THE UNIVERSITY OF ADELAIDE

Towards an estimation framework for some problems in computer vision

Darren J. Gawley

School of Computer Science
The University of Adelaide

Cooperative Research Centre for Sensor
Signal and Information Processing

September 2004

A THESIS SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN THE FACULTY OF ENGINEERING, COMPUTER AND MATHEMATICAL SCIENCES

ABSTRACT

This thesis is concerned with fundamental algorithms for estimating parameters of geometric models that are particularly relevant to computer vision. A general framework is considered which accommodates several important problems involving estimation in a maximum likelihood setting. By considering a special form of a commonly used cost function, a new, iterative, estimation method is evolved. This method is subsequently expanded to enable incorporation of a so-called ancillary constraint. An important feature of these methods is that they can serve as a basis for conducting theoretical comparison of various estimation approaches.

Two specific applications are considered: conic fitting, and estimation of the fundamental matrix (a matrix arising in stereo vision). In the case of conic fitting, unconstrained methods are first treated. The problem of producing ellipse-specific estimates is subsequently tackled. For the problem of estimating the fundamental matrix, the new constrained method is applied to generate an estimate which satisfies the necessary rank-two constraint. Other constrained and unconstrained methods are compared within this context. For both of these example problems, the unconstrained and constrained methods are shown to perform with high accuracy and efficiency.

The value of incorporating covariance information characterising the uncertainty of measured image point locations within the estimation process is also explored. Covariance matrices associated with data points are modelled, then an empirical study is made of the conditions under which covariance information enables generation of improved parameter estimates. Under the assumption that covariance information is, in itself, subject to estimation error, tests are undertaken to determine the effect of imprecise information upon the quality of parameter estimates. Finally, these results are carried over to experiments to assess the value of covariance information in estimating the fundamental matrix from real images. The use of such information is shown to be of potential benefit when the measurement process of image features is considered.

DECLARATION

This thesis contains no material that has been accepted for the award of any other degree or diploma in any University or other tertiary institution. To the best of my knowledge and belief, it contains no material previously published or written by any other person, except where due reference is made in the text.

I give consent for this thesis, when deposited in the University library, to be available for loan and photocopying.

Darren J. Gawley
September 2004

ACKNOWLEDGMENTS

I would primarily like to thank Mike Brooks and Wojciech Chojnacki for their excellent supervision, help, advice, and guidance during the course of my candidature. Additionally, my thanks also goes to the other members of the Computer Vision group – Anton van den Hengel, John Bastian, Rhys Hill, Daniel Pooley and Tony Scoleri – for many interesting discussions and shared experiences.

I would also like to acknowledge the Cooperative Research Centre for Sensor Signal and Information Processing, and the people therein, for all the great advice and support they have provided.

PUBLICATIONS

In undertaking research which contributes to this thesis, a number of papers were published.

[1] W. Chojnacki, M. J. Brooks, A. van den Hengel, and D. Gawley. On the fitting of surfaces to data with covariances. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1294-1303, 2000.

[2] M. J. Brooks, W. Chojnacki, D. Gawley, and A. van den Hengel. Is covariance information useful in estimating vision parameters? In S. F. El-Hakim and A. Gruen, editors, *Videometrics and Optical Methods for 3-D Shape Measurement*, San Jose, California, USA, Jan. 2001, volume 4309 of Proceedings of SPIE, pp. 195-203, 2001.

[3] M. J. Brooks, W. Chojnacki, D. Gawley, and A. van den Hengel. What value covariance information in estimating vision parameters? In Proceedings, *Eighth IEEE International Conference of Computer Vision*, Vancouver, British Columbia, Canada, July 2001, vol. 1, pp. 302-308. IEEE Computer Society.

[4] W. Chojnacki, M. J. Brooks, A. van den Hengel, and D. Gawley. A fast MLE-based method for estimating the fundamental matrix. In B. Mercer, editor, Proceedings of the *IEEE International Conference on Image Processing (ICIP-01)*, Thessaloniki, Greece, 7-10 October, 2001, vol 2, pp. 189-192. IEEE Computer Society, 2001.

[5] W. Chojnacki, M. J. Brooks, A. van den Hengel, and D. Gawley. From FNS to HEIV: A link between two vision parameter estimation methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):264-268, 2004.

[6] W. Chojnacki, M. J. Brooks, A. van den Hengel, and D. Gawley. A new constrained parameter estimator for computer vision applications. *Image and Vision Computing*, 22(2):85-91, 2004.

TABLE OF CONTENTS

LIST OF FIGURES

NOTATION

$\boldsymbol{a} \in \mathbb{R}^p$	:	column vector of length p
$\boldsymbol{A} \in \mathbb{R}^{n \times m}$	:	$n \times m$ dimensional matrix
$\boldsymbol{I}_p, \boldsymbol{I}$	:	$p \times p$ identity matrix (generally p is inferred from context)
$\boldsymbol{A}^\top$	:	transpose of $\boldsymbol{A}$
$\boldsymbol{A}^{-1}$	:	inverse of $\boldsymbol{A}$
$\boldsymbol{A}^-$	:	Moore-Penrose inverse of $\boldsymbol{A}$
$\lvert\boldsymbol{A}\rvert$	:	determinant of $\boldsymbol{A}$
$\lVert\boldsymbol{A}\rVert_{\mathrm{F}}$	:	Frobenius norm of $\boldsymbol{A}$
$\mathrm{rank}(\boldsymbol{A})$	:	rank of $\boldsymbol{A}$
$\mathrm{diag}(\sigma_1, \ldots, \sigma_n)$	:	$n \times n$ matrix with $\sigma_1, \ldots, \sigma_n$ along the diagonal and zeros elsewhere
$\mathrm{tr}(\boldsymbol{A})$	:	trace of matrix $\boldsymbol{A}$
$\mathrm{vec}(\boldsymbol{A})$	:	vectorisation of $\boldsymbol{A}$
$\partial_{\boldsymbol{x}} J$	:	gradient **row** vector $[\frac{\partial J}{\partial x_1}, \ldots, \frac{\partial J}{\partial x_n}]$
$\partial^2_{\boldsymbol{xx}} J$	:	Hessian matrix of J, $\left[\frac{\partial^2 J}{\partial x_i \partial x_j}\right]_{1 \le i,j \le n}$.
$\boldsymbol{H}_J$	:	alternative notation for the Hessian matrix of $J(\cdot)$
$p(X)$	:	probability of event X
$\mathsf{E}[\mathsf{x}]$	:	expected value of the random variable x
$\boldsymbol{A} \otimes \boldsymbol{B}$	:	Kronecker product of $\boldsymbol{A}$ and $\boldsymbol{B}$

Chapter 1

INTRODUCTION

1.1 Preface

This thesis is concerned with fundamental algorithms for estimating parameters of geometric models particularly relevant to computer vision. Parameter estimation for vision applications is not straightforward and many methods and techniques exist to deal with this task. This thesis describes new ways to estimate parameters together with a framework for estimation. This enables existing and new estimation methods to be seen, and understood, from a unified viewpoint.

Meaningful interpretation of images — a broad goal of computer vision — requires processing vast amounts of raw data. In order to obtain sensible and usable output, the information contained in input data must often be reduced to a small number of key parameters. These parameters encapsulate properties of a higher level entity of relevance. In this work, the entities will relate to the geometry of image formating, requiring an imposition of a specific mathematical model dictated by this underlying geometry. The model employed will have associated parameters, which are related to input data derived from one or more images. The task is to estimate these parameters from given data.

A large body of work in classical statistics (summarised extensively by Kendall [35]) exists in regard to parameter estimation. The computer vision experience is somewhat unusual when compared with this conventional statistical estimation theory because of the non-standard method of 'measuring' data through image acquisition. In a broad sense, one goal of classical statistics is to estimate parameters given *multiple samples* of a random variable. In computer vision an image is taken in an instant, and thus for many problems there is in effect only a single sample of various random variables. In this context, estimation can become a moving target – each new measurement changes the parametric model increasing its dimensionality (by adding so-called nuisance pa-

rameters). It is critical then to tease out the complex relationship between parameters, measured data, and the underlying model.

Over the last 15 years, significant progress has been made towards understanding the geometry behind the image formation process. This was related with the development of parametric models describing the geometry. The key question is to devise practical methods for estimating parameters based on data points derived from images. This thesis presents such an estimation problem and treats existing and newly proposed estimation methods within a common context.

Following the terminology of Zhang [68], the parameter estimation problem can be broken down into four categories:

- Modelling: Choosing an overall mathematical model describing the data at hand and explaining the relationship to underlying geometry. The specifics of the model are embodied by model parameters. Additionally, any such model should mitigate the deleterious effects of the errors inherent in measured data.

- Criterion: Choosing a particular cost function to minimise. In many respects this is the most challenging of problems, as *a priori* there can be no definitive way of determining the "best" cost function.

- Estimation: Given a specific cost function, how is its minimiser found?

- Design and implementation: For a given minimisation method, an efficient and accurate means must be chosen for its implementation. Care must be taken to ensure minimal corruption due to numerical instability and roundoff. As well, efficiency of computation plays a significant rôle in this regard.

This thesis focuses on the middle two categories: criterion and estimation. All existing methods discussed are provided in the context of an explicit cost function that is to be minimised – something not always present in the original derivations. Furthermore, general optimisation methods are compared to those designed with specific knowledge of the cost function form.

1.2 Outline of contribution

This thesis presents a fairly general and unified approach to the problem of parameter estimation in computer vision. With an explicitly defined model, existing and new

methods are analysed within a consistent context. A new estimator is presented which is simply expressed, efficient, and easy to implement. As important vision problems involve constraints on parameters, the topic of constrained estimation is also given prominence, with a constrained version of the new estimator also developed.

Unconstrained and constrained estimation methods are applied to two example problems: conic section fitting and estimating the fundamental matrix. These methods are seen to be accurate and efficient when compared to existing methods.

Additionally the topic of covariance matrices, and their use as input to estimation methods, is explored. Although many estimators are able to utilise covariance information as input, this is often ignored and the identity matrix is taken as a default covariance matrix. Chapter 6 presents results of novel experiments conducted with the use of non-identity covariance matrices. These help elucidate in which circumstances the incorporation of covariance information is useful.

The first part of this thesis deals with the underlying theory of parameter estimation, specifically related to a mathematical model appropriate for computer vision. This section describes a consistent framework whereby many different estimators may be compared. The second half demonstrate how general techniques may be applied to specific vision problems.

In the initial theoretical section a parametric model is defined with its associated assumptions about parameters, data, and their organisation. Built on this is a description of the estimation process defined explicitly using cost functions. Increasingly sophisticated cost functions are defined, culminating with a maximum likelihood cost function, and an approximated maximum likelihood cost function. Following the definition of these cost functions, the next chapter looks at how respective minimisers may be isolated. A variational argument is used to derive a new method. This method is upgraded to take into account an ancillary constraint.

Following the theoretical material, an application of various estimation techniques is presented in the context of conic (and, later, ellipse specific) fitting. Several existing methods, which are either unconstrained (and estimate general conics), or have an ellipse specific constraint imposed are described. Based on the theory developed in previous sections, some of these methods can be further extended. Experiments are shown, with specific emphasis not only on the accuracy of estimators, but also on their computational efficiency.

The next application considered is that of estimating the fundamental matrix. Again,

existing estimators are presented, and, building on the theory developed, new methods are put forward. Experimental results are presented in a similar way to those for conic fitting. In particular, the constrained estimator is put to use, accounting for the so-called rank-two constraint.

A chapter is devoted to the issue of using covariance matrices as input to the estimation process. In this case, input data points are assumed to come equipped with a characterisation of data uncertainties. Several key synthetic experiments are performed to highlight how information about uncertainties may help, or hinder the estimation process. These ideas are applied to example real stereo image pairs.

The final chapter is a review of all of the material presented. Several areas for further work, including possible extensions of the model, are discussed.

Chapter 2

PROBLEM FORMULATION

This chapter describes groundwork necessary for discussing parameter estimation. It starts with an explicit definition of a parametric model where all specific assumptions are laid bare. These include how data are considered, what the noise affecting the data is, and what constraints are placed on the model's parameters. Within the parametric model specified, a least squares approach to estimation is then discussed. The total least squares method is explored in some detail, as it is an example of a simple non-iterative estimation technique. After considering questions of orthogonal projection, a maximum likelihood formulation is introduced. It is demonstrated that the maximum likelihood form is difficult to work with because of its reliance on "nuisance" parameters. Finally, guided by the work of Sampson, an approximated maximum likelihood form is derived which makes no explicit use of nuisance parameters.

2.1 Parametric model

A *parametric model* can be set up to describe the relationships between image "tokens". These relationships reflect a particular geometric property of interest. Image tokens, for example the locations of a 2-D points on an image, are taken as data and are represented by a vector $\boldsymbol{x} = [x_1, \ldots, x_k]^\top$. A parametric model relates these data vectors with a vector of parameters $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_l]^\top$. The length of $\boldsymbol{\theta}$ may exceed the number of the underlying degrees of freedom in the geometric situation at hand.

Such a parametric model may be described by an associated principal *constraint equation*

$$\boldsymbol{f}(\boldsymbol{\theta}, \boldsymbol{x}) = \mathbf{0}. \tag{2.1}$$

Here, $\boldsymbol{f}$ is a non-linear vector function with m components. The equation is described as a 'constraint' because, for a given data vector, the possible values of parameter vectors are constrained to those which satisfy the equation.

A fundamental aspect of these problems is that, for an arbitrary set of data, contaminated by noise, there is unlikely to be a single $\boldsymbol{\theta}$ for which the constraint equation will hold for all elements of data. And even if such a value of $\boldsymbol{\theta}$ existed, it would be too dependent on noise to be useful.

2.2 Specialisations

To facilitate analysis and tractability, some specialisations are necessarily applied to the general model.

Model order

Some problems lend themselves to being naturally described by a parametric model for which the range space of the principal constraint function $\boldsymbol{f}$ has dimension greater than one. Such models are termed *multi-objective* models. By expressing the individual elements of $\boldsymbol{f}$ in the constraint equation

$$\boldsymbol{f}(\boldsymbol{\theta}, \boldsymbol{x}) = \begin{bmatrix} f_1(\boldsymbol{\theta}, \boldsymbol{x}) \\ f_2(\boldsymbol{\theta}, \boldsymbol{x}) \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \end{bmatrix},$$

the equation is formed as a collection of scalar functions, all of which must be satisfied for a valid parameter vector. The first simplification imposed is to adopt a *single objective* model, which has only one scalar-valued constraint function. Several important computer vision problems are naturally described with a single objective principal constraint function. This restriction has the advantage of simplifying the analysis and necessary derivations. Once the theory for single objective problems is elaborated, the multi-objective case can be developed via analogy.

Non-linearity

The constraint function $\boldsymbol{f}$ is non-linear in general. There are three possible ways for this function to be non-linear: it can be non-linear in $\boldsymbol{x}$ but linear in $\boldsymbol{\theta}$, it can be non-linear in $\boldsymbol{\theta}$ but linear in $\boldsymbol{x}$, or it can be non-linear in both $\boldsymbol{x}$ and $\boldsymbol{\theta}$. The rest of this thesis will consider only the first case: models which are non-linear in the data and linear in the parameters. For such models, it is possible to write the principal constraint function as

$$f(\boldsymbol{\theta}, \boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}) = 0. \tag{2.2}$$

The function $\boldsymbol{u}(\boldsymbol{x})$, which encapsulates the non-linearity in the data, is termed a *carrier* function. It is sometimes abbreviated as $\boldsymbol{u}_{\boldsymbol{x}}$. (This 'function-argument-as-subscript' notation is used throughout the thesis; for example, a matrix $\boldsymbol{M}(\boldsymbol{\theta})$ which is a function of $\boldsymbol{\theta}$ may be interchangeably written as $\boldsymbol{M}_{\boldsymbol{\theta}}$.)

The carrier $\boldsymbol{u}(\boldsymbol{x}) = [u_1(\boldsymbol{x}), \ldots, u_l(\boldsymbol{x})]^\top$ is a vector with the data point transformed so that: firstly, each component $u_i(\boldsymbol{x})$ is quadratic in the compound vector $[\boldsymbol{x}^\top, 1]^\top$, and secondly the last component $u_l(\boldsymbol{x})$ is equal to one.

Noise model

Each data point $\boldsymbol{x}_i$ can be thought of as a perturbation from some *ideal value* $\overline{\boldsymbol{x}}_i$, where

$$\boldsymbol{x}_i = \overline{\boldsymbol{x}}_i + \boldsymbol{\delta}_{\boldsymbol{x}_i} \,. \tag{2.3}$$

Each ideal point will satisfy the principal constraint, with $\boldsymbol{\theta}^\top \boldsymbol{u}(\overline{\boldsymbol{x}}_i) = 0$ for all i, for a common value of $\boldsymbol{\theta}$. The set of all consistent points is said to lie on a *consistency manifold* defined as

$$\mathcal{M}_{\boldsymbol{\theta}} = \left\{ \boldsymbol{x} \mid \boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}) = 0 \right\} \,. \tag{2.4}$$

The next assumption applied is that the distributions of the perturbations are Gaussian and that the noise affecting each element of data is independent of the noise contaminating all other data points and is also independent of the value of $\boldsymbol{x}_i$. The noise depends only on a $k \times k$ *covariance matrix* $\boldsymbol{\Lambda}_{\boldsymbol{x}_i}$. The associated probability density function (p.d.f.) for these perturbations is therefore given by

$$p(\boldsymbol{\delta}_{\boldsymbol{x}} \mid \boldsymbol{\Lambda}_{\boldsymbol{x}}) = \frac{1}{\sqrt{(2\pi)^{\mathrm{k}} |\boldsymbol{\Lambda}_{\boldsymbol{x}}|}} \exp\left(-\frac{\boldsymbol{\delta}_{\boldsymbol{x}}^\top \boldsymbol{\Lambda}_{\boldsymbol{x}}^{-1} \boldsymbol{\delta}_{\boldsymbol{x}}}{2} \right), \tag{2.5}$$

where $|\boldsymbol{\Lambda}_{\boldsymbol{x}}|$ denotes the determinant of $\boldsymbol{\Lambda}_{\boldsymbol{x}}$. The relationship between $\boldsymbol{x}$, $\overline{\boldsymbol{x}}$, and $\boldsymbol{\Lambda}_{\boldsymbol{x}}$ is shown in Fig. 2.1.

Ancillary constraint

The principal constraint (Eq. (2.2)) describes a system whereby any parameter vectors are compared to given data points. For many relevant problems an additional *ancillary* constraint of the parameter vector is required. Such a constraint is completely independent of any data points. Regardless of the parameters which may be the best fit to

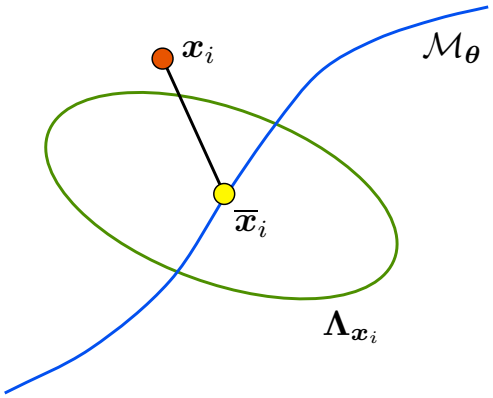


Figure 2.1: A *consistent* point lying on the manifold of all consistent points for a given $\boldsymbol{\theta}$. The ellipse represents a level-set of the p.d.f. of the perturbations dictated by the particular covariance matrix

given data, this ancillary constraint dictates a form to which the parameter vector must conform. Such a constraint is described by the ancillary constraint equation,

$$\psi(\boldsymbol{\theta}) = 0, \tag{2.6}$$

where ψ is a non-linear, scalar valued function of the parameter vector $\boldsymbol{\theta}$ only. In many cases, the ancillary constraint can be described by using a *homogeneous* function of degree κ, that has the property

$$\psi(\lambda\boldsymbol{\theta}) = \lambda^{\kappa}\psi(\boldsymbol{\theta}), \qquad \lambda \neq 0. \tag{2.7}$$

2.3 Example problem: (simple) parabola estimation

A simple example of an estimation problem of the above form is that of fitting a parabola to points on a plane. In this model the data elements are taken as 2-D planar points, and the parameters will describe a simple parabola. Such a parabola can be described by the quadratic equation

$$y = \alpha x^2 + \beta x + \gamma. \tag{2.8}$$

This problem may be posed in the form of the model defined above by writing the basic parabola equation in a homogeneous form

$$ax^2 + bx + cy + d = 0. \tag{2.9}$$

In this case, each element of data is a simple vector $\boldsymbol{x}_i = [x_i, y_i]^\top$. By assigning $\boldsymbol{\theta} = [a, b, c, d]^\top$ and adopting the carrier function $\boldsymbol{u}(\boldsymbol{x}) = [x^2, x, y, 1]^\top$, Eq. (2.9) may be written as per Eq. (2.2): $\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}) = 0$. Note that for a given parameter vector $\boldsymbol{\theta}$, the equation will describe the same parabola when multiplied by a non-zero scalar. An example parabola, with associated data and consistent points is shown in Fig. 2.2. These forms of data and carriers provide a simple example that is used to illustrate properties of carrier vectors in subsequent sections. A more general version of this problem, fitting conic sections to data points, is the topic of Chapter 4. Estimation problems arising from stereo vision are also readily amenable to this form. The task of finding the fundamental matrix is examined in Chapter 5. A different but related problem of estimating a *differential fundamental matrix* [6], can additionally be posed in the above form [3]. Kanatani and Ohta show that employing an ordinary fundamental matrix is superior to using the differential form [31], and so estimation of this latter form is not considered further in this thesis.

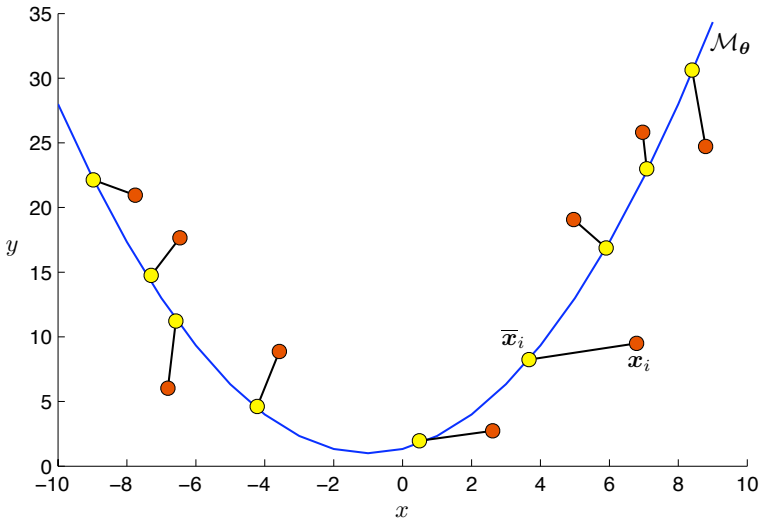


Figure 2.2: A consistency manifold corresponding to $\boldsymbol{\theta} = [1, 2, -3, 4]$, with sample true points and noisy data points

2.4 Least squares fitting

A classical means of solving fitting or regression problems is to use (ordinary) least squares. The ordinary least squares method (OLS) assumes that there are some non-

random explanatory variables, and associated (measured) 'noisy' response variables. In many kinds of experimentation this is exactly the case. For example, if measurements are made at certain time intervals, the only source of error might be the quantity measured, and not the independent variable of the time at which the measurements were taken.

In the simplest case of line fitting, given an explanatory variable x_i, the corresponding point on a line is

$$\bar{y}_i = \alpha x_i + \beta, \tag{2.10}$$

where α and β are parameters describing the line. Each response variable y_i is a measurement of $\bar{y}_i$ corrupted by δ_{y_i}:

$$y_i = \bar{y}_i + \delta_{y_i}. \tag{2.11}$$

Substituting Eq. (2.10) into Eq. (2.11) and adopting a vector form yields

$$\begin{aligned} y_i &= \begin{bmatrix} x & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} + \delta_{y_i} \\ &= \boldsymbol{v}(x_i)^\top \boldsymbol{\rho} + \delta_{y_i}, \end{aligned} \tag{2.12}$$

where $\boldsymbol{\rho} = [\alpha, \beta]^\top$ and $\boldsymbol{v}(x) = [x, 1]^\top$. Eq. (2.12) describes the relationship for each measurement. All of the measurements may be combined into a single matrix equation

$$\boldsymbol{y} = \boldsymbol{V}\boldsymbol{\rho} + \boldsymbol{\delta}, \tag{2.13}$$

with $\boldsymbol{y} = [y_1, \ldots, y_n]^\top$ encapsulating all of the y measurements, and similarly $\boldsymbol{\delta} = [\delta_{y_1}, \ldots, \delta_{y_n}]$ representing the errors due to noise. The classical method for determining an estimate $\widehat{\boldsymbol{\rho}}_{\mathrm{OLS}}$ is to use the Moore-Penrose inverse (sometimes called the *pseudo-inverse*) [4], whereby

$$\widehat{\boldsymbol{\rho}}_{\mathrm{OLS}} = (\boldsymbol{V}^\top \boldsymbol{V})^{-1} \boldsymbol{V}^\top \boldsymbol{y}. \tag{2.14}$$

Typically in computer vision applications all variables (generally describing the location of image feature points) are contaminated by noise. Therefore, as there is no distinction made between explanatory and response variables, an alternative method which imposes no distinction between variables must be considered.

2.4.1 Total least squares

The OLS method minimises the vertical distances from measured data points to a line. The adopted noise model from Sect. 2.2 has *all* measured variables perturbed by noise.

Hence an orthogonal distance minimisation is more appropriate. Fig. 2.3 shows an example of distances minimised.

The line equation $y = \alpha x + \beta$ is retained, and now both x and y are perturbed with noise, with

$$y_i = \bar{y}_i + \delta_i \qquad \text{and} \qquad x_i = \bar{x}_i + \epsilon_i. \tag{2.15}$$

The orthogonal distance between a line represented by α, β and a general point (x_i, y_i) is given by

$$d_i^{\perp} = \frac{|y_i - (\alpha x_i + \beta)|}{\sqrt{1+\alpha^2}}\,. \tag{2.16}$$

A line may be fit to a set of data points by minimising the sum of these squared distances. A natural cost function based on these orthogonal distances is

$$J_{\perp}(\alpha, \beta) = \sum_{i=1}^{n} (d_i^{\perp})^2 = \sum_{i=1}^{n} \frac{|y_i - (\alpha x_i + \beta)|^2}{1+\alpha^2} \tag{2.17}$$

At the minimiser,

$$[\partial_{\alpha} J_{\perp}] = \frac{-2}{1+\alpha^2} \sum_{i=1}^{n} \big(y_i - (\alpha x_i + \beta)\big) = 0\,. \tag{2.18}$$

Solving for β,

$$\beta = \tilde{y} - \alpha \tilde{x}, \tag{2.19}$$

where

$$\tilde{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{and} \qquad \tilde{y} = \frac{1}{n}\sum_{i=1}^{n} y_i.$$

Substituting the value of β into Eq. (2.17) yields

$$J_{\perp} = \sum_{i=1}^{n} \frac{|(y_i - \tilde{y}) + \alpha(x_i - \tilde{x})|^2}{1+\alpha^2}\,. \tag{2.20}$$

Adopting a vector form, with $\tilde{\boldsymbol{x}} = [\tilde{x}, \tilde{y}]^{\top}$, $\boldsymbol{x} = [x_i, y_i]^{\top}$, and $\boldsymbol{\eta} = [\alpha, 1]^{\top}$, gives

$$J_{\perp} = \sum_{i=1}^{n} \frac{|\boldsymbol{\eta}^{\top}(\boldsymbol{x}_i - \tilde{\boldsymbol{x}})|^2}{\|\boldsymbol{\eta}\|^2}\,. \tag{2.21}$$

Minimising $J_{\perp}$ with respect to $\boldsymbol{\eta}$ will produce an estimate of α. This can be substituted back into Eq. (2.19) in order to obtain an estimate of β, and hence the line.

As this strategy considers all variables undergoing noisy perturbation, it is referred to as *total least squares* (TLS) [19, 65]. The TLS estimate can be shown to be optimal when the noise effecting each data point is Gaussian, and independent of every other data point.

Moving beyond straight-line fitting, the TLS method can be used as inspiration to define an estimator for the general problem described in the previous section. Eq. (2.21) is adapted to define a TLS cost function

$$J_{\mathrm{TLS}}(\boldsymbol{\theta}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \sum_i \frac{|\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}_i)|^2}{\|\boldsymbol{\theta}\|^2}, \tag{2.22}$$

with an associated estimate

$$\widehat{\boldsymbol{\theta}}_{\mathrm{TLS}} = \arg\min_{\boldsymbol{\theta}} J_{\mathrm{TLS}}(\boldsymbol{\theta}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n).$$

The most significant change is that the term $(\boldsymbol{x}_i - \tilde{\boldsymbol{x}})$ has been replaced with carrier vectors, which involve non-linear transformations of the data points. This has implications for the assumptions of the TLS cost function and is explored in Sect. 2.4.4. Zhang [68], is one of many authors who saliently point out that there is no *a priori* justification for using the algebraic distance $|\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x})|$ apart from convenience and ease of implementation. Later sections will examine estimators which have firmer roots in the underlying geometry of the model at hand.

Minimising J_{TLS} amounts to minimising the sum of squared residuals of the algebraic constraint function defined in Eq. (2.2), subject to the constraint $\|\boldsymbol{\theta}\|^2 = 1$. It is possible to contemplate minimising the squared residuals subject to a different constraint on $\boldsymbol{\theta}$, which would yield a different estimatior which is yet still based on algebraic least squares. The TLS estimator can therefore be considered part of a family of 'algebraic least squares' estimators which act to minimise algebraic residuals under some given constraint on the paramater values.

The details of how a minimiser of J_{TLS} may be sought will prove useful for the derivation of more sophisticated methods described later.

2.4.2 Finding the TLS estimate

The squared sum of algebraic residuals may be written in matrix form as

$$\begin{aligned} \sum_i |\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}_i)|^2 &= \boldsymbol{\theta}^\top \Big(\sum_i \boldsymbol{u}(\boldsymbol{x}_i) \boldsymbol{u}(\boldsymbol{x}_i)^\top \Big) \boldsymbol{\theta} \\ &= \boldsymbol{\theta}^\top \boldsymbol{U}^\top \boldsymbol{U}\, \boldsymbol{\theta} \\ &= \boldsymbol{\theta}^\top \boldsymbol{S}\, \boldsymbol{\theta}, \end{aligned} \tag{2.23}$$

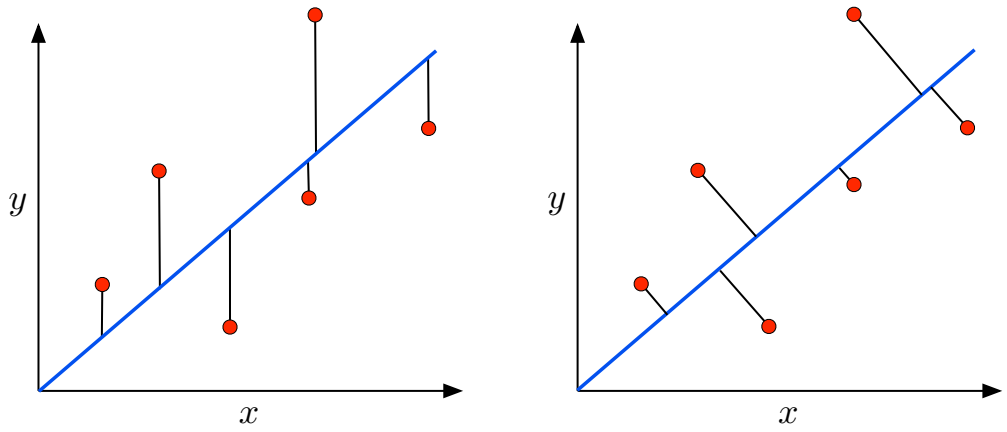


Figure 2.3: Ordinary least squares errors (left), and orthogonal distance errors (right). The errors in the OLS case are dependent on the position and orientation of the model. They become particularly unstable as the line tends to parallel with the $y-$axis

with the *design matrix* taken as

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{u}(\boldsymbol{x}_1)^\top \\ \vdots \\ \boldsymbol{u}(\boldsymbol{x}_n)^\top \end{bmatrix}, \tag{2.24}$$

and the *scatter matrix* $\boldsymbol{S}$ given by $\boldsymbol{S} = \boldsymbol{U}^\top \boldsymbol{U}$.

The solution to the constrained problem, minimising $\boldsymbol{\theta}^\top \boldsymbol{S}\, \boldsymbol{\theta}$ subject to $\|\boldsymbol{\theta}\|^2 = 1$, may be determined via the method of *Lagrange multipliers* [2, §17.6]. This technique relies on that, at the minimiser, the gradient of the objective function is perpendicular to the surface described by the constraint. Thus, at the minimiser the gradient of the objective function and the gradient of the constraint function have the same direction, and so one is a scalar multiple of the other. This (*a priori* unknown) scalar multiple is the Lagrange multiplier and is denoted λ. Applied to the TLS estimator, the estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{TLS}}$ is characterised by

$$\Big[\partial_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^\top \boldsymbol{S}\, \boldsymbol{\theta}\big)\Big]_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}_{\mathrm{TLS}}} = \lambda \Big[\partial_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^\top \boldsymbol{\theta} - 1\big)\Big]_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}_{\mathrm{TLS}}}. \tag{2.25}$$

Differentiating with respect to $\boldsymbol{\theta}$ yields

$$2\boldsymbol{U}^\top \boldsymbol{U}\boldsymbol{\theta} = 2\lambda\boldsymbol{\theta}, \tag{2.26}$$

or equivalently

$$\boldsymbol{S}\boldsymbol{\theta} = \lambda\boldsymbol{\theta}. \tag{2.27}$$

Therefore the Lagrange multiplier λ is an eigenvalue of $\boldsymbol{S}$, and the required minimiser is the associated eigenvector. As $\boldsymbol{S}$ is an $l \times l$ matrix, there will be potentially l different eigenvalue/eigenvector pairs to choose from. So which to choose? By pre-multiplying Eq. (2.27) by $\boldsymbol{\theta}^\top$ and dividing by $\boldsymbol{\theta}^\top\boldsymbol{\theta}$,

$$\lambda = \frac{\boldsymbol{\theta}^\top \boldsymbol{S}\, \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{\theta}} = J_{\mathrm{TLS}}(\boldsymbol{\theta}). \tag{2.28}$$

As the eigenvalue is directly related to the cost function value, and the minimum value of J_{TLS} is required, the solution $\widehat{\boldsymbol{\theta}}_{\mathrm{TLS}}$ is the eigenvector associated with the *smallest* eigenvalue of $\boldsymbol{S}$.

The appropriate eigenvector of the scatter matrix $\boldsymbol{S}$ may be found by performing a *singular value decomposition* (SVD) of the design matrix $\boldsymbol{U}$. This is desirable as it improves the numerical conditioning: the scatter matrix has condition number given by the square of that of the design matrix. The SVD approach, using only the design matrix that is computed directly from the data, avoids the computation of the less conditioned scatter matrix.

The matrix $\boldsymbol{U}$ may be decomposed as

$$\boldsymbol{U} = \boldsymbol{J}\boldsymbol{D}\boldsymbol{K}^\top, \tag{2.29}$$

with $\boldsymbol{J}$ and $\boldsymbol{K}$ orthogonal ($\boldsymbol{J}^\top\boldsymbol{J} = \boldsymbol{K}^\top\boldsymbol{K} = \boldsymbol{I}$), and $\boldsymbol{D} = \mathrm{diag}(\sigma_1, \ldots, \sigma_l)$. The diagonal elements of $\boldsymbol{D}$ are called the *singular values* of $\boldsymbol{U}$.

The SVD and its usefulness in many applications (including in relation to least squares estimation) is described by Golub & van Loan [19, §2.5.3] and Kalman [27] among others. Practical information is given by Press *et al* in *Numerical Recipes* [52, §2.6] and Lancaster and Tismenetsky [36, §5.7].

Substituting the SVD form (Eq. (2.29)) for $\boldsymbol{S}$ gives

$$\boldsymbol{S} = \boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{K}\boldsymbol{D}^2\boldsymbol{K}^\top, \tag{2.30}$$

with $\boldsymbol{D}^2 = \boldsymbol{D}^\top\boldsymbol{D} = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2)$, the diagonal matrix with the squares of the singular values. Pre-multiplying by $\boldsymbol{K}^\top$ yields

$$\boldsymbol{K}^\top\boldsymbol{S} = \boldsymbol{D}^2\boldsymbol{K}^\top,$$

and as $\boldsymbol{S}$ is symmetric, then

$$(\boldsymbol{S}\boldsymbol{K})^\top = \boldsymbol{D}^2\boldsymbol{K}^\top.$$

Hence

$$\boldsymbol{S}\boldsymbol{K} = (\boldsymbol{D}^2\boldsymbol{K}^\top)^\top. \tag{2.31}$$

Denoting $\boldsymbol{k}_i$ the i^{th} column of $\boldsymbol{K}$, $\boldsymbol{K} = [\boldsymbol{k}_1 \mid \ldots \mid \boldsymbol{k}_l]$. In view of the diagonal nature of $\boldsymbol{D}$, Eq. (2.31) can be written as a system of equations

$$\boldsymbol{S}\boldsymbol{k}_i = \sigma_i^2\boldsymbol{k}_i, \qquad \text{for } \; i = 1, \ldots, l. \tag{2.32}$$

It follows that the TLS estimate is given by the column $\boldsymbol{k}_i$ associated with the smallest σ_i^2. If the SVD is defined such that the diagonal matrix $\boldsymbol{D}$ has the singular values in decreasing order (which is quite often the case), the estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{TLS}}$ can simply be taken as the right-most column $\boldsymbol{k}_l$.

2.4.3 Generalised TLS

Leedan and Meer [37] describe a *generalised total least squares* (GTLS) estimator that utilises a common covariance for the carriers $\boldsymbol{\Lambda_u}$. It is expressed as the minimiser of

$$J_{\mathrm{GTLS}}(\boldsymbol{\theta}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \frac{\boldsymbol{\theta}^\top \boldsymbol{S}\,\boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{\Lambda_u}\boldsymbol{\theta}}. \tag{2.33}$$

Like standard TLS, the estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{GTLS}}$ may also be expressed in closed form, in this case requiring a generalised eigendecomposition of $(\boldsymbol{S}, \boldsymbol{\Lambda_u})$. A short iterative algorithm is used to compute an estimate $\widehat{\boldsymbol{\Lambda}}_{\boldsymbol{u}}$ of $\boldsymbol{\Lambda_u}$. With $\widehat{\boldsymbol{\Lambda}}_{\boldsymbol{u}}$ replacing $\boldsymbol{\Lambda_u}$, the GTLS solution may be found directly as the generalised eigenvector corresponding to the smallest generalised eigenvalue of

$$\boldsymbol{S}\boldsymbol{\theta} = \lambda\widehat{\boldsymbol{\Lambda}}_{\boldsymbol{u}}\boldsymbol{\theta}.$$

2.4.4 Orthogonality and optimality

The "orthogonality" of the TLS method is not a straightforward issue. In the case of line fitting, shown in Fig. 2.3, the TLS method seeks an estimate which minimises the sum of orthogonal distances from the data points to the line.

In the presence of carriers, the TLS seeks to minimise the orthogonal distance between the transformed data points $\boldsymbol{u}(\boldsymbol{x}_1), \ldots, \boldsymbol{u}(\boldsymbol{x}_n)$ and candidate hyperplanes of the form $\{\boldsymbol{z} \mid \boldsymbol{\theta}^\top\boldsymbol{z} = 0\}$. For example, for parabola fitting (Sect. 2.3), the TLS estimate finds an estimate which is minimum in four dimensional algebraic space and not in the 2-D space in which the problem is posed (and data measured).

This problem is highlighted by further statistical analysis. The TLS method can be considered (in some sense) an optimal estimator given the assumption that the data points are identically and independently distributed [65]. In this case, however, the 'data' are the carrier vectors $\boldsymbol{u}_{\boldsymbol{x}_i}$, not the raw data vectors $\boldsymbol{x}_i$. Recall that each element of data is a random perturbation from an underlying true value, and that these perturbations are sampled from a Gaussian distribution:

$$\boldsymbol{x}_i = \overline{\boldsymbol{x}}_i + \Delta\boldsymbol{x}_i \qquad \Delta\boldsymbol{x}_i \sim N(\boldsymbol{0}, \boldsymbol{\Lambda}_{\boldsymbol{x}_i}).$$

Under this model, the noise effecting each element of data is independent of all of the other elements and of the exact value of the data point itself. However, the same is not true for the carrier vectors $\boldsymbol{u}_{\boldsymbol{x}_i}$. It can be shown that there are dependencies in each value of $\boldsymbol{u}_{\boldsymbol{x}_i}$ based on the value of $\boldsymbol{x}_i$.

Consider a first order Taylor expansion of $\boldsymbol{u}_{\boldsymbol{x}_i}$ around the true carrier $\boldsymbol{u}_{\overline{\boldsymbol{x}}_i}$ as follows

$$\boldsymbol{u}_{\boldsymbol{x}_i} \approx \boldsymbol{u}_{\overline{\boldsymbol{x}}_i} + \left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right](\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i), \tag{2.34}$$

with $\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})$ denoting the Jacobian of the carrier function. Using the above approximation, then

$$\boldsymbol{u}_{\boldsymbol{x}_i} - \boldsymbol{u}_{\overline{\boldsymbol{x}}_i} \approx \boldsymbol{u}_{\overline{\boldsymbol{x}}_i} + \left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right](\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i) - \boldsymbol{u}_{\overline{\boldsymbol{x}}_i} = \left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right](\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i). \tag{2.35}$$

The covariance of the carrier is given by

$$\begin{aligned}
\boldsymbol{\Lambda}_{\boldsymbol{u}_{\boldsymbol{x}_i}} &= \mathsf{E}\left[(\boldsymbol{u}_{\boldsymbol{x}_i} - \boldsymbol{u}_{\overline{\boldsymbol{x}}_i})(\boldsymbol{u}_{\boldsymbol{x}_i} - \boldsymbol{u}_{\overline{\boldsymbol{x}}_i})^\top\right] \\
&\approx \mathsf{E}\Big[\Big(\left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right](\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i)\Big)\Big(\left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right](\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i)\Big)^\top\Big] \\
&\approx \left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right]\mathsf{E}\Big[(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i)(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i)^\top\Big]\left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right]^\top \\
&\approx \left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right]\boldsymbol{\Lambda}_{\boldsymbol{x}_i}\left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right]^\top.
\end{aligned} \tag{2.36}$$

In the canonical case, with $\boldsymbol{\Lambda}_{\boldsymbol{x}_1} = \cdots = \boldsymbol{\Lambda}_{\boldsymbol{x}_n} = \boldsymbol{I}$, the covariances of the carriers will have explicit data dependencies, because

$$\boldsymbol{\Lambda}_{\boldsymbol{u}_{\boldsymbol{x}_i}} = \left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right]\left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right]^\top \neq \boldsymbol{I}.$$

In the case of the parabola fitting example from Sec. 2.3, the Jacobian is

$$\left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})\right] = \begin{bmatrix} 2x & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix},$$

and therefore the covariance of the carrier is

$$\mathbf{\Lambda}_{\boldsymbol{u}_{\boldsymbol{x}_i}} = \begin{bmatrix} 4x^2 & 2x & 0 & 0 \\ 2x & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

It is clear that the distribution of each $\boldsymbol{u}_{\boldsymbol{x}_i}$ is not identical, but rather data dependent. Using a second order Taylor expansion, it can be shown that the expected value of $\boldsymbol{u}_{\boldsymbol{x}_i}$ does not vanish, but rather has a non-zero value which is dependent on the data [41]. This means that the underlying assumption of the TLS method, to the effect that data are independent, is not true in the case of problems requiring non-linear carriers. Of course the TLS algorithm may still be applied, but optimality will be lost.

An alternative method may be adopted where a cost function is defined using a more geometrically meaningful quantity to minimise than algebraic residuals. The key is to consider orthogonal distances in the original data space, appealing to the geometry of the problem at hand.

2.5 Maximum likelihood

A classical statistical estimation method is maximum likelihood (ML). It is well studied and in many cases may be proved optimal in some sense. In this section an ML estimator is derived suitable for the problem specified in earlier sections. A likelihood function may be determined which specifies the likelihood of a set of model parameters given the observed data, and the ML estimate is one which maximises this likelihood function. In practice, once the likelihood function is formed its logarithm (times -1) is taken and then minimised (as this is often simpler).

2.5.1 ML cost function

Maximum likelihood theory when applied to computer vision departs from classical ML (such as that studied in [35]) because no *repeated measurements* of a random variable are available. Rather, the necessary model requires a set of random variables, each for any particular element of data. We assume that each datum $\boldsymbol{x}_i$ is a sample of a Gaussian random variable X_i with mean $\overline{\boldsymbol{x}}_i$ and covariance $\mathbf{\Lambda}_{\boldsymbol{x}_i}$. Further adding to the model is

a relationship between each $\overline{\boldsymbol{x}}_i$ so that the principal constraint Eq. (2.2) is satisfied

$$\boldsymbol{\theta}^\top \boldsymbol{u}(\overline{\boldsymbol{x}}_1) = \ldots = \boldsymbol{\theta}^\top \boldsymbol{u}(\overline{\boldsymbol{x}}_n) = 0, \tag{2.37}$$

for some particular $\boldsymbol{\theta}$. All of the model parameters may be combined in a larger state vector

$$\boldsymbol{\eta} = (\boldsymbol{\theta}; \overline{\boldsymbol{x}}_1, \ldots, \overline{\boldsymbol{x}}_n; \boldsymbol{\Lambda}_{\boldsymbol{x}_1}, \ldots, \boldsymbol{\Lambda}_{\boldsymbol{x}_n}) \tag{2.38}$$

The model parameters are 'driven' in a sense by the choice of the X_i. The observed data, under this model, will have come from some underlying true set of random variables X_i^*, but of course these may never be known exactly. What is required is a means to *estimate* them as accurately as possible. The ML philosophy is to "choose the parameters which most likely account for the measured data". So the ML estimate is taken as the maximiser of the conditional probability

$$\widehat{\boldsymbol{\eta}}_{\mathrm{ML}} = \max_{\boldsymbol{\eta}} p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n | \, \boldsymbol{\eta}). \tag{2.39}$$

Each data point is a sample of a Gaussian random variable, having a probability density function

$$p(\boldsymbol{x}_i \mid \boldsymbol{\eta}) = \frac{1}{\sqrt{(2\pi)^{\mathrm{k}} |\boldsymbol{\Lambda}_{\boldsymbol{x}_i}|}} \exp\left(-\frac{1}{2} (\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i)^\top \boldsymbol{\Lambda}_{\boldsymbol{x}_i}^{-1} (\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i) \right), \tag{2.40}$$

and as these are separate random variables, each p.d.f. is independent of the others, therefore

$$\begin{aligned} p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \mid \boldsymbol{\eta}) &= \prod_{i=1}^{n} p(\boldsymbol{x}_i \mid \boldsymbol{\eta}) \\ &= (2\pi)^{-kn/2} \prod_{i=1}^{n} |\boldsymbol{\Lambda}_{\boldsymbol{x}_i}|^{-1/2} \times \exp\left\{ -\frac{1}{2} \sum_{i=1}^{n} (\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i)^\top \boldsymbol{\Lambda}_{\boldsymbol{x}_i}^{-1} (\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i) \right\}. \end{aligned} \tag{2.41}$$

In order to enhance the tractability of the model, the covariance matrices of each X_i are not treated as unknown parameters. Instead, they are assumed to be known *a priori*, and may be considered an *input* to the estimation process together with noisy data points. The topic of covariance matrices, and their use as input, is explored further in Chapter 6. Because covariances are no longer part of $\boldsymbol{\eta}$, then the coefficient in front of the exponent in Eq. (2.40) is not a function of $\boldsymbol{\eta}$ and may be ignored in maximisation. Equivalent to maximising the likelihood is minimising the log-likelihood. Therefore the ML cost function, given by

$$J_{\mathrm{ML}}(\boldsymbol{\eta}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \sum_{i=1}^{n} (\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i)^\top \boldsymbol{\Lambda}_{\boldsymbol{x}_i}^{-1} (\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i), \tag{2.42}$$

is the sum of squares of *Mahalanobis distances* between each data point and the closest ideal point.

2.5.2 Nuisance parameters

The ML cost function is unwieldy because $\boldsymbol{\eta}$ increases in size for every new data point added. Each new measurement $\boldsymbol{x}_i$ requires a corresponding $\overline{\boldsymbol{x}}_i$ to be added to the parameter vector. The model parameters may be thought of as comprising *principal parameters* $\boldsymbol{\theta}$, which are of ultimate interest, and *nuisance parameters* $(\overline{\boldsymbol{x}}_1, \ldots, \overline{\boldsymbol{x}}_n)$, only of intermediate use and not required as final output. The minimisation of J_{ML} is at best a complex problem, at worst intractable. It must be considered with a specific knowledge of the problem at hand. A serious stumbling block is the large space of the parameters due to the presence of nuisance parameters.

As highlighted in Eq. (2.37), each $\overline{\boldsymbol{x}}_i$ is not arbitrary, but tied to a particular value of $\boldsymbol{\theta}$. It is therefore possible to contemplate splitting the minimisation of J_{ML} into two stages. One, in which for each fixed $\boldsymbol{\theta}$ the $\overline{\boldsymbol{x}}_i$ are subject to minimisation alone. And the other, where the values of J_{ML} at the minimisers from the first step are minimised over varying $\boldsymbol{\theta}$. These stages are shown in Fig. 2.4.

An alternative approach is to eliminate the nuisance parameters from the cost function, and hence minimisation may proceeded directly over values of $\boldsymbol{\theta}$.

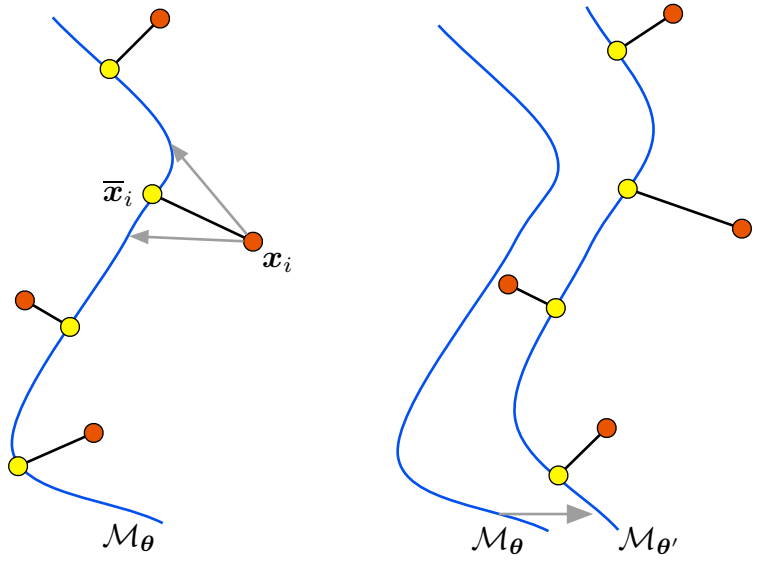


Figure 2.4: Minimising J_{ML} as a two stage process

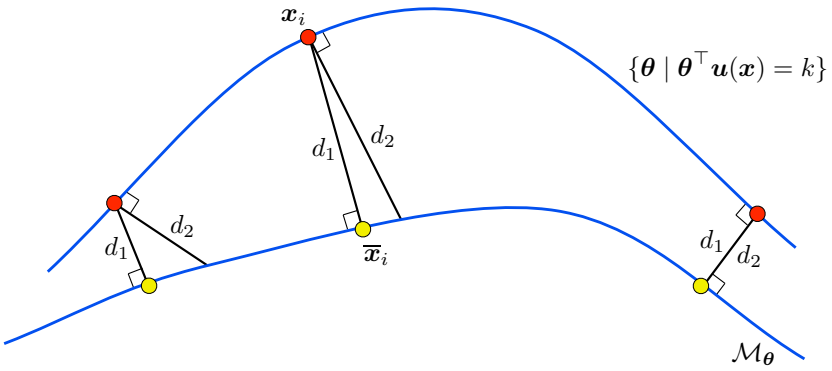


Figure 2.5: Sampson's approximated distance d_2 and the true ML distance d_1

2.6 Sampson's approximation

One of the disadvantages of the ML cost function is that the number of parameters increases as the number of data points grows larger. Although it is possible to form an estimator using the full ML cost function, it must be done in a problem dependent manner. The high dimensionality of the minimisation required by this so-called *direct approach* means that, at the very least, finding the minimiser of the ML cost function is computationally very expensive.

An alternative tactic is to adopt a so-called *reduced approach*. Here, a new cost function is used which is expressed only in terms of the actual parameters to be estimated. In deriving such a cost function, it is necessary to introduce approximations.

To derive an *approximated maximum likelihood* (AML) cost function, consider a first order Taylor expansion of the principal constraint (Eq. (2.2)) about $\overline{\boldsymbol{x}}$:

$$\begin{aligned} f(\boldsymbol{x}, \boldsymbol{\theta}) &\simeq f(\overline{\boldsymbol{x}}, \boldsymbol{\theta}) + \left[\partial_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{\theta})\right] (\boldsymbol{x} - \overline{\boldsymbol{x}}) \\ &= f(\overline{\boldsymbol{x}}, \boldsymbol{\theta}) + \|\boldsymbol{x} - \overline{\boldsymbol{x}}\| \, \| \left[\partial_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{\theta})\right] \| \cos\alpha \end{aligned} \tag{2.43}$$

where α is the angle between the vectors $(\boldsymbol{x} - \overline{\boldsymbol{x}})$ and $\left[\partial_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{\theta})\right]^\top$. Sampson's assumption is that the gradient is the same at $\boldsymbol{x}$ and $\overline{\boldsymbol{x}}$, which means that $\alpha = 0$ (see Fig. 2.5). By definition, $f(\overline{\boldsymbol{x}}, \boldsymbol{\theta}) = 0$, which leads to a first order approximation to the geometric residual

$$\|\boldsymbol{x} - \overline{\boldsymbol{x}}\| \simeq \frac{f(\boldsymbol{x}, \boldsymbol{\theta})}{\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{\theta})\|} \,. \tag{2.44}$$

Using this idea, an AML cost function can be fleshed out in more detail. In the next section, an AML cost function is derived which incorporates data covariance matrices.

2.6.1 Derivation of a reduced cost function

The ML cost function, although not directly operating on $\boldsymbol{\theta}$, may be written as $J_{\mathrm{ML}}(\boldsymbol{\theta}) = \sum_i J^i_{\mathrm{ML}}$, with $J^i_{\mathrm{ML}} = (\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i)^\top \boldsymbol{\Lambda}^{-1}_{\boldsymbol{x}_i}(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i)$. The function is to be minimised subject to the system of constraints $\boldsymbol{\theta}^\top \boldsymbol{u}(\overline{\boldsymbol{x}}_i) = 0$. Using a Lagrange multiplier approach, the gradients of these two functions are proportional:

$$\begin{aligned}\left[\partial_{\overline{\boldsymbol{x}}} J^i_{\mathrm{ML}}(\boldsymbol{\theta})\right]^\top &= \lambda_i \left[\partial_{\boldsymbol{x}}(\boldsymbol{\theta}^\top \boldsymbol{u}(\overline{\boldsymbol{x}}_i))\right]^\top \\ &= \lambda_i \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}}_i)\right]^\top \boldsymbol{\theta}. \end{aligned} \tag{2.45}$$

The function J^i_{ML} is easily differentiated, with

$$\partial_{\boldsymbol{y}}[(\boldsymbol{x}_i - \boldsymbol{y})^\top \boldsymbol{\Lambda}^{-1}_{\boldsymbol{x}_i}(\boldsymbol{x}_i - \boldsymbol{y})]^\top = -2\boldsymbol{\Lambda}^{-1}(\boldsymbol{x} - \boldsymbol{y}).$$

So Eq. (2.45) may be rewritten

$$\boldsymbol{\Lambda}^{-1}(\boldsymbol{x} - \overline{\boldsymbol{x}}) = \lambda \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right]^\top \boldsymbol{\theta}, \tag{2.46}$$

with the i indices omitted, the covariance matrix $\boldsymbol{\Lambda}_{\boldsymbol{x}_i}$ contracted to $\boldsymbol{\Lambda}$ for clarity, and the -2 factor absorbed into a 'new' λ. Multiplying both sides by $\boldsymbol{\Lambda}^{\frac{1}{2}}$ gives

$$\boldsymbol{\Lambda}^{-\frac{1}{2}}(\boldsymbol{x} - \overline{\boldsymbol{x}}) = \lambda \boldsymbol{\Lambda}^{\frac{1}{2}} \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right]^\top \boldsymbol{\theta}.$$

Note that $\boldsymbol{\Lambda}^{-\frac{1}{2}} = \boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{\Lambda}^{-1}$, the inverse of $\boldsymbol{\Lambda}^{\frac{1}{2}}$. Solving for λ gives

$$\lambda = \frac{\boldsymbol{\Lambda}^{-\frac{1}{2}}(\boldsymbol{x} - \overline{\boldsymbol{x}})}{\boldsymbol{\Lambda}^{\frac{1}{2}} \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right]^\top \boldsymbol{\theta}},$$

and therefore

$$\lambda^2 = \frac{(\boldsymbol{x} - \overline{\boldsymbol{x}})^\top \boldsymbol{\Lambda}^{-1}(\boldsymbol{x} - \overline{\boldsymbol{x}})}{\boldsymbol{\theta}^\top \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right] \boldsymbol{\Lambda} \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right]^\top \boldsymbol{\theta}}. \tag{2.47}$$

Returning to Eq. (2.46), this time multiplying both sides by $(\boldsymbol{x} - \overline{\boldsymbol{x}})^\top$ gives

$$(\boldsymbol{x} - \overline{\boldsymbol{x}})^\top \boldsymbol{\Lambda}^{-1}(\boldsymbol{x} - \overline{\boldsymbol{x}}) = \lambda (\boldsymbol{x} - \overline{\boldsymbol{x}})^\top \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right]^\top \boldsymbol{\theta}. \tag{2.48}$$

To proceed with a derivation of a reduced system, an approximation must be employed. Consider a first-order Taylor expansion of $\boldsymbol{u}(\boldsymbol{x}) - \boldsymbol{u}(\overline{\boldsymbol{x}})$ about $\overline{\boldsymbol{x}}$,

$$\boldsymbol{u}(\boldsymbol{x}) - \boldsymbol{u}(\overline{\boldsymbol{x}}) \approx \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right](\boldsymbol{x} - \overline{\boldsymbol{x}}).$$

By adopting this approximation as an equality, it is possible to proceed with the derivation. Pre-multiplying both sides by $\boldsymbol{\theta}$ yields the equation

$$\begin{aligned}\boldsymbol{\theta}^\top \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right](\boldsymbol{x}-\overline{\boldsymbol{x}}) &= \boldsymbol{\theta}^\top(\boldsymbol{u}(\boldsymbol{x})-\boldsymbol{u}(\overline{\boldsymbol{x}})) \\ &= \boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}) - \boldsymbol{\theta}^\top \boldsymbol{u}(\overline{\boldsymbol{x}}),\end{aligned}$$

as, by definition, $\boldsymbol{\theta}^\top \boldsymbol{u}(\overline{\boldsymbol{x}}) = 0$, the above reduces to

$$\boldsymbol{\theta}^\top \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right](\boldsymbol{x}-\overline{\boldsymbol{x}}) = \boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}). \tag{2.49}$$

Substituting this result into Eq. (2.48) leaves

$$(\boldsymbol{x}-\overline{\boldsymbol{x}})^\top \boldsymbol{\Lambda}^{-1}(\boldsymbol{x}-\overline{\boldsymbol{x}}) = \lambda \boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}).$$

Squaring both sides,

$$\left[(\boldsymbol{x}-\overline{\boldsymbol{x}})^\top \boldsymbol{\Lambda}^{-1}(\boldsymbol{x}-\overline{\boldsymbol{x}})\right]^2 = \lambda^2 \boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x})\boldsymbol{u}(\boldsymbol{x})^\top \boldsymbol{\theta},$$

and substituting for λ^2 from Eq. (2.47), then

$$\left[(\boldsymbol{x}-\overline{\boldsymbol{x}})^\top \boldsymbol{\Lambda}^{-1}(\boldsymbol{x}-\overline{\boldsymbol{x}})\right]^2 = \frac{(\boldsymbol{x}-\overline{\boldsymbol{x}})^\top \boldsymbol{\Lambda}^{-1}(\boldsymbol{x}-\overline{\boldsymbol{x}})\, \boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x})\boldsymbol{u}(\boldsymbol{x})^\top \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right] \boldsymbol{\Lambda} \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}})\right]^\top \boldsymbol{\theta}}.$$

On dividing both sides by $(\boldsymbol{x}-\overline{\boldsymbol{x}})^\top \boldsymbol{\Lambda}^{-1}(\boldsymbol{x}-\overline{\boldsymbol{x}})$, the final approximated cost function is obtained in the form

$$J^i_{\mathrm{ML}}(\boldsymbol{\eta}) = (\boldsymbol{x}_i-\overline{\boldsymbol{x}}_i)^\top \boldsymbol{\Lambda}^{-1}_{\boldsymbol{x}_i}(\boldsymbol{x}_i-\overline{\boldsymbol{x}}_i) = \frac{\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}_i)\boldsymbol{u}(\boldsymbol{x}_i)^\top \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}}_i)\right] \boldsymbol{\Lambda}_{\boldsymbol{x}_i} \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}}_i)\right]^\top \boldsymbol{\theta}}.$$

The second approximation employed above replaces the $\partial_{\boldsymbol{x}} \boldsymbol{u}(\overline{\boldsymbol{x}}_i)$ gradients at the (unknown) true point, with $\partial_{\boldsymbol{x}} \boldsymbol{u}(\boldsymbol{x}_i)$, the gradients around the measured points, which are easily computed.

This leads to a reduced cost function

$$J_{\mathrm{AML}}(\boldsymbol{\theta}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \sum_{i=1}^{n} \frac{\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}_i)\boldsymbol{u}(\boldsymbol{x}_i)^\top \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\boldsymbol{x}_i)\right] \boldsymbol{\Lambda}_{\boldsymbol{x}_i} \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\boldsymbol{x}_i)\right]^\top \boldsymbol{\theta}}. \tag{2.50}$$

This cost function may be written more compactly as

$$J_{\mathrm{AML}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \frac{\boldsymbol{\theta}^\top \boldsymbol{A}_i\, \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{B}_i\, \boldsymbol{\theta}}, \tag{2.51}$$

with $\boldsymbol{A}_i = \boldsymbol{u}(\boldsymbol{x}_i)\boldsymbol{u}(\boldsymbol{x}_i)^\top$ and $\boldsymbol{B}_i = \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\boldsymbol{x}_i)\right] \boldsymbol{\Lambda}_{\boldsymbol{x}_i} \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\boldsymbol{x}_i)\right]^\top$.

The minimiser of J_{AML} is called the *approximated maximum likelihood* (AML) estimate and is denoted $\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}$.

2.6.2 Alternative derivation of the approximated ML cost function

An alternative derivation of J_{AML} relates to the non uniform distribution of the carriers $\boldsymbol{u}_{\boldsymbol{x}}$. As shown in Sect. 2.4.4, algebraic based estimators are unsatisfactory because the carriers are not identically distributed and, as a result, the covariances of algebraic residuals are data dependent. A possible method of correction therefore is to divide the algebraic residuals by their respective covariances in an attempt to provide an even weighting for each residual. Doing this leads to a gradient weighted (GW) cost function

$$J_{\mathrm{GW}}(\boldsymbol{\theta}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \sum_{i=1}^{n} \frac{\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}_i) \boldsymbol{u}(\boldsymbol{x}_i)^\top \boldsymbol{\theta}}{\mathrm{Cov}[\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}_i)]}. \tag{2.52}$$

Using the form of the carrier covariance from Eq. (2.36), the above generic cost function may be expressed as

$$\begin{aligned} J_{\mathrm{GW}}(\boldsymbol{\theta}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) &= \sum_{i=1}^{n} \frac{\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}_i) \boldsymbol{u}(\boldsymbol{x}_i)^\top \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{\Lambda}_{\boldsymbol{u}(\boldsymbol{x}_i)} \boldsymbol{\theta}} \\ &= \sum_{i=1}^{n} \frac{\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}_i) \boldsymbol{u}(\boldsymbol{x}_i)^\top \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\boldsymbol{x}_i)\right] \boldsymbol{\Lambda}_{\boldsymbol{x}_i} \left[\partial_{\boldsymbol{x}} \boldsymbol{u}(\boldsymbol{x}_i)\right]^\top \boldsymbol{\theta}}. \end{aligned} \tag{2.53}$$

This form is precisely the same as that of J_{AML} given in Eq. (2.50). The equivalence of the two forms is not coincidental given that both methods employ a Taylor expansion in their derivation.

This dual motivation, leading to equivalent cost functions, means that the terms "gradient weighted" and "approximated maximum likelihood" in this case describe identical cost functions.

This approach yields a cost function which is determined in terms of the necessary parameters only. Its advantage is that the form is generic and applies to any problem which conforms to the original parameter model specification. It is possible, however, to consider the issue of eliminating nuisance parameters by considering the specific form of an estimation problem at hand. In the case of optical flow estimation, for example, Ohta [48] shows how an estimator can be built which eliminates nuisance parameters by assuming they conform to a particular statistical distribution.

Unlike the estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{TLS}}$ (Sect. 2.4.2), the estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}$ does not admit a closed-form expression. Hence, an iterative algorithm must be employed to minimise J_{AML}, obviously involving greater complexity and computational cost. Several algorithms are presented in the next chapter which aim to minimise J_{AML}. They take advantage of

the specific form of the cost function and, because of this, offer significant efficiency advantages over general minimisation techniques.

Chapter 3

COST FUNCTION MINIMISATION

This chapter presents methods for finding the minimiser of a cost function. General methods capable of working with more or less arbitrary cost functions are described. Next, to account for the special nature of the model defined in the previous chapter more specific methods are derived which act specifically on J_{AML}. In the first instance, the ancillary constraint is not considered: the methods aim to find a minimiser with no restriction on the parameter space. This deficiency is addressed in a subsequent section where a constrained method is derived.

3.1 General methods

A global minimiser $\widehat{\boldsymbol{\theta}}_g$ of a function $J(\boldsymbol{\theta})$ will satisfy $J(\widehat{\boldsymbol{\theta}}_g) \leq J(\boldsymbol{\theta})$ for all values of $\boldsymbol{\theta}$. Finding such a global minimiser is often an arduous task, depending heavily on the specific form of the cost function. A realistic, tractable alternative often employed is to find a *local* minimum $\widehat{\boldsymbol{\theta}}_l$. Giving an initial starting point $\boldsymbol{\theta}_0$, the local minima satisfies

$$J(\widehat{\boldsymbol{\theta}}_l) \leq J(\boldsymbol{\theta}) \tag{3.1}$$

for all values of $\boldsymbol{\theta}$ in a *neighbourhood* of $\widehat{\boldsymbol{\theta}}_l$ that contains $\boldsymbol{\theta}_0$. The initial estimate $\boldsymbol{\theta}_0$ allows the reduction of the search space to a small region around $\boldsymbol{\theta}_0$.

There are many strategies for computing minimisers of cost functions. These include the so-called Monte Carlo methods, such as simulated annealing and genetic programming [18, §13.3 & §13.4] that are robust to deal with unstable or very 'rough' cost functions. They typically involve significant overhead. In this thesis such methods will not be considered, rather iterative methods which are geared for finding local minimisers of more tolerable cost functions will be pursued.

It should be noted that, for a general non-linear cost function, some form of iterative method is required. Only a very restricted class of cost functions, generally deriving from models in which the parameters and data interact linearly, may have their minimisers computed directly. The TLS method is one such non-iterative method, in the

sense that the solution is computed directly by performing a single SVD of the design matrix. As highlighted in Sect. 2.4.4, the TLS method is only optimal for purely linear situations.

An iterative optimisation algorithm searching for a local minimum seeks to produce, given a current guess $\boldsymbol{\theta}_c$ of the solution, a "better" guess $\boldsymbol{\theta}_+$. Of course, there must also be some mechanism whereby the iterations are stopped, and the final $\boldsymbol{\theta}_+$ is taken as the minimum. Such stopping conditions will be discussed after the different algorithms have been presented. Different schemes for computing an updated estimate of the solution can be divided into two categories: those which make use of the gradient of the cost function, and those that do not.

3.1.1 Direct search

Direct search methods do not utilise the gradient of the cost function. They are useful when the cost function is not smooth and potentially contains discontinuities, or when the cost function is not readily differentiable or computing numerical gradients is considered too computationally expensive. One such example may be the case where $J(\boldsymbol{\theta})$ itself is the result of a minimisation.

A widely used direct search method is *Nelder-Mead* [45], also known as *downhill simplex*, or the *Amoeba method* [52, §10.4]. This method works by employing a simplex for which the cost function is evaluated at each vertex. A series of rules are specified as to how to update the vertices of the simplex at each iteration.

The main deficiency of the Nelder-Mead algorithm is that it has poor convergence properties [43]. Other problems include the tendency to get 'stuck' too easily in non-optimal (local) minima, and the need for many more iterations than gradient based methods.

3.1.2 Gradient-based methods

When the first derivative of the cost function is known *a priori*, or can be calculated numerically, then this gradient information may be used by so-called gradient based methods.

A first order gradient method is *gradient descent* (sometimes known as *steepest descent*). Quite simply this method specifies that a step should be taken along the steepest

gradient at the current estimate. So, given a current guess $\boldsymbol{\theta}_c$, the next guess is

$$\boldsymbol{\theta}_+ = \boldsymbol{\theta}_c - k \left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_c)\right]^\top , \tag{3.2}$$

with k a (relatively small) constant. To analyse how $\boldsymbol{\theta}_+$ improves on $\boldsymbol{\theta}_c$, consider a first order Taylor expansion of $J(\boldsymbol{\theta})$ around $\boldsymbol{\theta}_c$:

$$J(\boldsymbol{\theta}_+) \approx J(\boldsymbol{\theta}_c) + (\boldsymbol{\theta}_+ - \boldsymbol{\theta}_c) \left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_c)\right] . \tag{3.3}$$

Substituting the updated $\boldsymbol{\theta}_+$ from Eq. (3.2) gives a new value of the cost function

$$J(\boldsymbol{\theta}_+) = J(\boldsymbol{\theta}_c) - k \| \left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta})\right]_{\boldsymbol{\theta}_c} \|^2, \tag{3.4}$$

which is an improvement, with $J(\boldsymbol{\theta}_+) < J(\boldsymbol{\theta}_c)$, as long as k is small enough to make the Taylor approximation meaningful. In general, the gradient descent method does not have very good convergence properties, even in the case of a smooth and well behaved cost function [34, §3.2.2].

Gauss-Newton, a second-order method, is another method which incorporates gradients. The cost function is approximated with a second order Taylor expansion

$$J(\boldsymbol{\theta}_+) = J(\boldsymbol{\theta}_c) + (\boldsymbol{\theta}_+ - \boldsymbol{\theta}_c) \left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_c)\right] + (\boldsymbol{\theta}_+ - \boldsymbol{\theta}_c)^\top \boldsymbol{H}_J \, (\boldsymbol{\theta}_+ - \boldsymbol{\theta}_c), \tag{3.5}$$

where $\boldsymbol{H}_J$ is the Hessian matrix of the cost function J evaluated at $\boldsymbol{\theta}_c$. Differentiating with respect to $\boldsymbol{\theta}_+$ gives

$$\left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_+)\right] = \left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_c)\right] + \boldsymbol{H}_J(\boldsymbol{\theta}_+ - \boldsymbol{\theta}_c). \tag{3.6}$$

The necessary condition of the minimiser is that the gradient of the cost function should vanish at that point. So, substituting $\left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_+)\right] = \mathbf{0}$ into the above equation yields the identity

$$(\boldsymbol{\theta}_+ - \boldsymbol{\theta}_c) = \boldsymbol{H}_J^{-1} \left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_c)\right] . \tag{3.7}$$

The update $\boldsymbol{\theta}_+$ is then given by $\boldsymbol{\theta}_+ = \boldsymbol{\theta}_c + \boldsymbol{H}_J^{-1} \left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_c)\right]$. For such a step to make sense the Hessian must be positive semi-definite, and unfortunately this is not always the case. In fact, it is quite common for the Hessian to be rank deficient at points close to the minimiser. A number of variations on the Gauss-Newton method, known as quasi-Newton methods, use more sophisticated expressions in place of $\boldsymbol{H}_J$ in Eq. (3.7) [52]. These clever constructions of replacements for $\boldsymbol{H}_J$ permit avoiding problems with singularities.

Both first order and second order gradient methods are almost always preferable to direct searching as long as gradient information is readily available. The performance of the Gauss-Newton methods depends on how well the surface of the cost function is approximated by its second-order Taylor expansion (Eq. (3.5)).

3.1.3 Least squares

An important special case of optimisation problems is the least squares cost function of the form

$$J(\boldsymbol{\theta}) = \frac{1}{2}\sum_i r_i^2(\boldsymbol{\theta}) = \frac{\boldsymbol{r}(\boldsymbol{\theta})^\top \boldsymbol{r}(\boldsymbol{\theta})}{2}\,, \tag{3.8}$$

where $\boldsymbol{r} = [r_1(\boldsymbol{\theta}), \ldots, r_n(\boldsymbol{\theta})]^\top$ is a vector of residuals. This cost function may be minimised using second order gradient methods because it is possible to analytically derive the Jacobian and Hessian of the cost function J. From Eq. (3.8), the Jacobian is

$$[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta})] = \boldsymbol{r}(\boldsymbol{\theta})^\top \left[\partial_{\boldsymbol{\theta}} \boldsymbol{r}(\boldsymbol{\theta})\right]. \tag{3.9}$$

Note that the term $[\partial_{\boldsymbol{\theta}} \boldsymbol{r}(\boldsymbol{\theta})]$ is a Jacobian *matrix* (as $\boldsymbol{r}$ is itself a vector), as opposed to $[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta})]$ which is a *vector*. Differentiating again, the Hessian is

$$\boldsymbol{H}_J = \left[\partial^2_{\boldsymbol{\theta}\boldsymbol{\theta}} J(\boldsymbol{\theta})\right] = \left[\partial_{\boldsymbol{\theta}} \boldsymbol{r}(\boldsymbol{\theta})\right]\left[\partial_{\boldsymbol{\theta}} \boldsymbol{r}(\boldsymbol{\theta})\right]^\top + \sum_i r_i(\boldsymbol{\theta}) \boldsymbol{H}_{r_i}. \tag{3.10}$$

The right-most second order term is usually discarded from the form of the Hessian, and a simplified version

$$\boldsymbol{H}_J \approx \left[\partial_{\boldsymbol{\theta}} \boldsymbol{r}(\boldsymbol{\theta})\right]\left[\partial_{\boldsymbol{\theta}} \boldsymbol{r}(\boldsymbol{\theta})\right]^\top \tag{3.11}$$

is used. Apart from being expensive to compute, the second order term may in fact cause de-stabilisation in subsequent minimisation [52, §15.5].

The least squares problem may be solved using gradient based methods described above. For gradient descent, the Jacobian of Eq. (3.9) is used, and for Gauss-Newton, the Hessian of Eq. (3.11) is taken. Because of the poor convergence properties of the gradient descent method, the Gauss-Newton method is preferable except when the solution of Eq. (3.7) yields an updated estimate which increases the cost function value. If this happens, gradient descent can be employed ensuring that the cost function value will decrease. A sophisticated algorithm which can dynamically "swap" between these two modes of operation is the Levenberg-Marquardt method.

Levenberg-Marquardt

The *Levenberg-Marquardt* method (LM) is a hybrid of the Gauss-Newton inverse Hessian method and the method of gradient descent. It aims to include the best attributes of both methods by favouring Gauss-Newton when the Hessian is reliable enough to facilitate rapid convergence, and switching to gradient descent when the Gauss-Newton method might go astray. The underlying insight is to replace the ordinary Hessian matrix used in step Eq. (3.11), adding an extra diagonal weight, yielding

$$(\boldsymbol{\theta}_c - \boldsymbol{\theta}_+) = (\boldsymbol{H}_J + \upsilon \boldsymbol{I})^{-1} \left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_c)\right]. \tag{3.12}$$

The parameter υ controls the behaviour of the update. When υ is very small, the RHS of the equation reduces to the inverse Hessian of the Gauss-Newton. However, when υ is large, then the identity matrix causes the diagonal elements to dominate $\boldsymbol{H}_J$, and the equation tends to

$$(\boldsymbol{\theta}_c - \boldsymbol{\theta}_+) = \upsilon \boldsymbol{I} \left[\partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_c)\right], \tag{3.13}$$

which is precisely the gradient descent update of Eq. (3.2). Therefore, by varying υ before an iteration, this method can in essence 'switch' between descent and Gauss-Newton.

The main 'tweaking' in the L-M algorithm is to specify how υ is updated from iteration to iteration. The basic form of the L-M algorithm is given in Alg. 3.1, which is an outline of a commonly used update method [52]. The Gauss-Newton step is deemed to be useful when successive estimates are produced with a lower cost function value. While this is happening, the factor υ is reduced more and more, causing the algorithm to rely increasingly on the Hessian information and become more like Gauss-Newton. When successive estimates produce higher values of the cost function, then the Hessian information is deemed to be less reliable, and υ is increased. This causes the algorithm to fall back to taking steps more resembling gradient descent.

In Alg. 3.1, a simple multiply- and divide-by ten rule is used to increase and decrease υ. Other, more complicated, schemes for updating υ exist that rely, for example, on polynomial interpolation of each $r_i(\boldsymbol{\theta})$ [26, p. 2–24].

Levenberg-Marquardt is widely regarded as the best optimisation method to use for least squares-type problems. It is used extensively in computer vision [23, App. 4]

Algorithm 3.1 (L-M) *Computes the minimiser $\widehat{\boldsymbol{\theta}}_{\mathrm{LM}}$ of a least squares cost function $J(\boldsymbol{\theta}) = \frac{1}{2}\,\boldsymbol{r}(\boldsymbol{\theta})^\top \boldsymbol{r}(\boldsymbol{\theta})$ using the Levenberg-Marquardt method.*

1. *Choose an initial estimate $\boldsymbol{\theta}_0$ and set υ to a small value, typically $k = 10^{-3}$.*
2. *Using $\boldsymbol{\theta}_c$, compute the Hessian $\boldsymbol{H}_J$ as per Eq. (3.11), and hence solve Eq.(3.12) yielding $\boldsymbol{\theta}_+$.*
3. *Find the cost function value at $\boldsymbol{\theta}_+$: $J(\boldsymbol{\theta}_+) = \frac{1}{2}\,\boldsymbol{r}(\boldsymbol{\theta}_+)^\top \boldsymbol{r}(\boldsymbol{\theta}_+)$.*
4. *If $J(\boldsymbol{\theta}_+) < J(\boldsymbol{\theta}_c)$, then update $\upsilon = \upsilon/10$, and set $\boldsymbol{\theta}_c = \boldsymbol{\theta}+$.*
5. *Otherwise, retain $\boldsymbol{\theta}_c$ and set υ to 10υ.*
6. *If the decrease in the cost function falls below a threshold, then terminate and take $\widehat{\boldsymbol{\theta}}_{\mathrm{LM}} = \boldsymbol{\theta}_+$; otherwise go to step 2.*

Minimising J_{AML} using L-M

It is straightforward to use the L-M algorithm to find a minimiser of $J_{\mathrm{AML}}(\boldsymbol{\theta}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$. All that is necessary is to represent J_{AML} as a sum of squared residuals. This is achieved by taking

$$\boldsymbol{r}(\boldsymbol{\theta}) = \left[\sqrt{\frac{\boldsymbol{\theta}^\top \boldsymbol{A}_1\, \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{B}_1\, \boldsymbol{\theta}}}, \ldots, \sqrt{\frac{\boldsymbol{\theta}^\top \boldsymbol{A}_n\, \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{B}_n\, \boldsymbol{\theta}}} \right]^\top . \tag{3.14}$$

3.2 AML Specific methods

Each minimisation method described above applies to increasingly specialised cost functions. Indeed, the direct search methods require only function evaluations, but gradient based methods require the additional calculation of the Jacobian, and potentially the Hessian. Further still, least squares methods require the cost function to be of the form as per Eq. (3.8). Cost functions of more special form are amenable to better optimisation methods in terms of convergence, accuracy, and tolerance of poor starting initial solutions.

The process of refining the optimisation methods can be extended to cope specif-

ically with J_{AML}. The LM method is generally favoured when applied to the general least squares problems. But methods even more specific, described in this section, can have extra advantages. Usually it is increased efficiency – reducing the computation time required to find an estimate – but also may be the reduced sensitivity to the initial seed, and better convergence.

3.2.1 Eigenfit: Taubin's meta-approximation

Taubin [59] proposed an approximation to the ML cost function (with identity covariances, $\boldsymbol{\Lambda}_{\boldsymbol{x}_i} = \boldsymbol{I}$)

$$J^*_{\mathrm{ML}} = \sum_{i=1}^{n} \left(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i\right)^\top \left(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i\right) \approx J_{\mathrm{TAU}}(\boldsymbol{\theta}) = \frac{\sum_{i=1}^{n} \boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}_i)\boldsymbol{u}(\boldsymbol{x}_i)^\top \boldsymbol{\theta}}{n^{-1}\sum_{i=1}^{n} \boldsymbol{\theta}^\top \left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x}_i)\right]\left[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x}_i)\right]^\top \boldsymbol{\theta}}. \tag{3.15}$$

The cost function J_{TAU} can be thought of as a second tier approximation to J_{ML}, as it is an approximation of J_{AML} which itself is an approximation to J_{ML}. Specifically, the standing approximation

$$J_{\mathrm{ML}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i\right)^\top \boldsymbol{\Lambda}_{\boldsymbol{x}_i}^{-1} \left(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i\right) \approx J_{\mathrm{AML}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \frac{\boldsymbol{\theta}^\top \boldsymbol{A}_i \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta}}, \tag{3.16}$$

is replaced with the approximation

$$J_{\mathrm{AML}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \frac{\boldsymbol{\theta}^\top \boldsymbol{A}_i \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta}} \approx J_{\mathrm{TAU}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \frac{\boldsymbol{\theta}^\top \boldsymbol{A}_i \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{T} \boldsymbol{\theta}}, \tag{3.17}$$

where

$$\boldsymbol{T} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{B}_i. \tag{3.18}$$

The variable denominator in each summand of J_{AML} is replaced in J_{TAU} simply by the 'average' denominator value $\boldsymbol{\theta}^\top \boldsymbol{T}\, \boldsymbol{\theta}$. Recall that $\boldsymbol{S} = \boldsymbol{U}^\top \boldsymbol{U} = \sum_{i=1}^{n} \boldsymbol{A}_i$, so J_{TAU} may be written

$$J_{\mathrm{TAU}}(\boldsymbol{\theta}) = \frac{\boldsymbol{\theta}^\top \boldsymbol{S}\, \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{T}\, \boldsymbol{\theta}}. \tag{3.19}$$

With $\lambda = J_{\mathrm{TAU}}(\boldsymbol{\theta})$, Eq. (3.19) may be written as

$$\boldsymbol{S}\boldsymbol{\theta} = \lambda \boldsymbol{T}\boldsymbol{\theta}\,. \tag{3.20}$$

Hence the minimiser of J_{TAU} can be found by solving a generalised eigenvalue problem: $\widehat{\boldsymbol{\theta}}_{\mathrm{TAU}}$ will be the generalised eigenvector associated with the *smallest* generalised

eigenvalue. Because the solution may be found directly in this way, this method was originally termed "eigenfit", although it is also referred to simply as Taubin's method.

Taubin's original method involved using $\widehat{\boldsymbol{\theta}}_{\mathrm{TAU}}$ as an initial value for a subsequent Levenberg-Marquardt minimisation of a first-order approximation to J_{ML} (with identity covariance matrices). A later extension to this work incorporated a second order approximation to J_{ML} [60]. However, the significance here is that of the approximated function J_{TAU} leading to the direct solution. The estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{TAU}}$ may of course be used as an input to any iterative algorithm.

The accuracy of Taubin's approximation, compared for example to J_{AML}, is determined by the variation in each of the denominators $(\boldsymbol{\theta}^{\top}\boldsymbol{B}_i\,\boldsymbol{\theta})$. If this variation is minimal then the average $\boldsymbol{\theta}^{\top}\boldsymbol{T}\,\boldsymbol{\theta}$ will be a reasonable replacement in the sum. As $\boldsymbol{B}_i = [\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x}_i)]\,\boldsymbol{\Lambda}_{\boldsymbol{x}_i}\,[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x}_i)]^{\top}$, variation will depend in part on the inhomogeneity of the set of covariances $\{\boldsymbol{\Lambda}_{\boldsymbol{x}_i}\}$. Of course in the case where $\boldsymbol{\Lambda}_{\boldsymbol{x}_i} = \boldsymbol{I}$ the covariance matrices will provide no variation at all. This leaves the other source of variation in the gradient $[\partial_{\boldsymbol{x}}u(\boldsymbol{x}_i)]$. The quality of the Taubin approximation therefore will depend on the amount of curvature in the manifold $\mathcal{M}_{\boldsymbol{\theta}}$.

Fitzgibbon [15] (among others) points out that the Taubin method may be considered a member of the algebraic estimator family (Sect. 2.4.1). The Taubin estimate minimises the sum of squared algebraic residuals under the constraint $\|\boldsymbol{T}\boldsymbol{\theta}\|^2 = 1$.

3.2.2 Iteratively re-weighted least squares

In addition to his 'approximated distance', Sampson also put forward a algorithm for its minimisation, which has come to be known as *iteratively re-weighted least squares* (IRWLS) [56]. The difficulty in finding a closed form for $\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}$ is that $\boldsymbol{\theta}$ appears in both the numerator and denominator of J_{AML} (Eq. (2.51)). The IRWLS method overcomes this problem by breaking the minimisation into two steps. Firstly, a fixed value $\boldsymbol{\theta} = \boldsymbol{\theta}_0^{\#}$ is taken in the denominators $(\boldsymbol{\theta}^{\top}\boldsymbol{B}_i\,\boldsymbol{\theta})$. Now, only minimising over the numerators leaves a modified cost function

$$\begin{aligned} J_{\mathrm{SMP}}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\#}) &= \sum_{i=1}^{n} \frac{\boldsymbol{\theta}^{\top}\boldsymbol{A}_i\,\boldsymbol{\theta}}{{\boldsymbol{\theta}_0^{\#}}^{\top}\boldsymbol{B}_i\,\boldsymbol{\theta}_0^{\#}} \\ &= \boldsymbol{\theta}^{\top}\left(\sum_{i=1}^{n} ({\boldsymbol{\theta}_0^{\#}}^{\top}\boldsymbol{B}_i\,\boldsymbol{\theta}_0^{\#})^{-1}\boldsymbol{A}_i\right)\boldsymbol{\theta} \\ &= \boldsymbol{\theta}^{\top}\boldsymbol{M}_{\boldsymbol{\theta}_0^{\#}}\,\boldsymbol{\theta}, \end{aligned} \tag{3.21}$$

with

$$\boldsymbol{M}_{\boldsymbol{\xi}} = \sum_i \frac{1}{(\boldsymbol{\xi}^\top \boldsymbol{B}_i \boldsymbol{\xi})} \boldsymbol{A}_i. \tag{3.22}$$

The minimiser of $J_{\mathrm{SMP}}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\#})$ is sought with respect to $\boldsymbol{\theta}$ (subject to $\|\boldsymbol{\theta}\|^2 = 1$) and $\boldsymbol{\theta}^{\#}$ is kept fixed. The above cost function is identical in form to the TLS cost function, with $\boldsymbol{S}$ replaced by $\boldsymbol{M}_{\boldsymbol{\theta}}$. The same analysis applies leading to the analogous characterisation of the minimiser as the eigenvector of $\boldsymbol{M}_{\boldsymbol{\theta}}$ associated with the smallest eigenvalue. Such a solution may now be used to determine a new $\boldsymbol{\theta}_1^{\#}$, which can then be used to compute new denominators (re-weighting). The process may be repeated until convergence and is described in algorithm 3.2.

The IRWLS has a fundamental problem in that it does not act to compute a minimiser of $J_{\mathrm{AML}}(\boldsymbol{\theta})$. It will be shown in a subsequent section why this is the case. The theoretical limitation of this method is borne out in a later section.

Algorithm 3.2 (IRWLS) *Computes the IRWLS estimate* $\widehat{\boldsymbol{\theta}}_{\mathrm{IRWLS}}$

1. *Choose an initial estimate* $\boldsymbol{\theta}_0 = \widehat{\boldsymbol{\theta}}_{\mathrm{TLS}}$ *and set* $k = 0$
2. *Compute* $\boldsymbol{M}_{\boldsymbol{\theta}_k}$ *as per Eq. (3.22)*
3. *Take* $\boldsymbol{\theta}_{k+1}$ *as the eigenvector of* $\boldsymbol{M}_{\boldsymbol{\theta}_k}$ *associated with the smallest eigenvalue*
4. *If* $\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_{k+1}\| > \varepsilon$ *then increment* k *and go back to step 2. Otherwise, terminate and set the estimate* $\widehat{\boldsymbol{\theta}}_{\mathrm{IRWLS}} = \boldsymbol{\theta}_{k+1}$

3.2.3 Fundamental numerical scheme

The IRWLS algorithm was inspired directly from the specific form of J_{AML}. Unfortunately this algorithm exhibits significant bias. An alternative approach is to start not with the cost function, but with an equation characterising the minimiser.

An obvious condition satisfied by a minimiser of J_{AML} is that the first derivative vector of J_{AML} must vanish when evaluated at $\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}$, namely

$$\left[\partial_{\boldsymbol{\theta}} J_{\mathrm{AML}}(\boldsymbol{\theta})\right]^\top_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}} = \boldsymbol{0}. \tag{3.23}$$

Of course, this equation is satisfied not only at $\boldsymbol{\theta} = \widehat{\boldsymbol{\theta}}_{\mathrm{AML}}$, but also at any local minima or maxima of J_{AML}. An implicit assumption therefore is that an initial estimate produced using TLS (or some other more sophisticated method) will be good enough that the nearest point satisfying Eq. (3.23) will be the global minimiser.

The Jacobian of J_{AML} may be computed analytically. To simplify matters, J_{AML} is written as

$$J_{\mathrm{AML}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \frac{a_i(\boldsymbol{\theta})}{b_i(\boldsymbol{\theta})}, \tag{3.24}$$

where $a_i(\boldsymbol{\theta}) = \boldsymbol{\theta}^\top \boldsymbol{A}_i \boldsymbol{\theta}$, and $b_i(\boldsymbol{\theta}) = \boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta}$. A vectorised form of the 'quotient rule' of differentiation,

$$\partial_{\boldsymbol{\theta}} \left(\frac{a(\boldsymbol{\theta})}{b(\boldsymbol{\theta})} \right) = \frac{[\partial_{\boldsymbol{\theta}} a(\boldsymbol{\theta})]\, b(\boldsymbol{\theta}) - [\partial_{\boldsymbol{\theta}} b(\boldsymbol{\theta})]\, a(\boldsymbol{\theta})}{b(\boldsymbol{\theta})^2}, \tag{3.25}$$

may be used to construct a closed form for the Jacobian. Substituting the identities

$$[\partial_{\boldsymbol{\theta}} a(\boldsymbol{\theta})]^\top = 2\boldsymbol{A}_i \boldsymbol{\theta} \qquad \text{and} \qquad [\partial_{\boldsymbol{\theta}} b(\boldsymbol{\theta})]^\top = 2\boldsymbol{B}_i \boldsymbol{\theta}, \tag{3.26}$$

into Eq. (3.25) yields

$$\begin{aligned} \Big[\partial_{\boldsymbol{\theta}} J_{\mathrm{AML}}(\boldsymbol{\theta})\Big]^\top &= \sum_i \frac{2\boldsymbol{A}_i\boldsymbol{\theta}(\boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta}) - 2\boldsymbol{B}_i\boldsymbol{\theta}(\boldsymbol{\theta}^\top \boldsymbol{A}_i \boldsymbol{\theta})}{(\boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta})^2} \\ &= 2\left[\sum_{i=1}^{n} \left(\frac{\boldsymbol{A}_i}{\boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta}} - \frac{\boldsymbol{B}_i(\boldsymbol{\theta}^\top \boldsymbol{A}_i \boldsymbol{\theta})}{(\boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta})^2} \right)\right] \boldsymbol{\theta} \end{aligned} \tag{3.27}$$

with

$$\boldsymbol{X}_{\boldsymbol{\theta}} = \sum_{i=1}^{n} \frac{1}{\boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta}} \boldsymbol{A}_i - \sum_{i=1}^{n} \frac{(\boldsymbol{\theta}^\top \boldsymbol{A}_i \boldsymbol{\theta})}{(\boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta})^2} \boldsymbol{B}_i, \tag{3.28}$$

Eq. (3.27) can be written

$$\Big[\partial_{\boldsymbol{\theta}} J_{\mathrm{AML}}(\boldsymbol{\theta})\Big]^\top = 2\boldsymbol{X}_{\boldsymbol{\theta}} \boldsymbol{\theta}. \tag{3.29}$$

That the Jacobian of J_{AML} may be 'factored' in this way is crucial for the subsequent development of a minimisation algorithm. The above equation, called the *variational equation*, combined with the criterion of Eq. (3.23) means that the minimiser must satisfy

$$\boldsymbol{X}_{\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}} \widehat{\boldsymbol{\theta}}_{\mathrm{AML}} = \boldsymbol{0}. \tag{3.30}$$

The simple form of Eq. (3.30) suggests an iterative algorithm for finding $\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}$. A vector $\boldsymbol{\theta}$ will satisfy Eq. (3.30) if and only if it lies in the null space of $\boldsymbol{X}_{\boldsymbol{\theta}}$. Hence, if $\boldsymbol{\theta}_1$ is an initial guess at the minimiser, then an improved guess $\boldsymbol{\theta}_2$ can be taken as the vector in the eigenspace of $\boldsymbol{X}_{\boldsymbol{\theta}_1}$ which most closely approximates the null space

of $\boldsymbol{X}_{\boldsymbol{\theta}}$. This will be the eigenvector of $\boldsymbol{X}_{\boldsymbol{\theta}_1}$ associated with the eigenvalue closest to zero. This process can now be repeated, whereby a new factor matrix $\boldsymbol{X}_{\boldsymbol{\theta}_2}$ may be computed and used to generate a new estimate $\boldsymbol{\theta}_3$, and so on. This algorithm is termed the *fundamental numerical scheme* (FNS) and is described specifically by Algorithm 3.3.

Algorithm 3.3 (FNS) *Computes an estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{FNS}}$ of the minimiser of J_{AML}*

1. *Choose an initial estimate $\boldsymbol{\theta}_0$ and set $k = 0$.*
2. *Compute $\boldsymbol{X}_{\boldsymbol{\theta}_k}$ as per Eq. (3.28).*
3. *Take $\boldsymbol{\theta}_{k+1}$ as the eigenvector of $\boldsymbol{X}_{\boldsymbol{\theta}_k}$ associated with the eigenvalue closest to zero.*
4. *If $\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_{k+1}\| > \varepsilon_{\boldsymbol{\theta}}$ and $|J_{\mathrm{AML}}(\boldsymbol{\theta}_k) - J_{\mathrm{AML}}(\boldsymbol{\theta}_{k+1})| > \varepsilon_J$ then increment k and go back to step 2. Otherwise, terminate and set the estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{FNS}} = \boldsymbol{\theta}_{k+1}$.*

3.2.4 Iterative issues

For methods which require iteration, care needs to be taken at each end of the process: choosing an initial value to start, or seed, the method, and determining when to cease iterating.

Seeds

Although not completely necessary, a non-iterative method should ideally be used to generate a seed for any subsequent iterative algorithm. It is possible though to contemplate a sequence of methods where the output of one (faster but not quite as accurate) method is used to generate a seed for a second, slower but more accurate algorithm.

When choosing a method to seed FNS, there is primarily a choice between using an algebraic least squares method, for example TLS, or using Taubin's method. Experiments show that, for data contaminated with a very high level of noise, TLS will generate particularly bad estimates. When FNS is seeded with these estimates, the subsequent iterates may converge to a (distant) local minimum, or diverge completely.

Therefore, it is generally safest to initialise FNS with the Taubin estimate. Computationally, finding $\widehat{\boldsymbol{\theta}}_{\mathrm{TAU}}$ is more or less on par with a single iteration of FNS – certainly not a significant overhead.

Termination

Choosing when to stop an iterative algorithm also requires some thought. It is not desirable to stop too early, or to iterate for too long. Generally, one stopping condition is to cease iterating when $\|\widehat{\boldsymbol{\theta}}_k - \widehat{\boldsymbol{\theta}}_{k-1}\| < \varepsilon_{\boldsymbol{\theta}}$, for some small $\varepsilon_{\boldsymbol{\theta}}$. Additionally, a "safeguard" maximum number of iterations has been set, which will usually be reached in the event that the algorithm fails to converge at all. As the goal of iterative algorithms is to minimise a particular cost function, a second criteria for convergence, and hence stopping condition, is when successive estimates have small change in their *cost function* value. In the FNS case, we wish to stop when $|J_{\mathrm{AML}}(\boldsymbol{\theta}_i) - J_{\mathrm{AML}}(\boldsymbol{\theta}_{i-1})| < \varepsilon_J$. Again, there is a trade-off with computational cost, as more effort is required to explicitly compute the value of J_{AML} – for both estimates – at each iteration. However, an optimisation may be applied reducing the overhead of computing a value of J_{AML} at each iteration. Of course $J_{\mathrm{AML}}(\boldsymbol{\theta}_i)$ only need be computed once as its value can be stored for the next iteration (when $J_{\mathrm{AML}}(\boldsymbol{\theta}_{i-1})$ is required). Furthermore, at each iteration of the FNS algorithm, the matrix $\boldsymbol{X}_{\boldsymbol{\theta}_i}$ must be computed. As seen in Eq. (3.32), $\boldsymbol{X}_{\boldsymbol{\theta}} = \boldsymbol{M}_{\boldsymbol{\theta}} - \boldsymbol{R}_{\boldsymbol{\theta}}$, where $\boldsymbol{M}_{\boldsymbol{\theta}}$ is defined as

$$\boldsymbol{M}_{\boldsymbol{\theta}} = \sum_i \frac{\boldsymbol{A}_i}{\boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta}}.$$

As $J_{\mathrm{AML}}(\boldsymbol{\theta}) = \sum_i (\boldsymbol{\theta}^\top \boldsymbol{A}_i \boldsymbol{\theta})(\boldsymbol{\theta}^\top \boldsymbol{B}_i \boldsymbol{\theta})^{-1}$, then its value for the i^{th} estimate can be computed as

$$J_{\mathrm{AML}}(\boldsymbol{\theta}_i) = \boldsymbol{\theta}_i^\top \boldsymbol{M}_{\boldsymbol{\theta}_i} \boldsymbol{\theta}_i, \tag{3.31}$$

which requires only minimal overhead because $\boldsymbol{M}_{\boldsymbol{\theta}_i}$ has been computed anyway as a necessary part of the FNS iteration.

The stopping condition used is determined by a choice of three variables: the safe guard maximum number of iterations, the tolerance for difference in successive $\boldsymbol{\theta}_k$, and the tolerance for difference in the successive values of the cost function.

3.2.5 Bias in IRWLS estimator

The FNS and IRWLS algorithms appear similar, with one difference being the matrices involved: $\boldsymbol{M}_{\boldsymbol{\theta}}$ for IRWLS, and $\boldsymbol{X}_{\boldsymbol{\theta}}$ for FNS. Examination of the form of $\boldsymbol{X}_{\boldsymbol{\theta}}$, given in Eq. (3.28), shows that it is the sum of two matrices, the first of which is actually $\boldsymbol{M}_{\boldsymbol{\theta}}$, the IRWLS matrix. So

$$\boldsymbol{X}_{\boldsymbol{\theta}} = \boldsymbol{M}_{\boldsymbol{\theta}} - \boldsymbol{R}_{\boldsymbol{\theta}}, \tag{3.32}$$

with $\boldsymbol{M}_{\boldsymbol{\theta}}$ given in Eq. (3.22), and

$$\boldsymbol{R}_{\boldsymbol{\theta}} = \sum_{i=1}^{n} \frac{(\boldsymbol{\theta}^{\top} \boldsymbol{A}_i \, \boldsymbol{\theta})}{(\boldsymbol{\theta}^{\top} \boldsymbol{B}_i \, \boldsymbol{\theta})^2} \boldsymbol{B}_i. \tag{3.33}$$

At each step of the IRWLS algorithm, $\boldsymbol{\theta}_{k+1}^{\top} \boldsymbol{M}_{\boldsymbol{\theta}_k} \boldsymbol{\theta}_{k+1} = 0$ is minimised (subject to $\|\boldsymbol{\theta}_{k+1}\|^2 = 1$) using an eigendecomposition

$$\boldsymbol{M}_{\boldsymbol{\theta}_k} \boldsymbol{\theta}_{k+1} = \lambda \boldsymbol{\theta}_{k+1}.$$

As $\lambda = J_{\mathrm{SMP}}(\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k+1})$ which is similar to Eq. (2.28),

$$\boldsymbol{M}_{\boldsymbol{\theta}_k} \boldsymbol{\theta}_{k+1} = J_{\mathrm{SMP}}(\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k+1}) \, \boldsymbol{\theta}_{k+1}.$$

Upon convergence, the estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{IRWLS}} = \boldsymbol{\theta}_{\infty} = \lim_{k \to \infty} \boldsymbol{\theta}_k$ will satisfy the identity

$$\boldsymbol{M}_{\boldsymbol{\theta}_{\infty}} \boldsymbol{\theta}_{\infty} = J_{\mathrm{SMP}}(\boldsymbol{\theta}_{\infty}, \boldsymbol{\theta}_{\infty}) \, \boldsymbol{\theta}_{\infty}. \tag{3.34}$$

Given that $J_{\mathrm{SMP}}(\boldsymbol{\theta}, \boldsymbol{\theta}) = J_{\mathrm{AML}}(\boldsymbol{\theta})$, then the estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{IRWLS}}$ will satisfy

$$\big(\boldsymbol{M}_{\boldsymbol{\theta}} - J_{\mathrm{AML}}(\boldsymbol{\theta}) \boldsymbol{I}\big) \, \boldsymbol{\theta} = 0. \tag{3.35}$$

This equation is different from the variational equation, Eq. (3.30). Hence the $\widehat{\boldsymbol{\theta}}_{\mathrm{IRWLS}}$ does not minimise J_{AML}.

3.3 Incorporating the ancillary constraint

All of the estimation methods presented so far take no account of the ancillary constraint. In general they will return an estimate which does not satisfy the ancillary constraint. There are two ways to remedy this: one is to perform some kind of correction on an unconstrained estimate, and the other is to modify the estimation method itself to ensure that a constrained estimate is always returned. Each of these is now examined.

3.3.1 Reparameterisation

One way of obtaining a final estimate consistent with the ancillary constraint is to reparameterise the problem so that under the new parameterisation, the constraint is always satisfied. Obviously such a parameterisation must be chosen specifically for each kind of problem. Usually there is not a unique candidate for a new parameterisation.

With the new parameterisation adopted, the minimiser of J_{AML} (or other problem specific cost function) may be found by using for example Levenberg-Marquardt. Given a modified parameter space, none of the other minimisation methods described above may be used as the problem is no longer linear in the new parameters. Usually these methods can be quite complex and sometimes it is not easy to come up with a parameterisation that will work in the entire search space. In the cases where multiple parameterisations are possible, it is sometimes necessary to change the parameterisation during the minimisation.

Because of the inherent problem dependent nature of the reparameterisation method, specific ways of doing this are described later for each geometric problem.

3.3.2 Post-correction

Rather than completely changing the model to incorporate the ancillary constraint, it is possible instead to adopt a two step procedure. First, the unconstrained estimate is computed. The original estimate is then modified (or corrected) so that it satisfies the constraint. There are several possibilities for these so-called post-correction methods, both non-iterative and iterative.

Kanatani was the first to propose an iterative scheme [29, Chap. 5]. The idea is to find the new constrained estimate which is closest to the original unconstrained one. Closest here refers to minimising the Mahalanobis distance, where the covariance used is of the original estimate. Various modifications of this scheme are possible, depending on which specific form of the covariance matrix is chosen.

3.4 Direct constrained estimation

Reparameterisations techniques must be applied specifically for each problem form under consideration. Additionally, there is unlikely to be a unique parameterisation for a given problem, which means it may not be obvious *a priori* which one to choose. Once

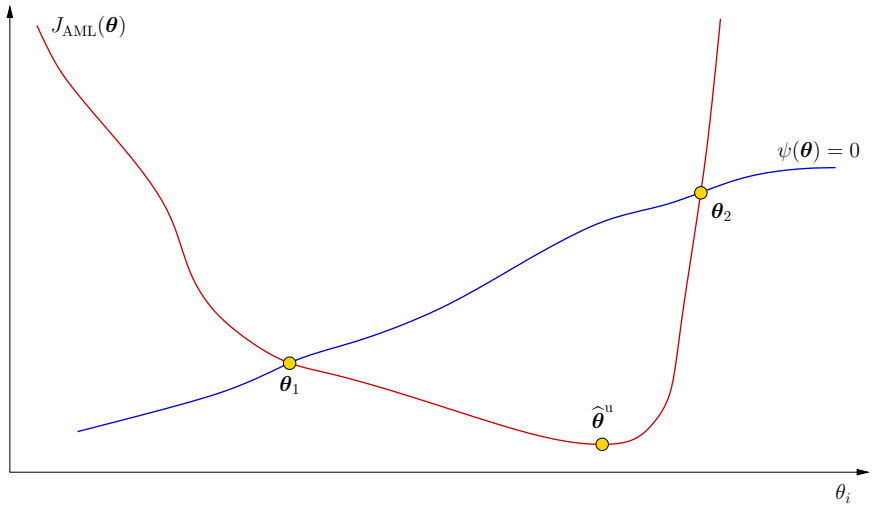


Figure 3.1: Correcting an unconstrained estimate. A 'slice' of the J_{AML} surface is shown superimposed with the curve $\psi(\boldsymbol{\theta}) = 0$. Even though $\boldsymbol{\theta}_2$ may be close to the unconstrained minimiser $\widehat{\boldsymbol{\theta}}^{\mathrm{u}}$ it has a higher cost function value than $\boldsymbol{\theta}_1$ which is a preferable choice

reparameterised, the minimisation must proceed using a general (relatively inefficient) minimiser such as Levenberg-Marquardt, as methods such as FNS will not apply to a parameterisation that fails to conform to the linear form of Eq. (2.2).

The post correction method alleviates both of these problems. It is (more or less) generally applicable using the generic equations of Sect. 3.3.2. In terms of efficiency, an unconstrained solution may be computed using FNS, and the correction itself iterative, typically requiring just two or three iterations. However, the very nature of *post* correction leaves open the question of whether the optimum solution is attained. Imagine for a given cost function there is an unconstrained minimiser $\widehat{\boldsymbol{\theta}}^{\mathrm{u}}$ minimising a cost function $J(\boldsymbol{\theta})$, and close to $\widehat{\boldsymbol{\theta}}^{\mathrm{u}}$ there are $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ both satisfying $\psi(\boldsymbol{\theta}_1) = \psi(\boldsymbol{\theta}_2) = 0$ and with $J(\boldsymbol{\theta}_1) < J(\boldsymbol{\theta}_2)$. The ideal choice as a constrained estimate is $\boldsymbol{\theta}_1$. It is possible that the post correction method may choose $\boldsymbol{\theta}_2$, particularly if it is the case that $\boldsymbol{\theta}_2$ is closer in some sense to $\widehat{\boldsymbol{\theta}}^{\mathrm{u}}$ than $\boldsymbol{\theta}_1$. This situation is shown in Fig . 3.1.

An alternative method would be desirable which considers the constraint from the outset, rather than tacking it on at the end similar to post correction. To maintain a just-iterative method the goal is to find a method similar to FNS where $\boldsymbol{X}_{\boldsymbol{\theta}}$ is replaced with a different matrix built incorporating the constraint. This method is termed constrained FNS (CFNS).

3.4.1 Preliminary method

Minimising J_{AML} subject to $\psi(\boldsymbol{\theta}) = 0$ may be written as a system incorporating a Lagrange multiplier. The required solution $\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}$ satisfies

$$\left[\partial_{\boldsymbol{\theta}} J_{\mathrm{AML}}(\boldsymbol{\theta})\right]^{\top}_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}} + \lambda \left[\partial_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta})\right]^{\top}_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}} = \mathbf{0} \tag{3.36}$$

$$\psi(\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}) = 0. \tag{3.37}$$

The key to progressing with the development of a constrained version of FNS is replacing the general constraint function with one that is so-called k-homogeneous (see Eq. (2.7)). It turns out that for several common problems, the associated ancillary constraint can be written in this form. Differentiating the identity $\psi(\rho\boldsymbol{\theta}) = \rho^k \psi(\boldsymbol{\theta})$ with respect to ρ gives

$$\left[\partial_{\boldsymbol{\theta}} \psi(\rho\boldsymbol{\theta})\right] \boldsymbol{\theta} = k\rho^{k-1} \psi(\boldsymbol{\theta}), \tag{3.38}$$

and setting $\rho = 1$ leaves the identity

$$\psi(\boldsymbol{\theta}) = k^{-1} \left[\partial_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta})\right] \boldsymbol{\theta}. \tag{3.39}$$

By setting $\boldsymbol{d}_{\boldsymbol{\theta}} = \left[\partial_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta})\right]^{\top} / 2$ (and omitting the specification $\boldsymbol{\theta} = \widehat{\boldsymbol{\theta}}_{\mathrm{AML}}$) the system may be written as

$$\frac{1}{2} \left[\partial_{\boldsymbol{\theta}} J_{\mathrm{AML}}(\boldsymbol{\theta})\right]^{\top} + \lambda \, \boldsymbol{d}_{\boldsymbol{\theta}} = \mathbf{0} \,,$$

$$\boldsymbol{d}_{\boldsymbol{\theta}}^{\top} \boldsymbol{\theta} = 0.$$

As shown in Sect. 3.2.3, the Jacobian of $J_{\mathrm{AML}}(\boldsymbol{\theta})$ may be factorised and written $\left[\partial_{\boldsymbol{\theta}} J_{\mathrm{AML}}(\boldsymbol{\theta})\right]^{\top} = 2\boldsymbol{X}_{\boldsymbol{\theta}} \, \boldsymbol{\theta}$. Substituting this into the above system gives

$$\boldsymbol{X}_{\boldsymbol{\theta}} \boldsymbol{\theta} + \lambda \, \boldsymbol{d}_{\boldsymbol{\theta}} = \mathbf{0} \,, \tag{3.40}$$

$$\boldsymbol{d}_{\boldsymbol{\theta}}^{\top} \boldsymbol{\theta} = 0, \tag{3.41}$$

which can be written as a single matrix equation

$$\begin{bmatrix} \boldsymbol{X}_{\boldsymbol{\theta}} & \boldsymbol{d}_{\boldsymbol{\theta}} \\ \boldsymbol{d}_{\boldsymbol{\theta}}^{\top} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\theta} \\ \lambda \end{bmatrix} = \boldsymbol{Y}_{\boldsymbol{\theta}} \, \boldsymbol{\eta} = \mathbf{0}. \tag{3.42}$$

Eq. (3.42) has a similar form to the original variational equation, Eq. (3.29), suggesting a revised algorithm in which $\boldsymbol{X}_{\boldsymbol{\theta}}$ is replaced with $\boldsymbol{Y}_{\boldsymbol{\theta}}$. However, $\boldsymbol{Y}_{\boldsymbol{\theta}}$ is an $(l+1) \times (l+1)$ matrix ($\boldsymbol{\theta}$ is an $l-$vector), meaning this new scheme would potentially operate in a higher dimensionality than "ordinary" FNS. In subsequent implementation, it turned out that a FNS scheme using $\boldsymbol{Y}_{\boldsymbol{\theta}}$ was not convergent. As an initial remedy, a further update is sought where a matrix of the same order of $\boldsymbol{\theta}$ is devised.

3.4.2 Revised method

The first step in deriving such a "same order" matrix is to solve for the Lagrange multiplier by multiplying both sides of Eq. (3.40) by $\boldsymbol{d}_{\boldsymbol{\theta}}^{\top}$,

$$\boldsymbol{d}_{\boldsymbol{\theta}}^{\top}\boldsymbol{X}_{\boldsymbol{\theta}}\boldsymbol{\theta} + \lambda \boldsymbol{d}_{\boldsymbol{\theta}}^{\top}\boldsymbol{d}_{\boldsymbol{\theta}} = 0, \tag{3.43}$$

so that

$$\lambda = \|\boldsymbol{d}_{\boldsymbol{\theta}}\|^{-2}\boldsymbol{d}_{\boldsymbol{\theta}}^{\top}\boldsymbol{X}_{\boldsymbol{\theta}}\boldsymbol{\theta}. \tag{3.44}$$

Substituting this into Eq. (3.40) gives

$$\left(\boldsymbol{X}_{\boldsymbol{\theta}} - \|\boldsymbol{d}_{\boldsymbol{\theta}}\|^{-2}\boldsymbol{d}_{\boldsymbol{\theta}}\boldsymbol{d}_{\boldsymbol{\theta}}^{\top}\boldsymbol{X}_{\boldsymbol{\theta}}\right)\boldsymbol{\theta} = \left(\boldsymbol{X}_{\boldsymbol{\theta}} - \boldsymbol{D}_{\boldsymbol{\theta}}\boldsymbol{X}_{\boldsymbol{\theta}}\right)\boldsymbol{\theta} = \left(\boldsymbol{I} - \boldsymbol{D}_{\boldsymbol{\theta}}\right)\boldsymbol{X}_{\boldsymbol{\theta}}\,\boldsymbol{\theta} = \boldsymbol{0}, \tag{3.45}$$

with $\boldsymbol{D}_{\boldsymbol{\theta}} = \|\boldsymbol{d}_{\boldsymbol{\theta}}\|^{-2}\boldsymbol{d}_{\boldsymbol{\theta}}\boldsymbol{d}_{\boldsymbol{\theta}}^{\top}$. Now, considering

$$\boldsymbol{D}_{\boldsymbol{\theta}}\,\boldsymbol{\theta} = \|\boldsymbol{d}_{\boldsymbol{\theta}}\|^{-2}\boldsymbol{d}_{\boldsymbol{\theta}}\boldsymbol{d}_{\boldsymbol{\theta}}^{\top}\boldsymbol{\theta} = \boldsymbol{0} \tag{3.46}$$

(substituting Eq. (3.41)), it is clear that $(\boldsymbol{I} - \boldsymbol{D}_{\boldsymbol{\theta}})\boldsymbol{\theta} = \boldsymbol{\theta}$. Combining this with Eq. (3.45), and including a factor of $\|\boldsymbol{\theta}\|^2$ to compensate for $\boldsymbol{X}_{\boldsymbol{\theta}}$ being homogeneous of degree -2, yields

$$\left[\|\boldsymbol{\theta}\|^2\left((\boldsymbol{I} - \boldsymbol{D}_{\boldsymbol{\theta}})\boldsymbol{X}_{\boldsymbol{\theta}}(\boldsymbol{I} - \boldsymbol{D}_{\boldsymbol{\theta}})\right) + \boldsymbol{D}_{\boldsymbol{\theta}}\right]\,\boldsymbol{\theta} = \boldsymbol{Y}'_{\boldsymbol{\theta}}\,\boldsymbol{\theta} = \boldsymbol{0}, \tag{3.47}$$

This equation is equivalent to the original system comprising Eq. (3.36) and Eq. (3.37). The matrix $\boldsymbol{Y}'_{\boldsymbol{\theta}}$ has size $l \times l$ which is the same order as the $\boldsymbol{\theta}$.

Again this equation suggests the adoption of a new FNS-like method where, in this case, $\boldsymbol{Y}'_{\boldsymbol{\theta}}$ is substituted for $\boldsymbol{X}_{\boldsymbol{\theta}}$. However, again, such a method fails to converge.

It is not always the case that equations of the form of Eq. (3.27), or Eq. (3.47), will lead to convergent iterative algorithms. The mapping of a particular matrix to its eigenvector corresponding to the eigenvalue closest to zero is a complicated function, not expressible in closed form. Therefore the convergent properties of any method are difficult to consider directly. In the case of Eq. (3.47) it is possible to consider a family of equivalent equations of the form $(\boldsymbol{Y}'_{\boldsymbol{\theta}} + \boldsymbol{N}_{\boldsymbol{\theta}})\,\boldsymbol{\theta} = \boldsymbol{0}$ where $\boldsymbol{N}_{\boldsymbol{\theta}}$ is a symmetric matrix such that $\boldsymbol{N}_{\boldsymbol{\theta}}\,\boldsymbol{\theta} = \boldsymbol{0}$ for all $\boldsymbol{\theta} \neq \boldsymbol{0}$ and that the function $\boldsymbol{\theta} \mapsto \boldsymbol{N}_{\boldsymbol{\theta}}$ is homogeneous of degree 0.

Considering this, a new matrix $\boldsymbol{Z}_{\boldsymbol{\theta}}$ is sought such that the equation $\boldsymbol{Z}_{\boldsymbol{\theta}}\,\boldsymbol{\theta} = \boldsymbol{0}$ is equivalent to Eq. (3.47). Firstly, necessary terms are given. The Hessian matrix of the AML cost function is written $\boldsymbol{H}_J = [\partial^2_{\boldsymbol{\theta}\boldsymbol{\theta}} J_{\mathrm{AML}}(\boldsymbol{\theta})]$ and the Hessian matrix of the ancillary constraint function is written $\boldsymbol{H}_{\psi} = [\partial^2_{\boldsymbol{\theta}\boldsymbol{\theta}}\psi(\boldsymbol{\theta})]$. Additionally, for each i from 1 to l,

define $\boldsymbol{e}_i$ as the basis vector which has its i^{th} element equal to one, and all others zero. Expansive calculation shows that $\boldsymbol{H}_J$ may be written in terms of the matrices $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$ as

$$\boldsymbol{H}_J = 2\left(\boldsymbol{X}_{\boldsymbol{\theta}} - \boldsymbol{T}_{\boldsymbol{\theta}}\right), \tag{3.48}$$

where

$$\boldsymbol{T}_{\boldsymbol{\theta}} = \sum_{i=1}^{n} \frac{2}{(\boldsymbol{\theta}^\top \boldsymbol{B}_i\, \boldsymbol{\theta})^2} \left[\boldsymbol{A}_i\, \boldsymbol{\theta}\boldsymbol{\theta}^\top \boldsymbol{B}_i + \boldsymbol{B}_i\, \boldsymbol{\theta}\boldsymbol{\theta}^\top \boldsymbol{A}_i - 2\frac{\boldsymbol{\theta}^\top \boldsymbol{A}_i\, \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \boldsymbol{B}_i\, \boldsymbol{\theta}} \boldsymbol{B}_i\, \boldsymbol{\theta}\boldsymbol{\theta}^\top \boldsymbol{B}_i\right]. \tag{3.49}$$

Using the above, a replacement matrix $\boldsymbol{Z}$ can be defined as $\boldsymbol{Z}_{\boldsymbol{\theta}} = \boldsymbol{Z}_{\boldsymbol{\theta}}^{(1)} + \boldsymbol{Z}_{\boldsymbol{\theta}}^{(2)} + \boldsymbol{Z}_{\boldsymbol{\theta}}^{(3)}$ where

$$\boldsymbol{Z}_{\boldsymbol{\theta}}^{(1)} = (\boldsymbol{I} - \boldsymbol{D}_{\boldsymbol{\theta}})\boldsymbol{H}_J\left(2\boldsymbol{\theta}\boldsymbol{\theta}^\top - \|\boldsymbol{\theta}\|^2 \boldsymbol{I}\right), \tag{3.50}$$

$$\boldsymbol{Z}_{\boldsymbol{\theta}}^{(2)} = \frac{\|\boldsymbol{\theta}\|^2}{\|\boldsymbol{d}_{\boldsymbol{\theta}}\|^2}\Big[\sum_{i=1}^{l} \left(\boldsymbol{H}_\psi \boldsymbol{e}_i \boldsymbol{d}_{\boldsymbol{\theta}}^\top + \boldsymbol{d}_{\boldsymbol{\theta}} \boldsymbol{e}_i^\top \boldsymbol{H}_\psi\right) \boldsymbol{X}_{\boldsymbol{\theta}}\, \boldsymbol{\theta} \boldsymbol{e}_i^\top - 2\|\boldsymbol{d}_{\boldsymbol{\theta}}\|^{-2} \boldsymbol{d}_{\boldsymbol{\theta}} \boldsymbol{d}_{\boldsymbol{\theta}}^\top \boldsymbol{X}_{\boldsymbol{\theta}}\, \boldsymbol{\theta} \boldsymbol{d}_{\boldsymbol{\theta}}^\top \boldsymbol{H}_\psi\Big], \tag{3.51}$$

$$\boldsymbol{Z}_{\boldsymbol{\theta}}^{(3)} = \frac{\kappa}{\|\boldsymbol{d}_{\boldsymbol{\theta}}\|^2}\left[\frac{\psi(\boldsymbol{\theta})}{4}\boldsymbol{H}_\psi + \boldsymbol{d}_{\boldsymbol{\theta}} \boldsymbol{d}_{\boldsymbol{\theta}}^\top - \frac{\psi(\boldsymbol{\theta})}{2}\|\boldsymbol{d}_{\boldsymbol{\theta}}\|^{-2} \boldsymbol{d}_{\boldsymbol{\theta}} \boldsymbol{d}_{\boldsymbol{\theta}}^\top \boldsymbol{H}_\psi\right]. \tag{3.52}$$

The form of $\boldsymbol{Z}_{\boldsymbol{\theta}}$ arises through algebraic manipulations only, there is no geometric interpretation to its form. Neither does the use of individual matrices, $\boldsymbol{Z}_{\boldsymbol{\theta}}^{(1)}$, $\boldsymbol{Z}_{\boldsymbol{\theta}}^{(2)}$, and $\boldsymbol{Z}_{\boldsymbol{\theta}}^{(3)}$ have any significance – they are used merely to break up a long expression. Note that

$$\begin{aligned}
\boldsymbol{Z}_{\boldsymbol{\theta}}^{(1)}\, \boldsymbol{\theta} &= \|\boldsymbol{\theta}\|^2 (\boldsymbol{I} - \boldsymbol{D}_{\boldsymbol{\theta}}) \boldsymbol{X}_{\boldsymbol{\theta}} (\boldsymbol{I} - \boldsymbol{D}_{\boldsymbol{\theta}}), \\
\boldsymbol{Z}_{\boldsymbol{\theta}}^{(2)}\, \boldsymbol{\theta} &= \boldsymbol{0}, \\
\boldsymbol{Z}_{\boldsymbol{\theta}}^{(3)}\, \boldsymbol{\theta} &= \boldsymbol{D}_{\boldsymbol{\theta}}\, \boldsymbol{\theta}.
\end{aligned}$$

Therefore the equation $\boldsymbol{Z}_{\boldsymbol{\theta}}\, \boldsymbol{\theta} = \boldsymbol{0}$ is equivalent to $\left[\|\boldsymbol{\theta}\|^2 \big((\boldsymbol{I} - \boldsymbol{D}_{\boldsymbol{\theta}}) \boldsymbol{X}_{\boldsymbol{\theta}} (\boldsymbol{I} - \boldsymbol{D}_{\boldsymbol{\theta}})\big) + \boldsymbol{D}_{\boldsymbol{\theta}}\right]\, \boldsymbol{\theta} = \boldsymbol{Y}'_{\boldsymbol{\theta}}\, \boldsymbol{\theta} = \boldsymbol{0}$, as intended.

Unlike the matrices $\boldsymbol{X}_{\boldsymbol{\theta}}$ and $\boldsymbol{Y}'_{\boldsymbol{\theta}}$, the above form for $\boldsymbol{Z}_{\boldsymbol{\theta}}$ is not symmetric. However, an equation equivalent to $\boldsymbol{Z}_{\boldsymbol{\theta}}\boldsymbol{\theta} = \boldsymbol{0}$ is

$$\left[\boldsymbol{Z}_{\boldsymbol{\theta}}^\top \boldsymbol{Z}_{\boldsymbol{\theta}}\right]\, \boldsymbol{\theta} = \boldsymbol{0} \tag{3.53}$$

where $\boldsymbol{Z}_{\boldsymbol{\theta}}^\top \boldsymbol{Z}_{\boldsymbol{\theta}}$ is a symmetric matrix. A constrained estimation algorithm may then be expressed with the original, unconstrained, matrix $\boldsymbol{X}_{\boldsymbol{\theta}}$ replaced with $\boldsymbol{Z}_{\boldsymbol{\theta}}^\top \boldsymbol{Z}_{\boldsymbol{\theta}}$. A constrained version of the FNS algorithm, termed CFNS, is defined by replacing the matrix $\boldsymbol{X}_{\boldsymbol{\theta}}$ with $\boldsymbol{Z}_{\boldsymbol{\theta}}$ in Alg. 3.3.

3.5 Summary

A parametric model was defined in which a principal constraint equation related parameters and data points. The function was defined to be linear in parameters, although non-linear in data points. The model of noise adopted was that of Gaussian perturbations from unknown ideal data points. Therefore, for each new data point added, an associated nuisance parameter must be incorporated to the overall model.

By considering purely algebraic residuals, a family of methods, known as algebraic least squares, can be be derived which estimate parameters minimising by the sum of squared residuals subject to a particular constraint. These methods are not generally satisfactory in general because they exhibit a statistical bias.

Using a ML cost function, an optimal estimator, under the defined parametric model, is

$$\widehat{\boldsymbol{\theta}}_{\mathrm{ML}} = \arg\min \sum_{i=1}^{n} (\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i)^{\top} \boldsymbol{\Lambda}_{\boldsymbol{x}_i}^{-1} (\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i).$$

In practice this cost function is very difficult to minimise because of the presence of nuisance parameters, the $\overline{\boldsymbol{x}}_i$. At best, progress can be made only for each individual problem. The ML cost function may be approximated by one that eliminates the nuisance parameters and depends only on the parameter vector of interest, $\boldsymbol{\theta}$. Following the method of Sampson, such a cost function is defined as

$$\widehat{\boldsymbol{\theta}}_{\mathrm{AML}} = \arg\min \sum_{i=1}^{n} \frac{\boldsymbol{\theta}^{\top} \boldsymbol{A}_i \boldsymbol{\theta}}{\boldsymbol{\theta}^{\top} \boldsymbol{B}_i \boldsymbol{\theta}}.$$

The iteratively re-weighted least squares algorithm attempts to minimise this cost function, but it has an identifiable theoretical bias. A more effective means of minimising J_{AML} can be found by considering that the gradient must vanish at the minimiser. By directly computing the gradient, a variational equation is formed

$$[\partial_{\boldsymbol{\theta}} J_{\mathrm{AML}}(\boldsymbol{\theta})]^{\top}_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}_{\mathrm{ALS}}} = 2\boldsymbol{X}_{\boldsymbol{\theta}}\boldsymbol{\theta} = \mathbf{0}\,.$$

Because $[\partial_{\boldsymbol{\theta}} J_{\mathrm{AML}}]^{\top}$ factorises in this particular way, a simple iterative algorithm is suggested. Here, a new estimate $\boldsymbol{\theta}_{i+1}$ is generated by finding the eigenvector of $\boldsymbol{X}_{\boldsymbol{\theta}_i}$ corresponding to the eigenvalue closest to zero.

The ordinary FNS algorithm does not take into account any ancillary constraint imposed on the parameters. However, it can be 'upgraded' to incorporate such a constraint by replacing $\boldsymbol{X}$ with a different matrix. This new matrix is derived by making a basic

assumption on the form of the constraint, namely that it must be k-homogeneous. Various manipulations of an initial form lead to the eventual description of a constrained estimation method, CFNS.

The following two chapters explore the application of these estimation methods to the problems of conic fitting and fundamental matrix estimation.

Chapter 4

APPLICATION I: CONIC FITTING

In this chapter estimation methods are applied to the task of fitting a *conic section* – an ellipse, parabola, or hyperbola – to planar data points. After considering this general problem of conic fitting, the harder problem of specifically fitting an ellipse is addressed. FNS is compared to several well known conic and ellipse fitting methods. Experiments show not only the efficacy of the FNS, but also highlights practicalities of the estimation process.

Ellipse estimation, or in general conic fitting, is a fundamental task in so-called 'early' image analysis, enabling a significant amount of raw information encoded in an image to be represented by a small number of parameters. It is an image analysis task with many applications. The perspective projection of a circle is an ellipse, hence ellipse fitting is used in many applications where calibration circles are included in a scene to be imaged. Later analysis can be performed whereby the centre points of estimated ellipses can be used for calibration and other higher-level analysis. Conic fitting is also applied to many other areas: describing raster images using 'vector' primitives [49] and interpreting results from physics experiments [33] are but two examples.

Putting into practice estimation theory of the previous chapters via the simply visualised problem of 2-D ellipse fitting can be useful for other related estimation tasks, because the parameters estimated have an easy 2-D visualisation. It is straightforward to directly draw an estimated conic and data points, therefore obtaining an immediate visual sense of the quality of the estimate. Such a visualisation is more difficult for higher order problems where the quantity estimated must first be manipulated, often non-trivially, to be rendered.

Methods used for estimating ellipses, or fitting conics, may be partitioned into two categories: model based and least-squares methods. The model based methods choose an estimate by considering a specific geometric model applied to a large set of data. Typically these methods deal with data which is very noisy, and sometimes they may need to estimate more than one conic for a given set of data. Often, data is parti-

tioned and multiple candidate solutions are chosen. Such methods are often based on a Kalman filter [51] or a Hough transform [53]. Using a Hough transform to estimate ellipses involves large state spaces which can lead to prohibitive time and space requirements. Many methods exploit gradient information or geometric properties [38] to reduce the size of the state space. More recently, evolutionary strategies such as genetic algorithms have been applied to further improve performance and tractability of the Hough transform method [33, 67].

Model based methods are 'big stick' techniques which have significant performance implications. Indeed, some of these methods measure execution time in *hours* [33]. Alternative methods, using least squares minimisation, are the subject of the rest of this chapter.

Least squares methods aim to fit a single conic section to a given set of input data points. A parametric model of a conic is given and used to define a residual function comparing a candidate curve with the data. Minimisation of the squared sum of residuals yields a unique estimate. In the following chapter, the parametric model of Chapter 2 is adopted, where there is no assumption made of the distribution of the set of ideal data points. Newsam and Redding [46] show that in some specific cases, assuming a distribution of ideal data points is advantageous.

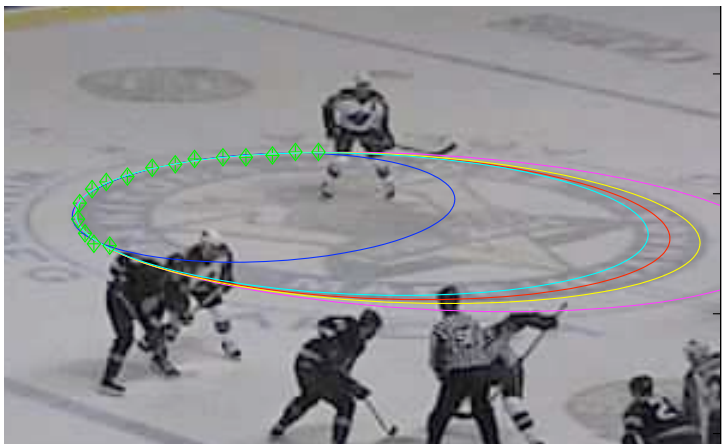


Figure 4.1: Several ellipses estimated from a set of points

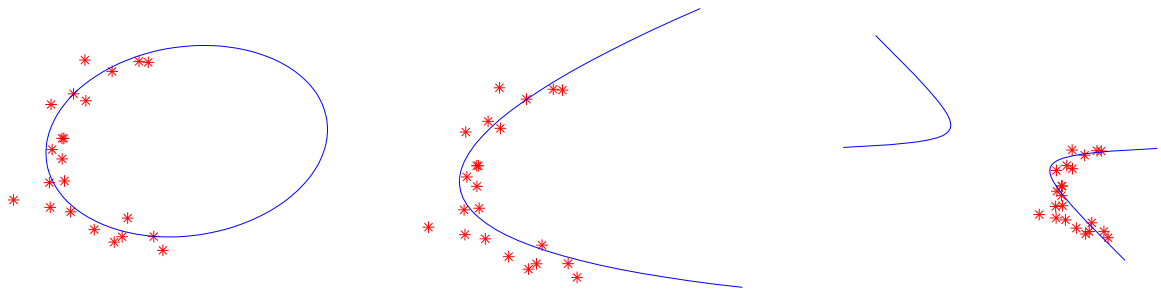


Figure 4.2: An example of an ellipse, parabola, and hyperbola which have been estimated given the same sample set of data points

4.1 Problem form

The general equation describing a conic section is

$$ax^2 + bxy + cy^2 + dx + ey + f = 0. \tag{4.1}$$

As this equation is linear in parameters and quadratic in data it can be expressed as per Eq. (2.2), $\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}) = 0$. The parameter vector is simply taken to be $\boldsymbol{\theta} = [a, b, c, d, e, f]^\top$, and the carrier vector is

$$\boldsymbol{u}(\boldsymbol{x}) = \left[x^2, xy, y^2, x, y, 1\right]^\top, \tag{4.2}$$

with each data point represented by the data vector $\boldsymbol{x} = [x, y]^\top$.

The conic equation may also be equivalently expressed in matrix form. Eq. (4.1) can be written

$$\boldsymbol{m}^\top \boldsymbol{Q}\, \boldsymbol{m} = 0, \tag{4.3}$$

with $\boldsymbol{m} = [x, y, 1]^\top$ and

$$\boldsymbol{Q}(\boldsymbol{\theta}) = \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix}. \tag{4.4}$$

This general conic equation describes a curve which may be a parabola, hyperbola, or an ellipse. For Eq. (4.3) to specify an ellipse the *discriminant* $\Delta = b^2 - 4ac$ must be negative. For $\Delta = 0$ the conic section is a parabola, and for $\Delta > 0$ it is a hyperbola. Fig. 4.1 shows several different conics estimated given a set of points. The three forms possible are shown in Fig. 4.2, where an ellipse, parabola, and hyperbola have been fit to the same data points.

The general conic fitting problem involves estimating coefficients of Eq. (4.1) (the elements of the parameter vector $\boldsymbol{\theta}$) given a set of planar data points. Depending on the data points and the method of generating an estimate, any type of conic section may result. A more specific problem is that of (constrained) ellipse fitting, where the best *ellipse* fitting the data points is sought. In this case, an ancillary constraint must be imposed, either directly through the specific design of an estimator, or as a post correction where a non-ellipse estimate generated from a general method is 'upgraded' to an ellipse.

Firstly, the general conic fitting problem is considered. Later, ellipse fitting is explored with the imposition of an additional constraint.

4.2 Unconstrained estimation

Least squares conic fitting methods can be divided into two classes: algebraic and geometric. The algebraic methods generate solutions which minimise the sum of squares of residuals of the (algebraic) conic equation. As described in Sect. 2.4.1, the algebraic quantities are used typically because solutions are easily found through algebraic manipulation and not because they have any relationship to the underlying geometry of the problem. In contrast, geometric algorithms aim to minimise a quantity which has geometrical significance. However, geometrically inspired quantities are more difficult to compute.

As the problem form is exactly that of Chapter 2, all of the methods described previously can be applied directly to conic fitting.

4.2.1 Applying algebraic least squares

The family of algebraic methods, which minimise the squares of the residuals of the conic equation (Eq. (4.1)), can be applied to conic fitting by simply adopting the carrier form of Eq. (4.2). These methods minimise a cost function whose general form is

$$J_{\mathrm{ALG}}(\boldsymbol{\theta}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \frac{|\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x})|^2}{\boldsymbol{\theta}^\top \boldsymbol{C}\, \boldsymbol{\theta}} \,. \tag{4.5}$$

The TLS method (Sect. 2.4.1), corresponds to the choice $\boldsymbol{C} = \boldsymbol{I}$, applying an effective normalisation of $a^2 + \cdots + f^2 = 1$. As shown in that section, the estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{TLS}}$ can

be found by taking a SVD of the design matrix

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{u}(\boldsymbol{x}_1)^\top \\ \vdots \\ \boldsymbol{u}(\boldsymbol{x}_n)^\top \end{bmatrix}.$$

Other normalisations can be applied by choosing, a potentially rank deficient, form of $\boldsymbol{C}$. Such normalisations are designed by considering the form of Eq. (4.1) and hence are directly applicable to conic fitting.

Bookstein's normalisation

Bookstein [5] proposed the normalisation $a^2 + \frac{1}{2}b^2 + c^2 = 1$, corresponding to a choice of $\boldsymbol{C} = \mathrm{diag}(1, \frac{1}{2}, 1, 0, 0, 0)$. Estimates produced under this normalisation are invariant to Euclidean transformations to the data points. This normalisation is *conic invariant* which means that minimising algebraic residuals subject to this constraint will produce a consistent estimate regardless of arbitrary Euclidean transformations of the data. Estimates produced by the TLS normalisation do not have this property. Rather, TLS estimates will depend on the exact position and scale of the data points. This artifact is shown in Fig. 4.3. In the figure, a TLS estimate is produced given some example data points. Then a common Euclidean transformation is applied to the estimate and the data. A new TLS estimate is produced for the transformed data, and is shown superimposed on the transformed original estimate. The transformed estimate does not coincide with the estimate generated from the transformed data points. In the case of estimation under the Bookstein constraint, shown below, the transformed original estimate and the estimate generated from the transformed data are seen to be the same.

For TLS normalisation, the minimiser may be found directly through an eigensystem (Sect. 2.4.2). Bookstein shows how a similar system can be derived for his invariant normalisation, and the solution found using an eigendecomposition. Gander *et al* [17] describe how a more efficient and numerically sound SVD can be used to find the Bookstein estimate.

Other normalisations

Several other normalisations have been employed. Notably

- $a + c = 1$, and

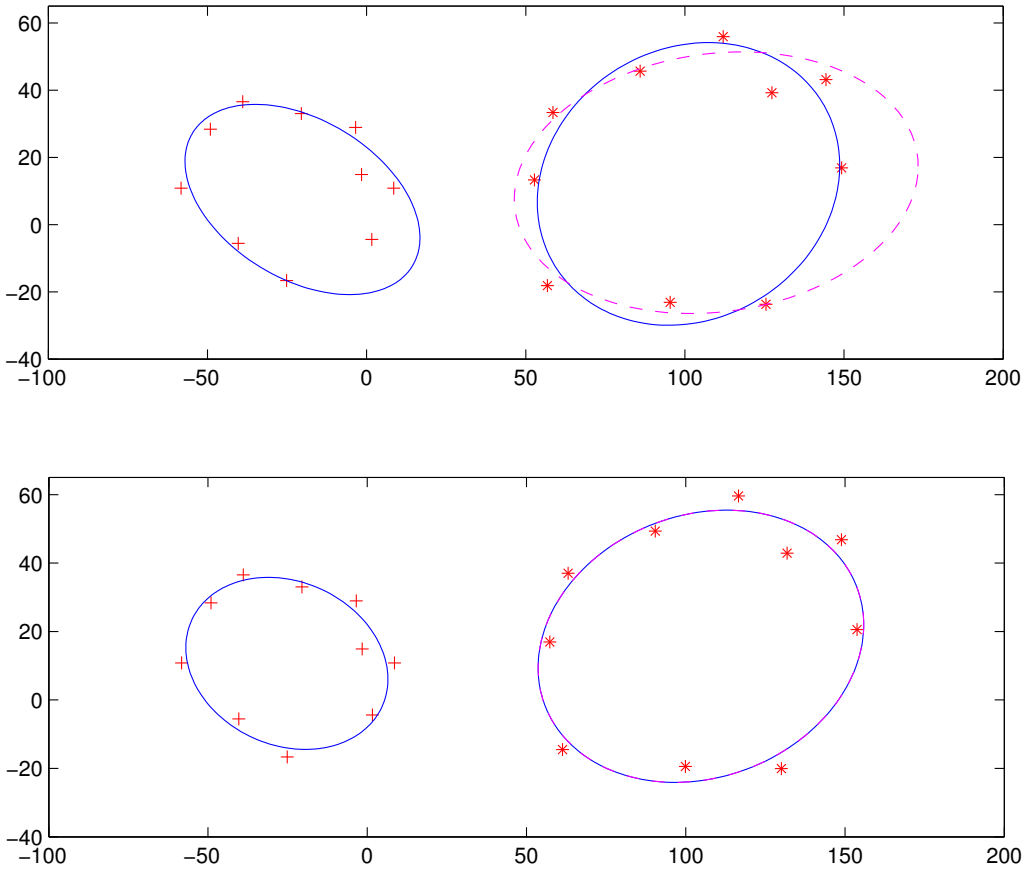


Figure 4.3: Above: A set of data points and associated TLS estimate is shown on the left. The points were transformed by applying a translation of $[170, 5]^\top$, a rotation of 35^o, and scaling by a factor of 1.6. A newly estimated TLS estimate is shown, along with the same transformations applied to the original estimate. Below: The Bookstein method is used in place of TLS and the newly estimated ellipse coincides with the transformed original

- $f = 1$

These have been used for example by Rosin [54, 55], and Ellis *et al* [9] among others. The case of $a + c = 1$ is invariant to Euclidean transformations of the data for ellipses only. Estimates may be computed under these normalisations by performing a SVD whose precise form is determined by algebraic manipulation similar to that used in the case of the Bookstein constraint [68].

4.2.2 Geometric interpretation for bias in algebraic methods

Under the assumption that each element of data is affected by the same level of noise, a cost function should provide equal weights to each residual. But in the case of the residuals of the conic equation, data points contribute unequal weightings. This varied contribution was first described Cooper and Yalabik [8]. Zhang describes a geometric interpretation of why this is the case [68].

Fig. 4.4 shows a 'canonical' ellipse: its centre is at the origin, and the major and minor axes are aligned to the coordinate axis. In this special case, the conic equation is reduced to

$$ax^2 + cy^2 + f = 0. \tag{4.6}$$

For a data point $\boldsymbol{x}_i = [x_i, y_i]^\top$, define $\tilde{\boldsymbol{x}}_i = [\tilde{x}_i, \tilde{y}_i]^\top$ as the point on the line joining $\boldsymbol{x}_i$ to the origin which intersects the circumference of the ellipse (see Fig. 4.4). Assuming this line makes an angle γ with the $x-$axis, then $\cot\gamma = \tilde{x}_i/\tilde{y}_i = x_i/y_i$ and so $x_i = \tilde{x}_i(y_i/\tilde{y}_i)$. The algebraic residual of the point $\boldsymbol{x}_i$ is

$$\begin{aligned} f(\boldsymbol{\theta}; \boldsymbol{x}_i) &= ax_i^2 + cy_i^2 + f \\ &= a\left(\tilde{x}_i^2\left(\frac{y_i^2}{\tilde{y}_i^2}\right)\right) + cy_i^2 + f \\ &= \frac{y_i^2}{\tilde{y}_i^2}\left(a\tilde{x}_i^2 + c\tilde{y}_i^2\right) + f \\ &= \frac{y_i^2}{\tilde{y}_i^2}\left(a\tilde{x}_i^2 + c\tilde{y}_i^2 + f\right) - \frac{y_i^2}{\tilde{y}^2}f + f \\ &= -f\left(\frac{y_i^2}{\tilde{y}_i^2} - 1\right) \qquad (\text{as } a\tilde{x}_i^2 + c\tilde{y}_i^2 + f = 0). \end{aligned} \tag{4.7}$$

Again, considering the angle γ, leads to the identities $\tilde{y}_i = e_i \sin\gamma$ and $y_i = (e_i + d_i)\sin\gamma$. Substituting them into Eq. (4.7) yields the relationship

$$f(\boldsymbol{\theta}; \boldsymbol{x}_i) \propto \left(\frac{d_i + e_i}{e_i}\right)^2 - 1,$$

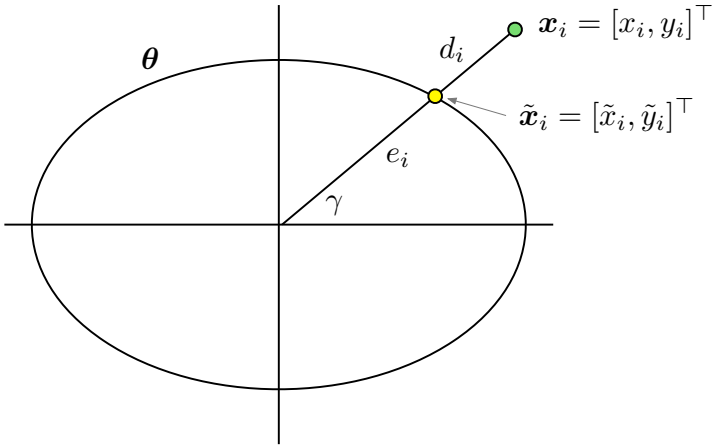


Figure 4.4: Bookstein geometry

where the algebraic residual of a point is related to the ratio of the distances d_i and e_i. At points of high curvature, e_i will be proportionately larger and hence $|f(\boldsymbol{\theta}; \boldsymbol{x}_i)|$ will be smaller. For low curvature, the reverse is true. Therefore, points close to low curvature sections will contribute more to the cost function of sum of squared residuals. Consequently, any algebraic method will tend to produce estimates which exhibit lower total curvature. This is a geometric interpretation of the uneven weighting effecting algebraic residuals.

4.2.3 Estimation methods

All of the geometric estimation methods from Chapter 3 can be directly applied to conic fitting. In addition to adopting the carrier vector of Eq. (4.2), the gradient vector is taken as

$$\partial_{\boldsymbol{x}} \boldsymbol{u}(\boldsymbol{x}) = \begin{bmatrix} 2x & y & 0 & 1 & 0 & 0 \\ 0 & x & 2y & 0 & 1 & 0 \end{bmatrix}^\top . \tag{4.8}$$

Geometric methods used in the experiments are:

- **TAU:** Taubin's meta-approximation method (Sect. 3.2.1)
- **FNS:** The fundamental numerical scheme (Sect. 3.2.3),
- **LM-A:** Direct Levenberg-Marquardt minimisation of J_{AML} (Sect. 3.1.3). The MATLAB optimisation function `lsqnonlin`, an implementation of the LM algorithm,

was used.

Algebraic methods used are:

- **TLS:** Total least squares, and
- **BOOK:** Bookstein's method.

Both of these methods use a SVD in their implementation. The MATLAB SVD implementation was used in the subsequent testing.

4.3 Conic fitting experiments

For the following experiments estimators are tested using synthetic data generated using a 'default' true ellipse. This ellipse, part of a 400×600 pixel image, is centred at $(200, 150)$, has a major axis of 300, a minor axis of 200, and is rotated (anti-clockwise) by 0.4 radians. The true ellipse is denoted $\boldsymbol{\theta}^*$. A set of 30 points is selected along each of two arcs covering the highest and lowest curvature portion of the default ellipse. Each arc has length equal to one third of the perimeter. The data set covering the high curvature portion is labelled 'A', and the data set on the lowest curvature section is labelled 'B'. In each case, underlying true data points are distributed evenly around the arc. The default ellipse, and the true data points for each data set, are shown in Fig. 4.5.

Noisy data points are generated by perturbing each true data point with Gaussian noise. For these experiments only homogeneous and isotropic noise is used; all covariance matrices are taken as $\boldsymbol{\Lambda}_{\boldsymbol{x}_i} = \sigma \boldsymbol{I}$ for $i = 1 \ldots n$. Fig. 4.6 shows some example data sets for increasing noise levels.

Given noisy data, each method will produce its own estimate. These estimates can be visualised by rendering the conic section they describe. In addition, error measures are employed to discriminate carious estimates.

4.3.1 Error measures

Choosing an error measure for the purpose of discerning estimates must be done with care. It is not always apparent what is a good measure to choose, and it is important to be careful to understand the implications of a particular choice. Particularly sensitive are those measures which are directly related to estimation methods.

- The first error measure used to compare estimates is $J_{\mathrm{AML}}(\widehat{\boldsymbol{\theta}}, \{\boldsymbol{x}_i\})$. This error measure will permit checking whether FNS minimises the cost function that it was designed to minimise. Using this measure, the convergence and correctness of the FNS can be compared directly with LM-A.

- Another potential error measure is the true maximum likelihood cost function, $J_{\mathrm{ML}}(\widehat{\boldsymbol{\theta}}, \{\boldsymbol{x}_i\})$. This function is the sum of squared distances between data points and the corresponding closest points on the conic represented by $\widehat{\boldsymbol{\theta}}$. This measure is useful as it allows a comparison between the real ML cost function and its approximated form, the AML cost function.

The above error measures are based solely on the estimate and the given data. If the data has been generated synthetically by adding random perturbations to underlying ideal values $\{\boldsymbol{x}_i^*\}$, then the following alternative measures can be used:

- $\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\|$. This measure has the property that it will vanish in the case that the estimate is perfect, $\widehat{\boldsymbol{\theta}} = \boldsymbol{\theta}^*$. However, this measure uses the values of $\boldsymbol{\theta}$ directly, which are the algebraic coefficients of the conic equation. This is an undesirable property as these values have no significant geometric meaning.

- $J_{\mathrm{ML}}^*(\widehat{\boldsymbol{\theta}}) = \sum_{i=1}^{n} \|\boldsymbol{x}_i^{\widehat{\boldsymbol{\theta}}} - \boldsymbol{x}_i^*\|^2$, where $\boldsymbol{x}_i^{\widehat{\boldsymbol{\theta}}}$ denotes the orthogonal projection of the data point $\boldsymbol{x}_i$ onto the estimated conic $\widehat{\boldsymbol{\theta}}$ and $\boldsymbol{x}_i^*$ is the ideal data point. This error measure also has the property of vanishing when $\widehat{\boldsymbol{\theta}} = \boldsymbol{\theta}^*$.

4.3.2 Minimising the AML cost function

The first set of experiments relate to FNS and the way it minimises the AML cost function. Firstly, the actual minimisation process with use of FNS is addressed. Fig. 4.7 shows the J_{AML} values of the current estimate after each iteration of the FNS algorithm. In these examples FNS was seeded with $\widehat{\boldsymbol{\theta}}_{\mathrm{TLS}}$. The value of $J_{\mathrm{AML}}(\widehat{\boldsymbol{\theta}}_{\mathrm{TAU}})$ is shown for reference. The sequence of estimates is not always decreasing in terms of J_{AML} values, and several examples of this can be seen. Fig. 4.8 shows the conic represented by the estimate after each iteration of FNS.

Based on the analysis of the above results, the following values were determined to use as stopping condition tolerances:

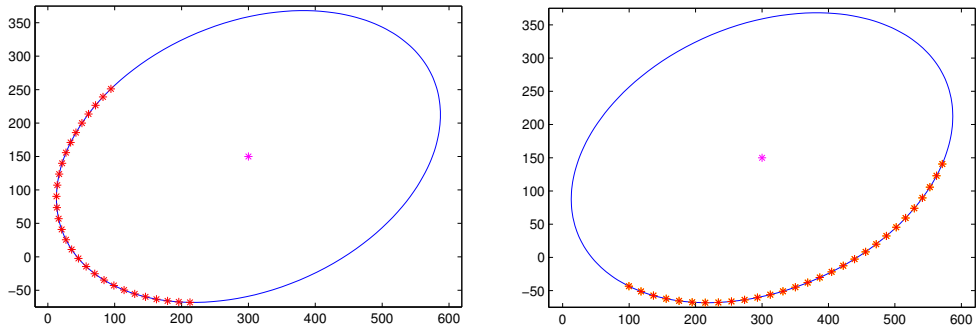


Figure 4.5: An example of an underlying 'true' ellipse with 30 true data points distributed along its arc length. Data set A (left) and B(right)

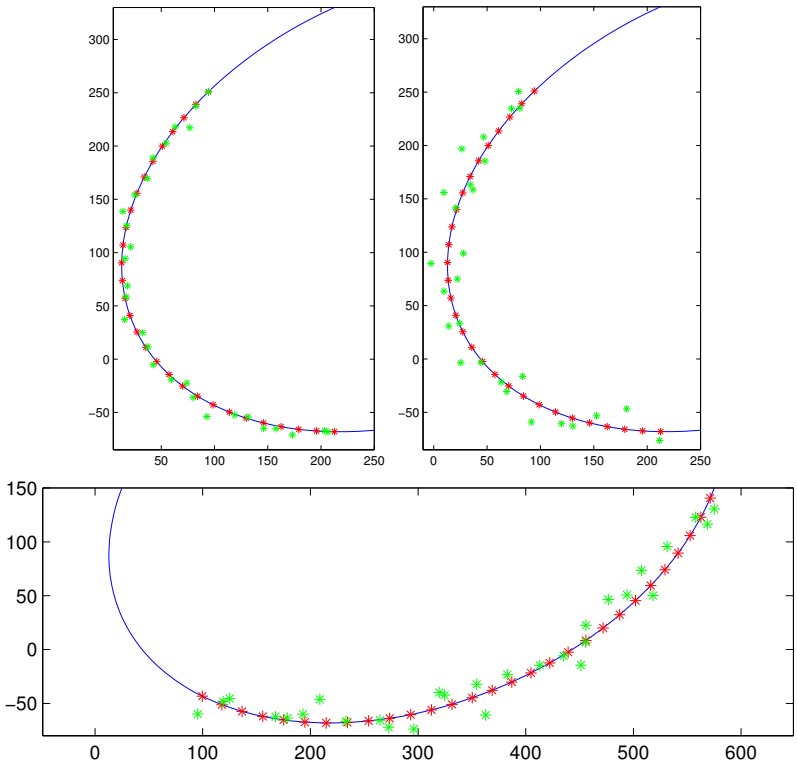


Figure 4.6: Above: noisy data for data set A, with $\sigma = 5.0$ pixels (left), and $\sigma = 10$ pixels (right). Below: noisy data for data set 'B' with $\sigma = 10$ pixels

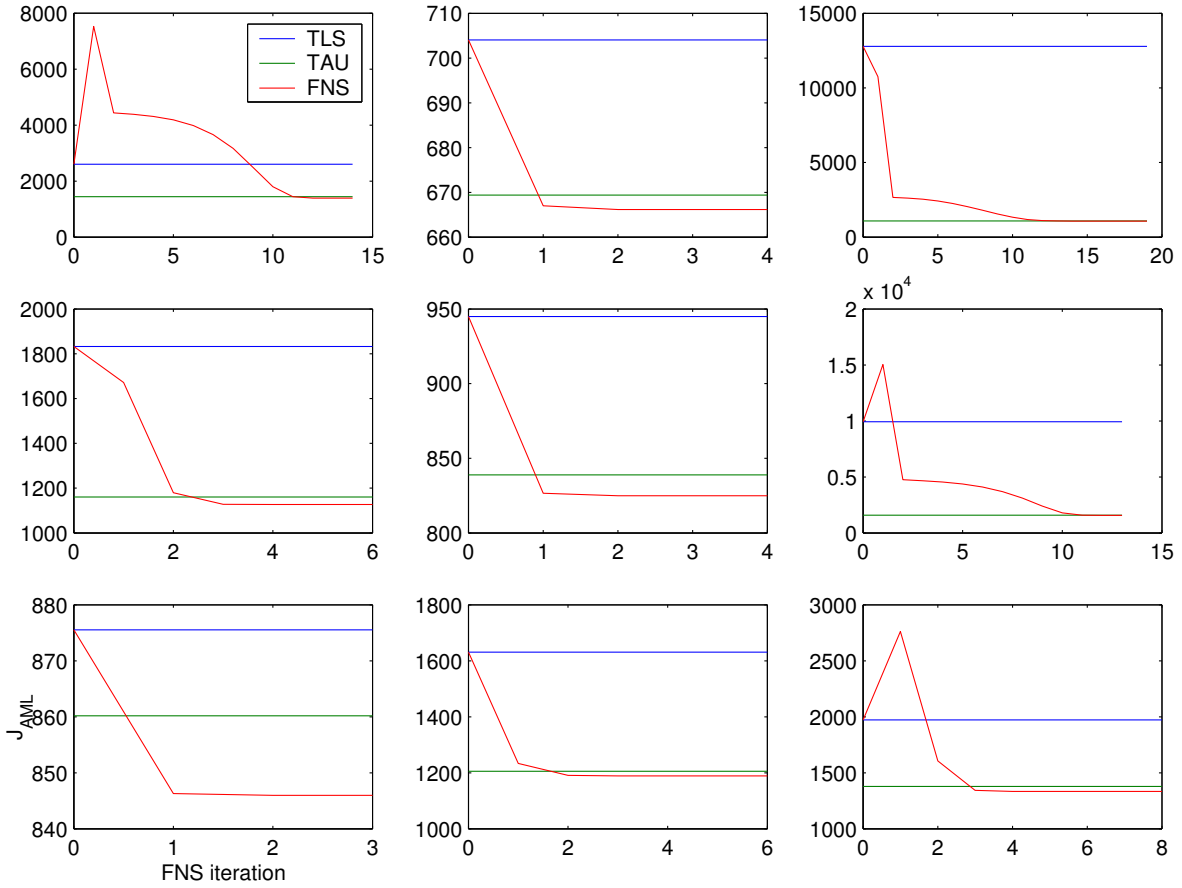


Figure 4.7: Examples of convergence of FNS for $\sigma = 7.0$ using data set A. On occasion, early iterations see an initial increase in the value of J_{AML}

- $\varepsilon_\theta = 1.0 \times 10^{-5}$ for the difference between successive estimates, and
- $\varepsilon_J = 1.0 \times 10^{-2}$ for the difference between successive values of J_{AML}.

The same values were used when running FNS and LM-A, to enable sensible comparison. In addition, for both methods a "safety" maximum number of iterations was adopted. For FNS, this was taken as 40. For LM-A, which often requires significantly more iterations to converge, the value was set to 200.

It is clear that, in the above examples, only a small number of iterations are required for convergence of the FNS algorithm. Figure 4.9 shows the average number of iterations, averaged over 200 trials, required for convergence for increasing amounts of noise added to the data points. The results were obtained by performing the following procedure:

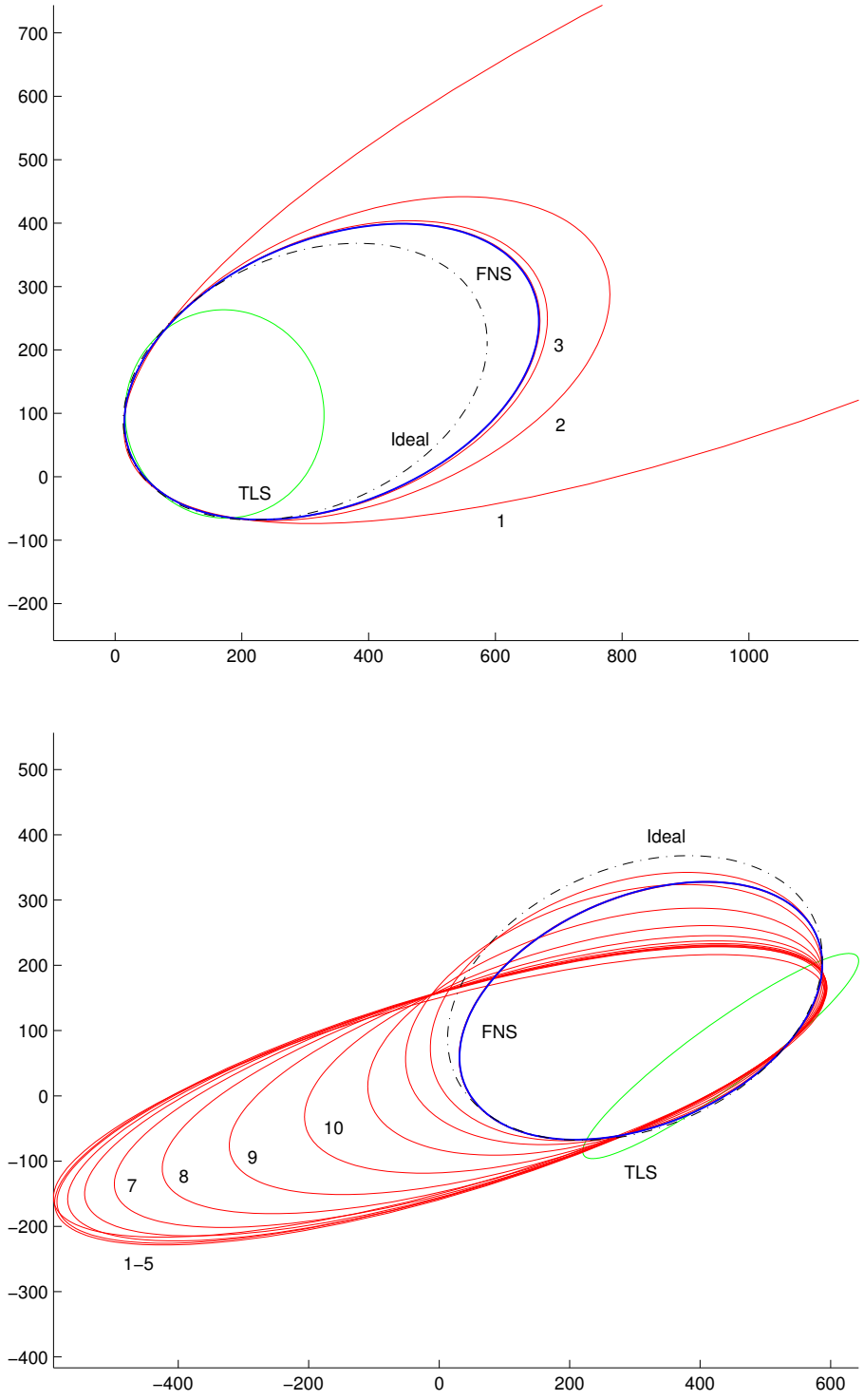


Figure 4.8: Individual FNS iterations shown for set A (left), and set B (right)

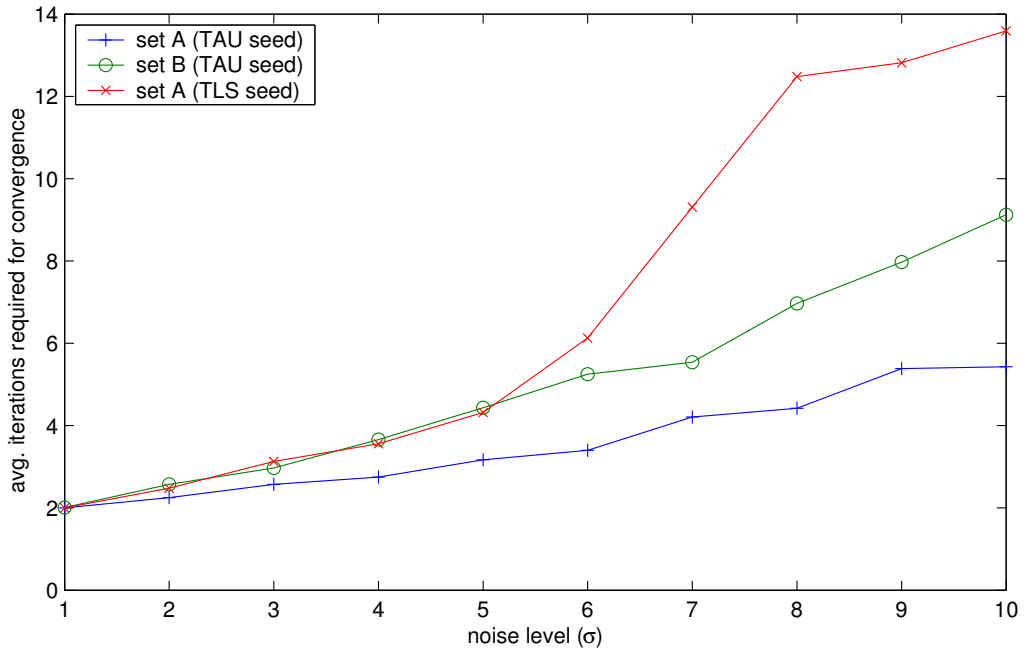


Figure 4.9: Average number of iterations required for convergence of FNS for increasing noise

1. Start with a given set of true data
2. Iterate over noise levels $\sigma = 1.0 \rightarrow 10.0$. For each noise level:
   (a) repeat 200 times:
      i. generate a new set of noisy data
      ii. compute the FNS estimate and count the number of iterations required
   (b) compute the average number of iterations

Although the number of iterations required steadily increases for higher noise, overall very few iterations are typically required for convergence.

If, instead of choosing data along an arc along the highest curvature part of the ellipse (data set A), data along the flattest part (data set B) are chosen, then the minimisation task becomes more difficult especially for higher levels of noise. The convergence of the FNS algorithm is shown in this case in Figure 4.10. Here, the benefit of seeding with the Taubin estimate is clear. Because the TLS estimate is so poor, seeding FNS with this estimate causes, in some cases, wild fluctuations before eventual convergence. If

Meth.	$\sigma = 2$	4	6	8	10
LM-A	0.724	0.953	2.17	1.53	3.175
FNS	0.066	0.095	0.153	0.163	0.188

Table 4.1: Average execution time for LM and FNS (in seconds) when using data set B. Values were obtained using the `cputime function in MATLAB`

the noise level is lifted higher still, the algorithm is shown to diverge, or converge to a non-local minimum.

4.3.3 Comparing LM-A to FNS

It is evident that FNS typically converges rapidly to a solution. The question remains as to whether it converges to the 'right' place. To check this, FNS estimates can be compared to those generated using Levenberg-Marquardt directly minimising J_{AML}, the LM-A method. Experiments were conducted as follows. Random noise, with $\sigma = 7.0$, was added to a set of canonical points 200 times. Each time, an FNS, LM-A, and TLS estimate was generated with its J_{AML} value recorded. Fig. 4.11 shows histograms of these values.

Both methods were seeded with the Taubin estimate. They also had the same values for the stopping tolerances. Figure 4.12 shows the percentage difference between the J_{AML} values for the FNS and TAU estimates, for both data sets. Most differences are within the stopping tolerance of 10^{-3}. In the case of data set B however, there are a handful of cases where the FNS and LM-A estimate differer by as much as 4%.

The difference in execution times between the two methods is shown in Table 4.1. Clearly, the FNS method requires significantly less execution time than LM-A.

facilities

The form of the J_{AML} cost function is shown in Figures 4.13. There are six graphs, one for each element of $\boldsymbol{\theta}$. Each graph is produced by evaluating a specially constructed parameter vector $\boldsymbol{\theta}_P$. For the jth graph, the elements of $\boldsymbol{\theta}_P$ are defined as

$$\theta_{i,P} = \begin{cases} \hat{\theta}_{i,\mathrm{FNS}} + k\hat{\theta}_{i,\mathrm{TLS}} & \text{if } i = j \\ \hat{\theta}_{i,\mathrm{FNS}} & \text{otherwise.} \end{cases}$$

The graph shows the values of $J_{\mathrm{AML}}(\boldsymbol{\theta}_P)$ as k goes from -1 to 1.

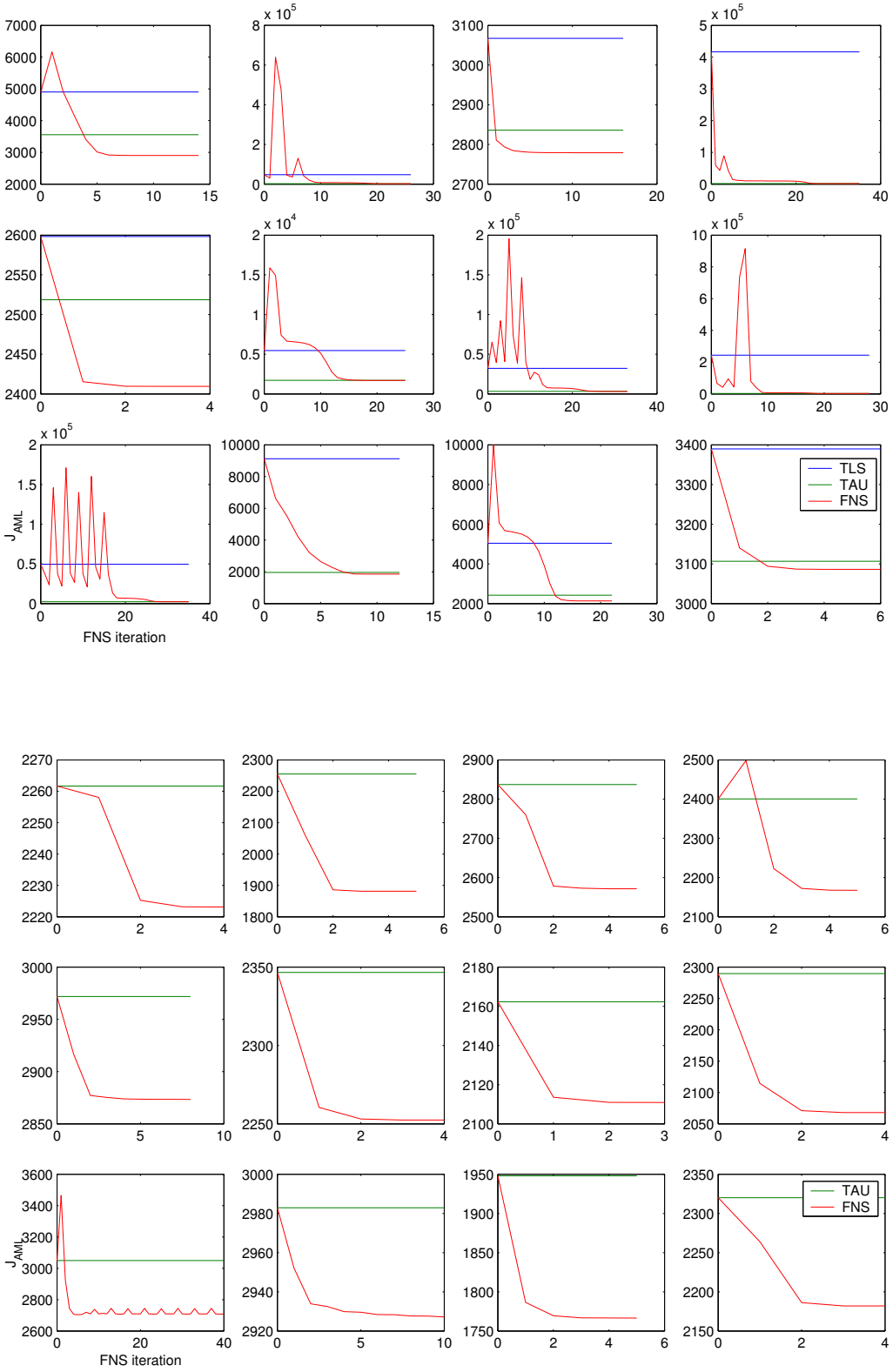


Figure 4.10: Using data set B and adding noise of $\sigma = 10$. The top graphs show convergence when FNS was seeded with BOOK. The bottom graphs show the improvement in convergence when FNS was seeded with TAU

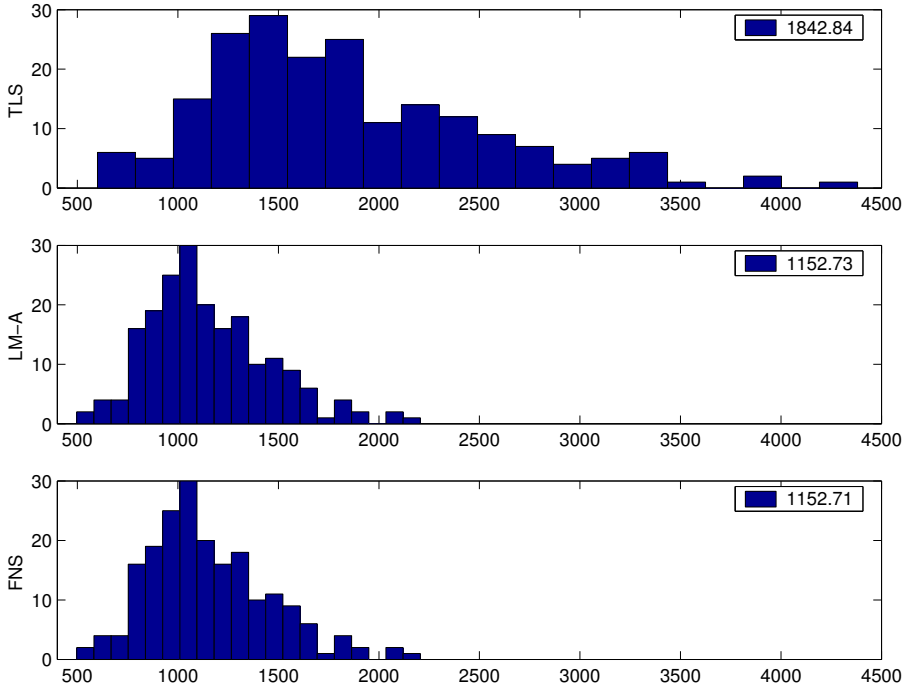


Figure 4.11: A histogram of J_{AML} values for FNS, LM-A, and TLS estimators for 200 trials, where $\sigma = 7.0$. The average value is shown as a legend for each histogram

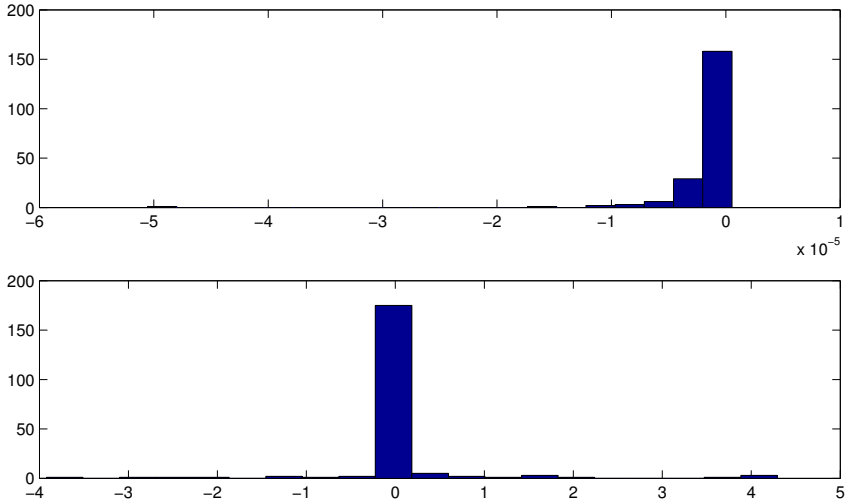
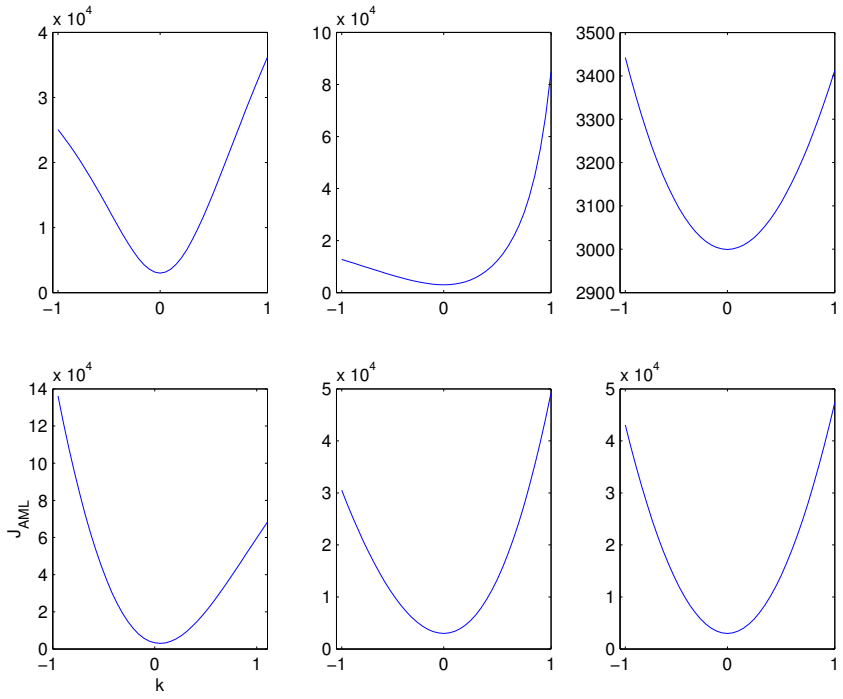


Figure 4.12: Histograms of the percentage difference between LM-A and FNS estimates for each of 200 trials. The noise level was taken as $\sigma = 10.0$ and data set A (upper) and B (lower) were both used. A positive value indicates that the FNS estimate has a lower value of J_{AML} than LM-A



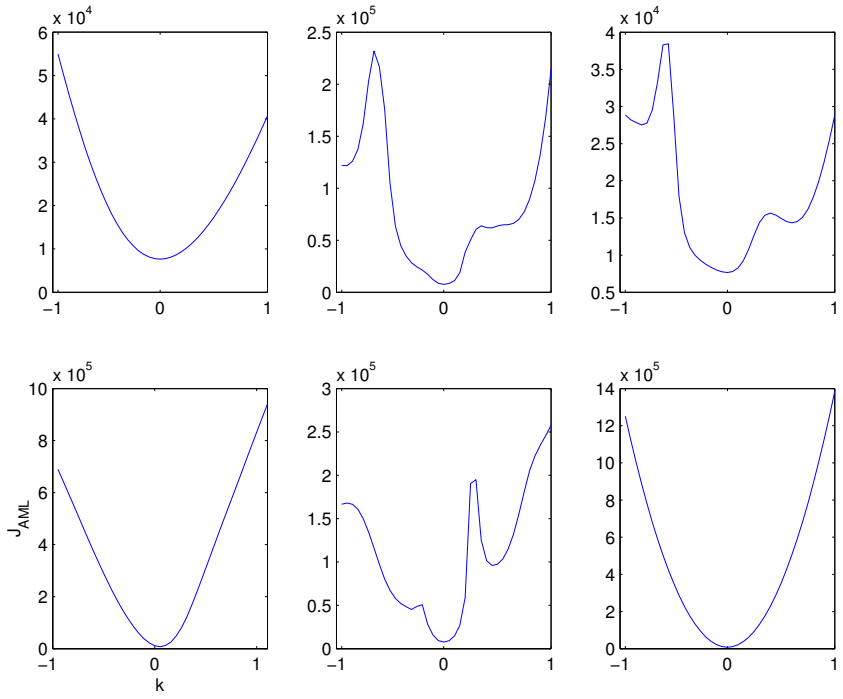


Figure 4.13: Slices of J_{AML} generated for $\sigma = 7.0$

4.4 Ellipse fitting

The problem of ellipse fitting is now considered. Here, an ellipse is sought that best fits a given data set even though a different kind of conic section may best fit that data. This is a common problem, particularly when it is known that the data points have been derived from an ellipse. The problem is somewhat ill-posed. Note , for example, that when the data points lie on a straight line, no finite ellipse fits them best.

4.4.1 Ellipse gold standard

An ellipse "gold standard" (GS) estimator directly minimises the ML cost function. It ensures that any estimate produced is an ellipse. This is achieved by adopting an ellipse-specific geometric re-parameterisation. The algebraic coefficients in $\boldsymbol{\theta}$ are replaced with the geometric parameters

$$\boldsymbol{g} = \left[c_x, c_y, r_a, r_b, \gamma\right]^\top, \tag{4.9}$$

where (c_x, c_y) is the centre of the ellipse, r_a and r_b are the major and minor axes respectively, and γ is the angle of rotation between the ellipse's major axis and the $x-$axis (See Fig. 4.14).

Expressed in terms of the above parameterisation, the conic equation, Eq. (4.1), is

$$\frac{(x\cos\gamma + y\sin\gamma - c_x\cos\gamma - c_y\sin\gamma)^2}{r_a^2} + \frac{(-x\sin\gamma + y\cos\gamma + c_x\sin\gamma - c_y\cos\gamma)^2}{r_b^2} - 1 = 0 \tag{4.10}$$

Noise is assumed to be homogeneous and isotropic so that for a given ellipse $\boldsymbol{g}$ and data point $\boldsymbol{x}_i$, $\overline{\boldsymbol{x}}_i^{\boldsymbol{g}}$ denotes the the point on the ellipse closest in Euclidean distance to $\boldsymbol{x}_i$. Geometrically, $\overline{\boldsymbol{x}}_i^{\boldsymbol{g}}$ is the point on the ellipse such that the line joining $\boldsymbol{x}_i$ and the ellipse is orthogonal to the tangent. The next section gives details of how $\overline{\boldsymbol{x}}_i^{\boldsymbol{g}}$ can be directly computed given $\boldsymbol{x}_i$ and $\boldsymbol{g}$.

The estimate is defined as

$$\widehat{\boldsymbol{g}}_{\mathrm{GS}} = \arg\min_{\boldsymbol{g}} J_{\mathrm{ML}}(\boldsymbol{g}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \sum_i \left(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i^{\boldsymbol{g}}\right)^\top \left(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i^{\boldsymbol{g}}\right). \tag{4.11}$$

This cost function can be minimised by using, for example, the Levenberg-Marquardt method (Sect. 3.1.3). Because at every iteration a fourth order polynomial needs to be solved, for each data point, the GS method is computationally very expensive.

Given an estimated ellipse represented in the geometric parameterisation by $\widehat{\boldsymbol{g}}$, an equivalent algebraic parameterisation may be computed directly. Let

$$\boldsymbol{R} = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \boldsymbol{T} = \begin{bmatrix} 1 & 0 & -c_x \\ 0 & 1 & -c_y \\ 0 & 0 & 1 \end{bmatrix} .$$

And let

$$\boldsymbol{Q} = (\boldsymbol{RT})^\top \begin{bmatrix} r_a^{-1} & 0 & 0 \\ 0 & r_b^{-1} & 0 \\ 0 & 0 & -1 \end{bmatrix} (\boldsymbol{RT}) . \tag{4.12}$$

Then the algebraic parameters are obtained from $\boldsymbol{Q}$ from the formula

$$\widehat{\boldsymbol{\theta}} = [q_{1,1}, 2q_{1,2}, q_{2,2}, 2q_{1,3}, 2q_{3,2}, q_{3,3}]^\top . \tag{4.13}$$

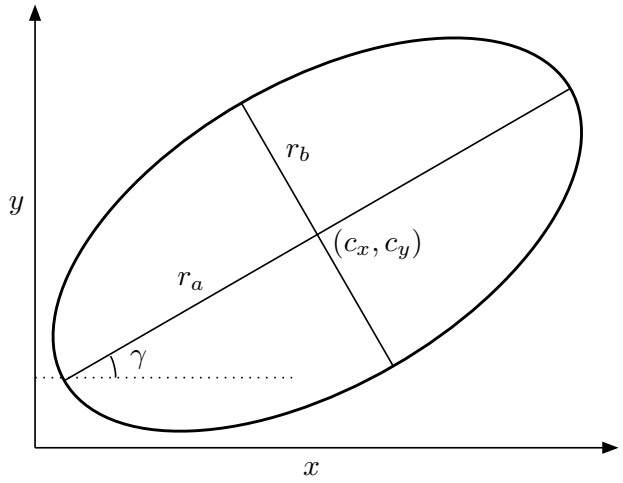


Figure 4.14: Geometric parameterisation of an ellipse

4.4.2 Finding the closest point

The problem of finding the closest point is much easier to deal with for a canonical ellipse, which is centred at the origin and has major and minor axes aligned with the coordinate axes. Such an ellipse is described by the equation $(x/a)^2 + (y/b)^2 - 1 = 0$ (for $a \neq 0$ and $b \neq 0$). In terms of the general conic equation, the parameter vector is

$\boldsymbol{\theta} = [1/a^2, 0, 1/b^2, 0, 0, -1]^\top$. The problem now is to find the closest point on this ellipse to an arbitrary point $\boldsymbol{p} = [u, v]^\top$. It may be phrased as a constrained minimisation problem: minimise $f(x, y) = (x-u)^2+(y-v)^2$ subject to $g(x, y) = (x/a)^2+(y/b)^2-1 = 0$. By using the method of Lagrange multipliers, this problem reduces to solving the system

$$\begin{bmatrix} 2x - 2u \\ 2y - 2v \end{bmatrix} = \lambda \begin{bmatrix} 2x/a^2 \\ 2y/b^2 \end{bmatrix} \tag{4.14}$$

for some scalar λ. In terms of x and y the above system is the same as to the equations $x = a^2u/(\lambda + a^2)$ and $y = b^2v/(\lambda + b^2)$. Substituting these values back into the original ellipse equation gives

$$\left(\frac{au}{\lambda + a^2}\right)^2 + \left(\frac{bv}{\lambda + b^2}\right)^2 = 1.$$

Multiplying through by both denominators leads to a fourth order polynomial parametric equation

$$\begin{aligned} p(\lambda) &= (\lambda + a^2)^2(\lambda + b^2)^2 - a^2u^2(\lambda + b)^2 - b^2v^2(\lambda + a^2)^2 \\ &= \lambda^4 + \left(2b^2 + 2a^2\right)\lambda^3 + \left(a^4 + b^4 + 4a^2b^2 - a^2u^2 - b^2v^2\right)\lambda^2 + \\ &\quad \left(2a^2b^4 + 2a^4b^2 - 2u^2a^2b^2 - 2b^2a^2v^2\right)\lambda + \left(a^4b^4 - u^2a^2b^4 - b^2v^2a^4\right) = 0. \end{aligned} \tag{4.15}$$

The largest real root $\widehat{\lambda}$ of $p(\lambda)$ represents the closest point on the ellipse. The coordinates (x, y) of the closest point may be found by substituting $\widehat{\lambda}$ back into Eq. (4.14). It is possible to use a specific method based on Newton's algorithm to find the largest root; however, a generic numerical polynomial root solver works well in practice.

That the polynomial $p(\lambda)$ is fourth order, and so has four potential solutions, makes sense geometrically. Given an arbitrary point, there can be up to four places on an ellipse at which the line joining that point to the ellipse is perpendicular to the ellipse itself. For example, if the point is the centre of the ellipse, then the lines joining that point with the ellipse along the major and minor axis will intersect the ellipse at four points. These four potential solutions correspond to the four (potentially real) roots of $p(\lambda)$.

This technique is easily adopted to a general ellipse: shift a given point appropriately, find the closest point on the canonical ellipse, then back-transform to obtain the closest point for the original ellipse. To improve numerical conditioning in the case of large values of r_a and r_b, the major and minor axes of the ellipse should be scaled so that the larger is equal to one. This should be done in addition to the translation applied to centre the ellipse at the origin.

4.5 Algebraic constrained ellipse fitting

The method of Fitzgibbon *et al* [15] is an important ellipse fitting algorithm, because it was the first to combine the properties of being computationally efficient and capable of producing estimates that are always ellipses. This method is non iterative, whereby the solution is found using a single generalised eigendecomposition. Because this method is non-iterative, it is a good choice for an initial input into more complicated iterative methods, especially those which require an ellipse rather than a general conic.

The Fitzgibbon method is based on algebraic least squares, and as such exhibits significant bias in certain cases. The bias is most obvious when the data points are random perturbations of real points confined to a subsection of half or less of the arc of an ellipse.

4.5.1 Finding the Fitzgibbon estimate

As described in earlier sections, when minimising $J_{\mathrm{LS}}(\boldsymbol{\theta})$ some additional constraint must be imposed on $\boldsymbol{\theta}$. The key idea underlying the Fitzgibbon method is to use this parametric constraint to impose the ellipse constraint. The general ellipse constraint is defined by the inequality $b^2 - 4ac < 0$, a form not particularly amenable to inclusion into a direct method. Fortunately, as the conic equation is homogeneous, any scalar multiple of $\boldsymbol{\theta}$ will represent the same conic; thus the scale of the constraint may be chosen arbitrarily. For example,

$$4ac - b^2 = 1. \tag{4.16}$$

This equality is quadratic in $\boldsymbol{\theta}$, and can be written in matrix form as $\boldsymbol{\theta}^\top \boldsymbol{F}\, \boldsymbol{\theta} = 1$, with

$$\boldsymbol{F} = \begin{bmatrix} \boldsymbol{F}' & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix}, \qquad \text{and} \quad \boldsymbol{F}' = \begin{bmatrix} 0 & 0 & 2 \\ 0 & -1 & 0 \\ 2 & 0 & 0 \end{bmatrix}. \tag{4.17}$$

The estimate is defined as the minimiser of $J_{\mathrm{LS}}(\boldsymbol{\theta})$ under the constraint $\boldsymbol{\theta}^\top \boldsymbol{F}\, \boldsymbol{\theta} = 1$. In exactly the same way as the TLS method *(Sect. 2.4.1), a Lagrange multiplier may be introduced, yielding

$$\Big[\partial_{\boldsymbol{\theta}} J_{\mathrm{LS}}(\boldsymbol{\theta})\Big]^\top - \lambda \left[\partial_{\boldsymbol{\theta}} (\boldsymbol{\theta}^\top \boldsymbol{F}\, \boldsymbol{\theta} - 1)\right]^\top = 0\,,$$

or

$$\Big[\partial_{\boldsymbol{\theta}} (\boldsymbol{\theta}^\top \boldsymbol{S}\, \boldsymbol{\theta})\Big]^\top - \lambda \left[\partial_{\boldsymbol{\theta}} (\boldsymbol{\theta}^\top \boldsymbol{F}\, \boldsymbol{\theta} - 1)\right]^\top = 0\,,$$

or

$$2\boldsymbol{S\theta} - 2\lambda\boldsymbol{F\theta} = 0\,. \tag{4.18}$$

Using this result, the required estimate must satisfy the system

$$\boldsymbol{S\theta} = \lambda\boldsymbol{F\theta}\,, \tag{4.19}$$

$$\boldsymbol{\theta}^\top \boldsymbol{F}\,\boldsymbol{\theta} = 1\,. \tag{4.20}$$

Fitzgibbon *et al* show that the solution to the above system can be found by taking a generalised eigendecomposition of $\boldsymbol{S}$ and $\boldsymbol{F}$. The estimate takes the form of the generalised eigenvector associated with the only positive generalised eigenvalue.

Algorithm 4.1 (FITZ) *Computes the Fitzgibbon 'direct ellipse' estimate* $\widehat{\boldsymbol{\theta}}_{\mathrm{FITZ}}$

1. *Compute the design matrix* $\boldsymbol{U}$ *as per Eq. (4.2.1) and then take* $\boldsymbol{S} = \boldsymbol{U}^\top\boldsymbol{U}$.
2. *Using* $\boldsymbol{F}$ *defined in Eq. (4.17), determine the generalised eigenvalues and associated eigenvectors* $(\lambda_i, \boldsymbol{\xi}_i)$ *from the generalised eigenproblem* $\boldsymbol{S\theta} = \lambda\boldsymbol{F\theta}$.
3. *Choose the estimate* $\widehat{\boldsymbol{\theta}}_{\mathrm{FITZ}} = \boldsymbol{\xi}_i$ *as the generalised eigenvector corresponding to the only positive generalised eigenvalue* λ_i.

4.5.2 Numerical considerations

One remaining problem of the above method is that while extremely noisy data sets yield good results, perfect (or near perfect) data sets fail to yield a sensible result. When there is low noise, $|\boldsymbol{\theta}^\top\boldsymbol{u}(\boldsymbol{x}_i)|$ will be small for each $\boldsymbol{x}_i$, and so will $\boldsymbol{\theta}^\top\boldsymbol{S}\,\boldsymbol{\theta}$. Thus for low noise, $\boldsymbol{S}$ will be close to singular and this can cause numerical instability for the generalised eigendecomposition. Halíř and Flusser [21] proposed a partitioning method to mitigate these numerical problems. The idea is to split $\boldsymbol{U}$ up and solve a modified, non-degenerate, form of Eq. (4.19) with $\boldsymbol{S}$ replaced by a matrix of reduced dimension.

A generalised eigenproblem involving potentially singular matrices is replaced with a lower dimensionality problem (with full rank matrices), and the 'extra bits' are able to be found in closed form as a post process.

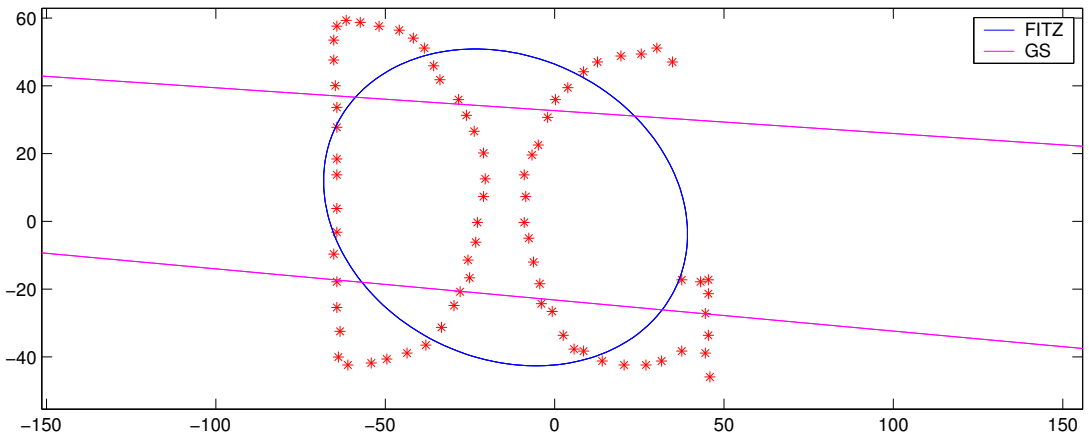


Figure 4.15: The Fitzgibbon method fits an ellipse to data, no matter how "un-elliptical" are the given data points

The design matrix is split in two matrices

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{U}_1 & \boldsymbol{U}_2 \end{bmatrix}$$

with

$$\boldsymbol{U}_1 = \begin{bmatrix} x_1^2 & x_1 y_1 & y_1^2 \\ & \vdots & \\ x_n^2 & x_n y_n & y_n^2 \end{bmatrix}, \qquad \boldsymbol{U}_2 = \begin{bmatrix} x_1 & y_1 & 1 \\ & \vdots & \\ x_n & y_n & 1 \end{bmatrix}. \tag{4.21}$$

The scatter matrix is segmented as

$$\boldsymbol{S} = \boldsymbol{D}^\top \boldsymbol{D} = \begin{bmatrix} \boldsymbol{U}_1^\top \boldsymbol{U}_1 & \boldsymbol{U}_1^\top \boldsymbol{U}_2 \\ \boldsymbol{U}_2^\top \boldsymbol{U}_1 & \boldsymbol{U}_2^\top \boldsymbol{U}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{S}_1 & \boldsymbol{S}_2 \\ \boldsymbol{S}_2^\top & \boldsymbol{S}_3 \end{bmatrix}. \tag{4.22}$$

Using these segmented forms, the eigenproblem of Eq. (4.19) is written as

$$\begin{bmatrix} \boldsymbol{S}_1 & \boldsymbol{S}_2 \\ \boldsymbol{S}_2^\top & \boldsymbol{S}_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix} = \lambda \begin{bmatrix} \boldsymbol{F}' & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix} \tag{4.23}$$

with the parameter vector $\boldsymbol{\theta} = [\boldsymbol{b}_1^\top, \boldsymbol{b}_2^\top]^\top$. An equivalent system is

$$\boldsymbol{S}_1 \boldsymbol{b}_1 + \boldsymbol{S}_2 \boldsymbol{b}_2 = \lambda \boldsymbol{F}' \boldsymbol{b}_1 \,, \tag{4.24}$$

$$\boldsymbol{S}_2^\top \boldsymbol{b}_1 + \boldsymbol{S}_3 \boldsymbol{b}_2 = \boldsymbol{0}. \tag{4.25}$$

The matrix $\boldsymbol{S}_3$ is exactly the scatter matrix for fitting a straight line through the data points. It will only be singular when all of the points lie on a line, which is a degenerate

case for ellipse fitting. In the non degenerate case, $\boldsymbol{S}_3$ will have full rank and may be safely inverted. Solving for $\boldsymbol{b}_2$ yields

$$\boldsymbol{b}_2 = -\boldsymbol{S}_3^{-1}\boldsymbol{S}_2^\top \boldsymbol{b}_1. \tag{4.26}$$

Substituting this back into Eq. (4.24) yields

$$\boldsymbol{S}_1\boldsymbol{b}_1 + \boldsymbol{S}_2\left(-\boldsymbol{S}_3^{-1}\boldsymbol{S}_2^\top \boldsymbol{b}_1\right) = \lambda \boldsymbol{F}'\boldsymbol{b}_1,$$

and, upon simplification,

$$\left(\boldsymbol{S}_1 - \boldsymbol{S}_2\boldsymbol{S}_3^{-1}\boldsymbol{S}_2^\top\right)\boldsymbol{b}_1 = \lambda \boldsymbol{F}'\boldsymbol{b}_1.$$

This is a 'reduced' generalised eigenproblem $\boldsymbol{M}\boldsymbol{b}_1 = \lambda\boldsymbol{F}'\boldsymbol{b}_1$ with

$$\boldsymbol{M} = \boldsymbol{S}_1 - \boldsymbol{S}_2\boldsymbol{S}_3^{-1}\boldsymbol{S}_2^\top, \tag{4.27}$$

that may be solved analogously to the full sized system yielding $\widehat{\boldsymbol{b}}_1$.

With an estimated value in $\widehat{\boldsymbol{b}}_1$, the remaining half of the parameters are found directly via

$$\widehat{\boldsymbol{b}}_2 = -\boldsymbol{S}_3^{-1}\boldsymbol{S}_2^\top \widehat{\boldsymbol{b}}_1,$$

resulting in the solution $\widehat{\boldsymbol{\theta}} = [\widehat{\boldsymbol{b}}_1^\top \ \widehat{\boldsymbol{b}}_2^\top]^\top$.

Halíř's partitioning only mitigates the specific problem of numerical stability in the case of low (or no) noise, and does nothing to address the inherent bias. In practice, this numerical conditioning is improved, but not completely avoided, by pre-normalising the input data. The implementation publicly released by Fitzgibbon[1] included data normalisation (unlike the original published version).

Halíř later expanded on this original partitioning work and presented a method based on M-estimators to attempt a reduction in the bias [20]. This approach, in manipulating and choosing a subset of the data, also fails to address the underlying statistical bias involved in using an 'algebraic' least squares cost function.

4.6 Applying an ellipse-specific correction

The Fitzgibbon method sets out to estimate an ellipse by minimising J_{LS} subject to a quadratic ellipse constraint $\boldsymbol{\theta}^\top\boldsymbol{F}\boldsymbol{\theta} = 1$. It is possible to consider minimising J_{AML},

[1] Available from: `http://www.robots.ox.ac.uk/~awf/ellipse/`

Algorithm 4.2 (HAL) *Computes the Fitzgibbon 'direct ellipse' avoiding a numerical instability*

1. *Compute $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$ as given in Eq. (4.21) and then take $\boldsymbol{S}_1, \boldsymbol{S}_2, \boldsymbol{S}_3$ as per Eq. (4.22).*

2. *With $\boldsymbol{F}'$ as defined in Eq. (4.17) determine the three eigenvalue/eigenvector pairs $(\lambda_i, \boldsymbol{\xi}_i)$ of the equation*

$$\left[\boldsymbol{F}'^{-1}(\boldsymbol{S}_1 - \boldsymbol{S}_2\boldsymbol{S}_3^{-1}\boldsymbol{S}_2^\top)\right]\boldsymbol{b}_1 = \lambda \boldsymbol{b}_1\,.$$

3. *Determine the constraint value*

$$\boldsymbol{\xi}_i^\top \boldsymbol{F}' \boldsymbol{\xi}_i$$

   for each $i = 1, 2, 3$, and set $\widehat{\boldsymbol{b}}_1 = \boldsymbol{\xi}_i$, the only $\boldsymbol{\xi}_i$ for which the above constraint value is positive.

4. *Compute $\widehat{\boldsymbol{b}}_2 = -\boldsymbol{S}_3^{-1}\boldsymbol{S}_2^\top \widehat{\boldsymbol{b}}_1$ and the estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{HAL}} = [\widehat{\boldsymbol{b}}_1^\top, \widehat{\boldsymbol{b}}_2^\top]^\top$.*

rather than J_{LS}, under the same (ellipse-specific) constraint. The key to progressing is the factorisation (described in Sec. 3.2.3)

$$\left[\partial_{\boldsymbol{\theta}} J_{\mathrm{AML}}(\boldsymbol{\theta})\right]^{\top} = 2\boldsymbol{X}_{\boldsymbol{\theta}}\,\boldsymbol{\theta}. \tag{4.28}$$

The original Lagrange system of Eq. (4.18) is updated to

$$\left[\partial_{\boldsymbol{\theta}} J_{\mathrm{AML}}(\boldsymbol{\theta})\right]^{\top} - \lambda\left[\partial_{\boldsymbol{\theta}}(\boldsymbol{\theta}^{\top}\boldsymbol{F}\,\boldsymbol{\theta} - 1)\right]^{\top} = 0, \tag{4.29}$$

whereby the factor matrix is substituted for the gradient of J_{AML} giving a new system

$$\boldsymbol{X}_{\boldsymbol{\theta}}\,\boldsymbol{\theta} = \lambda\boldsymbol{F}\boldsymbol{\theta}\,, \tag{4.30}$$

$$\boldsymbol{\theta}^{\top}\boldsymbol{F}\,\boldsymbol{\theta} = 1. \tag{4.31}$$

Given a specific matrix $\boldsymbol{X}_{\boldsymbol{\theta}}$, an ellipse estimate may be sought as for the original Fitzgibbon (or Halíř) method.

Solving the above system is not as straightforward as the original case, for two reasons. Firstly, while $\boldsymbol{S}$ can purely be determined from the input data, the matrix $\boldsymbol{X}_{\boldsymbol{\theta}}$ is a function of $\boldsymbol{\theta}$. Secondly, unlike $\boldsymbol{S}$, $\boldsymbol{X}_{\boldsymbol{\theta}}$ is not positive semi-definite. The system may be solved by considering the Halíř decomposition (described in the previous section).

Because an initial solution is already required, this method can be thought of as applying a 'correction' to an existing estimate. The technique is summarised in Alg. 4.3. Examples of how it may be used to correct a hyperbolic solution to an ellipse are shown in Fig. 4.16. In this figure, data set B, with $\sigma = 10.0$, was used to generate noisy data. Cases where the FNS estimate was a hyperbola were then selected. By applying Alg. 4.3, these estimates were corrected to produce ellipses. Generally, the corrected ellipses were closer to the ideal underlying ellipse than the Fitzgibbon estimates. This algorithm was successful mainly because the noisy points conformed to the adopted model. When the noise level is too high, or the points are simply too un-elliptical, then the COR method breaks down and no estimate can be produced.

4.7 Seeding the gold standard method

The GS method has one significant limitation in that it requires an initial value for a seed. As seen in unconstrained experiments, there is advantage in using a method based on AML, for example TAU, to seed LM minimisation of the J_{AML} cost function. The GS method requires an ellipse to be used as a seed. However, in general the TAU

Algorithm 4.3 (COR) *Computes a 'direct-ML' ellipse estimate* $\widehat{\boldsymbol{\theta}}_{\mathrm{COR}}$

1. *Compute the FNS estimate* $\widehat{\boldsymbol{\theta}}_0 = \widehat{\boldsymbol{\theta}}_{\mathrm{FNS}}$
2. *If* $(\widehat{\boldsymbol{\theta}}_0^{\top} \boldsymbol{F}\, \widehat{\boldsymbol{\theta}}_0) > 0$ *(the FNS estimate is already an ellipse), then set* $\widehat{\boldsymbol{\theta}}_{\mathrm{COR}} = \widehat{\boldsymbol{\theta}}_0$ *and finish.*
3. *Otherwise, compute* $\boldsymbol{X}(\boldsymbol{\theta})$ *from Eq. (3.28).*
4. *Follow Alg. 4.2 (HAL) with* $\boldsymbol{S}$ *replaced by* $\boldsymbol{X}_{\boldsymbol{\theta}}$.

estimate will not be an ellipse, and so in general it may not be able to be used as a seed. In these cases, an alternative is to use the FITZ estimate. However, the significant bias in the FITZ estimate will at best cause the LM method to take a very large number of iterations, but often will prevent it from converging to the correct solution. When the COR estimate can be found, it can be used to seed the GS method. As it exhibits much less bias, it enables the GS method to converge with significantly less iterations.

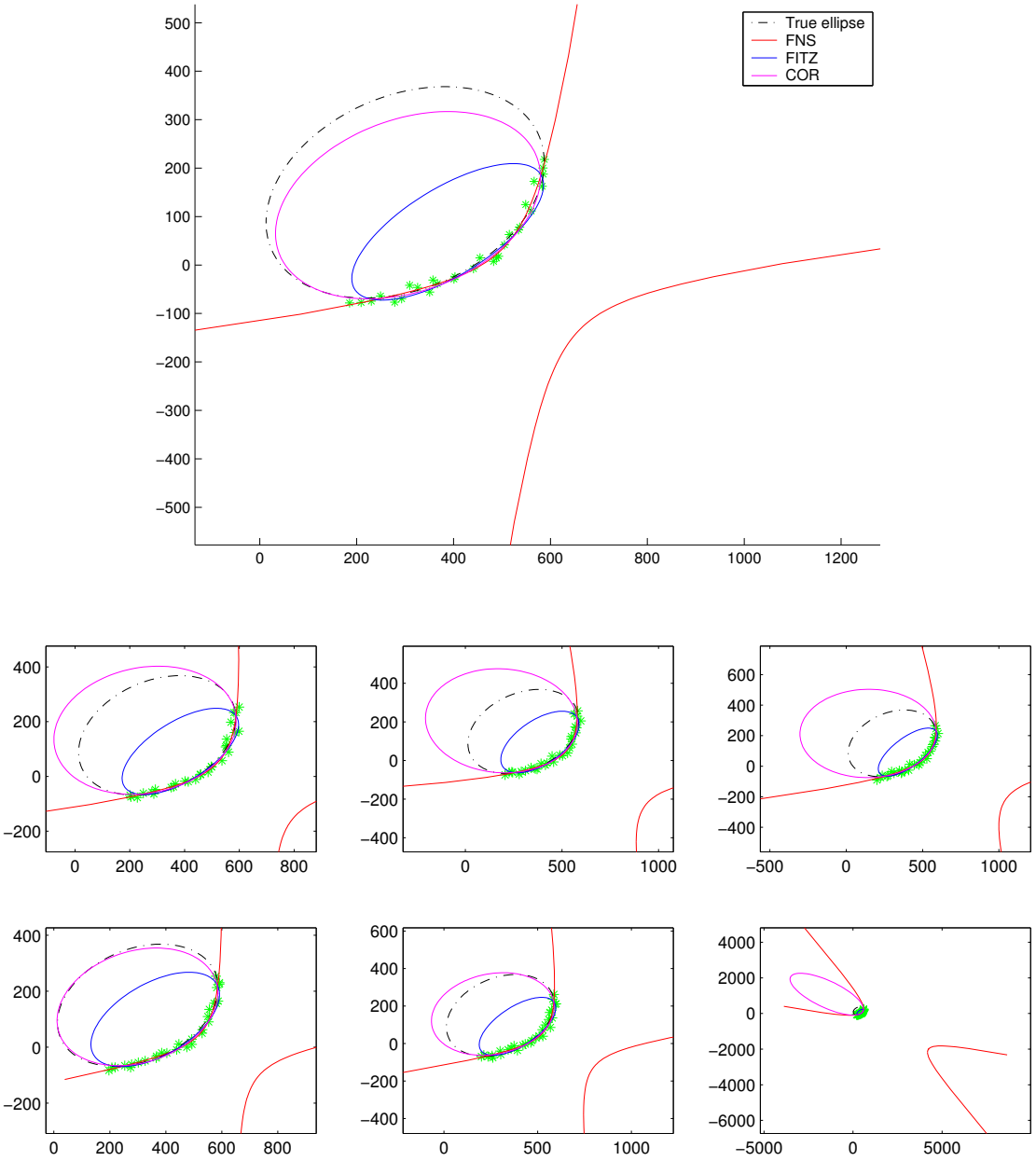


Figure 4.16: Examples of cases where the FNS method produces a hyperbola as its estimate. The COR estimate is shown, along with the underlying ideal ellipse along which the idea data points were generated

Chapter 5

APPLICATION II: FUNDAMENTAL MATRIX ESTIMATION

Stereo vision is a special case of multiple view geometry, where two images of a particular scene are available. Adding views can facilitate recovery of additional information, usually depth, from images. This chapter is concerned with estimation of the so-called fundamental matrix which is related to stereo vision. The fundamental matrix encodes key information about the internal configuration of cameras, and their external geometric relationship to each other. It has uses in aiding (dense) feature matching [70], motion segmentation [66], new view synthesis [7], video synchronisation [50], and initialisation of structure recovery algorithms [64], among many others.

General estimation methods described in previous chapters are placed in the context of existing well known methods of fundamental matrix estimation. The FNS and CFNS methods are applied, with CFNS taking into consideration a special constraint required of fundamental matrix estimates.

5.1 Epipolar geometry and the fundamental matrix

Under a pinhole camera model [12, §3], a homogeneous 3-D point $\boldsymbol{M} = (X, Y, Z, W)$ is projected onto an image plane to give an image point $\boldsymbol{m} = [m_x, m_y, 1]$. For a second image, the same 3-D point will project to $\boldsymbol{m}' = [m'_x, m'_y, 1]$. The pair of image points $\boldsymbol{m}$ and $\boldsymbol{m}'$ are said to be *corresponding points* as they are both projections of the same 3-D point $\boldsymbol{M}$.

Each point $\boldsymbol{M}_i$ in a set of n 3-D points will project to $\boldsymbol{m}_i$ and $\boldsymbol{m}'_i$ in the left and right images respectively. The *epipolar constraint* involves a 3×3 matrix $\boldsymbol{F}$ so that

$$\boldsymbol{m}_i'^{\top} \boldsymbol{F}\, \boldsymbol{m}_i = 0 \tag{5.1}$$

holds for all $i = 1 \dots n$. The matrix $\boldsymbol{F} = [f_{ij}]$ is termed the *fundamental matrix*, and is rank-two, and defined only up to a scale factor. When the optical centres of the left and right cameras coincide, there is only pure rotation and no translation between the

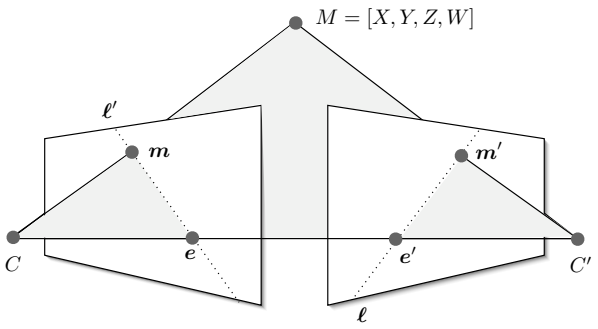


Figure 5.1: Epipolar geometry

two images. This is a degeneracy as regards the epipolar geometry and a fundamental matrix ceases to make sense. In this situation, a *homography matrix* can be used to relate the two images instead [13, §1.9] [28]. Often, in the case when little is known about underlying geometry of cameras, a first step is to analyse the available data to determine whether a fundamental matrix or homography is more appropriate to estimate [30, 61]. For the purposes of this chapter, however, this problem is ignored and only fundamental matrix estimation is considered.

The epipolar equation can be written in the (general) form of Eq. (2.2), $\boldsymbol{\theta}^\top \boldsymbol{u}(\boldsymbol{x}) = 0$, by letting $\boldsymbol{x} = [m_x, m_y, m'_x, m'_y]^\top$, $\boldsymbol{\theta} = [f_{11}, f_{12}, \ldots, f_{33}]^\top$, and taking

$$\boldsymbol{u}(\boldsymbol{x}) = [m_x m'_x, m_y m'_x, m'_x, m_x m'_y, m_y m'_y, m'_y, m_x, m_y, 1]^\top. \tag{5.2}$$

The ancillary constraint, $|\boldsymbol{F}| = 0$, is expressed explicitly as

$$\psi(\boldsymbol{\theta}) = \theta_1(\theta_5\theta_9 - \theta_6\theta_8) + \theta_2(\theta_3\theta_7 - \theta_1\theta_9) + \theta_3(\theta_4\theta_8 - \theta_3\theta_5). \tag{5.3}$$

This constraint function is homogeneous of degree 3: $\psi(\lambda\boldsymbol{\theta}) = \lambda^3\psi(\boldsymbol{\theta})$. These forms can be conveniently expressed using a *vectorisation* operator defined as follows. If $\boldsymbol{A} = [\boldsymbol{a}_1 \ldots \boldsymbol{a}_m]$ is an $n \times m$ matrix with $\boldsymbol{a}_i$ as its ith column vector (of length n), then $\mathrm{vec}\,(\boldsymbol{A}) = [\boldsymbol{a}_1^\top \ldots \boldsymbol{a}_m^\top]^\top$ is defined as the column vector of length mn composed as the concatenation of all $\boldsymbol{A}$'s column vectors. Using this operator, the definition of the parameter vector and carrier vector are now straightforward:

$$\boldsymbol{\theta} = \mathrm{vec}\left(\boldsymbol{F}^\top\right), \qquad \text{and} \qquad \boldsymbol{u}(\boldsymbol{x}) = \mathrm{vec}\left(\boldsymbol{m}\boldsymbol{m}'^\top\right).$$

As shown in Fig. 5.1, each point in one image has a corresponding *epipolar line* in the other. For a given left-image point $\boldsymbol{m}_i$, its associated epipolar line is $\boldsymbol{\ell}_i = \boldsymbol{F}^\top \boldsymbol{m}'_i$.

A right-image point $\boldsymbol{m}'_i$ has an associated epipolar line $\boldsymbol{\ell}'_i = \boldsymbol{F}\boldsymbol{m}_i$. The *epipoles* are the points of intersection with the image planes and the line joining the optical centres of both cameras. They are found as the left and right null spaces of the fundamental matrix.

Given a pair of stereo images, estimating the fundamental matrix requires first that a set of data points be extracted from the images. This is in itself a very challenging task as points from the left and right images which correspond to the same 3-D point must be unambiguously determined. It is known as the *correspondence problem* and the pairs of corresponding points are sometimes called *matching points*. For the purpose of this chapter, however, the correspondence problem will not be addressed. Rather it will be assumed that reliable data is available, and the second step, estimating the epipolar geometry given the matching points, will be considered.

Many methods have been put forward for estimating the fundamental matrix. The simplest techniques are direct, finding an estimate without recourse to an iteration procedure. More sophisticated methods use iteration as a means to obtain a more accurate solution. Within the category of iterative methods, there is variation as to the level of computation involved. Several reviews of existing methods are available [40] [16] . Additionally, when estimating the fundamental matrix, often there can be mis-matched points present in the data. The assumption of Gaussianity of noise is then violated, as mismatched points will be perturbed from ideal points by a potentially very large number of pixels. To deal with this situation, so-called robust methods have been developed [57, 63, 70]. Such methods are not considered in this chapter, rather estimation is considered in the absence of outliers.

5.2 Non-iterative methods

Direct methods for computing the fundamental matrix can be useful, for example, to obtain a starting value for an iterative method. Several of the following methods do not take into account the ancillary constraint that any estimated fundamental matrix must have rank-two. This deficiency is addressed in a subsequent section.

5.2.1 The eight-point algorithm

The eight-point algorithm for estimating the *essential matrix*, a matrix closely related to the fundamental matrix, was first proposed by Longuet-Higgins [39]. This method

is essentially the application of linear total least squares (Sect. 2.4.1) to estimating parameters arising from an equation very similar to Eq. (5.1). The eight-point algorithm is so named because there must be at least eight elements of data (pairs of corresponding points) for a (unique) estimate to be found (up to a arbitrary scale factor). Of course, if there are more than eight points, then the estimate minimising the total least squares cost function is obtained. The design matrix is computed using $\boldsymbol{u}(\boldsymbol{x}_i)$, and the eight-point (TLS) estimate is found by taking a SVD of this design matrix.

In the seminal paper *"In defence of the eight-point algorithm"* [24], Hartley shows that the performance of the eight-point algorithm may be significantly improved if the input data points are *normalised* by scaling prior to taking the SVD of the design matrix. This method has been widely adopted as a reasonable accurate and very fast method for computing the fundamental matrix. It is not immediately clear however why the somewhat ad-hoc step of prior normalisation of the data improves results. The Hartley normalised method is now summarised.

The carrier vector $\boldsymbol{u}(\boldsymbol{x})$ has some elements which are quadratic in the elements of $\boldsymbol{x}$, some linear, and its last element equal to 1. As the values comprising $\boldsymbol{x}$ are image coordinates, usually measured in pixels, their values can easily be 1000 or greater. Therefore, some elements of the design matrix can be of order 10^6 while others are of order one. Hence, the design matrix has poor numerical conditioning. The modified condition number of the design matrix in the form of the ratio of the largest to second smallest eigenvalues will tend to be large, a situation which leads to the smallest eigenvector being less stable. Any small perturbation in the values of the design matrix can lead to significant changes of this eigenvector, causing instability in the estimate which is taken as an eigenvector.

Hartley's normalisation method seeks to scale the data so that the image points are centred at the origin, and the mean distance of points from the origin is $\sqrt{2}$. The aim is to "even up" the scales of the elements of the carriers (and the design matrix which is composed of the carriers). This specification forms a "canonical basis" whereby the numerical conditioning can be much improved, and affords greater stability to the eigenvectors of the design matrix. *Transformation matrices* $\boldsymbol{T}$ and $\boldsymbol{T}'$ are sought to transform the image points within the left and right images respectively. The transformed points are taken as $\tilde{\boldsymbol{m}}_i = \boldsymbol{T}\boldsymbol{m}_i$ and $\tilde{\boldsymbol{m}}'_i = \boldsymbol{T}'\boldsymbol{m}'_i$. These transformed points have elements $\tilde{\boldsymbol{m}}_i = [\tilde{m}_{x,i}, \tilde{m}_{y,i}, 1]$ and $\tilde{\boldsymbol{m}}'_i = [\tilde{m}'_{x,i}, \tilde{m}'_{y,i}, 1]$, with the transformed data point taken as $\tilde{\boldsymbol{x}}_i = [\tilde{m}_{x,i}, \tilde{m}_{y,i}, \tilde{m}'_{x,i}, \tilde{m}'_{y,i}]$. The transformed data points are used to form a design ma-

trix,

$$\tilde{\boldsymbol{U}} = \begin{bmatrix} \boldsymbol{u}(\tilde{\boldsymbol{x}}_1)^\top \\ \vdots \\ \boldsymbol{u}(\tilde{\boldsymbol{x}}_n)^\top \end{bmatrix}, \tag{5.4}$$

and by performing an SVD of $\tilde{\boldsymbol{U}}$, a "transformed" estimate $\widehat{\tilde{\boldsymbol{F}}}_{\mathrm{TLS}}$ can be found. By substituting the transformed points into Eq. (5.1), the equality $\boldsymbol{m}'^\top \boldsymbol{F}\, \boldsymbol{m} = \tilde{\boldsymbol{m}}'^\top \tilde{\boldsymbol{F}}\, \tilde{\boldsymbol{m}}$ will hold if $\tilde{\boldsymbol{F}} = \boldsymbol{T}'^{-\top} \boldsymbol{F}\, \boldsymbol{T}^{-1}$. Therefore, the normalised Hartley estimate $\widehat{\boldsymbol{F}}_{\mathrm{HRT}}$ is taken as

$$\widehat{\boldsymbol{F}}_{\mathrm{HRT}} = \boldsymbol{T}'^\top \widehat{\tilde{\boldsymbol{F}}}_{\mathrm{TLS}}\, \boldsymbol{T}. \tag{5.5}$$

The transformation specified involves a translation of each point and a scaling. The centroids of the left and right sets of image points are involved in the scaling, and are defined by

$$\check{\boldsymbol{m}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{m}_i \quad \text{and} \quad \check{\boldsymbol{m}}' = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{m}'_i. \tag{5.6}$$

The image points can be centred around the origin by subtracting the appropriate centroid from each data point. The centred points are then scaled. The scaling factors are defined as follows,

$$s = \left(\frac{1}{2n} \sum_{i=1}^{n} \|\boldsymbol{m}_i - \check{\boldsymbol{m}}\|^2 \right)^{1/2} \quad \text{and} \quad s' = \left(\frac{1}{2n} \sum_{i=1}^{n} \|\boldsymbol{m}'_i - \check{\boldsymbol{m}}'\|^2 \right)^{1/2}. \tag{5.7}$$

The appropriate transformation matrices are defined in terms of these values, as follows:

$$\boldsymbol{T} = \begin{bmatrix} s^{-1} & 0 & -s^{-1}\check{m}_x \\ 0 & s^{-1} & -s^{-1}\check{m}_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{5.8}$$

and

$$\boldsymbol{T}' = \begin{bmatrix} s'^{-1} & 0 & -s'^{-1}\check{m}'_x \\ 0 & s'^{-1} & -s'^{-1}\check{m}'_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{5.9}$$

The normalised algorithm, whereby data points are transformed, an estimate produced, then back points are back-transformed, can be applied with not just TLS, but any estimator. This process will provide only numerical stability to estimators which are invariant to transformations of data points. However, as shown in Sect. 2.4.4 and Fig. 4.3, the TLS method is not invariant to uniform scaling of the data. Therefore,

applying the Hartley algorithm not only changes numerical conditioning but also the underlying associated cost function. Even on a computer capable of infinite precision calculations, the TLS estimate computed using un-normalised data would not produce an estimate the same as that of the Hartley normalisation method.

In upcoming experiments, the label for Hartley's normalisation method is **HRT**.

5.2.2 Taubin's method

The method of Taubin (Sect. 3.2.1) may be directly applied to estimating the fundamental matrix. Again, the specific fundamental matrix carrier form is directly used. As for conic fitting experiments, the label for Taubin's method is again **TAU**. The appropriate gradient matrix is

$$[\partial_{\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})] = \begin{bmatrix} m'_x & 0 & m_x & 0 \\ 0 & m'_x & m_y & 0 \\ 0 & 0 & 1 & 0 \\ m'_y & 0 & 0 & m_x \\ 0 & m'_y & 0 & m_y \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{5.10}$$

5.2.3 Invariant Fitting

For the conic fitting problem, there are several normalisations proposed as an alternative to TLS's $\boldsymbol{\theta}^\top\boldsymbol{\theta} = 1$. Bookstein in particular proposes a particular normalisation which is invariant to Euclidean transformations of the data points. Torr and Fitzgibbon [63] show that it is possible to apply this idea to estimating the fundamental matrix. It turns out that there is a natural normalisation that can be applied, leading to an non-iterative estimator which, while still algebraic in nature, leads to estimates which are invariant to an Euclidean transformation of the data points.

5.3 Iterative methods

Iterative methods for estimating the fundamental matrix aim for greater accuracy than those methods which compute their estimate directly. When describing an iterative

method, three factors must be considered. Firstly, what is the cost function? Secondly, how is the cost function to be minimised? And, thirdly, how will the ancillary, rank-two, constraint be enforced? The third point will be discussed later. Next, the first two points are addressed.

5.3.1 Cost functions

The (full) maximum likelihood cost function is regarded as an optimal cost function (in some sense). In the fundamental matrix case, it has a direct geometric interpretation of minimising the sum of squared distances between each data point and the closest consistent point:

$$J_{\mathrm{ML}}(\boldsymbol{F}) = \sum_{i=1}^{n} \mathrm{d}(\boldsymbol{m}_i, \overline{\boldsymbol{m}}_i)^2 + \mathrm{d}(\boldsymbol{m}'_i, \overline{\boldsymbol{m}}'_i)^2 \tag{5.11}$$

The pair of consistent points $(\overline{\boldsymbol{m}}_i, \overline{\boldsymbol{m}}'_i)$ is determined as a function of $\boldsymbol{F}$, and so minimising this cost function requires minimising over a significant number of parameters. Shortly, the gold standard algorithm is described which can minimise this cost function.

A 'reduced' cost function, which depends on the elements of the fundamental matrix, can be used instead. Such a cost function is the sum of distances from data points to epipolar lines. If covariance information is available then this cost function is taken as the sum of Mahalanobis from every data point to the epipolar line derived from the candidate fundamental matrix and its corresponding point in the other image. If covariance information is unavailable, then the simple Euclidean distance can be taken. This cost function can be explicitly written in terms of the data and a candidate fundamental matrix as

$$\begin{aligned} J_{\mathrm{EPL}}(\boldsymbol{F}) &= \sum_{i=1}^{n} \left(\mathrm{d}(\boldsymbol{m}_i, \boldsymbol{\ell}_i)^2 + \mathrm{d}(\boldsymbol{m}'_i, \boldsymbol{\ell}'_i)^2\right) \\ &= \sum_{i=1}^{n} \left(\frac{1}{\boldsymbol{m}_i^\top \boldsymbol{F}^\top \boldsymbol{\Lambda}_{\boldsymbol{m}'_i} \boldsymbol{F} \boldsymbol{m}_i} + \frac{1}{\boldsymbol{m}_i'^\top \boldsymbol{F}^\top \boldsymbol{\Lambda}_{\boldsymbol{m}_i} \boldsymbol{F} \boldsymbol{m}'_i} \right) \left(\boldsymbol{m}_i'^\top \boldsymbol{F}\, \boldsymbol{m}_i\right)^2. \end{aligned} \tag{5.12}$$

(If covariance matrices are unavailable, then the default matrices $\boldsymbol{\Lambda} = \mathrm{diag}\,(1, 1, 0)$ can be taken.)

Another reduced cost function is J_{AML}, the approximated ML cost function. It can also be expressed directly in terms of a fundamental matrix and matching points,

$$J_{\mathrm{AML}}(\boldsymbol{F}) = \sum_{i=1}^{n} \frac{1}{(\boldsymbol{m}_i^\top \boldsymbol{F}^\top \boldsymbol{\Lambda}_{\boldsymbol{m}'_i} \boldsymbol{F} \boldsymbol{m}_i) + (\boldsymbol{m}_i'^\top \boldsymbol{F}^\top \boldsymbol{\Lambda}_{\boldsymbol{m}_i} \boldsymbol{F} \boldsymbol{m}'_i)} \left(\boldsymbol{m}_i'^\top \boldsymbol{F}\, \boldsymbol{m}_i\right)^2. \tag{5.13}$$

This form of J_{AML} is very similar to that of J_{EPL} above, and so both cost functions exhibit similar characteristics. It can be expected that $\widehat{\boldsymbol{\theta}}_{\mathrm{AML}}$ will be very close to $\widehat{\boldsymbol{\theta}}_{\mathrm{EPL}}$ for example. The approximated maximum likelihood cost function is found to be (slightly) superior in comparative testing by Zhang [69] and others [13].

As seen in Chapter 3, the Levernberg Maquardt method, or FNS, can be employed to find a minimiser of J_{AML}. In upcoming experiments, these methods are designated **LM-A** and **FNS**, respectively. The cost function J_{EPL} cannot be minimised using FNS as it does not satisfy its assumptions. It can be minimised by using LM, but because of the similarity with the AML cost function, it was not included in experiments. Finding a minimiser of the full ML cost function requires that the geometry of the problem be employed. Such a method is called the "gold standard" and is summarised next.

5.3.2 Gold standard method

The Gold Standard (GS) method is so named because it is the method against which all others are typically compared. It directly minimises the maximum likelihood cost function, rather than an approximation or other algebraic entity. Therefore, under the given assumptions of Gaussian noise, it can be regarded as optimal. The GS method for estimating the fundamental matrix is an example of the general vision technique known as "bundle adjustment" which is applicable to an arbitrary number of images.

The GS method uses a purely geometric parameterisation: a stereo scene is represented by a 3×4 projection matrix for the left and right cameras, and a 3-D point for every pair of corresponding image points. The cost function is evaluated by projecting the 3-D point onto each image, and measuring the distance to the data point. The projection matrices and 3-D points are adjusted so as to minimise the sum of the reprojection disparities. Given an initial estimate of the fundamental matrix an initial values of the two projection matrices can be computed directly. Initial 3-D points can be computed via triangulation of the data points also using the initial fundamental matrix estimate [25].

The number of parameters involved in minimisation is ostensively $24 + 3n$, a significant enlargement of the nine required thus far. Without loss of generality, it is possible to implicitly define the first projection matrix, reducing the number of parameters required. And as projection matrices are defined only up to an unknown scale, this leaves only $11 + 3n$ required parameters. Nevertheless, the key inhibiting factor is that it is a function of n. The enlarged number of parameters causes an extra computational

burden to the LM algorithm, where the main problem is the requirement that the Jacobian matrix be inverted for each iteration. As such an inversion is typically O(n^3), the computational requirements increase rapidly for even modest values of n. It is possible to take advantage of the special form of the Jacobian to provide a much more efficient inversion [23,64], but nonetheless, the gold standard algorithm is still computationally intensive.

5.4 Experiments part I: Unconstrained estimation

Our first set of experiments for estimating the fundamental matrix disregard the ancillary rank-two constraint. The initial goal is to first ascertain that, for example, the FNS method is correctly minimising the AML cost function. Later, in the second part of experiments, the ancillary constraint is adopted.

5.4.1 Experimental setup

The experiments in the section are performed with synthetically generated data. This data are created by devising a stereo configuration, then generating a left and right projection matrices corresponding to synthetic cameras in the configuration. Then, 50 random 3-D points are generated (in the field of view of both cameras) and projected on to the left and right images yielding "true" corresponding points.

Two slightly different stereo configurations were used, labelled 'A' and 'B'. Configuration B is adopted as a more challenging geometric configuration, and uses a smaller baseline with less rotation of the cameras. Both configurations use non-coplanar optical axes. For both configurations, slightly different internal camera parameters were used. The method for generating the data and configurations was adapted from that of Torr [62]. The data points were projected to images which were of size 512×512 pixels. The underlying 'true' data points, and the associated epipoles, for each configuration are shown in Fig. 5.2.

Synthetic data is generated by taking the underlying 'true' points and perturbing each image point by adding to it random Gaussian noise. This noise is controlled by a single scalar, σ, the standard deviation of the noise added. With noisy data generated in this way, different estimation methods can be used to estimate an estimate of the fundamental matrix. To compare estimates, "geometric" error measures, such as J_{ML},

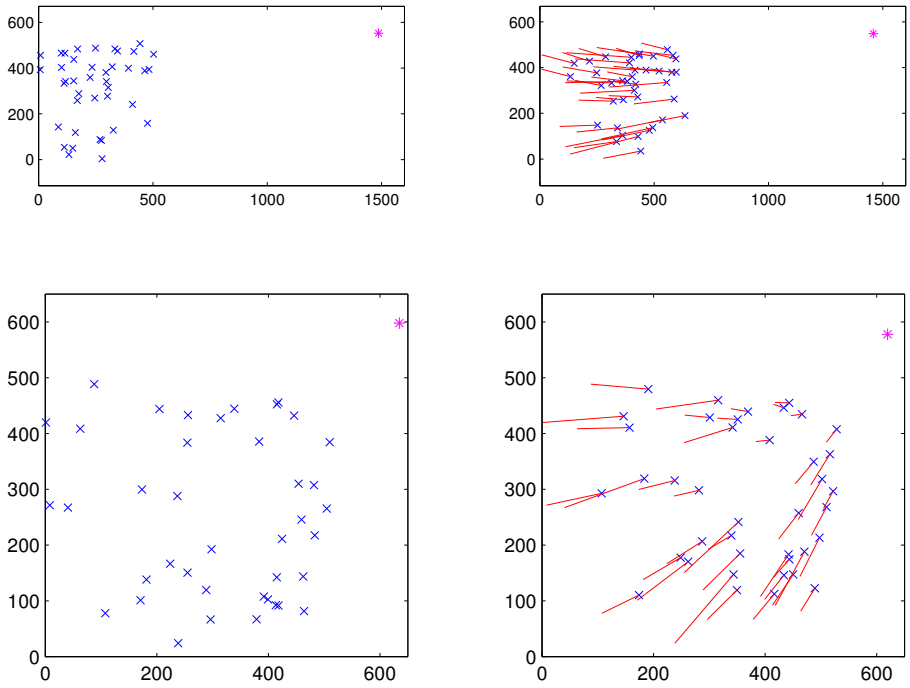


Figure 5.2: Synthetic stereo data generated from configurations A (top) and B (bottom). Left and right image data points, and their correspondences, are shown. Additionally, the position of epipoles is shown for reference (as an asterix)

cannot be applied as these estimates are not true fundamental matrices – they may not be rank-two. Instead, only general measures such as J_{AML} are employed.

5.4.2 Experiments

To estimate the (unconstrained) fundamental matrix, the FNS method was employed in a similar manner to that for estimating conic sections. It was seeded with the TAU estimate, $\widehat{\boldsymbol{F}}_{\mathrm{TAU}}$, and the stopping conditions were kept identical. The only difference in implementation was to replace the carrier function $\boldsymbol{u}$ relevant to conics (Eq. (4.2)) with that relevant to the fundamental matrix (Eq. (5.2), and to adopt the appropriate gradient as given by Eq. (5.10). A similar testing strategy was adopted too. For each of the configurations, the following procedure was performed:

- for a level of noise σ ranging from 1–10, repeat 200 times:
  - generate random data by perturbing the pure data by Gaussian noise (with standard deviation σ)

  - compute an estimated fundamental matrix using the TLS, HRT, FNS, and LM-A methods
  - record the number of iterations FNS took for convergence, and the value of J_{AML} of each estimate

Fig. 5.3 shows the average number of iterations required for convergence of the FNS method. It is seen that configuration B requires significantly more work than configuration A, particularly when the noise level is greater than seven pixels.

Histograms of cost function values are shown in Fig. 5.5. These results come from configuration B, and are at the highest noise level used of ten pixels. It is seen that the FNS and LM-A methods have estimates with the lowest values. This is to be expected as this is precisely the cost function they aim to minimise. As was found for experiments with estimating conic sections, the FNS method executes in a fraction of the time required for Levenberg-Maquardt.

The TLS results are not shown, because they have cost function values that are an order of magnitude higher. As expected, the TAU estimates are slightly inferior as TAU minimises an approximation to J_{AML} only. What is surprising from these results is how good the HRT estimates are as compared to the those of the iterative methods. By applying scaling to the data, the HRT method used total least squares to generate estimates which have improved from off the scale (TLS with no normalisation) to having cost function values almost as good as the iterative FNS and LM-A.

Adopting the method described in Sect. 4.3.3, slices of the cost function J_{AML} are shown in Fig. 5.4. As compared to the conic fitting problem, these slices show a more regular cost function shape. These slices are drawn using estimates produced using configuration B and $\sigma = 10$. They are centred on the FNS estimate, and the range is determined using the HRT method.

In summary, these results show that:

- FNS can be used to estimate the fundamental matrix, although for some configurations it make take more than ten iterations on average to converge.

- The FNS method and LM-A method do not always produce identical estimates. On average they are the same, but occasionally one method or the other may fail to just quite find the minimum.

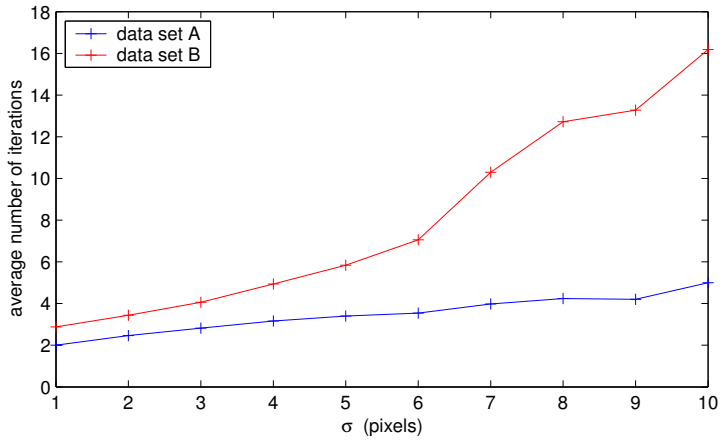


Figure 5.3: Average number of iterations required to compute FNS estimates

- The algebraic based HRT method produces estimates almost as good as the iterative and geometric based methods, even for more 'difficult' geometric configurations.

The reason for the excellent performance of the algebraic based HRT method is explored in the next section.

5.5 Rationalisation of Hartley normalisation

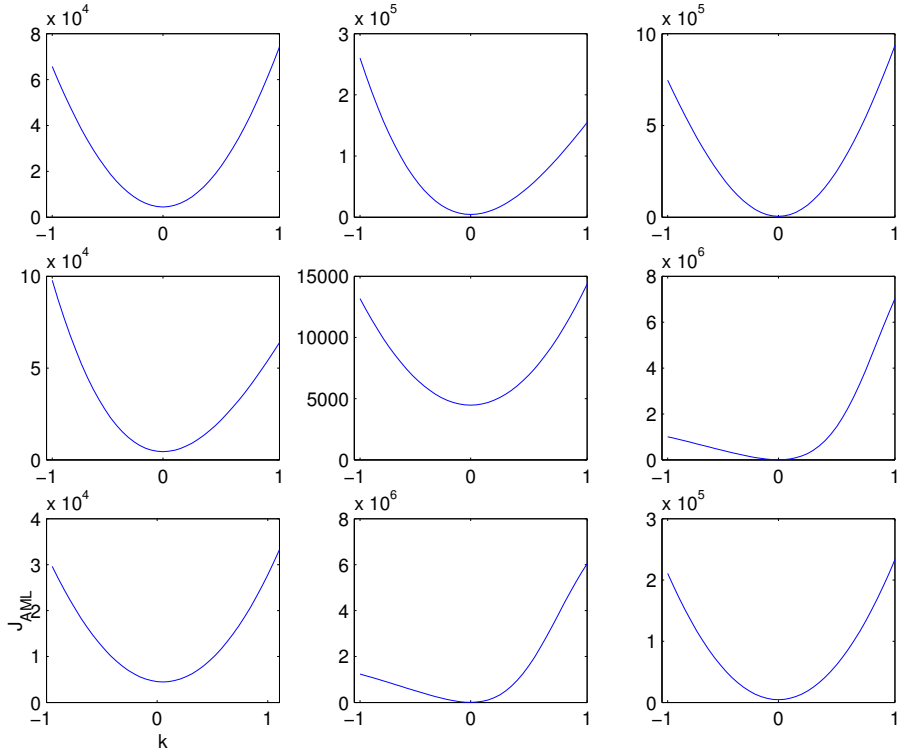
The Hartley normalisation method, when transforming the data, does more than providing additional numerical conditioning to the matrices involved in the estimation process. It changes the estimation process itself, so that the associated cost function is different for the normalised method than the ordinary un-normalised one. The changing of the underlying cost function turns out to be the reason for the significant improvement in performance of the normalised method.

The un-normalised TLS method minimises the J_{TLS} cost function given in Eq. (2.22). In terms of the fundamental matrix, $\boldsymbol{F}$, the function is

$$J_{\mathrm{TLS}}(\boldsymbol{F}) = \frac{\sum_{i=1}^{n} |\boldsymbol{m}_i'^{\top} \boldsymbol{F}\, \boldsymbol{m}_i|^2}{\|\boldsymbol{F}\|_{\mathrm{F}}}.$$

Following the Hartley algorithm, however, will result in an estimate which minimises the modified cost function

$$J_{\mathrm{HRT}}(\boldsymbol{F}) = \frac{\sum_{i=1}^{n} |\boldsymbol{m}_i'^{\top} \boldsymbol{F}\, \boldsymbol{m}_i|^2}{\|\boldsymbol{T}'^{\top} \boldsymbol{F}\, \boldsymbol{T}\|_{\mathrm{F}}}. \tag{5.14}$$



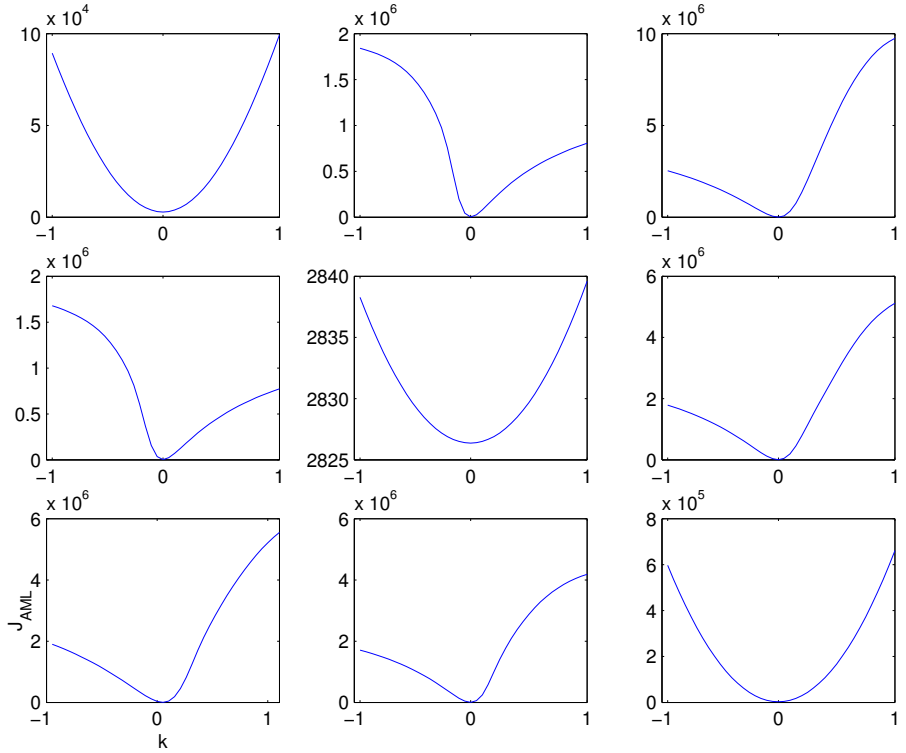


Figure 5.4: Slices of J_{AML} generated for $\sigma = 10.0$ for configuration A (above) and configuration B (below). The bounding estimate is HRT as the TLS estimates were too poor

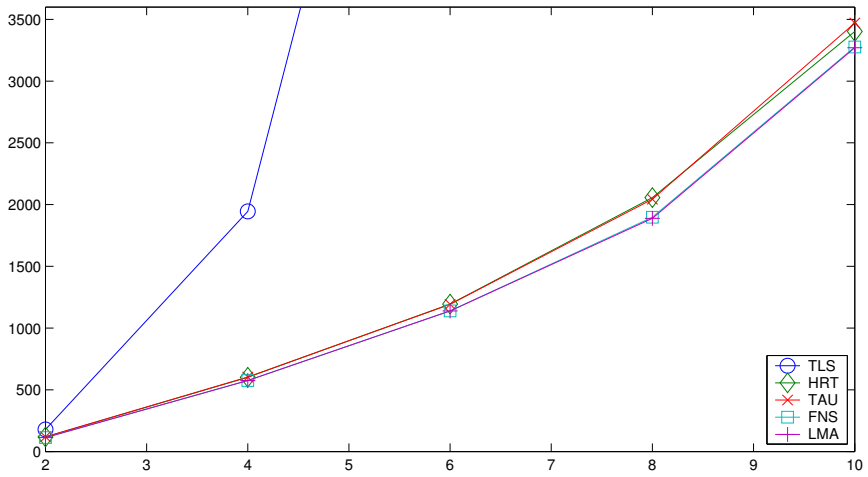


Figure 5.5: Values of J_{AML} for various estimates under increasing levels of noise

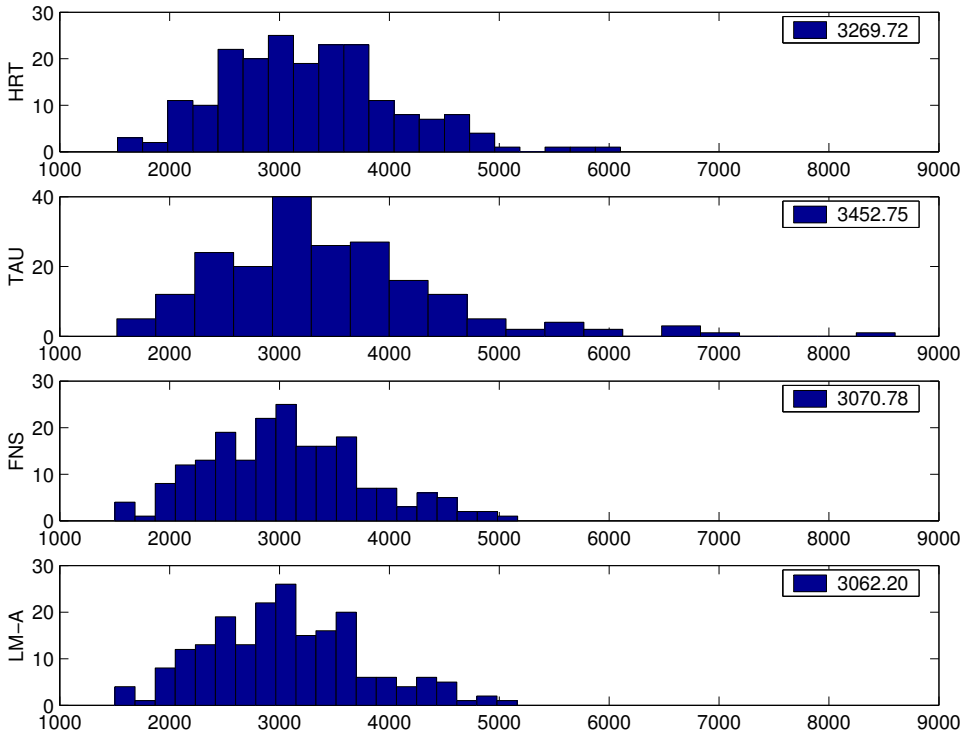


Figure 5.6: Histograms showing the J_{AML} values for 200 repeated estimates computed using different methods. Not shown are the values for TLS, as they are off the scale

Since the normalised and un-normalised fundamental matrices satisfy the identity $\boldsymbol{m}'^{\top}\boldsymbol{F}\,\boldsymbol{m} = \tilde{\boldsymbol{m}}'^{\top}\tilde{\boldsymbol{F}}\,\tilde{\boldsymbol{m}}$, it follows that

$$\begin{aligned} J_{\mathrm{HRT}}(\boldsymbol{F}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) &= \frac{\sum_{i=1}^{n} |\tilde{\boldsymbol{m}}_i'^{\top}\tilde{\boldsymbol{F}}\,\tilde{\boldsymbol{m}}_i|^2}{\|\tilde{\boldsymbol{F}}\|_{\mathrm{F}}} \\ &= J_{\mathrm{TLS}}(\tilde{\boldsymbol{F}}; \tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_n). \end{aligned} \tag{5.15}$$

This cost function can be written in terms of $\boldsymbol{\theta}$ by using the identity $\|\boldsymbol{T}'^{\top}\boldsymbol{F}\,\boldsymbol{T}\|_{\mathrm{F}} = \boldsymbol{\theta}^{\top}\boldsymbol{C}\,\boldsymbol{\theta}$, where $\boldsymbol{C}$ is given by

$$\boldsymbol{C} = \boldsymbol{T}'^{-1}\boldsymbol{T}'^{-\top} \otimes \boldsymbol{T}^{-1}\boldsymbol{T}^{-\top}. \tag{5.16}$$

Therefore, the normalised cost function is

$$J_{\mathrm{HRT}}(\boldsymbol{\theta}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \frac{\boldsymbol{\theta}^{\top}\boldsymbol{S}\,\boldsymbol{\theta}}{\boldsymbol{\theta}^{\top}\boldsymbol{C}\,\boldsymbol{\theta}}. \tag{5.17}$$

The minimiser may be determined directly from this cost function by solving the generalised eigenvalue problem

$$\boldsymbol{S}\boldsymbol{\theta} = \lambda\boldsymbol{C}\boldsymbol{\theta}\,. \tag{5.18}$$

The estimate $\widehat{\boldsymbol{\theta}}_{\mathrm{HRT}}$ is the generalised eigenvector corresponding to the smallest generalised eigenvalue. The TLS estimate is found via a SVD on the matrix $\boldsymbol{U}$ instead of an eigendecomposition of the numerically poorer conditioned $\boldsymbol{S}$. The same idea can be applied when solving the above system, using a generalised singular value decomposition (GSVD) of the pair of matrices $(\boldsymbol{U}, \boldsymbol{E})$, where $\boldsymbol{S} = \boldsymbol{U}^{\top}\boldsymbol{U}$ and $\boldsymbol{C} = \boldsymbol{E}^{\top}\boldsymbol{E}$. The matrix $\boldsymbol{U}$ is formed directly from the carriers, as per Eq. (2.24). The decomposition of $\boldsymbol{C}$ can be computed analytically based on Eq. (5.16). It is taken as $\boldsymbol{E} = \boldsymbol{T}'^{-1} \otimes \boldsymbol{T}^{-1}$.

By inspecting Eq. (5.17), it can be seen that the cost function J_{HRT} bears a resemblance to the cost function underlying the method of Taubin (see Sect. 3.2.1). Each is in the form of a Rayleigh quotient, having both numerator and denominator quadratic in $\boldsymbol{\theta}$. Indeed, the two cost functions differ only in the form of the matrix appearing in the denominator. When expanded, the matrix appearing in J_{HRT} has a similar form to that of the matrix involved in the Taubin cost function.

Earlier it was shown that the Taubin cost function can be regarded as an approximation to J_{AML}, where the denominator matrices $\boldsymbol{B}_i$ are replaced with an average matrix. In so far as the Hartley cost function resembles that of Taubin, it may be regarded as closer in nature to the geometrically derived cost function J_{AML} rather than to the purely algebraic cost function J_{TLS}.

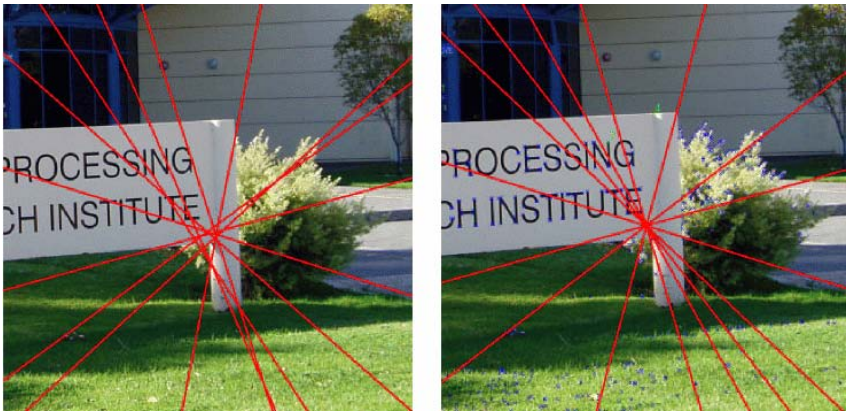


Figure 5.7: This figure shows two copies of a zoomed in portion of one image in a stereo pair. For the left image, an unconstrained fundamental matrix has been estimated from manually matched data points, and epipolar lines rendered. A fundamental matrix satisfying the rank-two constraint was used for the right image. The unconstrained estimate can still be used to compute epipolar lines, however they do not intersect at a common point (the epipole)

5.6 Incorporating the rank-two constraint

So far, the requirement that a fundamental matrix estimate be rank-two has been ignored. In this section, estimation of a rank-two matrix is considered. An unconstrained (full rank) estimate cannot be used as it is not consistent with the underlying epipolar geometry. For example, Fig. 5.7 (left) shows epipolar lines generated for a unconstrained estimate. As the matrix is not rank-two the epipolar lines do not intersect in a single point, the epipole. This is remedied in Fig. 5.7 (right) where epipolar lines corresponding to a rank-two fundamental matrix are shown. As discussed in Sect. 3.3, there are two courses of action available to produce estimates which satisfy an ancillary constraint. The estimation method itself can be modified to ensure the constraint is intrinsically satisfied, or an unconstrained estimate can be 'post-corrected'. Each of these ideas is applied specifically to the fundamental matrix problem.

5.6.1 Post correction

To use methods which do not consider the constraint, the ancillary constraint must be enforced as a post-process whereby the unconstrained estimate is corrected after it has been found. As with finding the estimates themselves, there are two ways of performing this correction: direct and iterative. The direct method seeks a rank-two estimate $\widehat{\boldsymbol{F}}_c$

which minimises the distance $\|\widehat{\boldsymbol{F}} - \widehat{\boldsymbol{F}}_c\|$. The minimiser can be easily found by firstly performing an SVD of $\widehat{\boldsymbol{F}} = \boldsymbol{J}\,\mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)\boldsymbol{K}^\top$, with $\sigma_1 \geq \sigma_2 \geq \sigma_3$. The corrected estimate is then taken as $\widehat{\boldsymbol{F}}_c = \boldsymbol{J}\,\mathrm{diag}(\sigma_1, \sigma_2, 0)\boldsymbol{K}^\top$. It can be proven that this SVD approach does in fact minimise the required Euclidean distance [70].

A more sophisticated correction method involves finding the minimiser of the Mahalanobis distance $(\widehat{\boldsymbol{\theta}} - \widehat{\boldsymbol{\theta}}_c)^\top \boldsymbol{\Lambda}_{\widehat{\boldsymbol{\theta}}}^{-1} (\widehat{\boldsymbol{\theta}} - \widehat{\boldsymbol{\theta}}_c)$. Unfortunately, no direct method is able to compute the minimiser of such a distance, so an iterative method must be used. Matei and Meer [42] phrase the correction problem as an analogy to the original estimation problem. They seek a minimiser of $(\widehat{\boldsymbol{\theta}} - \widehat{\boldsymbol{\theta}}_c)^\top \boldsymbol{\Lambda}_{\widehat{\boldsymbol{\theta}}}^{-1} (\widehat{\boldsymbol{\theta}} - \widehat{\boldsymbol{\theta}}_c)$ subject to $\boldsymbol{F}\boldsymbol{e} = 0$.

An alternative approach was originally proposed by Kanatani [29, Chap. 5]. An simple iterative algorithm is employed to find the corrected estimate minimising the above Mahalanobis distance. Starting with the unconstrained estimate as $\boldsymbol{\theta}_0$, each successive estimate is taken as

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \left(\left[\partial_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta}_k)\right] \boldsymbol{R}_{\boldsymbol{\theta}_k}^- \left[\partial_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta})\right]^\top \right)^{-1} \psi(\boldsymbol{\theta}_k) \boldsymbol{R}_{\boldsymbol{\theta}_k}^- \left[\partial_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta}_k)\right]^\top, \tag{5.19}$$

where $\boldsymbol{R}_{\boldsymbol{\theta}} = \boldsymbol{Q}_{\boldsymbol{\theta}} \boldsymbol{M}_{\boldsymbol{\theta}}^- \boldsymbol{Q}_{\boldsymbol{\theta}}$. The matrix $\boldsymbol{M}_{\boldsymbol{\theta}}$ is given in Eq. (3.22), and $\boldsymbol{Q}_{\boldsymbol{\theta}} = \boldsymbol{I} - \|\boldsymbol{\theta}\|^{-2} \boldsymbol{\theta}\boldsymbol{\theta}^\top$. The matrix $\boldsymbol{R}_{\boldsymbol{\theta}}$ can be computed once, from the unconstrained estimate, or re-computed at every iteration. Typically only a small number of iterations (sometimes as few as one or two) are necessary.

5.6.2 Parameterisation

The fundamental matrix has nine elements but only seven degrees of freedom. This is shown via a simple counting argument; there are nine elements of the matrix, subtracting one for the scale indeterminacy, and subtracting another one because the matrix must be rank-two, leaves seven. Generally, when minimising a cost function, for example by using Levenberg-Marquardt, it is possible to minimise only over seven parameters rather than nine. A particular parameterisation may be chosen whereby any values of the seven free parameters will correspond to a fundamental matrix which is guaranteed to be rank-two. One such parameterisation is described by Zhang and Loop [71]. Their method is not altogether straightforward however because the parameterisation is not unique – a family of parameterisations are defined which depend on the specific data values and underlying epipolar geometry. As the epipolar geometry is unknown before estimation (it is what is being sought), an artifact of this method is that it may be necessary to change parameterisations part-way through the LM minimisation process.

It is not possible to adopt such a parameterisation of the fundamental matrix for use with FNS. This is because such a re-parameterisation necessitates a non-linear transformation of parameters which is inconsistent with the form on of Eq. (2.2), the principal constraint. Therefore, to use FNS for the estimation of the fundamental matrix, direct or iterative correction must be applied. The CFNS method, of section 3.4, however can be used directly.

5.6.3 CFNS

As an alternative to these methods, CFNS can be used to estimate a fundamental matrix directly. The method outlined in Sect. 3.4 can be employed to directly compute an estimate which minimises J_{AML} while satisfying the ancillary, rank-two, constraint. The rank-two constraint is cubic in the elements of the fundamental matrix (indeed it is occasionally referred to as the *cubic constraint*), therefore when computing the relevant matrices the value $\kappa = 3$ is taken.

5.7 Experiments part II: Constrained estimation

This section presents results of tests conducted to evaluate the performance of CFNS as a method for estimating the fundamental matrix. The same experimental setup was used as in the previous, unconstrained, experiments. Only results for the more challenging configuration B are given, although similar results were obtained for configuration A. For these experiments, the following estimation methods were used: TLS, HRT, FNS, CFNS, and GS. Not all of these methods produce a constrained estimate, in which case a post correction is applied as the final step. This post-correction was applied to HRT and TLS. The suffix "++" designates that an iterative correction (see Sect. 5.6.1) was used, followed by SVD – to make entirely sure the constraint was satisfied. This was applied to FNS. The remaining methods, CFNS, and GS, produce constrained estimates by design, so no post-constraining was necessary.

The first experiment investigated the effectiveness of CFNS as a means of finding constrained estimates. Fig. 5.8 (left) shows average (absolute) values of the ancillary constraint function $|\psi|$ for estimates of CFNS and FNS under increasing levels of noise. While the (unconstrained) FNS estimates have non-zero values of the constraint, CFNS produces estimates which in fact do satisfy the constraint – even for increasing levels of

noise. The respective average values of J_{AML} are shown in Fig. 5.8 (right), where the small trade-off required to satisfy the ancillary constraint is apparent.

Fig. 5.9 shows histograms of J_{AML} values for estimates, all of which satisfy the rank-two constraint. On average, the J_{AML} values of the CFNS estimates are lower than the J_{AML} values of the FNS++ estimates. Also apparent is the difference between FNS+ and FNS++ estimates, with the iterative correction leading to J_{AML} values lower than those generated by the direct, SVD, correction. Fig. 5.10 contains histograms of reprojection errors to ideal data points. The results show that in almost all cases the CFNS and FNS++ methods produce estimates which are indistinguishable from the GS estimates – at a much reduced computational cost.

Left epipoles for each of the HRT+, TLS+, FNS++ and CFNS estimates are shown in Fig. 5.11. The reduction in the bias is evident between the un-normalised TLS and the normalised HRT.

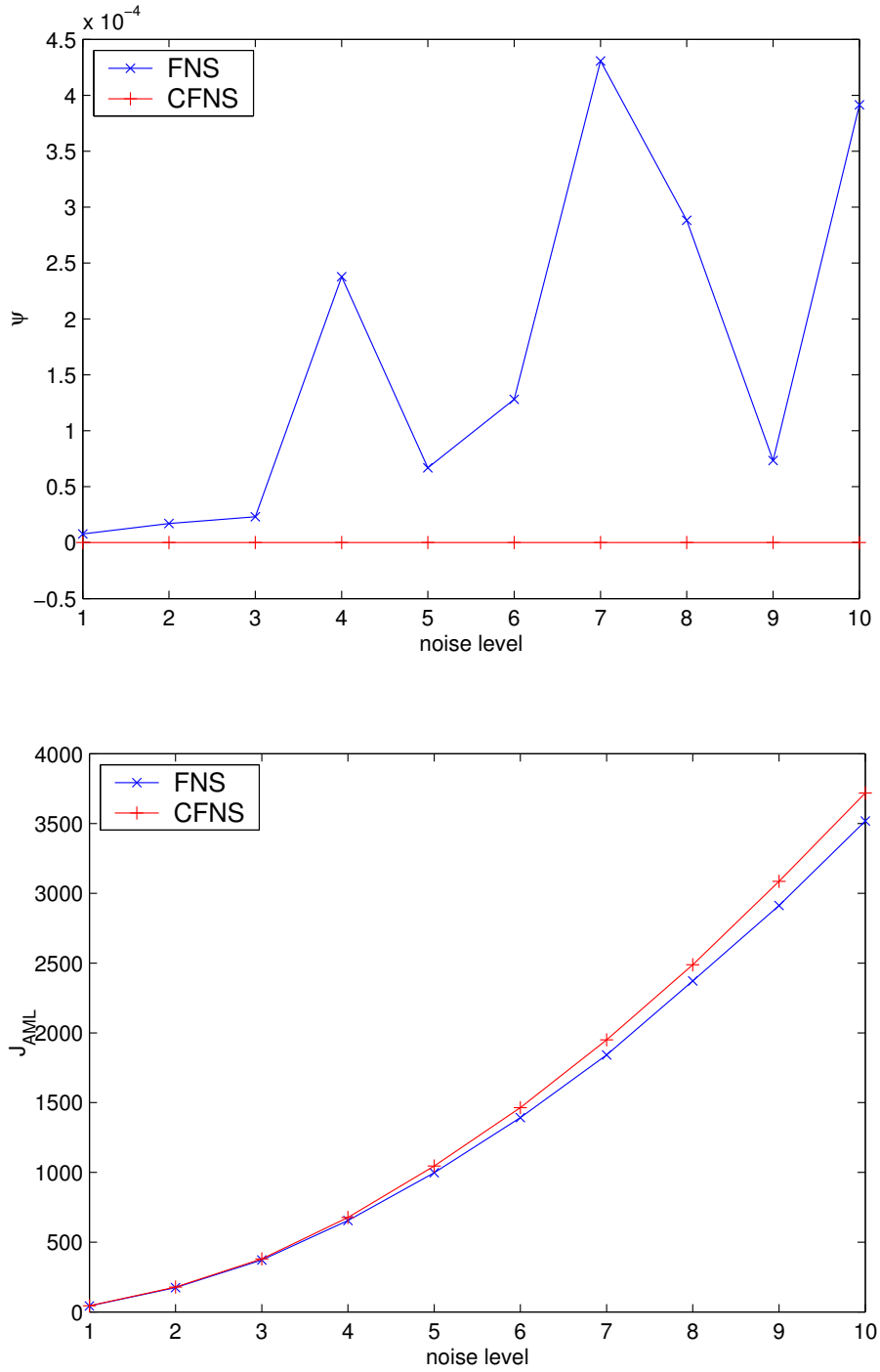


Figure 5.8: Average values of the (ancillary) constraint function for increasing noise (left). Average J_{AML} values for the estimates (right)

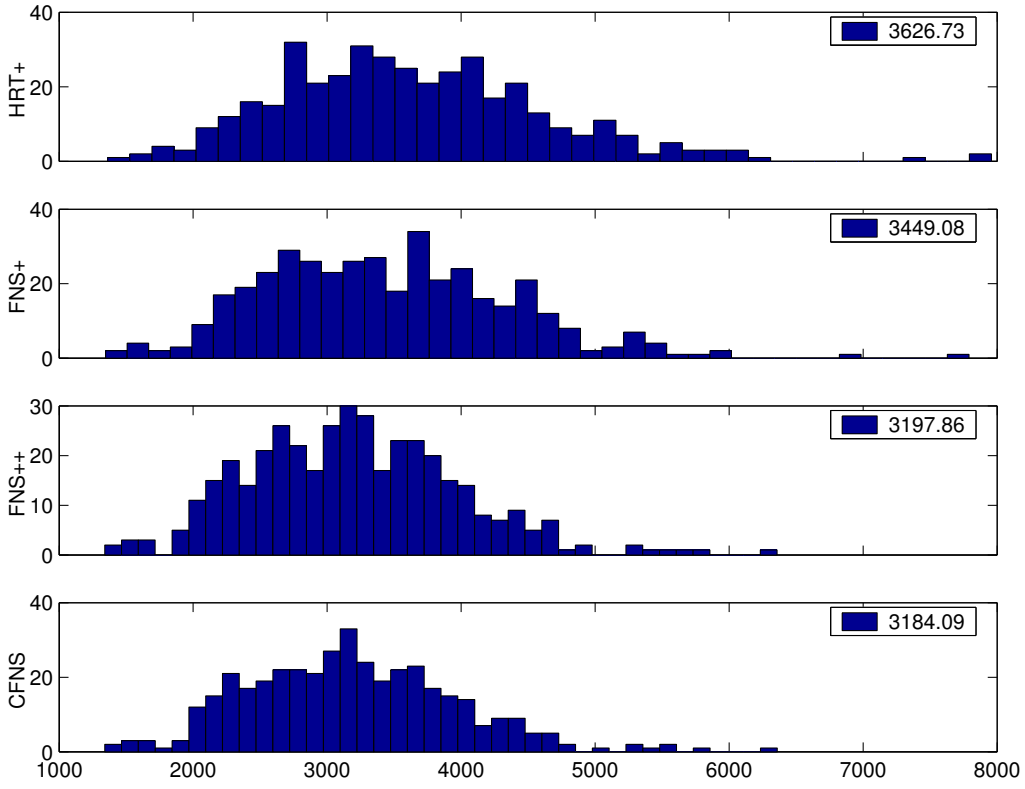


Figure 5.9: Histogram of J_{AML} values for estimates satisfying the constraint

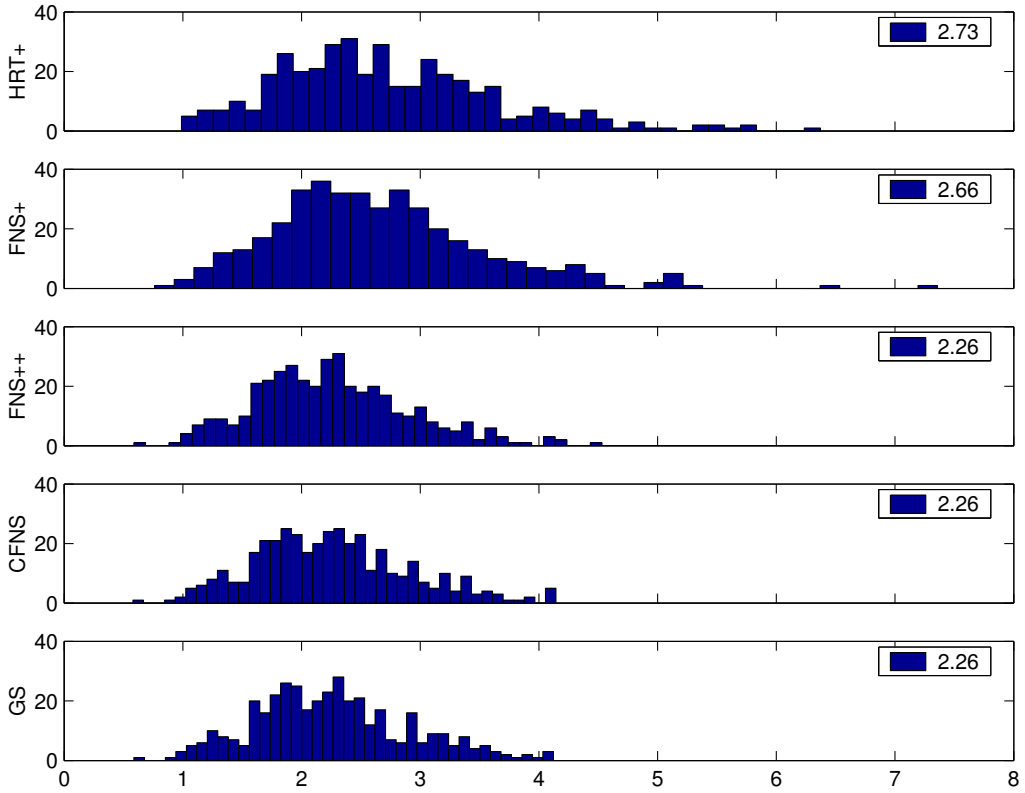


Figure 5.10: Histogram of reprojection to truth error values for estimates satisfying the constraint

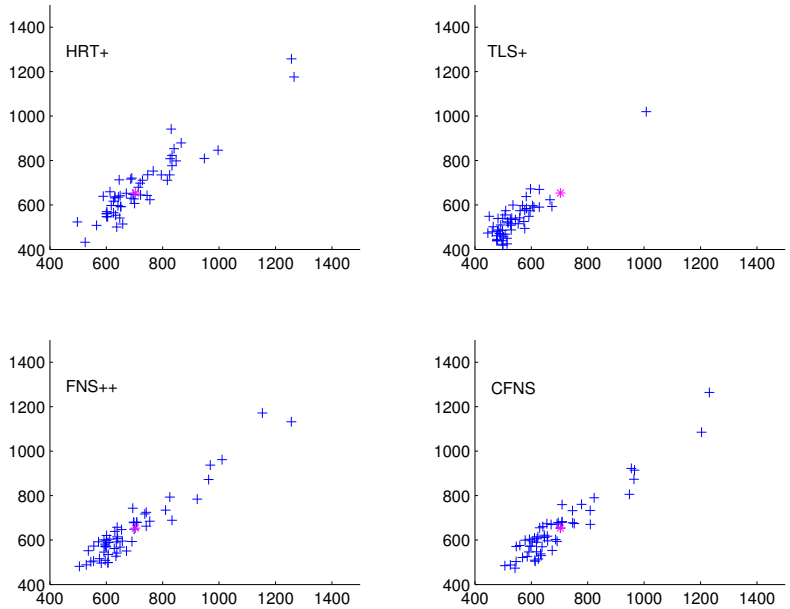


Figure 5.11: Left epipoles for various estimates. Epipoles associated with synthetic underlying true data are shown for reference

Chapter 6

COVARIANCE MATRICES

In the earlier description of the ML cost function (Sect. 2.5.1) , the covariance of each of the introduced random variables, of which each data point is a sample, were assumed either to be known, or set to a 'default' identity matrix. In almost all derivations of ML estimators used in computer vision, the above is the end of the story: covariance matrices appear as part of the Mahalanobis distance, or as an element of a ML cost function, but then are quickly assumed to be identity matrices, and forgotten. In this chapter the idea of using *non-default* covariance matrices as input to the estimation process is explored.

The chapter is divided into two parts: one deals with synthetic data, and the other real data. Using synthetic experiments it is possible to observe the effect of covariance information on estimators as precise control over the data and noise is available. A useful geometric parameterisation of a covariance matrix is presented and used in subsequent experiments where an estimator is given *different* input covariance matrices to those to generate the initial noisy data. Experiments involving real images show how covariance information might be obtained and used in epipolar geometry estimation. Several experiments are employed to ascertain the usefulness of this approach.

6.1 Covariance matrices

6.1.1 Single data point

A measured point $\boldsymbol{x} = (x, y)$ is a random perturbation from an underlying true point $\overline{\boldsymbol{x}}$. Under the ordinary model

$$\boldsymbol{x} = \overline{\boldsymbol{x}} + \Delta\boldsymbol{x}, \qquad \Delta\boldsymbol{x} \sim N(\mathbf{0}, \boldsymbol{\Lambda}_{\boldsymbol{x}}),$$

where $\Delta\boldsymbol{x} = (\Delta x, \Delta y)$ is the perturbation and $N(\mathbf{0}, \boldsymbol{\Lambda}_{\boldsymbol{x}})$ is the bivariate Gaussian distribution with mean $\mathbf{0}$ and covariance $\boldsymbol{\Lambda}_{\boldsymbol{x}}$. The individual components are defined

specifically by

$$\mathsf{E}[\Delta x] = \mathsf{E}[\Delta y] = 0\,, \tag{6.1}$$

$$\boldsymbol{\Lambda}_{\boldsymbol{x}} = \mathsf{E}\left[(\Delta \boldsymbol{x})(\Delta \boldsymbol{x})^{\top}\right] = \begin{bmatrix} \mathsf{E}[(\Delta x)^2] & \mathsf{E}[\Delta x \Delta y] \\ \mathsf{E}[\Delta y \Delta x] & \mathsf{E}[(\Delta y)^2] \end{bmatrix}. \tag{6.2}$$

In the case where x and y are uncorrelated, the covariance matrix will take the form

$$\boldsymbol{\Lambda}_{\boldsymbol{x}} = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}. \tag{6.3}$$

If $\sigma_x = \sigma_y$, then the covariance matrix is called *isotropic*, and the probability density function (p.d.f.) is symmetric with circular level sets. If $\sigma_x \neq \sigma_y$, then the covariance matrix is *anisotropic*, and the p.d.f. will have elliptical level sets (see Figure 6.1).

A canonical parameterisation of a covariance matrix as per Eq. (6.3) is

$$\boldsymbol{\Lambda}_{\boldsymbol{x}} = \alpha \begin{bmatrix} \beta & 0 \\ 0 & 1-\beta \end{bmatrix}, \tag{6.4}$$

where $\alpha = \sigma_x^2 + \sigma_y^2$ is the *scale*, and $\beta = \sigma_x^2/(\sigma_x^2 + \sigma_y^2)$ is the *eccentricity*.

The most general case is where there is some correlation between x and y. In this case, the covariance matrix may be written

$$\boldsymbol{\Lambda}_{\boldsymbol{x}} = \alpha \boldsymbol{R}_{\gamma} \begin{bmatrix} \beta & 0 \\ 0 & 1-\beta \end{bmatrix} \boldsymbol{R}_{\gamma}^{\top}, \tag{6.5}$$

where

$$\boldsymbol{R}_{\gamma} = \begin{bmatrix} \cos\gamma & -\sin\gamma \\ \sin\gamma & \cos\gamma \end{bmatrix} \tag{6.6}$$

is a rotation matrix parameterised by an angle of (anti-clockwise) rotation γ. In this case, the level sets of the associated p.d.f. will still be ellipses, now rotated by the angle γ. The scale of a covariance, α, must be greater than zero. The eccentricity must lie in the range $\frac{1}{2} \leq \beta \leq 1$, and the angle of rotation is restricted to $0 \leq \gamma < \pi$.

This parameterisation describes the geometry of the ellipses which are the level sets of the p.d.f.. The representation of a covariance matrix as (α, β, γ) will be useful in later synthetic experiments.

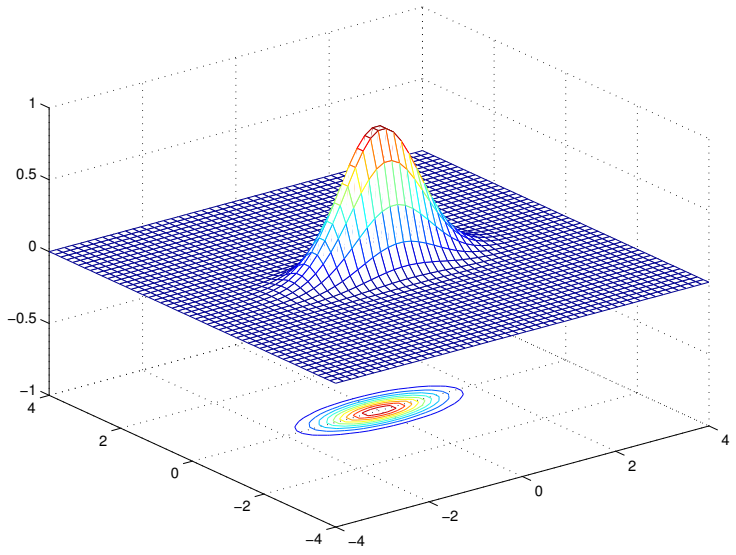


Figure 6.1: The level sets of a bivariate Gaussian p.d.f., characterised by a specific covariance matrix, are ellipses

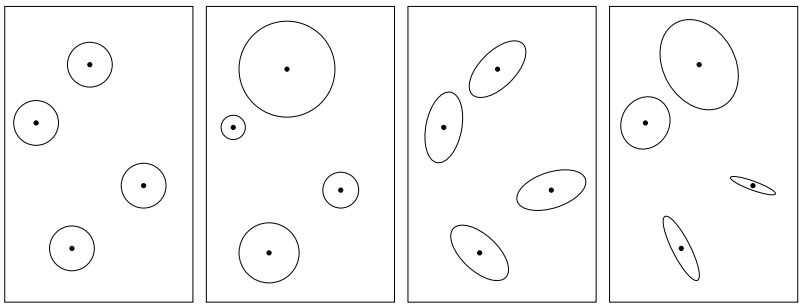


Figure 6.2: Different types of noise models: (a) isotropic homogeneous, (b) isotropic inhomogeneous, (c) anisotropic homogeneous, (d) anisotropic inhomogeneous

6.1.2 Collection of data

A collection of data points may be classified according to the type of associated covariance matrices. There are four categories:

- *Homogeneous isotropic* describes a model of measurement error in which all items of data are assumed to have the same, isotropic, distribution. An example is shown in Fig. 6.2(a). Covariance matrices in this case are called identity covariances, as they are all a common scalar multiple of the identity matrix.

- *Inhomogeneous isotropic* is a generalisation of the above category, where the noise is isotropic but the standard deviation may change from point to point. An example is shown in Fig. 6.2(b). Each covariance matrix is a (different) multiple of the identity matrix.

- *Homogeneous anisotropic* noise is shown in Fig. 6.2(c). Each covariance matrix has the same scale, but describes anisotropic noise.

- *Inhomogeneous anisotropic* is the most general case. As shown in Fig. 6.2(d), the scale and orientation of each covariance matrix may vary.

6.2 Experimental setup

The overall aim of these experiments is to understand how covariance information effects the estimation process. Firstly, several synthetic experiments are performed, so

that control can be exercised over the noise conditions. Synthetic testing is similar to that in previous applications chapters, except the process of adding noise is more involved. In addition to generating true data points, a covariance matrix is also generated, which is associated with each data point. Noisy data points are then generated consistent with these covariance matrices.

6.2.1 Estimation methods

To judge the effect of incorporating covariance information on estimation, several of the estimation methods used so far were employed. Methods capable of incorporating covariance information were run in two modes: ordinary and default ("starred"). For ordinary operation, the noisy data with available covariance information is supplied to the method. In default mode, the noisy data and *identity* covariance matrices only are supplied, regardless of whether non-default covariance information was used to generate the data. In this mode, we therefore test the use of default covariance matrices applied to the general problem.

6.2.2 Generating true and perturbed data

For estimating the fundamental matrix, the synthetic true data was created in the same way as in previous experiments. A stereo configuration, with non co-planar optical axes, and slightly differing left and right camera intrinsic parameters was fixed. A set of 50 randomly chosen 3-D points were then projected onto the images so as to generate many pairs of corresponding points $(\boldsymbol{m}^*, \boldsymbol{m}'^*)$, where, again, $\boldsymbol{m} \equiv [m_x, m_y, 1]^\top$ is a homogeneous point. Alternatively, image points $(\boldsymbol{m}^*, \boldsymbol{m}'^*)$ are described as $(\boldsymbol{p}^*, \boldsymbol{p}'^*)$ such that $\boldsymbol{p} = [m_x, m_y]^\top$ and $\boldsymbol{p}' = [m'_x, m'_y]^\top$.

Although the method for determining ideal points (and noisy points) is set, what is left open is how to choose the associated covariance matrix for each image point. By adopting the component parameterisation of a covariance matrix, how these covariances are chosen can be described in terms of their component parameters. With an average level of noise denoted as σ, a covariance matrix is found by

- choosing a random scale, such that α conforms to the uniform distribution $\boldsymbol{U}(0, 2\sigma)$,
- choosing a random skew, such that $\beta \sim \boldsymbol{U}(\frac{1}{2}, 1)$, and
- choosing a random rotation $\gamma \sim \boldsymbol{U}(0, \pi)$.

Since $\operatorname{tr} \boldsymbol{\Lambda}_{\boldsymbol{p}^*} = \alpha$ and $\mathsf{E}[\alpha] = \sigma$, it follows that $\mathsf{E}[\operatorname{tr} \boldsymbol{\Lambda}_{\boldsymbol{p}^*}] = \sigma$, giving the statistical interpretation of the assumed level of noise.

Given a true point $\boldsymbol{p}^*$ and an associated covariance matrix $\boldsymbol{\Lambda}_{\boldsymbol{p}^*}$, a noisy point $\boldsymbol{p}$ consistent with $\boldsymbol{\Lambda}_{\boldsymbol{p}^*}$ was obtained by adding a vector $\Delta\boldsymbol{p}$ to $\boldsymbol{p}^*$. The vector $\Delta\boldsymbol{p}$ was generated using the following algorithm:

1. Find a matrix $\boldsymbol{V}$ such that $\boldsymbol{\Lambda}_{\boldsymbol{p}^*} = \boldsymbol{V}\boldsymbol{V}^T$; this can be done by performing, say, Cholesky decomposition of $\boldsymbol{\Lambda}_{\boldsymbol{p}^*}$ [10, §3.2.2]. Note that the Cholesky decomposition of $\boldsymbol{A}$ determines a matrix $\boldsymbol{C}$ such that $\boldsymbol{C}^\top\boldsymbol{C} = \boldsymbol{A}$. Therefore, $\boldsymbol{V}$ is simply derived from the Cholesky decomposition of $\boldsymbol{\Lambda}_{\boldsymbol{p}^*}^\top$.

2. Generate a random vector $\boldsymbol{r}$ (of length two), with each component drawn independently from the Gaussian distribution with zero mean and unit standard deviation.

3. Set $\Delta\boldsymbol{p} = \boldsymbol{V}\boldsymbol{r}$.

The perturbation $\Delta\boldsymbol{p}$ has the required properties of mean $\boldsymbol{0}$ and covariance $\boldsymbol{\Lambda}_{\boldsymbol{p}^*}$, since

$$\begin{aligned}
\mathsf{E}[\Delta\boldsymbol{p}] &= \mathsf{E}[\boldsymbol{V}\boldsymbol{r}] = \boldsymbol{V}\mathsf{E}[\boldsymbol{r}] = \boldsymbol{0}, \\
\mathsf{E}\left[(\Delta\boldsymbol{p})(\Delta\boldsymbol{p})^T\right] &= \mathsf{E}\left[(\boldsymbol{V}\boldsymbol{r})(\boldsymbol{V}\boldsymbol{r})^T\right] \\
&= \mathsf{E}\left[\boldsymbol{V}(\boldsymbol{r}\boldsymbol{r}^T)\boldsymbol{V}^T\right] = \boldsymbol{V}\mathsf{E}\left[\boldsymbol{r}\boldsymbol{r}^T\right]\boldsymbol{V}^T \\
&= \boldsymbol{V}\boldsymbol{V}^T = \boldsymbol{\Lambda}_{\boldsymbol{p}^*}.
\end{aligned}$$

Finally, each $\boldsymbol{p}$ was associated with a covariance matrix $\boldsymbol{\Lambda}_{\boldsymbol{p}}$, which was taken to be $\boldsymbol{\Lambda}_{\boldsymbol{p}^*}$.

6.3 Experiments with synthetic data

To provide a point of reference, firstly tests were conducted in estimating the fundamental matrix from noisy data, with the use of perfect, underlying, covariances. The procedure for generating noise was that given above. Each test involved randomly choosing 50 pairs of true corresponding points, and generating an associated covariance matrix for each point as described in Sect. 6.2.2. These covariances were anisotropic and inhomogeneous, and parameterised by a (common) average level of noise. For each true point, a *noisy point* was randomly chosen consistent with the corresponding covariance matrix, as described above. Four methods, namely FNS, and FNS*, and

TAU, and TAU* were then employed to estimate the fundamental matrix from the noisy points. In the case of FNS, the associated covariance matrices were taken into account. An error measure was computed for each estimate. This measure was the sum of the Euclidean distances of the underlying *true points* (in both images) to the epipolar lines derived from the estimated fundamental matrix. This measure was chosen for two reasons. Firstly, it has the property that it vanishes when the estimate coincides with truth. Secondly, it can be computed based on an unconstrained estimate. The purpose of these experiments was to find out how covariance information can effect unconstrained estimation, hence enforcing the rank-two constraint was left aside.

The mean value of the error for each of the estimation methods was computed over 200 repetitions. The entire process was performed for an average noise level σ ranging from 2.0 to 10.0 in steps of 2.0. The results are presented in Fig. 6.3. This figure shows that there is a clear ordering to the quality of the estimates. Unsurprisingly, the FNS estimate have the lowest epipolar error, as they rely on additional covariance information. Note that there is little difference between TAU and TAU*. This is because the approximation "averages" out the variation in the denominator of J_{AML} which is contributed to from varying covariance matrices. The results show that there is significant benefit of incorporating covariance information when using FNS. However, any improvements are predicated on two factors, namely that

- the distributions of each individual data points are considerably varied, and
- it is possible to supply exact covariance information to an estimator.

The assumption in Sect. 6.1.2, to the effect that the component parameters α, β, and γ are distributed uniformly, means that a highly skewed covariance is as likely as an isotropic one. If such variation in the covariances associated with each point was reduced, it would be expected, heuristically, that the benefits of using covariances would be less. In practice, it is almost impossible to have the *exact* covariance information for each data point.

In the next sections, these issues are addressed by conducting further experiments. Firstly, the *distribution* of the component parameters is addressed. Secondly, experiments are performed where the estimators are given some, but not all, of the available covariance information.

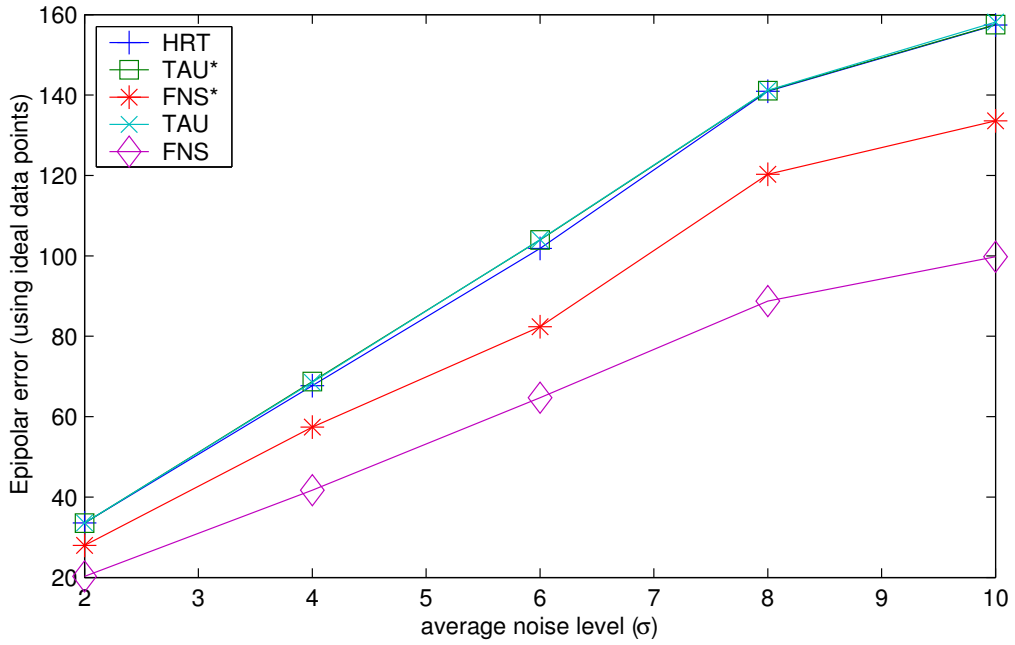


Figure 6.3: Average values of epipolar error for estimates produces with various methods

6.3.1 Varying the spread of covariances

A new experiment was carried out in which the level of anisotropicity and inhomogeneity was controlled. This was done by introducing the parameter $0 \leq \rho \leq 1$ to capture the 'spread' of the skew and scale parameters. Fixing the level of noise, variation of skew was set to conform to

$$\beta \sim \boldsymbol{U}(\frac{1}{2}(1-\rho), \frac{1}{2}(1+\rho))$$

and variation of scale was set to conform to

$$\alpha \sim \boldsymbol{U}(\sigma(1-\rho), \sigma(1+\rho)).$$

The following tests were performed:

- Skew was selected as above, and scale was kept constant, corresponding to homogeneous anisotropic noise.

- Scale was selected as above, and skew was set to $1/2$. This corresponds to inhomogeneous isotropic noise.

- Both parameters were selected as above. This corresponds to inhomogeneous anisotropic noise.

The testing regimen as above was repeated, although TAU was discarded with TLS was incorporated to obtain a sense of scale. The results are presented in Figure 6.4. As ρ increases, the error of the estimate generated using covariance matrices reduces. Under these circumstances, the improvement is more marked for the scale parameter than the skew parameter. For a given average level of noise, estimates obtained via FNS improve with increased diversity of covariance. Note that the average error for the two methods not using covariances remains constant, since the average level of noise remains constant. This is interesting in itself, because as ρ increases, the covariances generated are different, and hence noisy points generated will be differently distributed. For $\rho = 0$ there will be (in the third case of choosing both scale and skew) isotropic and homogeneous noise. As ρ increases the covariances are becoming more and more anisotropic and inhomogeneous. However, if the average level of noise is kept fixed, the FNS* and TLS methods produce estimates which, on average, have the same error.

6.3.2 Partially correct covariances

In practice noisy data points are measured, not the true underlying values, and so it is not possible to measure exactly the covariances driving the noise in the data. The next experiments aim to see how some loss of covariance information can effect the quality of estimates. Tests are conducted where a modified version of the real covariance matrices used to generate noisy points are supplied to estimators. True underlying covariance matrices are associated with each true point, and are used to generate noisy data. Then, a different kind of 'noise' is added to the covariance matrices themselves. The noisy data and noisy covariance matrices are then passed to an estimator.

The true covariance matrices were perturbed by multiplying each of their underlying component parameters by a random factor $(\alpha, \beta, \gamma) \mapsto (k_1\alpha, k_2\beta, k_3\gamma)$, where k_1, k_2, k_3 are chosen independently from the Gaussian distribution with mean of one, and standard deviation τ. In this way, the level of noise added to the covariances was controlled by the deviation τ. It should be noted that whenever the multiplication of the parameter caused it to exit the range specified in Sect. 6.2.2, it was clipped at the maximum or minimum appropriately. In particular, the new scale $k_1\alpha$ was kept to a maximum of 2σ to ensure the average level of noise was at least similar to that used originally.

The algorithm can be summarised as follows:

1. Generate synthetic *true* data.
2. Generate random covariance matrices as per Sect. 6.2.2, with a fixed average noise level.
3. Perturb data in accordance with the covariance matrices to obtain noisy data.
4. Create a new set of covariance matrices by adding noise to the original matrices.
5. Compare estimates obtained by supplying identity, true, and noisy covariance matrices.

Again, these steps were repeated 200 times to obtain an average error for the three estimation methods. Unlike the first experiments, where the level of noise added to the *data* was varied, in these experiments the average level of noise added to the data was held constant. What was varied was the level of noise added to the covariance matrices. Data points, along with these noisy covariance matrices, were used to generate an estimate using the FNS method. This was labelled FNS-n.

Figure 6.5 highlights the roughly linear increase in error of FNS estimates as the supplied covariance information becomes less accurate. In these experiments, we see that the FNS-n estimates are superior to FNS* esimates when $\tau < 0.3$ pixels. Hence using (noisy) covariances offer advantage over identity defaults up to that tolerance. For higher values of τ, the FNS-n estimate is worse than that of FNS*, so a better estimate is produced by ignoring any covariance information available (and falling back on identity defaults).

6.4 Considering real images

The problem now considered is to apply some of this knowledge to the task of incorporating covariance information when estimating the fundamental matrix from a pair of real images.

A key observation when dealing with covariance matrices associated with data points, for example those extracted from images, is that the error associated with their measurement is generated through a particular *measurement process*. Therefore it is entirely

possible that the same image point can have a different associated covariance matrix depending on the process by which it was measured. The procedure by which a covariance matrix can be associated with a data point can be derived in terms of the component parameters α, β, and γ of Eq. (6.5). Such a process can be described as follows:

- Determine the maximal direction along which the measurement process is susceptible to the largest perturbation. Under the assumption that the noise effecting true points is Gaussian, the direction with the minimal chance of perturbation is orthogonal. This maximal direction, once determined, can be used to determine γ. (It is set to the angle that the maximal direction makes with the $x-$axis.)

- Find a ratio r of the expected values of perturbations in the maximal and minimal directions. This can be used to determine the skew parameter, with β taken as $r(1+r)^{-1}$.

- Determine an overall measure of the scale of error by adding the expected values of perturbations in the maximal and minimal directions. This value can be taken for α. Although the estimation process will be unaffected by global scale change applied to all covariances, this parameter determines the relative scales of each covariance matrix.

This procedure proves difficult to follow precisely in practice, however it can give some guidance as to what to take for covariance matrices.

6.4.1 Covariance information from image gradients

An image is described by its intensity value $E(x, y)$ at every point $\boldsymbol{p} = [x, y]^\top$. Gradient information from the image is the main ingredient for analysis. The partial derivatives of E with respect to x and y, at a point, are denoted by $E_x(x, y)$ and $E_y(x, y)$. As the image is a discrete sample of intensities, the partial derivatives are approximated using simple differencing, with

$$E_x(x, y) = \frac{E(x+1, y) - E(x-1, y)}{2}, \tag{6.7}$$

and

$$E_y(x, y) = \frac{E(x, y+1) - E(x, y-1)}{2}. \tag{6.8}$$

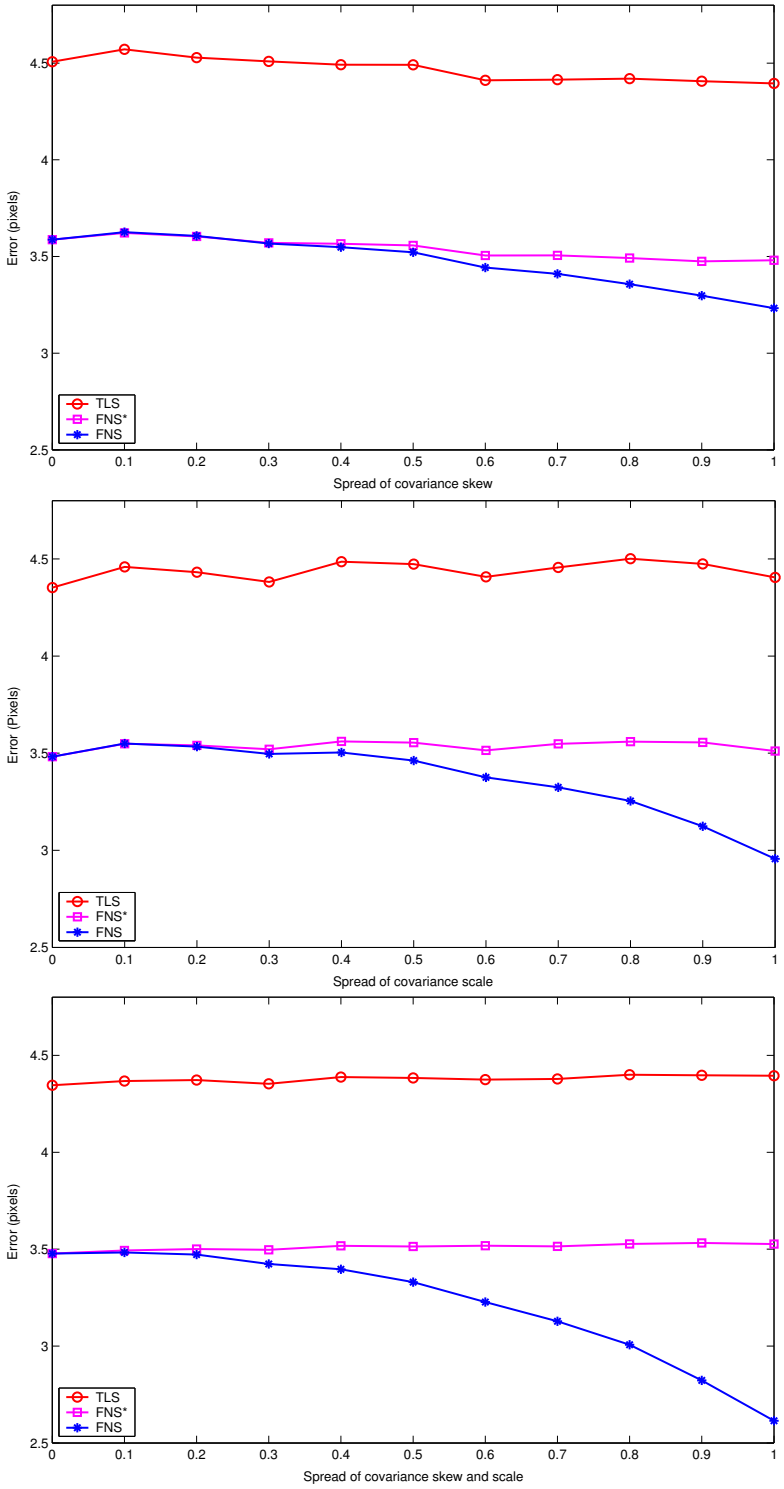


Figure 6.4: Errors in estimates for varying spread of covariance parameters

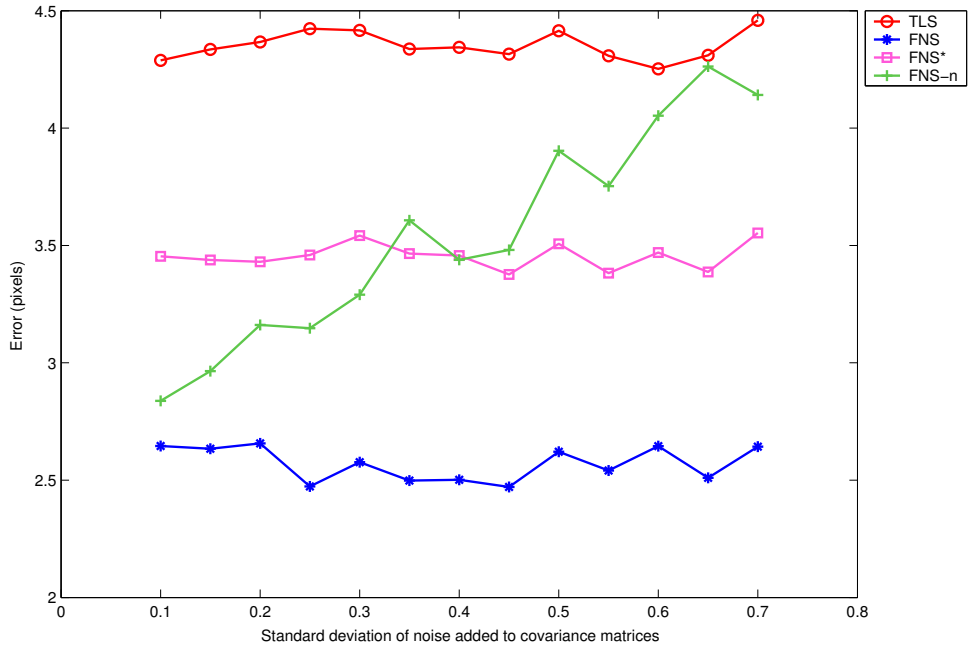


Figure 6.5: Average estimation error for increasing average error in covariance matrices

Often, rather than using these gradient values directly, often smoothed versions are employed. Applied to the gradients, the hat operator defines a smoothed version, defined formally as

$$\widehat{E_x^2} = \widehat{E_x^2}(x,y) = \sum_{(i,j)\in W} G(i,j) \times \left(E_x(x+i, y+j)\right)^2, \qquad \text{with,} \tag{6.9}$$

$$G(i,j) = (2\pi\sigma^2)^{-1} \exp\left(-(i^2+j^2)/2\sigma_W^2\right),$$

where W is a window of pixel values centred at $(0,0)$ (e.g. $W = [-k,k] \times [-k,k]$ for some positive integer k) and σ_W is a fixed standard deviation expressed in pixels. For subsequent experiments, the value of $k = 7$, and $\sigma_W = 1.0$ were used. Other combinations, $\widehat{E_y^2}$, $\widehat{E_y E_x}$ and $\widehat{E_x E_y}$ are defined analogously.

For a given points $\boldsymbol{p}$, a covariance matrix of the form

$$\boldsymbol{\Lambda}_{\boldsymbol{p}}^1 = \frac{1}{\widehat{E_y^2}\widehat{E_x^2} - \widehat{E_y E_x}\widehat{E_x E_y}} \begin{bmatrix} \widehat{E_y^2} & -\widehat{E_y E_x} \\ -\widehat{E_x E_y} & \widehat{E_x^2} \end{bmatrix}.$$

has been used in optical flow computation [47], texture segmentation [11], and shape from factorisation methods [1, 58]. This matrix form is derived given the idea that the particular image point is best located in the direction of highest intensity change.

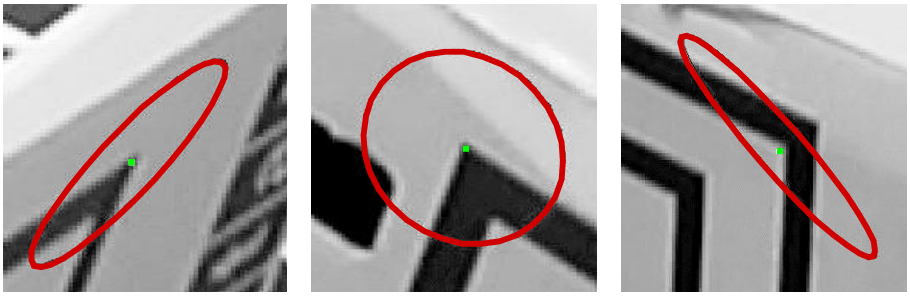


Figure 6.6: Ellipses representing covariance matrices for various image points

The leading scale coefficient ensures that overall,the uncertainty is inversely to the rate of change of image intensity: the smaller the rate of change in image intensities, the greater the uncertainty.

Fig. 6.6 shows ellipses representing covariances of form $\mathbf{\Lambda}^1$ for three different image points. The ellipses are rendered as follows. The covariance matrix is subjected to SVD, $\mathbf{\Lambda} = \boldsymbol{J}\boldsymbol{D}\boldsymbol{K}^\top$. As $\mathbf{\Lambda}$ is symmetric, it follows that $\boldsymbol{J} = \boldsymbol{K}$. The SVD form is similar to that in Eq. (6.5), with $\boldsymbol{J}$ replacing the rotation matrix $\boldsymbol{R}_\gamma$. The ellipse's major and minor axes are taken as the singular values (the diagonal elements of $\boldsymbol{D}$), and the rotation is determined from $\boldsymbol{J}$.

6.4.2 Harris corners

It is important to consider covariance information in the context of a measurement process. The measurement process adopted to find candidate points in an image (for subsequent matching) was the Harris corner detector [22]. The detector evaluates a corner response function $H(x, y)$ which is defined for every point in the image. Points whose value of H is high enough, and which are local maxima, are taken as a set of Harris points (or "corners") for the input image. Harris points are found for a left and right image, and it is only this subset of points which are used to determine corresponding points.

The response function is defined also in terms of smoothed gradients. Its form is

$$H(x,y) = \begin{vmatrix} \widehat{E_y^2} & -\widehat{E_y E_x} \\ -\widehat{E_x E_y} & \widehat{E_x^2} \end{vmatrix} - \kappa \left(\widehat{E_y^2} + \widehat{E_x^2} \right) \tag{6.10}$$

where κ is a small constant, assigned usually as $\kappa = 0.04$.

The smoothing applied to the gradient values is important. If un-smoothed values were used, for example $E_x E_y$ instead of $\widehat{E_x E_y}$, then the (left hand) matrix would have determinant zero, and therefore only the small correcting factor would be taken as the response function. Because the values are smoothed, then (in general) $\widehat{E_x^2}\widehat{E_y^2} \neq \widehat{E_x E_y}^2$, and the matrix has a non-zero determinant. If locally there is no variation in intensity then E_x and E_y will both be zero, and the determinant will in this case be zero. Consequently, H attains a small value – as it should given that this part of the image is not a feature of interest.

It is possible to estimate a fundamental matrix by taking a series of matched Harris points from a stereo image pair, and computing $\mathbf{\Lambda}^1_{\boldsymbol{p}}$ for each image point. The points and associated covariances can be passed to an estimator. A series of Harris points and their associated covariance matrices rendered as ellipses are shown in Fig. 6.8.

The form of $\mathbf{\Lambda}^1$ and the Harris response function $H(x, y)$ are very similar. When $\kappa = 0$, the scale factor of each covariance is simply $H^{-1}(x, y)$. So points with a high value of H are better located. The only points which are selected as candidates for matching, and hence used in estimation, are Harris points, by definition those with the highest values of $H(x, y)$. The corresponding covariance matrices will all have similar scales. As a result there will be little variation in the scales of a set of Harris points, and so the advantage of using covariance information in estimation will be lessened. In relation to the experiments from Sect. 6.3.1, the situation will be similar to that where ρ is close to zero. Kanazawa and Kanatani [32] present this argument in their analysis of the effectiveness of using covariance information in estimation.

However, looking at Fig. 6.8, it is clear that although the scale characteristics may be similar, there is a large variation in the skew (and rotation) components of covariance matrices for each point. However, in many cases, the direction of minimal error seems to be orthogonal from where it should be, at least heuristically. Many of the Harris points tend to be estimated "inside" a corner, where the direction of most uncertainty is along the bisector of the corner. However, there are many cases where the covariance matrix of form $\mathbf{\Lambda}^1$ produces the maximal error directions orthogonal to this. This suggests a slightly different form for covariance matrices, which use the same scale, but with a different orientation. This form is given as

$$\mathbf{\Lambda}^2 = \frac{1}{\widehat{E_x^2}\widehat{E_y^2} - \widehat{E_x E_y}\widehat{E_y E_x}} \begin{bmatrix} \widehat{E_x^2} & \widehat{E_x E_y} \\ \widehat{E_y E_x} & \widehat{E_y^2} \end{bmatrix} . \tag{6.11}$$

Covariance matrices of this form are shown in Fig. 6.9 for the same Harris points. It can

be seen that for points "inside" a corner, the axis of highest uncertainty tends to align with the bisector of the corner.

6.4.3 Real images

To test both forms of these covariance matrices, the fundamental matrix was computed for three different real stereo image pairs. For a given stereo pair,

- Harris corners were extracted from each of the left and right images,
- corresponding pairs of points were *manually* determined,
- for each of the left and right points, three corresponding covariances were used: those of form $\boldsymbol{\Lambda}^1$, $\boldsymbol{\Lambda}^2$, and $\boldsymbol{I}$ (the identity matrix),
- CFNS was used to estimate a fundamental matrix,
- the error measure $J_{\mathrm{ML}}(\widehat{\boldsymbol{F}})$ was evaluated for each estimate $\widehat{\boldsymbol{F}}$ (using data points instead of true points due to the lack of ground truth).

Points were matched manually to discount the presence of any mis-matched points causing errors. As the purpose of this experiment was to determine the usefulness of covariance information, it was important that mis-matched points be eliminated as a artificial source of error. For these experiments, as true data were obviously unavailable, the previous error measure involving the distance to left and right epipolar lines could not be used. Instead, the ML cost function was employed. To use this cost function however, the estimated fundamental matrices had to satisfy the rank-two constraint, necessitating the use of CFNS.

The results are presented in Table 6.4.3. This table shows that for each of the three tested stereo pairs, the fundamental matrix estimated using $\boldsymbol{\Lambda}^2$ covariances gives errors which is less than those when identity covariances are used. Also, the fundamental matrices generated using $\boldsymbol{\Lambda}^1$ covariances are either on par with, or worse than, using identity covariances.

images	$\mathbf{\Lambda}^1$	I	$\mathbf{\Lambda}^2$
grid	2.52	1.40	0.98
office	2.48	2.47	1.43
library	2.76	1.61	1.07

Table 6.1: Errors recorded for the three different stereo pairs

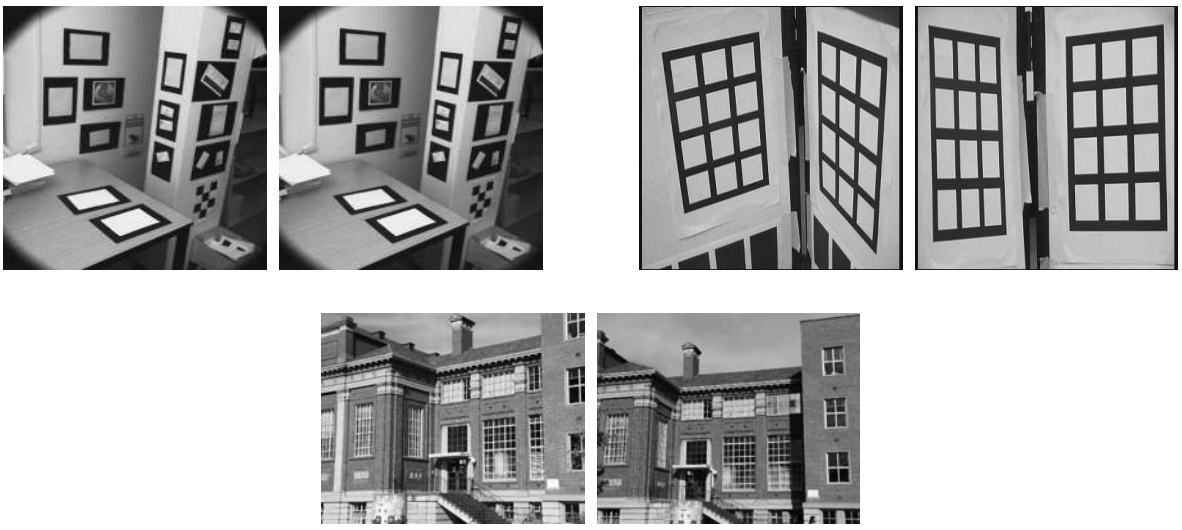


Figure 6.7: The office, grid, and library stereo image pairs

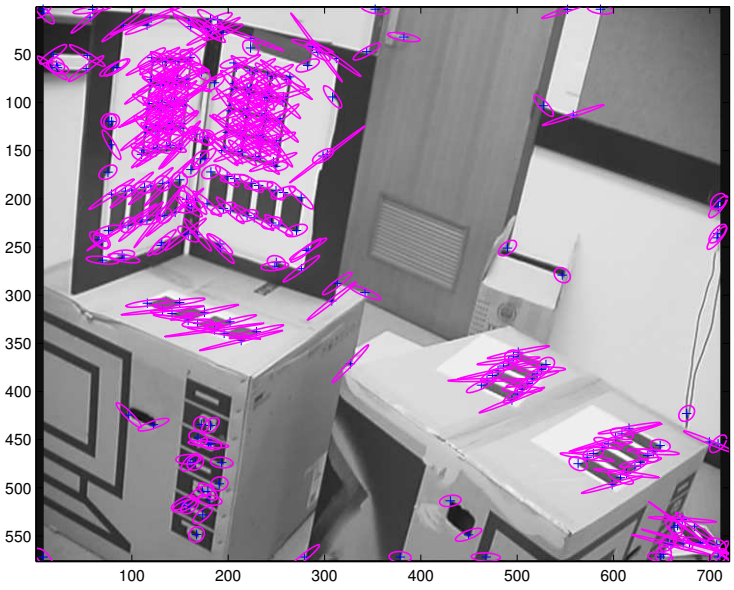


Figure 6.8: Covariance matrices of form Λ^1

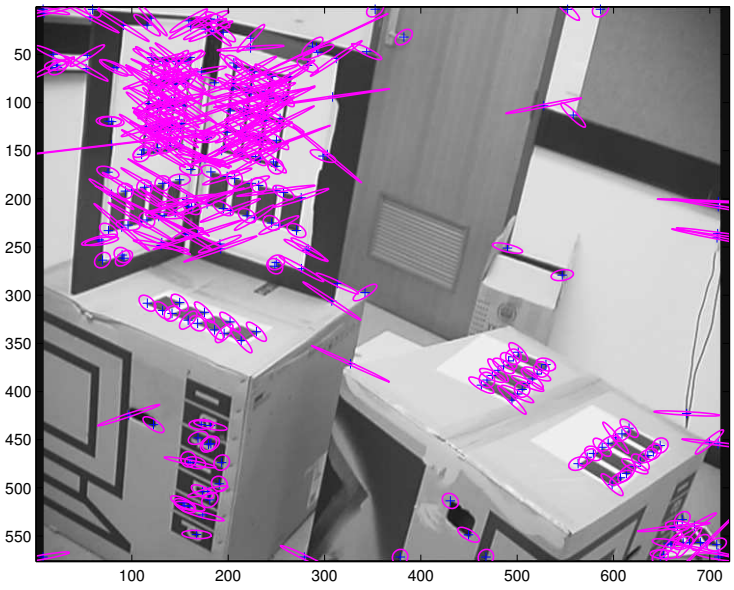


Figure 6.9: Covariance matrices of form Λ^2

Chapter 7

CONCLUSION

This work has explored issues in relation to estimating parameters that arise in certain computer vision problems. This chapter summarises the main results obtained and highlights areas for further exploration.

7.1 Summary

A parametric model was defined describing how image features, used as data, are related to model parameters. The model adopted was that of a single objective principal constraint function, sufficient to describe, for example, the problems of conic fitting and fundamental matrix estimation. Standard least-squares estimation techniques were applied to this form, and an algebraic estimator (total least squares) was described for reference. After showing that algebraic methods exhibit systematic bias, a conventional maximum likelihood (ML) cost function was adopted. Following the method of Sampson, a widely used approximated maximum likelihood (AML) cost function was formally derived and supported with geometric analogy.

Next, attention was turned to finding a minimiser for this cost function. General minimisation methods, including the Levenberg-Marquardt method, were discussed. A method specific to the AML cost function, iteratively re-weighted least squared, was also described. Next, a new approach was taken, whereby properties of the minimiser itself were considered explicitly. By analytically deriving the gradient of the AML cost function, a variational equation was defined describing the parameters at a (local) minimum of the cost function. Critically, the gradient was shown to have a special form whereby the parameter vector is involved through multiplication by a factor matrix. This insight was used to devise the FNS method for computing a minimiser of the AML cost function, where evaluation of this factor matrix plays a key rôle.

Through considering the form of this factor matrix, and the consequences of the variational equation, insights were also derived regarding other methods. For example,

Taubin's "eigenfit" method can readily be placed within this framework. It was shown theoretically that the iteratively re-weighted least squares method does not act to minimise the AML cost function. The form of the factor matrix was additionally applied to considering an appropriate stopping condition for the iterations involved in FNS.

The ordinary, unconstrained FNS method was expanded to incorporate an additional ancillary constraint on the parameters. This was achieved by considering an expanded system of equations, building on the variational form used previously, and incorporating an Lagrangian multiplier. A specific restriction of the form of the ancillary constraint, that it be homogeneous of some arbitrary degree, was used to derive key identities involving the constraint function. These identities were used in turn to build a system which bears resemblance to the original variational equation used for FNS. Algebraic manipulation was performed on this system to reduce its dimensionality, rendering a tractable form. Additional modifications were adopted to aid convergence. This final form was used to define the constrained estimation method CFNS.

Estimation methods were first applied to the problem of (unconstrained) conic fitting. Experiments were conducted whereby FNS was compared to minimising the AML cost function using the Levenberg-Marquardt optimiser (a method denoted LM-A). The results showed that, for different data sets generated using a different sub-arc of an ellipse, the FNS method was rapidly convergent, typically requiring only a handful of iterations to complete. The results further showed that the estimates produced by FNS coincide with those generated by LM-A, but were computed with significantly less computation cost. Successive estimates found using FNS were not necessarily monotonically decreasing in their cost function values, something that was borne out when the LM-A cost function surface was visualised using six "slices."

A harder problem than unconstrained conic fitting arises when the conic is constrained to be an ellipse. A popular algebraic method, of Fitzgibbon *et al* guarantees that its estimate will be an ellipse. After examining the form of this method, an analogous method was derived by considering the variational equation. This method allows an existing, unconstrained estimate to be upgraded to satisfy the ellipse constraint. Its advantage is that, in using the AML cost function, it exhibits significantly less bias than the algebraically based Fitzgibbon method. Experiments show that, for data which conforms to the proposed model, this correction produces estimates satisfying the ellipse constraint which also have a low value of the AML cost function. This method was shown to be additionally useful when used as a seed for the optimal (in a maximum

likelihood sense) gold standard method.

The second application considered was estimation of the fundamental matrix. Experiments with synthetic stereo data show again that, as was the case with estimating conic sections, the unconstrained estimator FNS typically produces fundamental matrix estimates on par with LM-A, but at a much lower computational cost. The well known method of Hartley normalisation was shown in the results to produce very good estimates. Even though it is an algebraically based method, it produced estimates almost as good as geometrically based methods. This method was examined in more detail, with a specific cost function developed showing exactly what entity was being minimised when following the normalisation procedure. It was shown that this cost function was an approximation to a geometrically inspired cost function, hence accounting for the quality of results. Critically, the improved results were seen to arise from the normalisation procedure causing the associated cost function to resemble this improved form, rather than from numerical benefits associated with better conditioned matrices.

Next, the problem of adopting the rank-two constraint was considered. The CFNS method was applied to the constrained fundamental matrix estimation problem and was compared to both the direct and iterative post correction methods, and the gold standard (true maximum likelihood) method. The CFNS method was shown to produce constrained estimates which minimise the AML cost function, while preserving the rank two constraint. Estimates produced using the CFNS method were compared to those generated by applying a post correction to unconstrained estimates. The results showed the CFNS estimates had cost function values at least as low as, and in some cases slightly lowers, than the post corrected estimates. The CFNS method was also shown to have superior convergence properties than the (unconstrained) FNS method.

Finally, the usefulness of incorporating covariance information as input to the estimation process was considered. Synthetic experiments were conducted under ideal conditions where covariance matrices used to create noisy data were passed directly to estimators. Unsurprisingly there was seen to be a benefit to the estimation process of extra information being made available. To more closely model a realistic situation, further experiments were conducted where the amount of variability in the underlying noise model was reduced, to enable the examination of the level of benefit still present. Further tests explored the issue of how much information must be known about an underlying covariance matrix for it to be generally useful for estimation. It was shown that critical to the adoption of covariance matrices when using data from real images

is to consider the likely sources of errors and use this knowledge when devising an appropriate covariance matrix. Experiments were conducted with real stereo image pairs with a particular form of covariance matrix adopted by considering the process by which image features were measured. This form was shown to improve the accuracy of estimates for the examples considered.

7.2 Future work

The parametric model considered in this thesis describes a single objective (scalar valued) principal constraint function. This model can be upgraded to consider the multi-objective case, where the principal constraint function is vector valued. Many of the forms involved can be adopted by analogously using the vector form and applying appropriate algebraic manipulation. The multi-objective case allows the immediate tackling of problems such as estimation of the Homography matrix [44] and tri-focal tensor [14]. It is possible to contemplate adopting other estimation problems from within and outside the realm of computer vision, which can be massaged into the appropriate form.

The usefulness of the FNS method (or even CFNS) could be explored in relation to its incorporation to robust estimation. When outliers are assumed to exist in the data, generally methods are followed whereby multiple random subsets of data are chosen, and estimates produced for each subset. Because the FNS method is typically able to be executed very rapidly, it may prove useful when incorporated at this stage.

In the constrained case, the CFNS method was used to impose the rank-two constraint when estimating the fundamental matrix. One possible extension of this work would be to impose additional constraints. For example, geometrically inspired constraints could be added such as requiring any estimated fundamental matrix to be consistent with a geometry where the cameras are in front of a scene. If information is known about the cameras, for example that they conform to a stereo-head configuration [6], or if they have fixed intrinsics, then this information could also be added. This would require a new version of CFNS to be devised capable of dealing with multiple constraints simultaneously.

Another avenue for exploration is further work with incorporating covariance information in the estimation process. The results from Chapter 6 indicate that there is potential benefit involved when estimating the fundamental matrix, as long as the

measurement process of data is considered. This work may be extended to consider incorporating covariance information in the case where more than two images are used, for example, by incorporating covariance information into general bundle adjustment techniques.

BIBLIOGRAPHY

[1] Henrik Aanæs, Rune Fisker, Kalle Åström, and Jens Michael Carstensen. Factorization with erroneous data. In *Proceedings of Photogrammetric Computer Vision (PVC '02)*, pages 15–23, Graz, Austria, September 2002.

[2] George B. Arfken and Hans-Jurgen Weber. *Mathematical Methods for Physicists*. Academic Press, 3rd edition, 1985.

[3] Luis Baumela, Michael J. Brooks, Wojciech Chojnacki, and Anton van den Hengel. Robust techniques for the estimation of structure from motion. In H. Burkhardt, editor, *Proceedings of the 5th European Conference on Computer Vision (ECCV '98)*, volume 1407 of *Lecture Notes in Computer Science*, pages 281–295, Freiburg, Germany, June 2–6 1998. Springer.

[4] Adi Ben-Israel and Thomas N.E. Greville. *Generalised Inverses: Theory and Applications*. Krieger, 1977.

[5] Fred L. Bookstein. Fitting conic sections to scattered data. *Computer Graphics and Image Processing*, 9:56–71, 1979.

[6] Michael J. Brooks, Wojciech Chojnacki, and Louis Baumela. Determining the egomotion of an uncalibrated camera from instantaneous optical flow. *Journal of the Optical Society of America A*, 14(10):2670–2677, October 1997.

[7] Keith Connor and Ian Reid. Novel view specification and synthesis. In *Proceedings of the 13th British Machine Vision Conference*, pages 243–252, Cardiff, UK, September 2002.

[8] David B. Cooper and Nese Yalabik. On the computational cost of approximating recognising noise-perturbed straight lines and quadratic arcs in the plane. *IEEE Transactions on Computing*, 25:1020–1032, 1976.

[9] Tim Ellis, Ahmed Abbood, and Beatrice Brillault. Ellipse detection and matching with uncertainty. *Image and Vision Computing*, 10(5):271–276, 1992.

[10] James E. Gentle *et al*. *Numerical Linear Algebra for Applications in Statistics*. Springer-Verlag, 1998.

[11] Annett Faber and Wolfgang Förstner. Scale characteristics of local autocovariances for texture segmentation. *International Archives of Photogrammetry and Remote Sensing*, 32(7-4-3W6), 1999.

[12] Oliver D. Faugeras. *Three-dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, Cambridge, Mass., USA, 1993.

[13] Oliver D. Faugeras and Quang-Tuan Luong. *The Geometry of Multiple Images*. MIT Press, 2001.

[14] Oliver D. Faugeras and Theo Papadopoulo. A nonlinear method for estimating the projective geometry of three views. In *Proceedings of the 6th International Conference on Computer Vision (ICCV '98)*, pages 477–484, Bombay, India, January 1998. IEEE Computer Society.

[15] Andrew W. Fitzgibbon, Maurizio Pilu, and Robert B. Fisher. Direct least square fitting of ellipses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):476–480, May 1999.

[16] Andrea Fusiello. Uncalibrated Euclidean reconstruction: a review. *Image and Vision Computing*, 18(6-7):555–563, May 2000.

[17] Walter Gander, Gene H. Golub, and Rolf Strebel. Least-squares fitting of circles and ellipses. *BIT*, 34(4):558–578, 1994.

[18] Neil Gershenfeld. *The Nature of Mathematical Modeling*. Cambridge University Press, 1999.

[19] Gene H. Golub and Charles F. van Loan. *Matrix Computations*. The Johns Hopkins University Press, 3rd edition, 1996.

[20] Radmin Halíř. Robust bias-corrected least squares fitting of ellipses. In *Proceedings of the 8th International Conference in Central Europe on Computer Graphics, Visualisation and Interactive Digital Media*, University of West Bohemia, Pilsen, Czech Republic, February 2000.

[21] Radmin Halíř and Jan Flusser. Numerically stable direct least suqares fitting of ellipses. In *Proceedings of the 6th International Conference in Central Europe on Computer Graphics and Visualisation*, pages 125–132, Pilsen, Czech Republic, 1998.

[22] Chris G. Harris and Michael Stephens. A combined corner and edge detector. In *Proceedings of the Fourth Alvey Vision Conference*, pages 141–151, 1988.

[23] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 1st edition, 2000.

[24] Richard I. Hartley. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):580–593, 1997.

[25] Richard I. Hartley and Peter Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157, 1997.

[26] The MathWorks Inc. *Optimisation toolbox: User's guide*. The MathWorks, 2000.

[27] Dan Kalman. A singularly valuable decomposition: The SVD of a matrix. *The College Mathematics Journal*, 27(10), January 1996.

[28] K. Kanatani and N. Ohta. Accuracy bounds and optimal computation of homography for image mosaicing applications. In *Proceedings of the 7th International Conference on Computer Vision (ICCV '99)*, pages 73–78, Kerkyra, Greece, September 1999. IEEE Computer Society.

[29] Kenichi Kanatani. *Statistical Optimisation for Geometric Computation: Theory and Practice*. Elsevier Science, 1996.

[30] Kenichi Kanatani. Model selection for geometric inference. In *Proceedings of the 5th Asian Conference on Computer Vision*, volume 1, pages 21–32, Melbourne, Australia, January 2002.

[31] Kenichi Kanatani and Naoya Ohta. Comparing optimal 3-D reconstruction for finite motion and optical flow. *Memoirs of the Faculty of Engineering, Okayama University*, 36(1):91–106, December 2001.

[32] Yasushi Kanazawa and Kenichi Kanatani. Do we really have to consider covariance matrices for image features? In *Proceedings of the 8th International Conference on Computer Vision (ICCV '01)*, pages 301–306, Vancouver, Canada, July 2001. IEEE Computer Society.

[33] Kay-Uwe Kasemir and Kalus Betzler. Detecting ellipses of limited eccentricity in images with high noise levels. *Image and Vision Computing*, 21(2):221–227, February 2003.

[34] C. T. Kelley. *Iterative Methods for Optimisation*. Society for Industrial & Applied Mathematics, 1999.

[35] Maurice G. Kendall, Alan Stuart, and J. Keith Ord. *Advanced Theory of Statistics*, volume 2. Edward Arnold, 5th edition, 1991.

[36] Peter Lancaster and Miron Tismenetsky. *The Theory of Matrices with Applications*. Academic Press, San Diego, 2 edition, 1985.

[37] Yoram Leedan and Peter Meer. Heteroscedastic regression in computer vision: Problems with bilinear constraint. *International Journal of Computer Vision*, 37(2):127–150, 2000.

[38] Yiwu Lei and Kok Cheong Wong. Ellipse detection based on symmetry. *Pattern Recognition Letters*, 20(1):41–47, January 1999.

[39] H. Christopher Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(10):133–135, September 1981.

[40] Quang-Tuan Luong, Rachid Deriche, Oliver D. Faugeras, and Theo Papadopoulo. On determining the fundamental matrix: Analysis of different methods and experimental results. Technical Report 1894, Institut National de Recherche en Informatique et en Automatique (INRIA), Sophia Antipolis, France, April 1993.

[41] Bogdan Matei. *Heteroscedastic Errors-in-Variables Models in Computer Vision*. PhD thesis, Department of Electrical and Computer Engineering, Rutgers University, 2001.

[42] Bogdan Matei and Peter Meer. A general method for errors-in-variables problems in computer vision. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR '00)*, volume 2, pages 18–25, South Carolina, USA, June 2000. IEEE Computer Society.

[43] Ken I.M. McKinnon. Convergence of the Nelder-Mead simplex method to a non-stationary point. *SIAM Journal on Optimisation*, 9:148–158, 1999.

[44] Matthias Mühlich and Rudolf Mester. A considerable improvement in non-iterative homography estimation using TLS and equilibration. *Pattern Recognition Letters*, 22(11):1181–1189, September 2001.

[45] John A. Nelder and Roger Mead. A simplex method for function minimisation. *Computer Journal*, 7:308–313, 1965.

[46] Garry N. Newsam and Nicholas J. Redding. Fitting the most likely curve through noisy data. In *Proceedings of the International Conference on Image Processing*, volume 2, pages 752–755, Washington, DC, USA, October 1997.

[47] Naoya Ohta. Image movement detection with reliability indices. *IEICE - Transactions on Communications*, E 74(10):3379–3388, October 1991.

[48] Naoya Ohta. Motion parameter estimation from optical flow without nuisance parameters. In *3rd International Workshop on Statistical and Computational Theories of Vision*, Nice, France, October 2003.

[49] Theodosios Pavlidis. Curve fitting with conic splines. *ACM Transactions on Graphics*, 2(1):1–31, January 1983.

[50] Daniel Pooley, Michael J. Brooks, Anton van den Hengel, and Wojciech Chojnacki. A voting scheme for estimating the synchrony of moving camera videos. In *Proceedings of the International Conference on Image Processing*, pages 413–416, Barcelona, Spain, September 2003.

[51] John Porrill. Fitting ellipses and predicting confidence envelopes using a bias corrected Kalman filter. *Image and Vision Computing*, 8(1):37–41, February 1990.

[52] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1993.

[53] Thomas Risse. Hough Transform for line recognition. *Computer Vision and Image Processing*, 46:327–345, 1989.

[54] Paul L. Rosin. A note on the least square fitting of ellipses. *Pattern Recognition Letters*, 14(10):799–808, 1993.

[55] Paul L. Rosin. Analysing error of fit functions for ellipses. *Pattern Recognition Letters*, 17(14):1461–1470, 1996.

[56] Paul D. Sampson. Fitting conics sections to 'very scattered' data: An iterative refinement of the Bookstein algorithm. *Computer Graphics and Image Processing*, 18:97–108, 1982.

[57] Charles V. Stewart. Robust parameter estimation in computer vision. *SIAM Review*, 41(3):513–537, 1999.

[58] Zhaohui Sun, Visvanathan Ramesh, and A. Murat Tekalp. Error characterization of the factorization method. *Computer Vision and Image Understanding*, 82:110–137, 2001.

[59] Gabriel Taubin. Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(11):1115–1138, 1991.

[60] Gabriel Taubin. An improved algorithm for algebraic curve and surface fitting. In *Proceedings of the 5th International Conference on Computer Vision (ICCV '93)*, pages 658–665, Berlin, Germany, May 1993. IEEE Computer Society.

[61] Philip H. S. Torr. An assessment of information criteria for motion model selection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR '97)*, pages 47–54, Puerto Rico, June 1997. IEEE Computer Society.

[62] Philip H. S. Torr. A structure and motion toolkit in Matlab. Technical Report MSR-TR-2002-56, Microsoft Research, Cambridge, UK, June 2002.

[63] Philip H. S. Torr and Andrew W. Fitzgibbon. Invariant fitting of two view geometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):648–650, May 2004.

[64] Bill Triggs, Philip McLauchlan, Richard Hartley, and Andrew Fitzgibbon. Bundle adjustment – a modern synthesis. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, Lecture Notes in Computer Science, pages 298–375. Springer Verlag, 2000.

[65] Sabine van Huffel and Joos Vandewalle. *The Total Least Squares Problem: Computational Aspects and Analysis*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1991.

[66] René Vidal and Shankar Sastry. Optimal segmentation of dynamic scenes from two perspective views. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR '03)*, pages 281–286, Wisconsin, USA, June 2003. IEEE Computer Society.

[67] Peng-Yeng Yin. A new circle/ellipse detector using genetic algorithms. *Pattern Recognition Letters*, 21(7):731–740, January 1999.

[68] Zhengyou Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *Image and Vision Computing*, 15(1):57–76, 1997.

[69] Zhengyou Zhang. On the optimisation criteria used in two-view motion analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(7):717–729, July 1998.

[70] Zhengyou Zhang, Rachid Deriche, Olivier D. Faugeras, and Quang-Tuan Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78:87–119, 1995.

[71] Zhengyou Zhang and Charles Loop. Estimating the fundamental matrix by transforming image points in projective space. *Computer Vision and Image Understanding*, 82(2):174–180, 2001.