Finding a Global Maximum

Method 2: Sawtooth Method

(Shubert-Piyavskii Method)

Requirements

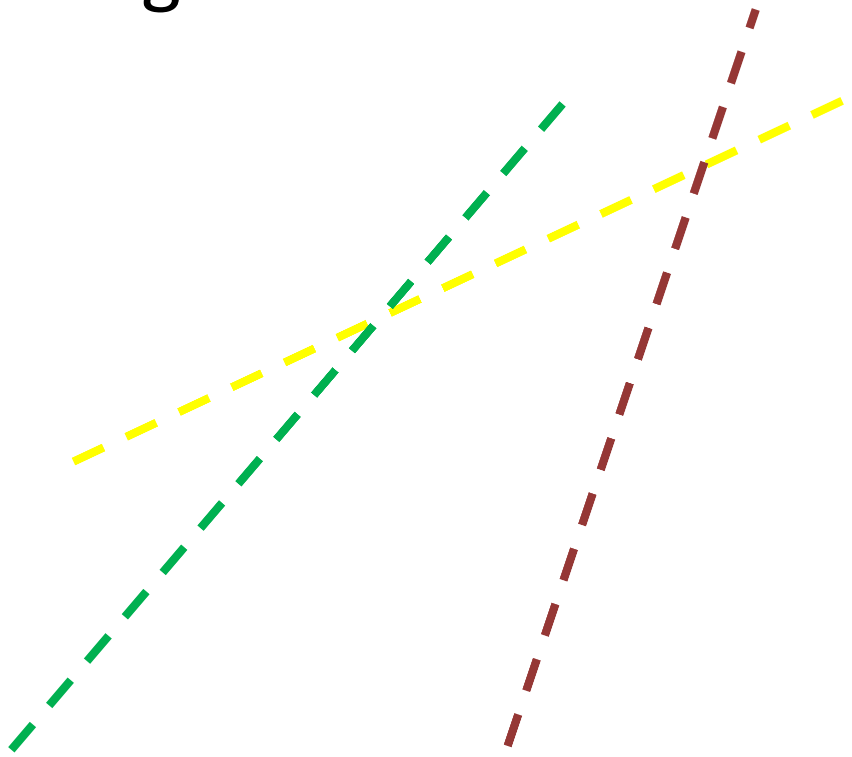
The sawtooth method requires two things that aren’t always obvious:

1. Left and right boundaries (from common sense or ceiling methods – same as previous lesson)
2. A boundary on the slope of the graph…

(What does that mean?)

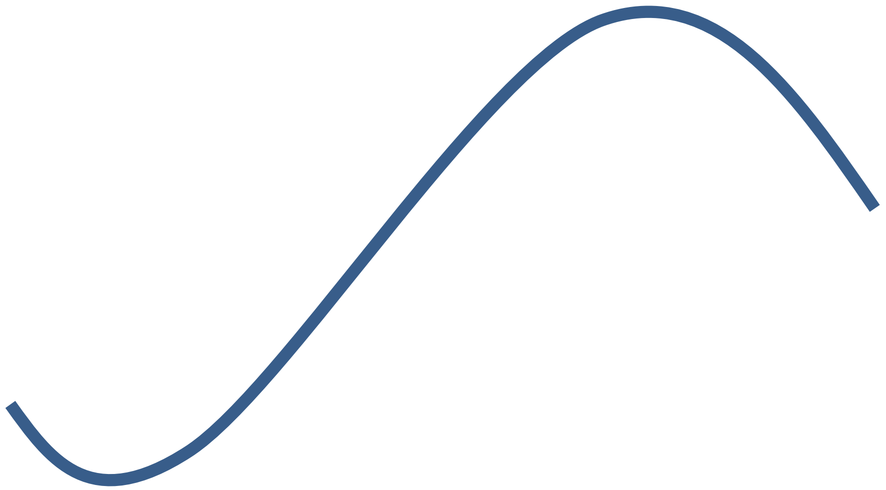
Curved Functions and Slope

Slope is a term used to describe the steepness of straight lines.



Curved Functions and Slope

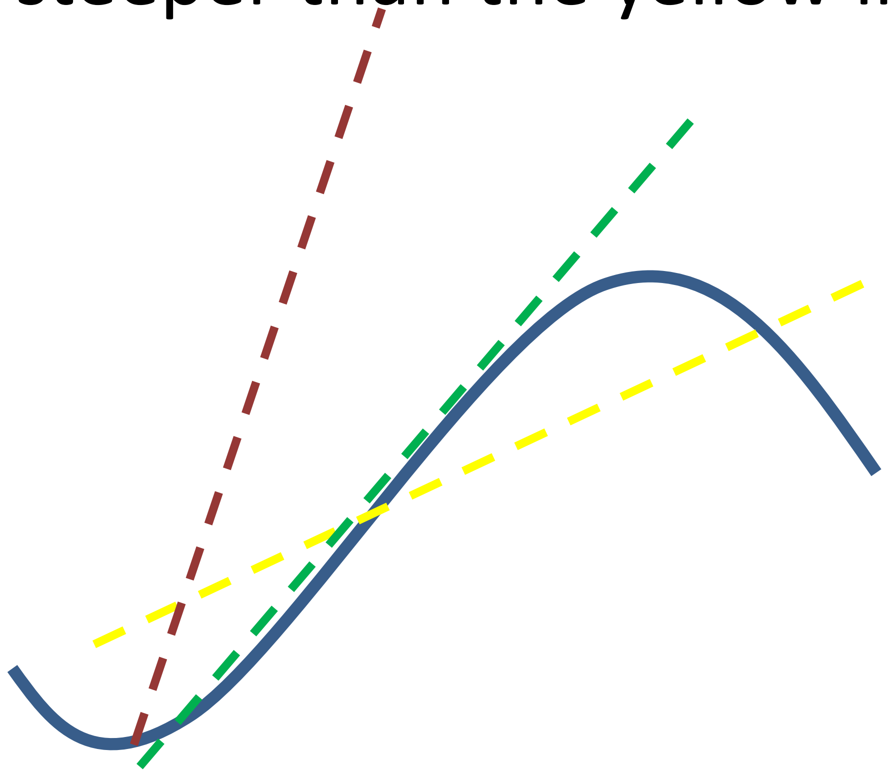
Slope is a term used to describe the steepness of straight lines.



But even curved functions have *steepness*.

Curved Functions and Slope

In this diagram, the steepest part of the curve is steeper than the yellow line…

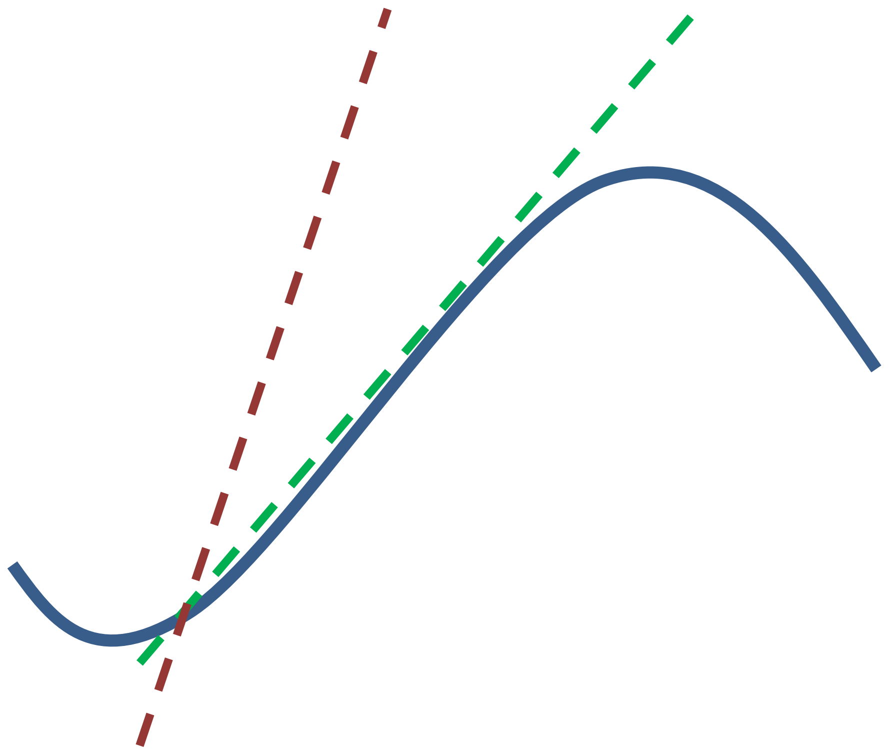


…and shallower than the red line…

…and about the same as the green line.

Requirements

For the Shubert-Piyavskii method, you need to have a boundary for the slope.



Both the green and red lines accomplish this – the curve is not steeper than either one.

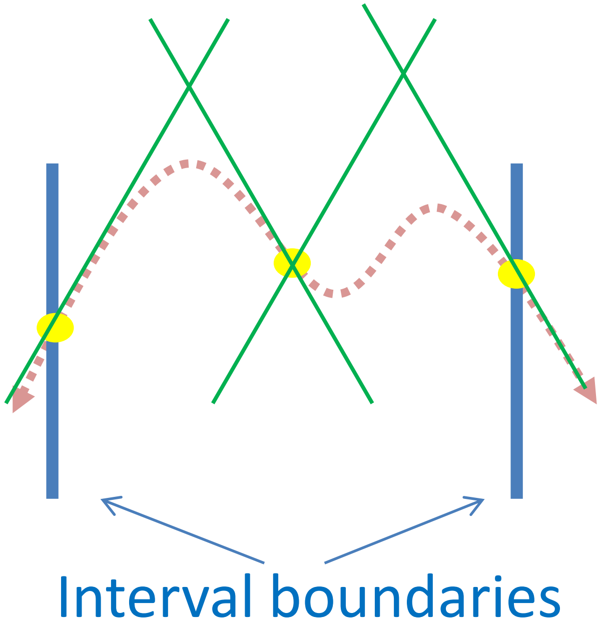
The Sawtooth Method

The Shubert-Piyavskii method, more often called the Sawtooth Method, is a way to find a global maximum on a known interval. Unlike testing points, it is guaranteed to find the global maximum on the interval.

It can be modified to find a global minimum; or you can maximize (–f(x)) using the method, which is the same as minimizing f(x).

The Sawtooth Method

For the first step, you find three points on the function: one at each boundary and one at the midpoint of the interval.



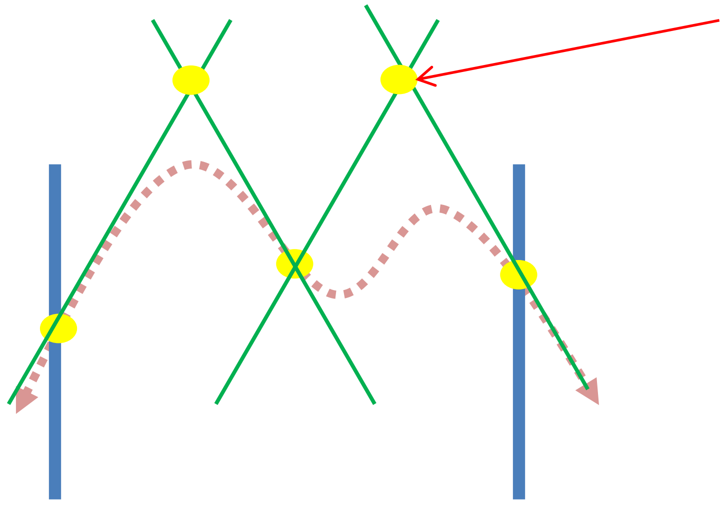
Then, using the boundary slope m, draw a line through each point with slope m or –m.

Practice Problem 1

1. Given the function $y = -x^4 + 4x^3 + 30x^2 - 50x + 200$, boundaries -5 and 7, and a maximum slope of 450…
   a) Find the three points (endpoints and midpoint) required by the Sawtooth Method
   b) Find the equations of the four lines with slope ±450 through those points.
   c) Find the two points where the left-hand lines cross and the right-hand lines cross.

The Next Step

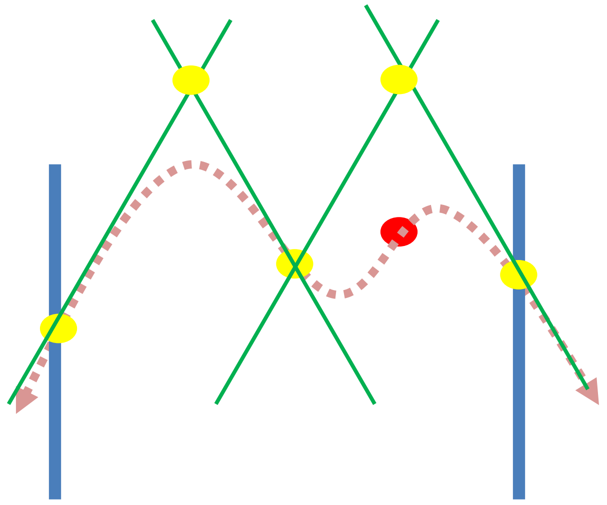
Once you have found where the lines cross, you have five points.



The next step is to choose the point with the highest y-value. This point will always be above the function graph if your boundary slope is chosen properly.

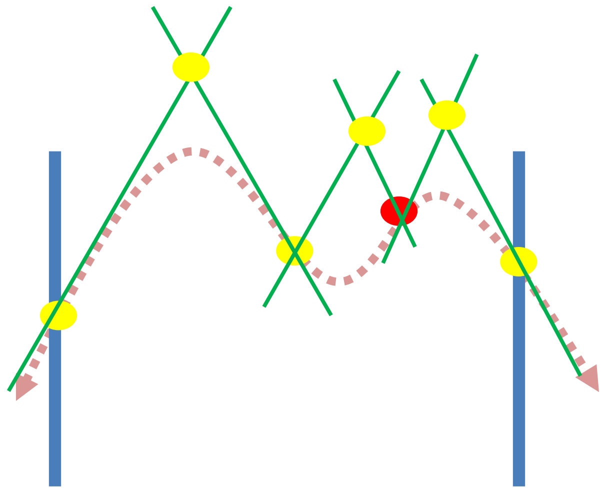
The Next Step

Take the x-value of the highest point and plug that into the function to find the actual y-value of the function at that location.



The Next Step

Take the x-value of the highest point and plug that into the function to find the actual y-value of the function at that location.



Create new lines from the new point. Then find the new crossing points.

Practice Problem 2

2. Given two equations:

$$y - y_1 = m(x - x_1)$$

$$y - y_2 = -m(x - x_2)$$

a. Write an equation that solves for y, given (x_1, y_1) and (x_2, y_2).
b. Write another equation that solves for x, given y in either of the equations above.
c. Write a program in Julia that finds the crossing point of two lines, given (x_1, y_1, x_2, y_2) and using a constant slope m.
d. Check your work on practice problem 1 using your program.

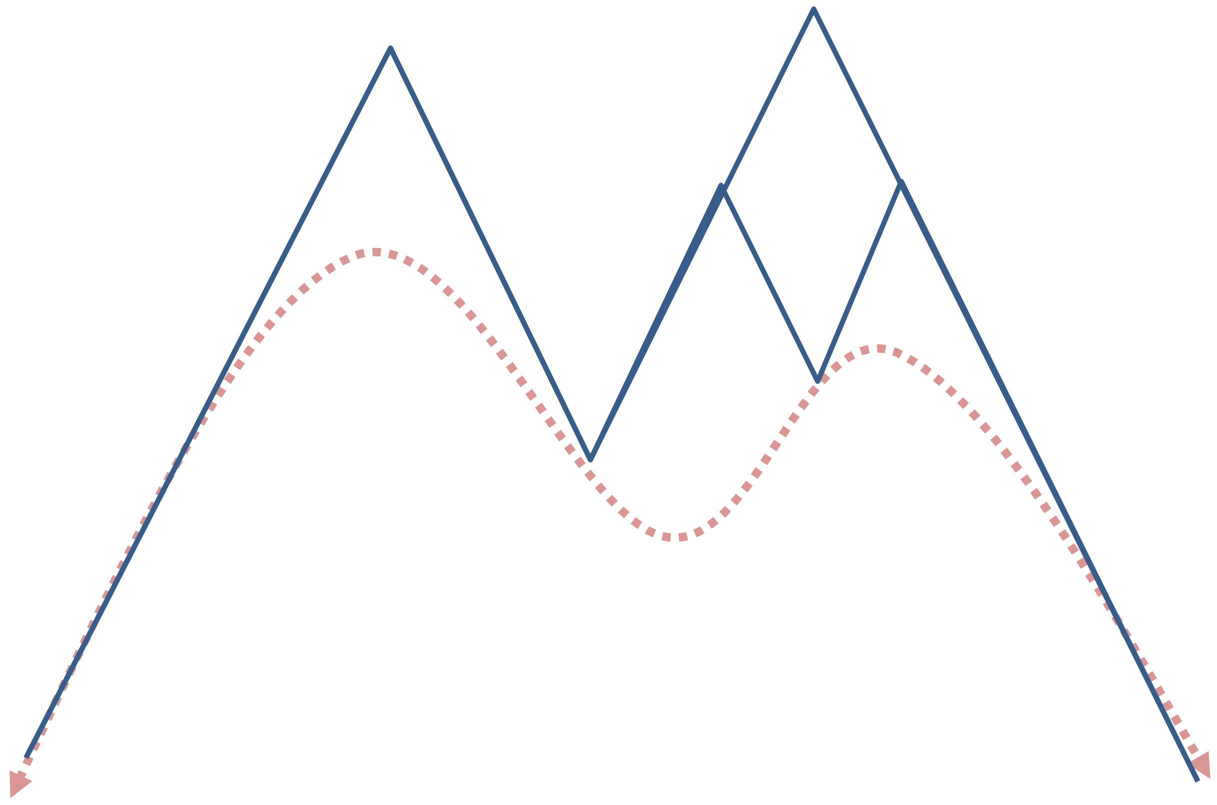
Practice Problem 3

Using $y = -x^4 + 4x^3 + 30x^2 – 50x + 200$ with boundary slope 450 and the five current points you have,

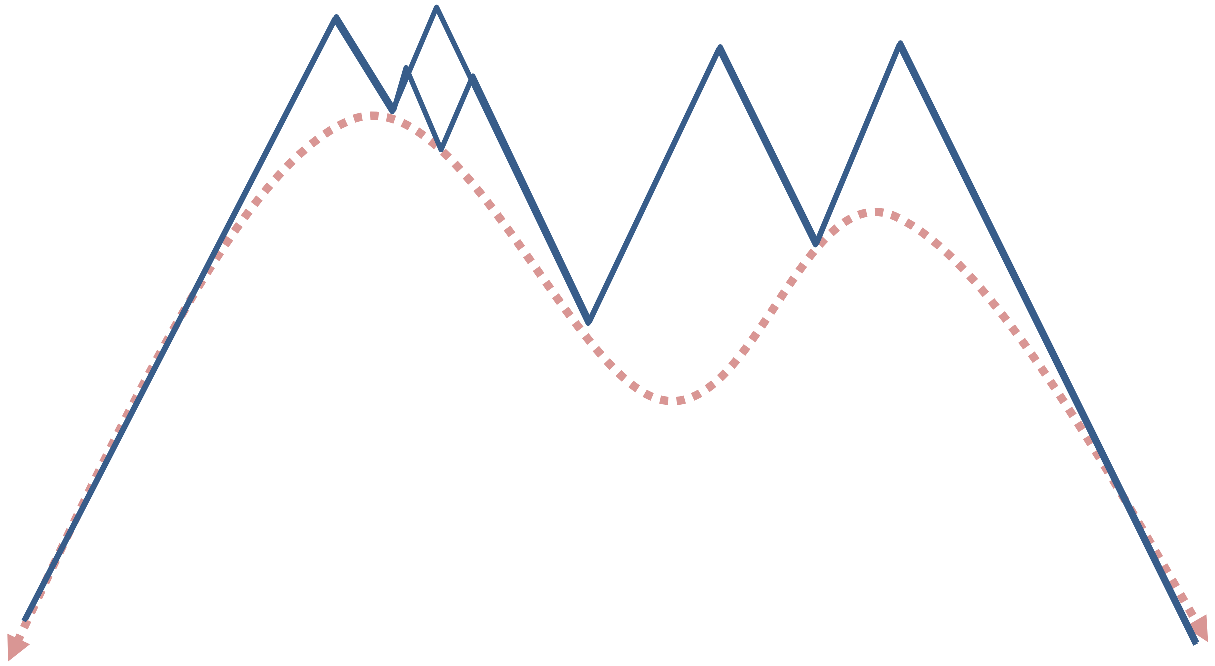
a. Choose the point with the highest y-value and find the corresponding function y-value.
b. Using your program, find where the lines from the new point cross the old lines.

Now you have seven points.

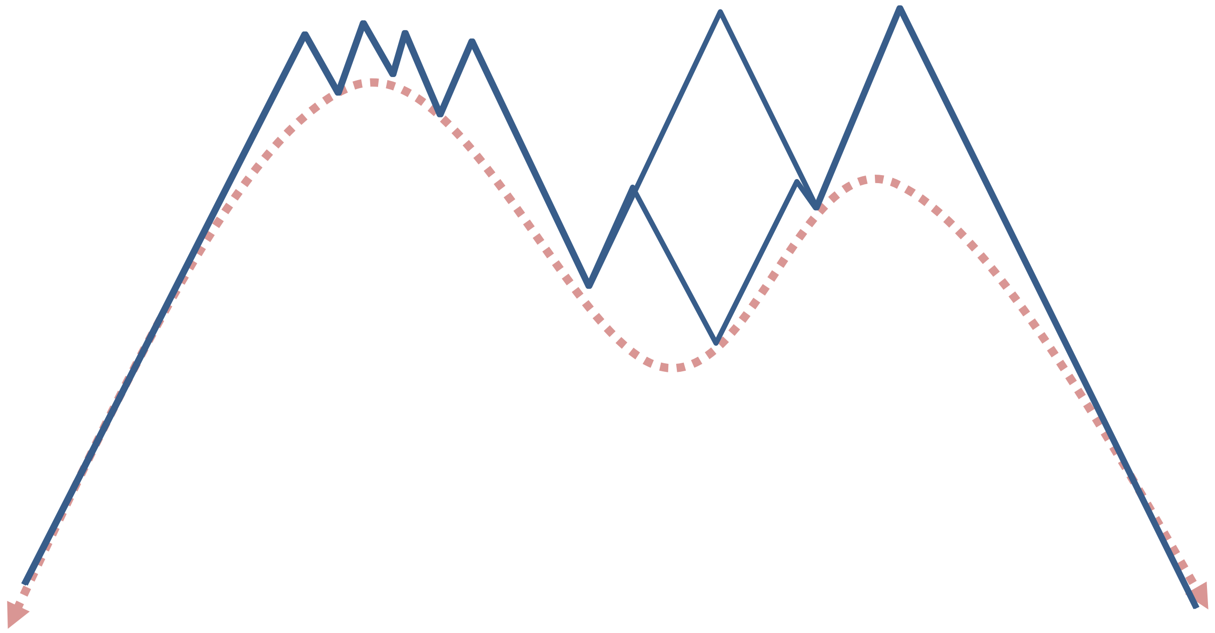
Next: Repeat…



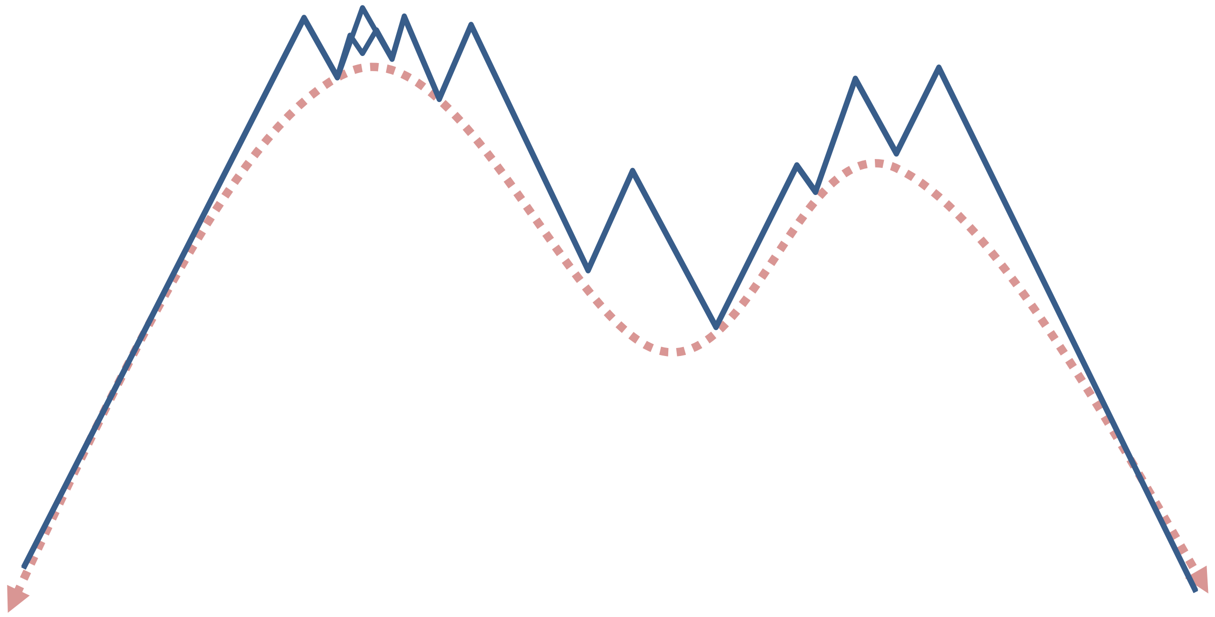
Next: Repeat…



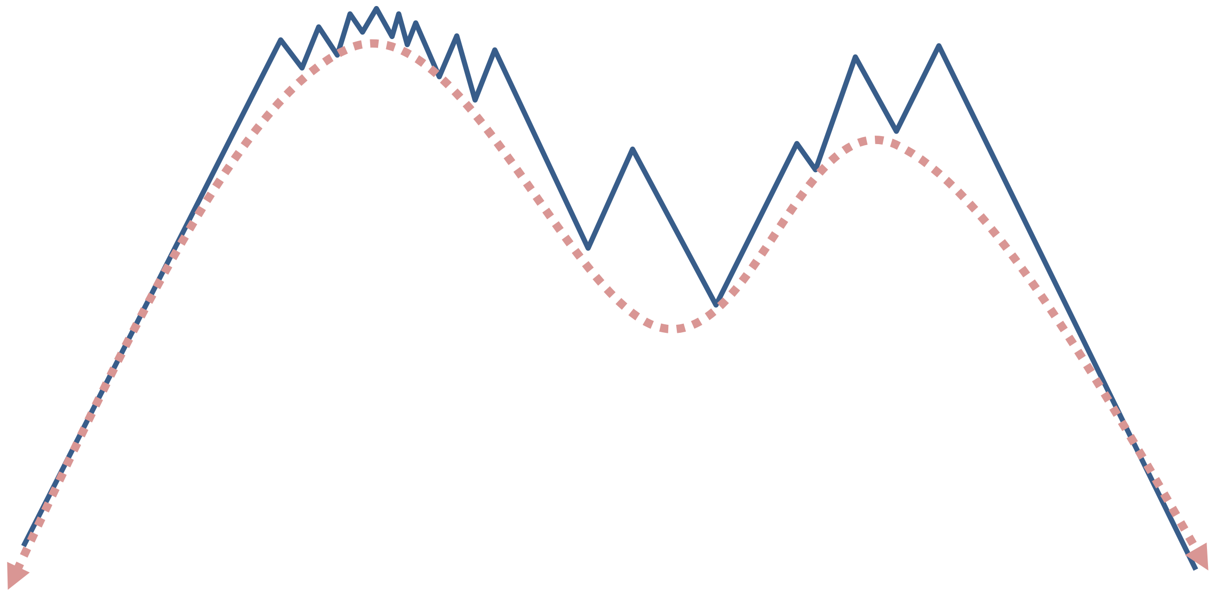
Next: Repeat…



Next: Repeat…



Next: Repeat…



…until the value of the highest point is close enough to the value of the function to stop.

Arrays in Julia, revisited

Go back to the original 3 points: (-5, 75), (1, 183), (7, 291). Enter them as an array A:

```
A = [-5 75; 1 183; 7 291]
```

If we want to add another point into A, we can enter:

```
A = vcat(A, [-1.88, 1479])
```

If we want to put the points in order, we can use

```
A = sortrows(A)
```

Output with “return”

The command “`A =`” on the previous slide replaced A with the result of the operation. To replace A with the result of a function or program, you will use the command

```
    return(A)
```

This command works like println, only instead of just printing out the result it will make the result available for assignment to a variable, as in

```
    A = function(A)
```

Practice Problem 4

Write a program listpoints(A, x, y) in Julia that adds the point (x, y) into array A, then sorts the rows. For practice, use “return” for your output.

Use your program to add in all the points you’ve found into A – seven total points. Use these statements:

```
A = listpoints([-5, 75], 1, 183)
A = listpoints(A, 7, 291)
A = listpoints(A, -1.88, 1479)
…
println(A)
```

Repetition Steps

To repeat the process, you would…

1. find the highest y-value on your list of points.
2. change that y-value to the function value.
3. find the intersection of the lines from this new point to the two adjacent points on the list.
4. Add both of those points to the list.
5. Repeat.

Practice Problem 5

Use a combination of your Julia programs – a one-line function for the equation $y = -x^4 + 4x^3 + 30x^2 - 50x + 200$; the program to find intersection points; and the point-listing program – to repeat the sawtooth method at least two more times on the equation . You will finish with at least 11 points in your array.

Saving the programs in this lesson is optional.

An Extension

Write a program in Julia that, given a function, maximum slope, and endpoints, runs the sawtooth method by either a set number of iterations, or until the highest linear y-value is within 0.01 of the function value.

One more useful command:

```
findmax(A[:, 2])
```

returns the highest y-value and its index

```
loc = findmax(A[:, 2])[2]
```

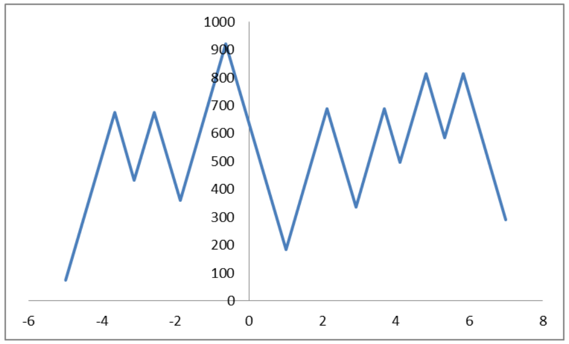
assigns the index with the highest y-value to the variable "loc".

If you complete this program, save it.

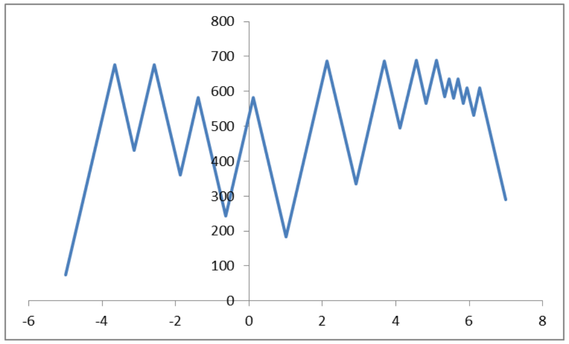
If you're curious...

Here are some scatterplots of data after ...

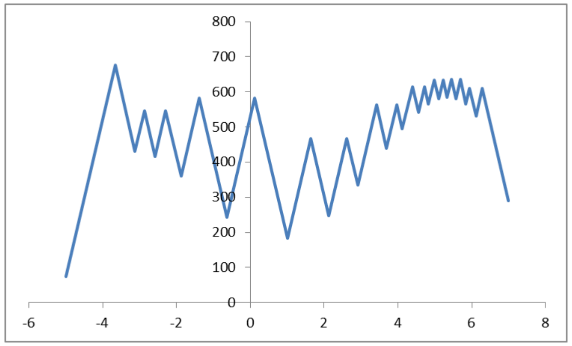
5 iterations...



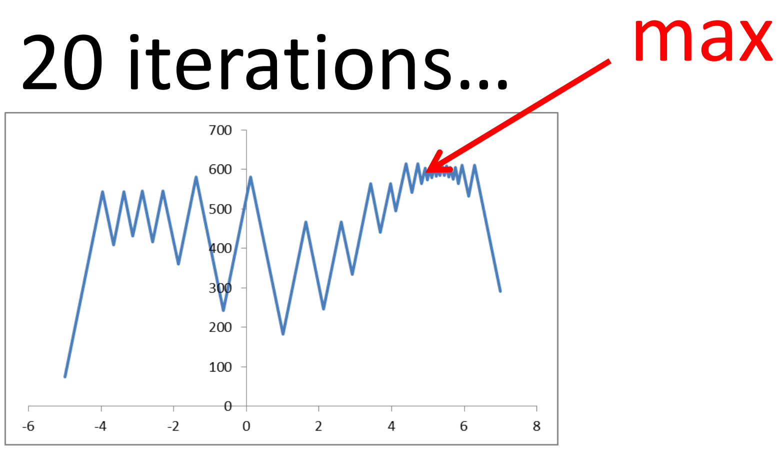
10 iterations...



15 iterations...



20 iterations...


max