Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

(ФГАОУ ВО СПбПУ)

Институт компьютерных наук и кибербезопасности

Высшая школа технологий искусственного интеллекта

**ЛАБОРАТОРНАЯ РАБОТА №12**

**"Фильтры"**

по дисциплине «Методы обработки экспериментальных данных»

Выполнил   
студент гр.5140203/30102 М. С. Антоненко

Преподаватель И. Н. Белых

Санкт-Петербург 2023

**Задание на лабораторную работу:**

1) В классе Model реализовать функцию дискретной свертки y = convolModel(x, N, h, M,…) заданных функций 𝑥(𝑡) = {𝑥𝑘} и ℎ(𝑡) = {ℎ𝑘}

2) На основе понятий линейных систем реализовать первое приближение модели кардиограммы длительностью 4 сек с помощью функции свертки convolModel() импульсной реакции модели сердечной мышцы и управляющей функции ритма, для этого:

а) импульсную реакцию модели сердечной мышцы реализовать с помощью функции мультипликативной модели h(t)=multModel(h1, h2, M,…) гармонического процесса harm() и нисходящего экспоненциального тренда trend()

б) управляющую функцию ритма 𝑥(𝑡) задать в виде четырех импульсов минимальной длительности с амплитудами 1 ± 0.1, следующих через равные интервалы времени, т.е. в виде массива данных длины N=1000, состоящего из нулей за исключением 4-х значений на позициях кратных 200, что при шаге dt=0.005 [сек] равно 1 сек.

в) Отобразить графики трех функций в разных окнах.

3) Используя функцию x(t)=spikes() и функцию h(t) из п.2а, а также функцию свертки из п.1 смоделировать и отобразить «патологическую» кардиограмму.

**Алгоритм программы**

|  |  |
| --- | --- |
| № | model.py |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65 | @staticmethod  def lpf(fc, m, dt):  d = [0.35577019, 0.2436983, 0.07211497, 0.00630165]  fact = 2 \* fc \* dt  lpw = []  lpw.append(fact)  arg = fact \* math.pi  for i in range(1, m + 1):  lpw.append(np.sin(arg \* i) / (math.pi \* i))  lpw[m] = lpw[m] / 2  sumg = lpw[0]  for i in range(1, m + 1):  sum = d[0]  arg = math.pi \* i / m  for k in range(1, 4):  sum += 2 \* d[k] \* np.cos(arg \* k)  lpw[i] = lpw[i] \* sum  sumg += 2 \* lpw[i]  for i in range(m + 1):  lpw[i] = lpw[i] / sumg  return lpw  @staticmethod  def lpf\_reverse(lpw):  reverse = []  for i in range(len(lpw) - 1, 0, -1):  reverse.append(lpw[i])  reverse.extend(lpw)  return reverse  @staticmethod  def hpf(fc, m, dt):  lpw = Proccessing.lpf\_reverse(Proccessing.lpf(fc, m, dt))  hpw = []  Loper = 2 \* m + 1  for k in range(Loper):  if k == m:  hpw.append(1 - lpw[k])  else:  hpw.append(- lpw[k])  return hpw  @staticmethod  def bpf(fc1, fc2, m, dt):  lpw1 = Proccessing.lpf\_reverse(Proccessing.lpf(fc1, m, dt))  lpw2 = Proccessing.lpf\_reverse(Proccessing.lpf(fc2, m, dt))  bpw = []  Loper = 2 \* m + 1  for k in range(Loper):  bpw.append(lpw2[k] - lpw1[k])  return bpw  @staticmethod  def bsf(fc1, fc2, m, dt):  bsw = []  lpw1 = Proccessing.lpf\_reverse(Proccessing.lpf(fc1, m, dt))  lpw2 = Proccessing.lpf\_reverse(Proccessing.lpf(fc2, m, dt))  Loper = 2 \* m + 1  for k in range(0, Loper):  if k == m:  bsw.append(1. + lpw1[k] - lpw2[k])  else:  bsw.append(lpw1[k] - lpw2[k])  return bsw |
|  | main.py |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34 | import matplotlib.pyplot as plt  from classes.proccessing import Proccessing  plt.rcParams["figure.figsize"] = [20, 7.5]  plt.rcParams["figure.autolayout"] = True  def main():  fc = 50  fc1 = 35  fc2 = 75  m = 64  dt = 0.002  low\_profile\_filter = Proccessing.lpf(fc, m, dt)  buf = low\_profile\_filter[::-1]  result = buf + low\_profile\_filter  high\_profile\_filter = Proccessing.hpf(fc, m, dt)  band\_pass\_filter = Proccessing.bpf(fc1, fc2, m, dt)  band\_stop\_filter = Proccessing.bsf(fc1, fc2, m, dt)  fig, ax = plt.subplots(nrows=5, ncols=1)  fig.suptitle("Задание 12", fontsize=15)  ax[0].plot(low\_profile\_filter)  ax[1].plot(result)  ax[2].plot(high\_profile\_filter)  ax[3].plot(band\_pass\_filter)  ax[4].plot(band\_stop\_filter)  plt.show()  main() |

**Полученные результаты**

На рисунках 1-2 приведены результаты работы программы в виде пяти графиков, представленных весовыми функциями ФНЧ, ФВЧ, ПФ и РФ соответственно.

Изображение выглядит как текст, снимок экрана, линия, График

Автоматически созданное описание

Рисунок 1 ‒ Весовая функция ФНЧ

Изображение выглядит как текст, линия, диаграмма, График

Автоматически созданное описание

Рисунок 2 ‒ Весовые функции ФВЧ, ПФ, РФ

Изображение выглядит как текст, диаграмма, линия, Параллельный

Автоматически созданное описание

Рис ‒ Импульсные реакции фильтров Поттера

Изображение выглядит как диаграмма, текст, Технический чертеж, План

Автоматически созданное описание

Рис ‒ АЧХ фильтров Поттера

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

(ФГАОУ ВО СПбПУ)

Институт компьютерных наук и кибербезопасности

Высшая школа технологий искусственного интеллекта

**ЛАБОРАТОРНАЯ РАБОТА №13**

**"функции antiShift и antiSpike"**

по дисциплине «Методы обработки экспериментальных данных»

Выполнил   
студент гр.5140203/30102 М. С. Антоненко

Преподаватель И. Н. Белых

Санкт-Петербург 2023