

## Anytime Exploratory Data Analysis for Massive Data Sets

Padhraic Smyth\*

Department of Information and Computer Science  
University of California, Irvine  
CA 92697-3425  
smyth@ics.uci.edu

David Wolpert

IBM Almaden Research,  
650 Harry Road, San Jose, CA 95120  
dhw@almaden.ibm.com

### Abstract

Exploratory data analysis is inherently an iterative, interactive endeavor. In the context of massive data sets, however, many current data analysis algorithms will not scale appropriately to permit interaction on a human time-scale. In this paper “anytime data analysis” is proposed as a general framework to enable exploratory data analysis of massive data sets. Anytime data analysis takes into account not only the quality of the model being fit but also the resources (time and memory) used to achieve that fit. The framework is discussed in some detail for interactive multivariate density estimation. Out-of-sample log-likelihood and model combination techniques (such as stacking) are used to greedily explore the data landscape. The method is applied to two significant scientific data sets where it is shown that it can be better to combine multiple “cheap-to-construct” models than to spend the same time optimizing the parameters of a single more complex model.

### Introduction and Motivation

The problem of massive data sets is now well-known. There is tremendous interest at present in techniques such as data mining and knowledge discovery to extract information from such massive data sets. Most of this interest is applications-driven: in medicine, finance, marketing, science, and government, users are clamoring for techniques which allow them to handle their vast volumes of data. These applications generate data sets which are increasingly beyond the boundaries of what existing data analysis algorithms can handle (cf. *Proceedings of the National Research Council Workshop on Massive Data Sets*, 1996). For example, major telecommunications companies can collect hundreds of millions of call-records per day.

In this paper *anytime data analysis* is proposed as a paradigm-shift away from current batch-oriented, resource-ignorant algorithms towards exploratory, incremental, resource-conscious algorithms. Anytime

data analysis addresses the problem of dealing with massive data sets by posing the problem as one of *resource-limited* data exploration. For example, many branches of science (including astrophysics, atmospheric and ocean sciences, geology, ecology, biology, and medicine) are currently being inundated with observational data of vast proportions. Scientists are not interested in the data *per se*. The data are a “medium” for the development and validation of scientific hypotheses at an abstract level.<sup>1</sup>

Note that “massive data set” is not a well-defined term. In the context of this paper it is interpreted in a broad sense to mean any data set which is beyond a particular user’s data analysis capabilities due to data set size. For example, one could have too many data points to fit in available main memory, the computational complexity of the available algorithms could be too large to complete the analysis in reasonable time, or the dimensionality of the problem (the number of variables) could be so large as to overwhelm any available model-fitting technique. Thus, the term “massive data set” is somewhat application and context-dependent.

The paper is organized as follows: first we provide a brief discussion of anytime algorithms in general and related work in exploratory data analysis. We then establish some necessary standard notation and background on kernel and mixture modeling. (In this paper we focus on *unsupervised learning*: much of the presentation is also directly applicable to supervised learning problems). The next section introduces a formal framework for anytime data analysis and discusses some key points which arise from this formalism. We then describe a new method, stacked density estimation, for combining models in an *unsupervised* framework: such model combining is essential in anytime data analysis. Finally, we illustrate the anytime data

\*Also with the Jet Propulsion Laboratory 525-3660, California Institute of Technology, Pasadena, CA 91109.

<sup>1</sup>Copyright ©1997, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

analysis framework by proposing some simple static anytime strategies and investigating their performance profiles on two large data sets.

## Related Work

Exact definitions of anytime algorithms vary with different authors, but in general an *anytime algorithm* is an algorithm which produces a set of outputs continuously over time (Zilberstein and Russell, 1996). Thus, in contrast to conventional algorithms, an anytime algorithm can be interrupted at “any time” and still produce an output of a certain quality. The foundational work in this area relies on a *decision-theoretic* framing of the problem, and in particular, the principle of maximum expected utility (Dean and Wellman (1991), Horvitz (1987), Russell and Wefald (1991)). Anytime algorithms have largely been developed in the context of artificial-intelligence approaches to planning and reasoning. We are not aware of any direct application of anytime algorithms to unsupervised data analysis to date.

There is a large body of work on *online learning* (where the data points are presumed to arrive sequentially in real-time) in the adaptive control, pattern recognition, and signal processing literatures (for recent work see for example, Moore and Schneider (1996)). For example, many neural network algorithms are designed to operate in an online fashion, and many statistical estimation algorithms can readily be formulated for online adaptation. However, this work is mainly concerned with *parameter estimation* rather than exploring a family of models and as such do not provide a general basis for anytime data analysis where model structure is unknown a priori. Batch data analysis algorithms which are *iterative* in nature have a natural anytime interpretation (i.e., one can always use the estimates from the most recent iteration). Indeed we will take advantage of this later in the paper. All of these methods can be viewed as restricted special cases of a general anytime data analysis framework.

## Notation and Background

Let  $\underline{x}$  be a particular realization of the  $d$ -dimensional multivariate variable  $\underline{X}$ . We will discuss data sets  $D = \{\underline{x}_1, \dots, \underline{x}_N\}$ , where each sample  $\underline{x}_i, 1 \leq i \leq N$  is an independently drawn sample from an underlying density function  $f(\underline{x})$ . A commonly used model for probabilistic clustering and density estimation is the *finite mixture model* with  $k$  components, defined as:

$$f^k(\underline{x}) = \sum_{j=1}^k \alpha_j g_j(\underline{x}), \quad (1)$$

which consists of a linear combination of  $k$  component distributions with  $\sum \alpha = 1$ . The component  $g_j$ 's are usually relatively simple unimodal distributions such as Gaussian distributions. The model can be generalized to include background noise models, outlier models, and so forth. Density estimation with mixtures involves finding the locations, shapes, and weights of the component densities from data (using for example the Expectation-Maximization (EM) procedure). *Kernel density estimation* can be viewed as a special case of mixture modeling where a component is centred at each data point, given a weight of  $1/N$ , and a common covariance structure (kernel shape) is estimated from the data.

In terms of time complexity, fitting a  $k$  component Gaussian mixture model with arbitrary covariance matrices in  $d$  dimensions scales as  $O(kNEd^2)$  where  $E$  is the number of iterations of the EM algorithm. The scaling of  $E$  as a function of  $d$  and  $N$  is not well understood in general: for some clustering applications it is sufficient to fix  $E$  as a constant (say 50 iterations), however this by no means guarantees the resulting fit will be even close to a local maximum of the likelihood function.

The quality of a particular probabilistic model can be evaluated by an appropriate scoring rule on independent out-of-sample data such as the test log-likelihood or log-scoring rule. Given a test data set  $D^{test}$ , the test likelihood is defined as

$$\log f(D^{test} | f^k(\underline{x})) = \sum_{D^{test}} \log f^k(\underline{x}_i) \quad (2)$$

An important point is that this log-scoring objective function is an effective and consistent method for comparing cluster models and density estimates in an unsupervised learning context, i.e., it plays the role of classification error in classification or squared error in regression.

## An Anytime Data Analysis Framework

We are given a data set  $D$ .  $D_j \subseteq D$  denotes a subset of the elements of  $D$  (e.g., a random sample, a bootstrap sample, etc). We also have a finite set of “unfitted” models available,  $\mathbf{M}^u = \{M_1^u, \dots, M_K^u\}$ , which are indexed by  $k$ , i.e.,  $M_k^u$  could be a mixture of a particular type with a particular number of components, or a type of kernel density model, and so forth. To fit the models we have a set of algorithms available,  $\mathbf{A} = \{A_1, \dots, A_L\}$ , e.g., the EM procedure for mixtures and various bandwidth-setting algorithms for kernels. An algorithm  $A_l$  can be viewed as a mapping from an instantiation of the tuple  $\langle D_j, k_i, q, \pi_t \rangle$  to a fitted model  $M_l$ . The parameter  $q$  is a scalar or

vector influencing the “quality” of the estimation algorithm, such as the number of steps or number of random restarts in an EM procedure, or the resolution of the bandwidth search in a kernel estimator.  $\pi_t$  represents in a general sense prior knowledge available at time  $t$  when the algorithm is run, including results from previous algorithm runs, user-defined constraints, and so forth.

For computational simplicity, we will restrict our attention to the maximum a posteriori (MAP) estimation framework where posterior modes are reported rather than full posterior densities. The generalization to a full Bayesian analysis in the anytime context is clearly of interest, but may be computationally impractical, and is not addressed here.

A *strategy*  $S$  is a particular sequence of algorithms and algorithm input parameters,

$$S = \{A_1(D_1, q_1, k_1, \pi_1), \dots, \\ A_j(D_j, q_j, k_j, \pi_j) \dots, \\ A_J(D_J, q_J, k_J, \pi_J)\}.$$

The *performance profile* of a strategy is a plot of utility versus computation time for a particular sequence of algorithm runs. Finding the optimal solution to the anytime data analysis problem in this context can be stated as follows:

Given  $D$ ,  $M^u$ ,  $A$ , and  $\pi_1$ , find a *strategy*  $S$  such that computational and memory constraints are obeyed and  $S$  is strictly dominant over all other strategies in terms of the performance profile.

There are some general comments worth making at this point:

- It seems very likely that any non-trivial instantiation of finding the best strategy will be NP-hard. Even in very restricted versions of this problem, such as choosing between the better of kernel estimators with different kernel shapes, there is no way to determine the optimal strategy without knowing the true  $f(\underline{x})$ . Thus, in practice, our attention will be limited to finding good heuristic (typically myopic) strategies as has been the practice with anytime algorithms in other problem domains.
- One can distinguish between *static* anytime strategies which are determined *a priori*, and *adaptive* anytime strategies where the algorithm and parameters at time  $t + 1$  are chosen adaptively based on results up to time  $t$ .
- Static strategies are clearly more limited than adaptive strategies, yet, already they form an interesting

generalization of typical data analysis methods. In this context, “standard” algorithms, such as iterative improvement learning procedures like backpropagation or EM, can be viewed as naive “single-step” strategies which ignore resource constraints.

- Adaptive strategies provide a powerful framework to adaptively explore data but are faced with great uncertainty in terms of trying to choose the next best step at any stage without knowing the true  $f(\underline{x})$ . Markov-chain Monte Carlo methods, which are currently popular in applied Bayesian statistics, can be viewed as adaptive strategies which use estimates of posterior probability densities to guide search in model space: however, resource constraints are typically ignored and there is no formal connection to an anytime algorithmic framework.

### Unsupervised Stacking for Model Combination

Model combination techniques have been widely used in regression and classification (e.g., Chan and Stolfo, 1996; Breiman 1996a, 1996b, Wolpert 1992), but relatively rarely in unsupervised learning. However, using estimated out-of-sample log-likelihood as the performance metric, it is straightforward to combine different density models.

Here we consider *stacked density estimation* which we briefly review (for details see Smyth and Wolpert, 1997). Consider that we have  $K$  individual estimates for the density of  $\underline{x}$ , namely,  $f^1(\underline{x}), \dots, f^K(\underline{x})$ . These could be kernel estimators with different bandwidths or shapes, mixture models with different numbers of components, etc. We would like to combine these  $f$ 's into a single predictive model. The straightforward approach is to treat the combined model as a mixture with  $K$  components,

$$f^{1, \dots, K}(\underline{x}) = \sum_{k=1}^K \beta_k f^k(\underline{x}) \quad (3)$$

where the  $\beta_k$  are unknown parameters which can be assumed to be non-negative and sum to 1 in the standard fashion.

In this context, stacked density estimation works as follows. First one runs  $v$ -fold cross-validation on the training data, and for each fold, train each of the models  $f^k(\underline{x})$  on the training partition and predict each of the test density points (i.e., calculate the  $f^k(\underline{x}_i)$  where  $\underline{x}_i \in \{\text{test partition}\}$ ). After cross-validation is complete, one has a table of  $N \times K$  density values, an out of sample prediction for each of the  $N$  data points for each of the  $K$  models. The EM procedure can be used directly on this table to estimate the  $\beta$  weights

Table 1: Performance of stacking multiple mixture models (using different weighting schemes) on the asteroid data set (described in more detail in the next section). 1500 data points were used for model-fitting and stacking. 2000 data points were used to evaluate the out-of-sample log-likelihood.

Weighting Method	Out-of-Sample Log-Likelihood
Best Single Model	6333.2
Uniform Weighting	6449.6
Stacking	6476.6

in the stacked mixture above. Finally the  $K$  models are retrained on all of the training data and weighted according to the  $\beta$ 's for prediction. An alternative approach in the context of massive data sets is to simply use a separate validation set to estimate the  $\beta$  weights: if data are "cheap," but computation time is "expensive," this strategy may be more effective than cross-validation.

The component density models  $f^k$  can themselves be mixture models with different numbers of components. This leads to an interesting representation, a hierarchical "mixture of mixtures" models, providing a natural multi-scale representation of the data. Each of the components models a different scale in the data, i.e., models with fewer mixture components will be broad in scale, while models with more components can reflect more detail. Stacking determines in a data-adaptive manner which scale is emphasized. More generally, one can mix and match many different types of density function models: kernel density estimators with different shapes and bandwidths, mixture models with non-Gaussian components, and Gaussian mixtures with different numbers of components and different covariance structures. For example, a useful combination (which we use later) is to combine finite support kernels (such as triangular kernels) with Gaussian mixtures: the Gaussian mixtures tend to regularize the tendency of finite support kernels to "blow up" (infinitely negative log-likelihood) when data points are outside the support of the model.

Results of a simple experiment with combining Gaussian mixtures are shown in Table 1. The table shows the out-of-sample test log-likelihood, where the test data was not used in any way during training, for three methods of combining mixture models with  $k = 1, 2, \dots, 9, 10$  components, using the asteroid data. The first method "cheats" and looks at the test data to pick the single best model. The second method uniformly weights the 10 models. The third uses cross-

validated stacking as described originally in Wolpert (1992) but adapted for density estimation: the stacking weights can be quickly estimated by EM (Smyth and Wolpert, 1997). Both combining methods outperform the single best model chosen by "cheating" (agreeing with what Breiman (1996a) found for supervised learning), and stacking outperforms uniform weighting. The log-likelihood scale is not particularly intuitive, but as argued by Madigan and Raftery (1994), a difference of 5 or so on this scale effectively means that the posterior probability of the more likely model given the data is an order of magnitude greater.

In experiments on several other data sets we have found that stacked density estimation consistently produces better density estimates than choosing a single model using cross-validation, a single model chosen by looking at the test data, or uniform weighting (Smyth and Wolpert, 1997). For anytime algorithms such combining methods are critically important since they allow the results of earlier model fitting to be incorporated into current estimates (rather than being discarded).

## Anytime Density Estimation

In this section we explore a particular set of static anytime strategies and evaluate their performance. These examples are intended to be *illustrative*.

A simple utility function is mutual information  $I(f(\underline{x}); \hat{f}(\underline{x}))$ , which measures the information between the encoder (the data generating process,  $f(\underline{x})$ ) and the receiver (the data analyst). It is straightforward to show that the mean out-of-sample log-likelihood  $1/N \sum_{i=1}^N \log \hat{f}(\underline{x}_i)$  is an unbiased estimator of the increase in mutual information (or reduction in uncertainty) about  $f(\underline{x})$  from knowing  $\hat{f}$ .

Given the limited space constraints, we briefly discuss four different types of strategies and illustrate their use on two scientific data sets. The anytime strategies (in increasing complexity) are:

**Single Mixture:** Fit a single Gaussian mixture model with  $k$  components and report the parameters in an anytime fashion at each iteration of EM. Select the highest-likelihood initialization from 5 different initializations of the EM algorithm using  $k$ -means.

**Single Best:** Fit a sequence of models in this order: triangular product kernels with bandwidths of  $\{0.1, 0.2, 0.5, 0.7, 1.0, 1.5, 2.0, 3.0\}$  times the standard deviation in each dimension, and Gaussian mixtures with  $k \in \{2, 4, 8, 16\}$  components each with arbitrary covariance matrices. In fitting the Gaussian mixtures, only run the EM algorithm for 10 iterations, and use the "best of 5"  $k$ -means initialization

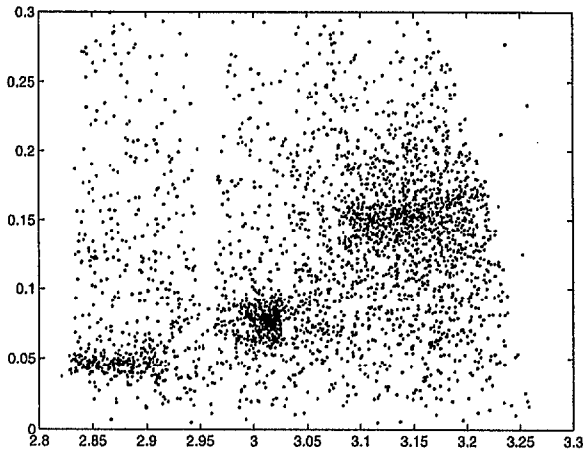


Figure 1: Portion of asteroid data set: see text for details

as in 1. Reserve some of the training data as a validation data set and as each new model is constructed choose the single model which performs best on the validation data set.

**Uniform Weights:** The same sequence as for “Single Best,” but combine the models at each step using uniform weights.

**Stacked Weights:** The same sequence as for “Single Best,” but combine the models at each step using stacking weights which are estimated on the validation subset of the training data.

Figure 1 shows a well-known asteroid data set, where each data point corresponds to a known asteroid in the Solar System. The  $x$ -axis corresponds to distance from the Sun in astronomical units (this particular region is between Mars and Jupiter) and the  $y$ -axis is the eccentricity of the orbit of the asteroid. The existence in such data of gaps and “families of asteroids” (i.e., clusters) are very much the subject of current debate in the planetary geology community (Zappala et al, 1995; Lagervist and Barucci, 1992). There are over 30,000 such data points and only a fraction of the data set is displayed here. This data set is typical in certain respects of observational science data: a large, continually growing, collection of data which is continually being analyzed by a variety of scientists interested in refining their understanding of the underlying structure.

The plots show out-of-sample log-likelihood per sample on the test data set, i.e., an estimate of “information gain” about the unknown density in each case.

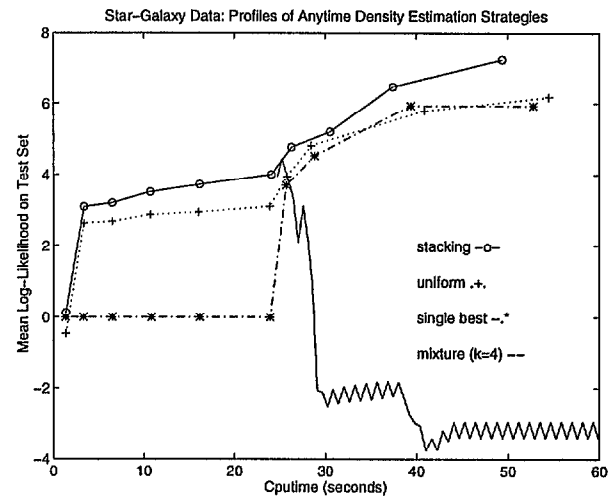
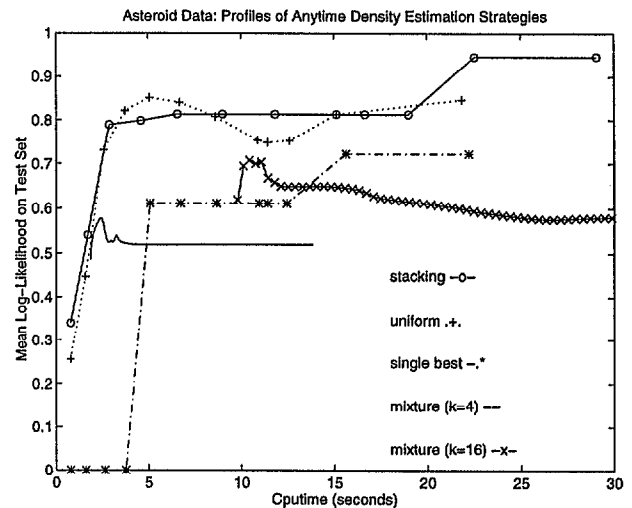


Figure 2: Profile information plot for (a) the asteroid data and (b) the star-galaxy data for various simple static strategies for anytime density estimation combining Gaussian mixture and triangular kernel density estimates. See text for details.

This information gain is computed in each case relative to a default prior consisting of a single Gaussian bump fitted to the data.

Figure 2(a) shows the estimated anytime profiles on the asteroid data with 1000 training data points, of which 500 are reserved for validation, and 1500 test data points. Stacking and uniform weighting dominate, and the single model strategies are universally worse. Note that the model combining strategies dominate the single-model strategies at all times.

Figure 2(b) shows the estimated profiles on an 8 dimensional data set describing stars and galaxies, with 1500 training data points, of which 600 are used for validation, and 1000 test data points. This is the SKI-CAT data with the class labels removed (Weir et al., 1995). Here, astronomers are interested in the existence of subclusters among the galaxies. The data used here is only a small fraction of what is a data collection with millions of records. Here stacking dominates all other strategies. Uniform weighting is competitive with stacking and the single best strategy eventually finds a model which is competitive with the weighting methods. On this data set, the  $k = 4$  mixture model (and  $k = 16$ , not shown, because it is so off-scale) overfit to the extent that the information gain is negative relative to a single Gaussian.

Generally speaking the plots clearly demonstrate that searching among multiple models is a much more effective strategy than fitting a Gaussian mixture model with some fixed number of components to these data sets. Furthermore, among the methods which consider multiple models, the stacking strategy dominated on these data sets and for these models.

These experiments only scratch the surface of the potential possibilities for anytime-type algorithms in exploratory data analysis. For example, for massive data sets one may want to consider building models on small subsets of the data and combining them together (e.g., Breiman, 1997). In addition, there are clear benefits to be gained from having *adaptive* strategies versus *static* ones, so that the exploration algorithm can adapt to the data "landscape" and eliminate low-utility models which take time to fit but produce negligible information gain.

## Conclusions

In this paper we proposed the concept of anytime data analysis as a flexible and practical framework for exploratory analysis of massive data sets. In particular, we used novel combining schemes based on stacking which support anytime density estimation and illustrated their utility on two real-world data sets. Anytime data analysis appears well-suited as a framework

for tackling the problems which are inherent to massive data sets.

## Acknowledgments

Part of the research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

## References

- Breiman, L., 'Stacked regressions,' *Machine Learning*, 24, 49-64, 1996a.
- Breiman, L., 'Bagging predictors,' *Machine Learning*, 26(2), 123-140, 1996b.
- Breiman, L., 'Pasting bites together for prediction in large data sets and on-line,' preprint, 1997.
- Chan, P. K., and Stolfo, S. J., 'Sharing learned models among remote database partitions by local meta-learning,' in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Menlo Park, CA: AAAI Press, 2-7.
- Dean, T. L., and Wellman, M. P., *Planning and Control*, San Mateo, California: Morgan Kaufmann, 1991.
- Hansen, E., and Zilberstein, S., 'Monitoring the progress of anytime problem-solving,' *Proceedings of the 13th National Conference on Artificial Intelligence*, Menlo Park, CA: AAAI Press, 1996.
- Horvitz, E., J., 'Reasoning about beliefs and actions under computational resource constraints,' in *Proceedings of the 1989 Workshop on Uncertainty and AI*, 1987.
- Lagervist, C. I., and Barucci, A., 'Asteroids: distributions, morphologies, origins, and evolution,' *Surveys in Geophysics*, 13, 165-208, 1992.
- Madigan, D., and Raftery, A. E., 'Model selection and accounting for model uncertainty in graphical models using Occam's window,' *J. Am. Stat. Assoc.*, 1994.
- Moore, A. W., and Schneider, J., 'Memory-based stochastic optimization,' in *Advances in Neural Information Processing 8*, D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo (eds.), Cambridge, MA: MIT Press, 1066-1072, 1996.
- Proceedings of the National Research Council Workshop on Massive Data Sets*, Washington, DC: National Academy Press, 1996.

Russell, S. J., and Wefald, E. W., *Do the Right Thing: Studies in Limited Rationality*, Cambridge, MA: MIT Press, 1991.

Smyth, P., 'Clustering using Monte-Carlo cross-validation,' in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Menlo Park, CA: AAAI Press, pp.126-133, 1996.

Smyth, P., and Wolpert, D., 'Stacked density estimation,' preprint, 1997.

Weir, N., Fayyad, U. M., Djorgovski, S. G., 'Automated star/galaxy classification for digitized POSS-II,' *The Astronomical Journal*, 1995.

Wolpert, D. 1992. 'Stacked generalization,' *Neural Networks*, 5, 241-259.

Zappala, V., et al., 'Asteroid families: search of a 12,487 sample using two different clustering techniques,' *Icarus*, 116, 291-314, 1995.

Zilberstein, S., and Russell, S. J., 'Optimal composition of real-time systems,' *Artificial Intelligence*, 82(1-2), 181-213, April 1996.