

Generalized Sensitivity Analysis: A Framework for Evaluating Data Analysis Results

Ronald K. Pearson*

Abstract

This paper describes a general framework for both the evaluation of data analysis results and the detection of anomalies in datasets. The essential idea is that a *good* result should not be highly sensitive to “reasonable” changes in either the analysis method or the data on which it is based. The general approach described here is closely related to the ideas of robust statistics and to the notion of *exchangeability*, which leads to a useful definition of “reasonableness” in the context required here. Further, this approach is particularly well-suited to the analysis of large datasets that support various subset selection strategies. It is shown how this general idea may be used to generate specific graphical procedures for informally assessing the character of both datasets and data models.

Keywords:

heuristic, metaheuristic, exchangeability, robustness, data analysis, boxplots

1 Introduction

Data mining is primarily concerned with the analysis of large datasets, usually collected under relatively uncontrolled conditions (e.g., financial transactions or industrial process data from historical databases). Hence, the available dataset is likely to contain anomalous data values that can influence standard data analysis results, sometimes quite adversely. A closely related second problem is the choice of an effective analysis method, balancing two often conflicting objectives: insensitivity to the presence of data anomalies, and the ability to accurately characterize the nominal (i.e., non-anomalous) portion of the available data. This paper proposes a framework for systematically examining the influence of both dataset variations and method options on the results obtained in a wide variety of data mining problems, based on the following general observation:

A good data analysis result should not be sensitive to small changes in the methods or the datasets on which it is based.

This statement is quite similar to the definition of a *robust* data analysis procedure as one that exhibits “insensitivity to small deviations from the assumptions” on which the analysis is based [13, p. 1]. Further, Huber [13, p. 7] also notes that, for practical purposes, robustness is essentially equivalent to *resistance*, which refers to a data analysis procedure that is insensitive to small changes in the *dataset* on which it is based, interpreted to mean either small changes in all of the data values, or large changes in a few of the data values. The idea proposed here is similar in concept but different in detail, leading to a very general framework for constructing simple graphical evaluation procedures for essentially any type of data analysis.

More specifically, the *Generalized Sensitivity Analysis (GSA)* procedure proposed here consists of the following five steps:

1. Based on application-specific considerations, select $q \geq 1$ *scenarios* Σ_ℓ to be compared;
2. Specify a mutually consistent *sampling scheme* to be applied to all scenarios, ideally generating m datasets S_j per scenario, each of the same size n ;
3. Select a real-valued *descriptor* $d(\cdot)$, used to characterize the data analysis results considered;
4. Select a set $\{\mathcal{M}_i\}$ of analysis methods that are consistent with the descriptor $d(\cdot)$;
5. Construct a boxplot [12, p. 44] to compare the results obtained by each method \mathcal{M}_i for all of the subsets S_j generated under each scenario Σ_ℓ .

The primary objective of this paper is to show how this sequence of steps may be turned into detailed procedures for comparing “equivalent” analysis methods across “equivalent” datasets. Here, analysis methods \mathcal{M}_i and \mathcal{M}_j will be declared equivalent (more precisely, $d(\cdot)$ -equivalent) if there exists a common descriptor $d(\cdot)$ that can be meaningfully applied to the results obtained by both methods. Datasets will be declared equivalent if they are *exchangeable* in the data-analytic sense proposed by Draper *et al.* [7].

*Daniel Baugh Institute for Functional Genomics and Computational Biology, Thomas Jefferson University, Philadelphia, USA.

2 Example 1: CWMF tuning

Before presenting a more detailed discussion of the GSA framework, it is useful to first illustrate the basic idea with the following simple example. The problem of dynamic data characterization arises frequently in areas as diverse as industrial process monitoring, financial data analysis, and the characterization of biomedical data sequences. An important practical aspect of dynamic data characterization is that standard methods are often quite sensitive to the presence of outliers in these data sequences. This observation has motivated the development of various nonlinear *data cleaning filters*, intended to remove these outliers from the observed data sequences, thus improving the quality of the results obtained via standard dynamic analysis methods. A procedure that is sometimes extremely effective in these applications is the one proposed by Martin and Thomson for spectrum estimation [16]. Despite its effectiveness, the practical utility of this procedure is restricted somewhat by its complexity, leading to the consideration of simpler alternatives like the *median filter* [17]. In its standard implementation, this filter maps a finite-length input sequence into an output sequence of the same length, where the filter output $w(k)$ at time k is equal to the median of the values in the moving data window from time $k - K$ to time $k + K$:

$$(2.1) \quad w(k) = \text{median} \{x(k - K), \dots, x(k), \dots, x(k + K)\}.$$

(Because this filter depends on future data values, it is not suitable for real-time applications, but it is quite useful in off-line data analysis applications; also, note that filter responses near the beginning and end of the input data sequence require special handling. For further discussion, see the book by Astola and Kuosmanen [1].)

The essence of the data cleaning problem is illustrated in Fig. 1, which shows plots of four data sequences. The upper left plot shows the nominal data sequence we would like to characterize. Here, this sequence was generated by the linear time-series model:

$$(2.2) \quad x(k) = 0.8x(k - 1) + 0.6\epsilon(k),$$

where $\{\epsilon(k)\}$ is a sequence of independent, identically distributed (i.i.d.), zero-mean, Gaussian random variables with variance $\sigma^2 = 0.36$, chosen so that the resulting data sequence $\{x(k)\}$ has unit variance. The upper right plot in Fig. 1 shows the observed sequence $\{z(k)\}$, mostly equal to the nominal data sequence, but contaminated with 10 additive outliers, each of magnitude +12 and randomly spaced in time. The lower left plot shows the result $\{w(k)\}$ of applying a standard median filter with $K = 2$ to the observed data sequence

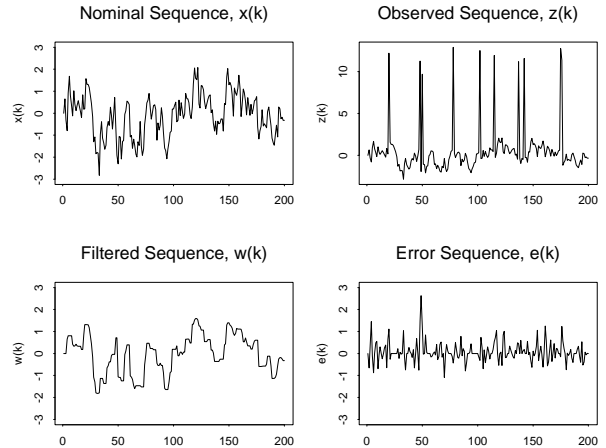


Figure 1: Nominal, observed, filtered, and error sequences

$\{z(k)\}$. Comparing the upper and lower left-hand plots, it is clear that, although the median filter has removed the outliers, it has also introduced significant distortion. To quantify this distortion, consider the error sequence $e(k) = w(k) - y(k)$ between the output of the median filter and the nominal data sequence we wish to recover. This sequence is shown in the lower right plot in Fig. 1.

The distortion of the nominal data sequence by the median filter seen in this example is typical. Further, because the median filter has only one tuning parameter (the window half-width K), our ability to overcome this problem is limited. One alternative that is somewhat more flexible is the *center weighted median filter (CWMF)* [25], whose output at time k is given by:

$$(2.3) \quad w(k) = \text{median} \{x(k - K), \dots, x(k - 1), W \diamond x(k), x(k + 1), \dots, x(k + K)\}.$$

Here, the symbol $W \diamond x(k)$ means that the central value, $x(k)$, is *repeated* W times in the indicated median. The center weight W is usually taken as an odd integer, with little loss of generality [25], and taking $W = 1$ reduces this filter to the standard median filter. If W is any odd integer greater than or equal to $2K + 1$, the central value $x(k)$ always dominates the median, reducing the filter to the identity $w(k) = x(k)$ for all input sequences $\{x(k)\}$. Hence, it is interesting to consider the performance of the center-weighted median filter for weights ranging from $W = 1$ (the standard median filter) to $W = 2K - 1$ (the largest non-trivial odd integer center weight).

Fig. 2 illustrates the use of the GSA procedure to assess the performance of the center-weighted median filter as a function of its two tuning parameters, K

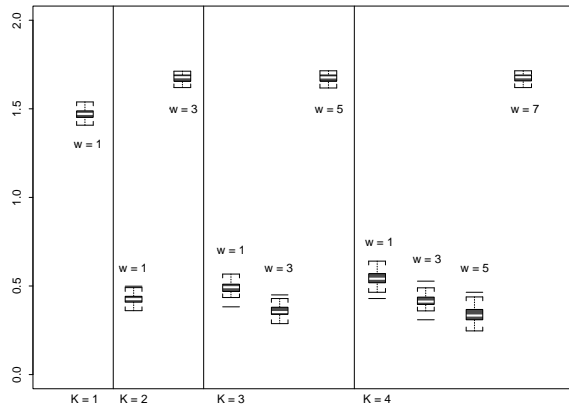


Figure 2: RMS data cleaning error vs. CWMF tuning parameters

and W . Here, the methods \mathcal{M}_i correspond to the four choices of window half-width parameter $K = 1, 2, 3,$ and 4 . Associated with each method are $q = K$ scenarios Σ_ℓ , corresponding to the K admissible odd integer center weights $W = 1, 3, \dots, 2K - 1$. Within each scenario, $m = 100$ datasets have been generated, as follows. First, 100 statistically independent zero-mean, i.i.d. Gaussian sequences with variance $\sigma^2 = 0.36$ and length $n = 200$ are generated and used to generate a nominal data sequence $\{y(k)\}$ according to Eq. (2.2). This sequence is then contaminated by adding 12 to $y(k)$ at the 10 points $k = 20, 48, 50, 78, 102, 115, 137, 142, 175,$ and 176 . The result defines the collection of m “equivalent” data sequences $\{S_j\}$ required in Step 2 of the GSA procedure.

The real-valued descriptor used to compare these analysis methods is the root-mean-square error between the nominal data sequence $\{y(k)\}$ that a perfect data cleaning filter would give us and the filtered data sequence $\{w(k)\}$ that we actually obtain:

$$(2.4) \quad d(\mathcal{M}_i S_j) = \sqrt{\frac{1}{n} \sum_{k=1}^n e(k)^2} = \sqrt{\frac{1}{n} \sum_{k=1}^n [w(k) - y(k)]^2}.$$

The final step of the GSA metaheuristic is the construction of a simple graphical display of these descriptor values. In its simplest form, a *boxplot* is a graphical summary of a dataset S of size N based on the following five numbers:

- the minimum value in S ;
- the *lower quartile* (i.e., the largest number less than or equal to 25% of the values in S);

- the *sample median*, equal to the middle value in S if N is odd, or the average of the two middle values if N is even;
- the *upper quartile* (i.e., the largest number less than or equal to 75% of the values in S);
- the maximum value in S .

Alternatively, another common boxplot construction replaces the sample minimum and maximum with the smallest and largest values not exceeding cutoff limits computed from the other three numbers, marking values that fall outside these limits as outliers [12, p. 44]. Because the identification of these outliers can be one of the key results of the GSA procedure, all of the boxplots presented here are based on this second construction.

The boxplots shown in Fig. 2 clearly demonstrate how the performance of the center-weighted median filter depends on the tuning parameters K and W . In particular, the reduction in RMS error is dramatic for the standard median filter ($W = 1$) in going from $K = 1$ to $K = 2$. For $K > 2$, we observe a slight degradation in performance, leading to the conclusion that $K = 2$ is the optimum window half-width for the standard median filter, at least for this example. Conversely, for $K > 2$, better results are achievable with intermediate values of the center weight parameter W . More specifically, the best results overall are obtained using $W = 2K - 3$ for all values $K \geq 2$. The key point is that the five-step GSA procedure described here provides a systematic basis for constructing graphical displays like the one shown in Fig. 2 that allow us to decide between different computational methods and method-specific options for a wide range of data analysis problems. Some possible extensions of the specific results presented here are discussed further in Sec. 4.

3 Exchangeability

The essential idea on which the GSA framework is based is that of comparing results obtained for different datasets that are “distinct but not essentially different.” In the center-weighted median filter tuning example just considered, these datasets represented statistically independent random sequences with the same general statistical character. To discuss the “equivalence” desired in Step 2 of the GSA procedure between the m datasets generated by the specified sampling scheme, it is useful to have a formal definition of “equivalent datasets.” As defined by Draper *et al.* [7], *exchangeability* is a judgement we make concerning distinct sources of data. Loosely speaking, two data sources (or datasets) are exchangeable if important characterizations of both give about the same results. To make this definition more

precise, Draper *et al.* introduce the following five mathematical components:

1. a set \mathcal{U} of *units*, which may be any sources of data;
2. a collection $\Omega = \{(S_i, S_j)\}$ of pairs of distinct datasets S_i and S_j , obtained from the units in \mathcal{U} ;
3. a *descriptor* $\delta(\cdot)$ mapping the sets S_i into R^n for some $n \geq 1$;
4. a *norm* $\|\cdot\|$ on the space R^n ;
5. a positive constant C called a *caliper*.

Under this definition, exchangeability is a judgement made regarding the units in \mathcal{U} , based on the pairs of datasets contained in the collection Ω . The authors call these pairs of datasets *comparisons* and it is important to note that, although the sets involved must be distinct (i.e., $S_i \neq S_j$), they need not be disjoint (i.e., $S_i \cap S_j \neq \emptyset$ is allowed). Taken together, the descriptor $\delta(\cdot)$ and the norm $\|\cdot\|$ permit us to make quantitative comparisons of the sets S_i and S_j . In particular, $\|\delta(S_i) - \delta(S_j)\|$ is a positive number, defined for all comparisons (S_i, S_j) in the collection Ω , that may be interpreted as a measure of distance or dissimilarity between these two datasets. Draper *et al.* define the units in \mathcal{U} to be exchangeable (or, more precisely, $(\Omega, \delta(\cdot), \|\cdot\|, C)$ -exchangeable) if the following condition holds:

$$\|\delta(S_i) - \delta(S_j)\| \leq C, \quad \text{for all } (S_i, S_j) \in \Omega,$$

where the constant $C > 0$ is the caliper defined in the above list. This constant represents a threshold parameter, quantifying the allowed mismatch between $\delta(S_i)$ and $\delta(S_j)$ for “equivalent” datasets S_i and S_j .

In the GSA framework proposed here, each scenario Σ_ℓ corresponds to a set \mathcal{U} of data generating units, and the nominal working assumption invoked in constructing specific implementations of the GSA framework is that these units are exchangeable. That is, our expectation is that, *in the absence of data anomalies*, the m datasets generated in Step 2 of the GSA procedure are exchangeable. As a practical matter, this exchangeability requirement reduces to something like an experimental design specification. The question of how we meet this requirement is discussed further in Sec. 4 and illustrated in the examples discussed subsequently. Here, the key point is that we regard exchangeable datasets S_i and S_j to be “small changes” of each other, and the objective of the GSA procedure is to assess whether these small changes cause correspondingly small changes in the results obtained using a specified analysis method \mathcal{M} . In particular, note that each of the boxplots constructed using the GSA procedure

may be regarded as an informal assessment of the range of variation of $\{d(\mathcal{M}_i S_j)\}$ for fixed i , over the subsets S_j generated by the sampling scheme specified in GSA Step 2. Indeed, the objective of Step 2 is to generate a collection Σ_ℓ of $\{\Omega, \delta(\cdot), \|\cdot\|, C\}$ -exchangeable subsets, and the objective of the GSA procedure as a whole is to determine whether each analysis method \mathcal{M}_i generates results $\mathcal{M}_i S_j$ that are exchangeable with respect to the (generally quite different) descriptor $d(\cdot)$ explicitly specified in GSA Step 3. By restricting consideration to scalar-valued descriptors $d(\cdot)$ and constructing graphical representations of the results, we eliminate the need for the norm and the caliper in this assessment of the exchangeability of the analysis results.

4 The GSA metaheuristic

The GSA framework proposed here represents a *metaheuristic*, which is a systematic procedure for generating *heuristics*, defined by Pearl [18, p. 3] as:

“...criteria, methods, or principles for deciding which among several alternative courses of action promises to be the most effective in order to achieve good results.”

Further, Pearl notes that heuristics represent compromises between the need to keep selection criteria simple and the need to keep them effective in distinguishing between good and bad alternatives. The five step GSA procedure described in this paper provides a systematic procedure for constructing graphical heuristics that compare the performance of different analysis methods across application-specific scenarios, helping us to choose between analysis methods and to assess the degree to which different scenarios appear to be similar or dissimilar. Hansen and Mladenovic [11] offer eight desirable properties of metaheuristics, beginning with the following three:

1. *simplicity*: metaheuristics should be based on a simple, broadly applicable principle;
2. *precision*: the steps of the metaheuristic should be formulated in precise, mathematical terms;
3. *coherence*: all steps for heuristics derived from the metaheuristic should follow naturally from its underlying principle.

The five-step GSA metaheuristic presented here exhibits all three of these desirable criteria: it is based on the informal definition of a good data analysis procedure offered at the beginning of this paper, each step is precisely specified, and all steps follow naturally from the basic notion of testing for exchangeability violations.

The following subsections consider various practical aspects of each of these steps.

4.1 Choosing scenarios The scenarios Σ_ℓ required in Step 1 of the GSA metaheuristic can be any systematic data generation procedure that is relevant to the application of interest. In general terms, a scenario defines a data source, and it corresponds essentially to the set \mathcal{U} of units on which the exchangeability definition of Draper *et al.* [7] is based. In practice, these scenarios should be chosen on the basis of field-specific knowledge, conjectures, hunches, or open questions. For example, a scenario might represent a collection of business data records corresponding to a specific department, store, geographic location, customer age range, or any other factor that might be of interest. The basic question to ask in specifying scenarios is, “what results would we like to compare?”

In the center-weighted median filter tuning example considered in Sec. 2, the scenarios Σ_ℓ were method-specific, representing the complete range of center-weight tuning parameters that were admissible for each method \mathcal{M}_i . Alternatively, another possibility would have been to consider different nominal data sequences and/or outlier types as scenarios; constructed this way, the results of the GSA procedure would give us insights into the range of performance variations to expect across different types of data. As a specific example, in dynamic data characterizations like spectrum estimation and linear system identification, not only is the concentration of outliers in the data sequences important, but so is their *pattern*, with “patchy outliers” that are all grouped together having different consequences than the same number of isolated outliers [20]. Since these different classes of outliers also respond differently to specific data cleaning filters, one potentially interesting choice of scenarios would be “patchy” and “isolated” outliers. The problem of choosing useful scenarios for generalized sensitivity analysis is illustrated further in Secs. 5 and 6.

4.2 Sampling schemes The sampling scheme required in Step 2 of the GSA metaheuristic may be any systematic procedure for generating data subsets S_j based on the scenarios Σ_ℓ specified in Step 1. As noted in Sec. 3, it is most desirable that this sampling scheme be specified so that the distinct subsets S_j and S_k it generates are exchangeable, ideally both within and across scenarios. The requirements that the size n of each dataset and the number m of datasets generated be the same across all scenarios is helpful in this regard, although these requirements are generally neither necessary nor sufficient by themselves to guarantee

exchangeability. Conversely, in the typical case where it is possible to use the same collection of datasets $\{S_j\}$ across the different scenarios and analysis methods considered, exchangeability across methods and scenarios follows immediately. Hence, the most important practical aspect of specifying a sampling scheme is usually that of guaranteeing that the datasets within this collection are exchangeable with respect to some meaningful descriptor $\delta(\cdot)$.

In simulation-based examples like those considered in Secs. 2 and 5, the sampling scheme specifies a systematic data generation procedure. In cases where this procedure is based on random number generation, the exchangeability requirement frequently reduces to one or more quality assumptions on the random number generators used to generate the datasets S_j . In particular, for the example considered in Sec. 2, exchangeability of the contaminated data sequences used to evaluate the effectiveness of the different CWMF tuning parameters follows from the assumption that the sequences $\{\epsilon(k)\}$ used in generating these sequences are accurate approximations of statistically independent, zero-mean i.i.d. Gaussian sequences with the specified variance. Similar conclusions hold for the simulation-based example described in Sec. 5.

In real data applications like that considered in Sec. 6, a scenario Σ_ℓ may represent a single large dataset S . In this case, it is reasonable to consider sampling schemes based on subset selection, regarding subsets of the same size as exchangeable *at least in the absence of anomalies or special structure*. The number of subsets of size s from a dataset of size r is given by:

$$(4.5) \quad \binom{r}{s} = \frac{r!}{s!(r-s)!},$$

and because this number grows so rapidly with increasing r and s , it is generally necessary to consider restricted selection strategies. For example, there are approximately 2×10^{40} ways of selecting subsets of size 50 from a dataset with 150 elements. Conversely, imposing restrictions on the form of an admissible subset can reduce this number dramatically. As a specific example, in dynamic characterizations like spectrum estimation and system identification, it is important to preserve the time order of the data samples, forcing us to consider *consecutive* data subsets. For a sequence of length r , the number of consecutive subsequences of length s is only $r - s + 1$.

Another important special case of subset selection forms the basis for *deletion diagnostics* [2, 3] and the *jackknife variance estimator* [8]. In both cases, the basic idea is to omit one data point at a time and characterize the influence of these omissions on our computed results.

For a dataset of size n , this process yields n datasets, each of size $n - 1$. Further, these datasets may usually be regarded as exchangeable if it is reasonable to regard the individual data points themselves as exchangeable. A useful extension of this idea is to delete more than one point at a time from the dataset, forming the basis for *multiple deletion diagnostics* [3] and various extensions of the classical leave-1-out jackknife [24]. Again, practical implementations of this idea are constrained by the combinatorial explosion of the number of possible subsets to consider: the number of subsets with s points omitted from a set of size r is also given by the expression in Eq. (4.5). As before, in cases like time-series analysis where it makes sense to restrict consideration to the deletion of s *consecutive* data observations, exhaustive comparison of all admissible subsets of fixed size remains practical even for large r and s [5].

Alternatively, random subsampling is often used in cases where the number of possible subsets is too large to permit complete characterization. This idea forms the basis for procedures like the *delete-p%-jackknife* [22, 24], where a large number of randomly selected subsets is considered, each having $p\%$ of the original observations deleted. As in the case of leave-1-out deletion diagnostics, the resulting subsets may usually be regarded as exchangeable if it is reasonable to regard the individual data points in the original dataset as exchangeable.

Finally, it should be noted that the ideas just described for developing GSA sampling schemes are closely related to two other important ideas in computational statistics. First, procedures like the jackknife mentioned above and the *bootstrap* [8] use samples drawn from a fixed dataset, either with replacement in the case of the bootstrap or without replacement in the case of the jackknife, to assess the natural variability of a computed result. Here, assumptions of exchangeability play a key role in supporting the interpretation of these results as “nominal variability,” to be expected in the absence of data anomalies. The other important idea related to GSA sampling schemes is that of explicitly testing for violations of exchangeability. Indeed, this is the objective of deletion diagnostics, since the deletion of anomalous (i.e., non-exchangeable) observations often causes significant changes in the computed results. A more formal version of this idea is embodied in *permutation tests*, which explicitly test a null hypothesis of exchangeability [9]. In the context of the GSA metaheuristic, the expectation is that within-scenario exchangeability violations will appear as outliers in the boxplots generated by the GSA procedure, calling our attention to specific data subsets S_j that appear to violate our exchangeability assumption. The extent to

which this expectation is met will depend on both the methods \mathcal{M}_i we consider and the descriptor $d(\cdot)$ from which these boxplots are constructed.

4.3 Choosing a descriptor The descriptor on which the exchangeability of the data subsets S_j generated in Step 2 of the GSA metaheuristic is based is typically not specified explicitly. Instead, this descriptor is usually implied by assumptions like, “datasets S_i and S_j have the same statistical character” imposed in the example discussed in Sec. 2. In contrast, the descriptor $d(\cdot)$ appearing in Step 3 of the GSA metaheuristic must be specified explicitly since it forms the basis for the boxplots constructed in Step 5 of this procedure. In fact, note that if we define $\delta_i(S_j) = d(\mathcal{M}_i S_j)$, we may view the GSA procedure as a graphical technique for assessing the exchangeability of the data subsets $\{S_j\}$ generated in Step 2 with respect to the descriptors $\delta_i(\cdot)$.

In the strict formulation described here, the GSA descriptor $d(\cdot)$ is required to be a map from the results generated by the analysis methods considered into the set of real numbers. This restriction has important practical consequences since it permits us to construct the boxplot generated in the final step of the GSA procedure. Although the general definition of exchangeability admits vector-valued descriptors, the use of such descriptors would require replacement of the boxplots considered here with something more complicated, like the biplots discussed by Gower and Hand [10].

An important class of scalar descriptors for the applications considered here are *figure-of-merit descriptors*, that have the following practical interpretation: if $d(\mathcal{M}_i S) > d(\mathcal{M}_j S)$, then the results obtained from dataset S by method \mathcal{M}_i are either clearly better than or clearly worse than those obtained by method \mathcal{M}_j . Examples of figure-of-merit descriptors include the RMS data cleaning error values discussed in Sec. 2, the prediction error variance discussed in Sec. 5 for any type of data prediction model, and the heterogeneity measures used in partition-based cluster analysis methods [14, ch. 3], all of which have the interpretation that “smaller is better.” It is not necessary to restrict consideration to figure-of-merit descriptors, but the use of other descriptors generally only allows us to assess *relative variability* across methods or scenarios, and not *relative quality*.

4.4 $d(\cdot)$ -equivalent methods As noted in the preliminary description of the GSA metaheuristic presented at the beginning of this paper, the analytical methods $\{\mathcal{M}_i\}$ specified in Step 4 of this procedure must be $d(\cdot)$ -equivalent with respect to the GSA descriptor chosen in Step 3. This requirement simply means that the value of $d(\mathcal{M}_i S_j)$ must be well-defined for all datasets S_j gen-

erated in Step 2 and all methods \mathcal{M}_i specified in Step 4. Clearly, if this condition is not met, it will not be possible to construct the boxplot summary required in Step 5 of the GSA procedure. Conversely, for this boxplot to be useful, the descriptor $d(\cdot)$ must provide a partial characterization of all of the different methods considered that is at least roughly comparable in its power as a summary statistic for the different methods considered.

This point is perhaps best illustrated with a simple example. Partition-based clustering methods attempt to classify a group of N objects into k distinct classes based on measured attributes of each object. Many different methods have been proposed to solve this problem [15, ch. 2], and it is natural to ask how the different methods compare. Various comparison measures have been proposed, including those like *cluster diameter* that characterize individual clusters, or *separation* that characterize the relationship between individual clusters. One measure that attempts to combine both of these characteristics is the *silhouette coefficient* [15, p. 85], defined for each object i as:

$$(4.6) \quad s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}},$$

where $a(i)$ is the average dissimilarity between object i and all other objects included in the same cluster, and $b(i)$ is average dissimilarity between object i and all objects of the closest neighboring cluster (i.e., the closest cluster of objects that does not contain object i). In a good clustering, we generally want small dissimilarities between objects within each cluster, and large dissimilarities between clusters. It follows from this observation that the silhouette coefficients for the objects in a good clustering should be as large as possible. Further, it can be shown that $s(i)$ is bounded between -1 and $+1$ for all objects, provided that there are at least two clusters and that each cluster contains at least two elements. Hence, the average of the individual silhouette coefficients $s(i)$ over all objects i constitutes a scalar figure-of-merit descriptor that is applicable to all such clusterings. In particular, note that this descriptor may be used to compare partition-based clusterings obtained by radically different methods, possibly involving different numbers of clusters.

5 Example 2: nonlinear dynamic models

The following example illustrates the application of the GSA metaheuristic to a problem of evaluating nonlinear dynamic models. This example also illustrates that $d(\cdot)$ -equivalent methods can be very different in character, underscoring the generality of the GSA framework.

The basic problem considered here is that of developing a “good” discrete-time approximation, of the

following general form:

$$(5.7) \quad y_k = f(y_{k-1}, \dots, y_{k-p}, u_k, \dots, u_{k-q}) + e_k,$$

to the simple continuous-time, nonlinear ordinary differential equation:

$$(5.8) \quad \frac{dy(t)}{dt} = -\alpha y(t) + \beta y(t)u(t) + \gamma u(t),$$

sampled uniformly every T time units; in this example, the parameters α , β , γ , and T are all assumed to have the value 1 for convenience. Motivation for this problem comes from the field of industrial process control, where discrete-time models of the form (5.7) are required to implement model-based control strategies like Nonlinear Model Predictive Control [21]. A more detailed discussion of this example is given in the book by Doyle, Pearson and Ogunnaike [6, Sec. 5.6.4].

For simple models like (5.8), one feasible strategy is to work directly with the equations, replacing derivatives with finite difference approximations. Here, if we assume $y_k = y(t_0 + kT)$ and $u_k = u(t_0 + kT)$, we obtain the following *Euler discretization*, designated Model E in subsequent discussions:

$$(5.9) \quad \begin{aligned} \frac{dy(t)}{dt} &\simeq \frac{y_k - y_{k-1}}{T} \\ \Rightarrow y_k &\simeq [1 - \alpha T]y_{k-1} \\ &\quad + \beta T y_{k-1} u_{k-1} + \gamma T u_{k-1} \end{aligned}$$

Alternatively, we can also develop approximate models empirically, generating a finite-length input sequence $\{u_k\}$, simulating the response of the continuous-time model (5.8) to this input sequence, and choosing model parameters that minimize some measure of goodness-of-fit between the response of the continuous-time model and the approximating empirical model. In the example considered here, this approach was used to fit approximate models of the following four forms:

$$\begin{aligned} \text{A: } y_k &= y_0 + ay_{k-1} + bu_{k-1} \\ \text{B: } y_k &= y_0 + ay_{k-1} + bu_{k-1} + cy_{k-1}u_{k-1}, \\ \text{H: } y_k &= y_0 + ay_{k-1} + bu_{k-1} + cu_{k-1}^2, \\ \text{Q: } y_k &= y_0 + ay_{k-1} + bu_{k-1} + cy_{k-1}^2. \end{aligned}$$

The abbreviations here designate the model structures compared, which are *affine* (A), *bilinear* (B), *Hammerstein* (H), and *quadratic NARX* (Q).

An important, inherent feature of nonlinear dynamic models is that answers to questions like, “which approximation is best?” can depend strongly on the range and general character of the input sequences considered. Hence, a useful choice of scenario Σ in evaluating these approximate models is *input sequence type*. In

the results presented here, all sequences $\{u_k\}$ belong to the general class of random step inputs [19, Sec. 8.4.3], generated as follows. First, a sequence $\{z_k\}$ of n independent, identically distributed random variables is generated, uniformly distributed over a specified range. An input sequence $\{u_k\}$ of length n is then generated by first setting $u_1 = z_1$ and then generating u_k for $k > 1$ according to the following random selection procedure:

$$(5.10) \quad u_k = \begin{cases} z_k & \text{with probability } p \\ u_{k-1} & \text{with probability } 1 - p. \end{cases}$$

The resulting input sequence $\{u_k\}$ consists of random steps, whose average duration is determined by the switching probability p , and whose instantaneous values are uniformly distributed over a specified range.

In the example considered here, this range is $[-0.5, 0.5]$ and four scenarios are considered, corresponding to $p = 1.00$, $p = 0.30$, $p = 0.10$, and $p = 0.05$. Note that for $p = 1.00$, we recover the uniformly distributed white noise sequence $\{z_k\}$ that changes value at every time step, and the number of transitions in $\{u_k\}$ decreases as p decreases. The sampling strategy considered here consists of generating $m = 100$ statistically independent sequences $\{u_k\}$, each of length $n = 200$. This strategy is implemented by generating 100 independent uniform random sequences $\{z_k\}$ and applying the random selection rule (5.10) to generate the input sequences $\{u_k\}$. Note that here, the exchangeability assumption on which the equivalence of these sequences is based corresponds to quality assumptions on the uniform random number generator used in generating the sequences $\{z_k\}$ and on the binary random number generator used to implement the selection procedure (5.10).

The descriptor $d(\cdot)$ considered in this example is the prediction error standard deviation for each model, computed as:

$$(5.11) \quad \sigma_p = \left(\frac{1}{n} \sum_{i=1}^n [y_k - \hat{y}_k]^2 \right)^{1/2},$$

where $\{y_k\}$ represents the simulated response of the continuous-time model (5.8) to the input sequence $\{u_k\}$, and $\{\hat{y}_k\}$ represents the response of one of the approximating models to this input sequence. Note that this descriptor is of figure-of-merit type, since better models are those for which \hat{y}_k more accurately predicts y_k , resulting in a smaller value of σ_p .

The models compared here are the Euler discretization (E) defined in Eq. (5.9), and the four empirical models A,B,H, and Q listed above. Note that these models are $d(\cdot)$ -equivalent since, given the predictions $\{\hat{y}_k\}$ generated by any of them, we may evaluate the descriptor $d(\cdot)$ according to Eq. (5.11). The importance

of this observation lies in the fact that these models are obtained by radically different methods. In particular, Model E is computed directly from the continuous-time model equation (5.8), independently of the datasets $\{u_k\}$ used in its evaluation. In contrast, the four other models cannot be obtained directly from the original model equation, but are identified from the response of this model to a fixed input sequence $\{u_k\}$.

In fact, two important aspects of these latter four models should be noted. First, whereas Model E represents a *single model*, to be evaluated with respect to each of 100 input sequences generated for each scenario Σ , the other four cases actually represent *sets* of models, one obtained for each input sequence. The second point is that the criteria used to obtain these models appears similar to the descriptor σ_p defined in Eq. (5.11), *but it is not the same*. Specifically, each of the models of type A, B, H, and Q considered here is obtained by minimizing the *one-step prediction error sum of squares*, or equivalently, the *one-step prediction error standard deviation*:

$$(5.12) \quad \tau_p(a, b, c) = \left(\frac{1}{n} \sum_{i=1}^n [y_k - \tilde{y}_k(a, b, c)]^2 \right)^{1/2},$$

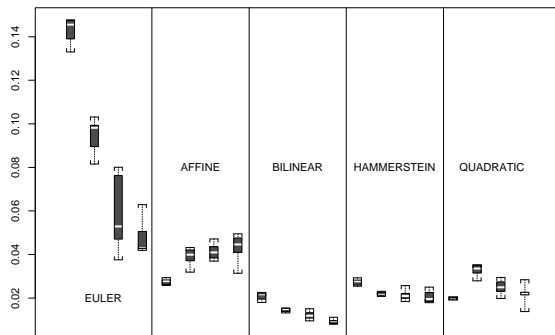
where $\tilde{y}_k(a, b, c)$ is the prediction of y_k from the previous input u_{k-1} *together with the previous output*, y_{k-1} . Given any input sequence $\{u_k\}$ and the corresponding sequence $\{y_k\}$ of simulated responses for the continuous-time model, we obtain the empirical model that best matches this input/output data by choosing parameters a , b , and c to minimize $\tau_p(a, b, c)$.

To illustrate the difference between \tilde{y}_k and \hat{y}_k , consider the bilinear model B:

$$(5.13) \quad \begin{aligned} \tilde{y}_k(a, b, c) &= ay_{k-1} + bu_{k-1} + cy_{k-1}u_{k-1} \\ \hat{y}_k &= a\hat{y}_{k-1} + bu_{k-1} + c\hat{y}_{k-1}u_{k-1}. \end{aligned}$$

The quantities σ_p and $\tau_p(a, b, c)$ can be significantly different, and in problems like model-based control where predictions beyond one time-step into the future are required [21], this difference can be important. The key point here is that, since σ_p is a practically important measure of model quality that, unlike $\tau_p(a, b, c)$, is not directly minimized in the empirical model fitting procedure, σ_p represents a reasonable quantity to evaluate with an approach like the GSA metaheuristic.

Fig. 3 shows the GSA results obtained for the five different approximations (Models E, A, B, H, and Q, respectively) to the observed responses of the continuous-time nonlinear dynamic model for four different input sequence scenarios defined by the switching probabilities $p = 1.00, 0.30, 0.10$, and 0.05 . Results for the different model structures are labelled and separated by

Figure 3: σ_p vs. model and switching probability

vertical lines, and the input scenarios appear in order of decreasing switching probability p . Because these boxplots represent summaries of a figure-of-merit descriptor, it is clear that the Euler discretization generally exhibits both the poorest performance in terms of σ_p values, and the least consistent performance: the range of variation seen in these values is significantly larger than that seen for the other four model types. Further, dependence of this performance on the input scenario (the switching probability p) is also quite dramatic, further supporting the general conclusion that Model E is a poor one, violating the informal statement of the GSA metaheuristic given at the beginning of this paper more severely than any of the other models.

In terms of consistency across input scenarios, the best models appear to be B and H, with Models A and Q exhibiting a wider range of variation than Models B and H, but much narrower than Model E. Also, note that the input sequence sensitivity appears to be slightly better for Model H than for Model B. However, because the descriptor considered here is of figure-of-merit type, we are led to prefer Model B, since it appears only slightly more sensitive than Model H, but exhibits consistently better (i.e., smaller) descriptor values. Similarly, comparing the performance of the five different models within a fixed input scenario generally also leads to the conclusion that Model E is significantly poorer than all of the others, that Models B and H are the best, and Models A and Q are of intermediate quality. As a specific case, note that we obtain the following model ranking within the scenario $p = 0.30$ (the second-left boxplot for each model):

$$B > H > Q > A \gg E.$$

In this particular example, this preference ordering generally holds across most input scenarios, based on

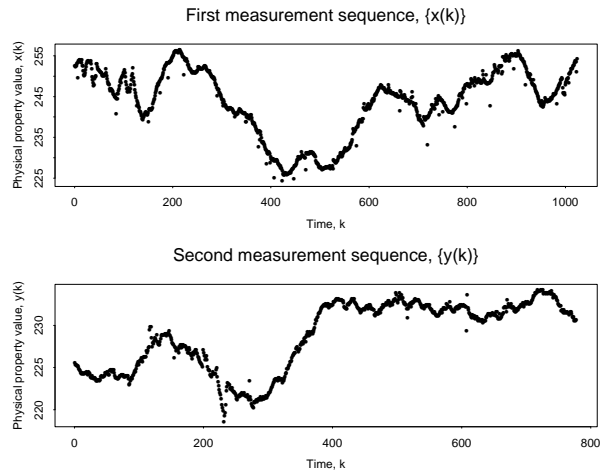


Figure 4: Two measurement data sequences

both the median values (a conclusion related to the fact that σ_p is a figure-of-merit descriptor), and on ranges of variability. Finally, comparing results across input scenarios for each model structure, these results also provide a nice illustration of the point made earlier, that the extent to which a particular nonlinear dynamic model approximates another one exhibits a significant input dependence, even for the best approximations.

6 Example 3: Process dynamics

This final example illustrates a typical use of the GSA metaheuristic in analyzing real data, and it is closely related to both of the other examples considered here. Fig. 4 shows two hourly sequences of physical property measurements made at the outlet of an intermediate product storage tank in an industrial manufacturing process. The upper plot shows a sequence $\{x(k)\}$ that has been discussed previously in connection with dynamic process characterization [19, ch. 8], and the lower plot shows the corresponding sequence $\{y(k)\}$ taken over a different time period, under different operating conditions. The example considered here uses the GSA metaheuristic to compare the effectiveness of different data cleaning strategies in characterizing and comparing the dynamics of these two data sequences. More specifically, these two data sequences will be taken as two scenarios and one of the questions of interest is whether these sequences exhibit similar or different dynamic characters.

A standard approach to time-series analysis is the construction of linear autoregressive models of the form:

$$(6.14) \quad \hat{x}(k) = \sum_{i=1}^p a_i \hat{x}(k-i) + \epsilon(k),$$

where $\{\epsilon(k)\}$ is a zero-mean, i.i.d. Gaussian sequence of unknown variance σ^2 which, along with the p unknown model coefficients $\{a_i\}$, are to be determined from the observed data sequence $\{x(k)\}$. Various algorithms have been developed for estimating these parameters [4], and the results presented here were obtained using Burg's method, one of the standard options available in the *S-plus* statistical analysis software package [23]. Regardless of the method selected, however, one of the practical challenges in fitting time-series models to observed data sequences is the choice of the model order p . Again, various methods have been proposed to address this problem, one of the most popular being to fit a range of models with p between 1 and some specified upper limit, p^* , selecting the model that minimizes the *Akaike Information Criterion (AIC)* [4]. This quantity balances goodness-of-fit, which improves uniformly with increasing model order p , with complexity, which becomes uniformly worse with increasing p .

Because the selection of model order is an important practical problem, this example takes the model order selected by the AIC as the descriptor $d(\cdot)$ in Step 3 of the GSA procedure. Also, because it is clear from both plots in Fig. 4 that the physical property sequences considered here contain outliers, it is of interest to compare the results obtained using the center-weighted median filter (CWMF) discussed in Sec. 2 as a data cleaning procedure. More specifically, the GSA procedure considered here takes the two data sequences shown in Fig. 4 as scenarios Σ_1 and Σ_2 and compares the performance of different CWMF tuning parameters with respect to consistency of the model order selected by optimizing the AIC. That is, the $d(\cdot)$ -equivalent methods considered here are the time-series models fit by applying one of the following data cleaning procedures to subsequences of the data sequences shown in Fig. 4:

1. no data cleaning filter is applied,
2. the CWMF is applied with $K = 1$ and $W = 1$,
3. the CWMF is applied with $K = 2$ and $W = 1$,
4. the CWMF is applied with $K = 3$ and $W = 1$,
5. the CWMF is applied with $K = 3$ and $W = 2$.

The primary question addressed by the analysis presented here are first, how effective does the AIC appear to be in selecting a model order for the two datasets considered here, second, how much are these results influenced by the different data cleaning strategies considered, and third, which data cleaning procedure appears to be the best?

Here, the sampling scheme required in Step 2 of the GSA procedure is based on the selection of consecutive subsequences of the two original data sequences shown in Fig. 4. In both cases, subsequences of length 100 are selected, each offset from the previous subsequence by 25 samples. For the data sequence $\{x(k)\}$ of length 1024, this procedure generates $m = 37$ subsequences, but because the data sequence $\{y(k)\}$ is only of length 779, this sampling scheme yields only $m = 28$ subsequences. Although this sampling scheme violates the stated objective of generating the same number of subsets S_j for both scenarios in Step 2 of the GSA meta-heuristic, these m values are comparable enough to provide a reasonable basis for comparison. Alternatively, we could truncate the first data sequence to the length of the second, forcing the number of scenarios to be equal in both cases. The subsequence lengths n were chosen to be long enough for the autoregressive modeling procedures considered here to yield meaningful results. Similarly, the offset value of 25 between subsequences was large enough to make the sequences somewhat different, but small enough to provide a reasonable number of subsequences for comparison. Clearly, both the sequence length and the offset between sequences are parameters that could be varied if we wished to explore their influence on the results obtained here. The key point is that the sampling scheme described here generates a useful collection of subsequences that are roughly comparable between scenarios and identical across methods.

The results of this particular GSA procedure are shown in Fig. 5, which presents boxplot comparisons of the AIC-selected model orders for both data sequences shown in Fig. 4 and for the five data cleaning options listed above. Results for the two datasets are grouped together, and those obtained for the different data cleaning options are separated by vertical lines. Each boxplot summarizes the model orders chosen for the m different subsequences of length 100 generated for each scenario and method option. Because the algorithm used to fit time-series models to these data sequences requires an upper limit p^* to be specified for the model order p , a common value of $p^* = 20$ was used throughout. This value corresponds to the default value $p^* = 10 \log_{10} n$ for the *S-plus* procedure used to identify these models, and it is indicated with the dashed horizontal line in Fig. 5.

Several points are immediately clear from the GSA comparison shown in Fig. 5. First and foremost, note that the model order selected by the AIC for this example is quite inconsistent, regardless of the data cleaning strategy chosen. For example, if no data cleaning filter is applied, the AIC procedure selects all possible model orders from $p = 1$ to $p = p^* = 20$ for the

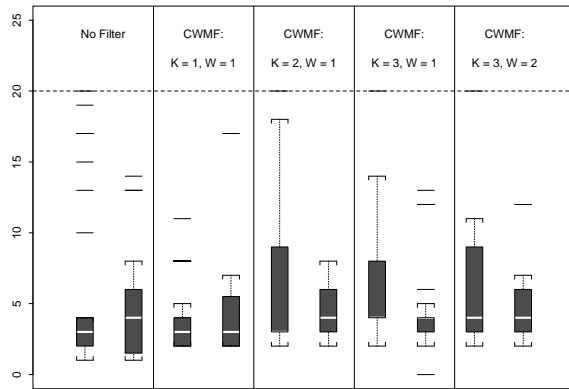


Figure 5: GSA comparison of AIC values across datasets and filtering strategies

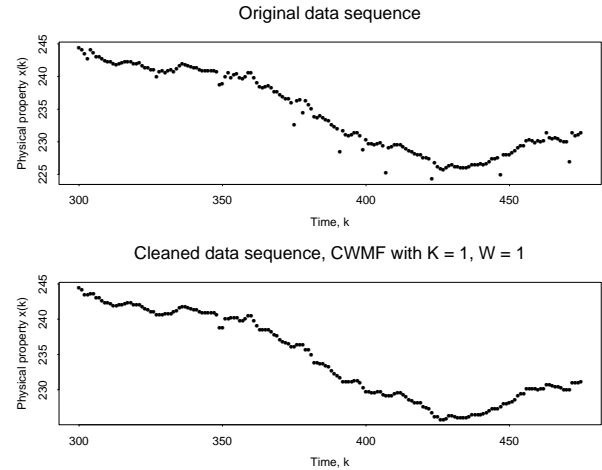


Figure 6: Original and cleaned data subsequences

first data sequence, although the majority (30 of 37) lie between 1 and 4. Similarly, for the second dataset, most values (25 of 28) lie between 1 and 8, while the other three selected model orders are 13, 13, and 14. Applying the standard median filter for $K = 1$ (i.e., the center-weighted median filter with $K = 1$ and $W = 1$) appears to improve the consistency of the AIC results somewhat. In particular, 73% of the estimated model orders are 2 or 3 for the first data sequence, whereas 81% of the values occupy the wider range from 1 to 4 when no data cleaning is applied. In contrast to the results presented in Sec. 2, increasing either K or W seems not to improve the consistency of the results obtained here. As a specific example, recall that among the standard median filters ($W = 1$), the choice $K = 2$ gave results that were unambiguously the best; in contrast, for the first data sequence considered here, this filter gives results that are significantly less consistent than those obtained for $K = 1$, and arguably worse than those obtained with no data cleaning at all.

Overall, the results presented in Fig. 5 suggest two general conclusions. First is the extreme sensitivity of the AIC model order selection procedure to what should be “small” changes in the dataset to which it is applied. Here, part of this difficulty arises from the presence of outliers in the first of the two data sequences considered here. This point may be seen clearly in Fig. 6, where the upper plot shows a subsequence of this original data sequence and the lower plot shows the results obtained using the CWMF data cleaner with $K = 1$ and $W = 1$. In particular, note the presence of the isolated points lying visibly off of the main curve seen in the upper plot and the absence of these points in the lower plot. This particular subsequence was chosen because it spans

four subsequences generated by the sampling scheme, for which the AIC model orders had the values 20, 19, 17, and 17. In contrast, the model orders chosen for the corresponding cleaned subsequences were 2, 8, 8, and 8, respectively.

The second primary conclusion from this example is that *none* of the data cleaning procedures considered here seem to dramatically improve the performance of the AIC model order selection procedure, for either of the two data sequences considered. This observation suggests one of two possible courses of action: either consider alternative data cleaning procedures, or consider alternative model order selection procedures. Both of these options are practical. Two alternative data cleaning procedures that could be applied are the Martin-Thomson data cleaning filter discussed briefly in Sec. 2 [16], and the *Hampel filter*, a moving window data cleaner like the center-weighted median filter that sometimes performs as well as the Martin-Thomson data cleaner [19, 20]. A number of alternatives to the AIC model order selection procedure also exist, such as the *Bayesian Information Criterion (BIC)* [4].

7 Summary

This paper has proposed a five-step metaheuristic called generalized sensitivity analysis (GSA) that may be used to generate graphical displays that are useful both in comparing the performance of alternative data analysis procedures and in detecting anomalies in datasets. Use of this procedure requires specification of one or more *scenarios* Σ_ℓ , which define specific data sources and a *sampling scheme*, which specifies how a collection $\{S_j\}$ of data subsets is to be generated for each scenario. The essential idea behind this procedure is the system-

atic comparison of computational results obtained from these datasets. To accomplish this comparison, a scalar-valued *descriptor* $d(\cdot)$ is specified, which should provide a useful partial summary of the computational results obtained using one or more specified methods, \mathcal{M}_i . Requiring that this descriptor be scalar-valued permits the construction of boxplots to compare results across scenarios and/or methods. If the datasets generated by the sampling procedure all conform to our expectations of “nominal” data (i.e., if the subsets S_j are all exchangeable or “non-anomalous”) and if our analysis methods are tolerant of “reasonable” data variations, the boxplots generated by the GSA procedure should provide a useful assessment of the nominal variation to be expected for the analysis results. Conversely, since the GSA results provide simultaneous comparisons across a range of conditions, they can reveal unexpected differences between scenarios that are believed to be similar, or between methods that are believed to be comparable. Also, the GSA procedure can lead to the identification of data subsets S_j that violate exchangeability assumptions, possibly leading to the discovery of significant data anomalies. The AIC example discussed in Sec. 6 illustrates this point.

As the examples presented here have attempted to demonstrate, this approach is applicable to an extremely wide range of data analysis procedures, including cluster analysis, dynamic characterizations of time-series, nonlinear dynamic model building, and many other possibilities. In addition, the GSA procedure is also applicable to related method-development problems like the selection and tuning of data cleaning filters to be used as part of a subsequent analysis. Further, because the procedure described here is completely data-based, it may be used to assess the performance of commercial “black-box” computational procedures based on proprietary software that is not clearly described in the available documentation.

References

- [1] J. Astola and P. Kuosmanen, *Fundamentals of Nonlinear Digital Filtering*, CRC Press, 1997.
- [2] A.C. Atkinson, *Plots, Transformations, and Regression*, Oxford, 1985.
- [3] D.A. Belsley, E. Kuh, and R.E. Welsch, *Regression Diagnostics*, Wiley, New York, 1980.
- [4] P.J. Brockwell and R.A. Davis, *Time Series: Theory and Methods*, Springer-Verlag, New York, 1991.
- [5] A.G. Bruce and R.D. Martin, *Leave-k-out diagnostics for time series*, J. Royal Statist. Soc., Ser. A, 51 (1989), pp. 363–424.
- [6] F.J. Doyle III, R.K. Pearson, and B.A. Ogunnaike, *Identification and Control Using Volterra Models*, Springer-Verlag, New York, 2002.
- [7] D. Draper, J.S. Hodges, C.L. Mallows, and D. Pregibon, *Exchangeability and data analysis*, J. Royal Statist. Soc., Ser. A, 159 (1993), pp. 9–37.
- [8] B. Efron and R.J. Tibshirani, *An Introduction to the Bootstrap*, Chapman and Hall, 1993.
- [9] P. Good, *Permutation Tests*, Springer, 1994.
- [10] J.C. Gower and D.J. Hand, *Biplots*, Chapman and Hall, New York, 1996.
- [11] P. Hansen and N. Mladenovic, *Variable Neighborhood Search*, in State-of-the-art Handbook of Metaheuristics, Kluwer, Boston, in press.
- [12] D.C. Hoaglin, F. Mosteller, and J.W. Tukey, *Fundamentals of Exploratory Analysis of Variance*, Wiley, New York, 1991.
- [13] P.J. Huber, *Robust Statistics*, Wiley, New York, 1981.
- [14] L. Hubert, P. Arabie, and J. Meulman, *Combinatorial Data Analysis: Optimization by Dynamic Programming*, SIAM, Philadelphia, 2001.
- [15] L. Kaufman and P.J. Rousseeuw, *Finding Groups in Data*, Wiley, New York, 1990.
- [16] R.D. Martin and D.J. Thomson, *Robust-resistant spectrum estimation*, Proc. IEEE, 70 (1982), pp. 1097–1114.
- [17] T.A. Nodes and N.C. Gallagher, *Median Filters: Some Modifications and Their Properties*, IEEE Trans. Acoustics, Speech, Signal Proc., 30 (1982), pp. 739–746.
- [18] J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley, Reading, MA, 1984.
- [19] R.K. Pearson, *Discrete-Time Dynamic Models*, Oxford, New York, 1999.
- [20] R.K. Pearson, *Outliers in Process Modeling and Identification*, IEEE Trans. Control Systems Tech., 10 (2001), pp. 55–63.
- [21] S.J. Qin and T.A. Badgewell, *An overview of nonlinear model predictive control applications*, in Proc. Internat. Symposium Nonlinear Model Predictive Control: Assessment and Future Directions, Ascona, Switzerland, (1998), pp. 128–145.
- [22] J. Shao and C.F.J. Wu, *A general theory for jackknife variance estimation*, Ann. Statist., 17 (1989), pp. 1176–1197.
- [23] *S-Plus 6.0 for UNIX, Guide to Statistics*, vol. 2, MathSoft, Inc., Seattle, 2000.
- [24] C. Weihs, *Cannonical Discriminant Analysis: Comparison of Resampling Methods and Convex-hull Approximation*, ch. 3 in *Recent Advances in Descriptive and Multivariate Analysis*, W.J. Krzanowski, ed., Oxford, 1995.
- [25] L. Yin, R. Yang, M. Gabbouj, and Y. Neuvo, *Weighted Median Filters: A Tutorial*, IEEE Trans. Circuits Syst.–II: Analog Digital Signal Proc., 43 (1996), pp. 157–192.