On Second Order Operators and Quadratic Operators[*]

Michael Felsberg
Linköping University, Dept. EE, Computer Vision Laboratory, S-58183 Linköping
mfe@isy.liu.se

Abstract

In pattern recognition, computer vision, and image processing, many approaches are based on second order operators. Well-known examples are second order networks, the 3D structure tensor for motion estimation, and the Harris corner detector. A subset of second order operators are quadratic operators. It is lesser known that every second order operator can be written as a weighted quadratic operator. The contribution of this paper is to propose an algorithm for converting an arbitrary second order operator into a quadratic operator. We apply the method to several examples from image processing and machine learning. The advantages of the alternative implementation by quadratic operators is two-fold: The underlying linear operators allow new insights into the theory of the respective second order operators and replacing second order networks with sums of squares of linear networks reduces significantly the computational burden when the trained network is in operation phase.

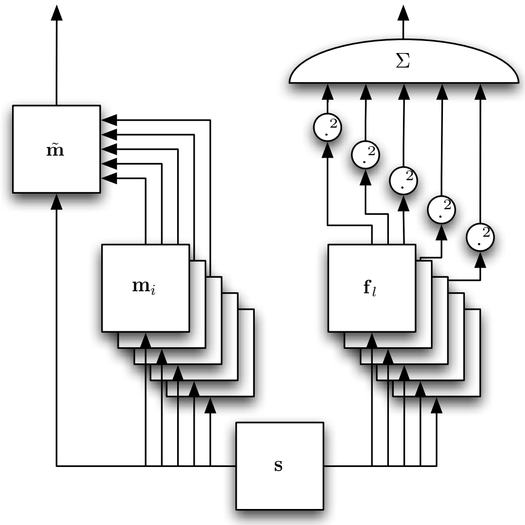

Figure 1. Conversion of second order operator into sums of squared linear operators. $\mathbf{m}_i^T$ are the rows of $\mathbf{M}$ and $\tilde{\mathbf{m}}$ is the vector of scalar products of the rows with the signal $\mathbf{m}_i^T\mathbf{s}$.

1. Introduction

In pattern recognition, computer vision, and image processing, many approaches are based on second order operators. Well-known examples are *second order networks*, a special case of higher order networks (HON) [6] or sigma-pi networks [13], the *3D structure tensor* for motion estimation, e.g. [9, 7], the *Harris corner detector* [8] and the *Förstner operator* [5], and more in general *second order Volterra filtering* [11]. A subset of second order operators are quadratic operators. It is lesser known that every second order operator can be written as a weighted quadratic operator.

The main contribution of this paper is to propose an algorithm for converting an arbitrary second order operator $\mathbf{M}^T = [\mathbf{m}_1\mathbf{m}_2\ldots]$ into a quadratic operator, see Fig. 1. In Sect. 3 we apply the method to examples from image processing and machine learning: The *2D structure tensor* based on Sobel operators and Scharr filters [14], the *Teager-Kaiser energy operator* in 1D [10] and the *energy tensor* in 2D [3], and solving a not linearly separable problem in a quadratic network derived from a second order network [6].

In Sect. 4 we discuss the advantages and insights resulting from the method. We find two main advantages: The underlying linear operators allow *new insights* into the theory of the respective second order operators. Replacing second order networks with sums of squares of linear networks *reduces significantly the computational burden* during the operation phase after training.

[*]The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 215078 DIPLECS and from the CENIIT project CAIRIS.

2. Materials and methods

2.1. Second order operators

In what follows, we only consider finite, discrete signals. To a certain extent, the results generalize to infinite and continuous signals, but for keeping things simple, we focus on the practically more relevant case of finite, discrete signals.

Let $s \in \mathbb{R}(\Omega_N)$ be an N-dimensional discrete signal defined on the compact domain $\Omega_N \subset \mathbb{Z}^N$. By stacking the dimensions of the N-D signal into a single dimension, we represent the signal as a $|\Omega_N|$-D column vector, denoted as $\mathbf{s} \in \mathbb{R}^{|\Omega_N|}$.

The point response of a general second order operator acting on s is defined as

$$\mathscr{O}_{\mathbf{p}}\{\mathbf{s}\} = \mathbf{s}^T\mathbf{M}\mathbf{s} \ , \tag{1}$$

where $\mathbf{M}$ is a $|\Omega_N| \times |\Omega_N|$ matrix.

The central topic of this paper is the following algorithm that converts any finite second order operator into a set of linear operators, such that the sum of squares of the latter can be used instead of the original operator.

2.2. The Conversion Algorithm

We provide an algorithm for computing the set of vectors $\{\mathbf{f}_l \in \mathbb{C}^{|\Omega_N|}\}_{l=1\dots l_{\max}}$, $l_{\max} \leq |\Omega_N|$ for a given matrix $\mathbf{M} \in \mathbb{R}^{|\Omega_N|} \times \mathbb{R}^{|\Omega_N|}$. The first step is to re-write the operator $\mathbf{M}$ as a symmetric operator $\mathbf{A}$:

$$\mathbf{s}^T\mathbf{M}\mathbf{s} = \mathbf{s}^T\frac{\mathbf{M}+\mathbf{M}^T}{2}\mathbf{s} = \mathbf{s}^T\mathbf{A}\mathbf{s} \ . \tag{2}$$

Since $\mathbf{A}$ is real and symmetric, it can be written according to the spectral theorem as

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T \ , \tag{3}$$

where $\mathbf{V} \in O(|\Omega_N|)$ is the matrix of eigenvectors of $\mathbf{A}$ and $\mathbf{\Lambda}$ is the diagonal matrix with the (sorted) eigenvalues λ_l. If $\mathbf{A}$ is of rank r, $\lambda_l = 0$ for all $l > r$, i.e. the number of eigenvalues $l_{\max} = r$ is less or equal to $|\Omega_N|$. Defining

$$\tilde{\mathbf{s}} = \mathbf{V}^T\mathbf{s} \text{ , we obtain} \tag{4}$$

$$\mathbf{s}^T\mathbf{M}\mathbf{s} = \sum_{l=1}^{r}(\sqrt{\lambda_l}\tilde{s}_l)^2 \ . \tag{5}$$

Finally, as (4) is a linear operator, we can rewrite the second order operator (1) as

$$\mathscr{O}_{\mathbf{p}}\{\mathbf{s}\} = \sum_{l=1}^{l_{\max}}(\mathbf{s}^T\mathbf{f}_l)^2 \ , \tag{6}$$

where $\mathbf{f}_l = \mathbf{v}_l\sqrt{\lambda_l}$ and $\mathbf{v}_l$ is the lth column of $\mathbf{V}$.

2.3. Comments

Note that in case of negative eigenvalues, the linear operator gets imaginary coefficients, i.e., we do not consider the norm of the operator responses where the imaginary part is conjugated, but the sum of squares. This distinguishes our approach from other, similar approaches: Appendix A of [1], PCA, and Cholesky decomposition. The proposed method differs from PCA since PCA aims at decorrelating input data dimensions, not operators. If PCA is applied to second order operators, obviously the same eigensystem as in the proposed method is obtained. However, the signed eigenvalues are lost (due to the outer product of the matrix in the PCA), i.e., one cannot replace the original operator with a PCA-modified operator. The Cholesky decomposition cannot be applied to general second order operators, as they are often (as all examples that we will show) not positive definite, i.e., the Cholesky decomposition does not exist. Similarly, [1] only discusses the case of positive definite matrices.

3. Results

The algorithm from Section 2.2 can be used for any second order operator. In this paper, we however focus on image processing operators and second order neural networks.

3.1. Structure tensor

The first example we consider is the structure tensor, which is the base for the Förstner operator [5] and the Harris detector [8]. The structure tensor is computed by averaging outer products of gradient estimates $\mathbf{d}$:

$$\mathbf{T} = \sum_{\mathbf{x}\in\mathcal{N}} w(\mathbf{x})\mathbf{d}(\mathbf{x})\mathbf{d}^T(\mathbf{x}) \ , \tag{7}$$

where $\mathbf{d}$ is obtained by, e.g. Sobel operators (see e.g. Sect. 12.7 in [9]) or Scharr filters [14, 9]. The diagonal elements of the outer product are quadratic terms anyway, so the only remaining non-quadratic term is the off-diagonal element d_1d_2 which is a special case for M of rank 2. In case of the (not normalized) Sobel filter, the spectral decomposition (6) of the horizontal and the vertical Sobel filter results in the two filters

$$f_1 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix} \quad f_2 = \begin{bmatrix} 0 & i & i \\ -i & 0 & i \\ -i & -i & 0 \end{bmatrix} \ .$$

As it turns out, these filters are just the derivatives in the diagonal directions. This is an interesting result, as this observation relates the gradient-based approach of the structure tensor to the quadrature filter based method to estimate the orientation tensor [7]. In the latter case, directed quadrature filters are computed along the coordinate axes and the diagonals and their magnitudes are used to compute the orientation tensor.

If we apply our method to the Scharr filters instead of the Sobel operator, the filter masks are

$$g = \begin{bmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{bmatrix} \quad h = \begin{bmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{bmatrix}$$

and we obtain with (6) integer valued kernels:

$$f_1 = \begin{bmatrix} -3 & -5 & 0 \\ -5 & 0 & 5 \\ 0 & 5 & 3 \end{bmatrix} \quad f_2 = \begin{bmatrix} 0 & -5\,i & -3\,i \\ 5\,i & 0 & -5\,i \\ 3\,i & 5\,i & 0 \end{bmatrix} .$$

3.2. Energy operators

The classical algorithm to computer the Teager-Kaiser energy operator reads [10]

$$\Psi[s(t)] = \dot{s}(t)^2 - s(t)\ddot{s}(t) \approx s(t)^2 - s(t-1)s(t+1) \ .$$

In the derivation of this operator, the squared gradient is approximated numerically and applying the algorithm from Sect. 2.2 it turns out that

$$\begin{aligned} \dot{s}(t)^2 &\approx (s(t) - s(t-1))(s(t+1) - s(t)) \\ &= (s(t+1) - s(t-1))^2 \\ &\quad -(s(t+1) - 2s(t) + s(t-1))^2 \\ &\approx \dot{s}(t)^2 - \ddot{s}(t)^2 \ , \end{aligned}$$

i.e., the approximation using left- and right differences becomes an approximation using the centered difference and the Laplacian.

The energy tensor has been introduced in [3] as a generalization of the energy operator, and opposed to [12] it also contains rotation information. In [4] a fast algorithm for the computation of the ET was proposed: a 2D Teager algorithm. In the computation, five products of linear responses are required, four of them non-quadratic:

$$g_1 = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix} \qquad h_1 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

$$g_2 = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad h_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

and with (6)

$$f_{11} = \frac{1}{2}\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \qquad f_{12} = -\frac{1}{2}\begin{bmatrix} i & 0 & i \end{bmatrix}$$

$$f_{21} = \frac{1}{2}\begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad f_{22} = -\frac{1}{2}\begin{bmatrix} i & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & i \end{bmatrix} .$$

The remaining operator g_3, h_3, g_4, h_4 are rotated versions of g_1, h_1, g_2, h_2 and so are f_{31}, f_{32}, f_{41}, f_{42} in relation to f_{11}, f_{12}, f_{21}, f_{22}.

It turns out that the new operators are mutually orthogonal, which simplifies the analysis of the energy tensor being negative definite or semi definite, see also [2]. Critical cases occur whenever the sum of neighbors in one direction is larger than the absolute difference of the neighbors. For 1D signals this problem is reduced by the zero DC assumption. For 2D signals with zero DC component, negative eigenvalues of the energy tensor occur if the signal is hyperbolic.

3.3. Second order networks

In this section, we apply the algorithm from Sect. 2.2 to machine learning problems. In particular, we consider higher order networks (HON) with order two in this section. According to Bishop [1], second order networks with single output variable can be written as

$$a = \mathbf{z}^T \mathbf{M} \mathbf{z} \ , \tag{8}$$

where $\mathbf{z}^T = [\mathbf{s},\, 1]$ contains the input dimensions and a constant coefficient.

These networks are useful if the dimensions of the input space are not statistically independent or the decision boundary is not monotonic. In order to learn a mapping of dependent variables, the outer product space of the variables need to be generated, i.e., we consider the joint distribution instead of the marginals only.

With the algorithm in Sect. 2.2 we can convert any second order network into a sum of squared linear combinations according to (6). The sum of squared linear combinations is a special case of projection pursuit regression, which reads [1]

$$a = \sum_j w_j \phi_j(\mathbf{f}_j^T \mathbf{z}) + w_0 \ . \tag{9}$$

where in our case $w_0 = 0$, $w_j = 1$ (or $w_j = \pm 1$ if all linear factors $\mathbf{f}_j$ shall be real), and $\phi_j(x) = x^2$. In a biological system, the real-valued linear operators $\mathbf{f}_j$ would correspond to excitatory cells, the imaginary linear operators (or alternatively the negative w_j) would correspond to inhibitory cells. A simple classification example with non-monotonic decision boundary is given in Fig. 2. The coordinate vector are embedded in homogenous coordinates ($\mathbf{z} = [s_1,\, s_2,\, 1]$) and a quadratic network is trained by using the pseudoinverse of the outer product of the indata. The conversion algorithm gives

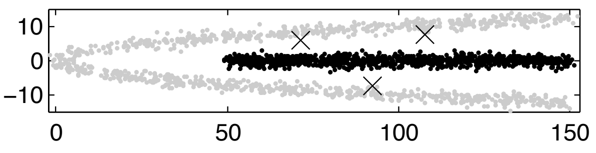

Figure 2. Experiment with two clusters of data (1000 samples each), separated using the sum of squares of three linear projections. Class 1: dark dots (all correct), class 2: bright dots (correct) and dark crosses (incorrect, 3 samples here).

according to (6) (and pruning values less than 10^{-3})

$$\begin{aligned}\mathbf{f}_1 &= \left[\begin{array}{ccc} 0.0110\,i & 0 & 0 \end{array}\right]^T \\ \mathbf{f}_2 &= \left[\begin{array}{ccc} 0 & -0.1312 & 0 \end{array}\right]^T \\ \mathbf{f}_3 &= \left[\begin{array}{ccc} -0.0156 & 0 & 1.0767 \end{array}\right]^T \ .\end{aligned}$$

Hence, the second order network has been converted into a projection pursuit regression, which is evaluated on a different test set in Fig. 2. Repeating the evaluation with $2 \cdot 10^7$ samples results in 99.9% correct classifications. The classification performance does not change under the transformation (6), but the computational load decreases significantly if the mapping is rank-deficit (not the case in the present example). The computational complexity is reduced from $2|\Omega_N|^2$ flops to $2|\Omega_N|l_{max}$ flops.

4. Discussion

In the present paper, we have shown that any kind of second order operator can be transformed into a quadratic operator. Examples from signal processing, image processing, and machine learning have been generated. The advantages of the method are twofold: The underlying linear operators allow new insights into the theory of the respective second order operators and replacing second order networks with sums of squares of linear networks reduces significantly the computational burden during operation mode.

The first advantage became fairly obvious in this paper: Novel insights into 2D gradient operators and energy operators have been gained. The second advantage concerning the field of machine learning will appear even more convincing when considering larger scale problems. The reformulation in terms of squares of linear network outputs allows to interprete the processing as a projection onto prototypes which can be located sparsely in space. In contrast to the ordinary second order network, only a subset of the whole space is considered and needs to be visited.

5. Acknowledgement

We acknowledge discussions with Josef Bigün and Klas Nordberg on the topics of this paper.

References

[1] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, 1995.

[2] A. C. Bovik and P. Maragos. Conditions for positivity of an energy operator. *IEEE Transactions on Signal Processing*, 42(2):469–471, 1994.

[3] M. Felsberg and G. Granlund. POI detection using channel clustering and the 2D energy tensor. In *26. DAGM Symposium Mustererkennung, Tübingen*, 2004.

[4] M. Felsberg and E. Jonsson. Energy tensors: Quadratic phase invariant image operators. In *DAGM 2005*, volume 3663 of *LNCS*, pages 493–500. Springer, 2005.

[5] W. Förstner and E. Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *ISPRS Intercommission Workshop, Interlaken*, pages 149–155, June 1987.

[6] C. L. Giles and T. Maxwell. Learning, invariance, and generalization in higher-order neural networks. *Applied Optics*, 26(23):4972–4978, 1987.

[7] G. H. Granlund and H. Knutsson. *Signal Processing for Computer Vision*. Kluwer Academic Publishers, Dordrecht, 1995.

[8] C. G. Harris and M. Stephens. A combined corner and edge detector. In *4th Alvey Vision Conference*, pages 147–151, 1988.

[9] B. Jähne. *Digital Image Processing*. Springer, Berlin, 6th edition, 2005.

[10] J. F. Kaiser. On a simple algorithm to calculate the 'energy' of a signal. In *Proc. IEEE Int'l. Conf. Acoust., Speech, Signal Processing*, pages 381–384, Albuquerque, New Mexico, 1990.

[11] T. Koh and E. Powers. Second-order Volterra filtering and its application to nonlinear system identification. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(6):1445–1455, December 1985.

[12] P. Maragos, A. C. Bovik, and J. F. Quartieri. A multidimensional energy operator for image processing. In *SPIE Conference on Visual Communications and Image Processing*, pages 177–186, Boston, MA, 1992.

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1: Foundations, pages 318–362. MIT Press, 1986.

[14] H. Scharr, S. Körkel, and B. Jähne. Numerische Isotropieoptimierung von FIR-Filtern mittels Querglättung. In *19. DAGM Symposium Mustererkennung, Braunschweig*, pages 367–374, 1997.