Tecnologie Digitali – AA 2025/26

Scheda di laboratorio n.00

Primi passi con MATLAB

Prima di cominciare con le misure vere e proprie, ci eserciteremo con le funzionalità di base di `MATLAB` per l'**analisi dei dati** e con una pseudo-scheda di rodaggio. In particolare, lo scopo sarà di capire come *importare* dei dati sperimentali con questo strumento *software* e come *manipolarli* per estrarne delle informazioni.

Task 1 Aprite `MATLAB` cliccando sulla *launch bar* di `Fedora`. Familiarizzate con la Graphical User Interface (`GUI`) del programma, verificando che siano presenti le finestre:

- la *Command Window* in cui è possibile inserire i comandi uno per volta;
- il *Workspace* dove vedrete visualizzate le variabili in memoria;
- il *Current Folder* che mostra la directory di lavoro attuale.

Se non fossero presenti, attivatene la visualizzazione utilizzando il bottone *Layout* nella barra dei comandi della *HOME* (vedere Fig.1.1).

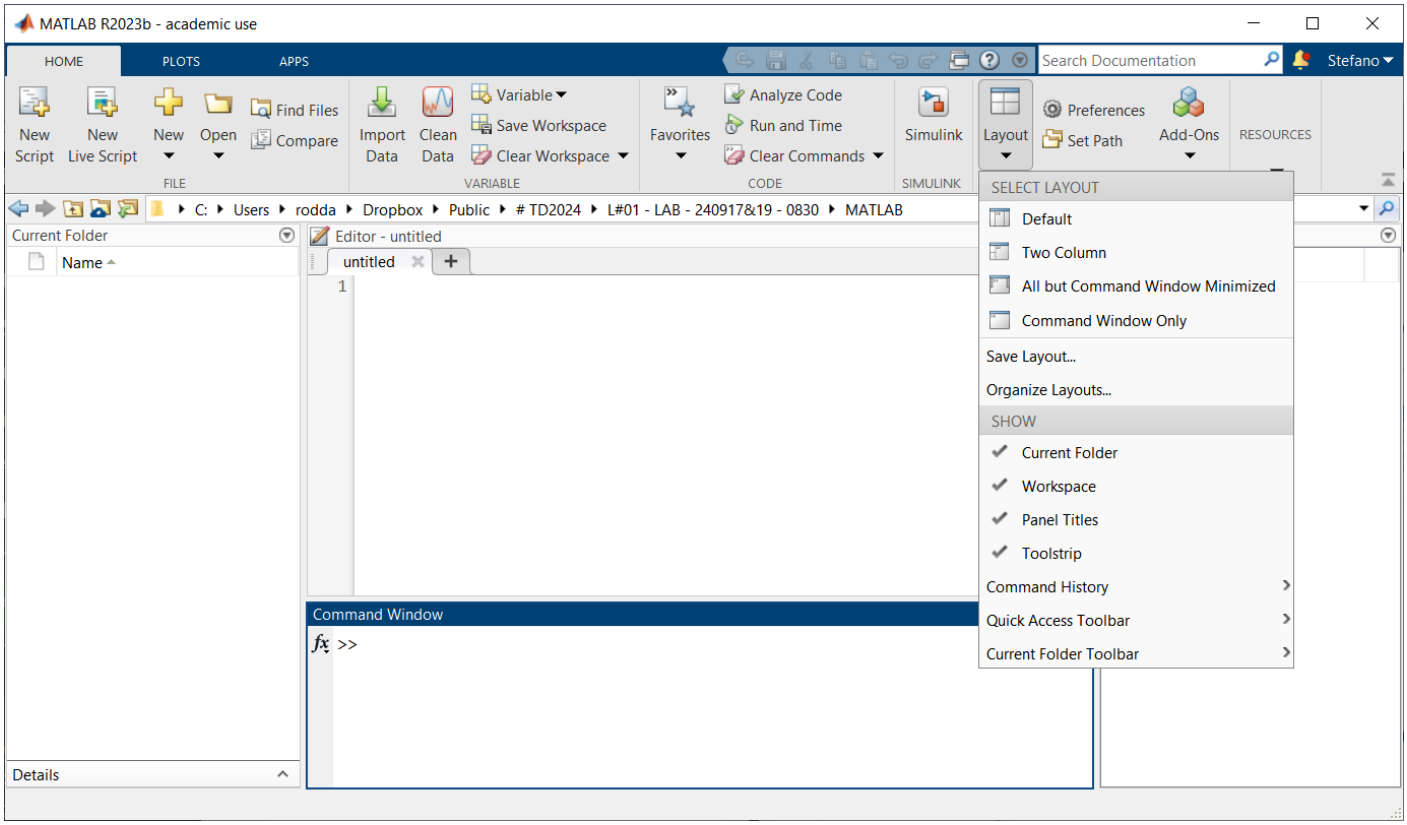

Figura 1.1: *Disposizione standard delle finestre nella `GUI` di `MATLAB`.*

Come abbiamo visto, `MATLAB` può essere usato direttamente come un calcolatore molto avanzato dove diverse operazioni e comandi possono essere direttamente eseguiti nella *Command Window*. Alternativamente, esistono due diverse modalità per scrivere un codice esteso che faccia diverse operazioni:

- il `LiveScript` – simile a `jupyter notebook` – permette di integrare codice `MATLAB` con testo formattato, immagini, ecc. Questa modalità richiede molte risorse e diventa farraginoso da utilizzare in macchine non ad alte prestazioni. Questa modalità verrà usata al più a scopo illustrativo.
- lo `Script` "tradizionale", che è un semplice file di testo di tipo `.m` che contiene una sequenza di comandi `MATLAB` da eseguire in sequenza. Questi possono essere editati sia all'interno dell'*Editor* della `GUI` di `MATLAB` che in altri programmi, come per esempio `VS-Code`.

Nella esercitazione di oggi l'obiettivo sarà di creare uno *Script* che, nell'ordine:

1. Importi un dato sperimentale da un file di dati. Nella fattispecie, useremo alcuni veri spettri di fotoluminescenza e di calibrazione raccolti su un campione di atomi di Rb.
2. Manipoli i dati sperimentali, per esempio eliminando un offset o trovando la posizione di picchi
3. Performi il *best fit* con delle funzioni modello, in modo da estrarre parametri numerici dalle curve sperimentali

1 Importazione dei dati

Nel corso del laboratorio, per ogni settimana di corso troverete una *directory* dedicata sulla sezione *Class Materials* del Team del corso. Nella fattispecie, troverete una directory chiamata `SOO - MATLAB`.

Task 2 Attraverso un browser, navigate alla pagina `https://teams.microsoft.com` per utilizzare `MS-Teams` online. Autenticandovi con le vostre credenziali UniPi dovreste già essere autorizzati ad accedere al canale del corso. Nella cartella della settimana, troverete il file `dati_Rb.txt`: scaricatelo e apritelo con un editor di testo per osservare come è strutturato.

I dati sperimentali sono organizzati all'interno di un file di testo codificato `ASCII` (*American Standard Code for Information Interchange*), in colonne di dati separate da virgola. Le colonne hanno un titolo (detto *header*) che descrive il dato presente nella colonna stessa. Nella fattispecie, dovreste trovare i seguenti header: `Voltage_V`, `Intensity_mW`, `Fabry-Perot_T_mW`.

Contenuto del file. Fare riferimento al finto log `LogRubidio.ipynb`, che riporta alcuni dettagli di base sulla misura, con un plot preliminare (come opportuno in un *log*). Non vogliamo entrare nei dettagli fini dell'esperimento, che sicuramente va al di là degli scopi di questa esercitazione di *elaborazione dei dati*, ma lasciamo qualche informazione utile.

1. L'esperimento consiste in una misura che evidenzia una serie di linee di fotoluminescenza dell'atomo di Rubidio (`Rb`), in funzione della lunghezza d'onda di un laser di eccitazione.
2. Per come è realizzato il set-up sperimentale, non possiamo davvero controllare la frequenza ma bensì un voltaggio che agisce sul *controller* del laser. Sfortunatamente, non è dato sapere con esattezza la corrispondenza fra questo voltaggio e la frequenza.
3. Lo stesso laser viene anche inviato in un Fabry-Perot, che sostanzialmente consiste in una cavità risonante fatta con due specchi che trasmette la luce solo a frequenze molto precise e distanti circa (qui assumeremo esattamente) 0.5 GHz.
4. Non vogliamo approfondire le questioni statistiche in questo momento, ma piuttosto lo scopo è semplicemente di fare pratica nella gestione di file, dati, analisi e grafici in `MATLAB`. Per questo motivo, tutte le analisi che vi verranno richieste andranno fatte assumendo errori costanti e relativi.

Task 3 Importare in `MATLAB` il file `dati_Rb.txt` in una matrice, di cui dovrete verificare le dimensioni per riconoscere la presenza dei 3 set di dati sopracitati. Usare una delle funzioni illustrate durante l'esercitazione, o se preferite testatene altre. Qui ne riportiamo alcune: `readmatrix`, `readtable`, oppure `importdata`. Ricordate sempre che potete facilmente accedere alla documentazione usando i comandi `help` oppure `doc`.

Task 4 Elaborare i dati importati secondo le seguenti regole:

- generare tre vettori per il voltaggio di controllo (`Vcont`), per lo spettro del Rubidio (`S_Rb`) e per lo spettro del Fabry-Perot (`S_FP`).
- Cambiare il segno del vettore `S_Rb` in modo che l'intensità dello spettro risulti (correttamente) positiva.
- Estrarre la posizione dei picchi del `S_FP` in voltaggio, utilizzando la funzione `findpeaks`. Leggere attentamente la documentazione per capire come estrarre il voltaggio di picco.
- Fare un fit lineare per estrarre la pendenza della posizione dei picchi in funzione dell'indice intero del numero del picco stesso.
- Utilizzare il risultato trovato per convertire il vettore `Vcont` da voltaggio a frequenza, creando un nuovo vettore `freq` e assumendo che la separazione in frequenza tra un picco e il successivo esattamente sia di 0.5 GHz.

Si noti che, per semplicità di pensiero, è sempre possibile utilizzare la funzione `nlinfit` per fittare qualsiasi cosa. Tuttavia, non è certo il massimo usarla nel caso di un semplice fit lineare. In questo caso è possibile anche usare funzioni più adeguate come `polyfit`. Anche in questo caso, non vi mostriamo esplicitamente come funziona ma rimandiamo alla documentazione di `MATLAB`.

Task 5 Graficare lo spettro del Rubidio in un grafico dell'intensità in funzione della frequenza relativa[a] e fittare la curva per estrarre frequenza di picco e allargamento delle 4 risonanze visibili. Identificare la funzione migliore di fit considerando multiple Lorentziane o multiple Gaussiane.

[a] relativa nel senso che non ci interessa il valore assoluto, che non possiamo conoscere con la procedura proposta e che non nemmeno particolarmente rilevante.

Lasciamo qualche indicazione finale, in particolare per l'esecuzione dei fit nel punto precedente.

- come illustrato negli esempi forniti, le funzioni di fit proposte non sono pre-esistenti in `MATLAB` e vanno definite dall'utente. Questo si può fare in vari modi fra cui: (1) definire una funzione in un file separato; (2) definire una funzione all'interno dello script stesso; (3) definendo una funzione in linea, usando una sintassi del tipo (qui il semplice caso di una retta...)

```
myfun = @(p,x) p(1) + p(2)*x
```

- chiaramente questo è un fit fortemente non-lineare e richiede l'utilizzo della funzione `nlinfit` (di nuovo, faremo una eccezione e chiediamo di non occuparsi degli errori o dei residui, a meno che non si abbia ampiamente finito tutto il resto... la procedura sostanzialmente considererà gli errori come costanti e relativi, e praticamente tenterà di stimarli ciecamente dalla variabilità del dato stesso rispetto alla funzione di fit) con sintassi

```
beta = nlinfit(x, y, myfun, beta0)
```

- Ricordiamo infine che per fare convergere un fit non-lineare come questo è importante partire da un *guess* iniziale sufficientemente vicino a valore di fit finale, altrimenti la convergenza è tutt'altro che garantita.

Task 6 Produrre un grafico finale `SpettroRb.pdf` fatto con tutti i crismi (eccetto errori): (1) etichette per gli assi, con una indicazione delle unità usate; (2) numeri leggibili; (3) grafico leggibile e scalato in maniera ragionevole; eccetera. Salvare anche lo script finale sul file `AnalisiRb.m`, sarà apprezzata la presenza di un livello ragionevole (né troppo, né poco, quello che pragmaticamente è utile) di commenti al codice. Riportare i risultati del fit sul *logbook*.