Why LAMP?

• AMP is very appealing for its efficiency and accurate recovery.
• It approximates the computationally intractable high-dimensional integration involved with calculating
  $$\hat{x} = \mathbb{E}(x | y).$$
• Its parameters can be learned from training data ⇒ LAMP.
• LAMP significantly improves upon both LIStA and AMP.
• Is the chosen parametric family of denoisers good for the source prior? We don’t know!

Main Idea

• We model the source signal prior as an independent and identically distributed (i.i.d.) Gaussian-mixture (GM) distribution.
• We adopt the optimal denoiser function that would minimize MSE had the assumed GM prior match the true unknown prior.
• The parameters of the GM are learned from training data.
• We, therefore, examine a general-purpose denoiser within the LAMP algorithm.

Why GM?

• The resulting denoiser function $$\eta(\cdot)$$, and its derivative $$\frac{\partial \eta(\cdot)}{\partial \theta}$$ can be calculated analytically.
• If the overall objective is to minimize MSE, a good approximation of a discrete component in the source prior is a Gaussian distribution with matching mean and a very small variance.
• A Gaussian mixture can model a variety of continuous distributions.

Learned Gaussian-mixture AMP

The Steps of LAMP

• The LAMP algorithm is initialized (at $$t = 0$$) according to
  $$\hat{x} = 0_{K \times 1}, \quad z^{(0)} = 0_{m \times 1}.$$
• At every iteration $$t = 1, 2, \ldots, T_{\text{max}}$$, the algorithm computes
  $$\begin{align*}
  \hat{x}^{(t)} &= x^{(t-1)} + Bz^{(t)}; \\
  z^{(t)} &= \eta(\hat{x}^{(t)}, \sigma^{(t)}, \Theta^{(t)}); \\
  \hat{y}^{(t)} &= \frac{1}{m} \sum_{m} \partial_{\hat{y}^{(t)}} \partial_{x^{(t)}} \hat{y}^{(t)} \\
  z^{(t)} &= \hat{y} - \hat{x}^{(t-1)} + \hat{y}^{(t-1)} x^{(t-1)}.
  \end{align*}$$
• $$B$$ is the learned weight (filter) matrix.
• $$\sigma^{(t)}$$ is the effective noise variance at the $$t$$-th layer, which can be estimated with $$\sigma^{(t)} = ||\hat{y}^{(t)}||/\sqrt{m}$$.
• The denoiser function $$\eta(\cdot, \cdot, \cdot)$$ that minimizes MSE is given by
  $$\eta(\hat{x}^{(t)}, \sigma^{(t)}, \Theta^{(t)}) = \mathbb{E}[(x^{(t)} - \hat{x}^{(t)})/\sigma^{(t)}].$$
• The Onsager term allows for the decoupled measurement model, i.e., $$x^{(t)} \sim x + \nu^{(t)}$$, where $$\nu^{(t)} \sim N(0, \sigma^{(t)} I_{N})$$.

GM prior distribution

• We assume GM prior distribution:
  $$p(x; \Theta_{\text{GM}}) = \sum_{i=0}^{I} \omega \mathcal{N}(x; \mu_{i}, \sigma^{2}_{i}),$$
  where $$\sum_{i=1}^{I} \omega_{i} = 1, 0 \leq \omega_{i} \leq 1, \forall i \in [I].$$
• ⇒ the conditional pdf of $$x$$ given $$\Theta_{\text{GM}}$$ can be written as
  $$p(x_{j}| x_{\setminus j}; \Theta_{\text{GM}}) = \sum_{i=1}^{I} \tilde{\omega}_{i} \mathcal{N}(x_{j}; \gamma_{ij}, \sigma^{2}_{i}).$$

Learning the L-GM-AMP Parameters

• A network with $$T_{\text{max}}$$ layers which has $$N m + 3L T_{\text{max}}$$ tunable parameters $$B L_{j=1}^{T_{\text{max}}} [\omega_{i}, \mu_{i}, \sigma^{2}_{i}].$$

Algorithm 1: L-GM-AMP parameter learning

1. Initialize $$\Theta_{\text{GM}}^{(0)} = \Theta_{\text{GM}}^{(1)}$$ (GM)
2. Learn $$\Theta_{\text{GM}}^{(j)}$$ with fixed $$\Theta_{\text{AMP}}^{(j)}$$.
3. Refine $$\Theta_{\text{AMP}}^{(j)} = (B, L_{j=1}^{T_{\text{max}}} \Theta_{\text{GM}}^{(j)})$$.

Conclusion

• Although reminiscent of Borgerding’s LAMP [1], it differs in the adoption of a universal plug and play denoising function.
• L-GM-AMP algorithm achieves state-of-the-art performance offered by (L)AMP with perfect knowledge of the source prior.