CS1112 Fall 2014 Project 2 due Thursday 9/18 at 11pm

You must work either on your own or with one partner. If you work with a partner you must first register as a group in CMS and then submit your work as a group. *Adhere to the Code of Academic Integrity.* For a group, "you" below refers to "your group." You may discuss background issues and general strategies with others and seek help from the course staff, but the work that you submit must be your own. In particular, you may discuss general ideas with others but you may not work out the detailed solutions with others. It is not OK for you to see or hear another student's code and it is certainly not OK to copy code from another person or from published/Internet sources. If you feel that you cannot complete the assignment on you own, seek help from the course staff.

Objectives

Completing this project will solidify your understanding of Chapters 2 and 3 in *Insight* (`for`-loops, `while`-loops, nested loops). In problem 2 you will further explore MATLAB graphics.

Ground Rule

Do not use arrays or the `break` command in this project. In fact, do not use the `break` command in this course.

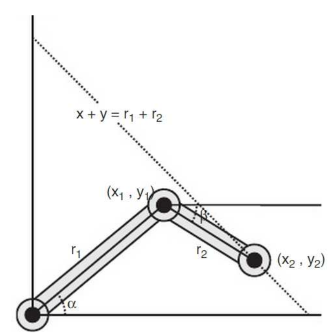
1 Verifier

Complete Problem **P2.1.7** in *Insight* (page 35). The problem asks you to verify the given inequalities for $n = 1, 2, \ldots, 100$. Your script should work through all 100 n values but produce one line of output for each value of n that is a multiple of 10. Each line of output should contain four values: n and the three values that are computed for verification for that n. Use the function `fprintf` to produce output that lines up neatly in a "table format", i.e., the numbers should line up column-wise. *Sanity check:* there are 10 rows of numbers in your output.

Your script should end by displaying one of these two messages: "the inequalities hold for all n values" or "the inequalities do not hold for at least one n value."

Save your script in a file `verify.m` and submit it on CMS.

2 Motion of a robotic arm



Imagine programming a two-hinged robotic arm so that its "end-effector"—the furthest point of the arm which can be fitted with a tool such as a gripper or a paint sprayer—moves in a straight line. The figure on the right illustrates the problem. The computation of the sequence of angles α and β that will guide the end-effector along the straight line $x + y = r_1 + r_2$ is a non-trivial task! We will solve a simplified problem where α is fixed and we determine the value(s) of β that will place the end-effector at some point on the line

$$x + y = r_1 + r_2 \tag{1}$$

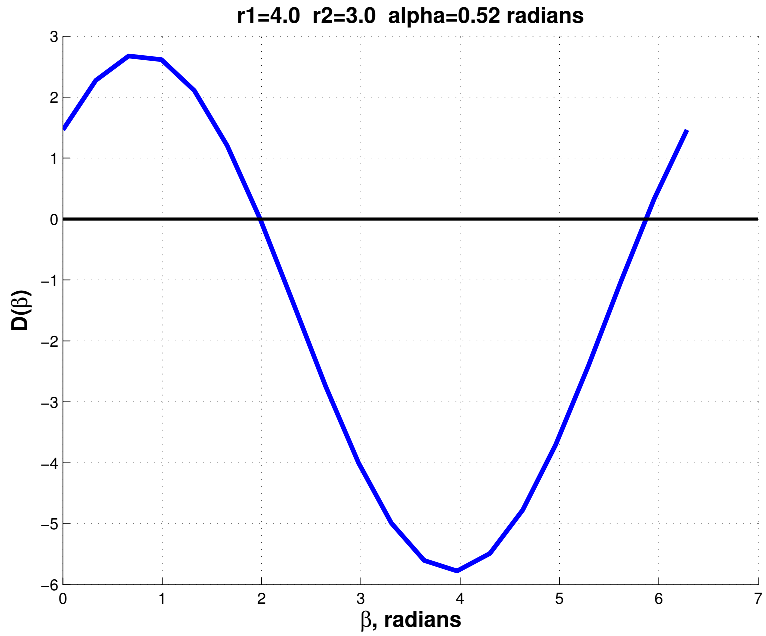
The position of the end-effector, (x_2, y_2), is expressed as

$$x_2 = r_1 \cos\alpha + r_2 \cos\beta \tag{2}$$
$$y_2 = r_1 \sin\alpha + r_2 \sin\beta \tag{3}$$

Combining Equations (1)–(3) gives the equation $D(\beta)$, the difference between the end-effector and the target line. We need to solve for $D(\beta) = 0$:

$$D(\beta) = r_1 \cos\alpha + r_2 \cos\beta + r_1 \sin\alpha + r_2 \sin\beta - (r_1 + r_2) \tag{4}$$

To begin the solution process, first let's see what $D(\beta)$ looks like given some example values of r_1, r_2, and α:



There are two zero crossings because there are two β values that can place the end-effector on the target line. (Take a closer look at the robot arm diagram on the previous page to confirm this.)

Instead of "eye-balling" the solutions off a graph, you will write a program to determine *one* of possibly two values of β that will place the end-effector on the target line given r_1, r_2, and α, where $0 \leq \alpha \leq \pi/2$. **Use this approach:** Calculate $D(\beta = 0)$. Then repeatedly increase β by some amount and calculate $D(\beta)$ until a sign change occurs. This stopping point of β is a solution. Along the way, your program should plot the calculated D against β.

Download the file `robotArm.m` from the *Projects* page. Read and run it. Since the program is incomplete, the output includes only a figure window with axis labels and the given values of r_1, r_2, and α printed to the Command Window. You will complete the program to determine and print to the Command Window a value of *beta* that would place the end-effector on the target line. You must use the approach described above. Additionally, the program should plot the calculated values of $D(\beta)$.

Additional specifications, considerations, and hints

- A hint for detecting a sign change: If $ab \leq 0$, then a and b have different signs or a and/or b is zero.
- Choose an appropriate step size for incrementing β. Some amount of error is expected, so use your best judgment—and experimentation—to come up with a reasonable value. The graph that your program produces will help you *see* when a step size is too big.
- Add an `fprintf` statement to print your solution, β, to the Command Window
- To plot a point, use the `plot` function. For example, `plot(x,y,'b.')` plots a blue dot. You can use any color and marker you like.
- Add a title to the figure to display the values of r_1, r_2, and α. Review Project 1 if you need a reminder on how to use the `title` and `sprintf` functions.
- The provided code specifies a set of values for r_1, r_2, and α, but your program should still work if the values are changed! You can expect that $0 \leq \alpha \leq \pi/2$.
- If $\alpha = 0$ or $pi/2$, the answer is simple and the iterative approach discussed above is not needed. So in those two cases, there would be just a single point on the graph.

- Test your program using various reasonable values of r_1, r_2, and α.
- Before you submit your code, change all the parameter values back to the original values given: $r_1 = 4$, $r_2 = 3$, and $\alpha = \pi/6$

Submit your completed script file `robotArm.m` on CMS.

3 Diamond is for~~ever~~-loop

Complete Problem **P3.1.8** in *Insight* (page 55). Recall that `fprintf(' ')` prints one space character while `fprintf('\n')` starts a new line. Save your script as `printDiamond.m` and submit it on CMS.