

```

clc;
clear;
close all;

% n 값 범위 설정
n_values = 2.^(1:8); %2의 몇 제곱까지 증가할 것인지 결정 (여기서는 2^8까지)

% 각 n 값에 따른 총 오차 계산
total_errors = zeros(size(n_values));

for idx = 1:numel(n_values)
    n = n_values(idx);

    % triangle mesh 생성하기
    m = n; % 세로 element 갯수

    % domain 범위 설정
    v1=[0,0];
    v2=[1,0];
    v3=[1,1];
    v4=[0,1];

    [v4e, c4v] = triangle2dmesh_domain(m, n, v1, v2, v3, v4);

    % I2_projection_2d
    f = @(x,y) sin(pi*x).*sin(pi*y);

    % M = local_M_2_M(v4e, c4v);

    area = 1/(2*m*n);
    M_k = [2 1 1; 1 2 1; 1 1 2] / 12 * area;
    M_k(:);
    M = sparse(repmat(v4e, 3, 1), repelem(v4e, 3, 1), repmat(M_k(:), 1 ,size(repelem(v4e, 3, 1),2) ));

    % b = local_b_2_b(v4e, c4v, f);

    x4e = c4v(v4e,1);
    y4e = c4v(v4e,2);
    b_k = f(x4e,y4e) / 3 * area;
    b = accumarray(v4e(:), b_k);

    Pf = MWb;

    % 각 요소의 부피를 저장할 변수 초기화
    integration_errors = zeros(size(v4e, 2), 1);

    % 각 요소에 대한 적분 값 계산
    for i = 1:size(v4e, 2)
        % 현재 요소의 꼭지점 좌표
        vertices = c4v(v4e(:, i), :);

        % 각 (x,y) 쌍에 대한 f(x,y) 계산
        x = vertices(:, 1);
        y = vertices(:, 2);
        result = arrayfun(f, x, y);

        % f_matrix 생성
        f_matrix = [x, y, result];

        % 세 점으로부터 평면 방정식 계산
        A = (y(2) - y(1)) * (result(3) - result(1)) - (result(2) - result(1)) * (y(3) - y(1));
        B = (result(2) - result(1)) * (x(3) - x(1)) - (x(2) - x(1)) * (result(3) - result(1));
        C = (x(2) - x(1)) * (y(3) - y(1)) - (y(2) - y(1)) * (x(3) - x(1));
        D = -A * x(1) - B * y(1) - C * result(1);

        % f 평면 방정식 저장
        f_plane = A * x + B * y + C * result + D;

        % Pf 평면 방정식 구하기

```

```

Pf_values = Pf(v4e(:,i));

% 세 점으로부터 평면 방정식 계산
v1 = [x(2)-x(1), y(2)-y(1), Pf_values(2)-Pf_values(1)];
v2 = [x(3)-x(1), y(3)-y(1), Pf_values(3)-Pf_values(1)];
n = cross(v1, v2); % 법선 벡터
A = n(1);
B = n(2);
C = n(3);
D = -A * x(1) - B * y(1) - C * Pf_values(1);

% 평면 방정식 저장
Pf_plane = [A, B, C, D];

% f_plane - Pf_plane 계산
f_plane_minus_Pf_plane = f_plane - Pf_plane;

% 적분 값 계산
integral_value = integral2(@(x,y) f_plane_minus_Pf_plane(1)*x + f_plane_minus_Pf_plane(2)*y + f_plane_minus_Pf_plane(3), ...
    min(x), max(x), min(y), max(y));

% 에러 계산
integration_errors(i) = abs(integral_value);
end

% 총 에러 계산
total_error = sqrt(sum(integration_errors));
total_errors(idx) = total_error;
end

% 결과 플롯
figure;
loglog(n_values, total_errors, '-o', 'LineWidth', 2);
title('Total Error vs. Number of Elements (n)');
xlabel('Number of Elements (n)');
ylabel('Total Error');
grid on;

% error rate 계산 및 출력
error_rate = -(log(total_errors(2:end)) - log(total_errors(1:end-1))) ./ log(2);
disp('Error rate:');
disp(error_rate);

```

Error rate:

1.4230	1.2452	1.1131	1.0500	1.0229	1.0108	1.0053
--------	--------	--------	--------	--------	--------	--------

