Abstract—The pure greedy algorithms matching pursuit (MP) and complementary MP (CompMP) are extremely computationally simple, but can perform poorly in solving the linear inverse problems posed by the recovery of compressively sampled sparse signals. We show that by applying a cyclic minimization principle, the performance of both are significantly improved while remaining computationally simple. Our simulations show that while MP and CompMP may not be competitive with state-of-the-art recovery algorithms, their cyclic variations are. We discuss ways in which their complexity can be further reduced, but our simulations show these can hurt recovery performance. Finally, we derive the exact recovery condition of CompMP and both cyclic algorithms.

I. INTRODUCTION

Under certain conditions, we can recover a vector \( x \in \mathbb{C}^N \) from measurements \( u = \Phi x \) where \( \Phi \in \mathbb{C}^{m \times N} \). Several methods have been developed and studied for recovering \( x \) from \( u \), which are obviously sensitive to both the size and content of \( \Phi \) relative to \( x \). The overdetermined problem \((N \leq m)\) has been formally studied in frame theory [1]. The underdetermined problem \((N > m)\), with \( x \) a sparse vector, has been studied in statistics, inverse problems, sparse approximation [1], [2], and compressed sensing [1], [3], [4].

We propose solving the noiseless underdetermined problem by a cyclic application of the pure greedy algorithms matching pursuit (MP) [2], [5], [6], and complementary MP (CompMP) [7]. A pure greedy algorithm has an extremely low computational complexity, but at the same time does not generally perform well in solving the above problem. This can be seen by a comparison of their empirical phase transitions [8], [9] (Fig. 1). A cyclic approach to recovery involves alternating between augmentation and refinement of the model, and here we employ a pure greedy algorithm for each to great effect.

Of course one should expect that such an approach will perform better than MP or CompMP, but we find the surprising result that the performance equals or exceeds those of more complex approaches while being more computationally simple. We show empirically that a pure greedy algorithm within a cyclic minimization framework, its performance can equal or exceed those of orthogonal MP (OMP) [2], [10], [11], and orthogonal least squares (OLS) [12], [13], and yet have a complexity in their \( k \)th iteration of \( O(Nk) \). In comparison, the complexity of OMP it is at least \( O(Nk + mk) \), and for OLS it is at least \( O(mN + mk) \), both using the QR decomposition approach. We compare their performance to six state-of-the-art recovery algorithms, and find CompMP is especially competitive. We propose a variety of ways to further reduce their algorithmic complexity, and test their performance in relation to their parameters. Finally, we derive an exact recovery condition (ERC) of CompMP [7], from which naturally follow the ERC of a cyclic pure greedy algorithm.

II. PURE GREEDY ALGORITHMS

MP attempts to iteratively solve

\[
\min_{x} \|x\|_0 \text{ subject to } u = \Phi x
\]  

for \( x \), where \( \Phi \) is the measurement matrix and \( u \) is the measurement vector. The goal is to find the sparsest vector \( x \) that satisfies the equation \( u = \Phi x \). One way to achieve this is through a greedy algorithm, which iteratively selects the vector that has the largest correlation with the residual at each iteration. This process is repeated until either a maximum number of iterations is reached or the residual is below a certain threshold.

![Diagram](image-url)
precomputes and stores the Gramian of $\Phi$ [5].

Instead of (1), CompMP [7] attempts to iteratively solve
\[
\min_x \| \Phi^* \|_{0, \text{subject to } y = \Xi x} \tag{4}
\]
where $y := \Phi^T (\Phi \Phi^T)^{-1} u$, and $\Xi = [\xi_1, \xi_2, \ldots, \xi_N] := \Phi^T (\Phi \Phi^T)^{-1} \Phi$. As MP, CompMP augments $\Omega_k$ with the index
\[
n_k := \arg \max_{n \in \Omega} \frac{\|y_k - (y_k - \xi_n)\|^2}{\|\xi_n\|^2} = \arg \max_{n \in \Omega} \frac{\|y_k - \xi_n\|^2}{\|\xi_n\|^2} \tag{5}
\]
where $y_k := \Phi^T (\Phi \Phi^T)^{-1} [u - \Phi x_k]$, and then updates the solution
\[
x_{k+1} := x_k + \frac{(y_k - \xi_n)}{\|\xi_n\|^2} e_n. \tag{6}
\]
As for MP, we initialize CompMP by $\Omega_0 := \emptyset$ and $x_0 := 0$, and can use the same stopping conditions, and procedures for duplicate atoms.

Note that we cannot guarantee in general that $\Xi$ consists of unit norm columns, and so CompMP must take their norms into account for the projections. If we define $\Psi := \Xi \Delta_\mathbb{R}$, where $\Delta_\mathbb{R}$ is a $N \times N$ diagonal full rank matrix, with its $i$th diagonal element defined $\|\xi_i\|_2^2$, then each column of $\Psi$ is unit norm. With $\Psi$ and $y$, then MP attempts to solve the problem $\min_x \| z \|_0$ subject to $y = \Psi z$ from which we obtain the solution to (4) if we then define $x := \Delta_\mathbb{R} y$. Thus, CompMP with $\Psi$ on $y$ is exactly as MP with $\Psi$ on $y$, and a final weight adjustment step. In any case, $\text{Supp}(z) = \text{Supp}(x)$ since $\Delta_\mathbb{R}$ is diagonal and full rank. From this, we see that each iteration of CompMP and MP have the same complexity.

The general stability and convergence properties of MP are well-known within the area of non-linear approximation [2], [5], [6], [15]. Let $\Omega^* := \{1, 2, \ldots, s\} \subseteq \Omega, s \leq m$, where, without loss of generality, the atoms a pure greedy algorithm must detect are the first $s$ columns of $\Phi$. We assume that $\Psi_{\Omega^*}$ is made of the columns of $\Phi$ indexed by $\Omega^*$, and has full column rank $s$. Consider the vector space $U_{\Omega^*} := \text{Image} (\Phi_{\Omega^*})$. Then, for every $u \in U_{\Omega^*}$, any solution assembled by a general MP algorithm involves only atoms indexed by $\Omega^*$
\[
\max_{n \in U_{\Omega^*}} \| \Phi_{\Omega^*}^T \psi_n \|_2 < 1. \tag{ERC MP($\Omega^*$)}
\]

The general stability and convergence properties of CompMP have yet to be formally studied, but from its near equivalence to MP shown above, and the fact that it is a general MP algorithm, it must have similar properties. For any $u \in U_{\Omega^*}$, we see that $y := \Phi^T (\Phi \Phi^T)^{-1} u = \Psi_{\Omega^*} z'$, where $\Psi_{\Omega^*}$ are the first $s$ columns of $\Psi$, and $z'$ is made of the first $s$ elements of $z$. So, $y \in \mathcal{Y}_{\Omega^*} := \text{Range} (\Psi_{\Omega^*})$. Now, given a $u \in U_{\Omega^*}$, any solution assembled by CompMP involves only atoms indexed by $\Omega^*$ if
\[
\max_{n \in \Omega^*} \| \Phi_{\Omega^*}^T \psi_n \|_2 < 1. \tag{ERC Comp MP($\Omega^*$)}
\]

The cyclic minimization principle has been applied to MP for audio coding and modeling [17], [18], as well as to OLS [18]. Recent work [19] applies it to the compressibility of synthetic sparse signals using OLS and a complementary form of OLS. This work [19] is related to what we propose here, however, Dymarski et al. refine the atoms using cyclic minimization at the very end of assembling the signal model, while we propose alternating between atom selection and model refinement. Our experiments below show that the latter approach significantly outperforms the former.

A. Cyclic MP (CMP) and Complementary CMP (CompCMP)

The cyclic pure greedy algorithm (CMP) [17], runs as MP with $\Phi$ at each iteration, but includes a solution refinement step that also runs as MP. Similarly, CompCMP runs as CompMP with $\Xi$ (or equivalently MP with $\Psi$) at every iteration, but includes a refinement step that also runs as CompMP (or MP). Since it requires only a trivial change of notation — $\chi$ becomes $\psi$, and $r_k := u - \Phi x_k$ becomes $y_k := \Phi^T (\Phi \Phi^T)^{-1} u - \Psi z_k$ — we present only CMP.

Define $\Omega_k(i)$ as the $i$th index in $\Omega_k$, $1 \leq i \leq |\Omega_k|$, and $r_k(0) := u - \Phi x_k$ as the residual before refinement. Starting with the first element of $\Omega_k$, CMP removes $\psi_{\Omega_k(1)}$ from the residual by adding it back with the same weight with which it was taken
\[
r_{k\setminus\setminus 1} := r_{k\setminus\setminus 0} + [x_{\Omega_k(1)}] \psi_{\Omega_k(1)} \tag{8}
\]
then removes it from the solution by $[x_{\Omega_k(1)}] \psi_{\Omega_k(1)} \leftarrow 0$, and finally finds the index of a replacement atom by the MP selection criterion (2):
\[
n_1 := \arg \max_{n \in \Omega} \| [r_{k\setminus\setminus 1} \psi_n] \|_2^2. \tag{9}
\]
CMP then updates $\Omega_k$ such that $\Omega_k(1) \leftarrow n_1$, and adjusts the solution
\[
x_k \leftarrow x_k + [x_{\Omega_k(1)}] \psi_{\Omega_k(1)} \tag{10}
\]
Note that $n_1$ could be the same as the original index $\Omega_k(1)$, in which case only its weight is adjusted. Finally, the residual is updated by $r_{k\setminus\setminus 1} := r_{k\setminus\setminus 1} - [x_{\Omega_k(1)}] \psi_{n_1}$, and the second atom is refined. CMP in general is shown above.

Refinement can be done any number of times for all elements of $\Omega_k$. Once finished, CMP then augments $\Omega_k$, and performs refinement cycles on the enlarged set. One can implement CMP to stop this refinement process after a certain number of cycles, or until the improvement between cycles decreases below a threshold. For
instance, defining an error measure of the \( k \)th-order solution after \( l \) refinement cycles \( J(\Omega_k) := \| \mathbf{u} - \Phi \mathbf{x}_k \|_2^2 \), we can stop refinement when \( J(\Omega_k) > \eta J(\Omega_0) \), \( 0 < \eta < 1 \), where \( \Omega_0 \) is the set after a refinement cycle. We can also stop CMP from augmenting the model when \( J(\Omega_k) > J(\Omega_{k-1}) \) for \( 0 < \epsilon < 1 \). Each model refinement and augmentation step of CMP (as well as CompCMP) never increases the residual \( \ell_2 \)-norm because of the atom selection criterion [18].

### B. Computational Efficiency

When \( \Phi \) is small enough such that its Gramian can be precomputed and stored, we can efficiently perform the refinement cycles using the same tricks as for MP [5]. Define \( \Gamma := \{ (\varphi_i, \varphi_n) \}_{n \in \Omega} \), which is the ordered set of the elements from the \( l \)th column of \( \Phi \). In the \( l \)th refinement, \( 1 \leq l \leq |\Omega_k| \), CMP defines the intermediate set

\[
I_{k,l-1} := \{ I_{k,l-1}(n) + x_{n_l} \Gamma_{n_l}(n) \}_{n \in \Omega}
\]

where \( I_{k,0} := \mathbb{I}_k \). This operation requires \( O(N) \) multiplies and adds. CMP then sets \( \alpha \leftarrow x_{n_l} \), and \( x_{n_l} \leftarrow 0 \). CMP then finds the index \( n_l' \) of the largest magnitude in \( I_{k,l-1} \), updates the set

\[
J_{k,l} := \{ I_{k,l-1}(n) - I_{k,l-1}(n_l') \}_{n \in \Omega}
\]

and updates the atom amplitude \( x_{n_l}' \leftarrow x_{n_l} + I_{k,l-1}(n_l) \), and computes the new residual energy by \( \| \mathbf{r}_k \|_2^2 \leftarrow \| \mathbf{r}_k \|_2^2 + |\alpha|^2 + 2\text{Re}\{\alpha I_{k,l-1}(n_l')\} - |I_{k,l-1}(n_l')|^2 \). Once refinement is finished, CMP sets \( I_k \leftarrow I_{k,l} \), and augments the model with a new atom.

All of this takes \( O(N) \) multiplies and adds, and a search with cost at most \( O(N) \). Thus, each refinement cycle of a \( k \) order model takes \( O(Nk) \). If CMP refines the model a total of \( L \) times at each of \( s \) augmentations, the computational complexity is \( O(sN + \sum_{k=2}^{L} kLN) = O(sN + NLs^2) \). For a naive implementation, the computational cost of each model augmentation is \( O(Nm + N) \), and for each refinement of a \( k \)-order model the total cost is \( O(Nmk + Nk) \). This brings its total computational complexity to \( O(Nms + NLms^2) \), for \( L \) refinements at each of \( s \) augmentations. The first approach thus provides savings, but still scales with \( s^2 \). We might overcome this problem by refining the model only after \( K \geq 1 \) additional augmentations (CMP-K), or only once the target sparsity is reached. Another possibility is refining the atom amplitudes while holding the atom indices constant. In this case, no search in the dictionary is needed, and CMP only changes the atom amplitudes by \( \| \mathbf{r}_k \|_2^2 \leftarrow |\alpha|^2 + 2\text{Re}\{\alpha I_{k-1}(n_l')\} - |I_{k-1}(n_l')|^2 \). Once refinement is finished, CMP sets \( I_k \leftarrow I_{k-1} \), and augments the model with a new atom.

### C. Stability Condition of CMP-K and CompCMP-K

Since CMP-K and CompCMP-K are general MP algorithms [15], they have the same stability conditions as MP and CompCMP, respectively. In short, as long as either (ERC\(_{\text{CMP}}(\Psi, \Omega^*)\)) or (ERC\(_{\text{CompCMP}}(\Psi, \Omega^*)\)) hold, all refinement steps of either MP or CompMP, respectively, will select atoms from those indexed by \( \Omega^* \).

### IV. Experimental Results

In our tests, we investigate the performance of the cyclic algorithms in recovering the support of the true solution to \( \mathbf{u} = \Phi \mathbf{x} \), i.e., the locations of the non-zero values in \( \mathbf{x} \). Our problem suite lives in an ambient dimension \( N = 400 \), and has 480 pairs of sparsities and indeterminacies: 30 linearly-spaced sparsities \( \rho := s/m \in [0.05, 1] \), and 16 linearly-spaced indeterminacies \( \delta := m/N \in [0.05, 0.5414] \). For each sparsity and indeterminacy pair, we sample the non-zero values from the uniform spherical ensemble [9], and find the proportion of 100 random vectors sensed by \( \Phi \) that are exactly recovered.

Our criteria for exact recovery is that the support of the recovered solution matches that of the true solution, i.e., no missed detections or false alarms. From these results, we find the empirical phase transition [8] by estimating the parameters of a linear model of the logistic of the frequencies [9]. To synthesize a \( s \)-sparse vector, we first determine indices of its non-zero elements by drawing at random and independently \( s \) members from \( \Omega := \{ 1, 2, \ldots, N \} \). Then, we find the values at these indices by sampling from one of two distributions: Normal (and all magnitudes greater than \( 10^{-10} \)), and Rademacher, or equiprobable in \( \{-1, 1\} \). The first distribution is favored by greedy algorithms, while the second appears to be the least favorable for all algorithms [9], [20].

We initialize every algorithm with the empty set, and set each one to perform a maximum of \( 2s \) augmentations, where \( s \) is the true signal sparsity. We debias the support of each solution from an algorithm by performing the following steps:

1. find indices of elements with magnitudes \( > \min(m, 2s) \) largest
2. refine amplitudes of these elements by least squares projection
3. define solution support as the indices of elements with magnitudes above \( 10^{-10} \).

Because of this addition, we denote MP as MP+, and CompMP as CompMP+. All algorithms are made to exit once either it has performed \( 2s \) iterations, or \( \| \mathbf{u} - \Phi \mathbf{x}_k \|_2^2 / \| \mathbf{u} \|_2^2 < 10^{-10} \) for MP and CMP, or \( \| \mathbf{y} - \Psi \mathbf{z}_k \|_2^2 / \| \mathbf{y} \|_2^2 < 10^{-10} \) for CompMP and CompCMP.
A. Computational Cost

In this experiment, we compare the computational complexities of all four algorithms, with respect to the number of multiplies, over the portion of the phase space which we are testing. We count the number of multiplies performed at each of 480 points, regardless of successful recovery, and average the results from 100 vectors. Figure 2 compares the complexities of all four algorithms. Though CompMP and CompCMP-1 have a high initial cost, they both scale remarkably well over the phase plane, and CompCMP shows nearly the same complexity as CompMP. Likewise, the complexity of CMP-1 stays close to that of MP+ until more iterations are required when the solution support is larger. In the same way, Fig. 3 compares the complexity of MP, CMP-1, and OMP. We see that though the empirical phase transitions of CMP-1 and OMP are practically identical, it is completely in the region where CMP is less expensive than OMP overall. Though each of its iterations have a lower computational complexity than that of OMP, CMP-1 can require many more iterations to reach the same model order.

B. Number of Refinement Cycles in CMP-1 and CompCMP-1

In this experiment, we force the cyclic algorithms to perform $L$ refinements after each model augmentation. Figure 4 shows the resulting empirical phase transitions from each signal distribution. For even just a single refinement cycle, we see for Normal signals significant improvements in both algorithms over MP+ and CompMP+. When $L=2$, the performance of CMP is comparable to that of OMP, but changes little for larger numbers of refinements. For only one refinement cycle, the performance of CompCMP exceeds that of OMP, but changes little for larger numbers of refinements. For only one refinement cycle, we see for Normal signals significant improvements in both algorithms over MP+ and CompMP+. When $L=2$, the performance of CMP is comparable to that of OMP, but changes little for larger numbers of refinements. For just a single refinement cycle, we see for Normal signals significant improvements in both algorithms over MP+ and CompMP+. When $L=2$, the performance of CMP is comparable to that of OMP, but changes little for larger numbers of refinements.

C. Effect of K in CMP-K and CompCMP-K

In this experiment, we look at the effect of refining the model only every $K$ augmentations. Each algorithm stops refinement when either $\|u - \Phi z_k\|^2 > (1-10^{-5})\|u - \Phi z_k\|^2$ (or for the complementary approaches $\|y - \Psi z_k\|^2 > (1-10^{-5})\|y - \Psi z_k\|^2$), where $z_k$ and $x_k$, or $z_k$ and $x_k$, are the solutions before after a refinement cycle, respectively, or when ten refinement cycles have been completed. Even with this maximum number of refinements to 10, neither algorithm refined the model that number of times. Figure 5 shows how the empirical phase transitions change for each signal distribution. We can clearly see that waiting several augmentations to refine the model reduces the computational complexity, but results in poorer recovery performance. Surprisingly, however, the performance of CMP is hurt less by this than CompCMP for signals distributed Normal.

D. Comparison with other recovery algorithms

In this final experiment, we compare the performance of CMP-1 and CompCMP-1 to the following collection of algorithms: MP+ and CompMP+; OMP [2], [10], [11]; orthogonal least squares (OLS) [12], [13]; “recommended versions” of iterative hard thresholding (IHT) and two-stage thresholding (TST) [9]; probabilistic OMP (PrOMP) [21]; smoothed $\ell_0$ (SL0) [22]; subspace pursuit (SP) [23]; approximate message passing (AMP) [24]. The cyclic algorithms stop refinement when either it performs two refinement cycles (based on the results of experiment 1 above), or using the same $\ell_2$-norm criteria in part C above. Figure 6 shows the empirical phase transitions for the two signal distributions. In both cases, CMP-1 performs nearly the same as OMP, but with smaller iteration-wise complexity: $O(Nk)$ versus $O(Nk + Mk)$ for the $k$th iteration. CompCMP-1 performs surprisingly well for Normal sparse signals, exceeding all others except SL0, and PrOMP at low indeterminacies. For Rademacher signals we see that CompMP still performs better than CompMP+, but all algorithms are here exceeded by AMP and $\ell_1$ minimizations (DT). Experiments at larger ambient dimensions ($N \geq 1000$) show only small changes.

V. Conclusion

We have shown empirically that in solving the inverse problem posed by CS, the performance of the pure greedy algorithms MP and CompMP can be greatly improved with a cyclic minimization principle for refining the model after a number of model augmentations. In this way, we can maintain their computational simplicity while boosting their performance. Under certain conditions too, we see they perform better than many more complex algorithms for the recovery of sparse vectors from compressive measurements. However, it appears from our experiments that waiting to refine the model can significantly hurt performance. Our future work will look at the performance of these algorithms when the measurements are noisy, and alternative strategies for deciding which support elements should be refined in order to further reduce the complexity of the algorithms.

ACKNOWLEDGMENT

This work is supported in part by a grant from the French Ambas- sador to Denmark, Afdelingen for Forsknings- og Universitetssamarbejde, No. 31/2011-CSU 8.2.1. R. Gribonval acknowledges support from EU Framework 7 FET-Open project FP7-ICT-225913-SMALL.
Fig. 5. Empirical phase transitions for CMP-K and CompCMP-K for several K as a function of signal sparsity s (labeled).

Fig. 6. Empirical phase transitions for several algorithms (labeled). Dashed line labeled ‘DT’ is theoretical phase transition for $\ell_1$-minimization [3], [8], [14].

REFERENCES