Computational challenges in data mining

Markus Hegland *

(Received 7 August 2000)

Abstract

Data mining is applied in business to find new market opportunities from data stored in operational data bases which are used for day-to-day management. The tools applied combine ideas from statistics, machine learning, data base technology and high performance computing to find nuggets of knowledge. Data mining is also applied in science for example to find taxonomies of variable stars and in the national administration for the management of health care.

*Computer Sciences Laboratory, RSISE, Australian National University and Advanced Computational Systems CRC, Canberra ACT 0200, AUSTRALIA.
mailto:Markus.Hegland@anu.edu.au

[0]See http://anziamj.austms.org.au/V42/CTAC99/AHeg for this article and ancillary services, Published 27 Nov 2000.

Major computational challenges originate in the size of the data and its complexity. The analysis of complex or high-dimensional data suffers from the curse of dimensionality which is made worse if very large data sets have to be processed. Many current techniques are very good in dealing with high-dimensional data sets of moderate size or with very large data sets of moderate complexity but hardly any techniques are able to analyse very large data sets of high complexity.

The challenges are further explored and computational techniques are examined with respect to their capability to handle these challenges. It is seen in particular that finite element methods are very good in dealing with very large data sets but suffer under the curse of dimensionality and radial basis functions can deal with very high dimensions but not with very large data sets. Additive functions lead to models which can be used to analyse both high-dimensional and very complex data sets, in particular when parallel computers are used for their identification. Examples include multivariate adaptive regression splines.

Contents

1 Introduction

For also knowledge itself is power.
Francis Bacon (1597)

Mathematics, and in particular, computational mathematics, is a corner stone of industrial society. Most of the goods we consume have been designed, tested or even produced using computational systems which are ultimately based on mathematical principles. The proceedings of the CTAC conference document a diversity of applications and the techniques which have been used. The main focus of the applications has been on simulations of industrial and environmental systems and the models are able to incorporate increasing complexities. While simulations maintain their importance, the area of large

scale data analysis is becoming more important. The following example illustrates this trend.

In 1997 a chess game between the computer Deep Blue and the chess grandmaster Garry Kasparov ended for the first time with the victory of the computer [26]. While high-performance computing power was an important factor in achieving this, the determining factor was the ability of the computer to sift through hundreds of thousands of earlier moves for guidance. Like Deep Blue, business people look at their huge data collections of earlier transactions, their customer data bases, in order to detect a competitive edge and new business opportunities. This systematic pursuit of relevant information is also called business intelligence. Special techniques have been developed to find relevant business information. Many pitfalls both in respect to aspects of the analysis and computability are encountered, but, once found, like gold, the information quickly translates into big economic gains. This analogy with gold mining has led to the term "data mining" for the collection of the special techniques applied.

The origins of data mining have been traced back to the origins of collection of business data which was done as early as 5500 years ago by people of the Sumerian and Elam cultures [35], where tax records were written into dried mud tablets. Since then, methods for the exploitation of the recorded information with a view to life improvement and business success have been developed. To this day, tax records are a valuable asset of the taxation office for the detection of tax fraud.

A few examples shall illustrate where data mining has been used. Examples from business include direct marketing, product placing and product development. A good first introduction to both data mining concepts and business applications can be found in [9].

In addition to business problems, data mining is applied in many other areas. For example, the IBM "Advanced Scout" [13] data mining tool suggested that the inclusion of two "bench players" would help the "Orlando Magic" basketball team beat the competition. Giving these two players important roles did help the success of Orlando Magic. In the mid 90s all games of the NBA (National Basketball Association) were analysed by computational techniques in order to develop new strategies.

Another example where data mining is applied is in health services, for example in the management of diabetes. In the U.S.A. there are 300,000 people who suffer from insulin-dependent diabetes and 6,000,000 who have diabetes which does not require insulin for treatment. Important goals in the management of diabetes include the early detection of people at risk of having diabetes and the management of secondary illnesses like diabetes-related blindness, which is the largest cause of blindness in the U.S.A. The analysis of medical records revealed that regular checks of eyes, kidneys, feet and cholesterol levels were essential to controlling the secondary effects. Also, 92 percent of the costs were used for 5 percent of the patients which suggests that most emphasis should be put on the improvement of the treatment of the most severe conditions. This information gathered from the medical records did help the health agencies and it was also provided to the doctors in the

form of advice on best practice (Husum [27]).

Data mining is often seen as a subprocess of the larger process of *Knowledge Discovery in Databases* or KDD. KDD has been defined as the non-trivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data [20]. The knowledge discovery process starts with the identification of business goals, what should be achieved with the discovered information and the identification of what the information should be. Then the data sets are selected, including operational data of the company and demographic data, e.g., from the bureau of statistics. After the data has been selected it needs to be made accessible and then the lengthy stage of preprocessing starts. In the preprocessing stage a subset of the data may be chosen, data transformations may be performed, outliers detected and basic statistics of the data may be performed. After that the actual data mining process begins and finally the discoveries are implemented, after which a new cycle of data mining may start. It is estimated that only about 10 percent of the overall process is data mining [35]. Computational techniques are used both in the data mining and in the data preprocessing phase.

Due to its origins and by its very nature, data mining is interdisciplinary. It includes aspects of information technology, including the access of data in very large data bases, statistics almost in every step of the data analysis and a preparation phase. Machine learning techniques are some of the most popular ones used. In order to deal with the large amounts of data, parallel and high performance computers are used with algorithms which have been developed by computational mathematicians. Last but maybe most impor-

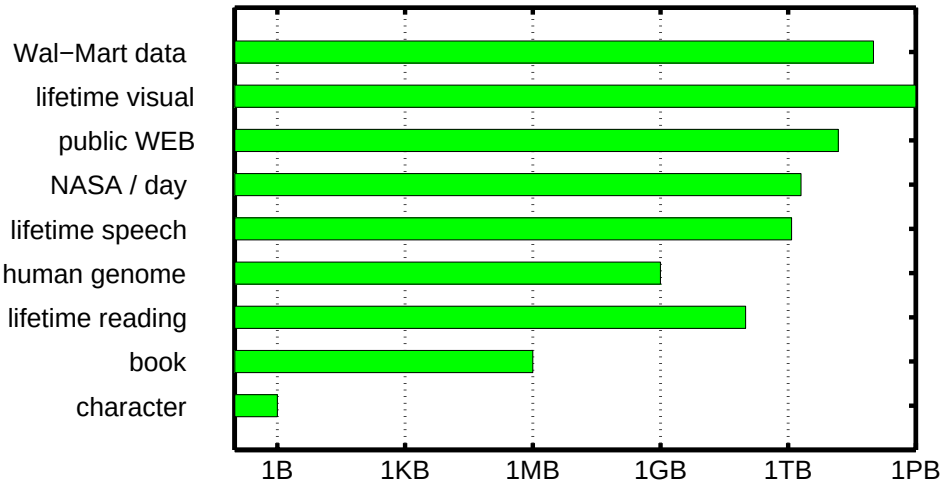

FIGURE 1: Data sizes in 1999 (partly from [7]).

tantly, the development and application of data mining techniques as well as the interpretation of the results does require a large portion of domain knowledge. Without the understanding of the business, one might find trivial or previously known information in the best case and draw totally wrong conclusions in the worst case.

The main driver for new computational techniques for data mining applications is the size of the data collections encountered. In 1999 we can

already see data warehouses (a systematic collection of data from heterogeneous sources) of the sizes of many terabytes (1 terabyte equals 10^{12} bytes). For example, Wal Mart, a large U.S.A. retailer, has just upgraded its data warehouse to hold 100 terabytes. See Figure 1 for some comparative data sets. A prime source of information, the current public Internet has been estimated to contain 10 terabytes and all the visual information a human encounters during a lifetime could be stored with a petabyte [7].

So the ultimate computational challenge of data mining is the very large data size. But we cannot assume that the data base is of fixed size as, however big, it grows with the company. It has been suggested that the data sizes grow at a similar rate as the hardware according to Moore's law, which means that the size would be doubled every 18 months [7]. This growth has a tremendous influence on the characteristics of the algorithms as they need to handle ever larger data sizes as well.

For example, assume that a data mining algorithm needs to solve a dense linear system of equations of size n, where n is the size of the data set. A direct solver requires $O(n^3)$ floating point operations, which is clearly not feasible if n is very large. Assume that this algorithm today would take 10 minutes. In 2010, according to Moore's law, computers will be 150 times faster but the data base will be 150 times larger as well. So an algorithm with $O(n)$ complexity would still take 10 minutes to process the entire data base whereas an algorithm with $O(n^3)$ complexity would require an entire year for the same task, see Figure 2. This has motivated our research into computational techniques which are scalable, i.e., have an $O(n)$ complexity and

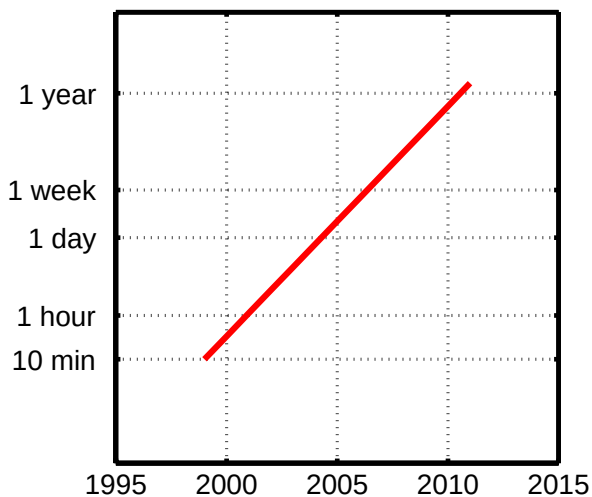

FIGURE 2: Time for an $O(n^3)$ algorithm for growing data bases and hardware speedup according to Moore's law.

are capable of handling complex data sets. In the following sections, these challenges will be further explored and some previous work in computational data mining will be reviewed.

An important class of tools provide predictive models which can then be either directly applied to predict likely customer behaviour or can be further analysed to find hidden structures in the data set. In particular, one is frequently interested to determine which variables do have an influence on the response at all. Examples include insurance companies which want to be able to predict how likely a new customer will lodge a claim. The taxation office wants to be able to predict if a subject is likely to commit taxation fraud. The ability to generalise observed behaviour to so far unseen instances is central to most data mining activities. Mathematically, this corresponds to the estimation of a function from data. For our purposes, we can characterise data as a collection of records. Each record contains predictor or independent variables $x_1, \ldots x_d$ and the response or dependent variable y. The predictor variables can be binary, categorical, ordered or real. From millions of records a functional relationship $y = f(x_1, \ldots, x_d)$ is then to be determined. Often a nonparametric (spline) method is used for the determination of f in order not to favour any particular behaviour. The techniques used to build models from data for both the independent variables x_i and the dependent variable y are termed supervised learning as the knowledge of y can be used to judge or supervise the model. In contrast to this are clustering techniques where the values of the y have to be extracted from the distribution of the x_i. Such techniques are called unsupervised. Typically there are tens to hundreds of predictor variables. Examples of

complex data include customer profiles of banks and insurances, multimedia and times series data. The number of features which describe these records can be from ten to over a thousand. The possible number of values of the (discretised) feature vectors grows exponentially with the dimension. This has been called the *curse of dimensionality* [8], and, combined with the largeness of the data sets poses the major computational challenge of data mining.

Here, we investigate computational aspects of the determination of f. An essential tool is approximation. Furthermore the determination of f from the data should be stable with respect to data errors and incomplete information. Typically, there is a trade-off between approximation and stability (or between bias and variance) and this is just another instance of the famous *Lax equivalence theorem*:

> The combination of consistency and stability is equivalent to convergence [36].

The approximation question deals with how to choose the function class from which a function is selected. Examples include neural nets, linear functions, radial basis function approximations and functions with bounded higher order derivatives. The choice of this function class does have a large influence on the approximation error, also called bias, and it provides a lower limit for the error. While the choice of the function class does have important implications for the approximation, we will mainly deal with computational

issues here and we will refer the interested reader to the literature when we deal with particular classes of functions.

In Section 2, a class of techniques based on radial basis functions which are known to effectively deal with the curse of dimensionality are shown to be limited to medium sized data sets as they are not scalable. In Section 3 we see that an important tool in simulation, the finite element method, is scalable with the number of data points and can be used effectively for modelling very large data sets. An alternative approach, based on regression splines is reviewed in Section 5. Maybe the most effective approach for extremely large and complex data sets is based on additive models and a parallel algorithm and feature selection is discussed in Section 4.

2 Computational issues in the determination of high dimensional predictors using radial basis functions

In order to explore the challenge posed by the curse of dimensionality further, we will investigate radial basis function approximation. In recent years, radial basis functions have received a lot of attention both theoretically and in applications. One of their outstanding features is that they are able to approximate high-dimensional functions very effectively. Thus they seem to be able to overcome the curse of dimensionality. In the case of real attributes

$x \in \mathrm{R}^d$ a radial basis function is of the form

$$f(x) = \sum_{i=1}^{n} c_i \, \rho(\|x - x^{(i)}\|) + p(x),$$

where $x^{(i)}$ are the data points. Examples for the function ρ include Gaussians $\rho(r) = \exp(-\alpha r^2)$, powers and thin plate splines $\rho(r) = r^\beta$ and $\rho(r) = r^\beta \ln(r)$ (for even integers β only), multiquadrics $\rho(r) = (r^2 + c^2)^{\beta/2}$ and others. The function $p(x)$ is typically a polynomial of low degree and in many cases it is zero. The radial basis function approach may be generalised to metric spaces where the argument of ρ is replaced by the distance $d(x, x_i)$. Reviews on radial basis function research can be found in [18, 31, 12]. Existence, uniqueness and approximation properties have been well studied.

The evaluation of $f(x)$ requires the computation of the distances between x and all the data points $x^{(i)}$. Thus the time required to compute one function value is $O(dn)$; the complexity is linear in the number of attributes d and the curse of dimensionality has been overcome. However, if many function values need to be evaluated, this is still very expensive. Fast methods for evaluation of radial basis functions have been studied by Beatson, Light, Newsam and Powell [3, 4, 5, 6]. For example, multipole method which reduces the complexity to $O((m + n)\log(n))$ has been suggested in [5, 14] for the evaluation of $f(x)$ for m values of x. For data mining applications, which have very large n, even this is still too expensive. What is required is an approximation for which the evaluation is independent of the data size n and does not suffer under the curse of dimensionality. In the following, we

will revisit the determination of the function from the data points and see how the geometry of high-dimensional spaces influences the computational costs.

The vector of coefficients of the radial basis functions $\mathbf{c} = (c_1, \ldots, c_n)$ and the vector of coefficients of the polynomial term $\mathbf{d} = (d_1, \ldots, d_m)$ are determined, in the case of smoothing, by a linear system of equations of the form:

$$\begin{bmatrix} A + \alpha I & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \mathbf{d} \end{bmatrix} = \begin{bmatrix} \mathbf{y} \\ 0 \end{bmatrix}. \tag{1}$$

The matrix $A = \left[\rho(\|x^{(i)} - x^{(j)}\|)\right]_{i,j=1\ldots n}$ has very few zero elements for the case of the thin plate splines and has to be treated as a dense matrix. However, the influence of the data points is local and mainly the observed points close to x do have an influence on the value of $f(x)$. This locality is shared with the nearest neighbour approximation techniques. However, in higher dimensions points get more sparse. For example, it is observed in [22] that the expected distance of the nearest neighbour in a d-dimensional hypercube grows like $O(n^{-1/d})$ with the dimension and that large numbers of data points are required to maintain a uniform coverage of the domain. In particular, a constant distance between a point and its nearest neighbour is obtained if $\log(n) = O(d)$, i.e., the number of points has to grow exponentially with the dimension. This is just another aspect of the curse of dimensionality.

If the function to be approximated is smooth enough the number of points available may be sufficient even in high dimensions. But a computational dif-

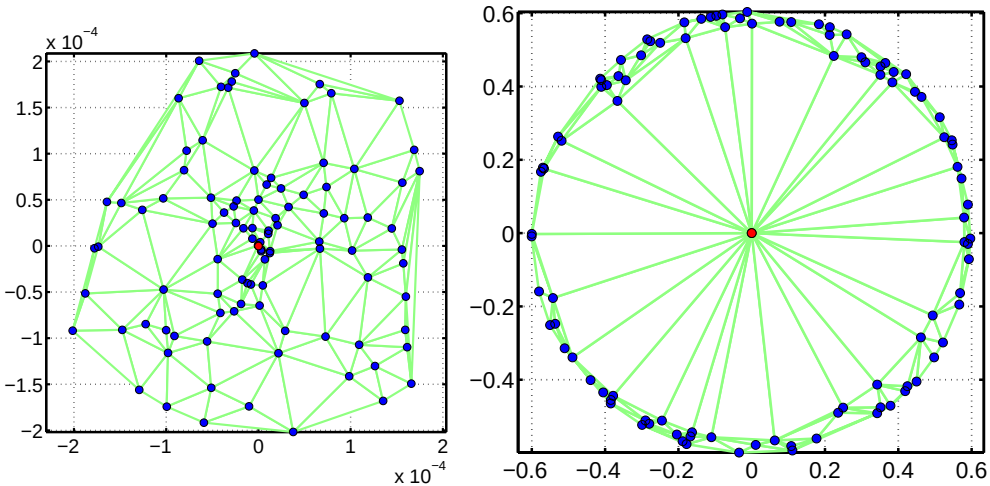

FIGURE 3: 100 nearest neighbours from 10^6 normal i.i.d. points. (Distance preserving mapping from 2D (left) and 100D (right).)

ficulty appears which is related to the concentration of measure [37]. The concentration of measure basically tells us that in high dimensions the neighbours of any point are concentrated close to a sphere around that point. This is illustrated in Figure 3, where the 100 nearest neighbours of a random point from a population of a Million normally distributed points are chosen in two and in one hundred dimensions. For the hundred-dimensional case the neighbouring points are mapped onto 2 dimensions in a random way such that the distance to the original point are preserved. One can clearly see that the

nearest neighbours are further away in higher dimensions but, more importantly, that the distance to most of them is very similar. This is the measure concentration in action. A very large number of neighbours may have to be considered for a good approximation.

The effect of this on the computation is severe as the determination of a good approximant will require visiting a large number of neighbouring points for each evaluation point. In an attempt to decrease the computational work, a compactly supported radial basis function may be used. However, the support will have to be chosen such that for a very large number of points the values of the radial basis function $\rho(\|x^{(i)} - x^{(j)}\|)$ will be nonzero. Thus the linear system of equations (1) will have a substantial number of nonzeros which ultimately will render the solution computationally infeasible. This is one motivation to find alternatives to radial basis functions with similar properties.

3 Finite elements are scalable

Thin-plate splines [17] are an important example of radial basis functions which provide smooth approximations. In the one-dimensional case, thin-plate splines are the piecewise cubic smoothing splines. They are computationally very tractable as they can be represented with the local B-spline basis [16]. However, explicit representations are also known in higher dimen-

sions. In two dimensions, for example, one has

$$f(x_1, x_2) = c_0 + c_1 x_1 + c_2 x_2 + \sum_{k=1}^{n} b_k \, \phi\left((x_1 - x_1^{(k)})^2 + (x_2 - x_2^{(k)})^2\right)$$

where $\phi(r^2) = r^2 \log(r^2)$, for higher dimensions see [39].

The coefficients of the thin plate splines are determined by a linear system of equations of the form

$$\begin{bmatrix} \Phi + \alpha I & X \\ X^T & 0 \end{bmatrix} \begin{bmatrix} b \\ c \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix}$$

where Φ is the n by n matrix with elements $\Phi_{i,j} = \phi(\|x^{(i)} - x^{(j)}\|^2)$, I is the identity, X is the n by 3 matrix with the i-th row containing the values 1, $x_1^{(i)}$ and $x_2^{(i)}$ (for the two-dimensional case), $b = (b_1, \ldots, b_n)^T$ and $c = (c_0, c_1, c_2)^T$.

Computationally, these equations are intractable for large data sizes n by standard direct or iterative methods, as even the formation of the matrix Φ requires $O(n^2)$ operations as it is dense. The standard techniques are thus *not scalable* in the data size. A few years ago it was thought that the feasibility of thin plate splines (and similar radial-basis function approaches) was limited to the case of a few hundred to thousand observations. However, new techniques have been developed since then which pushed these limits further. One school of thought uses the locality of the problem, i.e., the fact that the value $f(x)$ depends only on observations $x^{(k)}$ which are close to x.

This approach has successfully been used for interpolation [6, 3, 4, 19, 34, 33, 32], i.e., the case of $\alpha = 0$.

We have developed a different approach which is provably scalable and can be extended to higher dimensions as well. First, observe that the thin plate spline minimizes the functional

$$J_1(f) = \sum_{k=1}^{n} (f(x^{(k)}) - y^{(k)})^2 + \alpha \int_{\mathrm{R}} \left(\frac{\partial^2 f}{\partial x_1^2}\right)^2 + 2\left(\frac{\partial^2 f}{\partial x_1 \partial x_2}\right)^2 + \left(\frac{\partial^2 f}{\partial x_2^2}\right)^2 dx.$$

The minimum of this functional is approximated using a non-conforming finite-element space with piecewise bilinear functions on rectangular elements. In order to deal with the missing smoothness of the finite element space the gradients of f are approximated as piecewise bilinear functions as well. Instead of J_1 the following function is minimized (which is obtained by inserting the gradient in J_1):

$$J_2(f, u_1, u_2) = \sum_{k=1}^{n} (f(x^{(k)}) - y^{(k)})^2 + \alpha \int_{\mathrm{R}^2} \left(\left(\frac{\partial u_1}{\partial x_1}\right)^2 + \left(\frac{\partial u_1}{\partial x_2}\right)^2 + \left(\frac{\partial u_2}{\partial x_1}\right)^2 + \left(\frac{\partial u_2}{\partial x_2}\right)^2\right) dx.$$

The f-component of the minimum of J_2 is a thin plate spline if the $u = (u_1, u_2)$-component is defined by $\operatorname{curl} u = 0$ and the Neumann boundary value problem

$$\Delta f(x) = \operatorname{div} u(x), \quad x \in G$$

with the boundary conditions

$$\frac{\partial f}{\partial n}(x) = u_n(x), \quad x \in \partial G$$

where G is the domain containing the data points $x^{(k)}$. Practical tests showed that the curl condition was not important for a good approximation [29]. It can be seen that the curl condition may be replaced by tangential boundary conditions for u and, as the approximation is local, the effect of the boundary conditions does not have a big influence on the function values in the interior of the domain.

The finite element solution of the optimization problem proceeds in two stages:

1. The matrix and right-hand side of the linear system of equations is assembled. The matrix of this linear system is the sum of low rank matrices, one for each data point $x^{(i)}$.

2. The linear system of equations is solved.

The time for the first (assembly) stage depends linearly on the data size n and the time for the second (solution) stage is independent of n. Thus the overall algorithm scales with the number of data points. The data points only need to be visited once, so there is no need to either store the entire data set in memory nor revisit the data points several times. The basis

functions are piecewise bilinear and require a small number operations for their evaluation. With this technique the smoothing of millions of data points becomes feasible.

For increased performance and reduced response time, the parallelisation of this algorithm has been investigated in [15]. The parallel algorithm exploits different aspects of the problem for the assembly and the solution stage. The time required for the assembly stage grows linearly as a function of data size. For simplicity we assume that the data is initially equally distributed between the local disks of the processors. (If this is not the case initial distribution costs would have to be included in the analysis.) In a first step of the assembly stage a local matrix is assembled for each processor based on the data available on its local disk. The matrix of the full problem is then the sum of the local matrices and can thus be obtained through a reduction step. This algorithm was developed and tested on a cluster of 10 Sun Sparc-5 workstations networked with a 10 Mbit/s twisted pair Ethernet using MPI on a SUN SMP with 10 processors [15].

If the finite element solution of the Neumann boundary value problem is denoted by $f_h = Ku_h$ one can formulate the optimisation problem as a quadratic minimisation problem for the gradient u_h where

$$u_h^\alpha = \text{argmin } \|Ku_h - b\|^2 + \alpha u_h^T C u_h,$$

where C is the matrix which discretises the penalty term of J_2. This problem is then solved with a Krylov space approximation where the Krylov space is

spanned by

$$C^{-1}d,\ (C^{-1}M)C^{-1}d,\ (C^{-1}M)^2C^{-1}d,\ \ldots$$

for some initial vector d and $M = K^TK$. In the Krylov space basis (which is orthogonal to the scalar product defined by the matrix C), the minimisation problem then leads to a tridiagonal linear system of equations

$$(T + \alpha I)w = \|C^{-1}d\|e_1.$$

Note that the Krylov space is independent of the parameter α. Thus the minimisation problem can be solved for multiple α without having to regenerate the Krylov vectors, i.e., redo expensive matrix vector operations. This is important as the parameter α is determined with generalised cross validation [39, 28], which leads to an algorithm which requires the solution of the minimisation problem multiple times. When this approach is generalised to higher dimensions two challenges have to be overcome:

- The standard tensor-product finite element space of piecewise multilinear functions suffers from the curse of dimensionality. Work is in progress to address this curse using subspaces of the finite element space which have comparable approximation properties.

- The penalty function used for the thin plate spline in two dimensions does not produce smooth functions in higher than 3 dimensions. Thus one either has to use higher order derivatives [39], which would require the introduction of approximate higher derivatives of f in addition to

the gradient u, or tensor product spaces, or additive models of the type

$$f(x) = f_0 + \sum_{i=1}^{d} f_i(x_i) + \sum_{i,j=1}^{d} f_{i,j}(x_i, x_j).$$

These techniques are currently under investigation.

4 B-MARS

Tree-structured techniques are known to be good candidates for dealing with high dimensional data and they also provide easily interpretable models. They have been very successful in classification [11] in the form of binary tree structured classifiers. The domain is repeatedly split into subdomains. For the finest levels—in the smallest subdomains—the classifier returns a single class label. Each split is orthogonal to one axis x_k such that all the points with $x_k \leq \xi$ belong to one subset and all the points with $x_k > \xi$ belong to the other. The index k and the position ξ together with the parent subdomain determine this split. These parameters are selected such that the corresponding split has the largest impact on the quality of the resulting classifier. Thus this is a greedy algorithm and there might be some concern that a suboptimal partitioning may be found. However, the more important question is where to stop the partitioning. Good results have been reported for the generation of very deep trees and successive pruning based on an

estimator of the error of the classifier which may use a test data set, or, alternatively, cross-validation. The resulting classification or decision trees are one of the most important tools in data mining. Not only do they provide good predictive models, but they are also interpretable in the sense that they allow further analysis of the domain based on the values of the independent variables. For example, one may be able to characterise all customers which do have a certain property, e.g., are likely to respond to a certain marketing campaign.

Regression trees [11] implement the same ideas for regression, i.e., for real response variables. They do have the same advantages, and, in fact, they may be used with logistic regression for the determination classification probabilities. In contrast to finite element techniques based on tensor products of one-dimensional spaces, regression trees are extremely successful in dealing with the curse of dimensionality as they have a complexity proportional to the dimension. They have two shortcomings, however: They do not represent linear functions well and, more generally, they can not approximate smooth functions accurately as they are discontinuous, piecewise constant functions.

Better approximation properties are obtained by piecewise polynomial functions. They are used in the MARS algorithm (*Multivariate Adaptive Regression Splines*) [23]. Instead of explicitly generating a hierarchy of domains, MARS generates a hierarchy of basis functions which implicitly define a hierarchy of domains. An example of a basis function hierarchy is displayed in Figure 4. At the root of the tree is the constant basis function $B_0 \equiv 1$. At

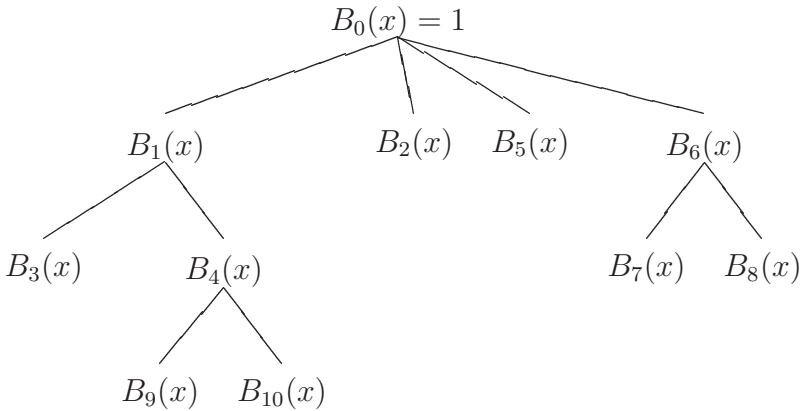

FIGURE 4: Hierarchy of MARS basis functions.

each "partitioning" step two new children are generated:

$$B_{\text{child}_1} = B_{\text{parent}}(x)(x_j - \xi)_+ \quad \text{and} \quad B_{\text{child}_2} = B_{\text{parent}}(x)(-x_j + \xi)_+$$

where $(z)_+$ denotes the usual truncated linear function. It is equal to z for $z > 0$ and equal to zero if $z < 0$. The parent, the variable x_j and the value ξ are all chosen such that the sum of squared residuals is minimised.

While each node can have offsprings several times there are some rules for the generation of this tree:

- The depth of the tree is bounded, typically by a value of 5 or less. This is thought to be sufficient for practical purposes [23] and the bound is

important to control the computational work required for the determination of the function values.

- A variable x_j is only allowed at most once in a factor of a basis function B_k. This guarantees that the function is piecewise multilinear.

The partitioning of the domain is defined such that on each partition the function is multilinear. Thus this partitioning does not have the same interpretative value as in the case of classification and is not recovered by the algorithm. However, the MARS method generates an ANOVA decomposition of the form [23]

$$f(x) = f_0 + \sum_{i=1}^{d} f_i(x_i) + \sum_{i,j=1}^{d} f_{i,j}(x_i, x_j) + \sum_{i,j,k=1}^{d} f_{i,j,k}(x_i, x_j, x_k) + \cdots .$$

This model may form the basis for further analysis, as it provides information on which variables may be most important for modelling and which combinations of variables interact. Similar models are obtained in the analysis of variance, hence the name ANOVA decomposition.

Each sum in the ANOVA decomposition corresponds to a level in the tree of the basis functions and thus, by limiting the depth of the tree, the number of variables occurring in the functions is limited. This does avoid the curse of dimensionality.

The computational complexity of the algorithm, after some clever updating ideas have been applied, is shown in [23] to be $O(dnM_{\max}^4/L)$ where d is

the dimension, n the number of data points, $M_{\max}$ the number of basis functions considered—some might not be used later because of pruning—and L is the number of levels of the tree. Thus the algorithm is scalable, i.e., the complexity is proportional to the data size. However, the proportionality constant can be very large due to the dependence of $O(M^4)$, which limits the number of basis functions which can be used and thus limits the approximation power of this approach. Another limitation is due to the fact that a greedy algorithm is used and the choice of basis functions may not be a global optimum. However, we think that more research into the computational performance of the MARS algorithm is required.

The basic one-dimensional functions used are truncated powers. It is known [16] that such functions lead to ill-conditioned linear systems of equations. Furthermore, the evaluation at a certain point may require the computation of many terms and cancellation errors are likely to occur especially if highly local effects are modelled. Finally, the coefficients of the basis functions only provide information about the kind of interaction but are non-informative otherwise. It would be useful if some information about the behaviour of the function could be obtained from the coefficients directly. Finally, in order to deal with very large data sizes, parallel processing becomes a necessity.

A new method called B-MARS [1] improves on MARS in all these aspects. First, it is based on hat-functions, the simplest type of continuous B-splines. The advantage of hat-functions is that they are local. From this, compared to the original MARS algorithm one gets the following advantages:

1. The equations are well conditioned.
2. The determination of a function value only requires the evaluation of a limited number of basis functions which leads to a reduction in both cancellation errors and evaluation time.
3. Finally, the coefficients of the basis functions are now good approximations of the function value.

In the case of the MARS algorithm each pair of one-dimensional basis functions is determined by one parameter ξ. This makes the implementation very efficient. However, in the case of hat functions, two parameters are required, a location and a width of the hat function. The scale parameter is determined in a hierarchical way. First an initial scale is chosen, so only the position needs to be found. After the approximation power of a particular scale has been exhausted, i.e., if a particular scale is unable to improve the fit, a next smaller scale is chosen which is a fraction $1/K$ of the original scale. In this way, scale is incorporated in a natural way into the hierarchy of basis functions. A side benefit of this approach is that there is a considerable reduction in the candidate basis functions which need to be tested at each iteration. Practical tests have confirmed that this reduction in complexity does not reduce the quality of the resulting models. On the contrary, it was found that small local nonconformities in the functions were better resolved in B-MARS [2].

Both MARS and B-MARS are scalable, i.e., have a complexity which is linear in the number of data points. However, as the number of data points

can be very large, parallel processing is necessary in order to avoid extreme computing times. (For example, the analysis of an insurance data set with a few millions of records required several hours processing time.) It has been found that the bulk of the computations going on are used to compute least squares fits which requires large numbers of scalar products. Using a data partitioning approach, these scalar products are processed in parallel, which resulted in an algorithm with very high parallel efficiency [2].

5 Additive Models

In the case where the tree of basis functions generated by MARS and B-MARS contains two levels, namely, the root $B_0 \equiv 1$ and its children, a model of the following form is generated:

$$f(x_1, \ldots, x_d) = f_0 + \sum_{i=1}^{d} f_i(x_i).$$

The univariate components f_i of this additive model are piecewise linear in the case of MARS. Other commonly used additive models are based on smoothing splines and local parametric (polynomial) approximations [24]. A unified treatment for all these approximations reveals that in all cases additive models are scalable with respect to data size and, like the multivariate regression splines, conquer the curse of dimensionality. While additive models are computationally very competitive they also show good performance in

practical applications. Like other regression models they provide good classification methods, especially if logistic regression is used. These *generalised additive models* can be seen to form the foundation of *boosting*, which is a very effective technique to improve simple classifiers [21].

If the probability distribution of the random variables $(X_1, \ldots, X_d, Y)$ which model the observations is known, the best (in the sense of expected squared error) approximation for the additive model is obtained when

$$f_i(x_i) = E(Y - f_0 - \sum_{\substack{k=1 \\ k \neq i}}^{d} f_k(X_k) \mid X_i = x_i) \tag{2}$$

and $f_0 = E(Y)$. (The symbol $E(Y \mid X = x)$ denotes the conditional expectation of the random variable Y when $X = x$.) These equations do not have a unique solution, but uniqueness can be easily obtained if one introduces constraints like $E(f_i(X_i)) = 0$.

In practical algorithms, the estimates are approximated by smooth functions. Let $\mathbf{f}_i$ be the vector of function values $\left(f_i(x_i^{(k)})\right)_{k=1}^{n}$. Furthermore, let S_i be the matrix representing the mapping between the data and the smooth $\mathbf{f}_i$. The matrix S_i depends on the observations $x_i^{(1)}, \ldots, x_i^{(n)}$. Replacement

of the estimation operator by the matrix S_i in equation (2) leads to

$$\begin{bmatrix} I & S_1 & \cdots & S_1 \\ S_2 & I & \cdots & S_2 \\ \vdots & \vdots & & \vdots \\ S_d & S_d & \cdots & I \end{bmatrix} \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \vdots \\ \mathbf{f}_d \end{bmatrix} = \begin{bmatrix} S_1\mathbf{y} \\ S_2\mathbf{y} \\ \vdots \\ S_d\mathbf{y} \end{bmatrix}.$$

If the eigenvalues of the S_i are in $(0, 1)$ and this linear system of equations is nonsingular, it can be seen that the Gauss-Seidel algorithm for this system converges. This method of determining the additive model is the *backfitting algorithm* [24]; it has complexity $O(nqd)$ where q denotes the number of iteration steps. The backfitting algorithm is very general and, in fact, is used even for nonlinear smoothers. For very large data sets, however, the algorithm becomes very costly—even though it is scalable in the data size and does not suffer from the curse of dimensionality—because it needs to revisit the data qd times.

The high cost of the solution of the previous linear system of equations resulted from the large size of its matrix. Smaller alternative systems are available for particular cases of smoothers S_i. For example, in the case of regression splines one can work with the system of normal equations. Consider the case of fitting with piecewise multilinear functions. The functions include the ones used in MARS and are, on each subdomain, linear combinations of products $x_{j_1} \cdots x_{j_k}$ where every variable x_j occurs at most once.

The basis functions of the full space of piecewise multilinear functions are products of hat functions of the form $b_1(x_1) \cdots b_d(x_d)$. For the full space,

the normal matrix of the least squares fitting problem becomes sparse with nonzeros on 3^d of the diagonals. But the matrix is huge, being of order m^d if each of the d dimensions is discretised by m hat functions. For a four-dimensional problem with $m = 10$, the nonzero structure of this matrix is displayed on the left in Figure 5.

MARS generates additive models consisting of sums of piecewise linear functions $f_i(x_i)$. Assume again that in each of the d dimensions, one has m hat functions as basis. Then the normal matrix of the regression problem is of order md which becomes manageable. However, closer inspection shows that this matrix is mostly dense. The nonzero structure for the case $m = 10$ and $d = 4$ is found on the right side of Figure 5. A direct solver would require $O(d^3m^3)$ operations. But any solver for the problem in the full space would have a cost at least of $O(3^dm^d)$ if all matrix elements are involved in the computation. For very large data sets the time to solve the normal equations for the additive model are also less than the time required for backfitting.

We have developed a parallel variant of the backfitting algorithm [25] for the determination of the components f_i of an additive model which is adapted to the analysis of very large data sets. The algorithm has four steps:

1. An equal number of data records is assigned to each processor in a random way.

2. Each processor determines an additive model for its partition. We have used local linear fitting as a smoother with a small bandwidth which

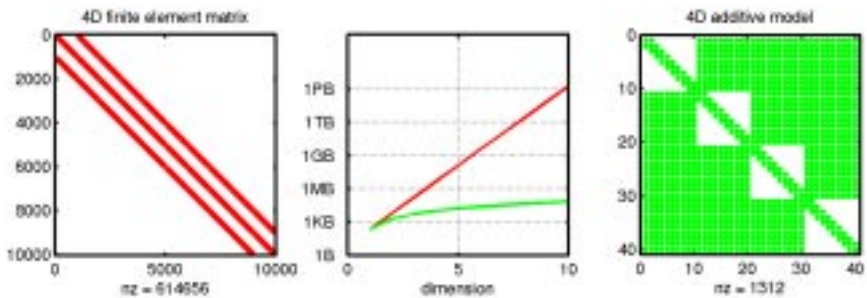

FIGURE 5: Left and right diagrams: The sparseness patterns of a $10 \times 10 \times 10 \times 10$ piecewise multilinear finite element matrix and the corresponding additive model matrix (nz = number of nonzeros). Middle diagram: The number of nonzeros of the finite element matrix and the additive model matrix as a function of dimension.

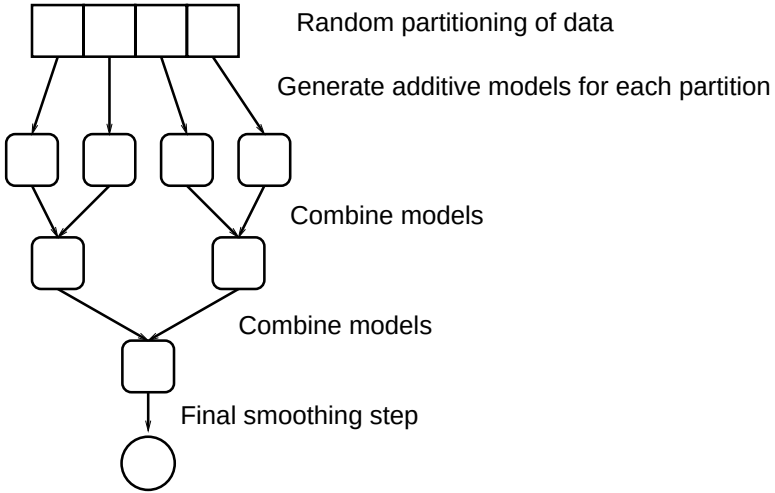

FIGURE 6: Parallel algorithm to generate additive model

has relatively high variance but low bias. The smoother is computed on a fine equidistant grid.

3. The additive models of the different partitions are combined.

4. In a final smoothing step the variance is reduced.

Figure 6 shows schematically these stages of the parallel additive algorithm. In practical analysis of data from insurance, we showed that this algorithm is substantially faster than the original MARS algorithm: on a SUN Enterprise

with 8 processors a generalised additive model was generated in 8 minutes for data with 1.5 million records and 17 attributes or variables where the original MARS algorithm required several hours. The algorithm is scalable in the number of observations and with respect to the number of processors.

In most practical applications not all observed variables are relevant and the values of some of the variables might depend on other variables and thus provide redundant information. The predictor variables need to be selected according to their importance in modelling the response. The LASSO algorithm [38] provides a mechanism for parameter selection for linear models. (Note that we use parameter here to denote the β_j to distinguish them from the variables x_i.) As the additive model of MARS is a linear model, LASSO can be applied here as well. Depending on the values of x_j this may result in a model which includes all variables x_j.

A model with less variables is obtained, if the parameters are treated in groups corresponding to the variables. For example, let the $\beta_{k,j}$ be the parameters characterising the function f_k:

$$f_k(x_k) = \sum_{j=1}^{p_k} \beta_{k,j} B_{k,j}(x_k)$$

where $B_{k,j}$ is the j-th basis function (which is a hat function) used to model f_k. We suggest a new block-variation of LASSO [2] where the functions f_k are

determined by

$$\beta = \operatorname{argmin} \sum_{i=1}^{n} (y^{(i)} - \sum_{k,j} \beta_{k,j} B_{k,j}(x_k^{(i)}))^2$$

subject to the constraint

$$\sum_{k=1}^{d} \sqrt{\sum_{j=1}^{p_k} \beta_{k,j}^2} \leq t.$$

For one-dimensional functions this is ridge regression and for the case of linear models (where each f_k is determined by one parameter) this is the LASSO algorithm. As for LASSO, the choice of the parameter t guides the selection of the variables. It has been demonstrated [2] that a small enough choice of t causes only one variable to be selected. Obviously, if t is large enough, the constraint is not active in the case of non-singular equations and one obtains a least squares problem.

One way to improve additive models is to include interaction terms $f_{i,j}$. It is thought that these interaction terms may have lower spatial resolution in each dimension than the one-dimensional functions. For some first ideas, see [30] in this proceedings.

6 Conclusion

Data mining is a new area of research, especially for computational mathematicians. It is becoming an important activity for many companies, especially in the service sector. The main computational problems relate to the size of the data sets encountered and to the complexity of the data. Traditional computational techniques like finite elements and splines are capable of handling the computational challenges. New methods like additive models are being tested.

At the ANU, a small team is working on computational techniques for data mining applications which are scalable with respect to the data size, deal with high dimensionality and use parallel computers. In future work we will further investigate the usage of wavelets and look at other applications like density estimation, clustering and time series analysis.

A different discussion of some of the challenges from the point of view of mathematical programming can be found in [10].

Acknowledgements: The research in this report was a joint effort by the author and P. Christen, W. Clarke, G. Hooker, I. McIntosh, O. Nielsen, N. Potter, S. Roberts, V. Pestov, P. Williams and C. Zoppou from the Computer Sciences Laboratory at the ANU, by S. Bakin and B. Turlach from the School of Mathematical Sciences at the ANU and by I. Altas from the Charles Sturt University and K. Burrage and R. Sidje from the University

of Queensland in Brisbane. Many other contributed their ideas and support. Some of the research is supported by the ADVANCED COMPUTATIONAL SYSTEMS CRC.

References

[1] S. Bakin, M. Hegland, and M. Osborne. Can MARS be improved with B-splines? In B.J. Noye, M.D. Teubner, and A.W. Gill, editors, *Computational Techniques and Applications: CTAC97*, pages 75–82, Singapore, 1998. World Scientific Publishing Co. C26

[2] S. Bakin. *Adaptive Regression and Model Selection in Data Mining Problems*. PhD thesis, ANU, 1999. C27, C28, C34, C35

[3] R.K. Beatson, G. Goodsell, and M.J.D. Powell. On multigrid techniques for thin plate spline interpolation in two dimensions. In *The Mathematics of Numerical Analysis (Park City, UT, 1995)*, volume 32 of *Lectures in Appl. Math.*, pages 77–97. Amer. Math. Soc., Providence, RI, 1996. C13, C18

[4] R.K. Beatson and W.A. Light. Fast evaluation of radial basis functions: methods for two-dimensional polyharmonic splines. *IMA J. Numer. Anal.*, 17(3):343–372, 1997. C13, C18

[5] R.K. Beatson and G.N. Newsam. Fast evaluation of radial basis functions. I. *Comput. Math. Appl.*, 24(12):7–19, 1992. Advances in the theory and applications of radial basis functions. C13, C13

[6] R.K. Beatson and M.J.D. Powell. An iterative method for thin plate spline interpolation that employs approximations to Lagrange functions. In *Numerical Analysis 1993 (Dundee, 1993)*, volume 303 of *Pitman Res. Notes Math. Ser.*, pages 17–39. Longman Sci. Tech., Harlow, 1994. C13, C18

[7] G. Bell and J.N. Gray. The revolution yet to happen. In P.J. Denning and R.M. Metcalfe, editors, *Beyond Calculation*, pages 5–32. Springer Verlag, 1997. C7, C8, C8

[8] R.E. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1991. C11

[9] M.J.A. Berry and G. Linoff. *Data Mining Techniques: For Marketing, Sales and Customer Support*. John Wiley and Sons, 1997. C5

[10] P.S. Bradley, U.M. Fayyad, and O.L. Mangasarian. Mathematical programming for data mining: formulations and challenges. *INFORMS J. Comput.*, 11(3):217–238, 1999. C36

[11] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Chapman and Hall/CRC, first reprint 1998. C22, C23

[12] M.D. Buhmann. New developments in the theory of radial basis function interpolation. In *Multivariate Approximation: from CAGD to Wavelets (Santiago, 1992)*, volume 3 of *Ser. Approx. Decompos.*, pages 35–75. World Sci. Publishing, River Edge, NJ, 1993. C13

[13] P. Cabena, P. Hadjinian, R. Stadler, J. Verhees, and A. Zanasi. *Discovering Data Mining—From Concept to Implementation*. Prentice Hall, 1977. C5

[14] J.B. Cherrie, R.K. Beatson, and G.N. Newsam. A Fast evaluation of radial basis functions: Methods for generalised multiquadrics in R^n. University of Canterbury, Technical Report,UCDMS2000/1 January 2000. C13

[15] P. Christen, I. Altas, M. Hegland, S. Roberts, K. Burrage, and R. Sidje. A parallel finite element surface fitting algorithm for data mining. Technical report, ANU, 1999. C20, C20

[16] C. de Boor. *A Practical Guide to Splines*. Springer, 1978. C16, C26

[17] J. Duchon. Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In *Constructive Theory of Functions of Several Variables (Proc. Conf., Math. Res. Inst., Oberwolfach, 1976)*, pages 85–100. Lecture Notes in Math., Vol. 571. Springer, Berlin, 1977. C16

[18] N. Dyn. *Interpolation and Approximation by Radial and Related Functions*, volume 1, pages 211–234. Academic Press, 1989. C13

[19] A.C. Faul and M.J.D. Powell. Proof of convergence of an iterative technique for thin plate spline interpolation in two dimensions. *Adv. Comput. Math.*, 11(2-3):183–192, 1999. Radial basis functions and their applications. C18

[20] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. In *Advances in Knowledge Discovery and Data Mining*, page 1. MIT PRESS, 1996. C6

[21] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. Technical report, Department of Statistics, Stanford University, 1998. C29

[22] J.H. Friedman. Flexible metric nearest neighbor classification. Technical report, Department of Statistics, Stanford University, 1994. C14

[23] J.H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19:1–141, 1991. C23, C24, C25, C25

[24] T.J. Hastie and R.J. Tibshirani. *Generalized additive models*, volume 43 of *Monographs on statistics and applied probability*. Chapman and Hall, 1990. C28, C30

[25] M. Hegland, I. McIntosh, and B.A. Turlach. A parallel solver for generalised additive models. *Computational Statistics & Data Analysis*, 31(4):377–396, 1999. C31

[26] P.M. Horn. Deep computing. In A. Redelfs, editor, *High Performance Computing – Contributions to Society*, pages 62–68. Tabor Griffins Communications Inc., 1998. C4

[27] D. Husum. Providing better managed care. In A. Redelfs, editor, *High Performance Computing – Contributions to Society*, pages 70–72. Tabor Griffins Communications Inc., 1998. C6

[28] M.F. Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Comm. Statist. Simulation Comput.*, 19(2):433–450, 1990. C21

[29] M. Hegland, I. Altas, and S. Roberts. Finite element thin plate splines for surface fitting. In B.J. Noye, M.D. Teubner, and A.W. Gill, editors, *Computational Techniques and Applications: CTAC97*, pages 289–296, Singapore, 1998. World Scientific Publishing Co. C19

[30] O. Nielsen. High-dimensional wavelet smoothing. In *Proceedings of the Computational Techniques and Applications Conference CTAC 99*, *ANZIAM J.*, 42, 2000. [Online] http://anziamj.austms.org.au/V42/CTAC99/Niel C35

[31] M.J.D. Powell. The theory of radial basis function approximation in 1990. In W. Light, editor, *Advances in Numerical Analysis, Vol. II (Lancaster, 1990)*, Oxford Sci. Publ., pages 105–210. Oxford Univ. Press, New York, 1992. C13

[32] M.J.D. Powell. A review of algorithms for thin plate spline interpolation in two dimensions. In *Advanced Topics in Multivariate Approximation (Montecatini Terme, 1995)*, pages 303–322. World Sci. Publishing, River Edge, NJ, 1996. C18

[33] M.J.D. Powell. A review of methods for multivariable interpolation at scattered data points. In *The State of the Art in Numerical Analysis (York, 1996)*, pages 283–309. Oxford Univ. Press, New York, 1997. C18

[34] M.J.D. Powell. Recent research at Cambridge on radial basis functions. In *New Developments in Approximation Theory (Dortmund, 1998)*, pages 215–232. Birkhäuser, Basel, 1999. C18

[35] D. Pyle. *Data Preparation for Data Mining*. Morgan Kaufmann Publishers Inc., 1999. C4, C6

[36] G. Strang. *Introduction to Applied Mathematics*. Wellesley Cambridge Press, Wellesley Massachusets 02181, 1986. C11

[37] M. Talagrand. A new look at independence. *Ann. Prob.*, 23:1–37, 1996. C15

[38] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc. Ser. B*, 58(1):267–288, 1996. C34

[39] G. Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, 1990. C17, C21, C21