

```

clear all
close all
clc

% --- System Parameters ---
h0 = 550; iK = 53; Nsat = 22; Norbits = 72; U = 50; dt = 10; Tmax = 3600;
deltaPhi = 30; M = Nsat * Norbits; T = Tmax / dt + 1;
user_latitudes = 45 + (50 - 45) * rand(U, 1);
user_longitudes = 55 + (60 - 55) * rand(U, 1);

elevation_ts_all = zeros(M, T, U); distance_ts_all = zeros(M, T, U);
for u = 1:U
    for ind = 1:Norbits
        base_idx = (ind - 1) * Nsat + 1;
        [elev_ts, dist_ts, ~] = generate_passage_singleorbit(h0, iK, ...
            (ind - 1) * deltaPhi, -(ind - 1) * 360 / Nsat / Norbits, ...
            Nsat, user_latitudes(u), user_longitudes(u), dt, Tmax);
        elevation_ts_all(base_idx:base_idx + Nsat - 1, :, u) = elev_ts;
        distance_ts_all(base_idx:base_idx + Nsat - 1, :, u) = dist_ts;
    end
end

% --- Elevation Quantization ---
% allowed_elevs = [20, 30, 45, 60, 70];
allowed_elevs = [30, 45, 60, 70]; % not using 20 since the threshold is
25
% partition = [5, 25, 37.5, 52.5, 65, inf];
partition = [25, 37.5, 52.5, 65, inf];
elevation_quantized_all = NaN(size(elevation_ts_all));
for u = 1:U
    quantized_indices = discretize(elevation_ts_all(:, :, u), partition);
    temp = NaN(size(elevation_ts_all(:, :, u)));
    valid_idx = ~isnan(quantized_indices);
    temp(valid_idx) = allowed_elevs(quantized_indices(valid_idx));
    temp(elevation_ts_all(:, :, u) < 5) = NaN;
    elevation_quantized_all(:, :, u) = temp;
end

% --- Channel Fading Traces ---
f_c = 2e9; envi = 'Suburban'; Tch = 5000;
channel_trace_map = containers.Map('KeyType', 'double', 'ValueType',
'any');
for k = 1:length(allowed_elevs)
    elev = allowed_elevs(k);
    try
        [y, ~, ~, ~, ~, ~] = coef_time_series_long_pedestrian(f_c, envi,
elev, Tch);
        channel_trace_map(elev) = abs(y).^2; % Only average used
    catch
        channel_trace_map(elev) = zeros(1, Tch);
    end
end

% --- Transmit power values (in Watts) ---
Ps_values = [0.1 0.2 0.5 1 1.5 2];

% Preallocate results for plotting and analysis
avg_throughput_ideal = zeros(length(Ps_values), 1);

```

```

avg_throughput_forecast = zeros(length(Ps_values), 1);
avg_snr_ideal           = zeros(length(Ps_values), 1);
avg_snr_forecast        = zeros(length(Ps_values), 1);

results_ideal           = cell(length(Ps_values), 1);
results_forecast        = cell(length(Ps_values), 1);

sum_snr_ideal           = cell(length(Ps_values), 1);
sum_snr_forecast        = cell(length(Ps_values), 1);
sum_rate_ideal          = cell(length(Ps_values), 1);
sum_rate_forecast       = cell(length(Ps_values), 1);

for i = 1:length(Ps_values)
    Ps = Ps_values(i);

    % --- Build weight matrices for both assignment and evaluation ---
    [W_SNR_ideal, W_Rate_ideal] = build_weight_matrices(Ps,
channel_trace_map, elevation_quantized_all, distance_ts_all,
allowed_elevs, f_c, 'ideal');
    [W_SNR_forecast, W_Rate_forecast] = build_weight_matrices(Ps,
channel_trace_map, elevation_quantized_all, distance_ts_all,
allowed_elevs, f_c, 'forecast');

    % --- Ideal Mode ---
    [snr_ideal, rate_ideal, snr_ideal_primary, snr_ideal_dual,
rate_ideal_primary, rate_ideal_dual] = ...
        simulate_ideal_mode_algo_1(W_SNR_ideal, W_Rate_ideal,
elevation_quantized_all, distance_ts_all, channel_trace_map,
allowed_elevs, f_c, Ps);
    avg_throughput_ideal(i) = mean(rate_ideal(:), 'omitnan');
    avg_snr_ideal(i)        = mean(snr_ideal(:), 'omitnan');
    results_ideal{i} = struct('snr', snr_ideal, 'rate', rate_ideal, ...
        'snr_primary', snr_ideal_primary, 'snr_dual', snr_ideal_dual, ...
        'rate_primary', rate_ideal_primary, 'rate_dual',
rate_ideal_dual);
    sum_snr_ideal{i} = sum(snr_ideal, 1, 'omitnan'); % [1 x T], sum
over users
    sum_rate_ideal{i} = sum(rate_ideal, 1, 'omitnan'); % [1 x T], sum
over users

    % --- Forecast Mode ---
    [snr_forecast, rate_forecast, snr_forecast_primary,
snr_forecast_dual, rate_forecast_primary, rate_forecast_dual] = ...
        simulate_forecast_mode_algo_1(W_SNR_forecast, W_Rate_forecast,
W_SNR_ideal, W_Rate_ideal, elevation_quantized_all, distance_ts_all,
channel_trace_map, allowed_elevs, f_c, Ps);
    avg_throughput_forecast(i) = mean(rate_forecast(:), 'omitnan');
    avg_snr_forecast(i)        = mean(snr_forecast(:), 'omitnan');
    results_forecast{i} = struct('snr', snr_forecast, 'rate',
rate_forecast, ...
        'snr_primary', snr_forecast_primary, 'snr_dual',
snr_forecast_dual, ...
        'rate_primary', rate_forecast_primary, 'rate_dual',
rate_forecast_dual);
    sum_snr_forecast{i} = sum(snr_forecast, 1, 'omitnan'); % [1 x T]
    sum_rate_forecast{i} = sum(rate_forecast, 1, 'omitnan'); % [1 x T]
end

```

```

% --- Plotting Section:

output_folder = fullfile(pwd, 'Plots_LEO_Handover_algo_1');
if ~exist(output_folder, 'dir')
    mkdir(output_folder);
end
save_plot = @(name) saveas(gcf, fullfile(output_folder, name));

i_plot = 3; user_idx = 1;
d_ideal = results_ideal{i_plot};
d_forecast = results_forecast{i_plot};

% 1. Average Throughput vs Transmit Power
figure;
semilogx(Ps_values, avg_throughput_ideal, 'b-o', 'LineWidth', 1.6); hold
on;
semilogx(Ps_values, avg_throughput_forecast, 'r--s', 'LineWidth', 1.6);
xlabel('Transmit Power (W)', 'FontWeight', 'bold');
ylabel('Average Throughput (bps/Hz)', 'FontWeight', 'bold');
ylim([0 inf]);
title(['Throughput vs Transmit Power', ...
    'U = 50 users, T = 361 time steps, Suburban Environment'],
    'FontWeight', 'bold');
legend('Ideal Mode', 'Forecast Mode', 'Location', 'best');
grid on;
save_plot('1_Throughput_vs_Ps_1.png');

% 2. Average SNR vs Transmit Power
figure;
semilogx(Ps_values, avg_snr_ideal, 'b-o', 'LineWidth', 1.6); hold on;
semilogx(Ps_values, avg_snr_forecast, 'r--s', 'LineWidth', 1.6);
xlabel('Transmit Power (W)', 'FontWeight', 'bold');
ylabel('Average SNR (dB)', 'FontWeight', 'bold');
ylim([0 inf]);
title(['SNR vs Transmit Power', ...
    'U = 50 users, T = 361 time steps, Suburban Environment'],
    'FontWeight', 'bold');
legend('Ideal Mode', 'Forecast Mode', 'Location', 'best');
grid on;
save_plot('2_SNR_vs_Ps_1.png');

% 3. System Sum Rate vs Time
figure;
plot(sum_rate_ideal{i_plot}, 'b-', 'LineWidth', 2); hold on;
plot(sum_rate_forecast{i_plot}, 'r--', 'LineWidth', 2);
xlabel('Time Step (10s)', 'FontWeight', 'bold');
ylabel('Total System Rate (bps/Hz)', 'FontWeight', 'bold');
legend('Ideal', 'Forecast', 'Location', 'best');
title('Total System Rate per Time Step', 'FontWeight', 'bold');
grid on;
save_plot('3_System_Sum_Rate_vs_Time_1.png');

% 4. System Sum SNR vs Time
figure;
plot(sum_snr_ideal{i_plot}, 'b-', 'LineWidth', 2); hold on;
plot(sum_snr_forecast{i_plot}, 'r--', 'LineWidth', 2);
xlabel('Time Step (10s)', 'FontWeight', 'bold');
ylabel('Total System SNR (dB)', 'FontWeight', 'bold');

```

```

legend('Ideal', 'Forecast', 'Location', 'best');
title('Total System SNR per Time Step', 'FontWeight', 'bold');
grid on;
save_plot('4_System_Sum_SNR_vs_Time_1.png');

% 5. Single-User Rate Evolution
figure;
plot(d_ideal.rate(user_idx,:), 'b-', 'LineWidth', 2); hold on;
plot(d_forecast.rate(user_idx,:), 'r--', 'LineWidth', 2);
xlabel('Time Step (10s)', 'FontWeight', 'bold');
ylabel('Rate (bps/Hz)', 'FontWeight', 'bold');
legend('Ideal', 'Forecast', 'Location', 'best');
title(sprintf('Rate Evolution - User %d', user_idx), 'FontWeight',
'bold');
grid on;
save_plot(sprintf('5_SingleUser_Rate_User%d_1.png', user_idx));

% 6. Single-User SNR Evolution
figure;
plot(d_ideal.snr(user_idx,:), 'b-', 'LineWidth', 2); hold on;
plot(d_forecast.snr(user_idx,:), 'r--', 'LineWidth', 2);
xlabel('Time Step (10s)', 'FontWeight', 'bold');
ylabel('SNR (dB)', 'FontWeight', 'bold');
legend('Ideal', 'Forecast', 'Location', 'best');
title(sprintf('SNR Evolution - User %d', user_idx), 'FontWeight',
'bold');
grid on;
save_plot(sprintf('6_SingleUser_SNR_User%d_1.png', user_idx));

% 7. System Primary vs Dual Sums
figure;
plot(sum(d_ideal.rate_primary, 1, 'omitnan'), 'b-', 'LineWidth', 1.8);
hold on;
plot(sum(d_ideal.rate_dual, 1, 'omitnan'), 'r--', 'LineWidth', 1.8);
plot(sum(d_ideal.rate, 1, 'omitnan'), 'k:', 'LineWidth', 2.2);
xlabel('Time Step (10s)', 'FontWeight', 'bold');
ylabel('Sum Rate (bps/Hz)', 'FontWeight', 'bold');
legend('Primary', 'Dual', 'Total', 'Location', 'best');
title('Ideal: System Primary, Dual, Total Rate', 'FontWeight', 'bold');
grid on;
save_plot('7_System_Primary_Dual_Total_Rate_Ideal_1.png');

figure;
plot(sum(d_ideal.snr_primary, 1, 'omitnan'), 'b-', 'LineWidth', 1.8);
hold on;
plot(sum(d_ideal.snr_dual, 1, 'omitnan'), 'r--', 'LineWidth', 1.8);
plot(sum(d_ideal.snr, 1, 'omitnan'), 'k:', 'LineWidth', 2.2);
xlabel('Time Step (10s)', 'FontWeight', 'bold');
ylabel('Sum SNR (dB)', 'FontWeight', 'bold');
legend('Primary', 'Dual', 'Total', 'Location', 'best');
title('Ideal: System Primary, Dual, Total SNR', 'FontWeight', 'bold');
grid on;
save_plot('8_System_Primary_Dual_Total_SNR_Ideal_1.png');

% 8. CDF of Average User Rate
user_rate_mean_ideal = mean(d_ideal.rate, 2, 'omitnan');
user_rate_mean_forecast = mean(d_forecast.rate, 2, 'omitnan');
valid_ideal = user_rate_mean_ideal(~isnan(user_rate_mean_ideal));

```

```

valid_forecast =
user_rate_mean_forecast(~isnan(user_rate_mean_forecast));
if isempty(valid_ideal) || isempty(valid_forecast)
    warning('No valid user averages for RATE CDF plot.');
```

else

```

    figure;
    cdfplot(valid_ideal); hold on;
    cdfplot(valid_forecast);
    legend('Ideal', 'Forecast', 'Location', 'best');
    xlabel('Average Rate per User (bps/Hz)', 'FontWeight', 'bold');
    ylabel('Cumulative Probability', 'FontWeight', 'bold');
    title('CDF of Average User Rate', 'FontWeight', 'bold');
    grid on;
    save_plot('9_CDF_AvgUserRate_1.png');
```

end

```

% 9. CDF of Average User SNR
user_snr_mean_ideal = mean(d_ideal.snr,2,'omitnan');
user_snr_mean_forecast = mean(d_forecast.snr,2,'omitnan');
valid_ideal_snr = user_snr_mean_ideal(~isnan(user_snr_mean_ideal));
valid_forecast_snr =
user_snr_mean_forecast(~isnan(user_snr_mean_forecast));
if isempty(valid_ideal_snr) || isempty(valid_forecast_snr)
    warning('No valid user averages for SNR CDF plot.');
```

else

```

    figure;
    cdfplot(valid_ideal_snr); hold on;
    cdfplot(valid_forecast_snr);
    legend('Ideal', 'Forecast', 'Location', 'best');
    xlabel('Average SNR per User (dB)', 'FontWeight', 'bold');
    ylabel('Cumulative Probability', 'FontWeight', 'bold');
    title('CDF of Average User SNR', 'FontWeight', 'bold');
    grid on;
    save_plot('10_CDF_AvgUserSNR_1.png');
```

end