Greedy Conical Hull Algorithms and their Applications

November 24, 2017

ACKNOWLEDGEMENTS

I wish to express my sincere thanks to my guide, **Prof. Rahul Garg**, Department of Computer Science and Engineering, IIT Delhi for his valuable support through this work. I would also like to thank my project supervisor, **Prof. B. Sivaselvan** for his constant support and valuable suggestions.

Vignesh Sairaj
EDM14B039
IIITDM

Contents

List of Figures

Abstract

In this study, we investigate the application of greedy variants of existing conical hull algorithms for the near-separable nonnegative matrix factorization (NMF) problem to functional Magnetic Resonance Imaging and evaluate the relative performance of the variants and some of the existing methods on synthetic data. The synthetic data has been generated on a number of different test parameters. We also investigate the potential application of the technique in music audio source separation.

Chapter 1

Motivation

1.1 Functional Magnetic Resonance Imaging

Functional Magnetic Resonance Imaging (fMRI) is a noninvasive imaging technique used to detect activations in the brain. It uses the blood flow in a region as an indirect measure of the neural activity in the region. The most commonly used technique is Blood-oxygen-level dependent contrast imaging (BOLD contrast imaging) which detects the change in magnetic properties of the blood with the level of oxygenation. The observed data is thus a series of 3D images composed of intensities mapped to uniformly spaced cubes called voxels that form a lattice. The images are sampled at uniform time intervals, which when stacked over one another form a 4D array. This can be interpreted as a 1D time-series sampled at the area corresponding to each voxel.

The goal of fMRI data analysis is to discover relationships between the activations in the brain and the stimulus/task that the subject is exposed to/performs. Another goal is to map activations to various cognitive states such as memory and recognition.

1.2 General Linear Model

Correlations with the stimulus can be detected using the General Linear Model (GLM) which regresses the BOLD signal of each voxel as a linear

function of an expected signal (which is computed by convolving the input stimulus with an empirically obtained Haemodynamic Response function (HRF)). Depending on the sign and magnitude of the coefficients used to express the BOLD signal of a voxel, the voxel can be identified as being correlated with the stimulus (i.e., task-positive (TP)), anticorrelated (task-negative (TN)) or uncorrelated.

The limitations of the model are that it assumes the BOLD response to be a linear time-invariant (LTI) function of the input stimulus. Also, the application of this technique requires that the stimulus is presented as a repetitive block design. Thus GLM cannot be used to study complex natural tasks such as watching a movie. The HRF too may vary across subjects or between regions.

1.3 Inter Subject Correlation

Inter-Subject Correlation (ISC) is a method to identify regions in the brain that are activated as a response to a given stimulus. It is based on the assumption that corresponding regions in the brain resond to the same stimulus similarly across subjects. This analysis is model-independent and does not require prior knowledge of the exact nature of the input stimulus. The pairwise correlation of the time-series of corresponding voxels are averaged across subjects and thresholded to identify those voxels that are highly correlated scross subjects.

Although ISC can identify large regions in the brain that correspond to activations in resonse to the task, it cannot identify components within the set of highly correlated voxels in a way that can be easily interpreted in relation to the stimulus; it cannot even distinguish between TP and TN voxels.

1.4 Nonnegative Matrix Factorization

Nonnegative Matrix Factorization (NMF) is a linear dimensionality reduction technique to approximate a matrix $M_{m\times n}$ as a product of matrices of rank

at most r, where $r << min(m, n)$. Given $M_{m\times n}$ and target rank r, matrix factorization is the problem of finding $W_{m\times r}$ and $H_{r\times n}$, such that $M = WH$. Expressed another way, this is equivalent to taking weighted linear combinations of the columns of W with the corresponding columns of H as the weights for each column in the data matrix M. i.e., $M_{:j} = \sum_{k=1}^{r} W_{:r}H_{kj}$, where $M_{:j}$ denotes the j^{th} column of M. When the input matrix is guaranteed to be nonnegative, and the matrices W and H are also required to be nonnegative, the problem becomes nonnegative matrix factorization. NMF has many practical applications where it is preferred over other LDRs due to its purely additive constraints such as in hyperspectral imaging, text mining, and facial feature extraction. In these application, each column of the data matrix is expressed a linear mixture of features that are combined in a purely additive manner that conforms to inherent model. The decision version of the NMF problem (i.e., determining if a given matrix M supports an exact nonnegative r-factorization) was shown to be NP-Hard by [Vavasis, 2009] [11].

1.5 Separable NMF

A matrix $M_{m\times n}$ is said to be r-Separable if it permits a nonnegative factorization of the form $M = W[I_r, H']\Pi$, where I_r is the r-Identity matrix and Π an $n \times n$ permutation matrix, and H' is $r \times (n - r)$. This is equivalent to $M = M(:, \kappa)H$, where κ is an r-subset of the columns of M and $H_{r\times n} > 0$. A matrix M is said to be near-separable if it differs from a separable matrix by a noise matrix η for $\|\eta\| < \epsilon$, for some choice of norm $\|\cdot\|$and ϵ. When the columns are looked at as m-dimensional vectors, this is the same as assuming that the components of the NMF are the edges of the conical hull of all column vectors in the data matrix. Separable NMF was first shown to be tractable by [Arora, et al., 2012] [3]. [Donoho and Stodden, 2003] [6] also showed that separabilty implied that the factorization is unique, modulo scaling and permutation. The near separability assumption is satisfied in many natural applications. The separability condition can be interpreted as the pure-component assumption where it is assumed that there exists a column in the data matrix that comprises of exactly one component in the factorizarion for every component in W.

1.6 Separable NMF as a Second Stage to ISC

Nonnegative matrix factorization can be used to find the key components within the data that approximate each time-series response of the voxels as nonnegative linear combinations (or mixtures) of the principal components. The nonnegativity constraint has the added benefit of separating out anti-correlated components into different groups as they can not be expressed as nonnegative combinations of one another. This way, we can separate out the different components that together *explain* the time-series data of the voxels as linear mixtures. This technique is also model-independent as it does not assume the BOLD signals to follow a reference response function of the stimulus. On the other hand, the factorization can be used to create maps that correspond to the proportion of the contribution of each component in the mixture of components for each voxel. These components themselves are time-series signals of certain representative voxels in the brain.

1.7 Music Audio Source Separation

Audio signals are typically complex mixtures of different sound sources. The sound sources can be several people talking simultaneously in a room, different instruments playing together, or a speaker talking in the foreground with music being played in the background. The decomposition of a complex sound mixture into its constituent components is referred to as source separation. A popular source example of the separation problem is the cocktail party problem, where the objective is to separate the voice of a specific speaker from a mixture of conversations with multiple speakers and background noises. When decomposing a music signal, music-specific properties and additional musical knowledge can be exploited. The number of spectral signatures in music signals is usually fewer than other types of audio and lends itself effectively to NMF techniques. The matrix M is obtained by performing a Short-Time Fourier Transform (STFT) on the audio signal and mapping the intensities of each bin in the resulting spectogram into a component of a spectrum-vector. The spectrum vectors of each time interval are grouped togethr as column vectors to give M. W can be interpreted as the spectral signatures of a few fundamental notes/clusters of notes and H contains the temporal information for when each of these notes is "sounded". Constrained NMF techniques have been investigated for this purpose in [8],

[12], and [10]. We look to investigate the application of near-separable NMF algorithms for the same application.

Chapter 2

Existing Algorithms for Separable NMF

Here we mention some algorithms for the near-separable NMF problem.

AKGM This was the first algorithm (by Arora, et al. 2012 [3]) to solve the separable NMF problem which proved that the separable NMF problem is tractable. It uses n feasibility LP formulations to obtain a provably good factorization for near-separable matrices under a bound on the $(\infty, 1)$ norm of the noise. However, since each each LP involves $O(n)$ variables, it cannot be scaled up to large datasets.

Hottopixx Another LP-based method with provable guarantees that involves $O(n^2)$ variables[Bittorf et al. 2012 [4]]. Hottoppix was shown by [Chayan, et al. 2017 [5]] to perform poorly in noisy data due to its use of the $(\infty, 1)$ norm to define the simplex for the lp which is sensitive to outliers. Also, the value of ϵ (i.e., the upper bound for the $\|\cdot\|_{\infty,1}$ norm constraintin the LP formulation) has to be known beforehand.

LP formulation LP ALCD An LP-based method similar to hottopixx was proposed by [Chayan, et al. 2017 [5]]. This algorithm overcomes the limitations of hottoppix by moving the norm constraint into the objective function and using the $\|\cdot\|_{1,1}$ instead. The separability constraint, as in hottopixx, can be reformulated in terms of a *Factorization Localizing Matrix*

$C_{(n\times n)}$ as:

$$\begin{aligned} M = WH &= M[I_r, 0]\Pi H \\ &= MC \end{aligned}$$

The Factorization Localizing Matrix is obtained by solving the following LP:

$$\begin{aligned} &\min_{\forall C \in R_+^{n\times n}} \|M - MC\| \\ &\text{subject to: } c_{ij} \leq a_i \quad \forall i, j \\ &\sum_{i=1}^{n} a_i \leq r \end{aligned} \tag{2.1}$$

The columns selected by the algorithm, then, are those that correspond to the indices of the rows in C with the highest row-sum. This algorithm has been shown to be more robust to noise under certain conditions in [5], however solving the LP still takes a considerable amount of time on large datasets.

XRAY XRAY (Exterior Ray Projection Algorithm) [Kumar, et al., 2013 [9]] is a fast conical hull algorithm for separable NMF that finds the end-members that form the conical hull of the column vectors in the data matrix. It is quite fast and robust to noise. However, it can fail to identify the right components in the rare case that there are more than two columns that maximize the selection criterion. There are a number of variants with different but similar criteria to pick the next column to be included in the basis set. The remaining columns are then projected on to the conical hull of the basis columns to compute the residue ($M-$ projection of M on basis columns), which is then used to select the next column to be included, and so on.

SNPA The Successive Nonnegative Projection Algorithm [Gillis, 2014] [7] is a modified version of the earlier Successive Projection Algorithm (SPA), first proposed by [Araùjo, 2001, [2]] which selects a column that maximizes a strongly convex function on the residue and then projects all remaining columns to the orthogonal complement of the selected vector. SNPA, instead, projects the remaining vectors to the convex hull of the columns and the origin to compute the residue. While SPA requires the columns of W to be

Algorithm 1: XRAY [9]: Fast Conical Hull Algorithm

Input : A near r-separable matrix $M = \tilde{M} + \eta$, where $\tilde{M} = WH$, $W = \tilde{M}_{:\kappa}$ for some r-subset κ, W is an α-simplicial, and $\|\eta\|_{\infty,1} \leq \epsilon$, and a factorization rank r

Output: Matrix W and H such that $M(:,\kappa) approx W$ for some index set κ and $M \approx WH$

Initialize $R \leftarrow M, \kappa \leftarrow \{\}$.
while $|\kappa| < r$ **do**
  Detection Step: Find an extreme ray.
    $j^* = \arg\max_j \dfrac{R_{:i}^T M_{:j}}{p^T M_{:j}}$ for any $i : \|R_{:i}\|_2 > 0$
    where p is a strictly positive vector (not collinear with R_i)
  Selection Step: Pick an exterior point using one of the following criteria:
      rand: any random $i : \|R_{:i}\|_2 > 0$.
      max: $i = \arg\max_k \|R_k\|_2$.
      dist: $i = \arg\max_k \|(R_k^T M)_+\|_2$.
  ***Greedy variant for choosing extreme ray*:** $j^* = \arg\max_j \dfrac{\|(R^T M_{:j})_+\|_2^2}{\|M_{:j}\|_2^2}$.
  Update $\kappa \leftarrow \kappa \cup \{j^*\}$.
  Projection Step: Project onto the cone:-
      $H \leftarrow \arg\min_{B \geq 0} \|M - M_{:\kappa} B\|_2^2$.
  Update residuals: $R \leftarrow M - M_{:\kappa} H$.
end while

of full rank, SNPA does not, and SNPA was also shown to be more robust to noise and applicable to a wider class of problems.

Algorithm 2: SNPA[Gillis, 2014] [7]: Successive Non-negative Projection Algorithm

Input : A near r-separable matrix $M = \tilde{M} + \eta$, where $\tilde{M} = WH$, $W = \tilde{M}_{:\kappa}$ for some r-subset κ, and $\|\eta\|_{\infty,1} \leq \epsilon$, a factorization rank r, and a strongly convex function f satisfying certain assumptions (in this study, the L2 norm)

Output: A set of indices J such that $M_(: J) \approx W$ upto permutation

Let $R = M$, $J = \{\}$, j=1.
while $R \neq 0$ and $j \leq r$ **do**
    $k^* = \arg\max_j f(R_{:j})$
    $J = J \cup \{k^*\}$.
    $R_{:i} = M_{:i} - M_{:J} H^*_{:i}$ for all i
        where $H^*_{:i} = \arg\min_{h \in \Delta} f(M_{:i} - M_{:J} h)$
    $j = j + 1$
end while

Chapter 3

Greedy Variants

We consider three variants of the greedy heuristic proposed in XRAY (however, we project the remaining columns to the convex hull of the basis columns and the origin in SNPA and notto the conical hull of the basis columns as in XRAY). The heuristic is based on the greedy strategy of trying to have the maximum possible sum of components of the residues along the new edge (basis column) to be chosen. This is an indicator of the magnitude of the drop in the objective (L2 norm of residue) to be minimized. By projecting every column of the residue matrix along every candidate basis column, we try to find the best candidate column that covers as much of the residue as possible. This greedy stepwise optimal strategy, however, is not guaranteed to output the right columns, even in the noiseless case. For example, if 4 points in 3 dimensions in the planar region $x + y + z = 0; x, y, z \geq 0$ are the vertices of an equilateral triangle and its center, then the matrix of column vectors that correspond to the position vectors of these points clearly supports a 3-separable factorization (with the 3 vertex points (column vectors) chosen as the basis columns), as the center can be expressed as a convex combination of the three vertices. However, the best 1-rank factorization (1-point approximation) would consist of just the point in the center of the triangle as part of the W matrix. The greedy heuristic would pick this point (column vector) in the first step and even if the next points it picks are on the vertices, since it does not drop a column (point) once picked, it returns an incorrect solution when the example clearly supports an exact 3-separable factorization.

Naturally, to attempt to try and remedy this shortcoming, we look at the possibility of dropping columns. Since the first column chosen tends to be

near the *center* of the data, we could drop the chronologically first column and replace it with a new column picked using the same heuristic as before. In our preliminary analysis, the columns picked by this variant, which we call greedyRepSNPA (for replacement), were identical to the columns picked by SNPA in the noiseless case. However, even this proved to be futile.

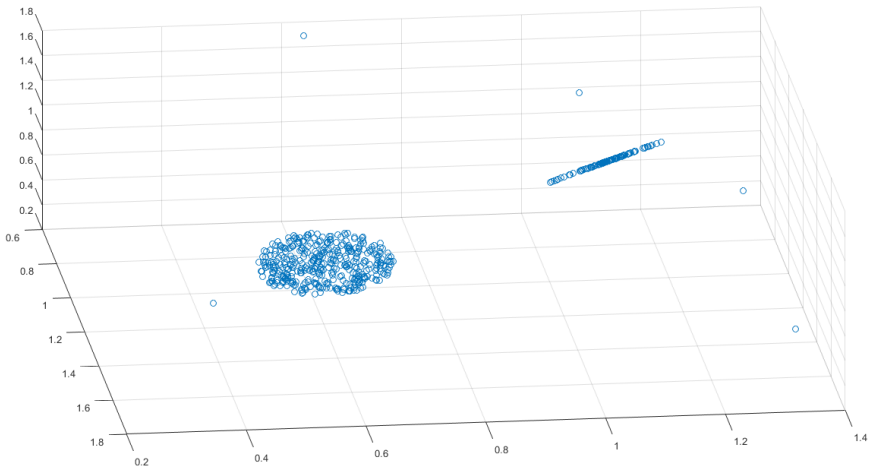

Figure 3.1: A noiseless example where greedyRepSNPA fails

In the example above, i.e., figure 3.2, the vertices of the pentagon in the planar region $x + y + z = 0; x, y, z \geq 0$ form the edges of the conical hull of the collection of points, however a sufficiently dense cluster of points near one of the vertices forces greedy to pick the first point within the cluster. The dense line (not as dense as the circular cluster) forces the next point to be picked as the endpoint of the line making the first *two* points picked by the greedy heuristic to be incorrect. Thus, we see that, even with replace, greedy cannot guarantee to obtain the right columns even in the noiseless case.

What if we replaced the first two points instead of just the first? The counter example given above in Fig. 3.2 can be easily extended and modified to deal with any number of replacemets, including an entire second pass, replacing every column in the basis. Though this algorithm (greedy2passSNPA) is not guaranteed to give the right columns, it is a good approximate solution

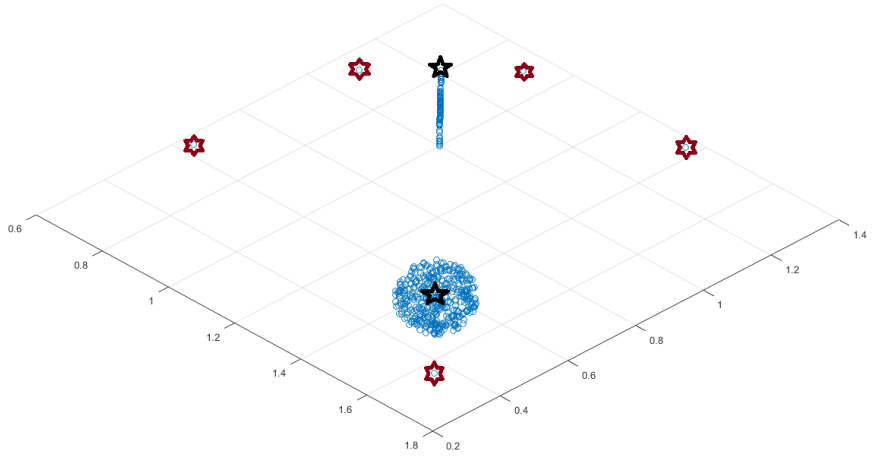

Figure 3.2: Rotated view of above image

most of the time.
When the data is sufficietly noisy, however, greedySNPA is comparable to the other algorithms even in column recovery and not just the residual.

Chapter 4

Other Strategies

We know that the stepwise-optimal strategy using the L2 norm doesn't work without dropping columns, so what if we could instead start with the complete convex hull and drop columns greedily instead? Could we achieve a factorization whose distance with the optimal is no more than some constant (Here, distance is measured in terms of the $L_{2,2}$ matrix norm)? Although this method is guaranteed to converge in the noiseless case. The answer, unfortunately, turns out to be no.

In figure 4.1 we see that convex hull of all the points are the vertices of the trapezium in light green. Now, the best 3-point factorization would include the two vertices in the bottom and the midpoint of the upper edge of the trapezium (dark green) that form a triangle enclosing all the internal points in red. Clearly, this is not achievable by dropping points from the overall convex hull. Also, the density of the points (in effect, the number of points) clustered close to the vertices (in red) of the triangle can be increased arbirariy to widen the gap between the output of the algorithm and the optimal factorization without any bounds that involve the $L_{2,2}$ norm. It may be possible to bound the error with an $L_{\infty,2}$ norm, however, which would be an interesting direction in the future.

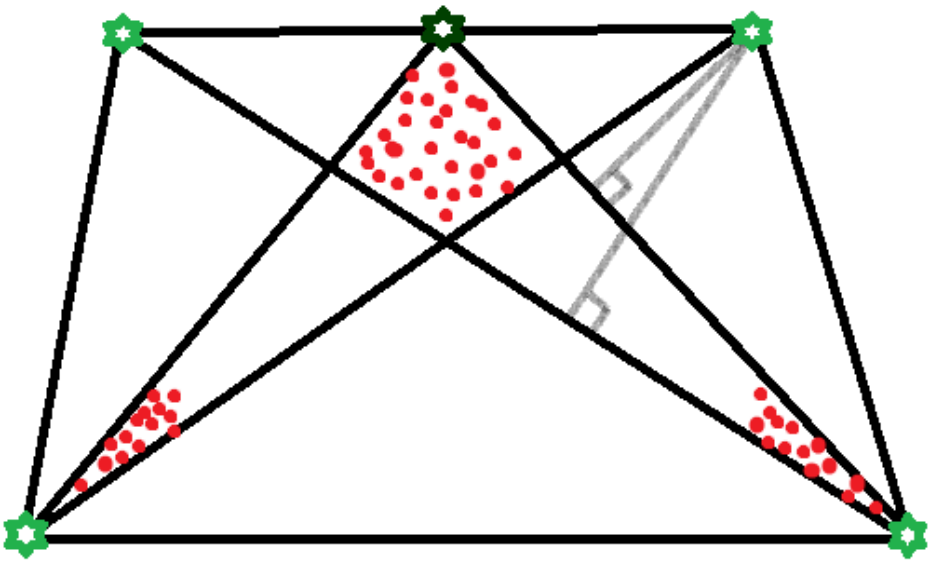

Figure 4.1: Counterexample for greedy dropping strategy

Chapter 5

Performance on Synthetic Data

We have compared the performance of six algorithms (LP ALCD, XRAY, SNPA, greedySNPA, greedyRepSNPA, and greedy2passSNPA) in this study and have compared the performance on different kinds of data:

1. Uniform distribution to generate W and H: number of columns, number of rows and basis are changed
2. Uniform Distribution, well conditioned and ill-conditioned to generate W, with H being drawn from a Dirichlet distribution or the rest of columns of the computed as a midpoint between every column in W.
3. Presence of duplicate columns of W (basis matrix) in the data matrix.
4. Uniform distribution: variation in sparsity of H.

Generation of W: The columns of W are drawn from a uniform distribution $U(0, 1)$. With just these columns, the data is well-conditioned. To generate data for the Ill-conditioned case, The SVD of the matrix obtained above is taken $W' = U\Sigma V^T$ and the diagonal matrix Σ is replaced with S where diagonal entries in S are of the form α^{i-1} for $i = 1, 2, ..., r$ such that $\alpha^{i-1} = 1000$. The condition number of this matrix is, thus, 1000. The negative entries in the new W are now made 0. The data is then L_1 (column) normalized.

Generation of H For the Dirichlet Distribution, a sample H' is drawn from a Dirichlet ditribution where the parameters are chosen uniformly in

$(0, 1)$. The actual H is then computed as $H = [I_r I_r H']$ such that the basis columns are repeated once in the data matrix.

In Middle-Points generation $H = [I_r H']$ and each column in H' is just all zeros except two of the rows that have 0.5 in each entry. So, all the columns in the data matrix except the basis columns are midpoints between a pair of basis columns. Thus there are $\binom{r}{2}$ columns in H'.

Noise For the basis, sample, dimension, and sparsity variations as well as for the Dirichlet distributions, the noise added is δ times η $N(0, 1)$ while for the Middle Points variation, the noise is $\eta_i = \delta(M_i - \bar{W})$.

Performance measures:

1. **Fraction of Columns extracted:-** Fraction of correctly extracted basis (columns of W).

$$\frac{\text{Number of correctly extracted columns of W}}{\text{Total number of columns of W (= r)}}$$

2. **Normalized Residual (L1):-**

$$\frac{\|M - M_{:J} H\|_{1,1}}{\|M\|_{1,1}} = \frac{\|M - WH\|_{1,1}}{\|M * scale\|_{1,1}}$$

3. **Normalized Residual (L2):-**

$$\frac{\|M * scale - M_{:J} H * scale\|_F}{\|M\|_F} = \frac{\|M * scale - WH * scale\|_F}{\|M * scale\|_F}$$

where $scale = diag(\frac{1}{sqrt(sumAlongColumns(M.^2))})$ so that $M * scale$ is l_2-normalized along the columns.

4. **Normalized Absolute Error:** $L_{1,1}$ norm of difference between original Factorization Localizing Matrix C and the one returned by the algorithm

$$\frac{\|C_{retrieved} - C_{original}\|_{1,1}}{\|C_{original}\|_{1,1}}$$

Performance measures for sparsity:

5. **Precision:** The precision measures the quality of the retrieved coefficients
$$\frac{|S_{retrieved} \cap S_{original}|}{|S_{retrieved}|}$$

6. **Recall:** The recall measures the quantity of retrieved coefficients
$$\frac{|S_{retrieved} \cap S_{original}|}{|S_{original}|}$$

where $S \in \{0,1\}^{n \times n}$ and $s_{ij} = 1, if c_{ij} \geq \alpha and 0 \, otherwise$.Here $|(\cdot)|$ is the number of zero-valued entries in $(\cdot)$ and intersection gives the number of common zeros between to sets.

Additional Tests:

- Points in the 2D plane projected onto plane $x + y + z = 1$ (3D). Simplicial points form vertices of a regular polygon, all other points are clustered around vertices of a smaller polygon of same number of sides, centered on the same point arranged in a staggered fashion (vertices face edges of larger polygon).
  `Variations`:
  - Number of sides
  - Radius of each cluster (relative to number of sides) (uniform random in a circular area)
  - Size of inner polygon (distance from the center, relative to size of outer polygon)
- Regular polytopes in $(n-1)$ dimensions projected onto hyperplane $(x_1+, ..., +x_n = 0)$ with vertices of hypercube forming the simplicial set and other points clustered around a smaller orthoplex (The dual of the hypercube) centered around the same point. (Variation in dimension only)
- Scattered simplicial points selected uniformly at random with clusters generated around points in the convex hull of the selected points with a few outliers selected uniformly at random.

Chapter 6

Results on Synthetic Data

6.1 Basis Variation

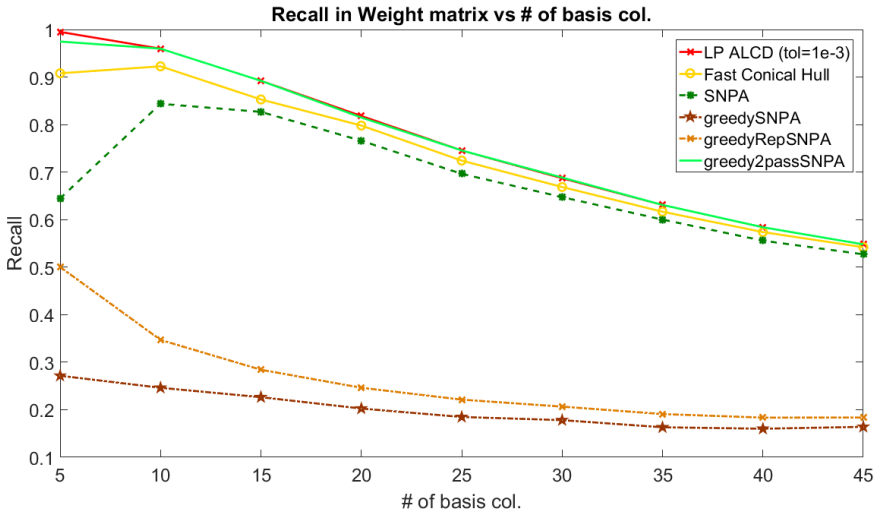

Figure 6.1: Basis Variation: Recall

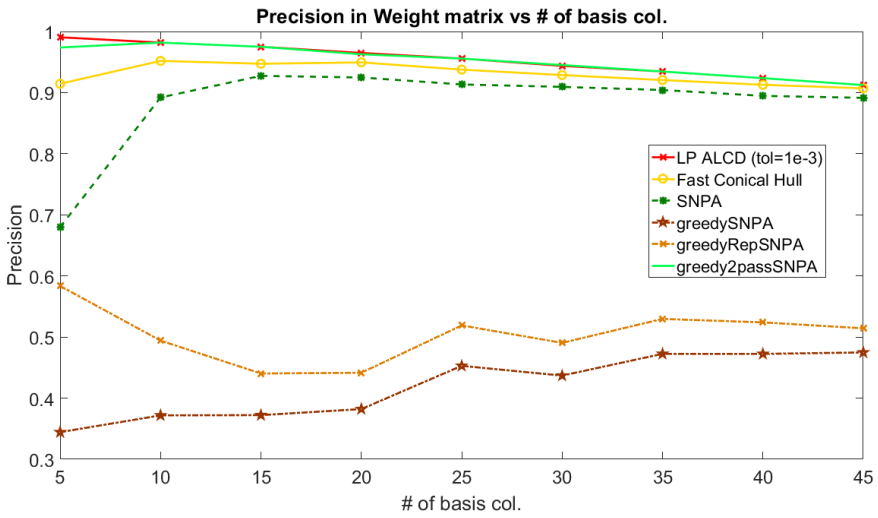

Figure 6.2: Basis Variation: Precision

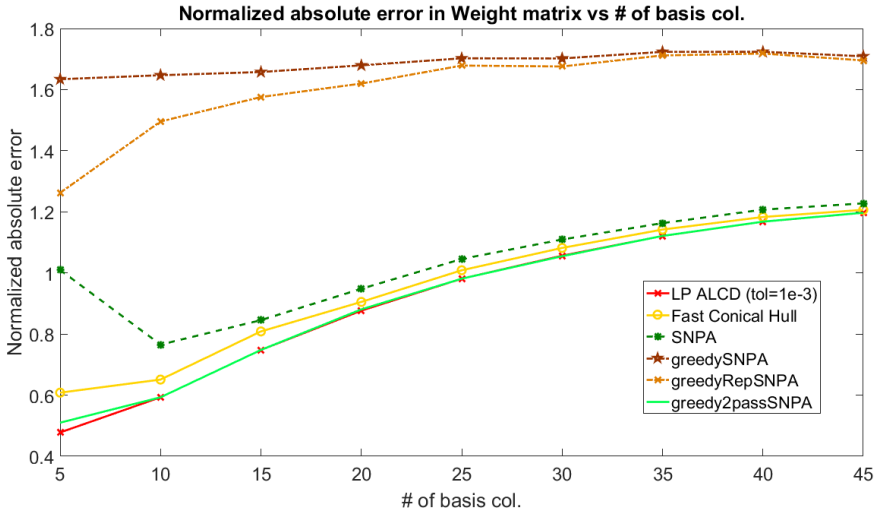

Figure 6.3: Basis Variation: Absolute Error

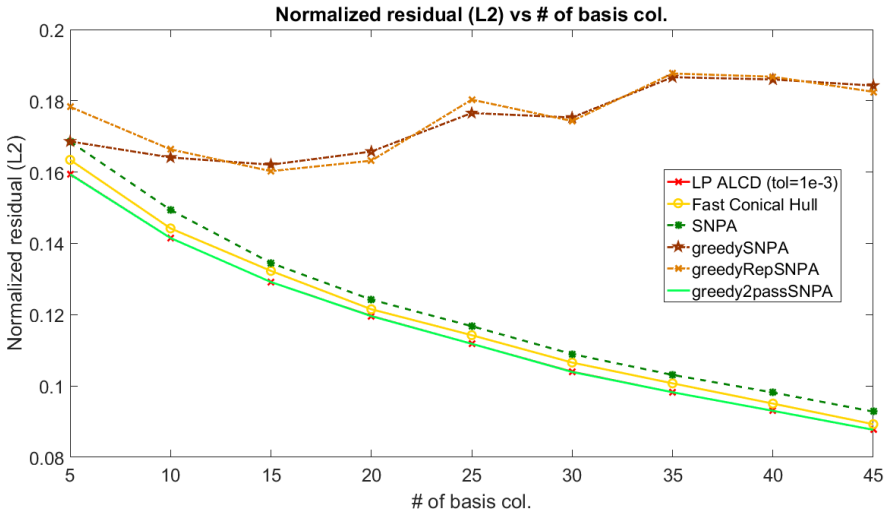

Figure 6.4: Basis Variation: L2 Residual

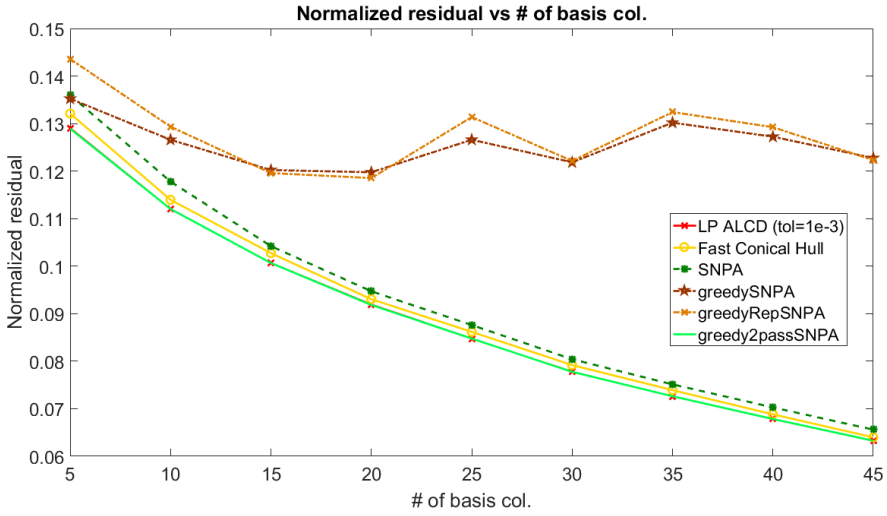

Figure 6.5: Basis Variation: L1 Residual

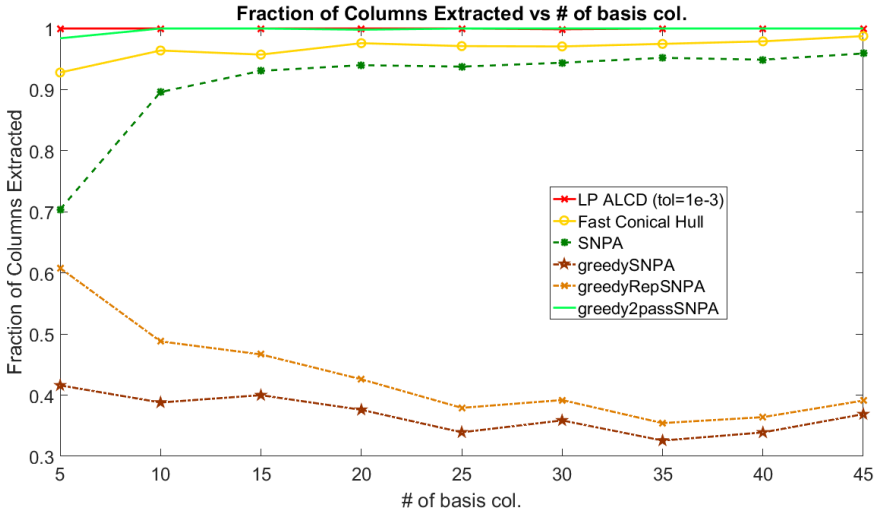

Figure 6.6: Basis Variation: Fraction of Columns

6.2 Dimension Variation

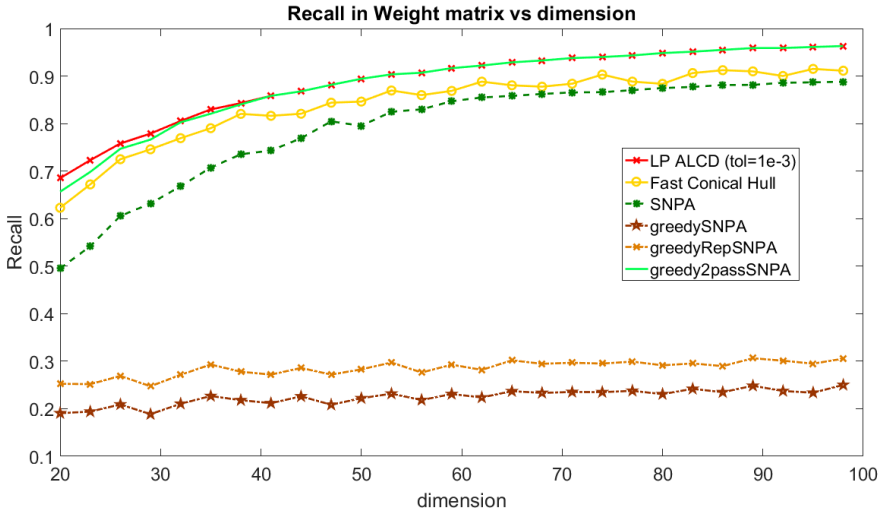

Figure 6.7: Dimension Variation: Recall

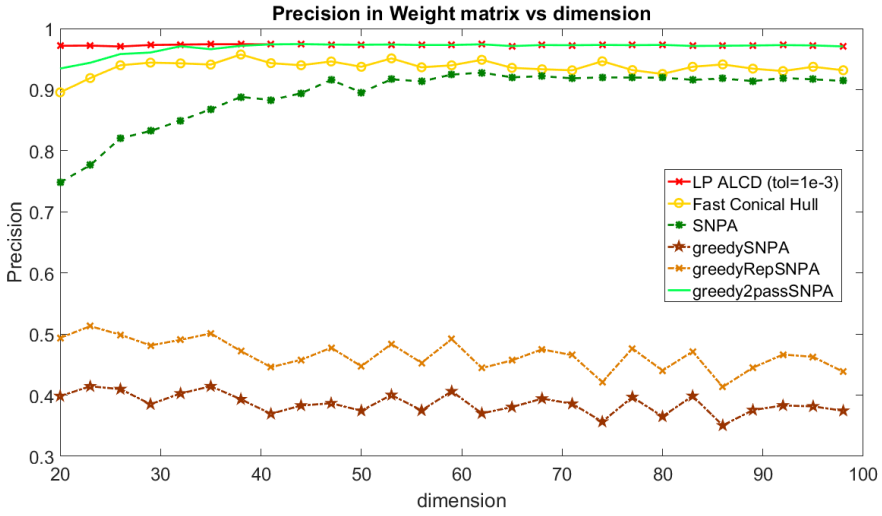

Figure 6.8: Dimension Variation: Precision

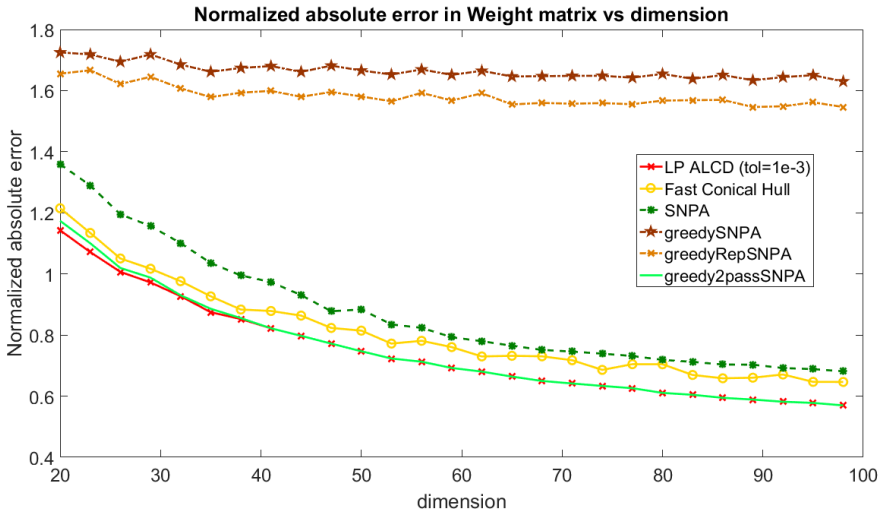

Figure 6.9: Dimension Variation: Absolute Error

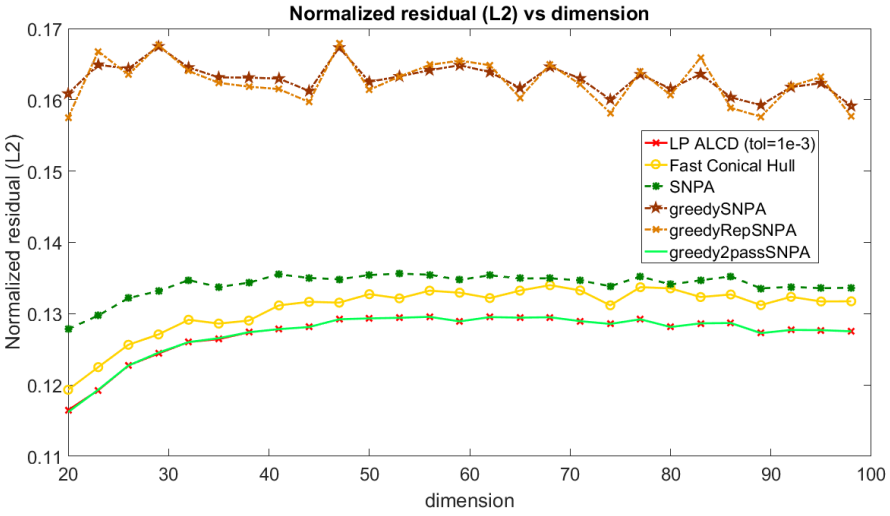

Figure 6.10: Dimension Variation: L2 Residual

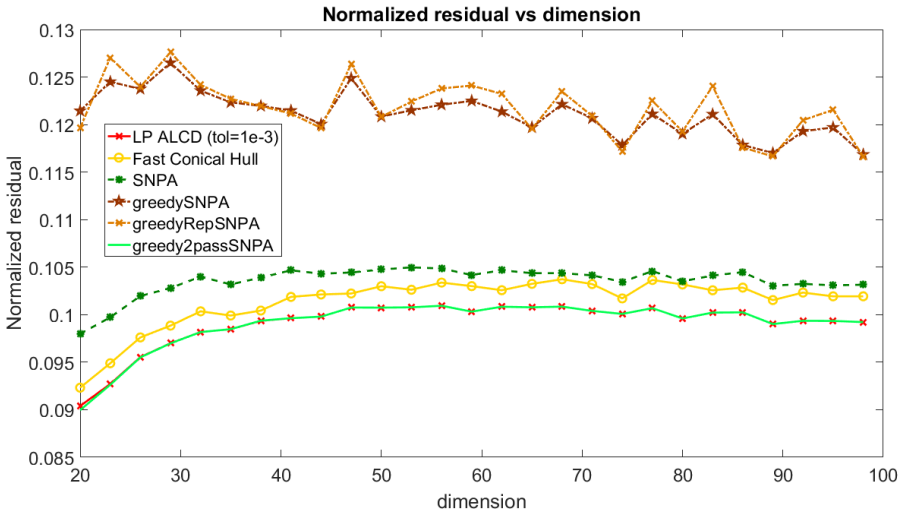

Figure 6.11: Dimension Variation: L1 Residual

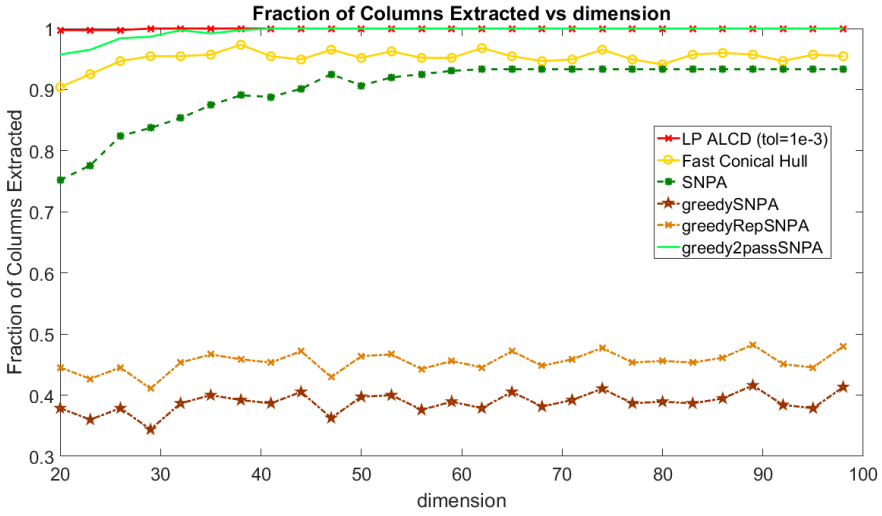

Figure 6.12: Dimension Variation: Fraction of Columns

6.3 Sample Variation

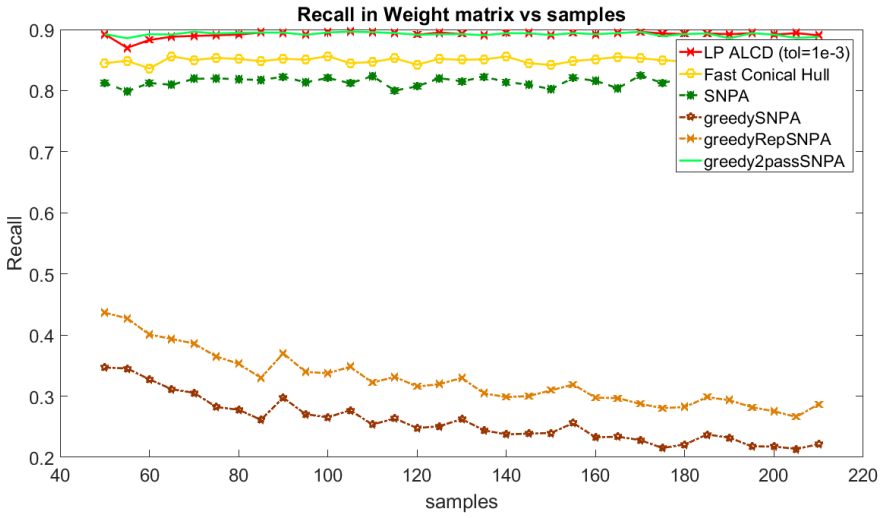

Figure 6.13: Sample Variation: Recall

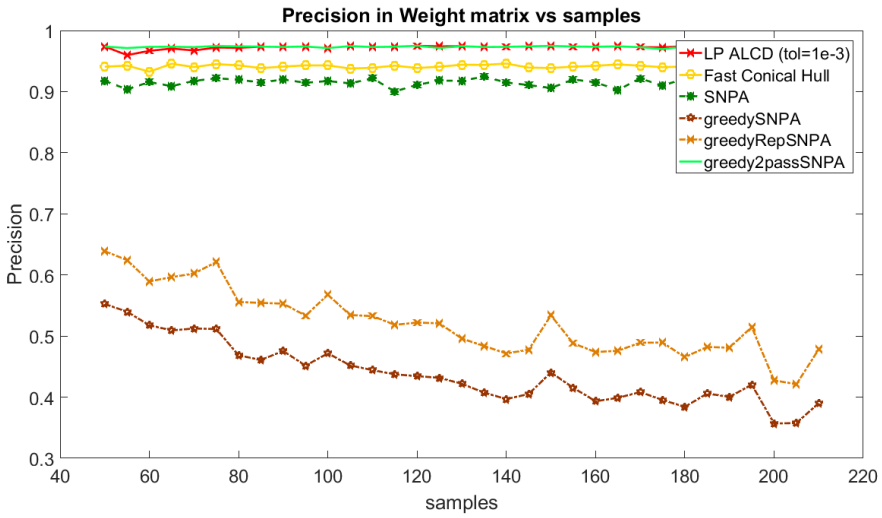

Figure 6.14: Sample Variation: Precision

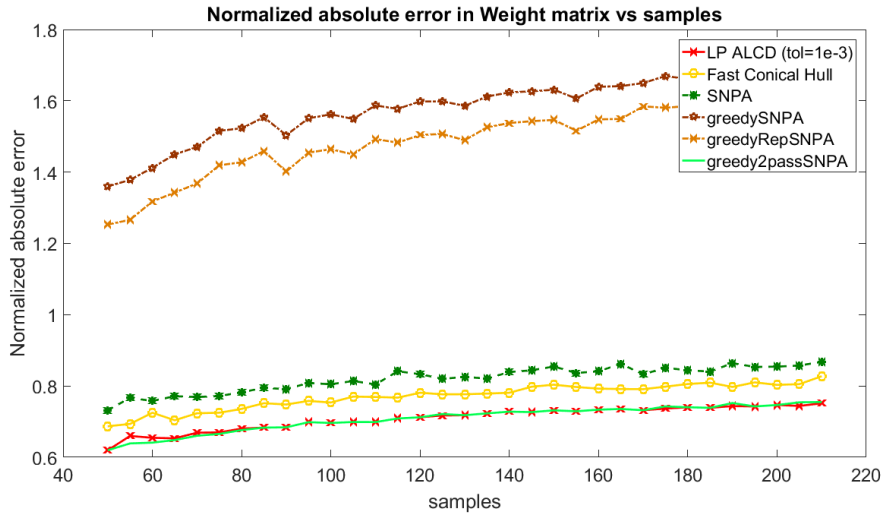

Figure 6.15: Sample Variation: Absolute Error

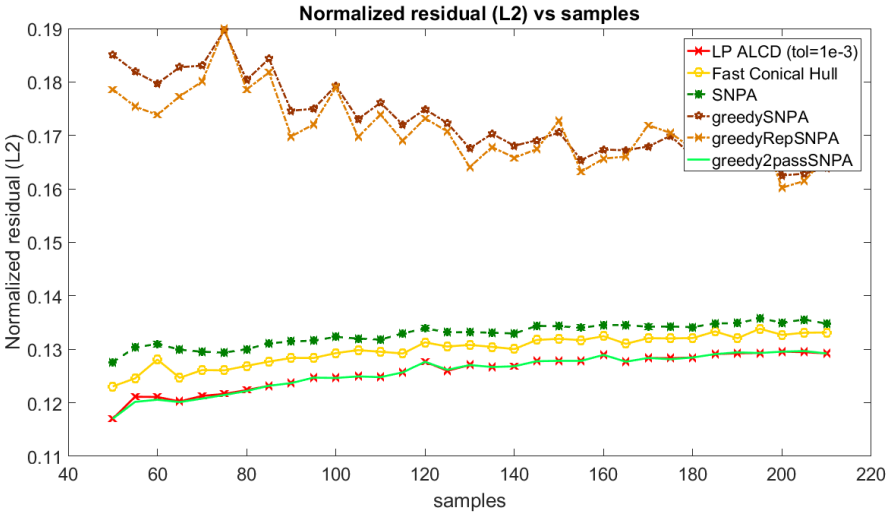

Figure 6.16: Sample Variation: L2 Residual

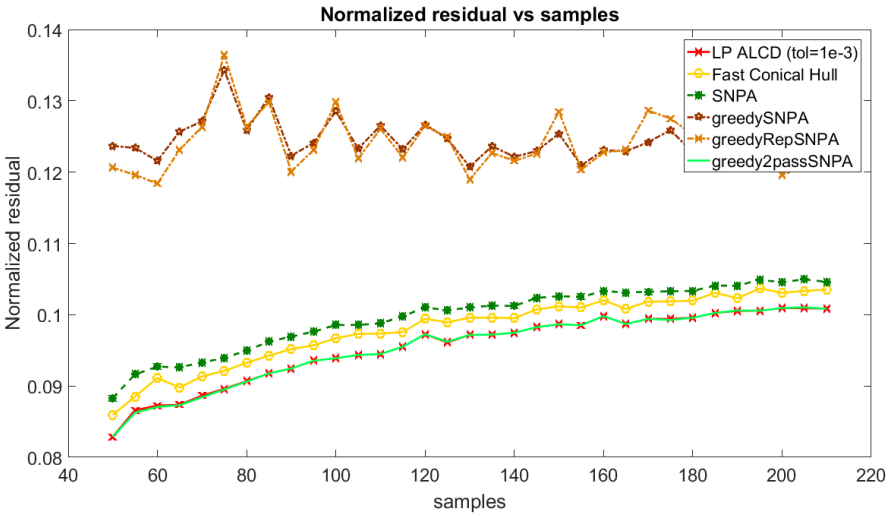

Figure 6.17: Sample Variation: L1 Residual

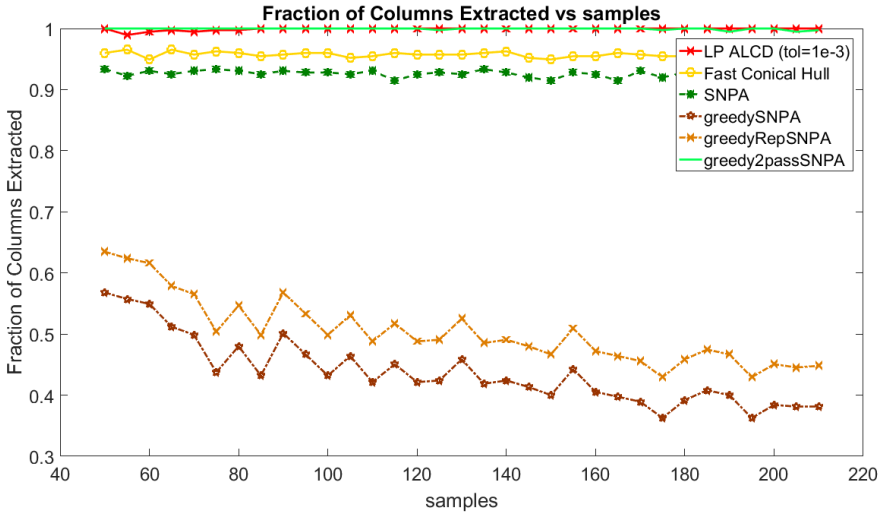

Figure 6.18: Sample Variation: Fraction of Columns

6.4 Ploygon, # of sides:

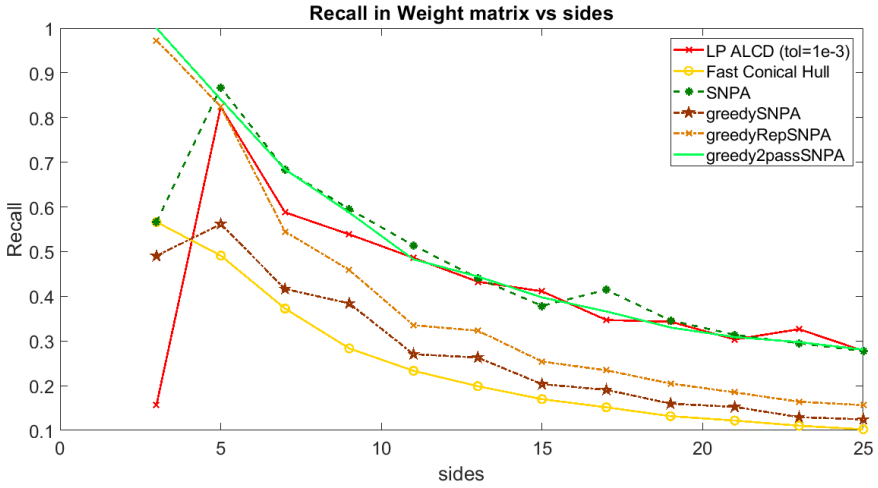

Figure 6.19: Polygon, # of sides variation: Recall

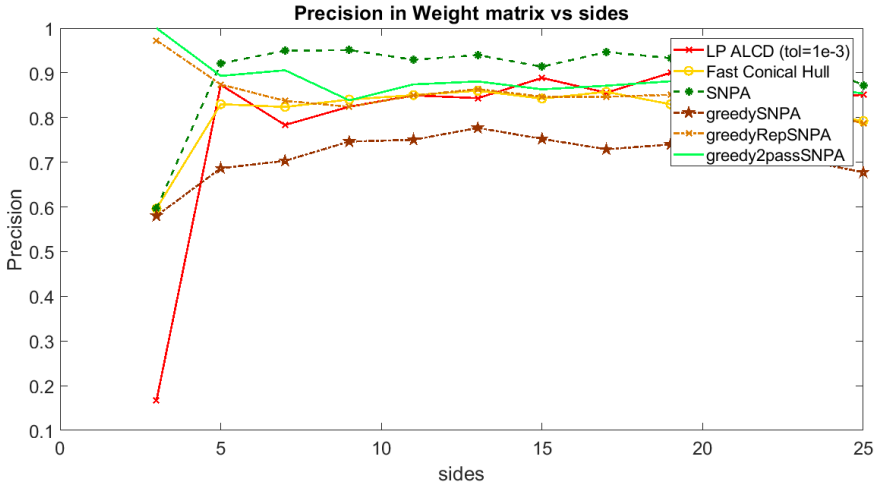

Figure 6.20: Polygon, # of sides variation: Precision

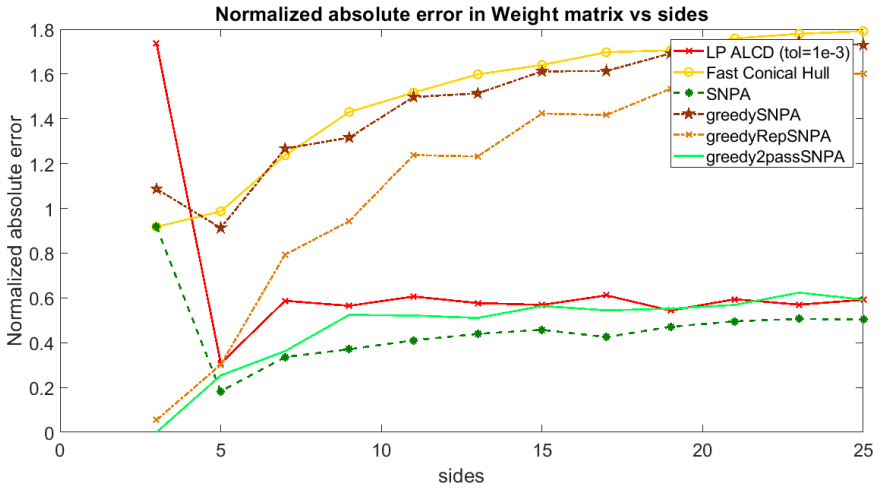

Figure 6.21: Polygon, # of sides variation: Absolute Error

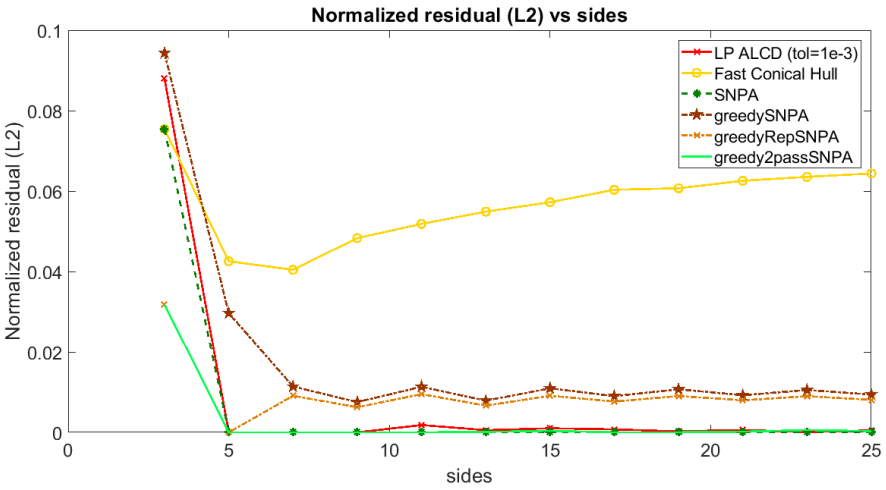

Figure 6.22: Polygon, # of sides variation: L2 Residual

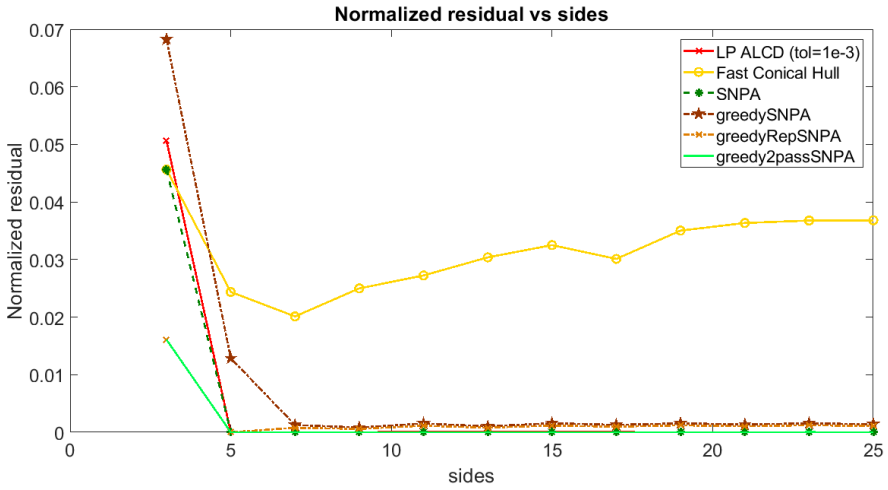

Figure 6.23: Polygon, # of sides variation: L1 Residual

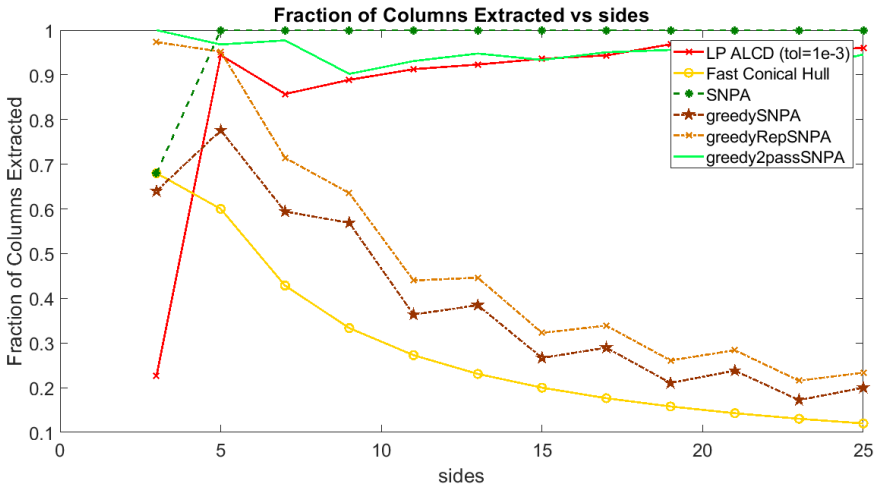

Figure 6.24: Polygon, # of sides variation: Fraction of Columns

6.5 Polygon, distance

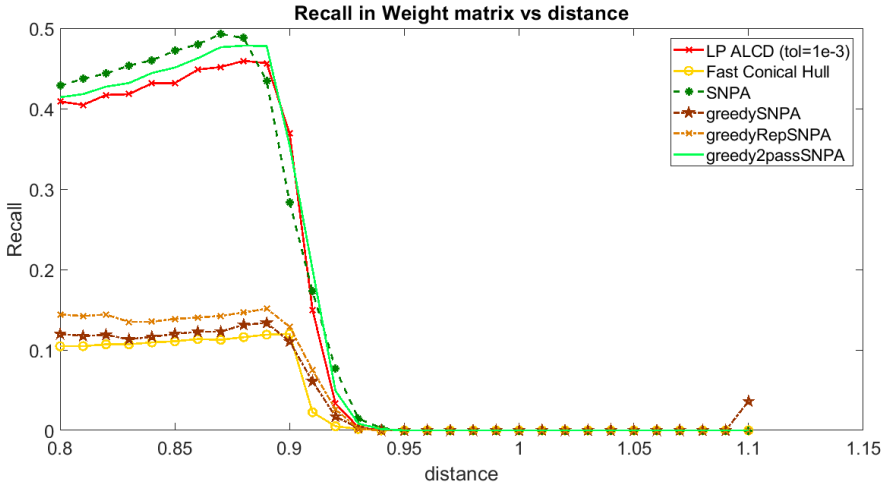

Figure 6.25: Polygon, distance variation: Recall

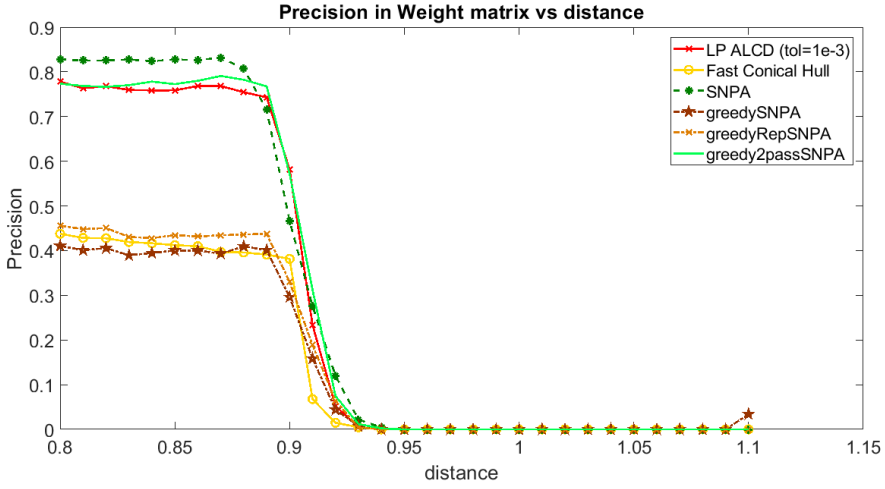

Figure 6.26: Polygon, distance variation: Precision

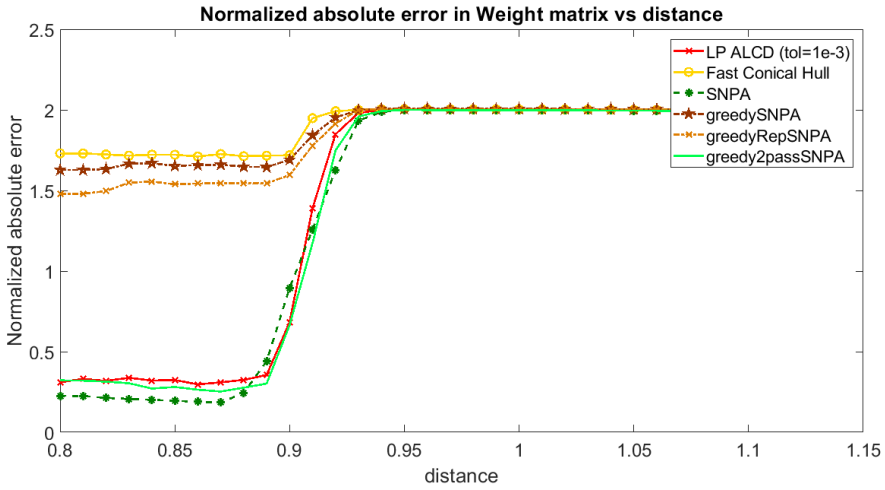

Figure 6.27: Polygon, distance variation: Absolute Error

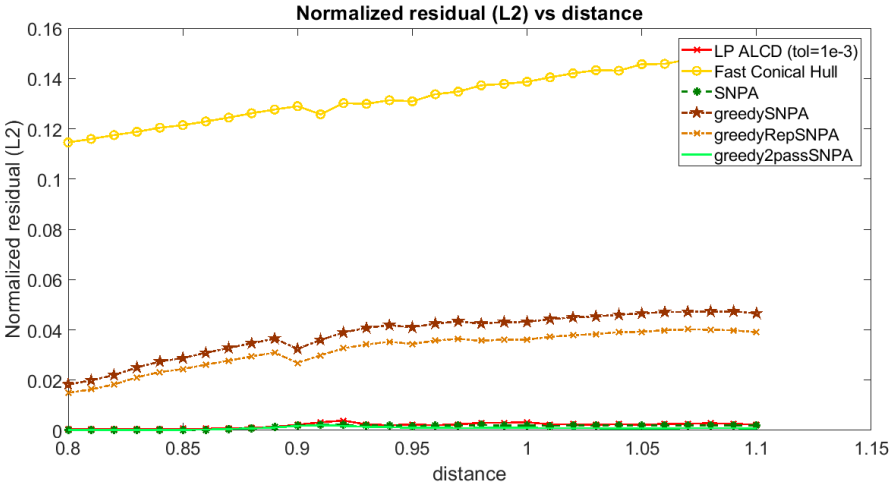

Figure 6.28: Polygon, distance variation: L2 Residual

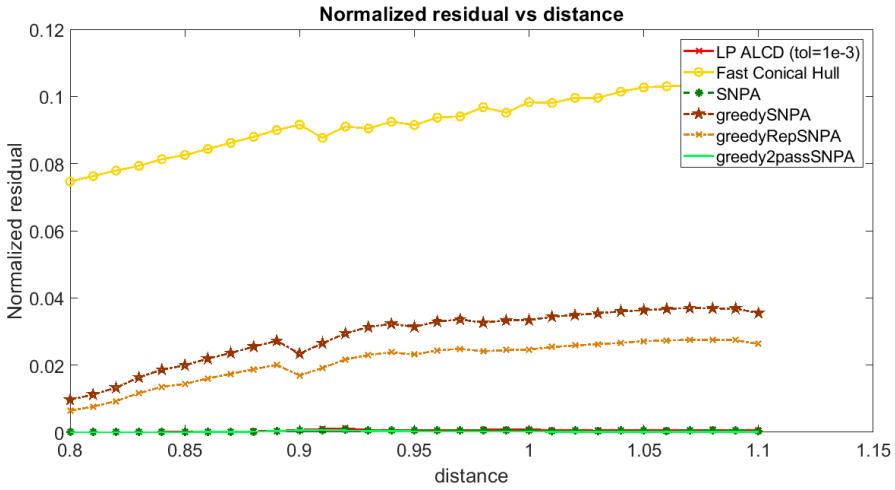

Figure 6.29: Polygon, distance variation: L1 Residual

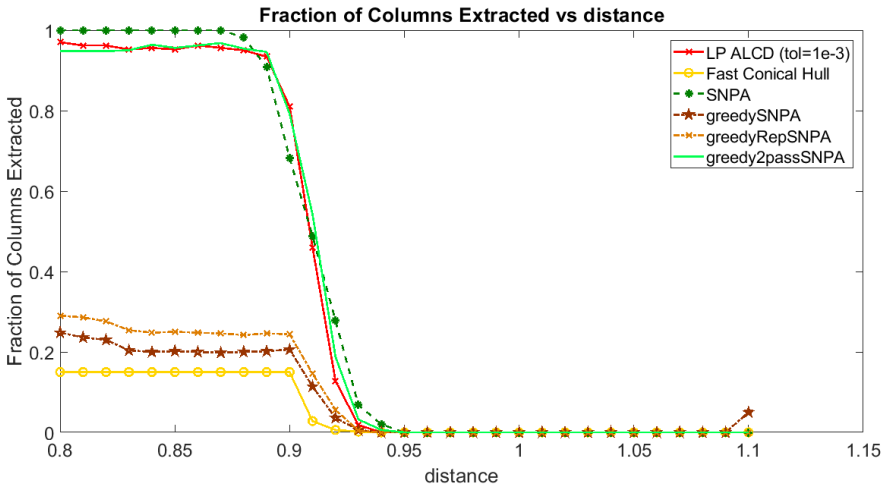

Figure 6.30: Polygon, distance variation: Fraction of Columns

6.6 Polygon, cluster radius

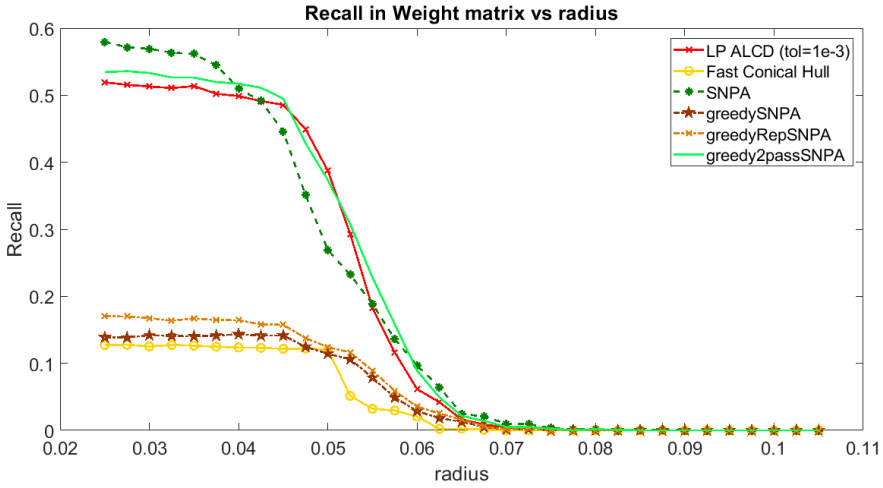

Figure 6.31: Polygon, cluster radius variation: Recall

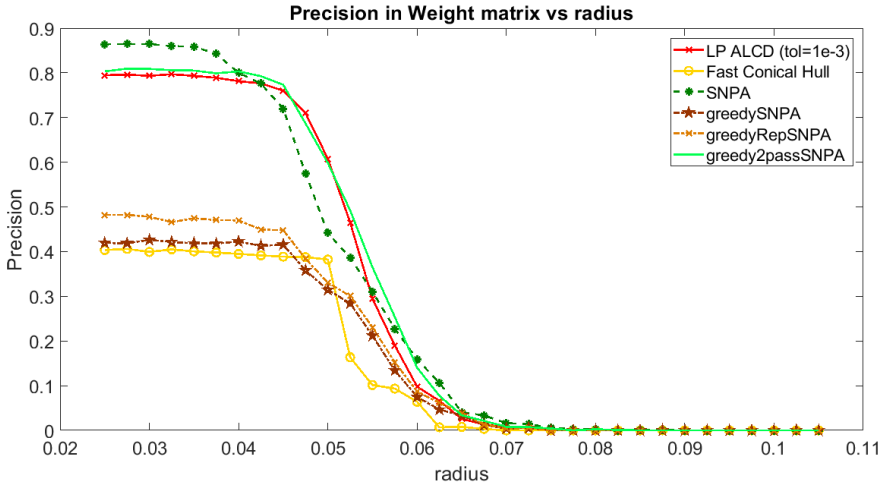

Figure 6.32: Polygon, cluster radius variation: Precision

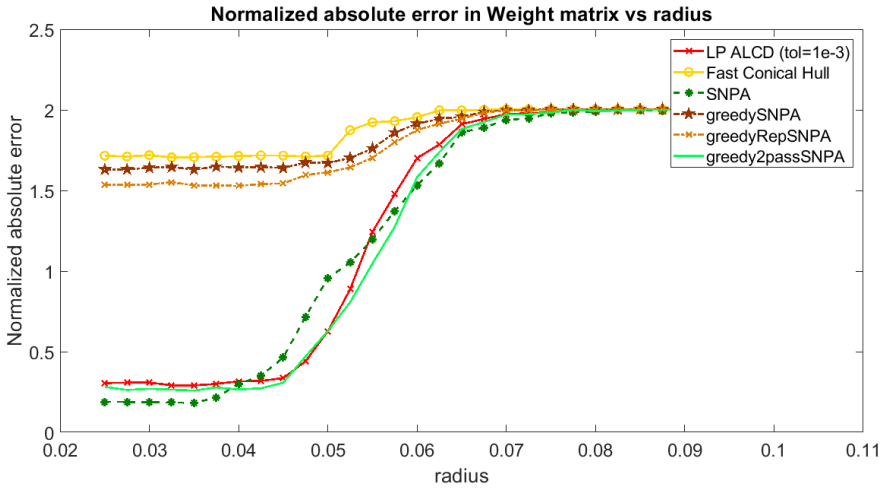

Figure 6.33: Polygon, cluster radius variation: Absolute Error

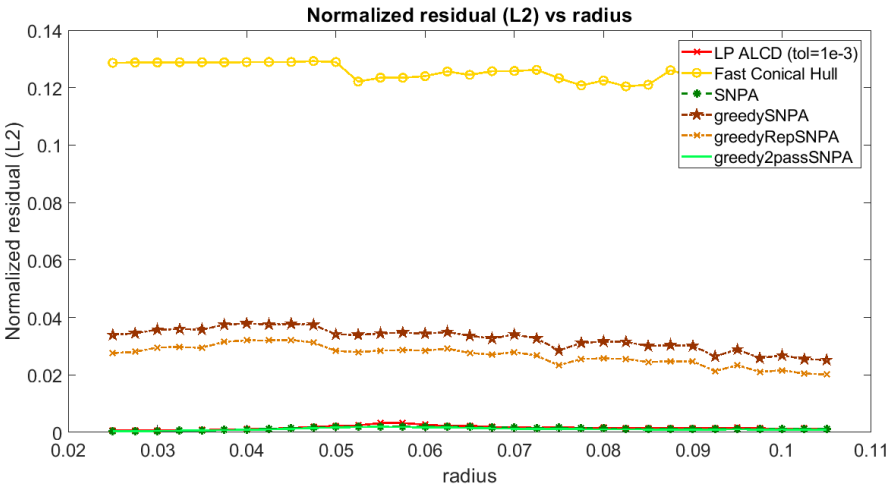

Figure 6.34: Polygon, cluster radius variation: L2 Residual

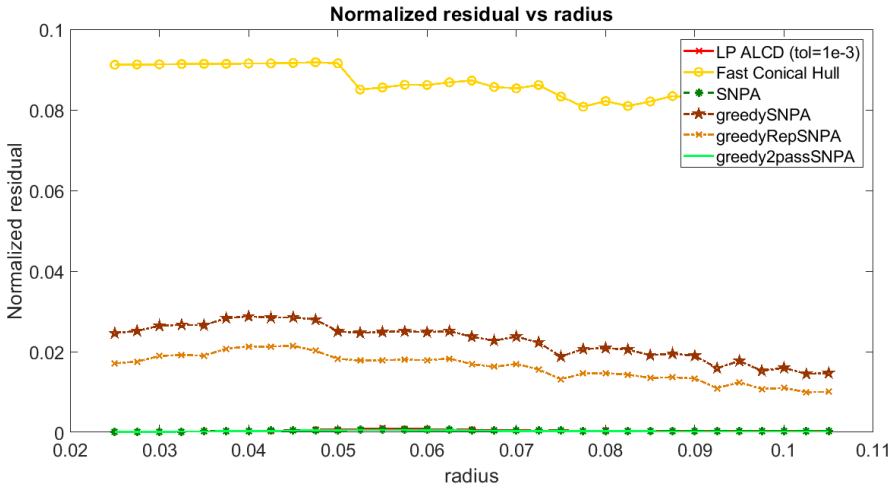

Figure 6.35: Polygon, cluster radius variation: L1 Residual

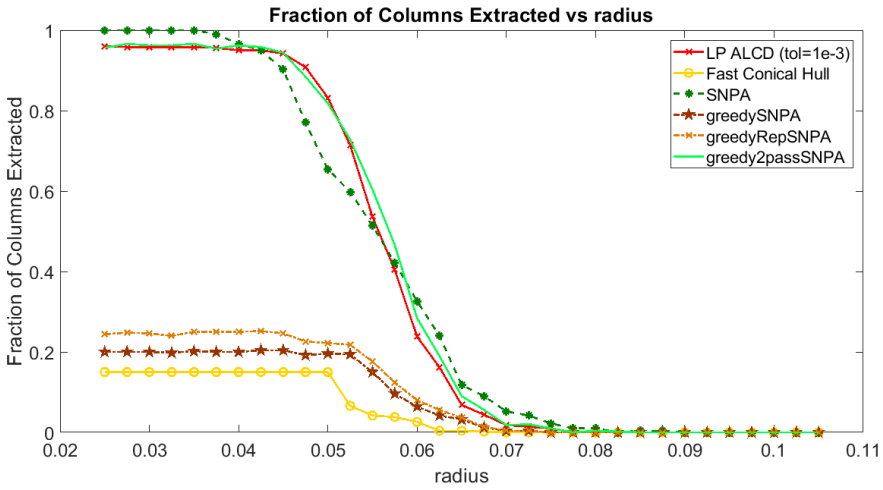

Figure 6.36: Polygon, cluster radius variation: Fraction of Columns

6.7 Results

In the uniformly generated data sest for basis, dimension, and sample variations, the greedy2pass variant and the LP ALCD algorithms outperform every other algorithm in almost every parameterand are virtually indistinguishable from one another in terms of the performance metrics. In the Uniform Dirichlet, Uniform MiddlePoints, Ill-Conditioned Dirichlet, and Ill-Conditioned MiddlePoints datasets, however, greedy2pass SNPA slightly lags behind LP ALCD, XRAY, and SNPA.

Surprisingly, however, in both the MiddlePoints datasets, as δ (noise) approaches 1, plain-vanilla greedySNPA surpasses every other algorithm. This effect is pronounced and markedly visible in the fraction of columns plots in figures of Ill-conditioned and Uniform Middle Point casess. This is probably because of the fact that even if many of the middle points are pushed away from the center of the conical hull, greedySNPA still tries to find the most *central* points in each iteration and is less affected by the skew.

Chapter 7

fMRI results

We ran SNPA, greedy2passSNPA, and LP ALCD on an fMRI dataset from a study by Alvarez, et al. [1] obtained from OpenfMRI.org (accession # ds000051). First, columns were pruned using ISC to obtain only the set of voxels that correlate well across subjects and are therefore believed to play an active role in relation to the task. Then, the data is averaged out across subjects and is flattened into an l_2 normalized $(time \times voxels)$ matrix which is the matrix that the NMF algorithms areused on. The resulting matrix is $M_{(numTimeInstances \times numActiatedVoxels)}$ which is to be factorized into $W_{(m \times r)} = M_{:J}$, and $H_{(r \times n)}$.

From Fig. 7, we can see that there is not much improvement in the residue after the number of basis columns increases beyond 5. We plot the components identified by SNPA and greedySNPA next to the (convolved) input stimulus.

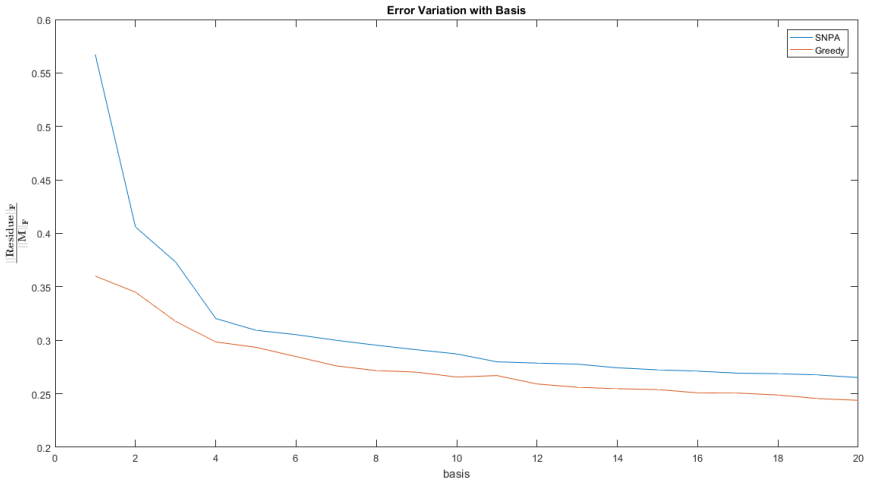

Figure 7.1: Variation of error with increase in number of basis columns used

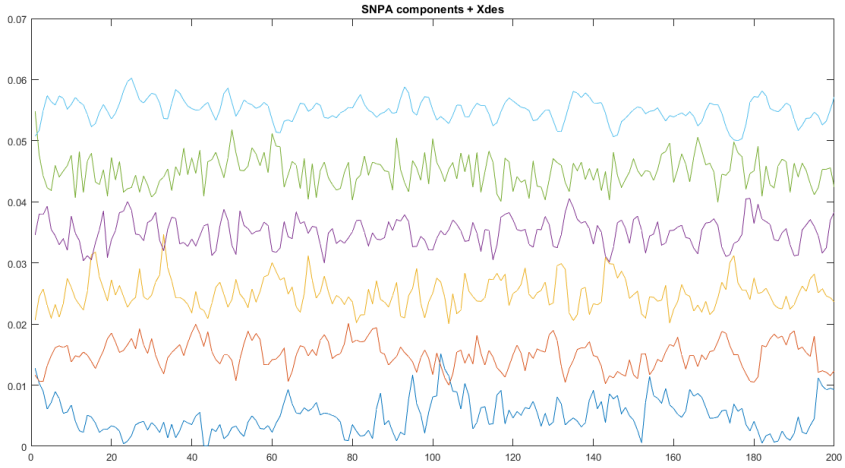

Figure 7.2: Components identified by SNPA with the input stimulus at the top

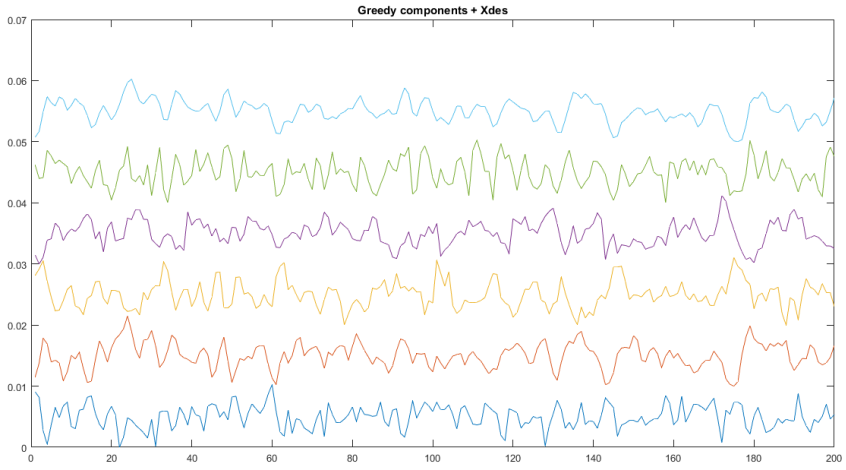

Figure 7.3: Components identified by greedy2passSNPA with the input stimulus at the top

We see that the signals of some components, particularly the 4^{th} from the bottom in Fig 7 and the 2^{nd} from the bottom in Fig 7 are positively correlated with the stimulus and some such as the 3^{rd} fromthe bottom in are negatively correlated.

To validate the results from NMF, we check if TP and TN voxels (independantly identified using (GLM)) are separated into different groups. To do this, we cluster the components identified by NMF into two groups based on the correlation values with each other and sum the H matrix along the columns for those rows in each group giving a $(2 \times n)$ H_map matrix. For the two clusters to have a good separability, each column in H_map must have values close to one and zero (if one entry is close to one, the other must be close to zero). To check this, we plot the histogram of all values in H_map for each algorithm.

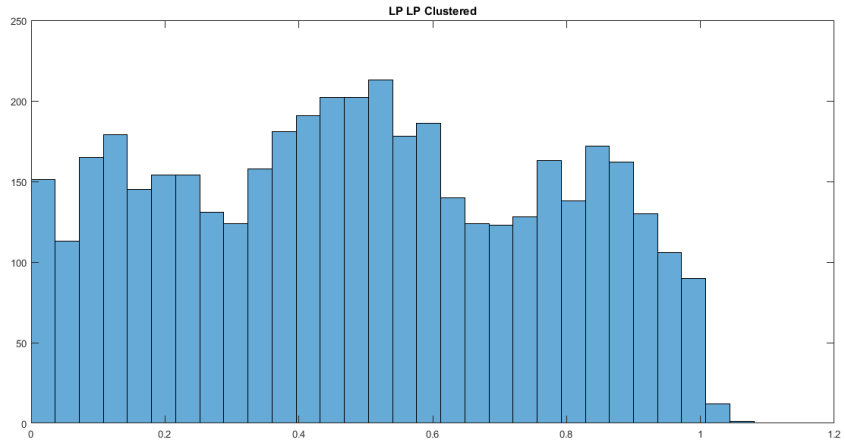

Figure 7.4: Histogram of H_map for LP ALCD

We can see that greedy2pass SNPA in Fig. 7 has the best separation with two peaks near one and zero as expected.

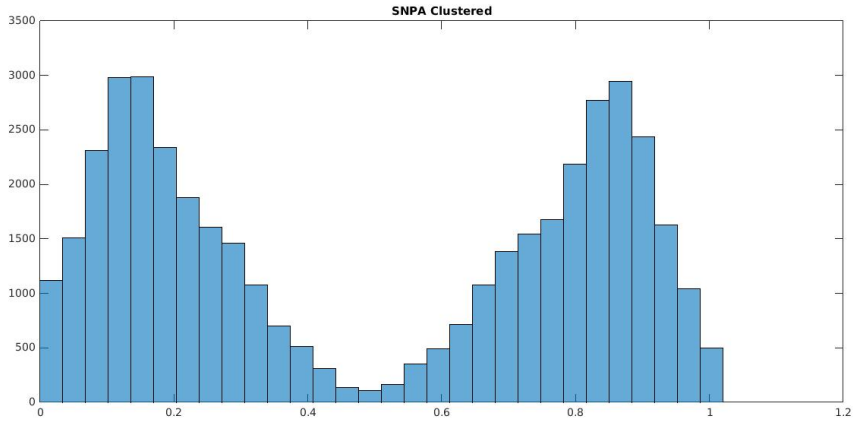

Figure 7.5: Histogram of H_map for SNPA

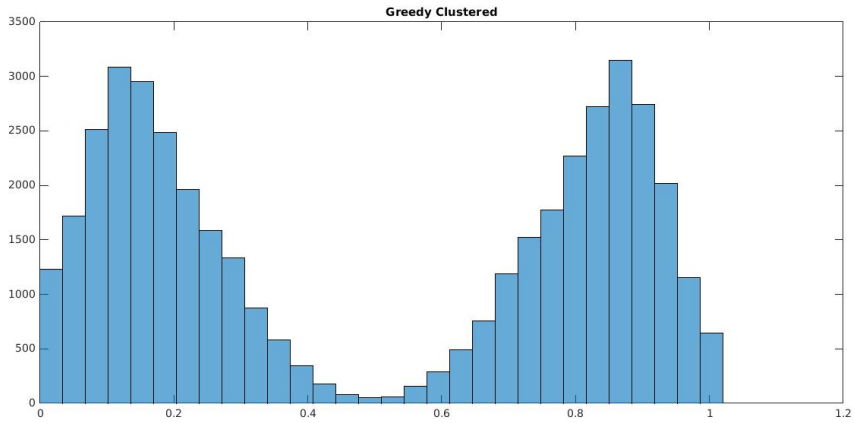

Figure 7.6: Histogram of H_map for greedy2pass

Chapter 8

Future Work

The greedy variants have a comparable performance to LP-based methods and are much faster and require far less memory to run. The greedy variants are also more robust to outliers than other conical hull algorithms. However, they have no error guarantees even in the noiseless case (as shown in Fig. 3.2). We need some bound on the error with respect to some norm. We could also derive some error bound for the greedy dropping strategy.

Bibliography

[1] G.2 ; Poldrack R. A. Alvarez, R. P. ; Jasdzewski. Building memories in two languages: An fmri study of episodic encoding in bilinguals. *Orlando, FL: Society for Neuroscience*, 2002.

[2] Saldanha B. Galvão R. Yoneyama T. Chame H. Visani V. Araùjo, U. The successive projections algorithm for variable selection in spectroscopic multicomponent analysis. *Chemometrics and Intelligent Laboratory Systems*, 2001.

[3] Ge Rong Kannan Ravi Arora, Sanjeev and Ankur Moitra. Computing a nonnegative matrix factorization provably. *STOC*, 2012.

[4] Recht Benjamin Re Christopher Bittorf, Victor and Joel A. Tropp. Factoring nonnegative matrices with linear programs. *NIPS*, 2012.

[5] Rahul Garg Chayan Sharma. 2016.

[6] D. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts? *NIPS*, 2003.

[7] Nicolas Gillis. Successive nonnegative projection algorithm for robust nonnegative blind source separation. 2014.

[8] M. D. Plumbley J. Fritsch. Score informed audio source separation using constrained nonnegative matrix factorization and score synthesis. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), in , Vancouver, Canada*, 2013.

[9] Sindhwani V. Kambadur P. Kumar, A. Fast conical hull algorithms for near-separable non-negativematrix factorization. *International Conference on Machine Learning*, 2013.

[10] S. A. Raczynski; N. Ono; S. Sagayama. Multipitch analysis with harmonic nonnegative matrix approximation. *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2007.

[11] Vavasis. On the complexity of non-negative matrix factorization. *SIAM Journal on Optimization*, 2009.

[12] N. Bertin; R. Badeau ; E. Vincent. Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription. *Transactions on Audio, Speech, and Language Processing*, 2010.