Using Semidefinite Programming to Decode Binary Linear Codes

Anmol Kagrecha
EE, IIT Bombay

James Saunderson
ECSE, Monash University

Abstract

We propose a semidefinite programming (SDP) based decoding algorithm for approximate maximum likelihood (ML) decoding of an arbitrary binary linear code transmitted over a binary symmetric channel (BSC). Firstly, we show that any integer solution of the SDP is a codeword, and so is the solution of the ML decoding problem. Second, we show that the probability of error is independent of transmitted codeword. Third, we show that whenever the linear programming (LP) based decoder of Feldman et al. [2006] succeeds, so does our decoder. However, on performing simulations using LDPC codes on the BSC channel, we observe that the SDP decoder doesn't give a significant improvement in performance when compared to the LP decoder.

1 Introduction

Maximum likelihood decoding for several classes of codes is a provably a NP-hard problem Berlekamp et al. [1978]. Practical decoding algorithms like belief propagation usually work well, but lack rigorous guarantees on performance. Feldman et al. [2006] used a LP decoder and showed that their decoder could correct a constant fraction of errors for a certain class of LDPC codes. Semidefinite programming is a more general and powerful technique compared to linear programming. It is quite natural to ask how much better the SDP decoders could perform in comparison to LP decoders practically and if better guarantees can be provided?

We are able to show that the performance of the SDP decoder is at least as good as the performance of the LP decoder. We also prove some basic results related to the SDP decoder, viz., any integer solution of the SDP is a codeword and the probability of error of the decoder is independent of the transmitted codeword for any memory-less channel.

Finally, we perform some numerical simulations on LDPC codes. It turns out that the performance of the SDP decoder is only marginally better than the LP decoder. This can be attributed to the sparsity of the LDPC codes. LP decoding relaxation captures the relations well enough and additional constraints in the SDP don't give a significant advantage.

2 Preliminaries

2.1 LDPC Codes

Consider a binary linear code $\mathcal{C}$ with a parity-check matrix H of size $m \times n$ and rate at least $1 - m/n$. Let $V = \{1, \cdots, n\}$ and $C = \{1, \cdots, m\}$ be the indices of columns and rows of H respectively. Code $\mathcal{C}$ can be represented as a bipartite graph with node sets V and C, and edges (v, c) between variable node v and check node c for all $v,\ c$ where $H_{c,v} = 1$. The bipartite graph $\mathcal{G}$ representation of a code is called the Tanner graph or the factor graph of the code. If the parity-check matrix is sparse, i.e., the number of non-zero entries in the matrix is bounded and independent of n, then the code is said to have low density.

Graph $\mathcal{G}$ can be used to visualize the code. Each variable node v is assigned a bit $\{0, 1\}$ from the codeword. A check node c is satisfied if the binary sum of bits assigned to variable nodes in the neighborhood of c is zero. The n bits form a valid codeword iff all the parity checks are satisfied.

2.2 Symmetric representation of a binary linear code

For a binary linear code $\mathcal{C}$, the original codeword polytope is defined as convex hull of all possible codewords

$$\text{poly}(\mathcal{C}) = \left\{ \sum_{\mathbf{f}\in\mathcal{C}} \lambda_{\mathbf{f}}\mathbf{f} : \lambda_{\mathbf{f}} \geq 0,\ \sum_{\mathbf{f}\in\mathcal{C}} \lambda_{\mathbf{f}} = 1 \right\}$$

For every codeword $\mathbf{f} \in \mathcal{C}$, consider $\mathbf{x}$ whose components are $x_i = (-1)^{f_i}\ \forall i \in \{1, ..., n\}$. We call $\mathbf{x}$ as the symmetric representation of $\mathbf{f}$. We further define

$$\mathcal{C}_x = \{\mathbf{x} \in \{\pm 1\}^n : x_i = (-1)^{f_i}\ \forall i \in \{1, ..., n\}\ \forall\ \mathbf{f} \in \mathcal{C}\}$$

$$\text{poly}(\mathcal{C}_x) = \left\{ \sum_{\mathbf{x}\in\mathcal{C}_x} \lambda_{\mathbf{x}}\mathbf{x} : \lambda_{\mathbf{x}} \geq 0,\ \sum_{\mathbf{x}\in\mathcal{C}_x} \lambda_{\mathbf{x}} = 1 \right\}$$

Let the neighborhood of a check $c \in C$ be denoted as $\mathcal{N}(c)$. If $\mathcal{N}(c) = \{v_{c,1}, ..., v_{c,k}\}$, then the check is satisfied if the binary sum $f_{v_{c,1}} + ... + f_{v_{c,k}} = 0$. Similarly, we say $\mathbf{x} \in \mathcal{C}_x$ *satisfies* the check c if $\prod_{i=1}^{k} x_{c,i} = 1$. For a check c involving k variable nodes define

$$\mathcal{C}_{x,c} = \{\mathbf{x} \in \{\pm 1\}^n : \prod_{i=1}^{k} x_{v_i} = 1,\ H_{c,v_i} = 1\}$$

$$\text{poly}(\mathcal{C}_{x,c}) = \left\{ \sum_{\mathbf{x}\in\mathcal{C}_{x,c}} \lambda_{\mathbf{x}}\mathbf{x} : \lambda_{\mathbf{x}} \geq 0,\ \sum_{\mathbf{x}\in\mathcal{C}_{x,c}} \lambda_{\mathbf{x}} = 1 \right\}$$

2.3 Projection based description of a parity polytope

Consider the following polytope:

$$\text{PAR}_n = \text{conv}\{(x_{0,1}, x_{0,2}, ..., x_{0,n}) \in \{-1, 1\}^n : \prod_{i=1}^{n} x_{0,i} = 1\}$$

It requires $\mathcal{O}(2^n)$ inequalities to describe PAR_n. However, a more efficient representation exists which requires only $\mathcal{O}(n)$ inequalities to describe. The is a projection based description and is described below.

If $n = 2m + 1$ introduce new variables $x_{1,1}, x_{1,2}, ..., x_{1,m+1}$ such that

$$\begin{gathered}
(x_{0,1},x_{0,2}, x_{1,1}) \in \text{PAR}_3 \\
(x_{0,3},x_{0,4}, x_{1,2}) \in \text{PAR}_3 \\
\vdots \\
(x_{0,2m-1},x_{0,2m}, x_{1,m}) \in \text{PAR}_3 \\
(x_{2m+1}, x_{1,m+1}) \in \text{PAR}_2
\end{gathered}$$

If $n = 2m$ introduce new variables $x_{1,1}, x_{1,2}, ..., x_{1,m}$ such that

$$\begin{gathered}
(x_{0,1},x_{0,2}, x_{1,1}) \in \text{PAR}_3 \\
(x_{0,3},x_{0,4}, x_{1,2}) \in \text{PAR}_3 \\
\vdots \\
(x_{0,2m-1},x_{0,2m}, x_{1,m}) \in \text{PAR}_3
\end{gathered}$$

Similarly for the $\left\lfloor \frac{n+1}{2} \right\rfloor$ new variables, introduce $\left\lfloor \frac{\left\lfloor \frac{n+1}{2} \right\rfloor + 1}{2} \right\rfloor$ new variables satisfying parity constraints

as shown above until there are only 2 or 3 new variables introduced.

$$\begin{aligned}\text{PRO}_n = \{(x_{0,1}, x_{0,2}, ..., x_{0,n}) : (x_{0,1},&x_{0,2}, x_{1,1}) \in \text{PAR}_3\\ (x_{0,3},&x_{0,4}, x_{1,2}) \in \text{PAR}_3\\ &\vdots\\ (x_{0,2\lfloor\frac{n}{2}\rfloor-1},&x_{0,2\lfloor\frac{n}{2}\rfloor}, x_{1,\lfloor\frac{n}{2}\rfloor}) \in \text{PAR}_3\\ &\vdots\}\end{aligned}$$

We next mention the inequalities describing PAR_2 and PAR_3. $(x_1, x_2) \in \text{PAR}_2$ if

$$\begin{aligned}x_1 - x_2 &\leq 0\\ x_2 - x_1 &\leq 0\\ |x_1| &\leq 1\\ |x_2| &\leq 1\end{aligned}$$

$(x_1, x_2, x_3) \in \text{PAR}_3$ if

$$\begin{aligned}x_1 - x_2 + x_3 &\leq 1\\ -x_1 + x_2 + x_3 &\leq 1\\ x_1 + x_2 - x_3 &\leq 1\\ -x_1 - x_2 - x_3 &\leq 1\end{aligned}$$

Observe $|x_i| \leq 1\ \forall i \in \{1, 2, 3\}$ is implied by the four equations given above.

2.4 Equivalent Parity Check Matrix

A parity check matrix of a binary linear code could have an arbitrary number of variables for a check. However, for our SDP decoder we require that the checks have parity 2 or 3. All the checks in the original parity check matrix can be converted to an equivalent check matrix by introducing additional variables as we did in the previous subsection. The polytopes corresponding to these two parity check matrices can be shown to be the same.

2.5 Communication Model

We assume that a codeword $\mathbf{f}$ from a binary linear code $\mathcal{C}$ is transmitted over a binary symmetric channel with bit flipping probability p. The output of this channel is denoted by $\mathbf{g}$. The communication process is equivalent to sending $\mathbf{x} \in \mathcal{C}_x$ over a channel which flips 1 to -1 with probability p and vice-versa. Let the output of this channel be denoted by $\mathbf{y}$. Given a parity check matrix H, we convert it to another parity check matrix H' using the method in Section 2.3. We use this modified code for communication.

3 Semidefinite programming based decoder

For any binary linear code $\mathcal{C}$, we formulate a semidefinite programming based decoder. First, we convert the parity check matrix H into H' where H' has checks with 2 or 3 variables using the ideas given in Section 2.3 . Denote the set of checks for H' as C' and the set of variables for H' as V'.

For a check $c \in C$, let the number of neighbors $|\mathcal{N}(c)|$ be denoted as k_c. Then index the variables in V' as follows: the first n are the 0^{th} stage variables, the next $\sum_{c\in C}\lfloor\frac{k_c+1}{2}\rfloor$ are the 1^{st} stage variables and so on.

The variable in our SDP is denoted by Z. Output of the SDP decoder will be $x^* = Z[2 : n + 1]$ if $x^* \in \{\pm 1\}^n$, otherwise we will declare an error. The size of the matrix Z is $(|V'| + 1) \times (|V'| + 1)$.

The first constraint of the SDP is that Z has to be a semidefinite matrix:

$$Z \succeq 0$$

We want the output of the SDP to be in $\{\pm 1\}^n$. Hence, the corresponding constraint is:

$$Z_{i,i} = 1 \ \forall i = \{1, ..., n\}$$

finally we should have constraints related to parity check H'. If we have a check $c' \in C'$ involving three variables z_1, z_2, z_3 having index i_1, i_2, i_3, then there should be constraints like

$$\begin{aligned} Z[i_1 + 1, 1] &= Z[i_2 + 1, i_3 + 1] \\ Z[i_2 + 1, 1] &= Z[i_3 + 1, i_1 + 1] \\ Z[i_3 + 1, 1] &= Z[i_1 + 1, i_2 + 1] \end{aligned}$$

and if we have a check $c' \in C'$ involving two variables z_1, z_2 having index i_1, i_2, then there should be constraints like

$$\begin{aligned} Z[i_1 + 1, 1] &= Z[i_2 + 1, 1] \\ Z[i_1 + 1, i_2 + 1] &= 1 \end{aligned}$$

Using the ideas given above we formulate the SDP for decoding -

$$\begin{aligned} \text{maximize } & \langle Y, Z \rangle \\ \text{s.t. } & Z \succeq 0 \\ & \langle E_{i,i}, Z \rangle = 1 \ \forall i \in \{1, ..., 1 + |V'|\} \\ & \langle B_{i,c'}, Z \rangle = 0 \ \forall i \ \forall c' \in C' \\ \text{where } Y &= \begin{bmatrix} 0 & \mathbf{y}^T/2 & \mathbf{0}_{1 \times |V'| - n} \\ \mathbf{y}/2 & \mathbf{0}_{|V'| \times |V'|} & \\ \mathbf{0}_{|V'| - n \times 1} & & \end{bmatrix} \\ E_{i,i}[j,k] &= \begin{cases} 1, & (j,k) = (i,i) \\ 0, & \text{otherwise} \end{cases} \end{aligned}$$

For a check c' involving three variables having index i_1, i_2, i_3, we will have three $B_{1,c'}, B_{2,c'}, B_{3,c'}$ of the form -

$$B_{1,c'}[j,k] = \begin{cases} 1, & (j,k) \in \{(1, i_1 + 1), (i_1 + 1, 1)\} \\ -1, & (j,k) \in \{(i_3 + 1, i_2 + 1), (i_2 + 1, i_3 + 1)\} \\ 0, & \text{otherwise} \end{cases}$$

For a check c' involving two variables having index i_1, i_2, we will have three $B_{1,c'}, B_{2,c'}$ of the form -

$$B_{1,c'}[j,k] = \begin{cases} 1, & (j,k) \in \{(i_1 + 1, i_1 + 1), (i_2 + 1, i_2 + 1)\} \\ -1, & (j,k) \in \{(i_1 + 1, i_2 + 1), (i_2 + 1, i_1 + 1)\} \\ 0, & \text{otherwise} \end{cases}$$

$$B_{2,c'}[j,k] = \begin{cases} 1, & (j,k) \in \{(i_1 + 1, 1), (1, i_1 + 1)\} \\ -1, & (j,k) \in \{(i_2 + 1, 1), (1, i_2 + 1)\} \\ 0, & \text{otherwise} \end{cases}$$

3.1 Codewords in Feasible Region of SDP

Consider a codeword $\mathbf{x}_0 \in \mathcal{C}_x$. Then we can construct $\mathbf{x}'$ such that $\mathbf{x}'$ satisfies all checks $c' \in C'$. We claim that the following lies in the feasible region of SDP -

$$Z' = \begin{bmatrix} 1 & \mathbf{x}'^T \\ \mathbf{x}' & \mathbf{x}'\mathbf{x}'^T \end{bmatrix}$$

Firstly, note that $Z' \succeq 0$. Then observe that $x_i'^2 = 1 \ \forall i \in \{1, ..., |V'|\}$. Hence, all constraints related to $E_{i,i}$ are satisfied. Finally, observe the following -

$$Z'[i+1, 1] = x'_i$$
$$Z'[i+1, j+1] = x'_{i+1}x'_{j+1}$$

The relations implied by $B_{i,c'}$ matrices reduce to parity constraints as can be seen above. As $\mathbf{x}'$ satisfies all the checks, all constraints related to $B_{i,c'}$ are satisfied. Hence, all codewords have a Z that lies in the feasible region of SDP.

3.2 Integer Solution from SDP

Here, we show that if the output $\mathbf{x}_0^*$ of the SDP is integral, i.e., $\mathbf{x}_0^* \in \{\pm 1\}^n$, then the output is indeed a codeword.

3.2.1 Rank one solution

On solving the SDP we get the solution Z^*. As $\mathbf{x}_0^* = Z^*[2 : n+1, 1]$. If $\mathbf{x}_0^* \in \{\pm 1\}^n$, then we claim that Z^* has rank one and

$$Z^* = \begin{bmatrix} 1 & \mathbf{x}'^T \\ \mathbf{x}' & \mathbf{x}'\mathbf{x}'^T \end{bmatrix}$$
$$\text{where } \mathbf{x}'[1:n] = \mathbf{x}_0^*$$
$$\text{and } \mathbf{x}' \in \{\pm 1\}^n$$

i.e., we don't need to check if all the variables have absolute value 1 to ensure that the code is valid. We can just check if the original values have absolute value 1 to see if the code is valid.

Proof - First, consider the case for $n = 2$

$$Z^* = \begin{bmatrix} 1 & x_{0,1}^* & x_{0,2}^* \\ x_{0,1}^* & 1 & Z_{2,3}^* \\ x_{0,2}^* & Z_{3,2}^* & 1 \end{bmatrix}$$

Note that $Z_{2,3}^* = Z_{3,2}^*$ because Z^* is semidefinite. The diagonal entries are 1 because they are constraints of the SDP. Determinant of Z^* should be positive. Hence we have,

$$\det(Z^*) = \begin{vmatrix} 1 & x_{0,1}^* & x_{0,2}^* \\ x_{0,1}^* & 1 & Z_{2,3}^* \\ x_{0,2}^* & Z_{3,2}^* & 1 \end{vmatrix} \geq 0$$

As $x_{0,i}^{*2} = 1 \ \forall i \in \{1, 2\}$

$$\det(Z^*) = -(Z_{2,3}^* - x_{0,1}^* x_{0,2}^*)^2$$
$$\Rightarrow -(Z_{2,3}^* - x_{0,1}^* x_{0,2}^*)^2 \geq 0$$
$$\Rightarrow Z_{2,3}^* = x_{0,1}^* x_{0,2}^*$$
$$\Rightarrow Z^* = \begin{bmatrix} 1 & x_{0,1}^* & x_{0,2}^* \\ x_{0,1}^* & x_{0,1}^{*2} & x_{0,1}^* x_{0,2}^* \\ x_{0,2}^* & x_{0,1}^* x_{0,2}^* & x_{0,2}^{*2} \end{bmatrix}$$

To prove this for general n, we use the following property of semidefinite matrices -

All principal minors of a semi-definite matrix are non-negative. For length n codes we have -

$$Z^* = \begin{bmatrix} 1 & x^*_{0,1} & \dots & x^*_{0,n} & \dots \\ x^*_{0,1} & 1 & \dots & Z^*_{n+1,2} & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x^*_{0,n} & Z^*_{n+1,2} & \dots & 1 & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots \end{bmatrix}$$

First, we show $Z^*_{i+1,j+1} = x^*_{0,i}x^*_{0,j}$ where $i \neq j$ and $i, j \leq n$. Consider a principal minor indexed by $s = \{1, i+1, j+1\}$

$$Z^*_s = \begin{bmatrix} 1 & x^*_{0,i} & x^*_{0,j} \\ x^*_{0,i} & 1 & Z^*_{i+1,j+1} \\ x^*_{0,j} & Z^*_{j+1,i+1} & 1 \end{bmatrix}$$

As Z^* is PSD,

$$\begin{aligned} & det(Z^*_s) \geq 0 \\ \Rightarrow & -(Z_{i+1,j+1} - x^*_{0,i}x^*_{0,j})^2 \geq 0 \\ \Rightarrow & Z_{i+1,j+1} = x^*_{0,i}x^*_{0,j} \end{aligned}$$

Now consider any constraint involving level 1 and level 0 variables. If n is odd, then we will have constraints $B_{i,c'}$ corresponding to a degree 2 check. Hence,

$$x_{0,2m+1} = x_{1,m+1}$$

For constraints constructed from degree 3 checks, consider a check $c' \in C'$. Without loss of generality, let it involve $x_{0,1}, x_{0,2}$ and $x_{1,1}$. Let the index for the the variables be 1,2 and $n+1$. We just showed $Z[2,3] = x_{0,1}x_{0,2}$. From the constraint $\langle B_{3,c'}, Z \rangle$, we have -

$$\begin{aligned} & Z[n+2,1] = Z[2,3] \\ \Rightarrow & x_{1,n+1} = x_{0,1}x_{0,2} \\ \Rightarrow & x_{1,n+1} \in \{\pm 1\} \end{aligned}$$

Similarly, we can show all the level 1 variables belong to $\{\pm 1\}$. We can now recursively show $Z[i+1, j+1] = x'_i x'_j$ and $x' \in \{\pm 1\}^{|V'|}$. Hence we have,

$$Z^* = \begin{bmatrix} 1 & \mathbf{x}' \\ \mathbf{x}' & \mathbf{x}'\mathbf{x}'^T \end{bmatrix}$$

i.e., Z^* is of rank one.

3.2.2 Validity of the decoded word

If we have

$$\begin{aligned} & Z^*[i+1, j+1] = x'_i x'_j \text{ where } i \neq j \\ & Z^*[i+1, 1] = x'_i \ \forall i \in \{1, ..., |V'|\} \\ \text{and } & \mathbf{x}' \in \{\pm 1\}^{|V'|} \end{aligned}$$

then the constraints $\langle B_{i,c'}, Z \rangle$ just reduce to parity checks. Because $\mathbf{x}'$ satisfies these constraints, it satisfies the parity checks as well. Hence, $\mathbf{x}' \in C'_x$ i.e., all the integer solutions of the SDP decoder are valid codewords.

3.3 SDP Decoder is as strong as the LP Decoder

3.3.1 Using Primal Problem

In this section, we would like to prove that the feasible region of the SDP decoder is at least as close to the convex hull of codewords as the feasible region of the LP decoder. This means that the SDP decoder will at least have the error correcting performance of the LP decoder.

Assume we had a check $c' \in C'$ which involved the checks z_1, z_2 and z_3. For simplicity of analysis and without loss of generality provide them index 1,2 and 3. Then the constraints for the LP are

$$\begin{aligned} -z_1 + z_2 + z_3 &\leq 1 \\ z_1 - z_2 + z_3 &\leq 1 \\ z_1 + z_2 - z_3 &\leq 1 \\ -z_1 - z_2 - z_3 &\leq 1 \end{aligned}$$

and the constraints for SDP are-

$$\begin{aligned} & Z \succeq 0 \\ & Z[i,i] = 1 \ \forall i \\ & Z[1+1, 2+1] = z_3 \\ & Z[2+1, 3+1] = z_1 \\ & Z[3+1, 1+1] = z_2 \\ \Rightarrow & Z = \begin{bmatrix} 1 & z_1 & z_2 & z_3 & \dots \\ z_1 & 1 & z_3 & z_2 & \dots \\ z_2 & z_3 & 1 & z_1 & \dots \\ z_3 & z_2 & z_1 & 1 & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots \end{bmatrix} \end{aligned}$$

We know that for a semidefinite matrix Z

$$\mathbf{a}^T Z \mathbf{a} \geq 0 \ \forall \ \mathbf{a} \in \mathbb{R}^{1+|V'|}$$

Put $\mathbf{a} = [-1, -1, 1, 1, 0, ..., 0]^T$, then we have

$$\begin{aligned} & 4(1 + z_1 - z_2 - z_3) \geq 0 \\ \Rightarrow & -z_1 + z_2 + z_3 \leq 1 \end{aligned}$$

Similarly, put $\mathbf{a} = [-1, 1, -1, 1, 0, ..., 0]^T$ to get

$$z_1 - z_2 + z_3 \leq 1$$

Put $\mathbf{a} = [-1, 1, 1, -1, 0, ..., 0]^T$ to get

$$z_1 + z_2 - z_3 \leq 1$$

Put $\mathbf{a} = [-1, -1, -1, -1, 0, ..., 0]^T$ to get

$$-z_1 - z_2 - z_3 \leq 1$$

It is easy to verify that the LP constraints for weight 2 checks can be derived from corresponding SDP constraints. It is easy to see that the semidefinite constraint on Z introduces many more constraints than the LP constraints. Hence, the SDP constraints are at least as strong as the LP constraints.

3.3.2 Using Dual Problem

The dual problem for the SDP is the following -

$$\begin{aligned}\text{minimize } & \sum_{i=1}^{1+|V'|} \theta_i + \sum_{c'\in C'}\sum_{i}\omega_{i,c'} \\ \text{s.t. } & \mathcal{S}(\theta,\omega,y) = \sum_{i=1}^{1+|V'|} E_{i,i}\theta_i + \sum_{c'\in C'}\sum_{i} B_{i,c'}\omega_{i,c'} - Y \succeq 0\end{aligned}$$

If $\mathbf{x} = 1^n$ is the unique solution of the SDP decoder then,

$$\begin{aligned}&\langle \mathcal{S}(\theta,\omega,y), \begin{smallmatrix}1 & \dots & 1\\ \vdots & \ddots & \vdots \\ 1 & \dots & 1\end{smallmatrix}\rangle = 0 \\ \Rightarrow &\mathcal{S}(\theta,\omega,y)1^{1+|V'|} = 0^{1+|V'|}\end{aligned}$$

Using the identity above, we can replace θ_i in terms of y_i and $\omega_{i,c'}$.

The condition for $1^{|V'|}$ to be optimum for the LP decoder is:

$$\begin{aligned}\mathbf{y}' &= \sum_{c}\sum_{j}\sum_{i}\alpha_{i,j,c}\mathbf{a}_{i,j,c} \\ &\alpha_{i,j,c} > 0 \ \forall i \ \forall j \ \forall c\end{aligned}$$

where $\mathbf{a}_{i,j,c}$ are the normals to hyperplanes at the intersection of which $1^{|V'|}$ lies. The statement above follows from complementary slackness theorem.

We will show that if $\mathbf{y}'$ has the form given above, then $\mathcal{S}(\theta,\omega,y) \succeq 0$. For the simplicity of exposition, consider a check $c' \in C'$ involving variables z_1, z_2, z_3 having index 1,2 and 3. Further consider the matrix given below -

$$s(\alpha,\omega,c') = \begin{matrix}\alpha_1+\alpha_2+\alpha_3-\omega_1-\omega_2-\omega_3 & \omega_1+\alpha_1-\alpha_2-\alpha_3 & \omega_2+\alpha_2-\alpha_1-\alpha_3 & \omega_3+\alpha_3-\alpha_1-\alpha_2 \\ \omega_1+\alpha_1-\alpha_2-\alpha_3 & -\omega_1-\alpha_1+\omega_2+\alpha_2+\omega_3+\alpha_3 & -\omega_3 & -\omega_2 \\ \omega_2+\alpha_2-\alpha_1-\alpha_3 & -\omega_3 & -\omega_2-\alpha_2+\omega_1+\alpha_1+\omega_3+\alpha_3 & -\omega_1 \\ \omega_3+\alpha_3-\alpha_1-\alpha_2 & -\omega_2 & -\omega_1 & -\omega_3-\alpha_3+\omega_1+\alpha_1+\omega_2+\alpha_2\end{matrix}$$

The additional subscripts related to check have been removed for the ease of readability. One can verify that $\mathcal{S}(\theta,\omega,y)$ can be written in terms of ωs and αs alone. Observe that

$$\mathcal{S}(\theta,\omega,y) = \sum_{c'\in C'} s'(\alpha,\omega,c')$$

where $s'(\alpha,\omega,c')$ has non-zero terms at appropriate index and is zero everywhere else.

We will show that for any set of $\alpha_{i,c'} \geq 0$, we can find $\omega_{i,c'}$ such that $s'(\alpha,\omega,c') \succeq 0 \ \forall c' \in C'$. Because sum of positive semidefinite matrices is a semidefinite matrix, $\mathcal{S}(\theta,\omega,y) \succeq 0$. Again for the ease of notation, consider the $s'(\alpha,\omega,c')$ where c' involves variables with index 1,2, and 3. Put

$$\begin{aligned}\omega_1 &= \frac{\alpha_2+\alpha_3-\alpha_1}{2} \\ \omega_2 &= \frac{\alpha_1+\alpha_3-\alpha_2}{2} \\ \omega_3 &= \frac{\alpha_1+\alpha_2-\alpha_3}{2}\end{aligned}$$

to get

$$s'(\alpha,\omega,c') = \begin{bmatrix} \frac{\alpha_1+\alpha_2+\alpha_3}{2} & \frac{\alpha_1-\alpha_2-\alpha_3}{2} & \frac{-\alpha_1+\alpha_2-\alpha_3}{2} & \frac{-\alpha_1-\alpha_2+\alpha_3}{2} & 0 & \dots \\ \frac{\alpha_1-\alpha_2-\alpha_3}{2} & \frac{\alpha_1+\alpha_2+\alpha_3}{2} & \frac{-\alpha_1-\alpha_2+\alpha_3}{2} & \frac{-\alpha_1+\alpha_2-\alpha_3}{2} & 0 & \dots \\ \frac{-\alpha_1+\alpha_2-\alpha_3}{2} & \frac{-\alpha_1-\alpha_2+\alpha_3}{2} & \frac{\alpha_1+\alpha_2+\alpha_3}{2} & \frac{\alpha_1-\alpha_2-\alpha_3}{2} & 0 & \dots \\ \frac{-\alpha_1-\alpha_2+\alpha_3}{2} & \frac{-\alpha_1+\alpha_2-\alpha_3}{2} & \frac{\alpha_1-\alpha_2-\alpha_3}{2} & \frac{\alpha_1+\alpha_2+\alpha_3}{2} & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \end{bmatrix}$$

Consider the following rank one matrices of the form $\mathbf{a}\mathbf{a}^T$ -

$$A_1 = \begin{bmatrix} 1 & 1 & -1 & -1 & 0 & \dots \\ 1 & 1 & -1 & -1 & 0 & \dots \\ -1 & -1 & 1 & 1 & 0 & \dots \\ -1 & -1 & 1 & 1 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 1 & -1 & 1 & -1 & 0 & \dots \\ -1 & 1 & -1 & 1 & 0 & \dots \\ 1 & -1 & 1 & -1 & 0 & \dots \\ -1 & 1 & -1 & 1 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 1 & -1 & -1 & 1 & 0 & \dots \\ -1 & 1 & 1 & -1 & 0 & \dots \\ -1 & 1 & 1 & -1 & 0 & \dots \\ 1 & -1 & -1 & 1 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \end{bmatrix}$$

Note that each of the above matrices are PSD. Also observe the following -

$$s'(\alpha, \omega, c') = \frac{\alpha_1}{2} A_1 + \frac{\alpha_2}{2} A_2 + \frac{\alpha_3}{2} A_3$$

Because each $\alpha \geq 0$, and each A is PSD, $s'(\alpha, \omega, c')$ is also PSD. It can be similarly shown that each $s'(\alpha, \omega, c')$ is PSD for all the checks $c' \in C'$. The proof can be easily extended to checks involving only two variables. Hence, all $s'(\alpha, \omega, c')$ are PSD and $\mathcal{S}(\theta, \omega, y) \succeq 0$.

So, this gives an alternative proof to the statement that if LP decoder succeeds, then SDP decoder succeeds.

3.4 Probability of Error is Independent of Transmitted Codeword

The proof is very similar to that for the LP decoder (see, Appendix 3 of Feldman et al. [2005]). The non-trivial part of the proof is to define the *relative solution* and to show that the relative solution is a valid solution to the SDP decoder.

$\mathrm{B}(\mathbf{x}) \subset \Sigma^n$ is the set of received words $\hat{\mathbf{y}}$ that cause the decoding failure, assuming $\mathbf{x}$ was transmitted. $\hat{\mathbf{y}}^1$ is defined as $\hat{y}_i^1 = \hat{y}_i x_i$ for the BSC channel. Define

$$X' = \begin{bmatrix} 1 & \mathbf{x}'^T \\ \mathbf{x}' & \mathbf{x}\mathbf{x}'^T \end{bmatrix}$$

where $\mathbf{x} \in \mathcal{C}_x$.

For any feasible solution Z of the SDP, define the relative solution Z^r with respect to X as the following -

$$Z^r_{i,j} = Z_{i,j} X'_{i,j}$$

Now we need to show that Z^r satisfies all the constraints of the SDP. Note the following -

$$\begin{aligned} Z^r_{i,i} &= Z_{i,i} X'_{i,i} \\ &= 1 * 1 \\ &= 1 \\ \Rightarrow \langle Z^r, E_{i,i} \rangle &= 1 \end{aligned}$$

For a check $c' \in C'$ involving three variables with index i,j and k we have -

$$\begin{aligned} Z_{i+1,j+1} &= Z_{k+1,1} \\ \text{and } X'_{i+1,j+1} &= Z_{k+1,1} \\ \because Z^r_{i+1,j+1} &= Z_{i+1,j+1} X'_{i+1,j+1} \\ \Rightarrow Z^r_{i+1,j+1} &= Z_{k+1,1} X'_{k+1,1} \\ \Rightarrow Z^r_{i+1,j+1} &= Z^r_{k+1,1} \end{aligned}$$

Similarly, the other two constraints of the type $\langle B_{i,c'} Z^r \rangle = 0$ will be satisfied. Hence, all the constraints due to checks involving 3 variables will be satisfied. The reader can easily verify this for the checks with weight 2.

Finally, we're left to prove that Z^r is a PSD matrix. First, define $x'_0 = 1$. Then notice that $X'_{i+1,j+1} = x'_i x'_j \ \forall i, j \geq 0$. Hence, $Z^r_{i+1,j+1} = Z_{i+1,j+1} x'_i x'_j$ or

$$Z^r = \begin{bmatrix} 1 & & & \\ & x'_1 & & \\ & & x'_2 & \\ & & & \ddots \end{bmatrix} Z \begin{bmatrix} 1 & & & \\ & x'_1 & & \\ & & x'_2 & \\ & & & \ddots \end{bmatrix}$$

which is of the form $A^T Z A$ where A is invertible and Z is PSD. Hence, Z^r is also PSD.

The rest of the proof is very similar to the proof for LP decoder. We eventually get that the probability of error for the SDP decoder is independent of the codeword transmitted.

4 Numerical Experiments

To compare the performance of the LP decoder and SDP decoder, we construct a pseudo-random LDPC code of length 60 which has a degree 4 for each check. We use MOSEK to solve the underlying LP and SDP. As can be seen in Figure 1 that performance of SDP decoder is only slightly better than LP decoder.

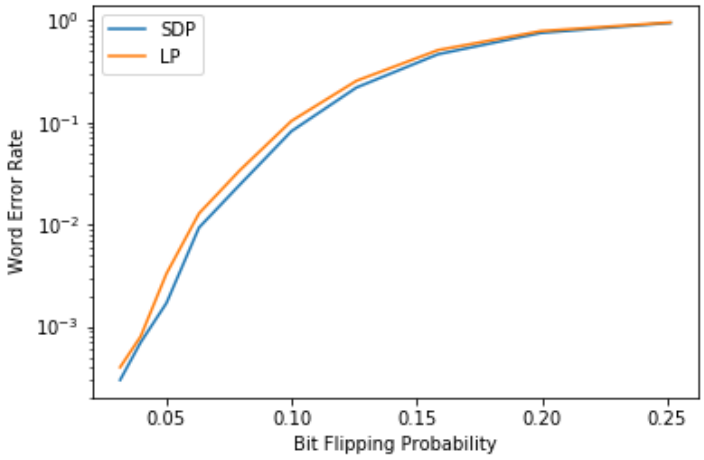

Figure 1: Performance of SDP and LP Decoders

Intuitively, LDPC codes are sparse and it seems that the LP decoder is able to capture the sparse connections well enough; additional constraints in the SDP decoder don't contribute a lot towards improving the performance.

References

Jon Feldman, Tal Malkin, Rocco A Servedio, Cliff Stein, and Martin J Wainwright. Lp decoding corrects a constant fraction of errors. *IEEE Transactions on Information Theory*, 53(1):82–89, 2006.

Elwyn Berlekamp, Robert McEliece, and Henk Van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.

Jon Feldman, Martin J Wainwright, and David R Karger. Using linear programming to decode binary linear codes. *IEEE Transactions on Information Theory*, 51(3):954–972, 2005.