A Novel Algorithm for Spectral Shaping of Binary Data Streams

Peter Vámos

***Abstract*—By generalizing the accumulated charge concept, we introduce a new class of constraints, the generalized charge constraint to control the spectral properties of a binary sequence. The new constraint limits the level at the output of a digital filter, diminishing those spectral components of the channel sequence being enhanced by the filter. A suitable coder structure, a feedback controlled bit stuff encoder is suggested to implement the new constraint. We demonstrate the spectral shaping property of the new coder structure and derive an approximate formula for the spectrum of the output binary signal. We also show that the coder performs a sigma-delta-like operation and the method is capable of implementing spectral and run-length constraints simultaneously. As a demonstration, we present a few particular spectral characteristics shaped by different examples of loop filters.**

Index Terms: Channel coding, Modulation codes, Run-length codes, Signal processing, Sigma-delta modulation.

I. INTRODUCTION

In many applications it is either impossible or at least difficult to shape the digital pulse. Typical examples are digital optical transmission and magnetic recording. In such applications it is important to perform an appropriate spectral shaping of the binary data stream [1], [2], [3]. Moreover, there are usually additional constraints on the code, such as a bound of the number of consecutive identical symbols, i.e., the run-length. The run-length upper bound (k constraint) ensures the reliable clock recovery, while establishing a lower bound (d constraint) diminishes the intersymbol interference, both practically important [4]. The d and k constraints together are referred to as run-length limiting or RLL constraints.

The most code constructions in the literature concentrate on dc-suppression [Chapters 7 and 8 of 2], and one can find only a few general purpose spectrum shaping algorithm, and even less capable to control the spectrum and the run-length simultaneously. In 1987 Marcus and Siegel published an algorithm which can produce spectral nulls at rational submultiples of the signalling frequency [5].

The guided scrambling algorithm developed by Fair et al. [6] makes dc-suppression by adding one or more redundant bits to the blocks of data stream. It minimize the accumulated charge on the output of a scrambler over the possible values of the redundant bits. Applying the weighted running digital sum (WRDS) concept introduced in present paper, the algorithm can be used even for general purpose spectral shaping. It is also claimed that algorithm limits the run-length as well, however, it is carried out partly by limiting the block length, and the k constraint can not be prescribed explicitly. It makes further difficulties to keep the run-length limit at the block boundaries, and guided scrambling is not suitable for imposing d constraint at all.

The algorithm proposed by Cavers and Marchetto can be taken as special case of guided scrambling with enhanced spectral shaping properties [7]. It minimizes (or maximizes) the output of a digital FIR filter representing the spectral constraint by inverting some data blocks along the filter. The inversion is marked on flag bits added to each block. By this method, however, can not be handled RLL constraints at all, and it uses the computationally expensive Viterbi algorithm for the encoding.

The bit stuffing approach has been applied for decades to control the run-length in binary sequences. The well known HDLC (High-level Data Link Control) protocol inserts a '0' bit after each sequence of five consecutive '1' bits [8]. In 1993 Bender and Wolf suggested a bit stuffing algorithm for generating run-length limited (RLL) sequences with spectral null at zero frequency [9]. However, their solution can scarcely be applied in practice due to its bent for infinite error propagation caused by the infinite memory of the decoder. Next to the error propagation, what can be kept under control by limiting the coder's memory, the only drawback of the bit stuff algorithm is that it requires buffering to keep the transmission rate constant.

In the second half of the 2000s many authors published improvements to the bit stuffing algorithm for coding (d, k) constrained channels [10], [11], [12]. The rates of these improved algorithms are very close to the channel capacity, and, in some specific cases, even they reach it. However, all they use that the bound of the current run-length is constant, what doesn't hold for charge and generalized charge constrained codes [13] presented in this paper.

In Section II-A of this paper we generalize the accumulated charge concept used for generating dc-suppressed code spectrum and introduce a new class of constraints the generalized charge constraint. The new constraint limits the level at the output of a digital filter, so the spectral requirements can be described easily. A feedback controlled bit stuff encoder with loop filter is suggested in Section II-B to implement the new constraint. This structure can perform a very effective and flexible spectrum shaping, and moreover, it can also control the run-length of the output bit stream. The flexibility is due to the digital loop filter, that can be implemented by inexpensive and readily available DSP components. In contrast to guided scrambling, the coder controls the output sequence

Manuscript received 19 August 2011, revised 8 February, accepted for publication 8 April, 2012.

The research was supported by the Foundation of Industry for Modern Technical Education (IKMA)

Peter Vámos is with the Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Budapest H-1521, Hungary. email: *vamos@tmit.bme.hu*

continuously, what makes easy to implement both k and d constraints, allows the application of IIR filters, and enhances the effectiveness of spectral shaping.

In Section II-C we give a brief analysis of the coder demonstrating its spectral shaping property and supply an approximate formula for the spectrum of the output binary signal in case of i.i.d. input. We also show that the coder performs a sigma-delta converter-like operation. We present that the coder can enforce spectral and run-length constraint simultaneously, and we extend the approximate formula for the spectrum for the case when k and spectral constraint are applied together.

In Section III we give a detailed analysis of coders with FIR loop filters. The performance of those coders can be described as finite sate machine (FSM) with the corresponding discrete Markov model. We also deal with the case when explicit run-length constraints are imposed next to the spectral one. As an example, we scrutinize an actual coder scheme with low-pass window filter generating dc-suppressed code for ac-coupled channels.

In Section IV coders with IIR loop filters are covered. These coders have an infinite state space and the corresponding Markov chain becomes unstable. So either we use functional equations for the description, or we approximate the actual Markov process with a finite space Markov chain. As example, we present dc-suppression, spectral notches and their combination shaped by different IIR loop filters.

II. The Coder's Structure and Performance

A. Generalization of the Accumulated Charge Concept

The most common application of spectral shaping is coding for ac-coupled channel. Those channels have a low frequency cut-off which affects the low frequency components of the code, causing a slow fluctuation in level at the receiver. To avoid this fluctuation, the coded sequence should be poor in low frequency spectral components. It is carried out by charge constrained codes when the accumulated charge of the channel sequence $Y_1, Y_2, \ldots$ is limited:

$$C_n = \sum_{i=0}^{n-1} Y_{n-i} \quad \text{and} \quad |C_n| \le c \ \text{ for any } \ n \in \mathbf{N}. \qquad (1)$$

In binary case ($Y_i \in \{-1, +1\}$) the sequence C is commonly called as running digital sum (RDS). One can see that the RDS is generated by a lowpass filter:

$$C(z) = \frac{1}{1 - z^{-1}}\, Y(z), \qquad (2)$$

where $Y(z)$ is the z- or discrete Fourier transform of sequence Y with $z = \exp(j2\pi f/f_0)$ and f_0 stands for the bit frequency. By limiting the RDS, we limit the level at the output of a lowpass filter. So, low frequency components of the channel sequence enhanced by the filter will be suppressed. (Actually, the filter in (2) has an infinite enhancement around the zero frequency, consequently, Y must have zero spectral density at zero frequency if RDS is limited. Pierobon [14] has proved the limited RDS is also a necessary condition.) It implies the idea using filters with other characteristics to form an RDS like quantity will also shape the spectrum according to the applied filter. For this purpose, let us generalize the RDS concept by introducing the weighted running digital sum (WRDS):

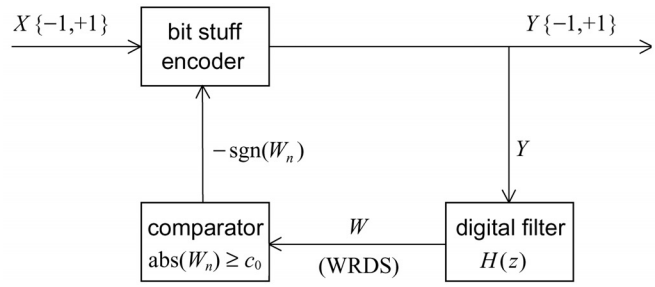

Fig. 1 Feedback controlled bit stuff encoder.

Definition *The weighted running digital sum is the convolution of a binary sequence $Y_i \in \{-1, +1\}$ and a sequence of given constants $h_i \in \mathbf{R}$:*

$$W_n = \sum_{i=0}^{n-1} h_i\, Y_{n-i}, \quad \text{or} \quad W(z) = H(z)\, Y(z). \qquad (3)$$

WRDS makes direct connection between the binary sequence and its spectrum. By limiting the WRDS we can enforce spectral constraints on the code. Those codes with limited WRDS we will refer to as generalized charge constrained or spectrum constrained codes.

B. Principle of the Coding Algorithm

For coding WRDS limited channels we are using the coder structure in Fig. 1. The coder is actually a feedback controlled bit stuff encoder. The bit stuff encoder has two states: It either transmits a bit from the input to the output or inserts a redundant bit. The bit stuffing is controlled by the feedback loop. Whenever the level at the filter's output reaches a given threshold c_0, the bit stuff encoder inserts a bit with opposite sign to the filter's output signal:

$$Y_{n+1} = \begin{cases} X_{m+1}, & \text{if} \quad |W_n| < c_0; \\ -\mathrm{sgn}(W_n), & \text{if} \quad |W_n| \ge c_0, \end{cases} \qquad (4)$$

where $X_i, Y_i \in \{-1, +1\}$, and $W_n = \sum_{i=0}^{\infty} h_i\, Y_{n-i}$. The indices of the input (X) and output (Y) sequences are different because of the previously stuffed bits: $n = m + s_n$, where s_n stands for the number of stuffed bits till Y_n. For threshold c_0 it must hold that

$$\begin{aligned} &c_0 > c_m = \min_{\epsilon_i} \left|\textstyle\sum \epsilon_i\, h_i\right|, \quad \epsilon_i \in \{-1, +1\}; \\ &c_0 \le c_M = \textstyle\sum |h_i|. \end{aligned} \qquad (5)$$

For $c_0 \le c_m$ the rate would be zero (no information could be transmitted), while for $c_0 > c_M$ the rate would be 1 (no spectral shaping could be performed). The above structure works actually as a negative feedback with loop filter. The coder tries to keep low the output of the filter $\widetilde{H}(z) = 1 + z^{-1}H(z)$. The spectral components enhanced by the filter will be dominant in the control of the bit stuff encoder, so the coder's interventions are diminishing their power.

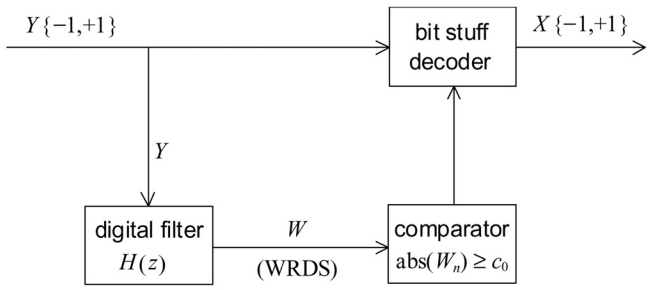

Fig. 2 The decoder's scheme.

The decoding can be performed with a similar, but feed-forward structure in Fig. 2. Whenever the forward filter's output reaches or exceeds the threshold the decoder removes a bit from the stream. Errors in transmission can cause additional errors during the decoding. In order to limit this error propagation we should limit the decoder's memory:

$$\sum |h_i| < \infty\,, \tag{6}$$

i.e., all the poles of the loop filter should be outside the unit circle. It implies that only finite but arbitrarily large suppressions can be realized in the spectrum. The condition (6) ensures that the error propagation in the decoder always remains finite [15]. Further improvement can be reached in error propagation if we diminish the error probability at the bit stuff decoder's input by the application of an outer error correcting code with a Bliss scheme [16], [17].

Keeping the WRDS high by applying the coding rule

$$Y_{n+1} = \begin{cases} X_{m+1}, & \text{if} \quad |W_n| \geq c_0; \\ \operatorname{sgn}(W_n), & \text{if} \quad |W_n| < c_0, \end{cases}$$

instead of (4) can shape the spectrum as well. In this case the spectral components enhanced by the filter $\widetilde{H}(z)$ will be enhanced in the output signal too, so the spectral density of the output will emulate the filter's characteristics. This concept can be useful when the loop filter suitable for the desired spectrum doesn't ensure the stable performance of coder using coding rule (4).

C. Demonstration of the Spectral Shaping Property

To demonstrate the spectral shaping property, let us suppose that the input sequence X is a series of independent and identically distributed (i.i.d.) random variables, i.e. the input is a binary white noise. Under this circumstance, as far as the statistical properties of the output signal are concerned, there is no matter whether the coder inserts a bit or overwrites the incoming one, so the following coding rule can be set instead of (4):

$$Y_{n+1} = \operatorname{sgn}(X_{n+1} - \frac{1}{c_0} W_n).$$

To analyze the corresponding nonlinear system in Fig. 3.a, let us replace the one-bit quantizer by a quantization noise generator (Fig. 3.b). On the basis of the figure we can write:

$$\left.\begin{aligned} Y(z) &= X(z) - \tfrac{z^{-1}}{c_0} W(z) + Q(z) \\ W(z) &= H(z)\, Y(z) \end{aligned}\right\}$$

Expressing $Y(z)$, for the z-transform of the output signal we get:

$$Y(z) = \frac{X(z) + Q(z)}{1 + \frac{z^{-1}}{c_0} H(z)}. \tag{7}$$

Now, supposing that process Q is an uncorrelated white noise [18], which is more or less satisfied while $0 < (c_0 - c_m)/(c_M - c_m) \ll 1$, i.e. when the coder performs definitely, for the spectral density of the output we have

$$S_Y(f) \approx \frac{1}{\left|1 + \frac{e^{-j2\pi f/f_0}}{c_0} H(e^{-j2\pi f/f_0})\right|^2}. \tag{8}$$

The above formula is an approximation, but one that describes well the main features of the code spectrum in most cases, and thus good basis for the coder's design.

To determine the suitable loop filter characteristics for the desired code spectrum, let us fix c_0 as 1. We can do it without loss of generality since the threshold's value can be included into the loop filter's characteristics, and it only trims the dynamics of the output spectrum, as can be seen in Fig. 8. So, the main features of the output spectrum are determined by the characteristics $\widetilde{H}(z) = 1 + z^{-1} H(z)$, and the loop filter can be designed on the basis of

$$H(z) = (\widetilde{H}(z) - 1)/z^{-1}.$$

From (7) one can see that the same filtering is applied both for the input signal X and the quantization noise Q. Using the notations

$$F(z) = \frac{1}{1 + \frac{z^{-1}}{c_0} H(z)} \quad \text{and} \quad X'(z) = F(z)\, X(z),$$

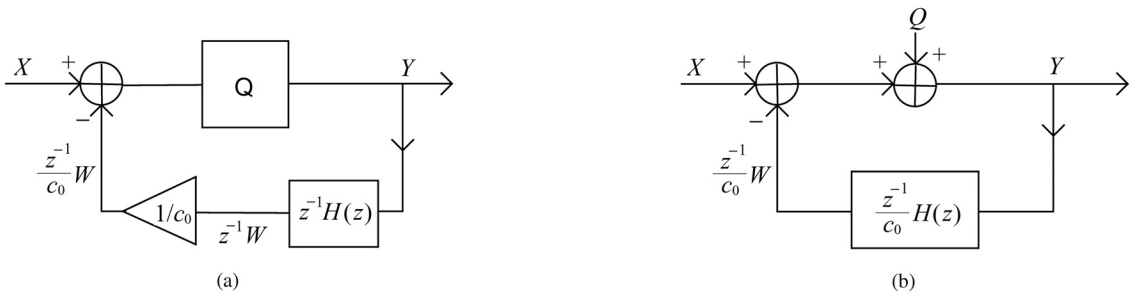

Fig. 3 The coder's nonlinear equivalent circuitry (a); and its linearization (b).

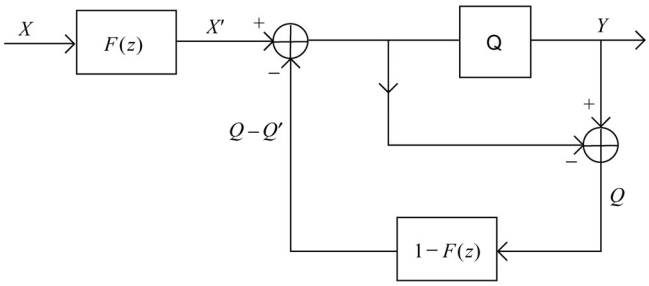

Fig. 4 The sigma-delta modulator-like equivalent of the coder structure.

as well as introducing the re-quantization noise as

$$Q'(z) = Y(z) - X'(z) = F(z)\,Q(z),$$

we can transform the layout in Fig. 3. The feedback quantizer on the right side of the yielding structure in Fig. 4 is exactly the circuitry suggested by Spang and Schultheiss for shaping the spectrum of the quantization noise [19], are widely used these days. It demonstrates the coder's performance well: the input binary signal is filtered according to the spectral requirements, then the resulted signal, in general with a continuous amplitude distribution, is re-quantized by a quantizer which also shapes the quantization noise spectrum in the same manner. We call the attention of the reader for the structural similarity of the layout in Fig. 4 and sigma-delta converters [20].

D. Mixing the Constraints

For a process with finite set of values there can be defined a corresponding run-length process:

Definition *A run is a substring of identical symbols. Let us define the transition times of the process Y as*

$$t_i = \min\{j > t_{i-1} | Y_j \neq Y_{j-1}\} \quad \textit{and} \quad t_0 = 0.$$

Then the run-lengths are given as the differences $T_i = t_i - t_{i-1}$, and the process $T(Y)$ is called as the run-length process associated with Y.

In most applications it is also important to limit the run-length [1], [2]. In addition to some spectral constraint, now let us impose a (d,k) constraint as well upon the code, so no runs shorter than $d+1$ bits and longer than $k+1$ bits are allowed. ($d<k$ is always required.) To implement these constraints we should insert additional feedback loops into the coder for monitoring the run-length. The coding rule will be the following:

$$Y_{n+1}=\begin{cases} -\mathrm{sgn}(W_n), & \text{if } |W_n| \geq c_0; \text{ (spectral constraint)} \\ -Y_n, & \text{if } \left|\sum_{i=0}^{k} Y_{n-i}\right| = k+1; \text{ (k constraint)} \\ Y_n, & \text{if } \left|\sum_{i=0}^{d} Y_{n-i}\right| < d+1; \text{ (d constraint)} \\ X_{m+1}, & \text{otherwise.} \quad \text{(no stuffing)} \end{cases} \tag{9}$$

Some kind of spectral constraints may occasionally clash with one or other run-length constraints, and they are forcing the bit stuff encoder inserting bit with different sign at the same time. This can be either prevented by imposing an auxiliary constraint upon the code, or resolved by setting priorities for the constraints. An example for the former solution is presented in Section III-D when a low-pass window-filter is applied in the feedback loop to form a dc-suppressed (d,k) constrained code. When priorities are set, usually it is advisable to order higher priority to the run-length constraints since those are more crucial if they are set and generally the false interventions will not deteriorate the spectrum too much.

From (9) one can see that we are using low-pass FIR loop filters for monitoring the run-length both for d and k constraints. Moreover, k constraint is implemented with a coding rule that complies with (4), which maintains the WRDS low, similarly to one applied for the spectral constraint. It implies that when only k constraint is applied with spectral constraint, in case of independent binary white noise input, the bit stuff encoder can be replaced with summing circuits and one-bit quantizers, similarly as we did in the previous section, so the output spectrum can be estimated as

$$S_Y(z) \approx \frac{1}{\left|1 + \frac{z^{-1}}{c_0} H(z) + \frac{z^{-1}}{k+1} \frac{1-z^{-(k+1)}}{1-z^{-1}}\right|^2}. \tag{10}$$

The above formula is less accurate than (8) due to the fact that for k constraint the bit stuff threshold is set to $k+1$, i.e. the condition $(c_0 - c_m)/(c_M - c_m) \ll 1$ is barely satisfied since the loop works at the performance limit.

E. Coding Biased Sources

So far we have tacitly supposed that the input sequence is unbiased, i.e., the probabilities $p = \Pr(X = +1)$ and $q = \Pr(X = -1)$ are both 1/2. If this is not satisfied (or we have no information about it), the input signal has (or might have) a discrete component in dc proportional to the square of the bias $p-q$. Since the coder can produce only finite suppression, the discrete components can not be fully suppressed, and moreover, they stimulate the coder for unnecessarily large number of interventions diminishing the code rate. The usual solution to this problem is the precoding of the biased source [2], [21]. The precoded signal X' is defined as $X'_m = X_m\, X'_{m-1}$, $(X, X' \in \{-1, +1\})$. With the mapping $+1 \to 0$ and $-1 \to 1$ one can see that the precoding process is a mod 2 integration: $X'_m = X_m \oplus X'_{m-1}$. If the input process is i.i.d., the output is an i.i.d. run-length process with geometrical distribution: $\Pr(T_i(X') = n) = qp^{n-1}$ for any $i \in \mathbf{N}$, which has really no discrete spectral components. When $p = q = 1/2$, the statistical properties of the output signal are the same as that of the input.

For the sake of simple implementation, we have integrated the precoder with the bit stuff encoder by changing the coding rule of "no stuffing" cases using $Y_{n+1} = X_{m+1}\, Y_n$ instead of $Y_{n+1} = X_{m+1}$ in coding rules (4) and (9):

$$Y_{n+1} = \begin{cases} X_{m+1} Y_n, & \text{if } |W_n| < c_0; \\ -\mathrm{sgn}(W_n), & \text{if } |W_n| \geq c_0. \end{cases} \tag{11}$$

The new coding rule will continue the current run with probability p (when "+1" inputted), and will start a new one

with probability q (when "-1" inputted) when no stuffing is applied. However, when the bias is non-zero, the precoded signal will not be uncorrelated, so (8) and (10) can not be used to estimate the output signal's spectrum.

III. CODING WITH FIR FILTERS

According to coding rules (4) and (11) the output is determined by the WRDS and the input. For coders applying FIR filters, the momentary WRDS W_n is given by the sequence $Y_{n-r}^{n} = Y_{n-r}, Y_{n-r+1}, \ldots, Y_n$, where r is the filter's order. Since Y is a binary sequence, the coder may have only 2^{r+1} states, so it can be described as FSM with the corresponding state transition matrix $\mathbf{Q}$. Let s_i denote a particular state of the coder, then the elements of $\mathbf{Q}$, i.e. the transition probabilities are given as $q_{i,j} = \Pr(Y_{n-r}^{n} = s_j \mid Y_{n-r-1}^{n-1} = s_i)$.

A. The Special Properties of the Transition Matrix

For the analysis we will use the following symmetry properties of the transition matrix.

Property 1 *Let denote N the number of internal states of the coder and $\bar{s}_i$ the bitwise inverse of the state s_i Then with the labelling*

$$\bar{s}_i = s_{N+1-i}, \quad (i = 1, 2, \ldots, N) \tag{12}$$

the coder's transition probability matrix is centrosymmetric, i.e., $q_{i,j} = q_{N+1-i,N+1-j}$, or with the exchange matrix $\mathbf{J}$ having ones only on the reverse diagonal:

$$\mathbf{J}\,\mathbf{Q}\,\mathbf{J} = \mathbf{Q}. \tag{13}$$

Proof: The states s_i and $\bar{s}_i$ have WRDS with the same magnitude but with opposite sign. The coding rule (11) implies that if s_i transits into s_j for a given input, then $\bar{s}_i$ transits into $\bar{s}_j$ for that very same input. So we can write:

$$\begin{aligned} q_{i,j} &= \Pr(Y_{n-r}^{n} = s_j \mid Y_{n-r-1}^{n-1} = s_i) \\ &= \Pr(Y_{n-r}^{n} = \bar{s}_j \mid Y_{n-r-1}^{n-1} = \bar{s}_i) \\ &= \Pr(Y_{n-r}^{n} = s_{N+1-j} \mid Y_{n-r-1}^{n-1} = s_{N+1-i}) \\ &= q_{N+1-i,N+1-j}\,. \end{aligned} \tag{14}$$

■

(13) implies that the transition matrix can be given in the following form [22]:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2\mathbf{J} \\ \mathbf{J}\mathbf{Q}_2 & \mathbf{J}\mathbf{Q}_1\mathbf{J} \end{bmatrix}. \tag{15}$$

Property 2 *It is a necessary and sufficient condition for centrosymmetry if the matrix has two invariant subspaces orthogonal to each other. An even one ($\mathcal{E}$) consisting of $\mathbf{v}_\mathrm{e} = [\mathbf{v},\ \mathbf{v}\mathbf{J}]$ even, and an odd one ($\mathcal{O}$) consisting of $\mathbf{v}_\mathrm{o} = [\mathbf{v},\ -\mathbf{v}\mathbf{J}]$ odd vectors.*

Proof: The necessity can be proven with the help of decomposition (15):

$$\mathbf{v}_\mathrm{e}\mathbf{Q} = [\mathbf{v}(\mathbf{Q}_1+\mathbf{Q}_2),\ \mathbf{v}(\mathbf{Q}_1+\mathbf{Q}_2)\mathbf{J}] = [\mathbf{v}\mathbf{Q}_\mathrm{e},\ \mathbf{v}\mathbf{Q}_\mathrm{e}\mathbf{J}]; \tag{16.a}$$

$$\mathbf{v}_\mathrm{o}\mathbf{Q} = [\mathbf{v}(\mathbf{Q}_1-\mathbf{Q}_2),\ -\mathbf{v}(\mathbf{Q}_1-\mathbf{Q}_2)\mathbf{J}] = [\mathbf{v}\mathbf{Q}_\mathrm{o},\ -\mathbf{v}\mathbf{Q}_\mathrm{o}\mathbf{J}], \tag{16.b}$$

where $\mathbf{Q}_\mathrm{e} = \mathbf{Q}_1 + \mathbf{Q}_2$ and $\mathbf{Q}_\mathrm{o} = \mathbf{Q}_1 - \mathbf{Q}_2$ are the equivalent transformations of the $N/2$ dimensional reduced subspaces.

The sufficiency follows from that for any $\mathbf{v} \in \mathcal{E} \cup \mathcal{O}$: $\mathbf{v}\mathbf{Q} = \mathbf{v}\mathbf{J}\mathbf{Q}\mathbf{J}$, that is, $\mathbf{Q}$ and $\mathbf{J}\mathbf{Q}\mathbf{J}$ are equivalent. ■

There are two important corollaries of the above properties:

- From (14) one can see that each state can be joined with its inverse, due to the symmetry. Then the state transition probability matrix reads as $\mathbf{Q}_1 + \mathbf{Q}_2 = \mathbf{Q}_\mathrm{e}$.
- Since a matrix and its any power have the same eigenvectors, if $\mathbf{Q}$ is centrosymmetric, $\mathbf{Q}^{\pm n}$ is alike, and $(\mathbf{Q}^{\pm n})_\mathrm{e} = \mathbf{Q}_\mathrm{e}^{\pm n}$ and $(\mathbf{Q}^{\pm n})_\mathrm{o} = \mathbf{Q}_\mathrm{o}^{\pm n}$.

Property 3 *Labelling the states according to the last output bit Y_n too as*

$$\begin{aligned} 1 \quad &\le i \le N/2, \quad \text{if} \quad Y_n = +1; \\ N/2 < i \le \ &N, \quad \text{if} \quad Y_n = -1; \end{aligned} \tag{17}$$

matrices $\mathbf{Q}_1$ and $\mathbf{Q}_2$ have actual physical meaning: $\mathbf{Q}_1$ stands for repeating the last input bit, i.e. continuing the current run, while $\mathbf{Q}_2$ stands for adding an inverse bit to the last one starting a new run with opposite sign. According to coding rule (11), it implies that any non-zero and non-one elements of $\mathbf{Q}_1$ and $\mathbf{Q}_2$ are p and q respectively.

Commonly, applying an order r FIR loop filter, the most plausible labelling satisfying both (12) and (17) is the lexicographical ordering of the filter's states:

$$i = \sum_{k=0}^{r} (1 - Y_{n-k})\, 2^{r-k-1} + 1,$$

$$\text{i.e.,} \quad s_1 = [+1, \ldots, +1], \ldots, s_{2^{r+1}} = [-1, \ldots, -1].$$

B. The Properties of Bit-Stuff Generated Finite State Codes

The redundancy of a bit stuff encoder stems from stuffings, so the coder's rate can be calculated from the stuffing probability P_stuff:

$$R = 1 - P_\mathrm{stuff}.$$

The stuffing probability is given by the sum of stationary probabilities of states where stuffing is applied:

$$P_\mathrm{stuff} = \sum_{|\mathrm{WRDS}(s_i)| \ge c_0,\, i \le \frac{N}{2}} \pi_i,$$

where π_i is the stationary probability of state s_i, i.e., the element of eigenvector $\boldsymbol{\pi}_\mathrm{e}$ of $\mathbf{Q}_\mathrm{e}$ associated with the maximal eigenvalue 1. (Due to the symmetry the stationary distribution $\boldsymbol{\pi}$ of $\mathbf{Q}$ must be element of $\mathcal{E}$: $\boldsymbol{\pi} = \frac{1}{2}[\boldsymbol{\pi}_\mathrm{e}, \boldsymbol{\pi}_\mathrm{e}\mathbf{J}]$; so it is determined by $\mathbf{Q}_\mathrm{e}$ only.)

If the coding algorithm is greedy, i.e., it can generate all the possible sequences obeying the given constraint, the set of edges of the coder's and the constrained channel's state transition graph are identical. So, one can get the constrained channel's adjacency matrix from the transition probability matrix by substituting ones in places of its nonzero elements: $\mathbf{A} = [\mathbf{Q} \neq 0]$. Since $\mathbf{A}$ is centrosymmetric too, and $\mathbf{A}_\mathrm{e} = [\mathbf{Q}_\mathrm{e} \neq 0]$ is non-negative, so it comes into the maximal

eigenvalue $\lambda_{\max}$ of $\mathbf{A}$, which determines the channel capacity given as $C = \log_2 \lambda_{\max}$ [2], [23].

The spectral density of a Markov chain is defined as the Fourier transform of the autocorrelation:

$$S(f) = \sum_{k=-\infty}^{\infty} R(k) e^{-j2\pi k f / f_0} \tag{18}$$

So, to get the spectral density, first we should determine the autocorrelation $R_Y(k) = \mathrm{E}(Y_n Y_{n+k})$. Using that the coder's states are ordered such a manner that $Y_n = +1$ for $S_1 \dots S_{\frac{N}{2}}$ and $Y_n = -1$ for $S_{\frac{N}{2}+1} \dots S_N$, and moreover, the symmetry of the stationary distribution $\boldsymbol{\pi}$, the autocorrelation can be written as

$$R_Y(k) = \frac{1}{2} \left[\boldsymbol{\pi}_{\mathrm{e}}, -\boldsymbol{\pi}_{\mathrm{e}}\mathbf{J}\right] \mathbf{Q}^{|k|} \begin{bmatrix} \mathbf{1} \\ -\mathbf{1} \end{bmatrix},$$

where $\mathbf{1}$ denotes the full-of-one vector $[1, 1, \dots, 1]^*$. Since the vector on the left is element of $\mathcal{O}$, according to (16.b), the autocorrelation can be expressed with $\mathbf{Q}_{\mathrm{o}}$:

$$R_Y(k) = \frac{1}{2} \left[\boldsymbol{\pi}_{\mathrm{e}}\mathbf{Q}_{\mathrm{o}}^{|k|}, -\boldsymbol{\pi}_{\mathrm{e}}\mathbf{Q}_{\mathrm{o}}^{|k|}\mathbf{J}\right] \begin{bmatrix} \mathbf{1} \\ -\mathbf{1} \end{bmatrix} = \boldsymbol{\pi}_{\mathrm{e}}\mathbf{Q}_{\mathrm{o}}^{|k|}\mathbf{1}. \tag{19}$$

Substituting (19) into (18), for the spectrum we get:

$$S_Y(z) = \boldsymbol{\pi}_{\mathrm{e}}[(\mathbf{I} - z\mathbf{Q}_{\mathrm{o}})^{-1} + (\mathbf{I} - z^{-1}\mathbf{Q}_{\mathrm{o}})^{-1} - \mathbf{I}]\,\mathbf{1}.$$

C. Taking the Run-Length into Account

To describe the simultaneously spectrum and (d, k) constrained channel, let us set out from the state transition graph of the RLL channel in Fig. 5. The labelling of the vertices corresponds to the current run-length. Taking into account the spectral constraint, we should use a hyper graph: Each

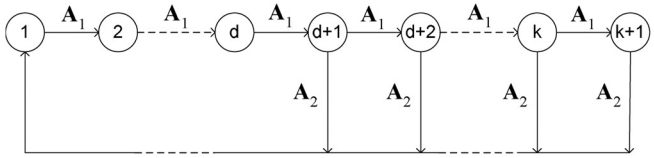

Fig. 5 The state transition graph of the (d, k) constrained channel.

vertex contains a set of states with same length of closing run represented by vectors, and each edge corresponds to an edge adjacency matrix describing the connection between the state vectors of neighboring vertices. There are two kinds of edge matrices: $\mathbf{A}_1$ which continues the current run, and $\mathbf{A}_2$ which closes it, starting a new run with opposite sign, as $\mathbf{Q}_1$ and $\mathbf{Q}_2$ do. So, those can be derived from the original state transition probability matrix $\mathbf{Q}$ by putting ones in place of nonzero elements of $\mathbf{Q}_1$ and $\mathbf{Q}_2$ respectively:

$$\mathbf{A}_1 = [\mathbf{Q}_1 \neq 0] \quad \text{(the run is continuing);}$$
$$\mathbf{A}_2 = [\mathbf{Q}_2 \neq 0] \quad \text{(a new run is starting).}$$

Using the variable length symbol representation [24], [25], i.e. the edges can correspond to sequences of different length, the state transition diagram can be reduced into a one-vertex graph (Fig. 6), which is described with the following adjacency matrix:

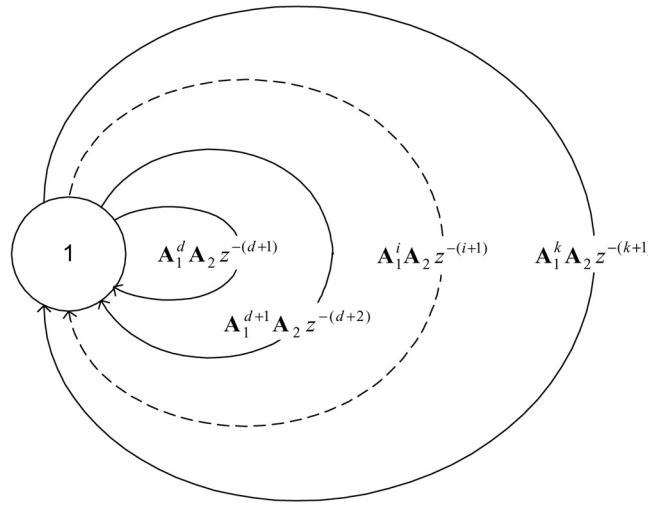

Fig. 6 The variable length graph of the simultaneously spectrum and (d, k) constrained channel.

$$\mathbf{A}_{d,k}(z) = \sum_{i=d}^{k} z^{-(i+1)} \mathbf{A}_1^i \mathbf{A}_2.$$

Then the channel capacity is given as the base two logarithm of the largest root of the characteristic polynomial $\det[\mathbf{A}_{d,k}(z) - \mathbf{I}]$ [23], [24].

The zero capacity indicates that the spectral and run-length constraints can not be matched without breaching either of them, so setting priorities is inevitable. Conversely, the non-zero capacity means that the constraints can be straight matched or with the help of some auxiliary constraints at most.

The state transition diagram of a coder for simultaneously spectrum and (d, k) constrained channel can be seen in Fig. 7, which is a hyper graph too. The edge matrix $\widetilde{\mathbf{Q}}_1$ corresponds to continuing the current run, while $\widetilde{\mathbf{Q}}_2$ stands for closing it by inserting $d+1$ bits with opposite sign, and $\widetilde{\mathbf{A}}_2$ does the same but unconditionally. If the run-length and spectral constraints never clash, these matrices can be straight derived from the original spectrum constrained system's transition matrix:

$$\widetilde{\mathbf{Q}}_1 = \mathbf{Q}_1; \quad \widetilde{\mathbf{Q}}_2 = \mathbf{Q}_2\mathbf{A}_1^d; \quad \widetilde{\mathbf{A}}_2 = \mathbf{A}_2\mathbf{A}_1^d.$$

The clash of constraints means that there are states with transition probability less than 1. Making the matrix $\widetilde{\mathbf{Q}}_1 + \widetilde{\mathbf{Q}}_2$ stochastic by successive removing the dead end states or

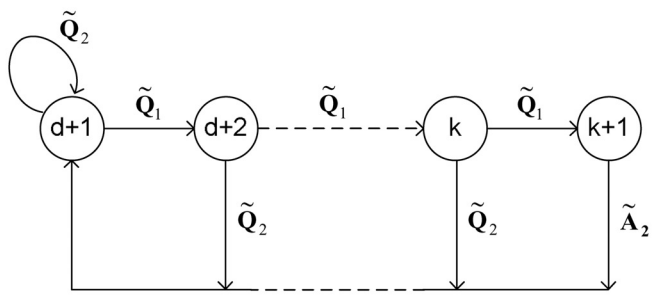

Fig. 7 The state transition diagram of the coder.

inserting new transitions, and then setting the probability of lonely transitions to 1, we can patch the broken Markov chain in vertices "$d{+}1$",...,"k". Establishing new transitions in $\widetilde{\mathbf{Q}}_2$ corresponds to setting priority of d constraint over the spectral constraint, while by removing dead end states or turning the probability of the single transitions to 1, we implement auxiliary constraints which can preserve the validity of both constraints. Moreover, we should secure the prevailing of k constraint in vertex "$k+1$" by adding transitions to $\widetilde{\mathbf{A}}_2$ in states where k and spectral constraints collide.

On the basis of Fig. 7 we can construct the variable length transition matrix of the coder:

$$\mathbf{Q}_{d,k}(z) = \sum_{i=0}^{k-d-1} z^{d+1+i}\,\widetilde{\mathbf{Q}}_1^i\widetilde{\mathbf{Q}}_2 + z^{k+1}\,\widetilde{\mathbf{Q}}_1^{k-d}\widetilde{\mathbf{A}}_2.$$

Let $\boldsymbol{\pi}_{d,k} = \boldsymbol{\pi}_{d,k}\mathbf{Q}_{d,k}(1)$ the stationary distribution vector for vertex "$d+1$", i.e., the stationary distribution under the condition that a new run has started. Then the generating function of the distribution of run-length can be given as

$$g(z) = \boldsymbol{\pi}_{d,k}\mathbf{Q}_{d,k}(z)\,\mathbf{1}. \tag{20}$$

To get the coder's rate, first we determine the average run-length at the output, what is given by $g'(1)$:

$$\overline{N}_{\text{out}} = \boldsymbol{\pi}_{d,k}\left[\sum_{i=0}^{k-d-1}(d{+}1{+}i)\,\widetilde{\mathbf{Q}}_1^i\widetilde{\mathbf{Q}}_2 + (k{+}1)\,\widetilde{\mathbf{Q}}_1^{k-d}\widetilde{\mathbf{A}}_2\right]\mathbf{1}.$$

Introducing the indicator vector $\mathbf{i}=\mathbf{1}{-}[\widetilde{\mathbf{Q}}_1\mathbf{1}{==}1]{-}[\widetilde{\mathbf{Q}}_2\mathbf{1}{==}1]$, which selects the states when no stuffing is applied and the coder gets a bit from the input, the average number of input bits during an output run reads as

$$\overline{N}_{\text{in}} = \boldsymbol{\pi}_{d,k}\sum_{i=0}^{k-d-1}\widetilde{\mathbf{Q}}_1^i\,\mathbf{i}.$$

Then the rate is given as $R = \overline{N}_{\text{in}}/\overline{N}_{\text{out}}$.

The spectral density of the output signal can also be calculated with the help of generating function (20) by the formula published in [26]:

$$S_Y(z) = \frac{4\,\boldsymbol{\pi}_{d,k}[(\mathbf{I}+\mathbf{Q}_{d,k}(z))^{-1}+(\mathbf{I}+\mathbf{Q}_{d,k}(z^{-1}))^{-1}-\mathbf{I}]\,\mathbf{1}}{\overline{N}_{\text{out}}\,|1-z|^2}.$$

D. An Example: The Window-Charge Constraint

In many applications (e.g. ac-coupled channels) it is an important requirement that the code spectrum should be poor around the zero frequency [2], [27], [28]. With the application of a low-pass loop filter we can satisfy this requirement. Using the order r window filter $H(z) = \sum_{i=0}^{r} z^{-i}$ as loop filter, according to (8), the output sequence will have a dc-suppressed spectrum with ripples in passband region (Fig. 8). Increasing the order of the filter, the suppression will grow, while the width of the suppressed region will diminish. Certainly, diminishing the threshold will enhance the suppression. To reach the same suppression the filter with higher order will result in a higher code rate.

Now the WRDS is defined as $W_n = \sum_{i=0}^{r} Y_{n-i}$. The values of possible W_n's form a finite set of even or odd integers depending on the value of r. Without loss of generality, we can confine the coders's threshold value on that very same set, supposing that c_0 is a mod 2 congruent integer to $r+1$: $(r+1-c_0) \bmod 2 \equiv 0$. Then the coding rule reads as

$$Y_{n+1} = \begin{cases} X_{m+1}Y_n, & \text{if} \quad |\sum_{i=0}^{r} Y_{n-i}| < c_0; \\ -\text{sgn}(W_n), & \text{if} \quad |\sum_{i=0}^{r} Y_{n-i}| = c_0. \end{cases} \tag{21}$$

Taking the incoming bit into account, one can see that the coder limits the accumulated charge in an $r+2$ bit long sliding window, and the channel sequence will comply with the following constraint:

$$\left|\sum_{i=0}^{r+1} Y_{n-i}\right| \le c_0 - 1 \;\text{ for any }\; n \in \mathbf{N}.$$

We refer the constraints of above type to as window-charge, or shortly (w, c) constraint. The w and c code parameters stand for the window's length, now $w = r+2$, and the charge limit, now $c = c_0 - 1$. The coding algorithm defined by (21) is greedy for the above constraint, i.e., it can generate all the possible sequences obeying the constraint with parameters $(r+2, c_0-1)$. Certainly, the output sequence will also comply with the constraint $(r+1, c_0)$, however, the coding algorithm will not be greedy for that constraint because it makes an unnecessary stuffing when $|W_n| = c_0$ and the outgoing bit $Y_{n-r} = \text{sgn}(W_n)$. It implies that a sequence obeying the constraint $(r{+}1, c_0)$ will not necessarily comply with $(r{+}2, c_0{-}1)$, i.e., the latter one is a stronger condition.

The performance of the coder can be analyzed with the help of the loop filter's states. Let us call a state light if its disparity is smaller than the bound: $|W| < c_0$; and heavy if those are equal: $|W| = c_0$. Introducing the notations $n_1 = (r+1-c_0)/2$ and $n_2 = (r{+}1{+}c_0)/2$ for the minimum and maximum number of identical bits in the window, the number of light and heavy states are

$$N_l = \sum_{i=n_1+1}^{n_2-1}\binom{r+1}{i};\quad N_h = \binom{r+1}{n_1} + \binom{r+1}{n_2} = 2\binom{r+1}{n_1}.$$

In case of unbiased, i.i.d. input the stationary distribution can be determined simply:

Theorem *If the input process X is i.i.d. with* $\Pr(X=+1) = \Pr(X=-1) = 1/2$*, the stationary distribution of the coder's states depends only on the weight of the states. The probabilities of all light states and the probabilities of all heavy states are equal, and the probability of a light state is the double of a heavy one:*

$$p_l = \frac{2}{2N_l + N_h}; \qquad p_h = \frac{1}{2N_l + N_h}.$$

Proof: According to coding rule, each light state has two parents and two children, while a heavy state has only one of each. Since the input process is i.i.d. and $\Pr(X=+1) = \Pr(X=-1) = 1/2$, the stationary probability distribution is

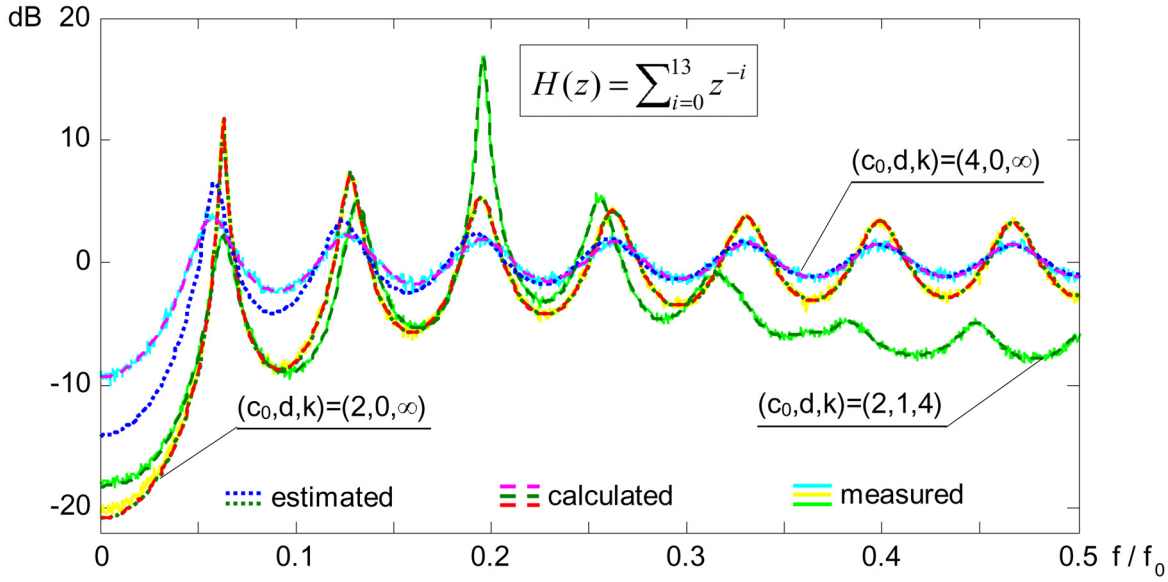

Fig. 8 Dc-suppressed code spectra generated by an order 13 window filter.

determined by the set of following four kinds of equations:

$$\begin{aligned}
&\pi_i = \tfrac{1}{2}\pi_j + \tfrac{1}{2}\pi_k, && \text{if state } i,\ j \text{ and } k \text{ are each light;}\\
&\pi_i = \tfrac{1}{2}\pi_j + \pi_k, && \text{if state } i \text{ and } j \text{ are light while } k \text{ is heavy;}\\
&\pi_i = \tfrac{1}{2}\pi_j, && \text{if state } i \text{ is heavy and } j \text{ is light;}\\
&\pi_i = \pi_j, && \text{if state } i \text{ and } j \text{ are both heavy.}
\end{aligned}$$

The above system of linear equations can be satisfied with any π_i's such that

$$\pi_i = \begin{cases} 2p_h, & \text{if state } i \text{ is light;}\\ p_h, & \text{if state } i \text{ is heavy.}\end{cases}$$

Then the probability p_h can be determined by the condition $\sum \pi_i = 1$:

$$p_h = \frac{1}{2N_l + N_h}.$$

According to Perron–Frobenius theorem [29], the above solution is unique as well. ■

The redundancy is given by the stuffing probability $\Pr(|W| = c)$, so the rate can be calculated as

$$R = 1 - \Pr(|W| = c_0) = 1 - p_h N_h = \frac{2N_l}{2N_l + N_h}.$$

The error propagation, i.e., the expected value of the number of false detections induced by a single error, can be calculated under the condition that the error and the channel process are independent. An error during the transmission can cause two kinds of errors at the decoder. The type one error is when an originally light state with disparity c_0-2 is detected as heavy, causing a false removal. Its probability is

$$p_{e1} = 2p_l \frac{n_1+1}{r+1}\binom{r+1}{n_1+1} = 2p_h \frac{n_2}{r+1} N_h.$$

The type two error occurs when a heavy state is detected as light, leaving a redundant bit in the stream. The probability of this error is

$$p_{e2} = 2\,p_h \frac{n_2}{r+1}\binom{r+1}{n_2} = p_h \frac{n_2}{r+1} N_h.$$

Then the total decoding error probability, under the condition that a single error has occurred during the transmission, reads as

$$p_e = p_{e1} + p_{e2} = 3\,\frac{n_2}{r+1}\,p_h N_h = \frac{3}{2}\left(1 + \frac{c_0}{r+1}\right)(1-R).$$

Since the erroneous bit remains in the memory of the decoder for $r+1$ steps, the expected value of the number of false detections induced by a single error is

$$\mathrm{E}(n_e) = (r+1)\,p_e = \frac{3}{2}\,(r+1+c_0)\,(1-R).$$

The (w, c) constraint also imposes an upper bound on the run-length, since a run can not be longer than $(w + c)/2$ bits and the special case when $w = c+2$ exactly corresponds to the run-length constraint $k = c$. However, we can prescribe RRL constraints explicitly as well. According to (9), the coding rule will be the following:

$$Y_{n+1} = \begin{cases} -\mathrm{sgn}(W_n), & \text{if } \left|\sum_{i=0}^{r} Y_{n-i}\right| = c_0\,;\\ -Y_n, & \text{if } \left|\sum_{i=0}^{k} Y_{n-i}\right| = k+1;\\ Y_n, & \text{if } \left|\sum_{i=0}^{d} Y_{n-i}\right| < d+1;\\ Y_n, & \text{if } Y_n \sum_{i=0}^{r-d} Y_{n-i} \le d - c_0\,; \quad (*)\\ X_{m+1}Y_n, & \text{otherwise} \quad \text{(no stuffing).}\end{cases}$$

The last stuffing case denoted by $(*)$ is an auxiliary constraint. It is applied to avoid the collision of (w, c) and d constraints after short runs, forcing the coder to lengthen the current run.

The coder's transition matrix should be modified accordingly by turning the probabilities of lonely transitions to 1 making the matrix stochastic:

$$\widetilde{\mathbf{Q}}_1 = \mathrm{diag}(\mathbf{1} - \widetilde{\mathbf{Q}}_2\mathbf{1})\mathbf{A}_1.$$

The spectrum of the coded signal can be seen in Fig. 8.

IV. CODING WITH IIR FILTERS

In case of IIR loop filters, there are infinite many non-zero among the coefficients h_i in definition of WRDS. So, as the process advances, the number of possible values of W, i.e., state space of the Markov chain is growing permanently and tends to infinite. Moreover, along with that growing, the Markov chain is getting unstable, i.e., the probabilities of the states tend to zero, so the discrete Markov model can not be used anymore. For the exact mathematical description the process should be taken as Markov process with a multidimensional continuous state space. The dimension of sate space corresponds to the applied filter's order. The transitions are described as functions of the state space, and the stationary distribution can be earned as the solution of a functional equation. We present the method with first order low-pass loop filter, however, the whole method is cumbersome and hard to solve even in this simplest case.

Rather than following the exact model, we can approach the original Markov process with a Markov chain by discretizing the state space. Then it can be handled with the matrix method applied for coders with FIR loop filters. It can fairly model the original system since, in the strict sense, the actual coder also performs as Markov chain rather than Markov process due to the roundoff errors of the filter circuits. The only difficulty with the method is the state space is prone to getting fast unmanageably large by growing the filter's order. Therefore, it is useful to find the exact support of the multidimensional distribution function before the discretization, which can be itself a hard problem.

A. The α-Charge Constrained Code

Applying the IIR low-pass filter $H(z) = 1/(1-\alpha z^{-1})$ $(0<\alpha<1)$ as loop filter, the WRDS is defined as

$$W_n = \sum_{i=0}^{n-1} \alpha^i Y_{n-i} = Y_n + \alpha W_{n-1}. \tag{22}$$

In accordance with (8), it will also result in a DC-suppressed code spectrum, as can be seen in Fig. 9. The greater α we use, the deeper and steeper suppression, but the error propagation is also increased. On the basis of (5), for threshold c_0 it should hold that $1/(1+\alpha) < c_0 < 1/(1-\alpha)$. According to coding rule (4), the WRDS defined by (22) is bound by $c = \alpha c_0 + 1$, and the output sequence will obey the following constraint:

$$\left| \sum_{i=0}^{n-1} \alpha^i Y_{n-i} \right| < c \quad \text{for any} \quad n \in \mathbf{N}.$$

This constraint will be referred to as α-charge or shortly (α, c) constraint. When $\alpha = 1$, we get the conventional charge (RDS) constraint, however, this value of α should not be used with the bit stuffing method since it breaches the finite memory condition (6) causing an infinite error propagation with a probability of 1 for any finite error rate. The coding algorithm defined by (4) and (22) is greedy in generating $(\alpha, \alpha c_0 + 1)$ constrained sequences.

To find the capacity of α-charge constrained channel, let us consider the set $\mathcal{A}^n$ of n dimensional binary vectors: $[a_1, a_2, \ldots, a_n]$, $a_i \in \{-1, +1\}$, and define the rectified WRDS (RWRDS) on the elements of $\mathcal{A}^n$ as

$$\begin{aligned} \widehat{W}_i &= a_i \textstyle\sum_{j=0}^{i-1} \alpha^j a_{i-j} \quad (i = 1, \ldots, n) \\ &= 1 + \frac{a_i}{a_{i-1}} \alpha \widehat{W}_{i-1} \ (i = 2, \ldots, n) \quad \text{and} \quad \widehat{W}_1 = 1. \end{aligned} \tag{23}$$

The RWRDS of an α-charge constrained sequence is confined within the interval $(1-\alpha c,\ c)$. For the lower bound we will use the shortcut $c' = 1 - \alpha c$. It is convenient to rectify the WRDS according to the sign of the current run since RWRDS is always increasing during a run, so it is enough to set an upper bound:

$$\left| \sum_{i=0}^{n-1} \alpha^i Y_{n-i} \right| < c \ \Leftrightarrow \ \sum_{i=0}^{n-1} Y_n \alpha^i Y_{n-i} < c$$

Now let us consider the subset $\mathcal{A}_c^n$ of $\mathcal{A}^n$ where the RWRDS is limited: $\widehat{W}_1, \widehat{W}_2, \ldots, \widehat{W}_n < c$. Let $S(n)$ denote the number of elements in $\mathcal{A}_c^n$ and $F_n(x)$ the distribution of $\widehat{W}_n$ on the set, i.e., the probability that the RWRDS of a randomly chosen element of $\mathcal{A}_c^n$ is smaller than x: $F_n(x) = \Pr(\widehat{W}_n < x)$. Then, on the basis of (23), we can write:

$$\begin{aligned} S(n)F_n(x) = \ & S(n-1)F_{n-1}(\tfrac{x-1}{\alpha}) \\ & + S(n-1)[1 - F_{n-1}(-\tfrac{x-1}{\alpha})]. \end{aligned} \tag{24}$$

The term $F_{n-1}(\frac{x-1}{\alpha})$ in the above equation refers to the case when the current run is continued, while $1 - F_{n-1}(-\frac{x-1}{\alpha})$ is standing for starting a new run, and the arguments are the values of RWRDS of the step before. Specially, for $x = c$ $F_n(c) = 1$, while, since the value of the second argument $(1-c)/\alpha = -c_0$ is always below c', the probability $F_{n-1}(-\frac{c-1}{\alpha})$ is zero. Thus for $x = c$ (24) reads as

$$S(n) = S(n-1)\,[1 + F_{n-1}(\tfrac{c-1}{\alpha})]. \tag{25}$$

Combining (24) and (25), function $F_n(x)$ can be given recurrently:

$$F_n(x) = \frac{F_{n-1}(\frac{x-1}{\alpha}) + [1 - F_{n-1}(-\frac{x-1}{\alpha})]}{1 + F_{n-1}(\frac{c-1}{\alpha})}.$$

On the basis of the above recurrence, for the stationary distribution $F(x) = \lim_{n\to\infty} F_n(x)$ we get the following functional equation:

$$F(x) = \begin{cases} 0, & \text{if } x \le c'; \\ \dfrac{1 + F(\frac{x-1}{\alpha}) - F(-\frac{x-1}{\alpha})}{1 + F(\frac{c-1}{\alpha})}, & \text{if } c' < x \le c; \\ 1, & \text{if } x > c. \end{cases} \tag{26}$$

From (25) one can see that the number $S(n)$ of elements in $\mathcal{A}_c^n$ increases exponentially for large n's:

$$S(n) \asymp [1 + F(\tfrac{c-1}{\alpha})]^n,$$

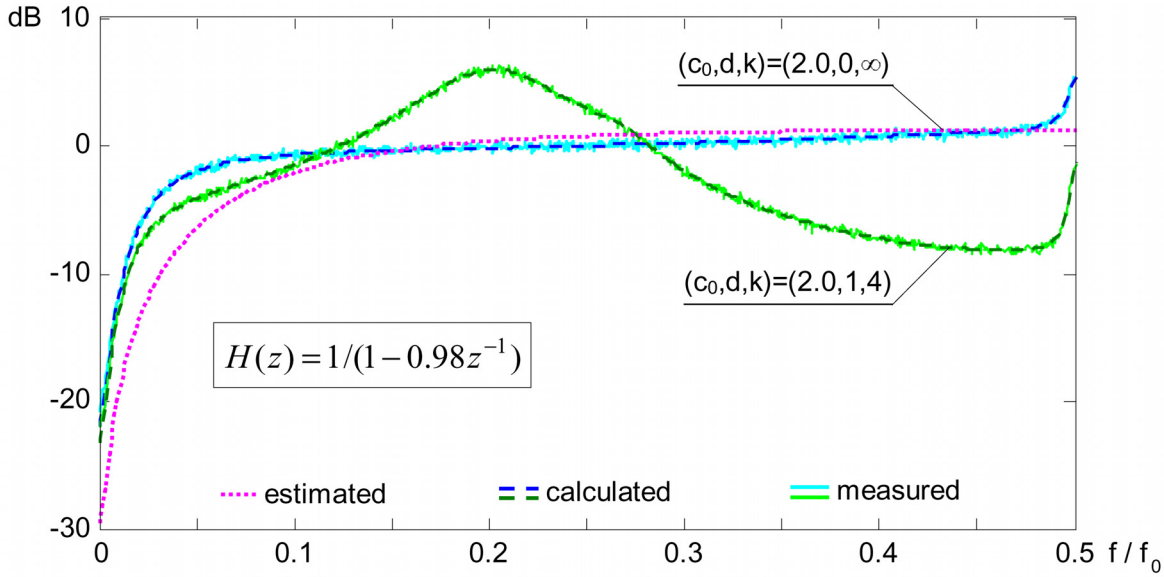

Fig. 9 Dc-suppression by IIR low-pass loop filter with no run-length constraint (blue/magenta), and with an additional (d, k) constraint (green).

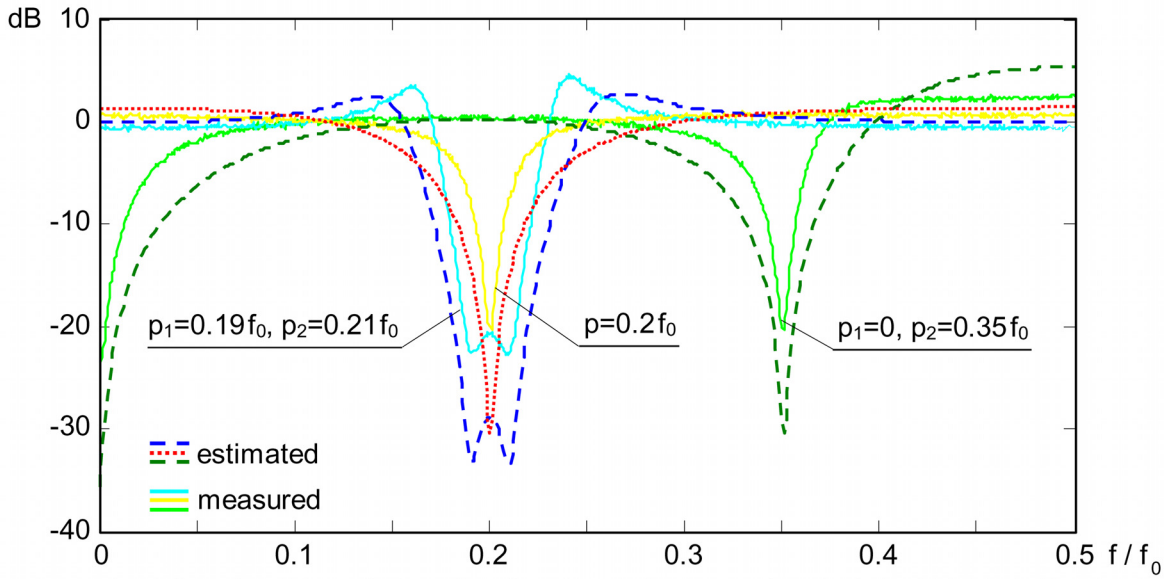

Fig. 10 Spectral notch by IIR band pass loop filter with a single pole at $0.2f_0$ (yellow/red), widened by double poles at $0.19f_0$ and $0.21f_0$ (cyan/blue), and combined with dc-suppression (light green/green).

thus the channel capacity is given as

$$C = \lim_{n\to\infty} \frac{\log_2 S(n)}{n} = \log_2 [1 + F(\tfrac{c-1}{\alpha})].$$

Using the rectified WRDS, the coding rule for the α-charge constrained channel with threshold $c_0 = (c-1)/\alpha$ will be the following:

$$Y_{n+1} = \begin{cases} -Y_n, & \text{if } \widehat{W}_n \geq c_0; \\ X_{m+1}Y_n, & \text{if } \widehat{W}_n < c_0. \end{cases} \tag{27}$$

Supposing that the input process is i.i.d. with $\Pr(X=+1) = p$ and $\Pr(X=-1) = q$, for the stationary distribution $G(x) = \lim_{n\to\infty} \Pr(\widehat{W}_n < x)$ we can get a functional equation similar to (26):

$$G(x) = \begin{cases} 0, & \text{if } x \leq 1-\alpha c = c'; \\ 1 - G(-\frac{x-1}{\alpha}), & \text{if } c' < x \leq 1-\alpha c_0 = 2-c; \\ p\,G(\frac{x-1}{\alpha}) + q\,[1 - G(-\frac{x-1}{\alpha})] & \\ \qquad + p\,[1 - G(\frac{c-1}{\alpha})], & \text{if } 2-c < x \leq c; \\ 1, & \text{if } x > c. \end{cases} \tag{28}$$

The second row of (28) refers to the transitions when stuffing is applied forcing to close the current run. The low values of RWRDS can be reached only this way since, as we have seen, $-c_0 < c'$, therefore the term $G(\frac{x-1}{\alpha})$ standing for continuing the current run is missing. The third row refers to the transitions when no stuffing is applied. The constant

TABLE I The most important parameters of some spectrum constrained codes

loop filter characteristics		threshold c_0	channel capacity	rate	error propagation
$\sum_{i=0}^{r} z^{-i}$	$r = 13$	2.0	0.626	0.533	11.2
		4.0	0.887	0.825	4.7
	$r = 13$, d=1, k=4	2.0	0.297	0.261	5.3
$\frac{1}{1-\alpha z^{-1}}$	$\alpha = 0.98$	2.0	0.853	0.808	26.2
	α=0.98, d=1, k=4	2.0	0.503	0.468	16.3
$H(z,p)=\frac{\alpha\cos(p\,2\pi)-\alpha^2 z^{-1}}{1-2\alpha\cos(p\,2\pi)+\alpha^2 z^{-2}}$	α= 0.98, p = 0.2	1.8	no data	0.813	36.3
$H(z,p_1)+H(z,p_2)+H(z,p_1)H(z,p_2)z^{-1}$	$\alpha = 0.98$, p_1= 0.19, p_2= 0.21	3.0	no data	0.597	42.4
	$\alpha = 0.99$, p_1= 0, p_2= 0.3	1.2	no data	0.684	42.5

term $\frac{1}{2}[1-G(\frac{c-1}{\alpha})]$ provides the continuity of the distribution function.

Earlier we have showed that the rate of a bit stuff encoder is given as $1-P_{\text{stuff}}$, so according to (27), the coder's rate is

$$R = 1 - P_{\text{stuff}} = 1 - \Pr(\widehat{W} \geq c_0) = G(c_0).$$

The α-charge constraint, as most of the constraints defined by low pass filter, constitutes an upper bound on the run-length in and of itself:

$$T_{\max} = \left\lceil \frac{\log[1-(1-\alpha)c_0] - \log[1+(1-\alpha)(1+\alpha c_0)]}{\log \alpha} \right\rceil,$$

which corresponds to the k constraint $k = T_{\max} - 1$. However, explicit RLL constraints can also be given. Neither k nor d constraint can collide with (α, c) constraint if the condition $d+1 < T_{\max}$ is satisfied. The spectrum of a simultaneously α-charge and (d,k) constrained sequence can be seen in Fig. 9.

B. Forming Spectral Notches

Applying a loop filter which sets bandpass characteristics to $\widetilde{H}(z)$ will result in a bandstop-like code spectrum forming a notch in the spectrum. Such code spectra are used to accommodate auxiliary information in spectrum [7], e.g., pilot tracking tone for head positioning mechanism of digital magnetic and optical recorders [2]. Applying a notch at $f_0/2$ will suppress the power around the half of the symbol rate rendering protection against band-edge filter distortion. The IIR filter

$$H(z,p) = \frac{\alpha\cos(p\,2\pi) - \alpha^2 z^{-1}}{1 - 2\alpha\cos(p\,2\pi)\,z^{-1} + \alpha^2 z^{-2}}$$

it will generate a spectral notch at pf_0 (Fig. 10). It behaves similarly to the characteristics by the low-pass IIR filter, that is, the greater α is chosen, the deeper and steeper suppression will be, with a higher error propagation susceptibility. Placing another pole in $\widetilde{H}(z)$ next to the first one, we can enhance the width of the spectral notch without diminishing the suppression (Fig. 10):

$$H(z,p_1,p_2) = H(z,p_1) + H(z,p_2) + H(z,p_1)H(z,p_2)z^{-1} \quad (29)$$

Letting $p_1 = 0$ in (29), the first pole will appear in dc. It combines the spectral notch with a dc-suppressed code spectrum (Fig. 10), which is often required.

In Table I we have collected the most important parameters of spectrum constrained codes discussed in this paper. The error propagation is defined as the average number of false detections (false removals or remanent stuffed bits) induced by a single error. It has been measured at a BER of 10^{-4}.

V. CONCLUSION

In this paper we have generalized the accumulated charge concept and introduced a new class of constraints the generalized charge constraint. With the new constraint the spectral requirements can be described easily in the time domain. A feedback controlled bit stuff encoder with loop filter is suggested to implement the new constraint. We have studied the performance of the new coder structure and demonstrated its spectral shaping property. We have presented a few spectral characteristics of practical interest generated by low- and bandpass loop filters.

ACKNOWLEDGMENT

I wish to thank Peter Tatai who has first called my attention to sigma-delta converters and László Osváth for his insightful discussions and comments, and for his contribution to the proof of Theorem on stationary distribution of window-charge constrained code. I also wish to express my thanks to Géza Gordos and Gyula Sallai, who supported my work, and to Rudy Kalman, who gave me the occasion to talk over my ideas.

REFERENCES

[1] H. Kobayashi, "A survey of coding schemes for transmission or recording of digital data," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 1067–1100, Dec. 1971.

[2] K.A.S. Immink, *Coding Techniques for Digital Recorders.* London: Prentice Hall, 1991.

[3] Y.L. Hsueh, M.S. Rogge, et al., "Smooth upgrade of existing passive optical networks with spectral-shaping line-coding service overlay," *IEEE/OSA Journal of Lightwave Technology*, vol. 23, No. 5, pp. 2629–2637, Sept. 2005.

[4] D.T. Tang and L.R. Bahl, "Block codes for a class of constrained noiseless channels," *Information and Control*, vol. 17, pp. 436–461, 1970.

[5] B.H. Marcus and P.H. Siegel, "On codes with spectral nulls at rational submultiples of the symbol frequency," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 557–568, July 1987.

[6] I.J. Fair, W.D. Grover, W.A. Krzymien and R.I. McDonald, "Guided scrambling: A new line coding technique for high bit rate fiber optic transmission systems," *IEEE Trans. Commun.*, vol. COM-39, pp. 289–236, Feb. 1991.

[7] J.C. Cavers and R.F. Marchetto, "A new coding technique for spectral shaping of data," *IEEE Trans. Commun.*, vol. COM-40, pp. 1418–1422, Sept. 1992.

[8] ISO/IEC 3309 standard.

[9] P.E. Bender and J.K. Wolf, "A universal algorithm for generating optimal and nearly optimal run-length-limited charge-constrained binary sequences," in *Proc. IEEE Symp. on Inform. Theory*, San Antonio, Texas, Jan. 1993, p. 6.

[10] S. Aviran, P.H. Siegel and J.K. Wolf, "An improvement to the bit stuffing algorithm," *IEEE Trans. Inform. Theory*, vol. IT-51, pp. 2885–2891, August 2005.

[11] Y. Sankarasubramaniam and S.W. McLaughlin, "Capacity achieving code constructions for two classes of (d, k) constraints," *IEEE Trans. Inform. Theory*, vol. IT-52, pp. 3333–3343, July 2006.

[12] S. Aviran, P.H. Siegel and J.K. Wolf, "Optimal parsing trees for run-length coding of biased data," *IEEE Trans. Inform. Theory*, vol. IT-54, pp. 841–849, Feb. 2008.

[13] P. Vámos, "A coding theorem of runlength limited channel," in *Proceedings of IEEE International Symposium on Information Theory*, Washington, DC, June 24-29, 2001, p.66.

[14] G.L. Pierobon, "Codes for zero spectral density at zero frequency," *IEEE Trans. Inform. Theory*, vol. IT-30, pp. 435–439, March 1984.

[15] P. Vámos and L. Osváth, "The error propagation of bit-stuff coding systems," in *Proceedings of IEEE Symp. on Inform. Theory*, Seattle, WA, July 2006.

[16] W.G. Bliss, "Circuitry for performing error correction calculations on baseband encoded data to eliminate error propagation," *IBM Techn. Discl. Bul.*, vol. 23, No. 10, pp. 4633–4634, March 1981.

[17] K.A.S. Immink, "Code configuration for avoiding error propagation," *Electronics Letters*, vol. 32, pp. 2191–2192, Nov. 1996.

[18] R.M. Gray, "Quantization noise spectra," *IEEE Trans. Inform. Theory*, vol. IT-36, pp. 1220–1244, Nov. 1990.

[19] H.A. Spang and P.M. Schulthess, "Reduction of quantizing noise by use of feedback," *IRE Trans. Commun. Systems*, vol. COM-10, pp. 373–380, Dec. 1962.

[20] H. Inose and Y. Yasuda, "A unity bit coding method by negative feedback," *Proc. IEEE*, vol. 51, pp. 1524–1535, Nov. 1963.

[21] W.H. Kautz, "Fibonacci codes for synchronization control," *IEEE Trans. Inform. Theory*, vol. IT-11, pp. 284–292, April 1965.

[22] A.R. Collar, "On centrosymmetric and centroskew matrices," *Quart. Journ. Mech. and Appl. Math.*, vol. 15, pp. 265–281, 1962.

[23] C.E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pt. I, pp. 379–423, 1948; pt. II, pp. 623–656, 1948.

[24] C.D. Heegard, B.H. Marcus and P.H. Siegel, "Variable-length state splitting with applications to average runlength-constrained (ARC) codes," *IEEE Trans. Inform. Theory*, vol. IT-37, pp. 759–777, May 1991.

[25] K.J. Kerpez, A.Gallopoulos and C.D.Heegard, "Maximum entropy charge-constrained run-length codes," *IEEE Journ. Sel. Areas Commun.*, vol. 10, pp. 242–253, Jan. 1992.

[26] A.Gallopoulos, C.D.Heegard and P.H. Siegel, "The power spectrum of run-length-limited codes," *IEEE Trans. Commun.*, vol. COM-37, pp. 906–917, Sept. 1989.

[27] K.A.S. Immink, "Performance of simple binary dc-constrained codes," *Philips J. Res.*, vol. 40, No. 1, pp. 1–21, 1985.

[28] A.X. Widmer and P.A. Franaszek, "Transmission code for high-speed fiber-optic data networks," *Electronics Letters*, vol. 19, pp. 202–203, March 1983.

[29] F.R. Gantmacher, *Theory of Matrices.* New York: Chelsea Publishing Co., 1960.