Appendix

The code for genetic algorithm implementation in this study

```
% Number of periods
n = 12; % Example for 12 time periods

% Data
S_min = 2.4;   % Minimum storage (million cubic meters)
S_max = 236.69;  % Maximum storage
R_min = 0;    % Minimum release
R_max = 150;  % Maximum release
I = [24.13, 41.09, 34.60, 21.84, 23.51, 12.54, 5.79, 3.42, 1.34, 2.46, 3.68, 8.65]; %
Inflow for each period (Mm3)
D = [0.78, 0.78, 0.78, 22.69, 29.73, 44.92, 61.09, 21.99, 0.78, 0.78, 0.78, 41.73]; %
Demand for each period (Mm3)
E = [0.70, 0.99, 1.15, 1.08, 0.77, 0.46, 0.23, 0.12, 0.08, 0.09, 0.13, 0.23]; %
Evaporation for each period (Mm3)
initial_storage = 53.01; % Initial reservoir storage

% Constraints
lb = ones(1, n) * R_min; % Lower bounds for releases
ub = D; % Upper bounds for releases

% Genetic Algorithm Options
options = optimoptions('ga', ...
    'PopulationSize', 500, ...
    'MaxGenerations', 100, ...
    'CrossoverFraction', 0.8, ...
    'CrossoverFcn', @crossoverscattered, ...
    'MutationFcn', {@mutationadaptfeasible, 0.01, 0.5}, ...
    'Display', 'iter', ...
    'PlotFcn', {@gaplotbestf}, ...
    'OutputFcn', @trackBestFitness); % Add output function; % Plot best fitness

% Solve using GA
[optimal_release, min_deficiency] = ga(@(R) objective(R, I, D, E, S_min, S_max,
initial_storage), ...
    n, [], [], [], [], lb, ub, [], options);

 % Compute Storage Values
S = computeStorage(optimal_release, I, E, S_min, S_max, initial_storage);

% Display Results
disp('Optimal Release Policy (Mm³):');
disp(optimal_release);
disp('Final Storage Levels (Mm³):');
```

```
disp(S);
disp('Minimum Total Squared Relative Deficiency:');
disp(min_deficiency);
figure;
plot(best_fitness_history, 'LineWidth', 2);
xlabel('Generation');
ylabel('Best Fitness Value');
title('Convergence of GA Optimization');
grid on;

% Objective Function
function J = objective(R, I, D, E, S_min, S_max, initial_storage, ~)
    n = length(I); % Number of periods
    S = zeros(1, n+1); % Storage at each time step
    S(1) = initial_storage; % Initial storage
    total_deficiency = 0;
    penalty = 0; % Penalty for violating constraints

    for t = 1:n
        SP = max(0, S(t) + I(t) - E(t) - R(t) - S_max); % Compute spill
        S(t+1) = S(t) + I(t) - R(t) - SP - E(t); % Water balance equation

        % Squared relative deficiency
        if D(t) > 0
            total_deficiency = total_deficiency + ((D(t) - R(t)) / D(t))^2;
        end

        % Penalize infeasible solutions
        if S(t+1) < S_min || S(t+1) > S_max
            penalty = penalty + 1e6; % Large penalty
        end
    end

    J = total_deficiency + penalty; % Add penalty to the objective
end

% Compute Storage Function
function S = computeStorage(R, I, E, S_min, S_max, initial_storage)
    n = length(I);
    S = zeros(1, n+1);
    S(1) = initial_storage;

    for t = 1:n
        SP = max(0, S(t) + I(t) - E(t) - R(t) - S_max); % Spill
        S(t+1) = S(t) + I(t) - R(t) - SP - E(t); % Water balance
    end
end
```

```
function [state, options, flag] = trackBestFitness(options, state, flag)
   persistent bestFitnessHistory
   switch flag
      case 'init'
         bestFitnessHistory = []; % Initialize the history at the beginning
      case 'iter'
         bestFitnessHistory = [bestFitnessHistory; state.Best(end)]; % Track the best
fitness
      case 'done'
         assignin('base', 'best_fitness_history', bestFitnessHistory); % Save to base
workspace
   end
end
```

CLIMATE CHANGE IMPACT ON OPERATION POLICY AND PERFORMANCE INDICES OF A RESERVOIR USING MACHINE LEARNING TECHNIQUES

By

Er. VINNAKOTA YESUBABU
(2020-28-002)

ABSTRACT

Submitted in partial fulfilment of the requirement for the degree

of

DOCTOR OF PHILOSOPHY

IN

AGRICULTURAL ENGINEERING

(Soil and Water Conservation Engineering)

Faculty of Agricultural Engineering and Technology

Kerala Agricultural University

Department of Soil and Water Conservation Engineering

KELAPPAJI COLLEGE OF AGRICULTURAL ENGINEERING AND FOOD TECHNOLOGY

TAVANUR, MALAPPURAM-679573

KERALA, INDIA

2025