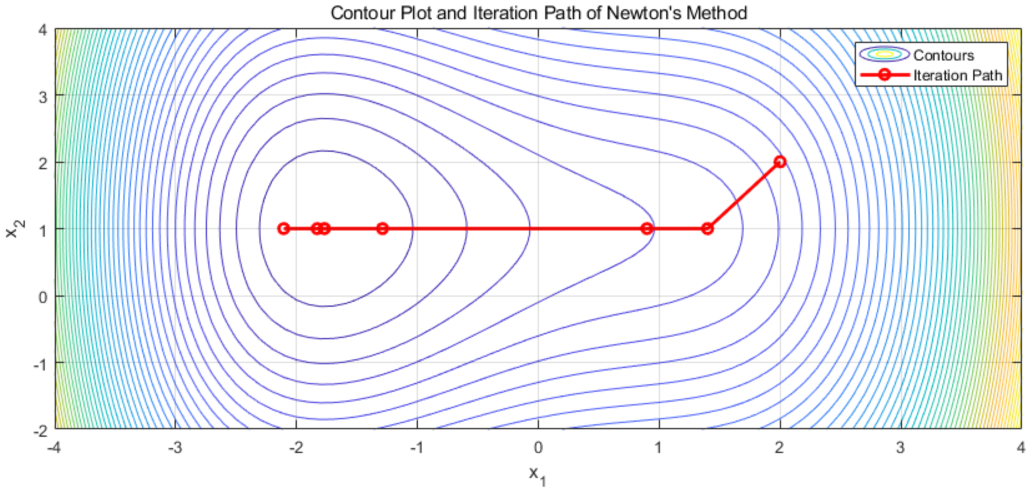
```matlab
% Optimization Howework3 Problem3(b)

% MATLAB Implementation of Newton's Method

% Define the cost function f
f = @(x1, x2) (x1.^4 / 4) - (x1.^2) + 2*x1 + (x2 - 1).^2;

% Gradient of f
grad_f = @(x1, x2) [x1^3 - 2*x1 + 2; 2*(x2 - 1)];

% Hessian matrix of f
Hessian_f = @(x1, x2) [3*x1^2 - 2, 0; 0, 2];

% Initialize x0 avoiding the points in problem (a)
x = [2; 2];

% Determine some parameters for Newton's method
tolerance = 1e-6;
max_iterations = 100;
iterations = 0;

% Store iteration points for visualization
path = x';

while norm(grad_f(x(1), x(2))) > tolerance && iterations < max_iterations

    % Compute the Hessian matrix and gradient at current point (x1, x2)
    H = Hessian_f(x(1), x(2));
    grad = grad_f(x(1), x(2));

    % Update step using Newton's method
    x = x - inv(H) * grad;

    % Store the new point
    path = [path; x'];

    % Increment iteration counter
    iterations = iterations + 1;

end

% Show the contour plot and the path traced out by the iterations

% Gerenate a 2D grid
[x1_vals, x2_vals] = meshgrid(-4:0.1:4, -2:0.1:4);

% Calculate values of f on the grid
f_vals = f(x1_vals, x2_vals);

% figure;
figure('Position', [100, 100, 1000, 400]);
% Show the contour plot
```

```
contour(x1_vals, x2_vals, f_vals, 50);
hold on;
plot(path(:, 1), path(:, 2), '-ro', 'LineWidth', 2, 'MarkerSize', 6);
xlabel('x_1');
ylabel('x_2');
title("Contour Plot and Iteration Path of Newton's Method");
grid on;
legend('Contours', 'Iteration Path');
```



Published with MATLAB® R2023b