INDIAN INSTITUTE OF TECHNOLOGY BHUBANESWAR

Signals and Systems Laboratory

School of Electrical Sciences

Name: Shorya Sharma
Roll Number: 19EE01017

Assignment 1

Generation of Standard Signals

Aim

- To generate standard signals without using built-in command in MATLAB using user-specified parameters.
- To understand and realize even and odd parts of the signals implemented above.
- To perform three fundamental signal transformation operations in the time domain with user specified input and comparing with their theoretical outcomes.

Theory: -

SIGNALS: Signals are detectable physical quantities that contains information about time varying phenomena.

SYSTEM: A system is any process that produces an output signal in response to an input signal.

Equations of Standard Signals:

- Sine: y = Asin(2πft) (1)
- Square: $y(t)$ = { A, $t \in$ (− T 2 , T 2) 0, $Otherwise$ (2)
- Saw tooth: y = ∑ $y(t - nT)$ n = ∞ n= −∞ , $where\ y(t)$ = { At, $t \in$ (0, T) 0, $Otherwise$ (3)
- Triangular: y = ∑ $y(t - nT)$ n = ∞ n= −∞ , $where\ y(t)$ = { At, $t \in$ (0, T/2) $A(T - t)$, $t \in$ (T/2, T) (4)
- Impulse: $y(t)$ = { 1, t = 0 0, $Otherwise$ (5)
- Step: $y(t)$ = { A, $t \geq$ 0 0, $Otherwise$ (6)
- Pulse: y = ∑ $y(t - nT)$ n = ∞ n= −∞ , $where\ y(t)$ = { At, $t \in$ (0, βT) 0, $t \in$ (βT, T)

$Where, \beta$: $Duty\ Cycl$e

Even and Odd Components of a signal:

If a signal is termed as x(t) then it's even and odd components can be given by the following equations:

$x_e(t) = 0.5(x(t) + x(-t))$

$x_o(t) = 0.5(x(t) - x(-t))$

Signal Transformation:

- Amplitude Scaling:

  If a signal x(t) is amplitude-scaled by A, then its output will be Ax(t).
- Time scaling:

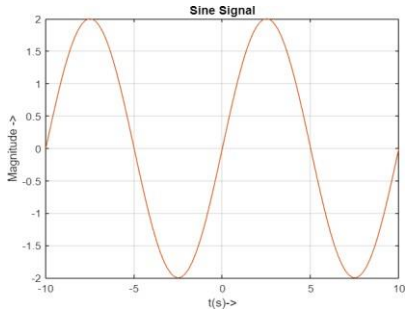
  If a signal x(t) is expanded by a factor τ, then its equation will be x (t/τ) and vice versa.
- Time shifting:

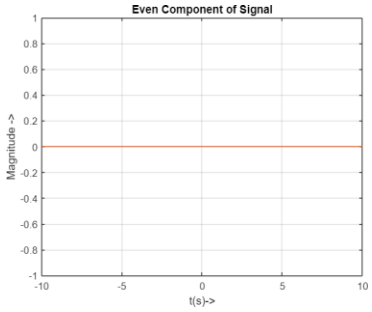
  If a signal x(t) is delayed by a factor τ, then its equation will be x(t − τ) and vice versa.
  Folding:

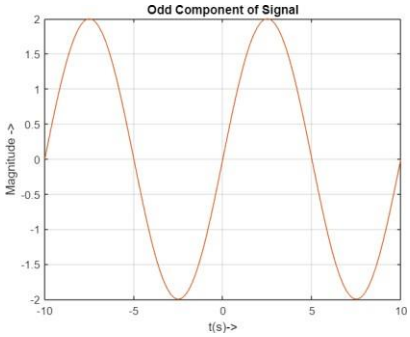
- A signal x(t), after folding, becomes x(−t).

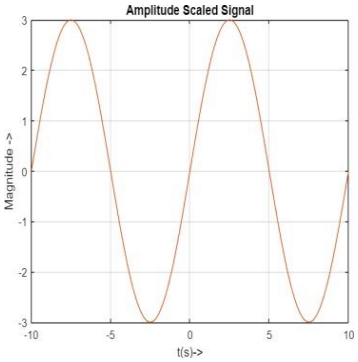
Results:

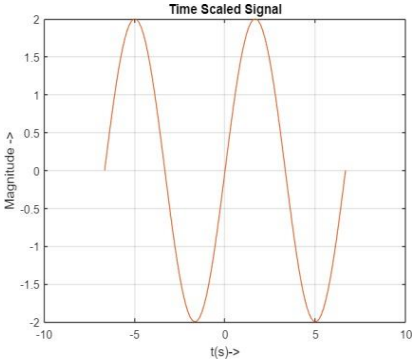
Sine

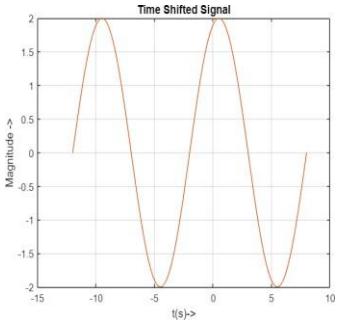


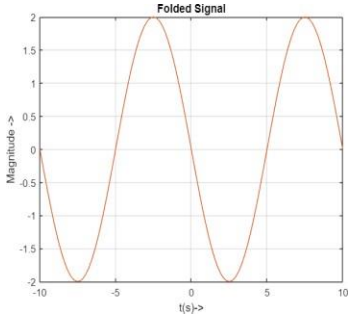




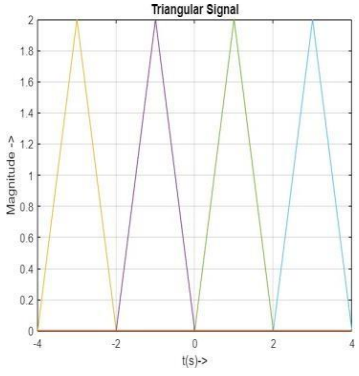


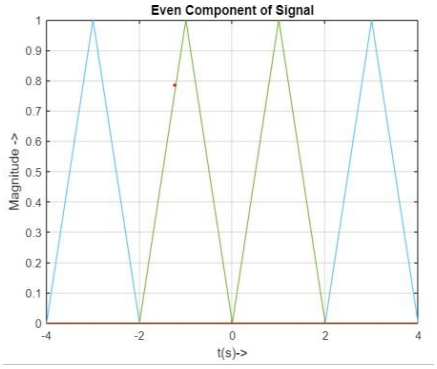


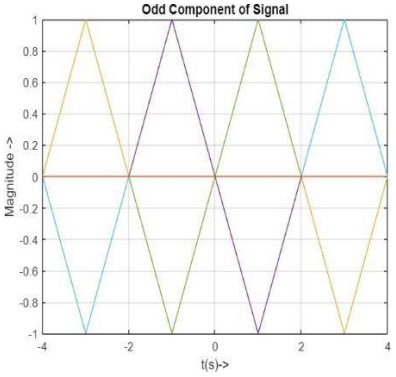


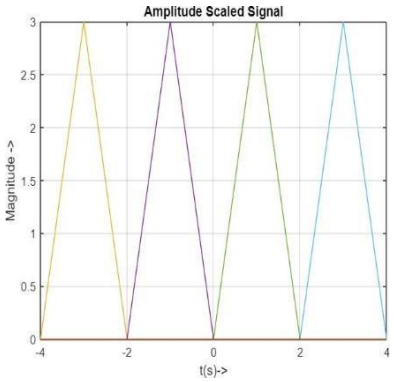


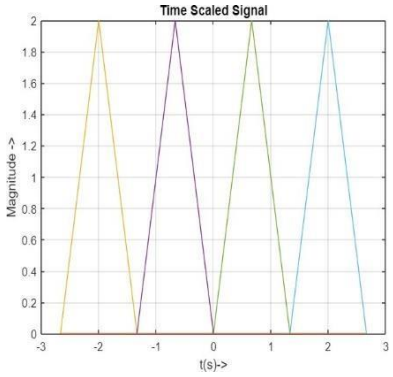
Triangular

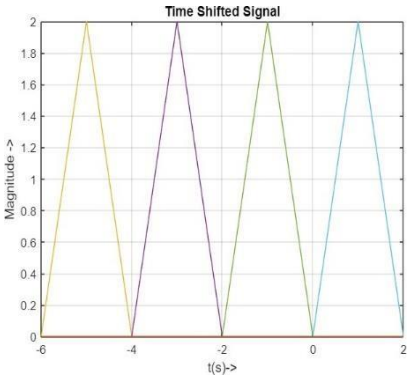


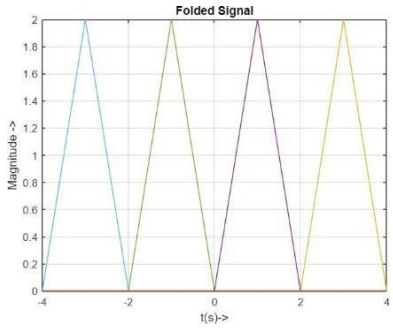




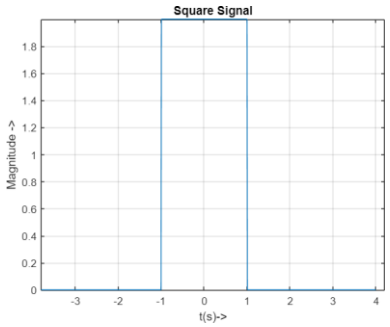

Amplitude Scaled Signal
Magnitude ->
t(s)->
3
2.5
2
1.5
1
0.5
0
-4
-2
0
2
4

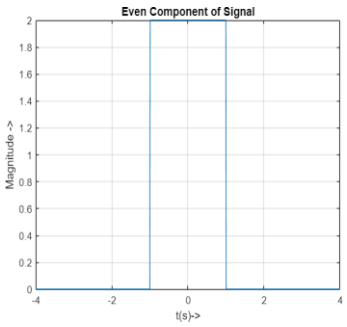

Time Scaled Signal
Magnitude ->
t(s)->
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-3
-2
-1
0
1
2
3

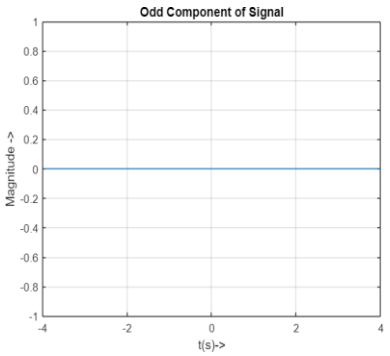

Time Shifted Signal
Magnitude ->
t(s)->
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-6
-4
-2
0
2

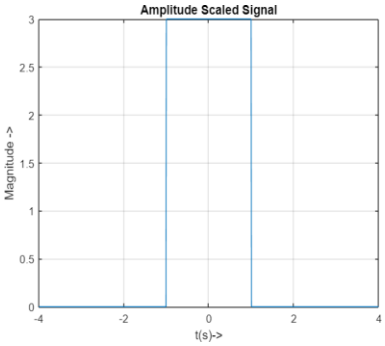

Folded Signal
Magnitude ->
t(s)->
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-4
-2
0
2
4

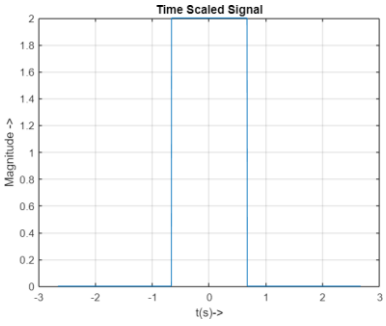
Square

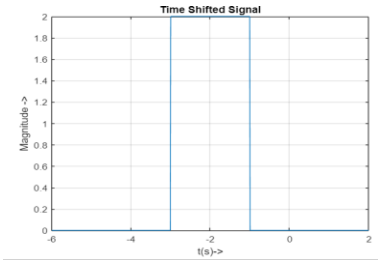

Square Signal
Magnitude ->
t(s)->
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-3
-2
-1
0
1
2
3
4

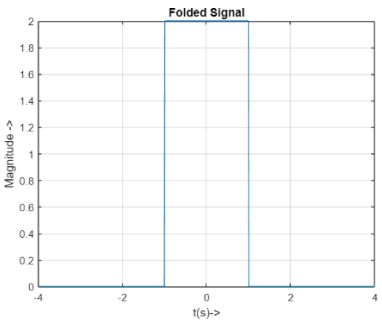

Even Component of Signal
Magnitude ->
t(s)->
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-4
-2
0
2
4


Odd Component of Signal
Magnitude ->
t(s)->
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-4
-2
0
2
4

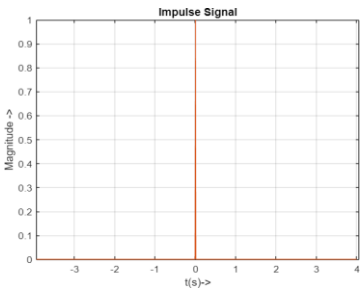

Amplitude Scaled Signal
Magnitude ->
t(s)->
3
2.5
2
1.5
1
0.5
0
-4
-2
0
2
4

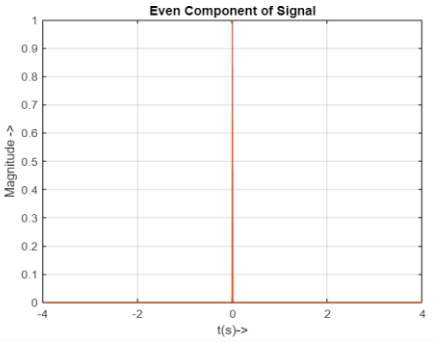

Time Scaled Signal
Magnitude ->
t(s)->
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-3
-2
-1
0
1
2
3

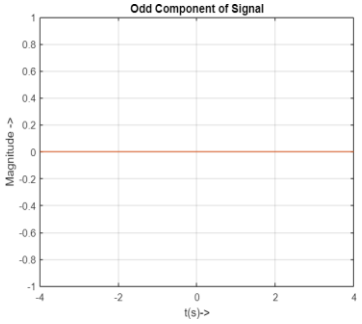

Time Shifted Signal
Magnitude ->
t(s)->
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-6
-4
-2
0
2

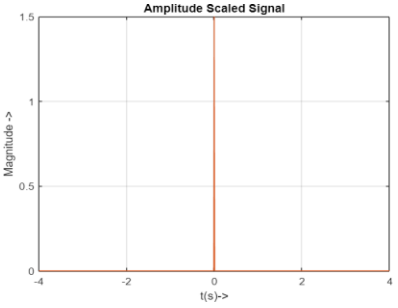

Folded Signal
Magnitude ->
t(s)->
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-4
-2
0
2
4

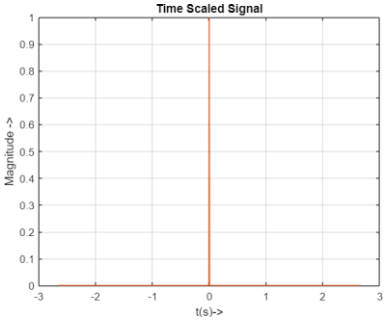
Impulse

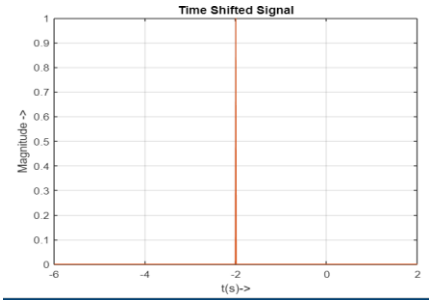

Impulse Signal
Magnitude ->
t(s)->
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-3
-2
-1
0
1
2
3
4

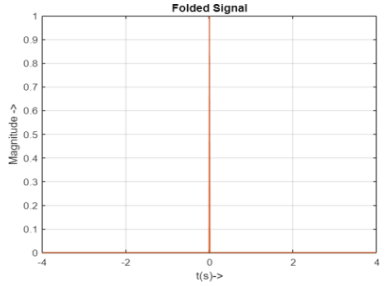

Even Component of Signal
Magnitude ->
t(s)->
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-4
-2
0
2
4


Odd Component of Signal
Magnitude ->
t(s)->
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-4
-2
0
2
4

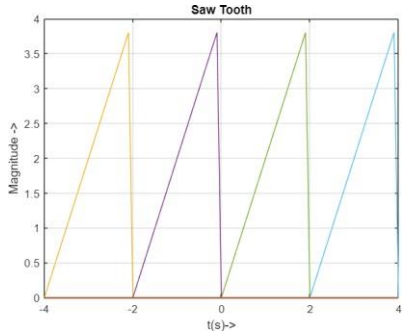

Amplitude Scaled Signal
Magnitude ->
t(s)->
1.5
1
0.5
0
-4
-2
0
2
4

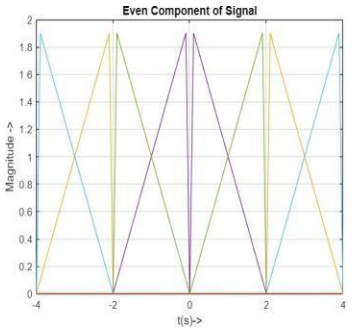

Time Scaled Signal
Magnitude ->
t(s)->
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-3
-2
-1
0
1
2
3

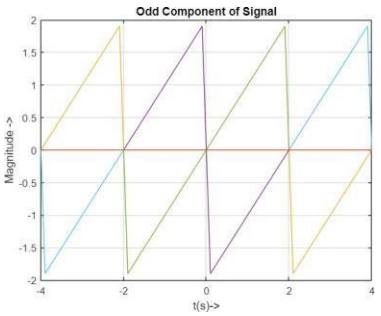

Time Shifted Signal
Magnitude ->
t(s)->
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-6
-4
-2
0
2

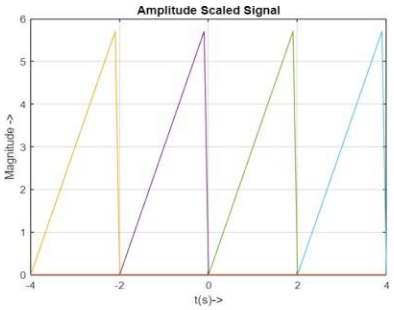

Folded Signal
Magnitude ->
t(s)->
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-4
-2
0
2
4

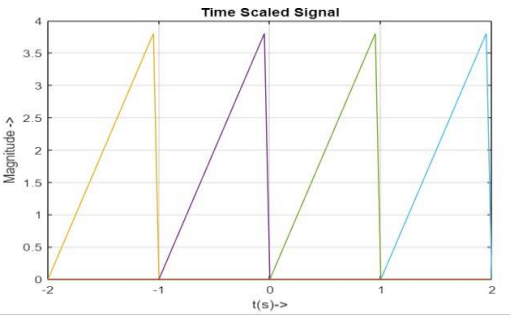
Saw Tooth

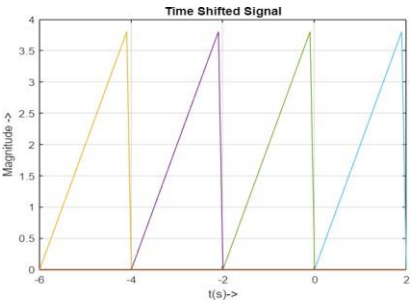


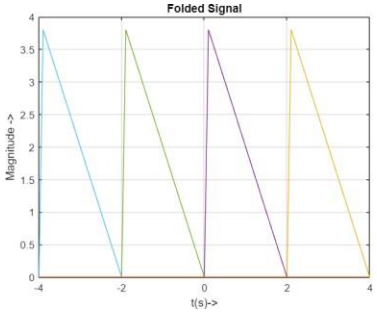




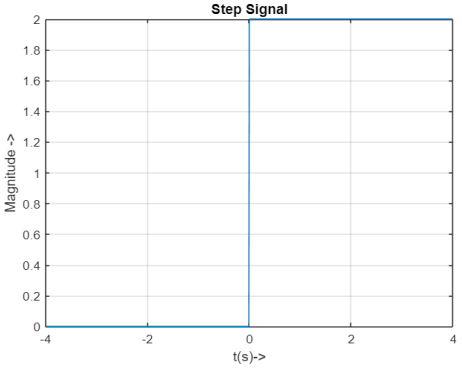


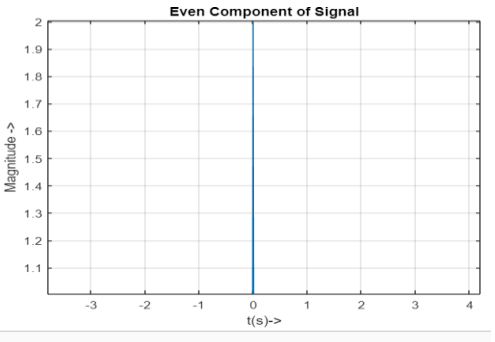


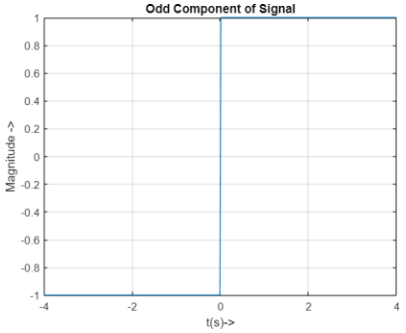


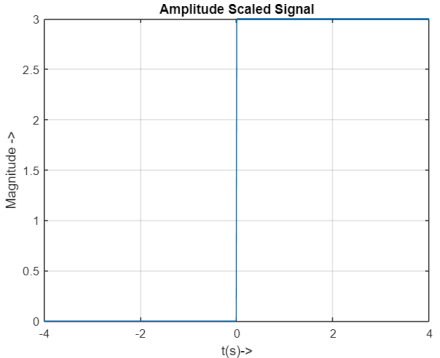


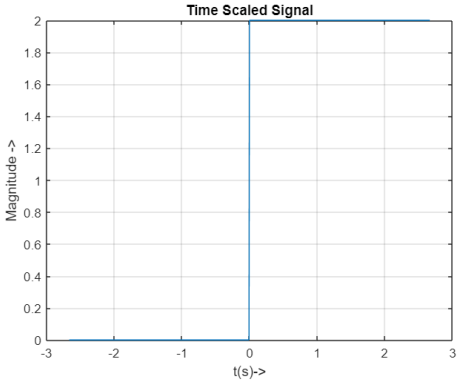
Step

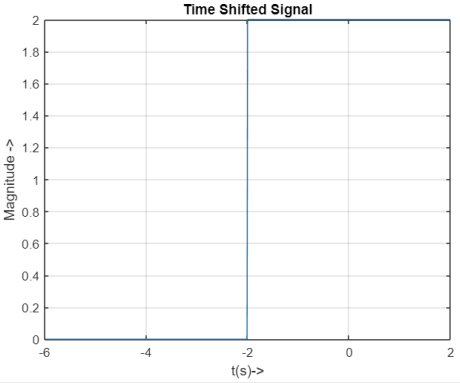


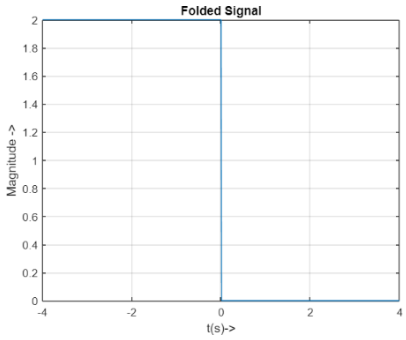




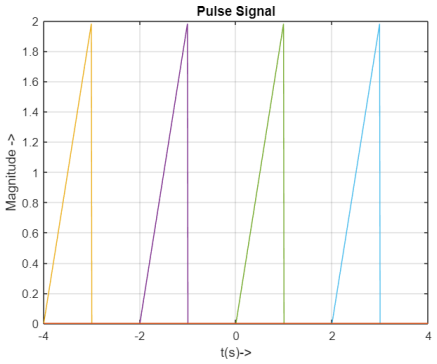


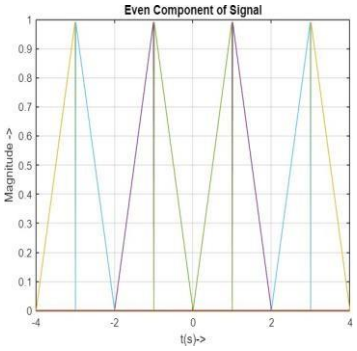


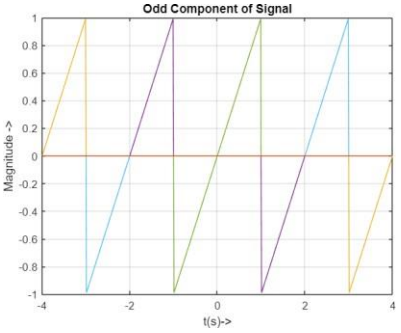


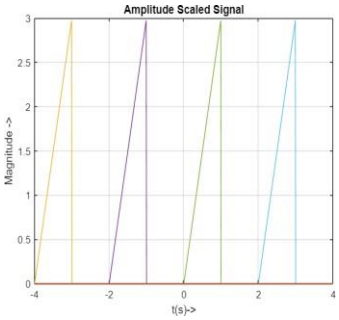


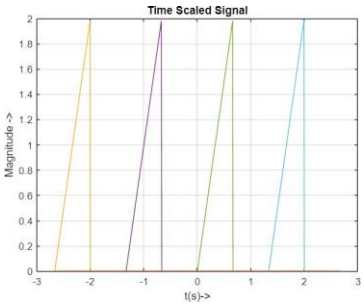
Pulse

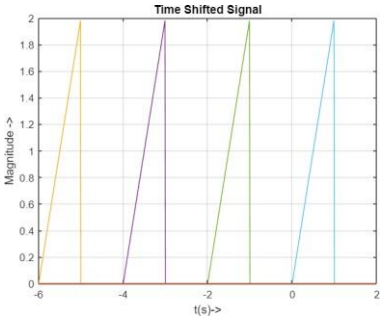


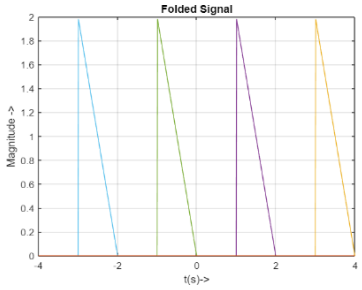












Discussions:

All mathematical operations yield graphs with alterations but preserving the originality, useful in many electrical applications.

Conclusion

From this experiment, we learnt how to generate standard signals without using inbuilt commands and also

performed combinations of transformation operations on all the signals. In addition to the sin and cos functions in MATLAB, the toolbox offers other functions that produce periodic signals such as sawtooth and square.

Appendix:

1) Sine

```
to=input('Enter the Lower Bound of time in seconds:');
tf=input('Enter the Upper Bound of time in seconds:');
A=input('Enter Amplitude of function');
T=input('Enter Time Period of function');
Ai=input('Enter Amplitude to be scaled');
Ti=input('Enter Time scaling factor');
Ts=input('Enter time delaying factor with sign');
t=to:0.01:tf;
[x,z]=size(t);
y=t;
% This portion is common for all signals. Other signals can be generated by changing the definition of y
y=A*sin(2*pi*f*t);
figure(1)
plot(t,y);
hold on;
title('Sine Signal')
xlabel('t(s)->');
ylabel('Magnitude ->');
grid on;
figure(2)
plot(t,even(y));
hold on;
title('Even Component of Signal')
xlabel('t(s)->');
ylabel('Magnitude ->');
grid on;
figure(3)
plot(t,odd(y));
hold on;
title('Odd Component of Signal')
xlabel('t(s)->');
ylabel('Magnitude ->');
grid on;
figure(4)
plot(t,amplitude(y,Ai));
hold on;
title('Amplitude Scaled Signal')
xlabel('t(s)->');
ylabel('Magnitude ->');
grid on;
figure(5)
plot(TimeScaling(t,Ti),y);
hold on;
title('Time Scaled Signal')
xlabel('t(s)->');
ylabel('Magnitude ->');
grid on;
figure(6)
plot(TimeShifting(t,Ts),y);
hold on;
title('Time Shifted Signal')
xlabel('t(s)->');
ylabel('Magnitude ->');
grid on;
SINE
figure(7)
plot(folding(t),y);
hold on;
title('Folded Signal')
xlabel('t(s)->');
ylabel('Magnitude ->');
grid on;
ax=gca;
c=ax.FontSize;
ax.FontSize=20;
axis([t(1) t(z) -(A*T)-1 (A*T)+1])
function od=odd(y)
[x,z]=size(y);
c=0;
for i=1:z
od(i)=(y(1+c)-y(z-c))/2;
c=c+1;
end
end
function ev=even(y)
[x,z]=size(y);
c=0;
for i=1:z
ev(i)=(y(1+c)+y(z-c))./2;
c=c+1;
end
end
function time=TimeScaling(t,c)
[x,z]=size(t);
for i=1:z
time(i)=t(i)./c;
end
end
function time=TimeShifting(t,c)
[x,z]=size(t);
for i=1:z
time(i)=t(i)-c;
end
end
function fold=folding(t)
[x,z]=size(t);
for i=1:z
fold(i)=-t(i);
end
end
function amp=amplitude(y,A)
[x,z]=size(y);
for i=1:z
amp(i)=A*y(i);
end
end
```

2) Triangular

```
to=input('Enter the Lower Bound of time in seconds:'); tf=input('Enter the Upper Bound of time in seconds:'); A=input('Enter Amplitude of function');
T=input('Enter Time Period of function'); Ai=input('Enter Amplitude to be scaled'); Ti=input('Enter Time scaling factor'); Ts=input('Enter time delaying factor with sign'); t=to:0.01:tf;
[x,z]=size(t); y=t;
% This portion is common for all signals. Other signals can be generated by changing the definition of y.

for n=-200:200 for i=1:z
if((t(i)-(n*T))>=0 && (t(i)-(n*T))<=T/2) y(i)=A*(t(i)-(n*T));
elseif((t(i)-(n*T))>=T/2 && (t(i)-(n*T))<=T)
y(i)=A*(T-(t(i)-(n*T)));
else y(i)=0;
end end

figure(1) plot(t,y); hold on;
title('Sine Signal') xlabel('t(s)->'); ylabel('Magnitude ->'); grid on;
figure(2) plot(t,even(y)); hold on;
title('Even Component of Signal') xlabel('t(s)->'); ylabel('Magnitude ->');
grid on; figure(3) plot(t,odd(y)); hold on;
title('Odd Component of Signal') xlabel('t(s)->'); ylabel('Magnitude ->');
grid on; figure(4)
plot(t,amplitude(y,Ai)); hold on;
title('Amplitude Scaled Signal') xlabel('t(s)->'); ylabel('Magnitude ->');
grid on; figure(5)
plot(TimeScaling(t,Ti),y); hold on;
title('Time Scaled Signal') xlabel('t(s)->'); ylabel('Magnitude ->'); grid on;
figure(6) plot(TimeShifting(t,Ts),y);

hold on;
title('Time Shiftsd Signal') xlabel('t(s)->'); ylabel('Magnitude ->');
grid on; figure(7)
plot(folding(t),y); hold on;
title('Folded Signal') xlabel('t(s)->'); ylabel('Magnitude ->'); grid on;
c=ax.FontSize; ax.FontSize=20;
axis([t(1) t(z) -(A*T)-1 (A*T)+1])

function od=odd(y) [x,z]=size(y); c=0;
for i=1:z
od(i)=(y(1+c)-y(z-c))/2; c=c+1;
end end

function ev=even(y) [x,z]=size(y); c=0;
for i=1:z ev(i)=(y(1+c)+y(z-c))./2; c=c+1;
end end

function time=TimeScaling(t,c) [x,z]=size(t);
for i=1:z time(i)=t(i)./c; end
end

function time=TimeShifting(t,c) [x,z]=size(t);
for i=1:z time(i)=t(i)-c; end
end

function fold=folding(t) [x,z]=size(t);
for i=1:z fold(i)=-t(i); end
end

function amp=amplitude(y,A) [x,z]=size(y);
for i=1:z

amp(i)=A*y(i); end
end
```

3) Impulse

```
t=-10:0.001:10;y=zeros(1,length(t));
for i=1:length(t)
 if(t(i)==0)
 y(i)=1;
 else
 y(i)=0;
 end
end
plot(t,y);title("IMPULSE WAVE FUNCTION");xlabel("time(t) -->");ylabel("Output
Signal -->");
e=evenFunction(y);plot(t,e,"LineWidth",2);title("EVEN PART OF IMPULSE
WAVEFUNCTION");xlabel("time(t) -->");ylabel("Output Signal -->");o=odd(y);
plot(t,o,"LineWidth",2);title("ODD PART OF IMPULSE
WAVEFUNCTION");xlabel("time(t) -->");ylabel("Output Signal -->");
factor=input("Enter time scaling factor");tscale=timeScaling(t,factor);
plot(tscale,y,"LineWidth",2);title("TIME SCALED IMPULSE WAVEFUNCTION BY A
FACTOR OF "+factor);xlabel("time(t) -->");ylabel("Output Signal -->")
k = input("enter amplitude scaling factor");a=amplitude_scaling(y,k);
plot(t,a,"LineWidth",2);
title("AMPLITUDE SCALING IMPULSE WAVEFUNCTION BY A FACTOR OF "+k);
xlabel("time(t) -->");ylabel("Output Signal -->");
d=input("enter time(in milli-seconds) by which the signal is to be delayed");
ts=time_shifting(t,d);grid on;plot(ts,y,"LineWidth",2);
title("TIME SHIFTED IMPULSE WAVEFUNCTION DELAYED BY "+d+" mS");
xlabel("time(t) -->");ylabel("Output Signal -->");fold=folding(t);
plot(fold,y,"LineWidth",2);title("FOLDED IMPULSE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
tscale2=timeScaling(ts,factor);plot(tscale2,y,"LineWidth",2);
title("(Time shifting + Time scaling) IMPULSE WAVEFUNCTION");
xlabel("time(t) -->");
ylabel("Output Signal -->");fold2=folding(ts);plot(fold2,y,"LineWidth",2);
title("(Time shifting + Folding) IMPULSE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
plot(tscale,a,"LineWidth",2);
title("(Amplitude scaling + Time Scaling) IMPULSE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");fold3=folding(tscale);
plot(fold3,y,"LineWidth",2);title("(Time scaling + Folding) IMPULSE
WAVEFUNCTION");xlabel("time(t) -->");ylabel("Output Signal -->");
plot(fold,a,"LineWidth",2);
title("(Amplitude scaling + Folding) IMPULSE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");plot(ts,a,"LineWidth",2);
title("(Amplitude scaling + time shifting) IMPULSE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
```

4) SAWTOOTH

```
f=input("enter frequency of signal");A=input("enter amplitude of signal");
t1=input("Beginning time in milli-seconds");t2=input("ending time in milliseconds");T=1/f;t=t1:0.003:t2; l=length(t); y=zeros(1,l);
for i=-100:100
 for j=1:l
 p=t(j)-i*T;
 if(p>0 && p<T)
 y(j)= A*p;
 end
 end
end
plot(t,y);title("SAWTOOTH WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");e=evenFunction(y);plot(t,e);
title("EVEN PART OF SAWTOOTH WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");o=odd(y);plot(t,o);
title("ODD PART OF SAWTOOTH WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");factor=input("Enter time scaling factor");
tscale=timeScaling(t,factor);plot(tscale,y);
title("TIME SCALED SAWTOOTH WAVEFUNCTION BY A FACTOR OF "+factor);
xlabel("time(t) -->");ylabel("Output Signal -->")
k = input("enter amplitude scaling factor");a=amplitude_scaling(y,k);
plot(t,a);
title("AMPLITUDE SCALING SAWTOOTH WAVEFUNCTION BY A FACTOR OF "+k);
xlabel("time(t) -->");ylabel("Output Signal -->");
d=input("enter time(in milli-seconds) by which the signal is to be delayed");
ts=time_shifting(t,d);plot(ts,y);
title("TIME SHIFTED SAWTOOTH WAVEFUNCTION DELAYED BY "+d+" mS");
xlabel("time(t) -->");ylabel("Output Signal -->");fold=folding(t);
plot(fold,y);title("FOLDED SAWTOOTH WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");tscale2=timeScaling(ts,factor);
plot(tscale2,y);title("(Time shifting + Time scaling) SAWTOOTH WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");fold2=folding(ts);
plot(fold2,y);title("(Time shifting + Folding) SAWTOOTH WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
plot(tscale,a);title("(Amplitude scaling + Time Scaling) SAWTOOTH
WAVEFUNCTION");xlabel("time(t) -->");ylabel("Output Signal -->");
fold3=folding(tscale);plot(fold3,y);
title("(Time scaling + Folding) SAWTOOTH WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");plot(fold,a);
title("(Amplitude scaling + Folding) SAWTOOTH WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
plot(ts,a);title("(Amplitude scaling + time shifting) SAWTOOTH WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
```

5) SQUARE

```
f=input("Enter frequency of signal");A=input("Enter amplitude of signal");
T=1/f;t=linspace(-T,T,200);l=length(t);y=zeros(1,l);
for i=1:l
 if t(i)>(-T/2) && t(i)<(T/2)
 y(i)=A;
 else
 y(i)=0;
 end
end
plot(t,y,"LineWidth",2);title("SQUARE WAVE FUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
e=evenFunction(y);plot(t,e,"LineWidth",2);
title("EVEN PART OF SQUARE WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");o=odd(y);plot(t,o,"LineWidth",2);
title("ODD PART OF SQUARE WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");factor=input("Enter time scaling factor");
tscale=timeScaling(t,factor);plot(tscale,y,"LineWidth",2);
title("TIME SCALED SQUARE WAVEFUNCTION BY A FACTOR OF "+factor);
xlabel("time(t) -->");ylabel("Output Signal -->");
k = input("enter amplitude scaling factor");a=amplitude_scaling(y,k);
plot(t,a,"LineWidth",2);
title("AMPLITUDE SCALING SQUARE WAVEFUNCTION BY A FACTOR OF "+k);
xlabel("time(t) -->");ylabel("Output Signal -->");
d=input("enter time(in milli-seconds) by which the signal is to be delayed");
ts=time_shifting(t,d);grid on;plot(ts,y,"LineWidth",2);
title("TIME SHIFTED SQUARE WAVEFUNCTION DELAYED BY "+d+" mS");
xlabel("time(t) -->");ylabel("Output Signal -->");fold=folding(t);
plot(fold,y,"LineWidth",2);title("FOLDED SQUARE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
tscale2=timeScaling(ts,factor);plot(tscale2,y,"LineWidth",2);
title("(Time shifting + Time scaling) SQUARE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");fold2=folding(ts);
plot(fold2,y,"LineWidth",2);title("(Time shifting + Folding) SQUARE
WAVEFUNCTION");xlabel("time(t) -->");ylabel("Output Signal -->");
plot(tscale,a,"LineWidth",2);title("(Amplitude scaling + Time Scaling) SQUARE
WAVEFUNCTION");xlabel("time(t) -->");ylabel("Output Signal -->");
fold3=folding(tscale);plot(fold3,y,"LineWidth",2);
title("(Time scaling + Folding) SQUARE WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");plot(fold,a,"LineWidth",2);
title("(Amplitude scaling + Folding) SQUARE WAVEFUNCTION");xlabel("time(t) --
>");ylabel("Output Signal -->");plot(ts,a,"LineWidth",2);
title("(Amplitude scaling + time shifting) SQUARE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
```

6) STEP

```
A=input("enter amplitude");t1=input("enter starting time");
t2=input("enter ending time");t=t1:0.01:t2;y=zeros(1,length(t));
for i=1:length(t)
 if(t(i)>=0)
 y(i)=A;
 end
end
plot(t,y,"LineWidth",3);title("STEP WAVE FUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");e=evenFunction(y);plot(t,e,"LineWidth",3);
title("EVEN PART OF STEP WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");o=odd(y);plot(t,o,"LineWidth",3);
title("ODD PART OF STEP WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");factor=input("Enter time scaling factor");
tscale=timeScaling(t,factor);plot(tscale,y,"LineWidth",3);
title("TIME SCALED STEP WAVEFUNCTION BY A FACTOR OF "+factor);
xlabel("time(t) -->");ylabel("Output Signal -->");
k = input("enter amplitude scaling factor");a=amplitude_scaling(y,k);
plot(t,a,"LineWidth",3);
title("AMPLITUDE SCALING STEP WAVEFUNCTION BY A FACTOR OF "+k);
xlabel("time(t) -->");ylabel("Output Signal -->");
d=input("enter time(in milli-seconds) by which the signal is to be delayed");
ts=time_shifting(t,d);plot(ts,y,"LineWidth",3);
title("TIME SHIFTED STEP WAVEFUNCTION DELAYED BY "+d+" mS");
xlabel("time(t) -->");ylabel("Output Signal -->");fold=folding(t);
plot(fold,y,"LineWidth",3);title("FOLDED STEP WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
tscale2=timeScaling(ts,factor);plot(tscale2,y,"LineWidth",3);
title("(Time shifting + Time scaling) STEP WAVEFUNCTION");xlabel("time(t) --
>");ylabel("Output Signal -->");fold2=folding(ts);
plot(fold2,y,"LineWidth",3);title("(Time shifting + Folding) STEP
WAVEFUNCTION");xlabel("time(t) -->");ylabel("Output Signal -->");
plot(tscale,a,"LineWidth",3);title("(Amplitude scaling + Time Scaling) STEP
WAVEFUNCTION");xlabel("time(t) -->");ylabel("Output Signal -->");
fold3=folding(tscale);plot(fold3,y,"LineWidth",3);
title("(Time scaling + Folding) STEP WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");
plot(fold,a,"LineWidth",3);title("(Amplitude scaling + Folding) STEP
WAVEFUNCTION");xlabel("time(t) -->");ylabel("Output Signal -->");
plot(ts,a,"LineWidth",3);title("(Amplitude scaling + time shifting) STEP
WAVEFUNCTION");xlabel("time(t) -->");ylabel("Output Signal -->");
```

7) PULSE

```
f=input("enter frequency of signal");
A=input("enter amplitude of signal");B=input("enter duty cycle");
t1=input("Beginning time in milli-seconds");t2=input("ending time in milliseconds");
T=1/f;t=t1:0.0003:t2; l=length(t); y=zeros(1,1);
for i=-100:100
 for j=1:l
 p=(j)-i*T;
 if(p>0 && p<(B*T))
 y(j)= A*p;
 elseif(p>(B*T) && p<T)
 y(j)= 0;
 end
 end
end
plot(t,y);title("PULSE WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");e=evenFunction(y);plot(t,e);
title("EVEN PART OF PULSE WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");o=odd(y);plot(t,o);
title("ODD PART OF PULSE WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");factor=input("Enter time scaling factor");
tscale=timeScaling(t,factor);plot(tscale,y);
title("TIME SCALED PULSE WAVEFUNCTION BY A FACTOR OF "+factor);
xlabel("time(t) -->");
ylabel("Output Signal -->");k = input("enter amplitude scaling factor");
a=amplitude_scaling(y,k);plot(t,a);
title("AMPLITUDE SCALED PULSE WAVEFUNCTION BY A FACTOR OF "+k);
xlabel("time(t) -->");ylabel("Output Signal -->");
d=input("enter time(in milli-seconds) by which the signal is to be delayed");
ts=time_shifting(t,d);plot(ts,y);
title("TIME SHIFTED PULSE WAVEFUNCTION DELAYED BY "+d+" mS");xlabel("time(t) -
->");ylabel("Output Signal -->");fold=folding(t);plot(fold,y);
title("FOLDED PULSE WAVEFUNCTION");xlabel("time(t) -->");
ylabel("Output Signal -->");tscale2=timeScaling(ts,factor);plot(tscale2,y);
title("(Time shifting + Time scaling) PULSE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");fold2=folding(ts);
plot(fold2,y);title("(Time shifting + Folding) PULSE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");plot(tscale,a);
title("(Amplitude scaling + Time Scaling) PULSE WAVEFUNCTION");
xlabel("time(t) -->");
ylabel("Output Signal -->");fold3=folding(tscale);plot(fold3,y);
title("(Time scaling + Folding) PULSE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");plot(fold,a);
title("(Amplitude scaling + Folding) PULSE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");plot(ts,a);
title("(Amplitude scaling + time shifting) PULSE WAVEFUNCTION");
xlabel("time(t) -->");ylabel("Output Signal -->");
```

BASIC OPERATIONS FUNCTIONS

EVEN COMPONENT OF SIGNAL

```
function e=evenFunction(y)
 l=length(y); d=l;
 for i=1:l
 e(i)=( y(d) + y(l-d+1) )/2; d=d+1;
end
```

ODD COMPONENT OF SIGNAL

```
function od=odd(y)
l = length(y);c=0;
for i=1:l
 od(i)=(y(l+c)-y(l-c))/2; c=c+1;
end
end
AMPLITUDE SCALING
function a=amplitude_scaling(y,k)
 l=length(y);
 for i=1:l
 a(i)=y(i)*k;
 end
end
```

TIME SCALING

```
function ts=timeScaling(t,factor)
 l=length(t);
 for i=1:l
 ts(i)=t(i)/factor;
 end
end
TIME SHIFTING
function ts=time_shifting(t,d)
 l=length(t);
 for i=1:l
 ts(i)=t(i)+d;
 end
end
FOLDING
function f=folding(t)
 l=length(t);
 for i=1:l
 f(i)=t(i)*(-1);
 end
end
```