ECE 58000 FunWork5

Shaunak Mukherjee

Spring 2024

Problem 1 Minimize the Rastrigin's function from Problem 9.5 on page 142 in the textbook
Given function, $f(x_1, x_2) = 20 + \left(\frac{x_1}{10}\right)^2 + \left(\frac{x_2}{10}\right)^2 - 10\left(\cos\left(\frac{2\pi x_1}{10}\right) + \cos\left(\frac{2\pi x_2}{10}\right)\right)$

Over the region $\begin{bmatrix} -11, & 11 \end{bmatrix} \times \begin{bmatrix} -11, & 11 \end{bmatrix}$ using the canonical genetic algorithm. Produce plots of the best, average, and the worst objective function values in the population for every generation.

Solution Below I show the matlab code for CGA implementation on the Rastrigin's function.

```matlab
% Clear workspace and command window
close all
clear
clc

% Given  Rastrigins  objective function
f = @(x1, x2) 20 + (x1/10)^2 + (x2/10)^2 - 10 * (cos(2*pi*x1/10) + cos(2*pi*x2
    /10));

% Genetic Algorithm Parameters
pop_size = 100;     % Population size
num_genes = 10;     % Number of bits for each variable
num_gens = 50;     % Number of generations
p_crossover = 0.7;  % Crossover probability
p_mutation = 0.01;  % Mutation probability
elitism_ratio = 0.1;% Elitism ratio (percentage of population to keep)

% Define search space
x_min = -11; % Minimum value of x1 and x2
x_max = 11;  % Maximum value of x1 and x2

% Initialize population
pop = initialize_population(pop_size, num_genes);

% Initialize arrays to store best, average, and worst objective function values
    per generation
best_obj_per_gen = zeros(num_gens, 1);
average_obj_per_gen = zeros(num_gens, 1);
worst_obj_per_gen = zeros(num_gens, 1);

% Main loop for generations
for generation = 1:num_gens
    % Evaluate objective function value of each individual
```

```
    obj_values = zeros(pop_size, 1);
    for i = 1:pop_size
        x1 = bin2real(pop(i, 1:num_genes), x_min, x_max); % Decode x1
        x2 = bin2real(pop(i, num_genes+1:end), x_min, x_max); % Decode x2
        obj_values(i) = f(x1, x2);
    end

    % Record best, average, and worst objective function values of this generation
    best_obj_per_gen(generation) = min(obj_values);
    average_obj_per_gen(generation) = mean(obj_values);
    worst_obj_per_gen(generation) = max(obj_values);

    % Create surface plot for current generation
    x1_vals = linspace(x_min, x_max, 100);
    x2_vals = linspace(x_min, x_max, 100);
    [X1, X2] = meshgrid(x1_vals, x2_vals);
    Z = f(X1, X2);

    contourf(X1, X2, Z, 'LevelStep', 5, 'LineWidth', 0.1, 'LineStyle', '--', 'ShowText','On', 'LabelFormat','%0.1f');
    hold on;
    scatter3(x1, x2, obj_values, 'r', 'filled');
    hold off;
    title(['Generation ', num2str(generation)]);
    xlabel('x1');
    ylabel('x2');
    zlabel('Objective Function Value');
    drawnow;

    % Selection: Roulette Wheel Selection
    probabilities = 1 ./ (obj_values - min(obj_values) + 1); % Add 1 to avoid division by zero
    probabilities = probabilities / sum(probabilities); % Normalize probabilities
    selected_indices = randsample(1:pop_size, pop_size, true, probabilities);
    selected_pop = pop(selected_indices, :);

    % Elitism: Keep the best individuals without changes
    num_elites = round(pop_size * elitism_ratio);
    [~, elite_indices] = min(obj_values);
    elite_pop = pop(elite_indices, :);
    selected_pop(1:num_elites, :) = repmat(elite_pop, num_elites, 1);

    % Crossover: Single-point crossover
    crossover_points = rand(pop_size/2, 1);
    crossover_indices = find(crossover_points < p_crossover);
    num_crossovers = length(crossover_indices);

    for i = 1:2:num_crossovers*2
        idx1 = crossover_indices((i + 1) / 2);
        idx2 = crossover_indices((i + 1) / 2);
        crossover_point = randi([1, num_genes*2 - 1]);
        temp = selected_pop(idx1, crossover_point+1:end);
        selected_pop(idx1, crossover_point+1:end) = selected_pop(idx2, crossover_point+1:end);
        selected_pop(idx2, crossover_point+1:end) = temp;
    end
```

```

    % Mutation: Bit-flip mutation
    for i = 1:pop_size
        for j = 1:num_genes*2
            if rand() < p_mutation
                selected_pop(i, j) = ~selected_pop(i, j); % Flip the bit
            end
        end
    end

    % Replace population with offspring
    pop = selected_pop;
end

% Plot generation vs best, average, and worst objective function values as a connected scatter plot
figure;
scatter(1:num_gens, best_obj_per_gen, 'b', 'filled');
hold on;
scatter(1:num_gens, average_obj_per_gen, 'g', 'filled');
scatter(1:num_gens, worst_obj_per_gen, 'r', 'filled');
plot(1:num_gens, best_obj_per_gen, 'b-', 'LineWidth', 1.5);
plot(1:num_gens, average_obj_per_gen, 'g-', 'LineWidth', 1.5);
plot(1:num_gens, worst_obj_per_gen, 'r-', 'LineWidth', 1.5);
xlabel('Generation');
ylabel('Objective Function Value');
title('Generation vs Objective Function Value');
legend('Best', 'Average', 'Worst');
grid on;

% Function to initialize population with binary encoding
function pop = initialize_population(pop_size, num_genes)
    pop = randi([0, 1], pop_size, num_genes*2);
end

% Function to convert binary string to real value
function x_dec = bin2real(x_bin, x_min, x_max)
    x_dec = bin2dec(num2str(x_bin)) / (2^length(x_bin) - 1) * (x_max - x_min) + x_min;
end
```

Here is CGA minimization of the function visualization with 100 population size with 50 generations, 0.7 cross over rate, 0.01 mutation rate and 0.1 elitism ratio. Minimization lies at $(x_1, x_2) = (-0.0108, 0.0108)$

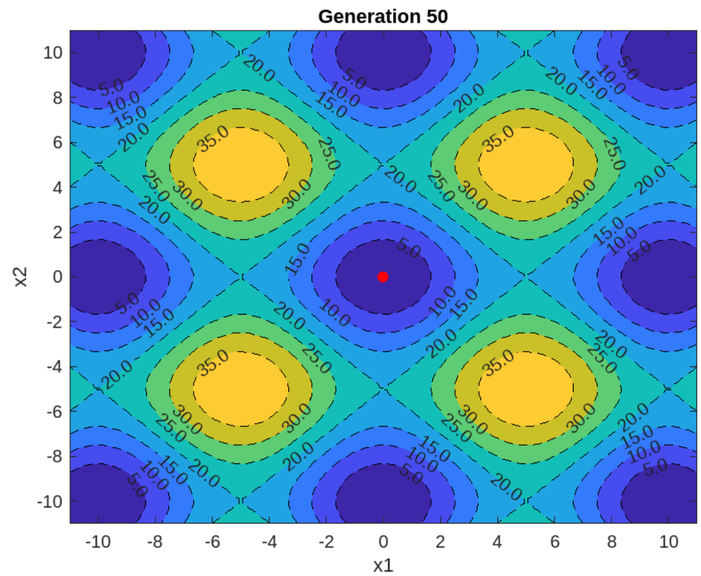

Figure 1: CGA: Minimization contour

and finally, plot of best, average and worst objective function value

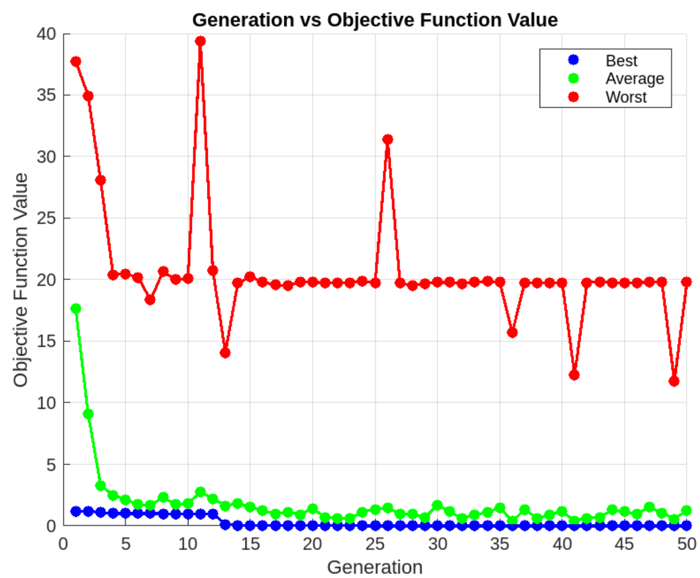

Figure 2: CGA: Best, ave and worst Obj func val

——**Answer**——

Problem 2 Minimize the above Rastrigin's function using a real-number genetic algorithm. Produce plots of the best, average, and the worst objective function values in the population for every generation.

Solution Below I show the matlab code for CGA implementation on the Rastrigin's function.

```
% Clear workspace and command window
close all
clear
clc

% Given function (Rastrigin's function)
f = @(x1, x2) 20 + (x1/10)^2 + (x2/10)^2 - 10 * (cos(2*pi*x1/10) + cos(2*pi*x2/10));

% Genetic Algorithm Parameters
pop_size = 100;         % Population size
num_genes = 2;          % Number of genes (variables)
num_gens = 100;          % Number of generations
p_crossover = 0.7 ;      % Crossover probability
p_mutation = 0.01;      % Mutation probability
elitism_ratio = 0.1;    % Elitism ratio (percentage of population to keep)

% Define search space
x_min = -11;  % Minimum value of x1 and x2 (for Rastrigin's function)
x_max = 11;   % Maximum value of x1 and x2 (for Rastrigin's function)

% Initialize population
pop = initialize_population(pop_size, num_genes, x_min, x_max);

% Initialize arrays to store best, average, and worst objective function values per generation
best_obj_per_gen = zeros(num_gens, 1);
average_obj_per_gen = zeros(num_gens, 1);
worst_obj_per_gen = zeros(num_gens, 1);

% Main loop for generations
for generation = 1:num_gens
    % Evaluate objective function value of each individual
    obj_values = zeros(pop_size, 1);
    for i = 1:pop_size
        x1 = pop(i, 1); % Extract x1
        x2 = pop(i, 2); % Extract x2
        obj_values(i) = f(x1, x2);
    end

    % Record best, average, and worst objective function values of this generation
    best_obj_per_gen(generation) = min(obj_values);
    average_obj_per_gen(generation) = mean(obj_values);
    worst_obj_per_gen(generation) = max(obj_values);

    % Create surface plot for current generation
    x1_vals = linspace(x_min, x_max, 100);
    x2_vals = linspace(x_min, x_max, 100);
    [X1, X2] = meshgrid(x1_vals, x2_vals);
```

```matlab
    Z = f(X1, X2);

    contourf(X1, X2, Z, 'LevelStep', 5, 'LineWidth', 0.1, 'LineStyle', '--', 'ShowText','On', 'LabelFormat','%0.1f');
    hold on;
    scatter3(pop(:, 1), pop(:, 2), obj_values, 'r', 'filled');
    hold off;
    title(['Generation ', num2str(generation)]);
    xlabel('x1');
    ylabel('x2');
    zlabel('Objective Function Value');
    drawnow;

    % Selection: Roulette Wheel Selection
    probabilities = 1 ./ (obj_values - min(obj_values) + 1); % Add 1 to avoid division by zero
    probabilities = probabilities / sum(probabilities); % Normalize probabilities
    selected_indices = randsample(1:pop_size, pop_size, true, probabilities);
    selected_pop = pop(selected_indices, :);

    % Elitism: Keep the best individuals without changes
    num_elites = round(pop_size * elitism_ratio);
    [~, elite_indices] = min(obj_values);
    elite_pop = pop(elite_indices, :);
    selected_pop(1:num_elites, :) = repmat(elite_pop, num_elites, 1);

    % Crossover: Single-point crossover
    crossover_points = rand(pop_size/2, 1);
    crossover_indices = find(crossover_points < p_crossover);
    num_crossovers = length(crossover_indices);

    for i = 1:2:num_crossovers*2
        idx1 = crossover_indices((i + 1) / 2);
        idx2 = crossover_indices((i + 1) / 2);
        crossover_point = randi([1, num_genes]);
        temp = selected_pop(idx1, crossover_point+1:end);
        selected_pop(idx1, crossover_point+1:end) = selected_pop(idx2, crossover_point+1:end);
        selected_pop(idx2, crossover_point+1:end) = temp;
    end

    % Mutation: Uniform mutation
    for i = 1:pop_size
        for j = 1:num_genes
            if rand() < p_mutation
                % Generate a new random value within the search space
                selected_pop(i, j) = rand() * (x_max - x_min) + x_min;
            end
        end
    end

    % Replace population with offspring
    pop = selected_pop;
end

% Plot generation vs best, average, and worst objective function values as a connected scatter plot
```

```
figure;
scatter(1:num_gens, best_obj_per_gen, 'b', 'filled');
hold on;
scatter(1:num_gens, average_obj_per_gen, 'g', 'filled');
scatter(1:num_gens, worst_obj_per_gen, 'r', 'filled');
plot(1:num_gens, best_obj_per_gen, 'b-', 'LineWidth', 1.5);
plot(1:num_gens, average_obj_per_gen, 'g-', 'LineWidth', 1.5);
plot(1:num_gens, worst_obj_per_gen, 'r-', 'LineWidth', 1.5);
xlabel('Generation');
ylabel('Objective Function Value');
title('Generation vs Objective Function Value');
legend('Best', 'Average', 'Worst');
grid on;

% Function to initialize population with random real numbers
function pop = initialize_population(pop_size, num_genes, x_min, x_max)
    pop = rand(pop_size, num_genes) * (x_max - x_min) + x_min;
end
```

Here is CGA minimization of the function visualization with 100 population size with 100 generations, 0.7 cross over rate, 0.01 mutation rate and 0.1 elitism ratio. Minimization lies at $(x_1, x_2) = (0.1497, 0.1299)$

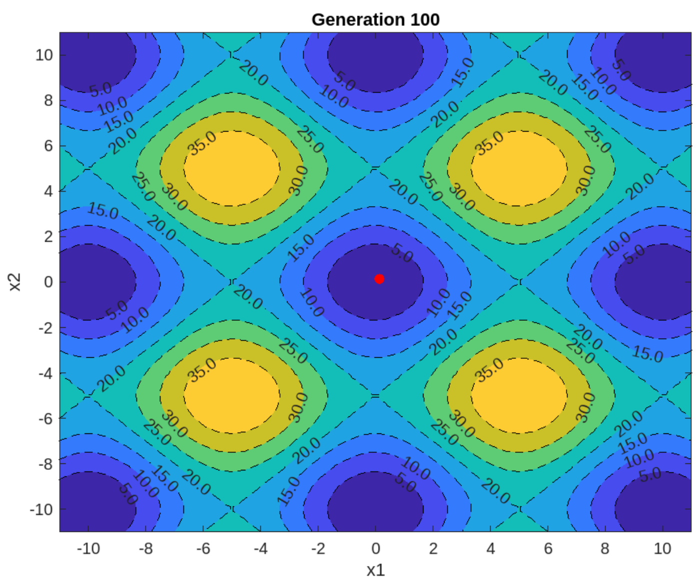

Figure 3: Real number CGA: Minimization contour

and finally, plot of best, average and worst objective function value

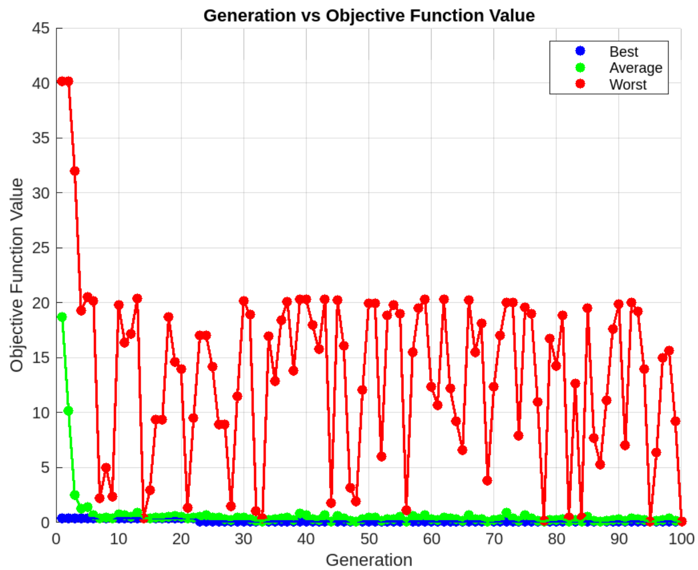

Figure 4: Real number CGA: Best, ave and worst Obj func val

——**Answer**——

Problem 3 Exercise 16.18 a. on page 306 in the textbook. Given a LP problem minimize

$$\begin{aligned} \text{minimize} \quad & -\frac{3}{4}x_4 + 20x_5 - \frac{1}{2}x_6 + 6x_7 \\ \text{subject to} \quad & x_1 + \frac{1}{4}x_4 - 8x_5 - x_6 + 9x_7 = 0 \\ & x_2 + \frac{1}{2}x_4 - 12x_5 - \frac{1}{2}x_6 + 3x_7 = 0 \\ & x_3 + x_6 = 1 \\ & x_1, \ldots, x_7 \geq 0 \end{aligned}$$

Apply the simplex algorithm to the problem using the rule that q is the index corresponding to the most negative r_q. (As usual, if more than one index i minimizes y_{i0}/y_{iq}, let p be the smallest such index.) Start with x_1, x_2, and x_3 as initial basic variables. Notice that cycling occurs.

Solution: Forming the tableu as below which is in canonical form and will be solved using simplex technique.

1	0	0	1/4	-8	-1	9	0
0	1	0	1/2	-12	-1/2	3	0
0	0	1	0	0	1	0	1
0	0	0	-3/4	20	-1/2	6	0

Pivot column should be 4th and the Pivot element (1, 4), we do elementary row operations to obtain, resulting tableau below:

4	0	0	1	-32	-4	36	0
-2	1	0	0	4	3/2	-15	0
0	0	1	0	0	1	0	1
3	0	0	0	-4	-7/2	33	0

Pivoting about (2, 5)th element and performing elementary row operations, resulting tableau below:

-12	8	0	1	0	8	-84	0
-1/2	1/4	0	0	1	3/8	-15/4	0
0	0	1	0	0	1	0	1
1	1	0	0	0	-2	18	0

Pivoting about (1, 6)th element and performing elementary row operations, resulting tableau below:

-3/2	1	0	1/8	0	1	-21/2	0
1/16	-1/8	0	-3/64	1	0	3/16	0
3/2	-1	1	-1/8	0	0	21/2	1
-2	3	0	1/4	0	0	-3	0

Pivoting about (2, 7)th element, and performing elementary row operations we get tableau below:

2	-6	0	-5/2	56	1	0	0
1/3	-2/3	0	-1/4	16/3	0	1	0
-2	6	1	5/2	-56	0	0	1
-1	1	0	-1/2	16	0	0	0

Pivoting about (1, 1)th element, we get below tableau:

1	-3	0	-5/4	28	1/2	0	0
0	1/3	0	1/6	-4	-1/6	1	0
0	0	1	0	0	1	0	1
0	-2	0	-7/4	44	1/2	0	0

Pivoting about (2, 2)th element, we get final tableau:

1	0	0	1/4	-8	-1	9	0
0	1	0	1/2	-12	-1/2	3	0
0	0	1	0	0	1	0	1
0	0	0	-3/4	20	-1/2	6	0

This identical to initial tableau we began with hence we show cycling occurs.
Answer

Problem 4 Repeat Problem 3 using Bland's rule for choosing q and p:

$$q = \min\{i : r_i < 0\};$$

$$p = \min\{j : y_{j0}/y_{jq} = \min\{\frac{y_{i0}}{y_{iq}} : y_{iq} > 0\}$$

.
Initial tableau and pivot about the (1, 4)th element to obtain

4	0	0	1	-32	-4	36	0
-2	1	0	0	4	3/2	-15	0
0	0	1	0	0	1	0	1
3	0	0	0	-4	-7/2	33	0

Pivoting about (2, 5)th element, we get

-12	8	0	1	0	8	-84	0
-1/2	1/4	0	0	1	3/8	-15/4	0
0	0	1	0	0	1	0	1
1	1	0	0	0	-2	18	0

Pivoting about (1, 6)th element, we get

-3/2	1	0	1/8	0	1	-21/2	0
1/16	-1/8	0	-3/64	1	0	3/16	0
3/2	-1	1	-1/8	0	0	21/2	1
-2	3	0	1/4	0	0	-3	0

Pivoting about (2, 1)th element, we get,

0	-2	0	-1	24	1	-6	0
1	-2	0	-3/4	16	0	3	0
0	2	1	1	-24	0	6	1
0	-1	0	-5/4	32	0	3	0

Pivoting about (3, 2)th element, we get,

0	0	1	0	0	1	0	1
1	0	1	1/4	-8	0	9	1
0	1	1/2	1/2	-12	0	3	1/2
0	0	1/2	-3/4	20	0	6	1/2

Pivoting about (3, 2)th element, we get,

0	0	1	0	0	1	0	1
1	-1/2	3/4	0	-2	0	15/2	3/4
0	2	1	1	-24	0	6	1
0	3/2	5/4	0	2	0	21/2	5/4

The reduced cost coefficient are all non-negative. Hence, the optimal solution to the problem is

$$\mathbf{v} = \begin{bmatrix} 3/4 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}^T$$

The corresponding optimal cost is -5/4. **Answer**

Problem 5 Exercise 20.2 c. on page 394 in the textbook. Find the extremizers for the following optimization problem: Maximize x_1x_2 subject to $x_1^2 + 4x_2^2 = 1$

Solution:

The Lagrangian function for this problem is:

$$L(x_1, x_2, \lambda) = x_1x_2 - \lambda(x_1^2 + 4x_2^2 - 1)$$

First order conditions: Taking partial derivatives of L with respect to x_1, x_2, and λ:

$$\frac{\partial L}{\partial x_1} = x_2 - 2\lambda x_1 = 0 \quad (1)$$

$$\frac{\partial L}{\partial x_2} = x_1 - 8\lambda x_2 = 0 \quad (2)$$

$$\frac{\partial L}{\partial \lambda} = x_1^2 + 4x_2^2 - 1 = 0 \quad (3)$$

From equations (1) and (2), we can solve for x_1 and x_2:
From (1): $x_2 = 2\lambda x_1$
Substitute into (2):

$$x_1 - 8\lambda(2\lambda x_1) = 0$$

$$x_1(1 - 16\lambda^2) = 0$$

Since we're looking for non-zero solutions, $1 - 16\lambda^2 = 0$.
Solving for λ, we get:

$$\lambda^2 = \frac{1}{16}$$

$$\lambda = \pm\frac{1}{4}$$

For $\lambda = \frac{1}{4}$:

$$x_2 = 2\left(\frac{1}{4}\right)x_1 = \frac{1}{2}x_1$$

Substitute into the constraint equation (3):

$$x_1^2 + 4\left(\frac{1}{2}x_1\right)^2 = 1$$

$$2x_1^2 = 1$$

$$x_1 = \pm\frac{1}{\sqrt{2}}$$

For $x_1 = \frac{1}{\sqrt{2}}$:

$$x_2 = \frac{1}{2}\left(\frac{1}{\sqrt{2}}\right) = \frac{1}{2\sqrt{2}}$$

For $x_1 = -\frac{1}{\sqrt{2}}$:

$$x_2 = \frac{1}{2}\left(-\frac{1}{\sqrt{2}}\right) = -\frac{1}{2\sqrt{2}}$$

Therefore, the extremizers are $\left(\frac{1}{\sqrt{2}}, \frac{1}{2\sqrt{2}}\right)$ and $\left(-\frac{1}{\sqrt{2}}, -\frac{1}{2\sqrt{2}}\right)$. **Answer**

Problem 6 Exercise 20.9 on page 395 in the textboook. Find all maximizers of the function

$$f(x_1, x_2) = \frac{18x_1^2 - 8x_1x_2 + 12x_2^2}{2x_1^2 + 2x_2^2}$$

Solution: Rewiring $f(x_1, x_2)$ in quadratic form we have, $f(x_1, x_2) = \frac{x^\top Qx}{x^\top Px}$. If a point x is a maximizer of $f(x_1, x_2)$ then any nonzero multiple of this point will also be a maximizer since

$$\frac{(sx)^\top Q(sx)}{(sx)^\top P(tx)} = \frac{s^2 x^\top Qx}{s^2 x^\top Px} = \frac{x^\top Qx}{x^\top Px}.$$

Thus, we need to find a maximizer of $f(x_1, x_2)$. Then, any nonzero multiple of the solution is also a solution. Next, we represent the original problem in an equivalent form,

$$\text{maximize } x^\top Qx = 18x_1^2 - 8x_1x_2 + 12x_2^2$$

$$\text{subject to } x^\top Px = 2x_1^2 + 2x_2^2 = 1.$$

So, to maximize $f(x_1, x_2) = 18x_1^2 - 8x_1x_2 + 12x_2^2$ subject to the equality constraint, $h(x_1, x_2) = x2x_1^2 + 2x_2^2 - 1$. We will use Lagrange's method and form the Lagrangian function and find its gradient and critical points.

$$l(x, \lambda) = f + \lambda h,$$

$$\begin{aligned}
\nabla_x l &= \nabla_x \left(x^\top \begin{bmatrix} 18 & -4 \\ -4 & 12 \end{bmatrix} x + \lambda \left(1 - x^\top \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} x \right) \right) \\
&= 2 \begin{bmatrix} 18 & -4 \\ -4 & 12 \end{bmatrix} x - 2\lambda \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} x \\
&= 0.
\end{aligned}$$

or,

$$\left(\lambda I_2 - \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}^{-1} \begin{bmatrix} 18 & -4 \\ -4 & 12 \end{bmatrix} \right) x = 0.$$

Solving it like eigenvalue-eigenvector problem,

$$\left(\lambda I_2 - \begin{bmatrix} 9 & -2 \\ -2 & 6 \end{bmatrix} \right) x = \begin{bmatrix} \lambda - 9 & 2 \\ 2 & \lambda - 6 \end{bmatrix} x = 0.$$

or,

$$\det \begin{bmatrix} \lambda - 9 & 2 \\ 2 & \lambda - 6 \end{bmatrix} = \lambda^2 - 15\lambda + 50 = (\lambda - 5)(\lambda - 10).$$

The eigenvalues are 5 and 10. Since we want to find maximizer, value of the maximized function is 10, An eigenvector can easily be found by taking any nonzero column of the adjoint matrix of

$$10I_2 - \begin{bmatrix} 9 & -2 \\ -2 & 6 \end{bmatrix}.$$

$$\text{adj} \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} = \begin{bmatrix} 4 & -2 \\ -2 & 1 \end{bmatrix}.$$

Now, any nonzero column of adj(A)(A) can serve as an eigenvector. Let's take, for example, the first column $\begin{bmatrix} -2 \\ 1 \end{bmatrix}$ as an eigenvector. So,

$$\sqrt{0.1} \begin{bmatrix} -2 \\ 1 \end{bmatrix}$$

is a maximizer for the equivalent problem. Any multiple of the above vector is a solution of the original maximization problem. **Answer**

Problem 7 Exercise 21.3 on page 411 in the textbook. Find local minimizers for $x_1^2 + x_2^2$ subject to $x_1^2 + 2x_1x_2 + x_2^2 = 1$, $x_1^2 - x_2 \leq 0$.

Solution: To find the local minimizers for $x_1^2 + x_2^2$ subject to the given constraints using KKT. Following example 21.6 from the book, For all $x \in \mathbb{R}^2$, $Dh(x) = [2x_1 + 2x_2, 2x_1 + 2x_2]$, $Dg(x) = [2x_1, -1]$, $Df(x) = [2x_1, 2x_2]$
KKT Conditions:

$$Df(x) + \lambda Dh(x) + \mu Dg(x)$$

$$= [2\mathrm{x}_1 + 2\lambda x_1 + 2\lambda x_2 + 2\mu x_1,\ 2\mathrm{x}_2 + 2\lambda x_1 + 2\lambda x_2 - \mu] = 0^T \quad (1)$$

$$\lambda(x_1^2 + 2x_1x_2 + x_2^2 - 1) = 0 \quad (2)$$

$$\mu(x_1^2 - x_2) = 0 \quad (3)$$

$$\mu \geq 0 \quad (4)$$

$$x_1^2 + 2x_1x_2 + x_2^2 - 1 = 0 \quad (5)$$

$$x_1^2 - x_2 \leq 0 \quad (6)$$

We need to solve this system of equations to find the local minimizers. We first try $\mu > 0$, which implies $(x_1^2 - x_2) = 0 \quad (7)$ or $x_1^2 = x_2 \quad (8)$ Also note eqn (5) now, is $(x_1 + x_2)^2 = 1$, or $(x_1 + x_2) = \pm 1$.
First case, When $(x_1 + x_2) = 1$, plugging in value of x_2 from eqn (8) we get, $(x_1^2 + x_1 - 1) = 0$ so we have two roots to this quadratic equation. So, the solutions for x_1 are:

$$x_1 = \frac{-1 + \sqrt{5}}{2} = 0.618 \quad (9)$$

or

$$x_1 = \frac{-1 - \sqrt{5}}{2} = -1.618 \quad (10)$$

Second case, when $(x_1 + x_2) = -1$,plugging in value of x_2 from eqn (8) we get, $(x_1^2 + x_1 + 1) = 0$, which will have two imaginary roots and can be neglected $\forall x \in \mathbb{R}^2$. Using solutions 9 and 10 and using it in equation 7 we obtain two solutions again,

$$x_1 = 0.618, \quad x_2 = 0.3814 \quad (11)$$

or

$$x_1 = -1.618, \quad x_2 = 2.618 \quad (12)$$

Plugging in values of x_1 and x_2 into equation 1 we will get
For $x_1 = 0.618$ and $x_2 = 0.3814$:

$$\begin{cases} 2(0.618) + 2\lambda(0.618) + 2\lambda(0.3814) + 2\mu(0.618) & = 0 \\ 2(0.3814) + 2\lambda(0.618) + 2\lambda(0.3814) - \mu & = 0 \end{cases}$$

For $x_1 = -1.618$ and $x_2 = 2.618$:

$$\begin{cases} 2(-1.618) + 2\lambda(-1.618) + 2\lambda(2.618) + 2\mu(-1.618) & = 0 \\ 2(2.618) + 2\lambda(-1.618) + 2\lambda(2.618) - \mu & = 0 \end{cases}$$

For the pair $x_1 = 0.618$ and $x_2 = 0.3814$:

$$\lambda \approx -0.676, \quad \mu \approx -0.582$$

For the pair $x_1 = -1.618$ and $x_2 = 2.618$:

$$\lambda \approx -1.922, \quad \mu \approx -6.766.$$

But above is not a legitimate solution to the KKT condition, because we obtained $\mu < 0$ which contradicts the assumption that $\mu > 0$.
Next, we try, $\mu = 0$, then we have from equation 1 and 2,

$$2x_1 + 2\lambda x_1 + 2\lambda x_2 = 0 \quad (13)$$

$$2x_2 + 2\lambda x_1 + 2\lambda x_2 = 0 \quad (14)$$

(13) - (14) solving these two equation we get $x_1 = x_2$. Plugging in the values in equation (5) we get,

$$(x_1, x_2) = \left(-\frac{1}{2}, -\frac{1}{2}\right)$$

This however is not valid solution since

$$x_1^2 - x_2 \leq 0$$

or

$$1/4 + 1/2 = 3/4 \nleq 0$$

Hence our solution is below and this satisfies the constrait $g(x*) \leq 0$.
The point x* satisying the KKT necessary condition is therefore the candidate for being the minimizer.

$$(x_1, x_2) = \left(\frac{1}{2}, \frac{1}{2}\right)$$

Answer