Adaptive sparse grids

M. Hegland*

(Received 1 June 2001)

Abstract

Sparse grids, as studied by Zenger and Griebel in the last 10 years have been very successful in the solution of partial differential equations, integral equations and classification problems. Adaptive sparse grid functions are elements of a function space lattice. Such lattices allow the generalisation of sparse grid techniques to the fitting of very high-dimensional functions with categorical and continuous variables. We have observed in first tests that these general adaptive sparse grids allow the identification of the ANOVA structure and thus provide comprehensible models. This is very important for data mining applications. Perhaps the main advantage of these models is that they do not include any spurious interaction terms and thus can deal with very high dimensional data.

*Mathematical Sciences Institute, Australian National University, Canberra, AUSTRALIA. mailto:Markus.Hegland@anu.edu.au

[0]See http://anziamj.austms.org.au/V44/CTAC2001/Hegl for this article,
 Published 1 April 2003. ISSN 1446-8735

Contents

1 Introduction

Data is currently collected at an enormous rate. We have seen collections with GBytes, TBytes and even PBytes of data. Increasingly, not only are more data records collected but the complexity of the data is growing as well. We are now analysing image and multimedia data, temporal and spatial data, and text. An important task in modelling this data is the development of *predictive models* or functional relationships between selected features. The identification of such predictive models is hampered by the complexity of the data. This is the *curse of dimensionality* [2] and results in very high dimensional function spaces from which the model is selected. In order to control the complexity we advocate an adaptive strategy where, starting from simple functions (constant, for example), increasingly complex functions are fitted to the data until the best approximation is reached. The main question discussed here is how to choose the function spaces. In particular, we suggest a class of functions which allows good approximation for a reasonable cost.

Adaptive function fitting is nothing new. Maybe one of the most popular tools in machine learning, decision and regression trees [3, 10] deal with data complexity in an adaptive way. They are based on a successive partitioning of the data domain and approximating the predictive model with constants on each subdomain. Piecewise constants, however, do not lead to continuous functions in the case of real variables. This was rectified by the Multivariate Adaptive Regression Splines (MARS) [4], for a stabilised version of this see [1]. Methods for feature and variable selection allow substantial simplifications and attack the curse of dimensionality by dimension reduction. Another approach to adaptive complexity management can be seen in the Analysis of Variance. Here one first identifies main effects of the variables and then looks for successively higher order interactions. This leads to ANOVA decompositions [11] of the form

$$f(x_1, \ldots, x_d) = f_0 + \sum_i f_i(x_i) + \sum_{i,j} f_{i,j}(x_i, x_j) + \cdots ,$$

where the components are successively identified. This approach is also applied in the MARS algorithm and it is suggested in [4] that typically one needs interactions of order two or three, at most of order five. Note that, in addition to the computational difficulties, the interpretation of interactions of many factors gets more problematic. A simple case of the ANOVA decomposition approach are the Additive Models [7] and a smoothing approach for ANOVA decomposition [11].

In recent years, sparse grid techniques [12, 6] have provided a tool to substantially reduce computational complexity of high dimensional problems. They are based on combinations of solutions on different grids. The framework for sparse grid approximations are tensor products of hierarchical component function spaces. The sparse grid approximations are linear combinations of solutions in

tensor products of appropriately chosen subspaces of the component function spaces. An adaptive sparse grid has been described in [5]. In the following we discuss a general framework for adaptive sparse grids for data fitting. We show that lattices of function spaces provide such a framework. In particular, projection formulas for very general spaces will be derived, a greedy algorithm will be suggested and tested for the case of interpolation. The function space lattice framework also allows the analysis of the complexity of the search procedure. We will see that this framework includes the earlier approaches like regression trees and multivariate splines in addition to the sparse grids.

The next section introduces lattices of function spaces. We also review a probabilistic model of data fitting. In Section 3 we see how any function space lattice can be used to construct a new function space lattice by combination of functions from the primary lattice. In Section 4 the adaptive search algorithm for the new lattice is presented and initial comparisons with the traditional sparse grid and with full grid approximation will be discussed. We conclude (§5) by mentioning some open questions.

2 Data fitting in lattices of function spaces

Data fitting consists of finding a function $f : T \to \mathbb{R}$ such that, for two random variables X and Y with domains T and $\mathbb{R}$ respectively the expected squared residual

$$\mathcal{J}(f) := E\left[(Y - f(X))^2\right] \tag{1}$$

is minimised where $E[\cdot]$ denotes the expectation. The solution is well known to be the conditional expectation

$$f(x) = E\left[Y \mid X = x\right], \tag{2}$$

However, as the joint distribution of X and Y is unknown this formula cannot be easily applied. To make things more specific, one determines the function f from a function space V which satisfies

$$\mathcal{J}(f) \leq \mathcal{J}(g), \quad \text{for all } g \in V. \tag{3}$$

We will rely on linear function spaces and can thus represent f by a linear combination of basis elements b_i of V as:

$$f = \sum_{i=1}^{m} \gamma_i b_i. \tag{4}$$

Inserting this into equation (1) one gets

$$\mathcal{J}(f) = E[Y^2] - 2\gamma^T E[Yb] + \gamma^T E[bb^T]\gamma, \tag{5}$$

and for the coefficient vector γ of the minimising f one obtains:

$$\gamma = E[bb^T]^{-1} E[Yb]. \tag{6}$$

As the probability distribution is not known one approximates these terms using observed data. For simplicity, we assume that we have a data set $D_N = \{(X^{(i)}, Y^{(i)})\}$ of independent, identically distributed data. Then, an approximation f_λ is found which minimises the functional

$$\mathcal{J}_\lambda(f) = \frac{1}{N} \sum_{i=1}^{N} \left[Y^{(i)} - f(X^{(i)})\right]^2 + \lambda \|Lf\|^2. \tag{7}$$

The second term is introduced to reduce the variance of the error and the parameter λ controls the tradeoff of variance (large λ gives

simple functions and small variance) and bias (large λ introduces a potentially large approximation error). From the family of functions f_λ the best one uses a smoothing parameter λ:

$$\lambda = \operatorname*{argmin}_\lambda E\left[(Y - f_\lambda(X))^2 \mid D_N\right] . \tag{8}$$

We will now assume that the function f is approximated in a lattice of function spaces V_α where the index α is typically from a lattice $\mathbb{N}^d$ of integer tuples and characterises the complexity (or information) of the function space V_α. Recall that a lattice is a partially ordered set in which any two elements have a least upper bound and a greatest lower bound. The ordering is given in our case by $V_\alpha \subset V_\beta$. The greatest lower bound is the space with index which contains as components the pairwise minima of the respective components of the two spaces and so $V_\alpha \cap V_\beta = V_{\alpha\wedge\beta}$. Examples of function space lattices include:

1. Functions defined on $[0,1]$ which are constant on the subintervals $[i/2^n, (i+1)/2^n]$. This leads to Haar wavelets and more generally, wavelets, which form a chain, the simplest case of a lattice.

2. Continuous functions on $[0,1]$ which are linear on the subintervals $[i/2^n, (i+1)/2^n]$, or more generally, splines.

3. When one allows different sizes of subintervalls (for example, for adaptive local grid refinement) one gets the continuous functions on $[0,1]$ which are linear on each $[i_j/2^n, (i_{j+1})/2^n]$, where typically n is fixed.

4. Furthermore, functions $f : C \to \mathbb{R}$ which are constant on A_i where $\bigcup A_i = C$ is a partition of the set of classes C.

5. Finally, regression trees and multivariate splines form a lattice.

An important function space lattice is obtained when the function space is a tensor product $V = V_1 \times \cdots V_d$ and each component function space V_k is hierarchical:

$$V_{j,0} \subset V_{j,1} \subset \cdots V_{j,m_j} = V_j \,. \tag{9}$$

In this case a function space lattice is defined by

$$V_\alpha = V_{1,\alpha_1} \times \cdots \times V_{d,\alpha_d} \,. \tag{10}$$

The hierarchies of the component function spaces may have a variety of origins, including: approximation (splines), symmetries (Fourier), domain knowledge (biological taxonomies) or the data. Given families of projections $P_{i,\alpha_i} : V_i \to V_{i,\alpha_i}$ one obtains projections $P_\alpha = \bigotimes P_{i,\alpha_i}$. More concisely, one gets directly:

Proposition 1 (Projections in Lattice Spaces) *For every lattice space generated from a tensor product of hierarchical spaces there are linear operators P_α on V with range $R(P_\alpha) = V_\alpha$ and $P_\alpha P_\beta = P_{\alpha\wedge\beta}$. Furthermore $P_\alpha^2 = P_\alpha$ and $P_\alpha P_\beta = P_\beta P_\alpha$.*

For a fixed λ (which can be chosen as before) one now looks for a function f_α such that

$$\mathcal{J}_\lambda(f_\alpha) \leq \mathcal{J}_\lambda(f) \,, \quad \text{for all } f \in V_\alpha \,. \tag{11}$$

The choice of the index α is done adaptively, where one starts with $\alpha = 0$ and obtains a chain $0 \prec \alpha_1 \prec \alpha_2 \ldots$ where the index is determined to get the best f_α such that

$$\alpha_{k+1} = \operatorname*{argmin}_{\beta \succ \alpha_k} E\left[(Y - f_\beta(X))^2 | D_N\right] \,. \tag{12}$$

(As before the expectation will have to be estimated from the data.) One then gets a chain of approximation spaces $V_0 \subset V_{\alpha_1} \subset V_{\alpha_2} \cdots \subset$

V_{α_m} and, starting from the top, one improves the quality through pruning by choosing the best α to be

$$\alpha'_{k-1} = \operatorname*{argmin}_{\beta \prec \alpha'_k} E\left[(Y - f_\beta(X))^2 | D_N\right] . \tag{13}$$

This algorithm may, in many cases, give a reasonable approximation of f. For higher dimensions the approximation may contain many degrees of freedom which are wasted, as they are not used to model anything useful, just noise. In general, however, lattices of function spaces provide a useful tool in the adaptive approximation of functions from very large and high-dimensional function sets.

3 Combination of function spaces

A common difficulty of the tensor-product based lattices is that to obtain a good approximation order high dimensional spaces are required. Sparse grid approximations [12] combine several spaces from the lattice to get better approximations at lower costs. In the combination technique [6] the appropriate multiples of the lower order approximations are added up to form the higher order approximations. In this section we see how these ideas can be implemented adaptively, in particular, how the multiples in the combination technique are updated when new spaces are included in the combination.

In ordered sets a *downset* I is a subset which contains with any elements all the smaller elements, that is,

$$\alpha \in I \quad \text{and} \quad \beta \leq \alpha \Rightarrow \beta \leq I . \tag{14}$$

The set of all downsets $\mathcal{O}(\mathcal{A})$ of a lattice $\mathcal{A}$ forms a lattice of subsets and for each $I \in \mathcal{A}$ we denote by $\downarrow I$ the smallest downset which

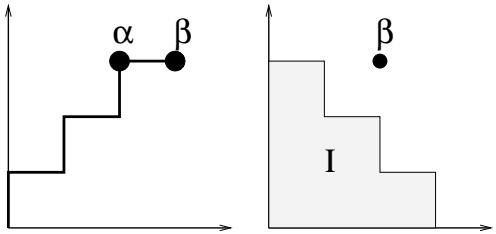

FIGURE 1: Cover for the case of $\mathbb{N}^2$ and the combination lattice.

contains I. The the *combination space* lattice is

$$V_I = \sum_{\alpha \in I} V_\alpha \,. \tag{15}$$

The combination space is a subspace of the lowest upper bound, $V_I \subset \bigwedge_{\alpha \in I} V_\alpha$. While equality holds for chains, the combination space can be substantially smaller for more complex lattices. In [5] these sets I are called *active indices*.

The fitting in lattice spaces algorithm of the previous section is now applied to the combination space lattice. It is instructive to compare the cover relation $\alpha \prec \beta$ in the original lattice and the combination lattice. Recall that β covers α or $\alpha \prec \beta$ if $\alpha \leq \beta$ and there is no element γ such that $\alpha \leq \gamma \leq \beta$. In the case of the lattice of indices $\mathbb{N}^d$ one has $\alpha \prec \beta$ when $|\alpha - \beta| = 1$, whereas in the the second lattice one has $I \prec J$ if $J = I \cup \{\alpha\}$ such that for all $\beta \prec \alpha$ one has $\alpha \in I$, see Figure 1.

The projections of the original lattice space give rise to projections in the combined space and one has:

Proposition 2 *[Projection theorem] If the lattice V_α has projections P_α as in Proposition 1 then there are linear operators P_I on V with range $R(P_I) = V_I$ such that $P_I P_J = P_{I \cap J}$. Conversely, if P_I is*

a family of projections with these properties, then $P_\alpha = P_{\downarrow\alpha}$ *defines a family of projections as in Proposition 1.*

Proof: We define the linear operators

$$P_I = 1 - \prod_{\alpha\in I}(1 - P_\alpha) \tag{16}$$

As $P_\alpha P_\beta = P_\beta$ if $\beta \leq \alpha$ one gets $P_I = 1 - \prod_{\alpha\in \operatorname{Max} I}(1 - P_\alpha)$. Similarly, it follows that

$$P_I = \sum_{\alpha\in I} c_\alpha P_\alpha \tag{17}$$

and the *combination coefficients* c_α are zero if α is not in the sublattice generated by $\operatorname{Max} I$, the maximal elements of I. Thus the range of P_I is V_I.

Finally, introduce $Q = P_I P_J - P_{I\cap J}$. See that the range of this operator is $V_{I\cap J}$. Furthermore, $V_{I\cap J}$ is in the nullspace of Q. Observe that P_I maps elements of V_I onto themselves, that is, P_I is a projection and it follows that $Q^2 = Q$ and thus $Q = 0$. The converse follows directly. ♠

A direct consequence of the projection theorem provides formulas to compute the projections of a covering element. The proof is obtained by using the basic properties of P_I.

Corollary 3 (Updating Formulas) *Let* $J = I \cup \{\beta\}$ *be a covering element of* I *and the family of projections* P_I *as in Proposition 2*

and $P_\alpha = P_{\downarrow\alpha}$. *Then one has*

$$P_J - P_I = (1 - (1 - P_\beta)) \prod_{\alpha \in I} (1 - P_\alpha) = P_I P_\beta \tag{18}$$

$$= P_\beta \prod_{\alpha \in \operatorname{Max} I} (1 - P_\alpha) \tag{19}$$

$$= \sum_{\langle \operatorname{Max} I, \beta \rangle} d_\alpha P_\alpha \,. \tag{20}$$

The combination coefficients d_α *of the update satisfy* $d_\beta = 1$ *and* $d_{\gamma \wedge \alpha} = -d_\gamma$ *for all* γ *and* $\alpha \in \operatorname{Max} I$.

With these tools we now implement the standard lattice space algorithm for fitting.

4 The adaptive algorithm and first results

Using the updating formulas from Section 3 we now implement the lattice space adaptive fitting algorithm for the combination lattice, see Algorithm 1.

The grids obtained with this method range from very sparse grids for additive functions $f(x) = f_1(x_1) + \cdots + f_d(x_d)$, somewhat less sparse grids for very smooth functions and relatively full grids for functions with near singularities, see Figure 2.

The adaptive techniques deal with the curse of dimensionality by approximating in simpler spaces. However, the search space for the method can be very large. We have in particular:

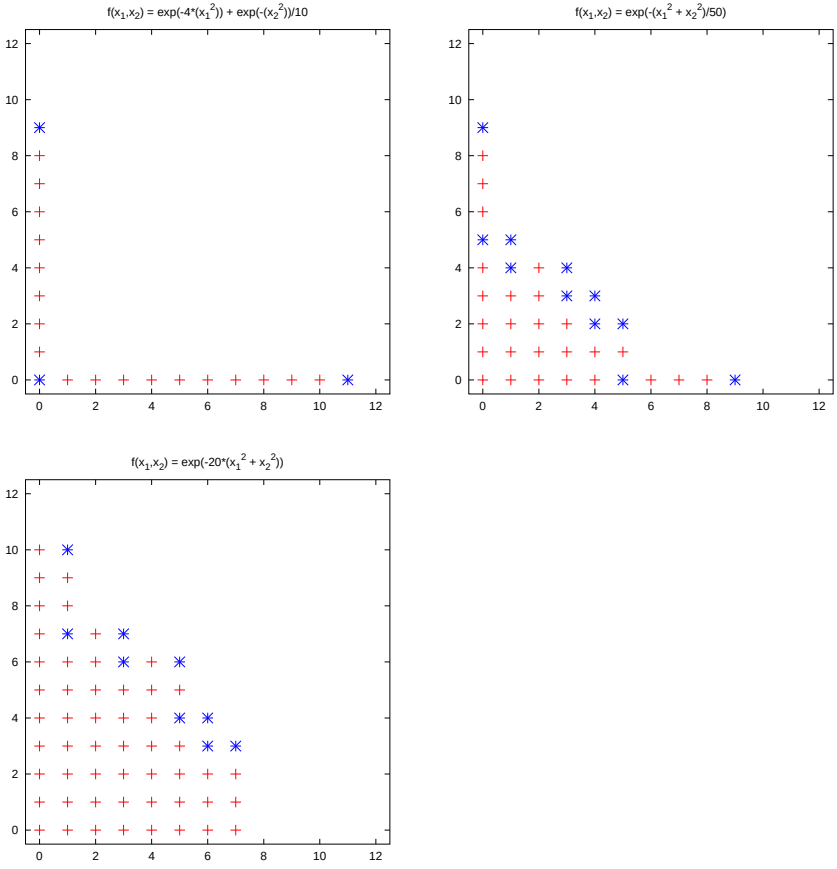

FIGURE 2: Three examples index sets I for the combination method

Algorithm 1: The fitting algorithm for the combination spaces

$I = 0$
$f_0 = P_0 f$
while $\mathcal{J}(f_I) > \epsilon$ **do**
  $J = \operatorname{argmin}_{J' \succ I} \mathcal{J}(P_{J'} f)$
  if $\mathcal{J}(P_J f) < \mathcal{J}(f_I)$ **then**
    $I = J$, possibly pruned
  else
    choose simple (random) I close to old ones
  $f_I = P_I f$

Proposition 4 *For a given index set I the search space of the adaptive algorithm is of the size:*

$$\sigma(I) = |\operatorname{Min} I^C|\,, \tag{21}$$

where $|\operatorname{Min} I^C|$ denotes the size of the set of minimal elements of the complement of (the downset) I.

Proof: The search needs to be done over all the elements $\beta \notin I$ which cover elements in I. These are just the minimal elements of I^C. ♠

Some special cases illustrate the variety of sizes possible:

- In many cases, only a finite number of features contribute to the predictor. This is modelled by $J \subset \mathbb{N}^k$ and $I = \{(j, 0, \ldots, 0) \mid j \in J\}$. In this case the algorithm is scalable in the dimension: $\sigma(I) = \sigma(J) + d - k$. This means that the search will have to consider either including one of the other $d - k$ variables or improving the model with the current

variables. Typically this bound will be combined with one of the bounds below.

- In the case of additive models one has $I = \{m_1 e_1, \ldots, m_d e_d\}$ for some $m_i \in \mathbb{N}$. In the case where all $m_i \neq 0$ one gets $\sigma(I) = d + \binom{d}{2} = \binom{d+1}{2}$ as either an interaction between two variables is included or alternatively, the additive model is improved.
- In the case of full grids the options are to refine any of the variables and so the search space is of size $\sigma(I) = d$.
- Sparse grids may occur for smooth functions which depend on all the variables. Here one has $\sigma(I) = \binom{m+1}{d-1} = \mathcal{O}(m^{d-1}/(d-1)!)$ where $m = |\alpha|$ for $\alpha \in \operatorname{Max} I$.

While sparse and even full grids can be obtained from the adaptive algorithm, one can also recast the sparse and full grid algorithms in the combination framework, see Algorithms 2 and 3. Note that for a sparse grid derived from a $m \times m \times \cdots \times m$ grid one requires $\mathcal{O}(m \times \log_2(m)^{m-1})$ storage space and time and for an $m \times m \times \cdots \times m$ grid the full grid approximation requires $\mathcal{O}(m^d)$ storage space and time.

Algorithm 2: Sparse Grid Algorithm

$I = 0$
$f_0 = P_0 f$
while $\mathcal{J}(f_I) > \epsilon$ **do**
  $J = \operatorname{argmin}_{J' \succ I} |J|$
  $I = J$
  $f_I = P_I f$

Finally, we have implemented the algorithm in order to provide a proof of concept. We have used the algorithm to compute

Algorithm 3: Full Grid Algorithm – Combination Version

$I = 0$
$f_0 = P_0 f$
while $\mathcal{J}(f_I) > \epsilon$ **do**
  $J = \text{argmin}_{J' \succ I} |J|_\infty$
  $I = J$
  $f_I = P_I f$

an adaptive approximation for a given function by interpolation which provides a projection of the kind required. In particular we display in Figure 3 the results for the function $f(x_1, x_2, x_3, x_4) = \exp[-x_1^2] + \exp[-(x_1^2 + x_2^2)]$ which is four-dimensional but only three dimensions contribute and at most two dimensional components occur. The error term was estimated using a random test data set. See the "plateaus" for the full and sparse grids where a further increase in complexity (likely related to the variable x_4 and interactions between x_1 and the other variables) did not improve the approximation. This is actually a type of overfitting.

5 Conclusion

We have seen that function space lattices are a basic tool for data fitting in high dimensions. Starting with any function space lattice one can obtain a flexible approximation space through combination. We have found in further experiments that the algorithm presented here is good in recovering the ANOVA structure of the underlying function and we have been able to recover functions for cases with up to around 40 variables. Many open questions remain, however. Some of these questions revolve around the ANOVA

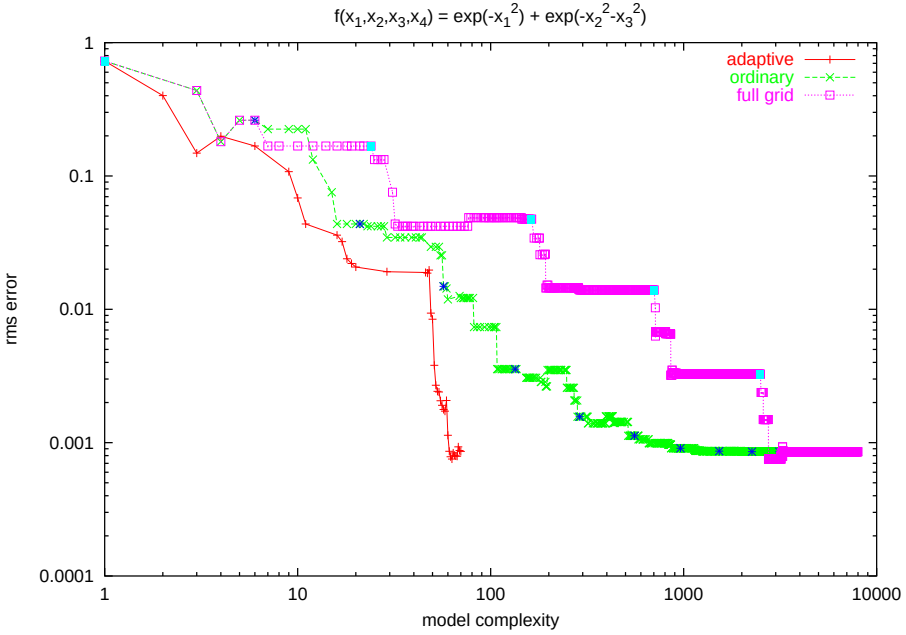

FIGURE 3: Errors as a function of model complexity for the interpolation of $f(x_1, x_2, x_3, x_4) = \exp[-x_1^2] + \exp[-(x_1^2 + x_2^2)]$.

structure of functions which occur in practice. Can functions be reduced to ones with relatively small numbers of variables? Are the interactions limited? And how do these properties relate to the smoothness of functions? We believe that these questions are closely related to the concentration of measure, see [8]. The combination technique is exact if one has available the projections P_α. What if one only has approximations of such projections? The sparse grid technique has been applied for the finite element solution of partial differential equations, and in some cases one can show that the combination method actually corresponds to extrapolation [6]. It would be of interest to understand when the adaptive method can really recover the exact ANOVA structure, and under which conditions an ANOVA structure is stable, that is, not further modified in later iterations. Finally, we are now working on the implementation of the technique using the penalised least squares fitting algorithm TPSFEM [9] and plan to have a first prototype available soon, see http://datamining.anu.edu.au for further information.

Acknowledgements: Earlier work has been supported by the ACSys CRC and current work is partially supported by the Australian Partnership for Advanced Computing.

References

[1] S. Bakin, M. Hegland, and M. R. Osborne. Parallel MARS algorithm based on B-splines. *Computational Statistics*, 15:463–484, 2000. C337

[2] R. Bellman. *Adaptive control processes: A guided tour*. Princeton University Press, Princeton, N.J., 1961. C336

[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and regression trees*. Wadsworth Advanced Books and Software, Belmont, CA, 1984. C337

[4] J.H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19:1–141, 1991. C337

[5] M. Griebel. Adaptive sparse grid multilevel methods for elliptic PDEs based on finite differences. *Computing*, 61(2):151–179, 1998. C338, C343

[6] M. Griebel, M. Schneider, and C. Zenger. A combination technique for the solution of sparse grid problems. In *Iterative methods in linear algebra (Brussels, 1991)*, pages 263–281. North-Holland, Amsterdam, 1992. C337, C342, C351

[7] T. J. Hastie and R. J. Tibshirani. *Generalized additive models*. Chapman and Hall Ltd., London, 1990. C337

[8] M. Hegland and V. Pestov. Additive models in high dimensions. Research Report 99-33, Victoria University of Wellington, School of Mathematical and Computing Sciences, 1999. http://xxx.lanl.gov/abs/cs.DS/9912020. C351

[9] M. Hegland I. Altas and S. Roberts. Finite element thin plate splines for surface fitting. In *Computational Techniques and Applications: CTAC97*, pages 289–296, 1998. C351

[10] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993. C337

[11] G. Wahba. *Spline models for observational data*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1990. C337

[12] C. Zenger. Sparse grids. In *Parallel algorithms for partial differential equations (Kiel, 1990)*, pages 241–251. Vieweg, Braunschweig, 1991. C337, C342