DYNAMICAL SYSTEM IMPLEMENTATIONS OF SPARSE BAYESIAN LEARNING

Matthew R. O'Shaughnessy, Mark A. Davenport, and Christopher J. Rozell

School of Electrical & Computer Engineering
Georgia Institute of Technology
{moshaughnessy6, mdav, crozell}@gatech.edu

ABSTRACT

Despite its state of the art performance in many applications, the sparse Bayesian learning (SBL) procedure can be expensive to implement, limiting its use in practice. In this paper, we use the locally competitive algorithm (LCA) framework to develop two continuous time dynamical systems whose trajectories converge to a minimum of the SBL objective. The resulting systems are neurally feasible and can be implemented using primitives from analog electronics, potentially opening the SBL procedure to new applications.

Index Terms— Compressed sensing, sparse Bayesian learning, dynamical systems, locally competitive algorithm, continuous time systems

1. INTRODUCTION

Recovering a sparse signal from noisy underdetermined measurements is a fundamental problem in applied mathematics and engineering [1, 2]. However, popular digital algorithms for performing inference in this setting require large computational and power budgets, significantly limiting their use in large real-time applications.

To reduce this computational and power complexity, recent literature has introduced a class of continuous-time dynamical systems called *locally competitive algorithms* (LCA) for solving a class of sparse recovery problems based on ℓ_1-like penalties [3, 4]. These biologically inspired algorithms can be mapped to analog electronics or neural architectures, allowing sparse inference to be performed much more efficiently than in corresponding digital implementations [5].

Although ℓ_1-like penalties are widely used in sparse recovery for their well-studied theoretical properties [6, 7], in many applications the *sparse Bayesian learning* (SBL) procedure of [8, 9] has been shown to exhibit superior performance, particularly in the presence of challenging dictionary structure [10, 11]. Therefore, there is a need for algorithms that can perform SBL inference in the same types of efficient analog architectures that LCAs implement other sparse recovery procedures with.

In this paper, we propose two methods for implementing the SBL inference procedure in the LCA framework, allowing applications that require analog or neurally feasible implementations to reap the performance benefits of SBL.

This work was supported in part by NSF grants CCF-1350616 and CCF-1409422, James S. McDonnell Foundation grant number 220020399, a gift from the Alfred P. Sloan Foundation, and the National Defense Science and Engineering Graduate (NDSEG) Fellowship.

2. BACKGROUND

2.1. Sparse Bayesian learning

We consider the problem of recovering a sparse signal $\boldsymbol{x} \in \mathbb{R}^n$ from the noisy, underdetermined measurements $\boldsymbol{y} = \boldsymbol{\Phi x} + \boldsymbol{e}$, where $\boldsymbol{\Phi} \in \mathbb{R}^{m \times n}$ represents the dictionary and $\boldsymbol{e} \in \mathbb{R}^m \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$ represents measurement noise.

The SBL procedure has been shown to recover $\boldsymbol{x}$ from measurements of this form more accurately than traditional ℓ_1-based recovery procedures in some settings, particularly when the dictionary contains challenging structure such as coherence and mismatched column magnitudes [12, 10, 11]. Operationally, SBL is a Bayesian strategy that consists of a *probability model* describing a generative model for the measurements $\boldsymbol{y}$ and an *inference procedure* that computes a point estimate of $\boldsymbol{x}$ using this probability model and the observations $\boldsymbol{y}$ [8, 9].

The probability model consists of:

- A *Gaussian likelihood* $p(\boldsymbol{y}|\boldsymbol{x}, \sigma^2) \sim \mathcal{N}(\boldsymbol{\Phi x}, \sigma^2 \boldsymbol{I})$, with a (conjugate) inverse gamma hyperprior on the noise variance $p(\sigma^2) \sim \mathcal{IG}(c, d)$; and
- A *Gaussian prior* $p(\boldsymbol{x}|\boldsymbol{\gamma}) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\gamma})$ with (conjugate) inverse gamma hyperpriors on the noise variances $p(\gamma_i) \sim \mathcal{IG}(a_i, b_i)$.

The Gaussian prior on $\boldsymbol{x}$ does not have the high kurtosis of distributions known to encourage sparsity; instead, SBL encourages sparsity in $\boldsymbol{x}$ through an inference procedure that adaptively selects the prior variances γ_i from the data $\boldsymbol{y}$ as

$$\begin{aligned} \boldsymbol{\gamma}^* &= \arg\max_{\boldsymbol{\gamma}} p(\boldsymbol{y}|\boldsymbol{\gamma}, \sigma^2) \\ &= \arg\min_{\boldsymbol{\gamma}} \underbrace{\log|\boldsymbol{C}| + \boldsymbol{y}^T \boldsymbol{C}^{-1} \boldsymbol{y}}_{\mathcal{L}(\boldsymbol{\gamma})}, \end{aligned} \tag{1}$$

where $\boldsymbol{C} = \sigma^2\boldsymbol{I} + \boldsymbol{\Phi}\boldsymbol{\Gamma}\boldsymbol{\Phi}^T$ is the covariance of the Gaussian posterior $p(\boldsymbol{y}|\boldsymbol{\gamma}, \sigma^2)$ and $\boldsymbol{\Gamma} = \mathrm{diag}(\boldsymbol{\gamma})$. Once the prior variances have been selected using (1), the point estimate $\widehat{\boldsymbol{x}} = \arg\max_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\gamma}^*, \sigma^2)$ can be computed in closed form as $\widehat{\boldsymbol{x}} = \sigma^{-2}\left(\boldsymbol{\Gamma}^{-1} + \sigma^{-2}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T\boldsymbol{y}$. Intuitively, this method for automatically generating a prior $p(\boldsymbol{x}|\boldsymbol{\gamma})$ promotes sparsity because when $\gamma_i^* \approx 0$, a large amount of evidence from the measurements is needed for the point estimate $\widehat{x}_i$ to be nonzero.

The main computational cost of the SBL procedure is minimizing the nonconvex objective (1), which is typically performed iteratively [8, 13, 14]. The contribution of this paper is two dynamical systems that allows this procedure to be performed efficiently using continuous time (e.g., analog or neural) systems.

2.2. Locally competitive algorithms

The locally competitive algorithm (LCA) [3, 4] is a dynamical system whose trajectory converges to the solution of the penalized least squares problem

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \left[\frac{1}{2}\left\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}\right\|_2^2 + \lambda C(\boldsymbol{x})\right], \tag{2}$$

where $C(\boldsymbol{x})$ is a sparsity-encouraging penalty function. The LCA is neurally feasible and can be implemented with primitives from analog electronics, allowing fast and power-efficient hardware implementations (see, e.g., [5]).

In the LCA framework, each coefficient x_i is associated with an internal state u_i that is mapped to the coefficients by a nonlinear thresholding function T_λ as

$$\boldsymbol{x}(t) = T_\lambda\left(\boldsymbol{u}(t)\right).$$

The evolution of the internal states is described by the dynamical system[1] [3]

$$\dot{\boldsymbol{u}}(t) = \frac{1}{\tau}\left[\boldsymbol{\Phi}^T\boldsymbol{y} - \left(\boldsymbol{\Phi}^T\boldsymbol{\Phi} - \boldsymbol{I}\right)\boldsymbol{x}(t) - \boldsymbol{u}(t)\right]. \tag{3}$$

Intuitively, each state u_i in the LCA can be thought of as being simultaneously "charged" by the similarity of dictionary column i and the measurements, "inhibited" by outputs x_j modulated by the similarity of dictionary elements $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$, and decaying at a rate proportional to the current state.

A gradient descent-like dynamical system implementing the problem (2) for general $C(\boldsymbol{x})$ can be realized by selecting $T_\lambda(\boldsymbol{u})$ to make the trajectory of $\boldsymbol{u}$ proportional to the negative gradient of the cost function. It can be shown [4] that this relationship holds when $T_\lambda(\boldsymbol{u})$ satisfies

$$\lambda\nabla_{\boldsymbol{x}}C(\boldsymbol{x}) = \boldsymbol{u} - T_\lambda(\boldsymbol{u}). \tag{4}$$

[1] We assume for conciseness that $\boldsymbol{\Phi}$ is scaled to have unit column norms, but this assumption can be relaxed with a simple modification of (3).

This relationship is used in [15] to implement a menagerie of sparse coding procedures. Here, we use it to derive two LCAs implementing the nonseparable, dictionary-dependent penalty of the sparse Bayesian learning algorithm.

3. LCA IMPLEMENTATIONS OF SBL

3.1. Direct implementation

Although it is not immediately obvious how to represent the SBL objective (1) in the form of the penalized least squares problem (2), by using matrix identities as in [16] we can upper bound $\mathcal{L}(\boldsymbol{\gamma})$ by

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\gamma}) = \frac{1}{2}\left\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}\right\|_2^2 + \frac{\lambda}{2}\left[\sum_{i=1}^{n}\gamma_i^{-1}x_i^2 + \log|\boldsymbol{C}|\right] \geq \mathcal{L}(\boldsymbol{\gamma}).$$

Therefore, SBL inference can be performed by minimizing $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\gamma})$, which fits the form of (2) with

$$C_{\boldsymbol{\gamma}}(\boldsymbol{x}) = \frac{1}{2}\sum_{i=1}^{n}\gamma_i^{-1}x_i^2 + \frac{1}{2}\log|\boldsymbol{C}|. \tag{5}$$

If the value of $\boldsymbol{\gamma}$ that minimizes (5) was known, we could implement SBL in the LCA framework simply by appealing to the relationship (4). However, because we need to minimize $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\gamma})$ *jointly* with respect to $\boldsymbol{x}$ and $\boldsymbol{\gamma}$, we instead create a second, *coupled* dynamical system which converges to the $\boldsymbol{\gamma}$ that minimizes $C_{\boldsymbol{\gamma}}(\boldsymbol{x})$. By using the trajectory of this coupled dynamical system as the value of $\boldsymbol{\gamma}$ in (5), we can derive the threshold function as if $\boldsymbol{\gamma}$ was fixed.

Specifically, we minimize the penalty (5) with respect to $\boldsymbol{\gamma}$ by introducing a coupled dynamical system of the form

$$\begin{aligned}\dot{\gamma}_i(t) &= -\frac{1}{\tau_\gamma}\frac{\partial}{\partial\gamma_i}C_{\boldsymbol{\gamma}}(\boldsymbol{x}) \\ &= \frac{1}{\tau_\gamma}\left[\gamma_i^{-2}(t)x_i^2(t) - \boldsymbol{\phi}_i^T\left(\lambda\boldsymbol{I} + \boldsymbol{\Phi}\boldsymbol{\Gamma}(t)\boldsymbol{\Phi}^T\right)^{-1}\boldsymbol{\phi}_i\right]\end{aligned} \tag{6}$$

for $i = 1, \ldots, n$. Then, solving (4) for $T_\lambda(\boldsymbol{u}(t))$ yields the closed-form expression for the thresholding function

$$\begin{aligned}T_\lambda(\boldsymbol{u}(t)) &= \boldsymbol{u}(t) - \lambda\nabla_{\boldsymbol{x}}C(\boldsymbol{x}(t)) \\ &= \left(\lambda\boldsymbol{\Gamma}^{-1}(t) + \boldsymbol{I}\right)^{-1}\boldsymbol{u}(t).\end{aligned} \tag{7}$$

The complete "direct" implementation of SBL using the LCA framework, which we call SBL-LCA, consists of the coupled dynamical systems (3) and (6) together with the threshold (7).

An important challenge when simulating and implementing these systems is that the system (6) can oscillate around zero when it is close to convergence, causing numerical problems when $\gamma_i(t)$ becomes negative. To avoid this, we can equivalently represent the system using the *inverse* variances

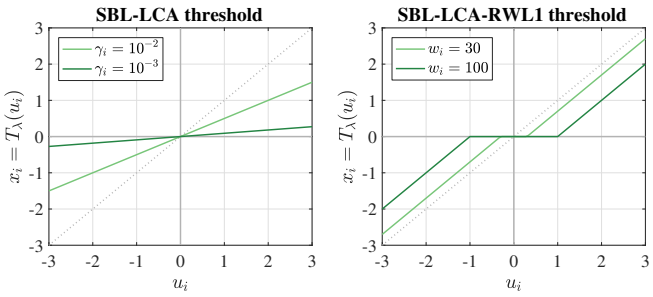

Fig. 1. *Left*: thresholding function for SBL-LCA, (4). *Right*: thresholding function for SBL-RWL1-LCA.

$\alpha_i = \gamma_i^{-1}$. Deriving the LCA using these inverse variances results in the dynamical system

$$\dot{\alpha}_i(t) = \frac{1}{\tau_\gamma}\left[-x_i^2(t) + \alpha_i^{-2}(t)\boldsymbol{\phi}_i^T\left(\lambda\boldsymbol{I} + \boldsymbol{\Phi}\boldsymbol{A}^{-1}(t)\boldsymbol{\Phi}^T\right)^{-1}\boldsymbol{\phi}_i\right],$$

where $\boldsymbol{A} = \mathrm{diag}(\boldsymbol{\alpha})$.

The threshold function for SBL-LCA (7) is depicted in Figure 1 (left) for two values of γ_i. This threshold and dynamical system can be intuitively viewed as encouraging sparsity using the mechanism of SBL: when γ_i is large, the slope of the liner threshold becomes smaller, reducing the magnitude of the corresponding x_i. At convergence, $\gamma_i \approx 0$ for many i, ensuring that these elements of x_i will be zero-valued.

However, this linear threshold does not induce sparsity in $\boldsymbol{x}(t)$ until the system describing $\boldsymbol{\gamma}(t)$ has reached convergence. This is a critical drawback, as the sparsity of the output $\boldsymbol{x}(t)$ over all time is an important factor in enabling low-power hardware and neural implementations.

3.2. Reweighted ℓ_1 implementation

We address this drawback by drawing on related work that separately connects both the LCA and SBL frameworks to the *reweighted* ℓ_1 (RWL1) estimator of [17]. The RWL1 procedure iteratively solves

$$\widehat{\boldsymbol{x}}^{(k+1)} = \arg\min_{\boldsymbol{x}} \frac{1}{2}\left\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}\right\|_2^2 + \lambda\sum_i w_i^{(k)}\left|x_i\right| \quad (8)$$

and updates the weights using a rule of the form $w_i^{(k+1)} = f_i\left(\boldsymbol{x}^{(k+1)}\right)$. (This connection has also been used as a starting point for implementing SBL using recurrent neural networks [18].) We show that using a similar procedure to implement SBL inference with the LCA framework results in a system that outputs a signal estimate $\boldsymbol{x}(t)$ that is sparse for a large portion of the trajectory, not just at convergence.

In [16, 14], Wipf and Nagarajan show that the SBL objective (1) can be minimized using RWL1 with the weight update

$$f_i\left(\boldsymbol{x}^{(k+1)}\right) = \sqrt{\boldsymbol{\phi}_i^T\left(\lambda\boldsymbol{I} + \boldsymbol{\Phi}\boldsymbol{\Gamma}^{(k+1)}\boldsymbol{\Phi}^T\right)^{-1}\boldsymbol{\phi}_i}, \quad (9)$$

where the variance parameters are computed at each step as $\gamma_j^{(k+1)} = \left|x_j^{(k+1)}\right| / w_j^{(k)}$.

Following the procedure of [15], this modified RWL1 procedure can be implemented in the LCA framework by introducing a coupled dynamical system to update the latent variables (now described by weights $w_i(t)$):

$$\dot{w}_i(t) = \frac{1}{\tau_w}\left[\frac{1}{w_i(t)} - \frac{1}{\sqrt{\boldsymbol{\phi}_i^T\left(\lambda\boldsymbol{I} + \boldsymbol{\Phi}\boldsymbol{\Gamma}(t)\boldsymbol{\Phi}^T\right)^{-1}\boldsymbol{\phi}_i}}\right]. \quad (10)$$

As before, the threshold $T_{\lambda,w}(\boldsymbol{u})$ can be obtained by using (4) (see [15] for details):

$$T_{\lambda,w_i}(u_i) = \begin{cases} 0, & |u_i| \le \lambda w_i \\ u_i - \lambda w_i \mathrm{sign}(u_i), & |u_i| > \lambda w_i. \end{cases} \quad (11)$$

This $T_{\lambda,\boldsymbol{w}}(\boldsymbol{u})$, depicted in Figure 1 (right), has the form of a *soft threshold* function. The shrinkage effect of applying this function is better able to promote *exact* sparsity in $\boldsymbol{x}(t)$ before the variances $\boldsymbol{\gamma}$ (represented in (11) through weights w_i) approach zero, allowing SBL-RWL1-LCA to present sparse outputs much sooner than SBL-LCA.

The complete RWL1-based implementation of SBL using the LCA framework, which we call SBL-RWL1-LCA, consists of the coupled dynamical systems (3) and (10) together with the threshold (11).

A challenge of implementing the complete SBL-RWL1-LCA system is that the *nonseparable* expression for $w_i(t)$ described by (9) requires the continuous update of a matrix inverse. Although this nonseparability has been shown to be an important component of SBL's advantage over ℓ_1-based methods [14], the large number of connections needed to compute the required matrix inverse can be difficult to implement in analog electronics [19].

We can begin to relax this requirement by using matrix identities to rewrite the argument of the radical in (10) as

$$\left(\lambda^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\left(\boldsymbol{I} + \lambda^{-1}\boldsymbol{\Gamma}(t)\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}\right)_{ii},$$

directly exposing the dependence on the Gram matrix $\boldsymbol{\Phi}^T\boldsymbol{\Phi}$. In many applications, the coherence structure between columns concentrates the energy of $\boldsymbol{\Phi}^T\boldsymbol{\Phi}$ near the diagonal so that it may be well-approximated by a banded matrix. Exploiting this observation when implementing the latent dynamical system (10) can significantly simplify computation.

4. NUMERICAL EXPERIMENT

In this section we present a brief demonstration of the operation of SBL-LCA and SBL-RWL1-LCA using the continuous time dynamical system simulator DynamicalSystems.jl [20]. We generate a small system with $\Phi \in \mathbb{R}^{16\times 32}$ in the noiseless setting, where the sparse signal $\boldsymbol{x}$ has $k = 2$ nonzero values set to unity. We fix the SBL noise variance parameter to

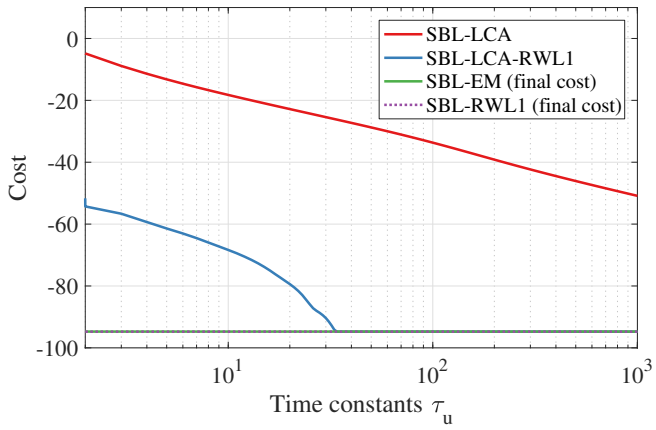

Fig. 2. Objective value of SBL-LCA and SBL-LCA-RWL1 as systems evolve, compared with the final objective value achieved by SBL-EM and SBL-RWL1.

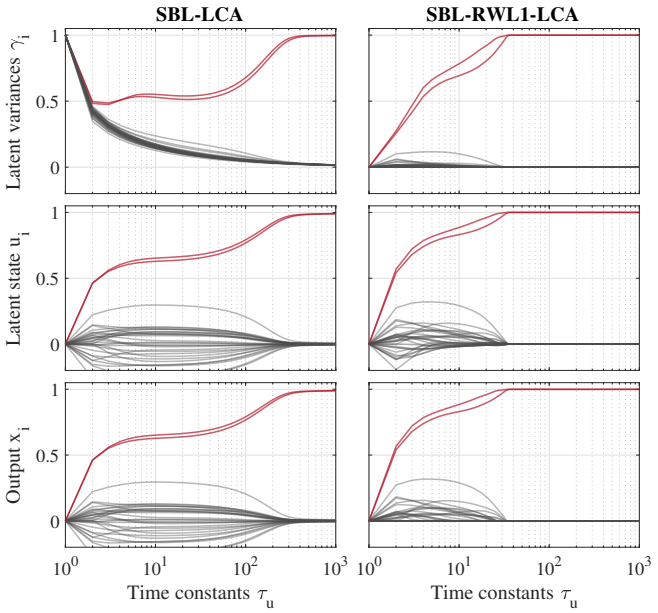

Fig. 3. Value of latent variables γ_i and u_i and output (signal estimate) x_i as the SBL-LCA and SBL-RWL1-LCA systems evolve. Red traces correspond to components on the support of the ground truth $\boldsymbol{x}$; translucent gray traces correspond to components off the support.

$\lambda = 10^{-4}$ and use the time constants $\tau_u = 10^{-2}$ and $\tau_\gamma = \tau_w = 10^{-3}$ (i.e., the states representing variances evolve an order of magnitude "faster" than the states u_i). We have found that this relationship between time constants works well as a rule of thumb, but a more complete exploration of this relationship remains for future work.

Figure 2 shows the value of the SBL objective as the systems evolve compared to the final objective value achieved by iterative implementations of SBL-EM and SBL-RWL1. We observe that SBL-LCA-RWL1 converges rapidly to the final objective value achieved by SBL-EM and SBL-RWL1, while SBL-LCA converges much more slowly.

In Figure 3, we examine the trajectory of the latent states $\{\gamma_i, u_i\}$ and the output (signal estimate) x_i as both LCA systems evolve. We observe from the state of γ_i (computed from w_i and x_i in SBL-RWL1-LCA) that both systems are able to quickly identify the correct support in both SBL-LCA and SBL-RWL1-LCA, but this convergence requires approximately an order of magnitude fewer time constants in SBL-RWL1-LCA. This faster convergence is also reflected in the trajectories of u_i: latent states u_i corresponding to zero-valued elements converge to near-zero much more quickly in the trajectory of SBL-RWL1-LCA. Further, we observe that the form of the thresholding function in SBL-RWL1-LCA both makes the output $\boldsymbol{x}$ sparser immediately after initialization and the off-support elements of $\boldsymbol{x}$ converge to zero more quickly.

5. DISCUSSION

In this paper we presented two dynamical systems for solving the SBL inference problem using the LCA framework. We demonstrated that the first algorithm, based on principles used to implement other sparse coding procedures with LCA, converges relatively slowly to a minimum of the SBL objective and does not have sparse outputs until convergence. To address this problem, we proposed a second method based on the reweighted ℓ_1 implementation of SBL and demonstrated that it both converges more quickly and promotes sparser outputs during the entire trajectory.

Both proposed algorithms can in principle be extended to many other applications of the SBL framework. For instance, the procedures of [21, 22, 23] uses principles from SBL to solve sparse coding problems in which *multiple measurements* of a potentially time-varying signal are available. Since the addition of a time dimension can make these algorithms particularly expensive, they are excellent candidates to benefit from the types of analog-feasible procedures described here.

There are several avenues for future work. First, although stability and convergence speed results have been shown for some LCA systems [24, 25], these results do not immediately apply to the coupled dynamical systems we construct to model SBL's latent variance parameters. Second, although we have found that the "partially separable" system described in Section 3 works well for approximately banded dictionaries, we do not have precise justification for how this affects the fixed points of the resulting system. Finally, the continuous time nature of these dynamical systems can be used to improve applications where the inputs are continuous signals, and a continuous time estimate of $\boldsymbol{x}$ is desired.

6. REFERENCES

[1] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.

[2] Y. C. Eldar and G. Kutyniok, Eds., *Compressed Sensing: Theory and Applications*, Cambridge University Press, Cambridge; New York, 2012.

[3] C. Rozell, D. Johnson, R. Baraniuk, and B. Olshausen, "Locally competitive algorithms for sparse approximation," in *Proc. IEEE Int. Conf. on Image Processing (ICIP)*, San Antonio, TX, USA, Sept. 2007, vol. 4, pp. IV–169–IV–172.

[4] C. J. Rozell, D. H. Johnson, R. G. Baraniuk, and B. A. Olshausen, "Sparse coding via thresholding and local competition in neural circuits," *Neural Computation*, vol. 20, no. 10, pp. 2526–2563, Apr. 2008.

[5] S. Shapero, C. Rozell, and P. Hasler, "Configurable hardware integrate and fire neurons for sparse approximation," *Neural Networks*, vol. 45, pp. 134–143, Sept. 2013.

[6] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[7] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.

[8] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, no. Jun, pp. 211–244, 2001.

[9] D. P. Wipf and B. D. Rao, "Sparse Bayesian learning for basis selection," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2153–2164, Aug. 2004.

[10] D. P. Wipf, "Sparse estimation with structured dictionaries," in *Proc. Adv. in Neural Inf. Process. Syst. (NeurIPS)*, Granada, Spain, Dec. 2011.

[11] Z. Zhang, "Comparison of sparse signal recovery algorithms with highly coherent dictionary matrices: The advantage of T-MSBL," Technical Report., 2012.

[12] D. P. Wipf, *Bayesian Methods for Finding Sparse Representations*, Ph.D. thesis, University of California at San Diego, La Jolla, CA, USA, 2006.

[13] M. E. Tipping and A. Faul, "Fast marginal likelihood maximisation for sparse Bayesian models," in *Proc. Int. Work. on Artificial Intell. and Stat. (AISTATS)*, Key West, FL, USA, Jan. 2003.

[14] D. P. Wipf and S. Nagarajan, "Iterative reweighted l1 and l2 methods for finding sparse solutions," *IEEE J. Sel. Top. Signal Process.*, vol. 4, no. 2, pp. 317–329, Apr. 2010.

[15] A. S. Charles, P. Garrigues, and C. J. Rozell, "A common network architecture efficiently implements a variety of sparsity-based inference problems," *Neural Computation*, vol. 24, no. 12, pp. 3317–3339, Sept. 2012.

[16] D. P. Wipf and S. Nagarajan, "A new view of automatic relevance determination," in *Proc. Adv. in Neural Inf. Process. Syst. (NeurIPS)*, Vancouver, BC, Canada, Dec. 2008.

[17] E. J. Candès, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted l1 minimization," *J. Fourier Anal. Appl.*, vol. 14, no. 5-6, pp. 877–905, Dec. 2008.

[18] H. He, B. Xin, S. Ikehata, and D. P. Wipf, "From Bayesian sparsity to gated recurrent nets," in *Proc. Adv. in Neural Inf. Process. Syst. (NeurIPS)*, Long Beach, CA, USA, Dec. 2017.

[19] A. Cichocki and R. Unbehauen, *Neural Networks for Optimization and Signal Processing*, J. Wiley, Chichester; New York, 1993.

[20] G. Datseris, "DynamicalSystems.jl: A Julia software library for chaos and nonlinear dynamics," *JOSS*, vol. 3, no. 23, pp. 598, Mar. 2018.

[21] D. P. Wipf and B. D. Rao, "An empirical Bayesian strategy for solving the simultaneous sparse approximation problem," *IEEE Trans. Signal Process.*, vol. 55, no. 7, pp. 3704–3716, July 2007.

[22] Z. Zhang and B. D. Rao, "Sparse signal recovery with temporally correlated source vectors using sparse Bayesian learning," *IEEE J. Sel. Top. Signal Process.*, vol. 5, no. 5, pp. 912–926, Sept. 2011.

[23] M. R. O'Shaughnessy, M. A. Davenport, and C. J. Rozell, "Sparse Bayesian Learning with Dynamic Filtering for Inference of Time-Varying Sparse Signals," *arXiv:1902.05362*, Feb. 2019.

[24] A. Balavoine, J. Romberg, and C. J. Rozell, "Convergence and Rate Analysis of Neural Networks for Sparse Approximation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 9, pp. 1377–1389, Sept. 2012.

[25] A. Balavoine, C. J. Rozell, and J. Romberg, "Discrete and Continuous-Time Soft-Thresholding for Dynamic Signal Recovery," *IEEE Trans. Signal Process.*, vol. 63, no. 12, pp. 3165–3176, June 2015.