

ANEXO A.2 CÁLCULO DE CAMPOS ELECTROMAGNÉTICOS

Siguiendo la Guía Metodológica N° 07: Cálculo de campos electromagnéticos en redes de distribución de EPM, se ha elaborado en software Matlab un código que halla la intensidad de los campos eléctrico y magnético debido a una red de distribución eléctrica en un punto específico. Esto con el fin de comparar los valores obtenidos con los valores límites exigidos por el Retie.

El código desarrollado se presenta a continuación:

```
%% Script: matriz_potencial.m
% Construye la matriz de coeficientes de potencial P usando valores ingresados
% manualmente y, además, calcula:
% - La matriz de capacitancia  $C = \text{inv}(P)$ 
% - Las tensiones (V) y, a partir de ellas, las cargas complejas  $Q = C*V$ 
% - El campo eléctrico en un punto de observación M, considerando que para cada fase
%   se utiliza su propia distancia horizontal (xm) respecto al punto M.
%
% Fórmulas:
% - Potencial:
%    $P_{kk} = (1/(2*\pi*\epsilon)) * \ln(4*hk/d_k)$ 
%    $P_{kl} = (1/(2*\pi*\epsilon)) * \ln(S'_{kl}/S_{kl})$ 
%
% - Campo eléctrico en M, para cada fase k:
%    $E_{kx} = (Q_k/(2*\pi*\epsilon)) * [ xm(k)/(xm(k)^2 + (hk(k)-H_M)^2) -$ 
 $xm(k)/(xm(k)^2 + (hk(k)+H_M)^2) ]$ 
%    $E_{ky} = (Q_k/(2*\pi*\epsilon)) * [ (H_M - hk(k))/(xm(k)^2 + (hk(k)-H_M)^2) -$ 
 $(H_M + hk(k))/(xm(k)^2 + (hk(k)+H_M)^2) ]$ 
%
% Se utiliza:
% - d: vector de diámetros de cada fase (m)
% - hk: vector de alturas de cada fase sobre el suelo (m)
% - xm: vector de distancias horizontales de cada fase al punto M (m)
% - S: matriz de distancias horizontales entre fases
% - S_p: matriz de distancias entre cada fase y la imagen de la otra
% - epsilon: Permisividad del vacío.
% - Q: Cargas complejas en las fases.

clc; clear; close all;

%% 1) Constantes y entrada de datos
epsilon = 8.854e-12; % Permisividad del vacío (F/m)
```

```

% Solicitar al usuario el número de fases
n = input('Ingrese el número de fases: ');

% Preasignar vectores de tamaño n
d = zeros(n,1); % Diámetros
hk = zeros(n,1); % Alturas sobre el suelo
xm = zeros(n,1); % Distancias horizontales desde cada fase hasta el punto M

% Ingreso manual de los valores para cada fase
d(1:n) = input(sprintf('Diámetro fase %d (mm): '), 1:n)*1e-3;
for k = 1:n
    hk(k) = input(sprintf('Altura sobre el suelo fase %d (m): ', k));
    xm(k) = input(sprintf('Distancia horizontal de la fase %d al punto M (m): ', k));
end
V_fase = input(sprintf('Tensión de la línea (V): '))/sqrt(3)

disp('Vector de diámetros (mm):'); disp(d);
disp('Vector de alturas (m):'); disp(hk);
disp('Vector de distancias xm (m):'); disp(xm);

%% Cálculo de la matriz S (distancia horizontal entre fases)
% Se asume que la separación horizontal entre las fases es la diferencia
% en sus distancias al punto M (asumiendo que están en una misma línea).
S = zeros(n);
for i = 1:n
    for j = 1:n
        if i == j
            S(i,j) = 0;
        else
            S(i,j) = abs(xm(i) - xm(j));
        end
    end
end
disp('Matriz S (distancia horizontal entre fases):');
disp(S);

%% Cálculo de la matriz S_p (distancia entre la fase i y la imagen de la fase j)
% La imagen se obtiene reflejando la altura: se asume que la imagen de la
% fase j está a -hk(j) (respecto al suelo).
S_p = zeros(n);
for i = 1:n
    for j = 1:n
        if i == j
            S_p(i,j) = 0;
        else

```

```

        % Se calcula S_p usando:
        % S'_ij = sqrt( (S_ij)^2 + (hk(i) + hk(j))^2 )
        S_p(i,j) = sqrt( S(i,j)^2 + (hk(i) + hk(j))^2 );
    end
end
end
disp('Matriz S_p (distancia entre cada fase y la imagen de la otra:');
disp(S_p);

%% 2) Construcción de la matriz de coeficientes de potencial P
P = zeros(n);

% Términos diagonales: P_kk = (1/(2*pi*epsilon)) * ln(4*hk(k)/d(k))
for k = 1:n
    P(k,k) = (1/(2*pi*epsilon)) * log((4*hk(k))/d(k));
end

% Términos fuera de la diagonal: P_kl = (1/(2*pi*epsilon)) * ln(S'_kl/S_kl)
for i = 1:n
    for j = i+1:n
        P_val = (1/(2*pi*epsilon)) * log(S_p(i,j)/S(i,j));
        P(i,j) = P_val;
        P(j,i) = P_val; % Se asume que la matriz es simétrica
    end
end

disp('Matriz de coeficientes de potencial P:');
disp(P);

%% 3) Cálculo de la matriz de capacitancias C = inv(P)
C = inv(P);
disp('Matriz de capacitancias C (C = inv(P)):');
disp(C);

%% 4) Cálculo de las tensiones y cargas complejas Q
% Se asume un sistema trifásico (por ejemplo, 44 kV L-L).

% Para fines de ejemplo se definen las tensiones en forma compleja:
V = [ V_fase * 1;
      V_fase * exp(-1i*pi/3);
      V_fase * exp(1i*pi/3) ];
disp('Vector de tensiones en cada fase (V):');
disp(V);

% Se calculan las cargas complejas en las fases: Q = C * V

```

```

Q = C * V;
disp('Cargas complejas en cada fase (Q):');
disp(Q);

%% 5) Cálculo del campo eléctrico en el punto de observación M
% Se ingresa la altura del punto M sobre el suelo.
H_M = input('Ingrese la altura del punto de observación M sobre el suelo (m): ');

E_x_total = 0;
E_y_total = 0;

for k = 1:n
    % Para cada fase se utiliza:
    % xm(k): distancia horizontal de la fase k al punto M,
    % hk(k): altura de la fase k,
    % Q(k): carga compleja de la fase k.
    factor = Q(k) / (2*pi*epsilon);

    % Componente x del campo eléctrico generado por la fase k
    E_kx = factor * ( xm(k) / ( xm(k)^2 + (hk(k) - H_M)^2 ) ...
        - xm(k) / ( xm(k)^2 + (hk(k) + H_M)^2 ) );

    % Componente y del campo eléctrico generado por la fase k
    E_ky = factor * ( (H_M - hk(k)) / ( xm(k)^2 + (hk(k) - H_M)^2 ) ...
        - (H_M + hk(k)) / ( xm(k)^2 + (hk(k) + H_M)^2 ) );

    E_x_total = E_x_total + E_kx;
    E_y_total = E_y_total + E_ky;
end

E_rms = sqrt( (real(E_x_total))^2 + (imag(E_x_total))^2 + ...
    (real(E_y_total))^2 + (imag(E_y_total))^2 )/1e3;

fprintf('\n-----\n');
fprintf('Campo eléctrico total en el punto de observación M:\n');
fprintf(' E_x_total = %g + j%g V/m\n', real(E_x_total), imag(E_x_total));
fprintf(' E_y_total = %g + j%g V/m\n', real(E_y_total), imag(E_y_total));
fprintf('E_rms = %g kV/m\n', E_rms);
fprintf('-----\n');

%% 6) [NUEVO] Cálculo del campo magnético en M
% Según las ecuaciones de B_kx y B_ky (Ecuaciones 17 y 18),
% se necesita la corriente compleja en cada fase. A modo de ejemplo, definimos:
I = 131.22.*[
    1 * exp(1i*0);
    1 * exp(1i*(-pi/3));

```

```

    1* exp(1i*(pi/3))
];
% (Valores en A, cada fase con magnitud 200 A y distintos ángulos.)
% Ajusta estos valores o ingrésalos manualmente según tu problema real.

B_x_total = 0;
B_y_total = 0;

mu_factor = 2e-7; % 2 x 10^-7 en SI

for k = 1:n
    % i_k = parte compleja de la corriente
    i_k = I(k); % Corriente compleja de la fase k

    % B_kx = mu_factor * i_k * ( x_M - x_k ) / [ (x_M - x_k)^2 + (H_M - H_k)^2 ]
    % En el código, xm(k) representa |x_M - x_k|.
    % Si fuese relevante el signo, ajusta la definición de xm(k).
    B_kx = mu_factor * i_k * ( xm(k) ) ...
           / ( xm(k)^2 + (H_M - hk(k))^2 );

    % B_ky = mu_factor * i_k * ( H_M - H_k ) / [ (x_M - x_k)^2 + (H_M - H_k)^2 ]
    B_ky = mu_factor * i_k * ( H_M - hk(k) ) ...
           / ( xm(k)^2 + (H_M - hk(k))^2 );

    B_x_total = B_x_total + B_kx;
    B_y_total = B_y_total + B_ky;
end

% Cálculo de B_rms
B_rms = sqrt( ...
    (real(B_x_total))^2 + (imag(B_x_total))^2 + ...
    (real(B_y_total))^2 + (imag(B_y_total))^2 ...
)*1e6;

fprintf('\n=== Campo magnético (B) en M ===\n');
fprintf('B_x_total = %g + j%g T\n', real(B_x_total), imag(B_x_total));
fprintf('B_y_total = %g + j%g T\n', real(B_y_total), imag(B_y_total));
fprintf('B_rms    = %g uT\n', B_rms);

```