Anisotropic Multidimensional Savitzky Golay kernels for Smoothing, Differentiation and Reconstruction

David J Thornley

Abstract

The archetypal Savitzky–Golay convolutional filter matches a polynomial to even-spaced data and uses this to measure smoothed derivatives. We synthesize a scheme in which heterogeneous, anisotropic linearly separable basis functions combine to provide a general smoothing, derivative measurement and reconsruction function for point coulds in multiple dimensions using a linear operator in the form of a convolution kernel. We use a matrix pseudo inverse for examples, but note that QR factorization is more stable when free weighting is introduced.

1 Introduction

In 1964, Savitzky and Golay formulated a one-dimensional convolutional kernel of odd size for application to evenly spaced data which gives a equivalent smoothed sample value or a derivative estimate at the central point [1]. Their key insight was that fitting a polynomial to evenly spaced data and measuring the n^{th} ($n \geq 0$) derivative of the fit can be achieved by the use of a convolutional kernel.

Luo *et al* [6] analyse the frequency response of digital smoothing and derivative filters based on current state of the art Savitkzy-Golay filters in one dimension. The Savitzky Golay approach in one dimension has been extended to include even-sized kernels [5], and measurements taken at arbitrary positions within the kernel [7]. We also see application to systems where the free variable is transformed from an even spaced lattice [8].

The correct choice of order of the polynomials to use in these filters is a compromise between retention of detail (favouring a higher order) and rejection of noise (favouring a lower order). This choice is commonly *ad hoc*, but work to identify appropriate general test statistics for application in an adaptive scheme [15] has allowed tuning of the approach for specific data sources *e.g.* [16].

We also find extensions of the one dimensional approach to two dimensions, using either simplified basis sets [9] analysed in a manner comparable with that of [1], or using orthogonal polynomials which allow the generation of simple closed forms for smoothing in two dimensions [4] which demonstrates superior properties [13][1] and for

[1] In reading Kuo, Wang and Pickup, note that they refer to the elements of the convolutional kernel as

measuring smoothed derivatives [3]. Kuo *et al* [4] explain that their multidimensional approach is cumbersome in more than two dimensions, and while it can be decomposed into a series of one dimensional filters, these results only operate at the centre of an odd sized patch.

Traditional averaging or median filters perform smoothing to the detrement of trends in the data. In one dimension, this means that peaks and troughs are de-emphasized. The Savitzky Golay polynomial filter improves the result by following trends at scales dictated by the order of the polynomial used. When working in more than one dimension, more complex trends in the data become apparent, such as edges in 2D images. It is desirable to smooth such images while retaining salient features such as edges and corners. When using a simple smoothing filter, it is necessary to limit the extent of the support in the direction of the detail we wish to retain.

Freemand and Adelson [14] describe steerable filters in which an arbitrarily oriented filter may be constructed from a finite bases set of filters, and Yang *et al* [12] describe a non-linear filter – later enhanced by Greenburg and Kogan [11] – which orients its kernel according to a novel measure of anisotropy. A filter based on matching a multidimensional spline attempts to circumvent this requirement by incorporating an approximation to the detail into the patch.

The earliest work referenced by Savitzky and Golay is that of Kerawala from 1941 [2] (which itself references Birge and Shea from 1924 regarding the propagation of errors) in which he formulated an accelerated means for calculating the least-squares fit of a polynomial to data in which the "values of the independent variable form an arithmetic series." The advent of the Moore-Penrose pseudo-inverse [18, 19] gives us a conceptually simpler solution to least squares problems [20].

The pseudo-inverse has been applied in solution for a 2D patch in a similar spirit to Savitzky and Golay's original work in a result reported by Krumm on the Web [17]. This uses a polynomial in x^m and y^n up to a maximum total order $m + n \leq k$ for filtering and derivative measurement at the centre of a patch of odd or even size, with the proviso that an even sized patch gives the measure at the centre of the central four points. This adds the use of even-sized kernel patches in two dimensions to the state of the art in two dimensions represented by [3] and [4]. We use the pseudo-inverse in the subset of our formulations which result in integer valued expressions for sake of simplicity and speed, and revert to the powerful QR factorization [21] for our more general filters in which the direct construction of the pseudo-inverse with non-integer values is commonly fraught with rounding and stability issues.

The present work began with examining the problem of mapping polynomials onto DNA sequencing trace data for the purpose of peak detection by measuring the second derivative, then broadening that interest to include image filtering and edge detection, and considering application to volumetric scans and time series as found in heart scans. This led to the observation that in extending the one dimensional case to two dimensions by – and we believe this is the first occasion on which it has been stated thus – allowing the parameters of the polynomial in any given dimension themselves to vary as polynomials in each additional dimension. This process can be continued to any

"weighting factors", which may be misleading with respect to additional functionality exposed in Meer and Weiss, and taken to a logical conclusion here. We distinguish between kernel elements and weighting applied to data points to modulate their significance in the solution for those kernel elements.

number of dimensions, simply resulting in a product of the polynomials in each dimension. We demonstrate the derivation in one, two and three dimensions, and briefly discuss how this is extended to include, for example, time, or additional annotations. The work is empirical in nature, relying on the extensive analyses of the response of polynomial models to describe sampled data in the existing literature to motivate extension to higher dimensions.

1.1 Polynomial fitting procedure

It has been amply demonstrated in the literature that any regular data lattice may be mapped onto one with unit spacing with no loss of generality when performing least squares filtering (even when the data actually lies on axes subject to simple transformations from even spacing [8]). We therefore bypass discussion of the mapping process and assume all indexed values of free variables lie on a unit lattice.

We begin with a one-dimensional filter. We define the order n polynomial model function $f(x)$ as follows:

$$f(x) := \sum_{p=0}^{n} a_p x^p \tag{1}$$

The coefficients a_i of the polynomial terms are to be identified as part of the least-squares sense fitting of the model to the data points. Let the x coordinate of the i^{th} data point be x_i, $0 \leq i \leq s-1$ and $x_0 = 0$, and its value be g_i. We choose to select $x_0 = 0$ for simplicity of reference, and to emphasize that it is not necessary to centre the coordinate system as would be the case using existing methods.

The required solution for a_i optimizes the fit of $f(x_i)$ to g_i in the simplified least squares sense[2].

$$\epsilon_i = f(x_i) - g_i \tag{2}$$

$$u = \sum_{i=1}^{s} {\epsilon_i}^2 \tag{3}$$

$$\frac{\partial u}{\partial a_i} = 0; 0 \leq i \leq n \tag{4}$$

[2] *i.e.* measured on constant x_i.

This is subsumed in the well-known pseudo-inverse solution of the following form [20]:

$$f(x_i) = g_i \tag{5}$$

$$B := \begin{pmatrix} 1 & x_1 & \cdots & x_1^p & \cdots & x_1^n \\ 1 & \vdots & & \vdots & & \vdots \\ 1 & x_i & & x_i^p & & x_i^n \\ 1 & \vdots & & \vdots & & \vdots \\ 1 & x_k & & x_k^n & & x_k^n \end{pmatrix} \tag{6}$$

$$a := (a_0 \cdots a_p \cdots a_n)^{\mathrm{T}} \tag{7}$$

$$Ba = g \tag{8}$$

$$\Rightarrow B^{\mathrm{T}}Ba = B^{\mathrm{T}}g \tag{9}$$

$$\Rightarrow a = (B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}}g \tag{10}$$

Here, B is a matrix whose rows provide the terms of the polynomial $f(x)$ which take the coefficients in a, and g is a column vector of the sample points. This matrix is generally not square, as we use more data points than the order of the polynomial in order to over-specify the solution and hence reject noise. It is possible to use a number of points one more than the order of the polynomial, but this commonly results in over-fitting.

We create a square system by premultiplying by the transpose of B, and this allows us to invert the system to give a in terms of the samples g.

The $(B^{\mathrm{T}}B)^{-1}$ term may seem computationally daunting, or at least potentially unstable, but all the elements are integers. This means that the determinant and cofactor matrix can be calculated stably, and the complexity of this calculation is $O(lg(n))$ for each value if we apply a divide and conquer regime. Given the model, we then require the particular value or derivative at a position within the region for which we have data. To provide this, we write the required value in terms of the coefficients a.

$$\frac{\partial^c f}{\partial x^c} = \sum_{i=c}^{n} \frac{i!}{(i-c)!} a_i x^{i-c} \tag{11}$$

The terms on the right hand side are constant multiples of the values in a, so these can be written:

$$\frac{\partial^c f}{\partial x^c} = s(B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}}g \tag{12}$$

$$s = \left(0, \ldots, c!, \frac{(c+1)!}{2}x, \ldots, \frac{i!}{(i-c)!}x^{i-c}, \ldots, \frac{n!}{(n-c)!}x^{n-c}\right) \tag{13}$$

$$\text{let } v = s(B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}} \tag{14}$$

Thus, v is a vector which, when dotted with the vector of samples, gives us the c^{th} derivate at the required x value. This vector therefore corresponds to a Savitzky-Golay kernel when $x = k/2$ and k is odd.

1.2 Additional functionality in one dimension

In one dimension, our simple formulation provides for calculation of the same filters as the original Savitzky Golay formulation, plus off-centre measurments and even sized kernels. We can also weight the data points arbitrarily (with the single proviso that the solution is stable, indicated, for example, by a non-zero $(B^{\mathrm{T}}B)$. We can also exclude certain points by ascribing them zero weight. The same effect is of course achieved by leaving the rows associated with those points out of the calculations. Depending on the size of the required kernel and the range of exclusion patterns required, coding may be simpler (though execution most likely less efficient) to use the expression in its entirety.

As we will see later in section 4, data points may be excluded if it is assumed that their information is unrealiable, or, for example, if a sensor element is defective, and that point provides a constant reading regardless of the underlying process being measured, or perhaps unhelpful noise.

When data are missing from a set we wish to analyse, that analysis may be amenable to modification to be robust to erroneous data. It may be simpler to "reconstruct" the bad data. A more obvious effect is achieved when applying reconstruction to two dimensional image data from a defective sensor. If, for example, we take data from a CCD with stuck pixels, those pixels can be identified and replaced with "smoothed" values which have not been influenced by the erroneous input values as would have been the case with an existing filter. This also allows reconstruction of dense clusters of stuck pixels if we use a suitable polynomial order, which is necessarily lower with an increase in dead pixel count.

2 Two dimensions

In two dimensions, expressions appear in the literature which provide filters responding to an uninterrupted regular rectangular lattice of data points. These give measurements at the centre of a square patch of odd size with a the option for approximated Gaussian weighting on the data points, and at the centre of a square patch of even size with constant weighting using (what we consider to be) malformed polynomial [17].

The missing functionality in two dimensions is characterized by; a demand that the measurement be at the centre, symmetry of weighting, a rectangular grid and inclusion of all grid points. The closest to providing a complete solution is Krumm, so we consider the formulation on his web page[3]. Krumm indicates a polynomial model of the following form:

$$f(x,y) := \sum_{p=0}^{n} \sum_{q=0}^{n-p} a_{p,q} x^p y^q \tag{15}$$

Where n is the maximum sum of the powers of each coordinate. This expression does

[3]We regret having to reference a website which can not be attributed any permanency, thus we cannot assume the content will not be changed. The content we respond to here is that seen on the page referred to as of July 2006

not conform with our notion of the filter allowing the descriptive polynomial to vary as an independently described polynomial in each variable.

Taking our one-dimensional expression and extending to two dimensions by allowing each parameter to vary as a polynomial, we have:

$$f(x,y) := \sum_{p=0}^{n_x} \sum_{q=0}^{n_y} a_{p,q} x^p y^q \tag{16}$$

The only difference between this and Krumm's formulation is in the limit of the second sum, and the relaxation of the requirement for equal polynomial order in the two dimensions.

To take derivatives, we operate on the coefficients of the polynomial model in a similar manner to the one dimensional case. Generally, we only take a derivative with respect to one of the free variables. We provide here an expression for the the m^{th} derivative with respect to x, which is trivially transformed in the expression for derivatives with respect to y by exchaning x for y:

$$\frac{\partial^m}{\partial x^m} = \sum_{i=m}^{n_x} \sum_{0}^{n_y} \frac{i!}{(i-m)!} a_{i,j} x^{i-m} y^j \tag{17}$$

and the construction of the kernel follows the same form as in one dimension given the matrix B in expression 12. The direction of greatest slope

2.1 Rotation of the basis set

For a filter which responds to directionality of detail more than is inherent in the freedom of deformation of an isotropic patch, we can choose to orient the axes of the polynomial model in a manner which allows us to reduce the order of the polynomial in directions which encode less detail. For example, in an image (2D set of regularly spaced itensity measurements) of a fingerprint – *c.f.* Greenberg and Kogan's second figure [11] – we see some regions well described by parallel stripes of different intensity. We may model the image with a low order polynomial along the stripes, and a high order polynomial perpendicular to the stripes to capture the edges (strong gradients or transitions).

If we consider the same scenario as figure 1 in Greenberg and Kogan[4], we can align our axes at θ to the x-axis by applying a transformation to the coordinates of the data points we use. This transformation is of course a rotation of $-\theta$ (minus theta). It might be tempting To extend the comparison with [11] to suggest using a polynomial of order proportional to the inverse of the radius of the elipse in the transformed axes. This would however be confusing support range with level of detail.

To perform rotation of the coordinate system, we simply apply a transformation to the coordinates of each data point when calculating the rows of B. Note that this in general results in non-integer entries in the B matrix, so QR factorization is indicated.

[4]This figure depicts an ellipse of unequal radii rotated by $\theta < \pi/2$ counterclockwise from the x-axis, the larger radius falling in the first quadrant.

While in general axis system transformations we commonly rotate and translate, in this case translation is not strictly necessary. Circumstances motivating translation would include a requirement for limiting the absolute magnitude of the particular values of the various order terms of the polynomial model calculated for the B matrix.

To measure derivatives, it is simplest to take a measurement within the transformed coordinate system, then transform back as required. Note, however, that in general, if we have elected to respond to image detail by aligning the filter, we might sensibly expect to be most interested in the derivatives parallel and normal to axes (or planes in higher dimension systems). This removes the need for transformation back into the data coordinate system. If we require the direction of a maximum of a derivative, then we calculate the derivative along each transformed axis, then apply the inverse of the transformation to that vector to obtain the result we require in the data coordinate system.

2.2 Non-rectangular lattice

Our general view of the system allows the the filtering of a regular lattice of alternative shape. The simplest would be composed of triangles rather than rectangles. Use in this manner relates perhaps most usefully to finite element or difference representations of systems. The simplest implementation involves mapping the triangular lattice onto a pair of orthogonal variables. Perhaps surprisingly, this does not lead to a requirement for a non-integer B matrix in 8.

We may index a point on the lattice i along the base direction with vector $(1, 0)$ and j up a side of an equilateral triangle at vector $(cos(2\pi/3), sin(2\pi/3)$. This can be transformed without loss of generality to vectors $(2, 0)$ and $(1, 1)$. This retains spatial ordering, and is achieved by a linear mapping in x and y, so the relationship between each point and the solution will be identical. This also preserves the integer nature of matrix B if we do not apply general weighting (see section 4).

The triangular lattice is simple to conceptualize, but we can use our formulation for any shape of array. There is a tacit assumption in the literature on polynomial filters that the system should repeat over the space of interest for consecutive points, but this is a restriction on the application, not the filter. We could draw attention to the nature of, for example, the edge detection layer in the eye. The sources of data are not regular, but they are immutable, so a static filter is effective, in which the weights have been learned so as to provide a suitable result. In our case, we calculate the filter directly based on knowledge of the positions of the data sources[5].

3 Three dimensional filter

To generate a three-dimensional filter we define a polynomial basis function:

$$f(x, y, z) := \sum_{p=0}^{n_x} \sum_{q=0}^{n_y} \sum_{r=0}^{n_z} a_{p,q,r} x^p y^q z^r \tag{18}$$

[5] Although, of course, a system which infers data positions based on, for example, a measure of continuity gleaneed using a polynomial filter would be interesting.

Where n_x is the order of the polynomial used to model parallel to the x axis, and analagously for n_y and n_z. The matrix B for solving this system has $(i+1)(j+1)(k+1)$ columns and $s_x s_y s_z$ rows, where s_x for example is the size of the data patch along the x axis. In constructing B, we place the value $x^i y^j z^k$ for data point (p,q,r) in column $\tau(x,y,z)$ and row $\pi(p,q,r)$, where τ and π are lexical mappings of the form:

$$\lambda(b_j, b_{j+1}, \ldots, b_k) := b_j + m_j \lambda(b_{j+1}, \ldots, b_k) \tag{19}$$

$$\lambda(b_k) := b_k \tag{20}$$

$$0 \leq b_i < m_j \tag{21}$$

We then manipulate the system of expression 8 with the data points laid out in vector g according to the lexical mapping $\pi(i,j,k)$

The $(B^{\mathrm{T}}B)^{-1}$ term may seem even more daunting than the one dimensional case, but all the elements are integers (but see weighting in section 4). This means that the determinant and cofactor matrix can be stably calculated efficiently. The complexity of this calculation is $O(lg(max(n)))$. If the coordinate system is centred, then symmetry can be exploited to decrease the constant.

The value we require is selected in the same manner as expression 11, amd this would be the case with any order of system. If we extend the nomenclature theme from 19, this becomes selecting the c^{th} derivative with respect to free variable ϕ_d (*i.e.* the d^{th} dimension), then our measurement is written as (h is the total number of dimensions):

$$\frac{\partial^c f}{\partial \phi_d^c} = \sum_{i_0=0}^{n_0} \cdots \sum_{i_d=c}^{n_d} \cdots \sum_{i_h=0}^{n_h} \frac{i!}{(i-c)!} a_{i_*} \phi_d^{i-c} \prod_{q \neq d} \phi_q^{i_q} \tag{22}$$

Writing the s vector from expression 13 out by hand becomes implausible, but it is simply defined as a vector of length $\prod_{0 \leq i < h}(1+n_i)$, with elements placed in h nested loops (which might be implemented recursively for higher numbers of dimensions) of value $\frac{i!}{(i-c)!}\phi_d^{i-c}\prod_{q\neq d}\phi_q^{i_q}$ at position $\pi(i*)$.

At this point we have a complete description of the multidimensional solution without rotation. This demonstrates the direct construction of a three dimensional Savitzky Golay style filter of arbitrary polynomial order in each direction and arbitrary patch size, for which there has previously been no practical method of construction. It is particular example of the general solution for a filter on multiple dimensions.

4 Weighting

The derivation so far assumes unit weight for each data point. If we wish to modulate the influence of the data points according to some constant map – for example, de-emphasizing distant pixels for a smoothing operation, or missing out pixels which we

know to be "dead" – this is formulated as below for expression 4:

$$\epsilon_i = f(x_i) - g_i \tag{23}$$

$$u = \sum_{i=1}^{k} w_i \epsilon_i^2 \tag{24}$$

$$g = Ba \tag{25}$$

$$B^{\mathrm{T}} W g = B^{\mathrm{T}} W B a \tag{26}$$

From this, we can take the pseudo-inverse solution of the form $a = (B^{\mathrm{T}} W B)^{-1} B^{\mathrm{T}} W g$, where W is a diagonal matrix of weights $w_i \geq 0$ which modify the significance of errors in the fit of the model at the data points. These weights may be integers, which preserve the simple solution for $(B^{\mathrm{T}} W B)^{-1}$, zero to exclude a data point, or arbitrary (positive) to allow matching with a general weighting regime. This latter case is most appropriately solved using QR factorization [21].

4.1 Reconstruction

If we wish to reconstruct a missing data point at x, y in an image, then we generate a convolution patch which applies zero weight to the datum at x, y, then select the value from the polynomial fit at that point.

5 Rotation and translation of the basis set

We observe that the filters we are constructing use basis functions each of which comprises an array of constants laid out on the data lattice, but the principal "directions" of the lattice need not define the principal axes of our model. We allow for the axes upon which the polynomial basis functions are defined to be rotated and translated with respect to the data lattice. This enables, for example, the polynomial order of the model parallel and normal to an edge to be selected independently.

6 Example application domains

6.1 Smoothing preserving edges

Currently, derivative calculation is most commonly performed using a smoothing filter followed by a differencing operator to estimate the derivatives of interest. This is probably due either to unfamiliarity with the Savitzky Golay approach (since there exist 2D filters for this purpose, but which seem a trifle daunting due to their particular formulation, which is an important motivation for the present article), or concerns about efficiency. We do not answer this concern here, but note that the stability of the solution to the problem using both approaches should certainly be explored for given problems.

6.2 Reconstruction in a faulty sensor

We can make a best guess about the intensity at a faulty location on a sensor by substituting a smoothed value generated by one of our filters of appropriate dimension. We can also calculate spatial derivatives ignoring broken pixels (or other sensor elements, such as fibres in an endoscope).

6.3 Image pyramid construction and Multigrid

Sub-sampling images to create resolution pyramid for multiscale analysis is already common, but with our formulation, this can be efficiently extended to higher-dimension problems including a restricted class of multigrid problems. Note that the lattice geometry need not be rectangular. It may be based on triangles, as long as they form a repeating pattern over the region to be smoothed/differentiated/sampled, or if the space can be mapped back to that required in a suitable manner.

6.4 Time series

We might model a sequence of images with coordinates x, y and t (time). A square patch in the image plane is most common, unless there are compelling reasons to use another aspect or shape, and we might use three time points (one before and one after the image of interest) with a cubic model in space to allow for edge detection (which commonly looks for a zero crossing of a second derivative) and a linear or model in time which assumes relatively small changes between frames.

7 Alternative basis sets

The polynomial basis set has a number of advantages, not least being the ease with which it is manipulated. We can also use other basis expressions for the approximating function, and obtain coefficients for the fit by the use of a convolution kernel. If we were to use Chebyshev polynomials, the solution we obtain in the end is exactly the same as with the general polynomial solution, but we can use, for example, sinusoidal functions, exponential functions, sigmoidal functions, or any function which can be parameterized to give a unique solution for the coefficients. This essentially corresponds the determinant of $B^{\mathrm{T}}B$ being non-zero. For example, we can use an exponential basis set, or a chirp-based set – perhaps $sin(e^{-ix})$ – if we believe the data is sensibly modelled that way.

8 Conclusions

The current state of the art provides formulations of general one-dimensional filters with odd or even numbers of sample points taking the measurement at any point, and of two dimensional filters of identical polynomial order in each dimension. Practical three dimensional convolutional polynomial matching kernels have not previously appeared in the literature.

In simplifying the components of the formulation as far as possible, we provide a transparent means for performing preparatory calculations which give convolutional kernels which smooth, find derivatives and reconstruct data points on an arbitrary polynomial model.

9 Acknowledgements

I would like to thank Uli Harder for discovering that Savitzky and Golay had trumped my synthesis of a 1D convolutional polynomial filter by forty years. My thanks go to Google and WoK for speeding the process of identifying the niche this current work fills.

References

[1] Abraham Savitzky and Marcel J. E. Golay, *Smoothing and Differentiation by simplified Least Squares Procedures Analytical Chemistry* Vol 36, No. 8 July 1964 pp 1672-1639

[2] Kerawala, S. M., *A rapid method for calculating the least squares solution of a polynomial of degree not exceedin the fifth* Indian Journal of Physics, 15, 241 (1941)

[3] Peter Meer and Isaac Weiss *Smoothed Differentiation Filters for Images*, Journal of Visual Communications and Image Representation Vol. 3, No. 1, March, pp. 58-72, 1992

[4] John E. Kuo, Hai Wang, Stephen Pickup, *Multidimensional Least Squares Smoothing Using Orthogonal Polynomials*, Anal. Chem. 1991 63, 630-635

[5] Jianwen Luo, Kui Ying, Jing Bai, *Savitzky-Golay smoothing and differentiation filter for even number data* Signal Processing 85 (2005) 1429-1434

[6] Jianwen Luo, Kui Ying, Jing Bai, *Properties of Savitzky-Golay digital differentiators* Digital Signal Processing 15 (2005) 122-136

[7] Peter A. Gorry, *General Least-Squares Smoothing and Differentiation by the Convolution (Savitzky Golay) Method*, Anal. Chem. 1990, 62, 570-573

[8] Peter A. Gorry, *General Least-Squares Smoothing and Differentiation of Nonuniformly Spaced Data by the Convolution Method*, Anal. Chem. 1991, 63, 534-536

[9] Kenneth L. Ratzlaff, Jean T. Johnson, *Comuputation of Two-Dimensional Polynomial Least Squares Convolutional Smoothing Integers*, Anal. Chem. 1989, 1303-1305

[10] Manfred U. A. Bromba and Horst Ziegler, *Application Hints for Savitzky-Golay Digital Smoothing Filters*, Anal. Chem. 1981, 53, 1583-1586

[11] Shlomo Greenberg, Daniel Kogan, *Improved structure-adaptive anisotropic filter*, Pattern Recognition Letters 27 (2006) 59-65

[12] G.Z. Yang, P. Burger, D.N.Furmin, S.R.Underwood, *Structure adaptive anisotropic image filtering*, Image and Vision Computing 14 (1996) 135-145

[13] P.Nikitas, A.Pappa-Louisi, *Comments on the two-dimensional smoothing of data*, Analytica Chimica Acta 415 (2000) 117-125

[14] William T. Freeman and Edward H. Adelson, *The design and use of steerable filters*, IEEE transactions on pattern analysis and machine intelligence, Vol. 13, No. 9, September 1991

[15] Philip Barak, *Smoothing and Differentiation by an Adaptive-Degree Polynomial Filter*, Anal. Chem., 1995, 67, 2758-2762

[16] Malgorzata Jakubowska, Wladyslaw W. Kubiak, *Adaptive-degree polynomial filter for voltammmetric signals*, Analytica Chimica Acta 512 (2004) 241-250

[17] http://research.microsoft.com/users/jckrumm/SavGol/SavGol.htm *We refer to this web page, which has been referenced in other publications, as it appears at the time of writing of this report.*

[18] E. H. Moore, *On the reciprocal of the general algebraic matrix*. Bulletin of the American Mathematical Society 26, 394-395 (1920)

[19] Roger Penrose, *A generalized inverse for matrices.* Proceedings of the Cambridge Philosophical Society, 51, 406-413 (1955)

[20] Roger Penrose *On best approximate solution of linear matrix equations*, Proceedings of the Cambridge Philosophical Society 52, 17-19 (1956)

[21] G. Golub & C. van Loan (1996), Matrix computations, third edition, The Johns Hopkins University Press, London