Recent Advances in Approximate Message Passing

Phil Schniter

THE OHIO STATE UNIVERSITY
Duke UNIVERSITY
iiD

Collaborators: **Sundeep Rangan** (NYU), **Alyson Fletcher** (UCLA), **Mark Borgerding** (OSU)

Supported in part by NSF grants IIP-1539960 and CCF-1527162.

SPARS — June 8, 2017

Overview

1. Linear Regression, AMP, and Vector AMP (VAMP)
2. VAMP, ADMM, and Convergence in the Convex Setting
3. VAMP Convergence in the Non-Convex Setting
4. VAMP for Inference
5. EM-VAMP and Adaptive VAMP
6. Plug-and-play VAMP & Whitening
7. VAMP as a Deep Neural Network
8. VAMP for the Generalized Linear Model

Outline

The Linear Regression Problem

Consider the following linear regression problem:

Recover $\boldsymbol{x}_o$ from

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_o + \boldsymbol{w} \quad \text{with} \quad \begin{cases} \boldsymbol{x}_o \in \mathbb{R}^N & \text{unknown signal} \\ \boldsymbol{A} \in \mathbb{R}^{M \times N} & \text{known linear operator} \\ \boldsymbol{w} \in \mathbb{R}^M & \text{white Gaussian noise.} \end{cases}$$

Typical methodologies:

1. Regularized loss minimization (or MAP estimation):

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \frac{\theta_2}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + R(\boldsymbol{x}; \boldsymbol{\theta}_1)$$

2. Approximate MMSE:

$$\widehat{\boldsymbol{x}} \approx \mathrm{E}\{\boldsymbol{x}|\boldsymbol{y}\} \ \text{ for } \ \boldsymbol{x} \sim p(\boldsymbol{x}; \boldsymbol{\theta}_1), \ \ \boldsymbol{y} \sim \mathcal{N}(\boldsymbol{A}\boldsymbol{x}, \boldsymbol{I}/\theta_2)$$

3. Plug-and-play: iteratively apply a denoising algorithm like BM3D

4. Train a deep network to recover $\boldsymbol{x}_o$ from $\boldsymbol{y}$.

The AMP Methodology

- All of the aforementioned methodologies can be addressed using the Approximate Message Passing (AMP) framework.[1]
- AMP tackles these difficult global optimization/inference problems through a sequence of simpler local optimization/inference problems.
- It does this by appropriate definition of a denoiser $\boldsymbol{g}_1(\cdot;\gamma,\boldsymbol{\theta}_1):\mathbb{R}^N\to\mathbb{R}^N$:
  - Optimization: $\boldsymbol{g}_1(\boldsymbol{r};\gamma,\boldsymbol{\theta}_1)=\arg\min_{\boldsymbol{x}} R(\boldsymbol{x};\boldsymbol{\theta}_1)+\frac{\gamma}{2}\|\boldsymbol{x}-\boldsymbol{r}\|_2^2 \triangleq$ "$\text{prox}_{R/\gamma}(\boldsymbol{r})$"
  - MMSE: $\boldsymbol{g}_1(\boldsymbol{r};\gamma,\boldsymbol{\theta}_1)=\mathrm{E}\left\{\boldsymbol{x}\,\middle|\,\boldsymbol{r}=\boldsymbol{x}+\mathcal{N}(\boldsymbol{0},\boldsymbol{I}/\gamma)\right\}$
  - Plug-and-play:[2] $\boldsymbol{g}_1(\boldsymbol{r};\gamma,\boldsymbol{\theta}_1)=\text{BM3D}(\boldsymbol{r},1/\gamma)$
  - Deep network: $\boldsymbol{g}_1(\boldsymbol{r};\gamma,\boldsymbol{\theta}_1)$ is learned.

[1] Donoho, Maleki, Montanari'09, [2] Metzler, Maleki, Baraniuk'14

AMP: the good, the bad, and the ugly

The good:

- With large i.i.d. sub-Gaussian $\boldsymbol{A}$, AMP performs provably[3] well, in that it can be rigorously characterized by a scalar state-evolution (SE). When this SE has a unique fixed point, AMP converges to the Bayes optimal solution.
- Empirically, AMP behaves well with many other "sufficiently random" $\boldsymbol{A}$ (e.g., randomly sub-sampled Fourier $\boldsymbol{A}$ & i.i.d. sparse $\boldsymbol{x}$).

The bad:

- With general $\boldsymbol{A}$, AMP gives no guarantees.

The ugly:

- With some $\boldsymbol{A}$, AMP may fail to converge! (e.g., ill-conditioned or non-zero-mean $\boldsymbol{A}$)

[3] Bayati,Montanari'15, Bayati,Lelarge,Montanari'15

The Vector AMP (VAMP) Algorithm 

Take SVD $\boldsymbol{A} = \boldsymbol{U}\,\mathrm{Diag}(\boldsymbol{s})\boldsymbol{V}^\mathsf{T}$, choose $\zeta \in (0,1]$ and Lipschitz $\boldsymbol{g}_1(\cdot;\gamma_1,\boldsymbol{\theta}_1) : \mathbb{R}^N \to \mathbb{R}^N$.

Initialize $\boldsymbol{r}_1, \gamma_1$.

For $k = 1, 2, 3, \ldots$

$\widehat{\boldsymbol{x}}_1 \leftarrow \boldsymbol{g}_1(\boldsymbol{r}_1;\gamma_1,\boldsymbol{\theta}_1)$ — denoising of $\boldsymbol{r}_1 = \boldsymbol{x}_o + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_1)$

$\eta_1 \leftarrow \gamma_1 N / \operatorname{tr}\left[\dfrac{\partial \boldsymbol{g}_1(\boldsymbol{r}_1;\gamma_1,\boldsymbol{\theta}_1)}{\partial \boldsymbol{r}_1}\right]$

$\boldsymbol{r}_2 \leftarrow (\eta_1 \widehat{\boldsymbol{x}}_1 - \gamma_1 \boldsymbol{r}_1)/(\eta_1 - \gamma_1)$ — Onsager correction

$\gamma_2 \leftarrow \eta_1 - \gamma_1$

$\widehat{\boldsymbol{x}}_2 \leftarrow \boldsymbol{g}_2(\boldsymbol{r}_2;\gamma_2,\theta_2)$ — LMMSE estimate $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{r}_2, \boldsymbol{I}/\gamma_2)$ from $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\theta_2)$

$\eta_2 \leftarrow \gamma_2 N / \operatorname{tr}\left[\dfrac{\partial \boldsymbol{g}_2(\boldsymbol{r}_2;\gamma_2,\theta_2)}{\partial \boldsymbol{r}_2}\right]$

$\boldsymbol{r}_1 \leftarrow \zeta(\eta_2 \widehat{\boldsymbol{x}}_2 - \gamma_2 \boldsymbol{r}_2)/(\eta_2 - \gamma_2) + (1-\zeta)\boldsymbol{r}_1$ — Onsager correction

$\gamma_1 \leftarrow \zeta(\eta_2 - \gamma_2) + (1-\zeta)\gamma_1$ — damping

where

$$\boldsymbol{g}_2(\boldsymbol{r}_2;\gamma_2,\theta_2) = \boldsymbol{V}\left(\theta_2\,\mathrm{Diag}(\boldsymbol{s})^2 + \gamma_2 \boldsymbol{I}\right)^{-1}\left(\theta_2\,\mathrm{Diag}(\boldsymbol{s})\boldsymbol{U}^\mathsf{T}\boldsymbol{y} + \gamma_2 \boldsymbol{V}^\mathsf{T}\boldsymbol{r}_2\right)$$

$$\eta_2 = \tfrac{1}{N}\textstyle\sum_{n=1}^N (\theta_2 s_n^2 + \gamma_2)^{-1}$$

two mat-vec mults per iteration!

Outline

PRS-ADMM

- Consider the optimization problem

$$\arg\min_{\boldsymbol{x}} f_1(\boldsymbol{x}) + f_2(\boldsymbol{x}) \quad \text{with, e.g.,} \quad \begin{cases} f_1(\boldsymbol{x}) = -\log p(\boldsymbol{x};\boldsymbol{\theta}_1) \\ f_2(\boldsymbol{x}) = \frac{\theta_2}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{y}\|^2 \end{cases}$$

  and define the augmented Lagrangian

$$L_\gamma(\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{s}) = f_1(\boldsymbol{x}_1) + f_2(\boldsymbol{x}_2) + \boldsymbol{s}^\mathsf{T}(\boldsymbol{x}_1-\boldsymbol{x}_2) + \frac{\gamma}{2}\|\boldsymbol{x}_1-\boldsymbol{x}_2\|^2.$$

- An ADMM variant (via Peaceman-Rachford splitting on the dual) is

$$\begin{aligned} \widehat{\boldsymbol{x}}_1 &\leftarrow \arg\min_{\boldsymbol{x}_1} L_\gamma(\boldsymbol{x}_1,\widehat{\boldsymbol{x}}_2,\boldsymbol{s}) \\ \boldsymbol{s} &\leftarrow \boldsymbol{s} + \gamma(\widehat{\boldsymbol{x}}_1-\widehat{\boldsymbol{x}}_2) \\ \widehat{\boldsymbol{x}}_2 &\leftarrow \arg\min_{\boldsymbol{x}_2} L_\gamma(\widehat{\boldsymbol{x}}_1,\boldsymbol{x}_2,\boldsymbol{s}) \\ \boldsymbol{s} &\leftarrow \boldsymbol{s} + \gamma(\widehat{\boldsymbol{x}}_1-\widehat{\boldsymbol{x}}_2) \end{aligned}$$

- PRS-ADMM has weaker convergence guarantees than standard ADMM, but is supposedly faster.

VAMP Connections to PRS-ADMM

- Now consider VAMP applied to the same optimization problem, but with $\gamma_1 = \gamma_2 \triangleq \gamma$ enforced at each iteration. Also, define

$$\boldsymbol{s}_i \triangleq \gamma(\widehat{\boldsymbol{x}}_i - \boldsymbol{r}_i) \text{ for } i = 1, 2.$$

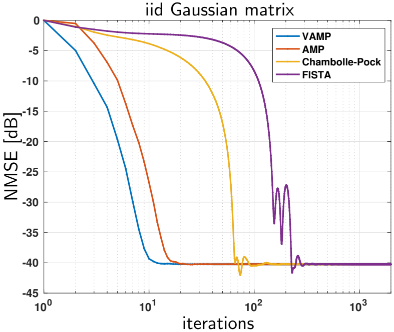
- This γ-forced VAMP manifests as

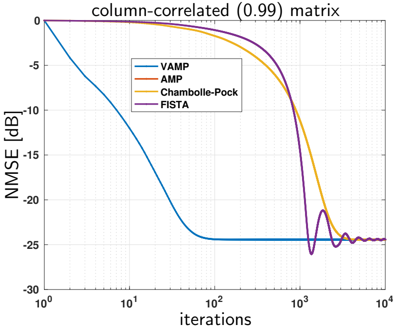
$$\begin{aligned}
\widehat{\boldsymbol{x}}_1 &\leftarrow \arg\min_{\boldsymbol{x}_1} L_\gamma(\boldsymbol{x}_1, \widehat{\boldsymbol{x}}_2, \boldsymbol{s}_1) \\
\boldsymbol{s}_2 &\leftarrow \boldsymbol{s}_1 + \gamma(\widehat{\boldsymbol{x}}_1 - \widehat{\boldsymbol{x}}_2) \\
\widehat{\boldsymbol{x}}_2 &\leftarrow \arg\min_{\boldsymbol{x}_2} L_\gamma(\widehat{\boldsymbol{x}}_1, \boldsymbol{x}_2, \boldsymbol{s}_2) \\
\boldsymbol{s}_1 &\leftarrow \boldsymbol{s}_2 + \gamma(\widehat{\boldsymbol{x}}_1 - \widehat{\boldsymbol{x}}_2)
\end{aligned}$$

  which is identical to Peaceman-Rachford ADMM.

- The full VAMP algorithm adapts γ_1 and γ_2 on-the-fly according to the local curvature of the cost function.

Example of VAMP applied to the LASSO Problem





Solving LASSO to reconstruct 40-sparse $\boldsymbol{x} \in \mathbb{R}^{1000}$ from noisy $\boldsymbol{y} \in \mathbb{R}^{400}$.

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda \|\boldsymbol{x}\|_1.$$

VAMP Convergence in the Convex Setting

- Consider arbitrary $\boldsymbol{A}$.
- A double-loop version of VAMP globally converges to a unique minimum when the Jacobian of the denoiser $\boldsymbol{g}_1$ is bounded as:

$$\exists c_1, c_2 > 0 \text{ s.t. } \frac{\gamma}{\gamma + c_1}\boldsymbol{I} \leq \frac{\partial \boldsymbol{g}_1(\boldsymbol{r}, \gamma)}{\partial \boldsymbol{r}} \leq \frac{\gamma}{\gamma + c_2}\boldsymbol{I},$$

  as occurs in optimization-VAMP under strictly convex regularization $R(\cdot; \boldsymbol{\theta}_1)$.
- For convergence, it suffices to choose the damping parameter $\zeta \in (0, 1]$ as

$$\zeta \leq \frac{2\min\{\gamma_1, \gamma_2\}}{\gamma_1 + \gamma_2}.$$

  Thus
  - the damping parameter ζ can be adapted using γ_1, γ_2, and
  - damping is not needed (i.e., $\zeta = 1$ suffices) if $\gamma_1 = \gamma_2$.

Outline

VAMP State Evolution

- Suppose the denoiser $\boldsymbol{g}_1(\cdot)$ has identical scalar components $g_1(\cdot)$, where g_1 and g_1' are Lipschitz.

- Suppose that $\boldsymbol{A}$ is right-rotationally invariant, in that its SVD
$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^{\mathsf{T}}$$
has Haar $\boldsymbol{V}$ (i.e., uniformly distributed over the set of orthogonal matrices). Since $\boldsymbol{U}$ and $\boldsymbol{S}$ are arbitrary, this includes iid Gaussian $\boldsymbol{A}$ as a special case.

- In the large-system limit, one can prove[4] that VAMP is rigorously characterized by a scalar state-evolution (using techniques inspired by Bayati-Montanari'10).

- This state-evolution establishes
  1. the convergence of VAMP in the non-convex setting,
  2. the correctness of the denoising model $\boldsymbol{r}_1 = \boldsymbol{x}_o + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_1)$.

[4] Rangan,Schniter,Fletcher'16

VAMP state evolution

Assuming empirical convergence of $\{s_j\} \to S$ and $\{(r^0_{1,j}, x_{o,j})\} \to (R^0_1, X_o)$ and Lipschitz continuity of g and g', the VAMP state-evolution under $\widehat{\tau}_w = \tau_w$ is as follows:

for $t = 0, 1, 2, \ldots$

$$\mathcal{E}_1^t = \mathrm{E}\left\{\left[g\big(X_o + \mathcal{N}(0, \tau_1^t); \overline{\gamma}_1^t\big) - X_o\right]^2\right\} \qquad \text{MSE}$$

$$\overline{\alpha}_1^t = \mathrm{E}\left\{g'(X_o + \mathcal{N}(0, \tau_1^t); \overline{\gamma}_1^t)\right\} \qquad \text{divergence}$$

$$\overline{\gamma}_2^t = \overline{\gamma}_1^t \tfrac{1-\overline{\alpha}_1^t}{\overline{\alpha}_1^t}, \quad \tau_2^t = \tfrac{1}{(1-\overline{\alpha}_1^t)^2}\left[\mathcal{E}_1^t - \left(\overline{\alpha}_1^t\right)^2 \tau_1^t\right]$$

$$\mathcal{E}_2^t = \mathrm{E}\left\{\left[S^2/\tau_w + \overline{\gamma}_2^t\right]^{-1}\right\} \qquad \text{MSE}$$

$$\overline{\alpha}_2^t = \overline{\gamma}_2^t \, \mathrm{E}\left\{\left[S^2/\tau_w + \overline{\gamma}_2^t\right]^{-1}\right\} \qquad \text{divergence}$$

$$\overline{\gamma}_1^{t+1} = \overline{\gamma}_2^t \tfrac{1-\overline{\alpha}_2^t}{\overline{\alpha}_2^t}, \quad \tau_1^{t+1} = \tfrac{1}{(1-\overline{\alpha}_2^t)^2}\left[\mathcal{E}_2^t - \left(\overline{\alpha}_2^t\right)^2 \tau_2^t\right]$$

More complicated expressions for $\mathcal{E}_2^t$ and $\overline{\alpha}_2^t$ exist for the case when $\widehat{\tau}_w \neq \tau_w$.

Outline

VAMP for Inference

- Now consider VAMP applied to the "inference" or "MMSE" problem.
  - assume a prior $p(\boldsymbol{x};\boldsymbol{\theta}_1)$,
  - choose the denoiser as $\boldsymbol{g}_1(\boldsymbol{r}_1;\gamma_1,\boldsymbol{\theta}_1) = \mathrm{E}\{\boldsymbol{x} \,|\, \boldsymbol{r}_1 = \boldsymbol{x} + \mathcal{N}(0, \boldsymbol{I}/\gamma_1)\}$.

- What is the corresponding cost function in this case?

- What can we say about convergence and performance?

- Can we tune the hyperparameters $\boldsymbol{\theta} = [\boldsymbol{\theta}_1, \theta_2]$ if they are unknown?

Variational Inference

- Ideally, we would like to compute the exact posterior density

$$p(\boldsymbol{x}|\boldsymbol{y}) = \frac{p(\boldsymbol{x};\boldsymbol{\theta}_1)\ell(\boldsymbol{x};\theta_2)}{Z(\boldsymbol{\theta})} \quad \text{for} \quad Z(\boldsymbol{\theta}) \triangleq \int p(\boldsymbol{x};\boldsymbol{\theta}_1)\ell(\boldsymbol{x};\theta_2)\,\mathrm{d}\boldsymbol{x},$$

but the high-dimensional integral in $Z(\boldsymbol{\theta})$ is difficult to compute.

- We can avoid computing $Z(\boldsymbol{\theta})$ through variational optimization:

$$p(\boldsymbol{x}|\boldsymbol{y}) = \arg\min_b D\big(b(\boldsymbol{x})\big\|p(\boldsymbol{x}|\boldsymbol{y})\big) \text{ where } D(\cdot\|\cdot) \text{ is KL divergence}$$

$$= \arg\min_b \underbrace{D\big(b(\boldsymbol{x})\big\|p(\boldsymbol{x};\boldsymbol{\theta}_1)\big) + D\big(b(\boldsymbol{x})\big\|\ell(\boldsymbol{x};\theta_2)\big) + H\big(b(\boldsymbol{x})\big)}_{\text{Gibbs free energy}}$$

$$= \arg\min_{b_1,b_2,q} \underbrace{D\big(b_1(\boldsymbol{x})\big\|p(\boldsymbol{x};\boldsymbol{\theta}_1)\big) + D\big(b_2(\boldsymbol{x})\big\|\ell(\boldsymbol{x};\theta_2)\big) + H\big(q(\boldsymbol{x})\big)}_{\triangleq J_{\mathsf{Gibbs}}(b_1,b_2,q;\boldsymbol{\theta})}$$

$$\text{s.t. } b_1 = b_2 = q,$$

but the density constraint keeps the problem difficult.

Expectation Consistent Approximation

- In expectation-consistent approximation (EC)[5], the density constraint is relaxed to moment-matching constraints:

$$p(\boldsymbol{x}|\boldsymbol{y}) \approx \arg\min_{b_1,b_2,q} J_{\text{Gibbs}}(b_1,b_2,q;\boldsymbol{\theta})$$
$$\text{s.t.} \begin{cases} \mathrm{E}\{\boldsymbol{x}|b_1\} = \mathrm{E}\{\boldsymbol{x}|b_2\} = \mathrm{E}\{\boldsymbol{x}|q\} \\ \mathrm{tr}(\mathrm{Cov}\{\boldsymbol{x}|b_1\}) = \mathrm{tr}(\mathrm{Cov}\{\boldsymbol{x}|b_2\}) = \mathrm{tr}(\mathrm{Cov}\{\boldsymbol{x}|q\}). \end{cases}$$

- The stationary points of EC are the densities

$$\begin{aligned} b_1(\boldsymbol{x}) &\propto p(\boldsymbol{x};\boldsymbol{\theta}_1)\mathcal{N}(\boldsymbol{x};\boldsymbol{r}_1,\boldsymbol{I}/\gamma_1) \\ b_2(\boldsymbol{x}) &\propto \ell(\boldsymbol{x};\theta_2)\mathcal{N}(\boldsymbol{x};\boldsymbol{r}_2,\boldsymbol{I}/\gamma_2) \\ q(\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{x};\widehat{\boldsymbol{x}},\boldsymbol{I}/\eta) \end{aligned} \quad \text{s.t.} \begin{cases} \mathrm{E}\{\boldsymbol{x}|b_1\} = \mathrm{E}\{\boldsymbol{x}|b_2\} = \widehat{\boldsymbol{x}} \\ \mathrm{tr}(\mathrm{Cov}\{\boldsymbol{x}|b_1\}) = \mathrm{tr}(\mathrm{Cov}\{\boldsymbol{x}|b_2\}) = N/\eta, \end{cases}$$

where VAMP iteratively solves for the quantities $\boldsymbol{r}_1,\gamma_1,\boldsymbol{r}_2,\gamma_2,\widehat{\boldsymbol{x}},\eta$.

- For large right-rotationally invariant $\boldsymbol{A}$, the these stationary points are "good" in that MSE($\widehat{\boldsymbol{x}}$) matches the MMSE predicted by the replica method.[6,7]

[5] Opper,Winther'04, [6] Kabashima,Vehkaperä'14, [7] Fletcher,Sahraee,Rangan,Schniter'16

The VAMP Algorithm for Inference

When applied to inference, the VAMP algorithm manifests as

Initialize $\boldsymbol{r}_1, \gamma_1$.

For $k = 1, 2, 3, \ldots$

$\widehat{\boldsymbol{x}}_1 \leftarrow \boldsymbol{g}_1(\boldsymbol{r}_1; \gamma_1, \boldsymbol{\theta}_1)$ — MMSE estimate of $\boldsymbol{x} \sim p(\boldsymbol{x}; \boldsymbol{\theta}_1)$ from $\boldsymbol{r}_1 = \boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_1)$

$\eta_1 \leftarrow \gamma_1 N / \operatorname{tr}\left[\frac{\partial \boldsymbol{g}_1(\boldsymbol{r}_1; \gamma_1, \boldsymbol{\theta}_1)}{\partial \boldsymbol{r}_1}\right]$ — posterior precision

$\boldsymbol{r}_2 \leftarrow (\eta_1 \widehat{\boldsymbol{x}}_1 - \gamma_1 \boldsymbol{r}_1)/(\eta_1 - \gamma_1)$

$\gamma_2 \leftarrow \eta_1 - \gamma_1$

$\widehat{\boldsymbol{x}}_2 \leftarrow \boldsymbol{g}_2(\boldsymbol{r}_2; \gamma_2, \theta_2)$ — LMMSE estimate of $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{r}_2, \boldsymbol{I}/\gamma_2)$ from $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\theta_2)$

$\eta_2 \leftarrow \gamma_2 N / \operatorname{tr}\left[\frac{\partial \boldsymbol{g}_2(\boldsymbol{r}_2; \gamma_2, \theta_2)}{\partial \boldsymbol{r}_2}\right]$ — posterior precision

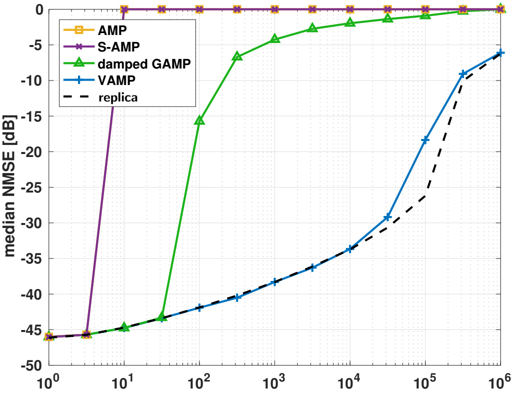
$\boldsymbol{r}_1 \leftarrow \zeta(\eta_2 \widehat{\boldsymbol{x}}_2 - \gamma_2 \boldsymbol{r}_2)/(\eta_2 - \gamma_2) + (1-\zeta)\boldsymbol{r}_1$

$\gamma_1 \leftarrow \zeta(\eta_2 - \gamma_2) + (1 - \zeta)\gamma_1$

and yields $\widehat{\boldsymbol{x}}_1 = \widehat{\boldsymbol{x}}_2 = \widehat{\boldsymbol{x}}$ and $\eta_1 = \eta_2 = \eta$ at a fixed point.

Experiment with Matched Priors

Comparison of several algorithms[8] with priors matched to data.



condition number $\kappa(\boldsymbol{A})$

$N = 1024$
$M/N = 0.5$

$\boldsymbol{A} = \boldsymbol{U}\,\mathrm{Diag}(\boldsymbol{s})\boldsymbol{V}^\mathsf{T}$
$\boldsymbol{U}, \boldsymbol{V} \sim$ Haar
$s_n/s_{n-1} = \phi\ \forall n$
ϕ determines $\kappa(\boldsymbol{A})$

$X_o \sim$Bernoulli-Gaussian
$\Pr\{X_0 \neq 0\} = 0.1$

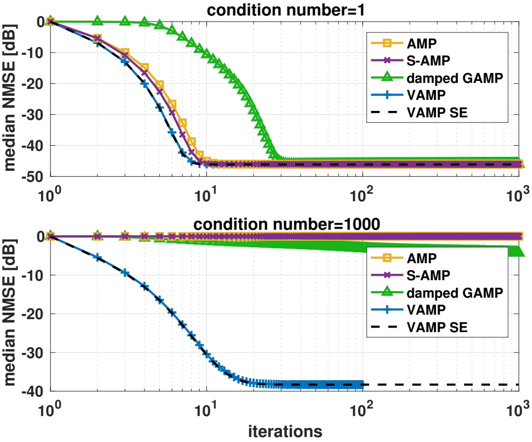
SNR$=40$dB

VAMP follows replica prediction[9] over a wide range of condition numbers.

[8] S-AMP: Cakmak,Fleury,Winther'14, AD-GAMP: Vila,Schniter,Rangan,Krzakala,Zdeborová'15
[9] Tulino,Caire,Verdú,Shamai'13

Experiment with Matched Priors

Comparison of several algorithms with priors matched to data.



$N = 1024$
$M/N = 0.5$

$\boldsymbol{A} = \boldsymbol{U}\,\mathrm{Diag}(\boldsymbol{s})\boldsymbol{V}^{\mathsf{T}}$
$\boldsymbol{U}, \boldsymbol{V} \sim$ Haar
$s_n / s_{n-1} = \phi \;\forall n$
ϕ determines $\kappa(\boldsymbol{A})$

X_o ∼Bernoulli-Gaussian
$\Pr\{X_0 \neq 0\} = 0.1$

SNR $=$ 40dB

VAMP is fast even when $\boldsymbol{A}$ is ill-conditioned.

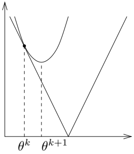
Outline

Expectation Maximization

- What if the hyperparameters $\boldsymbol{\theta}$ of the prior & likelihood are unknown?.
- The EM algorithm[10] is majorization-minimization approach to ML estimation that iteratively minimizes a tight upper bound on $-\ln p(\boldsymbol{y}|\boldsymbol{\theta})$:

$$\widehat{\boldsymbol{\theta}}^{k+1} = \arg\min_{\boldsymbol{\theta}} \Big\{ -\ln p(\boldsymbol{y}|\boldsymbol{\theta}) + \underbrace{D\big(b^k(\boldsymbol{x}) \big\| p(\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\theta})\big)}_{\geq 0} \Big\}$$

with $b^k(\boldsymbol{x}) = p(\boldsymbol{x}|\boldsymbol{y};\widehat{\boldsymbol{\theta}}^k)$



- We can also write EM in terms of the Gibbs free energy:[11]

$$\widehat{\boldsymbol{\theta}}^{k+1} = \arg\min_{\boldsymbol{\theta}} \underbrace{D\big(b^k(\boldsymbol{x}) \big\| p(\boldsymbol{x};\boldsymbol{\theta}_1)\big) + D\big(b^k(\boldsymbol{x}) \big\| \ell(\boldsymbol{x};\theta_2)\big) + H\big(b^k(\boldsymbol{x})\big)}_{J_{\mathsf{Gibbs}}(b^k, b^k, b^k; \boldsymbol{\theta})}$$

- Thus, we can interleave EM and VAMP to solve

$$\min_{\boldsymbol{\theta}} \min_{b_1,b_2,q} J_{\mathsf{Gibbs}}(b_1,b_2,q;\boldsymbol{\theta}) \text{ s.t. } \begin{cases} \mathrm{E}\{\boldsymbol{x}|b_1\} = \mathrm{E}\{\boldsymbol{x}|b_2\} = \mathrm{E}\{\boldsymbol{x}|q\} \\ \mathrm{tr}[\mathrm{Cov}\{\boldsymbol{x}|b_1\}] = \mathrm{tr}[\mathrm{Cov}\{\boldsymbol{x}|b_2\}] = \mathrm{tr}[\mathrm{Cov}\{\boldsymbol{x}|q\}]. \end{cases}$$

[10] Dempster,Laird,Rubin'77, [11] Neal,Hinton'98

The EM-VAMP Algorithm

Input conditional-mean $\boldsymbol{g}_1(\cdot)$ and $\boldsymbol{g}_2(\cdot)$, and initialize $\boldsymbol{r}_1, \gamma_1, \widehat{\boldsymbol{\theta}}_1, \widehat{\theta}_2$.

For $k = 1, 2, 3, \ldots$

$\widehat{\boldsymbol{x}}_1 \leftarrow \boldsymbol{g}_1(\boldsymbol{r}_1; \gamma_1, \widehat{\boldsymbol{\theta}}_1)$ — MMSE estimation

$\eta_1 \leftarrow \gamma_1 N / \operatorname{tr}\left[\partial \boldsymbol{g}_1(\boldsymbol{r}_1; \gamma_1, \widehat{\boldsymbol{\theta}}_1)/\partial \boldsymbol{r}_1\right]$

$\boldsymbol{r}_2 \leftarrow (\eta_1 \widehat{\boldsymbol{x}}_1 - \gamma_1 \boldsymbol{r}_1)/(\eta_1 - \gamma_1)$

$\gamma_2 \leftarrow \eta_1 - \gamma_1$

$\widehat{\theta}_2 \leftarrow \arg\max_{\theta_2} \mathrm{E}\{\ln \ell(\boldsymbol{x}; \theta_2) \,|\, \boldsymbol{r}_2; \gamma_2, \widehat{\theta}_2\}$ — EM update

---

$\widehat{\boldsymbol{x}}_2 \leftarrow \boldsymbol{g}_2(\boldsymbol{r}_2; \gamma_2, \widehat{\theta}_2)$ — LMMSE estimation

$\eta_2 \leftarrow \gamma_2 N / \operatorname{tr}\left[\partial \boldsymbol{g}_2(\boldsymbol{r}_2; \gamma_2, \widehat{\theta}_2)/\partial \boldsymbol{r}_2\right]$

$\boldsymbol{r}_1 \leftarrow \zeta(\eta_2 \widehat{\boldsymbol{x}}_2 - \gamma_2 \boldsymbol{r}_2)/(\eta_2 - \gamma_2) + (1-\zeta)\boldsymbol{r}_1$

$\gamma_1 \leftarrow \zeta(\eta_2 - \gamma_2) + (1 - \zeta)\gamma_1$

$\widehat{\boldsymbol{\theta}}_1 \leftarrow \arg\max_{\boldsymbol{\theta}_1} \mathrm{E}\{\ln p(\boldsymbol{x}; \boldsymbol{\theta}_1) \,|\, \boldsymbol{r}_1; \gamma_1, \widehat{\boldsymbol{\theta}}_1\}$ — EM update

Experiments suggest it helps to update $\widehat{\theta}_2$ several times per VAMP iteration.

State Evolution and Consistency

- EM-VAMP has a rigorous state-evolution when the prior is i.i.d. and $\boldsymbol{A}$ is large and right-rotationally invariant.[12]

- Furthermore, a variant known as "adaptive VAMP" can be shown to yield consistent parameter estimates with an i.i.d. prior in the exponential-family or with finite-cardinality $\boldsymbol{\theta}_1$.[12]

- Essentially, adaptive VAMP replaces the EM update

$$\widehat{\boldsymbol{\theta}}_1 \leftarrow \arg\max_{\boldsymbol{\theta}_1} \mathrm{E}\{\ln p(\boldsymbol{x};\boldsymbol{\theta}_1) \,|\, \boldsymbol{r}_1, \gamma_1, \widehat{\boldsymbol{\theta}}_1\}$$

with

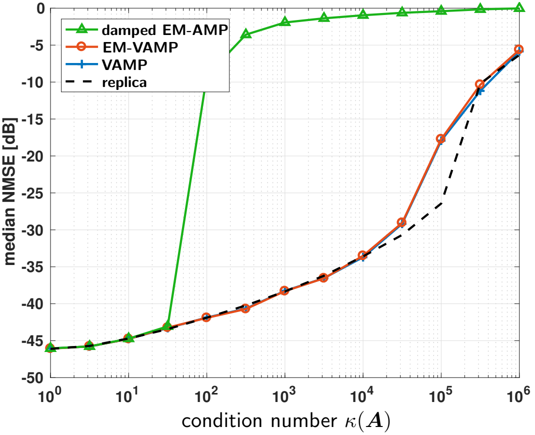
$$(\widehat{\boldsymbol{\theta}}_1, \widehat{\gamma}_1) \leftarrow \arg\max_{(\boldsymbol{\theta}_1,\gamma_1)} \mathrm{E}\{\ln p(\boldsymbol{x};\boldsymbol{\theta}_1) \,|\, \boldsymbol{r}_1, \gamma_1, \widehat{\boldsymbol{\theta}}_1\},$$

which also re-estimates the precision γ_1. (And similar for θ_2, γ_2.)

[12] Fletcher,Rangan,Schniter'17

Experiment with Unknown Hyperparameters $\boldsymbol{\theta}$

Learning both noise precision θ_2 and BG mean/variance/sparsity $\boldsymbol{\theta}_1$:



$N = 1024$
$M/N = 0.5$

$\boldsymbol{A} = \boldsymbol{U}\,\mathrm{Diag}(\boldsymbol{s})\boldsymbol{V}^{\mathsf{T}}$
$\boldsymbol{U}, \boldsymbol{V} \sim$ Haar
$s_n/s_{n-1} = \phi\ \forall n$
ϕ determines $\kappa(\boldsymbol{A})$

$X_o \sim$Bernoulli-Gaussian
$\Pr\{X_0 \neq 0\} = 0.1$

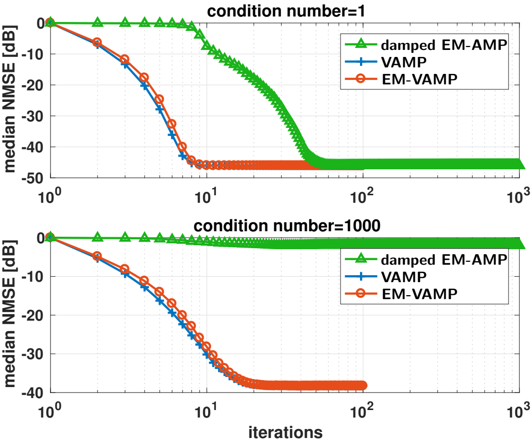
SNR $= 40$dB

EM-VAMP achieves oracle performance at all condition numbers![13]

[13] EM-AMP proposed in Vila,Schniter'11 and Krzakala,Mézard,Sausset,Sun,Zdeborová'12

Experiment with Unknown Hyperparameters $\boldsymbol{\theta}$

Learning both noise precision θ_2 and BG mean/variance/sparsity $\boldsymbol{\theta}_1$:



$N = 1024$
$M/N = 0.5$

$\boldsymbol{A} = \boldsymbol{U}\,\text{Diag}(\boldsymbol{s})\boldsymbol{V}^{\mathsf{T}}$
$\boldsymbol{U}, \boldsymbol{V} \sim$ Haar
$s_n/s_{n-1} = \phi \ \forall n$
ϕ determines $\kappa(\boldsymbol{A})$

X_o ~Bernoulli-Gaussian
$\Pr\{X_0 \neq 0\} = 0.1$

SNR $= 40\text{dB}$

EM-VAMP nearly as fast as VAMP and much faster than damped EM-GAMP.

Outline

Plug-and-play VAMP

- Recall that the nonlinear estimation step in VAMP (or AMP)

$$\widehat{\boldsymbol{x}}_1 \leftarrow \boldsymbol{g}_1(\boldsymbol{r}_1;\gamma_1) \text{ where } \boldsymbol{r}_1 = \boldsymbol{x}_o + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_1)$$

  can be interpreted as "denoising" the pseudo-measurement $\boldsymbol{r}_1$.

- For certain signal classes, very sophisticated non-scalar denoising procedures have been developed (e.g., BM3D for images).

- Such denoising procedures can be "plugged into" signal recovery algorithms like ADMM[14], AMP[15], or VAMP[16].

- For AMP and VAMP, the divergence can be approximated using Monte-Carlo:

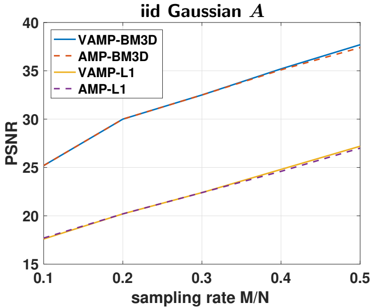
$$\frac{1}{N}\operatorname{tr}\left[\frac{\partial \boldsymbol{g}_1}{\partial \boldsymbol{r}_1}\right] \approx \frac{1}{K}\sum_{k=1}^{K}\frac{\boldsymbol{p}_k^{\mathsf{T}}\left[\boldsymbol{g}_1(\boldsymbol{r}+\epsilon\boldsymbol{p}_k,\gamma_1) - \boldsymbol{g}_1(\boldsymbol{r},\gamma_1)\right]}{N\epsilon}$$

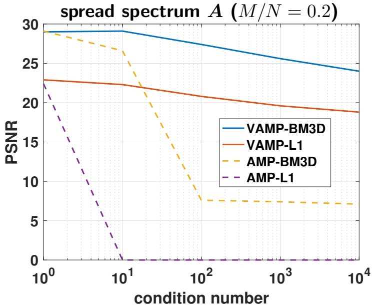
  with random vectors $\boldsymbol{p}_k \in \{\pm 1\}^N$ and small $\epsilon > 0$. Often, $K = 1$ suffices.

[14] Bouman et al'13, [15] Metzler,Maleki,Baraniuk'14, [16] Schniter,Rangan,Fletcher'16

Experiment: Image Recovery with Random Matrices

Plug-and-play versions of VAMP and AMP work similarly when $\boldsymbol{A}$ is i.i.d., but VAMP can handle a larger class of random matrices $\boldsymbol{A}$.





Results above are averaged over 128×128 versions of

lena, barbara, boat, fingerprint, house, peppers

and 10 random realizations of $\boldsymbol{A}, \boldsymbol{w}$.

Plug-and-play with Non-Random Matrices

- Many imaging applications (e.g., MRI) use low-frequency Fourier measurements, in which case $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\mathsf{T} = \boldsymbol{I}\,[\boldsymbol{I}\ \boldsymbol{0}]\,\boldsymbol{F}$.
- This causes problems for VAMP because the signal correlation structure interacts with $\boldsymbol{V}^\mathsf{T}$ in a way that VAMP is not designed to handle.
- Why? Say $\boldsymbol{x}$ is a natural image, and consider $\boldsymbol{q} = \boldsymbol{V}^\mathsf{T}\boldsymbol{x}$.
  - If $\boldsymbol{V}$ is large and Haar, then $\boldsymbol{q}$ will be iid Gaussian.
  - If $\boldsymbol{V}^\mathsf{T} = \boldsymbol{F}$, the low-freq entries of $\boldsymbol{q}$ will be much stronger than the others.

PnP VAMP treats $\boldsymbol{V}^\mathsf{T}\boldsymbol{x}$ as iid Gaussian and thus diverges when $\boldsymbol{V}^\mathsf{T} = \boldsymbol{F}$!

Whitened VAMP  for Image REcovery (VAMPire)

- To apply VAMP with non-random Fourier measurements, we propose to operate on the whitened signal:

$$\boldsymbol{y} = \underbrace{[\boldsymbol{I}\ \ \boldsymbol{0}]\boldsymbol{F}\boldsymbol{R}_x^{1/2}}_{\boldsymbol{A}}\boldsymbol{s} + \boldsymbol{w} \text{ for } \begin{cases} \boldsymbol{R}_x = \mathrm{E}\{\boldsymbol{x}\boldsymbol{x}^\mathsf{T}\} \\ \boldsymbol{s} = \text{whitened signal coefficients} \end{cases}$$

  and perform plug-and-play denoising from the whitened-coefficient space:

$$\widehat{\boldsymbol{s}}_1 = \boldsymbol{g}_1(\boldsymbol{r}_1, \gamma_1) = \boldsymbol{R}_x^{-1/2}\text{denoise}(\boldsymbol{R}_x^{1/2}\boldsymbol{r}_1; \gamma_1 N/\operatorname{tr}(\boldsymbol{R}_x)).$$

- In practice, we approximate $\boldsymbol{R}_x \approx \boldsymbol{W}^\mathsf{T}\,\mathrm{Diag}(\boldsymbol{\tau})^2\boldsymbol{W}$, where $\boldsymbol{W}$ is a wavelet transform and τ_i^2 specifies the energy of the ith wavelet coefficient (which is easy to predict for natural images).

Whitened VAMP  for Image REcovery (VAMPire)

- The resulting matrix $\boldsymbol{A} = [\boldsymbol{I}\ \boldsymbol{0}]\boldsymbol{F}\boldsymbol{W}\,\mathrm{Diag}(\boldsymbol{\tau})$ does not yield a right singular vector matrix $\boldsymbol{V}$ with a fast multiplication.

- But since $\boldsymbol{A}$ has a fast implementation, the LMMSE stage can be computed via (preconditioned) LSQR:

$$\boldsymbol{g}_2(\boldsymbol{r}_2;\gamma_2) = (\gamma_w \boldsymbol{A}^\mathsf{T}\boldsymbol{A} + \gamma_2 \boldsymbol{I})^{-1}(\gamma_w \boldsymbol{A}^\mathsf{T}\boldsymbol{y} + \gamma_2 \boldsymbol{r}_2) = \begin{bmatrix} \sqrt{\gamma_w}\boldsymbol{A} \\ \sqrt{\gamma_2}\boldsymbol{I} \end{bmatrix}^+ \begin{bmatrix} \sqrt{\gamma_w}\boldsymbol{y} \\ \sqrt{\gamma_2}\boldsymbol{r}_2 \end{bmatrix}$$

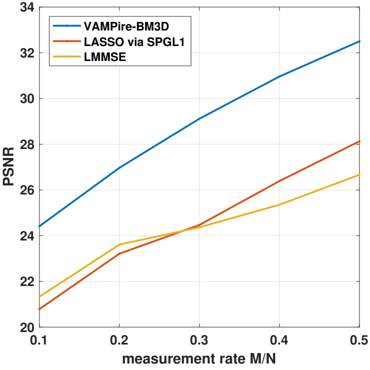
- The divergence $\langle \boldsymbol{g}_2'(\boldsymbol{r}_2;\gamma_2)\rangle$ can be approximated using Monte-Carlo:

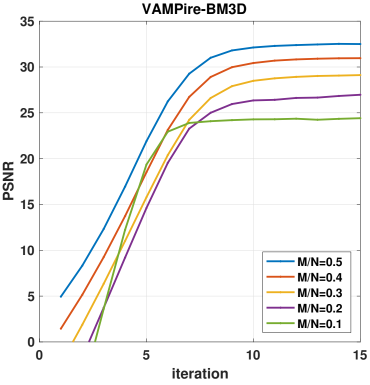
$$\langle \boldsymbol{g}_2' \rangle = \frac{\gamma_2}{N}\,\mathrm{tr}\left[\left(\gamma_w \boldsymbol{A}^\mathsf{H}\boldsymbol{A} + \gamma_2 \boldsymbol{I}\right)^{-1}\right] \approx \frac{1}{NK}\sum_{k=1}^{K} \boldsymbol{p}_k \begin{bmatrix} \sqrt{\gamma_w}\boldsymbol{A} \\ \sqrt{\gamma_2}\boldsymbol{I} \end{bmatrix}^+ \begin{bmatrix} \boldsymbol{0} \\ \sqrt{\gamma_2}\boldsymbol{p}_k \end{bmatrix},$$

  where $\mathrm{E}\{\boldsymbol{p}_k\boldsymbol{p}_k^\mathsf{H}\} = \boldsymbol{I}$. Here again, (preconditioned) LSQR can be used.
  In practice, $K = 1$ suffices.

Image Recovery Experiments

- Fourier measurements sampled at M lowest frequencies
- SNR=40dB
- 128×128 images {*lena, barbara, boat, fingerprint, house, peppers*}
- db1 wavelet decomposition, $D = 2$ levels

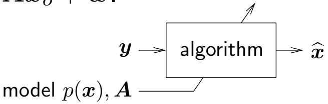




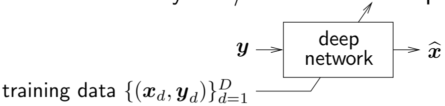
Outline

Deep learning for sparse reconstruction

- Until now we've focused on designing algorithms to recover $\boldsymbol{x}_o \sim p(\boldsymbol{x})$ from measurements $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_o + \boldsymbol{w}$.



- What about training deep networks to predict $\boldsymbol{x}_o$ from $\boldsymbol{y}$?
  Can we increase accuracy and/or decrease computation?



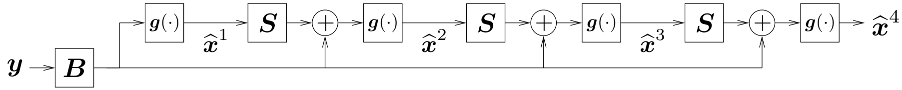
- Are there connections between these approaches?

Unfolding Algorithms into Networks

Consider, e.g., the classical sparse-reconstruction algorithm, ISTA.[17]

$$\begin{aligned} \boldsymbol{v}^t &= \boldsymbol{y} - \boldsymbol{A}\widehat{\boldsymbol{x}}^t \\ \widehat{\boldsymbol{x}}^{t+1} &= \boldsymbol{g}(\widehat{\boldsymbol{x}}^t + \boldsymbol{A}^\mathsf{T}\boldsymbol{v}^t) \end{aligned} \quad \Leftrightarrow \quad \widehat{\boldsymbol{x}}^{t+1} = \boldsymbol{g}(\boldsymbol{S}\widehat{\boldsymbol{x}}^t + \boldsymbol{B}\boldsymbol{y}) \text{ with } \begin{aligned} \boldsymbol{S} &\triangleq \boldsymbol{I} - \boldsymbol{A}^\mathsf{T}\boldsymbol{A} \\ \boldsymbol{B} &\triangleq \boldsymbol{A}^\mathsf{T} \end{aligned}$$

Gregor & LeCun[18] proposed to "unfold" it into a deep net and "learn" improved parameters using training data, yielding "learned ISTA" (LISTA):

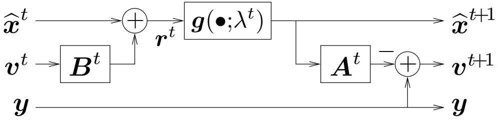


The same "unfolding & learning" idea can be used to improve AMP, yielding "learned AMP" (LAMP).[19]

[17] Daubechies,Defrise,DeMol'04. [18] Gregor,LeCun'10. [19] Borgerding,Schniter'16.

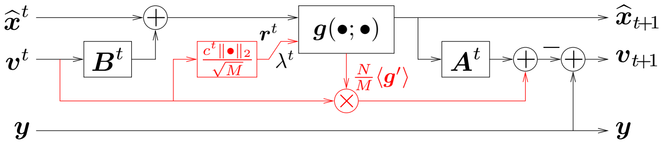
Onsager-Corrected Deep Networks

t^{th} LISTA layer:



to exploit low-rank $B^t A^t$ in linear stage $S^t = I - B^t A^t$.

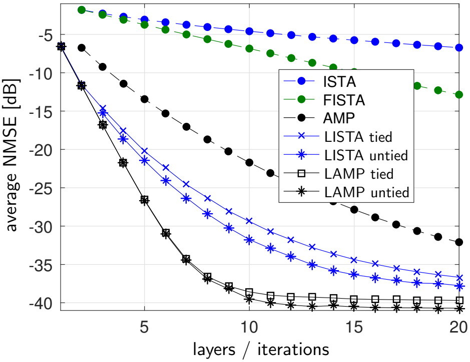
t^{th} LAMP layer:

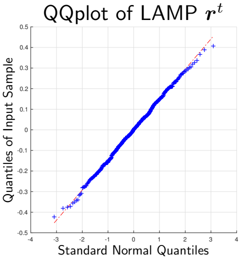


Onsager correction now aims to decouple errors across layers.

LAMP performance with soft-threshold denoising

LISTA beats AMP,FISTA,ISTA
LAMP beats LISTA
in convergence speed and asymptotic MSE.

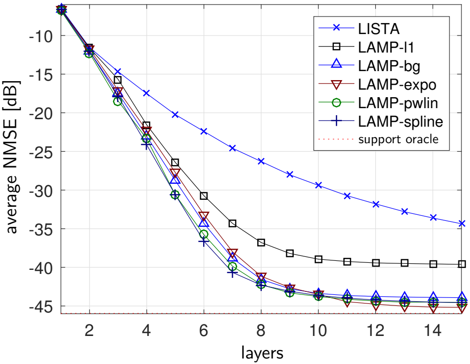




LAMP beyond soft-thresholding

So far, we used soft-thresholding to isolate the effects of Onsager correction.

What happens with more sophisticated (learned) denoisers?



Here we learned the parameters of these denoiser families:

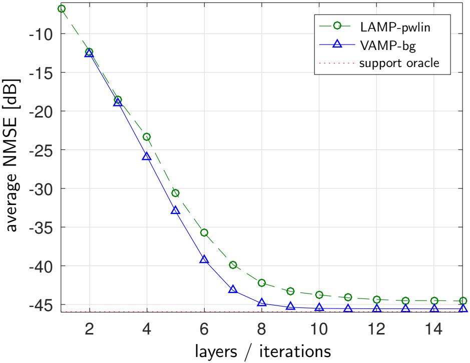
- scaled soft-thresholding
- conditional mean under BG
- Exponential kernel[20]
- Piecewise Linear[20]
- Spline[21]

Big improvement!

[20] Guo,Davies'15. [21] Kamilov,Mansour'16.

LAMP  versus VAMP 

How does our best Learned AMP compare to (unlearned) VAMP?

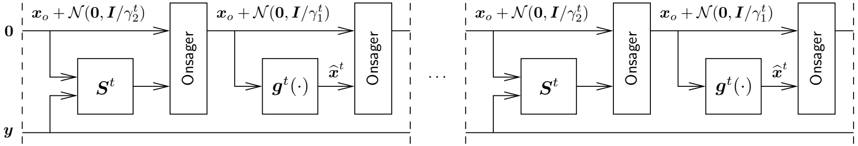


VAMP wins!

So what about "learned VAMP"?

Learned VAMP

- Suppose we unfold VAMP and learn (via backprop) the parameters $\{\boldsymbol{S}^t, \boldsymbol{g}^t\}_{t=1}^T$ that minimize the training MSE.



- Remarkably, backpropagation does not improve matched VAMP!

  VAMP is locally optimal

- Onsager correction decouples the design of $\{\boldsymbol{S}^t, \boldsymbol{g}^t(\cdot)\}_{t=1}^T$:

  Layer-wise optimal $\boldsymbol{S}^t, \boldsymbol{g}^t(\cdot)$ $\Rightarrow$ Network optimal $\{\boldsymbol{S}^t, \boldsymbol{g}^t(\cdot)\}_{t=1}^T$

Outline

Generalized linear models

- Until now we have considered linear regression: $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_o + \boldsymbol{w}$.
- VAMP can also be applied to the generalized linear model (GLM)[23]

$$\boldsymbol{y} \sim p(\boldsymbol{y}|\boldsymbol{z}) \text{ with hidden } \boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}_o$$

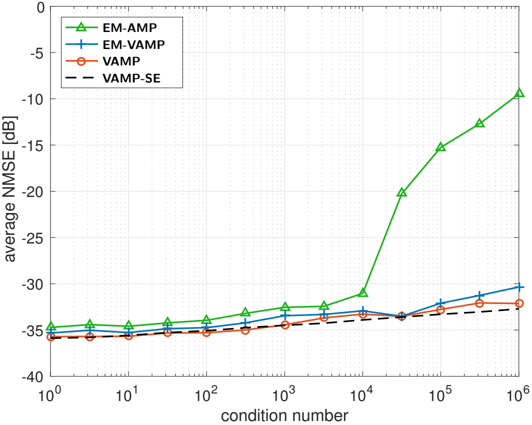
  which supports, e.g.,
  - $y_i = z_i + w_i$: additive, possibly non-Gaussian noise
  - $y_i = \text{sgn}(z_i + w_i)$: binary classification / one-bit quantization
  - $y_i = |z_i + w_i|$: phase retrieval in noise
  - Poisson y_i: photon-limited imaging
- How? A simple trick turns the GLM into a linear regression problem:

$$\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x} \quad \Leftrightarrow \quad \underbrace{\boldsymbol{0}}_{\widetilde{\boldsymbol{z}}} = \underbrace{\begin{bmatrix} \boldsymbol{A} & -\boldsymbol{I} \end{bmatrix}}_{\widetilde{\boldsymbol{A}}} \underbrace{\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{z} \end{bmatrix}}_{\widetilde{\boldsymbol{x}}}$$

[23] Schniter, Rangan, Fletcher'16

One-bit compressed sensing / Probit regression

Learning both θ_2 and $\boldsymbol{\theta}_1$:



$N = 512$
$M/N = 4$

$\boldsymbol{A} = \boldsymbol{U}\,\text{Diag}(\boldsymbol{s})V^{\mathsf{T}}$
$\boldsymbol{U}, \boldsymbol{V}$ drawn uniform
$s_n/s_{n-1} = \phi \ \forall n$
ϕ determines $\kappa(\boldsymbol{A})$

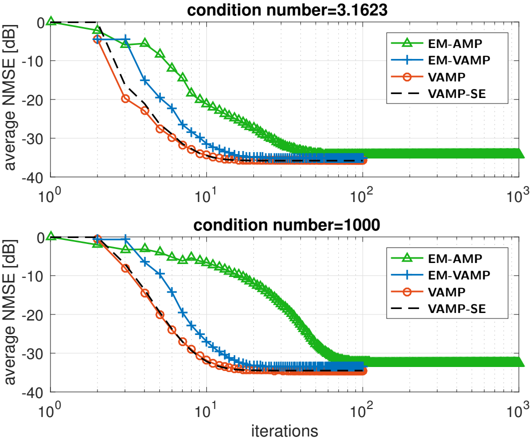
X_o ~Bernoulli-Gaussian
$\Pr\{X_0 \neq 0\} = 1/32$

SNR $= 40$dB

VAMP and EM-VAMP robust to ill-conditioned $\boldsymbol{A}$.

One-bit compressed sensing / Probit regression

Learning both θ_2 and $\boldsymbol{\theta}_1$:



$N = 512$
$M/N = 4$

$\boldsymbol{A} = \boldsymbol{U}\,\mathrm{Diag}(\boldsymbol{s})V^{\mathsf{T}}$
$\boldsymbol{U}, \boldsymbol{V}$ drawn uniform
$s_n/s_{n-1} = \phi\ \forall n$
ϕ determines $\kappa(\boldsymbol{A})$

X_o ~Bernoulli-Gaussian
$\Pr\{X_0 \neq 0\} = 1/32$

SNR $= 40$dB

EM-VAMP mildly slower than VAMP but much faster than damped AMP.

Conclusions   

- VAMP is an efficient algorithm for linear and generalized-linear regression.
- For convex optimization problems, VAMP is provably convergent and related to Peaceman-Rachford ADMM.
- For inference under right rotationally-invariant $\boldsymbol{A}$, VAMP has a rigorous state evolution and fixed-points that agree with the replica MMSE prediction.
- VAMP can be combined with EM to handle priors/likelihood with unknown parameters, again with a rigorous state evolution.
- Can unfold VAMP into an interpretable deep network.
- In non-convex settings (e.g., plug-and-play) with deterministic matrices, more work is needed to understand the performance and convergence of VAMP.
- Still lots to do! (multilayer generative models, bilinear problems . . .)