

Accelerated bisection search for minimizing unimodal functions

Vladimir Kodnyanko

Polytechnic Institute, Siberian Federal University, Krasnoyarsk, 660079, Russia

email: vkodnyanko@sfu-kras.ru

This review describes an accelerated bisection method. The essence of accelerating classical bisection search is to eliminate redundant calculations of the unimodal function being minimized, thereby increasing the method's speed. The classical bisection is approximately 1.5 times slower than the golden section search. However, accelerated bisection is virtually as fast as the golden section and, in some cases, even exceeds it. For the selected functions and intervals, the overall efficiencies of accelerated bisection and the golden section were approximately the same.

Keywords: unimodal function, bisection search, golden section search, speed of minimization methods

1. Introduction

Methods and software systems for minimizing a function of one variable $f(x)$ are among the computational tools for solving various theoretical and practical problems [1, 2, 3]. In theory, they are often used in the development of multidimensional minimization methods and, in practice, in solving various optimization problems, from agriculture to mechanical engineering [4, 5].

Among the well-known and most frequently used methods for minimizing unimodal functions in practice are two classical methods of high reliability: bisection search and golden section search.

It is believed that the golden section works faster than bisection [3, 6], mainly because, starting from the second iteration, the function is calculated only once, whereas bisection requires two calculations.

This review shows that by simply accelerating the bisection, one can achieve a speed that is not inferior to the golden section.

2. The idea of accelerating bisection

The essence of the method's improvement lies in eliminating unnecessary function calculations via the bisection method during the minimization of a unimodal function. Indeed, if the first calculated function value at the current iteration exceeds its previously found minimum, then in this case, a new boundary of the uncertainty interval can be determined without performing a second calculation.

3. Procedure codes

Below are the codes of the procedures of classical bisection, accelerated bisection [2] and the golden section [6] in the Delphi language [7], which were used in calculations to compare the speed of accelerated bisection and the golden section.

```
procedure Bi; //classical bisection search
var f1,f2: Extended; n: Integer;
begin k:= 0;
  repeat
    x:=(a+b)/2.0; f1:= f(xe); f2:= f(x+e); Inc(k,2);
    if f1 < f2 then b:= x else a:= x;
  until ba < e;
  x:=(a+b)/2.0;
end ;
```

```

procedure BiS; //accelerated bisection search
  var f1,f2,x1,x2,gx, gy: Extended;
begin k:= 0; gy= 1.0e65;
  repeat
    x:=(a+b)/2.0; f2:= f(x); Inc(k);
    if f2 < gy then
      begin
        gx:=x; gy:= f2; f1:= f(xe); Inc(k);
        if f1 < f2 then b:= x else a:= x;
      end
    else
      if x < gx then a:= x else b:= x;
    until ba < e;
    x:=(a+b)/2.0;
  end ;

```

```

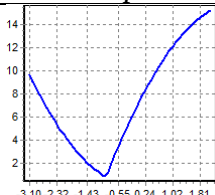
procedure Gold; //golden section search
  var f1,f2,x1,x2,G1,G2: Extended;
begin k:= 2;
  G1:= 0.5*(3.0-Sqrt(5.0)); G2:= 1.0-G1;
  x1:= G2*a+G1*b; f1:= f(x1);
  x2:= G1*a+G2*b; f2:= f(x2);
  repeat
    if f1 < f2 then
      begin
        b:= x2; x2:= x1; f2:= f1;
        x1:= G2*a+G1*b; f1:= f(x1);
      end
    else
      begin
        a:= x1; x1:= x2; f1:= f2;
        x2:= G1*a+G2*b; f2:= f(x2);
      end ;
    Inc(k);
  until ba < e;
  x := (a + b)/2.0;
end ;

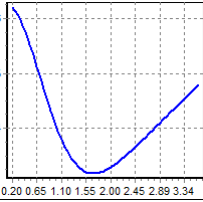
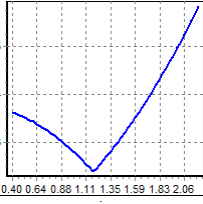
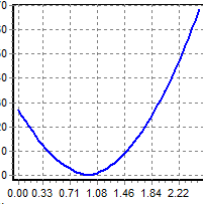
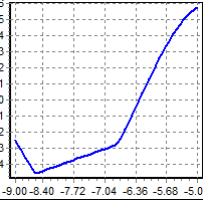
```

4. Comparative results

Table 1 provides the initial data, including uncertainty intervals, function graphs and their formulas, and presents the results of minimizing five unimodal functions.

The columns *Bi*, *BiS* and *Gold* give the number of function calls required to achieve a result with an accuracy of $e = 10^{-5}$.

<i>N</i>	<i>Graph</i>	$[a, b]$	$f(x)$	x	<i>Bi</i>	<i>BiS</i>	<i>Gold</i>
1		$[-3.1, 2.1]$	$\max(x^2, 16 - \sqrt{x-3})$	-0.89792	38	29	30

2		[0.2, 3.6]	$1.2 + 5\exp(-x^2) + x$	1.67963	38	33	32
3		[0.4, 2.2]	$1 + \max[x^2, 2.8 - x^2]$	1.11693	42	26	28
4		[0, 2.5]	$\max[-2(x-1), 8(x-1)] + 25(x-1)^2$	1	36	31	28
5		[-9, -5]	$\max[x + 4, 6\sin(x)]$	-8.56117	38	28	29
Σ					192	147	147

It is clear that classical bisection search is approximately 1.5 times slower than golden section search. However, accelerated bisection is practically as fast as the golden section and, in some cases, even exceeds it. For the chosen functions and intervals, the overall efficiencies of accelerated bisection and the golden section were approximately the same.

Conclusion

This review describes an accelerated bisection method for minimizing unimodal functions. The essence of this improvement over classical bisection search is to eliminate redundant calculations of the unimodal function being minimized, thereby increasing the method's speed. Using the example of minimizing five functions, it is shown that the method's speed is roughly comparable to that of the golden section search.

Funding. No funding was received for conducting this paper.

Author Contributions. All contributions to this work were made by the author.

Data availability. Datasets were generated and analyzed during this study.

Declarations Competing Interests The authors declare that they have no conflict of interest.

Human and Animal Rights Not Applicable.

Informed Consent Not Applicable.

References

1. Kochenderfer MJ, Wheeler TA Algorithms for Optimization. MIT press, 2019.
2. Bonnans JF, Gilbert J. Ch., Lemaréchal C., Sagastizábal C. Numerical Optimization - Theoretical and Numerical Aspect. 2006.
3. Kodnyanko V. Optimal Section Method for Minimizing Unimodal Functions. Comput Math Model 36, 235–252, 2025. <https://doi.org/10.1007/s10598-025-09641-z>
4. Santos, PAVH d., Silva, ACL d., Arce, JE, & Augustynczyk, ALD A Mathematical Model for the Integrated Optimization of Harvest and Transport Scheduling of Forest Products. Forests, 10(12), 1110. 2019. <https://doi.org/10.3390/f10121110>
5. Tian, F., Yang, C., Zhang, E., Sun, D., Shi, W., & Chen, Y. Design Optimization of Hydraulic Machinery Based on ISIGHT Software: A Review of Methods and Applications. *Water* , 15 (11), 2100, 2023. <https://doi.org/10.3390/w15112100>
6. Kiefer J. Sequential minimax search for a maximum, Proceedings of the American Mathematical Society, 4 (3): 502–506, 1953.
7. Bucknall, J. The Toms of Delphi: Algorithms and Data Structures. 2001. P. 255. ISBN1-55622-736-1