*Example MATLAB pseudocode:*

C = 0:0.1:10;

for i1 = 1:numel(C)

C1 = C(i1);

% Agent's payoff as a function of C2

J\_A = e\*(log10(C + 1).\*B2.\*(1 - sqrt(1 - (C1 + C)/10)) ...

- kappa \* C.^2 - phi \* (C - C1).^2);

[J\_Amax(i1), idx] = max(J\_A);

C2\_react(i1) = C(idx);

% Leader's payoff at agent's response

J\_S\_react(i1) = a\*(log10(C1 + 1)\*B1\*(1 - sqrt(1 - (C1 + C2\_react(i1))/10)) ...

- kappa \* C1^2);

end

% Find optimal C1 (leader)

[J\_Sopt, idxS] = max(J\_S\_react);

C1max = C(idxS);

C2max = C2\_react(idxS);

*MATLAB Implementation: Stackelberg Leader-Follower Model:*

Below is a revised MATLAB script that implements the full leader-follower scheme using two nested grid‐based optimizations (no use of fminbnd).

%% Stackelberg-communication model: the MATLAB script

clearvars; close all; clc;

% 1) Model parameters

kappa = 2; % cost per unit of effort

phi = 1; % penalty for inconsistency

C1MAX\_vals = 0:0.1:10; % grid by C1MAX

C2MAX\_vals = C1MAX\_vals; % grid by C2MAX

% 2) Scenarios: [a, e, R1, R2, O1, O2, Ri1, Ri2]

scenarios = [

100, 80, 0.3, 0.3, 0.4, 0.4, 0.5, 0.5; % 1 Base

150, 80, 0.3, 0.3, 0.4, 0.4, 0.5, 0.5; % 2 High a

100, 120, 0.3, 0.3, 0.4, 0.4, 0.5, 0.5; % 3 High e

100, 80, 0.5, 0.5, 0.6, 0.6, 0.7, 0.7; % 4 High Risks

120, 100, 0.2, 0.2, 0.3, 0.3, 0.4, 0.4 % 5 Low Risks

];

nSc = size(scenarios,1);

% 3) Preparation for storing the results

results = zeros(nSc,4); % [C1MAX\*, C2MAX\*, J\_S\*, J\_A\*]

totalEff = zeros(numel(C1MAX\_vals),nSc);

J\_S\_all = zeros(numel(C1MAX\_vals),nSc);

colors = {'b','r','g','m','c'};

labels = {'Base','High a','High e','High Risks','Low Risks'};

% 4) The main scenario cycle

for s = 1:nSc

% Script Parameters

a = scenarios(s,1);

e = scenarios(s,2);

R1 = scenarios(s,3);

R2 = scenarios(s,4);

O1 = scenarios(s,5);

O2 = scenarios(s,6);

Ri1 = scenarios(s,7);

Ri2 = scenarios(s,8);

% We calculate the constant factors

B1 = (1-R1^2)\*(1-O1)\*(1-Ri1);

B2 = (1-R2^2)\*(1-O2)\*(1-Ri2);

% Time vectors

J\_S\_react = zeros(1,numel(C1MAX\_vals));

C2MAX\_react = zeros(1,numel(C1MAX\_vals));

J\_A\_max = zeros(1,numel(C1MAX\_vals));

% 4.1) Follower (agent) – internal optimization for each C1MAX

for i1 = 1:numel(C1MAX\_vals)

C1MAX = C1MAX\_vals(i1);

% Vector by C2MAX

J\_A\_row = zeros(1,numel(C2MAX\_vals));

J\_S\_row = zeros(1,numel(C2MAX\_vals));

for i2 = 1:numel(C2MAX\_vals)

C2MAX = C2MAX\_vals(i2);

T = 1 - (C1MAX + C2MAX)/10;

M1 = B1\*(1 - sqrt(T));

M2 = B2\*(1 - sqrt(T));

% Agent's profit

J\_A\_row(i2) = e\*( log10(C2MAX+1)\*M2 ...

- kappa\*C2MAX^2 ...

- phi\*(C2MAX-C1MAX)^2 );

% The «reactionary» profit of the head at the same time is C2MAX

J\_S\_row(i2) = a\*( log10(C1MAX+1)\*M1 - kappa\*C1MAX^2 );

end

% Choosing the best C2MAX\*

[J\_A\_max(i1), idxA ] = max(J\_A\_row);

C2MAX\_react(i1) = C2MAX\_vals(idxA);

J\_S\_react(i1) = J\_S\_row(idxA);

% We save the dependence of J\_S on the total effort for the graph

totalEff(i1,s) = C1MAX + C2MAX\_react(i1);

J\_S\_all(i1,s) = J\_S\_react(i1);

end

% 4.2) Leader (supervisor) – selects C1MAX\*, maximizing J\_S\_react

[J\_Sopt, idxS] = max(J\_S\_react);

C1MAXopt = C1MAX\_vals(idxS);

C2MAXopt = C2MAX\_react(idxS);

J\_Aopt = J\_A\_max(idxS);

results(s,:) = [C1MAXopt, C2MAXopt, J\_Sopt, J\_Aopt];

end

% 5) Chart 1: J\_S vs (C1MAX+C2MAX)

figure(1); hold on; grid on;

for s = 1:nSc

plot(totalEff(:,s), J\_S\_all(:,s), ...

colors{s}, 'LineWidth',1.5, 'DisplayName',labels{s});

plot(results(s,1)+results(s,2), results(s,3), ...

[colors{s} 'o'], 'MarkerSize',8,'LineWidth',2);

end

title('Supervisor Profit vs Total Communication Effort');

xlabel('C\_1 + C\_2'); ylabel('J\_S');

legend('Location','best');

% 6) Graph 2: C1MAX\* and C2MAX\* by scenario

figure(2); hold on; grid on;

scn = 1:nSc;

plot(scn, results(:,1), 'b-o','LineWidth',1.5, 'DisplayName','C\_1^\*');

plot(scn, results(:,2), 'g-s','LineWidth',1.5, 'DisplayName','C\_2^\*');

set(gca,'XTick',scn,'XTickLabel',labels);

title('Optimal Communication Efforts by Scenario');

xlabel('Scenario'); ylabel('Effort');

legend('Location','best');

% 7) Output of results to the console

disp(' Scn | C\_1max^\* C\_2max^\* J\_S^\* J\_A^\*');

disp([(1:nSc)', results]);

*The results of the model's work in Matlab:*

A screenshot of a computer

AI-generated content may be incorrect.

A screen shot of a graph

AI-generated content may be incorrect.

A screen shot of a graph

AI-generated content may be incorrect.