**Ralph Mouawad – ID 202204667.**

**ENMG 616 – Advanced Optimization Techniques & Algorithms.**

**Dr. Maher Nouiehed.**

## *Assignment 5 – MATLAB codes & interpretation.*

## Part 1.2 – Min-Max Problem - MATLAB codes:

Question 1:

```
n = 10;
B = randn(n,n);
A = B*B';
```

Questions 2 & 3:

Extra Gradient Method:

```
x = ones(n,1);
y = ones(n,1);
x_1_2 = ones(n,1);
y_1_2 = ones(n,1);
x1 = ones(n,1);
y1 = ones(n,1); % Initialize all variables
a = 0.01; % step size
error = zeros(1,15000); % Initialize an error list
for i = 1:15000
    x1 = x;
    y1 =y;
    x_1_2 = x1 - a*A*y1;
    y_1_2 = y1 + a*A'*x1; % I changed the order of
multiplication bcz x and y were becoming % 10x10 instead of
10x1 and error was nan
    x = x1 - a*A*y_1_2;
    y = y1 + a*A'*x_1_2;
```

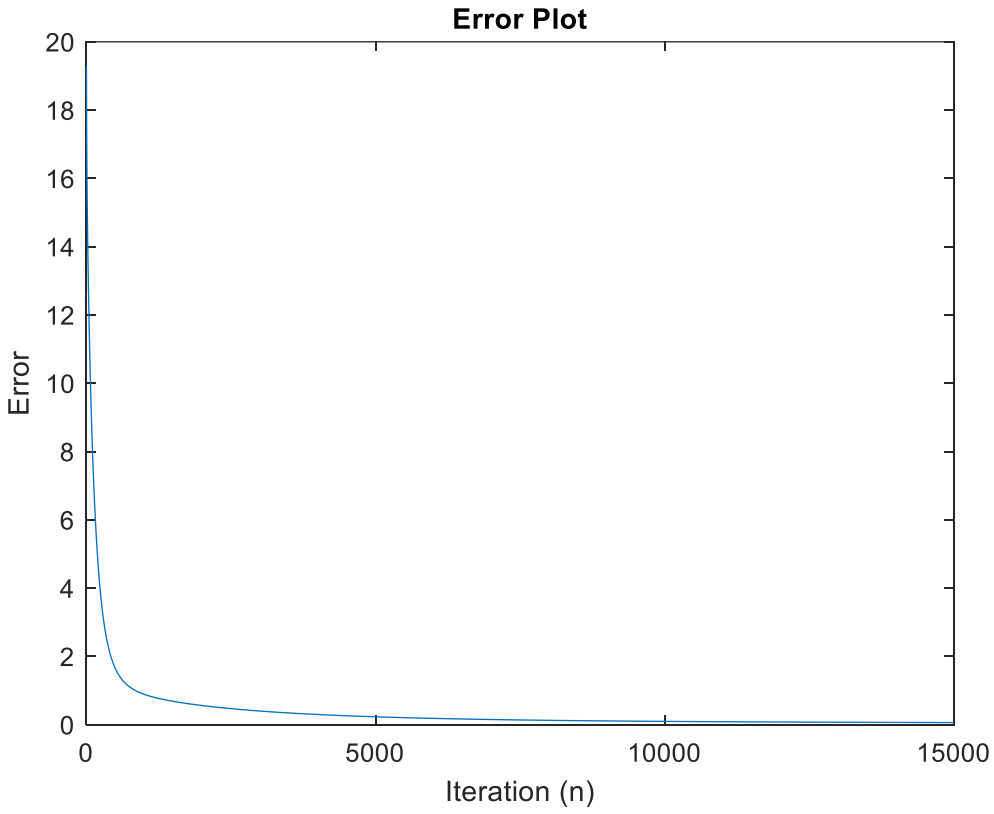
```
    e = norm(x)^2 + norm(y)^2;
    error(i) = e;
end

plot(1:15000, error);

% Add labels and title
xlabel('Iteration (n)');
ylabel('Error');
title('Error Plot');

error;
```

The obtained plot is:

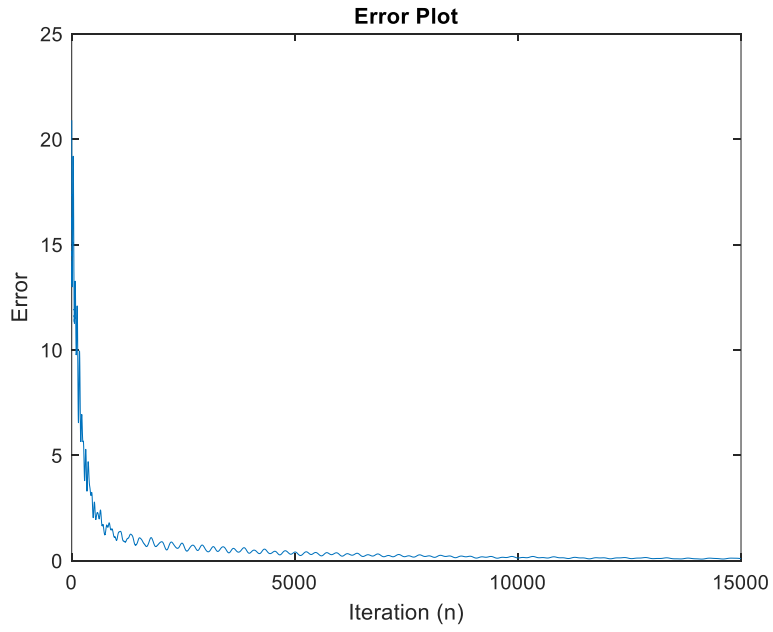


Optimistic Gradient Descent Ascent:

```
x = ones(n,1);
y = ones(n,1);
x1 = ones(n,1);
y1 = ones(n,1);
x2 = ones(n,1);
y2 = ones(n,1);
a = 0.01;
error = zeros(1,15000);
for i = 1:15000
    x1 = x;
    y1 = y;
    x = x1 - 2*a*A*y1 + a*A*y2;
    y = y2 + 2*a*A'*x1 - a*A'*x2;
    x2 = x1;
    y2 = y1;
    error(i) = norm(x)^2 + norm(y)^2;
end
plot(1:15000, error);

% Add labels and title
xlabel('Iteration (n)');
ylabel('Error');
title('Error Plot');
```

The plot is:



We can see more fluctuations when the error was being minimized.

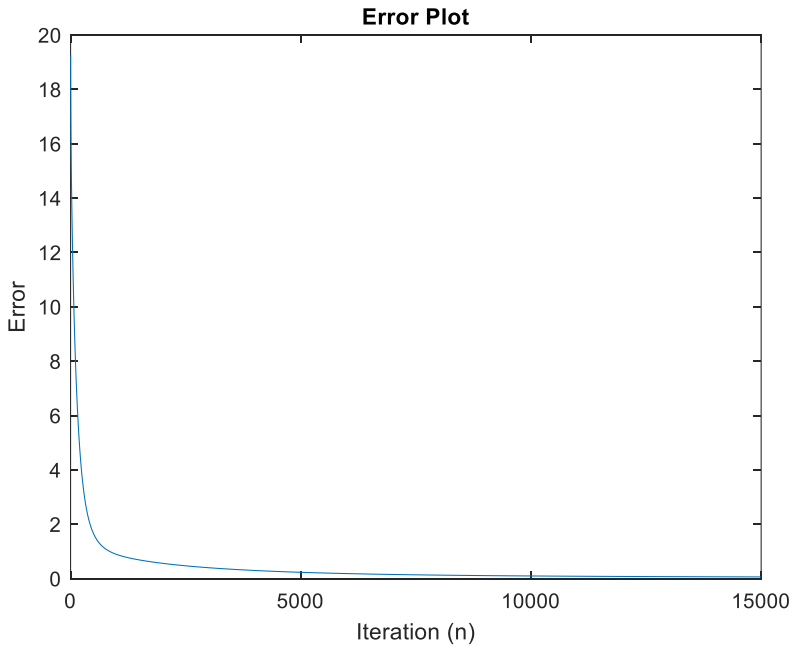
Proximal Point method:

```
x = ones(n,1);
y = ones(n,1);
a = 0.01;
error = zeros(1,15000);
for i = 1:15000
    x1 = x;
    y1 = y;
    x = inv(eye(n) + a^2*A*A')*(x1 - a*A*y1); % I changed
the positions bcz of the error in matrix multiplication
    y = y1 + a*A*x;
    error(i) = norm(x)^2 + norm(y)^2;
end

plot(1:15000, error);

% Add labels and title
xlabel('Iteration (n)');
ylabel('Error');
title('Error Plot');
```

The plot is:

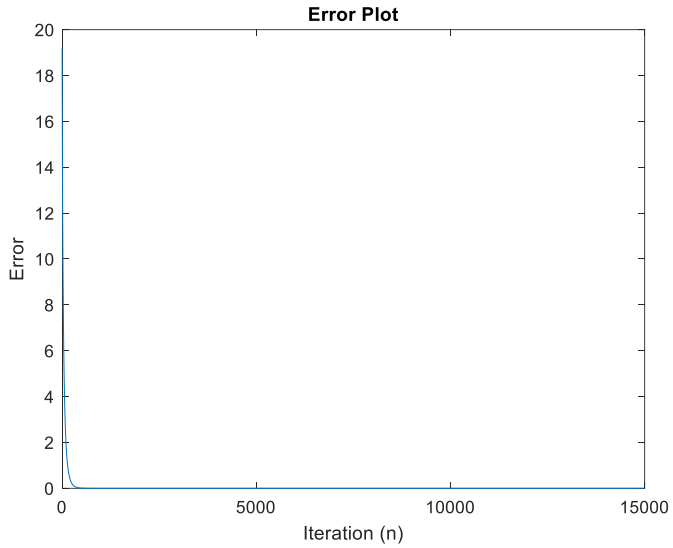


Question 4:

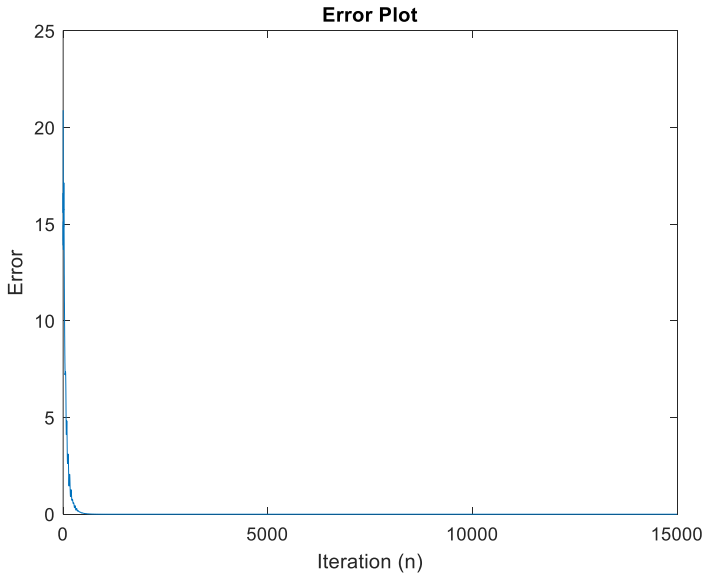
```
n = 10;
B = randn(n,n);
A = B*B';
A = A + 10*eye(n);
```

We obtained the following new plots:

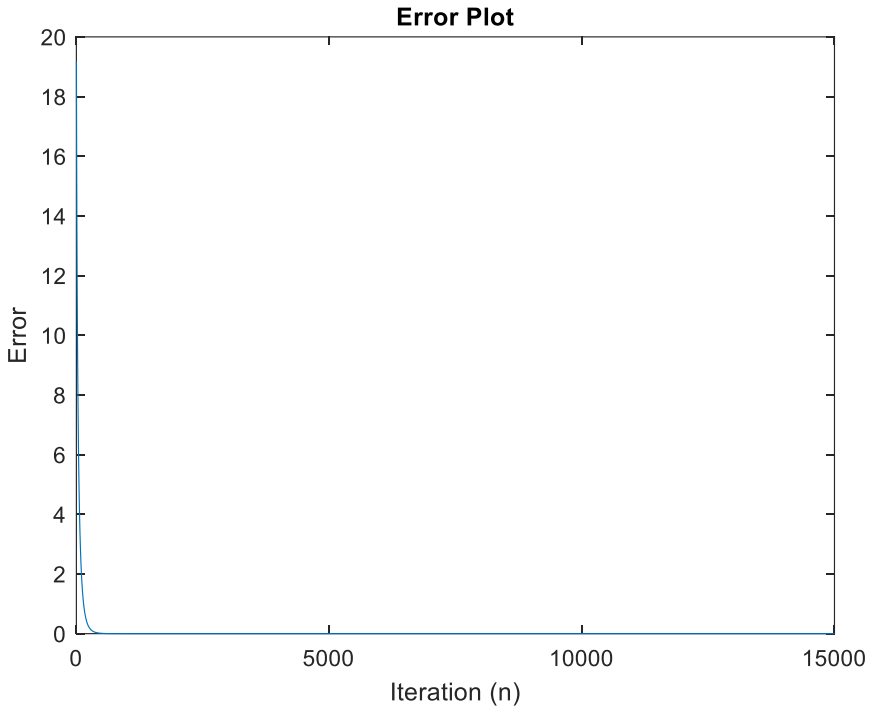
EG:




OGDA:



PP:




We can see that all 3 algorithms, after this change in the matrix **A**, converged faster to an error of zero.