

Attività di allineamento

Laboratorio Informatico-Statistico



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Corso di Laurea Magistrale in Statistica Economia e Impresa

Dipartimento di Scienze Statistiche Paolo Fortunati

Anno Accademico 2021/2022



Data visualization

Data Viz

From data to viz

L'elaborazione dei dati ha come fine ultimo quello di comunicare dei risultati.
La forma più semplice, diretta ed esaustiva di condivisione dei risultati è quella grafica

“A picture is worth a thousand words”

Con il termine data visualization si intende la rappresentazione di informazioni tramite grafici, diagrammi, immagini eccetera.
La dataviz non si limita ai soli grafici, ma a tutto ciò che è “narrato” in maniera grafica.

In questa parte del corso ci concentreremo su uno degli strumenti della dataviz, i grafici.

Data Viz

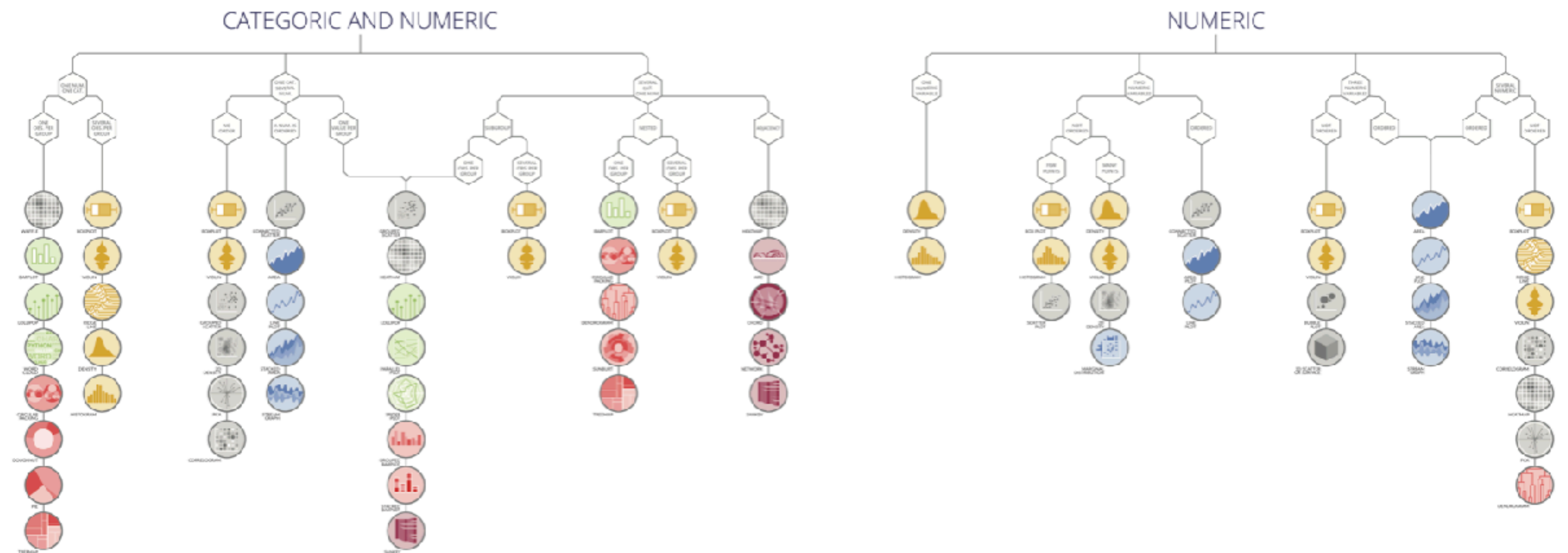
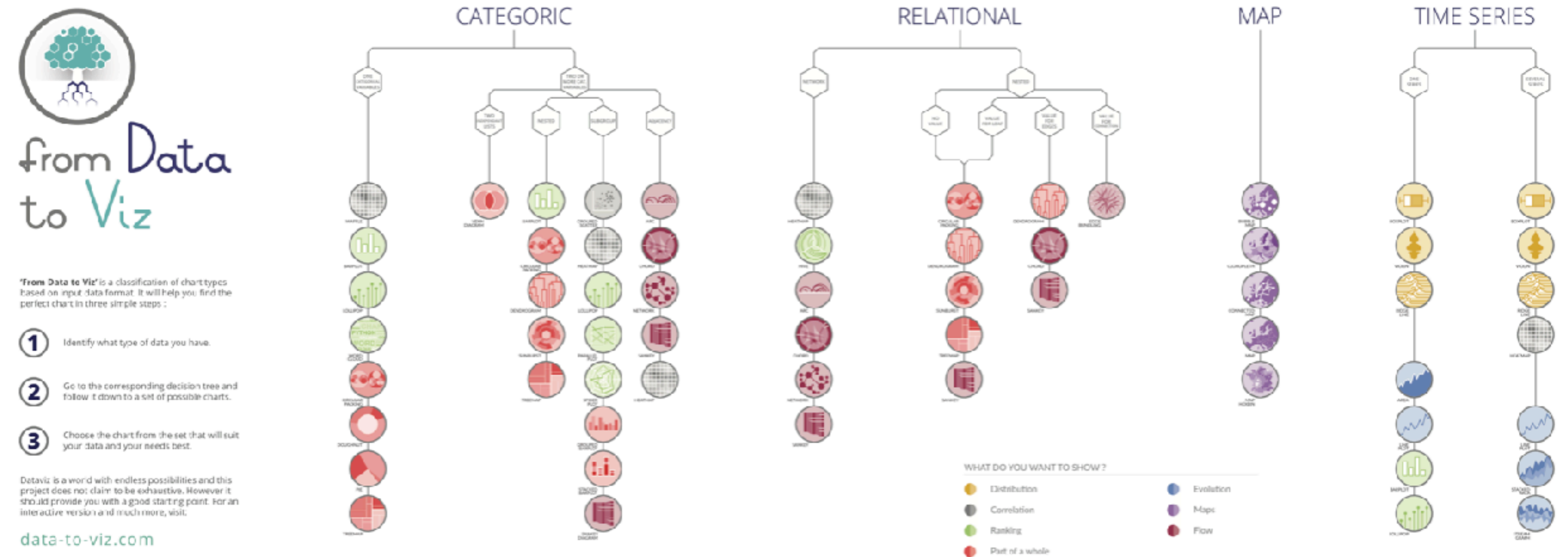
From data to viz

Quanti grafici esistono?

TROPPI.

Una tassonomia dei grafici è quella di *data-to-viz*:

https://www.data-to-viz.com/img/poster/poster_big.png



Data Viz

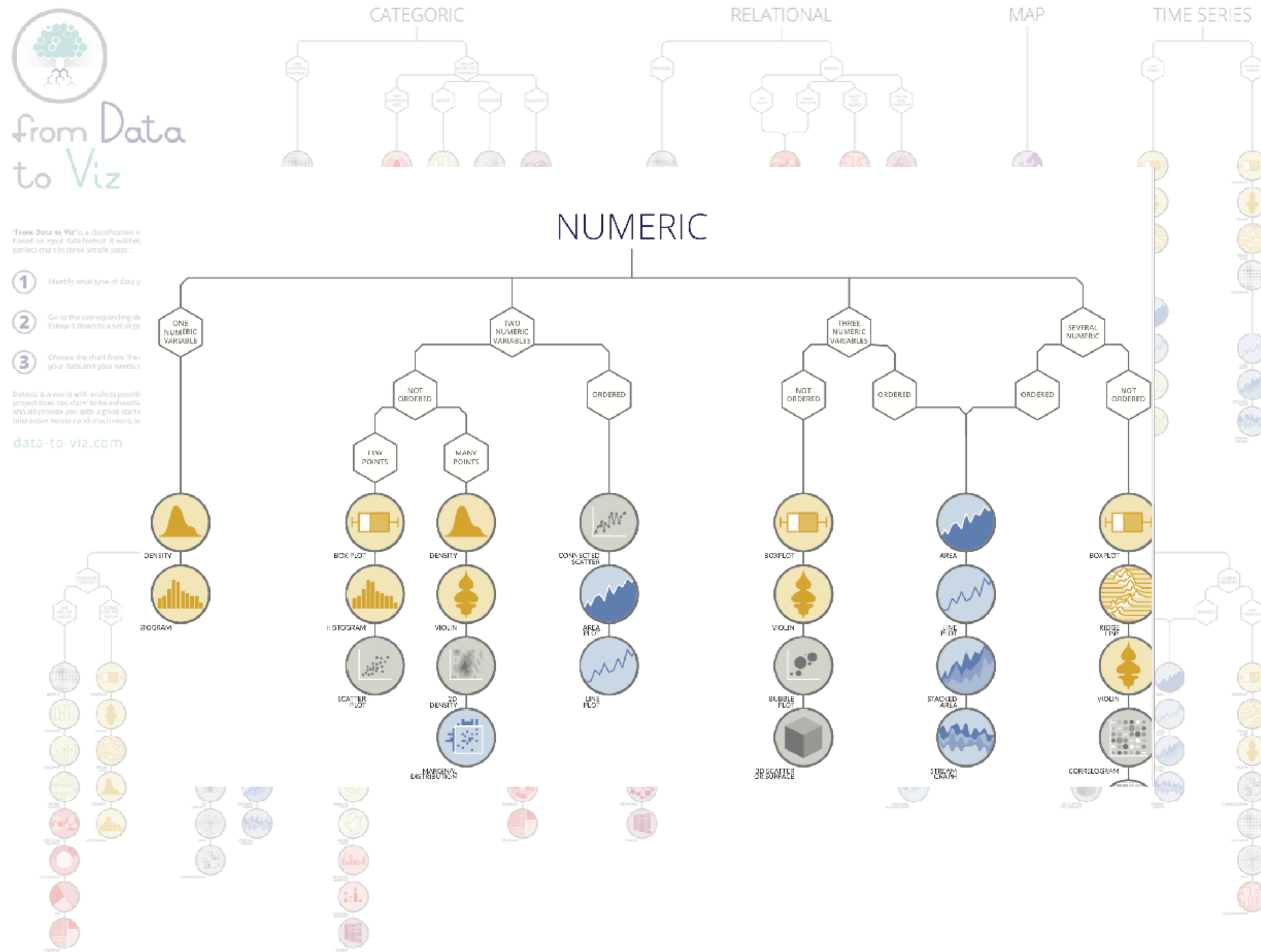
From data to viz

Quanti grafici esistono?

TROPPI.

Una tassonomia dei grafici è quella di *data-to-viz*:

https://www.data-to-viz.com/img/poster/poster_big.png



Data Viz

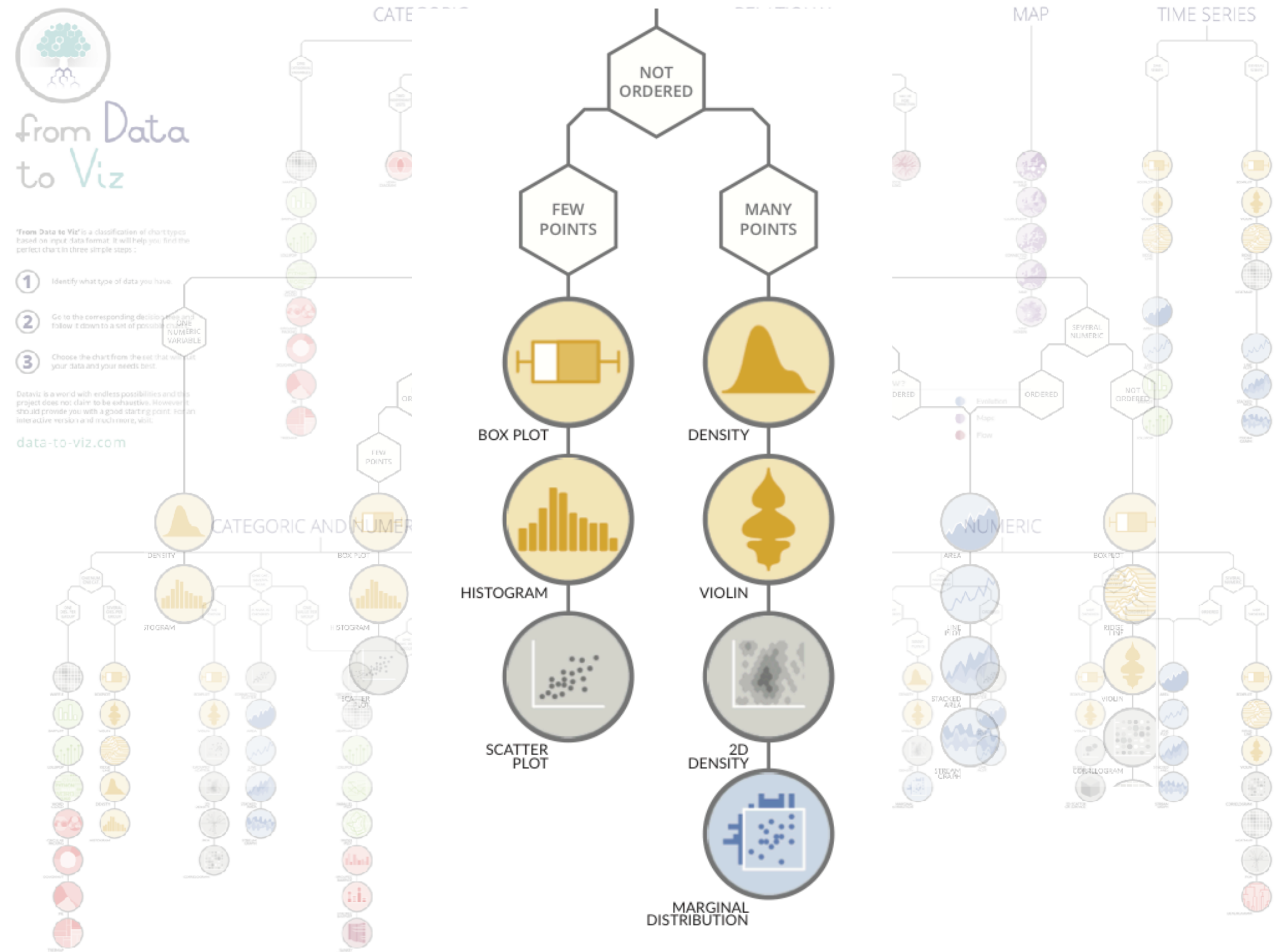
From data to viz

Quanti grafici esistono?

TROPPI.

Una tassonomia dei grafici
è quella di *data-to-viz*:

https://www.data-to-viz.com/img/poster/poster_big.png



Data Viz

From data to viz

Basta scegliere il grafico corretto per una buona infografica?

NO.


Gli errori più comuni da evitare sono riassunti in:

<https://www.data-to-viz.com/caveats.html>

CAVEATS


A collection of dataviz caveats by [data-to-viz.com](https://www.data-to-viz.com)

Show all Top 10 Improvement Misleading Map Bar




Order your data

When displaying the value of several entities, ordering them makes the graph much more insightful.




To cut or not to cut?

Cutting the Y-axis is one of the most controversial practice in data viz. See why.




The spaghetti chart

A line graph with too many lines becomes unreadable: it is called a spaghetti graph.



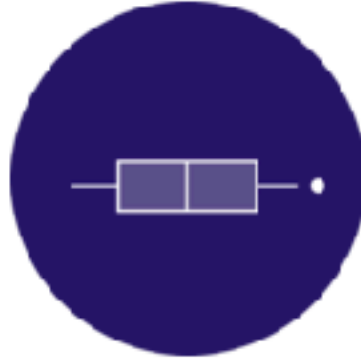
Pie chart

The human eye is bad at reading angles. See how to replace the most criticized chart ever.




Play with histogram bin size

Always try different bin sizes when you build a histogram, it can lead to different insights.




Do boxplots hide information?

Boxplots are a great way to summarize a distribution but hide the sample size and their distribution.



The problem with error bars

Barplots with error bars must be used with great care. See why and how to replace them.



Too many distributions.

If you need to compare the distributions of many variables, don't clutter your graphic.

Data Viz

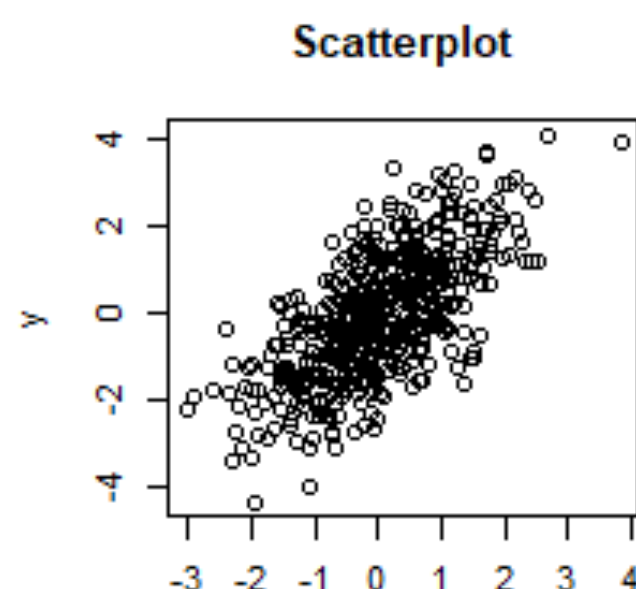
I grafici su R - *graphics*

Le 3 librerie più comuni per creare grafici su R sono *graphics* (che fa parte delle librerie “base” di R), *ggplot2* e *lattice*. Partiamo dal pacchetto più semplice, *graphics*.

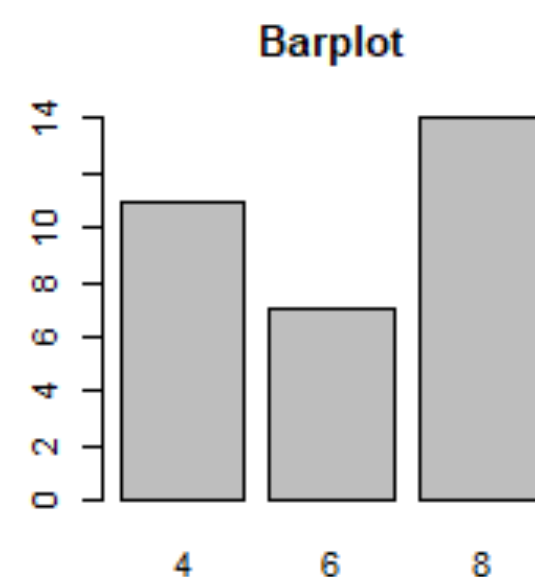
La funzione *plot()* permette di creare un generico grafico. La tipologia, l'apparenza e le specifiche del grafico vengono definite tramite gli argomenti.

Il primo argomento della funzione *plot()* è il dato da rappresentare. In base alla tipologia di dato passato alla funzione (vettore numerico, fattore, time series, ...) verrà restituito un grafico diverso

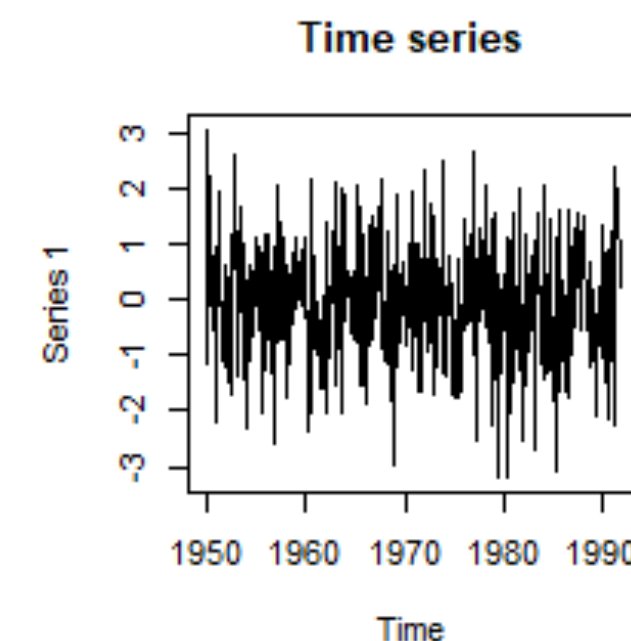
```
> x = rnorm(500)
> y = x + rnorm(500)
> plot(x, y, main="scatterplot")
```



```
> data(mtcars)
> attach(mtcars)
> myfactor = factor(mtcars$cyl)
> plot(myfactor, main="barplot")
```



```
> myts = ts(matrix(rnorm(500), nrow =
  500, ncol = 1), start = c(1950, 1),
  frequency = 12)
> plot(myts, main="time series")
```



Data Viz

I grafici su R - *graphics*

Per stampare diversi grafici contemporaneamente si utilizza la funzione *par()* e si indicano il numero di grafici (e la loro disposizione) tramite l'argomento *mfrow*:

```
> par(mfrow = c(1,3))
```

```
> plot(...)
```

```
> plot(...)
```

```
> plot(...)
```

Produrrà 3 grafici in linea.

Per personalizzare il grafico esistono diversi argomenti. Ad esempio *type* permette di specificare la tipologia di rappresentazione dei dati (ma dipende dal tipo di dato passato), oppure *pch* permette di modificare il simbolo dei punti nel grafico

Data Viz

I grafici su R - *graphics*

I diversi elementi del grafico possono anche essere aggiunti in maniera sequenziale utilizzandoli come funzioni:

```
> plot(x,y)
```

```
> title("Titolo")
```

Equivale a:

```
> plot(x, y, main="Titolo")
```

Gli assi possono essere rinominati con i parametri *xlab* e *ylab* oppure si può rimuovere il loro nome tramite *ann*. Anche gli assi possono essere aggiunti (e gestiti) successivamente tramite la funzione *axis()*.

Una panoramica completa sulla personalizzazione dei grafici di base può essere trovata su:

<https://r-coder.com/plot-r/>

Data Viz

I grafici su R - *ggplot2*

ggplot2 è una libreria grafica molto potente basata sulla *Grammar of Graphics* di *Wilkinson*, pubblicazione che descrive le caratteristiche fondamentali di un grafico statistico. Le componenti principali di un grafico sono definite come i dati, gli attributi estetici e gli oggetti geometrici.

In generale, i grafici sono composti dai dati e da un *mapping* che descrive come le variabili vengono mappate agli attributi estetici. In particolare esistono 5 tipologie di mapping:

1. **layer**: collezione di elementi geometrici (*geom*) e trasformazioni statistiche (*stat*)
2. **scale**: mappa i valori dallo spazio dati allo *spazio estetico* (include l'uso di legenda, colori, forme e dimensioni)
3. **coord**: descrive come le coordinate dei dati vengono mappate sul piano del grafico
4. **facet**: specifica come dividere e mostrare subset di dati
5. **theme**: controlla le particolarità della visualizzazione, come colore di sfondo e dimensione dei caratteri

Data Viz

I grafici su R - *ggplot2*

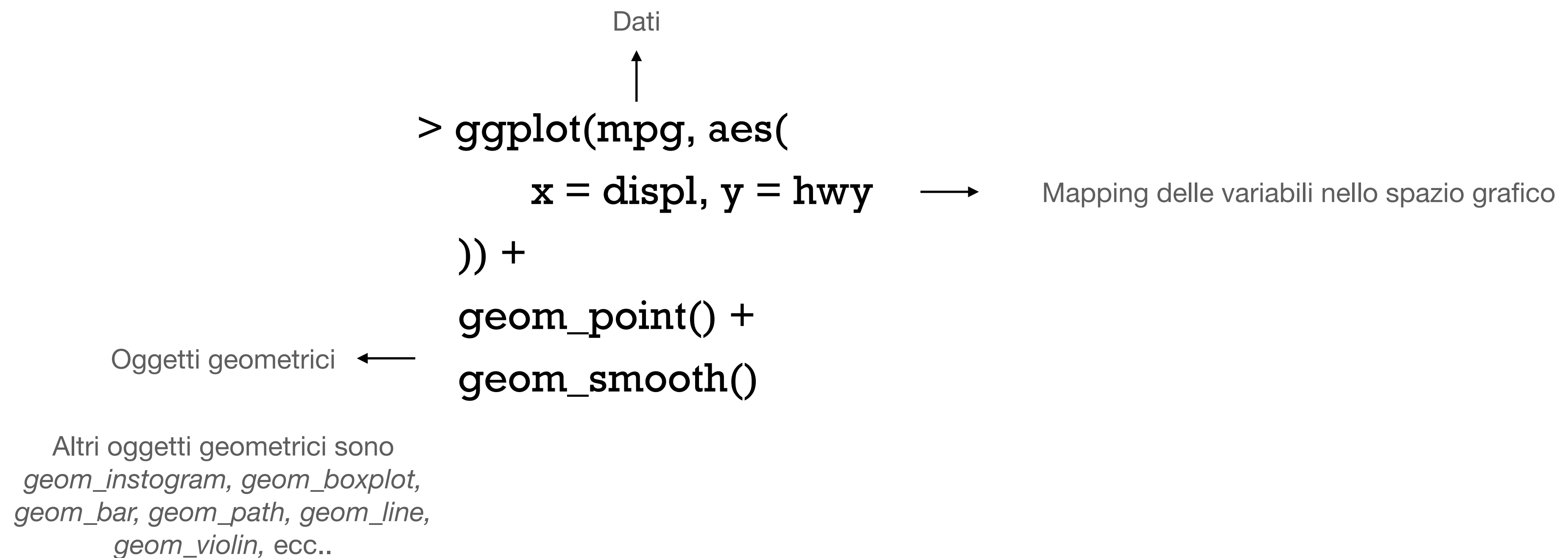
La funzione per produrre un grafico con *ggplot2* è *ggplot()* che accetta come argomenti principali i dati da cui estrarre il grafico e i parametri *aes*. A questi vanno aggiunti gli oggetti geometrici con l'operatore `+`:

```
          Dati
          ↑
> ggplot(mpg, aes(
      x = displ, y = hwy  → Mapping delle variabili nello spazio grafico
    )) +
Oggetti geometrici ← geom_point()
```


Data Viz

I grafici su R - *ggplot2*

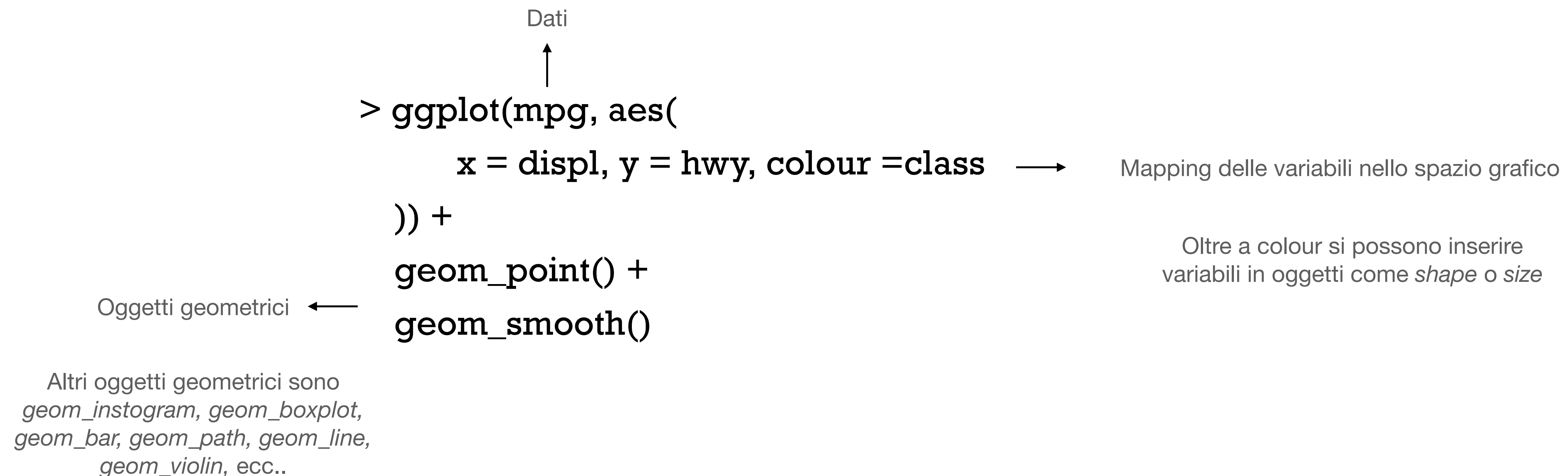
Si possono inserire anche più oggetti geometrici



Data Viz

I grafici su R - *ggplot2*

O aggiungere altre variabili in elementi grafici diversi dagli assi



Data Viz

I grafici su R - *ggplot2*

Gli assi vengono gestiti tramite le funzioni *xlab* e *ylab* per il nome, mentre si usano *xlim* e *ylim* per modificarne i limiti

```

      Dati
      ↑
> ggplot(mpg, aes(
    x = displ, y = hwy, colour = class → Mapping delle variabili nello spazio grafico
  )) +
  geom_point() +
  geom_smooth() +
  xlab("nome asse x") +
  ylab("nome asse y") +
  xlim(0,5) +
  ylim(0,30)
  ← Oggetti geometrici
  ← Parametri sugli assi
```

Data Viz

I grafici su R - *ggplot2*

Per salvare un grafico si può usare il tasto *export* nel tab *plots* di Rstudio oppure la funzione *ggsave()*

```
> ggsave("nomedelfile.estensione", graficodasalvare)
```

Per approfondire la creazione di grafici con ggplot si consiglia la lettura della documentazione:

<https://ggplot2-book.org/introduction.html>