ГУАП

КАФЕДРА № 42

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ПРЕПОДАВАТЕЛЬ

| Ассистент |  |  |  | Н.А. Янковский |
| --- | --- | --- | --- | --- |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №7 |
| --- |
| **БИХ фильтры**  Вариант 5 |
|  |
| по курсу: Цифровая обработка и передача сигналов |
|  |

РАБОТУ ВЫПОЛНИЛ

| СТУДЕНТ ГР. № | 4128 |  |  |  | В. А. Воробьев |
| --- | --- | --- | --- | --- | --- |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023

**1 Задание**

*f* = 3*N*, *T* = 10/*F*, где *N* – номер по списку.

Написать программу, которая позволит:

1. Провести дискретизацию функции *u(t)* = sin(2π*ft*) на заданном интервале с частотой дискретизации 15*f*.
2. Добавить к полученным дискретным отсчетам *u1, ..., un* выборку случайной величины *x1, ..., xn,* ***X~N*(0, 0.2)**.
3. Реализовать фильтр Баттерворта 2 и 3 порядка для полученного дискретного сигнала следующими способами:
4. Фильтр нижних частот
5. Фильтр верхних частот
6. Полосовой фильтр
7. Заграждающий фильтр

**2 Результат работы**



Рисунок 1 – Графики

**3 Вывод**

В ходе лабораторной работы мы успешно реализовали программу, выполняющую обработку сигналов с использованием БИХ фильтров. Исходный код был выложен на GitHub (URL: <https://github.com/vladcto/suai-labs/tree/3ef08e4ff73b055c86f994f5262f9e39ea10cc4c/5_semester/%D0%A6%D0%9E%D0%9F%D0%A1/lab7>) В рамках задания мы провели дискретизацию заданной функции и добавили случайные компоненты к полученным дискретным отсчетам. Затем мы реализовали фильтры Баттерворта второго и третьего порядка, включая фильтры нижних и верхних частот, полосовой и заграждающий фильтры.

Эти действия позволили нам изучить воздействие фильтров на сигнал в различных частотных диапазонах и оценить их эффективность в различных сценариях использования. Лабораторная работа позволила углубить наши знания в области БИХ фильтров и приобрести опыт в обработке дискретных сигналов.

**ПРИЛОЖЕНИЕ**

preview\_app.dart

import 'package:extend\_math/extend\_math.dart';

import 'package:flutter/material.dart';

import 'package:lab6/logic/calculations.dart';

import 'package:ui\_kit/ui\_kit.dart';

class PreviewApp extends StatelessWidget {

const PreviewApp({super.key});

@override

Widget build(BuildContext context) {

return Column(

children: [

Expanded(

child: KitTitleContainer(

title: "Исходный сигнал с шумом",

child: KitLineChart(

lines: [

KitLineData(

dots: Calculations.noisedDots

.map((e) => KitDot(e.x, e.y))

.toList(),

),

],

),

),

),

Expanded(

flex: 3,

child: Row(

children: [

Expanded(

child: KitColumn(

childFit: FlexFit.tight,

children: [

\_PreviewFilter(

title: "Медианный фильтр 4-го порядка",

Calculations.medianFiltered4,

Calculations.median4Diff,

),

\_PreviewFilter(

title: "Медианный фильтр 6-го порядка",

Calculations.medianFiltered6,

Calculations.median6Diff,

),

\_PreviewFilter(

title: "Медианный фильтр 8-го порядка",

Calculations.medianFiltered8,

Calculations.median6Diff,

),

],

),

),

Expanded(

child: KitColumn(

childFit: FlexFit.tight,

children: [

\_PreviewFilter(

title: "Скользящее среднее 4-го порядка",

Calculations.movingFiltered4,

Calculations.avg4Diff,

),

\_PreviewFilter(

title: "Скользящее среднее 6-го порядка",

Calculations.movingFiltered6,

Calculations.avg6Diff,

),

\_PreviewFilter(

title: "Скользящее среднее 8-го порядка",

Calculations.movingFiltered8,

Calculations.avg8Diff,

),

],

),

),

],

),

),

],

);

}

}

class \_PreviewFilter extends StatelessWidget {

final String title;

final List<Point2> points;

final double diff;

const \_PreviewFilter(

this.points,

this.diff, {

required this.title,

});

@override

Widget build(BuildContext context) {

return KitTitleContainer(

title: title,

child: KitLineChart(

xAxisName: "СКО: ${diff.toStringAsFixed(5)}",

lines: [

KitLineData(

color: Colors.deepOrange,

dots: Calculations.dots.map((e) => KitDot(e.x, e.y)).toList(),

),

KitLineData(

dots: points.map((e) => KitDot(e.x, e.y)).toList(),

),

],

),

);

}

}

variant.dart

import 'package:extend\_math/extend\_math.dart';

import 'dart:math';

abstract final class Variant {

static const \_n = 5;

static const fParam = 3 \* \_n;

static const T = 10 / fParam;

static const step = 1 / 15.0 / fParam;

static const interval = MathInterval(-T, T);

static double fn(double x) => sin(2 \* pi \* fParam \* x);

}

calculations.dart

import 'dart:math';

import 'package:extend\_math/extend\_math.dart';

import 'package:lab6/logic/variant.dart';

abstract final class Calculations {

static final random = Random();

static final dots = Variant.interval.applyFx(Variant.fn, step: Variant.step);

static final noisedDots =

dots.map((e) => Point2(e.x, e.y + random.nextDouble() \* 0.8 - 0.4)).toList();

static final xDots = noisedDots.xDots;

// Filtered

static final medianFiltered4 =

FiltersList.medianFilter(noisedDots.yDots, 3).joinX(xDots);

static final medianFiltered6 =

FiltersList.medianFilter(noisedDots.yDots, 5).joinX(xDots);

static final medianFiltered8 =

FiltersList.medianFilter(noisedDots.yDots, 7).joinX(xDots);

static final movingFiltered4 =

FiltersList.movingAverageFilter(noisedDots.yDots, 3).joinX(xDots);

static final movingFiltered6 =

FiltersList.movingAverageFilter(noisedDots.yDots, 5).joinX(xDots);

static final movingFiltered8 =

FiltersList.movingAverageFilter(noisedDots.yDots, 7).joinX(xDots);

// Diff

static final origDiff = MathList.calculateRMSE(dots.yDots, noisedDots.yDots);

static final median4Diff =

MathList.calculateRMSE(dots.yDots, medianFiltered4.yDots);

static final median6Diff =

MathList.calculateRMSE(dots.yDots, medianFiltered6.yDots);

static final median8Diff =

MathList.calculateRMSE(dots.yDots, medianFiltered8.yDots);

static final avg4Diff =

MathList.calculateRMSE(dots.yDots, movingFiltered4.yDots);

static final avg6Diff =

MathList.calculateRMSE(dots.yDots, movingFiltered6.yDots);

static final avg8Diff =

MathList.calculateRMSE(dots.yDots, movingFiltered8.yDots);

}

main.dart

import 'package:flutter/material.dart';

import 'package:lab6/ui/preview\_app.dart';

void main() {

runApp(const MyApp());

}

class MyApp extends StatelessWidget {

const MyApp({super.key});

// This widget is the root of your application.

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Flutter Demo',

theme: ThemeData(

colorScheme: ColorScheme.fromSeed(seedColor: Colors.blue),

useMaterial3: true,

),

home: const Scaffold(

body: PreviewApp(),

),

);

}

}

web\_plugin\_registrant.dart

// Flutter web plugin registrant file.

//

// Generated file. Do not edit.

//

// ignore\_for\_file: type=lint

void registerPlugins() {}