Golden Search Technique

When a function is not differentiable, or is difficult to differentiate, then we may end up using a search technique to identify the function minimum/maximum over a range of interest. Search techniques are generally straightforward, but can be computationally intense. For instance, if we wanted to search $f'(x) = -(x - 4.1)^2$ over the range [3, 5], we could arbitrarily split the range up into ten sections and evaluate the function at each point.

x	f(x)	Current Max
3.0	-1.210	-1.210
3.2	-0.810	-0.810
3.4	-0.490	-0.490
3.6	-0.250	-0.250
3.8	-0.090	-0.090
4.0	-0.010	-0.010
4.2	-0.010	-0.010
4.4	-0.090	-0.010
4.6	-0.250	-0.010
4.8	-0.490	-0.010
5.0	-0.810	-0.010

After 10 iterations, we could guess that the function has a local maximum of approximately –0.01 somewhere in the interval between 4.0 and 4.2. Of course, we could sub-divide this *interval of uncertainty* further and continue to iterate until such time as we found our local maximum with sufficient accuracy.

To make things a bit simpler, let us assume that we are solving the following problem:

Maximise $f(x)$

subject to $a \le x \le b$

Furthermore, let us assume that f(x) is *unimodal*.

Unimodal

A function $f(x)$ is *unimodal* on $[a, b]$ if for some point x^* on $[a, b]$, $f(x)$ is strictly increasing on $[a, x^*]$ and strictly decreasing on $[x^*, b]$.

Note

Concave and convex functions are unimodal.

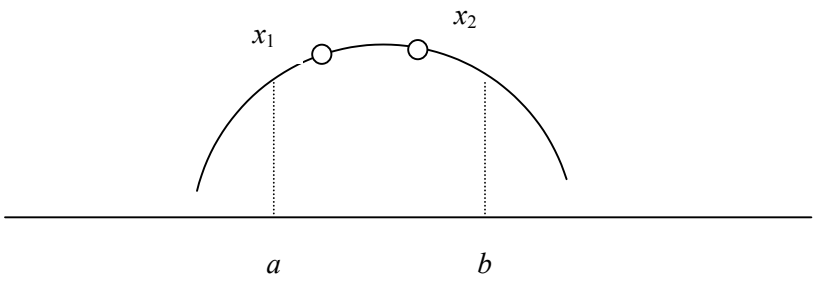
Assuming unimodal functions will make the search easier for us, since we know that we are looking for only one maximum (or minimum) point.

The Golden Search Algorithm

Searches are generally simple – but we want to be smart about how we do them.

We start with an interval of uncertainty (the interval in which we know that our maximum must lie) equal to [a, b], whose length is $b - a$.

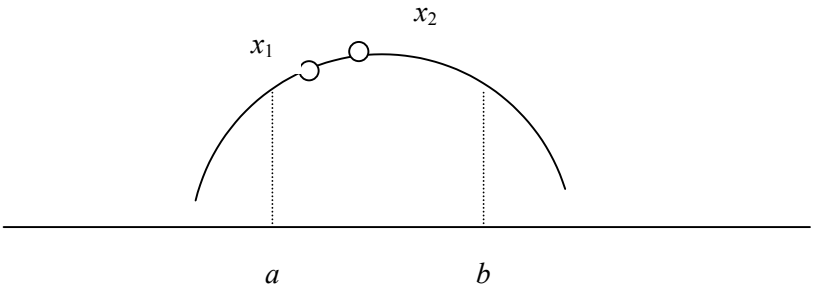
We pick two points in the interval [a, b] and evaluate the function at these points.



If $f(x_1) < f(x_2)$, then we know that in the range [x_1, x_2] that the function is increasing. Therefore, at worst, we know that the function value must be greater than $f(x_1)$. Since the function is unimodal, then we know that the maximum cannot be less than x_1. Thus, we may conclude that the maximum is in the range of (x_1, b].

Note

We cannot assume that the maximum is in the range [x_1, x_2]. See below:



If $f(x_1) > f(x_2)$, then we know the lower bound on the function is $f(x_2)$. Since the function is unimodal, the function's maximum must be greater than x_2. Therefore the maximum must lie in the range [a, x_2).

If $f(x_1) = f(x_2)$, then we know the maximum must lie in the range (x_1, x_2) since the points x_1 and x_2 have to be on either side of the maximum.

Given this information, all we have to do is update our interval of uncertainty and to restart the process. For example, if $f(x_1) < f(x_2)$, and the interval of uncertainty is [a, b], the new interval becomes (x_1, b]. If $f(x_1) > f(x_2)$, and the interval of uncertainty is [a, b], the new interval becomes [a, x_2). If $f(x_1) = f(x_2)$, and the interval of uncertainty is [a, b], the new interval becomes [x_1, x_2).

So how do we pick x_1 and x_2?

The values of x_1 and x_2 are not picked at random. In a golden search, the x_1 and x_2 are picked such that each point sub-divides the interval of uncertainty into two parts where:

$$\frac{\text{Length of whole line}}{\text{Length of larger fraction}} = \frac{\text{Length of larger fraction}}{\text{Length of smaller fraction}}$$

If we assume a line segment [0, 1] then

$$\frac{1}{r} = \frac{r}{1-r}$$

$$1 - r = r^2$$
$$r^2 + r - 1 = 0$$

Taking only the positive root from the quadratic equation, we find

$$r = \frac{-1+\sqrt{5}}{2}$$

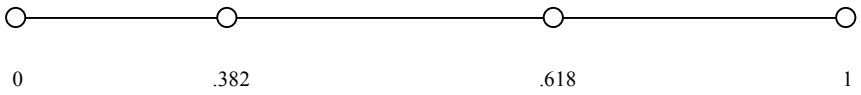
Evaluating this, we find $r = 0.618$.

To select x_1, we subtract $r(b - a)$ from b. (x_1 is 0.618 of the interval away from b).

To select x_2, we add $r(b - a)$ to a. (x_2 is 0.618 of the interval away from a).

Dividing the line segment up in this way gives us one obvious benefit. Each time we update the interval of uncertainty, we can re-use one of the two test points, if it turns out to be computationally expensive to calculate the function value.

For example, assume a search on [0, 1].



$a = 0$
$b = 1$

$x_1 = b - 0.618(b - a)$
$= 1 - 0.618$
$= .382$

$x_2 = a + .618(b - a)$
$= 0 + 0.618$
$= 0.618$

If $f(x_1) \geq f(x_2)$, x^* must be in $[a, x_2)$

$a = 0$
$b = 0.618$

$x_1 = b - 0.618(b - a)$
$= 0.618 - 0.618(0.618)$
$= 0.236$

$x_2 = 0 + 0.618(b - a)$
$= 0 + 0.618(0.618)$
$= 0.382$

If $f(x_1) < f(x_2)$, x^* must be in $(x_1, b]$

$a = 0.382$
$b = 1$

$x_1 = b - 0.618(b - a)$
$= 1 - 0.618(0.618)$
$= 0.618$

$x_2 = a + 0.618(b - a)$
$= 0.382 + 0.618(0.618)$
$= 0.764$

The Golden Search Algorithm

Assume we start with points (a, b) and a desired accuracy for $x^* = \varepsilon$.

Note that, at each iteration, the uncertainty interval $L_k = r^k(b_0 - a_0)$. Thus, we can determine the *number of iterations* necessary to find x^* before starting the process.

Step 1

Define the interval of uncertainty as $(a_0, b_0) = (a, b)$.

Step 2

Calculate $x_{1k} = b_k - 0.618(b_k - a_k)$.
Calculate $x_{2k} = a_k + 0.618(b_k - a_k)$.

Step 3

If $f(x_{1k}) < f(x_{2k})$, x^* must be in (x_{1k}, b_k).

$$a_{k+1} = x_{1k}$$
$$b_{k+1} = b_k$$

Else must be in (a_k, x_{2k}).

$$a_{k+1} = a_k$$
$$b_{k+1} = x_{2k}$$

Step 4

If $L_k = (b_k - a_k) \le \varepsilon$, stop. Otherwise, set $k = k + 1$ and go to **Step 2.**

Example

Assume that we want to solve the following problem to an accuracy of 0.001:

Maximise $f(x) = -(x - 4.1)^2$
subject to $3 \le x \le 5$

Note

Solving $0.001 = 0.618^k(5-3)$ will allow us to determine the number of iterations.

$$k = \left(\frac{\ln(0.001/2)}{\ln(0.618)} \right)$$

In this case, $k = 15.79$, so we will need at least 16 iterations.

Iteration	a	b	x1	x2	f(x1)	f(x2)	x* in interval	Lk
0	3.0000	5.0000	3.7640	4.2360	-0.1129	-0.0185	x* in (x1,b)	2.00000
1	3.7640	5.0000	4.2362	4.5278	-0.0185	-0.1831	x* in (a, x2)	1.23600
2	3.7640	4.5278	4.0558	4.2361	-0.0020	-0.0185	x* in (a, x2)	0.76385
3	3.7640	4.2361	3.9443	4.0557	-0.0242	-0.0020	x* in (x1,b)	0.47206
4	3.9443	4.2361	4.0558	4.1246	-0.0020	-0.0006	x* in (x1,b)	0.29173
5	4.0558	4.2361	4.1246	4.1672	-0.0006	-0.0045	x* in (a, x2)	0.18029
6	4.0558	4.1672	4.0983	4.1246	0.0000	-0.0006	x* in (a, x2)	0.11142
7	4.0558	4.1246	4.0821	4.0983	-0.0003	0.0000	x* in (x1,b)	0.06886
8	4.0821	4.1246	4.0983	4.1084	0.0000	-0.0001	x* in (a, x2)	0.04255
9	4.0821	4.1084	4.0921	4.0983	-0.0001	0.0000	x* in (x1,b)	0.02630
10	4.0921	4.1084	4.0983	4.1022	0.0000	0.0000	x* in (a, x2)	0.01625
11	4.0921	4.1022	4.0960	4.0983	0.0000	0.0000	x* in (x1,b)	0.01004
12	4.0960	4.1022	4.0983	4.0998	0.0000	0.0000	x* in (x1,b)	0.00621
13	4.0983	4.1022	4.0998	4.1007	0.0000	0.0000	x* in (a, x2)	0.00384
14	4.0983	4.1007	4.0992	4.0998	0.0000	0.0000	x* in (x1,b)	0.00237
15	4.0992	4.1007	4.0998	4.1001	0.0000	0.0000	x* in (x1,b)	0.00147
16	4.0998	4.1007	4.1001	4.1003	0.0000	0.0000	x* in (a, x2)	0.00091

Thus, we would conclude that the function reaches a maximum of (0.0000, 0.0000) in the range (4.0998, 4.1007).

Note

The Golden Section Search may not be the most efficient search technique. It works well when $f(x)$ is complicated (we can take advantage of the fact that we need only do one function determination at each interval other than iteration 0) and unimodal. You should also note that the golden section search could be tailored to work on functions that are not unimodal and not maximization problems. However, the way in which we have structured the algorithm above requires a unimodal maximization problem.