Correlated Shadow Fading for Cellular Network System-Level Simulations with Wrap-Around

Ming Ding, NICTA, Australia {Ming.Ding@nicta.com.au}
Meng Zhang, Shanghai Jiao Tong University, China {mengzhang@sjtu.edu.cn}
David López-Pérez, Bell Labs Alcatel-Lucent, Ireland {dr.david.lopez@ieee.org}
Holger Claussen, Bell Labs Alcatel-Lucent, Ireland {holger.claussen@alcatel-lucent.com}

***Abstract*—In system-level simulations, it is vital to appropriately model spatially correlated shadow fading, especially the auto-correlation, so as to emulate realistic scenarios. However, as the size of the network increases, the complexity of the straightforward approach involving the Cholesky decomposition grows exponentially. Alternatively, low-complexity oriented schemes to generate correlated shadow fading value (SFV) maps for large networks are more preferable. However, the existing schemes do not consider wrap-around, which should be taken into account in the generation of SFV maps to facilitate user equipment (UE) mobility simulation and avoid the reduced interference in the outer-rim region. In order to prevent discontinuous shadow fading across the border of the simulated area and the wrap-around area, new methods to efficiently generate SFV maps are required. In this paper, a novel scheme to generate SFV maps with wrap-around is proposed, and its significant accuracy is analysed using computer simulations.**

I. Introduction

Channel fading is one of the most fundamental characteristics in wireless communications. Basically, channel fading can be classified into two categories: multi-path fading and shadow fading. Multi-path fading is caused by the reflectors and scatters around transceivers, which disperse signals into different paths with distinct attenuation, delay, Doppler shift and phase rotation that may add up constructively or destructively at the receiver side. Shadow fading, on the other hand, is caused by the blocking of obstacles, e.g., buildings or hills. As a result, user equipments (UEs) in close proximity often have strongly correlated shadow fading values (SFVs) associated with the same base station (BS), which is usually referred to as auto-correlation. To capture the auto-correlation property, an exponential correlation model was introduced in [1], in which the normalized correlation function was expressed as an exponentially decreasing function with respect to the increasing distance. Note that multiple links to a given UE from different BSs, which are close to each other, may also observe a highly cross-correlated shadow fading, which is usually referred to as cross-correlation. Cross-correlation can be easily introduced by a weighted sum of a UE-specific SFV and a BS-specific SFV with the same auto-correlation [2].

In system-level simulations, it is vital to appropriately model spatially correlated shadow fading, especially the auto-correlation, so as to emulate realistic scenarios [3]. Theoretically, once the auto-correlation function of shadow fading is known, we can generate an SFV grid map through computations involving the Cholesky decomposition [4]. Then, for a UE located inside the grid, its SFV can be computed through standard interpolation from the SFVs of the nearby grid points. However, as the size of the network increases, the complexity of such approach grows exponentially. Alternatively, low-complexity oriented schemes to generate a spatially auto-correlated SFV map for large networks have been proposed in recent work such as [4], [5], etc.

Wrap-around should also be considered in the generation of SFV maps for system-level simulations. To be more specific, when using wrap-around, a few additional tiers of cells are defined outside the simulated area according to deterministic patterns, and such additional tiers of cells are mapped from the cells inside the simulated area. The purpose of introducing a few additional tiers of cells is to facilitate UE mobility simulation and avoid the reduced interference in the outer-rim region [6]. However, in order to prevent discontinuous shadow fading across the border of the simulated area and the wrap-around area, new methods to generate wrap-around SFV maps are required. To the best of our knowledge, this work is the first one to address the correlated shadow fading for system-level simulations with wrap-around. Note that the said wrap-around SFV map is BS specific. Therefore, it is straightforward to generate multiple SFV maps for multiple BSs and further introduce the cross-correlation [2]. Thus, we concentrate on the auto-correlation in this paper.

The remainder of this paper is structured as follows. The system model including the wrap-around structure is described in Section II. The proposed scheme for the generation of wrap-around SFV maps is presented in Section III, and is analysed using system-level simulations in Section IV. Finally, concluding remarks are drawn in Section V.

II. System Model

Let us consider a typical cellular network with M cells deployed in a regular hexagonal lattice. Fig. 1 shows the considered cellular network with $M = 7$ (one centre cell plus one tier of interfering cells). Our objective is to generate a spatially auto-correlated SFV map for this cellular network with wrap-around. Note that our proposed scheme can be easily extended to larger networks, e.g., with $M = 19$ (one centre cell plus two tiers of cells). Also note that cell sectorisation by means of directional antennas [3] has no impact on the SFV maps.

The radius of each hexagonal cell is denoted as R_0. The contour of the considered 7-cell model is non-convex and it

978-1-4673-6432-4/15/$31.00 ©2015 IEEE

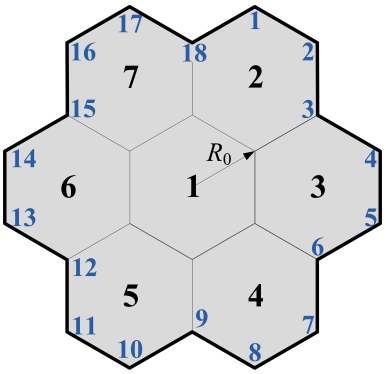

Fig. 1. Illustration of the considered cellular model ($M = 7$).

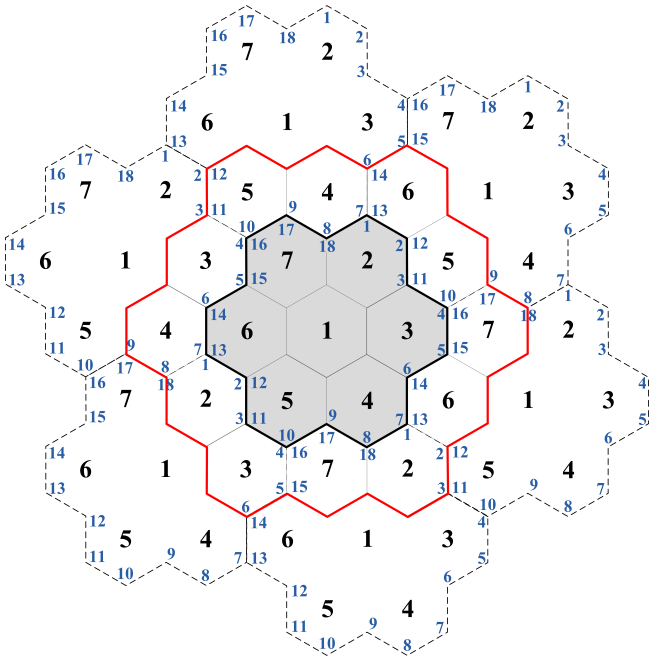

Fig. 2. Illustration of wrap-around ($M = 7$).

contains $K = 18$ vertexes/edges, which are shown in Fig. 1 and highlighted with a black outline. For convenience, the vertexes of the contour are labelled sequentially as $1, 2, \ldots, K$. Note that it is easy to verify that $K = 30$ when $M = 19$.

For convenience, we define the simulation area shown in Fig. 1 as the original area (OA). In Fig. 2, apart from the OA, we also plot the wrap-around area (WA), which is the area between the OA and the red outline. The WA shown in Fig. 2 wraps the OA with a second tier of cells, and the WA cells are mapped from the OA cells according to a deterministic pattern. Such wrap-around pattern is obtained from 6 shifted OA duplicates wrapping the OA, as shown in Fig. 2 by the dash outlines. Note that the centre cell in the OA, i.e., cell 1, does not appear in the WA because it has already been wrapped by the first tier of cells in the OA. Also note that for $M = 19$, the WA is comprised of two additional tiers of cells since there are two tiers of cells in the OA [6].

With wrap-around, the outer-rim cells, i.e., cells 2∼7 are connected with the WA cells that are associated with the cells in the OA. Note that the WA cells should not be interpreted as replicas of the OA cells since the wrap-around layout should be only understood in a toroidal space, and hence the purpose of the WA cells is to illustrate in a two-dimensional space that the OA cells are actually connected with each other in the toroidal space [6]. For example, in the OA depicted in Fig. 1, cell 2 only has three neighbour cells, i.e., cells 7, 1 and 3; while with wrap-around, cells 4, 6 and 5 are also its neighbour cells (see Fig. 2). This way, when a UE moves out of the OA to a WA cell, instead of getting lost, it will reappear in another OA cell corresponding with the respective WA cell. Moreover, interference from outside the OA can be explicitly modeled based on the WA cells so that the outer-rim cells will not experience an unrealistic signal-to-interference-plus-noise ratio (SINR) increase caused by the reduced interference due to the absence of some neighbour cells.

To define and realize the wrap-around structure, the vertexes and edges shown in Fig. 1 should not be constructed independently. This is because some vertexes and edges are the same as (duplicates of) others, as shown in Fig. 2. Indeed, based on Fig. 2, two tables are presented in the following, which are respectively the duplicated vertex table and the duplicated edge table resulting from wrap-around.

Table I
DUPLICATED VERTEXES

Dup. vertex group #	Vertex #1	Vertex #2	Vertex #3
1	1	7	13
2	2	12	/
3	3	11	/
4	4	10	16
5	5	15	/
6	6	14	/
7	8	18	/
8	9	17	/

Table II
DUPLICATED EDGES

Dup. edge group #	Edge #1	Edge #2
1	$Edge_{(1)(2)}$	$Edge_{(13)(12)}$
2	$Edge_{(2)(3)}$	$Edge_{(12)(11)}$
3	$Edge_{(3)(4)}$	$Edge_{(11)(10)}$
4	$Edge_{(4)(5)}$	$Edge_{(16)(15)}$
5	$Edge_{(5)(6)}$	$Edge_{(15)(14)}$
6	$Edge_{(6)(7)}$	$Edge_{(14)(13)}$
7	$Edge_{(7)(8)}$	$Edge_{(1)(18)}$
8	$Edge_{(8)(9)}$	$Edge_{(18)(17)}$
9	$Edge_{(9)(10)}$	$Edge_{(17)(16)}$

In Table I, duplicated vertexes are arranged into 8 groups. In Table II, $Edge_{(k1)(k2)}$ denotes the edge connecting vertex k_1 and vertex k_2, and duplicated edges are gathered into 9 groups. We have two remarks regarding the number of vertexes/edges in each duplicated vertexes/edge group as follows.

Remark 1. *The number of the vertexes in each duplicated vertex group is at most 3.*

This is because the internal angle associated with each vertex is either 120 degrees (for convex vertexes) or 240 degrees (for concave vertexes) due to the hexagonal lattice,

and thus at most 3 vertexes can be pinned together at the same point since the sum of three 120-degree internal angles reaches 360 degrees. Also note that no vertex can belong to two or more duplicated vertex groups. Otherwise, the rebirth place of any UE moving across any vertex would not be unique, which is not allowed in wrap-around.

Remark 2. *The number of the edges in each duplicated edge group is always 2.*

This is because each pair of duplicated edges serves as the border for two hexagonal cells, and the two edges in such edge pair satisfy an one-to-one mapping due to the fact that the two cells adjacent to any cell border are pre-defined according to the wrap-around pattern. Violation of this principle would imply that a certain cell could share the same cell border with two or more different cells, which is impossible in the considered two-dimensional cellular model shown in Fig. 2.

Although wrapping-around the cells in the hexagonal lattice solves the UE mobility problem and the reduced interference problem, it poses a new issue: the potential discontinuous shadow fading across any border between the OA and the WA, if SFVs are generated independently for each cell edge. This problem can be visualized in Fig. 2, where some geographically non-adjacent cells in the OA may share a duplicated edge in the wrap-around structure, e.g., $Edge_{(2)(3)}$ of cell 2 and $Edge_{(12)(11)}$ of cell 5. If the SFVs of these duplicated edges are generated independently, they could have very different shadow fading, giving rise to boundary issues due to discontinuity. It is thus reasonable to assume that the duplicated vertexes/edges shown in Table I/II should have the same SFVs. Otherwise, sudden SINR fluctuations will occur once a UE moves from an OA cell to a WA cell, or large SINR discrepancy will exist for UEs with close proximity to the border of the OA and the WA, which is unrealistic. As a result, in our example, $Edge_{(2)(3)}$ and $Edge_{(12)(11)}$ must have the same SFVs, which should be considered when generating SFV maps.

III. The Proposed Scheme for SFV Generation

In order to generate an auto-correlated shadow fading map for networks with wrap-around, we perform the SFV generation for the OA in three stages, i.e.,

1) Decomposition of the OA into string loops.
2) Generation of SFVs for the outmost string loop (the 1st string loop).
3) Generation of SFVs for the inner string loops (from the 2nd string loop to the last one).

The details of the proposed scheme are presented in the following.

A. Decomposition of the OA into string loops

First, we propose to decompose the OA into a number of string loops by continuously peeling off the contours of the OA. Such operation on the considered OA (shown in Fig. 1) is illustrated in Fig. 3 with $R_0 = 50$ m, where solid-line contours are string loops and circles denote the interested grid points, for which auto-correlated SFVs should be generated.

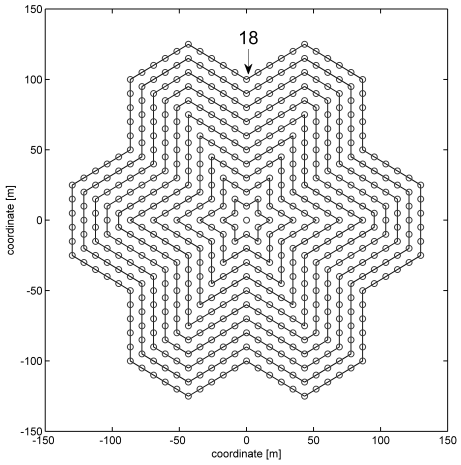

Fig. 3. Illustration of the decomposition of the OA into string loops ($M = 7$).

It can be seen that by properly choosing the distance between adjacent string loops, the OA can be decomposed into an integer number of string loops, denoted as L, with the last string loop degenerating to the centre point of the OA. To be more specific, denote the distance from vertex K to the centre point of the OA as y, then the distance between adjacent string loops should be set to

$$D = y/(L-1). \tag{1}$$

Note that y equals to $2R_0$ and $4R_0$ for $M = 7$ and $M = 19$, respectively. Consequently, the number of grid points on each edge becomes $Q_1 = \lfloor \frac{R_0}{D} \rfloor + 1$ for the outmost string loop.

B. Generation of SFVs for the outmost string loop

For the outmost string loop, we generate its SFVs edge by edge in a clockwise and sequential manner. To be more specific, firstly the SFVs for $Edge_{(1)(2)}$ will be generated, and then $Edge_{(2)(3)}$, $Edge_{(3)(4)}$ and so on will be sequentially treated. Finally, the SFVs for $Edge_{(K)(1)}$ will be generated. The reason why we propose the edge-by-edge approach, rather than using the point-by-point approach in [4], is due to the characteristics of wrap-around, explained in Tables I and II. In more detail, using the point-by-point approach, we would first generate the SFVs for vertex 1, and then for the grid points on $Edge_{(1)(2)}$. Subsequently, we would generate the SFVs for vertex 2, and then for the grid points on $Edge_{(2)(3)}$, and so on. However, using this approach there is no guarantee that the resulting SFVs would comply with the wrap-around requirement presented in Table I and II. For instance, when we generate the SFVs all the way from vertex 2 via $Edge_{(2)(3)}$,..., $Edge_{(11)(12)}$ to vertex 12, we cannot expect the SFVs of the points on $Edge_{(2)(3)}$ to be the same as those on $Edge_{(12)(11)}$, due to their independent generation. In order to circumvent this issue, we could generate the SFVs for the outmost string loop as a whole, considering a new auto-correlation function modified by wrap-around. But the computational complexity would be very high, particularly for large networks with, e.g., $M = 19$. In more detail, since the computational complexity of commonly used algorithms for the Cholesky decomposition

is $O(N^3)$ [7], the complexity of the proposed edge-by-edge approach in this paper is in the order of $O\left(KQ_1^3\right)$; while that of the SFV generation for the outmost string loop as a whole is in the order of $O\left(K^3Q_1^3\right)$, which is about 1000 times more than the proposed scheme when $M = 19$ (i.e., $K = 30$). Next, we will present our proposed edge-by-edge approach.

Suppose that $s_i^{\mathrm{E}_k}$ denotes the auto-correlated SFV for the i-th point on the k-th edge of the outmost string loop and $\mathbf{s}^{\mathrm{E}_k} = \left[s_1^{\mathrm{E}_k}, \ldots, s_i^{\mathrm{E}_k}, \ldots, s_{Q_1}^{\mathrm{E}_k}\right]^{\mathrm{T}}$ denotes the SFVs of all points including the two vertexes on the k-th edge of the outmost string loop. From [4], $\mathbf{s}^{\mathrm{E}_k}$ can be generated as

$$\mathbf{s}^{\mathrm{E}_k} = \left(\mathbf{R}^{\mathrm{E}_k}\right)^{\frac{1}{2}} \mathbf{a}^{\mathrm{E}_k}, \tag{2}$$

where

- $\mathbf{R}^{\mathrm{E}_k} = \begin{bmatrix} r_{11}^{\mathrm{E}_k} & \cdots & r_{1Q_1}^{\mathrm{E}_k} \\ \vdots & \ddots & \vdots \\ r_{Q_1 1}^{\mathrm{E}_k} & \cdots & r_{Q_1 Q_1}^{\mathrm{E}_k} \end{bmatrix}$, where $r_{ij}^{\mathrm{E}_k}$ is the correlation coefficient between the SFVs of the i-th point and the j-th point on the k-th edge of the outmost string loop and $r_{ij}^{\mathrm{E}_k}$ is calculated as [8]

$$r_{ij}^{\mathrm{E}_k} = \exp\left(-\frac{d_{ij}^{\mathrm{E}_k}}{d_{\mathrm{corr}}}\right), \quad d_{ij}^{\mathrm{E}_k} \geq 0, \tag{3}$$

  where $d_{ij}^{\mathrm{E}_k}$ is the distance between the i-th point and the j-th point on the k-th edge of the outmost string loop, and d_{corr} is the correlation length. In practice, d_{corr} is dependent on the environment. For example, in the Long Term Evolution (LTE) networks, d_{corr} is suggested to be 7$\sim$13 m for urban picocells [8]. Note that $\left(\mathbf{R}^{\mathrm{E}_k}\right)^{\frac{1}{2}}$ in (2) only needs to be calculated once for each k and it can be obtained by the Cholesky decomposition as

$$\left(\mathbf{R}^{\mathrm{E}_k}\right)^{\frac{1}{2}} = \mathrm{chol}\left(\mathbf{R}^{\mathrm{E}_k}\right), \tag{4}$$

  where $\mathrm{chol}(\cdot)$ denotes the Cholesky decomposition function and $\left(\mathbf{R}^{\mathrm{E}_k}\right)^{\frac{1}{2}}$ is a lower triangular matrix.
- $\mathbf{a}^{\mathrm{E}_k} = \left[s_{Q_1}^{\mathrm{E}_{k-1}}, a_2^{\mathrm{E}_k}, \ldots, a_i^{\mathrm{E}_k}, \ldots, a_{Q_1}^{\mathrm{E}_k}\right]^{\mathrm{T}}$, where $a_i^{\mathrm{E}_k}, i \in \{2, \ldots, Q_1\}$, denotes an independent and identically distributed (i.i.d.) random variable (RV) for the i-th point on the k-th edge of the outmost string loop. To be more specific, $a_i^{\mathrm{E}_k}$ follows a Gaussian distribution denoted as $\mathcal{N}\left(0, \sigma_{\mathrm{std}}^2\right)$, and its standard deviation σ_{std} depends on the scenario. For example, in LTE networks, σ_{std} is suggested to be 8 dB and 10 dB for macrocells and picocells, respectively [9]. Note that in the proposed scheme, the first entry of $\mathbf{a}^{\mathrm{E}_k}$ is fixed to $s_{Q_1}^{\mathrm{E}_{k-1}}$ so that the SFV of the start point of the k-th edge would be the same as that of the end point of the $(k-1)$-th edge. This is because that $\left(\mathbf{R}^{\mathrm{E}_k}\right)^{\frac{1}{2}}$ is a lower triangular matrix and its first diagonal entry is always 1 since $r_{11}^{\mathrm{E}_k} = 1$.

To meet the vertex duplication requirement defined in Table I, we propose to perform the following operations for the k-th edge. Suppose that the end point of the k-th edge belongs to the duplicated vertex group x in Table I, and in group x there exists a vertex v, whose SFV denoted as s_v, has already been generated, then we will continuously and randomly generate $\mathbf{a}^{\mathrm{E}_k}$ for the k-th edge until the difference of the SFV of the end point of the k-th edge and s_v is below a pre-defined tolerance parameter ϵ. This criterion can be formulated as

$$\left|s_v - s_{Q_1}^{\mathrm{E}_k}\right| < \epsilon, \tag{5}$$

where $s_{Q_1}^{\mathrm{E}_k}$ is obtained from $s_{Q_1}^{\mathrm{E}_k} = \left[\left(\mathbf{R}^{\mathrm{E}_k}\right)^{\frac{1}{2}}\right]_{Q_1,:} \mathbf{a}^{\mathrm{E}_k}$ and $[\mathbf{A}]_{i,:}$ denotes the i-th row of matrix $\mathbf{A}$.

Besides, the SFVs associated with an edge should be assigned to its duplicated edge shown in Table II due to wrap-around. The proposed algorithm for the generation of SFVs for the outmost string loop is formally presented in Algorithm 1.

Algorithm 1 Generation of SFVs for the outmost string loop

Initialization: Set the edge index $k = 1$ and randomly initialize $s_{Q_1}^{\mathrm{E}_0} \sim \mathcal{N}\left(0, \sigma_{\mathrm{std}}^2\right)$.

Repeat:

1) For the k-th edge ($k \leq K$), check Table II.
2) If the SFVs of its duplicate, denoted as the m-th edge, have already been generated, then assign $\mathbf{s}^{\mathrm{E}_m}$ to $\mathbf{s}^{\mathrm{E}_k}$ according to Table II, and go to Step 7;
3) Else, randomly generate $\mathbf{a}^{\mathrm{E}_k}$.
4) If (5) holds, then go to Step 6;
5) Else, go to Step 3.
6) Compute $\mathbf{s}^{\mathrm{E}_k}$ using (2).
7) $k = k + 1$ and go to Step 1.

Termination: Output $\left\{\mathbf{s}^{\mathrm{E}_1}, \ldots, \mathbf{s}^{\mathrm{E}_k}, \ldots, \mathbf{s}^{\mathrm{E}_K}\right\}$ as the results.

C. Generation of SFVs for the inner string loops

In this section, we address the SFV generation for the inner string loops, in a similar manner as proposed in [4], except by the fact that we propose to generate the SFVs for the inner string loops one by one, and for each string loop the generation is performed for its grid points one by one in a clockwise manner. To generate the SFV for each grid point, we take its N-nearest neighbour points into account. Illustrations of the considered pattern of N-nearest neighbour points with $N = 18$ and 36 are provided in Figs. 4 and 5, respectively, where red squares, blue squares and black triangles denote the interested grid point, the grid points with generated SFVs and the generation pattern of the N-nearest neighbour points.

It should be noted that not all of the N nearest neighbour points may have generated SFVs for a given interested grid point due to the complicated hexagonal network shapes, and thus only those grid points with generated SFVs are involved in the SFV generation for the interested grid point. For example, in Fig. 4, the SFV of the interested grid point on the 3rd sting loop is generated using the known SFVs of 10 neighbour points (shown as blue squares overlapping with black triangles), because the SFVs of the other N-nearest neighbour points are not available yet.

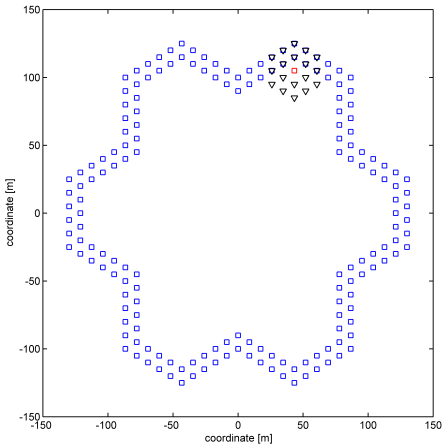

Fig. 4. Illustration of the N-nearest neighbour points ($N = 18$).

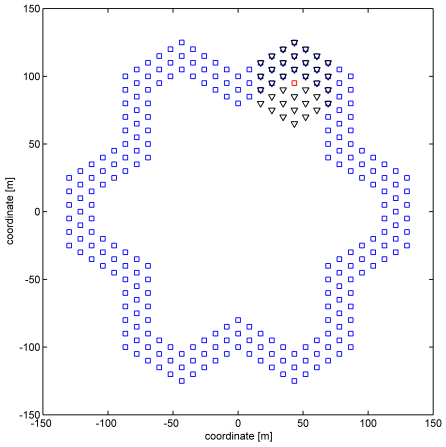

Fig. 5. Illustration of the N-nearest neighbour points ($N = 36$).

To formally describe the proposed algorithm, we have the following definitions.

- The u-th grid point on the l-th string loop is denoted as p_0^{lu}, the SFV of which is denoted as $s_u^{\mathrm{SL}_l}$ and $\mathbf{s}^{\mathrm{SL}_l} = \left[s_1^{\mathrm{SL}_l}, \ldots, s_u^{\mathrm{SL}_l}, \ldots, s_{Q_l}^{\mathrm{SL}_l}\right]^{\mathrm{T}}$, where Q_l is the number of grid points on the l-th string loop. Without loss of generality, the first grid point on the l-th string loop is defined as the one that is nearest to vertex 1.
- $\bar{N}$ denotes the number of the nearest neighbour points with generated SFVs for p_0^{lu}. Note that $\bar{N} \leq N$.
- The j-th nearest point with a generated SFV for p_0^{lu} is denoted as p_j^{lu}, $\left(j \in \left\{1, 2, \ldots, \bar{N}\right\}\right)$, the SFV of which is denoted as z_j^{lu} and $\mathbf{z}^{lu} = \left[z_1^{lu}, \ldots, z_j^{lu}, \ldots, z_{\bar{N}}^{lu}\right]^{\mathrm{T}}$.
- d_{ij}^{lu} denotes the distance between p_i^{lu} and p_j^{lu}, where $i, j \in \left\{0, 1, 2, \ldots, \bar{N}\right\}$.
- $\bar{\mathbf{R}}^{lu} = \begin{bmatrix} r_{11}^{lu} & \cdots & r_{1\bar{N}}^{lu} \\ \vdots & \ddots & \vdots \\ r_{\bar{N}1}^{lu} & \cdots & r_{\bar{N}\bar{N}}^{lu} \end{bmatrix}$, where r_{ij}^{lu} is calculated as (3) with the superscript E_k replaced by lu.
- $\bar{\mathbf{R}}^{lu}$ is extended as $\mathbf{R}^{lu} = \begin{bmatrix} & & & r_{10}^{lu} \\ & \bar{\mathbf{R}}^{lu} & & \vdots \\ & & & r_{\bar{N}0}^{lu} \\ r_{01}^{lu} & \cdots & r_{0\bar{N}}^{lu} & r_{00}^{lu} \end{bmatrix}$.

Algorithm 2 Generation of SFVs for the inner string loops

Initialization: Set the string loop index $l = 2$ and the grid point index $u = 1$.

Repeat:

1) For the l-th string loop ($l \leq L$)
2) For the u-th grid point ($u \leq Q_l$)
3) Generate $s_u^{\mathrm{SL}_l}$ using (6).
4) $u = u + 1$ and go to Step 2.
5) $l = l + 1$ and go to Step 1.

Termination: Output $\left\{\mathbf{s}^{\mathrm{SL}_2}, \ldots, \mathbf{s}^{\mathrm{SL}_l}, \ldots, \mathbf{s}^{\mathrm{SL}_L}\right\}$ as the results.

Based on these definitions and according to the point-by-point approach proposed in [4], $s_u^{\mathrm{SL}_l}$ can be generated as

$$s_u^{\mathrm{SL}_l} = \left[\left(\mathbf{R}^{lu}\right)^{\frac{1}{2}}\right]_{\bar{N}+1,:} \begin{bmatrix} \left(\bar{\mathbf{R}}^{lu}\right)^{-\frac{1}{2}} \mathbf{z}^{lu} \\ a_u^{\mathrm{SL}_l} \end{bmatrix}, \tag{6}$$

where $a_u^{\mathrm{SL}_l}$ denotes an i.i.d. RV with a Gaussian distribution denoted as $\mathcal{N}\left(0, \sigma_{\mathrm{std}}^2\right)$ and its standard deviation σ_{std} depends on the scenario. In LTE networks, σ_{std} is suggested to be 8 dB and 10 dB for macrocells and picocells, respectively [9]. The proposed algorithm for the generation of SFVs for the inner string loops is formally presented in Algorithm 2.

IV. Simulation and Discussions

In this section, we conduct system-level simulations to verify the effectiveness of the proposed scheme. In our simulations, $M = 7$, $R_0 = 50$ m, $N = 18$ or 36, $\sigma_{\mathrm{std}} = 10$ dB, $d_{\mathrm{corr}} = 10$ m, $\epsilon = 0.1$ dB, and $L = 31$. Besides, for the calculation of the auto-correlation coefficients of SFVs, we investigate 4 reference points (RPs) (see Fig. 2), which are

- RP #1 - the centre point of cell 1, coordinates: $(0, 0)$
- RP #2 - the centre point of cell 2, coordinates: $\left(\frac{\sqrt{3}}{2}R_0, \frac{3}{2}R_0\right)$
- RP #3 - vertex 1, coordinates: $\left(\frac{\sqrt{3}}{2}R_0, \frac{5}{2}R_0\right)$
- RP #4 - the midpoint of $Edge_{(1)(2)}$, coordinates: $\left(\frac{3\sqrt{3}}{4}R_0, \frac{9}{4}R_0\right)$

Furthermore, 10,000 experiments were conducted to collect the auto-correlation results.

In Fig. 6, we plot a random snapshot of the wrap-around SFV map in dB scale with $N = 18$. As can be seen from Fig. 6, no noticeable discontinuity of SFVs can be observed across the border of the OA and the WA, which perfectly preserves the wrap-around feature discussed in Section II. Note that the snapshot of the wrap-around SFV map with $N = 36$ is very similar to Fig. 6, and thus it is not shown for brevity. In contrast, Fig. 7 shows an SFV map generated by the point-by-point approach [4], i.e., the SFVs of all the points are generated using Algorithm 2 with l initialized as one. This figure shows significant SFV discontinuity at the OA borders, which would lead to unrealistic SINR distributions in the network.

To investigate the errors of the auto-correlation coefficients given by the proposed scheme from the theoretical ones, we tabulate the maximum absolute correlation errors in Table III. Here, the correlation error is defined as the theoretical results

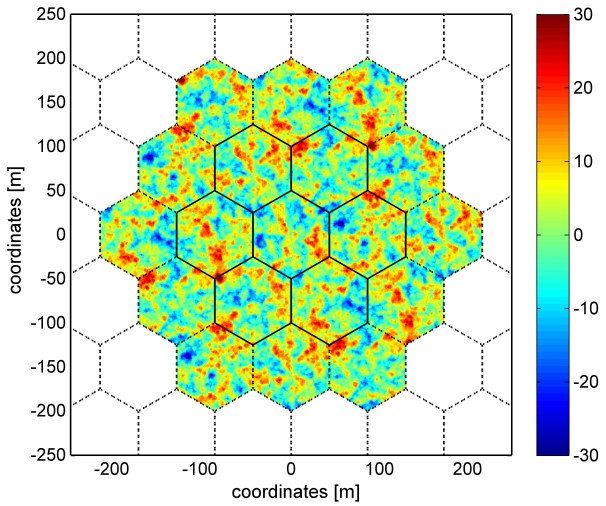

Figure 6. The SFV map generated by the proposed scheme ($N = 18$).

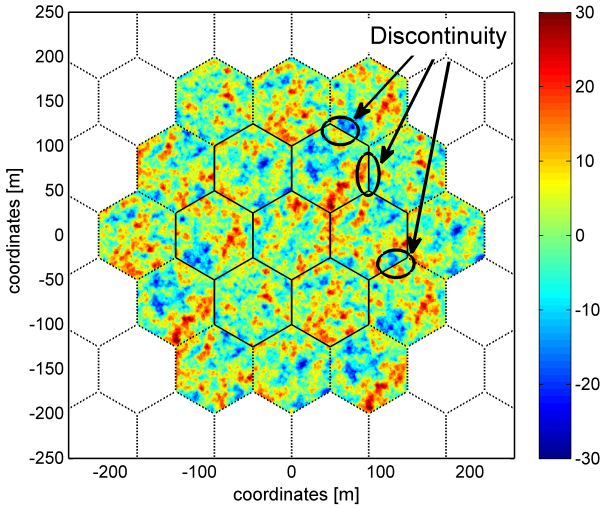

Figure 7. The SFV map generated by [4] ($N = 18$).

computed from (3) minus the average simulation results. From Table III, we can conclude that the maximum absolute correlation errors decrease quickly with the increase of N, especially for RPs #1 and #2, because more points are involved in the calculation of (6). When $N = 36$, the maximum absolute correlation error is only 0.14, which is reasonably small for practical system-level simulations. This verifies the accuracy and validity of the proposed scheme. It is interesting to note that the maximum absolute correlation errors for RPs #3 and #4 are smaller than those for RPs #1 and #2. This is because that RPs #3 and #4 are on the outmost string loop, which is treated before the inner string loops and its auto-correlation characteristics are well preserved since the other string loops follow the lead of the outmost one in the generation of SFVs. In comparison, when $N = 36$, the maximum absolute correlation errors resulted from the point-by-point approach [4] for RPs #1, #2, #3 and #4 are 0.1385, 0.1203, 1.0031 and 0.9937, respectively. Such large errors associated with RPs #3 and #4 are caused by the SFV discontinuity at the OA borders, as illustrated in Fig. 7.

Table III
MAXIMUM ABSOLUTE CORRELATION ERRORS (THE PROPOSED SCHEME)

RP	#1	#2	#3	#4
$N = 18$	0.2635	0.1728	0.0350	0.0619
$N = 36$	0.1400	0.1250	0.0345	0.0503

To illustrate the distribution of the correlation errors resulted from the proposed scheme, in Fig. 8, we plot the errors for the worst-case RP, i.e., RP #1, with $N = 36$. As can be observed from Fig. 8, the errors in most areas are below 0.1 indicated by the yellow color, which again confirms the effectiveness of the proposed scheme.

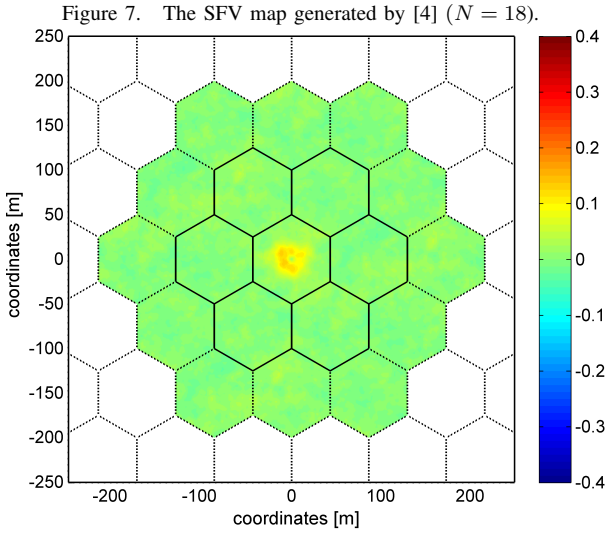

Fig. 8. Auto-correlation errors for RP #1 ($N = 36$).

V. CONCLUSION

In this paper, we propose a novel scheme for the generation of SFV maps with wrap-around. First, the OA is decomposed into a number of string loops. Then, SFVs are generated for the outmost string loop and the inner string loops using two proposed algorithms, respectively. Computer simulations verify the significant accuracy of the proposed scheme by showing that the maximum absolute correlation error of the proposed scheme can be controlled around 0.1 when $N = 36$.

REFERENCES

[1] M. Gudmundson, "Correlation model for shadow fading in mobile radio systems," *Electron. Lett.*, vol. 27, no. 2, pp. 2145-2146, Nov. 1991.
[2] 3GPP, "TR 25.896 (V6.0.0): Feasibility study for enhanced uplink for UTRA FDD," Mar. 2004.
[3] M. Ding, H. Luo, "Multi-point cooperative communication systems: theory and applications," Springer, 2013.
[4] H. Claussen, "Efficient modelling of channel maps with correlated shadow fading in mobile radio systems," *IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications 2005 (PIMRC '05)*, vol.1, pp.512-516, Sep. 2005.
[5] W. Ni, W. Zou, H. Wang, "Modeling of spatially cross-correlated shadow fading in distributed radio access networks," *IEEE International Conference on Commun. 2008 (ICC '08)*, pp.4472-4476, May 2008.
[6] X. Chu, D. Lopez-Perez, Y. Yang, F. Gunnarson, "Heterogeneous Cellular Networks," Cambridge University Press, 2013.
[7] L. N. Trefethen, D. Bau, "Numerical linear algebra," Philadelphia: Society for Industrial and Applied Mathematics, 1997.
[8] 3GPP, "TR 36.814 (V9.0.0): Further advancements for E-UTRA physical layer aspects," Mar. 2010.
[9] 3GPP, "TR 36.828 (V11.0.0): Further enhancements to LTE Time Division Duplex (TDD) for Downlink-Uplink (DL-UL) interference management and traffic adaptation," Jun. 2012.