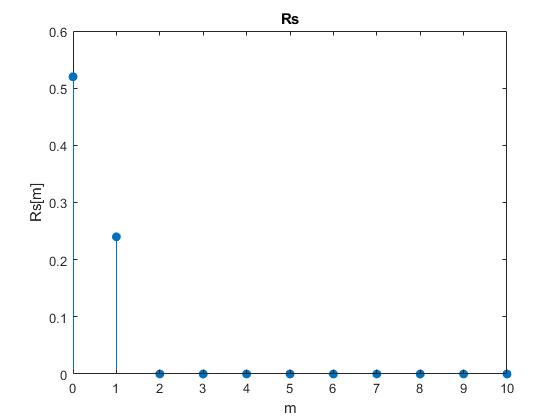
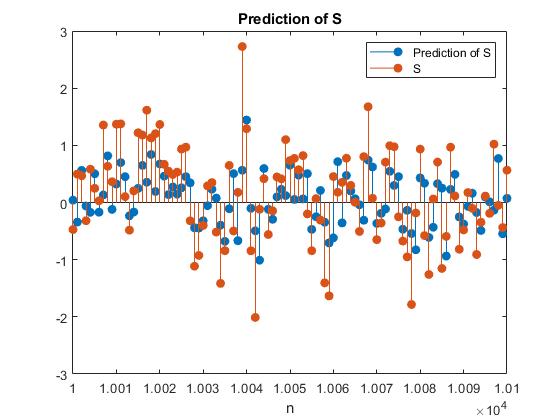
**Stochastic Process Programming Homework**

**309513047 沈衍薰**

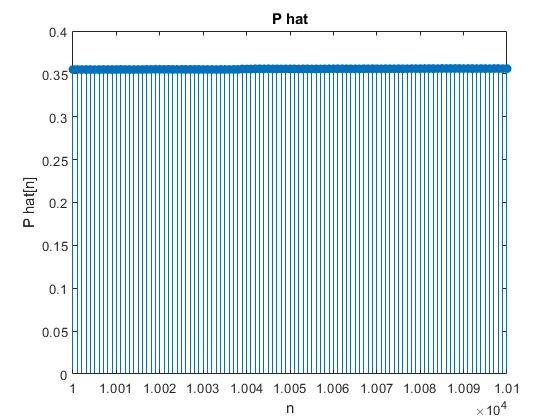
**(1)**

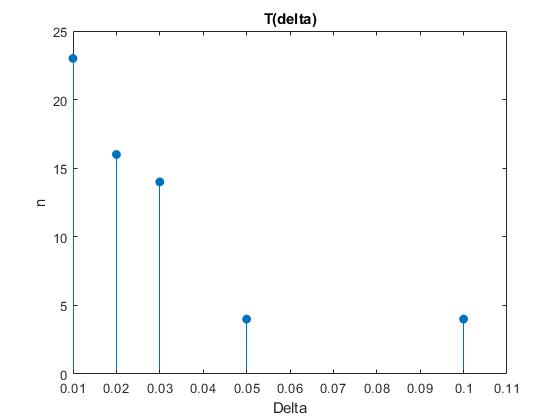


**(2)**



**(3)**



**(4)**

**(5)**

In my program, I define an array in the domain of 0~100001, then I move the origin from 0 to 50001, so, I can generate S in the domain of -50000~0~50000. The reason is that I think 50000 is large enough to represent infinity.

**Source code:**

%% generate i and s

b0 = 0.6;

b1 = 0.4;

i = wgn(100001, 1, 0);

var\_i = 1;

s = zeros(100001, 1);

s(1, 1) = 0;

for k = 2:100001

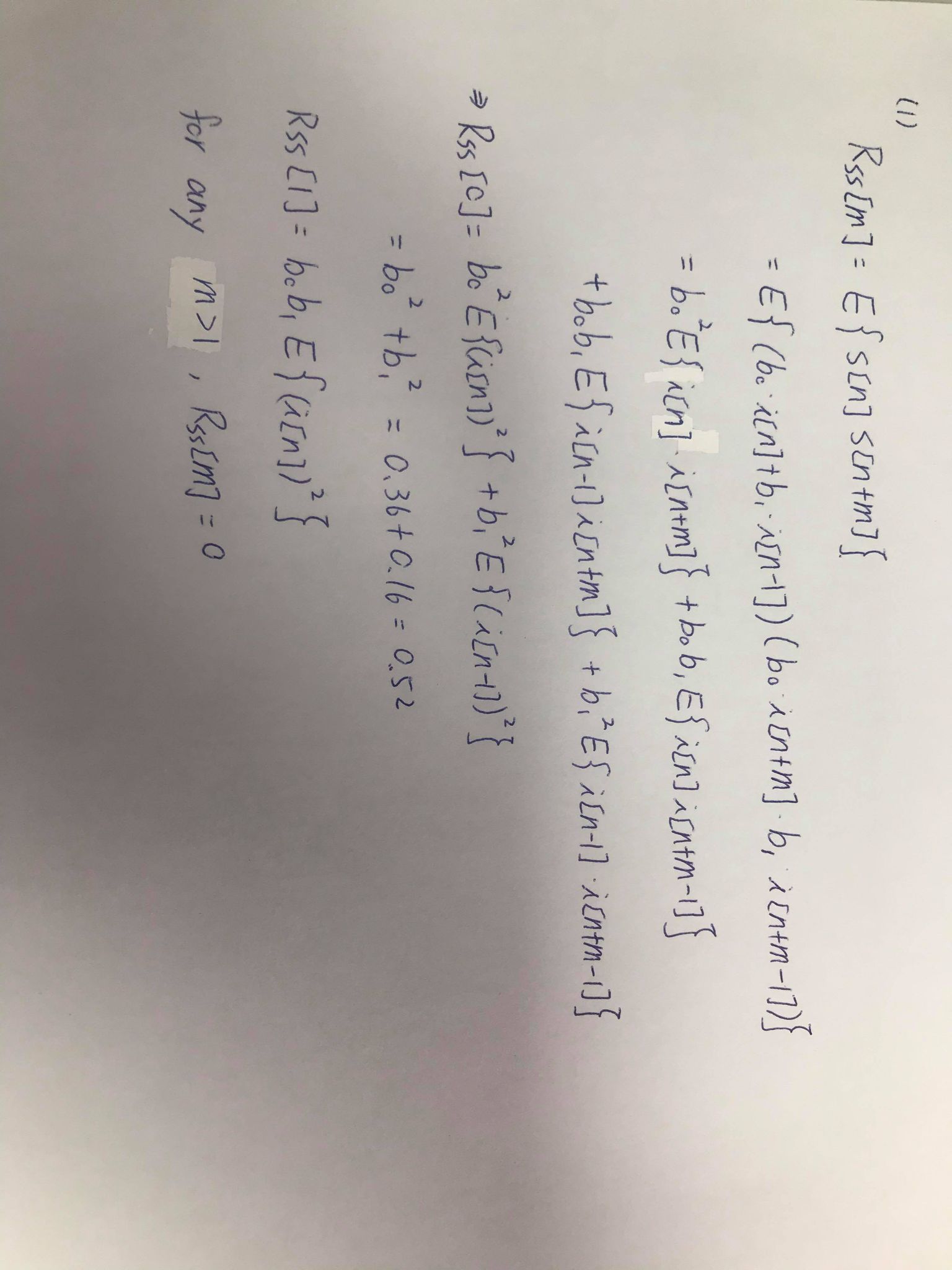
s(k, 1) = b0\*i(k, 1) + b1\*i(k-1, 1);

end

% move origin to 50001

origin = 50001;

**For question 1**, I generate Rs[m] by MATLAB program based on following statement:



**Source code:**

%% compute Rs[m]

Rs = [];

n = 100;

for m = 0:10

if m == 0

Rs(m+1) = b0\*b0\*var\_i+b1\*b1\*var\_i;

elseif (m == 1)

Rs(m+1) = b0\*b1\*var\_i;

else

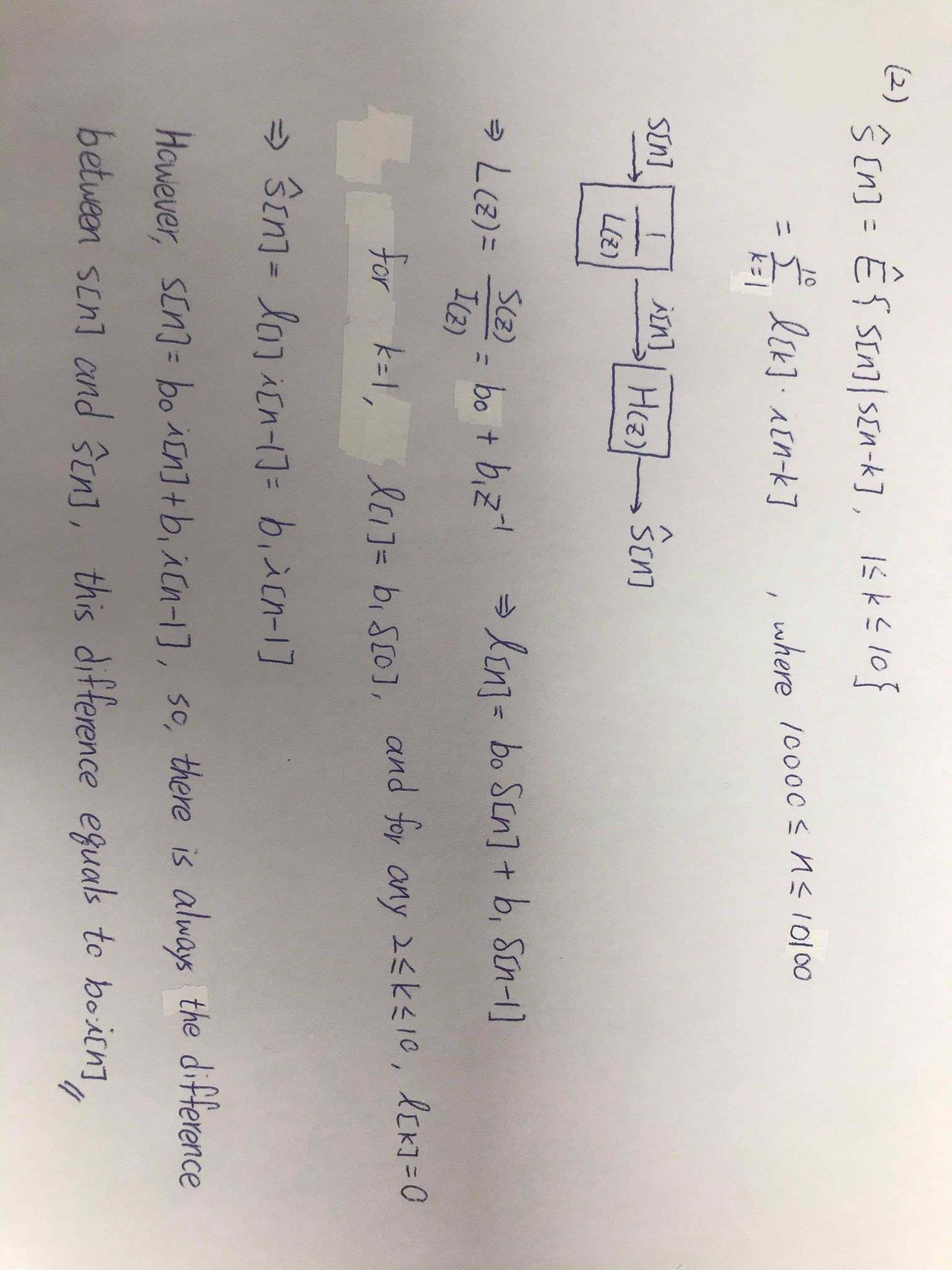
Rs(m+1) = 0;

end

end

figure, stem(0:1:10,Rs, 'filled'), title('Rs'), xlabel('m'), ylabel('Rs[m]');

**For question 2**, I generate by MATLAB program based on following statement:



**Source code:**

%% compute estimation of s[n]

estimate\_s = zeros(100001, 1);

l = zeros(100001, 1);

l(1, 1) = b1;

for n = 0:50000

for k = 1:10

estimate\_s(n + origin, 1) = estimate\_s(n + origin, 1) + l(k, 1)\*i(n + origin - 1);

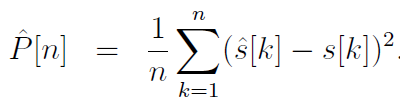
end

end

figure, stem(10000:1:10100, [estimate\_s((origin+10000:origin+10100), 1), s((origin+10000:origin+10100), 1)], 'filled'), legend('Prediction of S', 'S');

title('Prediction of S'), xlabel('n');

**For question 3**, based on the following equation:



By using **S** and defined in question 1, we can compute in the domain of 10000~10100.

**Source code:**

%% compute P

P = zeros(100001, 1);

for n = 1:50000

for k = 1:n

P(origin + n, 1) = P(origin + n, 1) + (estimate\_s(origin + k, 1) - s(origin + k, 1))^2;

end

P(origin + n, 1) = P(origin + n, 1)/n;

end

figure, stem(10000:1:10100, P((origin+10000:origin+10100), 1), 'filled'), title('P hat'), xlabel('n'), ylabel('P hat[n]');

**For question 4**, based on the following definition:

Combining with we generate in question 4, we can get the minimum n to satisfy this definition of T.

**Source code:**

%% compute T

T = zeros(5, 1);

delta = [0.01 0.02 0.03 0.05 0.1];

for i = 1:5

for n = 2:50000

if(abs(P(origin+n, 1)-P(origin+n-1, 1)) <= delta(i)\*P(origin+n-1, 1))

T(i, 1) = n;

break;

end

end

end

figure, stem(delta, T, 'filled'), title('T(delta)'), xlabel('Delta'), ylabel('n');