Interpolation and masking effects of heteroassociative compressive transformations

M Dryuchenko and A Sirota
Department of Computer Science, Voronezh State University, 1, University Square, Voronezh, 394018, Russia
E-mail: m_dryuchenko@mail.ru

Abstract. The article considers heteroassociative image transformations, which perform mutual mapping of two neighbouring or nested regions of arbitrary shape within rectangular fragments or blocks covering the image. Specifically, the study focuses on heteroassociative compressive transformations. In the article, we provide theoretical justification of various types of heteroassociative transformations performed using neural network based converters and numerical methods. We also study interpolation and masking effects of the transformations, which can be observed during image processing. These effects are accounted for by the possibility to represent an image as a sum of the deterministic component of the output (which is either linear of nonlinear function of the input) and the additive stochastic component of the output responsible for the prediction error. The article analyses occurrence, genetic and heuristic algorithms used to optimise the input and output configuration of the fragments covering the image. The algorithms used allowed us to maximise (minimise) the deterministic and the stochastic component and thus regulate interpolation and masking effects of heteroassociative transformations.

1. Introduction
Let the initial model of the processed multidimensional signal or image be its random field realization on a discrete grid \( w(x, y) \), \((x, y) \in \Psi = \{x = 1, n, y = 1, m \}\). In the general case, \( w(x, y) \in R^h \), where \( h \geq 1 \) determines the size of the data vector connected to each point \((x, y)\). Specifically, if \( h = 1 \), then \( w(x, y) \) can be interpreted as a monochromatic image. If \( h = 3 \), then \( w(x, y) \) is interpreted as a colour image. Let \( z \in R^N \) be a random vector of a certain subregion (fragment) \( \Omega \subset \Psi \) determined by unwrapping \( w(x, y) \), \((x, y) \in \Omega \) in a certain order \( z = S_A(w) \), where \( S_A(\ldots) \) is the function determining the unwrapping order. For instance, if \( \Omega \) is a rectangular subregion of size \( n_w \times m_w \), then \( N = n_w \times m_w \times h \).

To make it clear, let the mathematical expectation and the covariance matrix of vector \( z \) be: \( M[z] = 0 \), \( M[zz^T] = R_z \). Vector \( z \) can always be presented as a composite vector \( z = (z_1^T, z_2^T)^T \), where \( z_1 \in R^{N_1} \) represents a subregion \( \Omega_I \subset \Omega \) of the fragment \( \Omega \subset \Psi \), called the input, and \( z_2 \in R^{N_2} \) represents a subregion \( \Omega_O \subset \Omega \), called the output. At the same time, let \( \Omega_I \cup \Omega_O = \Omega \), \( \Omega_I \cap \Omega_O = \emptyset \). Our task is to create universal heteroassociative mappings

\[ F : Z_1 \rightarrow Z_{2/1}, \quad z_{2/1} = F(z_1), \quad z_1 \in Z_1, \quad z_{2/1} \in Z_{2/1}, \]

i.e. a transformation that maps the input data to the output data. It is obvious that \( z_{2/1} \)
always has a certain representation error

\[ z_2 = F(z_1) + V = z_{2/1} + V, \]  

(1)

where \( V \) is a stochastic component (a random vector) that determines the representation error.

The said transformations can be performed either in full (1), i.e. using, within the accepted model, all the information about \( z_2 \) with known \( z_1 \), or with a certain degree of information compression. We will further refer to the former transformations as heteroassociative transformations (HT). Transformations involving compression will be referred to as heteroassociative compressive transformations (HCT). The latter will be expressed as

\[ \tilde{F} : Z_1 \to Z_{2/1}', \quad z_{2/1}' = \tilde{F}(z_1), \quad z_1 \in Z_1, \quad z_{2/1}' \in Z_{2/1}', \]  

(2)

where \( z_{2/1}' = \tilde{F}(z_1) \) determines the approximate representation of \( z_{2/1} = F(z_1) \) in (1) during the compression, which, of course, results in additional error.

If \( z_1 = z_2 = z \) and regions \( \Omega_I = \Omega, \quad \Omega_O = \Omega \) coincide, the transformation is called autoassociative. Autoassociative transformation is an individual case of heteroassociative compressive transformation. It is obvious, that in this case only compressive transformation is possible, i.e. autoassociative compressive transformation (ACT) expressed as follows

\[ \bar{F} : Z \to \bar{Z} \subseteq Z, \quad \bar{z} = \bar{F}(z), \quad z \in Z, \quad \bar{z} \in \bar{Z}. \]  

(3)

Generally, during HCT, only the deterministic component of \( z_{2/1}' \) is compressed, while vector \( z^{(p)}_2 \) remains unchanged. The fragments are actually compressed only when ACT is performed.

Suppose that several non-overlapping regions \( \Omega \) with similar topology completely cover \( \Psi \), so that

\[ P \bigcap_{p=1}^P \Omega^{(p)} = \Psi, \quad \bigcup_{p=1}^P \Omega^{(p)} = \emptyset. \]

Hence, each region \( \Omega^{(p)} \) corresponds to the realisation of vector: \( z^{(p)} \). We can thus determine a set of realisations \( \{z^{(p)}_1, z^{(p)}_2, p = 1, P\} \) for the inputs and the outputs \( \Omega^{(p)}_I \cup \Omega^{(p)}_O = \Omega^{(p)} \) of the set of fragments \( \Omega^{(p)}, \quad p = 1, P \), which will be used as a training set for compressive transformation.

For universal HT, regions \( \Omega_I, \Omega_O \) may be of arbitrary configuration: rectangular, arbitrary grids inside a rectangular region, etc. It is also important, that in order to describe HT and HCT of random vectors \( z_1, z_2 \) both linear and nonlinear transformation models can be used.

In [1–3] we provided a theoretical justification and detailed the properties of HT and HCT, as well as the advantages of applying these transformations to various image processing problems. The said properties can be divided into three groups.

- Interpolation and masking effects can be observed when performing HT and HCT of images. These effects are accounted for by the possibility to represent any fragment of a function of multiple random variables as a sum of the deterministic component of the output (which is either linear or nonlinear function of the input) and the additive stochastic component of the output responsible for the prediction error.

- The reconstruction effect of HT which determines the possibility of embedding and subsequent restoration of signals or whole images. In this case, the compression does not distort the stored or transmitted data (lossless compression).

- The compression effect of HT and HCT which determines the possibility to modify signals and images by removing the spectral components using various algorithms and methods depending on their structure (lossy compression).
These properties can find various applications. Thus, performing HT and HCT based on the equations and algorithms suggested in [1–3], we can distort (modify) both the deterministic and the stochastic component. Such modifications of the deterministic component are used, for instance, in steganography and data compression problems [2, 3]. The stochastic component serves as a noise that masks the distortions, while the modified deterministic component or its part serves as a useful signal. Therefore, when analysing interpolation and masking effects of HCT, it is important to account for the ratio of the “energy” characteristics of the deterministic (interpolation) and the stochastic component formed during HT without compression.

Thus, the purpose of our study was to find a way to “control” the ratio of “energy” characteristics of the deterministic and the stochastic component. The solution involved determining the input and output configurations that would maximise (minimise) the stochastic component in proportion to minimisation (maximisation) of the deterministic component.

2. Materials and methods

2.1. Theoretical justification of heteroassociative compressive transformations

In [1, 3] we demonstrated that feedforward neural networks (ANN) can be used as universal data converters, when performing HT and HCT based on both linear and nonlinear models. The weights of such networks can be either calculated or adjusted by backpropagation training. Fig.1a presents a standard architecture of an ANN that can be used for HT. ANNs used for HCT usually have the architecture of a heteroassociative encoder with a reduced number of neurons \( M \leq \min\{N_1, N_2\} \) in the hidden layer (Fig.1b). Depending on the model used, the activation functions can be either linear and nonlinear (as shown in the figures), or only linear.

In [1,3] we determined the required minimum of the objective function (the mean squared error) for heteroassociative converters based on a linear model. It was presented in the form of linear matrix equations for the weights of two-layer neural networks. We also studied the estimates of the converters’ parameters and proved two theorems regarding the equivalence of

Figure 1. Architectures of the ANN-based data converters used in heteroassociative transformations.
HCT performed using ANN-based converters.

According to the first theorem, when applied to a vector describing the input of a random field fragment, HCT (Fig.1b) with minimum mean squared error at the output is equivalent to the resolution of the eigen vectors of the matrix that is the product of the sample inverse covariance matrix of the fragment’s input, the cross-covariance matrix of the input and output matrices, and its transpose. Columns of the weighting matrix of the first layer are a linear combination $M$ of the eigen vectors of the said matrix corresponding to maximum eigen values. Columns of the weighting matrix of the second layer are a linear combination $M$ of the eigen vectors of the sample covariance matrix of the optimal linear estimate of the output.

The second theorem states (Fig.1c) that, when applied to the vector describing the input of a random field fragment, HCT with minimum mean squared error at the output is equivalent (with regard to the degree of residual error) to autoassociative compressive transformation of the deterministic component of the output, with regard to the input.

The obtained results demonstrated that it is possible to apply universal linear contraction mappings to both random field and real image processing with minimal distortion. They also showed that there are two ways to perform heteroassociative compressive transformations.

The first way is to perform the compression in the original image (directly) using the architecture of the converter presented in Fig.1b. This can be done either by training the ANN with the said architecture, or by calculating the corresponding weighing coefficients based on the generalized eigenvalues and eigenvectors problem.

The second way is to perform the compression of the deterministic component (indirectly) using the architecture shown in Fig.1c and linear autoassociative compression of the reaction vector obtained at the output of the ANN presented in Fig.1a. This can be done either by means of two-stage training of the ANN, or by directly calculating the weights of the autoassociative converter of the linear estimate based on the problem of eigenvalues and eigenvectors of the covariance matrix of the output estimate.

For the nonlinear models, we determined the required minimum of the objective function with regard to the weighting matrices for converters with sigmoid activation functions. They are presented as nonlinear matrix equations. Initial approximations of these equations for the converters presented in Fig.1 yield linear matrix equations similar to the ratios obtained for linear converters. More accurate numerical solutions to the said linear equations are obtained during backpropagation training of ANNs based on one of the existing methods.

2.2. Integral heteroassociative transformations

Applying the above described transformations (HT or HCT) to the implementation of a random function (image), we can determine its deterministic component $E(x, y) = \hat{w}(x, y)$ for the whole range. It should be considered as an interpolated function determined using the known inputs of all the fragments (blocks) covering the range of the function $\Psi$. The interpolation results differ for every image, since all the transformations (1) are based on a specific training set. In the same way, we can determine the stochastic component for the whole image $V(x, y) = w(x, y) - \hat{w}(x, y)$, i.e. decompose the image into two independent components different from each other in their correlation and frequency properties. The partitioning is performed in all the fragments covering the image. This is done in order to perform direct and backward integral heteroassociative transformations (IHT). Direct transformation involves transformation of the input into the output. Backward transformation involves transformation of the output into the input. To ensure the statistical homogeneity of the deterministic component and the stochastic masking component, the input and the output should be of the same size and configuration. Now, that we can use direct and backward transformations, let us redefine previously introduced mapping of the input to the output $F : Z_1 \rightarrow Z_{2/1}, \hat{z}_{2/1} = F(z_1)$, as $F_{io} : Z_1 \rightarrow Z_{2/1}, \hat{z}_{2/1} = F_{io}(z_1)$. Similarly, backward mapping is expressed as $F_{oa} : Z_2 \rightarrow Z_{1/2}, \hat{z}_{1/2} = F_{oa}(\hat{z}_2)$.
Hence, direct IHT is represented by the map
\[ G_{io} : W_1 \rightarrow W_{2/1}, \quad w_{2/1} = G_{io}(w_1), \quad w_1 \in W_1, \quad w_{2/1} \in W_{2/1}, \]
where \( w_1(x,y), (x,y) \in \Psi_1 = \bigcup_{p=1}^{P} \Omega_1^{(p)} \) is a random field specified on the range of all the inputs. If \( (x,y) \in \Omega_1^{(p)} \), then \( w_1^{(p)}(x,y) = S_{c}^{-1}(z_{1/2}^{(p)}) \) is formed by reverse unwrapping of vector \( z_{2/1}(x,y) = F_{io}(z_{1}^{(p)}) \).

Backward IHT is represented by the map
\[ G_{oi} : W_2 \rightarrow W_{1/2}, \quad w_{1/2} = G_{oi}(w_2), \quad w_{1/2} \in W_{1/2}, \quad w_2 \in W_2, \]
where \( w_{1/2}(x,y), (x,y) \in \Psi_2 = \bigcup_{p=1}^{P} \Omega_2^{(p)} \) is a random field specified on the range of all the inputs. If \( (x,y) \in \Omega_2^{(p)} \), then \( w_{1/2}^{(p)}(x,y) = S_{c}^{-1}(z_{2}^{(p)}) \) is formed by reverse unwrapping of vector \( z_{2/1}^{(p)} = F_{oi}(z_{1}^{(p)}) \).

Therefore, for a specific \( w_2 \in W_2, \quad w_{2/1} \in W_{2/1} \), a random field of the stochastic component can be obtained as
\[ v_2 = v_2 - G_{io}(w_1) = w_2 - w_{2/1}, \quad v_2 \in V_2, \]
where \( v_2(x,y), (x,y) \in \Psi_2 \) is a random field specified on the range of all the outputs. If \( (x,y) \in \Omega_2^{(p)} \), then \( v_2^{(p)}(x,y) = S_{c}^{-1}(v_2^{(p)}) \) is formed by reverse unwrapping of vector \( v_2 = z_{2}^{(p)} - z_{2/1}^{(p)} = z_{2}^{(p)} - F_{io}(z_{1}^{(p)}) \).

Therefore, for a specific \( w_1 \in W_1, \quad w_{1/2} \in W_{1/2} \), a random field of the stochastic component can be obtained as
\[ v_1 = v_1 - G_{oi}(w_2) = w_1 - w_{1/2}, \quad v_1 \in V_1 \]
where \( v_1(x,y), (x,y) \in \Psi_1 \) is a random field specified on the range of all the outputs. If \( (x,y) \in \Omega_1^{(p)} \), then \( v_1^{(p)}(x,y) = S_{c}^{-1}(v_1^{(p)}) \) is formed by reverse unwrapping of vector \( v_1 = z_{1}^{(p)} - z_{1/2}^{(p)} = z_{1}^{(p)} - F_{oi}(z_{2}^{(p)}) \).

In the end, the general integral map of the formation of the deterministic and the stochastic component can be presented as follows
\[ G_{e} : W \rightarrow \hat{W}, \quad \hat{w} = G_{e}(w), \quad w = \{w_1, w_2\} \in W = \{W_1, W_2\}, \quad \hat{w} = \{w_{1/2}, w_{2/1}\} \in \hat{W} = \{W_{1/2}, W_{2/1}\}, \quad G_{m} : W \rightarrow V, \quad v = G_{m}(w), \quad v = \{v_1, v_2\} \in V = \{V_1, V_2\}. \]

A general scheme of such integral transformations is presented in Fig. 2. In order to control the deterministic and the stochastic component, we can alter the configuration of the fragments’ inputs and outputs, as well as set the residual error for training the ANN-based converters.

### 2.3. Optimisation problems and solution algorithms
The results obtained in [1, 2] demonstrate that in order to evaluate the interpolation and masking effects of HT, we can use equations for sample covariance matrices of the deterministic and the stochastic component:
\[ C_{E} = \hat{R}_{z21} \hat{R}_{z11}^{-1} \hat{R}_{z21}^{T}, \quad C_{V} = \hat{R}_{z22} - \hat{R}_{z21} \hat{R}_{z11}^{-1} \hat{R}_{z12}, \quad \hat{R}_{z22} = C_{E} + C_{V}, \]
Figure 2. A general scheme of IHT and image decomposition into a deterministic and a stochastic component.

\[
\tilde{R}_{z1} = \frac{1}{P} \sum_{p=1}^{P} z_2^{(p)} z_1^{(p)}, \tilde{R}_{z11} = \frac{1}{P} \sum_{p=1}^{P} z_1^{(p)} z_1^{(p)}, \tilde{R}_{z22} = \frac{1}{P} \sum_{p=1}^{P} z_2^{(p)} z_2^{(p)}.
\]

Analysing (5), we came to the following conclusions. First, for covariance matrices \(\tilde{R}_{z22}, C_E, C_V\) the following is true \(\text{tr} \tilde{R}_{z22} = \text{tr} C_E + \text{tr} C_V\). Then, in order to evaluate the ratio of the deterministic and the stochastic component, we can use \(\rho_{VE} = \text{tr} C_V/\text{tr} C_E\). The invariant to the fragment’s partitioning into the input and the output is the trace of the sample covariance matrix:

\[
\tilde{R}_z = \frac{1}{P} \sum_{p=1}^{P} z^{(p)} z^{(p),T} = \begin{pmatrix} \tilde{R}_{z11} & \tilde{R}_{z12} \\ \tilde{R}_{z21} & \tilde{R}_{z22} \end{pmatrix}, \text{tr} \tilde{R}_z = \text{tr} \tilde{R}_{z11} + \text{tr} \tilde{R}_{z22}.
\]

Elements of matrices \(\tilde{R}_{z11}, \tilde{R}_{z22}, \tilde{R}_{z12}\) depend significantly on the topology of the fragment partitioning into the input and the output.

To find a way to “control” the ratio of the “energy” characteristics of the deterministic and the stochastic component, we need to determine the configuration of the input and the output that would maximise (minimise) the trace of matrix \(C_V\), i.e. the stochastic component, in proportion to minimisation (maximisation) of the trace of matrix \(C_E\), i.e. the deterministic component.

We will now consider a solution to this problem in a situation, when the size of the blocks and the values of \(N_1, N_2, N_1 + N_2 = N\) are fixed. First, let us introduce a binary vector \(x_o = (x_{o1}, \ldots, x_{oN})^T, x_{os} \in \{0; 1\}, s = 1, N\), whose unit values correspond to the locations of the elements (pixels) of the output. Then, vector \(x_i = (x_{i1}, \ldots, x_{iN})^T, x_{is} \in \{0; 1\}, x_{is} = x_{os}, s = 1, N\), whose values are inverted with regard to the corresponding values of \(x_o\), determines the location of the input elements. We also introduce vectors

\[
c_o = (c_{o1}, \ldots, c_{oN_2})^T, c_{os} \in \{1, 2, \ldots, N\}, s = 1, N_2, c_i = (c_{i1}, \ldots, c_{iN_1})^T, c_{is} \in \{1, 2, \ldots, N\}, s = 1, N_1,
\]

whose components determine the indices of the output and input elements. Since both the introduced descriptions are equivalent, we can designate \(c_o = c_o(x), c_i = c_i(x), x\) is any one of the binary vectors \(x_o, x_i\), which will further be used as the solution. Without loss of generality, we will further use vector \(x_o\) as \(x\) and thus describe backward transformation as

\[
x = f(c_o), x = (x_{i1}, \ldots, x_{iN})^T, x_{is} \in \{0; 1\}, x_{is} = x_{os}, s = 1, N
\]

Let \(R\) be a random square symmetric matrix of a specific size \(N_x, N_x \geq N_1, N_x \geq N_2\). Then, designations \(R(c_i(x), c_i(x)), R(c_o(x), c_o(x))\), represent square matrices of size \(N_1 \times N_1, N_2 \times N_2\) that are submatrices of \(R\) and contain rows and columns \(R\) with numbers \(c_i = c_i(x)\) or \(c_o = c_o(x)\).
Accordingly, designations $R(c_i(x), c_0(x))$, $R(c_i(x), c_0(x))$ represent rectangular matrices of size $N_1 \times N_2$, $N_2 \times N_1$ containing rows and columns $R$ with numbers $c_i = c_i(x)$ and $c_0 = c_0(x)$. Using the designations we can describe the covariance matrices as follows

$$
\tilde{R}_{z11}(x) = \tilde{R}_z(c_i(x), c_i(x)), \tilde{R}_{z22}(x) = \tilde{R}_z(c_0(x), c_0(x)), \tilde{R}_{z21}(x) = \tilde{R}_z(c_0(x), c_i(x)).
$$

(7)

Taking this into account, we can formulate the following optimisation problem regarding the control over the ratio of the deterministic and the stochastic component. We need to find a solution $x$ (to make it clear, let $x = x_o$) that would maximise (minimise) the trace of matrix $C_V$, i.e. the stochastic component, and minimise (maximise) the trace of matrix $C_E$, i.e. the deterministic component, and at the same time set the optimal input and output configuration.

$$\hat{x}^{(1)} = \arg \max (\min) \left\{ \operatorname{tr} \left[ \tilde{R}_{z22}(x) - \tilde{R}_{z21}(x) \tilde{R}_{z11}^{-1}(x) \tilde{R}_{z12}(x) \right] \right\}, \sum_{s=1}^{N} x_s = N_2, \ x_s \in \{0; 1\}.
$$

(8)

Optimisation problems similar to (8) are problems of nonlinear integer programming and can have various solutions. These problems are solved with fixed values of $N, N_1, N_2$. If the values of these variables vary, the problems are solved by means of a brute-force search for optimal solutions for each value. Below we analyse alternative optimisation methods and algorithms.

**Occurrence algorithm (OA)** The simplest method is an exhaustive search for possible solutions for each parameter. This algorithm cannot be used, if the fragment is too large, since the number of the analysed combinations for fixed $N, N_1, N_2$ is $N_B = C_N^{N_1} = C_N^{N_2} = N!/(N_1!N_2!)$.

The main disadvantage of this method is the need to subsequently create new combinations enhancing the obtained solution, since it is impossible to enumerate all the possible combinations at once. Thus, even for a $6 \times 6$ fragment the number of combinations is of the $10^{10}$ order. Another disadvantage is that the calculation time increases exponentially for larger fragments. Nevertheless, we used this algorithm for small fragments in order to compare it with other algorithms and determine common tendencies. We suggest the following implementation of the algorithm:

1. $l \leftarrow N_2$.
2. Set the initial combination describing the input and output configuration as an array using the scheme $c_o = (1, 2, \ldots, N_2)^T, \ x = f(c_o), \ c_i = c_i(x)$; determine matrices $\tilde{R}_{z11} = \tilde{R}_z(c_i, c_i)$, $\tilde{R}_{z22} = \tilde{R}_z(c_0, c_0)$, $\tilde{R}_{z21} = \tilde{R}_z(c_0, c_i)$; calculate the value of the parameter $K(x) = \operatorname{tr}C_V$ and set the initial maximum (minimum) $m_a$ ($m_i$) of the objective function $K(x)$.
3. Check if $A_o(N_2) = N$, then $l \leftarrow l - 1$, else $l \leftarrow N_2$.
4. If $l \geq 1$, set a new combination using the scheme $c_o'(i) \leftarrow c_o(l)hj + i - l + 1, \ i = N_2 : -1 : l$, $x' = f(c_o'), \ m'_l < m_l, \ c_i' = c_i(x')$.
5. Determine matrices $\tilde{R}_{z11}' = \tilde{R}_z(c_i', c_i'), \tilde{R}_{z22}' = \tilde{R}_z(c_0', c_0')$, $\tilde{R}_{z21}' = \tilde{R}_z(c_0', c_i')$ and calculate the value of the used parameter $m'_a = K(x')$ ($m'_i = K(x')$).
6. If $m'_a > m_a$ ($m'_i < m_i$), then hold $c_i \leftarrow c_i'$ and $c_0 \leftarrow c_o'$, $m_a \leftarrow m'_a$ ($m_i \leftarrow m'_i$), else ignore the changes.
7. Repeat 3-6 until $l \geq 1$.
8. Hold the values of $c_i$ and $c_0$, $m_a$ ($m_i$) and set $\hat{x}_* \leftarrow f(c_o)$. Hold the required values $\max K(\hat{x}_*)$ ($\min K(\hat{x}_*)$).

The suggested algorithm creates subsequent combinations, the extremum being more precise each time. However, it cannot be used for large fragments, since it is rather time-consuming. There are two other methods that can be used in optimisation problems for larger fragments. These methods employ a genetic algorithm and a simulated annealing based heuristic algorithm.
Genetic algorithm (GA). The first approach is based on genetic optimisation algorithms. In our study, we used a standard scheme, for which the following data and hyperparameters need to be determined [4].

1. Integer vector \( x = (x_1, ..., x_n)^T \), determining the solution. In our study, it was the vector describing the indexing of the fragment’s output \( x = x_o \).

2. The objective function \( K(x) \) (fitness function) that needs to be minimised. In our study, we used different functions \( K(x) = \pm \text{tr} C_V(x) \) for different problem statements. The sign of the function was chosen depending on the type of the problem (minimisation or maximisation).

3. The boundaries of the search area, which are set as inequations \( L_b \leq x \leq U_b \), where \( L_b = (0,0,...,0) \), \( U_b = (1,1,...,1) \).

4. The equality and inequality constraints in the form of matrix equations of the \( Ax \leq b \) type, where for the studied problem \( A = (1,1,...,1) \), \( b = N_2 \).

5. The size of the population \( P_{size} \) and the maximum number of iterations \( N_{gen} \) before stopping.

6. Hyperparameters describing other standard features of the algorithm (mutation and crossover parameters, the precision parameter determining the termination conditions, when there is little difference in the fitness function between neighbouring iterations, etc).

Simulation annealing based heuristic algorithm (SA) The suggested algorithm performs guided search for the solution by subsequently shifting the elements between the input and the output. The algorithm was used, because it requires less time than the occurrence and genetic algorithms. It also allowed us to fine-tune the optimised parameter, i.e. control over the stochastic component. The main hyperparameter of the algorithm is the maximum number of iterations \( N_{sub} \). A general scheme of the algorithm is the following.

1. \( l \leftarrow 0 \).

2. Generate the initial random configuration of the input and the output as a solution to \( x = (x_1, ..., x_n)^T \), \( c_i = c_i(x) \) and \( c_o = c_o(x) \); determine matrices \( \tilde{R}_{111} = \tilde{R}_z(c_i, c_i) \), \( \tilde{R}_{22} = \tilde{R}_z(c_o, c_o) \), \( \tilde{R}_{21} = \tilde{R}_z(c_o, c_i) \), and calculate the matrix \( \tilde{C}_V(x) = \tilde{R}_{212}(x) - \tilde{R}_{21}(x) \tilde{R}_{z11}^{-1}(x) \tilde{R}_{212}(x) \). Set the initial maximum (minimum) \( m_a = \text{tr}(\tilde{C}_V(x)) \) (\( m_i = \text{tr}(\tilde{C}_V(x)) \)).

3. \( l \leftarrow l + 1 \). Set an array of indices of accidental requests for the input elements \( ind = (1,1,...,1) \).

4. Select any of the input elements, whose \( ind(j) = 1 \) (not selected previously), and remember its number \( j_x \).

5. Select an output element that can have the minimum (maximum) impact on the trace of the matrix \( C_V(x) \): \( i_x = \arg \min[ \text{diag}(C_V(x)) ] \) (\( i_x = \arg \max[ \text{diag}(C_V(x))] \)), \( \text{diag}(C_V) = (c_{o11}, c_{o22}, ..., c_{oN_2N_2}) \) and remember its number.

6. Shift the elements with numbers \( j_x, i_x \) between the input and the output by changing the corresponding elements of the arrays designated as \( c'_i \) and \( c'_o \). Determine matrices \( \tilde{R}'_{111} = \tilde{R}_z(c'_i, c'_i) \), \( \tilde{R}'_{22} = \tilde{R}_z(c'_o, c'_o) \), \( \tilde{R}'_{21} = \tilde{R}_z(c'_o, c'_i) \), and calculate matrices \( C'_V = \tilde{R}'_{212} - \tilde{R}'_{21} \tilde{R}'_{z12}^{-1} \tilde{R}'_{212} \) and \( m'_a = \text{tr} \tilde{C}'_V(b'_i = \text{tr} \tilde{C}'_V) \).

7. If \( m'_a > m_a \) (\( m'_i < m_i \)), then hold \( c_i \leftarrow c'_i \) and \( c_o \leftarrow c'_o \), \( m_a \leftarrow m'_a \) (\( m_i \leftarrow m'_i \)) and set \( \text{ind}(j_x) = 0 \), else ignore the changes and return to the previous indexation.

8. Repeat 4-7 \( N_1 \) times (until all the input elements have been shifted).

9. Repeat 3-8 until the iteration is registered at which no shift of the input or output elements takes place, or until the maximum number \( N_{sub} \) of iterations is reached.

10. Hold the values of \( c_i \) and \( c_o \), \( m_a(m_i) \) and set \( \tilde{x}(1) \leftarrow f(c_o) \). Hold the required values max \( K(\tilde{x}(1)) \) (min \( K(\tilde{x}(1)) \)).

When applying this algorithm, we observed a monotonic change in the optimised parameter, which ensures smooth regulation of the masking component. To minimise the possibility of being
trapped in shallow local minima, it is also recommended to run the algorithm several times with different initial configurations. It is highly likely that the heuristic algorithm is significantly faster than the occurrence and the genetic algorithm.

3. Results and discussion
In our study, we used the following objects: monochromatic images generated as realisations of Gaussian random fields with various spatial correlation functions; real monochromatic images and real colour images. Fig. 3 shows typical configurations of the input and the output, obtained for GA and SA, when \( n_w \times m_w = 10 \times 10 \) (the occurrence algorithm cannot be used in this case).

![Typical configurations of the fragment’s input and output](a)

**Figure 3.** Typical configurations of the fragment’s input and output and dependencies for the optimised values of the relation \( \rho_{VE} \) and the time required for the optimisation.

We also determined the dependencies of the maxima and minima of the relation \( \rho_{VE} \), obtained using the three algorithms and characterising the ratio of the deterministic (interpolation) an the stochastic (masking) component, on the size of a square fragment (Fig.3b). We also determined the dependencies for the calculation time (Fig.3c), which allowed us to compare the speed of the algorithms. The dependencies were obtained for a monochromatic image generated as a realisation of a random field. For other type of images the dependencies were similar. When
obtaining the dependencies the input and the output size was set with regard to the total number of points as $N_2 = \text{floor}(N/2)$, $N_1 = N - N_2$.

Studying both machine-generated and real images, we found that it is possible to change the ratio of the deterministic and the stochastic component significantly by altering the configuration of the fragment’s input and output (up to 10 times). The study also demonstrated that the suggested heuristic algorithm yields similar results to those of the occurrence and genetic algorithms. A slight overestimate of the maximum value of the parameter was observed, but it did not exceed 10%. The advantage of the suggested heuristic algorithm is that it worked much faster than the genetic algorithm in all the experiments, where the optimisation was performed with regard to the traces of the matrices. It also demonstrated similar performance level, when the determinant optimisation method was used.

4. Conclusions
In this article we considered the use of heteroassociative transformations performing mutual mapping of two neighbouring or nested local image regions. The article describes an integral form of such transformations, which involves decomposing the image into two independent components - a deterministic and a stochastic component. These components have interpolation and masking effects respectively and can be used in various image processing problems. The described statements of the optimisation problems and their solutions ensure the “control” over the ratio of the deterministic and the stochastic component depending on the configuration of the input and the output of the transformations.

References
[1] Sirota A A and Dryuchenko M A 2015 Computer Optics 5 751–761
[2] Sirota A A, Dryuchenko M A and Yu M E 2015 Radioelectronics and Communications Systems 1 3–16
[3] Sirota A A, Dryuchenko M A and Yu M E 2018 Computer Optics 3 483–494
[4] Yemelianov V V, Kureichuk V V and Kureichuk V M 2003 Theory and practice of evolutionary computation
(Moscow: Fizmatlit)