Topological Data Analysis Ball Mapper in R

Simon Rudkin – University of Manchester

20th Summer School in Risk, Finance and Stochastics

Authors

Prof Pawel Dlotko
Dioscuri Centre in Topological Data Analysis, IMPAN
Polish Academy of Sciences

Dr Simon Rudkin
Institute of Data Science and Artificial Intelligence
University of Manchester

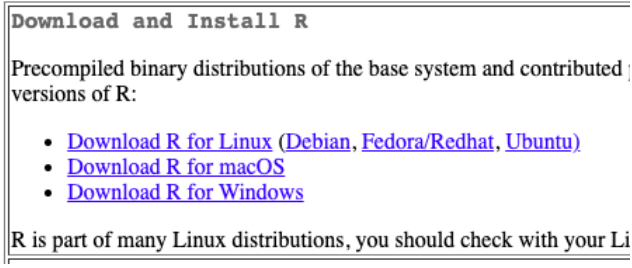
Dr Wanling Rudkin
University of Exeter Business School
University of Exeter

In this Presentation...

- R and R studio
- Introduction to *datasauRus* package
- Introduction to TDABM
- Exercises on the datasaurus
- Financial market data
- Exercises on the financial market data

Code to accompany this session is available via All files relating to this session on GitHub https://github.com/srudkin12/RFS2023

Download R



Download and Install R

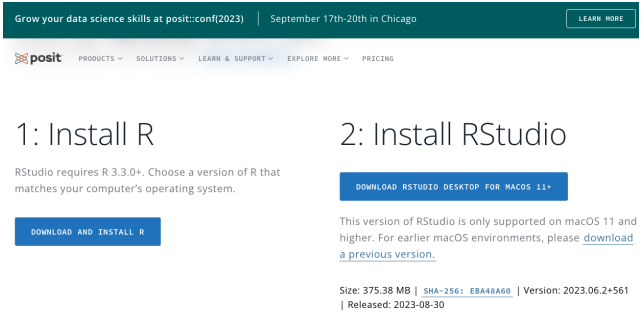
Precompiled binary distributions of the base system and contributed
versions of R:

- Download R for Linux (Debian, Fedora/Redhat, Ubuntu)
- Download R for macOS
- Download R for Windows

R is part of many Linux distributions, you should check with your Li

Download R studio from https://cran.r-project.org/

Download R Studio



Download R studio from https://posit.co/download/rstudio-desktop/

- Note that R Studio is not produced by R
- Many prefer R studio because of windows environment
- Packages for TDABM are also available in Python
- We are working on updates for all packages, please keep an eye on the Dioscuri centre website: https://dioscuri-tda.org/

Key Arguments

Visualising data is an essential phase of the modelling process

Humans cannot see in multiple dimensions - dimension reduction

Mapping helps rationalise space - look to map our data

Dimensionality considered in time series and the cross-section

Anscombe (1973) and Visualisation

Graphs in Statistical Analysis*

F. J. ANSCOMBE**

Graphs are essential to good statistical analysis. Ordinary scatterplots and "triple" scatterplots are discussed in relation to regression analysis.

1. *Usefulness of graphs*

Most textbooks on statistical methods, and most statistical computer programs, pay too little attention to graphs. Few of us escape being indoctrinated with these notions:

(1) numerical calculations are exact, but graphs are rough;

(2) for any particular kind of statistical data there is just one set of calculations constituting a correct statistical analysis;

(3) performing intricate calculations is virtuous, whereas actually looking at the data is cheating.

A computer should make *both* calculations *and* graphs. Both sorts of output should be studied; each will contribute to understanding.

through the computer. The analysis should be sensitive both to peculiar features in the given numbers and also to whatever background information is available about the variables. The latter is particularly helpful in suggesting alternative ways of setting up the analysis.

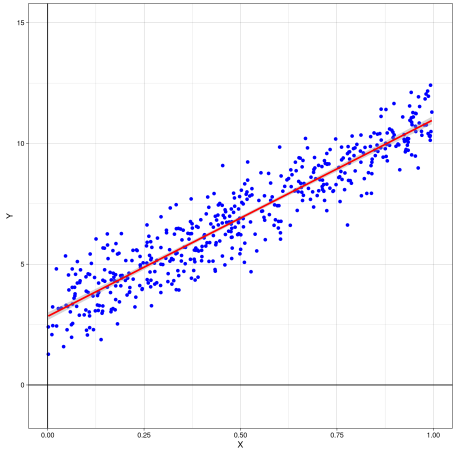
Thought and ingenuity devoted to devising good graphs are likely to pay off. Many ideas can be gleaned from the literature, of which a sampling is listed at the end of this paper. In particular, Tukey [7, 8] has much to say on the topics presented here.

A few simple types of statistical analysis are now considered.

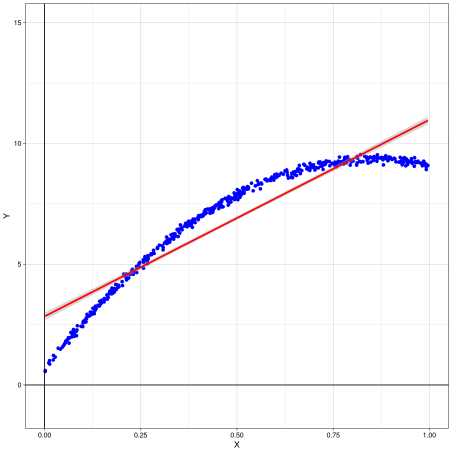
2. *Regression analysis—the simplest case*

Suppose we have values for one "dependent" variable y and one "independent" (exogenous, predictor)

Anscombe's Quartet 1

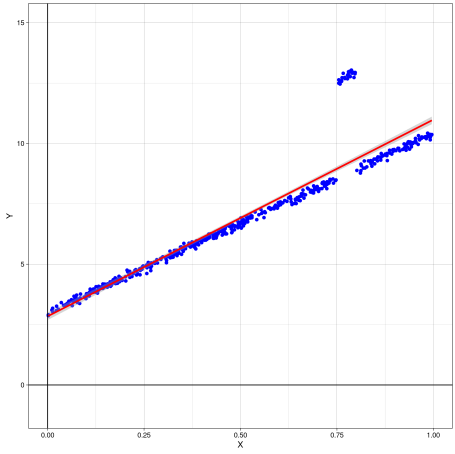


(a) Standard Data

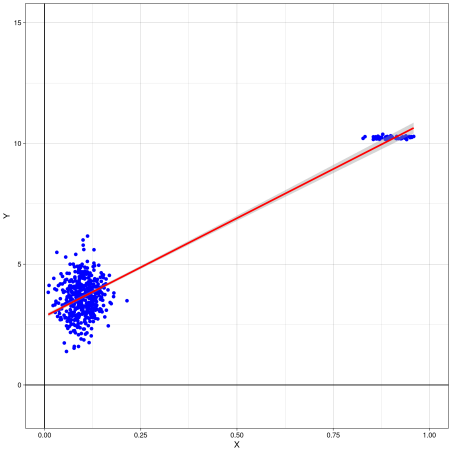


(b) Quadratic

Anscombe's Quartet 2



(c) Outlier



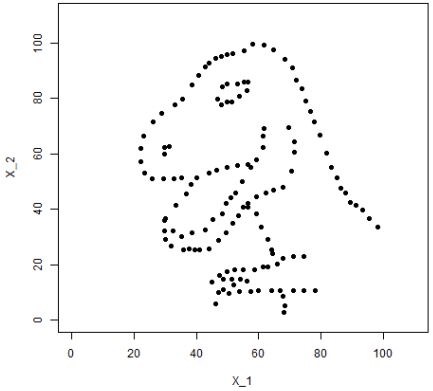
(d) High Leverage Point

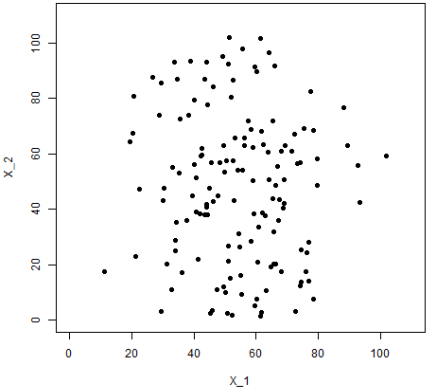
DatasauRus (Matejka and Fitzmaurice, 2017)

Sketch a scatterplot of two variables X and Y using the following information:

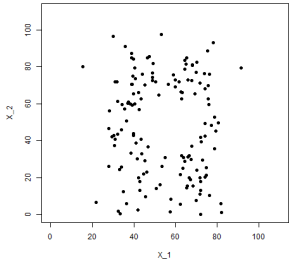
- Mean of X is 54.27
- Mean of Y is 47.83
- Standard deviation of X is 16.77
- Standard deviation of Y is 26.94
- Correlation between X and Y is -0.064
- All values are in the range 0 to 100

DatasauRus Answer?

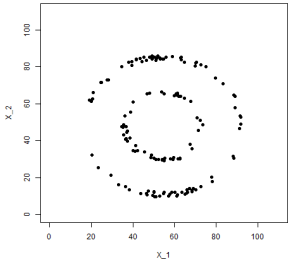




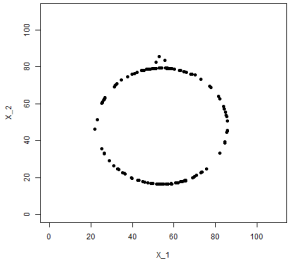
DatasauRus 2



(a) Away

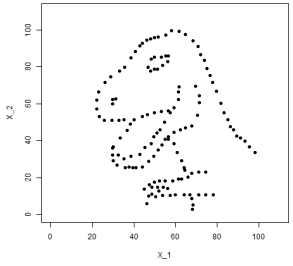


(b) Bullseye

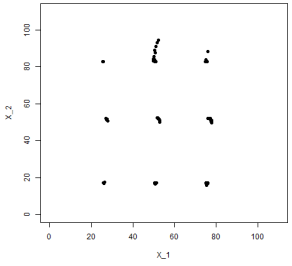


(c) Circle

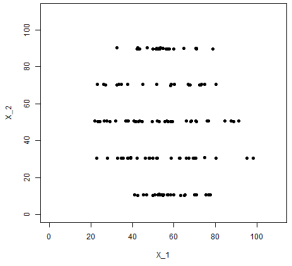
DatasauRus 2



(d) Dino

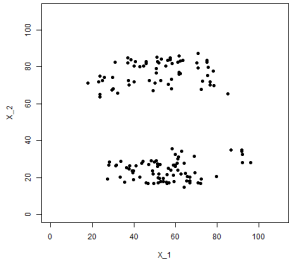


(e) Dots

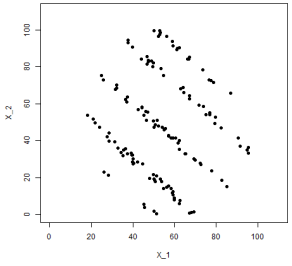


(f) H Lines

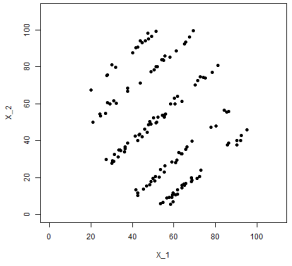
DatasauRus 2



(g) High Lines

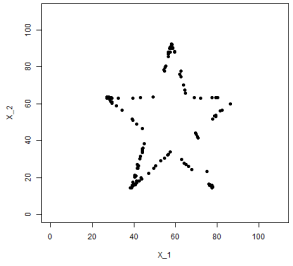


(h) Diagonal Down

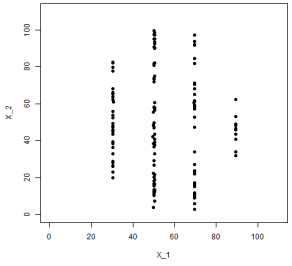


(i) Diagonal Up

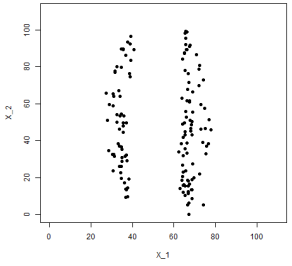
DatasauRus 2



(j) Star

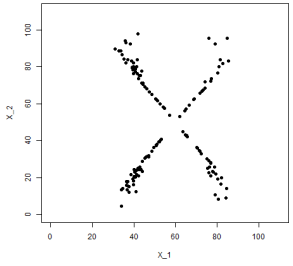


(k) V Lines

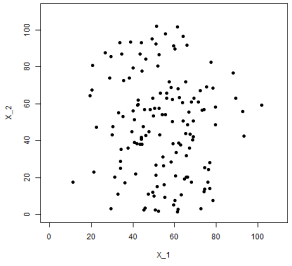


(l) Wide Lines

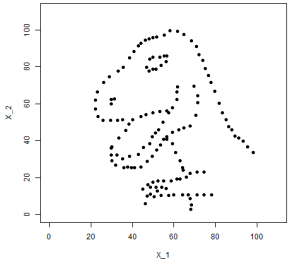
DatasauRus 2



(m) X

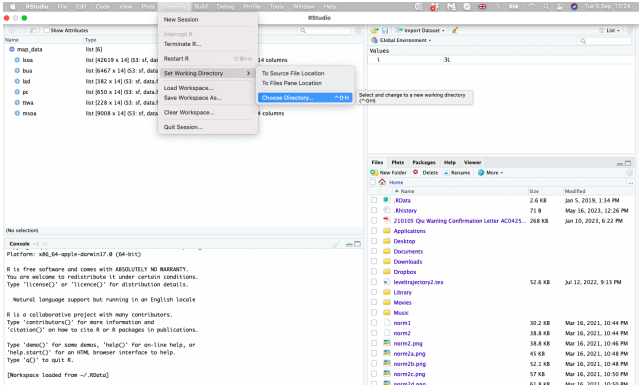


(n) Normal



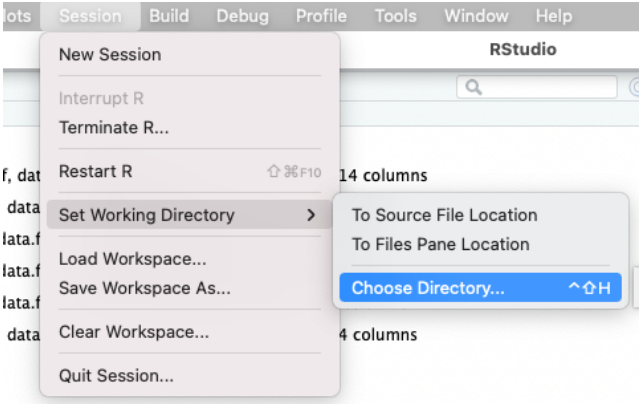
(o) Dino

Introduction to R Studio



- Session is illustrated with R Studio
- Code will work with R
- Use terminal window (bottom left) to input code for all elements

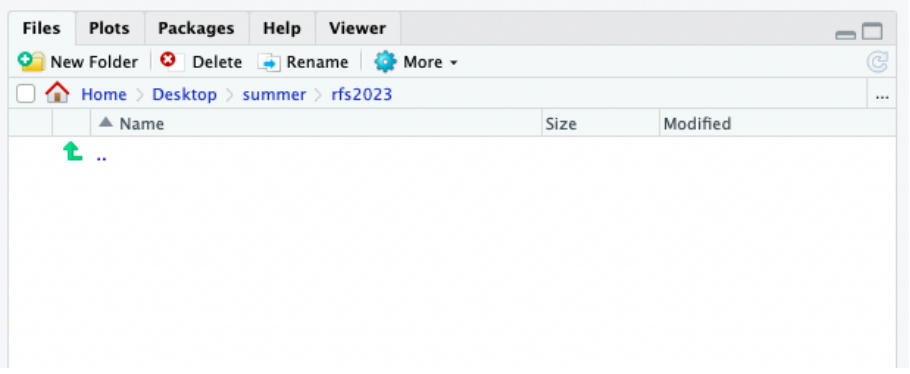
Introduction to R Studio 2



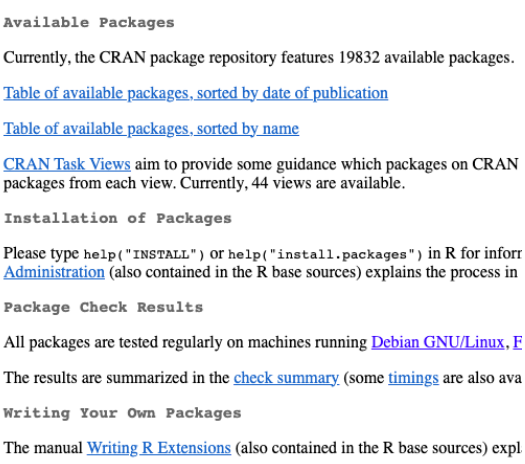
- Set working directory - choose carefully
- Keep all files within that folder

```
setwd("C://rfs2023/")
```

Introduction to R Studio 3



Introduction to Packages in R



Available Packages

Currently, the CRAN package repository features 19832 available packages.

Table of available packages, sorted by date of publication

Table of available packages, sorted by name

CRAN Task Views aim to provide some guidance which packages on CRAN packages from each view. Currently, 44 views are available.

Installation of Packages

Please type `help("INSTALL")` or `help("install.packages")` in R for inforr Administration (also contained in the R base sources) explains the process in

Package Check Results

All packages are tested regularly on machines running Debian GNU/Linux, F

The results are summarized in the check summary (some timings are also ava

Writing Your Own Packages

The manual Writing R Extensions (also contained in the R base sources) expl

```
install.packages(‘‘BallMapper’’)
```

```
library(BallMapper)
```

Note that there are no “ ” marks around the library command

datasauRus package Davies et al. (2022)

- Includes 12 datasets - " datasaurus dozen"
- Additionally includes the "Dino" dataset
- Our use will follow vignette at https://cran.r-project.org/web/packages/datasauRus/vignettes/Datasaurus.html

R Code File on GitHub

```
# Code to accompany RFS2023 session

# Code with # will not run. These are instructions or information

# Download either R https://cran.r-project.org/ or R Studio (https://posit.co/download/rstudio-desktop/) in order to u

# Set working directory

setwd("C://rfs2023/") # This line must be changed to your working directory, or set using the menus in R

# Load packages (The first time you use the packages you may need to use the install lines)

# install.packages("BallMapper") # Only use this if you need to install. Remove the # at the start of the line to run.
# install.packages("ggplot2") # Only use this if you need to install. Remove the # at the start of the line to run.
# install.packages("datasauRus") # Only use this if you need to install. Remove the # at the start of the line to run.
# install.packages("dplyr") # Only use this if you need to install. Remove the # at the start of the line to run.
# install.packages("crypto2") # Only use this if you need to install. Remove the # at the start of the line to run.


library(BallMapper)
library(ggplot2)
library(datasauRus)
library(dplyr)
```

R Code File in GitHub 2

```
library(BallMapper)
library(ggplot2)
library(datasauRus)
library(dplyr)

# First exercise is to verify the summary statisti

if(requireNamespace("dplyr")){
  suppressPackageStartupMessages(library(dplyr))
  datasaurus_dozen %>%
    group_by(dataset) %>%
    summarize(
      mean_x    = mean(x),
      mean_y    = mean(y),
      std_dev_x = sd(x),
      std_dev_y = sd(y),
      corr_x_y  = cor(x, y)
    )
}
```

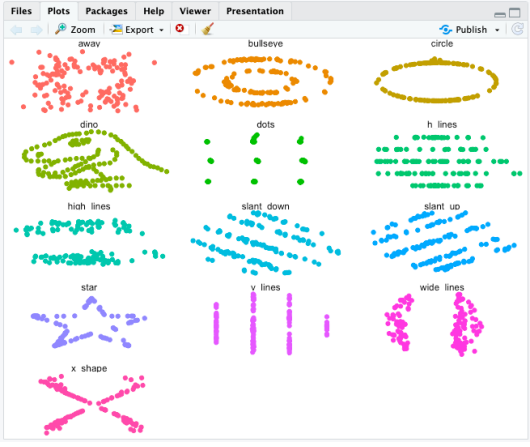
- All Code is available on GitHub
- This section follows vignette
- **Please do not try to copy from the slides**

R Code File on GitHub 3

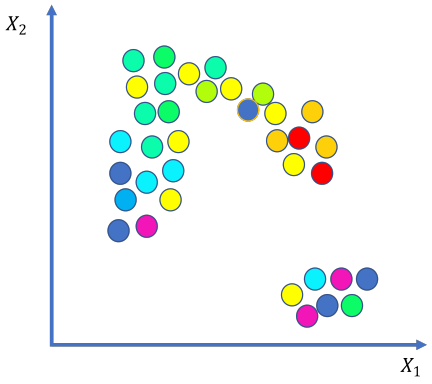
```
# Next let us plot the datasauRus data

if(requireNamespace("ggplot2")){
  library(ggplot2)
  ggplot(datasaurus_dozen, aes(x = x, y = y, colour = dataset))+
    geom_point()+
    theme_void()+
    theme(legend.position = "none")+
    facet_wrap(~dataset, ncol = 3)
}
```

DatasauRus Solutions

```
# A tibble: 13 × 6
   dataset    mean_x mean_y std_dev_x std_dev_y corr_x_y
   <chr>       <dbl>  <dbl>     <dbl>     <dbl>    <dbl>
 1 away         54.3   47.8      16.8      26.9  -0.0641
 2 bullseye     54.3   47.8      16.8      26.9  -0.0686
 3 circle       54.3   47.8      16.8      26.9  -0.0683
 4 dino         54.3   47.8      16.8      26.9  -0.0645
 5 dots         54.3   47.8      16.8      26.9  -0.0603
 6 h_lines      54.3   47.8      16.8      26.9  -0.0617
 7 high_lines   54.3   47.8      16.8      26.9  -0.0685
 8 slant_down   54.3   47.8      16.8      26.9  -0.0690
 9 slant_up     54.3   47.8      16.8      26.9  -0.0686
10 star         54.3   47.8      16.8      26.9  -0.0630
11 v_lines      54.3   47.8      16.8      26.9  -0.0694
12 wide_lines   54.3   47.8      16.8      26.9  -0.0666
13 x_shape      54.3   47.8      16.8      26.9  -0.0656
```



TDA Ball Mapper Overview



- Cover with balls of fixed radius
- Choose a point at random
- Draw a ball of fixed radius ϵ
- Points within ball are covered
- Continue to cover drawing around random uncovered points
- TDABM knows how many points are in each ball

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDA Ball Mapper Overview

- Cover with balls of fixed radius
- Ball colour is function on the members - colour according to average outcome of members, standard deviation of outcomes etc.
- Edges show points in overlap of balls
- Resize balls according to number of points contained within - indicative of density of space
- Remove information to leave abstract representation

TDABM in R

```
bm1<-BallMapper(<data>,<colour>,<radius>)
```

- Use package *BallMapper* (Dlotko, 2019)
- Create a BallMapper object (`bm1`)
- `<data>` is a `data.frame` object with axis variables
- `<colour>` is a `data.frame` object with colouration variable
- `<radius>` is a numeric value for the ball radius ϵ

TDABM in R 2

```
ColorIgraphPlot(<bmobject>,<seed_for_plotting>)
```

- <bmobject> is a BallMapper object generated using the BallMapper function
- <seed_for_plotting> is the plotting seed
- Recall TDABM produces abstract visualisation so you can play with the appearance to help interpretability

Preparing DatasauRus Data

```
head(datasaurus_dozen_wide)
```

```
> head(datasaurus_dozen_wide)
# A tibble: 6 × 26
  away_x away_y bullseye_x bullseye_y circle_x circle_y dino_x dino_y dots_x dots_y h_lines_x
   <dbl>  <dbl>      <dbl>      <dbl>    <dbl>    <dbl>  <dbl>  <dbl>  <dbl>  <dbl>      <dbl>
1   32.3   61.4       51.2       83.3     56.0     79.3   55.4   97.2   51.1   90.9       53.4
2   53.4   26.2       59.0       85.5     50.0     79.0   51.5   96.0   50.5   89.1       52.8
3   63.9   30.8       51.9       85.8     51.3     82.4   46.2   94.5   50.2   85.5       47.1
4   70.3   82.5       48.2       85.0     51.2     79.2   42.8   91.4   50.1   83.1       42.4
5   34.1   45.7       41.7       84.0     44.4     78.2   40.8   88.3   50.6   82.9       42.7
6   67.7   37.1       37.9       82.6     45.0     77.9   38.7   84.9   50.3   83.0       32.4
# ℹ 15 more variables: h_lines_y <dbl>, high_lines_x <dbl>, high_lines_y <dbl>,
#   slant_down_x <dbl>, slant_down_y <dbl>, slant_up_x <dbl>, slant_up_y <dbl>, star_x <dbl>,
#   star_y <dbl>, v_lines_x <dbl>, v_lines_y <dbl>, wide_lines_x <dbl>, wide_lines_y <dbl>,
#   x_shape_x <dbl>, x_shape_y <dbl>
```

Preparing DatasauRus Data 2

```
away<-as.data.frame(cbind.data.frame(datasaurus_dozen_wide$away_x,
                    datasaurus_dozen_wide $away_y))
```

- Code is one line - please use code file
- We then give names to the dataset

```
names(away)<-c("x","y")
```

ggplot DatasauRus

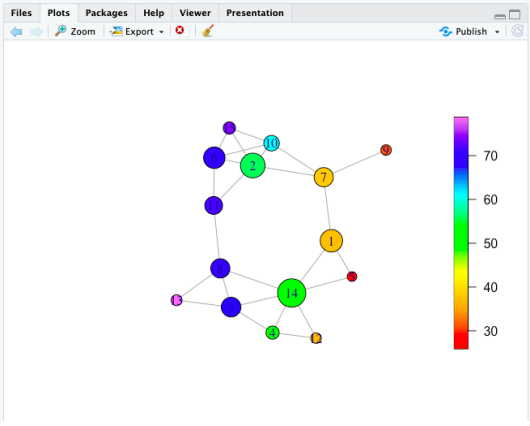
```
gaway <-ggplot(away,aes(x=x,y=y))+

  geom_hline(yintercept=0,color="black",lwd =1.5,linetype="solid")+
  geom_vline(xintercept=0,color="black",lwd =1.5,linetype="solid")+
   geom_point(color="blue",size=2) +
  labs(x="X",y="Y") +
  theme(axis.title.x = element_text(vjust=0,size=20),
  axis.title.y = element_text(vjust=0,size=20),
  axis.text = element_text(size=20),
  panel.grid.major = element_line(color="gray10", linetype = "dashed",  size = .5),
  panel.grid.minor = element_line(color="gray90", linetype = "dashed", size = .8))

gaway # Plot the ggplot object

ggsave("awayscatter.png",width=6,height=6) # Save the ggplot

# For TDABM we need the colour to be a dataframe
```

TDABM DatasauRus

```
# For TDABM we need the colour to be a dataframe

awayx<-as.data.frame(away$x)

# Now we create the TDABM plot

bmaway<-BallMapper(away,awayx,20)
ColorIgraphPlot(bmaway)

png("awaybm20.png")
ColorIgraphPlot(bmaway)
dev.off()
```

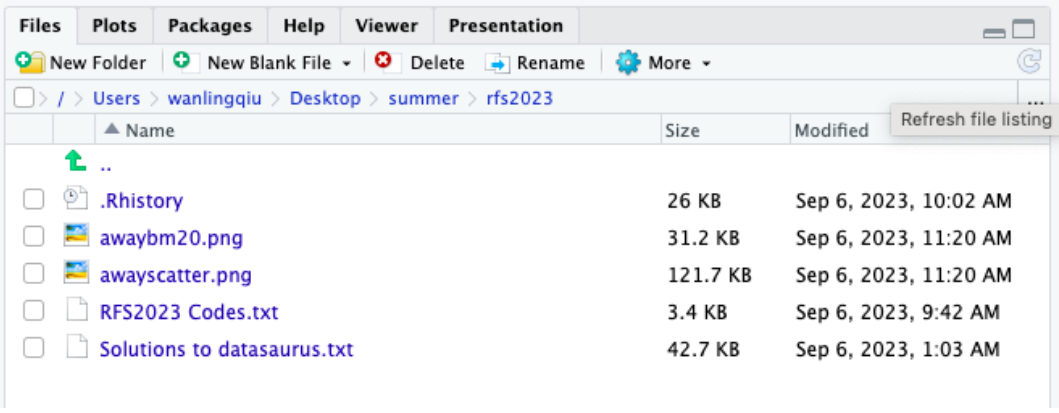
- Define `data.frame` for colouration
- `BallMapper` command generates object
- `ColorIgraphPlot` to visualise
- Final block to save as `.png` file

TDABM DatasauRus 2

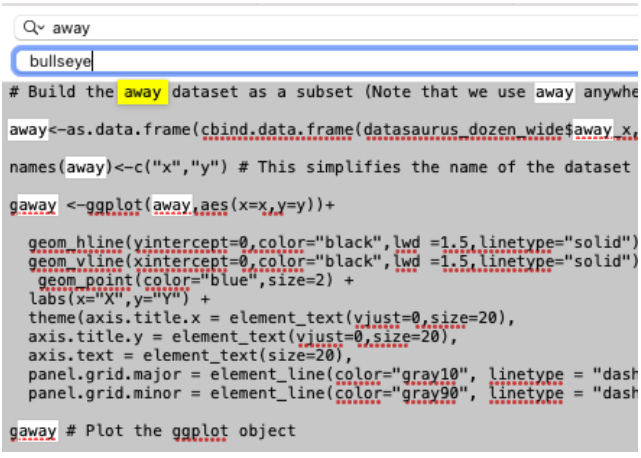


- Hole in the middle of the data is still visible
- Can see the hole in the middle
- Colours show us where X is low
- Some systems may pop the plot up in a new window

TDABM DatasauRus 3



Replace All

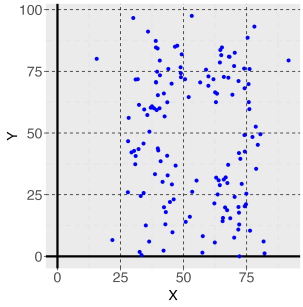

away
bullseye
# Build the away dataset as a subset (Note that we use away anywhe
away<-as.data.frame(cbind.data.frame(datasaurus_dozen_wide$away_x,
names(away)<-c("x","y") # This simplifies the name of the dataset
gaway <-ggplot(away,aes(x=x,y=y))+
geom_hline(yintercept=0,color="black",lwd =1.5,linetype="solid")
geom_vline(xintercept=0,color="black",lwd =1.5,linetype="solid")
geom_point(color="blue",size=2) +
labs(x="X",y="Y") +
theme(axis.title.x = element_text(vjust=0,size=20),
axis.title.y = element_text(vjust=0,size=20),
axis.text = element_text(size=20),
panel.grid.major = element_line(color="gray10", linetype = "dash
panel.grid.minor = element_line(color="gray90", linetype = "dash
gaway # Plot the ggplot object

DatasauRus Ball Mapper

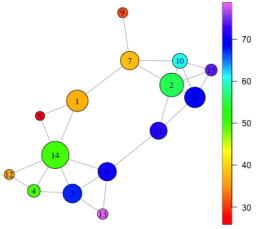
- Generate TDABM plots with radius 20 for the remaining 12 datasets
- What interesting patterns do you notice?
- What happens if you increase the radius to 30?
- What happens if you reduce the radius to 10?

Hint: You can use replace all to replace the eps=20 element. You should also change the filenames by changing the 20.png to an appropriate name

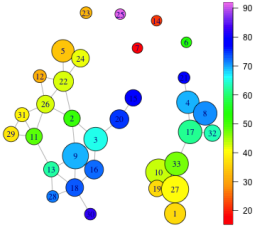
Example TDABM DatasauRus plots: Away



(a) Scatter

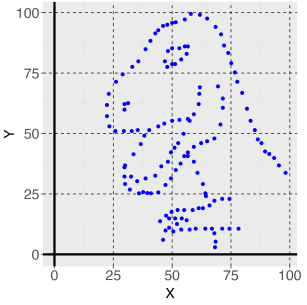


(b) TDABM $\epsilon = 20$

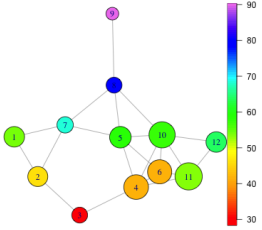


(c) TDABM $\epsilon = 10$

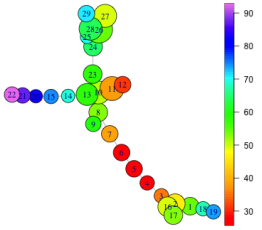
Example TDABM DatasauRus plots: Dino



(a) Scatter

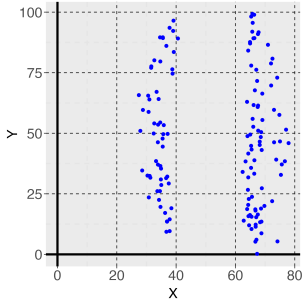


(b) TDABM $\epsilon = 20$

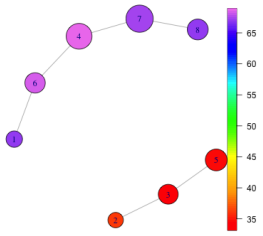


(c) TDABM $\epsilon = 10$

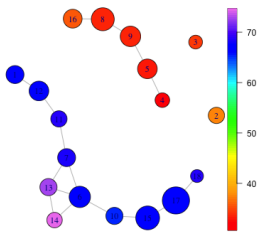
Example TDABM DatasauRus plots: Wide Lines



(a) Scatter

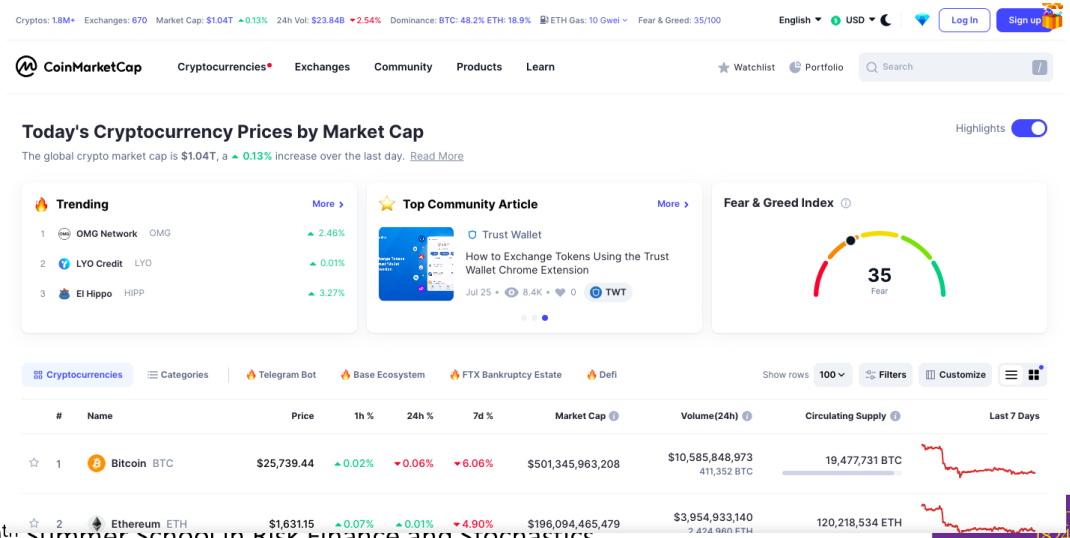


(b) TDABM $\epsilon = 20$



(c) TDABM $\epsilon = 10$

Cryptocurrencies: Coinmarketcap.com



Cryptocurrencies: *crypto2* in R

```
coins <- crypto_list(only_active=TRUE)
```

- Download a full list of active coins
- Use bulk download function to obtain data

```
coin_hist <- crypto_history(coins, limit=5,
start_date="20170101", end_date="20221231", finalWait=FALSE)
```

Cryptocurrencies: *crypto2* in R 2

```
› Scraping historical crypto data

- [2 / 2] [============================================================] 100% in 00:00:03 ETA:  0s
› Processing historical crypto data

| [5 / 5] [============================================================] 100% in 00:00:00 ETA:  0s
>
```

- Creates a *tibble* of the data (*tidyverse*)
- We will convert to data frame and process further for use with TDABM

Cryptocurrencies: *crypto2* in R 3

```
> head(coin_hist)
# A tibble: 6 × 16
  timestamp             id slug    name    symbol ref_cur  open  high   low close volume market_cap
  <dttm>             <int> <chr>   <chr>   <chr>  <chr>   <dbl> <dbl> <dbl> <dbl>  <dbl>      <dbl>
1 2017-01-01 23:59:59     1 bitcoin Bitco… BTC    USD      964. 1003.  959.  998. 1.48e8     1.61e10
2 2017-01-02 23:59:59     1 bitcoin Bitco… BTC    USD      999. 1031.  997. 1022. 2.22e8     1.64e10
3 2017-01-03 23:59:59     1 bitcoin Bitco… BTC    USD     1022. 1044. 1022. 1044. 1.85e8     1.68e10
4 2017-01-04 23:59:59     1 bitcoin Bitco… BTC    USD     1044. 1159. 1044. 1155. 3.45e8     1.86e10
5 2017-01-05 23:59:59     1 bitcoin Bitco… BTC    USD     1157. 1191.  910. 1013. 5.10e8     1.63e10
6 2017-01-06 23:59:59     1 bitcoin Bitco… BTC    USD     1014. 1047.  884.  902. 3.52e8     1.45e10
# ℹ 4 more variables: time_open <dttm>, time_close <dttm>, time_high <dttm>, time_low <dttm>
```

Cryptocurrencies: *crypto2* in R 4

```
table(coinh$symbol)
```

- Can confirm that there are the same number of daily returns for each coin
- Next make a smaller dataset with just date, symbol and closing price

```
coinh<-as.data.frame(cbind.data.frame(coinh$date,coinh$symbol,
  coinh$close))
names(coinh)<-c("date","symbol","close")
```

Reducing Crypto Data

```
> coins<-as.data.frame(table(coinh$symbol))
> names(coins)<-c("symbol","freq")
> head(coins)
  symbol freq
1    BTC 2191
2    LTC 2191
3    NMC 2191
4    PPC 2191
5    TRC 2191
```

- Create a table of coins and how often they appear
- Convert table to `data.frame`
- Provide names
- Loop code on next slide

Crypto Returns Loop

```
ret001<-as.data.frame(matrix(nrow=max(coins$freq),ncol=6))

for(i in 1:5){
 a001<-coins$symbol[i]
 temp<-subset(coinh,coinh$symbol==a001)
 a002<-nrow(temp)
 a003<-i+1
 for(j in 2:a002){
  a004<-j-1
  ret001[j,1]<-coinh$date[j]
  ret001[j,a003]<-100*(log(temp$close[j])-log(temp$close[a004]))
 }
}

ret001<-ret001[-1,] # Drop first row as cannot calculate return

names(ret001)<-c("date",as.character(coins$symbol))
```

Remember to take code from the code file

Crypto Returns New Table

```
> head(ret001) # View the data
   date         BTC         LTC         NMC         PPC         TRC
2 17168   2.3193236   2.9989613   1.9951582   2.123488  12.935220
3 17169   2.1389345  -0.4903415   1.0302817   1.913027   2.086309
4 17170  10.0960345   4.4734587  10.1050035   9.467706 -10.217996
5 17171 -13.0575250 -11.9549899 -12.8735035 -11.193786 -17.414739
6 17172 -11.6209241 -10.5926989  -9.6577315  -4.608785 -17.089439
7 17173   0.7051148   2.4209522   0.7399901   1.057629  20.112180
>
> ret001$date<-as.Date(ret001$date,format="%d-%m-%Y",origin="01-01-1970") # Convert date
> ret001$pt<-seq(1:nrow(ret001)) # Point identifier to match TDABM output
>
```

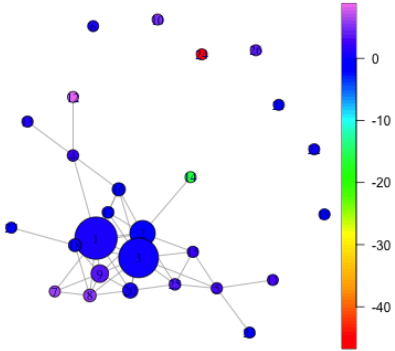
TDABM Plot of Crypto Returns

```
ret002<-as.data.frame(ret001[,2:6])
ret003<-as.data.frame(ret001[,2])
```

- We use the 5 coins returns as the axes for the plot
- We colour according to the return to BTC (column 2)

```
cbm01<-BallMapper(ret002,ret003,30)
ColorIgraphPlot(cbm01,seed_for_plotting=123)
```

Crypto Returns TDABM Plot



- Majority of returns are connected
- Outliers where BTC has lowest returns
- Some extreme returns connected to main plot
- Ball 12 has highest increase (10%)
- Ball 14 has larges fall (20%)

Points to Balls Function

```
# Points to Balls Function

points_to_balls<-function(l){
  a001<-length(l$landmarks)
  a1<-matrix(0,nrow=a001,ncol=2)
  a1<-as.data.frame(a1)
  names(a1)<-c("pt","ball")
  for(i in 1:a001){
    a<-as.data.frame(l$points_covered_by_landmarks[i])
    names(a)<-"pt"
    a$ball<-i
    a1<-rbind.data.frame(a1,a)
   }
  a1<-a1[2:nrow(a1),]
  return(a1)
}
```

- Can create additional functions on top of `BallMapper` object
- `BallMapper` object is labelled in the function as `l`
- Elements of `BallMapper` discussed later

Merge With Underlying Dataset

```
cbm01pb<-as.data.frame(points_to_balls(cbm01))
names(cbm01pb)<-c("pt","ball")
```

- First apply the `points_to_balls` function
- Convert to `data.frame` and name
- Merge `data.frame` to underlying data

```
ret004<-merge(ret001,cbm01pb,by="pt")
```

Query Ball Membership

```
ret00412<-subset(ret004,ret004$ball==12)
```

- Once we have queried the balls membership we may perform analysis on subset
- For example summary statistics `summary(ret00412)`
- To see the dates type `ret00412$date`

```
> ret00412$date
[1] "2017-05-18" "2019-10-25"
```

Elements of a `BallMapper` Object

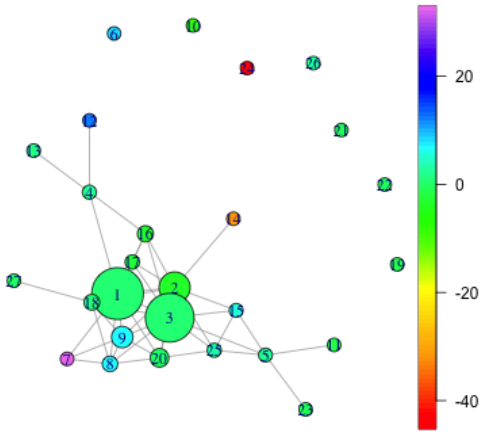
`vertices`	Number of points in each ball
`edges`	Point numbers where each edge starts and finishes
`edges_strength`	Number of points in intersection causing formation of edge (min=1)
`landmarks`	Point numbers for the landmarks used in plot
`coloring`	Value of colouration function for each ball

Coverage is ommitted as we use the `points_to_balls` function instead

Crypto Exercises

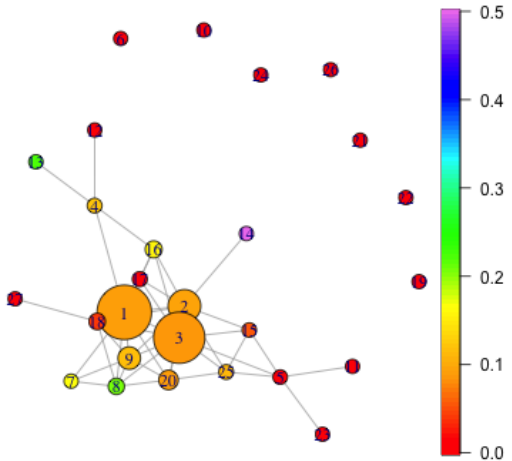
- Construct TDABM plots with radius 30 showing returns on Litecoin (LTC)
- Construct TDABM plots with radius 30 for the remaining 3 coins
- What happens if you increase the radius to 40?
- How would you create a plot coloured according to whether the date is during the Bitcoin Bubble (1st July 2017 to 31st January 2018)?

Crypto Returns TDABM Plot: Litecoin



- Outliers where LTC has lowest returns
- Some extreme returns connected to main plot
- Ball 12 has higher positive than neighbours
- Higher LTC returns to bottom right

Crypto Returns TDABM Plot: BTC Bubble 2017



- Consider 1st July 2017 to 31st January 2018
- Colour is proportion of observations in the period
- None of outliers from BTC bubble
- Highest is half of ball 14 - Also includes 19th May 2021 and 21st June 2021

Key Arguments

Visualising data is an essential phase of the modelling process

Humans cannot see in multiple dimensions - dimension reduction

Mapping helps rationalise space - look to map our data

Dimensionality considered in time series and the cross-section

Summary

- TDABM motivated by Anscombe (1973) and others on value of visualisation
- Function has three inputs - axes, colour, radius
- Use TDABM as basis for many analytical functions
- See applications in final session

Anscombe, F. J. (1973). Graphs in statistical analysis. *The American Statistician*, 27(1):17–21.

Davies, R., Locke, S., and D'Agostino McGowan, L. (2022). *datasauRus: Datasets from the Datasaurus Dozen*. R package version 0.1.6.

Dlotko, P. (2019). *BallMapper: The Ball Mapper Algorithm*. R package version 0.2.0.

Matejka, J. and Fitzmaurice, G. (2017). Same stats, different graphs: generating datasets with varied appearance and identical statistics through simulated annealing. In *Proceedings of the 2017 CHI conference on human factors in computing systems*, pages 1290–1294.