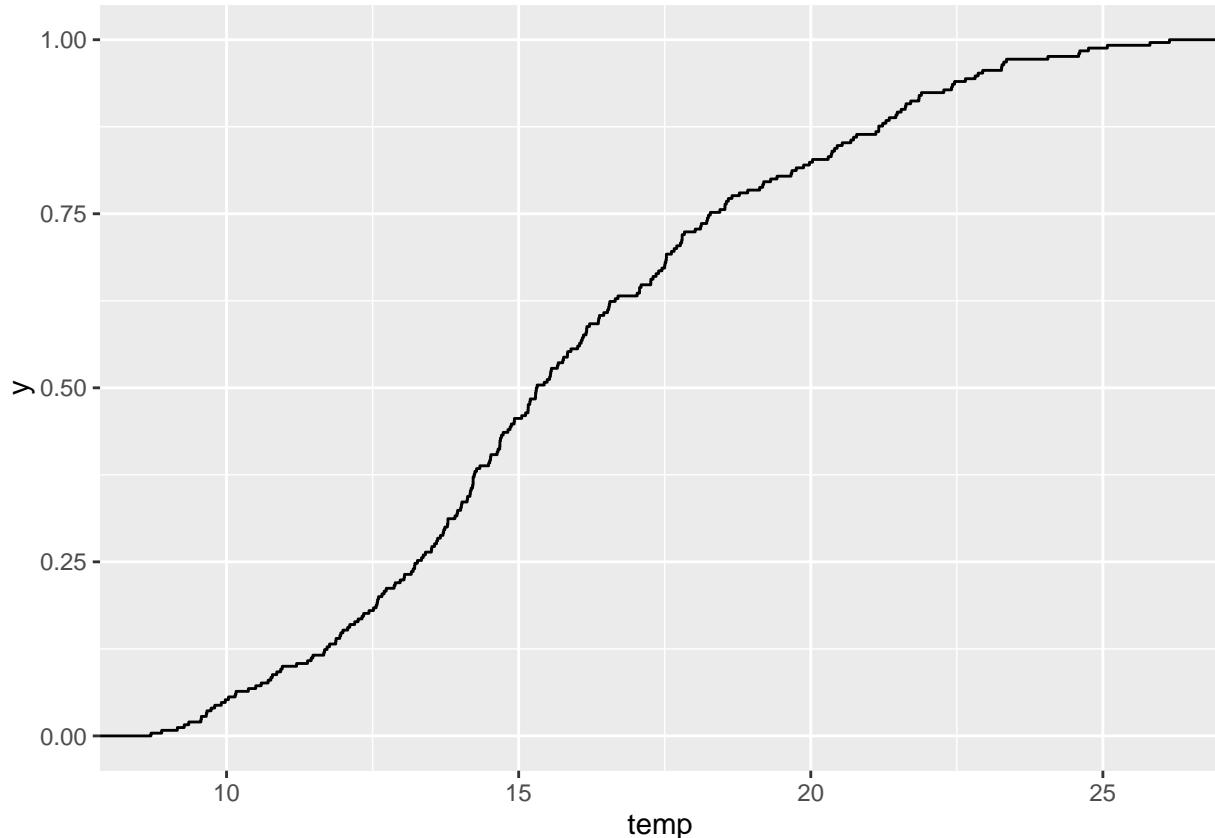
##      Min. 1st Qu. Median      Mean 3rd Qu.      Max.
##    8.706 13.280 15.310 15.950 18.270 26.140
```

```
punkty$temp_ind <- punkty$temp > 20
summary(punkty$temp_ind)
```

```
##      Mode   FALSE    TRUE     NA's
## logical     206      44       0
```

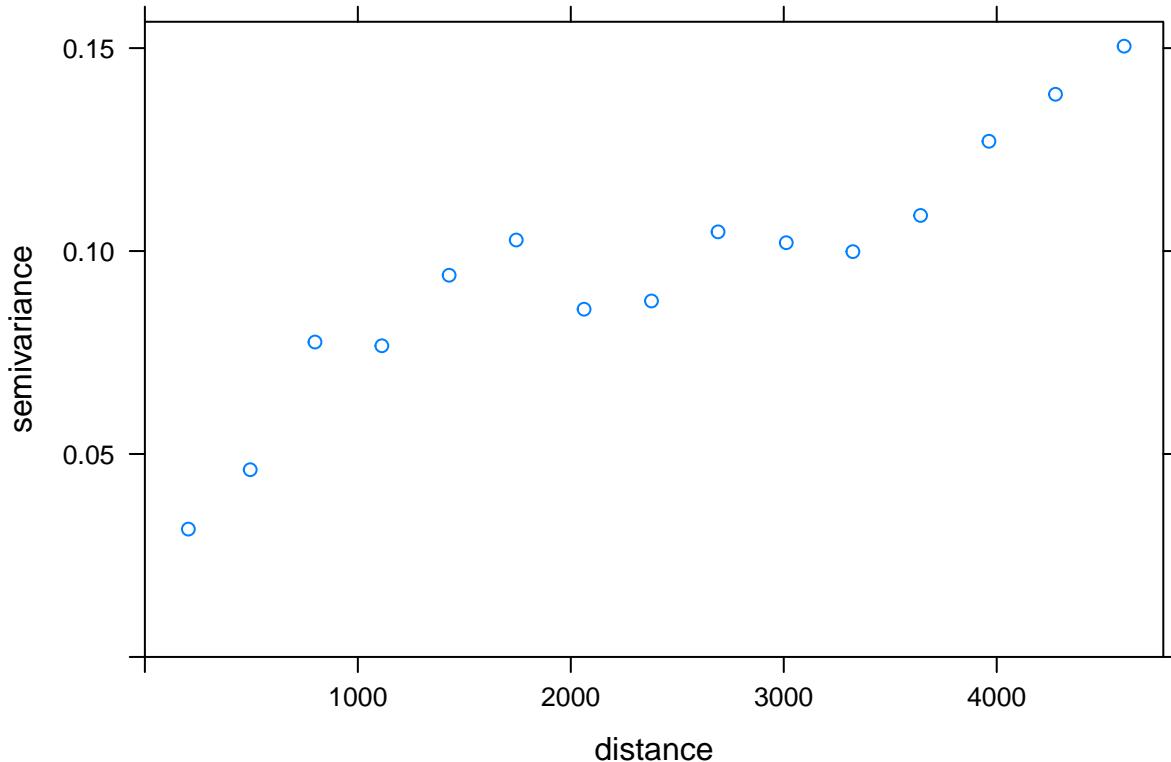
### 9.2.2 Eksploracja danych

```
ggplot(punkty@data, aes(temp)) + stat_ecdf()
```



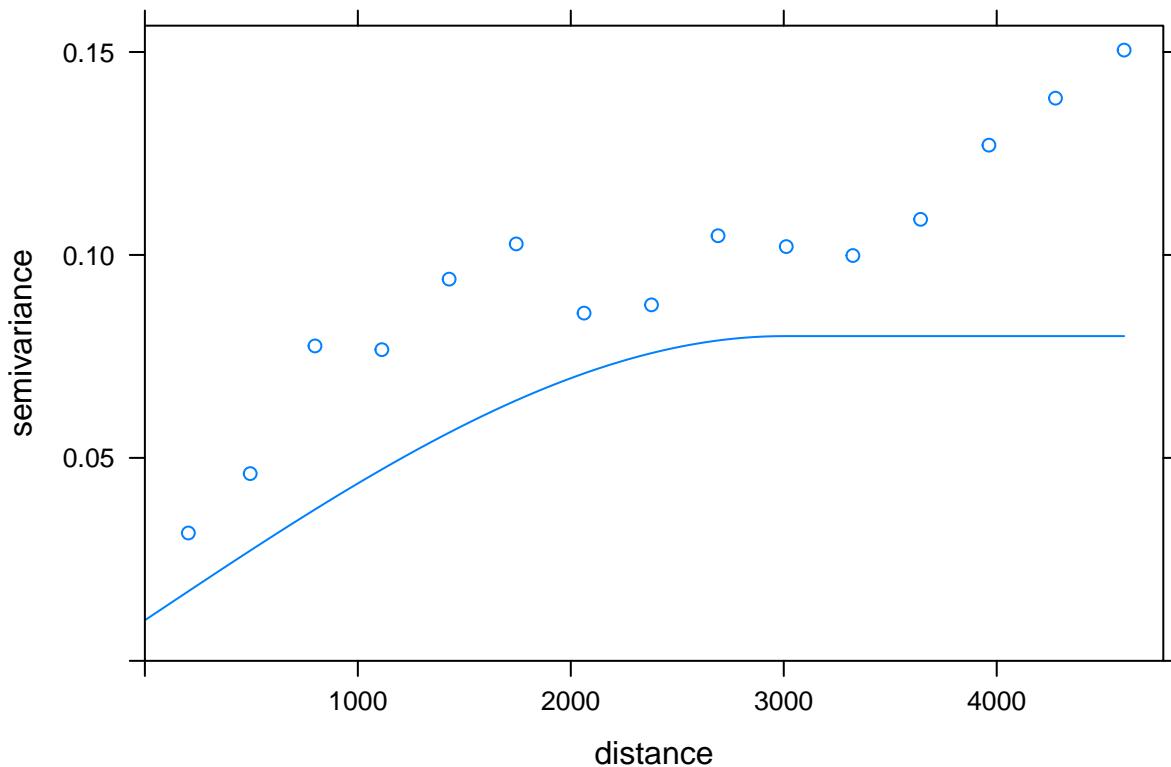
### 9.2.3 Kriging danych kodowanych (ang. *Indicator kriging*) | Semiwariogram

```
vario_ind <- variogram(temp_ind~1, punkty)
plot(vario_ind)
```



### 9.2.4 Kriging danych kodowanych (ang. *Indicator kriging*) | Modelowanie

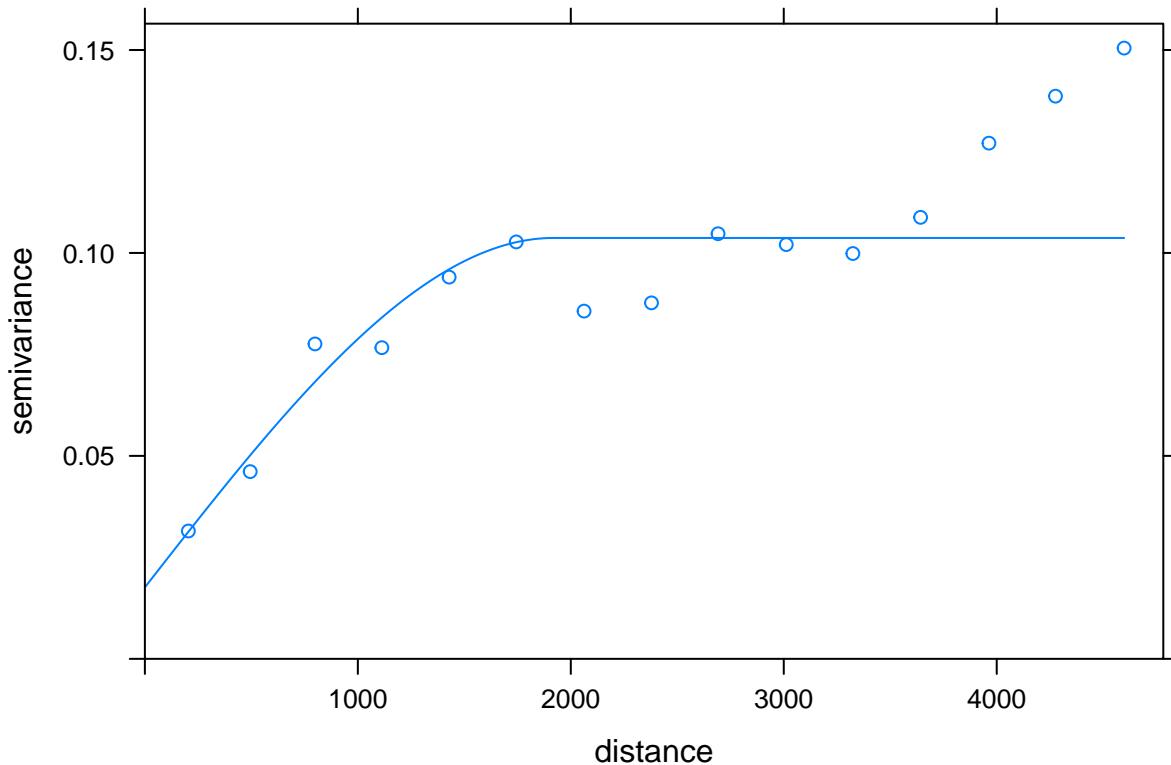
```
model_ind <- vgm(0.07, model = 'Sph', range = 3000, nugget = 0.01)
plot(vario_ind, model=model_ind)
```



```
fitted_ind <- fit.variogram(vario_ind, model_ind)  
fitted_ind
```

```
##   model      psill     range  
## 1   Nug 0.01761568 0.000  
## 2   Sph 0.08607731 1919.469
```

```
plot(vario_ind, model=fitted_ind)
```



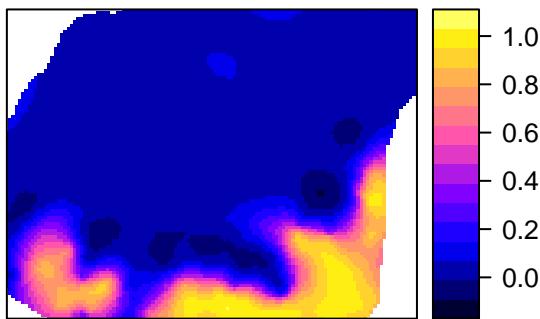
### 9.2.5 Kriging danych kodowanych (ang. *Indicator kriging*) | Interpolacja

```
ik <- krige(temp_ind~1, punkty, siatka, model=fitted_ind)
```

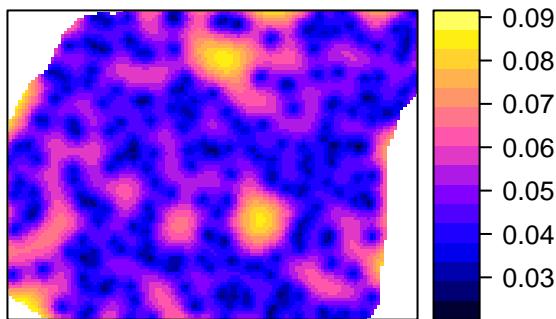
```
## [using ordinary kriging]
```

```
spplot(ik, 'var1.pred')
spplot(ik, 'var1.var')
```

## Prawdopodobieństwo Temp > 20



## Wariancja predykcji IK



### 9.2.6 Kriging danych kodowanych (ang. *Indicator kriging*)

```

vario_ind20 <- variogram(I(temp<20)~1, punkty)
fitted_ind20 <- fit.variogram(vario_ind20, vgm(0.08, 'Sph', 3500, nugget=0.01))
vario_ind16 <- variogram(I(temp<16)~1, punkty)
fitted_ind16 <- fit.variogram(vario_ind16, vgm(0.18, 'Sph', 3500, nugget=0.03))
vario_ind12 <- variogram(I(temp<12)~1, punkty)
fitted_ind12 <- fit.variogram(vario_ind12, vgm(0.13, 'Sph', 2000, nugget=0.03))

ik20 <- krige(I(temp<20)~1, punkty, siatka, model=fitted_ind20, nmax=30)

```

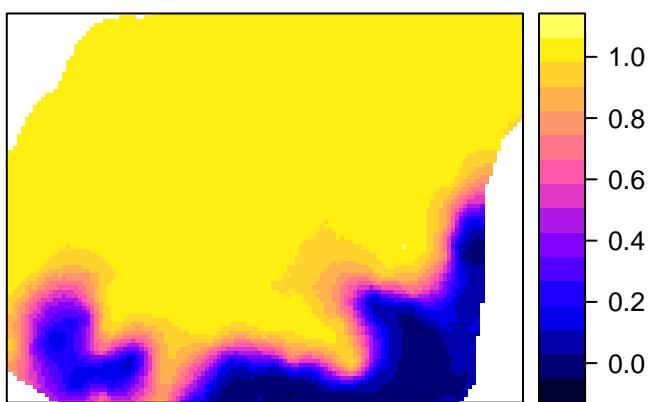
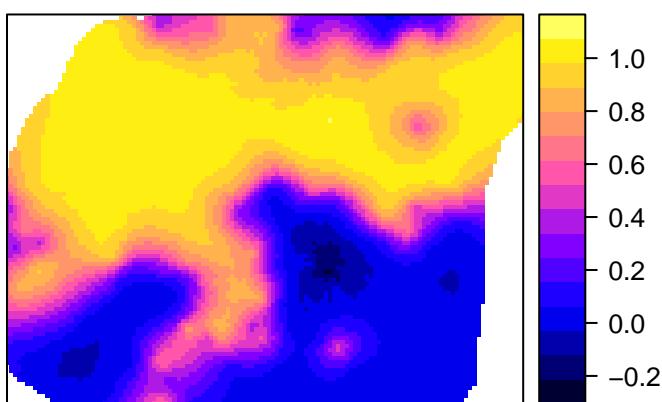
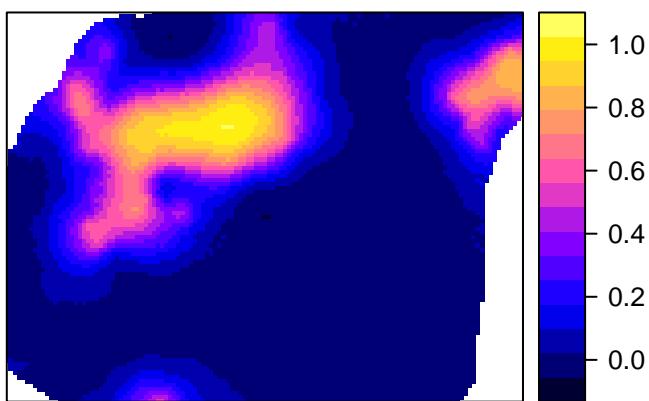
```
## [using ordinary kriging]
```

```
ik16 <- krige(I(temp<16)~1, punkty, siatka, model=fitted_ind16, nmax=30)
```

```
## [using ordinary kriging]
```

```
ik12 <- krige(I(temp<12)~1, punkty, siatka, model=fitted_ind12, nmax=30)
```

```
## [using ordinary kriging]
```

**Prawdopodobie..stwo Temp < 20****Prawdopodobie..stwo Temp < 16****Prawdopodobie..stwo Temp < 12**



# Chapter 10

## Estymacje wielozmienne

```
library('geostatbook')
data(punkty)
data(punkty_ndvi)
data(siatka)
```

### 10.1 Kokriging (prosty i zwykły, SCK i OCK)

#### 10.1.1 Kokriging (ang. *co-kriging*)

- Kokriging pozwala na wykorzystanie dodatkowej zmiennej (ang. *auxiliary variable*), zwanej inaczej kozmienną (ang. *co-variable*), która może być użyta do prognozowania wartości badanej zmiennej w nieopróbowanej lokalizacji
- Zmienna dodatkowa może być pomierzona w tych samych miejscowościach, gdzie badana zmienna, jak też w innych niż badana zmienna
- Możliwa jest też sytuacja, gdy zmienna dodatkowa jest pomierzona w dwóch powyższych przypadkach
- Kokriging wymaga, aby obie zmienne były istotnie ze sobą skorelowane
- Najczęściej kokriging jest stosowany w sytuacji, gdy zmienna dodatkowa jest łatwiejsza (tańsza) do pomierzenia niż zmienna główna
- W efekcie, uzyskany zbiór danych zawiera informacje o badanej zmiennej oraz gęściej opróbowane informacje o zmiennej dodatkowej
- Jeżeli informacje o zmiennej dodatkowej są znane dla całego obszaru wówczas bardziej odpowiednią techniką będzie kriging z zewnętrznym trendem (KED)

#### 10.1.2 Kokriging | Wybór dodatkowej zmiennej

- Wybór zmiennej dodatkowej może opierać się na dwóch kryteriach:
  - Teoretycznym
  - Empirycznym

## 10.2 Krossemiwiariogramy

### 10.2.1 Krossemiwiariogramy

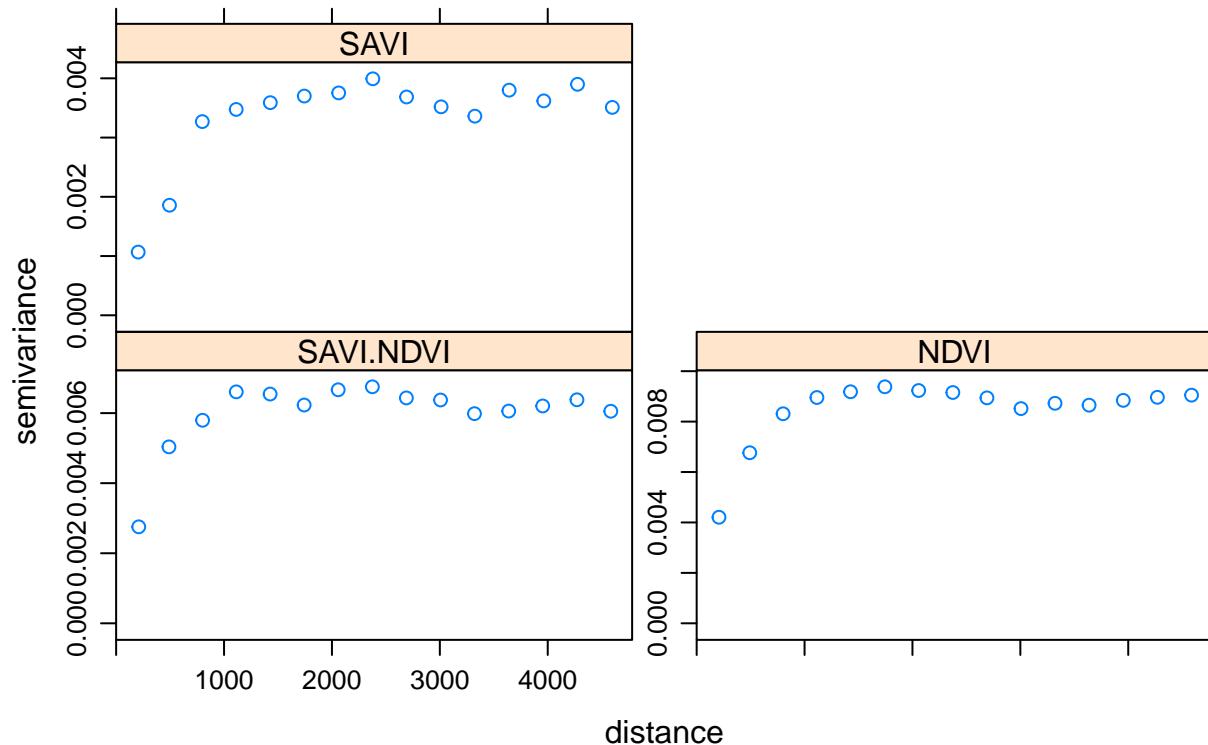
- Krossemiwiariogram jest to wariancja różnicy pomiędzy dwiema zmiennymi w dwóch lokalizacjach
- Wyliczając Krossemiwiariogram otrzymujemy empiryczne semiwatiogramy dla dwóch badanych zmiennych oraz krosvariogram dla kombinacji dwóch zmiennych
- Krossemiwiariogram znajduje swoje zastosowanie w technice zwanej kokrigingiem

### 10.2.2 Krossemiwiariogramy

```
g <- gstat(NULL, id='SAVI', form = savi~1, data = punkty)
g <- gstat(g, id='NDVI', form = ndvi~1, data = punkty_ndvi)
g
```

```
## data:
## SAVI : formula = savi`~`1 ; data dim = 250 x 5
## NDVI : formula = ndvi`~`1 ; data dim = 997 x 1
```

```
v <- variogram(g)
plot(v)
```



## 10.3 Modelowanie krossemiwariorogramów

### 10.3.1 Modelowanie krossemiwariorogramów | `fit.lmc`

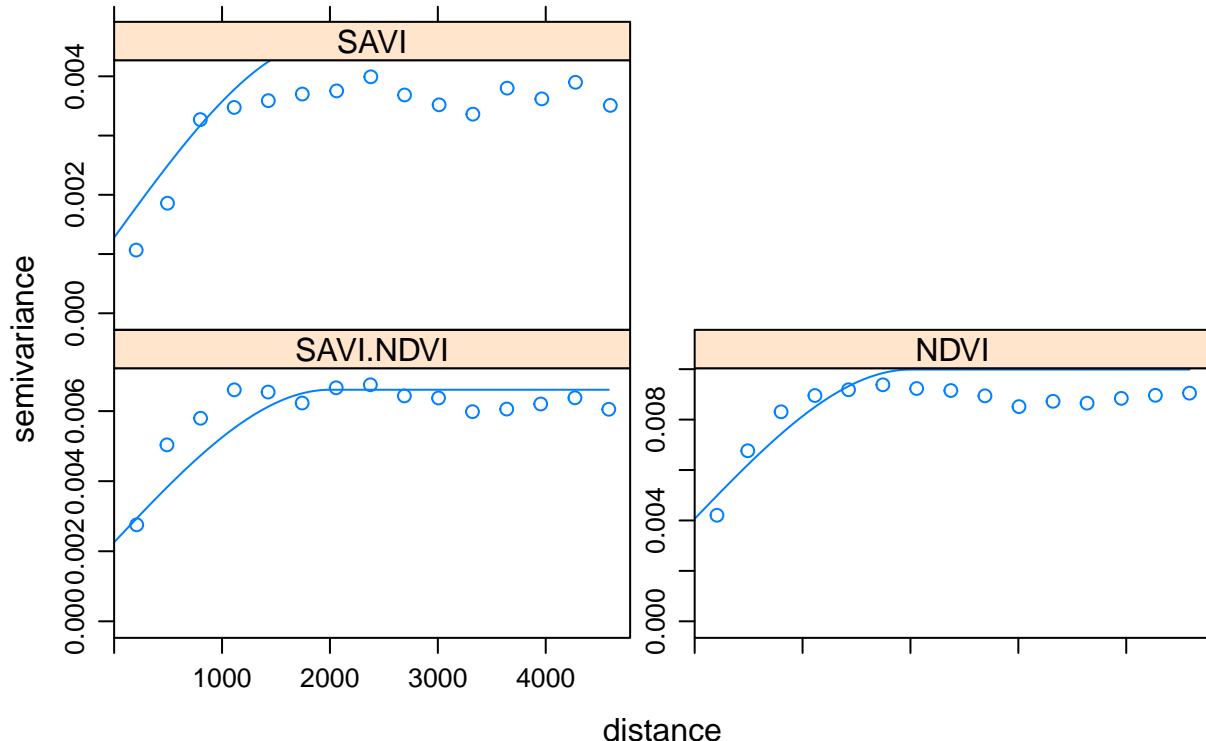
- Funkcja `fit.lmc` dopasowuje liniowy model koregionalizacji do semiwariogramów wielozmienych

### 10.3.2 Modelowanie krossemiwariorogramów

```
g <- gstat(g, model=vgm(0.006, 'Sph', 2000, 0.001), fill.all=TRUE)
g_fit <- fit.lmc(v, g, correct.diagonal = 1.01) #zmienne są bardzo mocno skorelowane
g_fit
```

```
## data:
## SAVI : formula = savi~`~1 ; data dim = 250 x 5
## NDVI : formula = ndvi~`~1 ; data dim = 997 x 1
## variograms:
##          model      psill range
## SAVI[1]    Nug 0.001278509     0
## SAVI[2]    Sph 0.003338946   2000
## NDVI[1]    Nug 0.004065525     0
## NDVI[2]    Sph 0.005918453   2000
## SAVI.NDVI[1] Nug 0.002257298     0
## SAVI.NDVI[2] Sph 0.004352821   2000
```

```
plot(v, g_fit)
```



## 10.4 Kokriging

### 10.4.1 Kokriging

```

ck <- predict(g_fit, siatka)

## Linear Model of Coregionalization found. Good.
## [using ordinary cokriging]

summary(ck)

## Object of class SpatialPixelsDataFrame
## Coordinates:
##      min     max
## x 745586.7 756926.7
## y 712661.2 721211.2
## Is projected: TRUE
## proj4string :
## [+init=epsg:2180 +proj=tmerc +lat_0=0 +lon_0=19 +k=0.9993
## +x_0=500000 +y_0=-5300000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0
## +units=m +no_defs]
## Number of points: 10993
## Grid attributes:
##   cellcentre.offset cellsizex cellsizex
## s1          745586.7      90      127
## s2          712661.2      90       96
## Data attributes:
##   SAVI.pred      SAVI.var      NDVI.pred      NDVI.var
## Min.    :0.1152  Min.    :0.001615  Min.    :0.2336  Min.    :0.004750
## 1st Qu.:0.3001  1st Qu.:0.001958  1st Qu.:0.4715  1st Qu.:0.005188
## Median  :0.3255  Median  :0.002046  Median  :0.5056  Median  :0.005336
## Mean    :0.3205  Mean    :0.002060  Mean    :0.4996  Mean    :0.005365
## 3rd Qu.:0.3480  3rd Qu.:0.002149  3rd Qu.:0.5365  3rd Qu.:0.005513
## Max.    :0.4197  Max.    :0.002862  Max.    :0.6280  Max.    :0.006780
## cov.SAVI.NDVI
## Min.    :0.002658
## 1st Qu.:0.003045
## Median  :0.003157
## Mean    :0.003178
## 3rd Qu.:0.003289
## Max.    :0.004232

spplot(ck, 'SAVI.pred')
spplot(ck, 'SAVI.var')

```

# Chapter 11

## Wykorzystanie do estymacji danych uzupełniających

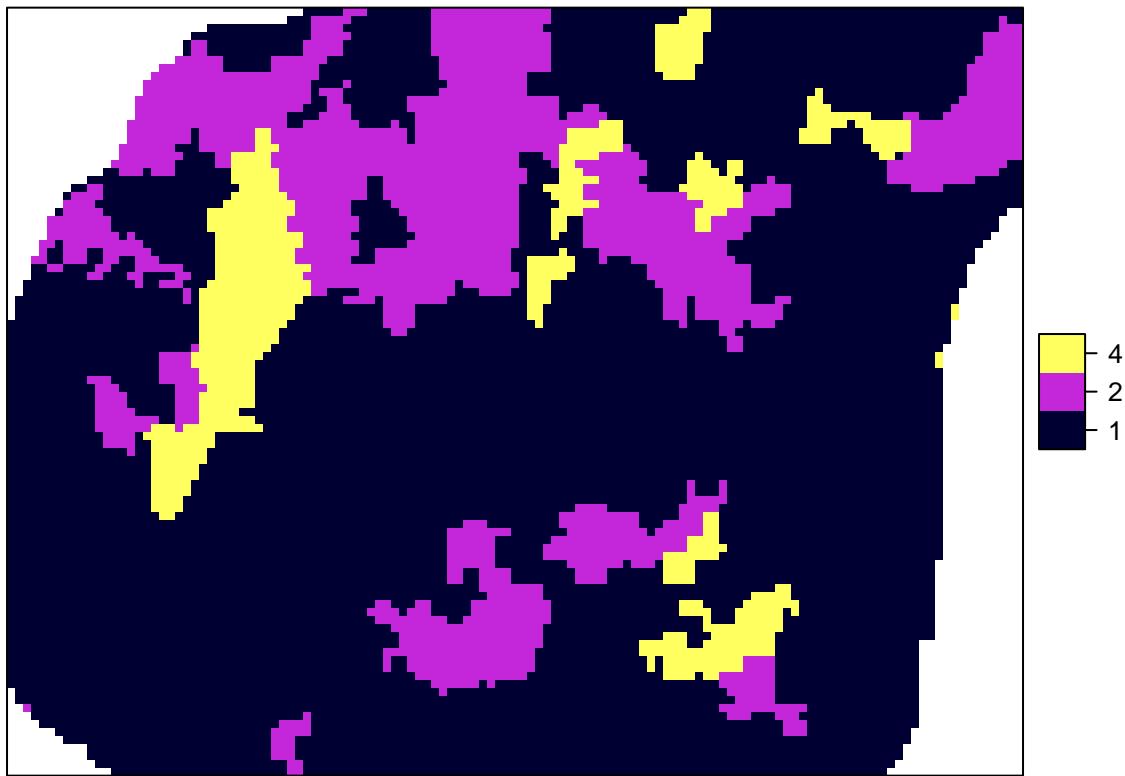
```
library('geostatbook')
data(punkty)
data(siatka)
```

### 11.1 Kriging stratyfikowany

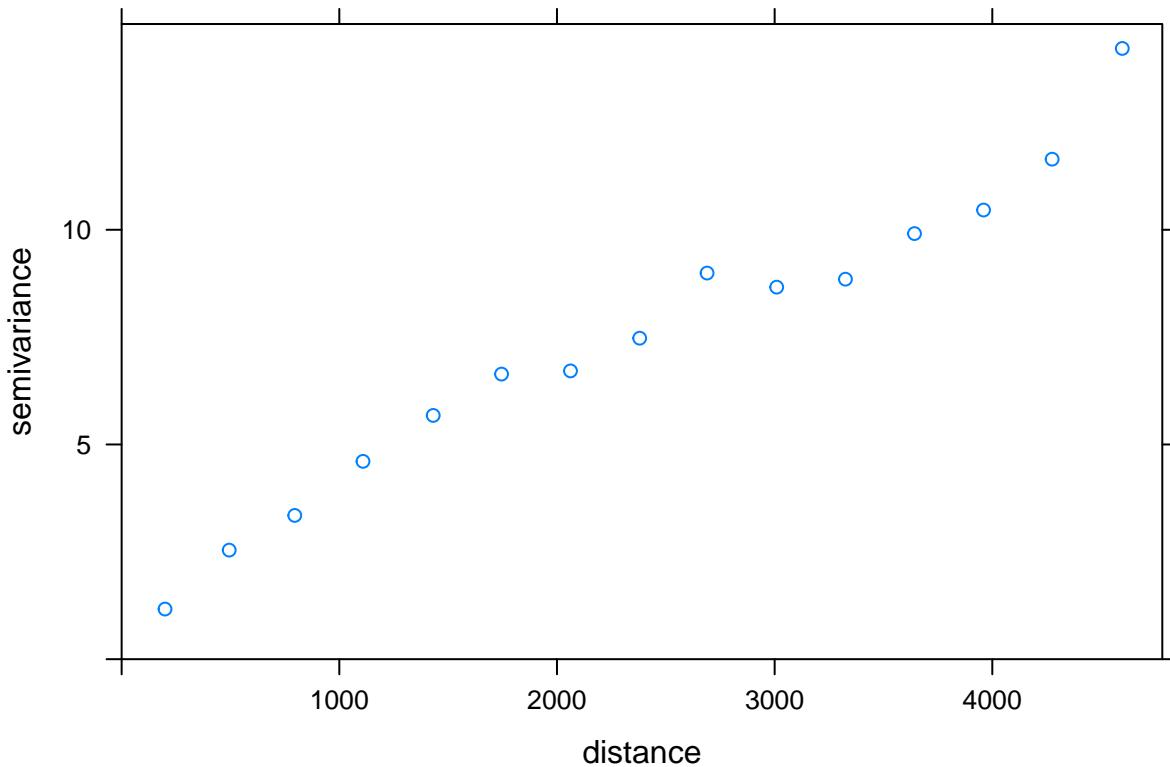
#### 11.1.1 Kriging stratyfikowany (ang. *Kriging within strata*)

- Zakłada on, że zmienność badanego zjawiska zależy od zmiennej jakościowej (kategoryzowanej)
- Przykładowo, zróżnicowanie badanej zmiennej jest różne w zależności od pokrycia terenu
- Kriging stratyfikowany wymaga posiadania danych zmiennej jakościowej (kategoryzowanej) na całym badanym obszarze

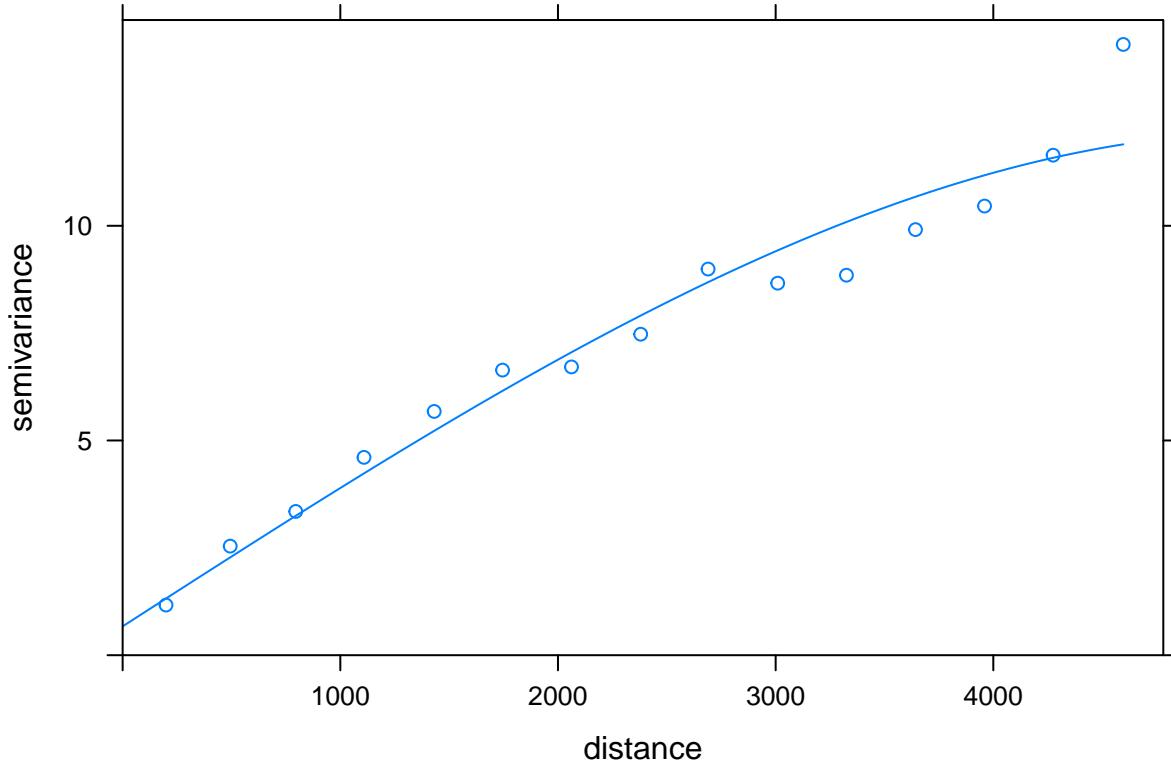
```
siatka$clc <- as.factor(siatka$clc)
spplot(siatka, 'clc')
```



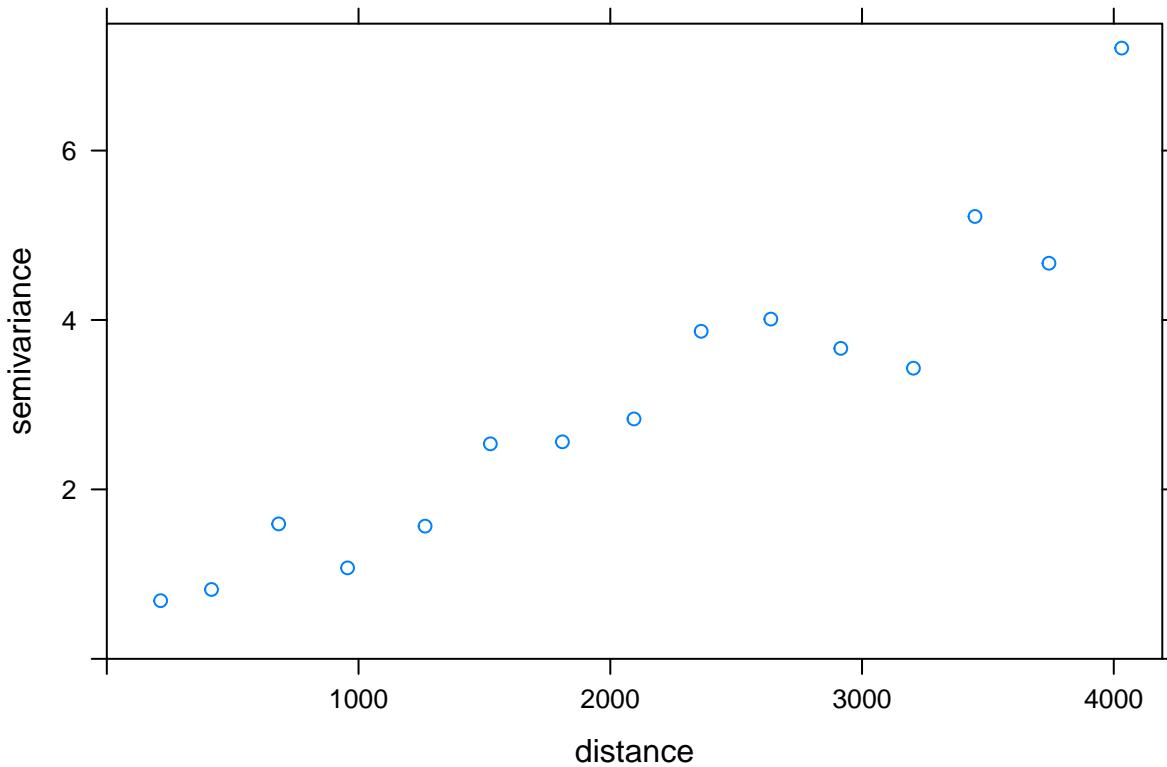
```
vario_kws1 <- variogram(temp~1, punkty[punkty$clc==1, ])
plot(vario_kws1)
```



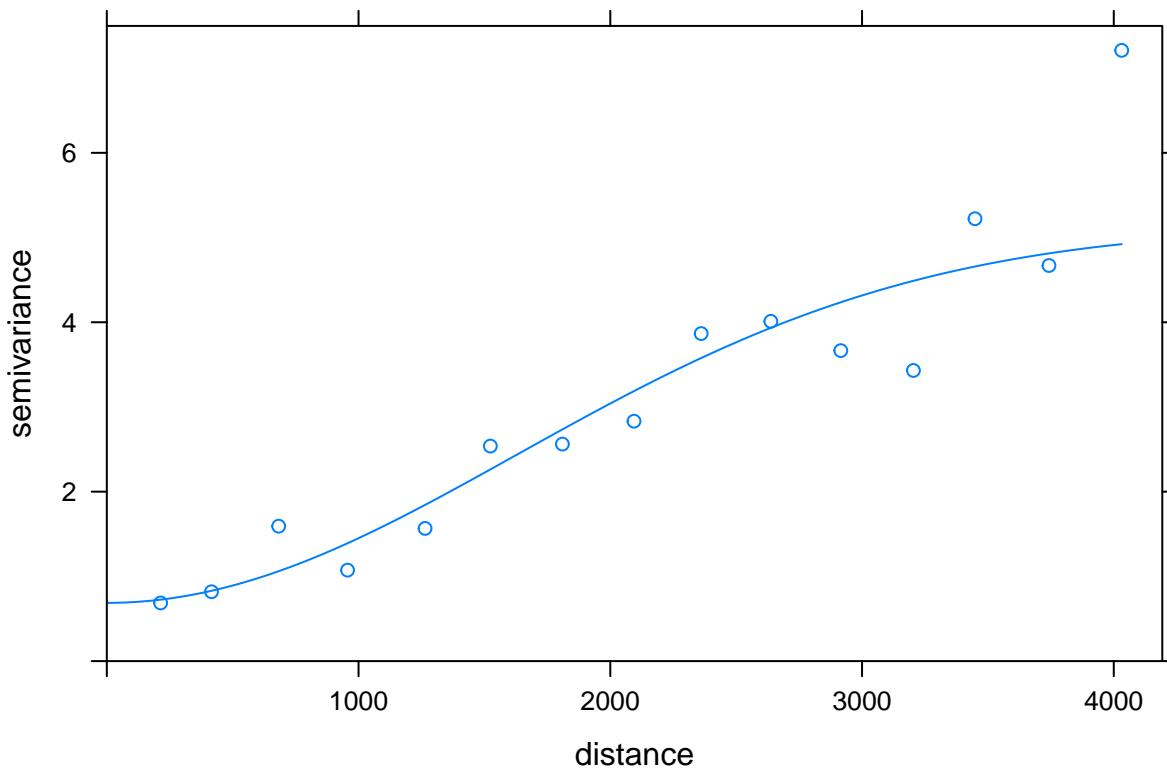
```
fitted_kws1 <- fit.variogram(vario_kws1, vgm(10, model = 'Sph', range = 4500, nugget = 0.5))
plot(vario_kws1, fitted_kws1)
```



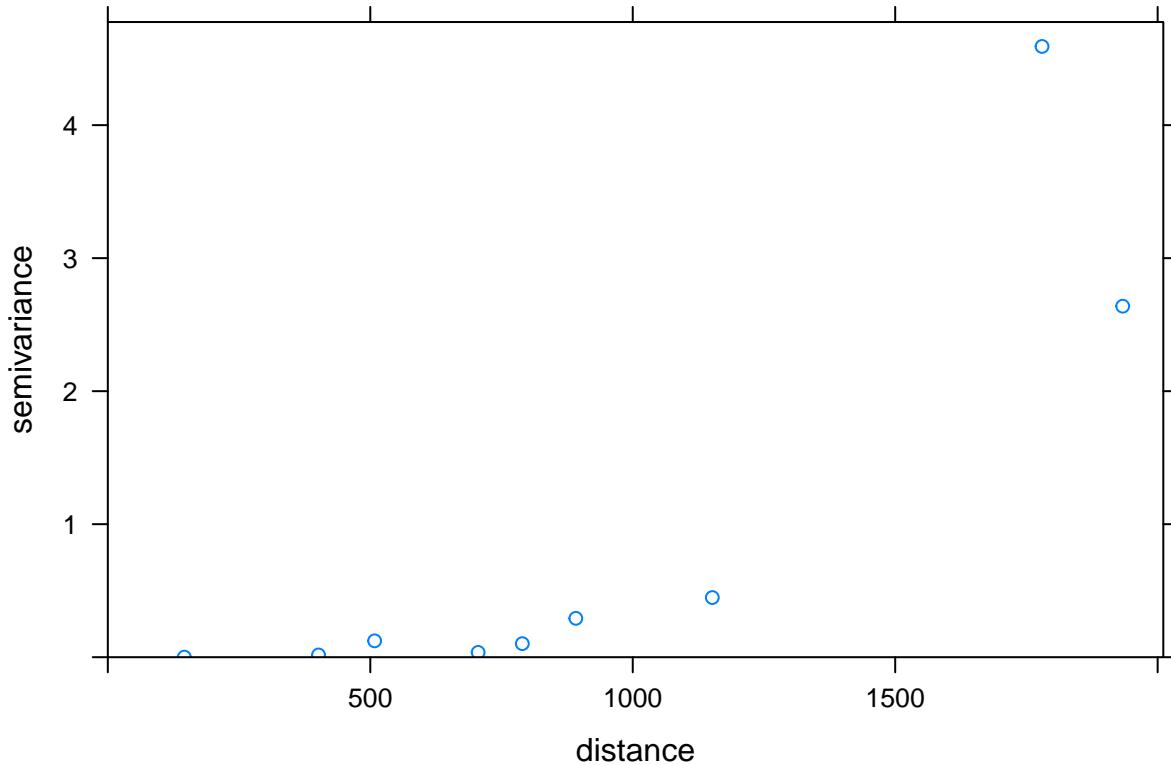
```
vario_kws2 <- variogram(temp~1, punkty[punkty$clc==2, ])
plot(vario_kws2)
```



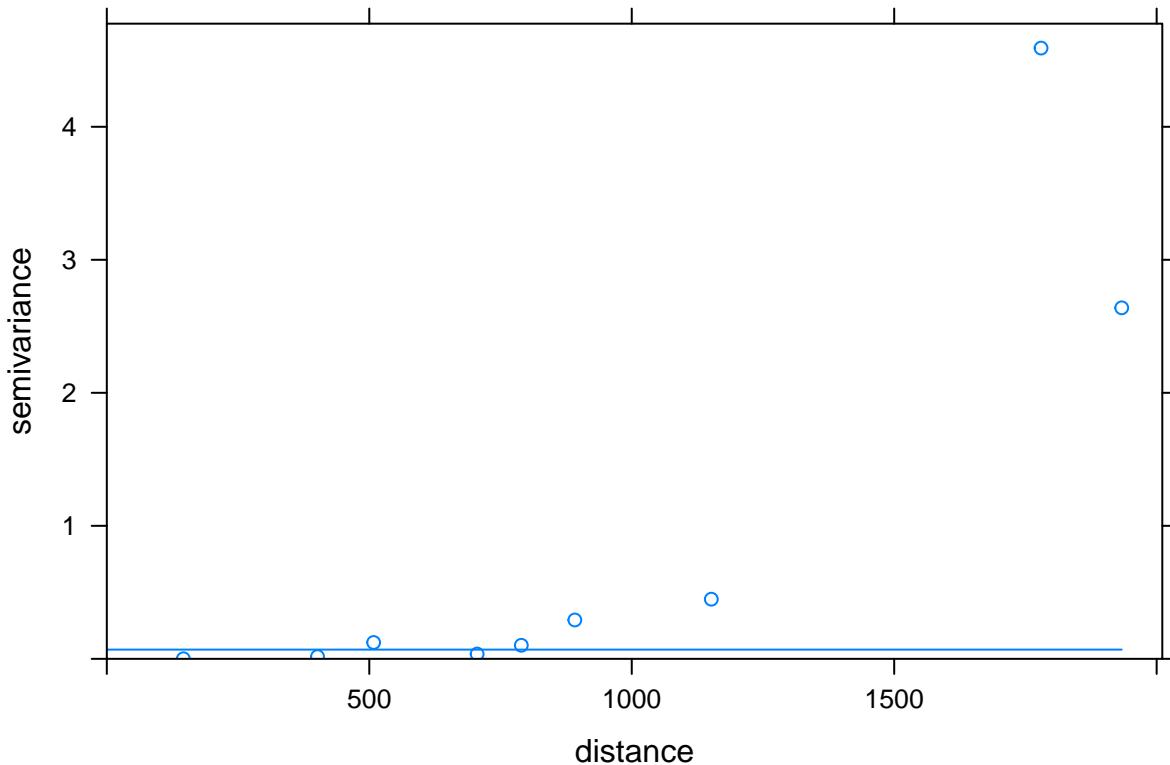
```
fitted_kws2 <- fit.variogram(vario_kws2, vgm(5, model = 'Gau', range = 4500, nugget = 0.1))
plot(vario_kws2, fitted_kws2)
```



```
vario_kws4 <- variogram(temp~1, punkty[punkty$clc==4, ])
plot(vario_kws4)
```



```
fitted_kws4 <- fit.variogram(vario_kws4, vgm(0.5, model = 'Nug', range = 0))
plot(vario_kws4, fitted_kws4)
```



```

kws1 <- kriging(temp~1, punkty[punkty$clc==1, ], siatka[na.omit(siatka$clc==1), ], model = fitted_kws1, n

## [using ordinary kriging]

kws2 <- kriging(temp~1, punkty[punkty$clc==2, ], siatka[na.omit(siatka$clc==2), ], model = fitted_kws2, n

## [using ordinary kriging]

kws4 <- kriging(temp~1, punkty[punkty$clc==4, ], siatka[na.omit(siatka$clc==4), ], model = fitted_kws4, n

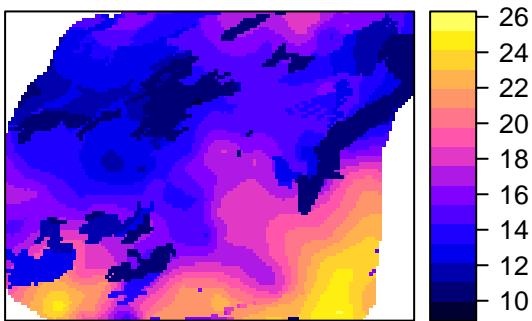
## [using ordinary kriging]

kws <- rbind(as.data.frame(kws1), as.data.frame(kws2), as.data.frame(kws4))
coordinates(kws) <- ~x+y
kws <- as(kws, 'SpatialPixelsDataFrame')

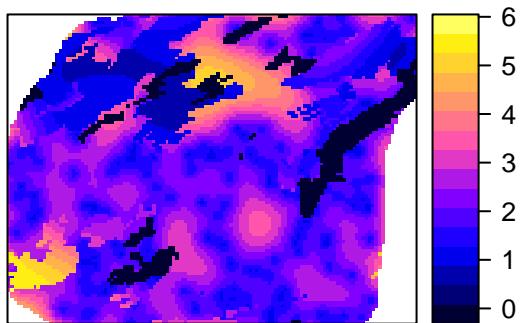
spplot(kws, 'var1.pred', sp.layout=punkty)
spplot(kws, 'var1.var', sp.layout=punkty)

```

### Predykcja KWS



### Wariancja predykcji KWS



## 11.2 Prosty kriging ze zmiennymi średnimi lokalnymi (LVM)

### 11.2.1 Prosty kriging ze zmiennymi średnimi lokalnymi (LVM)

- Prosty kriging ze zmiennymi średnimi lokalnymi zamiast znanej (stałej) stacjonarnej średniej wykorzystuje zmienne średnie lokalne uzyskane na podstawie innej informacji
- Lokalna średnia może być uzyskana za pomocą wyliczenia regresji liniowej pomiędzy zmienną badaną a zmienną dodatkową

```

coef <- lm(temp~srtm, punkty)$coef
coef

##  (Intercept)          srtm
## 17.506469957 -0.007291269

vario <- variogram(temp~srtm, punkty)
model_sim <- vgm(10, model = 'Sph', range = 4000, nugget = 1)
model_sim

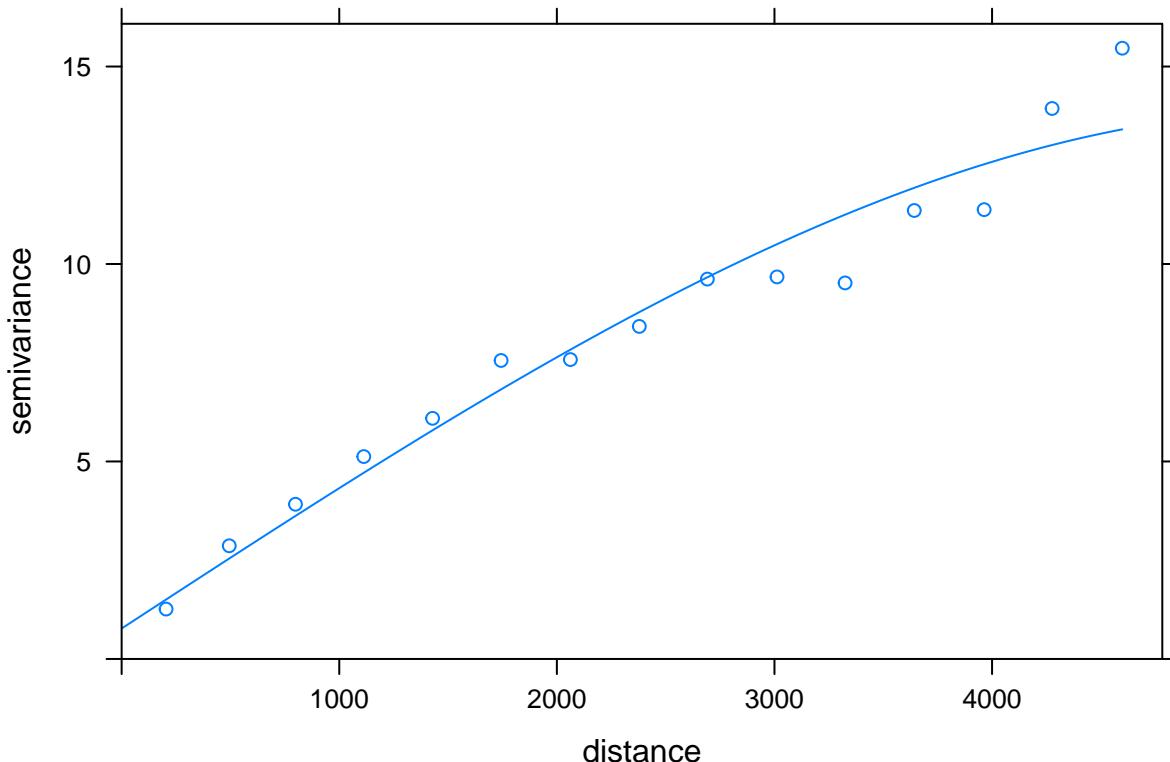
##   model psill range
## 1   Nug     1     0
## 2   Sph    10  4000

```

```
fitted_sim <- fit.variogram(vario, model_sim)
fitted_sim
```

```
##   model      psill      range
## 1   Nug  0.7705839  0.000
## 2   Sph 13.1155524 5474.628
```

```
plot(vario, model=fitted_sim)
```



```
sk_lvm <- krige(temp~srtm, punkty, siatka, model=fitted_sim, beta = coef)
```

```
## [using simple kriging]
```

```
summary(sk_lvm)
```

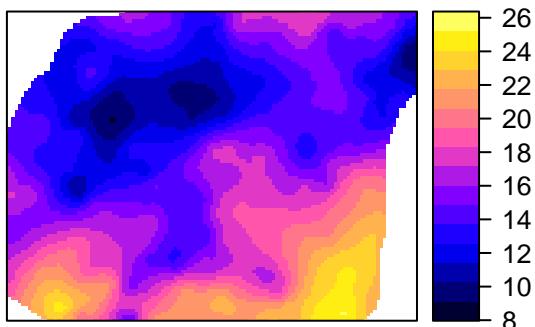
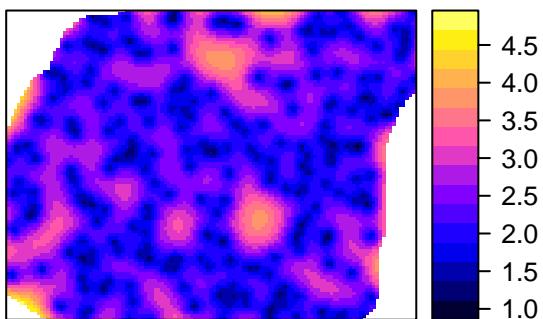
```
## Object of class SpatialPixelsDataFrame
## Coordinates:
##      min     max
## x 745586.7 756926.7
## y 712661.2 721211.2
## Is projected: TRUE
## proj4string :
## [+init=epsg:2180 +proj=tmerc +lat_0=0 +lon_0=19 +k=0.9993
## +x_0=500000 +y_0=-5300000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0
## +units=m +no_defs]
## Number of points: 10993
```

```

## Grid attributes:
##   cellcentre.offset cellsize cells.dim
##   s1           745586.7      90      127
##   s2           712661.2      90      96
## Data attributes:
##   var1.pred      var1.var
##   Min.    : 9.098  Min.    :1.108
##   1st Qu.:13.215  1st Qu.:1.931
##   Median  :15.237  Median  :2.241
##   Mean    :15.864  Mean    :2.312
##   3rd Qu.:18.049  3rd Qu.:2.637
##   Max.    :25.240  Max.    :4.705
##   NA's     :43       NA's     :43

spplot(sk_lvm, 'var1.pred')
spplot(sk_lvm, 'var1.var')

```

**Predykcja SK LVM****Wariancja predykcji SK LVM**

## 11.3 Kriging uniwersalny (ang. *Universal kriging*)

### 11.3.1 Kriging uniwersalny (ang. *Universal kriging*)

- Określany również jako kriging z trendem (ang. *Kriging with a trend model*)
- Zakłada on, że nieznana średnia lokalna zmienia się stopniowo na badanym obszarze

```

punkty$clc <- as.factor(punkty$clc)
vario_uk1 <- variogram(temp~clc, punkty)
vario_uk1

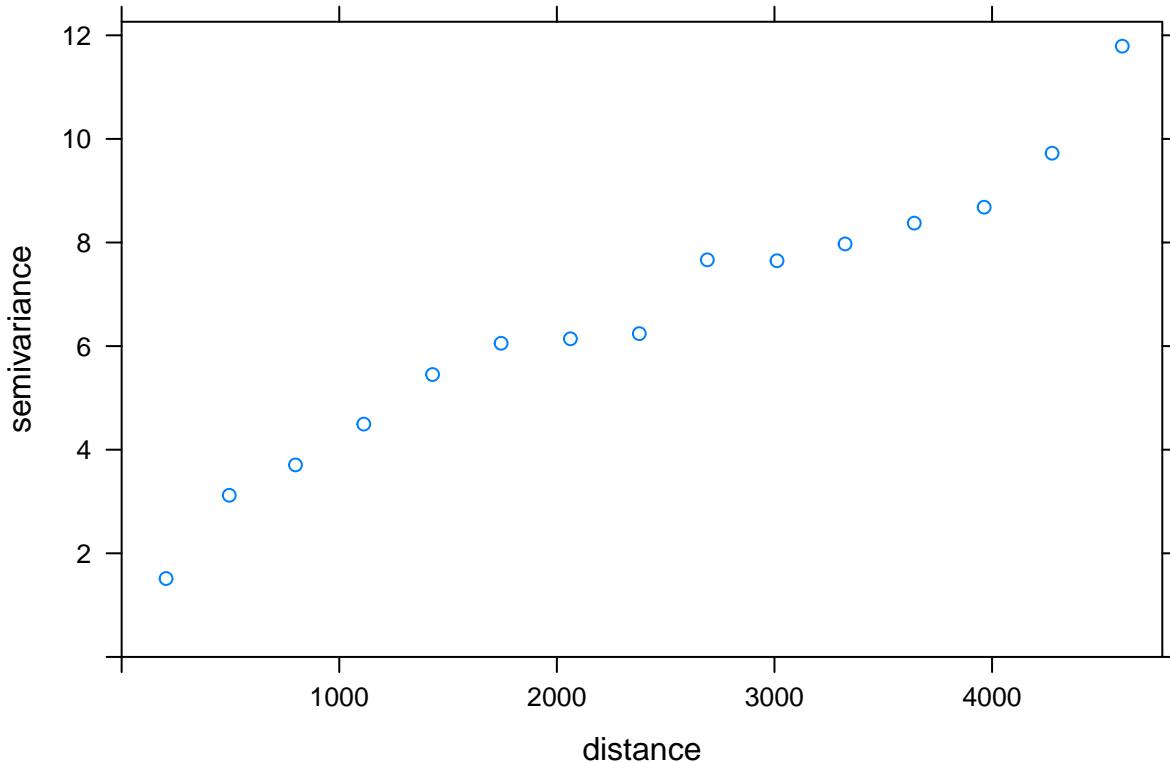
```

```

##      np      dist     gamma dir.hor dir.ver   id
## 1  127  204.2244  1.513128     0      0 var1
## 2  271  494.5572  3.120279     0      0 var1
## 3  522  798.7911  3.706435     0      0 var1
## 4  587 1112.7783  4.494623     0      0 var1
## 5  856 1428.7866  5.451369     0      0 var1
## 6  920 1743.7864  6.054877     0      0 var1
## 7 1068 2062.3041  6.140934     0      0 var1
## 8 1106 2378.9333  6.241295     0      0 var1
## 9 1184 2691.5206  7.665422     0      0 var1
## 10 1215 3011.7729  7.647765     0      0 var1
## 11 1362 3324.6705  7.973022     0      0 var1
## 12 1379 3642.5616  8.374149     0      0 var1
## 13 1405 3963.6776  8.681657     0      0 var1
## 14 1468 4276.0078  9.723868     0      0 var1
## 15 1482 4598.4144 11.789961     0      0 var1

```

```
plot(vario_uk1)
```



```

model_uk1 <- vgm(8, model = 'Sph', range = 3000, nugget = 1)
vario_fit_uk1 <- fit.variogram(vario_uk1, model=model_uk1)
vario_fit_uk1

```

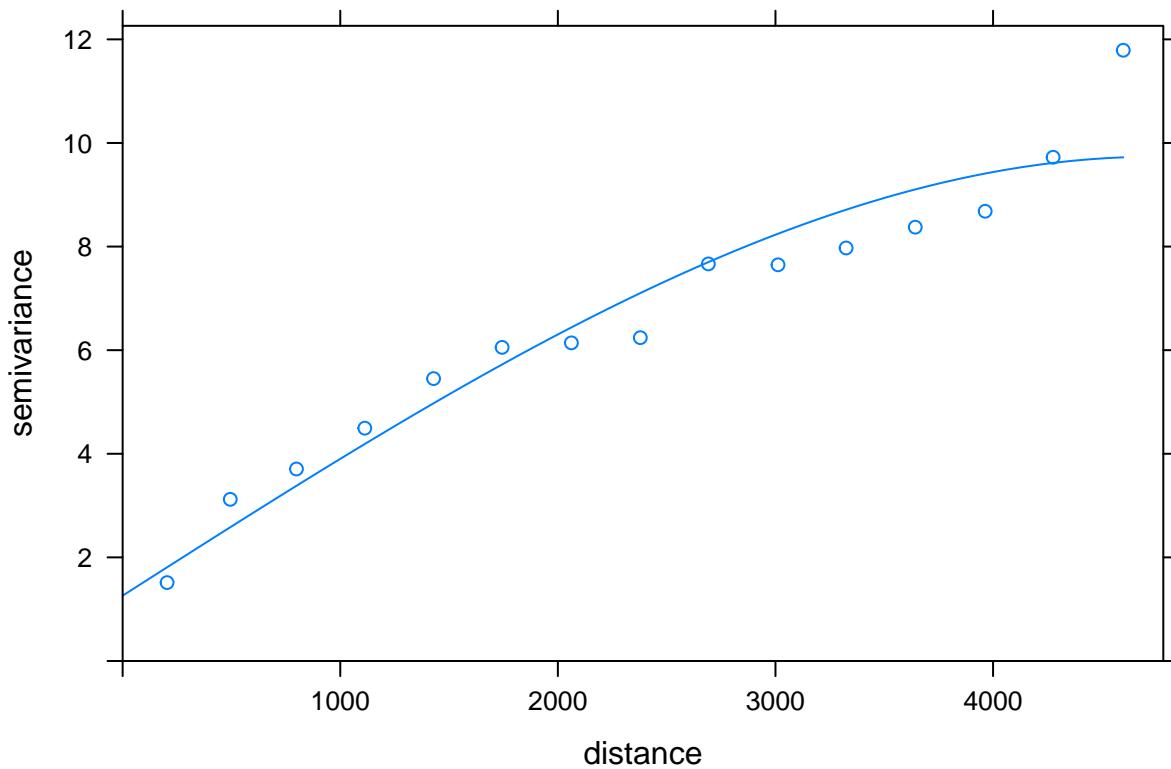
```

##    model    psill    range

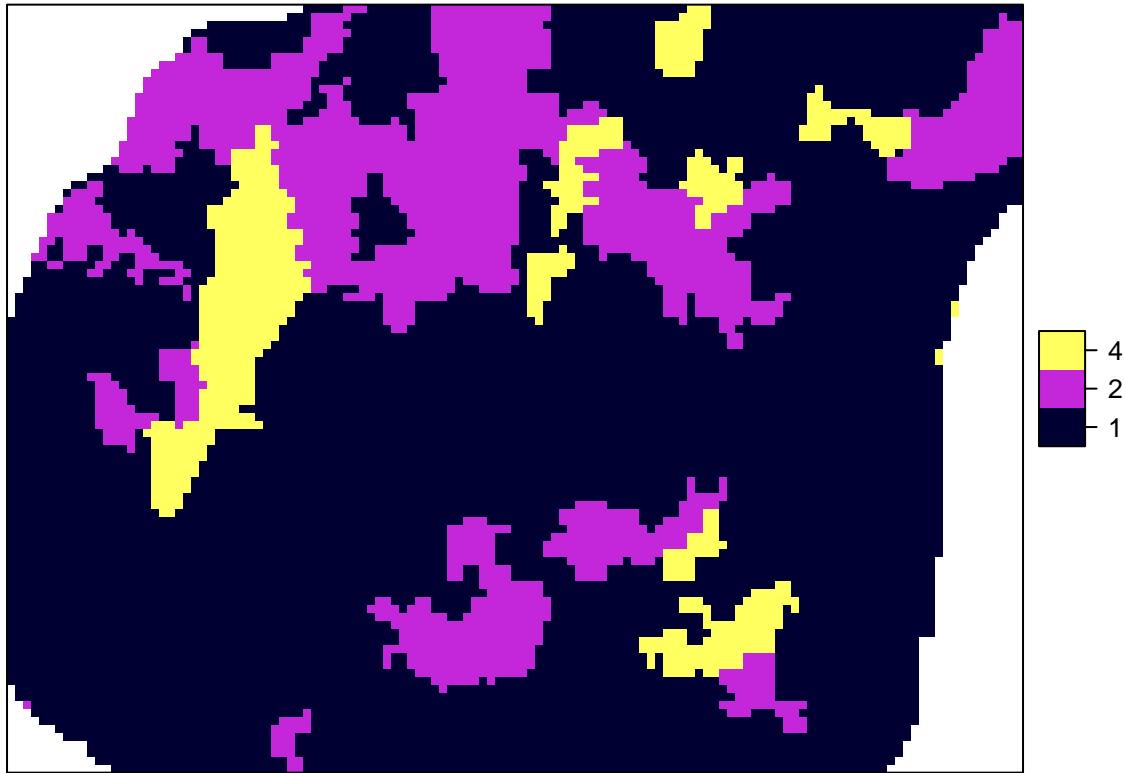
```

```
## 1 Nug 1.261541 0.000
## 2 Sph 8.472224 4742.211
```

```
plot(vario_uk1, vario_fit_uk1)
```



```
siatka$clc <- as.factor(siatka$clc)
spplot(siatka, 'clc')
```

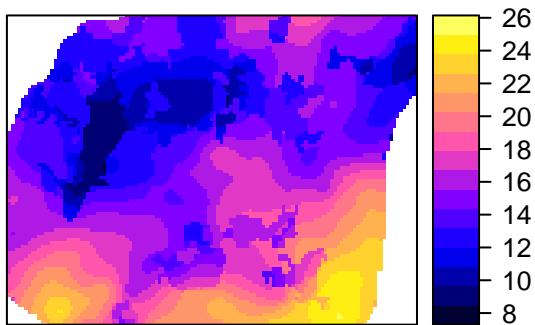


```
uk1 <- krige(temp~clc, locations = punkty, newdata=siatka, model=vario_fit_uk1)
```

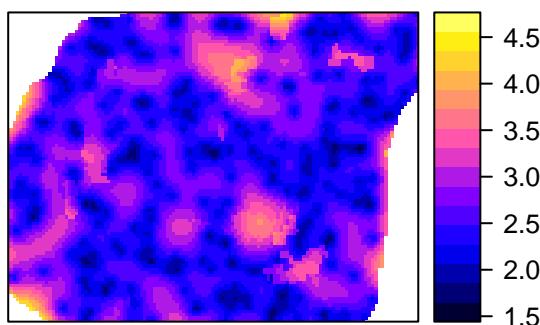
```
## [using universal kriging]
```

```
spplot(uk1, 'var1.pred')
spplot(uk1, 'var1.var')
```

### Predykcja KED



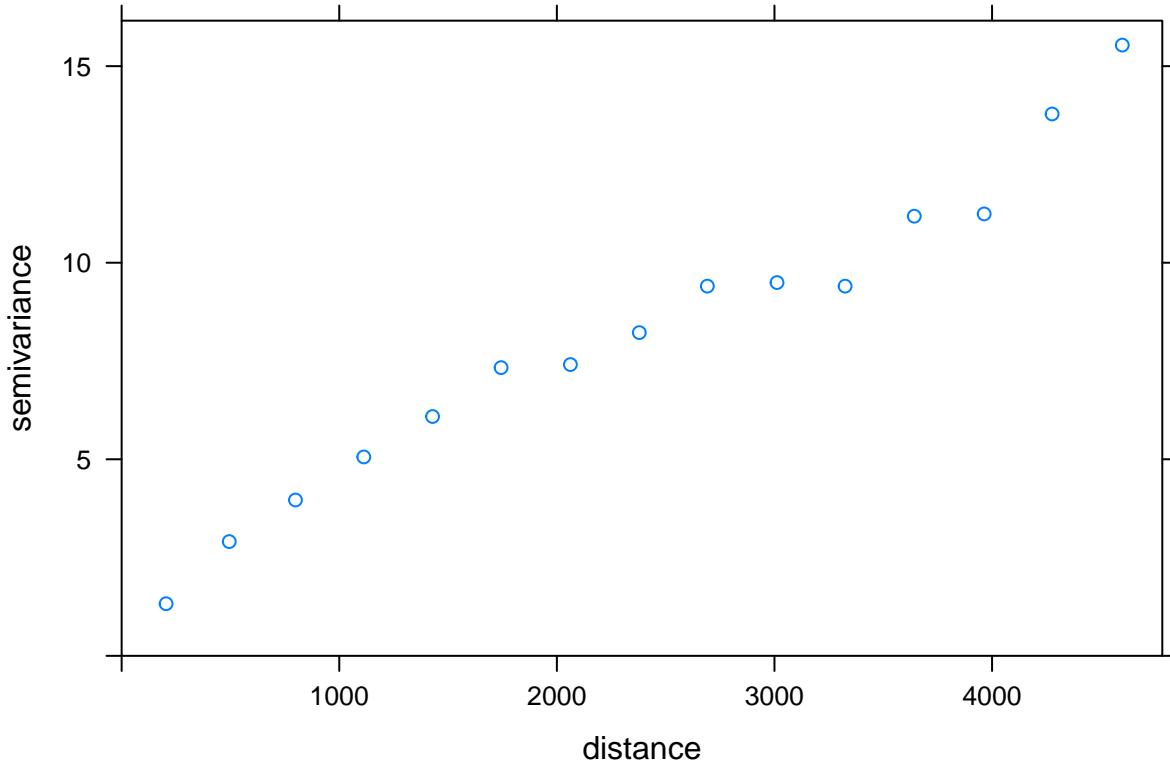
### Wariancja predykcji KED



```
vario_uk2 <- variogram(temp~ndvi+srtm, punkty)
vario_uk2
```

```
##      np      dist     gamma dir.hor dir.ver   id
## 1    127  204.2244  1.324959     0      0 var1
## 2    271  494.5572  2.906522     0      0 var1
## 3    522  798.7911  3.964307     0      0 var1
## 4    587 1112.7783  5.058684     0      0 var1
## 5    856 1428.7866  6.088010     0      0 var1
## 6    920 1743.7864  7.331287     0      0 var1
## 7   1068 2062.3041  7.409558     0      0 var1
## 8   1106 2378.9333  8.222238     0      0 var1
## 9   1184 2691.5206  9.404511     0      0 var1
## 10  1215 3011.7729  9.495409     0      0 var1
## 11  1362 3324.6705  9.403714     0      0 var1
## 12  1379 3642.5616 11.182666     0      0 var1
## 13  1405 3963.6776 11.241615     0      0 var1
## 14  1468 4276.0078 13.783975     0      0 var1
## 15  1482 4598.4144 15.536012     0      0 var1
```

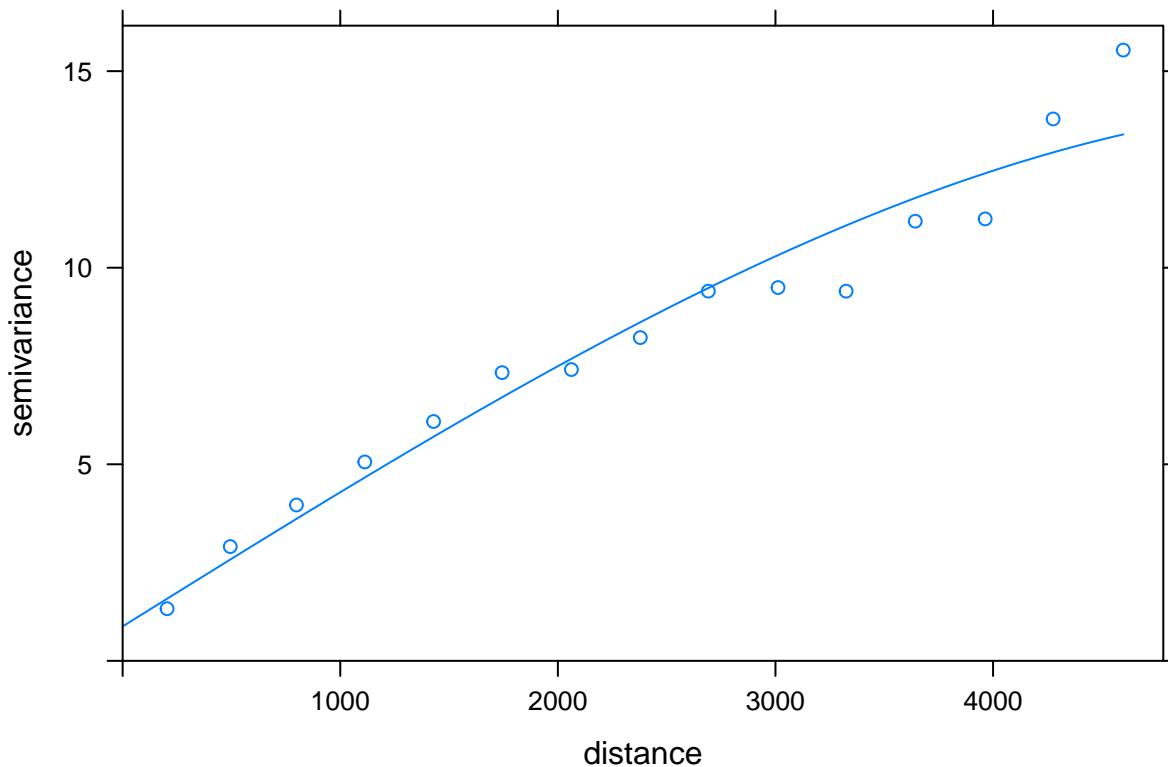
```
plot(vario_uk2)
```



```
model <- vgm(8, model = 'Sph', range = 3000, nugget = 1)
vario_fit_uk2 <- fit.variogram(vario_uk2, model=model)
vario_fit_uk2
```

```
##   model      psill      range
## 1   Nug  0.8757491  0.000
## 2   Sph 13.3016445 5789.379
```

```
plot(vario_uk2, vario_fit_uk2)
```

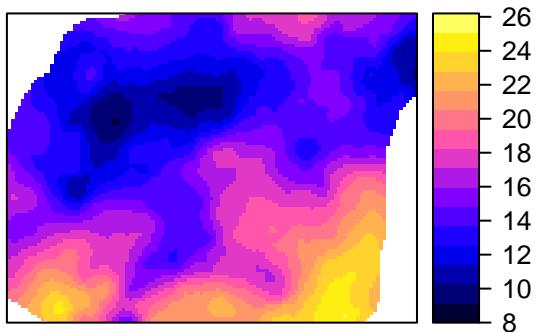


```
uk2 <- krige(temp~ndvi+srtm, locations = punkty, newdata=siatka, model=vario_fit_uk2)
```

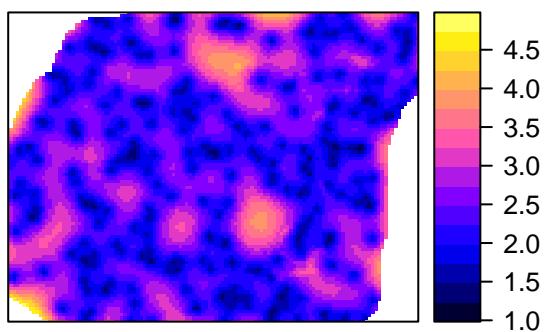
```
## [using universal kriging]
```

```
spplot(uk2, 'var1.pred')
spplot(uk2, 'var1.var')
```

### Predykcja KED



### Wariancja predykcji KED



# Chapter 12

## Ocena jakości estymacji

```
library('geostatbook')
data(punkty)
```

### 12.1 Statystyki jakości estymacji

#### 12.1.1 Statystyki jakości estymacji

- Służą do oceny i porównania jakości estymacji
- Do podstawowych statystyk ocen jakości estymacji należą:
  - Średni błąd predykcji (MPE, ang. *mean prediction error*)
  - Pierwiastek średniego błędu kwadratowego (RMSE, ang. *root square prediction error*)
  - Współczynnik korelacji
  - Rozkład błędu (np. 5. percentyl, mediana, 95. percentyl)

#### 12.1.2 Statystyki jakości estymacji

- Idealna estymacja dawałaby brak błędu oraz współczynnik korelacji pomiędzy pomiarami (całą populacją) i szacunkiem równy 1
- Wysokie, pojedyncze wartości błędu mogą świadczyć, np. o wystąpieniu wartości odstających

#### 12.1.3 Średni błąd estymacji

- Optymalnie wartość średniego błędu estymacji powinna być jak najbliżej 0

$$MPE = \frac{\sum_{i=1}^n (\hat{v}_i - v_i)}{n}$$

#### 12.1.4 Pierwiastek średniego błędu kwadratowego

- Optymalnie wartość pierwiastka średniego błędu kwadratowego powinna być jak najmniejsza

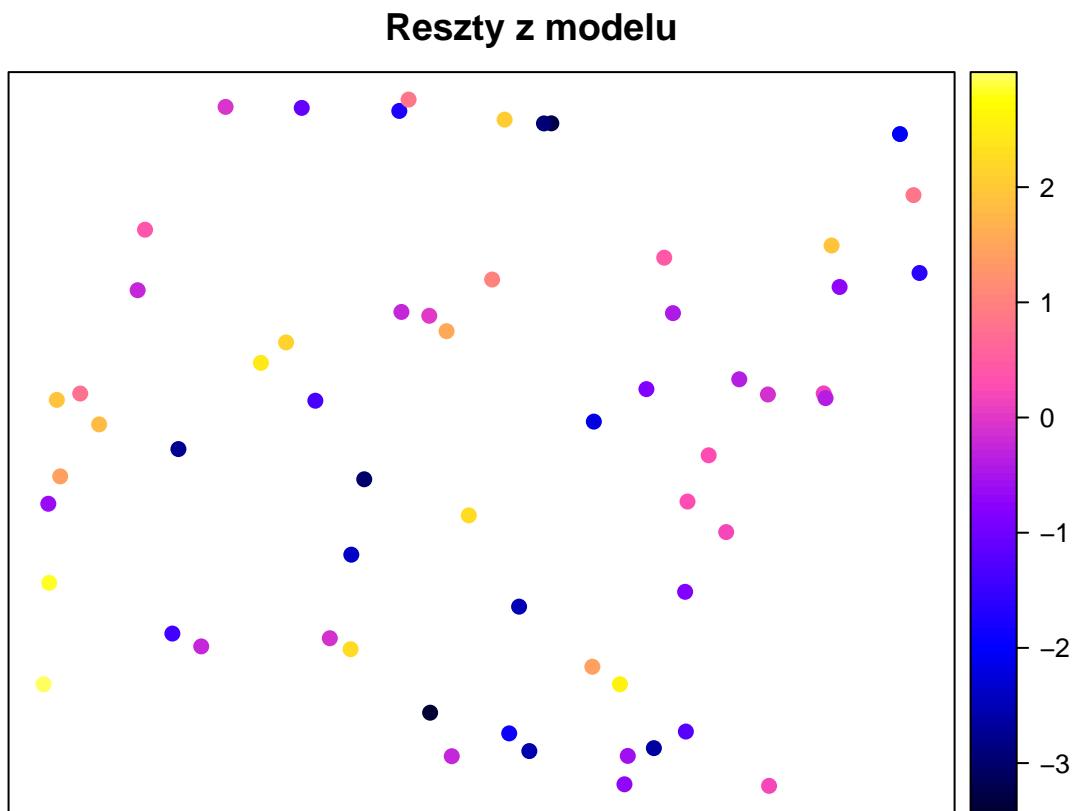
$$RMSE = \sqrt{\frac{\sum_{i=1}^n (v_i - \hat{v}_i)^2}{n}}$$

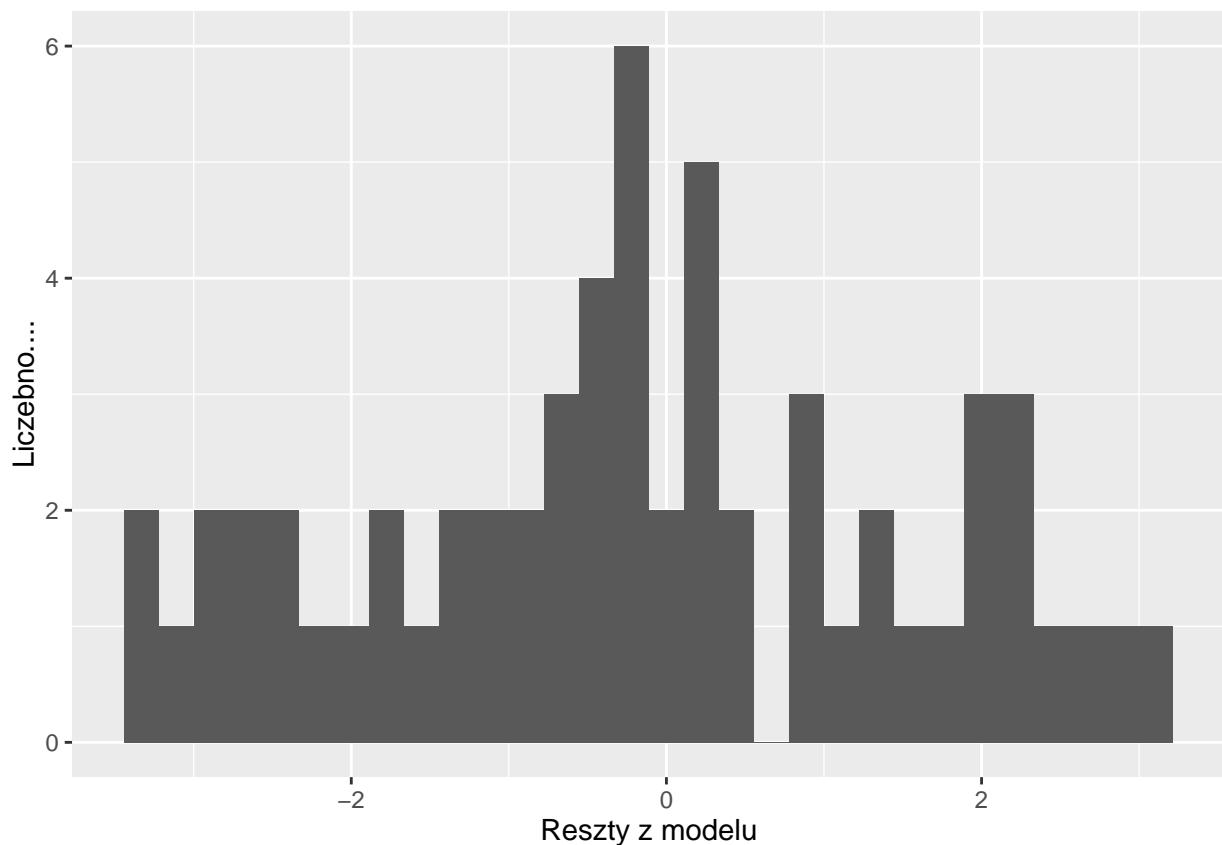
### 12.1.5 Współczynnik korelacji

- Optymalnie wartość współczynnika korelacji powinna być jak najwyższa

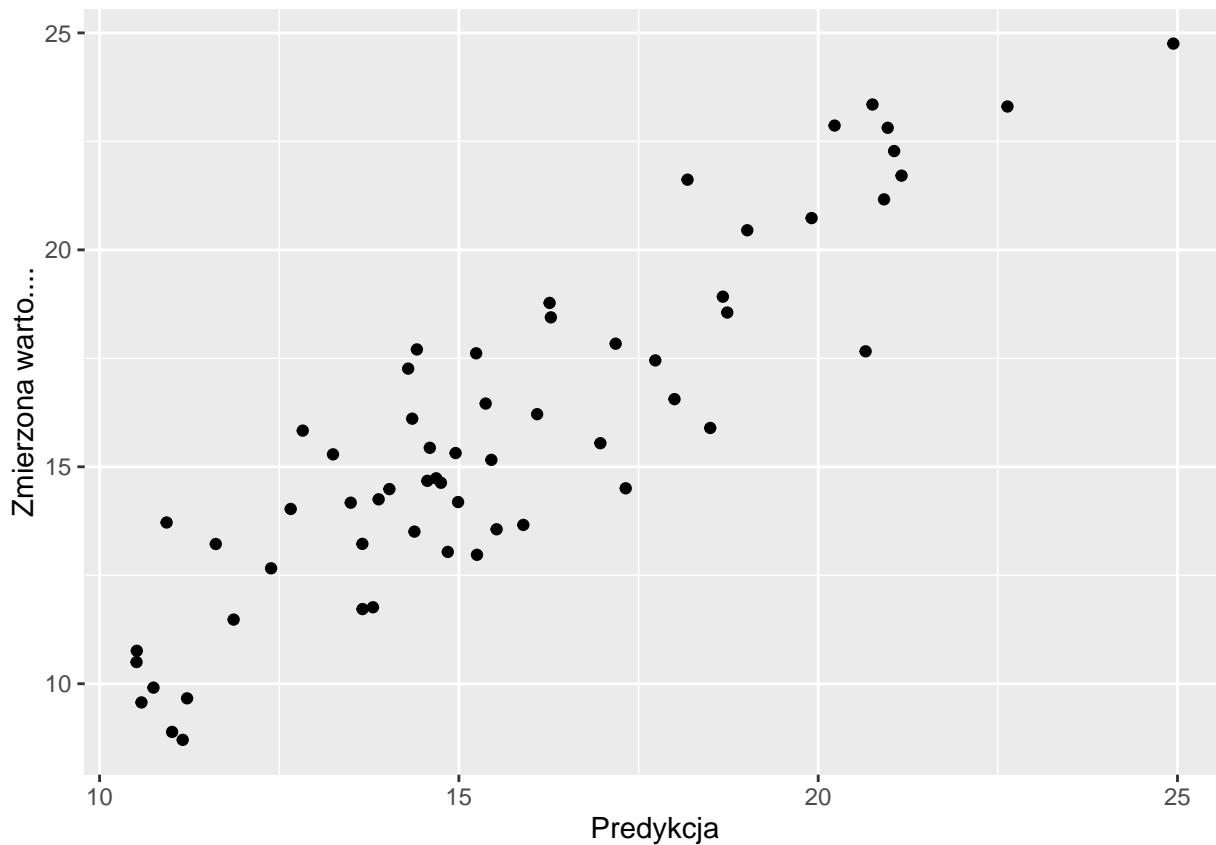
### 12.1.6 Statystyki jakości estymacji | Mapa

```
## [using simple kriging]
```



**12.1.7 Statystyki jakości estymacji | Histogram**

### 12.1.8 Statystyki jakości estymacji | Wykres rozrzutu



## 12.2 Walidacja wyników estymacji

### 12.2.1 Walidacja wyników estymacji

- Dokładne dopasowanie modelu do danych może w efekcie nie dawać najlepszych wyników
- W efekcie ważne jest stosowanie metod pozwalających na wybranie optymalnego modelu
- Do takich metod należy, między innymi, walidacja podzbiorem (ang. *jackknifing*) oraz kroswalidacja (ang. *crossvalidation*)

### 12.2.2 Walidacja podzbiorem

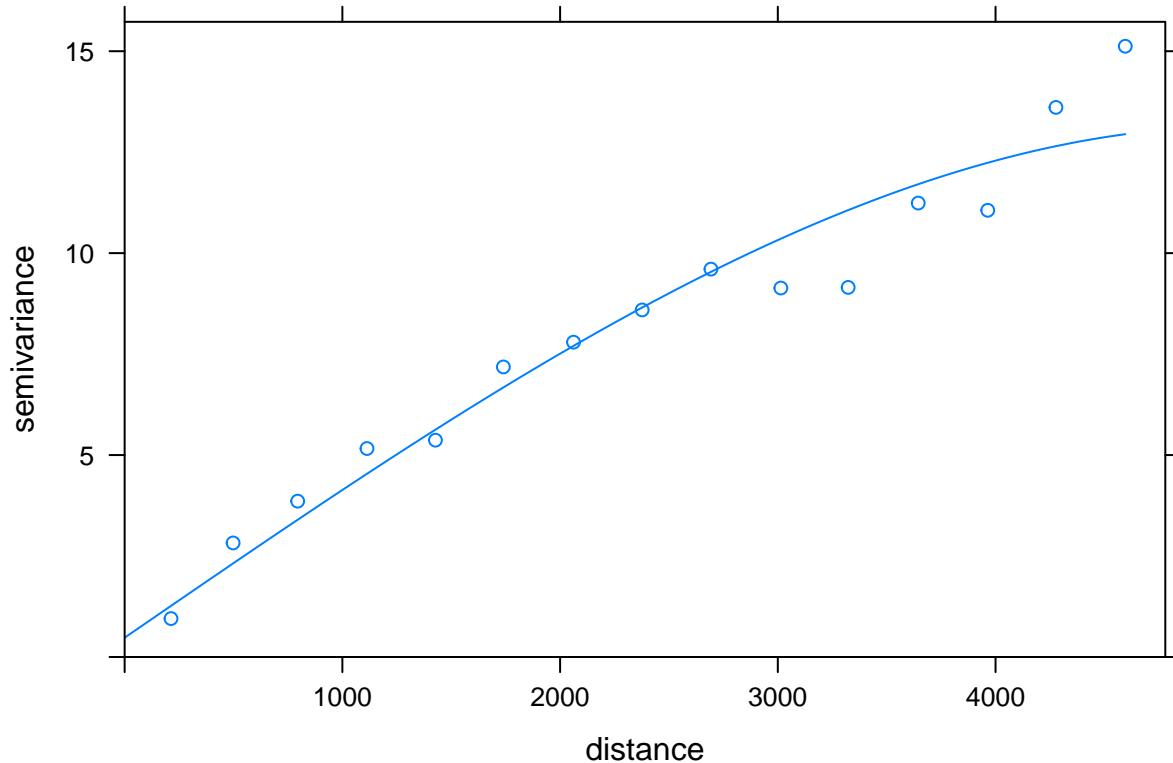
- Polega na podziale zbioru danych na dwa podzbiory - treningowy i testowy
- Zbiór treningowy służy do estymacji wartości
- Wynik estymacji porównywany jest z rzeczywistymi wartościami ze zbioru testowego
- Zaletą tego podejścia jest stosowanie danych niezależnych od estymacji
- Wadą jest konieczność posiadania dużego zbioru danych

### 12.2.3 Walidacja podzbiorem

```
set.seed(124)
indeks <- as.vector(createDataPartition(punkty$temp, p=0.75, list=FALSE))
indeks
```

```
## [1] 1 2 4 7 8 9 12 13 14 15 16 17 18 19 22 23 25
## [18] 26 27 29 30 32 34 35 36 37 38 40 41 42 43 45 49 51
## [35] 52 53 54 55 56 58 59 60 61 62 64 66 67 69 70 73 75
## [52] 76 78 79 81 82 83 84 85 86 87 88 89 90 91 94 96 97
## [69] 98 100 101 102 103 106 107 108 109 110 111 112 113 114 115 117 118
## [86] 120 122 124 125 127 128 129 130 132 133 134 135 137 138 139 140 141
## [103] 142 143 144 146 148 149 150 153 154 155 156 157 158 159 160 161 162
## [120] 163 167 168 169 171 173 174 175 176 177 180 181 183 184 185 186 187
## [137] 188 189 190 191 193 194 195 196 199 200 201 202 204 205 208 209 210
## [154] 212 213 214 215 216 217 218 219 220 221 223 225 226 227 228 229 230
## [171] 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247
## [188] 248 249 250
```

```
train <- punkty[indeks, ]
test <- punkty[-indeks, ]
vario <- variogram(temp~1, data=train)
model <- vgm(10, model = 'Sph', range = 4000, nugget = 0.5)
fitted <- fit.variogram(vario, model)
plot(vario, model=fitted)
```



```
test_sk <- krige(temp~1, train, test, model=fitted, beta=15.324)
```

```
## [using simple kriging]
```

```
summary(test_sk)
```

```
## Object of class SpatialPointsDataFrame
## Coordinates:
##      min     max
## x 745731 756567.0
## y 712629 721118.7
## Is projected: TRUE
## proj4string :
## [+init=epsg:2180 +proj=tmerc +lat_0=0 +lon_0=19 +k=0.9993
## +x_0=500000 +y_0=-5300000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0
## +units=m +no_defs]
## Number of points: 60
## Data attributes:
##   var1.pred      var1.var
##   Min. :10.51  Min. :1.002
##   1st Qu.:13.62 1st Qu.:1.654
##   Median :14.97 Median :2.084
##   Mean   :15.66 Mean   :2.187
##   3rd Qu.:18.05 3rd Qu.:2.608
##   Max.   :24.94 Max.   :5.602
```

```
reszta_sk <- test_sk$var1.pred - test$temp
summary(reszta_sk)
```

```
##      Min. 1st Qu. Median  Mean 3rd Qu.  Max.
## -3.4360 -1.3860 -0.2442 -0.1954  0.9098  2.9990
```

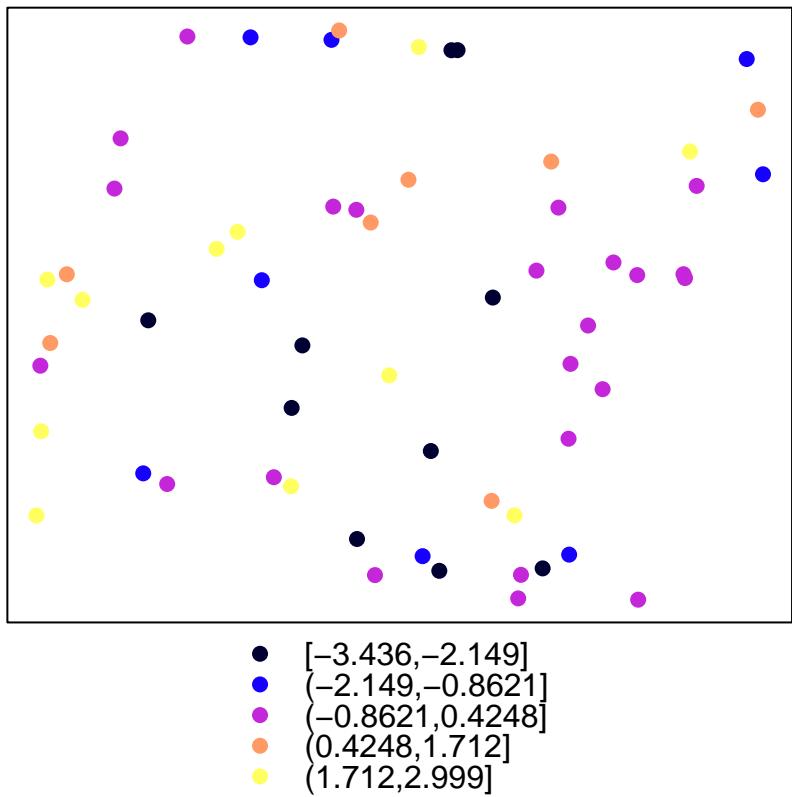
```
MPE <- mean(test_sk$var1.pred - test$temp)
MPE
```

```
## [1] -0.1953769
```

```
RMSE <- sqrt(mean((test$temp-test_sk$var1.pred)^2))
RMSE
```

```
## [1] 1.692512
```

```
test_sk$reszty <- reszta_sk
spplot(test_sk, 'reszty')
```



#### 12.2.4 Kroswalidacja

- W przypadku kroswalidacji te same dane wykorzystywane są do budowy modelu, estymacji, a następnie do oceny prognozy
  - Procedura kroswalidacji LOO (ang. *leave-one-out cross-validation*)
1. Zbudowanie matematycznego modelu z dostępnych obserwacji
  2. Dla każdej znanej obserwacji następuje:
    - Usunięcie jej ze zbioru danych
    - Użycie modelu do wykonania predykcji w miejscu tej obserwacji
    - Wyliczenie reszty (ang. *residual*), czyli różnicy pomiędzy znaną wartością a obserwacją
  3. Podsumowanie otrzymanych wyników
    - W pakiecie `gstat`, kroswalidacja LOO jest dostępna w funkcjach `krige.cv` oraz `gstat.cv`

#### 12.2.5 Kroswalidacja

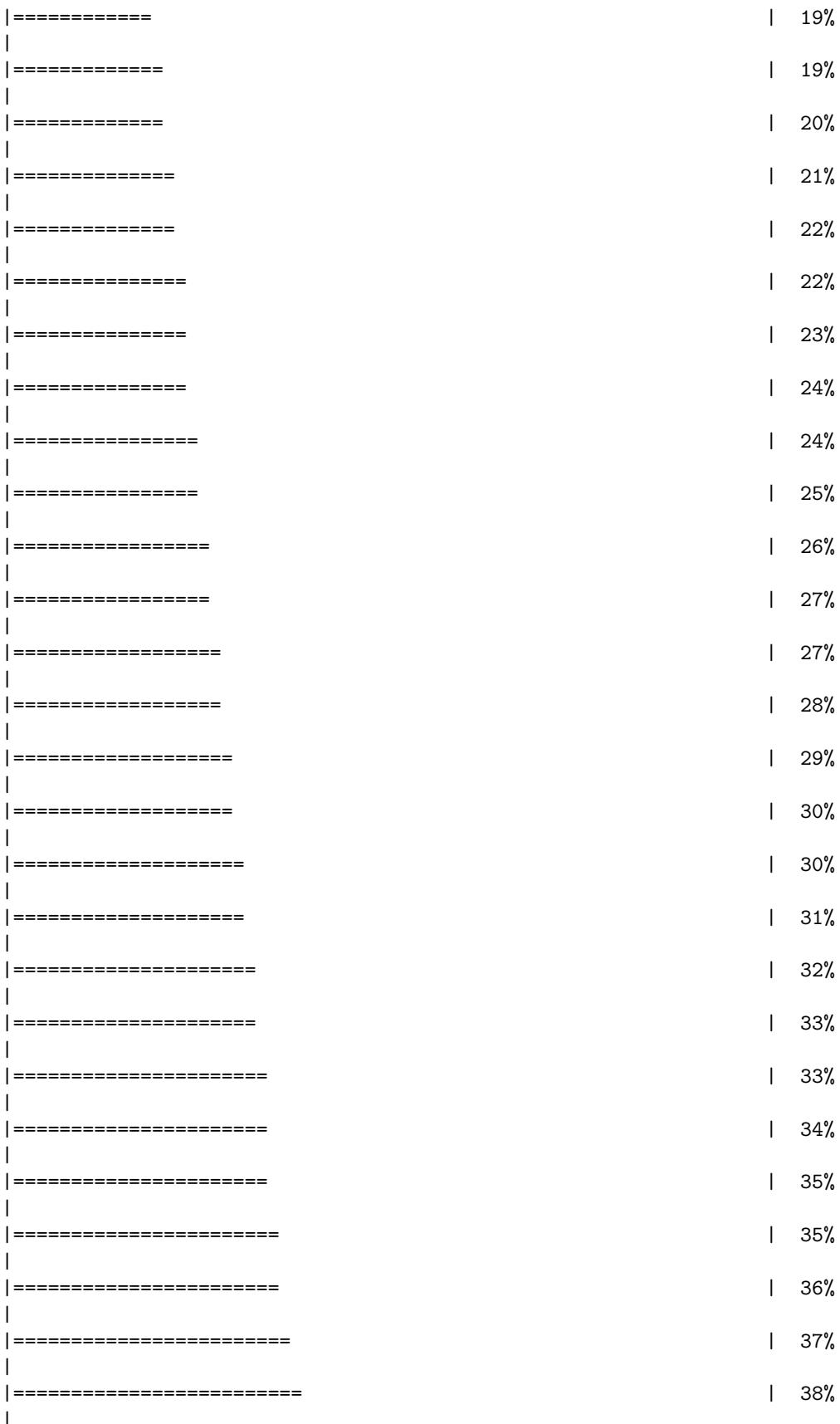
```

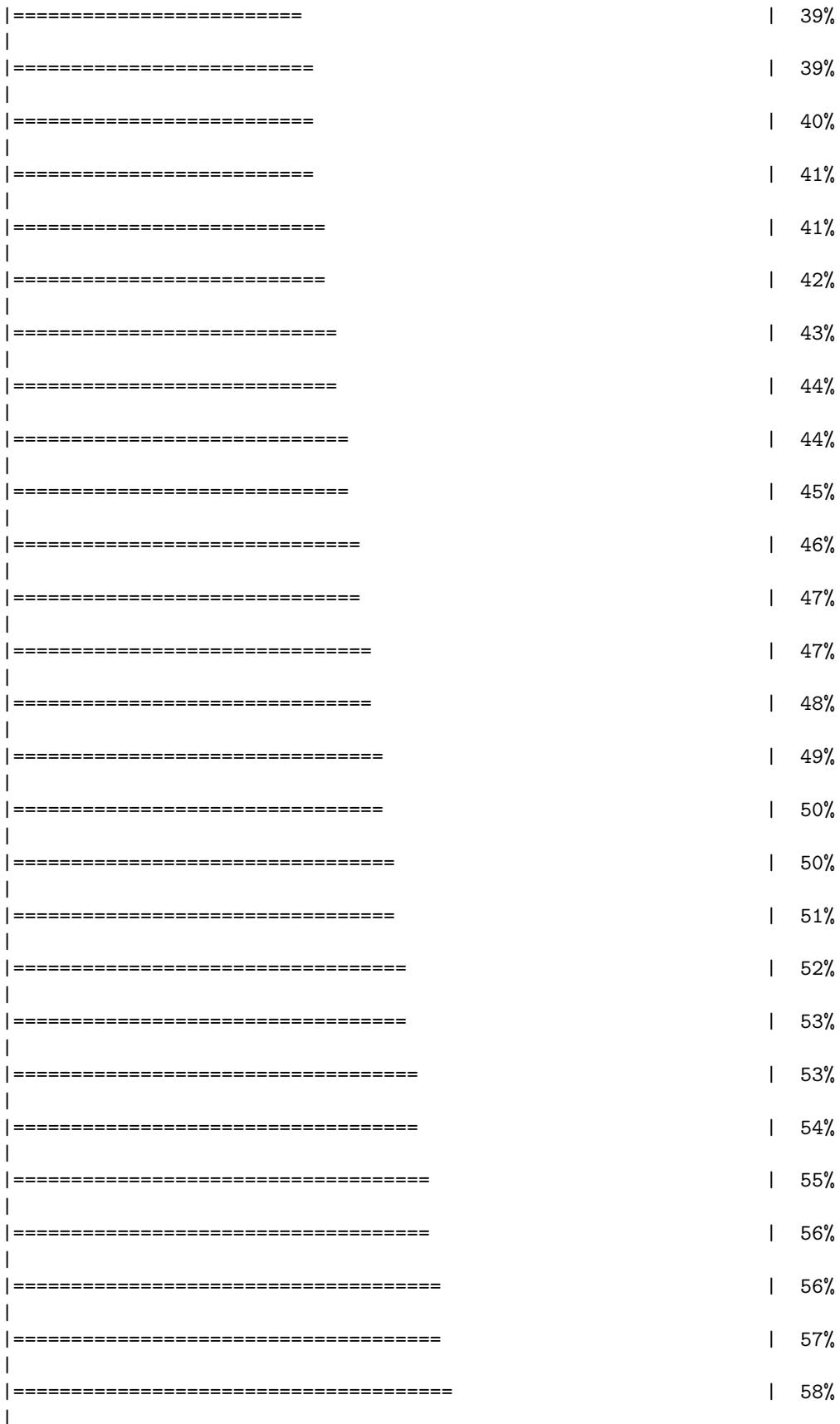
vario <- variogram(temp~1, data=punkty)
model <- vgm(10, model = 'Sph', range = 4000, nugget = 0.5)
fitted <- fit.variogram(vario, model)

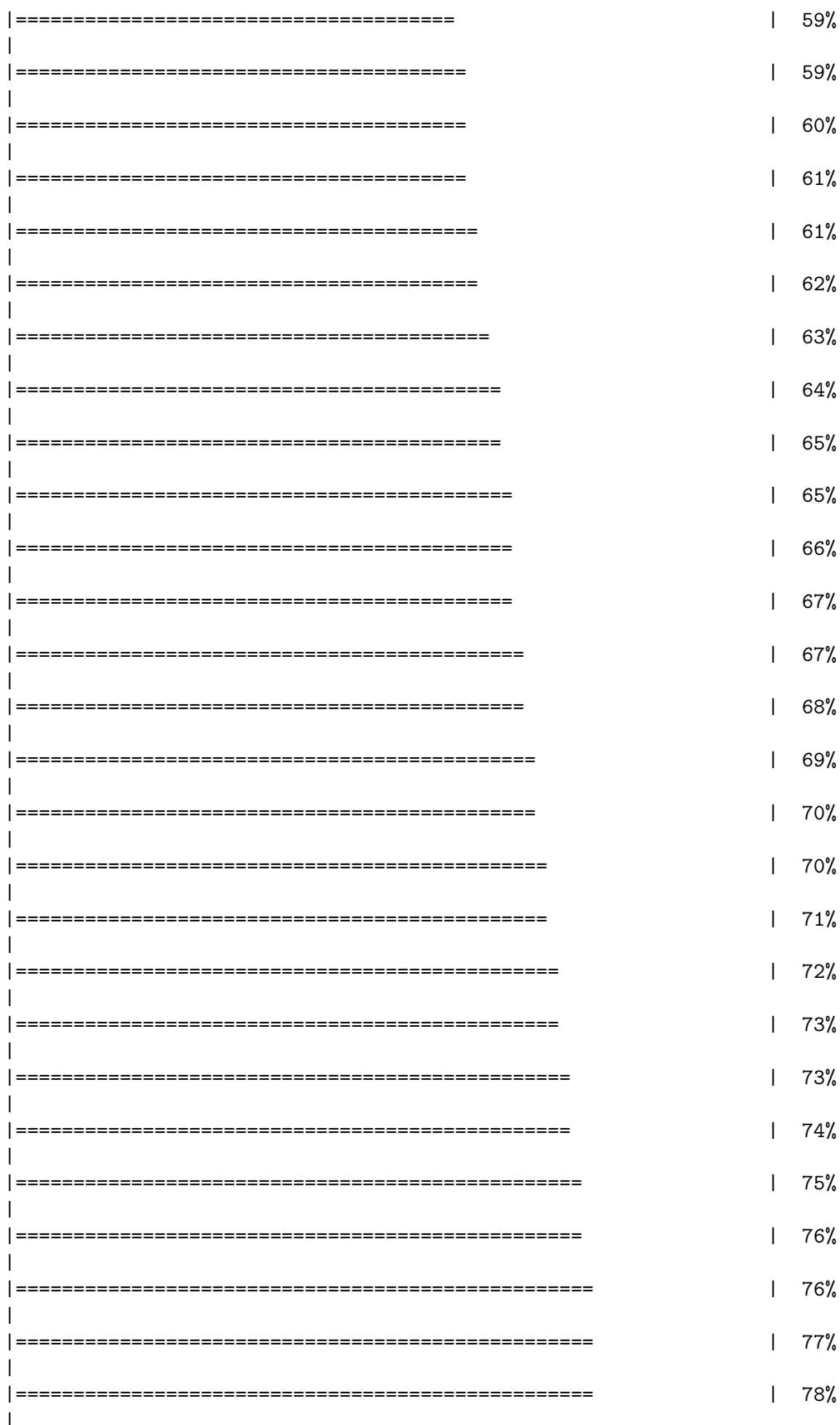
cv_sk <- krige.cv(temp~1, punkty, model=fitted, beta=15.324)

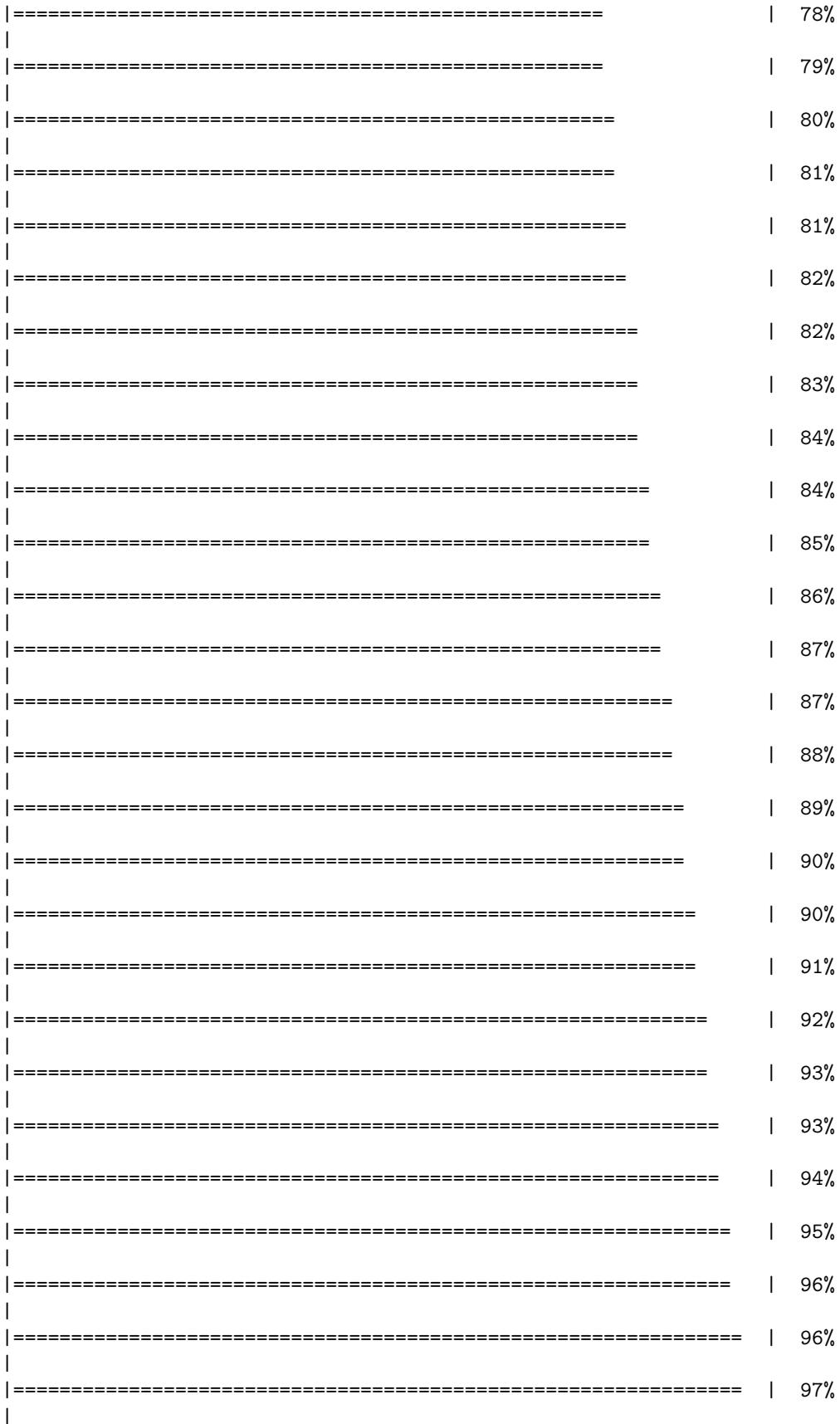
```

```
##  
|  
|  
|  
|= | 0%  
|  
|= | 1%  
|  
|  
|== | 2%  
|  
|== | 2%  
|  
|== | 3%  
|  
|== | 4%  
|  
|== | 4%  
|  
|== | 5%  
|  
|==== | 6%  
|  
|==== | 7%  
|  
|===== | 7%  
|  
|===== | 8%  
|  
|===== | 9%  
|  
|===== | 10%  
|  
|===== | 10%  
|  
|===== | 11%  
|  
|===== | 12%  
|  
|===== | 13%  
|  
|===== | 13%  
|  
|===== | 14%  
|  
|===== | 15%  
|  
|===== | 16%  
|  
|===== | 16%  
|  
|===== | 17%  
|  
|===== | 18%  
|  
|===== | 18%
```







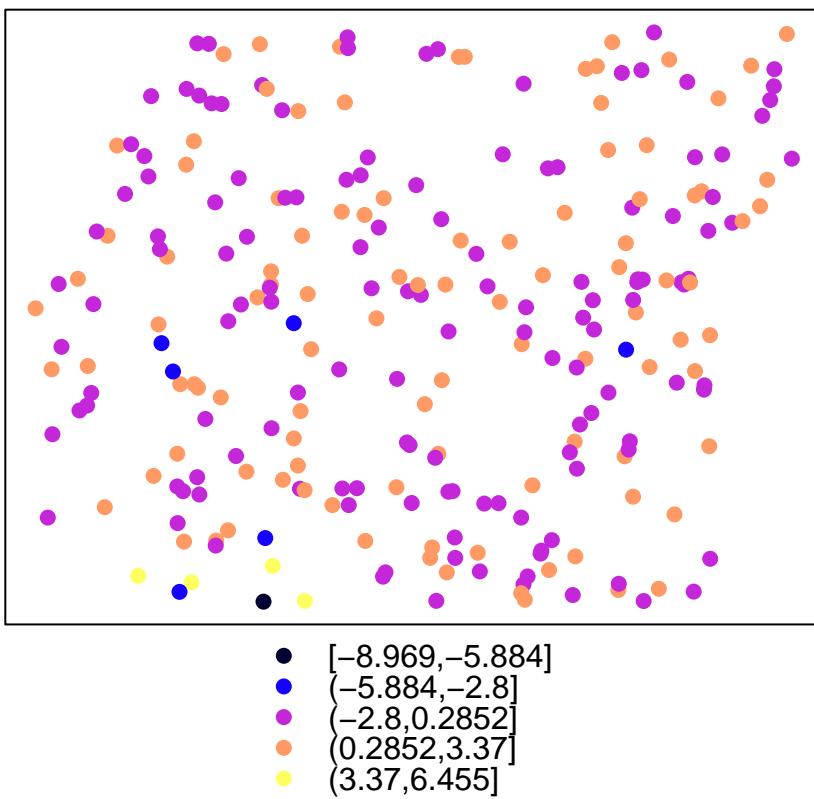


```
|===== | 98%
| |
|===== | 98%
| |
|===== | 99%
| |
|=====| 100%
```

```
summary(cv_sk)
```

```
## Object of class SpatialPointsDataFrame
## Coordinates:
##      min     max
## x 745546.9 756937.4
## y 712618.8 721192.6
## Is projected: NA
## proj4string : [NA]
## Number of points: 250
## Data attributes:
##   var1.pred      var1.var      observed      residual
##   Min.    : 9.786  Min.    :1.219  Min.    : 8.706  Min.    :-8.96908
##   1st Qu.:13.333  1st Qu.:1.806  1st Qu.:13.284  1st Qu.:-0.95234
##   Median :15.298  Median :2.093  Median :15.309  Median : 0.04980
##   Mean   :15.936  Mean   :2.220  Mean   :15.950  Mean   : 0.01423
##   3rd Qu.:18.062  3rd Qu.:2.477  3rd Qu.:18.273  3rd Qu.: 0.85669
##   Max.   :24.994  Max.   :5.320  Max.   :26.139  Max.   : 6.45468
##   zscore      fold
##   Min.    :-4.930132  Min.    : 1.00
##   1st Qu.:-0.647920  1st Qu.: 63.25
##   Median : 0.036433  Median :125.50
##   Mean   : 0.008082  Mean   :125.50
##   3rd Qu.: 0.598859  3rd Qu.:187.75
##   Max.   : 3.330852  Max.   :250.00
```

```
spplot(cv_sk, 'residual')
```



# Chapter 13

## Symulacje

```
library('geostatbook')
data(punkty)
data(siatka)
```

### 13.1 Symulacje geostatystyczne

#### 13.1.1 Symulacje geostatystyczne

- Kriging daje optymalne predykcje, czyli wyznacza najbardziej potencjalnie możliwą wartość dla wybranej lokalizacji
- Dodatkowo, efektem krigingu jest wygładzony obraz. W konsekwencji wyniki estymacji różnią się od danych pomiarowych
- Jest to tylko (aż?) predykcja. Prawdziwa wartość jest niepewna ...
- Korzystając z symulacji geostatystycznych nie tworzymy predykcji, ale generujemy równie prawdopodobne możliwości poprzez symulację z rozkładu prawdopodobieństwa (wykorzystując generator liczb losowych)

#### 13.1.2 Symulacje geostatystyczne | Cel

- Efekt symulacji ma bardziej realistyczny przestrzenny wzór (ang. *pattern*) niż kriging, którego efektem jest wygładzona reprezentacja rzeczywistości
- Każda z symulowanych map jest równie prawdopodobna
- Symulacje pozwalają na przedstawianie niepewności interpolacji
- Jednocześnie - kriging jest znacznie lepszy, gdy naszym celem jest jak najdokładniejsza predykcja

### 13.2 Typy symulacji

#### 13.2.1 Typy symulacji

- Symulacje bezwarunkowe (ang. Unconditional Simulations) - wykorzystuje semiwariogram, żeby włączyć informację przestrzenną, ale wartości ze zmierzonych punktów nie są wykorzystywane.
- Symulacje warunkowe (ang. Conditional Simulations) - opiera się ona o średnią wartość, strukturę kowariancji oraz obserwowane wartości

### 13.3 Symulacje bezwarunkowe

```

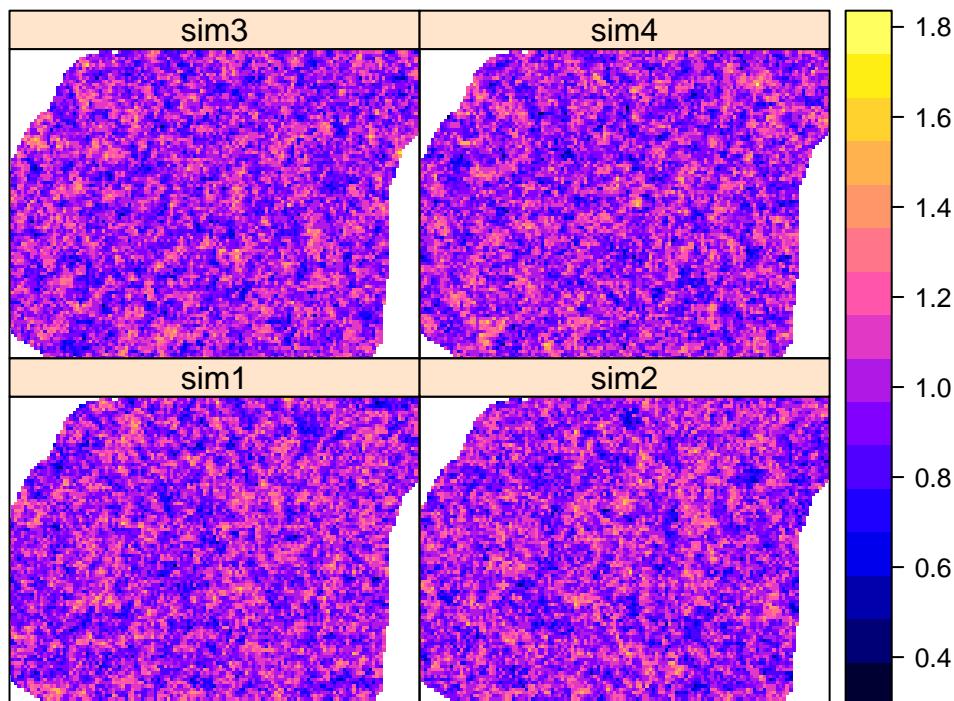
sym_bezw1 <- krige(formula=z~1, locations=NULL, newdata=siatka, dummy=TRUE,
                     beta=1, model=vgm(psill=0.025,model='Exp',range=100), nsim=4, nmax=30)

## [using unconditional Gaussian simulation]

spplot(sym_bezw1, main='Przestrzennie skorelowana powierzchnia \nśrednia=1, sill=0.025, zasięg=100, model wyk..adniczy')

```

**Przestrzennie skorelowana powierzchnia  
..rednia=1, sill=0.025, zasi..g=100, model wyk..adniczy**



```

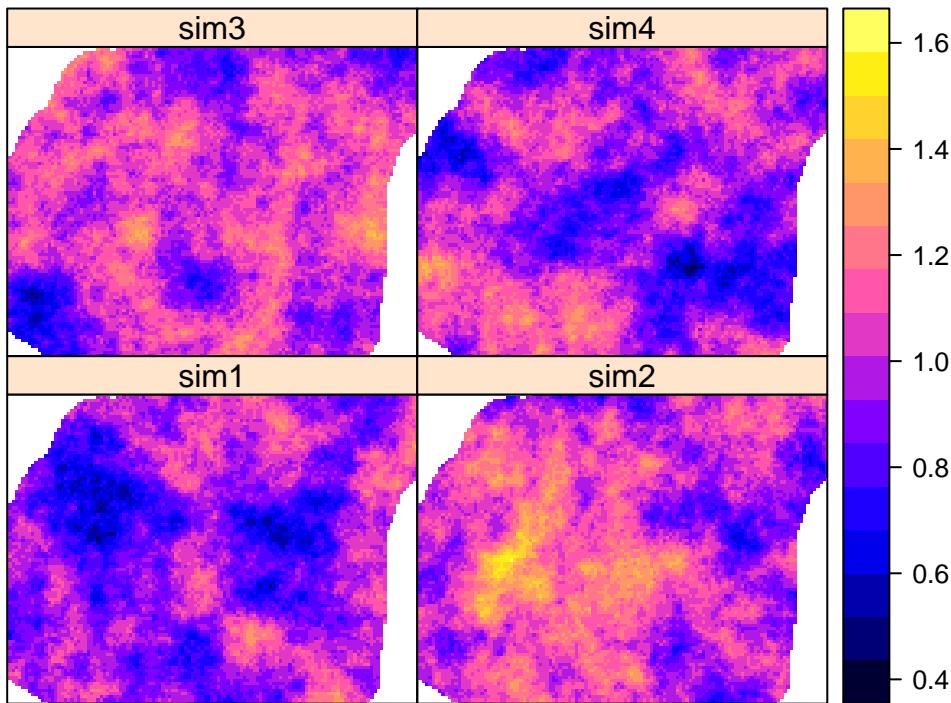
sym_bezw2 <- krige(formula=z~1, locations=NULL, newdata=siatka, dummy=TRUE,
                     beta=1, model=vgm(psill=0.025,model='Exp',range=1500), nsim=4, nmax=30)

## [using unconditional Gaussian simulation]

spplot(sym_bezw2, main='Przestrzennie skorelowana powierzchnia \nśrednia=1, sill=0.025, zasięg=1500, model wyk..adniczy')

```

**Przestrzennie skorelowana powierzchnia  
..rednia=1, sill=0.025, zasi..g=1500, model wyk..adniczy**

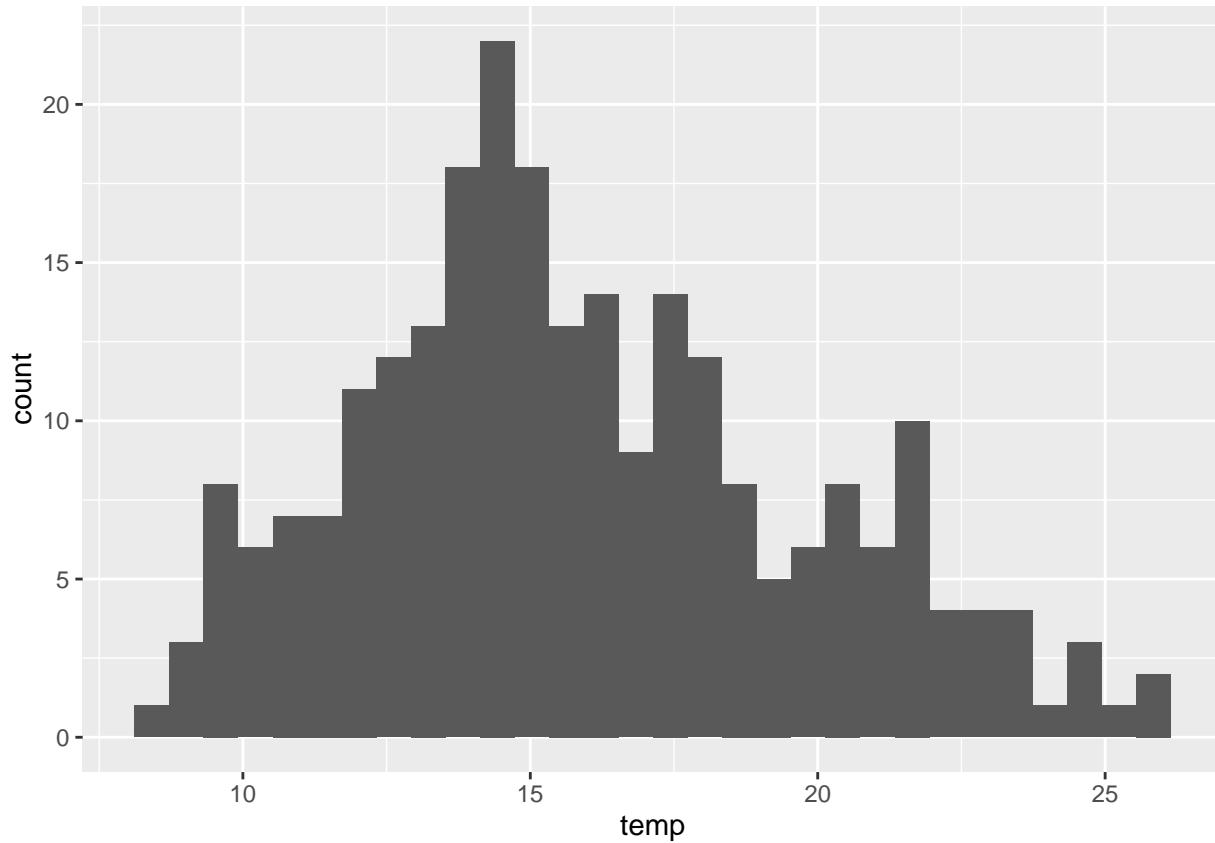


## 13.4 Symulacje warunkowe

### 13.4.1 Sekwencyjna symulacja gaussowska (ang. *Sequential Gaussian simulation*)

1. Wybranie lokalizacji nie posiadającej zmierzonej wartości badanej zmiennej
2. Kriging wartości tej lokalizacji korzystając z dostępnych danych, co pozwala na uzyskanie rozkładu prawdopodobieństwa badanej zmiennej
3. Wylosowanie wartości z rozkładu prawdopodobieństwa za pomocą generatora liczb losowych i przypisanie tej wartości do lokalizacji
4. Dodanie symulowanej wartości do zbioru danych i przejście do kolejnej lokalizacji
5. Powtórzenie poprzednich kroków, aż do momentu gdy nie pozostanie już żadna nieokreślona lokalizacja

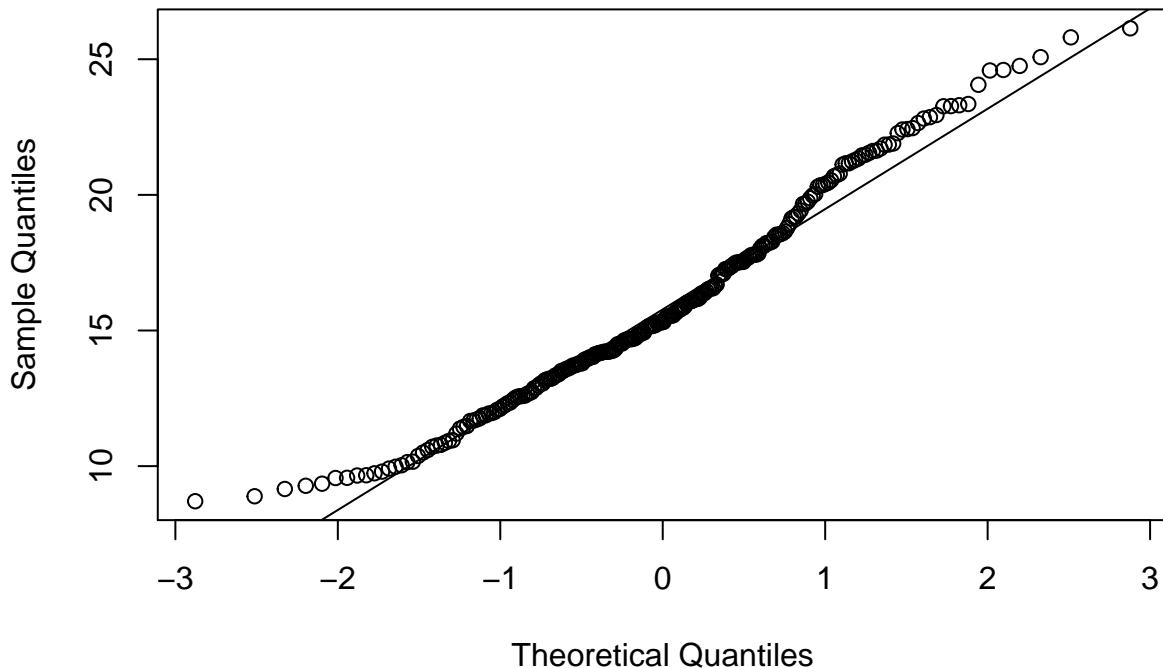
```
ggplot(punkty@data, aes(temp)) + geom_histogram()
```



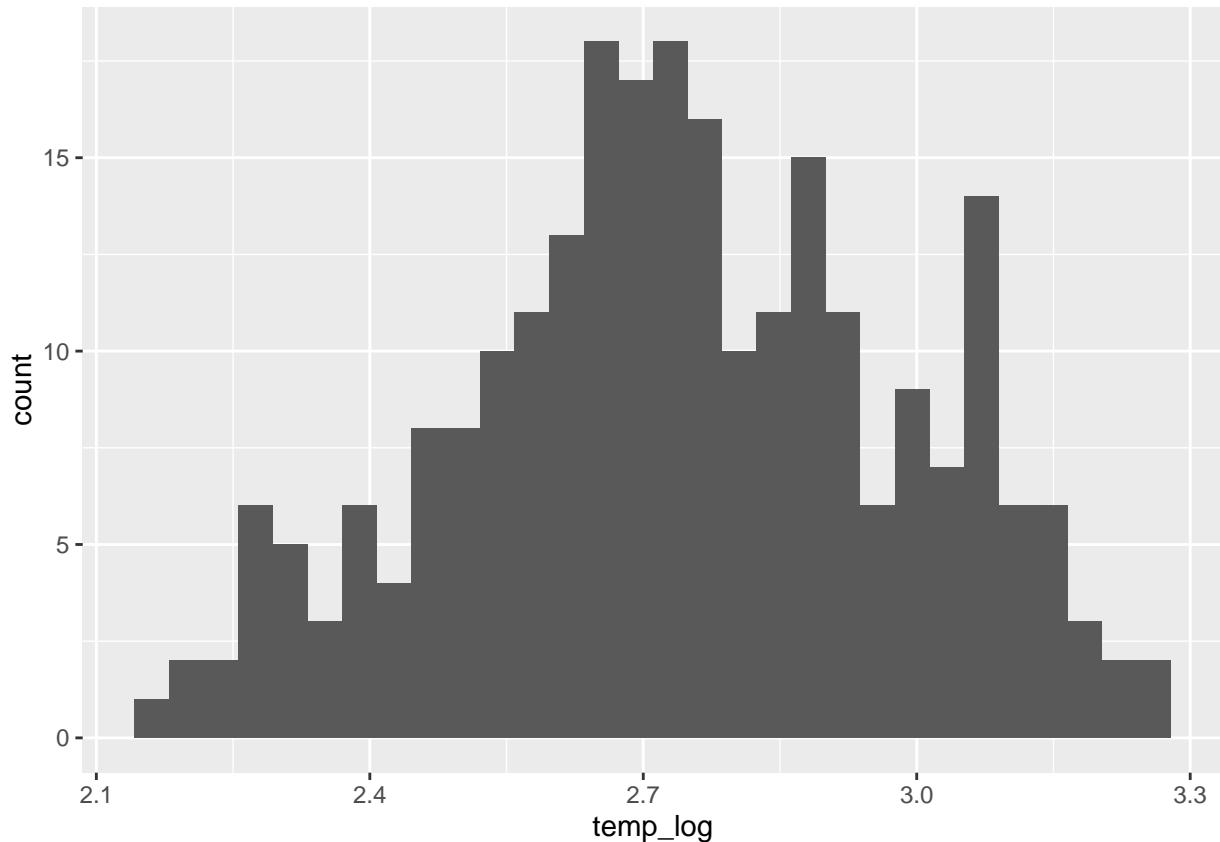
```
shapiro.test(punkty$temp)
```

```
##  
## Shapiro-Wilk normality test  
##  
## data: punkty$temp  
## W = 0.97683, p-value = 0.0004194
```

```
qqnorm(punkty$temp); qqline(punkty$temp)
```

**Normal Q-Q Plot**

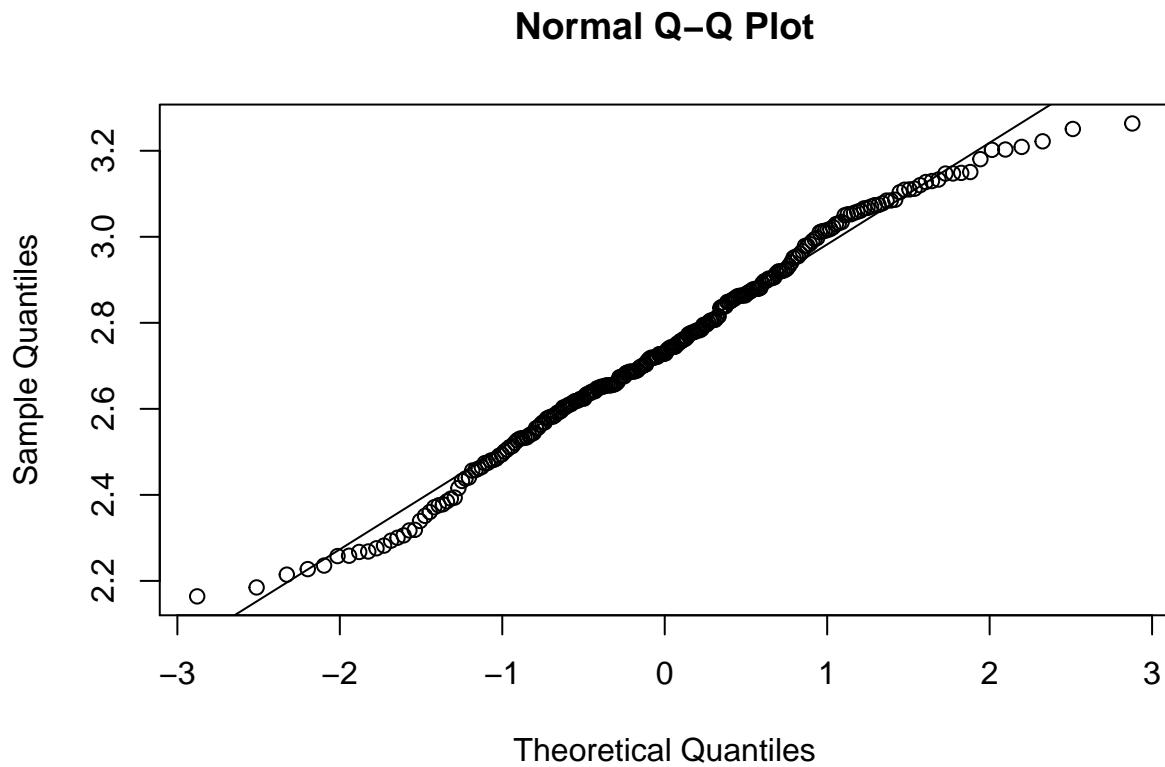
```
punkty$temp_log <- log(punkty$temp)
ggplot(punkty@data, aes(temp_log)) + geom_histogram()
```



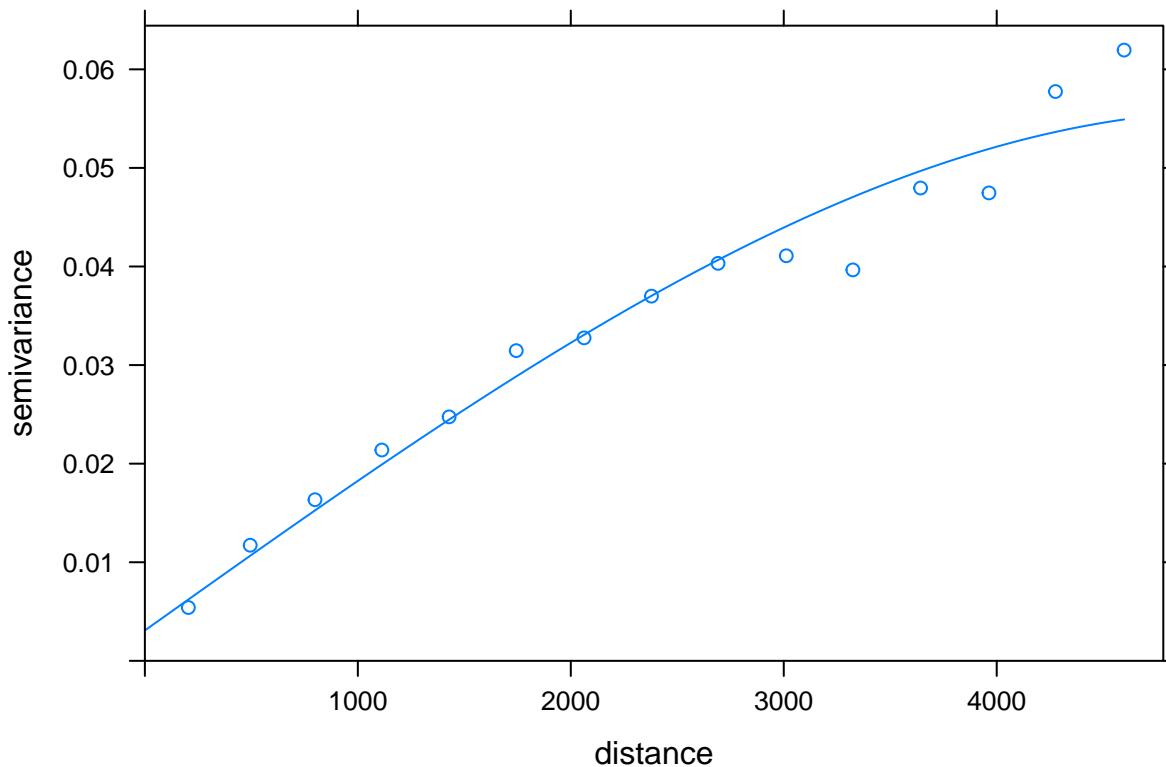
```
shapiro.test(punkty$temp_log)
```

```
##  
## Shapiro-Wilk normality test  
##  
## data: punkty$temp_log  
## W = 0.98907, p-value = 0.05568
```

```
qqnorm(punkty$temp_log);qqline(punkty$temp_log)
```



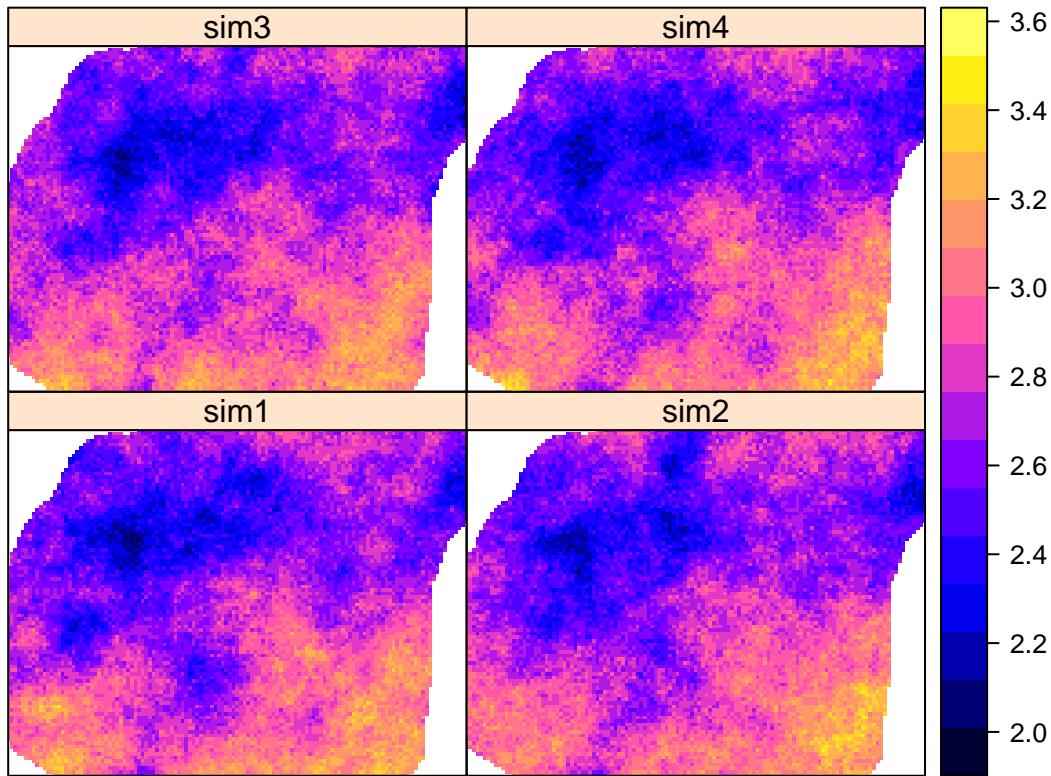
```
vario <- variogram(temp_log~1, punkty)  
model <- vgm(0.05, model = 'Sph', range = 4500, nugget=0.005)  
fitted <- fit.variogram(vario, model)  
plot(vario, model=fitted)
```



```
sym_ok <- krige(temp_log~1, punkty, siatka, model=fitted, nsim=4, nmax=30)
```

```
## drawing 4 GLS realisations of beta...
## [using conditional Gaussian simulation]
```

```
spplot(sym_ok)
```



```
summary(sym_ok)
```

```
## Object of class SpatialPixelsDataFrame
## Coordinates:
##      min     max
## x 745586.7 756926.7
## y 712661.2 721211.2
## Is projected: TRUE
## proj4string :
## [+init=epsg:2180 +proj=tmerc +lat_0=0 +lon_0=19 +k=0.9993
## +x_0=500000 +y_0=-5300000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0
## +units=m +no_defs]
## Number of points: 10993
## Grid attributes:
##   cellcentre.offset cellsize cells.dim
## s1      745586.7      90      127
## s2      712661.2      90      96
## Data attributes:
##       sim1        sim2        sim3        sim4
## Min. :2.008  Min. :2.067  Min. :2.008  Min. :2.014
## 1st Qu.:2.548 1st Qu.:2.563 1st Qu.:2.567 1st Qu.:2.548
## Median :2.710  Median :2.730  Median :2.730  Median :2.715
## Mean   :2.729  Mean   :2.740  Mean   :2.734  Mean   :2.730
## 3rd Qu.:2.915 3rd Qu.:2.919 3rd Qu.:2.894 3rd Qu.:2.912
## Max.   :3.425  Max.   :3.506  Max.   :3.361  Max.   :3.525
```

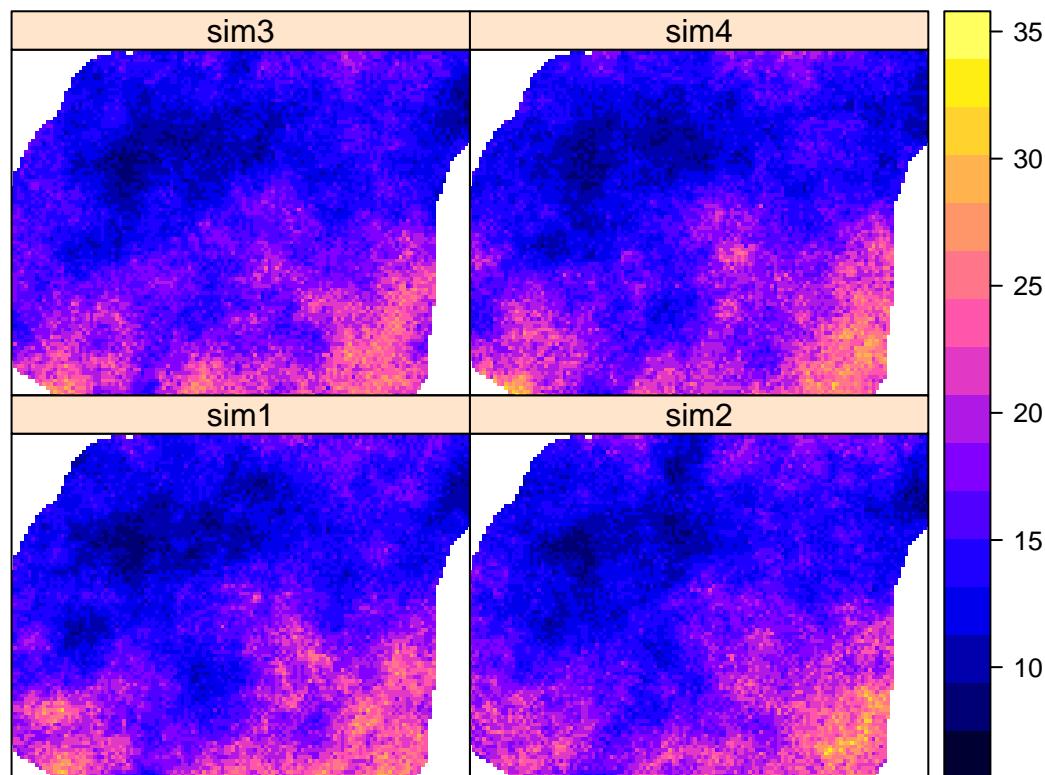
```

sym_ok@data <- as.data.frame(apply(sym_ok@data, 2, exp))
summary(sym_ok)

## Object of class SpatialPixelsDataFrame
## Coordinates:
##      min     max
## x 745586.7 756926.7
## y 712661.2 721211.2
## Is projected: TRUE
## proj4string :
## [+init=epsg:2180 +proj=tmerc +lat_0=0 +lon_0=19 +k=0.9993
## +x_0=500000 +y_0=-5300000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0
## +units=m +no_defs]
## Number of points: 10993
## Grid attributes:
##   cellcentre.offset cellsize cells.dim
## s1          745586.7    90      127
## s2          712661.2    90      96
## Data attributes:
##       sim1           sim2           sim3           sim4
## Min.   : 7.445   Min.   : 7.902   Min.   : 7.45   Min.   : 7.493
## 1st Qu.:12.787  1st Qu.:12.973  1st Qu.:13.02  1st Qu.:12.785
## Median  :15.035  Median  :15.329  Median  :15.33  Median  :15.107
## Mean    :15.814  Mean    :15.961  Mean    :15.83  Mean    :15.819
## 3rd Qu.:18.443  3rd Qu.:18.520  3rd Qu.:18.07  3rd Qu.:18.392
## Max.    :30.714  Max.    :33.311  Max.    :28.82  Max.    :33.940

```

```
spplot(sym_ok)
```



```
sym_sk <- krige(temp_log~1, punkty, siatka, model=fitted, beta=2.7, nsim=100, nmax=30)
```

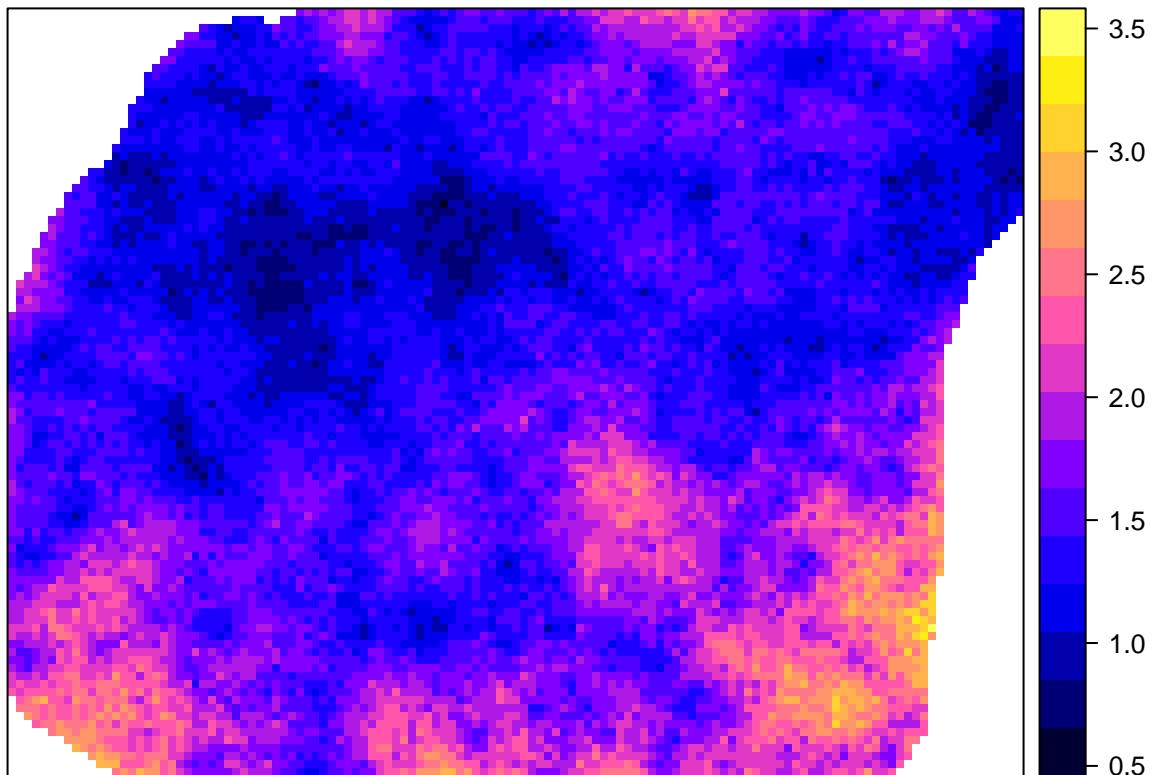
```
## [using conditional Gaussian simulation]
```

```
sym_sk@data <- as.data.frame(apply(sym_sk@data, 2, exp))
```

```
sym_sk <- stack(sym_sk)
```

```
sym_sk_sd <- calc(sym_sk, fun = sd)
```

```
spplot(sym_sk_sd)
```



## 13.5 Sekwencyjna symulacja danych kodowanych

### 13.5.1 Sekwencyjna symulacja danych kodowanych (ang. *Sequential indicator simulation*)

```
summary(punkty$temp)
```

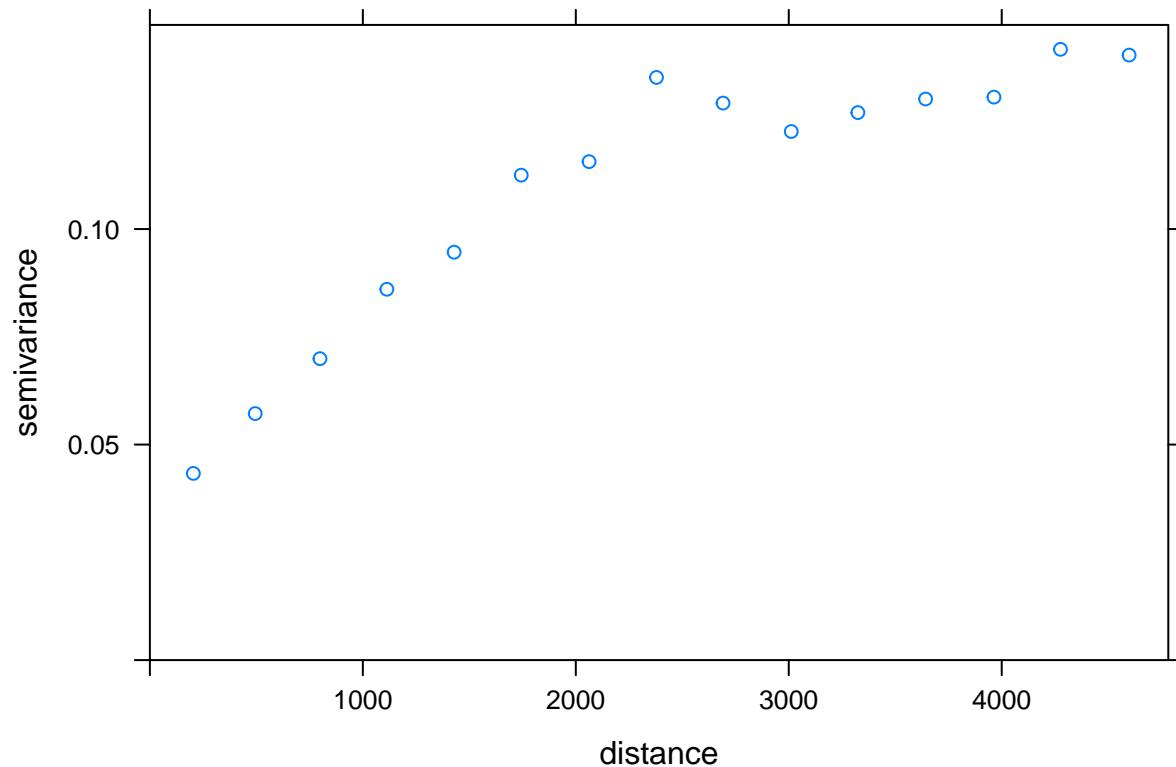
```
##      Min. 1st Qu. Median     Mean 3rd Qu.    Max.
## 8.706 13.280 15.310 15.950 18.270 26.140
```

```
punkty$temp_ind <- punkty$temp < 12
```

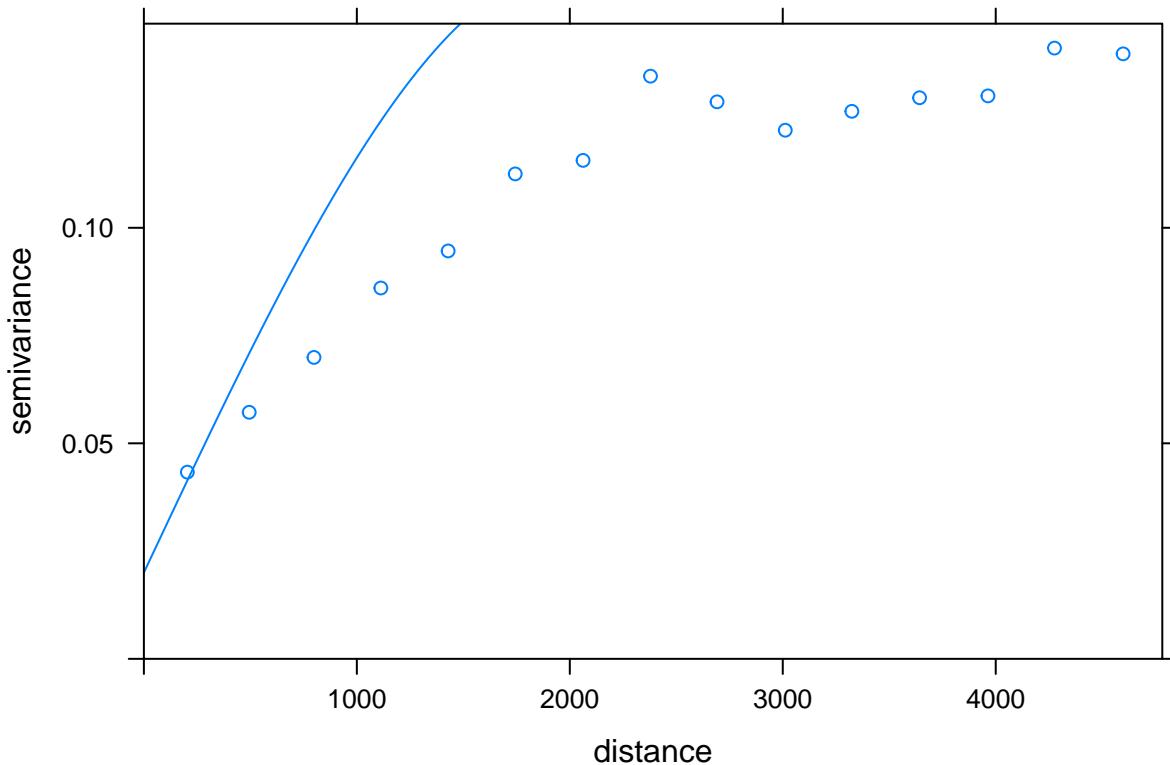
```
summary(punkty$temp_ind)
```

```
##      Mode    FALSE     TRUE    NA's
## logical      212      38       0
```

```
vario_ind <- variogram(temp_ind~1, punkty)
plot(vario_ind)
```



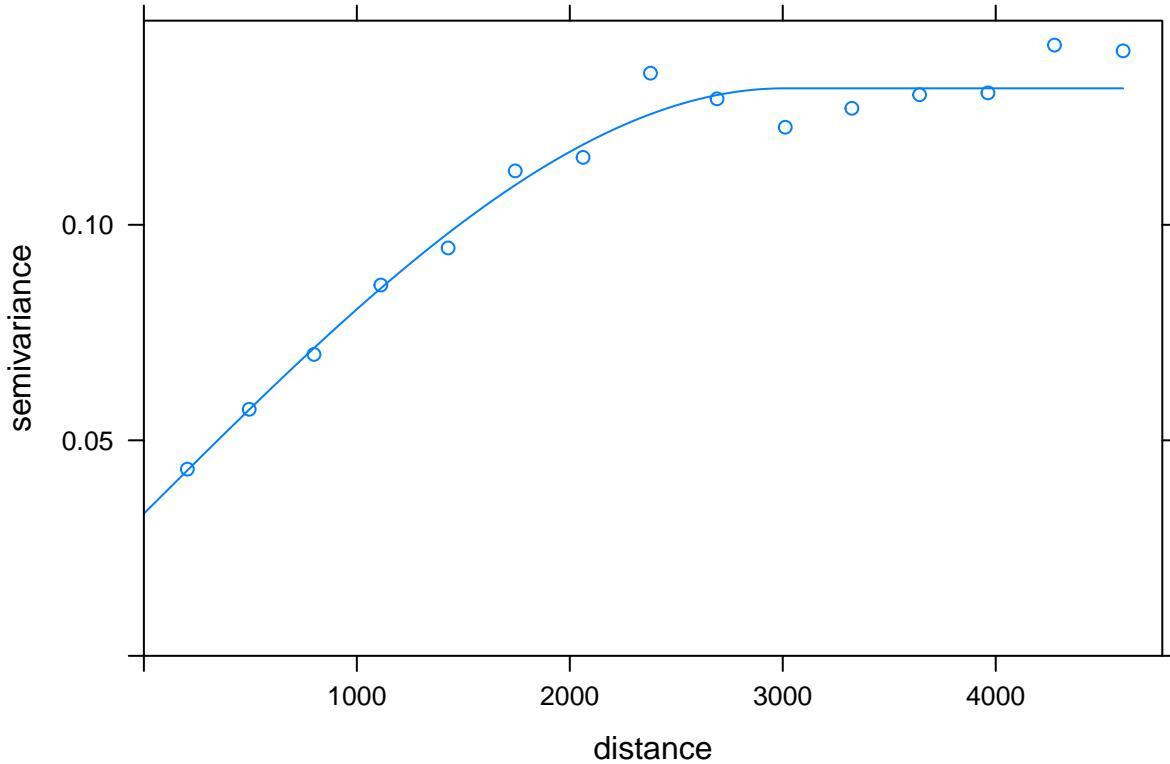
```
model_ind <- vgm(0.14, model = 'Sph', range = 2000, nugget = 0.02)
plot(vario_ind, model=model_ind)
```



```
fitted_ind <- fit.variogram(vario_ind, model_ind)  
fitted_ind
```

```
##   model      psill     range  
## 1   Nug 0.03299461 0.000  
## 2   Sph 0.09866551 3006.403
```

```
plot(vario_ind, model=fitted_ind)
```

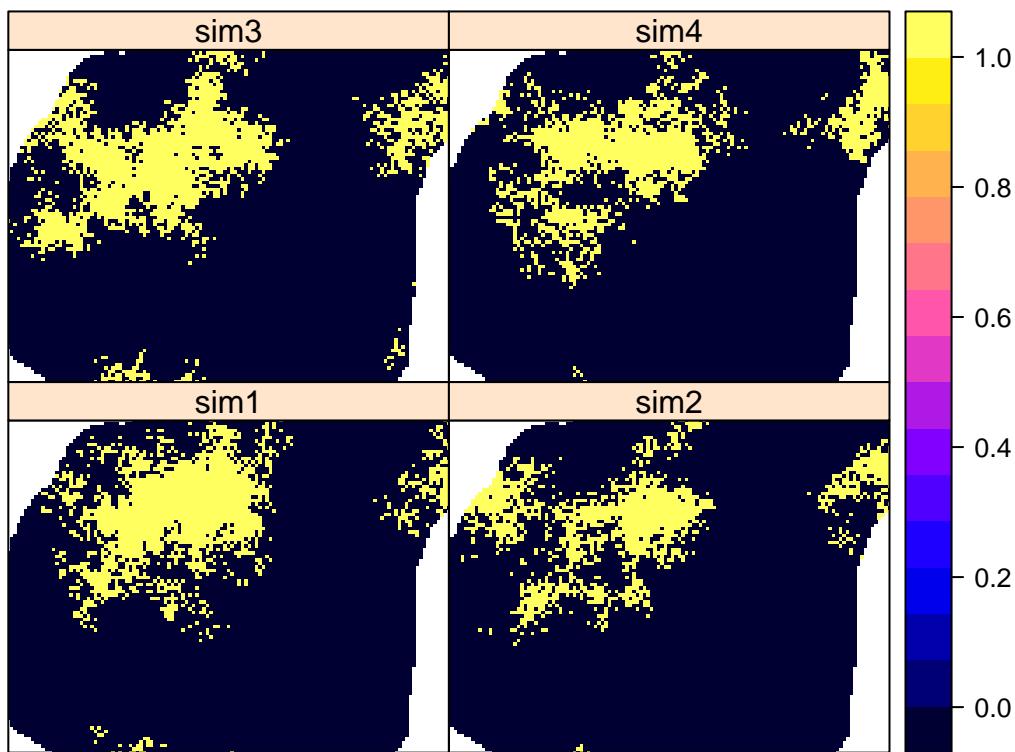


```
sym_ind <- krige(temp_ind~1, punkty, siatka, model=fitted_ind, indicators=TRUE, nsim=4, nmax=30)
```

```
## drawing 4 GLS realisations of beta...
## [using conditional indicator simulation]
```

```
spplot(sym_ind, main='Symulacje warunkowe')
```

## Symulacje warunkowe



- An Introduction to R - oficjalne wprowadzenie do R
- Przewodnik po pakiecie R, Programowanie w języku R, Statystyczna analiza danych z wykorzystaniem programu R - polskie wydawnictwa
- Applied Spatial Dala Analysis with R
- A Practical Guide to Geostatistical Mapping
- gstat user's manual
- CRAN Task View: Analysis of Spatial Data
- Applied Geostatistics, Statistics for spatial data
- Praktyczny poradnik - jak szybko zrozumieć i wykorzystać geostatystykę w pracy badawczej
- Wyszukiwarki internetowe Rseek, Duckduckgo, Google, itd.

# Bibliography