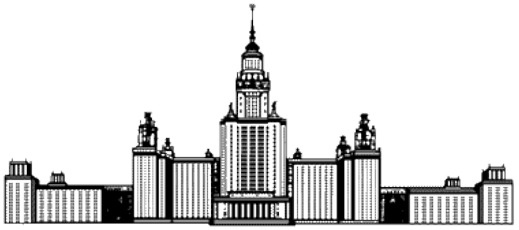
Московский Государственный Университет
имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математической физики



КУРСОВАЯ РАБОТА СТУДЕНТА 402 ГРУППЫ

Турганбаева Сатбека Амангельдыулы

Численное исследование одного метода восстановления волнового фронта

Научный руководитель:
д.ф-м.н. профессор Разгулин А. В.

Москва 2017

Содержание

Введение

Системы адаптивной оптики используемые в современных телескопах используют гибкие зеркала, для коррекции аберраций, вызванных турбулентность атмосферы Земли. Одним из устройств для измерения волнового фронта является датчик Шака-Гартмана. Данный датчик измеряет градиент волнового фронта, вместо него самого. Получаются матрицы градиентов X и Y.

В [1] был предложен метод восстановления волнового фронта по его наклонам с использованием преобразования Хаара. Идея метода заключается в том, что наклоны волнового фронта могут быть представлены в виде суперпозиции сверток с фильтрами преобразования Хаара. Используя это соотношение возможно выполнить "замену переменной и получить разложение волнового фронта по вейвлетам Хаара. Затем, применив стандартный алгоритм обратного преобразования получить исходный волновой фронт. Стоит отметить, что в [1] кроме наклонов волнового фронта и его интенсивности также требовалась и интенсивность правого нижнего квадранта ${}^{M-1}_{HH}\Phi$, но в более новой статье [2] было предложено более простое решение, и ослаблены требования к начальным данным.

Целью работы является ознакомление с методом , его программная реализация, а также реализация средств для его дальнейшего исследования. В ходе работы были разобраны статьи [1]- [2], и на их основе написана первая часть работы, знакомящая с преобразованием Хаара и методом. При программной реализации был написан ряд модулей, реализующих как сам метод, так и средства для его исследования. В работе описан их функционал. Также приведены примеры работы метода на полиномах Цернике.

1 Преобразование Хаара

1.1 Формулировки и определения

Z-преобразованием дискретного сигнала $\{s_j\}_{j\in Z}$ называется полином Лорана $P_s(z) = \sum_{j\in Z} s_j z^{-j}$.
$h = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}),\ g = (\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})$ - низкочастотный и высокочастотный фильтры преобразования Хаара.

$$H_L(z) = \frac{1+z^{-1}}{\sqrt{2}}$$

$$H_H(z) = \frac{1-z^{-1}}{\sqrt{2}}$$

$H_L(z)$, $H_H(z)$ - z - преобразования фильтров h и g соответственно.

Линейной сверткой двух дискретных сигналов $a(n),\ n = 0 \dots N-1$ и $b(n),\ n = 0 \dots M-1$ называется выражение:

$$h = a \otimes b$$

$$h(n) = \sum_{m=0}^{n} a(m)b(n-m)$$

В ввиду того, что сигналы конечномерные на границах возникает неопределенность из-за отсутствия соответствующих элементов. Проблема решается различными способами: дополнение одного из обоих сигналов 0-ми, константами, симметричное отражение и т.д.

Также одним из свойств z-преобразования является то, что z-преобразование свертки двух сигналов равно произведению z-преобразований этих сигналов.

$\uparrow s$ - операция, добавляющая 0 после каждого элемента сигнала s.

$\downarrow_k s$ - операция, удаляющая каждый k-ый элемент сигнала s.

Если $s = (1,2,3,4)$, то $\uparrow_2 s = (1,0,2,0,3,0,4,0)$, а $\downarrow_2 s = (1,3)$.

Также стоит отметить, что:

$$\downarrow_2 H_L(z^{2^k}) = H_L(z^{2^{k-1}}),\ k \geq 2$$

$$\downarrow_2 H_H(z^{2^k}) = H_H(z^{2^{k-1}}),\ k \geq 2$$

$$\uparrow_2 H_L(z^{2^{k-1}}) = H_L(z^{2^k}),\ k \geq 2$$

$$\uparrow_2 H_H(z^{2^{k-1}}) = H_H(z^{2^k}),\ k \geq 2$$

В дальнейшем под сигналом будет пониматься его z-преобразование и наоборот.

1.2 Одномерный сигнал

Будем предполагать в дальнейшем, что размерность сигналов равна $2^m,\ m \geq 1$.

Свернем сигнал $h_m(z)$, $dim\ h_m = 2^m$ с фильтрами $H_L(z)$, $H_H(z)$, а затем применим к получившемуся операцию $\downarrow_2$.

Получим сигналы:

$$h_{L_{m-1}}(z) = \downarrow_2 \{h_m(z)H_L(z)\}$$

$$h_{H_{m-1}}(z) = \downarrow_2 \{h_m(z)H_H(z)\}.$$

Их размерности будут в два раза меньше,размерности исходного сигнала h_m.

$h_{L_{m-1}}(z)$ называется низкочастотной составляющей сигнала h_m, а $h_{H_{m-1}}(z)$ высокочастотной.

Восстановление исходного сигнала происходит так:

$$h_m(z) = \uparrow_2 \{h_{L_{m-1}}(z)\}H_L(z) + \uparrow_2 \{h_{H_{m-1}}(z)\}H_H(z)$$

Описанные выше преобразования выполняют один шаг прямого и обратного преобразования Хаара. Прямое преобразование называется анализом, обратное синтезом.

Если положить, что $h_m = h_{L_m}$, то алгоритм анализа выглядит так:

k называется разрешением разложения. Анализ будет происходить до тех пор, пока не останется один элемент.

Algorithm 2 Алгоритм синтеза

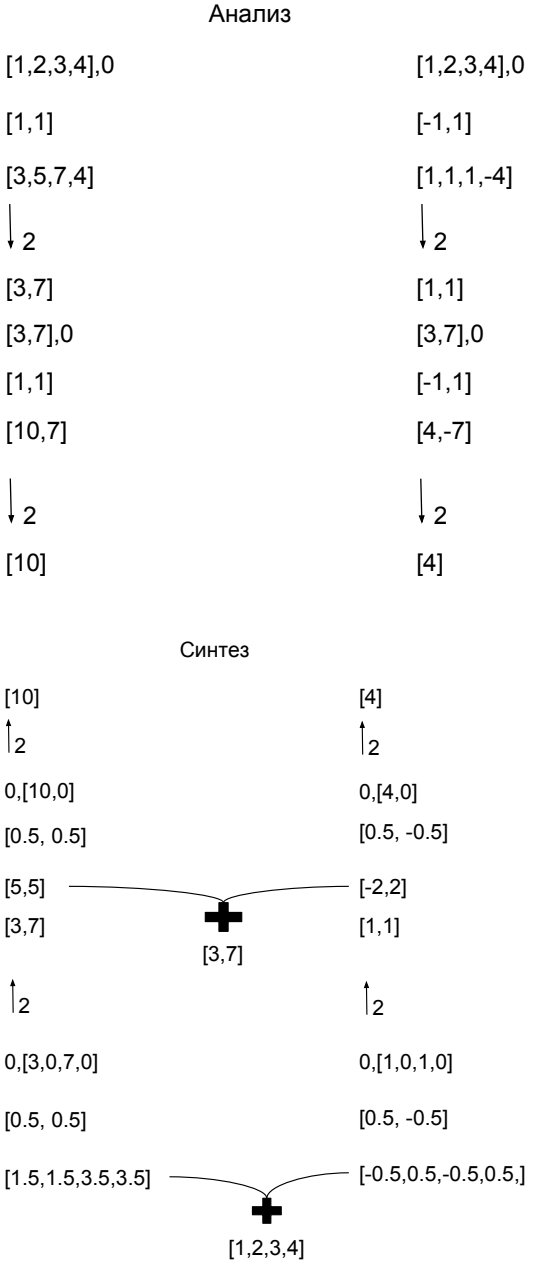
for $k = m \dots 1$ **do**
  $h_{L_{m-k+1}}(z) = \uparrow_2 \{h_{L_{m-k}}(z)\}H_L(z^{-1}) + \uparrow_2 \{h_{H_{m-k}}(z)\}H_H(z^{-1})$
end for

Algorithm 1 Алгоритм анализа

for $k = 1 \dots m$ **do**
$h_{L_{m-k}}(z) = \downarrow_2 \{h_{L_{m-k+1}}(z) H_L(z)\}$
$h_{H_{m-k}}(z) = \downarrow_2 \{h_{L_{m-k+1}}(z) H_H(z)\}$.
end for

1.3 Дополнение нулями

В описываемом методе при анализе будем дополнять нулями справа, а при синтезе слева. Ниже приведены примеры анализа и синтеза сигнала $(1, 2, 3, 4)$. Для наглядности используется нестандартные пары фильтров, $[1, 1]$, $[1, -1]$ и $[0.5, 0.5]$, $[0.5, -0.5]$.



1.4 Двумерное преобразование

Пусть дана матрица ${}^{M}\Phi, {}^{M}\Phi \in \mathbb{R}^{2^M \times 2^M}$. Применим к каждой строке матрицы один шаг анализа. В результате получим матрицы ${}^{m-1}_{L}\Phi$, ${}^{m-1}_{H}\Phi$. К каждому столбцу обеих матриц также применим шаг анализа. В итоге получим четыре матрицы ${}^{m-1}_{LL}\Phi$, ${}^{m-1}_{LH}\Phi$, ${}^{m-1}_{HL}\Phi$, ${}^{m-1}_{HH}\Phi$. ${}^{m-1}_{LL}\Phi$ - является низкочастотной составляющей двумерного сигнала, остальные три матрицы содержат детализирующую информацию. Таким образом будет выполнен первый шаг

двумерного преобразования Хаара. Нужно проделать аналогичные операции с ${}^{m-1}_{LL}\Phi$ для следующего шага. Такми образом, шаг двумерного преобразования свелся к композиции одномерных преобразований.

Синтез происходит аналогичным образом: в обратном анализу порядке дополняется нулями соответствующая размерность и применяются фильтры Хаара, а затем результаты складываются.

Пусть ${}^{M}\Phi = {}^{M}_{LL}\Phi$

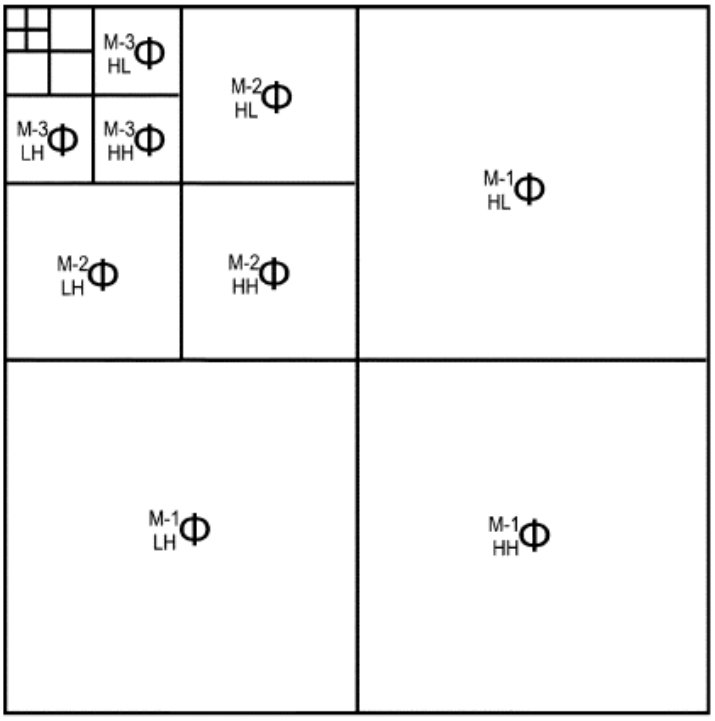
Algorithm 3 Алгоритм 2D-анализа

for $k = M \dots 1$ **do**
$${}^{k-1}_{LL}\Phi = \downarrow_2 \{{}^{k}_{LL}\Phi H_L(z_h)H_L(z_v)\}$$
$${}^{k-1}_{LH}\Phi = \downarrow_2 \{{}^{k}_{LL}\Phi H_L(z_h)H_H(z_v)\}$$
$${}^{k-1}_{HL}\Phi = \downarrow_2 \{{}^{k}_{LL}\Phi H_H(z_h)H_L(z_v)\}$$
$${}^{k-1}_{HH}\Phi = \downarrow_2 \{{}^{k}_{LL}\Phi H_H(z_h)H_H(z_v)\}$$
end for

Algorithm 4 Алгоритм 2D-синтеза

for $k = 1 \dots M$ **do**
$${}^{k-1}_{LL}\Phi = \uparrow_2 \{{}^{k-1}_{LL}\Phi H_L(z_v^{-1}) + {}^{k-1}_{LH}\Phi H_L(z_v^{-1})\}H_L(z_h^{-1}) + \uparrow_2 \{{}^{k-1}_{HL}\Phi H_L(z_v^{-1}) + {}^{k-1}_{HH}\Phi H_L(z_v^{-1})\}H_H(z_h^{-1})$$
end for

Полученное 2-D разложение можно представить в виде диаграммы, где на каждом уровне LL, LH, HL и HH составляющим соответствуют определенные квадранты.



1.5 Явные формулы для анализа

Из алгоритма анализа непосредственно выводятся формулы для LL, LH, HL и HH составляющих на каждом уровне.

$${}^{M-m}_{LL}\Phi = \downarrow_{2^m} \{{}^{M}\Phi \prod_{k=0}^{m-1} H_L(z_h^{2^k}) \prod_{k=0}^{m-1} H_L(z_v^{2^k})\} \tag{1}$$

$${}^{M-m}_{LH}\Phi = \downarrow_{2^m} \{{}^{M}\Phi H_H(z_v^{2^{m-1}}) \prod_{k=0}^{m-1} H_L(z_h^{2^k}) \prod_{k=0}^{m-2} H_L(z_v^{2^k})\} \tag{2}$$

$$ {}^{M-m}_{HL}\Phi =\downarrow_{2^m} \{{}^{M}\Phi H_H(z_h^{2^{m-1}}) \prod_{k=0}^{m-2} H_L(z_h^{2^k}) \prod_{k=0}^{m-1} H_L(z_v^{2^k})\} \tag{3} $$

$$ {}^{M-m}_{HH}\Phi =\downarrow_{2^m} \{{}^{M}\Phi H_H(z_v^{2^{m-1}}) H_H(z_v^{2^{m-1}}) \prod_{k=0}^{m-2} H_L(z_h^{2^k}) \prod_{k=0}^{m-2} H_L(z_v^{2^k})\} \tag{4} $$

2 Градиенты. Геометрия Хаджина и Фрайда

2.1 Геометрия Хаджина и Фрайда

В адаптивной оптике используются два различных способа представления наклонов волнового фронта. Согласно [3] это геометрии Хаджина и Фрайда. Геометрия Хаджина:

$$ {}_H x_{i,j} = -\phi_{i,j} + \phi_{i,j+1} $$

$$ {}_H y_{i,j} = -\phi_{i,j} + \phi_{i+1,j} $$

Получившиеся матрицы ${}^{M}_{H}X$, ${}^{M}_{H}Y$ имеют размерности $2^M \times (2^M - 1)$ и $(2^M - 1) \times 2^M$ соответственно. Геометрия Фрайда:

$$ {}_F x_{i,j} = \frac{{}_H x_{i,j} +_H x_{i+1,j}}{2} = \frac{-\phi_{i,j} + \phi_{i,j+1} - \phi_{i+1,j} + \phi_{i+1,j+1}}{2} $$

$$ {}_F y_{i,j} = \frac{{}_H y_{i,j} +_H y_{i,j+1}}{2} = \frac{-\phi_{i,j} - \phi_{i,j+1} + \phi_{i+1,j} + \phi_{i+1,j+1}}{2} $$

Получившиеся матрицы ${}^{M}_{F}X$, ${}^{M}_{F}Y$ имеют одинаковые размерности $2^M - 1 \times 2^M - 1$.

2.2 Градиенты и преобразование Хаара

Справедливы следующие соотношения.

$$ {}^{M}_{H}X =^{M} \Phi\sqrt{2}H_H(z_h) \tag{5} $$

$$ {}^{M}_{H}Y =^{M} \Phi\sqrt{2}H_H(z_v) \tag{6} $$

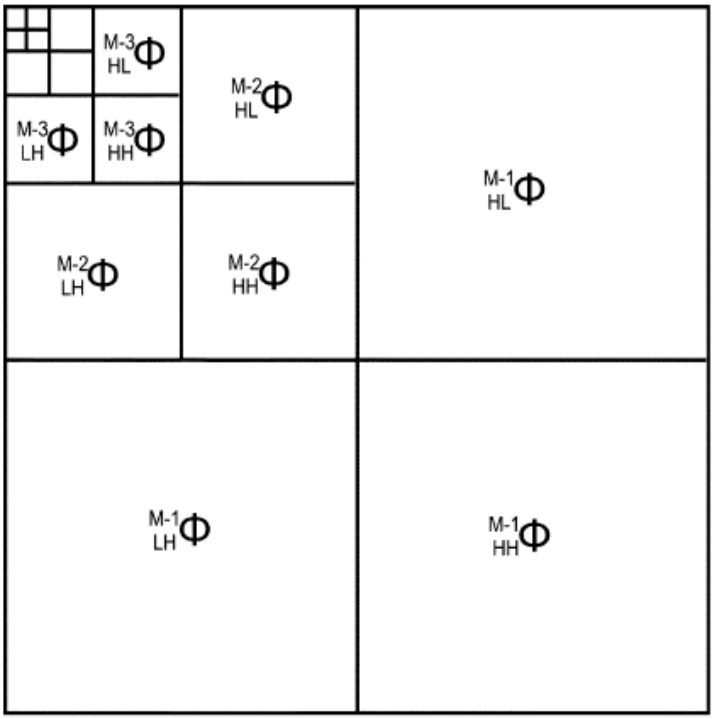
$$ {}^{M}_{F}X = \frac{{}^{M}_{H}XH_L(z_v)}{\sqrt{2}} =^{M} \Phi H_H(z_h)H_L(z_v) \tag{7} $$

$$ {}^{M}_{F}Y = \frac{{}^{M}_{H}YH_L(z_h)}{\sqrt{2}} =^{M} \Phi H_L(z_h)H_H(z_v) \tag{8} $$

3 Вывод формул прямого преобразования

Рассмотрим случай, когда известны вертикальные и горизонтальные наклоны, а также интенсивность волнового фронта. Иначе говоря имеются: ${}^{M}_{F}X$, ${}^{M}_{F}Y$, ${}^{M}_{H}X$, ${}^{M}_{H}Y$, ${}^{LL}_{0}\Phi$.

Необходимо получить из имеющихся данных получить $2D$-разложение волнового фронта, а затем восстановить сам волновой фронт, применив к полученному разложению стандартный алгоритм синтеза.[1] Диаграмма разложения которое необходимо получить будем аналогична диаграмме стандартного разложения:



Отличаться будут лишь формулы (1)-(4). "Идейно"метод заключается в замене переменной. Используя соотношения (5)-(8), получим аналоги (1)-(4).

3.1 Получение LH, HL квадрантов

Из соотношений (5), (6) и (3), (2) соответственно, непосредственно вытекает, что:

$$\begin{matrix}{}^{M-1}_{HL}\Phi = \downarrow_2 {}^{M}_{F}X \\ {}^{M-1}_{LH}\Phi = \downarrow_2 {}^{M}_{F}Y\end{matrix}$$

Докажем справедливость выражения $H_H(z^{2^{m-1}}) = \sqrt{2}^{m-1} H_H(z) \prod\limits_{k=0}^{m-2} H_L(z^{2^k})$

Доказательство. Запишем $H_H(z^{2^{m-1}})$ в виде полинома

$$H_H(z^{2^{m-1}}) = \frac{1 - z^{2^{m-1}}}{\sqrt{2}}$$

$$\frac{1 - z^{2^{m-1}}}{\sqrt{2}} = \frac{(1 - z^{2^{m-2}})(1 + z^{2^{m-2}})}{\sqrt{2}} = \ldots = \frac{(1-z)(1+z)(1+z^2)\ldots(1+z^{2^{m-2}})}{\sqrt{2}}$$

$$H_H(z^{2^{m-1}}) = \sqrt{2}^{m-1} H_H(z) \prod_{k=0}^{m-2} H_L(z^{2^k})$$

□

LH, $m > 1$**:**

$${}^{M-m}_{LH}\Phi = \downarrow_{2^m} \{{}^{M}\Phi[H_H(z_v^{2^{m-1}})][\prod_{k=0}^{m-1} H_L(z_h^{2^k})] \prod_{k=0}^{m-2} H_L(z_v^{2^k})\} =$$

$$= \downarrow_{2^m} \{{}^{M}\Phi[\sqrt{2}^{m-1} H_H(z_v) \prod_{k=0}^{m-2} H_L(z_v^{2^k})][H_L(z_h) \prod_{k=1}^{m-1} H_L(z_h^{2^k})] \prod_{k=0}^{m-2} H_L(z_v^{2^k})\}$$

$$= \sqrt{2}^{m-1} \downarrow_{2^m} \{{}^{M}\Phi H_L(z_h) H_H(z_v) \prod_{k=1}^{m-1} H_L(z_h^{2^k}) \prod_{k=0}^{m-2} H_L^2(z_v^{2^k})$$

$$= \sqrt{2}^{m-1} \downarrow_{2^m} \{{}^M_F Y \prod_{k=1}^{m-1} H_L(z_h^{2^k}) \prod_{k=0}^{m-2} H_L^2(z_v^{2^k})$$

HL, $m > 1$**:**

Выводится аналогично, LH

$${}^{M-m}_{HL}\Phi = \sqrt{2}^{m-1} \downarrow_{2^m} \{{}^M_F Y \prod_{k=0}^{m-2} H_L^2(z_h^{2^k}) \prod_{k=1}^{m-1} H_L(z_v^{2^k})\}$$

3.2 Получение HH квадранта

HH, $m = 1$**:**[2]

Докажем, что ${}^{M-1}_{HH}\Phi \downarrow_2 \{\frac{\sqrt{2}}{4}[{}^M_H X H_H(z_v) + {}^M_H Y H_H(z_h)]\}$

Доказательство. Известно, что ${}^{M-1}_{HH}\Phi = \downarrow_2 [{}^M\Phi H_H(z_h) H_H(z_v)]$.

Распишем ${}^{M-1}_{HH}\Phi = \downarrow_2 \{\frac{\sqrt{2}}{2}\sqrt{2}\ {}^M\Phi H_H(z_h) H_H(z_v)\}$, выделив из (7) ${}^M_H X$ получим:

$${}^{M-1}_{HH}\Phi = \downarrow_2 \{\frac{\sqrt{2}}{2}\ {}^M_H X H_H(z_v)\} = \downarrow_2 \{\frac{\sqrt{2}}{4} 2\ {}^M_H X H_H(z_v)\}$$

Разделив (7) на (8) получим:

$${}^M_H X H_H(z_v) = {}^M_H Y H_H(z_h)$$

Отсюда получим:

$${}^{M-1}_{HH}\Phi \downarrow_2 \{\frac{\sqrt{2}}{4}[{}^M_H X H_H(z_v) + {}^M_H Y H_H(z_h)]$$

□

HH, $m > 1$**:**

$${}^{M-m}_{HH}\Phi = \downarrow_{2^m} \{{}^M\Phi H_H(z_v^{2^{m-1}}) H_H(z_v^{2^{m-1}}) \prod_{k=0}^{m-2} H_L(z_h^{2^k}) \prod_{k=0}^{m-2} H_L(z_v^{2^k})\} =$$

$$= \downarrow_{2^m} \{{}^M\Phi\sqrt{2}^{m-1} H_H(z_h) \prod_{k=0}^{m-2} H_L(z_h^{2^k}) \prod_{k=0}^{m-2} H_L(z_h^{2^k}) H_H(z_v^{2^{m-1}}) H_L(z_v) \prod_{k=1}^{m-2} H_L(z_v^{2^k}) =\}$$

$$= \sqrt{2}^{m-1} \downarrow_{2^m} \{{}^M_F X \prod_{k=0}^{m-2} H_L^2(z_h^{2^k}) H_H(z_v^{2^{m-1}}) \prod_{k=1}^{m-2} H_L(z_v^{2k})\}$$

4 Программная реализация

Программная реализация представляет из себя две группы модулей на языке Python 3.6.

Первая группа реализует сам метод и предоставляет функции *analyze* и *syntesis*. Вторая группа отвечает за реализацию интерфейса для дальнейшего изучения метода.

Рассмотрим подробнее структуру первой группы модулей.

4.1 Программная реализация метода

Список группы модулей, реализующих сам метод:

- **data.py**
- **gradients.py**
- **measurements.py**
- **util.py**
- **decomposition.py**

data.py - модуль отвечает за задание начальных данных. Предоставляет функцию *get_ plane*, которая задает сетку размера $2^M \times 2^M$ для дальнейших вычислений.

gradients.py - модуль предоставляет функции для нахождения градиента от заранее известной матрицы. В модуле находятся функции: *Fried_ gradient_ model* и *Hudgin_ gradien_ model*, которые возвращают градиенты по геометриям Фрайда и Хаджина.

measurements.py - модуль предоставляет функции для нахождения метрик схожести изображений. Модуль включает функцию *mse*, которая находит средне квадратичное отклонение двух изображений.

util.py - служебный модуль, используемый другими. Предоставляет функции для свертки сигналов, работы с z-преобразованиями. Включает в себя функции:

- *downsample* - реализует операцию $\downarrow_k$ с одномерными и двумерными массивами
- *convolve_ 2d* - осуществляет свертку двумерного массива с фильтром либо по строкам, либо по столбцам;
- *GetH_ h* - возвращает вектор, соответствующий z-преобразованию $H_H(z^k)$, для $k \geq 1$.
- *GetH_ l* - возвращает вектор, соответствующий z-преобразованию $H_L(z^k)$, для $k \geq 1$.

decomposition.py - реализуется разложение двумерного сигнала по вейвлетам Хаара. Основными функциями являются *analyze* и *syntesis* - именно они реализуют прямое и обратное преобразования Хаара. Ввиду того, что уже существует крупная библиотека, реализующая вейвлет преобразования **pywt (PyWawelets)**, результат функции *analyze* не конфликтует с результатом библиотечной функции *pywt.dwt2*, реализующей прямые вейвлет преобразования двумерных сигналов. В виду того, что в методе используется стандартный алгоритм обратного преобразования Хаара функция *syntesis*, реализующая его является оберткой над библиотечной функций *pywt.idwt2*.

4.2 Программная реализация интерфейса для исследования метода

Интерфейс для исследования метода реализуется двумя модулями **research.py** и **view.py**.

research.py используя модули из другой группы, предоставляет функции, которые возвращают информацию необходимую для исследования метода. Модуль предоставляет функцию *get_ all*.

get_ all(func, grad_ x, grad_ y, x_ s, x_ e, y_ s, y_ e, M)

Функция принимает в качестве параметров три функции: исходную функцию и два её градиента в явном виде, затем функция принимает параметры для задания сетки , на которой будут заданы функция и ее градиенты, значение $LL[0]$. Функция вычисляет следующий набор данных:

- исходную функцию
- исходные градиенты
- ${}^M_H X$, ${}^M_H Y$ составляющие от градиента
- ${}^M_F X$, ${}^M_F Y$ составляющие от градиента
- LL, LH, HL, HH квадранты разложения
- среднеквадратичное отклонение исходной функции от восстановленной ($LL[M]$)
- размерность матрицы, M

view.py отвечает за визуализацию и детализацию информации полученной из **research.py**. Предоставляет функцию *show_ all*.

show_ all(func, grad_ x, grad_ y, x_ s, x_ e, y_ s, y_ e, M)

Принимает те же параметры, что и *get_ all*,вызывая ее внутри себя. Выводит изображения исходного и восстановленного сигналов. А также печатает следующие данные:

- *mse* от градиентов Хаджина и Фрайда с настоящим градиентом
- $LL[0]$ исходного изображения
- $LL[0]$ восстановленного изображения
- Максимум исходного изображения
- Максимум восстановленного изображения
- Минимум исходного изображения

- Минимум восстановленного изображения
- размерность изображения

Под mse и среднеквадратичным отклонением подразумевается:

$$\sqrt{\frac{\sum_{k=1}^{n}(X_k - Y_k)^2}{n\ S}}$$

S - "площадь"матрицы X или Y, произведение количества строк на количество столбцов.

5 Результаты вычислительных экспериментов

Проверка работоспособности метода осуществляется с помощью разработанного интерфейса, функции *view.show_all* на полиномах Чернике.

```
res1 = view.show_all(lambda x,y: x**2 + y**2, lambda x,y: 2*x, lambda x,y: 2*y, -10,10,-10,10, 8,1400)
```

```
MSE =  1.96548776779
M =  8
Погрешность X_H= 0.0416500410518
Погрешность Y_H= 0.0416500410518
Погрешность X_F= 0.0417316278902
Погрешность Y_F= 0.0417316278902
LL[0] исходного изображения  [[ 17200.52287582]]
LL[0] восстновленного изображения  1400.0
Максимум исходного изображения 200.0
Максимум восстновленного изображения 1718.80208333
Минимум исходного изображения 0.00307574009996
Минимум восстновленного изображения -851.237132353
```

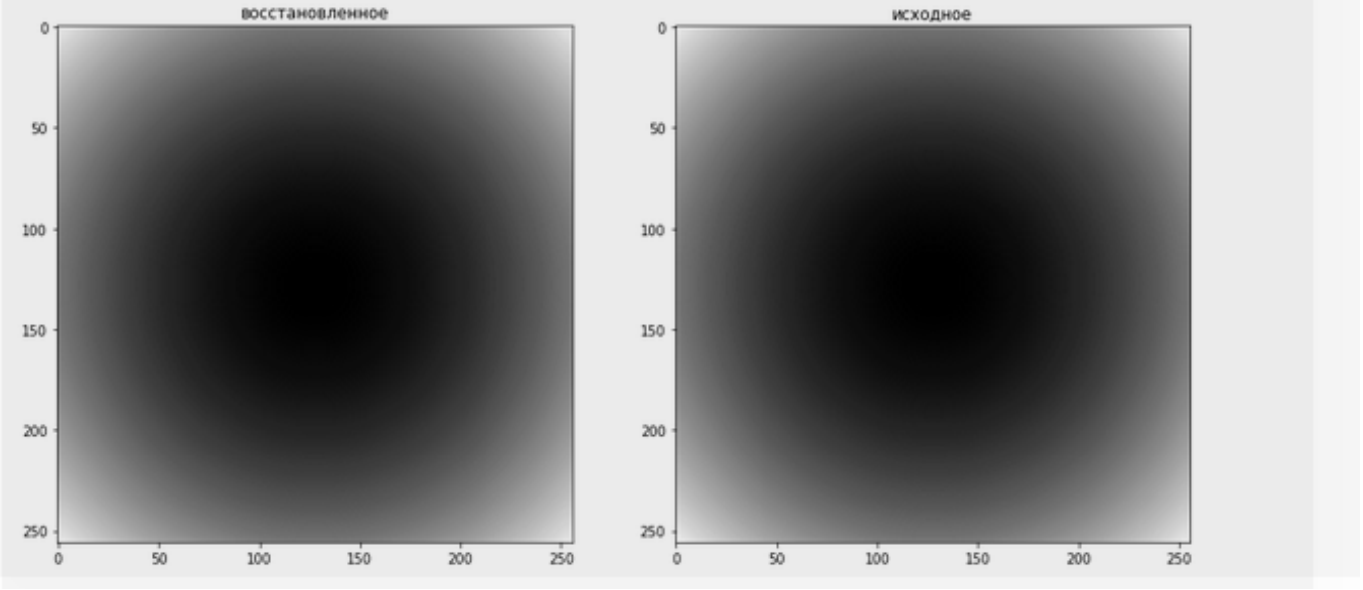


Рис. 1: Полином $R_2^2(x, y) = x^2 + y^2$

```
res4 = view.show_all(lambda x,y:np.sqrt(x**2 + y**2),lambda x,y: x/(np.sqrt(x**2 + y**2)),
                     lambda x,y: y/(np.sqrt(x**2 + y**2)), -1,1,-1,1,8 ,1)
```

```
MSE =  0.141349800341
M =  8
Погрешность X_H= 0.00274277930801
Погрешность Y_H= 0.00274277930801
Погрешность X_F= 0.00275122153648
Погрешность Y_F= 0.00275122153648
LL[0] исходного изображения  [[ 196.65714983]]
LL[0] восстановленного изображения  1.0
Максимум исходного изображения 1.41421356237
Максимум восстановленного изображения 83.077240418
Минимум исходного изображения 0.00554593553872
Минимум восстановленного изображения -97.9615391337
```

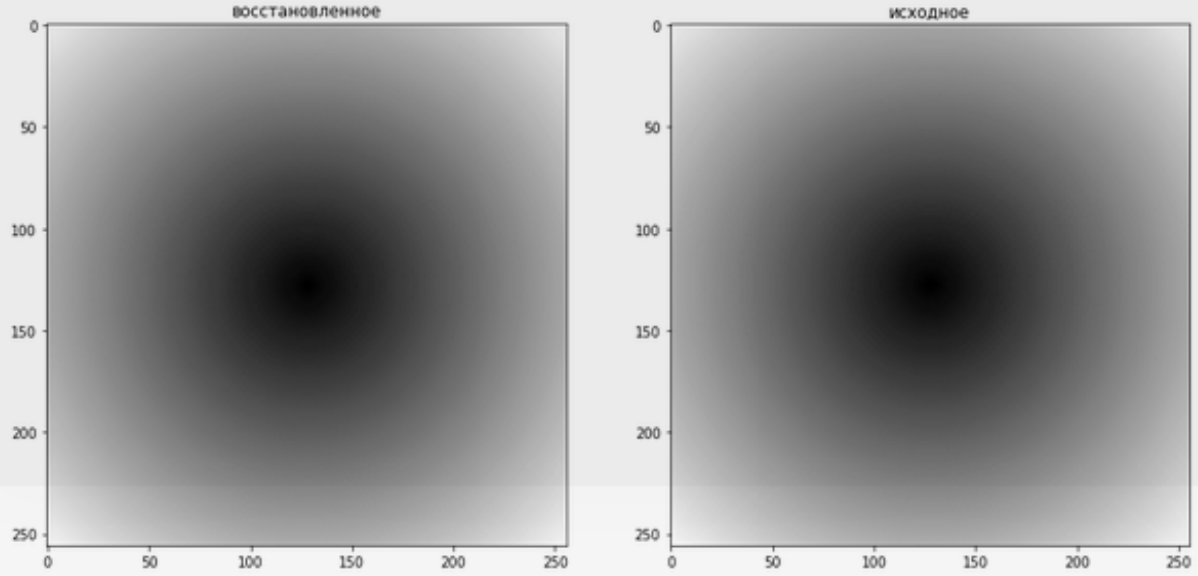


Рис. 2: Полином $R_1^1(x, y) = \sqrt{x^2 + y^2}$

```
res5 = view.show_all(lambda x,y:2*(x**2 + y**2) - 1,lambda x,y: 4*x,
                     lambda x,y: 4*y, -1,1,-1,1,8,88 )
```

```
MSE =  0.420003376637
M =  8
Погрешность X_H= 0.0089680179661
Погрешность Y_H= 0.0089680179661
Погрешность X_F= 0.00898558510922
Погрешность Y_F= 0.00898558510922
LL[0] исходного изображения  [[ 88.01045752]]
LL[0] восстновленного изображения  88.0
Максимум исходного изображения 3.0
Максимум восстновленного изображения 343.010416667
Минимум исходного изображения -0.999938485198
Минимум восстновленного изображения -170.997426471
```

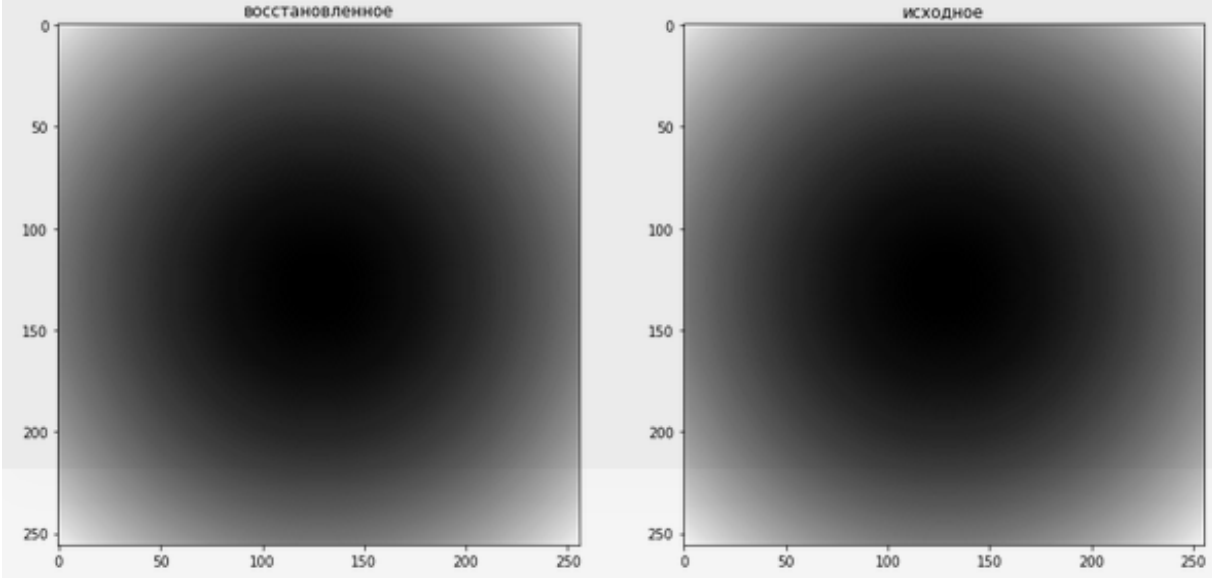


Рис. 3: Полином $R_2^0(x, y) = 2(x^2 + y^2) - 1$

```
res6 = view.show_all(lambda x,y:(x**2 + y**2)*np.sqrt(x**2 + y**2),
                     lambda x,y: 2*x*np.sqrt(x**2 + y**2) + x**3/(np.sqrt(x**2 + y**2)) + x*y**2/np.sqrt(x**2 + y**2),
                     lambda x,y:2*y*np.sqrt(x**2 + y**2) + y**3/(np.sqrt(x**2 + y**2)) + y*x**2/np.sqrt(x**2 + y**2),
                     -1,1,-1,1,8)
```

```
MSE =  0.270378428597
M =  8
Погрешность X_H= 0.00650719267634
Погрешность Y_H= 0.00650719267634
Погрешность X_F= 0.00651086342343
Погрешность Y_F= 0.00651086342343
LL[0] исходного изображения  [[ 162.45008259]]
LL[0] восстновленного изображения  1.0
Максимум исходного изображения 2.82842712475
Максимум восстновленного изображения 283.970569152
Минимум исходного изображения 1.70578563282e-07
Минимум восстновленного изображения -80.9072843798
```

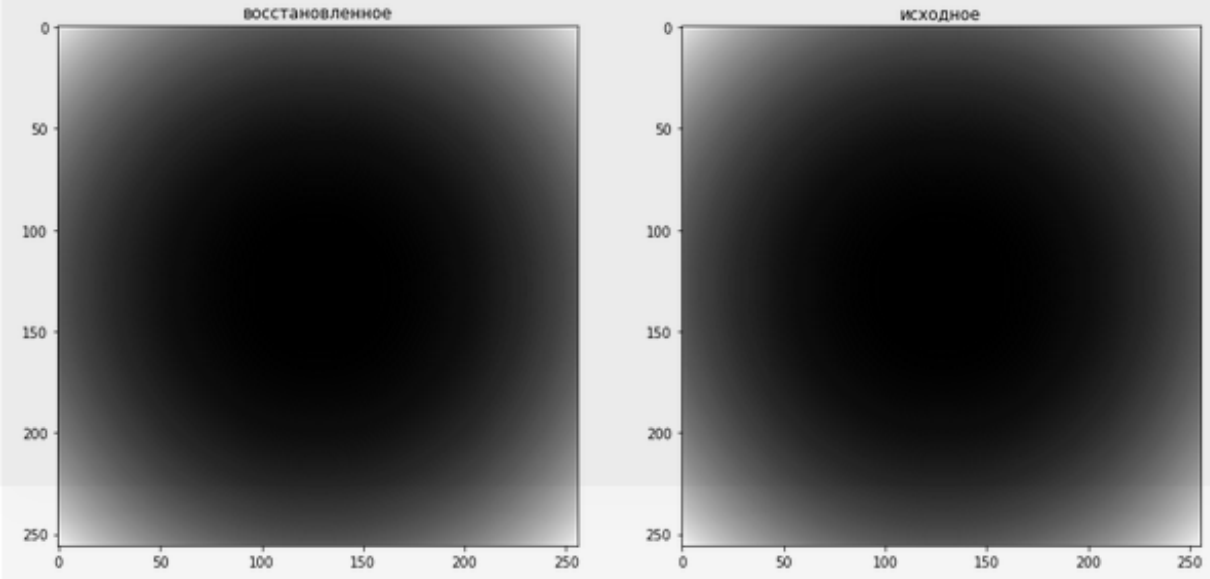


Рис. 4: Полином $R_3^3(x,y) = (x^2+y^2)\sqrt{x^2+y^2}$

```
res3 = view.show_all(lambda x,y: x**2 - y**2, lambda x,y: 2*x, lambda x,y: -2*y, -10,10,-10,10, 8)
```

```
MSE =  1.95064463103
M =  8
Погрешность X_H= 0.0416500410518
Погрешность Y_H= 0.0416500410518
Погрешность X_F= 0.0417316278902
Погрешность Y_F= 0.0417316278902
LL[0] исходного изображения  [[  2.27373675e-13]]
LL[0] восстновленного изображения  0.999999999996
Максимум исходного изображения 99.99846213
Максимум восстновленного изображения 1285.02351409
Минимум исходного изображения -99.99846213
Минимум восстновленного изображения -1285.01570159
```

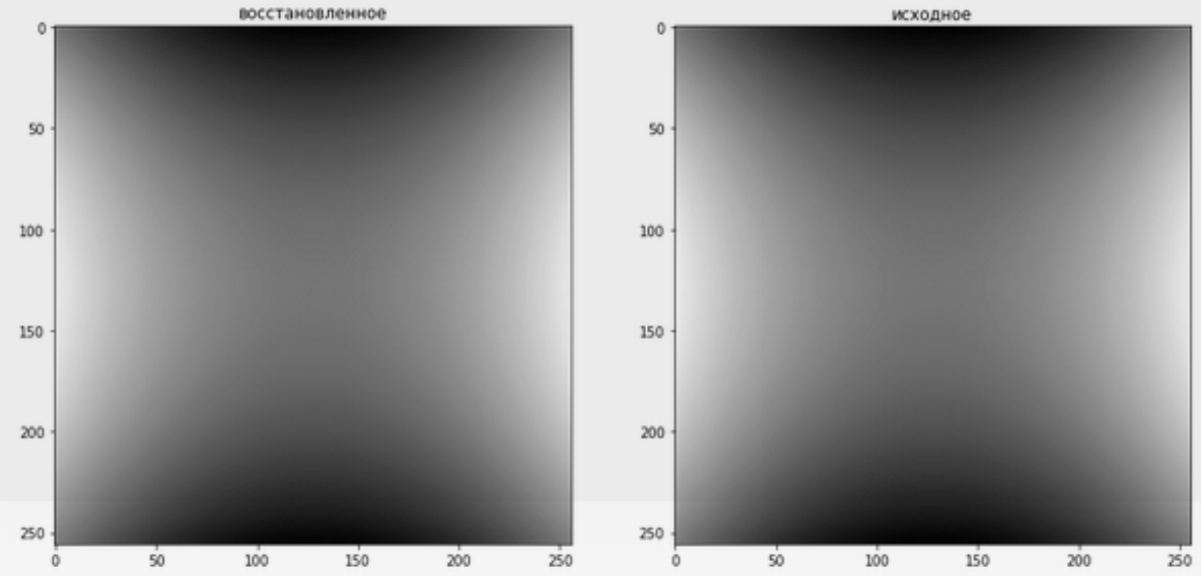


Рис. 5: $x^2 - y^2$

Заключение

В ходе работы удалось реализовать метод и проверить его работоспособность. Однако, реализовать именно тот алгоритм, который был предложен в [1] и [2] не удалось. Был использован менее эффективный по производительности алгоритм на основе формул для прямого преобразования. Были написаны модули для работы и исследования метода, с обособленной логикой, что позволит упростить дальнейшую разработку. Удалось инкапсулировать неоднозначную работу со сверткой сигналов и z-преобразованиями, и избежать конфликтов с модулем pywt, что может может в дальнейшем при тестировании.

Из полученных программой результатов можно сделать вывод, что точность восстановления зависит от точности аппроксимации производной моделями Хаджина и Фрайда. Однако это требует дополнительных исследований.

Стоит отметить, что метод обладает ресурсом параллелизма.

Список литературы

[1] IEEE Pan Agathoklis Senior Member IEEE Peter J. Hampton, Student Member and Colin Bradley. A new wave-front reconstruction method for adaptive optics systems using wavelets. *IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING*, 2(5):781–792, 10 2008.

[2] Panajotis Agathoklis Ioana S. Sevcenco, Peter J. Hampton. A wavelet based method for image reconstruction from gradient data with applications. *Springer Science+Business Media New York*, pages 717–737, 9 2013.

[3] D. T. Gavel L. A. Poyneer and J. M. Brase. Fast wavefront recon- struction in large adaptive optics systems with the fourier transform. *J. Opt. Soc. Amer. A*, 2002.