Image Transformation and Compression using Realization Theory and Balanced Model Reduction.

Bart Vanluyten[1], Katrien De Cock[2], Bart De Moor[3]

K.U.Leuven

Dept. Of Electrical Engineering (ESAT)

Research group SCD (SISTA)

Kasteelpark Arenberg 10

3001 Leuven-Heverlee

Belgium

Tel. 32/16/32 86 66

Fax. 32/16/32 19 70

WWW. http://www.esat.kuleuven.ac.be/sista

E-mail. bart.vanluyten@esat.kuleuven.ac.be

Abstract

Realization theory and balanced model reduction, two well-known methods from system theory, can be used for a wide range of applications that at first sight seem unrelated to system theory. We will show that two-dimensional image transformation and compression can be performed using these techniques.

[1]Bart Vanluyten is a research a ssistant with the Fund for Scientific Research-Flanders (FWO-Vlaanderen).

[2]Dr. Katrien De Cock is a postdoctoral researcher at the Katholieke Universiteit Leuven, Belgium.

[3]Prof. Dr. Bart De Moor is a full professor at the Katholieke Universiteit Leuven, Belgium.

Image Transformation and Compression using Realization Theory and Balanced Model Reduction.

Bart Vanluyten[1], Katrien De Cock[2], Bart De Moor[3]

Abstract

Realization theory and balanced model reduction, two well-known methods from system theory, can be used for a wide range of applications that at first sight seem unrelated to system theory. We will show that two-dimensional image transformation and compression can be performed using these techniques.

I. Introduction

In recent years, the available technology has become insufficient to fulfill the demand for data transmission. One-, two- and more-dimensional *signal compression* is nowadays a very important topic in the domain of digital signal processing. These techniques make an appropriate trade-off between the needed transmission bandwidth and local computational complexity. A typical compression algorithm consists of three steps: linear transformation, quantization and entropy encoding. For the first step a lot of transformations have been examined, for example: the Discrete Fourier Transform (see for instance [1]), the Discrete Cosine Transform [2], the Discrete Wavelet Transform (see for instance [3]). The quantizer in the second step reduces the amount of information to be stored. This can be done by reducing the precision of the transformed

[1]Bart Vanluyten is a research a ssistant with the Fund for Scientific Research-Flanders (FWO-Vlaanderen).

[2]Dr. Katrien De Cock is a postdoctoral researcher at the Katholieke Universiteit Leuven, Belgium.

[3]Prof. Dr. Bart De Moor is a full professor at the Katholieke Universiteit Leuven, Belgium.

coefficients or by making some coefficients equal to zero. This step is the irreversible part of the compression algorithm. Finally, the entropy encoder losslessly encodes the quantized coefficients in order to obtain a high overall compression.

A completely different domain, which at first sight does not have much to do with signal compression, is that of system theory where one studies *mathematical models* of *dynamical systems*. In *system identification*, a subdomain of system theory, the objective is to find a model starting from input-output measurements of a dynamical system. Very often, the order of the model is chosen quite high in order to allow the model to capture all the dynamics of the system. But high-order models are difficult to handle. For that reason, *model reduction techniques* have been developed.

At first sight the two research domains described above do not have much to do with each other. However, this paper describes how one can use realization theory [4], [5], [6] (see appendix A) and balanced model reduction [7], [8] (see appendix B), a system identification and model reduction technique respectively, to transform and compress 1D and 2D data. The claim of this paper is not that the described methods are better than existing methods of signal processing. We merely want to show that methods from system identification can be used for a wide range of applications, including signal transformation and compression. In recent years, the combination of system theory and image processing has also been investigated by some authors [9], [10].

The remainder of the paper is organized as follows: Section 2 introduces some notation. Section 3 gives a general overview of the transformation and compression methods proposed in this paper. Section 4 and Section 5 give the detailed methods for transformation and compression respectively. Section 6 shows some experimental results and finally Section 7 concludes the paper.

II. Notation

The paper basically concentrates on 2D signals (images) in greyscale. An image is represented by a two-dimensional matrix, the rows and columns of which correspond to the horizontal and

vertical dimensions of the image. Each individual element of the matrix contains a number that is a measure for the intensity of the corresponding *pixel* in the image. The value 0 corresponds to a pure black pixel, the value 1 to a pure white pixel.

We refer to matrices using capital letters (e.g. A). The submatrix bounded by the i-th and j-th row and k-th and l-th column, is denoted by $A(i:j, k:l)$. If a colon (:) is used on its own (e.g. $A(:, k:l)$), all available rows and/or columns are included in the submatrix. The same principles apply to vectors. Further extensions are trivial.

The behavior of a Discrete Time, Linear Time-Invariant (LTI) system can be described by a *state space model* of the form

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k, \\ y_k &= Cx_k + Du_k, \end{aligned}$$

with $u_k \in \mathbb{R}^{n_u}$ the input; $y_k \in \mathbb{R}^{n_y}$ the output; $x_k \in \mathbb{R}^{n_x}$ the state with n_x the system order and A, B, C and D real matrices of appropriate dimensions. The state space model is represented by the four-tuple of matrices (A, B, C, D).

The input-output behavior of an LTI system is invariant under a state transformation $x_k \rightarrow Tx_k$, where T is an arbitrary non-singular matrix, provided that the system matrices are adapted accordingly $(A, B, C, D) \rightarrow (TAT^{-1}, TB, CT^{-1}, D)$. A realization with the matrix A diagonal is called the *modal form* of the state space description of the system.

A system in state space form is called *internally stable* if for an input zero and arbitrary initial state x_0, the state x_k goes to zero as $k \rightarrow \infty$. It can be shown that a system in state space form is internally stable if and only if all eigenvalues of the matrix A are inside the unit circle in the complex plane.

The *impulse response sequence* $\{g_k\}_{k=0}^{\infty}$ of an LTI system is given by

$$\begin{aligned} g_0 &= D, \\ g_k &= CA^{k-1}B \quad k \geq 1. \end{aligned} \tag{1}$$

The *extended ($i > n_x$) observability matrix* $\mathcal{O}_i$ and the *extended ($j > n_x$) controllability matrix*

$\mathcal{C}_j$ of a realization (A, B, C, D) are defined respectively as

$$\mathcal{O}_i := \left[\begin{array}{cccc} C^T & (CA)^T & \dots & (CA^{i-1})^T \end{array} \right]^T, \tag{2}$$

$$\mathcal{C}_j := \left[\begin{array}{cccc} B & AB & \dots & A^{j-1}B \end{array} \right]. \tag{3}$$

III. General overview of transformation and compression methods

This section gives a general overview of the proposed transformation (Subsection III-A) and compression algorithms (Subsection III-B). Transformation methods make use of realization theory, compression algorithms of balanced model reduction.

A. Transformation

The idea of the transformation method is roughly to write a discrete image $H(l = 1 : N_y, k = 1 : N_x)$ as a sum of damped cosines. More exactly, we look for a complex matrix $P \in \mathbb{R}^{M_y \times M_x}$ and two ordered sets of complex numbers $\{a_y^{(q)}, q = 1 \dots M_y\}$ and $\{a_x^{(p)}, p = 1 \dots M_x\}$ with M_y and M_x as small as possible, such that

$$H(l,k) = \sum_{q=1}^{M_y} \sum_{p=1}^{M_x} P(q,p) \left(a_y^{(q)}\right)^{l-1} \left(a_x^{(p)}\right)^{k-1}, \tag{4}$$

for $l = 1 \dots N_y$ and $k = 1 \dots N_x$. For obvious reasons, we impose that the elements of the ordered sets $\{a_y^{(q)}, q = 1 \dots M_y\}$ and $\{a_x^{(p)}, p = 1 \dots M_x\}$ are all different from 0 and that the complex conjugate of each element of the set is also in the set. The elements are ordered by increasing argument[4].

Equation (4) can be rewritten as

$$H(l,k) = \widetilde{C}_y \left(\widetilde{A}_y\right)^{l-1} P \left(\widetilde{A}_x\right)^{k-1} \widetilde{B}_x \tag{5}$$

[4]By convention the argument of a complex number is contained in the interval $[0, 2\pi)$.

with $\widetilde{C}_y = \vec{1}^T$, $\widetilde{B}_x = \vec{1}$ and[5,6]

$$\widetilde{A}_y = \mathrm{diag}\left(\left[\begin{array}{cccc} a_y^{(1)} & a_y^{(2)} & \ldots & a_y^{(M_y)} \end{array}\right]\right), \tag{6}$$

$$\widetilde{A}_x = \mathrm{diag}\left(\left[\begin{array}{cccc} a_x^{(1)} & a_x^{(2)} & \ldots & a_x^{(M_x)} \end{array}\right]\right). \tag{7}$$

The complete image can now be written as

$$H = \left[\begin{array}{c} \widetilde{C}_y \\ \widetilde{C}_y\widetilde{A}_y \\ \widetilde{C}_y\widetilde{A}_y^2 \\ \vdots \\ \widetilde{C}_y\widetilde{A}_y^{N_y-1} \end{array}\right] P \left[\begin{array}{cccc} \widetilde{B}_x & \widetilde{A}_x\widetilde{B}_x & \widetilde{A}_x^2\widetilde{B}_x & \ldots & \widetilde{A}_x^{N_x-1}\widetilde{B}_x \end{array}\right].$$

In appendix C, it is shown that the diagonal elements of $\widetilde{A}_y$ and $\widetilde{A}_x$ are related to the *damping numbers* and *pulsations* of the basis of damped cosines. The elements of the matrix P are linked to the *amplitudes* and *phases* of the damped cosines.

The problem of finding $\widetilde{A}_y$, $\widetilde{A}_x$ and P from an image, can be split into the following steps:

1) Perform a reduced Singular Value Decomposition (SVD)[7] on the image $H = USV^T$ and take

$$\mathcal{O} = US^{\frac{1}{2}},$$

$$\mathcal{C} = S^{\frac{1}{2}}V^T.$$

[5] $\vec{1} = \left[\begin{array}{cccc} 1 & \ldots & 1 \end{array}\right]^T$. The size of $\vec{1}$ is clear from the context.

[6] The notation $\mathrm{diag}(\cdot)$ should be interpreted in MATLAB convention. If the entity between brackets is a vector, then $\mathrm{diag}(\cdot)$ delivers a square diagonal matrix with the components of the vector on its main diagonal.

[7] If the SVD of a matrix H is given by

$$H = \left[\begin{array}{cc} U_1 & U_2 \end{array}\right] \left[\begin{array}{cc} S_1 & 0 \\ 0 & 0 \end{array}\right] \left[\begin{array}{c} V_1^T \\ V_2^T \end{array}\right],$$

then the reduced SVD of the matrix H is defined as $H = U_1S_1V_1^T$.

2) Find a diagonal matrix $\widetilde{A}_y$ and a matrix $\widetilde{B}_y$ such that (for $l = 1 \dots N_y$)

$$\mathcal{O}(l,:) = \widetilde{C}_y \widetilde{A}_y^{l-1} \widetilde{B}_y, \tag{8}$$

with $\widetilde{C}_y = \vec{1}^T$.

3) Find a diagonal matrix $\widetilde{A}_x$ and a matrix $\widetilde{C}_x$ such that (for $k = 1 \dots N_x$)

$$\mathcal{C}(:,k) = \widetilde{C}_x \widetilde{A}_x^{k-1} \widetilde{B}_x, \tag{9}$$

with $\widetilde{B}_x = \vec{1}$.

4) Thus we obtain

$$P = \widetilde{B}_y \widetilde{C}_x.$$

Step 2 analyzes the image in the y-direction, Step 3 in the x-direction. Both steps are very similar, so that from now on we will concentrate on the y-direction only.

Depending on the image, we will need a different algorithm to solve the problem. A schematic overview of the different algorithms is given in Figure 1. The algorithms will be worked out in Section IV.

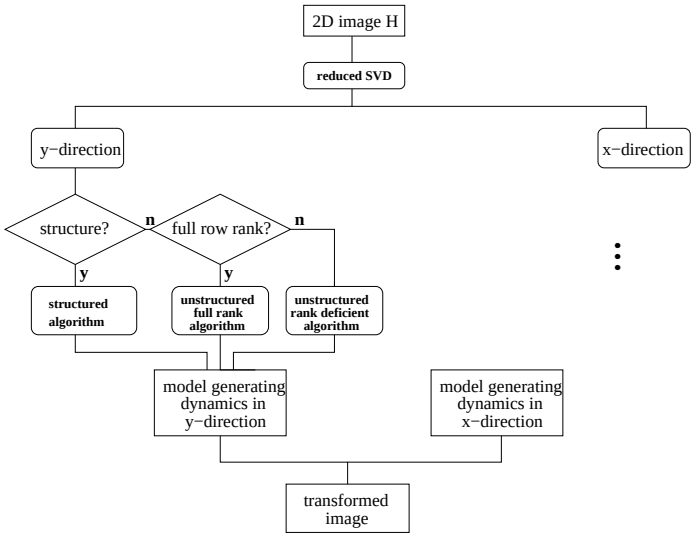


Fig. 1. Schematic overview of the proposed transformation algorithms.

In case the image is *structured in the y-direction*[8] the *structured algorithm* is used. This algorithm finds matrices $\widetilde{A}_y$ and $\widetilde{B}_y$, such that equation (8) holds, with $M_y < N_y$. In case the image is not structured in the y-direction and has full row rank, we have the *unstructured full rank algorithm*. For a rank deficient image without structure in the y-direction we use the *unstructured rank deficient algorithm*. These last two algorithms are only able to find matrices $\widetilde{A}_y$ and $\widetilde{B}_y$, such that equation (8) holds, if M_y is allowed to be equal to N_y. Furthermore in the rank deficient case, the elements of $\widetilde{A}_y$ can be chosen by the user.

From the previous explanation we conclude that Step 2 will always have a solution. Analogously, also Step 3 will always have a solution. This means that the decomposition of equation (5) is always possible. So the matrix P can always be found and is called the *transformed image*. The size of the matrix P depends on the structure (or lack of structure) in y- and/or x-direction[9].

In Section IV it will become clear that each of the different algorithms of Figure 1 is a kind of realization algorithm. Hence the matrices $\widetilde{C}_y$, $\widetilde{A}_y$ and $\widetilde{B}_y$ can be considered as matrices of a dynamical system. We say that the system ($\widetilde{A}_y$, $\widetilde{B}_y$, $\widetilde{C}_y$, 0) *generates the dynamics* of the image *in the y-direction*. Analogously, the system ($\widetilde{A}_x$, $\widetilde{B}_x$, $\widetilde{C}_x$, 0) *generates the dynamics* of the image *in the x-direction*. So the complete image is represented by two dynamical systems.

B. Compression

A schematic overview of the proposed compression methods is given in Figure 2.

In case the image is structured in a certain direction, the amount of memory needed to store the transformed image is smaller than the amount of memory needed to store the original image[10]. Nevertheless, the transformed image (together with the damping numbers and pulsations of the basis functions) contains all the information to find back the original image.

[8]The definition of *structure* in a direction will be given in Subsection IV-A.

[9]A transformed image has the property that the original image can be reconstructed exactly from it.

[10]We will always neglect the memory needs to store the damping numbers and pulsations (i.e. $\widetilde{A}_y$ and $\widetilde{A}_x$) of the basis functions.

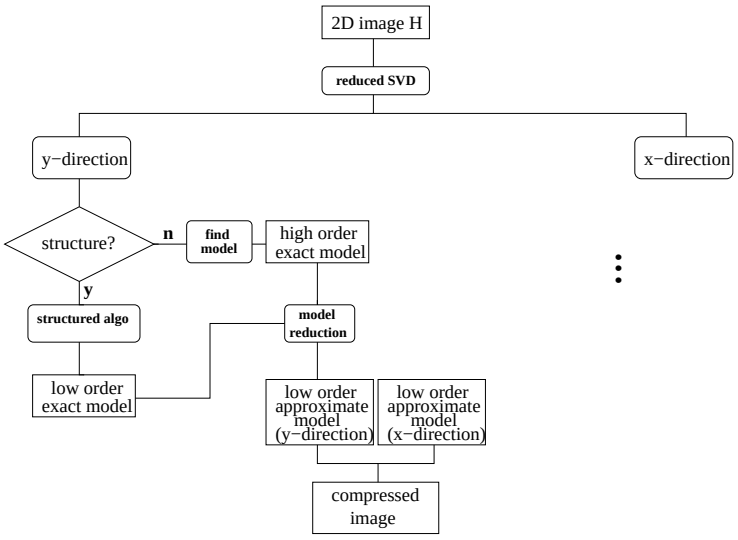


Fig. 2. Schematic overview of the proposed compression algorithm.

In case the image is not structured in a certain direction, we cannot exploit structure to reduce the memory needed to store the image. However, it is possible to compress the transformed images by allowing information loss. We propose to perform model reduction on the dynamical system generating the dynamics of the image in the direction we are considering. The P-matrix corresponding to the dynamical systems in y- and x-direction, of which at least one is of reduced order, is called the *compressed transformed image*. The amount of memory needed to store the compressed transformed image is much smaller than the amount needed to store the original image. But, because of the loss of information, the original image can not be reconstructed from the compressed transformed image anymore. The user needs to make the appropriate trade-off between memory needs and image quality. Of course, also in case the original image is structured, this compression scheme can be used to further compress the transformed image.

In the case where structure is expected to be found in both directions (from physical interpretations for instance) but is not found because of noise, we say the image has *underlying structure*[11]. A method based on the *structured algorithm* is proposed to find back the underlying structure. A

[11]Note that an image with *underlying structure* is in general not structured in y- nor in x-direction.

schematic overview of the method is given in Figure 3. Finding the underlying structured image from an image with underlying structure, corresponds to denoising that image. The amount of memory needed to store the underlying structured image is smaller than the amount of memory needed to store the original image. So compression is obtained by denoising the image.

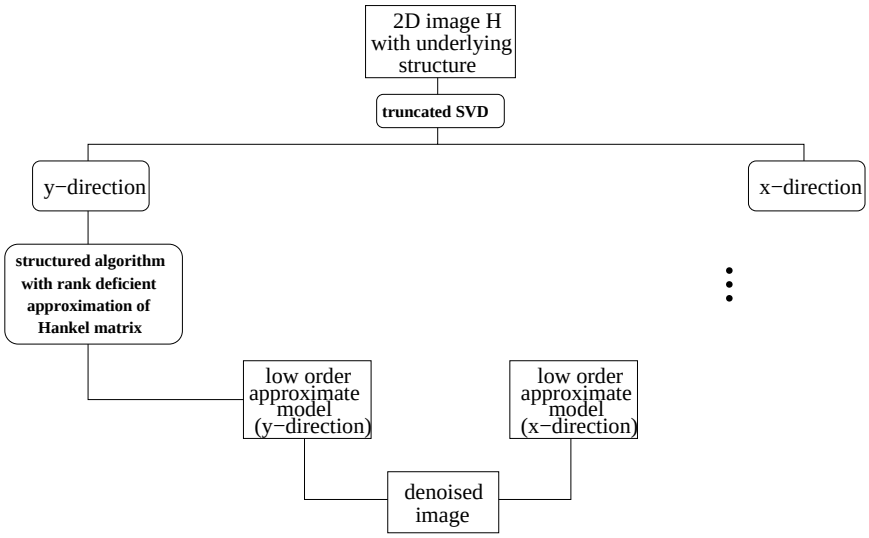


Fig. 3. Schematic overview of the proposed denoising algorithm.

IV. TRANSFORMATION OF IMAGES

In this section the different proposed transformation algorithms are explained in detail. As explained before, we will concentrate on the y-direction. Subsections IV-A and IV-B give the *structured algorithm* and the *unstructured algorithm* respectively.

A. Structured algorithm

To check whether an image is *structured in the y-direction*, we build a block Hankel matrix $H_{\mathcal{O}}$ in the following way

$$H_{\mathcal{O}} = \begin{bmatrix} \mathcal{O}(1,:) & \mathcal{O}(2,:) & \dots & \mathcal{O}(i+1,:) \\ \mathcal{O}(2,:) & \mathcal{O}(3,:) & & \mathcal{O}(i+2,:) \\ \vdots & & \ddots & \vdots \\ \mathcal{O}(N_y-i,:) & \mathcal{O}(N_y-i+1,:) & \dots & \mathcal{O}(N_y,:) \end{bmatrix}, \tag{10}$$

with i chosen such that the size of $H_{\mathcal{O}}$ is maximized:

$$i = \arg\max_i \left(\min\left(N_y - i, (i+1)\mathrm{rank}(H)\right)\right).$$

The original image H is now called *structured in the y-direction* if the corresponding Hankel matrix $H_{\mathcal{O}}$ is rank deficient. One easily sees that an image with full row rank can not be structured in the y-direction, so only images which are rank deficient in the row-direction need to be checked.

If the image H is structured in the y-direction, one can find matrices C_y, A_y and B_y with minimal dimensions such that (for $l = 1 \dots N_y$)

$$\mathcal{O}(l,:) = C_y A_y^{l-1} B_y. \tag{11}$$

Indeed by considering $\mathcal{O}(l,:)$, $l = 1 \dots N_y$ as the impulse response of a linear system, the realization algorithm (see Appendix A) with $H_{\mathcal{O}}$ as Hankel matrix, finds the matrices C_y, A_y and B_y. The size of the square matrix A_y is equal to the rank of $H_{\mathcal{O}}$.

From the matrices C_y, A_y and B_y, one can find matrices $\widetilde{A}_y$ and $\widetilde{B}_y$ with $\widetilde{A}_y$ diagonal such that equation (8) holds, as follows

1) Calculate the eigenvalue decomposition of A_y

$$A_y = V_y \widetilde{A}_y V_y^{-1}, \tag{12}$$

in which the eigenvalues are ordered in the same way as in equation (6).

2) Calculate $\widetilde{B}_y$ as

$$\widetilde{B}_y = \mathrm{diag}(C_y V_y) V_y^{-1} B_y. \tag{13}$$

B. Unstructured algorithm

In this subsection we will find a diagonal matrix $\widetilde{A}_y$ and a matrix $\widetilde{B}_y$ such that equation (8) holds for unstructured images. Because here the transformations of equations (12) and (13) are also possible, we can concentrate on finding matrices C_y, A_y and B_y such that equation (11) holds. This will only be possible if we allow the size of A_y to be equal to N_y. In that case the eigenvalues of A_y are no longer unique and can be chosen by the user. We make a distinction between H full row rank (part IV-B.1) and H not full row rank (part IV-B.2). In the last part of this subsection, the connection with the Fourier transform will be presented.

1) Unstructured full rank algorithm: We start with a proposition needed to understand the algorithm.

Proposition 1: Given a full rank $(N \times N)$ matrix X and a row vector $c = [c_1, c_2, \ldots, c_N]$ then the eigenvalues of the matrix[12] $A := \left(\underline{X_{ex}}\right)^{-1} \overline{X_{ex}}$ with $X_{ex} := \begin{bmatrix} X \\ a \end{bmatrix}$ formed by adding a row vector a to the matrix X, are equal to the roots of the N-th order polynomial $p(\lambda) = \lambda^N - c_N\lambda^{N-1} - c_{N-1}\lambda^{N-2} - \ldots - c_2\lambda - c_1$ if and only if the row vector a is equal to cX.

Proof: We split the proof into two parts

a. $\underline{\text{eigenvalues}(A) = \text{roots}(p(\lambda)) \quad \Rightarrow \quad a = cX}$

From $\underline{X_{ex}}A = \overline{X_{ex}}$ we know that if $C = X_{ex}(1,:)$, it holds for $l = 1 \ldots N+1$ that

$$X_{ex}(l,:) = CA^{l-1}.$$

This means that

$$a = CA^N. \tag{14}$$

Furthermore from Cayley-Hamilton we know that

$$A^N - c_N A^{N-1} - c_{N-1}A^{N-2} - \ldots - c_2 A - c_1 I = 0,$$

[12]If T is a $in_y \times jn_u$ block matrix, with $n_y \times n_u$ matrices as its blocks, then $\overline{T}$, $\underline{T}$ are $(i-1)n_y \times jn_u$ matrices constructed from T by omission of the first, last block row respectively. Similarly, $|T$, $T|$ are the $in_y \times (j-1)n_u$ block matrices constructed from T by omission of its first, last block column respectively. It this case the $n_y = 1$, so $\overline{X_{ex}}$, $\underline{X_{ex}}$ are equal to the matrix X_{ex} without the first, last row respectively.

and consequently

$$c_N CA^{N-1} + c_{N-1} CA^{N-2} + \ldots + c_2 CA + c_1 C = CA^N. \tag{15}$$

From equations (14) and (15) we find that $a = cX$ which completes the first part of the proof.

b. $\underline{a = cX \quad \Rightarrow \quad \text{eigenvalues}(A) = \text{roots}(p(\lambda))}$

Notice that A can be written as[13]

$$A = X^{-1} \begin{bmatrix} \overline{I_N} \\ c \end{bmatrix} X$$

So the eigenvalues of A are equal to the roots of

$$\det \left(\begin{bmatrix} \overline{I_N} \\ c \end{bmatrix} - \lambda I \right) = 0,$$

or

$$\det \begin{bmatrix} -\lambda & 1 & & & \\ & -\lambda & 1 & & \\ & & \ddots & \ddots & \\ & & & -\lambda & 1 \\ c_1 & c_2 & \ldots & c_{N-1} & c_N - \lambda \end{bmatrix} = 0.$$

This can also be written as[14]

$$\sum_{l=1}^{N-1} c_l (-1)^{N+l} \det(M_{N,l}) + (c_N - \lambda)(-1)^{2N} \det(M_{N,N}) = 0,$$

which is equal to

$$\sum_{l=1}^{N-1} c_l (-1)^{N+l} (-\lambda)^{l-1} + (c_N - \lambda)(-1)^{2N} (-\lambda)^{N-1} = 0,$$

[13] I_N is the unit matrix of size N.

[14] The determinant of a $(N \times N)$-matrix X can be decomposed as a weighted sum of N determinants of $((N-1) \times (N-1))$ matrices $M_{k,l}$, which are matrices formed by deleting row k and column l from X, as

$$\det(X) = \sum_{l=1}^{N} X(k,l)(-1)^{k+l} \det M_{k,l}$$

where k ($1 \leq k \leq N$) can be chosen by the user. We choose $k = N$.

and consequently we obtain

$$\lambda^N - c_N\lambda^{N-1} - c_{N-1}\lambda^{N-2} - \ldots - c_2\lambda - c_1 = 0.$$

■

Given a full row rank image $H(1:N_y, 1:N_x)$ and a polynomial $p(\lambda) = \lambda^{N_y} - c_{N_y}\lambda^{N_y-1} - \ldots - c_2\lambda - c_1$ of order N_y, the algorithm below finds matrices C_y and A_y such that equation (11) holds for $l = 1 \ldots N_y$ and such that the eigenvalues of A_y are equal to the roots of $p(\lambda)$.

- Define the matrix $\mathcal{O}_{ex} = \begin{bmatrix} \mathcal{O} \\ c\mathcal{O} \end{bmatrix}$ with $c = [c_1, c_2, \ldots, c_{N_y}]$.
- Take C_y equal to the first row of $\mathcal{O}_{ex}$.
- For A_y we have
$$A_y = \left(\underline{\mathcal{O}_{ex}}\right)^{-1} \overline{\mathcal{O}_{ex}}.$$
- Take B_y equal to $I_{\mathrm{rank}(H)}$.

2) Unstructured rank deficient algorithm: If H has no full row rank (i.e. $\mathcal{O}$ has no full rank), proposition 1 and consequently the *unstructured full rank algorithm* can not be used as such. We present a modified version of the algorithm which finds matrices C_y, A_y and B_y such that equation (11) holds.

- Perform a reduced SVD on the matrix $\begin{bmatrix} \mathcal{O} & I_{N_y} \end{bmatrix} = USV^T$ and take $\mathcal{O}^+ = US^{\frac{1}{2}}$ and $\mathcal{C}^+ = S^{\frac{1}{2}}V^T$.
- Define the matrix $\mathcal{O}^+_{ex} = \begin{bmatrix} \mathcal{O}^+ \\ c\mathcal{O}^+ \end{bmatrix}$ with $c = [c_1, c_2, \ldots, c_{N_y}]$.
- Take C_y equal to the first row of $\mathcal{O}^+_{ex}$.
- For A_y we have
$$A_y = \left(\underline{\mathcal{O}^+_{ex}}\right)^{-1} \overline{\mathcal{O}^+_{ex}}.$$
- Take B_y equal to the first rank(H) columns of $\mathcal{C}^+$.

3) Fourier transform: A two-dimensional image $H(l = 1 : N_y, k = 1 : N_x)$ and its Fourier transform $\mathcal{H}(q = 1 : N_y, p = 1 : N_x)$ are linked to each other by the definition of the inverse

Fourier transform

$$H(l,k) = \frac{1}{N_x}\frac{1}{N_y}\sum_{q=1}^{N_y}\sum_{p=1}^{N_x}\mathcal{H}(q,p)e^{j2\pi(q-1)(l-1)/N_y}e^{j2\pi(p-1)(k-1)/N_x}. \quad (16)$$

When comparing formula (16) and formula (4), one sees that the matrix P found by the *unstructured algorithm* will be equal (up to a scalar constant) to the Fourier transform $\mathcal{H}$ of the image if the diagonal elements of $\widetilde{A}_y$ and $\widetilde{A}_x$ are equal to

$$\begin{aligned} a_y^{(q)} &= e^{j2\pi(q-1)/N_y}, \quad q = 1 \ldots N_y, \\ a_x^{(p)} &= e^{j2\pi(p-1)/N_x}, \quad p = 1 \ldots N_x. \end{aligned}$$

This can be obtained by using the *unstructured algorithm* with $p(\lambda) = \lambda^{N_y} - 1$ for the y-direction and $p(\lambda) = \lambda^{N_x} - 1$ for the x-direction.

V. Compression of images

In this section the two proposed compression schemes will be worked out. The first scheme (Subsection V-A) works for all kind of images, structured and unstructured. However it is most useful for unstructured images, because there much compression is possible. The second scheme (Subsection V-B) is used for finding the underlying structured image of an image which is the noisy version of a structured image, which corresponds to denoising that last image. Because a structured image can be stored with less memory than an unstructured one (i.e. the image with underlying structure), the denoising process involves a compression.

A. Compression of arbitrary images

We know from Section IV that an image can be represented by two dynamical systems (A_y, B_y, C_y, 0) and (A_x, B_x, C_x, 0), generating the dynamics in the y- and x-direction respectively. This gives rise to a transformation of the image to a basis of damped cosines (formula (5)). In case the image is not structured, the size of the transformed image P, is exactly the same as the original image H.

The proposed compression method consists of performing a balanced model reduction (see Appendix B) on the two dynamical systems generating the image. Again, we will only concentrate on the system generating the y-direction.

We know from Subsection IV-B that the N_y eigenvalues of the matrix A_y can be chosen by the user in the unstructured case. The matrix $A_y^{[R_y]}$, the A-matrix of the reduced order model, will only have $R_y < N_y$ eigenvalues, which are in general no subset of the eigenvalues of the A_y-matrix of the full order model. Therefore, we will always start from the following easy realization before performing balanced model reduction to an unstructured image.

$$A_y = \begin{bmatrix} 0 & 1 & & \\ & 0 & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{bmatrix}, \quad B_y = \mathcal{O},$$
$$C_y = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}.$$

Note that these matrices ensure that equation (11) holds for $l = 1 \dots N_y$. The eigenvalues of the A_y-matrix of this realization are all equal to 0.

The compression ratio depends on the order of the reduced systems in y- and x-direction and is equal to $(R_y \times R_x)/(N_y \times N_x)$. We are not able anymore to reconstruct the original image from the compressed transformed image. The user should make the appropriate trade-off between memory needs and quality of the reconstructed image.

B. Denoising of structured images

An image is said to have *underlying structure*, if the image is a sum of a structured image (in y- and x-direction) and a two-dimensional noise image[15]. With some modifications the *structured algorithm* can be used as a denoising algorithm. In the rest of this subsection the proposed modifications will be explained. They are based on the realization algorithm for noisy measurements of the impulse response [6].

[15]The covariance of the noise is expected to be small enough, such that the effect of the structure is still *visible*.

An image with underlying structure, generally has full rank (so no structure neither in y- nor in x-direction). Nevertheless because there is an underlying structure present, the image is expected to have some dominant singular values among other small singular values. So the idea is to compute the reduced SVD of the image $H = USV^T$ with $S = \mathrm{diag}([\sigma_1, \sigma_2, \ldots, \sigma_{\min(N_y,N_x)}])$, and make a rank deficient approximation $H^{[R]} = US^{[R]}V^T$ with $S^{[R]} = \mathrm{diag}([\sigma_1, \ldots, \sigma_R, 0, \ldots, 0])$ by retaining only the R dominant singular values of H.

As before, we then define $\mathcal{O}^{[R]} = U\left(S^{[R]}\right)^{\frac{1}{2}}$ and $\mathcal{C}^{[R]} = \left(S^{[R]}\right)^{\frac{1}{2}}V^T$. The problem can again be solved for the y- and x-direction separately. So we will concentrate on the y-direction in the rest of the explanation.

The Hankel matrix $H_{\mathcal{O}^{[R]}}$ which is built from the matrix $\mathcal{O}^{[R]}$ (formula (10)) will not be rank deficient in general. Nevertheless, the Hankel matrix is expected to have some dominant singular values among other small singular values. Again, we make a rank deficient approximation $H^{[R_y]}_{\mathcal{O}^{[R]}}$ of $H_{\mathcal{O}^{[R]}}$ by only retaining the R_y dominant singular values.

The next step is to apply the *structured algorithm* to $H^{[R_y]}_{\mathcal{O}^{[R]}}$ as if it were a block Hankel matrix. Therefore we first take an SVD of $H^{[R_y]}_{\mathcal{O}^{[R]}} = USV^T$ and define $\mathcal{O}^{[R_y]}_{\mathcal{O}^{[R]}} = US^{\frac{1}{2}}$ and $\mathcal{C}^{[R_y]}_{\mathcal{O}^{[R]}} = S^{\frac{1}{2}}V^T$. Because $H^{[R_y]}_{\mathcal{O}^{[R]}}$ has no Hankel structure (see equation (19) in Appendix A), there will not exist a matrix $A^{[R_y]}_y$ such that $\underline{\mathcal{O}^{[R_y]}_{\mathcal{O}^{[R]}}}A^{[R_y]}_y = \overline{\mathcal{O}^{[R_y]}_{\mathcal{O}^{[R]}}}$. But we know that[16] $A^{[R_y]}_y = \left(\underline{\mathcal{O}^{[R_y]}_{\mathcal{O}^{[R]}}}\right)^{\dagger}\overline{\mathcal{O}^{[R_y]}_{\mathcal{O}^{[R]}}}$ is the solution to[17]

$$A^{[R_y]}_y = \arg\min_{A^{[R_y]}_y} \|\underline{\mathcal{O}^{[R_y]}_{\mathcal{O}^{[R]}}}A^{[R_y]}_y - \overline{\mathcal{O}^{[R_y]}_{\mathcal{O}^{[R]}}}\|_F.$$

So we conclude that in this case we can still use the realization algorithm with the matrix $H^{[R_y]}_{\mathcal{O}^{[R]}}$ as the *Hankel matrix* although this matrix has no Hankel structure.

With the method described above, we find diagonal matrices $\widetilde{A}^{[R_y]}_y$ and $\widetilde{A}^{[R_x]}_x$ and a matrix

[16] $X^{\dagger}$ is the Moore-Penrose pseudo inverse of X [11]:
$X^{\dagger}X = I$ if X has full column rank,
$XX^{\dagger} = I$ if X has full row rank,
$X^{\dagger}X = XX^{\dagger} = I$ if X is regular ($X^{\dagger} = X^{-1}$).

[17] The Frobenius norm of an $(N \times M)$ matrix A is defined as $\|A\|_F = \left(\sum_{n=1}^{N}\sum_{m=1}^{M}(a_{nm})^2\right)^{\frac{1}{2}}$.

$P^{[R_y \times R_x]}$ such that (for $l = 1 \ldots N_y$ and $k = 1 \ldots N_x$)

$$H(l,k) \simeq \widetilde{C}_y^{[R_y]} \left(\widetilde{A}_y^{[R_y]}\right)^{l-1} P^{[R_y \times R_x]} \left(\widetilde{A}_x^{[R_x]}\right)^{k-1} \widetilde{B}_x^{[R_x]},$$

with $\widetilde{C}_y^{[R_y]} = \vec{1}^T$, $\widetilde{B}_x^{[R_x]} = \vec{1}$. The matrix $P^{[R_y \times R_x]}$ has size $(R_y \times R_x)$, which is smaller than the original image size $(N_y \times N_x)$. The compression ratio is equal to $(R_y \times R_x)/(N_y \times N_x)$. We are not able to reconstruct the original image from the compressed transformed image anymore. However, the reconstructed image is a noise-free version of the original image.

VI. Experiments

A. Transformation

Structured algorithm: To test the *structured algorithm*, we generate an artificial image as a sum of damped cosines and try to find back the pulsations, damping numbers, amplitudes and phases that we used for the generation of the image.

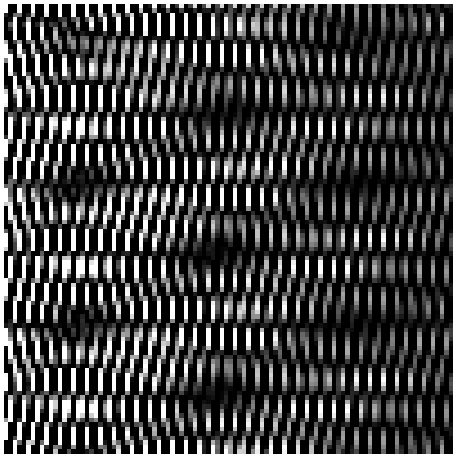


Fig. 4. Image H_1 on which the *structured algorithm* will be tested.

The image $H_1(l = 1:100, k = 1:100)$ is generated with formula (17) (with $k' = k - 1$ and

$l' = l - 1$) and is shown in Figure 4.

$$\begin{aligned} H_1(l,k) &= (0.98)^{k'} \left[\cos\left(0.6l' + 2.1k'\right) + 0.9\cos\left(0.6l' - 2.1k'\right)\right] + \\ &\quad (0.98)^{k'} (0.95)^{l'} \left[\cos\left(0.7l' + 2.1k' + \pi/2\right) + 0.8\cos\left(0.7l' - 2.1k'\right)\right] + \\ &\quad (0.99)^{k'} \left[\cos\left(0.4l' + 2k'\right) + 0.7\cos\left(0.4l' - 2k' - \pi\right)\right]. \end{aligned} \tag{17}$$

The algorithm finds structure in both the x and y-direction. The output of the algorithm is:

$$\begin{aligned} \widetilde{A}_y &= \operatorname{diag}\left(\left[\begin{array}{cccccc} e^{j0.4} & e^{j0.6} & 0.95e^{j0.7} & 0.95e^{-j0.7} & e^{-j0.6} & e^{-j0.4} \end{array}\right]\right), \\ \widetilde{A}_x &= \operatorname{diag}\left(\left[\begin{array}{cccc} 0.99e^{j2} & 0.98e^{j2.1} & 0.98e^{-j2.1} & 0.99e^{-j2} \end{array}\right]\right), \\ P &= \left[\begin{array}{cccc} 0.5 & 0 & 0 & -0.35 \\ 0 & 0.5 & 0.45 & 0 \\ 0 & e^{j1.5708} & 0.4 & 0 \\ 0 & 0.4 & e^{-j1.5708} & 0 \\ 0 & 0.45 & 0.5 & 0 \\ -0.35 & 0 & 0 & 0.5 \end{array}\right]. \end{aligned}$$

This result is exactly what we expected (see appendix C).

Unstructured algorithm: Next the algorithm will be tested on the image $H_2(l = 1:150, k = 1:160)$ shown in Figure 5. The image has full row rank, but is rank deficient in the column direction. So in the y-direction the *unstructured full rank algorithm* and in the x-direction the *unstructured rank deficient algorithm* will be used. Suppose one wants a decomposition into two linear systems (A_y, B_y, C_y, 0) and (A_x, B_x, C_x, 0), with the eigenvalues of A_y and A_x given in the table below.

Eigenvalues of A_y:	the 100th roots of 0.9
	the 50th roots of 0.01
Eigenvalues of A_x:	the 105th roots of 0.95
	the 55th roots of 0.2

The polynomials $p_y(\lambda)$ and $p_x(\lambda)$ needed for the algorithms, can be calculated as

$$\begin{aligned} p_y(\lambda) &= (\lambda^{100} - 0.9)(\lambda^{50} - 0.01), \\ p_x(\lambda) &= (\lambda^{105} - 0.95)(\lambda^{55} - 0.2). \end{aligned}$$

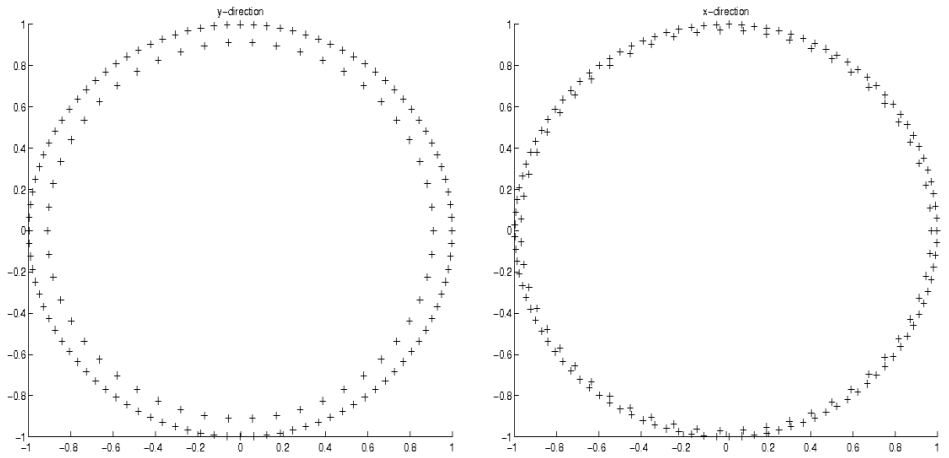
Figure 6 shows a complex plot of the eigenvalues of A_y and A_x, that are the output of the algorithm. The results are exactly as could be expected.



Fig. 5. Image H_2 on which the *unstructured algorithm* will be tested.

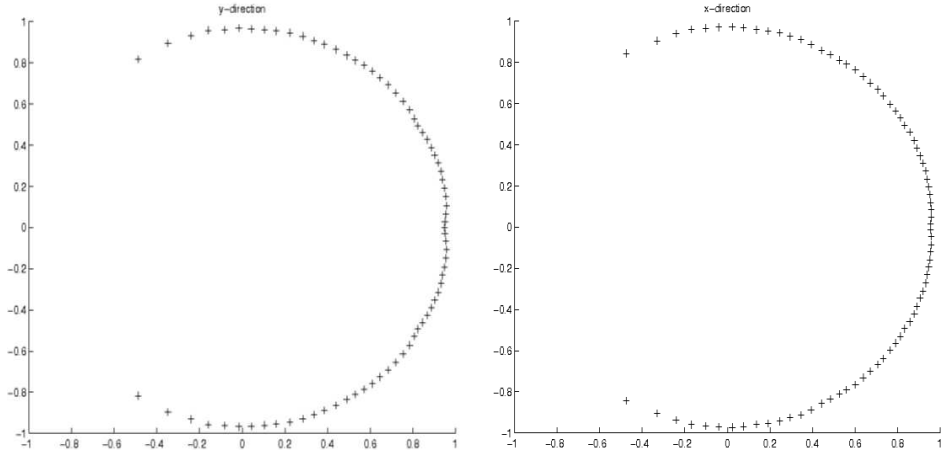
B. Compression

Compression of an arbitrary image: We will again work with image H_2 (Figure 5). First the systems generating the dynamics in the y- and the x-direction are built as explained in Section V-A. By construction all the eigenvalues of the matrices A_y and A_x are equal to 0. Then balanced model reduction is performed on both models, where we retain 75 eigenvalues in the y-direction and 80 eigenvalues in the x-direction. This means that the total amount of memory needed to store the compressed image is one fourth of the amount of memory needed to store the original image. A complex plot with the eigenvalues of $A_y^{[75]}$ and $A_x^{[80]}$ is given in Figure 7. The reconstructed compressed image is shown in Figure 8.



(a) (b)

Fig. 6. Complex plot of the eigenvalues of the matrices A_y (Subfigure (a)) and A_x (Subfigure (b)).



(a) (b)

Fig. 7. Complex plot of the eigenvalues of the matrices $A_y^{[75]}$ (Subfigure (a)) and $A_x^{[80]}$ (Subfigure (b)) .

One sees that the eigenvalues for the reduced order systems all have a modulus of approximately 1. This corresponds to the fact that there is no visible damping in the image. Furthermore there are more eigenvalues at low pulsations than at high pulsations. This corresponds to the fact that the most important information in the image is at low pulsations.



Fig. 8. Reconstruction of image H_2 after compression with balanced model reduction.

Denoising of a structured image: The image we will work with is a checkerboard image corrupted by additive gaussian noise with $\sigma^2 = 0.04$. The image is shown in Figure 9. Figure 10 shows

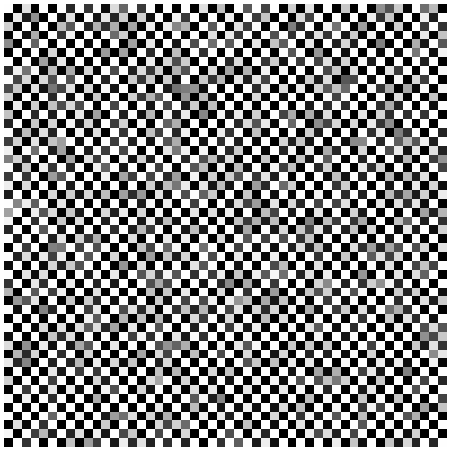


Fig. 9. Image H_3 on which the first proposed compression scheme will be applied (underlying structure present).

the singular values of the image. Because there is underlying structure present, we could have expected to find some dominant and other small singular values. In this case we have two dominant singular values, so we make a rank 2 approximation of the image. Next the *structured algorithm* will be applied on both $y-$ and x-direction. The matrices $H_{\mathcal{O}^{[R]}}$ and $H_{\mathcal{C}^{[R]}}$ are full

rank, but again in both matrices there are two dominant singular values, so that we perform a rank 2 approximation.

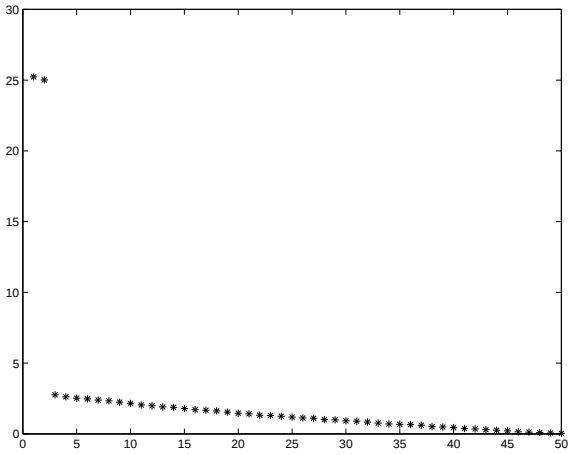


Fig. 10. Plot of the singular values of image H_3.

The calculated matrices $\widetilde{A}_y$, $\widetilde{A}_x$ and P are given by

$$\begin{aligned} \widetilde{A}_y &= \mathrm{diag}\left(\begin{bmatrix} -1.0017 & 1.0000 \end{bmatrix}\right), \\ \widetilde{A}_x &= \mathrm{diag}\left(\begin{bmatrix} -1.0001 & 1.0000 \end{bmatrix}\right), \\ P &= \begin{bmatrix} 0.4858 & 0.0007 \\ 0.0031 & 0.4982 \end{bmatrix}, \end{aligned} \tag{18}$$

while the matrices $\widetilde{A}_y^{\mathrm{checker}}$, $\widetilde{A}_x^{\mathrm{checker}}$, $\widetilde{P}^{\mathrm{checker}}$ of a checkerboard image are given by

$$\begin{aligned} \widetilde{A}_y^{\mathrm{checker}} &= \mathrm{diag}\left(\begin{bmatrix} -1 & 1 \end{bmatrix}\right), \\ \widetilde{A}_x^{\mathrm{checker}} &= \mathrm{diag}\left(\begin{bmatrix} -1 & 1 \end{bmatrix}\right), \\ P^{\mathrm{checker}} &= \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}. \end{aligned}$$

The presented algorithm finds a good approximation of the underlying structured image. The reconstructed image with the matrices of equation (18) is given in Figure 11. One hardly notices any difference with a checkerboard image.

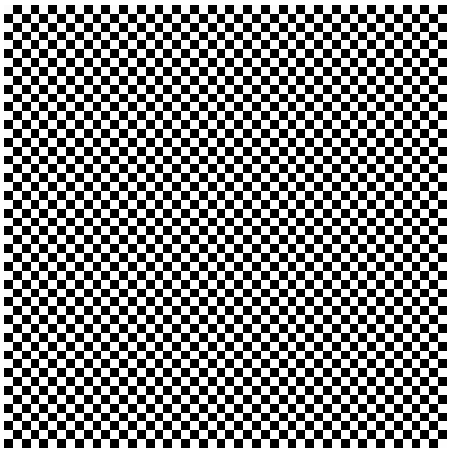


Fig. 11. Reconstructed image using the matrices of equation (18).

VII. Conclusion

In this paper it was shown that realization theory and balanced model reduction, two methods from system theory, can be used to transform and compress 2D images.

We propose a decomposition in a basis of damped cosines. We show that it is necessary to make a distinction between so-called structured and unstructured images. In both cases the decomposition is equivalent to a description of the image by two linear time invariant systems generating the dynamics in the y- and x-direction.

By using balanced model reduction on the two dynamical systems generating the image, compression of the image is obtained. This is most usefull for unstructured images, because there most compression is possible.

In case the image is a noisy version of a structured image, we propose a denoising algorithm based on the transformation algorithm for structured images. By denoising the image, a severe compression is obtained.

The fact that an image can be represented by two dynamical systems has interesting applications, for instance for clustering images. When a distance measure between dynamical systems is used,

one can cluster images based on their behavior in the y- and/or x-direction.

APPENDIX

A. Realization theory

Realization algorithms [4], [5], [6] aim to find a realization (A, B, C, D) of an LTI system starting from observations of the impulse response. Below we describe one possible realization algorithm.

Given: The impulse response sequence $g_k, k = 0, 1, \ldots, K$ of an unknown LTI system with unknown order n_x.

Problem: Find a realization (A, B, C, D) with minimal order of the LTI system which generated the impulse response data.

Method:

- The matrix D can be found easily as $D = g_0$.
- Build the $in_y \times jn_u$ *block Hankel matrix* $H_{i,j}$, with block dimensions i and j and $(i+j-1) \leq K$ which is defined as

$$H_{i,j} = \begin{bmatrix} g_1 & g_2 & g_3 & \cdots & g_j \\ g_2 & g_3 & g_4 & & g_{j+1} \\ g_3 & g_4 & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ g_i & g_{i+1} & \cdots & \cdots & g_{i+j-1} \end{bmatrix}. \tag{19}$$

  An obvious property following from equation (1) is that this block Hankel matrix can be factorized into the product of the extended observability matrix and the extended controllability matrix $H_{i,j} = \mathcal{O}_i \mathcal{C}_j$. If i and j are large enough, which will be assumed in the algorithm, the block Hankel matrix is rank deficient. Furthermore, its rank is equal to n_x, the minimal order of the underlying system.

- Perform a reduced SVD on the Hankel matrix $H_{i,j} = USV^T$ and take

$$\begin{aligned} \mathcal{O}_i &= US^{\frac{1}{2}}, \\ \mathcal{C}_j &= S^{\frac{1}{2}}V^T. \end{aligned}$$

  The rank of the block Hankel matrix, the minimal order of the underlying system, is equal to the number of non-zero singular values.
- C is formed from the first n_y rows of $\mathcal{O}_i$, while B is formed from the first n_u columns of $\mathcal{C}_j$.
- It follows from equation (2) that $\underline{\mathcal{O}_i}A = \overline{\mathcal{O}_i}$. So A can be calculated as

$$A = \left(\underline{\mathcal{O}_i}\right)^{\dagger} \overline{\mathcal{O}_i}.$$

  Analogously, A can also be calculated as

$$A = \left|\mathcal{C}_j \left(\mathcal{C}_j\right|\right)^{\dagger}.$$

B. Balanced model reduction

Identified/realized models frequently have a very high order, which makes them difficult to handle. Model reduction, another subdomain in system theory, reduces the order of models while preserving the dynamical behavior as much as possible. Below we describe the balanced model reduction scheme [7], [8].

The infinite controllability and observability Gramians P_∞ and Q_∞ of a stable system are solutions to the following Lyapunov equations:

$$\begin{aligned} AP_\infty A^T - P_\infty &= -BB^T, \\ A^T Q_\infty A - Q_\infty &= -C^T C. \end{aligned}$$

A state transformation $x_k \rightarrow Tx_k$, $(A, B, C, D) \rightarrow (TAT^{-1}, TB, CT^{-1}, D)$ changes the Gramians in a contragredient way: $(P_\infty, Q_\infty) \rightarrow (TP_\infty T^T, T^{-T} Q_\infty T^{-1})$.

It has been shown [7], [8] that there always exists a matrix T_b such that the transformed Gramians are diagonal and equal to each other:

$$\begin{aligned} P_\infty &\longrightarrow T_b P_\infty T_b^T = \Sigma, \\ Q_\infty &\longrightarrow T_b^{-T} Q_\infty T_b^{-1} = \Sigma, \\ \Sigma &= \operatorname{diag}\left(\left[\begin{array}{cccc} \sigma_1 & \sigma_2 & \dots & \sigma_{n_x} \end{array}\right]\right), \end{aligned}$$

with $\sigma_i \geq \sigma_j > 0,\ 1 \leq i < j \leq n_x$. The corresponding transformed state space model $(A_b, B_b, C_b, D_b) = (T_b A T_b^{-1}, T_b B, C T_b^{-1}, D)$ is called the *balanced representation* of the model.

It can be shown [7], [8] that in balanced form, the states are ordered from most to least important with respect to input-output energy transfer. In that way a reduced order model of order R can be obtained by retaining the R first states:

$$\begin{aligned} A^{[R]} &= A_b(1:R, 1:R), \\ B^{[R]} &= B_b(1:R, :), \\ C^{[R]} &= C_b(:, 1:R), \\ D^{[R]} &= D_b. \end{aligned}$$

C. Decomposition in a basis of damped cosines

In this appendix, the decomposition in complex form (formula (5)) will be written in real from. In general the matrices $\widetilde{A}_y$ and $\widetilde{A}_x$ of the decomposition can be written as

$$\begin{aligned} \widetilde{A}_y &= \operatorname{diag}\left(\left[\begin{array}{c|c|c|c|c} |a_y^{(q_0)}| \ \dots & |a_y^{(q_\omega)}| \exp(j\omega_y^{(q_\omega)}) \ \dots & -|a_y^{(q_\pi)}| \ \dots & |a_y^{(q_{-\omega})}| \exp(j\omega_y^{(q_{-\omega})}) \ \dots \end{array}\right]\right), \\ \widetilde{A}_x &= \operatorname{diag}\left(\left[\begin{array}{c|c|c|c|c} |a_x^{(p_0)}| \ \dots & |a_x^{(p_\omega)}| \exp(j\omega_x^{(p_\omega)}) \ \dots & -|a_x^{(p_\pi)}| \ \dots & |a_x^{(p_{-\omega})}| \exp(j\omega_x^{(p_{-\omega})}) \ \dots \end{array}\right]\right), \end{aligned}$$

with

$$\begin{aligned} q_0 &\in \{1,\ldots,Q_0\}, \\ q_\omega &\in \{Q_0+1,\ldots,Q_0+Q_\omega\}, \\ q_\pi &\in \{Q_0+Q_\omega+1,\ldots,Q_0+Q_\omega+Q_\pi\}, \\ q_{-\omega} &\in \{Q_0+Q_\omega+Q_\pi+1,\ldots,M_y\}, \\ |a_y^{(M_y+Q_0+1-q_\omega)}| &= |a_y^{(q_\omega)}|, \\ \omega_y^{(M_y+Q_0+1-q_\omega)} &= -\omega_y^{(q_\omega)}, \end{aligned}$$

and analogously for p_0, p_ω, p_π and $p_{-\omega}$. Q_0, Q_ω and Q_π are the number of positive elements, complex conjugated pairs and negative elements on the diagonal of $\widetilde{A}_y$ respectively, and analogous for the x-direction. Every combination of a type of elements of $\widetilde{A}_y$ and a type of elements of $\widetilde{A}_x$ will contribute in a different way to the decomposition we are looking for:

$$\begin{aligned} H(l,k) = \; & H_{0,0}(l,k) + H_{0,\omega}(l,k) + H_{0,\pi}(l,k) + \\ & H_{\omega,0}(l,k) + H_{\omega,\omega}(l,k) + H_{\omega,\pi}(l,k) + \\ & H_{\pi,0}(l,k) + H_{\pi,\omega}(l,k) + H_{\pi,\pi}(l,k) \end{aligned}$$

The different contributions are explained below.

Complex conjugated pairs in y- and x-direction: For every q_ω and p_ω, we find a submatrix in P which is of the form:

$$P([q_\omega, M_y+Q_0+1-q_\omega],[p_\omega, M_x+P_0+1-p_\omega]) = \frac{1}{2}\begin{bmatrix} |p_1^{(q_\omega p_\omega)}|e^{j\phi_1^{(q_\omega p_\omega)}} & |p_2^{(q_\omega p_\omega)}|e^{-j\phi_2^{(q_\omega p_\omega)}} \\ |p_2^{(q_\omega p_\omega)}|e^{j\phi_2^{(q_\omega p_\omega)}} & |p_1^{(q_\omega p_\omega)}|e^{-j\phi_1^{(q_\omega p_\omega)}} \end{bmatrix},$$

such that

$$\begin{aligned} H_{\omega,\omega}(l,k) = \; & \sum_{q_\omega}\sum_{p_\omega} \left|a_y^{(q_\omega)}\right|^{l'} \left|a_x^{(p_\omega)}\right|^{k'} \left[|p_1^{(q_\omega p_\omega)}|\cos\left(\omega_y^{(q_\omega)}l' + \omega_x^{(p_\omega)}k' + \phi_1^{(q_\omega p_\omega)}\right) + \right. \\ & \left. + |p_2^{(q_\omega p_\omega)}|\cos\left(\omega_y^{(q_\omega)}l' - \omega_x^{(p_\omega)}k' - \phi_2^{(q_\omega p_\omega)}\right)\right], \end{aligned}$$

with $l' = l-1$ and $k' = k-1$.

Positive real in y-direction and complex conjugated pair in x-direction: For every q_0 and p_ω, we find a submatrix in P which is of the form:

$$P(q_0,[p_\omega,M_x+P_0+1-p_\omega])=\frac{1}{2}\left[\begin{array}{cc} |p^{(q_0p_\omega)}|e^{j\phi^{(q_0p_\omega)}} & |p^{(q_0p_\omega)}|e^{-j\phi^{(q_0p_\omega)}} \end{array}\right],$$

such that

$$H_{0,\omega}(l,k) \;=\; \sum_{q_0}\sum_{p_\omega}\left|a_y^{(q_0)}\right|^{l'}\left|a_x^{(p_\omega)}\right|^{k'}|p^{(q_0p_\omega)}|\cos\left(\omega_x^{(p_\omega)}k'+\phi^{(q_\omega p_\omega)}\right),$$

with $l'=l-1$ and $k'=k-1$. The formulas for $H_{\omega,0}(l,k)$, $H_{\omega,\pi}(l,k)$ and $H_{\pi,\omega}(l,k)$ can be found analogously.

Positive real in y- and negative real in x-direction: For every q_0 and p_π, we find a submatrix of P which is of the form:

$$P(q_0,p_\pi)=|p^{(q_0p_\pi)}|,$$

such that

$$H_{0,\pi}(l,k) \;=\; \left|a_y^{(q_0)}\right|^{l'}\left|a_x^{(p_\pi)}\right|^{k'}|p^{(q_0p_\pi)}|\exp\left(j\pi k'\right),$$

with $l'=l-1$ and $k'=k-1$. The formulas for $H_{0,0}(l,k)$, $H_{\pi,0}(l,k)$ and $H_{\pi,\pi}(l,k)$ can be found analogously.

It should be clear now that a decomposition of the form of formula (5), can be seen as a decomposition in a basis of damped cosines. The modulus and argument of the diagonal elements of $\widetilde{A}_y$ are the *damping numbers* and the *pulsations* in the y-direction, analogously for the diagonal elements of $\widetilde{A}_x$. The modulus and argument of the elements of P will be the *amplitudes* and the *phases* of the cosines respectively.

ACKNOWLEDGEMENTS

Our research is supported by

Research Council KUL: GOA-Mefisto 666, GOA AMBioRICS, several PhD/postdoc and fellow grants; **Flemish Government**: FWO: PhD/postdoc grants, projects, G.0240.99 (multilinear algebra), G.0407.02 (support vector machines), G.0197.02 (power islands), G.0141.03 (Identification and cryptography), G.0491.03 (control for intensive care glycemia), G.0120.03 (QIT), G.0452.04 (new quantum algorithms), G.0499.04 (Robust SVM), G.0499.04 (Statistics) research communities (ICCoS, ANMMM, MLDM); AWI: Bil. Int. Collaboration Hungary/ Poland; IWT: PhD Grants,GBOU (McKnow); **Belgian Federal Science Policy Office**: IUAP P5/22 ('Dynamical Systems and Control: Computation, Identification and Modelling', 2002-2006) ; PODO-II (CP/40: TMS and Sustainability); **EU**: FP5-Quprodis; ERNSI; Eureka 2063-IMPACT; Eureka 2419-FliTE; **Contract Research/agreements**: ISMC/IPCOS, Data4s, TML, Elia, LMS, Mastercard

REFERENCES

[1] C. Burrus and T. Parks, *DFT/FFT and Convolution Algorithms Theory and Implementation*. Wiley, NY, 1985.

[2] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. on Computers*, vol. 23, pp. 90–93, Jan 1974.

[3] M. Vetterli and J. Kovacevic, *Wavelets and Subband Coding*. Englewood Cliffs, NJ, Prentice Hall, 1995.

[4] B. L. Ho and R. E. Kalman, "Effective construction of linear state-variable models from input/output functions," *Regelungstechnik*, vol. 14, no. 12, pp. 545–548, 1966.

[5] H. P. Zeiger and A. J. McEwen, "Approximate linear realizations of given dimension via Ho's algorithm," *IEEE Trans. on Automatic Control*, vol. 19, p. 153, April 1974.

[6] S. Y. Kung, "A new identification and model reduction algorithm via singular value decomposition," in *Proc. of the 12th Asilomar Conference on Circuits, Systems and Computers, Pacific Grove, CA*, 1978, pp. 705–714.

[7] B. C. Moore, "Principal component analysis in linear systems: Controllability, observability and model reduction," *IEEE Trans. on Automatic Control*, vol. 26, pp. 17–32, Feb 1981.

[8] L. Pernebo and L. Silverman, "Model reduction via balanced state space representation," *IEEE Trans. on Automatic Control*, vol. 27, pp. 382–387, April 1982.

[9] M. Sznaier, O. Camps, and C. Mazzaro, "Finite horizon model reduction of a class of neutrally stable systems with applications to texture synthesis and recognition," in *Proc. of the Conference on Decision and Control (CDC2004), Bahamas*, Dec 2004, pp. 3068–3073.

[10] G. Doretto, A. Chiuso, S. Soatto, and Y. Wu, "Dynamic textures," *International Journal of Computer Vision*, vol. 51(2), pp. 91–109, 2003.

[11] R. Penrose, "A generalized inverse for matrices," in *Proc. of the Cambridge Philosophical Society*, 1955, pp. 51:406–413.