ADAPTIVE DAMPING AND MEAN REMOVAL FOR THE GENERALIZED APPROXIMATE MESSAGE PASSING ALGORITHM

Jeremy Vila[⋆] *Philip Schniter*[⋆] *Sundeep Rangan*[†] *Florent Krzakala*[‡] *Lenka Zdeborová*[◦]

[⋆] Dept. of ECE, The Ohio State University, Columbus, OH 43202, USA.
[†] Dept. of ECE, Polytechnic Institute of New York University, Brooklyn, NY 11201, USA.
[‡] Sorbonne Universités, UPMC Univ Paris 06 and École Normale Supérieure, 75005 Paris, France.
[◦] Institut de Physique Théorique, CEA Saclay, and CNRS URA 2306, 91191 Gif-sur-Yvette, France.

ABSTRACT

The generalized approximate message passing (GAMP) algorithm is an efficient method of MAP or approximate-MMSE estimation of $\boldsymbol{x}$ observed from a noisy version of the transform coefficients $\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}$. In fact, for large zero-mean i.i.d sub-Gaussian $\boldsymbol{A}$, GAMP is characterized by a state evolution whose fixed points, when unique, are optimal. For generic $\boldsymbol{A}$, however, GAMP may diverge. In this paper, we propose adaptive-damping and mean-removal strategies that aim to prevent divergence. Numerical results demonstrate significantly enhanced robustness to non-zero-mean, rank-deficient, column-correlated, and ill-conditioned $\boldsymbol{A}$.

Index Terms— Approximate message passing, belief propagation, compressed sensing.

1. INTRODUCTION

Consider estimating a realization $\boldsymbol{x} \in \mathbb{R}^N$ of a random vector $\boldsymbol{\mathsf{x}}$ with statistically independent components $\mathsf{x}_n \sim p_{\mathsf{x}_n}$ from observations $\boldsymbol{y} = [y_m] \in \mathbb{R}^M$ that are conditionally independent given the transform outputs

$$\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x} \tag{1}$$

for some known matrix $\boldsymbol{A} = [a_{mn}] \in \mathbb{R}^{M\times N}$. Here, the likelihood function can be written as $p_{\boldsymbol{\mathsf{y}}|\boldsymbol{\mathsf{z}}}(\boldsymbol{y}|\boldsymbol{A}\boldsymbol{x})$ with separable $p_{\boldsymbol{\mathsf{y}}|\boldsymbol{\mathsf{z}}}$, i.e., $p_{\boldsymbol{\mathsf{y}}|\boldsymbol{\mathsf{z}}}(\boldsymbol{y}|\boldsymbol{z}) = \prod_{m=1}^M p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z_m)$. Such problems arise in a range of applications including statistical regression, inverse problems, and compressive sensing. Note that, for clarity, we use sans-serif fonts (e.g., $\boldsymbol{\mathsf{x}}, \mathsf{x}_n$) to denote random quantities and serif fonts (e.g., $\boldsymbol{x}, x_n$) to denote deterministic ones.

Assuming knowledge of the prior $p_{\boldsymbol{\mathsf{x}}}(\boldsymbol{x}) = \prod_{n=1}^N p_{\mathsf{x}_n}(x_n)$ and likelihood $p_{\boldsymbol{\mathsf{y}}|\boldsymbol{\mathsf{z}}}(\boldsymbol{y}|\boldsymbol{z})$, typical estimation goals are to compute the minimum mean-squared error (MMSE) estimate $\widehat{\boldsymbol{x}}_{\mathsf{MMSE}} \triangleq \int_{\mathbb{R}^N} \boldsymbol{x} p_{\boldsymbol{\mathsf{x}}|\boldsymbol{\mathsf{y}}}(\boldsymbol{x}|\boldsymbol{y}) d\boldsymbol{x}$ or the maximum a posteriori (MAP) estimate $\widehat{\boldsymbol{x}}_{\mathsf{MAP}} \triangleq \arg\max_{\boldsymbol{x}} p_{\boldsymbol{\mathsf{x}}|\boldsymbol{\mathsf{y}}}(\boldsymbol{x}|\boldsymbol{y}) = \arg\min_{\boldsymbol{x}} J_{\mathsf{MAP}}(\boldsymbol{x})$ for the MAP cost

$$J_{\mathsf{MAP}}(\widehat{\boldsymbol{x}}) \triangleq -\ln p_{\boldsymbol{\mathsf{y}}|\boldsymbol{\mathsf{z}}}(\boldsymbol{y}|\boldsymbol{A}\widehat{\boldsymbol{x}}) - \ln p_{\boldsymbol{\mathsf{x}}}(\widehat{\boldsymbol{x}}). \tag{2}$$

Recently, the *generalized approximate message passing* (GAMP) algorithm [1] has been proposed as a means of tackling these two problems in the case that M and N are large. Essentially, GAMP uses a high-dimensional approximation of loopy belief propagation to convert the MMSE or MAP inference problems into a sequence of tractable scalar inference problems.

GAMP is well motivated in the case that $\boldsymbol{A}$ is a realization of a large random matrix with i.i.d zero-mean sub-Gaussian entries. For such $\boldsymbol{A}$, in the large-system limit (i.e., $M, N \to \infty$ for fixed $M/N \in \mathbb{R}_+$), GAMP is characterized by a state evolution whose fixed points, when unique, are MMSE or MAP optimal [1–3]. Furthermore, for *generic* $\boldsymbol{A}$, it has been shown [4] that MAP-GAMP's fixed points coincide with the critical points of the cost function (2) and that MMSE-GAMP's fixed points coincide with those of a Bethe free entropy [5], as discussed in detail in Section 2.2.

For generic $\boldsymbol{A}$, however, GAMP may not reach its fixed points, i.e., it may diverge (e.g., [6]). The convergence of GAMP has been fully characterized in [7] for the simple case that p_{x_n} and $p_{\mathsf{y}_m|\mathsf{z}_m}$ are Gaussian. There, it was shown that Gaussian-GAMP converges if and only if the peak-to-average ratio of the squared singular values of $\boldsymbol{A}$ is sufficiently small. A *damping* modification was then proposed in [7] that guarantees the convergence of Gaussian-GAMP with arbitrary $\boldsymbol{A}$, at the expense of a slower convergence rate. For strictly log-concave p_{x_n} and $p_{\mathsf{y}_m|\mathsf{z}_m}$, the *local* convergence of GAMP was also characterized in [7]. However, the global convergence of GAMP under generic $\boldsymbol{A}$, p_{x_n}, and $p_{\mathsf{y}_m|\mathsf{z}_m}$ is not yet understood.

Because of its practical importance, prior work has attempted to robustify the convergence of GAMP in the face of "difficult" $\boldsymbol{A}$ (e.g., high peak-to-average singular values) for generic p_{x_n} and $p_{\mathsf{y}_m|\mathsf{z}_m}$. For example, "swept" GAMP (SwAMP) [8] updates the estimates of $\{\mathsf{x}_n\}_{n=1}^N$ and $\{\mathsf{z}_m\}_{m=1}^M$ sequentially, in contrast to GAMP, which updates them in parallel. Relative to GAMP, experiments in [8] show that SwAMP is much more robust to difficult $\boldsymbol{A}$, but it is slower and cannot facilitate fast implementations of $\boldsymbol{A}$ like an FFT. As another example, the public-domain GAMPmatlab implementation [9] has included "adaptive damping" and "mean removal" mechanisms for some time, but they have never been described in the literature.

In this paper, we detail the most recent versions of GAMPmatlab's adaptive damping and mean-removal mechanisms, and we experimentally characterize their performance on non-zero-mean, rank-deficient, column-correlated, and ill-conditioned $\boldsymbol{A}$ matrices. Our results show improved robustness relative to SwAMP and enhanced convergence speed.

2. ADAPTIVELY DAMPED GAMP

Damping is commonly used in loopy belief propagation to "slow down" the updates in an effort to promote convergence. (See, e.g., [10] for damping applied to the sum-product algorithm and [7, 9, 11] for damping applied to GAMP.) However, since not enough damping

This work has been supported in part by NSF grants IIP-0968910, CCF-1018368, and CCF-1218754, an allocation of computing time from the Ohio Supercomputer Center, and by European Unions 7th Framework Programme (FP/2007-2013)/ERC Grant Agreement 307087-SPARCS.

definitions for MMSE-GAMP:

$g_{\mathsf{z}_m}(\widehat{p},\nu^p) \triangleq \int z\, f_{\mathsf{z}_m}(z;\widehat{p},\nu^p)dz$ (D1)

for $f_{\mathsf{z}_m}(z;\widehat{p},\nu^p) \triangleq \frac{p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z)\mathcal{N}(z;\widehat{p},\nu^p)}{B_m(\widehat{p},\nu^p)}$

and $B_m(\widehat{p},\nu^p) \triangleq \int p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z)\mathcal{N}(z;\widehat{p},\nu^p)dz$

$g_{\mathsf{x}_n}(\widehat{r},\nu^r) \triangleq \int x\, f_{\mathsf{x}_n}(x;\widehat{r},\nu^r)dx$ (D2)

for $f_{\mathsf{x}_n}(x;\widehat{r},\nu^r) \triangleq \frac{p_{\mathsf{x}_n}(x)\mathcal{N}(x;\widehat{r},\nu^r)}{C_n(\widehat{r},\nu^r)}$

and $C_n(\widehat{r},\nu^r) \triangleq \int p_{\mathsf{x}_n}(x)\mathcal{N}(x;\widehat{r},\nu^r)dx$

definitions for MAP-GAMP:

$g_{\mathsf{z}_m}(\widehat{p},\nu^p) \triangleq \arg\max_z \ln p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z) + \frac{1}{2\nu^p}|z-\widehat{p}|^2$ (D3)

$g_{\mathsf{x}_n}(\widehat{r},\nu^r) \triangleq \arg\max_x \ln p_{\mathsf{x}_n}(x) + \frac{1}{2\nu^r}|x-\widehat{r}|^2$ (D4)

inputs:

$\forall m,n:\, g_{\mathsf{z}_m}, g_{\mathsf{x}_n}, \widehat{x}_n(1), \nu^x_n(1), a_{mn}, T_{\max} \geq 1, \epsilon \geq 0$

$T_\beta \geq 0, \beta_{\max} \in (0,1], \beta_{\min} \in [0,\beta_{\max}], G_{\text{pass}} \geq 1, G_{\text{fail}} < 1$

initialize:

$\forall m:\, \nu^p_m(1) = \sum_{n=1}^N |a_{mn}|^2\nu^x_n(1),\ \ \widehat{p}_m(1) = \sum_{n=1}^N a_{mn}\widehat{x}_n(1)$ (I2)

$J(1)=\infty,\ \beta(1)=1,\ t=1$ (I3)

while $t \leq T_{\max}$,

$\forall m:\, \nu^z_m(t) = \nu^p_m(t)\, g'_{\mathsf{z}_m}(\widehat{p}_m(t), \nu^p_m(t))$ (R1)

$\forall m:\, \widehat{z}_m(t) = g_{\mathsf{z}_m}(\widehat{p}_m(t), \nu^p_m(t))$ (R2)

$\forall m:\, \nu^s_m(t) = \beta(t)\Big(1-\frac{\nu^z_m(t)}{\nu^p_m(t)}\Big)\frac{1}{\nu^p_m(t)} + \big(1-\beta(t)\big)\nu^s_m(t-1)$ (R3)

$\forall m:\, \widehat{s}_m(t) = \beta(t)\frac{\widehat{z}_m(t)-\widehat{p}_m(t)}{\nu^p_m(t)} + \big(1-\beta(t)\big)\widehat{s}_m(t-1)$ (R4)

$\forall n:\, \widetilde{x}_n(t) = \beta(t)\widehat{x}_n(t) + \big(1-\beta(t)\big)\widetilde{x}_n(t-1)$ (R5)

$\forall n:\, \nu^r_n(t) = \beta(t)\frac{1}{\sum_{m=1}^M |a_{mn}|^2\nu^s_m(t)} + \big(1-\beta(t)\big)\nu^r_n(t-1)$ (R6)

$\forall n:\, \widehat{r}_n(t) = \widetilde{x}_n(t) + \nu^r_n(t)\sum_{m=1}^M a^{\mathsf{H}}_{mn}\widehat{s}_m(t)$ (R7)

$\forall n:\, \nu^x_n(t+1) = \nu^r_n(t)\, g'_{\mathsf{x}_n}(\widehat{r}_n(t), \nu^r_n(t))$ (R8)

$\forall n:\, \widehat{x}_n(t+1) = g_{\mathsf{x}_n}(\widehat{r}_n(t), \nu^r_n(t))$ (R9)

$\forall m:\, \nu^p_m(t+1) = \beta(t)\sum_{n=1}^N |a_{mn}|^2\nu^x_n(t+1) + (1-\beta(t))\nu^p_m(t)$ (R10)

$\forall m:\, \widehat{p}_m(t+1) = \sum_{n=1}^N a_{mn}\widehat{x}_n(t+1) - \nu^p_m(t+1)\,\widehat{s}_m(t)$ (R11)

$J(t+1) =$ eqn (2) for MAP-GAMP or eqn (11) for MMSE-GAMP (R12)

if $J(t+1) \leq \max_{\tau=\max\{t-T_\beta,1\},\ldots,t} J(\tau)$ or $\beta(t)=\beta_{\min}$ (R13)

then if $\|\widehat{\boldsymbol{x}}(t)-\widehat{\boldsymbol{x}}(t+1)\|/\|\widehat{\boldsymbol{x}}(t+1)\| < \epsilon$, (R14)

then stop (R15)

else $\beta(t+1) = \min\{\beta_{\max}, G_{\text{pass}}\beta(t)\}$ (R16)

$t = t+1$ (R17)

else $\beta(t) = \max\{\beta_{\min}, G_{\text{fail}}\beta(t)\}$, (R18)

end

outputs: $\forall m,n:\, \widehat{r}_n(t), \nu^r_n(t), \widehat{p}_m(t+1), \nu^p_m(t+1), \widehat{x}_n(t+1), \nu^x_n(t+1)$

Table 1. The adaptively damped GAMP algorithm. In lines (R1) and (R8), g'_{z_m} and g'_{x_n} denote the derivatives of g_{z_m} and g_{x_n} w.r.t their first arguments.

allows divergence while too much damping unnecessarily slows convergence, we are motivated to develop an *adaptive* damping scheme that applies just the right amount of damping at each iteration.

Table 1 details the proposed *adaptively damped GAMP* (AD-GAMP) algorithm. Lines (R3)-(R6) and (R10) use an iteration-t-dependent damping parameter $\beta(t) \in (0,1]$ to slow the updates,[1] and lines (R12)-(R18) adapt the parameter $\beta(t)$. When $\beta(t)=1\ \forall t$, AD-GAMP reduces to the original GAMP from [1]. Due to lack of space, we refer readers to [1, 4] for further details on GAMP.

2.1. Damping Adaptation

The damping adaptation mechanism in AD-GAMP works as follows. Line (R12) computes the current cost $J(t+1)$, as described in the sequel. Line (R13) then checks evaluates whether the current iteration "passes" or "fails": it passes if the current cost is at least as good as the worst cost over the last $T_\beta \geq 0$ iterations or if $\beta(t)$ is already at its minimum allowed value $\beta_{\min}$, else it fails. If the iteration passes, (R14)-(R15) implement a stopping condition, (R16) increases $\beta(t)$ by a factor $G_{\text{pass}} \geq 1$ (up to the maximum value $\beta_{\max}$), and (R17) increments the counter t. If the iteration fails, (R18) decreases $\beta(t)$ by a factor $G_{\text{fail}} < 1$ (down to the minimum value $\beta_{\min}$) and the counter t is *not* advanced, causing AD-GAMP to re-try the tth iteration with the new value of $\beta(t)$.

In the MAP case, line (R12) simply computes the cost $J(t+1) = J_{\mathsf{MAP}}(\widehat{\boldsymbol{x}}(t+1))$ for J_{MAP} from (2). The MMSE case, which is more involved, will be described next.

[1] The GAMPmatlab implementation [9] allows one to disable damping in (R6) and/or (R10).

2.2. MMSE-GAMP Cost Evaluation

As proven in [4] and interpreted in the context of Bethe free entropy in [5], the fixed points of MMSE-GAMP are critical points of the optimization problem

$$(f_{\boldsymbol{x}}, f_{\boldsymbol{z}}) = \arg\min_{b_{\boldsymbol{x}}, b_{\boldsymbol{z}}} J_{\text{Bethe}}(b_{\boldsymbol{x}}, b_{\boldsymbol{z}}) \text{ s.t. } \mathrm{E}\{\boldsymbol{z}|b_{\boldsymbol{z}}\} = \boldsymbol{A}\,\mathrm{E}\{\boldsymbol{x}|b_{\boldsymbol{x}}\} \quad (3)$$

$$J_{\text{Bethe}}(b_{\boldsymbol{x}}, b_{\boldsymbol{z}}) \triangleq D\big(b_{\boldsymbol{x}}\|p_{\boldsymbol{x}}\big) + D\big(b_{\boldsymbol{z}}\|p_{\boldsymbol{y}|\boldsymbol{z}}Z^{-1}\big) + H\big(b_{\boldsymbol{z}}, \boldsymbol{\nu}^{\boldsymbol{p}}\big) \quad (4)$$

$$H\big(b_{\boldsymbol{z}}, \boldsymbol{\nu}^{\boldsymbol{p}}\big) \triangleq \frac{1}{2}\sum_{m=1}^M \left(\frac{\mathrm{var}\{\mathsf{z}_m|b_{\mathsf{z}_m}\}}{\nu^p_m} + \ln 2\pi\nu^p_m\right), \quad (5)$$

where $b_{\boldsymbol{x}}$ and $b_{\boldsymbol{z}}$ are separable pdfs, $Z^{-1} \triangleq \int p_{\boldsymbol{y}|\boldsymbol{z}}(\boldsymbol{y}|\boldsymbol{z})d\boldsymbol{z}$ is the scaling factor that renders $p_{\boldsymbol{y}|\boldsymbol{z}}(\boldsymbol{y}|\boldsymbol{z})Z^{-1}$ a valid pdf over $\boldsymbol{z} \in \mathbb{R}^M$, $D(\cdot\|\cdot)$ denotes Kullback-Leibler (KL) divergence, and $H(b_{\boldsymbol{z}})$ is an upper bound on the entropy of $b_{\boldsymbol{z}}$ that is tight when $b_{\boldsymbol{z}}$ is independent Gaussian with variances in $\boldsymbol{\nu}^{\boldsymbol{p}}$. In other words, the pdfs $f_{\boldsymbol{x}}(\boldsymbol{x};\widehat{\boldsymbol{r}},\boldsymbol{\nu}^{\boldsymbol{r}}) = \prod_{n=1}^N f_{\mathsf{x}_n}(x_n;\widehat{r}_n,\nu^r_n)$ and $f_{\boldsymbol{z}}(\boldsymbol{z};\widehat{\boldsymbol{p}},\boldsymbol{\nu}^{\boldsymbol{p}}) = \prod_{m=1}^M f_{\mathsf{z}_m}(z_m;\widehat{p}_m,\nu^p_m)$ given in lines (D1) and (D2) of Table 1 are critical points of (3) for fixed-point versions of $\widehat{\boldsymbol{r}},\boldsymbol{\nu}^{\boldsymbol{r}},\widehat{\boldsymbol{p}},\boldsymbol{\nu}^{\boldsymbol{p}}$.

Since $f_{\boldsymbol{x}}$ and $f_{\boldsymbol{z}}$ are functions of $\widehat{\boldsymbol{r}},\boldsymbol{\nu}^{\boldsymbol{r}},\widehat{\boldsymbol{p}},\boldsymbol{\nu}^{\boldsymbol{p}}$, the cost J_{Bethe} can be written in terms of these quantities as well. For this, we first note

$$D\big(f_{\mathsf{x}_n}\big\|p_{\mathsf{x}_n}\big) = \int f_{\mathsf{x}_n}(x;\widehat{r}_n,\nu^r_n)\ln\frac{p_{\mathsf{x}_n}(x)\mathcal{N}(x;\widehat{r}_n,\nu^r_n)}{p_{\mathsf{x}_n}(x)C_n(\widehat{r}_n,\nu^r_n)}dx \quad (6)$$

$$= -\ln C_n(\widehat{r}_n,\nu^r_n) - \frac{\ln 2\pi\nu^r_n}{2} - \int f_{\mathsf{x}_n}(x;\widehat{r}_n,\nu^r_n)\frac{|x-\widehat{r}_n|^2}{2\nu^r_n}dx \quad (7)$$

$$= -\ln C_n(\widehat{r}_n,\nu^r_n) - \frac{\ln 2\pi\nu^r_n}{2} - \frac{|\widehat{x}_n-\widehat{r}_n|^2+\nu^x_n}{2\nu^r_n}, \quad (8)$$

where $\widehat{x}_n$ and ν^x_n are the mean and variance of $f_{\mathsf{x}_n}(\cdot;\widehat{r}_n,\nu^r_n)$ from (R9) and (R8). Following a similar procedure,

$$D\big(f_{\mathsf{z}_m}\|p_{\mathsf{y}_m|\mathsf{z}_m}Z_m^{-1}\big) = -\ln\frac{B_m(\widehat{p}_m,\nu^p_m)}{Z_m} - \frac{\ln 2\pi\nu^p_m}{2} - \frac{|\widehat{z}_m-\widehat{p}_m|^2+\nu^z_m}{2\nu^p_m}, \quad (9)$$

where $\widehat{z}_m$ and ν^z_m are the mean and variance of $f_{\mathsf{z}_m}(\cdot;\widehat{p}_m,\nu^p_m)$ from (R2) and (R1). Then, since $D(f_{\boldsymbol{x}}\|p_{\boldsymbol{x}}) = \sum_{n=1}^N D\big(f_{\mathsf{x}_n}\|p_{\mathsf{x}_n}\big)$ and $D(f_{\boldsymbol{z}}\|p_{\boldsymbol{y}|\boldsymbol{z}}Z^{-1}) = \sum_{m=1}^M D\big(f_{\mathsf{z}_m}\|p_{\mathsf{y}_m|\mathsf{z}_m}Z_m^{-1}\big)$, (4) and (5) imply

$$J_{\text{Bethe}}(\widehat{\boldsymbol{r}},\boldsymbol{\nu}^{\boldsymbol{r}},\widehat{\boldsymbol{p}},\boldsymbol{\nu}^{\boldsymbol{p}}) = -\sum_{m=1}^M\left(\ln B_m(\widehat{p}_m,\nu^p_m) + \frac{|\widehat{z}_m-\widehat{p}_m|^2}{2\nu^p_m}\right) - \sum_{n=1}^N\left(\ln C_n(\widehat{r}_n,\nu^r_n) + \frac{\ln\nu^r_n}{2} + \frac{\nu^x_n+|\widehat{x}_n-\widehat{r}_n|^2}{2\nu^r_n}\right) + \mathsf{const}, \quad (10)$$

where we have written $J_{\text{Bethe}}(f_{\boldsymbol{x}}, f_{\boldsymbol{z}})$ as "$J_{\text{Bethe}}(\widehat{\boldsymbol{r}},\boldsymbol{\nu}^{\boldsymbol{r}},\widehat{\boldsymbol{p}},\boldsymbol{\nu}^{\boldsymbol{p}})$" to make the $(\widehat{\boldsymbol{r}},\boldsymbol{\nu}^{\boldsymbol{r}},\widehat{\boldsymbol{p}},\boldsymbol{\nu}^{\boldsymbol{p}})$-dependence clear, and where const collects terms invariant to $(\widehat{\boldsymbol{r}},\boldsymbol{\nu}^{\boldsymbol{r}},\widehat{\boldsymbol{p}},\boldsymbol{\nu}^{\boldsymbol{p}})$.

inputs:
$g_{\mathsf{z}_m}, [\boldsymbol{A}\widehat{\boldsymbol{x}}]_m, \nu^p_m, \widetilde{p}_m(1), I_{\max} \geq 1, \epsilon_{\text{inv}} \geq 0, \alpha \in (0,1], \phi \geq 0$
for $i = 1 : I_{\max}$,
$e_m(i) = [\boldsymbol{A}\widehat{\boldsymbol{x}}]_m - g_{\mathsf{z}_m}\big(\widetilde{p}_m(i), \nu^p_m\big)$ (F1)
if $\big|e_m(i)/g_{\mathsf{z}_m}\big(\widetilde{p}_m(i), \nu^p_m\big)\big| < \epsilon_{\text{inv}}$, stop (F2)
$\nabla_m(i) = g'_{\mathsf{z}_m}\big(\widetilde{p}_m(i), \nu^p_m\big)$ (F3)
$\widetilde{p}_m(i+1) = \widetilde{p}_m(i) + \alpha \frac{e_m(i)\nabla_m(i)}{\nabla^2_m(i)+\phi}$ (F4)
end
outputs: $\widetilde{p}_m(i)$

Table 2. A regularized Newton's method to find the value of $\widetilde{p}_m$ that solves $[\boldsymbol{A}\widehat{\boldsymbol{x}}]_m = g_{\mathsf{z}_m}(\widetilde{p}_m, \nu^p_m)$ for a given $[\boldsymbol{A}\widehat{\boldsymbol{x}}]_m$ and ν^p_m.

Note that the iteration-t MMSE-GAMP cost is not obtained simply by plugging $(\widehat{\boldsymbol{r}}(t), \boldsymbol{\nu}^{\boldsymbol{r}}(t), \widehat{\boldsymbol{p}}(t+1), \boldsymbol{\nu}^{\boldsymbol{p}}(t+1))$ into (10), because the latter quantities do not necessarily yield $(f_{\boldsymbol{x}}, f_{\boldsymbol{z}})$ satisfying the moment-matching constraint $\mathrm{E}\{\boldsymbol{z}|f_{\boldsymbol{z}}\} = \boldsymbol{A}\,\mathrm{E}\{\boldsymbol{x}|f_{\boldsymbol{x}}\}$ from (3). Thus, it was suggested in [5] to compute the cost as

$$J_{\mathsf{MSE}}(\widehat{\boldsymbol{r}}(t), \boldsymbol{\nu}^{\boldsymbol{r}}(t)) = J_{\text{Bethe}}(\widehat{\boldsymbol{r}}(t), \boldsymbol{\nu}^{\boldsymbol{r}}(t), \widetilde{\boldsymbol{p}}, \boldsymbol{\nu}^{\boldsymbol{p}}(t+1)), \tag{11}$$

for $\widetilde{\boldsymbol{p}}$ chosen to match the moment-matching constraint, i.e., for

$$[\boldsymbol{A}\widehat{\boldsymbol{x}}(t+1)]_m = g_{\mathsf{z}_m}\big(\widetilde{p}_m, \nu^p_m(t+1)\big) \text{ for } m = 1, \ldots, M \tag{12}$$

where $\widehat{x}_n(t+1) = g_{\mathsf{x}_n}\big(\widehat{r}_n(t), \mu^r_n(t)\big)$ for $n = 1, \ldots, N$ from (R9). Note that, since $\boldsymbol{\nu}^{\boldsymbol{p}}(t+1)$ can be computed from $(\widehat{\boldsymbol{r}}(t), \boldsymbol{\nu}^{\boldsymbol{r}}(t))$ via (R8) and (R10), the left side of (11) uses only $(\widehat{\boldsymbol{r}}(t), \boldsymbol{\nu}^{\boldsymbol{r}}(t))$.

In the case of an additive white Gaussian noise (AWGN), i.e., $p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z_m) = \mathcal{N}(z_m; y_m, \nu^w)$ with $\nu^w > 0$, the function $g_{\mathsf{z}_m}(\widetilde{p}_m, \nu^p_m)$ is linear in $\widetilde{p}_m$. In this case, [5] showed that (12) can be solved in closed-form, yielding the solution

$$\widetilde{p}_m = \big((\nu^p_m(t+1) + \nu^w)[\boldsymbol{A}\widehat{\boldsymbol{x}}(t+1)]_m - \nu^p_m(t+1)y_m\big)/\nu^w. \tag{13}$$

For general $p_{\mathsf{y}_m|\mathsf{z}_m}$, however, the function $g_{\mathsf{z}_m}(\widetilde{p}_m, \nu^p_m)$ is nonlinear in $\widetilde{p}_m$ and difficult to invert in closed-form. Thus, we propose to solve (12) numerically using the regularized Newton's method detailed in Table 2. There, $\alpha \in (0,1]$ is a stepsize, $\phi \geq 0$ is a regularization parameter that keeps the update's denominator positive, and $I_{\max}$ is a maximum number of iterations, all of which should be tuned in accordance with $p_{\mathsf{y}_m|\mathsf{z}_m}$. Meanwhile, $\widetilde{p}_m(1)$ is an initialization that can be set at $\widehat{p}_m(t+1)$ or $[\boldsymbol{A}\widehat{\boldsymbol{x}}(t+1)]_m$ and ϵ_{inv} is a stopping tolerance. Note that the functions g_{z_m} and g'_{z_m} employed in Table 2 are readily available from Table 1.

2.3. Mean Removal

To mitigate the difficulties caused by $\boldsymbol{A}$ with non-zero mean entries, we propose to rewrite the linear system "$\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}$" in (1) as

$$\underbrace{\begin{bmatrix} \boldsymbol{z} \\ z_{M+1} \\ z_{M+2} \end{bmatrix}}_{\triangleq\, \overline{\boldsymbol{z}}} = \underbrace{\begin{bmatrix} \widetilde{\boldsymbol{A}} & b_{12}\boldsymbol{\gamma} & b_{13}\mathbf{1}_M \\ b_{21}\mathbf{1}^{\mathsf{H}}_N & -b_{21}b_{12} & 0 \\ b_{31}\boldsymbol{c}^{\mathsf{H}} & 0 & -b_{31}b_{13} \end{bmatrix}}_{\triangleq\, \overline{\boldsymbol{A}}} \underbrace{\begin{bmatrix} \boldsymbol{x} \\ x_{N+1} \\ x_{N+2} \end{bmatrix}}_{\triangleq\, \overline{\boldsymbol{x}}} \tag{14}$$

where $(\cdot)^{\mathsf{H}}$ is conjugate transpose, $\mathbf{1}_P \triangleq [1, \ldots, 1]^{\mathsf{H}} \in \mathbb{R}^P$, and

$$\mu \triangleq \tfrac{1}{MN}\mathbf{1}^{\mathsf{H}}_M \boldsymbol{A}\mathbf{1}_N \tag{15}$$

$$\boldsymbol{\gamma} \triangleq \tfrac{1}{N}\boldsymbol{A}\mathbf{1}_N \tag{16}$$

$$\boldsymbol{c}^{\mathsf{H}} \triangleq \tfrac{1}{M}\mathbf{1}^{\mathsf{H}}_M\big(\boldsymbol{A} - \mu\mathbf{1}_M\mathbf{1}^{\mathsf{H}}_N\big) \tag{17}$$

$$\widetilde{\boldsymbol{A}} \triangleq \boldsymbol{A} - \boldsymbol{\gamma}\mathbf{1}^{\mathsf{H}}_N - \mathbf{1}_M\boldsymbol{c}^{\mathsf{H}}. \tag{18}$$

The advantage of (14) is that the rows and columns of $\overline{\boldsymbol{A}}$ are approximately zero-mean. This can be seen by first verifying, via the definitions above, that $\boldsymbol{c}^{\mathsf{H}}\mathbf{1}_N = 0$, $\widetilde{\boldsymbol{A}}\mathbf{1}_N = \mathbf{0}$, and $\mathbf{1}^{\mathsf{H}}_M\widetilde{\boldsymbol{A}} = \mathbf{0}^{\mathsf{H}}$, which implies that the elements in every row and column of $\widetilde{\boldsymbol{A}}$ are zero-mean. Thus, for large N and M, the elements in all but a vanishing fraction of the rows and columns in $\overline{\boldsymbol{A}}$ will also be zero-mean. The mean-square coefficient size in the last two rows and columns of $\overline{\boldsymbol{A}}$ can be made to match that in $\widetilde{\boldsymbol{A}}$ via choice of $b_{12}, b_{13}, b_{21}, b_{31}$.

To understand the construction of (14), note that (18) implies

$$\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x} = \widetilde{\boldsymbol{A}}\boldsymbol{x} + b_{12}\boldsymbol{\gamma}\underbrace{\mathbf{1}^{\mathsf{H}}_N\boldsymbol{x}/b_{12}}_{\triangleq\, x_{N+1}} + b_{13}\mathbf{1}_M\underbrace{\boldsymbol{c}^{\mathsf{H}}\boldsymbol{x}/b_{13}}_{\triangleq\, x_{N+2}}, \tag{19}$$

which explains the first M rows of (14). To satisfy the definitions in (19), we then require that $z_{M+1} = 0$ and $z_{M+2} = 0$ in (14), which can be ensured through the Dirac-delta likelihood

$$p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z_m) \triangleq \delta(z_m) \;\text{ for }\; m \in \{M+1, M+2\}. \tag{20}$$

Meanwhile, we make no assumption about the newly added elements x_{N+1} and x_{N+2}, and thus adopt the improper uniform prior

$$p_{\mathsf{x}_n}(x_n) \propto 1 \;\text{ for }\; n \in \{N+1, N+2\}. \tag{21}$$

In summary, the mean-removal approach suggested here runs GAMP or AD-GAMP (as in Table 1) with $\overline{\boldsymbol{A}}$ in place of $\boldsymbol{A}$ and with the likelihoods and priors augmented by (20) and (21). It is important to note that, if multiplication by $\boldsymbol{A}$ and $\boldsymbol{A}^{\mathsf{H}}$ can be implemented using a fast transform (e.g., FFT), then multiplication by $\overline{\boldsymbol{A}}$ and $\overline{\boldsymbol{A}}^{\mathsf{H}}$ can too; for details, see the GAMPmatlab implementation [9].

3. NUMERICAL RESULTS

We numerically studied the recovery $\mathsf{NMSE} \triangleq \|\widehat{\boldsymbol{x}} - \boldsymbol{x}\|^2/\|\boldsymbol{x}\|^2$ of SwAMP [8] and the MMSE version of the original GAMP from [1] relative to the proposed mean-removed (M-GAMP) and adaptively damped (AD-GAMP) modifications, as well as their combination (MAD-GAMP). In all experiments, the signal $\boldsymbol{x}$ was drawn Bernoulli-Gaussian (BG) with sparsity rate τ and length $N = 1000$, and performance was averaged over 100 realizations. Average NMSE was clipped to 0 dB for plotting purposes. The matrix $\boldsymbol{A}$ was drawn in one of four ways:

(a) **Non-zero mean**: i.i.d $\mathsf{a}_{mn} \sim \mathcal{N}(\mu, \frac{1}{N})$ for a specified $\mu \neq 0$.

(b) **Low-rank product**: $\mathbf{A} = \frac{1}{N}\mathbf{U}\mathbf{V}$ with $\mathbf{U} \in \mathbb{R}^{M\times R}$, $\mathbf{V} \in \mathbb{R}^{R\times N}$, and i.i.d $\mathsf{u}_{mr}, \mathsf{v}_{rn} \sim \mathcal{N}(0,1)$, for a specified R. Note $\mathbf{A}$ is rank deficient when $R < \min\{M, N\}$.

(c) **Column-correlated**: the rows of $\mathbf{A}$ are independent zero-mean stationary Gauss-Markov processes with a specified correlation coefficient $\rho = \mathrm{E}\{\mathsf{a}_{mn}\mathsf{a}^*_{m,n+1}\}/\,\mathrm{E}\{|\mathsf{a}_{mn}|^2\}$.

(d) **Ill-conditioned**: $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathsf{H}}$ where $\mathbf{U}$ and $\mathbf{V}^{\mathsf{H}}$ are the left and right singular vector matrices of an i.i.d $\mathcal{N}(0,1)$ matrix and $\boldsymbol{\Sigma}$ is a singular value matrix such that $[\boldsymbol{\Sigma}]_{i,i}/[\boldsymbol{\Sigma}]_{i+1,i+1} = (\kappa)^{1/\min\{M,N\}}$ for $i = 1, \ldots, \min\{M,N\}-1$, with a specified condition number $\kappa > 1$.

For all algorithms, we used $T_{\max} = 1000$ and $\epsilon = 10^{-5}$. Unless otherwise noted, for adaptive damping, we used $T_\beta = 0$, $G_{\text{pass}} = 1.1$, $G_{\text{fail}} = 0.5$, $\beta_{\max} = 1$, and $\beta_{\min} = 0.01$. For SwAMP, we used the authors' publicly available code [12].

First we experiment with compressive sensing (CS) in AWGN at $\mathsf{SNR} \triangleq \mathrm{E}\{\|\boldsymbol{z}\|^2\}/\,\mathrm{E}\{\|\boldsymbol{y} - \boldsymbol{z}\|^2\} = 60$ dB. For this, we used

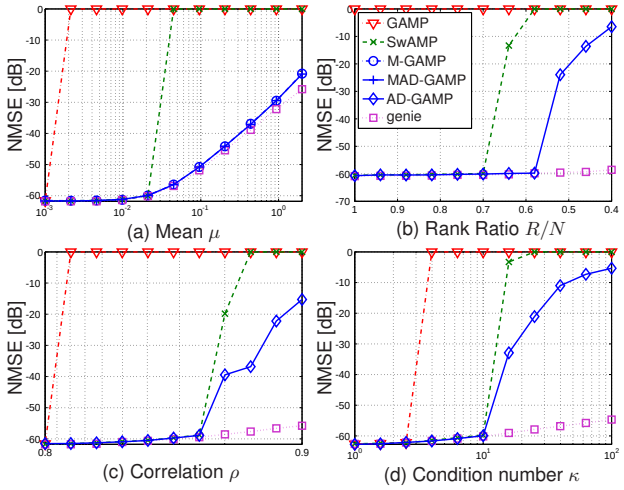

Fig. 1. AWGN compressive sensing under (a) non-zero-mean, (b) low-rank product, (c) column-correlated, and (d) ill-conditioned $\boldsymbol{A}$.

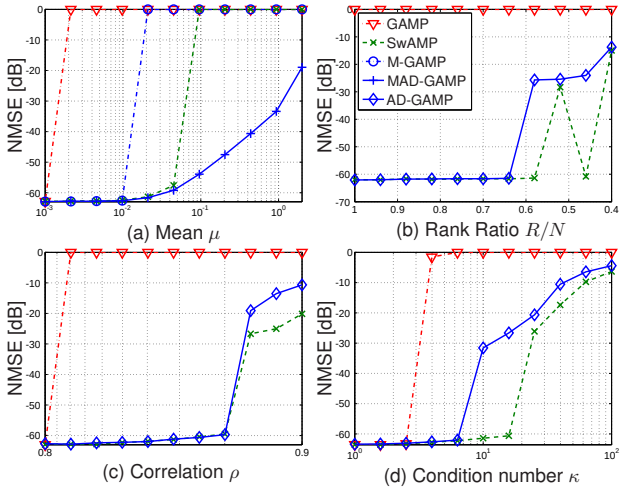

Fig. 2. "Robust" compressive sensing under (a) non-zero-mean, (b) low-rank product, (c) column-correlated, and (d) ill-conditioned $\boldsymbol{A}$.

$M = 500 = N/2$ measurements and sparsity rate $\tau = 0.2$. As a reference, we compute a lower-bound on the achievable NMSE using a genie who knows the support of $\boldsymbol{x}$. For non-zero-mean matrices, Fig. 1(a) shows that the proposed M-GAMP and MAD-GAMP provided near-genie performance for all tested means μ. In contrast, GAMP only worked with zero-mean $\boldsymbol{A}$ and SwAMP with small-mean $\boldsymbol{A}$. For low-rank product, correlated, and ill-conditioned matrices, Figs. 1(b)-(d) show that AD-GAMP is slightly more robust than SwAMP and significantly more robust than GAMP.

Next, we tried "robust" CS by repeating the previous experiment with sparsity rate $\tau = 0.15$ and with 10% of the observations (selected uniformly at random) replaced by "outliers" corrupted by AWGN at SNR $= 0$ dB. For (M)AD-GAMP, we set $\beta_{\max} = 0.1$ and $T_{\max} = 2000$. With non-zero-mean $\boldsymbol{A}$, Fig. 2(a) shows increasing performance as we move from GAMP to M-GAMP to SwAMP to MAD-GAMP. For low-rank product, correlated, and ill-conditioned matrices, Fig. 2(b)-(d) show that SwAMP was slightly more robust than AD-GAMP, and both where much more robust than GAMP.

Finally, we experimented with noiseless 1-bit CS [13], where $\boldsymbol{y} = \mathrm{sgn}(\boldsymbol{A}\boldsymbol{x})$, using $M = 3000$ measurements and sparsity ratio $\tau = 0.125$. In each realization, the empirical mean was subtracted

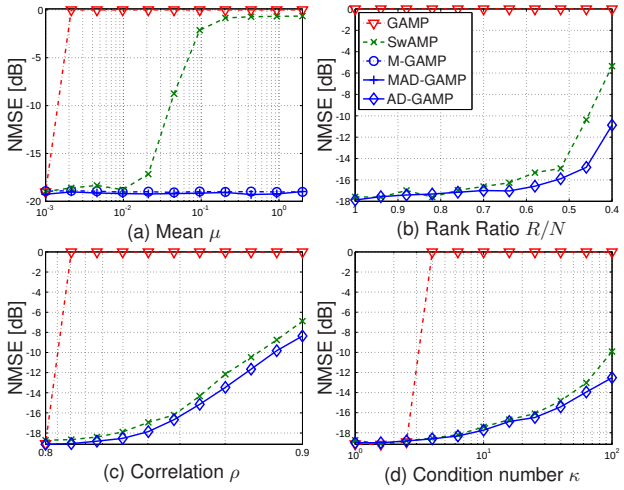

Fig. 3. 1-bit compressive sensing under (a) non-zero-mean, (b) low-rank product, (c) column-correlated, and (d) ill-conditioned $\boldsymbol{A}$.

		$\mu = 0.021$		$R/N = 0.64$		$\rho = 0.8$		$\log_{10}\kappa = 1$	
		MAD-GAMP	SwAMP	AD-GAMP	SwAMP	AD-GAMP	SwAMP	AD-GAMP	SwAMP
seconds	AWGN	**1.06**	1.90	**0.88**	2.74	**1.36**	3.84	**0.81**	1.49
	1-bit	**53.34**	83.21	**49.22**	137.46	**42.32**	149.40	**50.25**	117.62
	Robust	**3.47**	8.81	**2.66**	11.13	**3.33**	15.70	**2.38**	12.22
# iters	AWGN	42.9	**39.2**	130.0	**109.5**	221.9	**153.2**	121.4	**58.8**
	1-bit	947.8	**97.4**	942.7	**160.8**	866.2	**175.8**	927.3	**136.3**
	Robust	187.3	**42.2**	208.7	**56.1**	269.1	**79.2**	187.7	**61.7**

Table 3. Average runtime (in seconds) and # iterations of MAD-GAMP and SwAMP for various problem types and matrix types.

from the non-zero entries of $\boldsymbol{x}$ to prevent $y_m = 1\ \forall m$. For (M)AD-GAMP, we used $\beta_{\max} = 0.5$. For SwAMP, we increased the stopping tolerance to $\epsilon = 5 \times 10^{-5}$, as it significantly improved runtime without degrading accuracy. For non-zero-mean $\boldsymbol{A}$, Fig. 3(a) shows that M-GAMP and MAD-GAMP were more robust than SwAMP, which was in turn much more robust than GAMP. For low-rank product, correlated, and ill-conditioned matrices, Figs. 3(b)-(d) show that MAD-GAMP and SwAMP gave similarly robust performance, while the original GAMP was very fragile.

Finally, we compare the convergence speed of MAD-GAMP to SwAMP. For each problem, we chose a setting that allowed MAD-GAMP and SwAMP to converge for each matrix type. Table 3 shows that, on the whole, MAD-GAMP ran several times faster than SwAMP but used more iterations. Thus, it may be possible to reduce SwAMP's runtime to below that of MAD-GAMP using a more efficient (e.g., BLAS-based) implementation, at least for explicit $\boldsymbol{A}$. When $\boldsymbol{A}$ has a fast $O(N \log N)$ implementation (e.g., FFT), only (M)AD-GAMP will be able to exploit the reduced complexity.

4. CONCLUSIONS

We proposed adaptive damping and mean-removal modifications of GAMP that help prevent divergence in the case of "difficult" $\boldsymbol{A}$ matrices. We then numerically demonstrated that the resulting modifications significantly increase GAMP's robustness to non-zero-mean, low-rank product, column-correlated, and ill-conditioned $\boldsymbol{A}$ matrices. Moreover, they provide robustness similar to the recently proposed SwAMP algorithm, whilerunning faster than the current SwAMP implementation. For future work, we note that the sequential update of SwAMP could in principle be combined with the proposed mean-removal and/or adaptive damping to perhaps achieve a level robustness greater than either SwAMP or (M)AD-GAMP.

5. REFERENCES

[1] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *Proc. IEEE Int. Symp. Inform. Thy.*, Aug. 2011, pp. 2168–2172, (full version at *arXiv:1010.5141*).

[2] M. Bayati, M. Lelarge, and A. Montanari, "Universality in polytope phase transitions and iterative algorithms," in *Proc. IEEE Int. Symp. Inform. Thy.*, Boston, MA, June 2012, pp. 1643–1647, (full paper at *arXiv:1207.7321*).

[3] Adel Javanmard and Andrea Montanari, "State evolution for general approximate message passing algorithms, with applications to spatial coupling," *Inform. Inference*, vol. 2, no. 2, pp. 115–144, 2013.

[4] S. Rangan, P. Schniter, E. Riegler, A. Fletcher, and V. Cevher, "Fixed points of generalized approximate message passing with arbitrary matrices," in *Proc. IEEE Int. Symp. Inform. Thy.*, July 2013, pp. 664–668, (full version at *arXiv:1301.6295*).

[5] F. Krzakala, A. Manoel, E. W. Tramel, and L. Zdeborová, "Variational free energies for compressed sensing," in *Proc. IEEE Int. Symp. Inform. Thy.*, July 2014, pp. 1499–1503, (see also *arXiv:1402.1384*).

[6] F. Caltagirone, F. Krzakala, and L. Zdeborová, "On convergence of approximate message passing," in *Proc. IEEE Int. Symp. Inform. Thy.*, July 2014, pp. 1812–1816, (see also *arXiv:1401.6384*).

[7] S. Rangan, P. Schniter, and A. Fletcher, "On the convergence of generalized approximate message passing with arbitrary matrices," in *Proc. IEEE Int. Symp. Inform. Thy.*, July 2014, pp. 236–240, (full version at *arXiv:1402.3210*).

[8] Andre Manoel, Florent Krzakala, Eric W. Tramel, and Lenka Zdeborová, "Sparse estimation with the swept approximated message-passing algorithm," *arXiv:1406.4311*, June 2014.

[9] S. Rangan, P. Schniter, J. T. Parker, J. Ziniel, J. Vila, M. Borgerding, and et al., "GAMPmatlab," `https://sourceforge.net/projects/gampmatlab/`.

[10] T. Heskes, "Stable fixed points of loopy belief propagation are minima of the Bethe free energy," in *Proc. Neural Inform. Process. Syst. Conf.*, Vancouver, B.C., Dec. 2002, pp. 343–350.

[11] P. Schniter and S. Rangan, "Compressive phase retrieval via generalized approximate message passing," in *Proc. Allerton Conf. Commun. Control Comput.*, Monticello, IL, Oct. 2012, pp. 815–822, (full version at *arXiv:1405.5618*).

[12] Andre Manoel, Florent Krzakala, Eric W. Tramel, and Lenka Zdeborová, "SwAMP demo user's manual," `https://github.com/eric-tramel/SwAMP-Demo`.

[13] U. S. Kamilov, V. K. Goyal, and S. Rangan, "Message-passing de-quantization with applications to compressed sensing," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6270–6281, Dec. 2012.