סדנת עיבוד נתונים

בסדנה זו תלמדו לעבד את נתוני המעבדה ב**עזרת שפת Python**. אנחנו נשתמש בתוכנה Spyder לכתיבה והרצה של הקוד. המטרה היא ליצור מיומנות בסיסית של ניתוח נתונים והצגתם בגרפים כדי שתוכלו לעבד את הנתונים של המעבדה בקלות. בנוסף, הקוד שתכתבו בסדנה יישמר אצלכם ותוכלו לחזור ולהיעזר בו בעתיד.

היקף הסדנה הוא **6 שעות – 3 בכיתה ו-3 בבית**, בסופן יהיו ברשותכם קבצי קוד שיכילו את כל הפקודות שלמדתם ותוכלו לעשות בהם שימוש במהלך המעבדה. במפגש בכיתה נגיע עד וכולל ניסוי מיפוי פוטנציאל.

תוכן הסדנה

חלק א' – התקנת Anaconda ו-Spyder

בצעו התקנה של **Anaconda**:

https://docs.anaconda.com/anaconda/install/windows/

התקנה זו מתקינה גם Spyder.

חלק ב' – סביבת עבודה ב-Spyder

הערכת זמן: כ- 15 דק'

1. **צרו** תיקית קבצים ריקה שתוקדש לסדנה.
2. **פתחו** את Spyder.
3. הסתכלו על הממשק של Spyder **וזהו** את החלקים הבאים:
   a. סרגל פקודות ותפריטים, תיקיית ההרצה.
   b. חלון הקובץ – אזור כתיבת הקוד.
   c. **ה-Console**, שימו לב שישנם **שני Tabs** לחלון זה.
   d. חלון עזרה, שימו לב שישנם **4 tabs: Help, Variable Explorer, Plots, Files**
4. **שנו** את תיקיית ההרצה לתיקיית הסדנה – כדי שהקוד יוכל למצוא את הקבצים שנמצאים בה.
5. **צרו קובץ חדש tutorial.py** ושמרו אותו בתיקיה שתוקדש לסדנה.
6. **עיברו** עם העכבר מעל האייקונים של פקודות ההרצה

וקראו את קיצורי המקשים של הרצת קוד ב-Spyder.

נכתב ב-17/2/2022 על ידי גיל בן ארי

חלק ג' – שימוש בסיסי ב-Python

הערכת זמן: כ-45 דקות

היכנסו לאתר https://www.learnpython.org/en/Welcome.

כאן נלמד את השיעורים המוגדרים *כלימודי הבסיס וניתוח נתונים* -

אנחנו נבצע את השיעורים עד וכולל Pandas, לפי הרשימה (שימו לב, **מדלגים** על חלק מהשיעורים). אנחנו נקדיש לכך כ-45 דקות. במידה ויש צורך תוכלו להמשיך בבית.

Learn the Basics

- Hello, World!
- Variables and Types
- Lists
- Basic Operators
- String Formatting
- ~~Basic String Operations~~
- Conditions
- Loops
- Functions
- ~~Classes and Objects~~
- ~~Dictionaries~~
- ~~Modules and Packages~~

Data Science Tutorials

- Numpy Arrays
- Pandas Basics

דגשים חשובים (מאוד):

1. במקום להריץ באתר – **העתיקו** את הקוד לקובץ ב-Spyder **והריצו** את הפקודות שם. באופן זה תשמרו את כל מה שלמדתם.
2. אם אתם מבינים את דוגמת הקוד, אתם לא חייבים להריץ אותה. תעתיקו ותריצו רק מה שאתם רוצים לוודא שאתם מבינים.
3. **התמקדו** בלהבין את הדוגמאות (ופחות בהבנת הטקסט).
4. **בצעו** את התרגיל בסוף כל שיעור באתר (ב-Spyder).
5. מי שמכיר את כל התכנים האלו – שיעבור לבצע את חלק ד' של הסדנה. אין צורך באישור המדריך.

*לאחר הסדנה, אתם מוזמנים לבצע את שאר השיעורים אם אתם רוצים.

שמרו את הקוד שכתבתם בתיקיה.

הלינק אצלכם, הקוד אצלכם, תוכלו להמשיך ולהתקדם בכל זמן שתרצו, **הכול זמין עבורכם**.

חלק ד – עיבוד נתוני המעבדה

באופן כללי, הפעולות הנדרשות לעיבוד הנתונים הן:

1. כתיבת וקטור נתונים (למדתם בשיעור Numpy Arrays)
2. חישוב עקום תיאורטי
3. טעינת נתונים מקובץ
4. הצגת נתונים בגרף
5. ביצוע חישוב על וקטורי נתונים (כתיבת פונקציה)
6. ביצוע רגרסיה לינארית
7. ביצוע fit (לעקום לא לינארי)
8. ביצוע אינטגרציה לווקטור נתונים

בחלק זה של הסדנה תלמדו ותבצעו את כל הפעולות האלה, על נתונים מניסויי המעבדה.

כדי לעבד את הנתונים, נשתמש בmodules הבאים:

```
iimport numpy as np # math functions

import scipy # scientific functions

import matplotlib.pyplot as plt # for plotting figures and setting their properties

import pandas as pd # handling data structures (loaded from files)

from scipy.stats import linregress # contains linregress (for linear regression)

from scipy.optimize import curve_fit as cfit # non-linear curve fitting

from sklearn.metrics import r2_score # import function that calculates R^2 score
```

1. **פתחו** קובץ קוד חדש בשם **data_analysis.py**
2. **העתיקו** את השורות האלה לתחילת הקובץ של הקוד **והריצו** אותן.

נתחיל לעבור יחד על ניסוי המעבדה באמצעות רקע קצר על מנת שתוכלו לנתח את הנתונים

תרגיל - מיפוי פוטנציאל חשמלי

רקע תאורטי

בניסוי מיפוי פוטנציאל חשמלי אתם תמדדו את הפוטנציאל החשמלי V על דף מוליך שמחוברות אליו אלקטרודות. בתרגיל הזה אנחנו נשרטט את הפוטנציאל החשמלי התיאורטי של שתי אלקטרודות ונחשב במפורש את שינוי הפוטנציאל מאלקטרודה אחת לשנייה.

נניח שישנן שתי אלקטרודות (חיובית ושלילית), בנקודות $(a,0)$ ו-$(-a,0)$. הפוטנציאל מסביב לאלקטרודה בודדת הוא:

$$V(r) = V_0 - C \cdot \ln\left(\frac{r}{r_0}\right)$$

כאשר מתקיים (מהגדרה זו) $\mathrm{V}(\mathrm{r}_0) = V_0$. הפוטנציאל משתי אלקטרודות הוא סכום הפוטנציאלים מכל אחת מהן. נבחר את r_0 להיות ראשית הצירים, איפה ש- $\mathrm{V}_0 = 0$ ונקבל

$$\mathrm{V}(x,y) = -C \cdot \ln\left(\frac{\sqrt{(x-a)^2+y^2}}{a}\right) + C \cdot \ln\left(\frac{\sqrt{(x+a)^2+y^2}}{a}\right)$$

המשימה - נשרטט את הפוטנציאל.

1. **פתחו** *cell* חדש בקוד (*#%%*) בשם *potential*
2. **העתיקו** את השורות הבאות:

```
C = 1
a = 1
L = 3
N = 100
coord = np.linspace(-L, L , N) # defines coordinates
coord_x, coord_y = np.meshgrid(coord, coord)
```

3. **הריצו** את הקוד **והסתכלו** ב-*Variable Explorer* על משתני הקואורדינטות (*coord_x, coord_y, coord*). מה הפונקציה *linspace* ביצעה? מה הפונקציה *meshgrid* ביצעה (ניתן להסתכל בתיעוד/הסברים באינטרנט)?
4. **כתבו** פונקציה שמחשבת את הפוטנציאל: *def potential(x, y, a, C)*. (**היעזרו** ב: *np.log()*, *np.sqrt()*, *np.power()* לפי הצורך)
5. **חשבו** את הפוטנציאל בקואורדינטות הנתונות **וציירו** אותו באמצעות השורות הבאות:

```
V_xy = potential(coord_x, coord_y, a, C)

plt.figure()
plt.pcolormesh(coord_x, coord_y, V_xy)
plt.colorbar()
```

6. **הריצו** את הקוד **והוסיפו** הערות לכל שורת קוד מסעיף 5 (עם #) שמסבירות מה כל שורה עושה.
7. כעת נוסיף לגרף קווים שווי-פוטנציאל, בהם הפוטנציאל הוא בעל ערך קבוע. כלומר, כמו במפות טופוגרפיות בהן מסומנים קווים שווי גובה, נרצה לצייר על הגרף קווים בהם ערך הפוטנציאל קבוע. לשם כך נשתמש בפונקציה *contour* שמבצעת את החישוב הזה ומציירת קווים בערכי הפוטנציאל שנגדיר לה. *contour* מקבלת את נתוני הפוטנציאל -*x,y,V(x,y)*- ואת הרמות *levels* של הפוטנציאל שאנו רוצים לצייר.
הוסיפו לקוד שלכם את השורה הבאה:

```
plt.contour(coord_x, coord_y, V_xy, np.sort([-1 , 0 , 1]), cmap='hot')
```

ושנו את הקוד כך שיצייר לפחות 9 קווים שווי פוטנציאל בערכים שונים (לא חובה שיהיו במרווחים אחידים, והם לא חייבים לכלול את *-1,0,1*). שימו לב ש-*contour* צריכה שה-*levels* יופיעו בסדר עולה, אך הפונקציה *sort* מבטיחה זאת עבורנו.

הריצו את הקוד **ורשמו** בהערה מה הפרמטר *cmap* קובע.

כעת נצייר את הפוטנציאל <u>שבין שתי האלקטרודות</u> (על ציר *X*):

8. **הכינו** וקטור מתאים של *x*, **חשבו** עבורו את הפוטנציאל *V_x* וציירו <u>בגרף חדש</u> את העקום בעזרת *plt.plot(x,V_x,'.', label="calculated potential")*

<u>מה למדנו?</u>

סימלצנו פוטנציאל חשמלי (יצרנו עקום תיאורטי) והצגנו אותו בשלושה סוגי גרפים: (*plot, pcolormesh, contour*).

תרגיל - קבל לוחות

רקע תאורטי

קיבול הוא היחס בין המטען החשמלי למתח של גוף מוליך:

$$C = \frac{Q}{V}$$

הקיבול של קבל לוחות עם לוחות עגולים בקוטר D ומרווח d בין הלוחות הוא:

$$C = \frac{\epsilon_0 A}{d} = \frac{\epsilon_0 \pi D^2}{4d}$$

כאשר ϵ_0 הוא המקדם הדיאלקטרי של ריק. כשנותנים לקבל *C* להיפרק דרך נגדים בעלי התנגדות כוללת של R_{total}, המתח שעל הקבל דועך בצורה אקסופננציאלית:

$$V_c(t) = V_0 e^{-\frac{t}{\tau}}$$

כש-$\tau = R_{total} \cdot C$ הוא קבוע הדעיכה. אם נבצע *ln* לשני האגפים נקבל קשר לינארי:

$$(\ln V_c)(t) = \ln V_0 - \frac{1}{\tau} \cdot t$$

הזרם שזורם בקבל זורם גם בנגד *R* ולכן שווה למתח הנגד חלקי ההתנגדות:

$$I(t) = \frac{V_R(t)}{R}$$

מכיוון שהמטען הוא אינטגרל של הזרם, מקבלים שהשינוי במתח הקבל הוא:

$$\Delta V_C(t) = V_C(t) - V_C(0) = \frac{1}{C}\int_0^t I(t')dt' = \frac{1}{RC}\int_0^t V_R(t')dt'$$

המשימה: עיבוד נתוני הניסוי

1. **פתחו** cell חדש בקוד בשם capacitor
2. נתון: $D = 18cm, d = 0.5mm$. **חשבו** את הקיבול C_theoretical

```
eps0 = scipy.constants.epsilon_0 # F/m
D = 18e-2 # m
d = 0.5e-3 # m
```

3. נתון: $R = 977\ Ohm, R_{total} = 38.4\ kOhm$. **חשבו** את tau_theoretical

הקובץ capacitor.csv מכיל מדידות של הפריקה:

time (sec)	ch1	ch2
0.00E+00	3.858477	-0.08171
0.00E+00	3.793352	-0.08058
1.00E-06	3.706517	-0.08058
1.00E-06	3.641392	-0.08115
2.00E-06	3.576266	-0.08002
2.00E-06	3.532849	-0.07777
2.00E-06	3.446015	-0.07664
3.00E-06	3.380889	-0.07777
3.00E-06	3.337472	-0.07383

המתח על הנגד R הוא ch2 והמתח על הקבל הוא ch1-ch2.

4. **טענו** את הנתונים שבקובץ באופן הבא:

```
C_data = pd.read_csv('capacitor.csv')
```

הסתכלו ב-Variables Explorer איזה מידע מופיע ב-C_data.

5. **שנו** את השם של עמודת הזמן ל- "t" ואת השם של "ch2" ל-"V_R":

```
C_data = C_data.rename(columns = {"time (sec)":"t", "ch2":"V_R"})
```

6. **הוסיפו** עמודה של V_C ל-C_data:

```
C_data["V_C"] = C_data["ch1"] - C_data["V_R"]
```

7. **ציירו** את מתח הקבל כתלות בזמן. יש לציין label.
(**רמז:** למדנו לצייר גרף בתרגיל מיפוי פוטנציאל חשמלי – אתם יכולים להעתיק את הפקודות משם).
<u>הערה:</u> ניתן לשמור את הערכים שבטבלה במשתנים עם שם נוח יותר על ידי:

```
t = np.array(C_data['t'].values)
V_C = np.array(C_data['V_C'].values)
```

אם בחרתם באפשרות זאת – שימו לב כיצד יש להתאים את הקוד בהמשך והקלות של השימוש =)

8. **כתבו** תשובה מלאה בהערה בקוד: מה זה curve fitting?

9. **בצעו** fit לגרף בעזרת הקוד הבא, המכיל כתיבה של פונקציה המתארת את המודל אליו אנחנו מבצעים את ההתאמה וקריאה לפונקציה curve_fit (אותה כינינו cfit בפקודת ה-import בתחילת הקוד):

```
def V_decay(t,tau,V0):
   return V0*np.exp(-t/tau)

p_optimal, p_covariance = cfit(V_decay,C_data['t'], C_data["V_C"])
```

10. **רשמו** בהערה לשורה שמבצעת cfit מהם הפרמטרים שהפונקציה צריכה למצוא.
11. **חשבו** מתוצאת ה-cfit את V0_fit ואת tau_fit.
12. **קראו** בתיעוד של cfit מה המשמעות של p_covariance **וחשבו** את השגיאה ב-V0_fit ו-tau_fit.
13. **הוסיפו** לגרף הנתונים את עקום ה-fit, והוסיפו מקרא (legend) באופן הבא:

```
plt.plot(C_data['t'], V_decay(C_data['t'],p_optimal[0],p_optimal[1]),
         label='fitted curve')

plt.legend()
```

האם העקום של ה-fit מתאים למדידות?

14. קראו בתיעוד של cfit על הפרמטר p0 והשתמשו בו עם הערכים המתאימים. עדכנו את הקריאה לcfit עם הניחוש ההתחלתי והריצו גם את שאר החישובים לאחריה. האם כעת העקום מתאים למדידות?
15. נתון שהשגיאה במתחים V_C היא 0.05V ושאין שגיאה ב-t. **חשבו** את הערך של סטטיסטי chi2 עבור ההתאמה (ראו הנחיות בחוברת אנליזת נתונים). כמה דרגות חופש יש?
16. **חשבו** את הערך של p-value עבור chi2 ודרגות החופש שחישבתם. היעזרו ב-scipy.stats.chi2.cdf(chi2, dof)
17. **חשבו** את מדד R^2 עבור עקום ה-fit בעזרת r2_score(y_measured,y_predicted_by_fit). האם הערך שהתקבל סביר לדעתכם?
18. על גרף נפרד, **ציירו** את לוג המתח כתלות בזמן. אמור להתקבל קשר לינארי. **והוסיפו** grid: plt.grid()
19. לפי הגרף, **בחרו** מקטע זמנים $[t1, t2]$ בו מתקיים הקשר הלינארי, **צרו** וקטור של אינדקסים של מקטע זה, **וציירו** את המקטע שבחרתם:

```
inds = (C_data['t'] > t1) & (C_data['t'] < t2)
plt.plot(C_data['t'][inds], np.log(C_data["V_C"])[inds],'.', label='data')
```

כעת נבצע רגרסיה לינארית כדי לחלץ את V0 ואת tau.

20. **כתבו** תשובה מלאה בהערה בקוד: מהי רגרסיה לינארית Linear Regression?
21. **בצעו** רגרסיה לינארית:

```
reg = linregress(C_data['t'][inds], np.log(C_data["V_C"])[inds])
print(reg)
```

22. **חשבו** מתוך reg.slope ו-reg.intercept את V0_reg ואת tau_reg

הערה: אם תרצו לחשב קשר לינארי ללא intercept (כלומר intercept=0 במדויק) יש להשתמש ב-cfit.

23. חשבו את השגיאות ב-V0_reg ו-tau_reg בעזרת reg.stderr ו-reg.intercept_stderr (שימו לב - יש לחשב שגיאות נגררות).
24. קראו על rvalue שמוחזר על ידי הפונקציה linregress **וחשבו** את R^2 (כלומר, ה- coefficient of determination) של הרגרסיה.
25. **הוסיפו** את עקום הרגרסיה לגרף (חשבו, כיצד אפשר לעשות זאת?) והוסיפו גם מקרא.
26. **בצעו** אינטגרציה למתח על הנגד ושמרו את התוצאה ב-C_data["int_V_R"]. (השתמשו ב-scipy.integrate.cumtrapz(V_R, x = t, initial = 0))
27. בגרף חדש, **שרטטו** את השינוי במתח הקבל ($\Delta V_C = V_C(t) - V_C(0)$) כתלות באינטגרל של מתח הנגד וציינו label. אמור להתקבל קשר לינארי. אם הוא לינארי רק במקטע – **הגדירו** את האינדקסים של מקטע זה ב-inds2.
28. **בצעו** התאמה מתאימה (שימו לב, צריך לבצע linregress או cfit?) **וחשבו** את הקיבול C_meas.
29. **הוסיפו** לגרף את עקום הרגרסיה, מקרא, grid וכותרות לצירים:

```
plt.xlabel("integral of V_R")
plt.ylabel("$\Delta V_C$")
plt.legend()
plt.grid()
```

התרשמו ממידת ההתאמה של הערכים התיאורטיים לערכים שחישבתם מהנתונים. הם צריכים להיות לפחות מאותו סדר גודל.

<u>מה למדנו?</u>

למדנו כיצד לטעון נתונים מקובץ, לבצע עליהם חישובים, להוסיף עמודות נתונים, לחתוך מתוך הנתונים את המקטע הרצוי, לבצע אינטגרציה, לבצע fit ורגרסיה לינארית ולעצב את הגרפים.

(אם אתם לא חושבים שזה מה שלקחתם מהתרגיל –אנחנו מציעים בחום, קראו שוב את הקוד שיצרתם - הכול שם)

שימו לב! כעת אתם יודעים לבצע את כל הפעולות הנדרשות לעיבוד הנתונים – ראו את <u>הרשימה מעלה.</u>

התרגילים הבאים הם תרגול של פעולות אלו.

תרגיל – חוק אוהם

רקע תאורטי

כאשר זורם זרם בנגד, ההספק המתבזבז עליו הוא

$$\mathrm{P}(t) = V(t) \cdot I(t)$$

והאנרגיה הזאת הופכת לאנרגית חום:

$$\Delta Q(t) = \int_0^t P(t')dt'$$

טמפרטורת הנגד עולה, לפי קיבול החום שלו:

$$\Delta Q(t) = C_{heat} \cdot \Delta T(t)$$

וכאשר הוא מתחמם, ההתנגדות של הנגד משתנה באופן לינארי:

$$\mathrm{R(t)} = \mathrm{R}_0\big(1 + \alpha \Delta T(t)\big)$$

כלומר, בסך הכול:

$$R(\Delta Q) = R_0 + \frac{\alpha R_0}{C_{heat}} \Delta Q$$

וזהו הקשר הלינארי אותו מוצאים בניסוי.

הקובץ *ohm.csv* מכיל את המדידות הבאות:

dress:	USB0::0x0957::0x179	Analog Channels				Function Channels	
del:	DSO-X 2002A	Time (s)	1 (VOLT)	2 (VOLT)		Time (s)	M1
ial Number:	MY54313430	-0.02291	-0.1595	-0.07363		-0.02291	-0.08578
nware Version:	02.65.2021030741	-0.02275	-0.1595	-0.07363		-0.02275	-0.08578
rt Time:	36:34.1	-0.02259	-0.1595	-0.07363		-0.02259	-0.08578
		-0.02243	-0.1595	-0.07363		-0.02243	-0.08578

כאשר המתח על הנגד המתחמם הוא המתח בערוץ 1 פחות המתח בערוץ 2, והזרם הוא המתח בערוץ 2 חלקי התנגדות R1 של 5.48 אוהם.

1. **צרו** section חדש בשם Ohm
2. **כתבו** פונקציה שמחשבת זרם: def I_R(V2, R1)
3. **כתבו** פונקציה שמחשבת את המתח שעל הנגד: def V_R(V1, V2)
4. **כתבו** פונקציה שמחשבת את התנגדות הנגד: def R_t(V_R, I_R)
5. **כתבו** פונקציה שמחשבת את ההספק שמתבזבז בנגד: def P_t(V_R, I_R)
6. **כתבו** פונקציה שמחשבת את האנרגיה שמחממת את הנגד: def Energy(P_t, t)
7. **טענו** את הנתונים מהקובץ: R_data=pd.read_csv("ohm.csv", header=1)
8. **רשמו** בהערה מה המשמעות של הפרמטר header ולמה הוא נחוץ כאן.
9. **קראו** מה הפרמטר usecols של הפונקציה read_csv מבצע **ותקנו** את השורה בקוד כך שרק העמודות של

   Time (s) 1(VOLT) 2(VOLT)

   יטענו לתוך המשתנה R_data.
10. **שנו** את שם העמודה של הזמן ל-t ואת שמות הערוצים ל-V1, V2.

11. **השתמשו** בפונקציות שכתבתם וחשבו את התנגדות הנגד כתלות בזמן ואת האנרגיה כתלות בזמן.
12. **שרטטו** גרף של ההתנגדות כתלות באנרגיה.
13. **בצעו** רגרסיה לינארית (לקטע הלינארי) והוסיפו את העקום לגרף.
14. **חשבו** את R_0 ואת $\frac{\alpha}{C_{heat}}$. **השוו** עם חברים לכיתה את התוצאה. האם קיבלתם ערכים דומים?

תרגיל - השראות

בניסוי זה אתם תעבדו על סט של קבצי נתונים – **<u>וזו חשיבות התרגיל!</u>**

רקע תאורטי

כאשר יש שינוי בשטף המגנטי שבתוך סליל, מתפתח עליו מתח שנקרא כא"מ (כוח אלקטרו מניע) לפי הנוסחה הבאה:

$$\varepsilon = -\frac{d\Phi_B(t)}{dt}$$

במערכת הניסוי, מפילים מגנט דרך זוג סלילים ומודדים את המתחים שמתפתחים עליהם. מכיוון שהמתח תלוי במהירות המגנט, צריך לאפיין את התנועה שלו – כלומר, את המהירות ההתחלתית ואת התאוצה שלו.

לשם כך, נמדדו חמישה סטים של מתחים [Trace 0.csv, …., Trace 4.csv], כאשר הסליל העליון (ref) הוחזק במיקום קבוע והסליל התחתון (signal) הורחק ממנו ב-

$$h = [30, 24, 18, 14, 8]$$

סנטימטרים.

נגדיר את זמן המעבר בכל סליל בתור הזמן בו השטף הוא מקסימאלי. אם נקבע את $t = 0$ להיות זמן המעבר בסליל הראשון, אז הזמן t_{coil} בו עובר המגנט את הסליל השני מקיים:

$$h = v_0 t_{coil} + \frac{a}{2} t_{coil}^2$$

וניתן לרשום:

$$\frac{h}{t_{coil}} = \frac{a}{2} t_{coil} + v_0$$

קבצי המדידה נראים כך:

Address:	USB0::0x0957::0x17	Analog Channels		
Model:	DSO-X 2002A	Time (s)	1 (VOLT)	2 (VOLT)
Serial Number:	MY54313425	-0.03188	-0.00509	-0.00647
Firmware Version:	02.65.2021030741	-0.03156	-0.00509	-0.00786
Start Time:	40:17.7	-0.03125	-0.00509	-0.00786
		-0.03094	-0.00509	-0.00832
Analog Sample Count:	960	-0.03063	-0.00509	-0.00786
Function Sample Count:	0	-0.03031	-0.00509	-0.00786
Waveform Memory Sample Count:	0	-0.03	-0.00462	-0.00832
Digital Sample Count:	0	-0.02969	-0.00416	-0.00786
		-0.02938	-0.00416	-0.00832
Setup Text Information:		-0.02906	-0.00416	-0.00786
ANALOG		-0.02875	-0.00416	-0.01017
Channel 1		-0.02844	-0.00416	-0.00878
Scale	184 mV/	-0.02813	-0.00416	-0.00971

כאשר ערוץ 1 הוא המתח בסליל הראשון, הקבוע, וערוץ 2 הוא המתח בסליל הזז.

1. **פתחו** *section* חדש בשם *inductance*
2. **הגדירו** בקוד את *h* בתור *numpy array* (*np.array([…])*) ושימו לב שהיחידות יהיו במטרים.
3. **טענו** את המדידות באמצעות לולאת *for*:

שימו לב לשם הקבצים.

```
Ind_data = []
for n in range(0,5):
        df = pd.read_csv('Trace %d.csv'%n, header = 1)
        Ind_data.append(df)
```

משלב זה, אתם מוזמנים לשנות את שורות הקוד שבתוך לולאת ה-*for* כדי שיבצעו את החישובים הדרושים לכל קובץ נתונים.

4. **שנו** את השמות של וקטורי הנתונים: זמן ל-*"t"*, ערוץ 1 ל-*"ref"* וערוץ 2 ל-"*signal*"
5. **ציירו** בגרף את האותות *signal* ו-*ref* כתלות בזמן (את הנתונים מכל חמשת הקבצים באותו הגרף).
6. **כתבו** פונקציה לחישוב השטף מגנטי מתוך מתח. כלומר *flux(voltage,time)*
7. **ציירו** בגרף את השטף המגנטי המחושב מהמתחים *ref* ו-*signal* כתלות בזמן (כל הנתונים בגרף אחד).
8. **חשבו** את האינדקס בווקטור השטף בו השטף מקסימאלי (בערך מוחלט), ניתן להיעזר ב-*np.argmax()*.
9. **הוסיפו** לגרף של השטף את הנקודות שמצאתם (*plot(t[ind_max]),flux[ind_max], 'ro'*). האם אלו הנקודות הנכונות?
10. לכל מדידה, **תקנו** את וקטור הזמן *t* כך שהזמן בו השטף מקסימאלי ב-*ref* מוגדר להיות *t=0*.
11. לכל מדידה, **חשבו** את זמן המעבר בסליל ה-*signal* (זמן בו השטף מקסימאלי, לאחר התיקון בסעיף הקודם ושמרו אותו בווקטור נתונים *t_coil*.
12. **שרטטו** גרף לינארי מתוך *h* ו-*t_coil* (לפי הנוסחה). מה צריך להיות ציר *X*? מה צריך להיות ציר *Y*?

13. נניח שגיאה של *0.1* ס"מ ב-*h* ושגיאה של *2* מילישניות בזמן *t_coil*. **קראו** את התיעוד של הפונקציה *matplotlib.pyplot.errorbar()* **וצרו** גרף שמציג שגיאות אלו (יש לחשב שגיאות נגררות - ראו חומרי עזר באתר הקורס).
14. **חשבו** את המהירות ההתחלתית ואת התאוצה בעזרת רגרסיה לינארית. האם התקבלה התאוצה של נפילה חופשית?
15. **הוסיפו** את עקום הרגרסיה לגרף עם השגיאות.
16. **חשבו** את מדד *R squared* עבור הרגרסיה הלינארית.
17. **חשבו** את מדד *chi squared* ואת ה-*p-value* שנובע ממנו עבור הרגרסיה הלינארית.

<u>מטלות סיכום:</u>

1. נהוג להגדיר את הפונקציות (*def*) בתחילת הקוד, כדי למנוע קריאות שגויות. עבור כל *section*, **מקמו** את הפונקציות שכתבתם בתחילת ה*section*.
2. **רשמו** הערות לכל *section* – מה מבוצע בפקודות השונות ולאילו סעיפים הוא מתאים. ההערות צריכות להיות מספיק ברורות כך שתוכלו להבין אותן בעוד חודש.
3. לכל *figure* שיצרתם, **הוסיפו** את החלקים הדרושים לפי ההנחיות שבחומרי העזר, לא כולל *CAPTION*. **הוסיפו** גם *grid*. להגבלת הצירים לתחום מסוים יש להשתמש ב-*xlim(), ylim()*.
4. **צרו** *section* **וכתבו** בו בהערות:
   a. רשימה של הפקודות שלדעתכם יהיו הכי שימושיות בניתוח נתונים במעבדה, מתוך הקוד שכתבתם עד כה.
   b. רשימה של הפקודות שמופיעות בקוד שלכם ושאתם לא בטוחים עד הסוף מה הן מבצעות.
5. <u>בונוס</u>: **הדפיסו** (*print*) את ערכי הקבועים *e, mu_0, epsilon_0,m_e* בעזרת *scipy.constants*

מזל טוב! הגעתם לסוף הסדנה!

כעת יש להגיש את הקוד שכתבתם במקום המתאים באתר.

שימו לב: קוד זה ישמש אתכם בעיבוד הנתונים במעבדה, ספציפית בניסויים **מיפוי פוטנציאל חשמלי**, **קבל לוחות**, **חוק אוהם והשראות** (ראו לוח זמנים), אבל גם בניסויים האחרים הקוד הדרוש הוא דומה. אחרי שעבדתם כל כך קשה, תצטרכו רק להעתיק ולהדביק את החלקים הרלוונטיים.

טיפ למתקדמים: אפשר לקרוא את ההנחיות לביצוע לפני מפגשי המעבדה ולהכין קוד מותאם מראש לכל מפגש 😉 – חוסך זמן במעבדה ומקל להבין את הנתונים.

בהצלחה לכם!