METHODS AND FRAMEWORKS FOR SAMPLING $\mathcal{G}_I^0$ DATA

D. Chan*, A. Rey*, J. Gambini†, J. Cassetti ‡ and A. C. Frery§
*Universidad Tecnológica Nacional, Facultad Regional Buenos Aires, Ciudad Autónoma de Buenos Aires, Argentina
Email: {mchan,arey}@frba.utn.edu.ar
†Departamento de Ingeniería Informática - Instituto Tecnológico de Buenos Aires, Buenos Aires, Argentina and
Dpto. de Ingeniería en Computación, Universidad Nacional de Tres de Febrero, Pcia. de Buenos Aires, Argentina
‡Instituto de Desarrollo Humano, Universidad Nacional de General Sarmiento, Buenos Aires, Argentina
§Laboratório de Computação Científica e Análise Numérica, Universidade Federal de Alagoas, Maceió, AL, Brazil
Email: {juliana.gambini, julia.cassetti,acfrery}@gmail.com

***Abstract*—The $\mathcal{G}_I^0$ distribution is a competitive tool for SAR image description. This distribution is useful for describing speckled imagery because it models adequately areas with different degrees of texture. Data simulation is crucial for the development of new methods of automatic interpretation of this type of images. We compare four alternatives for generating data under the $\mathcal{G}_I^0$ distribution. The experiments are performed on a variety of programming languages and, a number of criteria to test the fidelity of the generated data are applied.**

***Index Terms*—Random variable generation, SAR image modeling, $\mathcal{G}_I^0$ distribution**

I. Introduction

The statistical modeling of SAR (Synthetic Aperture Radar) data is a well-known and powerful strategy for understanding this kind of images; the comprehensive review by Gao [1] discusses many of the available models. The $\mathcal{G}_I^0$ family of distributions [2] is an attractive option because it models areas with different degrees of texture, while not requiring the use of Bessel functions, which may not be available or exhibit numerical instabilities. Three parameters index the $\mathcal{G}_I^0$ distribution: scale (related to the brightness), the number of looks which is associated with the signal-to-noise ratio, and arguably the most important one that is connected to the degree of texture and with the tail index of this distribution [3], [4]. They are denoted, respectively by γ, L and α.

Many applications use simulated SAR data. The analysis of properties of parameter estimation procedures is hardly done on a theoretical basis only [5], [6]. Computer simulations are used to assess methods of automatic interpretation [7], [8].

We check the reliability reliable and efficiency of random number generators in terms of goodness of fit and computational requirements. The quality of the simulated data is measured using first and second order moments, goodness of fit tests, the maximum sample distribution and confidence intervals coverage [9]. Moreover, we experiment four programming languages examining their advantages and disadvantages. We study the noisiest case, namely, the single-look ($L = 1$) distribution.

The paper unfolds as follows. Section II recalls useful properties of the $\mathcal{G}_I^0$ model, presents the expressions for data generation, and explains how they are compared. Section III presents the results. Finally, in Section IV we draw conclusions and suggest future work.

II. Methodology

Under the multiplicative model, the return in SAR images can be modeled as the product of two independent random variables, one corresponding to the backscatter X and the other to the speckle noise Y. Thus, $Z = XY$ describes the return in each pixel.

Sensible assumptions are (i) the backscatter follows a reciprocal Gamma law (denoted $X \sim \Gamma^{-1}(-\alpha, \gamma)$), and (ii) the speckle obeys an unitary mean Gamma distribution ($Y \sim \Gamma(L, L)$). With this, the return follows a $\mathcal{G}_I^0$ distribution: $Z \sim \mathcal{G}_I^0(\alpha, \gamma, L)$ [2], whose density function is

$$f_{\mathcal{G}_I^0}(z) = \frac{L^L \Gamma(L-\alpha)}{\gamma^\alpha \Gamma(-\alpha)\Gamma(L)} \frac{z^{L-1}}{(\gamma + zL)^{L-\alpha}}, \qquad (1)$$

with $-\alpha, \gamma, z > 0$ and $L \geq 1$. In the single look case

this expression takes the form

$$f_{\mathcal{G}_I^0}(z) = \frac{-\alpha}{\gamma}\left(\frac{x}{\gamma}+1\right)^{\alpha-1}, \tag{2}$$

which is a Generalized Pareto Type II distribution, a particular case of a power law distribution [3].

The r-order moment for this $\mathcal{G}_I^0$ distribution is

$$\mathrm{E}(Z^r) = \left(\frac{\gamma}{L}\right)^r \frac{\Gamma(-\alpha-r)}{\Gamma(-\alpha)}\frac{\Gamma(L+r)}{\Gamma(L)}, \tag{3}$$

provided $\alpha < -r$, and it is infinite otherwise.

A. Data generation techniques

As previously stated, a $\mathcal{G}_I^0(\alpha,\gamma,1)$ random variable is the product of two independent random variables. We use the definition of the reciprocal Gamma law: if $U \sim \Gamma(\eta,\omega)$, then $V = 1/X \sim \Gamma^{-1}(-\eta,\omega)$. Also, the χ^2 distribution is a particular case of a Gamma distribution: a χ^2_m law is a $\Gamma(m/2,1/2)$ distribution. Finally, the scale property of Gamma random variables is also useful: if $X \sim \Gamma(m,n)$ then $aX \sim \Gamma(m,n/a)$ for $a > 0$. In face of this, we propose the following two ways of data generation, one based on Gamma laws, and another using χ^2 deviates:

- **Γ-generator:** Sample from $Z = Y/X'$ where X and Y are independent random variables such that $X' \sim \Gamma(-\alpha,\gamma)$ and $Y \sim \Gamma(1,1)$.
- **χ^2-generator:** Sample from $Z = \gamma X/Y$ where X and Y are independent random variables such that $X \sim \chi^2_2$ and $Y \sim \chi^2_{-2\alpha}$.

A third way of obtaining $\mathcal{G}_I^0$ deviates is using its relationship with the Snedecor's $\mathcal{F}$ variable: a quotient of two independent χ^2 random variables. With this we have the F-generator: $Z = -\gamma U/\alpha$, where $U \sim \mathcal{F}(2,-2\alpha)$.

The fourth way of generation uses the fact that the $\mathcal{G}_I^0(\alpha,\gamma,1)$ distribution is a particular case of Generalized Pareto Type II distribution [10]. Thus, we propose the P-generator as $Z \sim \mathrm{P(II)}(0,\gamma,-\alpha)$, with P(II) denoting Pareto Type II distribution.

All the programming languages and platforms henceforth considered provide native functions for sampling from the Gamma, χ^2 and Pareto distributions.

B. Programming languages

Simulations were performed using four programming languages:

- **R** [11], version 3.3.1, is a software environment for statistical computing and image processing, among other applications. It runs on several platforms and provides statistical and graphical tools.
- **Julia** [12], version 0.5.0, is a high-performance dynamic programming language for technical computing. It provides a compiler, distributed parallel execution with good numerical accuracy, and a library of mathematical functions.
- **Ox** [13], version 7.00 , is an object-oriented programming language with an extensive statistical function library. It is available for several platforms.
- **MatLab** [14], version R2016a, is a numerical computing framework. It has its own programming language and an image processing toolbox. MatLab allows a variety of matrix operations, and the creation of user interfaces.

The first three are free, and all of them are able to connect with programs written in other languages.

C. Comparison criteria

1) Moments accuracy: We consider $(\alpha,\gamma) \in \{-8,\,-5,\,-3\}\times\{0.1,\,1,\,10,\,100,\,1000\}$. For each point we generate 1000 samples of size 500, and we compute the sample mean and variance for each. Using (3), we know the exact values of the mean and the variance, so it is possible to study the deviation of the observed values from the true ones as an error. Denoting n the amount of samples, θ and $\hat{\theta}$ the true value and its estimation, respectively.

- The **mean squared error** is one of the most popular measures of error or population variance. It is defined as $\widehat{\mathrm{MSE}}(\widehat{\theta}) = n^{-1}\sum_{i=1}^n(\widehat{\theta}_i-\theta)^2$.
- The **mean absolute error** is defined as $\widehat{\mathrm{MAE}}(\widehat{\theta}) = n^{-1}\sum_{i=1}^n|\hat{\theta}_i-\theta|$. It is often applied in image processing instead of the mean squared error, because its memory requirements are noticeably smaller [15].
- The **maximum absolute error** measures the worst error case. It is defined as $\widehat{\mathrm{MxAE}}(\widehat{\theta}) = \max_{1\le i\le n}\{|\hat{\theta}_i-\theta|\}$.

The $\mathcal{G}_I^0$ distribution is heavy-tailed [4], so we are interested in the behavior of the tail of the observed data.

2) Tails behavior: With the double purpose of simplifying the calculations and making the results comparable, in this subsection we employ γ^* such that $E(Z) = 1$, which is implied by (3) and it is given by:

$$\gamma^* = 1-\alpha. \tag{4}$$

It is worth noting that using $X \sim \mathcal{G}_I^0(\alpha,1-\alpha,L)$ along with $Y \sim \Gamma(L,L)$ leads to the Kummer-$\mathcal{U}$ distribution for the return [16].

We generate 1000 samples of size 1000 for each $\alpha \in \{-8, -5, -3, -1.5\}$, using the proposed generators in each programming language. We then register the values of each maximum, third quartile, 90th percentile and the limits of 95 % level confidence interval for the last two order statistics based on bootstrap [17]. The reason for selecting these two order statistics is the heavy-tailed property of this distribution. Then, we compute how many times the true value belongs to the sample confidence intervals: the coverage.

Moreover, we assess the goodness of fit of each generated sample to the true distribution. The classical option is the Kolmogorov-Smirnov (KS) test [18], based on the maximum difference between the empirical cumulative distribution function and the cumulative distribution function specified by the selected distribution for the test. Since it is known that the KS test is more sensitive in the center of the distribution than in the tails, we use the Anderson-Darling (AD) test [19] which is more robust and it weights the tails. As the behavior of the extreme information is relevant for us, we also consider the AD test to evaluate the goodness of fit of both the complete sample, and the maxima. An attractive feature of this test is that its statistics distribution does not depend on the underlying distribution when the parameters are known.

Finally, taking into account samples of size $n \in \{50, 100, 500, 1000\}$, we generate 1000 samples of size 1000 of sample maxima and we analyze its goodness of fit applying both AD and KS tests. This choice is motivated by the absolutely outlier-prone property of the $\mathcal{G}_I^0$ distribution [3], [4].

Being $\{Z_1, Z_2, \ldots, Z_n\}$ independent random variables with $\mathcal{G}_I^0$ distribution, the maximum $U_n = \max_{1\leq i\leq n}\{Z_1, Z_2, \ldots, Z_n\}$ has cumulative distribution function

$$F_{U_n}(u) = (F_z(u))^n = \left[1 - \left(1 + \frac{z}{\gamma}\right)^{\alpha}\right]^n$$

and probability density function

$$\begin{aligned} f_{U_n}(u) &= n\,(F_z(u))^{n-1}\, f_z(u) \\ &= -\frac{n\alpha}{\gamma}\left(1 + \frac{z}{\gamma}\right)^{\alpha-1}\left[1 - \left(1 + \frac{z}{\gamma}\right)^{\alpha}\right]^{n-1}. \end{aligned}$$

3) Processing times: As part of our reckoning, we compute the processing time consumed by each generator for each language, in a computer with processor Intel© Core™, i7-6700K CPU 3.4 GHz, 16 GB RAM, System Type 64 bit operating system.

III. Results

Table I shows the generator which outperforms the others, according to the selected criteria of errors and goodness of fit for each programming language.

TABLE I
THE BEST METHOD FOR EACH LANGUAGE

Criteria	R	Matlab	Ox	Julia
Mean errors	χ^2	χ^2	P	Γ
Variance errors	χ^2	F, P	Γ	P
q_3 coverage	F	χ^2, Γ, F	F	P
p_{90} coverage	χ^2	χ^2, Γ, F	F	P
AD-KS raw data	χ^2	χ^2	χ^2	Γ, F
AD-KS maxima	χ^2, Γ	P	Γ	χ^2

Notice that when we refer to the mean and variance errors, we take into account the generators that most frequently produces the smallest error among all the considered parameter values.

Notice that there is no overall best generator; it depends on the programming language. The coverage percentage of the confidence intervals for the third quartile and the 90th percentile have a similar behavior for all the involved generators, though.

Table II exhibits the average time consumed to generate four samples of size 10^6, considering $(\alpha, \gamma^*) \in \{(-8, 7), (-5, 4), (-3, 2), (-1.5, 0.5)\}$ for each generator, measured in seconds.

TABLE II
PROCESSING TIME

	R	Matlab	Ox	Julia
Γ	0.225	0.110	0.416	0.130
χ^2	0.215	0.116	0.440	0.164
F	0.225	0.109	0.417	0.156
P	**0.088**	**0.052**	**0.293**	**0.108**

The generator that produces the shortest computation time is indicated in bold. The P generator outperforms the others in computational effort.

IV. Conclusions and Future Work

We study methods for generating data that follow the $\mathcal{G}_I^0$ distribution for the single-look case.

In order to compare these generators, we analyze error measures, goodness of fit, tails behavior and processing time.

We conclude that the choice of the most appropriate generator is largely determined by the programming language.

By programming language

The χ^2 generator is better than the others in the R language in most of the considered features. The χ^2-generator, followed by the F-generator, is the most suitable with MatLab. Both F and Γ generators prevail when using Ox. The P generator is the best for Julia language.

Alternatively we can classify the criteria in the following four categories: processing time, moments fidelity, quantiles coverage and, raw and sample maxima data goodness of fit.

The P generator outperforms the others in processing time for all the programming languages, although this advantage is more evident in R and Matlab. The χ^2 generator has the best performance, except in Julia language, in terms of goodness of fit for the case of raw data. The P and χ^2 generators are the best in terms of moments fidelity. The F and χ^2 generators do better than the others regarding to tail behavior. The Γ and χ^2 generators are both preferable considering the goodness of fit of maxima.

We conclude that the most suitable technique is the P-generator according to the simulation time consumed.

However, in terms of fidelity to the distribution, we strongly recommend the χ^2 generator.

As future work, we are interested in dealing with non single-look case simulations.

REFERENCES

[1] G. Gao, "Statistical modeling of SAR images: A survey," *Sensors*, vol. 10, no. 1, pp. 775–795, 2010.

[2] A. Frery, H. Müller, C. Yanasse, and S. Sant'Anna, "A model for extremely heterogeneous clutter," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 35, no. 3, pp. 648–659, 1997.

[3] J. Rojo, "Heavy tailed densities," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 5, no. 1, pp. 30–40, 2013.

[4] J. Gambini, J. Cassetti, M. Lucini, and A. Frery, "Parameter estimation in SAR imagery using stochastic distances and asymmetric kernel," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 1, pp. 365–375, 2015.

[5] M. Silva, F. Cribari-Neto, and A. C. Frery, "Improved likelihood inference for the roughness parameter of the GA0 distribution," *Environmetrics*, vol. 19, no. 4, pp. 347–368, 2008. [Online]. Available: http://www3.interscience.wiley.com/cgi-bin/abstract/114801264/ABSTRACT

[6] K. L. P. Vasconcellos, A. C. Frery, and L. B. Silva, "Improving estimation in speckled imagery," *Computational Statistics*, vol. 20, no. 3, pp. 503–519, 2005.

[7] J. Naranjo-Torres, J. Gambini, and A. C. Frery, "The geodesic distance between GI0 models and its application to region discrimination," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 10, no. 3, pp. 987–997, 2017.

[8] L. Gomez, L. Alvarez, L. Mazorra, and A. C. Frery, "Fully PolSAR image classification using machine learning techniques and reaction-diffusion systems," *Neurocomputing*, mar 2017.

[9] H. Shin, Y. Jung, C. Jeong, and J. Heo, "Assessment of modified Anderson-Darling test statistics for the Generalized Extreme Value and Generalized Logistic distributions," *Stochastic Environmental Research and Risk Assessment*, vol. 26, no. 1, pp. 105–114, 2012.

[10] M. Newman, "Power laws, Pareto distributions and Zipf's law," *Contemporary Physics*, vol. 46, pp. 323–351, 2005.

[11] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2016. [Online]. Available: https://www.R-project.org/

[12] J. Bezanson, A. Edelman, S. Karpinski, and V. Shah, *Julia: A Fresh Approach to Numerical Computing*, Cornell University Library, 2014. [Online]. Available: http://julialang.org/

[13] J. Doornik and M. Ooms, *Introduction to Ox*. Timberlake Consultants Press, London, 2006.

[14] The MathWorks, Inc., *Matlab, The Language of Technical Computing*, The MathWorks, Inc., 2016. [Online]. Available: http://www.mathworks.com/

[15] R. Roy, "Comparison of different techniques to generate Normal random variables," *Journal of East Central Europe*, vol. 545, pp. 5–6, 2002.

[16] L. Bombrun and J.-M. Beaulieu, "Fisher distribution for texture modeling of polarimetric SAR data," *IEEE Geoscience and Remote Sensing Letters*, vol. 5, no. 3, pp. 512–516, jul 2008.

[17] J. A. Villaseñor-Alva and E. González-Estrada, "A Bootstrap goodness of fit test for the Generalized Pareto distribution," *Computational Statistics & Data Analysis*, vol. 53, no. 11, pp. 3835–3841, 2009.

[18] F. Massey Jr, "The Kolmogorov-Smirnov test for goodness of fit," *Journal of the American Statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.

[19] N. Razali and Y. Wah, "Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests," *Journal of Statistical Modeling and Analytics*, vol. 2, no. 1, pp. 21–33, 2011.