

1 The Dual Linear Program

The traditional ℓ_1 -minimization problem

$$\min_{\mathbf{s}} \|\mathbf{s}\|_1 \quad \text{such that} \quad \mathbf{y} = \mathbf{A}\mathbf{s} \quad (1)$$

can be converted to its *standard form* using nonnegative coefficients

$$\min_{\tilde{\mathbf{s}}} \mathbf{1}^T \tilde{\mathbf{s}} \quad \text{such that} \quad \mathbf{y} = \tilde{\mathbf{A}}\tilde{\mathbf{s}}, \quad \tilde{\mathbf{s}} \geq 0 \quad (2)$$

where $\mathbf{1}$ is a column vector of ones, $\tilde{\mathbf{A}} = [\mathbf{A}, -\mathbf{A}]$ and $\tilde{\mathbf{s}}$ is the $2N$ nonnegative vector.

The above linear program has a corresponding *dual linear program*

$$\max_{\mathbf{c}} \mathbf{y}^T \mathbf{c} \quad \text{such that} \quad \tilde{\mathbf{A}}^T \mathbf{c} \leq \mathbf{1} \quad (3)$$

which has an optimum solution \mathbf{c}^* associated with the optimum solution \mathbf{s}^* of (2).

2 Polytope Faces Pursuit

Polytope Faces Pursuit (PFP) is a greedy algorithm that

- performs Basis Pursuit (BP) with Orthogonal Matching Pursuit (OMP) complexity
- adopts a path-following approach based on the geometry of the polar polytope associated with the dual linear program
- uses Cholesky factorization to update the solution vector at each iteration.

The solution vector of the dual linear problem is updated at the k -th step of the algorithm using the method of Cholesky factorization:

$$\tilde{\mathbf{s}}^k = (\tilde{\mathbf{A}}^k)^\dagger \mathbf{y} \quad (4)$$

The drawback of this method is that for large-scale problems it

- increases memory requirements as it requires storage of large matrices
- affects the convergence speed of the algorithm

3 PFP with directional updates

To overcome these Cholesky limitations, methods based on directional updates can be used instead (e.g. gradient descent or conjugate gradient).

However:

- at each iteration the algorithm encounters a new face of the polar polytope
- the new atom is added to the already selected subset
- the dimensionality of the least squares problem increases

Therefore, at the k -th step, the algorithm needs a full conjugate gradient update step.

Nevertheless, in cases where fast convergence is more important than accuracy, suboptimal or approximate methods that reduce the complexity of the full conjugate gradient step could be adopted.

4 PFP with approximate conjugate gradient updates

In that case, we attempt to estimate the new direction that will be “conjugate” to the directions of the previous steps of the algorithm. Therefore, the new direction, the solution vector and the corresponding residual at the k -th step of the algorithm will be:

$$\mathbf{d}^k = \tilde{\mathbf{A}}^k \mathbf{r}^{k-1} + \beta^k \mathbf{d}^k \quad (5)$$

$$\tilde{\mathbf{s}}^k = \tilde{\mathbf{s}}^{k-1} + \alpha^k \mathbf{d}^k \quad (6)$$

$$\mathbf{r}^k = \mathbf{r}^{k-1} - \alpha^k (\tilde{\mathbf{A}}^k)^T \mathbf{d}^k. \quad (7)$$

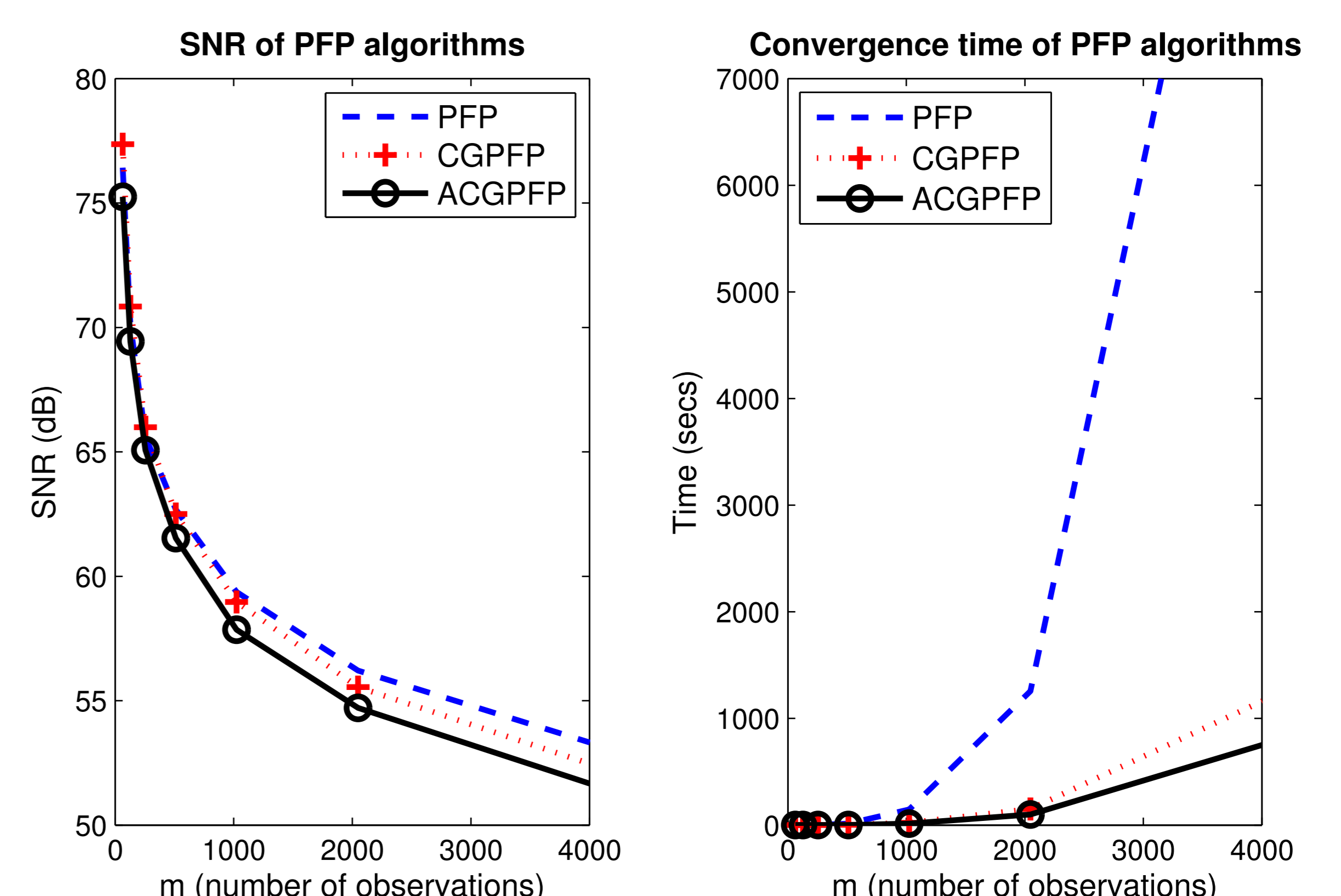
We can also estimate the basis vertex \mathbf{c} at step k using the iterative formula $\mathbf{c}^k = \mathbf{c}^{k-1} + \lambda^k (\mathbf{r}^{k-1} - \mathbf{r}^k)$, where λ^k is the atom selection criterion given by:

$$\lambda^k = \arg \min_{\mathbf{a}_i \notin \tilde{\mathbf{A}}^k} \frac{1 - \mathbf{a}_i^T \mathbf{c}^{k-1}}{\mathbf{a}_i^T \mathbf{r}^{k-1}}. \quad (8)$$

The resulting algorithm is called *Approximate Conjugate Gradient Polytope Faces Pursuit* (ACGPFP).

5 Experimental evaluation

To evaluate the proposed method’s performance, the number of observations m was increased from 32 to 4,096 while the number of variables was increased proportionally as $n = 2m$. The sparsity level k was kept constant at $k/m = 0.125$. The columns of the overcomplete dictionary $\mathbf{A} \in \mathbb{R}^{m \times n}$ were drawn from an i.i.d. normal distribution, the sparse vector \mathbf{s} was generated by drawing k elements from an i.i.d. distribution and placing them at k random entries of \mathbf{s} . Therefore, the observations vector was generated as $\mathbf{y} = \mathbf{A}\mathbf{s} + \mathbf{w}$, where \mathbf{w} is the i.i.d Gaussian additive noise. The SNR was kept constant at 90 dB for this experiment.



6 Conclusions

- The proposed method reduces the complexity of the CGPFP algorithm using an approximate conjugate gradient method
- The ACGPFP algorithm achieves faster convergence, but approximate sparse recovery.