Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное

образовательное учреждение высшего образования

Пермский национальный исследовательский

политехнический университет

Индивидуальное комплексное задание №1

Методы оптимизации

Выполнил:

Студент группы МИЭ-17 1б

Гневашев А.Д.

Проверил:

Доц. Макаревич Е.С.

Пермь

2021

**Аналитическое решение**

Необходимо решить задачу минимизации функции с ограничениями на переменную:

Аналитическое решение: 1). D – компакт, следовательно, по теореме Вейерштрасса, существует абсолютный экстремум.

2). Проверим знакоопределённость второй производной:

Вторая производная в найденной точке больше нуля, следовательно, это значение доставляет локальный минимум.

Абсолютный минимум может существовать только среди локальных. Так как найден единственный локальный минимум, значение в котором меньше значений на концах области определения, он же является абсолютным.

Таким образом, ответ, полученный аналитическим решением:

**Исследование функции**

Наша функция:

На графике 1 она выглядит следующим образом:

График 1

Построим график с учётом ограничения:

График 2.

В заданной области функция гладкая. Функция дважды непрерывно дифференцируема. Функция унимодальна: a = 3, α = β =, b = 5. Функция липшецева на области D: |I(x1)-I(x2)| ≤ 10\*|x1-x2|; L=10;

Имеются абсолютные экстремумы, в области D существует только абсолютный минимум, согласно аналитическому решению, это точка ().

**Реализация численных методов**

**Метод золотого сечения**

Метод золотого сечения — метод поиска экстремума действительной функции одной переменной на заданном отрезке. В основе метода лежит принцип деления отрезка в пропорциях золотого сечения, т.е. деление отрезка на две неравные части так, чтобы отношение длины всего отрезка к длине большей части равнялось отношению длины большей части к длине меньшей части отрезка. Является одним из простейших вычислительных методов решения задач оптимизации. Может использоваться для поиска экстремумов только в унимодальных функциях.

Алгоритм:

1. Вычисляем две точки  и  так, что  
2. Сравниванием значение функций в этих точках. Если , то сохраняем в точку *b* значение, рассчитанное для *x2*. Иначе, сохраним в точку *a* значение точки *x1*.
3. Процедура расчета действует пока 

Кусок программного кода реализованной программы для метода золотого сечения представлен ниже.

def golden\_ratio(a, b, n):  
 data1.append([n, a])  
 if abs((Fi \*\* n) \* (b - a)) < eps: return  
 x1 = a - (Fi - 1) \* (b - a)  
 x2 = a + Fi \* (b - a)  
 if F(x1) < F(x2):  
 golden\_ratio(a, x2, n + 1)  
 else:  
 golden\_ratio(x1, b, n + 1)

Данная программа находит решение нашей задачи менее чем за 2\*10-3 секунды. Для этого требуется 20 итераций.

На графике 3 находится кривая сходимости метода золотого сечения

График 3

Полученное решение: Хmin =3,8228593, отличается от аналитического решения на 0,00002.

**Метод Ньютона**

**Метод Ньютона**, **алгоритм Ньютона**  — это итерационный [численный метод](https://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D1%8B) нахождения корня ([нуля](https://ru.wikipedia.org/wiki/%D0%9D%D1%83%D0%BB%D1%8C_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8)) заданной [функции](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)). Поиск решения осуществляется путём построения последовательных приближений и основан на принципах [простой итерации](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BF%D1%80%D0%BE%D1%81%D1%82%D0%BE%D0%B9_%D0%B8%D1%82%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%B8). Метод обладает квадратичной [сходимостью](https://ru.wikipedia.org/wiki/%D0%A1%D0%BA%D0%BE%D1%80%D0%BE%D1%81%D1%82%D1%8C_%D1%81%D1%85%D0%BE%D0%B4%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8). Модификацией метода является [метод хорд и касательных](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D1%85%D0%BE%D1%80%D0%B4_%D0%B8_%D0%BA%D0%B0%D1%81%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D1%85)**.** Применятся для выпуклых функций

Пусть Разложим функцию в ряд Тейлора:



Воспользуемся теоремой Ферма и найдем 



Выразим :



Тогда итерационный процесс имеет вид:



Сходимость данного метода зависит от начального приближения

Реализован программно следующим образом:

def newton(xk, xk1, n):  
 data2.append([n, xk, xk1])  
 if abs(F(xk1) - F(xk)) < eps: return  
 newton(xk1, xk - dF(xk) / ddF(xk1), n)

Программа находит решение нашей задачи менее чем за 10-5 секунд и всего за 3 итерации.

График сходимости представлен на графике 4.

График 4

Полученное значение: Хmin =3,8228756, почти не отличается от результата аналитического решения.

**Метод касательных**

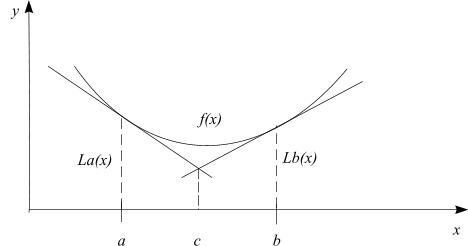
Если функция *f(x)* выпукла и дифференцируема на *[a,b]*, можно использовать метода касательных. Идея метода состоит в следующем. Пусть *[a,b]* - отрезок неопределённости и *f(a),f’(a),f(b),f’(b)* ­– результаты вычислений в точках *a* и *b*. По этой информации строится аппроксимирующая функция, представляющая из себя кусочно-линейную функцию, состоящую из касательной *La(x)=f(a)+f’(a)(x-a)* к *f(x)* в точке *a* и касательной *Lb(x)=f(b)+f’(b)(x-b)* к *f(x)* в точке *b*.

Рисунок 1

Полученная аппроксимирующая функция есть ломанная состоящая из прямой  на ** и  на  где *c* – точка пересечения касательных. Значение точки пересечения *с* можно определить по формуле:



В точке *с* производятся вычисления *f(c)* и *f’(c)*. Если *f’(c)=0*, то решением задачи будет *x\*=c*. Если же *f’(c)>0*, то в качестве следующего отрезка неопределённости будет *[a,c]*. Если же *f’(c)<0*, то – отрезок *[c,b]*. Процесс повторяется до тех пор, пока *f’(c)=0* или отрезок неопределённости не достигнет заданной точности.

На языке программирования метод реализован следующим образом:

def tangents(a, b, n):  
 data3.append([n, a, b])  
 if (abs(dF(a)) < eps or abs(dF(b)) < eps): return  
 x = (b \* dF(b) - a \* dF(a) - (F(b) - F(a))) / (dF(b) - dF(a))  
 if dF(x) < 0:  
 tangents(x, b, n + 1)  
 else:  
 tangents(a, x, n + 1)

Полученное решение: Хmin=3,8228756, отличие от аналитического решения ≈4\*10-5. Программе понадобилось 14 итераций и, в среднем, 0,001 секунды процессорного времени. На графике 5 отражена пошаговая тенденция сходимости.

График 5

**Выводы**

В ходе данной работы были проведены исследования трёх итерационных методов поиска решения нелинейного уравнения: метода золотого сечения, метода Ньютона и метода касательных. Метод золотого сечения оказался медленные остальных, так как для нахождения решения этим методом понадобилось наибольшее количество итераций. Однако, в отличие от второго по скорости метода касательных, он сходится равномерно. А самое главное его преимущество в том, что для решения этим методом не надо требовать от функции дифференцируемости, но необходима унимодальность. Самым эффективным оказался метод Ньютона – всего 3 итерации понадобилось для нахождения решения, но его главным недостатком является то, что функция должна быть дважды непрерывно дифференцируема.