Jupyter (autore: Vittorio Albertoni)

Premessa

Chi pensa che si tratti di un modo ortograficamente scorretto di nominare il capo degli Dei dell'antica Roma abbia subito chiaro che Giove non c'entra nulla.

Jupyter è semplicemente un editor di testo che ci consente di inserire, oltre ai soliti arricchimenti consistenti in illustrazioni e altri contenuti multimediali, codice interpretabile nello stesso editor per ottenere risultati di elaborazioni che pure entrano a far parte del testo.

Il nome è l'acronimo di JUlia, PYThon R, ad indicare i tre linguaggi di programmazione nativamente supportati, con in più una E eufonica ben collocata.

Dico nativamente supportati in quanto sono quelli che più facilmente possiamo utilizzare con Jupyter: infatti, per utilizzare il linguaggio Python non dobbiamo addirittura fare nulla in quanto Jupyter nasce in casa Python, per utilizzare il linguaggio Julia dobbiamo fare quasi nulla e per utilizzare il linguaggio R poco più di quasi nulla. Tutto a portata di dilettante.

Con qualche lavorio più da professionisti che da dilettanti possiamo fare in modo che Jupyter riconosca un'altra cinquantina di linguaggi: da C++ a Java, da Javascript a PHP, da Fortran a Ruby.

Le origini di Jupyter risalgono a quando due ricercatori dell'Università di Berkeley, Fernando Pérez e Robert Kern, danno avvio ad un progetto per creare un interprete interattivo per Python, IPython, che tuttora è utilizzabile come tale. Sono poi gli sviluppatori del team di questo progetto che, a partire dal 2010, creano una interfaccia arricchita, basata su IPython, che si chiama Jupyter Notebook.

Jupyter Notebook è la base di Jupyter.

Nel tempo sono intervenuti arricchimenti ed oggi il vecchio Jupyter Notebook sta passando il testimone a JupyterLab, che altro non è che un'interfaccia che ci consente di lavorare con più Notebook.

Abbiamo, infine, JupyterHub, che trasferisce JupyterLab su un server multiutente, in modo che sullo stesso progetto possano lavorare più persone distanziate nello spazio.

In questo manualetto presento Jupyter Notebook, esponendo quanto basta per il neofita e quanto comunque necessario conoscere per poter utilizzare anche gli altri due strumenti citati.

Indice

1 Installazione

Jupyter Notebook è un prodotto del mondo Python.

Per essere utilizzato occorre sia installato l'interprete Python e quello di eventuali altri linguaggi si vogliano utilizzare.

Chi ha installato Python Anaconda si trova Jupyter Notebook in Anaconda navigator senza dover fare nulla[1].

Visto che mi rivolgo a neofiti dilettanti che probabilmente nemmeno sanno cos'è Anaconda suggerisco la via più semplice: l'installatore pip[2].

Supponendo si voglia lavorare con Python3, l'installazione avviene con il comando a terminale

```
pip3 install jupyter
```

Con questo abbiamo a disposizione Jupyter Notebook abilitato per il linguaggio Python3 e per l'esportazione dei documenti in formato `.html`.

Per abilitare Jupiter Notebook anche al linguaggio Julia dobbiamo aprire il REPL di Julia con il comando a terminale

```
julia
```

entrare in modalità Package con il comando

```
]
```

e digitare il comando

```
add IJulia
```

Direi che per un dilettante potrebbe bastare.

Per poter esportare documenti nel formato PDF attraverso il formato HTML occorre che sul computer sia installato il browser Chromium.

Per poter esportare documenti in formato LaTeX (`.tex`), o in formato PDF attraverso il formato LaTeX, oltre ad avere installato una distribuzione LaTeX tipo Tex live (avendo cura che, tra gli altri, sia installato il package `texlive-xetex`), dobbiamo installare il software Pandoc, che troviamo all'indirizzo *https://pandoc.org/installing.html* nelle versioni per Linux, Mac e Windows. Chi usa Linux trova sicuramente Tex live e Pandoc nel repository della sua distro e può installare quanto serve con il gestore dei programmi.

Visto che l'installazione di questi software occupa parecchio spazio su disco, teniamo presente che i documenti in formato `.html` prodotti da Jupyter Notebook sono perfetti e fanno la loro bella figura anche loro e, per esportare in formato PDF ci basta aver installato Chromium.

2 Come funziona

Jupyter Notebook viene lanciato con il comando a terminale

```
jupyter notebook
```

A seconda del sistema operativo che usiamo, con l'installazione si può essere creato un lanciatore o una voce di menu che ci evita la fatica.

Possiamo anche creare noi un lanciatore in cui inserire il comando per il lancio.

Con il lancio si crea una nuova finestra del nostro browser web predefinito in quanto Jupyter Notebook è un'applicazione web basata su una struttura server-client.

Per default il server si attesta su localhost all'indirizzo 127.0.0.1:8888.

Se succede qualche cosa di strano e il server non si attiva provare a lanciare con il comando

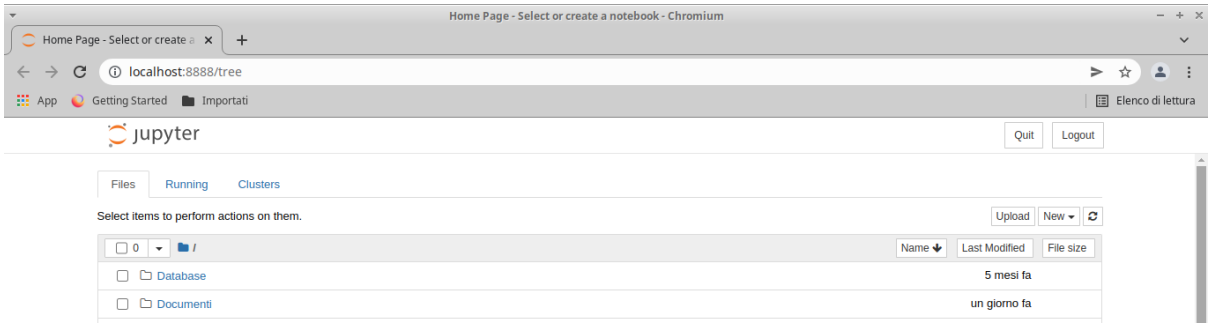
```
jupyter notebook --NotebookApp.use_redirect_file=False
```

La finestra che si apre al lancio è parzialmente riprodotta all'inizio della pagina seguente.

Essa non è altro che un gestore di file aperto sulla nostra home directory e ci invita, come sta scritto nella prima riga visibile in alto, a selezionare o a creare un notebook.

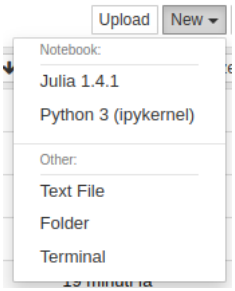
[1] Per chi fosse interessato segnalo il mio articolo sul blog *www.vittal.it* «Software libero per data scientists» dell'aprile 2019 con allegato il manualetto in formato PDF «python_anaconda».

[2] In proposito segnalo il mio articolo sul blog *www.vittal.it* «Python per tutti» del febbraio 2017, con allegato il manualetto in formato PDF «mondo_python».



Se vogliamo lavorare su un notebook già avviato lo possiamo cercare scorrendo la parte inferiore della finestra, dove abbiamo l'elenco delle sottodirectory e dei file contenuti nella nostra home directory, e caricarlo cliccandoci sopra.

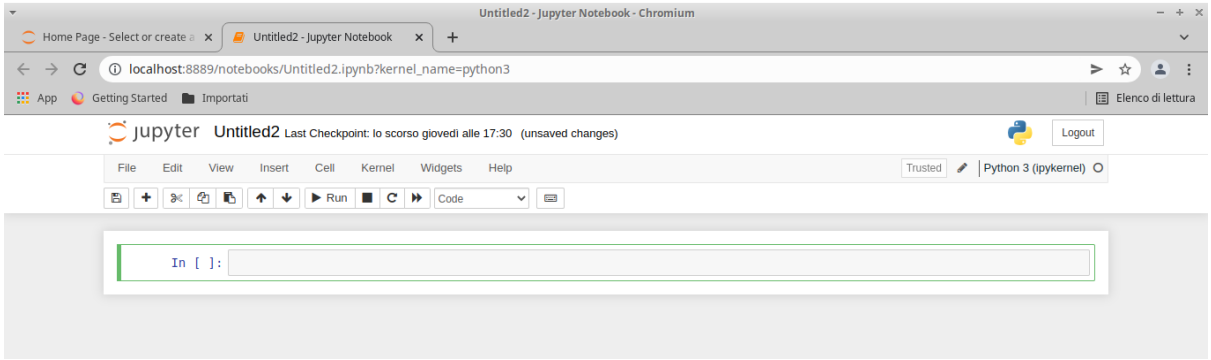
Se vogliamo creare un nuovo notebook apriamo il menu contenuto nella voce NEW sulla destra



e scegliamo il linguaggio di programmazione che intendiamo utilizzare per il codice che vi inseriremo.

Come si vede potremmo anche scegliere di fare altro, come un semplice file di testo. Ma direi che per fare queste altre cose è inutile disturbare Jupyter Notebook.

Sia che abbiamo caricato un notebook esistente sia che ne abbiamo creato uno nuovo, lo ritroviamo in una nuova finestra del nostro browser predefinito che si aggiunge alla home page vista prima.



L'illustrazione mostra un notebook nuovo.

Abbiamo una barra di menu esplorando la quale, sia pure in lingua inglese, veniamo informati di cosa possiamo fare.

Appena sotto abbiamo una barra di strumenti iconizzati che velocizzano alcune cose, le più ricorrenti, che potremmo fare con le voci di menu. Passando il mouse sulle varie icone siamo informati, in lingua inglese, di ciò che otteniamo cliccando su di esse.

Il tutto è abbastanza intuitivo.

La prima cosa utile che possiamo fare è dare un nome al notebook e lo facciamo cliccando sulla dizione `UntitledX` che compare sopra la barra di menu ed inserendo il nome voluto nella finestra di dialogo che si apre.

Ora la cosa più importante da capire è come si lavora nell'editor sottostante che, come si vede nell'illustrazione, presenta una finestra di immissione dati da tastiera, chiamata cell.

Tutto ciò che vogliamo far comparire nel notebook da scambiare con altri o da utilizzare per produrre un documento deve essere immesso utilizzando finestre come questa.

Nell'illustrazione la finestra è predisposta, ed è la situazione di default, per l'immissione di codice: lo si vede dalla scritta `In [ ]:` sulla sinistra e dalla scritta `Code` che compare nel penultimo widget della barra degli strumenti.

Nell'ultima finestrella della barra di menu vediamo che il codice deve essere in linguaggio Python3, secondo la scelta effettuata nel momento in cui abbiamo creato il notebook da NEW. Importante ricordare che il linguaggio scelto non è modificabile una volta creato il notebook e, nello stesso notebook, non possiamo alternarlo con altri linguaggi.

La finestra che vediamo nell'illustrazione è selezionata e lo si vede dal fatto che è intelaiata in una cornice con una barretta verticale di colore blu sulla sinistra.

Per inserire il codice dobbiamo cliccare all'interno dello spazio di inserimento in modo che compaia il cursore lampeggiante.

In presenza di più finestre, che si possono aggiungere cliccando sul tasto con l'icona **+** nella barra degli strumenti, per selezionare la finestra che interessa basta cliccarci sopra.

Per inserire del normale testo con eventuali arricchimenti (illustrazioni e altri contenuti multimediali) dobbiamo commutare la finestra di immissione dati selezionata in modo che si predisponga all'immissione di testo.

Lo possiamo fare aprendo la finestrella in cui è scritto `Code` nella barra degli strumenti e scegliendo `Markdown`.

La finestra predisposta per l'immissione di testo si riconosce per la mancanza della scritta `In [ ]:` sulla sinistra e dalla scritta `Markdown` che compare nel penultimo widget della barra degli strumenti.
Questa scritta richiama il fatto che anche il così detto testo che inseriamo in questa finestra, in realtà è un codice, precisamente è un testo scritto con un linguaggio di markup che si chiama Markdown.

Un modo veloce per commutare la finestra selezionata è il seguente:
. premendo il tasto ESC e poi il tasto `y` si ottiene una finestra per il codice,
. premendo il tasto ESC e poi il tasto `m` si ottiene una finestra per il markdown.
Se la finestra non è ancora pronta per l'inserimento (non è presente il cursore lampeggiante) si può evitare di premere il tasto ESC e basta premere i tasti `y` o `m`.

Lavorando nella finestra di immissione, con la pressione del tasto INVIO passiamo ad una riga successiva all'interno della finestra e con la pressione del tasto MAIUSCOLO (SHIFT) insieme al tasto INVIO provochiamo l'esecuzione di quanto abbiamo immesso nella finestra.

Appena fatto questo si crea una nuova finestra, per default predisposta per l'inserimento di codice.

Tutto ciò che abbiamo fatto nel notebook lo possiamo salvare in un file con estensione `.ipynb`.

Basta che clicchiamo sul pulsante  , il primo nella barra degli strumenti.

Troveremo così il file con il nome che abbiamo dato al notebook e l'estensione `.ipynb` nella home directory e un suo backup, che in terminologia Jupyter si chiama Checkpoint, nella sottodirectory nascosta `.ipynb_checkpoints`.

Questo file può essere ricaricato in un Jupyter Notebook, anche su un computer diverso, che abbia le stesse attrezzature software di quello su cui è stato creato, per proseguire il lavoro, per rieseguire il codice, modificarlo, migliorarlo, ecc.

Il file deve essere accompagnato dagli eventuali contenuti multimediali linkati nel testo e dai file di dati esterni eventualmente utilizzati dal codice.

L'esportazione in altri formati avviene da menu FILE ▷ DOWNLOAD AS scegliendo il formato nel menu che si apre.

Anche le esportazioni in formato `.html` e `.tex`, se destinate ad essere utilizzate su computer diversi, vanno accompagnate dagli eventuali contenuti multimediali linkati nel testo.

L'unico formato da opera definitivamente compiuta è il PDF: il file prodotto in questo formato viaggia da solo ma non è modificabile.

Per produrlo attraverso il formato HTML, avendo installato sul computer il browser Chromium, basta scrivere a terminale, posizionati nella home directory, il comando
`jupyter nbconvert --to webpdf --allow-chromium-download <nome_file>.ipynb`

La chiusura in modo ordinato di Jupyter Notebook dovrebbe avvenire secondo la seguente procedura.
. chiudere il tab del notebook nel browser cliccando sul simbolo `x` sulla destra nella tacca,
. nella home page che rimane aperta cliccare sulla scheda RUNNING,
. cliccare sul pulsante SHUT DOWN corrispondente al notebook da chiudere,
. non appena arriva la conferma che il notebook è stato chiuso, chiudere il browser,
. cliccare sulla finestra terminale che si era aperto al momento del lancio di Jupyter Notebook,
. chiudere il terminale con due `Ctrl-C` ravvicinati.

3 Editing del codice

Il codice può essere inserito ed eseguito comando per comando, come se lavorassimo in una shell, o per blocchi di comandi fino a formare un programma completo.

Ovviamente occorre conoscere il linguaggio di programmazione che si è scelto di utilizzare quando si è creato il notebook.

A chi fosse sprovvisto di questa conoscenza segnalo i seguenti manualetti introduttivi a due dei linguaggi base di Jupyter Notebook che si trovano nel mio blog all'indirizzo *www.vittal.it*:
. per il linguaggio Python, «python» allegato all'articolo «Python per principianti» dell'ottobre 2020,
. per il linguaggio Julia: «julia» allegato all'articolo «Un nuovo linguaggio per la data science» del febbraio 2021.

Sullo stesso blog vi sono poi vari manualetti su approfondimenti relativi al linguaggio Python e un volumetto che raggruppa, a livello divulgativo, un po' tutto si trova su Amazon: Vittorio Albertoni - Tutto Python per principianti.

4 Editing del contorno al codice

Per gli elementi diversi dal codice in linguaggio di programmazione utilizziamo la finestra predisposta per il linguaggio Markdown.

Qui di seguito espongo le marcature Markdown riconosciute da Jupyter Notebook per creare documenti anche di un certo impegno e varietà di contenuto.

4.1 Testo

Il normale testo non formattato va semplicemente scritto.

Titoli e intestazioni vanno scritti premettendo al testo da uno a cinque cancelletti (#) e lasciando uno spazio tra i cancelletti e il testo. Con un solo cancelletto si realizza il titolo di dimensione maggiore e tanti più cancelletti inseriamo, tanto più piccolo sarà il titolo.

Per andare a capo prima della fine riga occorre inserire la marca `<br>`.

Per iniziare un nuovo capoverso andando a capo occorre inserire una riga bianca.

Ciò che si vuole scrivere in corsivo va incluso tra due asterischi (*testo*).

Ciò che si vuole scrivere in grassetto va incluso tra due doppi asterischi (**testo**).

Ciò che si vuole scrivere in colore diverso dal nero di default va incluso tra le marche `<font color = "nome_colore">` e `</font>` senza lasciare spazi bianchi tra marche e testo.
I nomi colore riconosciuti sono: gray, silver, teal, yellow, green, lime, olive, red, fuchsia, maroon, aqua, blue, navy, purple (oltre a black e white, inutili in questa sede).

Un testo da indentare va preceduto dalla marca > senza lasciare spazio bianco tra marca e testo.

Elenchi non numerati vanno inseriti facendo precedere le voci da asterisco e lasciando uno spazio tra l'asterisco e la voce.

Elenchi numerati vanno inseriti facendo precedere le voci da `1.` e lasciando uno spazio tra il punto e la voce. La numerazione progressiva delle voci avviene automaticamente.

4.2 Formule ed espressioni matematiche

Vanno inserite secondo la sintassi LaTeX
. scrivendole tra due marche $ se devono essere riprodotte sulla riga stessa,
. scrivendole tra due marche $$ se devono essere riprodotte centrate in riga dedicata.

Esempi:
Con questo inserimento

```
Questa è una formula $y=\frac{\sin(x)}{x+1}$
```

si produce questo

Questa è una formula $y = \frac{\sin(x)}{x+1}$

Con questo inserimento

```
Questa è una formula $$\frac{\sin(x)}{x+1}$$
```

si produce questo

Questa è una formula

$$\frac{\sin(x)}{x+1}$$

Con questo inserimento

```
$$\lim_{n \to \infty}\sum_{k=1}^n \frac{1}{k^2}=\frac{\pi^2}{6}$$
```

si produce questo

$$\lim_{n\to\infty}\sum_{k=1}^{n}\frac{1}{k^2}=\frac{\pi^2}{6}$$

Per la sintassi LaTeX relativa alla creazione di formule ed espressioni matematiche rimando al Capitolo riguardante la matematica in Lorenzo Pantieri o Lorenzo Pantieri & Tommaso Gordini - L'arte di scrivere con LaTeX, che si trova all'indirizzo
http://www.lorenzopantieri.net/LaTeX_files/ArteLaTeX.pdf.

Fin dove può servire vale anche la sintassi Plain TeX, per la quale rimando al Capitolo 4 del manualetto plain_tex.pdf allegato al mio articolo del giugno 2022 sul blog *www.vittal.it* intitolato «Plain Tex».

Il rendering delle formule sulle pagine in formato HTML avviene senza che sia installato il software per LaTeX.

4.3 Link

Possiamo creare link esterni, verso un url esterno al documento, oppure interni, verso un punto all'interno del documento.

Il link esterno si crea con la sintassi

```
[nome_link](nome_url)
```

Per esempio, per creare un link verso l'indirizzo dove si trova la guida LaTeX richiamata alla fine del paragrafo precedente, scriviamo

```
[guida_latex](http://www.lorenzopantieri.net/LaTeX_files/ArteLaTeX.pdf)
```

Il link interno si crea innanzi tutto inserendo nel punto verso il quale si vuole linkare

```
<a id="nome_link"></a>
```

e inserendo nel punto dal quale si vuole linkare
`[nome_link](#nome_link)`

4.4 Contenuti multimediali

Possiamo inserire nel documento anche immagini, audio e video.

Per le immagini usiamo la sintassi
`![titolo_immagine](path_al_file)`
Per esempio, con
`![grafico](/home/vittorio/Immagini/logaritmo.png)`
inserisco nel documento, con la descrizione «grafico» l'immagine del grafico di una funzione che si trova nella directory Immagini della mia home.
In alternativa possiamo usare la sintassi del linguaggio HTML, attraverso la quale abbiamo il vantaggio di poter regolare le dimensioni dell'immagine:
`<img src="path" alt="titolo_immagine" height="pixel" width="pixel"/>`
Per esempio, con
`<img src="/home/vittorio/Immagini/logaritmo.png" alt="grafico" height="200" width="200"/>`
inserisco nel documento la stessa immagine di prima ridotta a 200 pixel per 200.

Per l'audio usiamo la sintassi del linguaggio HTML
`<audio src="path_al_file" controls/>`

Per il video usiamo pure la sintassi del linguaggio HTML
`<video width="pixel"_height="pixel" src="path_al_file" controls/>`

Le immagini inserite nel documento le troviamo presenti in tutte le esportazioni del documento stesso, siano esse in formato HTML, siano esse in formato PDF.

I file audio e video compariranno con una inutile traccia nelle esportazioni in PDF via HTML e questa traccia nemmeno si vedrà nelle esportazioni in PDF via LaTeX. Saranno invece ben presenti e potranno funzionare facendoci udire suono o facendoci vedere immagini soltanto dal formato HTML, a patto che i relativi file siano raggiungibili. Un modo per assicurarci questo è di aprire il file `.html` esportato da Jupyter Notebook e salvarlo con SALVA PAGINA CON NOME...: ciò crea una copia del file `.html`, alla quale possiamo dare un nuovo nome, e una directory, con lo stesso nome, contenente i file multimediali inseriti nel documento. Trasferendo file e directory, insieme, possiamo riprodurre i nostri contenuti multimediali dove vogliamo.

5 Esempio

Questo è un notebook con evidenziato quanto inserito nelle varie finestre per ottenere il testo e i codici eseguibili

```
### Documento dimostrativo
In questo notebook vediamo:

* come si possa creare un grafico di funzione
* come si possa calcolare una derivata

utilizzando il linguaggio Python 3.

Cominciamo dal grafico di funzione:
```

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(-5,5,50)
y = x**2
plt.plot(x,y,color="green")
plt.title("Grafico di funzione")
plt.grid()
plt.show()
```

```
Ora passiamo al calcolo della derivata:
```

In [ ]:
```
from sympy import *
x = symbols('x')
e = sympify('4*x**2-5')
pprint(diff(e))
```

Vi si alternano due caselle di testo e due di codice.

Selezionando ciascuna casella ed eseguendone il contenuto attraverso la pressione dei tasti Shift+Invio il notebook assume la forma che vediamo nell'illustrazione all'inizio della pagina seguente.

Documento dimostrativo

In questo notebook vediamo:

- come si possa creare un grafico di funzione
- come si possa calcolare una derivata

utilizzando il linguaggio Python 3.

Cominciamo dal grafico di funzione:

In [4]:
```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(-5,5,50)
y = x**2
plt.plot(x,y,color="green")
plt.title("Grafico di funzione")
plt.grid()
plt.show()
```

Ora passiamo al calcolo della derivata:

In [5]:
```
from sympy import *
x = symbols('x')
e = sympify('4*x**2-5')
pprint(diff(e))
```

```
8·x
```

Qui vediamo che il testo è diventato testo formattato e sotto alle due finestre di codice vediamo il risultato dell'esecuzione del codice stesso: il grafico della funzione x^2 indicata nel codice `In [4]` e il valore $(8x)$ della derivata della funzione $4x^2 - 5$ indicata nel codice `In [5]`.

Questo è ciò che appare nel file `.ipynb` dove abbiamo memorizzato il notebook, file che è ancora uno strumento di lavoro: per noi stessi sul computer dove l'abbiamo creato e per chiunque, su un altro computer, abbia il software Python 3 arricchito dei moduli matplotlib e sympy importati nei due spezzoni di codice eseguibile, è possibile intervenire per modificare il testo, previa selezione della zona in cui è contenuto e doppio click sulla relativa finestra, o per modificare il codice eseguibile, per migliorarlo o per applicarlo diversamente.

Se, per esempio vogliamo che il grafico si riferisca alla funzione $y = sin(x)$ nell'intervallo tra 0 e 2π, basta che nella finestra `In [4]`:
. tra le importazioni indichiamo anche `from math import *`,
. sostituiamo la riga `x = np.linspace(0,2*pi,50)` alla riga `x = np.linspace(-5,5,50)`
. sostituiamo la riga `y = np.sin(x)` alla riga `y = x**2`.

Allo stesso modo possiamo calcolare derivate di altre funzioni modificando, nella finestra `In [5]`, l'espressione della funzione da derivare.

Ovviamente tutto questo non si può fare sui rendering del notebook dopo la sua esportazione in formato `.html`, `.tex` o `.pdf` nei quali è semplicemente visibile il contenuto statico del notebook.