Componential coding in the condition monitoring of electrical machines
Part 1: principles and illustrations using simulated typical faults

C J S Webber[1]*, **B S Payne**[2], **F Gu**[2] and **A D Ball**[2]
[1]QinetiQ, Malvern, UK
[2]Maintenance Engineering Research Group, Manchester School of Engineering, University of Manchester, UK

Abstract: This paper (Part 1) describes the principles of a novel unsupervised adaptive neural network anomaly detection technique, called componential coding, in the context of condition monitoring of electrical machines. Numerical examples are given to illustrate the technique's capabilities. The companion paper (Part 2), which follows, assesses componential coding in its application to real data recorded from a known machine and an entirely unseen machine (a conventional induction motor and a novel transverse flux motor respectively). Componential coding is particularly suited to applications in which no machine-specific tailored techniques have been developed or in which no previous monitoring experience is available. This is because componential coding is an unsupervised technique that derives the features of the data during training, and so requires neither labelling of known faults nor pre-processing to enhance known fault characteristics. Componential coding offers advantages over more familiar unsupervised data processing techniques such as principal component analysis. In addition, componential coding may be implemented in a computationally efficient manner by exploiting the periodic convolution theorem. Periodic convolution also gives the algorithm the advantage of time invariance; i.e. it will work equally well even if the input data signal is offset by arbitrary displacements in time. This means that there is no need to synchronize the input data signal with respect to reference points or to determine the absolute angular position of a rotating part.

Keywords: neural network, componential coding, auto-encoder, condition monitoring

NOTATION

$\boldsymbol{a}$	anomaly vector $\boldsymbol{x} - \tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$
ADI	average discrimination index
b_c	cth basis vector scale parameter
E	mean square reconstruction error
f_0	principal frequency
$\mathbf{F}(\,)$	forward Fourier transform in the time domain
$\mathbf{F}^{-1}(\,)$	inverse Fourier transform in the time domain
ICAN	Independent Channel Architecture Network
JCAN	Joint Channel Architecture Network
k	integer index
M	monitored data-set
n_c	number of basis vectors
n_s	number of measurement sensors
n_t	number of time-samples
$N(t)$	random noise distributed uniformly between -0.1 and 0.1
$r(\xi)$	neuron output response
U	unseen control data-set
V	variance of $\lvert\boldsymbol{a}\rvert^2$ over a given data-set
VDI	variance discrimination index
W	training data-set
$\boldsymbol{w}_c$	cth basis vector
x	element of neural network input vector
$\boldsymbol{x}$	neural network input vector
$\boldsymbol{y}$	vector of neural output values
ε	random value (used specifically in the definition of synthetic data-sets)

The MS was received on 19 November 2002 and was accepted after revision for publication on 22 May 2003.
* *Corresponding author: QinetiQ, Malvern Technology Centre, Malvern WR14 3PS, UK.*

θ	angular phase angle
$\Delta\theta$	change in phase angle
ϑ	threshold value
λ	learning rate
ξ	projection $\boldsymbol{x} \cdot \boldsymbol{w}$
σ	softness definition of the threshold function

1 INTRODUCTION

Great efforts have been made in recent years to develop accurate and reliable methods for real-time plant condition monitoring. Since the processing of condition monitoring data is the fundamental issue for useful data representation and hence successful fault detection and diagnosis, most efforts have concentrated on the application of advanced data processing techniques. A brief overview of some of the main techniques is discussed below.

Many time–frequency data analyses have been investigated for machine condition monitoring. For example, Loughlin and Bernard [**1**] used a Cohen–Posch distribution, Gu *et al.* [**2**] used a Choi–Williams distribution and Gu *et al.* [**3**] used a Wigner–Ville distribution. In addition, timescale data processing has also been widely employed. Wang and McFadden [**4**] presents the use of Daubechies 4 and 20 orthogonal wavelets and Ball *et al.* [**5**] and Lin [**6**] demonstrate the capability of using Molet continuous wavelets. The work described by the citations demonstrates that these techniques enable the detection of incipient faults. However, due to data representation in high-dimensional space, these techniques are computationally demanding and it is difficult to identify data features for automated and online condition monitoring from this representation [**2**].

Higher-order statistics is a relatively new tool in the area of data processing. This method has been used by Howard [**7**] and Arthur and Penman [**8**] to identify non-linear phase modulation caused by turbo-pump, rolling element bearing and electric motor faults. In addition to the high computational overhead of these methods, they have been shown to be effective only for a limited range of faults.

As a non-linear adaptive data processing tool, independent component analysis has been used, by Li *et al.* [**9**] for example, to extract features from data with a high noise content. However, many components are extracted from the raw data using an independent component analysis and at least some of these components are often difficult to interpret physically.

Artificial neural networks have also been widely studied for fault detection and diagnosis purposes. They are often applied as a post-processing tool, with the input variables having been extracted by more conventional data processing techniques. For example, Murray and Penman [**10**] used data features extracted from higher-order statistics as input variables to a neural network. Zhang *et al.* [**11**] used features from time and frequency domain analyses. In addition to being used as a post-processing tool, artificial neural networks have been used by Gu *et al.* [**12**] for combustion process modelling using raw data as the input.

In general, capable and robust industrial applications are few and far between. Therefore, the development of advanced techniques for condition monitoring data processing is being increasingly addressed by both scientists and engineers.

Among the many different techniques, neural networks can be one of the most powerful tools for condition monitoring data processing. It has been shown [**13**] with many applications, including audio, video, speech, image, communication, geophysical, sonar, radar, medical, musical and others, that neural networks have a number of features particularly useful in processing condition monitoring data. Firstly, neural networks are capable of asynchronous parallel and distributed processing, thus allowing fast processing of a large amount of data, which could be from multiple sensors, and enabling online application. Secondly, the non-linear dynamics of neural networks allows non-linear machines and data to be modelled. Thirdly, self-organization of useful basis feature sets by neural networks enables automatic feature extraction. With these capabilities, neural networks can provide a primary foundation for solving many problems encountered in condition monitoring.

Based upon the fundamentals of neural networks, a novel unsupervised adaptive neural network, called componential coding, has been addressed in this study, demonstrating some of the above capabilities. A number of tools developed for fault detection and diagnosis are also presented. The capabilities and performance of componential coding in detecting machine faults and anomalies are demonstrated in Part 2 of this paper [**14**], by applying it to both a conventional induction motor and a novel transverse flux motor.

1.1 Definition of the engineering problem

This section defines the engineering problem that the componential coding technique addresses. As will be explained later in section 1, componential coding is a means of capturing the characteristics of a training data-set and thus subsequently determining how far the characteristics of any new, unseen data-set differ from those of the original training set. ('Unseen' means any data-set that was not used for training.) In neural network language, this defines the functionality called

anomaly detection; an anomaly is any characteristic of an unseen data-set that is different from the characteristics of the training data-set. Thus, any unseen data-set that has characteristics that are different from those of the training-set is said to be anomalous and any data-set that has characteristics that are indistinguishable from those of the training set is said to be non-anomalous.

Implicit in this definition of anomaly is that the anomalous characteristic is not a trivial characteristic such as the date on which it was recorded or the duration of the recording, but is some statistical property implicit in the sensor data that might in principle be useful for inferring real change in the physical properties of the system from which the data were recorded. Throughout these papers it is assumed that the componential coding algorithm is always trained on training data recorded from a machine under a 'healthy' operating condition, i.e. when no fault is present. One example of an anomalous data characteristic is a change in sensor data that results from operating a machine under a different operating condition from that which prevailed during training. Another example, of much greater interest for the maintenance engineer, is the change in sensor data that results from the onset of an incipient fault in the machine. For the purposes of these papers, therefore, 'fault detection' can be thought of as a special case of 'anomaly detection', in which the anomalous characteristic happens to arise as the result of a real fault in the engineering system being monitored. Thus, componential coding may be used in its capacity as an anomaly detection algorithm for the engineering application of fault detection, by providing an indication of how far the statistical characteristics of any new, unseen and potentially faulty data-set differ from the statistical characteristics of the healthy data-set used for training.

As will be explained later (section 2.7), the measure of how far the characteristics of an unseen data-set differ from those of the healthy training data-set is provided by a quantity called the discrimination index, which is effectively a measure of the degree of anomaly. In principle and often in practice, the discrimination index can therefore be used to calibrate the severity of a fault as well as to indicate its presence.

Having detected the presence of an anomaly or a real machine fault from a new, previously unseen data-set, the type of anomaly may be more specifically quantified (based on its occurrence and repeatability within the data-set for example) or the faulted system within the machine may be identified (for example the source of a gearbox fault may be attributed to a single cracked tooth). This process is referred to as diagnosis of the anomaly or fault.

The primary engineering applications of componential coding are: firstly, fault detection; secondly, discrimination of fault severity; and, thirdly, fault diagnosis. These three aspects of condition monitoring using componential coding are demonstrated in the Part 2 paper [**14**].

Componential coding is most relevant to applications where the input data signals take the form of digitized waveforms in the time domain, and are at least approximately periodic. One example of such periodic waveforms is vibration data recorded from a gearbox, where there is strong periodicity associated with the tooth-to-tooth mesh frequency and the fundamental rotation frequencies of each of the gears. Another example is the output of a magnetic flux sensor instrumented on a motor. This second example is addressed in practice and in depth in Part 2 of this paper. Periodic waveforms are extremely common from sensors used to instrument all kinds of rotating machinery; section 3 of the present paper works through specific numerical examples that illustrate a range of characteristics of typical faults that arise in rotating machinery, and the reader is referred to that section for more detailed specifics.

1.2 Overview of the process for applying the componential coding technique

This section provides an overview of the process by which componential coding is applied to the problems of detecting faults and discriminating their severity, for the example of a gearbox instrumented with one or more accelerometers to measure vibration. This process is valid for many other maintenance engineering applications, and a second example could equally well be provided by replacing the word 'gearbox' by 'induction motor' and the word 'accelerometer' by 'magnetic flux sensor'. The process is illustrated in the process diagram of Fig. 1.

In the simplest example of the process, each input data signal consists of a contiguous time-sequence of n_t digitized vibration-displacement samples recorded from a single accelerometer. Alternatively, if the gearbox is instrumented with a number n_s of accelerometers, each input data signal will consist of n_s such n_t-sample sequences recorded simultaneously in parallel (i.e. several 'channels'), one from each accelerometer. In neural network terminology, the input data signal is called the neural network's 'input vector' $\boldsymbol{x}$, or 'data vector' $\boldsymbol{x}$. The number of elements in each data vector $\boldsymbol{x}$ is given by the product $n_s n_t$. A 'data-set' $\{\boldsymbol{x}\}$, such as the training data-set for example, may consist of many such multichannel, multisample input signals/vectors, which may be recorded on different occasions but ideally under the same operating condition of the gearbox. Typically, a data-set may be obtained by dividing up one or a few long, contiguous, multichannel recordings into many shorter pieces, with each such piece constituting a vector $\boldsymbol{x}$ of the set.

The first phase of the process of using componential coding is to benchmark a data-set or data-sets known to be 'healthy', i.e. free of faults. This is done by recording

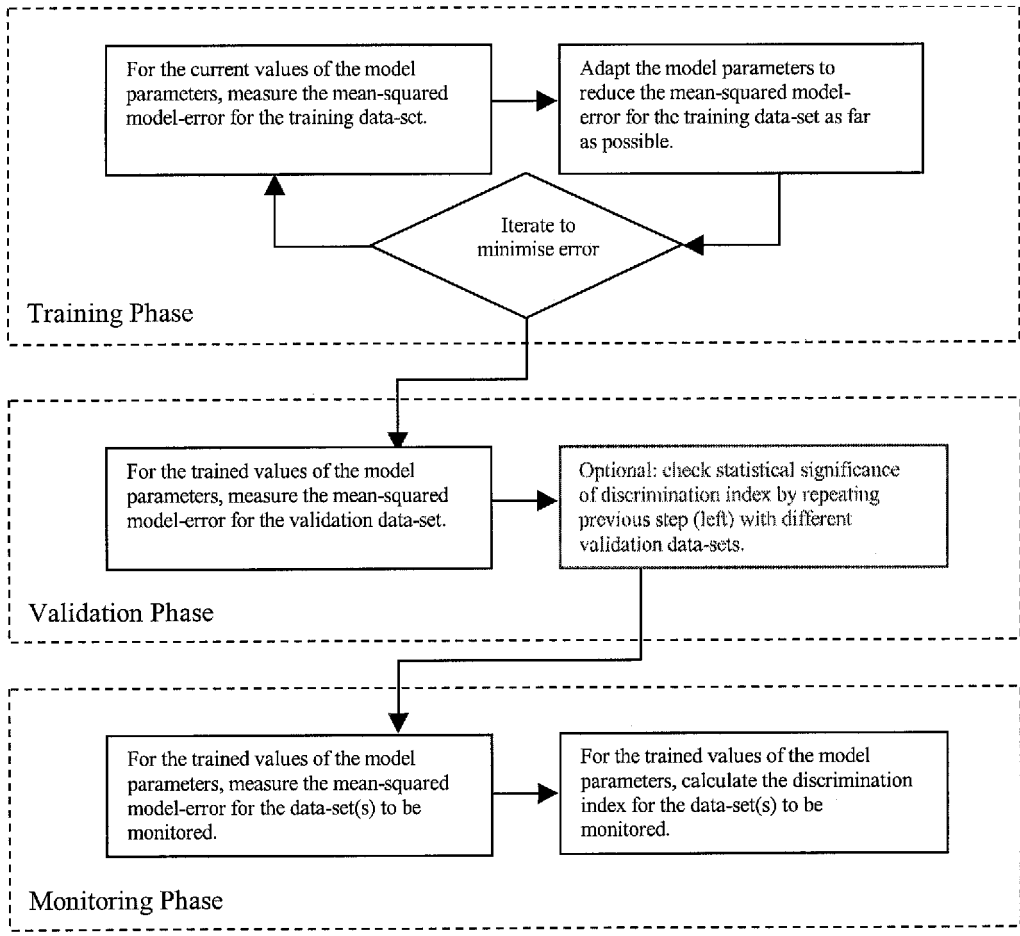

Fig. 1 Process diagram for application of componential coding in condition monitoring

from a known configuration of sensors that instrument the gearbox under a known operating condition or under several different known operating conditions, at a time when the gearbox is known to be 'healthy' (free of faults), and then by training the componential coding algorithm to model each of those healthy data-sets used to characterize each healthy operating condition.

Training the componential coding algorithm to model a training data-set is done by minimizing the mean-squared error, averaged over that training data-set, by which the data-set differs from the model. A small mean-squared error indicates that the model is well matched to the data-set, so the error-minimization training procedure results in a model that is as well matched to the training data-set as possible. How the model and the mismatch error are calculated is clarified in overview in section 1.3 and defined in full detail in section 2.

Following the training phase, a validation phase is performed to determine how closely the trained model matches other, unseen, healthy data. This is done by recording a new healthy data-set or data-sets, from the same configuration of sensors and under the same operating condition(s) as prevailed during training, and then measuring the mean-squared error by which the new healthy data-set(s) differ from the model obtained during the earlier training phase. Thus, the mean-squared error for the healthy validation data-set for each particular operating condition provides a measure of the natural intrinsic variability of the healthy machine under that operating condition. The purpose of this validation phase is solely to obtain the mean-squared error for the healthy validation data-set(s); no training/minimization procedure is involved in the validation phase. The validation data-set(s) must not include the same recordings as were used for training, otherwise the validation phase might underestimate the true intrinsic variability of the healthy machine by failing to measure directly the ability of componential coding to model healthy but unseen data-sets.

The training and validation phases together constitute a calibration procedure, designed to calibrate the componential coding algorithm to the properties of the sensor data from the healthy gearbox under each operating condition of interest. This calibration procedure may either be done once, perhaps when new plant is first instrumented, or alternatively may be repeated regularly to prevent the calibration becoming out-of-date if the general condition of the plant drifts slowly over time. If the former strategy is adopted, componential coding could be used to detect long-term drifts

by comparing against the initial one-off calibration; if the latter strategy is adopted, componential coding would be better able to detect shorter-term changes in the gearbox by comparing against a more recent calibration. In either case, the calibration is performed only when the gearbox is known or assumed to be free of faults.

A final 'monitoring' phase involves recording a new data-set or data-sets from the same configuration of sensors and under the same operating condition(s) as prevailed during training and then measuring the mean-squared error by which each such new data-set differs from the model obtained during the earlier training phase. By comparing the mean-squared error for these new 'monitored' data-set(s) with the mean-squared error for the healthy validation data-set (measured during the earlier calibration of the corresponding operating condition), it is possible to infer how much further each new monitored data-set differs from the trained model (of the appropriate operating condition) than does the healthy validation data-set. The presence of anomalies is inferred from the *average discrimination index*, which is defined as the ratio of the mean-squared error for the monitored data-set to the mean-squared error for the validation data-set, minus one. If the discrimination index is significantly greater than zero (a significant fraction of 1 or greater), this means that the trained model is a significantly poorer match to a new monitored data-set than it is to the healthy validation data-set used to calibrate that operating condition. This is clearly an indication that a physical change may have occurred in the gearbox, which has been responsible for greater variation in the sensor data than results from the natural intrinsic variability of the healthy machine under that particular operating condition. In other words, a potential fault has been detected in the monitored data-set when the discrimination index is a significant fraction of 1 or greater.

Throughout, it is assumed that sufficient training, validation and monitored data are available and that this discrimination index is statistically significant; i.e. that a large discrimination index is not merely the result of a statistical fluctuation due to insufficient data. In practice, very little data are required to render the discrimination index statistically significant, compared with how much sensor data are usually available, so the theoretical issue of statistical significance does not arise in practice and will not be treated here. In practice it is always simple to calibrate how big the discrimination index needs to be in order to have confidence that the anomaly detection is statistically significant. This is done by performing the steps of the monitoring phase several times on unseen monitored data-sets that are known to be healthy and checking how far the discrimination index rises above zero for those healthy data-sets. If the discrimination index is always less than (say) 0.01 for healthy data, then it is evident that a value much greater than 0.01 provides a statistically significant indication of an anomaly. This check need only be done once (per operating condition), as a last step in the calibration phase.

The magnitude of the discrimination index can be used to calibrate the severity of faults as well as detect them, because the more severe a fault, the further the monitored faulty data-set can differ from the trained healthy model.

1.3 Overview of how componential coding works

This section provides a non-mathematical overview of how componential coding measures the mean-squared error by which the characteristics of any given data-set differ from those of the trained model. This section also introduces and defines a range of neural network terminology needed to support the mathematical detail introduced later in section 2.

The componential coding neural network is an auto-encoder, which may be defined as an unsupervised neural network that models the input data signal from the sensors in such a way that the network is able to reconstruct a model-based replica of any given input data signal (Fig. 2). The componential coding training algorithm is designed to optimize the accuracy with which the model reconstructs the input data signal on average, i.e. to minimize the mean-squared error (averaged over the training data-set) by which that model-based replica differs from the actual input data signal.

The reason why componential coding is capable of detecting anomalous characteristics is that it is designed not to be able to reconstruct all data-sets as accurately as it can reconstruct data-sets that have similar characteristics to the training data-set, which by definition it has been trained to be able to reconstruct optimally accurately. Constraining the reconstruction to be less than perfectly accurate gives componential coding an ability to differentiate between data-sets having different statistical characteristics, by measuring just how inaccurately each data-set becomes reconstructed. Differentiating a 'monitored' data-set in this way from a healthy, validation data-set is the basis of the algorithm's fault detection capability, and differentiating different faulty data-sets from one another is the basis of its fault severity discrimination capability.

An auto-encoder can be viewed as a lossy data compression/reconstruction algorithm, which transforms the input data into a coded form and back again (Fig. 2). A conventional lossy data compression algorithm is designed to retain the maximum possible information about the input data when certain advantageous constraints are placed on the code, such as the constraint that the code should minimize file size or communication bandwidth. The advantageous constraints built into the

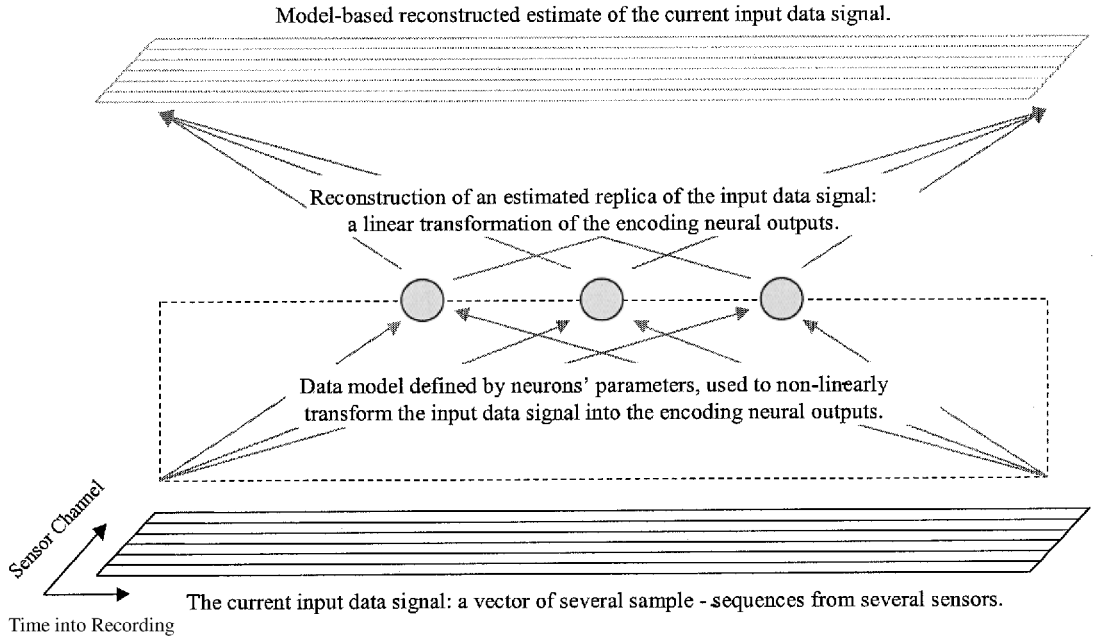

Fig. 2 The componential coding auto-encoder neural network

componential coding algorithm are different from size or bandwidth constraints, however. Instead, componential coding is designed to retain the maximum possible information about the input data subject to constraints designed to encourage the formation of codes having the property of 'sparseness', which will be defined in the next section. These are the same constraints that prevent componential coding from reconstructing all input signals/vectors with perfect accuracy, and thus give it the ability to distinguish between data-sets having different characteristics by measuring different values of the reconstruction error for different data-sets. The theoretical motivation for the particular constraints employed is summarized in the next section.

The componential coding neural network consists of a single layer of 'neurons', each of which receives the same input vector $\boldsymbol{x}$. As with all neural networks, each neuron of the network calculates a different 'neural output' value y, which is a function both of the current input vector $\boldsymbol{x}$ and of a number of 'neural parameters' whose values differ from neuron to neuron. These parameters are defined below and in section 2; the values of most of them are derived during the training phase. The set of neurons therefore serves to encode each input signal/vector $\boldsymbol{x}$ in terms of a new set of 'encoding coordinates' $\boldsymbol{y} = (y_1, y_2, \ldots)$, which are the values of the set of all the neurons' output values. As with most neural networks, this transformation is non-linear, because the neurons' output values $(y_1, y_2, \ldots)$ are all non-linear functions of the input $\boldsymbol{x}$. This non-linear transformation of one set of coordinates $\boldsymbol{x}$ into another set of coordinates $\boldsymbol{y}$ is analogous to the way Fourier analysis linearly transforms an input vector into a set of Fourier coefficients or the way principal component analysis linearly transforms an input vector into a set of coefficients that are the projections on to the basis set of the eigenvectors of the covariance matrix. (The standard terminology of linear algebra is used, for which the reader is referred to undergraduate engineering mathematics and computing texts, e.g. reference [**15**], [**16**] or [**17**].)

The auto-encoder also involves a linear inverse transformation $\boldsymbol{y} \rightarrow \tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$, which reconstructs the model-based replica $\tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$ of the current input from the output $\boldsymbol{y}$ encoding (see Fig. 2). As stated above, because of constraints implicit in the neural output functions $y(\boldsymbol{x})$, the transformation $\boldsymbol{x} \rightarrow \boldsymbol{y}$ is lossy; i.e. not all the information present in $\boldsymbol{x}$ is available in $\boldsymbol{y}$. Therefore, the reconstruction $\tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$ will not be exactly equal to the true input signal $\boldsymbol{x}$, but will differ from it by an error $|\boldsymbol{x} - \tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})|^2$.

The 'data model' is defined completely by the set of values of all the neural parameters implicit in the set of neural output functions $y(\boldsymbol{x})$, and thus the inaccuracy $|\boldsymbol{x} - \tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})|^2$ of the reconstruction of any given input data signal $\boldsymbol{x}$ is an implicit function of the data model, because $\tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$ depends on $\boldsymbol{y}$. In particular, the parameters that are called basis vectors (defined in section 2) define the features that the data model derives to describe the data. The degree to which any given data-set 'differs from the data model' is thus defined by the mean-squared error $\langle |\boldsymbol{x} - \tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})|^2 \rangle$ of the model-based reconstruction, averaged over that data-set. (It is conventional to use the parentheses $\langle \cdots \rangle$ to indicate an average over a data-set.)

Because the reconstruction error is a function of the data model, and because the data model is defined by the values of the parameters implicit in the functions $y(\boldsymbol{x})$, training the model to best match the training data-set amounts to finding the combination of all these parameter values that makes the mean-squared error of the model-based reconstruction, averaged over the training data-set, as small as possible. This defines the training algorithm. In the language of optimization theory, the training algorithm is a 'gradient descent' algorithm that searches around the parameter space by finding the steepest route down the error surface (the mean-squared error plotted as a function of the parameters) until it arrives at the bottom of a valley in the error surface from where it cannot reduce the mean-squared error any further. At this point, the training is said to have converged at its optimum and the data-model is then as well matched to its training data-set as it can get.

Those parameters implicit in the functions $y(\boldsymbol{x})$ derived in this way by training to minimize the mean-squared reconstruction error are called 'adaptive parameters' because they are derived from the data by an adaptive process. Those parameters implicit in the $y(\boldsymbol{x})$ that are adaptive parameters are the basis vectors and the basis vector scale values, defined in section 2. Other parameters implicit in the functions $y(\boldsymbol{x})$ are not adaptive (i.e. they are not derived from the gradient descent algorithm) and are called 'meta-parameters'. These are the neural threshold, softness, and the number of basis vectors, defined in section 2. Not being adaptive, these three parameters can be set 'manually', e.g. by trial and error; they are typically chosen to make the discrimination index as large as possible for a known faulty data-set. In other words, the meta-parameters are chosen to make componential coding as discriminating as possible at detecting faults.

1.4 Theoretical motivation for componential coding neural networks

Componential coding is so called because the neural network encodes whatever data patterns are fed to its inputs by combining elementary features, or components. The adaptive training process derives a basis set of features from the training data, various subsets of which the network combines in order to reconstruct an optimal replica of the current input data pattern; the training process optimizes, on average, the accuracy of these replicas. Neural networks designed to encode and then reconstruct their input in this way are called auto-encoders. Componential coding is a special kind of auto-encoder algorithm, developed on theoretical grounds, and previously demonstrated in the context of image processing [**18**].

Componential coding has the special 'sparseness' property that relatively few features from the basis set contribute significantly to the reconstruction of the input at any one time. (This means that every feature of the basis set will eventually be used in reconstruction if the network experiences a large enough number of input data patterns, but only relatively small subsets of those features will contribute significantly to the reconstruction of any *individual* input data pattern [**18**].) Familiar linear techniques, such as principal component analysis, do not encode their data sparsely, because they have no constraint to encourage the encoding coordinates to adopt values near zero. Consequently, such linear techniques are able to reconstruct their data well by combining large numbers of components at the same time, even if the technique's components do not individually contain very sophisticated information about the data. Componential coding, on the other hand, needs to represent sophisticated, high-order information about the data within each individual component, if it is to be able to reconstruct the input well by combining only a few components at a time.

This means that componential coding is sensitive to far higher order statistics of the data than familiar techniques (in principle, to all orders rather than to just the first and second). For example, one of the benefits of sensitivity to higher-order statistics is that componential coding can discover time-localized features within a signal having time-invariant statistics, whereas linear principal component analysis would be able to find nothing more than periodic eigenvectors (as proved in reference [**18**]). This novel sensitivity to higher-order statistics derives from the non-linearity of the componential coding neural output function. The practical implications are that componential coding can detect faults by encoding the data in terms of new kinds of features, which are different from the features used by conventional techniques such as principal component analysis or Fourier analysis. These features can access much more information about the data than the second-order moments on which principal component analysis relies, and so componential coding can detect faults with greater sensitivity as a result. This improved sensitivity is demonstrated with the aid of numerical simulations in section 3 and in the context of a real-world problem in Part 2 of this paper. Another practical advantage of componential coding is that it requires no time-synchronization of the input signal with respect to the absolute angular position of the rotating parts of the machine, as will be explained in section 2.2. Another advantage is that componential coding requires the minimum of expert knowledge and judgement of the application domain, because it derives its own feature-set automatically from the properties of the application-specific data, through an automatic adaptive training process, according to an objective optimization criterion.

2 THE COMPONENTIAL CODING ALGORITHM

2.1 Two variants of the auto-encoder neural network architecture

The set of features or components that constitute the trained data-model is encoded in a set of neural network weight vectors $\{\boldsymbol{w}_c\}$, where the index c labels the component and runs from 1 to n_c, the total number of components in the basis set. These adaptive parameters are analogous to the basis vectors of principal component analysis and may also be called 'basis vectors' in the context of the componential coding algorithm. The input data from the n_s monitoring sensors are fed to the auto-encoder network encoded in the input data vector $\boldsymbol{x}$, whose $n_s n_t$ coordinates are formed from n_s time sequences, each consisting of n_t sampled amplitude measurements, generated from the n_s sensors. Typically, the size of the basis set n_c is chosen to be much smaller than the dimensionality $n_s n_t$ of the input vector space $\boldsymbol{x}$.

In the most general variant of the neural network architecture, the so-called Joint Channel Architecture Network (JCAN), the coordinates of each basis vector $\boldsymbol{w}_c$ correspond one-to-one with the coordinates of the input data vector $\boldsymbol{x}$, so that each basis vector spans n_s channels of data if the input data comes from a number n_s of sensors. Therefore, the dimensionalities of *each* of the basis vectors will be $n_s n_t$ in the JCAN architecture, i.e. the same as the dimensionality of the vector space $\boldsymbol{x}$. Since the JCAN basis vectors span multiple sensors, they can encode correlations between different sensors, such as their mutual phase relationships; in principle, therefore, the JCAN variant can be used to detect anomalies in the correlations between sensors.

Another useful variant is called the Independent Channel Architecture Network (ICAN), in which each basis vector is associated with only one of the sensor channels, so that ICAN basis vectors have a dimensionality of only n_t. In the ICAN variant, therefore, different basis vectors encode features in different sensor channels, so there is no possibility that a basis vector can encode correlations between different sensors.

The componential coding algorithm for the ICAN is actually a constrained special case of the JCAN algorithm, in which all but n_t of a basis vector's $n_s n_t$ possible coordinates are constrained to have zero value (thus reducing the $n_s n_t$ degrees of freedom available for a JCAN basis vector to the n_t degrees of freedom available for an ICAN basis vector). In situations where the data from different sensors are only loosely mutually correlated, the ICAN variant can be better than the JCAN variant at optimizing individual features to match data from individual sensors, because ICAN basis vectors are not exposed to the loosely correlated 'clutter' from other sensors.

2.2 The correlation function and time-invariant template matching

In a typical neural network, the outputs or responses of the neurons are computed as (some function of) the scalar product $\xi = \boldsymbol{x} \cdot \boldsymbol{w}_c$. The same is true of the componential coding auto-encoder except that, for each $\boldsymbol{w}_c$, not just one scalar product but n_t scalar products are computed in order to form the periodic correlation function $\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c)$. It is well known that the correlation function between two n_t sample signals is equivalent to an ordered sequence of n_t scalar products, in which one of the two signals is translated with respect to the other by an incremented time-offset before computing each scalar product. Thus, each n_t-sample correlation function $\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c)$ can be thought of as an ordered set of n_t outputs of a sequence of n_t neurons, all exposed to the same input pattern $\boldsymbol{x}$, but whose weight vectors are all differently time-translated replicas of a single canonical template $\boldsymbol{w}_c$.

The correlation function $\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c)$ matches the input data pattern $\boldsymbol{x}$ with the template $\boldsymbol{w}_c$ n_t times, with $\boldsymbol{w}_c$ translated (with respect to $\boldsymbol{x}$) by every one of n_t possible time-offsets. It is this ability to match templates at all possible time-offsets that confers the property of *time invariance* on the componential coding algorithm; provided the n_t time-samples of the input signal span exactly a whole revolution (or a whole number of revolutions) of data from a rotary machine, so that $\boldsymbol{x}$ is a periodic signal having a periodic boundary condition, the algorithm will be independent of the absolute angular position of the rotor. This statement is justified analytically in the context of image processing [**18**], for which essentially the same auto-encoder neural network algorithm has a two-dimensional translation-invariance property exactly analogous to the one-dimensional time-invariance property of the condition-monitoring algorithm. The componential coding algorithm, therefore, has the advantage over non-correlation-based template-matching techniques that it requires no synchronization of the input signal with respect to the absolute angular position. (It does, however, require synchronization with respect to angular velocity, to ensure the requirement that the n_t time-samples span a whole number of revolutions.)

The convolution theorem allows the periodic correlation function $\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c)$ to be computed very efficiently by fast Fourier transform, in the order of $n_t \log(n_t)$ operations instead of the n_t^2 operations that would be required to compute the n_t offset scalar products explicitly. (For a discussion of the convolution theorem, the reader is referred to undergraduate engineering mathematics and computing texts, e.g. reference [**15**] or [**17**].) As has been explained above, the basis vectors $\boldsymbol{w}_c$ for the ICAN variant are one-dimensional time-signals corresponding to individual sensors, and each index c is therefore implicitly associated with a particular sensor

$s(c)$. In the case of the ICAN variant, therefore, the correlation function is computed using the convolution theorem as

$$\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c) = \boldsymbol{F}^{-1}\left(\boldsymbol{F}(\boldsymbol{x}_{s(c)}) \times (\boldsymbol{F}(\boldsymbol{w}_c))^*\right) \tag{1}$$

where $\boldsymbol{x}_s$ indicates the sequence of sampled input data from the sth sensor, the vector function $\boldsymbol{F}(\,)$ indicates the one-dimensional Fourier transform* in the time domain, $\boldsymbol{F}^{-1}(\,)$ indicates the inverse Fourier transform, $(\,)^*$ indicates complex conjugation and $\times$ indicates coordinate-wise multiplication of two vectors to yield a vector of coordinate products. The basis vectors $\boldsymbol{w}_c$ for the JCAN variant each span all n_s sensor channels, however, so for the JCAN variant the correlation function is computed as

$$\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c) = \sum_{s=1}^{n_s} \boldsymbol{F}^{-1}\left(\boldsymbol{F}(\boldsymbol{x}_s) \times (\boldsymbol{F}(\boldsymbol{w}_{c,s}))^*\right) \tag{2}$$

where $\boldsymbol{w}_{c,s}$ indicates the sth sensor channel of the cth basis vector. There is a summation over sensors because the purpose of the correlation is to compute a sequence of scalar products $\boldsymbol{x} \cdot \boldsymbol{w}_c$, and because those scalar products for JCAN neurons must clearly sum over all n_s sensors.

2.3 Model-based data reconstruction in the componential coding algorithm

In most neural network algorithms, a non-linear threshold neural response function $r(\xi)$ is applied to the results of the scalar products $\xi \equiv \boldsymbol{x} \cdot \boldsymbol{w}$ to compute the neurons' outputs y; in the componential coding algorithm also, a non-linear threshold function $r(\xi)$ is applied to (every element of) each of the n_c correlation functions $\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c)$ to yield n_c output vectors $\boldsymbol{y}_c(\boldsymbol{x})$, each of n_t samples:

$$\boldsymbol{y}_c(\boldsymbol{x}) = r(\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c))$$

i.e.

$$(\boldsymbol{y}_c(\boldsymbol{x}))_t \equiv r((\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c))_t) \qquad \text{for } t = 1, \ldots, n_t \tag{3}$$

The set of the n_c output vectors $\boldsymbol{y}_c(\boldsymbol{x})$ forms the auto-encoder's output code for the current input pattern $\boldsymbol{x}$. This information, which implicitly incorporates the adaptive data-model $\{\boldsymbol{w}_c\}$, is used to compute a model-based reconstruction $\tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$ of the current input pattern $\boldsymbol{x}$, by convolving each of the $\boldsymbol{y}_c(\boldsymbol{x})$ with the corresponding $\boldsymbol{w}_c$ and combining the n_c resulting convolution functions by the weighted summation

$$\tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x}) \equiv \sum_{c=1}^{n_c} b_c \boldsymbol{cnv}(\boldsymbol{w}_c, \boldsymbol{y}_c(\boldsymbol{x})) \tag{4}$$

The superscript W is present because of the implicit dependence of the model-based reconstruction $\tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$ on the model parameters $\{\boldsymbol{w}_c\}$. W will be used as a label to identify the training data-set used to optimize those parameters; the superscript W indicates where a reconstruction $\tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$ has been obtained using a data model $\{\boldsymbol{w}_c\}$ optimized for the particular training set W. The n_c numbers b_c that weight the sum are new parameters called basis scales; their values are determined by a (single-step) optimization procedure described in section 2.5.

The periodic convolution function $\boldsymbol{cnv}(\boldsymbol{w}_c, \boldsymbol{y}_c(\boldsymbol{x}))$ can be computed very efficiently by fast Fourier transform. Because the $\boldsymbol{w}_c$ for the ICAN variant are just one-dimensional time-signals corresponding to individual sensors $s(c)$, this convolution function is computed for the ICAN variant as

$$\boldsymbol{cnv}(\boldsymbol{w}_{s(c)}, \boldsymbol{y}_c(\boldsymbol{x})) = \boldsymbol{F}^{-1}\left(\boldsymbol{F}(\boldsymbol{w}_{s(c)}) \times \boldsymbol{F}(\boldsymbol{y}_c(\boldsymbol{x}))\right) \tag{5}$$

However, in the case of the JCAN variant, each $\boldsymbol{w}_c$ spans all n_s sensor channels, so the convolution with $\boldsymbol{y}_c(\boldsymbol{x})$ must be performed for all n_s channels in the JCAN case, i.e.

$$(\boldsymbol{cnv}(\boldsymbol{w}_c, \boldsymbol{y}_c(\boldsymbol{x}))) = \boldsymbol{F}^{-1}(\boldsymbol{F}(\boldsymbol{w}_{c,s}) \times \boldsymbol{F}(\boldsymbol{y}_c(\boldsymbol{x}))) \qquad \text{for } s = 1, \ldots, n_s \tag{6}$$

With these definitions of $\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c)$ and $\boldsymbol{cnv}(\boldsymbol{w}_c, \boldsymbol{y}_c(\boldsymbol{x}))$, it may be proven that the reconstruction $\tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$ is invariant with respect to (wrap-around) translation of any basis vector $\boldsymbol{w}_c$, by any arbitrary time-offset. Conversely, any wrap-around translation of $\boldsymbol{x}$ with respect to fixed basis vectors will just translate the reconstruction $\tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$ accordingly but not alter its shape; the accuracy of the reconstruction will always be independent of the absolute angular position of the rotor.

Because the reconstruction is insensitive to time-translation of any basis vector with respect to any other, the ICAN variant of the componential coding algorithm is insensitive to correlations in time between different sensors. However, the JCAN variant *is* sensitive to correlations between sensors, because individual JCAN basis vectors span more than one sensor.

* It is implicit that the normalization convention for the Fourier transform and its inverse are chosen so as to preserve vector Euclidean length, i.e. $|\boldsymbol{F}(z)| = |\boldsymbol{F}^{-1}(z)| = |z|$ for arbitrary vectors z.

2.4 Deriving matched basis vectors by minimizing the mean reconstruction error

Through the adaptive training process, the basis vectors become matched to the training data so as to optimize the reconstruction on average. The mean-squared reconstruction error E over the training set W is given by

$$E^{\mathrm{W}} = \left\langle |\boldsymbol{x} - \widetilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})|^2 \right\rangle_{\{\boldsymbol{x}\in \mathrm{W}\}} \tag{7}$$

where $\langle \ldots \rangle_{\{\boldsymbol{x}\in \mathrm{W}\}}$ indicates the average over all $\boldsymbol{x}$ in the data-set W (i.e. the training set) and where the vector Euclidean length $|\boldsymbol{a}| \equiv \sqrt{\boldsymbol{a} \cdot \boldsymbol{a}}$. The reconstruction is optimized by minimizing E^{W} with respect to $\widetilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$'s implicit adaptive parameters $\boldsymbol{w}_c$, by a simple iterative gradient descent on E^{W} in the vector space of the $\boldsymbol{w}_c$. This involves the replacement

$$\boldsymbol{w}_c \rightarrow \frac{\boldsymbol{w}_c + \lambda \Delta_c / |\Delta_c|}{|\boldsymbol{w}_c + \lambda \Delta_c / |\Delta_c||} \qquad \text{for each } c = 1, \ldots, n_c \tag{8}$$

at each iteration of the adaptive process, where

$$(\text{The transpose of}) \Delta_c \equiv -\frac{1}{2}\frac{\partial E^{\mathrm{W}}}{\partial \boldsymbol{w}_c} \qquad \text{for each } c = 1, \ldots, n_c \tag{9}$$

and where the learning rate λ is a positive constant less than 1. The particular form of gradient descent prescribed by equation (8) clearly maintains the length constraint

$$|\boldsymbol{w}_c| = 1 \tag{10}$$

on each basis vector at all times. Working out the partial derivatives $\partial E^{\mathrm{W}}/\partial \boldsymbol{w}_c$ gives

$$\Delta_c = b_c \left\langle \boldsymbol{cor}\left(\boldsymbol{x} - \widetilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x}), \boldsymbol{y}_c(\boldsymbol{x})\right) + \boldsymbol{cor}\left(\boldsymbol{x}, z_c(\boldsymbol{x}) \times \boldsymbol{h}_c(\boldsymbol{x})\right) \right\rangle_{\{\boldsymbol{x}\in \mathrm{W}\}} \tag{11}$$

where $z_c(\boldsymbol{x})$ are vectors of first derivatives of the neural outputs:

$$z_c(\boldsymbol{x}) = r'(\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c))$$

i.e.

$$(z_c(\boldsymbol{x}))_t \equiv \left.\frac{\mathrm{d}r}{\mathrm{d}\xi}\right|_{\xi = (\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c))_t} \qquad \text{for } t = 1, \ldots, n_t \tag{12}$$

and where the correlation with the single-channel vectors $\boldsymbol{y}_c(\boldsymbol{x})$ and $z_c(\boldsymbol{x}) \times \boldsymbol{h}_c(\boldsymbol{x})$ is computed for each channel of the n_s channel vectors $\boldsymbol{x}, \widetilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$ and Δ_c. The single-channel vectors $\boldsymbol{h}_c(\boldsymbol{x})$ are defined by

$$\boldsymbol{h}_c(\boldsymbol{x}) \equiv \boldsymbol{cor}\left(\boldsymbol{x} - \widetilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x}), \boldsymbol{w}_c\right) \tag{13}$$

which, analogous to the definition of $\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c)$, involves just the one channel $s(c)$ in the case of the ICAN, or the summation over all channels for the JCAN. At each iteration, the average $\langle \cdots \rangle_{\{\boldsymbol{x}\in \mathrm{W}\}}$ can either be taken over the full training set W or over a sufficiently statistically representative random subsample of W (which saves computation time).

Because equations (8) and (9) implement gradient descent on E^{W}, the adaptation is guaranteed to converge on a minimum of E^{W}, provided λ is not very large, and the training set not so unrepresentatively small and unrepeatable, that the basis vector updates just jump from one near-minimum to another [**18**]. This is a well-known provable property of all gradient descent optimization algorithms.

2.5 One-step optimization of the basis vector scale values

The best match of the basis vectors $\{\boldsymbol{w}_c\}$ to the features of the training data will be obtained when the basis vector scale parameters $\{b_c\}$ are set at values that minimize E^{W}; thus, $\{\boldsymbol{w}_c\}$ and $\{b_c\}$ should be optimized jointly to obtain the most accurate model of the data. It would be possible to optimize the $\{b_c\}$ by gradient descent on E^{W}, as was done for the $\{\boldsymbol{w}_c\}$, but there is a more direct method, which is possible because E^{W} depends quadratically on the $\{b_c\}$; a set of simultaneous equations can be solved for the optimal values of the $\{b_c\}$. The solution is obtained by the matrix multiplication

$$b_c = \sum_{c'=1}^{n_c} \left(\mathbf{M}^{-1}\right)_{cc'} \langle \boldsymbol{x} \cdot \boldsymbol{cnv}(\boldsymbol{w}_{c'}, \boldsymbol{y}_{c'}(\boldsymbol{x})) \rangle_{\{\boldsymbol{x}\in \mathrm{W}\}} \tag{14}$$

involving the inverse $\mathbf{M}^{-1}$ of the n_c-by-n_c matrix

$$\mathbf{M}_{cc'} \equiv \langle \boldsymbol{cnv}(\boldsymbol{w}_c, \boldsymbol{y}_c(\boldsymbol{x})) \cdot \boldsymbol{cnv}(\boldsymbol{w}_{c'}, \boldsymbol{y}_{c'}(\boldsymbol{x})) \rangle_{\{\boldsymbol{x}\in \mathrm{W}\}} \tag{15}$$

The scalar products in both of these equations involve summation over all time-samples and channels for the JCAN variant [the convolutions $\boldsymbol{cnv}(\boldsymbol{w}_c, \boldsymbol{y}_c(\boldsymbol{x}))$ are $n_s n_t$-dimensional vectors for JCAN]. For ICAN, these scalar products simply involve summation over the time-samples, for individual channels $s(c)$ and $s(c')$ [the $\boldsymbol{cnv}(\boldsymbol{w}_c, \boldsymbol{y}_c(\boldsymbol{x}))$ are n_t-dimensional vectors for ICAN]. For ICAN, $\mathbf{M}_{cc'} = 0$ if $s(c') \neq s(c)$, i.e. if the cth and c'th basis vectors are not associated with the same sensor.

2.6 The non-linear neural threshold function

The purpose of the non-linear threshold function $r(\xi)$ applied at the neurons' outputs is to enforce the property of *sparseness* on the code formed by that collection of outputs (see section 1.4 and reference [**18**]).

The idea is that the outputs of relatively few of the neurons should dominate the outputs of the rest, for any individual pattern $\boldsymbol{x}$; the larger output values will be sparsely distributed over the collection of neurons. This forces the adaptive algorithm to make optimal use of only a few basis vectors at a time, when reconstructing each pattern as an output-weighted summation of basis vectors. Consequently, the adaptive algorithm will be forced to pack high-order information about features of the data into individual basis vectors, if it is to be able to reconstruct any given data pattern accurately as a combination of relatively few basis vectors at a time.

The number of neurons whose outputs are large is reduced simply by thresholding all the neurons' outputs; if the threshold value ϑ is chosen appropriately, relatively few of the neurons will fire above threshold for any given data pattern $\boldsymbol{x}$—those neurons whose weight vector $\boldsymbol{w}$ matches the data pattern so well that $\boldsymbol{x} \cdot \boldsymbol{w} > \vartheta$. (Here the n_c correlation functions $\boldsymbol{cor}(\boldsymbol{x}, \boldsymbol{w}_c)$ with the n_c basis vectors can be envisaged as being equivalent to the set of scalar products $\boldsymbol{x} \cdot \boldsymbol{w}$ with $n_c n_t$ neurons' weight vectors $\boldsymbol{w}$, as discussed in section 2.2.)

One useful form of threshold function is

$$r(\xi) = \left\{ \sigma \log_e \left[1 + \exp\left(\frac{\xi - \vartheta}{\sigma} \right) \right] \right\}^2 \tag{16}$$

which has the limiting behaviours

$$\lim_{\xi \to +\infty} r(\xi) = (\xi - \vartheta)^2 \tag{17}$$

and

$$\lim_{\xi \to -\infty} r(\xi) = \sigma \exp\left(2 \frac{\xi - \vartheta}{\sigma} \right) \tag{18}$$

in the so-called above-threshold and subthreshold limits respectively. The threshold parameter ϑ determines how large the projection $\xi = \boldsymbol{x} \cdot \boldsymbol{w}$ (of a data vector $\boldsymbol{x}$ on to a basis vector $\boldsymbol{w}$) must be in order for the corresponding neuron's output to be above threshold, where ϑ is measured in the same physical units as the input data vectors (amperes or pascals for example). The softness parameter σ, which must be greater than zero, determines how smoothly the graph of $r(\xi)$ makes the transition from the subthreshold limit to the above-threshold limit; it is the width of the transition region on that graph. The parameter σ is also measured in the same physical units as the input data vectors. In all the demonstrations given in this paper and in the companion Part 2 paper [**14**], all input data vectors $\boldsymbol{x}$ were normalized ($\boldsymbol{x} \to \boldsymbol{x}/|\boldsymbol{x}|$) to dimensionless unit Euclidean length, so $\boldsymbol{x}$, ϑ and σ are measured in dimensionless units. Whenever input data vectors are normalized to unit length in this way, the match ξ can never exceed 1; thus, the requirement $\xi > \vartheta$ for at least one neuron's output to be above threshold (after training) sets an upper limit of 1 on the range of values that are appropriate for ϑ and σ. The advantage of the exponentially decreasing subthreshold behaviour is that the gradient $\mathrm{d}r/\mathrm{d}\xi$, which enters into the basis vector update equation (11), is never identically zero so, even if its output is below threshold for the entire training set, a neuron still has the capacity to change its basis vector and so 'bootstrap' its output above threshold [**18**]. The advantage of having a monotonically increasing gradient $\mathrm{d}r/\mathrm{d}\xi$ is that the gradient descent algorithm is less easily trapped in local minima than it would be if $\mathrm{d}r/\mathrm{d}\xi$ were to fall away to zero for large ξ.

2.7 Anomaly detection and the average and variance discrimination indices

If the training has optimized a good model of non-anomalous data, then the reconstruction $\tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$ of any non-anomalous data pattern $\boldsymbol{x}$ should be a good approximation to the actual data pattern $\boldsymbol{x}$, even if $\boldsymbol{x}$ is previously unseen (i.e. not a member of the training set W). Thus, the anomaly vector, defined as $\boldsymbol{a} \equiv \boldsymbol{x} - \tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})$, will typically have smaller vector length $|\boldsymbol{x} - \tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})|$ for a non-anomalous data pattern than for an anomalous one. Conversely, any previously unseen data-set M may be monitored for anomalies by comparing the value of its mean-squared anomaly vector length

$$E^{\mathrm{M}} = \left\langle |\boldsymbol{x} - \tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})|^2 \right\rangle_{\{\boldsymbol{x} \in \mathrm{M}\}} \tag{19}$$

with that of a previously unseen control data-set U known to be non-anomalous

$$E^{\mathrm{U}} = \left\langle |\boldsymbol{x} - \tilde{\boldsymbol{x}}^{\mathrm{W}}(\boldsymbol{x})|^2 \right\rangle_{\{\boldsymbol{x} \in \mathrm{U}\}} \tag{20}$$

The average discrimination index (ADI) for the data-set M to be monitored, defined as

$$\mathrm{ADI}^{\mathrm{M}} \equiv \frac{E^{\mathrm{M}}}{E^{\mathrm{U}}} - 1 \tag{21}$$

should clearly have a value relatively close to zero if M is non-anomalous, because then M should have similar statistical moments to those of the non-anomalous control data-set U, including similar E values. The ADI should be significantly greater than zero if M is anomalous, because the model $\{\boldsymbol{w}_c\}$ (trained on non-anomalous data W) should be able to reconstruct M less accurately than U. Significantly different ADI values for different anomalous data-sets can also be exploited to discriminate between different types of fault or between faults of different severity.

The ADI is based on averaged reconstruction errors over the monitored data-set M and, consequently, is most useful when a relatively large proportion of the

data vectors in M are anomalous. However, some anomalies manifest themselves only over a relatively small portion of the data. To highlight these types of anomalies, the variance discrimination index (VDI) was defined as

$$\mathrm{VDI}^{\mathrm{M}} \equiv \frac{V^{\mathrm{M}}}{V^{\mathrm{U}}} - 1 \tag{22}$$

where V^{M} is the variance of the reconstruction errors $|\boldsymbol{a}|^2 = |\boldsymbol{x} - \boldsymbol{x}^{\mathrm{W}}(\boldsymbol{x})|^2$ of all the data vectors $\boldsymbol{x}$ in a data-set M. Both the ADI and VDI may be used for an overall measure of anomaly detection.

2.8 Optimization of the discrimination index with respect to meta-parameters

The gradient descent algorithm only optimizes the adaptive parameters $\{\boldsymbol{w}_c\}$ and $\{b_c\}$, not the fixed parameters which are the threshold (ϑ), softness (σ) and the number of basis vectors (n_c). The fixed parameters of the gradient descent algorithm may be adjusted by a non-gradient search algorithm, such as a genetic algorithm, so as to optimize either $\mathrm{ADI}^{\mathrm{M}}$ or $\mathrm{VDI}^{\mathrm{M}}$ for a particular fault condition represented in a faulty data-set M. The genetic algorithm optimization of the 'meta-parameters' ϑ, σ and n_c is conducted as an outer loop; for each iteration of this outer loop, the gradient optimization of the adaptive parameters $\{\boldsymbol{w}_c\}$ and $\{b_c\}$ is iterated to convergence as an inner loop.

Maximizing $\mathrm{ADI}^{\mathrm{M}} = (E^{\mathrm{M}}/E^{\mathrm{U}}) - 1$ implicity acts to reduce E^{U} and so improves generalization from the training set W to the non-anomalous control set U. When optimizing a discrimination index, it is important that U actually contains distinct data from W, otherwise the genetic algorithm optimization of the meta-parameters may result in overfitting to the training set W at the expense of generalization to unseen data-sets.

3 DEMONSTRATIONS

The companion paper (Part 2 [**14**]) assesses componential coding in its application to real data recorded from a conventional induction motor and from a novel transverse flux motor. In this paper (Part 1), the principles and capabilities of the technique are illustrated in simple experiments using synthetically created data-sets, representative of the properties of condition monitoring data. In particular, the detection of small anomalies and discrimination characteristics are addressed by comparison of componential coding with conventional waveform examination and Fourier spectrum analysis.

3.1 Data-sets

Condition monitoring data from rotary machines usually exhibits periodicity [**19**]. The dominant (principal) frequency components may be, for example, the main shaft frequency, mesh frequency of a gearbox, power supply frequency in an electrical machine or firing frequency of an engine. In addition, the data are usually contaminated by noise. In this paper the capability of componential coding is illustrated using a synthetic training data-set W of data-vectors $\boldsymbol{x}^{\mathrm{W}} = (x_1^{\mathrm{W}}, \ldots)$ based on a simple signal model*

$$x_t^{\mathrm{W}} = \sin(2\pi f_0 t + \theta) + N(t) \tag{23}$$

where the single principal frequency f_0 is set at 1024 Hz and $N(t)$ is random noise distributed uniformly between -0.1 and 0.1 (i.e. the noise component is statistically independent from one sample to the next). With this noise amplitude range, the data have a signal-to-noise ratio of 55.5 dB. The sample interval for the synthetic data-set was fixed at $\Delta t = 1/16\,384$ s (i.e. corresponding to a theoretical sample rate of 16 384 Hz) while 65 536 data points were generated and divided equally into two subsets so that they could be separately used as training data (W) and unseen control data (U) respectively.

For an anomaly detection example, an anomaly data model was also developed by altering equation (23) to

$$x_t^{\mathrm{M}_l} = \sin(2\pi f_0 t + \theta + \Delta\theta) + \Delta x + \Delta_n N(t), \qquad l = 0, 1, 2, 3, 4 \tag{24}$$

With this data model, five monitored data-sets ($\mathrm{M}_l = \mathrm{M}_0, \ldots, \mathrm{M}_4$) were generated (as described below). With the exception of the healthy data-set M_0, each was specifically generated to simulate a different type of signal anomaly typically experienced in condition monitoring data. For all the generated data-sets described below it should be assumed, unless explicitly stated, that $\Delta\theta$ and Δx are set to 0 and Δ_n is set to 1; these are referred to as the default settings.

M_0: case 0

All the default settings for $\Delta\theta$, Δx and Δ_n were used so that equation (24) became equivalent to equation (23). This case represents a healthy baseline data-set, but because the noise is a random variable, the vector M_0 will not be identical to either the training data vector or the control data vector.

* All input data vectors $\boldsymbol{x}$ were subsequently normalized ($\boldsymbol{x} \to \boldsymbol{x}/|\boldsymbol{x}|$) to dimensionless unit Euclidean length, so $\boldsymbol{x}$, ϑ and σ are measured and given in dimensionless units.

M_1: case 1

A globally distributed random change was introduced with Δx being distributed randomly between -0.005 and 0.005. Such a condition may result from looseness within a machine (e.g. looseness of the stator end windings in an induction motor or looseness of one of the mounting bolts) or cavitation in a pump.

M_2: case 2

A small degree of frequency modulation was created with

$$\Delta\theta = 0.05\sin(16\pi t) \tag{25}$$

Such a condition may result from a broken rotor bar in an induction motor, eccentricity of a gear or a bent shaft for example.

M_3: case 3

Small and localized amplitude variation occurring almost periodically but with a small amount of positional variance was created by

$$\Delta x = \begin{cases} 0.05\sin(2\pi f_0 t), & (4k\pi) \leqslant 2\pi f_0 t + \varepsilon \leqslant (\pi + 4k\pi) \\ 0, & \text{otherwise} \end{cases} \tag{26}$$

where $k = 1, 2, 3, 4, \ldots$ and ε is a random value between 0 and 3π. Such a condition may result from a weakened tooth in a gearbox (perhaps being caused by a bending fatigue crack).

M_4: case 4

Small and localized transients occurring almost periodically but with a small amount of positional variance were seeded as described by

$$\Delta x = \begin{cases} 0.003\,\mathrm{e}^{-0.1t} & \\ \cos(2\pi \times 20 f_0 t), & (4k\pi) \leqslant 2\pi f_0 t + \varepsilon \leqslant (\pi + 4k\pi) \\ 0, & \text{otherwise} \end{cases} \tag{27}$$

where $k = 1, 2, 3, 4, \ldots$ and ε is a random value between 0 and 3π. Such a condition may result from the impact transients caused by pitting/hairline cracks in a bearing or those caused by a rolling element bearing during fatigue failure of the race. Alternatively, this condition may occur as a result of a faulty valve system in a diesel engine or compressor.

The seeded anomalies are so small that the healthy data (case 0) and anomalous data (cases 1 to 4) are indistinguishable by the naked eye, which is demonstrated by Fig. 3a, which shows all five waveform traces overlaid. Furthermore, simple waveform shape analysis

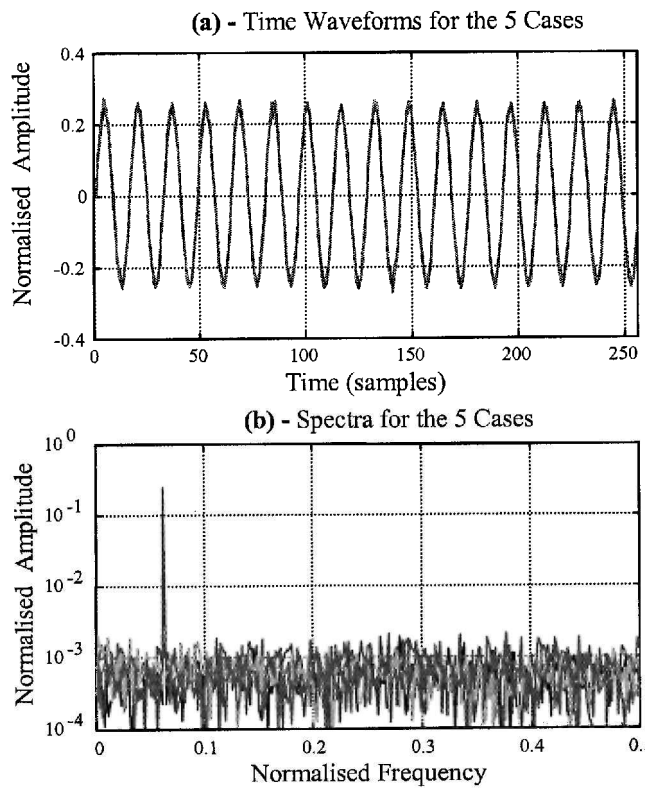

Fig. 3 Numerical data-sets and spectrum

Table 1 Waveform measures of the five simulated data-sets

Waveform measure	R.m.s.	Difference from case 0 (%)	Kurtosis	Difference from case 0 (%)
Case 0	0.17695	0.00000	1.50906	0.00000
Case 1	0.17695	0.00096	1.49742	0.77144
Case 2	0.17695	0.00004	1.51403	0.32909
Case 3	0.17695	0.00003	1.51026	0.07983
Case 4	0.17695	0.00002	1.51239	0.22073

such as root-mean-square (r.m.s.) and kurtosis similarly reveal no significant differences between the data-sets (Table 1). Within the frequency domain the five cases also overlay in a near identical fashion (Fig. 3b), with each of the spectra having a single principal frequency component and superimposed white-spectrum noise.

3.2 Network training and optimized configuration

To separate the simulated anomalies using componential coding, an ICAN was initially used. The dimension of the basis vectors (n_t) was set to 32 data points. This dimension was chosen so as to cover two periods of the principal frequency component and, therefore, allow better detection of local distortion of the waveform than if only one period was covered. Larger dimensions of the basis vector are likely to yield even better detection, but this increases the computational work required during training and optimization. As with all applications of this algorithm for periodic data, each such double period of data was selected and presented to the network without needing to synchronize the signal with any fixed point in time (such as with a once-per-revolution signal).

The threshold and the number of basis vectors were optimized through a genetic algorithm [**20**]. To do this, a new data-set was formed using equation (24) with the same default settings, apart from Δ_n which was set to 2 (i.e. the amount of noise was doubled). Network optimization was then achieved by aiming towards maximum discrimination (based on the ADI) between the new data set and the trained network model. The optimized parameters were subsequently found to be 1.06 for the threshold and 10 for the number of basis vectors.

Figure 4b shows the profiles of the 10 basis vectors formed following optimization. The profiles reflect the strongly periodic nature of the training data. However, some of the basis vectors exhibit sharp, localized variation (spikiness), indicating the noise contained within the training data.

The bars in Fig. 4a illustrate the amplitudes of the basis vector scale parameters. The amplitude of each scale parameter is related to the degree of similarity of a basis vector with the training data (with smaller-scale parameters corresponding to more similar basis vectors). Basis vector number 4, for example, has a high scale parameter and corresponds to a very spiky basis vector profile. Such spiky features are difficult to identify in the monitored data. On the other hand, the small-scale parameters for basis vectors 1, 3, 5 and 7 correspond to smoother basis vectors and, therefore, appear to be more similar to the monitored data. These similarity/dissimilarity characteristics can be utilized in condition monitoring [**20**].

3.3 Detection using the ICAN

Using the optimized network, anomaly detection was carried out by comparing the anomaly vectors and also by plotting the ADI and VDI in a scatter graph so that visual separation could be achieved. Figure 5 shows sections of the reconstruction error signals for the different data cases (produced by sequentially arranging the anomaly vectors into one time-sequence for each case). For case 0, the amplitude of the reconstruction error is relatively small and there appears to be no

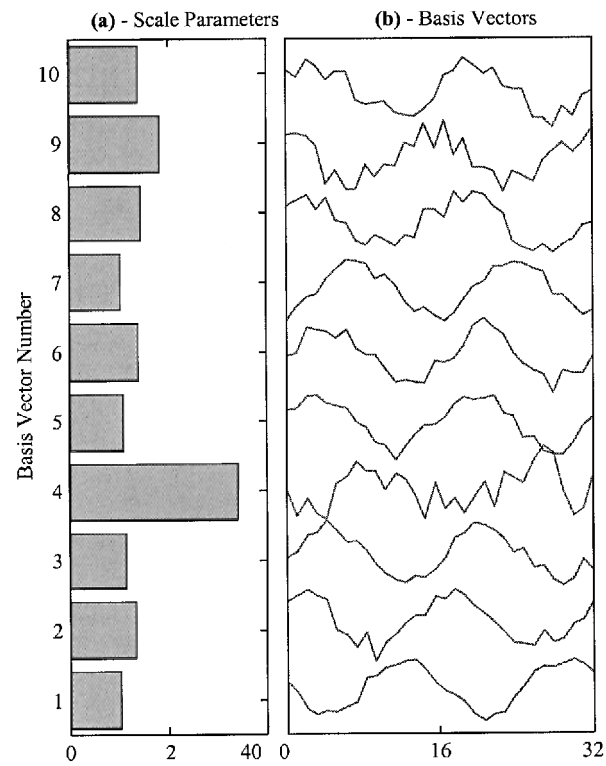

Fig. 4 Optimized network configuration

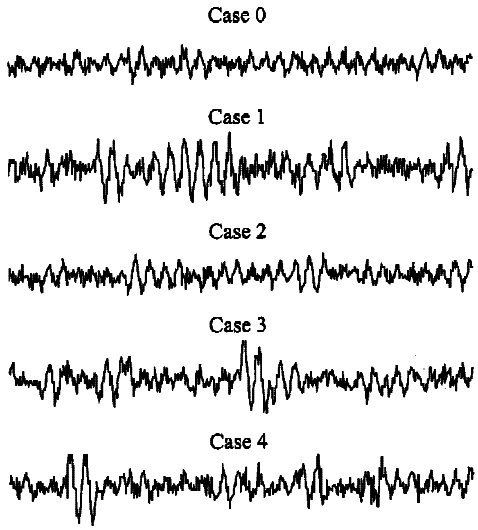

Fig. 5 Reconstruction error signals

significant localized changes. This is to be expected because case 0 is very similar to the training data-set (both were formed using the same signal model). For the other cases, the relatively large global amplitudes (particularly for case 1) and distinct localized distortions (particularly for cases 3 and 4) enable separation of the synthetically created anomalous data-sets (cases 1 to 4). This demonstrates that componential coding-based anomaly detection is more capable than conventional wave shape visualization (Fig. 3a) and spectrum analysis (Fig. 3b). (A detailed and systematic benchmarking assessment is provided in the accompanying Part 2 of the paper [**14**].)

From the scatter plot of the ADI against VDI (Fig. 6), it can be observed that the anomalous cases are clearly separated from the healthy baseline case (case 0). This demonstrates that componential coding can provide reliable and robust anomaly detection.

In addition, the discrimination performance of componential coding in separating varying degrees of anomaly severity was investigated. This was achieved by incrementally adjusting the severity of the seeded anomalies in the monitored data-sets and measuring the combined ADI and VDI (by the Euclidean distance to case 0) for each severity. The range of severities for each anomaly case are summarized below:

M_1: *case 1.* The vector elements Δx were created with random values between an increasing preset range (up to a range from -0.02 to 0.02).

M_2: *case 2.* $\Delta\theta$ was varied from 0 to $0.4\sin(16\pi t)$.

M_3: *case 3.* For $(4k\pi) \leqslant 2\pi f_0 t + \varepsilon \leqslant (\pi + 4k\pi)$, Δx was varied from 0 to $0.32\sin(2\pi f_0 t)$.

M_4: *case 4.* For $(4k\pi) \leqslant 2\pi f_0 t + \varepsilon \leqslant (\pi + 4k\pi)$, Δx was varied from 0 to $0.02\,\mathrm{e}^{-0.1t}\cos(2\pi \times 20 f_0 t)$.

Figure 7 shows that the combined discrimination index amplitudes (with respect to case 0) exhibit monotonously growing trends as the amplitudes of anomalies seeded are increased. This demonstrates that componential coding is also capable of making a correct assessment of anomaly/fault severity.

3.4 Detection using the JCAN

A network with two channels of data were used to study the capability of the JCAN variant in anomaly detection. The data were formed as for the ICAN study but a phase shift (θ) of 120° was introduced for the

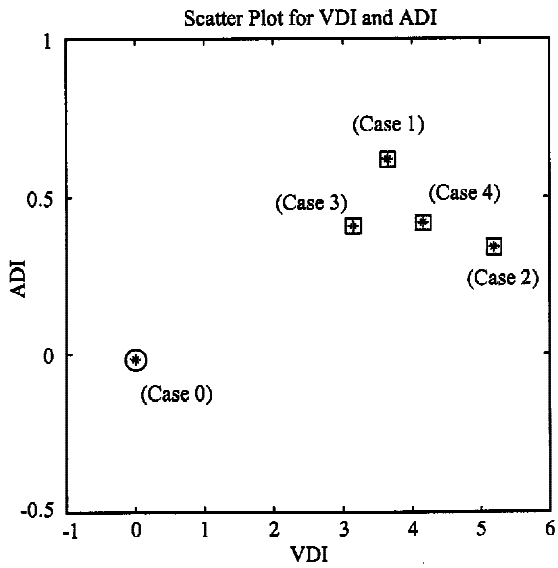

Fig. 6 Anomaly detection results

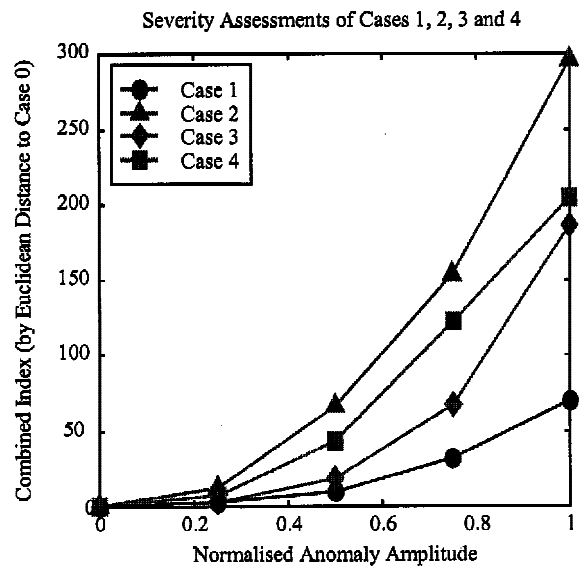

Fig. 7 Discrimination for the degrees of anomalies

Table 2 Anomaly detection by the JCAN and the ICAN (Euclidean distances of the ADI and VDI from case 0)

Anomaly case (Chx)	Case 1	Case 2	Case 3	Case 4	Average
JCAN: one-channel anomaly (channel 1 = case 0; channel 2 = Chx)	1.50	1.88	1.25	3.97	2.15
JCAN: two-channel anomaly (channel 1 = Chx; channel 2 = Chx)	3.79	5.60	3.58	6.34	4.83
ICAN (channel 1 = Chx)	3.75	5.21	3.18	4.19	4.07

synthetically created monitored data-sets. Based on these data, the JCAN was optimized using the same procedure as that used for the ICAN, and it was subsequently found that a threshold of 1.15 and 9 basis vectors provided the optimal configuration (based on the maximized ADI). This configuration is very close to that of the ICAN with both network variants requiring around 10 basis vectors and a high (close to unity) threshold for best anomaly detection.

With the optimized JCAN, the detection of phase variation was studied by inducing a small amount of phase shift (between 0.01 and 0.04 rad) to the data of the second of the two channels. Figure 8 shows that the combined ADI and VDI (by Euclidean distance) increases as the phase shift between the two data channels is increased. This demonstrates that the JCAN allows both detection and discrimination of phase variations (a potential anomalous feature).

Detection of the four anomalous cases (cases 1 to 4) by the JCAN was also conducted by applying two channels of data. Two studies were carried out: the first used the healthy data-set (case 0) along with one other anomalous data case (chosen from cases 1 to 4); the second study used identical data-sets (chosen from cases 1 to 4) for both of the channels. The severity of the anomaly induced in each channel was the same as that used in the ICAN study. Table 2 shows the detection results (measured by the Euclidean distance of the ADI and VDI from case 0) for the four anomalous cases and compares the JCAN with the ICAN. From both the individual and the average results, the JCAN provides better detection capability if the anomaly occurs in two channels simultaneously. However, the ICAN performs better if the anomaly occurs in one channel only. This comparison illustrates the principles explained in section 2.1 regarding the appropriateness of the ICAN for detecting anomalies in individual sensors and the JCAN for detecting anomalous correlations between sensors.

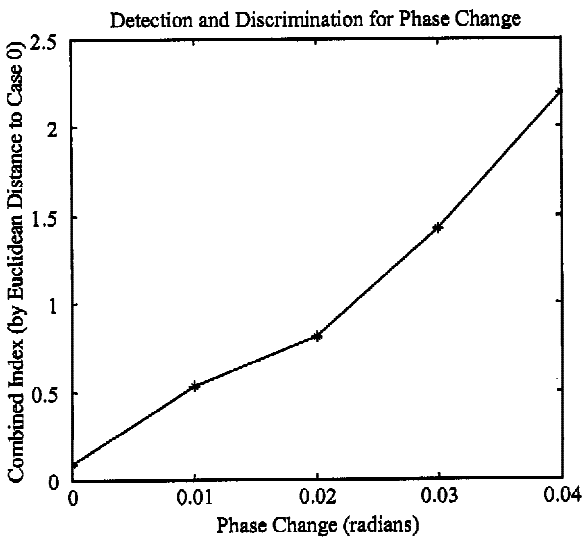

Fig. 8 Detection of phase variation by the JCAN

4 CONCLUSIONS

This paper explains the principles of componential coding in the context of its application to condition monitoring of rotating plant. It demonstrates that componential coding can be used to detect and discriminate anomalies in periodic signals, without needing to rely on prior knowledge of the nature of those signals and without needing to synchronize those signals with any fixed point in time, such as a once-per-revolution signal. The paper illustrates how componential coding can be used to detect a variety of (simulated) typical fault conditions that cannot be detected by direct inspection or by simple waveform shape analysis, such as root-mean-square and kurtosis. The paper further illustrates how componential coding can be used to measure and discriminate the severity of such faults. The paper also illustrates how one of the variants of the componential coding algorithm, the Joint Channel Architecture Network (JCAN), can be used to detect and discriminate anomalous correlations between the sensors in multisensor data (such as variations in the phase relationships between the sensors), and that the other variant, the Independent Channel Architecture Network (ICAN), is more appropriate for detecting anomalies intrinsic to individual sensors. The paper explains and illustrates how the basis vectors of these networks may be adaptively trained on healthy data and how other network parameters may be optimized with respect to faulty data so as to give the greatest detection or discrimination capability for any given application.

ACKNOWLEDGEMENTS

This work was carried out as part of Technology Group 10 of the MoD Corporate Research Programme. The componential coding algorithm is patented intellectual property of QinetiQ.

REFERENCES

1 **Loughlin, P. J.** and **Bernard, G. D.** Cohen-posch (positive) time–frequency distributions and their application to machine vibration analysis. *Mech. Systems and Signal Processing*, 1997, **11**(4), 561–576.

2 **Gu, S., Ni, J.** and **Yuan, J.** Non-stationary signal analysis and transient machining process condition monitoring. *Int. J. Mach. Tools Mf.*, 2002, **42**, 41–51.

3 **Gu, F., Ball, A. D.** and **Rao, K. K.** Diesel injector dynamic modeling and estimation of injection parameters from impact response. Part 2: prediction of injection parameters from monitored vibration. *Proc. Instn Mech. Engrs, Part D: J. Automobile Engineering*, 1996, **210**(D4), 303–312.

4 **Wang, W. J.** and **McFadden, P. D.** Application of orthogonal wavelets to early gear damage detection. *Mech. Systems and Signal Processing*, 1995, **9**(5), 497–507.

5 **Ball, A. D., Gu, F.** and **Li, W.** The condition monitoring of diesel engines using acoustic measurements—Part 2: fault detection and diagnosis. SAE Technical Paper Series 2000-01-0368, 2000, Book SP-1501.

6 **Lin, J.** Feature extraction of machine sound using wavelets and its application in fault diagnosis. *NDT&E Int.*, 2001, **34**, 25–30.

7 **Howard, I. M.** Higher-order spectral techniques for machine vibration condition monitoring. *Proc. Instn Mech. Engrs, Part G: J. Aerospace Engineering*, 1997, **211**(G4), 211–219.

8 **Arthur, N.** and **Penman, J.** Induction machine condition monitoring with higher order spectra. *IEEE Trans. Ind. Electronics*, 2000, **47**(5), 1031–1041.

9 **Li, W., Gu, F., Ball, A. D., Leung, A. Y. T.** and **Phipps, C. E.** A study of the noise from diesel engines using independent component analysis. *Mech. Systems and Signal Processing*, 2001, **15**(6), 1165–1184.

10 **Murray, A.** and **Penman, J.** Extracting useful higher order features for condition monitoring using artificial neural networks. *IEEE Trans. Signal Processing*, 1997, **45**(11), 2821–2828.

11 **Zhang, S., Ganesan, R.** and **Xistris, G. D.** Self-organising neural networks for automated machinery monitoring systems. *Mech. Systems and Signal Processing*, 1996, **10**(5), 517–532.

12 **Gu, F., Jacob, P. J.** and **Ball, A. D.** Non-parametric models in the monitoring of engine performance and condition. Part 2: non-intrusive estimation of diesel engine cylinder pressure and its use in fault detection. *Proc. Instn Mech. Engrs, Part D: J. Automobile Engineering*, 1999, **213**(D2), 135–143.

13 **Luo, F.** and **Unbehauen, R.** *Applied Neural Networks for Signal Processing*, 1997 (Cambridge University Press, Cambridge).

14 **Payne, B. S., Gu, F., Webber, C. J. S.** and **Ball, A. D.** Componential coding in the condition monitoring of electrical machines. Part 2: application to a conventional machine and a novel machine. *Proc. Instn Mech. Engrs, Part C: J. Mechanical Engineering Science*, 2003, **217**(C8), 901–915.

15 **Kreyszig, E.** *Advanced Engineering Mathematics*, 1993, 7th edition (John Wiley, New York).

16 **Webb, A. R.** *Statistical Pattern Recognition*, 1999 (Arnold, London).

17 **Press, W. H.** *Numerical Recipes in C: The Art of Scientific Computing*, 1988 (Cambridge University Press, Cambridge).

18 **Webber, C. J. S.** Emergent componential coding of a handwriting-image database by neural self-organisation. In *Network: Computation in Neural Systems*, 1998, Vol. 9, pp. 433–447 (IOP Press, Bristol).

19 **Randall, R. B., Antoni, J.** and **Chobsaard, S.** The relationship between spectral correlation and envelope analysis of cyclostationary machine signals—application to bearing fault diagnostics. *Mech. Systems and Signal Processing*, September 2001, **15**(5), 945–962.

20 **Gu, F., Payne, B. S.** and **Ball, A. D.** Optimisation of a self-organising auto-encoder for the condition monitoring of a novel electric motor. University of Manchester Document MERG-0600, 2000.