კოსმოსური თანამგზავრი VIKING 1

ფუნდამენტური საინჟინრო გამოთვლები

წარმოგიდგენთ მასალას, რომელიც ფუნდამენტურია ძირითადი საინჟინრო ამოცანებისათვის. მაგალითად MATLAB-ის ფუნქციები ტრიგონომეტრიული, ლოგარითმული და ექსპონენციალური გამოთვლებისათვის. ასევე განვიხილავთ კომპლექსურ რიცხვებს, რომელიც განსაკუთრებით მნიშვნელოვანია ელექტრო და მექანიკურ საინჟინრო დისციპლინებში. გარდა იმისა, რომ ვაწარმოოთ რიცხვითი ოპერაციები, უნდა შეგვეძლოს შედარების, ლოგიკური და ციკლის ოპერაციების ჩართვა პროგრამაში. ამიტომ თავში კონტროლის ოპერაციების შესახებ განხილულია საკითხი იმის შესახებ თუ როგორ უნდა დაისვას შეკითხვა პროგრამაში და პასუხზე დაყრდნობით როგორ უნდა შესრულდეს გარკვეული ოპერაციები. თავი სტსტისტიკური განაზომებისა და მათი ინტერპრეტაციის შესახებ ეხება ექსპერიმენტული შედეგების ანალიზს. განვიხილავთ შემთხვევითი სიდიდის გენერირების ფუნქციებს, რომლებიც დაგვჭირდება გარკვეული ფიზიკური პროცესების მოდელირებისათვის. MATLAB-ს ასევე გააჩნია დამატებითი ფუნქციები და ოპერტორები, რომლებიც მატრიცებისთვისაა განკუთვნილი. და ბოლოს, იმის გამო, რომ საინჟინრო კონცეფციათა არსის გაგება ბევრადაა დამოკიდებული მათ ვიზუალურ წარმოდგენაზე, ძალზე მნიშვნელოვანია მარტივად და მოხერხებულად შევძლოთ მათი წარმოჩენა. MATLAB–ს გააჩნია მძლავრი საშუალებები გრაფიკული ინტერპრეტაციისათვის.

3 სკალარული და მატრიცული გამოთვლები

შესავალი

შეკრების, გამოკლების, გამრავლების და გაყოფის ოპერაციები ძირითადი არითმეტიკული ოპერაციებია, რომელოთაც ყველაზე ხშირად ვსარგებლობთ. ასევე ხშირად ვასრულებთ ისეთ ოპერაციებს, როგორიცაა კვადრატული ფესვის ამოღება, რიცხვის ლოგარითმის ან კუთხის ტრიგონომეტრიული ფუნქციის გამოთვლა. შესაძლებელია ეს ოპერაციები შესრულდეს როგორც ერთი სიდიდისათვის ასევე მათი ერთობლიობისათვისაც – ვექტორის და მატრიცისათვის. ამ თავში გავეცნობით, როგორ ახორციელებს MATLAB ამ ოპერაციებს. გარდა ამისა, ვნახავთ, როგორ გამოვიყენოთ MATLB–ის შესაძლებლობები კომპლექსური რიცხვებისათვის. ერთ-ერთ პრაქტიკულ მაგალითში განვიხილავთ საკომუნიკაციო სიგნალის ექოს, მეორე კი ეხება ჰიდროაკუსტიკურ სიგნალს.

3.1 MATLAB სპეციალური სიდიდეები და მატრიცები

MATLAB-ს აქვს გარკვეული რაოდენობის წინასწარგანსაზღვრული მუდმივები, სპეციალური სიდიდეები და სპეციალური მატრიცები. მათი უმრავლესობა შექმნილია MATLAB-ის მიერ თავისივე ფუნქციების გამოყენებით. ამ თავში გაგაცნობთ MATLAB ფუნქციათა რამდენიმე მაგალითს. ხოლო 3.4 განყოფილებაში მიმოვიხილავთ MATLAB-ის დამატებითი ფუნქციების რამდენიმე მაგალითს.

3.1.1 სპეციალური სიდიდეები

pi	MATLAB ამ სახელიან ცვლადს აღიქვამს როგორც ჩვეულბრივ რიცხვს π.
i, J	გამოიყენება ლომპლექსული რიცხვებისათვის და MATLAB-ი მათ აღიქვამს როგორც √-1
inf	ეს ცვლადი აღნიშნავს უსასრულობას, რომელიც, ჩვეულბრივ, გამოთვლებისას მიიღება როგორც ნულზე გაყოფის შედეგი (∞)
NaN	წარმოადგენს სიდიდეს, რომელიც არ არის რიცხვი – განუზღვრელობა, სისდიდე, რომელიც მიიღება მაგალითად ნულის გაყოფით ნულზე.
clock	გვაძლევს მიმდინარე დროს ექვსელემენტიანი ვექტორის სახით: წელი, თვე, საათი, წუთი და წამი.
date	გვაძლევს მიმდინარე თარიღს სიმბოლოთა მწკრივის სახით მაგ. 20-Jun-2006
eps	მოცემული კომპიუტერის სიზუსტე (რიცხვის მოცემის) მცოცავი მძიმის რეჟიმში. ეს არის სხვაობა 1.0- სა და მის მომდევნო უახლოეს ათწილადს შორის, რომლის წარმოება მოც. კომპიუტერს შეუძლია.
ans	გამოიყენება იმ ცვლადის აღსანიშნავად, რომელის სახელიც განსაზღვრული არ არის.

3.1.2 სპეციალური მატრიცები

MATLAB აქვს ფუნქციათა ჯგუფი, რომლის საშუალებით სპეციალური ტიპის მატრიცები იქმნება, რომელთაც გარკვეული დანიშნულება აქვთ რიცხვითი მეთოდების გამოყენებაში. ზოგირთ მათგანს კი უფრო ზოგადი გამოყენება აქვს.

მაგიური კვადრატი. n რიგის მაგიური კვადრატი არის n×n მატრიცა, რომლს ელემენტებია მთელი რიცხვები 1 დან n^2 – მდე. მისი ელემენტები ისეა განლაგებული, რომ ყოველ სტრიქონში და სვეტში ელემენტების ჯამი ერთიდაიგივე რიცხვია.

ბრძანება **zeros** აწარმოებს მატრიცას, რომლის ყველა ელემენტი ნულის ტოლია. თუ ფუნქციის არგუმენტი სკალარია, მიიღება კვადრეტული მატრიცა, რომლის სვეტების და სტრიქონების რიცხვი არგუმენტით განისაღზღვრება. თუ ფუნქციას ორი სკალარული არგუმენტი ქვს, (**zeros**(m,n)), მიღება n სტრიქონიანი და m სვეტიანი მატრიცა, რომლის ყველა ელემენტი ნულის ტოლია. თუ არგუმენტი მატრიცაა, მაშინ შეიქმნება ისეთივე ზომის მატრიცა, როგორიც არგუმენტია. შემდეგი ბრძანებები შეესაბამება აღწერილ შემთხვევებს:

```
A = zeros(3);
B = zeros(3,2);
C = [ 1 2 3; 4 5 6];
D = zeros( size(C));
```

$$A = \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix} \qquad B = \begin{matrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{matrix}$$

$$C = \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{matrix} \qquad D = \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix}$$

ფუნქცია **ones** აწარმოებს მატრიცას, რომლის ყველა ელემენტი 1-ის ტოლია, ისევე, როგორც **zeros** აწარმოებს 0 –ების შემცველ მატრიცას. თუ ფუნქციის არგუმენტი სკალარია, მიიღება კვადრეტული მატრიცა, რომლის სვეტების და სტრიქონების რიცხვი არგუმენტით განისაღზღვრება. თუ ფუნქციას ორი სკალარული არგუმენტი ქვს, (**ones**(m,n)), მიღება n სტრიქონიანი და m სვეტიანი მატრიცა, რომლის ყველა ელემენტი ერთის ტოლია. თუ არგუმენტი მატრიცაა, მაშინ შეიქმნება ისეთივე ზომის მატრიცა, როგოროც არგუმენტია. შემდეგი ბრძანებები შეესაბამება აღწერილ შემთხვევებს:

```
A = ones(3);
B = ones(3,2);
C = [ 1 2 3; 4 5 6];
D = ones(size(C));
```

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \qquad D = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

ერთეულოვანი მატრიცა ისეთი მატრიცაა, რომლის მთავარი დიაგონალის ელემენტები 1-ის ტოლია, ყველა დანარჩენი კი ნულის. მაგ. შემდეგი მატრიცა არის 4×4 ერთეულოვანი მატრიცა:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

აღვნიშნავთ, რომ მთავარი დიაგონალი არის ის დიაგონალი, რომლის ელემენტების ინდექსებიც ერთმანეთის ტოლია – (1,1), (2,2), (3,3) და ა.შ.

MATLAB – ში ერთეულოვანი მატრიცა შეგვიძლია შევქმნათ **eye** ფუნქციის საშუალებით. ამ ფუნქციის არგუმენტი მსგავსია **ones** და **zeros** ფუნქციათა არგუმენტების. თუ ფუნქციის არგუმენტი სკალარია, მიიღება კვადრეტული მატრიცა, რომლის სვეტების და სტრიქონების რაოდენობა არგუმენტით განისაზღვრება. თუ ფუნქციას ორი სკალარული არგუმენტი ქვს, (**eye**(m,n)), მიიღება n სტრიქონიანი და m სვეტიანი მატრიცა, რომლის მთავარი დიაგონალის ელემენტები ერთის ტოლია, ყველა დანარჩენი კი ნულის. თუ არგუმენტი მატრიცაა, მაშინ შეიქმნება იგივე ზომის ერთეულოვანი მატრიცა. შემდეგი ბრძანებები შეესაბამება აღწერილ შემთხვევებს:

```
A = eye(3);
B = eye(3,2);
C = [ 1 2 3; 4 5 6];
D = eye( C );
```

$$A = \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix} \qquad B = \begin{matrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{matrix}$$

$$C = \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{matrix} \qquad D = \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{matrix}$$

სასურველია ერთეულოვან მატრიცას არასოდეს დავარქვათ i, ამან შესაძლოა გამოიწვიოს პრობლემები თუ იგივე პროგრამაში კომპლექსურ რიცხვებს იყენებთ.

პასკალის სამკუთხედი. ბრძანება **pascal** აწარმოებს კვადრატულ მატრიცას, რომლის ელემენტები პასკალის სამკუთხედს შეესაბამება. პასკალის სამკუთხედი გვაძლევს ბინომიალური გაფართოების კოეფიციენტებს ფორმით: $(a + b)^n$. მაგალითად მეოთხე ხარისხის სამკუთხედი:

$$\begin{matrix} & & & & 1 & & & & \\ & & & 1 & & 1 & & & \\ & & 1 & & 2 & & 1 & & \\ & 1 & & 3 & & 3 & & 1 & \\ 1 & & 4 & & 6 & & 4 & & 1 \end{matrix}$$

თუ მივცემთ ბრძანებას **pascal**(5), მივიღებთ მატრიცას:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 3 & 6 & 10 & 15 \\ 1 & 4 & 10 & 20 & 35 \\ 1 & 5 & 15 & 35 & 70 \end{bmatrix}$$

თუ კარგად დააკვირდებით ამ მატრიცაში ადვილად იპოვით პასკალის სამკუთხედს.

3.2 სკალარული ოპერაციები

არითმეტიკული ოპერაციები ხორციელდება გამოსახულებათა საშუალებით. გამოსახულება შესაძლოა იყოს მარტივი, როგორიცაა მაგალითად მუდმივა - კონსტანტა, ანდა შეიცავდეს მატრიცებს და კონსტანტებს, შეერთებულს არითმეტიკული ოპერაციების ნიშნებით. ამ თავში განვიხილავთ მხოლოდ სკალარული სიდეების შემცველ გამოსახულებებს.

ცხრილი აღწერს არითმეტიკულ ოპერაციებს საკალარებს შორის.

ოპერაცია	ალგებრული ფორმა	MATLAB
შეკრება	a + b	a + b
გამოკლება	a - b	a - b
გამრავლება	a x b	a * b
მარჯვენა გაყოფა	a/b	a/b
მარცხენა გაყოფა	b/a	a\b
ახარისხება	a^b	a^b

გამოსახულება: x = a + b, a და b მნიშვნელობბები შეიკრიბება და მიღებული შედეგი მიენიჭება x –ს. ხოლო გამოსახულება count = count + 1 – ნიშნავს რომ count მნიშვნელობას მიემატება 1 და შედეგი ჩაიწერწბა ისევ count ცვლადში, შეცვლის რა მის წინა მნიშვნელობას, ე. ი. count მნიშვნელობა 1 –ით გაიზრდება.

მნიშვნელოვანია აღინიშნოს, რომ ცვლადს არ შეიძლება ჰქონდეს ერთდროულად ორი მნიშვნელობა. მაგალითად ერთმანეთის მომდევნო ეს ორი ბრძანება:

```
time = 0.0;
time = 5.0;
```

შედეგად გვაძლევს time = 5.0.

3.1.3 არითმეტიკული ოპერაციების შესრულების რიგი

რამდენადაც რამდენიმე არითმეტიკული ოპერაცია შეიძლება გაერთიანდეს ერთი გამოსახულების სახით, მნიშვნელოვანია დავიცვათ მათი შესრულების რიგი. ცხრილი გვიჩვენებს არითმეტიკული ოპერაციების რიგითობას MATLB გამოსახულებაში, რაც ისეთივეა, როგორც ჩვეულებრივ ალგებრული ოპერაციების რიგი.

პროცედურა	ოპერაცია
1	ფრჩხილები
2	ახარისხება მარცხნიდან მარჯვნივ
3	გამრავლება და გაყოფა, მარცხნიდან მარჯვნივ
4	შეკრება – გამოკლება, მარცხნიდან მარჯვნივ

დავუშვათ გვსურს გამოვთვალოთ ტრაპეციის ფართობი, სადაც სკალარი height შეიცავს სიმაღლის სიგრძეს, ხოლო სკალარები base_1 და base_2 - მის ორ ფუძეს. ტრაპეციის ფართობი შეგვიძლია გამოვთვალოთ შემდეგი ბრძანებებით:

area = 0.5* height *(base_1 + base_2);

თუ გამოგვრჩა ფრჩხილები:

area = 0.5* height *base_1 + base_2;
მაშინ ეს ბრძანება იდენტური იქნება ბრძანების:

area = (0.5* height *base_1) + base_2;
და შედეგიც შესაბამისად არასწორი მოჰყვება, თუმცა პროგრამა ვერ განსაზღვრავს რომ შეცდომაა დაშვებული და არც მიგვითითებს ამის თაობაზე, ამიტომ პროგრამის შედგენისას სიფრთხილე უნდა გამოვიჩინოთ, რათა უზრუნველვყოთ ატითმეტიკული მქმედებების სათანადო რიგი. განვიხილოთ განტოლება:

$$f = \frac{x^3 - 2x^2 + x - 6.3}{x^2 + 0.05005x - 3.14}$$

f მნიშვნელობა უნდა გამოვითვალოთ შემდეგი ბრძანებებით:

numerator = x^3 – 2*n^2 + x – 6.3;
denumerator = x^2 + 0.05005 * x – 3.14;
f = numerator/denumerator;

უკეთესია ბრძანებები ასეთნაირად გავყოთ, ვიდრე ერთი გრძელი გამისახულება დავწეროთ, რომელიც მათ გააერთიანებს. ეს უფრო მარტივიცაა და შეცდომებსაც აგვაცილებს თავიდან

სავარჯიშო

დაწერეთ ბრძანებები შემდეგი გამოსახულებების გამოსათვლელად:

1. ხახუნის კოეფიციენტი საბურავსა და ქვაფენილს შორის:

$$\text{friction} = \frac{v^2}{30s}$$

2. correction faqtor(შესწორების კოეფიციენტი) წნევის გამოთვლისას:

$$factor = 1 + \frac{b}{\upsilon} + \frac{c}{\upsilon^2}$$

3. ორ წერტილს შორის დახრა:

$$slope = \frac{y_2 - y_1}{x_2 - x_1}$$

4. პარალელური წრედის წინაღობა:

$$resis\tan ce = \frac{1}{\frac{1}{r_1} + \frac{1}{r_2} + \frac{1}{r_3}}$$

3.1.4 შეზღუდვები გამოთვლებში

კომპიუტერში ცვლადმა შეიძლება მიიღოს რიცხვითი მნიშვნელობა ფართო დიაპაზონში, ხშირ შემთხვევაში ეს არის 10^{-308} – 10^{308} შუალედი. უმრავლეს შემთხვევაში ეს საკმარისია. თუმცა ზოგჯერ შედეგმა შესაძლოა გადააჭარბოს აღნიშნულ ზღვარს. დავუშვათ გვაქვს:

```
x = 2.5e200;
y = 1.0e200;
z = x*y;
```

x და y ზემოაღნიშნულ შუალედში ხვდებიან, მაგრამ z მას აჭარბებს.
ასეთ შეცდომას უწოდებენ ექსპონენციალურ გადაკავებას (overflow). MATLAB ასეთ ცვლადს მიანიჭებს მნიშვნელობას ∞.
ასევე შეიძლება მივიღოთ ძალიან მცირე რიცხვი – ექსპონენციალური უკმარისობა (underflow),

```
x = 2.5e-200;
y = 1.0e200;
z = x/y;
```

ამ შემთხვევაში z სიდიდე მიიღებს ნულოვან მნიშვნელობას.

3.3 ოპერაციები მასივებზე

მასივებზე ოპერაცია ხორციელდება შესაბამის ელემენტებს შორის. მაგალითად, დავუშვათ A არის ხუთელემენტიანი სტრიქონი ვექტორი. B ასევე ხუთელემენტიანი სტრიქონი ვექტორია. იმისათვის, რომ შეიქმნას ვექტორი, რომლის ელემენტებიც ამ ორი ვექტორის ელემენტების ნამრავლი იქნება, ერთი გზა ასეთია:

```
C(1) = A(1) * B (1);
C(2) = A(2) * B (2);
C(3) = A(3) * B (3);
C(4) = A(4) * B (4);
C(5) = A(5) * B (5);
```

ეს ოპერაციები სკალარულია, რადგან შეიცავს მხოლოდ სკალარულ სიდიდეებს.

თუ გვსურს ორი ერთნაირი ზომის ვექტორი ან მატრიცა ერთმანეთზე წევრ-წევრად გადავამრავლოთ, მოქმედების ნიშნის წინ უნდა დავწეროთ წერტილი:

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

წერტილის გამოტოვება ამ ოპერაციაში სერიოზული შეცდომაა, რადგან ასეთ შემთხვევაში მოხდება მატრიცული გადამრავლება, რომელიც როგორც ვიცით, სრულიადაც არ ნიშნავს ელემენტების ერთმანეთზე წევრ-წევრად გადამრავლებას.

შეკრებისა და გამოკლებისათვის მატრიცული და მასივებზე ოპერაციები ერთიდაიგივეა, ხოლო გამრავლების გაყოფის, ახარისხების მატრიცული ოპერაციები განსხვავდება მასივებზე ოპერაციებისაგან, ამიტომ არ უნდა დაგვავიწყდეს მოქმედების ნიშნის წინ წერტილის დასმა, როცა ეს საჭიროა (მასივებზე ოპერაციის დროს). ეს წესები შეჯამებულია ცხრილში:

ოპერაციები შესაბამის ელემენტებზე

ოპერაცია	ალგებრული ფორმა	MATLAB
შეკრება	a + b	a + b
გამრავლება	a – b	a - b
გამრავლება	a x b	a. * b
მარჯვენა გაყოფა	a/b	a./b
მარცხენა გაყოფა	b/a	a.\b
ახარისხება	a^b	a.^b

```
B = 3*A; - თუ  A სკალარია,
B = 3.*A; - თუ  A არ არის სკალარი.
```

შედეგად მიღებული ვექტორი იგივე ზომის იქნება, როგორიც მოქმედებაში მონაწილე ვექტორები.

```
A = [2 5 6]
B = [2 3 5]
```

A ,B მასივური ნამრავლი:

```
C = A. *B  = [4 15 30].
```

MATLAB აქვს გაყოფის ორგვარი ოპერაცია: მარჯვენა და მარცხენა. (/, \).

C = A./B – მარჯვენა გაყოფა ხორციელდება A მატრიცის ელემენტების გაყოფით B- მატრიცის შესაბამის ელემენტებზე. ხოლო მარცხნიდან გაყოფა – B ელემენტების გაყოფით A ელემენტებზე.

მასივური ახარისხებაც შესაბამისად წევრ-წევრად ხდება. მაგალითად:

```
C = A.^2;
D = A.^B;
```

შესაბამისად მოგვცემს:

```
C = [4   25  36]
D = [4   125   7776]
```

შეგვიძლია ახარისხებისას ფუძედ სკალარი ავიღოთ, მაჩვენებლად – ვექტორი:

```
C = 3.0.^ A – მოგვცემს C = [9 243 729].
```

ექვივალენტურია ბრძანება:

```
C = (3).^A;
```

ამგვარად დიდი სიფრთხილეა საჭირო რომ სწორად განვსაზღვროთ და მივუთითოთ პროგრამაში რა ტიპის ოპერაცია გვჭირდება მატრიცული თუ მასივური.

მოყვანილ მაგალითებში ვიხილავდით ვექტორებს. ყოველივე რაც მათ შესახებ ითქვა, ეხება მატრიცებსაც.

```
d = [1:5; -1 : –1 : -5];
z = ones(d);
s = d – z;
p = d.*s;
sq = d. ^3;
```

მიიღება:

$$d = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ -1 & -2 & -3 & -4 & -5 \end{bmatrix} \qquad z = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$s = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ -2 & -3 & -4 & -5 & -6 \end{bmatrix} \qquad p = \begin{bmatrix} 0 & 2 & 6 & 12 & 20 \\ 2 & 6 & 12 & 20 & 30 \end{bmatrix}$$

$$sq = \begin{bmatrix} 1 & 8 & 27 & 64 & 125 \\ -1 & -8 & -27 & -64 & -125 \end{bmatrix}$$

სავარჯიშო

გამოითვალეთ C ვექტორის მნიშვნელობები, თუ A = [2 –1 5 0] B = [3 2 –1 4]

1. C = A- B;
2. C = B +A – 3;
3. C = 2*A + A.^B;
4. C = B. /A;
5. C = B. \A;
6. C = A. ^ B;
7. C = (2).^B +A;
8. C = 2*B/3.0.*A;

ამოცანა - საკომუნიკაციო სიგნალის ექო

შესრულებული იქნა საინტერესო გამოკვლევა კომპიუტერული სისტემის განვითარებისათვის, რომელიც ემყარება ხმოვან ბრძანებებს. ასეთი სისტემა გულისხმობს, რომ მიკროფონი, რომლითაც ბრძანებები გადაიცემა რაც შეიძლება ზუსტად გადასცემდეს მეტყველებას. სამწუხაროდ სენსორებს, მათ შორის მიკროფონსაც ახასიათებს ხმაური. ორმხრივი კომუნიკაციის სისტემებს ახასიათებს აგრეთვე ექო. ვიდრე შეუდგებოდეს სიტყვების

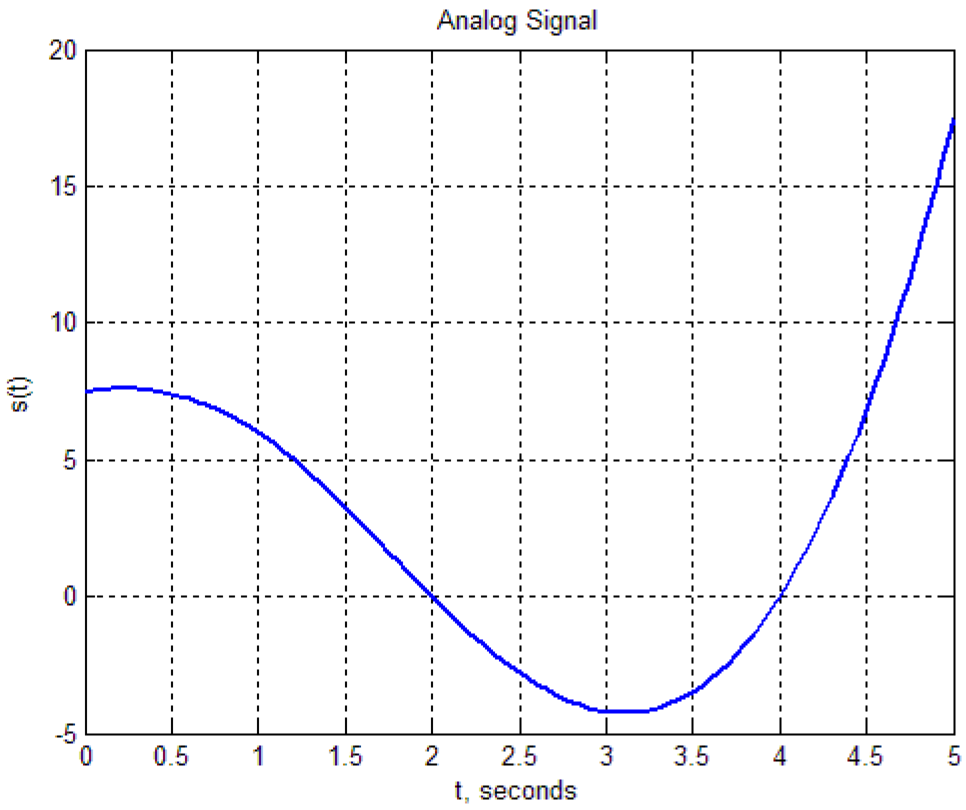
გაიგივებას, მეტყველების გამაიგივებელ სისტემას უნდა შეეძლოს გადაამუშაოს სიგნალი და გაასუფთაოს იგი არასასურველი კომპონენტებისაგან, როგორიცაა მაგალითად ექო. იმისათვის, რომ შევამოწმოთ პროგრამა, რომელმაც უნდა გაასუფთაოს სიგნალი ექოსაგან, გვჭირდება შევქმნათ ციფრული სიგნალი და მას დავურთოთ ექოები. შემდეგ მიღებულ სიგნალს დავამუშავებთ გამოსაცდელი პროგრამის საშუალებით და ვნახავთ რამდენად კარგად შევძელით დასახული ამოცანის შესრულება. განვსაზღვროთ ციფრული სიგნალი და დავწეროთ MATLAB პროგრამა, რომელიც მას დაუმატებს რამდენიმე ექოს.

ციფრული სიგნალი

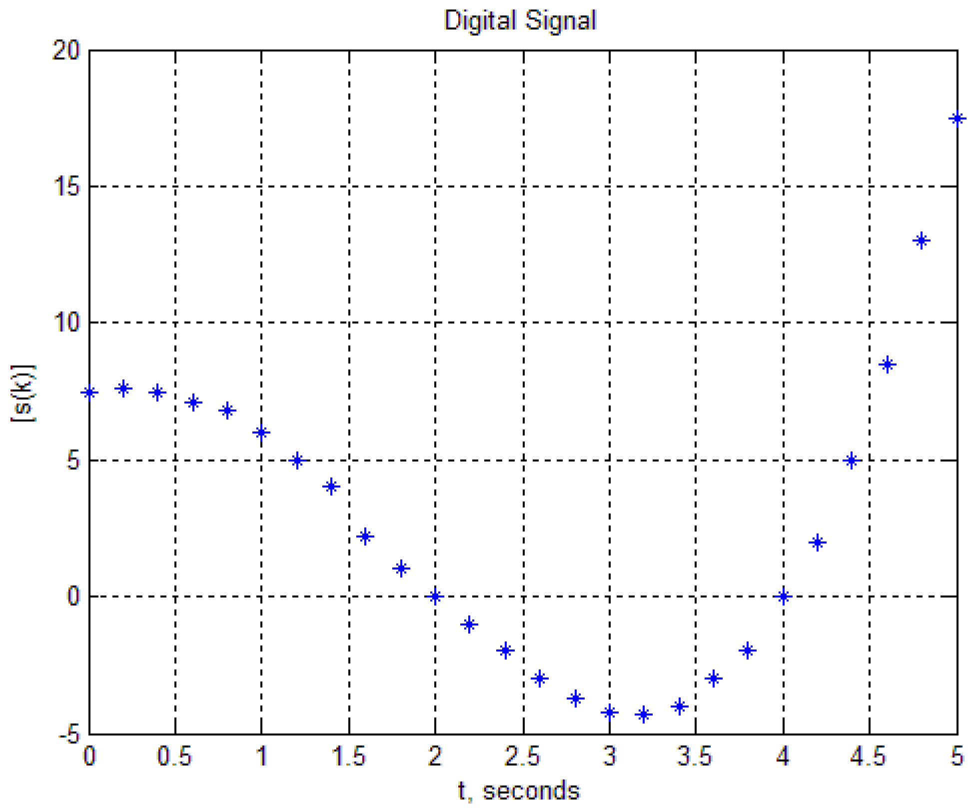
სიგნალი არის ფუნქცია (ჩვეულებრივ დროისა) რომელიც გვაწვდის ინფორმაციას. ინფორმაცია, ანუ მონაცემები შეკრებილია სენსორის მიერ. ჩვეულებრივ ასეთი მიმღებია მიკროფონი, რომელიც ზომავს აკუსტიკურ ან ხმოვან სიდიდეებს(მაგალითად მეტყველება), სეისმომეტრი, რომელიც ზომავს დედამიწის ქერქის მოძრაობას, ფოტომიმღები, რომელიც ზომავს სინათლის ინტენსივობას, თერმომეტრი- ტემპერატურას, ვოლტმეტრი – ძაბვას. სენსორი ჩვეულებრივ მიერთებულია სხვა ინსტრუმენტთან, ე. წ. ანალოგურ-ციფრულ გარდამქმნელთან(A/D), რომელიც აგროვებს სიგნალს პერიოდულად, ჩაიწერს დროიით ანათვალს და სიგნალის შესაბამის მნიშვნელობას მონაცემთა ფაილის სახით. საწყისი, შემომავალი სიგნალი ჩვეულებრივ უწყვეტი (ანალოგური) ფუნქციაა. ამგვარად მიღებულ მონაცემთა მწკრივს ციფრულ სიგნალს უწოდებენ. ნახ 3.1 წარმოადგენს უწყვეტი სიგნალის მაგალითს, 3.2 - მონაცემთა მწკრივს, რომელიც მიიღება ანალოგური სიგნალიდან - ციფრულ სიგნალს. ციფრული სიგნალი შეიცავს მონაცემთა მწკრივს, რომელიც შეიძლება ჩაიწეროს ფაილის სახით და შემდეგ გამოვიყენოთ MATLAB –ში დაწერილ პროგრამაში, ავაგოთ გრაფიკი სიგნალის მნიშვნელობებისა დროის მიმართ. გრაფიკის აგებისას წერტილებს ერთმანეთთან ვაერთებთ სეგმენტებით, რომ შედეგი უფრო თვალსაჩინო იყოს.

ექოს გენერირება

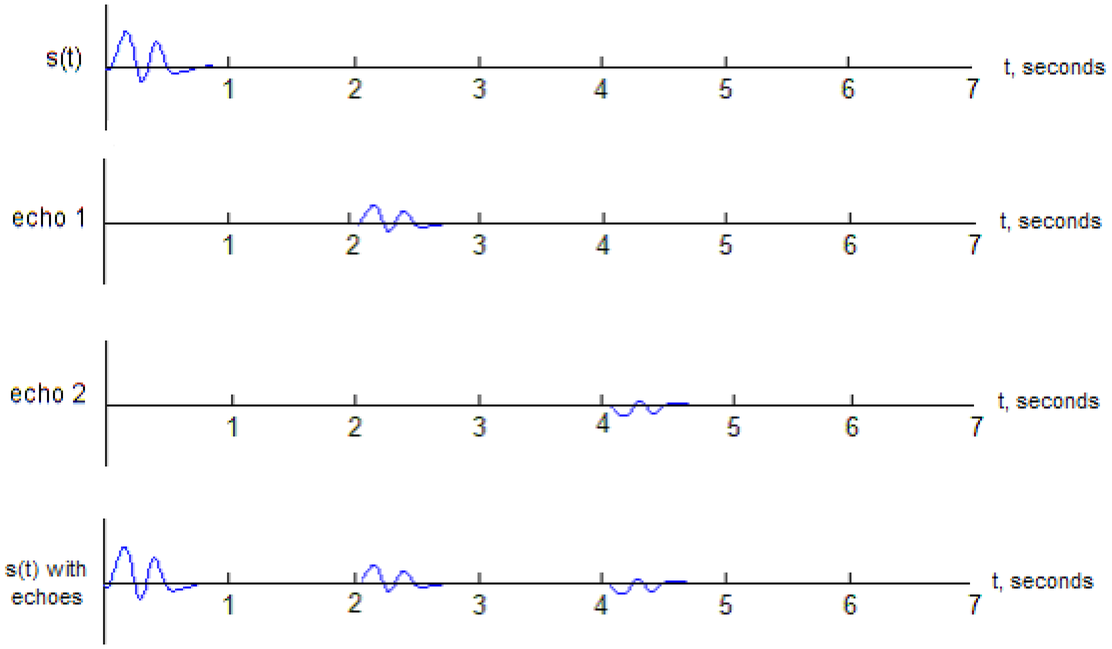
ექო განიხილება როგორც შესუსტებული ანარეკლი ორიგინალური სიგნალისა, რომელიც სიგნალს დაერთვის გარკვეული შეყოვნებით. მაგალითად ნახ 3.3 შეიცავს ორიგინალურ სიგნალს, s(t), მეორე გრაფაში სიგნალის მნიშვნელობა შესუსტებულია 0.5 ჯერ და 2 წამის დაგვიანებით დაერთვის სიგნალს. მესამე გრაფაში საიგნალს ერთვის ექო, რომელიც შესუსტებულია 0.3 ჯერ და დაგვიანებული 5 წამით. ეს ე.წ. ჩახვეული (rolled) ექოა,რადგან ექოს მნიშვნელობები უარყოფითი სიდიდეებია. მეოთხე გრაფაში წარმოდგენილია ორიგინალური სიგნალი თანდართული ორივე ექოთი.



ნახ. 3.1 ანალოგური სიგნალი



ნახ. 3.2 ციფრული სიგნალი



ნახ. 3.3 სიგნალი ექოებითურთ

დავუშვათ მონაცემები სიგნალის შესახებ გროვდებოდა 10 წამის განმავლობაში, ყოველ 0.1 წამში. პირველი წამის განმავლობაში შეგროვდა კოორდინატთა შემდეგი წყება:

დრო, წმ	სიგნალის მნიშვნელობა
00	0.0
0.1	0.5
0.2	1.0
0.3	-0.5
0.4	0.75
0.5	0.0
0.6	-0.02
0.7	-0.10
0.8	0.0
0.9	0.0
1.0	0.0

სიგნალის შემდგომი მნიშვნელობები 0-ის ტოლია.

დაწერეთ MATLAB პროგრამა, რომელიც შექმნის სიგნალს, რომელიც შედგება საწყისი სიგნალისა და დართული სამი ექოსაგან. პირველი ექო – სიგნალი შესუსტებულია ფაქტორით - 0.5 და დაგვიანებული 2 წამით, მეორე ექო – ჩახვეული(rolled) ექო 4 წამის დაგვიანებით და შესუსტებული ფაქტორით -0.3, მესამე ექო დაგვიანებულია 7.5 წამით და შესუსტებული ფაქტორით 0.1. ააგეთ საწყისი სიგნალი და ახალი სიგნალი სამივე ექოთი. შემდეგ შეინახეთ ორივე სიგნალის მონაცემები M ფაილის სახით – echo.m.

1. ამოცანის დასმა

მოცემულია საწყისი სიგნალი, შევქმნათ ახალი სიგნალი, რომელიც შეიცავს როგორც საწყის სიგნალს, ასევე სამ სხვადასხვაგვარად შესუსტებულ და დაგვიანებულ ექოს.

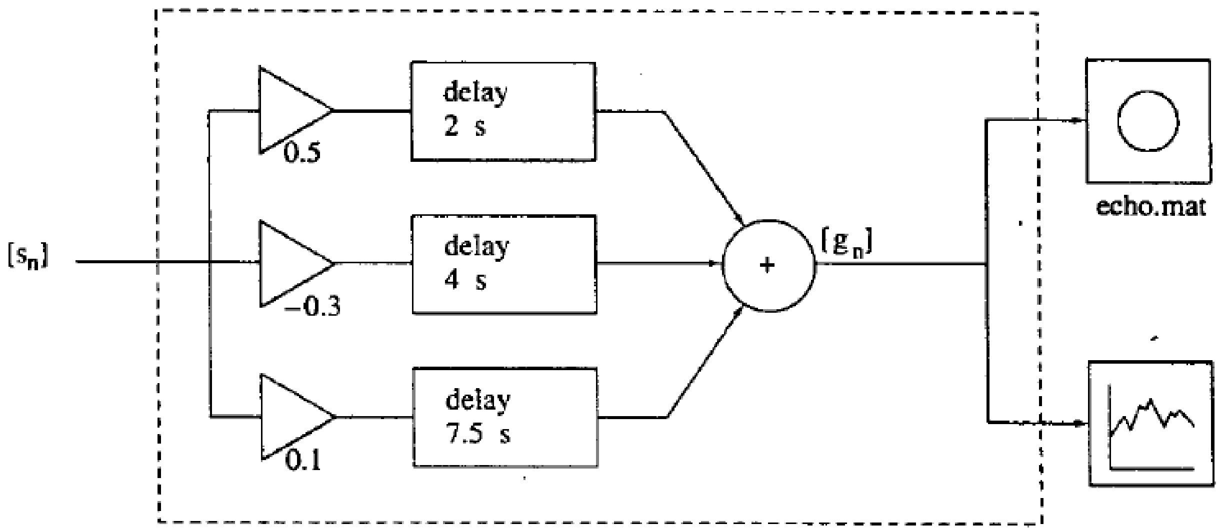
2. INPUT/OUTPUT აღწერა

ნახ. 3.4 –ის პუნქტირით შემოსაზღვრული არე მოიცავს საწყისი სიგნალისგან (s_n) ექოს გენერირების დაწვრილებით სურათს. სიგნალის მნიშვნელობები გამრავლებულია განსაზღვრულ კოეფიციენტზე, რომ მივიღოთ ექოს შესაბამისი მნიშვნელობები, შემდეგ საწყისი სიგნალი და სამივე ექო შეკრებილია ახალი სიგნალის სახით (g_n), რომელიც უნდა ავაგოთ და შევინახოთ ფაილში echo.mat.

3. სახელდახელო ამოხსნა

ავიღოთ პირველი 3 მონაცემი:

დრო, წმ	სიგნალის მნიშვნელობა
00	0.0
0.1	0.5
0.2	1.0

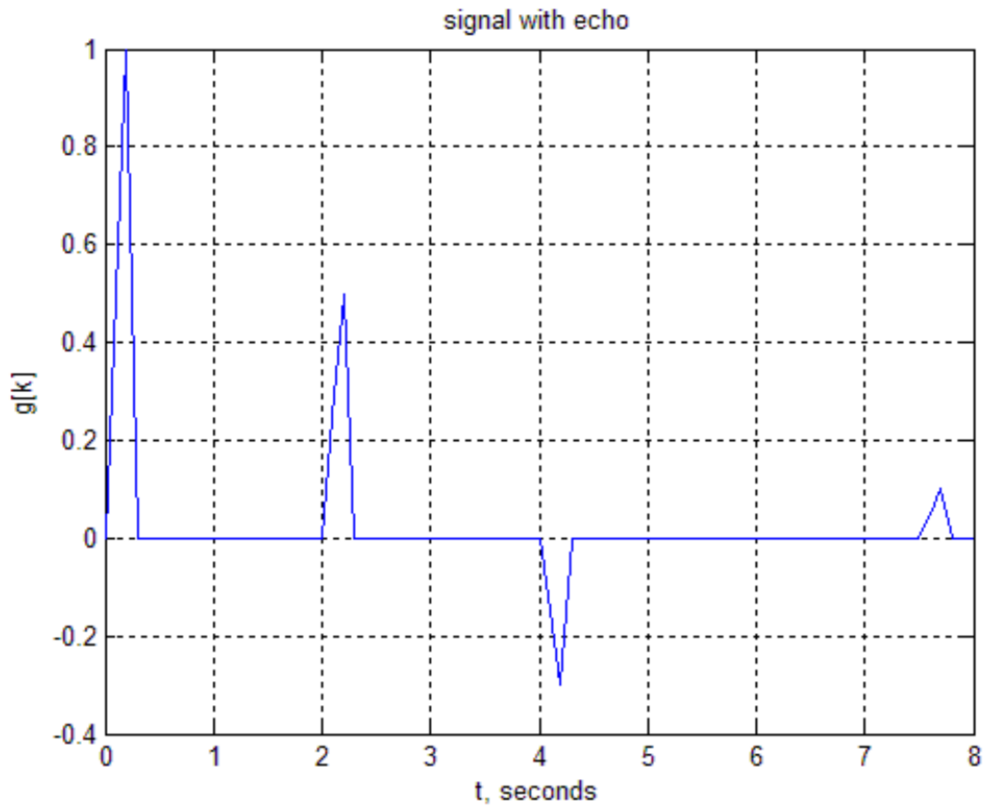


ნახ. 3.4 I/O დიაგრამა

მივიღებთ:

2.0	(0.5)*(0.0) = 0.0
2.1	(0.5)*(0.5) = 0.25
2.2	(0.5)*(1.0) = 0.5
4.1	(-0.3)*(0.5) = -0.15
4.2	(-0.3)*(1.0) = -0.3
7.5	(0.1)*(0.0) = 0.0
7.6	(0.1)*(0.5) = 0.05
7.5	(0.1)*(1.0) = 0.1

ნახაზზე 3.5 მოცემულია საწყისი სიგნალი + სამი ექოს გრაფიკი



ნახ. 3.5 საწყისი სიგნალი 3 ექოთი

4. MATLAB ამოხსნა

იმისათვის, რომ გამოვითვალოთ ინდექსები ექოსათვის, გავიხსენოთ ფაქტი, რომ ათვლის ინტერვალია 0.1. ამიტომ იმისათვის, რომ შევქმნათ ექო 2 წამის დაგვიანებით უნდა დავაგვიანოთ ექო ათვლის 20 წერტილით. s(1) მნიშვნელობამ უნდა შექმნას echo_1(21) მნიშვნელობა, s(2) – echo_1(22) და ა.შ. MATLAB პროგრამა ასე გამოიყურება:

```
%                    This program generates tree echoes from an
%                     original signal
```

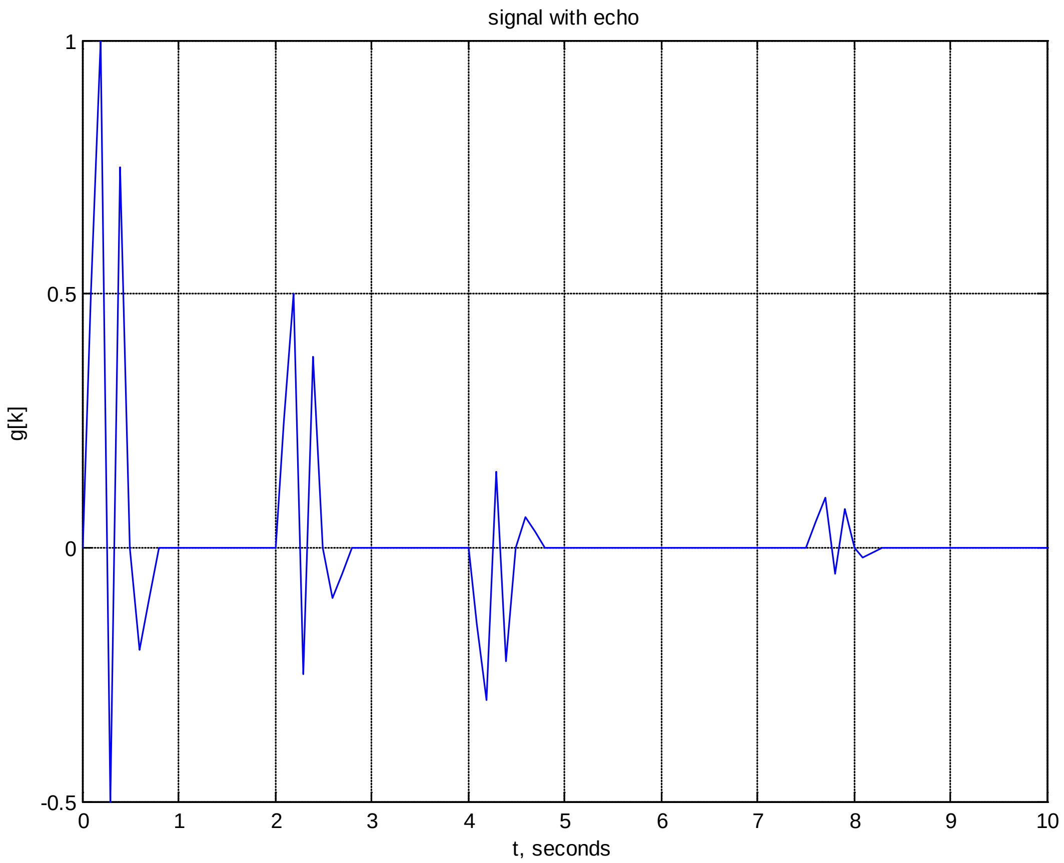
```
%                The sum of the original signal and the tree
%                 echoes is stored
%                 in a data file and then plotted.

t=0.0:0.1:10;
s=zeros(size(t));
s(1:8)=[0 0.5 1 -0.5 0.75 0 -0.2 -0.1];
%
%        Generate three echoes by scaling and
%        delaiing  original signal.
%
echo_1=zeros(size(t));
echo_1(21:101)=0.5*s(1:81);
echo_2=zeros(size(t));
echo_2(41:101)=-0.3*s(1:61);
echo_3=zeros(size(t));
echo_3(76:101)=0.1*s(1:26);
%
%         Add echoes to original signal giving new signal.

%        Save time signal and new signal.
%
g=s+echo_1+echo_2+echo_3;
save echo3 t g;
%
%          plot new signal.
%
plot(t,g),...
   title('signal with echo'),...
xlabel('t, seconds'),...
ylabel('g[k]'),...
grid
```

5. შემოწმება

გრაფიკი 3.6 შეიცავს საწყის სიგნალს და სამ ექოს.



ნახ. 3.6 საწყისი სიგნალი 3 ექოთი

3.4 ძირითადი მათემატიკური ფუნქციები

გარდა მარტივი არითმეტიკული ოპერაციებისა საჭიროა სხვა ძირითადი მათემატიკური ოპერაციები როგორიცაა ლოგარითმი, ტრიგონომეტრიული ფუნქციები, და სხვა. მათთვის MATLAB გააჩნია ფუნქციები. მაგალითად თუ გვინდა გამოვითვალოთ კუთხის სინუსი, ვიყენებთ შესაბამის ფუნქციას:

```
b = sin(angle);
```

იგულისხმება რომ **sin** ფუნქციისს არგუმენტი რადიანებაშია გამოსახული. თუ არგუმენტი გრადუსებშია მოცემული, იგი წინასწარ უნდა გადავიყვანოთ რადიანებში:

```
b = sin(angle*pi/180);
```

შეგვიძლია კუთხე წინასწარ გადავიყვანოთ რადიანებში:

```
angle_radians = angle*pi/180;
b = sin(angle_radians);
```

ეს ბრძანებები ვარგისია იმ შემთხვევაშიც, როცა კუთხე სკალარია, და მაშინაც, როცა კუთხის მნიშვნელობები ვექტორის სახითაა მოცემული. თუ კუთხის ნიშვნელობები მატრიცის სახითაა მოცემული, მაშინ sin დაითვლება თითოეული ელემენტისათვის.

განვიხილოთ წესები, რომელიც MATLAB –ის ფუნქციებს შეეხება. ფუნქციას გააჩნია არგუმენტი - პარამეტრები, რომელიც ფრჩხილებში იწერება ფუნქციის სახელის შემდეგ. ფუნქციას შესაძლოა არც გააჩნდეს არგუმენტი, ან გააჩნდეს ერთი ან რამდენიმე არგუმენტი. მაგალითად pi ფუნქციაა, რომელსაც არგუმენტი არ აქვს. როცა ჩვენ მივუთითებთ pi იგი ავტომატურად გვაძლევს π მნიშვნელობას. თუ ფუნქციას გააჩნია ერთი ან რამდენიმე არგუმენტი, ძალიან მნიშვნელოვანია არგუმენრების მიმდევრობის დაცვა. ზოგიერთი ფუნქცია მოითხოვს, რომ არგუმენტი მოცემული იყოს განსაზღვრულ ერთეულში. მაგალითად, ტრიგონომეტრიული ფუნქციების არგუმენტის განზომილება უნდა იყოს რადიანი. ზოგიერთ ფუნქციას შესაძლოა ჰქონდეს არგუმენტების ცვალებადი რაოდენობა იმის მიხედვით, თუ რა გვსურს შედეგად მივიღოთ. ასე მაგალითად, ფუნქციას **zeros** შეიძლება ჰქონდეს ერთი ან ორი არგუმენტი რაც განსაზღვრავს შედეგას.

არ შეიძლება ფუნქციას მოჰყვებოდეს ტოლობის ნიშანი, რადგან იგი ფუნქციაა და არა ცვლადი, რომელმაც რაღაც მნიშვნელობა უნდა მიიღოს, მაგრამ დასაშვებია, რომ ფუნქცია თავად მოჰყვებოდეს ტოლობის ნიშანს. შესაძლებელია, აგრეთვე, რომ რაიმე ფუნქცია სხვა ფუნქციის არგუმენტის ნაწილი იყოს. მაგალითად, შემდეგი ბრძანება გამოითვლის x ცვლადის აბსოლუტური სიდიდის ლოგარითმს:

```
log_x = log(abs(x));
```

როცა ერთი ფუნქცია გამოიყენება იმისათვის, რომ გამოვითვალოთ სხვა ფუნქციის არგუმენტი, კარგად დააკვირდით, რომ არგუმენტები შესაბამისი ფუნქციების მისთვის განკუთვნილ ფრჩხილებში იქნას ჩაწერილი.

განვიხილოთ ფუნქციათა ის კატეგორია, რომელიც ყველაზე ხშირად გამოიყენება საინჟინრო ამოცანეების პროგრამირებისას.

3.1.5 ელემენტარული მათემატიკური ფუნქციები.

ეს ფუნქციები საჭიროა იმისათვის, რომ შევასრულოთ ზოგადი გამოთვლები,როგორიცაა ვიპოვოთ აბსოლუტური სიდიდე ან გამოვთვალოთ კვადრატული ფესვი. MATLAB –ს გააჩნია რამდენიმე ფუნქცია რიცხვითი სიდიდის დამრგვალებისათვის:

abs(x)	გამითვლის x ცვლადის აბსოლუტურ სიდიდეს
sqrt(x)	ამოიღებს კვადრატულ ფესვს x სიდიდიდან
round(x)	ამრგვალებს x სიდიდეს უახლოეს მთელამდე
fix(x)	ამრგვალებს x სიდიდეს უახლოეს მთელამდე 0 მიმართულებით
floor(x)	ამრგვალებს x სიდიდეს უახლოეს მთელამდე -∞მიმართულებით
ceil(x)	ამრგვალებს x სიდიდეს უახლოეს მთელამდე +∞ მიმართულებით
sign(x)	გვაძლევს –1, თუ x<0, 0 – თუ x=0, 1 – თუ x>0
rem(x,y)	გვაძლევს x სიდიდის y –ზე გაყოფის შედეგად მიღებულ ნაშთს
exp(x)	გვაძლევს e^x, სადაც e ნატურალური ლოგარითმის ფუძეა (ნეპერის რიცხვია) ~ 2.718282
log(x)	გვაძლევს x-ის ლოგარითმს ნატურალური ფუძით
log10(x)	გვაძლევს x-ის ლოგარითმს 10-ის ფუძით

სავარჯიშო

განსაზღვრეთ შემდეგი გამოსახულებები და პასუხები შეამოწმეთ MATLAB საშუალებით

1. round(-2.6)
2. fix(-2.6)
3. floor(-2.6)
4. ceil(-2.6)
5. sign(-2.6)
6. abs(round(-2.6))
7. sqrt(floor(10.7))
8. rem(15,2)
9. floor(ceil(10.8))
10. log10(100) + log10(0.001)
11. abs(-5:5)
12. round([0:0.3:2, 1:0.75:4])

ტრიგონომეტრიული ფუნქციებით სარგებლობისას უნდა გვახსოვდეს, MATLAB არგუმენტს აღიქვამს რადიანებში. იმისათვის, რომ რადიანი გადავიყვანოთ გრადუსებში ან პირიქით, გამოვიყენოთ შემდეგი გამოსახულებები, ვიცით რა $180^o = \pi$ რადიანს:

```
angle_degrees = angle_radians*(180/pi);
angle_radians = angle_degrees*(pi/180);
```

sin(x)	გამოითვლის x ცვლადის სინუსს, x გამოსახულია რადიანებში
cos(x)	გამოითვლის x ცვლადის კოსინუსს, x გამოსახულია რადიანებში
tan(x)	გამოითვლის x ცვლადის ტანგენსს, x გამოსახულია რადიანებში
asin(x)	გამოითვლის x ცვლადის არკსინუსს, ანუ შებრუნებულ სინუსს,სადაც x მდებარეობს –1 და 1 შორის. ფუნქცია გვაძლევს კუთხეს -π/2 და π/2 შორის.
acos(x)	გამოითვლის x ცვლადის არკკოსინუსს, ანუ შებრუნებულ კოსინუსს,სადაც x მდებარეობს –1 და 1 შორის. ფუნქცია გვაძლევს კუთხეს -π/2 და π/2 შორის.
atan2(x,y)	გამოითვლის არკტანგენსს, ანუ შებრუნებულ ტანგენსს სიდიდისა y/x. ფუნქცია იძლევა კუთხეს რადიანებში, რომელიც მდებარეობს საზღვრებში -π + π, x და y ნიშნების მიხედვით.

სხვა ტრიგონომეტრიული ფუნქციები შეიძლება გამოვითვალოთ:

$$\sec x = \frac{1}{\cos x}$$

$$\csc x = \frac{1}{\sin x}$$

$$\cot x = \frac{1}{\tan x}$$

სავარჯიშო

მიეცით MATLABS ბრძანებები იმისათვის, რომ გამოითვალოთ შემდეგი სიდიდეები, იმ დაშვებით, რომ განტოლებებში შემავალი ცვლადები სკალარული სიდიდეებია და წინასწარ ცნობილია მათი რიცხვითი მნიშვნელობები.

1. თანაბრად აჩქარებული მოძრაობა

$$motion = \sqrt{\upsilon\iota^2 + 2 \cdot a \cdot x}$$

2. ელექტრული რხევის სიხშირე:

$$frequency = \frac{1}{\sqrt{\frac{2\pi \cdot c}{L}}}$$

3. ტყვიის მოძრაობის ტრაექტორია:

$$range = 2\upsilon i^2 \cdot \frac{\sin(b) \cdot \cos(b)}{g}$$

4. length contraction

$$length = k\sqrt{1 - \left(\frac{\upsilon}{c}\right)^2}$$

5. fillet რგოლის მოცულობა

$$volume = 2\pi x^2\left(\left(1 - \frac{\pi}{4}\right) \cdot y - \left(0.833 - \frac{\pi}{4}\right) \cdot x\right)$$

6. მანძილი მიზიდულობის ცენტრიდან წაკვეთილი ცილინდრის კვეთის სიბრტყიდან (distance of the center of gravity from a reference plane in a hollow cylinder sector):

$$center = \frac{38.1972 \cdot \left(r^3 - s^3\right)\sin\alpha}{\left(r^2 - s^2\right) \cdot \alpha}$$

ჰიპერბოლური ფუნქცია წარმოადგენს ნატურალური ხარისხის (e^x) ფუნქციას. შებრუნებული ჰიპერბოლური ფუნქცია კი x სიდიდის ნატურალური ფუძით ლოგარითმია (ln x). ეს ფუნქციები გამოიყენება ისეთ ამოცანებში, როგორიცაა მაგალითად ზოგიერთი ტიპის ციფრული ფილტრის დიზაინი. MARLAB - ში არსებობს სპეციალური ფუნქციები მათ გამოსათვლელად.

sinh(x) გამითვლის x ცვლადის ჰიპერბოლურ სინუსს, რომელიც ტოლია $\frac{e^x - e^{-x}}{2}$

cosh(x) გამითვლის x ცვლადის ჰიპრბოლურ კოსინუსს რომელიც ტოლია $\frac{e^x + e^{-x}}{2}$

tanh(x) გამითვლის x ცვლადის ჰიპერბოლურ ტანგენსს, x რომელიც ტოლია $\frac{\sinh x}{\cosh x}$

asinh(x) გამოითვლის x ცვლადის შებრუნებულ ჰიპერბოლურ სინუსს -

$\ln\left(x+\sqrt{x^2+1}\right)$

acosh(x) გამოითვლის x ცვლადის შებრუნებულ ჰიპერბოლურ კოსინუსს - $\ln\left(x+\sqrt{x^2-1}\right)$, x >= 1-თვის

atanh(x) გამოითვლის x სიდიდის შებრუნებულ ჰიპერბოლურ ტანგენსს- $\frac{1}{2}\ln\left(\frac{1+x}{1-x}\right)$ |x| < 1-თვის.

სხვა ჰიპერბოლური ფუნქციები გამოითვლება შემდეგი ფორმულების საშუალებით:

$$\coth x = \frac{\cosh x}{\sinh x} \qquad x \neq 0$$

$$\sec hx = \frac{1}{\cosh x}$$

$$\csc hx = \frac{1}{\sinh x}$$

$$a\coth x = \frac{1}{2}\ln\left(\frac{x+1}{x-1}\right) \quad |x| > 1$$

$$a\sec hx = \ln\left(\frac{1+\sqrt{1-x^2}}{x}\right) \quad 0 < x < 1$$

$$aschx = \ln\left(\frac{1}{x}+\frac{\sqrt{1+x^2}}{|x|}\right) \quad x \neq 0$$

სავარჯიშო

გამოითვალეთ MATLAB საშუალებით შემდეგი გამოსახულებები. x მნიშვნელობა წინასწარ ცნობილია ან შედეგია უკვე ჩატარებული გამოთვლებისა.

1. coth x
2. sec x
3. csc x
4. acoth x
5. asech x
6. acsc x

3.1.6 M ფაილები – MATLAB –ის ახალი ფუნქციები

ზოგიერთი სახის გამოთვლები, რომლებიც უფრო ხშირად გჭირდებათ, შეგიძლიათ აქციოთ MATLAB ფუნქციებად. ამისათვის უნდა დაიწეროს პროგრამა, რომელიც საჭირო გამოთვლებს ჩაატარებს და ჩაიწეროს იგი M ფაილის სახით. შეგიძლიათ ეს ფუნქცია გამოიყენოთ, როგორც MATLAB სხვა ფუნქციები. ასეთი ფაილები გარკვეული წესების დაცვით უნდა დაიწეროს. ვიდრე განვიხილავთ ამ წესებს, მოვიყვანთ M-file ფუნქციის მაგალითს, რომელიც ჩაწერილია circum.m ფაილის სახით:

```
function c = circum ( r )
%   CIRCUM          Circumference of circle with radius r.
%                   For matrices, CIRCUM ( r ) returns a matrix
%                   containing the circumferences of circles
%                   with radii equal to the values in the original vector.
c = pi*2*r;
```

ეს ფუნქცია შეგვიძლია გამოვიყენოთ r რადიუსიანი წრეწირის სიგრძის გმოსათვლელად

```
r = [ 0 1.4 pi ];
circum( r );
```

MATLAB ახალი ფუნქციის შექმნისას უნდა დავიცვათ შემდეგი წესები:

1. პირველი სტრიქონი უნდა დაიწყოს სიტყვით function, რომელსაც მოჰყვება შედეგად მისაღები არგუმენტები, ტოლობის ნიშანი და ფუნქციის სახელი. ფუნქციის საწყისი არგუმენტები იწერება ფრჩხილებში ფუნქციის სახელის შემდეგ. ეს სტრიქონი განსაზღვრავს შესავალ და გამოსავალ არგუმენტებს (input – output) და იგი განასხვავებს ასეთ ფაილს სხვა m ფაილებისაგან.
2. შემდეგი რამდენიმე ხაზი უნდა დაეთმოს კომენტარს, რომელიც ეკრანზე გამოვა, თუ MATLAB მივცემთ ბრძანებას **help** circum.
3. შედეგად მიღებული მნიშვნელობები მიენიჭება გამოსავალ არგუმენტს, ამიტომ შეამოწმეთ: ფუნქციის ტექსტი უნდა შეიცავდეს ბრძანებას, რომელიც გამოსავალ არგუმენს მიანიჭებს გამოთვლილ მნიშვნელობას.
4. ფუნქციასა და პროგრამაში, რომელიც მას იყენებს მატრიცებისთვის ერთნაირი სახელები შეგვიძლია გამოვიყენოთ, რადგან პროგრამა და ფუნქცია სრულიად გამიჯნულნი არიან ურთიერთისგან. თუმცა პროგრამისათვის ხელმისაწვდომია მხოლოდ ფუნქციის გამოსავალი არგუმენტის მნიშვნელობა.
5. თუ ფუნქცია შედეგად იძლევა ერთზე მეტი სიდიდის მნიშვნელობას, ეს სიდიდეები უნდა იყოს მითითებული როგორც ვექტორი ბრძანებაში function ქვემოთმოყვანილ მაგალითად:
   function [dist, vel, accel] = motion (x)
   ფუნქცია უნდა ითვლიდეს სამივე სიდიდეს.

6. თუ ფუნქციას აქვს რამდენიმე შესავალი არგუმენტი, ისინი ჩამოთვლილი უნდა იყოს function ბრძანებაში: **function** error = mse(w,d)
7. სპეციალური ცვლადები nargin და nargout უნდა გამოვიყენოთ, რომ განვსაზღვროთ მიცემული ფუნქციის შესავალი და გამოსავალი არგუმენტების რაოდენობა. ეს ცვლადები მისაწვდომია მხოლოდ ფუნქციის შიგნით.

გავიხსენოთ, რომ ბრძანება **what** ჩამოთვლის ყველა M-ფაილს მოცემულ სამუშაო სივრცეში. ბრძანება **type,** რომელსაც მოყვება ფაილის სახელი გამოგვიყვანს ამ ფაილის შინაარსს ეკრანზე. თუ გაფართოება .m არ მივუთითეთ, **type** ბრძანება ავტომატურად გულისხმობს, რომ ეს .m ფაილია.

პრობლემა: ჰიდროაკუსტიკური სიგნალი

ჰიდროაკუსტიკური კვლევა საჭიროებს ბგერითი სიგნალის შექმნას, მის გავრცელებას წყალში და არეკლილი სიგნალის მიღებას. ოკეანოლოგიური ამოცანები მოიცავს ოკეანის ტოპოგრაფიის, რუქის აგების და ბიოლოგიური სიგნალების გაზომვის პრობლემებს. ინდუსტრიული ჰიდროაკუსტიკა იკვლევს თევზების ადგილსამყოფელს, ნავთობისა და მინერალების გავრცელების არეალს. საზღვაო ჰიდროაკუსტიკა გამოიყენება წყალქვეშა გემების ადგილმდებარეობის განსაზღვრისათვის. აქტიური ჰიდროაკუსტიკური სისტემა აწარმოებს სიგნალის გადაცემას, რომელიც ჩვეულებრივ გარკვეული სიხშირის სინუსოიდაა. ამ სიგნალის ანარეკლი ან ექო უკან ბრუნდება და ხდება მისი მიღება და ანალიზი. პასიური ჰიდროაკუსტიკური სისტემა სიგნალებს არ გადასცემს, მაგრამ აგროვებს და აანალიზებს მათ.

პირველ რიგში განვიხილავთ სინუსოიდას, რადგან იგი ძირითადი სიგნალია, რომელიც ჰიდროაკუსტკურ სისტემაში გამოიყენება. შემდეგ შევადგენთ MATLAB პროგრამას ჰიდროაკუსტიკური სიგნალის მოდელირებისათვის.

სინუსოიდა

ჩვენ ვიცით **sin** – კუთხის ფუნქცია. სინუსოიდა არის **sin** ფუნქცია , რომელიც ჩაწერილია როგორც დროის ფუნქცია:

$$g(t) = \sin(2\pi ft)$$

სადაც f სინუსოიდის სიხშირეა. მისი ერთეულია ციკლი/წამში, ან ჰერცი. (შევნიშნავთ, რომ 2πft-ის ერთეულია რადიანი.

თუ სინუსოიდის სიხშირე 5 ჰერცია, გვაქვს

$$g(t) = \sin(2\pi 5t) = \sin(10\pi t)$$

თუ ავაგებთ ამ ფუნქციის გრაფიკს, ვნახავთ, წამში მართლაც ხუთი რხევაა. ნახ 3.7. სინუსოიდის პერიოდი ეწოდება დროის შუალედს, რომელშიც იგი ერთ სრულ რხევას ასრულებს. ასე, რომ ამ სინუსოიდის პერიოდია 0.2 წმ. სიხშირესა და პერიოდს შორის ასეთი დამოკიდებულება არსებობს:

$$f = \frac{1}{p}; \qquad p = \frac{1}{f};$$

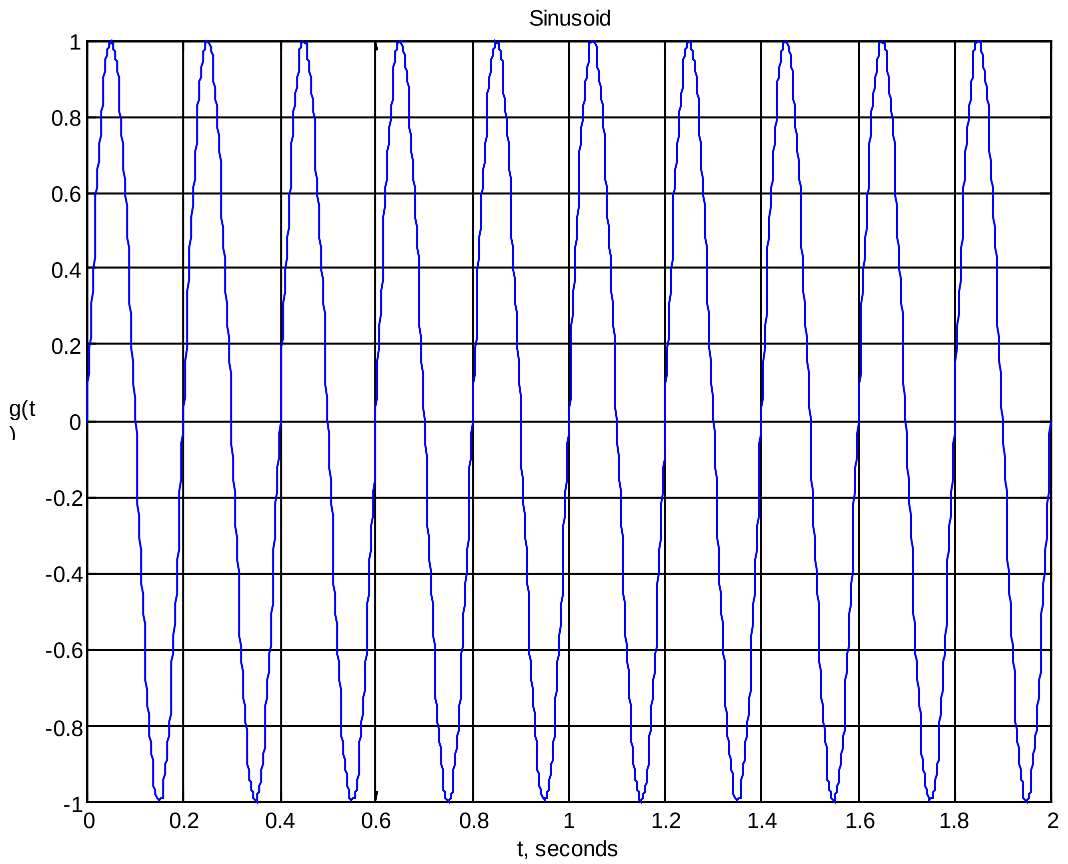
სადაც f სიხშირეა ჰერცებში, P პერიოდი წამებში.

თუ სინუსოიდას გავამრავლებთ სკალარზე A, განტოლება ჩაიწერება ფორმით:

$$g(t) = A\sin(2\pi ft)$$

ამ სკალარს უწოდებენ სინუსოიდის ამპლიტუდას. სინუსოიდა ფაზური წანაცვლებით ϕ ასე ჩაიწერება:

$$g(t) = A\sin(2\pi ft + \phi)$$



ნახ. 3.7 5 ჰერციანი სინუსოიდა

თუ ფაზური წანაცვლება π/2 რადიანის ტოლია, გვექნება:

$$A \cdot \sin\left(2\pi f t + \frac{\pi}{2}\right) = A \cdot \cos(2\pi f t)$$

სინუსოიდა არის დროის ფუნქცია, რომელიც შეიძლება ჩაიწეროს სინუსის ან კოსინუსის საშუალებით და შეიძლება ჰქონდეს ან არ ჰქონდეს ფაზური წანაცვლება.

ჰიდროაკუსტიკური სიგნალის გენერირება

აქტიური ჰიდროაკუსტიკური სისტემა გადასცემს განსაზღვრული სიხშირის სიგნალს. ამ სიგნალის ანარეკლი ან ექო უკანვე მიიღება და ხდება მისი ანალიზი გარკვეული ამოცანების შესასწავლად. პასიური სისტემა მხოლოდ არეკლილი სიგნალის მიღებას და ანალიზს აწარმოებს. ჰიდროაკუსტიკურ სისტემაში ძირითადად სინუსოიდა გამოიყენება. სინუსოიდა ასეთი სახითაა მოცემული

$$s(t) = \begin{cases} \sqrt{\dfrac{2E}{PD}} \cos(2\pi f_c t) & 0 \le t \le PD \\ 0 & elsewhere \end{cases}$$

სადაც E გადაცემის ენერგიაა, PD – პულსის ხანგრძლივობა წამებში, f_c – სიხშირე ჰერცებში.
პულსის ხანგრძლივობა შესაძლოა იყოს მილიწამის ნაწილიდან რამდენიმე წამამდე. სიხშირული დიაპაზონი რამდენიმე ასეული ჰერციდან ათეულობით კილოჰერცამდეა.

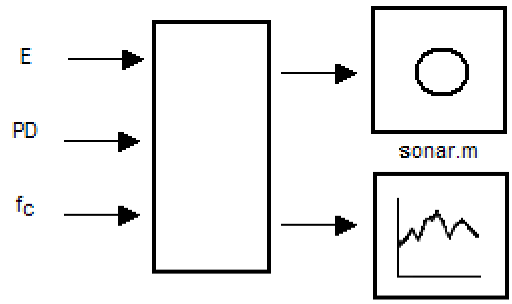
დაწერეთ MATLAB პროგრამა, რომელიც საშუალებას მოგცემთ მიაწოდოთ პროგრამას E, PD, f_c მნიშვნელობები ჰიდროაკუსტიკური სიგნალის გენერირებისათვის, შეინახავს სიგნალის მნიშვნელობებს ფაილში sonar.mat. სიგნალის ათვლის ხანგრძლივობა უნდა ფარავდეს პულსის ხანგრძლივობას, ყოველი პერიოდის განმავლობაში 10 ანათვალი უნდა იქნას აღებული. გარდა ამისა, სიგნალს უნდა დაემატოს 200 წერტილი სიჩუმისა პულსის შეწყვეტის შემდეგ.

1. ამოცანის დასმა

დაწერეთ პროგრამა იმისათვის, რომ შეიქმნას გარკვეული ხანგრძლივობის ჰიდროაკუსტიკური სიგნალი, რომელიც შეიცავს 10 ანათვალს ყოველი პერიოდის განმავლობაში.

2. INPUT/OUTPUT აღწერა

სიდიდეები: E გადაცემის ენერგია ჯოულებში, PD – პულსის ხანგრძლივობა წამებში, f_c სიხშირე ჰერცებში არის INPUT, შესავალი მნიშვნელობები. გამოსავალი მნიშვნელობაა მონაცემთა ფაილი სახელით sonar.mat, რომელიც შეიცავს დროის და სიგნალის შესაბამის მნიშვნელობებს. ავაგებთ აგრეთვე სიგნალის გრაფიკს:



ნახ. 3.8 I/ O დიაგრამა

3. სახელდახელო ამოხსნა

ვისარგებლოთ შემდეგი მონაცემებით:

E = 500 ჯოული
PD = 5 მილიწამი
f_c = 3.5 კილოჰერცი

სინუსოიდის პერიოდია 1/3500 = 0.3 მილიწამი. იმისათვის, რომ გვქონდეს 10 ანათვალი პერიოდში, ათვლის ინტერვალი უნდა იყოს 0.03 მილიწამი. პულსის ხანგრძლივობაა 5 მილიწამი და ამიტომ გვჭირდება 167 (5/0.03) ანათვალი შემდეგი სიგნალისა:

$$s(t) = \sqrt{\frac{2E}{PD}}\cos(2\pi f_c t) = \sqrt{\frac{1000}{0.005}}\cos(2\pi(3500)t)$$

ქვემოთ მოყვანილია სიგნალის პირველი რამდენიმე მნიშვნელობა:

t(მწმ)	s(t)
0.00	447.2
0.03	353.4
0.06	11.2
0.09	-177.6
0.12	-391.9
0.15	-441.7
0.18	-306.1
0.21	-42.1
0.24	239.6
0.27	420.8
0.30	425.3
0.33	251.4

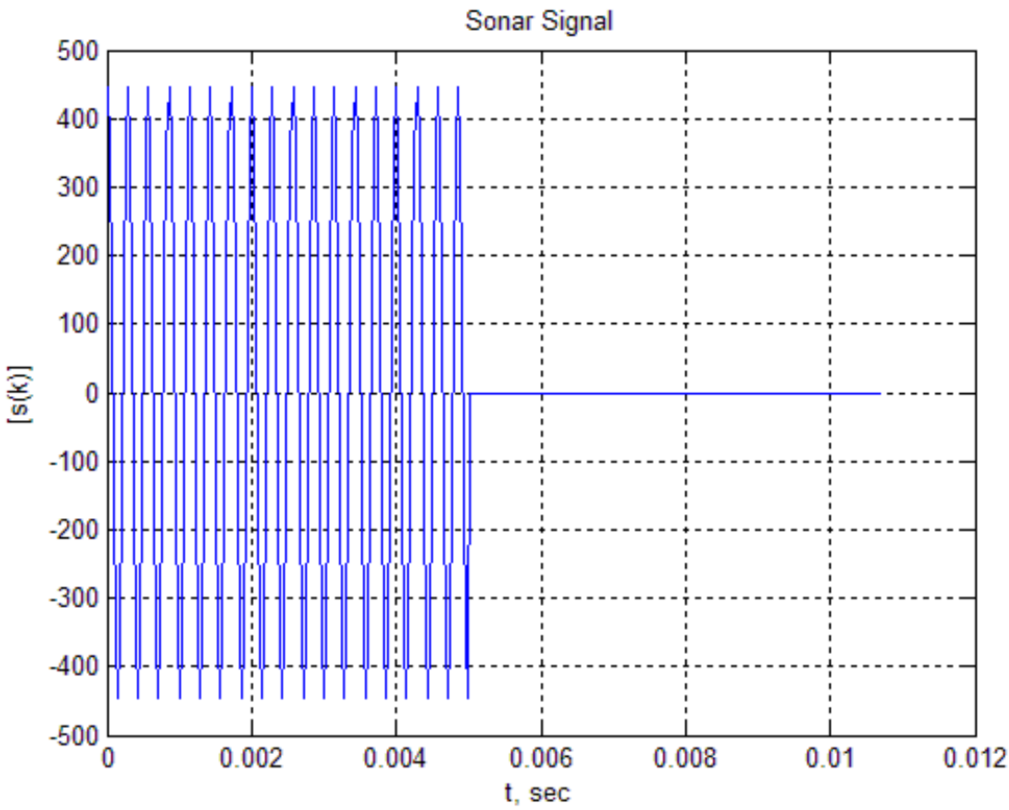
ამას დავუმატებთ 200 წერტილს, როცა სიგნალის მნიშვნელობა 0 –ის ტოლია.

4. MATLAB ამოხსნა

ამ პროგრამის ყველაზე მნიშვნელოვანი ნაწილია სიჩუმისათვის დროის შესაბამისი მნიშვნელობების გამოთვლა. დროის მნიშვნელობებს უნდა დავუმატოთ 200 წერტილი. ჯერ შევქმნით სკალარს სახელად silence_start, რომელიც მიიღებს დროის იმ მნიშვნელობას, როცა სიგნალი შეწყდა და დაიწყო სიჩუმე. შემდეგ შევქმნით სიჩუმის პერიოდის საბოლოო მნიშვნელობას და მათ შორის ორწერტილოვანი ოპერატორით შევქმნით ვექტორს, რომელიც შეიცავს ამ ინტერვალში დროის მნიშვნელობებს. ამის შემდეგ შევქმნით ახალ ვექტორს სიგნალის მნიშვნელობებისათვის, რომელიც შეიცავს ზუსტად იმდენ ნულოვანი მნიშვნელობის ელემენტს, რამდენსაც დროის ახალი ვექტორი.

```
%               This program generates a sonar signal
%               using values obtained from the user for
%               energy, pulse duration and frequency.
%               An additional 200 points of silence are added
%               to the signal
%               before it is stored.
  energy = input('Enter energy in joules ');
  duration =input('Enter puls duration in seconds ');
  fc =input('Enter frequency in Hz ');
%
%Generate sonar signal
%
  A=sqrt(2*energy/duration);
  period=1/fc;
  t_incr=period/10;
  t=[0:t_incr:duration];
  s=A*cos(2*pi*fc*t);
%
```

```
%generate and add 200 points of silence
%
last=length(t);
silence_start=t(last)+t_incr;
silence_end=t(last)+200*t_incr;
silence=[silence_start:t_incr:silence_end];
t=[t silence];
s=[s zeros(size(silence))];
%
%Save and plot sonar signal
%
save sonar t s
plot(t,s),...
    title('Sonar Signal'),...
    xlabel('t, sec'),...
    ylabel('[s(k)]'),...
    grid
```



ნახ. 3.9 ჰიდროაკუსტიკური სიგნალი

3.5 კომპლექსური რიცხვები

მრავალი საინჟინრო ამოცანა მოითხოვს ვიპოვოთ შემდეგი განტოლების ფესვები:

y = f(x)

სადაც ფესვები არის x –ის ის მნიშვნელობები, რომელთათვისაც $y = 0$.

კვადრატული განტოლებისათვის (მეორე რიგის პოლინომი) ვიყენებთ სპეციალურ ფორმულას. განვიხილოთ კვადრატული განტოლება:

$$f(x) = x^2 + 3x + 2$$

თუ დავშლით მამრავლებად, მივიღებთ:

$$f(x) = (x+1) \cdot (x+2)$$

როგორც ვნახეთ ამ განტოლების ფესვებია -1 და -2.
ახლა განვიხილოთ კვადრატული განტოლება:

$$f(x) = x^2 + 3x + 3$$

მისი ფესვებია:

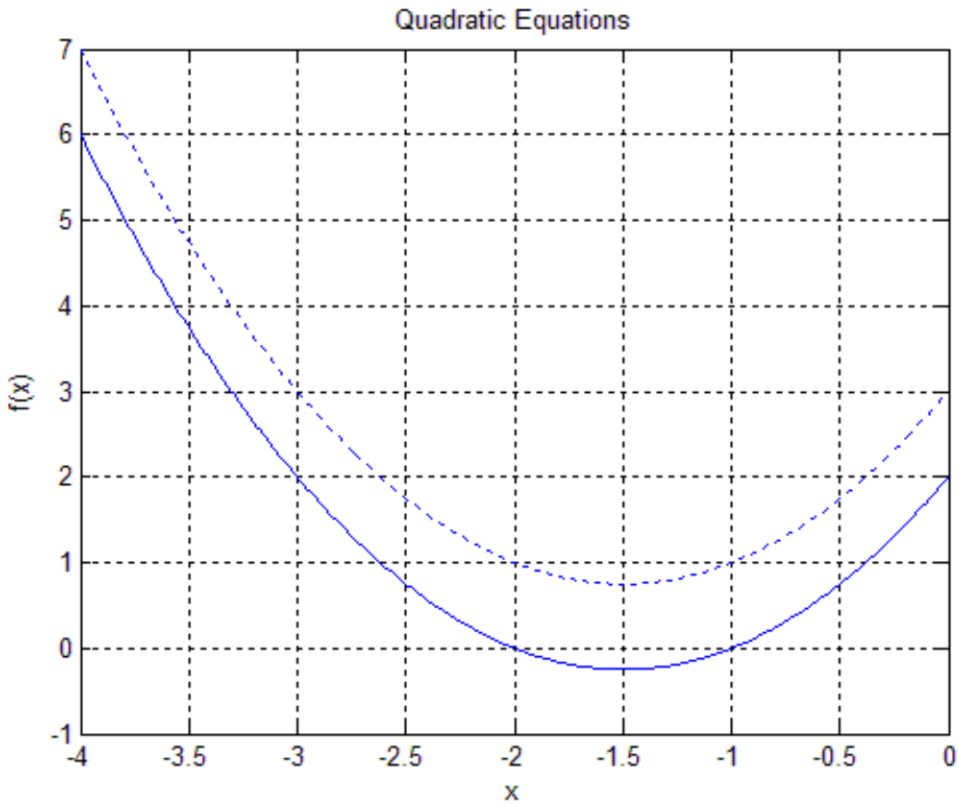
$$x_1 = -1.5 + 0.87\sqrt{-1}$$
$$x_2 = -1.5 - 0.87\sqrt{-1}$$

იმისათვის, რომ ამ გამოსახულებას აზრი ჰქონდეს, უნდა განვსაზღვროთ უარყოფითი რიცხვიდან კვადრატული ფესვის მნიშვნელობა. ამისათვის არსებობს კომპლექსურ რიცხვთა სიმრავლე, $a + ib$, სადაც $I = \sqrt{-1}$. თუ (b =0 შემთხვევაში იგი ნამდვილი რიცხვია).
ნახ 3.10 წარმოადგენს ორივე კვადრატული პოლინომის გრაფიკს. როგორც მოსალოდნელი იყო, პირველი მათგანის გრაფიკი კვეთს x ღერძს ორ წერტილში, რომლებიც მის რეალურ ფესვებს შეესაბამება, ხოლო მეორე პოლინომის გრაფიკი არ გადაკვეთს x ღერძს, რადგან მას გააჩნია ორი კომპლექსური ფესვი.

განვიხილოთ n რიგის პოლინომის ზოგადი სახე:

$$a_1 x^n + a_2 x^{n-1} + a_3 x^{n-2} + \cdots + a_{n-2} x^2 + a_{n-1} x + a_n = 0$$

n რიგის პოლინომს აქვს n ფესვი, ზოგიერთი მათგანი ნამდვილია, ზოგიც კომპლექსური. ამ წიგნის მე-10 თავში განვიხილავთ MATLAB ფუნქციას პოლინომის ფესვების საპოვნელად. ამ თავში კი განვიხილავთ მოქმედებებს კომპლექსურ რიცხვებზე და MATLAB ფუნქციებს, რომლებიც კომპლექსურ რიცხვებს უკავშირდება.



ნახ. 3.10 ორი სხვადასხვა კვადრატული ფუნქციის გრაფიკი

3.1.7 არითმეტიკული ოპერაციები კომპლექსურ რიცხვებზე

მრავალი საინჟინრო პრობლემის ამოხსნისას კომპლექსური რიცხვები მნიშვნელოვან როლს ასრულებს:

მოქმედება	შედეგი
$c_1 + c_2$	$(a_1+a_2)+i(b_1+b_2)$
$c_1 - c_2$	$(a_1-a_2)+i(b_1-b_2)$
$c_1 * c_2$	$(a_1 a_2 - b_1 b_2)+i(a_1 b_2 + a_2 b_1)$
c_1 / c_2	$((a_1 a_2 + b_1 b_2)/(a_2^2 + b_2^2)) +$ $i((a_2 b_1 + b_2 a_1)/(a_2^2 + b_2^2))$
$\|c_1\|$	$\sqrt{a_1^2 + b_1^2}$ (magnitude)
c_1^*	$a_1 - ib_1$ (conjugate of c_1)

სადაც $c_1 = a_1 + ib_1$ $c_2 = a_2 + ib_2$

ბრძანება x = 1 – 0.5*i გვაძლევს კომპლექსურ რიცხვს x. როცა ვაწარმოებთ მოქმედებებს კომპლექსურ რიცხვებზე, MATLAB ავტომატურად ასრულებს ზემოთმოყვანილ ცხრილში აღწერილ მოქმედებებს. თუ მოქმედება სრულდება ნამდვილსა და კომპლესურ რიცხვს შორის, MATLAB გულისხმობს, რომ ნამდვილი რიცხვი ეს არის კომპლექსური რიცსვი, რომლის წარმოსახვითი ნაწილი 0 –ის ტოლია.

MATLAB – ს აქვს რამდენიმე ფუნქცია კომპლექსური რიცხვებისათვის:

real(x) ითვლის კომპლექსური რიცხვის ნამდვილ ნაწილს

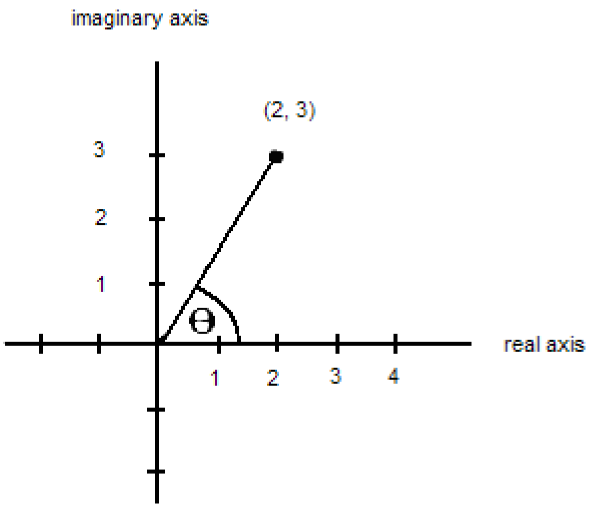
imag(x)	ითვლის კომპლექსური რიცხვის არმოსახვით ნაწილს
conj(x)	ითვლის კომპლექსური x რიცხვის შეუღლებულს
abs(x)	ითვლის კომპლექსური რიცხვის მოდულს.
angle(x)	ითვლის კუთხეს atan2(imag(x), real(x)) ფუნქციის საშუალებით

ეს ფუნქციები აადვილებს კომპლექსური რიცხვის ერთი ფორმიდან მეორეში გადაყვანას.

3.1.8 კოორდინატთა დეკარტის და პოლარული სისტემა

კომპლექსურ რიცხვთა სისტემა შეგვიძლია წარმოვადგინოთ როგორც სიბრტყე ნამდვილი და წარმოსახვითი ღერძებით. რეალური რიცხვები წარმოადგენს x ღესძს, წარმოსახვითი რიცხვები (ნამდვილი ნაწილის გარეშე) – y ღერძს, ხოლო ყველა სხვა კომპლექსური რიცხვი, რომელსაც გააჩნია როგორც ნამდვილი, ისე წარმოსახვითი ნაწილი, დაიკავებს შესაბამის ადგილს სიბრტყეზე. როცა ვიხილავთ კომპლექსურ რიცხვს ფორმით $2 - i3$, საქმე გვაქვს კომპლექსური რიცხვის მართკუთხა აღნიშვნასთან. კომპლექსური რიცხვები შეგვიძლია წარმოვადგინოთ θ კუთხისა და r რადიუსის საშუალებით – პოლარული აღნიშვნა. ნახ. 3.11 –დან მარტივად შეგვიძლია განვსაზღვროთ დამოკიდებულება მართკუთხა და პოლარულ კოორდინატებს შორის:

$$r = \sqrt{a^2 + b^2}; \theta = \tan^{-1}\frac{b}{a}$$



ნახ. 3.11 კონპლექსური სიბრტყე

პოლარულიდან დეკარტის სისტემაში გადასაყვანი ფორმულები:

$$a = r\cos\theta$$
$$b = r\sin\theta$$

კალკულატორს შეუძლია შეარულოს გადასაყვანი ფორმულების გამოთვლა. აქ საჭიროა სიფრთხილე, რომ ზუსტად განვსაზღვროთ რომელ მეოთხედში უნდა მდებარეობდეს θ კუთხე. ხშირ შემთხვევაში კალკულატორი ავტომატურად იძლება კუთხის მნიშვნელობას პირველ მეოთხედში, მისი ყოველგვარი ანალიზის გარეშე. MATLAB-ში ფუნქცია **atan(x)** კუთხეს გვაძლევს შუალედში -π/2 - +π/2, მაშინ როცა **atan2(y,x)** გვაძლევს y/x ტანგენსის შებრუნებულს -π - +π შუალედში. ფუნქცია **angle(x)** არგუმენტია კომპლექსური რიცხვი x და გვაძლევს სიდიდეს atan2(imag(x), real(x)).

თუ x კომპლექსური რიცხვია მისი მოდული და ფაზა გამოითვლება შემდეგი ბრძანებით:

```
r = abs(x);
theta = angle(x);
```

თუ გვსურს გამოვითვალოთ კომპლექსური რიცხვი, როცა მოცემულია მისი მოდული და ფაზა:

```
y = r*exp(I*theta);
```

კომპლექსური რიცხვის ნამდვილი და წარმოსახვითი ნაწილი შეგვიძლია გამოვითვალოთ შემდეგი ბრძანებებით:

```
a = real(x);
b = imag(x);
```

რომ წარმოვადგინოთ კომპლექსური რიცხვი მისი ნამდვილი და წარმოსახვითი ნაწილის საშუალებით, დავწერთ: y = a + i*b.

სავარჯიშო

წარმოადგინეთ კომპლექსური რიცხვები პოლარული ფორმით, შეამოწმეთ პასუხები MATLAB საშუალებით:

1. 3 – i2
2. –I
3. –2
4. 0.5 + i

გარდაქმენით კომპლექსური რიცხვები ექსპონენციალურიდან მართკუთხა ფორმაში. შეამოწმეთ პასუხები MATLAB საშუალებით. hh

5. ei
6. eiπ0.75
7. 0.5ei2.3
8. 3.5 ei3π

3.1.9 ეილერის ფორმულა

კომპლექსური რიცხვების ზოგიერთი მნიშვნელოვანი თვისების მისაღებად განვიხილოთ Maclaurin მწკრივი:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \cdots$$

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} - \cdots$$

ვთქვათ x არის წარმოსახვითი სიდიდე ib, მაშინ გვექნება:

$$e^{ib} = 1 + ib + \frac{(ib)^2}{2!} + \frac{(ib)^3}{3!} - \cdots = 1 + ib - \frac{b^2}{2!} - i\frac{b^3}{3!} + \frac{b^4}{4!} - \frac{b^5}{5!} + \cdots$$

საბოლოოდ მივიღებთ:

$$e^{ib} = \cos b + i \sin b$$

ეს ძალიან მნიშვნელოვანი ფორმულა და ეილერის ფორმულას უწოდებენ. მისგან მიიღება კიდევ 2 გამოსახულება:

$$\sin\theta = \frac{e^{i\theta} - e^{-i\theta}}{2i}$$

$$\sin\theta = \frac{e^{i\theta} + e^{-i\theta}}{2}$$

ეილერის ფორმულის საშუალებით კომპლექსური რიცხვი შეგვიძლია გამოვსახოთ პოლარულ და მართკუთხა ფორმით.

$$a + ib = (r\cos\theta) + i(r\sin\theta) = r(\cos\theta \_ i\sin\theta)$$

ეილერის ფორმულის გამოყენებით მივიღებთ:

$$a + ib = re^{i\theta}$$

sadac $r = \sqrt{a^2 + b^2}; \theta = \tan^{-1}\frac{b}{a}; a = r\cos\theta; b = r\sin\theta$

ამ თავში განვიხილეთ მათემატიკური ოპერაციები, MATLAB ფუნქციები მატრიცების შესაქმნელად და უკვე არსებული მატრიცებისაგან ახალ მატრიცების საწარმოებლად. განვიხილეთ მათემატიკური ოპერაციები მატრიცების შესაბამის ელემენტებს შორის ე.წ. მასივური ოპერაციები. ვისწავლეთ აგრეთვე როგორ შევქმნათ ახალი ფუნქცია MATLAB-ში m-ფაილის სახით. განხილული იქნა აგრეთვე კომპლექსური რიცხვები და მათზე წარმოებული მათემატიკური ოპერაციები. განვიხილეთ რამდენიმე ამოცანა, რომელიც

დაკავშირებული იყო საკომუნიკაციო სიგნალის ექოსთან და ჰიდროაკუსტიკური სიგნალის მოდელირებასთან.

ყველა სპეცსიმბოლო, ბრძანება და ფუნქცია, რომელიც ამ თავში იქნა განხილული მოკლე მიმოხილვითირთ.

სპეცსიმბოლოები

+	სკალარული და მასივური შეკრება
-	სკალარული და მასივური გამოკლება
*	სკალარული გამრავლება
.*	მასივური გამრავლება
/	სკალარული მარჯვნივ გაყოფა
./	მასივური მარჯვნივ გაყოფა
\	სკალარული მარცხნივ გაყოფა
.\	მასივური მარცხნივ გაყოფა
^	სკალარული ახარისხება
.^	მასივური ახარისხება

ბრძანებები და ფუნქციები:

/	სკალარული მარჯვნივ გაყოფა
./	მასივური მარჯვნივ გაყოფა
abs	გამოითვლის მოდულს
acos	ითვლის არკკოსინუსს
acosh	ითვლის ჰიპერბოლურ კოსინუსს
angle	ითვლის კომპლექსური რიცხვის ფაზურ კუთხეს
ans	აგებს გამოსახულების მნიშვნელობას
asin	ითვლის არკსინუსს
asinh	ითვლის ჰიპერბოლური სინუსის შებრუნებულს
atan	ითვლის არკტანგენსს მეოთხედში
atan2	ითვლის არკტანგენსს მეოთხედში
atanh	გამოითვლის ჰიპერბოლური ტანგენსის შებრუნებულს
ceil	ამრგვალებს ∞ მიმართულებით
clock	გვაძლევს მიმდინარე დროს
conj	გვაძლევს კომპლექსური რიცხვის შეუღლებულს
cos	გვაძლევს კუთხის კოსინუსს
cosh	გვაძლევს ჰიპერბოლურ კოსინუს
date	გვაძლევს მიმდინარე თარიღს
eps	ათწილადურ სიზუსტეს
exp	ითვლის ხარისხს e ფუძით
eye	ქმნის ერთეულოვან მატრიცას
function	ქმნის ახალ ფუნქციას MATLAB-ში
fix	ამრგვალებს ნულის მიმართულებით

floor	ამრგვალებს - ∞ მიმართულებით
i	კვადრატული ფესვი –1 დან
imag	ითვლის კომპლექსური რიცხვის წარმოსახვით ნაწილს
inf	უსასრულობა ∞
j	კვადრატული ფესვი –1 –დან
log	გამოითვლის ლოგარიტჰმს ნატურალური ფუძით
log10	გამოითვლის ლოგარიტჰმს 10 ფუძით
magic	აწარმოებს მაგიურ კვადრატს
NaN	განუზღვრელობა
nargin	გვაძლევს ფუნქციის არგუმენტების რაოდენობას
nargout	განსაზღვრავს ფუნქციის გამოსავალი არგუმენტების რაოდენობას
ones	ქმნის მატრიცას, რომლის ყველა ელემენტი 1-ის ტოლია
pascal	ქმნის პასკალის სამკუთხედა მატრიცაში
pi	π
real	ითვლის კომპლექსური რიცხვის ნამდვილ ნაწილს
rem	ითვლის მთელი რიცხვების გაყოფის შედეგად მიღებულ ნაშტს
round	ამრგვალებს უახლოეს მთელამდე
sign	გვაზლევს –1, 0 ან 1
sin	ითვლის კუთხის სინუსს
sinh	ითვლის ჰიპერბოლურ სინუსს
sqrt	ითვლის კვადრატუ ფესვს
tan	ითვლის კუთხის ტანგენსს
tanh	ითვლის ჰიპერბოლურ ტანგენსს
zeros	ქმნის მატრიცას, რომლის ყველა ელემენტი 0-ის ტოლია

ამოცანები

პრობლემა 1-8 უკავშირდება ამ თავში განხილულ ამოცანებს, ხოლო 9-23 მიეკუთვნება განსხვავებულ საინჟინრო პრობლემებს.

ეს პრობლემები დაკავშირებულია ექოს შემცველ სიგნალთან. გარდა იმისა, რომ შევქმნათ სიგნალი, რომელიც ექოს შეიცავს, უნდა შევძლოთ ექოს შემცველი სიგნალის მოდიფიცირება.

1. აწარმიეთ და ააგეთ 500 წერტილი სიგნალისა, რომელიც შეიცავს საწყის სიგნალს, რომელიც წარმოდგენილია ამ თავში განხილულ პრობლემაში ექო, ექოს, რომელიც შესუსტებულია ფაქტორით 0.3 და დაგვიანებულია 3 წამით.
2. შექმენით და ააგეთ 500 წერტილი სიგნალისა, რომელიც შეიცავს წინა ამოცანაში მითითებულ საწყის სიგნალს, ჩახვეულ ექოს სკალირებულს ფაქტორით 0.6 და დაგვიანებულს 5.5 წამით
3. აწარმიეთ და ააგეთ 500 წერტილი სიგნალისა, რომელიც შეიცავს საწყის სიგნალს, რომელიც წარმოდგენილია ამ თავში განხილულ პრობლემაში ექო და კიდევ 2 ექოს. პირველი დაგვიანებულია 0.5 წამით და შესუსტებულია ფაქტორით 0.7, მეორე – დაგვიანებულია 4 წამით და შესუსტებული ფაქტორით 0.1.

4. შექმენით და ააგეთ 500 წერტილი სიგნალისა, რომელიც შეიცავს მხოლოდ3 პრობლემაში ასახულ 2 ექოს.
5. აიღეთ სიგნალი ექოსთან ერთად, რომელიც განხილულია ამ თავში, ააგეთ ახალი სიგნალი, რომელიც შეიცავს სიგნალის და ექოების აბსოლუტურ სიდიდეებს.
6. აიღეთ სიგნალი ექოსთან ერთად, რომელიც განხილულია ამ თავში, ააგეთ ახალი სიგნალი, რომელიც შეიცავს სიგნალის და ექოების კვადრატში აყვანილ მნიშვნელობებს.

ჰიდროაკუსტიკური სიგნალი.

7. შექმენით და ააგეთ ჰიდროაკუსტიკური სიგნალი, რომელიც შეიცავს სიგნალს განხილულს ამ თავის შესაბამის განყოფილებაში (ჰიდროაკუსტიკური სიგნალი), რომელსაც მოჰყვება სიჩუმის 200 წერტილი და შემდეგ იგივე სიგნალი 2 მილიწამის ხანგრძლივობით.
8. შექმენით და ააგეთ ჰიდროაკუსტიკური სიგნალი, რომელიც შეიცავს ამ თავის შესაბამის განყოფილებაში განხილულ სიგნალს (ჰიდროაკუსტიკური სიგნალი), რომელსაც მოჰყვება სიჩუმე 1 მილიწამის ხანგრძლივობით

ფორმულები გვიჩვენებს გრადუსებში გამოსახულ ტემპეტატურათა დამოკიდებულებას სხვადასხვა სისტემაში: ფარენჰეიტის გრადუსი(T_F), ცელსიუსის გრადუსი(T_C), კელვინის გრადუსი(T_K) და რანკინის გრადუსი(T_R):

$$T_F = T_R - 459.67^oR$$
$$T_F = 9/5\ T_C + 32^oF$$
$$T_R = 9/5\ T_K$$

9. დაწერეთ პროგრამა იმისათვის, რომ შეიქმნას ცხრილი ფარენჰეიტის ტემპერატურიდან ცელსიუსის ტემპერატურაში გადასაყვანად 0^o დან 100^o –მდე ყოველი 5 გრადუსის შემდეგ.

10. დაწერეთ პროგრამა იმისათვის, რომ შეიქმნას ცხრილი ფარენჰეიტის ტემპერატურიდან კელვინის ტემპერატურაში გადასაყვანად 0^o დან 100^o –მდე ყოველი 5 გრადუსის შემდეგ.

ბაქტერიების გამრავლება. ბაქტერიების რაოდენობრივი ზრდა შეიძლება მოდელირებული იქნას შემდეგი ფორმულის მიხედვით:

$$Y_{new} = Y_{old}\, e^{1.386\, t}$$

სადაც Y_{new} ბაქტერიების ახალი რაოდენობა მოცემულ გარემოაში, Y_{old} ბაქტერიების საწყისი რაოდენობაა, t დრო საათებში.

11. ისარგებლეთ ამ განტოლებით რომ იწინასწარმეტყველოთ ბაქტერიების რაოდენობა 6 საათის შემდეგ, თუ მათი საწყისი რაოდენობა 1-ის ტოლია. დაბეჭდეთ ცხრილი, რომელიც უჩვენებს ბაქტერიების რაოდენობას ყოველი საათის შემდეგ.
12. შეცვალე 11 დავალება ისე, რომ მომხმარებელმა თვითონ შეძლოს დროის მითითება საათებში.
13. შეცვალე 11 დავალება ისე, რომ მომხმარებელმა მიუთითოს დრო წუთებში, თუმცა ფორმულა დროს სათებში აღიქვამს.

14. შეცვაე 11 დავალება ისე, რომ მომხმარებელს შეეძლოს ბაქტერიების საწყისი რაოდენობის მითითება.
15. შეცვალე 11 დავალება ისე, რომ მომხმარბელმა მიუთითოს დროის ორი მნიშვნელობა, პროგრამამ კი დაითვალოს დროის ამ ინტერვალში წარმოქმნილი ბაქტერიების რაოდენობა.

დათარიღება ნახშირბადის შემცველობის მიხედვით.

ეს არის მეთოდი, რომლის საშუალებითაც ხდება ორგანული ნაერთების ნაშთის, მაგალითად ნიჟარა, მცენარის თესლი, ასაკის დადგენა. ეს მეთოდი ადგენს რადიაციული ნახშირბადის carbon14 წილობრივ შემცველობას მოცემულ ნიმუშში. განტოლება იძლევა ნარჩენის ასაკს წლებში:

$$age = \frac{-\log_e(carbon14 \cdot proportion \cdot remaining)}{0.0001216}$$

carbon14 სიცოცხლის ხანგრძლივობა დაახლოებით 11400 წელია. თუ პროპორციულ შემცველობას ავიღებთ 0.5, ასაკი ტოლი იქნება 5700.22.

16. დაწერეთ პროგრამა, რომელიც მომხმარებელს საშუალებას მისცემს მიუთითოს ნარჩენი carbon14 პროპორციული შემცველობა ეს სიდიდე უნდა იყოს 0 და 1 –ს შორის). დაბეჭდეთ ნიმუშის ასაკი.
17. შეცვალე 16 დავალება ისე, რომ მიღებული შედეგი დამრგვალდეს უახლოეს წლამდე.
18. შეცვალე 16 დავალება ისე, რომ მიღებული შედეგი დამრგვალდეს უახლოეს საუკუნემდე.

სიხშირული გაზომვები.

სინუსოიდის სიხშირე შეიძლება მიცემული იყოს ციკლების რაოდენობით წამში (ჰერცი) ან რადიანებით წამში, სადაც ერთი ციკლი წამში ტოლია 2π რადიანისა წამში.

19. დაწერეთ პროგრამა, რომელიც საშუალებას მოგცემთ მიუთითოთ სიხშირე ციკლი/წამში და გადაიყვანოთ იგი ერთეულში რადიანი/წამში.
20. დაწერეთ პროგრამა, რომელიც საშუალებას მოგცემთ მიუთითოთ სიხშირე რადიანი/წამში და გადაიყვანოთ იგი ჰერცებში.
21. ნორმირებული სიხშირე ტოლია ωT, სადაც ω არის სიხშირე რადიანი/წამში და T გამოსახულია წამებში. ე.ი. ნორმირებული სიხშირის ერთეულია რადიანი. დაწერეთ პროგრამა, რომელიც საშუალებას იძლევა მიუთითოთ სიხშირე რადიანი/წამში და T სიდიდე და გამოითვალოთ შესაბამისი ნორმირებული სიხშირე.
22. შეცვალეთ 21 დავალება ისე, რომ მიუთითოთ სიხშირე ჰერცებში.
23. შეცვალეთ 21 დავალება ისე, რომ ნორმირებული სიხშირე ყოველთვის მიიღოთ შუალედში 0 - 2π. (გამოაკელით მიღებულ შედეგს 2π ჯერადი მნიშვნელობა). მაგალითად: თუ ნორმირებული სიხშირე ტოლია 2.5π, ჩვენთვის სასურველი შედეგი უნდა იყოს 0.5π; თუ ნორმირებული სიხშირე ტოლია 7π, უნდა მივიღოთ - π. (მითითება: გაიხსენეთ ფუნქცია rem)