Санкт-Петербургский Государственный Электротехнический Университет "ЛЭТИ"

кафедра физики

Задание №1 по дисциплине

"Физические основы информационных технологий"

Название: Фазовые кривые

Фамилия И.О.: Виноградов К.А.

группа: 6304

Преподаватель: Альтмарк А.М.

Итоговый балл:

Санкт-Петербург 2018

**Цель работы.**

Определить фазовые и групповые скорости для волновода с определенными параметрами.

**Задание.**

Построить серию из 10 зависимостей фазовой скорости от частоты, а также серию из 10 зависимостей групповой скорости от частоты для цилиндрического волновода, заполненного диэлектриком.

Вариант 19

Данные: R = 0.01 м, eps = 5.4 Ф/м

**Ход работы.**

Для начала нам нужно вручную подобрать фазовую частоту при которой функция Бесселя имеет ровно 10 корней. Для этого в формуле фазовой частоты ω = 2 \* Pi \* 109 \* coef нужно подобрать переменную coef. При значении больше 64 функция имеет 10 корней, возьмем значение 67.

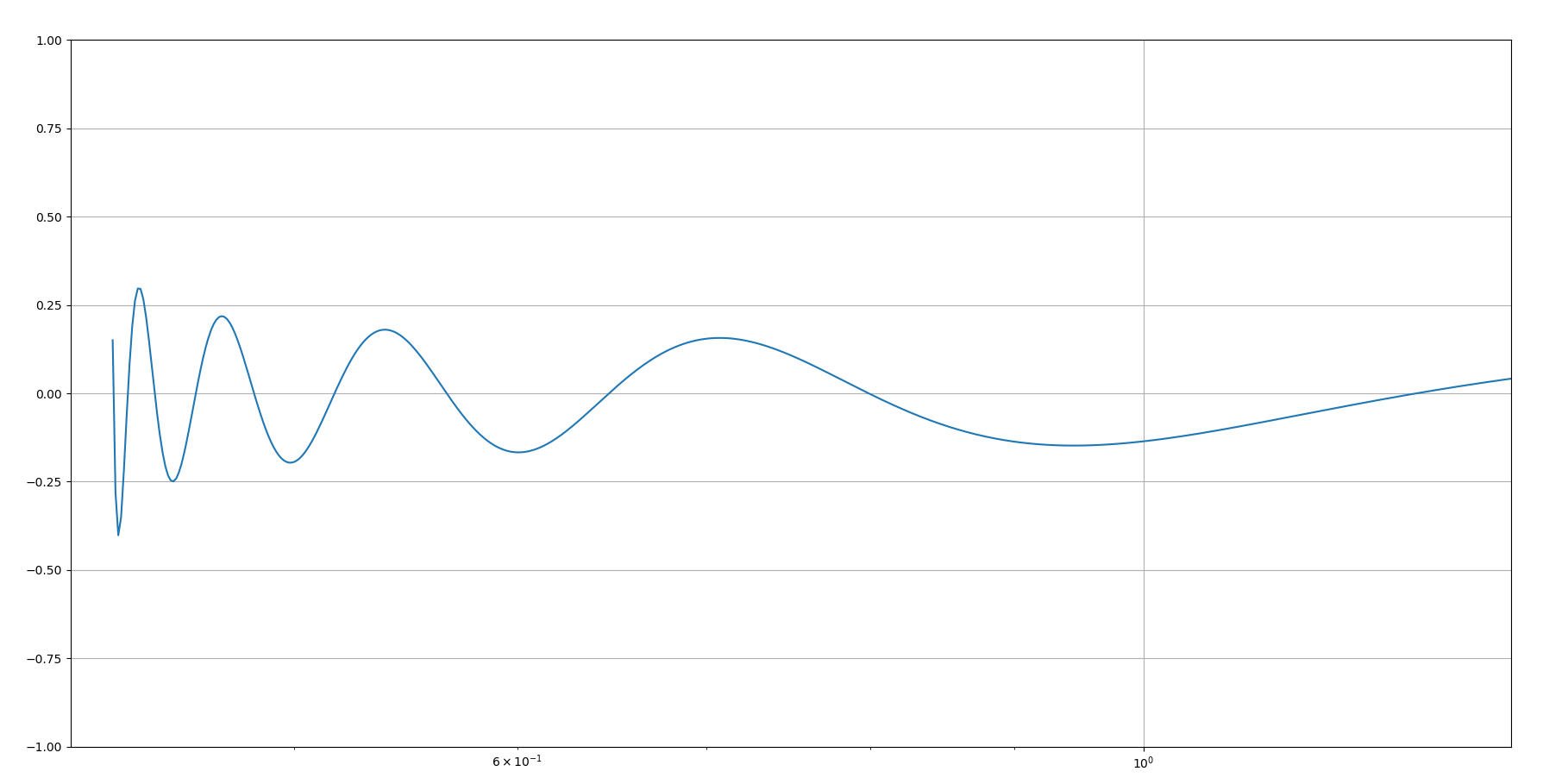


Рисунок 1 - Функция Бесселя

Теперь приступим к нахождению непосредственно фазовых скоростей. Зададим последовательность частот, относительно которых будем рассчитывать скорость. Зададим 10000 точек для построения графика. Далее установим интервалы, на которых меняется знак и применим метод бисекции для нахождения корней.

Имея корни, рассчитаем групповые скорости в зависимости от частоты и построим график для каждого из корней Бесселя. Вместе с этим также рассчитаем и групповые скорости и нанесем их на этот же график. Результаты вычислений представлены на рис. 2.

Для написания работы использовался язык Python версии 3.7 с некоторыми стандартными библиотеками. Код программы представлен в приложении А.

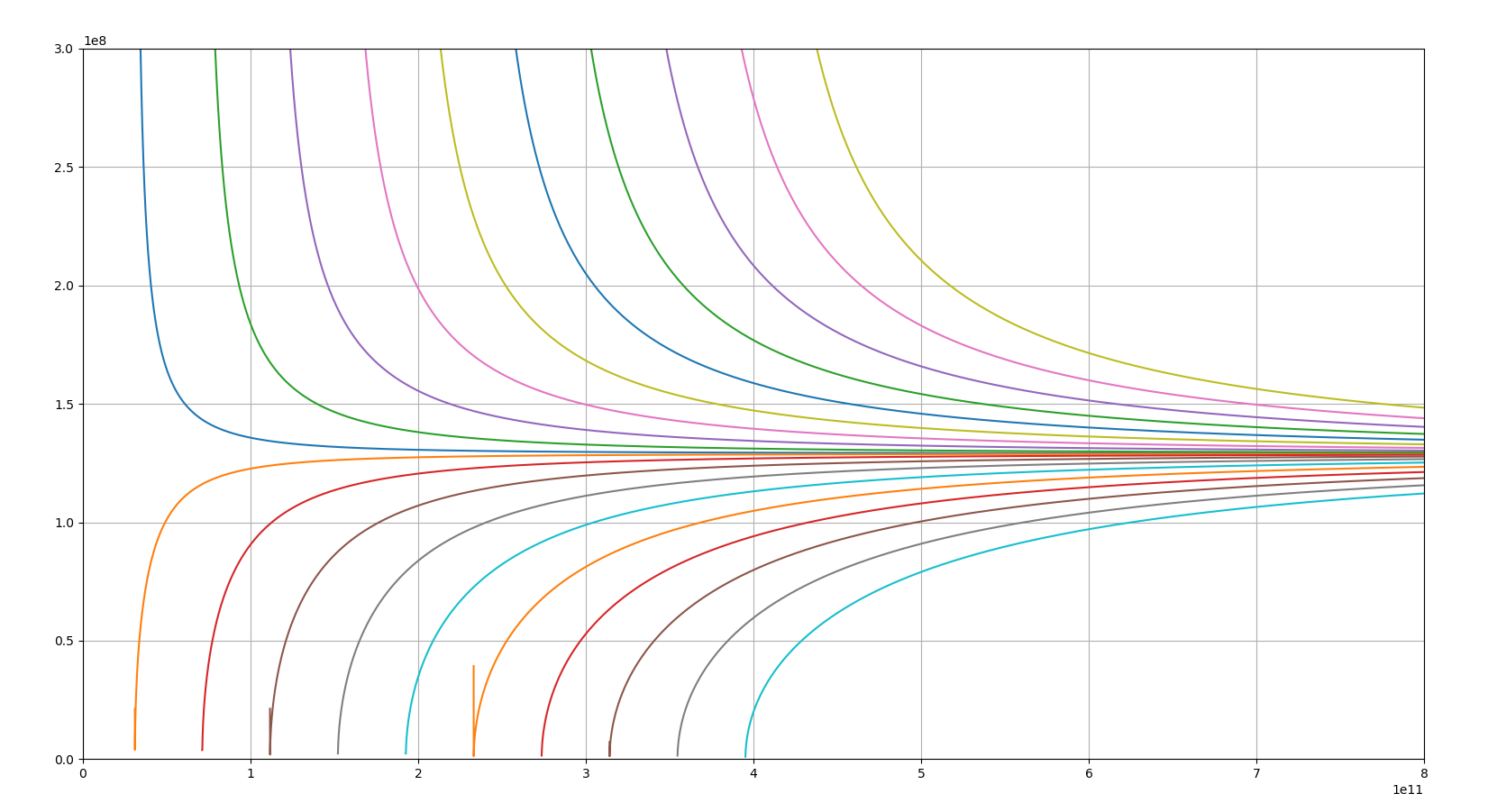


Рисунок 2 – Фазовые кривые

**Выводы.**

В ходе данной работы были изучены понятия фазовой и групповой скорости. На практике освоены методы их вычисления с применением найденных корней функции Бесселя.

**ПРИЛОЖЕНИЕ А**

**КОД ПРОГРАММЫ**

from matplotlib import pyplot  
from matplotlib import mlab  
from scipy import special  
from scipy import optimize  
from scipy import constants  
import math  
import numpy  
  
  
R = 0.01  
eps = 5.4  
omg\_param = 2 \* constants.pi \* 10 \*\* 9 \* 67  
  
  
def get\_intervals(rel\_spd\_rng\_loc, bs\_rng\_loc, limit=0):  
 prev = bs\_rng\_loc[0]  
 intervals = []  
 for i in range(1, len(bs\_rng\_loc)):  
 if bs\_rng\_loc[i] \* prev < 0:  
 intervals.append([rel\_spd\_rng\_loc[i - 1], rel\_spd\_rng\_loc[i]])  
 if 0 < limit <= len(intervals):  
 break  
 prev = bs\_rng\_loc[i]  
 return intervals  
  
  
def phase\_spd\_func(omg\_loc, bs\_root\_loc):  
 temp = omg\_loc \*\* 2 \* eps / constants.speed\_of\_light \*\* 2 - bs\_root\_loc \*\* 2 / R \*\* 2  
 if temp <= 0:  
 return math.inf  
 else:  
 return omg\_loc / math.sqrt(temp)  
  
  
def group\_spd\_func(omg\_loc, phase\_spd\_loc):  
 if phase\_spd\_loc[0] \* phase\_spd\_loc[1] != 0:  
 temp = omg\_loc[1] / phase\_spd\_loc[1] - omg\_loc[0] / phase\_spd\_loc[0]  
 if temp != 0:  
 return (omg\_loc[1] - omg\_loc[0]) / temp  
 else:  
 return math.inf  
  
  
def kp\_func(rel\_spd\_loc, omg\_loc):  
 temp = eps - 1 / rel\_spd\_loc \*\* 2  
 if temp <= 0:  
 return math.inf  
 else:  
 return omg\_loc / constants.speed\_of\_light \* math.sqrt(temp)  
  
  
def bs\_func(rel\_spd\_loc, omg\_loc=omg\_param):  
 return special.j0(kp\_func(rel\_spd\_loc, omg\_loc) \* R)  
  
  
def check\_omg\_func():  
 i\_rng = range(0, 1000)  
 rel\_spd\_loc = [1 / math.sqrt(eps) + 4 / 1000 \* i for i in i\_rng]  
 bessel\_rng = [bs\_func(i, omg\_param) for i in rel\_spd\_loc]  
 root\_intervals = get\_intervals(rel\_spd\_loc, bessel\_rng)  
 rel\_spd\_roots\_loc = [optimize.bisect(bs\_func, i[0], i[1]) for i in root\_intervals]  
 return rel\_spd\_roots\_loc  
  
  
rel\_spd\_rng = numpy.geomspace(1, 1000, 10000)  
rel\_spd\_rng = [i - 1 for i in rel\_spd\_rng]  
bs\_rng = [bs\_func(i, omg\_param) for i in rel\_spd\_rng]  
  
  
rel\_spd\_roots = check\_omg\_func()  
axes = pyplot.figure().add\_axes([0.08, 0.1, 0.87, 0.8])  
axes.plot(rel\_spd\_rng, bs\_rng)  
axes.set\_ylim(-1, 1)  
axes.set\_xlim(rel\_spd\_roots[0] - 0.02, rel\_spd\_roots[-1] + 0.1)  
axes.set\_xscale('log', basex=10)  
axes.grid()  
pyplot.show()  
  
  
sup\_range = mlab.frange(0.0, 5000.0, 0.1)  
sup\_results = [special.j0(i) for i in sup\_range]  
sup\_intervals = get\_intervals(sup\_range, sup\_results, 10)  
sup\_roots = [optimize.bisect(special.j0, i[0], i[1]) for i in sup\_intervals]  
omg\_rng = mlab.frange(omg\_param / 1000, omg\_param \* 2, omg\_param / 10000)  
  
  
for root in sup\_roots:  
 pyplot.plot(omg\_rng, [phase\_spd\_func(omg, root) for omg in omg\_rng])  
 pyplot.plot(omg\_rng[:-1], [group\_spd\_func([omg\_rng[i], omg\_rng[i + 1]], [phase\_spd\_func(omg\_rng[i], root), phase\_spd\_func(omg\_rng[i + 1], root)]) for i in range(len(omg\_rng) - 1)])  
  
  
pyplot.xlim(0, 8e11)  
pyplot.ylim(0, 3e8)  
pyplot.grid()  
pyplot.show()