VAMP with Individual Variances and Sequential Processing for Compressed Sensing

Carmen Sippel, Robert F.H. Fischer
Institute of Communications Engineering, Ulm University, Germany
{carmen.sippel, robert.fischer}@uni-ulm.de

***Abstract*—The compressed sensing (CS) problem is considered, where a sparse signal with known structure has to be recovered from an under-determined system of linear equations. Iterative algorithms in CS alternate between two subproblems, which are given by the structure of the signal and the system of equations, respectively. Hereby, the components of the signal vector are usually updated in parallel. Alternatively, a sequential processing of the components of the signal vector is possible, where the order of processing can be varied and individual variances are utilized. A suitable scheduling can speed up the convergence and thereby reduce the complexity, which is necessary to achieve a certain performance level. In this paper, we introduce a sequential algorithm for CS, discuss a variety of schedules and show by numerical simulations that the number of signal components to be processed in order to attain a specific performance level can be significantly reduced with suitable schedules.**

***Index Terms*—Compressed sensing, sequential processing, VAMP**

I. Introduction

Signal reconstruction in compressed sensing (CS) [3], [5] is the task of finding a suitable solution for a given observation under two constraints. The first constraint is established by an under-determined system of equations, defined by the channel observations and the sensing matrix. The second constraint is prior knowledge of the signal, in particular its probability density function (pdf). A standard approach to solve this problem is to iteratively consider the problem under one of the constraints. *Vector approximate message passing (VAMP)* [15], derived from the expectation-consistent (EC) approximate inference framework [14], is an example of such an iterative algorithm.

VAMP uses an average variance as indicator for the reliability of the estimated signal. This can be improved by using individual variances for each element of the estimate, as has been shown in [7]. However, this results in higher complexity, since the necessary matrix inversion cannot be avoided using a singular value decomposition (SVD) of the sensing matrix, as is possible for the average variances case [15, Alg. 2]. Nevertheless, the usage of individual variances allows for processing the elements of the signal sequentially, thereby enabling to profit from insights at earlier stages of processing and an optimized scheduling.

Sequential processing has been addressed in code-division multiple-access for successive interference cancellation, see [6] for an overview, in decoding of low-density parity-check codes [4], and in sparse Bayesian learning [19].

In the CS literature, such approaches are used for sequentially processing the observations [12], sequentially processing dynamically changing signals [10], or sequential design of the sensing matrix [9]. In contrast, we pursue a sequential, element-wise processing of a given fixed signal.

Other applications to CS are, e.g., given in [16], where the focus lies on the design of the sensing matrix, in [1] for a Bernoulli-Gaussian prior in the noiseless case, and in [13], where approximate message passing (AMP) [11] is adapted to sequential processing. In contrast to the latter two, we consider a discrete prior and the update from [7], which cannot be derived from belief propagation or similar frameworks. Unlike AMP, we employ the minimum mean-squared error (MMSE) estimator for the channel-constrained part of the problem.

In this paper, we assess a sequential processing for the algorithm given in [7] by utilizing a rank-one update for the solution of the under-determined system of equations. The resulting algorithm is in spirit of the one stated in [14, App. D], however, i) we apply it to the CS setting, ii) adapt the update according to [7], and iii) optimize the processing order. By numerical simulations, we show that with a suitable schedule, the increase in complexity due to the usage of individual variances can be avoided by evaluating subsets of the signal vector and an overall faster convergence.

The paper is organized as follows. In Sec. II we introduce the system model for compressed sensing and briefly review VAMP. Then, we present the sequential approach and discuss suitable schedules in Sec. III. Results from numerical simulations are presented and discussed in Sec. IV. Finally, we conclude our work in Sec. V.

II. Problem Formulation

A. System Model for Compressed Sensing

The noisy measurements in CS are given by[1]

$$\mathsf{y} = \boldsymbol{A}\mathsf{x} + \mathsf{n} \in \mathbb{R}^M \ , \tag{1}$$

[1]We denote scalars by small letters, e.g., x, vectors by bold ones, e.g., $\boldsymbol{x}$, matrices by upper case bold, e.g., $\boldsymbol{X}$, and random variables in sans-serif font, i.e., x, x, and X, respectively. $\boldsymbol{I}_m$: $m \times m$ identity matrix, $\mathbf{0}$: all-zero vector, $\mathbf{diag}(\cdot)$: diagonal matrix with given entries, $f_{\mathsf{x}}(x)$: pdf of x, $\Pr\{\cdot\}$: probability, $\mathrm{E}\{\cdot\}$: expectation, $\pi(\cdot)$: random permutation, $[\cdot]_j$: jth entry, $||\cdot||_{\mathrm{F}}$: Frobenius norm.

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) — FI 982/16-1.

where the sensing matrix $\boldsymbol{A} \in \mathbb{R}^{M\times N}$, $M \ll N$, is known and the noise $\mathsf{n} \sim \mathcal{N}(\mathbf{0}, \sigma_\mathsf{n}^2 \boldsymbol{I}_M)$ is i.i.d. Gaussian. The elements x_j of $\mathsf{x} = [\mathsf{x}_1, \ldots, \mathsf{x}_N]^\top$ are assumed to be i.i.d. with marginal pdf $\mathsf{f}_\mathsf{x}(x_j)$, i.e.,

$$\mathsf{f}_\mathsf{x}(\boldsymbol{x}) = \prod\nolimits_{j=1}^N \mathsf{f}_\mathsf{x}(x_j) \; . \tag{2}$$

The pdf of x is hence separable. In CS, the signal x is usually sparse, i.e., has few non-zero entries, which is reflected by the marginal pdf. The task of recovering the signal x is given by the estimation problem

$$\mathrm{E}_\mathsf{x}\{\mathsf{x} \mid \boldsymbol{y},\, \boldsymbol{A},\, \sigma_\mathsf{n}^2\} = \frac{1}{\mathsf{f}_\mathsf{y}(\boldsymbol{y})} \int \boldsymbol{x}\, \mathsf{f}_\mathsf{x}(\boldsymbol{x}) \mathsf{f}_\mathsf{y}(\boldsymbol{y} \mid \boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \; . \tag{3}$$

For a high-dimensional signal x this computation becomes infeasible. Therefore, iterative schemes have been proposed that split the problem into two subproblems and alternate between these.

B. Approach and VAMP

The subtasks (denoted with indices "c" for *channel constraint* and "s" for *signal constraint*) can be solved by a Gaussian assumption on the pdfs $\mathsf{f}_\mathsf{x}(\boldsymbol{x})$ and $\mathsf{f}_\mathsf{y}(\boldsymbol{y} \mid \boldsymbol{x})$ in (3), respectively. The Gaussian assumption makes the computation feasible. The preceding iteration delivers mean $\tilde{\boldsymbol{x}}_\bullet$ and variance $\tilde{\sigma}_\bullet^2$ ($\bullet \in \{\mathrm{c}, \mathrm{s}\}$). Under a minimum mean-squared error (MMSE) criterion, the optimal solution to either of the subtasks is given by the conditional mean

$$\boldsymbol{m}_\bullet = \mathrm{E}_{\mathsf{x},\bullet}\{\mathsf{x} \mid \tilde{\boldsymbol{x}}_\bullet,\, \tilde{\sigma}_\bullet^2\} \; , \quad \bullet \in \{\mathrm{c},\, \mathrm{s}\} \; . \tag{4}$$

Since for the channel constraint the argument of the integral becomes Gaussian, the corresponding solution is a joint linear estimator and given by

$$\boldsymbol{m}_\mathrm{c} = \tilde{\boldsymbol{x}}_\mathrm{c} + \left(\boldsymbol{A}^\top \boldsymbol{A} + \sigma_\mathsf{n}^2 \tilde{\boldsymbol{\Phi}}_\mathrm{c}^{-1}\right)^{-1} \boldsymbol{A}^\top (\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_\mathrm{c}) \; , \tag{5}$$

with $\tilde{\boldsymbol{\Phi}}_\mathrm{c} = \tilde{\sigma}_\mathrm{c}^2 \boldsymbol{I}_N$.

Due to the separability of x, the estimation for the signal constraint can be calculated individually for each variable x_j; giving rise to individual, non-linear estimators (NLEs)

$$m_{\mathrm{s},j} = \frac{1}{\sqrt{2\pi\tilde{\sigma}_\mathrm{s}^2}} \int x_j \mathsf{f}_\mathsf{x}(x_j) \exp\!\left(\frac{(x_j - \tilde{x}_{\mathrm{s},j})^2}{2\tilde{\sigma}_\mathrm{s}^2}\right) \mathrm{d}x_j \; . \tag{6}$$

Because of the Gaussian assumptions, mean and variance are necessary for the computations. For the given problems, the average variances can be computed by

$$\sigma_\mathrm{c}^2 = \frac{1}{N} \mathrm{trace}\!\left(\sigma_\mathsf{n}^2 \left(\boldsymbol{A}^\top \boldsymbol{A} + \sigma_\mathsf{n}^2 \tilde{\boldsymbol{\Phi}}_\mathrm{c}^{-1}\right)^{-1}\right) , \tag{7}$$

$$\sigma_\mathrm{s}^2 = \frac{1}{N} \sum_{j=1}^N \mathrm{E}_{\mathsf{x},\mathrm{s}}\big\{(\mathsf{x}_j - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_\mathrm{s}^2\big\} \; . \tag{8}$$

Using the notion of exponential families [2], it has been derived in [14] that the connection (crossover) between the variables of the estimators needs to be given by

$$\frac{\tilde{x}_{\mathrm{c},j}}{\tilde{\sigma}_\mathrm{c}^2} = \frac{m_{\mathrm{s},j}}{\sigma_\mathrm{s}^2} - \frac{\tilde{x}_{\mathrm{s},j}}{\tilde{\sigma}_\mathrm{s}^2} \; , \quad \frac{1}{\tilde{\sigma}_\mathrm{c}^2} = \frac{1}{\sigma_\mathrm{s}^2} - \frac{1}{\tilde{\sigma}_\mathrm{s}^2} \; , \tag{9}$$

respectively

$$\frac{\tilde{x}_{\mathrm{s},j}}{\tilde{\sigma}_\mathrm{s}^2} = \frac{m_{\mathrm{c},j}}{\sigma_\mathrm{c}^2} - \frac{\tilde{x}_{\mathrm{c},j}}{\tilde{\sigma}_\mathrm{c}^2} \; , \quad \frac{1}{\tilde{\sigma}_\mathrm{s}^2} = \frac{1}{\sigma_\mathrm{c}^2} - \frac{1}{\tilde{\sigma}_\mathrm{c}^2} \; . \tag{10}$$

This also fits to the notion of transferring extrinsics [8] and bias compensation [7], [18].

The resulting algorithm is identical to *vector approximate message passing* (VAMP) [15], which can be derived from the EC framework [14].

C. VAMP with Individual Variances

The usage of individual variances gives the estimator more insight into the signal estimates at hand. Hence, the performance and convergence speed of respective algorithms may be better. A straightforward adaption to individual variances based on VAMP is obtained by replacing in (5), (6), (9), and (10) σ_c^2 by $\sigma_{\mathrm{c},j}^2 = [\boldsymbol{\Phi}_\mathrm{c}]_{jj}$, σ_s^2 by $\sigma_{\mathrm{s},j}^2$, $\tilde{\sigma}_\mathrm{c}^2$ by $\tilde{\sigma}_{\mathrm{c},j}^2$, and $\tilde{\sigma}_\mathrm{s}^2$ by $\tilde{\sigma}_{\mathrm{s},j}^2$, where

$$\boldsymbol{\Phi}_\mathrm{c} = \sigma_\mathsf{n}^2 \left(\boldsymbol{A}^\top \boldsymbol{A} + \sigma_\mathsf{n}^2 \tilde{\boldsymbol{\Phi}}_\mathrm{c}^{-1}\right)^{-1} \; , \tag{11}$$

$$\begin{aligned} \sigma_{\mathrm{s},j}^2 &= \mathrm{E}_{\mathsf{x},\mathrm{s}}\big\{(\mathsf{x}_j - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\big\} \\ &= \frac{1}{\sqrt{2\pi\tilde{\sigma}_{\mathrm{s},j}^2}} \int (x - m_{\mathrm{s},j})^2 \mathsf{f}_\mathsf{x}(x) \exp\!\left(\frac{(x - \tilde{x}_{\mathrm{s},j})^2}{2\tilde{\sigma}_{\mathrm{s},j}^2}\right) \mathrm{d}x \; , \end{aligned} \tag{12}$$

and $\tilde{\boldsymbol{\Phi}}_\mathrm{c} = \mathbf{diag}(\tilde{\sigma}_{\mathrm{c},j}^2)$.

In [7] it has been shown that this algorithm can be improved by a signal processing view on the crossover after the NLE. The improved crossover is obtained by

$$\tilde{\sigma}_{\mathrm{c},j}^2 = \sigma_{\mathrm{s},j}^2 + \left(\tfrac{\sigma_j^2}{\tilde{\sigma}_{\mathrm{s},j}^2 - \sigma_j^2} (m_{\mathrm{s},j} - \tilde{x}_{\mathrm{s},j})\right)^2 \; , \tag{13}$$

$$\tilde{x}_{\mathrm{c},j} = \left(\tfrac{1}{\sigma_j^2} - \tfrac{1}{\tilde{\sigma}_{\mathrm{s},j}^2}\right)^{-1} \left(\tfrac{m_{\mathrm{s},j}}{\sigma_j^2} - \tfrac{\tilde{x}_{\mathrm{s},j}}{\tilde{\sigma}_{\mathrm{s},j}^2}\right) \; , \tag{14}$$

with $\sigma_j^2 = \mathrm{E}_{\tilde{\mathsf{x}}_\mathrm{s}}\{\sigma_{\mathrm{s},j}^2\}$ being the mean-squared error (MSE). We call this improved version of the algorithm VAMPire (VAMP with individual reliabilities enhanced).

However, when using individual variances, the matrix inversion in (11) is unavoidable, which increases the complexity significantly when comparing to the average variance case of VAMP. This problem can be overcome by a sequential update, which is introduced in the following.

III. Sequential Updates

VAMP with an average variance performs a *parallel update*, i.e., the non-linear estimation is carried out for all elements of vector $\boldsymbol{x}$ in parallel, followed by the computation of the linear estimate, again for the complete vector $\boldsymbol{x}$. The procedure is indicated in the factor graph of the problem in Fig. 1. The usage of individual variances offers the possibility to treat the signal components individually. This can be done by computation of the signal-constrained estimate $m_{\mathrm{s},j}$ for one element x_j and a subsequent update of the channel-constrained estimate $\boldsymbol{m}_\mathrm{c}$ according to the change at position j. Note that because of the coupling in the linear estimator (due to sensing

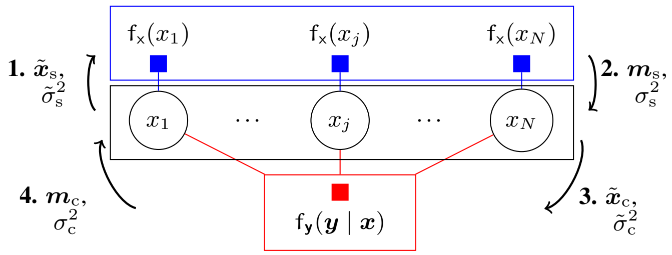

Fig. 1. Parallel update of VAMP in the factor graph for CS. The signal components x_j are treated jointly (vector $\boldsymbol{x}$), the computations of the estimates are given in (5)–(10).

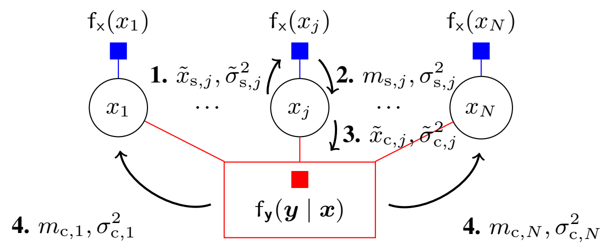

Fig. 2. Individual update (inner iteration) for the sequential algorithm. The computations are given by (6) and (10) (with individual variances), as well as (12)–(16).

matrix $\boldsymbol{A}$) the update has to be propagated to the entire signal vector. The update strategy is depicted in Fig. 2. We denote one round of the described update, where one vector element is processed, as an *inner iteration*. On the contrary, a loop over the vector elements is called an *outer iteration*.

A. Rank-one Update

The update for the linear estimator is a rank-one adjustment, obtained from the matrix inversion lemma. Basis of the computation is a change in the input parameters to the channel constraint, $\tilde{x}_{c,j}$ and $\tilde{\sigma}_{c,j}^2$. Let $\Delta\sigma^2 = \tilde{\sigma}_{c,j}^{2[k-1]} - \tilde{\sigma}_{c,j}^{2[k]}$ and $\Delta\tilde{x} = \tilde{\sigma}_{c,j}^{2[k-1]}\tilde{x}_{c,j}^{[k]} - \tilde{\sigma}_{c,j}^{2[k]}\tilde{x}_{c,j}^{[k-1]}$, where superscripts k and $k-1$ denote current and former value of the parameters, respectively. With these substitutions, a given covariance matrix $\boldsymbol{\Phi}_c = [\boldsymbol{\phi}_{c,1}, \dots, \boldsymbol{\phi}_{c,N}]$ and conditional mean $\boldsymbol{m}_c$ can be updated by (cf. [14])

$$\boldsymbol{m}_c^{\text{updated}} = \boldsymbol{m}_c + \frac{\Delta\tilde{x} - \Delta\sigma^2 m_{c,j}}{\tilde{\sigma}_{c,j}^{2[k]}\tilde{\sigma}_{c,j}^{2[k-1]} + \Delta\sigma^2[\boldsymbol{\Phi}_c]_{jj}}\boldsymbol{\phi}_{c,j} \,, \tag{15}$$

$$\boldsymbol{\Phi}_c^{\text{updated}} = \boldsymbol{\Phi}_c - \frac{\Delta\sigma^2}{\tilde{\sigma}_{c,j}^{2[k]}\tilde{\sigma}_{c,j}^{2[k-1]} + \Delta\sigma^2[\boldsymbol{\Phi}_c]_{jj}}\boldsymbol{\phi}_{c,j}\boldsymbol{\phi}_{c,j}^{\top} \,. \tag{16}$$

B. Schedules

The rank-one update for the linear estimator concludes one inner iteration. Subsequently, another element of the signal vector is considered. There are many ways to choose, which element is to be processed next.

First, we take a look at procedures, which loop through all positions in the outer iteration. The most natural ordering is to keep the order given in vector $\boldsymbol{x}$, i.e., process positions according to the sequence $s_{\mathcal{J}} = [1, 2, \dots, N]$. However, this does not make use of the individual variances, i.e., the measure for reliability of an individual estimate.

An alternative strategy is therefore to consider the values of the variances before each outer iteration and change the order accordingly. There are two deterministic possibilities; processing the elements in order of ascending or descending variances. Furthermore, a random order can be considered. Intuitively, it is beneficial to process lower variances first, because the update of a more reliable estimate for x_j yields more certainty for the estimates of other signal components.

C. Subset Update

The sequential processing enables evaluation of results before a loop through all positions is complete. We consider this by performing $T < N$ inner iterations per outer iteration. For a fixed number of outer iterations, this decreases complexity, since the overall number of rank-one updates is reduced.

Since only a subset of the signal components is processed, suitable choices for the subset need to be found. In order to implement this, a subset $\mathcal{J} \subseteq \{1, \dots, N\}$ (with cardinality $|\mathcal{J}| = T$) of positions is chosen before each outer iteration. By ordering the index set $\mathcal{J}$ in a sequence $s_{\mathcal{J}}$, this strategy can be combined with the schedules defined above.

1) Suitable Strategies: As indicated by the *state evolution* analysis (see, e.g., [15]), a suitable algorithm needs to make sure that the reliability in the estimate increases over the iterations. In the sequential algorithm, the reliability is characterized by individual variances. It is thus necessary, to ensure that all individual variances decrease over the iterations, which is done by processing the respective signal components.

This means, a higher variance indicates the need for processing. Hence, it is useful to focus on positions with high variances, i.e., choose $\mathcal{J}$ such that higher variances are preferred. One can use probabilistic or deterministic choices here.

For the probabilistic choice a distribution is constructed, where the probability for a position j to be drawn is proportional to its corresponding variance $\sigma_{c,j}^2$. For this, the index set $\mathcal{J} = \{j_1, \dots, j_T\}$ is successively built. Say, $\ell < T$ positions have already been drawn, i.e., $\mathcal{J}^{(\ell)} \stackrel{\text{def}}{=} \{j_1, \dots, j_\ell\}$. Since it is necessary to draw unique positions, these positions must be left out from the distribution, i.e., we create probabilities for the other positions by

$$\Pr\{\mathrm{j}_{\ell+1} = j\} = \frac{\sigma_{c,j}^2}{\sum_{j=1, j\notin\mathcal{J}^{(\ell)}}^{N} \sigma_{c,j}^2} \,. \tag{17}$$

A deterministic choice is to order the positions according to their variances and process only the T highest ones. The opposite, choosing the lowest variances, leads to a bad performance, since positions that have already been processed will be chosen again and again, whereas there is a high chance that some signal components are never processed.

Additionally, one can make sure that each position is processed equally often, by tracking the number of processings conducted for each position j with a counter c_j. We denote the set of positions with minimal counters by

$$\mathcal{J}_c = \{j \mid c_j \leq c_{j'} \ \forall j, j' \in \{1, \dots, N\}\} \,. \tag{18}$$

In combination with this, also the choice of positions with lowest variances becomes applicable.

2) Complexity: Due to the matrix inversion, the complexity of VAMPire is $\mathcal{O}(N^3)$. Processing only T signal components per outer iteration, during the same amount of iterations, the sequential algorithm has complexity $\mathcal{O}(N^2T)$. Furthermore, the additional operations for the choice of the index set $\mathcal{J}$ and the sequence $s_\mathcal{J}$ (especially sorting, random drawing or counter) can be implemented with complexity $\mathcal{O}(T^2)$. Hence, the complexity is reduced for $T < N$ and a fixed number of (outer) iterations.

D. Sequential Algorithm

The complete procedure is given in Algorithm 1. A similar algorithm has been stated in [14, App. D]; the difference lies in the adapted update from [7] and the optimized scheduling.

Algorithm 1: $\boldsymbol{m}_\mathrm{s} = \texttt{seqVAMPire}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_\mathsf{n}^2,\, \sigma_\mathsf{x}^2, T)$

$\tilde{\boldsymbol{x}}_\mathrm{c} = \boldsymbol{0},\ \tilde{\sigma}_{\mathrm{c},j}^2 = \sigma_\mathsf{x}^2\ \forall j$ `// Initialization`

$\boldsymbol{m}_\mathrm{c} = \tilde{\boldsymbol{x}}_\mathrm{c} + (\boldsymbol{A}^\top \boldsymbol{A} + \frac{\sigma_\mathsf{n}^2}{\sigma_\mathsf{x}^2}\boldsymbol{I}_N)^{-1}\boldsymbol{A}^\top(\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_\mathrm{c})$

$\boldsymbol{\Phi}_\mathrm{c} = \sigma_\mathsf{n}^2(\boldsymbol{A}^\top \boldsymbol{A} + \frac{\sigma_\mathsf{n}^2}{\sigma_\mathsf{x}^2}\boldsymbol{I}_N)^{-1}$

while *stopping criterion not met* **do**

- Choose index set $\mathcal{J} \subseteq \{1,\, \dots,\, N\}$ with $|\mathcal{J}| = T$
- Specify sequence $s_\mathcal{J} = [j_1,\, j_2,\, \dots,\, j_T] : j_i \in \mathcal{J}\ \forall i$
- **for** $j \in s_\mathcal{J}$ **do**
  - $\sigma_{\mathrm{c},j}^2 = [\boldsymbol{\Phi}_\mathrm{c}]_{jj},\ \boldsymbol{\phi}_{\mathrm{c},j} = [[\boldsymbol{\Phi}_\mathrm{c}]_{1j},\, \dots,\, [\boldsymbol{\Phi}_\mathrm{c}]_{Nj}]^\top$
  - $\tilde{\sigma}_{\mathrm{s},j}^2 = (1/\sigma_{\mathrm{c},j}^2 - 1/\tilde{\sigma}_{\mathrm{c},j}^2)^{-1}$ `// crossover`
  - $\tilde{x}_{\mathrm{s},j} = \tilde{\sigma}_{\mathrm{s},j}^2(m_{\mathrm{c},j}/\sigma_{\mathrm{c},j}^2 - \tilde{x}_{\mathrm{c},j}/\tilde{\sigma}_{\mathrm{c},j}^2)$
  - $m_{\mathrm{s},j} = \mathrm{E}_{\mathsf{x},\mathrm{s}}\{\mathsf{x}_j \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\}$ `// NLE`
  - $\sigma_{\mathrm{s},j}^2 = \mathrm{E}_{\mathsf{x},\mathrm{s}}\{(\mathsf{x}_j - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\}$
  - $\sigma_j^2 = \mathrm{E}_{\tilde{\mathsf{x}}_\mathrm{s}}\{\sigma_{\mathrm{s},j}^2\}$ `//` $\tilde{\mathsf{x}}_\mathrm{s} = \mathsf{x}_j + \mathsf{e},\ \mathsf{e} \sim \mathcal{N}(0, \tilde{\sigma}_{\mathrm{s},j}^2)$
  - $\tilde{\sigma}_\mathrm{old}^2 = \tilde{\sigma}_{\mathrm{c},j}^2,\ \tilde{x}_\mathrm{old} = \tilde{x}_{\mathrm{c},j}$
  - $\tilde{\sigma}_{\mathrm{c},j}^2 = \sigma_{\mathrm{s},j}^2 + (\frac{\sigma_j^2}{\tilde{\sigma}_{\mathrm{s},j}^2 - \sigma_j^2}(m_{\mathrm{s},j} - \tilde{x}_{\mathrm{s},j}))^2$
  - $\tilde{x}_{\mathrm{c},j} = \frac{\sigma_j^2\tilde{\sigma}_{\mathrm{s},j}^2}{\tilde{\sigma}_{\mathrm{s},j}^2 - \sigma_j^2}(m_{\mathrm{s},j}/\sigma_j^2 - \tilde{x}_{\mathrm{s},j}/\tilde{\sigma}_{\mathrm{s},j}^2)$
  - $\Delta\sigma^2 = \tilde{\sigma}_\mathrm{old}^2 - \tilde{\sigma}_{\mathrm{c},j}^2,\ \Delta\tilde{x} = \tilde{\sigma}_\mathrm{old}^2\tilde{x}_{\mathrm{c},j} - \tilde{\sigma}_{\mathrm{c},j}^2\tilde{x}_\mathrm{old}$
  - $\boldsymbol{m}_\mathrm{c} = \boldsymbol{m}_\mathrm{c} + \frac{\Delta\tilde{x} - \Delta\sigma^2 m_{\mathrm{c},j}}{\tilde{\sigma}_{\mathrm{c},j}^2\tilde{\sigma}_\mathrm{old}^2 + \Delta\sigma^2[\boldsymbol{\Phi}_\mathrm{c}]_{jj}}\boldsymbol{\phi}_{\mathrm{c},j}$
  - $\boldsymbol{\Phi}_\mathrm{c} = \boldsymbol{\Phi}_\mathrm{c} - \frac{\Delta\sigma^2}{\tilde{\sigma}_{\mathrm{c},j}^2\tilde{\sigma}_\mathrm{old}^2 + \Delta\sigma^2[\boldsymbol{\Phi}_\mathrm{c}]_{jj}}\boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{\phi}_{\mathrm{c},j}^\top$

IV. Numerical Results

We cover the performance of the above introduced algorithm by numerical simulations and compare it to VAMP and the variant with individual variances, VAMPire.

Denoting the sparsity by s and the Dirac delta function by $\delta(x)$, we use the prior pdf

$$\mathsf{f}_\mathsf{x}(x) = \tfrac{s}{2N}\delta(x+1) + \tfrac{N-s}{N}\delta(x) + \tfrac{s}{2N}\delta(x-1)\ , \qquad (19)$$

which has zero-mean and a-priori variance $\sigma_\mathsf{x}^2 = s/N$. The formulas for the signal-constrained estimates $m_{\mathrm{s},j}$ and $\sigma_{\mathrm{s},j}^2$ can be found in [17]. The sensing matrix $\boldsymbol{A}$ is i.i.d. Gaussian distributed. In order to model the distribution of transmit powers, we implement an exponential power profile and amplify the columns of $\boldsymbol{A}$ accordingly. The power profile is given by $r_p = d^{(p-1)/(N-1)}$ ($p \in \{1,\, \dots,\, N\}$); the assignment to the jth column is obtained from a random permutation $j = \pi(p)$. The simulations are obtained for a factor $d = 0.2$. After applying the power profile (by $\boldsymbol{A} \cdot \mathbf{diag}(r_{\pi(p)})$), the sensing matrix is scaled such that it has Frobenius norm $||\boldsymbol{A}||_\mathrm{F} = \sqrt{N}$. The signal-to-noise ratio is fixed to $-10\log_{10}(\sigma_\mathsf{n}^2) \mathrel{\widehat{=}} 17$ dB in all simulations. For numerical stability, especially in the subtraction of the inverse variances in (9), we clip $\sigma_{\mathrm{s},j}^2$ to the interval $[10^{-8},\, 10^{8}]$ and the other variances to $[10^{-12},\, 10^{12}]$.

As performance measure the average (over the number of realizations) symbol error ratio (SER), i.e., the relative number of incorrectly recovered symbols $\mathrm{SER} = |\{m_{\mathrm{s},j} \neq x_j\}|/N$ is considered. After reconstruction, the known sparsity is used by taking the s (in magnitude) largest $m_{\mathrm{s},j}$ values, quantizing them to $\mathrm{sgn}(m_{\mathrm{s},j})$ and setting all others to 0, cf. [17]. The convergence is shown by plotting the SER over the number of outer iterations. We compare the following schedules:

- random: $s_\mathcal{J} = [\pi(1),\, \dots,\, \pi(N)]$,
- sort $\sigma^2\searrow$: order of descending variances,
- sort $\sigma^2\nearrow$: order of ascending variances.

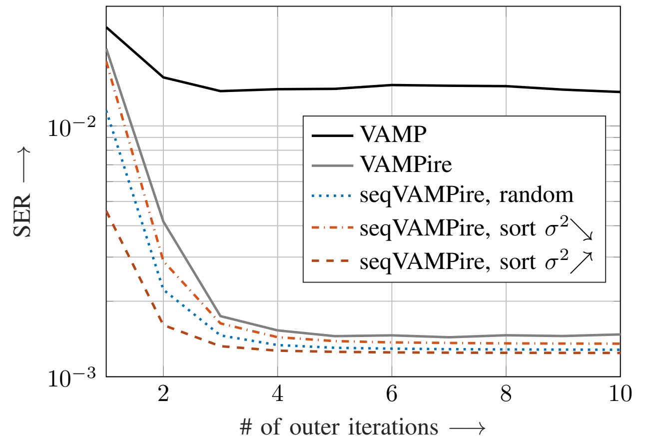

Fig. 3. Convergence analysis for different schedules. Parameters: $N = 100$, $M = 50$, $s = 5$, $-10\log_{10}(\sigma_\mathsf{n}^2) \mathrel{\widehat{=}} 17$ dB, $T = N$, 10^6 simulations.

The result is shown in Fig. 3. One can see, that the sequential algorithms converge faster than the other two, which can be explained by the fact that throughout the loop over the positions, the signal component to be processed benefits from the knowledge that was gained from previously processed positions. The processing in the order of ascending variances has most benefit in convergence speed and steady-state error. This can be explained by the fact that the update of a more reliable estimate for x_j yields more certainty for the estimates of other signal components.

In the following, the opposite is examined. We set $T = 1$, so that after each processed signal component, we evaluate and choose a new position j to process. The choice of a schedule $s_\mathcal{J}$ is hence superfluous. We compare the following strategies:

- minv: choose $j = \mathrm{argmin}_{j' \in \mathcal{J}_\mathrm{c}}\, \sigma_{\mathrm{c},j'}^2$ with $\mathcal{J}_\mathrm{c}$ from (18),
- maxv: choose $j = \mathrm{argmax}_{j' \in \mathcal{J}_\mathrm{c}}\, \sigma_{\mathrm{c},j'}^2$ with $\mathcal{J}_\mathrm{c}$ from (18),
- rv: draw position from $\{1,\, \dots,\, N\}$ randomly with probability given by (17),

and combinations thereof. The combinations are realized by switching strategies after every processed signal component.

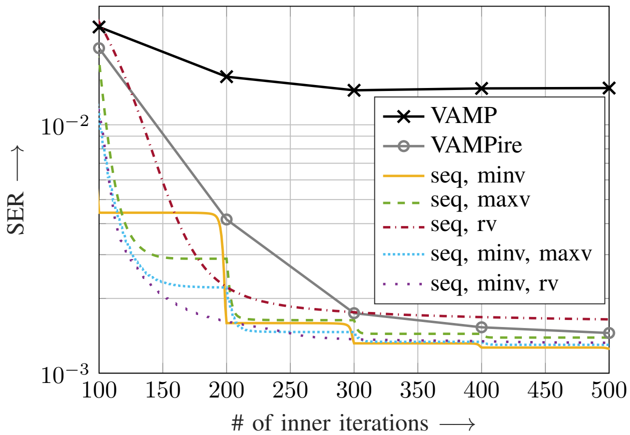

Fig. 4. Convergence analysis for different index choices. Parameters: $N = 100$, $M = 50$, $s = 5$, $-10\log_{10}(\sigma_n^2) \mathrel{\hat{=}} 17$ dB, $T = 1$, 10^6 simulations.

The convergence over inner iterations is shown in Fig. 4. As a comparison, we plot the evaluation of the parallel algorithms at multiples of signal dimension N. The counters in the first two strategies ensure that all signal components are processed once between consecutive multiples of N. This forces the processing order to loop through all positions between these multiples. The choice of the highest variance (maxv) from the (per loop) non-processed signal components shows a fast convergence at the beginning of such a loop, since those positions require processing at most. Towards the end of the cycle, the performance flattens out, since only positions with relatively high variances are processed. This leads to a staircase behavior. The opposite strategy, choosing the lowest variance (minv) behaves contrarily. At the beginning of a loop almost no change is visible. Towards the end, a sudden drop appears, since at that point the positions, which require processing most, are considered. Because the lower variances were processed first in this strategy, which gives additional reliability to subsequently processed positions, the drop at the end of the loop surpasses the performance of the strategy with high variances first (maxv). The random approach does not make use of the counters and therefore converges smoothly, but since it is not ensured that every position is processed regularly, the performance is worse.

Combining opposing strategies, it is possible to achieve both, a fast convergence and well performance at convergence. By alternatingly choosing high and low variances, the convergence property of high variance preference is obtained, while achieving better performance at the end of a cycle through all signal components, however the staircase behavior stays. On the other hand, the combination of the probabilistic approach with low variance preference converges smooth and overall well.

V. Conclusion

A sequential version of a VAMP algorithm with individual variances was introduced and assessed. The sequential processing offers a variety of possibilities to adjust the necessary effort to attain a desired performance level. Strategies for schedules were presented and discussed. The algorithm shows a faster convergence than parallel algorithms and offers evaluation of subsets of the signal vector. It was shown by numerical simulations that with suitable schedules the algorithm performs already very well for only relatively few processed signal components, thereby avoiding the higher complexity caused by the usage of individual variances.

Acknowledgment

The authors would like to thank Ralf R. Müller for valuable discussions and acknowledge support by the state of Baden-Württemberg through bwHPC.

References

[1] A. Braunstein, A. P. Muntoni, A. Pagnani, and M. Pieropan, “Compressed Sensing Reconstruction using Expectation Propagation,” *Journal of Physics A: Mathematical and Theoretical*, vol. 53, no. 18, 2020.

[2] L. D. Brown, *Fundamentals of Statistical Exponential Families: with Applications in Statistical Decision Theory*. Institute of Mathematical Statistics, 1986.

[3] 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, 2006.

[4] A. I. V. Casado, M. Griot, and R. D. Wesel, “Informed Dynamic Scheduling for Belief-Propagation Decoding of LDPC Codes,” in *IEEE International Conference on Communications*, 2007, pp. 932–937.

[5] D. L. Donoho, “Compressed Sensing,” *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[6] A. Duel-Hallen, J. Holtzman, and Z. Zvonar, “Multiuser Detection for CDMA Systems,” *IEEE Pers. Comm.*, vol. 2, no. 2, pp. 46–58, 1995.

[7] R. F. H. Fischer, C. Sippel, and N. Goertz, “VAMP with Vector-Valued Diagonalization,” in *2020 International Conference on Acoustics, Speech and Signal Processing*, 2020.

[8] Q. Guo and D. D. Huang, “A Concise Representation for the Soft-In Soft-Out LMMSE Detector,” *IEEE Commun. Lett.*, vol. 15, no. 5, pp. 566–568, 2011.

[9] J. Haupt, R. Baraniuk, R. Castro, and R. Nowak, “Sequentially Designed Compressed Sensing,” in *IEEE Statistical Signal Processing Workshop (SSP)*, Oct. 2012, pp. 401–404.

[10] M. Leinonen, M. Codreanu, and M. Juntti, “Sequential Compressed Sensing With Progressive Signal Reconstruction in Wireless Sensor Networks,” *IEEE Wireless Commun.*, vol. 14, no. 3, pp. 1622–1635, 2015.

[11] A. Maleki, “Approximate Message Passing Algorithms for Compressed Sensing,” Ph.D. dissertation, Sep. 2011.

[12] D. M. Malioutov, S. R. Sanghavi, and A. S. Willsky, “Sequential Compressed Sensing,” *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 2, pp. 435–444, 2010.

[13] A. Manoel, F. Krzakala, E. Tramel, and L. Zdeborovà, “Swept Approximate Message Passing for Sparse Estimation,” in *International Conference on Machine Learning*, 2015, pp. 1123–1132.

[14] M. Opper and O. Winther, “Expectation Consistent Approximate Inference,” *Journal of Machine Learning Research*, vol. 6, pp. 2177–2204, Dec. 2005.

[15] S. Rangan, P. Schniter, and A. K. Fletcher, “Vector Approximate Message Passing,” *IEEE Trans. Inf. Theory*, vol. 65, no. 10, pp. 6664–6684, 2019.

[16] M. W. Seeger and H. Nickisch, “Compressed Sensing and Bayesian Experimental Design,” in *Proceedings of the 25th International Conference on Machine Learning*, 2008, pp. 912–919.

[17] S. Sparrer and R. F. H. Fischer, “Algorithms for the Iterative Estimation of Discrete-Valued Sparse Vectors,” in *11th International ITG Conference on Systems, Communications and Coding*, Feb. 2017.

[18] ——, “Unveiling Bias Compensation in Turbo-Based Algorithms for (Discrete) Compressed Sensing,” in *25th European Signal Processing Conference (EUSIPCO)*, 2017, pp. 2091–2095.

[19] M. E. Tipping and A. C. Faul, “Fast Marginal Likelihood Maximisation for Sparse Bayesian Models,” in *Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics*, 2003.