Chapter 38

The Gauntlet

38.1 Overview

You have mastered the Bridge of Doom and navigated Flatland. Now you face the most challenging challenge you've ever been challenged with. In the Gauntlet, you will help your robot dodge through obstacles on your way to the ultimate prize – knowledge.

Throughout this semester, you have applied many quantitative engineering analysis tools. The Gauntlet will challenge you to combine many of those tools: we'll see the return of optimization tools such as gradient descent, contour plots (of our potential fields), rotations, translations, reference frames and more!

Learning Objectives

This challenge has varying levels of difficulty. The learning objectives vary based on the level you choose.

1. Fit models of lines and circles to laser scan data (level 2 and 3).
2. Create suitable potential functions and vector fields to guide a robot to a goal while avoiding obstacles (all levels)
3. Transform between multiple coordinate systems (all levels)

38.2 The Challenge

Your challenge is to write a program to pilot your Neato through the Gauntlet, past a series of obstacles, and gently tap the mythical Ball of Benevolence (BoB).

For this challenge, you will need to detect obstacles using your robot's lidar sensor, which can detect obstacles within a 5m range. The Neato starts at the origin facing along the $+\hat{\imath}$ direction (x-axis).

This final QEA challenge has a hierarchy of missions, which get progressively more difficult. You may choose any combination of challenges, but we suggest aiming for at least some version of level 2. Here are the conditions for the missions:

Level 1 You may assume a known location for the BoB, you may not use prior knowledge of the walls and obstacles. Instead, you must use your LIDAR sensors to detect and avoid obstacles. You can pre-plan your path based on previous LIDAR scans or dynamically update the path as the robot moves.

Level 2 You tackle the same challenge as level 1, but you are only given the radius of the BoB, not the coordinates (i.e., you must use LIDAR scans to identify and locate the BoB). You can plan your entire path based on your initial LIDAR scan or dynamically update your path as you go. To identify the

BoB, you will need to differentiate the distinctive, circular shape of the goal from the linear shape of the obstacles. If you choose this mission, we have some materials on circle fitting to provide to you.

Level 3 Do everything in level 2, and apply your Gauntlet algorithm... to a new map! Show that your code can generalize by placing your Neato in a new environment and seeing if it can reach the BoB. To earn extra credit (see below), the map must be sufficiently different (and at least of comparable difficulty).

You may also choose to extend any of the above Gauntlet missions by applying or creating another goal-seeking/obstacle-avoiding algorithm, trying to reach the target as soon as possible, or adding another challenge that is exciting to you!

Note: that part of the challenge will be learning about the concept of potential fields for robot navigation and (optionally) circle fitting. You should make sure to use the potential field method for navigation.

38.3 Deliverables (due 5/2)

We hope you won't think of these deliverables as only items to check off, but rather as scaffolding to get maximum learning from the task at hand. By creating these intermediate deliverables, you will be able to more easily debug your code as well as your understanding of the algorithms you are utilizing.

Prepare a video and a **brief** writeup of your work on this challenge. Your final writeup should center around critical information and informative figures. Plots should be clearly labeled and readable (decent size fonts!). The main report should not include any code; feel free to include code in an appendix.

Note: The report is out of 20 pts. Difficulty levels above 1 are eligible for extra credit.

The report can be as short as you want, but it must contain the following components:

Overview

1. A short introduction stating the goal and a brief summary of the strategy you used to solve the challenge including any relevant equations. [2 pts]

Mapping and Path Planning: These are the first steps in the challenge - we recommend aiming to have these done early!

1. A map of the Gauntlet using LIDAR scan data. The plot should have appropriate labels and units. Please see details of this map for each level of challenge. You should have this from homework 8, so hopefully you just have to clean it up a bit.
   (a) **Level 1 (min)** An ensemble of data points from laser scans collected at various positions around the Gauntlet and translated/rotated to the global coordinate frame, as in your previous homework. The points from each scan should be shown in a different color, and all walls, obstacles, and the BoB should be captured by the scans. [3 pts]
   (b) **Level 2-3** An additional fit for the BoB. [+1 pt]
2. An equation and a graphical representation (contour and/or 3-D plot) of the potential field you developed for navigating the gauntlet.
   (a) **Level 1** The potential field should be generated based on the discrete lidar data. [3 pts]
   (b) **Level 2/3** The potential field is generated from LIDAR as well as the automated identification of the BoB in the scan data. [+1 pt]
3. A quiver plot of the gradient of your potential field. You can use the quiver() function in Matlab, but you will need to reduce your indices (i.e. as in previous challenges, you should not try to plot every single vector. Alternatively, you can also use the Matlab streamslice() function. [1 pt]
4. A path of gradient descent from the starting point to the BoB over a contour plot. [2 pt]

Navigating the Gauntlet:

1. Some experimental data that shows, quantitatively, how well your system worked, plus a few sentences of explanation [1 pt]. At minimum you should have:
   (a) A plot with: a map of the relevant features of the Gauntlet based on a LIDAR scan, the intended path of gradient descent, and the path your robot took, calculated from the wheel encoder data. This should be a legible plot with axis labels, a legend, and a caption...the works. [3 pts]
   (b) The time it took your robot to get to the BoB (for whichever mission you choose) and the distance it traveled. [1 pt]
2. A link to a video of your robot in action. Note: please let us know if you do NOT want to share your video with the class [2 pts].
3. Should you choose to accept Level 3 (or possibly other challenges) - another bonus point is possible!

With your writeup, you should also turn in your (commented and readable!) code. This can be as link (preferred) e.g. to a github repository or matlab drive, or copied into the report at the end as an appendix. *It should NOT be included in the main text of the report.* [2 pt]

38.4 Potential Fields for Robot Navigation

The material in this section will be useful to solving the Gauntlet challenge. It's up to you how you engage with the problems below.

In an earlier assignment we used the method of steepest ascent to navigate the NEATO to the top of a "mountain", which we defined using a scalar field $H(x, y)$.

In the Gauntlet challenge we are going to use targets (ball of benevolence) and objects (walls, boxes, etc) to create an artificial mountain landscape, and we will navigate the NEATO using the **method of steepest descent** so as to avoid the obstacles and hone in on the target. (We use steepest descent in order to connect this with ideas from physics.)

To accomplish this task, we are going to introduce a particular scalar field $V(x, y)$ which can be used to create useful mountain landscapes. It is known as a **potential** field.

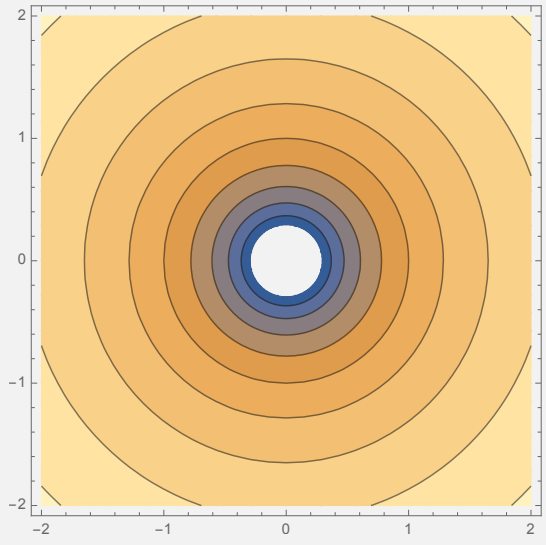
Exercise 38.1

Let $V(x, y) = \ln \sqrt{x^2 + y^2}$. Note that V is not defined at the origin!

1. What is V in polar coordinates? (note: this question is just here to make you realize that it's actually a pretty function, and to make it easier to visualize)
2. Create a contour plot of V.
3. Determine the gradient field defined by ∇V, and visualize it.
4. How would a NEATO performing gradient descent behave if you put it on a Flatland defined by V and started it at $(1, 2)$?

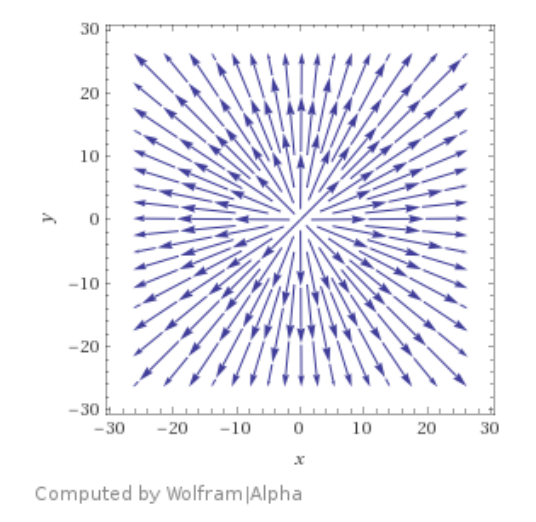
Solution 38.1

1. In polar coordinates $V(r, \theta) = \ln(r)$, so the contours of V should be circles.

2. 

3. The gradient is given by

$$\nabla V = \frac{x}{x^2+y^2}\hat{\imath} + \frac{y}{x^2+y^2}\hat{\jmath}$$



This is not actually the gradient field - we've produced instead the streamlines using the "streamplot" function in WolframAlpha. The streamlines are the path that the Neato would take if it continuously followed the direction of steepest ascent.

4. It would move toward the origin (gradient descent)!

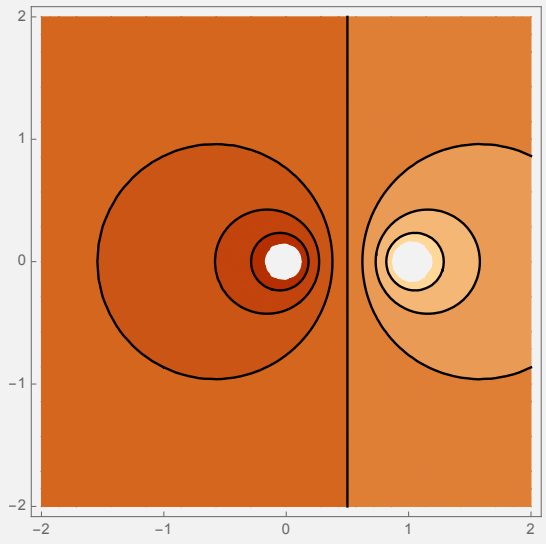
Exercise 38.2

Now consider $V(x,y) = \ln\sqrt{x^2+y^2} - \ln\sqrt{(x-1)^2+y^2}$. Notice that there is a strong "trough" at $(0,0)$ and a strong "peak" at $(1,0)$. The sign in front of the logarithm determines whether it is a peak or a trough.

1. Create a contour plot of V.

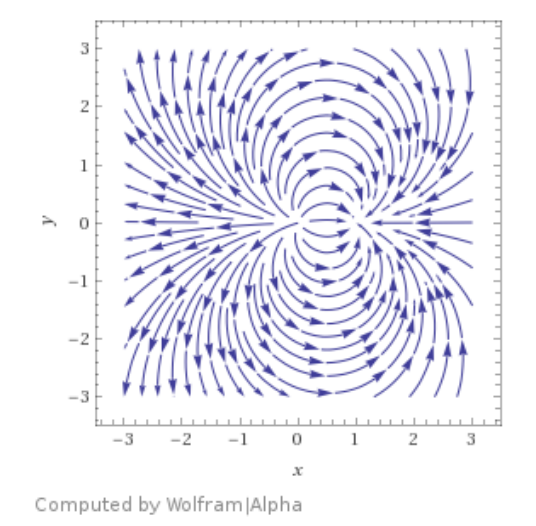
2. Determine the gradient field defined by ∇V, and visualize it.
3. How would a NEATO performing gradient descent behave if you put it on a Flatland that looked like V and started it at $(1, 2)$?

Solution 38.2

1. 

2. The gradient field is given by

$$\nabla V = \frac{x}{x^2+y^2}\hat{\imath} + \frac{y}{x^2+y^2}\hat{\jmath} - \left(\frac{(x-1)}{(x-1)^2+y^2}\hat{\imath} + \frac{y}{(x-1)^2+y^2}\hat{\jmath}\right)$$



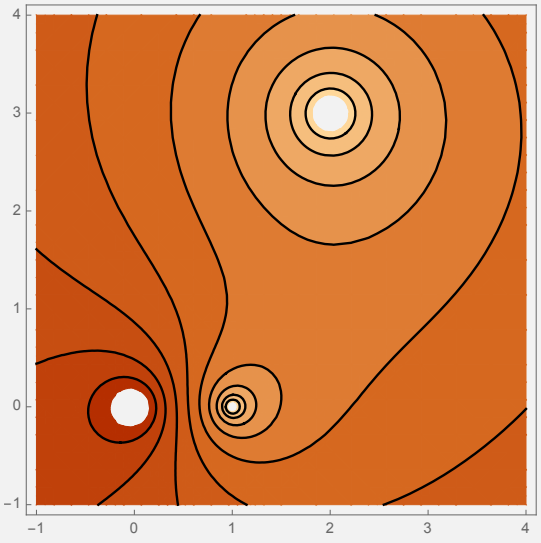
3. It would move away from $(1, 2)$ and eventually find its way to $(0, 0)$.

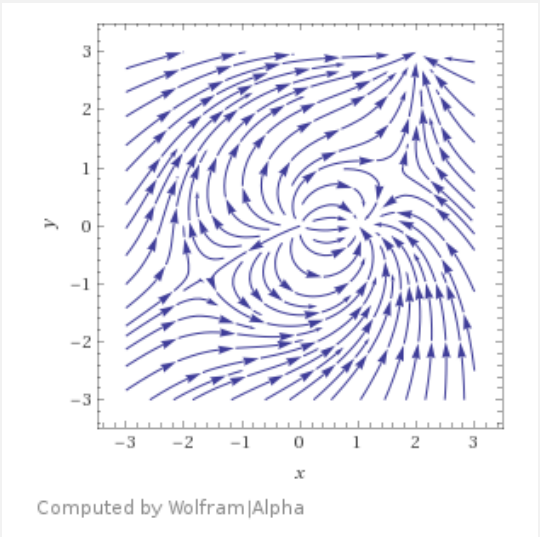
Exercise 38.3

In the last question, a NEATO would be attracted by the trough at $(0, 0)$ but repelled by the peak at $(1, 0)$. We will refer to the point at $(0, 0)$ as a "sink" and the point at $(1, 0)$ as a "source".

1. Define a scalar field $V(x, y)$ that has a sink at $(0, 0)$, and sources at $(1, 0)$ and $(2, 3)$.
2. Visualize the scalar field V and the gradient field ∇V.
3. How would a NEATO performing gradient descent behave if you put it on a Flatland that looked like V and started it at $(1, 2)$?

Solution 38.3

1. $V(x, y) = \ln \sqrt{x^2 + y^2} - \ln \sqrt{(x-1)^2 + y^2} - \ln \sqrt{(x-2)^2 + (y-3)^2}$
2. 

3. 

4. It would move away from $(1, 2)$ and eventually find its way to $(0, 0)$.

Exercise 38.4

1. What is the potential field and gradient vector field for a **source** or **sink** at the location (a,b).

Solution 38.4

1. Include a negative in front to turn a sink into a source.
2. The potential field for a sink at (a,b) is

$$V(x,y) = \ln \sqrt{(x-a)^2 + (y-b)^2}$$

which has the following gradient vector field

$$\nabla V = \frac{x-a}{(x-a)^2 + (y-b)^2}\hat{\imath} + \frac{y-b}{(x-a)^2 + (y-b)^2}\hat{\jmath}$$

Exercise 38.5

1. Sketch the contour levels for a **sink** at the origin.
2. Read the following MATLAB script and make sure you understand the purpose of every statement. Then implement it and compare the result with your sketch.

```
[x,y]=meshgrid(-3:0.05:3,-3:0.05:3);
v = log(sqrt(x.^2+y.^2));
contour(x,y,v,'k','ShowText','On')
axis equal
```

3. Write down the potential field for a **source** at $(1,2)$.
4. Sketch the contour levels with a **sink** at the origin and a **source** at (1,2).
5. Modify the MATLAB script above to confirm your prediction.

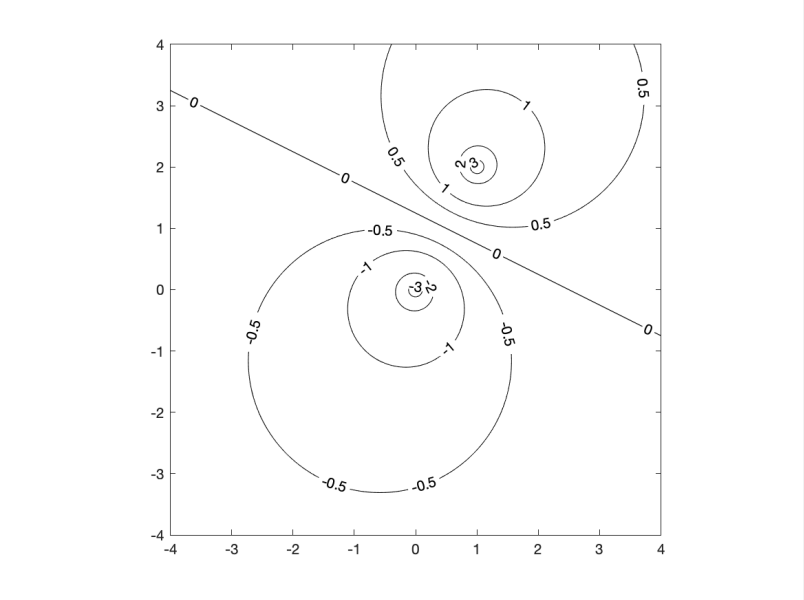
Solution 38.5

1. Circles surrounding the origin.
2.
```
%build a mesh
[x,y]=meshgrid(-3:0.05:3,-3:0.05:3);
%define the potential at all mesh points
v = log(sqrt(x.^2+y.^2));
%plot contours with levels marked
contour(x,y,v,'k','ShowText','On')
%so that circles look like circles
axis equal
```

3.

$$V(x,y) = -\ln \sqrt{(x-1)^2 + (y-2)^2}$$

4. Circles when close to the sink or source. As you move away there are still circles but their center shifts. There is a line of constant potential that bisects the line that connects the sink and the source.

5. 

Exercise 38.6

1. Sketch the gradient vector field for a **sink** at the origin.
2. Read the following MATLAB script and make sure you understand the purpose of every statement. Then implement it and compare the result with your sketch.

```
[x,y]=meshgrid(-3:0.3:3,-3:0.3:3);
dv_dx = x./(x.^2+y.^2);
dv_dy = y./(x.^2+y.^2);
quiver(x,y,dv_dx,dv_dy)
axis equal
```

3. Write down the gradient vector field for a **source** at $(1, 2)$.
4. Sketch the gradient vector field with a **sink** at the origin and a **source** at $(1, 2)$.
5. Modify the MATLAB script above to confirm your prediction.

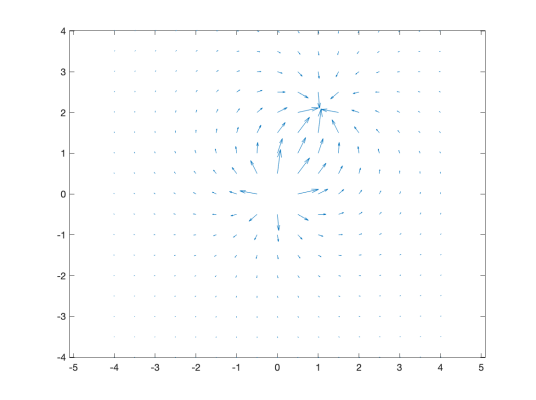
Solution 38.6

1. Arrows pointing radially outward.
2.
```
%create the mesh, less points so vectors visible
[x,y]=meshgrid(-3:0.3:3,-3:0.3:3);
%define the components of the gradient
dv_dx = x./(x.^2+y.^2);
```

```
dv_dy = y./(x.^2+y.^2);
%alternatively, use [dv_dx dv_dy]=gradient(v);
%draw the arrows at the mesh points
quiver(x,y,dv_dx,dv_dy)
%so circles look like circles
axis equal
```

3. $$\nabla V = -\frac{x-1}{(x-1)^2+(y-2)^2}\hat{\imath} - \frac{y-2}{(x-1)^2+(y-2)^2}$$

4. Arrows points away from the sink at the origin, and converging on the source at (1,2).

5. 

Exercise 38.7

1. Consider a source "circle" of radius 1 centered at the origin. Sketch the contours of the potential field and the gradient vector field (outside of the circle).

2. Read the following MATLAB script and make sure you understand the purpose of every statement. Then implement it and compare the result with the contours on your sketch.

```
[x,y]=meshgrid(-3:0.01:3,-3:0.01:3);
v = 0;
for theta = 0:0.1:2*pi
    a = cos(theta);
    b = sin(theta);
    v = v - log(sqrt((x-a).^2 + (y-b).^2));
end
contour(x,y,v,'k','ShowText','On')
axis equal
```

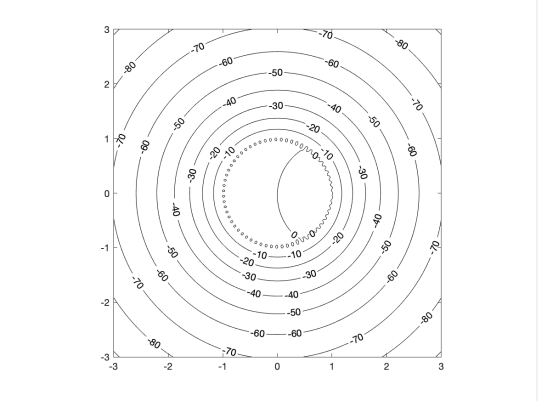
3. How would you compute and visualize the gradient vector field for this circle source? Go ahead and do so.

4. How would you edit the script to make the source be a circle of radius 2, centered at (-1,2)? Go ahead and do so.

Solution 38.7

1. The contours will be circles, but they will decrease more slowly than the case of a single source at the origin. The gradient vectors will be radiating inwards.

2. ```
   %create a mesh
   [x,y]=meshgrid(-3:0.01:3,-3:0.01:3);
   %initialize the potential to zero
   v = 0;
   %loop through angles from 0 to 2 pi
   for theta = 0:0.1:2*pi
   %sources have coordinates (a,b)
       a = cos(theta);
       b = sin(theta);
       v = v - log(sqrt((x-a).^2 + (y-b).^2));
   end
   %visualize the contours
   contour(x,y,v,'k','ShowText','On')
   %make circles look like circles
   axis equal
   ```



3. We would sum up the gradient field from all the sources on the circle.

4. We would change the source coordinates as follows

   ```
   a = -1+2.*cos(theta);
   b = +2+2.*sin(theta);
   ```

Exercise 38.8

Now that you've thought about potential fields for sources and sinks (including lines and circles), we'd like you to conceptually think about how this might apply to helping a NEATO navigate the Gauntlet.

1. Review the Gauntlet—it can be found earlier in Chapter 38.

2. Thinking just about the walls, sketch the potential field and the gradient vector field if we treat the walls as a collection of point sources.

3. Thinking just about the ball of benevolence, sketch the potential field and the gradient vector
   ```

field if we treat the BoB as a circle sink.

4. Can you imagine how you would add all of the above sketches to build a sketch for the entire Gauntlet?
5. How would you use the computational tools from the earlier exercises to determine the actual potential field and gradient vector field for the Gauntlet?

Solution 38.8

38.5 Circle Fitting

The material in this section will be useful for solving level 2 and 3 of the Gauntlet Challenge. It's up to you how you engage with the problems below.

If you plan to identify and locate the Ball of Benevolence in The Gauntlet challenge, you will need to be able to find and fit a circle. These steps will guide you through the process of creating a circle-fitting algorithm of your own design.

Exercise 38.9

Assume that you have collected a set of M data points (x_i, y_i). There are multiple approaches to fitting a circle to this data, but we would like you to utilize the tools you have learned in QEA, specifically linear regression.

1. What are the different ways of describing a circle? Think about explicit, implicit, and parametric representations. Don't forget that there are various ways of describing a circle using each of these representations.

2. We would like you to set up the circle fitting problem using linear regression techniques from linear algebra; i.e., an over-constrained system of linear equations where $\mathbf{Aw} = \mathbf{b}$. Denote the center of the circle by (x_c, y_c) and the radius by r. A point (x, y) is on the circle if $(x - x_c)^2 + (y - y_c)^2 = r$. We can therefore formulate the fitting problem (with M points) as

$$\min_{x_c,\, y_c,\, r} \left\{ \sum_{i=1}^{M} \left[(x_i - x_c)^2 + (y_i - y_c)^2 - r^2 \right]^2 \right\}.$$

   If we do a change of variable $z = x_c^2 + y_c^2 - r^2$, then we can write the above problem as a linear least squares problem

$$\min_{\mathbf{w}} ||\mathbf{Aw} - \mathbf{b}||^2,$$

   where $\mathbf{w} = \begin{bmatrix} x_c \\ y_c \\ z \end{bmatrix}$. Find $\mathbf{A}$ and $\mathbf{b}$ in this least squares formulation. They should only depend on the input data points $[\mathbf{x}, \mathbf{y}]$, and not on any of the unkown variables $x_c,\ y_c,\ r$, or z. What are the dimensions of $\mathbf{A}$ and $\mathbf{b}$?

3. How would you solve this system?

4. Now imagine that you wanted to fit a circle to the subset of points that define a circle. Describe the algorithm you would use, assuming that you knew the radius of the circle your were fitting (a la BoB).

Solution 38.9

1. Implicit:

$$(x - x_c)^2 + (y - y_c)^2 = r^2$$

   Parametric:

$$x = x_c + r\cos t$$

$$y = y_c + r\sin t$$

   General Form:

$$x^2 + y^2 + Ax + By + C = 0$$

2. After the change of variable, the minimization problem becomes:

$$\begin{aligned}
&\min_{x_c,\, y_c,\, z} \left\{ \sum_{i=1}^{M} \left[-2x_i x_c - 2y_i y_c + z + x_i^2 + y_i^2 \right]^2 \right\} \\
&= \min_{x_c,\, y_c,\, z} \left\{ \sum_{i=1}^{M} \left[2x_i x_c + 2y_i y_c - z - x_i^2 - y_i^2 \right]^2 \right\} \\
&= \min_{\mathbf{w}} ||\mathbf{A}\mathbf{w} - \mathbf{b}||^2,
\end{aligned}$$

where $\mathbf{w} = \begin{bmatrix} x_c \\ y_c \\ z \end{bmatrix}$, the i^{th} entry of $\mathbf{b}$ (an $M \times 1$ vector) is $x_i^2 + y_i^2$, and the $M \times 3$ matrix $\mathbf{A}$ has $2\mathbf{x}$ in the first column, $2\mathbf{y}$ in the second column, and -$\mathbf{1}$ in the third column.

3. In MATLAB, we would use the backslash operator: $\mathbf{w} = \mathbf{A}\backslash\mathbf{b}$, which solves the over-constrained system using linear regression.

4. A circle fitting algorithm could go like this:

   - Select three consecutive points from the laser scan. At minimum you need two points to fit the circle, but with only two points you have two solutions for the location of the circle center. Selecting three points removes this complication
   - Using the three points, perform linear regression as outlined above to find the parameters $x_c,\ y_c,\ z$ of the candidate circle
   - Find the radius $r = \sqrt{x_c^2 + y_c^2 - z}$
   - Find the distance d of each point (x_i, y_i) in the laser scan from the candidate circle edge as: $d = \|\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} - r\|$
   - Determine for each point if it is "close enough" to the fitted circle (we call these inliers).
   - If a candidate circle has the most inliers **and** has a radius close of the radius of the BoB, keep the candidate circle as the current best fit. If not, try again
   - Repeat

Exercise 38.10

At this point, you have a couple of approaches in mind, and you've thought through the details of each. Let's test your ideas now by implementing in MATLAB.

1. Write pseudocode for finding the best-fit circle using one of your methods.

2. Using the following code, generate some test points that lie on a circle of radius 2. Rather than generating points on the entire circle, let's generate points on an arc of 120 degrees to mimic the extent of the data we might get from the Neato. Start without any noise.

```
function [x y rs thetas] = makeCircle(r, anglemin, anglemax, noiseLevel)
    thetas = linspace(anglemin, anglemax, (anglemax-anglemin+1))';
    rs = r*ones(size(thetas));
    rs = rs + randn(size(rs))*noiseLevel;
    x = cosd(thetas).*rs;
    y = sind(thetas).*rs;
end
```

3. Test your methods on this data. How good is your best-fit circle? Do your results depend on which method you use, or how you initialize the method?
4. Now incorporate a little noise into your test data set, perturbing the radius of each of your data points. How does this affect your results? Make sure to try your code a couple of times to get a representative picture of how well it works on different noisy data samples.
5. Now implement the pseudocode you generated above into a working circle fitting algorithm in MATLAB. Test your code on the file playpensample.mat, which has one instance of the BoB with r=0.1329m.

Solution 38.10

Your plots should look something like these.