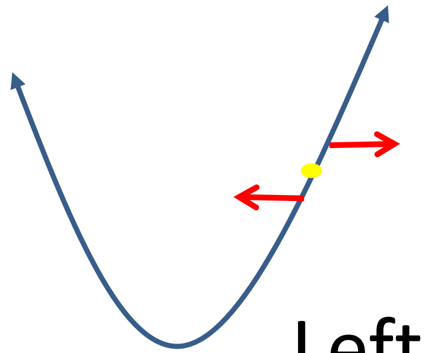
Minimizing in 3D

Methods 1 and 2:

Brute Force, Brute Intelligence

The Premise

In single-variable optimization, there are only two directions you can go from a single point: left or right.

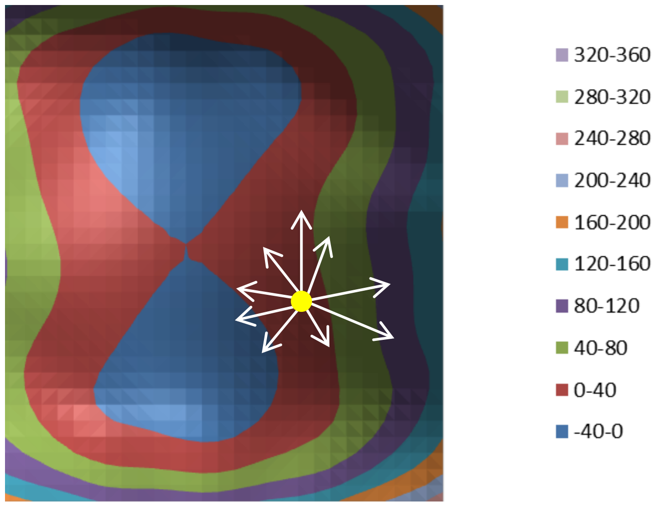


Right: this function is increasing

Left: this function is decreasing

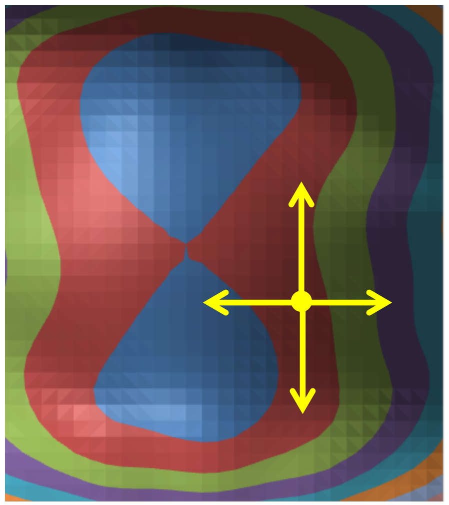
The Premise

In two-variable problems, there are an infinite number of directions you can go from a single point.



The Premise

In two-variable problems, there are an infinite number of directions you can go from a single point.



Fortunately, it’s sufficient to just test 4.

Practice Problem 1

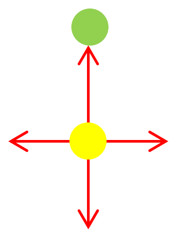
Consider the function $f(x_1, x_2) = (x_1)^2 + 5(x_2)^2$.

a) Enter this as a one-line function f(a, b). Use it to evaluate the points (3, 5), (2, 0) and (-4, -1).

b) Create a program that evaluates a given function $f(x_1, x_2)$ at (x_1, x_2), $(x_1 + 0.1, x_2)$, $(x_1 - 0.1, x_2)$, $(x_1, x_2 + 0.1)$, and $(x_1, x_2 - 0.1)$, then returns the point with the lowest value. Run this program for each of the three points listed in a).

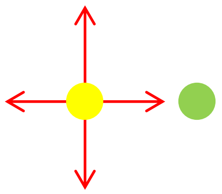
And Then, Iterations

Once we find the minimum point from the five candidates around and including the original, we can repeat the procedure.



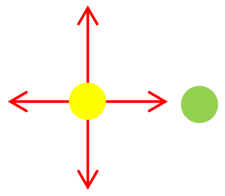
And Then, Iterations

Once we find the minimum point from the five candidates around and including the original, we can repeat the procedure.



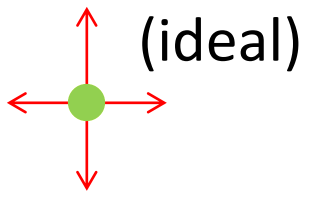
And Then, Iterations

Once we find the minimum point from the five candidates around and including the original, we can repeat the procedure.



And Then, Iterations

Once we find the minimum point from the five candidates around and including the original, we can repeat the procedure.



When the center point *is* the ideal point, then we are done.

Practice Problem 2

Embed your program from practice problem 1 into a loop such that your program runs in successive iterations until the center point is the ideal point.

Test your code with the function $f(x_1, x_2) = (x_1)^2 + 5(x_2)^2$, using the starting point (3, 5).

Then test the function $f(x_1, x_2) = (x_1 - 4)^2 + 5(x_2 + 3)^2$, with any starting point.

Concerns with the Current Program

There are two concerns with the program as it is currently.

First, the test value of 0.1 is relatively large – it guarantees a certain lack of accuracy. But raising the accuracy greatly raises the number of iterations.

Second, this method will only find the local minimum closest to the starting point, which may not be a global minimum.

First Concern: Accuracy

In our program, the accuracy is established at 0.1. Reducing this number, for example to 0.01, will increase accuracy but also iterations.

One way around this is to run the program with a larger step size until an ideal point is found, then reduce the step size and run it again from the ideal point.

As long as you're doing it this way, you can reduce the iterations greatly by starting with a much larger step size.

Practice Problem 3

Consider the function

$f(x_1, x_2) = (x_1 + x_2)^2 + (\sin(x_1 + 2))^2 + (x_2)^2 + 10.$

a) Find the ideal point using a step value of 1.
b) Using that ideal point as your starting point, repeat with a step value of 0.1.
c) Repeat for 0.01, 0.001 and 0.0001.

Practice Problem 4

Write a program that will start with a step value of 1 and successively reduce the step value through multiple iterations of the program.

Document and save this program!

Second Concern: Global

There is no “sawtooth method” for 3D functions, so the simplest thing is to test points. It still won’t guarantee a global minimum, but it will help.

(Note that testing points in 2 variables involves a lot more points because the lattice will be a 2-dimensional array rather than a 1-dimensional segment.)

Some Useful Code

Here's an example of a nested "for" loop:

```
for x = 1:4
    for y = 1:4
        println(x, ", ", y)
    end
end
```

You can also do it this way:

```
for x = 1:4, y = 1:4
    println(x, ",  ", y)
end
```

More Useful Code

A reminder of how to use “for” loops for non-integer values:

```
for x = 1:0.25:12
    println(x)
end
```

Here’s another way, in which you could also enter any function you like into a “for” loop:

```
for x in {1+.25n for n = 0:12}
    println(x)
end
```

Practice Problem 5

Write a program that will test points on a 2-dimensional interval (x_1 from a to b, x_2 from c to d) by dividing both intervals into 5 equal sub-intervals and returning the point with the lowest value.

Test with this function, known as “Rosenbrock’s Function”:

$$f(x_1, x_2) = 100(x_1 - x_2)^2 + (1 - x_2)^2$$

x_1 from -3 to 3, x_2 from -2 to 5

Document and save this program!

An Extension Problem

Modify your intelligent brute force program (practice problem 4) for a function in 3 variables, $f(x_1, x_2, x_3)$.