Neural Networks 37 (2013) 132–140

Contents lists available at SciVerse ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Noise-enhanced clustering and competitive learning algorithms

Osonde Osoba, Bart Kosko *

Department of Electrical Engineering, Signal and Image Processing Institute, University of Southern California, Los Angeles, CA 90089-2564, USA

ARTICLE INFO

Keywords:
Noise injection
Stochastic resonance
Clustering
EM algorithm
Competitive learning
k-means clustering

ABSTRACT

Noise can provably speed up convergence in many centroid-based clustering algorithms. This includes the popular *k*-means clustering algorithm. The clustering noise benefit follows from the general noise benefit for the expectation–maximization algorithm because many clustering algorithms are special cases of the expectation–maximization algorithm. Simulations show that noise also speeds up convergence in stochastic unsupervised competitive learning, supervised competitive learning, and differential competitive learning.

1. Noise benefits in clustering

We show below that noise can speed convergence in many clustering algorithms. This noise benefit is a form of *stochastic resonance* (Benzi, Parisi, Sutera, & Vulpiani, 1983; Brey & Prados, 1996; Bulsara, Boss, & Jacobs, 1989; Bulsara & Gammaitoni, 1996; Bulsara & Zador, 1996; Castro & Sauer, 1997; Chapeau-Blondeau, 1997; Chapeau-Blondeau & Rousseau, 2004; Chen, Varshney, Kay, & Michels, 2009; Collins, Chow, & Imhoff, 1995; Cordo et al., 1996; Gammaitoni, Hänggi, Jung, & Marchesoni, 1998; Grifoni & Hänggi, 1996; Guerra et al., 2010; Hänggi, 2002; Kosko, 2006; Kosko & Mitaim, 2003; Lee, Liu, Zhou, & Kosko, 2006; Löcher et al., 1999; McDonnell, Stocks, Pearce, & Abbott, 2008; Mitaim & Kosko, 1998; Moss, Bulsara, & Shlesinger, 1993; Patel & Kosko, 2008, 2009, 2010, 2011; Wilde & Kosko, 2009): small amounts of noise improve a nonlinear system's performance while too much noise harms it. This noise benefit applies to clustering because many of these algorithms are special cases of the expectation–maximization (EM) algorithm (Celeux & Govaert, 1992; Xu & Wunsch, 2005). We recently showed that an appropriately noisy EM algorithm converges more quickly on average than does a noiseless EM algorithm (Osoba, Mitaim, & Kosko, 2011, in review). The Noisy Expectation Maximization (NEM) Theorem 1 in Section 2.1 restates this noise benefit for the EM algorithm.

Fig. 1 shows a simulation instance of the corollary noise benefit of the NEM Theorem for a two-dimensional Gaussian mixture model with three Gaussian data clusters. Theorem 3 below states that such a noise benefit will occur. Each point on the curve reports how much two classifiers disagree on the same data set. The first classifier is the EM-classifier with fully converged EM-parameters. This is the reference classifier. The second classifier is the same EM-classifier with only partially converged EM-parameters. The two classifiers agree eventually if we let the second classifier's EM-parameters converge. But the figure shows that they agree faster with some noise than with no noise.

We call the normalized number of disagreements the *misclassification rate*. The misclassification rate falls as the Gaussian noise power increases from zero. It reaches a minimum for additive white noise with standard deviation 0.3. More energetic noise does not reduce misclassification rates beyond this point. The optimal noise reduces misclassification by almost 30%.

Fig. 2 shows a similar noise benefit in the simpler *k*-means clustering algorithm on 3-dimensional Gaussian mixture data. The *k*-means algorithm is a special case of the EM algorithm as we show below in Theorem 2. So the EM noise benefit extends to the *k*-means algorithm. The figure plots the average convergence time for noise-injected *k*-means routines at different initial noise levels. The figure shows an instance where decaying noise helps the algorithm converge about 22% faster than without noise.

Clustering algorithms divide data sets into clusters based on similarity measures (Duda, Hart, & Stork, 2001; Kohonen, 2001; Xu & Wunsch, 2005, 2009). The similarity measure attempts to quantify how samples differ statistically. Many algorithms use the Euclidean distance or Mahalanobis similarity measure. Clustering algorithms assign similar samples to the same cluster. Centroid-based clustering algorithms assign samples to the cluster with the closest centroid $\mu_1, \ldots, \mu_k$.

This clustering framework attempts to solve an optimization problem. The algorithms define data clusters that minimize the total *within-cluster* deviation from the centroids. Suppose y_i are samples of a data set on a sample space D. Centroid-based clustering partitions D into the k decision classes $D_1, \ldots, D_k$ of D. The algorithms look for optimal cluster parameters that minimize an objective function. The *k*-means clustering method (MacQueen, 1967) minimizes the total sum of squared Euclidean within-cluster

* Corresponding author.
E-mail address: kosko@usc.edu (B. Kosko).

0893-6080/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2012.09.012

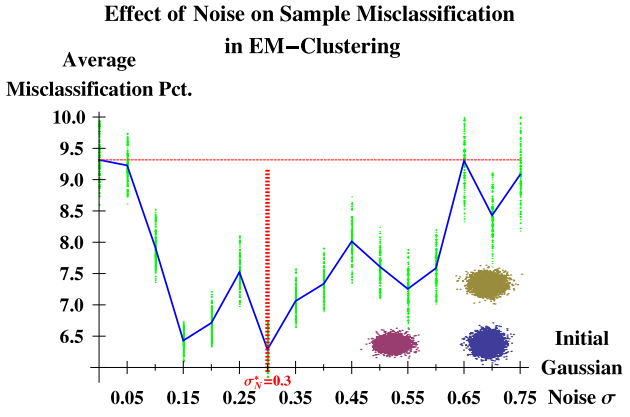

Fig. 1. Noise benefit based on the misclassification rate for the Noisy Expectation–Maximization (NEM) clustering procedure on a 2-D Gaussian mixture model with three Gaussian data clusters (inset) where each has a different covariance matrix. The plot shows that the misclassification rate falls as the additive noise power increases. The classification error rises if the noise power increases too much. The misclassification rate measures the mismatch between a NEM classifier with unconverged parameters Θ_k and the optimal NEM classifier with converged parameters Θ_*. The unconverged NEM classifier's NEM procedure stops a quarter of the way to convergence. The dashed horizontal line indicates the misclassification rate for regular EM classification without noise. The red dashed vertical line shows the optimum noise standard deviation for NEM classification. The optimum noise has a standard deviation of 0.3.

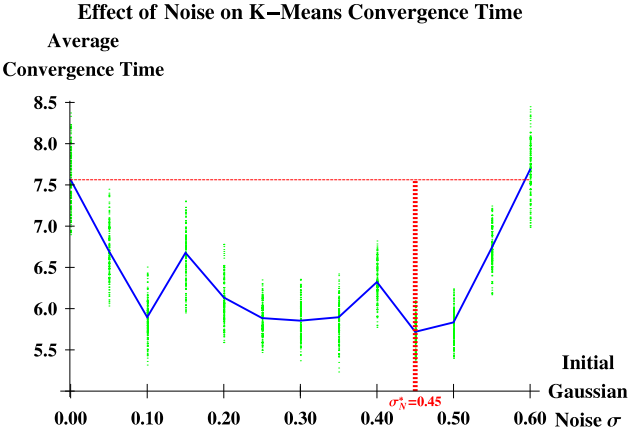

Fig. 2. Noise benefit in k-means clustering procedure on 2500 samples of a 3-D Gaussian mixture model with four clusters. The plot shows that the convergence time falls as additive white Gaussian noise power increases. The noise decays at an inverse square rate with each iteration. Convergence time rises if the noise power increases too much. The dashed horizontal line indicates the convergence time for regular k-means clustering without noise. The red dashed vertical line shows the optimum noise standard deviation for noisy k-means clustering. The optimum noise has a standard deviation of 0.45: the convergence time falls by about 22%.

distances (Celeux & Govaert, 1992; Xu & Wunsch, 2005):

$$\sum_{j=1}^{K}\sum_{i=1}^{N}\|y_i-\mu_j\|^2\mathbb{I}_{D_j}(y_i) \tag{1}$$

where $\mathbb{I}_{D_j}$ is the indicator function that indicates the presence or absence of pattern y in D_j:

$$\mathbb{I}_{D_j}(y)=\begin{cases}1 & \text{if } y\in D_j\\ 0 & \text{if } y\notin D_j.\end{cases} \tag{2}$$

There are many approaches to clustering (Duda et al., 2001; Xu & Wunsch, 2005). Cluster algorithms come from fields that include nonlinear optimization, probabilistic clustering, neural networks-based clustering, fuzzy clustering (Bezdek, Ehrlich, & Fill, 1984; Höppner, Klawonn, & Kruse, 1999), graph-theoretic clustering (Cherng & Lo, 2001; Hartuv & Shamir, 2000), agglomerative clustering (Zhang, Ramakrishnan, & Livny, 1996), and bio-mimetic clustering (Fogel, 1994; Hall, Ozyurt, & Bezdek, 1999).

Iterative maximum likelihood clustering algorithms can benefit from noise injection. This noise benefit derives from the application of the Noisy Expectation Maximization (NEM) theorem to the Expectation–Maximization (EM) clustering framework. The next section reviews the recent NEM Theorem (Osoba et al., 2011, in review) and applies it to clustering algorithms.

2. The noisy EM algorithm

The regular Expectation–Maximization (EM) algorithm is a maximum likelihood procedure for corrupted or missing data. Corruption can refer to the mixing of subpopulations in clustering applications. The procedure seeks a maximizer θ_* of the likelihood function:

$$\Theta_* = \operatorname*{argmax}_{\Theta} \ln f(y|\Theta). \tag{3}$$

The EM algorithm iterates an E-step and an M-step:

EM Algorithm

E–Step: $Q(\theta|\theta_k) \leftarrow \mathbb{E}_{Z|Y,\theta_k}[f(\mathbf{y}, z|\theta)]$

M–Step: $\theta_{k+1} \leftarrow \operatorname{argmax}_\theta \{Q(\theta|\theta_k)\}$.

2.1. NEM theorem

The Noisy Expectation–Maximization (NEM) (Osoba et al., 2011, in review) Theorem states a general sufficient condition when noise speeds up the EM algorithm's convergence to the local optimum. The NEM Theorem uses the following notation. The noise random variable N has pdf $f(n|y)$. So the noise N can depend on the data Y. $\{\theta_k\}$ is a sequence of EM estimates for θ. $\theta_* = \lim_{k\to\infty}\theta_k$ is the converged EM estimate for θ. Define the noisy Q function $Q_N(\theta|\theta_k) = \mathbb{E}_{Z|Y,\theta_k}[\ln f(y+N, z|\theta)]$. Assume that the differential entropy of all random variables is finite. Assume also that the additive noise keeps the data in the likelihood function's support. Then we can state the NEM theorem.

Theorem 1 (*Noisy Expectation–Maximization (NEM)*)**.** *The EM estimation iteration noise benefit*

$$Q(\theta_*|\theta_*) - Q(\theta_k|\theta_*) \geq Q(\theta_*|\theta_*) - Q_N(\theta_k|\theta_*) \tag{4}$$

or equivalently

$$Q_N(\theta_k|\theta_*) \geq Q(\theta_k|\theta_*) \tag{5}$$

holds if the following positivity condition holds on average:

$$\mathbb{E}_{Y,Z,N|\theta^*}\left[\ln\left(\frac{f(Y+N, Z|\theta_k)}{f(Y, Z|\theta_k)}\right)\right] \geq 0. \tag{6}$$

The NEM Theorem states that a suitably noisy EM algorithm estimates the EM estimate θ_* in fewer steps on average than does the corresponding noiseless EM algorithm.

The Gaussian mixture EM model in the next section greatly simplifies the positivity condition in (6). The model satisfies the positivity condition (6) when the additive noise samples $n = (n_1, \ldots, n_d)$ satisfy the following algebraic condition (Osoba et al., 2011, in review):

$$n_i\left[n_i - 2n_i\left(\mu_{j_i} - y_i\right)\right] \leq 0 \quad \text{for all } j. \tag{7}$$

This condition applies to the variance update in the EM algorithm. It needs the current estimate of the centroids μ_j. The NEM

algorithm also anneals the additive noise by multiplying the noise power σ_N by constants that decay with the iteration count. We found that the best application of the algorithm uses inverse-square decaying constants (Osoba et al., in review):

$$s[k] = k^{-2} \tag{8}$$

where $s[k]$ scales the noise N by a decay factor of k^{-2} on the kth iteration. The annealed noise $N_k = k^{-2}N$ must still satisfy the NEM condition for the model. Then the decay factor $s[k]$ reduces the NEM estimator's jitter around its final value. All noise-injection simulations used this annealing cooling schedule to gradually reduce the noise variance.

EM clustering methods attempt to learn mixture model parameters and then classify samples based on the optimal pdf. EM clustering estimates the most likely mixture distribution parameters. These maximum likelihood parameters define a pdf for sample classification. A common mixture model in EM clustering methods is the Gaussian mixture model (GMM) that we discuss next.

2.2. Gaussian mixture models

Gaussian mixture models sample from a convex combination of a finite set of Gaussian sub-populations. K is now the number of sub-populations. The GMM population parameters are the mixing proportions (convex coefficients) $\alpha_1, \ldots, \alpha_K$ and the pdf parameters $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K$ for each population. Bayes' theorem gives the conditional pdf for the E-step.

The mixture model uses the following notation and definitions. Y is the observed mixed random variable. Z is the latent population index random variable. The joint pdf $f(y, z|\Theta)$ is

$$f(y, z|\Theta) = \sum_{j=1}^{K} \alpha_j \delta[z - j] f(y|j, \theta_j) \tag{9}$$

where
$$f(y|\Theta) = \sum_{j=1}^{K} \alpha_j f(y|j, \theta_j), \tag{10}$$

$$\delta[z - j] = \begin{cases} 1 & \text{if } z = j \\ 0 & \text{if } z \neq j \end{cases}, \tag{11}$$

and
$$p_Z(j|y, \Theta) = \frac{\alpha_j f(y|Z = j, \theta_j)}{f(y|\Theta)} \tag{12}$$

for
$$\Theta = \{\alpha_1, \ldots, \alpha_K, \theta_1, \ldots, \theta_K\}. \tag{13}$$

We can rewrite the joint pdf in exponential form as follows:

$$f(y, z|\Theta) = \exp\left[\sum_{j=1}^{K} \left[\ln(\alpha_j) + \ln f(y|j, \theta_j)\right] \delta[z - j]\right], \tag{14}$$

$$\ln f(y, z|\Theta) = \sum_{j=1}^{K} \delta[z - j] \ln[\alpha_j f(y|j, \theta_j)], \tag{15}$$

$$Q(\Theta|\Theta(t)) = E_{Z|y,\Theta_k}[\ln f(y, Z|\Theta)] \tag{16}$$

$$= \sum_{z=1}^{K} \left(\sum_{j=1}^{K} \delta[z - j] \ln[\alpha_j f(y|j, \theta_j)]\right) \times p_Z(z|y, \Theta(t)) \tag{17}$$

$$= \sum_{j=1}^{K} \ln[\alpha_j f(y|j, \theta_j)] p_Z(j|y, \Theta(t)). \tag{18}$$

Eq. (18) states the E-step for the mixture model. The Gaussian mixture model (GMM) uses the above model with Gaussian subpopulation pdfs for $f(y|j, \theta_j)$.

Suppose there are N data samples of the GMM distributions. The EM algorithm estimates the mixing probabilities α_j, the subpopulation means μ_j, and the subpopulation covariance Σ_j. The current estimate of the GMM parameters is $\Theta(t) = \{\alpha_1(t), \ldots, \alpha_K(t), \mu_1(t), \ldots, \mu_K(t), \Sigma_1(t), \ldots, \Sigma_K(t)\}$. The iterations of the GMM–EM reduce to the following update equations:

$$\alpha_j(t + 1) = \frac{1}{N} \sum_{i=1}^{N} p_Z(j|y_i, \Theta(t)) \tag{19}$$

$$\mu_j(t + 1) = \frac{\sum_{i=1}^{N} p_Z(j|y_i, \Theta(t)) y_i}{\sum_{i=1}^{N} p_Z(j|y_i, \Theta(t))} \tag{20}$$

$$\Sigma_j(t + 1) = \frac{\sum_{i=1}^{N} p_Z(j|y_i, \Theta(t))(y_i - \mu_j(t))(y_i - \mu_j(t))^T}{\sum_{i=1}^{N} p_Z(j|y_i, \Theta(t))}. \tag{21}$$

These equations update the parameters α_j, μ_j, and Σ_j with coordinate values that maximize the Q function in (18) (Duda et al., 2001). The updates combine both the E-steps and M-steps of the EM procedure.

2.3. EM clustering

EM clustering uses the membership probability density function $p_Z(j|y, \Theta_{EM})$ as a maximum *a posteriori* classifier for each sample y. The classifier assigns y to the jth cluster if $p_Z(j|y, \Theta_{EM}) \geq p_Z(k|y, \Theta_{EM})$ for all $k \neq j$. Thus

$$EMclass(y) = \operatorname*{argmax}_j p_Z(j|y, \Theta_{EM}). \tag{22}$$

This is the *naive Bayes classifier* (Domingos & Pazzani, 1997; Rish, 2001) based on the EM-optimal GMM parameters for the data. NEM clustering uses the same classifier but with the NEM-optimal GMM parameters for the data:

$$NEMclass(y) = \operatorname*{argmax}_j p_Z(j|y, \Theta_{NEM}). \tag{23}$$

2.4. k-means clustering as a GMM–EM procedure

k-means clustering is a non-parametric procedure for partitioning data samples into clusters (MacQueen, 1967; Xu & Wunsch, 2005). Suppose the data space $\mathbb{R}^d$ has K centroids $\mu_1, \ldots, \mu_K$. The procedure tries to find K partitions $D_1, \ldots, D_K$ with centroids $\mu_1, \ldots, \mu_K$ that minimize the within-cluster Euclidean distance from the cluster centroids:

$$\operatorname*{argmin}_{D_1, \ldots D_K} \sum_{j=1}^{K} \sum_{i=1}^{N} \|y_i - \mu_j\|^2 \mathbb{I}_{D_j}(y_i) \tag{24}$$

for N pattern samples $y_1, \ldots, y_N$. The class indicator functions $\mathbb{I}_{D_1}, \ldots, \mathbb{I}_{D_K}$ arise from the nearest-neighbor classification in (26) below. Each indicator function $\mathbb{I}_{D_j}$ indicates the presence or absence of pattern y in D_j:

$$\mathbb{I}_{D_j}(y) = \begin{cases} 1 & \text{if } y \in D_j \\ 0 & \text{if } y \notin D_j. \end{cases} \tag{25}$$

The k-means procedure finds local optima for this objective function. k-means clustering works in the following two steps:

k-Means Clustering Algorithm

Assign Samples to Partitions:

$$y_i \in D_j(t) \text{ if } \|y_i - \mu_j(t)\| \leq \|y_i - \mu_k(t)\| \quad k \neq j \tag{26}$$

Update Centroids:

$$\mu_j(t + 1) = \frac{1}{|D_j(t)|} \sum_{i=1}^{N} y_i \mathbb{I}_{D_j(t)}(y_i). \tag{27}$$

k-means clustering is a special case of the GMM–EM model (Celeux & Govaert, 1992; Hathaway, 1986). The key to this subsumption is the "degree of membership" function or "cluster-membership measure" $m(j|y)$ (Hamerly & Elkan, 2002; Xu & Wunsch, 2010). It is a fuzzy measure of how much the sample y_i belongs to the jth subpopulation or cluster. The GMM–EM model uses Bayes' theorem to derive a soft cluster-membership function:

$$m(j|y) = p_Z(j|y, \Theta) = \frac{\alpha_j f(y|Z = j, \theta_j)}{f(y|\Theta)}. \tag{28}$$

k-means clustering assumes a hard cluster-membership (Hamerly & Elkan, 2002; Kearns, Mansour, & Ng, 1997; Xu & Wunsch, 2010):

$$m(j|y) = \mathbb{I}_{D_j}(y) \tag{29}$$

where D_j is the partition region whose centroid is closest to y. The k-means assignment step redefines the cluster regions D_j to modify this membership function. The procedure does not estimate the covariance matrices in the GMM–EM formulation.

Theorem 2 (*The Expectation–Maximization Algorithm Subsumes k-Means Clustering*)**.** *Suppose that the subpopulations have known spherical covariance matrices Σ_j and known mixing proportions α_j. Suppose further that the cluster-membership function is hard:*

$$m(j|y) = \mathbb{I}_{D_j}(y). \tag{30}$$

Then GMM–EM reduces to k-means clustering:

$$\frac{\sum_{i=1}^{N} p_Z(j|y_i, \Theta(t))y_i}{\sum_{i=1}^{N} p_Z(j|y_i, \Theta(t))} = \frac{1}{|D_j(t)|} \sum_{i=1}^{N} y_i \mathbb{I}_{D_j(t)}(y_i). \tag{31}$$

Proof. The covariance matrices Σ_j and mixing proportions α_j are constant. So the update Eqs. (19) and (21) do not apply in the GMM–EM procedure. The mean (or centroid) update equation in the GMM–EM procedure becomes

$$\mu_j(t+1) = \frac{\sum_{i=1}^{N} p_Z(j|y_i, \Theta(t))y_i}{\sum_{i=1}^{N} p_Z(j|y_i, \Theta(t))}. \tag{32}$$

The hard cluster-membership function

$$m_t(j|y) = \mathbb{I}_{D_j(t)}(y) \tag{33}$$

changes the tth iteration's mean update to

$$\mu_j(t+1) = \frac{\sum_{i=1}^{N} y_i m_t(j|y_i)}{\sum_{i=1}^{N} m_t(j|y_i)}. \tag{34}$$

The sum of the hard cluster-membership function reduces to

$$\sum_{i=1}^{N} m_t(j|y_i) = N_j = |D_j(t)| \tag{35}$$

where N_j is the number of samples in the jth partition. Thus the mean update is

$$\mu_j(t+1) = \frac{1}{|D_j(t)|} \sum_{i=1}^{N} y_i \mathbb{I}_{D_j(t)}(y_i). \tag{36}$$

Then the EM mean update equals the k-means centroid update:

$$\frac{\sum_{i=1}^{N} p_Z(j|y_i, \Theta(t))y_i}{\sum_{i=1}^{N} p_Z(j|y_i, \Theta(t))} = \frac{1}{|D_j(t)|} \sum_{i=1}^{N} y_i \mathbb{I}_{D_j(t)}(y_i). \quad \square \tag{37}$$

The known diagonal covariance matrices Σ_j and mixing proportions α_j can arise from prior knowledge or previous optimizations. Estimates of the mixing proportions (19) get collateral updates as learning changes the size of the clusters.

Approximately hard cluster membership can occur in the regular EM algorithm when the subpopulations are well separated. An EM-optimal parameter estimate Θ^* will result in very low posterior probabilities $p_Z(j|y, \Theta^*)$ if y is not in the jth cluster. The posterior probability is close to one for the correct cluster. Celeux and Govaert proved a similar result by showing an equivalence between the objective functions for EM and k-means clustering (Celeux & Govaert, 1992; Xu & Wunsch, 2005). Noise-injection simulations confirmed the predicted noise benefit in the k-means clustering algorithm.

2.5. k-means clustering and adaptive resonance theory

k-means clustering resembles Adaptive Resonance Theory (ART) (Carpenter & Grossberg, 1987; Kosko, 1991a; Xu & Wunsch, 2005). And so ART should also benefit from noise. k-means clustering learns clusters from input data without supervision. ART performs similar unsupervised learning on input data using neural circuits.

ART uses interactions between two fields of neurons: the comparison neuron field (or bottom-up activation) and the recognition neuron field (or top-down activation). The comparison field matches against the input data. The recognition field forms internal representations of learned categories. ART uses bidirectional "resonance" as a substitute for supervision. Resonance refers to the coherence between recognition and comparison neuron fields. The system is stable when the input signals match the recognition field categories. But the ART system can learn a new pattern or update an existing category if the input signal fails to match any recognition category to within a specified level of "vigilance" or degree of match.

ART systems are more flexible than regular k-means systems because ART systems do not need a pre-specified cluster count k to learn the data clusters. ART systems can also update the cluster count on the fly if the input data characteristics change. Extensions to the ART framework include ARTMAP (Carpenter, Grossberg, & Reynolds, 1991a) for supervised classification learning and Fuzzy ART for fuzzy clustering (Carpenter, Grossberg, & Rosen, 1991b). An open research question is whether NEM-like noise injection will provably benefit ART systems.

3. The clustering noise benefit theorem

The noise benefit of the NEM Theorem implies that noise can enhance EM-clustering. The next theorem shows that the noise benefits of the NEM Theorem extend to EM-clustering. The noise benefit occurs in misclassification relative to the EM-optimal classifier. Noise also benefits the k-means procedure as Fig. 2 shows since k-means is an EM-procedure. The theorem uses the following notation:

- $class_{opt}(Y) = \operatorname{argmax} \; p_Z(j|Y, \Theta_*)$: EM-optimal classifier. It uses the optimal model parameters Θ_*
- $P_M[k] = P\left(EMclass_k(Y) \neq class_{opt}(Y)\right)$: Probability of EM-clustering misclassification relative to $class_{opt}$ using kth iteration parameters
- $P_{M_N}[k] = P\left(NEMclass_k(Y) \neq class_{opt}(Y)\right)$: Probability of NEM-clustering misclassification relative to $class_{opt}$ using kth iteration parameters.

Theorem 3 (*Clustering Noise Benefit Theorem*)**.** *Consider the NEM and EM iterations at the k*th *step. Then the NEM misclassification probability* $P_{M_N}[k]$ *is less than the noise-free EM misclassification probability* $P_M[k]$*:*

$$P_{M_N}[k] \leq P_M[k] \tag{38}$$

when the additive noise N in the NEM-clustering procedure satisfies the NEM Theorem condition from (6):

$$\mathbb{E}_{Y,Z,N|\theta^*}\left[\ln\left(\frac{f(Y+N,Z|\theta_k)}{f(Y,Z|\theta_k)}\right)\right] \geq 0. \tag{39}$$

This positivity condition (39) in the GMM-NEM model reduces to the simple algebraic condition (7) Osoba et al. (2011, in review) for each coordinate i:

$$n_i\left[n_i - 2n_i\left(\mu_{j_i} - y_i\right)\right] \leq 0 \quad \textit{for all } j.$$

Proof. Misclassification is a mismatch in argument maximizations:

$$EMclass_k(Y) \neq class_{opt}(Y) \quad \text{if and only if}$$
$$\operatorname{argmax} p_Z(j|Y, \Theta_{EM}[k]) \neq \operatorname{argmax} p_Z(j|Y, \Theta_*). \tag{40}$$

This mismatch disappears as Θ_{EM} converges to Θ_*. Thus

$$\operatorname{argmax} p_Z(j|Y, \Theta_{EM}[k]) \text{ converges to } \operatorname{argmax} p_Z(j|Y, \Theta_*)$$

since

$$\lim_{k\to\infty} \|\Theta_{EM}[k] - \Theta_*\| = 0. \tag{41}$$

So the argument maximization mismatch decreases as the EM estimates get closer to the optimum parameter Θ_*. But the NEM condition (39) implies that the following inequality holds on average at the *k*th iteration:

$$\|\Theta_{NEM}[k] - \Theta_*\| \leq \|\Theta_{EM}[k] - \Theta_*\|. \tag{42}$$

Thus for a fixed iteration count *k*:

$$P\left(NEMclass_k(Y) \neq class_{opt}(Y)\right)$$
$$\leq P\left(EMclass_k(Y) \neq class_{opt}(Y)\right) \tag{43}$$

on average.
So

$$P_{M_N}[k] \leq P_M[k] \tag{44}$$

on average. Thus noise reduces the probability of EM clustering misclassification relative to the EM-optimal classifier on average when the noise satisfies the NEM condition. This means that an unconverged NEM-classifier performs closer to the fully converged classifier than does an unconverged noise-less EM-classifier on average. □

We next state the noise-enhanced EM GMM algorithm in 1-D.

The *D*-dimensional GMM-EM algorithm runs the *N*-step component-wise for each data dimension.

Fig. 1 shows a simulation instance of the predicted GMM noise benefit for 2-D cluster-parameter estimation. The figure shows that the optimum noise reduces GMM-cluster misclassification by almost 30%.

Noisy GMM–EM Algorithm (1-D)

Require: $y_1, \ldots, y_N$ GMM data samples

$k = 1$
while $(\|\theta_k - \theta_{k-1}\| \geq 10^{-tol})$ **do**
N-Step:
$z_i = y_i + n_i$ (45)
where n_i is a sample of the truncated Gaussian $\sim N(0, \frac{\sigma_N}{k^2})$ such that $n_i\left[n_i - 2n_i\left(\mu_{j_i} - y_i\right)\right] \leq 0$ for all i, j
E-Step:
$$Q(\Theta|\Theta(t)) = \sum_{i=1}^{N}\sum_{j=1}^{K} \ln[\alpha_j f(z_i|j,\theta_j)] p_Z(j|y,\Theta(t)) \tag{46}$$
M-Step:
$$\theta_{k+1} = \operatorname*{argmax}_{\theta}\{Q(\theta|\theta_k)\} \tag{47}$$
$k = k + 1$
end while

$$\hat{\theta}_{NEM} = \theta_k. \tag{48}$$

4. Competitive learning algorithms

Competitive learning algorithms learn centroidal patterns from streams of input data by adjusting the weights of only those units that win a distance-based competition or comparison (Grossberg, 1987; Kohonen, 2001; Kosko, 1991a, 1991b). Stochastic competitive learning behaves as a form of *adaptive quantization* because the trained synaptic fan-in vectors (centroids) tend to distribute themselves in the pattern space so as to minimize the mean-squared-error of vector quantization (Kosko, 1991a). Such a quantization vector also converges with probability one to the centroid of its nearest-neighbor class (Kosko, 1991b). We will show that most competitive learning systems benefit from noise. This further suggests that a noise benefit holds for ART systems because they use competitive learning to form learned pattern categories.

Unsupervised competitive learning (UCL) is a blind clustering algorithm that tends to cluster like patterns together. It uses the implied topology of a two-layer neural network. The first layer is just the data layer for the input patterns y of dimension d. There are K-many competing neurons in the second layer. The synaptic fan-in vectors to these neurons define the local centroids or quantization vectors $\mu_1, \ldots, \mu_K$. Simple distance matching approximates the complex nonlinear dynamics of the second-layer neurons competing for activation in an on-center/off-surround winner-take-all connection topology (Kosko, 1991a) as in an ART system. Each incoming pattern stimulates a new competition. The winning *j*th neuron modifies its fan-in of synapses while the losing neurons do not change their synaptic fan-ins. Nearest-neighbor matching picks the winning neuron by finding the synaptic fan-in vector closest to the current input pattern. Then the UCL learning law moves the winner's synaptic fan-in centroid or quantizing vector a little closer to the incoming pattern.

We first write the UCL algorithm as a two-step process of distance-based "winning" and synaptic-vector update. The first step is the same as the assignment step in *k*-means clustering. This equivalence alone argues for a noise benefit. But the second step differs in the learning increment. So UCL differs from *k*-means clustering despite their similarity. This difference prevents a direct subsumption of UCL from the EM algorithm. It thus prevents a direct proof of a UCL noise benefit based on the NEM Theorem.

We also assume in all simulations that the initial K centroid or quantization vectors equal the first K *random* pattern samples: $\mu_1(1) = y(1), \ldots, \mu_K(K) = y(K)$. Other initialization schemes

could identify the first K quantizing vectors with any K other pattern samples so long as they are random samples. Setting all initial quantizing vectors to the same value can distort the learning process. All competitive learning simulations used linearly decaying learning coefficients $c_j(t) = 0.3(1 - t/1500)$.

Unsupervised Competitive Learning (UCL) Algorithm

Pick the winner:

The jth neuron wins at t if

$$\|y(t) - \mu_j(t)\| \leq \|y(t) - \mu_k(t)\| \quad k \neq j. \tag{49}$$

Update the Winning Quantization Vector:

$$\mu_j(t+1) = \mu_j(t) + c_t\left[y(t) - \mu_j(t)\right] \tag{50}$$

for a decreasing sequence of learning coefficients $\{c_t\}$.

A similar stochastic difference equation can update the covariance matrix Σ_j of the winning quantization vector:

$$\Sigma_j(t+1) = \Sigma_j(t) + c_t\left[(y(t) - \mu_j(t))(y(t) - \mu_j(t))^T - \Sigma_j(t)\right]. \tag{51}$$

A modified version can update the pseudo-covariations of alpha-stable random vectors that have no higher-order moments (Kim & Kosko, 1996). The simulations in this paper do not adapt the covariance matrix.

We can rewrite the two UCL steps (49) and (50) into a single stochastic difference equation. This rewrite requires that the distance-based indicator function $\mathbb{I}_{D_j}$ replace the pick-the-winner step (49) just as it does for the assign-samples step (26) of k-means clustering:

$$\mu_j(t+1) = \mu_j(t) + c_t\, \mathbb{I}_{D_j}(y(t))\, \left[y(t) - \mu_j(t)\right]. \tag{52}$$

The one-equation version of UCL in (52) more closely resembles Grossberg's original deterministic differential-equation form of competitive learning in neural modeling (Grossberg, 1982; Kosko, 1991a):

$$\dot{m}_{ij} = S_j(y_j)\left[S_i(x_i) - m_{ij}\right] \tag{53}$$

where m_{ij} is the synaptic memory trace from the ith neuron in the input field to the jth neuron in the output or competitive field. The ith input neuron has a real-valued activation x_i that feeds into a bounded nonlinear signal function (often a sigmoid) S_i. The jth competitive neuron likewise has a real-valued scalar activation y_j that feeds into a bounded nonlinear signal function S_j. But competition requires that the output signal function S_j approximate a zero–one decision function. This gives rise to the approximation $S_j \approx \mathbb{I}_{D_j}$.

The two-step UCL algorithm is the same as Kohonen's "self-organizing map" algorithm (Kohonen, 1990, 2001) if the self-organizing map updates only a single winner. Both algorithms can update direct or graded subsets of neurons near the winner. These near-neighbor beneficiaries can result from an implied connection topology of competing neurons if the square K-by-K connection matrix has a positive diagonal band with other entries negative.

Supervised competitive learning (SCL) punishes the winner for misclassifications. This requires a teacher or supervisor who knows the class membership D_j of each input pattern y and who knows the classes that the other synaptic fan-in vectors represent. The SCL algorithm moves the winner's synaptic fan-in vector μ_j away from the current input pattern y if the pattern y does not belong to the winner's class D_j. So the learning increment gets a minus sign rather than the plus sign that UCL would use. This process amounts to inserting a *reinforcement function* r into the winner's learning increment as follows:

$$\mu_j(t+1) = \mu_j(t) + c_t r_j(y)\left[y - \mu_j(t)\right] \tag{54}$$

$$r_j(y) = \mathbb{I}_{D_j}(y) - \sum_{i \neq j} \mathbb{I}_{D_i}(y). \tag{55}$$

Russian learning theorist Ya Tsypkin appears to be the first to have arrived at the SCL algorithm. He did so in 1973 in the context of an adaptive Bayesian classifier (Tsypkin, 1973).

Differential Competitive Learning (DCL) is a hybrid learning algorithm (Kong & Kosko, 1991; Kosko, 1991a). It replaces the win–lose competitive learning term S_j in (53) with the *rate* of winning $\dot{S}_j$. The rate or differential structure comes from the differential Hebbian law (Kosko, 1986):

$$\dot{m}_{ij} = -m_{ij} + \dot{S}_i\dot{S}_j \tag{56}$$

using the above notation for synapses m_{ij} and signal functions S_i and S_j. The traditional Hebbian learning law just correlates neuron activations rather than their velocities. The result is the DCL differential equation:

$$\dot{m}_{ij} = \dot{S}_j(y_j)\left[S_i(x_i) - m_{ij}\right]. \tag{57}$$

Then the synapse learns only if the jth competitive neuron changes its win–loss status. The synapse learns in competitive learning only if the jth neuron itself wins the competition for activation. The time derivative in DCL allows for both positive and negative reinforcement of the learning increment. This polarity resembles the plus–minus reinforcement of SCL even though DCL is a blind or unsupervised learning law. Unsupervised DCL compares favorably with SCL in some simulation tests (Kong & Kosko, 1991).

We simulate DCL with the following stochastic difference equation:

$$\mu_j(t+1) = \mu_j(t) + c_t \Delta S_j(z_j)\left[S(y) - \mu_j(t)\right] \tag{58}$$

$$\mu_i(t+1) = \mu_i(t) \quad \text{if } i \neq j \tag{59}$$

when the jth synaptic vector wins the metrical competition as in UCL. $\Delta S_j(z_j)$ is the time derivative of the jth output neuron activation. We approximate it as the signum function of the time difference of the training sample z (Dickerson & Kosko, 1994; Kong & Kosko, 1991):

$$\Delta S_j(z_j) = sgn\left[z_j(t+1) - z_j(t)\right]. \tag{60}$$

The competitive learning simulations in Fig. 5 used noisy versions of the competitive learning algorithms just as the clustering simulations used noisy versions. The noise was additive white Gaussian vector noise n with decreasing variance (annealed noise). We added the noise n to the pattern data y to produce the training sample z: $z = y + n$ where $n \sim N(\mathbf{0}, \Sigma_\sigma(t))$. The noise covariance matrix $\Sigma_\sigma(t)$ was just the scaled identity matrix $(t^{-2}\sigma)I$ for standard deviation or noise level $\sigma > 0$. This allows the scalar σ to control the noise intensity for the entire vector learning process. We annealed or decreased the variance as $\Sigma_\sigma(t) = (t^{-2}\sigma)I$ as in (Osoba et al., 2011, in review). So the noise vector random sequence $\{n(1), n(2), \ldots\}$ is an independent (white) sequence of similarly distributed Gaussian random vectors. We state for completeness the three-step noisy UCL algorithm.

Noise similarly perturbed the input patterns $y(t)$ for the SCL and DCL learning algorithms. This leads to the following algorithm statements for SCL and DCL:

Fig. 3 shows that noise injection speeded up UCL convergence by about 25%. Fig. 4 shows that noise injection speeded up SCL convergence by less than 5%. Fig. 5 shows that noise injection speeded up DCL convergence by about 20%. All three figures used the same

four symmetric Gaussian data clusters in Fig. 3 (inset). We also observed similar noise benefits for additive uniform noise.

Noisy UCL Algorithm

Noise Injection:

Define $z(t) = y(t) + n(t)$

for $n(t) \sim N(\mathbf{0}, \Sigma_\sigma(t))$ and annealing schedule

$$\Sigma_\sigma(t) = \frac{\sigma}{t^2} I. \tag{61}$$

Pick the Noisy Winner:

The jth neuron wins at t if

$$\|z(t) - \mu_j(t)\| \leq \|z(t) - \mu_k(t)\| \quad k \neq j. \tag{62}$$

Update the Winning Quantization Vector:

$$\mu_j(t+1) = \mu_j(t) + c_t \left[z(t) - \mu_j(t)\right] \tag{63}$$

for a decreasing sequence of learning coefficients $\{c_t\}$.

Noisy SCL Algorithm

Noise Injection:

Define $z(t) = y(t) + n(t)$

for $n(t) \sim N(\mathbf{0}, \Sigma_\sigma(t))$ and annealing schedule

$$\Sigma_\sigma(t) = \frac{\sigma}{t^2} I. \tag{64}$$

Pick the Noisy Winner:

The jth neuron wins at t if

$$\|z(t) - \mu_j(t)\| \leq \|z(t) - \mu_k(t)\| \quad k \neq j. \tag{65}$$

Update the Winning Quantization Vector:

$$\mu_j(t+1) = \mu_j(t) + c_t\, r_j(z) \left[z(t) - \mu_j(t)\right] \tag{66}$$

where

$$r_j(z) = \mathbb{I}_{D_j}(z) - \textstyle\sum_{i \neq j} \mathbb{I}_{D_i}(z) \tag{67}$$

and $\{c_t\}$ is a decreasing sequence of learning coefficients.

Noisy DCL Algorithm

Noise Injection:

Define $z(t) = y(t) + n(t)$

for $n(t) \sim N(\mathbf{0}, \Sigma_\sigma(t))$ and annealing schedule

$$\Sigma_\sigma(t) = \frac{\sigma}{t^2} I. \tag{68}$$

Pick the Noisy Winner:

The jth neuron wins at t if

$$\|z(t) - \mu_j(t)\| \leq \|z(t) - \mu_k(t)\| \quad k \neq j. \tag{69}$$

Update the Winning Quantization Vector:

$$\mu_j(t+1) = \mu_j(t) + c_t\, \Delta S_j(z_j) \left[S(x) - \mu_j(t)\right] \tag{70}$$

where

$$\Delta S_j(z_j) = sgn\left[z_j(t+1) - z_j(t)\right] \tag{71}$$

and $\{c_t\}$ is a decreasing sequence of learning coefficients.

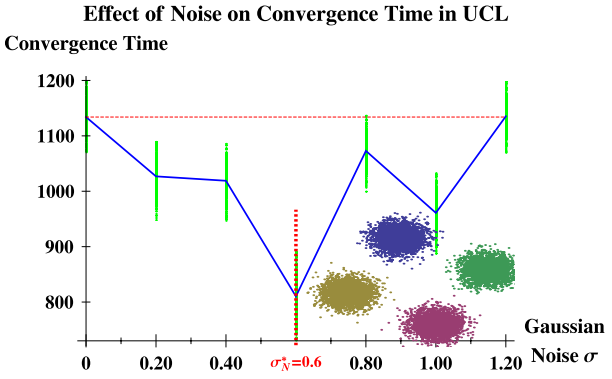

Fig. 3. Noise benefit in the convergence time of Unsupervised Competitive Learning (UCL). The inset shows the four Gaussian data clusters with the same covariance matrix. The convergence time is the number of learning iterations before the synaptic weights stayed within 25% of the final converged synaptic weights. The dashed horizontal line shows the convergence time for UCL without additive noise. The figure shows that a small amount of noise can reduce convergence time by about 25%. The procedure adapts to noisy samples from a Gaussian mixture of four subpopulations. The subpopulations have centroids on the vertices of the rotated square of side-length 24 centered at the origin as the inset figure shows. The additive noise is zero-mean Gaussian.

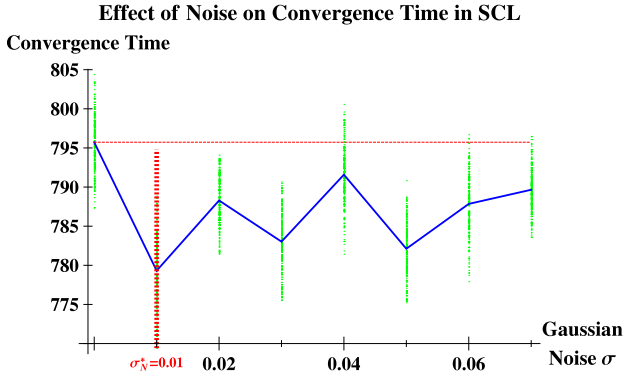

Fig. 4. Noise benefit in the convergence time of Supervised Competitive Learning (SCL). The convergence time is the number of learning iterations before the synaptic weights stayed within 25% of the final converged synaptic weights. The dashed horizontal line shows the convergence time for SCL without additive noise. The figure shows that a small amount of noise can reduce convergence time by less than 5%. The procedure adapts to noisy samples from a Gaussian mixture of four subpopulations. The subpopulations have centroids on the vertices of the rotated square of side-length 24 centered at the origin as the inset in Fig. 3 shows. The additive noise is zero-mean Gaussian.

Fig. 6 shows how noise can also reduce the centroid estimate's jitter in the UCL algorithm. The jitter is the variance of the last 75 synaptic fan-ins of a UCL run. It measures how much the centroid estimates move after learning has converged.

5. Conclusion

Noise can speed convergence in expectation–maximization clustering and in at least three types of competitive learning under fairly general conditions. This suggests that noise should improve the performance of other clustering and competitive-learning algorithms. An open research question is whether the Noisy Expectation–Maximization Theorem (Osoba et al., 2011, in review) or some other mathematical result can guarantee the observed noise benefits for the three types of competitive learning and for similar clustering algorithms.

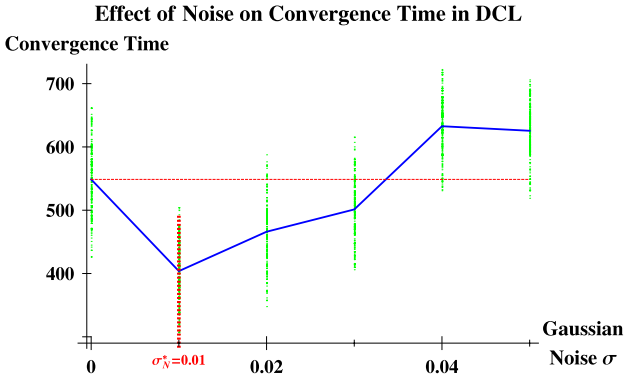

Fig. 5. Noise benefit in the convergence time of Differential Competitive Learning (DCL). The convergence time is the number of learning iterations before the synaptic weights stayed within 25% of the final converged synaptic weights. The dashed horizontal line shows the convergence time for DCL without additive noise. The figure shows that a small amount of noise can reduce convergence time by almost 20%. The procedure adapts to noisy samples from a Gaussian mixture of four subpopulations. The subpopulations have centroids on the vertices of the rotated square of side-length 24 centered at the origin as the inset in Fig. 3 shows. The additive noise is zero-mean Gaussian.

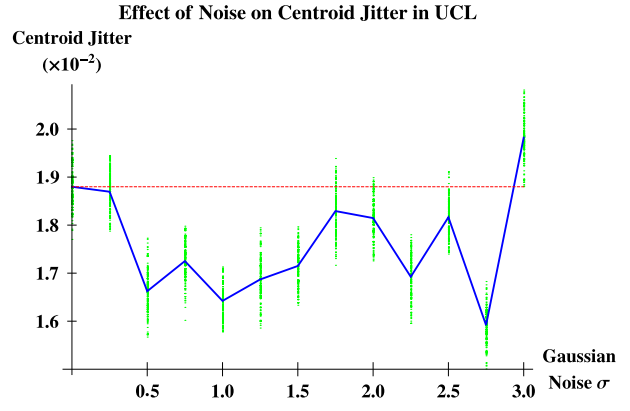

Fig. 6. Noise reduces centroid jitter for unsupervised competitive learning (UCL). The centroid jitter is the variance of the last 75 centroids of a UCL run. The dashed horizontal line shows the centroid jitter in the last 75 estimates for a UCL without additive noise. The plot shows that the some additive noise makes the UCL centroids more stationary. Too much noise makes the UCL centroids move more. The procedure adapts to noisy data from a Gaussian mixture of four subpopulations. The subpopulations have centroids on the vertices of the rotated square of side-length 24 centered at the origin as the inset in Fig. 3 shows. The additive noise is zero-mean Gaussian.

References

Benzi, R., Parisi, G., Sutera, A., & Vulpiani, A. (1983). A theory of stochastic resonance in climatic change. *SIAM Journal on Applied Mathematics*, *43*, 565–578.

Bezdek, J., Ehrlich, R., & Fill, W. (1984). FCM: the fuzzy c-means clustering algorithm. *Computers & Geosciences*, *10*, 191–203.

Brey, J. J., & Prados, A. (1996). Stochastic resonance in a one-dimension Ising model. *Physics Letters. A*, *216*, 240–246.

Bulsara, A. R., Boss, R. D., & Jacobs, E. W. (1989). Noise effects in an electronic model of a single neuron. *Biological Cybernetics*, *61*, 211–222.

Bulsara, A. R., & Gammaitoni, L. (1996). Tuning in to noise. *Physics Today*, 39–45.

Bulsara, A. R., & Zador, A. (1996). Threshold detection of wideband signals: a noise-induced maximum in the mutual information. *Physical Review E*, *54*, R2185–R2188.

Carpenter, G., & Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, *37*, 54–115.

Carpenter, G., Grossberg, S., & Reynolds, J. (1991a). ARTMAP: supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks*, *4*, 565–588.

Carpenter, G., Grossberg, S., & Rosen, D. (1991b). Fuzzy ART: fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, *4*, 759–771.

Castro, R., & Sauer, T. (1997). Chaotic stochastic resonance: noise-enhanced reconstruction of attractors. *Physical Review Letters*, *79*, 1030–1033.

Celeux, G., & Govaert, G. (1992). A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics & Data Analysis*, *14*, 315–332.

Chapeau-Blondeau, F. (1997). Noise-enhanced capacity via stochastic resonance in an asymmetric binary channel. *Physical Review E*, *55*, 2016–2019.

Chapeau-Blondeau, F., & Rousseau, D. (2004). Noise-enhanced performance for an optimal Bayesian estimator. *IEEE Transactions on Signal Processing*, *52*, 1327–1334.

Chen, H., Varshney, P., Kay, S., & Michels, J. (2009). Noise enhanced nonparametric detection. *Institute of Electrical and Electronics Engineers. Transactions on Information Theory*, *55*, 499–506.

Cherng, J., & Lo, M. (2001). A hypergraph based clustering algorithm for spatial data sets. In *Proceedings of the IEEE international conference on data mining*, ICDM 2001 (pp. 83–90).

Collins, J. J., Chow, C. C., & Imhoff, T. T. (1995). Stochastic resonance without tuning. *Nature*, *376*, 236–238.

Cordo, P., Inglis, J. T., Vershueren, S., Collins, J. J., Merfeld, D. M., Rosenblum, S., Buckley, S., & Moss, F. (1996). Noise in human muscle spindles. *Nature*, *383*, 769–770.

Dickerson, J. A., & Kosko, B. (1994). Virtual worlds as fuzzy cognitive maps. *Presence*, *3*, 173–189.

Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine learning*, *29*, 103–130.

Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern classification* (2nd ed.). Wiley-Interscience.

Fogel, D. (1994). An introduction to simulated evolutionary optimization. *IEEE Transactions on Neural Networks*, *5*, 3–14.

Gammaitoni, L., Hänggi, P., Jung, P., & Marchesoni, F. (1998). Stochastic resonance. *Reviews of Modern Physics*, *70*, 223–287.

Grifoni, M., & Hänggi, P. (1996). Nonlinear quantum stochastic resonance. *Physical Review E*, *54*, 1390–1401.

Grossberg, S. (1982). Studies of mind and brain: neural principles of learning, perception, development, cognition, and motor control, D. Reidel Publishing Company.

Grossberg, S. (1987). Competitive learning: from interactive activation to adaptive resonance. *Cognitive Science*, *11*, 23–63.

Guerra, D., Bulsara, A., Ditto, W., Sinha, S., Murali, K., & Mohanty, P. (2010). A noise-assisted reprogrammable nanomechanical logic gate. *Nano Letters*, *10*, 1168–1171.

Hall, L., Ozyurt, I., & Bezdek, J. (1999). Clustering with a genetically optimized approach. *IEEE Transactions on Evolutionary Computation*, *3*, 103–112.

Hamerly, G., & Elkan, C. (2002). Alternatives to the k-means algorithm that find better clusterings. In *Proceedings of the eleventh international conference on information and knowledge management* (CIKM 2002) (pp. 600–607).

Hänggi, P. (2002). Stochastic resonance in biology. *ChemPhysChem*, *3*, 285–290.

Hartuv, E., & Shamir, R. (2000). A clustering algorithm based on graph connectivity. *Information Processing Letters*, *76*, 175–181.

Hathaway, R. (1986). Another interpretation of the EM algorithm for mixture distributions. *Statistics & Probability Letters*, *4*, 53–56.

Höppner, F., Klawonn, F., & Kruse, R. (1999). *Fuzzy cluster analysis: methods for classification, data analysis and image recognition*. John Wiley.

Kearns, M., Mansour, Y., & Ng, A. (1997). An information-theoretic analysis of hard and soft assignment methods for clustering. In *Proceedings of the thirteenth conference on uncertainty in artificial intelligence* (pp. 282–293). Morgan Kaufmann Publishers Inc..

Kim, H., & Kosko, B. (1996). Fuzzy prediction and filtering in impulsive noise. *Fuzzy Sets and Systems*, *77*, 15–33.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, *78*, 1464–1480.

Kohonen, T. (2001). *Self-organizing maps*. Springer.

Kong, S., & Kosko, B. (1991). Differential competitive learning for centroid estimation and phoneme recognition. *IEEE Transactions on Neural Networks*, *2*, 118–124.

Kosko, B. (1986). Differential hebbian learning. In *AIP conference proceedings. vol. 151* (pp. 277–282).

Kosko, B. (1991a). *Neural networks and fuzzy systems: a dynamical systems approach to machine intelligence*. Prentice Hall.

Kosko, B. (1991b). Stochastic competitive learning. *IEEE Transactions on Neural Networks*, *2*, 522–529.

Kosko, B. (2006). *Noise*. Viking.

Kosko, B., & Mitaim, S. (2003). Stochastic resonance in noisy threshold neurons. *Neural Networks*, *16*, 755–761.

Lee, I., Liu, X., Zhou, C., & Kosko, B. (2006). Noise-enhanced detection of subthreshold signals with carbon nanotubes. *IEEE Transactions on Nanotechnology*, *5*, 613–627.

Löcher, L. G. M., Bulsara, A., Hänggi, P., Neff, J., Wiesenfeld, K., Ditto, W., & Inchiosa, M. E. (1999). Controlling stochastic resonance. *Physical Review Letters*, *82*, 4574–4577.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. vol. 1* (pp. 281–297).

McDonnell, M., Stocks, N., Pearce, C., & Abbott, D. (2008). *Stochastic resonance: from suprathreshold stochastic resonance to stochastic signal quantization*. Cambridge University Press.

Mitaim, S., & Kosko, B. (1998). Adaptive stochastic resonance. *Proceedings of the IEEE: Special Issue on Intelligent Signal Processing*, *86*, 2152–2183.

Moss, F., Bulsara, A., & Shlesinger, M. (Eds.) (1993). *Journal of statistical physics, special issue on stochastic resonance in physics and biology*: *vol. 70 no. 1/2. Proceedings of the NATO advanced research workshop*. Plenum Press.

Osoba, O., Mitaim, S., & Kosko, B. (2011). Noise benefits in the expectation–maximization algorithm: NEM theorems and models. In *The 2011 international joint conference on neural networks (IJCNN)* (pp. 3178–3183). IEEE.

Osoba, O., Mitaim, S., & Kosko, B. (2012). The noisy expectation–maximization algorithm (in review).

Patel, A., & Kosko, B. (2008). Stochastic resonance in continuous and spiking neurons with levy noise. *IEEE Transactions on Neural Networks*, *19*, 1993–2008.

Patel, A., & Kosko, B. (2009). Error-probability noise benefits in threshold neural signal detection. *Neural Networks*, *22*, 697–706.

Patel, A., & Kosko, B. (2010). Optimal mean-square noise benefits in quantizer-array linear estimation. *IEEE Signal Processing Letters*, *17*, 1005–1009.

Patel, A., & Kosko, B. (2011). Noise benefits in quantizer-array correlation detection and watermark decoding. *IEEE Transactions on Signal Processing*, *59*, 488–505.

Rish, I. (2001). An empirical study of the naive Bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence. vol. 3* (pp. 41–46).

Tsypkin, Y. Z. (1973). *Foundations of the theory of learning systems*. Academic Press.

Wilde, M., & Kosko, B. (2009). Quantum forbidden-interval theorems for stochastic resonance. *Journal of Physical A: Mathematical Theory*, *42*.

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, *16*, 645–678.

Xu, R., & Wunsch, D. (2010). Clustering algorithms in biomedical research: a review. *IEEE Reviews in Biomedical Engineering*, *3*, 120–154.

Xu, R., & Wunsch, D. C. (2009). *Clustering*. IEEE Press & Wiley.

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: an efficient data clustering method for very large databases. *ACM SIGMOD Record*, *25*, 103–114.