# Вейвлеты

Лекция 4.2

*Лектор – доцент каф. ВТ ИК ТПУ,*
*Болотова Юлия Александровна*

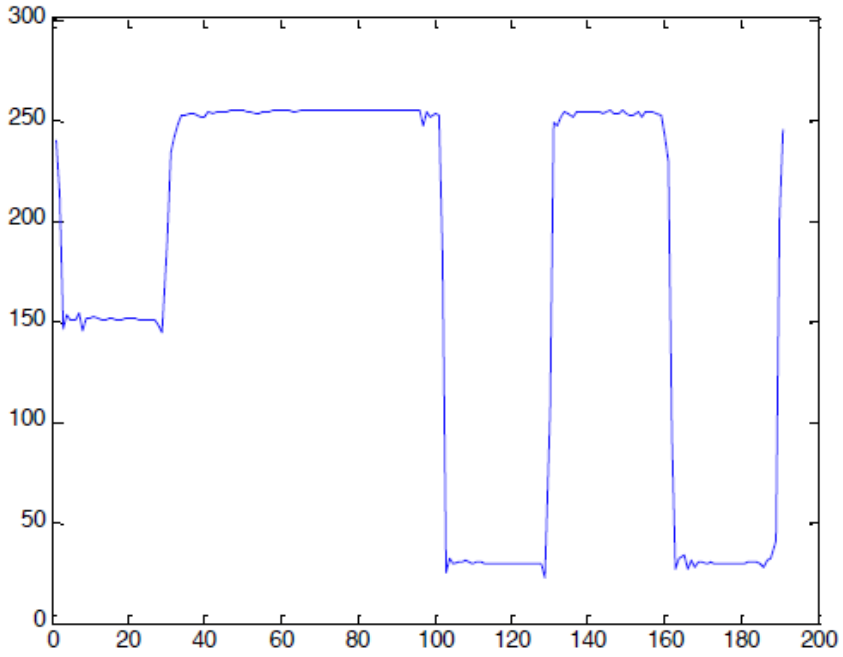
# Растровые изображения

- Трудно поддаются анализу.
- Занимают много памяти, сложности с хранением и передачей.

Можно ли описать их с помощью функций, чтобы впоследствии восстановить без потерь или с минимальными потерями???

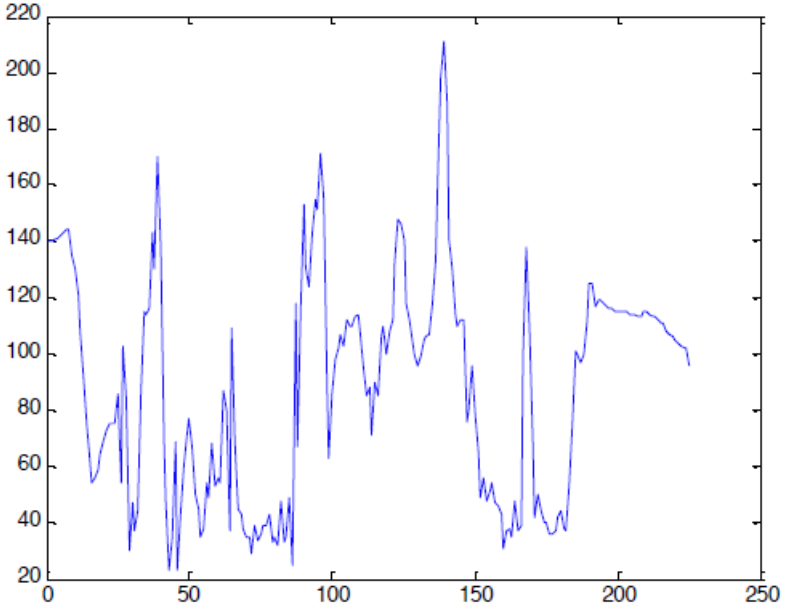
# Проблема – описание изображений

Представим «одномерную картинку»

# Проблема – описание изображений

1-D изображение

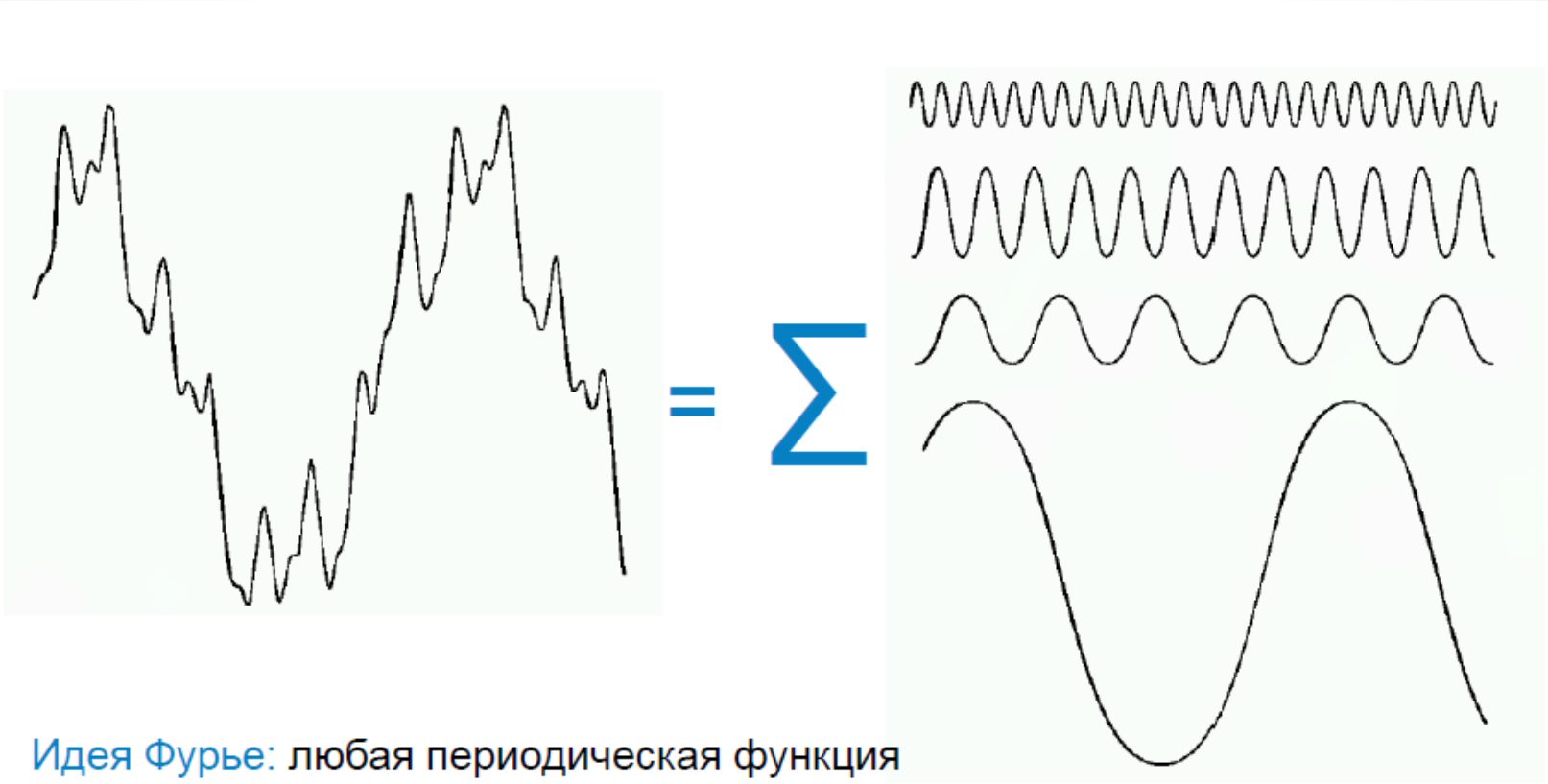


# Преобразование Фурье

- Книга «*Аналитическая теория тепла*» 1822 г.

- **Ряд Фурье -** любая функция, ***периодически воспроизводящая свои значения***, может быть представлена в виде ***суммы синусов*** и/или ***косинусов*** различных частот, умноженных на некоторые коэффициенты (сложность поведения функции при этом не имеет значения).

# Частотное представление – основная идея



Идея Фурье: любая периодическая функция может быть представлена в виде суммы синусов и косинусов, умноженных на некоторые коэффициенты (ряд Фурье)

# Когда функция непериодическая

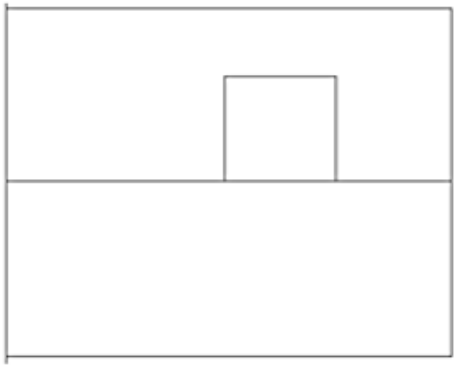
- **Преобразование Фурье:** Когда функция не является периодической, но площадь под ее графиком конечна, она может быть выражена в виде интеграла от синусов и/или косинусов, умноженных не некоторую весовую функцию.
- **Важно**: Функция, заданная как рядом, так и преобразованием Фурье может быть полностью, без потери информации, восстановлена при помощи некоторой процедуры обращения.

# Быстрое преобразование Фурье

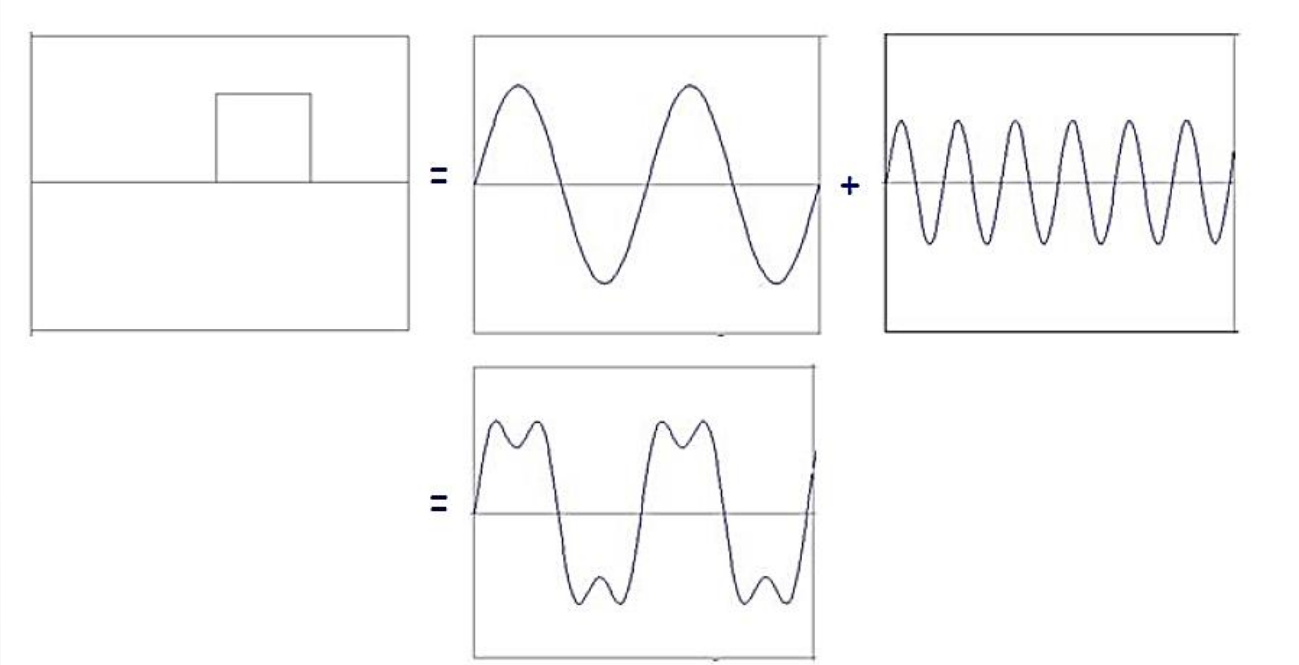
- **Быстрое преобразование Фурье** (БПФ) в конце 50-х гг. произвело революцию в области обработки сигналов, сделав возможным обработку и интерпретацию огромной совокупности сигналов в различных сферах от медицинской диагностики до новейших средств электронной связи.

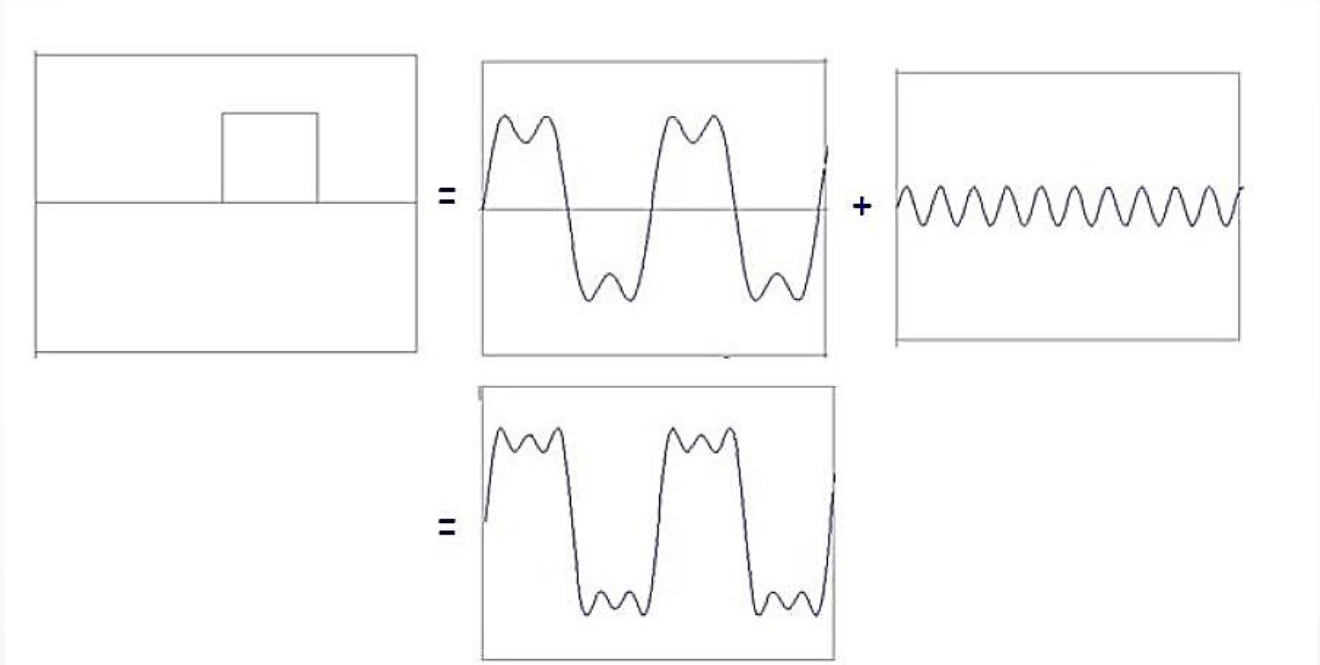
- При обработке изображений мы будем иметь дело только с функциями (изображениями) конечной протяженности, нас будет интересовать именно **преобразование Фурье**.

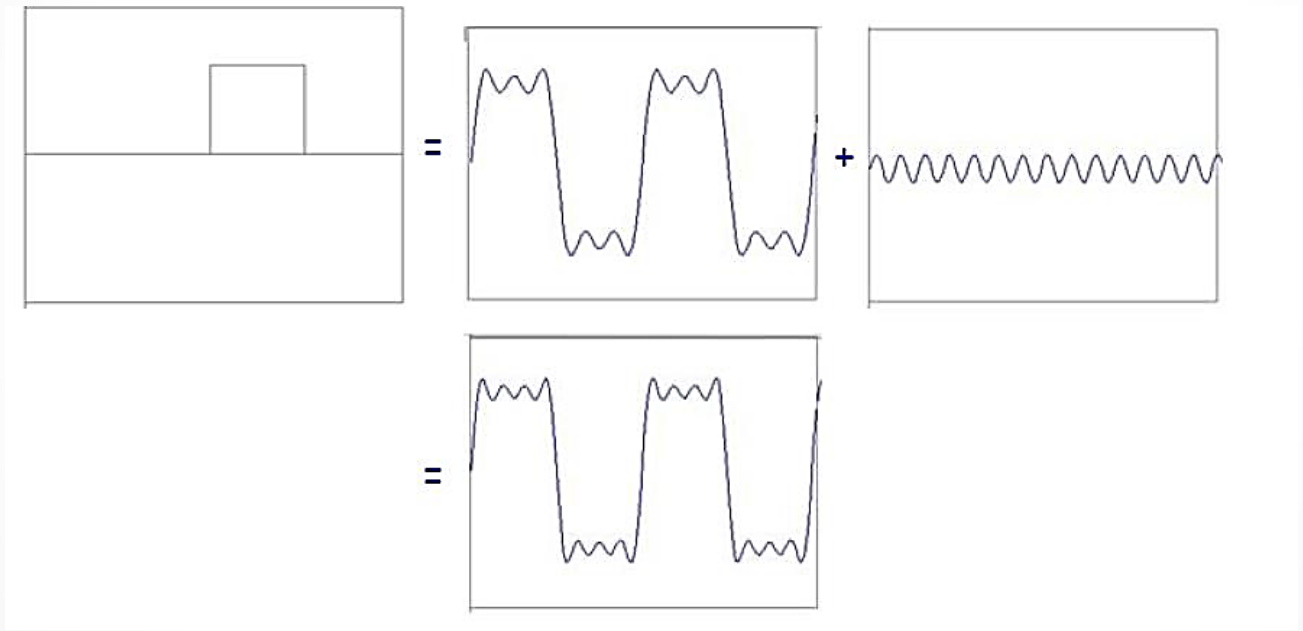
# Прямоугольный сигнал

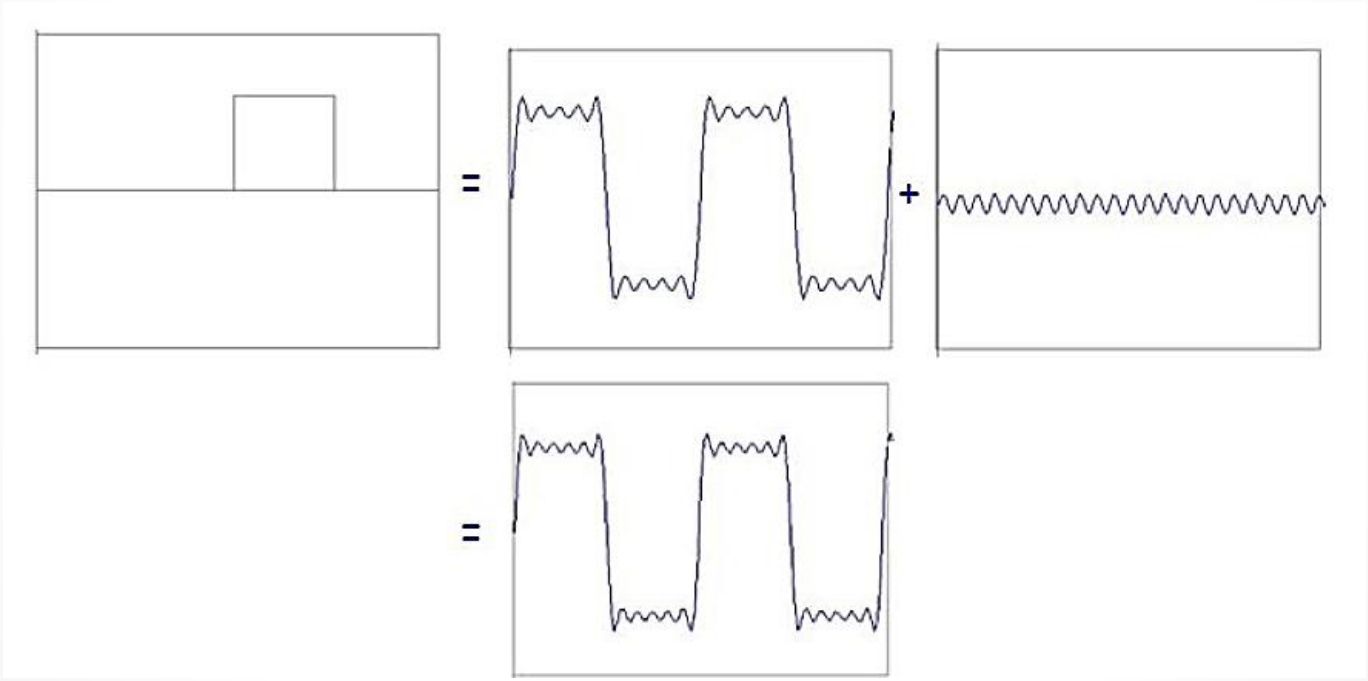


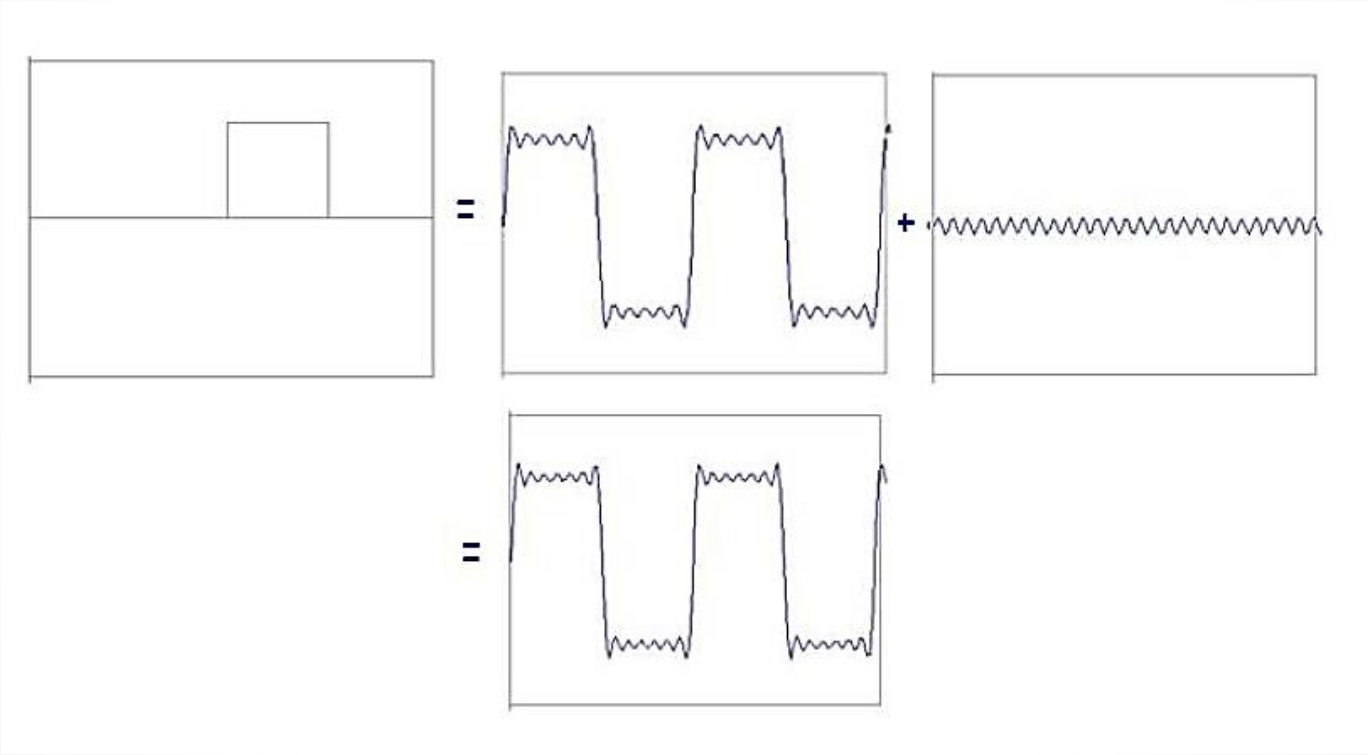
# Прямоугольный сигнал

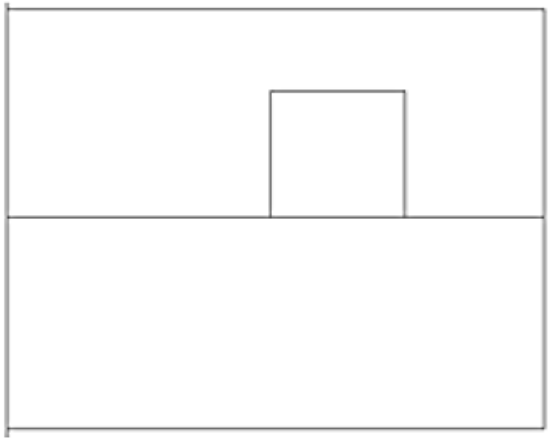






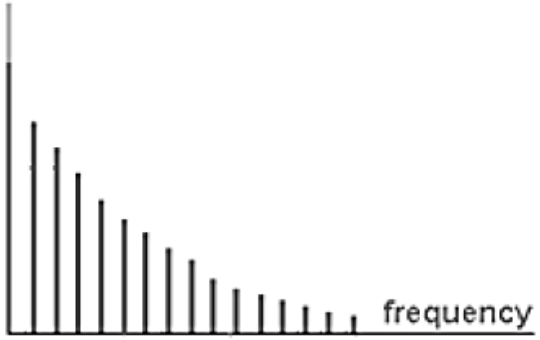






=

$$A\sum_{k=1}^{\infty}\frac{1}{k}\sin(2\pi kt)$$



# Преобразование Фурье

Применение:

- Анализ изображений
- Выделение значимых признаков
- Фильтрация
- Удаление шумов

Но!

- С помощью ПФ невозможно выявлять и анализировать закономерности сигнала.
- Невозможно полностью восстановить исходный сигнал.

# Вейвлеты

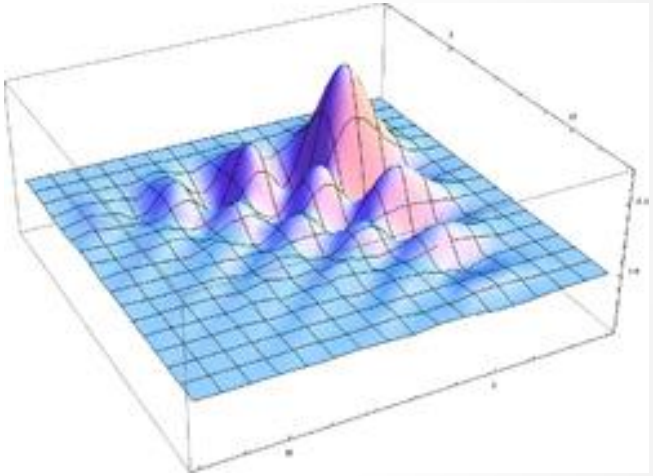
- Слово **"*wavelet*"**, является переводом на английский язык с французского слова **"*ondelette*"**,означающего небольшие волны, следующие друг за другом, рябь;

- Т.е. разложение функции по малым волнам.

# Вейвлеты

Область применения:

- Сжатие изображений
- Получение информативных признаков
- …



Разберем понятие вейвлетов на алгоритмах сжатия

# Алгоритмы сжатия

Алгоритмы сжатия «любят», когда в данных есть **закономерность**. Лучше всего сжимаются длинные последовательности нулей.

Чтобы записывать в память 100 нулей, можно записать просто число 100 (с пометкой, что это именно количество нулей). Декодирующая программа «поймёт», что имелись в виду нули и воспроизведёт их.

! Однако если в нашей последовательности в середине вдруг окажется единица, то одним числом 100 ограничится не удастся.

# Алгоритмы сжатия

Но зачем кодировать абсолютно все детали?

При просмотре фотографии незначительные колебания яркости незаметны.

При кодировании возможно изменить изображение, при этом степень сжатия сразу вырастет. При этом декодированное изображение будет незначительно.

# Алгоритмы сжатия

Цель — преобразовать изображение так, чтобы оно хорошо сжималось классическими алгоритмами. Нужно изменить изображение так, чтобы получить длинные цепочки нулей.

# Алгоритмы сжатия

У «реальных» изображений, таких как фотографии, есть особенность: яркость соседних пикселей обычно отличается на небольшую величину.

Как правило, контрастные перепады занимают лишь малую часть изображения.

# Дельта-кодирование

- Рассмотрим фрагмент первой строки яркостей изображения:

154, 155, 156, 157, 157, 157, 158, 156.

Соседние числа очень близки.

Алгоритм:

1. закодировать первое число;
2. рассматривать лишь отличия каждого числа от предыдущего.

# Дельта-кодирование

Получаем:

154, 155, 156, 157, 157, 157, 158, 156

154, 1, 1, 1, 0, 0, 1, -2.

Такой метод в самом деле используется и называется **дельта-кодированием**.

Недостаток — **нелокальный**. Т.е. нельзя взять кусочек последовательности и узнать: какие именно яркости в нём закодированы без декодирования всех значений перед этим кусочком.

# Преобразование Хаара, 1909

Попробуем поступить иначе. Для этого разобьём все числа на пары и найдём полусуммы и полуразности значений в каждой из них:

(154, 155), (156, 157), (157, 157), (158, 156)
(154.5, 0.5), (156.5, 0.5), (157, 0.0), (157, -1.0)

Почему именно полусуммы и полуразности?

Зная полусумму a и полуразность *d* можно найти и сами значения:

первое значение в паре = a — d,
второе значение в паре = a + d.

# Преобразование Хаара. Где сжатие?

Полученные числа можно перегруппировать, разделив полусуммы и полуразности:

154.5, 156.5, 157, 157; 0.5, 0.5, 0.0, -1.0.

Числа во второй половине последовательности как правило будут небольшими. Почему так?

# Преобразование Хаара. Где сжатие?

В реальных изображениях соседние пиксели **редко** отличаются друг от друга значительно.

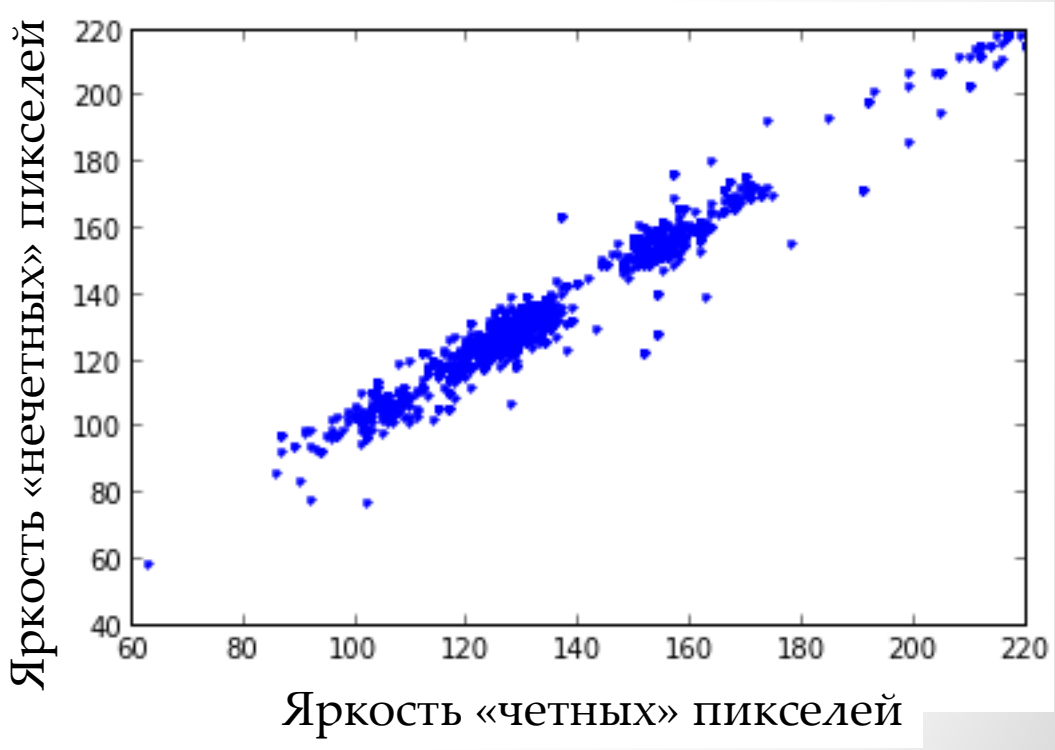
Если значение одного велико, то и другого велико, т.е. **пиксели коррелированы**.

# Преобразование Хаара. Где сжатие?

Рассмотрим первые 2000 пар соседних пикселей и каждую пару представим на графике точкой.

Практически во всех реальных изображениях точки выстраиваются вдоль одной прямой линии.

Верхний левый и нижний правый углы изображения практически всегда пусты.

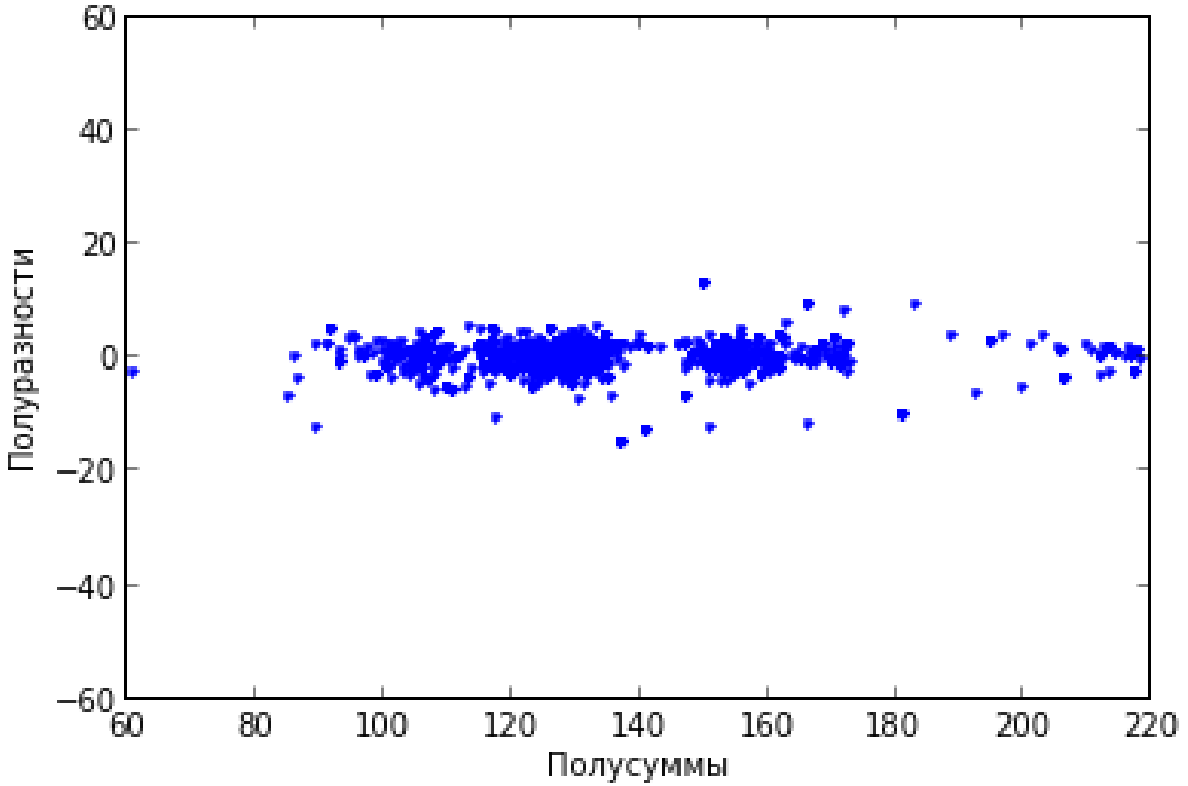


# Преобразование Хаара. Где сжатие?

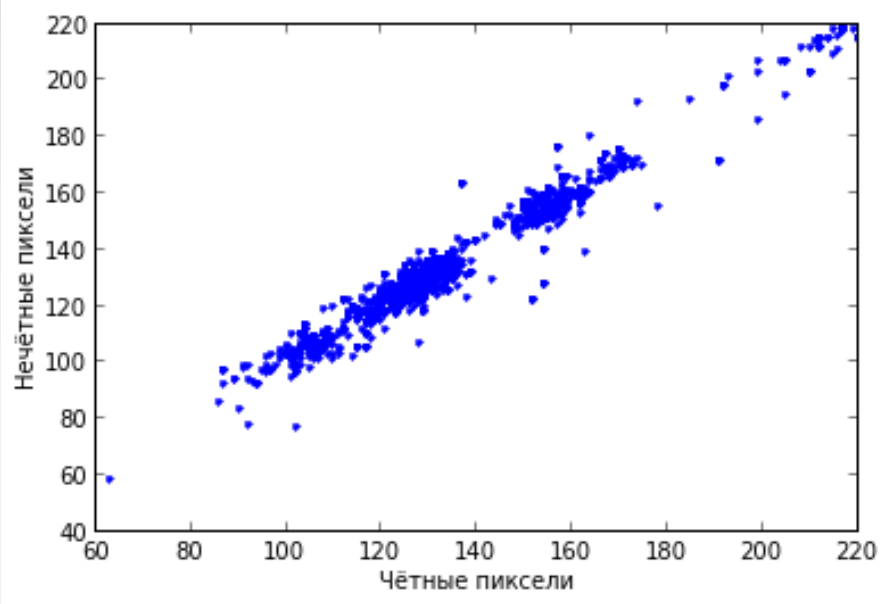
А теперь рассмотрим график, точками в котором будут полусуммы и полуразности.

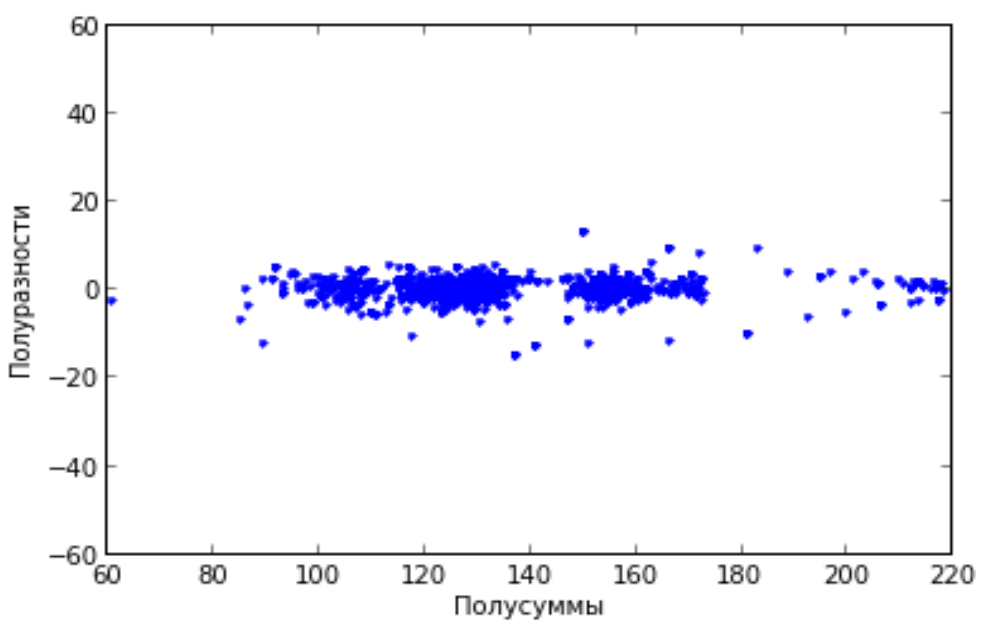
Полуразности находятся в более узком диапазоне значений.

На их кодирование можно потратить меньше одного байта.



# В чем отличие графиков?




**Рисунки из точек на двух графиках одинаковы. Разница лишь в повороте на угол в 45°.**

**То есть, преобразование Хаара — это просто аффинное преобразование - поворот точек таким образом, чтобы их можно было удобно и компактно закодировать.**

# Преобразование Хаара. Матричное представление

У нас есть пара пикселей (вектор) $\begin{pmatrix} x \\ y \end{pmatrix}$,
Надо получить пару $\begin{pmatrix} \frac{y+x}{2} \\ \frac{y-x}{2} \end{pmatrix}$.

Такое преобразование описывается матрицей:

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{y+x}{2} \\ \frac{y-x}{2} \end{pmatrix}$$

# Преобразование Хаара. Нормализация

При аффинных преобразованиях может меняться площадь фигуры. Коэффициент изменения площади равен определителю матрицы.

$$\det \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix} = \frac{1}{2} \cdot \frac{1}{2} - \left(-\frac{1}{2}\right) \cdot \frac{1}{2} = \frac{1}{2}$$

Чтобы определитель стал равен единице, надо умножить каждый элемент матрицы на $\sqrt{2}$.
На угол поворота (а значит, и на «**сжимающую способность**» преобразования) это не повлияет.

# Матрица для преобразования Хаара

В итоге получаем матрицу:

$$H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$$

**Как применить преобразование для всего изображения?**

В этом случае тоже можно описать преобразование матрицей, но большей по размеру. Диагональ этой матрицы будет состоять из матриц ***H***:

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & & & \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & & & \\ & & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \\ & & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \\ & & & & \ddots \end{pmatrix}$$

То есть исходный вектор просто обрабатывается независимо по парам.

# Обратное преобразование

Как известно, если у матрицы определитель не равен нулю, то для неё существует обратная матрица, «отменяющая» её действие.

Если мы найдём обратную матрицу для ***H***, то декодирование будет заключаться просто в умножении векторов с полусуммами и полуразностями на неё.

# Декодирование

Рассмотрим поближе нашу матрицу. Она состоит из двух вектор-строк: $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$ и $\begin{pmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$. Назовём их $v_1$ и $v_2$.

Их свойства:

1. Их длины равны 1, то есть $v_1 v_1^T = v_2 v_2^T = 1$ .

Умножение вектор-строки на транспонированный вектор-строку — это скалярное произведение.

2. Во-вторых, они ортогональны, то есть $v_1 v_2^T = v_2 v_1^T = 0$ .

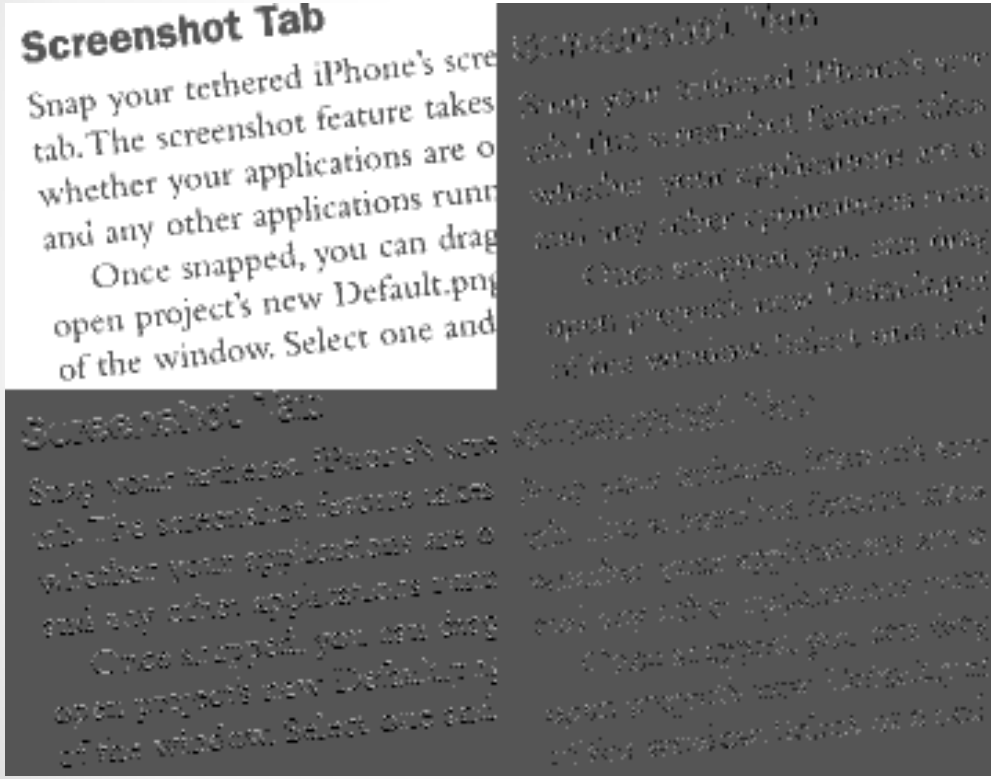
Матрица, строки которой обладают указанными свойствами называется **ортогональной**. Следовательно, обратную матрицу для них можно получить простым транспонированием:

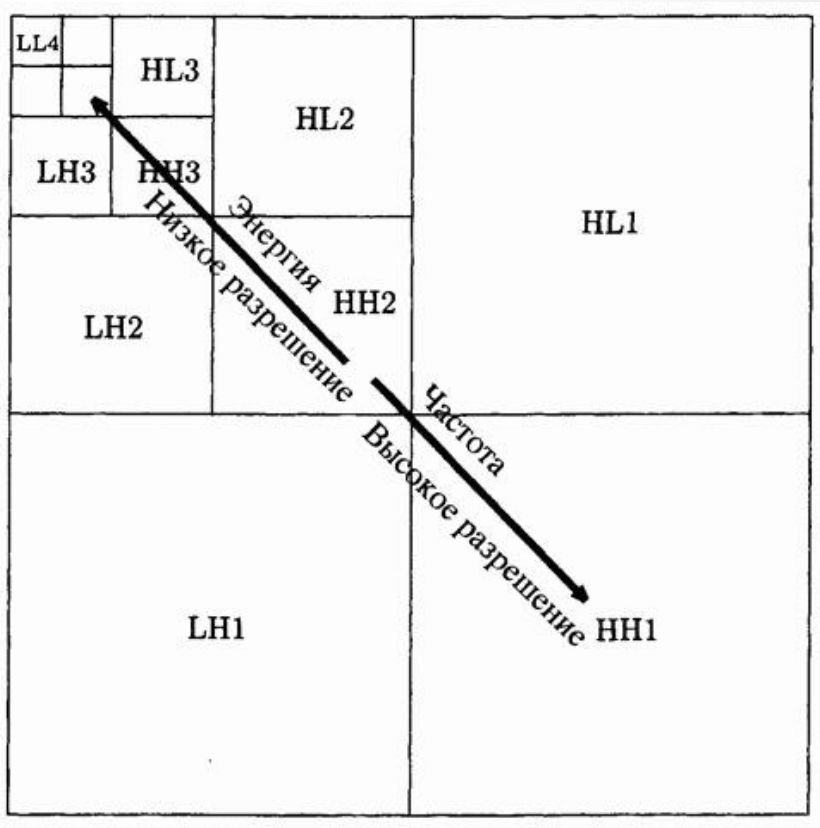
$$H^{-1} = H^T = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}^T = \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$$

# Что в итоге получили?

- Полученные «полусуммы»— это средние значения в парах пикселей. То есть значения «полусумм» — это уменьшенная копия исходного изображения! Уменьшенная потому, что «полусумм» в два раза меньше, чем исходных пикселей.
- Полусуммы **усредняют** значения яркостей, то есть «отфильтровывают» случайные всплески значений. Можно считать, что это некоторый **частотный фильтр**.
- Аналогично, разности «выделяют» среди значений межпиксельные «всплески» и устраняют константную составляющую. То есть, они «отфильтровывают» **низкие частоты**.

# Преобразование Хаара на изображении текста



# Преобразование Хаара

- Таким образом, преобразование Хаара — это пара фильтров, разделяющих сигнал на низкочастотную и высокочастотную составляющие.
- Чтобы получить исходный сигнал, нужно просто снова объединить эти составляющие.

# Что нам это даёт?

Пусть у нас есть фотография-портрет. Низкочастотная составляющая несёт в себе информацию об общей форме лица, о плавных перепадах яркости. Высокочастотная — это шум и мелкие детали.

При сжатии часть высокочастотных данных можно отбросить. Тем более, что, как мы выяснили, она обычно имеет меньшие значения, а значит более компактно кодируется.

Степень сжатия можно увеличить, применяя преобразование Хаара многократно. После повторного применения, высокочастотная информация будет занимать уже 75%.

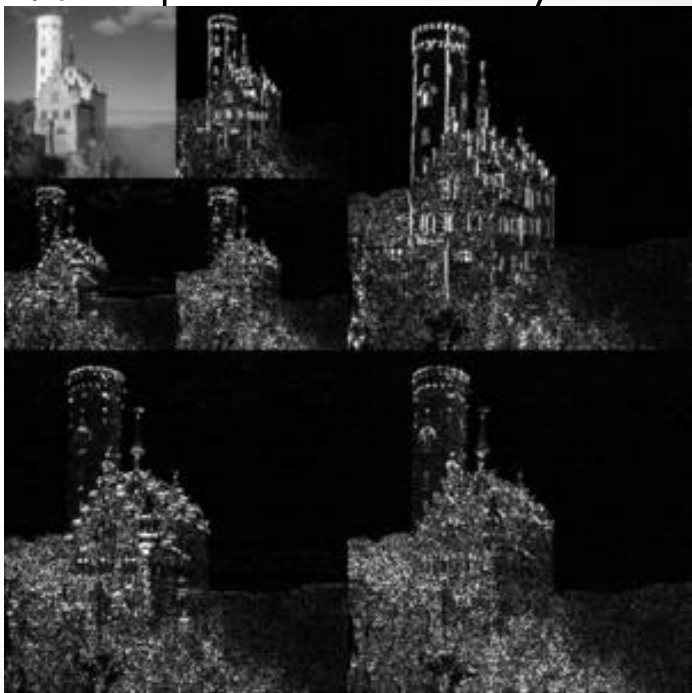
# Что нам это даёт?

Черные области соответствуют низкой яркости, то есть значениям, близким к нулю:

1. Их можно закодировать с большей эффективностью.
2. Их можно обнулить.

В алгоритме сжатия JPEG реализован подобный подход, только вместо преобразования Хаара используется дискретное косинус-преобразование.

Меняя число обнуляемых коэффициентов, можно регулировать степень сжатия!



# Когда преобразование Хаара будет давать наилучший результат?

Очевидно, когда изображение содержит длинные участки одинаковых значений яркости, тогда все разности обнулятся. Это может быть, например, рентгеновский снимок, отсканированный документ…

**Т.о. преобразование Хаара устраняет константную составляющую**.

Но всё же в реальных фотографиях областей с одинаковой яркостью не так много. Попробуем усовершенствовать преобразование, чтобы оно **обнуляло** ещё и **линейную составляющую**.

Эту задачу решила **Ингрид Добеши** — одна из создателей теории вейвлетов.

# Преобразование Добеши

Для нашего усовершенствованного преобразования уже будет мало двух точек. Поэтому будем брать по четыре значения, смещаясь каждый раз на два.

То есть, если исходная последовательность — 1, 2, 3, 4, 5, 6,…, N-1, N, то будем брать четвёрки (1, 2, 3, 4), (3, 4, 5, 6) и т. д. Последняя четвёрка «кусает последовательность за хвост»: (N-1, N, 1, 2).

Точно так же попробуем построить два фильтра: высокочастотный и низкочастотный. Каждую четвёрку будем заменять на два числа. Так как четвёрки перекрываются, то количество значений после преобразования не изменится.

# Преобразование Добеши

Для того, чтобы было удобно считать обратную матрицу потребуем также ортогональности преобразования. Тогда поиск обратной матрицы сведётся к транспонированию.

Пусть значения яркостей в четвёрке равны x, y, z, t. Тогда **первый фильтр** запишем в виде: $a = c_1 x + c_2 y + c_3 z + c_4 t$

Четыре коэффициента, образующих вектор-строку матрицы преобразования, пока нам неизвестны.

Чтобы вектор-строка коэффициентов **второго фильтра** был **ортогонален первому**, возьмём те же коэффициенты но переставим их и поменяем знаки:

$$d = c_4 x - c_3 y + c_2 z - c_1 t$$

# Преобразование Добеши

- Матрица преобразования будет иметь вид:

$$\begin{pmatrix} c_1 & c_2 & c_3 & c_4 & & & \\ c_4 & -c_3 & c_2 & -c_1 & & & \\ & & c_1 & c_2 & c_3 & c_4 & \\ & & c_4 & -c_3 & c_2 & -c_1 & \\ & & & & & & \ddots \end{pmatrix}$$

- Требование ортогональности выполняется для первой и второй строк автоматически. Потребуем, чтобы строки 1 и 3 тоже были ортогональны: $c_3 c_1 + c_4 c_2 = 0$

# Преобразование Добеши

- Векторы должны иметь единичную длину (иначе определитель будет не единичным)

$$c_1^2 + c_2^2 + c_3^2 + c_4^2 = 1$$

- Преобразование должно обнулять цепочку одинаковых значений (например, (1, 1, 1, 1)):

$$1c_4 - 1c_3 + 1c_2 - 1c_1 = 0$$

- Преобразование должно обнулять цепочку линейно растущих значений (например, (1, 2, 3, 4)):

$$1c_4 - 2c_3 + 3c_2 - 4c_1 = 0$$

# Преобразование Добеши

- Получили 4 уравнения, связывающие коэффициенты. Решая их, получаем:

$$c_1 = \frac{1+\sqrt{3}}{4\sqrt{2}}$$

$$c_2 = \frac{3+\sqrt{3}}{4\sqrt{2}}$$

$$c_3 = \frac{3-\sqrt{3}}{4\sqrt{2}}$$

$$c_4 = \frac{1-\sqrt{3}}{4\sqrt{2}}$$

# Преобразование Добеши

Подставив их в матрицу, получаем искомое преобразование.

После его применения к фотографиям получим больше нулей и малых коэффициентов, что позволит сжать изображение сильнее.

Это преобразование получило название вейвлета D4.

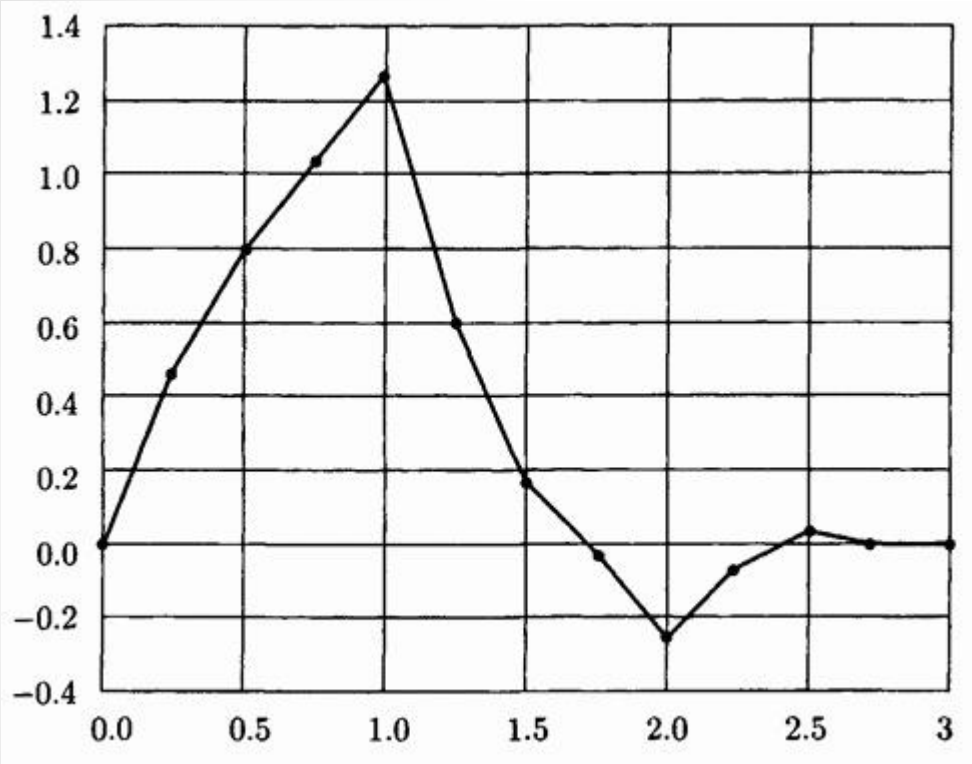
# Другие вейвлеты

Мы, конечно, можем не остановиться на этом, и потребовать устранения параболической составляющей (момент 2-го порядка) и так далее. В результате получим вейвлеты D6, D8…

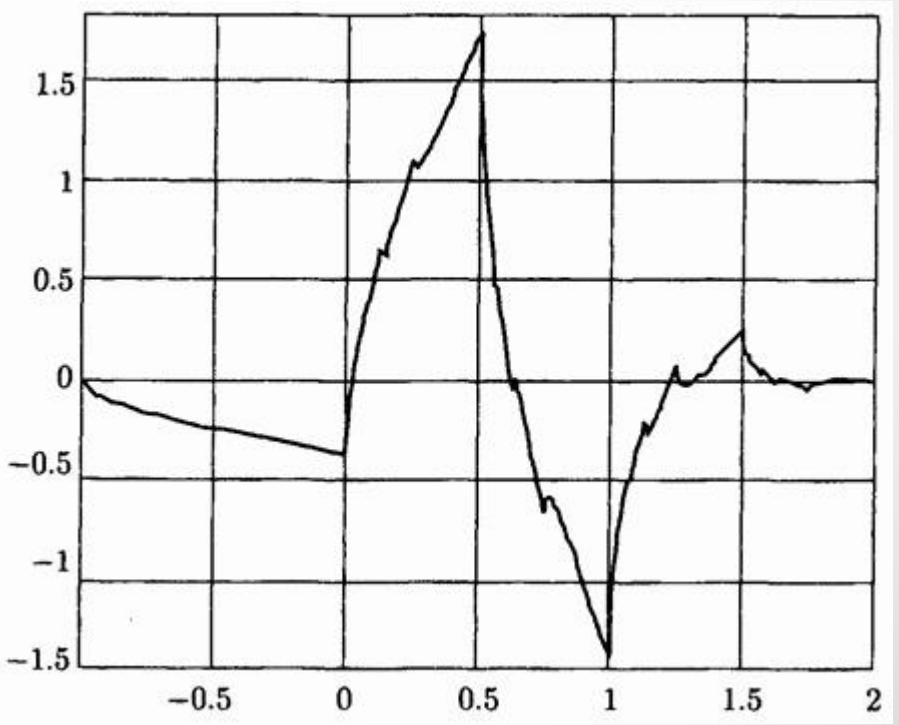
# Литература


1. Демьянович Ю.К., Ходаковский В.А. Введение в теорию вэйвлетов. Курс лекций.-СПб., 2007 г.
2. Яковлев А.Н. Введение в вейвлет-преобразования: Учеб. пособие. –Новосибирск: Изд-во НГТУ, 2003. – 104 с.
3. Уэлстид С. Фракталы и вейвлеты для сжатия изображений в действии. — М.: Триумф, 2003.
4. Штарк Г.-Г. Применение вейвлетов для ЦОС. — М.: Техносфера, 2007.
5. http://www.ptc.com/engineering-math-software/mathcad

# Преобразование Добеши




Функция шкалы Добеши в 13 точках.




Функция шкалы Добеши в 3073 точках.