Министерство образования и науки Российской Федерации Федеральное агентство по образованию

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Отчет по лабораторной работе №1 дисциплины

«Вычислительная математика»

Выполнил студент группы ИВТ-22 /Крючков И. С/ Проверил /Исупов К. С./

Киров 2021

# Задание:

1. Построить график функции f(x) и отделить один из корней уравнения: f(x).
2. Сузить интервал изоляции корня, если необходимо, проверив условие M <= 2m.
3. Уточнить корень с погрешностью e <= 0.00001 двумя численными методами: комбинированным методом и методом итераций.
4. Проверить полученное значение корня, используя систему Mathcad.

Вариант 10

Уравнение: x^3 – sin(x) = 0

Интервал: [0.1, 1.0]

**Теоретические сведения:**

Теоретические сведения об уточнении корней комбинированным методом.

Основная суть метода — сузить интервал изоляции с двух сторон. В зависимости от неподвижного конца интервала изоляции используются две пары формул, и на каждом шаге вычисляются приближенное значение корня по недостатку xk+1 и по избытку k+1.

Если неподвижна точка *a*:

* значение по недостатку вычисляется методом касательных:

* Значение по избытку вычисляется методом хорд:

Если неподвижна точка *b*:

* значение по недостатку вычисляется методом хорд:

* Значение по избытку вычисляется методом касательных:

На каждом новом шаге расчета используют суженный с двух сторон интервал [,].

Теоретические сведения об уточнении корней методом простых итераций.

Вещественное число ξ называется неподвижной точкой функции φ, если φ(ξ) = ξ.

Алгоритм:

1. Преобразуем уравнение к каноническому виду: f(x) = 0 → x = φ(x).
2. Выберем начальное приближение x0 — любую точку из интервала [a, b].
3. Итерационный процесс осуществляем в соответствии с рекуррентным соотношением: xn+1 = φ(xn), n = 1, 2, 3, . . .

Итерационный процесс сходится при условии, что первая производная итерационной функции φ(x) по модулю меньше единицы: |φ 0 (x)| < 1.

Если условие |φ’(x)| < 1 выполняется, то на отрезке [a, b], на котором локализован корень, в качестве начального приближения можно взять любую точку x0 ∈ [a, b]. Скорость сходимости зависит от абсолютной величины производной: чем меньше |φ’ (x)|, тем быстрее сходится процесс, что непосредственно следует из формулы (18): εn+1 = εnφ 0 (ξ).

Тип сходимости и критерий останова:

* Двусторонняя сходимость:

Если φ’(x) < 0, то сходимость двусторонняя.

Критерий окончания итерационного процесса |xn+1 – xn| ≤ ε является объективным.

* Монотонная сходимость:

Если φ’(x) > 0, то сходимость односторонняя.

Критерий окончания итерационного процесса:

, где q = max| φ’(x)| на интервале [a, b]

Наибольшая скорость сходимости итерационного процесса:

Xn+1 = φ(xn), n = 1, 2, . . .

достигается при

φ’(x) = 1 + k · f’(x) = 0

Этого можно добиться, если выбрать параметр k в уравнении φ(x) = x + k · f(x) зависящим от x в виде

k =

При этом итерационная формула переходит в формулу метода касательных Ньютона:

**Практическая часть**

Исходная функция:

Первая производная:

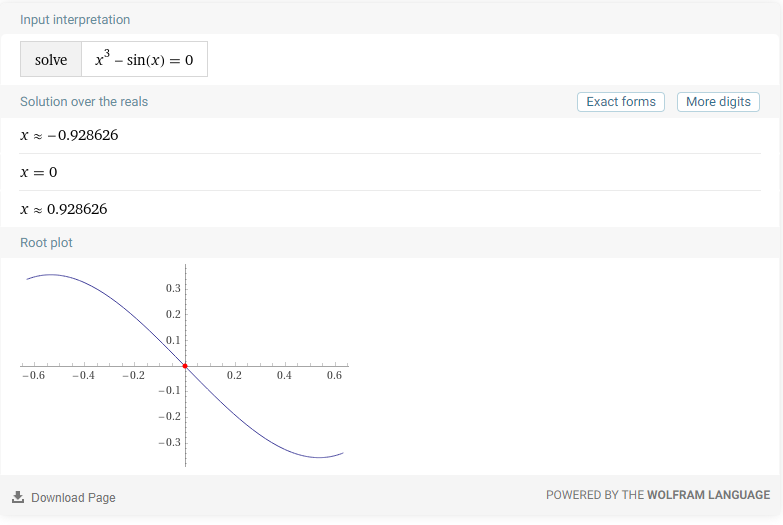
Вторая производная:

Канонический вид:

Ответ комбинированного метода: 0.928626 ± 0,00001

Ответ метода итераций: 0.928626 ± 0,00001

Проверка в системе wolframalpha:



**Экранные формы:**

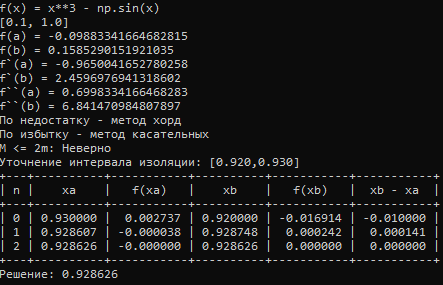
****

Рисунок 1 Комбинированный метод

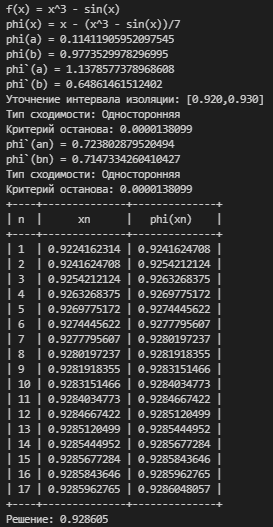


Рисунок 2 Метод простых итераций

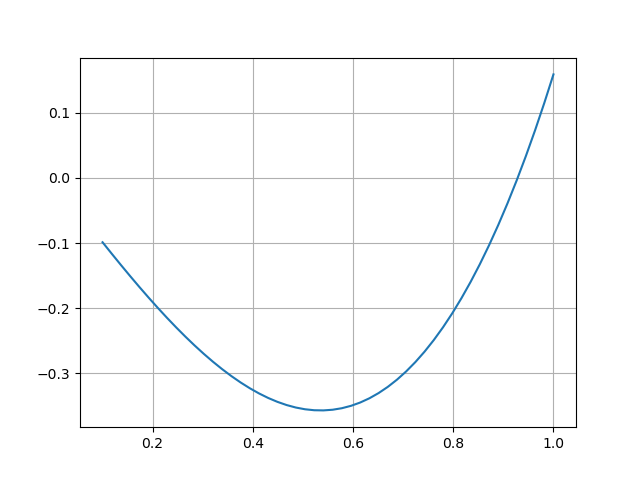
****

Рисунок 3 График функции

**Листинг кода:**

import numpy as np

import matplotlib.pyplot as plt

from prettytable import PrettyTable

fig, ax = plt.subplots()

ax.grid()

# интервал

a, b = 0.1, 1.0

# погрешность

eps = 0.00005

# функция

pf = "x^3 - sin(x)"

f = lambda x: x\*\*3 - np.sin(x)

# первая производная

df = lambda x: 3 \* x\*\*2 - np.cos(x)

#вторая производная

ddf = lambda x: 6 \* x + np.sin(x)

k = 7

#phi(x)

pphi = "x - (x^3 - sin(x))/"+str(k)

phi = lambda x: x - ((x\*\*3 - np.sin(x))/k)

#первая производная phi(x)

dphi = lambda x: 1-(1/k)\*df(x)

pdphi = "1 - 1/"+str(k)+" \* 3x^2 - cos(x)"

res = a

x = np.linspace(a, b)

def combined\_method(left, right):

def hords(xa, xb):

return xb - f(xb) \* (xb - xa) / (f(xb) - f(xa))

def casat(xa):

return xa - f(xa) / df(xa)

fixed = f(left) \* ddf(left) > 0

if fixed:

xa, xb = left, right

else:

xa, xb = right, left

i = 0

yield i, xa, f(xa), xb, f(xb), xb - xa

while abs(xb - xa) > eps and i < 50:

i += 1

#Если неподвижна A

if fixed:

xb = hords(xa, xb)

xa = casat(xa)

else:

xa = hords(xb, xa)

xb = casat(xb)

yield i, xa, f(xa), xb, f(xb), xb - xa

def iter\_method(left, right, er):

i = 0

xn = left

xn1 = right

delta = xn-xn1

while abs(delta) > er and i < 50:

i += 1

xn1 = phi(xn)

delta = xn-xn1

xn = xn1

yield i, xn, phi(xn)

def checkform(left):

if f(left) \* ddf(left) > 0:

r1 = 'По недостатку - метод касательных'

r2 = 'По избытку - метод хорд'

else:

r1 = 'По недостатку - метод хорд'

r2 = 'По избытку - метод касательных'

return r1, r2

def isolate(left, right, acc):

while f(left + acc) \* f(right - acc) < 0:

left += acc

right -= acc

while f(left) \* f(left + acc) > 0:

left += acc

while f(right) \* f(right - acc) > 0:

right -= acc

return left, right

def combined():

print(f"f(x) = {pf}")

print(f"[{a}, {b}]")

print(f"f(a) = {f(a)}")

print(f"f(b) = {f(b)}")

print(f"f`(a) = {df(a)}")

print(f"f`(b) = {df(b)}")

print(f"f``(a) = {ddf(a)}")

print(f"f``(b) = {ddf(b)}")

r1, r2 = checkform(a)

print(r1)

print(r2)

mm = abs(ddf(b)) <= abs(2\*df(a))

print(f"M <= 2m: {'верно' if mm else 'Неверно'}")

an, bn = isolate(a,b, 0.01)

print("Уточнение интервала изоляции: [{:.3f},{:.3f}]".format(an, bn))

tablex = PrettyTable()

for t in combined\_method(an,bn):

# print('{}\t{:.15f}\t{:.15f}\t{:.15f}\t{:.15f}\t{:.15f}'.format(\*t))

tablex.field\_names = ["n", "xa", "f(xa)", "xb","f(xb)", "xb - xa"]

tablex.add\_row(t)

tablex.float\_format = '.6'

res = t[1]

print(tablex)

print(f'Решение: {res:.6f}')

def iteration():

print(f"f(x) = {pf}")

print(f"f(a) = {f(a)}")

print(f"f(b) = {f(b)}")

print(f"f`(a) = {df(a)}")

print(f"f`(b) = {df(b)}")

print(f"phi(x) = {pphi}")

print(f"phi(a) = {phi(a)}")

print(f"phi(b) = {phi(b)}")

print(f"phi`(a) = {dphi(a)}")

print(f"phi`(b) = {dphi(b)}")

an, bn = isolate(a,b, 0.01)

print("Уточнение интервала изоляции: [{:.3f},{:.3f}]".format(an, bn))

if dphi(an) < 0 and dphi(bn) < 0:

ts = "Двусторонняя"

e = eps

else:

ts = "Односторонняя"

e = abs(1 - max(abs(dphi(an)), abs(dphi(bn)))) \* eps

print(f"Тип сходимости: {ts}")

print(f"Критерий останова: {e:.10f}")

print(f"phi`(an) = {dphi(an)}")

print(f"phi`(bn) = {dphi(bn)}")

tablex = PrettyTable()

for t in iter\_method(an,bn,e):

# print('{}\t{:.15f}\t{:.15f}'.format(\*t))

tablex.field\_names = ["n", "xn", "phi(xn)"]

tablex.add\_row(t)

tablex.float\_format = '.10'

res = t[2]

print(tablex)

print(f'Решение: {res:.6f}')

# combined()

iteration()

plt.plot(x, f(x))

plt.show()

**Вывод:** в ходе выполнения лабораторной работы был закреплен лекционный материал по теме «Численные методы решения нелинейных уравнений». Закреплены на практике численные методы решение нелинейных уравнений: комбинационный метод и итерационный метод, а так же изучены особенности их применения.