F/A-18 High Alpha fighter testing high angle-of-attack aerodynamics in 80x120ft W.T.

2 MATLAB, შესავალი

შესავალი

მოდელირება არის აბსტრაქტული, კონცეპტუალური გრაფიკული ან მათემატუკური მოდელის შექმნა. მეცნიერება გვთავაზობს უამრავ მეთოდს, ტექნიკასა და თეორიას მოდელირებისათვის.

მოდელირება ძირითადი, გაუყოფელი ნაწილია თანამედროვე სამეცნიერო საქმიანობაში. თიტოეულ სამეცნიერო სფეროს თავისი სპეციფიკური მეთოდები გააჩნია მოდელირებისათვის.

მოდელი ნიშნავს ობიექტის, მოვლენისა თუ ფიზიკური პროცესის ემპირიულ წარმოდგენას ლოგიკური და ობიექტური გზით. მოდელი არის რეალობის გამარტივებული ასახვა. მაგრამ მიუხედავად მათი სიყალბისა, ზალზე მნიშვნელოვან ინფორმაციას იძლევა. მოდელის აგება სიმულირება ფუნდამენტურ როლს თამაშობს თანამედროვე მეცნიერებაში.

საზოგადოდ მოდელირებას მიმართავენ როცა შეუძლებელია, ან არაპრაქტიკულია ეხპერიმენტული პირობების შექმნა, სადაც მეცნიერებმა უნდა უშუალოდ შეაფასონ ეხპერომენტის შედეგი. კონტროლირებად პირობებში ჩატარებული ექსპერიმენტის შედეგები რა თქმა უნდა უფრო ზუსტია, ვიდრე მოდელირების. იმიტომ რომ მოდელირებისას ხშირ შემთხვევაში უამრავ დაშვებას მივმართავთ, რაც თავიდანაა აცილებული ეხპერიმენტის ჩატარებისას.

სიმულაცია არის მოდელის განხორციელება. მდგარდი მდგომარეობის სიმულაცია გვაწვდის ინფორმაციას დროის გარკვეულ მომენტში, ხოლო დინამიური სიმულაცია იძლევა

ინფორმაციას დროის გარკვეულ შუალედში. სიმულაციას მოდელი მოქმედებაში მოყავს და გვიჩვენებს როგორ იქცევა კონკრეტული ობიექტი, ან რა მოსდის გარკვეულ პროცესს დროში. სიმულაცია გამოიყენება ტესტირებისა და ანალიზისათვის როცა რეალური სისტემა ან კონცეპცია შესაძლებელია წარმოვადგინოთ მოდელის სახით.

მოდელი ძირითადად იყენებს მოვლენის მხოლოდ ზოგიერთ ასპექტს და ორი სხვადასხვა მოდელი ერთიდაიგივე მოვლენისა, შესაძლოას სრულიად განსხვავდებოდეს ერთმანეთისაგან. მოდელის აგებისას კარგად უნდა გვესმოდეს მოდელირების ძირითადი მიზანი და ის დაშვებანი, რასაც ვიყენებთ მოდელირებისას

კომპიუტერი ძირითადი საშუალებაა საინჟინრო პრობლემების ამოხსნის საქმეში. ამ თავში მოკლედ მიმოვიხილავთ გამოთვლით სისტემებს, მათ შორის კომპიუტერის ტექნიკურ(ინსტრუმენტალ) აღჭურვილობას და პროგრამულ უზრუნველყოფას. შემდეგ შევუდგებით MATLAB-ს, რომელიც გახლავთ წამყვანი პროგრამული პაკეტი ინტერაქტიული რიცხვითი გამოთვლების, მონაცემთა ანალიზის და მათი გრაფიკული ინტერპრეტაციისათვის. მოკლე შესავლის შემდეგ განვიხილავთ MATLAB-ში მონაცემთა წარმოდგენის საშუალებებს, შემდეგ არსებული მონაცემების ბაზაზე ავაგებთ გრაფიკს.

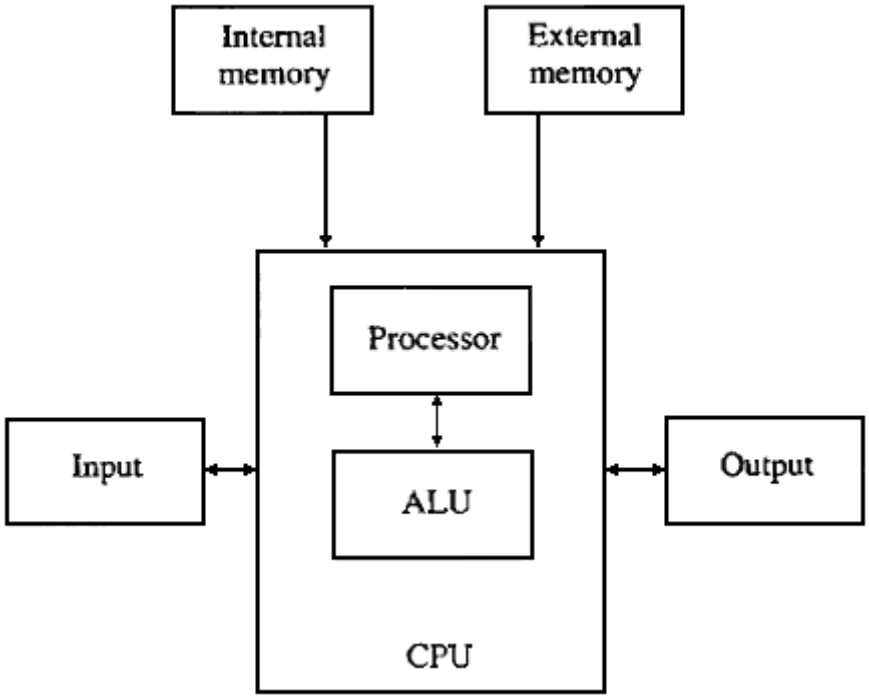
2.1 გამოთვლითი სისტემები

ვიდრე შევუდგებით MATLAB-ის შესწავლას, მოკლედ მიმოვიხილოთ კომპიუტერული სისტემები. კომპიუტერი არის მოწყობილობა, რომელიც შეიქმნა იმისათვის, რომ აწარმოოს ოპერაციები, რომელნიც განსაზღვრულია ინსტრუქციების მწკრივით, ეგრეთ წოდებული პროგრამით. კომპიუტერის აღჭურვილობას (HARDWARE) წარმოადგენს ისეთი მოწყობილობანი, როგორიცაა კლავიატურა, მაუსი, მყარი დისკი, პრინტერი და სხვა. კომპიუტერის პროგრამული უზრუნველყოფა (SOFTWARE) წარმოადგენს პროგრამას, რომელიც მოიცავს იმ ბრძანებებს, რაც კომპიუტერმა უნდა შეასრულოს.

კომპიუტერის აღჭურვილობა

ყველა კპმპიუტერს აქვს ზოგადად ერთნაირი შიდა ორგანიზაცია რომელიც ნაჩვენებია ნახ. 2.1 პროცესორი არის კომპიუტერის ნაწილი, რომელიც ყველა სხვა დანარჩენს აკონტროლებს. ის ღებულობს შესავალ მნიშვნელობებს (მაგალითად კლავიატურიდან) და ათავსებს მათ მეხსიერებაში. ის აგრეთვე ინტერპრეტაციას უკეთებს პროგრამაში მითითებულ ინსტრუქციებს. თუ გვინდა ერთმანეთს მივუმატოთ ორი სიდიდე, კომპიუტერმა უნდა მოძებნოს ეს სიდიდეები მეხსიერებაში და გააგზავნოს ის არითმეტიკულ ლოგიკურ მოწყობილობაში (ALU). ALU შეასრულებს შეკრებას და პროცესორი მიღებულ შედეგს მოათავსებს მეხსიერებაში. პროცესორი და ALU იყენებენ მეხსიერების მცირე ნაწილს-შიდა მეხსიერებას. მონაცემთა უმრავლესობა განთავსებულია გარე მეხსიერებაში-მყარი დისკი, მაგნიტური დისკი, რომლებიც დაკავშირებულია პროცესორთან. პროცესორი და ALU ერთად წარმოადგენს ცენტრალურ პროცესორს – CPU. მიკროპროცესორი არის CPU, რომელიც მოთავსებულია მცირე ზომის მიკროსქემაში (integrated circuit chip), რომელიც ასევე შეიცავს ათასობით სხვა კომპონენტს და მისი ზომა დაახლოებით თქვენი ფრჩხილისოდენაა.

გამოთვლილი სიდიდეები შესაძლებელია დავინახოთ მონიტორზე, ან დავბეჭდოთ პრინტერზე. (ლაზერული, მატრიცული, ჭავლური). მონაცემები შეგვიძლია აგრეთვე ჩავწეროთ მაგნიტურ დისკზე. დაბეჭდილ ინფორმაციას უწოდებენ მყარ ასლს (HARDCOPY), ხოლო მაგნიტურ ასლს – ელექტრონულ ასლს.



ნახ. 2.1 კომპიუტერის შიდა ორგანიზაცია

არსებობს სხვადასხვა ზომის და ფორმის კომპიუტერი. პრაქტიკაში უფრო ხშირად გვხვდება პერსონალური კომპიუტერი. უფრო მძლავრი მინიკომპიუტერი და უნივერსალური კომპიუტერი ძირითადად გამოიყენება ლაბორატორიებში, ქსელური კომპიუტერი (WORKSTATION), სუპერკომპიუტერი - უსწრაფესი კომპიუტერი, რომელიც გამოიყენება ურთულესი ამოცანების ამოსახსნელად, რომელთაც სხვა ტიპის კომპიუტერი ვერ უმკლავდება.

კომპიუტერი უნდა შეირჩეს იმ ამიცანის შესაბამისად, რომლის ამოხსნასაც ვისახავთ მიზნად. მაგალითად თუ კომპიუტერი საშინაო უსაფრთხოების სისტემის ნაწილია, მიკროპროცესორიც საკმარისია, მაგრამ თუ მიზნად ვისახავთ რაიმე პროცესის მოდელირებას, უნივერსალური კომპიუტერია საჭირო. კომპიუტერული ქსელი სასარგებლოა იმით, რომ შესაძლებელია მათ შორის მონაცემთა გაცვლა და მათი რესურსების გაერთიანება.

პროგრამული უზრუნველყოფა

პროგრამული უზრუნველყოფა შეიცავს ინსტრუქციებს, ბრძანებებს, რომელიც კომპიუტერმა უნდა შეასრულოს. არსებობს პროგრამული უზრუნველყოფის რამდენიმე მნიშვნელოვანი კატეგორია. განვიხილოთ ისინი:

ოპერაციული სისტემა. ოპერაციული სისტემა პროგრამული უზრუნველყოფის ის ფორმაა, რომელიც საშუალებას გვაძლევს ვიურთიერთოთ კომპიუტერთან. ეს არის ინტერფეისი, მოხერხებული გარემო, სადაც შეგვიძლია შევარჩიოთ და გავუშვათ სხვადასხვა პროგრამა. ოპერაციულ სისტემას აქვს პროგრამათა ჯგუფი, ე.წ. utilities, რომლებიც საშუალებას გვაძლევს შევასრულოთ ფაილების ორგანიზება, კოპირება, დაბეჭდვა. ეს პროცესი საერთოა ყველა სახის ოპერაციული სისტემისათვის, თუმცა მათი განხორციელების ფორმა განსხვადება.

პროგრამული საშუალებები. პროგრამული საშუალებები არის პროგრამები, რომელთა საშუალებით ხორციელდება ზოგადი ოპერაციები. მაგალითად Microsoft Word და Word Perfect გვეხმარება შევქმნათ ტექსტი და მოვახდინოთ მისი ფორმატირება, წავშალოთ სიტყვები და წინადადებები, შევცვალოთ შრიფტი და შევამოწმოთ ტექსტის მართლწერა.

ცხრილური პროგრამები საშუალებას გვაძლევს წარმოვადგინოთ ჩვენს ხელთ არსებული მონაცემები ცხრილის სახით. ეს პროგრამები თავიდან შექმნილი იყო ფინანსური და საბუღალტრო მონაცემებისათვის, მაგრამ მათ დიდი გამოყენება ჰპოვეს მეცნიერებაშიც. ცხრილურ პროგრამებს აქვთ აგრეთვე გრაფიკული საშუალებებიც, რაც მათ უფრო მოხერხებულს ხდის.

ერთერთი პოპულარული პროგრამული საშუალებაა მონაცემთა ბაზის მართვის პროგრამა, რომელიც საშუალებას იძლევა ვიმუშაოთ მონაცემთა ვრცელ სისტემასთან. მარტივად ამოვიღოთ მონაცემთა ჩვენთვის სასურველი ნაწილი. ამ პროგრამას ფართოდ იყენებენ ბანკებში, სასტუმროებსა თუ მომსახურების სხვა სისტემაში. მას ფართოდ იყენებენ მეცნიერებაშიც, მაგალითად, მეტეოროლოგიური მონაცემები მაგალითია სამეცნიერო მონაცემევისა, რომელიც მოითხოვს მონაცემთა ვრცელი ბაზის შექმნას და შემდგომში მის მეცნიერულ ანალიზს.

გრაფიკული პაკეტი საშუალებას იძლევა ავაგოთ მრავალფეროვანი გრაფიკები, ორგანზომილებიანი, სამგანზომილებიანი, ბარ და კონტურული გრაფიკები. KAD (Komputer-aidid design) პაკეტებს, როგორიცაა AutoCAD, AutoSketch და MathCAD აქვთ აგრეთვე მრავალფეროვანი გრაფიკული შესაძლებლობანი.

არესებობს აგრეთვე მძლავრი კომპიუტერული გამოთვლითი საშუალებები, როგორიცაა MATLAB, Mathematika. გარდა იმისა, რომ მათ გაჩნიათ მათემატიკური გამოთვლების მძლავრი საშუალებები, ფართო გრაფიკულ შესაძლებლობებსაც ფლობენ. ეს კომბინაცია მძლავრი მათემატიკური აპარატისა და ვიზუალიზაციისა, განაპირობებს ამ პროგრამული პაკეტების ფართო გამოყენებას საინჟინრო ამოცანების გადაწყვეტაში.

კომპიუტერული ენა. ეს არის ენა, რომელიც "ესმის" კომპიუტერს. დღეისათვის კპმიუტერის მუშაობა ემყარება ორსაფეხურიან (two-state) ტექნოლოგიას. (მოწყობილობა ორი მდგომარეობით როგორც ღია ან შეკრული წრედი, ან ჩართული და გამორთული, უარყოფითი ან დადებითი მუხტი), მანქანური ენა იწერება ორი სიმბოლოს საშუალებით 0 და 1. მანქანის ენა არის ორობოთი და ინსტრუქციები იწერება 0 და 1 შედგენილი მწკრივის სახით - ორობითი სტრიქონი. მანქნურ ენას უწოდებენ აგრეთვე დაბალი დონის ენას და იგი დამოკიდებულია კომპიუტერის სტრუქტურაზე, მაგ. SAN მანქანების ენა განსხვავდება VAX კომპიუტერების მანქანური ენისაგან.

მაღალი დონის ენას უწოდებენ კომპიუტერულ ენას, რომელსაც აქვს ჩვეულებრივი სალაპარაკო ინგლისური ენის მსგავსი ბრძანებები და ინსტრუქციები. ესენია C, Fortran, Pascal, Basic. პროგრამის დაწერა მაღალი დონის ენაზე რა თქმა უნდა გაცილებით მარტივია, ვიდრე დაბალი დონის ენაზე. მაღალი დონის ენაზე დაწერილი პროგრამა უნდა შეიცავდეს დაწვრილებით ინსტრუქციებს იმის შესახებ თუ რა უნდა გააკეთოს კომპიუტერმა. გარდა ამისა მაღალი დონის ენას გაჩნია უამრავი ბრძანება და სინტაქსი.

თუ შესაძლებელია პრობლემის ამოხსნა პროგრამული საშუალებებით, უმჯობესია გამოვიყენოთ ეს შესაძლებლობა, რადგან ეს მისი გადაჭრის უსწრაფესი გზაა. მაგრამ თუ სისტემური საშუალებებით არ ხერხდება დასახული ამოცანის გადაჭრა, კომპიუტერული

ენები საშუალებას გვაძლევს დავწეროთ ახალი პროგრამა ჩვენთვის სასურველი საკითხის გადასაწყვეტად.

2.2 ზოგადი ინფორმაცია MATLAB-ის შესახებ

MATLAB-ი შეიქმნა როგორც "მატრიცული ლაბორატორია". დღეისათვის იგი წარმოადგენს იტერაქტიულ სისტემურ და პროგრამულ ენას ფართო სამეცნიერო და ტექნიკური გამოთვლებისათვის. მისი ძირითადი ელემენტია მატრიცა. MATLAB-ში პროგრამის დაწერა გაცილებით მარტივია, ვიდრე რომელიმე მაღალი დონის ენაზე. ამ თავში განვიხილავთ რამდენიმე ძირითად ინფორმაციას სამუშაო სივრცის (workspace) შესახებ.

სტუდენტური ვერსია

სტუდენტური ვერსია პროფესიულის იდენტურია გარდა რამდენიმე მახასიათებლისა.

- ყოველი ვექტორი ან მატრიცა შეიცავს არაუმეტეს 1024 ელემენტისა
- მეტაფაილი და გრაფიკული პოსტპროცესორი გრაფიკების დასაბეჭდად შეუძლებელია
- მათემატიკური თანაპროცესორი არ არის საჭირო, მაგრამ შესაძლოა გამოყენებული იქნას, თუ შესაძლებელია
- სიგნალის და სისტემური ტულბოქსი

სამუშაო სივრცე (**workspace**). იმისათვის რომ შევუდგეთ მუშაობას MATLAB–ში, შეარჩიეთ იგი თქვენს ოპერაციულ სისტემაში ან აკრიფეთ MATLAB კლავიატურაზე. დაინახავთ ნიშანს (>>), რაც იმის მანიშნებელია, რომ MATLAB –ი ელოდება თქვენს ბრძანებას. თუ ეს თქვენი პირველი ნაბიჯია MATLAB–ში, ისარგებლეთ ბრძანებით **demo** იმისათვის რომ ნახოთ სადემონსტრაციო პროგრამების სია, რომელიც შეგიძლიათ გაუშვათ და ნახოთ MATLAB-ის შესაძლებლობები. იმისათვის, რომ დაასრულოთ მუშაობა MATLAB-ში ისარგებლეთ ბრძანებებით: **quit** ან **exit**. მაგრამ თუ გსურთ შეინახოთ თქვენს მიერ შექმნილი ცვლადები მუშაობის დამთავრებამდე, ისარგებლეთ ბრძანებით **save**. ასე შეინახავთ ცვლადებს ფაილში, რომლის სახელი იქნება matlab.mat. როცა ხელახლა გახსნით MATLAB-ს, შეგიძლიათ აღადგინოთ ცვლადები ბრძანებით **load**. ბრძანება **computer** გვაძლევს ინფორმაციას კომპიუტერის ტიპის შესახებ.

MATLAB-ს აქვს ბრძანებების ფანჯარა, რომელიც საშუალებას გვაძლევს შევიყვანოთ ბრძანებები და მივიღოთ შედეგები იქვე, ეკრანზე ან დავბეჭდოთ და გრაფიკების ფანჯარა. ორივე ფანჯარა ცარიელია, როცა იწყებთ მუშაობას. თუ გსურთ გაასუფთაოთ ბრძანებების ფანჯარა მუშაობის პროცესში, უნდა გამოიყენოთ ბრძანება **clc**. გრაფიკების ფანჯრის გასასუფთავებლად გამოიყენება ბრძანება ცლფ. ასევე შესაძლებელია გავასუფთაოთ სამუშაო სივრცე – **clear**. ამით ყველა ცვლადი, რომელიც შეიქმნა მოცემულ სამუშაო სივრცეში, წაიშლება.

მნიშვნელოვანია ვიცოდეთ როგორ შევწყვიტოთ უკვე მიცემული ბრძანება. მაგალითად როცა რომელიმე ბრძანების გამო კომპიუტერი გაუთავებლად იმეორებს რაიმე ციკლს, ჩაიციკლება. პროცესის შესაწყვეტად ისარგებლეთ ბრძანებით **ctrl+c**.

იმისათვის, რომ გავიგოთ ჩვენს მიერ შექმნილი ცვლადების და მატრიცების სტატუსი MATLAB-ს აქვს რამდენიმე ბრძანება. ბრძანება **who** გვაძლევს ცვლადების სიას, რომელიც შევქმენით, ხოლო **whos** დამატებით ინფორმაციასაც იძლევა მათი ზომის, მანქანური მოცულობის და კლასის შესახებ.

ეს ბრძანებები ცვლადებში მოიხსენიებენ ცვლადს სახელით ანს, რომელიც შესაძლოს თქვენ არ შეგიქმნიათ. ეს სახელი გამოიყენება ცვლადისათვის, რომელიც გამოვითვალეთ, მაგრამ სახელი არ მივანიჭეთ. ბრძანება **size** საჭიროა თუ გვსურს გავიგო რომელიმე ცვლადის ზომა. მაგ size(A).

MATLAB-ის ბრძანებები ჩვეულებრივ იკრიფება ახალ სტრიქონში, მაგრამ შეგვიძლია რამდენიმე ბრძანება მივცეთ ერთად, თუ მათ წერტილ-მძიმით გამოვყოფთ. ტექსტი, რომელიც მოყვება ნიშანს **%** MATLAB-ის მიერ იგნორირებულია და გამოიყენება კომენტარისათვის. ჩვენ გამოვიყენებთ ამ ნიშანს ჩვენს მაგალითებში კომენტარებისათვის, მაგალითად ცვლადების გამნმარტებისთვის.

MATLAB-ის მნიშვნელოვანი რესურსია **help** ბრძანება. მას მოჰყვება სია იმ საკითხებისა, რომელთათვისაც შეგვიძლია ვისარგებლოთ ამ ბრძანებით.

M – ფაილები

გარდა იმისა, რომ ბრძანებები შეგვიძლია მივცეთ ბრძანებების ფანჯრიდან, შეგვიძლია წინასწარ შევქმნათ ბრძანებათა მწკრივი ტექსტური ფაილის სახით, რომლის სახელს უნდა ჰქონდეს გაფართოება .m, მაგ. labtest.m ასეთ ფაილებს ეწოდება **M ფაილები.** ფაილის სახით შენახული ბრძანებათა ასეთი მწკრივი შეგვიძლია შემდგომშიც გამოვიყენოთ. ამ ფაილებს აგრეთვე უწოდებენ ხელნაწერ ფაილებს, რადგან წარმოადგენს ტექსტურ ASCII ფაილს და შეგვიძლია შევქმნათ ტექსტური რედაქტორით ან წორდ პროცესორით. თუ გვინდა ბრძანებების ფანჯარაში გამოვიყვანოთ .m ფაილის ტექსტი უნდა ვისარგებლოთ ბრძანებით **echo**

M ფაილები ასევე გამოიყენება MATLAB-ის ახალი ფუნქციის შესაქმნელად.

ბრძანება **what** გვაძლევს იმ ფაილების სიას, რომლების განთავსებულია მიმდინარე დირექტირიაში. ბრძანება **type** გვაძლევს მითითებული ფაილის შინაარსს.

2.3 მატრიცა, ვექტორი, სკალარი

საინჟინრო ამოცანის ამოხსნისას მნიშვნელოვანია მოვახდინოთ პრობლემასთან დაკავშირებული მონაცემების ვიზუალიზაცია. ზოგჯერ ეს მონაცემი ერთი რიცხვია, მაგ. წრის რადიუსი. სხვა შემთხვევაში ეს შესაძლოა იყოს წერტილის კოორდინატები (x,y) სიბრტყეზე, ანდა შეიძლება გვქონდეს x-y-z კოორდინატების 4 წყება, რომელიც წარმოადგენს სივრცეში პირამიდის ოთხ წვეროს. ეს მონაცემები შეგვიძლია წარმოვადგინოთ მონაცემთა სპეციალური სტრუქტურის – მატრიცის სახით. მატრიცა ეს არის მონაცემთა მწკრივი დალაგებული სტრიქონების და სვეტების სახით. ერთი წერტილი შეგვიძლია განვიხილოთ როგორც მატრიცა, რომელიც შედგება ერთი სტრიქონისა და ერთი სვეტისაგან, x, y წყვილი – მატრიცა ერთი სტრიქონით და ორი სვეტით, ხოლო x-y-z კოორდინატების 4 წყება – მატრიცა ოთხი სტრიქონისა და სამი სვეტისაგან. მაგალითად:

```
A = [3.5]
B = [1.5 3.2]
```

$$C = \begin{matrix} -1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 2 \end{matrix}$$

თუ მატრიცა შეიცავს ერთ სტრიქონსა და ერთ სვეტს, იგი სკალარია, თუ იგი შეიცავს 1 სტრიქონს და რამდენიმე სვეტს – სტრიქონი ვექტორია, ხოლო თუ შეიცავს რამდენიმე სტრიქონსა და 1 სვეტს – იგი სვეტი ვექტორია.

მატრიცის ელემენტებს მიუთითებენ ორი ინდექსით – სტრიქონის და სვეტის ნომერი. მაგ. C მატრიცის მეოთხე სტრიქონის და მესამე სვეტის ელემენტია $C_{4,3}=0$.

მატრიცის ზომა განისაზღვრება სტრიქონების და სვეტების რაოდენობით. მაგ. C მატრიცის ზომაა 4×3. თუ მატრიცის სვეტების და სტრიქონების რაოდენობა ერთიდაიგივეა, მას კვადრატულს უწოდებენ.

სავარჯიშო

უპასუხეთ კითხვებს შემდეგი მატრიცის მაგალითზე:

$$C = \begin{bmatrix} 0.6 & 1.5 & 2.3 & -0.5 \\ 8.2 & 0.5 & -0.1 & -0.2 \\ 5.7 & 8.2 & 9.0 & 1.5 \\ 0.5 & 0.5 & 2.4 & 0.5 \\ 1.2 & -2.3 & -4.5 & 0.5 \end{bmatrix}$$

1. რა ზომისაა მატრიცა?
2. არის თუ არა იგი კვადრატული?
3. დაწერეთ აღნიშვნა მატრიცის ელემენტებისა, რომელთა მნიშვნელობა 0.5 ტოლია
4. დაწერეთ აღნიშვნა მატრიცის ელემენტებისა, რომელთა მნიშვნელობა უარყოფითია

მატრიცის ინიციალიზაცია

როგორ განვსაზღვროთ მატრიცა MATLAB-ში? განვიხილავთ ოთხ სხვადასხვა მეთოდს. პირველი მეთოდი ცხადად განვსაზღვროთ მატრიცის ელემენტები, მეორე მეთოდი – წავიკითხოთ მატრიცა ფაილიდან, მესამე – გამოვიყენოთ (ორწერტილიანი) კოლონოპერატორი (:), მეოთხე- შევიყვანოთ მონაცემები კლავიატურიდან.

პირველი მეთოდი – ცხადად განვსაზღვროთ მატრიცა. ჩავწეროთ კვადრატულ ფრჩხილებში მატრიცის ელემენტები:

```
A= [3.5];
B= [1.5, 3.1];
```

```
C= [-1, 0, 0; 1, 1, 0; 1,- 1, 0; 0, 0, 2];
```

ეს ბრძანებები მაგალითია, თუ როგორ შევქმნათ სხვადასხვა ზომის მატრიცა. მატრიცის სახელი უნდა იწყებოდეს ასოითი გამოსახულებით და შედგებოდეს არაუმეტეს 19 ნიშნისა (დასაშვებია ციფრები, ასოები და ქვედა ტირე) და უნდა ეწეროს უდრისის ნიშნის მარცხნივ. ტოლობის ნიშნის მარჯვენა მხარე წარმოადგენს კვადრატულ ფრჩხილებში ჩასმულ მატრიცის ელემენტების მნიშვნელობებს, წერტილმძიმით სტრიქონები გამოიყოფა ერთმანეთისაგან, ხოლო სტრიქონებში მონაცემები შეიძლება გამოვყოთ მძიმით ან მის გარეშე. გამოსახულება შეიძლება შეიცავდეს + ან – ნიშანს, ათწილადის ნიშანს “.”და არა “,”. როდესაც განვსაზღვრავთ მატრიცას, თუ ბოლოში არ დავუსვამთ წერტილ-მძიმეს, MATLB ბრძანებების ფანჯარაში გამოგვიყვანს მის გამოსახულებას. სცადეთ შეიყვანოთ ზემოთაღნითნული მონაცემები ბოლოში წერტილ-მძიმის დაურთველად.

მატრიცა შეგვიძლია განვსაზღვროთ ისე, რომ სრტიქონები წერტი-მძიმით კი არ გამოვყოთ ერთმანეთისგან, არამედ გადავიტანოთ შემდეგ ხაზზე “ ნტერ” კლავიშის საშუალებით.

```
C = [-1 0 0
     1 1 0
     1 –1 0
     0 0 2] ;
```

თუ მატრიცის სტრიქონში ბევრი ელემენტებია, შეგვიძლია გარკვეული რაოდენობით ელემენტების შემდეგ დავწეროთ სამი წერტილი და სტრიქონი გავაგრძელოთ შემდეგ ხაზზე. მაგალითად:

```
F = [ 1 0 5 7 3 9 6 7 8 0 4];
```

იგივეა რაც

```
F = [ 1 0 5 7 3 . . .
      9 6 7 8 0 4];
```

MATLAB საშუალებას იძლევა განვსაზღვროთ ახალი მატრიცა უკვე შექმნილის ბაზაზე. მაგალითად განვიხილოთ შემდეგი ბრძანებები:

```
B = [1.5, 3.1];
S = [3.0 B];
```

ეს იგივეა, რაც:

```
S = [3.0 1.5 3.1];
```

ასევე შეგვიძლია შევცვალოთ მატრიცის ელემენტების მნიშვნელობები, ან დავუმატოთ ახალი ელემენტები:

```
S(2) = -1.0;
```

ეს ბრძანება შეცვლის შ მატრიცის მეორე ელემენტს 1.5 მიანიჭებს რა მას ახალ მნიშვნელობას –1.0.

ასევე შესაძლებელია მატრიცას დავუმატოთ ახალი ელემენტები:

```
S(4) = 5.5;
```

ამის შედეგად სამელემენტიანი მატრიცა ოთხელემენტიანად იქცევა. მაგრამ თუ მივცემთ ბრძანებას:

```
S(8) = 9.5;
```

მატრიცა გახდება 8 ელემენტიანი, ხოლო მისი მეხუთე, მეექვსე და მეშვიდე ელემენტები მიიღებენ 0-ის ტოლ მნიშვნელობებს.

სავარჯიშო

განსაღვრეთ მტრიცის ზომა და შეამოწმეთ იგი MATLAB-ის საშუალებით:

1. A = [1, 0, 0, 0, 0, 1];
2. B = [2; 4; 6; 10];
3. C = [5 3 6; 6 2 –3];
4. D = [3 4

   05 7
   9 10];
5. E = [3 5 10 0; 0 0 . . .
   0 3; 3 9 9 8];
6. T = [4 24 9];
   Q = [T 0 T];
7. X = [3 6];
   Y = [D; X];
8. R = [C; X; 5];
9. V = [C(2,1); B];
10. A(2,1) = -3;

მატრიცა შეგვიძლია შევქმნათ აგრეთვე იმ ინფორმაციის საშუალებით, რომელიც ჩაწერილია ტექსტურ ფაილში – MAT, რომელიც შეიცავს ინფორმაციას ორობით ფორმატში ან ASCII ფაილში.

MAT ფაილი იქმნება MATLAB-ის მიერ **save** ბრძანების საშუალებით. მაგალითად ბრძანება

save data1 x y;

შეინახავს x და y მონაცემებს ფაილში data1.mat .mat გაფართოებას პროგრამა ავტომატურად დაურთავს ფაილის სახელს.

ამ მონაცემთა სამუშაო სივრცეში გამოსაძახებლად ვსარგებლობთ ბრძანებით:

load data1;

ASCII ფაილი, რომელიც გვინდა MATLAB –ის პროგრამაში გამოვიყენოთ, უნდა შეიცავდეს მხოლოდ რიცხვით მნიშვნელობებს, ამასთანავე, ყოველი სტრიქონი ელემენტთა ერთნაირ

რაოდენობას უნდა შეიცავდეს. ეს ფაილი შეიძლება შევქმნათ ნებისმიერი ტექსტური რედაქტორის საშუალებით და აგრეთვე MATLAB-ის მეშვეობით თუ გამოვიყენებთ ბრძანებას:

save data1.dat Z /ascii;

Z მატრიცის ყოველი სტრიქონი ჩაიწერება ფაილის ცალკე სტრიქონის სახით. mat გაფართოება არ დაერთვის ascii ფორმატის ფაილს, მაგრამ უმჯობესია ფაილს მივანიჭოთ dat გაფართოება რათა ვიზუალურად განვასხვაონ MAT და M ფაილებისაგან.

დავუშვათ ASCII ფაილი data1.dat შეიცავს დროის და შესაბამისი მანძილის მნიშვნელობათა მწკრივს. ამ ფაილის პირველი სტრიქონები დაახლოებით ასე უნდა გამოიყურებოდეს:

0.00	0.00
0.01	0.1255
0.02	0.2507

ბრძანება:

load data1.dat;

მიმართავს მითითებულ ფაილს და შექმნის ორსვეტიან მატრიცას data1 სამუშაო სივრცეში.

ორწერტილოვანი ოპერატორის საშუალებით შეგვიძლია მატრიცისაგან შევქმნათ ვექტორი. მაგალითად თუ გვსურს ავაგოთ გრაფიკი x-y მონაცემთა მიხედვით, მოხერხებულია თუ გვექნება x და y მონაცემები ვექტორის სახით. როცა ორი წერტილი გვაქვს მატრიცის ელემენტთა აღნიშვნაში, ის მიუთითებს ყვალა სტრიქონს ან ყველა სვეტს. მაგალითად თუ ავიღებთ ჩვენს მიერ განხილულ მონაცემებს ფაილიდან data1.dat შემდეგი ბრძანებები ააგებს ორ სვეტ ვექტორს x და y :

x = data1(: , 1);
y = data1(: , 2);

ეს ოპერატორი ასევე გამოიყენება ახალი მატრიცის საწარმოებლად. თუ იგი დგას ორ მთელ რიცხვს შორის, შეიქმნება ვექტორი, რომლის ელემენტები იქნება ყველა მთელი რიცხვი, რომელიც ამ ორ რიცხვს შორის მდემარეობს, მათი ჩათვლით. მაგალითად:

```
H = 1:8
```

იძლევა ვექტორს

```
H = [1 2 3 4 5 6 7 8];
```

თუ ორწერტილოვანი ოპერატორი 3 რიცხვს გამოყოფს ერთმანეთისაგან, მაშინ შეიქმნება ვექტორი, რომლის ელემენტები იქნება რიცხვები პირველსა და მესამე რიცხვს შორის, ნაზრდით მეორე რიცხვი:

```
Time = 0.0:0.5:5.0;
```

გვაძლევს

```
Time = [0.0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5];
```

ნაზრდი შეიძლება იყოს უარყოფითი.

ორწერტილოვანი ოპერატორი გამოიყენება აგრეთვე იმ შემთხვევაში, თუ გვინდა ამოვიღოთ მატრიცა უკვე შექმნილი მატრიციდან. დავუშვათ გვაქვს მატრიცა :

$$C = \begin{bmatrix} -1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

თუ მივცემთ ბრძანებებს:

```
C_PARTIAL = C( : , 2 : 3);

C_PARTIAL = C(3 : 4, 1 : 2);
```

მივიღებთ ორ ახალ მატრიცას:

$$C\_PARTIAL\_1 = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ -1 & 0 \\ 0 & 2 \end{bmatrix} \qquad C\_PARTIAL\_2 = \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix}$$

თუ ორწერტილოვანი ოპერატორი გამოყენებულია მატრიცის არასწორი ინდექსისათვის, MATLAB მიგვითითებს შეცდომაზე.

MATLAB –ში ზოგჯერ გვჭირდება ცარიელი მატრიცა, იგი განსხვავდება მატრიცისაგან, რომლის ყველა ელემენტი ნულის ტოლია.

```
a = [];
b = 4 : -1 : 5;
```

ორივე ბრძანება ქმნის ცარიელ მატრიცას.

გამოსახულება C (:) თუ C მატრიცაა, ქმნის ერთ გრძელ სვეტ ვექტორს, როლის ელემენტებია C მატრიცის პიველი სვეტის ელემენტები, რომელსაც მოყვება მეორე სვეტის ელემენტები და ა. შ.

ამით არ ამოიწურება ორწერტილოვანი ოპერატორის ფუნქციები, ჩვენ მას კიდევ დავუბრუნდებით.

შესაძლებელია მატრიცის მნიშვნელობები შევიყვანოთ პირდაპირ კლავიატურიდან **input** ბრძანების საშუალებით, რომელსაც ეკრანზე გამოჰყავს ტექსტი და ელოდება მონაცემებს.

მომხმარებლის მიერ მიწოდებული მონაცემებით შეიქმნება შესაბამისი მატრიცა. თუ არ შევიყვანთ არანაირ მონაცემს (enter . . .enret), შეიქმნება ცარიელი მატრიცა. თუ ბრძანება არ მთავრდება წერტილ-მძიმით, შეყვანილი მონაცემების საფუძველზე შექმნილი მატრიცა დაიბეჭდება ბრძანებათა ფანჯარაში.

განვიხილოთ ბრძანება:

```
z = input('Enter value for z');
```

ეს ბრძანება მოგვცემს ეკრანზე ტექსტს: Enter value for z. უნდა შევიყვანოთ მონაცემები [5.1 6.3 –18.0], რომელიც მიენიჭება z. რადგან ბრძანება მთავრდება წერტილბძიმით, z მნიშვნელობა არ გამოჩნდება ეკრანზე.

სავარჯიშო

იპოვეთ ზომა და შემადგენლობა შემდეგი მატრიცებისათვის, თუ :

$$G = \begin{bmatrix} 0.6 & 1.5 & 2.3 & -0.5 \\ 8.2 & 0.5 & -0.1 & -2.0 \\ 5.7 & 8.2 & 9.0 & 1.5 \\ 0.5 & 0.5 & 2.4 & 0.5 \\ 1.2 & -2.3 & -4.5 & 0.5 \end{bmatrix}$$

11. A = G(: , 2);
12. B = G(4, :);
13. C = [10 : 15];
14. D = [4 : 9; 1 : 6];
15. E = [-5, 5];
16. F = [0.0 : 0.1 : 1.0];
17. T1 = G(4 : 5, 1 : 3);
18. T2 = G(1 :2 : 5, :);

მატრიცის დაბეჭდვა

არსებობს რამდენიმე ხერხი მატრიცის ეკრანზე გამოსაყვანად. უმარტივესია შევიყვანოთ მატრიცის სახელი და Enter. ბრძანებათა ფანჯარაში გამოჩნდება მატრიცის სახელი და მისი გამოსახულება.

როცა მატრიცის მნიშვნელობები ეკრანზე გამოგვყავს, მთელი რიცხვები გამოდის, როგორც მთელი, ხოლო ათწილადები გამოდის სპეციალური ფორმატით ე.წ. მოკლე ფორმატით - მძიმის შემდეგ 5 ნიშანი, თუ არ მივუთითებთ განსხვავებულ ფორმატს ჩვენ თვითონ. თუ გვსურს გამოვიყვანოთ რიცხვითი გამოსახულება მძიმის შემდეგ 15 ნიშნით, ვისარგებლებთ ბრძანებით **format long**. თუ კვლავ მოკლე ფორმატს გვინდა დავუბრუნდეთ – **format short.**

როცა რიცხვითი გამოსახულება ძალიან დიდია, ან ძალზე მცირე, ათწილადური ფორმა მოუხერხებელი და არაზუსტია. მაგალითად ქიმიაში ხშირად გამოიყენება ავოგადროს რიცხვი, რომლის მნიშვნელობა = 602, 300, 000, 000, 000, 000, 000, 000. მივმართავთ გამოსახულიბის ჩაწერის ეხპონენციალურ (მაჩვენებლიან) ფორმას: 1 და 10 შორის რიცხვი

გამრავლებული 10-ის შესაბამის ხარისხზე. ავოგადროს რიცხვი ასე ჩაიწერება 6.023 X 1023. 6.023 ეწოდება რიცხვის მანტისა, ხოლო 23 - ექსპონენტა. MATLAB-ში ასეთი ფორმით რიცხვის ჩაწერისას მანტისა და ექსპონენტა გამყოფილია ნიშნით e. მაგალითად 6.023 X 1023 = 6.023e+23. თუ გვინდა რიცხვითი გამოსახულება გამოვიყვანოთ ექსპონენციალური ფორმატით, ვსარგებლობთ ბრძანებით: **format short e** ან **format long e.**

ფორმატირების შემდეგი ბრძანებაა **format +.** ამ შემთხვევაში დადებითი რიცხვის მაგიერ დაიბეჭდება ნიშანი +, ისევე როგორც უარყოფითის მაგირ – ნიშანი, ხოლო თუ რიცხვის მნიშვნელობა 0 – ის ტოლია მის მაგიერ ცარიელი ადგილი დარჩება.

ტექსტის გამოყვანა ეკრანზე. ტექსტის გამოსაყვანად ეკრანზე MATLAB – ს აქვს ბრძანება **disp** , რომლის შემდეგ ფრჩხილებში იწერება ბრჭყალებში მოთავსებული ტექსტი. ეს ბრძანება ასევე გამოიყენება მატრიცის, ვექტორის ან სკალარის ეკრანზე გამოსაყვანად. მაგ. თუ სკალარი სახელით temp შეიცავს ტემპერატურის რიცხვით მნიშვნელობას-78 ფარენგეიტის გარდუსებში, ბრძანება: **disp(temp)**; **disp('degrees F')** გვაძლევს:

```
78
    degrees F
```

ფორმატირებული ბეჭდვა. ფორმატირების უფრო ვრცელ საშუალებას იძლევა ბრძანება **fprintf**. იგი ერთდროულად ტექსტის და მატრიცის დაბეჭდვის და რიცხვითი გამოსახულების ფორმატირების საშუალებას იძლევა. მისი ზოგადი ფორმა ასეთია:

fprintf (*format, matrices*)

სადაც **format** შეიცავს ბრჭყალებში ჩასმულ ტექსტს და ფორმატის სპეციფიკაციებს. ტექსტის შიგნით **%e, %f da %g** გამოიყენება იმისათვის, რომ მივუთითოთ სად უნდა დაიბეჭდოს მატრიცის რიცხვითი მნიშვნელობები. **%e** შემთხვევაში რიცხვითი გამოსახულება მოიცემა ექსპონენციალური ფორმით, **%f** შემთხვევაში რიცხვითი გამოსახულება მოიცემა ათწილადური ფორმით, **%g** შემთხვევაში – აირჩევა უმოკლესი ფორმა ექსპონენციალურსა და ათწილადურ ფორმას შორის. თუ ტექსტში არის ნიშანი **\n** ეს ნიშნავს, რომ მის მარჯვნივ მდგომი ინფორმაცია გადავა შემდეგ ხაზზე.

fprintf (' The temperature is %f degrees F \n ' , temp)

შეესაბამება:

The temperature is 78.000000 degrees F

თუ ბრძანებას ასე შევცვლით:

fprintf (' The temperature is \n %f degrees F \n ' , temp)

მივიღებთ:

```
The temperature is
78.000000 degrees F
```

ფორმატის განმსაზღვრელი ნიშნები %f, %g, %e შეიძლება შეიცავდეს ინფორმაციას თუ რამდენ ნიშანს უნდა შეიცავდეს რიცხვითი გამოსახულება და რამდენი ნიშანი დაიბეჭდოს მძიმის შემდეგ.

fprintf (‘ The temperature is %4.1f degrees F \n ’ , temp)

იძლევა:

The temperature is 78.0 degrees F

x-y გრაფიკის აგება

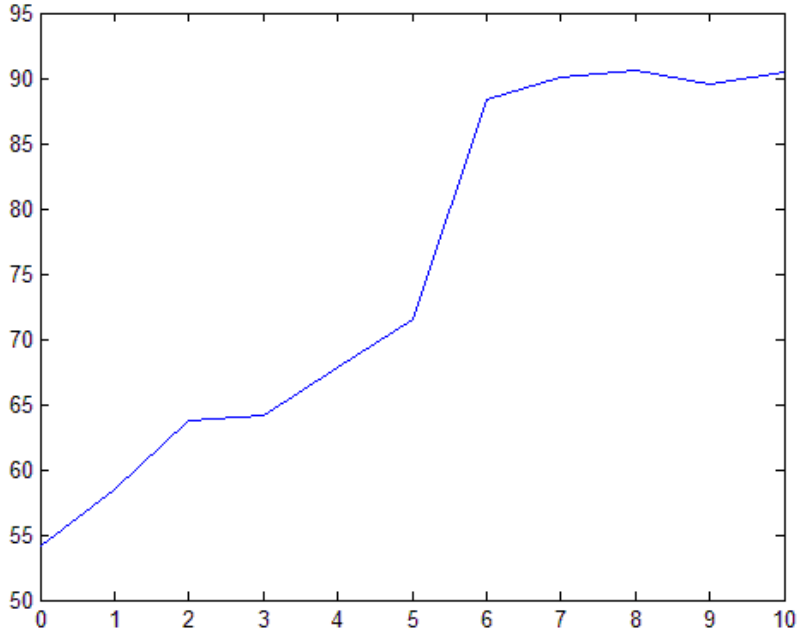
მე-7 თავში დაწვრილებით განვიხილავთ MATLAB-ის გრაფიკულ შესაძლებლობებს. ახლა კი იმის საილუსტრაციოდ თუ რაოდენ მარტივი და მოხერხებულია გრაფიკის აგება MMATLAB-ში ავაგოთ გრაფიკი x და y ვექტორების სახით წარმოდგენილი მონაცემების მიხედვით.

დავუშვათ გვინდა ავაგოთ ტემპერატურის მონაცემები, რომელიც ექსპერიმენტის ჩატარების დროს შევკრიბეთ.

დრო, წამებში	ტემპერატურა, ფარენჰეიტის გრადუსებში
0	54.2
1	58.5
2	63.8
3	64.2
4	67.8
5	71.5
6	88.3
7	90.1
8	90.6
9	89.5
10	90.4

ასევე დავუშვათ, რომ დრო მოცემული გვაქვს როგორც x ვეტორის, ხოლო ტემპერატურა – y ვექტორის ელემენტები. (ეს ვექტორები წინასწარ უნდა შევქმნათ ან ფაილიდან გამოვიძახოთ ჩვენს სამუშაო სივრცეში). გრაფიკის ასაგებად ვიყენებთ ბრძანებას **plot**, არგუმენტით x, y, სადაც ორივე ან სვეტი ვექტორია, ან სტრიქონი.

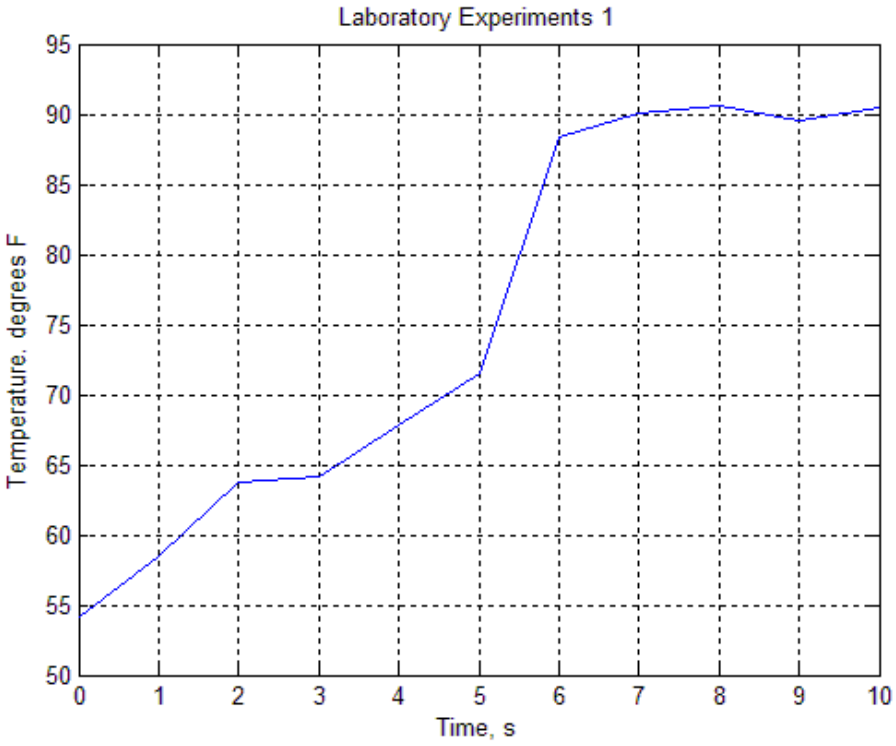
plot(x,y)



ნახ. 2.2 x, y გრაფიკი

ავტომატურად შეიქმნება გრაფიკი ნახ. 2.2. იმისათვის, რომ გრაფიკი უფრო ინფორმაციული იყოს, გავუკეთოთ მას სათაური, ღერძებს დავაწეროთ შესაბამისი სახელები და დავიტანოთ ნახაზზე საკოორდინატო ბადე ნახ. 2.3.

```
plot(x,y),. . .
title('Laboratory Experiments 1'), . . .
xlabel('Time, s'), . . .
ylabel('Temperature. degrees F'), . . .
grid
```



ნახ. 2.3 გაუმჯობესებული გრაფიკი

სამი წერტილი ყოველი სტრიქონის ბოლოს საჭიროა იმისათვის, რომ MATLAB – მა ეს ხუთივე ბრძანება ერთდროულად გაუშვას. ამის მაგივრად შეგვეძლო თითოეული ბრძანება ცალ-ცალკე მიგვეცა ერთმანეთის მიყოლებით.

თუ ერთიდაიგივე პროგრამაში გვინდა ავაგოთ ორი ნახაზი, ან ავაგოთ გრაფიკი და გავაგრძელოთ შემდგომი გამოთვლები, MATLAB გამოიყვანს ნახაზს და გააგრძელებს პროგრამაში მითითებულ ბრძანებებს, მაგრამ თუ გვინდა, რომ ნახაზს შევხედოთ, ვიდრე პროგრამა გააგრძელებს მუშაობას, **plot** ბრძანების შემდეგ უნდა დავწეროთ ბრძანება **pause**. ასეთ შემთხვევაში MATLAB დაელოდება ჩვენს დასტურს (ENTER კლავიში) პროგრამის გაგძელებისათვის.

MATLAB – ის გრაფიკულ შესაძლებლობებს დაწვრილებით გავეცნობით მე-7 თავში.

პრობლემა – გასროლილი ქვის ტრაექტორია

მცირე ზომის ობიექტი გაისოლეს დედამიწიდან 50 მილი/საათი სიჩქარით. დედამიწის ზედაპირის მიმართ 30 გრადუსი დახრის კუთხით. განვსაზღვროთ ფრენის დრო და დედამიწაზე დაცემამდე განვლილი მანძილი.

1. ამოცანის დასმა

მცირე ზომის ობიექტი გაისოლეს დედამიწიდან 50 მილი/საათი სიჩქარით. დედამიწის ზედაპირის მიმართ 30 გრადუსი დახრის კუთხით. განვსაზღვროთ ფრენის დრო და დედამიწაზე დაცემამდე განვლილი მანძილი.

საჭიროა დამატებითი ინფორმაცია:

- ობიექტის პარამეტრები და ფრენის გარემო გავლენას მოახდენს ფრენის ტრაექტორიაზე. მაგალითად ტუ ობიექტი მსუბუქია, აქვს დიდი ზედაპირული ფართი და მოძრაობს ჰაერში ჰაერის წინააღმდეგობა მნიშვნელოვნად იმოქმედეს მის რტრაექტორიაზე. გარდა ამისა, თუ ქარი ქრის, სხეული ტრაექტორია შეიცვლის. თუ ეს ინფორემაცია არ გვაქვს, უნდა დავუშვათ რომ გარემო არ მოქმედებს სხეულის გადაადგილებაზე.
- გრავიტაციული აჩქარება ასევე მოქმედებს ფრენაზე. თუ ხვა მონაცემები არ გვაქვს, ვვარაუდობთ, რომ სხეულზე მოქმედებს დედამიწის ზედაპირზე მოქმედი აჩქარება.
- უნდა შევაფასოთ საწყისი სიჩქარის და გასროლის კუთხის გაზომვის სიზუსტე, ასევე გასროლის მდებარეობაც. საიდან მოხდა გასროლა დედამოწის ზედაპირიდან თუ ადამიანის მკლავის სიმაღლიდან
- ზომის ერთეულების თანაფარდობა:
  1 მილი = 5280 ფუტი
  1 საათი = 60 წუთი = 3600 წამი
  360 გრადუსი = 2π რადიანი

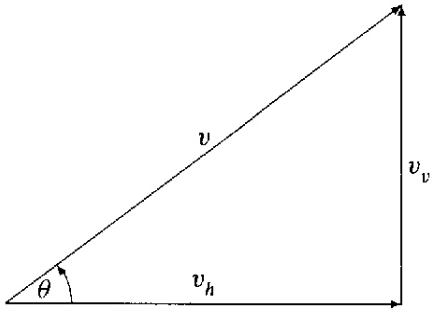
2. input/output აღწერა

- საწყისი მონაცემებია: საწყისი სიჩქარე და 50 მილი/საათში და კუთხე 30 გრადუსი ჰორიზონტალუტი მიმართულების მიმართ.
- შედეგად უნდა მივიღოთ ფრენის დრო და განვლილი მანძილი. შევარჩიოთ ზომის ერთეულებიც: წამი დროისათვის და ფუტი მანძილისათვის.

3. მათემატიკური მოდელი

შემოვიღოთ აღნიშვნები:

- დრო – t (წმ), t = 0 სხეულის გასროლის მომენტში
- საწყისი სიჩქარე = 50 მილი/საათში
- გასროლის კუთხე 30 გრადუსი
- სხეულის ჰორიზონტალური მდებარეობა x(t) (ფუტი)
- ვერტიკალური მდებარეობა - y(t) (ფუტი)
- გრავიტაციული აჩქარება g = 32.2 ft/s2, y მიმართულების საპირისპიროდ



ტრიგონომეტრიიდან ცნობილია

$V_h = V \cos \theta$
$V_v = V \sin \theta$

საწყისი სიჩქარე დავშალეთ ვერტიკალურ და ჰორიზონტალურ მდგენელებად.
წარმოვადგინოთ სიჩქარეები როგორც დროის ფუნქცია:

$$x(t) = vt\cos\theta$$

$$y(t) = vt\sin\theta - \frac{1}{2}gt^2$$

4. გამოთვლის მეთოდი

ობიექტის გასროლა ხდება როცა მისი ვერტიკალური მდებარეობა ნულის ტოლია

$$y(t) = vt\sin\theta - \frac{1}{2}gt^2$$

მას ორი ამონახსნი აქვს

$$t = 0 \qquad vt\sin\theta - \frac{1}{2}gt^2 = 0$$

მეორე ამონახსნიდან გამომდინარე ობიექტი დაეცემა დედამიწაზე დროის მომენტში:

$$t_g = \frac{2v\sin\theta}{g}$$

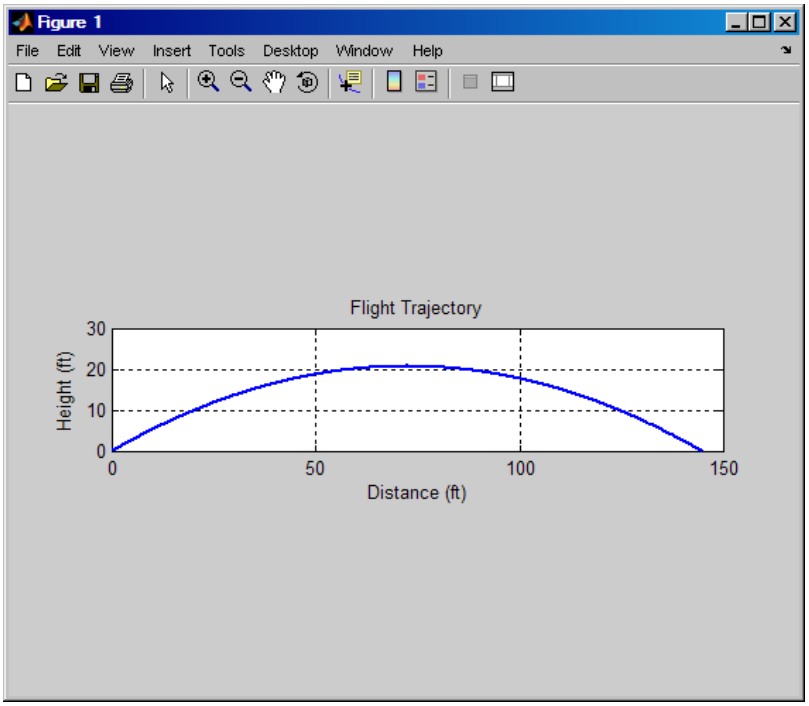
ამ დროისათვის ჰორიზონტალური მდებარეობა (განვლილი მანძილი) ტოლია:

$$x(t_g) = vt_g\cos\theta$$

5. MATLAB ამოხსნა

```
%      Flight trajectory computation
%
%     Initial values
g = 32.2; % gravity, ft/s^2
v = 50 * 5280/3600; % launch velocity, ft/s
theta = 30 * pi/180; % launch angle, radians
%     Compute and display results
disp('time of flight (s):') % label for time of flight
tg = 2 * v * sin(theta)/g % time to return to ground, s
disp('distance traveled (ft):') % label for distance
xg = v * cos(theta) * tg % distance traveled
%     Compute and plot flight trajectory
t = linspace(0,tg,256);
x = v * cos(theta) * t;
y = v * sin(theta) * t - g/2 * t.^2;
plot(x,y), axis equal, axis([ 0 150 0 30 ]), grid, ...
xlabel('Distance (ft)'), ylabel('Height (ft)'), title('Flight
Trajectory')
```

6. შემოწმება



პრობლემა – აეროდინამიკური გვირაბი

აეროდინამიკური გვირაბი – ეს არის სპეციალური შენობა სხვადასხვა სიჩქარის ქარის მოდელირებისათვის. სიჩქარის ნაცვლად შესაძლოა ვისარგებლოთ მახის რიცხვით, რომელიც

= ქარის სიჩქარე / ბგერის სიჩქარე. შენობაში თავსდება საჰაერო ხომალდის ზუსტი მოდელი და იზომება რა ძალით მოქმედებს ქარი მასზე სხვადასხვა სიჩქარისა და მოდელის მიმართ მიმართულების სხვადასხვა კუთხის შემთხვევაში. ხანგრძლივი ტესტირების შემდეგ მონაცემთა ვრცელი წყება გროვდება და მისი ანალიზის საფუძველზე დგინდება სხვადასხვა აეროდინამიკური პარამეტრი (აეროდინამიკური ამწევი ძალა-lift, აეროდინამიკური წინაღობა - drag) მოცემული მოდელისათვის.

ამ მაგალითს ჩვენ კიდევ რამდენჯერმე დავუბრუნდებით. ახლა კი დავუშვათ, რომ ტესტირებისას მიღებული მონაცემები ჩაწერილი ASCII ფაილში რომლის სახელია wind1.dat. უნდა ვნახოთ ამ მონაცემების გრაფიკი. თითოეული ხაზი ფაილში შეიცავს ფრენის მიმართულების კუთხეს გრადუსებში და შესაბამის lift კოეფიციენტს. მაგალითისათვის ავითოთ ასეთი მონაცემები:

ფრენის კუთხე (გრადუსებში)	კოეფიციენტი
-4	-0.202
-2	-0.050
0	0.108
2	0.264
4	0.422
6	0.573
8	0.727
10	0.880
12	1.027
14	1.150
15	1.195
16	1.225
17	1.224
18	1.250
19	1.245
20	1.221
21	1.177

თუმცა ვიღებთ მონაცემთა მხოლოდ მცირე ნაწილს, მაგრამ ესეც საკმარისია დავრწმუნდეთ მონაცემთა ფაილის გამოყენების უპირატესობაში. ფაილში ჩაწერილი მონაცემები მრავალგზის შეგვიძლია გამოვიყენოთ სხვადასხვა ამოცანებისათვის.

წარმოგიდგენთ ხუთ საფეხურიან პროცესს იმის საილუსტრაციოდ რომ ეს მარტივი პროცესია.

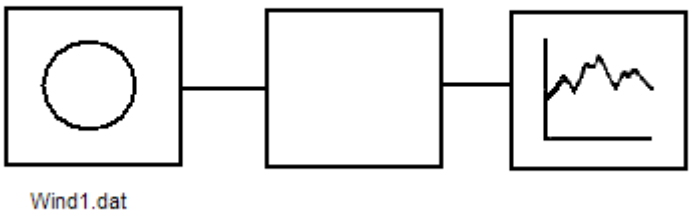
1. ამოცანის დასმა

ავაგოთ ფრენის მიმართულების კუთხის გრაფიკი კოეფიციენტის მიმართ.

2. input/output აღწერა

სადაც ეს შესაძლებელია ვისარგებლებთ I/O დიაგრამით, როგორც ეს ნაჩვენებია ნახ. 2.4. ამ მაგალითში ვკითხულობთ ინფორმაციას ფაილიდან და MATLAB საშუალებით ვაგებთ

გრაფიკს. დიაგრამა შეიცავს სიმბოლოს დისკეტისათვის, რაც წარმოადგენს მინაცემთა ფაილს – input. output არის გრაფიკული გამოსახულება და მისთვის ვიყენებთ სხვა აღნიშვნას.



ნახ. 2.4 I/O დიაგრამა

3. სახელდახელო ამოხსნა

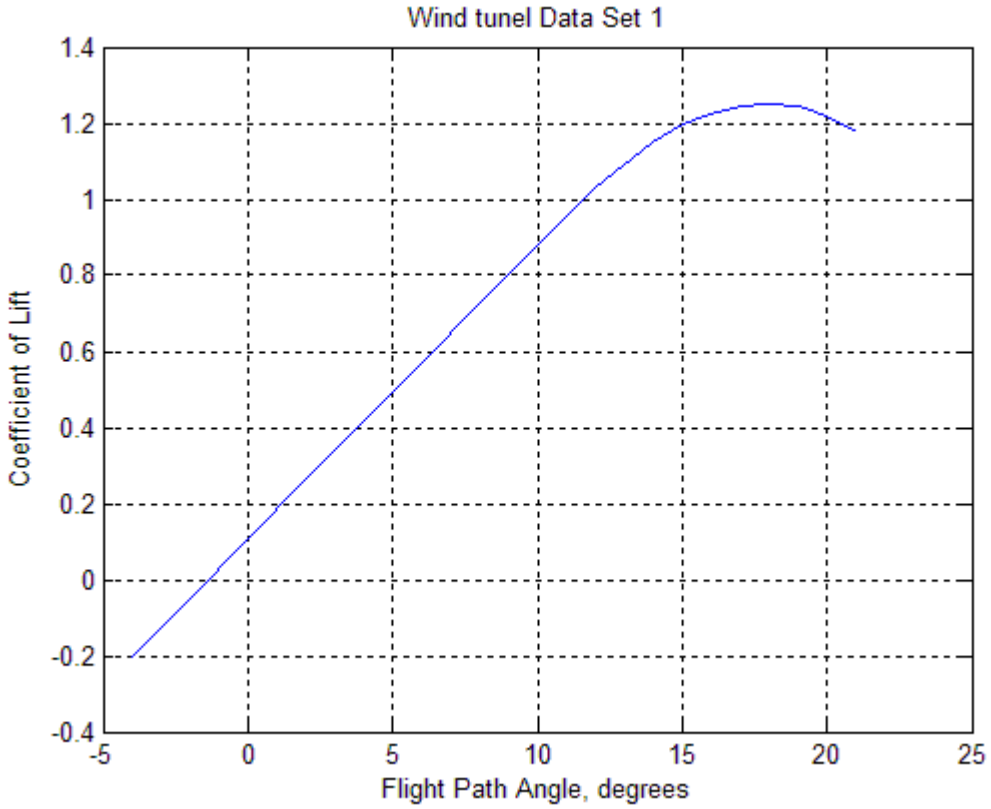
იმისათვის, რომ დაახლოებით წარმოდგენა გვქონდეს, რა შედეგი უნდა მივიღოთ, შევასრულოთ ამოცანა მცირე მასშტაბით. ამ შემთხვევაში თუ დავხედავთ მონაცემებს ვნახავთ, რომ კოეფიციეფიციენტი იზრდება –0.2 დან და აღწევს მაქსიმუმს 1.25, 18 გრადუსზე.

4. MATLAB ამოხსნა

```
%
%     This program reads and plots information
%     in data file containing flight path angles
%     and coefficients of lift.
%
load wind1.dat;       % read data from file to matrix
x = wind1(:,1);       % copy first column into x vector
y = wind1(:,2);       % copy second column into y vector
plot(x,y),...         % plot x and y
title('Wind tunel Data Set 1'),...
xlabel('Flight Path Angle, degrees'),...
ylabel('Coefficient of Lift'),...
grid
```

5. შემოწმება

თუ გავუშვებთ ამ პროგრამას MATLAB –ში მივიღებდებთ გრაფიკს. ნახ. 2.5.



ნახ. 2.5 აეროდინამიკური ექსპერიმენტის მონაცემთა გრაფიკი

ამ თავში ზოგადად წარმოგიდგინეთ MATLAB – ის გარემო. მისი ძირითადი სტრუქტურული ერთეულია მატრიცა, რომელიც შეიძლება იყოს სკალარი – ჩვეულებრივი რიცხვი, ვექტორი – რიცხვითი მწკრივი, ან რიცხვთა სტრიქონებად და სვეტებად განლაგებული ერთობლიობა. გაეცანით მატრიცის შექმნის მეთოდებს, ASCII ან MAT ფაილის სახით ჩაწერილი მონაცემების გამოძახებას, გაჩვენეთ როგორ უნდა აიგოს x-y გრაფიკი. თავის დასასულს ავაგეთ აეროდინამიკური ტესტირების შედეგად მიღებული მონაცემების გრაფიკი.

ჩამოთვლილია ყველა სპეციალური სიმბოლო, ბრძანება და ფუნქცია, რომელიც განხილული იყო ამ თავში. თითოეულ მათგანს ერთვის მოკლე განმარტება.

სპეცსიმბოლოები:

[ ]	გამოიყენება მატრიცის შესაქმნელად
( )	მატრიცის ელემენტების იდექსებისათვის
,	გამოყოფს ერთმანეთისაგან ინდექსებსა და მატრიცის ელემენტებს
;	გამოყოფს მატრიცებს ან ბრძანებებს, კრძალავს ბეჭდვას
>>	მოგვიხმობს შემდეგი ბრძანებისაკენ
. . .	აგრძელებს ბრძანებას შემდეგ ხაზზე
%	განსაზღვრავს კომენტარს ან ფორმატს
:	მატრიცის შესაქმნელად
\n	განსაზღვრავს ახალ ხაზს
^C	ბრძანების ლოკალურად შეწყვეტა

ბრძანებები და ფუნქციები

ans	ცვლადი, თუ მას შექმნისას სახელს არ მივანიჭებთ
clc	ასუფთავებს ბრძანებების ფანჯარას
clear	ასუფთავებს სამუშაო სივრცეს
clf	ასუფთავებს გრაფიკულ ფანჯარას
computer	გვიჩვენებს კომპიუტერის ტიპს
demo	იწყებს MATLAB დემონსტრირებას
disp	გამოყავს ეკრანზე მატრიცა ან ტექსტი
echo	ეკრანზე გამოგვიყვანს M ფაილის შინაარსს
exit	შეზყვეტს MATLAB - ს
format +	გამოყავს მხოლოდ ნიშანი და არა რიცხვითი გამოსახულება
format long	ათწილადი გამოყავს 15 ნიშნით მძიმის შემდეგ
format short	ათწილადი გამოყავს 5 ნიშნით მძიმის შემდეგ
format long e	რიცხვი გამოყავს ექსპონენციალური ფორმით
format short e	რიცხვი გამოყავს ექსპონენციალური ფორმით
fprintf	ბეჭდავს ფორმატირებულ ტექსტს
grid	გრაფიკულ გამოსახულებაზე გამოჰყავს საკოორდინატო ბადე
help	გამოიძახბს MATLAB help
input	საშუალებას გვაძლევს მონაცემი შევიყვანოთ კლავიატურიდან
load	გამოიძახებს მატრიცას ფაილიდან
pause	დროებით წყვეტს პროგრამას
plot	აგებს გრაფიკს
quit	წყვეტს MATLAB
save	ინახავს ცვლადს ფაილში
size	განსაზღვრავს მატრიცის ზომას
title	გრაფიკული გამოსახულების სათაური
type	დაგვიბეჭდავს ფაილის შინაარსს
what	ჩამოგვითვლის M ფაილებს მოც. სამუშაო სივრცეში
who	ჩამოთვლის ცვლადებს მოც. სამუშაო სივრცეში

whos	ჩამოთვლის ცვლადებს მოც. სამუშაო სივრცეში + ზომები
xlabel	გრაფიკზე x ღერძის სახელი
ylabel	გრაფიკზე y ღერძის სახელი

ამოცანები

1 – 10 ამოცანაში გამოიყენეთ ორწერტილოვანი ოპერატორი, რომ შექმნათ მითითებული ვექტორი და შემდეგ დაბეჭდეთ შესაბამისი ცხრილი თქვენ არ გჭირდებათ არცერთი არითმეტუკული ოპერაცია MATLAB-ში ამ პრომლემების გადასაწყვეტად, მაგრამ შესაძლოა დაგჭირდეთ ნაზრდის გამოთვლა.

1. შექმენით გრადუსების რადიანებში გადასაყვანი ცხრილი. პირველი ხაზი უნდა შიცავდეს სიდიდეებს 0 გრადუსისთვის, მეორე ხაზი – 10 გრადუსისთვის და ა. შ. ბოლო ხაზი უნდა შეიცავდეს სიდიდეებს 360 გრადუსისათვის.
2. შექმენით რადიანების გრადუსებში გადასაყვანი ცხრილი. დაიწყეთ რადიანების სვეტი 0.0 ნაზრდით $\pi/10$ 2π-მდე. (შეგახსენებთ $\pi=180^0$).
3. შექმენით დუიმების სანტიმეტრებში გადასაყვანი ცხრილი. დაიწყეთ დუიმების სვეტი 0.0 ნაზრდით 0.5, ბოლო სტრიქონი უნდა შეიცავდეს მონაცემს 20 დუიმი (1 დუიმი = 2.54 სმ).
4. შექმენით სანტიმეტრების დუიმებში გადასაყვანი ცხრილი. დაიწყეთ სანტიმეტრების სვეტი 0.0 ნაზრდით 0.5, ბოლო სტრიქონი უნდა შეიცავდეს მონაცემს 50 სმ. (1 დუიმი = 2.54 სმ).
5. შექმენით მილი/საათების ფუტი/წმ.-ში გადასაყვანი ცხრილი. დაიწყე მილი/საათების სვეტი 0.0 ნაზრდით 5 მილი/საათი ბოლო სტრიქონი უნდა შეიცავდეს მონაცემს 65 მილი/საათი (1 მილი = 5.280 ფუტს)
6. შექმენით ფუტი/წმ. მილი/საათებში გადასაყვანი ცხრილი. დაიწყეთ ფუტი/წმ. სვეტი 0.0 ნაზრდით 5 ფუტი/წმ. ბოლო სტრიქონი უნდა შეიცავდეს მონაცემს თქვენს მიერ შერჩეულ სიდიდეს (1 მილი = 5.280 ფუტს)

შემდეგი პირობები ვრცელდება 7 – 10 ამოცანებზე:

$ = 5.3 ფრანკი

1 იენი = 0.0079 $

1.57 გერმანული მარკა = 1 $

7. შექმენით ფრანკების დოლარებში გადასაყვანი ცხრილი. დაიწყე ფრანკების სვეტი 5ფრანკით, ნაზრდით 5 ფრანკი. დაბეჭდეთ ასეთი ცხრილის 25 სტრიქონი.
8. შექმენით გერმანული მარკის ფრანკში გადასაყვანი ცხრილი ცხრილი. დაიწყე მარკების სვეტი 1გერმანული მარკით, ნაზრდით 5 გ.მ. მიეცი საშუალება მომხმარებელს თვითონ შეიყვანოს სტრიქონების რიცხვი კლავიატურიდან.
9. შექმენით იენის გერმანულ მარკებში გადასაყვანი ცხრილი. დაიწყე იენის სვეტი 100 იენით და დაბეჭდეთ ასეთი ცხრილის 25 სტრიქონი, ისე, რომ იენის საბოლოო მნიშვნელობა იყოს 10 000.

10. შექმენით დოლარის ფრანკებში, გერმანულ მარკაში და იენში გადასაყვანი ცხრილი. დაიწყე დოლარის სვეტი 1 დოლარით. დაბეჭდეთ ასეთი ცხრილის 50 სტრიქონი.

11-15 ამოცანები მომართავენ მონაცემთა ASCII ფაილს სახელით temp.dat . მონაცემთა ფაილი შეიცავს 100 სტრიქონს. თითოეული სტრიქონი შეიცავს ახალი ძრავის ტესტირების შედეგებს. პირველი სიდიდე სტრიქონში(0.0) არის დრო, როცა ძრავამ მუშაობა დაიწყო. დროის ყოველ მოცემულ მომენტში იზომებოდა ტემპერატურა ძრავის 4 სხვადასხვა ადგილას. ამრიგად, ყოველი სტრიქონი შეიცავს 5 მონაცემს: დრო და 4 ტემპერატურა (T1, T2, T3, T4).

11. ააგეთ გარფიკი, სადაც x ღერძზე გადაზომილია დრო, y ღერძზე კი T1. დააწერეთ ღერძებს შესაბამისი აღნიშვნები.
12. ააგეთ გარფიკი, სადაც x ღერძზე გადაზომილია დრო, y ღერძზე კი T2. დააწერეთ ღერძებს შესაბამისი აღნიშვნები.
13. ააგეთ გარფიკი, სადაც x ღერძზე გადაზომილია დრო, y ღერძზე კი T3. დააწერეთ ღერძებს შესაბამისი აღნიშვნები.
14. ააგეთ გარფიკი, სადაც x ღერძზე გადაზომილია დრო, y ღერძზე კი T4. დააწერეთ ღერძებს შესაბამისი აღნიშვნები.
15. ააგეთ გარფიკი, სადაც x ღერძზე გადაზომილია T4, y ღერძზე კი დრო.