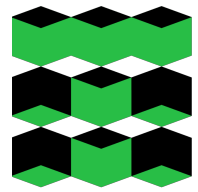
**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное учреждение

высшего образования



**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Инженерная школа информационных технологий и робототехники

Отделение информационных технологий

Направление подготовки 09.04.04 Программная инженерия

**Отчёт по лабораторной работе №1**

**МЕТОДЫ ФИЛЬТРАЦИИ СИГНАЛОВ**

по дисциплине Основы теории управления автономными системами

Выполнил студент гр. 8ПМ4Л \_\_\_\_\_\_\_ \_\_\_\_\_\_ Сокуров Р.Е.

Подпись Дата Фамилия И.О.

Проверил доцент ОАР \_\_\_\_\_\_\_ \_\_\_\_\_\_ Хожаев И.В.

Подпись Дата Фамилия И.О.

Томск 2024 г.

**Цель**

Изучить основные методы фильтрации измерительных сигналов.

**Задачи**

1) Изучить принцип работы следующих фильтров:

– фильтра экспоненциального сглаживания;

– фильтра скользящего среднего;

– медианного фильтра;

– фильтра с ограничением скорости нарастания сигнала;

2) Сгенерировать полезный сигнал и добавить к нему равномерно распределенный шум и всплески большой амплитуды;

3) Реализовать каждый из ранее перечисленных фильтров любым известным способом, проверить работоспособность фильтров на сгенерированном ранее зашумленном сигнале и изучить влияние настроечного параметра фильтра на качество обработки сигнала;

4) Подобрать комбинацию фильтров, отделяющих полезный сигнал от шума обоих типов; сравнить исходный полезный сигнал и отфильтрованный;

5) Оформить отчет.

**Ход работы**

**1. Генерация полезного сигнала и добавление к нему равномерно распределённого шума и всплесков большой амплитуды.**

В качестве генерируемого сигнала была выбрана функция . Данная функция была рассчитана на массиве , состоящем из 800 элементов. Затем, с помощью библиотеки «NumPy» языка «Python» был добавлен равномерно распределённый шум, а также высокоамплитудные случайные всплески (Листинг 1).

Листинг 1 – Создание зашумлённого сигнала

1. x = np.linspace(-4\*np.pi, 4\*np.pi, 800) # Создание массива иксов

2. y = np.sin(x) # Функция

3. y\_clean = np.sin(x)

4. noise = np.random.uniform(-0.1,0.1,800) # Добавляем равномерно распределенный шум

5. spikes = np.random.choice([0, 1], size=x.shape, p=[0.98, 0.02])  # Случайные всплески (2% вероятность)

6. amplitude = np.random.uniform(-1, 1, size=x.shape)  # Случайная амплитуда всплесков

7. y += noise + spikes \* amplitude # дальше наш сигнал это зашумлённый сигнал

Затем данный зашумлённый сигнал был выведен на график:

Листинг 2 – Вывод зашумлённого сигнала

1. plt.figure(figsize=[8,3])

2. plt.plot(x,y, color='gray', linewidth=3, label='Зашумлённый сигнал')

3. plt.plot(x,y\_clean, color='black', linewidth=1.5, label='Оригинальный сигнал')

4. plt.grid()

5. ax = plt.gca()

6. ax.set\_xlim([-4\*np.pi,4\*np.pi])

7. ax.set\_ylim([-1.5,3])

8. plt.xlabel('x')

9. plt.ylabel('y')

10. plt.legend()

11. plt.show()

Изображение выглядит как линия, График, диаграмма, Шрифт

Автоматически созданное описание

Рисунок 1 – Результат выполнения листинга 2

**2. Реализация фильтра экспоненциального сглаживания.**

Фильтр экспоненциального сглаживания – простейший фильтр, выходной сигнал которого равен взвешенной сумме последнего измеренного значения сигнала и значения отфильтрованного сигнала на предыдущем шаге:

,

где  – измеренные значения (зашумлённый сигнал),  – значения отфильтрованного сигнала,  – настроечный параметр фильтра.

Данный фильтр был реализован в функции «Exponential\_smoothing\_filter»:

Листинг 3 – Функция фильтра экспоненциального сглаживания

1. def Exponential\_smoothing\_filter(a, alpha):

2.     y = a.copy()

3.     for i in range(1, y.size):

4.         y[i] = alpha \* y[i] + (1-alpha) \* y[i-1]

5.     return y

Данная функция была использована с тремя разными значениями коэффициента :

Листинг 4 – Функция фильтра экспоненциального сглаживания

1. plt.figure(figsize=[8, 9])  # Размер всей фигуры: 8x9 (3 графика по 8x3)

2.

3. for idx, alpha in enumerate(ALPHAS):

4.     y\_exp = Exponential\_smoothing\_filter(y, alpha)

5.     plt.subplot(len(ALPHAS), 1, idx + 1)

6.     plt.plot(x, y, color='gray', linewidth=3, label='Зашумлённый сигнал')

7.     plt.plot(x, y\_exp, color='black', linewidth=1.5, label=f'Отфильтрованный сигнал')

8.     plt.grid()

9.     plt.xlim([-4 \* np.pi, 4 \* np.pi])

10.     plt.ylim([-1.5, 4])

11.     plt.xlabel('x')

12.     plt.ylabel('y')

13.     plt.legend()

14.     plt.title(f'Отфильтрованный сигнал, α = {alpha}')

15.

16. plt.tight\_layout()

17. plt.show()

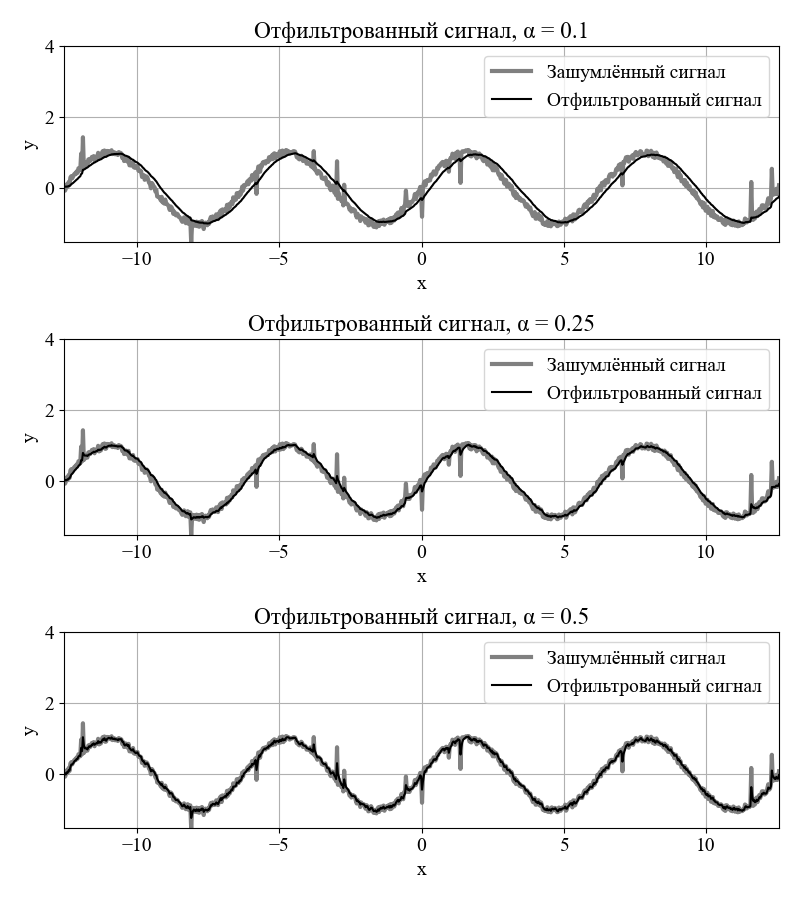


Рисунок 2 – Демонстрация работы фильтра экспоненциального сглаживания

По итогам анализа полученных графиков, оптимальным было принято значение , поскольку данное значение значительно сглаживает высокочастотные помехи в сигнале, но ещё не создаёт столь значимое запаздывание, как, например, значение .

**3. Реализация фильтра скользящего среднего**

Фильтр скользящего среднего хранит последние  измерений зашумленного сигнала и выдает на выход их среднее арифметическое.

Данный фильтр был реализован в функции «Exponential\_smoothing\_filter»: