HW2

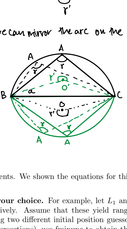
Worked with Vishwa Mehta

Thursday, October 10, 2024 4:44 AM

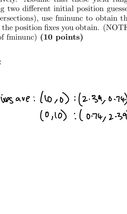
1) Problem 1. (Total: 10 points)

Let A, B, C be points in 2D space. Let a be the distance between points B and C and let γ be the value of the obtuse angle $\angle BAC$ in degrees. In class, we derived the position fix for the case of a subtended angle with two finite beacons. For the case of a right angle and for the case of an acute angle, we proved that the position fix is a portion of a circle with a specified center and radius. **Prove the corresponding result for the case where the subtended angle is obtuse** by showing that A is located in a portion of a circle that contains points A, B, and C, and give the radius of the circle in terms of a and γ. (HINT: $\gamma > 90°$, but $180° - \gamma < 90°$.)

- O is the center of the circle
- Using double angle theorem, $\gamma' = 2\gamma$
- Since $\gamma > 90°$, $\gamma' > 180°$
- ∴ the point A can be anywhere inside arc BC



- If we are unsure about facing north or south, then we can mirror the arc on the other side of BC, where the center of the circle is O'
- ∴ the point A can lie on either side of the line BC, on the arc BC with O as the center of the circle or on the arc BC with O' as the center of the circle
- this full shape is like a football, or a squeezed circle.

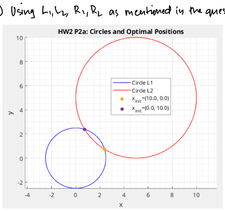


2) Problem 2. (Total: 30 points)

Consider the 2D case of a position fix using two range measurements. We shown the equations for this case in class. Use fminunc to solve these equations.

a) **Test and demonstrate your code using an example of your choice.** For example, let L_1 and L_2 be positioned at coordinates (0, 0) m and (5, 5) m respectively. Assume that these yield range measurements of $R_1 = 2.5$ m and $R_2 = 5$ m, respectively. Using two different initial position guesses ((10, 0) and (0, 10) for example, NOT near the actual circle intersections), use fminunc to obtain the two possible position fixes. Plot the two circles and place dots on the position fixes you obtain. (NOTE: Use the 'OptimalityTolerance' option to increase the accuracy of fminunc) **(10 points)**

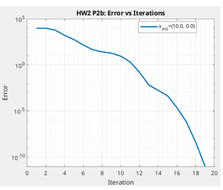
2.a) Using L_1, L_2, R_1, R_2 as mentioned in the question:



Fixes are: (10,0) : (2.38, 0.76)
(0,10) : (0.76, 2.38)

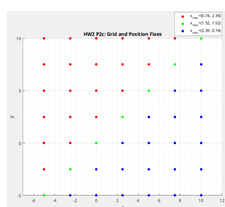
b) **Using one of your initial guesses from a), plot the distance error versus iteration plot from the fminunc optimization to check on the rate of convergence.** Let the distance error be the measured distance between the final output of fminunc and the position at each iteration of the optimization. Use semilogy. (NOTE: plot should look similar to the ones in Figures 2 and 4 from the fminunc tutorial in the Canvas page) **(10 points)**

2.b) Plotting rate of convergence for guess 1 (10, 0)



c) Since there are two possible position fixes, **start fminunc at a grid of points in a rectangle in the plane and determine what position fix each point converges to.** Then, use scatter to plot each of these initial points and color code them depending on which of the position fixes they converged to. For example, let the points of a grid be such that $p \in \{-5, -2.5, \ldots, 7.5, 10\} \times \{-5, -2.5, \ldots, 7.5, 10\}$. Then, put a red dot on each of the points that converges to one position fix and put a blue dot on each one that converges to the other position fix. (NOTE: this is a 2D version of Figure 6 from the fminunc tutorial in the Canvas page) **(10 points)**

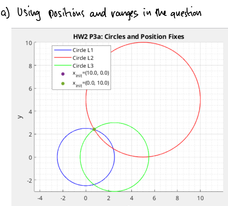
2.c) Using the grid given in the question



3) Problem 3. Modify your code in Problem 2 for the case of three range measurements. **(Total: 30 points)**

a) **Choose locations of the beacons and your location to test and demonstrate your code.** For example, let the beacons L_1, L_2, and L_3 be at (0, 0), (5, 5), and (2.5, 0) m respectively and let your position be (0.7212, 2.4080) m. Then, the distance from each beacon to you is approximately $R_1 = 2.5$ m, $R_2 = 5$ m, and $R_3 = 3$ m respectively. Use the positions of the beacons and their measured distances to obtain your position fix. Plot the circles and place dots on the position fixes you obtain through different initial guesses. **(10 points)**

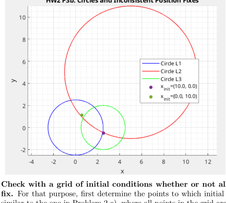
3.a) Using positions and ranges in the question



There is a single fix for both initial guesses: (0.72, 2.40) which matches the position in the question

b) **Test your code on an inconsistent case, that is, where there is no point where the three circles meet.** For example, in the previous case, change the measured distances to $R_1 = 2.5$ m, $R_2 = 6$ m, and $R_3 = 2$ m. Plot the circles and place dots on the position fixes you obtain through different initial guesses. **(10 points)**

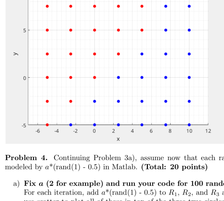
3.b)



Fixes: (10,0) : (2.52, −0.49)
(0,10) : (0.52, 1.11)

c) **Check with a grid of initial conditions whether or not all converge to the same position fix.** For that purpose, first determine the points to which initial guesses converge and present a grid similar to the one in Problem 2 c), where all points in the grid are color coded depending on the point to which they converge. **(10 points)**

3.c) Using the same grid as in (2.c)

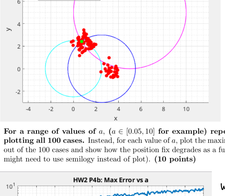


All initial guesses converge to one of
- (2.52, −0.49)
- (0.52, 1.11)

4) Problem 4. Continuing Problem 3a), assume now that each range measurement is corrupted by noise modeled by a*(rand(1) - 0.5) in Matlab. **(Total: 20 points)**

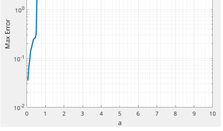
a) **Fix a (2 for example) and run your code for 100 random cases using a single initial guess.** For each beacon, add a*(rand(1) - 0.5) to R_1, R_2, and R_3 and save the obtained position fix. Then, use scatter to plot all of these in top of the three true circles in red. Put a dot of a different color on the true position fix (chosen in Problem 3). **(10 points)**

4.a)



b) **For a range of values of a, ($a \in [0.05, 10]$ for example) repeat what you did in a) without plotting all 100 cases.** Instead, for each value of a, plot the maximum distance from the true position out of the 100 cases and show how the position fix degrades as a function of sensor error. (NOTE: You might need to use semilogy instead of plot). **(10 points)**

4.b)



We can clearly see that as the noise (a) increases, the worst case scenario of position fix error degrades.

```
clc; clear; close all;
```

setup problem 2

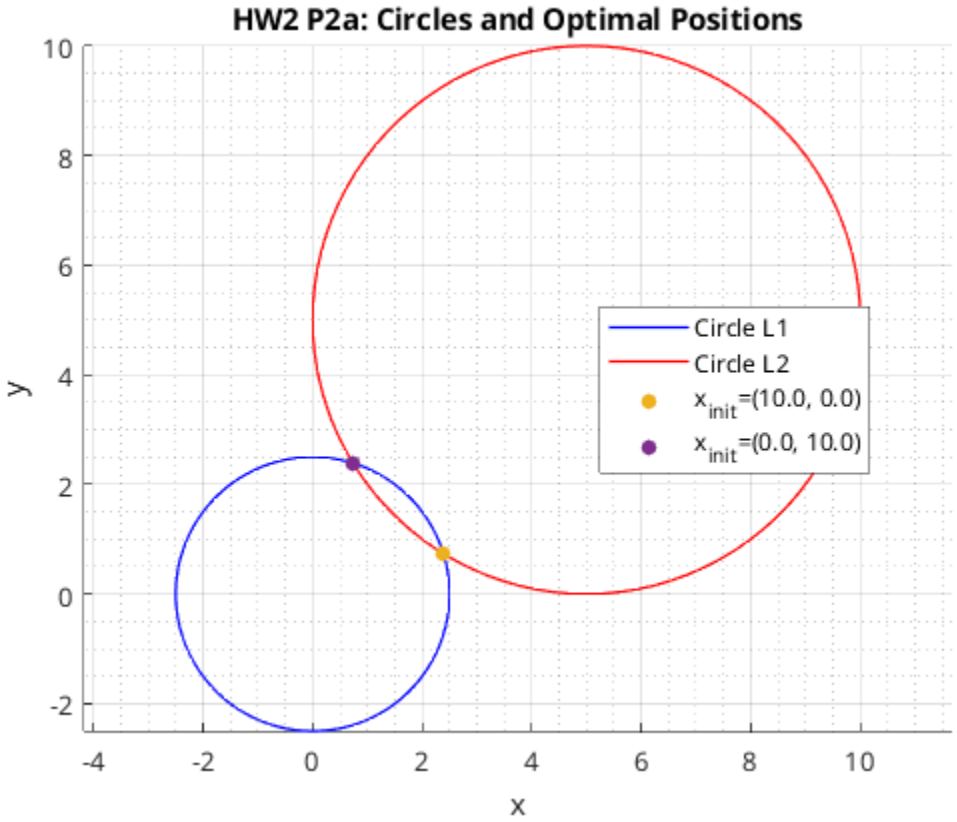
```
L1 = [0, 0]; L2 = [5, 5];
R1 = 2.5; R2 = 5;

% setup minimization
guesses = [
    10, 0;
    0, 10;
    ];

global iters iterates errvals;
iters = zeros(size(guesses, 1), 1);
iterates = zeros(size(guesses, 1), size(guesses, 2), 100);
errvals = zeros(size(guesses, 1), 100);

for i = 1:size(guesses, 1)
    options = optimoptions('fminunc', 'OptimalityTolerance', 1e-12,
'OutputFcn', @(x, optimValues, state) outfun(x, optimValues, state, i));
    [x, fval] = fminunc(@(x) circle_fix_eq_2d(x, L1, L2, R1, R2),
guesses(i, :), options);
    errvals(i, 1:iters(i)) = errvals(i, 1:iters(i)) - fval;
    fprintf('Optimal position for guess %d(%.1f, %.1f): (%.2f, %.2f)\n', i,
guesses(i, 1), guesses(i, 2), x(1), x(2));
end
```

```
Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 1(10.0, 0.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 2(0.0, 10.0): (0.74, 2.39)
```

2.a plot the circles and the optimal positions

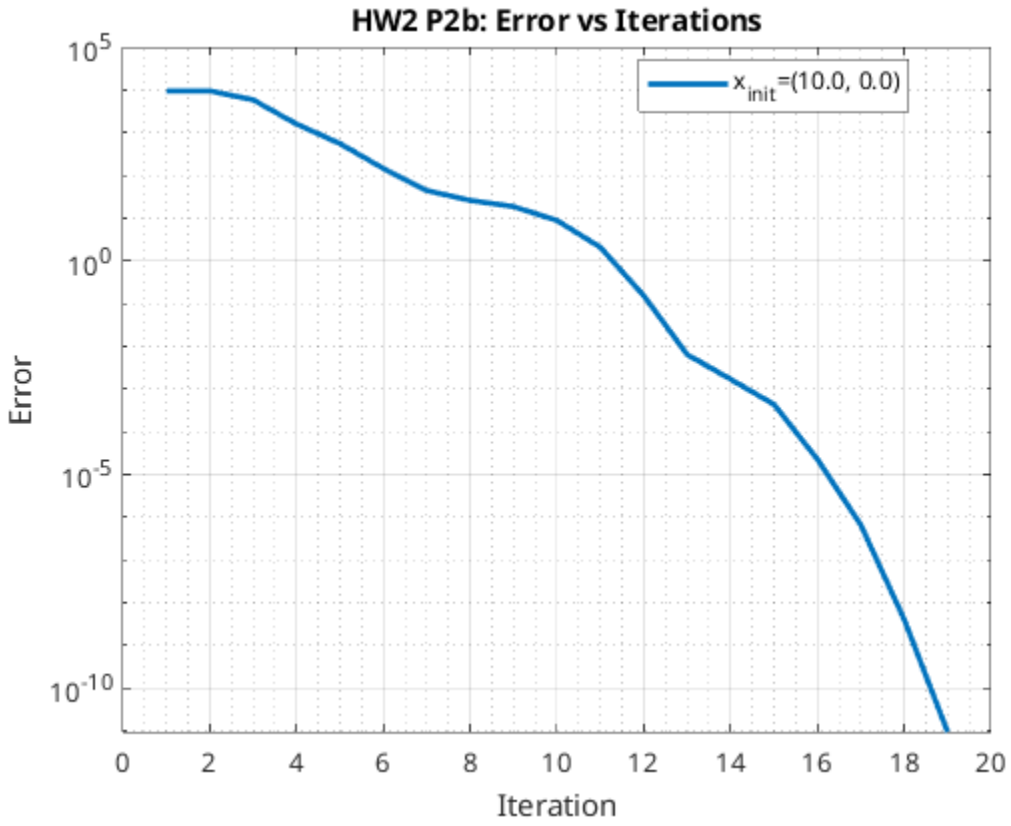
```
figure;
hold on;
th = 0:pi/50:2*pi;
x1 = R1*cos(th) + L1(1);
y1 = R1*sin(th) + L1(2);
x2 = R2*cos(th) + L2(1);
```

```
y2 = R2*sin(th) + L2(2);
plot(x1, y1, 'b', 'LineWidth', 1, 'DisplayName', 'Circle L1');
plot(x2, y2, 'r', 'LineWidth', 1, 'DisplayName', 'Circle L2');
for i = 1:size(guesses, 1)
    plot(iterates(i, 1, iters(i)), iterates(i, 2, iters(i)), '.',
'MarkerSize', 20, 'DisplayName', sprintf('x_{init}=(%.1f, %.1f)', guesses(i,
1), guesses(i, 2)));
end
axis equal;
xlabel('x'); ylabel('y');
title('HW2 P2a: Circles and Optimal Positions');
legend('Location', 'best');
grid on; grid minor;
% saveas(fig, 'p2a_circles_fixes.svg');
```



2.b plot the error vs iterations for guess 1

```
figure;
semilogy(1:iters(1), errvals(1, 1:iters(1)), 'LineWidth', 2, 'DisplayName',
sprintf('x_{init}=(%.1f, %.1f)', guesses(1, 1), guesses(1, 2)));
xlabel('Iteration'); ylabel('Error');
title('HW2 P2b: Error vs Iterations');
legend('Location', 'best');
grid on; grid minor;
% saveas(fig, 'p2b_error_vs_iterations.svg');
```



2.c plot position fixes form a grid of initial guesses

```
guess_grid = -5:2.5:10;
guesses = zeros(length(guess_grid)^2, 2);
for i = 1:length(guess_grid)
    for j = 1:length(guess_grid)
        guesses((i-1)*length(guess_grid)+j, :) = [guess_grid(i),
guess_grid(j)];
    end
end

iters = zeros(size(guesses, 1), 1);
iterates = zeros(size(guesses, 1), size(guesses, 2), 100);
errvals = zeros(size(guesses, 1), 100);
xmin = zeros(size(guesses, 1), 2);
for i = 1:size(guesses, 1)
    options = optimoptions('fminunc', 'OptimalityTolerance', 1e-12,
'OutputFcn', @(x, optimValues, state) outfun(x, optimValues, state, i));
    [x, fval] = fminunc(@(x) circle_fix_eq_2d(x, L1, L2, R1, R2),
guesses(i, :), options);
    errvals(i, 1:iters(i)) = errvals(i, 1:iters(i)) - fval;
    xmin(i, :) = x;
    fprintf('Optimal position for guess %d(%.1f, %.1f): (%.2f, %.2f)\n', i,
```

```
guesses(i, 1), guesses(i, 2), x(1), x(2));
end

% separate the guesses into groups based on thier final positions
xmin_uniq_x = uniquetol(xmin(:, 1), 0.1);
guess_groups = cell(length(xmin_uniq_x), 1);
xmin_groups = cell(length(xmin_uniq_x), 1);
for i = 1:length(xmin_uniq_x)
    guess_groups{i} = guesses(ismembertol(xmin(:, 1), xmin_uniq_x(i),
0.1), :);
    xmin_groups{i} = xmin(ismembertol(xmin(:, 1), xmin_uniq_x(i), 0.1), :);
end

group_colors = {'r', 'g', 'b'}; % make sure there are enough for the number
of groups
```

```
Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 1(-5.0, -5.0): (1.52, 1.52)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 2(-5.0, -2.5): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 3(-5.0, 0.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 4(-5.0, 2.5): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 5(-5.0, 5.0): (0.74, 2.39)

Local minimum possible.
```

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 6(-5.0, 7.5): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 7(-5.0, 10.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 8(-2.5, -5.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 9(-2.5, -2.5): (1.52, 1.52)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 10(-2.5, 0.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 11(-2.5, 2.5): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 12(-2.5, 5.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 13(-2.5, 7.5): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 14(-2.5, 10.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 15(0.0, -5.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 16(0.0, -2.5): (2.39, 0.74)

Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

Optimal position for guess 17(0.0, 0.0): (1.52, 1.52)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 18(0.0, 2.5): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 19(0.0, 5.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 20(0.0, 7.5): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

```
Optimal position for guess 21(0.0, 10.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 22(2.5, -5.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 23(2.5, -2.5): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 24(2.5, 0.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 25(2.5, 2.5): (1.52, 1.52)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 26(2.5, 5.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 27(2.5, 7.5): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 28(2.5, 10.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
```

the value of the step size tolerance.

Optimal position for guess 29(5.0, -5.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 30(5.0, -2.5): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 31(5.0, 0.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 32(5.0, 2.5): (2.39, 0.74)

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Optimal position for guess 33(5.0, 5.0): (1.52, 1.52)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 34(5.0, 7.5): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 35(5.0, 10.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 36(7.5, -5.0): (2.39, 0.74)

Local minimum possible.

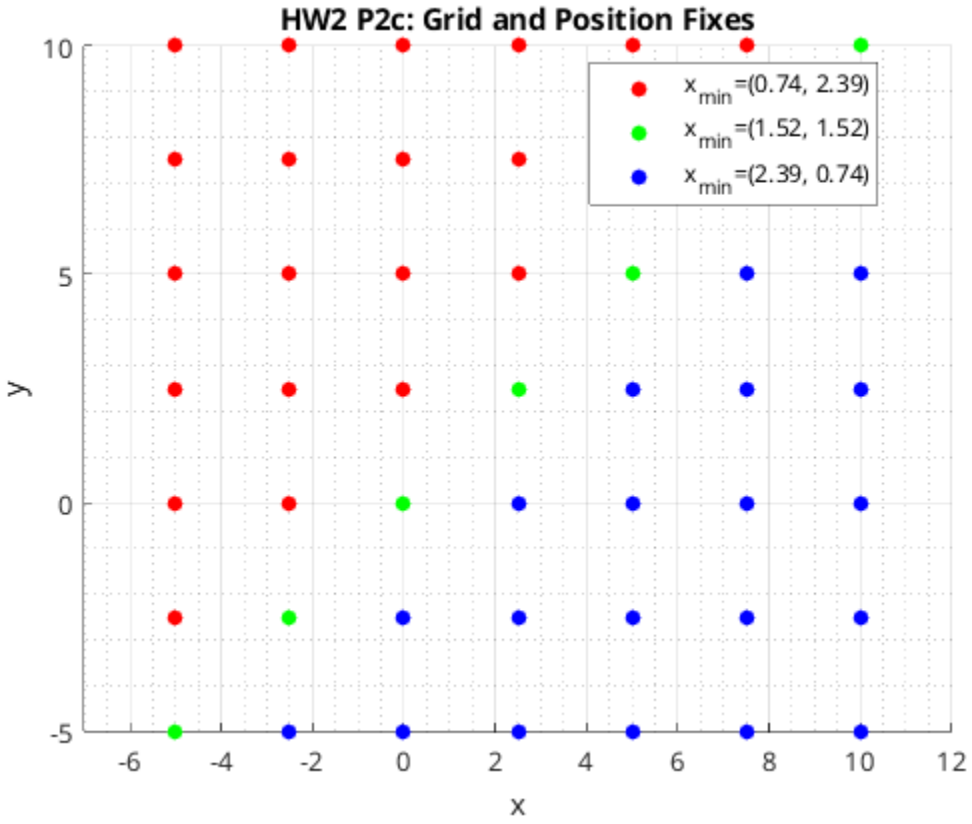
```
fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 37(7.5, -2.5): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 38(7.5, 0.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 39(7.5, 2.5): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 40(7.5, 5.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 41(7.5, 7.5): (1.52, 1.52)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 42(7.5, 10.0): (0.74, 2.39)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 43(10.0, -5.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 44(10.0, -2.5): (2.39, 0.74)
```

```
Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 45(10.0, 0.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 46(10.0, 2.5): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 47(10.0, 5.0): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 48(10.0, 7.5): (2.39, 0.74)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 49(10.0, 10.0): (1.52, 1.52)
```

plot the circles and the guess groups

```
figure;
hold on;
% th = 0:pi/50:2*pi;
% x1 = R1*cos(th) + L1(1);
% y1 = R1*sin(th) + L1(2);
% x2 = R2*cos(th) + L2(1);
% y2 = R2*sin(th) + L2(2);
% plot(x1, y1, '--c', 'LineWidth', 1, 'DisplayName', 'Circle L1');
% plot(x2, y2, '--m', 'LineWidth', 1, 'DisplayName', 'Circle L2');

for i = 1:length(guess_groups)
    scatter(guess_groups{i}(:, 1), guess_groups{i}(:, 2), 35, 'filled', ...
        'DisplayName', sprintf('x_{min}=(%.2f, %.2f)', mean(xmin_groups{i}(:,
1)), mean(xmin_groups{i}(:, 2))), ...
        'MarkerFaceColor', group_colors{i});
```

```
end

axis equal;
xlabel('x'); ylabel('y');
title('HW2 P2c: Grid and Position Fixes');
legend('Location', 'best');
grid on; grid minor;
% saveas(fig, 'p2c_grid_fixes_groups.svg');

% function to calculate the error of the position given points and ranges
function error = circle_fix_eq_2d(x, L1, L2, R1, R2)
L1_error = (x(1) - L1(1))^2 + (x(2) - L1(2))^2 - R1^2;
L2_error = (x(1) - L2(1))^2 + (x(2) - L2(2))^2 - R2^2;
error = L1_error^2 + L2_error^2;
end

% function to capture iterations and their values
function stop = outfun(x, optimValues, ~, guessnum)
global iters iterates errvals;
iters(guessnum) = iters(guessnum) + 1;
iterates(guessnum, :, iters(guessnum)) = x;
errvals(guessnum, iters(guessnum)) = optimValues.fval;
stop = false;
end
```



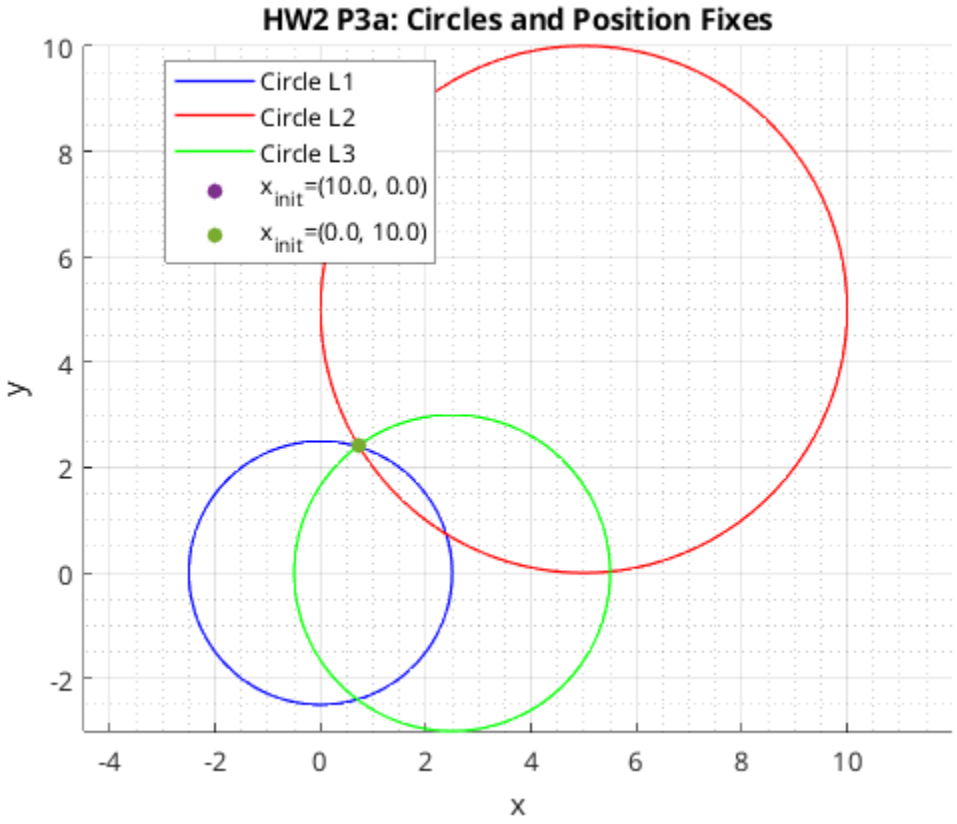
Published with MATLAB® R2023b

```
clc; clear; close all;
```

setup problem 3.a

```
L1 = [0, 0]; L2 = [5, 5]; L3 = [2.5, 0];
R1 = 2.5; R2 = 5; R3 = 3;

% setup minimization
guesses = [
    10, 0;
    0, 10;
    ];

global iters iterates errvals;
iters = zeros(size(guesses, 1), 1);
iterates = zeros(size(guesses, 1), size(guesses, 2), 100);
errvals = zeros(size(guesses, 1), 100);

for i = 1:size(guesses, 1)
    options = optimoptions('fminunc', 'OptimalityTolerance', 1e-12,
'OutputFcn', @(x, optimValues, state) outfun(x, optimValues, state, i));
    [x, fval] = fminunc(@(x) circle_fix_eq_2d(x, L1, L2, L3, R1, R2, R3),
guesses(i, :), options);
    errvals(i, 1:iters(i)) = errvals(i, 1:iters(i)) - fval;
    fprintf('Optimal position for guess %d(%.1f, %.1f): (%.2f, %.2f)\n', i,
guesses(i, 1), guesses(i, 2), x(1), x(2));
end

% %% 2.a plot the circles and the optimal positions
figure;
hold on;
th = 0:pi/50:2*pi;
x1 = R1*cos(th) + L1(1);
y1 = R1*sin(th) + L1(2);
x2 = R2*cos(th) + L2(1);
y2 = R2*sin(th) + L2(2);
x3 = R3*cos(th) + L3(1);
y3 = R3*sin(th) + L3(2);
plot(x1, y1, 'b', 'LineWidth', 1, 'DisplayName', 'Circle L1');
plot(x2, y2, 'r', 'LineWidth', 1, 'DisplayName', 'Circle L2');
plot(x3, y3, 'g', 'LineWidth', 1, 'DisplayName', 'Circle L3');
for i = 1:size(guesses, 1)
    plot(iterates(i, 1, iters(i)), iterates(i, 2, iters(i)), '.',
'MarkerSize', 20, 'DisplayName', sprintf('x_{init}=(%.1f, %.1f)', guesses(i,
1), guesses(i, 2)));
end
axis equal;
xlabel('x'); ylabel('y');
title('HW2 P3a: Circles and Position Fixes');
legend('Location', 'best');
grid on; grid minor;
% saveas(fig, 'circles_fixes.svg');
```

```
Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 1(10.0, 0.0): (0.72, 2.41)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 2(0.0, 10.0): (0.72, 2.41)
```



setup problem 3.b

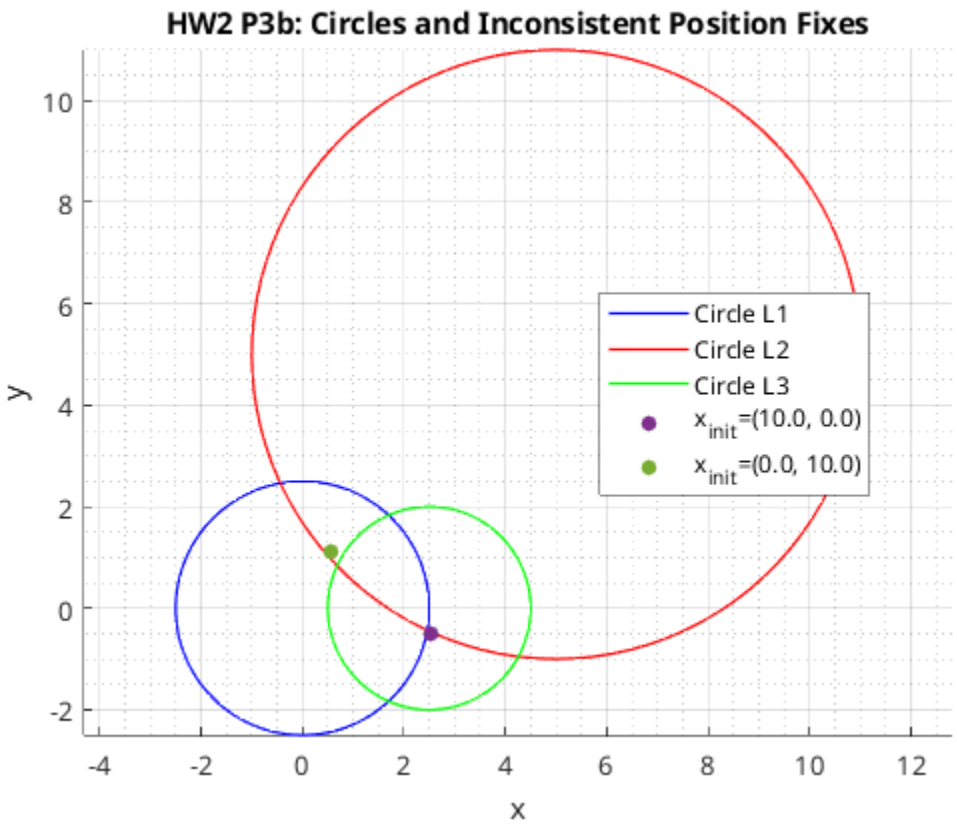
```
L1 = [0, 0]; L2 = [5, 5]; L3 = [2.5, 0];
R1 = 2.5; R2 = 6; R3 = 2;

iters = zeros(size(guesses, 1), 1);
iterates = zeros(size(guesses, 1), size(guesses, 2), 100);
errvals = zeros(size(guesses, 1), 100);

for i = 1:size(guesses, 1)
    options = optimoptions('fminunc', 'OptimalityTolerance', 1e-12,
```

```
'OutputFcn', @(x, optimValues, state) outfun(x, optimValues, state, i));
    [x, fval] = fminunc(@(x) circle_fix_eq_2d(x, L1, L2, L3, R1, R2, R3),
guesses(i, :), options);
    errvals(i, 1:iters(i)) = errvals(i, 1:iters(i)) - fval;
    fprintf('Optimal position for guess %d(%.1f, %.1f): (%.2f, %.2f)\n', i,
guesses(i, 1), guesses(i, 2), x(1), x(2));
end
```

```
Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 1(10.0, 0.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 2(0.0, 10.0): (0.54, 1.11)
```

3.b plot the circles and the optimal positions for inconsistent circles

```
figure;
hold on;
th = 0:pi/50:2*pi;
x1 = R1*cos(th) + L1(1);
y1 = R1*sin(th) + L1(2);
x2 = R2*cos(th) + L2(1);
y2 = R2*sin(th) + L2(2);
x3 = R3*cos(th) + L3(1);
y3 = R3*sin(th) + L3(2);
plot(x1, y1, 'b', 'LineWidth', 1, 'DisplayName', 'Circle L1');
plot(x2, y2, 'r', 'LineWidth', 1, 'DisplayName', 'Circle L2');
plot(x3, y3, 'g', 'LineWidth', 1, 'DisplayName', 'Circle L3');
for i = 1:size(guesses, 1)
    plot(iterates(i, 1, iters(i)), iterates(i, 2, iters(i)), '.',
'MarkerSize', 20, 'DisplayName', sprintf('x_{init}=(%.1f, %.1f)', guesses(i,
1), guesses(i, 2)));
end
axis equal;
xlabel('x'); ylabel('y');
title('HW2 P3b: Circles and Inconsistent Position Fixes');
legend('Location', 'best');
grid on; grid minor;
```



3.c plot position fixes form a grid of initial guesses

```
guess_grid = -5:2.5:10;
guesses = zeros(length(guess_grid)^2, 2);
for i = 1:length(guess_grid)
    for j = 1:length(guess_grid)
        guesses((i-1)*length(guess_grid)+j, :) = [guess_grid(i),
guess_grid(j)];
    end
end

iters = zeros(size(guesses, 1), 1);
iterates = zeros(size(guesses, 1), size(guesses, 2), 100);
errvals = zeros(size(guesses, 1), 100);
xmin = zeros(size(guesses, 1), 2);
for i = 1:size(guesses, 1)
    options = optimoptions('fminunc', 'OptimalityTolerance', 1e-12,
'OutputFcn', @(x, optimValues, state) outfun(x, optimValues, state, i));
    [x, fval] = fminunc(@(x) circle_fix_eq_2d(x, L1, L2, L3, R1, R2, R3),
guesses(i, :), options);
    errvals(i, 1:iters(i)) = errvals(i, 1:iters(i)) - fval;
    xmin(i, :) = x;
    fprintf('Optimal position for guess %d(%.1f, %.1f): (%.2f, %.2f)\n', i,
```

```
guesses(i, 1), guesses(i, 2), x(1), x(2));
end

% separate the guesses into groups based on thier final positions
xmin_uniq_x = uniquetol(xmin(:, 1), 0.1);
guess_groups = cell(length(xmin_uniq_x), 1);
xmin_groups = cell(length(xmin_uniq_x), 1);
for i = 1:length(xmin_uniq_x)
    guess_groups{i} = guesses(ismembertol(xmin(:, 1), xmin_uniq_x(i),
0.1), :);
    xmin_groups{i} = xmin(ismembertol(xmin(:, 1), xmin_uniq_x(i), 0.1), :);
end

group_colors = {'r', 'b'}; % make sure there are enough for the number of
groups
```

```
Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 1(-5.0, -5.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 2(-5.0, -2.5): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 3(-5.0, 0.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 4(-5.0, 2.5): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 5(-5.0, 5.0): (0.54, 1.11)

Local minimum possible.
```

```
fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 6(-5.0, 7.5): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 7(-5.0, 10.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 8(-2.5, -5.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 9(-2.5, -2.5): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 10(-2.5, 0.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 11(-2.5, 2.5): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 12(-2.5, 5.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 13(-2.5, 7.5): (0.54, 1.11)
```

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 14(-2.5, 10.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 15(0.0, -5.0): (2.52, -0.49)

Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

Optimal position for guess 16(0.0, -2.5): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 17(0.0, 0.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 18(0.0, 2.5): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 19(0.0, 5.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 20(0.0, 7.5): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 21(0.0, 10.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 22(2.5, -5.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 23(2.5, -2.5): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 24(2.5, 0.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 25(2.5, 2.5): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 26(2.5, 5.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Optimal position for guess 27(2.5, 7.5): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 28(2.5, 10.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than

the value of the step size tolerance.

Optimal position for guess 29(5.0, -5.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 30(5.0, -2.5): (2.52, -0.49)

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Optimal position for guess 31(5.0, 0.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 32(5.0, 2.5): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 33(5.0, 5.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 34(5.0, 7.5): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 35(5.0, 10.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 36(7.5, -5.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 37(7.5, -2.5): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 38(7.5, 0.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 39(7.5, 2.5): (2.52, -0.49)

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Optimal position for guess 40(7.5, 5.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 41(7.5, 7.5): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 42(7.5, 10.0): (0.54, 1.11)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Optimal position for guess 43(10.0, -5.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

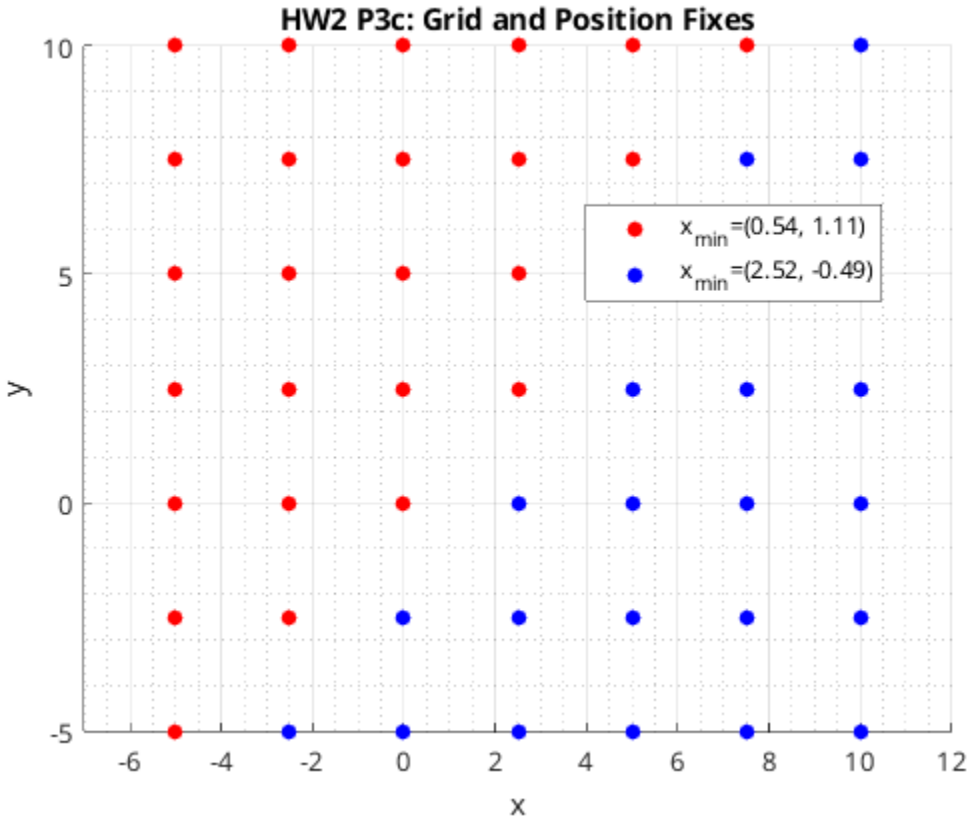
Optimal position for guess 44(10.0, -2.5): (2.52, -0.49)

```
Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 45(10.0, 0.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 46(10.0, 2.5): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 47(10.0, 5.0): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 48(10.0, 7.5): (2.52, -0.49)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Optimal position for guess 49(10.0, 10.0): (2.52, -0.49)
```

plot the circles and the guess groups

```
figure;
hold on;

for i = 1:length(guess_groups)
    scatter(guess_groups{i}(:, 1), guess_groups{i}(:, 2), 35, 'filled', ...
        'DisplayName', sprintf('x_{min}=(%.2f, %.2f)', mean(xmin_groups{i}(:,
1)), mean(xmin_groups{i}(:, 2))), ...
        'MarkerFaceColor', group_colors{i});
end

axis equal;
xlabel('x'); ylabel('y');
title('HW2 P3c: Grid and Position Fixes');
legend('Location', 'best');
grid on; grid minor;
```

```
function error = circle_fix_eq_2d(x, L1, L2, L3, R1, R2, R3)
L1_error = (x(1) - L1(1))^2 + (x(2) - L1(2))^2 - R1^2;
L2_error = (x(1) - L2(1))^2 + (x(2) - L2(2))^2 - R2^2;
L3_error = (x(1) - L3(1))^2 + (x(2) - L3(2))^2 - R3^2;
error = L1_error^2 + L2_error^2 + L3_error^2;
end

% function to capture iterations and their values
function stop = outfun(x, optimValues, ~, guessnum)
global iters iterates errvals;
iters(guessnum) = iters(guessnum) + 1;
iterates(guessnum, :, iters(guessnum)) = x;
errvals(guessnum, iters(guessnum)) = optimValues.fval;
stop = false;
end
```



Published with MATLAB® R2023b

```
clc; clear; close all;

% Setup problem 4

L1 = [0, 0]; L2 = [5, 5]; L3 = [2.5, 0];
R1 = 2.5; R2 = 5; R3 = 3;

guess = [10, 0];
rand_samples = 100;
true_pos = [0.7212, 2.4080]; % from the question
```

4.a Position fix with noise

Setup optimization

```
xmin = zeros(rand_samples, 2);
a = 2;
options = optimoptions('fminunc', 'OptimalityTolerance', 1e-12);
for i= 1:rand_samples
    R1_noisy = add_noise(R1, a);
    R2_noisy = add_noise(R2, a);
    R3_noisy = add_noise(R3, a);
    [xmin(i, :), fval] = fminunc(@(x) circle_fix_eq_2d(x, L1, L2, L3,
R1_noisy, R2_noisy, R3_noisy), guess, options);
end

% Plot circles and optimal positions
figure;
hold on;
th = 0:pi/50:2*pi;
x1 = R1*cos(th) + L1(1);
y1 = R1*sin(th) + L1(2);
x2 = R2*cos(th) + L2(1);
y2 = R2*sin(th) + L2(2);
x3 = R3*cos(th) + L3(1);
y3 = R3*sin(th) + L3(2);
plot(x1, y1, 'c', 'LineWidth', 1, 'DisplayName', 'Circle L1');
plot(x2, y2, 'm', 'LineWidth', 1, 'DisplayName', 'Circle L2');
plot(x3, y3, 'b', 'LineWidth', 1, 'DisplayName', 'Circle L3');
scatter(xmin(:, 1), xmin(:, 2), 'filled', 'DisplayName', 'Position Fixes',
'MarkerFaceColor', 'r');
scatter(true_pos(1), true_pos(2), 'filled', 'DisplayName', 'True Position
Fix', 'MarkerFaceColor', 'g');
axis equal;
xlabel('x'); ylabel('y');
title('HW2 P4a: Circles and Position Fixes with noise');
legend('Location', 'best');
grid on; grid minor;
```

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Local minimum found.

Optimization completed because the size of the gradient is less than the value of the optimality tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum found.

Optimization completed because the size of the gradient is less than the value of the optimality tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than
the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

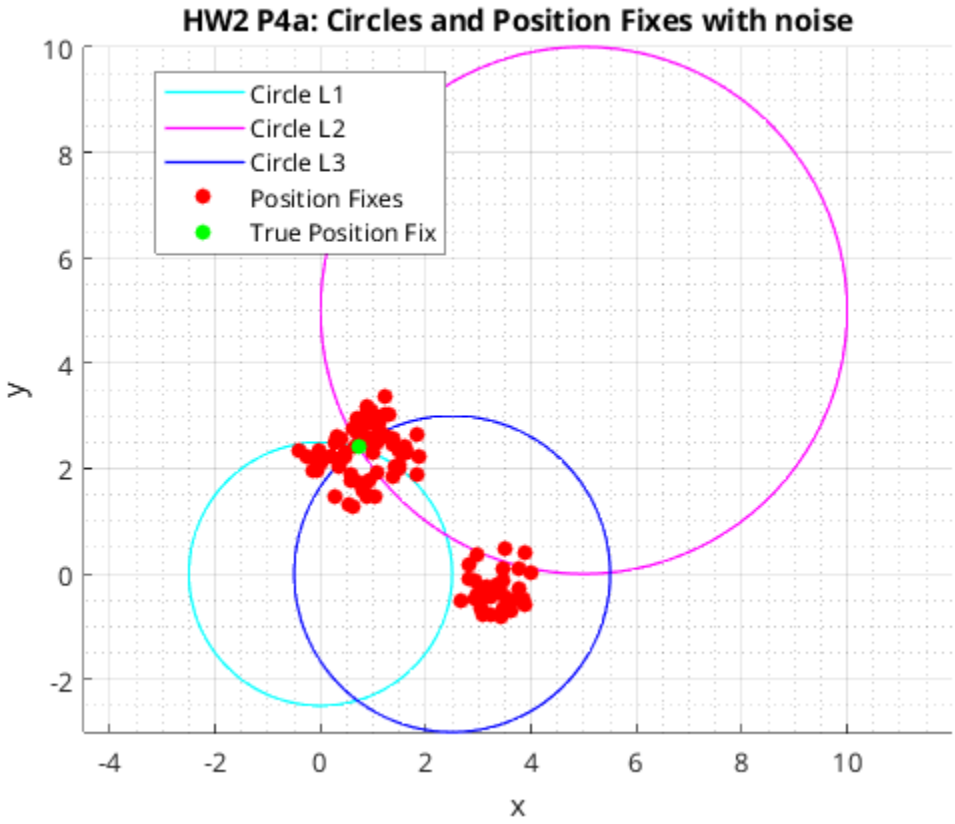
fminunc stopped because the size of the current step is less than the value of the step size tolerance.

Local minimum possible.

fminunc stopped because it cannot decrease the objective function along the current search direction.

Local minimum possible.

```
fminunc stopped because the size of the current step is less than
the value of the step size tolerance.
```



4.b Max error over a range of a

```
a_range = 0.05:0.05:10;
options = optimoptions('fminunc', 'OptimalityTolerance', 1e-12, 'Display',
'none');
max_errors = zeros(length(a_range), 1);
for i = 1:length(a_range)
    xmin = zeros(rand_samples, 2);
    a = a_range(i);
    for j= 1:rand_samples
        R1_noisy = add_noise(R1, a);
        R2_noisy = add_noise(R2, a);
        R3_noisy = add_noise(R3, a);
        [xmin(j, :), fval] = fminunc(@(x) circle_fix_eq_2d(x, L1, L2, L3,
R1_noisy, R2_noisy, R3_noisy), guess, options);
    end
    max_error = max(sqrt((xmin(:, 1) - true_pos(1)).^2 + (xmin(:, 2) -
true_pos(2)).^2));
    max_errors(i) = max_error;
    fprintf('Max error for a=%.2f: %.4f\n', a, max_error);
end
```

```
% Plot max error vs a
figure;
semilogy(a_range, max_errors, 'LineWidth', 2);
xlabel('a'); ylabel('Max Error');
title('HW2 P4b: Max Error vs a');
grid on; grid minor;


function error = circle_fix_eq_2d(x, L1, L2, L3, R1, R2, R3)
L1_error = (x(1) - L1(1))^2 + (x(2) - L1(2))^2 - R1^2;
L2_error = (x(1) - L2(1))^2 + (x(2) - L2(2))^2 - R2^2;
L3_error = (x(1) - L3(1))^2 + (x(2) - L3(2))^2 - R3^2;
error = L1_error^2 + L2_error^2 + L3_error^2;
end

function noisy_R = add_noise(R, a)
noisy_R = R + a * (rand(1) - 0.5);
end

% function to capture iterations and their values
function stop = outfun(x, optimValues, ~, guessnum)
global iters iterates errvals;
iters(guessnum) = iters(guessnum) + 1;
iterates(guessnum, :, iters(guessnum)) = x;
errvals(guessnum, iters(guessnum)) = optimValues.fval;
stop = false;
end
```

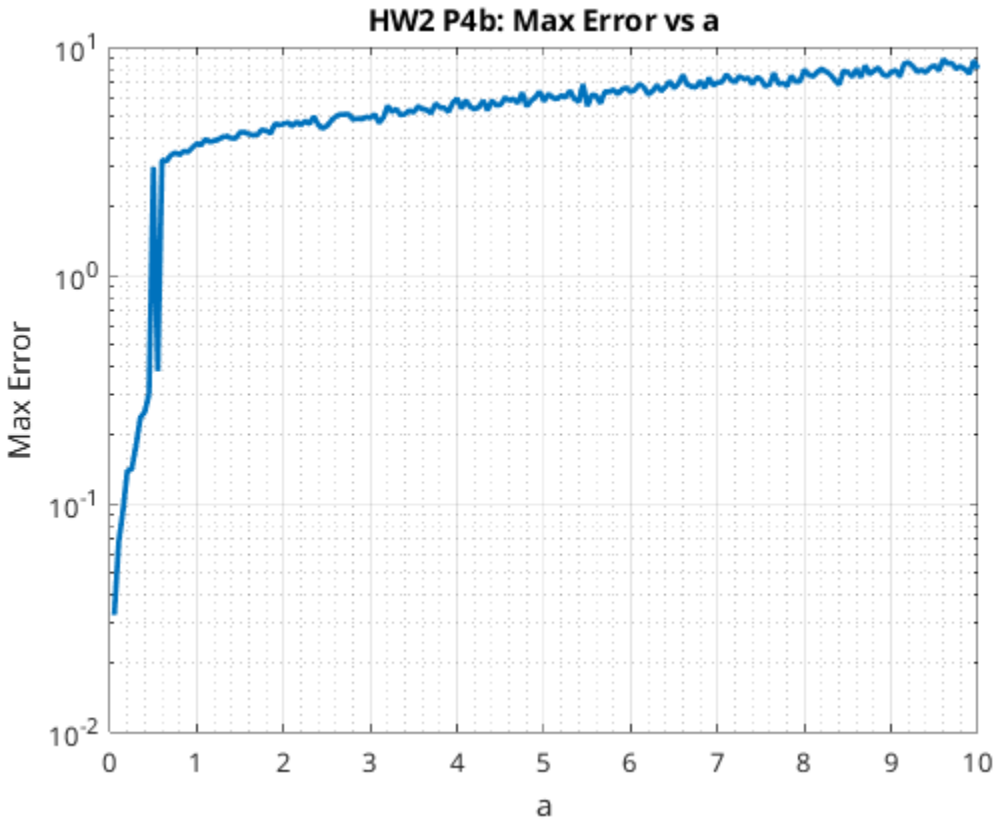
```
Max error for a=0.05: 0.0325
Max error for a=0.10: 0.0675
Max error for a=0.15: 0.0933
Max error for a=0.20: 0.1393
Max error for a=0.25: 0.1416
Max error for a=0.30: 0.1795
Max error for a=0.35: 0.2380
Max error for a=0.40: 0.2523
Max error for a=0.45: 0.3035
Max error for a=0.50: 2.9803
Max error for a=0.55: 0.3801
Max error for a=0.60: 3.2014
Max error for a=0.65: 3.1621
Max error for a=0.70: 3.3555
Max error for a=0.75: 3.4403
Max error for a=0.80: 3.3789
Max error for a=0.85: 3.4902
Max error for a=0.90: 3.4734
Max error for a=0.95: 3.6295
Max error for a=1.00: 3.7726
Max error for a=1.05: 3.7226
Max error for a=1.10: 3.9502
Max error for a=1.15: 3.8393
Max error for a=1.20: 3.8899
Max error for a=1.25: 3.9367
Max error for a=1.30: 4.0313
```

```
Max error for a=1.35: 4.0760
Max error for a=1.40: 3.9827
Max error for a=1.45: 3.9882
Max error for a=1.50: 4.2280
Max error for a=1.55: 4.2331
Max error for a=1.60: 4.1608
Max error for a=1.65: 4.1011
Max error for a=1.70: 4.1377
Max error for a=1.75: 4.3367
Max error for a=1.80: 4.2996
Max error for a=1.85: 4.1815
Max error for a=1.90: 4.6108
Max error for a=1.95: 4.5550
Max error for a=2.00: 4.6025
Max error for a=2.05: 4.6669
Max error for a=2.10: 4.5094
Max error for a=2.15: 4.6961
Max error for a=2.20: 4.5594
Max error for a=2.25: 4.7442
Max error for a=2.30: 4.6135
Max error for a=2.35: 4.9675
Max error for a=2.40: 4.5642
Max error for a=2.45: 4.4019
Max error for a=2.50: 4.5173
Max error for a=2.55: 4.7718
Max error for a=2.60: 4.9535
Max error for a=2.65: 5.0651
Max error for a=2.70: 5.0681
Max error for a=2.75: 5.0751
Max error for a=2.80: 4.8266
Max error for a=2.85: 4.8606
Max error for a=2.90: 4.8355
Max error for a=2.95: 4.9353
Max error for a=3.00: 4.8940
Max error for a=3.05: 5.0721
Max error for a=3.10: 4.6594
Max error for a=3.15: 4.9120
Max error for a=3.20: 5.5296
Max error for a=3.25: 5.1987
Max error for a=3.30: 5.3454
Max error for a=3.35: 5.0248
Max error for a=3.40: 5.0978
Max error for a=3.45: 5.2607
Max error for a=3.50: 5.1846
Max error for a=3.55: 5.5054
Max error for a=3.60: 5.3556
Max error for a=3.65: 5.3068
Max error for a=3.70: 5.1255
Max error for a=3.75: 5.6442
Max error for a=3.80: 5.4406
Max error for a=3.85: 5.4091
Max error for a=3.90: 5.1982
Max error for a=3.95: 5.7179
Max error for a=4.00: 5.9366
```

```
Max error for a=4.05: 5.3873
Max error for a=4.10: 5.8334
Max error for a=4.15: 5.4808
Max error for a=4.20: 5.3525
Max error for a=4.25: 5.4668
Max error for a=4.30: 5.8217
Max error for a=4.35: 5.2856
Max error for a=4.40: 5.8170
Max error for a=4.45: 5.5743
Max error for a=4.50: 5.5908
Max error for a=4.55: 6.0035
Max error for a=4.60: 5.8365
Max error for a=4.65: 5.9259
Max error for a=4.70: 5.6886
Max error for a=4.75: 6.3101
Max error for a=4.80: 5.5053
Max error for a=4.85: 5.8017
Max error for a=4.90: 6.0341
Max error for a=4.95: 6.3715
Max error for a=5.00: 5.7895
Max error for a=5.05: 6.2339
Max error for a=5.10: 5.9976
Max error for a=5.15: 5.9261
Max error for a=5.20: 6.1252
Max error for a=5.25: 6.0344
Max error for a=5.30: 6.4405
Max error for a=5.35: 5.9281
Max error for a=5.40: 5.7510
Max error for a=5.45: 6.8698
Max error for a=5.50: 5.5139
Max error for a=5.55: 6.1836
Max error for a=5.60: 6.1207
Max error for a=5.65: 5.6726
Max error for a=5.70: 6.3588
Max error for a=5.75: 6.3717
Max error for a=5.80: 6.4941
Max error for a=5.85: 6.2390
Max error for a=5.90: 6.5158
Max error for a=5.95: 6.6302
Max error for a=6.00: 6.3608
Max error for a=6.05: 6.5206
Max error for a=6.10: 6.8775
Max error for a=6.15: 6.6667
Max error for a=6.20: 6.2018
Max error for a=6.25: 6.3970
Max error for a=6.30: 6.8156
Max error for a=6.35: 6.4149
Max error for a=6.40: 6.7065
Max error for a=6.45: 7.0608
Max error for a=6.50: 6.6183
Max error for a=6.55: 6.9188
Max error for a=6.60: 7.5390
Max error for a=6.65: 6.8657
Max error for a=6.70: 6.7168
```

```
Max error for a=6.75: 6.6632
Max error for a=6.80: 7.0963
Max error for a=6.85: 6.6160
Max error for a=6.90: 7.3271
Max error for a=6.95: 6.8046
Max error for a=7.00: 6.9843
Max error for a=7.05: 7.0183
Max error for a=7.10: 7.5814
Max error for a=7.15: 7.2115
Max error for a=7.20: 7.0037
Max error for a=7.25: 7.4514
Max error for a=7.30: 7.2399
Max error for a=7.35: 7.3250
Max error for a=7.40: 6.8741
Max error for a=7.45: 7.3602
Max error for a=7.50: 7.2114
Max error for a=7.55: 6.7289
Max error for a=7.60: 6.8815
Max error for a=7.65: 7.6863
Max error for a=7.70: 6.8986
Max error for a=7.75: 6.9571
Max error for a=7.80: 6.7483
Max error for a=7.85: 7.4788
Max error for a=7.90: 7.0537
Max error for a=7.95: 7.0835
Max error for a=8.00: 7.9623
Max error for a=8.05: 7.5497
Max error for a=8.10: 7.4454
Max error for a=8.15: 7.7736
Max error for a=8.20: 8.0091
Max error for a=8.25: 7.8194
Max error for a=8.30: 7.5489
Max error for a=8.35: 7.1916
Max error for a=8.40: 6.8370
Max error for a=8.45: 7.8380
Max error for a=8.50: 7.7727
Max error for a=8.55: 7.4022
Max error for a=8.60: 8.0070
Max error for a=8.65: 7.3724
Max error for a=8.70: 8.2911
Max error for a=8.75: 7.6145
Max error for a=8.80: 7.8667
Max error for a=8.85: 8.1253
Max error for a=8.90: 7.6343
Max error for a=8.95: 7.4525
Max error for a=9.00: 7.7513
Max error for a=9.05: 7.9154
Max error for a=9.10: 7.4117
Max error for a=9.15: 8.4724
Max error for a=9.20: 8.5435
Max error for a=9.25: 8.2370
Max error for a=9.30: 7.8099
Max error for a=9.35: 7.9293
Max error for a=9.40: 7.7524
```

```
Max error for a=9.45: 8.1279
Max error for a=9.50: 8.2937
Max error for a=9.55: 7.8791
Max error for a=9.60: 8.8363
Max error for a=9.65: 8.4557
Max error for a=9.70: 8.5091
Max error for a=9.75: 8.0393
Max error for a=9.80: 8.2250
Max error for a=9.85: 7.9864
Max error for a=9.90: 7.6497
Max error for a=9.95: 8.7202
Max error for a=10.00: 8.0386
```



Published with MATLAB® R2023b