On the Contractive Nature of Autoencoders: Application to Missing Sensor Restoration

Benjamin B. Thompson, Robert J. Marks II, and Mohamed A. El-Sharkawi

Computational Intelligence Applications (CIA) Laboratory, Department of Electrical Engineering, University of Washington, Seattle, WA 98195

Abstract

The neural network autoencoder is a useful tool for the restoration of missing sensors when enough known sensors with some relation to those missing are available. Through the idea of a contraction mapping, this paper provides some insight into the convergence of several iterative methods of sensor restoration using the autoencoder to some unique answer given a specific operating point (i.e., the known sensor values), regardless of how the missing sensor values are initialized

I. Introduction

Previous work has established the ability of the autoassociative neural network encoder (or simply "autoencoder") to aid in the restoration of sensor data which may be missing or corrupt, given some sort of correlation between the numeric outputs of the various sensors in a system. [1], [2] Narayanan *et al.* [1] describe a method by which the missing sensor data may be reconstructed using an iterative approach; in this paper we show that, under a set of conditions relating to the specific parameters of the neural network, we can provide a sufficient condition for the convergence of the iterative approach to sensor restoration. We approach this through the idea of a *contraction mapping.* Moreover, we will show compelling evidence that there exists a unique point of convergence for a fully trained autoencoder given an "operating point" defined by the set of known sensors, and that this convergence point should be reached regardless of how the missing sensors are initialized.

II. Contraction

A *contractive mapping* is defined [3],[4] as a mapping $O:X\rightarrow X$ on a complete metric space (X, d) in which, for any x and y in that space:

$$\begin{aligned} &d(Ox,Oy)\le k\cdot d(x,y) \\ &0\le k<1 \end{aligned} \tag{1}$$

or, more clearly, a contraction mapping is one in which the output distance between two points is less than the input distance. Now let us look at this property in $\Re^1$, where our metric is simply the Euclidean norm, and O is simply some functional mapping $f(x)$:

$$\left\|f(x)-f(y)\right\|\le k\cdot\left\|x-y\right\| \tag{2}$$

Now, suppose we replace y with $x+dx$, yielding, with some minor rearrangement:

$$\frac{\left\|f(x)-f(x+dx)\right\|}{\left\|dx\right\|}\le k \tag{3}$$

the limit of which as $dx\rightarrow 0$ gives us, as a less strict requirement for contraction

$$\left|\frac{df(x)}{dx}\right|<1 \tag{4}$$

This idea is demonstrated clearly in Figure 1. It would then be sufficient to show that, for some function f: $\Re^1\rightarrow\Re^1$, the derivative of f is less than unity for all x.

Consider the ***Banach Fixed-Point Theorem***: If f is a contractive mapping, then there exists a *unique* fixed point x_0 for which $f(x_0) = x_0$. Moreover, the sequence $\{x_n\}$, for which any element $x_{n+1} = f(x_n)$, converges, and that convergent point is x_o. With this theorem, it becomes much clearer how any contractive tendency of the autoencoder can help us show whether or not the sensor restoration process will converge to some unique value.

III. Missing Sensor Restoration with Autoencoders

There are three methods that we will examine for the restoration of missing sensors. The first is a simple application of alternating projections onto convex sets (POCS). The second and the third both employ search

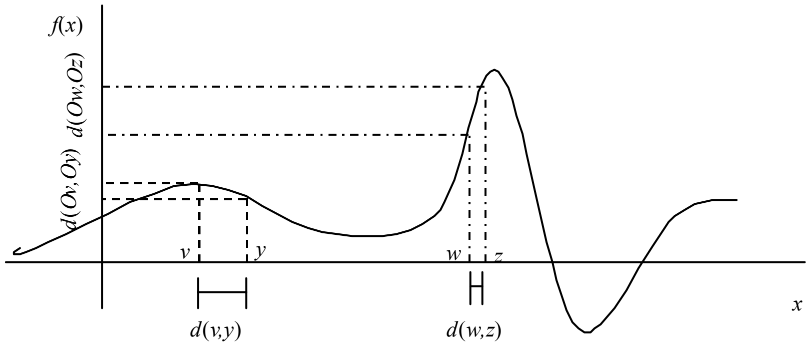

Figure 1 - For a function whose derivative is less than unity, the input distance of two points will always be greater than the output distance (the distances projected onto the horizontal and vertical axes, respectively). For a derivative greater than one, we achieve *expansion* rather than *contraction*. Note that points v and y exist where $|df(x)/dx|<1$, and w and z exist where $|df(x)/dx|>1$.

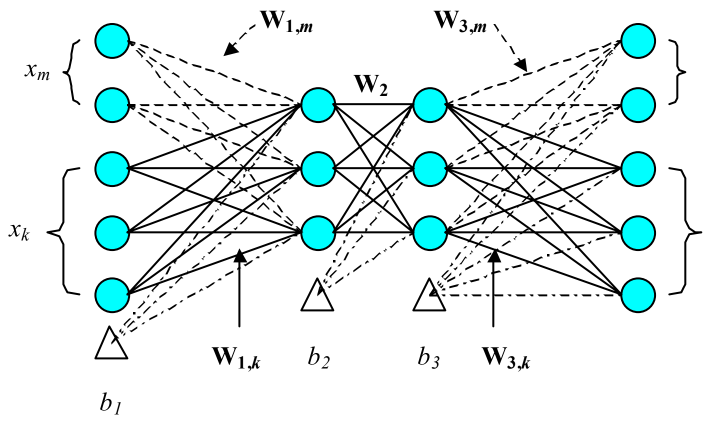

Figure 2 - a diagram of a generic 2-layer autoencoder, with appropriate labels, as described in the text.

techniques; the first involves simply minimizing the error between the missing sensor inputs and outputs on the autoencoder, while the second looks at the error between the entire input pattern and output pattern (both missing and known sensors) to achieve a final answer. The merits of each are discussed below. First, however, a few definitions are in order.

The input to the neural network is comprised of two concatenated vectors whose total dimension is N, the input dimension (and necessarily the output dimension as well). The first vector, x_k, can be thought of as the *operating point* of the restoration process, and is defined as the set of known sensor values for a given input pattern. The second vector is then x_m, the set of missing sensor values. Without loss of generality, let us formulate the input as some vector x:

$$\vec{x} \equiv \{x_{m1}, x_{m2}, \ldots, x_{mM}, x_{k1}, x_{k2}, \ldots, x_{kK}\}^T \qquad (5)$$

where M is the number of missing sensors, K is the number of known sensors, and of course $K+M = N$. Likewise, on the outputs, we have:

$$\vec{\hat{x}} \equiv \{\hat{x}_{m1}, \hat{x}_{m2}, \ldots, \hat{x}_{mM}, \hat{x}_{k1}, \hat{x}_{k2}, \ldots, \hat{x}_{kK}\}^T \qquad (6)$$

Figure 2 shows a single-hidden-layer autoencoder with the appropriate labels. Given that the encoder is trained as a feedforward multilayered perceptron (MLP) [3] we also have the following elements: $\mathbf{W}_{1,k}$ is the matrix of weights whose $(i, j)^{th}$ element is the weight connecting the i^{th} known sensor value to the j^{th} neuron in the first hidden layer; $\mathbf{W}_{1,m}$ is the corresponding matrix for the missing sensors, b_l is the vector of bias weights for the l^{th} layer; $\mathbf{W}_2$ is the weight matrix connecting the first hidden layer to the second; and finally, $\mathbf{W}_{3,k}$ and $\mathbf{W}_{3,m}$ are the counterparts to $\mathbf{W}_{1,k}$ and $\mathbf{W}_{1,m}$ on the output. Note that these can easily be extended for an encoder with more than a single hidden layer.

A. POCS

As described by Narayanan *et al.* [1], a straightforward method for missing sensor restoration using a trained autoencoder is the use of POCS [5] to achieve a convergent value. Under the assumption of convexity, our two sets are then a) the space defined as the output of the autoencoder, and b) the set of all input patterns to the neural network containing x_k, the known sensors, and an arbitrary x_m. While the second set is definitely convex, the first requires the assumption of convexity. By choosing some initial x_m to create an input vector x, we then obtain $\hat{x}$, the output of the autoencoder. This corresponds to a projection onto the first set, the operator for which we will denote P_1. We then change the outputs $\hat{x}_k$ to x_k to perform the projection onto the first set, denoted as P_2. If we alternate between these projections, under the assumption of convexity, the series will converge to an answer representing the intersection of the two sets. Thus, a single iteration of this process is defined as the successive application of P_1 and then P_2.

B. Unconstrained Search

Because of the potential lack of convexity and other performance issues, we are motivated to find a better method for discovering the true point of convergence. In this case, our iterative operator O_u is simply a single iteration of any search algorithm which seeks to minimize the error between the missing sensor values and the

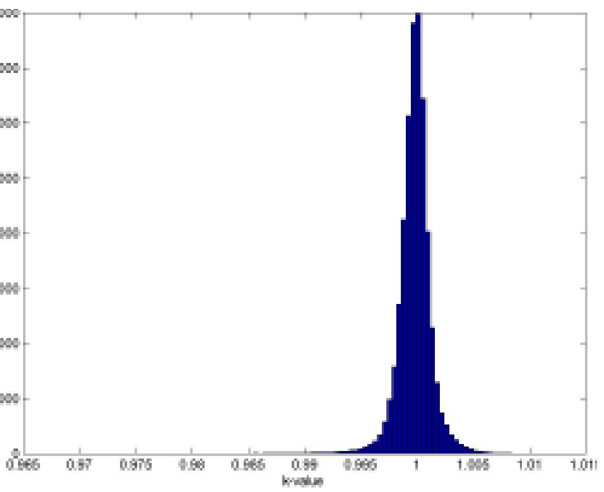

Figure 4 - a histogram of the *k*-values based on the training data. Note the dynamic range of the plot is [0.966, 1.015].

outputs of the autoencoder corresponding to those missing sensor values, or:

$$\arg\min_{x_m} \left\| x_m - \hat{x}_m \right\| \tag{7}$$

This method allows for greater refinement of the missing sensor values over the POCS method described above; moreover, it should be noted that, if the assumption of convexity were true, O_u and O_p would converge to the same value, assuming the two sets intersect.

C. Constrained Search

A notable shortcoming of both POCS and the unconstrained search is that neither one uses the information contained in $\vec{x}_k$ to better refine the final answer. Thus, we define a third operator, O_c, which is similar to O_u except that it corresponds to a search algorithm which seeks to minimize the *entire* output error of the autoencoder; namely:

$$\arg\min_{x_m} \left\| x - \hat{x} \right\| \tag{8}$$

recalling that x is a vector composed of x_m and x_k. This way, we actually ensure a smoother match between the input and the output, which can help eliminate spurious answers that, while minimizing the error between consecutive iterations on x_m, tend not to make sense in the context of the known sensors.

IV. Analysis Results

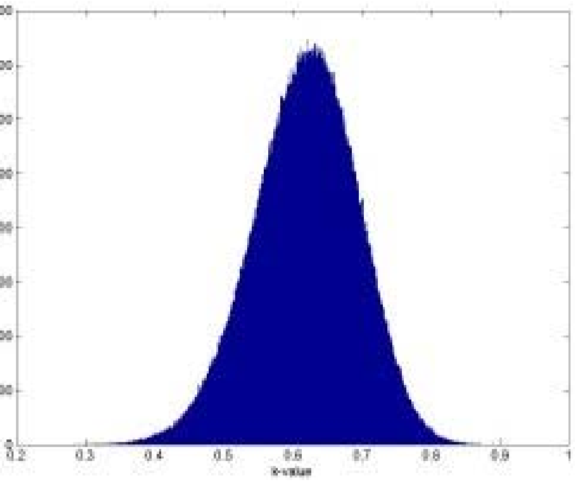

Figure 3 - a histogram of the *k*-values associated with our autoencoder as a whole, for a randomly generated data set. The scale of the *x*-axis is from 0.2 to 1. Note that the largest tail value is actually less than unity.

x be contractive. Recall that, by definition, a perfectly trained autoencoder is one for which we have the following relationship:

$$\begin{aligned} &O_{NN}\left(\vec{x}\right) = \vec{x} \\ &x \in C \end{aligned} \tag{9}$$

where O_{NN} is the neural network treated as an operator, and C is the set of all training data. Thus, except in the case of an autoencoder trained on a single pattern, a perfectly trained autoencoder *guarantees* that the Banach Fixed-Point Theorem *cannot* hold, and thus the operator is not contractive.

An operator is *nonexpansive* in (10) when, instead of $k<1$, we allow $k\leq 1$. A convex set orthogonal projection operator is nonexpansive [6] so this is a result to be expected from the autoencoder. If O is nonexpansive, the operation $x_{n+1} = O(x_n)$ will converge to a fixed point. This point, however, is not unique and is dependent on initialization .

A. Contraction of the Entire Autoencoder

While we have shown that the autoencoder itself is neither strictly contractive nor nonexpansive, it is informative to see how closely it approaches these conditions. As described in (3), there is a k-value associated with a set of two inputs and their corresponding outputs. If, for a very large set of input pairs, we can show that that k-value is less than or equal to one, then we have justification for treating the operator as *nearly* nonexpansive We examine the k-values from a specific example of a trained autoencoder. For the purposes of this

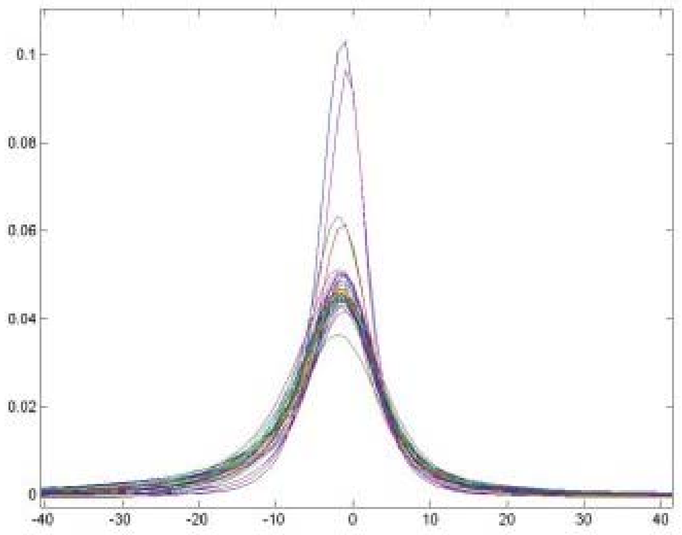

Figure 5 - average derivative of our overall restoration operator for a single missing sensor using randomly-initialized operating points x_k

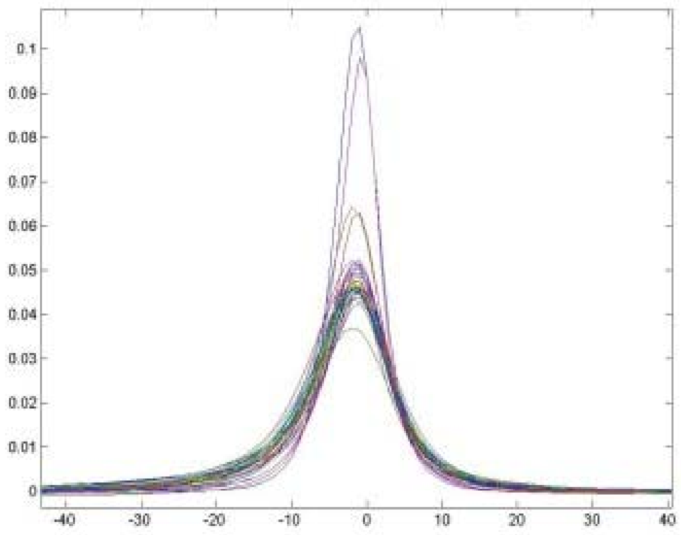

Figure 6 - same as Fig. 5 with training data used instead of randomly-initialized x_k

paper, we have trained an autoencoder on Mackey-Glass chaos, defined by the nonlinear difference equation [7]:

$$x[t] = \frac{A \cdot \theta^n \cdot x[t-\tau]}{\theta^n + x^n[t-\tau]} + (1-B) \cdot x[t] \qquad (10)$$

where A, B, n, θ, and τ are defined parameters, along with some $x[0]$ value. We generate a data set using this function, and train a 40-20-40 autoencoder using input patterns taken as consecutive 40-point "windows" of the data set. All data is normalized to the interval [0,1] before training.

The autoencoder thus trained, we then generate a large (on the order of 10^5 patterns) set of randomly generated input vectors (from a uniform distribution on [0,1]). We then select, at random, two different vectors from this set as vectors x and y as per equation (3). From this, we can calculate a corresponding k-value. With a sufficiently large number of these k-values, we can create estimate the probability density function of k, to examine how it behaves, particularly around the value of 1.

Figure 3 shows the result of this experiment. Clearly we have that, for a randomly generated data set, we never even approach the limits of being contractive; that is, our autoencoder behaves statistically as though it were in fact contractive. The largest k-value it achieves in this simulation, in fact, is 0.9206, well below the threshold beyond which it would no longer be contractive.

While this demonstrates the behavior of the autoencoder towards randomly generated data, we next perform a more interesting experiment. Given that the autoencoder is trained such that the output mirrors the input as closely as possible, we would expect the k-values for the actual training data to be very near 1 for each training pattern (recall that, for a perfectly trained autoencoder, k would be exactly unity for every single one). Thus, we have motivation to repeat the above experiment, replacing the randomly-generated data with the training data itself.

We see the result of this experiment in Figure 4. From this histogram, we have proof that our initial conjecture holds true even for this "imperfect" autoencoder – due to the k-values above 1, the operator is not strictly nonexpansive. However, it would clearly be fair to say that, from the evidence presented in this figure, our operator is *nearly* nonexpansive, since k never deviates from unity by more than 0.01.

B. Contraction of Subsets of the Autoencoder

At this point, we then want to show that, while the autoencoder *as a whole* is neither strictly contractive nor nonexapnsive, the autoencoder *at some operating point* may be contractive as it operates on a subset of the input vector; namely, x_m. At this point, it is useful to write out the functional form of the neural network as an operator. Let us formulate this for a two-hidden layer neural network as described in Figure 2, although it can easily be generalized for greater or fewer dimensions:

$$\vec{f}(\vec{x}_m, \vec{x}_k) = \begin{bmatrix} \hat{\vec{x}}_m \\ \hat{\vec{x}}_k \end{bmatrix} = \boldsymbol{\sigma}\Big(\mathbf{W}_{3,m} \cdot \boldsymbol{\sigma}\Big(\mathbf{W}_2 \cdot \boldsymbol{\sigma}\Big(\mathbf{W}_{1,m} \cdot \vec{x}_m + \mathbf{W}_{1,k} \cdot \vec{x}_k + \vec{b}_1\Big) + \vec{b}_2\Big) + \vec{b}_3\Big) \qquad (11)$$

where $\boldsymbol{\sigma}$ is a vector operator that imposes a sigmoid nonlinearity on each element of the applied vector, and all other parameters are as described above. This function

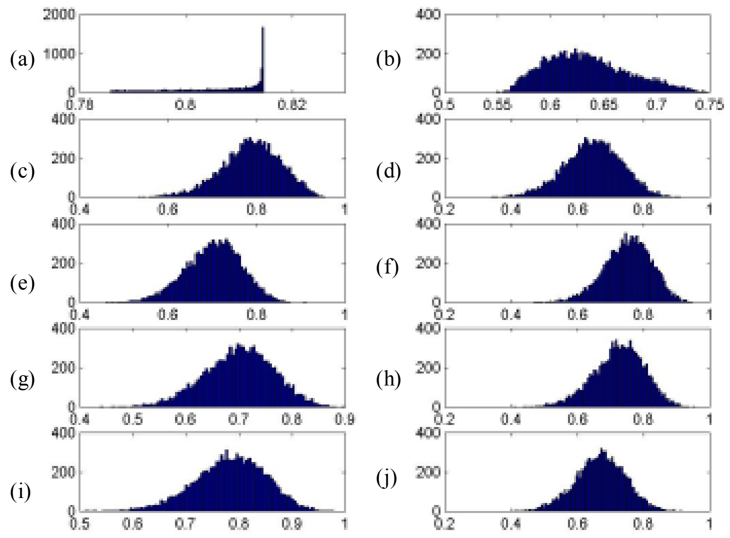

Figure 7 - histograms of *k*-values various combinations of missing sensors. Figs. (a)-(j) correspond to 1, 5, 9, 13, 17, 21, 25, 29, 33, and 37 missing sensors, respectively. The specific missing sensors were chosen at raondom, and the operating point, selected from the training data, was the same for each case.

represents the functional form of our operator O_1, as described above. Likewise, we can define out operator O_2 as:

$$\vec{g}\left(\vec{\hat{x}}, \vec{x}_k\right) = \begin{bmatrix} \vec{\hat{x}}_m \\ \vec{x}_k \end{bmatrix} = \mathbf{T}\cdot\vec{\hat{x}} + \mathbf{B}\cdot\vec{x}_k \qquad (12)$$

where $\mathbf{T}$ is an $N\times N$ matrix in which:

$$\mathbf{T}_{i,j} = \begin{cases} 1 & i = j \le M \\ 0 & else \end{cases} \qquad (13)$$

and $\mathbf{B}$ is an $N\times K$ matrix defined as:

$$\mathbf{B} = \begin{bmatrix} \mathbf{0}_{M\times K} \\ \mathbf{I}_K \end{bmatrix} \qquad (14)$$

Thus, combining equations (11) and (12) yields our operator O_p:

$$\vec{O}_p\left(\vec{x}_m, \vec{x}_k\right) = \mathbf{T}\cdot f\left(\vec{x}_m, \vec{x}_k\right) + \mathbf{B}\cdot\vec{x}_k \qquad (15)$$

With this, we have a framework for which we can examine the contractiveness of the entire process. Specifically, we can look at the case in which only one sensor is missing. If this is the case, then x_m is a scalar, and equation (4) can be applied.

Explicitly calculating the derivative of O_p is a cumbersome task, particularly if x_m is not scalar. For the purposes of this paper, we again apply a randomly initialized simulation to show that the derivative *tends* to be less than one for various x_k operating points. We perform this experiment using the same Mackey-Glass autoencoder as used above. First, we examine all forty "sensor" values by randomly generating x_k (as a vector of uniform random variables on [0,1]) and calculating the corresponding derivatives. Figure 5 shows the overlaid plot of the derivatives for each of the forty sensors. Each curve represents an average over multiple realizations of x_k. The maximum standard deviation at any point for any of the sets of curves was as small as 0.0042, giving us a great deal of confidence that, for random x_k, and a single missing sensor, we will always converge to a unique answer, since the less-than-unity derivative implies contractiveness of the sub-operator as it acts around a fixed point.

Next, we perform the same experiment, again replacing the randomly initialized portion (in this case, the value of the fixed point x_k) with the actual training data. Figure 6 displays the results clearly, in a form identical to Figure 5. In this case, the maximum standard deviation for any value of x_m over any set of the curves was 0.0036, which gives us even greater confidence of our conclusion. Comparing Fig. 5 to Fig. 6, we see that they are almost completely indistinguishable. No difference is graphically discernable. This gives us substantial reason to believe that the derivative is largely insensitive to the actual value of the operating point (as long as the operating point is within the unit-cube in K dimensions – which is reasonable since it is possible to *define* the valid range of sensors to be within that limit).

Finally, we perform an experiment to demonstrate the contractive characteristics of situations in which more than a single sensor are missing. For this, we calculate a series of *k*-values as above, the exception being that some fixed-point x_k is chosen, and the remaining sensors x_m are randomly initialized as above. We then perform this for a variety of missing-sensor configurations (obviously, all the possible permutations would take a prohibitive amount of time to calculate even for a relatively small autoencoder, and even more so for our situation using the Mackey-Glass autoencoder).

Figure 7 displays these results, for 10 different cases corresponding to 1, 5, 9, 13, 17, 21, 25, 29, 33, and 37 missing sensors. The specific sensors in each case were selected at random from the 40 possible sensors. We note that, in every single plot, we are well below the unity threshold required for contraction. Moreover, it is interesting to note that the upper limit of the *k*-value seems to approach unity gradually as the number of missing sensors increases (implying that the operation is "more contractive" for fewer missing sensors).

.

V. Conclusions

By demonstrating the contractive nature of the autoencoder as a method for restoring missing sensors, we have given compelling evidence that such iterative procedures will, for the case examined, converge to a unique answer dependent only on the neural network autoencoder itself, and the operating point (the known sensor values) about which the process is implemented. We have shown that the autoencoder itself is *nearly* nonexpansive to most types of data, the marginal exception being the training data itself. Finally, we have provided reason to believe that, the fewer sensors that are missing, the more likely the autoencoder-method of restoring missing sensors is to have such a unique value of convergence.

VI. References

[1] Narayanan, S., R.J. Marks II , J. L. Vian, J.J. Choi, M.A. El-Sharkawi & B. B. Thompson, "Set Constraint Discovery: Missing Sensor Data Restoration Using Auto-Associative Regression Machines", Proceedings of the 2002 International Joint Conference on Neural Networks, 2002 IEEE World Congress on Computational Intelligence, May12-17, 2002, Honolulu, pp. 2872-2877.

[2] Reed, R. D. and R.J. Marks II, *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks*, MIT Press, Cambridge, MA, 1999.

[3] Naylor, A. W., and G. R. Sell *Linear Operator Theory in Engineering and Science*, Springer, New York City, NY, 1982.

[4] Luenberger, D. G., *Optimization by Vector Space Methods,* John Wiley & Sons, April 1997

[5] Marks, R.J. II, "Alternating Projections onto Convex Sets", in *Deconvolution of Images and Spectra*, edited by Peter A. Jansson, (Academic Press, San Diego, 1997), pp.476-501.

[6] Goldburg, M.H. and R.J. Marks II, "Signal synthesis in the presence of an inconsistent set of constraints", IEEE Transactions on Circuits and Systems, vol. CAS-32 pp. 647-663 (1985).

[7] Glass, L. and M. C. Mackey, *From Clocks to Chaos, The Rhythms of Life,* Princeton University Press, Princeton, NJ, 1988.