Università degli Studi di Perugia



Dipartimento di Matematica e Informatica

Report di Laboratorio S.E.S.I

Python

|  |  |
| --- | --- |
| **Studenti**  Michal Horowski  Cristiano Moscetti Castellani |  |

Anno Accademico 2024-2025

Sommario

[Capitolo 1. Registrare e riprodurre suoni usando la libreria python Soundcard 4](#_Toc1220509266)

[1.1 Set-up di Soundcard 4](#_Toc858113795)

[1.2 Registrazione traccia audio 4](#_Toc1284550768)

[1.3 Riproduzione traccia audio 5](#_Toc705367868)

[1.4 Alterare riproduzione della traccia audio 5](#_Toc901745956)

[Capitolo 2. Creare grafici usando il modulo pyplot della libreria Python matplotlib 6](#_Toc412323726)

[2.1 La libreria matplotlib.pyplot 7](#_Toc1236351974)

[2.2 Introduzione all’uso di pyplot 7](#_Toc1412068020)

[2.3 Formattare lo stile di un plot 8](#_Toc1213130907)

[Capitolo 3. Registrare un suono e visualizzarne la forma dell’onda tramite Python 9](#_Toc1384908907)

[3.1 Registrazione e riproduzione 10](#_Toc1310308733)

[3.2 Preparazione dei dati 10](#_Toc214818437)

[3.3 Plot della waveform 10](#_Toc1159445526)

Introduzione

Useremo il linguaggio di programmazione Python insieme a delle librerie, per registrare suoni, riprodurli e visualizzare la forma d’onda di questi ultimi.

Nel primo capitolo vedremo come registrare i suoni e riprodurli tramite la libreria Soundcard, nel secondo capitolo useremo la libreria matplotlib.Pyplot per stampare a schermo dei grafici, nel terzo e ultimo capitolo uniremo le conoscenze acquisite dai capitoli precedenti per registrare suoni, riprodurli e disegnarne la forma d’onda o *waveform*.

# Registrare e riprodurre suoni usando la libreria python Soundcard

In questa sezione vedremo come acquisire e riprodurre audio usando il microfono e gli altoparlanti del nostro pc utilizzando la libreria dedicata di python “soundcard” commentando il relativo codice.

## Set-up di Soundcard

Per poter utilizzare la libreria “soundcard” è prima necessario scaricarla e installarla (su windows: pip install SoundCard)

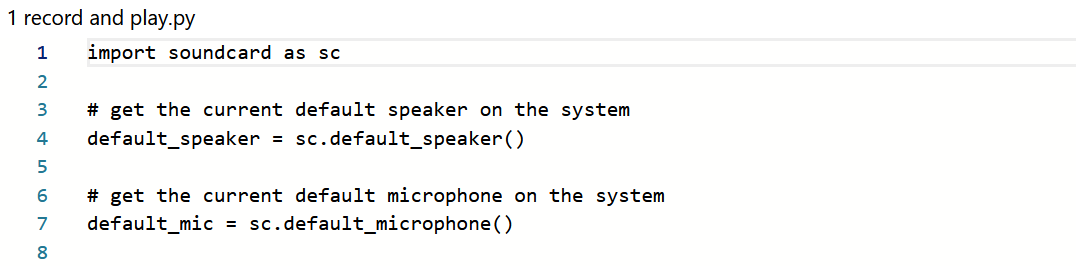


Figura 1.1-Codice per il set-up delle interfacce audio

Line 1 -- Importiamo quindi la libreria “soudcard” e per comodità lo abbreviamo con l’alias sc, usando le funzioni di questa libreria potremo acquisire e poi riprodurre suoni

Line 4 – Salviamo l’oggetto sc.default\_speaker() nella variabile default\_speaker per potrvi accedere successivamente al momento della riproduzione dell’audio

Line 7 – In modo analogo allo speaker ricaviamo l’oggetto sc.default\_microphone e lo salviamo nella variabile default\_mic per poter usare il microfono impostato di default del nostro pc e registrare una traccia audio

## Registrazione traccia audio

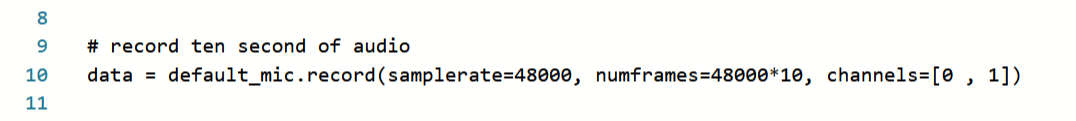


Figura 1.2-Codice per la registrazione del sample audio

Line 10 -- Salviamo la registrazione nella variabile “data” usando il microfono di default e la funzione record() di soundcard quindi: default\_mic.record(). Atraverso tre attributi la funzione ci permette di gestire il tipo di traccia audio in input:

Il parametro “samplerate” indica la frequenza di registrazione in Hertz, assegnando il valore di 48000 significa fare un campionamento di audio ogni secondi, ovvero 48000 campionamenti al secondo

“numframes” ci permette di gestire il numero di frame da registrare e quindi la durata della registrazione, nel nostro caso 10 secondi poiche assegnamo a numframes il “il numero di frame in un secondo” moltiplicato per 10 ()

L'attributo channels specifica su quali canali registrare, nel nostro caso sinistro (0) e destro (1), quindi audio stereo

## Riproduzione traccia audio

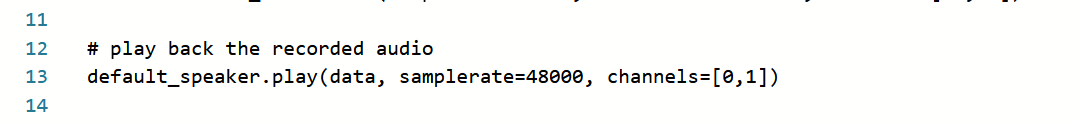


Figura 1.3-Codice per la riproduzione del sample audio

Line 13 – Applicando al nostro default\_speaker la funzione .paly() possiamo riprodurre una traccia audio, passiamo quindi come parametri della funzione la variabile “ data ” che he memorizzato l’audio precedentemente registrato, riproduciamo i campioni della traccia con la stessa frequenza della registrazione assegnando “” e riproduciamo l’audio in modalita stereo quindi “channels = [0,1]”

## Alterare riproduzione della traccia audio

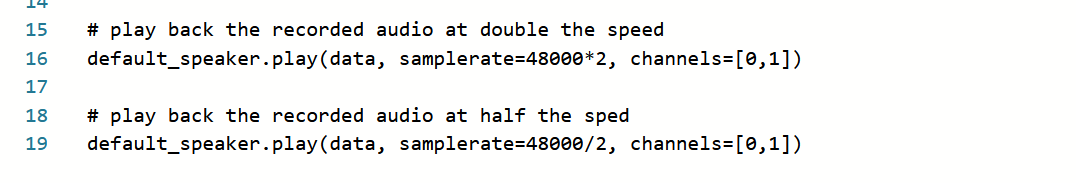


Figura 1.4Codice per la riproduzione del sample con velocità alterate

Nella funzione .play() modificando il valore attribuito a “samplerate” possiamo gestire la frequenza con cui vengono riprodotti i campioni audio.

Per esempio per raddoppiare la velocita di riproduzione si deve raddoppiare la frequenza di riproduzione dei campioni, se la registrazione e stata fatta a 48000 Hertz si devono riprodurre i campioni ad una frequenza di Hertz, ovvero 96000 Hertz:

Line 16

Una procedura analoga si puo applicare per riprodurre l’audio al rallentatore a velocita dimezzata, questa volta dovremo dividere per due la frequenza di riproduzione dei campioni:

Line 19

# Creare grafici usando il modulo pyplot della libreria Python matplotlib

In questo capitolo vedremo come creare dei grafici e stamparli a schermo, usando Python e la libreria Matplotlib, nello specifico il modulo Pyplot

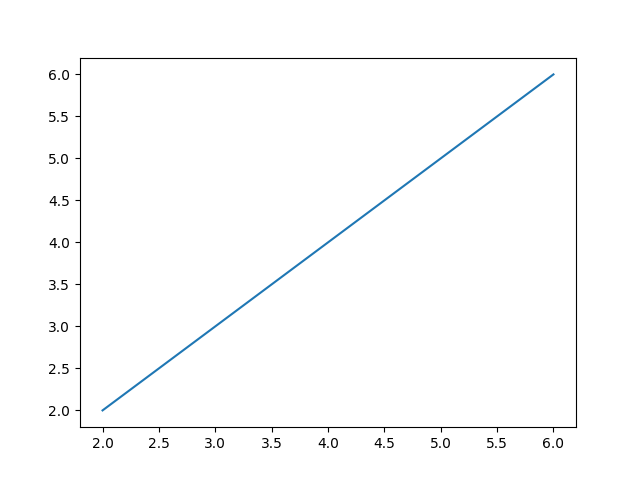
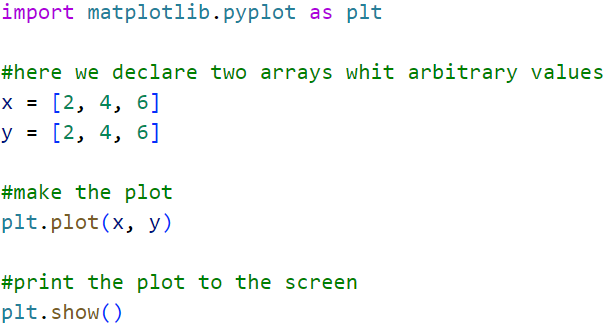
## La libreria matplotlib.pyplot

Matplotlib è una libreria open-source usata per creare grafici di vario tipo, come ad esempio grafici statici, grafici animati e grafici interattivi, mentre il modulo Pyplot appartenente a quest’ultima ci fornisce un interfaccia simile a quella del software MATLAB

## Introduzione all’uso di pyplot

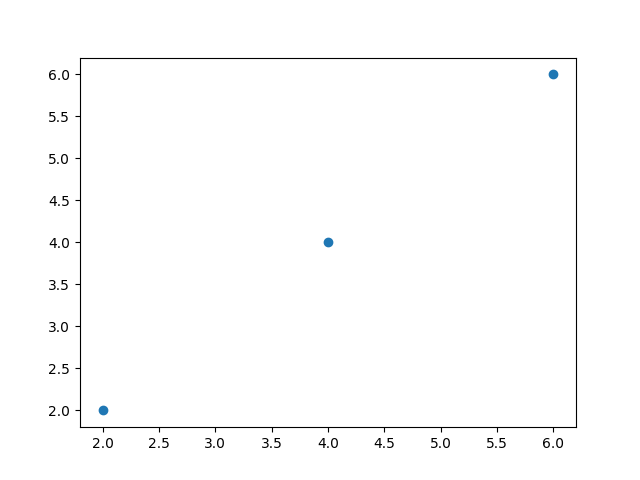
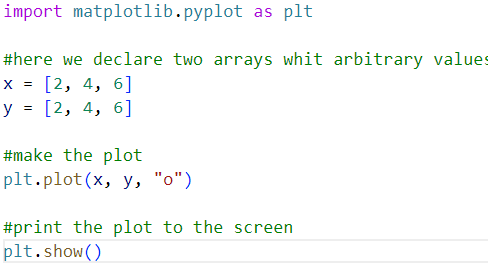
Prima di iniziare a scrivere il nostro script in python, dobbiamo inanzitutto installare il modulo usando pip(manager dei pacchetti python) tramite il comando “pip install matplotlib” ed importarlo nello script python tramite “import matplotlib.pyplot”, una volta fatto possiamo provare a disegnare o meglio dire “plottare” un grafico di esempio[**2.1**]

Figura 2.1. Plot con due array arbitrari



Nell’immagine [2.1] la linea è continua, questo è dovuto al comportamento di default di pyplot, per visualizzare i singoli punti dobbiamo aggiungere il parametro **“o”** all’interno di **plot(),** come nell’immagine[2.2]

Figura 2.2 Plot di tre punti



## Formattare lo stile di un plot

Per rendere più leggibile un plot è consigliabile aggiungere più informazioni possibili, come un titolo per far capire cosa si sta guardando, un etichetta su entrambi gli assi ed eventualmente una legenda, tutte le opzioni e parametri sono disponibili sulla documentazione uffciale di pyplot

Figura 2.3 Plot con opzioni di formattazione

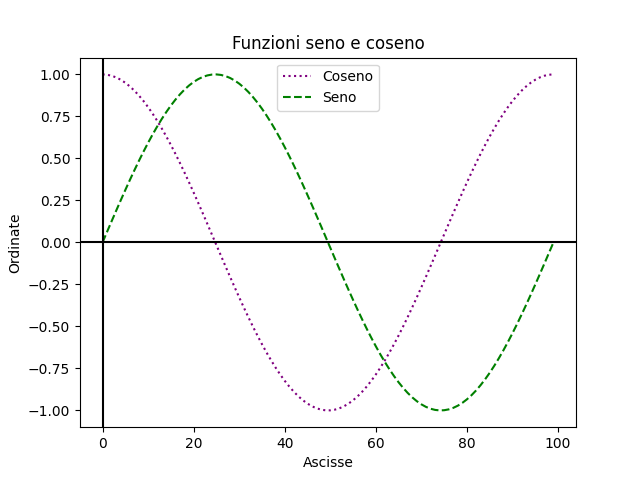
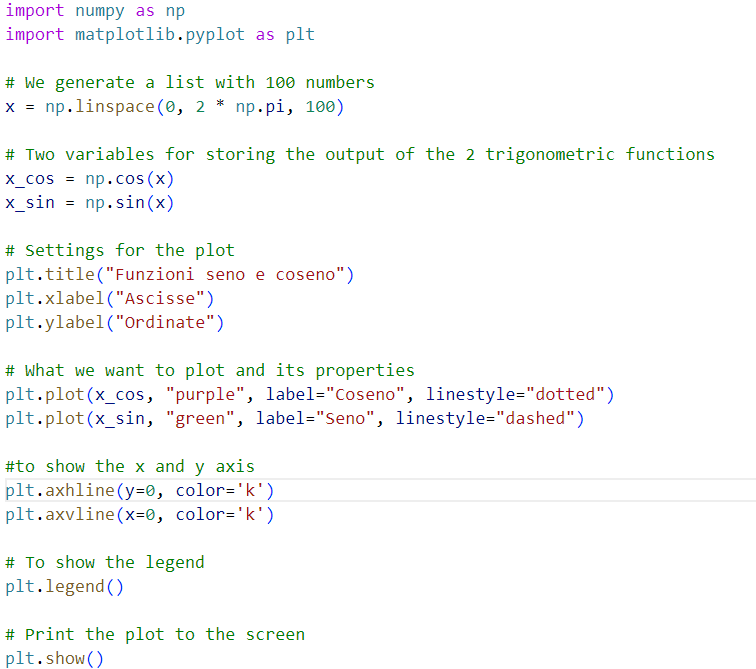


Figura 2.4 Plot di Sene e Coseno

Nell immagine [2.3] è presente il codice utilizzato per plottare [2.4], per generare l’array “x” è stata usata la libreria Numpy, succesivamente vengono configurati il titolo e le etichette sugli assi, le due funzioni vengono visualizzate tramite 2 diversi colori e tipi di liena, inoltre sono stati aggiunti anche gli assi e una legenda per una maggiore chiarezza, nell’immagine [2.5] possiamo vedere come non plottare un grafico

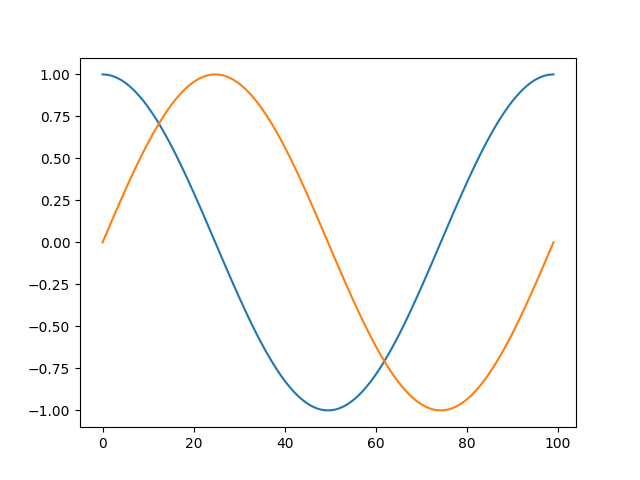


Figura 2.5 Plot senza alcuna formattazione

[] è lo stesso plot ma non è stata effettuata alcuna formattazzione, quindi non è chiaro cosa stiamo guardando.

# Registrare un suono e visualizzarne la forma dell’onda tramite Python

In questo capitolo combineremo cio che abbiamo visto nei due capitoli precedenti, allo scopo di visualizzare la forma d’onda della nostra registrazione

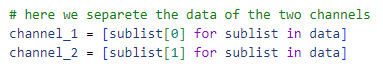
## Registrazione e riproduzione

Per questa parte non c’è nulla di nuovo, basta assegnare gli oggetti *default\_mic* e *default\_speaker*[1.1] a delle variabili, per poi chiamare il metodo *record()*, sul oggetto *default\_mic*, imagganzinando l’output in una varibile[1.2], per riproddure cio che è stato acquisito, dobbiamo passare al nostro oggetto *default\_speaker*, il dato(l’audio appena registrato), il samplerate ed infine quanti canali si vogliono riproddure[1.3]

## Preparazione dei dati

Dato che stiamo registrando su più di un canale, per visualizzare i dati in maniera dobbiamo prima separare i due canali, per farlo possiamo usare le list comprehension, per estrapolare rispettivamente il primo ed il secondo canale[3.1]

Figura 3.1



In caso di pù di due canali eseguire lo stesso procedeimento per i rimanenti

## Plot della waveform

Prima di plottare la waveform, creiamo dei subplot, per visualizzare in unico plot il primo canale, il secondo canale, ed infine i due canali sovrapposti[3.2]

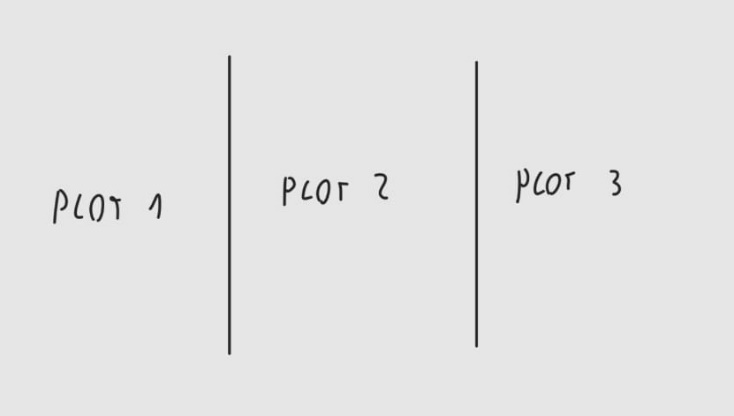


Figura 3.2

Infine per plottare i dati passiamo al primo plot, il primo canale , al secondo plot il secondo canale, e per visualizzare i due canli sovrapposti al terzo plot passiamo tutti e due i canali[3.3]

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Figura 3.3

Per selezionare su quale grafico operare usiamo *axis[]*, un array che contiene tre subplots, anche in questo caso vengono usati vari parametri per formattare il plot e renderlo più leggibile[3.3]

Immagine che contiene diagramma, Diagramma, linea

Descrizione generata automaticamente

Figura 3.4

Nell’immagine [3.4] possiamo vedere il risultato finale del codice [3.3] il plot numero 3 è pressoche identico al primo ed al secondo, in quanto non possendendo un microfono particolarmente “avanzato” non si riescono a distinguere a livello visivo, ma a livello di dati c’è una differenza, per vederla basta stampare i valori dei due canali e metterli a confronto

**Conclusioni**

In questa sessione di laboratorio, siamo riusciti ad apprendere come interfacciarci all’ I/O del nostro pc tramite la libreria soundcard, registrare e riprodurre suoni tramite ques’ultima per poi visualizzarne la forma d’onda tramite la libreria Pyplot

**Bibliografia**

1. [Documentazione ufficiale del linguaggio Python](https://docs.python.org/3/) 02/11/2024
2. [Documentazione libreria Soundcard](https://soundcard.readthedocs.io/en/latest/) 02/11/2024
3. [Documentazione ufficiale di matplot.pyplot](https://matplotlib.org/2.0.2/api/pyplot_api.html) 02/11/2024
4. [Documentazione della libreria Numpy](https://numpy.org/doc/) 02/11/2024