GENERATING BINARY PROCESSES WITH ALL-POLE SPECTRA

Petros Boufounos

MIT Digital signal Processing Group,
77 Massachusetts Avenue, Rm. 36-615, Cambridge, MA 02139
petrosb@alum.mit.edu

ABSTRACT

This paper presents an algorithm to generate autoregressive random binary processes with predefined mean and predefined all-pole power spectrum, subject to specific constraints on the parameters of the all-pole spectrum. The process is generated recursively using a linear combination of the previously generated values to bias the generation of the next value. It is shown that an all-zero filter whitens the process, and, therefore, the process has an all-pole spectrum. The process is also described using an ergodic Markov chain, which is used to determine the appropriate initialization and to prove convergence if the algorithm is not initialized properly. The all-pole parameter range for which the algorithm is guaranteed to work is also derived. It is shown to be a linear constraint on the all-pole parameters and their magnitude, subject to the desired mean for the process. The example and simulations presented elucidate and confirm the theoretic developments.

Index Terms— autoregressive, binary sequence, stochastic process

1. INTRODUCTION

The generation of random binary processes with pre-specified power spectra is important in a variety of signal processing applications such as randomized sampling, radar waveform generation and others [1–4]. For example, in randomized sampling applications, spectrally shaped binary zero-one random processes are used to dictate the sampling times and mitigate the effects of aliasing [1]. In radar applications, binary processes taking values in $\{-1, 1\}$ are useful because they exhibit low peak-to-average power ratio [3]. Unfortunately, the generation of binary random processes with arbitrary pre-determined power spectra is not straightforward. Although the generation of a random process with an arbitrary spectrum is possible by linear filtering of a white process, the output of the linear filter cannot be guaranteed to be a binary sequence, even if the input is binary. This paper presents an algorithm to generate binary random processes that have pre-specified all-pole spectra, subject to certain conditions on the spectrum parameters.

Several algorithms exist to generate binary processes (for examples see [5–7]). However, the spectrum of these processes is often difficult to compute and analyze. The algorithm presented is parametrized by the coefficients of the desired centralized power spectrum—subject to well-defined constraints—making the generation of a process with a desired spectrum straightforward. Although the algorithm implements a Markov chain, the presentation in this paper focuses mostly on the spectral properties. It should be noted that it is not always possible to generate a binary process with an arbitrary power spectrum [4, 8]. The algorithm described in this paper is also a proof that all-pole spectra with parameters in the range described are realizable.

The problem can be considered as an infinite-length extension to the problem of generating a finite-length sequence of random binary variables with a pre-specified mean for each variable and pre-specified correlation structure among them. The finite-length problem has been extensively studied in the statistical literature (for examples, see [9–14] and references within). However, the solutions developed in the finite-length case cannot be immediately applied to generate a random process. Most of the solutions require that the length of the vector is known in advance. Furthermore, some have computational complexity that does not scale well with the length of the sequence. The algorithm presented in this paper can be considered an extension of [13] to random processes. Although the generation principle is similar, this paper considers issues unique to random processes such as stationarity, initialization, and convergence of the algorithm. A whitening argument is also introduced to prove that the algorithm generates the desired process.

The next section defines the problem and establishes the notation. Section 3 describes the algorithm to generate the process. Section 4 proves that under specific assumptions the algorithm generates the process desired and that the algorithm converges for any initialization. An example with simulations is presented in Sec. 5. The Appendix proves the conditions on the parameters that guarantee the assumptions in Sec. 4.

2. PROBLEM FORMULATION

The problem is to generate a random binary sequence, denoted using $x[n]$ and taking values in $\{0, 1\}$, with a predetermined mean $\mu = E\{x[n]\}$, and a predetermined autocorrelation denoted using:

$$R_{xx}[m] = E\{x[n]x[n+m]\} = K_{xx}[m] + \mu^2, \qquad (1)$$

in which $K_{xx}[m]$ denotes the autocovariance. This is defined as the autocorrelation of the centered process $x_c[n] = x[n] - \mu$, which is zero mean and takes values in $\{-\mu, 1-\mu\}$:

$$K_{xx}[m] = E\{(x[n]-\mu)(x[n+m]-\mu)\} = R_{x_c x_c}[m]. \qquad (2)$$

The power spectrum is denoted using $S_{xx}(e^{j\omega})$, and the covariance spectrum using $\Phi_{xx}(e^{j\omega})$. They are equal to the Fourier transforms of $R_{xx}[m]$ and $K_{xx}[m]$ respectively.

The covariance spectrum of $x[n]$ should have the form:

$$\Phi_{xx}(e^{j\omega}) = S_{x_c x_c}(e^{j\omega}) = \frac{A}{|1 - \sum_{k=1}^{p} a_k e^{-j\omega k}|^2}, \qquad (3)$$

The author is currently with Rice University, Electrical and Computer Engineering Dept. The work was supported in part by: the Texas Instruments Leadership University Consortium Program, BAE Systems Inc., and MIT Lincoln Laboratory.

The author would like to thank Sourav S. Dey, Alan V. Oppenheim, Charles Rohrs, and Matthew S. Willsey for their help in writing this paper.

in which the all-pole parameters are denoted using a_k, the order of the model is denoted using p, and the scaling constant A depends on the variance of the process. Since the variance of a binary process is a function of the process mean, the constant A is determined by μ and the all-pole parameters a_k. The all-pole parameters are assumed pre-determined, such that the algorithm generates a process with the spectral density desired in the application. The determination of the parameters is not examined in this paper. The only assumption is that they satisfy the constraints described in Sec. 4.4.

3. ALGORITHM

The algorithm generates $x[n]$ iteratively using $x[n-1], \ldots, x[n-p]$ as follows:

1. The bias $x_b[n]$ for the generation of $x[n]$ is computed according to the relationship:

$$x_b[n] = \mu + \sum_{k=1}^{p} a_k (x[n-k] - \mu) \tag{4}$$

$$= \mu + \sum_{k=1}^{p} a_k x_c[n-k] \tag{5}$$

   in which the a_k and μ are the parameters of the algorithm.

2. The sample $x[n]$ is randomly generated from a binary distribution biased by $x_b[n]$ as follows:

$$x[n] = \begin{cases} 1 & \text{with probability } x_b[n] \\ 0 & \text{with probability } 1 - x_b[n] \end{cases} \tag{6}$$

Conditional on $x_b[n]$ the generation of $x[n]$ is independent on any other variable $x[n-k]$. Unless otherwise noted, the assumption in the execution and the analysis of this algorithm is that the bias computed in Eq. (4) is within the interval $[0, 1]$. Section 4.4 discusses the necessary constraints to guarantee that the bias is within the bounds. It should be noted that the algorithm describes a p^{th} order Markov process.

4. ANALYSIS

In this section it is demonstrated that the mean of $x[n]$ is μ, and that the power spectrum of $x_c[n]$, which corresponds to the covariance spectrum of $x[n]$, is the all-pole spectrum in Eq. (3), as desired. The properties of the implied Markov chain are used in section 4.3 to demonstrate the stationarity and the convergence of the algorithm.

4.1. Mean

The mean of $x[n]$ can be evaluated as follows:

$$E\{x[n]\} = E_{x_b[n]}\{E\{x[n]|x_b[n]\} = E\{x_b[n]\} \tag{7}$$

$$= \mu + \sum_{k=1}^{p} a_k (E\{x[n-k]\} - \mu). \tag{8}$$

The sum remains equal to μ as long as the expected value of all previous $x[n-k], k = 1, \ldots, p$ is also μ. Thus, if the algorithm is initialized with p biased binary random variables with mean equal to μ, the mean of the process will stay constant at μ.

Even if the algorithm is not initialized as described, the mean of $x[n]$ converges to μ at a rate governed by the magnitude of the system poles. This can be shown using the discrete-time final value theorem on the dynamic equation describing $E\{x_c[n]\}$:

$$E\{x_c[n]\} = \sum_{k=1}^{p} a_k E\{x_c[n-k]\}. \tag{9}$$

4.2. Power Spectrum

This section demonstrates that the algorithm performs as desired by evaluating the power spectrum of the generated process. In particular, the optimal causal minimum mean squared error (MMSE) predictor is used to predict the next value of the process. It is demonstrated that the MMSE predictor is linear and time-invariant (LTI) by the construction of the algorithm. By the orthogonality principle, the prediction error is white. Thus the spectrum is derived by inverting the LTI system that computes the error.

The optimal causal MMSE predictor for $x_c[n]$ is the expectation conditional on all the previous values $x_c[n-k], k \geq 1$:

$$\hat{x}_c[n] = E\{x_c[n]|x_c[n-k], k = 1, 2, \ldots\}, \tag{10}$$

which, using step 2 of the algorithm, can be computed using a linear combination of the previous p samples:

$$\hat{x}_c[n] = E\{x_c[n]|x_b[n]\} = x_b[n] - \mu \tag{11}$$

$$= \sum_{k=1}^{p} a_k x_c[n-k]. \tag{12}$$

The prediction error, which is white by the orthogonality principle, is equal to:

$$e[n] = x_c[n] - \hat{x}_c[n] \tag{13}$$

$$= x_c[n] - \sum_{k=1}^{p} a_k x_c[n-k]. \tag{14}$$

Therefore, the LTI system $H(z) = 1 - \sum_{k=1}^{p} a_k z^{-k}$ whitens the generated process $x_c[n]$ by calculating the prediction error. The power spectrum of $x_c[n]$ follows:

$$S_{x_c x_c}(e^{j\omega}) = \frac{A}{|H(e^{j\omega})|^2} = \frac{A}{|1 - \sum_{k=1}^{p} a_k e^{-j\omega k)}|^2}, \tag{15}$$

for some constant A, as desired. The stationarity and the convergence of the algorithm is demonstrated in the next section.

4.3. Markov Chain

As noted in section 3, the algorithm defines a p^{th} order Markov process, in which the $\{x[n-k], k = 1, \ldots, p\}$ determine the state at any given time n. Since the $x[n-k]$ only take discrete binary values, the process can also be represented using a 2^p-state Markov chain. Using the properties of this Markov chain, this section demonstrates that the algorithm reaches a stationary steady state.

In this section $(x[n-1], \ldots, x[n-p])$ denotes the state of the system before each iteration of the algorithm in section 3 has been executed. The state transition probabilities are:

$$P\left((x[n], \ldots, x[n-p+1])|(x[n-1], \ldots, x[n-p])\right) =$$

$$= \begin{cases} \mu + \sum_{k=1}^{p} a_k (x[n-k] - \mu), & \text{if } x[n] = 1 \\ 1 - (\mu + \sum_{k=1}^{p} a_k (x[n-k] - \mu)), & \text{if } x[n] = 0, \end{cases} \tag{16}$$

which is equal to $x_b[n]$ and $1-x_b[n]$ if $x[n] = 1$ and 0, respectively. The transition probabilities are zero for all other state transitions. It is assumed that the parameters are such that the probabilities in (16) are both strictly positive, which is equivalent to $0 < x_b[n] < 1$ for all n, as discussed in section 4.4.

Under this assumption, it can be shown any state can be reached with positive probability within p transitions from any other state. Assuming p is finite, it follows that the Markov chain is ergodic, and, therefore, the chain has a unique stationary distribution. The algorithm can be initialized by randomly starting in one of the 2^p states, according to the steady state distribution. Alternatively, the initialization can be arbitrary, and the ergodicity property guarantees convergence to the steady state distribution. This implies that the autocorrelation and the power spectrum of the process converge to the all-pole model. The rate of convergence is governed by the second largest eigenvalue of the implied $2^p \times 2^p$ transition matrix. At the steady state, the process is strict sense stationary, as desired.

4.4. Parameter Range

A necessary condition in the discussion above is that the bias, $x_b[n]$, computed at every iteration of the algorithm is always within the range $(0, 1)$. This can be guaranteed for a given set of parameters a_k if the following inequalities are satisfied:

$$0 < \mu + \sum_{k=1}^{p} a_k (x[n-k] - \mu) < 1 \tag{17}$$

$$\Leftrightarrow -\mu < \sum_{k=1}^{p} a_k x_c[n-k] < 1 - \mu, \tag{18}$$

in which $x_c[n]$ can take either of two values: $1-\mu$, which is positive, and $-\mu$, which is negative. In the Appendix it is shown that (18) is equivalent to the following constraint on the parameters:

$$\left(\sum_{k=1}^{p} a_k - 1 \right) > \frac{1}{|1-2\mu|} \left(\sum_{k=1}^{p} |a_k| - 1 \right), \tag{19}$$

which corresponds to the shaded area in Fig. 1. As μ tends to 1/2, the constraint is relaxed, eventually becoming:

$$\sum_{k=1}^{p} |a_k| < 1 \text{ for } \mu = 1/2. \tag{20}$$

This condition is sufficient but not necessary for $x_b[n]$ to be within the bounds. There exist combinations of parameters a_k and μ that do not satisfy the bound derived, and, yet, the algorithm does not overflow if properly initialized. For example, consider the trivial case of $\mu = 0$, or 1 for any set of a_k outside the constraint in (19), initialized with $x[0] = \ldots = x[p-1] = 0$ or 1, respectively. However, with all such combinations of parameters it can be shown that there is a state for which $x_b[n]$ is exactly equal to 0 or 1, and with a small perturbation of the parameters the algorithm overflows. Furthermore, the constraint in Eq. (19) is necessary to guarantee that arbitrary initialization does not cause $x_b[n]$ to overflow. It is also necessary in proving the ergodicity of the implied Markov chain using the argument in Sec. 4.3.

Although this algorithm demonstrates that it is possible to generate autoregressive processes with parameters that satisfy (19), there is no implication about autoregressive processes with parameters that do not satisfy (19). It might still be possible to generate such processes using different algorithms.

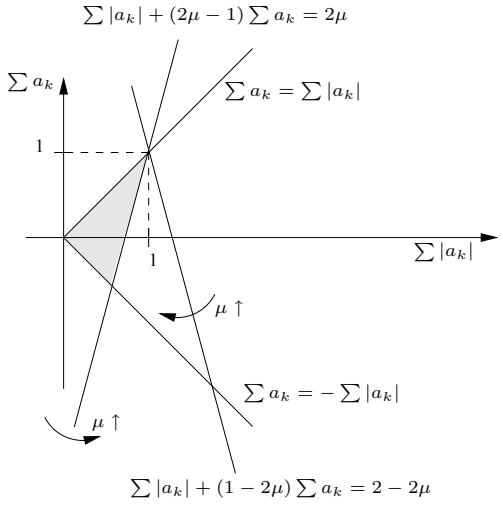

Fig. 1. Coefficient Space for $\mu < 1/2$. The shaded area is the set of coefficients for which the algorithm is guaranteed not to overflow. As μ increases, the two constraints due to Eq. (25) and (26) pivot around the point $(1, 1)$, as shown in the plot. The shaded area is maximized at $\mu = 1/2$. For $\mu > 1/2$ the constraints cross over each other, and the shaded area is identical to the shaded area for $1 - \mu$.

5. TWO-POLE EXAMPLE

In this section a simple two-pole example with $a_1 = 0.1$, and $a_2 = -0.5$ is considered and simulated for different values of the process mean μ. Although this is only one, arbitrarily chosen, example, simulations for various parameter choices verify the results. For this choice of parameter values the constraint in (19) is equivalent to $5/14 < \mu < 9/14$. Therefore, for μ within that range, the bias $x_b[n]$ is guaranteed to be with the bounds $[0, 1]$.

One way to accommodate overflows is to hard limit the bias $x_b[n]$ to be 0 or 1 whenever it is computed below or above the range $[0, 1]$, respectively. This is the method used in the simulations presented. For the parameter values chosen this implies that for $\mu < 5/14$, the algorithm sometimes hard-limits the bias to zero, while for $\mu > 9/14$ the algorithm hard-limits the bias to one. In these cases, the true mean of the generated process is respectively greater or less than μ. Of course, if the algorithm overflows, the results in the previous sections do not hold, and the algorithm is not guaranteed to converge or to produce a stationary process.

Figure 2 presents simulation results for various values of μ. To facilitate comparison, the generated process is centralized using the sample mean and normalized to have unit sample variance. The power spectral density is subsequently computed using the average periodogram method for a window size of 256 taps with 128 points overlap. The figure plots the computed sample power spectral density. The ideal power spectral density corresponding to Eq. 3 is also plotted for the purposes of comparison. The figure only plots the results for $\mu = 0.1, 0.2, 0.3, 0.4$. The results for $\mu = 0.5$ coincide with $\mu = 0.4$ since there is no overflow for both values, and the results for $\mu > 0.5$ coincide with the ones for $1 - \mu$.

The simulations confirm that if μ is within the constraints of Eq. (19) the sample power spectral density of the generated process is the one desired. If μ is outside the constraints, the algorithm overflows, and the sample mean and the sample power spectral density

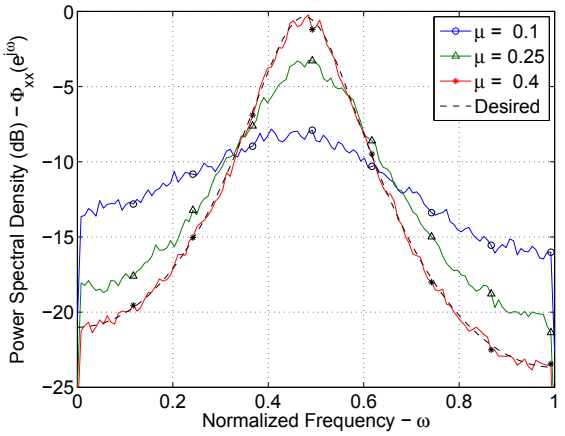

Fig. 2. Simulation results for a two-pole process with $a_1 = 0.1$ and $a_2 = -0.5$, for various values of μ. The plots present the experimental power spectrum, centralized by removing the sample mean and normalized such that the sample variance is unity. The dashed line plots the desired spectrum properly normalized. The results for $\mu = 0.5$ (not plotted) coincide with the results for $\mu = 0.4$, as expected. The results for $\mu > 0.5$ coincide with the results for $1 - \mu$.

do not coincide with the algorithm parameters.

6. REFERENCES

[1] S. Dey and A. V. Oppenheim, "Frequency shaped randomized sampling," in *Proc. IEEE Int. Conf. Acoustics Speech and Signal Processing (ICASSP 2007)*, Honolulu, Hawaii, April 2007, IEEE.

[2] M. R. Said and A. V. Oppenheim, "Discrete-time randomized sampling," in *Proc. Int. Conf. on Electronics, Circuits, and Systems (ICECS-2001)*, Malta, September 2001.

[3] M. I. Skolnik, Ed., *Radar Handbook*, McGraw-Hill, second edition, January 1990.

[4] J. L. Martins De Carvalho and J. M. C. Clark, "Characterizing the autocorrelations of binary sequences," *IEEE Transactions on Information Theory*, vol. 29, no. 4, pp. 502–508, July 1983.

[5] G. Grunwald, R. J. Hyndman, and L. Tedesco, "A unified view of linear ar(1) models," Research report, Department of Statistics, University of Melbourne, June 1996.

[6] E. McKenzie, "Some simple models for discrete variate time series," *Water Resources Bulletin*, vol. 21, no. 4, pp. 645–650, 1985.

[7] E. McKenzie, "Discrete variate time series," in *Handbook of Statistics*, C.R. Rao and D.N. Shanbhag, Eds., pp. 573–606. Elsevier Science B.V., Amsterdam, 2003.

[8] X. K. Karakostas and H. P. Wynn, "On the covariance function of stationary binary sequences with given mean," *IEEE Transactions on Information Theory*, vol. 39, no. 5, pp. 1684–1687, September 1993.

[9] L. J. Emrich and M. R. Piedmonte, "A method for generating high-dimensional multivariate binary variables," *The American Statistician*, vol. 45, no. 4, pp. 302–304, Nov. 1991.

[10] C. G. Park, T. Park, and D. W. Shin, "A simple method for generating correlated binary variates," *The American Statistician*, vol. 50, no. 4, pp. 306–310, Nov. 1996.

[11] S. D. Oman and D. M. Zucker, "Modelling and generating correlated binary variables," *Biometrika*, vol. 88, no. 1, pp. 287–290, 2001.

[12] M. A. Al-Osh and S. J. Lee, "A simple approach for generating correlated binary variates," *J. Statist. Comput. Simul.*, vol. 70, pp. 231–255, 2001.

[13] B. F. Qaqish, "A family of multivariate binary distributions for simulating correlated binary variables with specified marginal means and correlations," *Biometrika*, vol. 90, no. 2, pp. 455–463, 2003.

[14] N. Rao Chaganty and H. Joe, "Range of correlation matrices for dependent bernoulli random variables," *Biometrika*, vol. 93, no. 1, pp. 197–206, 2006.

A. OVERFLOW CONSTRAINTS

The sum in (18) is maximized when $x_c[n-k] = 1-\mu$ for all positive a_k and $x_c[n-k] = -\mu$ for all negative a_k. Similarly, the sum is minimized when $x_c[n-k] = -\mu$ for all positive a_k and $x_c[n-k] = 1-\mu$ for all negative a_k. To guarantee $x_b[n]$ stays within the bounds, it is sufficient to ensure that the maximum of the sum is less than $1-\mu$ and that the minimum is greater than $-\mu$, which implies:

$$(1-\mu)\sum_{k\in A^+} a_k - \mu\sum_{k\in A^-} a_k < 1-\mu, \tag{21}$$

$$\mu\sum_{k\in A^+} a_k - (1-\mu)\sum_{k\in A^-} a_k < \mu, \tag{22}$$

in which the sets $A^+ = \{k|a_k \geq 0\}$ and $A^- = \{k|a_k < 0\}$ denote the indices of the positive and negative coefficients, respectively.

The sums of the positive and the negative coefficients can also be used to express the sum of the coefficients and the sum of their magnitude using:

$$\sum_{k=1}^{p} a_k = \sum_{k\in A^+} a_k + \sum_{k\in A^-} a_k \tag{23}$$

$$\sum_{k=1}^{p} |a_k| = \sum_{k\in A^+} a_k - \sum_{k\in A^-} a_k, \tag{24}$$

Substituting into (21) and (22), and rearranging:

$$\sum_{k=1}^{p} |a_k| + (1-2\mu)\sum_{k=1}^{p} a_k < 2-2\mu \tag{25}$$

$$\sum_{k=1}^{p} |a_k| + (2\mu-1)\sum_{k=1}^{p} a_k < 2\mu. \tag{26}$$

Combined with the bounds on the sum of the coefficients,

$$-\sum_{k=1}^{p} |a_k| \leq \sum_{k=1}^{p} a_k \leq \sum_{k=1}^{p} |a_k|, \tag{27}$$

the constraints produce Fig. 1. In the figure it is demonstrated that only one of (26) and (25) is an active constraint for any given μ. Thus, the constraints can be summarized using the inequality in Eq. (19). It is important to note that if $\sum_{k=1}^{p} |a_k| < 1$ it is impossible to determine a mean μ for which the algorithm can violate both sides of the inequality (18). This is not the case if $\sum_{k=1}^{p} |a_k| > 1$.