

Experiments with a Wavelet-Based Approximate Proper Orthogonal Decomposition

Geert Uytterhoeven Dirk Roose

Report TW263, September 1997



Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

Experiments with a Wavelet-Based Approximate Proper Orthogonal Decomposition

Geert Uytterhoeven *Dirk Roose*

Report TW 263, September 1997

Department of Computer Science, K.U.Leuven

Abstract

The Proper Orthogonal Decomposition (POD) or Karhunen-Loève Transform (KLT) is a powerful tool to obtain low-dimensional models for large scale dynamical systems, described by partial differential equations. Starting from a set of solutions (obtained by experiment or computation), called snapshots, the method computes an “optimal” basis of eigenmodes for the snapshots, which can be used to construct a low-dimensional model of the dynamical system. Unfortunately the construction of a POD and also the projection of the original problem onto the eigenmodes are very expensive operations.

We have done experiments with an Approximate Proper Orthogonal Decomposition, based on data compression using symmetric biorthogonal wavelets. A wavelet packet transform allows to decorrelate and compress the snapshots, after which we compute the POD of the compressed data set of much lower dimension, while we still obtain good approximations for the eigenmodes of the original system.

We will present results for the computation of an Approximate POD for a system of partial differential equations (the one-dimensional Brusselator reaction-diffusion model), using symmetric wavelets (Cohen-Daubechies-Feauveau).

Keywords : wavelets, dynamical systems

AMS(MOS) Classification : 41A30, 58G28.

Experiments with a Wavelet-Based Approximate Proper Orthogonal Decomposition

Geert Uytterhoeven*

Dirk Roose

Department of Computer Science — Katholieke Universiteit Leuven†

Abstract

The Proper Orthogonal Decomposition (POD) or Karhunen-Loève Transform (KLT) is a powerful tool to obtain low-dimensional models for large scale dynamical systems, described by partial differential equations. Starting from a set of solutions (obtained by experiment or computation), called snapshots, the method computes an “optimal” basis of eigenmodes for the snapshots, which can be used to construct a low-dimensional model of the dynamical system. Unfortunately the construction of a POD and also the projection of the original problem onto the eigenmodes are very expensive operations.

We have done experiments with an Approximate Proper Orthogonal Decomposition, based on data compression using symmetric biorthogonal wavelets. A wavelet packet transform allows to decorrelate and compress the snapshots, after which we compute the POD of the compressed data set of much lower dimension, while we still obtain good approximations for the eigenmodes of the original system.

We will present results for the computation of an Approximate POD for a system of partial differential equations (the one-dimensional Brusselator reaction-diffusion model), using symmetric wavelets (Cohen-Daubechies-Feauveau).

1 The Proper Orthogonal Decomposition

The POD (Proper Orthogonal Decomposition, also known as KLT — Karhunen-Loève Transform) is a tool to analyze a given set of data by decomposing the data into an “ideal” orthonormal basis.

With “ideal” basis we mean an orthonormal basis that gives the best approximation using as few basis vectors as possible. The basis vectors are ordered such that the best approximation using only i vectors is obtained by using the first i vectors. Thus they are ordered by decreasing “importance”.

Suppose we have S linear independent snapshots X_i of size N (i.e. linear independent vectors with N elements) with $S \ll N$. We assume that the average of the snapshots

$$\bar{X} = \frac{1}{S} \sum_{i=0}^{S-1} X_i$$

*Geert.Uytterhoeven@cs.kuleuven.ac.be, <http://www.cs.kuleuven.ac.be/~geert/>

†Celestijnenlaan 200A, B-3001 Heverlee, Belgium

is zero. If this would not be the case, we can always subtract the average \bar{X} from all snapshots:

$$X_i \leftarrow X_i - \bar{X}, \quad i = 0, \dots, S-1.$$

One can show that the eigenvectors E_k of the $N \times N$ covariance matrix C

$$C = \frac{1}{S} \sum_{i=0}^{S-1} X_i X_i^T,$$

satisfying

$$C E_k = \lambda_k E_k, \quad k = 0, \dots, S-1$$

with

$$\lambda_0 \geq \lambda_1 \geq \dots \geq \lambda_{S-1} > 0$$

form the best basis we are looking for (the $N - S$ remaining eigenvalues of C are zero). Since C is symmetric, the eigenvalues λ_k are real and the eigenvectors E_k are orthogonal. For every eigenvector E_k , the corresponding eigenvalue λ_k indicates the relative importance of that eigenvector in the best basis. The eigenvectors are also called the “eigenmodes” of the original snapshots.

Although this approach results in the “best basis” and is ideal in terms of decorrelation, it is not ideal in terms of computing time. Constructing the covariance matrix C requires $\mathcal{O}(SN^2)$ operations.

Fortunately, one can use an alternative method, called the “method of the snapshots” [8]. Here one uses the property that the eigenvectors E_k are unique linear combinations of the linearly independent snapshots X_i and thus can be written as

$$E_k = \sum_{j=0}^{S-1} a_{k,j} X_j, \quad k = 0, \dots, S-1.$$

Now the eigenvalue equation becomes:

$$\begin{aligned} C E_k &= \lambda_k E_k, \\ \frac{1}{S} \sum_{i=0}^{S-1} X_i X_i^T \sum_{j=0}^{S-1} a_{k,j} X_j &= \lambda_k \sum_{j=0}^{S-1} a_{k,j} X_j, \\ \frac{1}{S} \sum_{i=0}^{S-1} X_i \left(\sum_{j=0}^{S-1} X_i^T X_j a_{k,j} \right) &= \sum_{i=0}^{S-1} \lambda_k a_{k,i} X_i. \end{aligned}$$

Because the X_i form a basis, we can drop the summations on both sides of the last equation and we get

$$\sum_{j=0}^{S-1} X_i^T X_j a_{k,j} = \lambda_k^* a_{k,i}, \quad \begin{cases} i = 0, \dots, S-1, \\ k = 0, \dots, S-1, \\ \lambda_k^* = S \lambda_k. \end{cases}$$

Define the matrix C^* as

$$C^* = [c_{i,j}^*] = [X_i^T X_j], \quad i, j = 0, \dots, S-1,$$

and let the vector $A_k = [a_{k,i}], i = 0, \dots, S - 1$, denote the coefficients of the eigenvector E_k in the basis of the snapshots, then the original eigenvalue problem is equivalent to the eigenvalue problem

$$C^* A_k = \lambda_k^* A_k, \quad \text{with } \lambda_k^* = S \lambda_k.$$

The construction of the matrix C^* requires $\mathcal{O}(NS^2)$ operations, which is much cheaper than the construction of the covariance matrix C , since $C \ll N$.

If $S > N$, the snapshots X_i are not independent and thus they no longer form a basis. As a consequence, the matrix C^* will have rank $r \leq N$. However, the method of the snapshots can still be used, because we have for $k = r, \dots, S - 1$:

- the additional eigenvalues λ_k will all be zero because of the rank deficiency of C^* ,
- the additional eigenvectors A_k will be unit vectors (up to a constant) because the mean of the snapshots is zero, and the resulting additional eigenmodes E_k will be zero. Note that in a strict sense the zero vector cannot be an eigenmode, but here we are considering it in a more general framework.

Note:

We can “normalize” the snapshots

$$X_i \leftarrow \frac{X_i}{\|X_i\| \sqrt{S}} \tag{1}$$

such that

$$\sum_{k=0}^{S-1} \lambda_k^* = 1.$$

Indeed, the sum of the eigenvalues of a matrix is equal to the sum of the diagonal elements of that matrix. In that case eigenmode E_k contains $100 \times \lambda_k^*$ % of the total energy.

2 The Approximate Proper Orthogonal Decomposition

The main idea behind the Approximate Proper Orthogonal Decomposition (APOD) is to decorrelate the original snapshots using wavelets and perform the normal Proper Orthogonal Decomposition (POD) on the compressed snapshots, in the hope that we will get a reasonable approximation to the original eigenmodes, with less computational effort.

2.1 Theory of Operation

To calculate the APOD, we perform the following operations:

1. Transform the snapshots X_i using an appropriate transform W that decorrelates the original data:

$$\hat{X}_i = W(X_i), \quad i = 0, \dots, S - 1.$$

2. Compress the vectors \widehat{X}_i by removing from every vector a number of entries:

$$\widehat{X}_i \rightsquigarrow \widehat{X}'_i \in \mathbf{R}^n, \quad i = 0, \dots, S-1,$$

with $n \ll N$.

Removing $m\%$ entries yields a compression rate of $\frac{100}{100-m} : 1$. The strategy to decide which entries to remove can be adapted to the set of snapshots, i.e. based on statistical properties of the set, or can be fixed, i.e. based on a previous statistical analysis.

3. Calculate the POD of the compressed snapshots \widehat{X}'_i . This gives the eigenmodes $\widehat{E}'_i \in \mathbf{R}^n$, while the eigenvalues λ_i indicate the energy distribution over the eigenmodes.
4. Decompress the eigenmodes \widehat{E}'_i by putting zeroes at the places where a vector entry was removed in step 2:

$$\widehat{E}'_i \rightsquigarrow \widehat{E}_i, \quad i = 0, \dots, S-1.$$

5. Perform the inverse transform W^{-1} on the eigenmodes \widehat{E}_i to get the approximate eigenmodes E_i^* , which are an approximation to the eigenmodes E_i of the snapshots X_i :

$$E_i^* = W^{-1}(\widehat{E}_i) \approx E_i, \quad i = 0, \dots, S-1.$$

A schematic overview of these operations can be found in fig. 1.

The APOD can be used instead of the genuine POD, for two possible goals:

- to reduce the computation time needed to construct (an approximation of) the eigenmodes E_i ,
- to use the APOD to construct a “low dimensional” model, based on the eigenmodes \widehat{E}'_i .

2.2 Decorrelating Transforms

We have chosen to use a Wavelet Packet Transform as the decorrelating transform. A wavelet packet basis is inherently suited for an adaptive “best basis” concept [11]. Furthermore it can be calculated with a complexity of $\mathcal{O}(n \log n)$. More details about wavelets and wavelet packets will be given in section 3. There we outline the use of wavelets to compute the APOD.

2.2.1 Orthogonal Transforms

If the transform W and its inverse W^{-1} are orthogonal transforms, the APOD will yield the same results as the POD if there is no compression. Indeed, because the POD is an *orthogonal* decomposition, applying an orthogonal transform before and its inverse after the POD has no influence on the result.

In the case of compression, we have the following: if we decompose a function in an orthonormal basis $\varphi_i(x)$,

$$f(x) = \sum_k a_k \varphi_k(x),$$

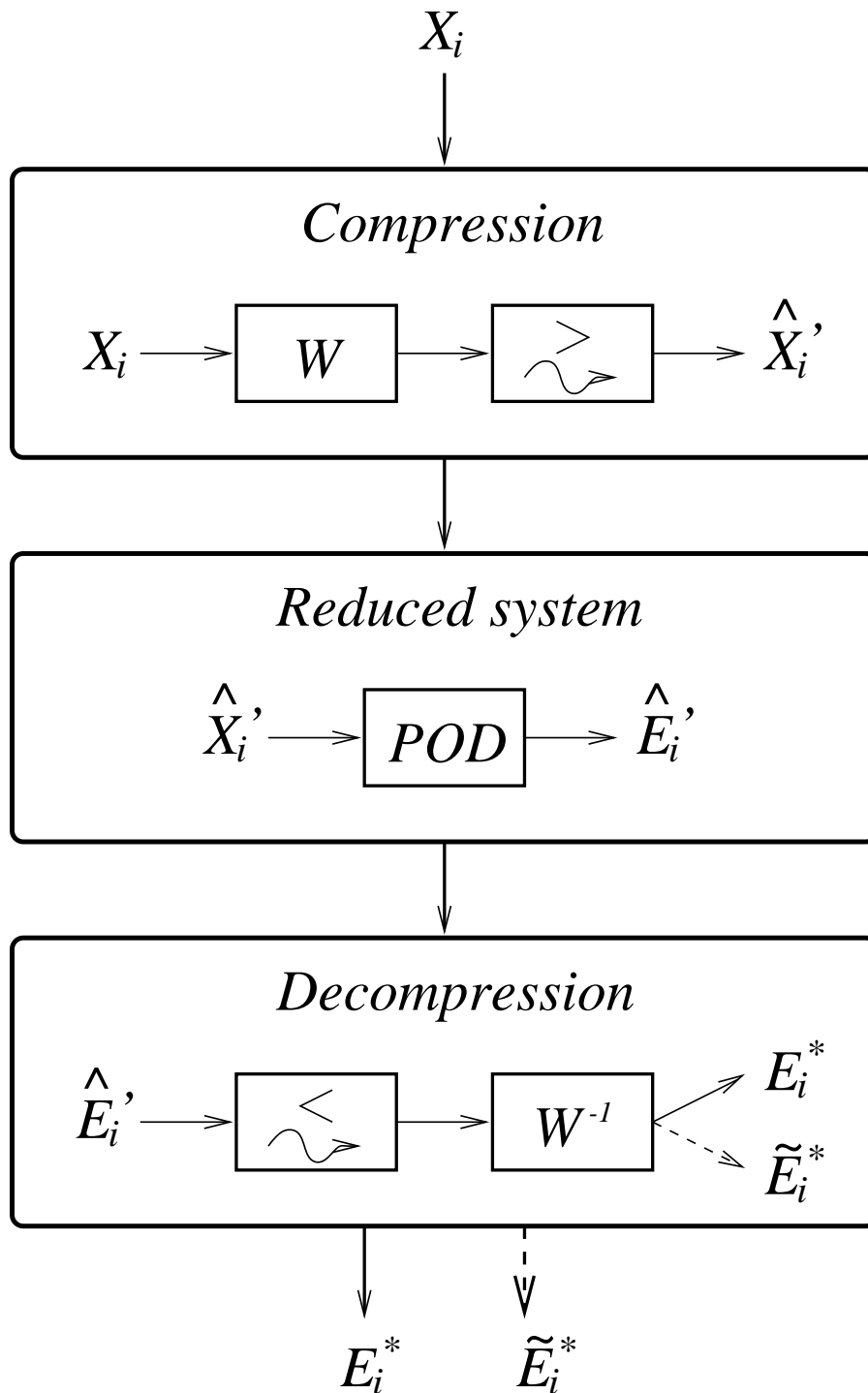


Figure 1: Schematic overview of the Approximate Proper Orthogonal Decomposition. The \tilde{E}_i^* are only relevant if W and W^{-1} are biorthogonal transforms.

Parseval's identity holds:

$$\sum_k a_k^2 = \|f\|^2.$$

Neglecting the smallest coefficients a_k will result in only a small error after reconstruction. In addition the orthogonality guarantees a good approximation.

Of course the results also depend on the criterion used in step 2.

2.2.2 Biorthogonal Transforms

If the transform W and its inverse W^{-1} are not orthogonal but only biorthogonal transforms, we cannot guarantee a good approximation of the real POD in all circumstances. In fact, if we do not compress the transformed snapshots, we will not even get the same eigenmodes as with the real POD.

However, if the transform is based on biorthogonal wavelets, the fact that the resulting basis is a Riesz basis will guarantee no longer a perfect but still a good approximation. In a Riesz basis, the coefficients of the decomposition show the following property:

$$A\|f\|^2 \leq \sum_k a_k^2 \leq B\|f\|^2, \quad (2)$$

with A and B positive constants¹.

Neglecting the smallest coefficients a_k will still result in only a small controllable error after reconstruction.

Note that in the biorthogonal case we not only have primal eigenmodes E_i^* , but also dual eigenmodes \tilde{E}_i^* , which are biorthogonal with respect to the primal eigenmodes, as shown below.

2.3 Decomposition into the Basis of Approximate Eigenmodes

If we want to write an arbitrary snapshot as a linear combination of the approximate normalized eigenmodes

$$X \approx \sum_i a_i E_i^*,$$

we can obtain the unknown coefficients a_i by calculating inner products with the eigenmodes, or the dual eigenmodes in the biorthogonal case:

- in case an orthogonal transform is used:

$$a_i = \langle X, E_i^* \rangle.$$

- in case a biorthogonal transform is used:

$$a_i = \langle X, \tilde{E}_i^* \rangle.$$

¹In the orthogonal case we have $A = B = 1$, i.e. we have Parseval's identity.

However, we can also obtain the coefficients a_i by calculating them in the wavelet domain. Let \hat{X}' be the result of the wavelet transform and compression of the input image X

$$\begin{aligned}\hat{X} &= W(X), \\ \hat{X} &\rightsquigarrow \hat{X}'.\end{aligned}$$

Due to the (bi)orthogonality of the wavelet packet transform and the orthogonality of the eigenmodes \hat{E}'_i of the transformed and compressed snapshots, we have

$$a_i = \langle \hat{X}', \hat{E}'_i \rangle.$$

Indeed, we have in the biorthogonal case:

$$\begin{aligned}a_i &= \langle X, \tilde{E}_i^* \rangle, \\ &= \langle \sum_j b_j \Psi_j, \sum_k c_{ik} \tilde{\Psi}_k \rangle, \\ &= \sum_{j,k} b_j c_{ik} \langle \Psi_j, \tilde{\Psi}_k \rangle, \\ &= \sum_j b_j c_{ij}, \\ &= \langle B, C_i \rangle, \\ &= \langle \hat{X}', \hat{E}'_i \rangle,\end{aligned}$$

with Ψ_i and $\tilde{\Psi}_i$ the biorthonormal basis functions in the wavelet domain, and $B = [b_i]$ and $C = [c_{i,j}]$.

Reconstruction in the wavelet domain is straightforward:

$$\begin{aligned}\hat{X}' &= \sum_i a_i \hat{E}'_i, \\ \hat{X}' &\rightsquigarrow \hat{X}, \\ X &\approx W^{-1}(\hat{X}).\end{aligned}$$

Which approach for the calculation of the coefficients a_i is most efficient depends on a number of factors. If we denote the number of retained eigenmodes by s and we assume that the images are square and count $\sqrt{N} \times \sqrt{N}$ pixels, we have:

- in the original domain:
 - s inner products in \mathbf{R}^N : $\mathcal{O}(sN)$,
- in the wavelet domain:
 - S wavelet packet transforms: $\mathcal{O}(SN \log(\sqrt{N}))$,
 - s inner products in \mathbf{R}^n : $\mathcal{O}(sn)$.

3 Wavelets

3.1 Multiresolution Analysis

Consider the space \mathbf{L}^2 , the vector space of square integrable functions in \mathbf{R} :

$$\mathbf{L}^2 = \left\{ f : \int_{-\infty}^{+\infty} f^2(x) dx < \infty \right\}.$$

In a multiresolution analysis [7] we decompose \mathbf{L}^2 in nested subspaces V_j

$$\cdots \subset V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset \cdots$$

such that the closure of their union is \mathbf{L}^2

$$\overline{\bigcup_{j=-\infty}^{+\infty} V_j} = \mathbf{L}^2,$$

and their intersection contains only the zero-function

$$\bigcap_{j=-\infty}^{+\infty} V_j = \{0\}.$$

In the dyadic case, a function $f(x)$ that belongs to one of these subspaces V_j has the following properties:

$$f(x) \in V_j \Leftrightarrow \text{dilation } f(2x) \in V_{j+1}, \quad (3)$$

$$f(x) \in V_0 \Leftrightarrow \text{translation } f(x+1) \in V_0. \quad (4)$$

If we can find a function $\varphi(x) \in V_0$ such that the set of functions consisting of $\varphi(x)$ and its integer translates

$$\{\varphi(x-k)\}_{k \in \mathbf{Z}}$$

form a basis for the space V_0 , we call it a *scaling function*. For the other subspaces $V_{j \neq 0}$ we define:

$$\varphi_{j,k}(x) \equiv 2^{j/2} \varphi(2^j x - k).$$

3.2 Wavelet Functions

Because the subspaces V_j are nested:

$$V_j \subset V_{j+1},$$

we can decompose V_{j+1} in V_j and W_j , the orthogonal complement of V_j in V_{j+1} :

$$\begin{aligned} V_j \oplus W_j &= V_{j+1}, \\ W_j &\perp V_j. \end{aligned}$$

The direct sum of the subspaces W_j is equal to \mathbf{L}^2 :

$$\overline{\bigcup_{j=-\infty}^{+\infty} V_j} = \bigoplus_{j=-\infty}^{+\infty} W_j = \mathbf{L}^2.$$

This means that V_j is a “coarse resolution” representation of V_{j+1} , while W_j carries the “high resolution” difference information between V_{j+1} and V_j .

If we can find a function $\psi(x) \in W_0$ that obeys the translation property (4), i.e.

$$\psi(x) \in W_0 \Leftrightarrow \text{translation } \psi(x+1) \in W_0,$$

and such that the set of functions consisting of $\psi(x)$ and its integer translates

$$\{\psi(x-k)\}_{k \in \mathbf{Z}}$$

form a basis for the space W_0 , we call it a *wavelet function*. For the other subspaces $W_{j \neq 0}$ we define:

$$\psi_{j,k}(x) \equiv 2^{j/2} \psi(2^j x - k).$$

3.3 The Fast Wavelet Transform (FWT)

Because both V_0 and W_0 are subspaces of V_1 :

$$V_0 \subset V_1 \text{ and } W_0 \subset V_1,$$

we can express $\varphi(x)$ and $\psi(x)$ in terms of the basis functions of V_1 :

$$\begin{aligned} \varphi(x) &= 2 \sum_k h_k \varphi(2x - k), \\ \psi(x) &= 2 \sum_k g_k \varphi(2x - k). \end{aligned}$$

Due to the multiresolution analysis, these relations are also valid between V_{j+1} , V_j and W_j for arbitrary j .

We call the h_k and g_k the filter coefficients that uniquely define the scaling function $\varphi(x)$ and the wavelet $\psi(x)$.

Since $V_{j+1} = V_j \oplus W_j$, we can express a function $f(x)$ that is written in terms of the basis functions of V_1 in the basis functions of V_0 and W_0 also:

$$\begin{aligned} f(x) &= \sum_k \lambda_{j+1,k} \varphi_{j+1,k}(x), \\ &= \sum_l \lambda_{j,l} \varphi_{j,l}(x) + \sum_l \gamma_{j,l} \psi_{j,l}(x). \end{aligned}$$

with the transform coefficients $\lambda_{j,l}$ and $\gamma_{j,l}$ defined by:

$$\begin{aligned} \lambda_{j,l} &= \sqrt{2} \sum_k h_{k-2l} \lambda_{j+1,k}, \\ \gamma_{j,l} &= \sqrt{2} \sum_k g_{k-2l} \lambda_{j+1,k}. \end{aligned}$$

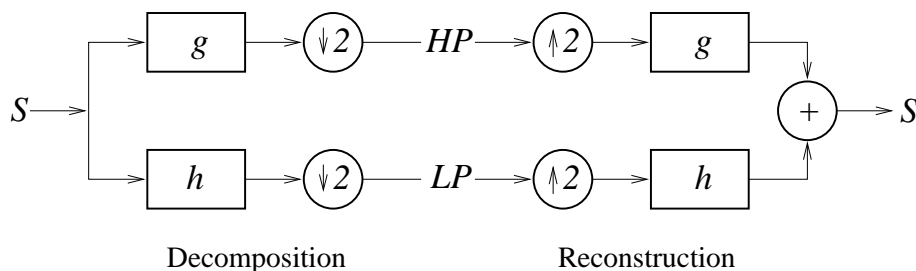


Figure 2: The filterbank algorithm.

The filter h_k is a low-pass filter, while g_k is a high-pass filter. This operation is known as the *Fast Wavelet Transform*. The inverse operation can be obtained in a similar way. This algorithm is known as the filterbank algorithm (fig. 2).

3.4 Orthogonal Wavelets

If the $\varphi_{j,k}(x)$ and $\psi_{j,k}(x)$ are orthogonal:

$$\begin{aligned} V_j &\perp W_j, \\ \langle \varphi_{j,l}, \varphi_{j,l'} \rangle &= \delta_{l-l'}, \\ \langle \psi_{j,l}, \psi_{j',l'} \rangle &= \delta_{j-j'} \delta_{l-l'}, \end{aligned}$$

the decomposition in the wavelet basis is guaranteed to be stable and we can calculate the coefficients of the decomposition

$$f(x) = \sum_l \lambda_{j,l} \varphi_{j,l}(x) + \sum_l \gamma_{j,l} \psi_{j,l}(x)$$

by taking inner products with the scaling and wavelet functions:

$$\begin{aligned} \lambda_{j,l} &= \langle f, \varphi_{j,l} \rangle, \\ \gamma_{j,l} &= \langle f, \psi_{j,l} \rangle. \end{aligned}$$

Examples of orthogonal wavelets are the orthogonal wavelets constructed by Daubechies [4].

3.5 Biorthogonal Wavelets

If we relax the orthogonality conditions to biorthogonality conditions we can obtain wavelets with some special properties that are not possible with orthogonal wavelets. In the biorthogonal case we have two multiresolution analyses, a primal and a dual:

- primal: $V_j, W_j, \varphi_{j,k}, \psi_{j,k}$,
- dual: $\tilde{V}_j, \tilde{W}_j, \tilde{\varphi}_{j,k}, \tilde{\psi}_{j,k}$.

The biorthogonality conditions imply:

$$\begin{aligned}\tilde{V}_j &\perp W_j, \\ V_j &\perp \tilde{W}_j, \\ \langle \tilde{\varphi}_{j,l}, \varphi_{j,l'} \rangle &= \delta_{l-l'}, \\ \langle \tilde{\psi}_{j,l}, \psi_{j',l'} \rangle &= \delta_{j-j'} \delta_{l-l'}.\end{aligned}$$

Because the biorthogonal wavelets form a Riesz-basis (eq. 2), the decomposition in the biorthogonal wavelet basis is still stable and the coefficients of the decomposition

$$f(x) = \sum_l \lambda_{j,l} \varphi_{j,l}(x) + \sum_l \gamma_{j,l} \psi_{j,l}(x)$$

are now obtained by calculating inner products with the dual basis functions:

$$\begin{aligned}\lambda_{j,l} &= \langle f, \tilde{\varphi}_{j,l} \rangle, \\ \gamma_{j,l} &= \langle f, \tilde{\psi}_{j,l} \rangle.\end{aligned}$$

We can still use the filterbank algorithm if we use the dual filterpair (\tilde{h}, \tilde{g}) (related to $\tilde{\varphi}(x)$ and $\tilde{\psi}(x)$) for the decomposition and the primal filterpair (h, g) (related to $\varphi(x)$ and $\psi(x)$) for the reconstruction.

3.6 The Wavelet Decomposition Tree

The subspaces V_j are nested, and each of them can be split in two subspaces V_{j-1} and W_{j-1} . If we start with the highest resolution subspace V_0 , we perform the following decompositions:

$$\begin{aligned}V_0 &= V_{-1} \oplus W_{-1}, \\ &= V_{-2} \oplus W_{-2} \oplus W_{-1}, \\ &= V_{-3} \oplus W_{-3} \oplus W_{-2} \oplus W_{-1}, \\ &\vdots\end{aligned}$$

and we obtain a decomposition tree as in fig. 3.

3.7 Higher Dimensions

The above wavelets were defined on the real line \mathbf{R} , i.e. on a one-dimensional domain. To create wavelets on higher dimensional domains, we can follow two approaches:

- We can perform the wavelet transform independently for each dimension. Because the wavelet transform can be written as a multiplication with a wavelet transform matrix and due to the associativity of the matrix product, the exact order is not important.

In the two-dimensional case we get a decomposition as shown in fig. 4. The square variant is most used and in that case the basis functions are tensor products of the one-dimensional basis functions, i.e. after one transform step we have:

$$\frac{\varphi(x) \otimes \varphi(x) \mid \varphi(x) \otimes \psi(x)}{\psi(x) \otimes \varphi(x) \mid \psi(x) \otimes \psi(x)}$$

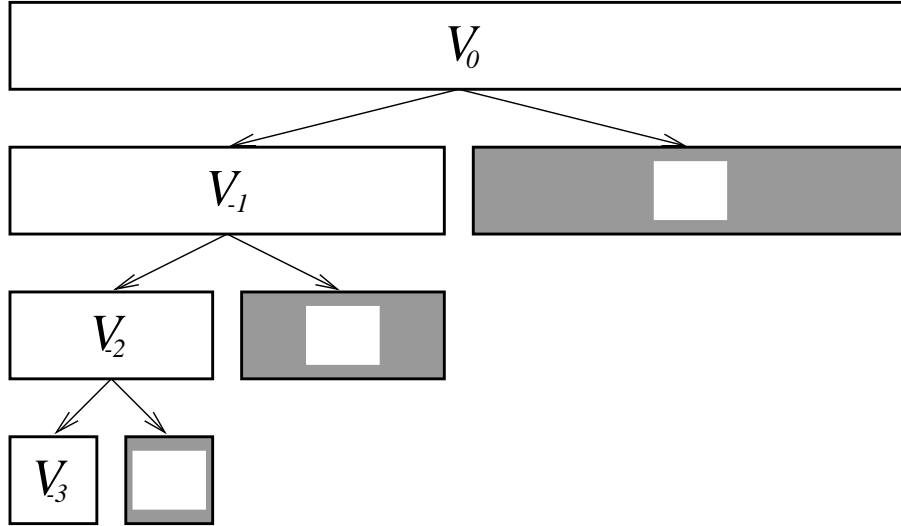


Figure 3: The wavelet decomposition tree.

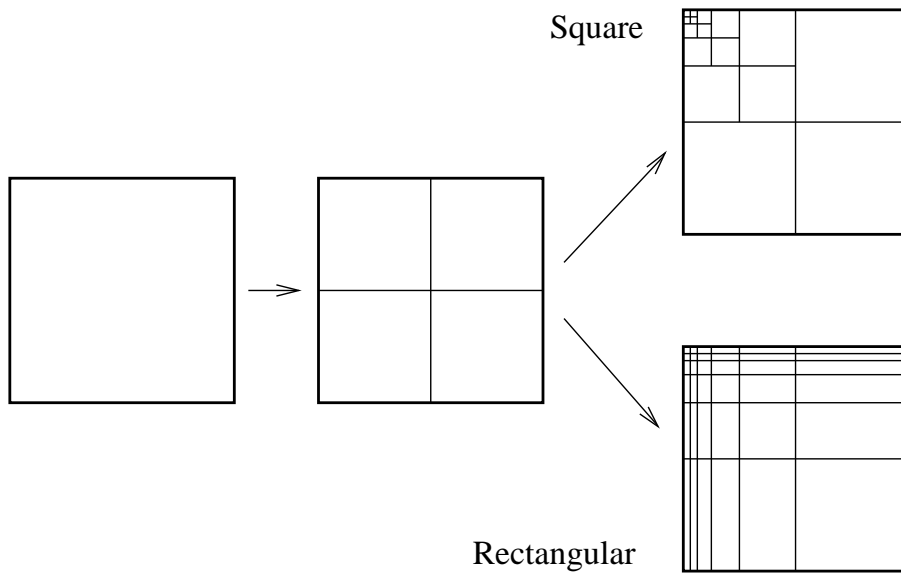


Figure 4: The two-dimensional wavelet transform: the square and the rectangular variant.

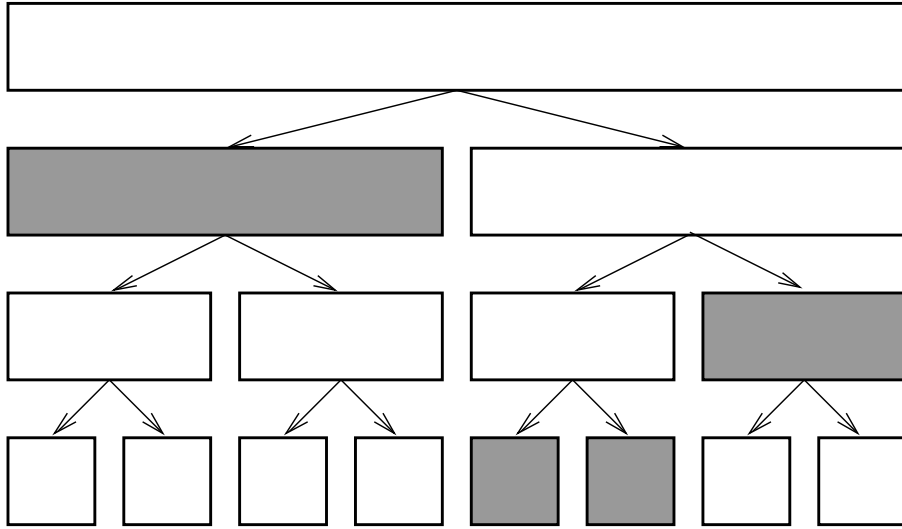


Figure 5: A wavelet packet decomposition tree.

- An alternative is to use real multi-dimensional wavelets, which are less anisotropic with respect to directional information in the original data.

3.8 Wavelet Packets

In every step of the wavelet transform the low frequency data of the signal is split in a low and a high frequency part, as can be seen in fig. 3. After decomposition the low frequency part with the lowest index and the high frequency parts will be stored.

In wavelet packet analysis, one also splits the high frequency part in a “low” and a “high” frequency part [12, 3]. This produces a decomposition tree as shown in the figures 5 (1D) and 6 (2D), and leads to a redundant tree of possible basis functions.

The aim of wavelet packet analysis is to choose a “best basis”, i.e. to find the set of functions that best decorrelates the input data and that is a subset of the basis functions in the decomposition tree.

If we transform a square image $X = [x_{ij}]$ with $n \times n$ elements, the two-dimensional wavelet packet decomposition tree $Y = [y_{ijk}]$ has $n \times n \times \log(n)$ elements, with a topology of $\log(n)$ “levels” of $n \times n$ elements.

3.8.1 Selection of the “Best Basis”

The selection of the best basis is equivalent to the answer to the question ‘Do we best split this part into a low and a high frequency part or not?’. We can make these decisions by starting at the bottom level of the tree in fig. 5 or 6 and comparing the “cost” of two (1D) respective four (2D) neighbouring “child” squares in this level with the “cost” of the “parent” square in the higher level. The “cost” can be seen as a measure for the number of bits we would need

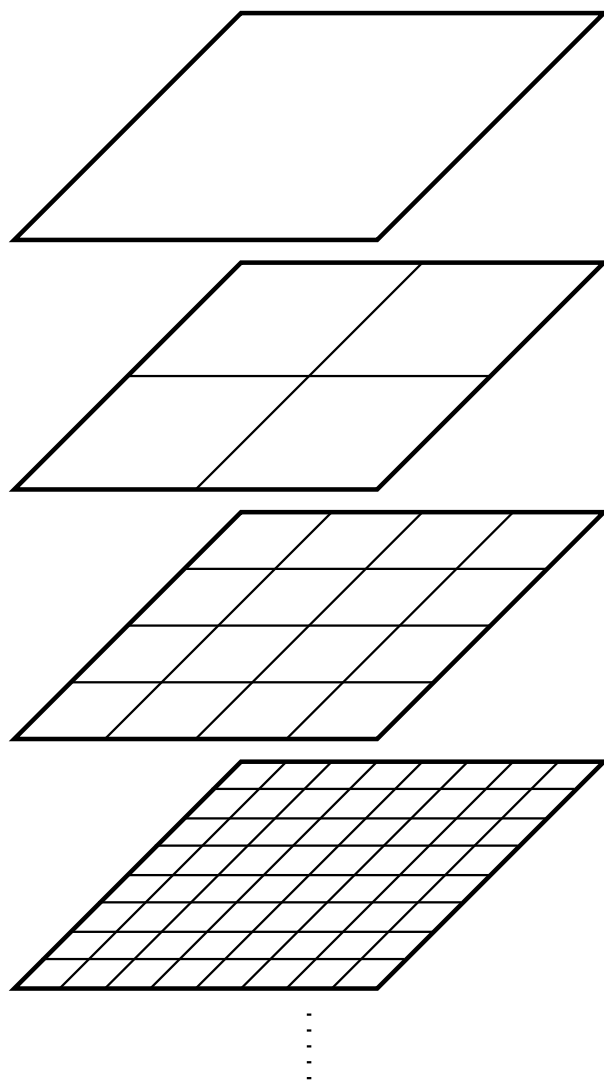


Figure 6: The 2D wavelet packet decomposition tree.

to store the data in the corresponding basis, i.e. a measure for the achieved decorrelation.

We retain the cheapest squares and proceed in the same way with the higher level, until we arrive at the top level. The basis functions that correspond to the retained squares form the best basis.

To calculate the “cost” of a matrix $v = [v_{ij}]$, we have several possibilities, e.g.:

- **1-Norm** based cost function:

$$\text{cost} = \sum_{i,j} |v_{ij}|.$$

- **Shannon Entropy** based cost function:

$$\text{cost} = - \sum_{i,j} v_{ij}^2 \log |v_{ij}|.$$

- **Threshold** based cost function:

$$\text{cost} = \#V_T \quad \text{with } V_T = \{v_{ij} \mid |v_{ij}| > T\},$$

with T a given threshold.

- **Logarithm of Energy** based cost function:

$$\text{cost} = \sum_{i,j} \log(v_{ij})^2,$$

which is equivalent with

$$\text{cost} = \sum_{i,j} \log |v_{ij}|.$$

3.8.2 Compression

To compress the representation in the best basis, we remove the $m\%$ coefficients that are the smallest in magnitude.

3.8.3 Best Basis Selection and Compression for a Set of Snapshots

If we want to find the best basis for a *set* of snapshots, we must choose the *same* wavelet packet basis for every snapshot. This basis must be based on the statistical properties of the set of snapshots.

To select this basis, we calculate the wavelet packet decomposition tree Y_l of every snapshot X_l

$$X_l = [x_{i,l}] \xrightarrow{\text{WPT}} Y_l = [y_{ij,l}],$$

square all entries in the Y_l and sum the corresponding entries of Y_l to obtain a “sum-squared” wavelet packet decomposition tree Y^* :

$$Y^* = [y_{ij}^*] \quad \text{with} \quad y_{ij}^* = \sum_l y_{ij,l}^2.$$

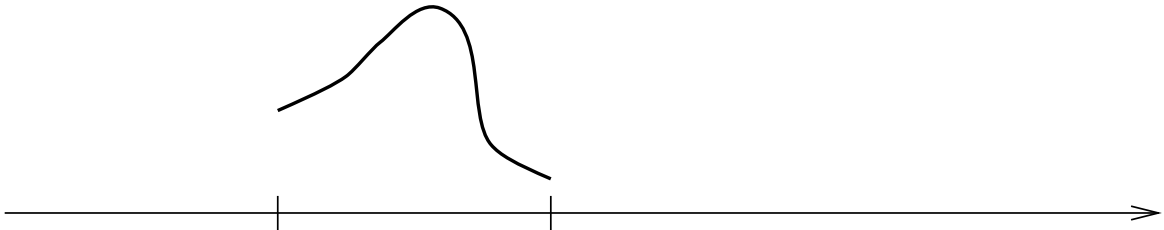


Figure 7: A finite signal.

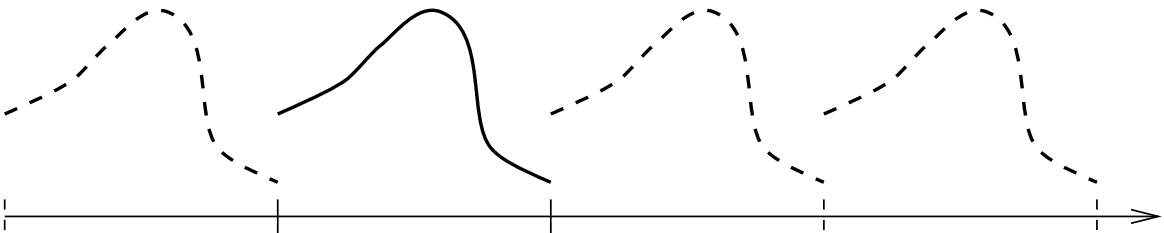


Figure 8: Periodic extension of a finite signal.

Then we apply the best basis selection algorithm to the “sum-squared” wavelet packet decomposition tree Y^* .

To compress the snapshots in their best basis representation, we remove the coefficients from the Y_l that correspond to the $m\%$ smallest coefficients (in absolute value) in the tree Y^* .

4 Signal Extensions

The classical wavelet transforms are defined for functions defined on the whole real axis (in the one-dimensional case). However, real world signals are finite (fig. 7), and padding the signal with zeroes would lead to much more wavelet coefficients than samples. If we truncate the transformed function to yield the same number of wavelet coefficients as samples, this will badly distort the signal after reconstruction. What we need is some kind of signal extension.

4.1 Periodic Extension

A first solution is the periodic signal extension: we extend the finite signal by putting copies of itself in front of and behind the original signal (fig. 8).

After a wavelet transform we can simply discard the coefficients that lie outside of the interval in which the original signal was defined, since these wavelet coefficients are simply the same as the retained coefficients, and thus can be recovered without any problems.

However, unless the first and the last sample have the same value, we introduce unwanted discontinuities at the boundaries of the original signal. These discontinuities will locally enlarge the wavelet coefficients and make compression of the signal more difficult.

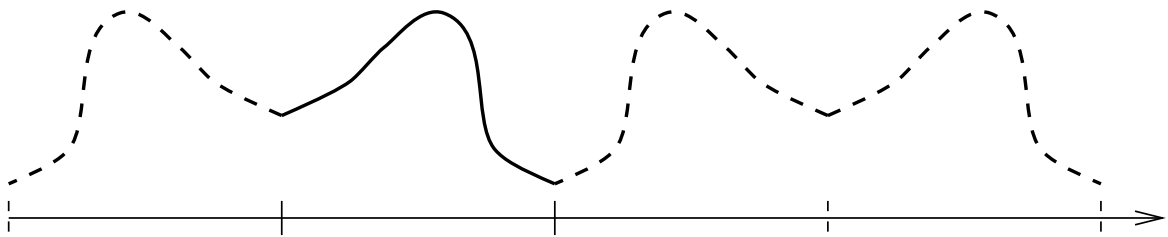


Figure 9: Symmetric extension of a finite signal.

4.2 Symmetric Extension

A better solution is the symmetric extension of the signal: we extend the finite signal by mirroring it around its endpoints, which makes the discontinuities disappear (fig. 9). But the higher order derivatives may still be discontinuous.

Symmetric extension doubles the number of wavelet coefficients, unless we use symmetric wavelets (cfr. DCT). That is why we use biorthogonal wavelets instead of orthogonal wavelets, since they can be symmetric.

4.3 Wavelets on an Interval

An alternative would be to use so called “second generation wavelets” (e.g. generated by the Lifting Scheme). They give us additional freedom we can use to make them defined on an interval [10, 9].

5 Wavelet Transforms

Let us denote the primal filterpair that is characteristic for a specific wavelet transform by (g, h) and the dual filterpair by (\tilde{g}, \tilde{h}) .

5.1 The Symmetric Wavelet Transform

While the periodic extension of a discrete (sampled) signal is straightforward and unique, there are four possible ways to extend a discrete signal symmetrically. One can duplicate the first and the last sample value or not. These cases are denoted by the notation (a, b) . If $a = 1$, the first sample value is not duplicated, while $a = 2$ indicates that the first sample value is duplicated. The value of b indicates the possible duplication of the last sample value (fig. 10).

However, to ensure to get a perfect reconstruction, one has to apply the correct extension in every step of the decomposition and reconstruction, as indicated in fig. 11 [1].

5.2 Wavelet Transforms

All wavelet transform operations on a vector can be denoted by matrix multiplications.

We use the following notations:

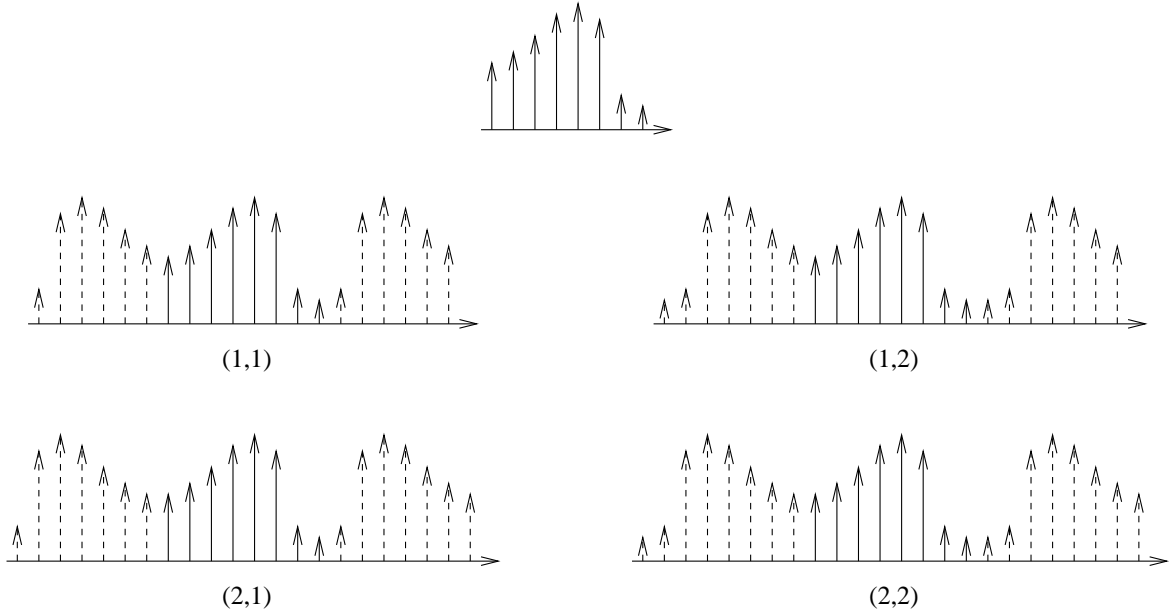


Figure 10: The four cases of symmetric signal extension of a discrete finite signal.

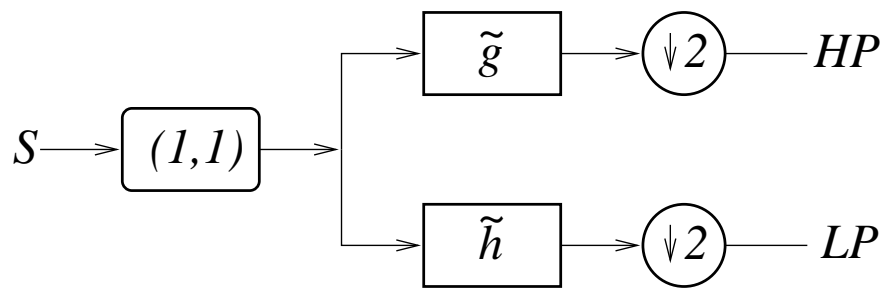
- W** Transformation matrix for the forward wavelet transform algorithm, using the filterpair (g, h) ,
- W*** Transformation matrix for the inverse wavelet transform algorithm, using the filterpair (g, h) ,
- W^T** Transpose (adjoint) of the transformation matrix for the forward wavelet transform algorithm, using the filterpair (g, h) ,
- W*^T** Transpose (adjoint) of the transformation matrix for the inverse wavelet transform algorithm, using the filterpair (g, h) .

The transform matrices for the corresponding dual transforms using the dual filter pair (\tilde{g}, \tilde{h}) are denoted by adding a tilde, e.g. \tilde{W} .

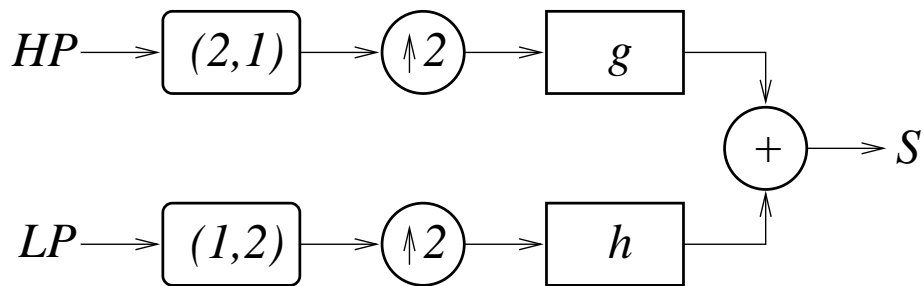
The properties of these transformation matrices depend on the type of wavelet transform (orthogonal vs. biorthogonal) and the type of signal extension (periodic vs. symmetric). We can distinguish the following cases:

5.3 Case 1

This is the generic case, i.e. for biorthogonal wavelets in combination with symmetric extension:



Decomposition



Reconstruction

Figure 11: The filter bank scheme: filter operations are indicated by boxes, extension operators by rounded boxes and up/downsampling or addition operations by circles.

Forward/Inverse FWT	Orthogonality
$W^* \cdot \widetilde{W} = I$	$W^{*T} \perp \widetilde{W}$
$\widetilde{W} \cdot W^* = I$	$\widetilde{W}^T \perp W^*$
$\widetilde{W}^* \cdot W = I$	$\widetilde{W}^{*T} \perp W$
$W \cdot \widetilde{W}^* = I$	$W^T \perp \widetilde{W}^*$

5.4 Case 2

If we use biorthogonal wavelets in combination with periodic extension, we can make the following simplifications:

$$\begin{aligned}
W^T &\equiv W^*, \\
W^{*T} &\equiv W, \\
\widetilde{W}^T &\equiv \widetilde{W}^*, \\
\widetilde{W}^{*T} &\equiv \widetilde{W},
\end{aligned}$$

and we get:

Forward/Inverse FWT	Orthogonality
$W^* \cdot \widetilde{W} = I$	$W \perp \widetilde{W}$
$\widetilde{W} \cdot W^* = I$	$\widetilde{W}^* \perp W^*$
$\widetilde{W}^* \cdot W = I$	
$W \cdot \widetilde{W}^* = I$	

5.5 Case 3

If we use orthogonal wavelets, we can simplify even further:

$$\begin{aligned}
\widetilde{W} &\equiv W, \\
\widetilde{W}^* &\equiv W^*,
\end{aligned}$$

and we get:

Forward/Inverse FWT	Orthogonality
$W^* \cdot W = I$	$W \perp W$
$W \cdot W^* = I$	$W^* \perp W^*$

The *forward/inverse FWT* properties are well-known and used in about any wavelet transform implementation.

However, for the APOD we also have to care about the *orthogonality* properties. Indeed, if we use a biorthogonal wavelet transform or wavelet packet transform, we will end up with *primal* and *dual* eigenmodes (E_i^* and \widetilde{E}_i^*) that need to be biorthogonal to be able to decompose a random snapshot into a precalculated basis of eigenmodes.

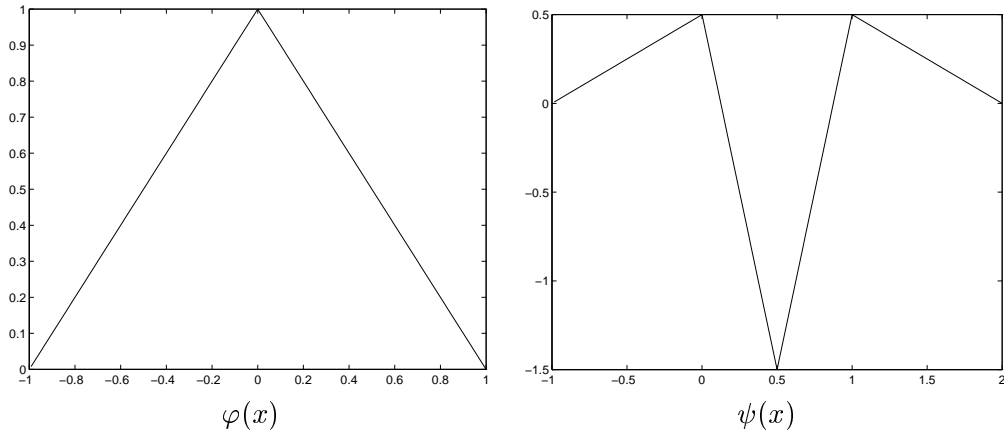


Figure 12: The biorthogonal scaling function and wavelet used in our test process. The dual scaling function and wavelet are not shown because of their irregular nature.

6 Results

We have evaluated the use of the Approximate Proper Orthogonal Decomposition to represent and to analyze the dynamical behavior of the discretization of a system of partial differential equations (PDEs), namely a reaction-diffusion system.

6.1 Wavelets

Because of their symmetry, we have chosen for biorthogonal wavelets. In our test cases, we used the biorthogonal Cohen-Daubechies-Feauveau wavelets (2 vanishing moments for both the primal and the dual wavelet [2]) with symmetric signal extension (fig. 12).

The cost criterium used for choosing the “best basis” among all possible wavelet packet bases is based on the logarithm of the energy

$$\text{cost} = \sum_i \log(x_i^2),$$

which is equivalent with

$$\text{cost} = \sum_i \log |x_i|.$$

6.2 The Dynamical System

We have used the one-dimensional Brusselator reaction-diffusion system [5]:

$$\begin{aligned} \frac{\partial X}{\partial t} &= \frac{D_X}{L^2} \frac{\partial^2 X}{\partial z^2} + X^2 Y - (B + 1)X + A, \\ \frac{\partial Y}{\partial t} &= \frac{D_Y}{L^2} \frac{\partial^2 Y}{\partial z^2} - X^2 Y + BX, \end{aligned}$$

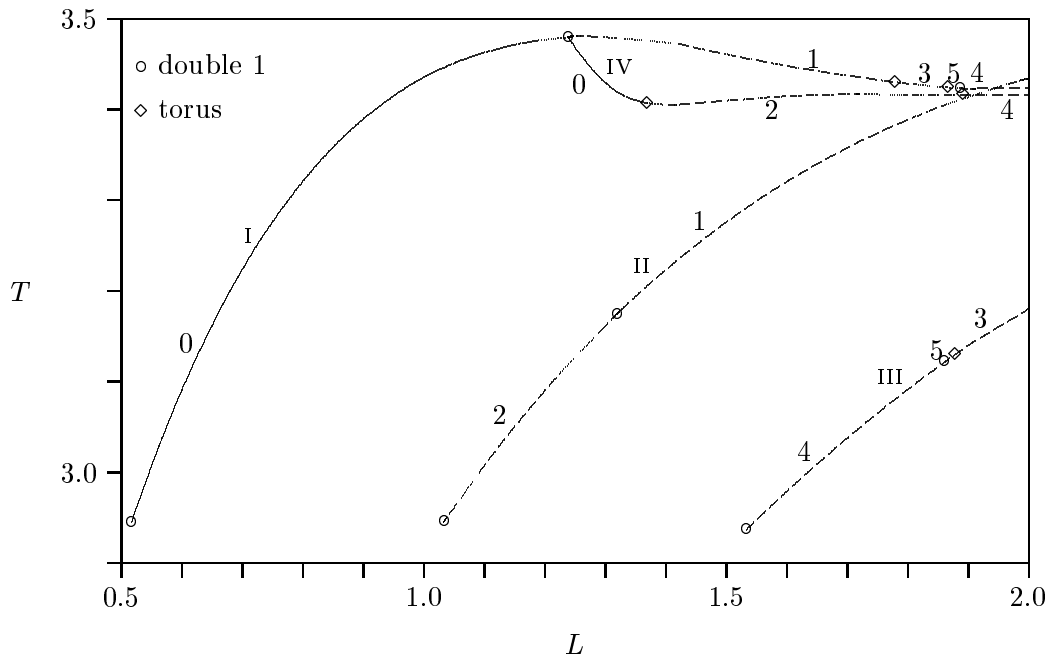


Figure 13: Bifurcation diagram of periodic (branches I–IV) and quasi-periodic (branch V) solutions for the Brusselator model, period T versus the reactor length L [6]. Note that this diagram is calculated using a 31-point discretization and that it is not complete.

with Dirichlet boundary conditions

$$\begin{aligned} X(t, z = 0) &= X(t, z = 1) = A, \\ Y(t, z = 0) &= Y(t, z = 1) = B/A. \end{aligned}$$

The reactor length L is used as a free parameter (continuation parameter), while we keep the other parameters fixed:

$$\begin{aligned} A &= 2.0, \\ B &= 5.45, \\ D_X &= 0.008, \\ D_Y &= 0.004. \end{aligned}$$

Branches of periodic solutions bifurcate from the trivial steady state branch ($X = A, Y = B/A$), as can be seen in the bifurcation diagram in figure 13 [6].

6.3 Snapshots

To create a low-dimensional model of our system, we started with four sets of snapshots, calculated for four values of the continuation parameter L , using time integration with a time step $\Delta t = 0.2$, as shown in table 1.

Set	L	Branch	# snapshots	
A	0.75	I	20	Periodic solution
B	1.25	I	20	Periodic solution (before period-doubling)
C	1.43	V	192	Quasi-periodic solution (torus)
D	1.50	IV	20	Periodic solution
Total			252	

Table 1: The snapshots used for testing and their bifurcation parameter values.



Figure 14: $L = 0.75$: 20 snapshots.

Each snapshot is a vector with 256 elements, containing the concentration X of the first reagens. Figures 14 through 17 give a graphical representation of the concentration X for the different parameter values. White indicates a high concentration, black a low concentration. The z (spatial) axis runs horizontally, while the t (time) axis runs vertically (time runs from the top to the bottom).

6.4 “Strange” Snapshots

To test how well this low-dimensional model can represent the dynamical behavior of the parameter dependent Brusselator model, we decomposed a set of “strange” snapshots — snapshots not belonging to the set of snapshots we used to construct the APOD model — in the APOD basis. The set consists of 100 snapshots for $L = 1.40$, obtained by time integration with a time step $\Delta t = 0.2$ starting from the perturbed trivial solution. The “strange” snapshots show a periodic solution including the transient behavior.

6.5 Eigenvalue Distribution

Since the snapshots were normalized by subtracting their mean and by rescaling them (see 1), the eigenvalues are equal to the fraction of captured energy.

To approximate the snapshots by using only a few eigenmodes, we need to keep the eigenmodes which capture the most energy and hence have the largest eigenvalues.



Figure 15: $L = 1.25$: 20 snapshots.

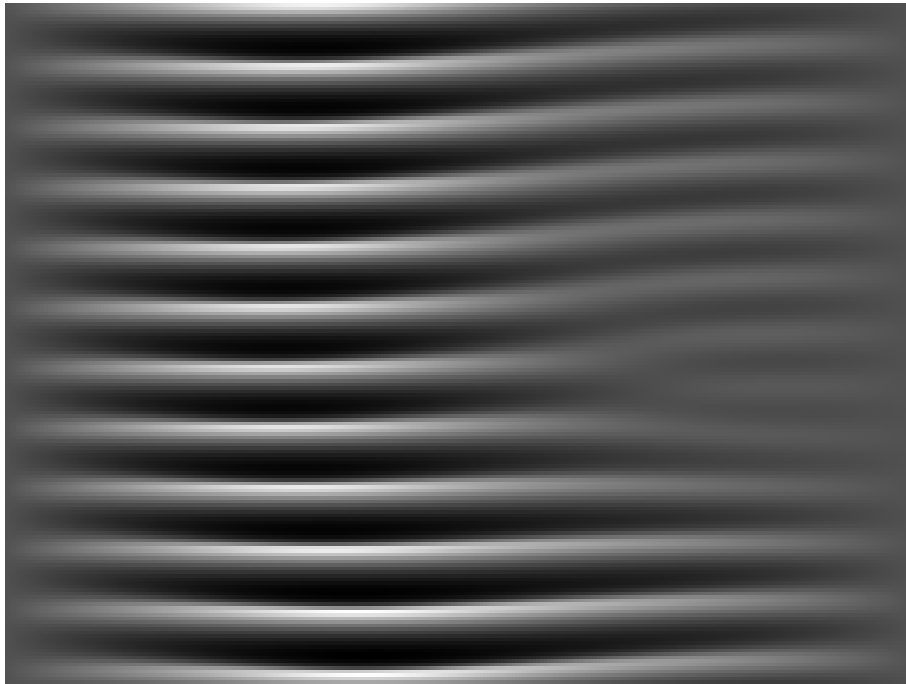


Figure 16: $L = 1.43$: 192 snapshots.



Figure 17: $L = 1.50$: 20 snapshots

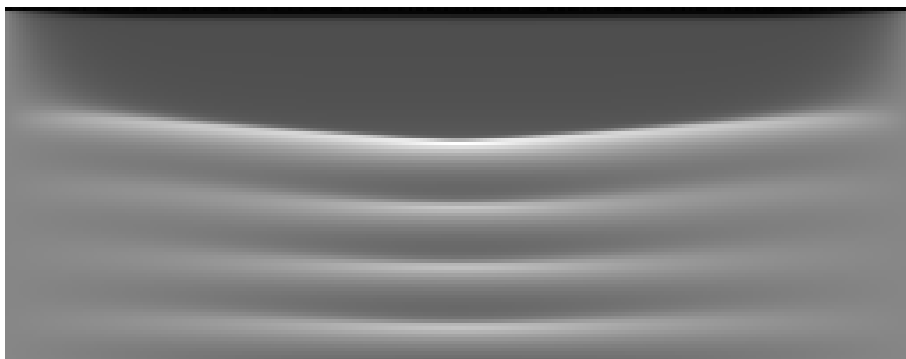


Figure 18: $L = 1.40$: 100 “strange” snapshots.

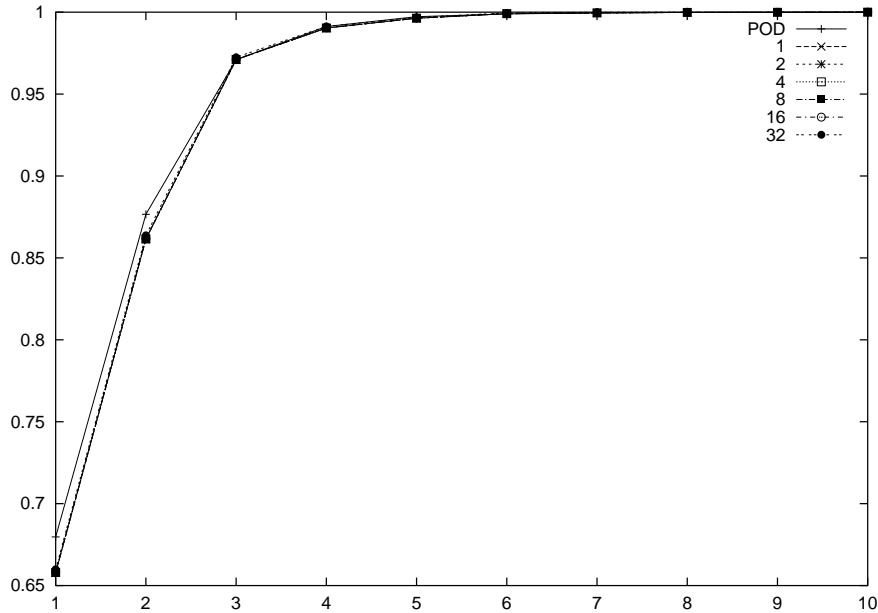


Figure 19: Sum of the eigenvalues vs. number of eigenvalues, both for the POD and the APOD with different compression rates.

In fig. 19 the sum of the largest i eigenvalues is plotted for different compression rates. The plot reveals that e.g. the 4 most important eigenmodes capture more than 99% of the energy present in the set of snapshots.

Observe that for a fixed number of eigenmodes the captured energy is increasing in function of the compression ratio. Due to the lossy compression some small non-correlated information is lost, and the resulting data is thus more correlated, resulting in the effect that the same number of eigenvalues capture more energy.

Note that in the case of compression rate 32, we retain only 8 eigenmodes because the dimension of the compressed system can not be higher due to the large compression ratio ($256/32 = 8$).

The distribution of the eigenvalues is a measure for the decorrelation achieved by the (approximate) POD. If many eigenvalues are very small, the original set of snapshots was highly correlated and can be represented well by only a few eigenmodes. In contrast, if all eigenvalues have about the same magnitude, the set of snapshots was not correlated at all.

6.6 Eigenmodes

The 6 most important eigenmodes, both for the POD and the Approximate POD with different compression rates, are shown in figure 20.

The difference between the results for the POD and the Approximate POD using compression rate 1 (i.e. no compression) is due to the biorthogonality of the wavelet packet basis.

Number of eigenmodes	POD	APOD with compression rate					
		1	2	4	8	16	32
1	67.97	68.11	68.11	68.11	68.11	68.10	67.97
2	87.66	87.69	87.69	87.69	87.69	87.64	87.04
3	97.10	97.17	97.17	97.17	97.17	97.17	96.83
4	99.13	99.10	99.10	99.10	99.09	99.07	98.50
5	99.72	99.68	99.68	99.68	99.68	99.65	99.04
6	99.89	99.90	99.90	99.90	99.89	99.86	99.23
7	99.95	99.94	99.94	99.94	99.93	99.90	99.24
8	99.99	99.98	99.98	99.98	99.97	99.93	99.25
9	99.99	99.99	99.99	99.99	99.98	99.94	
10	100.00	99.99	99.99	99.99	99.99	99.95	

Table 2: Mean percentage of energy after reconstruction of the snapshots vs. number of used eigenmodes for different compression rates.

If we would have used an orthogonal wavelet packet basis, there would be no difference.

6.7 Reconstruction of the Snapshots

If we reconstruct the snapshots (both the set used for the construction of the APOD model and the “strange” set) using only the most important eigenmodes, we get the following results. Tables 2 and 3 show the mean percentage of captured energy (over all reconstructed snapshots) after reconstruction by using the most important eigenmodes for different compression ratios.

6.8 Coefficients of the Snapshots in the APOD Basis

We can analyze the behaviour of the dynamical system by analyzing the coefficients of the projection of the snapshots onto the APOD basis. Figure 21 shows these coefficients for the two most important eigenmodes. The first 20 snapshots are from set *A*, the next 20 from set *B* and so on (cfr. table 1). The periodicity of the solutions is clearly visible. There is no visible difference among the coefficients for the APOD bases with different compression rates.

Figure 22 shows how the maximum error in the coefficients of the 10 most important eigenmodes depends on the compression rate.

The same results for the decomposition of the “strange” snapshots in the APOD basis are shown in the figures 23 and 24.

6.9 Timings

All timings were obtained on one node of an IBM SP2 cluster of RS/6000 POWER2 thin nodes, with an implementation in C++ using LAPACK++.

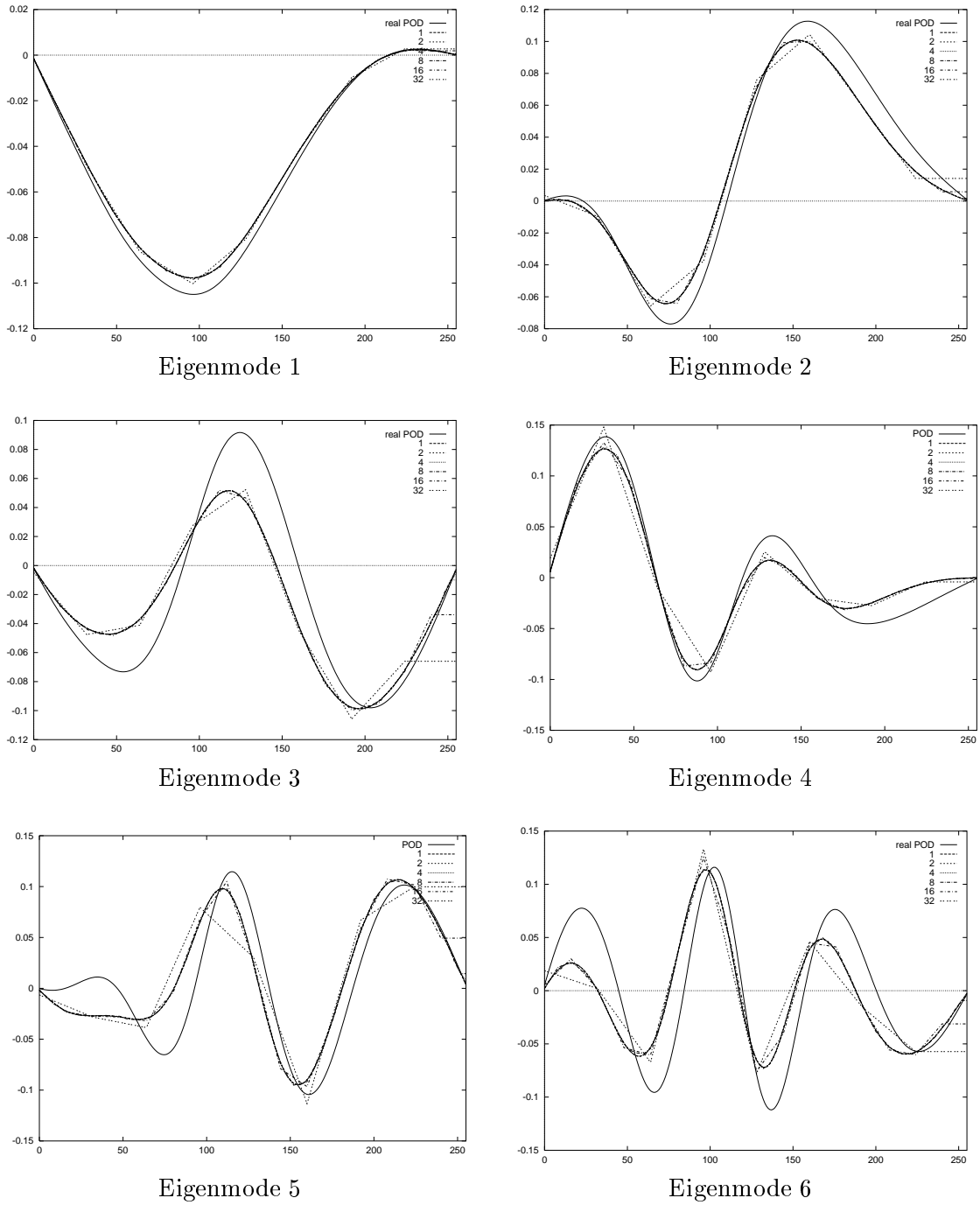
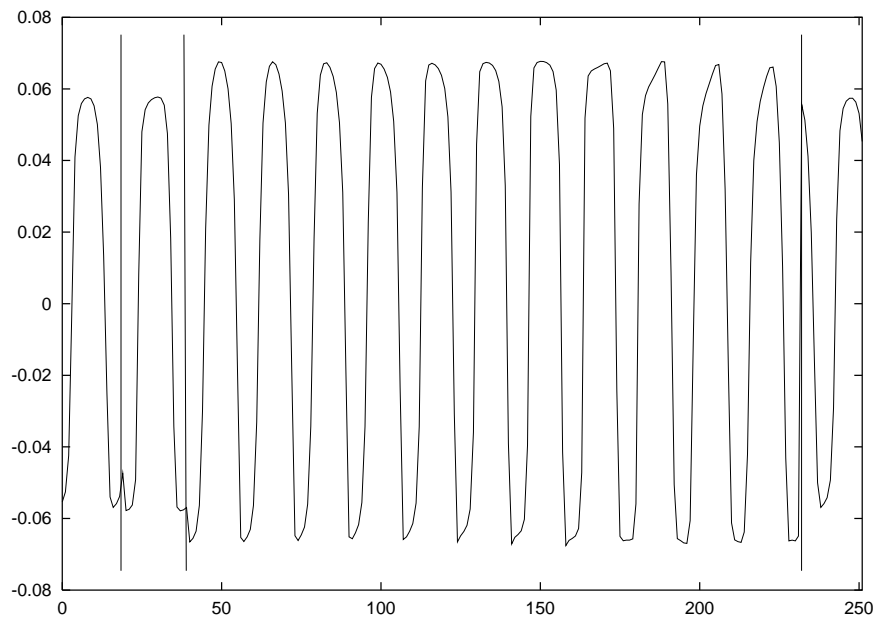
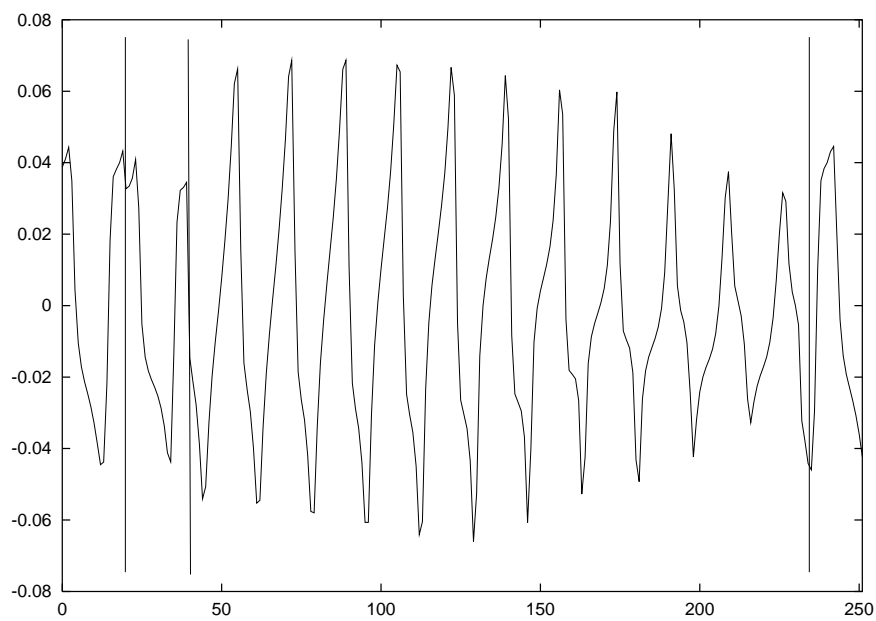


Figure 20: The 6 most important eigenmodes for the POD and for the APOD with different compression rates.



Coefficient 1



Coefficient 2

Figure 21: First two coefficients of the decomposition of the snapshots into the best basis.

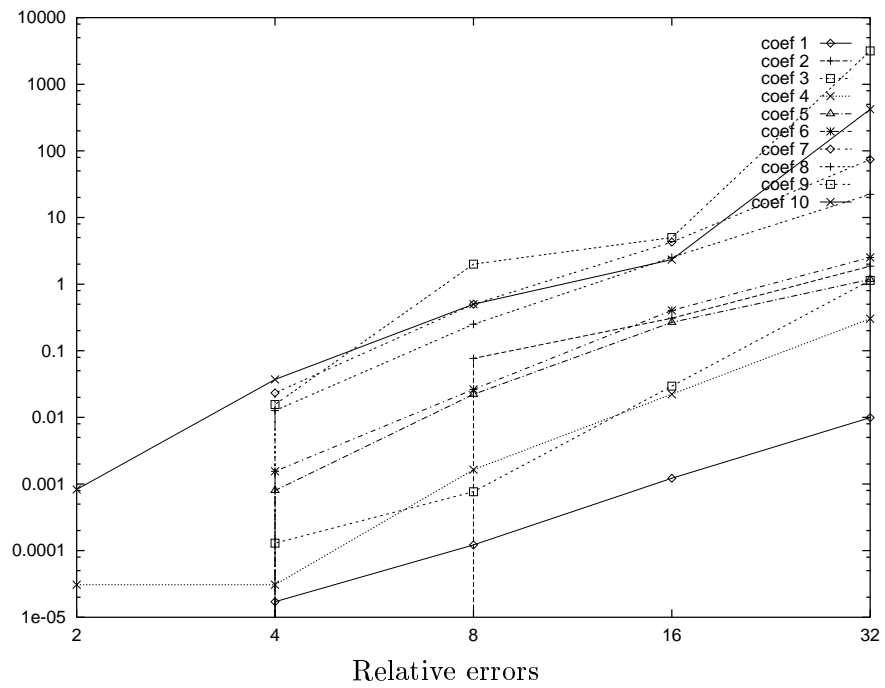
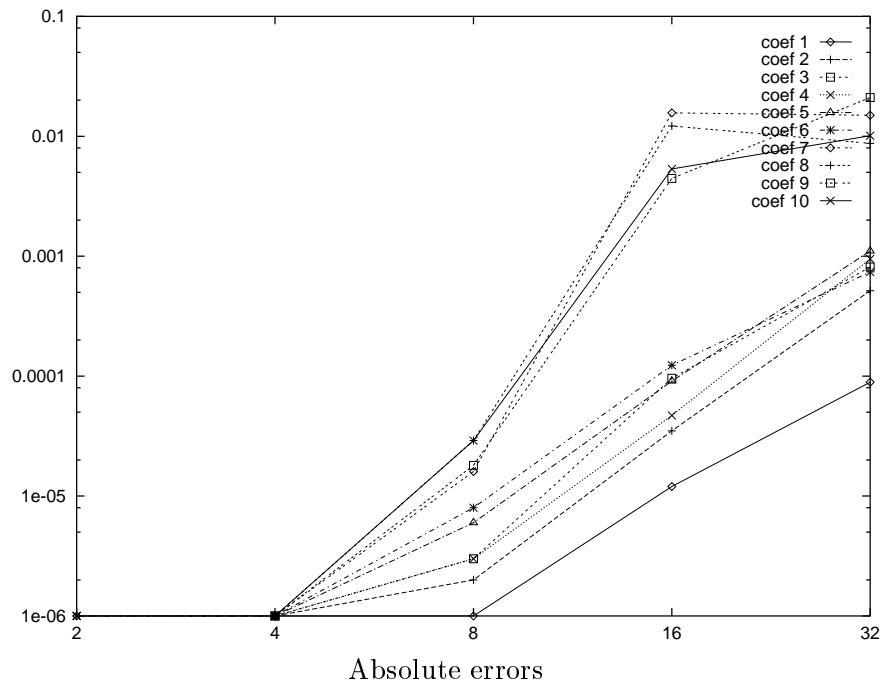


Figure 22: Maximum error in the coefficients (compared to the case without compression) for different compression rates.

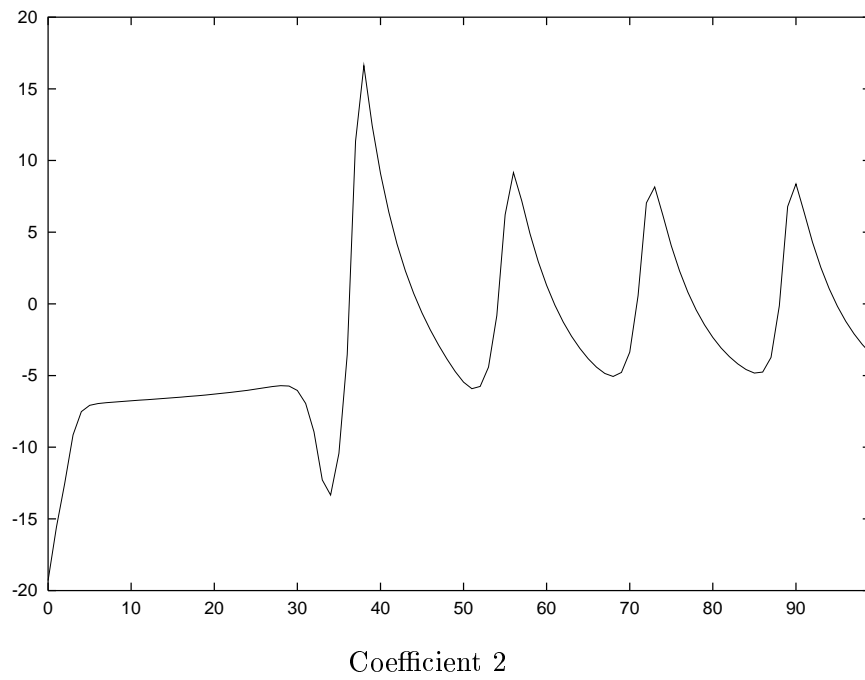
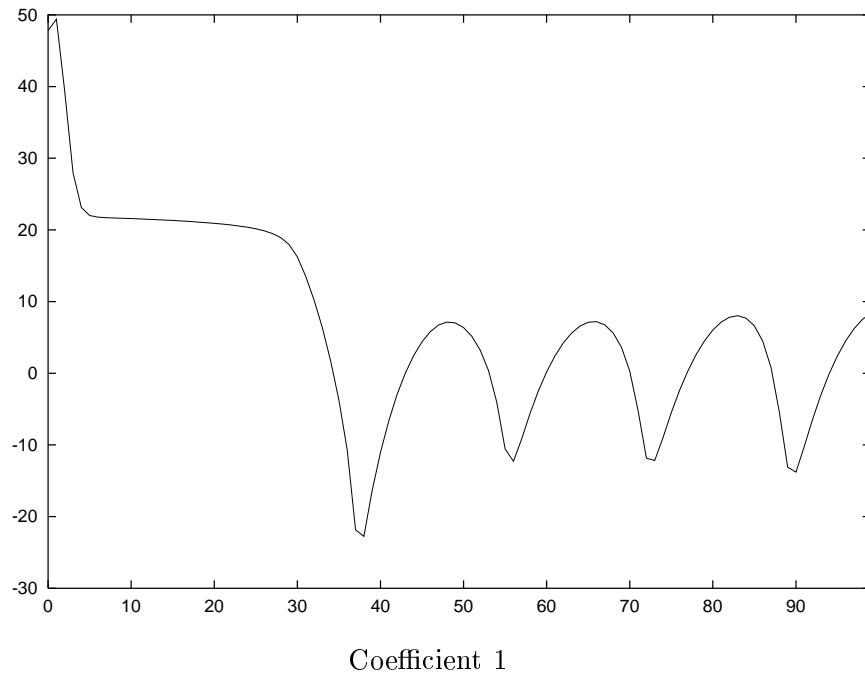


Figure 23: First two coefficients of the decomposition of the “strange” snapshots into the best basis.

Number of eigenmodes	APOD with compression rate					
	1	2	4	8	16	32
1	52.87	52.87	52.87	52.87	52.87	52.81
2	71.62	71.62	71.62	71.62	71.60	71.34
3	91.75	91.75	91.75	91.74	91.79	91.90
4	96.87	96.87	96.87	96.87	96.90	96.84
5	97.40	97.40	97.40	97.39	97.43	97.52
6	98.22	98.22	98.22	98.21	98.23	98.32
7	99.12	99.12	99.12	99.10	99.16	99.90
8	99.09	99.09	99.09	99.07	99.11	99.88
9	99.32	99.32	99.32	99.30	99.37	
10	99.43	99.43	99.43	99.40	99.47	

Table 3: Mean percentage of energy after reconstruction of the “strange” snapshots vs. number of used eigenmodes for different compression rates.

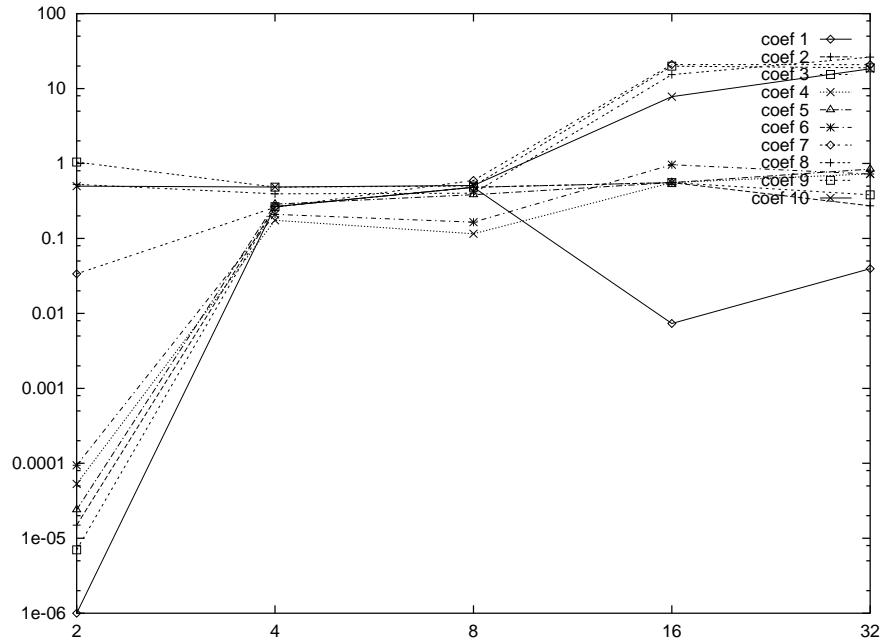


Figure 24: Maximum absolute error in the coefficients (compared to the case without compression) for different compression rates.

Compression rate	Calculation time	
1	45.30	$\approx 25+17.4$
2	37.30	$\approx 25+9$
4	31.15	
8	27.45	
16	25.62	
32	25.40	$\approx 25+0.5$

Table 4: Computation times (in seconds) for the Approximate Proper Orthogonal Decomposition for the various compression ratios. The real POD required ca. 17.40 seconds. The wavelet packet transform took ca. 25 seconds. The third column shows that the total cost of the APOD is approximately equal to the sum of the cost of the wavelet packet transform (ca. 25 seconds) and the cost of the construction of the matrix C^* for the POD of the reduced system (proportional to the reciprocal of the compression ratio).

Method	Complexity
POD	$\mathcal{O}(NS^2) + \mathcal{O}(S^3)$
FWPT	$\mathcal{O}(SN \log N)$
APOD	$\mathcal{O}(SN \log N) + \mathcal{O}(S^2N/C) + \mathcal{O}(S^3)$

Table 5: Comparison of the complexity of the various methods.

The normalization of the snapshots took 0.32 seconds. The decomposition timings for the APOD are shown in table 4. As a comparison, the real POD required 17.40 seconds. Note that the wavelet packet transform routines used — taking ca. 25 seconds — were very generic and were not optimized at all. A cautious implementation (using the Lifting Scheme algorithm) will speed it up an order of magnitude.

A comparison of the complexity of the various methods is shown in table 5. However, the factor $\log N$ is only valid for the one-dimensional case we considered here. For higher dimensions it becomes

$$\log \sqrt[d]{N},$$

with d the dimension of the snapshots and thus $\sqrt[d]{N}$ the number of data points per dimension. This factor decreases quickly for increasing dimensions, so for higher dimensions we eventually win using the APOD.

7 Conclusions

The Approximate Proper Orthogonal Decomposition using biorthogonal wavelet packets allows a good reconstruction of the original data. We only need very few eigenmodes (basis functions), while methods based on other approximations like DCT (Discrete Cosine Transform) and FWT (Fast Wavelet Transform) require much more basis functions.

The decomposition can be calculated in both the original domain and in the wavelet packet domain. The eigenmodes are not localized, but the underlying wavelet packets are, and thus the calculation of inner products in the wavelet packet domain is a cheaper operation.

This is useful for the construction of low-dimensional approximate models for PDEs (reaction-diffusion, Navier-Stokes, ...). A PDE of the form

$$u(x, t) = \sum_{i=0}^{K-1} a_i(t) E_i^*(x)$$

can be transformed into the form

$$\begin{aligned} \frac{da_i(t)}{dt} &= f(a_0, \dots, a_{K-1}) \\ &= c_i + c_{ij} a_j + c_{ijk} a_j a_k + \dots \end{aligned}$$

by means of a Galerkin projection. The Galerkin projection requires $\mathcal{O}(K^3)$ inner products of vectors of size N/C , which can be sped up by using the properties of wavelet packets (localization, biorthogonality).

Another application is the compressed storage of related data. In [8] Sirovich et al. suggest to use the POD to store a library of pictures of human faces by storing the most important POD eigenmodes and, for each face (snapshot), the coefficients in this basis. The APOD allows a further storage reduction by storing the compressed eigenmodes.

8 Acknowledgements

The authors would like to thank Wim Sweldens (Lucent Technologies, Bell Laboratories) and Adhemar Bultheel for suggestions and comments.

This text presents research results of the Belgian programme on Interuniversity Poles of Attraction (IUAP 17), initiated by the Belgian State—Prime Minister’s Service—Federal Office for Scientific, Technical and Cultural Affairs. The scientific responsibility is assumed by its authors.

References

- [1] C. M. Brislawn. Classification of nonexpansive symmetric extension transforms for multirate filter banks. *Appl. Comput. Harmon. Anal.*, 3:337–357, 1996.
- [2] A. Cohen, I. Daubechies, and J. Feauveau. Bi-orthogonal bases of compactly supported wavelets. *Comm. Pure Appl. Math.*, 45:485–560, 1992.
- [3] R.R. Coifman and V. Wickerhauser. Wavelets and adapted waveform analysis. In J.J. Benedetto and M. Frazier, editors, *Wavelets: mathematics and applications*, chapter 10. CRC Press, 1993.
- [4] I. Daubechies. Orthonormal bases of compactly supported wavelets. *Comm. Pure Appl. Math.*, 41:909–996, 1988.

- [5] M. Holodniok, P. Knedlik, and M. Kubiček. Continuation of periodic solutions in parabolic differential equations. In T. Küpper, R. Seydel, and H. Troger, editors, *Bifurcation: Analysis, Algorithms, Applications*, ISNM 79, pages 122–130. Birkhäuser, Basel, 1987.
- [6] K. Lust, D. Roose, A. Spence, and A.R. Champneys. An adaptive Newton-Picard algorithm with subspace iteration for computing periodic solutions. *SIAM J. Sci. Comput.*, 1996. Accepted.
- [7] S. G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Patt. Anal. Mach. Intell.*, 11(7):674–693, 1989.
- [8] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterisation of human faces. *J. Opt. Soc. Amer.*, 4(3):519–524, March 1987.
- [9] W. Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM J. Math. Anal.*, (to appear).
- [10] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Appl. Comput. Harmon. Anal.*, 3(2):186–200, 1996.
- [11] M. V. Wickerhauser. Lectures on wavelet packet algorithms. 1991.
- [12] M.V. Wickerhauser. *Adapted wavelet analysis from theory to software*. A.K. Peters, 289 Linden Street, Wellesley, MA 02181, 1994.