Fast recovery of the optic flow around a closed contour using stabilised regression onto Fourier components

Guy Scott

Oxford University, Oxford.

Abstract

Hildreth described a method of extracting the normal component of optic flow along a moving closed contour. She also described an algorithm - based on gradient descent - of recovering the smoothest *full* flow field consistent with the normal flow. Her iterative method is somewhat slow. It also leaves open the question of what, if anything, we are going to *say* about the full flow field after we have obtained it.

In this paper I describe a method of obtaining the smoothest flow field by expanding it in terms of Fourier components - the natural basis functions for a cyclical function. These have a number of nice properties that greatly assist us. Notably, the amplitude of any given frequency relates in a direct and simple manner to "smoothness" - a fact that enables us to "regularise" our procedure simply and economically.

1 Introduction

Function fitting involves "explaining" a dependent variable (which may be a vector) as the weighted sum of a set of functions defined over one or more independent variables. There are two canonical cases. Where the observations are dense - or have some distribution known in advance - a set of *orthogonal* functions may be derived in advance and each one fitted independently to any set of observations. Where the distribution of observations is not known in advance any set of explanatory functions will, in general, be found to co-vary. In this case a least squares regression is indicated.

Fourier functions are the prime exemplar of a set of orthogonal functions. If I define a scalar quantity F over the range $0 < x < 2\pi$ I may obtain its Fourier expansion - in terms of sines and cosines let us say - by taking

$$\frac{\int FE}{\int E^2} \tag{1}$$

(where E = cos(ωx) or sin(ωx)) for ω = 0, 1, 2, 3 etc. The Fast Fourier Transform incorporates a felicitous trick to perform this operation extremely fast (and in a manner ideally suited to parallel computation).

Suppose, however, that I do not know F for every value of x - there are missing observations. The Fourier transform is now undefined. There is an infinite number of Fourier transforms perfectly consistent with the data to hand - they correspond to the infinite number of "hallucinations" I can have concerning the missing observations.

In this situation it is necessary to invoke some simplicity, "energy" or smoothness constraint to govern the interpolation of the missing values. I could perform the interpolation directly (by locally based smoothing, say) and then do the conventional FT. Or I could truncate the Fourier functions at a suitable frequency and perform a least squares regression. Or - a modification of the truncation - I can perform a "regularised" LSR in which the high frequencies are penalised relative to the low ones. In this paper I will examine a combination of truncation and regularisation and show that it is neither as crudely heuristic, nor as computationally expensive, as one's initial "knee-jerks" might suggest. It is in fact an extremely quick and accurate method of interpolating a (cyclical) function. It may be extended to the recovery of a vector function from partial data - for example we can recover the full flow field around a closed contour, given only the normal component of flow.

2

The degree of "smoothness" of a function is normally defined as the integral of the square of some differential quantity. For a scalar function Q in one dimension the most common measure is:

$$\int |\frac{dQ}{dx}|^2 dx \tag{2}$$

Higher order "smoothness" may be defined in terms of higher derivatives. In general smoothness of "order n" is

$$\int |\frac{d^n Q}{dx^n}|^2 dx \tag{3}$$

Differentiation in the function domain corresponds to multiplication by iω in the Fourier domain. If we write

$$Q = a_0 + a_1 cos(x) + a_2 cos(2x) + a_3 cos(3x) ... + b_1 sin(x) + b_2 sin(2 \tag{4}$$

$$\frac{dQ}{dx} = -a_1 sin(x) - 2a_2 sin(2x) - 3a_3 sin(3x) ... + b_1 cos(x) + 2b_2 co \tag{5}$$

AVC 1989 doi:10.5244/C.3.38

Parseval's theorem tells us that the energy (integral of the square) of a function is the same as the energy in the Fourier domain i.e.

$$\int |\frac{dQ}{dx}|^2 dx = \sum_{\omega} \omega^2 (a_\omega^2 + b_\omega^2) \qquad (6)$$

This is a very useful result because it points the way to a very simple procedure for optimising smoothness in a Fourier expansion fitted to a partially sampled function. It also enables us to truncate the expansion in a principled way. The Fourier component of frequency omega costs us ω^{2n} times more in "roughness" than the component associated with the fundamental frequency. Depending on the required accuracy and the premium placed upon smoothness relative to compliance with the data, frequencies above some level can be safely ignored - particularly for higher-order definitions of smoothness.

3

Figure 1 shows observations made on a function F(x); the set of observations is not complete. Figure 2 shows the *weighting function* W(x) which, for simplicity, we will assume to have value 1 everywhere we have made an observation and zero elsewhere. The simple least squares fit of our observations to any given set of basis functions E is given by the solution of the linear equations $\mathbf{Va} = \mathbf{y}$.

$\mathbf{V}$ is the variance-covariance matrix of every pair of functions under the weighting function. viz. the element (i, j) is $\sum_x W^2 E_i E_j$. $\mathbf{a}$ is the vector of coefficients that we seek to estimate. $\mathbf{y}$ is the vector of covariances between the E and the weighted function WF. In this case that the E's are sines and cosines $\mathbf{y}$ is simply the Fourier Transform of WF suitably arranged.

If the number of basis functions E is in excess of the number of observation points on F the variance-covariance matrix is degenerate - and we might anyway desire to "regularise" our solution to be as smooth as is consistent with our trust in the data. Regularisation takes the form of adding positive terms to the diagonal elements of the variance-covariance matrix in a way that reflects prejudice against the corresponding basis functions.

The problem becomes $(\mathbf{V} + \mathbf{rI})\mathbf{a} = \mathbf{y}$ where $\mathbf{r}$ is a vector containing positive constants λ_i^2.

The solution minimises

$$W^2(\sum_x F - \sum_i a_i E_i)^2 + \sum_i \lambda_i^2 \qquad (7)$$

For nth order smoothness on we set λ equal to some constant κ multiplied by ω^n. Here κ acts as the "smoothing pressure". A low value implies that the solution will comply slavishly with the data (though any interpolation will be as smooth as possible). A high value implies a high premium on smoothness relative to compliance with the data.

A pleasant surprise awaits us when we come to form $\mathbf{V}$. We discover we can read its elements off the Fourier Transform of W^2! For example the element

$$\sum W^2 sin(ax)cos(bx) \qquad (8)$$

can be rewritten (ignoring a factor of 0.5);

$$\sum W^2 sin((a+b)x) - \sum W^2 sin((a+b)x) \qquad (9)$$

...a simple difference between two Fourier coefficients of the function W^2 (and similarly for other cross terms). It is thus unnecessary to explicitly evaluate the basis functions and to form and accumulate cross products of them. You just re-arrange The FFT of the square of the weighting function! The algorithm is:

1. FFT on W^2 and "rewrite" to obtain $\mathbf{V}$
2. FFT on WF to obtain $\mathbf{y}$
3. Add regularisers to the diagonal elements of $\mathbf{V}$
4. Solve for a
5. Reconstruct the interpolated function $\hat{F}$ by reverse FFT if desired.

4

Figures 3-6 show the curve fitted to F under four different regularisation conditions. Figure 3 shows first order smoothing combined with quite high smoothing pressure. Figure 4 shows first order smoothing with low pressure (the compliance with the data points is perfect). Figures 5 and 6 show the results of second order smoothing (n=2) for low and high pressure respectively. These results are to all intents and purposes identical to those that would be obtained with gradient descent methods - only the closed-form least-squares method is order of magnitude faster. Note further that iterative methods are intrinsically non-parallel whereas the FFT and matrix-inversion operations involved in our procedure are ideally suited to distributed computation on such devices as transputers.

In these examples 15 "explanatory functions" are used - the constant and the sine and cosine at the first seven frequencies. In most practical situations this would surely suffice the fussiest interpolator. The data would have to be *very* good to justify fitting further harmonics.

The superiority of the least-squares technique becomes more marked as the desired order of smoothness is increased. Iterative techniques for obtaining second- or higher order smoothness are excruciatingly slow . The least-squares technique actually gets *faster* with high orders because the regularisation term grows more rapidly with frequency and the series can be truncated at an early stage.

5

Extension to recovery of a vector flow field from "normal" components is relatively straightforward. The relationship between normal flow and the true flow at any point is given by the motion constraint equation:

$$v = U\dot{x} + W\dot{y} \qquad (10)$$

where v is the flow in the normal direction, $\dot{x}$ and $\dot{y}$ are the components of true motion, and U and W are the cosine and sine of the direction of the normal relative to the x-axis.

To obtain a function-fit to the full flow field we simply write $\dot{x}$ and $\dot{y}$ as (independent) Fourier expansions on normalised arc-length around the closed contour. We solve the set of linear equations that result from making observations of v, U and W in a (stabilised) least squares sense. Note that the variance-covariance for the problem contains products of the form $\sum U^2 E_i E_j$, $\sum V^2 E_i E_j$, and $\sum UV E_i E_j$. Three FFT's are required in order to obtain these terms (on U^2, V^2 and UV). Otherwise the algorithm is as given above for the scalar case.

6

I have described a closed-form method of obtaining the smoothest flow-field from edge data associated with a closed contour. This method exploits certain benign properties of the Fourier series. The method is much faster than iterative methods, even on serial hardware.

The reader will have seen many pictures of reconstructed flow fields, but have you ever seen an invisible flow field? A circle rotating around its centre has zero normal flow everywhere. In general: a curve yields zero normal flow in a field for which it is a streamline. The hypothesis of zero motion will be returned by a stabilised least squares method in these circumstances (it is certainly the smoothest solution available!). Suppose, however, that we insist on non-zero flow. We want the smoothest flow along the curve *subject* to the condition that the flow has a given level of energy. It can be easily shown that the coefficients of such a field are given by the eigenvector of the variance-covariance matrix that corresponds to the lowest eigenvalue. Figure 7 shows the result for the boundary of a fibroblast (courtesy ICRF). Note how the "internal flow" slows down in regions of high curvature. This raises the intriguing possibility of applications to the segmentation and "natural parametrisation" problems.

figure 1

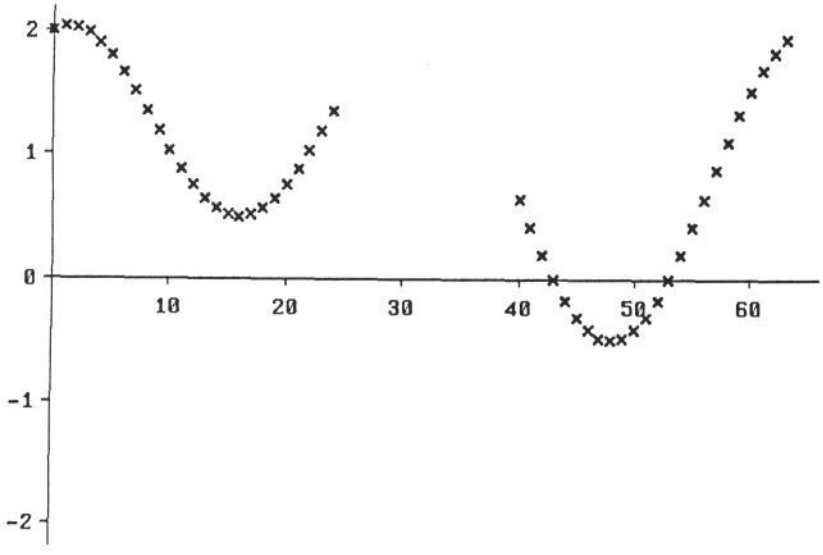

figure 2

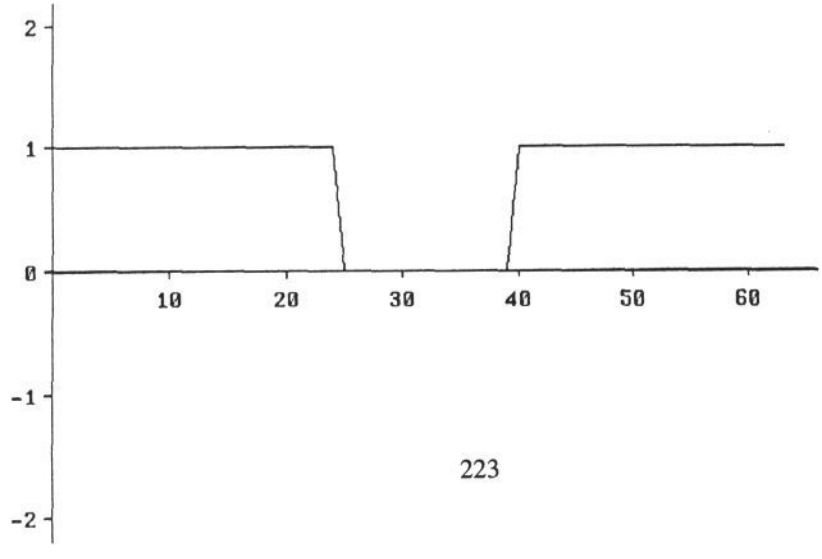

figure 3

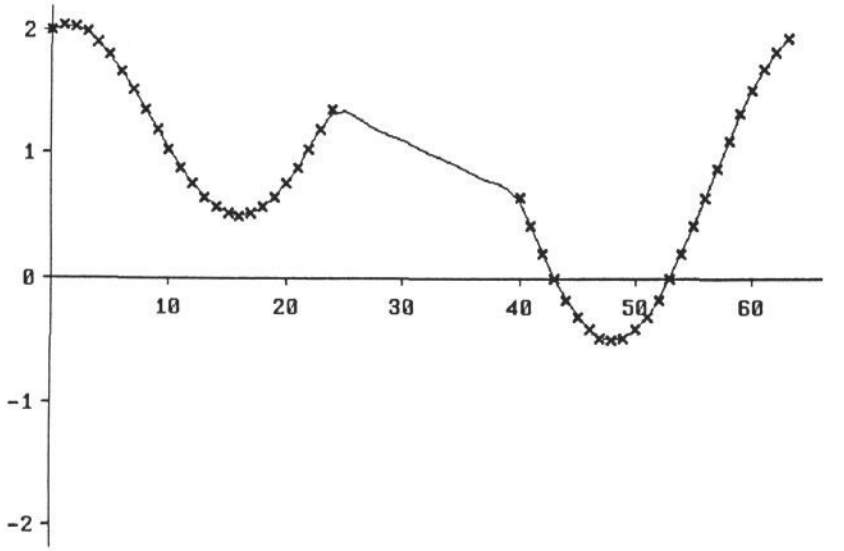

figure 4

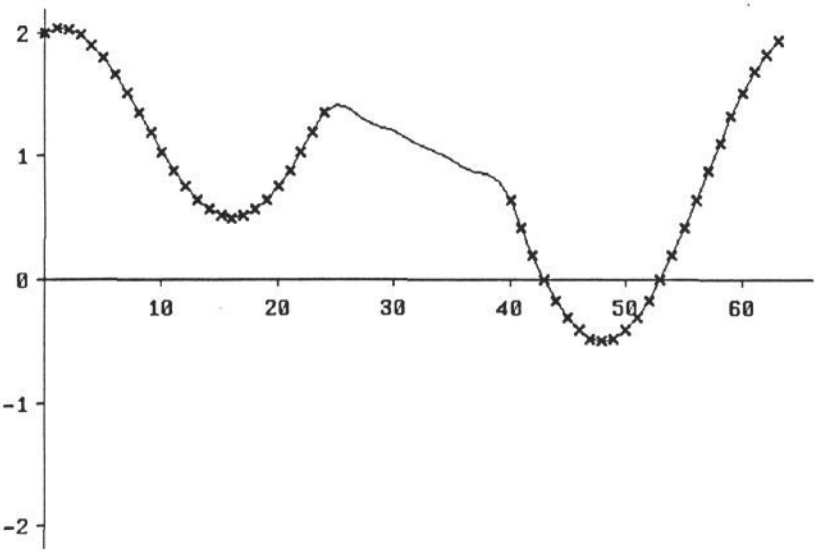

figure 5

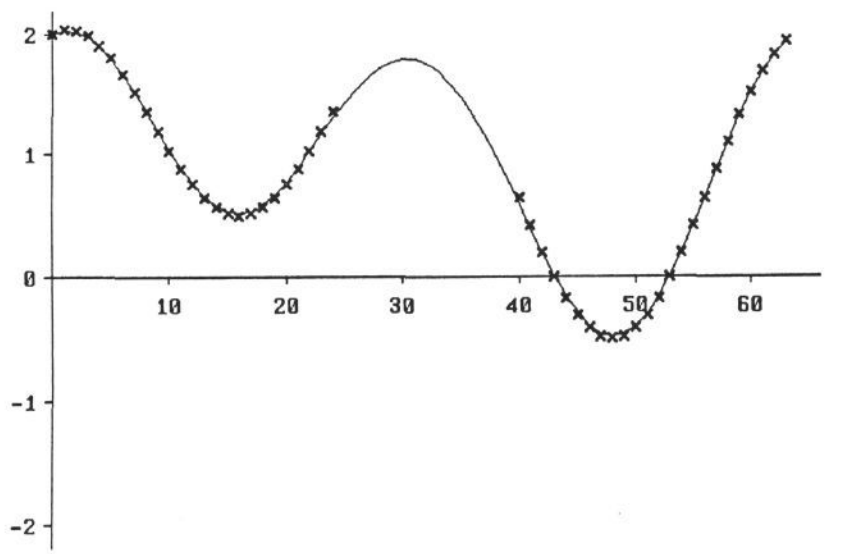

figure 6

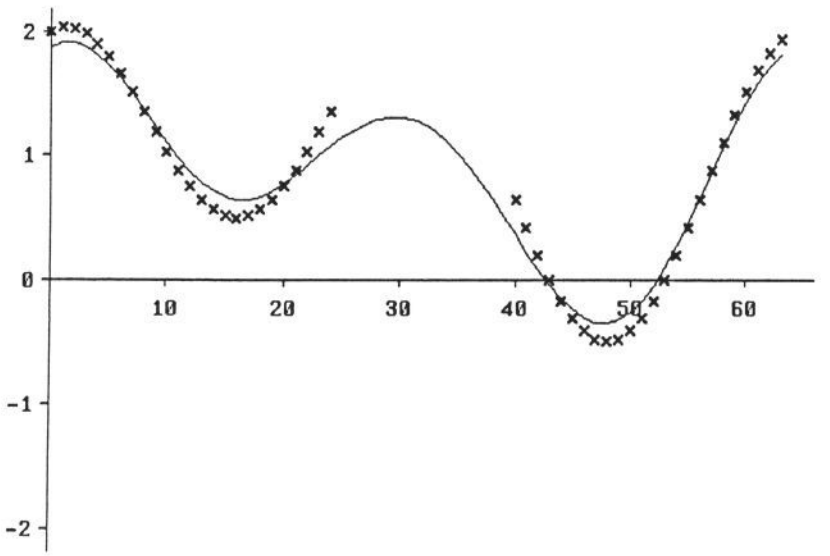

figure 7