

Chapter 2

A Quasi-Cyclic LDPC Code for GNSS Signal

Yi Yang, Changjian Liu and Xiaoqing Zhang

Abstract Considering that modernized GPS L1C signal utilizes random constructed LDPC as the FEC coding scheme, it's a waste of storage resources because of the irregularity of the parity-check matrix, though owning superb error-control performance. This paper proposed a constructing algorithm for Quasi-Cyclic LDPC code which is suitable for GNSS signal, it avoids the existence of short cycle with the length of 4 in the Tanner graph from the beginning, by limiting the cross-correlation values of sparse sequences. Better performance can be achieved by making the parity-check matrix sparser. Using 802.11n protocol as a reference, parity-check part is double diagonal, which makes the encoding process possible directly with parity-check matrix. Simulation results show this LDPS code performs slightly better than what GPS L1C uses, lower encoding and decoding complexity makes it more suitable for GNSS.

Keywords Satellite navigation · Low-density parity check (LDPC) code · Parity-check matrix · Quasi-cyclic code

2.1 Introduction

Growing number of new ideas are prominently emerging with the development of GNSS signal design [1], including the introducing of forward error correction (FEC) coding such as convolutional coding and LDPC code for the purpose of improving the signal's robustness. Conventional GPS C/A signal utilizes Hamming code to make itself possible to detect errors, but not able to forward error control. Modernized GPS signal generally uses convolutional code as the

Y. Yang (✉) · C. Liu · X. Zhang
Department of Electronics and Information Engineering, Huazhong
University of Science and Technology, Wuhan 430074, China
e-mail: yangyi@hust.edu.cn

FEC code with the coding efficiency of $1/2$, while GPS L1C uses LDPC code which results in higher coding gain in contrast with convolutional code as an exception.

Subframe 2 and subframe 3 of GPS L1C are separately encoded using rate $1/2$ LDPC codes, as a result, there are 1,200 coded bits for subframe 2 and 548 coded bits for subframe 3 [2]. L1C constructs the parity-check matrices randomly which makes them superb at error correcting when codeword is long, but on the other hand, encoding and hardware realization complexities are substantially increased due to the long codeword and irregularity of parity-check and generator matrices. Although GPS uses lower-triangular-matrix-based LDPC code, RU algorithm [3] gives out an encoding scheme with approximately linear complexity, randomly constructed parity-check matrices and relatively high average row weights still make the requirements of storage and calculation a demanding problem.

Structured LDPC code, especially Quasi-cyclic LDPC code is proposed for above reason, which owns lower encoding complexity and makes it easier for hardware realization. Shanbao et al. [4] proposes an algorithm to obtain quasi-cyclic parity-check matrix by finding a series of so-called sparse sequences, but the method for removing 6-cycle in the Tanner graph can just apply to the situation with small number of sequences, what's more, none of the sub-matrices is zero matrix and that has an negative effect on error correction performance. Combining with relevant theory, [5] discusses how to remove 4-cycles and 6-cycles, but not able to explain why there appear zero sub-matrices on the left side of parity-check matrix.

Based on the work of [4], this paper generates several sparse sequences make sure the cross-correlation values between any two arbitrary generated ones are no more than 1, by this way a parity-check matrix is obtained which is dual-diagonal by borrowing ideas from protocol IEEE 802.11n. Finally, better performance can be achieved by setting some of the sub-matrices zero randomly.

2.2 Quasi-Cyclic LDPC Code and its Encoding Algorithm

2.2.1 Characteristics of the Parity-Check Matrix

The characteristic of QC-LDPC code is that there is a number z , we can divide any available codeword with the length of $n * z$ into n pieces of sub-codewords and the resulting codeword is still available if we cyclic shift each sub-codeword by a common length separately, so the corresponding parity-check matrix consists of several sub-matrices with the same size ($z \times z$), assume each sub-matrix is a circulant permutation matrix or a zero matrix, then the parity-check matrix can be expressed as **B**:

$$\mathbf{B} = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,n} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{m,1} & p_{m,2} & \cdots & p_{m,n} \end{bmatrix} \quad (2.1)$$

which we call *basis matrix*. For any $p_{i,j} (-1 \leq p_{i,j} < z)$, it represents a sub-matrix derived by cyclic shifting identity matrix $\mathbf{I}(0)$ by $p_{i,j}$ steps if $p_{i,j} \geq 0$, and represents zero matrix if $p_{i,j} = -1$. It's clearly seen that the performance of codeword is totally determined by basis matrix once codeword length and size of sub-matrix z are given, so it should be meticulously designed and optimized. Wang et al. [6] gives out a theorem on judging if short cycle exists in the parity-check matrix by probing its basis matrix.

Since parity-check matrix can be expressed by its basis matrix which has lower dimension, storage requirements are significantly decreased. From the standpoint of hardware, QC-LDPC encoding process can be easily accomplished by circulating registers and trade off time complexity against space complexity [7].

Using dual-diagonal structure, the parity-check matrix can be directly used to encode thus there is no need to calculate generator matrix. Considering the case coding efficiency is $1/2$, which means $m = n/2$, basis matrix has the following form as (2.2) shows

$$\begin{aligned} \mathbf{B} &= [\mathbf{B}_s | \mathbf{B}_p]_{m \times 2m} \\ &= \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,m} & 1 & 0 & -1 & -1 & \cdots & -1 & -1 \\ p_{2,1} & p_{2,2} & \cdots & p_{2,m} & 0 & 0 & 0 & -1 & \cdots & -1 & -1 \\ p_{3,1} & p_{3,2} & \cdots & p_{3,m} & -1 & -1 & 0 & 0 & \cdots & -1 & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ p_{i,1} & p_{i,2} & \cdots & p_{i,m} & -1 & -1 & -1 & -1 & \cdots & -1 & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ p_{m-1,1} & p_{m-1,2} & \cdots & p_{m-1,m} & -1 & -1 & -1 & -1 & \cdots & 0 & 0 \\ p_{m,1} & p_{m,2} & \cdots & p_{m,m} & 1 & -1 & -1 & -1 & \cdots & -1 & 0 \end{bmatrix} \end{aligned} \quad (2.2)$$

\mathbf{B}_s corresponds to parity-check coefficients for information bits and \mathbf{B}_p corresponds to parity-check coefficients for parity bits, they are both $m \times m$ matrices. The first and last elements in the first column of \mathbf{B}_p are “1”, the second element is “0” and others are “-1”, the rest of \mathbf{B}_p is a $m \times (m-1)$ matrix with elements in its two diagonal lines all being “0” and the rest elements all being “-1”.

2.2.2 Encoding Scheme

We denote information bits as $\mathbf{u} = \{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_m\}$ and parity bits as $\mathbf{v} = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m\}$, the length of each \mathbf{u}_i and \mathbf{v}_i is z , and assume that the sub-matrix $\mathbf{H}_{i,j}$ in the parity-check matrix corresponds to $p_{i,j}$ in the basis matrix. The encoding process based on parity-check matrix is accomplished as follows

1. Calculate intermediate variables $\mathbf{c}_i (i = 1, 2, \dots, m)$ and then \mathbf{v}_1 afterwards

$$\mathbf{c}_i = \left(\sum_{j=1}^{m_b} \mathbf{H}_{i,j} \cdot \mathbf{u}_j^T \right)^T \quad (2.3)$$

$$\mathbf{v}_1 = \sum_{i=1}^{m_b} \mathbf{c}_i. \quad (2.4)$$

2. Update intermediate variables \mathbf{c}_i (just for $i = 1, 2$)

$$\mathbf{c}_i = \mathbf{c}_i + (\mathbf{H}_{i,m+1} \cdot \mathbf{v}_1^T)^T. \quad (2.5)$$

3. Calculate all other parity bits

$$\mathbf{v}_i = \begin{cases} \mathbf{c}_1 & i = 2 \\ \mathbf{c}_{i-1} + \mathbf{v}_{i-1} & i \in [3, m]. \end{cases} \quad (2.6)$$

The encoded codeword is $\{\mathbf{u}, \mathbf{v}\} = \{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_m, \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m\}$ according to above steps, and note that all the additions referred are Modulo 2 additions.

2.3 Construction of Parity-Check Matrix

For most cases decoding algorithms for LDPC code are iterative, the information received by the nodes in the Tanner graph tends to be no more independent thus degrades the decoding performance when cycles exist in the Tanner graph or parity-check matrix. On the other hand, the existence of cycles can benefit the codewords by improving the minimum code distance, for which we should take both factors into consideration in the process of LDPC code design. Generally speaking, 4-cycle plays a negative role for the convergence of decoding process and considered having the worst influence on the performance of LDPC code and that we can achieve fairly good performance by removing it. The method given in this chapter absolutely avoids the existence of 4-cycle.

We will discuss the construction of the left part of parity-check matrix, i.e., what \mathbf{B}_S corresponds since the right part \mathbf{B}_P has been certain. Binary sparse sequence is a sequence with certain number of “1” and the others are all “0”, the cross-correlation of two sequences s_I and s_2 with the length of N is defined as

$$R_{12}(k) = \sum_{i=1}^N s_I(i) \cdot s_2((i+k) \bmod N). \quad (2.7)$$

No 4-cycle exists in the parity-check matrix means the cross-correlation value is less than 2 between any two arbitrary rows if we view each row as a sequence.

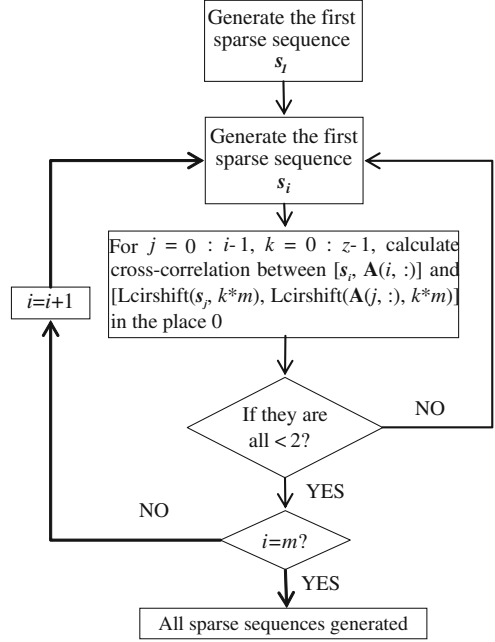
For each element in the $m \times z$ sparse sequence we label it with a number $(0, 1, \dots, m-1)$ which equals to its subscript modulo m and be sure that there is one and only one “1” for all the elements that share a common label, thus there are m “1” in the entire sequence. Z sequences can be derived if we left cyclic shift the mentioned sequence by $k \times m$ steps ($k = 1, 2, \dots, z-1$), which can be used to constitute a matrix with size of $z \times (m \times z)$. It's easy to find out that cross-correlation value $R_{12}(0)$ between any two arbitrary rows is 0 and quasi-cyclic form can be achieved by means of column exchange.

We should generate m sparse sequences which meet above requirements to get the parity-check matrix. The part which \mathbf{B}_P corresponds to also has an effect on the cross-correlation value, that's why we also take it into consideration. The construction process is as follows:

1. First construct a matrix $\mathbf{A} = \text{zeros}(m, m \times z)$ according to \mathbf{B}_P , $\mathbf{A}(i, j) = 1$ if $\mathbf{B}_P(i, j) = 0$ and we also set $\mathbf{A}(1, m \times (z-1) + 1) = 1$ and $\mathbf{A}(m, m \times (z-1) + 1) = 1$ which correspond to the two “1”s in \mathbf{B}_P .
2. Find m sparse sequences which should meet the requirements mentioned as Fig. 2.1 shows, $\text{Lcirshift}(a, b)$ means left cyclic shift sequence a by length of b .
3. Obtain z sequences with each sparse sequence by left cyclic shift by $k \times m$ steps ($k = 0, 1, 2, \dots, z-1$) which also means a $z \times (m \times z)$ matrix can be constructed with one sparse sequence if each row corresponds to a cyclic shifted sequence. A $(m \times z) \times (m \times z)$ matrix can be constructed with m sparse sequences found in step 2, left part of parity-check matrix \mathbf{H}_S can be derived by its column exchange.
4. Concatenate \mathbf{H}_P with \mathbf{H}_S , we get $\mathbf{H} = [\mathbf{H}_S | \mathbf{H}_P]$.

The removal of 4-cycle is ensured by limiting the cross-correlation values and sequences can still be found quickly with computer when m is large since the removal of 6-cycle is not considered. This method results in non-existence of “-1” in the basis matrix which makes the row weight and column weight be both m , the density is not low enough. We can set some of sub-matrices in \mathbf{H}_S as zero matrix to solve this problem which has been verified to be an efficient solution.

Fig. 2.1 Flowchart of generating all sparse sequences



2.4 Simulation Results

We set codeword length to 1,200 bits and coding efficiency 1/2 which is the same as one of L1C scheme, size of sub-matrix is set to 60×60 which means $z = 60$ and $m = 10$. The final parity-check matrix is obtained by set 60 sub-matrices in the left part of \mathbf{H}_S to zero matrices randomly, (2.8) gives out the corresponding basis matrix as an example.

$$\mathbf{B} = \begin{bmatrix} -1 & -1 & 9 & 25 & -1 & -1 & -1 & 51 & 20 & -1 & 1 & 0 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ 16 & -1 & -1 & -1 & 50 & 55 & -1 & 31 & 41 & -1 & 0 & 0 & 0 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & 26 & -1 & -1 & -1 & 12 & -1 & -1 & -1 & 7 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & 12 & -1 & 43 & -1 & -1 & 31 & -1 & 15 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 \\ 41 & -1 & -1 & -1 & -1 & -1 & 3 & 28 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 \\ 40 & -1 & -1 & 43 & 38 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 \\ -1 & -1 & 25 & 37 & 29 & -1 & -1 & -1 & 34 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 \\ -1 & -1 & -1 & -1 & 11 & -1 & -1 & 37 & 17 & 38 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 \\ -1 & 56 & 13 & -1 & -1 & 5 & -1 & 47 & 33 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 \\ -1 & 28 & 42 & -1 & -1 & 6 & 23 & -1 & -1 & -1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 \end{bmatrix} \quad (2.8)$$

LLR-BP algorithm is used as the decoding method to verify the performance of constructed LDPC code. Figure 2.2 gives out an example showing how bits error number changes with iteration times when SNR equals 1 dB which turns convergence successfully after 12 iteration steps, the parity-check matrix is constructed by the method as last chapter describes. This curve shows the contribution FEC makes for GNSS visually.

Fig. 2.2 An example showing relationship between bit error number and iteration times using LLR-BP decoding algorithm

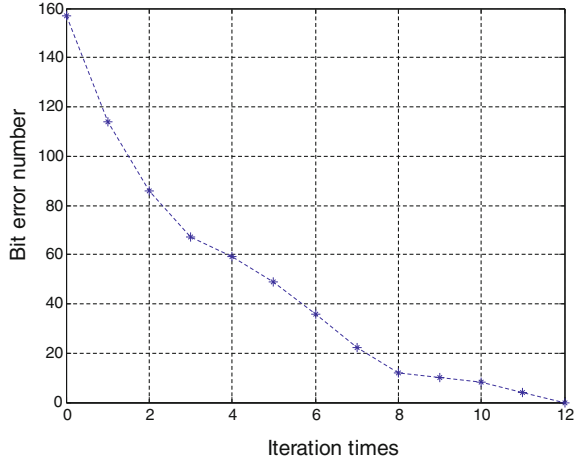
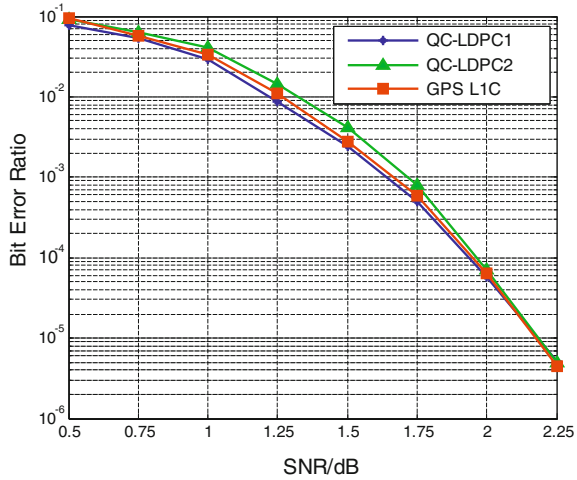


Fig. 2.3 BER performance comparison between QC-LDPC codes and GPS L1C LDPC codes



We simulated the bit error ratio (BER) performance of LDPC code used by GPS L1C signal, an optimized result [5] gives and our result for comparison to verify its performance. BPSK modulation is assumed to be used, i.e., “1” is mapped into “−1” and “0” is mapped into “1”, and the channel is assumed to be AWGN. The maximum iteration times is set to 50 for each decoding process.

Figure 2.3 presents their BER performance curves, it’s clearly shown that our result (blue curve) outperforms what [5] gives (green curve) and is slightly better than GPS L1C scheme when SNR is low, so it can be considered at least meet the performance of GPS L1C. Storage requirements are significantly reduced in the process of encoding and decoding with QC-LDPC scheme while achieving not lower performance.

2.5 Concluding Remarks

Considering the power of data channel is reduced by at least 3 dB due to the fact that generally half or more power of modernized GNSS signal is allocated to pilot channel, we can choose forward error correction (FEC) coding to make up for the loss of robustness of data demodulation due to power allocation. This paper briefly introduces the encoding scheme based on QC-LDPC code followed by proposing a method for constructing parity-check matrix without 4-cycle. Simulation result shows that its decoding performance can be as good as what GPS L1C uses. Further research can be focused on studying specific algorithm to “lighten” the check matrix which was randomly done for this paper.

References

1. Hu X, Tang Z et al (2009) Analysis on design principles of GPS and Galileo signal structure. *Syst Eng Electron* 31(10):2285–2293
2. GPS Navstar Joint Program Office. Navstar GPS space segment/user segment L1C interfaces [S], Draft IS-GPS-800, 2006
3. Richardson TJ, Urbanke RL (2001) Efficient encoding of low-density parity-check codes. *IEEE Trans Inform Theory* 47(2):P638–P656
4. Shanbao H, Liu CH, Yiming L (2009) Application of LDPC codes in satellite navigation systems. *Spacecraft Eng* 18(3):72–76
5. Qian H, Li GX, Chang J (2011) Application of quasi-cycle low-density parity check codes with high performance to satellite navigation signals. *J Comput Appl* 31(4):1145–1147
6. Wang Y, Yedidia JS, Draper SC (2008) Construction of high-girth QC-LDPC codes. 2008 5th International Symposium on Turbo Codes and Related Topics, Lausanne, 2008, pp 181–185
7. Li Z, Chen L, Zeng L et al (2006) Efficient encoding of quasis-cyclic low-density parity-check codes. *IEEE Trans Comm* 54(1):P71–P81

China Satellite Navigation Conference (CSNC) 2013
Proceedings
Satellite Navigation Signal System, Compatibility &
Interoperability • Augmentation & Integrity Monitoring •
Models & Methods
Sun, J.; Jiao, W.; Wu, H.; Shi, C. (Eds.)
2013, XV, 566 p., Hardcover
ISBN: 978-3-642-37403-6