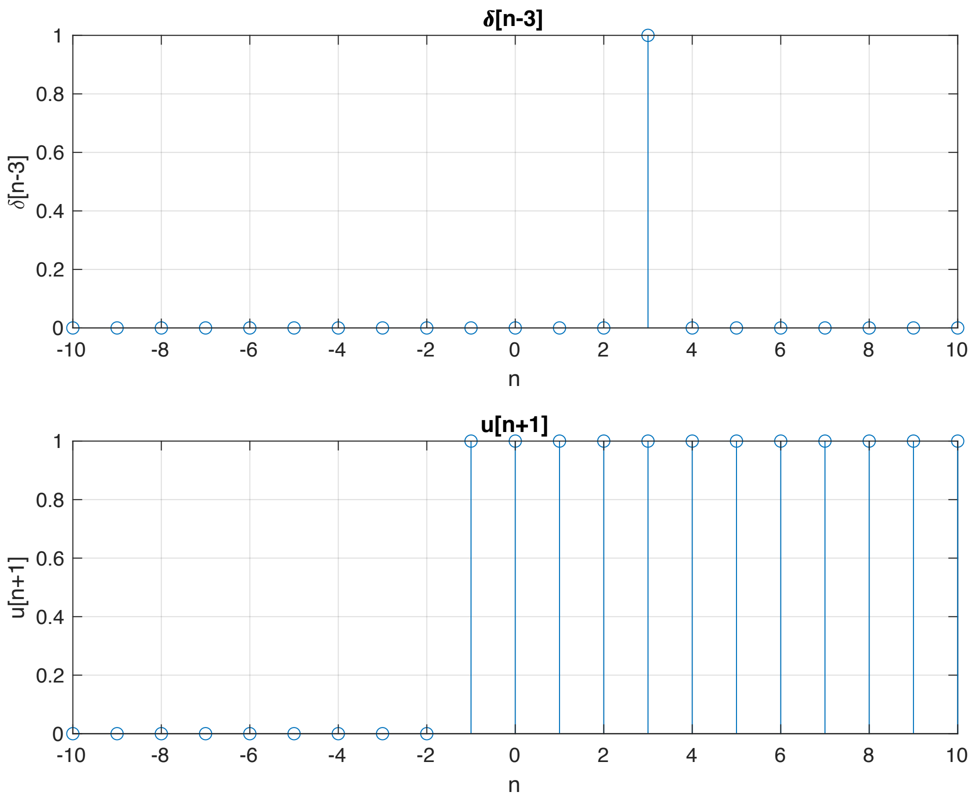
Time-Domain Analysis of Discrete-Time Systems – Part 1

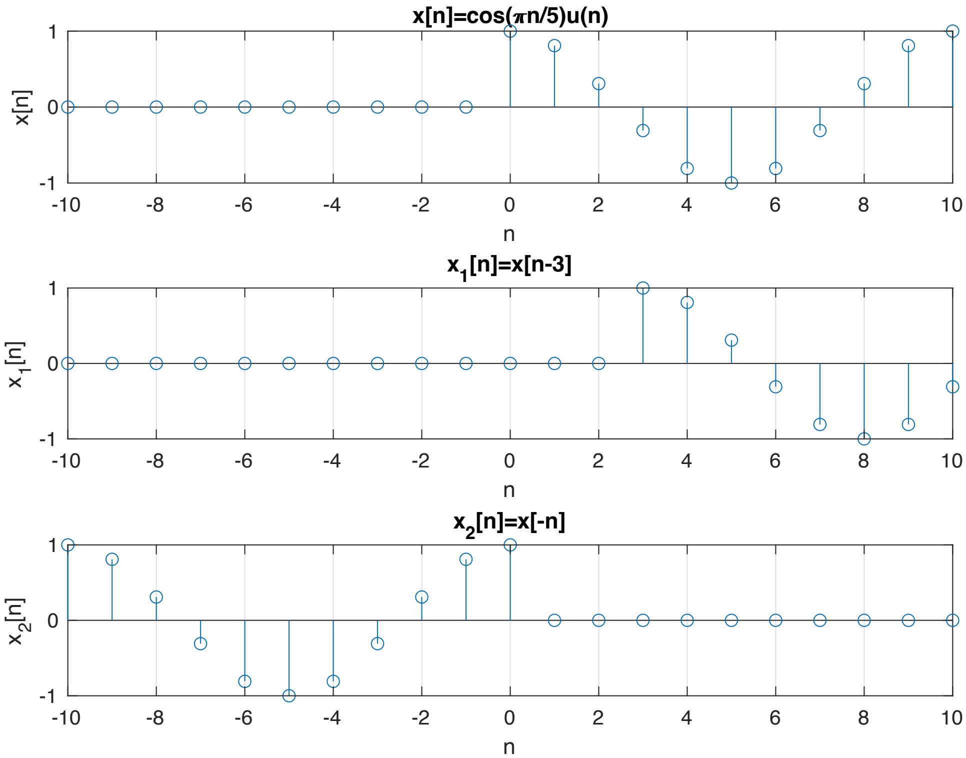
Table of Contents

A. Signal transformation

```
% A.1

n = -10:10; % Range for ploting

% A.1.I
impulse = @(n) (n == 0) * 1.0 .* (mod(n, 1) == 0);
a = impulse(n-3);

% A.1.II
u = @(n) (n >= 0) * 1.0 .* (mod(n,1)==0);
b = u(n+1);

% A.1.III
x = @(n) u(n) .* cos((n .* pi) / 5);
c = x(n);

% A.1.IV
x1 = @(n) x(n-3);
d = x1(n);

% A.1.V
x2 = @(n) x(-n);
e = x2(n);

% Plotting

figure;
subplot(2,1,1);
stem(n,a);
grid;
title('\delta[n-3]');
xlabel('n');
ylabel('\delta[n-3]');

subplot(2,1,2);
stem(n,b);
grid;
title('u[n+1]');
xlabel('n');
ylabel('u[n+1]');
```

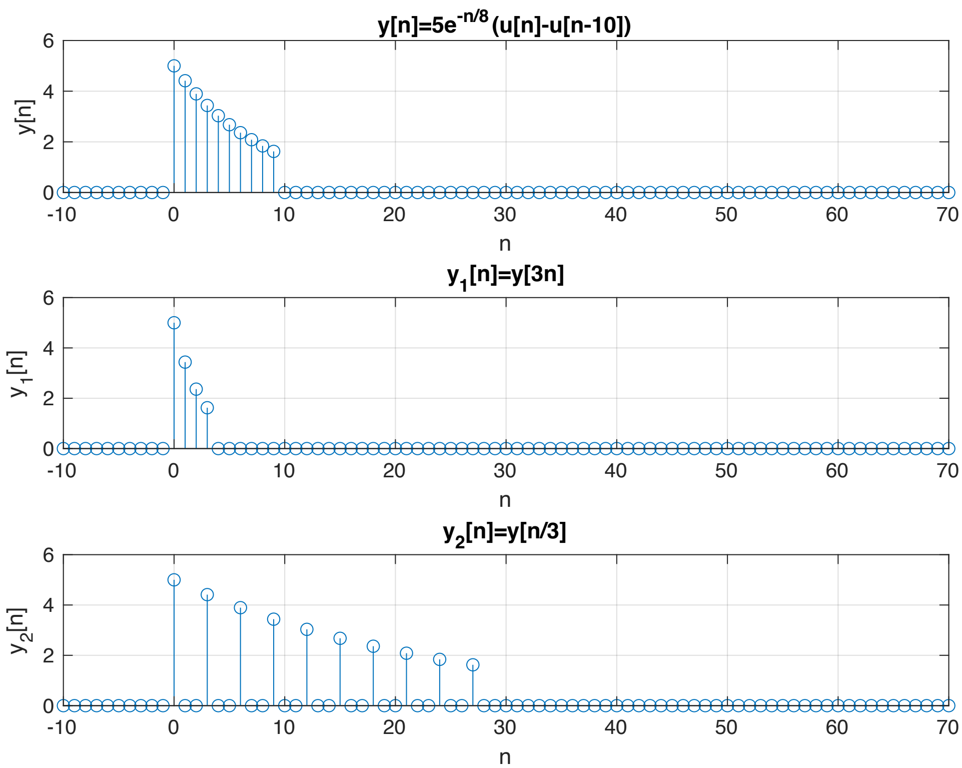



```
figure;
subplot(3,1,1);
stem(n,c);
grid;
title('x[n]=cos(\pin/5)u(n)');
xlabel('n');
ylabel('x[n]');

subplot(3,1,2);
stem(n,d);
grid;
title('x_1[n]=x[n-3]');
xlabel('n');
ylabel('x_1[n]');

subplot(3,1,3);
stem(n,e);
grid;
title('x_2[n]=x[-n]');
xlabel('n');
ylabel('x_2[n]');
```



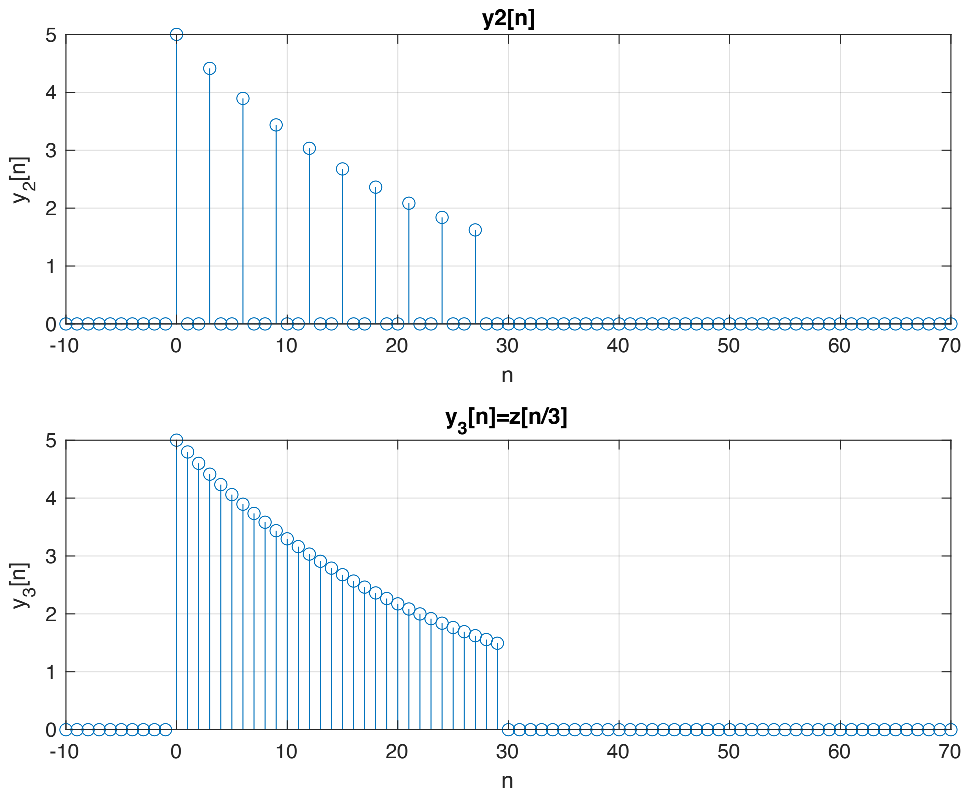

```
% x1[n] is being time shifted to the right by 3.
% x2[n] is being time reversed.

% A.2

% A.2.I
u = @(n) (n >= 0) * 1.0 .* (mod(n,1)==0);
y = @(n) 5*exp(-n/8).*(u(n)-u(n-10));
a = y(n);

% A.2.II
y1 = @(n) y(3*n);
b = y1(n);

% A.2.III
y2 = @(n) y(n/3);
c = y2(n);

n = -10:70; % Range for ploting

figure;
subplot(3,1,1);
stem(n,a);
grid;
title('y[n]=5e^{-n/8}(u[n]-u[n-10])');
xlabel('n');
ylabel('y[n]');

subplot(3,1,2);
stem(n,b);
```

```
grid;
title('y_1[n]=y[3n]');
xlabel('n');
ylabel('y_1[n]');

subplot(3,1,3);
stem(n,c);
grid;
title('y_2[n]=y[n/3]');
xlabel('n');
ylabel('y_2[n]');
```




```
% A.3.I

u = @(n) (n >= 0) * 1.0 .* (mod(n,1)==0);
y = @(n) 5*exp(-n/8).*(u(n)-u(n-10));
y2 = y(n/3);

u1 = @(n) (n >= 0) * 1.0;
z = @(n) 5*exp(-n/8).*(u1(n)-u1(n-10));
y3 = @(n) z(n/3);
n = -10:70;

figure;
subplot(2,1,1);
stem(n,y2);
grid;
title("y2[n]");
xlabel('n');
ylabel('y_2[n]');
```

```
subplot(2,1,2);
stem(n,y3(n));
grid;
title("y_3[n]=z[n/3]");
xlabel('n');
ylabel('y_3[n]');
```

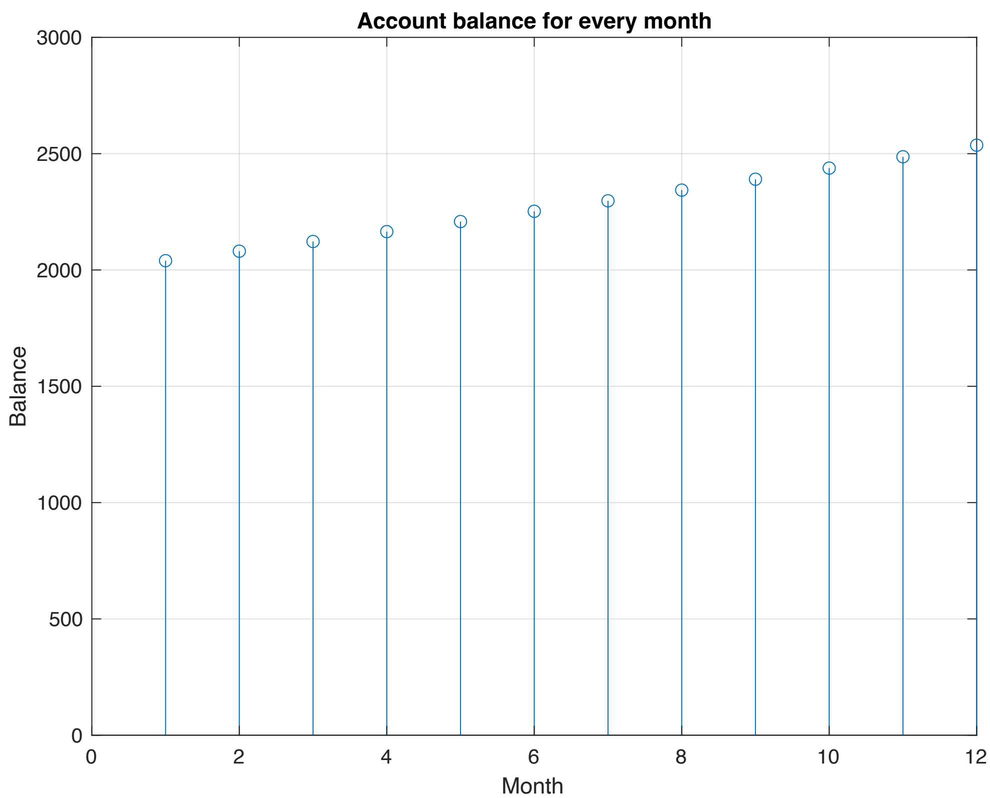



```
% A.3.II

% y3[n] has more data values than y2[n] because of the fact that the signal
% transformation was applied to the continuous signal first, allowing the
% sampling to sample values that now exist in discrete integer values, which
% previously didn't before stretching the continuous function.
```

B. Recursive solution of different equation

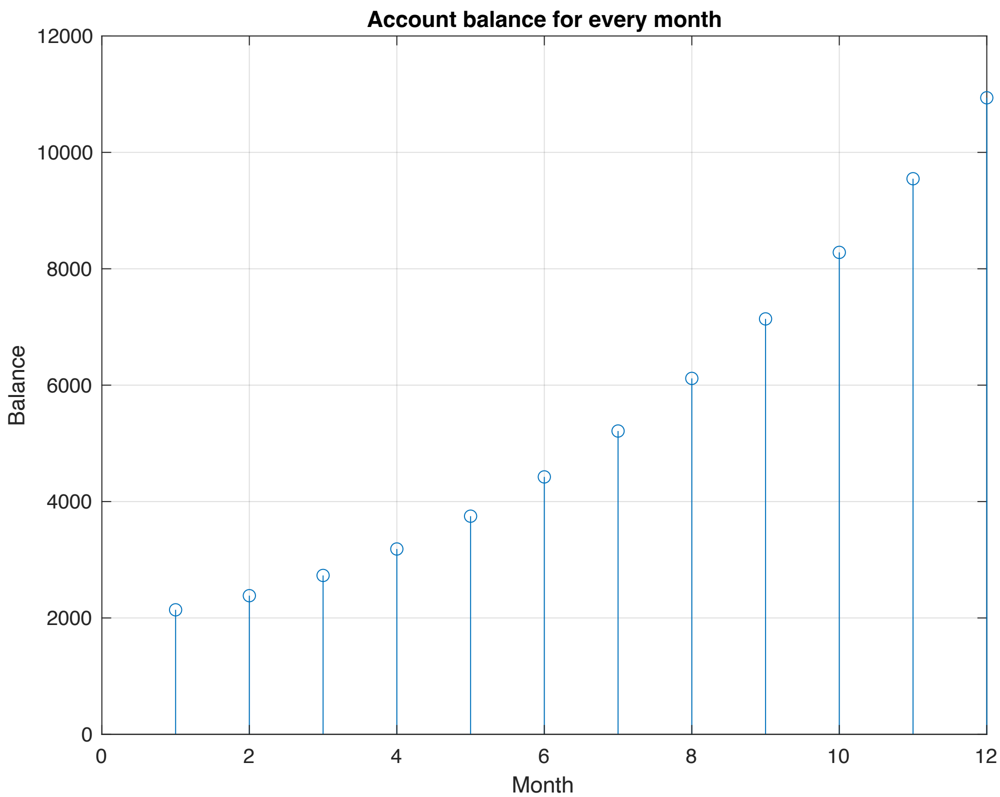
```
% B.1

% The equation relating the output y[n] (the balance) to the input x[n]
% (the deposit) is
% y[n] = y[n - 1] + 0.02 * y[n - 1] + x[n]
```

```
% B.2

y = zeros(1, 12);
y(1) = 1.02 * 2000;

for i = 2:12
    y(i) = y(i - 1) + 0.02 * y(i - 1);
end

figure;
stem(y);
grid;
title('Account balance for every month');
xlabel('Month');
ylabel('Balance');
```




```
% B.3

y = zeros(1, 12);
y(1) = 1.02 * 2000 + 100 * 1;

for i = 2:12
    y(i) = y(i - 1) + 0.02 * y(i - 1) + 100 * i;
```

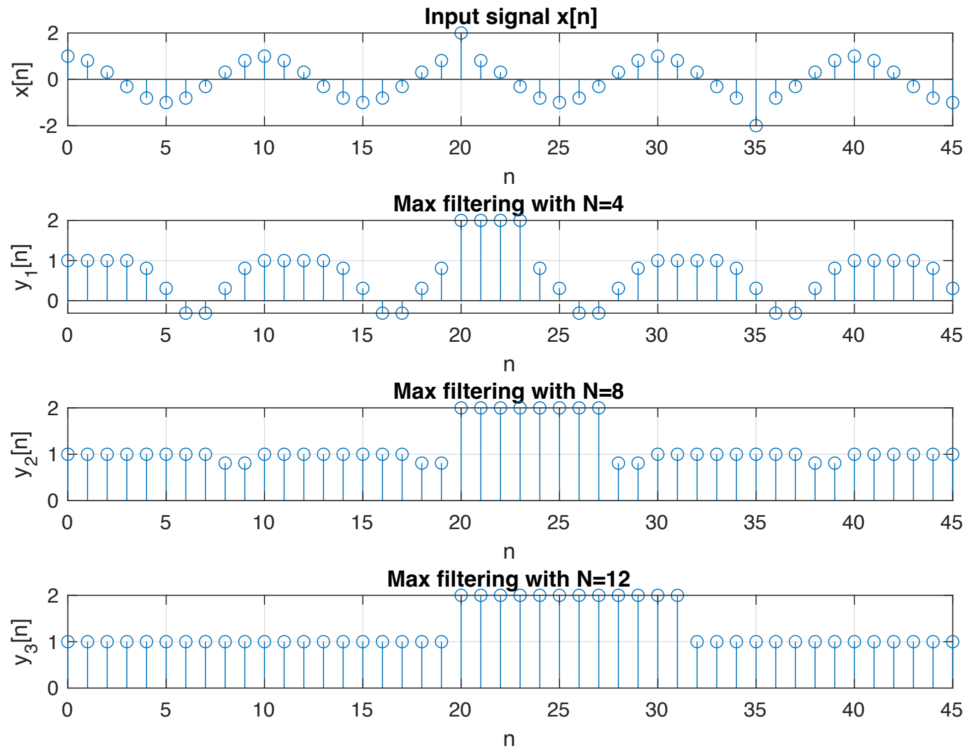
```
end

figure;
stem(y);
grid;
title('Account balance for every month');
xlabel('Month');
ylabel('Balance');
```



C. Design a filter: N-point maximum filter

```
% C.1

function y = maxFilter(x,N)
x1 = [zeros(1,N-1) x];          % zero padding
y1 = zeros(1,length(x1));
for count = N:length(x1)
    y1(count) = max(x1(count - (N-1):count));
end
y = y1(N:end);
end
% C.2

% Length of input
n = 0:45;

% Impulse function
```

```
impulse = @(n) (n == 0) * 1.0 .* (mod(n, 1) == 0);

% Input signal
x = @(n) (cos(pi*n/5)+impulse(n-20)-impulse(n-35));

% Outputs
y1 = maxFilter(x(n),4);
y2 = maxFilter(x(n),8);
y3 = maxFilter(x(n),12);

% Plotting

figure;
subplot(4,1,1);
stem(n,x(n));
grid;
title('Input signal x[n]');
xlabel('n');
ylabel('x[n]');

subplot(4,1,2);
stem(n,y1);
grid;
title('Max filtering with N=4');
xlabel('n');
ylabel('y_1[n]');

subplot(4,1,3);
stem(n,y2);
grid;
xlabel('n');
title('Max filtering with N=8');
ylabel('y_2[n]')

subplot(4,1,4);
stem(n,y3);
grid;
title('Max filtering with N=12');
xlabel('n');
ylabel('y_3[n]');
```




```
% As N approaches infinity, the output signal approaches a unit
% step function multiplied by the max value of the input signal x[n].
```

D. Energy and power of a discrete signal

```
% D.1

function [power,energy] = Part_D1(x)
    power = (1 / length(x)) .* sum(abs(x .^ 2));
    energy = sum(abs(x .^ 2));
end

% D.2

[power, energy] = Part_D1([-9 -6 -3 0 3 6 9])

power =

    36


energy =

   252
```