Rational Approximation of Discrete Data with Asymptotic Behaviour

Philip Cooper

A thesis submitted to the University of Huddersfield
in partial fulfilment of the requirements for
the degree of Doctor of Philosophy

The University of Huddersfield in collaboration with the
National Physical Laboratory.

June 2007

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This thesis is concerned with the least-squares approximation of discrete data that appear to exhibit asymptotic behaviour. In particular, we consider using rational functions as they are able to display a number of types of asymptotic behaviour. The research is biased towards the development of simple and easily implemented algorithms that can be used for this purpose. We discuss a number of novel approximation forms, including the Semi-Infinite Rational Spline and the Asymptotic Polynomial. The Semi-Infinite Rational Spline is a piecewise rational function, continuous across a single knot, and may be defined to have different asymptotic limits at $\pm\infty$. The continuity constraints at the knot are implicit in the function definition, and it can be fitted to data without the use of constrained optimisation algorithms. The Asymptotic Polynomial is a linear combination of weighted basis functions, orthogonalised with respect to a rational weight function of nonlinear approximation parameters. We discuss an efficient and numerically stable implementation of the Gauss-Newton method that can be used to fit this function to discrete data. A number of extensions of the Loeb algorithm are discussed, including a simple modification for fitting Semi-Infinite Rational Splines, and a new hybrid algorithm that is a combination of the Loeb algorithm and the Lawson algorithm (including its Rice and Usow extension), for fitting ℓ_p rational approximations. In addition, we present an extension of the Rice and Usow algorithm to include ℓ_p approximation for values $p < 2$. Also discussed is an alternative representation of a polynomial ratio denominator, that allows pole free approximations to be fitted to data with the use of unconstrained optimisation methods. In all cases we present a large number of numerical applications of these methods to illustrate their usefulness.

Chapter 1

INTRODUCTION

The main focus of this thesis is the fitting of least-squares approximations to discrete data using rational functions. In this chapter we introduce some of the methods and notation that will be used throughout the thesis and outline the industrial requirements that have motivated this study. We start with a general introduction to the concept of linear approximation and describe common methods used for fitting linear functions to data. We then go on to define generalised rational functions, and describe some of the properties that make them particularly suitable for certain approximation problems. Such problems include the approximation of data or functions that contain singularities, or exhibit asymptotic behaviour such as exponential decay. We then highlight some of the disadvantages and difficulties associated with the use of rational functions for data fitting. Finally we discuss the aims of this project and why it is important to the area of metrology, and include a brief overview of some real world data fitting problems that are particularly well suited for approximation by rational functions.

1.1 Metrology

Metrology is the scientific study of measurement, and the development of methods for modelling measurement data is an active area of research in this field. A common problem that arises in metrology is the need to model the functional relationship between two variables x and f, that is represented by a set of m discrete pairs of data points $\{(x_i, f_i)\}_{i=1}^{m} \subset \mathbb{R}^2$, that have been obtained by experimental measurements.

A practical example of such a situation is the modelling of the relationship between temperature and pressure of a gas contained in a fixed volume. In this example, each of the x_i values represent a specified temperature at which a measurement of the pressure is made and recorded as a value f_i. The x_i values are considered as fixed, exact values, while the f_i are treated as inexact values due to the presence of some measurement error. In practice, there are also measurement errors in the x_i values, but for many approximation problems, including those presented in this thesis, we regard the x_i values as fixed and without error. The general situation in Metrology that we consider are problems where the x_i are known values at which the experimenter is able to record a measurement of the quantity of interest f_i. Many of the datasets that we use within this project were obtained in this manner.

The contamination of the f_i values can be explained by a number of physical reasons, such as imperfections in the measurement instrument, poor calibration, or mechanical defects.

Given the data, we are then faced with the problem of trying to find a function that is able to approximate it well. We can represent the noise in the data mathematically as

$$f_i = h(x_i) + \epsilon_i \tag{1.1}$$

where $h(x)$ represents an unknown function that explains the true functional relationship underlying the data, and ϵ_i is a component of measurement error associated with the measurement f_i. We treat the function $h(x)$ as a function that describes the data in the absence of any error, and it is this function that we then try and approximate. For any particular problem there may be experimental or theoretical knowledge that will assist in the approximation process, or at least suggest an appropriate choice of function to approximate with. For example, it would be more appropriate to fit experimental data exhibiting exponential decay with a gaussian function rather than a polynomial, or to fit periodic data using trigonometric polynomials rather than standard polynomials.

It may also be that the statistical distribution of the measurement errors ϵ_i is known, which is useful knowledge and affects the choice of method used to approximate the data as we will explain in more detail later.

1.2 Linear approximation of discrete data

As described previously, we consider the problem of finding a function that approximates a discrete set of data points $\{(x_i, f_i)\}_{i=1}^m \subset \mathbb{R}^2$. The function chosen to approximate the data is denoted by $F(x)$ and is called an *approximation form*, and is usually chosen from a particular class of functions. Although the functional form is considered to be fixed, $F(x)$ will have a large degree of flexibility due to its dependence on a number of variable coefficients called *approximation parameters*. The value of these parameters are initially unknown and need to be evaluated in such a way that results in $F(x)$ approximating the data as well as possible. The function $F(x)$ is called a *linear* approximation if it can be expressed in the form

$$F(\mathbf{a}, x) = \sum_{j=1}^{n} a_j \phi_j(x), \tag{1.2}$$

where $\{\phi_j(x)\}_{j=1}^n$ is a set of specified linearly independent basis functions and $\mathbf{a} = (a_1, \ldots, a_n)^T \in \mathbb{R}^n$ is the vector whose elements are the approximation parameters. It is usually the case that $n \leq m$ which results in a larger number of data points than unknown approximation parameters. Commonly used basis functions are orthogonal polynomials, splines, radial basis functions [48], and trigonometric functions, as these choices result in a linear form $F(\mathbf{a}, x)$ that is simple to evaluate, differentiate, and integrate. Linear approximation forms (1.2) are widely used in data fitting as they are linear with respect to their approximation parameters, and as a consequence these parameters are often easy to evaluate.

Once an approximation form $F(\mathbf{a}, x)$ has been chosen, it is necessary to find a method of calculating the parameters $\mathbf{a}$ that will yield a good approximation to the data. In order to achieve this, a means of assessing the quality of an approximation is needed, which will enable the distinction between good and poor approximations to be made. At any individual point x_i we define the *approximation error*, or *residual* as

$$e_i = f_i - F(\mathbf{a}, x_i), \tag{1.3}$$

and define the error or residual vector $\mathbf{e}$ as

$$\mathbf{e} = (e_1, \ldots, e_m)^T. \tag{1.4}$$

The smaller the value of the residuals, the better the approximation is, and so we try and find values of $\mathbf{a}$ that will minimise the error vector in some way.

1.2.1 Norms

A *semi-norm* is a function $g : \mathbb{R}^n \to \mathbb{R}$ that satisfies the following properties

1. $g(\mathbf{c}) \geq 0.$

2. $g(\lambda\mathbf{c}) = |\lambda| g(\mathbf{c}),$ for all $\lambda \in \mathbb{R}$.

3. $g(\mathbf{c} + \mathbf{d}) \leq g(\mathbf{c}) + g(\mathbf{d}).$

A *norm* is a function g that qualifies as a semi-norm and also satisfies the property

$$g(\mathbf{c}) = 0 \Leftrightarrow \mathbf{c} = \mathbf{0}.$$

The norm of a vector $\mathbf{c}$ is commonly represented by $\|\mathbf{c}\|$, and we will use this notation throughout the thesis. Norms are multi-dimensional abstractions of the absolute value function, and provide a means with which to measure the size of error vectors and hence the quality of approximations.

For discrete approximation problems, the most common choices of norm are the ℓ_p and ℓ_∞ norms, which are defined as

$$\|\mathbf{e}\|_p = \left[\sum_{i=1}^{m} |e_i|^p\right]^{\frac{1}{p}}, \text{ for } 1 \leq p < \infty, \tag{1.5}$$

and

$$\|\mathbf{e}\|_\infty = \max_i |e_i|, \qquad i = 1, \ldots, m \tag{1.6}$$

respectively.

1.3 Best approximations

If possible, it would be desirable to find an approximation that will minimise the error norm over every possible choice of the approximation parameters. If there exists a parameter vector $\mathbf{a}^* \in \mathbb{R}^n$ such that

$$\|\mathbf{f} - \mathbf{F}(\mathbf{a}^*, \mathbf{x})\| \leq \|\mathbf{f} - \mathbf{F}(\mathbf{a}, \mathbf{x})\| \qquad \forall \mathbf{a} \in \mathbb{R}^n, \tag{1.7}$$

where

$$\mathbf{f} = (f_1, \ldots, f_m)^T, \tag{1.8}$$

$$\mathbf{x} = (x_1, \ldots, x_m)^T, \tag{1.9}$$

$$\mathbf{F}(\mathbf{a}, \mathbf{x}) = (F(\mathbf{a}, x_1), \ldots, F(\mathbf{a}, x_m))^T, \tag{1.10}$$

then we refer to $F(\mathbf{a}^*, \mathbf{x})$ as a *best approximation* to $\mathbf{f}$. Provided that the basis functions are linearly independent, a best approximation will always exist whatever the norm [46], but uniqueness of a best approximation will depend on the particular norm that is being used, and on the choice of basis functions used to define $F(\mathbf{a}, x)$. Uniqueness has been proved for best ℓ_2 and ℓ_∞ (provided the basis functions for a Chebyshev set) approximations [11, 46], but the best linear ℓ_1 approximation is not necessarily unique. For some practical problems it may not be possible or even necessary to find a best approximation (although one will always exist), and often an approximation that represents data well and has a small error norm is deemed satisfactory.

1.4 ℓ_1 *approximation*

ℓ_1 approximations are usually fitted to data that are known to be largely accurate, but contain a small number of wild points or outliers. An ℓ_1 approximation has a tendency to ignore these outliers, resulting in small error values at the accurate data points and large errors at the outliers. This property is clearly illustrated by the ℓ_1 approximation in Figure 1.1. For given abscissae and data vectors $\mathbf{x} \in \mathbb{R}^m$ and $\mathbf{f} \in \mathbb{R}^m$ (where $f_i = f(x_i)$ and $i = 1, \ldots, m$), we now state without proof a characterization theorem for a best ℓ_1 polynomial approximation.

Theorem 1 (ℓ_1 Characterization Theorem) *Let $F(\mathbf{a}^*, x)$ be an arbitrary polynomial approximation of degree n. We define the set $A = \{x_i; F(\mathbf{a}^*, x_i) = f_i\}$, and define $s(x)$ as the sign function*

$$s^*(x_i) = \begin{cases} 1, & f_i > F(\mathbf{a}^*, x_i) \\ 0, & f_i = F(\mathbf{a}^*, x_i) \\ -1, & f_i < F(\mathbf{a}^*, x_i). \end{cases}$$

The element $F(\mathbf{a}^*, x)$ is a best ℓ_1 approximation to $\mathbf{f}$ if and only if

$$\left| \sum_{i=1}^{m} s^*(x_i) F(\mathbf{a}^*, x_i) \right| \leq \sum_{x_i \in A} |F(\mathbf{a}, x_i)|,$$

for all $\mathbf{a} \in \mathbb{R}^{n+1}$.

This theorem is taken from [46] (from which a proof can also be found), and has been modified to accomodate the notation used in this chapter. An additional characteristic of a best polynomial ℓ_1 approximation is that it will interpolate the data at at least $n + 1$ points, where n is the degree of the approximating polynomial. There are a number of methods of fitting a best ℓ_1 approximation, including the widely used algorithm of Barrodale and Roberts [9], which is the algorithm that is used in this thesis for fitting linear ℓ_1 approximations.

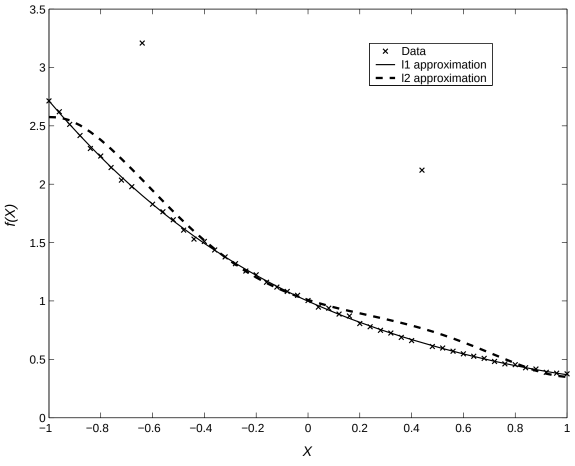

Figure 1.1: Comparison of ℓ_1 and ℓ_2 polynomial approximations of data containing outliers.

1.5 Chebyshev approximation

For a given approximation problem where it is known that the data is accurate, contains no outliers, and that the error in the data is uniformly distributed, it is appropriate to fit the data by minimizing the ℓ_∞ norm of the residual vector. The resulting approximations obtained in this way are commonly referred to as a *Chebyshev*, *minimax*, or *uniform* approximations. The existence and uniqueness of best Chebyshev approximations can be proved for the case when the approximation form is linear [11]. In addition, the best linear Chebyshev approximation has the following characterization theorem, a proof of which can be found in [46].

Theorem 2 (Chebyshev Characterization Theorem) *Let $F(\mathbf{a}^*, x)$ be an element from a linear approximation space of degree n, spanned by basis functions $\{\phi_j(x)\}_{j=1}^n$ that form a Chebyshev Set. $F(\mathbf{a}^*, x)$ is a best Chebyshev approximation to $\mathbf{f}$ if and only if there exists $n+1$ points $\{\gamma_1, \ldots, \gamma_{n+1}\} \subset \mathbf{f}$ with $\gamma_i < \gamma_{i+1}$ that satisfy*

1. $|F(\mathbf{a}^*, \gamma_i) - \gamma_i| = \|F(\mathbf{a}^*, \mathbf{x}) - \mathbf{f}\|_\infty \qquad i = 1, \ldots, n+1.$

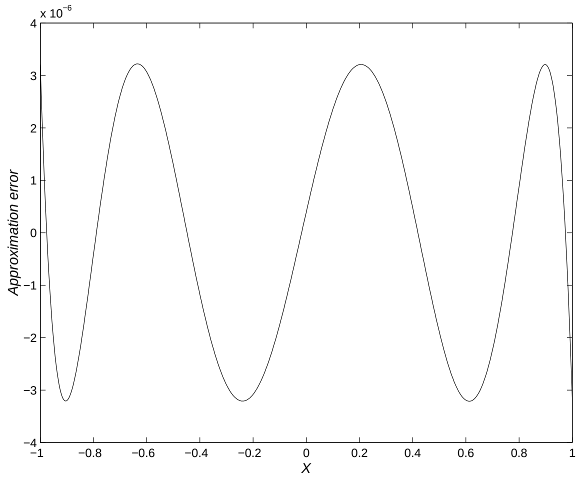

Figure 1.2: Equioscillation property of a linear ℓ_∞ approximation error.

2. $F(\mathbf{a}^*, \gamma_i) - \gamma_i = -(F(\mathbf{a}^*, \gamma_{i+1}) - \gamma_{i+1}) \qquad i = 1, \ldots, n.$

This alternation property exhibited by the Chebyshev error function is sometimes referred to as the *equioscillation* property and is illustrated in figure 1.2. There are a number of methods of fitting Chebyshev approximations to data, of which the most widely used is probably the algorithm of Barrodale and Phillips [8]. This is the algorithm of choice for any linear Chebyshev approximation problems that are presented in this thesis (unless explicitly stated otherwise).

1.6 Least-squares approximation

A *least-squares* approximation refers to an approximation that has been fitted with respect to the ℓ_2 norm (also known as the *Euclidean* or *least-squares* norm). Least-squares approximation is the primary focus of this thesis, and for this reason it is discussed in greater detail than approximation with respect to other norms. Although we are mainly concerned with nonlinear approximations, we will begin with a discussion of linear least-squares problems and solution methods first, as many nonlinear solution methods require solving a set of linearised problems as part of an iterative process.

For data that are subject to uncorrelated errors that are believed to follow a Normal distribution with mean zero and constant variance, it is appropriate to fit a least-squares approximation. Under these assumptions on the error distribution, it can be proved (for the case of linear least-squares approximation) that the maximum likelihood estimates for the approximation parameters are identical to those obtained using the least-squares method. The least-squares norm of the error vector (1.4) is defined as

$$\|\mathbf{e}\|_2 = \left[\sum_{i=1}^{m}(f_i - F(\mathbf{a}, x_i))^2\right]^{\frac{1}{2}}, \tag{1.11}$$

and is the quantity we wish to minimise over all possible approximation parameters $\mathbf{a}$. The function (1.11) has the same minimum as its square, which is easier to work with, and we express this in matrix form as

$$E(\mathbf{a}) = \|\mathbf{f} - C\mathbf{a}\|_2^2 = (\mathbf{f} - C\mathbf{a})^T(\mathbf{f} - C\mathbf{a}), \tag{1.12}$$

where C is the $m \times n$ *observation matrix* which is defined as the matrix having i, jth element $C_{ij} = \phi_j(x_i)$. This is a quadratic function of the elements of the parameter vector, and will have a turning point at $\mathbf{a}$ if it is a solution of the set of equations

$$\frac{\partial E(\mathbf{a})}{\partial a_k} = 0, \qquad i = 1, \ldots, n. \tag{1.13}$$

The system of equations (1.13) are referred to as the *normal equations*, which have a global solution given by

$$\mathbf{a} = (C^T C)^{-1} C^T \mathbf{f}. \tag{1.14}$$

The normal equations will only have a solution if the the observation matrix C is of full rank. This is not be the case when data contain repeated abscissae values x_i. The parameter vector solution to the normal equations gives a turning point of the function E, but we need verify that it is a global minimum. The first partial derivatives of E with respect to the parameters can be expressed in matrix form as

$$\frac{\partial E}{\partial \mathbf{a}} = 2C^T(\mathbf{f} - C\mathbf{a}). \tag{1.15}$$

The $n \times n$ Hessian matrix of E is defined as the matrix having i, jth element

$$H_{ij} = \frac{\partial^2 E}{\partial a_i \partial a_j}, \tag{1.16}$$

and can be represented with respect to the observation matrix C as

$$H = 2C^T C. \tag{1.17}$$

H is symmetric positive definite if the observation matrix C is of full rank. As has been mentioned previously, this will be the case provided the abscissae are distinct and the basis functions are linearly independent. Therefore, under these conditions, the solution vector $\mathbf{a}$ given by the normal equations is the minimum of $E(\mathbf{a})$. This solution must be unique as E is a quadratic function of the approximation parameters, and therefore has a unique stationary point.

1.6.1 QR Factorisation

A potential problem with the normal equations (1.13) is that they may suffer from ill-conditioning due to the fact that the condition number of the matrix $C^T C$ is dependent on the square of the condition number of the observation matrix C. This will have little effect if basis functions are chosen to be orthogonal polynomials, but for other basis functions it can pose a serious problem.

If we are presented with an ill-conditioned system of normal equations, solving them via matrix inversion will exacerbate the problem. This can be avoided with the use of *QR factorisation*. The $m \times n$ observation matrix C can be factorised as

$$C = Q \begin{bmatrix} R \\ \mathbf{0} \end{bmatrix}, \tag{1.18}$$

where Q is an $m \times m$ orthogonal matrix, R is an $n \times n$ upper triangular matrix, and $\mathbf{0}$ is the $(m-n) \times n$ zero matrix. Given the matrix Q, we can also factorise $\mathbf{f}$ as

$$\mathbf{f} = Q \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}. \tag{1.19}$$

from which we can rewrite (1.12) as

$$E(\mathbf{a}) = \left\| Q \left[\begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} - \begin{pmatrix} R \\ \mathbf{0} \end{pmatrix} \mathbf{a} \right] \right\|_2^2 . \tag{1.20}$$

Due to the fact that the ℓ_2 norm of a vector is invariant with respect to multiplication by an orthogonal matrix, this reduces to

$$E(\mathbf{a}) = \left\| \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} - \begin{pmatrix} R \\ \mathbf{0} \end{pmatrix} \mathbf{a} \right\|_2^2 . \tag{1.21}$$

Thus $E(\mathbf{a})$ is minimised when $\mathbf{a}$ is a solution to the equation

$$R\mathbf{a} = \theta_1, \tag{1.22}$$

and its minimum value given by $\|\theta_2\|_2^2$. Using QR factorisation does not require matrix inversion as equation (1.22) can be solved by back substitution. It is also possible to form the QR factorisation where $Q \in \mathbb{R}^{m\times n}$ is formed of orthogonal columns, and $R \in \mathbb{R}^{n\times n}$ is a square matrix [24]. There are a number of ways to obtain a QR factorisation of a matrix, some more numerically stable than others. A discussion of a number of such methods and their numerical stability can be found in [24].

1.6.2 Orthogonal polynomials

The ill-conditioning of the normal equations is dependent on the type of basis functions used. As an example, the use of monomial basis functions for least-squares approximation typically results in an ill-conditioned observation matrix, particularly when the abscissae are spaced uniformly. This ill-conditioning can be improved with the use of different polynomial basis functions such as *orthogonal polynomials.* Given vectors $\mathbf{f}, \mathbf{g} \in \mathbb{R}^m$, an *inner product* is defined as a function from $\mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$, whose result is denoted by $< \mathbf{f}, \mathbf{g} >$, and satisfies

1. $< \mathbf{f}, \mathbf{g} > \geq 0$, with equality when $\mathbf{f} = \mathbf{0}$

2. $< \mathbf{f}, \mathbf{g} >=< \mathbf{g}, \mathbf{f} >$

3. $< \mathbf{f}, a\mathbf{g} + b\mathbf{h} >= a < \mathbf{f}, \mathbf{g} > +b < \mathbf{f}, \mathbf{h} >$, where $a, b \in \mathbb{R}$.

A simple example of such a function is the standard scalar product in $\mathbb{R}^n$ defined by

$$< \mathbf{f}, \mathbf{g} >= \sum_{j=1}^{m} f_j g_j.$$

The definition of an inner product also applies when arguments are continuous functions rather than vectors. For example, the following function taking arguments $f, g \in C[-1, 1]$ defined by

$$< f, g >= \int_{-1}^{1} f(x)g(x)w(x)dx,$$

also satisfies the axioms for an inner product, where $w(x) \in C[-1, 1] > 0$. A set of polynomial basis functions $\phi_j(x)_{j=1}^m \in \mathbb{R}^n$ are said to be *orthogonal polynomials* if

$$< \phi_l(x), \phi_k(x) >= c_{lk}\delta_{lk},$$

where c_{lk} is a constant and δ_{lk} is the Kronecker delta symbol that takes value 1 when $l = k$, and 0 when $l \neq k$. If the constant c_{lk} is equal to 1 for all values of l, k then the polynomials are *orthonormal.*

Use of orthogonal polynomial basis functions usually results in a well conditioned observation matrix. One of the most widely used orthogonal polynomials are the Chebyshev polynomials, which are defined on $[-1, 1]$ and are orthogonal with respect to the inner product

$$< \phi_l(x), \phi_k(x) >= \int_{-1}^{1} \frac{\phi_l(x)\phi_k(x)}{(1 - x^2)^{\frac{1}{2}}} dx. \tag{1.23}$$

Specifically these are referred to as *Chebyshev polynomials of the first kind* [40], and the degree n Chebyshev basis function denoted as $T_n(x)$. As well as having nice numerical properties, the basis functions $T_n(x)$ also naturally exhibit the equioscillation property between -1 and 1 on the interval [-1,1]. The function $2^{(n-1)}T_n(x)$ is the best Chebyshev approximation to zero on [-1,1] and this property makes them useful for fitting best Chebyshev approximations to data.

There are a number of other orthogonal polynomials that are widely used including Hermite, Legendre, Laguerre, and Chebyshev polynomials of the second, third, and fourth kind [38],[39],[40]. All orthogonal polynomials also have a general three term recurrence relation

$$\phi_i(x) = (a_i x - b_i)\phi_{i-1}(x) - c_i\phi_{i-2}(x),$$

where the constants a_i, b_i, c_i depend on the type of orthogonal polynomial. In the case of Chebyshev polynomials, this recurrence relation is given by

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \tag{1.24}$$

with $T_0(x) = 1$ and $T_1(x) = x$.

1.7 Rational approximation forms

We define a *generalised rational function* of degree (n, m) as

$$R^n_m(\mathbf{p}, \mathbf{q}, x) \;= \frac{P_n(\mathbf{p}, x)}{Q_m(\mathbf{q}, x)} \tag{1.25}$$

where

$$\begin{aligned} P_n(\mathbf{p}, x) &= \sum_{i=0}^{n} p_i\phi_i(x), \\ Q_m(\mathbf{q}, x) &= \sum_{j=0}^{m} q_j\psi_j(x), \end{aligned}$$

$\{\phi_i(x)\}_{i=1}^n$,$\{\psi_j(x)\}_{j=1}^m$ are sets of linearly independent basis functions, and

$$\mathbf{p} \;=\; (p_0, \dots, p_n)^T \in \mathbb{R}^n, \tag{1.26}$$

$$\mathbf{q} \;=\; (q_0, \dots, q_m)^T \in \mathbb{R}^m, \tag{1.27}$$

are vectors of approximation parameters.

This defines a broad class of rational functions but in this thesis we are mainly concerned with polynomial ratios using monomial or orthogonal polynomial basis functions. In addition, we will only be concerned with rational approximation forms (1.25) with real-valued approximation parameters and basis functions.

Data fitting for rational functions proceeds in the same way as it does in the linear case, that is we calculate approximation parameters $\mathbf{q}, \mathbf{p}$ that minimise

$$\left\| \mathbf{f} - \mathbf{R}_m^n(\mathbf{p}, \mathbf{q}\,\mathbf{x}) \right\| , \tag{1.28}$$

where

$$\mathbf{f} = (f_1, \ldots, f_N)^T \in \mathbb{R}^N, \tag{1.29}$$

$$\mathbf{x} = (x_1, \ldots, x_N)^T \in \mathbb{R}^N, \tag{1.30}$$

and

$$\mathbf{R}_m^n(\mathbf{p}, \mathbf{q}, \mathbf{x}) = (R_m^n(\mathbf{p}, \mathbf{q}, x_1), \ldots, R_m^n(\mathbf{p}, \mathbf{q}, x_N))^T \in \mathbb{R}^N. \tag{1.31}$$

When using rational functions for approximation purposes, it is also necessary to force a normalisation constraint on the approximation parameters. The reason for this is that for any non-zero choice of parameters $\mathbf{p}$ and $\mathbf{q}$ we have

$$R_m^n(\mathbf{p}, \mathbf{q}, x) = R_m^n(\lambda\mathbf{p}, \lambda\mathbf{q}, x)$$

for any constant $\lambda \neq 0$, which allows a particular rational function to be defined by an infinite number of different parameter vectors. Clearly this will pose a problem for approximation purposes, and so we normalise the parameters, usually with the constraint $q_0 = 1$ or forcing $\|\mathbf{q}\| = 1$.

Although the principles of data fitting with rational functions are the same as for linear functions, the problem is considerably more complicated as the error norms that we wish to minimise are nonlinear with respect to the approximation parameters. As a consequence we are often required to use methods applicable to nonlinear problems, which are in general more complicated and computationally intensive than the linear methods described previously. Polynomial ratios are the most widely used rational approximation form and are the study of the majority of research literature in rational approximation. The most notable exception to this is the study of non-uniform rational B-splines (NURBS) that are used extensively in the area of Computer Aided Geometric Design (CAGD). A slightly more detailed introduction to NURBS is given in chapter 2.

1.8 Properties of Rational Functions

Rational approximation forms, such as those defined in (1.25) have a number of features and intrinsic properties that make them a particularly suitable as choice of form, especially when the data or function being approximated exhibits certain types of behaviour. In such cases, rational forms can provide far superior approximations to those obtained using linear forms. We now describe the kind of situations and behaviour for which rational function approximations are particularly well suited, and also discuss some of the advantages and disadvantages associated with their use.

1.8.1 Existence of poles

If we consider the problem of approximating $f(x) = tan(x)$ on the range $[0, \pi]$, we are dealing with a function that has a simple pole at $x = \frac{\pi}{2}$. This particular example illustrates a type of functional behaviour that linear approximations (particularly those of low degree) lack the ability to approximate effectively. This can be seen in figure 1.3 which shows a least-squares polynomial approximation to a discrete set of points sampled from $f(x) = tan(x)$ on $[0, \pi]$. The approximation is generally poor over the entire interval, but appears to be particularly bad in the vicinity of the pole. Unlike linear forms, a rational function is also able to have a simple pole and provides an approximation form that has the potential to reproduce the same asymptotic be-

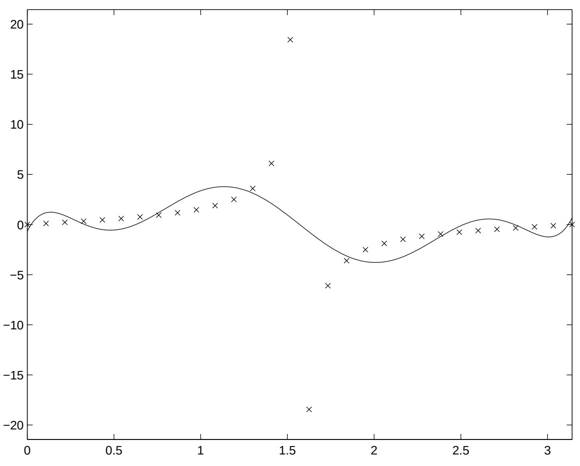

Figure 1.3: Degree 8 ℓ_2 polynomial approximation of $f(x) = tan(x)$ at 30 equally spaced points on the interval [0,π].

haviour as the data being approximated. Figure 1.4 shows a least-squares polynomial ratio approximation of the same set of points used in Figure 1.3. The superiority of this rational approximation is clear to see, and the pole at $x = \frac{\pi}{2}$ has been fitted in the rational function to an accuracy of 6 decimal places. The rational form fitted to the data in this example is of degree (2,2) and despite having fewer approximation parameters than the polynomial approximation of figure 1.3 it still provides a superior approximation.

The ability of rational forms to have poles is clearly advantageous in the case of the previous example, but this may not always be the case. Sometimes a pole is reproduced in the approximant with much less precision, and sometimes not reproduced at all. This is particularly true when approximating a function with a large number of poles. Careful thought needs to be given to the degree of rational function used to approximate a function or data known to contain poles, as the maximum number of poles the approximation may have is equal to the degree of the denominator.

Although it has been shown that pole fitting is a potentially useful property of rational functions, it can also cause problems. It is particularly troublesome when

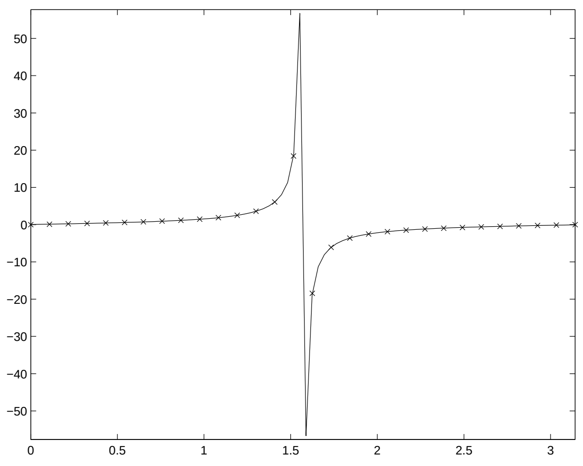

Figure 1.4: Degree (2,2) rational ℓ_2 approximation of $f(x) = tan(x)$ at 30 equally spaced points on the interval [0,π].

the data or function being fitted does not contain poles, but the resultant rational approximation does contain poles. In such cases this may not be a problem provided the poles lie outside the interval of approximation and away from the data. If there are poles present in the approximation range then this commonly leads to an increase in the approximation error in the vicinity of the pole.

Figure 1.5 shows a degree $(3, 5)$ rational least-squares approximation of a set of equally spaced points from the function $f(x) = e^{-x^2}$ to which a small amount of normally distributed noise has been added. It can be seen that the approximation is good on the majority of the range, but an unwanted pole has been fitted at $x = 3.823$. Figure 1.6 provides a magnified view of the region near the pole and clearly illustrates the detrimental effect that this has on the quality of the approximation.

1.8.2 Numerical considerations

Numerical error also needs to be considered in relation to the problem of unwanted poles. Consider the case where we have used an arbitrary algorithm to fit a rational

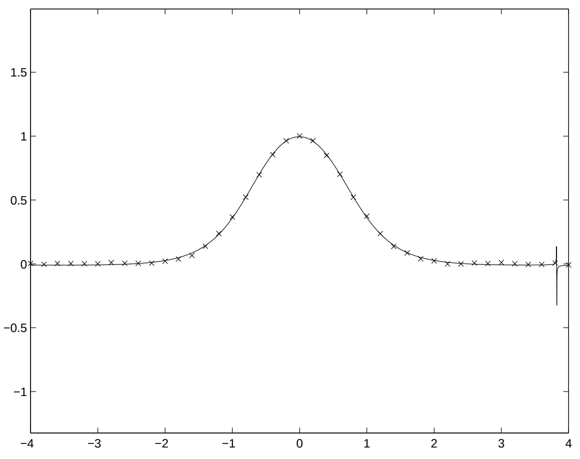

Figure 1.5: Degree $(3, 5)$ ℓ_2 rational approximation of $f(x) = e^{-x^2}$ containing normally distributed noise.

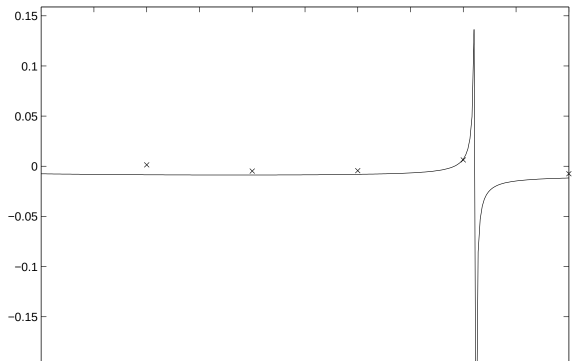

Figure 1.6: The effect of the pole at $x = 3.823$ on the approximation of Figure 1.5.

approximation to a set of data, and the algorithm has converged to a solution. Let us further assume that this solution takes the form

$$\frac{(x-c)\sum_{i=1}^{n-1} p_i x^i}{(x-c)\sum_{i=1}^{m-1} q_i x^i}, \tag{1.32}$$

where $c \in \mathbb{R}$. If this is the case, the factor $(x - c)$ common to denominator and numerator is able to be removed from the function. However, it may be the case (due to numerical issues such as rounding error) that the algorithm has converged to a solution that is of the form

$$\frac{(x-(c+\delta_1))\sum_{i=1}^{n-1} p_i x^i}{(x-(c+\delta_2))\sum_{i=1}^{m-1} q_i x^i}, \tag{1.33}$$

where $\delta_1 \neq \delta_2$ are very small non-zero real numbers. In this case cancellation is not possible and so we are faced with the question of whether the pole in the approximation should be there or not. Rational functions that have had all factors common to numerator and denominator removed are referred to as *irreducible*. To help avoid this kind of potential problem, it is possible to prevent unwanted poles by constraining the parameters in a way that forces the denominator to be strictly positive on the approximation range. It is also possible to try and force the rational function to fit poles outside of the range of approximation, or explicitly define a factor $(x-c)$ in the denominator, where $c \in \mathbb{R}$ lies outside the approximation range. In particular we can try and enforce the roots of the denominator polynomial to be complex, and some simple methods to achieve this are presented in chapter 5.

1.8.3 Nonlinearity

As has been previously mentioned, the major difficulty associated with rational approximation is the nonlinearity of the approximation parameters. Methods for fitting

linear approximations are not immediately applicable to nonlinear problems, and instead we will often need to utilise nonlinear optimisation methods. There are a number of optimisation techniques available for fitting general nonlinear forms to data and these can be applied to the rational approximation problem. Optimisation methods are useful tools but they can be computationally intensive, and often their convergence is conditional on an appropriate selection of initial values for the approximation parameters. Some of the most commonly used optimisation methods (such as the Newton (2.8.1) and Gauss-Newton methods (2.8.2)) are utilised extensively in this thesis and will be described in detail in Chapter 2.
Despite the nonlinearity of the parameters in a rational function, there are a number of methods available for fitting rational approximations that only require the solution of linear systems of equations (usually as part of an iterative process). Such methods include the Gauss-Newton and Newton methods [18] mentioned previously. Others include methods for finding rational interpolants [51],[23],[33], Padé approximations [4], Thiele interpolants [14], and weighted iterative methods for fitting rational approximations (such as Loeb's algorithm [7] and the Differential Correction Method [27]). We will describe some of these methods in greater detail in Chapter 2.

1.8.4 Asymptotic limits

Another advantageous feature of rational functions is that they may be used to approximate over an infinite range. They can also be used to approximate functions that exhibit certain types of asymptotic behaviour as their arguments tend to infinity. A simple example of this is the approximation of $tanh(x)$ on the interval $[0, \infty)$. The function $tanh(x)$ the finite limit

$$\lim_{x \to \infty} tanh(x) = 1 \tag{1.34}$$

and so the use of a polynomial to approximate it on this interval would be seem to be an unsuitable choice of form, due to the fact that all polynomials have an asymptotic limit of $\pm\infty$. However, there is a subset of generalised rational functions that also have a constant asymptotic limit, and it would then seem natural to use such an approximant for this problem. We now describe some of the types of asymptotic behaviour that rational functions are able to model effectively.

1.8.5 Decay to a constant value as $x \to \pm\infty$

We consider the problem of approximating data sampled from a function $f(x)$ that has known asymptotic behaviour given by

$$\lim_{x\to\infty} f(x) = \alpha, \tag{1.35}$$

where $\alpha \neq 0$ is a real valued constant. Asymptotic decay to a constant is an intrinsic property of ratio of equal degree polynomials, and so it would seem sensible to approximate $f(x)$ using an approximation of this kind. The exact asymptotic limit of a degree (n, n) polynomial ratio $R_n^n(\mathbf{p}, \mathbf{q}, x)$ is given by

$$\lim_{x\to\infty} R_n^n(\mathbf{p}, \mathbf{q}, x) = \frac{p_n}{q_n}, \tag{1.36}$$

where p_n and q_n are the coefficients of the highest degree polynomial basis functions used in the numerator and denominator respectively. Therefore, we can explicitly force a degree (n, n) polynomial ratio approximant to have exactly the same limit as $f(x)$ by imposing the parameter constraint

$$p_n = q_m\alpha. \tag{1.37}$$

This constraint can be enforced explicitly in the function definition using substitution, or could be utilised in the form of a parameter constraint as part of the approximation process. We have specified an asymptotic limit as $x \to \infty$ for illustrative purposes, and the same results will apply to a limit specified as $x \to -\infty$. We investigate the quality of approximation using rational forms with the same limiting behaviour enforced upon them in later chapters.

1.8.6 Decay to zero as $x \to \pm\infty$

Now suppose we wish to approximate data sampled from a function $f(x)$ that has known asymptotic behaviour given by

$$\lim_{x\to\infty} f(x) = 0. \tag{1.38}$$

There are a large number of generalised rational functions that have the same asymptotic limit as that in (1.38). The most obvious choice of such a function would be a polynomial ratio whose numerator degree is less than that of the denominator, which leads to a large amount of potential choices for the degrees n, m.

1.8.7 Approximating limiting behaviour of the type x^k as $x \to \pm\infty$

Now we consider the problem of approximating data from a function $f(x)$ that has the limit

$$\lim_{x\to\infty} \frac{f(x)}{x^k} = \alpha, \tag{1.39}$$

where $\alpha \in \mathbb{R}$ and $k \in \mathbb{N}$ are known constants. This type of limiting behaviour can be achieved once again by a ratio of polynomials, provided that the numerator degree is larger than that of the denominator. We restrict our choice of approximant to degree (n, m) rational approximant by $R^n_m(\mathbf{p}, \mathbf{q}, x)$ which has positive asymptotic limit given by

$$\lim_{x\to\infty} \frac{R^n_m(\mathbf{p}, \mathbf{q}, x)}{x^{(n-m)}} = \left(\frac{p_n}{q_m}\right), \qquad n > m, \tag{1.40}$$

where p_n and q_m are the coefficients of the highest degree numerator and denominator basis functions respectively. Clearly our rational approximant will then share the asymptotic limit (1.39) provided that the following conditions are satisfied

$$n - m = k, \tag{1.41}$$

$$p_n = q_m\alpha. \tag{1.42}$$

The first of these conditions is easily imposed as the degree of the approximating form needs to be fixed prior to approximation anyway. The second condition can be imposed explicitly within the definition of the form by direct substitution. It may also be able to achieve this constraint as a part of the approximation algorithm being used for data fitting.

We now consider approximation of functions with the negative asymptotic limit

$$\lim_{x \to -\infty} \frac{f(x)}{x^k} = \alpha, \tag{1.43}$$

for $k \in \mathbb{N}$ and $\alpha \in \mathbb{R}$. In the same way as in the previous example we will need to satisfy constraint (1.41), and then the equivalent negative asymptotic limit for the approximant will be

$$\lim_{x \to -\infty} \frac{R_m^n(\mathbf{p}, \mathbf{q}, x)}{x^k} = \left(\frac{p_n}{q_m}\right). \tag{1.44}$$

From this equation it is clear that we can make the approximant satisfy the limit (1.43) with the constraint (1.42) as before. We do not have to deal separately with the issue of the limit of x^k as $x \to -\infty$ for odd number values of k, as (1.42) ensures the correct limiting behaviour of the function.

1.8.8 Approximation of double sided asymptotic limits

We now need to address the problem of approximating a function or data over the entire real line. In this case we are faced with task of modelling two asymptotic limits, one as $x \to -\infty$, and one as $x \to +\infty$. We will consider the problem of approximation of functions $f(x)$ that have asymptotic limits of the form

$$\begin{array}{rcl} \lim_{x \to \infty} \frac{f(x)}{x^{k_1}} & = & \alpha_1, \\ \lim_{x \to -\infty} \frac{f(x)}{x^{k_2}} & = & \alpha_2, \end{array} \tag{1.45}$$

where α_1, $\alpha_2 \in \mathbb{R}$ and $k_1, k_2 \in \mathbb{N}$. An example of such a dataset exhibiting double-sided asymptotic behaviour is illustrated in figure 1.7. As in the previous section, a polynomial ratio would seem to be the most appropriate choice of rational approximation form. The limits of a polynomial ratio are either zero, or given by (1.40) or (1.44) depending on the degrees of numerator and denominator. Therefore, it is not possible

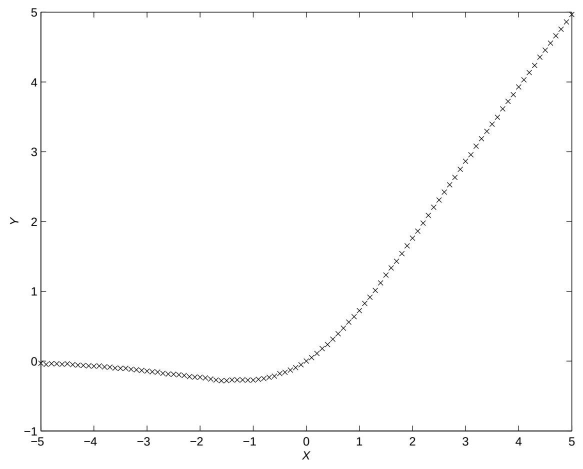

Figure 1.7: Artificial data exhibiting double-sided asymptotic behaviour sampled from $f(x) = x(1 + e^{-x})^{-1}$.

for a standard polynomial ratio approximant to possess the limits (1.45) except for special cases where $k_1 = k_2$ and $\alpha_1 = \alpha_2$. In the case of modelling asymptotic decay to zero in both directions this will not pose a problem, as we can set $k_1 = k_2 = 0$, but for the more generic problem involving mixed limits, we require a new rational form to approximate the data. This problem is revisited in more detail in chapter 3. We also consider problems with double limits specified by (1.45) but extend the definition to deal with non-integer values for the exponents $k1, k2$. This vastly increases the different types of data that may be approximated, as we are no longer restricted to consideration of integer powers of x as a limit. Clearly this problem cannot be dealt with using standard polynomial ratios, and so new approximation forms and methods are required. This work will be presented in chapter 4.

The least-squares approximation of data that exhibits these types of asymptotic behaviour is the primary focus of this project, and this is the reason that specific study of rational functions has been undertaken. The particular types of asymptotic behaviour described here have been chosen as they are of specific interest to the National Physical Laboratory (NPL) who are the collaborating partial sponsor of this project.

The reason for their interest in these specific types of asymptotic behaviour is because they occur frequently in physical systems (particularly decay to zero or decay to a constant), or are exhibited by the solutions to some types of differential equations. The objective of the project is to obtain good approximations to these specific types of data set with the use of traditional rational functions and some new nonlinear approximation forms that are of a rational nature. Specifically, we hope to obtain good approximations with the use of carefully selected approximation forms that mimic the asymptotic behaviour of the observed data, and investigate whether they are superior to those obtained with approximation forms that do not (such as polynomials and splines). It is important to specify that we will consider two types of problem

1. The approximation of data that has a specified type of asymptotic behaviour that is known beforehand. An example of such a problem is the modelling of data from an experiment where there is some theoretical knowledge of the physics of the experiment that provides knowledge of the asymptotic behaviour.

2. The approximation of data that has an *implied* asymptotic behaviour based on the shape of the data in question, but for which there is no theoretical justification.

Within this project, approximation of data that falls into the second of these categories is dealt with more frequently than the first.

1.9 Example applications from industry

This chapter has highlighted some of the properties of rational functions that make them a particularly appropriate form for modelling the kinds of asymptotic behaviour exhibited by many physical systems. These types of asymptotic behaviour are found to occur in many of the physical systems currently being investigated by the National Physical Laboratory (NPL) in the area of Metrology, and explains their interest in this project. An example from physics that illustrates a problem suitable for rational functions is the approximation of the solution of the Blasius equation.

1.9.1 The Blasius equation

The Blasius equation occurs in the boundary layer problem of hydrodynamics, and is a differential equation which can be expressed as

$$\frac{d^3y}{dx^3} + y\frac{d^2y}{dx^2} = 0 \tag{1.46}$$

subject to the boundary conditions $y(x) = y'(x) = 0$ at $x = 0$ and $y'(x) \to \gamma$ as $x \to \infty$, where γ is a known positive constant. We see immediately from the last of these boundary conditions that the solution to this equation has the asymptotic behaviour described in section (1.8.7) and is therefore a suitable candidate for approximation using a polynomial ratio. Finding rational function solutions to the Blasius equation (1.46) has been studied by Mason [37].

1.9.2 Mesopic efficiency functions

Another application of rational forms is the approximation of data that has been used to calculate a mesopic efficiency function. This is an experiment undertaken by City University, the aim of which is to model the sensitivity of the eye to light of a variety of wavelengths. Data have been collected experimentally from an experiment which records the length of time it takes a human subject to respond to light signals of differing wavelength and intensities. As it is known that the human eye can only detect electromagnetic radiation lying inbetween infra-red and ultra-violet wavelengths, we know that the response variable will decay to zero as wavelength increases and decreases. The actual data from one of these experiments are shown in figure 1.8 . This particular experiment and approximation of the recorded data is discussed again in chapter 4.

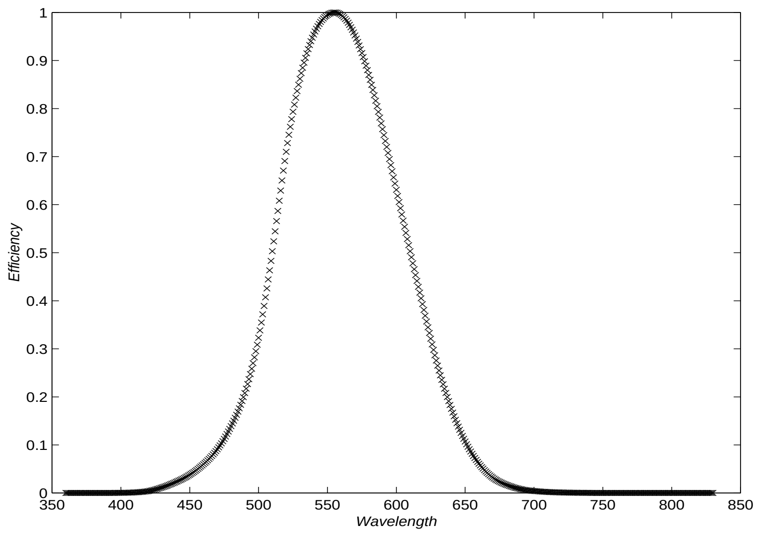

Figure 1.8: Experimental data from the mesopic efficiency experiment.

Chapter 2

EXISTING METHODS FOR FITTING RATIONAL APPROXIMATIONS

2.1 Introduction

In this chapter we describe some of the more popular methods used for fitting rational approximations of the form (1.25) to discrete data or functions. Some of these methods have been proved to converge to a best approximation in certain norms. The majority of the methods presented here are used at some stage in the thesis, with others mentioned only to illustrate areas of rational approximation that have been researched and found not to be directly applicable to the project. We also describe some common non-linear optimization techniques that may be used to fit rational functions.

2.2 The Loeb algorithm

Loeb's algorithm is a weighted iterative procedure that can be used to fit a generalised rational function to a set of discrete data points $\{(x_i, f_i)\}_{i=1}^{N}$. Instead of minimising the norm of the residual vector (1.28), the algorithm works by solving

$$\min_{\mathbf{p}^{(k)},\mathbf{q}^{(k)}} \|\mathbf{\Delta}^{(k)}(\mathbf{p}^{(k)}, \mathbf{q}^{(k)}, \mathbf{x})\| \tag{2.1}$$

at iteration k where $\mathbf{\Delta}^{(k)} \in \mathbb{R}^N$ is the vector with ith element

$$\Delta_i^{(k)} = \frac{1}{Q_m(\mathbf{q}^{(k-1)}, x_i)}(P_n(\mathbf{p}^{(k)}, x_i) - f_i Q_m(\mathbf{q}^{(k)}, x_i)). \tag{2.2}$$

The function $(Q_m(\mathbf{q}^{(k-1)}, x))^{-1}$ is treated as a known weight function, and is obtained by evaluating the denominator function using the parameters $\mathbf{q}^{(k-1)}$ from the previous iteration. The result of this is that the original nonlinear approximation problem has been simplified to an iterative weighted linear problem, with the fixed weight vector replacing the denominator function. The approximation parameters are usually normalised by setting $q_0 = 1$ and the algorithm initialised with weight $Q_m(\mathbf{p}^{(0)}, \mathbf{q}^{(0)}, x) = 1$. Variation of the choice of start weight has been found to have very little effect on the performance of the algorithm.

The Loeb algorithm may be applied to any norm , although it was originally suggested for the ℓ_∞ norm by Loeb [32] and subsequently for the ℓ_2 norm by Whittmeyer [53]. Barrodale and Mason [7] apply the Loeb algorithm to fit approximations to discrete data using the ℓ_1, ℓ_2 and ℓ_∞ norms. This method is attractive due to its ease of implementation, although it is sometimes unreliable due to lack of convergence. From our experience with the algorithm, we found it to converge in the vast majority of cases and when it does converge it does so to good approximations, particularly for least-squares problems. These findings are also verified by the results of Barrodale and Mason [7] who have used the method to approximate a large number of functions, including those with poles. A drawback of the algorithm is that, as far as we are aware, there is no convergence proof for the algorithm, and when used for least-squares problems even if the algorithm does converge, it is almost certain not to converge to a best approximation [7]. Despite this, the parameters obtained from one or two iterations of the Loeb algorithm are often a good choice of start parameters for classical optimisation methods such as Newtons method (described in section 2.8.1). We have also noticed that the application of optimisation methods from these start parameters, only generates a very minor improvement in the quality of the approximation.

2.2.1 Least-squares approximation with the Loeb algorithm

We now describe application of the Loeb algorithm to the specific case of least-squares approximation with polynomial ratios, as it will be applied extensively throughout the thesis. For the ℓ_2 rational approximation problem we wish to minimise the quantity

$$\sum_{i=1}^{N}\left(f_i - \frac{P(\mathbf{p}^{(k)}, x_i)}{Q(\mathbf{q}^{(k)}, x_i)}\right)^2, \tag{2.3}$$

at iteration k. We linearise this quantity by multiplying each residual by the denominator and a fixed weight term $w_i^{(k)}$ to give

$$\sum_{i=1}^{N} w_i^{(k)}\Big(f_i Q(\mathbf{q}^{(k)}, x_i) - P(\mathbf{p}^{(k)}, x_i)\Big)^2, \tag{2.4}$$

where the weight

$$w_i^{(k)} = \frac{1}{Q(\mathbf{q}^{(k-1)}, x_i)} \tag{2.5}$$

is chosen to simulate the effect of the denominator. The use of a fixed weight reduces the problem to a linear system which can be solved easily.

Because of the normalisation condition $q_0 = 1$, we can write (2.4) as

$$\sum_{i=1}^{N} w_i^{(k)}\Big(f_i + f_i \sum_{j=2}^{m} q_j^{(k)} x_i^j - \sum_{j=1}^{n} p_j^{(k)} x_i^j\Big)^2. \tag{2.6}$$

and this least-squares problem can be represented in matrix form at iteration k as

$$W^{(k)}\mathbf{f} = WC\mathbf{d}^{(k)}. \tag{2.7}$$

Here the observation matrix $C \in \mathbb{R}^{N\times(m+n+1)}$ is defined by its i, jth element

$$\begin{aligned} C_{ij} &= x_i^j, \qquad (j = 1, \ldots, n+1), && (2.8)\\ C_{ij} &= -f_i x_i^{(j-(n+1))}, (j = n+2, \ldots, m+n+1), && (2.9)\end{aligned}$$

$W^{(k)} \in \mathbb{R}^{N\times N}$ is a diagonal matrix of weight terms defined by its i, jth element

$$\begin{aligned} W_{ii}^{(k)} &= w_i^{(k-1)}, (i = 1, \ldots, N), && (2.10)\\ W_{ij}^{(k)} &= 0, (i \neq j), && (2.11)\end{aligned}$$

and $\mathbf{d}^{(k)} = \{p_0^{(k)}, \ldots, p_n^{(k)}, q_1^{(k)}, \ldots, p_m^{(k)}\} \in \mathbb{R}^{m+n+1}$ is the vector of approximation parameters at step k. The Loeb algorithm is then easily implemented by solving the linear least-squares system (2.7) using QR factorization (1.6.1).

2.2.2 Ill-conditioning associated with the Loeb algorithm

The matrix C defined in equation (2.8) is susceptible to ill-conditioning as a consequence of its structure. Furthermore, this ill-conditioning is not dependent on the choice of polynomial basis used. To illustrate this, we compare the condition number of the matrix C formed from a degree $(5, 5)$ polynomial ratio using a monomial basis and a Chebyshev polynomial basis. As data points we take 51 equally spaced abscissae on $[-1, 1]$ at which we evaluate the function $cos(x)$ to represent the data points f_i. Forming the observation matrix C using the monomial basis leads to a condition number of 6.266×10^9 while the same matrix calculated with Chebyshev polynomial basis has condition number 5.155×10^9. Approximation using polynomials with uniformly spaced abscissae results in ill-conditioning [40], but if we change the abscissae to be the zeros of the degree 51 Chebyshev polynomial basis function $T_{51}(x)$ the condition number of C (using Chebyshev bases) is now 4.517×10^9. The use of Chebyshev basis functions that are evaluated at Chebyshev zeros gives a well conditioned system of equations [40], but in this case it is not so, and we conclude that the ill-conditioning is associated with the Loeb method itself.
In an attempt to improve the conditioning we tried degree $(5, 5)$ rational functions defined with different orthogonal polynomial basis functions in numerator and denominator. The condition number for a selection of these different choices is shown in table 2.1.

We can explain the possible reasons for this ill-conditioning by writing the matrix C as

$$\left(P\middle|BQ\right) \tag{2.12}$$

where P and Q are the standard observation matrices obtained from the numerator and denominator basis functions $\{\phi_i(x)\}_{i=0}^n$ and $\{\psi_j\}_{j=1}^m$ respectively (1.25), and B is the diagonal matrix with ith diagonal element $B_{ii} = f_i$ and all other elements equal to zero. The matrix B has the effect of multiplying row i of Q by the constant value

Table 2.1: Effect on condition number on choices of orthogonal polynomial basis functions

Numerator basis	Denominator basis	Cond(C)
Chebyshev	Monomial	4.024×10^9
Chebyshev	Legendre	4.189×10^9
Legendre	Chebyshev	5.130×10^9
Hermite	Monomial	1.088×10^{11}

f_i, and so when we use the same basis functions in numerator and denominator we end up with columns that are identical up to a diagonal matrix multiplication. Now if we consider the case of approximating data from a function having almost zero gradient, using monomial basis functions then we will have column j of P almost equal to column $j+1$ of Q and such linear dependence between columns will result in serious ill-conditioning or numerical rank deficiency.

This is of particular importance when bearing in mind the kind of approximation problems that we are interested in as described in sections (1.8.7) using monomial basis functions. In such cases the multiplicative function values $f_i \simeq x_i^k$ for integer values k (1.39), which leads to linear dependence between certain columns of P and Q.

The conditioning is reasonable (in the region of 10^6) for degree (n, m) approximations for which $n + m \leq 8$ but quickly becomes worse for higher degrees. We have also observed that in general when using a monomial basis, the condition number of C is minimal for the choice of degree with $n = m$. This will be due to the fact that the matrices P and Q are themselves monomial basis observation matrices, which are generally ill-conditioned to start with. The highest degree monomial basis function contained in C will be $x^{max(n,m)}$ and clearly $max(n, m)$ is minimised when $n = m$.

Observations on the behaviour of Loeb's algorithm

For least-squares problems, we have found that the Loeb algorithm is very easy to apply and in general gives good approximations, that are only slightly poorer in quality to those obtained using optimisation techniques. However, after extensive use of this algorithm, we have noticed some patterns of behaviour that are worth mentioning. Firstly, it appears that in a large number of cases where the algorithm fails to converge it seems to do so through a poor choice of degree for the shape of the data. As an example, a degree (3,3) polynomial ratio was used to approximate 51 sampled points of the function $f(x) = e^{-x^2}$ on the interval $[-4, 4]$, with a small amount of gaussian noise present. Convergence was defined as occurring when the Chebyshev norm of the difference between the solution vectors from two successive iterations was less than 10^{-10}. Using the Loeb algorithm to fit the (3,3) polynomial ratio, convergence still had not occurred after 250 iterations. However, by approximating with a (2,2) polynomial ratio, the algorithm converged within 10 iterations, despite having two less approximation parameters than the (3,3) degree approximation. This behaviour has been observed in other cases also, usually when the function being approximated is even, and the degree of the approximant is odd (and vice versa).
Another observation is that the algorithm can often get 'stuck' and solution parameters from successive iterations will oscillate between 2 or more sets of solution parameters. In some cases the algorithm has been seen to get stuck between a set of 6 distinct sets of solution parameters. In all cases where we have observed this behaviour, at least one of sets of parameters yields an approximation that contains poles in the approximation range. In such cases, it is possible to choose to terminate the algorithm choosing the best set of parameters from the set that the algorithm oscillates between. This behaviour is most commonly seen when the degree of the approximant is too small and can be overcome by increasing the degree of numerator or denominator (or both).

2.3 The Differential Correction Method (DCM)

The differential correction method is an iterative algorithm specific to the ℓ_∞ norm, and is proven to converge to the best Chebyshev rational approximation on a discrete data set. The algorithm as we describe it here is specific to the case of approximation with polynomial ratios. The DCM has two variations, the first of which was put forward in [12]. We describe here the modified version given in [11] in which a proof of (at least linear) convergence is also given.

Starting with the same data points and approximation problem described in (2.2), the DCM begins by choosing an arbitrary initial rational function $R^{(0)}(x) = P^{(0)}(x)/Q^{(0)}(x)$ where we define

$$P^{(k)}(x) = P_n(\mathbf{p}^{(k)}, x), \tag{2.13}$$

$$Q^{(k)} = Q_m(\mathbf{q}^{(k)}, x), \tag{2.14}$$

for ease of notation. The only restriction placed on $R^{(0)}$ is that it has no poles within the interval of approximation. Then at iteration k, denote the maximum current approximation error as

$$\Delta^{(k)} = \|\mathbf{f} - \mathbf{R}^{(k)}(\mathbf{x})\|_\infty, \tag{2.15}$$

and then minimise the quantity

$$\delta^{(k)} = \max_{1\leq i\leq N} \left\{ |f_i Q^{(k+1)}(x_i) - P^{(k+1)}(x_i)| - \Delta^{(k)} Q^{(k+1)}(x_i) \right\} \tag{2.16}$$

is minimised with respect to the parameters $\mathbf{p}^{(k)}, \mathbf{q}^{(k)}$ subject to the constraint

$$\|\mathbf{Q}^{(k+1)}(\mathbf{x})\|_\infty = 1. \tag{2.17}$$

The algorithm terminates when we find a function $R^{(k+1)}(x)$ such that $\delta^{(k)} \geq 0$ and the best approximation is then given by $R^{(k)}(x)$.

The original DCM [12] varies from the modified version only in the definition of $\delta^{(k)}$ which becomes

$$\delta^{(k)} = \max_{1\leq i\leq N} \left\{ \frac{|f_i Q^{(k+1)}(x_i) - P^{(k+1)}(x_i)| - \Delta^{(k)} Q^{(k+1)}(x_i)}{Q^{(k)}(x_i)} \right\}. \tag{2.18}$$

The modified version of the DCM was widely in favour of the original version due to its proved linear convergence properties, however, the original version was subsequently proved to have quadratic convergence [27].
There are a number of variations of the DCM that have been published, most involving some form of constrained approximation. Kauffman and Taylor implement a version that allows linear constraints to be placed on the approximation parameters [28] as well as another version that places a strictly positive lower bound on the denominator function [29]. Gugat [25] presents an implementation of the algorithm that forces the denominator to be bounded above and below by continuous functions. The general DCM was also extended by Cheney and Powell [13] for approximation using generalised rational functions and proved to have superlinear convergence subject to a unique solution.

2.4 Other work on rational approximation

In addition to the Differential Correction method and its variants, there are a number of other methods for fitting Chebyshev rational approximations, for both discrete data, and function approximation. The exchange algorithm of Remes [14] is another method often referred to, and is the rational equivalent of the exchange algorithm used to fit best linear Chebyshev approximations. Kauffman, Leeming and Taylor [20] consider an approach using a combined Remes-Differential correction method for fitting approximations on subsets of $[0, \infty)$, and another method for approximation on the same interval using polynomial reciprocals [19]. A number of other methods of fitting Chebyshev rational approximations are put forward by Maehly in [34, 35].
We have not found a great deal of material in the field of least-squares rational approximation. There is a huge amount of material and research on nonlinear least-squares optimisation methods, but we have only found a small amount of material specifically concerning rational approximation of discrete data. The paper of Pomentale [45] deals

with fitting pole free least-squares rational approximations with a denominator function $Q(x)$ bounded below by a parameter $\epsilon > 0$. We have also found some research on complex variable least-squares rational approximation [5]

2.5 Padé approximation

The area of Padé approximation is very large and there is a considerable amount of literature in this field, and for this reason, any discussion of rational approximation would be incomplete without mentioning Padé approximations. Let us assume that we are trying to approximate a function, and that this function can be defined by a power series

$$f(z) = \sum_{i=0}^{\infty} f_i z^i. \tag{2.19}$$

If there exist two polynomials $P_n(z)$ and $Q_m(z)$ of degrees n and m respectively such that

$$Q_m(z)f(z) - P_n(z) = O(z^{n+m+1}), \tag{2.20}$$

then the function $\pi_{nm} = P_n/Q_m$ is called the *Padé approximant* of order (n, m) of $f(z)$. It can be viewed as matching the power series $f(z)$ truncated at the $(n+m+1)$th term. This definition has been taken from [43] which also describes how to calculate the Padé approximant as follows. If we express the polynomials P, Q by

$$\begin{aligned} P_n(z) &= a_n z^n + \ldots + a_1 z + a_0, \\ Q_m(z) &= b_m z^m + \ldots + b_1 z + b_0, \end{aligned}$$

we can see that condition (2.20) is equivalent to equating the coefficients of z^k to zero for the function $Q_m(z)f(z) - P_n(z)$ for values $k = 0, \ldots, n + m$. This leads to the requirements

$$\begin{aligned} a_0 &= f_0 b_0, \\ a_1 &= f_0 b_1 + f_1 b_0, \\ &\ldots \\ a_n &= \textstyle\sum_{i=1}^{\min(n,m)} f_{n-i} b_i + f_n b_0, \end{aligned}$$

and

$$\begin{aligned}
b_0 f_{n+1} + b_1 f_n + \ldots + b_m f_{n-m+1} &= 0, \\
b_0 f_{n+2} + b_1 f_{n+1} + \ldots + b_m f_{n-m} &= 0, \\
\ldots & \\
b_0 f_{n+m} + b_1 f_{n+m-1} + \ldots + b_m f_n &= 0,
\end{aligned}$$

where $f_i = 0$ when $i < 0$. The second system of equations above always has a solution as it is a linear system of m equations in $m+1$ unknowns $b_0, \ldots, b_m$, from which the coefficients of $P_n(z)$ are easily obtained using the first system of equations above. There is a large amount of research into Padé approximation, not only in the univariate case but in the multivariate case too [16],[1]. Other areas are Newton-Padé approximation [17], Vector Padé approximation and multipoint Padé approximation. As the specific problem that this project is faced with is approximation of discrete data rather than functions (or their power series expansions), we have found that the field of Padé approximation is not directly applicable to the project. It has been mentioned due to the fact that makes a large contribution to the area of rational approximation as a whole.

2.6 *Appel's algorithm*

We consider the specific problem of approximating data exhibiting exponential decay as described in section 1.8.6. A suitable approximation form for this purpose is a rational approximation $R(x)$ of the form

$$R(x) = \frac{s(x)}{(Q_m(\mathbf{q}, x))^r}, \tag{2.21}$$

where $s(x)$ is a specified fixed function, $Q_m(\mathbf{q}, x)$ is a linear denominator function as defined in (1.25), $r \in \mathbb{Z}$ is suitably chosen to model decay effectively. This form can be fitted to a discrete data set with a simple one step algorithm due to Appel [3] which we now describe. At each data point x we have an error component $e(x)$ given by

$$f(x) = R(\mathbf{q}, x) + e(x). \tag{2.22}$$

Substitution of (2.21) into this equation gives

$$Q_m(\mathbf{q}, x) = \left(\frac{s(x)}{f(x) - e(x)}\right)^{\frac{1}{r}} = \left(\frac{s(x)}{f(x)}\right)^{\frac{1}{r}} \left(1 - \frac{e(x)}{f(x)}\right)^{-\frac{1}{r}}. \tag{2.23}$$

Taking a Taylor expansion of the second term of the right hand side of (2.23) and ignoring quadratic and higher order terms gives

$$Q_m(\mathbf{q}, x) \approx \left(\frac{s(x)}{f(x)}\right)^{\frac{1}{r}} \left(1 + \frac{e(x)}{rf(x)}\right), \tag{2.24}$$

and so

$$G(x)Q(\mathbf{q}, x) - rf(x) \approx e(x), \tag{2.25}$$

where

$$G(x) = rf(x)\left(\frac{f(x)}{s(x)}\right)^{\frac{1}{r}}. \tag{2.26}$$

Thus we can expect to obtain (close to best) approximations to the nonlinear problem

$$\min_{\mathbf{q}} \|R(\mathbf{q}, x) - f(x)\|, \tag{2.27}$$

by solving the linear approximation problem

$$\min_{\mathbf{q}} \|G(x)Q(\mathbf{q}, x) - rf(x)\|. \tag{2.28}$$

2.7 Non-Uniform Rational B-Splines

Non-uniform rational B-splines (NURBS) are another kind of widely used rational function. These are not only used within the field of approximation, but are very powerful tools for shape representation and are used extensively within the area of computer aided geometric design (CAGD) and within CAD software. A NURBS curve is a parametric curve that is a weighted combination of fixed geometric points called *control points* multiplied by ratios of B-Spline basis functions. The following formal definition of a NURBS curve is taken from [44]. A degree p NURBS curve $\mathbf{C}(u)$ is defined over a parametric interval $[a, b]$ by

$$\mathbf{C}(u) = \frac{\displaystyle\sum_{i=0}^{n} N_{i,p}(u) w_i \mathbf{P}_i}{\displaystyle\sum_{i=0}^{n} N_{i,p}(u) w_i}, a \leq u \leq b \tag{2.29}$$

where $\mathbf{P}_i$ are the set of $n + 1$ control points, $w_i > 0$ are the *weights* associated with each control point, and $N_{i,p}(u)$ are the degree p B-Spline basis functions defined on the knot vector

$$U = \underbrace{a, \ldots, a}_{p+1}, u_{p+1}, \ldots, u_{m-p-1}, \underbrace{b, \ldots, b}_{p+1}.$$

The interval $[a, b]$ is commonly assumed to be $[0, 1]$, and the curve is often defined in the form

$$\mathbf{C}(u) = \sum_{i=1}^{n} R_{i,p}(u) \mathbf{P}_i, \tag{2.30}$$

where

$$R_{i,p}(u) = \frac{N_{i,p}(u) w_i}{\displaystyle\sum_{i=0}^{n} N_{i,p}(u) w_i}, \tag{2.31}$$

are called the *rational basis functions.* The number of control points $(n + 1)$, the degree p, and the number of knots $(m + 1)$ are related by

$$m = n + p + 1. \tag{2.32}$$

Each weight w_i is associated with control point $\mathbf{P}_i$ and describes the affinity that the curve has for that control point. If the weight of a control point is increased, then the effect is that the curve will be more strongly attracted to its control point. This is illustrated in figure 2.1, where the curve labelled NURB2 has a weight of 5 times that of curve NURB1 associated with the second control point. This also illustrates why NURBS are a powerful tool for designers and users of CAD software, as control points and weights can be manipulated until the desired curve or surface shape is obtained.

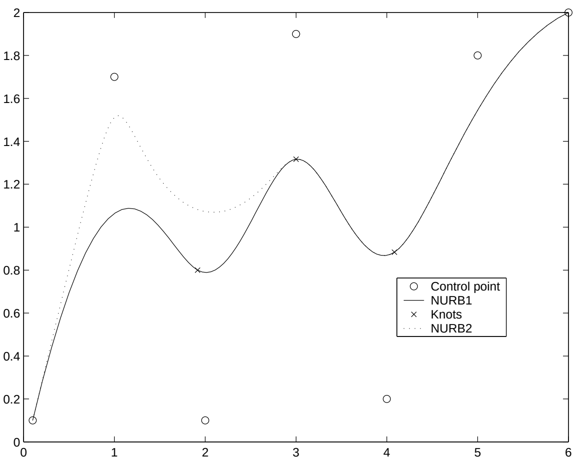

Figure 2.1: The effect of increasing a control point's weight.

The theory of NURBS also has links with projective geometry [21]. NURBS have not been directly utilised in this thesis, but a study of NURBS motivated the work of chapter 3.

2.8 Classical non-linear optimization techniques

The solve rational approximation problems described previously, we need to find a set of parameters that minimise a multi-dimensional non-linear error surface. This is a typical example of the kind of problems that can solved with the use of classical optimisation methods. A general optimisation problem requires the minimisation of a function $f : \mathbb{R}^n \to \mathbb{R}$, called the *objective function*, with respect to a set of parameters $\mathbf{a} \in \mathbb{R}^n$. An *unconstrained* optimisation problem is one where there are no restrictions on the values of the parameter vector $\mathbf{a}$. If we are required to minimise $f(\mathbf{a})$ subject to the restriction that

$$\mathbf{a} \in \ \Omega,$$

where $\Omega \subset \mathbb{R}^n$, then the problem is referred to as a *constrained* optimisation problem, and the set Ω is termed the *feasible* set or *constraint* set. In this thesis we will mainly be concerned with unconstrained approximation problems, although we will look at constrained problems when we wish to fit rational approximations that are without poles in the approximation range. There are a large number of optimisation methods available, with many being subtle variants of others [50], [15]. We now describe some of the more commonly used optimisation methods that we will use in later chapters. The descriptions for the Newton, Gauss-Newton, and Levenberg-Marquardt methods presented here are summarized versions of the explanations given in [15], using a very similar notation.

2.8.1 Newton's method

We present a summarized version of the description of Newton's method as given in [15]. Suppose we are given a n-dimensional objective function $h(\mathbf{a})$ which we wish to minimise with respect to the parameters $\mathbf{a} = (a_1, \ldots, a_n)$. Provided that $h(\mathbf{a})$ has continuous first and second derivatives, we can obtain a Taylor series expansion of h about an arbitrary point $\mathbf{a}^{(k)}$. Neglecting terms of order three and above, this expansion about $\mathbf{a}^{(k)}$ will be denoted by $q(\mathbf{a})$ and is given by

$$q(\mathbf{a}) = h(\mathbf{a}^k) + (\mathbf{a} - \mathbf{a}^k)^T \nabla h(\mathbf{a}^k) + \frac{1}{2}(\mathbf{a} - \mathbf{a}^k)^T H(\mathbf{a}^k)(\mathbf{a} - \mathbf{a}^k), \tag{2.33}$$

where $H(\mathbf{a}^k)$ is the *Hessian matrix* of h at $\mathbf{a}^k$ defined by its i, jth element

$$H_{ij}(\mathbf{a}^{(k)}) = \frac{\partial^2 h}{\partial a_i \partial a_j}(\mathbf{a}^{(k)}), \tag{2.34}$$

and $\nabla h(\mathbf{a}^{(k)}) \in \mathbb{R}^n$ is the *gradient vector* having ith component

$$(\nabla h(\mathbf{a}^{(k)}))_i = \frac{\partial h}{\partial a_i}. \tag{2.35}$$

The function $q(\mathbf{a})$ provides a quadratic approximation to the objective function h in the neighbourhood of $\mathbf{a}^{(k)}$ and has the same first and second derivatives as h at this point. The principle behind the Newton method is to then minimize the approximation $q(\mathbf{a})$ instead of the objective function itself, and then use this minimum as a new point at which to construct another Taylor approximation. The process then continues iteratively until convergence of the $\mathbf{a}^{(k)}$ occurs (although convergence is not guaranteed), at which point a minimum of h has been obtained.

The function $q(\mathbf{a})$ is quadratic and has a unique stationary point at the value $\mathbf{a}_{stat}$ that satisfies

$$0 = \nabla q(\mathbf{a}_{stat}). \tag{2.36}$$

The function $\nabla q(\mathbf{a})$ is given by

$$\nabla q(\mathbf{a}) = \nabla h(\mathbf{a}^{(k)}) + H(\mathbf{a}^{(k)})(\mathbf{a} - \mathbf{a}^{(k)}), \tag{2.37}$$

and will be equal to zero at the point $\mathbf{a}_{stat}$ given by

$$\mathbf{a}_{stat} = \mathbf{a}^{(k)} - H(\mathbf{a}^{(k)})^{-1}\nabla h(\mathbf{a}^{(k)}) \tag{2.38}$$

This will be a minimum provided that the Hessian matrix at $\mathbf{a}^{(k)}$ is positive definite. Since this minimum forms the starting point of the next iteration, the process can be defined recursively as

$$\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \Delta\mathbf{a}^{(k)}, \tag{2.39}$$

where $\Delta\mathbf{a}^{(k)}$ is given by the solution to

$$H(\mathbf{a}^{(k)})\Delta\mathbf{a}^{(k)} = -\nabla h(\mathbf{a}^{(k)}), \tag{2.40}$$

and is referred to as the *update parameter* at iteration k.

The Newton method works well provided that the Hessian is positive definite. However, even if this is the case convergence cannot always be guaranteed unless the start point $\mathbf{a}^{(0)}$ is reasonably close to the true minimum of the objective function h. In spite of this, however, Newton's method is a popular method as it is proven to converge quadratically [15] when implemented with a choice of start parameter close to the solution. Despite the quadratic convergence, the Newton method can be computationally expensive due to the calculation of the Hessian matrix.

2.8.2 The Gauss-Newton method

When applying Newton's method to a nonlinear least-squares problem, the objective function we are trying to minimize is of the form

$$h(\mathbf{a}) = \sum_{l=1}^{m}(e_l(\mathbf{a}))^2 = \mathbf{e}(\mathbf{a})^T\mathbf{e}(\mathbf{a}), \tag{2.41}$$

where $\mathbf{e}(\mathbf{a}) \in \mathbb{R}^m$ is the vector of approximation errors or residuals as defined in (1.4). Application of Newton's method to solve this problem requires the calculation of the Hessian and the gradient of the objective function h. The gradient vector $\nabla h(\mathbf{a})$ (2.35) for this problem can be expressed as

$$\nabla h(\mathbf{a}) = 2J(\mathbf{a})^T\mathbf{e}(\mathbf{a}), \tag{2.42}$$

where J$(\mathbf{a})$ represents the *Jacobian matrix* of $\mathbf{e}$ evaluated at $\mathbf{a}$ and is defined by its i, jth element

$$J(\mathbf{a})_{ij} = \frac{\partial e_i}{\partial a_j}(\mathbf{a}). \tag{2.43}$$

The i, jth component of the Hessian matrix of h evaluated at $\mathbf{a}$ is given by

$$\begin{aligned} H_{ij}(\mathbf{a}) &= \frac{\partial^2 h}{\partial a_i \partial a_j}(\mathbf{a}) & (2.44)\\ &= \frac{\partial}{\partial a_i}\left(2\sum_{l=1}^{m} e_l(\mathbf{a})\frac{\partial e_l}{\partial a_j}(\mathbf{a})\right) & (2.45)\\ &= 2\sum_{l=1}^{m}\left(\frac{\partial e_l}{\partial a_i}(\mathbf{a})\frac{\partial e_l}{\partial a_j}(\mathbf{a}) + e_l(\mathbf{a})\frac{\partial^2 e_l}{\partial a_i \partial a_j}(\mathbf{a})\right). & (2.46)\end{aligned}$$

The first term on the right hand side of the last line of equation (2.44) can be seen to be the i, jth element of the matrix $2J(\mathbf{a})^T J(\mathbf{a})$, and so we can write the Hessian matrix as

$$H(\mathbf{a}) = 2(J(\mathbf{a})^T J(\mathbf{a}) + S(\mathbf{a})), \tag{2.47}$$

where $S(\mathbf{a})$ is the matrix with i, jth element

$$S_{ij}(\mathbf{a}) = e_i(\mathbf{a})\frac{\partial^2 e_i}{\partial a_i \partial a_j}(\mathbf{a}). \tag{2.48}$$

With the Hessian and gradient calculated, the application of Newton's method (2.39) to the problem (2.41) is given by

$$\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \Delta\mathbf{a}^{(k)}, \tag{2.49}$$

where the update parameter is given by the solution to

$$(J(\mathbf{a}^{(k)})^T J(\mathbf{a}^{(k)}) + S(\mathbf{a}^{(k)}))\Delta\mathbf{a}^{(k)} = -2J(\mathbf{a}^{(k)})^T \mathbf{e}(\mathbf{a}^{(k)}). \tag{2.50}$$

When the objective function h has low curvature around $\mathbf{a}^{(k)}$, its second partial derivatives that form the elements of the matrix $S(\mathbf{a})$ are often very small and so can be ignored. When the matrix $S(\mathbf{a})$ is omitted from the calculation of the update parameter, equation (2.50) reduces to

$$J(\mathbf{a}^{(k)})^T J(\mathbf{a}^{(k)})\Delta\mathbf{a}^{(k)} = -2J(\mathbf{a}^{(k)})^T \mathbf{e}(\mathbf{a}^{(k)}), \tag{2.51}$$

which are the normal equations for the solution to the overdetermined system

$$J(\mathbf{a}^{(k)})\Delta\mathbf{a}^{(k)} = -\mathbf{e}(\mathbf{a}^{(k)}). \tag{2.52}$$

The method described above is referred to as the *Gauss-Newton method*, and is a popular choice of algorithm for solving nonlinear least-squares problems. A nice feature of the Gauss-Newton method is that it only requires knowledge of the first derivatives of the residual vector $\mathbf{e}(\mathbf{a})$, thus avoiding the potentially expensive calculation of the Hessian. The method works well when the objective function $h(\mathbf{a})$ has low curvature (and hence small second derivatives), and as a result the matrix $J(\mathbf{a})^T J(\mathbf{a})$ is a good approximation to the Hessian matrix. When the objective function exhibits high curvature, this approximation is less good and the Gauss-Newton method may fail to converge. In this case better results may be obtained using Newton's method. When it does converge, the order of convergence of the Gauss-Newton method is linear. As with the Newton method, the Gauss-Newton method is not guaranteed to work well for choices of start parameter that are not close to the true solution. In regions of high curvature, it is not guaranteed to converge, even if it is arbitrarily close to a local minimum.

An important point for consideration is the positive definiteness of the matrix $J(\mathbf{a})^T J(\mathbf{a})$. If this is not the case then even if the algorithm does converge it may not converge to a local minimum. This is also true of the Newton method if the Hessian is not positive definite. This problem can be overcome with the use of the Levenberg-Marquardt algorithm which is described below.

2.8.3 The Levenberg-Marquardt algorithm

Application of either the Newton or Gauss-Newton methods generates at each iteration k an update parameter $\Delta\mathbf{a}^{(k)}$ from which we obtain the next parameter

$$\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \Delta\mathbf{a}^{(k)}.$$

When close to the minimum, the *search direction* given by

$$\mathbf{d}^{(k)} = \mathbf{a}^{(k+1)} - \mathbf{a}^{(k)},$$

is such that the objective function decreases at each iteration

$$f(\mathbf{a}^{(k+1)}) < f(\mathbf{a}^{(k)}),$$

and when this is the case, the search direction is said to point in a *descent direction.* When the matrix $J(\mathbf{a})^T J(\mathbf{a})$ is not positive definite, the search direction is not guaranteed to point in a descent direction. To ensure a descent direction with each iteration it is possible to calculate the update parameter from the equation

$$(J(\mathbf{a}^{(k)})^T J(\mathbf{a}^{(k)}) + \mu_k I)\Delta\mathbf{a}^{(k)} = -J(\mathbf{a}^{(k)})^T \mathbf{e}(\mathbf{a}^{(k)}), \tag{2.53}$$

where the parameter μ_k is chosen to be such that the matrix

$$A_k = J(\mathbf{a}^{(k)})^T J(\mathbf{a}^{(k)}) + \mu_k I, \tag{2.54}$$

is now positive definite. We can be certain of the positive definiteness of A_k for sufficiently large μ_k for the following reason. We will denote the set of eigenvalues of $J^T J$ by

$$\lambda_1, \ldots, \lambda_m,$$

which may or may not be distinct. By definition, if $J^T J$ is not positive definite, then at least one of these eigenvectors is not positive. The eigenvalues of the matrix A_k will be

$$\lambda_1 + \mu_k, \ldots, \lambda_m + \mu_k,$$

due to the fact that

$$\begin{aligned} A_k c_i &= (J^T J + \mu_k I) c_i \\ &= J^T J c_i + \mu_k I c_i \\ &= \lambda_i c_i + \mu_k c_i \\ &= (\lambda_i + \mu_k) c_i, \end{aligned}$$

where the vector c_i is the eigenvector with corresponding eigenvalue λ_i. Thus if the value of μ_k is large enough (larger than the smallest eigenvalue λ_i), the eigenvalues of A_k will be positive and therefore A_k will be positive definite.

There are a number of heuristic algorithms for choosing the value of μ_k. Usually the value is increased whenever the value of the error increases at a particular iteration. In such cases, the value of μ_k is increased by a factor until there is a decrease in the error. Similarly, when the error is decreasing steadily, then the value of μ_k is decreased at each iteration. A method similar to this was originally proposed by Marquardt [36].

The method described above is referred to as the *Levenberg-Marquardt algorithm* and is described in greater detail in [41]. A similar approach may be used for the Newton method to ensure positive definiteness of the Hessian by addition of a multiple of the identity matrix. The Levenberg-Marquardt method is more robust than the Gauss-Newton method and is not as reliant on a good choice of start parameter in order for convergence to occur. In addition, there are some variations of the Levenberg-Marquardt algorithm that have been shown to be globally convergent [42].

Application of the Gauss-Newton method to generalised rational approximation

We have defined a generalised rational function in section 1.7 as

$$R_m^n(\mathbf{p},\mathbf{q},x) = \frac{P_n(\mathbf{p},x)}{Q_m(\mathbf{q},x)} = \frac{\sum_{i=0}^{n} p_i\phi_i(x)}{\sum_{j=0}^{m} q_j\psi_j(x)}. \tag{2.55}$$

and the rational approximation residual vector $\mathbf{e}(\mathbf{p},\mathbf{q})$ as the column vector that has ith element

$$e_i(\mathbf{p},\mathbf{q}) = f_i - R_m^n(\mathbf{p},\mathbf{q},x_i). \tag{2.56}$$

Evaluation of the partial derivatives of $\mathbf{e}$ with respect to the approximation parameters then gives required elements of the Jacobian matrix (2.43) and these are given by

$$\begin{aligned} \frac{\partial e_i}{\partial p_j} &= \frac{\phi_j(x_i)}{Q_m(\mathbf{q},x_i)} & j &= 0,\dots,n \\ \frac{\partial e_i}{\partial q_k} &= \psi_k(x_i)\frac{P_n(\mathbf{p},x_i)}{(Q_m(\mathbf{q},x_i))^2} & k &= 0,\dots,m. \end{aligned} \tag{2.57}$$

If we define the vector of approximation parameter estimates at iteration k as

$$\mathbf{a}^{(k)} = (p_0^{(k)},\dots,p_n^{(k)},q_0^{(k)},\dots,q_m^{(k)})^T, \tag{2.58}$$

then we can define the Jacobian matrix $J \in \mathbb{R}^{N\times(m+n+2)}$ as the matrix with i,jth element

$$\begin{aligned} J_{ij} &= \frac{\phi_{j-1}(x_i)}{Q_m(\mathbf{q},x_i)} & j &= 1,\dots,n+1, \\ J_{ij} &= \psi_{j-(n+2)}(x_i)\frac{P_n(\mathbf{p},x_i)}{(Q_m(\mathbf{q},x_i))^2} & j &= n+2,\dots,m+n+2. \end{aligned} \tag{2.59}$$

With the Jacobian matrix given, we can now apply the Gauss-Newton method as described in section 2.8.2.

2.9 Summary

In this section we have presented some of the major areas of rational approximation, some of which are utilised in the rest of the thesis. Some of the described methods are not used anywhere in this thesis as they were found not to be directly applicable to the specific problem addressed by the project. These methods are included here as they were researched initially to assess suitability for the project, and also because they are major components of the vast subject of rational approximation. The Loeb algorithm and the nonlinear optimisation methods have been described in more detail as they are used extensively in the future chapters.

Chapter 3

EXTENSIONS OF THE LOEB ALGORITHM

3.1 Introduction

In this section we describe some modifications to the basic Loeb algorithm (2.2) that provide good rational approximations that have been forced to share the asymptotic behaviour exhibited by the data being approximated. In addition, we have found that this type of approximation can have good extrapolation properties. A slightly modified version of Loeb's algorithm is applied to fit rational functions with a constrained asymptotic limit, and we compare the quality of the resulting approximations to those obtained with the standard Loeb algorithm. We also introduce the semi-infinite rational spline, which is a new rational form that is capable of having different asymptotic limits as $x \to +\infty$ and $x \to -\infty$, and we show how to fit this form to discrete data with the use of the Loeb algorithm.

3.2 Asymptotically constrained approximation using Loeb's algorithm

In this section we look at the problem of approximating a set of discrete data that is known to come from a function that has a specified asymptotic limit. We consider the general problem of approximating a set of data $\{(x_i, f_i)\}_{i=1}^{m} \subset \mathbb{R}^2$, on an positive interval $[\alpha, \beta]$ where the f_i are assumed to be sampled from an unknown function $f(x)$ that has the asymptotic limit

$$\lim_{x \to +\infty} \frac{f(x)}{x^{\gamma}} = \mu \tag{3.1}$$

where $\mu \in \mathbb{R}$ and $\gamma \in \mathbb{Z}^+$ are coefficients that are known prior to approximation. Our aim is to use an approximation form that is forced to share the same asymptotic limit (3.1) as the function $f(x)$, and investigate whether this provides a better approximation than when we impose no constraints the approximant. This general problem of approximating limits of the form (3.1), was described in sections (1.8.7),(1.8.5) and (1.8.6), for the specific case of polynomial ratios, and in these sections, suitable parameter constraints for fixing their asymptotic limits were derived. In this section we apply these constraints and fit polynomials ratios to the data with the use of the Loeb algorithm. We define our degree (n, m) polynomial ratio approximation form by

$$R_m^n(\mathbf{p}, \mathbf{q}, x) = \frac{\sum_{i=0}^{n} p_i x^i}{1 + \sum_{i=1}^{m_1} q_i x^i}, \tag{3.2}$$

where we have used the parameter normalisation condition $q_0 = 1$.
We recall from Chapter 1 that a polynomial ratio will have the asymptotic limit (3.1), provided that the following constraints are satisfied

$$n - m = \gamma, \tag{3.3}$$

$$p_n = q_m \mu. \tag{3.4}$$

For the case where the limit (3.1) is defined with $\mu = 0$ (the case of modelling asymptotic decay to zero), then the only constraint we require for a suitable polynomial ratio approximant is $n < m$.
Given a polynomial ratio $R_m^n(x)$ that has degrees m, n chosen to satisfy (3.3), we can enforce the parameter constraint (3.4) directly within the definition of $R_m^n(x)$ by substituting p_n with $q_m \gamma$. Our approximating function $R_m^n(x)$ is now explicitly defined

by

$$R_m^n(\mathbf{p}, \mathbf{q}, x) = \frac{\sum_{i=0}^{n-1} p_i x^i + \mu q_m x^n}{1 + \sum_{i=1}^{m} q_i x^i}. \tag{3.5}$$

Although this results in the loss of one free approximation parameter, the function $R_m^n(\mathbf{p}, \mathbf{q}, x)$ now has exactly the same limiting behaviour as the function we are approximating.

3.2.1 Adequate representation of asymptotic behaviour by the data

We next need to consider whether or not the data sufficiently represents the asymptotic behaviour specified in (3.1). We will assume that the interval of approximation $[\alpha, \beta]$ contains data from the region where the function $f(x)$ actually starts to exhibit the asymptotic behaviour we are trying to model. If this is not the case, we would have no justifiable reason to expect that an approximant with the same limit as $f(x)$ would provide a better choice of form than any other arbitrary form. To illustrate this situation we look at figures 3.1 and 3.2 which show the same function $f(x) = tanh(x)$ on two different intervals. The interval in figure 3.1 is not an interval where the functions asymptotic behaviour starts to take effect, unlike the one shown in 3.2, where it can clearly be seen that the function values start to approach the asymptotic limit as $x \to \infty$.

3.2.2 Approximation of the data

Assuming that we have fixed the parameters of the approximant $R(x)$ as described previously, we can now apply the Loeb algorithm and fit $R(x)$ to the data. The Loeb algorithm is described in detail in section 2.2, but for this constrained problem, we need to modify the observation matrix slightly. We recall that for the standard Loeb algorithm we need to minimise the quantity (2.6), which is done by solving the least-

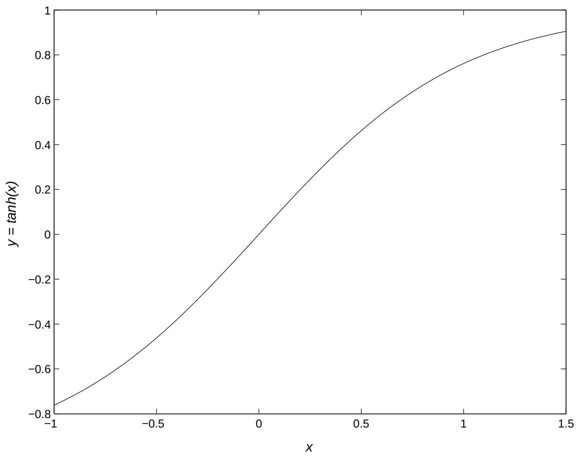

Figure 3.1: $y = tanh(x)$ on the interval $[-1, 1.5]$

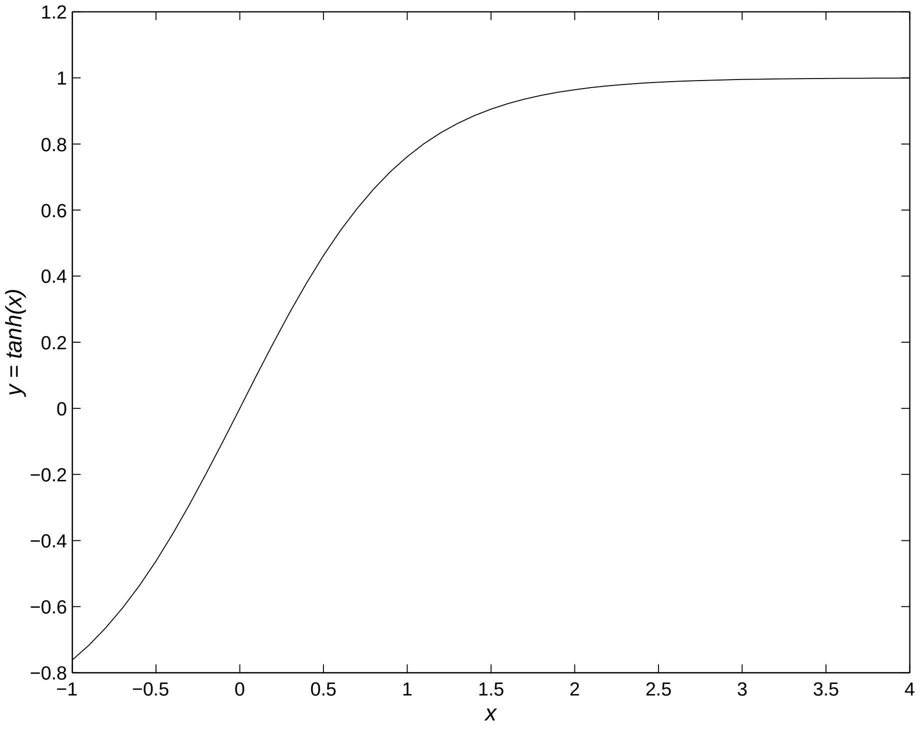

Figure 3.2: $y = tanh(x)$ on the interval $[-1, 4]$

squares system (2.7). For the constrained problem described here, with the rational form (3.5), the ℓ_2 error norm at iteration k is given by

$$\sum_{i=1}^{N} w_i^{(k)} \Big(f_i + f_i \sum_{j=1}^{m-1} q_j^{(k)} x_i^j + (f_i x_i^m - \mu x_i^n) q_m^{(k)} - \sum_{j=0}^{n-1} p_j^{(k)} x_i^j \Big)^2, \tag{3.6}$$

with weight as defined in (2.5). We can then represent this weighted linear least-squares problem at iteration k in matrix form by

$$W^{(k)}\mathbf{f} = WC\mathbf{d}^{(k)}, \tag{3.7}$$

where the observation matrix $C \in \mathbb{R}^{N\times(m+n)}$ is defined by its i, jth element

$$C_{ij} = \begin{cases} x_i^{j-1} & j = 1, \ldots, n \\ -f_i x_i^{(j-n)} & j = n+1, \ldots, m+n-1 \\ \mu x_i^{j-m} - f_i x_i^{j-n} & j = n+m \end{cases} .$$

The weighting matrix $W^{(k)} \in \mathbb{R}^{N\times N}$ is defined in equation (2.10), and

$$\mathbf{d}^{(k)} = (p_0^{(k)}, \ldots, p_{n-1}^{(k)}, q_1^{(k)}, \ldots, p_m^{(k)})^T \in \mathbb{R}^{m+n}$$

is the vector of approximation parameters, obtained by solving the normal equations for (3.7), or using QR factorisation.

3.2.3 Numerical results

We now present some results obtained from application of the asymptotically constrained Loeb method to some sample data sets.

Example 1. Sigmoid type functions

We took 51 uniformly spaced abscissae x_i on the interval $[-1, 4]$ at which we evaluated function values $f_i = tanh(x_i)$. These were approximated using a polynomial ratio of degree (n, m). To implement the constrained Loeb algorithm for this problem, we require the parameter constraints

$$\begin{aligned} n &= m, \\ p_n &= q_m, \end{aligned}$$

to ensure that the approximant has the same asymptotic limit as the function $tanh(x)$. We then applied this algorithm to the data with a choice of degrees $m = n = 4$, and assumed that convergence at iteration k when

$$\|\mathbf{d}^{(k)} - \mathbf{d}^{(k-1)}\|_\infty < 10^{-10}. \tag{3.8}$$

We then applied the standard Loeb algorithm to the same data using the same $(4, 4)$ polynomial ratio. A comparison of both algorithms is made in Table 3.1, and plots of the approximation errors are shown in Figure 3.3, where $\mathbf{e}$ represents the residual vector of the approximation obtained at convergence.

Table 3.1: Comparison of results from constrained and unconstrained Loeb algorithms for approximation of $tanh(x)$.

Algorithm	Iterations	$\|\mathbf{e}\|_2$	Cond(C)
Unconstrained	7	5.6029×10^{-5}	1.4077×10^{5}
Constrained	10	3.6370×10^{-4}	5.8904×10^{3}

Example 2. Approximation of $f(x)$, where $f(x) \to kx$ as $x \to \infty$

We took the same set of abscissae values as in the previous example, and evaluated the set of function values $f_i = f(x_i)$ this time using the function

$$f(x) = \frac{x}{2(1 + e^{-x})}.$$

This function behaves like $\frac{1}{2}x$ as $x \to \infty$, and so for a degree (n, m) polynomial ratio approximation to behave similarly we need to set

$$\begin{aligned} n &= m - 1, \\ p_n &= \tfrac{1}{2}q_m. \end{aligned}$$

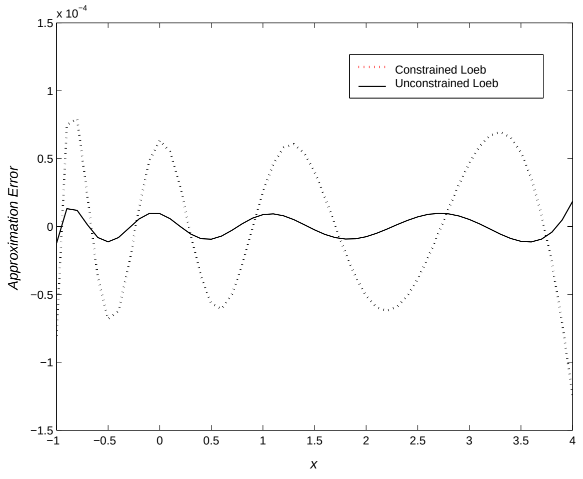

Figure 3.3: Error of degree $(4, 4)$ polynomial ratio approximations to $tanh(x)$ obtained with constrained and unconstrained Loeb algorithms

As before, we applied both the constrained and the unconstrained Loeb algorithms to this data using a degree $(5, 4)$ polynomial ratio. The results for both of the resulting approximations are shown in Table 3.2, and a plot of the different error curves is displayed in figure 3.4.

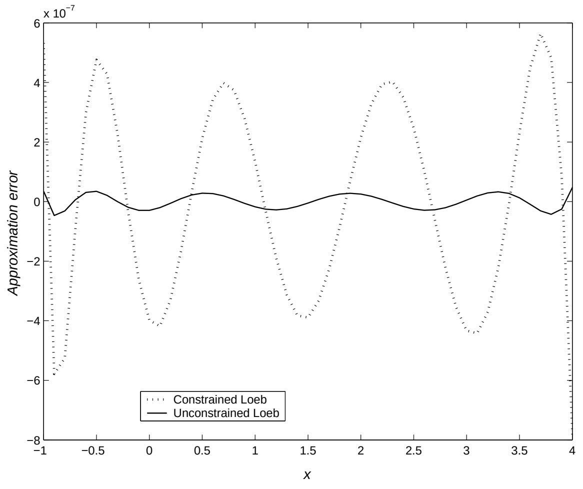

Figure 3.4: Approximation error of degree $(5, 4)$ polynomial ratios fitted using constrained and unconstrained Loeb algorithms

Table 3.2: Comparison of results from constrained and unconstrained Loeb algorithms for approximation of $0.5x(1+e^{-x})^{-1}$.

Algorithm	Iterations	$\lVert\mathbf{e}\rVert_2$	Cond(C)
Unconstrained	6	1.7162×10^{-7}	4.4525×10^{7}
Constrained	7	2.4483×10^{-6}	8.8900×10^{5}

3.2.4 Extrapolation of the data

We can see from the previous examples that the unconstrained algorithm provides a better approximation than the constrained version, which we might naturally expect as the constrained form loses one free parameter in order to satisfy (3.4). We also note that the condition number of the observation matrix for the constrained version is significantly smaller, which we would again expect to be the case through having one less column. The constrained algorithm was specifically applied in an attempt to improve approximations on the approximation interval itself, but our results show that the algorithm is unsuccessful for this purpose. However, despite the apparent lack of improvement on the approximation interval itself, we did make an interesting observation on the behaviour of the approximations outside the interval. We found that the approximation obtained using the constrained version is far superior to that of the standard Loeb method when it comes to extrapolation outside the interval (in the direction of the asymptotic limit we have modelled). This is illustrated in figure 3.5 which shows the extrapolation error of the approximations obtained from both algorithms, plotted on an interval ten times larger than the approximation interval. We obtain a similar improvement in the extrapolation error for the second example 3.2.3, however, it is not as good as for the constant limit asymptote. A proposed reason for this is functions like that in the second example, will generally have an

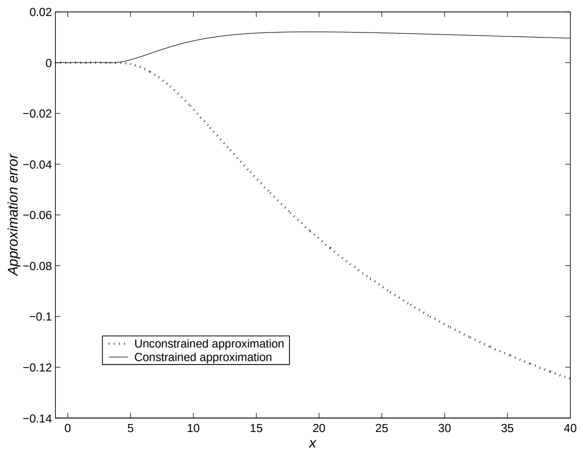

Figure 3.5: Extrapolation error of degree $(4, 4)$ polynomial ratio approximations to $tanh(x)$

asymptote of the form

$$y = a + bx,$$

where a and b are real constants. Our algorithm constrains the asymptotic behaviour, and this fixes the gradient b of the asymptote (3.2.4), but we have not specified any constraint that will provide the correct value of the intercept a. In this case we would expect our extrapolation error to converge to the value of the intercept a. This is illustrated in figure 3.6 which shows the extrapolation error of both approximations over a much larger interval. With the constant asymptote example, we do not have the same problem, and so our constrained rational form has exactly the same limit as the function being approximated.

When comparing the differences in asymptotic limits between the approximations obtained from the two algorithms, we have always observed that the extrapolation residuals from one algorithm are opposite in sign to those of the other, although we are unclear as to exactly why this is so and cannot provide a theoretical reason for it.

In summary, we have found that the constrained algorithm yields a better conditioned observation matrix, and generates good approximations that have much smaller ex-

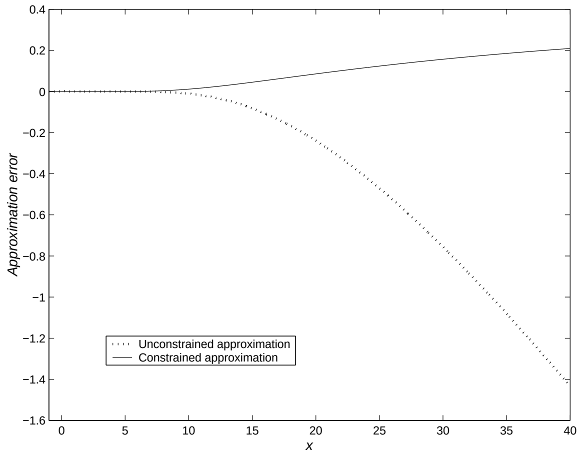

Figure 3.6: Extrapolation error of degree $(5, 4)$ polynomial ratio approximations to $0.5x(1 + e^{-x})^{-1}$

trapolation errors than the unconstrained approximations. However, it has also been found that there is no notable improvement in approximation error on the approximation interval itself, and also in general it takes a slightly larger number of iterations to converge than the standard algorithm does.

3.3 Semi-infinite rational splines

We now introduce a new approximation form, motivated by the need to approximate double sided asymptotic limits as discussed in section 1.8.8. Consider the approximation problem described in section 1.8.8 where we wish to approximate a set of data $\{(x_i, f_i)\}_{i=1}^m \subset \mathbb{R}^2$, where the f_i come from a function $f(x)$ having two specified asymptotic limits

$$\lim_{x\to-\infty} \frac{f(x)}{x^{\alpha_1}} = \mu_1 \tag{3.9}$$

$$\lim_{x\to+\infty} \frac{f(x)}{x^{\alpha_2}} = \mu_2 \tag{3.10}$$

where $\mu_1, \mu_2 \in \mathbb{R}$ and $\alpha_1, \alpha_2 \in \mathbb{N}$.

As in the previous section we wish to approximate this data with an approximant that has the same asymptotic limits (3.9),(3.10). A polynomial ratio with suitably numerator and denominator degrees can be forced to share one of these asymptotic limits, but it will rarely be able to have both. The exceptions will be for cases where both asymptotic limits are zero ($\mu_1 = \mu_2 = 0$), or are constants of equal value ($\mu_1 = \mu_2 \neq 0, \alpha_1 = \alpha_2 = 0$).

To allow the two different asymptotic limits (3.9),(3.10) to be modelled simultaneously, we consider the following piecewise rational function defined by

$$R^n_{m_1,m_2}(\mathbf{p},\mathbf{b},\mathbf{c},x) = \begin{cases} \dfrac{P(\mathbf{p},x)}{B(\mathbf{b},x)} = \dfrac{\sum_{i=0}^{n} p_i x^i}{1+\sum_{i=1}^{m_1} b_i x^i} = R_-(\mathbf{p},\mathbf{b},x) & \text{for } x \leq 0 \\ \dfrac{P(\mathbf{p},x)}{C(\mathbf{c},x)} = \dfrac{\sum_{i=0}^{n} p_i x^i}{1+\sum_{i=1}^{m_2} c_i x^i} = R_+(\mathbf{p},\mathbf{c},x) & \text{for } x \geq 0 \end{cases} \tag{3.11}$$

where $\mathbf{p} \in \mathbb{R}^{(n+1)}, \mathbf{b} \in \mathbb{R}^{m_1}, \mathbf{c} \in \mathbb{R}^{m_2}$ are vectors of approximation parameters defined in the usual way. With an appropriate choice of n, m_1, m_2, (3.11) provides a function that can potentially model both of the asymptotic limits we require. The piecewise rational functions (3.11) are defined in a way that ensures that $R^n_{m_1,m_2}(\mathbf{p},\mathbf{b},\mathbf{c},x)$ is C^0 continuous across the value $x = 0$ which behaves like a knot does in a spline function. We will refer to functions of the form (3.11) as *semi-infinite rational splines* (SIRS). We mention at this point that we could have chosen to define the SIRS with a fixed denominator, and variable numerator. The reason for not doing so is that by considering two denominator functions, we have more chance that our rational approximation will have no poles. That is to say that it is possible for the function $R_-(\mathbf{p},\mathbf{b},x)$ to have real poles (if there are any) for $x > 0$ and $R_+(\mathbf{p},\mathbf{c},x)$ to have real poles for $x < 0$. In such cases, the resulting SIRS approximation will have no poles anywhere on the real line. It is important to mention also that by the same argument, it is possible that we are allowing twice as many poles to be present in the approximation by considering two seperate denominators, but we feel that the variable denominator approach is favourable because of the potential for pole free approximations to be fitted.

3.3.1 Continuity conditions at the knot

In an analogous way to splines, we now look at the conditions required for a SIRS to be continuous at the knot at $x = 0$. As a consequence of its definition, the SIRS already has C^0 continuity at $x = 0$ as

$$R^n_{m_1,m_2}(\mathbf{p}, \mathbf{b}, \mathbf{c}, 0) = p_0, \tag{3.12}$$

for all possible values of $\mathbf{q}, \mathbf{b}, \mathbf{c}$.
To obtain conditions for C^1 continuity, we first evaluate the derivatives of $R_-(x)$ and $R_+(x)$ with respect to x, and these are given by

$$R'_-(x) = \frac{P'(x)}{B(x)} - \frac{P(x)}{B(x)^2}B'(x), \tag{3.13}$$

$$R'_+(x) = \frac{P'(x)}{C(x)} - \frac{P(x)}{C(x)^2}C'(x). \tag{3.14}$$

Evaluating these derivatives at the knot gives

$$R'_-(0) = p_1 - p_0 b_1, \tag{3.15}$$

$$R'_+(0) = p_1 - p_0 c_1, \tag{3.16}$$

which are equal (for non-zero p_0) provided that

$$b_1 = c_1, \tag{3.17}$$

and so we have C^1 continuity at $x = 0$ if this constraint is satisfied. In the special case of $p_0 = 0$, the SIRS is C^1 continuous for all possible choices of $\mathbf{b}, \mathbf{c}$.
Differentiating the functions (3.13),(3.14) again gives the second derivatives as

$$R''_-(x) \quad = \quad \frac{P''(x)}{B(x)} - \frac{P'(x)}{B(x)^2}B'(x) - \frac{P(x)}{B(x)^2}B''(x)$$

$$- B'(x)\left(\frac{P'(x)}{B(x)^2} - \frac{2P(x)}{B(x)^3}B'(x)\right) \tag{3.18}$$

$$\begin{aligned} R''_+(x) &= \frac{P''(x)}{C(x)} - \frac{P'(x)}{C(x)^2}C'(x) - \frac{P(x)}{C(x)^2}C''(x) \\ &\quad -C'(x)\left(\frac{P'(x)}{C(x)^2} - \frac{2P(x)}{C(x)^3}C'(x)\right), \end{aligned} \tag{3.19}$$

with values at the knot given by

$$R''_-(0) = p_2 - p_1b_1 - p_0b_2 - b_1(p_1 - 2p_0b_1), \tag{3.20}$$

$$R''_+(0) = p_2 - p_1c_1 - p_0c_2 - c_1(p_1 - 2p_0c_1). \tag{3.21}$$

If we assume that the conditions (3.17) for C^1 continuity hold, the second derivatives at the knot will be equal (for non-zero p_0) provided that

$$b_2 = c_2, \tag{3.22}$$

and hence the SIRS will be C^2 continuous at $x = 0$ if this constraint is satisfied. We could go on to obtain higher order continuity conditions, but will restrict ourselves to the study of SIRS that are C^2 continuous at the knot, as we feel it is sufficient for our requirements.

3.3.2 Satisfying the continuity requirements

In a similar manner to that of section 3.2, we can explicitly satisfy the continuity constraints (3.17) and (3.22) by substituting the parameters c_1 and c_2 with b_1 and b_2 respectively. We now redefine the SIRS accordingly as

$$R^n_{m_1,m_2}(\mathbf{d}, x) = \begin{cases} \dfrac{\sum_{i=0}^{n} p_i x^i}{1 + \sum_{i=1}^{m_1} b_i x^i} = R_-(x) & \text{for } x \le 0 \\ \dfrac{\sum_{i=0}^{n} p_i x^i}{1 + b_1 x + b_2 x^2 + \sum_{i=3}^{m_2} c_i x^i} = R_+(x) & \text{for } x \ge 0 \end{cases}, \tag{3.23}$$

where

$$\mathbf{d} = (p_0, \ldots, p_n, b_1, \ldots, b_{m_1}, c_3, \ldots, c_{m_2})^T$$

is the vector of combined approximation parameters. Although we have lost two approximation parameters c_1 and c_2, the function (3.23) is now C^2 continuous at $x = 0$ for all values $\mathbf{d}$, and is capable of modelling two different asymptotic limits as $x \to +\infty$ and $x \to -\infty$.

Up to this point we have considered the knot at $x = 0$ to be fixed, the reason being that the monomial basis functions x^i evaluated at this point are zero ($i \neq 0$) which results in the very simple continuity constraints we have derived. Had we used different set of basis functions, such as Chebyshev polynomials, these constraints would be significantly more complicated, since for an even integer n, the nth Chebyshev basis function $T_n(0) \neq 0$. However, the use of monomial basis functions is likely to result in numerical instability for approximation intervals far away from this knot at $x = 0$. Another reason for consideration of a variable knot value is that the SIRS reduces to a standard polynomial ratio if the knot value itself does not lie in the interval of approximation. In order to avoid these problems, and in an attempt to provide more flexibility, we consider approximating with a SIRS defined as a function of the transformed variable

$$u = x - \lambda. \tag{3.24}$$

In this way we have obtained a new shifted SIRS with knot at $x = \lambda$, ($u = 0$), defined as a polynomial ratio in powers of u rather than x. $R^n_{m_1,m_2}(\mathbf{p}, \mathbf{b}, \mathbf{c}, u)$ then provides us with an approximation form that has exactly the same constraints for continuity at the knot, as those derived for $R^n_{m_1,m_2}(\mathbf{p}, \mathbf{b}, \mathbf{c}, x)$. We now describe an effective method of fitting the SIRS to a discrete set of data.

3.3.3 Least-squares SIRS approximation using Loeb's algorithm

Now that we have defined a new approximation form (3.23) that has the required level of continuity and asymptotic limits, we need a method of fitting it to data. Due to its ease of implementation, we consider fitting least-squares SIRS approximations using the Loeb algorithm (2.2). One of the reasons for this choice is that it can be easily applied to the SIRS due to the way that the required continuity conditions are satisfied immediately from its definition (3.23). The Loeb algorithm has already been discussed and we now describe how to implement it for the SIRS. We are approximating the data set $\{(x_i, f_i)\}_{i=1}^{m} \subset \mathbb{R}^2$, and firstly assume that the data points have been ordered with respect to ascending values of the abscissae x_i. To begin with, we choose the knot λ to be the midpoint of the approximation interval, and define the integer t to be the largest integer for which $u_t \leq 0$, where $u_t = x_t - \lambda$. Alternatively we could take λ to be the median value of the abscissae x_i in the case of non-uniformly spaced abscissae. It seems sensible to ensure that we have an approximately equal number of abscissae on either side of the knot, as we are approximating simultaneously with $R_-(x)$ and $R_+(x)$. If this is the case we will approximate roughly half the data with $R_-(x)$ and half with $R_+(x)$, and so this should prevent either one of these functions from dominating the approximation. We will discuss other factors to bear in mind when choosing the knot in a later section.

As we are approximating with $R_-(x)$ and $R_+(x)$ simultaneously, the observation matrix A, needed to implement Loeb's algorithm, is defined as having i, jth element

A_{ij} given by

$$A_{ij} = \begin{cases} u_i^{j-1} & \begin{cases} i = 1, \dots, N, \\ j = 1, \dots, n+1 \end{cases} \\ -f_i u_i^{j-n-1} & \begin{cases} i = 1, \dots, N, \\ j = n+2, n+3 \end{cases} \\ -f_i u_i^{j-n-1} & \begin{cases} i = 1, \dots, t, \\ j = n+3, \dots, n+m_1+1 \end{cases} \\ 0 & \begin{cases} i = 1, \dots, t, \\ j = n+m_1+2, \dots, n+m_1+m_2-1 \end{cases} \\ 0 & \begin{cases} i = t+1, \dots, N, \\ j = n+3, \dots, n+m_1+1 \end{cases} \\ -f_i u_i^{j-(n+m_1+1)} & \begin{cases} i = t+1, \dots, N, \\ j = n+m_1+2, \dots, n+m_1+m_2-1 \end{cases} \end{cases} \tag{3.25}$$

Once the observation matrix has been formed, we now require the Loeb weight vector. This weight vector at iteration k will be denoted by $\mathbf{w}^{(k)} \in \mathbb{R}^N$, and is defined as the vector having ith element

$$w_i^k = \frac{1}{B(\mathbf{b}^{(k)}, u_i)} \text{ for } i = 1, \dots, t, \tag{3.26}$$

$$w_i^k = \frac{1}{C(\mathbf{c}^{(k)}, u_i)} \text{ for } i = t+1, \dots, N \tag{3.27}$$

where $\mathbf{b}^{(k)}, \mathbf{c}^{(k)}$ are the denominator parameters obtained at iteration k. Given the weight vector $\mathbf{w}^{(k)}$, the modified SIRS Loeb algorithm proceeds by finding at the $(k+1)$th iteration the least-squares solution vector $\mathbf{d}^{(k+1)}$ of the equation

$$D^{w^k} A \mathbf{d}^{(k+1)} = D^{w^k} \mathbf{f} \tag{3.28}$$

where the matrix $D^{w^k} \in \mathbb{R}^{N \times N}$ is the diagonal matrix with elements

$$D_{ii}^{w^k} = w_i^k, \qquad i = 1, \dots, N. \tag{3.29}$$

The algorithm is initialised by setting D^0 as the $N \times N$ identity matrix, and then applied until the solution vectors $\mathbf{d}^{(k+1)}$ converge. If the knot is coincident with a data point then we have to approximate using both spline functions at the knot. This leads to the same abscissa value $x_i = \lambda$ having 2 rows in the observation matrix. If we do this with for standard linear least squares methods, then the algorithm breaks down as the observation matrix is not of full rank. We can do this with the SIRS as we are fitting the same point twice but with different sets of approximation parameters. We now present some results from the application of this algorithm.

Example 3

We will consider the approximation of the function $f(x) = 2 + tanh(x)$ on the interval $[-4, 4]$. This is exactly the type of function suitable for SIRS approximation as it has asymptotic limits

$$\lim_{x \to -\infty} f(x) = 1, \tag{3.30}$$

$$\lim_{x \to +\infty} f(x) = 3. \tag{3.31}$$

We chose the abscissae vector $\mathbf{x}$ to consist of 50 uniformly spaced points on the interval $[-4, 4]$, with corresponding function values $f_i = 2 + tanh(x_i)$.
We applied the Loeb algorithm described previously to fit a degree (5,5) SIRS to this data set, with a choice of knot $\lambda = 0$. We also fitted a standard degree (5,5) polynomial ratio to the data and used the same convergence criteria (3.8) for both methods. We compare the resulting approximations in Table 3.3 and Figure 3.7.
We can clearly see that we get fast convergence to good approximations using both forms. It also appears that the SIRS provides a superior approximation at the ends of the interval, but performs less well in the vicinity of the knot. This is highly likely to be due to the fact that the SIRS has only C^2 continuity at the knot. Also, because the SIRS has been constrained at the knot to ensure this level of continuity, the approximation is less flexible in this region.

Table 3.3: Comparison of SIRS ($\lambda = 0$) and polynomial ratio approximations to $f(x)$ fitted with Loeb's algorithm.

Approximation	Iterations	$\|\mathbf{e}\|_2$	Cond(C)
Polynomial ratio	6	5.3800×10^{-5}	1.7366×10^{6}
SIRS	7	1.0226×10^{-4}	5.6611×10^{6}

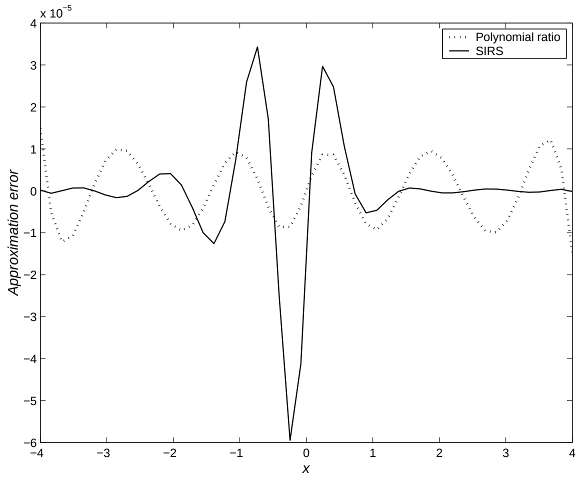

Figure 3.7: Error curves from degree (5,5) polynomial ratio and SIRS approximations to $f(x) = 2 + tan(x)$.

3.3.4 Changing the position of the knot λ

For the data given in the previous example, the approximation obtained after convergence did not contain any unwanted poles in the interval of approximation, but this is not always the case for a given choice of λ. By changing the value of the knot, we can obtain approximations that are pole-free on the range of interest as we show with the following example. In addition, should the Loeb algorithm fail to converge for a particular knot value we can modify the knot slightly and achieve convergence as the next example illustrates.

Example 4.

We now approximate $f(x) = tanh(x)$ on the same interval and number of points as in the previous example. Again we choose degree 5 numerator and denominator, and again we fix the knot at $\lambda = 0$. Using the same convergence criteria as for the previous example, the algorithm went through 250 iterations and still failed to converge, due to the oscillatory behaviour in the parameters as was described in section 2.2.2. However, approximation of the data with a choice of knot $\lambda = 0.5$, resulted in convergence after 21 iterations, to a very good approximation. Table 3.4 shows how the speed of convergence and quality of approximation varies according to the value of the knot for this particular problem.

Table 3.4: The effect of variation of λ on the SIRS approximation.

λ	Iterations	$\|\mathbf{e}\|_2$	Cond(C)
0.0	Failed	-	-
0.2	Failed	-	-
0.4	63	1.0396×10^{-4}	1.7895×10^{6}
0.6	15	7.1490×10^{-5}	2.3837×10^{6}
0.8	13	4.7651×10^{-5}	4.1835×10^{6}
1.2	13	7.5587×10^{-5}	2.0446×10^{6}
1.5	24	8.5695×10^{-5}	4.8264×10^{7}

3.4 Variable numerator SIRS

We now consider another form of SIRS that incorporates a separate numerator for each rational spline, providing the approximation form with greater flexibility. We define this new approximation form as

$$R^{n_1,n_2}_{m_1,m_2}(\mathbf{g},\mathbf{h},\mathbf{b},\mathbf{c},u) = \begin{cases} \dfrac{G(\mathbf{g},u)}{B(\mathbf{b},u)} = \dfrac{\sum_{i=0}^{n_1} g_i u^i}{1+\sum_{i=1}^{m_1} b_i u^i} = R_-(\mathbf{g},\mathbf{b},u) & \text{for } u \le 0 \\ \dfrac{H(\mathbf{h},u)}{C(\mathbf{c},u)} = \dfrac{\sum_{i=0}^{n_2} h_i u^i}{1+\sum_{i=1}^{m_2} c_i u^i} = R_+(\mathbf{h},\mathbf{c},u) & \text{for } u \ge 0 \end{cases}, \tag{3.32}$$

where $u = x - \lambda$ as before.

This type of SIRS is likely to provide greater flexibility due to the extra parameters we have introduced. The parameter constraints needed to impose C^2 continuity at $u = 0$ are very similar to those for the single numerator SIRS described previously. It is evident that the function (3.32) will be C^0 continuous at the knot provided that

$$g_0 = h_0. \tag{3.33}$$

The first derivatives of each component of (3.32) are given by

$$R'_-(u) = \frac{G'(u)}{B(u)} - \frac{G(u)}{B(u)^2}B'(u), \tag{3.34}$$

$$R'_+(u) = \frac{H'(u)}{C(u)} - \frac{H(u)}{C(u)^2}C'(u). \tag{3.35}$$

and evaluated at $u = 0$ take the values

$$R'_-(0) = g_1 - g_0 b_1, \tag{3.36}$$

$$R'_+(0) = h_1 - h_0 c_1. \tag{3.37}$$

Provided that g_0 and h_0 are non-zero, and assuming the C^0 continuity constraint (3.33) is satisfied, the variable numerator SIRS will be C^1 at the knot if

$$g_1 = h_1 - g_0(c_1 - b_1). \tag{3.38}$$

A simple set of constraints that can be used to ensure this equation is satisfied is to set

$$g_1 = h_1, \tag{3.39}$$

$$c_1 = b_1. \tag{3.40}$$

It is clear that the use of these constraints is more restrictive than (3.38) and that their use will result in us considering only a subset of all possible C^1 SIRS for approximation purposes. However, these constraints are much easier to enforce and once again they can be implemented within the definition of the approximation form itself, allowing us to avoid the use of complex constrained optimisation techniques.

For C^2 continuity we need the second derivatives of the SIRS and these are given by

$$\begin{aligned} R''_-(u) &= \frac{G''(u)}{B(u)} - \frac{G'(u)}{B(u)^2}B'(u) - \frac{G(u)}{B(u)^2}B''(u) \\ &\quad - B'(u)\left(\frac{G'(u)}{B(u)^2} - \frac{2G(u)}{B(u)^3}B'(u)\right) \end{aligned} \tag{3.41}$$

$$\begin{aligned} R''_+(u) &= \frac{H''(u)}{C(u)} - \frac{H'(u)}{C(u)^2}C'(u) - \frac{H(u)}{C(u)^2}C''(u) \\ &\quad -C'(u)\left(\frac{H'(u)}{C(u)^2} - \frac{2H(u)}{C(u)^3}C'(u)\right), \end{aligned} \tag{3.42}$$

and at $u = 0$ take the values

$$R''_-(0) = g_2 - g_1b_1 - g_0b_2 - b_1(g_1 - 2g_0b_1), \tag{3.43}$$

$$R''_+(0) = h_2 - h_1c_1 - h_0c_2 - c_1(h_1 - 2h_0c_1). \tag{3.44}$$

If we assume that the conditions (3.39),(3.33) for C^1 continuity are satisfied, we will have C^2 continuity at $u = 0$ if

$$g_2 = h_2, \tag{3.45}$$

$$b_2 = c_2. \tag{3.46}$$

As mentioned for the C^1 parameters, the set of SIRS that satisfy the constraints (3.33),(3.39) and (3.45) is only a subset of the set of all C^2 continuous SIRS, however, the constraints derived here are attractive due to their simplicity and the ease with which they can be satisfied. As before with the original SIRS, we will now impose C^2 continuity by redefining the SIRS with the constraints explicitly satisfied. The resulting SIRS is now defined as

$$R^{n_1,n_2}_{m_1,m_2}(\mathbf{d},u) = \begin{cases} \dfrac{\sum\limits_{i=0}^{n_1} g_i u^i}{1+\sum\limits_{i=1}^{m_1} b_i u^i}, & u \le 0 \\ \dfrac{g_0 + g_1 u + g_2 u^2 + \sum\limits_{i=3}^{n_2} h_i u^i}{1 + b_1 u + b_2 u^2 + \sum\limits_{i=3}^{m_2} c_i u^i}, & u \ge 0 \end{cases}, \tag{3.47}$$

where

$$\mathbf{d} = \{g_0, \ldots, g_{n_1}, h_3, \ldots, h_{n_2}, b_1, \ldots, b_{m_1}, c_3, \ldots, c_{m_2}\}^T, \tag{3.48}$$

is the vector whose elements are the remaining approximation parameters. As before this form can be easily fitted with an appropriate modification of Loeb's algorithm similar to that shown in section 3.3.3. We now present some results from the application of this method.

3.4.1 Examples of fitting the variable numerator SIRS

We have found that a variable numerator SIRS to data using the Loeb algorithm yields an observation matrix that has significantly higher condition number than for the fixed numerator SIRS and standard polynomial ratios. However, we have found that we can usually ensure a condition number of the order 10^6 or less provided that we keep to fairly low degrees of numerator and denominator. In general, we have found that we get a condition number of 10^7 or less provided that

$$\max(n_1 + m_1, n_2 + m_2) \le 8. \tag{3.49}$$

This is only a rough estimate based on our experience of repeated applications of the algorithm to approximate data that is defined on intervals that are of a reasonable size (intervals such as [-5,5] or even smaller). Over much larger intervals (such as [-10,10] for example) then we are faced with poorer conditioning.
We now present some numerical examples that illustrate the improvement in approximation quality given by variable numerator SIRS compared with standard polynomial ratios.

Example 5.

The function
$$f(x) = \frac{x}{1+e^{-x}},$$
is a good example of a function having two different asymptotic limits, and so we approximated this at 100 equally spaced abscissae on the interval [-4,4], using the SIRS $R^{4,5}_{5,4}$ with initial knot value $\lambda = 0$. Table 3.5 shows how the SIRS approximation compares with the standard polynomial ratio approximation of degree (5,4) to the same data, and Figure 3.8 compares the approximation errors. This example illustrates

Table 3.5: Comparison of $R^{4,5}_{5,4}$ ($\lambda = 0$) and degree (5,4) polynomial ratio approximations.

Approximation	Iterations	$\|\mathbf{e}\|_2$	Cond(C)
Polynomial ratio	4	3.2004×10^{-5}	6.8604×10^{5}
SIRS	6	6.9154×10^{-6}	3.0789×10^{6}

the superior quality of the approximation using the SIRS over the polynomial ratio, even though both approximations have roughly the same convergence and condition number for the observation matrix.

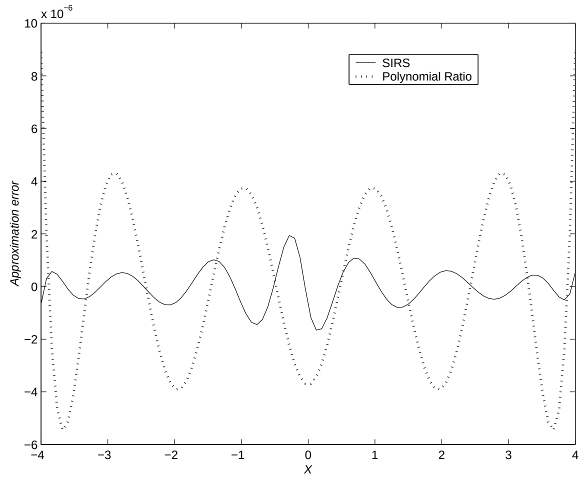

Figure 3.8: Error curves from degree (5,4) polynomial ratio and $R^{4,5}_{5,4}$ approximations to $f(x)$.

Example 6.

We approximated $f(x) = tanh(x)$ at 100 equally spaced points on $[-5, 5]$, using a degree (5,5) polynomial ratio and the SIRS $R^{5,5}_{5,5}$. With an initial knot value of $\lambda = 0$ the SIRS converged to a solution in a reasonable number of iterations. The resulting approximation had a comparable quality with the standard polynomial ratio, but had the undesirable side effect of having 2 poles within the approximation interval. We then tried to refit the data using various different values for the knot. The properties of the resulting approximations are shown in Table 3.6. The standard degree (5,5) polynomial ratio had an error norm of 2.9047×10^{-4}, converged in 6 iterations and had better conditioning than any of the SIRS knot values. This example clearly shows that we can obtain convergence for a number of different values of the knot, and that changing this value can result in better approximations, and can also result in the elimination of poles from the range. This is something that cannot be done with the standard Loeb algorithm. If the algorithm fits a pole where it is is not wanted then there is nothing that can be done about it. It may be the case that there are other

Table 3.6: The effect of variation of λ on the SIRS approximation.

λ	Iterations to convergence	$\|\mathbf{e}\|_2$	Cond(C)	Poles present in approximation range
0.0	26	2.0402×10^{-4}	2.4556×10^{6}	Yes
0.5	Failed	-	-	-
1.0	Failed	-	-	-
1.5	13	1.2838×10^{-4}	1.2123×10^{8}	No

data sets for which we cannot find a value of λ that gives a pole-free approximation, but this example illustrates the added flexibility that the SIRS has over standard polynomial ratios for approximation purposes.

3.4.2 Optimal choice for λ

The previous example illustrates how the value of λ has a huge effect on the properties of the resulting approximation, in particular convergence, approximation quality, and occurrence of poles. It has been shown that it is possible to modify the knot value in an arbitrary manner until a satisfactory approximation has been obtained, but this is a trial and error approach to data fitting. This approach may be sufficient for certain problems, but it would be desirable to find an optimal value for λ, or at least find a method that allows it to be treated as a free approximation parameter. We cannot hope to continue fitting the SIRS to data using Loeb's algorithm if we wish to use this approach, but by using the Gauss-Newton method we have an algorithm that allows us to treat the knot value as an additional parameter. However, we can still utilise a few steps of the Loeb algorithm for a specific choice of λ to provide a set of initial values for the approximation parameters with which to apply the Gauss-Newton method. Using the definition (3.47) for the SIRS, the partial derivatives of the ith residual

$$e_i = f_i - R^{n_1,n_2}_{m_1,m_2}(\mathbf{d}, u_i), \tag{3.50}$$

with respect to each of the approximation parameters are given by

$$\frac{\partial e_i}{\partial d_j} = \begin{cases} \dfrac{u_i^j}{B(\mathbf{b}, u_i)} & u \leq 0, \\ 0 & u \geq 0 \end{cases}, \tag{3.51}$$

for $j = 0, \ldots, n_1$,

$$\frac{\partial e_i}{\partial d_j} = \begin{cases} \dfrac{u_i^j}{C(\mathbf{c}, u_i)} & u \geq 0, \\ 0 & u \leq 0 \end{cases}, \tag{3.52}$$

for $j = n_1 + 1, \ldots, n_2 + n_1 - 1$,

$$\frac{\partial e_i}{\partial d_j} = \begin{cases} \dfrac{u_i^k G(\mathbf{g}, u_i)}{(B(\mathbf{b}, u_i))^2} & u \leq 0, \\ 0 & u \geq 0 \end{cases}, \tag{3.53}$$

for $j = n_1 + n_2, \ldots, n_1 + n_2 + m_1$,

$$\frac{\partial e_i}{\partial d_j} = \begin{cases} \dfrac{u_i^k H(\mathbf{h}, u_i)}{(C(\mathbf{c}, u_i))^2} & u \geq 0, \\ 0 & u \leq 0 \end{cases}, \tag{3.54}$$

for $j = n_1 + n_2 + m_1 + 1, \ldots, n_1 + n_2 + m_1 + m_2 - 2$.

Finally for the knot λ we have

$$\frac{\partial e_i}{\partial \lambda} = \begin{cases} \dfrac{G(\mathbf{g}, u_i)B'(\mathbf{b}, u_i) - G'(\mathbf{g}, u_i)B(\mathbf{b}, u_i)}{(B(\mathbf{b}, u_i))^2} & u \leq 0, \\ \\ \dfrac{H(\mathbf{g}, u_i)C'(\mathbf{b}, u_i) - H'(\mathbf{g}, u_i)C(\mathbf{b}, u_i)}{(C(\mathbf{b}, u_i))^2} & u \geq 0 \end{cases}. \tag{3.55}$$

These partial derivatives are used to form the Jacobian matrix which is then used to implement the Gauss-Newton method as described in section 2.8.2. However, care is needed in the application of the Gauss-Newton method, as if at any iteration we end up with the knot λ lying outside the approximation interval, then we will have a rank deficient Jacobian matrix due to repeated columns of zeros. We have applied the full step Gauss-Newton method (2.52) to a variety of functions, (implemented using the Loeb algorithm solution parameters for the initial Gauss-Newton parameters) using a number of different initial values for the knot, and found that the algorithm did not converge in the vast majority of cases. In many cases this was due to the knot being placed outside of the approximation interval as described above. This could be due to high curvature which will result in the matrix $J^T J$ being a poor approximation to the Hessian. This is not entirely unexpected as by allowing the knot to become a free parameter, we have introduced a much higher degree of nonlinearity in the approximation form. Alternatively, it was possible that we were unable to find a good enough start parameter for the knot, and so with this in mind, we applied the MATLAB implementation of the Levenberg-Marquardt algorithm, as it will be more likely to converge with a poor choice of start parameters. This was much more successful than the Gauss-Newton method, although convergence in many cases was very slow, on occasion taking more than 200 iterations. However, based on the application of this method to a variety of different problems we would recommend the Levenberg-Marquardt method if it is desired to optimise for the knot. We have also observed that in general, the best choice of start value for the knot parameter is the midpoint of the approximation interval.

To illustrate the performance of the fixed knot variable numerator SIRS approximations, we present some numerical results in table 3.7. We compared the SIRS approximations obtained using the Loeb algorithm with fixed knot, with approximations obtained using the Levenberg-Marquardt method optimising for the knot also. We tested a number of functions, in each case using 50 equally spaced abscissae on the approximation interval and using a degree (3,3) polynomial ratio. We use the abbreviation L-M to refer to the Levenberg-Marquardt algorithm in these results. These results clearly show that the SIRS approximation optimising for the knot value

Table 3.7: Comparison of fixed knot SIRS fitted with Loeb algorithm, with variable knot SIRS fitted with L-M algorithm.

$f(x)$	Interval	Loeb convergence	Loeb $\|\mathbf{e}\|_2$	L-M convergence	L-M $\|\mathbf{e}\|_2$
$tanh(x)$	[-2,2]	7	7.3875×10^{-4}	13	2.8395×10^{-7}
$tanh(x)$	[-3,3]	97	2.3341×10^{-2}	13	2.6575×10^{-6}
$1 + \frac{1}{(1+e^{-x})}$	[-4,4]	7	3.6937×10^{-4}	4	7.4797×10^{-8}

gives a much better approximation to that of the fixed knot case.
Using unconstrained optimisation methods, we have no way of ensuring that the value of the knot lies within the approximation interval, and, as mentioned previously, in some cases we have had the algorithm break down for this very reason. However, we have not attempted to apply the algorithm using constrained techniques, as the motivation behind the thesis is to experiment with and develop simple algorithms that are easy to apply, and in the majority of cases, we get very good approximations using the basic fixed knot, SIRS Loeb algorithm. If the resulting approximations are not sufficiently accurate, we can also treat the knot value as an additional approximation parameter and fit the SIRS using the Levenberg-Marquardt method.

3.5 Conclusion

We have seen that the use of polynomials with constrained asymptotic limit provide a useful tool for approximation, particularly for extrapolation purposes. Also the various forms of the SIRS are useful tools for approximation and extrapolation of data or functions that exhibit double sided asymptotic limits. The two approaches can be combined to provide improved extrapolation properties. In particular, we observe a very small approximation error toward the ends of the approximation interval when approximating this kind of data using an appropriate choice of the SIRS form. This approximation error is generally much smaller than that of standard polynomial ratio approximations on the same data. We also have seen that these forms can be fitted with a modification of the Loeb algorithm, which is considerably simpler than standard optimisation techniques. Furthermore, there is little loss of accuracy using the modified Loeb algorithm in place of optimisation methods in the case of a fixed knot SIRS approximations. However, in general we have found that to obtain the most accurate approximations, the best approach is to optimise for the value of the knot and fit the variable numerator SIRS using the Levenberg-Marquardt method. This method was found to be the most robust and the most accurate.

Chapter 4

ASYMPTOTIC POLYNOMIALS

4.1 Introduction

Within the field of metrology, it is quite common for there to be some type of asymptotic behaviour associated with the physical system that is being studied. Some of these types of behaviour have already been described in sections 1.8.5, 1.8.6, 1.8.7. Empirical models such as polynomials, splines and Fourier series [6, 10] are not well suited for modelling this kind of behaviour, and so in this chapter, we consider an easily implemented method to allow various classes of asymptotic behaviour to be modelled effectively. This is achieved by modifying a set of polynomial basis functions using a nonlinear weighting function designed to enable the correct type of asymptotic behaviour to be modelled. We refer to these weighted polynomials as *asymptotic polynomials*. The weight function itself is a rational function dependent on a small number of parameters that can be regarded as being fixed, or as free approximation parameters. In the latter case we describe numerically stable algorithms for approximation with asymptotic polynomials that exploit the fact that nonlinearity is introduced through nonlinear diagonal weighting matrices. We also deal with the problem of modelling variable asymptotic behaviour at $\pm\infty$ (as described in section 1.8.8) by using a piecewise continuous weight function. Finally, in section 4.4.2, we compare asymptotic polynomial and standard (Chebyshev) polynomial fits to metrology data. The vast majority of the work presented in this chapter has been published as a collaborative paper with Professors Alistair Forbes and John Mason, in the proceedings of the 5th conference on Algorithms for approximation (A4A5) which took place at Chester in the UK in July 2005.

4.2 Asymptotic polynomials

Let $\{\phi_j(x)\}_{j=0}^n$ be a set of polynomial basis functions such as Chebyshev polynomials [40], and define the weight function

$$w(x) = w(x, \mathbf{b}) = \frac{1}{(1 + s^2(x-t)^2)^{k/2}}, \quad s > 0, \quad k > 0, \tag{4.1}$$

where $\mathbf{b} = (s, t, k)^{\mathrm{T}}$. The weighting function $w(x)$ is continuous with $0 < w(x) \leq 1$ and is a rational function that has the desirable property of being pole free over the entire real line. In addition, $w(x)$ behaves like $|x|^{-k}$ as $|x| \to \infty$. The choice of this weight was inspired by some research into radial basis functions [47]. It can be seen that this function is similar to the multiquadric radial basis function

$$\phi(r) = (r^2 + c^2)^{\frac{1}{2}}, \tag{4.2}$$

where $c \in \mathbb{R} > 0$ and $r = \|x - t\|$ for some $t \in \mathbb{R}$. This function has been modified slightly with the introduction of the parameter k in order to achieve the desired type of asymptotic behaviour. We now define a modified basis function

$$\tilde{\phi}_j(x, \mathbf{b}) = w(x, \mathbf{b})\phi_j(x), \tag{4.3}$$

and then consider approximation with linear combinations

$$\tilde{\phi}(x, \mathbf{a}, \mathbf{b}) = \sum_{j=0}^{n} a_j \tilde{\phi}_j(x, \mathbf{b}), \tag{4.4}$$

where $\mathbf{a} = (a_1, \ldots a_n)^{\mathrm{T}}$. We refer to the function (4.4) as an *asymptotic polynomial*, and we refer to $\mathbf{b} = (s, t, k)^{\mathrm{T}}$ as the *auxiliary* parameters associated with the model $\tilde{\phi}(x, \mathbf{a}, \mathbf{b})$.

Each one of the auxiliary parameters has a different effect on the behaviour of the asymptotic polynomial. For x limited to a finite interval, the parameter s controls the

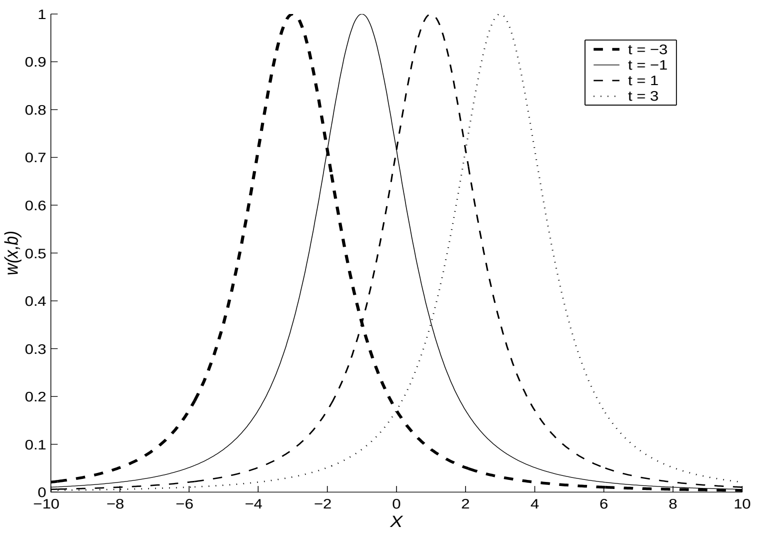

Figure 4.1: The effect of varying the auxiliary parameter t of the weight function $w(x, \mathbf{b})$ with $s = 1$, $k = 2$ held constant.

degree to which asymptotic behaviour is imposed on the model within that interval. The parameter t is where the weight function attains its maximum value, and acts like a centre around which there is radial symmetry. Finally, the parameter k provides control over the limiting behaviour of the asymptotic polynomial according to

$$\lim_{|x|\to\infty} \frac{\tilde{\phi}(x, \mathbf{a}, \mathbf{b})}{|x|^{n-k}} = \frac{a_n}{s}. \tag{4.5}$$

It is the effect of this parameter k that makes the asymptotic polynomial a useful tool for modelling a wide variety of asymptotic limits, including cases of specific interest described in sections 1.8.5, 1.8.6, 1.8.7. Figures 4.1, 4.2 and 4.3 illustrate the effect that varying each of the auxiliary parameters has on the shape of the weight function.

4.3 Approximation with asymptotic polynomials

We now consider the problem of obtaining a least-squares approximation to a set of m pairs of discrete data points $\{(x_i, y_i)\}_{i=1}^{m}$ using an asymptotic polynomial. Given the abscissae $\mathbf{x} = (x_1, \ldots, x_m)^{\mathrm{T}}$, we denote by C the basis matrix generated from the

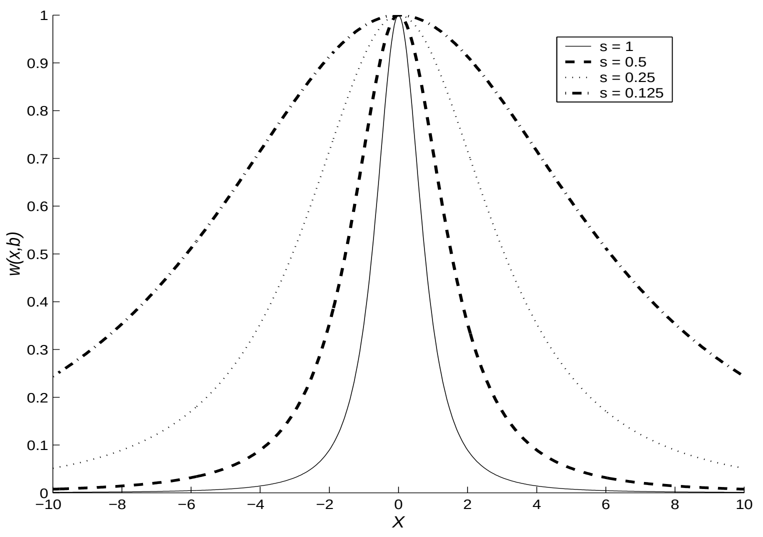

Figure 4.2: The effect of varying the auxiliary parameter s of the weight function $w(x, \mathbf{b})$ with $t = 0$, $k = 2$ held constant.

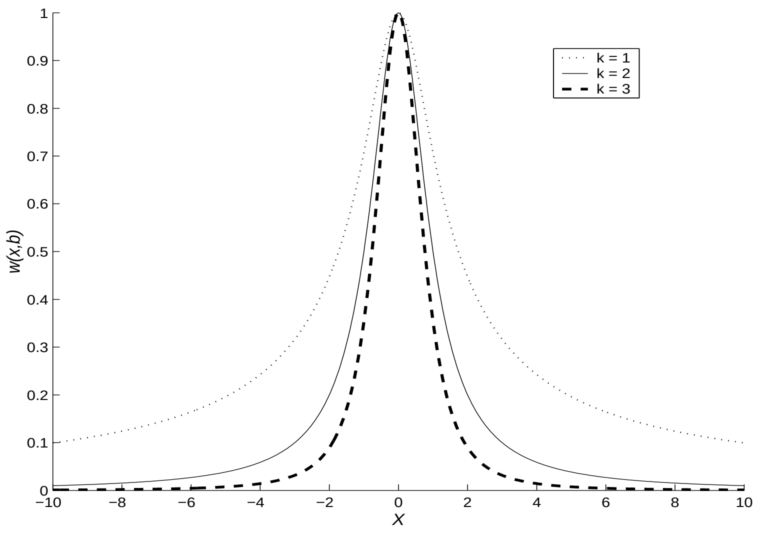

Figure 4.3: The effect of varying the auxiliary parameter k of the weight function $w(x, \mathbf{b})$ with $s = 1$, $t = 0$ held constant.

ϕ_i which has is its i,jth element

$$C_{ij} = \phi_j(x_i). \tag{4.6}$$

Similarly, we define $\tilde{C} = \tilde{C}(\mathbf{b})$ to be the modified observation matrix generated from the modified basis functions $\tilde{\phi}_i$, which has i,jth element given by

$$\tilde{C}_{ij} = \tilde{\phi}_j(x_i) = w_i C_{ij}, \tag{4.7}$$

where $w_i = w(x_i, \mathbf{b})$.

We can now attempt to approximate the data in one of two ways. In the first case we can consider the auxiliary parameters as being fixed, in which case the problem reduces to a weighted linear problem. Alternatively, we can treat them as additional approximation parameters, in which case we are faced with a nonlinear least-squares optimisation problem.

4.3.1 Fixed auxiliary parameters

With the auxiliary parameters fixed, the approximation form $\tilde{\phi}(x, \mathbf{a}, \mathbf{b})$ is just a weighted linear combination of the basis functions $\{\phi_j(x)\}_{j=0}^{n}$. In this case we merely need to choose a suitable degree n for $\tilde{\phi}(x, \mathbf{a}, \mathbf{b})$ and calculate the modified observation matrix $\tilde{C}(\mathbf{b})$ defined in (4.7) and then find the linear least-squares solution parameters $\mathbf{a}$ to the system

$$\min_{\mathbf{a}} \|\mathbf{y} - \tilde{C}\mathbf{a}\|^2, \tag{4.8}$$

where $\mathbf{y} = (y_1, \ldots, y_m)^T$. This approach using fixed auxiliary parameters may be particularly useful if the type of asymptotic behaviour exhibited by the data is explicitly known beforehand. The value of $\mathbf{b}$ can then be chosen and fixed to match this behaviour in a similar manner as was done for polynomial ratios in section 3.2. This type of approximation is similar to that proposed by Kilgore [30] who considers approximation on $[0, \infty)$ using polynomials of degree equal to or less than $2n$, weighted by the function $(1 + x^2)^{-n}$.

Generation of orthogonal polynomials

We have suggested the use of Chebyshev polynomials for the choice of the basis functions $\{\phi_j(x)\}_{j=0}^n$ due to their numerical stability and discrete orthogonality property. Use of the Chebyshev polynomial basis results in mutually orthogonal columns of the observation matrix C. However, this orthogonality may be lost when we form the modified observation matrix $\tilde{C}$ due to the multiplication of the original Chebyshev polynomials by the weight function $w(\mathbf{b})$. If we can generate a set of basis functions that are orthogonal with respect to the weight function $w(\mathbf{b})$ then the resulting observation matrix $\tilde{C}$ formed from these functions will be orthogonal. This can be achieved with the use of the Forsythe method [22] which generates polynomial basis functions that are orthogonal with respect to a specified weight function $w(x)$. We now describe this approach.

Given abscissae $\mathbf{x} = (x_1, \ldots, x_m)^{\mathrm{T}}$ and weights $\mathbf{w} = (w_1, \ldots, w_m)^{\mathrm{T}}$, we generate polynomial basis functions $\phi_j(x)$ of degree j such that

$$\sum_{i=1}^{m} w_i^2 \phi_j^2(x_i) \quad = \quad 1, \tag{4.9}$$

$$\sum_{i=1}^{m} w_i^2 \phi_j(x_i)\phi_l(x_i) \quad = \quad 0, \quad l \neq j. \tag{4.10}$$

We can rewrite these equations more concisely by setting $\boldsymbol{\phi}_j = (\phi_j(x_1), \ldots, \phi_j(x_m))^{\mathrm{T}}$, which results in 4.9 simplifying to

$$\|\tilde{\boldsymbol{\phi}}_j\| \quad = \quad 1, \tag{4.11}$$

$$\tilde{\boldsymbol{\phi}}_j^{\mathrm{T}} \tilde{\boldsymbol{\phi}}_l \quad = \quad 0, \quad l \neq j. \tag{4.12}$$

The following algorithm constructs the $m \times (n+1)$ matrices C and $\tilde{C}$ along with $(n+1)$ vector $\boldsymbol{\alpha}$, n vector $\boldsymbol{\beta}$ and $(n-1)$ vector $\boldsymbol{\gamma}$. The vectors $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_n)^{\mathrm{T}}$, $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_{n-1})^{\mathrm{T}}$ and $\boldsymbol{\gamma} = (\gamma_0, \ldots, \gamma_{n-2})^{\mathrm{T}}$ are such that

$$\phi_0(x) \quad = \quad 1/\alpha_0, \tag{4.13}$$

$$\phi_1(x) \quad = \quad \alpha_0(x + \beta_0)\phi_0(x)/\alpha_1, \tag{4.14}$$

and, for $j = 2, \ldots, n$,

$$\phi_j(x) = \alpha_{j-1}(x + \beta_{j-1})\phi_{j-1}(x)/\alpha_j + \alpha_{j-2}\gamma_{j-2}\phi_{j-2}(x)/\alpha_j. \tag{4.15}$$

I Evaluate $\tilde{\boldsymbol{\phi}}_0$ and α_0.

Set $\alpha_0 = \|\mathbf{w}\|$ and, for $i = 1, \ldots, m$, set $\phi_{i,0} = 1$ and $\tilde{\phi}_{i,0} = w_i\alpha_0$.

II Evaluate $\tilde{\boldsymbol{\phi}}_1$, α_1 and β_0.

Set

$$\beta_0 = -\sum_{i=1}^{m} x_i\phi_{i,0}^2, \tag{4.16}$$

and, for $i = 1, \ldots, m$,

$$\phi_{i,1} = (x_i + \beta_0)\phi_{i,0}, \quad \tilde{\phi}_{i,1} = w_i\phi_{i,1}. \tag{4.17}$$

Set $\alpha_1 = \|\tilde{\boldsymbol{\phi}}_1\|$ and normalize $\tilde{\boldsymbol{\phi}}_1 := \tilde{\boldsymbol{\phi}}_1/\alpha_1$.

III For $j = 2, \ldots, n$, calculate α_j, β_{j-1} and γ_{j-2} and $\tilde{\boldsymbol{\phi}}_j$ from $\boldsymbol{\phi}_{j-1}$ and $\boldsymbol{\phi}_{j-2}$. Set

$$\beta_{j-1} = -\sum_{i=1}^{m} x_i\phi_{i,j-1}^2, \tag{4.18}$$

and $\gamma_{j-2} = -(\alpha_{j-1}/\alpha_{j-2})^2$, and, for $i = 1, \ldots, m$,

$$\phi_{i,j} = (x_i + \beta_{j-1})\phi_{i,j-1} + \gamma_{j-2}\phi_{i,j-2}, \tag{4.19}$$

$$\tilde{\phi}_{i,j} = w_i\phi_{i,j}. \tag{4.20}$$

Set $\alpha_j = \|\tilde{\boldsymbol{\phi}}_j\|$ and normalize $\tilde{\boldsymbol{\phi}}_j := \tilde{\boldsymbol{\phi}}_j/\alpha_j$.

IV Normalize $\boldsymbol{\phi}_j$: for $j = 0$ to n, set $\boldsymbol{\phi}_j = \boldsymbol{\phi}_j/\alpha_j$.

It can be seen that evaluation of the basis functions require only the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Since $\tilde{C}$ is orthogonal the best fit parameters $\mathbf{a}$ are given by $\mathbf{a} = \tilde{C}^{\mathrm{T}}\mathbf{y}$. Figure 4.4 shows the first four orthogonal basis functions $\tilde{\phi}_j$ defined on the interval $[-1, 1]$ using the weight function $w(\mathbf{b})$ when $\mathbf{b} = (3, 0, 4)^{\mathrm{T}}$.

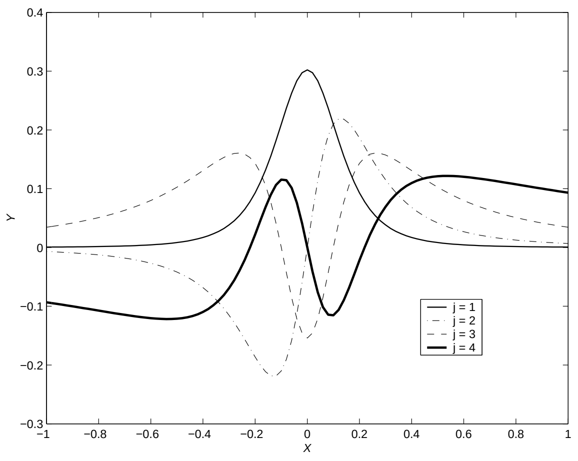

Figure 4.4: First four orthogonal asymptotic polynomials $\tilde{\phi}_j(\mathbf{b}, x)$ generated with $\mathbf{b} = (3, 0, 4)^{\mathrm{T}}$.

4.3.2 Auxiliary parameters as additional approximation parameters

The asymptotic polynomial will have greater flexibility if we regard one or more of s, t and k as additional parameters to be determined as part of the optimization. In this case the matrix $\tilde{C} = \tilde{C}(\mathbf{b})$ is now a nonlinear function of $\mathbf{b}$ and the we are required to find parameters $\mathbf{a}$, $\mathbf{b}$ that solve the nonlinear least-squares system

$$\min_{\mathbf{a},\mathbf{b}} \|\mathbf{y} - \tilde{C}(\mathbf{b})\mathbf{a}\|_2^2. \tag{4.21}$$

The optimization problem (4.21) can be solved using the Gauss-Newton algorithm which has been described in section 2.8.2. If we define the $m \times m$ matrix $W(\mathbf{b})$ to be the diagonal matrix formed from the elements of the weight vector

$$\mathbf{w}(\mathbf{b}) = (w(x_1, \mathbf{b}), \ldots, w(x_m, \mathbf{b}))^T, \tag{4.22}$$

we can now express $\tilde{C}(\mathbf{b})$ as

$$\tilde{C}(\mathbf{b}) = W(\mathbf{b})C. \tag{4.23}$$

We then form the Jacobian matrix of partial derivatives of the quantity

$$\mathbf{h}(\mathbf{a}, \mathbf{b}) = \mathbf{y} - W(\mathbf{b})C\mathbf{a}, \tag{4.24}$$

with respect to the optimization parameters. These partial derivatives are given by

$$\frac{\partial \mathbf{h}}{\partial a_j} = -\tilde{\phi}_j, \tag{4.25}$$

$$\frac{\partial \mathbf{h}}{\partial b_l} = -\left(\frac{\partial W}{\partial b_l}\right) C\mathbf{a}, \tag{4.26}$$

and using these the Gauss-Newton method can then implemented.
Given an initial estimate $\mathbf{b}_0$ of the parameters $\mathbf{b}$, the polynomial basis can be chosen to be orthogonal with respect to the weight vector $\mathbf{w}(\mathbf{b}_0)$ so that for $\mathbf{b}$ close to $\mathbf{b}_0$, the associated Jacobian matrix is relatively well-conditioned. In order to maintain well-conditioned matrices, we can periodically reparametrize the polynomials based on the current estimate of the auxiliary parameters $\mathbf{b}$.

4.3.3 Elimination of the parameters $\mathbf{a}$

For the case where we treat the auxiliary parameters as being free approximation parameters rather than fixed values, it is possible to eliminate the parameters $\mathbf{a}$ from the optimization, thus reducing the complexity of the problem. The least-squares problem (4.21) can be rewritten as

$$\min_{\mathbf{a},\mathbf{b}} \mathbf{h}^{\mathrm{T}}(\mathbf{a},\mathbf{b})\mathbf{h}(\mathbf{a},\mathbf{b}), \tag{4.27}$$

where

$$\mathbf{h}(\mathbf{a},\mathbf{b}) = \mathbf{y} - \tilde{C}(\mathbf{b})\mathbf{a}, \tag{4.28}$$

and $\tilde{C}(\mathbf{b})$ is an $m \times n$ matrix, $m > n$, depending on parameters $\mathbf{b}$. If we fix the value of $\mathbf{b}$, then the optimal value of $\mathbf{a}$ in (4.27) is the solution of the linear least-squares problem

$$\min_{\mathbf{a}} \|\mathbf{y} - \tilde{C}(\mathbf{b})\mathbf{a}\|_2^2, \tag{4.29}$$

where $\tilde{C}(\mathbf{b})$ is now a fixed matrix. The solution parameters $\mathbf{a}$ to this problem must satisfy the normal equations

$$\tilde{C}^{\mathrm{T}}(\mathbf{b})\tilde{C}(\mathbf{b})\mathbf{a} = \tilde{C}^{\mathrm{T}}(\mathbf{b})\mathbf{y}, \tag{4.30}$$

and these equations can be seen to implicitly define the parameters $\mathbf{a}$ as a function of the auxiliary parameters $\mathbf{b}$. If we now define the quantity

$$\mathbf{f}(\mathbf{b}) = \mathbf{y} - \tilde{C}(\mathbf{b})\mathbf{a}(\mathbf{b}), \tag{4.31}$$

we see that the initial nonlinear least-squares problem of equation (4.27) is equivalent to

$$\min_{\mathbf{b}} \mathbf{f}^{\mathrm{T}}(\mathbf{b})\mathbf{f}(\mathbf{b}). \tag{4.32}$$

This is also a nonlinear least-squares problem, but is much simpler as we only need to solve for the auxiliary parameters $\mathbf{b}$. Once we have solved for $\mathbf{b}$, the parameters $\mathbf{a}$ are easily obtained from the normal equations (4.30).
In order to solve this using the Gauss-Newton method, it is necessary to calculate $\mathbf{a}$, $\mathbf{f}$ and the partial derivatives of $\mathbf{f}$ with respect to the auxiliary parameters $\mathbf{b}$. For any fixed value of $\mathbf{b}$, the optimal parameters $\mathbf{a}$ are easily obtained from the normal equations. Once the parameters $\mathbf{a}$ are known, the value of $\mathbf{f}$ can be calculated from (4.31). The partial derivatives of $\mathbf{f}$ are given by

$$\mathbf{f}_l = -\tilde{C}_l\mathbf{a} - \tilde{C}\mathbf{a}_l, \tag{4.33}$$

where we use the subscript l to represent a derivative with respect b_l. Differentiation of the normal equations (4.30) with respect to b_l, leads to

$$\tilde{C}_l^{\mathrm{T}}\tilde{C}\mathbf{a} + \tilde{C}^{\mathrm{T}}\tilde{C}_l\mathbf{a} + \tilde{C}^{\mathrm{T}}\tilde{C}\mathbf{a}_l = \tilde{C}^{\mathrm{T}}\mathbf{y}, \tag{4.34}$$

which reduces

$$\mathbf{a}_l = \left(\tilde{C}^{\mathrm{T}}\tilde{C}\right)^{-1}\left[\tilde{C}_l^{\mathrm{T}}\mathbf{f} - \tilde{C}^{\mathrm{T}}\tilde{C}_l\mathbf{a}\right], \tag{4.35}$$

to after substituting $\mathbf{y}$ with $\mathbf{f} + \tilde{C}\mathbf{a}$ (4.31). We note here that from equation (4.33), evaluation of $\mathbf{f}_l$ only requires the calculation of the product $\tilde{C}\mathbf{a}_l$ and not the vector $\mathbf{a}_l$ itself. If we assume $\tilde{C}$ has QR decomposition $\tilde{C} = QR$, ($Q \in \mathbb{R}^{m\times n}$, $R \in \mathbb{R}^{n\times n}$) [24], then substituting this into (4.35) we get

$$(QR)^T\tilde{C}\mathbf{a}_l = \tilde{C}_l^T\mathbf{f} - (QR)^T\tilde{C}_l\mathbf{a}.$$

Further manipulation gives

$$\begin{aligned} R^T Q^T \tilde{C}\mathbf{a}_l &= \tilde{C}_l^T \mathbf{f} - R^T Q^T \tilde{C}_l \mathbf{a}, \\ \tilde{C}\mathbf{a}_l &= Q^{-T} R^{-T} [\tilde{C}_l^T \mathbf{f} - R^T Q^T \tilde{C}_l \mathbf{a}], \end{aligned}$$

from which we obtain the following expression for $\tilde{C}\mathbf{a}_l$

$$\tilde{C}\mathbf{a}_l = Q R^{-\mathrm{T}} \tilde{C}_l^{\mathrm{T}} \mathbf{f} - Q Q^{\mathrm{T}} \tilde{C}_l \mathbf{a}. \tag{4.36}$$

If we now consider vectors $\mathbf{c}_l$ and $\mathbf{q}_l$ such that

$$R^{\mathrm{T}} \mathbf{c}_l = \tilde{C}_l^{\mathrm{T}} \mathbf{f}, \tag{4.37}$$

$$\mathbf{q}_l = Q^{\mathrm{T}} (\tilde{C}_l \mathbf{a}), \tag{4.38}$$

then we can rewrite (4.36) as

$$\tilde{C}\mathbf{a}_l = Q(\mathbf{c}_l - \mathbf{q}_l). \tag{4.39}$$

In this way, the derivatives $\mathbf{f}_l$ (4.33) with respect to parameters b_l can be calculated from the expression

$$\mathbf{f}_l = -\tilde{C}_l \mathbf{a} - Q(\mathbf{c}_l - \mathbf{q}_l). \tag{4.40}$$

At this point we note that the function $\mathbf{f}(\mathbf{b}) = \mathbf{y} - \tilde{C}(\mathbf{b})\mathbf{a}(\mathbf{b})$ and its derivatives are independent of the choice of basis functions used to represent the polynomials. In particular, we can choose use the Forsythe method to generate the basis functions so that $\tilde{C}$ is orthogonal. Using this basis means that the upper triangular matrix R in equation (4.38), must be the identity matrix, which also means $Q = \tilde{C}$. Using equations (4.38),(4.37) and the fact that R is the identity matrix, equation (4.40) reduces to

$$\mathbf{f}_l = -\tilde{C}_l \mathbf{a} - \tilde{C}(\tilde{C}_l^T \mathbf{f} - \tilde{C}^T (\tilde{C}_l \mathbf{a})). \tag{4.41}$$

This means that the derivatives of $\mathbf{f}$ can be calculated using matrix-vector multiplication, and avoids the need for computationally expensive matrix inversions. Also the orthogonality of $\tilde{C}$ results in a simplification of the normal equations (4.30) and the parameters $\mathbf{a}$ are given by

$$\mathbf{a} = \tilde{C}^T\mathbf{y} \tag{4.42}$$

An additional efficiency gain can be made using the fact that $\tilde{C} = W(\mathbf{b})C$, where $W(\mathbf{b})$ is a diagonal weighting matrix with diagonal elements $w_i(\mathbf{b})$ with $w_i(\mathbf{b}) > 0$. If we write

$$d_{i,l} = \frac{1}{w_i}\frac{\partial w_i}{\partial b_l}, \tag{4.43}$$

then we have

$$\tilde{C}_l = D_l\tilde{C}, \tag{4.44}$$

where D_l is the diagonal matrix with diagonal elements $d_{i,l}$. The quantities

$$\tilde{C}_l^T\mathbf{f} \;=\; \tilde{C}^{\mathrm{T}}(D_l\mathbf{f}), \tag{4.45}$$

$$\tilde{C}_l\mathbf{a} \;=\; D_l(\tilde{C}\mathbf{a}), \tag{4.46}$$

used in (4.38) and (4.37) allows (4.41) to be written as

$$\mathbf{f}_l = -D_l(\tilde{C}\mathbf{a}) - \tilde{C}(\tilde{C}^T(D_l\mathbf{f}) - \tilde{C}^T(D_l(\tilde{C}\mathbf{a}))). \tag{4.47}$$

This means that we can calculate the derivatives of $\mathbf{f}$ using only $\mathbf{f}$, $\mathbf{a}$, D_l, and $\tilde{C}$. We can now calculate the elements of the Jacobian matrix J from the elements of the partial derivative vectors $\mathbf{f}_l$ according to

$$J_{il} = \partial f_i/\partial b_l,$$

and implement the Gauss-Newton method which iteratively updates the current parameter estimate to $\mathbf{b} = \mathbf{b} + \mathbf{p}_{GN}$, where

$$J^{\mathrm{T}}J\mathbf{p}_{GN} = -J^{\mathrm{T}}\mathbf{f}.$$

The Gauss-Newton algorithm for minimizing a sum of squares $F(\mathbf{b}) = \mathbf{f}^{\mathrm{T}}(\mathbf{b})\mathbf{f}(\mathbf{b})/2$ works well if the Jacobian matrix J, is such that $J^{\mathrm{T}}J$ is a good approximation to the

Hessian matrix of second partial derivatives of $F(\mathbf{b})$. In cases where the approximating form has high curvature, it is common for $J^{\mathrm{T}}J$ to provide a poor approximation to the Hessian and this can be the case with asymptotic polynomials. This can prevent the convergence of the Gauss-Newton method and in such cases we may achieve more success by applying the Newton method instead, although again convergence is not always guaranteed. We recall that the Newton step $\mathbf{p}_N$ to update $\mathbf{b} := \mathbf{b} + \mathbf{p}_N$ is the solution of

$$H\mathbf{p}_N = -\mathbf{g}, \quad \mathbf{g} = J^{\mathrm{T}}\mathbf{f},$$

where H is the Hessian matrix of $F(\mathbf{b})$. If convergence occurs, the Newton update step leads to quadratic convergence near the solution, and for this reason there can be computational advantages in using a Newton update. In our implementation, we have used finite differences to approximate H using an implementation provided by Professor Alistair Forbes of NPL.

4.3.4 Choice of initial values of the auxiliary parameters

When we fit data with an asymptotic polynomial using either the Gauss-Newton or Newton methods, careful thought needs to be given to the initial values of the auxiliary parameters. As has been mentioned in chapter 2, the convergence of these methods is dependent on a suitable set of start parameters close to a local minimum of the objective function - in this case the least-squares approximation error surface. Although we can also employ other methods that are not so sensitive to start values (such as the Levenberg-Marquardt method), it is still possible to choose good start values for the Gauss-Newton and Newton methods based on a visual assessment of the data. Although the guidelines we now propose for start parameter selection are far from rigorous, they are often very effective at ensuring convergence of the optimization algorithms used.

The auxiliary parameter t acts like a centre around which the weight function has radial symmetry. Due to this behaviour, we have found that a good starting value for t can usually be chosen to coincide with a region of symmetry within the data if there is one. If there is not a great deal of symmetry in the data then we have found that an alternative is to choose t to coincide with a stationary point of the data. Figure 4.5 shows a plot of measurements of some material properties of aluminium (provided by NPL). This data exhibits some loose symmetry centred somewhere in the region $155 < x < 157$ and so for an initial choice of t we would try a value in this range, probably concentrating on values close to the two localised peaks in this data $x = 155.7$ and $x = 156.3$.

The auxiliary parameter k is usually chosen to mimic the asymptotic behaviour exhibited by the data in conjunction with the degree of the orthogonal polynomial basis in the numerator. For example, suppose we are approximating data that appears (or is known theoretically) to behave like $\sqrt{x}$ as $x \to \infty$, using a degree n asymptotic polynomial. In order for the correct limiting behaviour to be achieved by the asymptotic polynomial, we would require a choice of $k = n - \frac{1}{2}$ because

$$\lim_{x\to\infty} \frac{\alpha x^n}{x^{n-k}(1+s^2(x-t)^2)^{\frac{k}{2}}} = \frac{\alpha}{s^k}. \tag{4.48}$$

The choice for start value of s is the most difficult of the three auxiliary parameters, because it requires a guess of how quickly the asymptotic behaviour takes effect. Figures 4.6 and 4.7 show artificial sets of data that have the same asymptotic limit, but with different rates at which these limits are attained. It is clear that the data in Figure 4.6 approaches its asymptotes faster than that of Figure 4.7 and so we would expect to find that an asymptotic polynomial approximation of the first dataset would have a larger value of s than that of the second dataset. This only gives an indication of a good start value of s relative to some other measure, but at least gives some indication that it is necessary to use larger s values for data that attains its limits quickly.

Using these guidelines it is often possible to choose start parameters that result in convergence of the Gauss-Newton or Newton methods. In cases where convergence is not possible, we can either make adjustments to the initial parameter values, or implement the Levenberg-Marquardt method described in section 2.8.3. This is less sensitive to the choice of start parameter and in our experience has given good results for these problem cases. For a more robust choice of algorithm, the Levenberg-Marquardt method could be used all the time, but for a good selection of initial parameters, we would expect faster convergence with the Newton method.

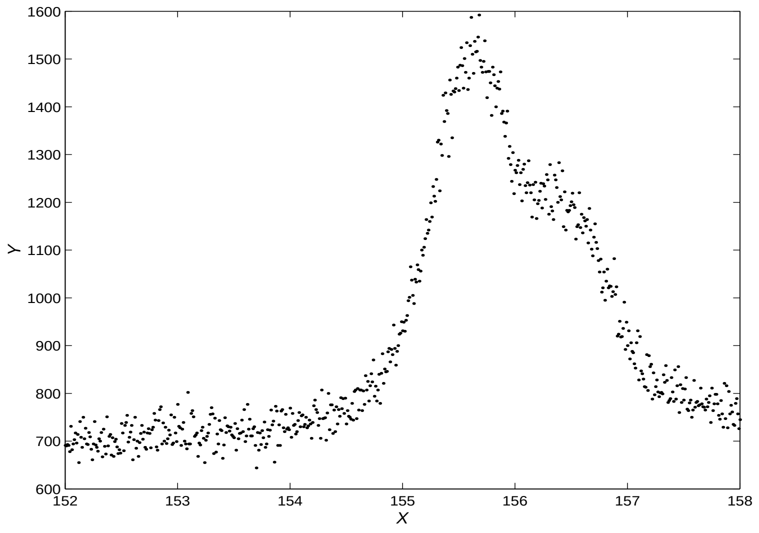

Figure 4.5: 601 experimental measurements obtained from material properties of aluminium.

4.4 *Example applications*

We now present some applications of the asymptotic polynomial to the approximation of experimental data obtained from industry, and make some comparisons with linear approximations to the same data. We note here that instead of approximating directly with the auxiliary parameter $s > 0$, we optimised for e^s, thus ensuring the required positivity of the parameter without the need for constrained optimisation methods.

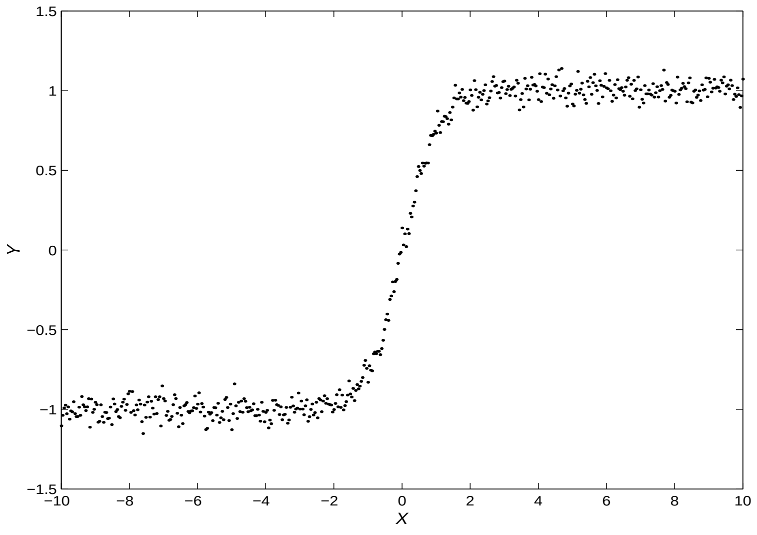

Figure 4.6: Artificial data with asymptotic limits.

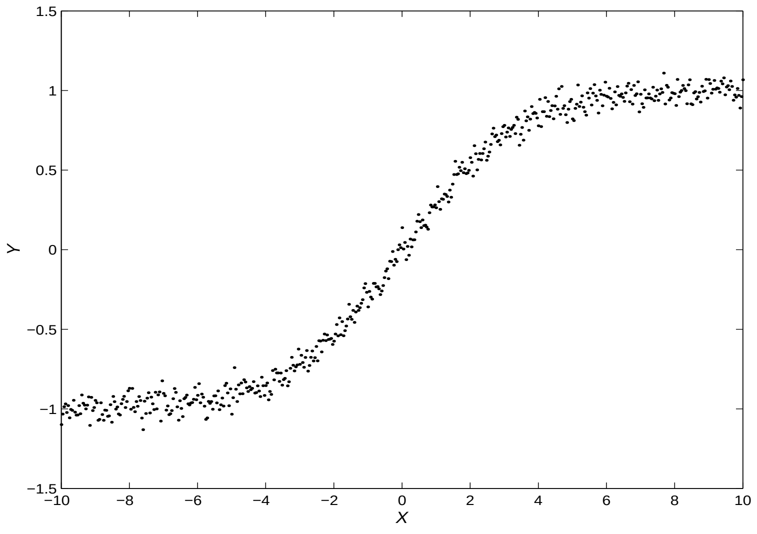

Figure 4.7: Artificial data with asymptotic limits.

4.4.1 Photometric data

This experiment was described in section 1.9.2, and here we present the results of approximating the collected data with asymptotic polynomials. The data is composed of 471 measurements, and an initial inspection of the data reveals a shape similar to a guassian curve with fairly mild asymptotic behaviour. Clearly the data exhibits exponential decay and so using the guidelines for start parameter selection (section 4.3.4) we chose initial auxiliary parameters $\mathbf{b}_0 = (-10, 505, 7)^T$. An asymptotic polynomial of degree 6 was then fitted to the photometric data using both Newton (via finite difference approximation to the Hessian) and Gauss-Newton methods. Table 4.1 compares the norms of the update parameter $\mathbf{p}$ calculated at consecutive iterations of these methods, and illustrates the superior convergence of the Newton method for this example. Figure 4.8 shows degree 9 Chebyshev polynomial approximations and

Table 4.1: Norm of update step $\mathbf{p}$ in Newton and Gauss-Newton methods for the photopic efficiency function example (Fig. 4.8).

Iteration	$\|\mathbf{p}_{GN}\|_2$	$\|\mathbf{p}_N\|_2$
1	0.8496	0.6573
2	0.3354	0.2203
3	0.1380	0.0019
4	0.0568	2.075 e-06
5	0.0235	3.855 e-13
6	0.0097	

degree 6 asymptotic polynomial approximations to the photometric data, from which the superior performance of the latter is evident.

4.4.2 Approximation of sigmoid function

Figure 4.9 shows a polynomial of degree 6 and an asymptotic polynomial of degree 3 fits to the sigmoid curve

$$y = \frac{2}{1+e^{-x}} - 1. \tag{4.49}$$

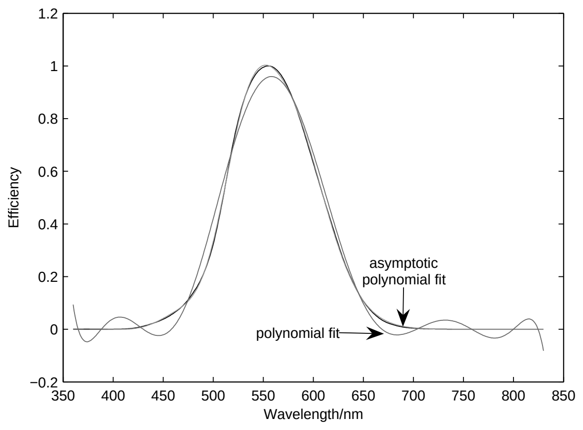

Figure 4.8: Asymptotic polynomial approximation of photopic efficiency function.

(In many circumstances the response of a system to a step change in input has a sigmoid-type behaviour.) We chose to fit the function to 501 equally spaced points from the sigmoid over the interval [-10,10]. The asymptotic polynomial fit is indistinguishable from the sigmoid curve and the maximum error of approximation is less than 2.5×10^{-4}. The degree 6 polynomial fit is much worse. (In the examples considered here the degree of the standard polynomial is 3 more than the asymptotic polynomial so that both models have the same number of parameters.)

Finally, figure 4.10 shows standard polynomial and asymptotic polynomial fits to another experimental dataset (provided by NPL). This dataset consists of 601 measurements of material properties of aluminium, and again we can clearly see the superior performance of the asymptotic polynomial over the Chebyshev polynomial approximation.

4.5 Modelling two asymptotic limits simultaneously

Up to now we have used asymptotic polynomials to model various types of asymptotic behaviour more effectively than standard polynomials. This works well when we are

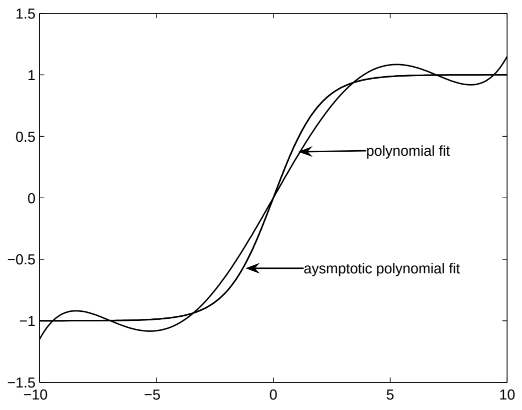

Figure 4.9: Polynomial of degree 6 and asymptotic polynomial of degree 3 fits to a sigmoid curve.

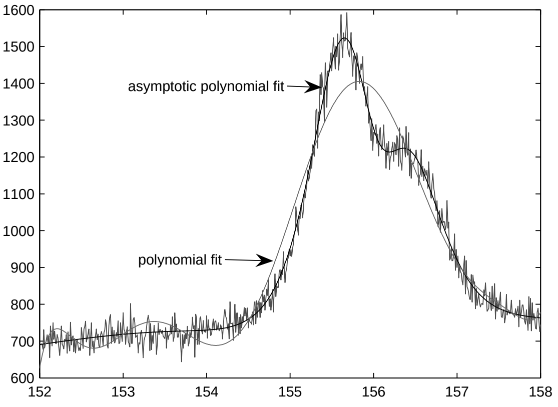

Figure 4.10: Polynomial of degree 9 and asymptotic polynomial of degree 6 fits to 601 measurements of material properties (for aluminium).

dealing with a one sided asymptotic limit, but may not be effective at modelling the double sided asymptotic behaviour that was described in section 1.8.8. To recap, this section described the problem of modelling a function $f(x)$ that has twin asymptotic limits

$$\lim_{x \to -\infty} \frac{f(x)}{x^{\beta_1}} = \alpha_1, \tag{4.50}$$

$$\lim_{x \to +\infty} \frac{f(x)}{x^{\beta_2}} = \alpha_2, \tag{4.51}$$

where $\beta_1, \beta_2, \alpha_1, \alpha_2 \in \mathbb{R}$.

In chapter 3 we introduced the semi-infinite rational spline as an approximation tool for modelling this type of asymptotic behaviour, and in this section we utilise a similar approach to allow asymptotic polynomials to model two different limits simultaneously. This is achieved by considering a new piecewise continuous weight function defined by

$$w(\mathbf{b}, x) = \begin{cases} w_-(\mathbf{b}, x) = \dfrac{1}{(1 + s_1^2(x-\lambda)^2)^{\frac{k_1}{2}}} & \text{for } x \leq \lambda \\ w_+(\mathbf{b}, x) = \dfrac{1}{(1 + s_2^2(x-\lambda)^2)^{\frac{k_2}{2}}} & \text{for } x \geq \lambda \end{cases} \tag{4.52}$$

where $\mathbf{b} = (s_1, s_2, k_1, k_2, \lambda)^T$. As was the case with the SIRS of chapter 3, we regard the parameter λ as a knot (which we can treat as fixed or variable) and examine the conditions required for continuity of the weight function at this knot. Clearly $w(\mathbf{b}, x)$ is C^0 continuous at the knot as we have

$$w_-(\mathbf{b}, x) = w_+(\mathbf{b}, x) = 1, \tag{4.53}$$

when $x = \lambda$.

For C^1 continuity at the knot we require the first derivatives of $w_-(\mathbf{b}, x)$ and $w_+(\mathbf{b}, x)$ with respect to x to have the same value at $x = \lambda$. These derivatives are given by

$$\begin{aligned} w'_-(\mathbf{b}, x) &= \frac{-k_1 s_1^2 (x-\lambda)}{(1 + s_1^2(x-\lambda)^2)^{\frac{k_1}{2}+1}}, \\ w'_+(\mathbf{b}, x) &= \frac{-k_2 s_2^2 (x-\lambda)}{(1 + s_2^2(x-\lambda)^2)^{\frac{k_2}{2}+1}}, \end{aligned} \tag{4.54}$$

and C^1 continuity is verified as being an intrinsic property of the weight function as we have

$$w'_-(\mathbf{b}, x) = w'_+(\mathbf{b}, x) = 0, \tag{4.55}$$

when $x = \lambda$.

So far we have C^1 continuity at the knot satisfied automatically due to the definition of the weight function. However, C^2 continuity is not so straight forward to enforce. The second derivative of the spline functions $w_-(\mathbf{b}\,x)$ and $w_+(\mathbf{b}\,x)$ are given by

$$w_-''(\mathbf{b},x) = (k_1+2)s_1^4k_1(x-\lambda)^2u_1^{-(\frac{k_1}{2}+2)} - s_1^2k_1u_1^{-(\frac{k_1}{2}+1)}, \tag{4.56}$$

$$w_+''(\mathbf{b},x) = (k_2+2)s_2^4k_1(x-\lambda)^2u_2^{-(\frac{k_2}{2}+2)} - s_2^2k_2u_2^{-(\frac{k_1}{2}+1)}, \tag{4.57}$$

where

$$u_1 = 1 + s_1^2(x-\lambda)^2, \tag{4.58}$$

$$u_2 = 1 + s_2^2(x-\lambda)^2. \tag{4.59}$$

Evaluating these second derivatives at $x = \lambda$ and forcing their equality leads to the constraint

$$s_1^2k_1 = s_2^2k_2, \tag{4.60}$$

and as a consequence, we are now faced with a nonlinear optimisation problem involving nonlinear constraints on the parameters. If we are approximating data where the asymptotic behaviour is known explicitly, then we could consider fixing the values of k_1 and k_2 which would result in a significant simplification of the constraint (4.60) as we could then explicitly define s_1 in terms of s_2. However, we would then be faced with an optimisation problem involving a single parameter (either s_1 or s_2), and this is likely to reduce the flexibility of the asymptotic polynomial significantly, and so we will not consider this option further. Due to the complexities of constrained optimisation, we will therefore settle for C^1 continuity of the weighting function, due to the fact that this requires no constraints at all and is an intrinsic property of $w(\mathbf{b},x)$. Approximation of data with this modified version of the asymptotic polynomial can then be achieved in exactly the same way as described previously, the only difference being the addition of extra optimisation parameters k_2 and s_2.

When approximating double asymptotic limits, we have found that this splined version of the asymptotic polynomial yields better approximations to those obtained using the standard asymptotic polynomial form. We finish this section by presenting figure 4.11, which illustrates this superiority of the splined version of a degree 10 asymptotic polynomial approximation of $tanh(x)$ at 101 equally spaced abscissae on $[-5, 5]$.

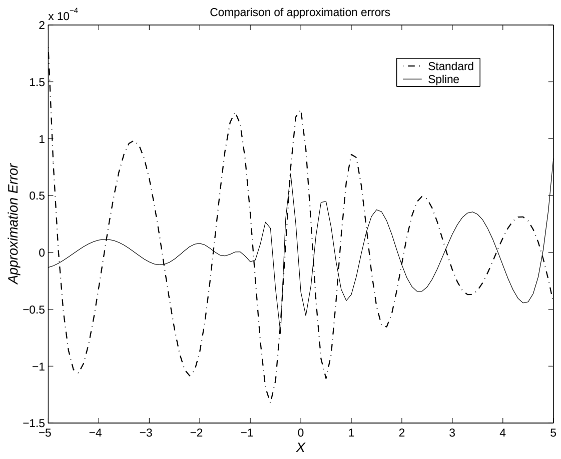

Figure 4.11: Approximation errors of degree 10 spline and standard asymptotic polynomial fits to $tanh(x)$.

4.6 Concluding remarks

In this chapter we have demonstrated that data reflecting asymptotic behaviour can be modelled by polynomial basis functions multiplied by a nonlinear weighting function depending on three auxiliary parameters. In addition, we have developed numerically stable optimization algorithms using polynomial basis functions orthogonal with respect to the weighting function. Further simplification has been achieved with the implementation of a parameter elimination scheme that allows the approximation problem to solved compactly. The model can easily be extended to allow for different asymptotic behaviour as $x \to \infty$ and $x \to -\infty$, by using a piecewise continuous weighting function that itself has variable asymptotic behaviour. We also have observed some guidelines for selecting initial values for the parameters that in many cases result in convergence of our algorithm. However, in cases where we cannot find a good set of starting parameters, we can still fit the asymptotic polynomial to data using the more robust Levenberg-Marquardt algorithm. The examples presented have shown that these asymptotic polynomial approximations can be much more effective than standard polynomial approximations.

Chapter 5

POLE FREE LEAST-SQUARES RATIONAL APPROXIMATION

5.1 Introduction

In this chapter we introduce a method of fitting polynomial ratios to data in such a way that the denominator function has no real roots, resulting in the rational approximation having no singularities anywhere on the real line. We do this by presenting an alternative representation of the denominator polynomial and applying nonlinear optimisation techniques to evaluate parameters. This approach is very simple and we illustrate how to combine it with the SIRS discussed in Chapter 3, resulting in pole-free SIRS approximations.

5.2 Positive denominator rational approximation

Unless we are faced with a specific problem requiring the modelling of a singularity in the data, we will most often wish for a rational approximation to have no poles on the approximation range. For Chebyshev rational approximation there are a number of methods available to constrain the approximation parameters, and using these methods we may be able to find a way to ensure that the denominator has no zeros. Most of these methods are based on modifications of the Differential Correction Method which was presented in Chapter 2. Kauffman and Taylor have implemented variations on the DCM that allow linear constraints to be placed on the approximation parameters [28], and another that places a strictly positive lower bound on the denominator function [29]. Another method has been developed by Gugat [25], which forces the denominator to be bounded above and below by continuous functions. Our primary interest is in least-squares approximation, as far as we are currently aware, there are no equivalent methods to those described above, other than constrained optimisation techniques.

5.3 Requirements for a strictly positive denominator

When faced with the general problem of approximating discrete points on an interval $[a, b]$ it is not immediately obvious what constraints on approximation parameters will lead to a positive polynomial denominator. If we are instead faced with an approximation problem on an interval $[c, d]$ where $c, d > 0$, then strict positivity can be achieved with the simple constraint

$$0 < q_i, i = 1, \ldots, m, \tag{5.1}$$

where our denominator function

$$Q_m(\mathbf{q}, x) = \sum_{j=0}^{m} q_j x^j, \tag{5.2}$$

is defined as in previous chapters. In such cases we can enforce these constraints and get pole free approximations. With these constraints satisfied, we now have a monotonically increasing denominator on the approximation interval, and hence some flexibility in the approximation is lost. If we are approximating on a general range $[a, b]$ with $a < b < 0$, or $a < 0$ then we could consider approximating using a denominator of the form

$$Q_m(\mathbf{q}, x) = \sum_{j=0}^{m/2} q_j x^{2j}. \tag{5.3}$$

Using such a denominator removes the odd monomial basis functions, and then enforcing the constraint (5.1) will again provide a positive denominator. However, again we have lost some flexibility with the removal of the odd functions.

5.4 Implementing positive denominator constraints

In an attempt to avoid the use of constrained optimisation techniques, we propose a simple method of data fitting, using a subset of the set of all pole-free rational approximations. This method works by considering an alternative representation of the denominator polynomial in the rational form. Up to now we have mainly been using polynomial ratios $R_m^n(\mathbf{p}, \mathbf{q}, x)$ for approximation, where n, m are chosen according to the shape of the data, and are generally of low degree to avoid ill-conditioning problems with our proposed methods. For the moment, we will restrict our discussion to cases where the denominator polynomial $Q_m(\mathbf{q}, x)$ has even degree. This denominator has at most m distinct roots, and we can represent $Q_m(\mathbf{q}, x)$ as

$$Q_m(\mathbf{q}, x) = q_m \prod_{i=1}^{m} (x - \alpha_i) \tag{5.4}$$

where the $\alpha_i \in \mathbb{C}$ are the denominator roots. If it is the case that all of the roots are complex, then our denominator will be strictly positive over the entire real line resulting in a pole free rational approximation. We now wish to examine some conditions that we can apply to enforce this situation. Because we have chosen m to be an even integer, we can also express the denominator (suitably normalised with $q_0 = 1$) as a product of quadratic polynomials

$$Q_m(\mathbf{q}, x) = \prod_{i=1}^{m/2} (\gamma_i x^2 + \xi_i x + 1), \tag{5.5}$$

for constants $\gamma_i, \xi_i \in \mathbb{R}$. Now if we can ensure that each of the quadratics in the product (5.5) have complex roots, then we can be sure that Q_m has no real roots and as a consequence the rational approximation contains no poles. For all roots of the equation (5.5) to be complex, we require that

$$4\gamma_i > \xi_i^2, \tag{5.6}$$

for $i = 1, \ldots, \frac{m}{2}$.

Now if we apply the following explicit parameter constraints

$$\xi_i^2 = \gamma_i, \tag{5.7}$$

for $i = 1, \ldots, \frac{m}{2}$, then the conditions (5.6) are automatically satisfied. Thus if we consider rational approximation using denominators of the form

$$Q_m(\mathbf{q}, x) = \prod_{i=1}^{m/2} (q_i^2 x^2 + q_i x + 1), \tag{5.8}$$

then we are guaranteed to have no real poles on the entire real line. In addition, this representation defines a degree m polynomial given in terms of only $\frac{m}{2}$ coefficients, and so we have reduced the number of denominator approximation parameters by half. Obviously the space of polynomials described by (5.8) is only a subset of the space of all degree m polynomials, and so we are reducing the size of the space of approximation forms that we are using. In particular, the set of polynomials given in (5.8) is only a subset of set of polynomials strictly greater than zero. Despite these limitations, the simplicity of the constraint provides us with exactly what we require, without the need for constrained optimization algorithms, and has the advantage of reducing the number of optimisation parameters by $\frac{m}{2}$.

5.5 Parameter evaluation

In order to try and evaluate the parameters we will need to apply an optimisation algorithm such as Gauss-Newton, and will need to evaluate the Jacobian matrix for this problem. Using the denominator representation (5.8), out rational approximant is defined as

$$R_m^n(\mathbf{p}, \mathbf{q}, x) = \frac{P(\mathbf{p}, x)}{Q(\mathbf{q}, x)} = \frac{\displaystyle\sum_{i=0}^{n} p_i x^i}{\displaystyle\prod_{i=1}^{m} (q_i^2 x^2 + q_i x + 1)}. \tag{5.9}$$

The Jacobian matrix of partial derivatives of the residual vector is defined in (2.43) and for approximations of the form (5.9), is defined as the matrix with elements

$$J(\mathbf{p},\mathbf{q})_{ij} = \begin{cases} \dfrac{x_i^{j-1}}{Q(\mathbf{q},x_i)}, & j = 1,\dots,n+1 \\ \\ R(\mathbf{p},\mathbf{q},x_i)\dfrac{(2q_j x_i^2 + x_i)}{(q_j^2 x_i^2 + q_j x_i + 1)}, & j = n+2,\dots,n+m+2 \end{cases} \tag{5.10}$$

With the Jacobian defined we can apply the Gauss-Newton method to fit the polynomial ratio representation (5.9) to a set of data. The main problem we are faced with in attempting to do this is that it is not obvious how to obtain a good a set of start parameters for the algorithm. We would hope for the Gauss-Newton method to have good local convergence in the vicinity of a local minimum (although this may not be the case), but even if this is the case we have no way of ensuring a start set that will converge to a solution. With the standard representation of the rational function where the denominator parameters appear linearly in the denominator, we would usually apply the Loeb algorithm to obtain a suitable set of start parameters with which to start off the Gauss-Newton process, as was discussed in Chapter 2. However, with the quadratic product representation for the denominator we have no such method and so we must use an optimisation technique that converges for a wide range of start parameters. We have used the MATLAB optimisation toolbox to implement the Levenberg-Marquardt method to obtain convergence from arbitrarily chosen start parameters. In general we have used randomly selected parameters from the interval $[-1, 1]$. The problem with this approach is that it is quite likely that the optimisation technique has converged to a local minimum only, and we have no idea how this compares with the global minimum for our given objective function. In addition, the approach is very much 'trial and error', as for certain sets of start parameters we either get no convergence, or very slow convergence. However, after numerous applications of this approach on a large number of datasets, we have found that most of the time the algorithm converges to good pole-free rational approximations within an acceptable number of iterations. In addition, it was encouraging to see that in most cases, the algorithm converged to the same set of solution parameters for a variety of different starting parameters. Table 5.1 shows the results obtained from application of this method to data points taken from a number of different functions. In each case, we fitted a degree [4,4] rational approximation to 51 equally spaced samples on $[-2, 2]$ from each function, with approximation start parameters as randomly selected numbers from $[-1, 1]$. It is important to note here that for each function we used a seperate random selection of start parameters, but we repeated the algorithm a number of times for each function using different random parameters and arrived at the same set of parameters at convergence (when convergence occurred). Figure 5.1 shows one of the actual approximations obtained using this method.

Table 5.1: Error norms and iterations to convergence of pole-free rational approximations.

Function	Convergence	$\|\mathbf{e}\|_2$
$e^{0.1x}cos(2x)$	20	6.2567×10^{-2}
$tanh(x)$	38	7.7715×10^{-2}
$cosh(x)$	38	1.1547×10^{-2}
$tan(x)$	42	57.574
$sin(2x)$	14	5.7270×10^{-2}
e^{-x^2}	14	2.0982×10^{-2}
e^{x^2}	-	-
$ln(x+3)$	34	1.0541×10^{-2}
$e^{-x^2}tanh(x)$	12	1.9510×10^{-2}

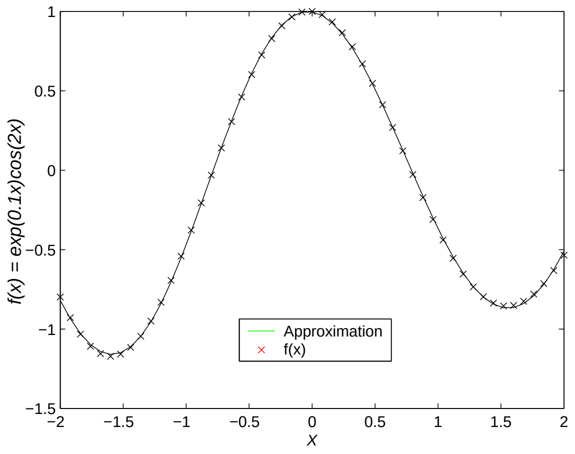

Figure 5.1: Degree [4,4] pole-free rational approximation to $e^{0.1x}cos(2x)$.

5.5.1 Ill-conditioning issues

The columns of the Jacobian matrix that correspond to the partial derivatives of the residual vector with respect to the denominator parameters are symmetric, in the sense that if at any step of our algorithm we update values such that 2 (or more) of the denominator parameters are equal, we will have 2 (or more) identical columns in the Jacobian which will then be rank deficient. This will also be the case should any of the denominator parameters converge to the same value of any of the others. This is an important issue to consider when choosing the start parameters. When we generated random start parameters as described previously, we first checked to ensure that the initial Jacobian matrix generated with them was not too ill conditioned. If this was the case, parameters too close together we manually adjusted, or a new set of random parameters were created. In the vast majority of cases we found that for the [4,4] rational pole free approximation, we would not see a Jacobian with condition number higher than the order of 10^3. For the case of a [6,6] pole free rational form, we found the condition number generally to be no larger than 10^6. We also examined the [8,8] form, but in this case, we were often faced with condition numbers higher than 10^9. We also found that convergence was generally slower for pole free approximations higher than degree [6,6], and would not recommend fitting these higher degree approximations for these reasons.

5.5.2 Consideration of other types of constraint

The approximation form described in the previous section is guaranteed to be pole free on the entire real line, but in some cases we may not require such strict conditions. For example, if we were not intending to use the approximation to extrapolate outside the approximation interval, we may be happy to have poles in the approximation, provided they do not occur in the approximation interval itself.

Another example is the approximation on a real interval $[a, b]$ where $0 < a < b$. In this case we can ensure strict positivity of the denominator on the range $[0, \infty)$ by defining the denominator polynomial in the usual way as

$$Q_m(\mathbf{q}, x) = \sum_{j=0}^{m} q_j x^j, \tag{5.11}$$

and then set constraints on the parameters in the form

$$0 < q_0 < q_1 < \ldots < q_m. \tag{5.12}$$

However, this has the slight disadvantage that it is a monotonically increasing function, and such a denominator will provide less flexibility than polynomial ratios formed with using (5.8) which is not necessarily monotonically increasing. In addition it is hard to see how the approximation form with denominator constraints (5.12) will result in a superior approximation to that of the asymptotic polynomial of the previous chapter, that also has a strictly positive denominator, but has fewer coefficients and is more stable numerically.

The constraints in (5.7) are very simple to implement, and also serve to reduce the dimensionality of the approximation parameter vector. However, the set of approximations generated using these constraints is only a subset of the entire space of pole free rational approximations, and so we could gain even greater flexibility by considering the constraints

$$\xi_i = \lambda_i \gamma_i^2, \tag{5.13}$$

where $\lambda_i > \frac{1}{4}$. Using these constraints, does increase the dimensionality of the approximation parameter vector, but it increases the space of approximations to include all possible strictly positive denominators, and provides a more flexible denominator, and therefore potentially more flexible and superior rational approximations.

5.5.3 Pole-free SIRS approximations

The simplicity of the constraints (5.7), allows us to extend the SIRS work of Chapter 3 to include pole-free constraints in the denominator. If we define a SIRS approximation be composed of only one denominator defined by (5.8) we can introduce the continuity constraints across the knot of the SIRS merely through the numerator functions. Explicitly, the resulting pole free SIRS approximation will be defined by

$$R_m^{n_1,n_2}(\mathbf{g},\mathbf{h},\mathbf{q},u) = \begin{cases} \dfrac{\sum\limits_{i=0}^{n_1} g_i u^i}{\prod\limits_{i=1}^{m/2}(q_i^2x^2 + q_i x + 1)} = R_-(\mathbf{g},\mathbf{q},u) & \text{for } u \le 0 \\ \dfrac{\sum\limits_{i=0}^{n_2} h_i u^i}{\prod\limits_{i=1}^{m/2}(q_i^2x^2 + q_i x + 1)} = R_+(\mathbf{h},\mathbf{q},u) & \text{for } u \ge 0 \end{cases}, \tag{5.14}$$

where $u = x - \lambda$, and λ is the SIRS knot.

As mentioned previously, we cannot utilise the Loeb algorithm as we could with the standard SIRS approximation, and for this particular case we would use the Levenberg-Marquardt algorithm to obtain a solution.

5.6 Conclusion

In this chapter, we have suggested a very simple factorisation that provides an alternative representation of the approximation parameters, that with the application of some simple constraints, results in a pole free rational approximation over the entire real line. In addition we have shown that we can successfully fit such approximations using established optimisation techniques. While it is accepted that these approximations do not give as small an approximation error as the free rational approximation form does, it will obviously be a superior approximation to cases where an unwanted pole is fitted in the approximant of an unconstrained form. This representation also has the advantage of reducing the number of approximation parameters in the problem, at the expense of some flexibility. Should such approximations not be as accurate as desired, greater flexibility can be obtained using the full set of approximation parameters as given by the wider range of constraints (5.13).

Chapter 6

ℓ_P RATIONAL APPROXIMATION

6.1 Introduction

In this chapter we consider another modification of the Loeb algorithm, this time applying it to ℓ_p rational approximations with the use of Lawson's algorithm [31]. Lawson's algorithm is an iterative weighted least squares procedure, first studied by Lawson in [31] and has been proven to converge to a best linear ℓ_∞ approximation on discrete data sets. The algorithm was later extended by Rice and Usow in [49] and has been proved to converge to best ℓ_p approximations for $p > 2$. We describe this algorithm and present a modification that extends its applicability to ℓ_p approximation for $p < 2$, along with numerical results to support this. Finally we propose a combined Lawson - Loeb algorithm for use in generating ℓ_p rational approximations on discrete data sets. We describe a general combined algorithm which has a number of subtle variants, and present some numerical results. The work of this chapter is published in the conference proceedings from the 2004 conference on Mathematical Methods for Curves and Surfaces held in Tromso. This paper also contains other work on the Loeb algorithm that was done in collaboration with Professor John Mason, and is contained in Appendix A. This other work is not included as a part of this thesis as it is only partially the authors work. The reason for mentioning this is because it forms part of a joint effort to attempt to prove convergence of the Loeb algorithm, which is a central theme of the thesis. Given the wide use of the Loeb algorithm within this work, we feel it appropriate to show that some attempt was made to prove its convergence.

6.2 The Lawson algorithm, and the Rice and Usow extension

The Lawson algorithm [31] is used to fit linear ℓ_∞ approximations to discrete data. It is an iterative procedure that has proven convergence to a best linear ℓ_∞ approximation, and its implementation involves fitting weighted ℓ_2 approximations at each iteration. This algorithm was extended by Rice and Usow [49] to include ℓ_p approximation problems ($p > 2$) using the same weighted least squares approach, and also has been proved to converge to best linear ℓ_p approximation. The original Lawson algorithm has also been utilised in more recent work on the Huber M-Estimator [2]. We will now describe the Rice and Usow algorithm in more detail. Consider the linear form

$$L(\mathbf{A}, x) = \sum_{i=0}^{n} a_i \phi_i(x) \tag{6.1}$$

where $\mathbf{A} = (a_0, \ldots, a_n)$ is a set of approximation parameters and $\phi_i(x)$ are a set of linearly independent basis functions. We wish to fit ℓ_p approximations to a set of function values $f(x_j) = f_j$ $(j = 1, \ldots, m)$ defined on a discrete point set $\mathbf{x} = (x_1, x_2, \ldots, x_m)$. This is achieved by finding solution parameters $\mathbf{A}$ that minimise the quantity

$$\|\mathbf{e}(\mathbf{A}, \mathbf{x})\|_p = \Big(\sum_{i=j}^{m} |e_j(\mathbf{A}, x_j)|^p \Big)^{1/p} \tag{6.2}$$

where

$$e_j(\mathbf{A}, x_j) = L(\mathbf{A}, x_j) - f_j, \tag{6.3}$$

are the elements of the residual vector $\mathbf{e}(\mathbf{A}, \mathbf{x})$.

The basic principle behind the algorithm is to convert the problem into a weighted least-squares problem, by considering

$$|e_j(\mathbf{A}, x_j)|^p \approx w|e_j(\mathbf{A}, x_j)|^2, \tag{6.4}$$

where w is a suitable weight term. Specifically, the Rice and Usow algorithm works by updating the weight at iteration $(k + 1)$ according to the formula

$$w_j^{k+1} = (w_j^k |e_j^k|)^{\frac{p-2}{p-1}} \tag{6.5}$$

where

$$e_j^k = f_j - L(\mathbf{A}^k, x_j) \tag{6.6}$$

is the residual at $x = x_j$ at iteration k.

If

$$w_j^k \approx |e_j^k|^{p-2} \tag{6.7}$$

then $w_j^k|e_j^k|^2 \approx |e_j^k|^p$ and so we have an ℓ_p approximation. Now if $w_j^k \approx |e_j^k|^{p-2}$ then

$$w_j^{k+1} \approx (|e_j^k|^{p-2}|e_j^k|)^{\frac{p-2}{p-1}} = |e_j^k|^{p-2} \tag{6.8}$$

and so $w_j \to |e_j|^{p-2}$.

As has been mentioned, this algorithm is proved to converge to a best linear ℓ_p approximation for $p > 2$, and an extensive proof of this can be found in [49]. Before discussing the application to rational approximation, we firstly propose a further extension of this algorithm to the case of linear ℓ_p approximation for $p < 2$. Our proposed algorithm differs from that of Rice and Usow by updating the weight at iteration $(k + 1)$ according to the formula

$$w_j^{k+1} = \left(\frac{w_j^k}{|e_j^k|}\right)^{\frac{2-p}{3-p}} \tag{6.9}$$

Now if $w_j^k \approx |e_j^k|^{p-2}$ we have

$$w_j^{k+1} \approx \left(\frac{|e_j^k|^{p-2}}{|e_j^k|}\right)^{\frac{2-p}{3-p}} = |e_j^k|^{p-2} \tag{6.10}$$

and so $w_j \to |e_j|^{p-2}$ as required.

We now describe the implementation of the algorithm before presenting some numerical results of its application.

At iteration k, define $e_j^k = f_j - L(\mathbf{A}, x_j)$ and w_j^k as the approximation error and weight at x_j respectively, and choose an initial set of weights $w_j^1 = \frac{1}{m}$.

1. Find the best ℓ_2 approximation $L(A, x)$ to f with weight w_j^k at x_j and calculate e_j^k.

2. Calculate new weights w_j^{k+1} using (6.5) for $p > 2$ or from (6.9) for $p < 2$.

3. Normalise the weights
$$w_j^{k+1} := \frac{w_j^{k+1}}{\sum_{j=1}^{m} w_j^{k+1}}. \tag{6.11}$$

4. Proceed from step 1 until the solution has converged.

As has been shown, the principle behind the algorithms is to generate a weight term approximately equal to $|e|^{p-2}$, thus reducing the ℓ_p approximation problem to a weighted least squares problem. This can be presented more generally by writing the equations for updating the weights as

$$w_j^{k+1} = (w_j^k)^{\lambda_1} |e_j^k|^{\lambda_2} \tag{6.12}$$

for general indices λ_1, λ_2. The requirement that $w_j \approx |e_j|^{p-2}$ leads to

$$|e|^{p-2} = |e|^{(p-2)\lambda_1} |e|^{\lambda_2}. \tag{6.13}$$

If we set $\lambda_1 = \lambda_2 = \lambda$ and then equate indices in (6.13) we are left with the Rice and Usow algorithm (6.5). The algorithm is easy to implement but can be rather slow to converge although there are proposed methods of accelerating convergence described in [49]. Due to the division by the approximation error in (6.9), we have set a maximum weight value to avoid infinite or very large weights in the case of data points having residuals very close to zero. We have chosen to set an upper bound of 10^6 and set any weights greater than this equal to 10^6.

We now present some numerical results for this algorithm for various values of $p < 2$. In practice we have found that choices of $p \leq 1$ generally fits a good ℓ_1 approximation to data, and so we compare them with the best ℓ_1 approximation obtained using the Barrodale and Roberts algorithm [9]. In particular we compare the data points that are interpolated by these methods. In all cases we have chosen a Chebyshev polynomial basis and taken 21 equally spaced abscissae $x_i \in [-1, 1]$.

Algorithm	Interpolation points x_i	$\|L(x) - f\|_1$	Iterations
Barrodale Roberts	±0.9, ±0.3	0.00542	-
Lawson ($p = 1$)	±0.9, -0.4, 0.3	0.00550	10
Lawson ($p = 0.1$)	±0.9, -0.4, 0.3	0.00550	7
Lawson ($p = -1$)	±0.9, -0.4, 0.3	0.00550	6

Table 6.1: Degree 3 ℓ_p approximation of $\log(x + 3)$ using Lawson's algorithm

Algorithm	Interpolation points x_i	$\|L(x) - f\|_1$	Iterations
Barrodale Roberts	±0.9, ±0.5, 0	0.05849	-
Lawson ($p = 1$)	±0.9, ±0.5, 0	0.05849	10
Lawson ($p = 0.1$)	±0.9, -0.4, 0.3	0.05849	8
Lawson ($p = -1$)	±0.9, -0.4, 0.3	0.05849	7

Table 6.2: Degree 3 ℓ_p approximation of $tanh(x)$ using Lawson's algorithm

Algorithm	Interpolation points x_i	$\|L(x) - f\|_1$	Iterations
Barrodale Roberts	±1.0, ±0.8, ±0.5, ±0.2	2.3487×10^{-3}	-
Lawson ($p = 1$)	±1.0, ±0.8, ±0.5, ±0.2	2.3487×10^{-3}	8
Lawson ($p = 0.1$)	±1.0, ±0.8, ±0.5, ±0.2	2.3487×10^{-3}	6
Lawson ($p = -1$)	±1.0, ±0.8, ±0.5, ±0.2	2.3487×10^{-3}	6

Table 6.3: Degree 6 ℓ_p approximation of e^{-x^2} using Lawson's algorithm

As can be seen from Tables 6.1, 6.2 and 6.3, using the Lawson algorithm with $p = 1$ has generated a best or near-best ℓ_1 approximation. Good ℓ_1 approximations are also obtained for other choices of $p < 1$. We have also found that these approximations successfully ignore outliers in the data, as one would expected with an ℓ_1 approximation [9]. In addition we have generally observed that, as the value of p decreases, the system becomes increasingly more ill-conditioned, but that the algorithm converges more quickly. For choices of p less than 1.5, we have also noticed that the resulting approximation interpolates at $n + 1$ data points, where n is the degree of the approximation. For p values inbetween 1.5 and 2, the approximation starts to exhibit behaviour consistent with least squares solutions. In this sense it would appear that for values $1 < p < 2$ the resulting approximation is a compromise between ℓ_1 and ℓ_2 approximation, interpolating at some data points and approximating the remainder. Examining the behaviour of approximations for various values in this range, the transition from ℓ_1 to ℓ_2 approximations can clearly be seen, as shown in Figure 6.1. Here we can see the effect of outliers on some approximations fitted to data for various values of p.

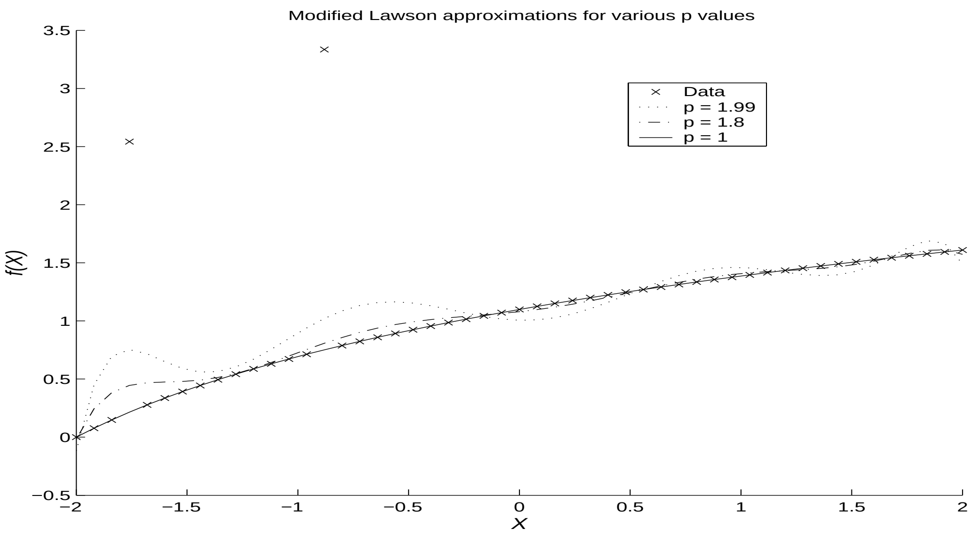

Figure 6.1: Lawson algorithm approximations to data with outliers, for various $p < 2$

6.3 *Application of Lawson's algorithm to ℓ_p Rational approximation*

The study of the Rice and Usow variant of Lawsons, in addition to the previous work on the Loeb algorithm, inspired the idea of combining these two approaches into a single algorithm for use in fitting ℓ_p rational approximations. This thought process was mainly due to the fact that both algorithms are iterative weighted least-squares methods and it seemed reasonable to try and combine the two. The development of this algorithm is further motivated by the fact that there seems to be a shortage of methods available for this type of data fitting. We have not seen a great deal of published work on algorithms specifically dealing with ℓ_p rational approximation, other than the work of Watson [52], who considers the problem via the Gauss-Newton method with numerator and denominator variable separation.

6.4 A combined Rice-Usow-Loeb Algorithm

We consider the general ℓ_p rational approximation problem in which we wish to approximate a set of function values $f(x_i) = f_i$ defined on a discrete point set $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ by a polynomial ratio $R(\mathbf{p}, \mathbf{q}, x)$ as defined in section 1.7. We propose to minimize the quantity

$$\|f - R(\mathbf{p}, \mathbf{q}, x)\|_p \tag{6.14}$$

by iteratively minimizing

$$\left\| \frac{w_k^l (fQ(\mathbf{q}_k^l, x) - P(\mathbf{p}_k^l, x))}{W_k^l} \right\|_2 \tag{6.15}$$

with respect to the approximation parameters $\mathbf{q}, \mathbf{p}$, over two iteration variables k, l. In this sense we are using two weights, one of which (the w term) corresponds to the linear ℓ_p approximation weight from the Rice-Usow algorithm, and the other (the W term) corresponding to a Loeb type weight term defined as the denominator obtained using the denominator solution parameters obtained at the previous iteration. We update the w_k^l terms when iterating over k in a similar manner to that the Rice-Usow variant algorithm according to the formula

$$w_{k+1}^l = \begin{cases} (w_k^l |e_k^l|)^{\frac{p-2}{p-1}} & \text{for } p > 2 \\ \left(\frac{w_k^l}{|e_k^l|}\right)^{\frac{2-p}{3-p}} & \text{for } p < 2 \end{cases} \tag{6.16}$$

where

$$e_k^l(x) = fQ(\mathbf{q}_k^l, x) - P(\mathbf{p}_k^l, x), \tag{6.17}$$

with initial weights set as $w_1^1 = \frac{1}{N}$.

We update the W_k^l when iterating over the l variable by setting

$$W_k^{l+1} = \frac{1}{Q(\mathbf{q}_k^l, x)} \tag{6.18}$$

as is the case with Loeb's algorithm. Clearly we may proceed in a number of ways by choosing the way in which to iterate over the variables k, l. We suggest the most obvious:

1. Cycle through l until convergence with k fixed and then through k with l fixed. Repeat until convergence.

2. Cycle through k until convergence with l fixed and then through l with k. Repeat until convergence.

3. Set $k = l$ and update both weights simultaneously.

We now present some numerical results from the application of the third of the above algorithms. For these results we fitted a degree $(5, 5)$ polynomial ratio to 101 equally spaced values on $[-2, 2]$ from the function $tanh(x)$.

Table 6.4 shows iterations to convergence for a number of different values of p. In each case the algorithm converged successfully to a pole free solution. As can be seen in Figure 6.2, the error curve from both approximations for $p = 4$ and $p = 16$ behave like a minimax error curve, with the larger p value exhibiting the equioscillation property to a larger extent than the smaller. We have observed similar bevahiour in all cases we have tested, with the error curve tending to a minimax error curve for increasingly larger values of p. This behaviour suggests that we can obtain approximately minimax rational approximations using this algorithm for large values of p.

p value	Iterations	$\|\|\mathbf{e}\|\|_2$
3	10	3.3157×10^{-7}
4	15	3.3902×10^{-7}
8	30	3.4559×10^{-7}
12	53	3.4786×10^{-7}
16	51	3.4900×10^{-7}
32	93	3.5070×10^{-7}

Table 6.4: Degree $(5, 5)$ ℓ_p rational approximation of $tanh(x)$ using Rice-Usow-Loeb algorithm

Of the 3 different variants on the proposed Rice-Usow-Loeb algorithm, we have found that the third provides the fastest convergence. All three methods give similarly good approximations, but the faster convergence of the third makes it our preferred option. To illustrate the performance of this algorithm we compared it with the method of Watson to fit ℓ_p rational approximations for various values of p. In each case, we fit degree (2,2) rational approximations to 51 equally spaced points from the function $y = tanh(x)$ over the interval [0,1]. The method used by Watson was applied using the solution parameters from the Rice-Usow-Loeb algorithm at convergence. The results are displayed in table 6.4, in which we used the abbreviation RUL for Rice-Usow-Loeb algorithm. From these results, we can see that the RUL algorithm performs

Norm	RUL Convergence	Lawson $\|\|\mathbf{e}\|\|_p$	Watson Convergence	Watson $\|\|\mathbf{e}\|\|_p$
ℓ_3	12	1.28184×10^{-3}	18	1.28180×10^{-3}
ℓ_8	18	6.57338×10^{-4}	55	6.57330×10^{-4}
ℓ_{16}	44	5.48020×10^{-4}	113	5.48016×10^{-4}

Table 6.5: Comparison of approximations obtained using the method of Watson and the RUL algorithm

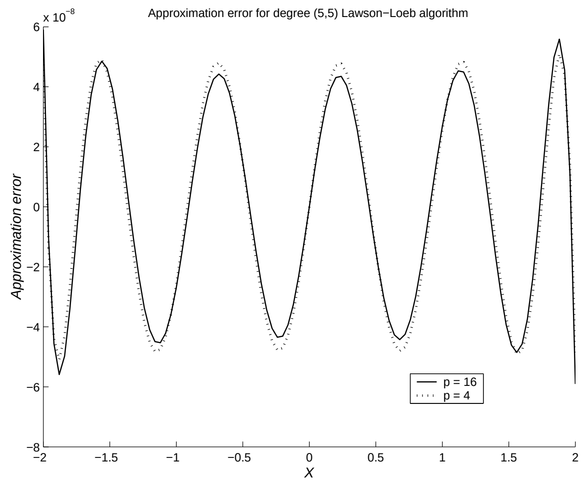

Figure 6.2: Error plots of ℓ_4 and ℓ_{16} rational approximations to $tanh(x)$ using the Rice-Usow-Loeb algorithm.

well, generating equally good approximations as the method of Watson, but also with faster convergence.

6.5 A Lawson-Loeb algorithm

Up to now, we have only discussed the Rice-Usow variation of the original Lawson algorithm, used for ℓ_p approximations ($p > 2$). The original Lawson algorithm is a linear weighted least-squares method, that is proved to converge to a best linear Chebyshev approximation [31]. It is implemented as follows

For a linear approximation form $L(\mathbf{A}, x)$, we define at iteration k, the residual $e_j^k = f_j - L(\mathbf{A}^k, x_j)$, and w_j^k as the weight at x_j, and finally, choose an initial set of weights $w_j^1 = \frac{1}{m}$, where m is the number of data points we are approximating.

1. Find the best ℓ_2 approximation $L(A, x)$ to f with weights w_j^k at x_j and calculate the residuals e_j^k.

2. Calculate new weights according to the equation

$$w_j^{k+1} = \frac{w_j^k |e_j^k|}{\sum_{j=1}^m w_j^k |e_j^k|}. \tag{6.19}$$

3. Proceed from step 1 until the solution has converged.

As with the case of the Rice-Usow-Loeb algorithm, we propose a Lawson-Loeb algorithm of the same kind, but this time using the first weight w_j^{k+1} calculated according to (6.19), and the second using the formula

$$W_k^{l+1} = \frac{1}{Q(\mathbf{q}_k^l, x)}. \tag{6.20}$$

With these weights, we can again propose the same three variants on the algorithm

1. Cycle through l until convergence with k fixed, and then through k with l fixed. Repeat until convergence.

2. Cycle through k until convergence with l fixed and then through l with k. Repeat until convergence.

3. Set $k = l$ and update both weights simultaneously.

As was the case with the Rice-Usow-Loeb algorithm, we have again found that the third option provides the fastest convergence, and is the method we would recommend. Table 6.5 shows the results of the application of this algorithm to a variety of functions, showing iterations to convergence, and approximation error of the resulting approximations. In all cases, the data points were evaluated at 101 equally spaced points on the interval $[-2, 2]$ and approximated using a degree $(5, 5)$ polynomial ratio. As the results indicate, even when convergence does occur, it is very slow. In some cases for the approximations in table 6.5 convergence occurred with a slight modification to the degree of the approximation (using even degrees for $cos(x)$ for example) but again, the convergence was slow. In fact, we have observed that this algorithm

Function	Iterations	$\|\mathbf{e}\|_{\ell_\infty}$
$sin(x)$	531	4.7141×10^{-8}
$cos(x)$	na	na
$tanh(x)$	360	4.8890×10^{-8}
e^{-x}	725	1.9652×10^{-10}
$cosh(x)$	na	na
$cos(x) + sin(x)$	1272	1.2220×10^{-7}

Table 6.6: Degree $(5, 5)$ ℓ_∞ rational approximations using Lawson-Loeb algorithm

is even more sensitive to a good choice of degree for the approximation than the standard Loeb algorithm or the Rice-Usow-Loeb algorithm. When convergence does occur, the resulting error curves exhibit the equioscillation property perfectly, as is shown in figure 6.3. The inconsistency of the performance of the algorithm, combined with its speed is slightly disappointing, as it had been hoped that this would provide a fast, weighted least-squares algorithm to obtain Chebyshev rational approximations. Judging by the results, it would appear that from a performance perspective, it would be better to try to obtain near best rational Chebyshev approximations using high values of p in the Rice-Usow-Lawson algorithm instead. For example, fitting the same data used in for $tanh(x)$ using the Rice-Usow-Loeb algorithm, with a choice of $p = 16$, the resulting Chebyshev norm of the error is 5.0937×10^{-8}, which is almost the same as that obtained using the Lawson-Loeb method, and converges in 30 steps, rather than 360 as shown table 6.5.

6.6 Conclusion

We have proposed an extension of the Rice-Usow algorithm for cases $p < 2$, which converges quickly to near best ℓ_1 linear approximations, and generally compares

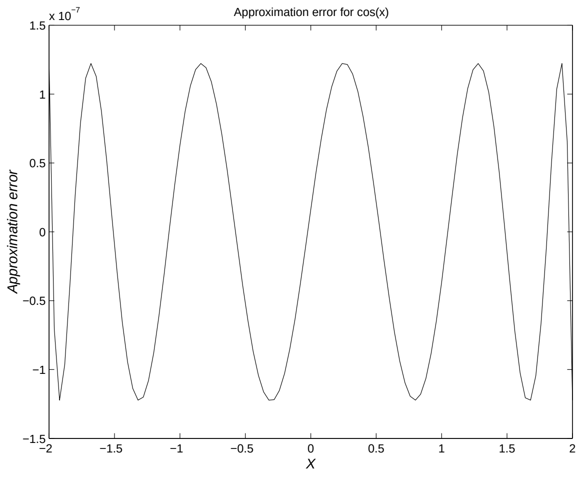

Figure 6.3: Error plot of ℓ_∞ rational approximation to $tanh(x)$ using the Lawson-Loeb algorithm.

favourably with approximations obtained with the Barrodale-Roberts algorithm. Also this algorithm is considerably easier to implement than the Barrodale-Roberts algorithm, although may not be favoured over it due to lack of a convergence proof.

We have also observed good results with the proposed Rice-Usow-Loeb algorithm, which exhibits good convergence properties with the data that we have tested it on, and produces good ℓ_p approximations that compare favourably to those obtained using the optimisation methods proposed by Watson. As an extension of this, for fitting Chebyshev rational approximations, we have also proposed a Lawson-Loeb algorithm that is similar in style to the Rice-Usow-Loeb algorithm. This algorithm in general is not so successful or robust, with slow convergence, and in some cases no convergence at all. However, when it does converge, it appears to converge to an approximation whose error exhibits the equioscillation property. If this algorithm was more reliable with regards to convergence, it would be interesting to research some methods of improving the speed of convergence, but as it stands the algorithm is slow and unreliable. As a consequence of this poor performance, if it was desired to fit a Chebyshev rational approximation to data, we would recommend the use of an established algorithm such as the differential correction method which has proven convergence properties.

Chapter 7

CONCLUSIONS

7.1 Introduction

We have presented a number of algorithms and new rational approximation forms for the approximation of discrete data in general, but with particular emphasis on data that exhibits the type of asymptotic behaviour described in chapter 1. This research has been motivated by the sponsoring institution (NPL) requirement for new approximation methods to better approximate discrete data of this kind. In this final chapter, we shall summarise the research presented in this thesis, and highlight some potential areas where this work could be extended with future research.

7.2 The Semi-Infinite Rational Spline

The work of chapter 3 introduces the semi-infinite rational spline approximation form, and discusses a modification to the Loeb method that provides an algorithm to fit this form to discrete data, that is straight forward to implement. The SIRS provides an approximation form that is able to have different asymptotic limits as x approaches $\pm\infty$. This makes it a useful tool in the approximation of data that has a similar asymptotic behaviour, and we have shown that we can obtain very good approximations to such data, particularly in the region where the asymptotic behaviour starts to dominate the shape of the data. In many cases such approximations are superior to those of a standard polynomial ratio fitted using the Loeb algorithm.

The first very obvious extension to the work of chapter 3 is that this could be very easily extended to cover SIRS approximation in the other norms ℓ_1 and ℓ_∞, as the work of Barrodale and Mason [7] has shown the Loeb algorithm to be effective and easily implemented in these norms also. Similarly, it seems conceivable that the Differential Correction Method (2.3) could be modified to approximation with the SIRS, as its continuity constraints at the knot are implicitly satisfied by the definition of the function. Although this has not been looked into in great detail, it certainly seems to be a potential area of extension of the algorithm.

It will be evident that this thesis has been of a very practical nature, with almost no results or proofs of existence or uniqueness of best SIRS approximations. This is another important area for future work, and would certainly make the SIRS more attractive as an approximation tool if such proofs could be established.

7.3 Asymptotic polynomial approximation

The Asymptotic polynomial work of chapter 4 is another useful tool for approximation of data with asymptotic behaviour. It has the advantage of being numerically stable, by orthogonalising the numerator basis functions with respect to the denominator weight function. We have found that this work gives very good approximations, which have the advantage of being pole free on the entire real line due to the strictly positive denominator function. One way in which this work could be extended would be to consider another weight function very similar to that described in chapter 4, but having two knots instead of one. We define this new weight function by

$$w(\mathbf{b},x) = \begin{cases} w_-(\mathbf{b},x) & = \dfrac{1}{(1+s_1^2(x-\lambda_1)^2)^{\frac{k_1}{2}}} & \text{for } x \leq \lambda_1, \\ w_0(\mathbf{b},x) & = 1 & \text{for } \lambda_1 \geq x \leq \lambda_2, \\ w_+(\mathbf{b},x) & = \dfrac{1}{(1+s_2^2(x-\lambda_2)^2)^{\frac{k_2}{2}}} & \text{for } x \geq \lambda_2, \end{cases} \tag{7.1}$$

where $\lambda_1 < \lambda_2$, and $\mathbf{b} = (s_1, s_2, k_1, k_2, \lambda_1, \lambda_2)^T$.

Approximation with asymptotic polynomials defined with weight function (7.1) is effectively fitting a linear approximation to the data on the interval $[\lambda_1, \lambda_2]$, and only fits weighted polynomials outside this interval to enforce the correct type of asymptotic behaviour. Again we could consider the knots to be fixed or variable and it would be interesting to see if this type of weight function would give superior approximations to the original method.

Another very interesting extension would be to consider approximation using bivariate asymptotic polynomials. The paper of Huhtanen and Larsen [26] describes an algorithm that generates a set of bivariate orthogonal polynomials. Unfortunately this work only deals with the case of orthogonality with respect to a weight function that is unity, which for our needs would result in bivariate linear approximation. If it were possible to extend this work to generate bivariate orthogonal polynomials with respect to arbitrary weight functions (or a small class of special functions appropriate for our needs), it may be possible to extend the asymptotic polynomial to apply to bivariate approximation.

7.4 Pole free rational approximation

Chapter 5 introduces a very simple and novel approach that can be used to fit polynomial ratios that are guaranteed to be pole free over the entire real line, using the Levenberg-Marquardt algorithm. Here we have considered representing the denominator polynomial as a product of quadratics, each of which is forced to have a strictly negative discriminant, and hence complex roots, via the denominator approximation parameter definitions. Again, it would be nice if existence proofs could be established for this form, and therefore we highlight this as an area of future work. As was the case for the SIRS, it seems reasonable to assume that we could apply this approach to approximation in other norms, in particular it may be possible to modify the Differential Correction Method to fit pole free Chebyshev rational approximations. If this was possible, it may also be possible to establish convergence proofs for the algorithm applied to this form. This potential extension to other norms would seem to

apply to almost all of the work in this thesis, which has dealt almost exclusively with least-squares approximations in accordance with the needs of the collaborating CASE sponsor, NPL. The natural assumption would be that the work could be extended to the problem of Chebyshev rational approximation, as this is a well established field, with a great deal of research material available.
Another area of interesting research would be to look at other constraints on the individual quadratic factors of the denominator. For example, if it was only necessary for the denominator to be pole free on the approximation interval itself (rather than the entire real line), then we would be faced with a less stringent set of conditions on the parameters, and a potentially more flexible approximation as a result.

7.5 ℓ_p Rational Approximation

Chapter 6 introduces some new algorithms that are combinations of Loeb-Lawson and Rice-Usow-Loeb for ℓ_∞ and ℓ_p rational approximation respectively. Also we have proposed an extension to the Rice-Usow algorithm for ℓ_p approximation for the cases $p < 2$. Clearly we would have liked to have produced a proof of convergence of the algorithm, so this is one area for future consideration. Similarly, would be the case for the Rise-Usow-Loeb algorithm, although we would anticipate considerable difficulty here, as the original Loeb algorithm itself has so far not been proved to converge (as far as we are aware), so finding a proof for the hybrid algorithm may well require a proof for the original Loeb method first. We found that the Loeb-Lawson algorithm was generally not so reliable with many cases of non-convergence discovered, and that for cases when it did converge, the solution was very close to the same solution as for the Rice-Usow-Loeb method using a large p value. Also the latter method converged much more quickly. Had the performance of the Loeb-Lawson algorithm been more impressive we would have considered a direct comparison with the solutions of the DCM (2.3) which converges to the best rational Chebyshev approximation.

Another possibility is to combine the semi-infinite rational spline work with this algorithm, thus to create an algorithm for ℓ_p SIRS rational approximation.

7.6 Radial basis function rational approximation

Another area that was researched (although not presented in the thesis) was fitting ratios of linear combinations of radial basis functions [48]. This was only a small amount of research, fitting approximations using the Loeb algorithm, using various combinations of the abscissae as the centres of the radial basis functions. We found that the resulting approximations were generally very good, but this study did give rise to a large number of questions, such as how to choose the centres to begin with, then how to select those appearing in the numerator and those in the denominator. For linear interpolation using radial basis functions, the centres are usually chosen to be the data points themselves, but when approximating with radial basis functions, there are other methods of choosing the centres, such as clustering. After some initial research, we found that performing any significant research in this area would take the thesis into a very different direction and so after the initial encouraging results were obtained, the work was left aside to continue on work more appropriate to the rest of the thesis. However, we feel that this is a very interesting and new area to work on, but as yet we have seen no research in this area. The nature of radial basis functions makes them a popular tool for multivariate approximation and interpolation, so if some algorithms for fitting ratios of linear combinations of radial basis functions could be developed it would be a significant achievement of interest to members of the multivariate approximation, radial basis function, and rational approximation research community.

APPENDIX A

Here we present the remainder of the contents of the paper that was jointly published by the author of this thesis. It is the first part of the paper "Rational and ℓ_p Approximations - Renovating Existing Algorithms" and is published in the proceedings of the 2004 conference on Mathematical Methods for Curves and Surfaces held in Tromso. The work of this part of the paper cannot be claimed to be the sole work of the author, and was done in collaboration with Professor John Mason. It is reproduced as it appears in the conference proceedings, and is included for completeness, as it does complement the other work in this thesis, being largely concerned with obtaining some detailed error analysis for a slightly modified Loeb algorithm.

Abstract

In this paper we present a modified version of Loeb's algorithm for ℓ_2 rational approximation together with an error analysis and numerical results that suggest linear convergence. We also propose an extension to Lawson's algorithm to include ℓ_p approximation for $p < 2$, and a combined Lawson-Loeb algorithm for use in generating ℓ_p rational approximations to discrete data.

Introduction

The fact that ratios of linear forms can operate in infinite ranges and can have asymptotic limits and poles make them a very desirable tool for use in the modelling of physical systems. Rational functions are particularly well suited for the modelling of functions that are known to have a particular asymptotic behaviour such as decay, or singularities. In consequence the subject of rational approximation needs to be updated, as do some of the very early, but re-usable algorithms.

First of all we investigate the use of Loeb's algorithm [14] in fitting least squares rational approximations to discrete data. This is a very simple yet effective iterative procedure which has been shown to converge quickly to near-best rational approximations for a large number of functions [7]. We present a modified version of this algorithm that yields an informative error analysis which, in conjuction with our numerical results suggests linear convergence. This algorithm also has the property of fitting a relative approximation, while fitting simultaneously the data values and their reciprocals.

We then go on to study the Lawson algorithm. This is an iterative weighted least squares procedure, first studied by Lawson in [31] and has been proven to converge to a best linear ℓ_∞ approximation on discrete data sets. The algorithm was later extended by Rice and Usow in [49] and has been proved to converge to best ℓ_p approximations for $p > 2$. We describe this algorithm and present a modification that extends its applicability to ℓ_p approximation for $p < 2$, along with numerical results to support this. Finally we propose a combined Lawson - Loeb algorithm for use in generating ℓ_p rational approximations on discrete data sets. There appears to be a shortage of methods available for this approximation problem and so we describe a general method which has a number of subtle variants, along with some numerical results.

ℓ_2 *Rational approximation - Loeb's algorithm*

Consider the rational approximation form

$$R(x) = \frac{A(x)}{B(x)} = \frac{a_0\phi_0(x) + \ldots + a_n\phi_n(x)}{b_0\phi_0(x) + \ldots + b_m\phi_m(x)} \tag{7.2}$$

where the $\phi_i(x)$ form a set of linearly independent basis functions, and $\{a_0, \ldots, a_n, b_0, \ldots, b_m\}$ is the set of approximation parameters.

We consider the problem of finding the best ℓ_2 rational approximation to a given set of function values $f(x_i) = f_i$ defined on a discrete point set $X = \{x_1, x_2, \ldots, x_N\}$. Thus we seek the parameter vector that minimizes the quantity $\|f(x) - R(x)\|_2$.
One method available for fitting rational functions to discrete data is Loeb's algorithm [14]. This is an iterative procedure in which the quantity

$$\|w_k(fB_k - A_k)\|_2 \tag{7.3}$$

is minimized with respect to the approximation parameters at the kth iteration. Here A_k and B_k respectively denote the numerator and denominator functions $A(x)$ and $B(x)$ obtained at the current step k. The term w_k is a weight function defined as $1/B_{k-1}$ where B_{k-1} is the denominator function obtained from the previous iteration and evaluated at the data points x_i. To prevent the trivial solution $A(x) = B(x) \equiv 0$, the set of approximation parameters is usually normalized by setting $b_0 = 1$, and the iteration usually started with initial weight $w_1 = 1$. In practice this is a very reliable algorithm and is attractive due to its ease of implementation and fast convergence to near-best approximations in the majority of cases, particularly when used to obtain least squares approximations [7].
We propose a modified version of Loeb's algorithm which uses the weight function

$$w_k = \left[\frac{1}{2(fB_{k-1})^2} + \frac{1}{2(A_{k-1})^2}\right]^{\frac{1}{2}} \tag{7.4}$$

in place of the original weight $1/B_{k-1}$. The reasons for this choice of weight function are as follows

1. The weight is suitable for relative approximation of f by $\frac{A}{B}$:

   $$\frac{fB_k - A_k}{fB_{k-1}} \to \frac{f - \frac{A}{B}}{f} \equiv \frac{fB - A}{fB}. \tag{7.5}$$

   It is also suitable for relative approximation of the reciprocal function f^{-1} by $\frac{B}{A}$:

   $$\frac{-(f^{-1}A_k) - B_k}{f^{-1}A_{k-1}} \to -\frac{f^{-1} - \frac{B}{A}}{f^{-1}} \equiv \frac{fB - A}{A}. \tag{7.6}$$

Thus we have chosen the new weight w as the r.m.s of the weights $\frac{1}{fB}$ and $\frac{1}{A}$, so as to approximate both f and f^{-1}.

2. w treats A and fB alike and involves both.

3. $fB \simeq A$ and so, to a modest accuracy

$$w \simeq \frac{1}{fB} \simeq \frac{1}{A}. \tag{7.7}$$

It is assumed here that both $f(x)$ and $f^{-1}(x)$ have no zeros in the domain of the data, so that the weight has no poles. We can find this approximation $\frac{A}{B}$ with weight (7.4) using an iterative procedure in which a Galerkin solution is obtained at step k. Consider the following inner product for functions $u(x)$, $v(x)$ defined by

$$\langle u, v\rangle = \sum_{i=1}^{N} u(x_i)v(x_i). \tag{7.8}$$

We can then obtain a solution at step k by solving the system of $m+n+1$ equations:

$$\langle (fB_k - A_k), w_k^2\phi_j\rangle = 0, \quad j = 0, \ldots, m+n \tag{7.9}$$

with respect to the $m+n+1$ approximation parameters. This is equivalent to finding the ℓ_2 solution of the overdetermined set of equations

$$\begin{aligned} (fB_{k-1})^{-1}(fB_k - A_k)(x_i) = 0 \quad i = 1, \ldots, N \\ (A_{k-1})^{-1}(fB_k - A_k)(x_i) = 0 \quad i = 1, \ldots, N \end{aligned} \tag{7.10}$$

with respect to the same set of approximation parameters.

Convergence and Error Analysis

At step k we obtain functions A_k and B_k that satisfy the Galerkin property (7.9). We define the following limiting functions which also satisfy the Galerkin property (7.9)

$$A = \lim_{k\to\infty} A_k \tag{7.11}$$

$$B = \lim_{k\to\infty} B_k \tag{7.12}$$

$$w = \lim_{k\to\infty} w_k \tag{7.13}$$

Although we have assumed existence of A,B and w, we have found that these limits exist in practice for reasonably behaved data. From (7.9) and (7.8) we have

$$\sum_{j=0}^{m+n} \langle (fB_k - A_k), w_k^2\phi_j\rangle\phi_j = 0 \tag{7.14}$$

and

$$\sum_{j=0}^{m+n} \langle (fB - A), w^2\phi_j\rangle\phi_j = 0. \tag{7.15}$$

Subtracting (7.14) from (7.15) we are left with

$$\sum_{j=0}^{m+n} \langle (f\delta B_k - \delta A_k)w^2 + (w^2 - w_k^2)(fB_k - A_k), \phi_j\rangle\phi_j = 0 \tag{7.16}$$

where $\delta B_k = B - B_k$ and $\delta A_k = A - A_k$.

From the definition of the weight (7.4) we have

$$w^2 - w_k^2 = \frac{1}{2}\left(\frac{1}{(fB)^2} + \frac{1}{A^2} - \frac{1}{(fB_{k-1})^2} - \frac{1}{(A_{k-1})^2}\right) \tag{7.17}$$

which reduces to

$$\frac{-\delta B_{k-1} f(2B - \delta B_{k-1})}{2f^4B^2B_{k-1}^2} - \frac{-\delta A_{k-1} f(2A - \delta A_{k-1})}{2A^2A_{k-1}^2}. \tag{7.18}$$

If we neglect quadratic terms involving δ this is approximately equal to

$$\frac{-\delta(A_{k-1} + E_{k-1})}{(A+E)(A_{k-1} + E_{k-1})^2} - \frac{\delta A_{k-1}}{AA_{k-1}^2} \tag{7.19}$$

where

$$\begin{aligned} E_k &= fB_k - A_k & (7.20)\\ E &= fB - A & (7.21)\\ \delta E_k &= E - E_k. & (7.22)\end{aligned}$$

Since we are dealing with good approximations, the E terms are small, and so we are left with

$$w^2 - w_k^2 \simeq -\frac{2\delta A_{k-1}}{AA_{k-1}^2}. \tag{7.23}$$

Substituting (7.23) into (7.16) and rearranging gives

$$\sum_{j=0}^{m+n} \langle (f\delta B_k - \delta A_k)w + 2w^{-1}(fB_k - A_k)\frac{\delta A_{k-1}}{A_{k-1}^2}, w\phi_j\rangle \phi_j \simeq 0. \tag{7.24}$$

Using (7.7), equation (7.24) reduces to

$$\sum_{j=0}^{m+n} \langle \delta E_k w, w\phi_j\rangle w\phi_j - 2\sum_{j=0}^{m+n} \langle E_k \frac{A\delta A_{k-1}}{A_{k-1}^2} w, w\phi_j\rangle w\phi_j \simeq 0. \tag{7.25}$$

In a Galerkin space, the best linear ℓ_2 approximation of degree p to a function G is given by

$$G \simeq \sum_{i=1}^{p} \langle G, \phi_j\rangle \phi_j \tag{7.26}$$

and the weighted equivalent defined by

$$Gw \simeq \sum_{i=1}^{p} \langle Gw, w\phi_j\rangle w\phi_j = \sum_{i=1}^{p} \langle Gw^2, \phi_j\rangle w\phi_j. \tag{7.27}$$

Also in a Galerkin space the following inequality is valid

$$\|Gw\|_2 \leq 2\|\sum_{i=1}^{p} \langle Gw, w\phi_j\rangle w\phi_j\|_2. \tag{7.28}$$

Thus from (7.27) it is evident that the left hand side quantity in (7.25) is a best ℓ_2 approximation to $\delta E_k w$ and the right hand side quantity is a best ℓ_2 approximation to $2wE_k\delta A_{k-1}AA_{k-1}^{-2}$. Therefore

$$\delta E_k \simeq 2E_k \delta A_{k-1} A_{k-1}^{-1} \tag{7.29}$$

which, using (7.7) can be expressed as

$$\frac{\delta E_k}{E_k} \simeq 2\frac{\delta F_{k-1}}{F_{k-1}} \tag{7.30}$$

where $F_k = fB_k + A_k$ and $\delta F_k = f\delta B_k + \delta A_k$. Integrating both sides of (7.30) leads to

$$E_k \simeq cF_{k-1}^2 \tag{7.31}$$

for some constant of integration c. Finally, from (7.28) and (7.29) we obtain

$$\|\delta E_k\|_2 \leq 4\|E_k \delta A_{k-1} A^{-1}\|_2. \tag{7.32}$$

We now go on to present some numerical results to support some of the above results. In all cases we have chosen a monomial basis, a quadratic numerator and denominator in the approximation form, and have chosen $\{x_i\}_{i=1}^{21} \in [-1, 1]$. The results presented

k	$\|E_k\|$	$\|F_{k-1}^2\|$	$\frac{\|E_k\|}{\|F_{k-1}^2\|}$	$\|\delta E_k\|$	$2\left\|\frac{E_k \delta A_{k-1}}{A_{k-1}}\right\|$
2	5.5475 e-06	20.3554	2.7253 e-07	3.1809 e-06	5.9954 e-06
3	6.7892 e-06	32.4337	2.0933 e-07	1.4810 e-08	1.9351 e-08
4	6.7975 e-06	32.5078	2.0910 e-07	6.1040 e-11	7.9779 e-11
5	6.7975 e-06	32.5081	2.0910 e-07	2.5109 e-13	3.2857 e-13
6	6.7975 e-06	32.5081	2.0910 e-07	1.9179 e-15	1.3506 e-15
7	6.7975 e-06	32.5081	2.0910 e-07	0.0	6.0130 e-18

Table 7.1: Approximation of $\log(x + 3)$ using Loeb's algorithm

k	$\|E_k\|$	$\|F_{k-1}^2\|$	$\frac{\|E_k\|}{\|F_{k-1}^2\|}$	$\|\delta E_k\|$	$2\left\|\frac{E_k \delta A_{k-1}}{A_{k-1}}\right\|$
2	9.3961 e-04	50.8086	1.8493 e-05	9.0611 e-05	2.3917 e-03
3	9.4561 e-04	88.4315	1.0693 e-05	1.7738 e-07	2.6125 e-07
4	9.4560 e-04	88.4269	1.0694 e-05	7.4319 e-10	1.2449 e-09
5	9.4560 e-04	88.4269	1.0694 e-05	3.5068 e-12	5.5449 e-12
6	9.4560 e-04	88.4269	1.0694 e-05	0	2.6475 e-14

Table 7.2: Approximation of $e^x + e^{-0.5x}$ using Loeb's algorithm

in Table 1 and Table 2 show that the quantities E_k and F_{k-1}^2 are proportional as in (7.31). The results also support the validity of equations (7.29) and (7.32). In our testing of this algorithm, we have found that it generally converges in the same number of iterations (or less) as the existing Loeb algorithm. It also compares favourably in terms of the size of the norm of the approximation error at convergence.

BIBLIOGRAPHY

[1] J. Abouir and A. Cuyt. Multivariate partial Newton-Padé and Newton-Padé type approximants. *Journal of Approximation Theory*, 72:301–316, 1993.

[2] I. Anderson, J. C. Mason, and C. Ross. Extending Lawson's algorithm to include the Huber M-estimator. In Albert Cohen, Christophe Rabut, and Larry L. Schumaker, editors, *Curve and Surface Fitting*, pages 1–8. Vanderbuilt University Press, 2000.

[3] K. Appel. Rational approximation of decay-type functions. *Nordisk Tidskr. Informationsbehandling*, 2:69–75, 1962.

[4] G. A. Baker and P. R. Graves-Morris. *Padé Approximants : Basic Theory*. Addison-Wesley, 1981.

[5] M. Van Barel and A. Bultheel. Discrete linearized least squares rational approximation on the unit circle. *J. Comput. Appl. Math.*, 50:545–563, 1994.

[6] R. M. Barker, M. G. Cox, A. B. Forbes, and P. M. Harris. Software Support for Metrology Best Practice Guide 4: Modelling Discrete Data and Experimental Data Analysis. Technical report, National Physical Laboratory, Teddington, UK, 2004.

[7] I. Barrodale and J. C. Mason. Two simple algorithms for discrete rational approximation. *Mathematics Of Computation*, 24(112):877–891, 1970.

[8] I. Barrodale and C. Phillips. Solution of an overdetermined system of linear equations in the Chebyshev norm. *ACM Transactions on Mathematical Software*, 1(3):264–270, 1975.

[9] I. Barrodale and F. D. K. Roberts. An improved algorithm for discrete ℓ_1 approximation. *SIAM J. Numer. Anal.*, 10:839–848, 1973.

[10] R. Boudjemaa, A. B. Forbes, P. M. Harris, and S. Langdell. Multivariate empirical models and their use in metrology. Technical Report CMSC 32/03, National Physical Laboratory, Teddington, UK, 2003.

[11] E. W. Cheney. *Introduction to Approximation Theory.* McGraw Hill, 1966.

[12] E. W. Cheney and H. L. Loeb. Two new algorithms for rational approximation. *Numer. Math.*, 3:72–75, 1961.

[13] E. W. Cheney and M. J. D. Powell. The differential correction algorithm for generalized rational functions. *Constr. Approx.*, 3:249–256, 1987.

[14] E. W. Cheney and T. H. Southard. A survey of methods for rational approximation. *SIAM Review*, 5(3):219–231, 1963.

[15] Edwin K. P. Chong and Stanislaw H. Żak. *An Introduction to Optimization.* Wiley, 2001.

[16] A. A. M. Cuyt. The QD-algorithm and multivariate Padé-approximants. *Numer. Math*, 42:259–269, 1983.

[17] A. A. M. Cuyt and B. M. Verdonk. General order Newton-Padé approximants for multivariate functions. *Numer. Math*, 43:293–307, 1984.

[18] L. C. W. Dixon. *Nonlinear Optimization.* The English Universities Press Limited, 1972.

[19] D. J. Leeming E. H. Kaufman and G. D. Taylor. Uniform approximation on subsets of $[0, \infty)$ by reciprocals of polynomials. In *Approximation Theory IV*, pages 553–559. Academic Press, Inc, 1983.

[20] D. J. Leeming E. H. Kaufman and G. D. Taylor. Uniform approximation on subsets of $[0, \infty)$ by rational functions. In *Approximation Theory 5*, pages 407–410. Academic Press, Inc, 1986.

[21] G. E. Farin. *NURB curves and surfaces, from projective geometry to practical use.* A. K. Peters, Wellesley, Massachusetts, 1995.

[22] G. E. Forsythe. Generation and use of orthogonal polynomials for data fitting with a digital computer. *SIAM J.*, 5:74–88, 1957.

[23] Luca Gemignani. Chebyshev rational interpolation. *Numerical Algorithms*, 15:91–110, 1997.

[24] G. H. Golub and C. F. Van Loan. *Matrix Computations.* John Hopkins University Press, Baltimore, 3rd edition, 1996.

[25] M. Gugat. The Newton differential correction algorithm for rational Chebyshev approximation with constrained denominators. *Numerical Algorithms*, 13:107–122, 1996.

[26] M. Huhtanen and R. M. Larsen. On generating discrete orthogonal bivariate polynomials. *BIT*, 42:393–407, 2002.

[27] M. J. D. Powell I. Barrodale and F. D. K. Roberts. The differential correction algorithm for rational ℓ_∞ approximation. *SIAM J. Numer. Anal.*, 9(3):493–504, 1972.

[28] E. H. Kaufman Jr. and G. D. Taylor. Linearly constrained generalised rational approximation. In Charles K. Chui, L. L. Schumaker, and J. D. Ward, editors, *Approximation Theory 6*, volume 2, pages 353–356. Academic Press, Inc, 1989.

[29] E. H. Kauffman and G. D. Taylor. Uniform approximation by rational functions having restricted denominators. *J. Approx Theory*, 32:9–26, 1981.

[30] T. Kilgore. Rational approximation on infinite intervals. *Computers and Mathematics with Appl*, 48(9):1335–1344, 2004.

[31] C. L. Lawson. Contributions to the theory of linear least maximum approximation, Ph.D. thesis, UCLA, 1961.

[32] H. L. Loeb. On rational fraction approximations at discrete points. *Convair Astronautics Applied Mathematics, ser. 9*, 1957.

[33] Nathaniel Macon and D. E. Dupree. Existence and uniqueness of interpolating rational functions. *The American Mathematical Monthly*, 69(8):751–759, 1962.

[34] H. J. Maehly. Methods for fitting rational approximations, Part 1: Telescoping procedures for continued fractions. *J. ACM*, 7:150–162, 1960.

[35] H. J. Maehly. Methods for fitting rational approximations, Parts II and III. *J. ACM*, 10:257–277, 1963.

[36] D. W. Marquardt. An algorithm for least squares estimation of non-linear parameters. *SIAM J. Appl. Math.*, 11:431–441, 1963.

[37] J. C. Mason. Some new approximations for the solution of differential equations, DPhil thesis, The Queens College, Oxford, UK, 1965.

[38] J. C. Mason. Laurent-Padé approximants to four kinds of Chebyshev polynomial expansions part 1: Maehly type approximants. *Journal of Numerical Algorithms*, 38:3–18, 2005.

[39] J. C. Mason. Laurent-Padé approximants to four kinds of Chebyshev polynomial expansions part 2: Clenshaw-Lord type approximants. *Journal of Numerical Algorithms*, 38:19–29, 2005.

[40] J. C. Mason and D. C. Handscomb. *Chebyshev Polynomials*. Chapman and Hall / CRC Press, London, 2003.

[41] Jorg J. Moré. The Levenberg-Marquardt algorithm: Implementation and theory. *Lecture Notes on Mathematics*, 630:105–116, 1977.

[42] M. R. Osborne. Nonlinear least squares - the Levenberg algorithm revisited. *Journal of the Australian Mathematical Society*, 19:343–357, 1976.

[43] P. P. Petrushev and V. A. Popov. *Rational Approximation of Real Functions (Encyclopedia of mathematics and its applications: v. 28)*. Cambridge University Press, 1987.

[44] Les Piegl and Wayne Tiller. *The NURBS Book*. Springer, 2nd edition, 1997.

[45] Tomaso Pomentale. On discrete rational least squares approximation. *Numerische Mathematik*, 12:40–46, 1968.

[46] M. J. D. Powell. *Approximation Theory and Methods*. Cambridge University Press, 1981.

[47] M. J. D. Powell. The theory of radial basis function approximation in 1990. Technical Report DAMTP/1990/NA11, University of Cambridge, Cambridge UK, 1990.

[48] M. J. D. Powell. Recent research at Cambridge on radial basis functions. Technical Report DAMTP 1998/NA05, University of Cambridge, Cambridge UK, 1998.

[49] J. R. Rice and K. H. Usow. The Lawson algorithm and extensions. *Mathematics of Computation*, 22:118–127, 1968.

[50] Adrian J. Shepherd. *Second-Order Methods for Neural Networks.* Springer-Verlag, 1997.

[51] Jeiqing Tan and Yi Fang. Newton-Thiele's rational interpolants. *Numerical Algorithms*, 24:141–157, 2000.

[52] G. A. Watson. Discrete ℓ_p approximation by rational functions. In P. R. Graves-Morris, E. B. Saff, and R. S. Varga, editors, *Rational Approximation and Interpolation*, pages 489–501. Springer-Verlag, 1983.

[53] L. Wittmeyer. Rational approximation of empirical functions. *Nordisk Tidskr. Informationsbehandling*, 2:53–60, 1962.