Sparse Approximation for Signal Recovery

Tanmay Agarwal, Hitesh Arora, Shubhankar Deshpande

16-811:Maths Fundamentals For Robotics Course Project Report

Abstract

Many real world machine learning problems require solving a non-convex optimization problem and this has been a niche area of research due to non-convex problems being NP-hard to solve. In this project, we explore the space of non-convex optimization for common machine-learning problems. We motivate the non-convex problem formulations of sparse recovery and low-rank matrix recovery with its important real world applications and share the fundamental ideas about the two main solution approaches: *convex relaxation* and *direct non-convex optimization*. We also explain some of the popular non-convex optimization algorithms that have been shown to be very efficient in practice and yield provably optimal solutions in polynomial time, such as projected gradient descent and its variants. We then discuss one such application of sparse recovery in astronomy and share results from our implementation of start-of-the-art algorithm (**Högboms CLEAN**) to denoise noisy spatial images.

CONTENTS

1 INTRODUCTION

A general optimization problem can be formulated as:

$$\begin{aligned} &\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) \\ &\text{s.t. } \mathbf{x} \in \mathbf{C} \end{aligned}$$

where $\mathbf{x}$ is the variable of the problem with $\mathbf{x} \in \mathbb{R}^p$, $f : \mathbb{R}^p \to \mathbb{R}$ is the objective function, and $\mathbf{C} \subseteq \mathbb{R}^p$ is the constraint set of the problem.

An optimization problem is called as **convex** optimization problem if both the objective function and constraint set are convex, i.e., the objective is a convex function and the constraint set is a convex set. An optimization problem that violates either of these conditions, i.e., if it has a non-convex objective function or a non-convex constraint set, or both, is said to be a **non-convex** optimization problem.

1.1 *Motivation*

In many real world machine learning problems where we desire to learn a model from available data, we often come across problems having extremely high-dimensional feature space, while having a fewer number of training samples. Examples include web-scale document classification where n-grams based representations can have feature dimensionalities in millions, recommendation systems with millions of items being recommended to millions of users, gene-expression analysis where gene expression data on patients is used to discover genetic bases of diseases.
To deal with such high feature dimensionality, one often needs to impose structural constraints on the learning models being estimated from the data. As we will see, such constraints often turn out to be non-convex and the problem becomes a non-convex optimization problem.

1.2 *Sparse Regression*

We will motivate the sparse regression problem using the example of gene expression analysis.

With the availability of DNA micro-array gene expression data, it is becoming possible to discover genetic explanations for a wide range of physiological traits, particularly diseases. In this problem setting, a data sample consists of expression levels of a large number p of genes encoded as a real vector $\mathbf{x}_i \in \mathbb{R}^p$, and the corresponding phenotypical trait $y_i \in \mathbb{R}$, and the total data consists of n data samples from from n human subjects/patients in the study, i.e., $\{\mathbf{x}_i, y_i\}_{i=1,\ldots,n}$.

For simplicity, we assume a linear model to predict phenotypical response from gene expression levels, i.e. $y_i = \mathbf{x}_i^T \mathbf{w}^* + \eta_i$, where $\mathbf{w}^* \in \mathbb{R}^p$ is the underlying linear model and η_i is some noise. A popular approach is to formulate this problem as a linear regression problem using the least squares formulation:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w} \in \mathbb{R}^p} \sum_{i=1}^{N} (y_i - \mathbf{x}_i^T \mathbf{w})^2$$

The standard linear regression methods fail to solve this problem in cases where, either
(a) the number of features (ie. size of parameter $\mathbf{w}$) is much larger than the available data samples i.e. $n \ll p$, or
(b) if we expect only a few of the features to be actually relevant to the problem.

In the gene expression problem, the number of genes whose expressions levels are recorded are typically very large, while the number of samples (human subjects) are not typically as large, i.e. $n \ll p$. Secondly, we do not expect all the genes being recorded to influence the given phenotype. In fact, a major objective of such studies is to identify the small set of genes which most significantly influence the given phenotype. This implies the underlying vector $\mathbf{w}$ is very sparse and leads us to define sparse regression problem.

In a sparse regression problem, we assume the underlying model is sparse, i.e. a vector having no more than s non-zero entries (called as s-sparse vector). The least formulation with this assumption is modified into a sparse recovery problem, defined below:

$$\begin{aligned}\hat{\mathbf{w}_{sp}} &= \underset{\mathbf{w}\in\mathbb{R}^p}{\arg\min} \sum_{i=1}^{N} (y_i - \mathbf{x}_i^T \mathbf{w})^2 \\ \text{s.t. } & \|\mathbf{w}\|_0 \le s\end{aligned}$$

where $\|\mathbf{w}\|_0$ represents the L_0 norm of $\mathbf{w}$, which is equivalent to the number of non-zero entries in the vector. Although the objective function in the above formulation is convex, the constraint $\|\mathbf{w}\|_0 \le s$ is a non-convex constraint set. This can be observed from the following example where a convex combination of two 1-sparse vectors in set $\|\mathbf{w}\|_0 \le 1$ produces a 2-sparse vector which lies outside the set.

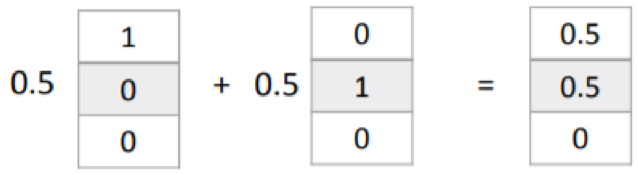


Figure 1: Example showing $\|\mathbf{w}\|_0 \le s$ is a non-convex constrained set.
A convex combination (average) of two 1-sparse vectors in set $\|\mathbf{w}\|_0 \le 1$ produces a 2-sparse vector which lies outside the set.

Sparse recovery handles the twin problems of identifying the relevant features and countering data-starvation since typically only $n \ge s\log(p)$ data samples are required for sparse recovery to work as opposed to $n \ge p$ in linear regression. However, in general, sparse recovery is an NP-hard problem and we will explore some settings where the nice structure allows us to have optimization algorithms which work in polynomial time in the next chapter.

1.3 ***Low-Rank Matrix Recovery***

Let us now discuss about the problem of low-rank matrix recovery, which finds applications in recommendation systems. Recommendation systems are popularly used to model the preference of users and make good estimates of how each user likes each item (say a song) or would benefit from it (say a drug). However, users typically rate only a handful of the millions of songs, and it is not feasible to administer every drug to a user. Thus, for a vast majority of user-item pairs, we have no direct information.

We can visualize this problem as a matrix completion problem: for a set of m users $u_1, u_2, ..., u_m$ and n items $a_1, a_2, ..., a_n$, we have a m x n preference matrix $\mathbf{A} = \mathbf{A}_{ij}$, where $\mathbf{A}_{ij}$ encodes the preference of i^{th} user for the j^{th} item. Now, we have only k entries of this matrix available (with k $\ll$ m x n entries of $\mathbf{A}$), corresponding to the available user-item pair data. For recommendation, we need to recover all the remaining entries i.e. complete the matrix.
If there is no structure in the matrix, and by extension, in the way users rate items, there would be no relation between the unobserved entities and the observed ones and hence there would be no unique way to complete the matrix. Therefore, it becomes essential to impose structure on the matrix and a typical assumption is for the matrix to have a low rank. This is equivalent to assuming that there is an r-dimensional vector $\mathbf{u}_i$ denoting i^{th} user and an r-dimensional vector $\mathbf{a}_j$ denoting j^{th} such that $\mathbf{A}_{ij} \approx \langle \mathbf{u}_i, \mathbf{a}_j \rangle$.

If we denote $\Omega \subset [m]$ x $[n]$ as the set of observed entries of $\mathbf{A}$, then low-rank matrix recovery problem can be formulated as:

$$\hat{\mathbf{A}_{lr}} = \underset{\mathbf{X} \in \mathbb{R}^{mxn}}{\arg\min} \sum_{(i,j)\in\Omega} (\mathbf{X}_{ij} - \mathbf{A}_{ij})^2$$
$$\text{s.t. } rank(\mathbf{X}) \leq r$$

This formulation also has a convex objective function, but a non-convex rank constraint. In general, matrix recovery problem is NP-hard. However, if the matrix has nice structures (such as Restricted Isometry Property), we can get polynomial time optimal algorithms. We will discuss one such algorithm (SVP) in the next chapter.

1.4 Convex Relaxation Approach

Due to the challenge of non-convexity and the associated NP-hardness, the most popular approach in literature is of convex relaxation. The relaxation approach modifies the problem itself by *relaxing* the non-convexity in objective function or the constraint, so it becomes a convex optimization problem.

In case of sparse linear regression, the relaxation approach relaxes the constraint set by changing the constraint to use L1 norm instead of L0 norm, i.e.

$$\begin{aligned}\hat{\mathbf{w}}_{relaxed} = &\arg\min_{\mathbf{w}\in\mathbb{R}^p} \sum_{i=1}^{N} (y_i - \mathbf{x}_i^T \mathbf{w})^2 \\ &\text{s.t. } \|\mathbf{w}\|_1 \leq s\end{aligned}$$

or by using its regularization based version:

$$\hat{\mathbf{w}}_{regularized} = \arg\min_{\mathbf{w}\in\mathbb{R}^p} \sum_{i=1}^{N} (y_i - \mathbf{x}_i^T \mathbf{w})^2 + \lambda \|\mathbf{w}\|_1$$

The above relaxations convert the problem to the popular LASSO formulations which are convex and have polynomial time solutions. Now, in general, such problems can change the problem drastically, and solutions to the relaxed formulations can be poor solutions to the original problem. However, it has been shown that if the problem possesses certain nice structure, then under careful relaxation, the solutions to relaxed problems are optimal for the original non-convex problems as well.

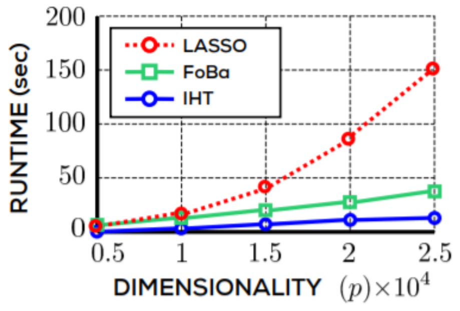
1.5 ***Non-Convex Optimization Approach***

Although the relaxed convex optimization problems are solvable in polynomial time, it is often challenging to solve them efficiently for large-scale problems. The non-convex optimization approach is to not relax the non-convex problem and solve them directly.

The most common techniques used to solve non-convex optimization problems include simple and efficient algorithms like projected gradient descent, alternating minimization, expectation maximization algorithm, stochastic optimization and their variants.

In recent literature it has been shown that if the problem or the data posses nice structure (such as satisfying Restricted Isometry Property), non-convex optimization approaches avoid NP-hardness and provide polynomial-time solutions which lead to global minima. Also, interestingly, it turns out that the problem structures that allow non-convex approaches to avoid NP-hardness are similar to those that allow their convex relaxation counterparts to avoid large relaxation gap. And in fact, many real world problems possess such nice structures leading to both convex-relaxation and non-convex techniques to work.

It has been shown in recent works, that non-convex approaches outperform relaxation based approaches in terms of speed and scalability making them more scalable approaches. The below figure shows an empirical comparison of non-convex and relaxation approaches for sparse recovery problem.



(a) Sparse Recovery (§ 7)

Figure 2: An empirical comparison of run-times by various approaches for sparse recovery problem. LASSO is the convex-relaxation approach, while FoBa (Forward-Backward algorithm) and IHT (Iterative Hard Thresholding) are non-convex approaches. Jain and Kar [1]

2 NON-CONVEX TECHNIQUES

In this chapter, we discuss the most popular technique in non-convex optimization i.e. Projected Gradient Descent. We discuss the fundamentals of this approach and how it is used to solve the problems of sparse recovery and low-rank matrix recovery and discuss the corresponding algorithms - Iterative Hard Thresholding (IHT) for sparse recovery and Singular Value Projection (SVP) for low-rank matrix recovery.

Before diving further into these algorithms, let us first discuss some of the ideas that come from convex function analysis. For most cases, unless explicitly stated otherwise, we will assume that functions are continuously differentiable.

Convex Combination - A convex combination of a set of n vectors $\mathbf{x_i} \in \mathbb{R}^p$, $i = 1,...,n$ in an arbitrary real space is a vector $\mathbf{x}_\theta := \sum_{i=1}^{n} \theta_i \mathbf{x_i}$ where $\theta = (\theta_1, \theta_2, ..., \theta_n)$, $\theta_i \geq 0$ and $\sum_{i=1}^{n} \theta_i = 1$.

Convex Set - A set $\mathbf{C} \in \mathbb{R}^p$ is considered convex if, for every $\mathbf{x}, \mathbf{y} \in \mathbf{C}$ and $\lambda \in [0,1]$, we have $(1-\lambda)\mathbf{x} + \lambda\mathbf{y} \in \mathbf{C}$ as well.

Convex Function - A continuously differentiable function $f : \mathbb{R}^p \to \mathbb{R}$ is considered convex if for every $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$ we have $f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle$, where $\nabla f(\mathbf{x})$ is the gradient of f at $\mathbf{x}$.

Now that we know some basic definitions in our convex function analysis, we now introduce the concept of projections first for convex settings and then extend it to non-convex problems.

2.1 *Convex Projections*

The concept of projection plays an important role in the projected gradient descent technique both for convex as well as for non-convex problems. Given any closed set $\mathbf{C} \in \mathbb{R}^p$, the projection operator $\pi_{\mathbf{C}}(.)$ is defined as

$$\pi_{\mathbf{C}}(\mathbf{z}) := \underset{\mathbf{x} \in \mathbf{C}}{\arg\min} \|\mathbf{x} - \mathbf{z}\|_2$$

In general, we may use any L^p-norm to define the above projection operator, but we see that the L^2-norm is the most commonly used. If $\mathbf{C}$

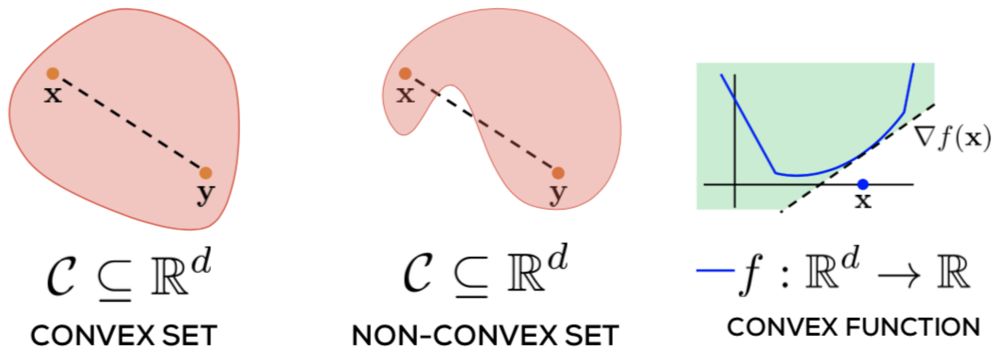
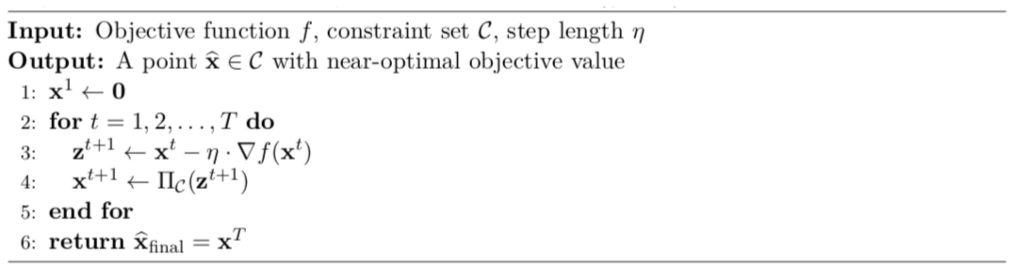

x
y
$\mathcal{C} \subseteq \mathbb{R}^d$
CONVEX SET
x
y
$\mathcal{C} \subseteq \mathbb{R}^d$
NON-CONVEX SET
$\nabla f(\mathbf{x})$
x
$f : \mathbb{R}^d \to \mathbb{R}$
CONVEX FUNCTION

Figure 3: **Left:**Convex sets. **Middle:**Non-convex sets. **Right:**Convex functions.

is a convex set, then the above problem reduces to a convex optimization problem.

2.2 Projected Gradient Descent (PGD)

The projected gradient descent algorithm is an extremely simple and efficient technique that extends the vanilla gradient descent method by defining projection after every gradient update step. This technique effortlessly scales well to large problems and can be applied to non-convex problems directly. The figure below explains the algorithm in brief.



```
Input: Objective function $f$, constraint set $\mathcal{C}$, step length $\eta$
Output: A point $\widehat{\mathbf{x}} \in \mathcal{C}$ with near-optimal objective value
1: $\mathbf{x}^1 \leftarrow \mathbf{0}$
2: for $t = 1, 2, \ldots, T$ do
3:     $\mathbf{z}^{t+1} \leftarrow \mathbf{x}^t - \eta \cdot \nabla f(\mathbf{x}^t)$
4:     $\mathbf{x}^{t+1} \leftarrow \Pi_{\mathcal{C}}(\mathbf{z}^{t+1})$
5: end for
6: return $\widehat{\mathbf{x}}_{\text{final}} = \mathbf{x}^T$
```

Figure 4: Projected Gradient Descent (PGD) algorithm

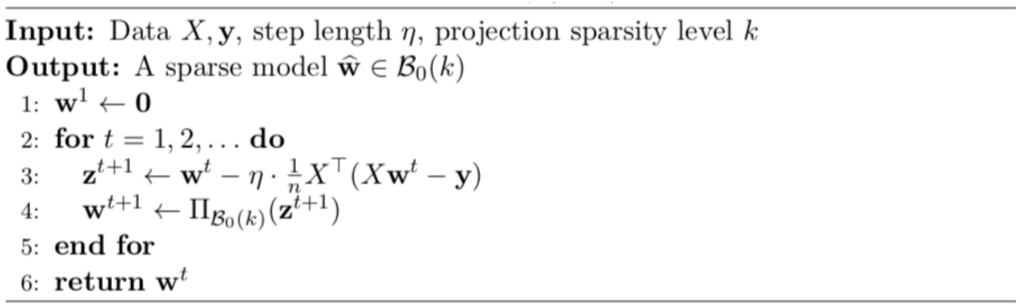
Now let us study projections for non-convex settings. The above definition of projection is easily transferable and can be defined over any type of set **C**. Depending on the non-convex set, the projection can be taken accordingly. We will see how it is undefined for non-convex sets in case of sparse recovery and low-rank matrix recovery problems below.

2.3 *Sparse Recovery via Projected Gradient Descent*

For a sparse recovery setting, the problem breaks down to applying projected gradient descent algorithm that requires projections onto the set of s-sparse vectors. Also, for a $\mathbb{R}^n$ dimensional space, the problem of projected into a space of s-sparse vectors involves sorting the coordinates of the vector $\mathbf{z}$ according to magnitude and setting all but the top-s coordinates to zero. We thus project in this way onto the non-convex constraint set $\mathbf{C}$ such that the vectors are s-sparse. A interesting thing to note is that our objective function is convex whereas the constraint set here makes the problem non-convex.

$$f(\mathbf{w}) = \|\mathbf{y} - X\mathbf{w}\|_2^2$$

This formulation would tempt the readers to adapt the projected gradient descent algorithm as studied above. Indeed a variant of the above algorithm called as the Iterative Hard Thresholding (IHT) algorithm is used to solve such a system. The figure below discusses the algorithm in brief.



```
Input: Data $X, \mathbf{y}$, step length $\eta$, projection sparsity level $k$
Output: A sparse model $\widehat{\mathbf{w}} \in \mathcal{B}_0(k)$
$\mathbf{w}^1 \leftarrow \mathbf{0}$
for $t = 1, 2, \ldots$ do
    $\mathbf{z}^{t+1} \leftarrow \mathbf{w}^t - \eta \cdot \frac{1}{n} X^\top (X\mathbf{w}^t - \mathbf{y})$
    $\mathbf{w}^{t+1} \leftarrow \Pi_{\mathcal{B}_0(k)}(\mathbf{z}^{t+1})$
end for
return $\mathbf{w}^t$
```

Figure 5: Iterative Hard Thresholding algorithm for Sparse Recovery

This algorithm is extremely simple to implement as well as extremely fast in execution, given that only gradient and projection steps are required.

Another class of algorithms that are famous for sparse recovery applications are the pursuit-style algorithms where we iteratively discover support elements. This family of algorithms includes Orthogonal Matching Pursuit (OMP) [Tropp and Gilbert, 2007], Orthogonal Matching Pursuit with Replacement (OMPR) [Jain et al., 2011], Compressive Sampling Matching Pursuit (CoSaMP) [Needell and Tropp, 2008], and the Forward-backward (FoBa) algorithm [Zhang, 2011].

Pursuit-style methods work by gradually looking for the elements in the support of the true model vector $\mathbf{w}^*$. At every time-step, a new support element is added to a set that is initially which is then used to solve a least-squares problem. A common practice is to add the coordinate where the gradient of the objective function is highest in magnitude among coordinates not already in the support. These algorithms are applicable wherever the structure of the constraint set can be represented as a combination of a small number of subsets of constraint set (atoms).

2.4 Low-Rank Matrix Recovery via Projected Gradient Descent

For matrix recovery problem that finds applications in recommendation systems, the projection involves computing the low-rank matrix $\mathbf{X}$ that is defined by the following rule,

$$\pi_{\mathbf{C}}(\mathbf{A}) := \arg\min_{\mathbf{X}\in\mathbf{C}} \|\mathbf{A} - \mathbf{X}\|_F$$

This projection can be computed using the Singular Value Decomposition on the matrix $\mathbf{A}$ and retaining the top r singular values and their corresponding vectors.

The above algorithm thus helps us to recover a low-rank matrix $\mathbf{X}$ given by the following optimization rule,

$$\min \frac{1}{2} \|\mathcal{A}(\mathbf{X}) - \mathbf{y}\|_2^2$$
$$\text{s.t. } rank(\mathbf{X}) \leq r$$

Applying the PGD algorithm to the above formulation, gives us the Singular Value Projection (SVP) algorithm that involves finding projections onto the set of low rank matrices which as seen before can be easily found by computing the singular value decomposition of the iterates. The figure below explains the algorithm in brief.

```
Input: Linear map $\mathcal{A}$, measurements $\mathbf{y}$, target rank $q$, step length $\eta$
Output: A matrix $\widehat{X}$ with rank at most $q$
1: $X^1 \leftarrow \mathbf{0}^{m \times n}$
2: for $t = 1, 2, \dots$ do
3:     $Y^{t+1} \leftarrow X^t - \eta \cdot \mathcal{A}^\top(\mathcal{A}(X^t) - \mathbf{y})$
4:     Compute top $q$ singular vectors/values of $Y^{t+1}$: $U_q^t, \Sigma_q^t, V_q^t$
5:     $X^{t+1} \leftarrow U_q^t \Sigma_q^t (V_q^t)^\top$
6: end for
7: return $X^t$
```

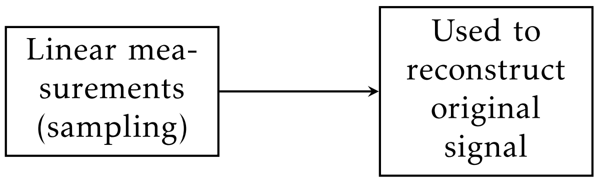
Figure 6: Singular Value Projection algorithm for Low-rank Matrix Recovery.

This algorithm too offers ease of implementation and speed similar to the IHT algorithm for sparse recovery.

One key takeaway from this discussion is that the PGD algorithm actually does not require the objective function to be convex over the entire $\mathbb{R}^p$ but requires a careful analysis of the structure of the objective function and constraint set to get the optimal solution in polynomial time.

3 SPARSE SIGNAL TRANSMISSION AND RECOVERY

As motivated in prior sections, the basic task of sparse recovery algorithms is to deduce an estimate for $\mathbf{w}^*$, in the equation $y_i = \mathbf{x_i^T} \mathbf{w}^* + \eta_i$. Here, we aim to recover an estimate of $\mathbf{w}^*$ which is sparse, which cannot be guaranteed through the use of traditional sparse regression approaches. This mathematical apparatus, turns out to be a clean way to model both the transmission and recovery of sparse signals. Note, that here we define a sparse signal in the exact same way as before, in that the L_0 norm (non-zero components in basis representation of the signal), is minimized. Natural signals are an excellent example of sparsity, and this model more broadly finds its application where linear measurements samples are used to reconstruct an approximation of the original signal.


Linear measurements (sampling)
Used to reconstruct original signal

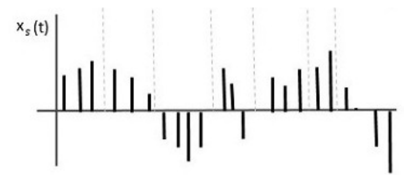


Figure 7: Sampled signal

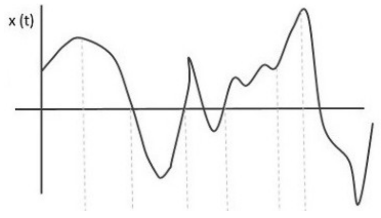


Figure 8: Recovered signal

In contrast to other sparse recovery problems, for sparse signal recovery, we must come up with both a design matrix $\mathbf{X}$ and the recovery algorithm $\mathbf{A} : \mathbb{R}^n \times \mathbb{R}^{nxp} \rightarrow \mathbb{R}^p$. Compare this to gene expression analysis, where we don't have control over the design matrix $\mathbf{X}$, hence we are restricted to only designing the recovery algorithm. Hence, for the task of recovering sparse signals, we may make strict assumptions about the design matrix (which lead to nicer properties and subsequent tractability). Note, that natural data may not satisfy many of the simplifying assumptions that we make, which leads to us having to relax a subset of these assumptions.

3.1 *IHT for signal recovery*

The IHT algorithm which was introduced in an earlier section may be used for the sparse recovery of signals. It has the advantages of being:

1. Simple to implement (also more understandable)
2. Extremely fast in practice

However, it has unclear recovery guarantees, due to the problem itself being NP-hard. It turns out that we can solve this problem efficiently if **X** has special structural properties, namely being isometric. An **isometry** is any transformation that maps elements to the same (or another metric space), such that the distance between the image elements in the new metric space is equal to the distance between the elements in the original metric space. Even if our design matrix **X** does not possess global isometry, but has the restricted isometry property (RIP), which preserves the space of sparse vectors, we can recover the signal.

3.2 *Basis Pursuit Family of Algorithms*

3.2.1 *Basis Pursuit (BP)*

Ideal sparse reconstruction minimizes $||x||_0$ while being consistent with $Ax = y$. However, since the problem is intractable, we typically make a convex relaxation of this problem in practice, and instead solve the following problem:

$$\begin{aligned} &\min \, ||x||_1 \\ &\quad \text{s.t } \mathbf{Ax} = \mathbf{y} \end{aligned}$$

3.2.2 *Basis Pursuit with epsilon noise (BP_ϵ)*

$$\begin{aligned} &\min \, ||x||_1 \\ &\quad \text{s.t } ||\mathbf{y} - \mathbf{Ax}||_2 \leq \epsilon \end{aligned}$$

3.2.3 *Quadratic Programming (QP_λ)*

$$\min \, ||\mathbf{y} - \mathbf{Ax}||_2 + \lambda ||\mathbf{x}||_1$$

In practice, the measured signal has noise, hence we formulate the problem as either basis pursuit with epsilon noise or as quadratic programming. Here, we would tune the parameters λ, ϵ based on the signal-noise ratio (SNR).

3.3 Matching pursuit algorithm

A more commonly used technique is the matching pursuit algorithm, which is the fundamental tool behind a large variant of algorithms used for sparse signal recovery.

```
Input: Signal $f(t)$, dictionary $D$ with normalized columns $g_i$
Output: List of co-efficients $a_n$ and indices for corresponding
1 Initialization
  $R_1 \leftarrow f(t)$
  $n \leftarrow 1$
2 Repeat $g_{\gamma n} \in D$ with max $| < R_n, g_{\gamma n} > |$
  $a_n \leftarrow | < R_n, g_{\gamma n} > |$
  $R_{n+1} \leftarrow R_n - a_n g_{\gamma n}$
  $n \leftarrow n + 1$
3 Until stop condition $||R||_n < thresh$
4 Return
```

Algorithm 1: Matching Pursuit

Properties

- Algorithm converges i.e. the residual term $R_n \leftarrow 0$, for any signal f that is in the space spanned by the dictionary D.
- Error $||R||_n$ decreases monotonically
- As of each step the residual is orthogonal to the selected filter.
- For the condition where the vectors in D are orthonormal, the algorithm is a form of PCA.

Applications

- Signal, image, and video coding
- Shape representation and recognition

Note: The main drawback (or bottleneck) to this algorithm is that a large dictionary D has to be searched at each iteration.

3.4 ***Orthogonal matching pursuit***

This is a common variant of the matching pursuit algorithm in which all the coefficients extracted so far are updated by computing the orthogonal projection of the signal onto the subspace spanned by the set of atoms so far. This leads to nicer results (with the given overhead of increased computation).

4 SIGNAL RECOVERY FOR RADIO ASTRONOMY

Radio astronomy is a subfield within astronomy which studies celestial objects in the radio frequency band of the electromagnetic spectrum. The signals are captured through large radio antennas (telescopes), which are then correlated using techniques from aperture synthesis or interferometry. We picked up a sparse recovery application in radio astronomy and implemented a state-of-the-art algorithm (Högboms CLEAN) for recovering cleaned images from the noisy spatial images.

4.1 Högboms CLEAN Algorithm

The CLEAN algorithm is the precursor to a wide range of algorithms which are designed to recover the original 'true' image from noisy measurements. The mathematical foundation of the algorithm is the matching pursuit technique for sparse signal recovery which we previously introduced. The CLEAN algorithm assumes that the original image consists of a number of point sources (which are commonly assumed to be Gaussian in nature). We can think of the dictionary D (see OMP explanation for context), consisting of atoms which are point sources of varying parameters (which would be differing variance in the case of gaussians). The algorithm will iteratively find the highest intensity source in the image, and subtract a smaller gain of this source convolved with the point spread function (PSF), until the highest value is lesser than a threshold (residual). For further context, we invite the reader to refer [2].

An illustration of this is shown below:

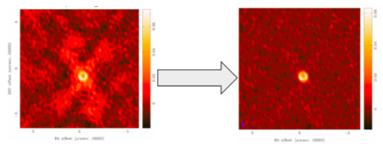


Figure 9: Conversion from noisy image to CLEAN image

We now list interesting points we came across in the context of OMP and CLEAN:

- Though, theoretically the orthogonal matching pursuit (OMP) algorithm has a larger overhead, in practice it is preferred to plain MP, due to its' observed faster convergence property.

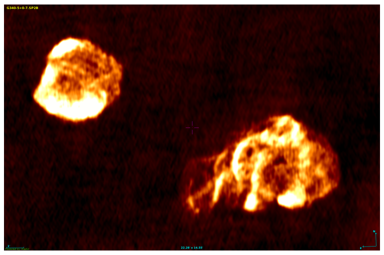


Figure 10: Supernova remnants at the centre of our galaxy, data collected at 325 MHz

- OMP is typically faster than basis pursuit (BP) and simpler to code.
- BP is convex, hence will converge to a single global optimum (with respect to the convex relaxation of the original l_0 norm)
- Högboms CLEAN is identical to matching pursuit (MP), but forms the residual in image space instead of measurement space.
- Discovered in the early '70s, the CLEAN algorithm was originally intended for radio astronomy (where it is now the dominant tool for deconvolution of images), but has now also found use in processing MRI scans.

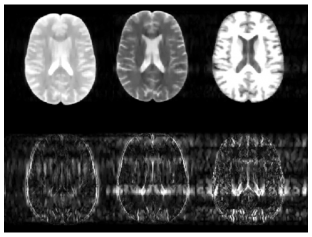


Figure 11: MRI Scans

5 REFERENCES

[1] Jain, P. and Kar, P. (2017). Non-convex optimization for machine learning. *Foundations and Trends in Machine Learning*, 10(3-4):142–336.

[2] Thompson, A. R., Moran, J. M., and Swenson, G. W. (1998). *Interferometry and synthesis in radio astronomy*. Krieger Pub.