This article was downloaded by: [144.214.89.16] On: 18 September 2017, At: 07:51
Publisher: Institute for Operations Research and the Management Sciences (INFORMS)
INFORMS is located in Maryland, USA

INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Toward Automated Intelligent Manufacturing Systems (AIMS)

Hoi-Ming Chi, Okan K. Ersoy, Herbert Moskowitz, Kemal Altinkemer,

To cite this article:
Hoi-Ming Chi, Okan K. Ersoy, Herbert Moskowitz, Kemal Altinkemer, (2007) Toward Automated Intelligent Manufacturing Systems (AIMS). INFORMS Journal on Computing 19(2):302-312. https://doi.org/10.1287/ijoc.1050.0171

Full terms and conditions of use: http://pubsonline.informs.org/page/terms-and-conditions

This article may be used only for the purposes of research, teaching, and/or private study. Commercial use or systematic downloading (by robots or other automatic processes) is prohibited without explicit Publisher approval, unless otherwise noted. For more information, contact permissions@informs.org.

The Publisher does not warrant or guarantee the article's accuracy, completeness, merchantability, fitness for a particular purpose, or non-infringement. Descriptions of, or references to, products or publications, or inclusion of an advertisement in this article, neither constitutes nor implies a guarantee, endorsement, or support of claims made of that product, publication, or service.

Copyright © 2007, INFORMS

Please scroll down for article—it is on subsequent pages

INFORMS is the largest professional society in the world for professionals in the fields of operations research, management science, and analytics.
For more information on INFORMS, its publications, membership, or meetings visit http://www.informs.org

INFORMS Journal on Computing
Vol. 19, No. 2, Spring 2007, pp. 302–312
ISSN 1091-9856 | EISSN 1526-5528 | 07 | 1902 | 0302

DOI 10.1287/ijoc.1050.0171
© 2007 INFORMS

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

Toward Automated Intelligent Manufacturing Systems (AIMS)

Hoi-Ming Chi, Okan K. Ersoy
School of Electrical and Computer Engineering, Purdue University, Electrical Engineering Building,
465 Northwestern Avenue, West Lafayette, Indiana 47907-2035, USA {hoiming@alumni.virginia.edu, ersoy@purdue.edu}

Herbert Moskowitz, Kemal Altinkemer
Krannert School of Management, Purdue University, 403 West State Street,
West Lafayette, Indiana 47907-2056, USA {herbm@purdue.edu, kemal@purdue.edu}

Information technology (IT) has been the driver of increased productivity in the manufacturing and service sectors, bringing real-time information to decision makers and process owners to improve process behavior and performance. Thus, organizations have invested heavily in training their employees to use IT in a disciplined, scientific way to make process improvements. This has spawned such popular initiatives as Six Sigma, yielding significant returns, but at considerable investment in training in statistical-analysis and decision-making tools. Can aspects of the decision-making process be automated, letting humans do what they do best (create, define, and measure) and machines (e.g., learning machines) do what they do best (analyze)? We propose an *automated intelligent manufacturing system* (AIMS) for analysis and decision making that mines real-time or historical data, and uses statistical and computational-intelligence algorithms to model and optimize enterprise processes. The algorithms employed involve a regression support vector machine (SVM) for model construction and a genetic algorithm (GA) for model optimization. Performance of AIMS was compared to Six-Sigma-trained teams employing statistical methodologies, such as design of experiments (DOE), to improve a simulated manufacturing operation, a three-stage TV-manufacturing process, where the objectives were to maximize yield, minimize cycle time and its variation, and minimize manufacturing costs, which were affected by conflicting defects and their causes. AIMS generally outperformed the teams on the above criteria, required relatively little data and time to train the SVM, and was easy to use. AIMS could serve as a productivity springboard for enterprises in existing and emergent technologies, such as nanotechnology and biotechnology/life sciences, where environment and miniaturization may make human monitoring and intervention difficult or infeasible.

Key words: support vector machine; genetic algorithm; Six Sigma; design of experiments (DOE); response-surface designs; machine learning; computational intelligence; intelligent systems
History: Accepted by Michel Gendreau, Area Editor for Heuristic Search and Learning; received April 2004; revised February 2005, May 2005, June 2005, October 2005; accepted December 2005.

1. Introduction

Manufacturing industries are concerned with producing higher-quality products, while minimizing production cost and time. However, many factors or drivers can affect this goal, and their interactions and effects are often difficult to uncover. In a TV production line, factors affecting quality and process performance might include environmental working conditions, design of the printed-circuit board, machine technology, and operator skills. Like total quality management (TQM) in the 1980s, which changed quality thinking, information technology (IT) is transforming quality-improvement practices from art to science by providing real-time data on the state of a process to owners for improvement. This spawned a business strategy and a disciplined, information-intensive problem prevention/solving process known as Six Sigma.

First advanced by Motorola around 1986 (Buetow 1996), Six Sigma is a scientific approach to problem prevention/solving that involves rigorous training in statistical tools to improve customer satisfaction by reducing process variation. It identifies the root cause of defects in products, processes, and services, resulting in cost reduction that can be passed on to the customer. This eliminates waste and reduces cycle time and its variation, which translates to on-time delivery (Harry 1994a, b). There are five phases (DMAIC):

- Define: Identify customers, individuals or entities that would be affected by the product, and define the problem with customer and organizational performance metrics.
- Measure: Assess process performance via unit costs, yields, cycle time, etc. Determine "hidden factory" (nonvalue-added, waste) costs suggesting improvement via reduced production to satisfy demand, reduced cost per unit, and decreased cycle time.

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

- Analyze: Determine the relation between system input and output variables. Develop a strategy and priority for process improvement—what system elements and metrics should one improve, and what is the impact on the system in terms of creating new bottlenecks, etc.?
- Improve: Apply Six Sigma statistical tools such as design of experiments (DOE), response surfaces, and optimization, to improve the process and assess impact on process performance and customer satisfaction.
- Control: Position and monitor statistical-process control (SPC) in real time to assess the process and the impact of process improvements, and detect special causes or disturbances (anomalies, malfunctions) that can disrupt and adversely affect system performance.

The first two phases require creativity, best performed by humans. The last three phases are statistical analysis, more effectively performed automatically via, e.g., data mining, machine learning, and computation. Common practice is to train people in statistics as "black belts" and "green belts" to perform these latter three phases, but this is difficult, expensive, and easily forgotten if not used repetitively, or if there is turnover.

Advances in IT have motivated manufacturers to automate data/information gathering, so it is logical to automate aspects of the (Six Sigma) decision-making process. Such a semi-automated system may be applicable for both existing enterprises and emerging technologies such as nanotechnology, biotechnology, and IT, needing very fast statistical and computational tools, and iterative global optimization. Many variables, parameters, and dimensions are involved and human intervention is minimal.

The statistical and computational algorithms are a regression *support vector machine* (SVM) and a *genetic algorithm* (GA) for modeling and optimizing. Artificial neural networks in finance (Fadlalla and Lin 2001) and credit-risk evaluation (Baesens et al. 2003) have been proposed, but few applications have been attempted in manufacturing.

In the next section, the basic structure of our *automated intelligent manufacturing system* (AIMS), which includes SVM for regression and model reduction, least-squares support vector machines (LSSVM) for regression, and the GA, are introduced. Section 3 describes an experiment comparing AIMS against Six-Sigma-trained manufacturing-student teams and some existing multivariate statistical techniques in a simulated manufacturing environment. Conclusions and future research are in Section 4. AIMS and DOE are compared, and their differences are discussed, in Appendix 1 in the Online Supplement to this paper on the journal's website.

2. AIMS Structure

AIMS uses data mining to model a factory (or any) operation and adjust the input settings and configurations to optimize performance, and complement classical DOE. Unlike DOE, AIMS may not require experimental designs, so can save significant money and time.

AIMS has two major components: training a learning algorithm (e.g., a second- or higher-order polynomial SVM for regression) (the modeling process), and a global optimization algorithm like GA (the optimization process).

2.1. SVMs for Regression

Support-vector learning (Vapnik 1995) is a machine-learning algorithm from statistical learning theory, formulated as a convex quadratic-programming problem (QPP) with inequality constraints. It implicitly maps the input vectors to the feature vectors through kernel functions, and then constructs a linear function with an ε-insensitive loss function (see below) in the feature space for best generalization. SVMs have attractive features and promising empirical performance over other conventional learning algorithms, e.g., medical-image segmentation (Wang et al. 2001), signal processing (Vapnik et al. 1997), time-series prediction (Muller et al. 1997), and financial forecasting (Trafalis and Ince 2000).

The primal QPP formulation of a SVM (Smola and Scholkopf 1998) is

$$\text{minimize} \quad \tfrac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}(\zeta_i + \zeta_i^*) \tag{1}$$

$$\text{subject to} \quad \begin{cases} y_i - \mathbf{w}^T\phi(\mathbf{x}_i) - b \le \varepsilon + \zeta_i \\ \mathbf{w}^T\phi(\mathbf{x}_i) + b - y_i \le \varepsilon + \zeta_i^* \\ \zeta_i, \zeta_i^* \ge 0, \end{cases}$$

where $\mathbf{x}_i$ is the ith data vector, y_i is its output response, ξ_i is the slack variable, N is the number of data vectors, $\mathbf{w}$ is the coefficient vector of the decision function, $\phi(\cdot)$ is the nonlinear mapping function from the input space to the (high-dimensional) feature space, C is the regularization parameter, and b is the bias. In SVM for regression, a special loss function, called the *ε-insensitive loss function*, is used. It linearly penalizes deviations larger than ε, and is zero otherwise. The regularization parameter C determines the tradeoff between the flatness $\|\mathbf{w}\|^2$, which relates to the generalization ability of the decision function, and the deviations ζ_i and ζ_i^* larger than ε from the actual target values. The relationship between ε and of ζ_i and ζ_i^* is given in Smola and Scholkopf (1998).

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

The primal formulation (1) can be converted to a dual using the Karush-Kuhn-Tucker (KKT) conditions (Fletcher 1987):

$$\text{maximize} \quad -\tfrac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)K(\mathbf{x}_i,\mathbf{x}_j) - \varepsilon\sum_{i=1}^{N}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{N}y_i(\alpha_i-\alpha_i^*) \qquad (2)$$

$$\text{subject to} \quad \begin{cases} \sum_{i=1}^{N}(\alpha_i-\alpha_i^*)=0 \\ 0\le \alpha_i,\alpha_i^*\le C, \end{cases}$$

where α_i and α_i^* are the Lagrangian multipliers and $K(\mathbf{x}_1,\mathbf{x}_2)$ is the kernel function $\phi^T(\mathbf{x}_1)\phi(\mathbf{x}_2)$ if Mercer's condition (Vapnik 1995) is satisfied.

The optimal w is

$$\mathbf{w}=\sum_{i=1}^{N}(\alpha_i-\alpha_i^*)\phi(\mathbf{x}_i). \qquad (3)$$

The corresponding decision function is

$$f(x)=\mathbf{w}^T\phi(\mathbf{x})+b=\sum_{i=1}^{N}(\alpha_i-\alpha_i^*)K(\mathbf{x}_i,\mathbf{x})+b. \qquad (4)$$

$K(\cdot,\cdot)$ is simply a function of the input vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ and can take on different forms, like polynomial, radial basis, or hyperbolic tangent. This implicit kernel mapping avoids computation in the high-dimensional feature space, and thus avoids the curse of dimensionality.

SVM has been shown to possess predictive power and can also have explanatory power by selecting an appropriate kernel function, as we show using a SVM with a second-order polynomial kernel function

$$K(\mathbf{x},\mathbf{y})=(\mathbf{x}^T\mathbf{y}+1)^2, \qquad (5)$$

where $\mathbf{x}$ and $\mathbf{y}$ are the input vectors. Thus, SVM with a second-order polynomial kernel function was used to model the manufacturing process.

Having explanatory power means the following mathematical model is produced for a second-order polynomial kernel function:

$$y=w_0+w_1x_1+\cdots+w_mx_m+w_{m+1}x_1^2+\cdots+w_{2m}x_m^2 + w_{2m+1}x_1x_2+w_{2m+2}x_1x_3+\cdots. \qquad (6)$$

Namely, the response y is a linear combination of linear, quadratic, and two-way interaction terms of the input variables (similar to regression analysis or DOE). (6) has explanatory power because each w represents the effect of each right-hand-side term on the output response y. If w_i is larger, then x_i has a larger effect on y, assuming the data have been properly normalized. Therefore, unlike artificial neural networks that operate like a "black-box," an SVM with a second-order polynomial kernel function possesses explanatory power, which allows for easy explanation, interpretation, and understanding of the process. If the process is more complex, a higher-order polynomial or other functional SVM may be used instead. GA can then seek the optimum input settings for maximizing single or multiple conflicting criteria performance.

Proof that a second-order polynomial kernel function produces a decision function consisting of a linear combination of linear, quadratic, and two-way interaction terms of the input variables in (6) is given in Appendix 2 in the Online Supplement.

2.2. SVM Model Reduction

In real situations, many of the terms in (6), especially the interaction and quadratic terms, are insignificant. In fact, by removing the insignificant terms, prediction accuracy can often be improved due to the bias-variance trade-off (Hastie et al. 2001), as well as producing a more parsimonious model. Thus, we apply subset selection to remove insignificant terms.

Two subset-selection algorithms were considered: forward and backward stepwise (Draper and Smith 1981). We used forward stepwise selection because the final equation usually consists of only a few terms.

In forward stepwise selection, we start with only the intercept initially, and then add into the model the term that most improves the fit, measured by

$$F=\frac{\mathrm{RSS}(\widehat{\mathbf{w}})-\mathrm{RSS}(\widetilde{\mathbf{w}})}{\mathrm{RSS}(\widetilde{\mathbf{w}})/(N-k-2)} \qquad (7)$$

(Hastie et al. 2001), where RSS is the residual sum of squares

$$\mathrm{RSS}(\mathbf{w})=\sum_{i=1}^{N}\Bigg(y_i-w_0-\sum_{j=1}^{m}x_{ij}w_j-\sum_{j=1}^{m}x_{ij}^2w_{j+m} -\sum_{j=1}^{m}\sum_{k=1}^{m}x_{ij}x_{ik}w_{2m+(j-1)m+k}\Bigg)^2 \qquad (8)$$

and $\widehat{\mathbf{w}}$ is the set of the k coefficients of the current model, and $\widetilde{\mathbf{w}}$ is the set of the $k+1$ coefficients after adding a term. The term with the highest F value is added to the model, and the procedure iterates, terminating when no term produces an F value greater than the upper 95th percentile of the $F_{1,N-k-2}$ distribution.

In forward (or backward) stepwise selection, the second-order polynomial kernel function can no longer be used. Whenever a term is selected in the forward procedure (or removed in the backward procedure), it is very difficult, if not impossible, to find the

RIGHTSLINK

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

corresponding kernel function with the term selected (or removed). Thus, we use the linear kernel function and manually include (or exclude) the two-way interaction and quadratic terms in the input space. The major drawback of this is the computational load. If n is the dimension of the input space, in the first iteration, the number of linear SVMs that need to be trained to select the first term is

$$n+n+n(n-1)/2. \tag{9}$$

The first n is the number of linear terms, the second n is the number of quadratic terms, and $n(n-1)/2$ is the number of two-way interactions, so the number of linear SVMs constructed is $O(kn^2)$ where k is the number of terms added to the final equation. To speed up training, least-squares support-vector machines (LSSVMs) can be used in lieu of the regular SVMs (Suykens and Vandewalle 1999). LSSVMs differ from regular SVMs by changing the inequality constraints to equality constraints and using the quadratic loss function. Thus, the LSSVM formulation becomes a linear-programming problem instead of a QPP in regular SVMs, so it can be solved much faster. This is described further below.

2.3. Least-Squares Support Vector Machines (LSSVMs) for Regression

An LSSVM originates from the classical SVM described in Section 2.1 by changing the inequality constraints to equality constraints with a formulation in the least-squares sense (quadratic-loss function).

The primal formulation of the LSSVM for regression is

$$\min \tfrac{1}{2}\|\mathbf{w}\|^2 + C\tfrac{1}{2}\sum_{i=1}^{N} e_i^2 \tag{10}$$

subject to

$$e_i = y_i - (\mathbf{w}^T\varphi(\mathbf{x}_i)+b) \quad i=1\ldots N,$$

where e_i is the error between the real target response of the ith data vector and the predicted output by the LSSVM. The Lagrangian and the KKT conditions (Fletcher 1987) become

$$L_{\text{LSSVM}} = \tfrac{1}{2}\|\mathbf{w}\|^2 + C\tfrac{1}{2}\sum_{i=1}^{N} e_i^2 + \sum_{i=1}^{N}\alpha_i[y_i - (\mathbf{w}^T\varphi(\mathbf{x}_i)+b) - e_i] \tag{11}$$

$$\begin{cases} \dfrac{\partial L_{\text{LSSVM}}}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{N}\alpha_i\varphi(\mathbf{x}_i) \\ \dfrac{\partial L_{\text{LSSVM}}}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N}\alpha_i = 0 \\ \dfrac{\partial L_{\text{LSSVM}}}{\partial e_i} = 0 \Rightarrow \alpha_i = Ce_i \quad i=1\ldots N \\ \dfrac{\partial L_{\text{LSSVM}}}{\partial \alpha_i} = 0 \Rightarrow y_i - [\mathbf{w}^T\varphi(\mathbf{x}_i)+b] - e_i = 0 \\ \qquad\qquad i=1\ldots N. \end{cases} \tag{12}$$

Since this is a convex optimization problem, any solution that satisfies all the KKT conditions is the global minimizer (Chong and Zak 1996), which is given by the solution to

$$\begin{bmatrix} 0 & \mathbf{1}_{N\times 1}^T \\ \mathbf{1}_{N\times 1} & \Omega + (1/C)I \end{bmatrix} \begin{bmatrix} b \\ \vec{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{y} \end{bmatrix}, \tag{13}$$

where $\vec{\alpha} = [\alpha_1, \ldots, \alpha_N]^T$, $\mathbf{1}_{N\times 1}$ is an $N\times 1$ column vector containing all one's, $\mathbf{y}$ is an $N\times 1$ column vector containing the target response for all training vectors, and Ω is an $N\times N$ Hessian matrix with elements $\Omega_{ij} = \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j)$ if Mercer's condition is satisfied.

Unlike classical SVMs, the global minimizer for LSSVMs is much easier to obtain, by solving (13) instead of quadratic programming, using any optimization package. However, this comes with a price: sparseness of the support vectors is lost. In classical SVMs, most of the Lagrangian multipliers α_i's are zero, which is not true in LSSVMs because the α_i's are proportional to the errors, e_i's, as shown in the third KKT condition in (12). A pruning method has been proposed to overcome this problem (Suykens et al. 2002a). LSSVM is closely related to Fisher's linear discriminant analysis in the feature space with a regularization term (ridge regression) (Van Gestel et al. 2004), and has been extended beyond regression and classification to, e.g., kernel principal component analysis (PCA) (Suykens et al. 2002c) and recurrent networks (Suykens and Vandewalle 2000).

2.4. Genetic Algorithm (GA) for Our Optimization

In a manufacturing process, there are usually several conflicting objectives, which results in a highly multimodal objective function, especially with an increasing number of input factors or output responses. Thus, we use GAs for our optimization.

3. A Process-Improvement Experiment: AIMS vs. Six-Sigma-Trained Teams

This experiment compared and evaluated AIMS against Six-Sigma-trained teams to improve a simulated TV-manufacturing process. AIMS performance and behavior could also be compared to the "true" model and optimal solution, which was predetermined by the experimenter. The Six-Sigma team and the competition exercise are described in Appendix 3 in the Online Supplement.

3.1. Computer Animated Simulation of Manufacturing/Assembly Process

An animated simulation model of a TV production line was developed, including Automatic Component Insertion (ACI), Manual Assembly (MA), and

RIGHTSLINK

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

Solder (S). A description is in Seo and Moskowitz (2002) and a flowchart of the process is in Figure A2 in the Online Supplement. For each subsystem (ACI, MA, and S), there are eight "X" input variables that can cause six specific types of conflicting defects ("Y" output variables), described in Figure A3 in the Online Supplement. After each subsystem, the product is checked for quality based on the values of the Y's, and is classified as good, rework, or scrap. Reworked units are recycled through the operation until they become good or are scrapped. X's and Y's are related by "hidden functions" through the simulation model. The optimal relationships are inputted by and known to the experimenter (but not to AIMS or the Six-Sigma teams). The objective was to specify the X's at each station that would maximize yield at each station and for the entire system as a whole by driving down all the various types of defects.

3.2. Training AIMS

3.2.1. Modeling Using SVMs or LSSVMs. Simulated data were generated and all training input variables were normalized to zero mean and unit variance. Using this normalized data set, a second-order polynomial (SVM or LSSVM) was trained for each output Y in each subsystem (ACI, MA, S) to fit (6). The second-order polynomial SVM was implemented in MATLAB using SVMlight (Joachims 1998), and its MATLAB interface by Schwaighofer (2002). The LSSVM was implemented in MATLAB using the LSSVMlab toolbox by Suykens et al. (2002b). Before training an SVM or LSSVM, certain parameters (the kernel type and its associated parameters, as well as the regularization parameter C) must be chosen to ensure good prediction performance. This is called *model selection*, and is problem-specific. Typically there is a grid search over a wide range of the parameter space to locate the parameter values that yield the lowest validation error, which is computationally intensive. But for us, since a second-order polynomial kernel function is usually sufficient to model a typical manufacturing process, the only parameter left is the regularization parameter C. This reduces the model-selection problem to a one-dimensional search. C was chosen using the validation method via training the SVM or LSSVM on the training data and validating it on a separate set of data, and selecting C to minimize validation mean-square error.

3.2.2. Optimization Using GAs. After training all the SVMs or LSSVMs, the GA was applied to each of the three subsystems based on the models generated by the SVMs or LSSVMs. The GA was implemented in MATLAB using the GA toolbox by Chipperfield et al. (1994). Most of the input variables were integer-valued, and so integer encoding was used. Since there are six outputs in each subsystem, we have a multiple conflicting-criteria optimization problem. Therefore, a desirability function was used in the GA as the objective function:

$$D = \prod_{i=1}^{6} d_i^{1/6}, \tag{14}$$

where

$$d_i = \begin{cases} 0 & \hat{y}_i > A \\ \left(\dfrac{\hat{y}_i - A}{B - A}\right)^t & B \le \hat{y}_i \le A \\ 1 & \hat{y}_i < B \end{cases}$$

and $\hat{y}_i$ is estimated response i from the SVM or LSSVM, B is the target value (0 in this case), A is the maximum acceptable response (set to 50), and t is a subjective weight (set to 10). The overall desirability D is simply the geometric mean of all individual desirabilities d_i. This is the same desirability function used in DOE when dealing with multiple conflicting-criteria optimization problems. If one of the estimated responses $\hat{y}_i$ is greater than A, its corresponding individual desirability will become zero, making the overall desirability D zero as well. In the GA toolbox, there is another linear scaling function that converts the objective-function values (the same as the overall desirability) to a fitness value, upon which the GA will select the individuals for reproduction. The purpose of this scaling function is to differentiate further the "goodness" of each individual objective's overall desirability. The GA gave the optimum input settings for each subsystem. By using these optimum input settings in the simulator, the overall yield should reach a maximum with the learned model.

Similar to training an SVM or LSSVM, some parameters in the GAs must be chosen properly to ensure good performance, both in terms of finding a global (or a very deep) optimum and a rapid rate of convergence. These parameters include the size of parent population, parent/offspring ratio, selection type, crossover type, crossover rate, mutation type, and mutation rate. These parameters are all inter-related and have great impact on the GA's performance. For example, a large parent population requires more computation but allows a more thorough search for a global optimum over the entire input space. On the other hand, a small population may speed up convergence (due to fewer computations in each generation), but it may yield a local optimum as the GA terminates because it is not the case that the entire input space is searched thoroughly. The effects of the other parameters are more subtle, and probably can best be discovered through experimentation. Choice of these parameters is usually problem-specific, but fortunately, there are usually some guidelines.

RIGHTSLINK

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

Table 1 Initial and Final GA Settings Used in Experiment

		Final settings		
Parameter	Initial settings for ACI, MA, and S	ACI	MA	S
Parent population	20	20	20	170
Parent/offspring ratio	1:7	1:3	1:3	1:7
Selection type	Stochastic universal sampling	Stochastic universal sampling	Stochastic universal sampling	Stochastic universal sampling
Crossover type	Single-point	Single-point	Single-point	Single-point
Crossover rate	0.85	0.85	0.85	0.95
Mutation type	Uniform	Uniform	Uniform	Uniform
Mutation rate	0.1	0.1	0.1	0.1

The parameters were initially chosen according to the settings for the robust genetic algorithm proposed by Ortiz et al. (2004), as shown in Table 1. Due to implementation restrictions of the GA toolbox, the selection type was chosen to be "Stochastic Universal Sampling" instead of "Tournament." In addition, since the inputs were discrete variables, the mutation type and the mutation rate were set to "Uniform" and 0.1, respectively. Also shown are the final settings. For ACI and MA, the parent/offspring ratio was reduced to 1:3 to save computation but without any loss in performance. For S, the parent population size was raised to 170 to increase the search space for locating the global optimum.

A screen capture (Figure A1) of a preliminary version of the AIMS software interface and its description are given in Appendix 4 in the Online Supplement.

3.3. Experimental Results

The Six Sigma simulator was set to run to produce 250 good units (a job order of TV sets) using the optimum input settings for each of the three subsystems, ACI, MA, and S, obtained from the following various sources and approaches: (a) Six-Sigma-trained teams using DOE, response-surface design, and optimization, (b) stepwise regression using Minitab to model the processes and a GA to optimize the process, (c) the full AIMS using an SVM and a GA, (d) the reduced AIMS using an LSSVM and a GA, and (e) the true optimum. Here, the true optimum refers to the optimal solution obtained from Excel Solver using the equations relating X and Y in the Six Sigma simulator.

3.3.1. Prediction Accuracy of Each Modeling Technique. Before comparing the percent yield via various approaches, the prediction accuracy of each approach was first assessed using the root mean square (RMS) error between the predicted outputs and the actual outputs of an independent test data set, which was not used in training the algorithms. The test-data set contained 750 samples in ACI and MA, and 500 in S. Table 2 shows the test and training RMS errors obtained using stepwise regression, full AIMS, and reduced AIMS. The smallest training and test RMS errors for each output are in bold. In general, reduced AIMS produced the smallest test RMS error for most outputs, indicating that it is the most accurate model. The reason the reduced AIMS outperformed stepwise regression is believed to be due to its robustness by having the regularization parameter C control the trade-off between the empirical training error and the model complexity. By choosing an appropriate value of C, overfitting can be avoided, achieving better generalization. The full AIMS did not perform as well because a full model consists of all the main effects, quadratic terms, and interaction terms, many of which were not significant. Namely, only a few interaction terms (or main and quadratic terms) were significant for the actual (optimal) equations in the Six Sigma simulator. Therefore, a full

Table 2 Training and Test Root Mean Square Errors Obtained Using Three Methods in ACI, MA, and S

	Training RMS error			Test RMS error		
Output	Stepwise regression	Full AIMS	Reduced AIMS	Stepwise regression	Full AIMS	Reduced AIMS
ACI						
1	5.469	**5.322**	5.559	6.379	7.306	**6.254**
2	1.345	**1.342**	1.389	1.963	2.012	**1.917**
3	**4.843**	4.869	5.221	6.661	7.454	**6.408**
4	**1.482**	1.483	1.510	1.709	1.769	**1.681**
5	**1.508**	1.574	1.566	1.671	1.713	**1.624**
6	2.149	**2.109**	2.111	**2.182**	2.341	2.195
MA						
1	**3.909**	4.018	4.070	4.900	5.223	**4.879**
2	1.385	1.386	**1.379**	1.600	**1.563**	1.607
3	**6.819**	7.076	7.587	8.828	9.205	**8.270**
4	**2.746**	2.758	2.749	**2.906**	3.086	2.912
5	1.796	**1.763**	1.841	2.336	**2.333**	2.340
6	2.858	**2.847**	3.045	3.770	4.010	**3.520**
S						
1	5.975	5.521	**5.465**	6.739	6.727	**6.508**
2	2.202	2.291	**1.645**	2.472	2.634	**1.854**
3	29.385	**23.246**	24.817	28.756	**24.772**	25.158
4	186.563	**62.105**	63.541	189.173	75.627	**68.551**
5	0.401	**0.146**	0.147	0.401	0.152	**0.140**
6	1.832	1.877	**1.815**	**1.930**	2.059	1.959

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

second-order polynomial SVM model consumed valuable degrees of freedom in estimating parameters of insignificant variables (in fact, some of the equations consisted of only the main effects), and this also led to overfitting. As with statistical models, it demonstrates the importance of selecting the proper terms for the models.

Finally, although the reduced AIMS in general gave the smallest test RMS errors, the other two algorithms performed well in most cases, except in output 4 of the S subsystem, where stepwise regression resulted in a significantly higher test RMS error. This result also explains the poor performance occurring when applying the GA to the S models generated by stepwise regression (see Section 3.3.3).

3.3.2. GA Results. Since the methodology of applying the GA to the models generated by each approach (stepwise regression, full AIMS, and reduced AIMS) is very similar, results from only the reduced AIMS model will be shown. The GA was performed 30 times for each subsystem, with a different initial parent population in each trial. The final GA parameter settings in Table 1 were determined experimentally to ensure that the GA converged to the same final optimum value in all 30 trials, while simultaneously minimizing computation. Figure 1 shows a typical plot of the average overall desirability of all the individuals in each generation vs. the generation step. For all three subsystems, the average overall desirability converged to a final value after several (fewer than ten in all three subsystems) generations.

As most of the time required by the GA was spent in computing the objective function (or the overall desirability) of the offspring, the total number of objective-function evaluations provides a good indication of the computational load required. Since the

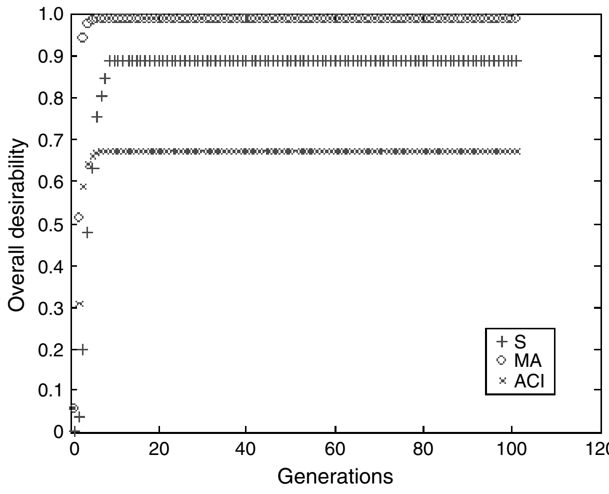

Figure 1 **Overall Desirability vs. Generations for ACI, MA, and S Using Reduced AIMS**

Table 3 **Number of Objective-Function Evaluations Required in Reduced AIMS**

	ACI	MA	S
Exhaustive brute force grid search	8,000	1,600	2.062×10^9
GA	512	396	1.19×10^5

input variables were discrete in our experiment, an exhaustive enumeration of all possible input settings can be obtained, and the objective function can be evaluated for all these input settings to locate the global optimum, although this is extremely computationally intensive. Table 3 shows the total number of objective-function evaluations in each subsystem needed by the brute-force exhaustive grid search and by the GA. Only a small fraction of the entire input space, especially in the S subsystem, was searched by the GA, attesting to its efficiency. The total number of objective-function evaluations required by the GA in Table 3 are overestimated, because in our experiment the goal was to locate the single optimum with the highest overall desirability in every trial. However, the usual practice is to achieve within 10% of the known true optimum (Ortiz et al. 2004).

The optimum input settings and the corresponding overall desirabilities for each subsystem are in Table 4. The standard deviation of the optimum setting of each input variable was normalized by its input range to obtain the standard-deviation-to-range ratio, which indicates the degree of variation of each optimum input setting, and allows comparison among all input variables. In ACI and MA, the GA converged to the same optimum input setting in every trial, indicated by the zero variance in the overall desirability and zero standard-deviation-to-range ratio in the optimum input settings. In S, the large standard-deviation-to-range ratios of x_3 and x_7 indicate that they have small impact on the optimum solution. This also agrees with the "do-not-care" values in the true optimum settings, which will be shown later in Table 5.

3.3.3. Percent Yield: Rolled Throughput (RTY). After obtaining the optimum input settings from the GA for each modeling method, they were fed into the Six Sigma simulator to simulate the actual production process. The yield (%) of each subsystem and overall system were averaged over ten independent simulations, each producing 250 good units, and are in Table 5. Overall system yield (RTY) was highly negatively correlated with performance metrics like cost-average cycle time (CT), and the standard deviation of CT, which is a measure of CT variation (Table 6). Hence, a high-yield RTY yields low average manufacturing costs, average cycle times, and cycle time variation.

RIGHTSLINK

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

Table 4 Mean and Standard Deviation/Range of the Final Optimum Input Settings and Mean and Variance of Overall Desirability over 30 Trials, Using Final GA Parameter Settings for the Reduced AIMS

	ACI		MA		S	
	Mean	Standard-deviation-to-range ratio	Mean	Standard-deviation-to-range ratio	Mean	Standard-deviation-to-range ratio
x_1	5	0	2	0	486.03	0.0036
x_2	5	0	0	0	475.13	0.0068
x_3	0	0	0	0	524.53	0.27
x_4	0	0	5	0	827.03	0.084
x_5	0	0	1	0	5.13	0.069
x_6	1	0	0	0	10	0
x_7	1	0	0	0	2.70	0.36
x_8	0	0	0	0	1	0
	Mean	Variance	Mean	Variance	Mean	Variance
Overall desirability	0.67	0	0.99	0	0.89	0.00030

Both the full and reduced AIMS (93.78% and 94.53%, respectively) performed essentially the same as the true optimum system performance (94.16%), and the best Six Sigma trained team (94.68%). Slight differences in the results and the fact that the true optimum achieved a slightly lower percent yield are believed to be due to sampling errors. Moreover, the three best generated solutions by the reduced AIMS all did pretty well (74.14% to 95.15%). However, there was considerable difference in performance between the best and worst Six-Sigma-trained teams (94.68% vs. 62.26% yield). The average overall yield across all five Six-Sigma teams was 86.06%.

For stepwise regression, there were five sets of input settings that all gave the same highest objective-function value in ACI using the GA, and the percent yields of these five sets of input settings were averaged and given in Table 5. Similarly, there were four sets of input settings that all gave the same highest objective-function value in MA using the GA, and the percent yields of these four sets of input settings were also averaged and given in Table 5.

Stepwise regression performed well for ACI and acceptably for MA (100% and 89.58% yields, respectively), but poorly for S, in which the GA returned many different optimum input settings with the same highest objective-function value. However, when these optimum input settings given by the GA were inputted into the Six Sigma simulator, the percent yield obtained ranged from 10% to 98% (this is why no optimum input settings for S using stepwise regression were shown in Table 5). This inconsistent performance by stepwise regression was believed to be due to an inaccurate explanatory model constructed by stepwise regression for output 4 of the S subsystem (see Section 3.3.1 and Table 2). As will be discussed in Section 3.3.7, the outputs of the S model were extremely sensitive to x_1, which is also a significant term in the equation of output 4. Therefore, a less accurate model of output 4 will result in slightly

Table 5 Optimum Settings Determined By Each Method Used in the Experiment and Corresponding Actual Percent Yield

	ACI		MA		S		
Sources and methods	Optimum input settings x_1–x_8	% Yield	Optimum input settings x_1–x_8	% Yield	Optimum input settings x_1–x_8	% Yield	Overall % yield
	1, 5, 0, 0, 0, 1, 1, 0	100	NA*, 0, 0, 5, 1, 0, 0, 0	95.95	485, 475, NA, 830, 5, 10, NA, 1	98.17	
True optimum	1, 5, 0, 0, 0, 1, 1, 0	100	3, 0, 0, 5, 1, 0, 0, 0	96.53	485, 475, 470, 818, 5, 10, 5, 1	98.08	94.16
Best Six-Sigma team	1, 5, 0, 0, 1, 1, 1, 1	73.18	5, 0, 1, 5, 1, 0, 0, 0	95.85	485, 475, 598, 830, 5, 10, 5, 0	88.78	94.68
Worst Six-Sigma team	1, 5, 0, 0, 0, 1, 1, 0 2, 5, 0, 0, 0, 1, 1, 0 3, 5, 0, 0, 0, 1, 1, 0 4, 5, 0, 0, 0, 1, 1, 0 5, 5, 0, 0, 0, 1, 1, 0	100	5, 0, 1, 5, 1, 0, 1, 0 5, 0, 1, 5, 1, 0, 0, 0 5, 0, 0, 5, 1, 0, 1, 0 5, 0, 0, 5, 1, 0, 0, 0	89.58	See explanation in text.		62.26
Stepwise regression	5, 4, 0, 0, 0, 1, 1, 0	99.98	3, 0, 1, 5, 1, 0, 0, 0	95.69	486, 475, 580, 821, 6, 10, 4, 1	98.02	
Full AIMS (SVM)	5, 5, 0, 0, 0, 1, 1, 0	100	2, 0, 0, 5, 1, 0, 0, 0	96.69	486, 475, 520, 828, 5, 10, 1, 1	97.78	93.78
Reduced AIMS (LSSVM)							94.53

*NA means do not care.

RIGHTSLINK Copyright Clearance Center

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

Table 6 Correlation Matrix Relating System Performance Metrics

	RTY	Cost	Avg CT	Std dev CT
RTY	1.00			
Cost	−0.81 (0.016)	1.00		
Avg CT	−0.86 (0.006)	0.98 (0.000)	1.00	
Std dev CT	−0.91 (0.002)	0.97 (0.000)	0.99 (0.000)	1.00

deviated optimum input settings by the GA. However, because of the model's high sensitivity to x_1, some of these slightly deviated optimum input settings yielded low percent yield, which explains the inconsistent performance by stepwise regression.

Although the full and reduced AIMS, as well as the best Six-Sigma team, obtained yields essentially equivalent to the true optimum, the input settings for the subsystems were slightly different and can be accounted for by the insensitivity of these inputs on performance. Comparing the full and reduced AIMS, we observed that the input settings were slightly different for all three subsystems, although the yields were comparable. However, reduced AIMS has fewer variables and a simpler mathematical structure (fewer variables, interaction, and second-order polynomial terms; see Tables A3a–c in the Online Supplement).

3.3.4. Data Quantity and Implementation. We also evaluated and compared the amount of data needed to train AIMS versus the time required by Six Sigma teams to optimize the process. Table 7 shows that AIMS required significantly less data (less than half on the ACI and S subsystems) than did the best Six Sigma team. Moreover, Six Sigma teams spent from 20 to 60 hours creating, performing, and analyzing experiments to improve and optimize the process. This compares to virtually instantaneous training of AIMS, which does not require any experimentation and only a few minutes of computer time for (in this case) the second-order polynomial SVM or LSSVM and the GA. While it may be argued that in real-world settings where data cost money, time, and resources, experimentation, data collection, and analysis would be more parsimonious. Nevertheless, it seems clear that AIMS in this application and as a general concept can be much more efficient, and at least as effective as, Six-Sigma-trained teams for process improvement and optimization.

3.3.5. Variation in Data Needed to Train AIMS. The quantity of data required to train AIMS properly is problem-dependent. For example, for the simulated manufacturing process, using the full second-order polynomial SVM model, the relationship can be observed in Figure 2. ACI and MA were comparatively easier to model than S because many inputs to ACI and MA were binary, while the equation that generated the data for S in the simulator contained quadratic terms. Thus, S required more data to model than ACI or MA.

Table 7 Amount of Data Needed to Train AIMS vs. Amount Used by Six Sigma Teams to Optimize Process

	ACI	MA	S
Best Six Sigma team	446	250	1,033
AIMS	200	200	500

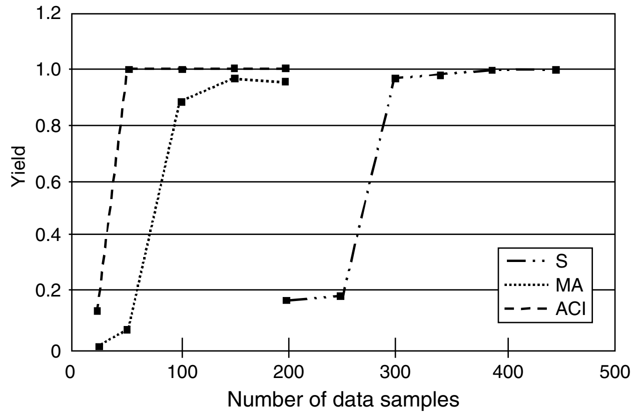

Figure 2 Yield vs. Number of Training Data Samples Using a Full Second-Order Polynomial AIMS

As a general and perhaps obvious note, the data being collected should always have sufficient variability to represent the input space well. As with all modeling, the model generated by AIMS will only be confident to be accurate in this region of the entire input space. The GA will still be able to find the "global" optimum in this small region based on the model generated, but this "global" optimum in this small region may actually be just a "local" optimum in the entire input space.

3.3.6. Verification of the Explanatory Power of AIMS. To verify the explanatory power of AIMS, we compared the true equations of the Six-Sigma simulator with those obtained from stepwise regression and AIMS using a reduced second-order polynomial LSSVM model. The results are in Tables A3a–c in the Online Supplement. Almost all actual outputs in the simulator were generated from Poisson random processes. The only exception is Y_2 in the ACI subsystem, which is Gaussian. The reduced AIMS usually contained terms that appear in the true equations, plus some additional terms, so the reduced AIMS was able to identify the most significant terms. The number of additional terms in the model can be adjusted by changing the critical F statistic.

In general, the explanatory power by stepwise regression was worse than that of the reduced AIMS model. More terms that were in the true equations were missing from the equations generated by stepwise regression than by reduced AIMS.

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

3.3.7. Sensitivity Analysis. To illustrate the effect of each input variable on the outputs, sensitivity analysis was performed. Since a second-order polynomial function was used to approximate the X-Y relationships, it is very easy to analyze sensitivity of the mathematical model generated by AIMS. The importance of term j can be measured by squaring and standardizing the regression coefficients in (6) for each m-th output as follows (Saltelli et al. 2000):

$$(w^m_{j,\,\text{standardized}})^2 = \left(\frac{w^m_j \hat{s}_j}{\hat{s}^m}\right)^2, \tag{15}$$

where

$$\hat{s}^m = \left[\sum_{k=1}^{N} \frac{(y^m_k - \bar{y}^m)^2}{n-1}\right]^{1/2}, \qquad \hat{s}_j = \left[\sum_{k=1}^{N} \frac{(x_{kj} - \bar{x}_j)^2}{n-1}\right]^{1/2}. \tag{16}$$

Note that (16) is valid for all coefficients of the linear terms $x_1, \ldots, x_8$. For the coefficients of the quadratic and interaction terms, $\bar{x}_j$ is replaced by the mean of the corresponding quadratic or interaction term. This is multivariate sensitivity analysis, as there are six outputs $y^1, \ldots, y^6$. (15) gives only the sensitivity measure of one particular y^i with respect to the term j. To compute the overall sensitivity for each input factor x_i, a weighted average is used (Fassò 2002):

$$\text{Overall Sensitivity of } j\text{th Term} = \frac{\sum_{m=1}^{6} (w^m_{j,\,\text{standardized}})^2 \hat{s}^{m2}}{\sum_{m=1}^{6} \hat{s}^{m2}}. \tag{17}$$

The overall sensitivities of the outputs of the three subsystems to all linear, quadratic, and interaction terms are shown in Table A4 in the Online Supplement. The reduced AIMS was used here to approximate the three subsystems. The important observation in Table A4 is that in the S subsystem, the overall sensitivities of x_1 and x_1^2 were significantly larger than those of the other terms (about 10^5 larger), indicating that the S outputs are extremely sensitive to the variation in x_1. A small perturbation in x_1 should result in significant change in the S subsystem outputs, and so is the percent yield. More specifically, if the input setting of x_1 was changed by ± 2 from the true optimum setting of 485 (see Table 5), then the percent yield of the S process was experimentally determined to drop from 98% to almost 0%. Therefore, careful attention should be paid to adjusting the setting of x_1 in the S subsystem.

Sensitivity can be measured by using (15) only if the underlying mathematical model is a kth-order polynomial function. As mentioned before, the SVM can use a hyperbolic or a radial basis kernel function to model a more complex process. In these cases, sensitivity can be measured only by applying a variance-based approach, such as the Fourier amplitude sensitivity test (FAST) (Cukier et al. 1973, Sobol 1993).

4. Conclusions and Future Research

In part, based on our experimental study, AIMS shows promise as a potential methodology in manufacturing system modeling and optimization. It is automatic, requiring no experimental design. AIMS may be especially effective for emerging technologies and associated complex manufacturing enterprises of the future, such as nanotechnology, biotechnology, and pharmaceuticals. These industries usually involve numerous small components that cannot be tracked or monitored easily by humans. Therefore, successful deployment of AIMS has potential to play a useful role in such emerging technologies.

AIMS can be further modified and improved by the following:

- Incorporating Bayesian analysis that makes use of the prior knowledge of the training data. This allows selecting an appropriate model for the system using probabilities in the Bayesian sense.
- Building a feedback loop to track and adapt continually to any changes present in the manufacturing system. If there is a sudden internal change in the system, the feedback loop can detect the change and alarm the user. The user then probably needs to collect new data and retrain the AIMS.
- Incorporating sensitivity analysis to illustrate the relative importance of each input variable or factor. Sensitivity analysis is particularly essential and beneficial in cases where the process system is modeled by a complex nonlinear mathematical structure, other than a second-order polynomial SVM. In such situations, the variance-based approach to global sensitivity analysis, such as the Fourier amplitude sensitivity test (FAST) (Cukier et al. 1973, Sobol 1993), can be used to discover how variations in the outputs of a model can be related to variations in the model inputs.
- Investigating different model-reduction or feature-selection methods in AIMS.
- Implementing AIMS in an actual manufacturing environment. We are in the process of doing so in a major pharmaceutical firm. The results of this study, which so far has been successful, will further help to assess its feasibility and applicability in real-world systems.

Acknowledgments

Special thanks to Professor W. David Kelton, the Editor-in-Chief of *INFORMS Journal on Computing*, for improving the clarity and the crispness of this paper, to Professor Gary Koehler of the University of Florida for his valuable comments, especially on the genetic algorithm, and to the reviewers for their time and valuable comments on this paper.

RIGHTSLINK

Downloaded from informs.org by [144.214.89.16] on 18 September 2017, at 07:51 . For personal use only, all rights reserved.

References

Baesens, B., R. Setiono, C. Mues, J. Vanthienen. 2003. Using neural network rule extraction and decision tables for credit-risk evaluation. *Management Sci.* **49** 312–329.

Buetow, R. 1996. The Motorola quality process: Six Sigma. G. M. Bounds, ed. *Cases in Quality*. Irwin, Chicago, IL, 229–249.

Chipperfield, A. J., P. J. Fleming, H. Pohlheim, C. M. Fonseca. 1994. Genetic algorithm toolbox user's guide. ACSE Research Report 512, University of Sheffield, Sheffield, UK.

Chong, E. K. P., S. H. Zak. 1996. *An Introduction to Optimization*. Wiley, New York.

Cukier, R. I., C. M. Fortuin, K. E. Shuler, A. G. Petschek, J. H. Schaibly. 1973. Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. I Theory. *J. Chemical Phys.* **59** 3873–3878.

Draper, N. R., H. Smith. 1981. *Applied Regression Analysis*, 2nd ed. Wiley, New York.

Fadlalla, A., C. H. Lin. 2001. An analysis of the applications of neural networks in finance. *Interfaces* **31** 112–122.

Fassò, A. 2002. Nonlinear multivariate statistical sensitivity analysis of environmental models with application to heavy metal adsorption from contaminated wastewater. *Internat. Environmetrics Soc.* (*TIES*) *Conf.*, Genova, Italy.

Fletcher, R. 1987. *Practical Methods of Optimization*. Wiley, New York.

Harry, M. J. 1994a. *The Vision of Six Sigma: A Roadmap for Breakthrough*. Sigma, Phoenix, AZ.

Harry, M. J. 1994b. *The Vision of Six Sigma: Tools and Methods for Breakthrough*. Sigma, Phoenix, AZ.

Hastie, T., R. Tibshirani, J. Friedman. 2001. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, New York.

Joachims, T. 1998. Making large-scale SVM learning practical. B. Schölkopf, C. Burges, A. Smola, eds. *Advances in Kernel Methods: Support Vector Learning*. MIT Press, Cambridge, MA, 169–184.

Muller, K. R., A. Smola, G. Ratsch, B. Scholkopf, J. Kohlmorgen, V. Vapnik. 1997. Predicting time series with support vector machines. *Internat. Conf. Artificial Neural Networks* (*ICANN*), Lausanne, Switzerland, 999–1004.

Ortiz, F., J. R. Simpson, J. J. Pignatiello, A. Heredia-Langner. 2004. A genetic algorithm approach to multiple-response optimization. *J. Quality Tech.* **36** 432–450.

Saltelli, A., K. Chan, E. M. Scott. 2000. *Sensitivity Analysis*. Wiley, Chichester, England.

Schwaighofer, A. 2002. *MATLAB Interface to SVM Light*. Institute for Theoretical Computer Science, Graz University of Technology, Graz, Austria, http://www.cis.tugraz.at/igi/aschwaig/software.html.

Seo, H., H. Moskowitz. 2002. *Six Sigma Simulator*. Unpublished Research Report, Krannert School of Management, Purdue University, West Lafayette, IN.

Smola, A., B. Scholkopf. 1998. A tutorial on support vector regression. NeuroCOLT2 Technical Report Series, NC2-TR-1998-030.

Sobol, I. M. 1993. Sensitivity analysis for nonlinear mathematical models. *Math. Model. Comput. Exp.* **1** 407–414.

Suykens, J. A. K., J. Vandewalle. 1999. Least squares support vector machine classifiers. *Neural Processing Lett.* **9** 293–300.

Suykens, J. A. K., J. Vandewalle. 2000. Recurrent least squares support vector machines. *IEEE Trans. Circuits and Systems-I* **47** 1109–1114.

Suykens, J. A. K., J. De Brabanter, L. Lukas, J. Vandewalle. 2002a. Weighted least squares support vector machines: Robustness and sparse approximation. *Neurocomputing* **48** 85–105.

Suykens, J. A. K., T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle. 2002b. *Least Squares Support Vector Machines*. World Scientific, Singapore, http://www.esat.kuleuven.ac.be/sista/lssvmlab/.

Suykens, J. A. K., T. Van Gestel, J. Vandewalle, B. De Moor. 2002c. A support vector machine formulation to PCA analysis and its kernel version. Internal Report 02-68, ESAT-SISTA, K. U. Leuven, Leuven, Belgium.

Trafalis, T. B., H. Ince. 2000. Support vector machine for regression and applications to financial forecasting. *Internat. Joint Conf. Neural Network* (*IJCNN*) *2000*, Como, Italy, 348–353.

Van Gestel, T., J. A. K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. De Moor, J. Vandewalle. 2004. Benchmarking least squares support vector machine classifiers. *Machine Learn.* **54** 5–32.

Vapnik, V. 1995. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

Vapnik, V., S. Golowich, A. Smola. 1997. Support vector method for function approximation, regression estimation and signal processing. M. Mozer, M. Jordan, T. Petsche, eds. *Advances in Neural Information Processing Systems*, Vol. 9. MIT Press, Cambridge, MA.

Wang, S., W. Zhu, Z. Liang. 2001. Shape deformation: SVM regression and application to medical image segmentation. *Internat. Conf. Comput. Vision* (*ICCV'01*), Vol. 2. Vancouver, BC, Canada, 209–216.