1. ОДНОМЕРНЫЙ SSA

1.1 ПОСТРОЕНИЕ ТРАЕКТОРНОЙ МАТРИЦЫ

Первым шагом SSA является разбиение временного ряда $\boldsymbol{F}$ длинны $\boldsymbol{N}$ на последовательность векторов. Пусть целое число $\boldsymbol{L}$ - длина окна, $2 \le L \le N / 2$. Мы перемещаем это окно вдоль временного ряда F формируя вектор $X_i = (f_i, f_{i+1}, \dots, f_{i+L-1})$, где $i \in [0, N-L]$. То есть

$$\begin{gathered} X_0 = (f_0, f_1, f_2, \dots, f_{L-1})^T \\ X_1 = (f_1, f_2, f_3, \dots, f_L)^T \\ X_2 = (f_2, f_3, f_4, \dots, f_{L+1})^T \\ \vdots \\ X_{N-L} = (f_{N-L}, f_{N-L+1}, f_{N-L+2}, \dots, f_{N-1})^T. \end{gathered}$$

Эти вектора формируют L-траекторную матрицу $\boldsymbol{X}$ временного ряда F, где $\boldsymbol{K} = N - L + 1$ является количеством столбцов траекторной матрицы.

$$X = \begin{bmatrix} f_0 & f_1 & f_2 & & f_K \\ f_1 & f_2 & f_3 & \cdots & f_{K+1} \\ f_2 & f_3 & f_4 & & f_{K+2} \\ & \vdots & & \ddots & \vdots \\ f_{L-1} & f_L & f_{L+1} & \cdots & f_{N-1} \end{bmatrix} \quad (1)$$

Стоит отметить, что в матрице X на всех диагоналях, перпендикулярных главной, стоят равные элементы, следовательно она является ганкелевой матрицей.

1.2 СИНГУЛЯРНОЕ РАЗОЖЕНИЕ

Пусть $S = XX^T$. Выполним сингулярное разложение (Singular Value Decomposition, далее SVD) матрицы S. Пусть $\lambda = \{\lambda_1, \dots, \lambda_L\}$ – собственные числа матрицы S, взятые в неубывающем порядке и $U = \{U_1, \dots, U_L\}$ – ортонормированная система собственных векторов матрицы S, соответствующих собственным числам.

Пусть d равняется рангу матрицы X, но для простоты будем считать, что $d = L$. Сингулярное разложение матрицы X может быть представлено как

$$X = X_1 + X_2 + \cdots + X_d \quad (2)$$

Видно, что траекторная матрица X является суммой элементарных матриц X_i, где $i \in [1, L]$. X_i можно представить как:

$$X_i = \sqrt{\lambda_i} U_i V_i^T,$$

где

$$V_i = \frac{X^T U_i}{\sqrt{\lambda_i}}$$

Набор $(\sqrt{\lambda_i}, U_i, V_i)$ мы будем называть i-й собственной тройкой сингулярного разложения, а матрицы U и V

$$U = (U_1 U_2 \ldots U_L) \in R^{LxL}$$
$$V = (V_1 V_2 \ldots V_L) \in R^{KxL}$$

обозначаются как матрица эмпирических ортогональных функций и матрица главных компонент соответственно.

1.3 ГРУППИРОВККА

На основе разложения (2) процедура группировки делит все множество индексов $\{1, \ldots, d\}$ на m непересекающихся подмножеств $I_1, \ldots, I_m$.

Пусть $I = \{i_1, \ldots, i_p\}$. Тогда результирующая матрица X_i, соответствующая группе I, определяется как

$$X_I = X_{i_1} + \cdots + X_{i_p}.$$

Такие матрицы вычисляются для $I = I_1, \ldots, I_m$, тем самым разложение (2) может быть записано в сгруппированном виде

$$X = X_{I_1} + \cdots + X_{I_m}. \quad (3)$$

Типичной группировкой является та, где $m = L$, то есть $p = 1$, которая относится к случаю, когда каждый набор состоит только из одной компоненты. В общем случае вклад каждой матрицы X_I в траекторную матрицу в сингулярном разложении связан с ее собственными значениями и поэтому может быть получен как:

$$\eta_I = \frac{\sum_{i \in \mathrm{I}} \lambda_i}{\sum_{i=1}^{L} \lambda_i}. \tag{4}$$

1.4 ДИАГОНАЛЬНОЕ УСРЕДНЕНИЕ

Матрицы X_{I_m}, полученные путем группировки, не обязательно являются ганкелевыми матрицами, как это было с траекторной матрицей. Чтобы спроецировать каждую из этих матриц в одномерный сигнал, они должны быть ганкелезированы. Этот шаг выполняется путем получения среднего значения по всем побочным диагоналям каждого X_{I_m}.

Пусть $\boldsymbol{Y}$ – некоторая L x K матрица с элементами $y_{i,j}$, где $1 \le i \le L$, $1 \le j \le K$ и $N = L + K - 1$. Диагональное усреднение переводит матрицу Y в ряд $g_0, \dots, g_{N-1}$по формуле

$$g_k = \begin{cases} \dfrac{1}{k+1} \displaystyle\sum_{m=1}^{k+1} y_{m,k-m+2}\,, \text{для } 0 \le k < L-1, \\ \dfrac{1}{L} \displaystyle\sum_{m=1}^{L} y_{m,k-m+2}\,, \text{для } L-1 \le k < K, \\ \dfrac{1}{N-k} \displaystyle\sum_{m=k-K+2}^{N-K+1} y_{m,k-m+2}\,, \text{для } K \le k < N. \end{cases} \tag{5}$$

Применяя диагональное усреднение (5) к результирующим матрицам X_{I_k} мы получим ряды $\widetilde{F^{(k)}} = (\widetilde{f_0^{(k)}}, \dots, \widetilde{f_{N-1}^{(k)}})$, и, следовательно, исходный ряд $(f_0, \dots, f_{N-1})$ раскладывается в сумму m рядов:

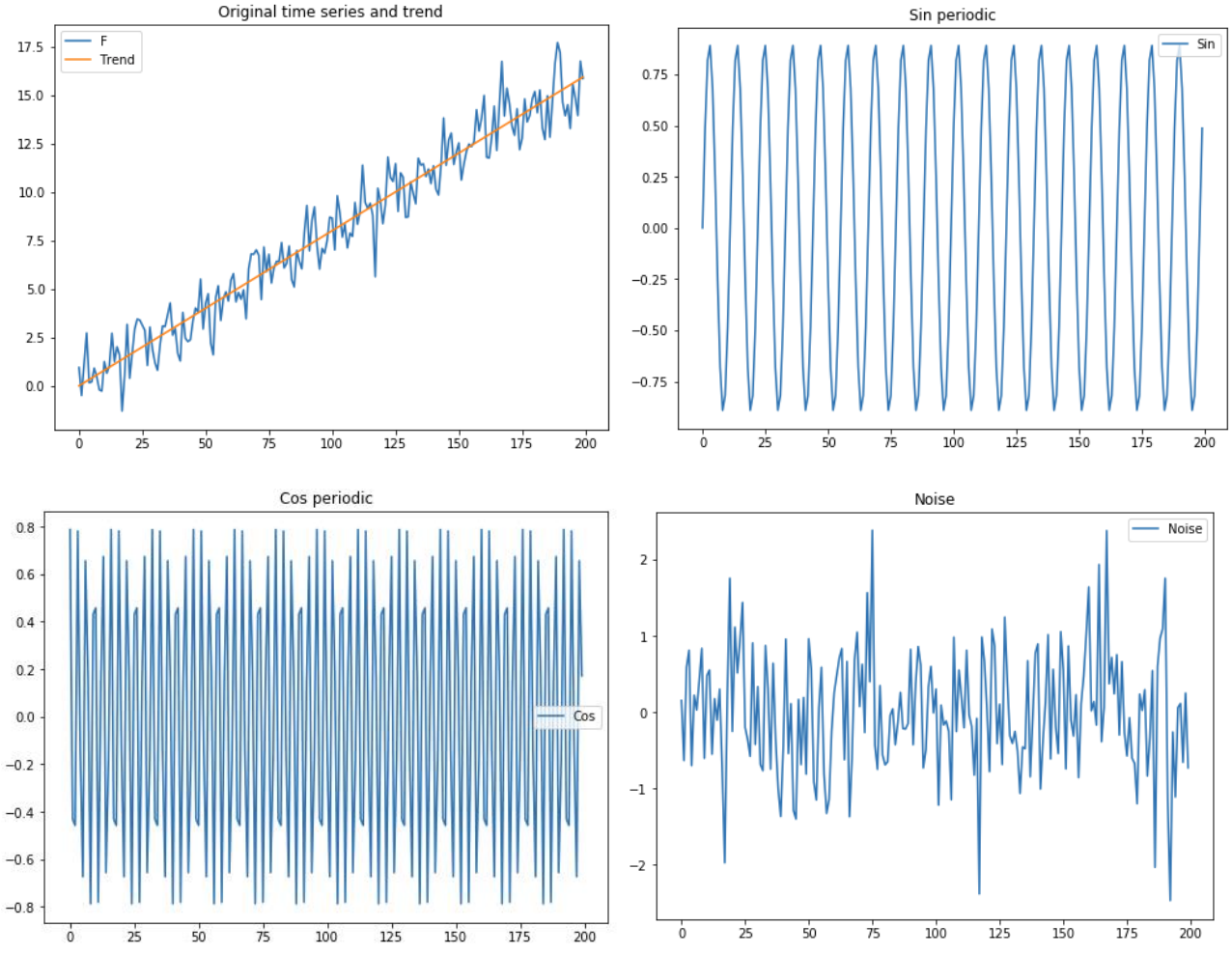
$$f_n = \sum_{k=1}^{m} \widetilde{f_n^{(k)}}$$

1.5 ТЕСТИРОВАНИЕ ОДНОМЕРНОГО SSA

Протестируем данный алгоритм на модельных данных. Зададим временной ряд:

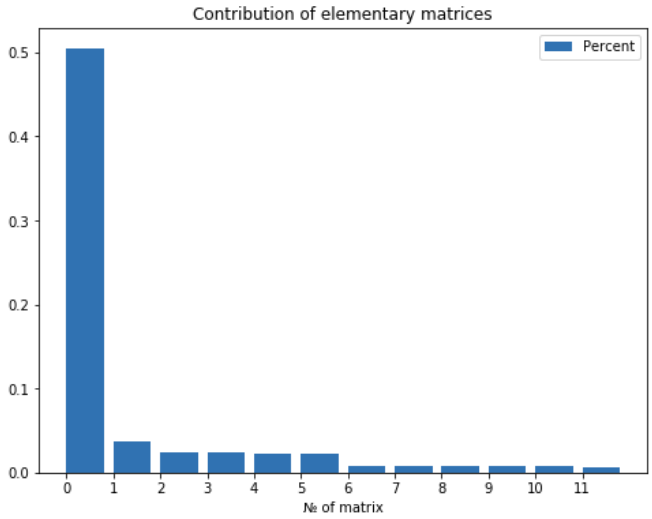
$$f(t) = 0{,}08t + 0{,}9sin\left(\frac{2\pi t}{11}\right) + 0{,}8cos\left(\frac{5\pi(t + 0{,}09)}{8}\right) + \text{noise},$$

где noise – случайный шум, распределенный по нормальному закону распределения. Сгенерируем ряд размером $N = 200$.

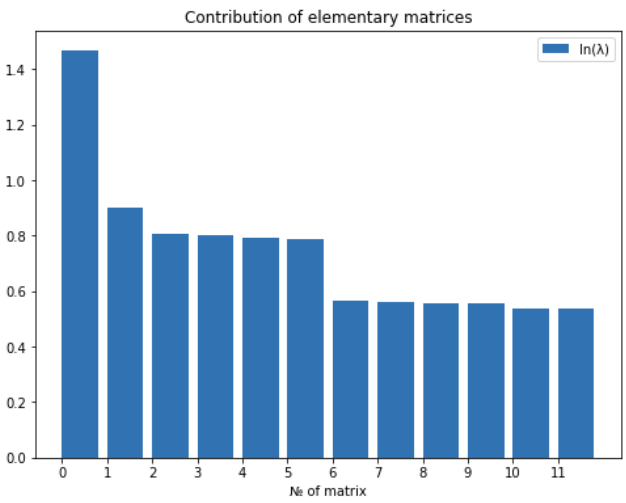


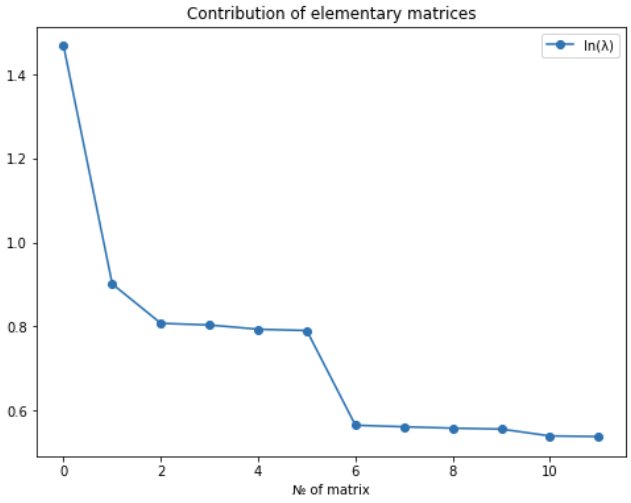
Применим SSA с размером окна $L = 90$ к сгенерированному ряду. Построим траекторную матрицу, произведем SVD разложение, и, для выполнения шага

группировки, посчитаем вклад первых 12 элементарных матриц во временной ряд по формуле (4).



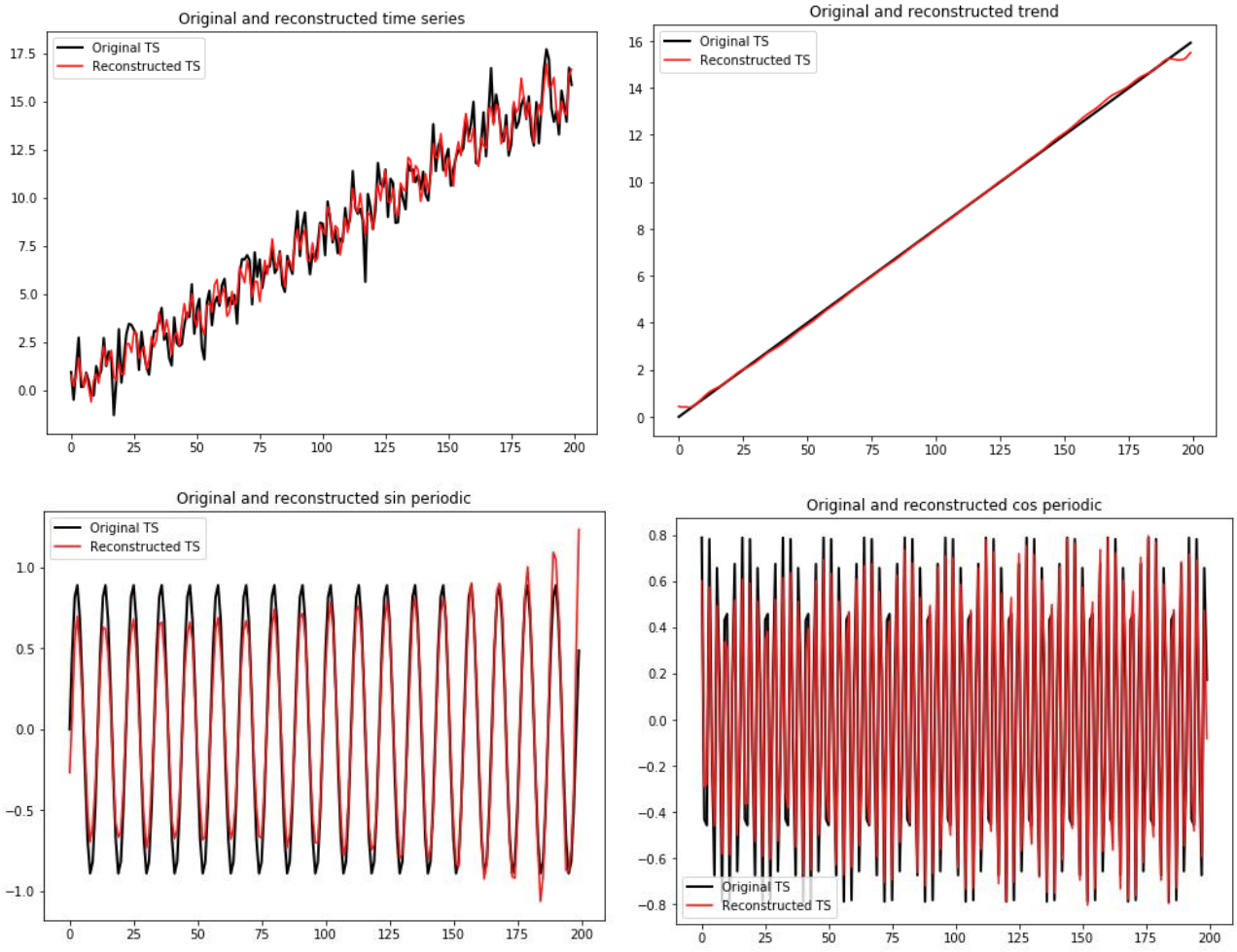
Для более удобного представления вклада матриц в ряд можно взять натуральный логарифм собственных значений матриц.





Из графиков видно, что содержательный смысл несут первые 5 элементарных матриц, остальные, вероятно, шум. Можно предположить, что нулевая и первая матрицы являются трендом, 2 и 3 — первой периодической составляющей, 4 и 5 — второй, остальное — шум. Складывая элементарные матрицы, которые находятся на одном уровне на графике, мы сможем восстановить компоненты ряда.

Восстановим ряд и его компоненты:



Как видно из графиков, SSA отфильтровал изначальный временной ряд и вполне успешно восстановил его компоненты.

2. ДВУМЕРНЫЙ SSA

Для фильтрации и анализа изображений используется расширенный вариант SSA – двухмерный SSA (2D SSA). Далее будет описан алгоритма 2D SSA и рассмотрены его отличия от 1D SSA.

2.1 ПОСТРОЕНИЕ ТРАЕКТОРНОЙ МАТРИЦЫ

Предположим, что у нас есть изображение I размера h × w, представленное в виде матрицы

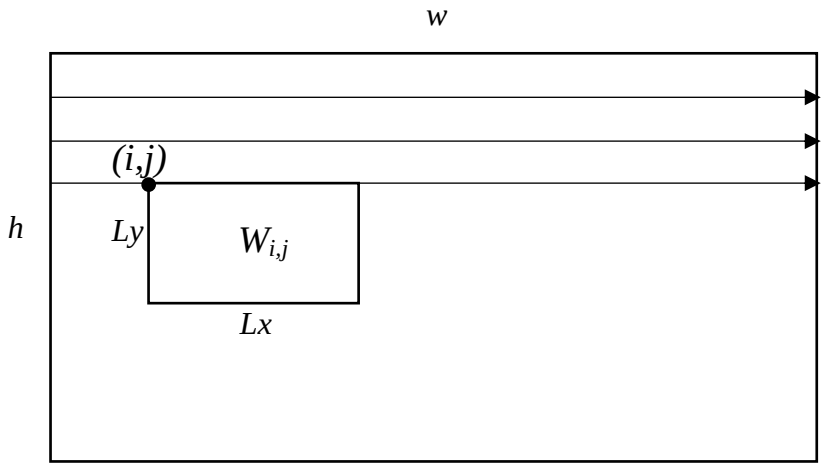
$$I = \left\| I_{i,j} \right\|_{\substack{i=1,\dots,h \\ j=1,\dots,w}}$$

где, например, значения $I_{i,j}$ обозначают интенсивность цвета или уровня серого, $0 \leq I_{i,j} \leq 255$.

Как и в случае с одномерным SSA, нам нужно определить окно, которое будет перемещаться по изображению. В отличие от одномерного случая, наше окно имеет не только ширину, но и высоту. Пусть окно имеет размер Ly × Lx, $1 \leq Ly \leq h, 1 \leq Lx \leq w$, и $Ky = h - Ly + 1, Kx = w - Lx + 1$.

Назовем верхнюю левую точку окна как «ориентировочная точка окна»; диапозоном всех ориентировочных точекявляется (i, j), где $1 \leq i \leq Ky, 1 \leq j \leq Kx$. Окно движется слева направо, сверху вниз и проходит всю матрицу I. Мы получаем окно Wi,j, где ориентировачные точкт покрывают регион изображения размером Ly × L:

$$W_{i,j} = \left\| I_{i+k-1,j+l-1} \right\|_{\substack{k=1,\dots,Ly \\ l=1,\dots,Lx}}$$



Далее элементы всех окон W преобразуются в векторы с помощью операции vec; то есть, записывая строки одну за другой последовательно в вектор:

$$\overrightarrow{W} = vec(W) = (W_1^T, W_2^T, \dots, W_{Ly}^T)^T,$$

где Wi – i-я строка окна W. Таким образом, все окна Wi,j преобразуются в векторы

$$\overrightarrow{W_{i,j}} = vec(W_{i,j})$$
$$= (I_{i,j}, I_{i,j+1}, \dots, I_{i,j+Lx-1}, I_{i+1,j}, \dots, I_{i+1,j+Lx-1}, \dots, I_{i+Ly-1,j+Lx-1})^T \in R^{LyLx}.$$

Последним шагом траекторная матрица X размером p x q, где $p = LyLx$ и $q = KyKx, p \leq q$, составляется из векторов $\overrightarrow{W_{i,j}}$ следующим образом:

$$X = (\overrightarrow{W_{1,1}}, \overrightarrow{W_{1,2}}, \dots, \overrightarrow{W_{1,Kx}}, \overrightarrow{W_{2,1}}, \dots, \overrightarrow{W_{2,Kx}}, \dots, \overrightarrow{W_{Ky,Kx}}) \quad (6)$$

Процедура формирования траекторной матрицы X представлена ниже для изображения I размера h x w и окном размера 2 x 2:

$$\begin{pmatrix} \boxed{I_{1,1}} & \boxed{I_{1,2}} & I_{1,3} & \dots \\ \underline{\boxed{I_{2,1}}} & \underline{\boxed{I_{2,2}}} & I_{2,3} & \dots \\ I_{3,1} & I_{3,2} & I_{3,3} & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} \to \begin{pmatrix} I_{1,1} & \boxed{I_{1,2}} & \boxed{I_{1,3}} & \dots \\ I_{2,1} & \underline{\boxed{I_{2,2}}} & \underline{\boxed{I_{2,3}}} & \dots \\ I_{3,1} & I_{3,2} & I_{3,3} & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} \to \dots$$

$$\to \begin{pmatrix} \dots & I_{1,w-2} & \boxed{I_{1,w-1}} & \boxed{I_{1,w}} \\ \dots & I_{2,w-2} & \underline{\boxed{I_{2,w-1}}} & \underline{\boxed{I_{2,w}}} \\ \dots & I_{3,w-2} & I_{3,w-1} & I_{3,w} \\ \dots & \dots & \dots & \dots \end{pmatrix} \to \dots \to \begin{pmatrix} \dots & \dots & \dots & \dots \\ \dots & I_{h-2,w-2} & I_{h-2,w-1} & I_{h-2,w} \\ \dots & I_{h-1,w-2} & \boxed{I_{h-1,w-1}} & \boxed{I_{h-1,w}} \\ \dots & I_{h,w-2} & \underline{\boxed{I_{h,w-1}}} & \underline{\boxed{I_{h,w}}} \end{pmatrix}$$

Что дает траекторную матрицу X:

$$X = \begin{pmatrix} I_{1,1} & I_{1,2} & \dots & I_{1,w-1} & \dots & I_{h-1,w-1} \\ I_{1,2} & I_{1,3} & \dots & I_{1,w} & \dots & I_{h-1,w} \\ I_{2,1} & I_{2,2} & \dots & I_{2,w-1} & \dots & I_{h,w-1} \\ I_{2,2} & I_{2,3} & \dots & I_{2,w} & \dots & I_{h,w} \end{pmatrix}$$

Отметим, что один и тот же элемент Ii,j появляется несколько раз (до p раз) в траекторной матрице X, так что X имеет определенную структуру, которая немного отличается от структуры траекторной матрицы в 1D SSA; эта структура известна как ганкель – блок – ганкель.

2.2 СИНГУЛЯРНОЕ РАЗЛОЖЕНИЕ И ГРУППИРОВКА

Выполнение шагов сингулярного разложения и группировки в 2D-SSA ничем не отличаются от выполнения этих шагов в 1D-SSA. После выполнения этих шагов мы должны получить матрицу эмпирических ортогональных функций U, матрицу главных компонент V, вектор собственных чисел λ и сгруппированные матрицы X_{I_m}.

2.3 РЕКОНСТРУКЦИЯ ИЗОБРАЖЕНИЯ

Для реконструкции изображения расчитаем матрицу

$$\tilde{X} = \sum_{k=1}^{m} V_{ik} U_{ik} X$$

как приближение к траекторной матрице X. Преобразование матрицы $\tilde{X}$ в изображение $\tilde{I}$ размера h x w с помощью модифицированного диагонального усреднения (5).

Для диагонального усреднения матрицы $\tilde{X}$ преобразуем ее в матрицу размером Ly x Ky

$$\tilde{X} = \begin{pmatrix} H_0 & H_1 & H_2 & \dots & H_{Ky-1} \\ H_1 & H_2 & H_3 & \dots & H_{Ky} \\ H_2 & H_3 & H_4 & \dots & H_{Ky+1} \\ \dots & \dots & \dots & \dots & \dots \\ H_{Ly-1} & H_{Ly} & H_{Ly+1} & \dots & H_{Ly+Ky-1} \end{pmatrix},$$

где

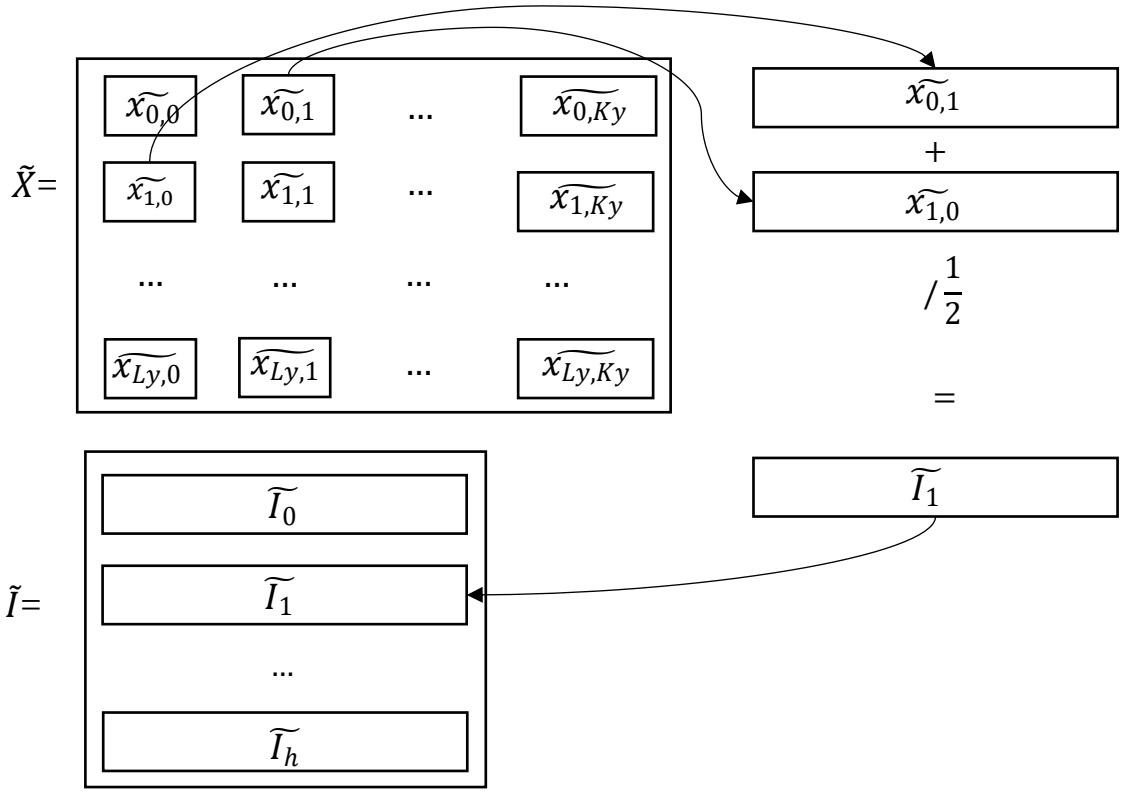
$$H_j = \begin{pmatrix} I_{0,j} & I_{1,j} & I_{2,j} & \dots & I_{Kx-1,j} \\ I_{1,j} & I_{2,j} & I_{3,j} & \dots & I_{Kx,j} \\ I_{2,j} & I_{3,j} & I_{4,j} & \dots & I_{Kx+1,j} \\ \dots & \dots & \dots & \dots & \dots \\ \tilde{x}_{Lx-1,j} & I_{Lx,j} & I_{Lx+1,j} & \dots & I_{Lx+Kx-1,j} \end{pmatrix}$$

матрица размером Lx x Kx. То есть мы делим матрицу $\tilde{X}$ на LyKy матриц.

Далее мы применяем диагональное усреднение (5) к каждой Hj и переводим эти матрицы в вектора длинной $w = Lx + Kx - 1$. Последним шагом мы применяем диагональное усреднение к $\tilde{X}$:

$$\tilde{I}_s = \begin{cases} \frac{1}{s+1}\sum_{l=0}^{s+1} \tilde{x}_{l,s-l}, & \text{для } 0 \le s < Ly-1, \\ \frac{1}{Ly}\sum_{l=0}^{Ly} \tilde{x}_{l,s-l}, & \text{для } Ly-1 \le s < Ky, \\ \frac{1}{Ky+Ly-s-1}\sum_{l=s-Ky+1}^{Ly} \tilde{x}_{l,s-l}, & \text{для } Ky \le s < Ky+Ly-1. \end{cases},$$

где $\tilde{x}$ – элемент матрицы $\tilde{X}$. Далее на изображении показан принцип реконструкции изображения.

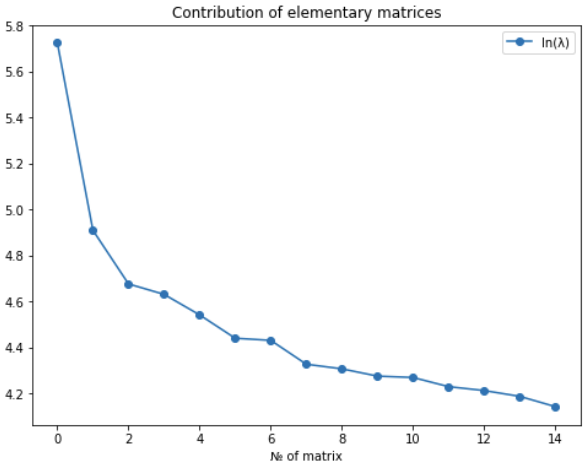


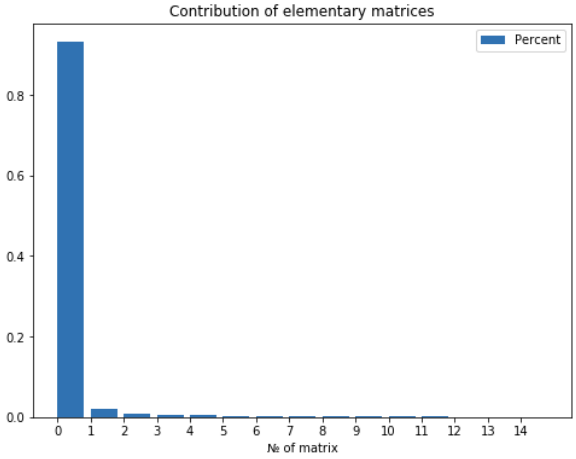
2.4 ТЕСТИРОВАНИЕ ДВУМЕРНОГО SSA

Для тестирования возьмём несколько изображений и применим к ним алгоритм.

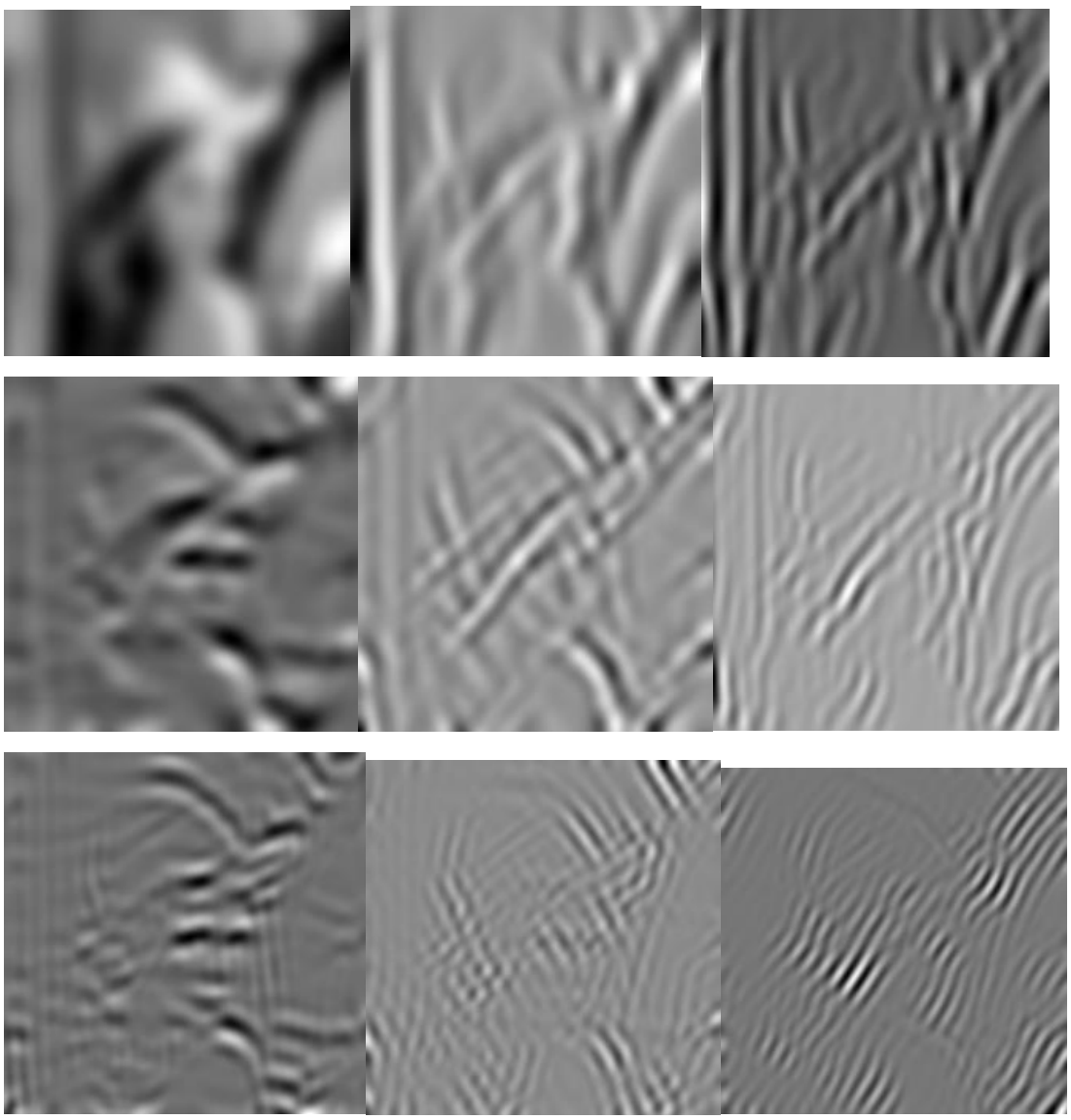
Применим алгоритм для разложения и восстановления изображения размером 256 x 256 пикселей, используя окно размером 20 x 20.


Для восстановления изображения мы использовали 10 первых элементарных матриц. Проанализируем вклад матриц в изображение





и рассмотрим изображения, восстановленные используя только одну элементарную матрицу

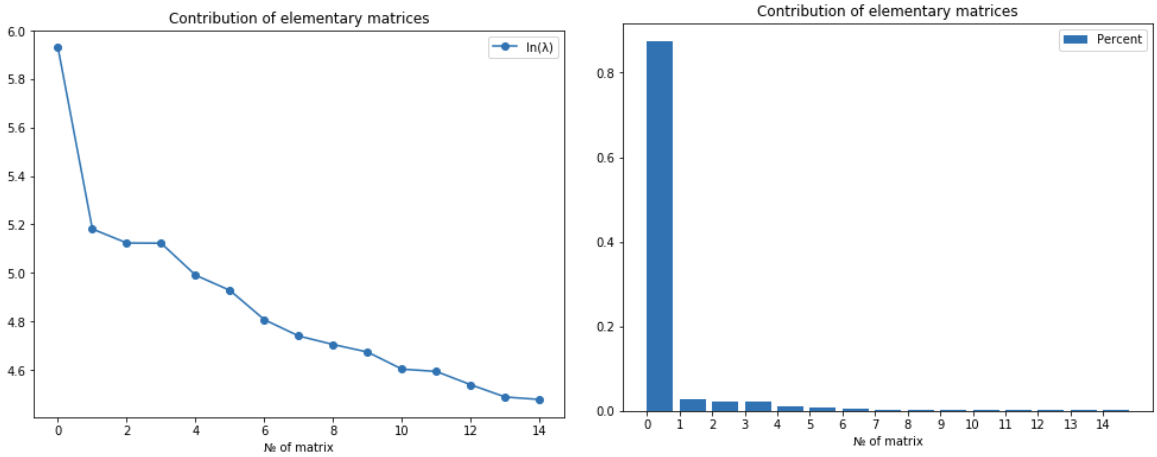


0-5 10-12

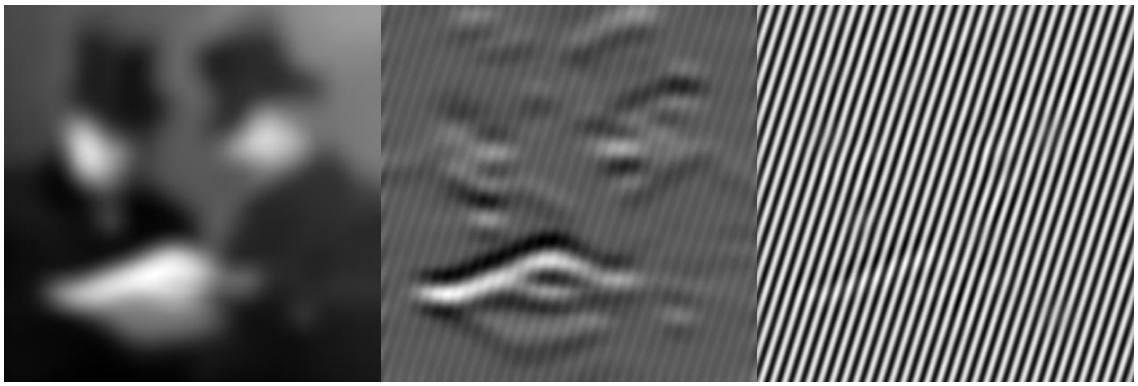
Как видно из графиков и восстановленных изображений наибольший вклад в конечное изображение вносит нулевая элементарная матрица, элементарные матрицы до 12-й похожи на контуры изображения, остальное, вероятно, шум.

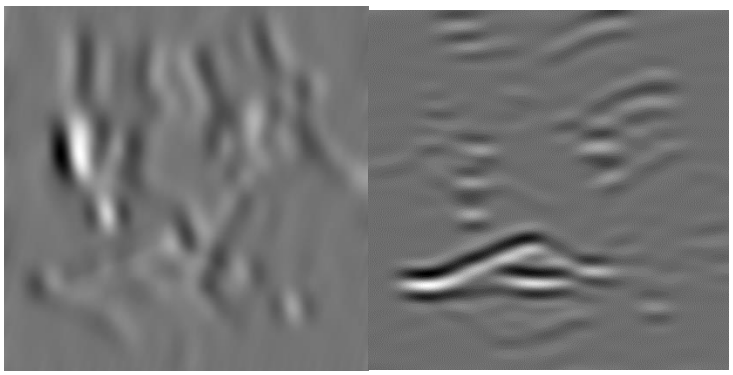
Применим алгоритм для разложения и фильтрации изображения размером 420x420 px с шумом:

Применим SSA с окном 30x30. Проанализируем вклад матриц в изображение.



Судя по графикам, 2-я и 3-я матрицы должны быть в одной группе.





0,1,2+3,4,5

Видно, что 1,2 и 3 элементарные матрицы являются шумом. Восстановим изображение без этих компонент

Таким образом нам удалось избавиться от шума, но качество изображения пострадало.

2.5 ИСПОЛЬЗОВАНИЕ SSA ДЛЯ ОБНАРУЖЕНИЯ ТЕКСТУР

Предположим, что у нас есть изображение I размером HxW на котором необходимо найти текстуру и изображение T размером hxw с текстурой, которую необходимо найти. Для локализации текстуры необходимо применить первые два шага SSA к I и к T. Размер окна для обоих изображений должен быть одинаковым.

После успешного выполнения этих шагов мы должны получить ковариационные матрицы XI и XT, матрицы эмпирических ортогональных функций UI и UT и матрицы главных компонент VI и VT для изображений I и T соответственно.

Далее рассчитываем приближение к XI матрицы VT и UT изображения с текстурой

$$\tilde{X} = \sum_{k=1}^{m} VT_{ik} UT_{ik} XI$$

и восстанавливаем изображение.

В качестве примера возьмем изображение размера 512x512 и сэмпл с текстурой размером 40x40.

Применим к ним алгоритм с размером окна 7x7и восстановим изображение