

# A Fast and Accurate Rayleigh Fading Simulator

Christos Komninakis

Applied Wave Research, Inc.  
1960 E. Grand Ave. Suite 430  
El Segundo, CA 90245

Email: [chkomn@ee.ucla.edu](mailto:chkomn@ee.ucla.edu)

URL: <http://www.ee.ucla.edu/~chkomn>

**Abstract**—This paper presents a simulator that efficiently generates correlated complex Gaussian variates with a power spectral density as described by Jakes in [1]. The simulator consists of a fixed IIR filter followed by a variable polyphase interpolator, to accommodate different Doppler rates. The IIR filter was designed using an iterative optimization technique known as the ellipsoid algorithm, following an optimization technique that is more generally applicable and can be used to approximate any given magnitude frequency response. Software to implement both the IIR filter design technique and the complex Rayleigh fading simulator itself are available at the author’s website or by email.

## I. INTRODUCTION

In wireless transmission scenarios where a receiver is in motion relative to a transmitter with no line-of-sight path between their antennas it is often the case that Rayleigh fading is a good approximation of realistic channel conditions. The term Rayleigh fading channel refers to a multiplicative distortion  $h(t)$  of the transmitted signal  $s(t)$ , as in  $y(t) = h(t) \cdot s(t) + n(t)$ , where  $y(t)$  is the received waveform and  $n(t)$  is the noise. This paper discusses how to design a simulator that mimics a sampled version of the channel waveform  $h(t)$  in a statistically accurate and computationally efficient fashion.

The channel waveform  $h(t)$  is modeled as a wide-sense-stationary complex Gaussian process with zero-mean, which makes the marginal distributions of the phase and amplitude at any given time uniform and Rayleigh respectively, hence the term Rayleigh fading. The autocorrelation properties of the random process  $h(t)$  are governed by the Doppler frequency  $f_D$ , as in [1]:

$$R(\tau) = E\{h(t)h^*(t - \tau)\} \sim \mathcal{J}_0(2\pi f_D \tau), \quad (1)$$

where  $\mathcal{J}_0(\cdot)$  is the zero-order Bessel function of the first kind. This gives rise to the well-known, non-rational power spectral density (PSD) of the channel process

$$S_{hh}(f) = \begin{cases} \frac{1}{\pi f_D} \frac{1}{\sqrt{1 - \left(\frac{f}{f_D}\right)^2}}, & |f| < f_D \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

Even when the transmission rate is high enough to make the channel frequency-selective (i.e., more than one time-varying channel taps, unlike above), a good channel model is the WSSUS model of Bello [2]. According to this, each channel tap is a zero-mean complex Gaussian random process like  $h(t)$

described above, uncorrelated with –and thus also independent from– any other tap process, but having time-autocorrelation as described by (1). It is important to note that even if there is a significant line-of-sight component in the channel, the only change in the above model of multiplicative distortion is that each channel coefficient (tap) now has non-zero mean. However, its correlation properties are still described by (1), and its PSD is given by (2) to within an additive constant.

The problem of efficiently generating one –or in the case of frequency selective channels more than one– random complex Gaussian process with statistics as described above for the purpose of simulating a wireless channel has been approached in three main ways. The first approach, known as the “sum of sinusoids” was proposed by Jakes in [1], and amounts to the superposition of a number of sinusoids having equal amplitudes and random uniformly distributed phases in order to generate  $h(t)$ . This approach was refined in [3], and extended somewhat to generate multiple uncorrelated fading processes needed in frequency-selective channels. It was recently improved with additional randomization in [4] and [5]. This method, however, is computationally cumbersome due to the large number of expensive  $\sin(\cdot)$  function calls needed in the simulation.

The second approach to generating correlated complex Gaussians with the PSD in (2) was presented in [6]. The idea was to multiply a series of independent complex Gaussian variables by a frequency mask equal to the square root of the spectral shape in (2). Then the resulting sequence is zero-padded and an inverse FFT is applied. The resulting series of variables is still Gaussian by virtue of the linearity of the IFFT, and has the desired spectrum (2), and hence also the autocorrelation of (1). Computationally this approach is quite efficient, since the heaviest effort is required by the IFFT, which only costs  $\mathcal{O}(N \log_{\epsilon} N)$  operations, where  $N$  is the number of time-domain sampled Rayleigh channel coefficients. One disadvantage of the IFFT method is its block-oriented nature, requiring all channel coefficients to be generated and stored before the data is sent through the channel, which increases memory requirements and precludes continuous transmission.

The third approach, followed also in this paper, adopts the filtering of independent Gaussian variables, easily generated in a pseudorandom Gaussian generator, by a filter with frequency response equal to the square root of (2). Since the filter

performs a linear operation, the resulting sequence remains Gaussian, with a spectrum  $S_{out}(f) = S_{in}(f)|H(f)|^2$ , where  $|H(f)|^2$  is the squared magnitude response of the filter, chosen equal to (2). If the input sequence is independent (flat PSD), the spectral shape of the output Gaussian sequence will follow (2), as desired. There are two main remaining design challenges, and this paper addresses both of them in a way that guarantees accuracy and computational efficiency.

First, note that the sampled channel waveform is band-limited to a discrete frequency of  $f_D T$ , where  $1/T$  is the sampling rate. For many common applications, this discrete frequency is very small: for instance, for a system at 900 MHz, with a vehicular speed of 50 mph,  $f_D = 67$  Hz, and at a sampling rate of, say, 1 Msample/sec, the discrete Doppler rate becomes  $f_D T = 0.000067$ . This means that the filter used for spectral shaping of the sequence would have to be extremely narrowband, with a very sharp cutoff and infinite attenuation in the stopband. While these requirements would lead to an impractically long FIR filter, they can be satisfied a lot more easily with an IIR filter combined with a nearly ideal interpolator, similar to [7]. The combination of IIR filter and polyphase interpolator employed here presents a more accurate and computationally faster solution than a FIR filter.

The second main source of problems is that the square root of the spectrum in (2) –the target filter magnitude response– is irrational, and does not lend itself to any of the straightforward filter design methods. All-pole or auto-regressive (AR) filter design [8] requires high orders to approximate well the autocorrelation at large lags. Fortunately, a technique developed in a 1970 paper by Steiglitz [9] allows the design of an IIR filter with arbitrary magnitude response. This technique is more general than is needed here, and we present it briefly adjusted to the needs of this particular problem, along with the iterative optimization algorithm used to converge to the optimum solution.

The paper is organized as follows. Section II presents the architecture of the simulator and discusses the polyphase interpolator. Section III describes the fixed IIR filter design algorithm. Section IV presents simulation results using the proposed system to simulate Rayleigh fading for various Doppler rates and compares them to results known from theory. Finally, Section V concludes the paper.

## II. SIMULATOR DESCRIPTION

The Rayleigh fading simulator is composed of two basic parts and a scaling factor, as shown in the block diagram of Fig. 1. First, an IIR filter of relatively low order, designed such that its squared magnitude response approximates the spectrum in (2) for a given discrete maximum Doppler rate of  $\rho = 0.2$ . Following that, a polyphase interpolator ideally resamples the process, such that any discrete Doppler rate desired by the user can be approximated with minimum computational effort. Obviously, this structure can be replicated for as many paths (i.e., complex channel taps) as the user desires, each with possibly distinct Doppler rates, in order to simulate a time-varying frequency-selective channel.

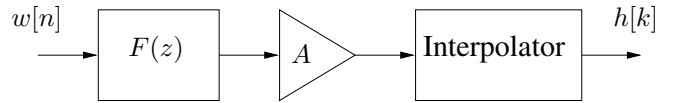


Fig. 1. The system block diagram for the proposed Rayleigh fading simulator. Sampled complex white Gaussian noise  $w[n]$  is generated and fed to the fixed filter  $F(z)$ , designed in Section III. Before the polyphase interpolator, a scaling factor  $A$  normalizes the power of the final resulting channel waveform  $h[k] = h(kT)$  to 1.0. Clearly, the sampling rate at the output is intended to be several times larger than that of the input. The two are related by the interpolation factor  $I$ .

Strictly speaking the above structure is not flexible enough to produce a Rayleigh fading waveform with arbitrary discrete Doppler rate  $f_D T$ , since only rational fractions of the chosen fixed rate  $(f_D T)^\circ = \rho = 0.2$  can result from this filtering/interpolation combination. For example, while  $f_D T = 0.05$  and  $f_D T = 0.001$  are simulated exactly by interpolating  $I = 4$  and  $I = 200$  times the output process of the IIR filter, it is extremely cumbersome to simulate exactly  $f_D T = 0.0773$  (interpolation ratio of  $773/2000$ ), and outright impossible for  $f_D T = \pi/100$ , using the above technique. We argue, however, that this poses no serious limitation to a wireless handset designer simulating the effects of fast fading on a mobile receiver, for two main reasons.

First, extreme accuracy in the Doppler rate is not of primary importance in fading simulations testing the reliability of a wireless link: if a design works with  $f_D T = 0.02$  and  $f_D T = 0.04$ , both of which can be easily and accurately simulated (with  $I = 10$  and  $I = 5$ ), then simulating  $f_D T = \pi/100$  becomes a mere esoteric exercise. Secondly, although the C++ routines provided in the author's website implement only integer interpolating factors of  $\rho = (f_D T)^\circ = 0.2$  (e.g., using the provided routines, the simulation for  $f_D T = 0.0773$  would be indistinguishable from that for  $f_D T = 0.06667 = 0.2/3$ ), the routines provided for the design of the IIR filter in the next section can implement *any* fixed Doppler rate, and then the implementation routines can be modified accordingly. For example, the irrational  $f_D T = \pi/100$  can be implemented by designing an IIR filter with bandwidth  $\pi/10 = 0.31415$  and then interpolating by a factor of  $I = 10$ . The software routines available at the author's website implement only integer interpolation factors  $I$ , taken to be  $I = \lfloor \rho/f_D T \rfloor$ , but extension to any rational  $I$  is possible.

The interpolator is implemented as a polyphase filter with a windowed  $\text{sinc}(\cdot)$  function impulse response. We have found that about  $G = 7$  one-sided periods of the impulse response are enough to yield very good results with small computational complexity. In this polyphase implementation, and counting only real multiplications, the cost per complex output sample at the desired Doppler rate  $f_D T = \rho/I$  is:

$$2 \cdot \left( \frac{4 \cdot K}{I} + 2 \cdot G \right), \quad (3)$$

where  $K$  is the number of biquads in the IIR filter. A small constant overhead must be added to the cost of (3), because of a short initialization phase, where the simulator runs idle for a

few samples to eliminate transient behavior in the IIR filter and the interpolator. The memory required is insignificant. For a typical wireless channel case of, say,  $f_D T = 0.01$ , i.e.,  $I = 20$ , and for the  $K = 7$  biquads we implemented, the cost evaluates to 30.8 real multiplications per sample. This compares very favorably against both FIR filtering (an order of 31 would be too small for any  $f_D T$ ) and the "sum-of-sinusoids" method in [1]. For perspective, with a block-IFFT solution, where for a block of, say,  $N = 10000$  samples, the total cost is  $2 \cdot (N \log_2 N)$  real multiplications, amounting to 26.5 real multiplications per complex output sample, even ignoring the cost of the initial Doppler mask multiplications as well as the large memory requirement. Therefore, the complexity of the proposed method is linear, and actually slightly improves for decreasing  $f_D T$ , as can be seen in (3).

### III. IIR FILTER DESIGN

To design an IIR filter approximating the square root of the magnitude frequency response of (2), or, more accurately, the square root of the spectrum of (2) as it is translated into discrete frequency, we follow an approach proposed by K. Steiglitz [9] in 1970. An IIR filter of order  $2K$ , synthesized as a cascade of  $K$  second-order canonic sections (biquads), is designed. We use an iterative optimization procedure known as the ellipsoid algorithm [10] to search for the optimum real coefficients  $a_k, b_k, c_k, d_k, k = 1, \dots, K$  and the scaling factor  $A$ , such that the magnitude response of the filter

$$H(z) = A \prod_{k=1}^K \frac{1 + a_k z^{-1} + b_k z^{-2}}{1 + c_k z^{-1} + d_k z^{-2}}, \quad (4)$$

for  $z = e^{j\omega}$  approaches the desired magnitude response  $Y^d(\omega)$ . For completeness, we summarize the algorithm in [9] for the IIR filter design below.

Set  $Y_i^d = Y^d(\omega_i)$ , where we define  $M+1$  frequency points in the Nyquist interval  $W_i \in [0, 1], i = 0, 1, \dots, M$ , with  $W_i = i/M$ , and their discrete frequency counterparts  $\omega_i = \pi W_i, \omega_i \in [0, \pi]$ . Also define  $z_i = e^{j\omega_i} = e^{j\pi W_i}$ . Then, for the chosen discrete cutoff frequency  $\rho = (f_D T)^o = 0.2$  define  $L = \lfloor 2\rho M \rfloor$  as the largest frequency index not exceeding the chosen discrete Doppler rate  $\rho$ . According to [6] the desired response to be approximated by our filter is:

$$Y_i^d = \begin{cases} \sqrt{\frac{1}{\sqrt{1 - (\frac{i}{L})^2}}}, & i = 0, 1, \dots, L-1 \\ \sqrt{L(\frac{\pi}{2} - \arcsin(\frac{L-1}{L}))}, & i = L \\ 0, & i = L+1, \dots, M \end{cases}$$

where the response for  $i = L$  was determined from the requirement that the *area* under the sampled spectrum be equal to the theoretical case given in (2), as shown on [6].

Next define the vector  $\mathbf{x}$  of length  $4K$ , containing the filter coefficients  $a_k, b_k, c_k, d_k, k = 1, \dots, K$ . Express  $H(z) = AF(z; \mathbf{x})$ , where  $F(z)$  is the product of biquadratic transfer functions in (4), excluding the scale factor  $A$ . The filter design

problem amounts to minimizing the squared error:

$$Q(A, \mathbf{x}) = \sum_{i=0}^M (|AF(z_i; \mathbf{x})| - Y_i^d)^2 \quad (5)$$

Since the optimum scaling factor  $A^o$  is positive, we can differentiate  $Q(A, \mathbf{x})$  in (5) with respect to  $|A|$  and set to zero, to obtain:

$$A^o = |A|^o = \frac{\sum_{i=0}^M |F(z_i; \mathbf{x})| Y_i^d}{\sum_{i=0}^M |F(z_i; \mathbf{x})|^2} \quad (6)$$

Now we have to optimize  $R(\mathbf{x}) = Q(A^o, \mathbf{x})$  in  $4K$  dimensions. If we compute the gradient  $\nabla_{\mathbf{x}} R(\mathbf{x}) = \left\{ \frac{\partial R}{\partial \mathbf{x}_n}, n = 1, \dots, 4K \right\}$ , each partial derivative is given by:

$$\frac{\partial R(\mathbf{x})}{\partial \mathbf{x}_n} = 2A^o \cdot \sum_{i=0}^M (A^o |F(z_i; \mathbf{x})| - Y_i^d) \cdot \frac{\partial |F(z_i; \mathbf{x})|}{\partial \mathbf{x}_n}. \quad (7)$$

To evaluate (7) for the frequency index  $i = 0, \dots, M$  and for the biquad index  $k = 1, \dots, K$ , we have:

$$\frac{\partial |F(z_i; \mathbf{x})|}{\partial a_k} = |F(z_i; \mathbf{x})| \cdot \Re \left\{ \frac{z_i^{-1}}{1 + a_k z_i^{-1} + b_k z_i^{-2}} \right\} \quad (8)$$

$$\frac{\partial |F(z_i; \mathbf{x})|}{\partial b_k} = |F(z_i; \mathbf{x})| \cdot \Re \left\{ \frac{z_i^{-2}}{1 + a_k z_i^{-1} + b_k z_i^{-2}} \right\} \quad (9)$$

$$\frac{\partial |F(z_i; \mathbf{x})|}{\partial c_k} = -|F(z_i; \mathbf{x})| \cdot \Re \left\{ \frac{z_i^{-1}}{1 + c_k z_i^{-1} + d_k z_i^{-2}} \right\} \quad (10)$$

$$\frac{\partial |F(z_i; \mathbf{x})|}{\partial d_k} = -|F(z_i; \mathbf{x})| \cdot \Re \left\{ \frac{z_i^{-2}}{1 + c_k z_i^{-1} + d_k z_i^{-2}} \right\} \quad (11)$$

At this point, the basic quantities for iterative optimization exist. Given a vector  $\mathbf{x}$  we can evaluate  $F(z_i; \mathbf{x}), i = 0, \dots, M$ , and using those compute the optimum scaling factor  $A^o$  from (6). Then, setting  $E_i = A^o |F(z_i; \mathbf{x})| - Y_i^d, i = 0, \dots, M$ , the value of the cost function becomes:

$$R(\mathbf{x}) = \sum_{i=0}^M E_i^2 \quad (12)$$

while from (7) the elements of the gradient vector are:

$$[\nabla_{\mathbf{x}} R(\mathbf{x})]_n = 2A^o \cdot \sum_{i=0}^M E_i \frac{\partial |F(z_i; \mathbf{x})|}{\partial \mathbf{x}_n} \quad (13)$$

in which we substitute one of (8)-(11), depending on whether  $\mathbf{x}_n$  is one of the  $a_k, b_k, c_k$  or  $d_k$  coefficients, for  $n = 1, \dots, 4K$  and  $k = 1, \dots, K$ .

Since it is clear how to evaluate both the cost function and its gradient, we can use a very basic Ellipsoid algorithm [10] to iteratively optimize the filter coefficients and converge to the optimum solution. This algorithm starts at a random point  $\mathbf{x}$  in the  $4K$ -dimensional real space, and defines an initial large ellipsoid matrix  $\mathbf{B}$ , for example  $\mathbf{B} = 100 \cdot \mathbf{I}_{4K}$ . Then it evaluates upper and lower bounds to the optimum solution and keeps track of the distance between them:

$$\beta = \sqrt{[\nabla_{\mathbf{x}} R(\mathbf{x})]^T \mathbf{x} [\nabla_{\mathbf{x}} R(\mathbf{x})]} \quad (14)$$

Since the algorithm guarantees that the distance of the target function from its optimum point is always upperbounded by  $\beta$  above, we keep performing the following ellipsoid updates of the vector  $\mathbf{x}$  and the matrix  $\mathbf{B}$ :

$$\tilde{\mathbf{g}} = \frac{\nabla_{\mathbf{x}}R(\mathbf{x})}{\sqrt{[\nabla_{\mathbf{x}}R(\mathbf{x})]^T \mathbf{x} [\nabla_{\mathbf{x}}R(\mathbf{x})]}} \quad (15)$$

$$\mathbf{x} := \mathbf{x} - \frac{1}{4K+1} \mathbf{B} \tilde{\mathbf{g}} \quad (16)$$

$$\mathbf{B} := \frac{(4K)^2}{(4K)^2 - 1} \cdot \left( \mathbf{B} - \frac{2}{4K+1} \mathbf{B} \tilde{\mathbf{g}} \tilde{\mathbf{g}}^T \mathbf{B} \right) \quad (17)$$

until the distance  $\beta$  of the two bounds becomes less than a specified accuracy  $\epsilon$ . This allows the user to maintain control over the quality of convergence. Furthermore, since a negative lower bound to our quadratic target function in (12) is not informative, we consider the lower bound to be the largest of zero and the target function less the quantity  $\beta$  of (14) at every iteration. If at any time during the iterations of the ellipsoid algorithm the lower bound becomes positive, and especially larger than the specified accuracy, the process must be halted and a different starting point for  $\mathbf{x}$  and/or higher filter order  $K$  have to be picked. A strictly positive lower bound implies that the constraints of our design (possibly too low  $K$  or too large  $M$  for very small  $\epsilon$ ) bound the target function  $R(\mathbf{x})$  away from zero and preclude further minimization.

An important detail is that the algorithm may converge to unstable or non-minimum phase filters, because one or more of the  $K$  biquads have poles or zeros outside the unit circle. Since this is undesirable for the designed filter, if the optimization leads to such biquads, we invert the misbehaving poles or zeros to bring them back into the unit circle, and restart the optimization from that point, as recommended in [9]. In this fashion the filter magnitude response remains unchanged and presumably close to the desired, and the final resulting filter is a stable and minimum-phase design. Figure 2 shows the resulting magnitude response for a filter with  $K = 7$  biquads and specified ellipsoid accuracy of  $\epsilon = 0.01$ . The number of frequency points was  $M + 1 = 501$ . To make discrepancies clear we plot the response in dB.

It should be noted that the ellipsoid algorithm [10] is certainly not the fastest iterative optimization technique, because it is not a descent method. It is similar to a quasi-Newton method with fixed step length, and converges slower than other methods such as BFGS, but despite those shortcomings it was chosen because it possesses several advantages. First, it is extremely simple to code, allowing custom optimization routines based on it to be written in any high level language. Secondly, it is a good method to ascertain feasibility in a fast and straightforward way, and tends to work well when the target function of the optimization is highly non-linear or even non-differentiable (which is not the case here though). And finally, speed of optimization is not the highest priority in this case, since the filter design is only done once; after that, the coefficients of the biquads are stored and obviously not optimized again for every run of the simulator.

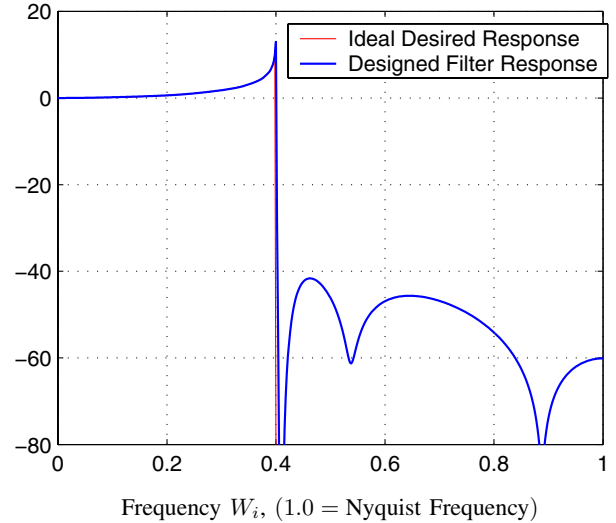


Fig. 2. Magnitude frequency response for the designed filter overlaid against the desired theoretical response. The match is very good.

#### IV. PERFORMANCE EVALUATION

We wrote C++ routines to implement the simulator in its form discussed above, and tested the correlation properties of the simulated Rayleigh fading waveform against the theoretical predictions. Some of the comparisons that can be made are summarized in Fig. 3, where we plot the real and imaginary parts of the autocorrelation of the simulated complex channel waveform. The real part agrees very well with the theoretical autocorrelation given by (1). The imaginary part of the autocorrelation should ideally be zero everywhere, as implied by (1), and indeed we observe it remains extremely low for the simulated waveform. The autocorrelations in Fig. 3, both theoretical and simulated, were computed for  $f_D T = 0.05$ , but similar good agreement is true for any other discrete Doppler rate. Hence, we conclude that the approach taken is rather successful in that with very reasonable computational complexity, the simulated fading waveform captures the statistical behavior expected by theory.

Another statistical characteristic often measured in Rayleigh fading waveforms is their Level Crossing Rate (LCR),  $N_R$ . This is defined in [1] as the expected rate at which the magnitude of the fading waveform crosses a specified signal level  $R$  in the positive direction. Its formal definition for a continuous time fading waveform is:

$$N_R = \int_0^\infty \dot{r} p(R, \dot{r}) d\dot{r} \quad (18)$$

where  $p(R, \dot{r})$  is the joint density function of the amplitude of the fading and its time-derivative, also given in [1]. From the definition and for the spectrum in (2) the LCR is:

$$N_R = \sqrt{2\pi} f_D \lambda e^{-\lambda^2}, \quad (19)$$

where  $\lambda$  is the signal level being crossed normalized with the rms of the fading waveform,  $\lambda = R/R_{rms}$ . For three different discrete Doppler rates  $f_D T = 0.05, 0.01$  and  $0.002$

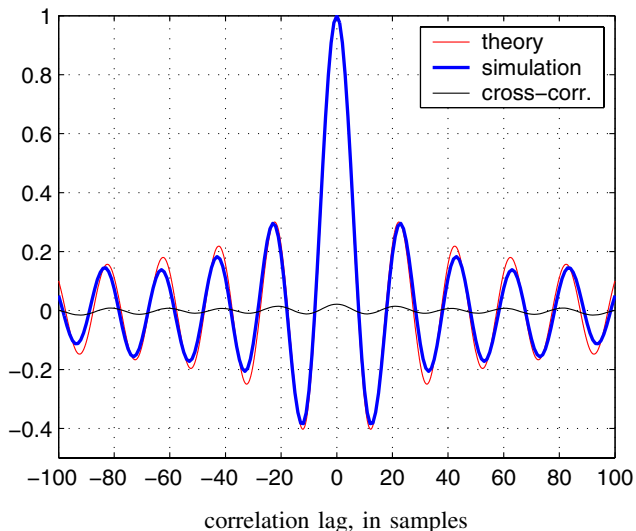


Fig. 3. Real and imaginary part of the autocorrelation function of the sampled Rayleigh fading waveform for Doppler rate  $f_D T = 0.05$ , plotted against theoretical Bessel autocorrelation. The real part compares well to theory and the imaginary part remains insignificant, as expected.

(interpolation factors for our simulator structure  $I = 4, 20, 100$  respectively), we obtain simulated results for the LCR  $N_R$ . These appear in Fig. 4 against the theoretical values of (19).

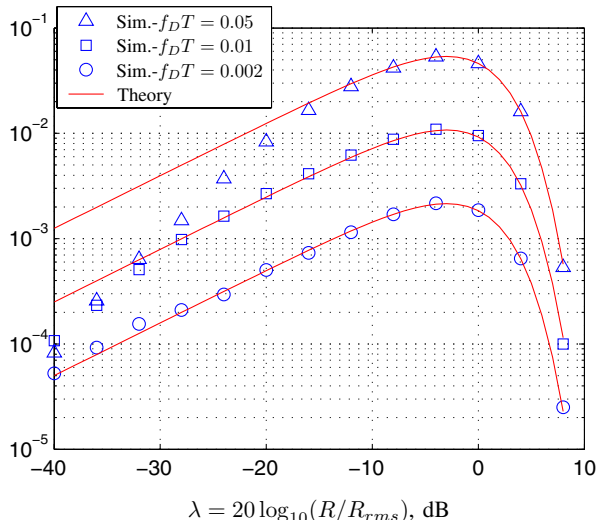


Fig. 4. Level crossing rates  $N_R$  of the amplitude of the sampled Rayleigh fading waveform for Doppler rates  $f_D T = 0.05, 0.01$  and  $0.002$ . Agreement with theory is excellent everywhere for the lowest Doppler rate, but for very high Doppler rate and low crossing levels it is not as good. This is a general problem, unrelated to the particular simulation technique used, and is caused by the sparse sampling of the implied continuous fading waveform.

From Fig. 4 we conclude that the transfer of the level crossing notion from the continuous to the discrete (sampled in time) world is not without consequences. For the finely sampled fading waveform with the very low Doppler rate  $f_D T = 0.002$ , the agreement is excellent for *any* crossing level, because given the slow variation of the fading and the density of the sampling, the representation of the continuous-

time properties in the discrete-time world is accurate. For the middle Doppler rate  $f_D T = 0.01$  the result deviates somewhat from theory for extremely low  $\lambda$ , but is almost identical to a plot in [8] using a large AR model and IFFT to generate the Rayleigh fading waveform. For fast Doppler  $f_D T = 0.05$ , however, and for the low crossing levels (more than 20 dB below the rms amplitude) the more sparse sampling of the fading waveform causes underestimation of the LCR. This is not particular to our simulator, but rather a general problem, arising due to sampling. The reason it surfaces at high Doppler rate and low  $\lambda$  is that the sparsely sampled waveform “misses” some of the very low (and also very short in duration) minima of the implied continuous waveform. A portion of the low extremes are unavoidably overlooked by the sampled fading because of their small duration in continuous time, leading to the underestimation of the LCR.

## V. CONCLUSIONS

This paper designs a simulation structure for correlated complex baseband Rayleigh fading waveforms. A design method for arbitrary IIR filters and the ellipsoid algorithm were applied to obtain a fixed IIR filter (in biquad-form for robustness). This filter is used to generate a fading waveform at a fixed Doppler rate. Subsequently, fast polyphase interpolation accommodates the variety of Doppler rates that may be desired. This fixed filter - variable interpolator system ensures adaptability, good statistical properties, and computational efficiency. Simulated results compare well against theoretically expected respective ones, which confirms the validity of the proposed simulation technique. Subroutines in Matlab (filter design) and C++ (simulator) are available at the author’s website or upon request by email.

## REFERENCES

- [1] W. C. Jakes, Jr. *Microwave Mobile Communications*. John Wiley & Sons, NY, 1974.
- [2] P. A. Bello. Characterization of randomly time-variant linear channels. *Trans. on Communication Systems*, CS(11):360–393, Dec. 1963.
- [3] P. Dent, G. E. Bottomley, and T. E. Croft. Jakes fading model revisited. *Electronics Letters*, 19(13):1162–1163, June 1993.
- [4] M. F. Pop and N. C. Beaulieu. Design of wide-sense stationary sum-of-sinusoids fading channel simulators. in *Proc. of IEEE ICC*, Vol. 2, April 2002, pp. 709–716.
- [5] Y. R. Zheng and C. Xiao. Simulation models with correct statistical properties for Rayleigh fading channels. *IEEE Trans. on Comm.*, 51(6):920–8, June 2003.
- [6] D. J. Young and N. C. Beaulieu. On the generation of correlated Rayleigh random variates by Inverse Discrete Fourier Transform. *Proc. of ICUPC - 5th International Conference on Universal Personal Communications*, vol. 1, Cambridge, MA, 29 Sept.-2 Oct. 1996, pp. 231–235.
- [7] A. Anastasopoulos and K. M. Chugg. An efficient method for simulation of frequency-selective isotropic Rayleigh fading. *Proc. of Veh. Tech. Conf.*, Phoenix, AZ, May 1997, pp. 539-543.
- [8] K. E. Baddour and N. C. Beaulieu. Autoregressive models for fading channel simulation. *IEEE Global Telecommunications Conference, Globecom 2001*, 2:1187–1192.
- [9] K. Steiglitz. Computer-aided design of recursive digital filters. *IEEE Trans. on Audio and Electroacoustics*, AU-18:123–129, June 1970. Reprinted in *Digital Signal Processing*, L. R. Rabiner and C. R. Rader, editors, IEEE Press, 1972, pp. 143–149.
- [10] S. Boyd and L. Vandenberghe. *Convex optimization*. Course Reader for EE236B, Nonlinear Programming, UCLA, 1997.