* **Part: 1: Signal Generator:**
* **Gold Code design with the size 1023 and using given polynomials:**

Using HW MatLab code, required 1023 size code can be constructed as below.

The main changes have been made in between HW2and this program is

1. Stage: Number of the stages are taken as 10 compare to previous value 3. Since given polynomial is of 10 stages.
2. Ptal1 = [3 10] which is taken from the given equation.
3. Ptap2= [2 3 6 8 9 10]: from the g(D) sequence.
4. Regi1 & regi2.

% Program\*

%

% Project\_gold sequece.m

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Spreading code initialization \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

IPOINT = 5;

n=1023;

user = 1; % number of users

stage = 10; % number of stages

ptap1 = [3 10]; % position of taps for 1st

ptap2 = [2 3 6 8 9 10]; % position of taps for 2nd

regi1 = [1 1 1 1 1 1 1 1 1 1]; % initial value of register for 1st

regi2 = [1 1 1 1 1 1 1 1 1 1]; % initial value of register for 2nd

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Generation of the gold code \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

m1 = mseq(stage,ptap1,regi1);

m2 = mseq(stage,ptap2,regi2);

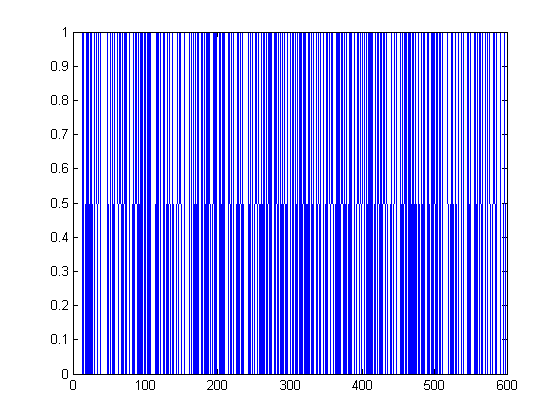
code = goldseq(m1,m2,user);

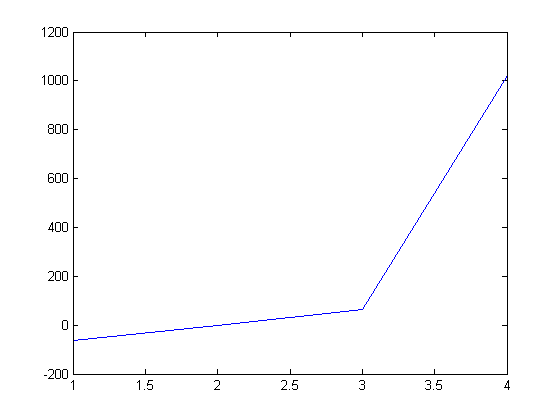
code= code\*2-1;

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* end of file \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Plotting the Autocorrelation Function.

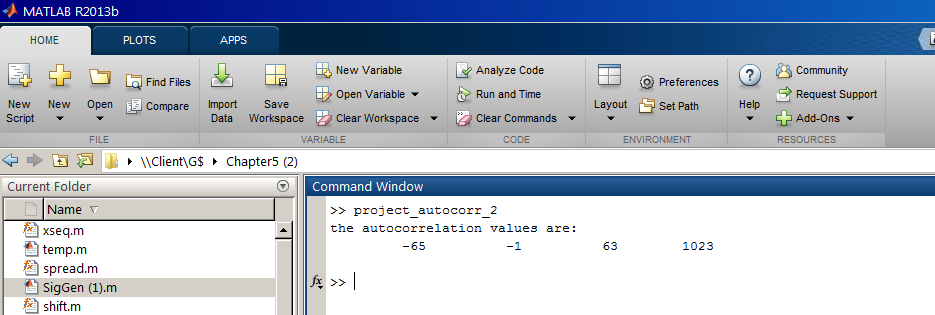
By taking the above code and taking its autocorrelation, we get the following output.





As We can see here, autocorrelaton function has total number of 4 values in the graph.

* It has peak value of 1023
* Other tab peak, it has three values as followed, as shown in the command window.
  + -65
  + -1 &
  + 63



* Increasing the Number of Sample per chip to 4 samples and plotting the autocorrelation function again.

%

% Project\_golad sequece.m

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Spreading code initialization \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

user = 1; % number of users

stage = 10; % number of stages

ptap1 = [3 10]; % position of taps for 1st

ptap2 = [2 3 6 8 9 10]; % position of taps for 2nd

regi1 = [1 1 1 1 1 1 1 1 1 1]; % initial value of register for 1st

regi2 = [1 1 1 1 1 1 1 1 1 1]; % initial value of register for 2nd

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Generation of the spreading code \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

m1 = mseq(stage,ptap1,regi1);

m2 = mseq(stage,ptap2,regi2);

code = goldseq(m1,m2,user);

code= code\*2-1;

y=interp(code,4); % increasing the number of sample per chip to 4samples

figure

plot(y)

figure

plot(autocorr(y));

vals = unique(autocorr(y));

plot(vals);

disp('the autocorrelation values are: ');

disp(vals);

% m1 = mseq(stage,ptap1,regi1);

% m2 = mseq(stage,ptap2,regi2);

% code = goldseq(m1,m2,user);

% code= code\*2-1;

% y=interp(code,4);

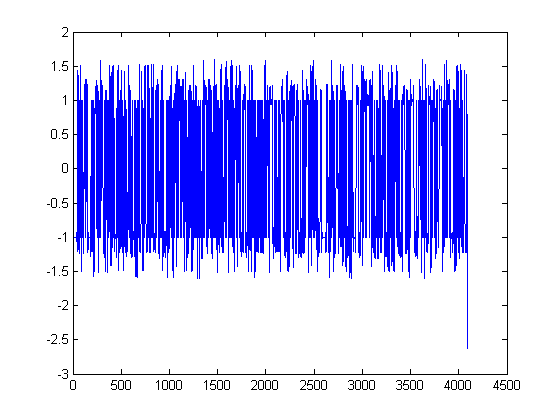
% out=oversamp(code);

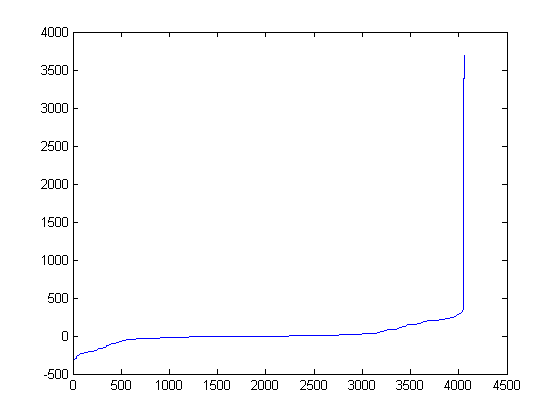
% figure

% plot(out)

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*end\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

When we generate the oversampled signal and takes autocorrelation, we may get the following output.





* **Adding Doppler Sinusoid by multiplying code signal to sinusoid.**

%

% Project\_golad sequece.m

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Spreading code initialization \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

IPOINT = 5;

n=1023;

user = 1; % number of users

stage = 10; % number of stages

ptap1 = [3 10]; % position of taps for 1st

ptap2 = [2 3 6 8 9 10]; % position of taps for 2nd

regi1 = [1 1 1 1 1 1 1 1 1 1]; % initial value of register for 1st

regi2 = [1 1 1 1 1 1 1 1 1 1]; % initial value of register for 2nd

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Generation of the spreading code \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

m1 = mseq(stage,ptap1,regi1);

m2 = mseq(stage,ptap2,regi2);

code = goldseq(m1,m2,user);

code= code\*2-1;

y=interp(code,4);

out=oversamp(code);

figure

plot(out)

Z=exp(2\*pi\*1i\*n\*f\*ts);

figure

plot(z);

comp = y\*z;

figure

plot(comp);

figure

plot(y)

figure

plot(autocorr(y));

vals = unique(autocorr(y));

plot(vals);

disp('the autocorrelation values are: ');

disp(vals);

By multiplying the signal we get the following output.



* **Part: 2 Study Degradation Effect:**
* Cross correlation one oversampled code with Doppler sinusoidal modulation with Doppler free oversampled code.

user = 1; % number of users

stage = 10; % number of stages

ptap1 = [3 10]; % position of taps for 1st

ptap2 = [2 3 6 8 9 10]; % position of taps for 2nd

regi1 = [1 1 1 1 1 1 1 1 1 1]; % initial value of register for 1st

regi2 = [1 1 1 1 1 1 1 1 1 1]; % initial value of register for 2nd

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Generation of the spreading code \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

m1 = mseq(stage,ptap1,regi1);

m2 = mseq(stage,ptap2,regi2);

code = goldseq(m1,m2,user);

code= code\*2-1;

y=interp(code,4);

out=oversamp(code);

figure

plot(out)

Z=exp(2\*pi\*1i\*n\*f\*ts);

figure

plot(z);

comp = y\*z;

figure

plot(comp);

a=crosscorr(y,z);

figure

plot(a);