מגמת הנדסת אלקטרוניקה ומחשבים

תכנית לימודים במקצוע

מעבדת יישומי MATLAB באלקטרוניקה

כיתה י"ג

תכנית הלימודים במקצוע

מעבדת יישומיMATLAB באלקטרוניקה

כיתה י"ג

כתב: מרק טסליצקי

ינואר 2020

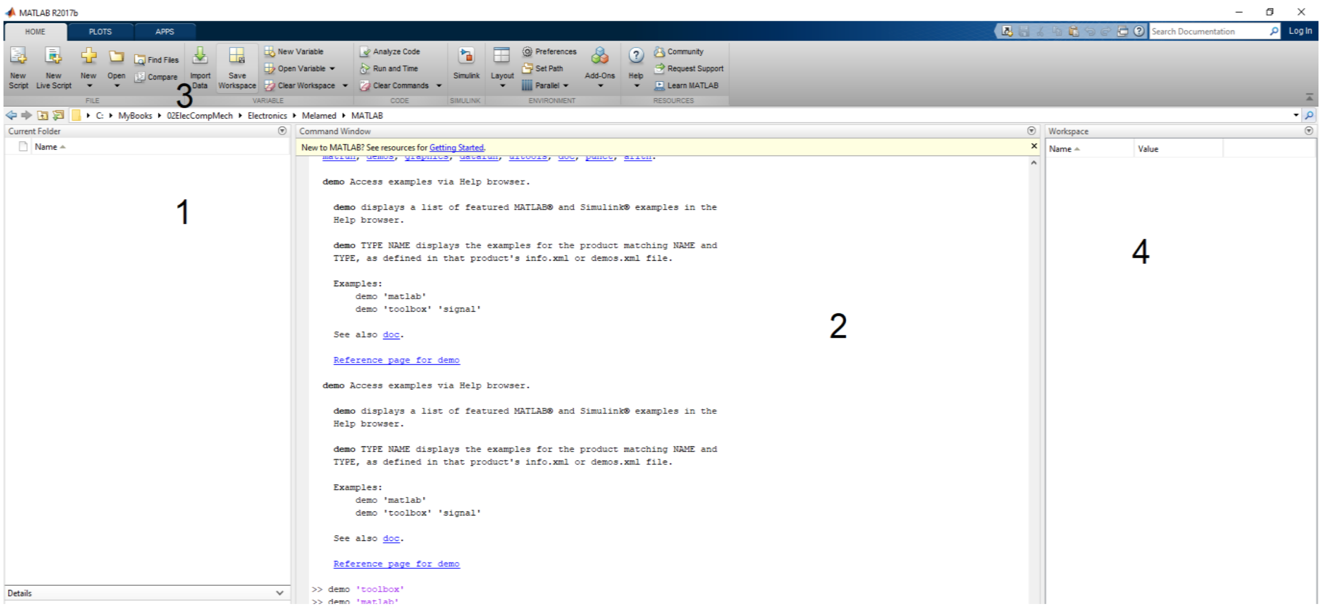
הגהה: יעקב ברזילי.

רעיון של עיצוב גרפי: גדי הרמן.

תוכן עניינים

פרק 1 - הכרת סביבת העבודה .

א. מבוא לסביבת MATLAB.



תוכנת MATLAB אינה עובדת בתור קומפיילר, אלא בתור Interpreter.

מה הבדל? קומפיילר מקבל קוד מהתחלה עד הסוף, בודק תקינותו ואז מריץ אותו, לעומת זאת Interpreter מקבל קוד ומפעיל אותו פקודה אחר פקודה.

בשלב מתקדם יותר נוכל לכתוב תוכנה בשפת C ולהפעיל אותה מסביבת MATLAB בנוסף לקוד הבסיסי.

נתחיל עם הכרת תפריט התוכנה.
שדה "1" – מכיל קבצים של ספרייה שאנו נמצאים בה.

שדה "2" – הינו שדה העבודה שלנו. כאן נכתוב פקודות ונקבל תוצאות. מה שנכתוב לא נקרא "תוכנה" אלא SCRIPT – סקריפט, במושג הזה נשתמש בכל הקורס שלנו.

בכדי לחזור על פקודה שכתבנו קודם נשתמש בחיצים הבאים: ↓ או ↑.

שדה "3" – הינו התפריט של פונקציות מיוחדות שאת תפקידן נכיר בעתיד.

שדה "4" – ניתן לראות בו פעולות קודמות ומשתנים שהגדרנו. פעולת >>clear מנקה את הזיכרון של שדה זה ואת כל המשתנים שהגדרנו קודם.

פקודת >>clc מנקה את כל חלון הפקודות, אם רוצים "לנקות" (למחוק) משתנה מסוים יש לרשום את שמו לאחר פקודת clear. לדוגמא הגדרנו x=5 ואז רוצים להסירו נרשום – clear x – פקודה זו תוציא אותו מרשימת המשתנים.

ב. ביצוע פעולות אריתמטיות .

עבודה עם מספרים רגילים:

קלט:

```
>> 2+3-4/5*6
```

פלט:

```
ans =
   0.2000
```

אופרטור של חזקה:

קלט:

```
>> 3^4
```

פלט:

```
ans =
   81
```

קלט:

```
>> 2^10
```

פלט:

```
ans =
       1024
```

יש לכתוב את הביטוי ללחוץ "ENTER" ונקבל תשובה.

סדר פעולות מתמטיות:

סוגריים ← חזקה ← פעולות כפל וחילוק משמאל לימין ← פעולות חיבור וחיסור משמאל לימין.

במקרה של הדוגמה הנתונה סדר הפעולות יבוצע כך: מחלקים 4 ב-5, מכפילים ב- 6 (מקבלים 4.8). מוסיפים ל-2 את 3 ומורידים מהתוצאה שמתקבלת את התוצאה של חישוב הקודם 5-4.8 = 0.2.

ג. משתנים בסביבת MATLAB.

הגדרת משתנים:

קיימים 5 סוגי משתנים בסביבת MATLAB:

הסבר	דוגמא	שם
מספר שלם כלשהו	0 ,111 ,-234 ,123	Integer
מספר לא שלם	0.123, -5.123, -95.12, 32.12	Real
מספר מרוכב	12.12-23.1i, -i, i , 2i,	Complex
אין סוף	1/0	Inf
מספר לא קיים	0/0	NaN

עבודה עם משתנים:

כעת נגדיר משתנים שונים וקצת נשחק איתם:

קלט:

```
>> x=12
```

פלט:

```
x =
   12
```

קלט:

```
>> y=3.1
```

פלט:

```
y =
   3.1000
```

קלט:

```
>> z=x+y
```

פלט:

```
z =
  15.1000
```

במידה ולא רוצים שהתוכנה תרשום לנו את תוצאות הקליטה בכל פעם שמכניסים נתון אפשר להשתמש בסימן ";"

קלט:

```
>> x=12;
>> y=3.1;
>> z=x+y
```

פלט:

```
z =
  15.1000
```

אם רוצים שהתוכנה תקבל ערכים ולא תחשב אותם, ניתן להשתמש ב- SHIFT + ENTER במקום ENTER :

קלט:

```
>> x=12;
y=3.1;
z=x+y
```

פלט:

```
z =

  15.1000
```

קלט:

```
>> x=12
y=3.1
z=x+y
```

פלט:

```
x =
   12
y =
   3.1000
z =
  15.1000
```

במידה ולא הגדרנו משתנה – נקבל הודעת שגיאה:

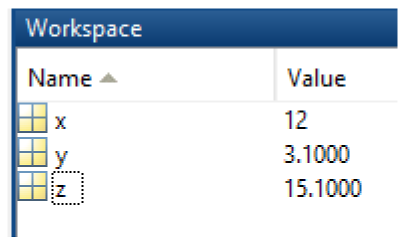
קלט:

```
>> w=e/1
```

פלט:

```
Undefined function or variable 'e'.
>>
```

בחלון המשתנים, הנתונים ייראו כך:



שימו לב שמשתנה w לא קיים עקב הוראת השגיאה.

כזכור, פקודת clc<< מנקה את כל חלון הפקודות, אם רוצים "לנקות" (למחוק) משתנה מסוים יש לרשום את שמו לאחר פקודת clear. לדוגמה הגדרנו x=5 ואז רוצים להסירו נרשום – clear x פקודה זו תוציא אותו מרשימת המשתנים.

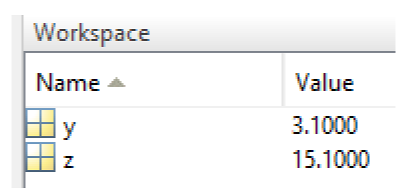
קלט:

```
>> clear x
>> a=x/2
```

פלט:

```
Undefined function or variable 'x'.
```

בחלון המשתנים, הנתונים ייראו כך:



שמות משתנים:

כמו בשפת C, שם המשתנה יכול להכיל אותיות, ספרות וסימן "_" אך תו ראשון של שם המשתנה חייב להתחיל באות. אורך המשתנה יכול להיות עד 63 תווים:

דוגמאות:

קלט:

```
>> 1_a=5
```

פלט:

```
1_a=5
 ↑
Error: The input character is not valid in MATLAB statements or expressions.
```

קלט:

```
>> a_1=5
```

פלט:

```
a_1 =
    5
```

קלט:

```
>> asasdfadfasdfasdfasdfasdfasdfadfasdfasd=1
```

פלט:

```
asasdfadfasdfasdfasdfasdfasdfadfasdfasd =
    1
```

קלט:

```
>> a!12=1
```

פלט:

```
a!12=1
 ↑
Error: Unexpected MATLAB operator.
```

קלט:

```
>>
asasdfadfasdfasdfasdfasdfasdfadfasdfasdasasdfadfasdfasdfasdfasdfasdfadfasdfasdasasdfadfasdfasdfasdfasdfasdf
adfasdfasd=1
```

פלט:

```
Warning:
'asasdfadfasdfasdfasdfasdfasdfadfasdfasdasasdfadfasdfasdfasdfasdfasdfadfasdfasdasasdfadfasdfasdfasdfasdfasdf
adfasdfasd'
exceeds the MATLAB maximum name length of 63 characters and will be truncated to
'asasdfadfasdfasdfasdfasdfasdfadfasdfasdasasdfadfasdfasdfasdfasd'.
asasdfadfasdfasdfasdfasdfasdfadfasdfasdasasdfadfasdfasdfasdfasd =
    1
```

כמו בשפת C גם כאן יש הבדל בין אותיות קטנות לגדולות:

קלט:

```
>> x=5
```

פלט:

```
x =
    5
```

קלט:

```
>> x
```

פלט:

```
x =
    5
```

קלט:

```
>> X
```

פלט:

```
Undefined function or variable 'X'.
Did you mean:
>> x
```

ניתן להגדיר משתנים בשורה:

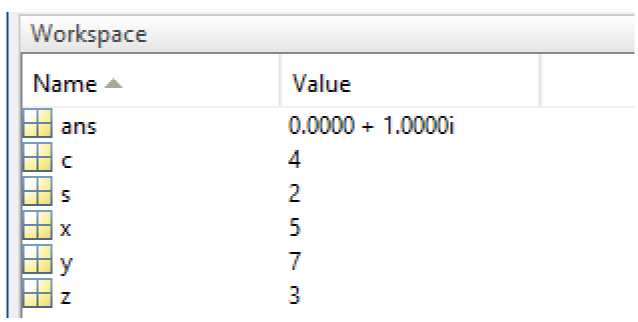
קלט:

```
>> x=5; y=7; z=3, s=2,c=4
```

פלט:

```
z =
    3
s =
    2
c =
    4
```

בחלון המשתנים, הנתונים ייראו כך:

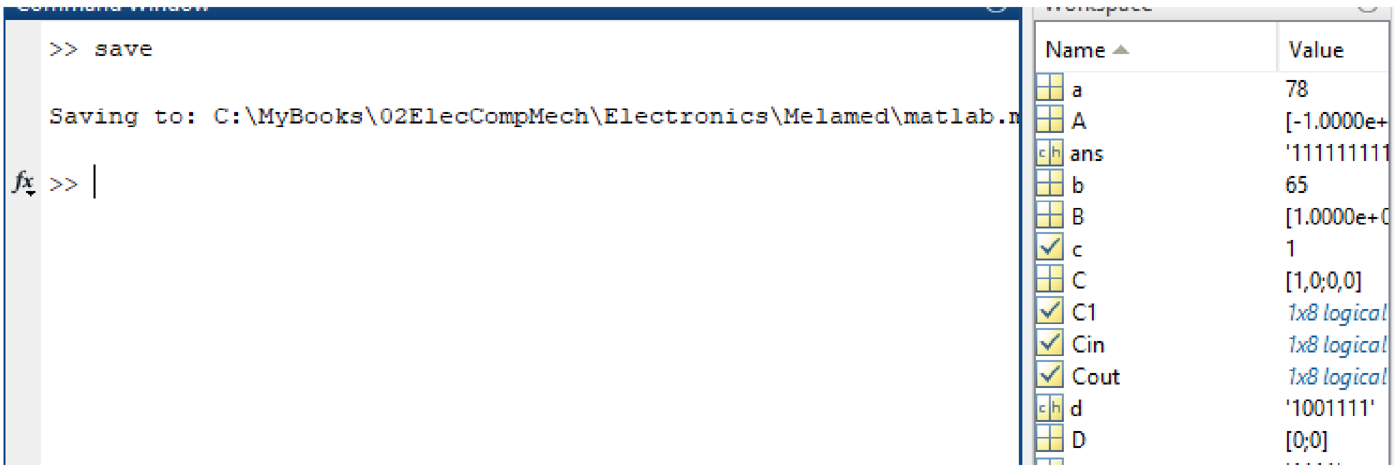


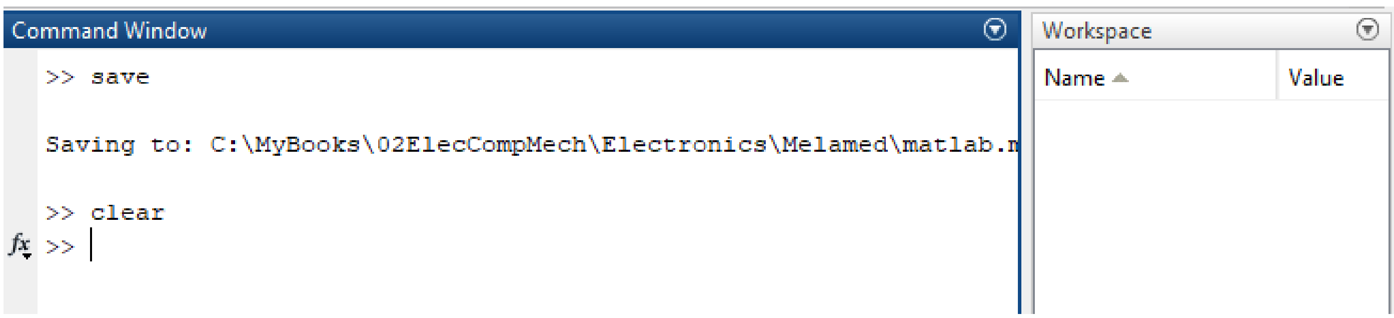
שימו לב שכאשר רושמים ";" לאחר שם המשתנה – המשתנה קיים ברשימה (מוצג בחלון עבודה), אבל לא מוצג חלון הפקודות ואם לאחר שם המשתנה יש "," הוא מוצג בשני חלונות.

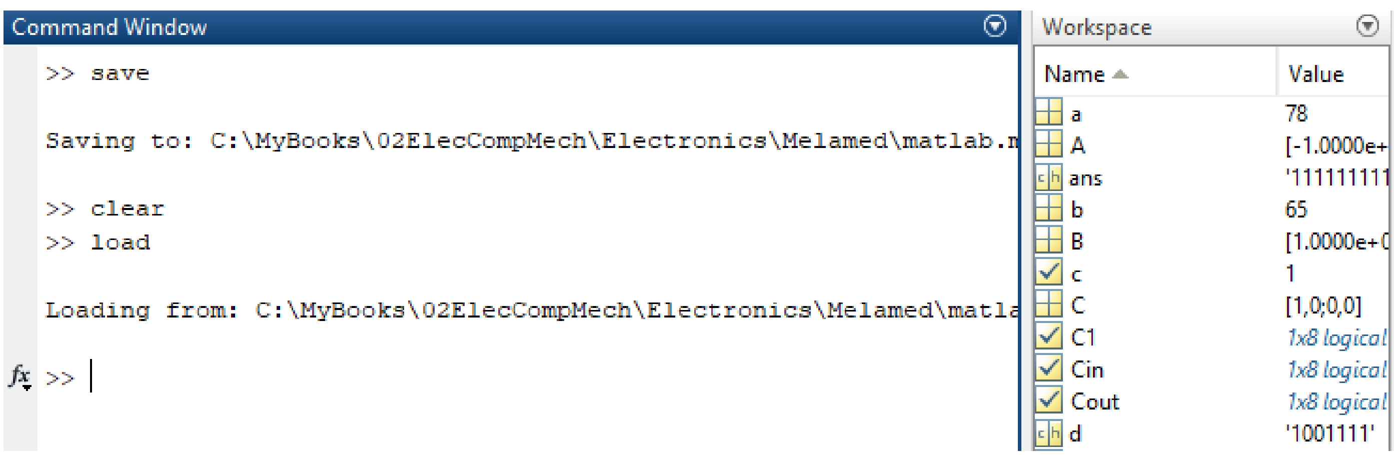
<u>טעינה ושמירת משתנים:</u>

לפעמים יש לנו צורך לשמור משתנים מסוימים בזיכרון, כדי שתהיה אפשרות לטעון אותם מחדש לאחר פעולות מסוימות או בפעם הבא שנפעיל את התוכנה מחדש.

כדי לשמור משתנים לדיסק משתמשים בפקודה save, כדי לשחזר אותם משתמשים בפקודה load.







ד. קבועים בסביבת MATLAB:

קלט:

```
>> eps
```

פלט:

```
ans =
  2.2204e-16
```

קלט:

```
>> pi
```

פלט:

```
ans =
   3.1416
```

קלט:

```
>> i
```

פלט:

```
ans =
  0.0000 + 1.0000i
```

קלט:

```
>> j
```

פלט:

```
ans =
  0.0000 + 1.0000i
```

ה. פעולות עם מספרים מרוכבים:

פונקציה real מחזירה ערך ממשי של מספר מרוכב:

קלט:

```
>> a=2-i
```

פלט:

```
a =
  2.0000 - 1.0000i
```

קלט:

```
>> real(a)
```

פלט:

```
ans =
    2
```

קלט:

```
>> isreal(a)
```

פלט:

```
ans =
  logical   0
```

פונקציה isreal בודקת האם המספר ממשי או מרוכב (האם יש לו חלק מדומה או אין). במקרה והמספר ממשי הפונקציה מחזירה "1" ובמקרה שלמספר יש גם חלק מדומה היא מחזירה 0.

פונקציה imag מחזירה ערך מדומה של מספר מרוכב:

קלט:

```
>> imag(a)
```

פלט:

```
ans =
   -1
```

פונקציה conj מחזירה "צמוד" של מספר – זאת אומרת למספר 2 – i היא תחזיר 2 + i.

קלט:

```
>> conj(a)
```

פלט:

```
ans =
  2.0000 + 1.0000i
```

פונקציה angle מחזירה זווית של מספר מרוכב ברדיאן. כדי לקבל זווית ברדיאן אנו צריכים להכפיל אותו ב-180 ולחלק ב-פאי.

(החישוב נעשה לפי הנוסחה $\alpha = \arctan\left(\frac{\text{Im}}{\text{Re}}\right)$)

קלט:

```
>> b=1+I
```

פלט:

```
b =
   1.0000 + 1.0000i
```

קלט:

```
>> angle(b)
```

פלט:

```
ans =
    0.7854
```

קלט:

```
>> ans*180/pi
```

פלט:

```
ans =
    45
```

קלט:

```
>> angle(a)
```

פלט:

```
ans =
  -0.4636
```

קלט:

```
>> ans*180/pi
```

פלט:

```
ans =
  -26.5651
```

פונקציה abs משחקת 2 תפקידים: במידה ומדובר במספר ממשי, הפונקציה מחזירה ערך מוחלט של המספר. במידה ומדובר במספר מרוכב, הפונקציה מחזירה את אורך הווקטור r שניתן לחשב לפי נוסחה: $r = \sqrt{\mathrm{Re}^2 + \mathrm{Im}^2}$

קלט:

```
>> x=-1
```

פלט:

```
x =
   -1
```

קלט:

```
>> abs(x)
```

פלט:

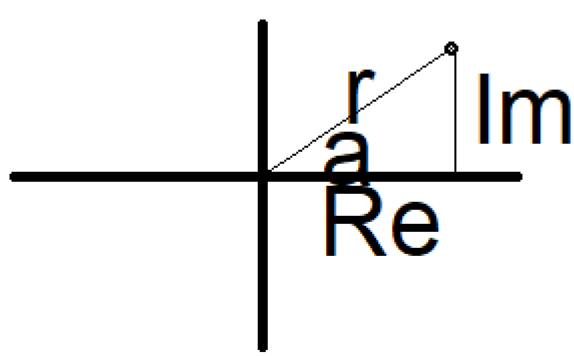
```
ans =
    1
```

קלט:

```
>> abs(a)
```

פלט:

```
ans =
   2.2361
```



המרת מספרים מייצוג קרטזי לפולרי ולהיפך.

על חלק פעולות דיברנו בסעיף א 1.1, כאן נמשיך לעבוד עם מספרים מרוכבים. שימו לב שכל הזוויות נתונות ברדיאן ואם אנחנו רוצים להעביר אותם למעלות יש לבצע פעולה נוספת. נראה את זה בהקדם:

פרוש פקודות:	קוד בתוכנת MATLAB
הגדרת משתנים מרוכבים בעזרת מחלקה בשם COMPLEX	קלט: `>> z=complex(3,-4)` פלט: `z = 3.0000 - 4.0000i`
הגדרה של מספר מרוכב בצורה ידנית	קלט: `>> z=2+3i` פלט: `z = 2.0000 + 3.0000i`
חישוב של אורך הווקטור לייצוג הנדסי	קלט: `>> R=abs(z)` פלט: `R = 3.6056`
חישוב של אורך הווקטור לייצוג הנדסי (זווית ברדיאן)	קלט: `>> theta=angle(z)` פלט: `theta = 0.9828`
חישוב של אורך הווקטור לייצוג הנדסי (זווית במעלות)	קלט: `>> theta=angle(z)*180/pi` פלט: `theta = 56.3099`
הפיכת מספר מרוכב מצורה הנדסית לצורה אלגברית. (שימו לב שבמקרה שלנו זווית במעלות ואנו צריכים להפוך אותו לרדיאנים לפני הפיכה).	קלט: `>> Z1=R*exp(i*(theta/180*pi))` פלט: `Z1 = 2.0000 + 3.0000i`

ביצוע פעולות אריתמטיות בסיסיות: חיבור, חיסור, כפל וחילוק

פרוש פקודות:	קוד בתוכנת MATLAB
הגדרת מספר ראשון	קלט: `>> z1=2+3i` פלט: `z1 = 2.0000 + 3.0000i`
הגדרת מספר שני	קלט: `>> z2=4-i` פלט: `z2 = 4.0000 - 1.0000i`
סכום של 2 מספרים מרוכבים	
הפרש של 2 מספרים מרוכבים	

<table>
<tr>
<td>כפל בין מרוכבים

חילוק בין 2 מספרים מרוכבים</td>
<td>קלט:
>> a=z1+z2
פלט:
a =
6.0000 + 2.0000i
קלט:
>> b=z1-z2
פלט:
b =
-2.0000 + 4.0000i
קלט:
>> c=z1*z2
פלט:
c =
11.0000 +10.0000i
קלט:
>> d=z1/z2
פלט:
d =
0.2941 + 0.8235i</td>
</tr>
</table>

תרגיל מסכם:

1.1 כתוב תוכנית לחיפוש שברים של משוואה ריבועית. (פונקציה למציאת שורש ריבועי – sqrt()).

רמז: שימו לב לביצוע פעולות בסדר נכון ולהשתמש בסוגריים איפה שצריך.

יש לבדוק את כל המקרים (2 שורשים ממשיים, שורש אחד, 2 שורשים מרוכבים).

ו. שימוש בפונקציות שבעזרתן ניתן לעגל מספרים.

פונקציות מיוחדות:

פונקציה ceil מעגלת מספר כלפי מעלה לכיוון פלוס אין סוף:

קלט:

```
>> a=1.234567
```

פלט:

```
a =
   1.2346
```

קלט:

```
>> b=-1.33
```

פלט:

```
b =
  -1.3300
```

קלט:

```
>> ceil(a)
```

פלט:

```
ans =
    2
```

קלט:

```
>> ceil(b)
```

פלט:

```
ans =
   -1
```

פונקציה fix מעגלת מספר לכיוון 0:

קלט:

```
>> fix(a)
```

פלט:

```
ans =
    1
```

קלט:

```
>> fix(b)
```

פלט:

```
ans =
   -1
```

פונקציה floor מעגלת מספר כלפי מטה לכיוון מינוס אין סוף:

קלט:

```
>> floor(a)
```

פלט:

```
ans =
    1
```

קלט:

```
>> floor(b)
```

פלט:

```
ans =
   -2
```

פונקציה round מעגלת מספר למספר שלם הקרוב ביותר .

קלט:

```
>> round(a)
```

פלט:

```
ans =
    1
```

קלט:

```
>> round(b)
```

פלט:

```
ans =
   -1
```

לפונקציה הזאת יש מקרה פרטי הבא:

round(X, N) למספרים חיוביים – מעגל מספר עד N ספרות אחרי נקודה:

קלט:

```
>> pi
```

פלט:

```
ans =
   3.1416
```

קלט:

```
>> round(pi,2)
```

פלט:

```
ans =
   3.1400
```

ז. פונקציות של ערך מוחלט ומספרים ראשוניים.

פונקציה sign מחזירה 1 אם מספר חיובי, 0 – אם מספר שווה ל-0, ו- 1- אם מספר שלילי.

קלט:

```
>> sign(a)
```

פלט:

```
ans =
    1
```

קלט:

```
>> b=-3
```

פלט:

```
b =
   -3
```

קלט:

```
>> sign(b)
```

פלט:

```
ans =
   -1
```

פונקציה factor שמפרקת מספר מורכב למספרים ראשונים:

קלט:

```
>>  factor(120)
```

פלט:

```
ans =
    2     2     2     3     5
```

קלט:

```
>> factor(131)
```

פלט:

```
ans =
  131
```

פונקציה isprime שבודקת האם מספר ראשוני או לא:

קלט:

```
>> isprime(120)
```

פלט:

```
ans =
  logical     0
```

קלט:

```
>> isprime(13)
```

פלט:

```
ans =
 logical   1
```

שארית של חילוק:

קלט:

```
>> mod(7,2)
```

פלט:

```
ans =

1
```

ח. פונקציות טריגונומטריות:

exp	פונקציה אקספוננציאלית	**abs**	ערך מוחלט
log	לוגריתם בבסיס e	**asin**	ארק סינוס
log10	לוגריתם של בסיס 10	**acos**	ארק קוסינוס
sqrt	שורש ריבועי	**atan**	ארק טנגנס
sin	סינוס	**sinh**	סינוס היפרבולי
cos	קוסינוס	**cosh**	קוסינוס היפרבולי
tan	טנגנס	**tanh**	טנגנס היפרבולי
cot	קוטנגנס	**asinh**	ארק סינוס היפרבולי
cec	סקנס	**acosh**	ארק קוסינוס היפרבולי
csc	קוסקנס	**atanh**	ארק טנגנס היפרבולי

שימו לב שהזוויות הן ברדיאן.

קלט:

```
>> acos(0.5)
```

פלט:

```
ans =
   1.0472
```

קלט:

```
>> sin(0.5)
```

פלט:

```
ans =
   0.4794
```

קלט:

```
>> exp(2)
```

פלט:

```
ans =
   7.3891
```

ט. הגדרת פונקציות:

קלט:

```
>> f=inline('1/x^2','x')
```

פלט:

```
f =
    Inline function:
    f(x) = 1/x^2
```

קלט:

```
>> f(2)
```

פלט:

```
ans =
   0.2500
```

קלט:

```
>> f(0.5)
```

פלט:

```
ans =
    4
```

י. פונקציה לחיפוש תאריך:

קלט:

```
>> calendar(2020,3)
```

פלט:

```
         Mar 2020
   S    M   Tu    W   Th    F    S
   1    2    3    4    5    6    7
   8    9   10   11   12   13   14
  15   16   17   18   19   20   21
  22   23   24   25   26   27   28
  29   30   31    0    0    0    0
```

יא. שימוש בתפריט עזרה

פקודה: help matlab\general << פותחת את תפריט העזרה לפקודות בסיסיות של המערכת.

```
>> help matlab\general
  General purpose commands.
  MATLAB Version 9.3 (R2017b) 24-Jul-2017

  General information.
    syntax         - Help on MATLAB command syntax.
    demo           - Run demonstrations.
    ver            - MATLAB, Simulink and toolbox version information.
    version        - MATLAB version information.
    verLessThan    - Compare version of toolbox to specified version string.
    logo           - Plot the L-shaped membrane logo with MATLAB lighting.
    membrane       - Generates the MATLAB logo.
    bench          - MATLAB Benchmark.

  Managing the workspace.
    who            - List current variables.
    whos           - List current variables, long form.
    clear          - Clear variables and functions from memory.
    onCleanup      - Specify cleanup work to be done on function completion.
    pack           - Consolidate workspace memory.
    load           - Load workspace variables from disk.
    save           - Save workspace variables to disk.
    saveas         - Save Figure or model to desired output format.
    memory         - Help for memory limitations.
    recycle        - Set option to move deleted files to recycle folder.
    quit           - Quit MATLAB session.
    exit           - Exit from MATLAB.

  Managing commands and functions.
    what           - List MATLAB-specific files in directory.
    type           - Display MATLAB program file.
    open           - Open files by extension.
    which          - Locate functions and files.
    pcode          - Create pre-parsed pseudo-code file (P-file).
    mex            - Compile MEX-function.
    inmem          - List functions in memory.
    namelengthmax - Maximum length of MATLAB function or variable name.
```

פקודה help matlab\ops << נותנת הסבר על אופרטורים בסיסיים שיש בתוכנה.

```
>> help matlab\ops
  Operators and special characters.

  Arithmetic operators.
    plus        - Plus                                   +
    uplus       - Unary plus                             +
    minus       - Minus                                  -
    uminus      - Unary minus                            -
    mtimes      - Matrix multiply                        *
    times       - Array multiply                        .*
    mpower      - Matrix power                           ^
    power       - Array power                           .^
    mldivide    - Backslash or left matrix divide        \
    mrdivide    - Slash or right matrix divide           /
    ldivide     - Left array divide                     .\
    rdivide     - Right array divide                    ./
    idivide     - Integer division with rounding option.
    kron        - Kronecker tensor product

  Relational operators.
    eq          - Equal                                 ==
    ne          - Not equal                             ~=
    lt          - Less than                              <
    gt          - Greater than                           >
    le          - Less than or equal                    <=
    ge          - Greater than or equal                 >=

  Logical operators.
    relop       - Short-circuit logical AND             &&
    relop       - Short-circuit logical OR              ||
    and         - Element-wise logical AND               &
    or          - Element-wise logical OR                |
    not         - Logical NOT                            ~
    punct       - Ignore function argument or output ~
    xor         - Logical EXCLUSIVE OR
    any         - True if any element of vector is nonzero
    all         - True if all elements of vector are nonzero

  Special characters.
    colon       - Colon                                  :
    paren       - Parentheses and subscripting          ( )
    paren       - Brackets                              [ ]
    paren       - Braces and subscripting               { }
```

פקודה help matlab\lang << מציגה פעולות של שפת תכנות שניתן להשתמש בהן.

```
>> help matlab\lang
  Programming language constructs.

  Control flow.
    if              - Conditionally execute statements.
    else            - Execute statement if previous IF condition failed.
    elseif          - Execute if previous IF failed and condition is true.
    end             - Terminate scope of control statements.
    for             - Repeat statements a specific number of times.
    parfor          - Parallel FOR-loop.
    while           - Repeat statements an indefinite number of times.
    break           - Terminate execution of WHILE or FOR loop.
    continue        - Pass control to the next iteration of a loop.
    switch          - Switch among several cases based on expression.
    case            - SWITCH statement case.
    otherwise       - Default SWITCH statement case.
    try             - Begin TRY block.
    catch           - Begin CATCH block.
    return          - Return to invoking function.
    error           - Display message and abort function.
    MException      - Constructs MATLAB exception object.
    assert          - Generate an error when a condition is violated.
    rethrow         - Reissue error.

  Evaluation and execution.
    eval            - Execute string with MATLAB expression.
    evalc           - Evaluate MATLAB expression with capture.
    feval           - Execute the specified function.
    evalin          - Evaluate expression in workspace.
    builtin         - Execute built-in function from overloaded method.
    assignin        - Assign variable in workspace.
    run             - Run script.

  Scripts, functions, classes, and variables.
    script          - About MATLAB script files.
    function        - Add new function.
    global          - Define global variable.
    persistent      - Define persistent variable.
    mfilename       - Name of currently executing MATLAB code file.
    lists           - Comma separated lists.
    exist           - Check if variables or functions are defined.
    mlock           - Prevents clearing function from memory.
    munlock         - Allow clearing function from memory.
```

פקודה help matlab\elmat << מציגה עזרה בפונקציות מתמטיות בסיסיות בהקשר מטריצות ופעולות בינהן.

```
>> help matlab\elmat
  Elementary matrices and matrix manipulation.

  Elementary matrices.
    zeros       - Zeros array.
    ones        - Ones array.
    eye         - Identity matrix.
    repmat      - Replicate and tile array.
    repelem     - Replicate elements of an array.
    linspace    - Linearly spaced vector.
    logspace    - Logarithmically spaced vector.
    freqspace   - Frequency spacing for frequency response.
    meshgrid    - X and Y arrays for 3-D plots.
    accumarray  - Construct an array with accumulation.
    :           - Regularly spaced vector and index into matrix.

  Basic array information.
    size        - Size of array.
    length      - Length of vector.
    ndims       - Number of dimensions.
    numel       - Number of elements.
    disp        - Display matrix or text.
    isempty     - True for empty array.
    isequal     - True if arrays are numerically equal.
    isequaln    - True if arrays are numerically equal, treating NaNs as equal.

  Matrix manipulation.
    cat         - Concatenate arrays.
    reshape     - Reshape array.
    diag        - Diagonal matrices and diagonals of matrix.
    blkdiag     - Block diagonal concatenation.
    tril        - Extract lower triangular part.
    triu        - Extract upper triangular part.
    fliplr      - Flip matrix in left/right direction.
    flipud      - Flip matrix in up/down direction.
    flip        - Flip the order of elements.
    rot90       - Rotate matrix 90 degrees.
    :           - Regularly spaced vector and index into matrix.
    find        - Find indices of nonzero elements.
    end         - Last index.
    sub2ind     - Linear index from multiple subscripts.
    ind2sub     - Multiple subscripts from linear index.
    bsxfun      - Binary singleton expansion function.
```

פקודה help matlab\elfun<< מציגה עזרה עבור פונקציות מתמטיות בסיסיות.

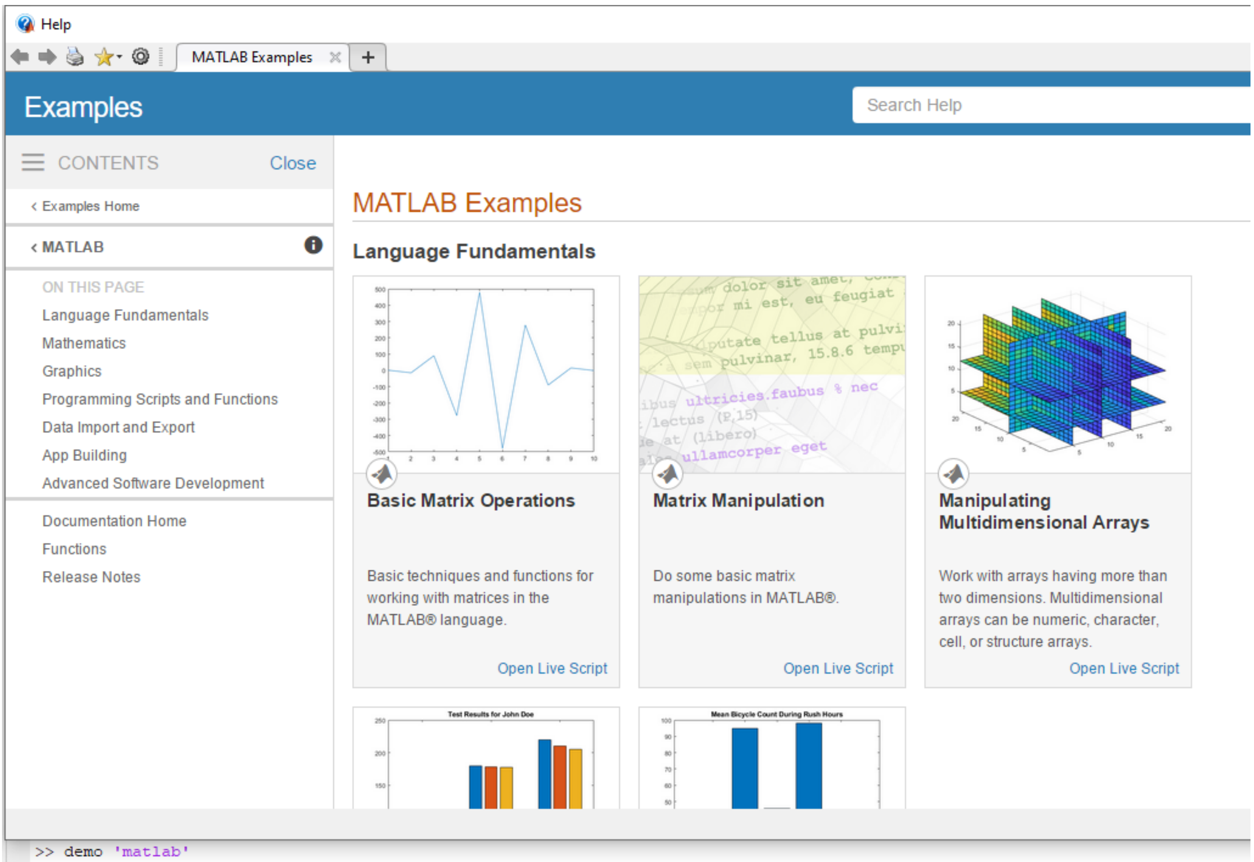
```
>> help matlab\elfun
  Elementary math functions.

  Trigonometric.
    sin         - Sine.
    sind        - Sine of argument in degrees.
    sinh        - Hyperbolic sine.
    asin        - Inverse sine.
    asind       - Inverse sine, result in degrees.
    asinh       - Inverse hyperbolic sine.
    cos         - Cosine.
    cosd        - Cosine of argument in degrees.
    cosh        - Hyperbolic cosine.
    acos        - Inverse cosine.
    acosd       - Inverse cosine, result in degrees.
    acosh       - Inverse hyperbolic cosine.
    tan         - Tangent.
    tand        - Tangent of argument in degrees.
    tanh        - Hyperbolic tangent.
    atan        - Inverse tangent.
    atand       - Inverse tangent, result in degrees.
    atan2       - Four quadrant inverse tangent.
    atan2d      - Four quadrant inverse tangent, result in degrees.
    atanh       - Inverse hyperbolic tangent.
    sec         - Secant.
    secd        - Secant of argument in degrees.
    sech        - Hyperbolic secant.
    asec        - Inverse secant.
    asecd       - Inverse secant, result in degrees.
    asech       - Inverse hyperbolic secant.
    csc         - Cosecant.
    cscd        - Cosecant of argument in degrees.
    csch        - Hyperbolic cosecant.
    acsc        - Inverse cosecant.
    acscd       - Inverse cosecant, result in degrees.
    acsch       - Inverse hyperbolic cosecant.
    cot         - Cotangent.
    cotd        - Cotangent of argument in degrees.
    coth        - Hyperbolic cotangent.
    acot        - Inverse cotangent.
    acotd       - Inverse cotangent, result in degrees.
    acoth       - Inverse hyperbolic cotangent.
```

<u>כדי לחפש כל מה שקשור למושג מסוים יש לכתוב:</u>

```
>> lookfor boolean
conboolean                         - Boolean combination of constrai
getAndResetFilesAddedOrRemovedFlag - getAndResetFilesAddedOrRemo
getFilesAddedOrRemovedFlag         - returns a boolean value which
boolean                            - Convert numeric values to logic
```

בכדי לקבל עזרה מלאה בכל הפונקציות של התוכנה יש לרשום: >> demo 'matlab' ואז נקבל את החלון הבא:



במידה ורוצים לדעת על פקודה מסוימת:

קלט:

```
>> help clc
```

פלט:

```
clc    Clear command window.
   clc clears the command window and homes the cursor.
    See also home.
   Reference page for clc
```

במידה ורשמתם פקודה לא נכונה MATLAB ירשום עזרה לפקודה הקרובה ביותר למה שרשמתם:

קלט:

```
>> help calc
```

פלט:

```
--- help for clc ---
 clc    Clear command window.
    clc clears the command window and homes the cursor.
     See also home.
    Reference page for clc
```

פתרון תרגיל 1.1

קלט:

```
>> a=2;
b=3;
c=1;
>> d=b*b-4*a*c
```

פלט:

```
d =
     1
```

קלט:

```
>> x1=(-b+sqrt(d))/(2*a)
```

פלט:

```
x1 =
   -0.5000
```

קלט:

```
>> x2=(-b-sqrt(d))/(2*a)
```

פלט:

```
x2 =
    -1
```

קלט:

```
>> b=1;
>> d=b*b-4*a*c
```

פלט:

```
d =
    -7
```

קלט:

```
>> x1=(-b+sqrt(d))/(2*a)
```

פלט:

```
x1 =
  -0.2500 + 0.6614i
```

קלט:

```
>> x2=(-b-sqrt(d))/(2*a)
```

פלט:

```
x2 =
  -0.2500 - 0.6614i
```

פרק 2. ייצוג ווקטורים, מטריצות ופעולות מתמטיות ביניהם

א. הצגת שיטות שונות לאתחול ווקטורים ומטריצות

הגדרת ווקטור אופקי.

ווקטור – הינו מערך חד ממדי עם מספר איברים. כדי להגדיר ווקטור ניקח את האיברים הנמצאים בסוגריים מרובעות:

קלט:

```
>> v=[1 2 3 4 5]
```

פלט:

```
v =
5    4    3    2    1
```

כדי לדעת מספר איברים בווקטור ניתן להשתמש בפקודה length

קלט:

```
>> length(v)
```

פלט:

```
ans =
5
```

ניתן לבצע פעולות בין ווקטורים.

קלט:

```
>> a = [9 8 8 6 5]
```

פלט:

```
a =
   9    8    8    6    5
```

קלט:

```
>> c=a-v
```

פלט:

```
c =
   8    6    5    2    0
```

קלט:

```
>> d=a+v
```

פלט:

```
d =
   10   10   11   10   10
```

קלט:

```
>> e=a*v
```

פלט:

```
Error using  *
Inner matrix dimensions must agree.
```

קלט:

```
>> c=a*3
```

פלט:

```
c =
15    18    24    24    27
```

קלט:

```
>> f=a*3-v
```

פלט:

```
f =
   26    22    21    14    10
```

ניתן להגדיר ווקטור בצורה נוספת:

במידה ויש 2 ערכים וביניהם נקודתיים, הווקטור שווה לכל הערכים מהראשון עד האחרון בהפרש של 1. זה יעבוד רק לכיוון פלוס אינסוף.

קלט:

```
>> d=[1:5]
```

פלט:

```
d =
     1     2     3     4     5
```

קלט:

```
>> e=[1.2:6.5]
```

פלט:

```
e =
    1.2000    2.2000    3.2000    4.2000    5.2000    6.2000
```

קלט:

```
>> e=[5:1]
```

פלט:

```
e =
  1×0 empty double row vector
```

קלט:

```
>> d=[-10:-5]
```

פלט:

```
d =
   -10    -9    -8    -7    -6    -5
```

קלט:

```
>> d=[-4:4]
```

פלט:

```
d =
    -4    -3    -2    -1     0     1     2     3     4
```

במידה ובווקטור יש 3 ערכים וביניהם נקודתיים, הווקטור שווה לכל הערכים מהראשון עד לאחרון עם צעד שמוגדר בערך השני.
כאן יש אפשרות לעבוד גם לכיוון פלוס אין סוף וגם לכיוון מינוס אין סוף.

קלט:

```
>> v=[3:0.2:4]
```

פלט:

```
v =
   3.0000    3.2000    3.4000    3.6000    3.8000    4.0000
```

קלט:

```
>> v=[3:-0.2:2]
```

פלט:

```
v =
   3.0000    2.8000    2.6000    2.4000    2.2000    2.0000
```

קלט:

```
>> v=[1.1:0.2:1.6]
```

פלט:

```
v =
   1.1000    1.3000    1.5000
```

קלט:

```
>> v=[1:1:-1]
```

פלט:

```
v =
  1×0 empty double row vector
```

קלט:

```
>> v=[-1:2:-3]
```

פלט:

```
v =
  1×0 empty double row vector
```

קלט:

```
>> v=[1:1:1]
```

פלט:

```
v =
    1
```

ניתן לאחד מספר סדרות באותו ווקטור:

קלט:

```
>> v=[1:2:5, -1, -2, -5]
```

פלט:

```
v =
   1     3     5    -1    -2    -5
```

קלט:

```
>> v=[1,2,3,4:2:9]
```

פלט:

```
v =
   1    2    3    4    6    8
```

ניתן לקבל חלק מאיברים של הווקטור שהגדרנו קודם:

קלט:

```
>> v(4:6)
```

פלט:

```
ans =
    4    6    8
```

זה לא משנה את הווקטור עצמו, אלא רק מדפיס ערכים ממקומות 4 עד 6 – אם נדפיס את ווקטור V כולו – נקבל את הערכים התחלתיים שהגדרנו:

קלט:

```
>> v
```

פלט:

```
v =
    1    2    3    4    6    8
```

ניתן להציג איברים לא ברצף, אלא עם צעד מסוים:

קלט:

```
>> v(2:2:6)
```

פלט:

```
ans =
    2    4    8
```

קלט:

```
>> v(6:-2:2)
```

פלט:

```
ans =
    8    4    2
```

כדי להגדיר פונקציה של ווקטור נקודות משתמשים באופציה הבאה:

קלט:

```
>>  f=inline(vectorize('1/x^2'),'x')
```

פלט:

```
f =
```

```
    Inline function:
    f(x) = 1./x.^2
```

קלט:

```
>> f(1:0.1:2)
```

פלט:

```
ans =
  Columns 1 through 6
    1.0000    0.8264    0.6944    0.5917    0.5102    0.4444
  Columns 7 through 11
    0.3906    0.3460    0.3086    0.2770    0.2500
```

ניתן להגדיר פונקציה לשני משתנים:

קלט:

```
>> g=inline(vectorize('1/(x^2+y^2)'),'x','y')
```

פלט:

```
g =
    Inline function:
    g(x,y) = 1./(x.^2+y.^2)
```

קלט:

```
>> g([1 2],[3 4])
```

פלט:

```
ans =
    0.1000    0.0500
```

הגדרת ווקטור אנכי.

קלט:

```
>> v=[-1;2;-3]
```

פלט:

```
v =
    -1
     2
    -3
```

קלט:

```
>> v=[3
4
5
6]
```

פלט:

```
v =
     3
     4
```

```
    5
    6
```

הגדרת מטריצות:

קלט:

```
>> c=[2 3 4; 5 6 7]
```

פלט:

```
c =
    2    3    4
    5    6    7
```

קלט:

```
>> c=[2 3 4; 5 6 ]
```

פלט:

```
Dimensions of matrices being concatenated are not consistent.
```

קלט:

```
>> c=[1:2:8; 2:3:11]
```

פלט:

```
c =
    1    3    5    7
    2    5    8   11
```

ב. מטריצות מיוחדות ones ,zeros, eye, rand והדגמת השימוש בהן

ones	zeros	eye	rand
קלט: >> ones(3) פלט: ans = 1 1 1 1 1 1 1 1 1 קלט: >> ones(3,2) פלט: ans = 1 1 1 1 1 1 קלט: >> ones פלט: ans = 1	קלט: >> a=zeros פלט: a = 0 קלט: >> a=zeros(3) פלט: a = 0 0 0 0 0 0 0 0 0 קלט: >> a=zeros(2,3) פלט: a = 0 0 0 0 0 0	קלט: >> x=eye(3) פלט: x = 1 0 0 0 1 0 0 0 1 קלט: >> x=eye פלט: x = 1 קלט: >> x=eye(3,4) פלט: x = 1 0 0 0 0 1 0 0 0 0 1 0	קלט: >> R=rand(4) פלט: R = 0.8147 0.6324 0.9575 0.9572 0.9058 0.0975 0.9649 0.4854 0.1270 0.2785 0.1576 0.8003 0.9134 0.5469 0.9706 0.1419 קלט: >> R=rand פלט: R = 0.4218 קלט: >> R=rand(4)*10 פלט: R = 9.1574 0.3571 7.5774 1.7119 7.9221 8.4913 7.4313 7.0605 9.5949 9.3399 3.9223 0.3183 6.5574 6.7874 6.5548 2.7692 \

תרגיל 1.2

תכנן מטריצה 3 על 4 עם ערכים אקראיים שלמים מ-10 עד 30.

קלט:

```
>> R=round(rand(3,4)*20+10)
```

פלט:

```
R =
   11   24   11   25
   12   16   19   26
   26   29   18   14
```

ג. פעולות אריתמטיות : חיבור, חיסור, כפל וחלוקת מטריצות ווקטורים

חיבור מטריצות או ווקטורים אפשר לעשות אך ורק בתנאי שמטריצות או ווקטורים יהיו זהים מבחינת הממדים.

קלט:
```
>> a=[1:3; 2:4; 3:5]
```
פלט:
```
a =
    1    2    3
    2    3    4
    3    4    5
```
קלט:
```
>> b=[4:6;5:7;2:4]
```
פלט:
```
b =
    4    5    6
    5    6    7
    2    3    4
```
קלט:
```
>> a*b
```
פלט:
```
ans =
   20   26   32
   31   40   49
   42   54   66
```
קלט:
```
>> 2*a
```
פלט:
```
ans =
    2    4    6
    4    6    8
    6    8   10
```
קלט:
```
>> a/2
```
פלט:
```
ans =
   0.5000    1.0000    1.5000
   1.0000    1.5000    2.0000
   1.5000    2.0000    2.5000
```

קלט:
```
>> a=[1:3; 2:4]
```
פלט:
```
a =
    1    2    3
    2    3    4
```
קלט:
```
>> b=[4:6;5:7]
```
פלט:
```
b =
    4    5    6
    5    6    7
```
קלט:
```
>> a+b
```
פלט:
```
ans =
    5    7    9
    7    9   11
```
קלט:
```
>> a-b
```
פלט:
```
ans =
   -3   -3   -3
   -3   -3   -3
```

ד. פעולות מיוחדות על מטריצות כגון: אלכסון, היפוך וחשוב דטרמיננטה

כדי להפוך את מטריצה או ווקטור (Transpose)

קלט:

```
>> R=round(rand(3,4)*20+10)
```

פלט:

```
R =
   11   24   11   25
   12   16   19   26
   26   29   18   14
```

קלט:

```
>> R'
```

פלט:

```
ans =
   11   12   26
   24   16   29
   11   19   18
   25   26   14
```

קלט:

```
>> v=[1:3:9]
```

פלט:

```
v =
    1    4    7
```

קלט:

```
>> v'
```

פלט:

```
ans =
    1
    4
    7
```

במידה ומטריצה מכילה מספרים מרוכבים היא עושה היפוך וגם ממספר היא עושה את הצמוד שלו:

קלט:

```
>> v=[1+2i, 2-3i; 2-4i, 9+6i]
```

פלט:

```
v =
   1.0000 + 2.0000i   2.0000 - 3.0000i
   2.0000 - 4.0000i   9.0000 + 6.0000i
```

קלט:

```
>> v'
```

פלט:

```
ans =
```

```
   1.0000 - 2.0000i   2.0000 + 4.0000i
   2.0000 + 3.0000i   9.0000 - 6.0000i
```

קלט:

```
>> v=[1+2i, 2-3i, 5-2i; 2-4i, 9+6i, 7+2i]
```

פלט:

```
v =
   1.0000 + 2.0000i   2.0000 - 3.0000i   5.0000 - 2.0000i
   2.0000 - 4.0000i   9.0000 + 6.0000i   7.0000 + 2.0000i
```

קלט:

```
>> v'
```

פלט:

```
ans =
   1.0000 - 2.0000i   2.0000 + 4.0000i
   2.0000 + 3.0000i   9.0000 - 6.0000i
   5.0000 + 2.0000i   7.0000 - 2.0000i
```

ה. ריכוז פעולות עם מטריצות:

פקודה	הסבר (מה הפונקציה מחזירה)
A+B	חיבור מטריצות A ו- B בתנאי ששניהם במידות שוות.
A–B	חיסור מטריצות A ו- B בתנאי ששניהם במידות שוות.
A*B	כפל מטריצות, כאשר $(A_{n\times m} \cdot B_{m\times k}=C_{n\times k})$
A.*B	כפל של איברים של מטריצה ראשונה על איברים באותם מקומות של מטריצה שנייה – A ו- B חייבים להיות באותם מידות.
A./B	חילוק של איברים של מטריצה ראשונה על איברים באותם מקומות של מטריצה שנייה – A ו- B חייבים להיות באותם מידות.
A.^k	עליית בחזרה k של איברים של מטריצה.
A’	מטריצה A^t
A^–1 inv(A)	חישוב של מטריצה הפוכה (A_{-1})
det(A)	חישוב דטרמיננטה של מטריצה ריבועית (|A|)
size(A)	חישוב של גודל (ממד) של מטריצה A
trace(A)	סכום איברים באלכסון ראשי.
sum(A)	סכום איברים בכל עמודה
prod(A)	תוצאת הכפל של מספרים בכל עמודה
diag(A)	פונקציה מייצרת ווקטור ומעתיקה עליו את כל האיברים של אלכסון ראשי של מטריצה A
sort(A)	מיון של כל העמודות של מטריצה בסדר עולה
max(A)	מציאת ערך מקסימלי בכל עמודה של מטריצה A
min(A)	מציאת ערך מינימלי בכל עמודה של מטריצה A
mean(A)	מציאת ערך ממוצע בכל עמודה של מטריצה A
rot90(A)	סיבוב מטריצה ב $90°$ שמאלה
fliplr(A)	היפוך מטריצה משמאל לימין
flipud(A)	היפוך מטריצה מלמלא למטה

דוגמאות:

נגדיר שתי מטריצות ונבצע איתם פעולות מהטבלה:

קלט:

```
>> A=[1 3 2 6; 2 1 5 3; 9 7 4 6; 2 4 1 6]
```

פלט:

```
A =
    1    3    2    6
    2    1    5    3
    9    7    4    6
    2    4    1    6
```

קלט:

```
>> B=[11 3 2 6; 2 21 5 13; 9 27 4 6; 2 4 21 6]
```

פלט:

```
B =
   11    3    2    6
    2   21    5   13
    9   27    4    6
    2    4   21    6
```

קלט:

```
>> C=A-B
```

פלט:

```
C =
  -10    0    0    0
    0  -20    0  -10
    0  -20    0    0
    0    0  -20    0
```

קלט:

```
>> C=A+B
```

פלט:

```
C =
   12    6    4   12
    4   22   10   16
   18   34    8   12
    4    8   22   12
```

קלט:

```
>> C=A.*B
```

פלט:

```
C =
   11    9    4   36
    4   21   25   39
   81  189   16   36
    4   16   21   36
```

קלט:

```
>> C=A*B
```

פלט:

```
C =
   47  144  151   93
   75  174   92   73
  161  306  195  205
   51  141  154  106
```

קלט:

```
>> C=C.^2
```

פלט:

```
C =
   0.0083    1.0000    1.0000    1.0000
   1.0000    0.0023    1.0000    0.0533
   1.0000    0.0672    1.0000    1.0000
   1.0000    1.0000    0.0023    1.0000
```

קלט:

```
>> C=A./B
```

פלט:

```
C =
   0.0909    1.0000    1.0000    1.0000
   1.0000    0.0476    1.0000    0.2308
   1.0000    0.2593    1.0000    1.0000
   1.0000    1.0000    0.0476    1.0000
```

קלט:

```
>> A*inv(A)
```

פלט:

```
ans =
   1.0000         0    0.0000   -0.0000
  -0.0000    1.0000    0.0000   -0.0000
  -0.0000    0.0000    1.0000   -0.0000
  -0.0000         0         0    1.0000
```

קלט:

```
>> inv(A)
```

פלט:

```
ans =
    8.2500   -3.0000    1.7500   -8.5000
  -14.2500    5.0000   -2.7500   14.5000
   -5.0000    2.0000   -1.0000    5.0000
    7.5833   -2.6667    1.4167   -7.5000
```

קלט:

```
>> size(A)
```

פלט:

```
ans =
    4    4
```

קלט:

```
>> det(A)
```

פלט:

```
ans =
  -12.0000
```

קלט:

```
>> trace(A)
```

פלט:

```
ans =
   12
```

קלט:

```
>> sum(A)
```

פלט:

```
ans =
   14   15   12   21
```

קלט:

```
>> A
```

פלט:

```
A =
    1    3    2    6
    2    1    5    3
    9    7    4    6
    2    4    1    6
```

קלט:

```
>> prod(A)
```

פלט:

```
ans =
   36   84   40   648
```

קלט:

```
>> mean(A)
```

פלט:

```
ans =
   3.5000   3.7500   3.0000   5.2500
```

קלט:

```
>> sort(A)
```

פלט:

```
ans =
    1    1    1    3
    2    3    2    6
    2    4    4    6
    9    7    5    6
```

קלט:

```
>> diag(A)
```

פלט:

```
ans =
    1
    1
    4
    6
```

קלט:

```
>> max(A)
```

פלט:

```
ans =
    9    7    5    6
```

קלט:

```
>> min(A)
```

פלט:

```
ans =
    1    1    1    3
```

קלט:

```
>> rot90(A)
```

פלט:

```
ans =
    6    3    6    6
    2    5    4    1
    3    1    7    4
    1    2    9    2
```

קלט:

```
>> fliplr(A)
```

פלט:

```
ans =
    6    2    3    1
    3    5    1    2
    6    4    7    9
    6    1    4    2
```

קלט:

```
>> flipud(A)
```

פלט:

```
ans =
    2    4    1    6
    9    7    4    6
    2    1    5    3
    1    3    2    6
```

תרגול: נתונות מטריצות D,C,B,A.

$$A=\begin{pmatrix}1 & 2 & -1\\ 2 & 3 & 1\\ -1 & -2 & 2\end{pmatrix},\ B=\begin{pmatrix}-3 & 1 & 2\end{pmatrix},\ C=\begin{pmatrix}2 & 4 & 3\\ -2 & 1 & 2\\ 4 & 5 & 1\end{pmatrix},\ D=\begin{pmatrix}2\\ -1\\ 3\end{pmatrix}.$$

הגדר אותן וחשב:

1. B^{T}.
2. $A+C^{-1}$.
3. $A^{-1}\cdot B^{\mathrm{T}}$.
4. $B\cdot D$.
5. $A\cdot C$.
6. $C\cdot A$.
7. $A\cdot A\cdot A$.
8. $2\cdot D+B^{\mathrm{T}}$.
9. $|A|+|C|$.
10. $(A^{\mathrm{T}}+C)^{2}$.
11. $A\cdot D\cdot B$.
12. $3\cdot A/C$.

פרק 3.: אלגברה בוליאנית בעזרת MATLAB.

א. פונקציות להמרה מבסיס אחד לשני:

קלט:

```
>> dec2bin(1101101)
```

פלט:

```
ans =
   '100001100110100101101'
```

קלט:

```
>> dec2bin(35)
```

פלט:

```
ans =
   '100011'
```

קלט:

```
>> bin2dec('100011')
```

פלט:

```
ans =
   35
```

קלט:

```
>> hex2dec('15EF')
```

פלט:

```
ans =
       5615
```

ב. פעולות לוגיות על סיביות.

ניתן לבצע פעולות רגילות OR, AND, XOR, NOT, הזזה על הסיביות, כדי להציג אותם נראה את הדוגמא הבאה:

קלט:

```
>> a=78; b=65;
>> dec2bin(a)
```

פלט:

```
ans = '1001110'
```

קלט:

```
>> dec2bin(b)
```

פלט:

```
ans =  '1000001'
```

קלט:

```
>> c=bitand(a,b)
```

פלט

```
c =   64
```

קלט:

```
>> dec2bin(c)
```

פלט:

```
ans =  '1000000'
```

קלט:

```
>> d=dec2bin(bitor(a,b))
```

פלט:

```
d =    '1001111'
```

קלט:

```
>> e=dec2bin(bitxor(a,b))
```

פלט:

```
e =    '1111'
```

קלט:

```
>> dec2bin(a)
```

פלט:

```
ans =    '1001110'
```

קלט:

```
>> dec2bin(bitshift(a,3))
```

פלט:

```
ans =    '1001110000'
```

קלט:

```
>> dec2bin(bitshift(a,-3))
```

פלט:

```
ans =    '1001'
```

קלט:

```
>> dec2bin(bitcmp(12131235234542))
```

פלט:

```
ans =
    '1111111111111111111101001110111011110100000100010000000000000000'
```

שימו לב שפעולות אלה עובדות רק על משתנים מסוג INT ולכן הוא מוסיף בהתחלה "1" מיותרים.

ג. הגדרת טבלת אמת.

כדי להגדיר טבלת אמת נגדיר ווקטורים לכל משתנה ולתוצאה של הפונקציה:

קלט:

```
>> a=[0 0 0 0 1 1 1 1];
b=[0 0 1 1 0 0 1 1];
c= [0 1 0 1 0 1 0 1];
```

דוגמא למימוש מסכם חלקי, עם 2 כניסות A, B ויציאות S ו- Cout.

טבלת אמת נראית כך:

A	B	Sum	Cout
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

נגדיר ווקטור לכל המצבים של A לפי סדר בטבלת אמת
נגדיר ווקטור לכל המצבים של B לפי סדר בטבלת אמת

נחשב סכום ונשא ל- 4 אפשרויות

נציג כותרות

קלט:

```
>> A = logical([0 0 1 1]); %define logical input A
B = logical([0 1 0 1]);%define logical input B
for k = 1:4
S(k) = ~A(k)&B(k)|(A(k)&~B(k)) ;
C(k) = A(k)&B(k) ;
end
disp('The truth table of half adder is:')
disp('A B S C')
table=[A; B; S; C];
fprintf ('%i %i %i %i\n', table)
```

פלט:

```
The truth table of half adder is:
A B S C
0 0 0 0
0 1 1 0
1 0 1 0
1 1 0 1
```

יש עוד 2 אפשרויות לכתוב את אותה פונקציה:

קלט:

```
>> A = logical([0 0 1 1]); %define logical input A
B = logical([0 1 0 1]);%define logical input B
for k = 1:4
S(k) = xor(A(k),B(k)) ;
C(k) = and(A(k),B(k)) ;
end
disp('The truth table of half adder is:')
disp('A B S C')
table=[A; B; S; C];
fprintf ('%i %i %i %i\n', table);
```

קלט:

```
>> A = logical([0 0 1 1]); %define logical input A
B = logical([0 1 0 1]);%define logical input B
S = xor(A,B);
C= and(A,B) ;
disp('The truth table of half adder is:')
disp('A B S C')
table=[A; B; S; C];
fprintf ('%i %i %i %i\n', table);
```

פלט:	פלט:
The truth table of half adder is: A B S C 0 0 0 0 0 1 1 0 1 0 1 0 1 1 0 1	The truth table of half adder is: A B S C 0 0 0 0 0 1 1 0 1 0 1 0 1 1 0 1

נעשה מסכם מלא:

שימו לב שאי-אפשר לעשות פעולה xor(A,B,Cin) – וצריכים לפצל אותם.

קלט:

```
>> A = logical([0 0 0 0 1 1 1 1]); %define logical input A
B = logical([0 0 1 1 0 0 1 1]);%define logical input B
Cin = logical([0 1 0 1 0 1 0 1]);%define logical input C
S = xor(xor(A,B),Cin) ;
Cout=or(or(and(A,B),and(A,Cin)),and(B,Cin))
disp('The truth table of full adder is:')
disp('A B Cin S Cout');
table=[A; B; Cin; S; Cout];
fprintf ('%i %i %i %i %i\n', table);
```

פלט:

```
The truth table of full adder is:
A B Cin S Cout
0 0 0 0 0
0 0 1 1 0
0 1 0 1 0
0 1 1 0 1
1 0 0 1 0
1 0 1 0 1
1 1 0 0 1
1 1 1 1 1
```

פרק 4. ייצוגים גרפיים

א. הגדרה ידנית של גרפים.

פונקציות דו ממדיות y=f(x)

ניקח לדוגמא את הפונקציה y=sin3πx בקטע בין 0 ל- 1. אנו צריכים לבצע את הגרף למספר גדול של נקודות. נגדיר N – מספר נקודות , H – יהיה קושר הפרדה או 1/N ואז הווקטור שלנו נראה כך:

קלט:

```
>> N=10
```

פלט:

```
N =
   10
```

קלט:

```
>> H=1/N
```

פלט:

```
H =
   0.1000
```

קלט:

```
>> x=[0:H:1]
```

פלט:

```
x =
  Columns 1 through 6
      0    0.1000    0.2000    0.3000    0.4000    0.5000
  Columns 7 through 11
   0.6000    0.7000    0.8000    0.9000    1.0000
```

קלט:

```
>> y=sin(3*pi*x)
```

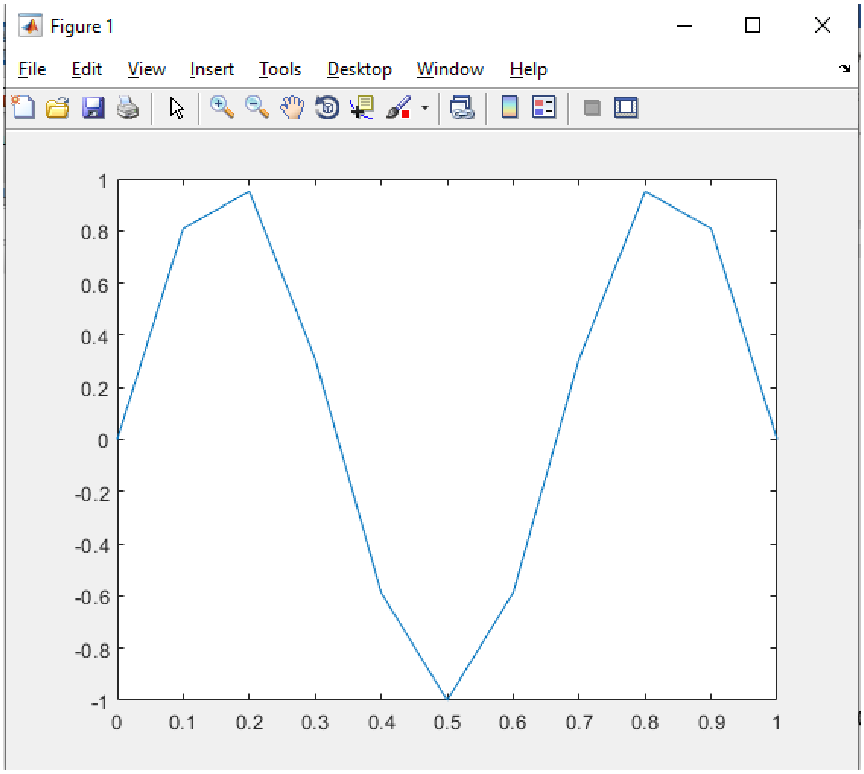
פלט:

```
y =
  Columns 1 through 6
      0    0.8090    0.9511    0.3090   -0.5878   -1.0000
  Columns 7 through 11
  -0.5878    0.3090    0.9511    0.8090    0.0000
```

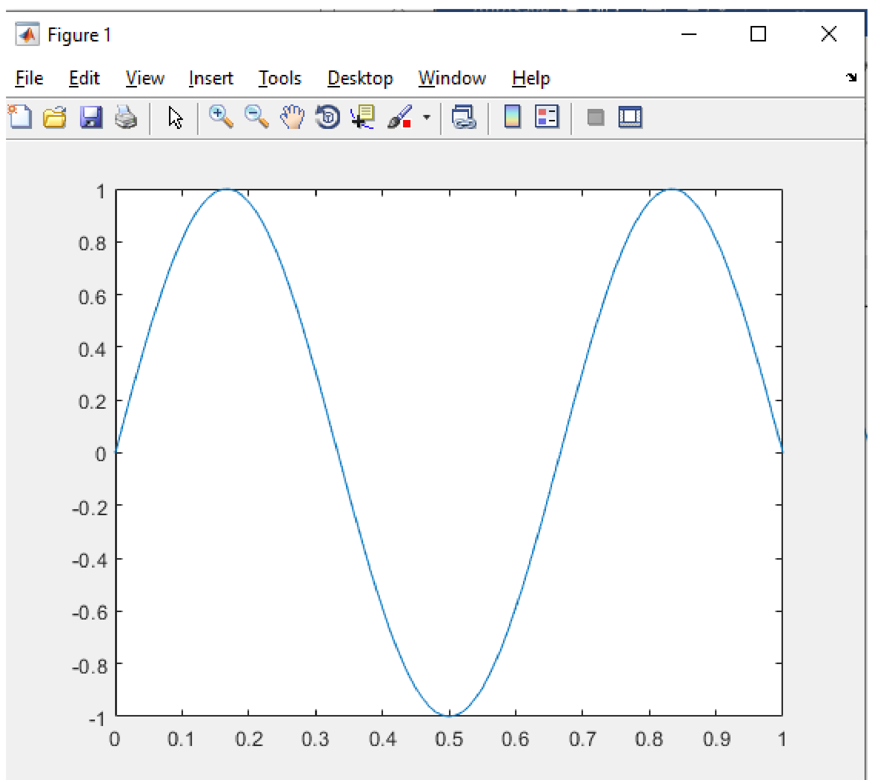
קלט:

```
>> plot(x,y)
```

פלט:



במידה ונגדיל את כמות נקודות ה-N ל-100 נקבל את התמונה הבאה:

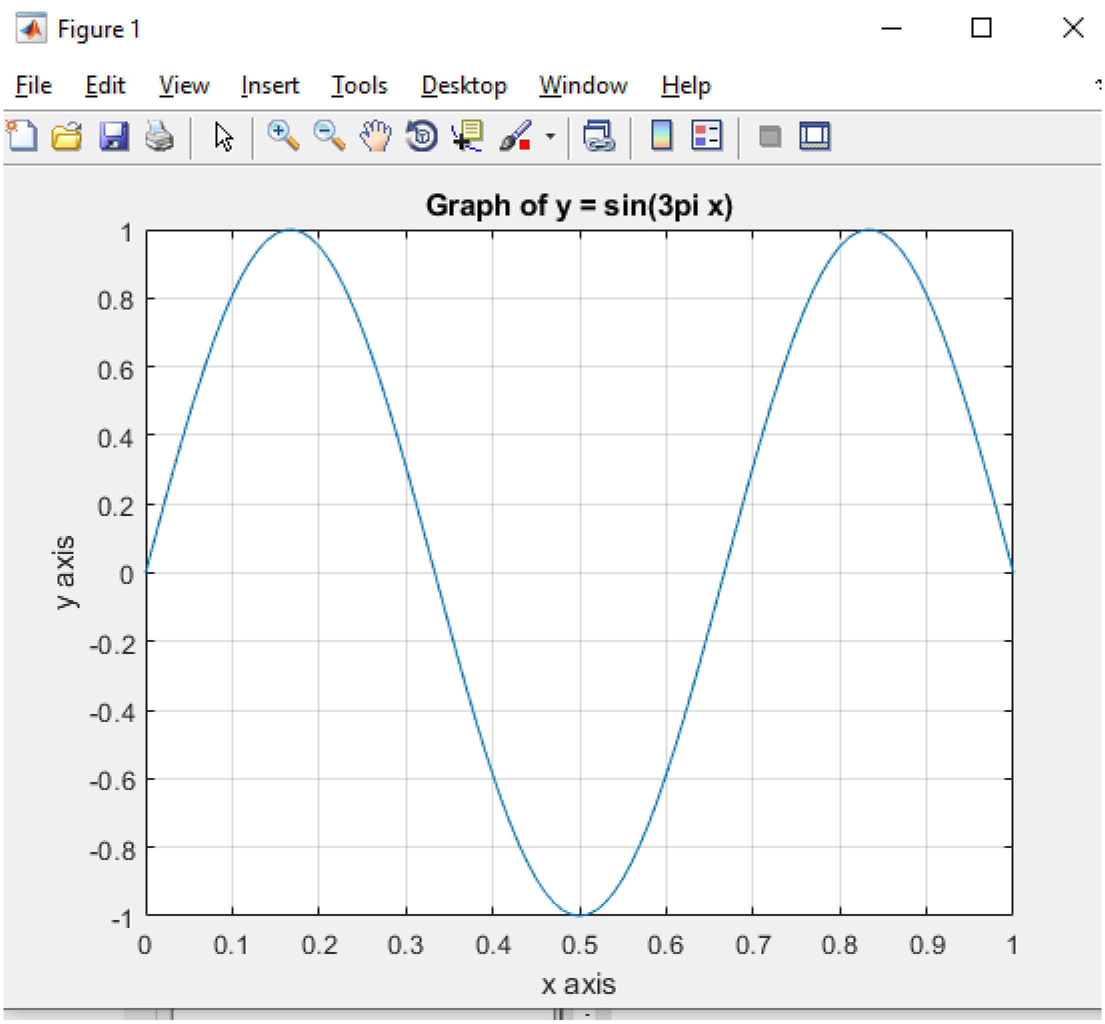


פקודה TITLE – נותנת שם לגרף, xlabel - שם של ציר X, ylabel – שם של ציר Y, פקודה grid – מוסיפה קווים לגרף,

קלט:

```
>> plot(x,y)
>> title('Graph of y = sin(3pi x)')
>> xlabel('x axis')
>> ylabel('y axis')
>> grid
```

פלט:

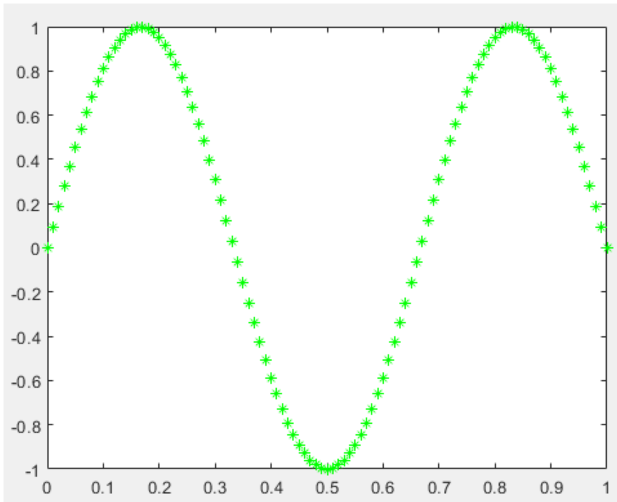


ישנם פרמטרים נוספים שמאפשרים לעצב את הגרף, לדוגמא:

קלט:

```
>> plot(x,y,'g*')
```

פלט:



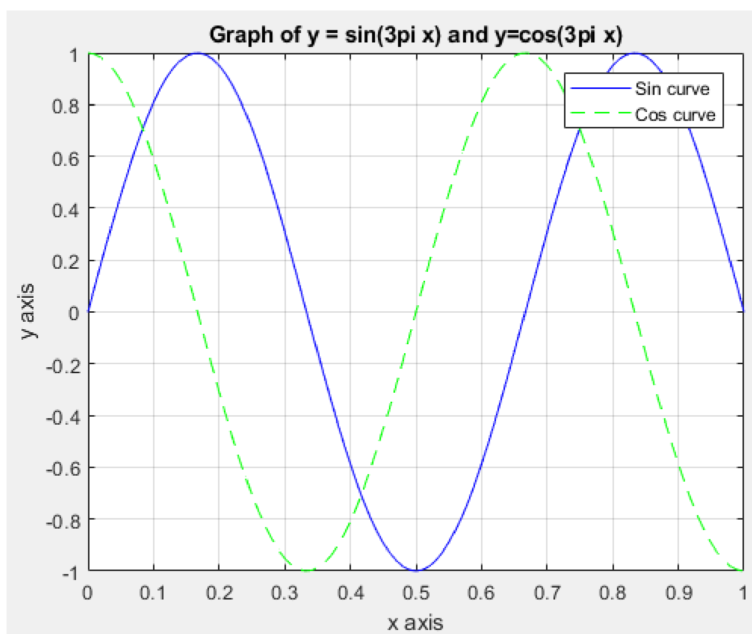
Colours		Line Styles	
y	yellow	.	point
m	magenta	o	circle
c	cyan	x	x-mark
r	red	+	plus
g	green	-	solid
b	blue	*	star
w	white	:	dotted
k	black	-.	dashdot
		--	dashed

למקרה ורוצים לייצג 2 פונקציות או יותר באותו גרף נבצע את הפעולות הבאות:

קלט:

```
>> plot(x,y,'b-',x,cos(3*pi*x),'g--')
>> legend('Sin curve','Cos curve')
>> grid
>> xlabel('x axis')
>> ylabel('y axis')
>> title('Graph of y = sin(3pi x) and y=cos(3pi x)')
```

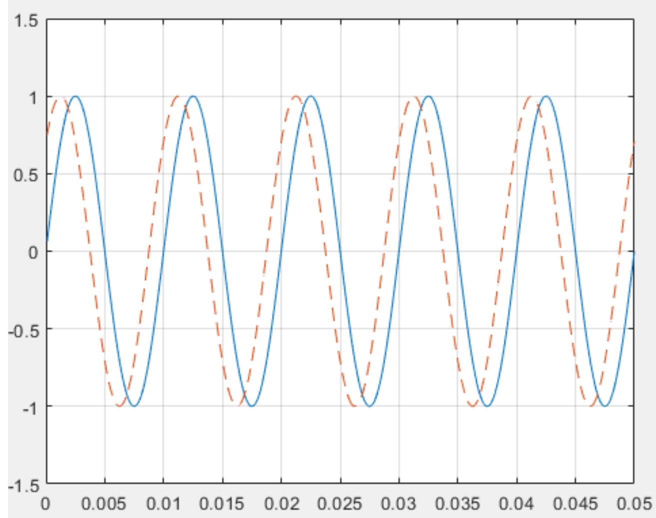
הגרף ייראה כך:



ניתן לעשות "העתק" "הדבק" של כל הפקודות ביחד לתוך MATLAB ובלחיצת ENTER לראות את התוצאה:

קלט:

```
>> t = [1:1:500]/500/20;
f = 100;
y1 = sin(2*pi*f*t);
y2 = sin(2*pi*f*t+pi/4);
plot(t,y1,'-',t,y2,'--');
axis([0 0.05 -1.5 1.5]);
grid
```



ריכוז פקודות חשובות (ללא שימוש בחלון אוטומטי):

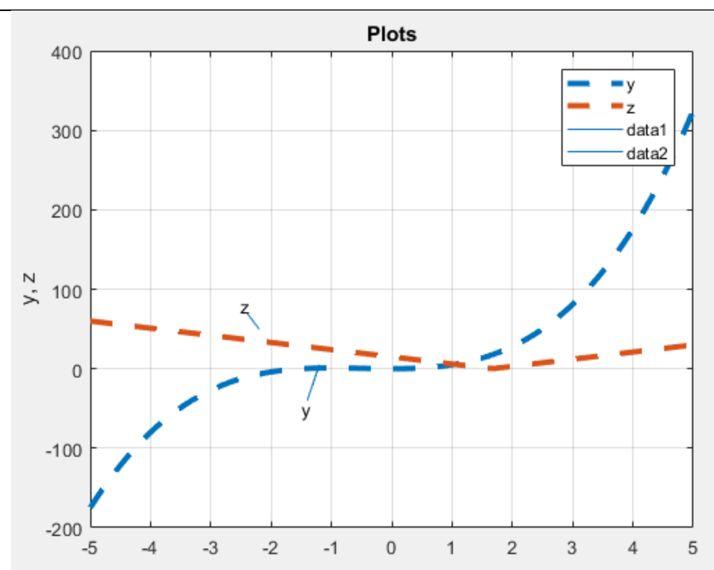
פקודה	הסבר
grid	הוספת רשת
xlabel(‘text’)	הוספת TEXT בציר X
ylabel(‘text’)	הוספת TEXT בציר Y
title(‘text’)	הוספת שם לגרף
legend(‘text’)	הוספת תיאור לגרף
text(X,Y,’text’)	הוספת TEXT בנקודה ספציפית XY
line([x1 x2],[y1 y2])	העברת הקו מנקודה X1Y1 לנקודה X2Y2
axis([xmin xmax ymin ymax])	הגדרת צירים X ו- Y
set(graph,’name_parametr1’,par1,’name_par2’,par2....)	הגדרת תכונות של גרף

דוגמא:

נבנה גרף לפונקציות הבאות: $y = 2x^3 + 3x^2, z = 3 \cdot |3x - 5|$ בקטע [-5;5] עם צעד 0.1

קלט:

```
>> x=-5:0.1:5;
y=2*x.^3+3*x.^2;
z=3*abs(3*x-5);
g=plot(x,y,x,z)
set(g,'linewidth',3,'linestyle','--')
title('Plots')
xlabel('x')
ylabel('y, z')
grid
legend('y','z')
text(-2.5,80,'z')
line([-2.4 -2.2],[70 50])
text(-1.5,-50,'y')
line([-1.4 -1.2],[-40 5])
```



תרגול:

כתוב קטע קוד לייצוג גרפים הבאים:

1) $f=\ln x + x^2$, $x\in[1; 7]$, צעד 0,4;

2) $f=x^2$, $y=\sin x$, $x\in[-5; 5]$, צעד 0,5;

3) $f=\sin x^2 - \cos x$, $y= x^2 - 3$, $x\in[-4; 4]$, צעד 0,3;

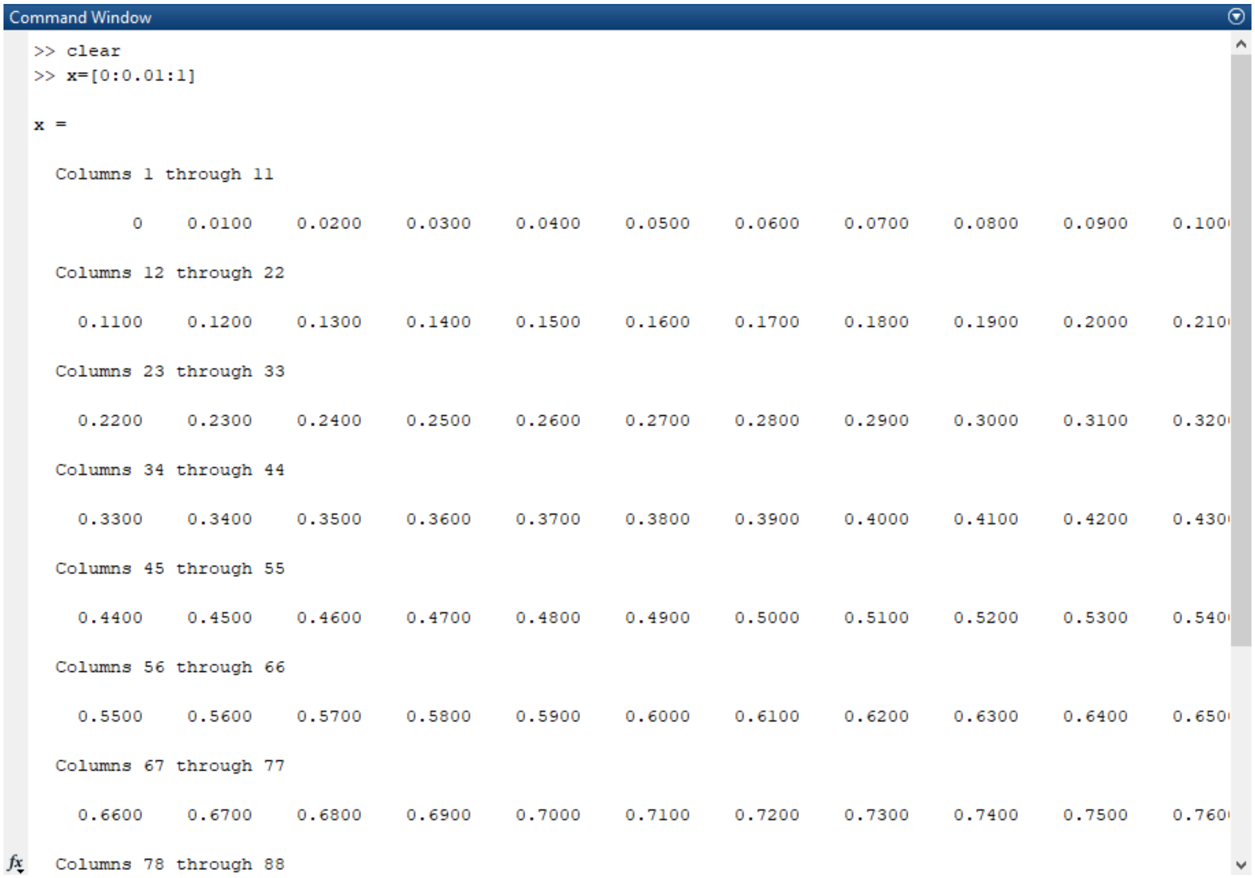
4) $f= \sin x^2 + \cos x$, $y=x^2 - 4$, $z=\sqrt{|x|} - 0{,}8$, $x\in[-4; 4]$, צעד 0,4.

ב. שימוש בתפריט מאפייני גרף

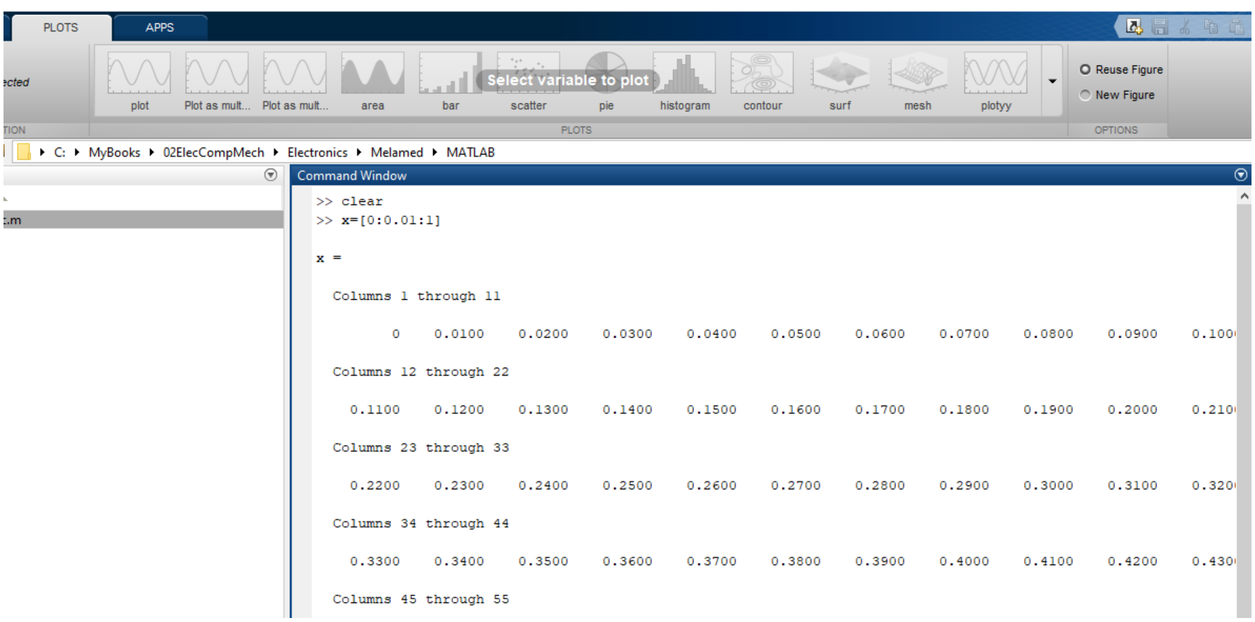
<u>מאפייני הגרף:</u>

בתוכנת MATLAB ישנה אופציה לבנות גרפים בעזרת תפריט.

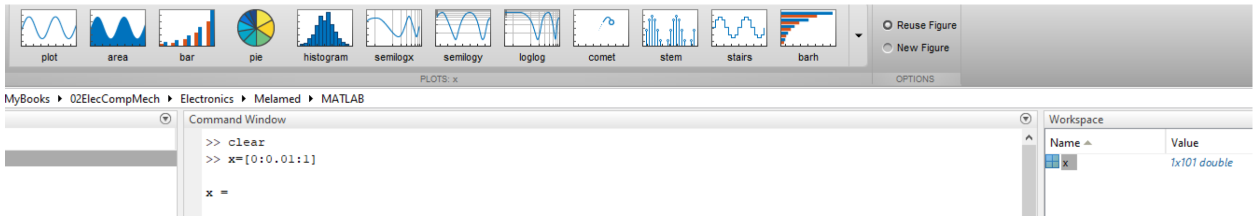
נגדיר ווקטור X בין 0 ל-1 עם צעד 0.01 .

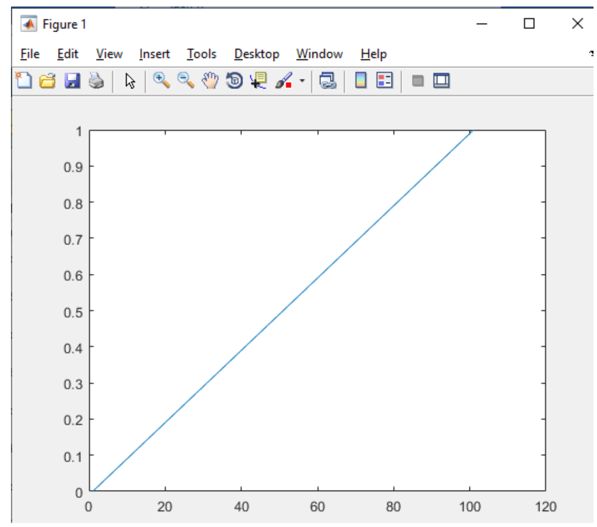


נבחר תפריט PLOT.



ונבחר משתנה לבניית גרף ונבחר סוג הגרף:





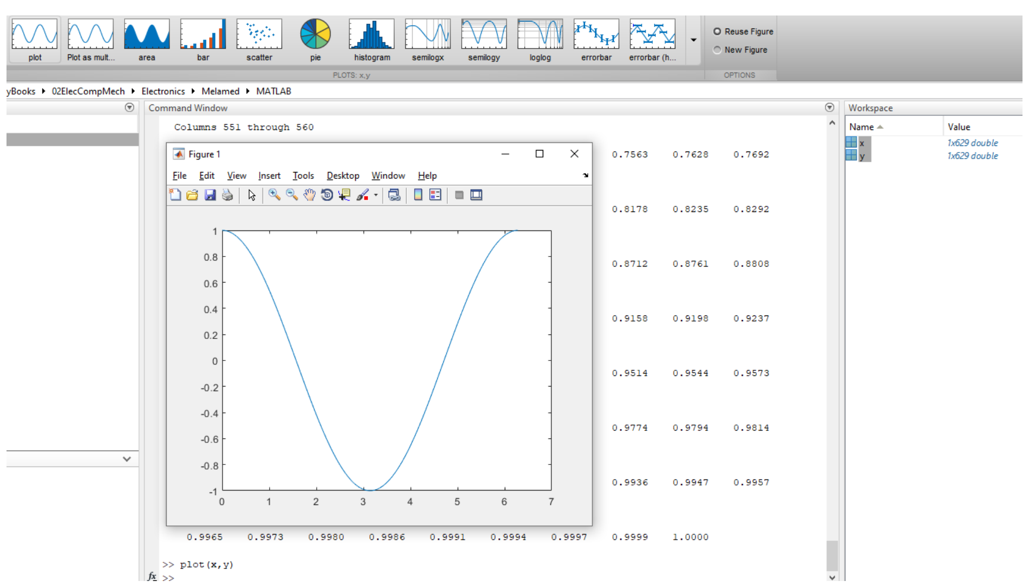
נעשה את אותה פעולה לפונקציה קוסינוס.

נגדיר משתנה X ומשתנה Y:

קלט:

```
>> x=[0:0.01:2*pi]
>> y=cos(x)
```

כדי לבחור פונקציה של 2 משתנים יש להשתמש בלחצן CTRL ביחד עם לחצן ימני על העכבר.



ג. ריכוז פרמטרים חשובים של הגרף.

ניתן לשנות פרמטרים חשובים של הגרף בעזרת חלון ההגדרות.

פרמטרים חשובים לבניית גרפים:

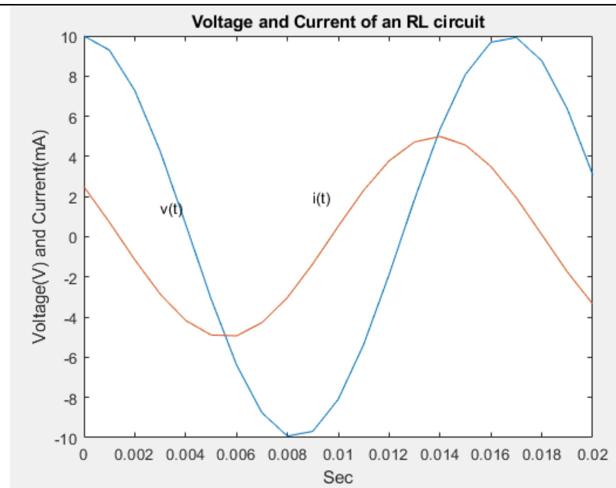
סוג פעולה	תיאור	פונקציה
חלון של הגרף	שם של הגרף כולו	Figure Name
	צבע של חלון הגרף	Figure Color
הגרף עצמו	שם הגרף	Title
	צבע של רקע של אזור הגרף	Colors
	רשת	Grid
	ציר X שם של ציר X קני מידה של ציר X סוג של קני מידה X רשתית על צירים	X Axis: X Label X Limits X Scale Ticks
	גופן	Font
קו של הגרף	שם של קו ספציפי	Display Name
	סוג הגרף	Plot Type
	סוג הקו	Line Style
	עובי של הקו	Line Width
	צבע של הקו	Color
	סוג המרקר	Marker
	גודל של המרקר	Marker Size
	צבע של המרקר	Marker Edge Color
הערות	סוג הקו	Line Style
	עובי של הקו	Line Width
	צבע של הקו	Edge Color
	צבע של הרקע	Background

דוגמא: גרף של פונקציה למעגל RL.

$v(t)=10 \cos(377t)$

$i(t)=5 \cos(377t+60^\circ)$

תכנן גרף של מתח וזרם בין 0 עד 20 מילי שניות.

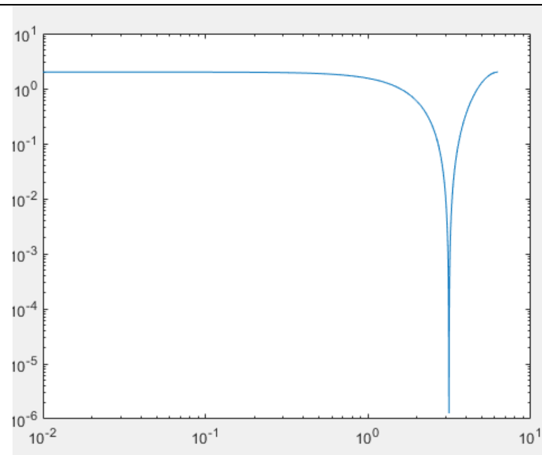
	קלט:
	`>> t = 0:0.001:0.02; %time from 0 to 20 msec`
	`% with 1 msec interval`
	`v = 10*cos(377*t); % voltage`
	`a_rad = (60*pi/180); % angle in radians`
	`i = 5*cos(377*t + a_rad); %current`
	`plot(t,v,'-',t,i,'-')`
	`title('Voltage and Current of an RL circuit')`
	`xlabel('Sec')`
	`ylabel('Voltage(V) and Current(mA)')`
	`text(0.003, 1.5, 'v(t)');`
	`text(0.009,2, 'i(t)')`

ד. פונקציות לוגריתמיות וחצי-לוגריתמיות.

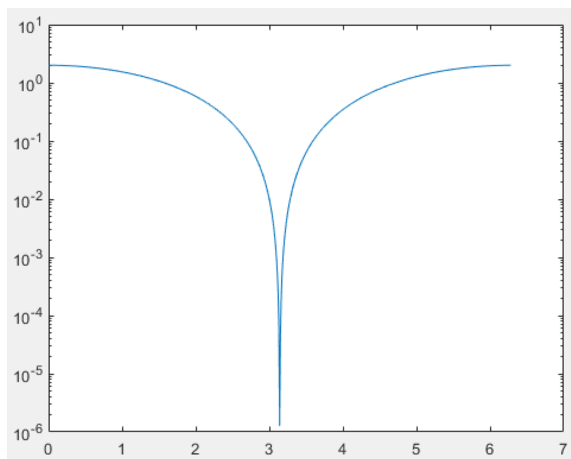
כדי לעשות גרפים מיוחדים ודי שימושים באלקטרוניקה יש להשתמש בפקודות הבאות:

קלט:

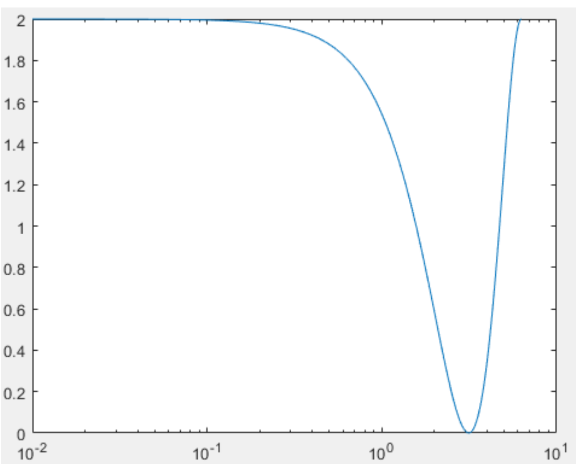
```
>> x=[0:0.01:2*pi]
>> y=1+cos(x)
>> loglog(x, y)
```




```
>> semilogy(x, y)
```




```
>> semilogx(x, y)
```



דוגמא:

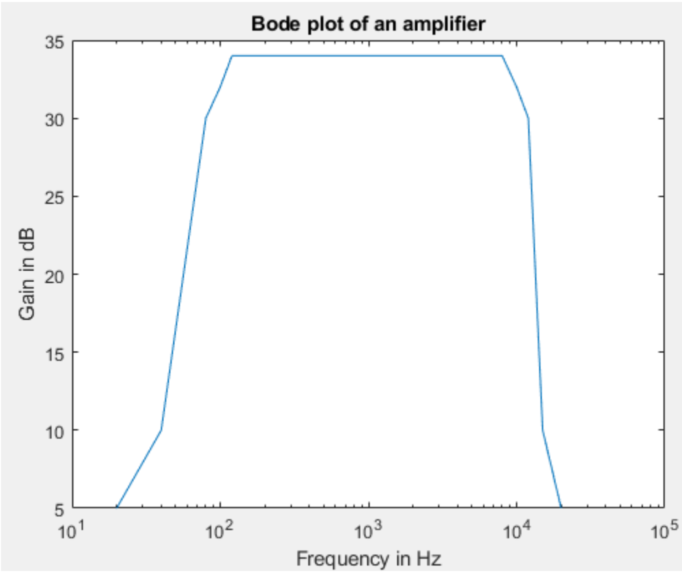
נגדיר טבלה הגברים של מסנן בהתאם לתדר:

תדר Hz	הגבר dB	תדר Hz	הגבר dB
20	5	2000	34
40	10	5000	34
80	30	8000	34
100	32	10000	32
120	34	12000	30

הקוד שיש לכתוב בתוכנת MATLAB ייראה כך:

```
>> f = [20 40 80 100 120 2000 5000 8000 10000 12000 15000 20000];
g = [ 5 10 30 32 34 34 34 34 32 30 10 5];
semilogx(f, g)
title('Bode plot of an amplifier')
xlabel('Frequency in Hz')
ylabel('Gain in dB')
```

פלט:



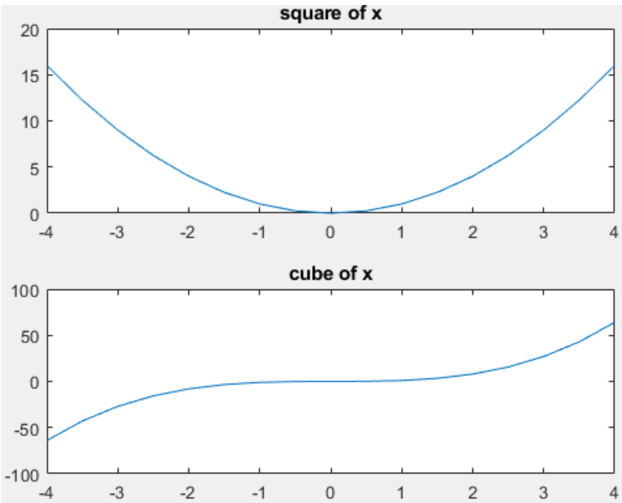
Bode plot of an amplifier
Gain in dB
Frequency in Hz
35
30
25
20
15
10
5
10^1
10^2
10^3
10^4
10^5

ה. ישנה אפשרות לראות 2 גרפים על אותו עמוד. (שימוש בפונקציה subplot).

קלט:

```
>> x = -4:0.5:4;
y = x.^2; % square of x
z = x.^3; % cube of x
subplot(211), plot(x, y), title('square of x')
subplot(212), plot(x, z), title('cube of x')
```

פלט:



תרגיל:

בנה מעגל בתוכנת מולטיסים עם מסנן LPF ומסנן HPF (RC ו- CR) לתדרים מ- 10 הרץ עד 100 קילו הרץ.

מצא את ההגברים כתלות בתדר ורשום אותם בטבלה. העבר את המדידות לתוכנת MATLAB ובנה 2 גרפים אחד מתחת לשני של 2 המסננים הנ"ל. (השתמש בנגד של 1K וקבל של 1 uF).

פרק 5. חקירת פונקציות

א. ישנן שלוש אופציות למציאת פתרון של פונקציה.

<u>אופציה ראשונה: בניית גרף.</u>

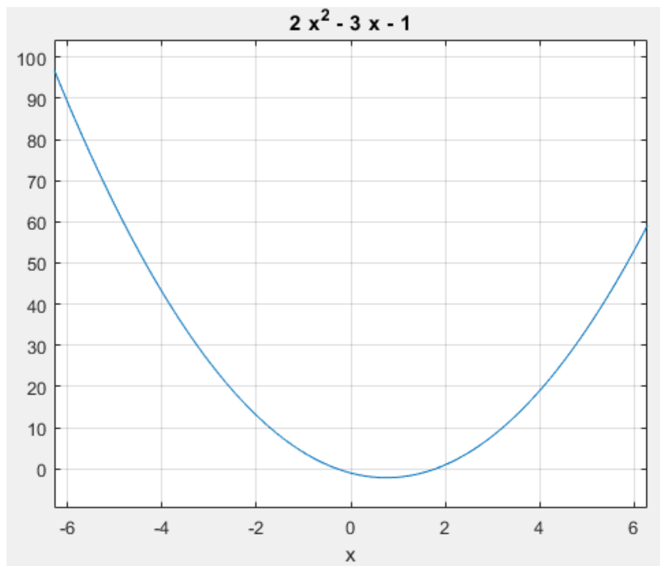
דוגמא:

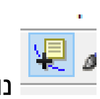
נתונה הפונקציה: $2x^2 - 3x + 1 = 0$, מצא את השורשים של המשוואה.

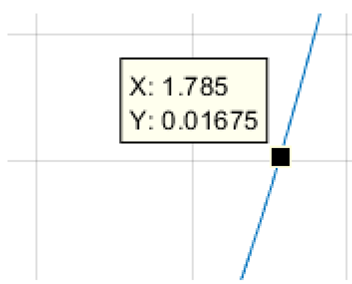
א. נגדיר משתנים.
ב. נגדיר את הפונקציה.
ג. נבנה את גרף הפונקציה.
ד. נוסיף קווי רשת (grid).
ה. בעזרת הגרף נוכל למצוא את הפתרון (או פתרונות).
ו. קלט:

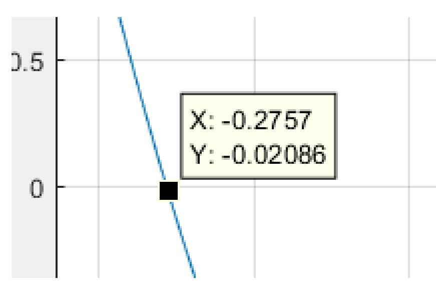
```
>> syms x f
f=2*x^2-3*x-1;
ezplot(f)
grid
```

פלט:



אם נשתמש בכלי  ובכלי  נוכל למצוא שורשים בצורה גרפית:





כמובן שהפתרון לא מדויק ויש דרך אחרת לפתרון מתמטי של המשוואה:

<u>אופציה שנייה:</u>

במקרה והפונקציה היא פולינום ניתן להשתמש בפונקציה root(), ראשית יש להגדיר את המקדמים של הפולינום, מהחזקה הגבוהה ביותר עד הנמוכה ביותר.

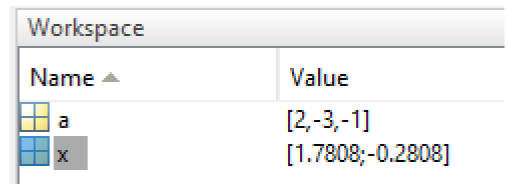
קלט:

```
>> a=[2 -3 -1];
x=roots(a)
```

פלט:

```
x =
   1.7808
  -0.2808
```

בחלון המשתנים, הנתונים ייראו כך:



:

<u>אופציה שלישית:</u>

אופציה זאת מתאימה לכל המשוואות ולא רק לפולינומים. ניתן להשתמש בפונקציה solve().

א. נגדיר משתנים.
ב. נגדיר את הפונקציה.
ג. נפעיל את הפונקציה SOLVE.
ד. נדפיס את התוצאה עם הדיוק הרצוי.

קלט:

```
>> syms x f
f=2*x^2-3*x-1;
r=solve(f,x)
vpa(r,5)
```

פלט:

```
r =
 3/4 - 17^(1/2)/4
 17^(1/2)/4 + ¾
ans =
 -0.28078
1.7808
```

דוגמא נוספת לפתרון פונקציה.

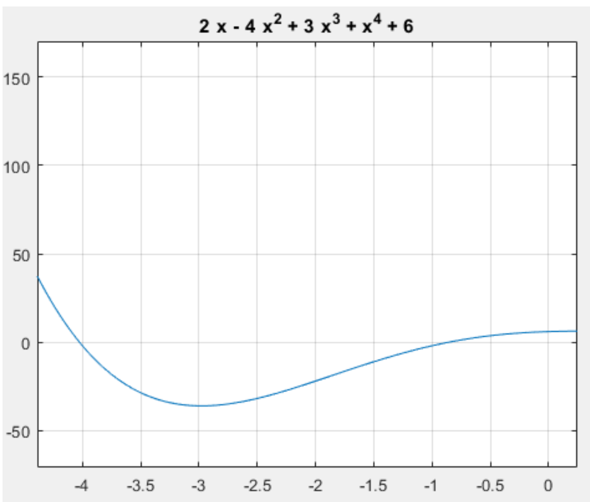
נתונה המשוואה הבאה:

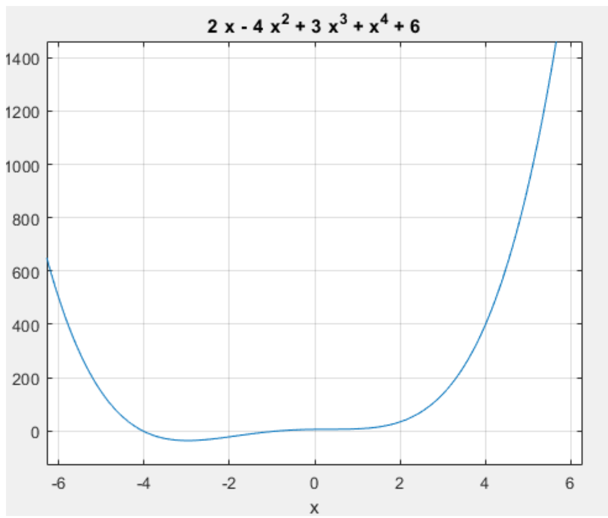
$$f(x) = x^4 - 3x^3 - 4x^2 + 2x - 6$$

<u>פתרון בשיטה הראשונה: (יצירת גרף)</u>

קלט:

```
>> syms x f
f=x^4+3*x^3-4*x^2+2*x+6
ezplot(f)
grid
```





<u>פתרון בשיטה השנייה: (בעזרת פולינום)</u>

קלט:

```
>> a=[1 -3 -4 2 -6];
>> x=roots(a)
```

פלט:

```
x =
  3.9752 + 0.0000i
 -1.6237 + 0.0000i
  0.3243 + 0.9080i
  0.3243 - 0.9080i
```

שימו לב ששני שורשים הם מספרים ממשיים ושני מספרים מדומים.

<u>פתרון בשיטה השלישית: (SOLVE)</u>

קלט:

```
>> syms x f
>> f=x^4-3*x^3-4*x^2+2*x-6
 f =
```

```
x^4 - 3*x^3 - 4*x^2 + 2*x - 6
>> r=solve(f,x)
```

פלט:

```
r =
 root(z^4 - 3*z^3 - 4*z^2 + 2*z - 6, z, 1)
 root(z^4 - 3*z^3 - 4*z^2 + 2*z - 6, z, 2)
 root(z^4 - 3*z^3 - 4*z^2 + 2*z - 6, z, 3)
 root(z^4 - 3*z^3 - 4*z^2 + 2*z - 6, z, 4)
```

קלט:

```
>> vpa(r,5)
```

פלט:

```
ans =
           -1.6237
            3.9752
 0.32426 + 0.90798i
 0.32426 - 0.90798i
```

תרגול:

פתור את המשוואות הבאות בצורה גרפית ובעזרת SOLVE.

a. $|x+5|-|2x-1|=3;$
b. $\sqrt{|2x+1|}-1=0;$
c. $\ln(x+2)=2;$
d. $|x+1|+|x+2|-2=0;$
e. $x^3+2|x-1|=0$

פתור את המשוואות הבאות בצורה בעזרת SOLVE ובעזרת ROOT.

a. $1=2x^2-3x-6$
b. $0=3x^3-8x^2+2x+2$
c. $0=x^4-3x^3-4x^2+2x-6$
d. $0=10x^3-3x^2-2x+0.5$
e. $0=2x^2+x-2$

ב. חישוב אינטגרלים וחישוב שטחים בעזרת שיטת טרפז.

חישוב אינטגרל לא מסוים (ללא גבולות).

לדוגמא עבור הפונקציה x^3 שהאינטגרל שלה הינו: $\int x^3 dx$

נכתוב את הקוד הבא:

קלט:

```
>> syms x
f=x^3;
int(f,x)
```

פלט:

```
ans =
 x^4/4
```

חישוב אינטגרל מסוים (שטח של פונקציה בין גבול תחתון לגבול עליון).

עבור אותה פונקצייה בגבולות 1 עד 3 שהאינטגרל שלה הינו: $\int_1^3 x^3 dx$

נכתוב את הקוד הבא:

קלט:

```
>> syms x
f=x^3;
int(f,x,1,3)
```

פלט:

```
ans =
 20
```

או בצורה אחרת:

קלט:

```
>>syms x
f=x^3;
vpa(int(f,x,1,3),3)
```

פלט:

```
ans =
 20.0
```

דוגמא:

מנורה עם הספק 100W והתנגדות 144 אוהם מחוברת למקור מתח $v(t) = 120\sqrt{2}\cos(120\pi t)$.

כמה אנרגיה מבזבזת המנורה במשך שעה?

פתרון:

נמצא זרם דרך המנורה:

$$i(t)=\frac{v(t)}{R}=\frac{120\sqrt{2}\cos(120\pi t)}{144}=\frac{5\sqrt{2}}{6}\cos(120\pi t)\ A$$

מכאן נמצא את ההספק:

$$p(t)=v(t)\cdot i(t)=\left[120\sqrt{2}\cos(120\pi t)\right]\cdot\left[\frac{5\sqrt{2}}{6}\cos(120\pi t)\right]=200\cos^2(120\pi t)W$$

כדי לחשב אנרגיה נצטרך לחפש את האינטגרל המסוים מ-0 עד 3600 שניות.

$$W=\int_0^{3600}200\cos^2(120\pi t)dt=200\left[\frac{t}{2}+\frac{1}{480}\sin(240\pi t)\right]_0^{3600}=200\cdot 1800=360{,}000J$$

נבדוק את הפתרון שלנו בעזרת MATLAB:

קלט:

```
>>syms t
f=200*cos(120*pi*t)^2
vpa(int(f,t,0,3600),3)
 f =
 200*cos(120*pi*t)^2
```

פלט:

```
ans =
3.6e5
```

דרך נוספת לבדוק שטחים של הפונקציה – היא שימוש בחישוב שטחים של טרפז.

לצורך כך אפשר להשתמש בפונקציה מובנית:

קלט:

```
area=trapz(t,x)
```

כאשר t מסמן את הווקטור של הנתונים בציר X ו- x מסמן ערכים של פונקציה t לפי ציר Y.

דוגמא:

נחשב מתח ממוצע של חצי גל חיובי, כאשר מתח כניסה הוא מתח AC ומתח מקסימלי הוא 170V .

נגדיר גבול עליון של חצי גל: T=pi

נגדיר מתח מקסימלי: Vm=170

נגדיר ווקטור שמתחיל ב-0 מסתיים ב-T ומכיל 1024 ערכים בין הגבולות האלה: theta=linspace(0,T,1024)

נחשב מהו מתח בכל נקודה של הווקטור של theta: v=Vm*sin(theta)

נחשב את הערך הממוצע של גל כניסה: V0=1/T*trapz(theta,v)

קלט:

```
>> T=pi;
Vm=170;
theta=linspace(0,T,1024);
v=Vm*sin(theta);
V0=1/T*trapz(theta,v)
```

פלט:

```
V0 =
  108.2253
```

נבדוק את המתחה הממוצע לפי נוסחה של חצי גל: $V_{av}=\frac{2V_m}{\pi}$

קלט:

```
>> Vav=2*Vm/pi
```

פלט:

```
Vav =
  108.2254
```

דיוק החישוב שהתקבל הינו גבוה מאוד.

כדי לבדוק את דיוק השיטה נחשב מתח ממוצע של גל שלם, כאשר מתח כניסה הוא מתח AC ומתח מקסימלי הוא 170V.

אנו יודעים שמתח ממוצע של הגל שווה ל-0.

קלט:

```
>> T=2*pi;
Vm=170;
theta=linspace(0,T,1024);
v=Vm*sin(theta);
V0=1/T*trapz(theta,v)
```

פלט:

```
V0 =
   1.8222e-15
```

אם כן ניתן לראות שדיוק החישוב הינו גבוה מאוד.

נחשב ערכים של מתח וזרם אפקטיבי בעזרת אינטגרל:

נוסחאות של החישוב הם: $I_{RMS}=\sqrt{\frac{1}{T}\cdot\int_0^T i^2(t)dt}$ $V_{RMS}=\sqrt{\frac{1}{T}\cdot\int_0^T v^2(t)dt}$

נחשב את המתח האפקטיבי לפי שני דרכים:

דרך אינטגרל:

קלט:

```
>> syms t
f=170*sin(120*pi*t)
T=pi
vrms=sqrt(1/T*int(f^2,t,0,T))
vpa(vrms,3)
```

פלט:

```
ans =
120.0
```

קלט:

```
>> vpa(vrms,7)
```

פלט:

```
ans =
 120.2096
```

דרך חישוב טרפז:

קלט:

```
>>T=pi;
Vm=170;
theta=linspace(0,T,1024);
v=Vm*sin(theta);
Vrms=sqrt(1/T*trapz(theta,v.^2))
```

פלט:

```
Vrms =
  120.2082
```

ג. נגזרות.

לצורך חישוב נגזרות ישנן 4 פקודות שימושיות:

נגזרת מסדר ראשון	diff(f)
נגזרת מסדר k	diff(f,k)
נגזרת מסדר ראשון על משתנה x (שימושי כאשר בפונקציה יש מספר משתנים)	diff(f,x)
נגזרת מסדר ראשון k על משתנה x (שימושי כאשר בפונקציה יש מספר משתנים)	diff(f,x,k)

נתונה הפונקציה: y=2x3–3x2+3 , נגזור אותה (כמובן לפי X).

קלט:

```
>> syms x
y=2*x^3-3*x^2+3;
diff(y)
```

פלט:

```
ans =
 6*x^2 - 6*x
```

נתונה הפונקציה: y=sin(x+h) מצא נגזרת לפי X.

קלט:

```
>> syms x h
y=sin(x+h);
diff(y,x)
```

פלט:

```
ans =
cos(h + x)
```

לצורך חיפוש מינימום או מקסימום יש למצוא נגזרת שנייה של הפונקציה y=5/x .

קלט:

```
>> syms x
diff(5/x,2)
```

פלט:

```
ans =
10/x^3
```

נתונה הפונקציה: y=3x3h–2h2x2+3 מצא נגזרת שנייה של הפונקציה הנתונה.

קלט:

```
>> syms x h
y=3*x^3*h-2*h^2*x^2+3
diff(y,x,2)
```

פלט:

```
y =
 - 2*h^2*x^2 + 3*h*x^3 + 3
 ans =
 - 4*h^2 + 18*x*h
```

ד. משוואות דיפרנציאליות.

במקצוע בקרה יש צורך לפתור משוואות דיפרנציאליות. MATLAB מהווה כלי שימושי גם בפתרון בעיות אלה.

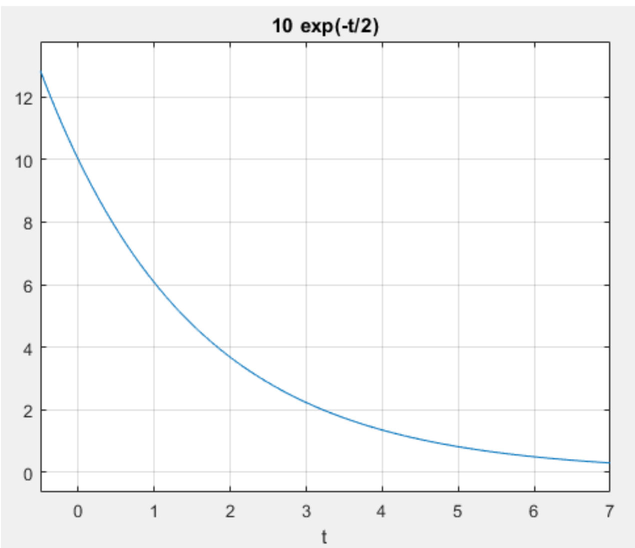
נניח שאנו צריכים לפתור משוואה $\frac{dx}{dt} = -0{,}5x$ עם ערך התחלתי של x(0)=10 בקטע - [–0,5; 7].

קלט:

```
>> x=dsolve('Dx=-0.5*x','x(0)=10')
ezplot(x,[-0.5,7]);
grid
```

פלט:

```
x =
 10*exp(-t/2)
```



ישנה אפשרות לפתור מערכת משוואות דיפרנציאליות ולקבל את הפתרון בהצגה גרפית וגם באמצעות הצגה של שתי פונקציות:

נחפש פתרון למערכת הבא בשתי נקודות: x1(0)=0, x2(0)=1 בקטע: [–0,5; 13].

שתי פונקציות:

$$\begin{cases} \frac{dx_1}{dt} = -0{,}5x_2; \\ \frac{dx_2}{dt} = 3x_1; \end{cases}$$

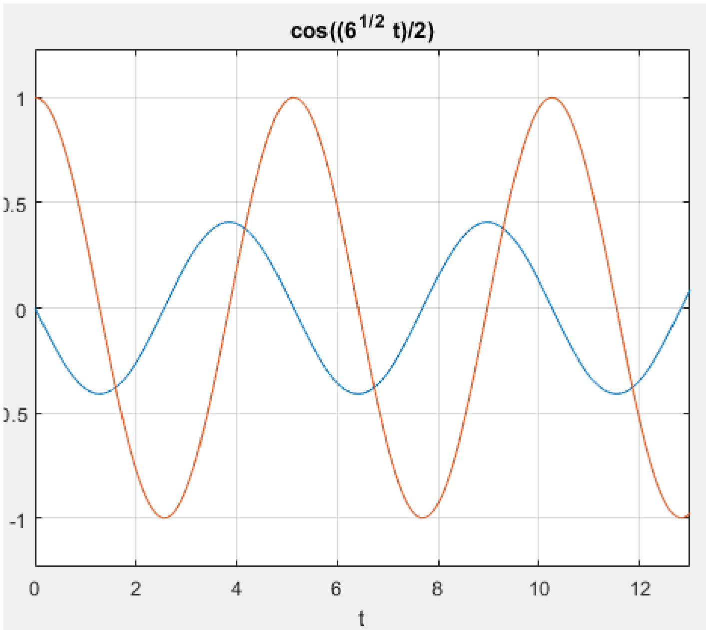
קלט:

```
>> [x1,x2]=dsolve('Dx1=-0.5*x2','Dx2=3*x1','x1(0)=0','x2(0)=1')
ezplot(x1,0,13)
grid
hold on
ezplot(x2,[0,13])
```

פלט:

```
x1 =
 -(6^(1/2)*sin((6^(1/2)*t)/2))/6
 x2 =
cos((6^(1/2)*t)/2)
```

גרף:



תרגול עצמי:

יש לחשב את האינטגרלים הבאים:

א. לא מסוימים.

1) $\int \cos x dx;$

2) $\int x^2 dx;$

3) $\int (e^x - x) dx;$

4) $\int (3x + \ln x) dx;$

5) $\int (x^3 + 3x^2 + 1) dx;$

6) $\int \frac{1}{x} dx;$

7) $\int \sqrt{x^2 + 1}\, dx;$

8) $\int \frac{\sqrt{x} + 1}{x^2 + 2x} dx.$

ב. מסוימים.

1) $\int_1^3 \sin x dx;$

2) $\int_1^3 (x^2-\cos 2x)dx;$

3) $\int_1^2 (\sin x+\frac{x^2}{2})dx;$

4) $\int_0^5 (0{,}5^x\cdot\cos x)dx;$

5) $\int_0^4 (\sin(2x+3)-2\cos 5x)dx;$

6) $\int_{-5}^0 (e^x\sin(x+3))dx.$

ג. מצא נגזרות של הפונקציות הבאות:

1) $y(x)=x^2+3x+1;$

2) $y(x)=\frac{2x^2+1}{\sqrt{x+2}};$

3) $y(x)=\arcsin 2x;$

4) $y(x)=\sqrt{x-3}+\frac{\mathrm{tg}2x}{4};$

5) $y(x)=x-\cos x;$

6) $y(x)=\sin x^2+x^2;$

7) $y(x)=0{,}5^x\cdot\cos x;$

8) $y(x,z)=5(\sin x-\cos 5z);$

9) $y(x,z)=\sin(z+3)+2^x.$

ד. מצא נגזרת שנייה של הפונקציות הבאות:

1) $y(x)=x^2-\cos x,$

2) $y(x)=e^{-2x}+x^3,$

3) $y(x)=e^x+x^4/3,$

ה. מצא פתרון למשוואות דיפרנציאליות כאשר נתון תנאי התחלתי.

1) $\frac{dx}{dt}=t\cos t+\frac{x}{t},\ x(1)=0;$

2) $\frac{dx}{dt}=\frac{1}{e^{t-x}},\ x(1)=1;$

3) $\frac{dx}{dt}=\frac{1+x^2}{1+t^2},\ x(0)=1;$

4) $\frac{dx}{dt}+x=\cos t,\ x(0)=\frac{1}{2};$

5) $\frac{dx}{dt}-x\cdot\mathrm{tg}t=\frac{1}{\cos t},\ x(0)=0;$

6) $(x+t)\frac{dx}{dt}=1,\ x(-1)=0;$

7) $(1+e^t)x\frac{dx}{dt}=e^t,\ x(0)=1;$

8) $\frac{dx}{dt}=t^2x^4-\frac{x}{t},\ x(1)=\frac{1}{\sqrt[3]{3}}.$

פרק 6. מבוא לתכנות ושימוש בקבצי פקודות.

א. מבוא לתכנות:

כמו בשפת C גם בסביבת MATLAB יש שפת תכנות הדומה בעיקרה לשפת C, אך קצת שונה מבחינה דיקדוקית.

נתחיל ממשפט if. בסביבת MATLAB המבנה יראה כך:

If תנאי

פעולה או פעולות

elseif תנאי

פעולה או פעולות

else פעולה או פעולות

end

דוגמא של בדיקת סוג המשולש לפי צדדים a,b,c:

קלט:

```
>> a=3;
b=4;
c=5;
>> if a<b+c
disp('triangle valid')
else disp('Triangle not valid')
end
```

פלט:

```
triangle valid
```

כמובן שתנאי עבור צד אחד לא מספיק – ולכן נבצע בדיקה יותר מורכבת ונלמד איך לשלב תנאים.

פעולת AND לוגי	&	שווה	==
פעולת NOT לוגי	~	גדול או שווה	>=
פעולת OR לוגי	|	קטן או שווה	<=
פעולת XOR לוגי	xor	גדול מ-	>
בודק האם כל האיברים של מערך במצב "אמת"	all	קטן מ-	<
בודק האם יש לפחות אחד בין האיברים של מערך במצב "אמת"	any	שונה מ-	~=
0 לוגי	false	בודק האם מערכים שווים	isequal
חיפוש של כל האיברים שאינם אפסים	find		
הפיכת ערכים מספרים לערכים לוגיים	logical		
1 לוגי	true		

תוכנית שבודקת את כל האפשרויות בעזרת פונקציות לוגיות:

קלט:

```
>> a=3;
b=4;
c=5;
>> if a<b+c & b<a+c & c<b+a
disp('triangle valid')
else disp('Triangle not valid')
end
```

פלט:

```
triangle valid
```

משפט switch case.

במידה ויש בתנאי ערכים בודים (discrete) ניתן להשתמש במשפט הנ"ל.

מבנה כללי:

switch משתנה לבדיקה

  case ערך בודד או מספר ערכים בודדים

    פעולות או פעולה

  case ערך בודד או מספר ערכים בודדים

    פעולות או פעולה

  ...

  otherwise

    ערך בודד או מספר ערכים בודדים

end

ניתן להציג נתונים באמצעות גרף במספר תצוגות שונות.

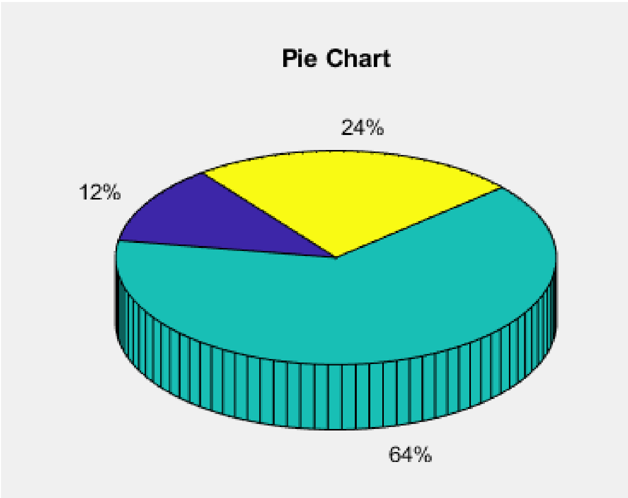
לדוגמה, כאשר נבצע את הקוד הבא:

קלט:

```
x = [12 64 24];
plottype = 'pie3';

switch plottype
    case 'bar'
        bar(x)
        title('Bar Graph')
    case {'pie','pie3'}
        pie3(x)
        title('Pie Chart')
otherwise
        warning('Unexpected plot type. No plot created.')
end
```

נקבל כתוצאה את הגרף:

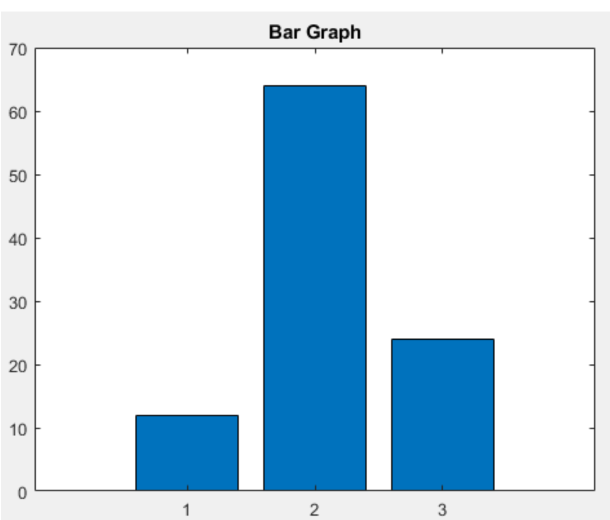


כאשר נשנה את הערך במשתנה plottype ל-bar.

קלט:

```
>> x = [12 64 24];
plottype = 'bar';
switch plottype
   case 'bar'
      bar(x)
      title('Bar Graph')
   case {'pie','pie3'}
      pie3(x)
      title('Pie Chart')
   otherwise
      warning('Unexpected plot type. No plot created.')
end
```

נקבל את הגרף:



לולאות.

כאשר רוצים למלא טור או ווקטור בערכים סידרתיים. ניתן לבצע פעולה זאת בעזרת לולאות.

סוג ראשון של לולאות הינו לולאת for.

בהבדל מלולאת for בשפת C, ב-MATLAB ישנם שתי אופציות:

- אופציה ראשונה כאשר יש רק שני שדות – ערך התחלתי וערך סופי והצעד הוא 1 או 1- כברירת מחדל.
- אופציה שנייה כאשר אנו מגדירים ערך התחלתי, צעד וערך סופי.

נדגים את שתי האופציות.

אופציה א:

קלט:

```
>> for v = 1.0:5.0
  disp(v)
end
```

פלט:

```
     1
     2
     3
     4
     5
```

אופציה ב:

קלט:

```
>> for v = 1.0:0.5:5.0
  disp(v)
end
```

פלט:

```
1
   1.5000
     2
   2.5000
     3
   3.5000
     4
   4.5000
   5
```

לולאת while עובדת כמו בשפת C לא לפי כמות החזרות, אלא לפי תנאי מסוים.

סינטקס הפקודה דומה מאוד לשפת C.

```
while תנאי
   הוראה או הוראות
end
```

לדוגמה נכתוב תוכנית לחישוב פעולת עצרת: (5!=1*2*3*4*5=120)

קלט:

```
>> n = 5;
f = n;
while n > 1
   n = n-1;
   f = f*n;
end
disp('n! =  ' )
disp(f)
```

פלט:

```
n! =
 120
```

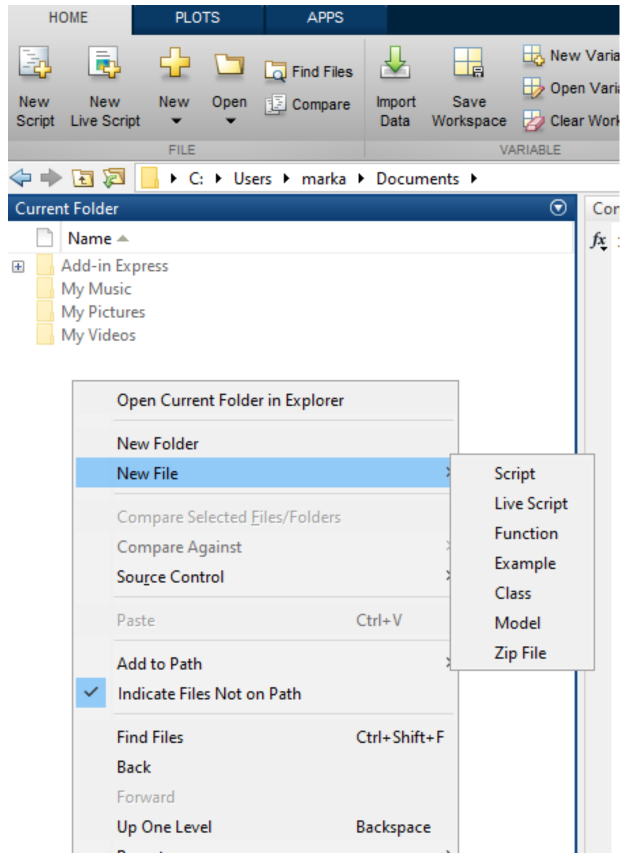
ב. הגדרת קבצים:

בתוכנת MATLAB ישנם שני סוגים של קבצי פקודות: פונקציות או סקריפטים. הם מכילים רצף פקודות ללא קלט או פלט.

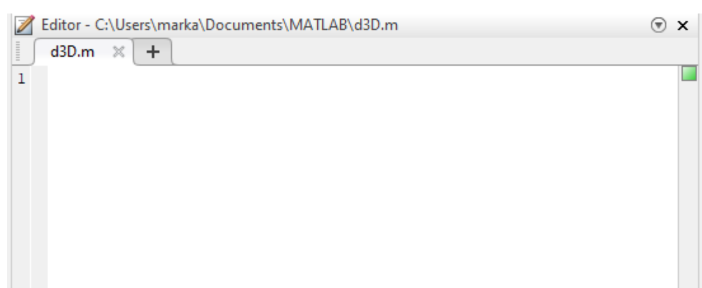
נבנה לדוגמא סקריפט למציאת אורך של ווקטור במרחב כאשר שתי נקודות נתונות מראש והמשתמש צריך להכניס נקודה שלישית:

נבחר בחלון הקבצים מקום לשמירת הנתונים (כדאי לשים לב שנתיב שמירת הקובץ לא יכלול אותיות בעברית, ערבית, רוסית או כל שפה אחרת משום שחלק מגרסאות התוכנה עובדות באנגלית בלבד).

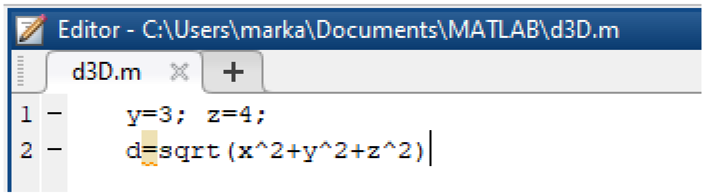
לאחר לחיצה על כפתור ימין של העכבר ייפתח החלון הבא, בו נבחר בערך סקריפט, ניתן לו שם ונלחץ פעמיים עם הלחצן השמאלי של עכבר.



נפתח חלון – Script Editor.



בחלון זה נרשום את הפקודות הבאות:



נשמור את התוצאה ונפעיל את הקובץ. ניתן לבצע הפעלה בשתי דרכים: (בשני המקרים קודם נצטרך להשלים את הנתון מהמשתמש שחסר, לדוגמה x=5 ואז נפעיל את הסקריפט).

1. ניתן לבצע פקודת RUN לאחר לחיצה על הקובץ.
2. כתיבת שם הסקריפט ישירות בתוך מסך ראשי:

קלט:

```
>> x=5
```

פלט:

```
x =
    5
```

קלט:

```
>> d3D
```

פלט:

```
d =
   7.0711
```

ניתן להגדיר סקריפט ללא פרמטרים בתוך הסקריפט.

נדגים זאת ע"י משוואה ריבועית. נגדיר קובץ sq_eq.m ונרשום את הסקריפט הבא:

קלט:

```
D=b^b-4*a*c
if D<0
   disp('No Roots')
elseif D==0
   x=-b/(2*a)
   disp('X= ')
   disp(x)
else
   x1=(-b+sqrt(D))/(2*a)
   x2=(-b-sqrt(D))/(2*a)
   disp('x1 =')
   disp (x1)
   disp('x2 =')
   disp (x2)
end
```

נגדיר את הערכים של a,b,c ונפעיל את הסקריפט:

קלט:

```
>> a=1;
b=4;
c=1;
 sq_eq
```

פלט:

```
D =
      252
x1 =
        5.9373
x2 =
       -9.9373
```

השימוש בקבצי הפקודות יכול להיות כscript- או כפונקציה. יש ליצור קבצי פקודות בהתאם למשימות ולתעד אותם כראוי:

פרק 7. תורת חשמל בעזרת MATLAB.

א. ניתוח מעגל RL במתח חילופין.

נתון מתח מקור עם מתח אפקטיבי של 120V ועומס Z_L=10+7j. יש לחשב:

זווית בין מתח לזרם

ערך מוחלט של התנגדות הצרכן

הספק ממשי

הספק מדומה

הספק מרוכב

קלט:

```
>> Vs=120
```

פלט:

```
Vs =
  120
```

קלט:

```
>> ZL=10+j*7
```

פלט:

```
ZL =
  10.0000 + 7.0000i
```

כדי למצוא ערך ממשי וערך מדומה של העומס יש להשתמש בפונקציות הבאות:

קלט:

```
>> R = real(ZL)
```

פלט:

```
R =
   10
```

קלט:

```
>> X=imag(ZL)
```

פלט:

```
X =
    7
```

כדי למצוא ערך מוחלט וזווית של העומס יש להשתמש בפונקציות הבאות:

קלט:

```
>> absZL=abs(ZL)
```

פלט:

```
absZL =
  12.2066
```

קלט:

```
>> phiZL=angle(ZL)
```

פלט:

```
phiZL =
   0.6107
```

ישנה גם אפשרות לחשב אותם לפי נוסחאות מתמטיות:

קלט:

```
>> absZL=sqrt(R^2+X^2)
```

פלט:

```
absZL =
  12.2066
```

קלט:

```
>> phiZL=atan(X/R)
```

פלט:

```
phiZL =
   0.6107
```

כדי להעביר זווית מרדיינים למעלות יש להשתמש ביחס - $\frac{\alpha^{\circ}}{180^{\circ}} = \frac{\alpha_{\text{rad}}}{\pi}$, מכאן נקבל:

קלט:

```
>> phiZLM=phiZL*180/pi
```

פלט:

```
phiZLM =
  34.9920
```

נחשב זרם במעגל:

קלט:

```
>> i=Vs/ZL
```

פלט:

```
i =
  8.0537 - 5.6376i
```

קלט:

```
>> iabs=abs(i)
```

פלט:

```
iabs =
   9.8308
```

קלט:

```
>> iphi=angle(i)
```

פלט:

```
iphi =
  -0.6107
```

נחשב הספקים:

קלט:

```
>> P=Vs*iabs*cos(phiZL)
```

פלט:

```
P =
 966.4430
```

קלט:

```
>> Q=Vs*iabs*sin(phiZL)
```

פלט:

```
Q =
 676.5101
```

קלט:

```
>> S=Vs*iabs
```

פלט:

```
S =
  1.1797e+03
```

קלט:

```
>> S1=sqrt(P^2+Q^2)
```

פלט:

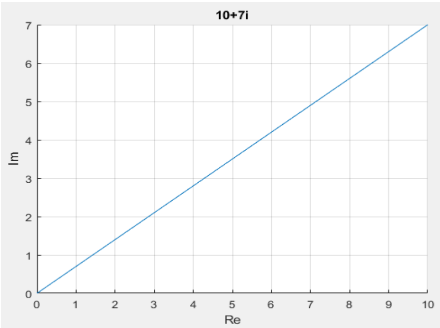
```
S1 =
  1.1797e+03
```

כדי להציג מספר מרוכב בצורה גרפית נשתמש בפונקציה LINE בצורה הבאה:

קלט:

```
>> line([0 10],[0 7])
xlabel('Re')
ylabel('Im')
grid
title('10+7i')
```

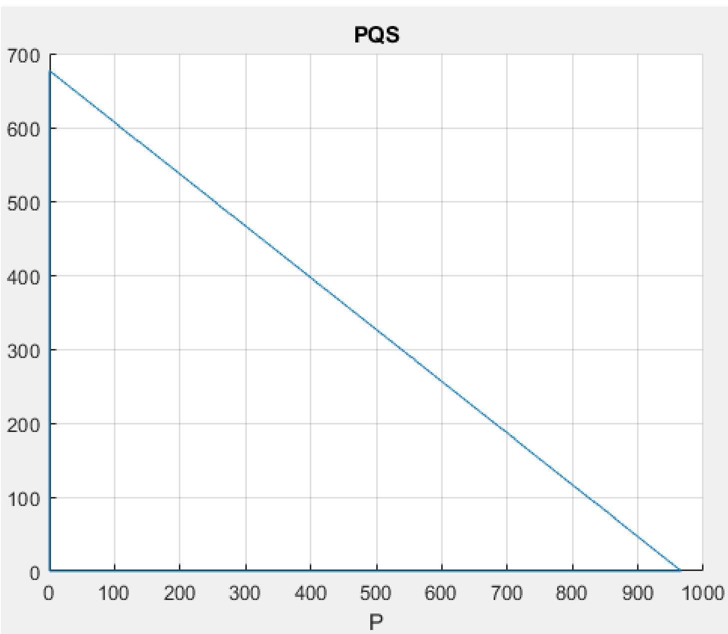
נקבל גרף:



באותה צורה ניתן לצייר משולש הספקים:

קלט:

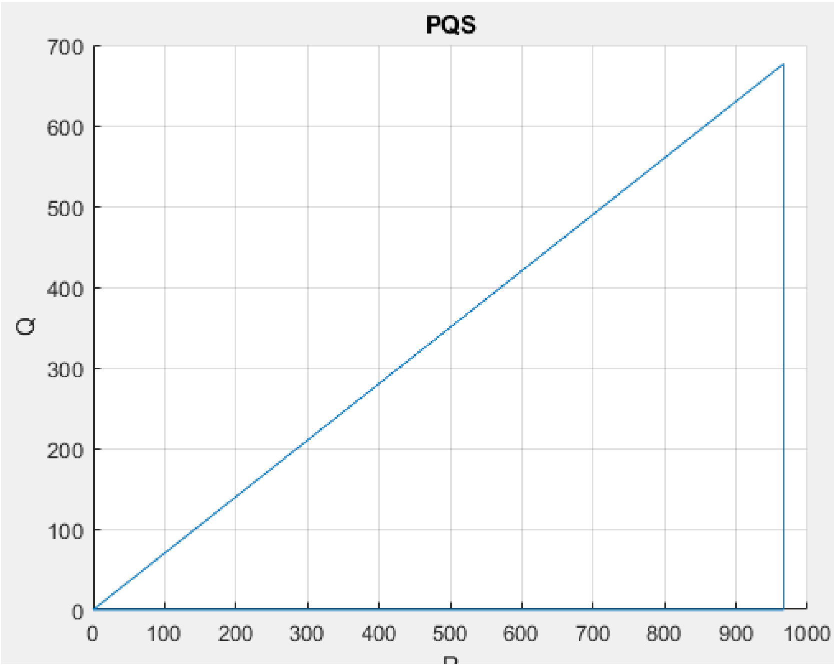
```
>> line([0 0],[0 676.5101])
line([0 966.443],[0 0 ])
line([0 966.433],[676.5101 0])
grid
xlabel('P')
ylabel('Q')
title('PQS')
```



או בצורה אחרת:

קלט:

```
>> line([0 966.443],[0 0 ])
line([966.443 966.443],[0 676.5101 ])
line([0 966.443],[0 676.5101 ])
grid
xlabel('P')
ylabel('Q')
title('PQS')
```


PQS
Q
P
0
100
200
300
400
500
600
700
800
900
1000

ב. פתרון של רשת חשמלית ליניארית .

<u>כתיבת פונקציה לפתירת רשת חשמלית בשיטת זרמי חוגים או מתחי צמתים</u>.

הרשת כוללת לפחות ארבעה חוגים (עבור פתרון בשיטת זרמי חוגים) או לפחות,

ארבעה צמתים (עבור פתרון בשיטת מתחי צמתים.)

א. הפתרון מבוצע בשיטת קרמר.

נניח ונתונה מטריצה של מקדמים של 3 חוגים:

$$\begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix}$$

ווקטור של איברים חופשיים

$$\begin{pmatrix} E_1 \\ E_2 \\ E_3 \end{pmatrix}$$

ווקטור של משתנים:

$$\begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix}$$

המשוואה הכללית נראית כך:

$$\begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \cdot \begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ E_3 \end{pmatrix}$$

$$\text{Det} = \begin{vmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{vmatrix}$$

$$\text{DetA} = \begin{vmatrix} i_1 & R_{12} & R_{13} \\ i_2 & R_{22} & R_{23} \\ i_3 & R_{32} & R_{33} \end{vmatrix}$$

$$\text{DetB} = \begin{vmatrix} R_{11} & i_1 & R_{13} \\ R_{21} & i_2 & R_{23} \\ R_{31} & i_3 & R_{33} \end{vmatrix}$$

$$\text{DetC} = \begin{vmatrix} R_{11} & R_{12} & i_1 \\ R_{21} & R_{22} & i_2 \\ R_{31} & R_{32} & i_3 \end{vmatrix}$$

i1=DetA/Det; i2=DetB/Det; i3=DetC/Det

במתמטיקה רגילה יש צורך לבדוק ש- Det שונה מ-0, אחרת יכול להיות מצב שאין פתרון או שאין סוף פתרונות.

בחשמל ואלקטרוניקה תמיד יש פתרון אחד ויחיד.

נתרגם את כל זה לשפת MATLAB:

קלט:

```
>> R=[8 -3 -3; -3 9 -2; -3 -2 6]
```

פלט:

```
R =
   8   -3   -3
  -3    9   -2
  -3   -2    6
```

קלט:

```
>> R1=[-1 -3 -3; -7 9 -2; 4 -2 6]
```

פלט:

```
R1 =
  -1   -3   -3
  -7    9   -2
   4   -2    6
```

קלט:

```
>> R2=[8 -1 -3; -3 -7 -2; -3 4 6]
```

פלט:

```
R2 =
   8   -1   -3
  -3   -7   -2
  -3    4    6
```

קלט:

```
>> R3=[8 -3 -1; -3 9 -7; -3 -2 4]
```

פלט:

```
R3 =
   8   -3   -1
  -3    9   -7
  -3   -2    4
```

קלט:

```
>> i1=det(R1)/det(R)
```

פלט:

```
i1 =
  -0.3755
```

קלט:

```
>> i2=det(R2)/det(R)
```

פלט:

```
i2 =
  -0.8603
```

קלט:

```
>> i3=det(R3)/det(R)
```

פלט:

```
i3 =
   0.1921
```

<u>הפתרון מבוצע באמצעות פעולות בסיסיות בין מטריצות ווקטורים.</u>

נניח ונתונה מטריצה של מקדמים של 3 חוגים:

$$\begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix}$$

ווקטור של איברים חופשיים

$$\begin{pmatrix} E_1 \\ E_2 \\ E_3 \end{pmatrix}$$

ווקטור של משתנים:

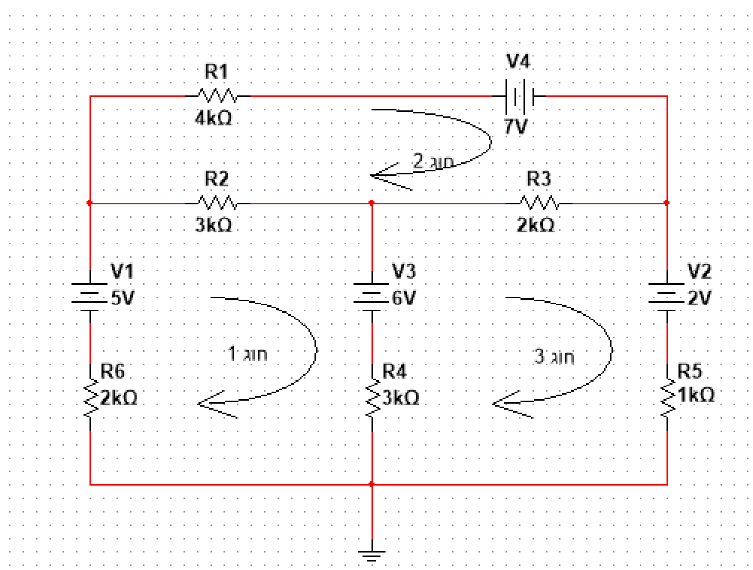
$$\begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix}$$

המשוואה הכללית נראית כך:

$$\begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \cdot \begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ E_3 \end{pmatrix}$$

כדי לפתור את המערכת הזאת אפשר להשתמש במטריצה הפוכה:

$$[R] \cdot [i] = [E] \rightarrow [R^{-1}] \cdot [R] \cdot [i] = [R^{-1}] \cdot [E] \rightarrow [i] = [R^{-1}] \cdot [E]$$



נרשום מטריצות:

$$\begin{pmatrix} 8k & -3k & -3k \\ -3k & 9k & -2k \\ -3k & -2k & 6k \end{pmatrix} \cdot \begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix} = \begin{pmatrix} -1 \\ -7 \\ 4 \end{pmatrix}$$

נעביר את המטריצות לMATLAB: (שימו לב שתוצאות הן במילי אמפר).

קלט:

```
>> R=[8 -3 -3; -3 9 -2; -3 -2 6]
```

פלט:

```
R =
    8   -3   -3
   -3    9   -2
   -3   -2    6
```

קלט:

```
>> E=[-1; -7; 4]
```

פלט:

```
E =
   -1
   -7
    4
```

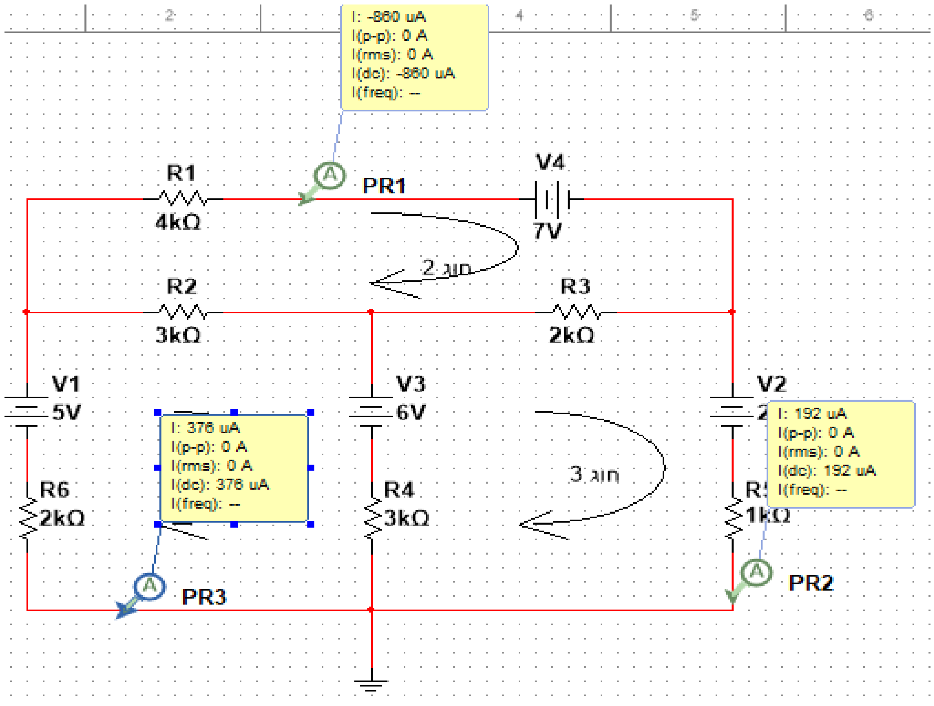
קלט:

```
>> I=R^-1*E
```

פלט:

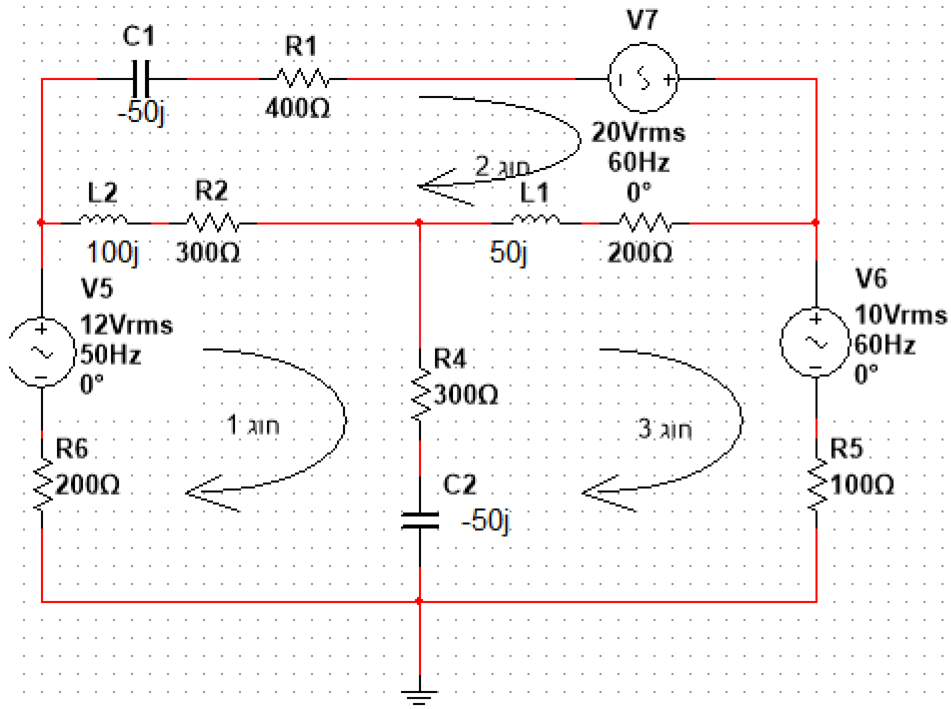
```
I =
  -0.3755
  -0.8603
   0.1921
```

נריץ סימולציה בתוכנת מולטיסים ונראה את התוצאה:



ג. פתרון רשת חשמלית AC.

נדגים איך לפתור מעגל AC בעזרת MATLAB שאי אפשר לפתור בעזרת מחשבון CASIO המצוי.



$$\begin{pmatrix} 800+50j & -300-100j & -300+50j \\ -300-100j & 900+100j & -200-50j \\ -300+50j & -200+50j & 600 \end{pmatrix} \cdot \begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix} = \begin{pmatrix} 12 \\ 20 \\ -10 \end{pmatrix}$$

קלט:

```
>> R=[800+50j -300-100j -300+50j; -300-100j 900+100j -200-50j; -300+50j -200+50j 600]
```

פלט:

```
R =
   1.0e+02 *
   8.0000 + 0.5000i  -3.0000 - 1.0000i  -3.0000 + 0.5000i
  -3.0000 - 1.0000i   9.0000 + 1.0000i  -2.0000 - 0.5000i
  -3.0000 + 0.5000i  -2.0000 + 0.5000i   6.0000 + 0.0000i
```

קלט:

```
>> E=[12; 20; -10]
```

פלט:

```
E =
    12
    20
   -10
```

קלט:

```
>> I=R^-1*E
```

פלט:

```
I =
  0.0323 - 0.0016i
  0.0358 - 0.0019i
  0.0111 - 0.0071i
```

פתרון מערכת משוואות בעזרת פונקציה SOLVE:

נרשום את המטריצות שכתבנו בצורת מערכת משוואות:

$$\begin{pmatrix} 8k & -3k & -3k \\ -3k & 9k & -2k \\ -3k & -2k & 6k \end{pmatrix} \cdot \begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix} = \begin{pmatrix} -1 \\ -7 \\ 4 \end{pmatrix}$$

$$\begin{cases} 8ki_1 - 3ki_2 - 3ki_3 = -1 \\ -3k\,i_1 + 9ki_2 - 2ki_3 = -7 \\ -3ki_1 - 2ki_2 + 6ki_3 = 4 \end{cases}$$

קלט:

```
>> syms i1 i2 i3;
>> [i1,i2,i3]=solve('8*i1-3*i2-3*i3=-1', '-3*i1+9*i2-2*i3=-7', '-3*i1-2*i2+6*i3=4');
>> vpa(i1,4)
```

פלט:

```
ans =
 -0.3755
```

קלט:

```
>> vpa(i2,4)
```

פלט:

```
ans =
 -0.8603
```

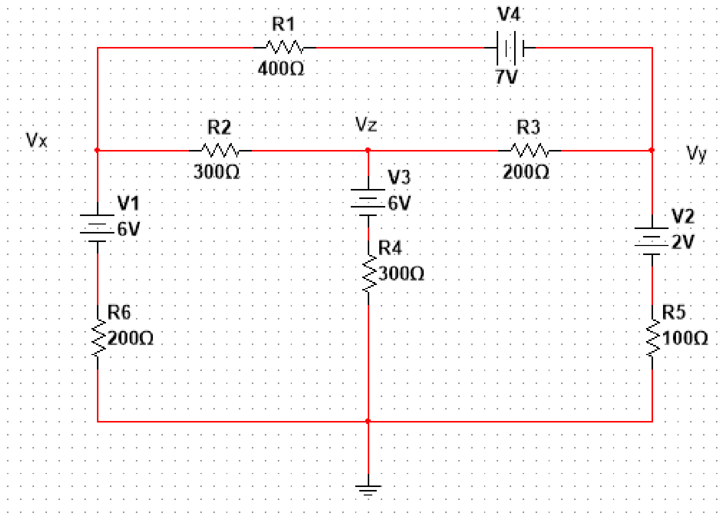
קלט:

```
>> vpa(i3,4)
```

פלט:

```
ans =
 0.1921
```

ד. פתרון מעגלים בשיטת מתחי צמתים:



נרשום מערכת משוואות המתאימה למעגל זה:

$$\begin{cases} \frac{Vx-6}{200}+\frac{Vx-Vz}{300}+\frac{Vx-7-Vy}{400}=0 \\ \frac{Vz-Vx}{300}+\frac{Vz-6}{300}+\frac{Vz-Vy}{200}=0 \\ \frac{Vy+7-Vx}{400}+\frac{Vy-Vz}{200}+\frac{Vy-2}{100}=0 \end{cases}$$

כדי להעביר את המערכת הזאת למערכת "קנונית" אפשר לבחור בשתי דרכים: למצוא מכנה משותף או לחלק כל מספר במכנה משלו.

נציג את שתי הדרכים:

$$1.\ \begin{cases} \frac{6Vx-36}{1200}+\frac{4Vx-4Vz}{1200}+\frac{3Vx-21-3Vy}{1200}=0 \\ \frac{2Vz-2Vx}{600}+\frac{2Vz-12}{600}+\frac{3Vz-3Vy}{600}=0 \\ \frac{Vy+7-Vx}{400}+\frac{2Vy-2Vz}{400}+\frac{4Vy-8}{400}=0 \end{cases} \rightarrow \begin{cases} 6Vx+4Vx-4Vz+3Vx-3Vy=57 \\ 2Vz-2Vx+2Vz+3Vz-3Vy=12 \\ Vy-Vx+2Vy-2Vz+4Vy=1 \end{cases}$$

$$\rightarrow \begin{cases} 13Vx-3Vy-4Vz=57 \\ -2Vx-3Vy+7Vz=12 \\ -Vx+7Vy-2Vz=1 \end{cases}$$

$$2.\ \begin{cases} \frac{Vx-6}{200}+\frac{Vx-Vz}{300}+\frac{Vx-7-Vy}{400}=0 \\ \frac{Vz-Vx}{300}+\frac{Vz-6}{300}+\frac{Vz-Vy}{200}=0 \\ \frac{Vy+7-Vx}{400}+\frac{Vy-Vz}{200}+\frac{Vy-2}{100}=0 \end{cases} \rightarrow \begin{cases} \frac{Vx}{200}-\frac{6}{200}+\frac{Vx}{300}-\frac{Vz}{300}+\frac{Vx}{400}-\frac{7}{400}-\frac{Vy}{400}=0 \\ \frac{Vz}{300}-\frac{Vx}{300}+\frac{Vz}{300}-\frac{6}{300}+\frac{Vz}{200}-\frac{Vy}{300}=0 \\ \frac{Vy}{400}+\frac{7}{400}-\frac{Vx}{400}+\frac{Vy}{200}-\frac{Vz}{200}+\frac{Vy}{100}-\frac{2}{100}=0 \end{cases} \rightarrow$$

$$\rightarrow \begin{cases} 0.0108Vx-0.0025Vy-0.0033Vz=0.0475 \\ -0.0033Vx-0.0033Vy+0.0117Vz=0.02 \\ 0.0025Vx+0.0175Vy-0.005Vz=0.0025 \end{cases}$$

נפתור את המערכת הזאת לפי שלושת השיטות:

- קרמר:

קלט:

```
>> R=[13 -3 -4; -2 -3 7; -1 7 -2]
```

פלט:

```
R =
        13   -3   -4
        -2   -3    7
        -1    7   -2
```

קלט:

```
>> R1=[57 -3 -4; 12 -3 7; 1 7 -2]
```

פלט:

```
R1 =
        57   -3   -4
        12   -3    7
         1    7   -2
```

קלט:

```
>> R2=[13 57 -4; -2 12 7; -1 1 -2]
```

פלט:

```
R2 =
        13   57   -4
        -2   12    7
        -1    1   -2
```

קלט:

```
>> R3=[13 -3 57; -2 -3 12; -1 7 1]
```

פלט:

```
R3 =
        13   -3   57
        -2   -3   12
        -1    7    1
```

קלט:

```
>> x=det(R1)/det(R)
```

פלט:

```
x =
   6.3144
```

קלט:

```
>> y=det(R2)/det(R)
```

פלט:

```
y =
    2.3362
```

קלט:

```
>> z=det(R3)/det(R)
```

פלט:

```
z =
4.5197
```

- מטריצות:

קלט:

```
>> R=[13 -3 -4; -2 -3 7; -1 7 -2]
```

פלט:

```
R =
   13   -3   -4
   -2   -3    7
   -1    7   -2
```

קלט:

```
>> V = [ 57 ;12; 1]
```

פלט:

```
V =
   57
   12
    1
```

קלט:

```
>> Vxyz=R^-1*V
```

פלט:

```
Vxyz =
   6.3144
   2.3362
   4.5197
```

- :SOLVE

קלט:

```
>> syms Vx Vy Vz;
>> [Vx, Vy, Vz] = solve('13*Vx - 3*Vy-4*Vz=57', '-2*Vx - 3*Vy+7*Vz=12', '-1*Vx +7*Vy-2*Vz=1');
>> vpa(Vx,4)
```

פלט:

```
ans =
6.314
```

קלט:

```
>> vpa(Vy,4)
```

פלט:

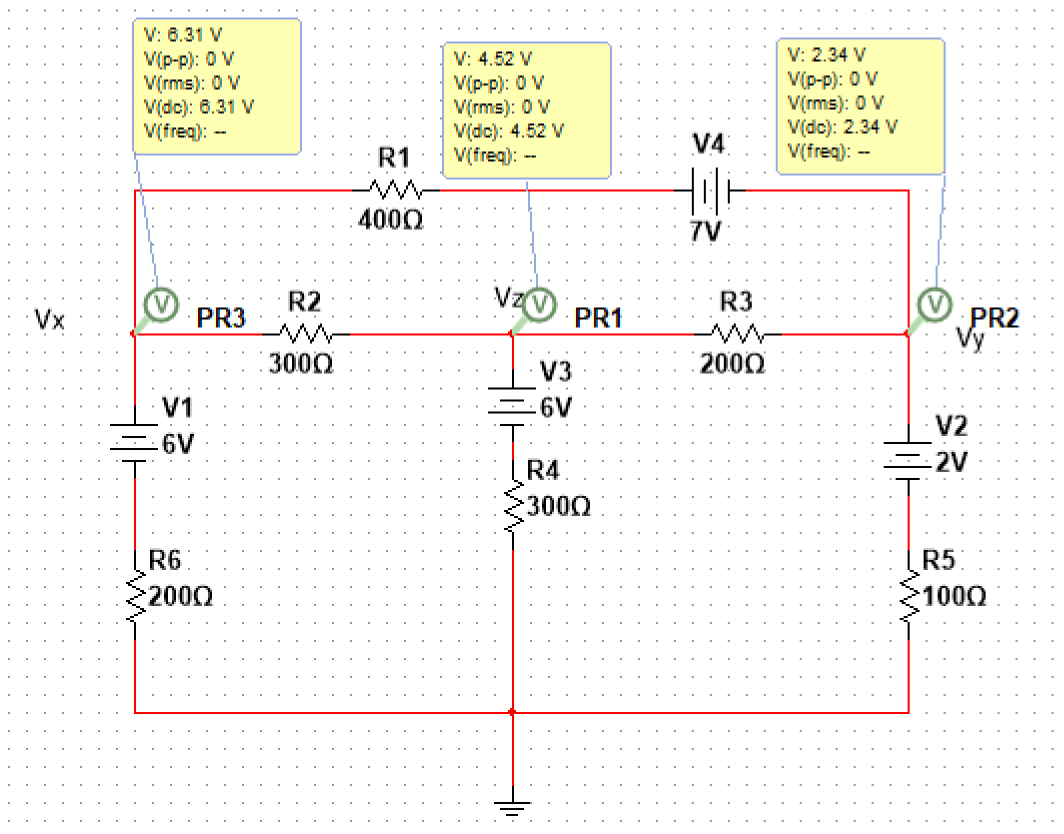
```
ans =
 2.336
```

קלט:

```
>> vpa(Vz,4)
```

פלט:

```
ans =
 4.52
```



תרגול:

1. מצא פתרון למערכת משוואות בעזרת שיטת קרמר, מטריצה הפוכה ובעזרת SOLVE:

$$1.\ \begin{cases} 2x_1 + x_2 + x_4 = 8; \\ x_1 - 3x_2 + 2x_3 + 4x_4 = 9; \\ -5x_1 - x_3 - 7x_4 = -5; \\ x_1 - 6x_2 + 2x_3 + 6x_4 = 0. \end{cases}$$

$$3.\ \begin{cases} 2x_1 + 3x_2 - 4x_3 + x_4 = 3{,}1; \\ 0{,}1x_1 - 2x_2 - 5x_3 + x_4 = 2; \\ 0{,}15x_1 - 3x_2 + x_3 - 4x_4 = 1; \\ 10x_1 + 2x_2 - x_3 + 2{,}1x_4 = -4{,}7. \end{cases}$$

$$2.\ \begin{cases} 3x_1 - x_2 = 5; \\ -2x_1 + x_2 - x_3 = 0; \\ 2x_1 - x_2 + 4x_3 = 15. \end{cases}$$

$$4.\ \begin{cases} 4x_1 + 0{,}24x_2 - 0{,}08x_3 = 8; \\ 0{,}09x_1 + 3x_2 - 0{,}15x_3 = 9; \\ 0{,}04x_1 - 0{,}08x_2 + 4x_3 = 20. \end{cases}$$

2. מצא פתרון של מערכת משוואות בעזרת שיטת קרמר, מטריצה הפוכה, בעזרת SOLVE ובצורה גרפית.

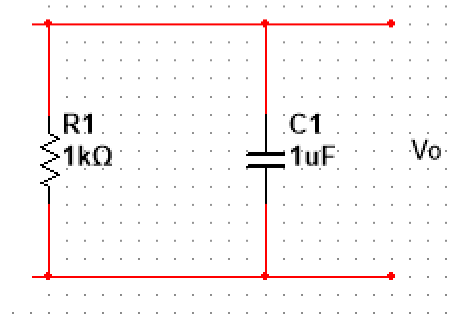
1. $\begin{cases} 2x - y = 4; \\ 3x + 2y = 3. \end{cases}$

2. $\begin{cases} 5x + 2y = 10; \\ 2x - y = 7. \end{cases}$

3. $\begin{cases} 2x + 3y = 1; \\ x + 2y = 4. \end{cases}$

4. $\begin{cases} 7x + 2y = 2; \\ x + 4y = 1. \end{cases}$

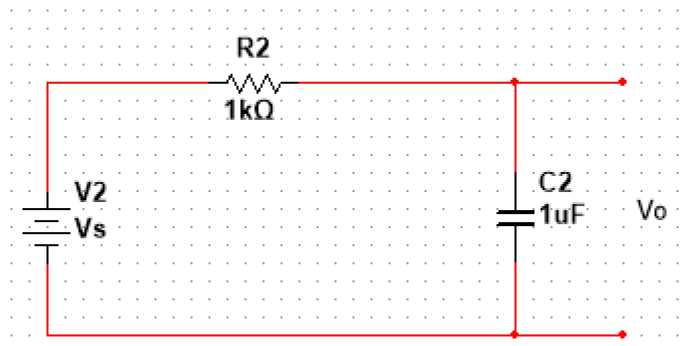
ה. מציאת התגובה של רשת חשמלית RC מקבילי ומעגל RC טורי:



לפי חוק קירכהוף לזרמים נכתוב את המשוואה הבאה:

$$C\frac{dVo(t)}{dt}+\frac{Vo(t)}{R}=0\ \rightarrow\frac{dVo(t)}{dt}+\frac{Vo(t)}{RC}=0\rightarrow Vo(t)=Vm\cdot e^{-\frac{t}{RC}}$$

Vm – מתח התחלתי על הקבל, RC – קבוע של מעגל RC.

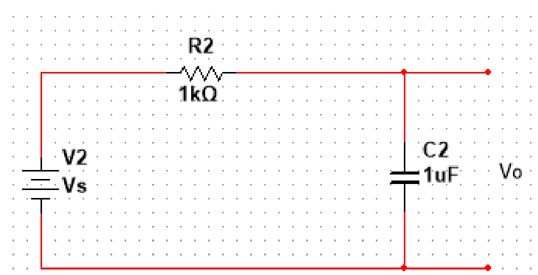


$C\frac{dVo(t)}{dt}+\frac{Vo(t)-Vs}{R}=0$. במידה ולפני הפעלת המעגל מתח ביציאה שווה ל-0 המשוואה תהיה:

$$Vo(t)=Vs\cdot\left(1-e^{-\frac{t}{RC}}\right)$$

דוגמא 1.

נתון מעגל RC .



נניח וקבל ניטען ע"י מקור מתח של 10V כאשר קיבול הקבל הינו 10 מיקרו פרד. צייר גרף של טעינת הקבל במקרה שערך הנגד יהיה 5 קילו אוהם, 10 קילו אוהם, ו-20 קילו אוהם.

נגדיר את מתח המקור

נגדיר את ערך הקבל
נגדיר טווח זמנים של הגרף

R1 נחשב קבוע זמן
נציב הכול לנוסחה של מתח על הקבל

R2 נחשב קבוע זמן
נציב הכול לנוסחה של מתח על הקבל

R3 נחשב קבוע זמן
נציב הכול לנוסחה של מתח על הקבל

נבנה גרפים של מתחים כתלות בזמן

קלט:

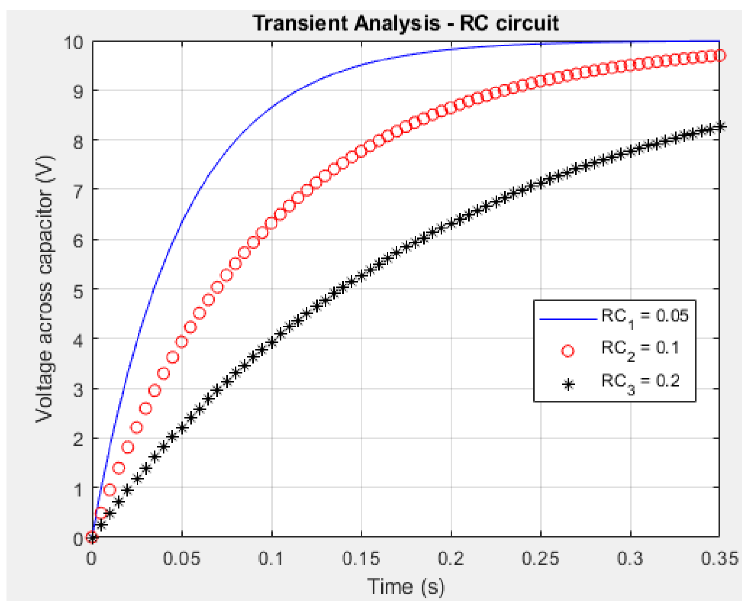
```
>> Vs = 10;
C = 10e-6;
t = 0 : 0.005 : 0.35;

R1 = 5e3;
tau1 = R1*C;
V1 = Vs * ( 1 - exp(-t/tau1) );

R2 = 10e3;
tau2 = R2*C;
V2 = Vs * ( 1 - exp(-t/tau2) );

R3 = 20e3;
tau3 = R3*C;
V3 = Vs * ( 1 - exp(-t/tau3) );

plot(t, V1, 'b-', t, V2, 'ro', t, V3, 'k*')
grid on
title('Transient Analysis - RC circuit')
xlabel('Time (s)')
ylabel('Voltage across capacitor (V)')
legend(['RC_1 = ' num2str(tau1)],...
     ['RC_2 = ' num2str(tau2)],...
     ['RC_3 = ' num2str(tau3)], 'location', 'best')
```



דוגמא 2:

באותו מעגל ניתן מתח כניסה בתור מדרגה בגובה 10V במשך 0.5 שניות, כאשר קיבול של הקבל הינו 10 מיקרו פרד.

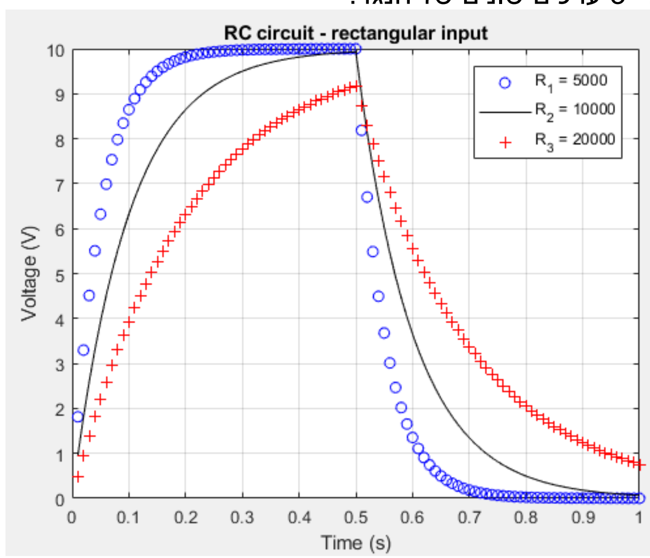
צייר את גרף טעינת הקבל במקרה שערך הנגד יהיה 5 קילו אוהם, 10 קילו אוהם ו-20 קילו אוהם.

נבנה פונקציה שמחשבת מתח יציאה בהתאם למתח כניסה, ערך הנגד וערך הקבל ונשמור אותה בנפרד. פונקציה מחזירה שני ווקטורים – ווקטור של זמן טעינה ופריקה ווקטור של מתחים בפריקה וטעינה.

קלט:

```
function [v, t] = rectangular_RC(vs, r, c)
tau = r * c;
t1 = linspace(.01, 0.5, 50);
v1 = vs * (1 - exp(-t1/tau));
Vm = v1(end);
t2 = linspace(0.51, 1, 50);
v2 = Vm * exp(-t1/tau);
t = [t1 t2];
v = [v1 v2];
end
```

נשתמש בפונקציה הזאת כדי לחשב טעינה ופריקה של קבל ל- 3 ערכים שונים של הנגד:



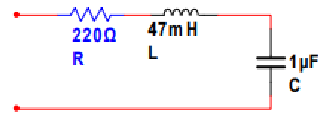
קלט:

```
vs = 10;
c = 10e-6;

r1 = 5e3;
[v1, t1] = rectangular_RC(vs, r1, c);
r2 = 10e3;
[v2, t2] = rectangular_RC(vs, r2, c);
r3 = 20e3;
[v3, t3] = rectangular_RC(vs, r3, c);
grid on
plot(t1, v1, 'bo', t2, v2,'k-',t3,v3, 'r+')
title('RC circuit - rectangular input')
xlabel('Time (s)')
ylabel('Voltage (V)')
legend(['R_1 = ' num2str(r1)], ...
      ['R_2 = ' num2str(r2)],...
      ['R_3 = ' num2str(r3)])
```

ו. תגובה של מעגל RLC לאות מדרגה בכניסה.

נתון מעגל:



ניקח את מתח המקור בתור Vin ומתח היציאה Vo שווה למתח על הקבל Vc. i – זה הזרם שזורם דרך מעגל .

$$Vo = Vc = \frac{1}{C}\int i\,dt \rightarrow i = C\frac{dVo}{dt}$$

מתח על הנגד ניתן לחשב לפי נוסחה: $V_R = Ri = RC\frac{dVo}{dt}$

מתח על הסליל שווה : $V_L = L\frac{di}{dt} = LC\frac{d}{dt}\left(\frac{dVo}{dt}\right) = LC\frac{d^2Vo)}{dt^2}$

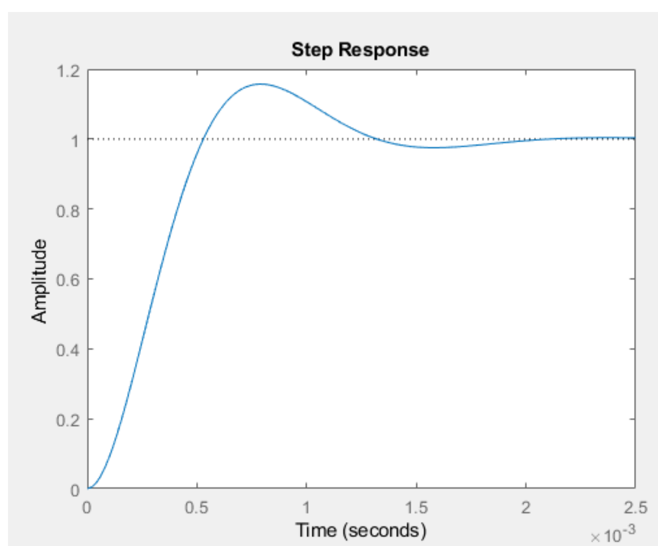
$$Vin(t) = RC\frac{dVo}{dt} + LC\frac{d^2Vo)}{dt^2} + Vo(t)$$

נפתור את המשוואה בעזרת התמרת לפלס, כאשר מצב התחלתי שווה ל-0.

$$Vin(s) = sRCVo(s) + s^2LCVo(s) + Vo(s) = Vo(s)(sRC + s^2LC + 1)$$

$$\frac{Vin(s)}{Vo(s)} = \frac{1}{s^2LC + sRC + 1}$$

נגדיר ערך של סליל
נגדיר את הערך של הקבל
נגדיר את הערך של הנגד
כדי לפתור משוואה לפלס יש להגדיר מונה ומקדמים של מכנה
נגדיר פונקציה G בתור התמרת לפלס עם מונה ומכנה
נפעיל פונקציה step על G ונקבל את הגרף הבא:



קלט:

```
>> L=0.047
C=0.000001
R=220
Num=1
Denum=[L*C R*C 1]
G=tf(Num,Denum)
step(G)
```

פלט:

```
L =
   0.0470
C =
  1.0000e-06
R =
  220
Num =
   1
Denum =
   0.0000    0.0002    1.0000
G =            1
  --------------------------
  4.7e-08 s^2 + 0.00022 s + 1
 Continuous-time transfer function.
```

- פתרון יותר מדויק למתקדמים:
כאשר אנו מנסים לפתור משוואה ולמצוא שורשים של S:

$$\frac{\mathrm{Vin(s)}}{\mathrm{Vo(s)}} = \frac{1}{s^2 LC + sRC + 1}$$

$$s_{1,2} = \frac{-RC \pm \sqrt{(RC)^2 - 4 \cdot LC \cdot 1}}{2 \cdot LC}$$

לפתרון המשוואה הזאת (כמו לכל משוואה ריבועית) קיימים שלושה סוגי פתרונות: שני פתרונות ממשים ($(RC)^2 - 4 \cdot LC \cdot 1 > 0$, פתרון אחד ($(RC)^2 - 4 \cdot LC \cdot 1 = 0$), שני שורשים מרוכבים ($(RC)^2 - 4 \cdot LC \cdot 1 < 0$).

באנגלית המקרים האלה נקראים: overdamped , critically damped ו- underdamped בהתאמה.

ישנו פתרון מסוים לכל מקרה:

Overdamped $\mathrm{Vo(t)} = \mathrm{Vin} + A1e^{s1t} + A2e^{s2t}$ עודף ריסון

critically damped $\mathrm{Vo(t)} = \mathrm{Vin} + (A1 + A2t)e^{-\alpha t}$ ריסון קריטי (סף יציבות)

underdamped תת-ריסון $\mathrm{Vo(t)} = \mathrm{Vin} + (A1\cos\omega t + A2\sin\omega t)e^{-\alpha t}$, כאשר: $\alpha = \frac{r}{2L}, \omega = \frac{1}{\sqrt{LC}}$, ערכים A_1, A_2 קבועים שנתונים לפי תנאי התחלה. לדוגמא לגל מדרגה $\mathrm{Vo(0)} = 0, \frac{d\mathrm{Vo(0)}}{dt} = 0$.

נניח והערכים של הקבל והסליל ידועים לנו וערך הנגד משתנה כל פעם, כך שנקבל את כל שלושת המצבים. נניח שערך הסליל 10 מילי הנרי, קבל 1 מיקרו פרד, מתח המקור 1V. נפתור את המשוואה: $(RC)^2 - 4 \cdot LC = 0$ ונקבל: $R = \frac{2\sqrt{L}}{\sqrt{C}} = 200\ \mathrm{Ohm}$, נבחר ערכים של נגדים פי 5 יותר גבוה מהערך הקריטי ופי 5 יותר קטן מהערך קריטי: R_u=40 Ohm, R_c=200 Ohm, R_o=1 K.

<u>מקרה ראשון:</u> עודף ריסון – ערכים של נגד R_o=1 KOhm , ערך הסליל 10 מילי הנרי, קבל 1 מיקרו פרד, מתח המקור 1V.

בהצבה למשוואה

$$s_{1,2} = \frac{-RC \pm \sqrt{(RC)^2 - 4 \cdot LC \cdot 1}}{2 \cdot LC}$$

נקבל ערכים של s_1=-1010.2, s_2=-98990. פתרון של המשוואה $\mathrm{Vo(t)} = 1 + A1e^{-1010.2t} + A2e^{-98990t}$, במידה וערכים ההתחלתיים יהיו: t=0, Vo=0, יתקבל הפתרון הבא: A_1+A_2=-1. נעשה נגזרת נקבל:

$\mathrm{Vo'(t)} = -1010.2A1e^{-1010.2t} - 98990\ A2e^{-98990t}$, בהצבה שנגזרת של מתח יציאה שווה ל-0 וזמן יהיה שווה ל-0, נקבל:

$$0 = -1010.2A1 - 98990\ A2$$

פתרון של שני משוואות אלו ייתן לנו ערכים של A1 ו– A2 : A1 = -0.989898 A2 = -0.010102

עקב כך הפתרון הסופי יהיה: $\mathrm{Vo(t)} = 1 - 0.9898981e^{-1010.2t} - 0.010102e^{-98990t}$

<u>מקרה שני:</u> ריסון קריטי. ערכים של נגד R_o=200 Ohm , ערך הסליל 10 מילי הנרי, קבל 1 מיקרו פרד, מתח המקור 1V.

$$s_{1,} = s_2 = \frac{-RC}{2 \cdot LC} = -\frac{R}{2L} = -10\,000$$

המשוואה תיראה בצורה הבאה:

$$\mathrm{Vo(t)} = 1 + (A_1 + A_2 t)e^{-10\,000t}$$

תנאי התחלה בזמן t=0, Vo=0

במקרה זה הפתרון ייראה כך: A_1=-1

כדי לקבל מקדם A2 יש לגזור את המשוואה ולהציב תנאי התחלה:

$$Vo(t) = 1 + (A_1 + A_2 t)e^{-10\,000t} \rightarrow \frac{dVo}{dt} = A_2 \cdot e^{-10\,000t} - 10\,000(A_1 + A_2 t)e^{-10\,000t}$$

בהצבה של $\frac{dVo}{dt} = 0$ בזמן t=0, נקבל A_2-10 000A_1=0 או בהצבה של A_1=-1 נקבל ש- A_2=-10 000.

בסופו של דבר נקבל את המשוואה:

$$Vo(t) = 1 - (1 + 10\,000t)e^{-10\,000t}$$

<u>מקרה שלישי:</u> תת ריסון.

ערכים של נגד R_o=40 Ohm , ערך הסליל 10 מילי הנרי, קבל 1 מיקרו פרד, מתח המקור 1V. פתרון המשוואה ייתן לנו שני שורשים: $s_{1,2} = \alpha \pm j\omega \rightarrow \alpha = \frac{R}{2L} = 2000, \omega = \frac{1}{\sqrt{LC}} = 10\,000$. נכתוב משוואה בהתאם לנתונים אלו:

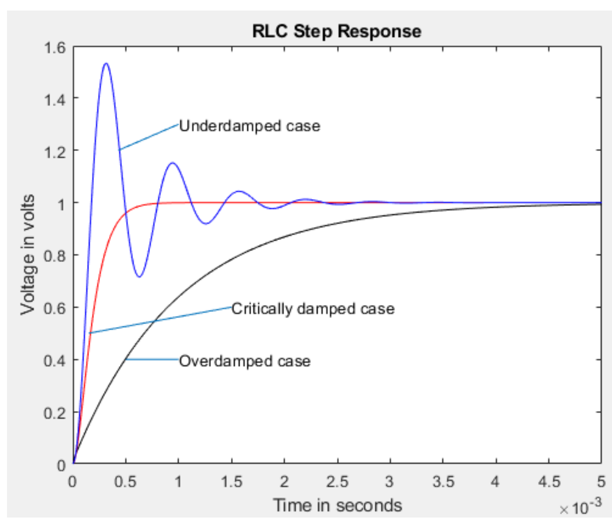
$$Vo(t) = Vin + (A1 \cos \omega t + A2 \sin wt)e^{-\alpha t} \text{ if } t = 0, Vo(t) = 0 \rightarrow A_1 = -1$$

בהצבה של $\frac{dVo}{dt} = 0$ בזמן t=0, נקבל -2000 A_1+10 000 A_2 = 0 או A_2 = -0.2, מכאן נקבל את המשוואה:

$$Vo(t) = 1 - (\cos 10\,000t + 0{,}2 \sin 10\,000t)e^{-2000t}$$

במקום הגדרת הווקטור אפשר גם להשתמש בלולאת for: (שימו לב שפעולת VECTROR יותר מהירה מפעולת FOR.

```
for i=1:length(t)
vc(i) = 1 - .989898*exp(-1010.2*t(i)) -
.010102*exp(-98990*t(i));
end
```



קלט:

```
>> t = 0:.00001:.005;
%Overdamped - R = 1K, L = 10mh, C = 1uf
vc = 1 - .989898*exp(-1010.2*t) - .010102*exp(-
98990*t);
figure(1);clf;
plot(t, vc, 'k'); %plot the overdamped case in black
%Turn on hold so all plots go to figure 1
hold on;
%Critically damped - R = 200, l = 10mh, C = 1uf
vc = 1 - (1 + 10000*t).*exp(-10000*t);
plot(t, vc, 'r'); %plot the critically damped case in red
%Underdamped - R = 40, L = 10mh, C = 1uf
w = 10000;
vc = 1 - (cos(w*t) + .2*sin(w*t)).*exp(-2000*t);
plot(t, vc, 'blue'); %plot the underdamped case in blue
%Add title and x and y axis labels
title('RLC Step Response');
xlabel('Time in seconds');
ylabel('Voltage in volts');
%Add text to mark the cases and draw a line from the
% text to each graph
text(.001, 1.3, 'Underdamped case');
x = [.001 .00043]; y = [1.3 1.2];
line(x, y);
text(.001, .4, 'Overdamped case');
x = [.001 .0005]; y = [.4 .4];
line(x, y);
text(.0015, .6, 'Critically damped case');
x = [.0015 .00015]; y = [.6 .5];
line(x, y);
```

פתרון נוסף לאותו מעגל:

ניקח את המעגל RLC טורי עם הערכים הבאים: נגד 10 אוהם, סליל 10 מילי הנרי, קבל 1 ננו פרד.

מתח על הסליל נוכל לחשב לפי משוואה: $V_L = L\frac{di}{dt}$, מצד שני ניתן לחשב אותו לפי חוק קירכהוף למתחים: $L\frac{di}{dt} = V - V_R - V_C$.

או $\frac{di}{dt} = \frac{V - R\cdot i - Vc}{L}$. באותה שיטה ניתן לכתוב מהו זרם הטעינה של קבל (שזה אותו זרם במעגל) $i = C\frac{dVc}{dt}$ או במילים אחרות: $\frac{dVc}{dt} = \frac{i}{C}$

נשלב 2 משוואות של זרם ומתח במערכת משוואות ונרשום אותם בצורת מטריצה:

$$\frac{d}{dt}\begin{bmatrix} i \\ Vc \end{bmatrix} = \begin{bmatrix} -R/L & -1/L \\ 1/C & 0 \end{bmatrix} \cdot \begin{bmatrix} i \\ Vc \end{bmatrix} + \begin{bmatrix} 1/L \\ 0 \end{bmatrix} V$$

משוואה הזאת ניתן לרשום בצורה הבאה: $\begin{cases} x = Ax + Bu \\ y = Cx + Du \end{cases}$,

כאשר $x = \begin{bmatrix} i \\ Vc \end{bmatrix}, A = \begin{bmatrix} -R/L & -1/L \\ 1/C & 0 \end{bmatrix}, B = \begin{bmatrix} 1/L \\ 0 \end{bmatrix}, y = \begin{bmatrix} i \\ 0 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \\ R & 0 \end{bmatrix}, D = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

ניתן להשתמש בפונקציה SS לפתרון משוואות דיפרנציאליות מסוג:

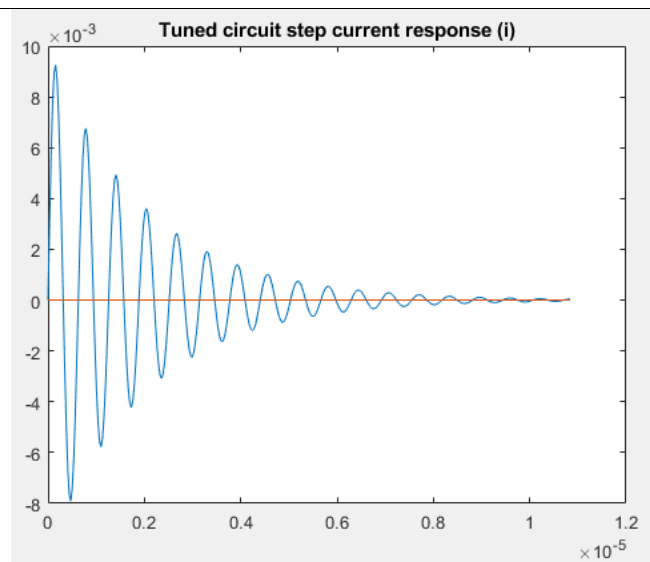
dx/dt = Ax(t) + Bu(t)

y(t) = Cx(t) + Du(t)

פונקציה STEP עובדת על מערכת דינמית.

קלט:

```
>>R = 10;
L=10e-6;
C=1e-9;
A = [-R/L -1/L; 1/C 0];
B = [1/L; 0];
C = [1 0; 0 0];
D = [0; 0];
system = ss(A,B,C,D);
[y,t] = step(system);
plot(t, y);
title('Tuned circuit step current response (i)');
```



פרק 8. אלקטרוניקה בעזרת MATLAB.

א. חקירת דיודה.

מציאת נקודת העבודה במעגל הכולל מקור מתח לזרם ישר, דיודת סיליקון ונגד המחוברים בטור, כאשר הדיודה נתונה באמצעות המודל המדויק.

את נקודת העבודה ניתן למצוא בצורה גרפית או באמצעות חישוב נומרי.

דיודה:

נתחיל בחישובים בסיסיים של דיודה מעשית.

נוסחה של זרם כתלות במתח בדיודה מעשית נתונה בצורה הבאה: $\mathrm{If} = \mathrm{Io}\left(e^{\frac{\mathrm{Vd}}{\eta \mathrm{Vt}}} - 1\right)$, כאשר:

If - זרם קדמי דרך דיודה

Vd - מתח שנופל על הדיודה, כאשר הדיודה מוליכה.

η – מקדם החומר, מספר שלם שתלוי בסוג החומר (גרמניום – 1, סיליקון – 2).

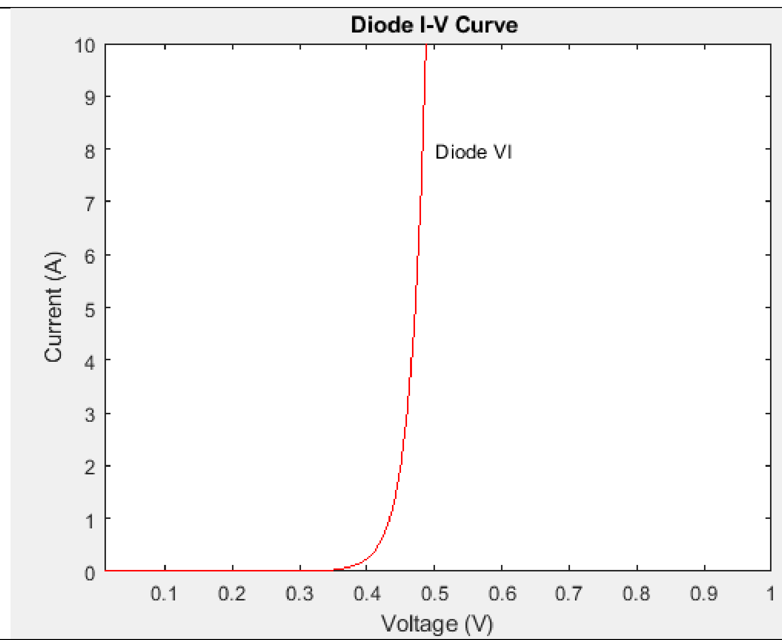
Vt – מתח טרמי. ניתן לחשב אותו לפי הנוסחה - $\mathrm{V_T} = \frac{\mathrm{kT}}{\mathrm{q}}$, $\mathrm{k} = 1.38 \cdot 10^{-34}\ \mathrm{J/°K}, \mathrm{q} = 1.6 \cdot 10^{-19}\mathrm{C}, \mathrm{T} = \mathrm{tempr}\ °\mathrm{K}$

Io – זרם זליגה. זרם זליגה תלוי בטמפרטורת הסביבה. כדי לחשב זרם זליגה בטמפרטורה מסוימת, צריכים לדעת את זרם הזליגה בטמפרטורה אחרת ואז ניתן לחשב זרם זליגה חדש בטמפרטורה חדשה לפי הנוסחה: $\mathrm{Io}_{(\mathrm{T2})} = \mathrm{Io}_{(\mathrm{T1})} \cdot 2^{\frac{\mathrm{T2}-\mathrm{T1}}{10}}$

דוגמא לחישוב זרמים דרך דיודת סיליקון כתלות במתח בטמפרטורת החדר, כאשר זרם זליגה 10 ננו אמפר.

קלט:

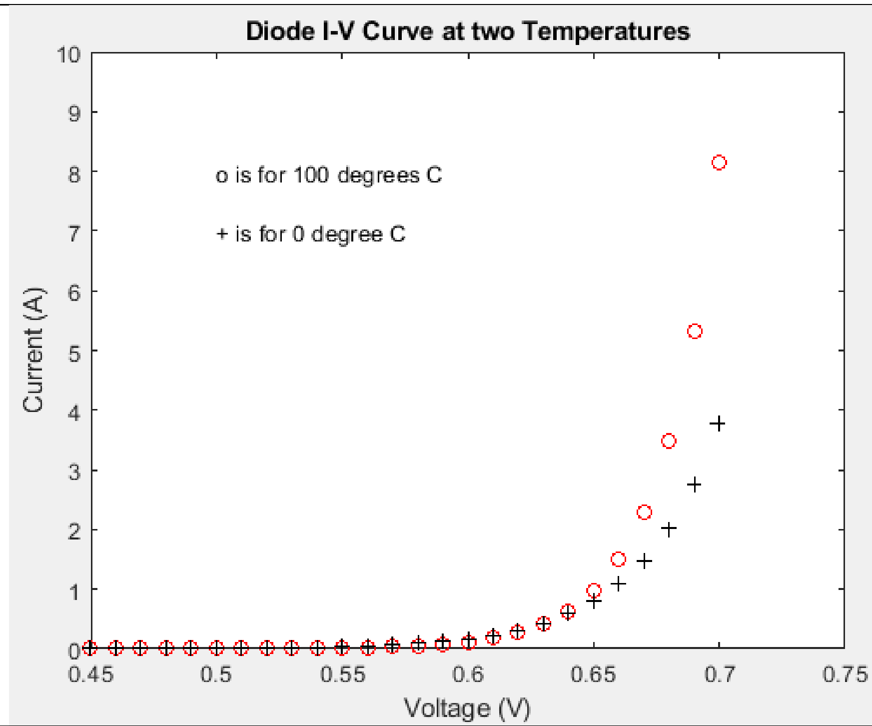
```
k = 1.38e-23; q = 1.6e-19;
t = 300;
I0 = 10e-9;
ks = 0.072;
v = 0.01:0.01:1;
I1 = Is1*exp(q*v/(k*t1));
plot(v,I1,'r-')
axis([0.01,1,0,10])
title('Diode I-V Curve ')
xlabel('Voltage (V)')
ylabel('Current (A)')
text(0.5,8,'Diode VI')
```



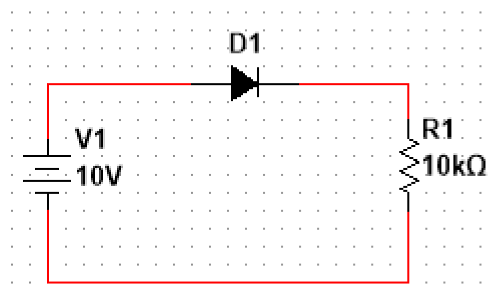
דוגמא לבדיקת השפעה של טמפרטורה, כאשר זרם זליגה של דיודות 1 פיקו אמפר ב-25 מעלות צלסיוס, עם מקדם החומר 1.9 . טמפרטורה של הדיודה הראשונה 0 מעלות צלסיוס ושל הדיודה השנייה 100 מעלות צלסיוס.

קלט:

```
k = 1.38e-23; q = 1.6e-19;
t1 = 273 + 0;
t2 = 273 + 100;
Is1 = 1.0e-12;
ks = 0.072;
Is2 = Is1*exp(ks*(t2-t1));
v = 0.45:0.01:0.7;
I1 = Is1*exp(q*v/(k*t1));
I2 = Is2*exp(q*v/(k*t2));
plot(v,I1,'ro',v,I2,'k+')
axis([0.45,0.75,0,10])
title('Diode I-V Curve at two
Temperatures')
xlabel('Voltage (V)')
ylabel('Current (A)')
text(0.5,8,'o is for 100 degrees C')
text(0.5,7, '+ is for 0 degree C')
```



נתון מעגל טורי עם דיודה ונגד:



זרם זליגה 1 פיקו אמפר, מקדם החומר שווה ל-2, טמפרטורת הסביבה 25 מעלות צלסיוס.

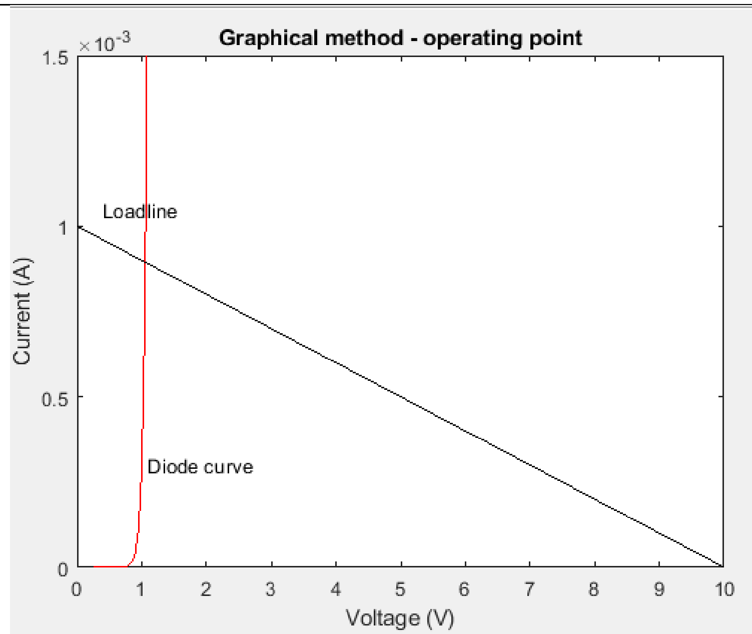
כדי למצוא נקודת עבודה יש 2 אפשרויות: פתרון גרפי ופתרון אלגברי:

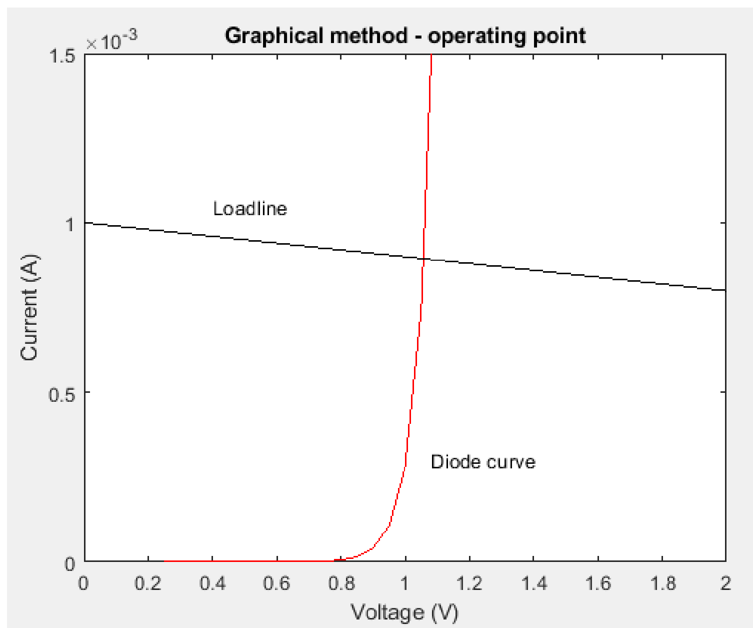
פתרון גרפי:

נכתוב משוואה של מעגל לפי חוק קירכהוף למתחים: $V_{DC} = RI_D + V_D$

:קלט

```
>> % diode equation
k = 1.38e-23;q = 1.6e-19;
t1 = 273 + 25; vt = k*t1/q;
v1 = 0.25:0.05:1.1;
i1 = 1.0e-12*exp(v1/(2.0*vt));
% load line 10=(1.0e4)i2 + v2
vdc = 10;
r = 10.0e3;
v2 = 0:2:10;
i2 = (vdc - v2)/r;
% plot
plot(v1,i1,'r', v2,i2,'k')
axis([0,10, 0, 0.0015])
title('Graphical method - operating point')
xlabel('Voltage (V)')
ylabel('Current (A)')
text(0.4,1.05e-3,'Loadline')
text(1.08,0.3e-3,'Diode curve')
```





ב. מיישר חצי גל:

נתון מתח מקור $V_s(t)=18\sin(120\pi t)$ V שמטעין סוללה עם מתח רצוי 11.8 V, דרך נגד 100 אוהם ודיודה אידיאלית.

בעזרת MATLAB יש למצוא את הנתונים הבאים:

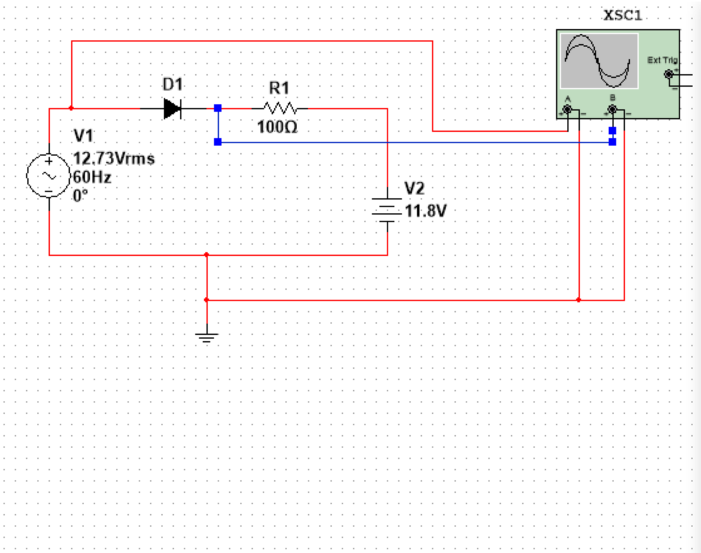
א. גרף של מתח כניסה.
ב. גרף של זרם דרך דיודה.
ג. חישוב זווית הולכה הדיודה.
ד. חישוב מתח השיא.

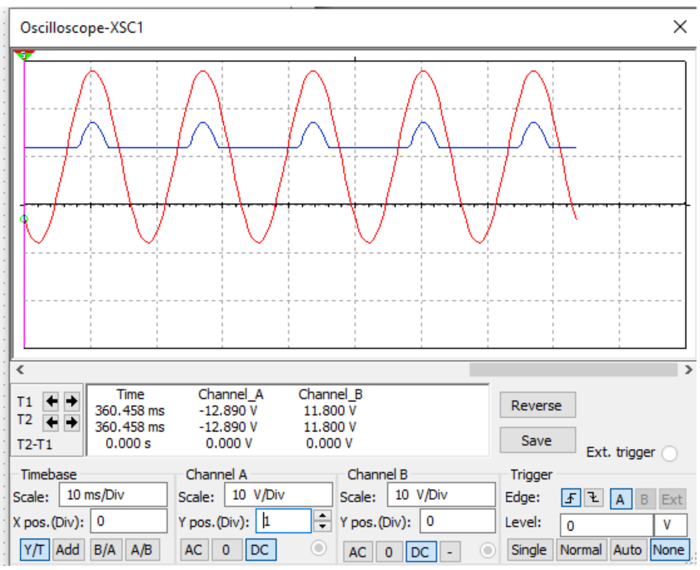
$$i_d = \frac{V_s - V_B}{R}$$

דיודה מתחילה להוליך כאשר Vs >= V_B ומסיימת להוליך כאשר Vs <= V_B.

$$18\sin(120\pi t_1) = 11.8V \quad 18\sin(120\pi t_2) = 11.8V$$

עקב סימטריות אפשר לרשום: $\theta_2 = \pi - \theta_1$





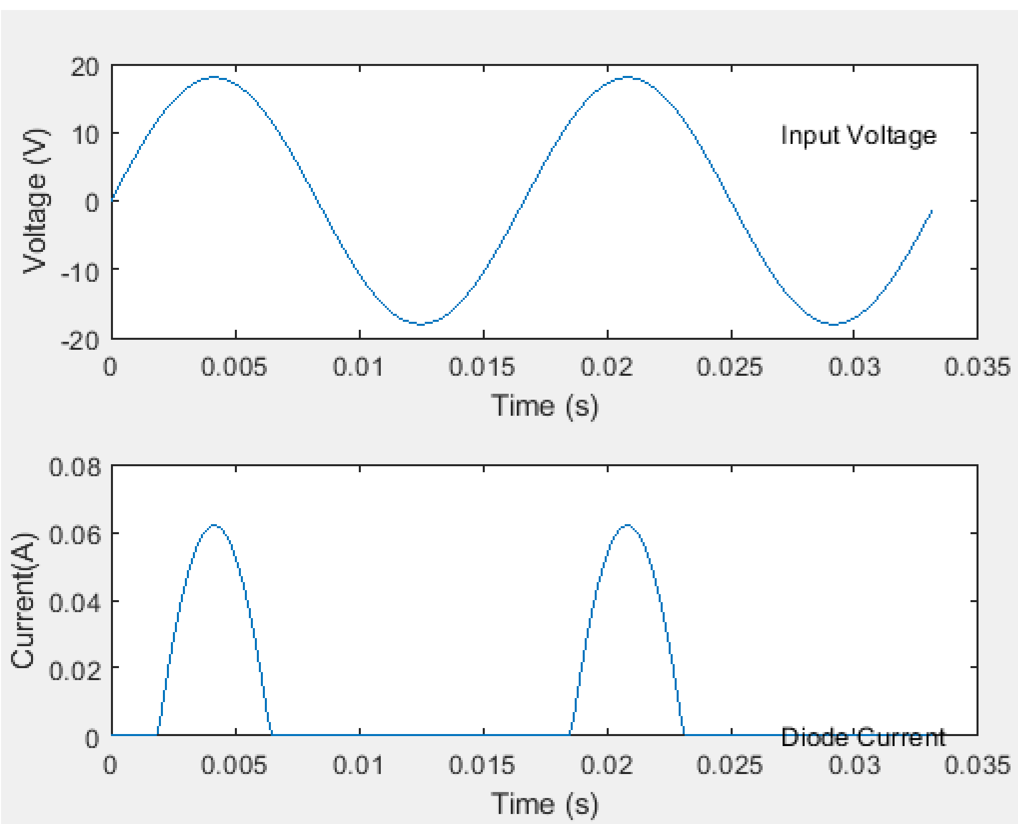
קלט:

```
>> % Baltery charging circuit
r=100;
period = 1/60;
 period2 = period*2;
inc =period/100;
npts = period2/inc;
vb = 11.8;
t = [];
for i = 1:npts
 t(i) = (i-1)*inc;
 vs(i) = 18*sin(120*pi*t(i));
 if vs(i) > vb
 idiode(i) = (vs(i) -vb)/r;
else
 idiode(i) = 0;
 end
end
```

```
subplot(211), plot(t,vs)
%title('Input Voltage')
xlabel('Time (s)')
ylabel('Voltage (V)')
text(0.027,10, 'Input Voltage')
subplot(212), plot(t,idiode)
%title('Diode Current')
xlabel('Time (s)')
ylabel('Current(A)')
text(0.027, 0.7e-3, 'Diode Current')
% conduction angle
theta1 = asin(vb/18); theta2 = pi - theta1;
acond = (theta2 -theta1)/(2*pi)
% peak current
pcurrent = (18*sin(pi/2) - vb)/r
% pcurrent = max(idiode)
acond_deg=acond/pi*180
```

פלט:

```
acond =
   0.2724
pcurrent =
   0.0620
acond_deg =
  15.6093
```



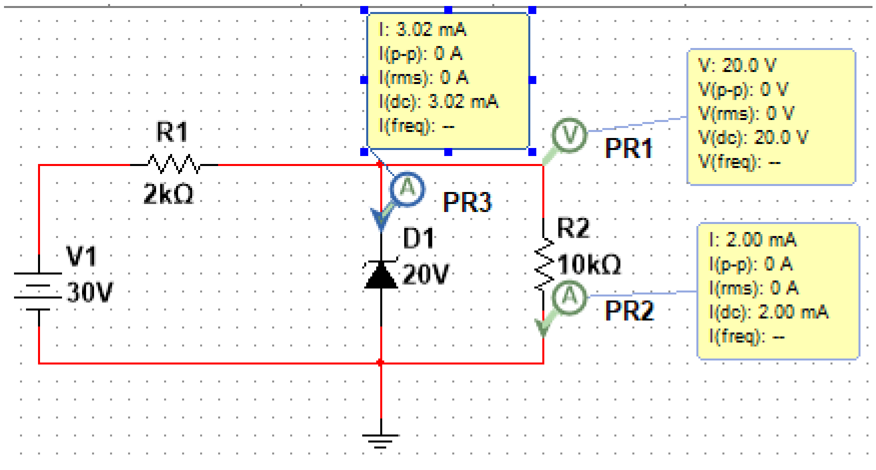
ג. דיודת זנר.

נתונה דיודת זנר עם מתח פריצה של 20V ($V_Z = -20 + 0.005I$ לזרמים: $-100\ mA < I < 0$).

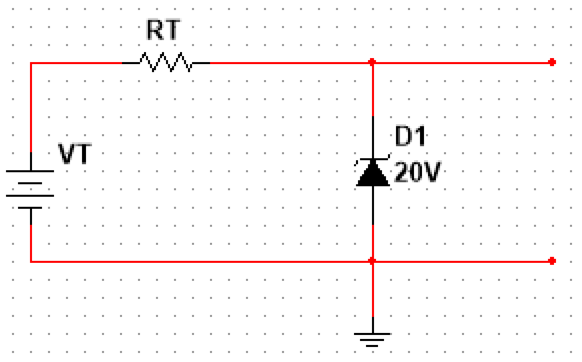
מתח מקור נע בין : $30 < V_S < 35\ V$. נגד העומס $R_L = 10K$, נגד מקור $R_S = 2K$.

נשתמש בתוכנת MATLAB כדי להציג את הנתונים הבאים:

א. גרף של דיודת זנר.
ב. גרפים של מתח יציאה למתחי כניסה 35V ו- 30V.
ג. ערכים של מתח יציאה למתחי כניסה 35V ו- 30V.



נהפוך את המעגל למעגל פשוט יותר בעזרת שיטת תבנין:



כדי לחשב מקור מתח תבנין $V_T = \frac{V_S \cdot R_L}{R_L + R_S}$, $R_T = R_L \,||\, R_S$.

$R_T = 10\ K \,||\, 2\ K = 1.67\ K\ Ohm$, $V_S = 30\ V \rightarrow V_T = 25\ V$, $V_S = 35\ V \rightarrow V_T = 29.17\ V$

משוואת הקו: $V_T = R_T I + V_Z$, מכאן נובע $V_Z = V_T - R_T I$ ומצד שני $V_Z = -20 + 0.05\ I$. בשילוב של משוואות האלה נקבל:

$$I = \frac{V_T + 20}{R_T + 0.05}$$

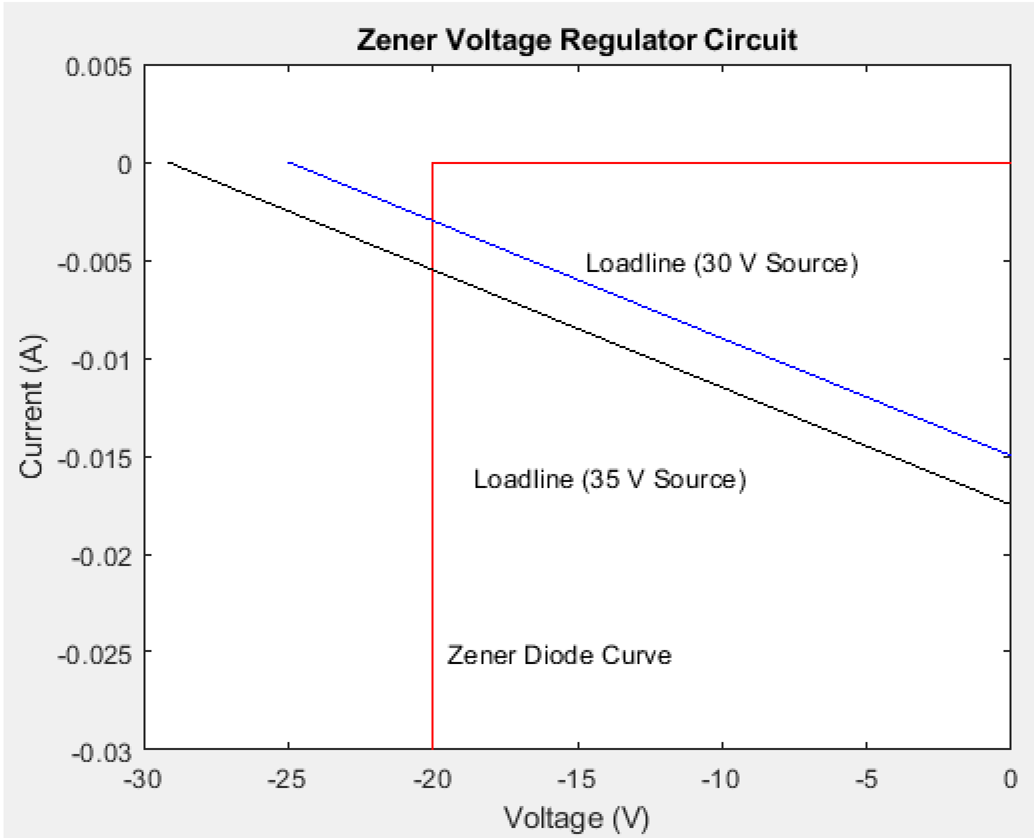
נציב את התוצאה לתוך משוואה: $V_Z = V_T - R_T I$ ואז נקבל: $V_Z = V_T - \frac{R_T(V_T + 20)}{R_T + 0.05}$

קלט:

```
>> % Zener diode voltage regulator
vs1 = -30; vs2 = -35; rl =10e3; rs = 2e3;
i = -50e-3: 5e-3 :0;
vz = -20 + 0.05*i;
m = length(i);
i(m+1) = 0; vz(m+1) = -10;
i(m+2) = 0; vz(m+2) = 0;
% loadlines
vt1 = vs1*rl/(rl+rs);
vt2 = vs2*rl/(rl+rs);
rt = rl*rs/(rl+rs);
l1 = vt1/20;
l2 = vt2/20;
v1 = vt1:abs(l1):0;
i1 = (vt1 - v1)/rt;
v2 = vt2:abs(l2):0;
i2 = (vt2 - v2)/rt;
% plots of Zener characteristics, loadlines
plot(vz,i,'r',v1,i1,'b',v2,i2,'k')
axis([-30,0,-0.03,0.005])
title('Zener Voltage Regulator Circuit')
xlabel('Voltage (V)')
ylabel('Current (A)')
text(-19.5,-0.025,'Zener Diode Curve')
text(-18.6,-0.016, 'Loadline (35 V Source)')
text(-14.7,-0.005,'Loadline (30 V Source)')
% output voltage when vs = -30v
ip1 = (vt1 + 20)/(rt + 0.05)
vp1 = vt1 - rt*(vt1+20)/(rt + 0.05)
% output voltage when vs = -35v
ip2 = (vt2 + 20)/(rt + 0.05)
vp2 = vt2 - rt*(vt2+20)/(rt + 0.05)
```

פלט:

```
ip1 =
  -0.0030
vp1 =
 -20.0001
ip2 =
  -0.0055
vp2 =
 -20.0003
```


Zener Voltage Regulator Circuit
0.005
0
-0.005
-0.01
-0.015
-0.02
-0.025
-0.03
Current (A)
-30
-25
-20
-15
-10
-5
0
Voltage (V)
Loadline (30 V Source)
Loadline (35 V Source)
Zener Diode Curve