

Rician K -Factor-Based Analysis of XLOS Service Probability in 5G Outdoor Ultra-Dense Networks

Hatim Chergui, *Member, IEEE*, Mustapha Benjillali, *Senior Member, IEEE*,
and Mohamed-Slim Alouini, *Fellow, IEEE*

Abstract

In this report, we introduce the concept of Rician K -factor-based radio resource and mobility management for fifth generation (5G) ultra-dense networks (UDN), where the information on the gradual visibility between the new radio node B (gNB) and the user equipment (UE)—dubbed X-line-of-sight (XLOS)—would be required. We therefore start by presenting the XLOS service probability as a new performance indicator; taking into account both the UE serving and neighbor cells. By relying on a lognormal K -factor model, a closed-form expression of the XLOS service probability in a 5G outdoor UDN is derived in terms of the multivariate Fox H-function; wherefore we develop a GPU-enabled MATLAB routine and automate the definition of the underlying Mellin-Barnes contour via linear optimization. Residue theory is then applied to infer the relevant asymptotic behavior and show its practical implications. Finally, numerical results are provided for various network configurations, and underpinned by extensive Monte-Carlo simulations.

Index Terms

5G, GPU, multivariate Fox H-function, Rician K -factor, UDN, XLOS service probability.

This is a companion technical report of [1].

H. Chergui and M. Benjillali are with the Communication Systems Department, INPT, Rabat, Morocco. [e-mail: chergui@ieee.org, benjillali@ieee.org].

M.-S. Alouini is with King Abdullah University of Science and Technology (KAUST), Computer, Electrical and Mathematical Science and Engineering Division (CEMSE), Thuwal 23955- 6900, Saudi Arabia. [e-mail: slim.alouini@kaust.edu.sa].

I. INTRODUCTION

THE emergence of massive-MIMO and millimeter-wave (mmWave) as key enablers for 5G ultra-dense networks [2] will certainly prompt the reshaping of radio resource and mobility management algorithms, wherefore a new set of measured quantities might be required as inputs. In this context, the Rician K -factor can serve as an accurate channel metric to measure the gradual visibility condition of a radio link, termed *X-line-of-sight* (XLOS) here, and encompassing LOS, obstructed-LOS (OLOS) and non-LOS (NLOS) as discrete regimes. In localization services for instance, while the availability of a LOS path is quintessential for the classical triangulation-based schemes such as time-of-arrival (TOA) and direction-of-arrival (DOA), the massive-MIMO-based space-time processing approaches can deliver very concise localization thanks to the high angular resolution of the large scale antennas, and may therefore operate in the worst OLOS/NLOS conditions, yet at the expense of a higher complexity [3]. To optimize the computational cost, an operator may adopt a hybrid network configuration where, according to a fine-tuned target K -factor threshold, the 5G gNB can switch between the simpler conventional methods and the massive-MIMO ones. On the other hand, in future UDNs with co-located sub-6GHz/mmWave deployment, operators might configure only an anchor internet of things (IoT) carrier on the sub-6GHz—given the scarcity of spectrum—and offload a great part of the traffic to mmWave stand-alone IoT carriers. For energy savings considerations, the 5G network might prevent IoT devices from performing the periodic inter-frequency measurements and reporting by triggering blind inter-frequency re-selections [4] to a mmWave IoT carrier once the sub-6GHz K -factor exceeds a preset threshold¹. Zooming out from the applications, a mathematical characterization of XLOS is yet to be established.

In this letter, we propose the XLOS service probability as a performance indicator, and start by introducing a broader definition of the concept thereof; accommodating the monitoring of both the UE serving and neighbor cells. By invoking a distance-based lognormal K -factor model for outdoor fixed and/or sporadically moving UEs [6]², we then derive—in terms of the multivariate Fox H-function [7, A.1]—a closed-form expression for the XLOS service probability in a 5G OFDMA-based multi-tier heterogeneous network (HetNet), where a majority of cellular IoT devices and outdoor customer premise equipments (CPEs) are either fixed or changing their locations in an intermittent manner (e.g., position sensors with event-driven reporting) [4], and where the K -factor variations stem also from the movement of the scattering objects (e.g., vehicles, containers, windblown leaves). Finally, the asymptotic behavior highlighting the effect of different network and channel parameters is studied using the residue theory.

¹This threshold can be determined using, e.g., machine learning approaches such as [5], where mmWave measurements are inferred from sub-6GHz ones.

²The extension of the model to the vehicular case is left for future works.

II. SYSTEM MODEL

Consider an outdoor 2 GHz orthogonal frequency division multiple access (OFDMA)-based 5G [8] N -tiers UDN, where each cell class n ($n = 1, \dots, N$) is modeled as a homogeneous Poisson point process (PPP) Φ_n , and distinguished by its deployment density λ_n , maximum transmit power per resource element (RE) P_n , antennas height h_n and beamwidth θ_n . The corresponding channel is presenting a large scale fading, with constant path-loss exponent ν and lognormal shadowing \mathcal{X}_n of mean μ_n and standard deviation σ_n . Assuming that UE locations follow an independent PPP Φ_u of density λ_u , the downlink analysis is performed at a typical UE located at the origin [9]. To model a massive IoT and CPE device ecosystem, we further suppose that UE locations are initially fixed, but may change every now and then.

A. Cell Monitoring Criteria

As we are dealing with an outdoor context, we suppose that all tier's cells are open access (including femtocells). We also adopt a reference signal receive power (RSRP)-based cell selection, wherein each UE periodically monitors the collection of the M strongest cells, dubbed here *monitoring set*, and ends up connecting to the best server. Since UE measurements rely on the long-term frequency-domain post-equalization receive power, small-scale fading variations do not impact cell selection/reselection and are not, therefore, reflected in the actual RSRP that reads

$$P_{x_n} = P_n \mathcal{X}_n \|x_n\|^{-\nu}, \quad (1)$$

where $\|x_n\|^{-\nu}$ stands for the standard path-loss between a typical UE and an n^{th} -tier BS located at $x_n \in \Phi_n$.

B. K -Factor Model

The K -factor—like all large scale parameters (LSPs)—follows a lognormal distribution with mean and variance depending on the frequency band, environment and transmission/reception schemes (cf. [10], [11] and references therein). Without loss of generality, let us adopt the findings of [6] for instance, where we assume that the narrowband K -factor periodically measured by a UE at independent positions can be empirically modeled for the n^{th} -tier as,

$$K_{x_n} = K_n \gamma_n \|x_n\|^{-\alpha}, \quad (2)$$

where $\alpha > 0$, K_n is the K -factor intercept defined as³

$$K_n = (h_n/h_0)^{\kappa_1} (\theta_n/\theta_0)^{\kappa_2} K_0, \quad (3)$$

³According to [6], this model involves also a seasonal factor F_s that reflects the vegetation. For the sake of simplicity and without loss of generality, we consider the Summer's dense vegetation case $F_s = 1$.

with $\kappa_1 > 0$, $\kappa_2 < 0$, $K_0 > 0$, and γ_n is an independent lognormal variable, whose decibel value is zero mean with a standard deviation σ_K . Accurate values of these model parameters can be obtained through a calibration process according to the target environment. New Jersey's measurement campaign in [6], for instance, yields $h_0 = 3$ m, $\theta_0 = 17^\circ$, $\alpha = 0.5$, $\kappa_1 = 0.46$, $\kappa_2 = -0.62$, $K_0 = 10$, and $\sigma_K = 8$ dB.

C. Equivalent Formulation

Since manipulating distances in PPPs is easier, let us transform the RSRP process (1) into a simple unit-power PPP $\tilde{\Phi}_n$, where the strongest power would correspond to the nearest neighbor cell to the typical UE. By invoking the random displacement theorem [9, 1.3.9], [12, Corollary 3] shows that the two-dimensional (2D) process (1) is equivalent to another 2D process $P_{y_n} = \|y_n\|^{-\nu}$, such that $y_n \in \tilde{\Phi}_n$ with density $\tilde{\lambda}_n = \lambda_n \Omega_n$, where $\Omega_n = P_n^{2/\nu} \mathbf{E} \left[\mathcal{X}_n^{2/\nu} \right]$ and the finite lognormal fractional moment $\mathbf{E} \left[\mathcal{X}_n^{2/\nu} \right] = \exp \left[\frac{\ln 10}{5} \frac{\mu_n}{\nu} + \frac{1}{2} \left(\frac{\ln 10}{5} \frac{\sigma_n}{\nu} \right)^2 \right]$. By means of the mapping theorem [9, 1.3.11], the K -factor can also be re-expressed as

$$K_{y_n} = K_n \gamma_n \Omega_n^{-\alpha/2} \|y_n\|^{-\alpha}, y_n \in \tilde{\Phi}_n. \quad (4)$$

III. XLOS SERVICE PROBABILITY

XLOS service probability in the vicinity of a UE, P_{XLOS} , is defined as the probability that at least one cell in the monitoring set presents a K -factor higher than a threshold, say K_{th} , that can be fine-tuned depending on the target service, i.e.,

$$P_{\text{XLOS}}(K_{\text{th}}) \triangleq \Pr \left[\bigcup_{m=1}^M K_{y_{n_m}} > K_{\text{th}}, \mathbf{n} \in \mathcal{M} \right], \quad (5)$$

where $\mathbf{n} = (n_1, \dots, n_M)$ and $\mathcal{M} = \{1, \dots, N\}^M$. In the sequel, we derive a closed-form expression for the XLOS service probability and study its asymptotic behavior.

A. Closed-Form Analysis

Using the total probability theorem as well as the independence between γ_{n_m} , $m = 1, \dots, M$, the definition (5) can be rewritten as

$$\begin{aligned} P_{\text{XLOS}}(K_{\text{th}}) &= 1 - \Pr \left[\bigcap_{m=1}^M K_{y_{n_m}} \leq K_{\text{th}}, \mathbf{n} \in \mathcal{M} \right] \\ &= 1 - \sum_{\mathbf{n} \in \mathcal{M}} \Pr \left[y_{n_m} \in \tilde{\Phi}_{n_m}, m = 1, \dots, M \right] \\ &\quad \times \int_0^{z_{n_2}} \int_0^{z_{n_M}} \int_0^{+\infty} \prod_{m=1}^M \text{CDF}_{\gamma_{n_m}} \left(\frac{K_{\text{th}} \Omega_{n_m}^{\alpha/2} z_{n_m}}{K_{n_m}} \middle| z_{n_m} \right) f(z_{n_1}, \dots, z_{n_M}) dz_{n_1} \dots dz_{n_M}, \end{aligned} \quad (6)$$

where $z_{n_m} = \|y_{n_m}\|^\alpha$ and $f(\cdot)$ is the joint probability density function (PDF) whose variables verify $0 \leq z_{n_1} \leq z_{n_2} \leq \dots \leq z_{n_M}$. Moreover, the independence between the homogeneous PPPs $\tilde{\Phi}_{n_m}$ as well as the superposition theorem [9, 1.3.3] imply that the sampling probability $\Pr[y_{n_m} \in \tilde{\Phi}_{n_m}, m = 1, \dots, M] = \prod_{m=1}^M \rho_{n_m}$, where $\rho_{n_m} = \tilde{\lambda}_{n_m}/\lambda_T$ and $\lambda_T = \sum_{n=1}^N \tilde{\lambda}_n$. To further develop (6), let us introduce the following new theorem.

Theorem 1 (Unified Expression for the Product of Lognormal CDFs⁴). *Consider M independent lognormal random variables γ_m ($m = 1, \dots, M$), with mean μ_m (dB) and standard deviation σ_m (dB). A unified expression for the product of their individual CDFs—that is equal to their joint CDF $\text{CDF}_{\gamma_1, \dots, \gamma_M}(\gamma_{\text{th},1}, \dots, \gamma_{\text{th},M})$ —is given by*

$$\prod_{m=1}^M \text{CDF}_{\gamma_m}(\gamma_{\text{th},m}) = \frac{1}{\pi^{M/2}} \sum_{l=1}^L w_l \prod_{m=1}^M \text{H}_{1,1}^{0,1} \left[\begin{matrix} \gamma_{\text{th},m} \\ \omega_{l,m} \end{matrix} \middle| \begin{matrix} (1, 1) \\ (0, 1) \end{matrix} \right], \quad (7)$$

where $\omega_{l,m} = 10^{(\sqrt{2}\sigma_m u_{l,m} + \mu_m)/10}$ for $l \in \{1, \dots, L\}$, w_l and $(u_{l,1}, \dots, u_{l,M})$ are respectively the weight and the M abscissas of the L^{th} -order M -dimensional Gaussian weight Stroud monomial cubature [14], [15], with $\sum_{l=1}^L w_l = \pi^{M/2}$.

Proof: cf. Appendix A. ■

On the other hand, an explicit expression of the joint PDF $f(\cdot)$ can be obtained via the following corollary.

Corollary 1 (of Theorem [22, Appendix]). *In a multi-tier random network modeled in terms of N independent PPPs $\tilde{\Phi}_n$ ($n = 1, \dots, N$) with densities $\tilde{\lambda}_n$, let $z_m = r_m^\alpha$ ($m = 1, \dots, M$), such that r_m is the distance of the m^{th} neighbor with respect to a certain origin. The joint PDF of z_1, \dots, z_M unconditionally to $\{\tilde{\Phi}_n\}$ reads*

$$f(z_1, \dots, z_M) = \left(\frac{2\pi\lambda_T}{\alpha} \right)^M e^{-\pi\lambda_T z_M^{2/\alpha}} \prod_{m=1}^M z_m^{2/\alpha-1}, \quad (8)$$

where $\lambda_T = \sum_{n=1}^N \tilde{\lambda}_n$.

Proof: cf. Appendix B. ■

By making use of the aforementioned sampling probability as well as Theorem 1 and Corollary 1, the XLOS service probability (6) can be rewritten after some algebraic manipulations as,

$$P_{\text{XLOS}}(K_{\text{th}}) = 1 - \left(\frac{2\sqrt{\pi}}{\alpha} \right)^M \sum_{\mathbf{n} \in \mathcal{M}} \prod_{m=1}^M \tilde{\lambda}_{n_m} \sum_{l=1}^L w_l \times I_1, \quad (9)$$

⁴This theorem can be viewed as a generalization of the well-established Gauss-Hermite representations of the lognormal PDF and CDF (see e.g., [13]).

$$I_1 = \frac{\alpha}{2} (\pi \lambda_T)^{-M} \mathbf{H}_{M, M-1}^{0, M} \left[\begin{array}{c} 0, 1 : \dots : 0, 1 \\ \underbrace{1, 1 : \dots : 1, 1}_{M\text{-times}} \end{array} \left| \begin{array}{c} \frac{K_{\text{th}} \Lambda_{n_1}^{\alpha/2}}{\omega_{l,1} K_{n_1}} \\ \vdots \\ \frac{K_{\text{th}} \Lambda_{n_M}^{\alpha/2}}{\omega_{l,M} K_{n_M}} \end{array} \right. \begin{array}{c} (1 - \frac{2i}{\alpha}; \mathbb{1}_{1 \leq i}, \dots, \mathbb{1}_{M \leq i})_{1 \leq i \leq M-1}, \left(1 - M; \overbrace{\frac{\alpha}{2}, \dots, \frac{\alpha}{2}}^{M\text{-times}}\right) \\ (1, 1) \\ (0, 1) \end{array} \left| \dots \left| \begin{array}{c} (1, 1) \\ (0, 1) \end{array} \right. \right. \right] \quad (15)$$

where the multidimensional integral I_1 is expressed as

$$I_1 = \int_0^{z_{n_2}} \dots \int_0^{z_{n_M}} \int_0^{+\infty} \prod_{m=1}^M z_{n_m}^{2/\alpha-1} \mathbf{H}_{1,1}^{0,1} \left[\begin{array}{c} \frac{K_{\text{th}} \Omega_{n_m}^{\alpha/2} z_{n_m}}{\omega_{l,m} K_{n_m}} \end{array} \left| \begin{array}{c} (1, 1) \\ (0, 1) \end{array} \right. \right] \times e^{-\pi \lambda_T z_{n_M}^{2/\alpha}} dz_{n_1} \dots dz_{n_M}. \quad (10)$$

To derive a closed-form solution for (10), let us recall the representation of the involved Fox H-functions in terms of Mellin-Barnes integrals [7, Eq. (1.1.1)], i.e.,

$$\mathbf{H}_{1,1}^{0,1} \left[z \left| \begin{array}{c} (1, 1) \\ (0, 1) \end{array} \right. \right] = \frac{1}{2\pi j} \int_{\mathcal{C}_m} \phi(\zeta_m) z^{\zeta_m} d\zeta_m, \quad (11)$$

where $\phi(\zeta_m) = \Gamma(\zeta_m) / \Gamma(1 + \zeta_m)$, and contours \mathcal{C}_m ($m = 1, \dots, M$) are defined such that $\text{Re}(\zeta_m) > 0$; the highest pole on the left. Combining (11) with (10) and interchanging the order of the real and contour integrals⁵ yields

$$I_1 = \left(\frac{1}{2\pi j} \right)^M \int_{\mathcal{C}_1} \dots \int_{\mathcal{C}_M} \Psi(\zeta_1, \dots, \zeta_M) \times \prod_{m=1}^M \phi(\zeta_m) \left(\frac{K_{\text{th}} \Omega_{n_m}^{\alpha/2}}{\omega_{l,m} K_{n_m}} \right)^{\zeta_m} d\zeta_1 \dots d\zeta_M, \quad (12)$$

with the multivariate term Ψ given by

$$\Psi(\zeta_1, \dots, \zeta_M) = \int_0^{z_{n_2}} \dots \int_0^{z_{n_M}} \int_0^{+\infty} e^{-\pi \lambda_T z_{n_M}^{2/\alpha}} \times \prod_{m=1}^M z_{n_m}^{2/\alpha + \zeta_m - 1} dz_{n_1} \dots dz_{n_M}. \quad (13)$$

Given that $\alpha \in \mathbb{R}^{+*}$ and $\text{Re}(\zeta_m) > 0$, and using the identity $1/a = \Gamma(a) / \Gamma(1 + a)$, the iterated integrals with respect to $z_{n_1}, \dots, z_{n_{M-1}}$ in (13) can be successively resolved by induction. The resulting integral relating to z_{n_M} is then obtained using [16, Eq. (3.478.1)], which leads to

$$\Psi(\zeta_1, \dots, \zeta_M) = \frac{\alpha}{2} (\pi \lambda_T)^{-(M + \frac{\alpha}{2} \sum_{m=1}^M \zeta_m)} \times \Gamma \left(M + \frac{\alpha}{2} \sum_{m=1}^M \zeta_m \right) \prod_{i=1}^{M-1} \frac{\Gamma \left(\frac{2i}{\alpha} + \sum_{m=1}^M \mathbb{1}_{m \leq i} \zeta_m \right)}{\Gamma \left(1 + \frac{2i}{\alpha} + \sum_{m=1}^M \mathbb{1}_{m \leq i} \zeta_m \right)}. \quad (14)$$

By plugging (14) into (12), we recognize that integral I_1 can be re-expressed in terms of the multivariate Fox H-function [7, A.1] as given by (15) on top of this page, where parameter $\Lambda_{n_m} \triangleq \Omega_{n_m} / \pi \lambda_T$ is encompassing network density, power and shadowing effects. Finally, a closed-form expression for P_{XLOS} is deduced by substituting (15) in (9).

⁵Which is permissible given the absolute convergence of the involved integrals.

B. Asymptotic Behavior

As depicted in Table I, the two asymptotic regimes of the ratio $K_{\text{th}}\Lambda_{n_m}^{\alpha/2}/\omega_{l,m}K_{n_m}$ reflect many practical scenarios, wherefore it is interesting to establish the corresponding XLOS service probability expressions; denoted \bar{P}_{XLOS} in the sequel. Let \mathcal{H} stand for the multivariate Fox H-function in (15) where

$$\mathcal{H} = \left(\frac{1}{2\pi j}\right)^M \int_{\mathcal{C}_1} \dots \int_{\mathcal{C}_M} F(\zeta_1, \dots, \zeta_M) d\zeta_1 \dots d\zeta_M. \quad (16)$$

In view of the series representations of the monivariate Fox H-function [17, Theorem 1.2] (while noticing the inverted definition of the H-function therein), an asymptotic expression of (15) is obtained as follows.

Low ratio regime: Since the integrand F has no poles on the right of the M individual contours in (16), [17, Eq. (1.2.23)] implies that $\mathcal{H} \simeq 0$, and thereby $\bar{P}_{\text{XLOS}} = 1$.

High ratio regime: By applying [17, Eq. (1.2.22)] to the M individual contour integrals, an approximation of \mathcal{H} is given in terms of the residues of F as

$$\begin{aligned} \mathcal{H} &\simeq \mathbf{Res}[F, (0, \dots, 0)] + \mathbf{Res}\left[F, \left(-\frac{2}{\alpha}, 0, \dots, 0\right)\right] \\ &\simeq \lim_{\zeta_M \rightarrow 0} \dots \lim_{\zeta_1 \rightarrow 0} \prod_{m=1}^M \zeta_m F(\zeta_1, \dots, \zeta_M) + \lim_{\zeta_M \rightarrow 0} \dots \lim_{\zeta_2 \rightarrow 0} \lim_{\zeta_1 \rightarrow -\frac{2}{\alpha}} \left(\zeta_1 + \frac{2}{\alpha}\right) \prod_{m=2}^M \zeta_m F(\zeta_1, \dots, \zeta_M), \end{aligned} \quad (17)$$

which evaluates to

$$\mathcal{H} \simeq \left(\frac{\alpha}{2}\right)^{M-1} \left[1 - \left(\frac{K_{\text{th}}\Lambda_{n_1}^{\alpha/2}}{\omega_{l,1}K_{n_1}}\right)^{2/\alpha}\right]. \quad (18)$$

Finally, combining (9), (15) and (18), as well as recalling that $\sum_{l=1}^L \omega_l = \pi^{M/2}$, we obtain after some algebraic manipulations

$$\bar{P}_{\text{XLOS}} = \frac{1}{\pi^{M/2-1}} \sum_{\mathbf{n} \in \mathcal{M}} \frac{\lambda_T}{\Omega_{n_1}} \prod_{m=1}^M \rho_{n_m} \sum_{l=1}^L \omega_l \left(\frac{\omega_{l,1}K_{n_1}}{K_{\text{th}}}\right)^{2/\alpha}. \quad (19)$$

Table I
XLOS SERVICE PROBABILITY ASYMPTOTIC EXPRESSIONS

Case	Practical Scenarios	\bar{P}_{XLOS}
$\frac{K_{\text{th}}\Lambda_{n_m}^{\alpha/2}}{\omega_{l,m}K_{n_m}} \rightarrow 0$	<ul style="list-style-type: none"> • Poor LOS (i.e., low K_{th}), • Fair LOS quality in a UDN with high power and narrow-beam antennas (i.e., high λ_T and K_{n_m}). 	1
$\frac{K_{\text{th}}\Lambda_{n_m}^{\alpha/2}}{\omega_{l,m}K_{n_m}} \rightarrow +\infty$	<ul style="list-style-type: none"> • High quality LOS (i.e., high K_{th}), • Fair LOS quality in a low density HetNet with low power and large beamwidth antennas (i.e., low λ_T and K_{n_m}). 	Equation (19)

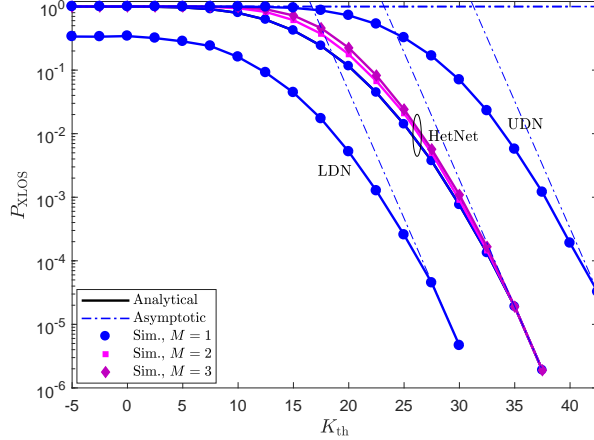


Figure 1. XLOS probability versus K_{th} for UDN, HetNet (Macro/Femto) and LDN. Path-loss exponent $\nu = 3$ and shadowing mean $\mu_n = 0$ for all tiers $n = 1, \dots, N$.

IV. NUMERICAL RESULTS AND MATHEMATICAL SOFTWARE

To validate our theoretical findings, we conduct Monte-Carlo simulations for three practical scenarios as depicted in Table II, and we adopt New Jersey’s calibration presented in II-B with $\sigma_K = 3$ dB. The analytical expressions are evaluated via a degree-11 Stroud cubature for which $L = (4M^5 - 20M^4 + 140M^3 - 130M^2 + 96M + 15)/15$. To that end, we make use of Stenger’s tabulations [18] to update the Matlab code in [19]. Moreover, using the quasi Monte-Carlo framework, we introduce in Appendix C an efficient GPU-oriented MATLAB routine to calculate the multivariate Fox H-function. By translating the Mellin-Barnes contour constraints into a linear optimization problem, we come up with a code automating the contour definition in Appendix D. A test example is finally provided in Appendix E. Note that we have already introduced a C/MEX version of the multivariate Fox H-function in our package [20]. An excerpt of the source and test examples are presented in Appendices F and G, respectively. Also, a Python implementation for the same generalized function can be found in [21].

Fig. 1 shows that, in the UDN case, the low ratio asymptotic regime is easily established, and good LOS conditions (e.g., $K_{\text{th}} = 12$ dB) are obtained with probability 1. Conversely, the low density network (LDN) scenario unfolds in NLOS situations with non-negligible probability (e.g., $K < -5$ dB with probability 0.5). By considering the neighboring cells ($M = 2, 3$) in the HetNet (Macro/Femto) case for instance, we remark that a substantial increase of the XLOS probability is achieved only in the non-asymptotic regime. Indeed, a high K_{th} requirement can be fulfilled merely by the serving cell, since the K -factors of neighbor cells become limited by the corresponding path-losses.

Table II
NETWORK AND TRANSMISSION SETTINGS

Case	N	λ_n	P_n (dBm)	θ_n (°)	h_n (m)	σ_n (dB)
UDN	1	3×10^{-2}	5.2	45	15	5
HetNet	2	$10^{-5}, 5 \times 10^{-4}$	15.2, -4.8	39, 180	25, 10	8, 4
LDN	1	3×10^{-8}	5.2	65	30	5

V. CONCLUSION

In this letter, we have introduced the XLOS service probability as a new K -factor-based performance indicator, and provided its analytical and asymptotic expressions that establish a link between 5G network and transmission parameters and the gradual visibility condition of radio links. As exemplified in the introduction, by tweaking a K -factor threshold K_{th} , the XLOS metric can be used by network optimization engineers as a switching probability between e.g., conventional localization schemes—requiring LOS—and Massive-MIMO ones operating in OLOS conditions. As a perspective, the adopted K -factor model from [6] can be extended to the vehicular case in future works.

APPENDIX A

PROOF OF THEOREM 1

First, by making a simple variable change, the product of lognormal PDFs p_{γ_m} , $p = \prod_{m=1}^M p_{\gamma_m}$, can be reformulated as

$$p = \frac{1}{\pi^{M/2}} \int_{\mathbb{R}^M} e^{-(u_1^2 + \dots + u_M^2)} Q(u_1, \dots, u_M) du_1 \dots du_M, \quad (20)$$

where $Q(u_1, \dots, u_M) = \prod_{m=1}^M \delta(\gamma_m - 10^{(\sqrt{2}\sigma_m u_m + \mu_m)/10})$. By applying the Gaussian-weight Stroud monomial cubature [14], [15] to (20), and recalling that $\delta(\gamma_m - a) = H_{0,0}^{0,0} \left[\frac{\gamma_m}{a} \middle| \begin{matrix} - \\ - \end{matrix} \right]$, we get

$$p = \frac{1}{\pi^{M/2}} \sum_{l=1}^L w_l \prod_{m=1}^M H_{0,0}^{0,0} \left[\frac{\gamma_m}{\omega_{l,m}} \middle| \begin{matrix} - \\ - \end{matrix} \right], \quad (21)$$

with w_l and $(u_{l,1}, \dots, u_{l,M})$ are respectively the l^{th} weight and abscissas of the M -dimensional cubature, and $\omega_{l,m} = 10^{(\sqrt{2}\sigma_m u_{l,m} + \mu_m)/10}$. Finally, by invoking [7, Eq. (2.53)], the integration of (21) with respect to γ_m from 0 to $\gamma_{\text{th},m}$ ($m = 1, \dots, M$) leads to (7). ■

APPENDIX B

PROOF OF COROLLARY 1

It immediately follows from applying the superposition theorem [9, 1.3.3] to the equivalent PPP $\Phi_T = \bigcup_{n=1}^N \tilde{\Phi}_n$, and performing a PDF transformation to the joint distance distribution of the first M neighbors given by theorem [22, Appendix]. ■

APPENDIX C

GPU-ENABLED MULTIVARIATE FOX H-FUNCTION MATLAB CODE

```

1 function out = mfoxh(z, Contour, an, Alphan, ap, Alphap, bq, Betaq, varargin)
2 % For dim >= 4, we recommend the use of GPU-enabled HPC servers
3 % an = [a1,...,an] and Alphan = [alpha,1,1 ... alpha,n,1;...; alpha,1,r...alpha,n,r]
4 % varargin form (i = 1 ..r): [ci,1 ...ci,n ; gammai,1 ... gammai,n],
5 %[ci,n+1 ...ci,p ; gammai,n+1 ... gammai,p],
6 %[di,1 ...di,m ; deltai,1 ... deltai,m], [di,m+1 ...di,q ; deltai,m+1 ... deltai,q]
7 % See notation in A. Mathai, The H-function, Theory and Applications, Annex A.1
8 %=====
9 dim = size(Contour,1);
10
11 N = 2^8 * 3^3 * 5^(2+dim); % For a better performance, N is made of powers of prime numbers
12 %===== Multivariate Quasi Monte-Carlo Integration =====
13 p = haltonset(dim,'Skip',1e3,'Leap',1e2);
14 p = scramble(p,'RR2');
15 in = gpuArray(net(p,N));
16 G = gpuArray(ones(1,N));
17 C = gpuArray(Contour);
18 points = kron(G,C(:,1)) + kron(G, C(:,2)-C(:,1)).* in';
19 mceval = Integrand(points);
20 mcsum = sum(mceval,2);
21 v = prod(C(:,2) - C(:,1)); % volume
22 out = v * mcsum / N; % Integral
23 %===== Integrand =====
24 function f = Integrand(s)
25 j = sqrt(-1);
26 r = length(z);
27 Nvar = length(varargin);
28 for nvar = 1 : Nvar
29     if isempty(cell2mat(varargin(nvar)))
30         varargin{nvar} = zeros(2,0);
31     end
32 end
33 Phi = 1;
34 for i = 1 : r
35     cni = gpuArray(cell2mat(varargin(4*(i-1)+1)));
36     cpi = gpuArray(cell2mat(varargin(4*(i-1)+2)));
37     dmi = gpuArray(cell2mat(varargin(4*(i-1)+3)));
38     dqi = gpuArray(cell2mat(varargin(4*(i-1)+4)));
39     Phi = Phi .* ((GammaProd(1-cni(1,:),cni(2,:), s(i,:)).* GammaProd(dmi(1,:),-dmi(2,:), s(i,:)))...
40         ./(GammaProd(cpi(1,:),-cpi(2,:), s(i,:)).* GammaProd(1-dqi(1,:),dqi(2,:), s(i,:)))).* z(i).^s(i,:);
41 end
42 Psi = GammaProd(1-an,Alphan,s)/(GammaProd(ap,-Alphap,s).* GammaProd(1-bq,Betaq,s));
43 f = (1/(2*j*pi)^r) * Phi .* Psi;

```

```

44 end
45 %===== GammaProd =====
46 function output = GammaProd(p,m,s)
47 if (isempty(p) || isempty(m))
48     output = ones(size(s(1,:)));
49 else
50 L1    = size(s,1);
51 comb = 0;
52 for i = 1 : L1
53 [pp ss] = meshgrid(p,s(i,:));
54 mm      = meshgrid(m(i,:),s(i,:));
55 comb    = comb + mm .* ss;
56 end
57     output = reshape(prod(gammas(pp + comb),2),size(s(1,:)));
58 end
59 end
60 end
61 % gammas function here is the complex gamma, available in
62 % www.mathworks.com/matlabcentral/fileexchange/3572-gamma
63 end

```

APPENDIX D

AUTOMATIC CONTOUR GENERATOR

```

1 function c = mfoxcontour(W, dim, an, Alphan, varargin)
2 % an = [a1,...,an] and Alphan = [alpha,1,1 ... alpha,n,1;...; alpha,1,r...alpha,n,r]
3 % varargin form (i = 1 ..r): [ci,1 ...ci,n ; gammai,1 ... gammai,n],
4 % [di,1 ...di,m ; deltai,1 ... deltai,m]
5 % See notation in A. Mathai, The H-function, Theory and Applications, Annex A.1
6 % W : control the width of the integration interval in [-i\infty +i\infty]
7 % dim : stands for the dimension
8
9 Nvar = length(varargin);
10 epsilon = 1/10;
11 f = ones(1,dim);
12 Q = -Alphan.';
13 b = 1-an-epsilon;
14 lb = [];
15 ub = [];
16
17 for i = 1 : Nvar/2
18     cni = cell2mat(varargin(2*(i-1)+1)); % [c1,i ...cn,i;gamma_1,i...gamma_n,i]
19     if(isempty(cni)) cni = [-1e10;1]; end
20     dmi = cell2mat(varargin(2*(i-1)+2)); % [d1,i ...dm,i;delta_1,i...delta_m,i]
21     if(isempty(dmi)) dmi = [1e10;1]; end

```

```

22 lb = [lb max((cni(1,:)-1)./cni(2,:))]+epsilon;
23 ub = [ub min(dmi(1,:)./dmi(2,:))]-epsilon;
24 end
25
26 options = optimoptions('linprog','Algorithm','interior-point');
27 out = linprog(f, Q, b, [], [], lb, ub, options);
28 c = [out.' - li * W; out.' + li * W];

```

APPENDIX E

MATLAB-GPU TEST CODE

In this example, we evaluate both a trivariate and bivariate Fox H-functions, respectively given by,

$$H_1 = H_{2,1:1,0:1,0:1,0}^{0,1:0,1:0,1:0,1} \left(3, 2, 0.5 \left| \begin{array}{c} (1.5; 1, 1, 1), (2; 1, 1, 1) \\ (2; 1, 1, 1) \end{array} \right| \begin{array}{c} - \\ (0, 1) \end{array} \left| \begin{array}{c} - \\ (3, 1) \end{array} \left| \begin{array}{c} - \\ (1, 1) \end{array} \right. \right), \quad (22)$$

and

$$H_2 = H_{2,1:1,0:1,0}^{0,1:0,1:0,1} \left(3, 2 \left| \begin{array}{c} (1.5; 1, 1), (2; 1, 1) \\ (2; 1, 1) \end{array} \right| \begin{array}{c} - \\ (0, 1) \end{array} \left| \begin{array}{c} - \\ (3, 1) \end{array} \right. \right). \quad (23)$$

The user may set the contour manually or generate it via `mfoxcontour` routine provided in Appendix D.

```

1 % Trivariate example
2 z = [3 2 0.5];
3 an = [1.5];
4 ap = [2];
5 Alphan = [1 ; 1 ; 1];
6 Alphap = [1 ; 1 ; 1];
7 bq = [2];
8 Betaq = [1 ; 1 ; 1];
9 Contour = mfoxcontour(10, 3, an, Alphan, [], [0;1], [], [3;1], [], [1;1]);
10 Contour =
11 -3.1000 -10.0000i -3.1000 +10.0000i
12 2.8000 -10.0000i 2.8000 +10.0000i
13 0.9000 -10.0000i 0.9000 +10.0000i
14 H1 = mfoxh(z, Contour, an, Alphan, ap, Alphap, bq, Betaq, [], [], [0;1], [], [], [3;1], [], [], [1;1], [])
15 H1 =
16 0.4886 + 0.0035i
17
18 % Bivariate example
19 z = [3 2];
20 an = [1.5];
21 ap = [2];
22 Alphan = [1 ; 1];
23 Alphap = [1 ; 1];

```

```

24 bq = [2];
25 Betaq = [1 ; 1];
26 Contour = [-1.5-10i -1.5+10i ; 2.5-10i 2.5+10i]; % contour set manually
27 H2 = mfoxh(z, Contour, an, Alphan, ap, Alphap, bq, Betaq, [], [], [0;1], [], [], [], [3;1], [])
28 H2 =
29 -0.6014 + 0.0011i

```

APPENDIX F

PARALLEL-CPU C/MEX CODE: EXCERPT FROM OUR PACKAGE [20]

```

1
2 #include <omp.h>
3 #include <stdio.h>
4 #include <math.h>
5 #include <gsl/gsl_qrng.h>
6 #include <gsl/gsl_complex.h>
7 #include <gsl/gsl_complex_math.h>
8 #include <gsl/gsl_sf_gamma.h>
9 #include <gsl/gsl_blas.h>
10 #include <gsl/gsl_vector_complex_double.h>
11 #include <gsl/gsl_matrix_complex_double.h>
12 #include <gsl/gsl_matrix_int.h>
13 #include "mfox.h"
14 #include "mex.h"
15
16 void mexFunction(int nlhs, mxArray *plhs[],
17                 int nrhs, const mxArray *prhs[]){
18
19     size_t i, j, k, mx, nx, dim;
20     double *ind, *xr, *xi, *zr, *zi, *max_call, *tol;
21     gsl_complex x;
22     gsl_vector_complex *xl, *xu;
23     gsl_matrix_int *index;
24     gsl_matrix_complex *Arg[20], *Emp;
25
26     if((nrhs -7) % 2 == 0){
27         dim = (size_t)((nrhs -7)/2);}
28     else{mexErrMsgTxt("Number of inputs is incorrect\n");}
29
30     // Retrieve vector [0, n, m1, n1, ..., mM, nM]
31     if(mxIsComplex(prhs[0]){mexErrMsgTxt("Indices should be integers\n");}
32     nx = mxGetN(prhs[0]);
33     if(nx != 2*dim+2){mexErrMsgTxt("Missing input(s) or extra elements in the first argument (index)\n");}
34     index = gsl_matrix_int_alloc(1, nx);

```

```

35 ind    = mxGetPr(prhs[0]);
36 for(j = 0; j < nx; j++){
37 gsl_matrix_int_set(index, 0, (const size_t)j, (int)ind[j]);
38 }
39 // Retrieve matrices Ai and Bi
40
41 for(k = 0; k < nrhs-4; k++){
42 if(! mxIsEmpty(prhs[k+1]))
43 {
44     /* Get the length of each input vector. */
45     mx = mxGetM(prhs[k+1]);
46     nx = mxGetN(prhs[k+1]);
47     /* Check input parameters size */
48
49 if(k == 0 && nx != dim){mexErrMsgTxt("Missing Fox H argument(s) z\n");}
50 if(k > 0 && nx ==1){
51     mexPrintf("Error in parameter # %d\n",k+2);
52     mexErrMsgTxt("Input size is incorrect\n");
53     return;
54 }
55 if(mxIsComplex(prhs[k+1])){
56     /* Get pointers to real and imaginary parts of the inputs. */
57     xr = mxGetPr(prhs[k+1]);
58     xi = mxGetPi(prhs[k+1]);
59
60     Arg[k] = gsl_matrix_complex_alloc(mx, nx);
61
62 for(i = 0; i < mx; i++)
63 {
64     for(j = 0; j < nx; j++)
65     {
66         GSL_SET_COMPLEX(x,xr[i + mx*j],xi[i + mx*j]);
67         gsl_matrix_complex_set(Arg[k],
68             (const size_t)i, (const size_t)j, x);
69     }
70 }
71 }//end of if
72
73 else{
74 /* Get pointers to real part */
75     xr = mxGetPr(prhs[k+1]);
76     Arg[k] = gsl_matrix_complex_alloc(mx, nx);
77
78 for(i = 0; i < mx; i++)
79 {
80     for(j = 0; j < nx; j++)

```

```

81     {
82         GSL_SET_COMPLEX(x,xr[i + mx*j],0.0);
83         gsl_matrix_complex_set(Arg[k],
84             (const size_t)i, (const size_t)j, x);
85     }
86 }
87 //end of else
88 }
89 else{
90     Arg[k] = gsl_matrix_complex_alloc(2,1);
91     gsl_matrix_complex_set_all(Arg[k],GSL_COMPLEX_ONE);
92 }
93 // end of for(k=0...
94 // Retrieve integration intervals
95 if(mxGetN(prhs[nrhs-3]) < dim || mxGetM(prhs[nrhs-3]) < 2){
96     mexErrMsgTxt("Contour matrix size incorrect");
97 }
98 xl = gsl_vector_complex_alloc(dim);
99 xu = gsl_vector_complex_alloc(dim);
100 /* Get pointers to real and imaginary parts of the inputs. */
101 xr = mxGetPr(prhs[nrhs-3]);
102 xi = mxGetPi(prhs[nrhs-3]);
103 // Initialize integration domains
104 for (i = 0; i < dim; i++) {
105     gsl_vector_complex_set (xl, i, gsl_complex_rect(xr[2*i],xi[2*i]));
106     gsl_vector_complex_set (xu, i, gsl_complex_rect(xr[2*i+1],xi[2*i+1]));
107 }
108 // Retrive max_call and tolerance
109 if(!mxIsComplex(prhs[nrhs-2]) && !mxIsComplex(prhs[nrhs-1])){
110     max_call = mxGetPr(prhs[nrhs-2]);
111     tol      = mxGetPr(prhs[nrhs-1]);
112 }
113 else{mexErrMsgTxt("MaxFunEval and AbsTol must be real");}
114
115 // Perform integration
116 gsl_complex result = GSL_COMPLEX_ZERO, error= GSL_COMPLEX_ZERO;
117 gsl_qrng* qrng = gsl_qrng_alloc(gsl_qrng_reversehalton, dim);
118 /* Available sequences: gsl_qrng_halton , gsl_qrng_sobol , gsl_qrng_niederreiter_2 , ↵
119     gsl_qrng_reversehalton */
120 quasi_monte_state* s = quasi_monte_alloc(dim);
121 quasi_monte_integrate(xl, xu, dim, max_call[0], 0, tol[0], qrng, s, &result, &error, index, Arg);
122 quasi_monte_free(s);
123 gsl_qrng_free(qrng);
124 /* Create a new complex array and set the output pointer to it. */
125 plhs[0] = mxCreateDoubleMatrix(1, 1, mxCOMPLEX);
126 zr = mxGetPr(plhs[0]);

```

```

126 zi = mxGetPi(plhs[0]);
127 zr[0] = GSL_REAL(result);
128 zi[0] = GSL_IMAG(result);
129 }

```

APPENDIX G

MEX TEST CODE: EXCERPT FROM OUR PACKAGE [20]

```

1 %***** 2-Dimensions *****
2 index = [0 1 1 1 1 1]; % [0 m m1 n1 m2 n2 ...mM nM]
3 z = [1 2]; % [z1...zM]
4 A = [1.5 1.0 1.0; 2.0 1.0 1.0]; % [a1 alpha_1,1...alpha_1,M; ...; ap alpha_p,1...alpha_p,M]
5 B = [2.0 1.0 1.0]; % [b1 beta_1,1...beta_1,M;...; bq beta_q,1 ...beta_q,M]
6 A1 = [-1 1]; % [a1_1 alpha1_1;...;a1_p1 alpha1_p1]
7 B1 = [0 1]; % [b1_1 beta1_1;...;b1_q1 beta1_q1]
8 A2 = [-1 1];
9 B2 = [3 1];
10 %c = [-0.5-10i 1.5-10i;-0.5+10i 1.5+10i];
11 c = mfoxcontour(10, 2, index, A, A1, B1, A2, B2);
12 % Tolerance settings
13 MaxFunEval = 2*1e5; % increase it to get more precision (especially for more than 2 dimensions)
14 AbsTol = 1e-4;
15 tic;
16 mfoxh(index, z, A, B, A1, B1, A2, B2, c, MaxFunEval, AbsTol)
17 toc;
18 %***** 3-Dimensions *****
19 index = [0 1 1 1 1 1 1];
20 z = [1 2 0.5];
21 A = [1.5 1.0 1.0 1; 2.0 1.0 1.0 1];
22 B = [2.0 1.0 1.0 1];
23 A1 = [-1 1];
24 B1 = [0 1];
25 A2 = [-1 1];
26 B2 = [3 1];
27 A3 = [0 1];
28 B3 = [1 1];
29 %c = [-0.5-10i 1.5-10i 0.5-10i;-0.5+10i 1.5+10i 0.5+10i];
30 c = mfoxcontour(10, 3, index, A, A1, B1, A2, B2, A3, B3)
31 % Tolerance settings
32 MaxFunEval = 2*1e6; % increase it to get more precision (especially for more than 2 dimensions)
33 AbsTol = 1e-4;
34 tic;
35 mfoxh(index, z, A, B, A1, B1, A2, B2, A3, B3, c, MaxFunEval, AbsTol)
36 toc;
37 %

```


REFERENCES

- [1] H. Chergui, M. Benjillali and M.-S. Alouini, "Rician K -factor-based analysis of XLOS service probability in 5G outdoor ultra-dense networks," Submitted as a Letter.
- [2] S. Sun *et al.*, "MIMO for millimeter wave wireless communications: Beamforming, spatial multiplexing, or both?," *IEEE Comm. Mag.*, vol. 52, no. 12, pp. 110-121, Dec. 2014.
- [3] N. Garcia *et al.*, "Direct localization for massive MIMO," *IEEE Trans. Sig. Proc.*, vol. 65, no. 10, pp 2475-2487, May 2017.
- [4] 3GPP, "Cellular system support for ultra-low complexity and low throughput Internet of Things (CIoT) (Release 13)," *TR 45.820*, Nov. 2015.
- [5] F. B. Mismar and B. L. Evans, "Machine learning approach to estimating mmWave signal measurements during handover," 2017. [Online]. Available: arxiv.org/abs/1710.01879.
- [6] L. J. Greenstein *et al.*, "Ricean K -factors in narrow-band fixed wireless channels: Theory, experiments, and statistical models," *IEEE Trans. Veh. Tech.*, vol. 58, no. 8, pp. 4000-4012, Oct. 2009.
- [7] A. M. Mathai, R. K. Saxena, and H. J. Haubold, *The H-Function: Theory and Applications*, Springer, New York, 2010.
- [8] 3GPP, "NR: Physical channels and modulation (Release 15)," *TS 38.211*, Sep. 2017.
- [9] F. Baccelli and B. Blaszczyszyn, *Stochastic Geometry and Wireless Networks, Volume I—Theory*. NOW: Foundations and Trends in Networking, 2009.
- [10] 3GPP, "Study on channel model for frequencies from 0.5 to 100 GHz (Release 14)," *TR 38.901*, May. 2017.
- [11] H. Tataria *et al.*, "Impact of Line-of-Sight and Unequal Spatial Correlation on Uplink MU-MIMO Systems," *IEEE Wireless Commun. Lett.*, vol. 6, no. 5, pp. 634-637, Oct. 2017.
- [12] P. Madhusudhanan *et al.*, "Downlink performance analysis for a generalized shotgun cellular systems," 2012. [Online]. Available: arxiv.org/abs/1002.3943.
- [13] F. Yilmaz and M.-S. Alouini, "A novel unified expression for the capacity and bit error probability of wireless communication systems over generalized fading channels," *IEEE Trans. Commun.*, vol. 60, no. 7, pp. 1862-1876, July 2012.
- [14] A. H. Stroud, *Approximate Calculation of Multiple Integrals*, Englewood Cliffs, N.J.:Prentice-Hall, 1971.
- [15] R. Cools and P. Rabinowitz, "Monomial cubature rules since "Stroud": a compilation," *Journal of Computational and Applied Mathematics*, vol. 48, no. 3, pp. 309-326, Nov. 1993.
- [16] I. S. Gradshteyn and I. M. Ryzhik, *Table of Integrals, Series, and Products*, 7th ed., Academic Press, 2007.
- [17] A. Kilbas and M. Saigo, *H-Transforms: Theory and Applications. Analytical Methods and Special Functions*, Chapman & Hall/CRC, Boca Raton-London-New York-Washington, D.C., 2004.
- [18] F. Stenger, "Tabulation of certain fully symmetric numerical integration formulas of degree 7, 9 and 11," *Mathematics of Computation*, vol. 25, no. 116, pp. 935 and S58-S125, Oct. 1971.
- [19] J. Burkardt, "Stroud numerical integration in M dimensions," 2010. [Online]. Available: people.sc.fsu.edu/~jburkardt/m_src/stroud/stroud.html.
- [20] H. Chergui, M. Benjillali and M.-S. Alouini, "Multivariate Fox H-Function C/MEX Package: mfoxh," Zenodo, April, 2018. DOI: 10.5281/zenodo.1217925.
- [21] H. R. Alhennawi *et al.*, "Closed-form exact and asymptotic expressions for the symbol error rate and capacity of the H -function fading channel," *IEEE Trans. on Veh. Tech.*, vol. 65, no. 4, pp. 1957-1974, Apr. 2016.
- [22] H. R. Thompson, "Distribution of distance to n th neighbour in a population of randomly distributed individuals," *Ecology*, vol. 37, no. 2, pp. 391-394, Apr. 1956.