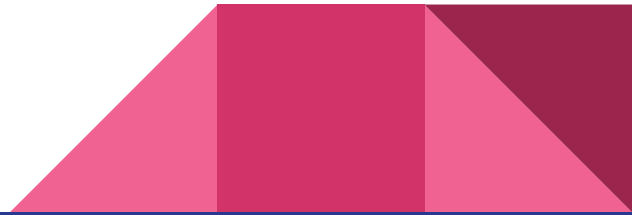
08. Tool per la visualizzazione dati

Corso di Python per il Calcolo Scientifico

Angelo Cardellicchio
angelo.cardellicchio@stiima.cnr.it

Outline

- Matplotlib
  - Il primo plot
  - Figure ed assi
- Seaborn
  - Visualizzare le relazioni tra dati
  - Visualizzare le distribuzioni
  - Dati categorici
  - Heatmap



Il primo plot

- Proviamo ad effettuare un plot.

```
rng = np.random.default_rng(42)
x = np.arange(1, 6)
y = rng.integers(low=0, high=10, size=5)
fig, ax = plt.subplots()
ax.plot(x, y)
plt.show()
```

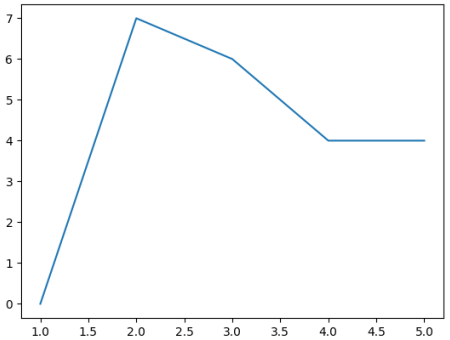


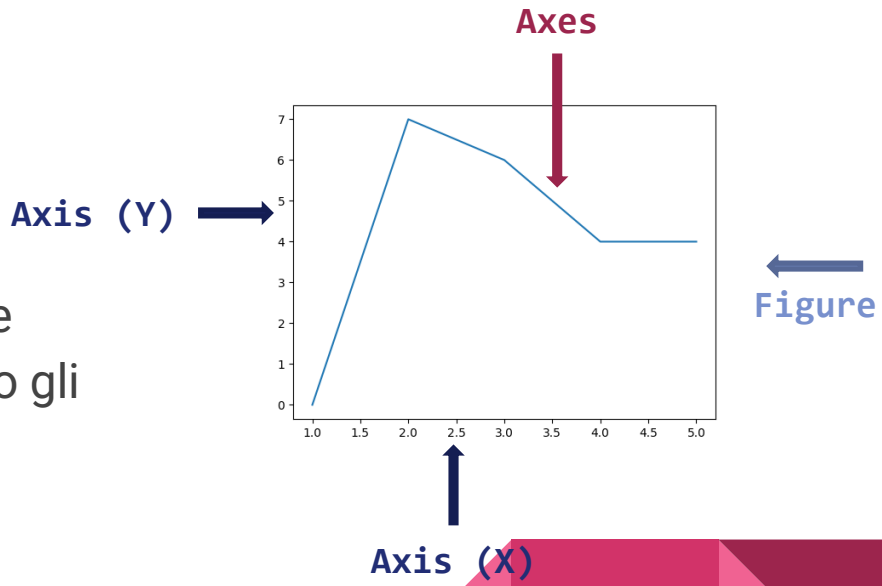
Figure ed assi

- Una **`Figure`** rappresenta l'intera figura mostrata da Matplotlib.
- Gli **`Axes`** sono i plot veri e propri.

```
fig, ax = plt.subplots()
```

- All'intero di ogni **`Axes`** ci sono due o tre oggetti di tipo **`Axis`**, che rappresentano gli assi.

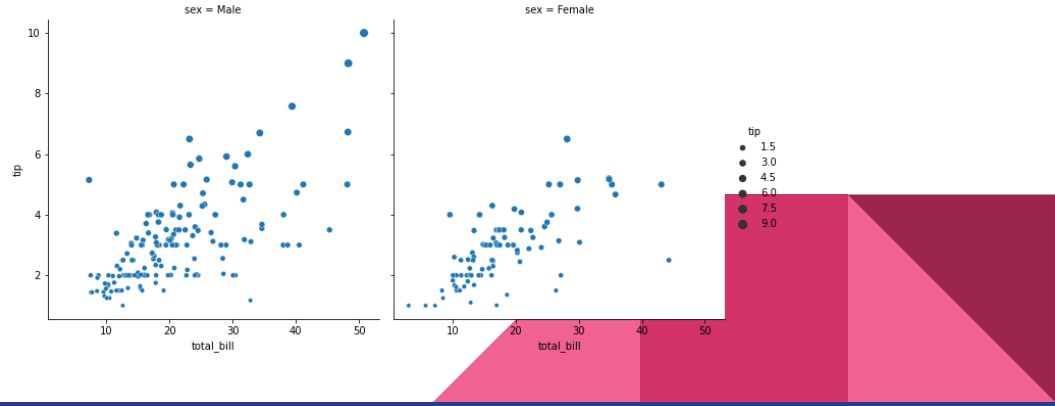
```
ax.plot(x, y)
```



Visualizzare le relazioni tra dati

- **Seaborn** è una libreria basata su Matplotlib per la visualizzazione avanzata dei dati
  - In tal senso, ci offre una serie di strumenti appositamente dedicati
- Possiamo visualizzare la relazione tra dati in un dataframe con la funzione **`relplot()`**:

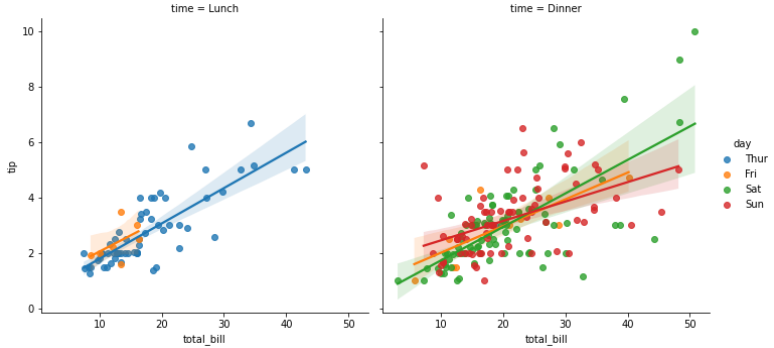
```
sns.relplot(
    data=tips,
    x='total_bill',
    y='tip',
    col='sex',
    size='tip')
```



Visualizzare le relazioni tra dati

- La funzione **`lmplot()`** estende la `relplot()` effettuando un'interpolazione dei dati:

```
sns.lmplot(
    data=tips,
    x='total_bill',
    y='tip',
    col='time',
    hue='day')
```

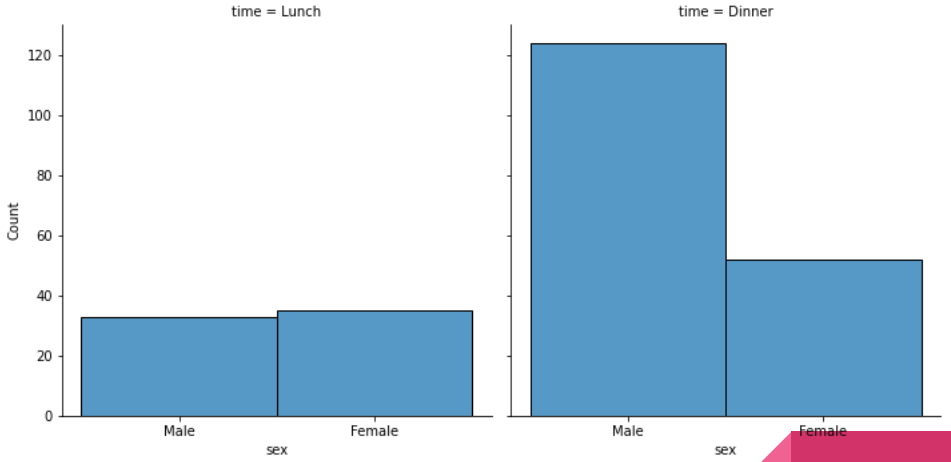


- È possibile selezionare varie tecniche di regressione passando diversi parametri alla funzione.

Visualizzare le distribuzioni

- La funzione **`displot()`** ci permette di analizzare la distribuzione dei dati:

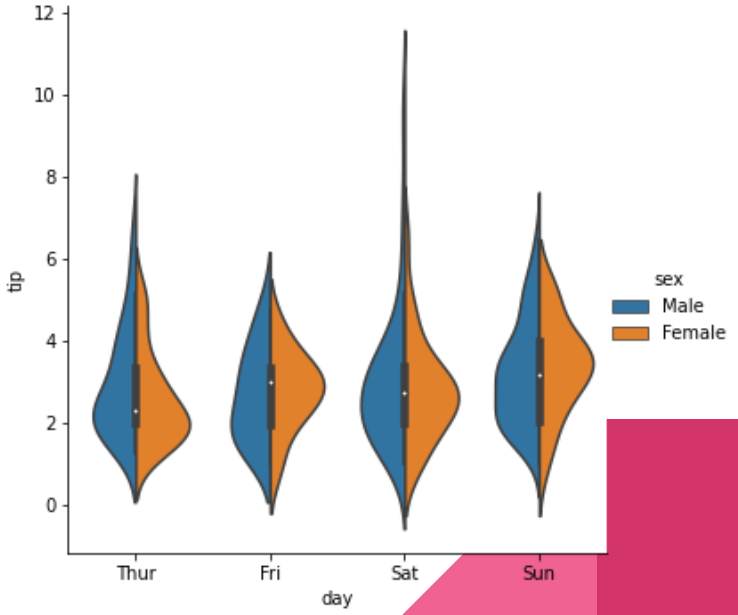
```
sns.displot(
    data=tips,
    x='sex',
    col='time',
    kde=True)
```



Dati categorici

- Mediante la funzione **`catplot()`** possiamo avere ulteriori dettagli visivi per analizzare la distribuzione di dati di tipo categorico:

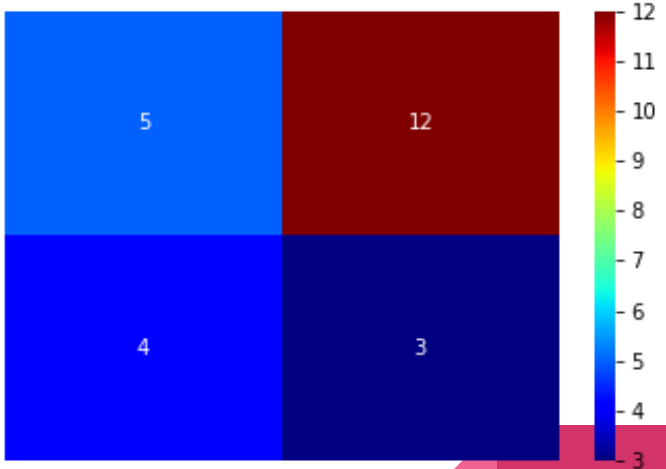
```
sns.catplot(
    data=tips,
    kind='violin',
    x='day',
    y='tip',
    hue='sex',
    split=True)
```



Heatmap

- La funzione **heatmap()** ci permette di mostrare i grafici omonimi:

```
ar = np.array([[5, 12], [4, 3]])
sns.heatmap(
    ar,
    cmap='jet',
    annot=True,
    xticklabels=False,
    yticklabels=False)
```



Domande?

42