Estimation methods in the errors-in-variables context

PhD dissertation

Levente Hunyadi

Budapest University of Technology and Economics
Department of Automation and Applied Informatics

Advisor:

Dr. István Vajk, DSc
professor, head of department
Budapest University of Technology and Economics
Department of Automation and Applied Informatics

Budapest, Hungary
2013

Summary

Constructing a computer model from a large set of data, typically contaminated with noise, is a central problem to fields such as computer vision, pattern recognition, data mining, system identification or time series analysis. In these areas our objective is often to capture the internal laws that govern a system with a succinct parametric representation. Despite the amount and high dimensionality of the data, the equation that relates data points is usually expressed in a compact manner. Unfortunately, nonlinearity in the system under study and the presence of noise means that conventional tools in statistics cannot be directly applied to estimate unknown system parameters.

The dissertation explores estimation methods focused on three related areas of errors-in-variables systems: fitting a nonlinear function to data where the fit is subject to constraints; fitting a union of several elementary nonlinear functions to a data set; and estimating the parameters of discrete-time dynamic systems.

Curve and surface fitting is a well-studied problem but the presence of noise and nonlinearity in the function that relates data points introduces bias and increases estimation variance, which is typically addressed with costly iterative methods. The thesis introduces non-iterative direct methods that fit data subject to constraints, with emphasis on fitting quadratic curves and surfaces, which are nevertheless close to estimates obtained by maximum likelihood methods.

Partitioning a data set into groups whose members are captured by the same relationship in an unsupervised manner is a common task in machine learning, referred to as clustering. While most approaches use a single point as a cluster representative, or cluster data into subspaces, less attention has been paid to nonlinear functions, or manifold clustering. The thesis applies constrained and unconstrained fitting and projection methods in the errors-in-variables context to construct an iterative and a non-iterative algorithm for manifold clustering, which incurs modest computational cost.

Identification of discrete-time dynamic systems is a well-understood problem but its errors-in-variables formulation, when both input and output is polluted by noise, introduces interesting challenges. Several papers discuss system identification of linear errors-in-variables systems but the estimation problem is more difficult in the nonlinear setting. The thesis combines the generalization of the Koopmans–Levin method, an approach to estimate parameters of a linear dynamic system with a scalable balance between accuracy and computational cost, with the nonlinear extension to the original Koopmans method, which gives a non-iterative approach to estimate parameters of a static system described by a polynomial function. The result is an effective system identification method for dynamic errors-in-variables systems with polynomial nonlinearities.

Összefoglaló

A számítógépes látás, mintafelismerés, adatbányászat, rendszeridentifikáció és idősorelemzés egy-egy központi feladata számítógépes modellt alkotni olyan nagyméretű adathalmazból, amely tipikusan zajjal terhelt. Ezeken a területeken a feladatunk gyakran az, hogy egy tömör paraméteres leírással ragadjuk meg azokat a belső törvényszerűségeket, amelyek az adott rendszert irányítják. Az adatok nagy mennyisége és magas dimenziószáma ellenére azonban az adatok közötti összefüggés többnyire tömör formában leírható. Ugyanakkor a vizsgált rendszerekben a nemlinearitás és a zaj jelenléte sajnos ahhoz vezet, hogy a hagyományos statisztikai megközelítések közvetlenül nem alkalmazhatók a rendszer ismeretlen paramétereinek becslésére.

Az értekezés három kapcsolódó területet tárgyal az errors-in-variables rendszerek témaköréből: nemlineáris függvények illesztése korlátozások mellett; elemi nemlineáris függvények sokaságának együttes illesztése; illetve diszkrét idejű dinamikus rendszerek paraméterbecslése.

A görbe- és felületillesztés jól ismert feladat, de a zaj és az illesztett függvényben lévő nemlinearitás jelenléte torzítást visz a becslésbe, és növeli a becslés szórását, amit tipikusan költséges iteratív módszerekkel küszöbölnek ki. Az értekezés olyan közvetlen nemiteratív módszereket mutat be, különös hangsúllyal másodfokú görbék és felületek illesztésére, amelyek korlátozások mellett végeznek görbe- és felületillesztést, mindazonáltal a maximum likelihood módszerekhez közeli eredményt adva.

A gépi tanulás területén egy gyakori feladat a klaszterezés, azaz egy adathalmazt ellenőrizetlen módon olyan csoportokra bontani, amelynek tagjait hasonló összefüggés kapcsolja össze. Amíg a legtöbb megközelítés egyetlen pontot használ egy klaszter reprezentatív elemeként, vagy éppen alterekre bontja a teljes adathalmazt, kevesebb figyelmet kaptak a klaszterezésben a nemlineáris függvények. Az értekezés korlátozásokkal és korlátozások nélküli illesztési és vetítési módszereket errors-in-variables környezetben alkalmazva egy iteratív és nemiteratív algoritmust mutat be nemlineáris klaszterezésre, mérsékelt számítási költség mellett.

A diszkrét idejű dinamikus rendszerek identifikációja klasszikus feladat, de errors-in-variables környezetben való megfogalmazása, amikor mind a bemenet, mind a kimenet zajjal terhelt, izgalmas kihívásokat rejt. Számos cikk tárgyalja lineáris errors-in-variables rendszerek identifikációját, de a becslési feladat nehezebb nemlineáris megfogalmazásban. Az értekezés az általánosított Koopmans–Levin módszer és az eredeti Koopmans módszer nemlineáris kiterjesztésének egy ötvözetét mutatja be; előbbi egy olyan megközelítés lineáris dinamikus rendszerek paraméterbecslésére, amely tetszőlegesen skálázható pontosság és számítási költség között, míg utóbbi egy nemiteratív megközelítés olyan statikus rendszerek paraméterbecslésére, amelyet polinomiális jellegű nemlinearitások írnak le. Az eredmény egy hatékony rendszeridentifikációs módszer polinomiális nemlinearitásokból álló dinamikus errors-in-variables rendszerek paraméterbecslésére.

Science begins with the world we have to live in, accepting its data and trying to explain its laws. From there, it moves toward the imagination: it becomes a mental construct, a model of a possible way of interpreting experience. The further it goes in this direction, the more it tends to speak the language of mathematics, which is really one of the languages of the imagination, along with literature and music.

Northrop Frye

Statement of authorship

I, Levente Hunyadi, confirm that this doctoral dissertation has been written solely by myself, and I have used only those sources that have been explicitly identified. Every part that has been adopted from external sources, either quoted or rephrased, has been unequivocally referenced, with proper attribution to the source.

Budapest, February 1, 2013

Alulírott Hunyadi Levente kijelentem, hogy ezt a doktori értekezést magam készítettem, és abban csak a megadott forrásokat használtam fel. Minden olyan részt, amelyet szó szerint, vagy azonos tartalomban, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Budapest, 2013. február 1.

Hunyadi Levente

Acknowledgments

While fully individual work, I am grateful to several people who have indirectly contributed to this thesis. First and foremost, fruitful discussions with István Vajk, my scientific advisor, highlighted important aspects of research that I might have easily overlooked, and helped steer work in a most productive direction. I would like to express my thanks to my colleagues at the Department of Automation and Applied Informatics for the friendly and inspiring atmosphere, and all interesting ideas our discussions have sparked. I am grateful for my colleagues at the Test Competence Center of Ericsson Research and Development Hungary for an open and supportive environment, which helped me harmonize all research and development projects I have been involved in.

This work has been supported by the fund of the Hungarian Academy of Sciences for control research. The results in this work are related to the project *Developing quality-oriented coordinated R+D+I strategy and operating model at BME*, which is supported by the *New Hungary Development Plan* (grant number TÁMOP-4.2.1/B-09/1/KMR-2010-0002). Partially, this work was supported by the European Union and the European Social Fund through the project FuturICT.hu (grant number TÁMOP-4.2.2.C-11/1/KONV-2012-0013).

Contents

Table of notations

The thesis uses the following general notation for scalars, vectors and matrices, and operations on scalars, vectors and matrices:

w	a scalar
$\mathbf{w}$	a column vector
$\mathbf{w}^\top$	a row vector
$\mathbf{W}$	a matrix
$\hat{\mathbf{w}}$	an estimate vector quantity
$\bar{\mathbf{w}}$	the unobservable true value of a vector quantity
$\tilde{\mathbf{w}}$	the noise component of a vector quantity
$f(x)$	a scalar function $f : \mathbb{R} \to \mathbb{R}$
$f(\mathbf{x})$	a scalar-valued function that takes a vector $f : \mathbb{R}^n \to \mathbb{R}$
$\mathbf{f}(\mathbf{x})$	a vector-valued function that takes a vector $f : \mathbb{R}^n \to \mathbb{R}^m$
$\partial_{\mathbf{x}} f(y)$	gradient of $f(y)$ w.r.t. the vector $\mathbf{x}$
$\mathbf{M}^\dagger$	Moore–Penrose pseudo-inverse
$\otimes$	Kronecker product
diag$\mathbf{w}$	a diagonal matrix of elements in $\mathbf{w}$
vec$\mathbf{W}$	vectorization of matrix $\mathbf{W}$ (columns of $\mathbf{W}$ stacked below each other)
$\mathbb{E}x$	expected value of a random variable x
σ_x^2	variance of a random variable x
$\sigma_{\mathbf{x}}^2$	vector of variances of each component in $\mathbf{x}$
$\mathbf{C}_{\sigma_{\mathbf{x}}^2}$	a diagonal covariance matrix of elements in $\sigma_{\mathbf{x}}^2$

Definitions are terminated with the symbol ■, proofs with □, and examples with ♣.

In addition to general notation, certain concepts, variables and indices have a typical common notation throughout the thesis:

$\mathbf{g}$	model (system) parameter vector (combined input and output parameters)
$\boldsymbol{\theta}$	transformed model (system) parameter vector
$\mathbf{b}$	input parameters (for dynamic systems)
$\mathbf{a}$	output parameters (for dynamic systems)
i	data index for static systems
k	data sequence index for dynamic systems (observation at time k)
r, s	component index
$\mathbf{x}_i$	vector of coordinates for data point i (for static systems)
$\mathbf{z}_i$	transformed vector of coordinates for data point i (for static systems)
u_k	input observation at time k (for dynamic systems)
y_k	output observation at time k (for dynamic systems)
$\mathbf{f}_{data}$	lifting function for the data vector, usually $\mathbf{f}_{data}(\mathbf{x}_i) = \mathbf{z}_i$
$\mathbf{f}_{par}$	lifting function for the model parameter vector, usually $\mathbf{f}_{par}(\mathbf{g}) = \boldsymbol{\theta}$
$\mathbf{D}$	covariance matrix built from sample data
$\mathbf{C}$	noise covariance matrix or noise covariance matrix polynomial

List of Figures

CHAPTER

1 Introduction

A recurring problem in engineering is constructing a computer model from a possibly large set of data. In fields such as computer vision, pattern recognition, image reconstruction, speech and audio processing, signal processing, modal and spectral analysis, data mining, system identification, econometrics or time series analysis, the goal is often to identify or describe the internal laws that govern a system rather than to predict its future behavior. In other words, one is interested in reconstructing how the measured variables are related given a set of observations. The amount and dimensionality of the observed data may be large yet one is often able to express the equation that relates data points in a succinct manner. In a system identification context, for instance, one could be interested in the parameters of a discrete-time dynamic system model, based on a measured input and output data sequence, contaminated with noise. In a pattern recognition context, on the other hand, one would seek to capture a set of unorganized data points with a number of simple shapes, such as lines, ellipses, parabolas, etc. While these tasks appear to be wildly different, they share some common properties, which outline the characteristics of *parametric errors-in-variables* systems:

- The relationships that characterize the system under investigation admit a structure. Even if the data accumulated may be large, the number of parameters is limited: the system can be explained by a set of relatively simple relationships, each known up to the parameters. A discrete-time dynamic system may be described by a low-order polynomial that relates the output variable with past values of the input and output variables, even if the data sequences may comprise of thousands of observations. Points obtained from a manufactured object by a laser scanner could be modeled with a couple of quadric surfaces.

- There are no distinguished variables in general. The equation that relates data points has the implicit form $f_{imp}(\mathbf{x}) = 0$ rather than an explicit form $y = f_{exp}(\mathbf{z})$, i.e. errors-in-variables systems have an inherent symmetry regarding the variables. For example, in the parametric description of an ellipse $Q(\mathbf{x}) = ax_1^2 + bx_2^2 - 1 = 0$ in canonical form, neither the component x^2, nor y^2 has special significance. Implicit functions can typ-

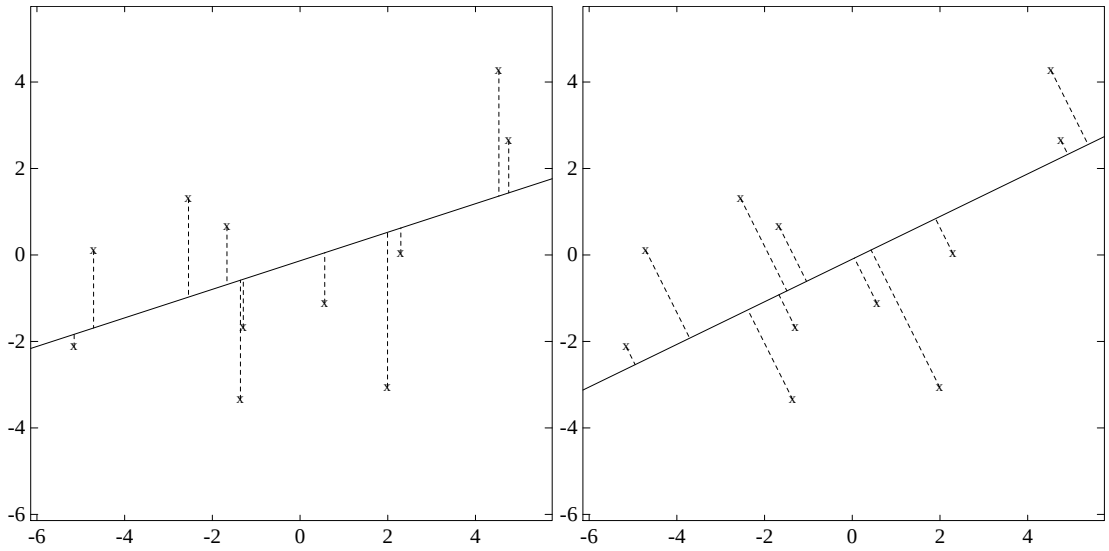


Figure 1.1: Visual comparison between the standard least-squares and the errors-in-variables approach.

ically capture a wider range of systems, and can often result in a more succinct representation than an equivalent explicit form $y = f_{exp}(\mathbf{z})$ where $\mathbf{x}^\top = \begin{bmatrix} y & \mathbf{z}^\top \end{bmatrix}$. However, techniques applicable to the explicit representation do not necessarily carry over to the implicit one.

- The underlying data space is usually homogeneous and isotropic with no inherent coordinate system. The estimation process should be invariant to changes of the coordinate system with respect to which the data are described. For instance, in a three-dimensional scan, there is no special significance of any of the coordinates x, y or z, data may be rotated and translated.

- Measured data is contaminated with noise. Unlike the usual assumption in statistics, there is in most cases no meaningful distinction between independent (noise-free) and dependent (noisy) variables. In the errors-in-variables context all variables are assumed to be measured quantities, hence contaminated with noise. Both the input and the output of the dynamic system is treated as a sequence of noisy measurements, and the points that are captured with a scanner are polluted with noise in all coordinates.

Identifying the underlying relationship that governs a system gives us a compact representation that is easier to manipulate and a key to understanding. For instance, a pattern recognition algorithm that exploits that an approximated 85% of manufactured objects can be modeled with quadratic surfaces [16], and subsequently reduces to fitting these surfaces, uses far fewer parameters than a primarily non-parametric approach that uses only locality information. Furthermore, the reverse-engineered model lends itself better to future transformations such as constructive solid geometry operations.

Figure 1.1 illustrates the difference between the standard least-squares approach in statistics and the errors-in-variables approach for linear data-fitting in two dimensions [20]. The

set of data at our disposal is identical in the two cases. The least-squares approach distinguishes an independent variable (x-axis) and a dependent variable (y-axis). Any measurement error is assumed to concentrate in the dependent variable, hence the optimal solution is to fit a line that minimizes the error w.r.t. the dependent variable. In contrast, the errors-in-variables approach is symmetric: there is no distinction between independent and dependent variables. As observations are treated as polluted with noise w.r.t. any variable, the best-fit line can be vastly different from that in the standard least squares case. Apparently, the errors-in-variables approach is more suited to a situation in which we try to understand how a system works using a set of measurements, none of which can be treated as completely accurate. Obviously, the least-squares approach is a special case of the errors-in-variables approach where some variables take no error.

Depending on the structure imposed on the approximated model, multiple different estimation problems can be outlined. Systems can be classified into *static* or *dynamic* systems whether observations are independent or coupled in time. The problem can be a pure *parameter estimation* problem where the solution is known up to a fixed number of free system parameters or a *non-parametric* problem where discovering a suitable model structure is part of the problem. In addition, parameter estimation problems can be further classified into *linear* or *nonlinear*, both in terms of data and parameters.

When the relationship is linear, standard tools in mathematics and statistics, such as singular value decomposition, can be applied to static systems, and iterative algorithms may be formulated for dynamic systems. Nonlinearity in the system, however, makes it difficult to draw conclusions for the original (noise-free) relationship based on (noisy) measured variables; traditional approaches may lead to substantial bias as additive noise contaminates nonlinear variables of the system. The objective of the thesis is to extend the results for linear errors-in-variables systems to a relatively rich set of the nonlinear case when the system is captured by polynomial functions. This leads us to the following open questions:

- How can we extend the principles of methods developed with the usual statistical assumption to the errors-in-variables context? Moving from the assumption of problem separation into independent (known accurately) and dependent (measured with error) variables to the (inseparable) errors-in-variables domain, a large number of additional unknown quantities are introduced into the model, often repositioning the problem in a more difficult context (e.g. multiple equivalent solutions and necessary normalization).

- How can we generalize existing methods for linear systems in the errors-in-variables framework to nonlinear systems? Linear systems usually reduce to a non-iterative solution but nonlinear systems typically demand an iterative approach where convergence and the choice of initial values are important issues.

- How can the errors-in-variables principles help us improve estimation accuracy while imposing only a moderate computational cost?

- How can existing errors-in-variables methods be improved to apply to a wider range of systems? Many existing methods have a limited applicability to general conditions, or demand excessively large samples or obtain reliable results.

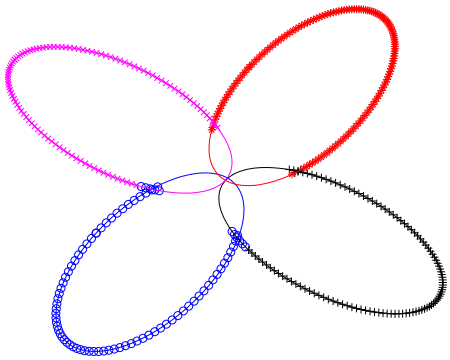


Figure 1.2: Identifying the decomposition of a complex curve.

- How can we incorporate estimation methods for nonlinear parametric errors-in-variables systems in machine learning applications such as model construction from point clouds? Unlike parameter estimation problems where the structure of the problem is a priori known, reconstruction problems must discover a feasible partitioning of the data set into groups that exhibits a specific parametric relationship as well as estimate the unknown parameters for data within a group.

The thesis investigates algorithms related to identification and recognition problems that belong to the domain of nonlinear parametric errors-in-variables systems. We shall look into both static and dynamic (time-dependent) systems, with focus on low-order polynomial functions, and assume the noise that contaminates measurements is Gaussian. Contributions address both the pure parametric case, where the system is known up to the unknown parameters, as well as the machine learning case, where we assume the system is known to comprise of a set of constituents, each captured with a well-known structure defined by a set of unknown parameters.

First, we venture onto the field of static systems where the estimation may be subject to constraints. In a computer vision application, for instance, one may be interested in fitting an ellipse rather than a general quadratic curve to an unorganized point set. Previous work in the computer vision domain showed how to integrate constraints such as the quadratic curve representing an ellipse (rather than e.g. a hyperbola) into the estimation but failed to adequately take into account the nonlinear distortions induced by noise. The method proposed in the thesis takes a step further, and incorporates constraints into parameter estimation while canceling the effect of noise.

Second, we investigate a machine learning or reverse engineering application, namely clustering, where the system is originally built up of several constituents, each described by a polynomial relationship, more restrictively quadratic curves and surfaces, but only an unorganized set of noisy data is at our disposal, and we aim to discover the original structure of the system and estimate parameters of the constituents. Unlike standard parameter estimation methods where the model to reconstruct is structured, i.e. is known up to a few free parameters to estimate, the problem is more challenging when structure discovery is part of the problem, i.e. a natural decomposition of the entity under study exists but is not

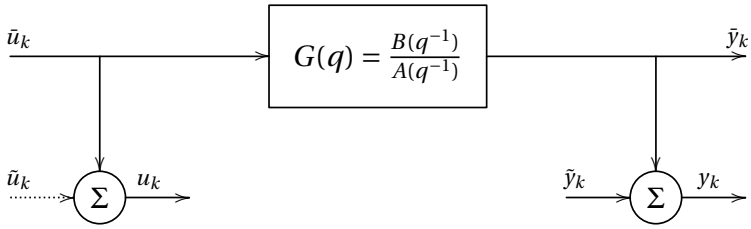


Figure 1.3: A linear dynamic discrete-time errors-in-variables system.

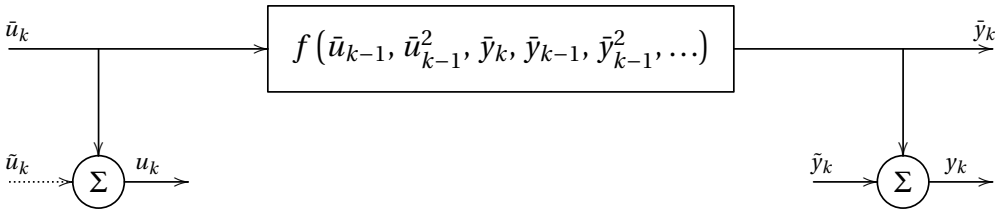


Figure 1.4: An example of a polynomial dynamic discrete-time errors-in-variables system.

known to us. Several algorithms exist that can tackle the so-called (multi-)subspace clustering or hybrid linear modeling problem, where the objective is to build a model where each partition of the data is captured by a linear relationship and the entire model is a composition of these partitions. Not every data set, however, admits a decomposition into linear relationships, and applying linear methods to essentially nonlinear relationships loses the simplicity and explanation power of these methods. A natural generalization of subspace clustering is (multi-)manifold clustering, which we explore in this thesis, where each curved manifold is captured by some nonlinear (or more specifically, polynomial) relationship. Figure 1.2 shows such a setup: a combination of points along four ellipses make up a complex curve. Here, the complex curve comprises of four independent quadratic curves, which the complex curve can be seen as a union of.

Finally, we explore dynamic systems, where we consider discrete-time but nonlinear systems that can be re-cast in a linear setting using a lifting function. Figure 1.3 shows how a linear errors-in-variables system compares to a classical dynamic system setup in system identification. In the classical case, only system output is measured with noise, system input can be observed noise-free (i.e. no noise contribution by the dashed line), which is a well-understood problem. The case is more subtle when both the system input and output is contaminated with noise but many algorithms exist for the case when the ratio of input and output noise is at our disposal. The thesis takes a step further and deals with dynamic systems where the input and output are no longer related in a linear manner but captured by a polynomial function. Such a system model is shown in Figure 1.4. The thesis combines a generalization of the Koopmans–Levin method [65] with a nonlinear extension to the original Koopmans method [66]. The generalized Koopmans–Levin method is an approach to estimate parameters of a linear dynamic system with a scalable balance between accuracy

and computational cost, whereas the nonlinear extension to the Koopmans method gives a non-iterative approach to estimate parameters of a static system described by a polynomial function. The algorithm proposed in the thesis alloys the two approaches, and estimates parameters of dynamic systems whose variables are related with a polynomial function from a set of data polluted with Gaussian noise.

A working implementation with several examples is essential to the popularization of any new methods. The results in this thesis are augmented with source code, which demonstrate the feasibility of the proposed algorithms.

CHAPTER

2 Parameter estimation over point clouds

Fitting a model to measured data that captures the underlying relationship, in particular, fitting (a certain a class of) quadratic curves and surfaces (e.g. ellipses and ellipsoids) occurs frequently in computer vision, computer-aided design, pattern recognition and image processing applications. Unlike the large mass of scattered data acquired with some measuring device (e.g. a laser scanner), the curves and surfaces fitted to data can be represented with only a few parameters, which lends itself to compact storage and easier manipulation. In particular, low-order implicit curves and surfaces are a practical choice in grasping the relationship since they are closed under several geometric operations (e.g. intersection, union, offset) while they offer a higher degree of smoothness than their counterparts with the same number of variables but cast in an explicit form, and may be preferred especially if the object under study itself is a composition of geometric shapes. It has been reported that 85% of manufactured objects can be modeled with quadratic surfaces [16], which highlights the significance of methods that can fit such surfaces to measured data.

The general estimation problem we face can be formalized using the notation

$$f(\bar{\mathbf{x}}_i) = 0$$

where $\bar{\mathbf{x}}_i$ is a vector of noise-free data, $\mathbf{x}_i = \bar{\mathbf{x}}_i + \tilde{\mathbf{x}}_i$ is a data vector $i = 1, \ldots, N$ with N being the number of data points, $\tilde{\mathbf{x}}_i$ is a noise contribution, and we seek to capture the relationship f, often referred to as a level-set function (or in this particular case, a zero-level set). However, if structural information is available as to the relationship f of the point set, the estimates can be improved by incorporating that information in the estimation procedure. Parametric estimation methods take this approach and come down to choosing an appropriate model and finding values for the model parameters. A general nonlinear parametric system takes the form

$$f(\bar{\mathbf{x}}_i, \mathbf{g}) = 0$$

where $\bar{\mathbf{x}}_i$ is a vector of noise-free data, $\mathbf{g}$ is a vector of model parameters (e.g. curve or surface parameters), and f is a nonlinear function that relates data and parameters. From the formulation, it follows that the system is assumed structured, i.e. it is known up to (a few) model parameters. Our goal is to estimate $\mathbf{g}$ from noisy samples $\mathbf{x}_i$.

The field of parametric estimation methods targeting nonlinear systems is rather broad. Some approaches that do not aim at discovering the internal structure in the data employ techniques that allow sufficient flexibility of an approximating function $\hat{f}$ to adapt to local features (using many components in $\mathbf{g}$). A prominent example is spline-like approaches whereby a continuous curve with given order and possibly knot sequence is fit to the data points (gradually pulling the function $\hat{f}$ towards data points), minimizing an error measure and the associated complexity of the curve. Even while these approaches are suitable for approximating the data but are of little help explaining the data: the spline itself does not facilitate understanding the underlying structure. Thus, we shall explore parameter estimation methods where low-order polynomial functions such as quadratic curves (for 2D) and surfaces (for 3D) capture the relationship between data points.

Thus, it is natural to restrict our investigation to a narrower scope; we will discuss nonlinear systems that assume a polynomial form in terms of data, which are captured by the implicit equation

$$\mathbf{f}_{data}(\bar{\mathbf{x}}_i)^\top \mathbf{f}_{par}(\mathbf{g}) = 0$$

where $\mathbf{x}_i = \bar{\mathbf{x}}_i + \tilde{\mathbf{x}}_i$ is a data vector $i = 1, \ldots, N$ with N being the number of data points, $\tilde{\mathbf{x}}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_{\boldsymbol{\sigma}^2_{\mathbf{x}}})$ is a noise contribution, $\mathbf{g}$ is a parameter vector. The (polynomial) function $\mathbf{f}_{data} : \mathbb{R}^n \to \mathbb{R}^m$ is the lifting function for data, mapping a data vector $\bar{\mathbf{x}}_i \in \mathbb{R}^n$ into m-dimensional space, and the (polynomial) function $\mathbf{f}_{par} : \mathbb{R}^p \to \mathbb{R}^m$ is the lifting function for model parameters, mapping a parameter vector $\mathbf{g} \in \mathbb{R}^p$ into m-dimensional space. The matrix $\mathbf{C}_{\boldsymbol{\sigma}^2_{\mathbf{x}}} = \operatorname{diag}\left(\boldsymbol{\sigma}^2_{\mathbf{x}}\right)$ is the noise covariance matrix for the homoskedastic normal (Gaussian) noise $\mathcal{N}$ (i.e. noise has the same variance for all data points) where $\boldsymbol{\sigma}^2_{\mathbf{x}}$ is a vector of variances for each component of an $\tilde{\mathbf{x}}_i$. $\boldsymbol{\sigma}^2_{\mathbf{x}}$ is assumed to be known up to scale, i.e. $\boldsymbol{\sigma}^2_{\mathbf{x}} = \mu\bar{\boldsymbol{\sigma}}^2_{\mathbf{x}}$ where the vector $\bar{\boldsymbol{\sigma}}^2_{\mathbf{x}}$ is known but the scalar μ is unknown. The noise covariance matrix $\mathbf{C}_{\boldsymbol{\sigma}^2_{\mathbf{x}}}$ is (in general) of full rank: there is no distinguished variable that we can observe noise-free, which is what we call the errors-in-variables approach.

Example 1. Observations of a static system transformed by a lifting function. Suppose we have a set of noisy data in two dimensions

$$\mathbf{x}_i = \begin{bmatrix} x_i & y_i & 1 \end{bmatrix}^\top$$

and our goal is to fit an ellipse minimizing algebraic distance

$$e = \frac{1}{N}\sum_{i=1}^{N} \left(f(\mathbf{x}_i, \mathbf{g})\right)^2.$$

The parameters of the ellipse are

$$\mathbf{g} = \begin{bmatrix} a & b & c & p & q & d \end{bmatrix}^\top$$

where we ideally have

$$ax^2 + bxy + cy^2 + px + qy + d = 0 \qquad \text{s.t. } b^2 < ac$$

where the constraint ensures the estimator always produces an ellipse. Our lifting function for data is then

$$\mathbf{f}_{data}(\mathbf{x}_i) = \begin{bmatrix} x_i^2 & x_i y_i & y_i^2 & x_i & y_i & 1 \end{bmatrix}^\top$$

and the lifting function of parameters is the identity mapping

$$\mathbf{f}_{par}(\mathbf{g}) = \mathbf{g}.$$

♣

The rest of the chapter is structured as follows. Section 2.1 introduces the unconstrained fitting problem in which the objective is either to minimize a geometric or an algebraic distance, but without any ancillary constraints. Sections 2.1.1 and 2.1.2 cover maximum likelihood methods, which minimize Euclidean distance of the scattered points to the curve or surface being estimated, whereas Sections 2.1.3, 2.1.4 and 2.1.5 survey related work that are based on simpler and computationally less expensive approaches, which reduce to solving a regular or a generalized eigenvalue problem. Section 2.1.6 discusses the nonlinear Koopmans method, whose principles are integral to the noise cancellation step part of the constrained fitting algorithm proposed in this thesis, with numerical improvements that further reduce computational cost. Section 2.2 concentrates on fitting quadratic curves and surfaces subject to constraints, with a two-stage algorithm comprising of a noise cancellation step and a constrained fitting step, which is one of the main results in this thesis. Section 2.2.1 uses a known numerical technique to reduce matrix dimensions to the size of the nonzero part of the constraint matrix, whereas Sections 2.2.2, 2.2.3 and 2.2.4 describe algorithms for fitting various classes of quadratic curves and surfaces. While the algorithms themselves are known results, the context in which they are used, namely, operating on a noise-compensated matrix, is a new contribution of this thesis, and formulated as a single algorithm in Section 2.2.5. The new algorithm in Section 2.2.5 is also a key ingredient to another major result in this thesis covered in Chapter 3, a clustering method fitting at most quadratic curves and surfaces, where an inexpensive yet effective estimation method is indispensable.

2.1 Unconstrained fitting

An important class of fitting problems falls into the category of unconstrained fitting where no constraints are imposed other than the implicit equation

$$\mathbf{f}_{par}\left(\mathbf{g}\right)\mathbf{f}_{data}\left(\bar{\mathbf{x}}_i\right) = \boldsymbol{\theta}^\top \bar{\mathbf{z}}_i = 0.$$

Here, we seek to find the best $\mathbf{g}$ that minimizes the fitting error using noisy samples $\mathbf{x}_i = \bar{\mathbf{x}}_i + \tilde{\mathbf{x}}_i$ where $\bar{\mathbf{x}}_i$ is noise-free data and $\tilde{\mathbf{x}}_i$ is a noise contribution, and $\bar{\mathbf{z}}_i = \mathbf{f}_{data}\left(\bar{\mathbf{x}}_i\right)$. For simplicity, we assume that $\mathbf{f}_{par}\left(\mathbf{g}\right)$ is an identity transformation and thus $\boldsymbol{\theta} = \mathbf{g}$.

One possible way to estimate $\mathbf{g}$ from $\mathbf{x}_i$ is to use geometric distance, and employ maximum likelihood methods, and minimize

$$e = \frac{1}{N}\sum_{i=1}^{N} d_i^2$$

where d_i measures the distance from the noisy point $\mathbf{x}_i$ to the curve or surface $f(\mathbf{x}, \mathbf{g}) = 0$. The distance might either be Euclidean distance if the noise covariance matrix for $\tilde{\mathbf{x}}_i$ is

$\mathbf{C}_{\boldsymbol{\sigma}^2_{\mathbf{x}}} = \sigma^2\mathbf{I}$ (i.e. all vector components are contaminated with equal amount of noise) or in general Mahalanobis distance (a scale-invariant distance measure that takes into account correlations in $\mathbf{C}_{\boldsymbol{\sigma}^2_{\mathbf{x}}}$).

Approaches that aim at minimizing Euclidean or Mahalanobis distance are *geometric fitting* methods, and include maximum likelihood, approximate likelihood [49, 18] or renormalization methods [36]. While highly accurate, they lead to an iterative formulation that may converge slowly (or even diverge) in some situations, and always requires a feasible initialization. More substantial levels of noise may impact convergence and have an adverse effect on estimation accuracy.

A more robust approach than geometric fitting is to use algebraic distance

$$e = \frac{1}{N}\sum_{i=1}^{N}\big(f(\mathbf{x}_i, \mathbf{g})\big)^2$$

where noisy points are substituted into the curve or surface equation. The *algebraic fit* minimizes this substitution error, which yields a fast, non-iterative approach that is typically less accurate than geometric fit. With proper compensation for the error induced by noise, however, it is possible to reduce adverse effects, and construct estimation methods, such as the nonlinear Koopmans method [66] or consistent algebraic least squares [42, 48], that achieve accuracy similar to maximum likelihood estimation, at much lower computational cost.

For the purposes of parameter estimation, we assume that the data has zero mean and has been normalized to its root mean square (RMS), which is meant to reduce numerical errors [17]. For each dimension x of the data set, this means

$$\begin{aligned} m_x &= \frac{1}{N}\sum_{i=1}^{N} x_i \\ x_i &\leftarrow \frac{x_i - m_x}{s} \\ \sigma_x &\leftarrow \frac{\sigma_x}{s} \end{aligned}$$

and likewise for all other dimensions y, z, etc. where

$$s = \sqrt{\frac{1}{2n}\sum_{i=1}^{n}\Big\{(x_i - m_x)^2 + \big(y_i - m_y\big)^2\Big\}}$$

for two dimensions and

$$s = \sqrt{\frac{1}{3n}\sum_{i=1}^{n}\Big\{(x_i - m_x)^2 + \big(y_i - m_y\big)^2 + (z_i - m_z)^2\Big\}}$$

for three dimensions. Reducing data spread also reduces the additive noise on components of $\mathbf{x}_i$ and the noise covariance matrix has to be updated accordingly. Translation and scaling also dilates model parameters $\mathbf{g}$. For instance, when estimating quadratic curves in two dimensions with the lifting function

$$\mathbf{f}_{data}(\mathbf{x}_i) = \begin{bmatrix} x_i^2 & x_i y_i & y_i^2 & x_i & y_i & 1 \end{bmatrix}^\top$$

parameters

$$\mathbf{g}^\top = \begin{bmatrix} g_1 & g_2 & g_3 & g_4 & g_5 & g_6 \end{bmatrix}$$

in the original space are recovered as follows:

$$\begin{aligned}
g_1 &\leftarrow g_1 s^2 \\
g_2 &\leftarrow g_2 s^2 \\
g_3 &\leftarrow g_3 s^2 \\
g_4 &\leftarrow -2g_1 s^2 m_x - g_2 s^2 m_y + g_4 s^3 \\
g_5 &\leftarrow -g_2 s^2 m_x - 2g_3 s^2 m_y + g_5 s^3 \\
g_6 &\leftarrow g_1 s^2 m_x^2 + g_2 s^2 m_x m_y + g_3 s^2 m_y^2 - g_4 s^3 m_x - g_5 s^3 m_y + g_6 s^4.
\end{aligned}$$

2.1.1 Maximum likelihood estimation

Estimating parameters of a linear system where all data points are related by the same function but polluted by Gaussian noise with a known structure is the well-understood and widely-used method of total least-squares fitting [35], solved as either an eigenvalue problem or a computationally more robust singular value problem, which yields maximum likelihood estimates. Without a priori information (in addition to preliminary structural information), maximum likelihood methods deliver the best possible estimates for the model parameters based on measured data of the system under investigation. In the maximum likelihood context, we seek to maximize the (joint) probability (compound probability density function)

$$p(\mathbf{x} \,|\, \mathbf{g}, \bar{\mathbf{x}}) = \prod_{i=1}^{N} p(\mathbf{x}_i \,|\, \mathbf{g}, \bar{\mathbf{x}}_i)$$

over the parameter vector $\mathbf{g}$ where $\mathbf{x}$ is a vector of all (observed) data points $\mathbf{x}_i$, $\bar{\mathbf{x}}$ is a vector of all (unknown) noise-free data points $\bar{\mathbf{x}}_i$, and

$$p(\mathbf{x}_i \,|\, \mathbf{g}, \bar{\mathbf{x}}_i) = \frac{1}{\sqrt{(2\pi)^{\dim\left(\mathbf{C}_{\sigma_\mathbf{x}^2}\right)} \det\left(\mathbf{C}_{\sigma_\mathbf{x}^2}\right)}} \exp\left(-\frac{1}{2}(\mathbf{x}_i - \bar{\mathbf{x}}_i)^\top \mathbf{C}_{\sigma_\mathbf{x}^2}^{-1}(\mathbf{x}_i - \bar{\mathbf{x}}_i)\right) \tag{2.1}$$

in which the noise covariance matrix $\mathbf{C}_{\sigma_\mathbf{x}^2}$ (available up to scale) is at our disposal, and we have assumed that the noise contributions over data point $\mathbf{x}_i$ are independent and identically distributed. ($\dim \mathbf{C}_{\sigma_\mathbf{x}^2}$ equals the number of components in $\mathbf{x}_i$ and $\bar{\mathbf{x}}_i$.) While tackled relatively easily for the linear case, the nonlinear case poses difficulty with the exploding number of unknowns.

Removing the leading constant term in (2.1) we get

$$p(\mathbf{x}_i \,|\, \mathbf{g}, \bar{\mathbf{x}}_i) \propto \exp\left\{-\frac{1}{2}(\mathbf{x}_i - \bar{\mathbf{x}}_i)^\top \mathbf{C}_{\sigma_\mathbf{x}^2}^{-1}(\mathbf{x}_i - \bar{\mathbf{x}}_i)\right\}$$

leading to the log likelihood cost function that takes the form

$$J = \sum_{i=1}^{N} (\mathbf{x}_i - \bar{\mathbf{x}}_i)^\top \mathbf{C}_{\sigma_\mathbf{x}^2}^{-1}(\mathbf{x}_i - \bar{\mathbf{x}}_i) \quad \text{s.t.}\, \mathbf{g}^\top \mathbf{f}_{data}(\bar{\mathbf{x}}_i) = 0 \tag{2.2}$$

for each $i = 1, \ldots, N$ where $\mathbf{f}_{data}(\bar{\mathbf{x}}_i)$ is a lifting of $\bar{\mathbf{x}}_i$. Minimizing the constrained function (2.2) (where the constraint enforces the model) is equivalent according to the method of Lagrange multipliers to minimizing the unconstrained function

$$\begin{aligned} J &= \sum_{i=1}^{N} (\mathbf{x}_i - \bar{\mathbf{x}}_i)^\top \mathbf{C}_{\sigma_{\mathbf{x}}^2}^{-1} (\mathbf{x}_i - \bar{\mathbf{x}}_i) \\ &+ \sum_{i=1}^{N} \eta_i \mathbf{g}^\top \mathbf{f}_{data}(\bar{\mathbf{x}}_i) \end{aligned} \tag{2.3}$$

When the lifting function $\mathbf{f}_{data}$ is linear, the constraints can be substituted directly into the constrained equation (2.2) and produce a simpler form. However, this is unfortunately not possible in the nonlinear case and minimizing (2.3) w.r.t. all variables is burdensome. Iterative methods, however, may offer a means to tackle the problem.

On the other hand, the objective function

$$J = \sum_{i=1}^{N} (\mathbf{x}_i - \bar{\mathbf{x}}_i)^\top \mathbf{C}_{\sigma_{\mathbf{x}}^2}^{-1} (\mathbf{x}_i - \bar{\mathbf{x}}_i)$$

can be interpreted as

$$J = \sum_{i=1}^{N} d^2\left(\mathbf{x}_i - \bar{\mathbf{x}}_i, \mathbf{C}_{\sigma_{\mathbf{x}}^2}\right) = \sum_{i=1}^{N} d_{\mathbf{C}}^2 (\mathbf{x}_i - \bar{\mathbf{x}}_i) \tag{2.4}$$

where

$$d(\mathbf{a} - \mathbf{b}, \mathbf{C}) = \sqrt{(\mathbf{a} - \mathbf{b})^\top \mathbf{C}^{-1} (\mathbf{a} - \mathbf{b})}$$

is the so-called Mahalanobis distance between data vectors $\mathbf{a}$ and $\mathbf{b}$ (or Euclidean distance if $\mathbf{C} = \mathbf{I}$). In the particular case, this means that the distance between observed data points $\mathbf{x}_i$ and the (unknown) true data points $\bar{\mathbf{x}}_i$ is to be minimized, given the constraint $\mathbf{g}^\top \mathbf{f}_{data}(\bar{\mathbf{x}}_i) = 0$.

This formulation allows us to set up an iterative procedure whereby the curve or surface to be found is pulled towards the observed data points, minimizing the distance metric (Figure 2.1). In fact, we seek the minimum of (2.4), which is a nonlinear least-squares optimization problem, since the distance depends on the parameters $\mathbf{g}$ in a nonlinear manner. If we compute

$$\frac{\partial}{\partial \mathbf{g}} d_{\mathbf{C}} (\mathbf{x}_i - \bar{\mathbf{x}}_i)$$

we may use gradient-based minimization (e.g. Levenberg–Marquardt algorithm) to find the local minimum. If seeded with an estimate for $\mathbf{g}$ that is close enough to the solution, the value of $\mathbf{g}$ is readily found.

Unfortunately, values of $\bar{\mathbf{x}}_i$ are not at our disposal but given a curve or surface defined with the (current) parameters $\mathbf{g}$, observed data points $\mathbf{x}_i$ can be projected to the curve or surface to obtain so-called foot points, i.e. points that satisfy the constraint $\mathbf{g}^\top \mathbf{f}_{data}(\bar{\mathbf{x}}_i) = 0$, in each step. The most natural choice for the projection is to minimize geometric distance, i.e.

$$\bar{\mathbf{x}}_i = \arg\min_{\mathbf{x}} d_{\mathbf{C}}\left(\mathbf{x}_i - \mathbf{x}(\mathbf{g})\right).$$

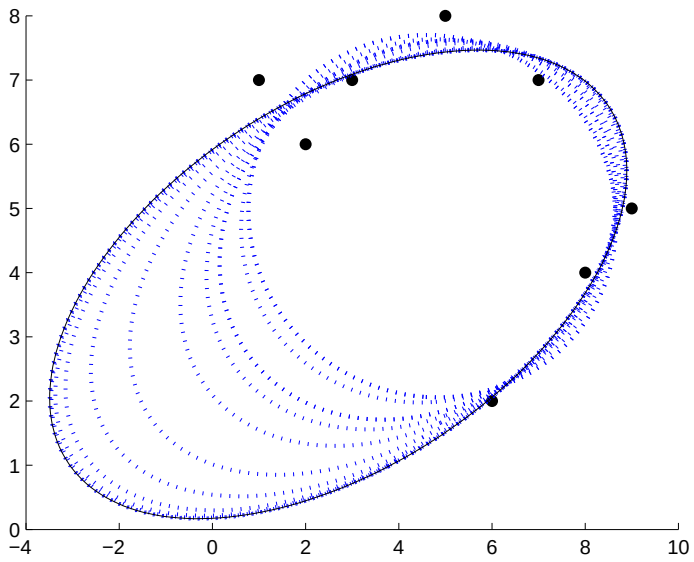


Figure 2.1: Iterative maximum likelihood estimation. The initial curve that gives a rough parameter estimate is gradually attracted towards the final solution (continuous line).

The expression $\mathbf{x}(\mathbf{g})$ indicates that $\mathbf{x}$ must satisfy $\mathbf{g}^\top \mathbf{f}_{data}(\mathbf{x}) = 0$ where $\mathbf{g}$ and $\mathbf{f}_{data}$ are given. This way, we can compute the objective function value J and its gradient $\frac{\partial}{\partial \mathbf{g}} J$ in each step. The entire algorithm can be summarized as follows:

$$\mathbf{g} = \arg\min_{\mathbf{g}} J(\mathbf{g}) = \arg\min_{\mathbf{g}} \sum_{i=1}^{N} d_{\mathbf{C}}\left(\mathbf{x}_i - \arg\min_{\mathbf{x}} d_{\mathbf{C}}\left(\mathbf{x}_i - \mathbf{x}(\mathbf{g})\right)\right).$$

There are two important implications of this approach:

1. Good-enough initial estimates for $\mathbf{g}$ that are close to the solution are crucial such that the local minimum obtained in the process is also a global minimum.

2. Fast projection algorithms are needed such that

$$\bar{\mathbf{x}}_i = \arg\min_{\mathbf{x}} d_{\mathbf{C}}\left(\mathbf{x}_i - \mathbf{x}(\mathbf{g})\right)$$

   can be obtained in an economical way.

The problem of initialization is tackled with relatively highly accurate but non-iterative methods such as Taubin's method (see [61] and Section 2.1.4) or consistent algebraic least squares (see [66, 42, 48] and Section 2.1.6). Projection, in general, requires solving polynomial equations but fast algorithms exist for the special case of projecting to at most quadratic curves and surfaces (covered in more depth in Section 3.3). This highlights that despite the accuracy of the maximum likelihood method, non-iterative low-cost methods with similar accuracy are much desired.

2.1.2 Approximated maximum likelihood estimation

The approximated maximum likelihood (AML) estimation method [18] is one of the various computational schemes that have been proposed for estimating curve or surface parameters, with accuracy close to those obtained by maximum likelihood methods. As its name suggests, the method does not minimize the unconstrained maximum log-likelihood function (2.3) but a simplification of it. First, the likelihood function is formulated in an alternative way, then functions of unobservable noise-free variables $\bar{\mathbf{x}}_i$ are substituted with their approximations obtained from available noisy data $\mathbf{x}_i$. Finally, an iterative scheme is employed to yield parameter estimates in a few iterations. Even while the approximated and the true maximum log-likelihood function are seldom minimized at the same objective function variable value, the estimates are close in practice.

The key to simplifying the objective function (2.3) of maximum likelihood estimation is its alternative form, to be developed based on [18].

The method of Lagrange multipliers implies that the gradient (column vector of partial derivatives) of $(\mathbf{x}_i-\mathbf{w})^\top \mathbf{C}_{\sigma_\mathbf{x}^2}^{-1}(\mathbf{x}_i-\mathbf{w})$ w.r.t. $\mathbf{w}$ is proportional to the gradient of $\mathbf{g}^\top \mathbf{f}_{data}(\mathbf{w})$ provided that both these gradients are evaluated at $\bar{\mathbf{x}}_i$. Comparing the gradients, it follows that

$$\mathbf{C}_{\sigma_\mathbf{x}^2}^{-1}(\mathbf{x}_i-\bar{\mathbf{x}}_i) = \lambda_i \mathbf{g}^\top \partial_\mathbf{x}\mathbf{f}_{data}(\bar{\mathbf{x}}_i) \tag{2.5}$$

for some scalar λ_i. Multiplying both sides of the equation by $\mathbf{C}_{\sigma_\mathbf{x}^2}$ and $\mathbf{g}^\top \partial_\mathbf{x}\mathbf{f}_{data}(\bar{\mathbf{x}}_i)$, we find

$$\mathbf{g}^\top \partial_\mathbf{x}\mathbf{f}_{data}(\bar{\mathbf{x}}_i)(\mathbf{x}_i-\bar{\mathbf{x}}_i) = \lambda_i \mathbf{g}^\top \left(\partial_\mathbf{x}\mathbf{f}_{data}(\bar{\mathbf{x}}_i)\mathbf{C}_{\sigma_\mathbf{x}^2}\partial_\mathbf{x}\mathbf{f}_{data}^\top(\bar{\mathbf{x}}_i)\right)\mathbf{g}$$

and hence

$$\lambda_i = \frac{\mathbf{g}^\top \partial_\mathbf{x}\mathbf{f}_{data}(\bar{\mathbf{x}}_i)(\mathbf{x}_i-\bar{\mathbf{x}}_i)}{\mathbf{g}^\top \partial_\mathbf{x}\mathbf{f}_{data}(\bar{\mathbf{x}}_i)\mathbf{C}_{\sigma_\mathbf{x}^2}\partial_\mathbf{x}\mathbf{f}_{data}^\top(\bar{\mathbf{x}}_i)\mathbf{g}} \tag{2.6}$$

On the other hand, multiplying both sides of (2.5) by $(\mathbf{x}_i-\bar{\mathbf{x}}_i)^\top$, we obtain

$$(\mathbf{x}_i-\bar{\mathbf{x}}_i)^\top \mathbf{C}_{\sigma_\mathbf{x}^2}^{-1}(\mathbf{x}_i-\bar{\mathbf{x}}_i) = \lambda_i (\mathbf{x}_i-\bar{\mathbf{x}}_i)^\top \partial_\mathbf{x}\mathbf{f}_{data}^\top(\bar{\mathbf{x}}_i)\mathbf{g}$$

Substituting the value of λ_i from (2.6) into this equation, we may conclude that

$$(\mathbf{x}_i-\bar{\mathbf{x}}_i)^\top \mathbf{C}_{\sigma_\mathbf{x}^2}^{-1}(\mathbf{x}_i-\bar{\mathbf{x}}_i) = \frac{\mathbf{g}^\top \partial_\mathbf{x}\mathbf{f}_{data}(\bar{\mathbf{x}}_i)(\mathbf{x}_i-\bar{\mathbf{x}}_i)(\mathbf{x}_i-\bar{\mathbf{x}}_i)^\top \partial_\mathbf{x}\mathbf{f}_{data}^\top(\bar{\mathbf{x}}_i)\mathbf{g}}{\mathbf{g}^\top \partial_\mathbf{x}\mathbf{f}_{data}(\bar{\mathbf{x}}_i)\mathbf{C}_{\sigma_\mathbf{x}^2}\partial_\mathbf{x}\mathbf{f}_{data}^\top(\bar{\mathbf{x}}_i)\mathbf{g}} \tag{2.7}$$

Consider the Taylor expansion of the γth component of $\mathbf{f}_{data}(\mathbf{x})$ about any particular $\mathbf{w}$ (the observation index i is omitted from $\mathbf{x}$ for clarity, the subscript is the component index in $\mathbf{x}$ or $\mathbf{w}$):

$$f_\gamma(x_1,\cdots,x_d) = \sum_{n_1=0}^{\infty}\cdots\sum_{n_d=0}^{\infty}\frac{\partial^{n_1}}{\partial x_1^{n_1}}\cdots\frac{\partial^{n_d}}{\partial x_d^{n_d}}\frac{f_\gamma(w_1,\cdots,w_d)}{n_1!\cdots n_d!}(x_1-w_1)^{n_1}\cdots(x_d-w_d)^{n_d}$$

which for a second order approximation simplifies to

$$f_\gamma(x) \approx f_\gamma(w) + \partial_x f_\gamma(w)(x-w) + \frac{1}{2}(x-w)^\top \partial_{xx} f_\gamma(w)(x-w)$$

or rearranged

$$\partial_x f_\gamma(w)(x-w) = f_\gamma(x) - f_\gamma(w) - \frac{1}{2}(x-w)^\top \partial_{xx} f_\gamma(w)(x-w)$$

Setting $\mathbf{w} = \bar{\mathbf{x}}_i$ and taking into account that $\mathbf{g}^\top \mathbf{f}_{data}(\bar{\mathbf{x}}_i) = 0$ yields

$$\mathbf{g}^\top \partial_{\mathbf{x}} \mathbf{f}_{data}(\bar{\mathbf{x}}_i)(\mathbf{x}_i - \bar{\mathbf{x}}_i) = \mathbf{g}^\top (\mathbf{f}_{data}(\mathbf{x}_i) - r(\mathbf{x}_i, \bar{\mathbf{x}}_i))$$

where r is a term encapsulating all second and higher order derivatives. With this, (2.7) can be rewritten as

$$(\mathbf{x}_i - \bar{\mathbf{x}}_i)^\top \mathbf{C}_{\sigma_{\mathbf{x}}^2}^{-1}(\mathbf{x}_i - \bar{\mathbf{x}}_i) = \frac{\left\{\mathbf{g}^\top (\mathbf{f}_{data}(\mathbf{x}_i) - r(\mathbf{x}_i, \bar{\mathbf{x}}_i))\right\}^2}{\mathbf{g}^\top \partial_{\mathbf{x}} \mathbf{f}_{data}(\bar{\mathbf{x}}_i) \mathbf{C}_{\sigma_{\mathbf{x}}^2} \partial_{\mathbf{x}} \mathbf{f}_{data}^\top(\bar{\mathbf{x}}_i) \mathbf{g}}$$

leading to the rearranged maximum likelihood objective function

$$J = \sum_{i=1}^{N} \frac{\left\{\mathbf{g}^\top (\mathbf{f}_{data}(\mathbf{x}_i) - r(\mathbf{x}_i, \bar{\mathbf{x}}_i))\right\}^2}{\mathbf{g}^\top \partial_{\mathbf{x}} \mathbf{f}_{data}(\bar{\mathbf{x}}_i) \mathbf{C}_{\sigma_{\mathbf{x}}^2} \partial_{\mathbf{x}} \mathbf{f}_{data}^\top(\bar{\mathbf{x}}_i) \mathbf{g}} \tag{2.8}$$

The reformulated maximum likelihood objective function (2.8) easily lends itself to approximations. A primary obstacle to direct minimization of (2.8) is that neither $\partial_{\mathbf{x}} \mathbf{f}_{data}(\bar{\mathbf{x}}_i)$ nor $r(\mathbf{x}_i, \bar{\mathbf{x}}_i)$ can be expressed as they depend on unknown noise-free observations $\bar{\mathbf{x}}_i$. In contrast, they have to be substituted with $\partial_{\mathbf{x}} \mathbf{f}_{data}(\mathbf{x}_i)$ and $r(\mathbf{x}_i)$ that are computed from noisy observations. Formally, this is reflected in the approximated cost function [18]

$$J = \sum_{i=1}^{N} \frac{\mathbf{g}^\top (\mathbf{f}_{data}(\mathbf{x}_i) - \hat{r}(\mathbf{x}_i))(\mathbf{f}_{data}(\mathbf{x}_i) - \hat{r}(\mathbf{x}_i))^\top \mathbf{g}}{\mathbf{g}^\top \partial_{\mathbf{x}} \hat{\mathbf{f}}_{data}(\mathbf{x}_i) \mathbf{C}_{\sigma_{\mathbf{x}}^2} \partial_{\mathbf{x}} \hat{\mathbf{f}}_{data}^\top(\mathbf{x}_i) \mathbf{C}_{\sigma_{\mathbf{x}}^2} \mathbf{g}} \tag{2.9}$$

where $\hat{\mathbf{f}}_{data}(\mathbf{x}_i)$ indicates approximation of $\mathbf{f}_{data}(\mathbf{x}_i)$ and $\hat{r}(\mathbf{x}_i)$ indicates approximation of $r(\mathbf{x}_i)$.

As compared to the original maximum likelihood function, (2.9) takes the following simplifications:

- The gradient $\partial_{\mathbf{x}} \mathbf{f}_{data}(\bar{\mathbf{x}}_i)$ is approximated with the gradient $\partial_{\mathbf{x}} \mathbf{f}_{data}(\mathbf{x}_i)$ computed based on noisy observations.

- Similarly to gradients, the residual components $r(\mathbf{x}_i, \bar{\mathbf{x}}_i)$ that result from the Taylor series expansion depend on the unknown true observations $\bar{\mathbf{x}}_i$. If the lifting function $\mathbf{f}_{data}(\bar{\mathbf{x}}_i)$ can be expressed as a quadratic form in terms of $\bar{\mathbf{x}}_i$, the true observations

cancel if we apply expected value:

$$\begin{aligned} r(\mathbf{x}_i, \bar{\mathbf{x}}_i) &= \frac{1}{2}(\mathbf{x}-\mathbf{y})^\top \partial_{\mathbf{xx}} \mathbf{f}_{data}(\mathbf{y})(\mathbf{x}-\mathbf{y}) \\ &= \operatorname{trace}\left(\frac{1}{2}(\mathbf{x}-\mathbf{y})^\top \partial_{\mathbf{xx}} \mathbf{f}_{data}(\mathbf{y})(\mathbf{x}-\mathbf{y})\right) \\ \mathbb{E}r(\mathbf{x}_i, \mathbf{x}_{0,i}) &= \mathbb{E}\left\{\operatorname{trace}\left(\frac{1}{2}(\mathbf{x}-\mathbf{y})^\top \partial_{\mathbf{xx}} \mathbf{f}_{data}(\mathbf{y})(\mathbf{x}-\mathbf{y})\right)\right\} \\ &= \frac{1}{2}\mathbb{E}\left\{\operatorname{trace}\left((\mathbf{x}-\mathbf{y})(\mathbf{x}-\mathbf{y})^\top \partial_{\mathbf{xx}} \mathbf{f}_{data}(\mathbf{y})\right)\right\} \\ &= \frac{1}{2}\operatorname{trace}\left(\mathbb{E}\left\{(\mathbf{x}-\mathbf{y})(\mathbf{x}-\mathbf{y})^\top\right\} \partial_{\mathbf{xx}} \mathbf{f}_{data}(\mathbf{y})\right) \\ &= \frac{1}{2}\operatorname{trace}\left(\mathbf{C}\,\partial_{\mathbf{xx}} \mathbf{f}_{data}(\mathbf{y})\right) \end{aligned}$$

This simplification applies to higher order terms as well, but dependence on the observations will not vanish. As dependence on unknown true observations is not desired, they have to be expressed in terms of actual observations, in the same fashion as for gradients.

One way to minimize (2.9) is to employ direct search methods. A more robust technique is to use an iterative scheme. For this end, introduce the compact notation

$$J = \sum_{i=1}^{N} \frac{\mathbf{g}^\top \mathbf{A}_i \mathbf{g}}{\mathbf{g}^\top \mathbf{B}_i \mathbf{g}}$$

so that for the approximated likelihood function to attain a minimum we need

$$\partial_{\mathbf{g}} J = \sum_{i=1}^{N} \frac{1}{\mathbf{g}^\top \mathbf{B}_i \mathbf{g}} \mathbf{A}_i \mathbf{g} - \sum_{i=1}^{N} \frac{\mathbf{g}^\top \mathbf{A}_i \mathbf{g}}{\left(\mathbf{g}^\top \mathbf{B}_i \mathbf{g}\right)^2} \mathbf{B}_i \mathbf{g} = 0.$$

Let us introduce the substitutions

$$\begin{aligned} \mathbf{M}_{\mathbf{g}} &= \sum_{i=1}^{N} \frac{1}{\mathbf{g}^\top \mathbf{B}_i \mathbf{g}} \mathbf{A}_i \\ \mathbf{N}_{\mathbf{g}} &= \sum_{i=1}^{N} \frac{\mathbf{g}^\top \mathbf{A}_i \mathbf{g}}{\left(\mathbf{g}^\top \mathbf{B}_i \mathbf{g}\right)^2} \mathbf{B}_i \\ \mathbf{X}_{\mathbf{g}} &= \mathbf{M}_{\mathbf{g}} - \mathbf{N}_{\mathbf{g}} \end{aligned}$$

so that we have

$$\partial_{\mathbf{g}} J = \mathbf{M}_{\mathbf{g}}\mathbf{g} - \mathbf{N}_{\mathbf{g}}\mathbf{g} = \mathbf{X}_{\mathbf{g}}\mathbf{g} = 0. \tag{2.10}$$

Finally, the solution to (2.10) can be iteratively sought in several ways. One possibility is the so-called fundamental numerical scheme [18], which seeks the solution as the eigenvector corresponding to the smallest eigenvalue in the eigenvalue problem

$$\mathbf{X}_{\mathbf{g}}\boldsymbol{\xi} = \mu\boldsymbol{\xi}. \tag{2.11}$$

This approach is inspired by the fact that a vector $\mathbf{g}$ satisfies (2.10) if and only if it is a solution to the ordinary eigenvalue problem (2.11) corresponding to the eigenvalue $\mu = 0$. Let $\mathbf{g}_{(k)}$ be the current approximate solution in iteration k, and $\mathbf{X}_{\mathbf{g}_{(k)}}$ be computed with the substitution of $\mathbf{g}_{(k)}$. In a single iteration, the updated solution $\mathbf{g}_{(k+1)}$ is chosen from that eigenspace of $\mathbf{X}_{\mathbf{g}_{(k)}}$ that most closely approximates the null space of $\mathbf{X}_{\mathbf{g}}$, which corresponds to the eigenvalue closest to zero in absolute value. In other words, we solve a series of eigenvalue problems

$$\mathbf{X}_{\mathbf{g}_{(k)}}\mathbf{g}_{(k+1)} = \mu_{(k+1)}\mathbf{g}_{(k+1)}. \tag{2.12}$$

Another possibility is to seek the eigenvalue closest to 1 in the eigenvalue problem as in [49]

$$\mathbf{M}_{\mathbf{g}}\boldsymbol{\xi} = \mu\mathbf{N}_{\mathbf{g}}\boldsymbol{\xi}$$

leading to a similar series of eigenvalue problems

$$\mathbf{M}_{\mathbf{g}_{(k)}}\mathbf{g}_{(k+1)} = \mu_{(k+1)}\mathbf{N}_{\mathbf{g}_{(k)}}\mathbf{g}_{(k+1)}. \tag{2.13}$$

Like all iterative schemes, (2.11) and (2.13) require feasible initialization to avoid slow convergence or divergence. Less accurate but non-iterative methods help provide initial estimates that may already be close to the final solution.

2.1.3 Nonlinear least-squares methods

As previously seen, maximum likelihood methods are solved with a gradient search or an iterative scheme, where the former needs proper initialization and the latter may diverge, especially in the presence of large noise. The principle of nonlinear least-squares methods is to minimize the residual sum of squares (RSS), or in other words, the sum of the squares of errors that result when we substitute data into the estimated objective function:

$$\hat{\mathbf{g}} = \arg\min_{\mathbf{g}} \sum_{i=1}^{N} \big(f(\mathbf{x}_i, \mathbf{g})\big)^2.$$

The motivation behind the approach is that standard least-squares optimization methods can be employed to find a minimum, and when true values of data and parameters are substituted into the expression, we get zero error:

$$\sum_{i=1}^{N} \big(f(\bar{\mathbf{x}}_i, \mathbf{g})\big)^2 = 0.$$

Let a linearization (lifting) of 2D data be

$$\mathbf{z}_i^\top = \begin{bmatrix} x_i^2 & x_i y_i & y_i^2 & x_i & y_i & 1 \end{bmatrix} \tag{2.14}$$

which captures quadratic curves, and a linearization (lifting) of 3D data be

$$\mathbf{z}_i^\top = \begin{bmatrix} x_i^2 & y_i^2 & z_i^2 & y_i z_i & x_i z_i & x_i y_i & x_i & y_i & z_i & 1 \end{bmatrix} \tag{2.15}$$

which captures quadratic surfaces where x_i, y_i and z_i are data point coordinates in two or three dimensions. (The same principles apply for higher-order polynomials.) The sample covariance matrix is then

$$\mathbf{D} = \frac{1}{N}\sum_i \left(\mathbf{z}_i - \frac{1}{N}\sum_j \mathbf{z}_j\right)\left(\mathbf{z}_i - \frac{1}{N}\sum_j \mathbf{z}_j\right)^\top .$$

The least-squares solution then solves the eigenvalue problem

$$\mathbf{Dg} = \lambda \mathbf{g}$$

for the minimum eigenvalue λ.

Unfortunately, simple as it is, directly minimizing algebraic error is statistically inaccurate and leads to heavily biased estimates. As apparent from (2.14) and (2.15) the effect of noise on various components may nonlinear, such as for x^2 in (2.14) where

$$\mathbf{z}_i^\top = \begin{bmatrix} (\bar{x}_i + \tilde{x}_i)^2 & (\bar{x}_i + \tilde{x}_i)(\bar{y}_i + \tilde{y}_i) & (\bar{y}_i + \tilde{y}_i)^2 & \bar{x}_i + \tilde{x}_i & \bar{y}_i + \tilde{y}_i & 1 \end{bmatrix}.$$

This must be taken into account in devising such estimation schemes. This motivates an approach to maintain the simplicity of least-squares methods yet combat their statistical inaccuracy, which can be accomplished with a noise cancellation scheme that removes the distortion effects of noise (as much as possible).

2.1.4 Estimation using geometric approximation

Estimates obtained with the least-squares approach are statistically inaccurate or biased [73]. A more robust way is to use a linear approximation of the geometric distance and minimize

$$J = \sum_{i=1}^{N} \frac{\left(f(\mathbf{x}_i, \mathbf{g})\right)^2}{\left\|\nabla f(\mathbf{x}_i, \mathbf{g})\right\|^2} \tag{2.16}$$

where the operator $\nabla = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \end{bmatrix}$ for 2D and $\nabla = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \end{bmatrix}$ for 3D. Unfortunately, (2.16) cannot be solved without iterations but a further simplification of it, called Taubin's method [61]

$$J = \frac{\sum_{i=1}^{N}\left(f(\mathbf{x}_i, \mathbf{g})\right)^2}{\sum_{i=1}^{N}\left\|\nabla f(\mathbf{x}_i, \mathbf{g})\right\|^2}$$

lends itself to a non-iterative solution. In two dimensions, for instance,

$$\begin{aligned}
\sum_{i=1}^{N}\left(f(x_i, y_i, \mathbf{g})\right)^2 &= \mathbf{g}^\top \sum_{i=1}^{N}\left(\begin{bmatrix} x^2 & xy & y^2 & x & y & 1 \end{bmatrix}^\top \begin{bmatrix} x^2 & xy & y^2 & x & y & 1 \end{bmatrix}\right)\mathbf{g} \\
\sum_{i=1}^{N}\left\|\nabla f(x_i, y_i, \mathbf{g})\right\|^2 &= \mathbf{g}^\top \sum_{i=1}^{N}\left(\begin{bmatrix} 2x & y & 0 & 1 & 0 & 0 \\ 0 & x & 2y & 0 & 1 & 0 \end{bmatrix}^\top \begin{bmatrix} 2x & y & 0 & 1 & 0 & 0 \\ 0 & x & 2y & 0 & 1 & 0 \end{bmatrix}\right)\mathbf{g}
\end{aligned}$$

such that the problem can be formulated as

$$J = \frac{\mathbf{g}^\top \mathbf{A}\mathbf{g}}{\mathbf{g}^\top \mathbf{B}\mathbf{g}}$$

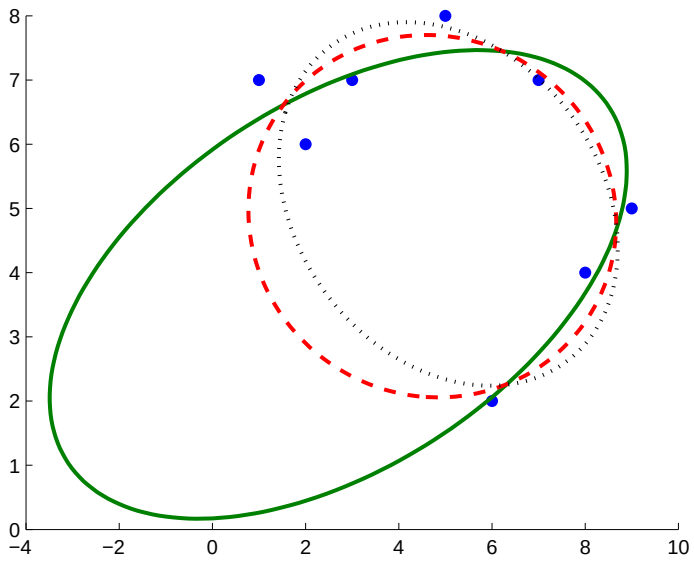


Figure 2.2: Comparison of maximum likelihood (continuous line), nonlinear least-squares (dotted line) and Taubin's (dashed line) ellipse fitting methods to a set of two-dimensional coordinates.

with $\|\mathbf{g}\| = 1$ and solved as a generalized eigenvector problem

$$\mathbf{g}^\top \mathbf{A}\mathbf{g} = \mu \mathbf{g}^\top \mathbf{B}\mathbf{g}.$$

The solution $\hat{\mathbf{g}}$, as previously seen, is the eigenvector that belongs to the smallest eigenvalue.

Figure 2.2 compares various quadratic curve fitting methods to a set of two-dimensional data points with coordinates (1;7), (2;6), (5;8), (7;7), (9;5), (3;7), (6;2) and (8;4). The best-fit ellipse obtained with the maximum likelihood method (Section 2.1.1), which minimizes the sum of squares of (signed) distances to the quadratic curve foot points using the Levenberg–Marquardt method, is shown in continuous line, whereas the non-iterative methods non-linear least-squares fit (Section 2.1.3) and estimation with geometric approximation using Taubin's fit (Section 2.1.4), are shown in dotted line and dashed line, respectively. The figure highlights that there might be substantial difference between the fit obtained with geometric distance and algebraic distance minimization.

2.1.5 Hyper-accurate methods

As we have seen, the ordinary least squares method solves the eigenvalue problem

$$\mathbf{D}\mathbf{g} = \lambda \mathbf{g}$$

for the minimum eigenvalue λ, effectively minimizing algebraic (substitution) error. However, we may reformulate the above problem as

$$\mathbf{D}\mathbf{g} = \lambda \mathbf{Q}\mathbf{g}$$

introducing a properly chosen normalization matrix $\mathbf{Q}$. For ordinary least squares, $\mathbf{Q} = \mathbf{I}$ but hyper-accurate methods choose $\mathbf{Q}$ such that higher-order bias terms induced by noise are canceled. This allows these (non-iterative) methods to completely eliminate estimation bias in $\hat{\mathbf{g}}$ and reduce variance [39], even if they do not attain the theoretical lower bound.[1]

Let

$$\mathbf{z}_i = \begin{bmatrix} x_i^2 & 2x_i y_i & y_i^2 & 2f_0 x_i & 2f_0 y_i & f_0^2 \end{bmatrix}$$

where f_0 is a scaling constant in the order of (mean-free) x_i and y_i,

$$\mathbf{D} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{z}_i \mathbf{z}_i^\top,$$

and

$$\mathbf{V}_i = 4\begin{bmatrix} x & xy & 0 & f_0 x & 0 & 0 \\ xy & x^2+y^2 & xy & f_0 y & f_0 x & 0 \\ 0 & xy & y^2 & 0 & f_0 y & 0 \\ f_0 x & f_0 y & 0 & f_0^2 & 0 & 0 \\ 0 & f_0 x & f_0 y & 0 & f_0^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

which is what Taubin's method effectively uses as normalization matrix. (For Taubin's method, $\mathbf{Q} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{V}_i$.)

The normalization matrix for hyper-accurate fitting [39] is calculated as

$$\mathbf{Q} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{V}_i + 2\mathscr{S}\left\{\mathbf{z}_c \mathbf{e}_{1,3}^\top\right\} - \frac{1}{N^2}\sum_{i=1}^{N}\left(\mathrm{trace}\left(\mathbf{D}^\dagger \mathbf{V}_i\right)\mathbf{z}_i\mathbf{z}_i^\top + \left(\mathbf{z}_i^\top \mathbf{D}^\dagger \mathbf{z}_i\right)\mathbf{V}_i + 2\mathscr{S}\left\{\mathbf{V}_i \mathbf{D}^\dagger \mathbf{z}_i \mathbf{z}_i^\top\right\}\right)$$

where

$$\begin{aligned} \mathbf{z}_c &= \frac{1}{N}\sum_i \mathbf{z}_i \\ \mathbf{e}_{1,3}^\top &= \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \end{aligned}$$

and the symbol $\mathscr{S}$ denotes symmetrization

$$\mathscr{S}\{\mathbf{A}\} = \frac{1}{2}\left(\mathbf{A} + \mathbf{A}^\top\right).$$

Hyper-accurate methods are preferred when the noise level is low. For higher noise level, consistent algebraic least squares (introduced in Section 2.1.6), is a more appropriate choice, which also uses fewer matrix operations and is nearly as accurate as hyper-accurate methods.

[1] This theoretical lower bound is called the Kanatani–Cramer–Rao lower bound $\mathbf{C}_{KCR}(\hat{\mathbf{g}})$ and will be covered in more detail in Section 2.1.7. Permitting iterative algorithms with an adaptive choice of the normalization matrix $\mathbf{Q}$, as in [5], it is possible to both entirely eliminate the bias in $\hat{\mathbf{g}}$ and achieve the theoretically lowest variance $\mathbf{C}_{KCR}(\hat{\mathbf{g}})$.

2.1.6 Nonlinear Koopmans estimator

As previously seen, estimation strategies may be grouped into two coarse categories. *High-accuracy* methods, such as maximum likelihood or approximated maximum likelihood estimation, are close to the best possible estimate that can be obtained from the measured data but are computationally more expensive and require proper initialization. *Low-accuracy* methods, such as least squares or Taubin's method, on the other hand, are easy to compute and require no initialization but are not nearly as accurate. A method that alloys the strengths of high-accuracy methods and mitigates the drawbacks of low-accuracy methods, even in the context of large noise level, is therefore much sought after. In particular, non-iterative schemes, even if they are not strictly optimal [39], can still deliver accurate estimates, and avoid any possibility of divergence inherent to iterative algorithms, especially in the presence of high levels of noise.

The principle of the nonlinear Koopmans estimator (also known as nonlinear extension to the Koopmans method [66] and consistent algebraic least squares [42, 48]), is to build the sample data covariance matrix from data subject to the lifting function $\mathbf{f}_{data}$ and use an appropriate (pre-computed) noise covariance matrix to cancel the matrix rank-increase induced by measurement noise. Loosely speaking, the method tries to match the sample data covariance matrix with the theoretical noise covariance matrix that corresponds to measurement noise. Like with AML in Section 2.1.2, the noise covariance matrix depends on $\bar{\mathbf{x}}_i$, which are unknown, but can be approximated with $\mathbf{x}_i$, which are available.

When the lifting function is an identity mapping, the estimation problem is captured by the simple linear relationship

$$\mathbf{g}^\top \mathbf{x} = 0.$$

The original work of Koopmans [40] addressed this estimation problem, and proposed a non-iterative but fairly inaccurate method to estimate model parameters from second-order statistical characteristics. The estimation method matches the data covariance matrix with the conceptual noise covariance matrix. The underlying assumption is that were it not for the noise present in observations $\mathbf{x}_i$, the data covariance matrix would be a singular matrix. Thus, finding the eigenvector of the data covariance matrix with respect to a noise covariance matrix with the smallest-magnitude eigenvalue, parameter estimates can be found [66, 42, 48].

For convenience, let us introduce the (sample) data matrix and the (sample) data covariance matrix.

Definition 1. Data matrix.

$$\mathbf{X}^\top = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 & \dots & \mathbf{x}_N \end{bmatrix}$$

■

Definition 2. (Sample) data covariance matrix.

$$\begin{aligned} \mathbf{D} &= \mathbb{E}\left\{(\mathbf{x}-\mathbb{E}\mathbf{x})\,(\mathbf{x}-\mathbb{E}\mathbf{x})^\top\right\} \\ &\approx \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{x}_i - \frac{1}{N}\sum_{j=1}^{N}\mathbf{x}_j\right)\left(\mathbf{x}_i - \frac{1}{N}\sum_{j=1}^{N}\mathbf{x}_j\right)^\top \end{aligned}$$

■

Definition 3. Data covariance matrix of unobserved true data.

$$\bar{\mathbf{D}} = \frac{1}{N}\sum_{i=1}^{N}\left(\bar{\mathbf{x}}_i - \frac{1}{N}\sum_{j=1}^{N}\bar{\mathbf{x}}_j\right)\left(\bar{\mathbf{x}}_i - \frac{1}{N}\sum_{j=1}^{N}\bar{\mathbf{x}}_j\right)^{\top}$$

■

By definition, the matrix $\bar{\mathbf{D}}$ is rank-deficient (singular) and the matrix $\mathbf{D}$ is typically regular (positive definite). The positive definiteness of $\mathbf{D}$ is attributable to measurement noise. The Koopmans estimator therefore uses the property that the data covariance matrix can be decomposed into a contribution by the noise-free data and the noise

$$\mathbf{D} = \bar{\mathbf{D}} + \mathbf{C}(\mu, \boldsymbol{\varphi})$$

where $\mathbf{C}(\mu, \boldsymbol{\varphi})$ is the noise covariance matrix that has a magnitude parametrized by μ and a structure parametrized by $\boldsymbol{\varphi}$:

$$\mathbf{C}(\mu, \boldsymbol{\varphi}) = \mu\mathbf{C}(\boldsymbol{\varphi}).$$

For the 2D case, $\mathbf{C}(\mu, \boldsymbol{\varphi})$ may admit the particularly simple form of

$$\mathbf{C}(\mu, \varphi) = \mu\begin{bmatrix} \sin^2\varphi & 0 \\ 0 & \cos^2\varphi \end{bmatrix}.$$

This special structure can be generalized as

$$\mathbf{C} = \mu\mathbf{C}_{\varphi} = \mu\mathrm{diag}\left\{\sigma^2_{\mathbf{x}}\right\}$$

where

$$\sigma^2_{\mathbf{x}} = \begin{bmatrix} \sigma^2_{x_1} & \dots & \sigma^2_{x_n} \end{bmatrix}$$

and each $\sigma^2_{x_r}$ corresponds to the variance of noise contaminating a particular component of x_r, assuming unit noise magnitude. In other words, $\left\|\sigma^2_{\mathbf{x}}\right\| = 1$ and the values of $\sigma^2_{x_r}$ reflect the relative magnitude of the noise variances w.r.t. one another. Colloquially, $\boldsymbol{\sigma}^2_{\mathbf{x}}$ is expressed as $\boldsymbol{\sigma}^2_{\mathbf{x}} = \mu\bar{\boldsymbol{\sigma}}^2_{\mathbf{x}}$ with μ being "magnitude" and $\bar{\boldsymbol{\sigma}}^2_{\mathbf{x}}$ "direction".

Indeed, we may observe that for random variables of noisy data components x_r and x_s we have

$$\begin{aligned} \mathbb{E}x_r^2 &= \mathbb{E}(\bar{x}_r + \tilde{x}_r)^2 = \mathbb{E}(\bar{x}_r^2 + 2\bar{x}_r\tilde{x}_r + \tilde{x}_r^2) = \mathbb{E}\bar{x}_r^2 + \sigma^2_{x,r} \\ \mathbb{E}x_r x_s &= \mathbb{E}(\bar{x}_r + \tilde{x}_r)(\bar{x}_s + \tilde{x}_s) = \mathbb{E}(\bar{x}_r\bar{x}_s + \bar{x}_r\tilde{x}_s + \tilde{x}_r\bar{x}_s + \tilde{x}_r\tilde{x}_s) = \mathbb{E}\bar{x}_r\mathbb{E}\bar{x}_s \end{aligned}$$

which would imply $\mathrm{cov}(x_r, x_r) = \bar{x}_r^2 + \sigma^2_{x,r}$ and $\mathrm{cov}(x_r, x_s) = 0$ elsewhere.

Multiplying by $\mathbf{g}$ from both sides we have

$$\mathbf{g}^{\top}\mathbf{D}\mathbf{g} = \mathbf{g}^{\top}\bar{\mathbf{D}}\mathbf{g} + \mathbf{g}^{\top}\mathbf{C}(\mu, \boldsymbol{\varphi})\mathbf{g} = \mathbf{g}^{\top}\mathbf{C}(\mu, \boldsymbol{\varphi})\mathbf{g}$$

as

$$\mathbf{g}^{\top}\bar{\mathbf{D}}\mathbf{g} = \mathbf{g}^{\top}\bar{\mathbf{X}}^{\top}\bar{\mathbf{X}}\mathbf{g}$$

where $\bar{\mathbf{X}}\mathbf{g} = \mathbf{0}$ by problem formulation.

Finding a model parameter vector $\mathbf{g}$ that satisfies

$$\begin{aligned} \mathbf{g}^\top \mathbf{D}\mathbf{g} &= \mathbf{g}^\top \mu \mathbf{C}(\boldsymbol{\varphi})\mathbf{g} \\ \mathbf{g}^\top \mathbf{g} &= 1 \end{aligned}$$

is a generalized eigenvalue problem and the optimum solution is the eigenvector $\mathbf{g}$ that belongs to the smallest eigenvalue μ. As the matrix $\mathbf{D}$ is positive definite, the null space of $\mathbf{D} - \mu\mathbf{C}$ yields the parameter vector $\mathbf{g}$ (up to scale, normalized to $\|\mathbf{g}\| = 1$). The implication that $\mu \geqq 0$ (which follows from the fact that by construction both $\mathbf{D}$ and $\mathbf{C}$ are positive definite) expresses a physical constraint that noise magnitude cannot be negative.

In the same fashion as with total least-squares estimation [20], the accuracy of the estimates can be improved if we use singular value decomposition instead of eigenvalue decomposition. Reformulating,

$$\begin{aligned} \mathbf{g}^\top \mathbf{X}^\top \mathbf{X}\mathbf{g} &= \mathbf{g}^\top \mu \mathbf{R}^\top \mathbf{R}\mathbf{g} \\ \mathbf{g}^\top \mathbf{g} &= 1 \end{aligned}$$

where $\mathbf{R}^\top \mathbf{R} = \mathbf{C}(\boldsymbol{\varphi})$, hence the parameters are estimated as a generalized singular value problem on the matrix pair $(\mathbf{X}, \mathbf{R})$. The right singular vector $\mathbf{g}$ that belongs to the smallest singular value s where $s^2 = \mu$ is the parameter vector we seek.

Unlike the case of the original Koopmans estimator, once data are subject to the nonlinear lifting function, the covariance matrix no longer assumes a simple linear structure, and cannot be directly decomposed into magnitude parameter and structure matrix. In fact, it turns out that in order to compute its entries, noise-free data would have to be at our disposal, which are clearly not available. However, when the lifting function is polynomial in terms of data, which is a relatively mild restriction, a series of simple steps can be taken to approximate the noise covariance matrix from data.

For compactness, let us write

$$\bar{\mathbf{z}}_i = \mathbf{f}_{data}(\bar{\mathbf{x}}_i).$$

In other words, once we make the substitution, the relationship

$$\mathbf{f}_{data}(\bar{\mathbf{x}}_i)^\top \mathbf{f}_{par}(\mathbf{g}) = 0$$

simplifies into a (pseudo-)linear relationship

$$\bar{\mathbf{z}}_i^\top \boldsymbol{\theta} = 0$$

that splits the expression into data and parameters. When the function $\mathbf{f}_{par}(\mathbf{g})$ is the identity mapping, we arrive at the simpler relationship

$$\bar{\mathbf{z}}_i^\top \mathbf{g} = 0.$$

Let us begin our discussion by adapting definitions for the linear cancellation method to the nonlinear case.

Definition 4. Linearized data matrix. Let

$$\mathbf{Z} = \begin{bmatrix} \mathbf{f}_{data}^{\top}(\mathbf{x}_1) \\ \mathbf{f}_{data}^{\top}(\mathbf{x}_2) \\ \vdots \\ \mathbf{f}_{data}^{\top}(\mathbf{x}_N) \end{bmatrix}_{N,n} = \begin{bmatrix} \mathbf{z}_1^{\top} \\ \mathbf{z}_2^{\top} \\ \vdots \\ \mathbf{z}_N^{\top} \end{bmatrix}_{N,n}$$

where N is the number of data points and $\mathbf{f}_{data}(\mathbf{x}) : \mathbb{R}^m \to \mathbb{R}^n$ is the lifting function. ■

Let us assume that $\mathbf{f}_{data}(\mathbf{x}) = \mathbf{z}$ is a polynomial function in terms of components x_r in

$$\mathbf{x}^{\top} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}.$$

For instance, we might have

$$\mathbf{z}^{\top} = \begin{bmatrix} x_1^2 & x_1 x_2 & x_2^2 & x_1 & x_2 & 1 \end{bmatrix}.$$

Definition 5. Let

$$\mathbf{f}_{data}(\mathbf{x}) = \mathbf{z} = \begin{bmatrix} z_1 & z_2 & \dots & z_{n_d} \end{bmatrix}.$$

Then the *linearized data covariance matrix* (and its approximation from finite N data samples) is

$$\begin{aligned} \mathbf{D} &= \mathbb{E}\left\{(\mathbf{z} - \mathbb{E}\mathbf{z})\,(\mathbf{z} - \mathbb{E}\mathbf{z})^{\top}\right\} \\ &\approx \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{z}_i - \frac{1}{N}\sum_{j=1}^{N}\mathbf{z}_j\right)\left(\mathbf{z}_i - \frac{1}{N}\sum_{j=1}^{N}\mathbf{z}_j\right)^{\top} \end{aligned}$$

■

Definition 6. Let

$$\mathbf{f}_{data}(\tilde{\mathbf{x}}) = \tilde{\mathbf{z}} = \begin{bmatrix} \tilde{z}_1 & \tilde{z}_2 & \dots & \tilde{z}_{n_d} \end{bmatrix}$$

where $\tilde{z}_r = z_r - \bar{z}_r$ in which $\bar{z}_r$ is the unobservable true value of the component. The *linearized noise cancellation matrix* (and its approximation from finite N data samples) is then

$$\begin{aligned} \mathbf{C} &= \mathbb{E}\left\{(\tilde{\mathbf{z}} - \mathbb{E}\tilde{\mathbf{z}})\,(\tilde{\mathbf{z}} - \mathbb{E}\tilde{\mathbf{z}})^{\top}\right\} \\ &\approx \frac{1}{N}\sum_{i=1}^{N}\left(\tilde{\mathbf{z}}_i - \frac{1}{N}\sum_{j=1}^{N}\tilde{\mathbf{z}}_j\right)\left(\tilde{\mathbf{z}}_i - \frac{1}{N}\sum_{j=1}^{N}\tilde{\mathbf{z}}_j\right)^{\top} \end{aligned}$$

■

The estimator then proceeds as follows. First, we assume that the data covariance matrix can be decomposed into a contribution by the noise-free data $\bar{\mathbf{D}}$ and the noise contribution $\mathbf{C}$, i.e.

$$\mathbf{D} = \bar{\mathbf{D}} + \mathbf{C}.$$

Multiplying by $\mathbf{g}$ from both sides,

$$\mathbf{g}^{\top}\mathbf{D}\mathbf{g} = \mathbf{g}^{\top}\bar{\mathbf{D}}\mathbf{g} + \mathbf{g}^{\top}\mathbf{C}\mathbf{g} = \mathbf{g}^{\top}\mathbf{C}\mathbf{g}$$

as

$$\mathbf{g}^\top \bar{\mathbf{D}}\mathbf{g} = \mathbf{g}^\top \bar{\mathbf{Z}}^\top \bar{\mathbf{Z}}\mathbf{g}$$

where $\bar{\mathbf{Z}}\mathbf{g} = \mathbf{0}$ by definition. This implies that solving the above matrix equality entails finding a model parameter vector $\mathbf{g}$ that satisfies

$$\begin{aligned} \mathbf{g}^\top \mathbf{D}\mathbf{g} &= \mathbf{g}^\top \mathbf{C}\mathbf{g} \\ \mathbf{g}^\top \mathbf{g} &= 1 \end{aligned}$$

in which $\mathbf{g}$ is unknown and $\mathbf{C}$ is known up to some noise parameters. Note that the matrix $\mathbf{C}$ for the nonlinear case no longer has the special structure the noise covariance matrix had for the linear case, namely

$$\mathbf{C} = \mu\mathbf{C}_\varphi \neq \mu \mathrm{diag}\left\{\sigma^2_{\mathbf{x}}\right\}.$$

Unfortunately, such a simplification is no longer possible when the components x_r are subject to a mapping $\mathbf{f}_{data}(x_1, x_2, \ldots, x_n)$. Nonetheless, we can follow the same approach as with the linear case to decompose the covariance of measured data into the covariance of unobservable true data and a noise contribution. If the mapping is a polynomial function, the algorithm can be broken down into a series of simple steps. This yields a polynomial generalization of Koopmans' original method [40].

Example 2. Covariance of a quadratic component. Suppose we would like to estimate the self-covariance of a term x^2, i.e. $\mathrm{cov}(x^2, x^2)$. This corresponds to the top left entry of the matrix $\mathbf{C} = \mathbb{E}\left(\tilde{\mathbf{z}}_i \tilde{\mathbf{z}}_i^\top\right)$ with data transformed as

$$\mathbf{z}^\top = \begin{bmatrix} x^2 & xy & y^2 & x & y & 1 \end{bmatrix}$$

where we have dropped the index i in x_i and y_i for brevity.

Let us calculate

$$\mathbb{E}\left\{\left(x^2\right)^2\right\} = \mathbb{E}\left(x^4\right) = \mathbb{E}\left(\bar{x} + \tilde{x}\right)^4 = \mathbb{E}\left(\bar{x}^4 + 4\bar{x}^3\tilde{x} + 6\bar{x}^2\tilde{x}^2 + 4\bar{x}\tilde{x}^3 + \tilde{x}^4\right)$$

where $\mathbb{E}\left(4\bar{x}^3\tilde{x}\right) = 4\bar{x}^3\mathbb{E}\left(\tilde{x}\right) = 0$ and $\mathbb{E}\left(4\bar{x}\tilde{x}^3\right) = 4\bar{x}\mathbb{E}\left(\tilde{x}^3\right) = 0$ (expected value of odd central moment equals zero), yielding

$$\mathbb{E}\left(x^4\right) = \mathbb{E}\left(\bar{x}^4 + 6\bar{x}^2\tilde{x}^2 + \tilde{x}^4\right)$$

where $\mathbb{E}\left(\tilde{x}^2\right) = \sigma_x^2$ and $\mathbb{E}\left(\tilde{x}^4\right) = 3\sigma_x^4$ (expected value of even central moment) such that

$$\mathbb{E}\left(x^4\right) = \bar{x}^4 + 6\bar{x}^2\sigma_x^2 + 3\sigma_x^4.$$

This means that $\mathrm{cov}(x^2, x^2)$ comprises of a noise-free part $\mathrm{cov}(\bar{x}^2, \bar{x}^2) = \mathbb{E}\left(\bar{x}^4\right) = \bar{x}^4$ and a noise part $6\sigma_x^2\mathbb{E}\left(\tilde{x}^2\right) + 3\sigma_x^4$. Unfortunately, $\mathbb{E}\left(\tilde{x}^2\right)$ is an unknown quantity as we cannot observe $\bar{x}$ but we observe (the noisy) x instead. ♣

As seen from the example, such decomposition typically yields a noise contribution that in turn depends on unobservable data. Fortunately, the approach can be applied recursively to cancel these terms.

$\mathbb{E}\left(w_i^p\right) = \mathbb{E}\left\{(\bar{w}_i + \tilde{w}_i)^p\right\}$	An observed quantity is split into an unobserved quantity and noise.
$\mathbb{E}\left(\tilde{w}_i^{2p}\right) = \sigma_w^{2p}(2p-1)!!$	Even central moments of the normally distributed variable $\tilde{w}$ where $(2p-1)!! = (2p-1)(2p-3)\ldots 1$.
$\mathbb{E}\left(\tilde{w}_i^{2p-1}\right) = 0$	Odd central moments of the normally distributed variable $\tilde{w}$.
$\mathbb{E}(w_i) \approx \frac{1}{N}\sum_{i=1}^{N} w_i$	Expected value is approximated with mean.

Table 2.1: Operations involved in separating a sample covariance matrix into a covariance matrix of noise-free components and noise contribution.

Example 3. Canceling unobservable data in the covariance of a quadratic component.

$$\begin{aligned}\mathbb{E}x^4 &= \bar{x}^4 + 6\sigma_x^2\bar{x}^2 + 3\sigma_x^4 \\ &= \bar{x}^4 + 6\sigma_x^2\left(\mathbb{E}x^2 - \mathbb{E}\tilde{x}^2\right) + 3\sigma_x^4 \\ &= \bar{x}^4 + 6\sigma_x^2\mathbb{E}x^2 - 6\sigma_x^4 + 3\sigma_x^4 \\ &= \bar{x}^4 + 6\sigma_x^2\mathbb{E}x^2 - 3\sigma_x^4\end{aligned}$$

in which the substitution for $\mathbb{E}x^2 - \mathbb{E}\tilde{x}^2$ comes from

$$\begin{aligned}\mathbb{E}x^2 &= \mathbb{E}(\bar{x}+\tilde{x})^2 = \mathbb{E}\left(\bar{x}^2 + 2\bar{x}\tilde{x} + \tilde{x}^2\right) \\ &= \bar{x}^2 + \mathbb{E}\left(\tilde{x}^2\right) \\ &= \bar{x}^2 + \sigma_x^2\end{aligned}$$

i.e. $\bar{x}^2 = \mathbb{E}x^2 - \sigma_x^2 \approx \text{average}\left(x^2\right) - \sigma_x^2$. As a result,

$$\mathbb{E}x^4 - \mathbb{E}\tilde{x}^4 = 6\sigma_x^2\mathbb{E}x^2 - 3\sigma_x^4.$$

♣

Notice how the expression finally reduces to components that depend on either a noise parameter such as σ_x^2 and σ_x^4, or components that can be approximated from observed samples such as $\mathbb{E}x^2 \approx \frac{1}{N}\sum_i x_i^2$.

Let w_i be an arbitrary component in $\mathbf{x}_i$ and its noise contribution $\tilde{w}_i$ be normally distributed with variance σ_w^2. Table 2.1 summarizes the typical operations involved in computing the noise contribution in entries of the linearized sample covariance matrix $\mathbf{D}$.

Once expressions for all entries are derived, we can set up a similar cancellation scheme as with the linear case. The algorithm produces a polynomial eigenvalue problem (PEP)

$$\boldsymbol{\Psi}(\mu)\mathbf{g} = \left(\mathbf{D} - \mathbf{C}(\mu)\right)\mathbf{g} = \mathbf{0}$$

in which $\mathbf{C}(\mu)$ is a polynomial in the scalar μ, meant to cancel noise effects and our goal is to find the smallest-magnitude (positive real) eigenvalue μ. In general, $\mathbf{C}(\mu) = \mathbf{C}(\mu, \sigma_{\mathbf{x}}^2, \mathbf{x}_i)$, i.e. $\mathbf{C}(\mu)$ depends on both noise magnitude μ, other noise parameters $\sigma_{\mathbf{x}}^2$ and noisy data $\mathbf{x}_i$, which approximate noise-free (but not observable) values $\bar{\mathbf{x}}_i$. For example, $\mathbf{C}$ for estimating parameters of a quadratic curve will take the form $\mathbf{C}(\mu) = \mu^2\mathbf{C}_2 + \mu\mathbf{C}_1$ where $\mathbf{C}_2$ is a matrix whose entries depend on σ_x^2 and σ_y^2, while those of $\mathbf{C}_1$ depend on σ_x^2 and σ_y^2 as well as $\mathbb{E}\left(x^2\right)$, $\mathbb{E}\left(y^2\right)$, $\mathbb{E}\left(xy\right)$, $\mathbb{E}(x)$ and $\mathbb{E}\left(y\right)$, all of which are approximated from finite samples.

2.1.6.1 Estimating parameters of quadratic curves

In order to illustrate how the procedure is applied to estimating parameters of quadratic curves in two dimensions, let us see how the covariance matrix polynomial $\mathbf{C}(\mu)$ is computed for this special case using the outlined noise cancellation approach. The lifting function for a quadratic curve is

$$\mathbf{f}_{data}(\mathbf{x}) = \begin{bmatrix} x^2 & xy & y^2 & x & y & 1 \end{bmatrix}^\top$$

where $\mathbf{x}_i^\top = \begin{bmatrix} x_i & y_i \end{bmatrix}$ are data points. Let $\overline{x} = \frac{1}{N}\sum_{i=1}^N x_i$ and $\overline{x^2} = \frac{1}{N}\sum_{i=1}^N x_i^2$ (and likewise for $\overline{y}$ and $\overline{y^2}$), and $\overline{xy} = \frac{1}{N}\sum_{i=1}^N x_i x_y - \mathrm{cov}(x,y)$. (Here, the notation $\overline{x}$ stands for average and is not to be confused with $\bar{x}$, which stands for noise-free data corresponding to x. Noise-free values do not occur in the derivation, as they can always be approximated in terms of noisy data.) Using the operations from Table 2.1, we may obtain the noise contribution part of the covariance matrix. With $\sigma_x^2 = \mu\bar{\sigma}_x^2$, $\sigma_y^2 = \mu\bar{\sigma}_y^2$ and $\bar{\sigma}_x^2 + \bar{\sigma}_y^2 = 1$, and the original noise covariance matrix of $\mathbf{x}_i$ having the special structure

$$\mathbf{\Sigma} = \mu \begin{bmatrix} \bar{\sigma}_x^2 & 0 \\ 0 & \bar{\sigma}_y^2 \end{bmatrix}$$

the calculations yield the following matrix polynomial in terms of μ:

$$\begin{aligned} \mathbf{C} \quad &= \mu^2 \begin{bmatrix} 3\bar{\sigma}_x^4 & 0 & \bar{\sigma}_x^2\bar{\sigma}_y^2 & 0 & 0 & 0 \\ 0 & \bar{\sigma}_x^2\bar{\sigma}_y^2 & 0 & 0 & 0 & 0 \\ \bar{\sigma}_x^2\bar{\sigma}_y^2 & 0 & 3\bar{\sigma}_y^4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\ &\quad -\mu \begin{bmatrix} 6\bar{\sigma}_x^2\overline{x}^2 & 3\overline{xy}\bar{\sigma}_x^2 & \overline{x}^2\bar{\sigma}_y^2 + \bar{\sigma}_x^2\overline{y}^2 & 3\overline{x}\bar{\sigma}_x^2 & \overline{y}\bar{\sigma}_x^2 & \bar{\sigma}_x^2 \\ 3\overline{xy}\bar{\sigma}_x^2 & \overline{x}^2\bar{\sigma}_y^2 + \bar{\sigma}_x^2\overline{y}^2 & 3\overline{xy}\bar{\sigma}_y^2 & \overline{y}\bar{\sigma}_x^2 & \overline{x}\bar{\sigma}_y^2 & 0 \\ \overline{x}^2\bar{\sigma}_y^2 + \bar{\sigma}_x^2\overline{y}^2 & 3\overline{xy}\bar{\sigma}_y^2 & 6\bar{\sigma}_y^2\overline{y}^2 & \overline{x}\bar{\sigma}_y^2 & 3\overline{y}\bar{\sigma}_y^2 & \bar{\sigma}_y^2 \\ 3\overline{x}\bar{\sigma}_x^2 & \overline{y}\bar{\sigma}_x^2 & \overline{x}\bar{\sigma}_y^2 & \bar{\sigma}_x^2 & 0 & 0 \\ \overline{y}\bar{\sigma}_x^2 & \overline{x}\bar{\sigma}_y^2 & 3\overline{y}\bar{\sigma}_y^2 & 0 & \bar{\sigma}_y^2 & 0 \\ \bar{\sigma}_x^2 & 0 & \bar{\sigma}_y^2 & 0 & 0 & 0 \end{bmatrix} \end{aligned}$$

Once the covariance matrix polynomial is at our disposal, the problem reduces to finding a solution to a special case of a polynomial eigenvalue problem called a quadratic eigenvalue problem (QEP)

$$\mathbf{g}^\top \big(\mathbf{D} - \mathbf{C}(\mu)\big)\mathbf{g} = \mathbf{g}^\top \big(\mathbf{D} - \mu\mathbf{C}_1 - \mu^2\mathbf{C}_2\big)\mathbf{g} = 0$$

which involves a $\mathbf{C}(\mu)$ with at most a quadratic dependence on μ and where the (real) eigenvalue closest to 0 and the corresponding eigenvector is the solution we seek.

One way to solve the QEP

$$\mathbf{\Psi}(\mu)\mathbf{g} = \big(\mathbf{D} - \mu\mathbf{C}_1 - \mu^2\mathbf{C}_2\big)\mathbf{g} = \mathbf{0}$$

is to apply linearization, thereby eliminating the polynomial dependence on μ at the expense of increasing the size of coefficient matrices, which is analogous to companion matrices constructed from polynomials where the eigenvector of the companion matrix yields the roots

of the polynomial. In particular, transformations that preserve symmetry are especially favored for their numerical stability.

A well-known result [63] for linearizing the QEP

$$\mu^2\mathbf{R}_2 + \mu\mathbf{R}_1 + \mathbf{R}_0$$

is with the first companion form

$$\Xi(\lambda) = \Xi_1 - \lambda\Xi_2 = \begin{bmatrix} \mathbf{0} & \mathbf{W} \\ -\mathbf{R}_0 & -\mathbf{R}_1 \end{bmatrix} - \lambda \begin{bmatrix} \mathbf{W} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_2 \end{bmatrix} \tag{2.17}$$

where the choice for the (arbitrary full-rank) $\mathbf{W} = -\mathbf{R}_0$ yields a generalized eigenvalue problem with symmetric matrices where the eigenvector $\mathbf{w}$ has a special structure

$$\mathbf{w} = \begin{bmatrix} \mathbf{v} \\ \lambda\mathbf{v} \end{bmatrix}.$$

In our case of quadratic curves and surfaces, the substitutions are

$$\begin{aligned} \mathbf{R}_0 &= \mathbf{D} \\ \mathbf{R}_1 &= \mathbf{C}_1 \\ \mathbf{R}_2 &= \mathbf{C}_2 \end{aligned}$$

Solving the resulting generalized eigenvalue problem is the linearized equivalent of solving the original polynomial eigenvalue problem. As the linearized problem has eigenvectors $\mathbf{w}$ of dimension mp rather than m, the true polynomial eigenvector that belongs to the eigenvalue μ becomes the column $\mathbf{v}$ of $\mathrm{vec}\mathbf{V} = \mathbf{w}$ of the linearized eigenvector $\Xi_1\mathbf{w} = \lambda\Xi_2\mathbf{w}$ that gives the smallest normalized residual, i.e.

$$\mathbf{v} = \arg\min_{\mathbf{u}} \frac{\sum_k \left|\left[\boldsymbol{\Psi}(\mu)\mathbf{u}\right]_k\right|}{\sum_k |[\mathbf{u}]_k|}$$

where $[\mathbf{u}]_k$ is the kth component of the vector $\mathbf{u}$.

Example 4. Comparing the accuracy of the nonlinear Koopmans estimator to other estimators. Figure 2.3 compares three non-iterative estimation methods: direct ellipse fit [26] (shown with dotted line), hyper-accurate ellipse fit [39] (shown with dashed line), and the nonlinear Koopmans estimator (shown with dash dot line). Direct ellipse fit is a non-iterative estimation method based on the nonlinear least-squares approach, discussed in Section 2.1.3. Hyper-accurate ellipse fit, as outlined in Section 2.1.5, is a non-iterative method that completely cancels second-order error and is therefore highly accurate.

In Figure 2.3, 250 original data points sampled evenly along a high-curvature arc of an ellipse (continuous line) are contaminated with Gaussian noise of $\sigma_{\mathbf{x}} = 0.1$ (plotted as black dots). The original ellipse that has generated the data points has center $(12, 13)$, semi-axes with lengths of 4 and 2, and angle of tilt $\frac{\pi}{6}$. The rotation and translation are present to illustrate that the fitting algorithms are not sensitive to such geometric transformations in the plane. The points are limited to an arc of the ellipse because all fitting algorithms give very

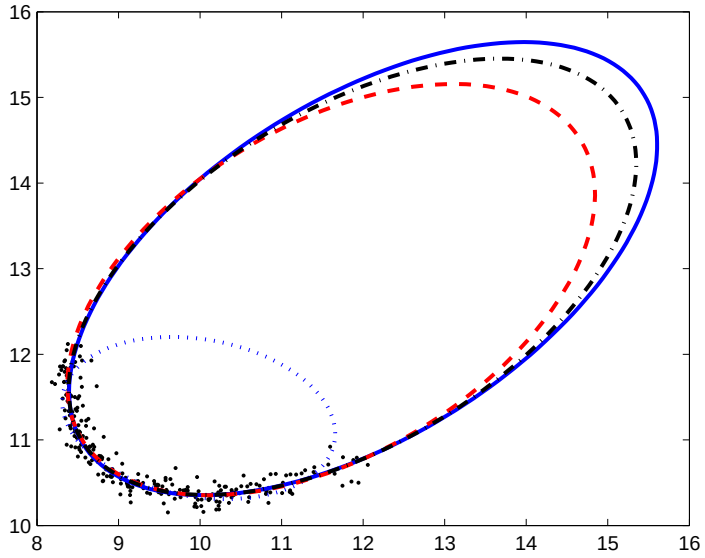


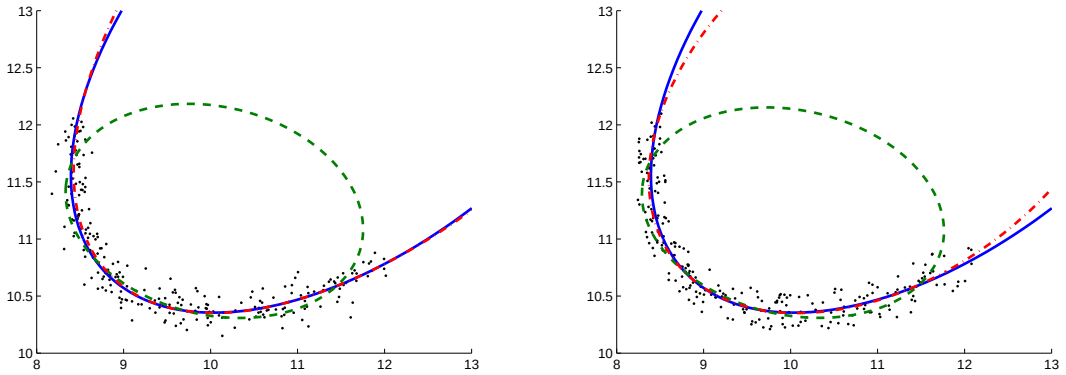
Figure 2.3: Comparing the accuracy of direct ellipse fit (dotted line), hyper-accurate ellipse fit (dashed line) and the estimation method with noise cancellation (dash dot line) given noisy data points (shown as dots).

similar results when the points are sampled along the entire ellipse, which would not lend itself to meaningful comparison.

By comparing the outcome of each method, it can be seen how the nonlinear Koopmans estimator can achieve an accuracy strikingly similar to that of hyper-accurate ellipse fit. The difference in fitting accuracy is not significant. Both algorithms are equally close to the original curve from which data has been sampled, whereas they far outperform direct ellipse fit, which is based on traditional least squares and exhibits a significant low-eccentricity bias. On the other hand, the nonlinear Koopmans fit amounts to solving a generalized eigenvalue problem whereas hyper-accurate fitting requires the computation of several matrix pseudo-inverses in addition. This means that the nonlinear Koopmans fit incurs a low computational cost yet it produces accurate estimates. ♣

In deriving the noise covariance matrix polynomial $\mathbf{C}(\mu)$, we have assumed Gaussian noise distribution. Despite the normality assumption, the approach is applicable to cases for symmetric but non-normal noise distributions.

Example 5. Comparing estimates for quadratic curves obtained from data contaminated with various types of noise. Figure 2.4 shows quadratic curve estimates obtained from a data set (1) contaminated with Gaussian white noise (Figure 2.4a) and (2) contaminated with uniform noise with the same standard deviation as the normal noise (Figure 2.4b). Even though the noise covariance matrix polynomial has been computed based on an assumption of normal noise, the results appear to be valid even if the underlying noise distribution is not normal. ♣



(a) normal noise **(b)** uniform noise

Figure 2.4: Comparing the accuracy of the nonlinear Koopmans estimator (dash dot line) and the direct fit least-squares estimator (dashed line) to the original curve (continuous line) under various noise conditions.

2.1.6.2 Eliminating the constant term

In the case of fitting quadratic curves, we have seen that the data vector subject to the lifting function

$$\mathbf{z}^\top = \begin{bmatrix} x^2 & xy & y^2 & x & y & 1 \end{bmatrix}$$

has the constant term 1 that corresponds to the last entry of the model parameter vector

$$\mathbf{g}^\top = \begin{bmatrix} a & b & c & p & q & d \end{bmatrix}.$$

giving rise to a sample data matrix

$$\mathbf{Z} = \begin{bmatrix} x_1^2 & x_1 y_1 & y_1^2 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_i^2 & x_i y_i & y_i^2 & x_i & y_i & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N^2 & x_N y_N & y_N^2 & x_N & y_N & 1 \end{bmatrix}$$

that has a last column of the constant 1, and likewise a sample data covariance matrix

$$\mathbf{D} = \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{z}_i - \frac{1}{N}\sum_{j=1}^{N}\mathbf{z}_j\right)\left(\mathbf{z}_i - \frac{1}{N}\sum_{j=1}^{N}\mathbf{z}_j\right)^\top$$

that has a last column and row that are statistically invariant [31]. (In a similar fashion, the bottom right entry of the noise covariance matrix polynomial $\mathbf{C}(\mu)$ is identically zero, and the matrix is rank-deficient.)

In fact, when we seek a linear fit $\mathbf{z}^\top\mathbf{g} = 0$ to the data matrix $\mathbf{Z}$, we operate in the space of $\begin{bmatrix} x^2 & xy & y^2 & x & y & 1 \end{bmatrix}$ and the parameters $\mathbf{g}$ define a hyperplane. However, subtracting

the mean vector

$$\begin{aligned} m &= \frac{1}{N}\begin{bmatrix} \sum_i x_i^2 & \sum_i x_i y_i & \sum_i y_i^2 & \sum_i x_i & \sum_i y_i & N \end{bmatrix} \\ &= \begin{bmatrix} m_{x^2} & m_{xy} & m_{y^2} & m_x & m_y & 1 \end{bmatrix} \end{aligned} \tag{2.18}$$

from the rows of the matrix $\mathbf{Z}$, the problem reduces to a fitting problem where the hyperplane passes through the origin, and the last component in $\mathbf{g}$ becomes redundant. This means we may partition the sample covariance matrix $\mathbf{D}$ such that

$$\mathbf{D} = \frac{1}{N}\mathbf{Z}^\top\mathbf{Z} = \begin{bmatrix} \mathbf{D}_\star & \mathbf{d}_\star \\ \mathbf{d}_\star^\top & d_0 \end{bmatrix} \tag{2.19}$$

where $\mathbf{D}_\star$ stands for the purely non-constant (quadratic and linear) terms, the entries in $\mathbf{d}_\star$ are mixed constant vs. non-constant terms, and $d_0 = 1$. This partitioning captures the different nature of quadratic and linear data, and the statistically invariant terms. Likewise, we may partition the model parameter vector such that

$$\mathbf{g}^\top = \begin{bmatrix} \mathbf{g}_\star^\top & g_0 \end{bmatrix}.$$

From the construction, we have

$$g_0 = -\mathbf{d}_\star^\top\mathbf{g}_\star \tag{2.20}$$

which corresponds to

$$\begin{aligned} g_0 &= -\frac{1}{N}\begin{bmatrix} \sum_i x_i^2 & \sum_i x_i y_i & \sum_i y_i^2 & \sum_i x_i & \sum_i y_i \end{bmatrix}\mathbf{g}_\star \\ &= -\begin{bmatrix} m_{x^2} & m_{xy} & m_{y^2} & m_x & m_y \end{bmatrix}\mathbf{g}_\star, \end{aligned}$$

directly stating that the hyperplane fit to the data matrix $\mathbf{Z}$ must pass through the centroid of the points, effectively forcing the hyperplane through the centroid of the data. This transformation not only ensures the Euclidean invariance of the fit, but also reduces the dimensionality of the problem. With mean subtraction, the problem is readily reduced to determining the orientation of the hyperplane to be fit, as the relative translation is now known. Once a fit in the reduced space is at our disposal, we can easily recover the original parameters.

Exploiting the decomposition in (2.19) and the corresponding parameter reduction in (2.20) on the original sample covariance matrix $\mathbf{D}$, we can write the matrix polynomial

$$\mathbf{\Psi}(\mu) = \mathbf{D} - \mathbf{C}(\mu) = \mathbf{D} - \mu\mathbf{C}_1 - \mu^2\mathbf{C}_2$$

as

$$\mathbf{\Psi}(\mu) = \begin{bmatrix} \mathbf{D}_\star & \mathbf{d}_\star \\ \mathbf{d}_\star^\top & 1 \end{bmatrix} - \mu\begin{bmatrix} \mathbf{C}_\star & \mathbf{c}_\star \\ \mathbf{c}_\star^\top & 0 \end{bmatrix} - \mu^2\begin{bmatrix} \mathbf{C}_{\star\star} & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{bmatrix}$$

where the bottom right entry of $\mathbf{C}(\mu)$ is zero as it corresponds to the constant (and thus noise-free) term. Writing the Schur complement of the bottom right entry of $\mathbf{\Psi}(\mu)$, we get the rank-one updated

$$\mathbf{\Psi}_\star(\mu) = \mathbf{D}_\star - \mu\mathbf{C}_\star - \mu^2\mathbf{C}_{\star\star} - \left(\mathbf{d}_\star - \mu\mathbf{c}_\star\right)\left(\mathbf{d}_\star - \mu\mathbf{c}_\star\right)^\top \tag{2.21}$$

which is a polynomial in μ of at most degree 2 and we seek

$$\det\left(\boldsymbol{\Psi}_\star(\mu)\right) = \det\left(\mathbf{M}_0 - \mu\mathbf{M}_1 - \mu^2\mathbf{M}_2\right) = 0. \tag{2.22}$$

One way to find the eigenvalue μ in (2.22) is to apply the first companion form linearization (2.17). In most applications, however, the value of $\mu^2\mathbf{M}_2$ is small, and may be ignored without introducing excessive numerical error, leading to a standard generalized eigenvalue problem

$$\boldsymbol{\Psi}_\star(\mu) \approx \mathbf{M}_0 - \mu\mathbf{M}_1.$$

Furthermore, when we operate on the matrix pair $(\mathbf{M}_0, \mathbf{M}_1)$, we may altogether avoid constructing

$$\mathbf{M}_0 = \mathbf{D}_\star - \mathbf{d}_\star\mathbf{d}_\star^\top$$

where $\mathbf{M}_0$ is the mean-free equivalent of the sample data covariance matrix $\mathbf{D}$, and use the pair $(\mathbf{Z}_m, \mathbf{L}_1)$ instead where

$$\mathbf{Z}_m = \begin{bmatrix} x_1^2 - m_{x^2} & x_1y_1 - m_{xy} & y_1^2 - m_{y^2} & x_1 - m_x & y_1 - m_y \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_i^2 - m_{x^2} & x_iy_i - m_{xy} & y_i^2 - m_{y^2} & x_i - m_x & y_i - m_y \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N^2 - m_{x^2} & x_Ny_N - m_{xy} & y_N^2 - m_{y^2} & x_N - m_x & y_N - m_y \end{bmatrix}$$

is a mean-free sample data matrix in which the values m_{x^2}, m_{xy}, m_{y^2}, m_x and m_y are from the mean vector m in (2.18), $\mathbf{L}_1^\top\mathbf{L}_1 = \mathbf{M}_1$ and $\mathbf{L}_1$ may be found by Cholesky decomposition of the positive definite matrix $\mathbf{M}_1$. This allows us to employ singular value decomposition of $(\mathbf{Z}_m, \mathbf{L}_1)$ instead, with improved numerical accuracy and lower computational cost.

2.1.7 Accuracy analysis

Having discussed several estimation methods, some of which are iterative, some of which are non-iterative, it is natural to ask what the price is that we pay in accuracy for the lower computational cost. A theoretical accuracy bound called the Kanatani–Cramer–Rao (KCR) lower bound [38] provides a means to assess the absolute and relative accuracy of various methods.

Among fitting algorithms, methods based on maximum likelihood (ML) are regarded as the most accurate, and they attain the KCR lower bound up to high-order error terms [37]. The KCR lower bound is analogous to the Cramer–Rao lower bound in the traditional domain of statistics, which is constructed by evaluating and inverting the *Fisher information matrix*, and eliminating so-called *nuisance parameters*. However, the size of the matrices involved is large, making the inversion analytically almost intractable. Nuisance parameters arise from the fact that traditional statistics deals with data $\mathbf{x}_i = \bar{\mathbf{x}}_i + \tilde{\mathbf{x}}_i$ where $\bar{\mathbf{x}}_i$ are are unknown and $\tilde{\mathbf{x}}_i$ are drawn from a probability distribution but the true parameters $\mathbf{g}$ we are truly interested in are not present, only indirectly via the implicit relationship $f(\bar{\mathbf{x}}_i, \mathbf{g})$. The KCR bound operates in a "dual framework" (as how [38] refers to it) where only parameters $\mathbf{g}$ of interest are present, not nuisance parameters $\bar{\mathbf{x}}_i = \bar{\mathbf{x}}_i(\mathbf{g})$, leading to a much simpler expression.

The KCR lower bound $\mathbf{C}_{KCR}(\hat{\mathbf{g}})$ on the variance of the parameter estimates measures the accuracy of the estimator, i.e. an arbitrary unbiased estimator $\hat{\mathbf{g}}$ of $\mathbf{g}$ has

$$\mathbf{C}(\hat{\mathbf{g}}) \succeq \mathbf{C}_{KCR}(\hat{\mathbf{g}}) \tag{2.23}$$

where $\mathbf{A} \succeq \mathbf{B}$ means that the left-hand side $\mathbf{A}$ minus the right $\mathbf{B}$ is positive semi-definite, with [15, 13]

$$\mathbf{C}_{KCR}(\hat{\mathbf{g}}) = \sigma^2 \left(\sum_{i=1}^{N} \frac{\left(\nabla_{\mathbf{g}} f(\bar{\mathbf{x}}_i, \mathbf{g})\right)\left(\nabla_{\mathbf{g}} f(\bar{\mathbf{x}}_i, \mathbf{g})\right)^{\top}}{\left\| \nabla_{\mathbf{x}} f(\bar{\mathbf{x}}_i, \mathbf{g}) \right\|^2} \right)^{\dagger}$$

for noise with equal variance for vector components where σ^2 is the noise variance, $\nabla_{\mathbf{w}} f$ is the gradient of f w.r.t. the vector $\mathbf{w}$ and the superscript $\mathbf{A}^{\dagger}$ denotes the generalized inverse, or with [39]

$$\mathbf{C}_{KCR}(\hat{\mathbf{g}}) = \left(\sum_{i=1}^{N} \frac{\bar{\mathbf{x}}_i \bar{\mathbf{x}}_i^{\top}}{\mathbf{g}^{\top} \mathbf{C}(\mathbf{f}_{data}(\mathbf{x}_i))\, \mathbf{g}} \right)^{\dagger} \tag{2.24}$$

where $\mathbf{C}(\mathbf{f}_{data}(\mathbf{x}_i))$ is the covariance matrix of $\mathbf{x}_i$ subject to the lifting function $\mathbf{f}_{data}$. In (2.24), we have assumed a homoskedastic Gaussian noise over the independent random variables $\mathbf{x}_i$ as well as $f(\mathbf{x}, \mathbf{g})$ being linear in $\mathbf{g}$.

2.2 Constrained fitting

We have so far seen how the accuracy of fitting algebraic distance

$$e = \frac{1}{N} \sum_{i=1}^{N} \left(f(\mathbf{x}_i, \mathbf{g}) \right)^2$$

where we measure the substitution error of noisy points plugged into the curve or surface equation, can be improved with a noise cancellation scheme and how the scheme helps the algebraic distance approximate geometric distance

$$e = \frac{1}{N} \sum_{i=1}^{N} d_i^2$$

where d_i measures the distance from the noisy point $\mathbf{x}_i$ to the curve or surface $f(\mathbf{x}, \mathbf{g}) = 0$. The presented non-iterative approach combines relatively low computational complexity with relatively high fitting accuracy, and requires no hint for initialization, which eliminates slow convergence (or divergence) experienced with methods inspired by minimizing geometric distance.

However, in many cases we are interested in fitting a quadratic curve subject to constraints [56], i.e. estimating the parameter $\mathbf{g}$ such that it represents a particular class of quadratic curves or surfaces such as ellipses or ellipsoids. This is especially important when data are measured with substantial noise, and noisy data points may guide the estimator to a fit that is otherwise not permitted in light of prior information.

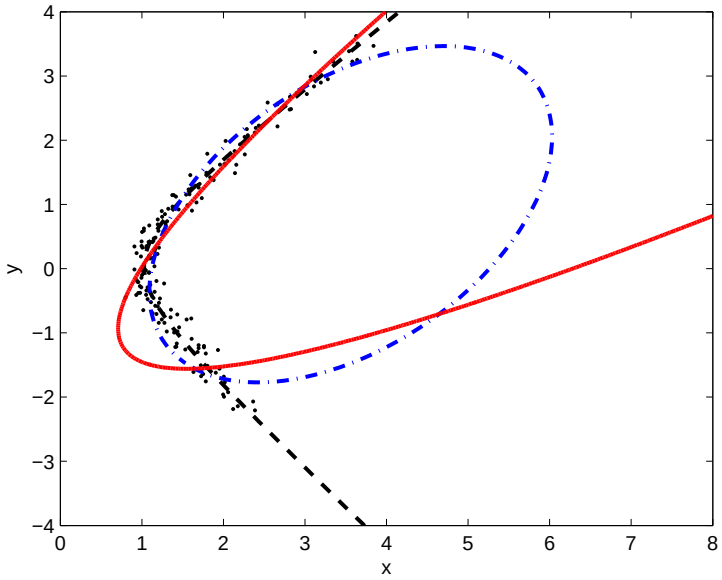


Figure 2.5: Data suggesting a hyperbola fitted without constraints (dashed), with ellipse-specific constraints (dash dot line) and parabola-specific constraints (continuous line).

Example 6. Constrained fitting. Figure 2.5 shows how constrained fitting always fits ellipses (or parabolas) even if data would suggest another quadratic curve. This is essential when the the noise contaminating the data has a large magnitude, which may destabilize the algorithm unless the a priori information is incorporated, and we expect the algorithm to pull the fit towards the desired state rather than provide a more accurate fit that diverges from the desired state. The dashed line in Figure 2.5 shows the nonlinear Koopmans fit (consistent algebraic least squares fit) without constraints while the dash dot line and the continuous line shows the fit incorporating ellipse constraints and parabola constraints, respectively. ♣

A straightforward approach to the constrained estimation problem is to formulate it in such a manner that the solution guarantees curves of a particular type. In the setting of quadratic curves, an ellipse can be parametrized in one of the following manners:

$$\begin{aligned} P(c_x, c_y, a, b, \phi) &= b^2\left((x-c_x)\cos\phi - (y-c_y)\sin\phi\right) + \\ &+ a^2\left((x-c_x)\sin\phi + (y-c_y)\cos\phi\right) - a^2b^2 \end{aligned} \tag{2.25}$$

$$P(p_x, p_y, q_x, q_y, a) = \sqrt{\left(x-p_x\right)^2 + \left(y-p_y\right)^2} + \sqrt{\left(x-q_x\right)^2 + \left(y-q_y\right)^2} - 2a \tag{2.26}$$

$$\begin{aligned} P(\mathbf{g}) &= \mathbf{g}^\top \begin{bmatrix} x^2 & xy & y & x & y & 1 \end{bmatrix} \\ &= ax^2 + bxy + cy^2 + dx + fy + g \end{aligned} \tag{2.27}$$

where $\mathbf{c}^\top = \begin{bmatrix} c_x & c_y \end{bmatrix}$ is the center of the ellipse, $\mathbf{p}^\top = \begin{bmatrix} p_x & p_y \end{bmatrix}$ and $\mathbf{q}^\top = \begin{bmatrix} q_x & q_y \end{bmatrix}$ are the foci of the ellipse, $a > 0$ and $b > 0$ are the semi-major and semi-minor axis lengths, respectively, and $\phi \in (-\frac{1}{2}\pi; \frac{1}{2}\pi]$ is the angle of tilt enclosed with the horizontal and

$$b^2 = a^2 - \frac{1}{4}\left\{\left(q_x - p_x\right)^2 + \left(q_y - p_y\right)^2\right\} > 0.$$

Among the parametrizations above, (2.25) is the standard parametrization, (2.26) is using Kepler's parameters of the ellipse and (2.27) is the general quadratic curve equation. (2.25) and (2.26) take five parameters, whereas (2.27) takes six parameters but with the constraint $\|\mathbf{g}\| = 1$. In all parametrizations, $P(\circ) > 0$ is the outside of the ellipse and $P(\circ) < 0$ is the inside.

Ellipses expressed in the form (2.25) and (2.26) can always be converted to the general quadratic curve equation (2.27) but if (2.27) represents an ellipse, it can be expressed in the other two forms as well. For example, let

$$\begin{aligned} \mathbf{g}^\top \begin{bmatrix} x^2 & xy & y & x & y & 1 \end{bmatrix} &= Ax^2 + Bxy + Cy^2 + Dx + Fy + G \\ M &= 2(AF^2 + CD^2 + GB^2 - 2BDF - ACG) \\ N &= B^2 - AC \\ S &= \sqrt{(A-C)^2 + 4B^2} \end{aligned}$$

then with

$$a_p = \sqrt{\frac{M}{NS - N(A+C)}} \qquad b_p = \sqrt{\frac{M}{-NS - N(A+C)}}$$

we have for the semi-axes

$$a = \max\{a_p, b_p\} \qquad b = \min\{a_p, b_p\}.$$

Likewise, we have for the center

$$c_x = \frac{CD - BF}{N} \qquad c_y = \frac{AF - BD}{N}$$

and for the angle of tilt

$$\phi = \begin{cases} \frac{1}{2}\cot^{-1}\left(\frac{A-C}{2B}\right) & \text{if } |A| < |C| \text{ and } B \neq 0 \\ \frac{1}{2}\pi + \frac{1}{2}\cot^{-1}\left(\frac{A-C}{2B}\right) & \text{if } |A| > |C| \text{ and } B \neq 0 \\ 0 & \text{if } |A| < |C| \text{ and } B = 0 \\ \frac{1}{2}\pi & \text{if } |A| > |C| \text{ and } B = 0. \end{cases}$$

All but (2.27) guarantee that the curve we fit is an ellipse. However, all but (2.27) have complicated implications when data are measured with noise, and the underlying true value of the measured data is not available. The nonlinear perturbations induced by noise are difficult to trace back across functions $\sin(\bullet)$, $\cos(\bullet)$ or $\sqrt{\bullet}$. In contrast, the general equation (2.27) lends itself to an intuitive noise cancellation scheme.

Let us re-visit traditional least squares, which solves the eigenvalue problem

$$\mathbf{Dg} = \lambda \mathbf{g}$$

for the minimum eigenvalue λ where $\mathbf{D}$ is the sample data covariance matrix. This is an unconstrained fit, which fits lines, ellipses, parabola, hyperbola, etc. in 2D, without imposing restriction on the class of curves. Even if we have prior information on the curve we are interested in, this is not taken into account by the algorithm. If minimum algebraic fit is achieved by a hyperbola rather than an ellipse, the least-squares fit returns a hyperbola, even

if we seek an ellipse. For rank-deficient cases, such as parabolas, it is practically impossible to recover the original shape, if the matrix $\mathbf{D}$ is built from data contaminated with noise, the estimator produces either an ellipse or a hyperbola.

In order to remedy the problem, an extension to least squares, called direct least-squares fitting of ellipses [26], uses a constraint (normalization matrix) $\mathbf{Q}$ to represent the constraint

$$\mathbf{g}^\top \mathbf{Q}\mathbf{g} > 0$$

and solves the eigenvalue problem

$$\mathbf{D}\mathbf{g} = \lambda \mathbf{Q}\mathbf{g}$$

for the only positive eigenvalue λ. In contrast to the least-squares fit, direct ellipse fit ensures an ellipse, even if data might suggest otherwise, increasing the robustness of the fit. However, direct ellipse fit is heavily biased towards low eccentricity, and exhibits huge distortion at higher levels of noise.

In an effort to combine the accuracy gain from noise cancellation and the robustness increase from imposing constraints, the proposed estimation approach comprises of the following consecutive steps:

1. noise cancellation
2. constrained quadratic least-squares fitting

This approach guarantees consistent estimation (step 1) while enforcing constraints even if data would suggest a curve or surface other than that to be fitted (step 2). The noise cancellation scheme (step 1) has been covered in depth in Section 2.1.6 where the problem has been shown to lead to a polynomial eigenvalue problem

$$\mathbf{\Psi}(\mu)\mathbf{g} = \left(\mathbf{D} - \mu\mathbf{C}_1 - \mu^2\mathbf{C}_2\right)\mathbf{g} = \mathbf{0}$$

which we can solve for the noise magnitude μ. With the notation

$$\mathbf{R} = \mathbf{\Psi}(\hat{\mu})$$

where $\hat{\mu}$ solves $\mathbf{\Psi}(\mu)\mathbf{g}$, our focus is now on constrained fitting (step 2), where the nonlinear perturbations by noise are no longer present in $\mathbf{R}$.

2.2.1 Reducing matrix dimension

In a large proportion of practical problems, we are interested in quadratic constraints that cover only a subset of the parameters. For example, confining a curve estimator to produce only ellipses, we have the parameter vector

$$\mathbf{g} = \begin{bmatrix} a & b & c & p & q & d \end{bmatrix}^\top$$

but our constraint involves only the parameters a, b and c (which correspond to quadratic terms x^2, xy and y^2), not the parameters p and q (which correspond to linear terms x and y). In other words, the optimization problem

$$\begin{aligned} \min_{\mathbf{g}} \quad & \mathbf{g}^\top \mathbf{R}\mathbf{g} \\ \text{s.t.} \quad & \mathbf{g}^\top \mathbf{Q}\mathbf{g} = 1 \end{aligned}$$

where the normalization $\mathbf{g}^\top \mathbf{Q}\mathbf{g} = 1$ helps improve numerical stability without compromising the accuracy of estimates ($\mathbf{g}$ is to be computed only up to scale), ultimately leads to the generalized eigenvalue problem

$$\mathbf{R}\mathbf{g} = \lambda \mathbf{Q}\mathbf{g},$$

where most entries in the matrix $\mathbf{Q}$ that represents constraints are zero. Let the vectors and matrices be decomposed as

$$\begin{aligned} \mathbf{g}^\top &= \begin{bmatrix} \mathbf{g}_1^\top & \mathbf{g}_2^\top \end{bmatrix}^\top \\ \mathbf{R} &= \begin{bmatrix} \mathbf{R}_1 & \mathbf{R}_2 \\ \mathbf{R}_2^\top & \mathbf{R}_3 \end{bmatrix} \\ \mathbf{Q} &= \begin{bmatrix} \mathbf{Q}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \end{aligned}$$

where $\mathbf{0}$ denotes a zero matrix, and $\mathbf{Q}_1$ represents the constraint (e.g. that confines the parameters of a quadratic curve to stand for an ellipse or a hyperbola) such that

$$\begin{bmatrix} \mathbf{R}_1 & \mathbf{R}_2 \\ \mathbf{R}_2^\top & \mathbf{R}_3 \end{bmatrix} \begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{Q}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \end{bmatrix}$$

As in [30, 31], let

$$\begin{aligned} \mathbf{T} &= -\mathbf{R}_3^{-1}\mathbf{R}_2^\top \\ \mathbf{S} &= \mathbf{R}_1 + \mathbf{R}_2\mathbf{T} \end{aligned} \tag{2.28}$$

where the inverse $\mathbf{R}_3^{-1}$ exists. (This is usually the case in quadratic curve or surface fitting problems where the inverse always exists unless points are co-linear or co-planar. If $\mathbf{R}_3$ were (exactly or approximately) singular, points $\{x_i, y_i\}$ would lie on a straight line or points $\{x_i, y_i, z_i\}$ would lie on the same plane, in which case fitting a quadratic curve or surface would be meaningless, and a line or plane fitting estimator should be used instead.) Thus,

$$\mathbf{g} = \begin{bmatrix} \mathbf{g}_1 \\ \mathbf{T}\mathbf{g}_1 \end{bmatrix}$$

where we solve the estimation problem by finding the eigenvector $\mathbf{g}_1$ of the reduced scatter matrix $\mathbf{S}$ that satisfies $\mathbf{g}_1^\top \mathbf{Q}_1 \mathbf{g}_1$, i.e. the eigenvector decomposition of

$$\mathbf{S}\mathbf{g}_1 = \lambda \mathbf{Q}_1 \mathbf{g}_1$$

is computed where we choose $\mathbf{g}_1$ such that it produces a $\mathbf{g}_1^\top \mathbf{Q}_1 \mathbf{g}_1$ that is consistent with the quadratic curve or surface to be fitted (e.g. positive for ellipses). In the sections that follow, we shall investigate the choice of $\mathbf{Q}_1$ for various types of quadratic curves and surfaces.

2.2.2 Fitting ellipses and hyperbolas

Fitting ellipses and hyperbolas involves the quadratic constraint $b^2 - 4ac < 0$ for ellipses and $b^2 - 4ac > 0$ for hyperbolas with

$$\mathbf{g} = \begin{bmatrix} a & b & c & p & q & d \end{bmatrix}^\top.$$

Notice that since $\mathbf{g}$ is unique up to scaling by a scalar (d is the coefficient of the constant term), the constraint $4ac - b^2 = \pm 1$ expresses both $b^2 - 4ac < 0$ (the discriminant condition for an ellipse) and $b^2 - 4ac > 0$ (the discriminant condition for hyperbolas) because it is up to us to choose a (negative or positive) scalar. Imposing the constraint $4ac - b^2 = \pm 1$, one can avoid degenerate solutions and ensure that the estimated parameters indeed define an ellipse or hyperbola (depending on sign), invariant to rotations and translations [26]. Written in matrix notation, ellipse and hyperbola fitting require the constraint

$$q = \mathbf{g}^\top \operatorname{diag}\left(\begin{bmatrix} 0 & 0 & 2 \\ 0 & -1 & 0 \\ 2 & 0 & 0 \end{bmatrix}, \mathbf{0}_{3\times 3} \right) \mathbf{g} = \mathbf{g}^\top \mathbf{Q} \mathbf{g} \tag{2.29}$$

where $q > 0$ for fitting ellipses and $q < 0$ for fitting hyperbolas. (Evaluating $\mathbf{g}^\top \mathbf{Q} \mathbf{g}$ we get $4ac - b^2$.) Using the matrix size reduction scheme from Section 2.2.1 we may write the constraint in the equivalent form

$$q_1 = \mathbf{g}_1^\top \begin{bmatrix} 0 & 0 & 2 \\ 0 & -1 & 0 \\ 2 & 0 & 0 \end{bmatrix} \mathbf{g}_1 = \mathbf{g}_1^\top \mathbf{Q}_1 \mathbf{g}_1.$$

Seen as an optimization problem, we may then seek the solution to

$$\begin{aligned} \min_{\mathbf{g}_1} \quad & \mathbf{g}_1^\top \mathbf{S} \mathbf{g}_1 \\ \text{s.t.} \quad & \mathbf{g}_1^\top \mathbf{Q}_1 \mathbf{g}_1 = 1 \end{aligned}$$

where $\mathbf{S}$ is as defined in (2.28).

Introducing the Lagrange multiplier, we have

$$\min_{\mathbf{g}_1, \lambda} \mathbf{g}_1^\top \mathbf{S} \mathbf{g}_1 - \lambda \mathbf{g}_1^\top \mathbf{Q}_1 \mathbf{g}_1$$

where $\lambda \geq 0$. Differentiating w.r.t. the parameters we get a system of equations

$$\begin{aligned} 2\mathbf{S}\mathbf{g}_1 - 2\lambda \mathbf{Q}_1 \mathbf{g}_1 &= \mathbf{0} \\ \mathbf{g}_1^\top \mathbf{Q} \mathbf{g}_1 &= 1 \end{aligned} \tag{2.30}$$

where we obtain $\mathbf{g}_1$ as the solution to the generalized eigenvector problem

$$\mathbf{S}\mathbf{g}_1 = \lambda \mathbf{Q}_1 \mathbf{g}_1. \tag{2.31}$$

Suppose $\boldsymbol{\xi}$ is a solution to the eigenvector problem, then so is $\kappa\boldsymbol{\xi}$. We can choose κ to satisfy the constraint, hence

$$\begin{aligned} \kappa^2 \boldsymbol{\xi}^\top \mathbf{Q}_1 \boldsymbol{\xi} &= 1 \\ \kappa &= \sqrt{\frac{1}{\boldsymbol{\xi}^\top \mathbf{Q}_1 \boldsymbol{\xi}}} \end{aligned}$$

and thus

$$\hat{\mathbf{g}}_1 = \kappa\boldsymbol{\xi}$$

solves the system (2.30).

From the above it follows that for ellipse-specific fitting [26], the solution $\mathbf{g}_1$ (and its expanded version $\mathbf{g}$) corresponds ideally (in the absence of numerical errors) to the smallest positive eigenvalue λ of the eigenvalue problem (2.31) with the constraint $\mathbf{g}_1^\top\mathbf{Q}_1\mathbf{g}_1 = 1$ (and its expanded form with the constraint $\mathbf{g}^\top\mathbf{Q}\mathbf{g} = 1$), whereas for hyperbola-specific fitting [53], the best choice of $\mathbf{g}_1$ (and its expanded version $\mathbf{g}$) among the eigenvectors is found by back-substitution into (2.29).

Summarizing, the generalized eigenvalue problem (2.31) has three valid solutions with respect to the constraints: one elliptical and two hyperbolic solutions to the conic fitting problem, depending on the eigenvalue λ. There is a symmetry between only one of the hyperbolic solutions and the elliptical solution, which enables us to uniquely identify the correct solutions [53].

Lemma 1. *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be symmetric square matrices, with* $\mathbf{A}$ *being positive definite. The signs of the generalized eigenvalues of*

$$\mathbf{A}\boldsymbol{\xi} = \lambda\mathbf{B}\boldsymbol{\xi} \tag{2.32}$$

are the same as those of the matrix $\mathbf{B}$*, up to a permutation of the indices.*

Proof. The proof is adopted from [55]. Let us define the spectrum $\sigma(\mathbf{A})$ as the set of eigenvalues of $\mathbf{A}$ and let $\sigma(\mathbf{A}, \mathbf{B})$ be the set of eigenvalues of the matrix pair $(\mathbf{A}, \mathbf{B})$ in (2.32). Let the inertia $i(\mathbf{B})$ be defined as the set of signs of $\sigma(\mathbf{B})$ and let $i(\mathbf{A}, \mathbf{B})$ be the inertia of $\sigma(\mathbf{A}, \mathbf{B})$. With these notations, we have to prove that $i(\mathbf{A}, \mathbf{B}) = i(\mathbf{B})$. As $\mathbf{A}$ is positive definite, it may be decomposed as $\mathbf{W}^2$ for symmetric $\mathbf{W}$, allowing us to write (2.32) as

$$\mathbf{W}^2\boldsymbol{\xi} = \lambda\mathbf{B}\boldsymbol{\xi}.$$

Substituting $\boldsymbol{\vartheta} = \mathbf{W}\boldsymbol{\xi}$ and multiplying by $\mathbf{W}^{-1}$ from the left, we get

$$\boldsymbol{\vartheta} = \lambda\mathbf{W}^{-1}\mathbf{B}\mathbf{W}^{-1}\boldsymbol{\vartheta}$$

so that $\sigma(\mathbf{A}, \mathbf{B}) = \sigma\left(\mathbf{W}^{-1}\mathbf{B}\mathbf{W}^{-1}\right)$ and thus $i(\mathbf{A}, \mathbf{B}) = i\left(\mathbf{W}^{-1}\mathbf{B}\mathbf{W}^{-1}\right)$. From Sylvester's law of inertia [28], we have that for any symmetric $\mathbf{A}$ and nonsingular $\mathbf{X}$, $i(\mathbf{A}) = i\left(\mathbf{X}^\top\mathbf{A}\mathbf{X}\right)$. Therefore, substituting $\mathbf{X} = \mathbf{X}^\top = \mathbf{W}^{-1}$ we have $i(\mathbf{B}) = i\left(\mathbf{W}^{-1}\mathbf{B}\mathbf{W}^{-1}\right) = i(\mathbf{A}, \mathbf{B})$. □

Lemma 2. *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be symmetric square matrices, with* $\mathbf{A}$ *being positive definite. If* λ_k *and the corresponding* $\boldsymbol{\xi}_k$ *are a solution to the eigensystem*

$$\mathbf{A}\boldsymbol{\xi} = \lambda\mathbf{B}\boldsymbol{\xi}$$

then the sign of λ_k *matches the sign of* $\boldsymbol{\xi}_k^\top\mathbf{B}\boldsymbol{\xi}_k$*.*

Proof. The proof is adopted from [55]. Multiplying both sides of (2.32) with $\boldsymbol{\xi}_k^\top$ from the left we have

$$\boldsymbol{\xi}_k^\top\mathbf{A}\boldsymbol{\xi}_k = \lambda\boldsymbol{\xi}_k^\top\mathbf{B}\boldsymbol{\xi}_k^\top.$$

Since $\mathbf{A}$ is positive definite, $\boldsymbol{\xi}_k^\top\mathbf{A}\boldsymbol{\xi}_k > 0$ and therefore λ_i and the scalar $\boldsymbol{\xi}_k^\top\mathbf{B}\boldsymbol{\xi}_k^\top$ must have the same sign. □

Theorem 3. *The solution of the conic fitting problem defined by the generalized eigensystem*

$$\mathbf{S}\mathbf{g} = \lambda\mathbf{Q}\mathbf{g}$$

subject to the constraint matrix

$$\mathbf{Q} = \begin{bmatrix} 0 & 0 & -2 \\ 0 & 1 & 0 \\ -2 & 0 & 0 \end{bmatrix}$$

defining the constraint $b^2 - 4ac < 0$ delivers three nontrivial solutions: one and only one elliptical and two hyperbolic solutions.[2]

Proof. Since the (non-zero) eigenvalues of $\mathbf{Q}$ are -2, 1 and 2, from Lemma 1 we have that there is one and only one negative eigenvalue and two positive eigenvalues, associated with the solutions $\mathbf{g}_k$. Since the equations are homogeneous, both the positive and negative eigenvalues correspond to eigenvectors that are valid solutions. Then according to Lemma 2, there are two cases: (1) for a negative eigenvalue the constraint $\mathbf{g}_k^\top \mathbf{Q}\mathbf{g}_k = b^2 - 4ac$ is negative, and $\mathbf{g}_k$ is a set of coefficients representing an ellipse; (2) for positive eigenvalues, the constraint $\mathbf{g}_k^\top \mathbf{Q}\mathbf{g}_k = b^2 - 4ac$ is positive, and $\mathbf{g}_k$ is a set of coefficients representing a hyperbola; there are two such solutions. From a geometric perspective, there are two hyperbolic solutions since the constraint only defines that two pairs of real asymptotes must exist. The solution can never produce a parabola, since this would require $\mathbf{g}_k^\top \mathbf{Q}\mathbf{g}_k = 0$, which is not permitted by problem formulation. □

Thus, there are two symmetric eigenvalues of the constraint matrix $\lambda_{k_1} = -2$ and $\lambda_{k_2} = 2$; these correspond to the optimal elliptical and hyperbolic solutions, respectively. The third eigenvalue $\lambda_{k_3} = 1$ also represents a hyperbola, however, it does not minimize the error as desired, but maximizes it. This means that the eigenvector $\mathbf{g}_{k_1}$ associated with the negative constraint coefficient

$$\kappa_{k_1} = b_{k_1}^2 - 4a_{k_1}c_{k_1}$$

is the best elliptical fit, while the eigenvector $\mathbf{g}_{k_2}$ associated with the positive constraint coefficient

$$\kappa_{k_2} = b_{k_2}^2 - 4a_{k_2}c_{k_2}$$

with the value nearest to that of the elliptical solution κ_{k_1} is the best hyperbolic fit.

2.2.3 Fitting parabolas

Unfortunately, parabola fitting cannot directly utilize the ellipse and hyperbola constraint (2.29). Fitting a parabola would require $q = 0$ but this would lead to a trivial solution. Instead, as in [31], a regular eigenvector decomposition of

$$\mathbf{S}_1\mathbf{g}_1 = \lambda\mathbf{g}_1$$

[2] For this theorem, the subscript $\bullet_1$ has been dropped from $\mathbf{Q}_1$ and $\mathbf{g}_1$. However, they refer to the reduced matrix and vector, respectively.

is computed which has the incorporated implicit constraint $\mathbf{Q} = \mathbf{I}$, confining the estimation to produce an unspecified but quadratic curve, and the parabola constraint is enforced explicitly rather than via the eigenvalue problem. Given the decomposition of $\mathbf{S}$, the solution we seek can be written as a linear combination of eigenvectors

$$\mathbf{g}_1 = \mathbf{v}_1 + s\mathbf{v}_2 + t\mathbf{v}_3$$

where s and t are scalars, and the eigenvectors $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$, with the corresponding eigenvalues λ_1, λ_2 and λ_3, form an orthonormal basis vector set for the space of the coefficients of the quadratic terms of all conics. Then we seek $\mathbf{g}_1$ with the minimum norm

$$\begin{aligned}
\mathscr{F}(s,t) &= \mathbf{g}_1^\top \mathbf{S}^\top \mathbf{S} \mathbf{g}_1 \\
&= (\mathbf{v}_1 + s\mathbf{v}_2 + t\mathbf{v}_3)^\top \mathbf{S}^\top \mathbf{S} (\mathbf{v}_1 + s\mathbf{v}_2 + t\mathbf{v}_3) \\
&= \mathbf{v}_1^\top \mathbf{S}^\top \mathbf{S} \mathbf{v}_1 + s^2 \mathbf{v}_2^\top \mathbf{S}^\top \mathbf{S} \mathbf{v}_2 + t^2 \mathbf{v}_3^\top \mathbf{S}^\top \mathbf{S} \mathbf{v}_3 \\
&= \lambda_1^2 + s^2 \lambda_2^2 + t^2 \lambda_3^2
\end{aligned}$$

in which we have used the orthogonality property of eigenvectors $\mathbf{v}_i^\top \mathbf{v}_j = 0$ for all $i \neq j$. Finding the minimum norm is motivated by the fact that the singular values of the matrix $\mathbf{S}$ (i.e. the square roots of the eigenvalues of $\mathbf{S}^\top \mathbf{S}$) are the 2-norm distances of the respective vectors to the null space of $\mathbf{S}$. The eigenvector associated with the smallest singular value λ_1 can be considered the minimizing solution. Assuming $\mathbf{v}_1$ is the best fit, we can use combinations of the other two eigenvectors to find optimal parabolic solutions. With

$$\mathbf{g} = \begin{bmatrix} a & b & c & p & q & d \end{bmatrix}^\top$$

we can express the parabola constraint $b^2 - 4ac = 0$ with the eigenvectors as

$$\mathscr{C}(s,t) = \left(v_{1,2} + sv_{2,2} + tv_{3,2}\right)^2 - 4\left(v_{1,1} + sv_{2,1} + tv_{3,1}\right)\left(v_{1,3} + sv_{2,3} + tv_{3,3}\right) = 0$$

where $v_{i,j}$ is the jth element of the ith eigenvector, and in which

$$\begin{aligned}
a &= v_{1,1} + sv_{2,1} + tv_{3,1} \\
b &= v_{1,2} + sv_{2,2} + tv_{3,2} \\
c &= v_{1,3} + sv_{2,3} + tv_{3,3}.
\end{aligned}$$

This leads us to the Lagrangian

$$\mathscr{L}(s,t,\alpha) = \mathscr{F}(s,t) + \alpha \mathscr{C}(s,t)$$

which yields a fourth-order polynomial in α after equating the derivatives w.r.t. the parameters with zero. Solving the polynomial for α and back-substituting it to get s and t, we eventually obtain $\mathbf{g}_1$ and thus the parameter vector $\mathbf{g}$.

2.2.4 Fitting ellipsoids

For ellipsoid-specific fitting [45], the constraint (in which $k \geqq 4$) is chosen as

$$\mathbf{Q} = \operatorname{diag}\left(\frac{1}{2}\begin{bmatrix} 0 & k & k \\ k & 0 & k \\ k & k & 0 \end{bmatrix} - \begin{bmatrix} k & k & k \\ k & k & k \\ k & k & k \end{bmatrix}, -\frac{1}{4}\begin{bmatrix} k & 0 & 0 \\ 0 & k & 0 \\ 0 & 0 & k \end{bmatrix}, \mathbf{0}_{4\times 4}\right). \tag{2.33}$$

Unlike the constraint matrix for ellipses and hyperbolas, which express simple constraints, the ellipsoid-specific constraint depends on a parameter k. Let

$$\mathbf{g}^\top = \begin{bmatrix} g_1 & g_2 & g_3 & g_4 & g_5 & g_6 & g_7 & g_8 & g_9 & g_{10} \end{bmatrix}$$

which pair with components of

$$\mathbf{z}^\top = \begin{bmatrix} x^2 & y^2 & z^2 & xy & xz & yz & x & y & z & 1 \end{bmatrix}$$

and

$$\begin{aligned} I &= g_1 + g_2 + g_3 \\ J &= g_1 g_2 + g_2 g_3 + g_1 g_3 - \frac{1}{4} g_4^2 - \frac{1}{4} g_5^2 - \frac{1}{4} g_6^2. \end{aligned}$$

When $4J - I^2 > 0$, the parameter vector $\mathbf{g}$ represents an ellipsoid. On the other hand, when the short radius of an ellipsoid is at least half of its major radius, we have $4J - I^2 > 0$ (corresponding to $k = 4$). Unfortunately, not all ellipsoids satisfy this sufficient condition, in particular, long thin and compressed ellipsoids fail to have $4J - I^2 > 0$. However, for any ellipsoid, there always exists a k such that $kJ - I^2 > 0$. It can be shown that when the short radius of an ellipsoid is at least $\frac{1}{\sqrt{k}}$ multiple of its major radius, then we will have $kJ - I^2 > 0$. A bisection method, starting from a large k, estimating parameters $\mathbf{g}$ given $\mathbf{Q}(k)$, and iterating until $\mathbf{g}$ identifies an ellipsoid, we may guarantee ellipsoid-specific fitting, as shown in [45].

Example 7. Comparing the accuracy of ellipsoid fitting with noise cancellation to other fitting methods. Figure 2.6 demonstrates ellipsoid fitting and the impact of noise cancellation on fitting accuracy. Scattered points are located in three-dimensional space, and are shown with dots. The solid shape in the three-dimensional figure corresponds to maximum likelihood fit (geometric distance minimization) while wire-frames correspond to direct ellipsoid fit [45] (closer to the shape of a sphere) and ellipsoid fit with noise cancellation using the nonlinear Koopmans method (closer to the solid shape obtained with maximum likelihood). The two-dimensional figures are projections of the three-dimensional figure to the x, y and z axes, where the coordinate system is centered at the ellipsoid. The ellipsoid estimated with the noise cancellation method (dash-dot line) blends with the ellipsoid fitted with the maximum likelihood method (continuous line), both of which are far apart from the ellipsoid obtained with direct ellipsoid fit (dashed line). Ellipsoid semi-axis lengths 15, 20 and 25 have been chosen such that the short radius and the major radius have the proper ratio to a make a search for k in (2.33) unnecessary. Points are concentrated in a single region of the ellipsoid surface; if the points were sampled along the entire surface of the ellipsoid, all fitting methods would give very similar results, preventing a meaningful comparison. The Gaussian noise that contaminates points has unit variance in all directions.

Inspecting the outcome of the various fitting algorithms, it is clearly shown how the noise cancellation method outperforms direct ellipsoid fit and how close it is to maximum likelihood fit but is computed without iterations. Whereas the noise cancellation method and direct ellipsoid fit have approximately the same computational cost, both of which amount to solving one or two eigenvalue or singular value problems, maximum likelihood methods either employ a parameter search in the fashion of the Levenberg–Marquardt algorithm or they employ iterative schemes, which may take 8-10 iterations (if they converge). ♣

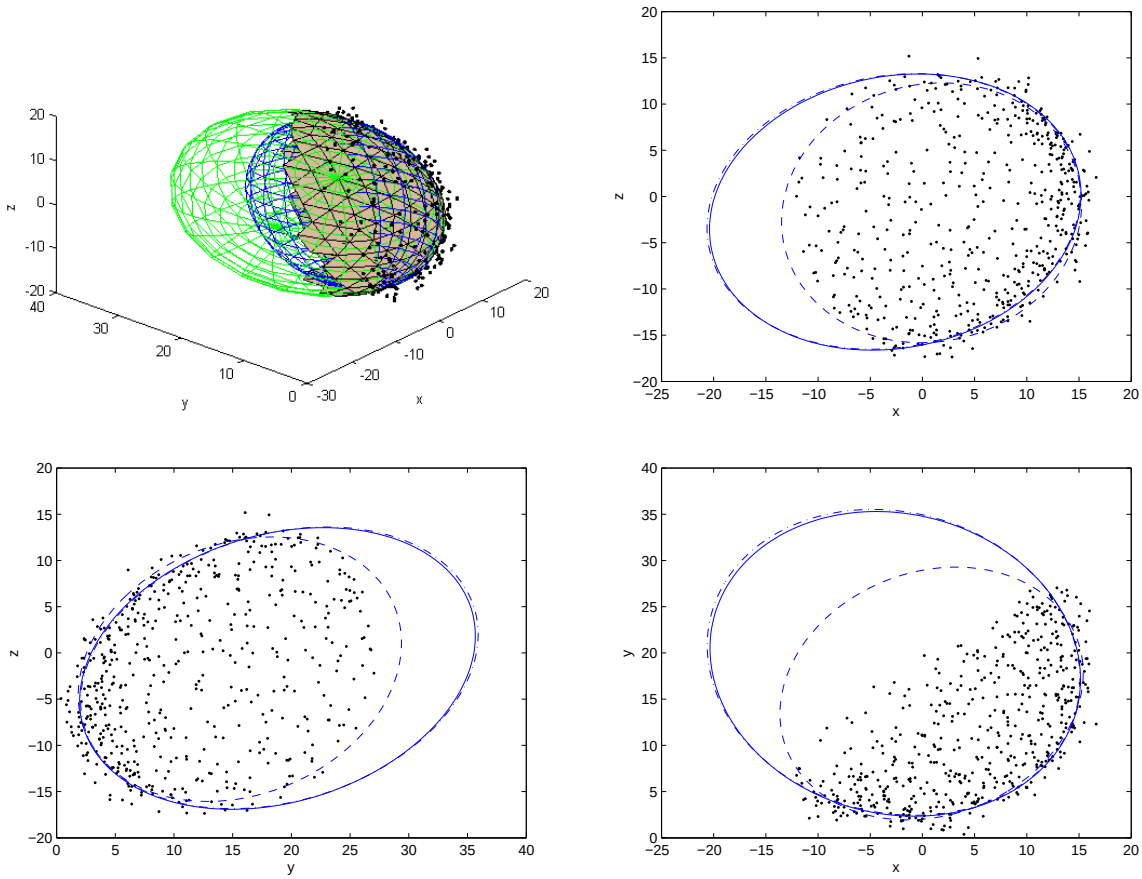


Figure 2.6: Fitting an ellipsoid to noisy data. The two-dimensional figures are projections of the three dimensional fit; maximum likelihood fit (continuous line), direct ellipsoid fit (dashed line) and ellipsoid fit with the proposed method (dash-dot line).

2.2.5 Constrained fitting with noise cancellation

Having surveyed unconstrained fitting methods in Section 2.1, with special emphasis on the noise cancellation scheme in Section 2.1.6, as well as constrained estimation methods in Sections 2.2.2, 2.2.3 and 2.2.4, we may now formulate a new errors-in-variables parameter estimation method for quadratic curves and surfaces subject to constraints, which comprises of the following two major steps:

1. noise cancellation

2. constrained quadratic least squares fitting

The first major part of the algorithm is the noise cancellation scheme for general quadratic curves and surfaces, which can be broken down into the following steps:

1. *Input:* noisy samples $\mathbf{x}_i$ and the relative noise magnitude vector $\bar{\sigma}^2_{\mathbf{x}}$ for each dimension.

2. Estimate the data covariance matrix $\mathbf{D}$ from noisy samples.

3. Estimate the noise covariance matrix polynomial coefficients $\mathbf{C}_1$ and $\mathbf{C}_2$ from noisy samples.

4. Compute the reduced-size matrices $\mathbf{D}_\star$, $\mathbf{C}_\star$ and $\mathbf{C}_{\star\star}$ by eliminating the statistically invariant terms in $\mathbf{D}$, $\mathbf{C}_1$ and $\mathbf{C}_2$.

5. Construct the matrix polynomial $\mathbf{\Psi}_\star(\mu) = \mathbf{D}_\star - \mu\mathbf{C}_\star - \mu^2\mathbf{C}_{\star\star}$.

6. Find the eigenvalue μ that solves $\det\big(\mathbf{\Psi}_\star(\mu)\big) = 0$.

7. *Output:* the noise-compensated (singular) matrix $\mathbf{R}_\star = \mathbf{D}_\star - \mu\mathbf{C}_\star - \mu^2\mathbf{C}_{\star\star}$.

The second major part is constrained fitting, which depends on the type of quadratic curve or surface the constraint identifies.

The following algorithm summarizes the constrained estimation scheme for ellipses and hyperbolas:

1. Construct the reduced scatter matrix $\mathbf{S}$ by writing the Schur complement of the quadratic terms in $\mathbf{R}_\star$.

2. Solve the generalized eigenvalue problem $\mathbf{S}\mathbf{g}_1 = \lambda\mathbf{Q}_1\mathbf{g}_1$ (or the corresponding singular value problem) for $\mathbf{g}_1$ with $\mathbf{Q}_1$ expressing the ellipse- and hyperbola-specific constraint.

3. Classify the eigenvectors $\mathbf{g}_{1,k}$ based on the eigenvalue λ_k and the constraint value $\mathbf{g}_{1,k}^\top\mathbf{Q}_1\mathbf{g}_{1,k}$.

4. Recover the original parameter vector $\mathbf{g}$.

The following algorithm summarizes parabola fitting:

1. Compute the eigenvector decomposition $\mathbf{S}\mathbf{g}_1 = \lambda\mathbf{g}_1$ with $\mathbf{S}$ being the Schur complement of the quadratic terms in $\mathbf{R}_\star$.

2. Based on $\mathbf{g}_1 = \mathbf{v}_1 + s\mathbf{v}_2 + t\mathbf{v}_3$, write the Lagrangian $\mathscr{L}(s,t,\alpha) = \mathscr{F}(s,t) + \alpha\mathscr{C}(s,t)$.

3. Solve the Lagrangian polynomial for α to recover the parameter vector $\mathbf{g}$.

The following algorithm summarizes the constrained estimation scheme for ellipsoids:

1. Construct the reduced scatter matrix $\mathbf{S}$ by writing the Schur complement of the quadratic terms in $\mathbf{R}_\star$.

2. Solve the generalized eigenvalue problem $\mathbf{S}\mathbf{g}_1 = \lambda\mathbf{Q}_1(k)\mathbf{g}_1$ (or the corresponding singular value problem) for $\mathbf{g}_1$ with $\mathbf{Q}_1(k)$ expressing the ellipsoid-specific constraint with a large $k \geqq 4$.

3. Use a bisection method on k until $\mathbf{g}_1$ identifies an ellipsoid.

4. Recover the original parameter vector $\mathbf{g}$.

Comparison with other methods

Many algorithms are available in unconstrained 2D fitting but fewer in constrained 2D fitting. Among constrained 2D fitting algorithms, two wide-spread algorithms include direct ellipse fitting [26] with the quadratic component reduction scheme [30], and maximum likelihood fitting. Direct ellipse fitting minimizes algebraic distance, uses covariance matrices, and involves no iterations but does not take into account the nonlinear perturbations induced by noise. In contrast, maximum likelihood fitting minimizes geometric distance with an iterative scheme where points are projected to the current curve estimates. Even while efficient projection schemes have been devised to project a point to an at most quadratic curve (Section 3.3 and Appendix A) , these remain relatively costly operations. The proposed algorithm alloys the benefits of the two approaches: it involves one or two eigenvalue or singular value problems on covariance matrices, uses no iterations but nevertheless compares favorably to the estimates that would be obtained via maximum likelihood. The proposed method delivers estimates close to those obtained with Taubin's method [61] or hyper-accurate ellipse fitting [39] but unlike these methods, the proposed method guarantees that the specified curve class is fitted, whereas Taubin's method and hyper-accurate ellipse fitting, despite what their name might suggest, may be misled by noise and fit other quadratic curve types if that appears to fit better to data.

For 3D ellipsoid fitting, the two competing algorithms are direct ellipsoid fitting [45] and maximum likelihood fitting. The same arguments as for 2D fitting apply, and the proposed method successfully combines the non-iterative nature of direct ellipsoid fitting with the accuracy of maximum likelihood estimation. Generalizations of Taubin's method [61] or hyper-accurate ellipse fitting [39] to 3D again lose their robustness by not enforcing the surface to be an ellipsoid, which may lead to undesired parameter estimates not compatible with a priori information.

2.3 Summary

This chapter has explored the parametric estimation problem over noisy point clouds. Data have been measured contaminated with Gaussian noise, and we have investigated how various parameter estimation methods fit curves and surfaces to the point set.

First, we have seen that iterative high-accuracy methods such as maximum likelihood or approximated maximum likelihood methods produce estimates close to the true curve or surface but at a substantial computation cost, with a risk of slow convergence or divergence with improper initialization. On the other hand, we have also seen how low-accuracy methods such as ordinary least squares or Taubin's method can deliver parameter estimates without iterations. The nonlinear Koopmans method, also known as consistent algebraic least squares, has been shown to combine the benefits of high-accuracy and low-accuracy methods, and yield accurate parameter estimates with a noise cancellation scheme, while involving moderate computational cost.

Second, we have seen how to impose constraints on the parameter estimates and thus produce a class of curves or surfaces that satisfy some property we are interested in. In particular, we have investigated fitting ellipses, hyperbola, parabola to two-dimensional data,

and ellipsoids to three-dimensional data. We have seen how the preliminary noise cancellation scheme adds estimation accuracy to least-squares-based methods, and how the estimates obtained in this manner are superior to those obtained with ordinary least squares.

Novel contributions for unconstrained and constrained fitting include:

- applying the principle of the nonlinear Koopmans estimator to fitting quadratic curves and surfaces;
- achieving noise cancellation by solving a symmetric quadratic eigenvalue problem;
- reducing matrix dimension by eliminating statistically invariant terms;
- applying the noise cancellation scheme to quadratic curve estimation and thus improving (type-specific) ellipse, hyperbola and parabola fit;
- applying the noise cancellation scheme to quadratic surface estimation and thus improving ellipsoid fit;
- a new cost-effective algorithm for fitting particular classes of quadratic curves and surfaces.

CHAPTER

3 Structure discovery

With the increasing availability of data from various sources and modalities, there has been escalated interest in acquiring, compressing, storing, transmitting and processing massive amounts of data. Most of these advances rely on the observations that even though data sets are often high-dimensional, their intrinsic dimension is often much smaller. For example, we may have a data set of three dimensional data points, yet we are interested in discovering lines, planes or ellipsoids in the data set. Conventional techniques, such as singular value decomposition, seek a low-dimensional representation of the higher dimensional space, a compact model that explains the data points using a limited set of base vectors, assuming all data points originate from a single low-dimensional space. In most situations, however, data points are seldom drawn from a single low-dimensional space. Rather, they belong to multiple subspaces, and modeling involves both identifying subspace membership as well as subspace fitting. Subspace clustering [68], as the problem is typically referred to, is a significantly more difficult problem with a number of challenges:

1. Data segmentation and model parameter estimation are strongly coupled. Were either the segmentation or the estimation known, the other could be easily derived using conventional estimation methods or projection. Simultaneous segmentation and estimation is, however, more demanding.

2. Uneven distribution of data points and the relative position of subspaces with respect to one another, including subspace intersections, seeks for more involved algorithms.

3. Model selection, i.e. choosing the right dimension for each of the subspaces, can be difficult if the subspaces admit more complex structures.

4. Noise corruption introduces more uncertainty to model accuracy compared to the case of a single subspace.

An intuitive way to address the problem of subspace discovery is to employ iterative refinement. Given an initial segmentation, a subspace can be fit to each group using classical techniques such as singular value decomposition. Then, given a model for each subspace, each

data point is assigned to its closest subspace. By iterating these two steps until convergence, an estimate of the subspaces and segmentation may be discovered.

Despite their simplicity, iterative algorithms are sensitive to initialization. If data points are randomly assigned, several restarts are required before a near-optimum solution is found. In order to reduce the number of restarts, algorithms that produce feasible results without any preliminary input are required. Spectral Local Best-fit Flats (SLBF) [71] is based on the observation that points and their nearest neighbors often belong to the same subspace, which provides estimates for the parameters of the subspace. A distance between a point and the locally estimated subspace around the other can serve as a(n asymmetric) distance measure between two data points. The thesis extends the principles of SLBF to the nonlinear case, and we shall fit manifolds, described by polynomial functions, rather than subspaces to data, and employ appropriate projections.

Structure discovery methods, including the algorithm presented here, contrast sharply with approximation methods. Many approaches exist that approximate a data set with a spline curve or surface, or some other weighting scheme where the model arises as a set of basis functions and associated weights. The approach may be geometrically motivated whereby approximation is interpreted as a continuous evolution process that drives an initial surface towards the target specified by the data points [4], or be a moving least squares variant where each data point is associated with a support region and has a local surface estimate, and these local estimates are blended together with a weighting matrix [29]. Other methods use a decomposition scheme to split the entire domain to suitably small domains that can be fit with a simple function [52] or merged to form larger clusters based on the inter-cluster distance and intra-cluster distance criteria [70]. These methods, however, typically do not aid in understanding the data by finding a natural decomposition that reflects the internal model from which data originate.

Partitioning the entire data set into clusters where points within the cluster are related in the same way whereas different clusters are captured by different relationships can identify internal structure. Beyond the simple case when the cluster is represented by a single point, such as k-means, work on partitioning data has been focusing on subspace clustering or hybrid linear modeling algorithms where clusters are assumed to be related in a linear manner. These approaches include k-flats (KF) and its variants [10, 1], generalized principal component analysis (GPCA) [69], local subspace affinity (LSA), random sample consensus (RANSAC) for hybrid linear modeling, agglomerative lossy compression (ALC) [46], spectral curvature clustering (SCC) [11], sparse subspace clustering (SSC) [25] and (spectral) local best-fit flats (SLBF/LBF) [72]. An overview of such algorithms is given in [68].

Extensions of hybrid linear modeling algorithms exist that are not limited to clusters with intra-cluster data related in a linear manner. nonlinear manifold clustering methods include locally linear manifold clustering (LLMC) [27], manifold clustering with node-weighted multidimensional scaling [59], kernel spectral curvature clustering (KSCC) [12] and mixture of probabilistic principal surfaces (MiPPS) [41]. Many of these are based on minimizing a cost function involving an affine combination of all points with different weights, or expectation maximization with general basis or kernel functions, shifting focus towards discovering the clusters rather than understanding the structure of the clusters themselves. A common characteristic is that the cluster definition is implicit (captured by some arrangement of data, e.g.

closeness to a cluster center) rather than explicit (present in the data itself, e.g. explained by the equation of an ellipse).

Section 3.1 describes an iterative algorithm for clustering a data set into groups fitted by at most quadratic curves and surfaces, which is a new contribution of the thesis. The iterative algorithm, which alternates between an update and an assignment step, expects proper initialization, thus Section 3.2 discusses another contribution specific to this thesis that does not need such initialization, and uses spectral clustering to find a segmentation without iterations. Both of these algorithms rely on effective projection algorithms onto quadratic curves and surfaces, these are covered in Section 3.3, which extends previously known results. Finally, Section 3.4 discusses an asymmetric spectral clustering algorithm the non-iterative clustering proposed in Section 3.2 uses, with parametrization specific to the context in which it is utilized.

3.1 Iterative algorithm for manifold clustering

Given a set of observations $\mathbf{x}_i$ with $i = 1, \ldots, N$ where each $\mathbf{x}_i$ is a d-dimensional vector, the standard k-means clustering aims to partition the N observations into k sets $\mathscr{S} = \{\mathscr{S}_1, \ldots, \mathscr{S}_k\}$ so as to minimize the within-cluster sum of squares

$$\arg\min_{\mathscr{S}} \sum_{p=1}^{k} \left(\sum_{\mathbf{x}_i \in \mathscr{S}_p} \left\| \mathbf{x}_i - \mathbf{m}_{\mathscr{S}_p} \right\|^2 \right)$$

where

$$\mathbf{m}_p = \mathbf{m}_{\mathscr{S}_p} = \frac{1}{\left|\mathscr{S}_p\right|} \sum_{\mathbf{x}_j \in \mathscr{S}_p} \mathbf{x}_j$$

is the mean of data points in the set $\mathscr{S}_p$.

Given an initial set of k means $\mathbf{m}_{1,(0)}$ to $\mathbf{m}_{k,(0)}$ (typically chosen randomly), the standard iterative algorithm proceeds by alternating between two steps: an *assignment step* and an *update step*. The so-called assignment step maps each observation to the cluster whose mean is closest to it, i.e.

$$\mathscr{S}_{p,(t)} = \left\{ \mathbf{x}_i : \left\| \mathbf{x}_i - \mathbf{m}_{p,(t)} \right\| \leq \left\| \mathbf{x}_i - \mathbf{m}_{j,(t)} \right\| \ \forall\ 1 \leq j \leq k \right\}$$

where each $\mathbf{x}_i$ is assigned to exactly one $\mathscr{S}_{p,(t)}$, even if it could be is assigned to two or more of them. The update step calculates the new means to be the centroids of the observations in the new clusters, i.e.

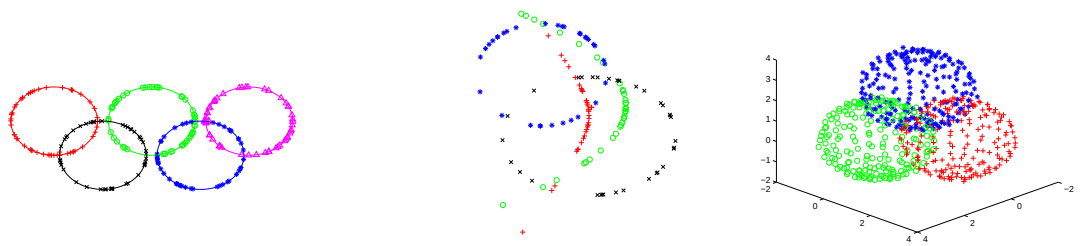
$$\mathbf{m}_{p,(t+1)} = \frac{1}{\left|\mathscr{S}_{p,(t)}\right|} \sum_{\mathbf{x}_j \in \mathscr{S}_{p,(t)}} \mathbf{x}_j$$

The algorithm has converged when the assignments no longer change.

While simple and in many cases effective, the standard iterative algorithm, unfortunately, suffers from several limitations. The entire point set $\mathscr{S}_p$ is assumed to be represented by a single point $\mathbf{m}_p$, which is typically not sufficient unless the sets to identify are covered by non-intersecting spheres. Otherwise, the algorithm fails to identify a natural grouping of data points, and will produce sub-optimal results.

	k-means [47]	LGA [4]	manifold clustering
Goal	find best centers	find best-fit lines	find best-fit curves
Error measure	distance from center	distance from line	distance from curve
Update step	compute mean	estimate parameters	estimate parameters
Assignment step	assign to closest center	project to line	project to curve

Table 3.1: Comparison of the standard algorithm for k-means, the linear grouping algorithm (LGA) and the proposed algorithm in terms of their objective, their update and assignment steps for the two-dimensional case.



(a) **(b)** **(c)**

Figure 3.1: Cluster assignments on some artificial data sets.

Linear grouping or k-flats algorithms [10, 64, 2, 1] are an improvement over standard k-means in that they are capable of finding linear patterns in a data set. Rather than using a single $\mathbf{m}_p$, each set $\mathscr{S}_p$ is captured by a set of parameters $\boldsymbol{\theta}_p$ where noise-free data points $\bar{\mathbf{x}}_i$ satisfy $\boldsymbol{\theta}_p^\top \bar{\mathbf{x}}_i = 0$ where $\bar{\mathbf{x}}_i \in \mathscr{S}_p$.[1] This is a leap forward as intersecting data sets are no longer a limitation, provided they are captured by different parameters $\boldsymbol{\theta}$. The way linear grouping algorithms work resembles the k-means algorithm but the goal is to find best-fit d-dimensional planes instead of best centroids, the error measure is the distance from a d-dimensional plane instead of the distance from a centroid, the update step uses total least squares (singular value decomposition) and the assignment step is a projection to a d-dimensional plane (Table 3.1). As singular value decomposition and projection to a hyperplane are well-understood and easy to compute, linear grouping algorithms can be effective if the underlying model is indeed linear.

Unfortunately, linear grouping or k-flats algorithms can solve the linear modeling problem but stumble into difficulty when the cluster models are nonlinear. Generalizing the fitting shape to include quadratic (or higher-order) curves and surfaces (expressed as an implicit polynomial function) is a natural extension to linear grouping algorithms. Such a method can discover a structural decomposition of data where members of each group are related by a low-order (linear or quadratic) implicit polynomial function.

[1] When parameters and data are related in a linear manner, that is, $\mathbf{f}_{par}(\mathbf{g})$ is an identity mapping, we have $\mathbf{g}_p^\top \bar{\mathbf{x}}_i = 0$ where $\bar{\mathbf{x}}_i \in \mathscr{S}_p$.

Example 8. Outcome of a manifold clustering algorithm. Figure 3.1 illustrates cluster outcomes when a nonlinear algorithm (the method to be presented shortly) is executed on various artificial data sets used in [12], either with no noise (data points along five intersecting circles in Figure 3.1a, and data points along a circle, an ellipse, a parabola and a hyperbola in Figure 3.1b) or low noise level (three intersecting spheres in Figure 3.1c). Neither k-means nor k-flats algorithms could handle these simple cases. ♣

The outline of the proposed iterative grouping algorithm resembles the iterative algorithm of the standard k-means procedure. The primary difference lies in the use of parameters instead of mean values, and data point projection replacing simple point-to-point distance (between cluster center and data points) (see Table 3.1). As with the standard k-means algorithm, the choice of initial data points affects convergence to an optimal solution.

Iterative grouping algorithm fitting polynomial functions.

1. Initialization. Choose k randomly chosen initial points $\mathbf{x}_i$ with $i = 1, \ldots, k$, and for each data point $\mathbf{x}_i$

   a) start with an initial neighborhood $\mathcal{N}(\mathbf{x}_i)$ around $\mathbf{x}_i$

   b) estimate the parameters $\boldsymbol{\theta}_{i,(0)}$ that best capture points in $\mathcal{N}(\mathbf{x}_i)$

   c) enlarge $\mathcal{N}(\mathbf{x}_i)$ by adding nearest-neighbor points

   d) compute new estimates $\boldsymbol{\theta}_{i,(n)}$ and compare them to the estimates $\boldsymbol{\theta}_{i,(n-1)}$ obtained in the previous iteration

   e) repeat until the neighborhood $\mathcal{N}(\mathbf{x}_i)$ cannot be enlarged without worsening the accuracy of $\boldsymbol{\theta}_{i,(n)}$

   f) let $\boldsymbol{\theta}_i$ be the best $\boldsymbol{\theta}_{i,(n)}$ that belongs to the optimum $\mathcal{N}(\mathbf{x}_i)$ around $\mathbf{x}_i$.

2. Initial grouping.

   a) project each data point $\mathbf{x}_j$ with $j = 1, \ldots, N$ to all candidate $\boldsymbol{\theta}_i$ with $i = 1, \ldots, k$

   b) form initial groups $\mathcal{S}_i$ with $i = 1, \ldots, k$ such that the distance is minimized.

3. Alternating optimization.

   a) Update step. Each group of data points $\mathcal{S}_i$ is used to estimate new shape parameters $\boldsymbol{\theta}_i$ (where the best-fit surface minimizes μ in the nonlinear estimator).

   b) Assignment step. Each data point $\mathbf{x}_j$ is assigned to shapes $\boldsymbol{\theta}_i$ it lies in the vicinity of (e.g. the closest surface that minimizes the distance between the data point and its foot point on the surface).

4. Finalization. Each data point is assigned to a single shape $\boldsymbol{\theta}_i$ it lies closest to.

5. Re-sampling. Repeat the algorithm with different randomly chosen initial locations.

The main idea of the algorithm is replacing linear subspace fitting with fitting curved manifolds, described by polynomial functions. The inputs of the algorithm are as follows:

$\mathbf{X}$	data set, d rows, N columns
d	number of dimensions (2 for plane, 3 for space)
N	number of data points
Σ	the $d \times d$ noise covariance matrix
k	the desired number of clusters

Each column of the data set matrix $\mathbf{X}$ is a data point $\mathbf{x}_i^\top = \begin{bmatrix} x_i & y_i \end{bmatrix}$ for 2 dimensions or $\mathbf{x}_i^\top = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}$ for 3 dimensions, and the covariance matrix is expressed as $\Sigma = \mathbb{E}\left(\tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^\top\right)$ where the data point $\mathbf{x}_i$ is split into a noise-free part and a noise part as $\mathbf{x}_i = \bar{\mathbf{x}}_i + \tilde{\mathbf{x}}_i$ where neither can be observed directly. As output, the algorithm produces a mapping (a membership set) $\mathscr{S}$ between data points, and one of the k clusters. Data points $\mathbf{x}_i$ in the same cluster are captured by the same polynomial function. Thus, after a few iterations, the method constructs a model as a union of polynomial functions fitted to data. As with the standard algorithm for k-means, the algorithm has converged to its final state with no data points re-grouped from one cluster to another.

The algorithm can be broken down into the following sub-problems:

1. Fitting polynomial curves and surfaces to data. One way to accomplish fitting curves and surfaces is with the nonlinear Koopmans (NK) method [66], which can be seen as an alternative formulation of consistent algebraic least squares [42, 48]. The method solves a polynomial eigenvalue problem, the matrix coefficients being the data covariance matrix and noise cancellation terms. The data covariance matrix is computed from the data $\mathbf{x}_i$ subject to the nonlinear transformation $\mathbf{f}_{data}(\mathbf{x}_i)$, and the noise cancellation terms mitigate the effect of this transformation on noise. The approach is not optimal in a maximum likelihood sense but nevertheless consistent: more data produces better estimates. Unconstrained fitting algorithms have been discussed in Sections 2.1. When a priori information is available on the curves or surfaces to be fitted, economical constrained fitting algorithms, such as those presented in 2.2 as a new contribution of the thesis, may be employed.

2. Identifying local neighborhoods. Any estimation method requires sufficient data to produce reliable parameter estimates, especially in the presence of noise. In most cases, the local neighborhood of a data point consists of other points that are related in a similar way, or in other words, captured by the same polynomial function. This is implicit in many multi-manifold clustering algorithms but is made explicit in our approach: points in a local neighborhood determine a fitting, and the fitting is deemed to apply in that neighborhood. Choosing the right neighborhood can be accomplished with an inflationary algorithm involving k_{nn} nearest neighbors, where k_{nn} is an increasingly larger number until a suitable error measure begins to increase rather than decline.

3. Projecting a point onto a curved manifold. While projecting to a subspace can be easily accomplished, projecting to a curved manifold captured by a polynomial function is computationally more intensive. Efficient methods that compute the so-called *foot*

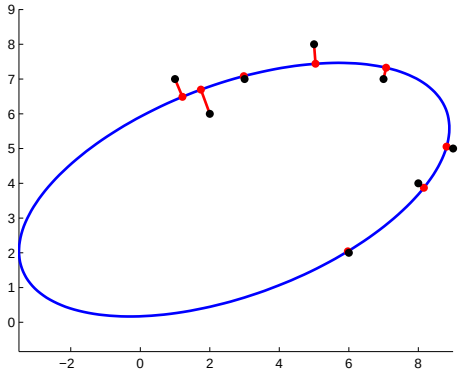


Figure 3.2: Data points and their associated foot points obtained by projecting the data points onto the ellipse, minimizing geometric distance.

point of a data point that does not lie on the curved manifold (Figure 3.2) are essential to the algorithm. Special cases for projecting to ellipses and ellipsoids, and quadratic curves and surfaces are of particular significance. Projection onto a manifold lets us determine the distance between a point and its foot point where the parameters of the curved manifold on which the foot point lies have been estimated from a local neighborhood. Projection algorithms are discussed in Section 3.3.

The iterative algorithm for manifold clustering has a significant element of randomness, namely the choice of initial data points. Ideally, initial points are chosen such that each lies on a different manifold, and each has a local neighborhood large enough to obtain reliable estimates. Unfortunately, this is not always the case. The re-sampling step aims to reduce the likelihood of improperly chosen initial locations, whereas the non-iterative algorithm for manifold clustering in Section 3.2 remedies the issue by involving all points in forming k clustering using a specially crafted pairwise distance measure.

3.2 Non-iterative algorithm for manifold clustering

The iterative algorithm for manifold clustering depends on a proper choice of initial data points, which in turn determine the initial grouping. A method that finds an initial segmentation without a preliminary assignment of data points, however, may both serve an initial state for an iterative algorithm, as well be a clustering algorithm of its own.

The proposed non-iterative method is based on constructing an asymmetric distance matrix and finding a best weighted cut in a complete graph, a task that is accomplished with (asymmetric) spectral clustering. The nodes in the graph are data points to be clustered, and the edges are weighted (inversely) by the entries a_{ij} of the asymmetric matrix (the less the distance, the greater the weight), which express distance between a point i and the surface estimated from points in the neighborhood of the other point j. The best weighted cut splits the graph into k components such that the weights running in between components are

minimized. This corresponds to a partition into k clusters where we aim to maximize the within-cluster sum-of-squares whereas minimize the inter-cluster sum-of-squares.

Grouping with spectral clustering remedies the issue of wrongly chosen initial locations. Spectral clustering eliminates the need for re-sampling and provides a clustering that is already close to an optimum solution.

Grouping algorithm fitting polynomial functions, driven by spectral clustering

1. Initialization. For each data point $\mathbf{x}_i$ with $i = 1, \ldots, N$
   a) start with an initial neighborhood $\mathcal{N}(\mathbf{x}_i)$ around $\mathbf{x}_i$
   b) estimate the parameters $\boldsymbol{\theta}_{i,(0)}$ that best capture points in $\mathcal{N}(\mathbf{x}_i)$
   c) enlarge $\mathcal{N}(\mathbf{x}_i)$ by adding nearest-neighbor points
   d) compute new estimates $\boldsymbol{\theta}_{i,(n)}$ and compare them to the estimates $\boldsymbol{\theta}_{i,(n-1)}$ obtained in the previous iteration
   e) repeat until the neighborhood $\mathcal{N}(\mathbf{x}_i)$ cannot be enlarged without worsening the accuracy of $\boldsymbol{\theta}_{i,(n)}$
   f) let $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_i)}$ be the best $\boldsymbol{\theta}_{i,(n)}$ that belongs to the optimum $\mathcal{N}(\mathbf{x}_i)$ around $\mathbf{x}_i$.
2. Compute asymmetric distances. For each pair of data points $\mathbf{x}_i$ and $\mathbf{x}_j$, obtain their asymmetric distances
   a) project $\mathbf{x}_i$ to the manifold parametrized by $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_j)}$ and calculate a_{ij}
   b) project $\mathbf{x}_j$ to the manifold parametrized by $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_i)}$ and calculate a_{ji}.
3. Spectral clustering.
   a) build an affinity matrix from asymmetric distances a_{ij}
   b) assign data points to groups based on the affinity matrix eigenvectors.

Conceptually, the method is a spectral clustering algorithm with a specialized distance matrix measuring the distance of a point and its projection onto a curved manifold, which has been estimated from a local neighborhood. As such, the inputs of the algorithm are as follows:

$\mathbf{X}$	data set, d rows, N columns
d	number of dimensions (2 for plane, 3 for space)
N	number of data points
Σ	the $d \times d$ noise covariance matrix
k	the desired number of clusters

As previously, each column of the data set matrix $\mathbf{X}$ is a data point $\mathbf{x}_i^\top = \begin{bmatrix} x_i & y_i \end{bmatrix}$ for 2 dimensions or $\mathbf{x}_i^\top = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}$ for 3 dimensions, and the covariance matrix $\Sigma = \mathbb{E}\left(\tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^\top\right)$ where the data point $\mathbf{x}_i$ is split into a noise-free part and a noise part as $\mathbf{x}_i = \bar{\mathbf{x}}_i + \tilde{\mathbf{x}}_i$ where neither can be observed directly. As output, the algorithm produces a mapping (a membership

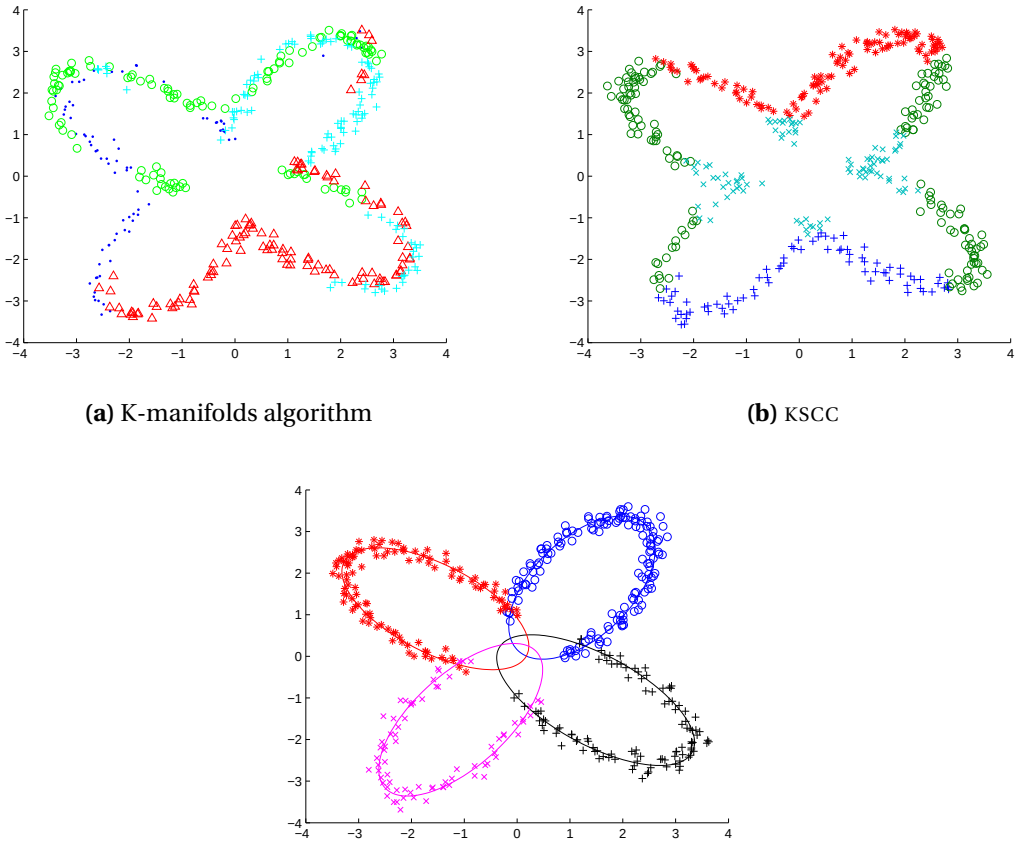
set) $\mathscr{S}$ between data points, and one of the k clusters. Data points $\mathbf{x}_i$ in the same cluster are captured by the same polynomial function. Thus, the method constructs a model as a union of polynomial functions fitted to data but without iterations.

As with the iterative algorithm for manifold clustering (see Section 3.1), fitting polynomial curves and surfaces to data (Sections 2.1 and 2.2), identifying local neighborhoods (as in Section 3.1) and projecting a point onto a curved manifold (Section 3.3) are essential sub-problems that have to be addressed to ensure the efficiency of the clustering method. In addition to these sub-problems, the non-iterative algorithm features spectral clustering using an asymmetric distance matrix. More specifically, a clustering method by weighted cuts in directed graphs is employed to find a partitioning of the entire data set into disjoint clusters, exploiting the asymmetry in the distance matrix. The method is based on a generalization of the original spectral approach involving symmetric matrices. The asymmetric distance matrix stems from curved manifolds estimated around the local neighborhood of points, and projecting all (typically other) points onto this manifold, and measuring the distance between point and its foot point. Spectral clustering reveals points that naturally group together, i.e. groups of points that are captured by the same polynomial relationship, which is what we ultimately seek. Spectral clustering is investigated in Section 3.4.

Once a spectral clustering of data is available, we may use it to seed an algorithm with potentially higher accuracy but higher sensitivity to the initial state, which would refine the grouping discovered by spectral clustering. A manifold clustering algorithm in flavor of k-planes, e.g. the iterative algorithm for manifold clustering in Section 3.1 using the nonlinear Koopmans method [66] or consistent algebraic least squares [42, 48] for estimation and Euclidean distance for projection can be used to refine both data point grouping and parameter estimates.

Example 9. Effectiveness of manifold clustering algorithms. Figure 3.3 illustrates the effectiveness of the proposed non-iterative manifold algorithm, comparing it to other manifold clustering algorithms that use either locality and connectivity information or the kernel trick, executed on the same data set. 440 data points have been generated along four intersecting ellipsoids, with no points sampled along the inner arcs, as in Figure 1.2. Data are polluted with medium level of Gaussian noise, $\sigma = 0.125$. The data set features the following important properties that manifold clustering algorithms should cope with: (1) data have been sampled from nonlinear surfaces that originally intersect, which has been identified [59] as a major challenge for such algorithms; (2) data points form a single shape yet the shape has natural partitions it can be broken down into, which may prove difficult for some algorithms that are steered towards connectedness; (3) data is noisy.

The K-manifolds algorithm [59] ran with default options and was set to look for four one-dimensional manifolds where each manifold would correspond to an ellipse. The cluster assignment in Figure 3.3a reveals how the K-manifolds algorithm fails to capture the presence of four distinct shapes and tends towards treating the entire data set as a whole, splitting the data set in a rather arbitrary way. This highlights a major deficiency of the K-manifolds algorithm, namely that it can only group data points into clusters but it cannot identify the relationship that explains them, which would help produce more insightful groupings. The weakness of the K-manifolds algorithm lies with the junctions where it fails to detect the discontinuity (i.e. split data points between the two ellipses involved).



(a) K-manifolds algorithm

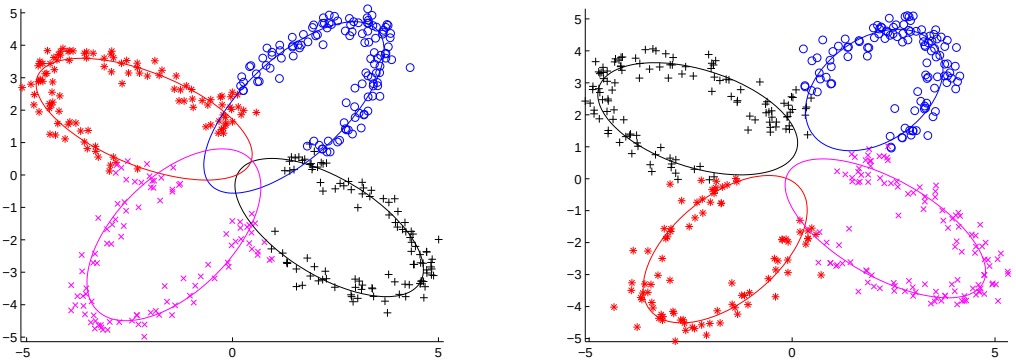
(b) KSCC

(c) proposed algorithm

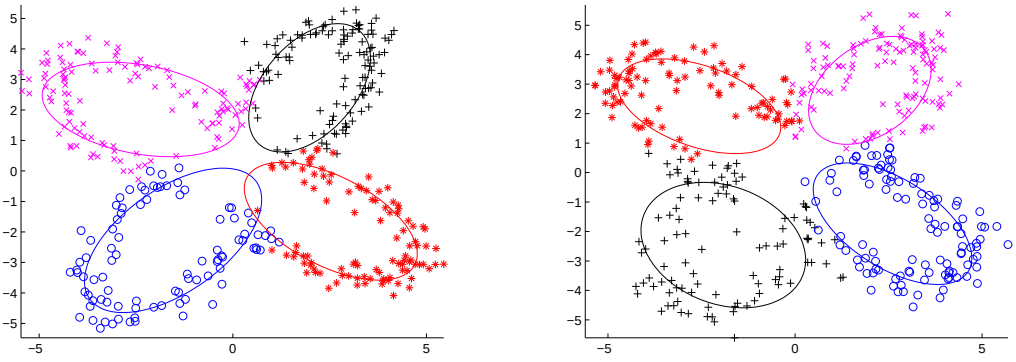
Figure 3.3: Comparison of manifold clustering algorithms. The K-manifolds algorithm and Kernel Spectral Curvature Clustering (KSCC) are significantly outperformed by the proposed non-iterative algorithm.

Similar results are obtained with Kernel Spectral Curvature Clustering (KSCC) [12, 11], which are shown in Figure 3.3b. As a kernel function, the standard quadratic polynomial kernel $k(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle + \langle \mathbf{x}^2, \mathbf{y}^2 \rangle$ has been chosen where the operator $\bullet^2$ is to be understood element-wise and $\langle \bullet, \bullet \rangle$ stands for dot product. Even while KSCC is also inspired by nonlinear transformation, namely, it transforms the feature space into kernel space using the nonlinear kernel function, the transformation is quite different from what we have seen in Sections 2.1 and 2.2. In fact, the choice of the proper kernel function is a key issue in KSCC, although it is not obvious how to choose it: one function may lead to a successful clustering, whereas another may not [12].

Finally, the outcome of the proposed algorithm is shown in Figure 3.3c. The algorithm almost perfectly identifies the four clusters (each marker symbol type corresponds to a different group), and the parameter estimates obtained from the four clusters (shown with continuous lines) reflect our own intuitive decomposition. The significant difference in favor of the proposed algorithm is explained by the fact that the algorithm uses structural informa-



(a) $\sigma = 0.15$ **(b)** $\sigma = 0.20$



(c) $\sigma = 0.25$ **(d)** $\sigma = 0.30$

Figure 3.4: Robustness of the proposed non-iterative manifold clustering algorithm in response to increasing level of noise.

tion (takes into account internal structure) and estimates parametric curves and surfaces, which improves its explanation power and in turn its fitting capabilities. This is unlike the K-manifolds algorithm, which partitions the data set but involves no parameter estimation step, only locality and connectivity information. Like KSCC, the proposed algorithm involves a polynomial mapping from the 2-dimensional plane into a 6-dimensional space with terms such as x^2, xy or y^2. In contrast to KSCC, however, the choice of the polynomial terms in the proposed algorithm depends only on the dimensionality of the curve or surface to be fitted and possibly restrictions we may want to impose (e.g. fit spheres, not ellipsoids, in which case we might use fewer parameters and omit terms such as xy), they are therefore much more straightforward to select. Due to the freedom of the choice of the kernel function, however, KSCC can fit data sets the proposed algorithm cannot (e.g. intertwining spirals), given the appropriate kernel function. ♣

Example 10. Robustness of the proposed manifold clustering algorithm. The noise-resilience of the algorithm and its robustness against uneven distribution of samples is demonstrated in Figures 3.4a, 3.4b, 3.4c and 3.4d, which show the outcome of the spectral clustering phase

with increasing levels of Gaussian noise contaminating the original data set of 440 points, with noise standard deviation $\sigma = 0.15$, $\sigma = 0.20$, $\sigma = 0.25$ and $\sigma = 0.30$, respectively. The distribution of data points is not even: points in the top right quadrant are twice as numerous as points in the bottom left quadrant, while the other two quadrants have approximately equal number of points. The pattern how data points have been sampled from the ellipses they originate from matches Figure 1.2 (as well as Example 9). The continuous lines show the curves whose parameters have been estimated from the clusters the proposed manifold clustering algorithm has identified. As illustrated in Figures 3.4a, 3.4b, 3.4c and 3.4d, even with substantial noise, when the points themselves hardly lie along ellipses any more, the algorithm is able to identify the four different shapes, albeit with worse precision. This contrasts sharply with other algorithms, such as K-manifolds or KSCC, which are rather sensitive to noise. ♣

3.3 Projection

Effective projection algorithms are a key to a manifold clustering algorithm, as they are the most resource-intensive step in the process. Every data point has to be projected to each curve and surface estimated in order to determine their foot points. Using the distance of the original point and its foot point, we may define an asymmetric distance measure, and can eventually map each point to the curve or surface it lies closest to.

Even while projection to a linear hypersurface (such as lines or planes) is easily accomplished, quadratic hypersurfaces (such as conics or quadrics) pose a greater demand on computer resources. Fortunately, an in-depth analysis of the special cases of quadratic curves and surfaces reveals an approach that computes foot points in a simple manner, with standard Newton's method. The specifics of the approach depend on the type of quadratic curve or surface to project to but otherwise the projection algorithm can be efficiently implemented on a parallel architecture, e.g. a general-purpose graphics processing unit (GPGPU).

3.3.1 Projection for linear data function

When linear estimation is used, the foot point is calculated as a simple point–line or point–plane distance. Let $\mathbf{n}$ be the normalized line or plane normal vector and c the constant in the Hessian normal form equation of the plane

$$L(\mathbf{x}) = \mathbf{n} \cdot \mathbf{x} + c = 0$$

such that $\boldsymbol{\theta}^\top = \begin{bmatrix} \mathbf{n}^\top & c \end{bmatrix}$ up to scaling by a constant. In such a case, the (signed) distance between the point and the line or plane is given by the projection

$$d_i = \mathbf{n} \cdot \mathbf{x}_i + c.$$

3.3.2 Projection for general quadratic data function

When quadratic estimation is used, obtaining the foot point is not as straightforward as in the linear case [22]. A quadratic curve in two dimensions or a quadric surface in three di-

mensions can be recast in the form

$$Q(\mathbf{x}) = \mathbf{x}^\top \mathbf{A}\mathbf{x} + \mathbf{b}^\top \mathbf{x} + c = 0 \tag{3.1}$$

where $\mathbf{A}$ is a symmetric matrix such that

$$\boldsymbol{\theta}^\top = \begin{bmatrix} \text{symvec}(\mathbf{A})^\top & \mathbf{b}^\top & c \end{bmatrix}$$

where symvec$(\mathbf{A})$ stacks the upper triangular part of the matrix $\mathbf{A}$ including the main diagonal into a column vector. Geometrically, the closest point $\mathbf{x}$ on the curve or surface to an arbitrary point $\mathbf{w}$ (typically not on the curve or surface) must satisfy the condition that $\mathbf{w}-\mathbf{x}$ is normal to the surface. Since the surface gradient $\nabla Q(\mathbf{x})$ is normal to the surface, the algebraic condition for the closest point is

$$\mathbf{w}-\mathbf{x} = t\nabla Q(\mathbf{x}) = t(2\mathbf{A}\mathbf{x}+\mathbf{b})$$

for some scalar t. Therefore, $\mathbf{x} = (\mathbf{I}+2t\mathbf{A})^{-1}(\mathbf{w}-t\mathbf{b})$ where $\mathbf{I}$ is the identity matrix.

Instead of immediately replacing $\mathbf{x}$ in the quadratic equation, factor $\mathbf{A}$ using an eigendecomposition to obtain $\mathbf{A} = \mathbf{R}\mathbf{D}\mathbf{R}^\top$ where $\mathbf{R}$ is an orthonormal matrix whose columns and $\mathbf{D}$ is a diagonal matrix whose diagonal entries are eigenvectors and eigenvalues of $\mathbf{A}$, respectively. Then

$$\begin{aligned} \mathbf{x} &= (\mathbf{I}+2t\mathbf{A})^{-1}(\mathbf{w}-t\mathbf{b}) \\ &= (\mathbf{R}\mathbf{R}^\top + 2t\mathbf{R}\mathbf{D}\mathbf{R}^\top)^{-1}(\mathbf{w}-t\mathbf{b}) \\ &= \left\{\mathbf{R}(\mathbf{I}+2t\mathbf{D})\mathbf{R}^\top\right\}^{-1}(\mathbf{w}-t\mathbf{b}) \\ &= \mathbf{R}(\mathbf{I}+2t\mathbf{D})^{-1}(\alpha - t\beta) \end{aligned}$$

where

$$\alpha = \mathbf{R}^\top \mathbf{w} \qquad \beta = \mathbf{R}^\top \mathbf{b}.$$

Re-substituting into the quadratic equation (3.1),

$$\begin{aligned} 0 &= (\alpha - t\beta)^\top (\mathbf{I}+2t\mathbf{D})^{-1} D (\mathbf{I}+2t\mathbf{D})^{-1}(\alpha - t\beta) \\ &+ \beta(\mathbf{I}+2t\mathbf{D})^{-1}(\alpha - t\beta) + c \end{aligned}$$

which is an at most fourth-degree polynomial for two dimensions, and an at most sixth-degree polynomial for three dimensions in terms of the scalar variable t. Once the roots t_k are found, they can be substituted into

$$\mathbf{x}_k = (\mathbf{I}+2t\mathbf{A})^{-1}(\mathbf{w}-t_k\mathbf{b})$$

where the smallest distance yields the foot point we seek:

$$\mathbf{x}_i = \arg\min_{\mathbf{x}_{i,k}} \left\|\mathbf{x}_{i,k} - \mathbf{w}_i\right\|^2.$$

3.3.3 Projection to specific quadratic curves and surfaces

Finding the foot points can be simplified further if the type of the quadratic curve or surface can be identified or constrained. Ellipses and ellipsoids, in particular, are important special cases; algorithms for other quadratic curves and surfaces can be derived in a similar manner. This section presents the outline for fast and reliable projection algorithms for ellipses [23] and ellipsoids [14]. The algorithms use few iterations and guarantee convergence to the correct projection. (The detailed algorithms for ellipses and ellipsoids, as well as other quadratic curve and surface types are discussed in Appendix A.)

3.3.3.1 Projection to an ellipse

Without loss of generality, we can assume the ellipse is in its canonical form, i.e. it is axis-aligned and centered at the origin. If not, a transformation matrix $\mathbf{M}$ that axis-aligns and centers the ellipse can be applied to the data points, and the inverse transformation matrix $\mathbf{M}^{-1}$ to the computed foot points. Thus, let the ellipse be defined as

$$Q(\mathbf{x}) = \frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} - 1 = 0.$$

Due to symmetry, it is enough to work in the first quadrant (i.e. for both data point coordinates we have $w_1 > 0$ and $w_2 > 0$), in which case the projection point will also be in the first quadrant (i.e. $x_1 > 0$ and $x_2 > 0$). Other points can be reflected to the first quadrant about the axes, and then the projection point can be reflected back.

For the distance between a data point $\mathbf{w} = \begin{bmatrix} w_1 & w_2 \end{bmatrix}$ and a foot point $\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}$ to be minimum, the distance vector must be normal to the ellipse, which means that the ellipse gradient vector and the distance vector should be equal up to magnitude. This implies (after rearrangements) that

$$x_1 = \frac{a^2 w_1}{t + a^2} \qquad x_2 = \frac{b^2 w_2}{t + b^2}$$

where t is a scalar where $t < 0$ for the inside and $t > 0$ for the outside of the ellipse. After substitutions, we have

$$\begin{aligned} Q(t) &= \frac{1}{a^2}\left(\frac{a^2 w_1}{t + a^2}\right)^2 + \frac{1}{b^2}\left(\frac{b^2 w_2}{t + b^2}\right)^2 - 1 \\ &= \left(\frac{a w_1}{t + a^2}\right)^2 + \left(\frac{b w_2}{t + b^2}\right)^2 - 1. \end{aligned}$$

which we need to solve for $Q(t) = 0$. Inspecting the first and second derivatives in the domain of interest, the function $Q(t)$ is strictly monotonic decreasing and concave, therefore a unique root t of $Q(t)$ must exist. One way to find this root is using Newton's method, which may be initialized [23] with

$$t_0 = b w_2 - b^2$$

or for faster convergence [14] with

$$t_0 = \max\left(a w_1 - a^2,\, b w_2 - b^2\right).$$

3.3.3.2 Projection to an ellipsoid

With canonical coordinates, an ellipsoid is defined as

$$Q(\mathbf{x}) = \frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} - 1 = 0$$

where for the semi-axes we have $a \geqq b \geqq c > 0$. Due to symmetry, our investigation may be restricted to $w_1 > 0$, $w_2 > 0$ and $w_3 > 0$ for data points, and likewise $x_1 > 0$, $x_2 > 0$ and $x_3 > 0$ for projection points. After substitutions, we get

$$Q(t) = \left(\frac{aw_1}{t+a^2}\right)^2 + \left(\frac{bw_2}{t+b^2}\right)^2 + \left(\frac{cw_3}{t+c^2}\right)^2 - 1$$

where we seek $Q(t) = 0$. Calculating the derivatives and inspecting behavior in the domain of interest, we find that the function $Q(t)$ is monotonically decreasing and concave. Starting Newton's method at any point t_0 where $Q(t_0) > 0$, specifically

$$t_0 = \max\left(aw_1 - a^2,\, bw_2 - b^2,\, cw_3 - c^2\right)$$

will converge to the unique solution.

3.4 Spectral clustering

One approach to identifying groups in a data set with a notion of an affinity matrix is spectral clustering [57, 51]. Spectral clustering is a non-iterative approach to split data into k groups, where the number k is known in advance. Conceptually, spectral clustering operates on an undirected complete graph where nodes correspond to data points, and edge weights correspond to the affinity measure between data points, and our goal is to find a best cut that splits the graph into k components, minimizing the weight of edges that run in between components. A typical way of generating an affinity measure is to use inverse distances, i.e. the closer the point, the larger its affinity measure.

Traditional spectral clustering is initialized with a symmetric scatter matrix $\mathbf{S}_S$ (where the subscript S stands for symmetric) with entries s_{ij} between 0 and 1, not at all similar and most similar, respectively. A possible way to convert a distance matrix (whose entries are squared distances) into a scatter matrix is via the exponential function with a negative exponent. With a symmetric distance matrix, this operation is straightforward, i.e.

$$\mathbf{S}_S = \exp(-\alpha \mathbf{A})$$

where the matrix $\mathbf{A}$ denotes a (symmetric) distance matrix and the operator $\exp(\bullet)$ is to be understood element-wise and the scaling factor $\alpha > 0$ is a scalar parameter. For algorithms that use point-to-point distances, such as the standard k-means algorithm, the matrix $\mathbf{A}$ is inherently symmetric. With an asymmetric distance matrix $\mathbf{A}$ whose entry a_{ij} represents the distance between data points $\mathbf{x}_i$ and $\mathbf{x}_j$, we might force symmetry of the scatter matrix with

$$\mathbf{S}_S = \exp\left(-\alpha \sqrt{\mathbf{A}^\top \mathbf{A}}\right)$$

where the operators $\exp(\bullet)$ and $\sqrt{\bullet}$ are again to be understood element-wise.

Once we have a scatter matrix $\mathbf{S}_S$, we may proceed as follows:

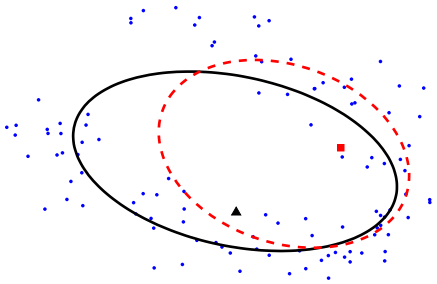


Figure 3.5: Asymmetric distance with spectral clustering. Proximity based on data projection yields an asymmetric distance measure.

- Normalize the scatter matrix $\mathbf{S}_S$. Let

$$d_i = \frac{1}{\sqrt{\sum_j [\mathbf{S}_S]_{ij}}}$$

  and $\mathbf{D}_S = \text{diag}(d_i)$ such that

$$\mathbf{H}_S = \mathbf{I} - \mathbf{D}_S \mathbf{S}_S \mathbf{D}_S$$

  .

- Compute the smallest eigenvectors of the matrix $\mathbf{H}_S = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$ where

$\mathbf{U}$	matrix of eigenvectors
$\mathbf{u}_i$	the ith smallest eigenvector
$\mathbf{U}_k$	the bottom k smallest eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \ldots$ arranged in a matrix
λ_i	the ith smallest eigenvalue
$\mathbf{\Lambda}_k$	a diagonal matrix of the bottom k smallest eigenvalues $\lambda_1, \lambda_2, \ldots$

- Perform a k-means clustering on $\mathbf{U}_k \mathbf{\Lambda}_k^{-\frac{1}{2}}$. The clusters are the result of the spectral clustering algorithm.

Unfortunately, such an approach destroys any asymmetry present in the original problem and may produce largely suboptimal results. In particular, when we identify the optimal neighborhood around each point, estimate parameters of the associated local model, and project data points to the estimated surface, the procedure we execute produces an asymmetric distance matrix. This is illustrated in Figure 3.5, in which the continuous line shows the ellipse estimated from the points that form the neighborhood of the data point with the filled triangle marker ▲, whereas the dashed line depicts the ellipse that belongs to the neighborhood of the point with the filled rectangle marker ■. Both the triangle marker point ▲

projected to the dashed line ellipse and the square marker point ■ projected to the the continuous line ellipse yield foot points that are relatively near to their originals, but these distances are usually not equal, as one can verify from the figure. This means that the distance of ▲ to ■ is not in general equal to the distance of ■ to ▲.

Formally, we may define a measure a_{ij} that represents the distance of data point $\mathbf{x}_i$ from its foot point obtained by projecting $\mathbf{x}_i$ onto the surface around $\mathbf{x}_j$, the latter of which is defined by parameters $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_j)}$.

Definition 7. Asymmetric distance measure for point–projection distances. Let

$$a_{ij} = d\left(\mathbf{x}_i, \mathbf{p}\left(\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_j)}, \mathbf{x}_i\right)\right)$$

where d denotes (point-to-point) Euclidean distance and $\mathbf{p}(\boldsymbol{\theta}, \mathbf{x}_i)$ denotes the projection of the point $\mathbf{x}_i$ to the curve or surface defined by $\boldsymbol{\theta}$, and $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_j)}$ are the parameters of the curve or surface estimated from the set of points $\mathcal{N}(\mathbf{x}_j)$, which is the local neighborhood of $\mathbf{x}_j$. Unlike in the case of the standard notion of distance, we have in general $a_{ij} \neq a_{ji}$. ■

With reference to Figure 3.5, we have

$$\begin{aligned} a_{▲■} &= d\left(▲, \mathbf{p}\left(\boldsymbol{\theta}_{\mathcal{N}(■)}, ▲\right)\right) = d\left(▲, \mathbf{p}\left(\text{- - - - -}, ▲\right)\right) = d_P\left(▲, \text{- - - - -}\right) \\ a_{■▲} &= d\left(■, \mathbf{p}\left(\boldsymbol{\theta}_{\mathcal{N}(▲)}, ■\right)\right) = d\left(■, \mathbf{p}\left(\text{———}, ■\right)\right) = d_P\left(■, \text{———}\right) \end{aligned}$$

where d_P denotes point-to-line (point-to-surface) distance (where the subscript P stands for projection).

Once we have a definition of asymmetric distance, iterating over all data points yields an asymmetric distance matrix $\mathbf{A}$ whose entry a_{ij} represents the distance of data point $\mathbf{x}_i$ from its foot point obtained by projecting $\mathbf{x}_i$ onto the surface around $\mathbf{x}_j$, defined by parameters $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_j)}$.

Definition 8. Asymmetric distance matrix. Let a_{ij} denote the asymmetric distance between points $\mathbf{x}_i$ and $\mathbf{x}_j$. The asymmetric distance matrix is defined as

$$\mathbf{A} = \left[a_{ij}\right].$$

■

It is natural to carry over the asymmetry in the distance measure to the affinity matrix.

Definition 9. Asymmetric affinity measure. With an asymmetric distance matrix $\mathbf{A}$, the asymmetric affinity measure may be defined as

$$\mathbf{S}_A = \exp\left(-\alpha \mathbf{A}\right)$$

where the operator $\exp(\bullet)$ is to be understood element-wise and $\alpha > 0$ is a scalar parameter. (The subscript A in $\mathbf{S}_A$ stands for asymmetric.) ■

With an asymmetric affinity measure, we may employ an algorithm that exploits the distance asymmetry in the original problem. Such an approach has been successfully applied in various disciplines. For instance, analyzing pen trajectories of handwritten characters collected using a graphics tablet leads to an asymmetric affinity measure [7]. Here, the data consists of the velocity and pen tip force information obtained from the device, and directed dynamic time warping distance may be used to compare pen trajectories to a set of reference trajectories. Dynamic time warping is an algorithm for measuring similarity between two sequences which may vary in time or speed to find an optimal match between them. The sequences are warped non-linearly in the time dimension to determine a measure of their similarity independent of certain nonlinear time-domain variations. The asymmetry arises from the different roles the trajectories have: the actual trajectories to be matched are always compared to the provably correct reference trajectories. Similar reasoning may be applied to vehicle trajectories tracked at an intersection even if the distance measure used is different [7].

Asymmetric spectral clustering, an algorithm to find a best cut in a weighted graph [50], remedies the asymmetry issue by procrastinating enforcing symmetry to a later phase of the algorithm. Let $\mathbf{S}_A$ be the (asymmetric) matrix initialized with entries s_{ij} between 0 and 1, not at all similar and most similar, respectively. The steps of the algorithm are as follows:

- Normalize the matrix $\mathbf{S}_A$. Let
$$d_i = \sqrt{\frac{1}{\sum_j [\mathbf{S}_A]_{ij}}}$$
and $\mathbf{D}_A = \operatorname{diag}(d_i)$ such that
$$\mathbf{H}_A = \mathbf{I} - \frac{1}{2}\mathbf{D}_A(\mathbf{S}_A + \mathbf{S}_A^\top)\mathbf{D}_A$$

- Compute the smallest eigenvectors of the scatter matrix $\mathbf{H}_A = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$ where $\mathbf{U}$ is a matrix of eigenvectors and a $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues.

- Normalize the rows of $\mathbf{U}_k$ to unit length. $\mathbf{U}_k$ is a matrix of the bottom k smallest eigenvectors with $\mathbf{u}_1, \mathbf{u}_2, \ldots$ arranged in a matrix.

- Perform a k-means clustering on normalized $\mathbf{U}_k$.

Note that even while asymmetric spectral clustering also starts with a symmetric matrix $\mathbf{S}_A + \mathbf{S}_A^\top$, the normalization matrix $\mathbf{D}_A$ is computed based on the asymmetric matrix $\mathbf{S}_A$, unlike (symmetric) spectral clustering where the normalization $\mathbf{D}_S$ is based on the already symmetric $\mathbf{S}_S$. The row index i of the matrix $\mathbf{S}_A$ represents data points and the column index j represents estimated curves or surfaces, and the entry $[\mathbf{S}_A]_{ij}$ can be interpreted as a support weight of the data point i in favor of the estimated curve or surface j. The normalization $\mathbf{D}_A$ makes all data points cast unit support in favor of the total number of estimated curves and surfaces, and the asymmetric clustering tends to avoid separating the data point from the curve or surface it has large support for.

3.5 Summary

In this chapter, we have extended our scope of the parameter estimation problem and explored the field when data are no longer covered by a single nonlinear function but a union of such functions. This has led us to the clustering problem, where our task is not merely estimating the unknown parameters of the function that relates the data but also discovering a feasible partitioning the data.

We have seen an iterative algorithm for the so-called manifold clustering problem, which mimics the standard iterative algorithm for k-means, albeit the clustering objective, the update and assignment steps are different. The algorithm is initialized with random points, the neighborhood of which gives the first manifold parameter estimates for the algorithm. The update step computes model parameter estimates from a scattered point cloud, assuming an assignment of points to groups, whereas the assignment step amounts to a projection to the closest manifold whose parameters have been computed by the update step. Projection is a computationally expensive step but for the important special cases of quadratic curves and surfaces, efficient algorithms have been shown to lead to fast convergence.

Iterative algorithms require initialization and possibly re-sampling to avoid the algorithm converging to a sub-optimal solution. Non-iterative algorithms, in contrast, can reliably deliver a best-possible solution. We have seen how the choice of an appropriate asymmetric distance measure and in turn asymmetric spectral clustering on the resulting affinity matrix may help us craft a non-iterative manifold clustering algorithm, which may both serve as input for an iterative algorithm and be a clustering algorithm of its own.

New scientific contributions related to the clustering problem include:

- an iterative algorithm for manifold clustering based on alternating optimization;
- utilizing unconstrained and constrained parameter estimation methods for the update step of the clustering algorithm;
- applying efficient projection algorithms to facilitate the assignment step of the clustering algorithm;
- a non-iterative algorithm for manifold clustering supported by spectral clustering;
- an asymmetric distance measure that leads to a sound pairwise distance measure between data points;
- applying asymmetric spectral clustering for minimizing the distance of intra-cluster data points.

CHAPTER

4 Dynamic systems

Parameter estimation of linear discrete-time dynamic errors-in-variables systems where the system description is known up to a few model parameters is a classical problem when Gaussian noise contaminates both input and output observations but the ratio of these noise contributions is a priori known. In such a case, a maximum likelihood estimator is the standard tool to derive the provably best attainable parameter estimates from second-order statistical characteristics. The maximum likelihood estimator takes noise-polluted input observations $\mathbf{u}$, noise-polluted output observations $\mathbf{y}$, relative noise distribution information represented as an angle $\varphi \in [0, \frac{\pi}{2}]$ (ranging from no output noise through equal input and output noise to no input noise), and produces model parameter estimates $\hat{\mathbf{g}}$ that approximate the true model parameters $\mathbf{g}$, and an estimate $\hat{\mu}$ for the noise magnitude μ. In contrast, a situation where no information on the relative noise distribution φ is available is recognized as a more difficult problem and the maximum likelihood estimator cannot be employed. In fact, it turns out that under general assumptions, the system is not identifiable, or put alternatively, it produces many equivalent results [58, 3].

While there has been more substantial research into estimating model and noise parameters of linear dynamic errors-in-variables systems, less attention has been paid to nonlinear systems, some exceptions are [43, 66]. However, most physical systems are inherently nonlinear, even if they are usually approximated with linear models for a particular operating range. A polynomial approximation being a natural approach to deal with a nonlinear system, polynomial models allow more satisfactory approximation for nonlinear processes over wider ranges of operation, increasing the accuracy and applicability of the models [54].

This chapter explores an estimation scheme for dynamic errors-in-variables systems that are captured with the separable implicit relationship

$$\mathbf{f}_{data}^{\top}(\bar{\mathbf{x}}_{m,k})\mathbf{f}_{par}(\mathbf{g}) = 0$$

where $\mathbf{f}_{data} : \mathbb{R}^{2m} \to \mathbb{R}^{2n}$ is a polynomial lifting (linearization) function that maps the (input and output) observation vector

$$\mathbf{x}_{m,k} = \begin{bmatrix} y_k & \dots & y_{k-m} & u_k & \dots & u_{k-m} \end{bmatrix}^{\top}$$

of a dynamic system with order m to n linear and nonlinear components. For simplicity, we assume the nonlinear components are polynomial functions of input and output variables.

Example 11. Observations of a dynamic system transformed by a lifting function. Suppose a dynamic system is described with the following difference equation:

$$\begin{aligned} \bar{y}_k &= g_1 \bar{y}_{k-1} + g_2 \bar{y}_{k-2} + g_3 \bar{u}_{k-1} + \\ &+ g_4 \bar{u}_{k-1}^2 + g_5 \bar{y}_{k-1} \bar{y}_{k-2} \\ &+ g_6 \bar{u}_{k-1} \bar{y}_{k-1} \end{aligned}$$

comprising of both linear and polynomial terms as well as cross-correlating terms. Clearly, the lifting function for data is

$$\mathbf{f}_{data}(\mathbf{x}_k) = \begin{bmatrix} \bar{y}_k & \bar{y}_{k-1} & \bar{y}_{k-2} & \bar{u}_{k-1} & \bar{u}_{k-1}^2 & \bar{y}_{k-1}\bar{y}_{k-2} & \bar{u}_{k-1}\bar{y}_{k-1} \end{bmatrix}^\top.$$

♣

In order to give background for nonlinear methods, we shall first look into existing methods for estimating parameters of linear dynamic errors-in-variables systems in Section 4.1, and the challenges such an estimation problem poses. Linear methods can be categorized into two main groups: methods with a known relative noise distribution between input and output, and methods without such information. The Koopmans–Levin method (Section 4.1.1), the maximum likelihood method for dynamic systems (Section 4.1.2) and the generalized Koopmans–Levin method (Section 4.1.3), which alloys the two, fall into the first group with known relative noise distribution. In contrast, bias-compensated least squares (Section 4.1.4) and the Frisch scheme (Section 4.1.5) are able to estimate model and noise parameters simultaneously. After discussing linear errors-in-variables systems, we shall move on to nonlinear systems in Section 4.2, in which the polynomial bias-compensated least squares (Section 4.1.4), which is known from literature, will be compared to a polynomial extension of the generalized Koopmans–Levin method (Section 4.2.2), which is a new contribution in this thesis.

4.1 Linear systems

Consider a linear discrete-time single-input single-output (SISO) errors-in-variables system $G(q^{-1})$, which is described by the so-called *difference equation*

$$A(q^{-1})\,\bar{y}_k = B(q^{-1})\,\bar{u}_k \tag{4.1}$$

where $\bar{y}_k$ and $\bar{u}_k$ are the (noise-free) output and input data, respectively, at time instant k, and q^{-1} denotes the backward shift operator such that $q^{-1}\,\bar{u}_k = \bar{u}_{k-1}$ and

$$\begin{aligned} A(q^{-1}) &= 1 + a_1 q^{-1} + \cdots + a_{ma} q^{-ma} \\ B(q^{-1}) &= b_1 q^{-1} + \cdots + b_{mb} q^{-mb} \end{aligned}$$

such that the difference equation (4.1) expands to

$$\bar{y}_k + a_1\bar{y}_{k-1} + \cdots + a_{ma}\bar{y}_{k-ma} = b_1\bar{u}_{k-1} + \cdots + b_{mb}\bar{u}_{k-mb}. \tag{4.2}$$

For simplicity, we assume that $m = ma = mb$, but we will see it later that it is relatively easy to relax this assumption. For notational convenience, observations and parameters can be stacked into vectors **y** and **u** as well as **a** and **b**, respectively.

Definition 10. Let

$$\begin{aligned} \mathbf{x}^y_{m,k} &= \begin{bmatrix} y_k & \dots & y_{k-m} \end{bmatrix}^\top \\ \mathbf{x}^u_{m,k} &= \begin{bmatrix} u_k & \dots & u_{k-m} \end{bmatrix}^\top \end{aligned}$$

where m is the known order (memory) of the dynamic system. Then the *observation vector* at time instant k is

$$\mathbf{x}_{m,k} = \begin{bmatrix} \mathbf{x}^y_{m,k} \\ \mathbf{x}^u_{m,k} \end{bmatrix}.$$

■

Definition 11. Let

$$\begin{aligned} \boldsymbol{\phi}^y_{m,k} &= \begin{bmatrix} y_{k-1} & \dots & y_{k-m} \end{bmatrix}^\top \\ \boldsymbol{\phi}^u_{m,k} &= \begin{bmatrix} u_{k-1} & \dots & u_{k-m} \end{bmatrix}^\top \end{aligned}$$

where m is the known order (memory) of the dynamic system. Then the *least-squares observation vector* at time instant k is

$$\boldsymbol{\phi}_{m,k} = \begin{bmatrix} \boldsymbol{\phi}^y_{m,k} \\ \boldsymbol{\phi}^u_{m,k} \end{bmatrix}.$$

■

Definition 12. Let

$$\begin{aligned} \mathbf{g}^a &= \begin{bmatrix} 1 & a_1 & \dots & a_m \end{bmatrix}^\top \\ \mathbf{g}^b &= \begin{bmatrix} 0 & -b_1 & \dots & -b_m \end{bmatrix}^\top. \end{aligned}$$

Then the *model parameter vector* for the investigated system is

$$\mathbf{g} = \begin{bmatrix} \mathbf{g}^a \\ \mathbf{g}^b \end{bmatrix}.$$

■

Using the compact definition for the observation vector and the model parameter vector, the system description in (4.2) simplifies into

$$\bar{\mathbf{x}}^\top_{m,k}\mathbf{g} = 0$$

where $\bar{\mathbf{x}}_{m,k}$ comprises of noise-free data. From the formula above, it can be seen that the model parameter vector is invariant to scaling, i.e. it is only defined up to multiplication by a scalar. Hence, we assume that $\|\mathbf{g}\|^2 = \mathbf{g}^\top \mathbf{g} = 1$.

However, in the errors-in-variables context, one is never able to directly observe the true data vector, only its noise-contaminated equivalent $\mathbf{x}_{m,k} = \bar{\mathbf{x}}_{m,k} + \tilde{\mathbf{x}}_{m,k}$ where $\tilde{\mathbf{x}}_{m,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_{\mu,\varphi,m})$ is a zero-mean Gaussian noise contribution with covariance $\mathbf{C}_{\mu,\varphi,m}$.

Definition 13. Let σ_u^2 and σ_y^2 be the variances of the noise that pollutes input and output observations, respectively. Let the normalized unit noise covariance be

$$\mathbf{C}_\varphi = \begin{bmatrix} \sin^2\varphi & 0 \\ 0 & \cos^2\varphi \end{bmatrix}$$

and the (unnormalized) unit noise covariance be

$$\mathbf{C}_{\mu,\varphi} = \mu \mathbf{C}_\varphi$$

where μ is the measure of noise "magnitude" and φ is the measure of noise "direction" (i.e. relative distribution of noise between input and output, with the extreme cases $\varphi = 0$ being input-only noise and $\varphi = \frac{\pi}{2}$ being output-only noise). Then the *normalized noise covariance matrix* is

$$\mathbf{C}_{\varphi,m} = \mathbf{C}_\varphi \otimes \mathbf{I}_m$$

and the *noise covariance matrix* is

$$\mathbf{C}_{\mu,\varphi,m} = \left(\mu \mathbf{C}_\varphi\right) \otimes \mathbf{I}_m.$$

■

The best understood estimation methods for identifying linear dynamic errors-in-variables systems are perhaps those that assume a known noise ratio, i.e. a known relative distribution of noise between input and output (up to magnitude). We shall investigate three strategies for estimating model parameters and noise magnitude. Section 4.1.1 introduces the simplest strategy inspired by the (linear) Koopmans estimator [40] that exhibits the most crude accuracy. Section 4.1.2 discusses the maximum likelihood estimator that delivers the best possible estimates but is computationally cumbersome. Section 4.1.3 shows an estimation strategy that is a compromise between the two, a scalable trade-off between the Koopmans approach and the maximum likelihood approach.

Unfortunately, not every estimation problem assumes a known relative distribution of noise between input and output. In a more general linear errors-in-variables problem for dynamic systems, we have to estimate both model parameters as well as noise parameters. In what follows, two methods that tackle the problem of unknown noise ratio will be investigated: the bias-compensating least-squares estimator (Section 4.1.4) and the Frisch scheme (Section 4.1.5).

4.1.1 Koopmans–Levin estimator

One of the simple methods to devise model parameter estimates when the normalized unit noise covariance matrix $\mathbf{C}_\varphi$ is known has been proposed by Levin [44] extending the work of Koopmans [40] for static systems, and can be termed the *Koopmans–Levin method*. This is a non-iterative but fairly inaccurate method to estimate model parameters from second-order statistical characteristics.

For convenience, let us introduce the observation matrix and the observation (sample) covariance matrix.

Definition 14. Observation matrix

$$\mathbf{X}_m = \begin{bmatrix} y_{m+1} & \dots & y_1 & u_{m+1} & \dots & u_1 \\ y_{m+2} & \dots & y_2 & u_{m+2} & \dots & u_2 \\ \vdots & & \vdots & \vdots & & \vdots \\ y_N & \dots & y_{N-m} & u_N & \dots & u_{N-m} \end{bmatrix}_{N-m,2(m+1)}$$

■

Definition 15. Observation covariance matrix

$$\begin{aligned} \mathbf{D}_m &= \mathbb{E}\left\{(\mathbf{x}_m - \mathbb{E}\mathbf{x}_m)\,(\mathbf{x}_m - \mathbb{E}\mathbf{x}_m)^\top\right\} \\ &\approx \frac{1}{N}\sum_{k=1}^{N}\left(\mathbf{x}_{m,k} - \frac{1}{N}\sum_{k=1}^{N}\mathbf{x}_{m,k}\right)\left(\mathbf{x}_{m,k} - \frac{1}{N}\sum_{k=1}^{N}\mathbf{x}_{m,k}\right)^\top \end{aligned}$$

■

The Koopmans–Levin estimator uses the principle that

$$\mathbf{D}_m = \bar{\mathbf{D}}_m + \mathbf{C}_{\mu,\varphi,m}$$

i.e. the observation covariance matrix can be decomposed into a contribution by the noise-free data $\bar{\mathbf{D}}_m$ and the noise contribution $\mathbf{C}_{\mu,\varphi,m}$. Multiplying by $\mathbf{g}$ from both sides,

$$\mathbf{g}^\top\mathbf{D}_m\mathbf{g} = \mathbf{g}^\top\bar{\mathbf{D}}_m\mathbf{g} + \mathbf{g}^\top\mathbf{C}_{\mu,\varphi,m}\mathbf{g} = \mathbf{g}^\top\mathbf{C}_{\mu,\varphi,m}\mathbf{g}$$

as

$$\mathbf{g}^\top\bar{\mathbf{D}}_m\mathbf{g} = \mathbf{g}^\top\bar{\mathbf{X}}_m^\top\bar{\mathbf{X}}_m\mathbf{g}$$

where $\bar{\mathbf{X}}_m\mathbf{g} = \mathbf{0}$ by definition.

Finding a model parameter vector $\mathbf{g}$ that satisfies

$$\begin{aligned} \mathbf{g}^\top\mathbf{D}_m\mathbf{g} &= \mathbf{g}^\top\mu\mathbf{C}_{\varphi,m}\mathbf{g} \\ \mathbf{g}^\top\mathbf{g} &= 1 \end{aligned}$$

is a generalized eigenvalue problem and the optimum solution is the eigenvector $\mathbf{g}$ that belongs to the smallest eigenvalue μ.

In the same fashion as with total least-squares estimation for the static case [20], the accuracy of the estimates can be improved if we use singular value decomposition instead of eigenvalue decomposition. Reformulating,

$$\begin{aligned} \mathbf{g}^\top \mathbf{X}_m^\top \mathbf{X}_m \mathbf{g} &= \mathbf{g}^\top \mu \mathbf{L}_{\varphi,m}^\top \mathbf{L}_{\varphi,m} \mathbf{g} \\ \mathbf{g}^\top \mathbf{g} &= 1 \end{aligned}$$

where $\mathbf{L}_{\varphi,m}^\top \mathbf{L}_{\varphi,m} = \mathbf{C}_{\varphi,m}$, hence the parameters are estimated as a generalized singular value problem on the matrix pair $(\mathbf{X}_m, \mathbf{L}_m)$. The right singular vector $\mathbf{g}$ that belongs to the smallest singular value s where $s^2 = \mu$ is the parameter vector we seek.

4.1.2 Maximum likelihood estimator

The maximum likelihood estimator maximizes the likelihood function which can be formulated for the dynamic case as

$$p(\mathbf{x}_N \mid \mathbf{g}) \propto \exp\left(-\frac{1}{2}(\mathbf{x}_N - \bar{\mathbf{x}}_N)^\top \mathbf{C}_N^{-1}(\mathbf{x}_N - \bar{\mathbf{x}}_N)\right) \tag{4.3}$$

where N is the number of observations,

$$\begin{aligned} \mathbf{x}_N &= \begin{bmatrix} \mathbf{x}_N^y \\ \mathbf{x}_N^u \end{bmatrix} \\ \mathbf{x}_N^y &= \begin{bmatrix} y_N & \dots & y_1 \end{bmatrix}^\top \\ \mathbf{x}_N^u &= \begin{bmatrix} u_N & \dots & u_1 \end{bmatrix}^\top \end{aligned}$$

and $\bar{\mathbf{x}}_N$ is defined likewise, and $\mathbf{C}_N = (\mu \mathbf{C}_\varphi) \otimes I_N$.

Definition 16. Let

$$\mathbf{G}_N^a = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ a_1 & 1 & 0 & \dots & 0 & 0 \\ a_2 & a_1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_m & a_{m-1} & a_{m-2} & \dots & 0 & 0 \\ 0 & a_m & a_{m-1} & \dots & 0 & 0 \\ 0 & 0 & a_m & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & a_m & a_{m-1} \\ 0 & 0 & 0 & \dots & 0 & a_m \end{bmatrix}_{N,N-m}$$

wherein $\mathbf{G}_N^a$ is a banded Toeplitz matrix of parameters a_i. Let $\mathbf{G}_N^b$ be defined in a similar way in terms of b_i. Then

$$\mathbf{G}_N = \begin{bmatrix} \mathbf{G}_N^a \\ -\mathbf{G}_N^b \end{bmatrix}_{2N,N-m}$$

is the *model parameter matrix* for the maximum likelihood estimator. ■

Using the model parameter matrix, it follows that $\bar{\mathbf{x}}_N^\top \mathbf{G}_N = \mathbf{0}$ for noise-free data and true model parameters. Taking this constraint into consideration,

$$\begin{aligned} J_{ML} &= \frac{1}{2}(\mathbf{x}_N - \bar{\mathbf{x}}_N)^\top \mathbf{C}_N^{-1}(\mathbf{x}_N - \bar{\mathbf{x}}_N) \\ &= \frac{1}{2}(\mathbf{x}_N - \bar{\mathbf{x}}_N)^\top \mathbf{G}_N \left(\mathbf{G}_N^\top \mathbf{C}_N \mathbf{G}_N\right)^{-1} \mathbf{G}_N^\top (\mathbf{x}_N - \bar{\mathbf{x}}_N) \\ &= \frac{1}{2}\mathbf{x}_N^\top \mathbf{G}_N \left(\mathbf{G}_N^\top \mathbf{C}_N \mathbf{G}_N\right)^{-1} \mathbf{G}_N^\top \mathbf{x}_N \end{aligned}$$

the likelihood function (4.3) is equivalent to minimizing the loss function

$$J_{ML} = \frac{1}{2}\mathbf{x}_N^\top \mathbf{G}_N \left(\mathbf{G}_N^\top \mathbf{C}_N \mathbf{G}_N\right)^{-1} \mathbf{G}_N^\top \mathbf{x}_N. \tag{4.4}$$

As a result, estimating the parameters of the system is achieved by minimizing J_{ML} in (4.4).

However, the matrices involved in J_{ML} are rather large and devising efficient algorithms that estimate its true value is not straightforward, see [67] for a possible approach.

4.1.3 Generalized Koopmans–Levin estimator

The Koopmans–Levin estimator provides a simple non-iterative way to estimate model parameters but the variance of estimates is fairly large. Meanwhile, the maximum likelihood estimation approach is much more robust but involves more unknowns, hence entails a greater computational complexity, and is typically formulated with an iterative approach. The generalized Koopmans–Levin estimator [65] unifies the Koopmans–Levin and maximum likelihood algorithms. The unified algorithm incorporates a scaling parameter q that allows us to freely trade estimation accuracy for efficiency.

Definition 17. Let q be an integer such that $m \leqq q < N$, where m is the system model order (memory) and N is the number of observations. Then the *extended observation vector* at time instant k is

$$\mathbf{x}_{q,k} = \begin{bmatrix} y_k & \dots & y_{k-q} & u_k & \dots & u_{k-q} \end{bmatrix}^\top.$$

■

Apparently, the appropriate choice of q returns the observation vector of the Koopmans–Levin estimator and the maximum likelihood estimator as special cases: $q = m$ is the observation vector the Koopmans–Levin method uses, while $q = N - 1$ yields the single large vector in the likelihood function. Following the principle of the extended observation vector, extended versions of the observation matrix and the observation covariance matrix may be introduced.

Definition 18. Extended observation matrix

$$\mathbf{X}_q = \begin{bmatrix} y_{q+1} & \dots & y_1 & u_{q+1} & \dots & u_1 \\ y_{q+2} & \dots & y_2 & u_{q+2} & \dots & u_2 \\ \vdots & & \vdots & \vdots & & \vdots \\ y_N & \dots & y_{N-q} & u_N & \dots & u_{N-q} \end{bmatrix}_{N-q,\,2(q+1)}$$

■

Definition 19. Let

$$\mathbf{G}_q^a = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ a_1 & 1 & 0 & \dots & 0 & 0 \\ a_2 & a_1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_m & a_{m-1} & a_{m-2} & \dots & 0 & 0 \\ 0 & a_m & a_{m-1} & \dots & 0 & 0 \\ 0 & 0 & a_m & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & a_m & a_{m-1} \\ 0 & 0 & 0 & \dots & 0 & a_m \end{bmatrix}_{q,\, q-m}$$

where $\mathbf{G}_q^a$ is banded Toeplitz matrix of parameters a_i. Assuming $\mathbf{G}_q^b$ can be constructed in a similar manner, the *extended model parameter matrix* is

$$\mathbf{G}_q = \begin{bmatrix} \mathbf{G}_q^a \\ -\mathbf{G}_q^b \end{bmatrix}.$$

■

Similarly to the case of the maximum likelihood estimation, $\bar{\mathbf{X}}_q\mathbf{G}_q = \mathbf{0}$ is satisfied. Likewise, matrices $\mathbf{D}_q$ and $\mathbf{C}_q$ can be defined in analogy to $\mathbf{D}_m$ and $\mathbf{C}_m$. With these notations at hand, the loss function of the generalized Koopmans–Levin estimator takes the form

$$J_{GKL} = \frac{1}{2}\text{trace}\left\{\left(\mathbf{G}_q^\top\mathbf{C}_q\mathbf{G}_q\right)^{-1}\mathbf{G}_q^\top\mathbf{D}_q\mathbf{G}_q\right\}. \tag{4.5}$$

Straightforward calculations show that J_{GKL} yields J_{KL} and J_{ML} with the special choices $q = m$ and $q = N-1$. First, let $q = m$, in which case $\mathbf{G}_m = \mathbf{g}$, i.e.

$$J_{GKL} = \frac{1}{2}\text{trace}\left\{\left(\mathbf{g}^\top\mathbf{C}_m\mathbf{g}\right)^{-1}\mathbf{g}^\top\mathbf{D}_m\mathbf{g}\right\} = \frac{\mathbf{g}^\top\mathbf{D}_m\mathbf{g}}{\mathbf{g}^\top\mathbf{C}_m\mathbf{g}} = J_{KL}$$

as the terms $\mathbf{g}^\top\mathbf{C}_m\mathbf{g}$ and $\mathbf{g}^\top\mathbf{D}_m\mathbf{g}$ involved are scalars.

Second, using the identity $\text{trace}(AB) = \text{trace}(BA)$,

$$J_{GKL} = \frac{1}{2}\text{trace}\left\{\mathbf{X}_q\mathbf{G}_q\left(\mathbf{G}_q^\top\mathbf{C}_q\mathbf{G}_q\right)^{-1}\mathbf{G}_q^\top\mathbf{X}_q^\top\right\}.$$

Let $q = N-1$, in which case $\mathbf{X}_q = \mathbf{x}_N$ where

$$\mathbf{x}_N^\top = \begin{bmatrix} y_N & \dots & y_1 & u_N & \dots & u_1 \end{bmatrix}$$

and therefore

$$\begin{aligned} J_{GKL} &= \frac{1}{2}\text{trace}\left\{\mathbf{x}_N^\top\mathbf{G}_N\left(\mathbf{G}_N^\top\mathbf{C}_N\mathbf{G}_N\right)^{-1}\mathbf{G}_N^\top\mathbf{x}_N\right\} \\ &= \frac{1}{2}\mathbf{x}_N^\top\mathbf{G}_N\left(\mathbf{G}_N^\top\mathbf{C}_N\mathbf{G}_N\right)^{-1}\mathbf{G}_N^\top\mathbf{x}_N = J_{ML}, \end{aligned}$$

again with omitting the trace operator on a scalar argument.

Minimizing the objective function with an iterative scheme

In order to avoid directly minimizing the objective function 4.5, it is possible to employ an iteration scheme. Upon each iteration, the objective function

$$e = \frac{\operatorname{trace}\left\{\left(\mathbf{G}_{q,(k)}^{\top}\mathbf{C}_q\mathbf{G}_{q,(k)}\right)^{-1}\mathbf{G}_{q,(k+1)}^{\top}\mathbf{D}_q\mathbf{G}_{q,(k+1)}\right\}}{\operatorname{trace}\left\{\left(\mathbf{G}_{q,(k)}^{\top}\mathbf{C}_q\mathbf{G}_{q,(k)}\right)^{-1}\mathbf{G}_{q,(k+1)}^{\top}\mathbf{C}_q\mathbf{G}_{q,(k+1)}\right\}}$$

is minimized where $\mathbf{G}_{q,(k)} = \mathbf{G}_q(\mathbf{g}_{(k)})$ comes from the previous iteration and

$$\mathbf{G}_{q,(k+1)} = \mathbf{G}_q(\mathbf{g}_{(k+1)})$$

is the variable with respect to which the optimization is performed. For brevity,

$$\begin{aligned}\mathbf{L}^{\top}\mathbf{L} &= \mathbf{L}_q^{\top}\mathbf{L}_q = \mathbf{C}_q\\ \mathbf{G} &= \mathbf{G}_{q,(k+1)}\\ \mathbf{g} &= \mathbf{g}_{(k+1)}\end{aligned}$$

and use QR-decomposition on the matrix $\mathbf{M}^{\top}\mathbf{M} = \mathbf{R}^{\top}\mathbf{Q}^{\top}\mathbf{Q}\mathbf{R} = \mathbf{R}^{\top}\mathbf{R}$ such that

$$\begin{aligned}\mathbf{Q}_{CG}\mathbf{R}_{CG} &= \mathbf{L}_q\mathbf{G}_{q,k}\\ \mathbf{Q}_X\mathbf{R}_X &= \mathbf{X}_q\end{aligned}$$

yielding

$$\begin{aligned}e &= \frac{\operatorname{trace}\left\{\left(\mathbf{R}_{CG}^{\top}\mathbf{R}_{CG}\right)^{-1}\mathbf{G}^{\top}\mathbf{R}_X^{\top}\mathbf{R}_X\mathbf{G}\right\}}{\operatorname{trace}\left\{\left(\mathbf{R}_{CG}^{\top}\mathbf{R}_{CG}\right)^{-1}\mathbf{G}^{\top}\mathbf{L}^{\top}\mathbf{L}\mathbf{G}\right\}}\\ &= \frac{\operatorname{trace}\left\{\mathbf{R}_{CG}^{-\top}\mathbf{G}^{\top}\mathbf{R}_X^{\top}\mathbf{R}_X\mathbf{G}\mathbf{R}_{CG}^{-1}\right\}}{\operatorname{trace}\left\{\mathbf{R}_{CG}^{-\top}\mathbf{G}^{\top}\mathbf{L}^{\top}\mathbf{L}\mathbf{G}\mathbf{R}_{CG}^{-1}\right\}}\\ &= \frac{\operatorname{vec}\left\{\mathbf{R}_X\mathbf{G}\mathbf{R}_{CG}^{-1}\right\}^{\top}\operatorname{vec}\left\{\mathbf{R}_X\mathbf{G}\mathbf{R}_{CG}^{-1}\right\}}{\operatorname{vec}\left\{\mathbf{L}\mathbf{G}\mathbf{R}_{CG}^{-1}\right\}^{\top}\operatorname{vec}\left\{\mathbf{L}\mathbf{G}\mathbf{R}_{CG}^{-1}\right\}}\\ &= \frac{\operatorname{vec}\{\mathbf{G}\}^{\top}\left(\mathbf{R}_{CG}^{-1}\otimes\mathbf{R}_X^{\top}\right)\left(\mathbf{R}_{CG}^{-\top}\otimes\mathbf{R}_X\right)\operatorname{vec}\{\mathbf{G}\}}{\operatorname{vec}\{\mathbf{G}\}^{\top}\left(\mathbf{R}_{CG}^{-1}\otimes\mathbf{L}^{\top}\right)\left(\mathbf{R}_{CG}^{-\top}\otimes\mathbf{L}\right)\operatorname{vec}\{\mathbf{G}\}}\\ &= \frac{\mathbf{g}^{\top}\mathbf{T}^{\top}\left(\mathbf{R}_{CG}^{-1}\otimes\mathbf{R}_X^{\top}\right)\left(\mathbf{R}_{CG}^{-\top}\otimes\mathbf{R}_X\right)\mathbf{T}\mathbf{g}}{\mathbf{g}^{\top}\mathbf{T}^{\top}\left(\mathbf{R}_{CG}^{-1}\otimes\mathbf{L}^{\top}\right)\left(\mathbf{R}_{CG}^{-\top}\otimes\mathbf{L}\right)\mathbf{T}\mathbf{g}}.\end{aligned}$$

where $\mathbf{A}^{-\top} = \left(\mathbf{A}^{-1}\right)^{\top}$ and $\mathbf{T}$ is a sparse matrix of zeros and ones chosen such that $\operatorname{vec}(\mathbf{G}_q) = \mathbf{T}\mathbf{g}$. The problem is tackled as a generalized singular value problem on the matrix pair

$$\left\{\left(\mathbf{R}_{CG}^{-\top}\otimes\mathbf{R}_X\right)\mathbf{T},\ \left(\mathbf{R}_{CG}^{-\top}\otimes\mathbf{L}_q\right)\mathbf{T}\right\}$$

where the right singular vector corresponding to the smallest singular value is of interest.

4.1.4 Bias-compensating least-squares estimator

An estimator for dynamic systems that can simultaneously estimate model and noise parameters is the bias-compensating least-squares estimator. Least-squares estimators are based on the linear regression principle, which can be concisely written as an overdetermined system of equations

$$\mathbf{y} = \boldsymbol{\Phi}\boldsymbol{\theta} + \boldsymbol{\varepsilon}$$

where $\boldsymbol{\Phi}$ is an N-by-$2m$ matrix representing N samples of $\boldsymbol{\phi}_{m,k}$ and the N-by-1 vector $\mathbf{y}$ corresponds to N samples of y_k, and $\boldsymbol{\varepsilon}$ represents the error. The least-squares estimate known from statistical literature can then be formulated for this case as

$$\hat{\boldsymbol{\theta}}_{LS} = \boldsymbol{\Phi}^{\dagger}\mathbf{y} = (\boldsymbol{\Phi}^{\top}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\top}\mathbf{y} \tag{4.6}$$

where $\mathbf{M}^{\dagger}$ denotes the Moore–Penrose generalized inverse of $\mathbf{M}$. In fact, the least-squares estimate minimizes the expectation of the squared error, that is,

$$\hat{\boldsymbol{\theta}}_{LS} = \arg\min_{\boldsymbol{\theta}} \mathbb{E}\varepsilon^2 = \arg\min_{\boldsymbol{\theta}} \mathbb{E}\left(y - \boldsymbol{\phi}^{\top}\boldsymbol{\theta}\right)^2.$$

Unfortunately, least-squares estimation method gives consistent estimates only under restrictive conditions, notably, only in the equation error case. Assuming white measurement noises,

$$\boldsymbol{\Phi}^{\top}\boldsymbol{\Phi} = \bar{\boldsymbol{\Phi}}^{\top}\bar{\boldsymbol{\Phi}} + \tilde{\boldsymbol{\Phi}}^{\top}\tilde{\boldsymbol{\Phi}}$$

where $\tilde{\boldsymbol{\Phi}}$ is the noise contribution, and

$$\boldsymbol{\Phi}^{\top}\mathbf{y} = \bar{\boldsymbol{\Phi}}^{\top}\bar{\mathbf{y}} = \bar{\boldsymbol{\Phi}}^{\top}\bar{\boldsymbol{\Phi}}\boldsymbol{\theta}$$

as $\bar{\mathbf{y}} = \bar{\boldsymbol{\Phi}}\boldsymbol{\theta}$ according to the system model. As a result,

$$\boldsymbol{\Phi}^{\top}\boldsymbol{\Phi}\hat{\boldsymbol{\theta}}_{LS} = \boldsymbol{\Phi}^{\top}\mathbf{y} = \bar{\boldsymbol{\Phi}}^{\top}\bar{\boldsymbol{\Phi}}\boldsymbol{\theta} = \left(\boldsymbol{\Phi}^{\top}\boldsymbol{\Phi} - \tilde{\boldsymbol{\Phi}}^{\top}\tilde{\boldsymbol{\Phi}}\right)\boldsymbol{\theta}, \tag{4.7}$$

which means that $\hat{\boldsymbol{\theta}}_{LS}$ is biased due to $\tilde{\boldsymbol{\Phi}}^{\top}\tilde{\boldsymbol{\Phi}}$.

The principle of bias compensated least-squares [74] methods is to adjust the least-squares estimate to eliminate the bias due to $\tilde{\boldsymbol{\Phi}}^{\top}\tilde{\boldsymbol{\Phi}}$. Consequently,

$$\hat{\boldsymbol{\theta}}_{BCLS} = \left(\boldsymbol{\Phi}^{\top}\boldsymbol{\Phi} - \tilde{\boldsymbol{\Phi}}^{\top}\tilde{\boldsymbol{\Phi}}\right)^{-1}\boldsymbol{\Phi}^{\top}\mathbf{y} \tag{4.8}$$

in which the unknown $\tilde{\boldsymbol{\Phi}}^{\top}\tilde{\boldsymbol{\Phi}}$, which depends on the noise parameters σ_y^2 and σ_u^2 has to be estimated in some way. Therefore, once the noise parameters σ_y^2 and σ_u^2 are substituted into the above equation, $\hat{\boldsymbol{\theta}}_{BCLS}$ is automatically obtained.

On the other hand, if the ratio of noise variances in unknown, (4.8) contains $2m+2$ unknowns but comprises of only $2m$ equations, one for each of the model parameters. Consequently, additional equations have to supplement the above set of equations. One relation can be obtained by using the minimum error V_{LS} of the least-squares estimate

$$V_{LS} = \min_{\boldsymbol{\theta}_{LS}} \mathbb{E}\varepsilon^2 = \min_{\boldsymbol{\theta}_{LS}} \mathbb{E}\left(y - \boldsymbol{\phi}^{\top}\boldsymbol{\theta}_{LS}\right)^2.$$

Expanding the expected value

$$\begin{aligned}\mathbb{E}\left(y-\boldsymbol{\phi}^\top\boldsymbol{\theta}_{LS}\right)^2 &= \mathbb{E}y^2+\mathbb{E}\left(\boldsymbol{\phi}^\top\boldsymbol{\theta}_{LS}\right)^2-2\mathbb{E}y\left(\boldsymbol{\phi}^\top\boldsymbol{\theta}_{LS}\right)\\ &= \mathbb{E}\left(\tilde{y}+\bar{y}\right)^2+\mathbb{E}\left(\boldsymbol{\phi}^\top\boldsymbol{\theta}_{LS}\right)^2\\ &= \mathbb{E}\tilde{y}^2+\mathbb{E}\bar{y}^2+2\mathbb{E}\tilde{y}\bar{y}+\mathbb{E}\left(\boldsymbol{\phi}^\top\boldsymbol{\theta}_{LS}\right)^2\\ &= \sigma_y^2+\mathbb{E}\bar{y}^2+\mathbb{E}\left(\boldsymbol{\phi}^\top\boldsymbol{\theta}_{LS}\right)^2.\end{aligned}$$

where we have used that

$$\begin{aligned}\mathbb{E}y\left(\boldsymbol{\phi}^\top\boldsymbol{\theta}_{LS}\right) &= \mathbb{E}y\,\mathbb{E}\left(\boldsymbol{\phi}^\top\boldsymbol{\theta}_{LS}\right)=0\\ \mathbb{E}\tilde{y}\bar{y} &= \mathbb{E}\tilde{y}\,\mathbb{E}\bar{y}=0\end{aligned}$$

since the current (zero-mean) noise contamination $\tilde{y}$ with $\mathbb{E}\tilde{y}=0$ in y is not correlated with the past (zero-mean) noise contamination in $\boldsymbol{\phi}^\top\boldsymbol{\theta}_{LS}$.

From (4.7), it follows that

$$\boldsymbol{\theta}^\top\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\hat{\boldsymbol{\theta}}_{LS}=\boldsymbol{\theta}^\top\left(\boldsymbol{\Phi}^\top\boldsymbol{\Phi}-\tilde{\boldsymbol{\Phi}}^\top\tilde{\boldsymbol{\Phi}}\right)\boldsymbol{\theta}$$

and

$$\hat{\boldsymbol{\theta}}_{LS}^\top\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\hat{\boldsymbol{\theta}}_{LS}=\hat{\boldsymbol{\theta}}_{LS}^\top\left(\boldsymbol{\Phi}^\top\boldsymbol{\Phi}-\tilde{\boldsymbol{\Phi}}^\top\tilde{\boldsymbol{\Phi}}\right)\boldsymbol{\theta}.$$

Combining these results,

$$\begin{aligned}\hat{\boldsymbol{\theta}}_{LS}^\top\left(\boldsymbol{\Phi}^\top\boldsymbol{\Phi}-\tilde{\boldsymbol{\Phi}}^\top\tilde{\boldsymbol{\Phi}}\right)\boldsymbol{\theta} &= \hat{\boldsymbol{\theta}}_{LS}^\top\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\hat{\boldsymbol{\theta}}_{LS}\\ \hat{\boldsymbol{\theta}}_{LS}^\top\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\boldsymbol{\theta}-\hat{\boldsymbol{\theta}}_{LS}^\top\tilde{\boldsymbol{\Phi}}^\top\tilde{\boldsymbol{\Phi}}\boldsymbol{\theta} &= \hat{\boldsymbol{\theta}}_{LS}^\top\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\hat{\boldsymbol{\theta}}_{LS}\\ \hat{\boldsymbol{\theta}}_{LS}^\top\tilde{\boldsymbol{\Phi}}^\top\tilde{\boldsymbol{\Phi}}\boldsymbol{\theta} &= \hat{\boldsymbol{\theta}}_{LS}^\top\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\boldsymbol{\theta}-\hat{\boldsymbol{\theta}}_{LS}^\top\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\hat{\boldsymbol{\theta}}_{LS}\\ &= \boldsymbol{\theta}^\top\boldsymbol{\Phi}_0^\top\boldsymbol{\Phi}_0\boldsymbol{\theta}-\hat{\boldsymbol{\theta}}_{LS}^\top\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\hat{\boldsymbol{\theta}}_{LS}\\ &= \bar{\mathbf{y}}^\top\bar{\mathbf{y}}-\hat{\boldsymbol{\theta}}_{LS}^\top\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\hat{\boldsymbol{\theta}}_{LS}\end{aligned}$$

Thus,

$$V_{LS}=\min_{\boldsymbol{\theta}}\mathbb{E}\varepsilon^2=\min_{\boldsymbol{\theta}}\mathbb{E}\left(y_k-\boldsymbol{\phi}_{m,k}^\top\boldsymbol{\theta}\right)^2=\sigma_y^2+\hat{\boldsymbol{\theta}}_{LS}^\top\mathrm{cov}\left(\tilde{\boldsymbol{\phi}}_{m,k}\right)\hat{\boldsymbol{\theta}}_{LS} \tag{4.9}$$

In a practical scenario, the expected value is not known but is computed using the available samples as well as the current estimates for $\boldsymbol{\theta}$. This suggests that unlike least-squares estimation, the compensated least-squares procedure is iterative.

In order to get a second extra equation, an extended model structure should be considered. A possible extension is appending an additional $-y_{k-m_a-1}$ to the regressor vector $\boldsymbol{\phi}$ and a corresponding a_{m_a+1} parameter to $\boldsymbol{\theta}$ (whose true value is 0) and using the extended versions in the formulas of the original model in (4.8) ($m=m_a+m_b$):

$$\begin{aligned}\boldsymbol{\theta} &\leftarrow \begin{bmatrix}-a_1 & \dots & -a_{m_a} & -a_{m_a+1} & b_1 & \dots & b_{m_b}\end{bmatrix}\\ \boldsymbol{\phi}_k &\leftarrow \begin{bmatrix}\boldsymbol{\phi}_{y,k}^\top & \boldsymbol{\phi}_{u,k}^\top\end{bmatrix}^\top\\ \boldsymbol{\phi}_{y,k} &\leftarrow \begin{bmatrix}y_{k-1} & \dots & y_{k-m_a} & y_{k-m_a-1}\end{bmatrix}^\top\\ \boldsymbol{\phi}_{u,k} &\leftarrow \begin{bmatrix}u_{k-1} & \dots & u_{k-m_b}\end{bmatrix}^\top\end{aligned}$$

These additional equations allow us to infer estimates for σ_u^2 and σ_y^2. Once these have been estimated, the bias of the least-squares estimate is eliminated to achieve consistent estimates.

The iterative bias-compensating estimation algorithm is therefore as follows [74]:

1. Set the initial value of $\hat{\boldsymbol{\theta}}_{(0)}$ to $\hat{\boldsymbol{\theta}}_{LS}$ according to (4.6).
2. Solve (4.9) and the equation(s) corresponding to the extended model using the current parameter estimates $\hat{\boldsymbol{\theta}}_{(k)}$ to get estimates for the noise elements $\hat{v}_{(k+1)} = [\ \hat{\sigma}_y^2 \quad \hat{\sigma}_u^2\ ]$.
3. Using (4.8), compute new parameter estimates $\hat{\boldsymbol{\theta}}_{(k+1)}$ using $\hat{\boldsymbol{\theta}}_{(k)}$ and $\hat{v}_{(k+1)}$, and repeat from step (2).

Provided that the iteration converges, the termination criterion is

$$\frac{||\hat{\boldsymbol{\theta}}_{(k+1)} - \hat{\boldsymbol{\theta}}_{(k)}||}{||\hat{\boldsymbol{\theta}}_{(k)}||} < \varepsilon$$

where ε is some small value.

In practice, the estimates obtained with the bias-compensated least-squares method are often rather crude, which can be significantly improved by augmenting multiple input or output parameters, giving rise to a family of estimators, called extended bias-compensated least-squares estimators [24, 33, 62]. As the number of equations in this case exceeds the number of unknowns, an overdetermined system of equations has to be solved in a least-squares sense. This means that instead of computing

$$\hat{\boldsymbol{\theta}}_{BCLS} = \left(\boldsymbol{\Phi}^\top\boldsymbol{\Phi} - \tilde{\boldsymbol{\Phi}}^\top\tilde{\boldsymbol{\Phi}}\right)^{-1}\boldsymbol{\Phi}^\top\mathbf{y} = \left(\boldsymbol{\Phi}^\top\boldsymbol{\Phi} - \tilde{\boldsymbol{\Phi}}^\top\tilde{\boldsymbol{\Phi}}\right)^{-1}\left(\boldsymbol{\Phi}^\top\mathbf{y} - \tilde{\boldsymbol{\Phi}}^\top\tilde{\mathbf{y}}\right)$$

where $\tilde{\boldsymbol{\Phi}}^\top\tilde{\mathbf{y}} \approx 0$ by independence of past and present noise contamination (white noise), we can compute

$$\hat{\boldsymbol{\theta}}_{EBCLS} = \left(\mathbf{Z}^\top\boldsymbol{\Phi} - \tilde{\mathbf{Z}}^\top\tilde{\boldsymbol{\Phi}}\right)^\dagger\left(\mathbf{Z}^\top\mathbf{y} - \tilde{\mathbf{Z}}^\top\tilde{\mathbf{y}}\right) \tag{4.10}$$

where $\mathbf{Z}$ contains so-called *instruments*. $\mathbf{Z}$ is often constructed in such a manner that it contains the same set of delayed observations as found in $\boldsymbol{\Phi}$ as well as some additional observations with more delay than the order m of the system. For instance, let

$$\boldsymbol{\phi}_{m,k}^\top = [\ y_{k-1} \quad \cdots \quad y_{k-m} \quad u_{k-1} \quad \cdots \quad u_{k-m}\ ]$$

then the Yule-Walker instrument vector $\mathbf{z}_{m,k}$ could be built of the shifted observations

$$\mathbf{z}_{m,k}^\top = [\ y_{k-1} \quad \cdots \quad y_{k-m} \quad u_{k-1} \quad \cdots \quad u_{k-m} \quad u_{k-m-1} \quad u_{k-m-2}\ ].$$

As with the standard bias-compensated scheme, (4.10) has $2m+2$ unknowns. Unlike the standard scheme, however, it has as many equations as elements in the instrument vector. Should two additional observations be added to the vector, it becomes possible to solve the system of equations, and appending more observations can improve the conditioning of the problem.

4.1.5 The Frisch scheme

The Frisch scheme, another approach to estimating both model and noise parameters, provides a recursive algorithm strikingly similar to the BCLS approach so that many of its variants may be interpreted as a special form of BCLS, operating on similar extended models [9] with comparable performance results [34]. It is based on the idea that the sample covariance matrix $\bar{\mathbf{D}}$ of the true values of observations $\bar{\mathbf{x}}$ yields the zero vector when multiplied by the true parameter values $\mathbf{g}$, i.e.

$$\bar{\mathbf{D}}\mathbf{g} = \frac{1}{N-m}\bar{\mathbf{X}}^\top\bar{\mathbf{X}}\mathbf{g} = \frac{1}{N-m}\bar{\mathbf{X}}^\top\mathbf{0} = \mathbf{0}$$

This implies that in terms of the measured quantities, it holds that

$$\bar{\mathbf{D}}\mathbf{g} = (\mathbf{D}-\tilde{\mathbf{D}})\mathbf{g} = \mathbf{0} \tag{4.11}$$

where we have used that $\mathbf{D} = \bar{\mathbf{D}} + \tilde{\mathbf{D}}$ where $\tilde{\mathbf{D}} = \tilde{\mathbf{D}}(\sigma_y^2, \sigma_u^2)$ is a(n estimated) covariance matrix corresponding to white noise on both output and input. Similarly to the BCLS case, we have more unknowns than equations in (4.11), namely, m linearly dependent unknowns in $\mathbf{g}$ and the two unknowns σ_y^2 and σ_u^2. However, assuming an estimate of σ_u^2 is available, σ_y^2 may be computed such that the difference matrix $\mathbf{D}-\tilde{\mathbf{D}}$ is singular. First, we have

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_{yy} & \mathbf{D}_{yu} \\ \mathbf{D}_{uy} & \mathbf{D}_{uu} \end{bmatrix}$$

in which $\mathbf{D}_{yy}$, $\mathbf{D}_{yu}$, $\mathbf{D}_{uy}$ and $\mathbf{D}_{uu}$ denote the sample covariance matrices belonging to output- and input-related entries in $\mathbf{x}$ where $\mathbf{D}_{uy} = \mathbf{D}_{yu}^\top$ due to symmetry. By construction, we have $\bar{\mathbf{D}}_{yu} = \mathbf{D}_{yu}$ since the input and output noises are assumed to be independent, and $\bar{\mathbf{D}}_{yy} = \mathbf{D}_{yy} - \tilde{\mathbf{D}}_{yy} = \mathbf{D}_{yy} - \sigma_y^2\mathbf{I}$ and $\bar{\mathbf{D}}_{uu} = \mathbf{D}_{uu} - \tilde{\mathbf{D}}_{uu} = \mathbf{D}_{uu} - \sigma_u^2\mathbf{I}$:

$$\bar{\mathbf{D}} = \begin{bmatrix} \mathbf{D}_{yy} - \sigma_y^2\mathbf{I} & \mathbf{D}_{yu} \\ \mathbf{D}_{uy} & \mathbf{D}_{uu} - \sigma_u^2\mathbf{I} \end{bmatrix}$$

What we seek is $\det\bar{\mathbf{D}} = 0$. Expanding the determinant we have

$$\begin{aligned} \det\bar{\mathbf{D}} &= \det\left(\begin{bmatrix} \mathbf{D}_{yy} - \sigma_y^2\mathbf{I} & \mathbf{D}_{yu} \\ \mathbf{D}_{uy} & \mathbf{D}_{uu} - \sigma_u^2\mathbf{I} \end{bmatrix}\right) \\ &= \det\left(\mathbf{D}_{uu} - \sigma_u^2\mathbf{I}\right)\det\left(\mathbf{D}_{yy} - \sigma_y^2\mathbf{I} - \mathbf{D}_{yu}\left(\mathbf{D}_{uu} - \sigma_u^2\mathbf{I}\right)^{-1}\mathbf{D}_{uy}\right) \end{aligned}$$

If the input signal provides sufficient excitation, $\det\left(\mathbf{D}_{uu} - \sigma_u^2\mathbf{I}\right) > 0$, which implies

$$\det\left(\mathbf{D}_{yy} - \mathbf{D}_{yu}\left(\mathbf{D}_{uu} - \sigma_u^2\mathbf{I}\right)^{-1}\mathbf{D}_{uy} - \sigma_y^2\mathbf{I}\right) = 0 \tag{4.12}$$

Since the only unknown in the above equation is the scalar σ_y^2, (4.12) is the characteristic equation of the matrix

$$\mathbf{D}_{yy} - \mathbf{D}_{yu}\left(\mathbf{D}_{uu} - \sigma_u^2\mathbf{I}\right)^{-1}\mathbf{D}_{uy}.$$

As we are interested in the smallest possible noise contribution, it follows that

$$\sigma_y^2 = \lambda_{\min}\left(\mathbf{D}_{yy} - \mathbf{D}_{yu}(\mathbf{D}_{uu} - \sigma_u^2\mathbf{I})^{-1}\mathbf{D}_{uy}\right) \tag{4.13}$$

where operator $\lambda_{\min}(\mathbf{M})$ denotes the minimum eigenvalue of the operand matrix $\mathbf{M}$.

In order to determine σ_u^2, one of the more robust approaches is to compute so-called residuals and compare their statistical properties to what can be predicted from the model [21]. A residual is defined as

$$\varepsilon(t) = A(q^{-1})y_k - B(q^{-1})u_k.$$

Introduce the covariance vector $\mathbf{r}$ belonging to shift $k \geqq 1$ with components

$$r_i = \mathbb{E}(w_k w_{k+i})$$

and its estimate from finite samples as

$$\hat{r}_i = \frac{1}{N-i}\sum_{k=1}^{N-i} w_k w_{k+i}$$

The idea is to compute the sample covariance vector $\mathbf{r}$ using ε_k where

$$\varepsilon_k(\hat{\boldsymbol{\theta}}) = \hat{A}(q^{-1})y_k - \hat{B}(q^{-1})u_k$$

in which $\hat{A}(q^{-1})$ and $\hat{B}(q^{-1})$ encapsulate current model parameter estimates, and compare it to a theoretical covariance vector based on

$$\varepsilon_k\left(\hat{\boldsymbol{\theta}}, \hat{\sigma}_y^2, \hat{\sigma}_u^2\right) = \hat{A}(q^{-1})\tilde{y}_k - \hat{B}(q^{-1})\tilde{u}_k$$

where $\tilde{y}_k$ as well as $\tilde{u}_k$ are independent white noise sequences with variance as determined by the current estimates $\hat{\sigma}_y^2$ and $\hat{\sigma}_u^2$. The dimension of the covariance vector $\mathbf{r}$ (i.e. the maximum shift i) is a user-supplied parameter. Essentially, we are minimizing

$$\boldsymbol{\delta} = \mathbf{r}\left(\varepsilon_k(\hat{\boldsymbol{\theta}})\right) - \mathbf{r}\left(\varepsilon_k\left(\hat{\boldsymbol{\theta}}, \hat{\sigma}_y^2, \hat{\sigma}_u^2\right)\right)$$

with some weights attached to the entries in $\mathbf{r}$ to reflect their relative importance. A possible choice is

$$\mathbf{W} = \begin{bmatrix} 2n & & & \\ & 2(n-1) & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}$$

where n is the size of $\mathbf{r}$ and the other entries in $\mathbf{W}$ are zero.

The entire algorithm runs as follows:

1. Assume an initial value for $\hat{\sigma}_u^2$.
2. Compute an estimate $\hat{\sigma}_y^2$ using (4.13).
3. Compute model parameters based on (4.11).
4. Determine the residuals $\varepsilon_k(\hat{\boldsymbol{\theta}})$ using estimated model parameters in $\hat{A}$ and $\hat{B}$ as well as observed output and input sequences $y(t)$ and $u(t)$, and compute the related sample covariance vector $\hat{r}$.

5. Determine the theoretical reference covariance vector using residuals $\varepsilon_k\left(\hat{\boldsymbol{\theta}}, \hat{\sigma}_y^2, \hat{\sigma}_u^2\right)$ generated by estimated process parameters and white noise sequences $\tilde{y}_k$ and $\tilde{u}_k$ where

$$\begin{aligned} \mathbb{E}\left(\tilde{y}_k\right)^2 &= \hat{\sigma}_y^2 \\ \mathbb{E}\left(\tilde{u}_k\right)^2 &= \hat{\sigma}_u^2 \end{aligned}$$

   Compare the sample and the reference covariance vectors by setting $V = \boldsymbol{\delta}^\top \mathbf{W} \boldsymbol{\delta}$ where $\boldsymbol{\delta}$ is a difference vector of covariances and $\mathbf{W}$ is a weighing matrix.

6. Repeat from step 1 minimizing V.

4.2 Nonlinear systems

Having seen linear errors-in-variables estimators that simultaneously produce model and noise parameter estimates, a natural extension of these approaches is their application to the nonlinear setting. First, we shall investigate the polynomial bias-compensated least-squares method (Section 4.2.1), which is a generalization of the bias-compensated principle to nonlinear systems that are captured by polynomial functions. Unfortunately, bias-compensated least squares, and especially its polynomial generalization, yields rather crude estimates. A more promising approach is the introduction of nonlinearity in the generalized Koopmans–Levin estimator (GKL), which has been shown to blend the advantages of the high-accuracy maximum likelihood, and the fast and robust non-iterative Koopmans–Levin estimators. Since the GKL method assumes a known relative distribution of noise between input and output, an additional step is required in its nonlinear generalization, the *polynomial generalized Koopmans–Levin* (PGKL) method (Section 4.2.2), so that it may produce a noise ratio estimate.

4.2.1 Polynomial bias-compensated least-squares method

A natural choice for constructing an estimator that targets polynomial systems is to extend the bias-compensated least-squares principle for linear systems to polynomial systems. As shown in Section 4.1.4, a bias-compensated least-squares estimate takes the form

$$\hat{\boldsymbol{\theta}}_{BCLS} = \left(\boldsymbol{\Phi}^\top \boldsymbol{\Phi} - \tilde{\boldsymbol{\Phi}}^\top \tilde{\boldsymbol{\Phi}}\right)^{-1} \boldsymbol{\Phi}^\top \mathbf{y} \tag{4.14}$$

where the matrix $\boldsymbol{\Phi}$ collects time-shifted output and input observations such that

$$\boldsymbol{\Phi} = \begin{bmatrix} y_m & \cdots & y_1 & u_m & \cdots & u_1 \\ y_{m+1} & \cdots & y_2 & u_{m+1} & \cdots & u_2 \\ \vdots & & \vdots & \vdots & & \vdots \\ y_N & \cdots & y_{N-m} & u_N & \cdots & u_{N-m} \end{bmatrix}$$

and $\tilde{\boldsymbol{\Phi}}$ is the noise contribution.

Investigating the structure of (4.14), it follows that once $\tilde{\boldsymbol{\Phi}}^\top \tilde{\boldsymbol{\Phi}}$ is expressed as a function of noise parameters σ_y^2 and σ_u^2, the bias-compensated least-squares estimate is readily available. For the linear case,

$$\mathbb{E}\boldsymbol{\phi}\boldsymbol{\phi}^\top \approx \frac{1}{N}\tilde{\boldsymbol{\Phi}}^\top \tilde{\boldsymbol{\Phi}}$$

was a simple function of σ_y^2 and σ_u^2 such that

$$\mathbb{E}\tilde{\boldsymbol{\phi}}\tilde{\boldsymbol{\phi}}^\top = \operatorname{diag}\left[\begin{array}{cccccc} \sigma_y^2 & \ldots & \sigma_y^2 & \sigma_u^2 & \ldots & \sigma_u^2 \end{array}\right].$$

For the polynomial case, the situation is more complex but still tractable. With a set of identities and approximations, and exploiting the independence of time-shifted observations, one is able to get an estimate of $\mathbb{E}\,\mathbf{f}_d(\tilde{\boldsymbol{\phi}})\mathbf{f}_d^\top(\tilde{\boldsymbol{\phi}})$:

$$\begin{aligned}
\mathbb{E}\left(x_k^p\right) &= \mathbb{E}\left(x_{0,k}+n_k\right)^p \\
\mathbb{E}\left(n_k^{2p}\right) &= (2p-1)(2p-3)\ldots 1\,\sigma^{2p} \\
\mathbb{E}\left(n_k^{2p-1}\right) &= 0 \\
\mathbb{E}\left(u_k y_k\right) &= \mathbb{E}\left(u_k\right)\mathbb{E}\left(y_k\right) \\
\mathbb{E}\left(u_k u_{k-p}\right) &= \mathbb{E}\left(u_k\right)\mathbb{E}\left(u_{k-p}\right) \quad \text{where } p>0 \\
\mathbb{E}\left(x_i\right) &\approx \bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i
\end{aligned}$$

The strategy is strikingly similar to that used in the case of static polynomial systems in Chapter 2.1. Noisy observations are expressed in terms of the noise-free data and a noise contribution where the noise contribution may involve dependence on lower-degree unobservable noise-free data, which are recursively expressed using the same approach.

Once an estimate of $\mathbb{E}\,\mathbf{f}_d(\tilde{\boldsymbol{\phi}})\mathbf{f}_d^\top(\tilde{\boldsymbol{\phi}})$ is available, one is able to construct the polynomial bias-compensated estimate as

$$\hat{\boldsymbol{\theta}}_{PBCLS} = \left(\mathbb{E}\,\mathbf{f}_d(\boldsymbol{\phi})\mathbf{f}_d^\top(\boldsymbol{\phi}) - \mathbb{E}\,\mathbf{f}_d(\tilde{\boldsymbol{\phi}})\mathbf{f}_d^\top(\tilde{\boldsymbol{\phi}})\right)^{-1}\mathbb{E}\,\mathbf{f}_d(\boldsymbol{\phi})\,y.$$

As with the linear case, one can augment the vector $\boldsymbol{\phi}$ with additional observations and get an instrument vector $\boldsymbol{\omega}$, ensuring a solution or improving the conditioning of the problem. Even while the expectation $\mathbb{E}\,\mathbf{f}_d(\tilde{\boldsymbol{\omega}})\mathbf{f}_d^\top(\tilde{\boldsymbol{\phi}})$ that thus results has entries not present in the original $\mathbb{E}\,\mathbf{f}_d(\tilde{\boldsymbol{\phi}})\mathbf{f}_d^\top(\tilde{\boldsymbol{\phi}})$, one can rely on the very same approach of decomposition into noise-free part and noise contribution to estimate their value. The polynomial extended bias-compensated least squares is thus readily obtained as

$$\hat{\boldsymbol{\theta}}_{PEBCLS} = \left(\mathbb{E}\,\mathbf{f}_d(\boldsymbol{\omega})\mathbf{f}_d^\top(\boldsymbol{\phi}) - \mathbb{E}\,\mathbf{f}_d(\tilde{\boldsymbol{\omega}})\mathbf{f}_d^\top(\tilde{\boldsymbol{\phi}})\right)^\dagger\left(\mathbb{E}\,\mathbf{f}_d(\boldsymbol{\omega})\,y - \mathbb{E}\,\mathbf{f}_d(\tilde{\boldsymbol{\omega}})\,y\right)$$

4.2.2 Polynomial generalized Koopmans–Levin method

The polynomial generalized Koopmans–Levin (PGKL) method applies the principles of the (linear) generalized Koopmans–Levin estimator (GKL) to the nonlinear setting. As GKL inherits some of the accuracy of the maximum likelihood estimator it is based on, we may expect

PGKL to exhibit similar favorable estimation accuracy. In fact, we shall see that the algorithm produces better estimates than other methods, in particular PBCLS.

The scheme comprises of two independent steps:

1. estimating model parameters $\boldsymbol{\theta}$ and noise magnitude μ with given relative distribution of noise φ over input and output noise ratio (known input/output noise ratio), and

2. estimating the relative distribution of noise φ.

Estimation of model parameters and noise magnitude can be broken down into estimating $\boldsymbol{\theta} = \mathbf{f}_{par}(\mathbf{g})$ given $\mathbf{f}_{data}^{\top}(\bar{\mathbf{x}}_{m,k})$, and approximating $\boldsymbol{\theta}$ such that $\mathbf{f}_{par}(\mathbf{g}) \approx \boldsymbol{\theta}$, possibly alternating between the two objectives with an iterative algorithm. For estimating $\boldsymbol{\theta}$, the vector $\mathbf{f}_{data}^{\top}(\bar{\mathbf{x}}_{m,k})$ is assumed to consist of blocks with elements in decreasing order of time of observation, e.g. one block of $\mathbf{f}_{data}^{\top}(\bar{\mathbf{x}}_{m,k})$ could be

$$\begin{bmatrix} u_k^2 y_k & \dots & u_{k-m+1}^2 y_{k-m+1} \end{bmatrix}^{\top}.$$

Let

$$\begin{aligned}
\mathbf{C}_q &= \mu \mathbf{C} \otimes \mathbf{I}_q \\
\boldsymbol{\theta}_p &= \begin{bmatrix} p_0 & \dots & p_m \end{bmatrix}^{\top} \\
\mathbf{G}_q &= \begin{bmatrix} \mathbf{G}_q^{\theta_1} \\ \mathbf{G}_q^{\theta_2} \\ \vdots \end{bmatrix} \\
\mathbf{G}_q^{\theta_p} &= \begin{bmatrix}
p_0 & 0 & 0 & \dots & 0 & 0 \\
p_1 & p_0 & 0 & \dots & 0 & 0 \\
p_2 & p_1 & p_0 & \dots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
p_m & p_{m-1} & p_{m-2} & \dots & 0 & 0 \\
0 & p_m & p_{m-1} & \dots & 0 & 0 \\
0 & 0 & p_m & \dots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \dots & p_m & p_{m-1} \\
0 & 0 & 0 & \dots & 0 & p_m
\end{bmatrix}_{q,q-m}
\end{aligned}$$

where p_n are the parameters corresponding to data elements in a block in decreasing order of time.

Let $\mathbf{C}_q(\mu) = \mathbb{E}\left(\mathbf{z}_m \mathbf{z}_m^{\top}\right) - \mathbb{E}\left(\bar{\mathbf{z}}_m \bar{\mathbf{z}}_m^{\top}\right)$ be a precomputed polynomial in terms of the free scalar μ whose coefficients are matrices that represent the noise covariance structure. As $\mathbb{E}\left(\bar{\mathbf{z}}_m \bar{\mathbf{z}}_m^{\top}\right)$ is not at our disposal, $\mathbf{C}_q(\mu)$ is approximated from observations, i.e. $\mathbf{C}_q(\mu) \approx \mathbf{C}_q(\mu, \mathbf{X})$. The approach is similar to that discussed in Section 2.1.6, with an additional expectation independence taken into account such that

$$\mathbb{E}(x_k x_{k-\tau}) = \mathbb{E}(x_k)\,\mathbb{E}(x_{k-\tau}).$$

Estimating parameters can then be accomplished either with minimizing a cost function, or with an iterative algorithm, similarly to that seen with GKL, quickly converging to the optimum value of that cost function. The PGKL objective function takes the form

$$J_{PGKL} = \text{trace}\left(\mathbf{G}_q^\top \mathbf{C}_q \mathbf{G}_q\right)^{-1}\left(\mathbf{G}_q^\top \mathbf{D}_q \mathbf{G}_q\right)$$

where $\mathbf{G}_q = \mathbf{G}_q(\boldsymbol{\theta})$ and $\mathbf{C}_q = \mathbf{C}_q(\mu)$. One approach to minimizing the objective function is with the Levenberg–Marquardt algorithm. However, a simple iterative scheme, as for GKL, may be formulated, which has similarly favorable convergence properties.

Suppose an initial value for $\boldsymbol{\theta}$ and μ is available, let these be $\boldsymbol{\theta}_{(1)}$ and $\mu_{(1)}$, respectively. Then

$$\boldsymbol{\theta}_{(k+1)},\, \mu_{(k+1)} = \arg\min_{\boldsymbol{\theta},\mu} \frac{\text{trace}\left(\mathbf{G}_{q,(k)}^\top \mathbf{C}_{q,(k)} \mathbf{G}_{q,(k)}\right)^{-1}\left(\mathbf{G}_q^\top \mathbf{D}_q \mathbf{G}_q\right)}{\text{trace}\left(\mathbf{G}_{q,(k)}^\top \mathbf{C}_{q,(k)} \mathbf{G}_{q,(k)}\right)^{-1}\left(\mathbf{G}_q^\top \mathbf{C}_q \mathbf{G}_q\right)}$$

where $\mathbf{G}_q = \mathbf{G}_q(\boldsymbol{\theta})$, $\mathbf{G}_{q,(k)} = \mathbf{G}_q(\boldsymbol{\theta}_{(k)})$, $\mathbf{C}_q = \mathbf{C}_q(\mu)$ and $\mathbf{C}_{q,(k)} = \mathbf{C}_q(\mu_{(k)})$ is a means of computing the successor estimates $\boldsymbol{\theta}_{(k+1)}$ and $\mu_{(k+1)}$ given $\boldsymbol{\theta}_{(k)}$ and $\mu_{(k)}$. The resulting iterative scheme

$$\arg\min_{\boldsymbol{\theta},\mu} \frac{\boldsymbol{\theta}^\top \mathbf{T}^\top \left(\left(\mathbf{G}_{q,(k)}^\top \mathbf{C}_{q,(k)} \mathbf{G}_{q,(k)}\right)^{-1} \otimes \mathbf{D}_q\right) \mathbf{T}\boldsymbol{\theta}}{\boldsymbol{\theta}^\top \mathbf{T}^\top \left(\left(\mathbf{G}_{q,(k)}^\top \mathbf{C}_{q,(k)} \mathbf{G}_{q,(k)}\right)^{-1} \otimes \mathbf{C}_q(\mu)\right) \mathbf{T}\boldsymbol{\theta}}$$

where $\mathbf{T}$ is a sparse matrix of zeros and ones chosen such that $\text{vec}(\mathbf{G}_q) = \mathbf{T}\boldsymbol{\theta}$ is a polynomial eigenvector problem with matrix coefficients

$$\begin{aligned} \boldsymbol{\Psi}(\mu) &= \mathbf{T}^\top \left(\mathbf{Q} - \mu\mathbf{R}_1 - \mu^2\mathbf{R}_2 - \ldots - \mu^p \mathbf{R}_p\right)\mathbf{T} \\ \mathbf{Q} &= \left(\mathbf{G}_{q,(k)}^\top \mathbf{C}_{q,(k)} \mathbf{G}_{q,(k)}\right)^{-1} \otimes \mathbf{D}_q \\ \mathbf{R}_j &= \left(\mathbf{G}_{q,(k)}^\top \mathbf{C}_{q,(k)} \mathbf{G}_{q,(k)}\right)^{-1} \otimes \mathbf{C}_q^{(j)}. \end{aligned} \tag{4.15}$$

In order to obtain a solution to $\boldsymbol{\Psi}(\mu)$, we need to solve the following optimization problem:

$$\begin{aligned} \min \quad & \mu \\ \text{s.t.} \quad & \boldsymbol{\Psi}(\mu) \succeqq \mathbf{0} \\ & \mu \geqq 0. \end{aligned}$$

The positive semi-definite constraint maintains the property that data originate from a covariance matrix, which is by definition always positive semi-definite, whereas the non-negativity constraint eliminates those cases where the noise magnitude would be negative, which is not feasible.

One way to simplify the problem is to apply linearization, thereby eliminating the polynomial dependence in μ at the expense of increasing the size of coefficient matrices. A well-known result [63] for linearizing the quadratic eigenvalue problem (QEP)

$$\mu^2\mathbf{M} + \mu\mathbf{C} + \mathbf{K}$$

is

$$\mathbf{A} - \mu\mathbf{B}$$

with

$$\begin{aligned}\mathbf{A} &= \begin{bmatrix} \mathbf{0} & \mathbf{W} \\ -\mathbf{K} & -\mathbf{C} \end{bmatrix} \\ \mathbf{B} &= \begin{bmatrix} \mathbf{W} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix}\end{aligned}$$

where the choice $\mathbf{W} = -\mathbf{K}$ yields a generalized eigenvalue problem with symmetric matrices where the eigenvector has a special structure

$$\mathbf{w} = \begin{bmatrix} \mathbf{v} \\ \mu\mathbf{v} \end{bmatrix}.$$

Unfortunately, this simple approach does not generalize to matrix polynomial eigenvalue problems (PEP) of higher order. However, a more elaborate and symmetry-preserving linearization approach [6] transforms the above problem to

$$\Xi(\mu) = \Xi_1 - \mu\Xi_2$$

which expands for even p as

$$\begin{aligned}\Xi_1 &= \operatorname{diag}\left\{\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \mathbf{R}_1 \end{bmatrix}, \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \mathbf{R}_3 \end{bmatrix}, \ldots, \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \mathbf{R}_{p-1} \end{bmatrix}\right\} \\ \Xi_2 &= \operatorname{diag}\left\{\mathbf{Q}^{-1}, \begin{bmatrix} -\mathbf{R}_2 & \mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{bmatrix}, \ldots, \begin{bmatrix} -\mathbf{R}_{p-2} & \mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{bmatrix}, -\mathbf{R}_p\right\}\end{aligned}$$

and for odd p as

$$\begin{aligned}\Xi_1 &= \operatorname{diag}\left\{\mathbf{Q}, \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \mathbf{R}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \mathbf{R}_4 \end{bmatrix}, \ldots\right\} \\ \Xi_2 &= \operatorname{diag}\left\{\begin{bmatrix} -\mathbf{R}_1 & \mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{bmatrix}, \begin{bmatrix} -\mathbf{R}_3 & \mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{bmatrix}, \ldots, -\mathbf{R}_p\right\}\end{aligned}$$

where the operator diag aligns its arguments to bring forth a block diagonal matrix.

Ideally (provided that the matrix inequality constraint is sharp), this problem can be solved using generalized eigenvalue problem, which is the linearized equivalent of solving the original polynomial eigenvalue problem. As the linearized problem has eigenvectors $\mathbf{w}$ of dimension mp rather than m, the true polynomial eigenvector that belongs to the eigenvalue μ becomes the column $\mathbf{v}$ of $\mathrm{vec}\mathbf{V} = \mathbf{w}$ of the linearized eigenvector $\Xi_1\mathbf{w} = \mu\Xi_2\mathbf{w}$ that gives the smallest normalized residual, i.e.

$$\mathbf{v} = \arg\min_{\mathbf{v}} \frac{\sum_k \left|\left[\Psi(\mu)\mathbf{v}\right]_k\right|}{\sum_k |[\mathbf{v}]_k|}$$

where $[\mathbf{v}]_k$ is the kth component of the vector $\mathbf{v}$.

Once an estimate for $\boldsymbol{\theta}_{(n+1)}$ has been computed, we can get $\mathbf{g}_{(n+1)}$ by finding

$$\arg\min_{\mathbf{g}} \left|\mathbf{f}_{par}(\mathbf{g}) - \boldsymbol{\theta}_{(n+1)}\right|.$$

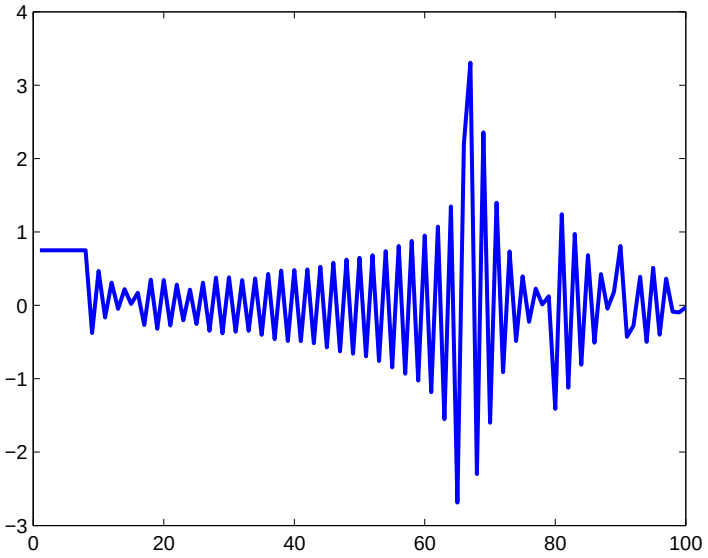


Figure 4.1: The first 100 samples of the investigated Mackey–Glass chaotic process.

Example 12. Consider a variant of the Mackey–Glass model for white blood cell production (Figure 4.1), which is captured by the difference equation

$$\bar{x}_{k+1} = -a\bar{x}_k + \frac{b\bar{x}_{k-\tau}}{1+\bar{x}_{k-\tau}^p}$$

that can be recast in an implicit, linearized form

$$\bar{x}_{k+1}\bar{x}_{k-\tau}^p + a\bar{x}_k\bar{x}_{k-\tau}^p + \bar{x}_{k+1} + a\bar{x}_k - b\bar{x}_{k-\tau} = 0 \tag{4.16}$$

where

$$\begin{aligned} \bar{x}_k &= a \quad \text{if } 0 \leqq k \leqq \tau \\ a &= 0.75 \\ b &= 0.35 \\ \tau &= 7 \\ p &= 3. \end{aligned}$$

Investigating (4.16), we can identify five different nonlinear components. Even though the components are nonlinear in terms of $\bar{x}_k$ and its time-shifted versions $\bar{x}_{k+1}$ and $\bar{x}_{k-\tau}$, they are linear in the parameters a and b. In addition, there are five components but only three parameters, whose sum of squares must add up to one to ensure independence of scaling. This can be easily tackled by inserting a structural constraint matrix $\mathbf{S}$ in (4.15) that enforces the equality of parameters of the respective components. Thus, a solution to

$$\boldsymbol{\theta}^\top \boldsymbol{\Psi}(\mu)\boldsymbol{\theta} = \boldsymbol{\theta}^\top \mathbf{S}^\top \mathbf{T}^\top \left(\mathbf{Q} - \mu\mathbf{R}_1 - \mu^2\mathbf{R}_2 - \ldots - \mu^p\mathbf{R}_p\right)\mathbf{T}\mathbf{S}\boldsymbol{\theta}$$

$N = 5000$

	True	PBCLS [43]			PGKL		
y_{k-1}	1.5	1.500	±	0.019	1.4998	±	0.0039
y_{k-2}	-0.7	-0.700	±	0.018	-0.6998	±	0.0035
u_{k-1}	1.0	1.001	±	0.023	0.9995	±	0.0083
u_{k-1}^2	-0.3	-0.286	±	0.126	-0.3056	±	0.0444
$y_{k-1}y_{k-2}$	-0.05	-0.050	±	0.003	-0.0501	±	0.0015
$u_{k-1}y_{k-1}$	0.1	0.094	±	0.037	0.1010	±	0.0129

$N = 50000$

	True	PBCLS [43]			PGKL		
y_{k-1}	1.5	1.500	±	0.006	1.4999	±	0.0012
y_{k-2}	-0.7	-0.700	±	0.005	-0.7000	±	0.0011
u_{k-1}	1.0	1.000	±	0.008	1.0002	±	0.0029
u_{k-1}^2	-0.3	-0.306	±	0.042	-0.3041	±	0.0136
$y_{k-1}y_{k-2}$	-0.05	-0.050	±	0.001	-0.0501	±	0.0005
$u_{k-1}y_{k-1}$	0.1	0.100	±	0.012	0.1012	±	0.0048

Table 4.1: Comparison of the polynomial bias-compensated least-squares (PBCLS) and the polynomial generalized Koopmans–Levin (PGKL) methods.

where

$$\mathbf{S} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

yields the solution (normalized to the third component)

$$\boldsymbol{\theta}^\top = \begin{bmatrix} a & b & 1 \end{bmatrix}.$$

♣

Example 13. Consider the dynamic system cast in a linear form given by the following difference equation [43]:

$$\bar{y}_k = 1.5\bar{y}_{k-1} - 0.7g_2\bar{y}_{k-2} + \bar{u}_{k-1} - 0.3\bar{u}_{k-1}^2 - 0.05g_5\bar{y}_{k-1}\bar{y}_{k-2} + 0.1\bar{u}_{k-1}\bar{y}_{k-1}$$

Given noise magnitudes $\sigma_u^2 = 0.001$ and $\sigma_y^2 = 0.01$, and an input signal $\bar{u}$ generated by

$$\bar{u}_k - 0.5\bar{u}_k = e_k + 0.7e_{k-1}$$

where e_k is a uniformly distributed, white, zero-mean bounded sequence with $|e_k| < 0.112$ resulting in $|u_{0,k}| < 0.370$, we get signal-to-noise ratios of $\mathrm{SNR}_u \approx 11.2$dB and $\mathrm{SNR}_y \approx 15.3$dB on input and output, respectively. Table 4.1 compares the performance of the polynomial generalized Koopmans–Levin (PGKL) method with a model order of $q = 4$ to that of the polynomial bias-compensated least-squares (PBCLS) method in 100 runs of a Monte-Carlo simulation with sample sizes of 5000 and 50000. The sample sizes have been chosen to match those used in [43], the PGKL method could work reliably with smaller sample sizes.

As seen in the table, PBCLS and PGKL give comparable results for linear terms y_{k-1}, y_{k-2} and u_{k-1} as well as the term $u_{k-1}y_{k-1}$ even though the variance of estimates given by PGKL is smaller. However, for the quadratic term u_{k-1}^2 PBCLS fails to give an accurate result since the variance associated with its estimate is unacceptably high, while PGKL works reliably. ♣

Given the estimates $\hat{\boldsymbol{\theta}}$ and $\hat{\mu}$ for a polynomial dynamic errors-in-variables system with a relative distribution of noise φ between input and output (i.e. an input/output noise ratio), we may introduce a measure of dissimilarity between the noise present in the data sample and our noise model $\mathbf{C}_q = \mu\mathbf{C}(\varphi)\otimes\mathbf{I}_q$ with our assumption of φ. Let $d_F(\mathbf{A},\mathbf{B})$ be the Frobenius norm of $\mathbf{A}-\mathbf{B}$ such that

$$d_F(\mathbf{A},\mathbf{B}) = \|\mathbf{A}-\mathbf{B}\|$$

and $d_{IS}(\mathbf{A},\mathbf{B})$ be the Itakura–Saito divergence between matrices $\mathbf{A}$ and $\mathbf{B}$ such that

$$d_{IS}(\mathbf{A},\mathbf{B}) = \operatorname{trace}\left(\mathbf{A}^{-1}\mathbf{B}\right) - \log\det\left(\mathbf{A}^{-1}\mathbf{B}\right).$$

Then, the matrix divergence

$$d\left(\mathbf{G}_q^\top(\hat{\boldsymbol{\theta}})\mathbf{D}_q\mathbf{G}_q(\hat{\boldsymbol{\theta}}),\, \mathbf{G}_q^\top(\hat{\boldsymbol{\theta}})\mathbf{C}_q(\hat{\mu})\mathbf{G}_q(\hat{\boldsymbol{\theta}})\right)$$

measures how close our noise model is to the noise in the observed data. A covariance matching scheme over φ can then give an estimate for the relative noise distribution

$$\varphi = \arg\min_{\varphi} d\left(\mathbf{G}_q^\top(\hat{\boldsymbol{\theta}})\mathbf{D}_q\mathbf{G}_q(\hat{\boldsymbol{\theta}}),\, \mathbf{G}_q^\top(\hat{\boldsymbol{\theta}})\mathbf{C}_q(\hat{\mu})\mathbf{G}_q(\hat{\boldsymbol{\theta}})\right)$$

where the estimates $\hat{\boldsymbol{\theta}} = \hat{\boldsymbol{\theta}}(\varphi)$ and $\hat{\mu} = \hat{\mu}(\varphi)$ are computed for a particular φ by the PGKL method.

Using the results discussed so far, we may sketch the outline of the polynomial extension to the generalized Koopmans–Levin method with a covariance matching scheme as follows:

PGKL estimation method with noise covariance matching

1. Initialize with a relative distribution of noise φ between input and output.
2. Choose initial parameter and noise magnitude estimates $\boldsymbol{\theta}_{(0)}$ and $\mu_{(0)}$ with a non-iterative scheme (e.g. least squares).
3. Apply the PGKL iterative scheme to compute estimates $\hat{\boldsymbol{\theta}}$ and $\hat{\mu}$ using

$$\boldsymbol{\theta}_{(k+1)},\, \mu_{(k+1)} = \arg\min_{\boldsymbol{\theta},\mu} \frac{\boldsymbol{\theta}^\top\mathbf{T}^\top\left(\left(\mathbf{G}_{q,(k)}^\top\mathbf{C}_{q,(k)}(\varphi)\mathbf{G}_{q,(k)}\right)^{-1}\otimes\mathbf{D}_q\right)\mathbf{T}\boldsymbol{\theta}}{\boldsymbol{\theta}^\top\mathbf{T}^\top\left(\left(\mathbf{G}_{q,(k)}^\top\mathbf{C}_{q,(k)}(\varphi)\mathbf{G}_{q,(k)}\right)^{-1}\otimes\mathbf{C}_q(\mu,\varphi)\right)\mathbf{T}\boldsymbol{\theta}}$$

   where $\mathbf{G}_q = \mathbf{G}_q(\boldsymbol{\theta})$, $\mathbf{G}_{q,(k)} = \mathbf{G}_q\left(\boldsymbol{\theta}_{(k)}\right)$, $\mathbf{C}_q = \mathbf{C}_q(\mu)$ and $\mathbf{C}_{q,(k)} = \mathbf{C}_q\left(\mu_{(k)}\right)$, which reduces to a polynomial eigenvalue problem

$$\boldsymbol{\Psi}(\mu)\boldsymbol{\theta} = \mathbf{0}.$$

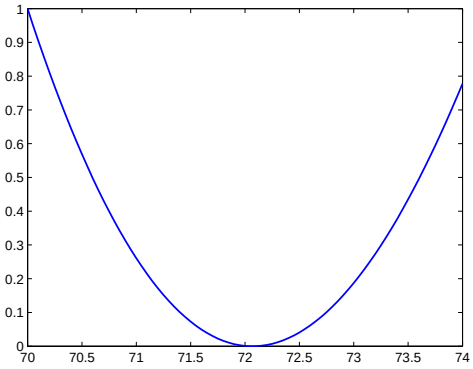


Figure 4.2: Discovering the input/output noise ratio using the Itakura–Saito matrix divergence.

	mean	±	deviation
$\hat{\varphi}$	72.4195	±	0.9017
$\hat{\mu}$	0.011052	±	0.000238
$\hat{\sigma}_u^2$	0.001010	±	0.000096
$\hat{\sigma}_y^2$	0.010042	±	0.000266

Table 4.2: Mean and variance of noise parameters in a 100-run Monte-Carlo study.

4. Minimize the divergence $d\left(\mathbf{G}_q^\top(\hat{\boldsymbol{\theta}})\mathbf{D}_q\mathbf{G}_q(\hat{\boldsymbol{\theta}}), \mathbf{G}_q^\top(\hat{\boldsymbol{\theta}})\mathbf{C}_q(\hat{\mu})\mathbf{G}_q(\hat{\boldsymbol{\theta}})\right)$ over φ.

Example 14. Consider the same system as in Example 13 but suppose the noise magnitudes $\sigma_u^2 = 0.001$ and $\sigma_y^2 = 0.01$ are unknown. With $\sigma_u = \mu\cos\varphi$ and $\sigma_y = \mu\sin\varphi$ we get that the relative input/output noise ratio $\varphi^\star = 72.45°$ where $\varphi^\star$ indicates true value. Varying the unknown variable φ in a range between $\varphi = 72°$ and $\varphi = 74°$ with a step size of 0.05, we can compute parameter estimates $\hat{\boldsymbol{\theta}}$ for each angle φ. Given the parameter estimates, we may evaluate the matrix divergence d_{IS} for each φ and plot the error measure. Figure 4.2 graphs the unit-normalized error measure against the angle φ. It is apparent that the estimated value minimizes near the true value. ♣

Example 15. In order to assess the accuracy of noise parameter estimates, let us conduct a Monte-Carlo simulation of 100 runs. Minimizing the distance measure d_{IS} in each run, we obtain estimates $\hat{\varphi}$ and $\hat{\mu}$. Table 4.2 shows the mean values of $\hat{\varphi}$ and $\hat{\mu}$ as well as $\hat{\sigma}_u^2$ and $\hat{\sigma}_y^2$ with their respective standard deviations.

♣

4.3 Summary

Following our investigation of the parameter estimation problem in Chapter 2 and the structure discovery problem in Chapter 3, both for static systems, we ventured onto the field of dynamic systems, where we considered discrete-time systems that can be re-cast in a linear

setting using a polynomial lifting function. With both the system input and output contaminated with noise, the identification problem that attempts to estimate model and noise parameters is even more difficult than in the linear errors-in-variables case. Nevertheless, combining the generalization of the Koopmans–Levin (GKL) method for linear dynamic systems, and the nonlinear extension to the Koopmans (NK) method for static systems, we have seen that the polynomial generalized Koopmans–Levin (PGKL) can deliver better estimates than other methods, inspired by the bias compensation and the least squares principle.

New contributions on the field of dynamic errors-in-variables systems include:

- an iterative algorithm that combines the GKL and the NK methods;
- applying a symmetric linearization of a polynomial eigenvalue problem to estimate model parameters;
- a noise covariance matching scheme using matrix divergences to estimate noise parameters.

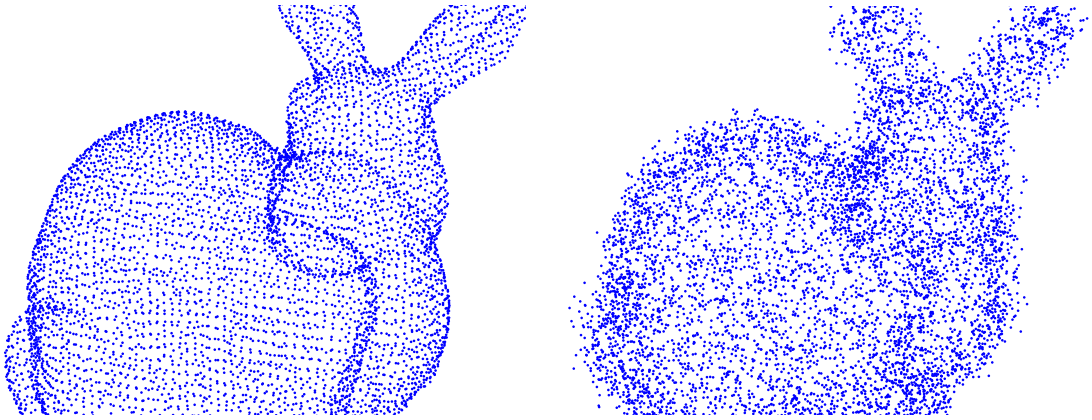
CHAPTER

5 Applications

The errors-in-variables approach is frequently encountered in several fields where the goal is reconstructing a model from a noisy point cloud. Notably, in the context of computer vision and pattern recognition, an important sub-problem is parametric curve and surface fitting. Our results in Chapter 2 are directly applicable to estimating parameters of quadratic curves and surfaces at modest computational cost. The estimates obtained with the proposed methods are close to those obtained with the maximum likelihood approach. On the other hand, maximum likelihood estimation entails iterations and each iteration involves projection to a nonlinear curve or surface, which are costly operations. The proposed methods, in contrast, are inexpensive to implement and they typically involve solving one or two eigenvalue or singular value problems.

Taking a step further, a wide variety of problems from computer vision, such as motion segmentation and face clustering on polygonal surfaces, involve data modeling by multiple subspaces. In tracking-based motion segmentation, feature points are clustered according to the different moving objects. Under the affine camera model, the vectors of feature point coordinates corresponding to a moving rigid object lie on an affine subspace of dimension at most three [19]. Thus, clustering different moving objects is equivalent to clustering different affine subspaces. Similarly, in face clustering, it has been proved that the set of all images of a Lambertian object under a variety of lighting conditions form a convex polyhedral cone in the image space, and this cone can be accurately approximated by a low-dimensional linear subspace of dimension at most 9 [8, 32]. The data modeling methods in Chapter 3 that operate with nonlinear manifolds can be applied in these contexts.

Self-organizing methods that autonomously discover a model from a set of (noisy) observations, such as the methods presented in Chapter 3, also facilitate geometric modeling. Reconstructing a surface as the zero-set of a scalar-valued composite implicit function $f(\mathbf{x}) = 0$ greatly simplifies constructive solid geometry (CSG) operations such as boolean union, difference and intersection. CSG operations are, in general, difficult to perform on boundary representations (such as polygonal mesh surfaces) as one has to take precautions to ensure the compactness ("water-tightness") of the resultant surface, which can be important in manufacturing and engineering applications. On the contrary, these operations are straight-



(a) noise-free model **(b)** noisy data source

Figure 5.1: Points extracted from the laser-scanned model of the Stanford Bunny.

forward on an implicit function representation, with no need for additional topological data or consistency checks. For instance, let $f_1(\mathbf{x}) = 0$ and $f_2(\mathbf{x}) = 0$ be implicit function representations of two objects, then $\max\big(f_1(\mathbf{x}), f_2(\mathbf{x})\big)$ defines the intersection, $\min\big(f_1(\mathbf{x}), f_2(\mathbf{x})\big)$ the union, provided that $f(\mathbf{x}) < 0$ is the inside of the object. For a similar reason, containment and collision detection are relatively easily performed on an implicit function representation. While common practice in implicit function reconstruction involves approximating objects with spheres or cuboids, allowing a more general set of shapes that better fit the original object leads to better approximation with possibly fewer parameters.

Figure 5.1 shows a data set comprising of points extracted from the laser-scanned model of the Stanford Bunny[1] [60], and the same data set polluted with Gaussian noise. Such a large data set takes up considerable storage, fails to grasp structure and does not easily lend itself to easy geometric manipulation. In contrast, the reconstructed model in Figure 5.2 that comprises mostly of ellipsoids, occupies much less storage, helps understand composition and structure, and allows geometric manipulation such as with CSG operators.

Finally, the system identification method in Chapter 4 is directly applicable to single-input single-output systems especially where we seek a low-order polynomial representation of a system under study. Most physical systems are inherently nonlinear and they are usually approximated with linear models for a particular operating range. There has been, however, a desire to extend techniques developed within the theoretical framework of linear systems to handle nonlinear systems. While the class of nonlinear systems is rather broad as the general term does not reflect the exact form of the nonlinearity manifested in the system, a polynomial approximation is a common way to deal with a system governed by a nonlinear function. The flexibility of polynomial models allows satisfactory approximation for many nonlinear processes over wide ranges of operation, with higher polynomial degree better approximating the original process. As a result, the use of such an approach increases the

[1] The Stanford Bunny is one of the most commonly used test models in computer graphics; it is a collection of 69,451 triangles with 35,947 vertices, and it was assembled from range images of a clay bunny that is roughly 7.5 inches high. The points in Figure 5.1 have been obtained by discarding point connectivity information (i.e. which point triplets form a triangle) and triangle surface normal vectors.

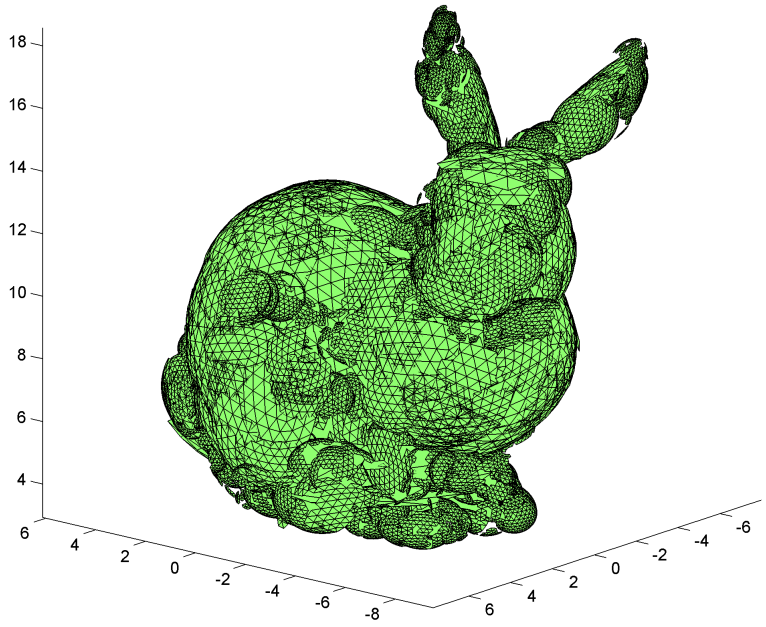


Figure 5.2: Scattered points of the Stanford Bunny captured with at most quadratic 3D shapes.

accuracy of the models substantially as compared to the standard linear modeling approach [54]. The method outlined in Chapter 4 alloys the benefit of increased flexibility of polynomial models with the generalized system description the errors-in-variables approach offers by treating both the input and the output sequence as data polluted with noise. Therefore, the proposed method can be employed in a wide range of contexts where a linear model would be inadequate but noise-free input data is not available either.

In order to facilitate integrating the proposed algorithms into future software, the thesis is only complete with a reference implementation. MatLab source code with several examples demonstrate the main points covered in the thesis, and provide a toolbox for solving unconstrained and constrained curve and surface fitting, clustering and dynamic system parameter estimation problems. The implementation includes the nonlinear Koopmans estimator for unconstrained fitting with matrix size reduction, as well as extensions to constrained fitting of 2D shapes lines, circles, ellipsoids, parabolas, hyperbolas, and 3D shapes planes, spheres, ellipsoids. For comparison, direct (non-iterative) methods and maximum likelihood methods are available. For solving the clustering problem, various projection schemes and (symmetric and asymmetric) spectral clustering methods are included in addition to the iterative and non-iterative methods introduced in the thesis. Related to dynamic system parameter estimation, both linear and polynomial methods are part of the implementation.

CHAPTER

6 Conclusions

The thesis has covered estimation problems of errors-in-variables systems, with emphasis on three important fields of research. First, the parametric estimation problem was investigated where data are assumed to be captured by a single nonlinear function, known up to a few model parameters. Applying the noise cancellation principle, a new constrained estimation approach was introduced for fitting several types of quadratic curves and surfaces. Second, the fast and robust non-iterative unconstrained and constrained estimation methods introduced in Chapter 2 were applied in Chapter 3 to build a new clustering algorithm that autonomously partitions a data set, comprising of several groups of points, each related by a single parametric function. Finally, our focus shifted from static systems to dynamic systems in Chapter 4, where we saw how estimation methods for linear dynamic systems and nonlinear static systems may be alloyed to construct an efficient model and noise parameter estimation scheme. The new scientific contributions of the dissertation are summarized below.

6.1 New contributions

Thesis 1. Fitting ellipses, parabolas, hyperbolas and ellipsoids with noise cancellation

I have proposed an errors-in-variables parameter estimation method fitting quadratic curves and surfaces subject to constraints. The method incorporates a noise cancellation step in existing constrained quadratic least squares fitting algorithms. The noise cancellation step can be written as a quadratic eigenvalue problem on symmetric matrices, where the eigenvalue problem yields a data covariance matrix with noise distortions being accounted for, and may be formulated without the statistically invariant terms in the data covariance matrix, leading to a simpler expression. Operating on a noise-compensated data covariance matrix, direct ellipse-specific, parabola-specific, hyperbola-specific and ellipsoid-specific least squares fitting exhibit substantially improved accuracy compared to their original formulation without noise cancellation.

The new errors-in-variables parameter estimation method for quadratic curves and surfaces subject to constraints comprises of the following two major steps:

1. noise cancellation
2. constrained quadratic least squares fitting

The first major part of the algorithm is the noise cancellation scheme for general quadratic curves and surfaces, which can be broken down into the following steps:

1. *Input:* noisy samples $\mathbf{x}_i$ and the relative noise magnitude vector $\bar{\sigma}_{\mathbf{x}}^2$ for each dimension.
2. Estimate the data covariance matrix $\mathbf{D}$ from noisy samples.
3. Estimate the noise covariance matrix polynomial coefficients $\mathbf{C}_1$ and $\mathbf{C}_2$ from noisy samples.
4. Compute the reduced-size matrices $\mathbf{D}_\star$, $\mathbf{C}_\star$ and $\mathbf{C}_{\star\star}$ by eliminating the statistically invariant terms in $\mathbf{D}$, $\mathbf{C}_1$ and $\mathbf{C}_2$.
5. Construct the matrix polynomial $\mathbf{\Psi}_\star(\mu) = \mathbf{D}_\star - \mu\mathbf{C}_\star - \mu^2\mathbf{C}_{\star\star}$.
6. Find the eigenvalue μ that solves $\det\left(\mathbf{\Psi}_\star(\mu)\right) = 0$.
7. *Output:* the noise-compensated (singular) matrix $\mathbf{R}_\star = \mathbf{D}_\star - \mu\mathbf{C}_\star - \mu^2\mathbf{C}_{\star\star}$.

The second major part is constrained fitting, which depends on the type of quadratic curve or surface the constraint identifies.

The following algorithm summarizes the constrained estimation scheme for ellipses and hyperbolas:

1. Construct the reduced scatter matrix $\mathbf{S}$ by writing the Schur complement of the quadratic terms in $\mathbf{R}_\star$.
2. Solve the generalized eigenvalue problem $\mathbf{S}\mathbf{g}_1 = \lambda\mathbf{Q}_1\mathbf{g}_1$ (or the corresponding singular value problem) for $\mathbf{g}_1$ with $\mathbf{Q}_1$ expressing the ellipse- and hyperbola-specific constraint.
3. Classify the eigenvectors $\mathbf{g}_{1,k}$ based on the eigenvalue λ_k and the constraint value $\mathbf{g}_{1,k}^\top\mathbf{Q}_1\mathbf{g}_{1,k}$.
4. Recover the original parameter vector $\mathbf{g}$.

The following algorithm summarizes parabola fitting:

1. Compute the eigenvector decomposition $\mathbf{S}\mathbf{g}_1 = \lambda\mathbf{g}_1$ with $\mathbf{S}$ being the Schur complement of the quadratic terms in $\mathbf{R}_\star$.
2. Based on $\mathbf{g}_1 = \mathbf{v}_1 + s\mathbf{v}_2 + t\mathbf{v}_3$, write the Lagrangian $\mathscr{L}(s,t,\alpha) = \mathscr{F}(s,t) + \alpha\mathscr{C}(s,t)$.
3. Solve the Lagrangian polynomial for α to recover the parameter vector $\mathbf{g}$.

The following algorithm summarizes the constrained estimation scheme for ellipsoids:

1. Construct the reduced scatter matrix $\mathbf{S}$ by writing the Schur complement of the quadratic terms in $\mathbf{R}_\star$.

2. Solve the generalized eigenvalue problem $\mathbf{S}\mathbf{g}_1 = \lambda\mathbf{Q}_1(k)\mathbf{g}_1$ (or the corresponding singular value problem) for $\mathbf{g}_1$ with $\mathbf{Q}_1(k)$ expressing the ellipsoid-specific constraint with a large $k \geqq 4$.

3. Use a bisection method on k until $\mathbf{g}_1$ identifies an ellipsoid.

4. Recover the original parameter vector $\mathbf{g}$.

Thesis 2.1. Iterative algorithm for manifold clustering

I have generalized linear grouping algorithms [10, 64, 2, 1] for finding linear patterns in a data set to a grouping algorithm fitting quadratic curves and surfaces. Alternating between an update (parameter estimation) and an assignment (data mapping) step, the method can discover a structural decomposition of data where members of each group are related by a low-order (linear or quadratic) implicit polynomial function.

The outline of the proposed iterative grouping algorithm resembles the iterative algorithm of the standard k-means procedure. The primary difference lies in the use of parameters instead of mean values, and simple point-to-point distance (between cluster center and data points) replaced with data point projection. As with the standard k-means algorithm, the choice of initial data points affects convergence to an optimal solution.

1. Initialization. Choose k randomly chosen initial points $\mathbf{x}_i$ with $i = 1, \ldots, k$ and for each data point $\mathbf{x}_i$

    a) start with an initial neighborhood $\mathcal{N}(\mathbf{x}_i)$ around $\mathbf{x}_i$

    b) estimate the parameters $\boldsymbol{\theta}_{i,(0)}$ that best capture points in $\mathcal{N}(\mathbf{x}_i)$

    c) enlarge $\mathcal{N}(\mathbf{x}_i)$ by adding nearest-neighbor points

    d) compute new estimates $\boldsymbol{\theta}_{i,(n)}$ and compare them to the estimates $\boldsymbol{\theta}_{i,(n-1)}$ obtained in the previous iteration

    e) repeat until the neighborhood $\mathcal{N}(\mathbf{x}_i)$ cannot be enlarged without worsening the accuracy of $\boldsymbol{\theta}_{i,(n)}$

    f) let $\boldsymbol{\theta}_i$ be the best $\boldsymbol{\theta}_{i,(n)}$ that belongs to the optimum $\mathcal{N}(\mathbf{x}_i)$ around $\mathbf{x}_i$

2. Initial grouping.

    a) project each data point $\mathbf{x}_j$ with $j = 1, \ldots, N$ to all candidate $\boldsymbol{\theta}_i$ with $i = 1, \ldots, k$

    b) form initial groups $\mathcal{S}_i$ with $i = 1, \ldots, k$ such that the distance is minimized.

3. Alternating optimization.

    a) Update step. Each group of data points $\mathscr{S}_i$ is used to estimate new shape parameters $\boldsymbol{\theta}_i$.

    b) Assignment step. Each data point $\mathbf{x}_j$ is assigned to shapes $\boldsymbol{\theta}_i$ it lies in the vicinity of.

4. Finalization. Each data point is assigned to a single shape $\boldsymbol{\theta}_i$ it lies closest to.

5. Re-sampling. Repeat the algorithm with different randomly chosen initial locations.

Thesis 2.2. Non-iterative algorithm for manifold clustering

I have proposed a clustering method fitting implicit polynomial functions that finds an initial segmentation without a preliminary assignment of data points. The clustering problem is modeled with a complete directed weighted graph where the nodes are data points, and the edge weights a_{ij} are an affinity measure reflecting the distance between a data point i and the curve or surface estimated from points in the neighborhood of the other point j. A best weighted cut algorithm is applied to split the graph into k components, and thereby produce an asymmetric spectral clustering of the data set with the given distance measure. The proposed method surpasses other manifold clustering methods that do not explicitly incorporate function fitting to data in clusters.

Grouping with spectral clustering addresses remedies the issue of wrongly chosen initial locations. Spectral clustering eliminates the need for re-sampling and provides a clustering that is already close to an optimum solution.

1. Initialization. For each data point $\mathbf{x}_i$ with $i = 1, \ldots, N$ and for each data point $\mathbf{x}_i$

    a) start with an initial neighborhood $\mathscr{N}(\mathbf{x}_i)$ around $\mathbf{x}_i$

    b) estimate the parameters $\boldsymbol{\theta}_{i,(0)}$ that best capture points in $\mathscr{N}(\mathbf{x}_i)$

    c) enlarge $\mathscr{N}(\mathbf{x}_i)$ by adding nearest-neighbor points

    d) compute new estimates $\boldsymbol{\theta}_{i,(n)}$ and compare them to the estimates $\boldsymbol{\theta}_{i,(n-1)}$ obtained in the previous iteration

    e) repeat until the neighborhood $\mathscr{N}(\mathbf{x}_i)$ cannot be enlarged without worsening the accuracy of $\boldsymbol{\theta}_{i,(n)}$

    f) let $\boldsymbol{\theta}_{\mathscr{N}(\mathbf{x}_i)}$ be the best $\boldsymbol{\theta}_{i,(n)}$ that belongs to the optimum $\mathscr{N}(\mathbf{x}_i)$ around $\mathbf{x}_i$

2. Compute asymmetric distances. For each pair of data points $\mathbf{x}_i$ and $\mathbf{x}_j$, obtain their asymmetric distances

    a) project $\mathbf{x}_i$ to the manifold parametrized by $\boldsymbol{\theta}_{\mathscr{N}(\mathbf{x}_j)}$ and calculate a_{ij}

    b) project $\mathbf{x}_j$ to the manifold parametrized by $\boldsymbol{\theta}_{\mathscr{N}(\mathbf{x}_i)}$ and calculate a_{ji}

3. Asymmetric spectral clustering.

   a) build an affinity matrix $\mathbf{H} = \mathbf{H}(\mathbf{A})$ from asymmetric distances a_{ij} where $\mathbf{A} = \left[a_{ij}\right]$

   b) assign data points to groups based on the affinity matrix eigenvectors

Thesis 3. Iterative method to estimate parameters of dynamic polynomial systems

*I have proposed the polynomial extension to the generalized Koopmans–Levin (*PGKL*) estimation method for dynamic systems by unifying the nonlinear Koopmans (*NK*) method for static polynomial systems with the generalized Koopmans–Levin (*GKL*) method for linear dynamic systems. The algorithm is formulated as an iterative procedure. I have demonstrated how a generalized eigenvalue problem as used with the iterative approach of the* GKL *method extends to a polynomial eigenvalue problem if the errors can be modeled as Gaussian noise. I have utilized a symmetric linearization of the polynomial eigenvalue problem to preserve the symmetry present in the original problem and maintain numerical robustness. The* PGKL *estimation method assumes a known relative distribution of noise between system input and output. I have shown how a noise covariance matching approach can estimate the relative noise distribution if this parameter is unknown.*

The outline of the polynomial extension to the generalized Koopmans–Levin method with a covariance matching scheme is as follows:

1. Initialize with a relative distribution of noise φ between input and output.

2. Choose initial parameter and noise magnitude estimates $\boldsymbol{\theta}_{(0)}$ and $\mu_{(0)}$ with a non-iterative scheme (e.g. least squares).

3. Apply the PGKL iterative scheme to compute estimates $\hat{\boldsymbol{\theta}}$ and $\hat{\mu}$ using

$$\boldsymbol{\theta}_{(k+1)},\, \mu_{(k+1)} = \arg\min_{\boldsymbol{\theta},\mu} \frac{\boldsymbol{\theta}^\top \mathbf{T}^\top \left(\left(\mathbf{G}_{q,(k)}^\top \mathbf{C}_{q,(k)}(\varphi) \mathbf{G}_{q,(k)} \right)^{-1} \otimes \mathbf{D}_q \right) \mathbf{T} \boldsymbol{\theta}}{\boldsymbol{\theta}^\top \mathbf{T}^\top \left(\left(\mathbf{G}_{q,(k)}^\top \mathbf{C}_{q,(k)}(\varphi) \mathbf{G}_{q,(k)} \right)^{-1} \otimes \mathbf{C}_q(\mu, \varphi) \right) \mathbf{T} \boldsymbol{\theta}}$$

   where $\mathbf{G}_q = \mathbf{G}_q(\boldsymbol{\theta})$, $\mathbf{G}_{q,(k)} = \mathbf{G}_q\left(\boldsymbol{\theta}_{(k)}\right)$, $\mathbf{C}_q = \mathbf{C}_q(\mu)$ and $\mathbf{C}_{q,(k)} = \mathbf{C}_q\left(\mu_{(k)}\right)$, which reduces to a polynomial eigenvalue problem

$$\boldsymbol{\Psi}(\mu)\boldsymbol{\theta} = \mathbf{0}.$$

4. Minimize the divergence $d\left(\mathbf{G}_q^\top(\hat{\boldsymbol{\theta}}) \mathbf{D}_q \mathbf{G}_q(\hat{\boldsymbol{\theta}}),\, \mathbf{G}_q^\top(\hat{\boldsymbol{\theta}}) \mathbf{C}_q(\hat{\mu}) \mathbf{G}_q(\hat{\boldsymbol{\theta}})\right)$ over φ.

6.2 Future work

We have seen that algorithms in Chapter 2 offer a computationally cheap alternative to constrained fitting; the grouping methods in Chapter 3 find a segmentation of a data set into feasible partitions; and the parameter estimation method in Chapter 4 outperforms other

algorithms in the field of dynamic errors-in-variables systems with polynomial nonlinearities. Nonetheless, several open questions remain. The algorithms in Chapter 2 could be extended to cover other quadratic curve and surface types beyond ellipses, parabolas, hyperbolas and ellipsoids. The grouping methods in Chapter 3 do not address how to choose the number of groups to partition the data into, or how to deal with outliers any least-squares minimizing method is sensitive to. Furthermore, the computational cost of the proposed algorithms could be dramatically reduced by more strategic selection of points that generate neighborhoods and employing incremental techniques to exploit the similarity of parameter estimates obtained from the neighborhood of data points that are close to one another. Finally, Chapter 4 involves calculations with large matrices, part of which could be omitted without serious loss of computational accuracy. These issues motivate future research work.

Bibliography

[1] Stefan Van Aelst, Xiaogang (Steven) Wang, Ruben H. Zamar, and Rong Zhu. Linear grouping using orthogonal regression. *Computational Statistics and Data Analysis*, 50 (5):1287–1312, March 2006. doi: 10.1016/j.csda.2004.11.011.

[2] Pankaj K. Agarwal and Nabil H. Mustafa. k-means projective clustering. In *Proc. of 23rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 155–165, 2004. doi: 10.1145/1055558.1055581.

[3] Juan C. Agüero and Graham C. Goodwin. Identifiability of errors in variables dynamic systems. *Automatica*, 44(2):371–382, 2008. ISSN 0005-1098. doi: 10.1016/j.automatica.2007.06.011.

[4] Martin Aigner and Bert Jüttler. Robust fitting of implicitly defined surfaces using Gauss–Newton-type techniques. *The Visual Computer: International Journal of Computer Graphics*, 25(8):731–741, 2009. ISSN 0178-2789. doi: 10.1007/s00371-009-0361-1.

[5] Ali Al-Sharadqah and Nikolai Chernov. A doubly optimal ellipse fit. *Computational Statistics and Data Analysis*, 56(9):2771–2781, September 2012. doi: 10.1016/j.csda.2012.02.028.

[6] Efstathios N. Antoniou and Stavros Vologiannidis. Linearizations of polynomial matrices with symmetries and their applications. *Electronic Journal of Linear Algebra*, 15: 107–114, February 2006.

[7] Stefan Emilov Atev. *Using asymmetry in the spectral clustering of trajectories*. PhD thesis, University of Minnesota, July 2011. URL http://purl.umn.edu/112921.

[8] Ronen Basri and David W. Jacobs. Lambertian reflectance and linear subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(2):218–233, February 2003. doi: 10.1109/TPAMI.2003.1177153.

[9] Sergio Beghelli, Roberto P. Guidorzi, and Umberto Soverini. The Frisch scheme in dynamic system identification. *Automatica*, 26(1):171–176, January 1990. doi: 10.1016/0005-1098(90)90168-H.

[10] Paul S. Bradley and Olvi L. Mangasarian. k-plane clustering. *Journal of Global Optimization*, 16(1):23–32, January 2000. doi: 10.1023/A:1008324625522.

[11] Guangliang Chen and Gilad M. Lerman. Spectral curvature clustering. *International Journal of Computer Vision*, 81(3):317–330, 2009. doi: 10.1007/s11263-008-0178-9.

[12] Guangliang Chen, Stefan Atev, and Gilad Lerman. Kernel spectral curvature clustering (KSCC). In *Proc. of IEEE International Conference on Computer Vision (ICCV 2009)*, pages 765–772, Kyoto, Japan, September 2009. doi: 10.1109/ICCVW.2009.5457627.

[13] Nikolai Chernov and Claire Lesort. Statistical efficiency of curve fitting algorithms. *Computational Statistics and Data Analysis*, 47(4):713–728, November 2004. doi: 10.1016/j.csda.2003.11.008.

[14] Nikolai Chernov and Hui Ma. Least squares fitting of quadratic curves and surfaces. In Sota R. Yoshida, editor, *Computer Vision*, pages 285–302. Nova Science Publishers, 2011. ISBN 978-1-61209-399-4.

[15] Nikolai Chernov, Claire Lesort, and Nándor Simányi. On the complexity of curve fitting algorithms. *Journal of Complexity*, 20:484–492, August 2004. doi: 10.1016/j.jco.2004.01.004.

[16] Pramod N. Chivate and Andrei G. Jablokow. Solid-model generation from measured point data. *Computer-Aided Design*, 25(9):587–600, 1993. doi: 10.1016/0010-4485(93)90074-X.

[17] Wojciech Chojnacki and Michael J. Brooks. Revisiting Hartley's normalized eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25 (9):1172–1177, September 2003. doi: 10.1109/TPAMI.2003.1227992.

[18] Wojciech Chojnacki, Michael J. Brooks, Anton van den Hengel, and Darren Gawley. FNS, CFNS and HEIV: A unifying approach. *Journal of Mathematical Imaging and Vision*, 23(2):175–183, September 2005. doi: 10.1007/s10851-005-6465-y.

[19] João Paulo Costeira and Takeo Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29(3):159–179, September 1998. ISSN 0920-5691. doi: 10.1023/A:1008000628999.

[20] Peter de Groen. An introduction to total least squares. *Nieuw Archief voor Wiskunde*, 4(14):237–253, 1996.

[21] Roberto Diversi, Roberto Guidorzi, and Umberto Soverini. A new criterion in EIV identification and filtering applications. In *Proc. of 13th IFAC Symposium on System Identification*, pages 1993–1998, 2003.

[22] David Eberly. Distance from point to a general quadratic curve or a general quadric surface, 1999, 2008. http://www.geometrictools.com/Documentation/DistancePointToQuadratic.pdf.

[23] David Eberly. Distance from a point to an ellipse, an ellipsoid, or a hyperellipsoid, 2011. http://www.geometrictools.com/Documentation/DistancePointEllipseEllipsoid.pdf.

[24] Mats Ekman, Mei Hong, and Torsten Söderström. A separable nonlinear least-squares approach for identification of linear systems with errors in variables. In *Proc. of 14th IFAC Symposium on System Identification*, pages 178–183, March 2006. doi: 10.3182/20060329-3-AU-2901.00022.

[25] Ehsan Elhamifar and René Vidal. Sparse subspace clustering. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2797, 2009. doi: 10.1109/CVPR.2009.5206547.

[26] Andrew W. Fitzgibbon, Maurizio Pilu, and Robert B. Fisher. Direct least squares fitting of ellipses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):476–480, May 1999. doi: 10.1109/34.765658.

[27] Alvina Goh and René Vidal. Segmenting motions of different types by unsupervised manifold clustering. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, June 2007. doi: 10.1109/CVPR.2007.383235.

[28] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins University Press, Baltimore, MD, USA, 3rd edition, 1996. ISBN 0-8018-5414-8.

[29] Gaël Guennebaud and Markus Gross. Algebraic point set surfaces. *ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH 2007*, 26(3), July 2007. doi: 10.1145/1275808.1276406. Article No. 23.

[30] Radim Halíř and Jan Flusser. Numerically stable direct least squares fitting of ellipses. In Vaclav Skala, editor, *Proc. of International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media*, pages 125–132, 1998.

[31] Matthew Harker, Paul O'Leary, and Paul Zsombor-Murray. Direct type-specific conic fitting and eigenvalue bias correction. *Image and Vision Computing*, 26(3):372–381, March 2008. doi: 10.1016/j.imavis.2006.12.006.

[32] Jeffrey Ho, Ming-Hsuan Yang, Jongwoo Lim, Kuang-Chih Lee, and David Kriegman. Clustering appearances of objects under varying illumination conditions. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages 11–18, 2003. ISBN 0-7695-1900-8. doi: 10.1109/CVPR.2003.1211332.

[33] Mei Hong, Torsten Söderström, and Wei Xing Zheng. Accuracy analysis of bias-eliminating least squares estimates for errors-in-variables systems. *Automatica*, 43 (9):1590–1596, 2007. ISSN 0005-1098. doi: 10.1016/j.automatica.2007.02.002.

[34] Mei Hong, Torsten Söderström, Umberto Soverini, and Roberto Diversi. Comparison of three Frisch methods for errors-in-variables identification. In *Proc. of 17th IFAC World Congress*, pages 420–425, Seoul, Korea, July 2008. doi: 10.3182/20080706-5-KR-1001.00070.

[35] Sabine Van Huffel and Philippe Lemmerling, editors. *Total Least Squares and Errors-in-Variables Modeling: Analysis, Algorithms and Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, February 2002. ISBN 978-1402004766.

[36] Kenichi Kanatani. Statistical bias of conic fitting and renormalization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(3):320–326, March 1994. doi: 10.1109/34.276132.

[37] Kenichi Kanatani. Further improving geometric fitting. In *Proc. of 5th International Conference on 3-D Digital Imaging and Modeling*, pages 2–13, Ottawa, Ontario, Canada, June 2005.

[38] Kenichi Kanatani. Statistical optimization for geometric fitting: Theoretical accuracy bound and high order error analysis. *International Journal of Computer Vision*, 80(2): 167–188, November 2008. doi: 10.1007/s11263-007-0098-0.

[39] Kenichi Kanatani and Prasanna Rangarajan. Hyper least squares fitting of circles and ellipses. *Computational Statistics and Data Analysis*, 55(6):2197–2208, June 2011. doi: 10.1016/j.csda.2010.12.012.

[40] Tjalling C. Koopmans. Linear regression analysis of economic time series. DeErven F. Bohn, Haarlem, Netherlands, 1937.

[41] Oluwasanmi Koyejo and Joydeep Ghosh. MiPPS: A generative model for multi-manifold clustering. In *Manifold Learning and its Applications: Papers from the AAAI Fall Symposium*, 2010.

[42] Alexander G. Kukush, Ivan Markovsky, and Sabine Van Huffel. Consistent estimation in an implicit quadratic measurement error model. *Computational Statistics and Data Analysis*, 47(1):123–147, August 2004. doi: 10.1016/j.csda.2003.10.022.

[43] Tomasz Larkowski, Jens G. Linden, and Keith J. Burnham. Identification of dynamic nonlinear polynomial models in the errors-in-variables framework. In *Proc. of 15th IFAC Symposium on System Identification*, pages 1580–1585, Saint-Malo, France, 2009. doi: 10.3182/20090706-3-FR-2004.00262.

[44] Morris J. Levin. Estimation of a system pulse transfer function in the presence of noise. In *Proc. of Joint Automatic Control Conference*, pages 452–458, 1963.

[45] Qingde Li and John G. Griffiths. Least squares ellipsoid specific fitting. In *Proc. of Geometric Modeling and Processing*, pages 335–340, 2004. doi: 10.1109/GMAP.2004.1290055.

[46] Yi Ma, Harm Derksen, Wei Hong, and John Wright. Segmentation of multivariate mixed data via lossy coding and compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(9):1546–1562, August 2007. doi: 10.1109/TPAMI.2007.1085.

[47] David MacKay. *Information Theory, Inference and Learning Algorithms*, chapter 20. An Example Inference Task: Clustering, pages 284–292. Cambridge University Press, 2003. ISBN 0-521-64298-1. http://www.inference.phy.cam.ac.uk/mackay/itprnn/ps/284.292.pdf.

[48] Ivan Markovsky, Alexander G. Kukush, and Sabine Van Huffel. Consistent least squares fitting of ellipsoids. *Numerische Mathematik*, 98(1):177–194, July 2004. doi: 10.1007/s00211-004-0526-9.

[49] Bogdan Matei and Peter Meer. A general method for errors-in-variables problems in computer vision. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages 18–25, June 2000. doi: 10.1109/CVPR.2000.854727.

[50] Marina Meilă and William Pentney. Clustering by weighted cuts in directed graphs. In *Proc. of 7th SIAM International Conference on Data Mining*, Minneapolis, Minnesota, USA, April 2007.

[51] Marina Meilă and Jianbo Shi. A random walks view of spectral segmentation. In Tommi Jaakkola and Thomas Richardson, editors, *Proc. of 8th International Workshop on Artificial Intelligence and Statistics*, Key West, FL, USA, 2001. URL http://www.gatsby.ucl.ac.uk/aistats/aistats2001/.

[52] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. *ACM Trans. Graph.*, 22(3):463–470, 2003. ISSN 0730-0301. doi: 10.1145/882262.882293.

[53] Paul O'Leary and Paul J. Zsombor-Murray. Direct and specific least-square fitting of hyperbolæ and ellipses. *Journal of Electronic Imaging*, 13(3):492–503, July 2004. doi: 10.1117/1.1758951.

[54] Ronald K. Pearson. *Discrete-Time Dynamic Models*. Oxford University Press, New York, USA, 1999.

[55] Maurizio Pilu, Andrew W. Fitzgibbon, and Robert B. Fisher. Ellipse-specific direct least-square fitting. In *Proc. of IEEE International Conference on Image Processing*, pages 599–602, Lausanne, September 1996. doi: 10.1109/ICIP.1996.560566.

[56] René Schöne and Tobias Hanning. Least squares problems with absolute quadratic constraints. *Journal of Applied Mathematics*, 2012. doi: 10.1155/2012/312985. Article ID 312985.

[57] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, August 2000. doi: 10.1109/34.868688.

[58] Torsten Söderström. Errors-in-variables methods in system identification. *Automatica*, 43(6):939–958, June 2007. doi: 10.1016/j.automatica.2006.11.025.

[59] Richard Souvenir and Robert Pless. Manifold clustering. In *Proc. of 10th International Conference on Computer Vision*, pages 648–653, 2005.

[60] Stanford Bunny. The "Stanford Bunny" from the Stanford University Computer Graphics Laboratory, 1994. http://graphics.stanford.edu/data/3Dscanrep/.

[61] G. 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, November 1991. doi: 10.1109/34.103273.

[62] Stéphane Thil, Marion Gilson, and Hugues Garnier. On instrumental variable-based methods for errors-in-variables model identification. In *Proc. of 17th IFAC World Congress*, pages 426–431, Seoul, Korea, July 2008. doi: 10.3182/20080706-5-KR-1001.00072.

[63] Françoise Tisseur and Karl Meerbergen. The quadratic eigenvalue problem. *SIAM Review*, 43(2):235–286, 2001.

[64] Paul Tseng. Nearest q-flat to m points. *Journal of Optimization Theory and Applications*, 105(1):249–252, 2000. doi: 10.1023/A:1004678431677.

[65] István Vajk. Identification methods in a unified framework. *Automatica*, 41(8):1385–1393, 2005. doi: 10.1016/j.automatica.2005.03.012.

[66] István Vajk and Jenő Hetthéssy. Identification of nonlinear errors-in-variables models. *Automatica*, 39:2099–2107, 2003. doi: 10.1016/j.automatica.2003.06.001.

[67] István Vajk and Jenő Hetthéssy. Efficient estimation of errors-in-variables models. In *Proc. of 17th IFAC World Congress*, pages 1384–1389, Seoul, Korea, July 2008. doi: 10.3182/20080706-5-KR-1001.00237.

[68] René Vidal. Subspace clustering. *IEEE Signal Processing Magazine*, 28(2):52–68, March 2011.

[69] René Vidal, Yi Ma, and Shankar Sastry. Generalized principal component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1945–1959, December 2005. doi: 10.1109/TPAMI.2005.244.

[70] Ureerat Wattanachon and Chidchanok Lursinsap. SPSM: A new hybrid data clustering algorithm for nonlinear data analysis. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(8):1701, 2009. doi: 10.1142/S0218001409007685.

[71] Teng Zhang, Arthur Szlam, Yi Grace Wang, and Gilad M. Lerman. Hybrid linear modeling via local best-fit flats. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1927–1934, 2010.

[72] Teng Zhang, Arthur Szlam, Yi Wang, and Gilad Lerman. Hybrid linear modeling via local best-fit flats. *International Journal of Computer Vision*, 100(3):217–240, December 2012. doi: 10.1007/s11263-012-0535-6.

[73] Zhengyou Zhang. Parameter estimation techniques: a tutorial with application to conic fitting. *Image and Vision Computing*, 15(1):59–76, January 1997. doi: 10.1016/S0262-8856(96)01112-2.

[74] Wei Xing Zheng. A bias correction method for identification of linear dynamic errors-in-variables models. *IEEE Transactions on Automatic Control*, 47(7):1142–1147, July 2002. doi: 10.1109/TAC.2002.800661.

List of publications

Journal articles

[1] Levente Hunyadi and István Vajk. Modeling by fitting a union of polynomial functions to data. *International Journal of Pattern Recognition and Artificial Intelligence*, 27(2), 2013. doi: 10.1142/S0218001413500043. IF (2011): 0.624.

[2] Levente Hunyadi and István Vajk. Reconstructing a model of implicit shapes from an unorganized point set. *Scientific Bulletin of "Politehnica" University of Timişoara, Transactions on Automatic Control and Computer Science (Buletinul Stiintific al Universitatii "Politehnica" din Timişoara, Romania, Seria Automatica si Calculatoare)*, 56 (70):57–64, 2011.

[3] Levente Hunyadi and István Vajk. Identifying dynamic systems with polynomial nonlinearities in the errors-in-variables context. *WSEAS Transactions on Systems*, 8(7): 793–802, July 2009.

[4] Levente Hunyadi and István Vajk. An errors-in-variables parameter estimation method with observation separation. *Scientific Bulletin of "Politehnica" University of Timişoara, Transactions on Automatic Control and Computer Science*, 54(68)(2):93–100, 2009.

[5] Levente Hunyadi and István Vajk. An identification approach to dynamic errors-in-variables systems with a preliminary clustering of observations. *Periodica Polytechnica Electrical Engineering*, 52(3-4):127–135, 2008. doi: 10.3311/pp.ee.2008-3-4.01. URL http://www.pp.bme.hu/ee/2008_3/pdf/ee2008_3_01.pdf.

Publications in conference proceedings

[6] Levente Hunyadi and István Vajk. Fitting a model to noisy data using low-order implicit curves and surfaces. In *Proc. of 2nd Eastern European Regional Conference on the Engineering of Computer Based Systems*, pages 106–114, Bratislava, Slovakia, September 2011. doi: 10.1109/ECBS-EERC.2011.24.

[7] Levente Hunyadi and István Vajk. Identifying unstructured systems in the errors-in-variables context. In *Proc. of 18th World Congress of the International Federation of Automatic Control*, pages 13104–13109, Milano, Italy, August–September 2011. ISBN 978-3-902661-93-7. doi: 10.3182/20110828-6-IT-1002.02390. Paper ThC23.2.

[8] Levente Hunyadi. Fitting implicitly defined shapes to scattered data points with parameters subject to constraints. In István Vajk and Renáta Iváncsy, editors, *Proc. of Automation and Applied Computer Science Workshop*, pages 197–208, Budapest, Hungary, June 2011. ISBN 978-963-313-033-9.

[9] Levente Hunyadi and István Vajk. Identification of linearizable dynamic systems in the errors-in-variables framework. In *Proc. of 1st International Scientific Workshop on Distributed Control Systems*, pages 37–42, Miskolc–Lillafüred, October 2011. ISBN 978-963-661-950-3.

[10] Levente Hunyadi. Fitting implicit curves to scattered data points. In István Vajk and Renáta Iváncsy, editors, *Proc. of Automation and Applied Computer Science Workshop (AACS)*, pages 37–48, Budapest, Hungary, June 2010. ISBN 978-963-313-004-9.

[11] Levente Hunyadi and István Vajk. Implicit model fitting to an unorganized set of points. In *Proc. of IEEE International Joint Conferences on Computational Cybernetics and Technical Informatics (ICCC-CONTI 2010)*, pages 487–491, Timişoara (Temesvár), Romania, May 2010. ISBN 978-1-4244-7431-8. doi: 10.1109/ICCCYB.2010.5491222. paper 85.

[12] Levente Hunyadi and István Vajk. Fitting an implicit model in the errors-in-variables context. In Attila K. Varga and József Vásárhelyi, editors, *Proc. of 11th International Carpathian Control Conference*, pages 359–362, Eger, Hungary, May 2010. ISBN 978-963-06-9289-2.

[13] Levente Hunyadi and István Vajk. Model reconstruction from a point cloud in the errors-in-variables context. In Bikfalvi Péter, editor, *Proc. of microCAD International Scientific Conference*, pages 65–70, Miskolc, Hungary, March 2010. ISBN 978-963-661-919-0.

[14] Levente Hunyadi. Implicit function reconstruction in the errors-in-variables context. In Anikó Szakál, editor, *Proc. of 10th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics*, pages 601–612, Budapest, Hungary, November 2009. ISBN 978-963-7154-96-6.

[15] Levente Hunyadi and István Vajk. An approach to identifying linearizable dynamic errors-in-variables systems. In *Proc. of 10th International PhD Workshop on Systems and Control*, Hluboka nad Vltavou, Czech Republic, September 2009. paper 102.

[16] Levente Hunyadi and István Vajk. Separation methods for dynamic errors-in-variables system identification. In *Proc. of European Control Conference (ECC)*, pages 460–465, Budapest, Hungary, August 2009. ISBN 978-963-311-369-1.

[17] Levente Hunyadi. A cross-validating method for simultaneous model and noise parameter estimation in errors-in-variables systems. In István Vajk, editor, *Proc. of Automation and Applied Computer Science Workshop*, pages 39–52, Budapest, Hungary, June 2009.

[18] Levente Hunyadi. A parameter estimation approach to dynamic errors-in-variables systems with polynomial nonlinearities. In István Vajk, editor, *Proc. of Automation and Applied Computer Science Workshop*, pages 25–38, Budapest, Hungary, June 2009.

[19] Levente Hunyadi and István Vajk. Estimating parameters of dynamic errors-in-variables systems with polynomial nonlinearities. In Metin Demiralp, N. A. Baykara, and N. E. Mastorakis, editors, *Signal Processing Systems, Proc. of 8th WSEAS International Conference on Signal Processing (SIP)*, pages 73–78, Istanbul, Turkey, June 2009.

[20] Levente Hunyadi and István Vajk. Polinomiális jellegű nemlinearitásokból álló dinamikus rendszer paraméterbecslése. In László Lehoczky, editor, *Proc. of microCAD International Scientific Conference*, pages 55–60, Miskolc, Hungary, March 2009. ISBN 978-963-661-879-7.

[21] Levente Hunyadi and István Vajk. Identifying dynamic systems with a nonlinear extension of the generalized Koopmans-Levin method. In László Lehoczky, editor, *Proc. of microCAD International Scientific Conference*, pages 49–54, Miskolc, Hungary, March 2009. ISBN 978-963-661-879-7.

[22] Levente Hunyadi. Separation techniques for errors-in-variables parameter estimation. In Anikó Szakál, editor, *Proc. of 9th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics*, pages 467–478, Budapest, Hungary, November 2008. ISBN 978-963-7154-82-9.

[23] Levente Hunyadi. An identification approach to dynamic errors-in-variables systems with a preliminary clustering of observations. In *Proc. of 6th Conference of PhD Students in Computer Science (CSCS)*, Szeged, Hungary, July 2008. Awarded “Best talk” in session Operation Research.

[24] Levente Hunyadi. Methods for simultaneous estimation of process and noise parameters in errors-in-variables systems. In István Vajk, editor, *Proc. of Automation and Applied Computer Science Workshop*, pages 115–126, Budapest, Hungary, June 2008. ISBN 978-963-420-955-3.

[25] Levente Hunyadi. A framework for comparing errors-in-variables estimation algorithms. In István Vajk, editor, *Proc. of Automation and Applied Computer Science Workshop*, pages 127–138, Budapest, Hungary, June 2008. ISBN 978-963-420-955-3.

[26] Levente Hunyadi and István Vajk. Identification of errors-in-variables systems using data clustering. In Gregor Rozinaj, Jozef Čepko, Peter Trúchly, Ján Vrabec, and Juraj Vojtko, editors, *Proc. of the 15th International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 197–200, Bratislava, Slovakia, June 2008. ISBN 978-80-227-2856-0. doi: 10.1109/IWSSIP.2008.4604401.

[27] Levente Hunyadi and István Vajk. Parameter estimation for errors-in-variables systems using partitioned input data. In Octavian Prostean, Gheorghe-Daniel Andreescu, and Dan Pescaru, editors, *Proc. of the 8th International Conference on Technical Informatics (CONTI)*, pages 47–52, Timişoara (Temesvár), Romania, June 2008.

[28] Levente Hunyadi. Identification methods for dynamic errors-in-variables systems. In László Lehoczky, editor, *Proc. of microCAD International Scientific Conference*, pages 57–62, Miskolc, Hungary, March 2008. ISBN 978-963-661-824-7.

APPENDIX

A Projection to quadratic curves and surfaces

These sections present the full derivation of fast and reliable projection algorithms for ellipses [23], hyperbolas [14] and parabolas [14] in two dimensions and several types of quadrics [14] in three dimensions.

A.1 Projection to an ellipse

Without loss of generality, we can assume the ellipse is in its canonical form, i.e. it is axis-aligned and centered at the origin. If not, a transformation matrix $\mathbf{M}$ that axis-aligns and centers the ellipse can be applied to the data points, and the inverse transformation matrix $\mathbf{M}^{-1}$ to the computed foot points. Thus, let the ellipse be defined as

$$Q(\mathbf{x}) = \frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} - 1 = 0. \tag{A.1}$$

Due to symmetry, it is enough to work in the first quadrant (i.e. for both data point coordinates we have $w_1 > 0$ and $w_2 > 0$), in which case the projection point will also be in the first quadrant (i.e. $x_1 > 0$ and $x_2 > 0$). Other points can be reflected to the first quadrant about the axes, and then the projection point can be reflected back.

For the distance between a data point $\mathbf{w} = \begin{bmatrix} w_1 & w_2 \end{bmatrix}$ and a foot point $\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}$ to be minimum, the distance vector must be normal to the ellipse, which means that the ellipse gradient vector

$$\frac{1}{2}\partial_{\mathbf{x}} Q(\mathbf{x}) = \begin{bmatrix} \frac{x_1}{a^2} & \frac{x_2}{b^2} \end{bmatrix}$$

and the distance vector should be equal up to magnitude:

$$\begin{aligned} w_1 - x_1 &= t\partial_{x_1} Q(\mathbf{x}) \\ w_2 - x_2 &= t\partial_{x_2} Q(\mathbf{x}) \end{aligned} \tag{A.2}$$

The above equations give

$$\begin{aligned} w_1 - x_1 &= t\frac{x_1}{a^2} \\ w_2 - x_2 &= t\frac{x_2}{b^2} \end{aligned}$$

where t is a scalar where $t < 0$ for the inside and $t > 0$ for the outside of the ellipse. This implies (after rearrangements) that

$$\begin{aligned} x_1 &= \frac{a^2 w_1}{t + a^2} \\ x_2 &= \frac{b^2 w_2}{t + b^2} \end{aligned} \tag{A.3}$$

Substituting (A.3) into (A.1) yields

$$\begin{aligned} Q(t) &= \frac{1}{a^2}\left(\frac{a^2 w_1}{t + a^2}\right)^2 + \frac{1}{b^2}\left(\frac{b^2 w_2}{t + b^2}\right)^2 - 1 \\ &= \left(\frac{a w_1}{t + a^2}\right)^2 + \left(\frac{b w_2}{t + b^2}\right)^2 - 1. \end{aligned}$$

which we need to solve for

$$Q(t) = 0.$$

Differentiating w.r.t. t we get the first and second derivatives

$$\begin{aligned} \frac{d}{dt}Q(t) &= -\frac{2a^2 w_1^2}{\left(t + a^2\right)^3} - \frac{2b^2 w_2^2}{\left(t + b^2\right)^3} \\ \frac{d}{dt^2}Q(t) &= \frac{6a^2 w_1^2}{\left(t + a^2\right)^4} + \frac{6b^2 w_2^2}{\left(t + b^2\right)^4}. \end{aligned}$$

Since we operate in the first quadrant, we have the constraints $t > -a^2$ and $t > -b^2$. Assuming, as usual, that $a > b$ we get a single constraint $t > -b^2$. Thus, on the domain $(-b^2, \infty)$ of interest, we have

$$\begin{aligned} \frac{d}{dt}Q(t) &< 0 \\ \frac{d}{dt^2}Q(t) &> 0 \end{aligned}$$

and

$$\begin{aligned} \lim_{t \to -b^2} Q(t) &= \infty \\ \lim_{t \to \infty} Q(t) &= -1 \end{aligned}$$

making the function $Q(t)$ strictly monotonic decreasing and concave, therefore a unique root t of $Q(t)$ must exist (Figure A.1).

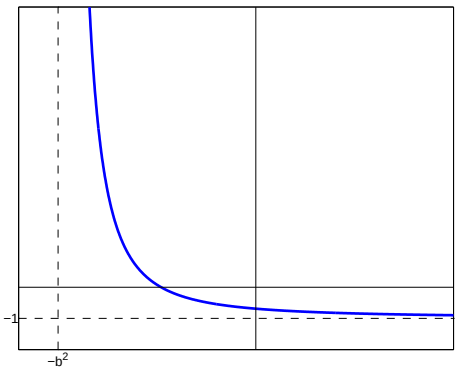


Figure A.1: Projection function $Q(t)$ for an ellipse.

One way to find this root is using Newton's method, which may be initialized [23] with

$$t_0 = bw_2 - b^2$$

or for faster convergence [14] with

$$t_0 = \max\left(aw_1 - a^2,\, bw_2 - b^2\right).$$

A.2 Projection to a hyperbola

As in the case of the ellipse, we assume the hyperbola is given in its canonical form

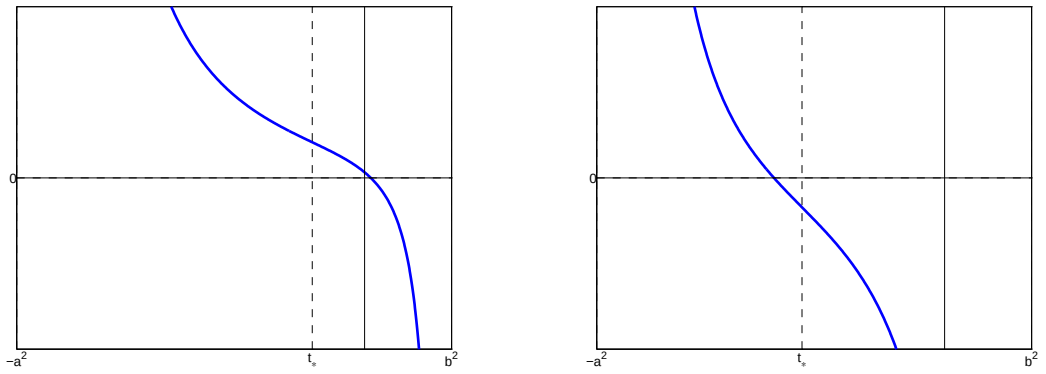
$$Q(\mathbf{x}) = \frac{x_1^2}{a^2} - \frac{x_2^2}{b^2} - 1 = 0 \tag{A.4}$$

where, from symmetry, we may restrict our investigations to the first quadrant (i.e. for both data point coordinates we have $w_1 > 0$ and $w_2 > 0$, and likewise for both projection point coordinates $x_1 > 0$ and $x_2 > 0$). Orthogonality conditions (A.2), i.e. the hyperbola gradient vector and the distance vector should be equal up to magnitude, yield

$$\begin{aligned} w_1 - x_1 &= t\frac{x_1}{a^2} \\ w_2 - x_2 &= -t\frac{x_2}{b^2} \end{aligned}$$

from which we have

$$\begin{aligned} x_1 &= \frac{a^2 w_1}{t + a^2} \\ x_2 &= \frac{b^2 w_2}{-t + b^2}. \end{aligned} \tag{A.5}$$



(a) **(b)**

Figure A.2: Projection function $Q(t)$ for a hyperbola.

Substituting (A.5) into (A.4) we obtain the quadratic function

$$\begin{aligned} Q(t) &= \frac{1}{a^2}\left(\frac{a^2 w_1}{t+a^2}\right)^2 - \frac{1}{b^2}\left(\frac{b^2 w_2}{-t+b^2}\right)^2 - 1 \\ &= \left(\frac{a w_1}{t+a^2}\right)^2 - \left(\frac{b w_2}{-t+b^2}\right)^2 - 1 \end{aligned}$$

where we seek $Q(t) = 0$. Again, we can compute the derivatives

$$\begin{aligned} \frac{d}{dt}Q(t) &= -\frac{2a^2 w_1^2}{\left(t+a^2\right)^3} - \frac{2b^2 w_2^2}{\left(-t+b^2\right)^3} \\ \frac{d}{dt^2}Q(t) &= \frac{6a^2 w_1^2}{\left(t+a^2\right)^4} - \frac{6b^2 w_2^2}{\left(-t+b^2\right)^4} \end{aligned}$$

and we note that our confinement to the first quadrant imposes the restriction $-a^2 < t < b^2$ with

$$\begin{aligned} \lim_{t\to -a^2} Q(t) &= \infty \\ \lim_{t\to b^2} Q(t) &= -\infty. \end{aligned}$$

From the above, we see that $\frac{d}{dt}Q(t) < 0$ on the domain of interest and $\frac{d}{dt^2}Q(t)$ decreases from $+\infty$ to $-\infty$ and is monotonic. Therefore, $Q(t)$ has a unique inflection point $t_\star$ within the domain $\left(-a^2, b^2\right)$, which we may calculate by equating the second derivative with zero, which yields

$$t_\star = \frac{b^2\sqrt{a w_1} - a^2\sqrt{b w_2}}{\sqrt{a w_1} + \sqrt{b w_2}}.$$

The inflection point may either lie above the coordinate x axis, or may lie below (Figure A.2), and we must choose the initial location t_0 carefully to ensure convergence for Newton's method. When the inflection point lies above the coordinate x axis (Figure A.2a), we

need $Q(t_0) < 0$, in which case we may successively try values

$$t_k = b^2 - \frac{b^2 - t_\star}{2^k}$$

for $k = 1, 2, \ldots$ until we find a $Q(t_k) < 0$. When the inflection point lies below the x axis (Figure A.2b), we may successively try values

$$t_k = -a^2 + \frac{t_\star + a^2}{2^k}$$

for $k = 1, 2, \ldots$ until we find a $Q(t_k) > 0$.

A.3 Projection to a parabola

Consider the canonical form of the parabola

$$x_2^2 - 2px_1 = 0 \tag{A.6}$$

where $p > 0$ is the distance from the focus to the directrix. Due to symmetry, we may restrict the method to $w_2 > 0$ when we also have $x_2 > 0$. The orthogonality conditions (A.2) now give

$$\begin{aligned} w_1 - x_1 &= -pt \\ w_2 - x_2 &= yt \end{aligned}$$

from which we have

$$\begin{aligned} x_1 &= w_1 + pt \\ x_2 &= \frac{w_2}{t+1}. \end{aligned} \tag{A.7}$$

Since $x_1 > 0$, we have constraint $t > -1$. Substituting (A.7) into (A.6) we obtain

$$Q(t) = \frac{w_2^2}{(t+1)^2} - 2pw_1 - 2p^2 t$$

whose root we need to find. Taking the derivatives,

$$\begin{aligned} \frac{d}{dt}Q(t) &= -\frac{2w_2^2}{(t+1)^3} - 2p^2 \\ \frac{d}{dt^2}Q(t) &= \frac{6w_2^2}{(t+1)^4} \end{aligned}$$

and our restriction of the domain of interest imposes the restriction $t > -1$ with

$$\begin{aligned} \lim_{t\to -1} Q(t) &= \infty \\ \lim_{t\to \infty} Q(t) &= -\infty. \end{aligned}$$

Since we have $\frac{d}{dt}Q(t) < 0$ and $\frac{d}{dt^2}Q(t) > 0$ on $(-1, \infty)$, the function $Q(t)$ is monotonically decreasing and concave (Figure A.3). Standard Newton's method starting at any point t_0 where $Q(t_0) > 0$ will converge to the unique root of $Q(t)$. We can try points $t_k = -1 + 2^{-k}$ for $k = 1, 2, \ldots$ until we find a $Q(t_k) > 0$.

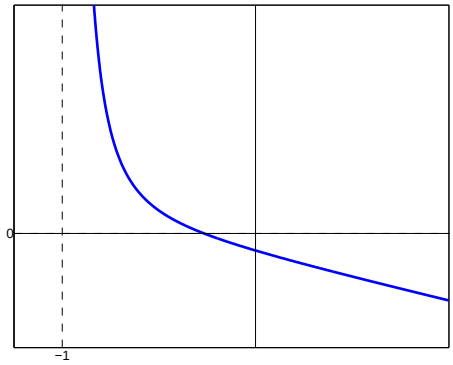


Figure A.3: Projection function $Q(t)$ for a parabola.

A.4 Projection to an ellipsoid

With canonical coordinates, an ellipsoid is defined as

$$Q(\mathbf{x}) = \frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} - 1 = 0 \tag{A.8}$$

where for the semi-axes we have $a \geqq b \geqq c > 0$. Due to symmetry, our investigation may be restricted to $w_1 > 0$, $w_2 > 0$ and $w_3 > 0$ for data points, and likewise $x_1 > 0$, $x_2 > 0$ and $x_3 > 0$ for projection points. The orthogonality conditions give

$$\begin{aligned} w_1 - x_1 &= t\frac{x_1}{a^2} \\ w_2 - x_2 &= t\frac{x_2}{b^2} \\ w_3 - x_3 &= t\frac{x_3}{c^2} \end{aligned}$$

from which we have

$$\begin{aligned} x_1 &= \frac{a^2 w_1}{t + a^2} \\ x_2 &= \frac{b^2 w_2}{t + b^2} \\ x_3 &= \frac{c^2 w_3}{t + c^2}. \end{aligned} \tag{A.9}$$

Since we are confined to the first quadrant, $t > \max\left(-a^2, -b^2, -c^2\right) = -c^2$. Substituting (A.9) into (A.8) we get

$$Q(t) = \left(\frac{a w_1}{t + a^2}\right)^2 + \left(\frac{b w_2}{t + b^2}\right)^2 + \left(\frac{c w_3}{t + c^2}\right)^2 - 1$$

for which we seek $Q(t) = 0$. Calculating the derivatives,

$$\begin{aligned}\frac{d}{dt}Q(t) &= -\frac{2a^2w_1^2}{\left(t+a^2\right)^3} - \frac{2b^2w_2^2}{\left(t+b^2\right)^3} - \frac{2c^2w_3^2}{\left(c^2+t\right)^3} \\ \frac{d}{dt^2}Q(t) &= \frac{6a^2w_1^2}{\left(t+a^2\right)^4} + \frac{6b^2w_2^2}{\left(t+b^2\right)^4} + \frac{6c^2w_3^2}{\left(t+c^2\right)^4}\end{aligned}$$

and inspecting behavior in the domain of interest $\left(-c^2, \infty\right)$, we find that the function $Q(t)$ is monotonically decreasing and concave. Starting Newton's method at any point t_0 where $Q(t_0) > 0$, specifically

$$t_0 = \max\left(aw_1 - a^2,\, bw_2 - b^2,\, cw_3 - c^2\right)$$

will converge to the unique solution.

A.5 Projection to an elliptic paraboloid

Canonical coordinates for an elliptic paraboloid

$$Q(\mathbf{x}) = \frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} - x_3 = 0 \tag{A.10}$$

and a restriction to $w_1 > 0$ and $w_2 > 0$, when we also have $x_1 > 0$ and $x_2 > 0$, produces from the orthogonality conditions (A.2) the set of equations

$$\begin{aligned}w_1 - x_1 &= t\frac{x_1}{a^2} \\ w_2 - x_2 &= t\frac{x_2}{b^2} \\ w_3 - x_3 &= -\frac{t}{2}\end{aligned}$$

from which we get

$$\begin{aligned}x_1 &= \frac{a^2w_1}{t+a^2} \\ x_2 &= \frac{b^2w_2}{t+b^2} \\ x_3 &= w_3 + \frac{t}{2}\end{aligned} \tag{A.11}$$

where from our assumptions $x_1 > 0$ and $x_2 > 0$ we have the constraint $t > \max\left(-a^2, -b^2\right) = -b^2$. Substituting (A.11) into (A.10) we obtain the function

$$Q(t) = \frac{a^2w_1^2}{\left(t+a^2\right)^2} + \frac{b^2w_2^2}{\left(t+b^2\right)^2} - w_3 - \frac{t}{2}$$

whose root we seek. Calculating the derivatives

$$\begin{aligned}\frac{d}{dt}Q(t) &= -\frac{2a^2w_1^2}{\left(a^2+t\right)^3}-\frac{2b^2w_2^2}{\left(b^2+t\right)^3}-\frac{1}{2}\\ \frac{d}{dt^2}Q(t) &= \frac{6a^2w_1^2}{\left(a^2+t\right)^4}+\frac{6b^2w_2^2}{\left(b^2+t\right)^4}\end{aligned}$$

and inspecting the behavior in the domain $\left(-b^2,\infty\right)$, we see that $\frac{d}{dt}Q(t)<0$ and $\frac{d}{dt^2}Q(t)>0$, making the function $Q(t)$ monotonically decreasing and concave, with a graph similar to that shown in Figure A.3. Starting Newton's method at any point t_0 where $Q(t_0)>0$ will converge to the unique solution.

A.6 Projection to a hyperbolic paraboloid

Canonical coordinates for a hyperbolic paraboloid (saddle surface)

$$Q(\mathbf{x})=\frac{x_1^2}{a^2}-\frac{x_2^2}{b^2}-x_3=0 \tag{A.12}$$

and a restriction to $w_1>0$ and $w_2>0$, when we also have $x_1>0$ and $x_2>0$, produces from the orthogonality conditions (A.2) the set of equations

$$\begin{aligned}w_1-x_1 &= t\frac{x_1}{a^2}\\ w_2-x_2 &= -t\frac{x_2}{b^2}\\ w_3-x_3 &= -\frac{t}{2}\end{aligned}$$

from which we get

$$\begin{aligned}x_1 &= \frac{a^2w_1}{t+a^2}\\ x_2 &= \frac{b^2w_2}{-t+b^2}\\ x_3 &= w_3+\frac{t}{2}\end{aligned} \tag{A.13}$$

where from our assumptions $x_1>0$ and $x_2>0$ we have the constraints $-a^2<t<b^2$. Substituting (A.13) into (A.12) we obtain the function

$$Q(t)=\frac{a^2w_1^2}{\left(t+a^2\right)^2}-\frac{b^2w_2^2}{\left(-t+b^2\right)^2}-w_3-\frac{t}{2}$$

whose root we seek. Calculating the derivatives

$$\begin{aligned}\frac{d}{dt}Q(t) &= -\frac{2a^2w_1^2}{\left(t+a^2\right)^3}-\frac{2b^2w_2^2}{\left(-t+b^2\right)^3}-\frac{1}{2}\\ \frac{d}{dt^2}Q(t) &= \frac{6a^2w_1^2}{\left(t+a^2\right)^4}-\frac{6b^2w_2^2}{\left(t-b^2\right)^4}\end{aligned}$$

and inspecting the behavior in the domain $\left(-a^2, b^2\right)$, we see that $\frac{d}{dt}Q(t) < 0$ and $\frac{d}{dt^2}Q(t)$ decreases from $+\infty$ (near $-a^2$) to $-\infty$ (near b^2), and is monotonic, making $Q(t)$ have a single inflection point. A similar reasoning as used for hyperbolas in Section A.2 yields the unique solution.

A.7 Projection to a hyperboloid of one sheet

Finally, let us project a point $(w_1; w_2; w_3)$ onto a hyperboloid of one sheet defined in its canonical coordinates as

$$Q(\mathbf{x}) = \frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} - \frac{x_3^2}{c^2} - 1 = 0 \tag{A.14}$$

where we can assume $a \geq b$. Due to symmetry, the method is restricted to $w_1 > 0$, $w_2 > 0$ and $w_3 > 0$ when we also have $x_1 > 0$, $x_2 > 0$ and $x_3 > 0$. The orthogonality conditions (A.2) now give

$$\begin{aligned} w_1 - x_1 &= t\frac{x_1}{a^2} \\ w_2 - x_2 &= t\frac{x_2}{b^2} \\ w_3 - x_3 &= -t\frac{x_3}{c^2} \end{aligned}$$

for some scalar t from which

$$\begin{aligned} x_1 &= \frac{a^2 w_1}{t + a^2} \\ x_2 &= \frac{b^2 w_2}{t + b^2} \\ x_3 &= \frac{c^2 w_3}{-t + c^2}. \end{aligned} \tag{A.15}$$

Since $x_1 > 0$, $x_2 > 0$ and $x_3 > 0$, we have constraints $-b^2 < t < c^2$. Substituting (A.15) into (A.14) we obtain a function and its derivatives

$$\begin{aligned} Q(t) &= \frac{a^2 w_1^2}{(t + a^2)^2} + \frac{b^2 w_2^2}{(t + b^2)^2} - \frac{c^2 w_3^2}{(-t + c^2)^2} - 1 \\ \frac{d}{dt}Q(t) &= -\frac{2a^2 w_1^2}{\left(a^2 + t\right)^3} - \frac{2b^2 w_2^2}{\left(b^2 + t\right)^3} + \frac{2c^2 w_3^2}{\left(t - c^2\right)^3} \\ \frac{d}{dt^2}Q(t) &= \frac{6a^2 w_1^2}{\left(a^2 + t\right)^4} + \frac{6b^2 w_2^2}{\left(b^2 + t\right)^4} - \frac{6c^2 w_3^2}{\left(t - c^2\right)^4} \end{aligned}$$

whose root we need to find.

Again, as before, $\frac{d}{dt^2}Q(t)$ decreases from $+\infty$ (near $-b^2$) to $-\infty$ (near c^2), and it is monotonic because $\frac{d}{dt^3}Q(t) < 0$, as one can easily verify. Thus, $Q(t)$ has a unique inflection point $t_\star$, within the interval $\left(-b^2, c^2\right)$, its graph looks like one of those shown in Figure (A.2). Unfortunately, it is not easy to determine whether the inflection point $t_\star > 0$ (Figure A.2a) or

$t_\star < 0$ (Figure A.2b) because we cannot solve the equation $\frac{d}{dt^2}Q(t) = 0$. However, one of the two iterative procedures described in the case of hyperbolas, i.e. Newton's method starting from the left or from the right, must work. Thus, we simply can choose one of the two procedures at random hoping that it converges. If it fails, i.e. if an iteration lands outside the interval $\left(-b^2, c^2\right)$, then we can switch to the other procedure, and it will surely converge. However, even if we start on the wrong side, Newton's iteration may land on the right side and then subsequently converge.