Курсовой проект по курсу Дискретный Анализ. Преобразование Фурье

Выполнил студент группы М8О-307Б-21 МАИ *Друхольский Александр.*

Условие

Задача:

> Реализуйте алгоритм быстрое преобразование Фурье для действительного сигнала.
>
> Преобразование проводится скользящим окном на 4096 отсчёта с шагом 1024. Перед преобразованием Фурье необходимо подействовать на отсчёты окном Ханна.

Метод решения

Для решения поставленной задачи нам требуется реализовать алгоритм БПФ - быстрое преобразование Фурье, которое позволяет получить частотный спектр сигнала. Для начала воспользуемся предложенной библиотекой, которая кодирует входящий сигнал и возвращает массив отсчётов. Эти значения мы и будем преобразовывать с помощью алгоритма Фурье.

Определим, что ширина окна N = 4096, а шаг 1024. Предварительно применяем к данным отсчётам оконную функцию Ханна (Хеннинга), которая определяется как $w(n) = 0.5 * (1 - cos(2 * pi * n/(N-1)))$ - оконная функция визуально улучшает частотный спектр.

Для быстрого преобразования Фурье можно использовать метод с прорежеванием сигнала по времени. Вообще формула Фурье выглядит, как сумма значений отсчётов умноженных на экспоненту в степени фазы. Для алгоритма с прореживанием по времени нам требуется разбить исходные N отсчётов на четные и нечетные позиции. Далее в каждом блоке продолжим делить на четные и нечетные, получая множества x1(n) и x2(n). В определенный момент мы будем обрабатывать один символ. Оказывается, при такой перестановке, у отсчёта позиция i, то он переместится на позицию $reverse(binary(i))$, то есть если мы представим i в двоичном виде и развернём, то полученное число - новая позиция отсчёта. Для получения значения X(k), мы можем воспользоваться формулой $X(k) = x_1(k) + {W_n}^k * x_2(k)$, где W - поворотный угол. $x_1(k)$ и $x_2(k)$ - периодические функции. Тогда справедлива формула $X(k + N/2) = x_1(k + N/2) + {W_N}^{(k+N/2)} * x_2(k + N/2)$. Подобный метод значительно сокращает количество вычислений, которое требуется произвести для окна.

В данном алгоритме количество вычислений, которое требуется произвести для N отсчётов равно приблизительно N/2 * log(N). То есть сложность алгоритма, если длина сигнала - M отсчётов: $O(\frac{M}{N} * N * log(N)) = O(M * log(N))$

Описание программы

Основные моменты:

1. $vector < short > decoder()$ - кодировщик исходного сигнала.
2. $int\ bitReversalPermutation(int\ sz, inta)$ - функция, которая с помощью битовых операций вычисляет битовый реверс числа для поиска новой позиции.
3. $void\ fastFourier(vector < double > \&\ window, vector < complex < double >> \&\ res\_window, complex < double > z, int\ left, int\ right)$ - рекурсивная функция, производящая вычисления самой функции. На каждом углублении рекурсии мы передаем z*z, согласно формуле. left и right - левая и правая границы рассматриваемого куска.
4. $void\ calculateSpectrum(vector < short > sample)$ - функция, запускающая процесс подсчёта функции Фурье для каждого окна шириной 4096 с шагом 1024. Здесь же мы применяем к отсчётам окно Ханна, про которое написано выше.

Дневник отладки

CE - 1 Выбор не того компилятора.

RE - 1 Ошибся с названием входного файла.

WA - 1 Ошибки: неправильный вывод, неправильная точность вывода, вывод лишней информации.

PE - 1 Ошибка со смещением окна отсчётов, что привело к выводу большего количества чисел.

ML - 2 Ошибка из=за хранения спектра целиком. Перестал хранить спектр в двумерном массиве и стал внутри функции calculateSpectrum для каждого окна сразу определять и выводить ответ.

Тест производительности

Исследуем алгоритм на скорость $O(M * log(N))$, где N - ширина окна, M - суммарное количество отсчётов. Ширина окна постоянна, поэтому зависимость будем изучать на суммарном количестве отсчётов. Возьмём сигнал длиной 6 сек (281856 отсчётов) и будем его резать на более маленькие части, чтобы количество отсчётов отличалось примерно в два раза.

№	Длина строки	Время, с
1	281856	1.0285
2	155567	0.565378
3	77231	0.292325
4	36911	0.125492
5	16175	0.049351

Тестирование показало, что сложность действительно зависит от M линейно, при постоянной ширине окна.

Выводы

Я познакомился с алгоритмом быстрого преобразования Фурье для действительного сигнала. Мне удалось реализовать этот алгоритм с методом прореживания по времени. Далее этот алгоритм может быть использован для реализации аудиопоиска, так как из каждого аудиосигнала мы можем получить частотный спектр, который используем для сравнения.