

Минимизация функций одного переменного в программном комплексе Maple

Вихляев Дмитрий Романович

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В работе описываются примеры реализации программных модулей минимизации функций одного переменного. Представлен программный код в системе Maple. Решения задачи представлено с помощью методов дихотомии, золотого сечения, Фибоначчи и сканирования.

Ключевые слова: метод Maple, минимизация функции, точка экстремума

Minimization of single-variable functions in the Maple software package

Vikhlyayev Dmitry Romanovich

Sholom-Aleichem Priamursky State University

Student

Abstract

The paper describes examples of implementing software modules for minimizing functions of a single variable. The software code is presented in the Maple system. The solution of the problem is presented using the methods of dichotomy, golden ratio, Fibonacci, and scanning.

Keywords: Maple, function minimization, extremum point

1 Введение

1.1 Актуальность

Основная задача оптимизации – минимизация функции одной переменной, то есть нахождение значения x (x^*) из заданного множества значений, для которых целевая функция $f(x)$ даст наименьшее значение. Если это множество представляет собой интервал, ограничивающий диапазон возможных значений x , то ответ на задачу минимизации может находиться либо внутри этого интервала, либо на его границе.

Для решения этой проблемы был разработан целый класс одномерных методов поиска, позволяющих постепенно сужать интервал неопределенности, содержащий точку экстремума. Все методы поиска не требуют вычисления производных функции, а основаны исключительно на сравнении значений функции в тщательно выбранных точках.

К числу наиболее популярных методов прямого поиска, используемых на практике, относятся метод дихотомии, метод золотого сечения, метод Фибоначчи и метод простого сканирования. Все они, по сути, имеют один и

тот же алгоритм. На каждом шаге процесса выбираются две пробные точки x_1 и x_2 в текущем диапазоне для проверки значений функции $f(x_1)$ и $f(x_2)$. После их сравнения исключается часть интервала, не включающая минимум. Завершение цикла (итеративного процесса) наступает, когда длина интервала L_n ($b_n - a_n$) становится меньше заданной точности ε .

1.2 Обзор исследований

В своей работе Ю.М.Данилин описал методы минимизации, основанные на аппроксимации исходного гладких функционалов при условии, что члены отрезки ряда Тейлора функционала являются выпуклыми в рассматриваемой области [1]. В.А.Коднянко исследовал минимизацию унимодальных функций методом оптимизации [2]. М.А.Гришкин продемонстрировал метод минимизации однопараметрической функции на ЭВМ [3]. Т.В.Бодякина, Т.Б.Власова провели исследование функции средствами Maple [4]. В.З.Аладьев описал основы программирования в Maple [5]. В работе Н.А.Шерстнева приведены примеры использования системы компьютерной математики Maple в учебном процессе [6]. В.А.Коднянко изучил локальную минимизацию функций одной переменной методом дихотомии подобных точек [7].

1.3 Цель исследования

Цель исследования – привести примеры минимизации функций одного переменного в программном комплексе Maple.

2 Материалы и методы

Исследуется задача минимизации функции вида $y = -\ln(x) + \frac{\cos(2x)}{1.5}$ с точностью $\varepsilon = 10^{-3}$ на отрезке $[1;6]$ в программном комплексе Maple методами дихотомии, золотого сечения, Фибоначчи и сканирования.

3 Результаты и обсуждения

Функция $y = -\ln(x) + \frac{\cos(2x)}{1.5}$ является примером сложной нелинейной зависимости, которая объединяет монотонно убывающий логарифм $-\ln(x)$ и периодическую тригонометрическую функцию $\cos(2x)$. Если взять производную и приравнять её к нулю для поиска экстремума, получится трансцендентное уравнение. Но его невозможно решить, аналитически выразив x через простые формулы. Поэтому применение численных методов одномерной оптимизации здесь необходимо.

На заданном отрезке $[1;6]$ функция имеет единственный глобальный минимум, что гарантирует точную и стабильную сходимость любого из методов исключения интервалов (рис.1).

Рис. 1. График исследуемой функции на отрезке $[1;6]$.

На рисунке 2 представлен код программы Maple, который реализует модифицированный алгоритм дихотомии для унимодальной функции с использованием встроенного компилятора математических выражений.

Поиск минимума осуществляется через проверку в цикле. Критерий остановки по длине полуинтервала $\frac{b-a}{2} > \varepsilon$. На каждой итерации вычисляются две пробные точки, смещенные от центра отрезка на параметр различимости eps . Функция `evalf` принудительно запускает программный сопроцессор вещественной арифметики, переводя выражения в тип `float` с плавающей запятой. Сужение интервала происходит путем динамического переприсвоения значений переменным границ a или b в зависимости от того, в какой точке значение функции оказалось меньше.

```
f := x -> -ln(x) + cos(2*x)/1.5 :
a := 1 : b := 6 : eps := 0.001 :
i := 0 : a := 1 : b := 6 :
while ((b-a)/2) > eps do
  i := i + 1 :
  x1 := (a + b - eps)/2 :
  f1 := f(x1) :
  x2 := (a + b + eps)/2 :
  f2 := f(x2) :
  if (evalf(f1) < evalf(f2)) then
    b := x2 :
  else
    a := x1 :
  fi:
od:
printf("Дихотомия\n%d\nX= %10.9f\nFх= %10.9f\n", i, (a + b)/2, f((a + b)/2));

Дихотомия
13
X= 4.790940491
Fх= -2.225183191
```

Рис. 2. Метод дихотомии

В коде, на рисунке 3, реализован метод золотого сечения, главное отличие которого от дихотомии заключается в использовании константы

золотой пропорции $q (\approx 1.618)$ для гарантированного сжатия интервала поиска на фиксированную величину ≈ 0.618 . Технически алгоритм гораздо экономичнее, так как начальные точки и значения функции инициализируются еще до входа в цикл, а внутри него за счет переприсвоения переменных () одна из старых точек всегда становится новой.

```
f := x → -ln(x) + cos(2*x)/1.5 :
a := 1 : b := 6 : eps := 0.001 :
i := 0 : a := 1 : b := 6 :
q := evalf((1 + sqrt(5))/2) :
x1 := evalf(b - (b-a)/q) :
x2 := evalf(a + (b-a)/q) :
y1 := f(x1) :
y2 := f(x2) :

while evalf(b-a) > eps do
  i := i + 1 :
  if evalf(y1) < evalf(y2) then
    b := x2 :
    x2 := x1 :
    y2 := y1 :
    x1 := evalf(b - (b-a)/q) :
    y1 := f(x1) :
  else
    a := x1 :
    x1 := x2 :
    y1 := y2 :
    x2 := evalf(a + (b-a)/q) :
    y2 := f(x2) :
  fi:
od:
printf("Метод золотого сечения\n%d\nX= %10.9f\nFх= %10.9f\n", i, (a + b)/2, f((a + b)/2));

Метод золотого сечения
18
X= 4.790971573
Fх= -2.225183193
```

Рис. 3. Метод золотого сечения

Код на рисунке 4 реализует метод Фибоначчи. Технической особенностью кода является условный блок `if k < Nfib - 1`, который предотвращает выход за границы массива на финальной итерации и останавливает пересчет точек, когда интервал гарантированно достиг целевой ширины.

```

f := x → -ln(x) + cos(2*x)/1.5 :
a := 1 : b := 6 : eps := 0.001 :
i := 0 : a := 1 : b := 6 :
F := Array(0..12) :
F[0] := 1 : F[1] := 1 : Nfib := 12 :
for k from 2 to Nfib do
  F[k] := F[k-1] + F[k-2] :
od :

x1 := a + (b-a) * (F[Nfib-2]/F[Nfib]) :
x2 := a + (b-a) * (F[Nfib-1]/F[Nfib]) :
f1 := f(x1) :
f2 := f(x2) :

for k from 2 to Nfib-1 do
  i := i + 1 :
  if evalf(f1) < evalf(f2) then
    b := x2 :
    x2 := x1 :
    f2 := f1 :
  else
    a := x1 :
    x1 := x2 :
    f1 := f2 :
  fi :
  if k < Nfib-1 then
    x1 := a + (b-a) * (F[Nfib-k-1]/F[Nfib-k+1]) :
    x2 := a + (b-a) * (F[Nfib-k]/F[Nfib-k+1]) :
    f1 := f(x1) :
    f2 := f(x2) :
  fi :
od :
printf("Метод Фибоначчи\n%d\nX= %10.9f\nFх= %10.9f\n", i, (a+b)/2, f((a+b)/2));

Метод Фибоначчи
10
X= 4.798283262
Fх= -2.225111932

```

Рис. 4. Метод Фибоначчи

На рисунке 5 код реализует метод сканирования, главное отличие которого заключается в полном отказе от идеи деления. Сужения интервала неопределенности в пользу последовательного шагового обхода всей области поиска. Технически этот метод является самым ресурсоемким, так как он не переносит старые точки в памяти и на каждом шаге выполняет новое вычисление функции.

```

f := x → -ln(x) + cos(2*x)/1.5 :
a := 1 : b := 6 : eps := 0.001 :
i := 0 : h := eps : a := 1 : b := 6 :
x1 := a :
fm := f(x1) :

while (x1 < b) do
  i := i + 1 :
  x1 := x1 + h :
  fm1 := f(x1) :
  if (evalf(fm1) < evalf(fm)) then
    fm := fm1 :
    x2 := x1 :
  fi:
od:
printf("Метод сканирования\n%d\nX= %10.3f\nFx= %10.9f\n", i, x2, fm);

Метод сканирования
5000
X=      4.791
Fx= -2.225183194

```

Рис. 5. Метод сканирования

В таблице 1 представлены сравнения реализованных методов по полученному результату и количеству операций.

Таблица 1. Сравнение реализации методов минимизации функции одного переменного

Метод	Количество итераций	Приближенное решение
Дихотомия	13	X= 4.790940491 Fx= -2.225183191
Метод золотого сечения	18	X= 4.790971573 Fx= -2.225183193
Метод Фибоначчи	10	X= 4.798283262 Fx= -2.225111932
Метод сканирования	5000	X= 4.791 Fx= -2.225183194

Все методы правильно решили задачу. Глобальный минимум функции находится в точке 4.791, а минимальное значение функции составляет -2.225183.

4 Выводы

Самую высокую точность по значению функции продемонстрировал метод сканирования -2.225183194 исключительно за счет огромного числа разбиений 5000 точек, что вычислительно неоправданно. Методы золотого сечения и дихотомии показали результаты -2.225183193 и -2.225183191. Результаты практически идентичны и достаточно точны, но при этом затрачено в сотни раз меньше ресурсов, чем при использовании метода сканирования. Методу Фибоначчи понадобилось всего лишь 10 шагов, что

связано со спецификой метода. Количество итераций в нем жестко задается заранее. Чтобы повысить его точность до уровня золотого сечения, достаточно было изначально заложить в алгоритм чуть меньшее значение допустимой погрешности, что автоматически увеличило бы количество шагов, например, до 12 -14.

Библиографический список

1. Данилин Ю.М. Методы минимизации, основанные на аппроксимации исходного функционала выпуклым //Журнал вычислительной математики и математической физики. 1970. Т. 10. №. 5. С. 1067-1080.
2. Коднянко В.А. Минимизация унимодальных функций методом оптимизации //Математический вестник. 2025. №. 1. С. 32.
3. Гришкин М.А. Демонстрация методов минимизации однопараметрической функции на ЭВМ //Новые информационные технологии в автоматизированных системах. 2004. №. 7. С. 119-130.
4. Бодякина Т.В., Власова Т.Б. Исследование функции средствами Maple //Вестник ИрГСХА. 2015. №. 71. С. 110-115.
5. Аладьев В.З. Основы программирования в Maple //Таллин: Международная академия ноосферы. 2006. Т. 301. С. Т4.
6. Шерстнева Н.А. Использование системы компьютерной математики Maple в учебном процессе //Международный журнал экспериментального образования. 2015. №. 11-2. С. 231-234.
7. Коднянко В.А. Локальная минимизация функций одной переменной методом дихотомии подобных точек //Cloud of Science. 2020. Т. 7. №. 1. С. 40-48.