Research Article

Computer Simulation of Stochastic Wind Velocity Fields for Structural Response Analysis: Comparisons and Applications

Filippo Ubertini¹ and Fabio Giuliano²

¹Department of Civil and Environmental Engineering, University of Perugia, Via G. Duranti 93, 06125 Perugia, Italy

²Department of Structural and Geotechnical Engineering, University of Rome "La Sapienza", Via Eudossiana 18, 00184 Rome, Italy

Correspondence should be addressed to Filippo Ubertini, filippo.ubertini@strutture.unipg.it

Received 14 October 2009; Revised 1 April 2010; Accepted 13 April 2010

Academic Editor: Chungxiang Li

Copyright © 2010 F. Ubertini and F. Giuliano. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The digital simulation of wind velocity fields, modeled as multivariate stationary Gaussian processes, is a widely adopted tool to generate the external input for response analysis of wind-sensitive nonlinear structures. The problem does not entail any theoretical difficulty, existing already a large number of well-established techniques, such as the accurate weighted amplitude wave superposition (WAWS) method. However, reducing the computational effort required by the WAWS method is sometimes necessary, especially when dealing with complex structures and high-dimensional simulation domains. In these cases, approximate formulas must be adopted, which however require an appropriate tuning of some fundamental parameters in such a way to achieve an acceptable level of accuracy if compared to that obtained using the WAWS method. Among the different techniques available for this purpose, autoregressive (AR) filters and algorithms exploiting the proper orthogonal decomposition (POD) of the spectral matrix deserve a special attention. In this paper, a properly organized way for implementing stochastic wind simulation algorithms is outlined at first. Then, taking the WAWS method as a reference from the viewpoint of the accuracy of the simulated samples, a comparative study between POD-based and AR techniques is proposed, with a particular attention to computational effort and memory requirements.

1. Introduction

The simulation of wind velocity fields has been one of the main topics of wind engineering for the last decades. In this framework, wind velocity is usually idealized as the sum of a mean part, assumed as constant within a conventional time interval, and a fluctuating part representing the atmospheric turbulence. This last is usually modeled as a stationary zero-mean Gaussian random process [1, 2].

Several techniques were proposed in the literature in order to simulate Gaussian wind velocity fields to be employed in structural analysis [3–5]. Among those, the classic WAWS method, based on the pioneering work by Shinozuka and Jan [6] and then modified by Deodatis [1] in such a way to achieve ergodic realizations and to be efficiently implemented through fast Fourier transform (FFT) algorithms, has proved to guarantee the best quality of the obtained results [7]. Nevertheless, such a proce-

dures requires the Cholesky factorization of the spectral matrix, which unfortunately leads to high computational expenses, especially when dealing with complex structures and high-dimensional simulation domains. These difficulties are mainly related to memory allocation and time consuming operations, thus requiring, on the one hand, the reduction of the problem size. On the other hand, an accurate wind simulation is essential for predicting the wind-induced response of flexible structures, such as transmission power lines, tall buildings, suspension, and cable-stayed bridges. Less demanding, yet approximate, procedures may be obtained by exploiting the properties of the POD decomposition of the spectral matrix, proposed in the papers by Li and Kareem [8], Di Paola [9], and Solari and Carassale [10]. A POD-based technique, in particular, was recently applied to simulate the wind velocity field on a domain representing a long-span suspension bridge [11] with significantly low computational efforts. A third well-established class of simulation formulas

FIGURE 1: Adopted computational scheme for wind simulation.

FIGURE 2: Grids for wind simulation: example 1 (a); example 2 (b).

FIGURE 3: Example 1: wind velocity field sample generated using the WAWS method: along-wind velocities (a); across-wind velocities (b).

FIGURE 4: Example 1 using WAWS method for $N_\omega = 2^{14}$ and $N_t = T_0/\Delta t$ (i.e., a process duration equal to one period T_0 ; black lines denote target functions, red lines denote numeric approximations): autospectrum of along-wind velocity at point 5 (a); corresponding autocorrelation function (b); cross-spectrum between along-wind velocities at points 5 and 9 (c); corresponding cross-correlation function (d).

is represented by autoregressive (AR) and autoregressive moving average (ARMA) filters, early introduced in the paper by Samaras et al. [12] and applied to wind simulation by Mignolet and Spanos [13], by Li and Kareem [8] and, more recently, by Di Paola and Gullo [14].

This paper addresses, at first, the problem of the efficient implementation of wind simulation algorithms by proposing a two-steps approach which allows to properly handle large numbers of simulations, which is usually the case for performing structural response analysis. Then, the WAWS method is adopted as a reference for calibrating the parameters of POD-based and AR techniques. These last methods are finally compared in terms of computational effort and memory requirements. To this end, two numerical examples are presented. The former, represented by the tower of a suspension bridge, is considered for models calibration. The latter, represented by an entire suspension bridge, is a more demanding case, which is here chosen as a benchmark in order to test the capability of AR and POD-based techniques in handling complex simulations and to compare their computational efficiency. Finally, the convergence rates of the accuracies of the two simplified methods are also investigated, in order to provide indications for tuning the parameters that affect their results.

The considered formulas for wind simulation are only some of those currently available in the literature and this work does not attempt in any way to give a comprehensive literature review on this large topic. In particular, it must be underlined that non-Gaussian techniques were also studied and applied in the literature with the aim of directly simulating wind pressure fields on structures. As examples, this problem was analyzed by Grigoriu [15, 16], by Popescu et al. [17], by Giofrè et al. [18] and by Borri and Facchini [19], among others. Non-Gaussian responses of wind-excited structures were also studied (see, e.g., Gusella and Materazzi [20, 21]). A quite exhaustive review of all the approaches currently available for simulating wind velocity fields and structural responses was recently proposed by Kareem [22] also considering the nonstationary and non-Gaussian cases.

2. Stochastic Wind Modeling

Let $Ox_1x_2x_3$ be the global Cartesian reference system with the origin O lying on the ground. In the following developments, without loss of generality, it is assumed that the x_3 axis is parallel to the gravity direction and the mean wind velocity U is parallel to the x_1 axis. Following these definitions,

FIGURE 5: Example 1 using WAWS method (black lines denote target functions, gray lines denote numeric approximations): autospectrum of along-wind velocity at point 5 (a); corresponding autocorrelation function (b); cross-spectrum between along-wind velocities at points 5 and 9 (c); corresponding cross-correlation function (d).

the wind velocity field $\mathbf{V}(\mathbf{x}, t)$ is idealized as the sum of a mean value $\mathbf{U}(x_3)$, function of the elevation from the ground x_3 , and a stationary zero-mean fluctuation $\mathbf{u}(\mathbf{x}, t) = (u_1, u_2, u_3)$, that depends on the position $\mathbf{x} = (x_1, x_2, x_3)$ and varies in time t . The three components u_1, u_2, u_3 of the vector $\mathbf{u}(\mathbf{x}, t)$ represent therefore the longitudinal, lateral and vertical components of turbulence, respectively. As customary in wind engineering, the classic logarithmic law [4] is assumed to represent the variability of the mean wind velocity $\mathbf{U}$ with x_3 .

The atmospheric turbulence $\mathbf{u}$ is usually modeled as a zero-mean, Gaussian, stationary random field that depends on time. If P and P' denote two points located at $\mathbf{x}$ and $\mathbf{x}'$, then, from a probabilistic point of view, the complete characterization of this field is ensured by the knowledge of the correlation function $R_{u_i u_j}(\mathbf{x}, \mathbf{x}', \tau)$ for every pair of turbulence components u_i, u_j , τ being the time lag. Assuming that $\mathbf{u}$ is ergodic, $R_{u_i u_j}(\mathbf{x}, \mathbf{x}', \tau)$ is given by the Fourier transform of the cross power spectral density (CPSD) function $S_{u_i u_j}(\mathbf{x}, \mathbf{x}', \omega)$ between u_i and u_j , ω being the circular frequency. Most of the theoretical models adopted in wind engineering express the CPSD in terms of auto-spectra $S_{u_i u_i}(\mathbf{x}, n)$, $S_{u_j u_j}(\mathbf{x}', n)$ and coherence function $\text{Coh}_{v_i v_j}(\mathbf{x}, \mathbf{x}', n)$, for $i, j = 1, 2, 3$, where $n = \omega/2\pi$ denotes the frequency [9]. Several models were developed in the

literature to give an analytical representation to $S_{u_i u_j}(\mathbf{x}, n)$, for $j = 1, 2, 3$. Here, the model by Solari and Piccardo [23] is adopted for this purpose and $\text{Coh}_{v_i v_j}(\mathbf{x}, \mathbf{x}', n)$ is expressed by means of the classic exponential law [11]. The imaginary part of the CPSD (called quadrature spectrum) introduces a time lag between the simulated velocities, which may become significant for points placed in the along-wind direction x_1 . Without loss of generality, the quadrature spectrum is here neglected, since the considered examples focus on simulation domains located on the $x_2 x_3$ plane. This, however, does not limit the generality of the presented results, which apply also to the case in which the CPSD has a nonnegative imaginary part.

For simulation purposes, the spatial domain is discretized into N points which usually represent significant nodes of the case study structure. The position of the k th point is identified by the vector $\mathbf{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, x_3^{(k)})$, with $k = 1, 2, \dots, N$. This spatial discretization allows to represent the wind field $\mathbf{V}^{(k)}(t) = \mathbf{V}(\mathbf{x}^{(k)}, t)$ as the sum of the mean velocity vectors $\mathbf{U}^{(k)} = \mathbf{U}(\mathbf{x}^{(k)})$ and the turbulence vectors $\mathbf{u}^{(k)}(t) = \mathbf{u}(\mathbf{x}^{(k)}, t) = (u_1^{(k)}, u_2^{(k)}, u_3^{(k)})$. Following this approach, the time-dependent random field $\mathbf{u}(\mathbf{x}, t)$ is transformed into a $3N$ -variate stationary random process $\underline{\mathbf{u}}(t)$, where $\underline{\mathbf{u}}(t)$ is a $3N$ -order vector containing the components $u_1^{(k)}, u_2^{(k)}, u_3^{(k)}$ of the turbulence vectors

FIGURE 6: Example 1 using WAWS method (black lines denote target functions, gray lines denote numeric approximations): autospectrum of across-wind velocity at point 5 (a); corresponding autocorrelation function (b); cross-spectrum between across-wind velocities at points 5 and 9 (c); corresponding cross-correlation function (d).

$\mathbf{u}^{(k)}(t)$ for $k = 1, 2, \dots, N$. The complete characterization of the process $\mathbf{u}(t)$ is given by its power spectral density (PSD) matrix $\mathbf{S}_{\mathbf{u}\mathbf{u}}(n)$ containing PSD and CPSD functions between turbulent velocities. Equivalently, the two-side PSD matrix $\mathbf{G}_{\mathbf{u}\mathbf{u}}(\omega)$ can also be utilized [11]. Having neglected the quadrature spectrum, both matrices $\mathbf{S}_{\mathbf{u}\mathbf{u}}(n)$ and $\mathbf{G}_{\mathbf{u}\mathbf{u}}(\omega)$ are real and positive definite.

3. Digital Simulation Formulas

Realizations of the process $\mathbf{u}(t)$ are generated along a sequence t_j for $j = 1, 2, \dots, N_t$, of time instants with constant time step Δt . The circular frequency domain is limited within the interval $[-\omega_c, \omega_c]$ where ω_c denotes the cut-off circular frequency. Such an interval is discretized by means of an equally spaced sequence ω_k for $k = 1, 2, \dots, N_\omega$ of circular frequencies, with step amplitude $\Delta\omega$, being thereby $\omega_c = N_\omega/2\Delta\omega$.

The most accurate and well-established algorithm for simulating samples of Gaussian processes is the spectral representation method, early proposed by Shinozuka and Jan [6]. This last was significantly improved by Deodatis [1] who proposed a simulation formula that ensures the generation of ergodic sample functions and which is suitable to be

implemented using efficient FFT algorithms. The Shinozuka-Deodatis formula is based on the following frequency-dependent decomposition of $\mathbf{G}_{\mathbf{u}\mathbf{u}}(\omega_k)$:

$$\mathbf{G}_{\mathbf{u}\mathbf{u}}(\omega_k) = \mathbf{T}(\omega_k)\mathbf{T}^*(\omega_k), \quad k = 1, 2, \dots, N_\omega, \quad (1)$$

where $*$ denotes the complex conjugate operator. It is worth noting that the decomposition (1) is not unique. To this regard, in the Shinozuka-Deodatis method the Cholesky factorization algorithm is adopted which provides $\mathbf{T}(\omega_k)$ in lower triangular form. Based on this approach, the following digital simulation formula was obtained [1]:

$$\begin{aligned} \underline{u}_i(t_j) &\cong \sqrt{\Delta\omega} \sum_{r=1}^i \sum_{k=1}^{N_\omega} T_{ir}(\omega_{rk}) \cos(\omega_{rk}t_j + \varphi_{rk}), \\ i &= 1, 2, \dots, 3N, \quad j = 1, 2, \dots, N_t, \end{aligned} \quad (2)$$

where $\omega_{rk} = \omega_k + r\Delta\omega/3N$, $T_{ir}(\omega_{rk})$ are the elements of matrix $\mathbf{T}(\omega_{rk})$ and φ_{rk} are independent random phases uniformly distributed in the interval $[0, 2\pi]$. Without loss of generality, (2) is written under the assumption of neglecting the quadrature spectrum, therefore $\mathbf{T}(\omega_{rk})$ being a real matrix. This method is sometimes called “weighted amplitude waves superposition” (WAWS) method. As observed

FIGURE 7: Example 1 using POD method (black lines denote target functions, gray lines denote numeric approximations): autospectrum of along-wind velocity at point 5 (a); corresponding autocorrelation function (b); cross-spectrum between along-wind velocities at points 5 and 9 (c); corresponding cross-correlation function (d).

by Di Paola [9], the central limit theorem ensures that the process simulated by means of (2) is asymptotically Gaussian as N_ω becomes large. Equation (2) can be efficiently implemented through a fast Fourier transform (FFT) algorithm, as discussed in [1].

An alternative approach to the WAWS formula is represented by the POD-based technique proposed by Carassale and Solari [11]. With such an approach the matrices $\mathbf{T}(\omega_k)$, in (1), are calculated as

$$\mathbf{T}(\omega_k) = \sum_{r=1}^{3N} \boldsymbol{\theta}_r^{(k)} \sqrt{\gamma_r^{(k)}} \quad k = 1, 2, \dots, N_\omega, \quad (3)$$

where $\gamma_r^{(k)}$, for $r = 1, 2, \dots, 3N$, are the eigenvalues of $\mathbf{G}_{\mathbf{u}\mathbf{u}}(\omega_k)$ and $\boldsymbol{\theta}_r^{(k)}$ are the corresponding eigenvectors. These last assume the well-known physical meaning of wind blowing mode shapes [9]. The use of (3) leads to the following simulation formula:

$$\mathbf{u}(t_j) \cong \sum_{r=1}^{3N} \sum_{k=1}^{N_\omega} e^{\iota \omega_k t_j} \sqrt{\Delta \omega \gamma_r^{(k)}} \boldsymbol{\theta}_r^{(k)} p_r^{(k)}, \quad j = 1, 2, \dots, N_t, \quad (4)$$

where ι is the imaginary unit and $p_r^{(k)}$ are complex-valued Gaussian random numbers with unit variance. Likewise (2),

(4) can be efficiently implemented through a fast Fourier transform (FFT) algorithm as discussed in [11], assuming again $N_\omega = N_t$ and $\Delta \omega = 2\pi/N_t \Delta t$.

Equation (4) generates $\mathbf{u}(t)$ as the superposition of $3N$ independent fully coherent stochastic processes, which represent the contributions of the different wind modes. A very convenient way for reducing both computer time and allocated memory using (4) is to calculate eigenvectors and eigenvalues of $\mathbf{G}_{\mathbf{u}\mathbf{u}}(\omega_k)$ only for a small number $\hat{N}_\omega \ll N_\omega$ of circular frequencies distributed along a reduced sequence $\hat{\omega}_k$, for $k = 1, 2, \dots, \hat{N}_\omega$, and then use interpolation formulas elsewhere. This approach, proposed by Carassale and Solari [11], is here adopted and briefly recalled in Appendix A.

Autoregressive methods generate the wind velocity field by filtering a $3N$ -order vector $\mathbf{a}(t)$ of uncorrelated band-limited Gaussian white noises $\mathbf{a}_j(t)$, for $j = 1, 2, \dots, 3N$, with unit variance. The turbulent wind process is thus expressed in the following form:

$$\begin{aligned} \mathbf{u}(t_j) \cong & \sum_{i=1}^p \boldsymbol{\Phi}_i \mathbf{u}(t_j - i \Delta t) \\ & + \sum_{i=0}^q \mathbf{B}_i \mathbf{a}(t_j - i \Delta t), \quad j = 1, 2, \dots, N_t, \end{aligned} \quad (5)$$

FIGURE 8: Example 1 using POD method (black lines denote target functions, gray lines denote numeric approximations): autospectrum of across-wind velocity at point 5 (a); corresponding autocorrelation function (b); cross-spectrum between across-wind velocities at points 5 and 9 (c); corresponding cross-correlation function (d).

where Φ_i and $\mathbf{B}_i$ are convenient $3N \times 3N$ matrices that can be easily calculated by imposing that the simulated process satisfies the covariance structure of the target one [12]. Equation (5) represents a general ARMA method, in which p denotes the order of autoregression and q is the order of the moving average component. As it is well known [7] an ARMA(p, q) can be approximated by an AR(p_1), with $p_1 \gg p$, where the AR(p_1) filter is readily obtained by assuming $q = 0$ in (5). Regarding this point, it must be mentioned that AR(20) and ARMA(5,5) provided good results for turbulent wind simulations in many different environmental conditions [7]. The necessary derivations for calculating the coefficient matrices of an AR(p_1) filter are reported in Appendix A.

4. Efficient Implementation of Wind Simulation Methods

The three simulation algorithms presented in Section 3 are implemented in the MATLAB R2009a [24] environment to generate samples of along-wind and across-wind turbulence velocities u_1 and u_3 , respectively. Without loss of generality, the turbulent wind field is thus reduced to a $2N$ -variate stationary Gaussian process.

Equation (2) is implemented as the simulation formula of the WAWS method, while (4) is adopted in the POD-based technique. Both in the WAWS and the POD methods, FFT algorithms are employed, as discussed in [1, 11], to improve their computational efficiency. An AR(p_1) filter, given by (5) with $q = 0$, is considered to represent autoregressive methods assuming a sufficiently large p_1 . A two-step approach is adopted in the implemented codes, following the diagram reported in Figure 1. Accordingly, an initial phase is established (indicated as “phase 0”), which must be run only once, after which a general number n of simulations (Sim) can be performed (phase 1). Typically, in the “phase 0”, all the data which are needed for the successive simulations (e.g., spectral matrix, factorizations, coefficients, etc.) are calculated and collected. This allows minimizing the number of operations when simulating several wind realizations, which is usually the case for performing Monte Carlo simulations of the structural response. The results of the “phase 0” are deterministic and must be stored in the computer memory.

The two main aspects that should be considered when comparing different wind simulation techniques, from the computational point of view, are the allocated memory in the “phase 0” and the time needed for each single simulation

FIGURE 9: Example 1 using AR method (black lines denote target functions, gray lines denote numeric approximations): autospectrum of along-wind velocity at point 5 (a); corresponding autocorrelation function (b); cross-spectrum between along-wind velocities at points 5 and 9 (c); corresponding cross-correlation function (d).

T_{sim} in the “phase 1”. The former aspect is crucial since collecting a lot of data in the computer memory may cause overflows and thus it may preclude the use of a certain procedure when the dimension of the simulation domain becomes large. The latter aspect is even more important since it gives the measure of the computational effort required by the considered algorithm. On the other hand, when the accuracy of the different methods is concerned, the most relevant points that should be considered are the agreement between target and simulated characteristics of the stochastic process, mainly spectra and correlation functions, and peak values.

The three methods described above are implemented as schematically outlined in the pseudocodes provided in Appendix B. Accordingly, the main computational steps are summarized in Table 1. The comparison between the operations required by the considered algorithms immediately outlines that the WAWS technique is the heaviest one from the computational viewpoint. On the contrary, both POD and AR techniques significantly reduce the computational effort with respect to the WAWS method. Table 1 also summarizes the memory that is allocated in the “phase 0” in each of the three implemented codes. It is worth underlying that the ratio between the memory allocated

in the POD-based method and the one allocated in the WAWS method is almost equal to $\hat{N}_\omega/N_\omega$. In the next section it will be shown that the POD-based algorithm gives sufficiently accurate results with $\hat{N}_\omega = 50$, if compared to the corresponding WAWS method with $N_\omega = 2^{14}$. The POD-based technique allows therefore drastically reducing the memory required by the WAWS method. This practically eliminates the risk of memory overflows in many technical conditions. Nonetheless, in the most demanding cases, an $\text{AR}(p_1)$ filter can be adopted as it further reduces memory allocation even with respect to the POD-based method. Indeed, the ratio between the memory allocated by the AR method and the one allocated by the POD method is equal to $(p_1 + 1)/\hat{N}_\omega$ and, usually, $p_1 + 1 \ll \hat{N}_\omega$.

5. Numerical Tests and Discussion

5.1. The Case Studies. The above described wind simulation techniques are commonly employed for the definition of the external inputs for time domain response analyses of slender structures, as described in [25–28], among others. Time domain response analysis are essential for evaluating the structural safety of wind sensitive structures, whose dynamic

FIGURE 10: Example 1 using AR method (black lines denote target functions, gray lines denote numeric approximations): autospectrum of across-wind velocity at point 5 (a); corresponding autocorrelation function (b); cross-spectrum between across-wind velocities at points 5 and 9 (c); corresponding cross-correlation function (d).

behavior is strongly nonlinear, for instance, in the case of structural cables and long-span bridges [29, 30].

Two numerical examples are considered in order to evaluate the performances of the simplified models, when their parameters are chosen in such a way to obtain a similar accuracy with respect to the WAWS method. In the former example the simulation domain is composed by 9 significant points disposed along the height of the tower of a suspension bridge (see Figure 2(a)). This case is here adopted in order to calibrate the parameters of POD-based and AR techniques. In the latter example the simulation domain is represented by an entire suspension bridge. In this case, the spatial domain is composed by 83 nodes located on the mean plane of the bridge (see Figure 2(b)). In both cases, the direction of the mean wind velocity is orthogonal to the plane of the simulation domain and its modulus is assigned by means of the classic logarithmic profile. In particular, a mean velocity of 48.0 m/s is assumed at the top of the tower in the former example (node number 9 in Figure 2(a)) and a mean velocity of 40.1 m/s is assumed at the mid-span of the bridge in the latter example (node 70 in Figure 2(b)). The exponential decay coefficients that appear in the expression of the coherence function are assumed to be equal to those

reported in [11], while the variances $\sigma_{u_j^{(k)}}^2$ of $u_j^{(k)}$, for $j = 1, 2, 3$, are calculated following [23]. A nil coherence is assumed between orthogonal turbulence components u_1 and u_3 . The following parameters are adopted in the simulations: $\Delta t = 0.05$ s, $N_t = N_\omega = 2^{14}$, $\Delta\omega = 0.00767$ rad/s, $\omega_c = 62.8$ rad/s. The simulations have been performed with the aid of a Core2 Duo Intel processor.

5.2. Calibration of POD-Based and AR Methods. Before discussing the performances of the simplified formulas, the parameters of POD-based and AR techniques must be properly calibrated in order to achieve similar accuracies with respect to the WAWS method. This task is here accomplished in the case of the first example. Literature studies [7, 11] suggest to choose, in similar cases, a reduced number of factorizations (for POD-based technique only) $\hat{N}_\omega = 50$ and an order of autoregression (for AR filter only) $p_1 = 20$. After performing several tests, it was found that similar values could be also adopted here, as it will be clearer in the rest of the paper by comparing the quality of the results obtained by means of the three methods. Therefore, the values $\hat{N}_\omega = 50$ and $p_1 = 20$ are chosen here, thus also allowing a direct comparability with the results presented

FIGURE 11: Example 2 using POD method (black lines denote target functions, gray lines denote numeric approximations): auto-spectra of along-wind velocities at points 1 and 70 (a), (b); corresponding autocorrelation functions (c), (d); cross-spectrum of along-wind velocities in points 1 and 70 (e); corresponding cross-correlation function (f).

in other literature works. As an example, a realization of the turbulent wind field generated by means of the WAWS method is shown in Figure 3.

The quality of the generated samples is analyzed in Figures 4–10. To this end, the turbulence velocities simulated in two points (number 5 and 9 in Figure 2(a)) of the spatial domain are considered. First of all, by virtue of the ergodicity of the process simulated by means of (2), it is expected that the WAWS method provides a perfect accuracy when one entire period $T_0 = 2N \cdot 2\pi/\Delta\omega$ (for a $2N$ -variate process) of

the process is simulated. This result has been confirmed here as shown, for instance, in the results presented in Figure 4, which have been obtained by assuming $N_t = T_0/\Delta t$ and $N_\omega = 2^{14}$. In this case, as expected, target and computed spectra and correlation functions are overlapped (some meaningless residual differences are related to numeric FFT calculations). Figures 5 and 6 refer to along-wind and across-wind (vertical) velocities, respectively, simulated by means of the WAWS method assuming $N_t = 2^{14}$, that is, $N_t \ll T_0/\Delta t$. The corresponding results obtained by means of POD-based

FIGURE 12: Example 2 using POD method (black lines denote target functions, gray lines denote numeric approximations): auto-spectra of across-wind velocities at points 1 and 70 (a), (b); corresponding autocorrelation functions (c), (d); cross-spectrum between along-wind velocities at points 1 and 70 (e); corresponding cross-correlation function (f).

and AR techniques, for the chosen parameters values, are shown in Figures 7, 8, 9, and 10. From the presented results it points out that the WAWS method provides a very good match between target and computed spectra and correlation functions, even for $N_t \ll T_0/\Delta t$. The chosen POD-based and AR methods are obviously less accurate than the WAWS formula, which is especially apparent by looking at the errors on auto- and cross-correlation functions. In particular, the analysis of the results, also extended to other points of the

spatial domain, confirms that POD-based and AR techniques produce errors in the correlation functions, although, in general, the agreement with the target functions is improved as the coherence is decreased (as, e.g., in the across-wind components). Clearly, the desired levels of accuracy could be obtained in both cases of POD-based and AR algorithms by choosing larger values of either the reduced number of factorizations $\hat{N}_\omega$, adopted in the former case, or the finite order p_1 chosen in the latter one. For the purposes of this

FIGURE 13: Example 2 using AR method (black lines denote target functions, gray lines denote numeric approximations): auto-spectra of along-wind velocities at points 1 and 70 (a), (b); corresponding autocorrelation functions (c), (d); cross-spectrum between along-wind velocities at points 1 and 70 (e); corresponding cross-correlation function (f).

study, the parameters $\hat{N}_\omega = 50$ and $p_1 = 20$ are considered as good compromises between accuracy and computational efficiency. Indeed, with such choices, the accuracies of the samples generated using POD-based and AR techniques are essentially comparable to each other and the errors with respect to the WAWS method can be regarded as acceptable in view of a structural response analysis [28]. In any case, the comments reported in the following developments of the work could be readily extended to other possible choices of the models' parameters.

5.3. Discussion. The results presented so far have permitted to choose the parameters of POD-based and AR methods that guarantee efficient simulations with an accuracy that is essentially comparable to that of the WAWS method. The second example is now worth considering to better compare POD and AR methods from the computational viewpoint.

The accuracy of the simulated samples is analyzed in Figures 11, 12, 13, and 14. To this end, to the along-wind and across-wind velocities at the points number 1 (top of one tower) and 70 (bridge mid-span), indicated in Figure 2(b),

FIGURE 14: Example 2 using AR method (black lines denote target functions, gray lines denote numeric approximations): auto-spectra of across-wind velocities at points 1 and 70 (a), (b); corresponding autocorrelation functions (c), (d); cross-spectrum between along-wind velocities at points 1 and 70 (e); corresponding cross-correlation function (f).

are considered. The presented results are analogous to those obtained in the first example, thus indicating that the quality of the simulated process is, as expected, independent on the dimension of the simulation domain. In particular, the results confirm that POD-based and AR methods, for the chosen values of $\hat{N}_\omega$ and p_1 , essentially guarantee similar accuracies. The samples generated by means of the three algorithms are also in good agreement as it concerns the peak values $u_1^{(k)}_{\max}$ and $u_3^{(k)}_{\max}$ of $u_1^{(k)}$ and $u_3^{(k)}$. As examples, some

relevant results are summarized and compared in Table 2 for the two numerical examples.

The computer times required by the considered methods in the two numerical examples are summarized in Table 3. These results emphasize that, as expected, POD and AR methods are significantly more computationally efficient than the WAWS formula. Although it is implicit that these results strongly depend on the environment of simulation, they still allow to derive some conclusions. Particularly,

FIGURE 15: Example 2: first ten nondimensional eigenvalues (POD) of atmospheric turbulence as a function of the circular frequency ω (a); detailed view (b).

FIGURE 16: Example 2: first 6 blowing mode shapes evaluated for the circular frequency value $\omega = 0.07$ rad/s.

FIGURE 17: Normalized average errors of simulated samples using AR filter (a) and POD-based technique (b) by increasing the order of autoregression (AR filter) and the reduced number of factorizations (POD-based method).

TABLE 1: Two-steps implementation of the considered simulation methods and corresponding memory allocation for a $2N$ -variate process (the symbol $\propto$ denotes proportionality).

	WAWS	POD	AR
Phase 0	calculate $2N \cdot N_w$ Cholesky matrices $\mathbf{T}(\omega_{rk})$ of dimension $2N \times 2N$ and store $2N \cdot N_w$ column vectors $\mathbf{T}_{:,r}(\omega_{rk})$ of length $2N$	calculate and store $\hat{N}_w$ matrices of eigenvectors $\boldsymbol{\theta}_r^{(k)}$ of dimension $2N \times 2N$ and $2N \cdot \hat{N}_w$ eigenvalues $\gamma_r^{(k)}$	calculate and store p_1 matrices $\boldsymbol{\Phi}_i$ of dimension $2N \times 2N$ and one matrix $\mathbf{B}_0$ of dimension $2N \times 2N$
Phase 1	sum $N(2N+1)N_w$ scalar numbers and perform $N(2N+1)$ inverse FFTs	perform N_w interpolations of $\gamma_r^{(k)}$ and $\boldsymbol{\theta}_r^{(k)}$, sum $N_w \cdot (2N+1)$ vectors of length $2N$ and perform $2N$ inverse FFTs	perform $(p_1+1)N_t$ multiplications of matrices $\boldsymbol{\Phi}_i$ and $\mathbf{B}_0$ by vectors of length $2N$ and sum $(p_1+1)N_t$ vectors of length $2N$
Allocated memory in phase 0	$\propto (2N)^2 N_w$	$\propto (2N)^2 \hat{N}_w$	$\propto (2N)^2 (p_1+1)$

TABLE 2: Comparison between peak values of samples generated using the three simulation methods.

	18-variate process (example 1)		166-variate process (example 2)	
Method	$u_{1 \max}^{(9)}$ (m/s)	$u_{3 \max}^{(9)}$ (m/s)	$u_{1 \max}^{(70)}$ (m/s)	$u_{3 \max}^{(70)}$ (m/s)
WAWS	18.9	9.5	—	—
POD	19.8	10.2	22.0	9.2
AR	18.0	9.7	21.0	9.5

the POD-based method seems to be more computationally efficient than the AR filter, both in performing preliminary calculations (“phase 0”), which could become relevant when computing only a few wind simulations, and wind simulations (“phase 1”). Moreover, the computational improvement in using the POD-based technique instead of

TABLE 3: Computational efforts required by the three different methods for digital wind simulation (T_0 and T_{sim} denote the computer times necessary for performing the “phase 0” and one single simulation in the “phase 1”, resp.).

	18-variate process (example 1)			166-variate process (example 2)		
Method	Memory (Phase 0)	T_0 (s)	T_{sim} (s)	Memory (Phase 0)	T_0 (s)	T_{sim} (s)
WAWS	11.3 MB	347.2	134.0	—	—	—
POD	0.1 MB	0.6	1.1	10.0 MB	5.3	8.2
AR	0.3 MB	10.8	1.7	2.2 MB	650.2	97.3

the AR filter seems to become more relevant as long as the dimension of the simulation domain becomes larger. On the other hand, as expected, the AR filter is the method that mostly reduces memory allocation, at least in the case of high-dimensional simulation domains, which might be greatly beneficial in practical cases.

It is also worth mentioning that the eigenvalue analysis of the spectral matrix, performed in the POD-based method, reveals interesting properties of the wind field in view of a structural analysis. In particular, as it is well-known, eigenvectors and eigenvalues of the spectral matrix represent respectively the shapes and the associated powers of the $3N$ stochastic processes summed in (4). The eigenvalue analysis usually reveals that only a few of these spectral modes have large associated powers and that the eigenvectors corresponding to the largest eigenvalues are very similar to the fundamental mode shapes of the structure. Thus, the response of a slender structure is expected to be dominated by the contribution of these waves [9].

As an example, Figure 15(a) shows the first ten eigenvalues $\gamma_r^{(k)}$ ($r = 1, \dots, 10$) multiplied by the frequency n and plotted versus the circular frequency ω , in the case of the second example. As inferable from this figure, the first few eigenvalues are, as expected, much greater than the others. It is also worth mentioning that, as shown in Figure 15(b), the eigenvalues of the spectral matrix are distinct and thus the lines $n\gamma_r^{(k)}$ versus ω do not cross. The turbulence eigenvectors corresponding to the first six eigenvalues, calculated for a fixed value of the circular frequency ($\omega = 0.07$ rad/s), are shown in Figure 16. The figure evidences that the first turbulence eigenvector involves the whole bridge in the

out-of-plane (along-wind) direction and it is very similar to the typical first structural mode shape of a suspension bridge. Similar observations can be made for the remaining eigenvectors shown in Figure 16. It is noteworthy that hybrid horizontal/vertical eigenvectors are not detected since a nil coherence has been assumed between along-wind and across-wind turbulence components.

The presented results have indicated that the POD-based technique exhibits a significant computational efficiency, while the AR filter is especially convenient for preserving the allocated memory. Another important aspect that needs to be addressed in the viewpoint of practical applications is the rate of convergence of the accuracies of the AR filter and the POD-based technique towards the target, when increasing the relevant control parameters, that is, the order of autoregression, p_1 , in the case of the AR filter, and the reduced number of factorizations, $\hat{N}_\omega$, in the case of the POD-based technique. This aspect is analyzed here, by considering the error, ε , between correlation functions of generated samples, $R_{u_1^{(i)} u_1^{(j)}}^{\text{generated}}$ and $R_{u_3^{(i)} u_3^{(j)}}^{\text{generated}}$, and the corresponding targets, $R_{u_1^{(i)} u_1^{(j)}}^{\text{target}}$ and $R_{u_3^{(i)} u_3^{(j)}}^{\text{target}}$. This error is here defined as:

$$\varepsilon = \sum_{i=1}^N \sum_{j=i}^N \left\| R_{u_1^{(i)} u_1^{(j)}}^{\text{generated}} - R_{u_1^{(i)} u_1^{(j)}}^{\text{target}} \right\| + \sum_{i=1}^N \sum_{j=i}^N \left\| R_{u_3^{(i)} u_3^{(j)}}^{\text{generated}} - R_{u_3^{(i)} u_3^{(j)}}^{\text{target}} \right\| \quad (6)$$

and calculated in the time lag interval 0–10 sec. The error defined in (6) is realization dependent and, so, an averaging of ε among a conveniently large number of generations N_{gen} is sought, where N_{gen} is here taken as 200.

Following the previous definitions, Figure 17 shows plots of the average errors versus p_1 and $\hat{N}_\omega$, for the AR filter and the POD-based technique, in the case of the first numerical example. For clarity of presentation, the results in Figure 17 are normalized with respect to the residual error of the WAWS method with $N_t = N_\omega = 2^{14}$. The results presented in Figure 17 confirm the expected trends that the errors decrease with increasing p_1 and $\hat{N}_\omega$. Moreover, they show that, in the case of the POD-based technique, the rate of convergence appears to be very fast for small values of $\hat{N}_\omega$. Namely, already for $\hat{N}_\omega = 2^4$ the system reaches an accuracy which is only slightly worse than that of the WAWS method with $N_t = N_\omega = 2^{14}$ (the normalized average error is about equal to 1.3), while for larger values of $\hat{N}_\omega$ the convergence rate becomes smaller with some erratic fluctuations of less than $\pm 5\%$. This result indicates that the interpolation expressions of eigenvectors and eigenvalues along the unequally spaced sequence $\hat{\omega}_k$ of circular frequencies, proposed by Carassale and Solari [11] and recalled in Appendix A, is very accurate when applied in the case of turbulence characteristics similar to those considered here. The results presented in Figure 17 might also suggest to choose values of $\hat{N}_\omega$ that are smaller than the one adopted by Carassale and Solari [11] in a similar application and chosen in the numerical examples presented above ($\hat{N}_\omega = 50$), without substantially decreasing the resulting accuracy. Nonetheless, calculations performed

aside have shown that the average error in predicting peak values, calculated with respect to the WAWS method and considering 200 wind generations, rapidly decreases for $\hat{N}_\omega$ up to about 50, where it is almost equal to 0.5%, while remaining essentially constant for larger values of $\hat{N}_\omega$. This result shows that, in the present case, choosing $\hat{N}_\omega \cong 50$ is convenient in terms of accuracy in predicting peak values.

In the case of the AR filter, the results presented in Figure 17 show that for large values of p_1 the error is still slightly larger than the error of the WAWS method with $N_t = N_\omega = 2^{14}$ (the ratio between the two is about 2.5). On this respect, a value of p_1 equal to 200 can be regarded as large if compared to the value $p_1 = 20$ recommended in the literature in similar conditions [7] and adopted in the numerical examples presented above. Looking at the results presented in Figure 17, the ratio between the average error of the AR filter and the residual error of the WAWS method, for $p_1 = 20$, is about equal to 3.2. Considering the high accuracy of the WAWS method, this result can be regarded as acceptable in many practical applications. It is also worth mentioning that, for $p_1 = 20$, the average error in predicting peak values is also small and approximately equal to 2.5%.

It is important to note that the results presented in Figure 17 do not allow to derive conclusions on the relative accuracy of the two simplified methods one with respect to the other. Indeed, for a fair and complete comparison between the two methods, different turbulence characteristics should be considered, which, however, goes beyond the purposes of the present investigation.

6. Conclusions

Simulating the wind velocity field in the case of high-dimensional domains may become a very demanding computational task due to memory occupation and time consuming simulations. In the present paper, a properly organized way for implementing wind simulation methods is presented at first. Then, the widely adopted method using waves superposition (WAWS method) is taken as a reference for calibrating the parameters of two well-known simplified methods (POD-based and AR formulas). These last are then compared, by devoting a special care to algorithm structure and computational expense.

Two numerical examples, with increasing complexity, are considered in the comparative study. The results indicate that POD-based and AR techniques significantly reduce both computer time and memory allocation with respect to the WAWS method, at the expense of smaller accuracies. The POD-based method appears to be more computationally efficient than the AR filter when the relevant parameters affecting their results are tuned in such a way to achieve similar accuracies. On the other hand, the AR filter is greatly preferable in terms of reduction of allocated memory.

Concerning the accuracy of the simulated samples, the WAWS method is known to guarantee the ergodicity of the generations. As shown in the paper, this means that if an entire period of the process is simulated, the match between target and simulated samples is perfect. Even when

considering shorter simulations, the accuracy of the WAWS method is seen to be very high, namely the differences between target and computed correlation functions are, for the considered parameters values, almost unnoticeable. The results presented in this paper also confirm that, although less accurate, both POD-based and AR techniques can provide good results if the models' parameters are properly chosen. It must be also mentioned that, regardless the chosen simulation formula, the POD decomposition of the spectral matrix is always a worth task to be tackled, as it provides useful information about the stochastic wind process in view of a structural analysis. Finally, the rates of convergence of the accuracies of the two simplified approaches with increasing tuning parameters have also been investigated in order to give preliminary indications for choosing these parameters in practical cases.

Appendices

A. Additional Details on the Adopted Simulation Formulas

The rules proposed by Carassale and Solari [11] for reducing the number of eigenvectors and eigenvalues of matrix $\mathbf{G}_{\mathbf{u}\mathbf{u}}(\omega_k)$ allocated in the computer memory, are described, at first, in this section. Then, the derivations for calculating the coefficient matrices of an AR (p_1) filter are briefly recalled.

Concerning the POD-based technique, the following sequence of circular frequencies must be introduced:

$$\hat{\omega}_h = \frac{2\pi}{N_t \Delta t} \left(\frac{N_t}{2} \right)^{h-1/\hat{N}_\omega - 1}, \quad (h = 1, \dots, \hat{N}_\omega). \quad (\text{A.1})$$

Eigenvectors and eigenvalues of matrix $\mathbf{G}_{\mathbf{u}\mathbf{u}}$, calculated for the circular frequency values reported in (A.1), are denoted by $\hat{\gamma}_r^{(h)}$ and $\hat{\theta}_r^{(h)}$, respectively. Then, by defining the sequence s_k :

$$s_k = \frac{\ln(k-1)}{\ln(N_t/2)} (\hat{N}_\omega - 1) + 1 \quad \left(k = 2, \dots, \frac{N_t}{2} + 1 \right) \quad (\text{A.2})$$

the eigenvalues $\gamma_r^{(k)}$ of matrix $\mathbf{G}_{\mathbf{u}\mathbf{u}}(\omega_k)$ are interpolated as:

$$\begin{aligned} \gamma_r^{(k)} &= (s_k - \text{round}^+(s_k)) \hat{\gamma}_r^{(\text{round}^-(s_k))} \\ &\quad + (1 - s_k + \text{round}^+(s_k)) \hat{\gamma}_r^{(\text{round}^+(s_k))} \\ &\quad \left(k = 2, \dots, \frac{N_t}{2} + 1 \right), \quad (\text{A.3}) \\ \gamma_r^{(k)} &= \gamma_r^{(N_t+2-k)} \quad \left(k = \frac{N_t}{2} + 2, \dots, N_t \right), \end{aligned}$$

where $\text{round}^+(\cdot)$ and $\text{round}^-(\cdot)$ provide the upper and lower integer rounds of the argument respectively. Finally, the eigenvectors are approximated stepwise as:

$$\theta_r^{(k)} = \begin{cases} \hat{\theta}_r^{(\text{round}(s_k))} & \left(k = 2, \dots, \frac{N_t}{2} + 1 \right), \\ \hat{\theta}_r^{(N_t+1-k)} & \left(k = \frac{N_t}{2} + 2, \dots, N_t \right), \end{cases} \quad (\text{A.4})$$

where $\text{round}(\cdot)$ returns the closest integer to the argument.

The simulation formula of an AR(p_1) filter is obtained, by assuming $q = 0$ in (5), as:

$$\mathbf{u}(t_j) = \sum_{i=1}^p \Phi_i \mathbf{u}(t_j - i\Delta t) + \mathbf{B}_0 \mathbf{a}(t_j). \quad (\text{A.5})$$

In order to calculate the coefficient matrices Φ_i and $\mathbf{B}_0$, the correlation matrix $\mathbf{R}_{\mathbf{u}\mathbf{u}}(\tau)$ must be defined as:

$$\mathbf{R}_{\mathbf{u}\mathbf{u}}(\tau) = E[\mathbf{u}(t)\mathbf{u}^T(t+\tau)]. \quad (\text{A.6})$$

Assuming the shorter notation $\mathbf{R}_{\mathbf{u}\mathbf{u}}(k\Delta t) = \mathbf{R}_{\mathbf{u}\mathbf{u}}(k)$, the following hyper-system of algebraic equations can be obtained [12]:

$$\begin{aligned} \mathbf{K} [\Phi_1 \quad \Phi_2 \quad \dots \quad \Phi_{p_1}]^T &= \mathbf{b}, \\ \mathbf{K} &= \begin{bmatrix} \mathbf{R}_{\mathbf{u}\mathbf{u}}(0) & \mathbf{R}_{\mathbf{u}\mathbf{u}}(1) & \dots & \mathbf{R}_{\mathbf{u}\mathbf{u}}(p-1) \\ \mathbf{R}_{\mathbf{u}\mathbf{u}}(-1) & \mathbf{R}_{\mathbf{u}\mathbf{u}}(0) & \dots & \mathbf{R}_{\mathbf{u}\mathbf{u}}(p-2) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{R}_{\mathbf{u}\mathbf{u}}(1-p) & \mathbf{R}_{\mathbf{u}\mathbf{u}}(2-p) & \dots & \mathbf{R}_{\mathbf{u}\mathbf{u}}(0) \end{bmatrix} \quad (\text{A.7}) \\ \mathbf{b} &= [\mathbf{R}_{\mathbf{u}\mathbf{u}}(-1) \quad \mathbf{R}_{\mathbf{u}\mathbf{u}}(-2) \quad \dots \quad \mathbf{R}_{\mathbf{u}\mathbf{u}}(-p)]^T \end{aligned}$$

Once the correlation matrices are calculated by means of inverse FFT algorithms applied to CPSD functions, the coefficient matrices Φ_i can be easily determined by solving the algebraic hyper-system (A.7). Then it is possible to calculate matrix $\mathbf{B}_0$, by post-multiplying (A.1) by $\mathbf{u}(t_j)^T$ and taking the average, namely:

$$\begin{aligned} E[\mathbf{u}(t_j)\mathbf{u}^T(t_j)] &= \sum_{i=1}^p \Phi_i E[\mathbf{u}(t_j - i\Delta t)\mathbf{u}^T(t_j)] \\ &\quad + \mathbf{B}_0 E[\mathbf{a}(t_j)\mathbf{u}^T(t_j)]. \end{aligned} \quad (\text{A.8})$$

By definition of correlation matrix, (A.8) may be rewritten in the following form:

$$\mathbf{R}_{\mathbf{u}\mathbf{u}}(0) = \sum_{i=1}^p \Phi_i \mathbf{R}_{\mathbf{u}\mathbf{u}}(i) + \mathbf{B}_0 \mathbf{R}_{\mathbf{a}\mathbf{u}}(0). \quad (\text{A.9})$$

From (A.9), it follows that the covariance structure of the process is preserved if the following condition is satisfied:

$$\mathbf{B}_0 \mathbf{R}_{\mathbf{a}\mathbf{u}}(0) = \mathbf{R}_{\mathbf{u}\mathbf{u}}(0) - \sum_{i=1}^p \Phi_i \mathbf{R}_{\mathbf{u}\mathbf{u}}(i). \quad (\text{A.10})$$

Thus, the choice of matrix $\mathbf{B}_0$ is not unique and a possible strategy is to assume $\mathbf{B}_0 = \mathbf{R}_{\mathbf{a}\mathbf{u}}(0)^T$ and to obtain it through the Cholesky decomposition.

B. Implemented Algorithms

The wind simulation algorithms implemented in this study are schematically outlined in this section, for a $2N$ -variate process. The simulated wind field is denoted by the matrix variable *field_u*. Variables definitions and adopted functions are implicit from the presented algorithms.

B.1. WAWS Algorithm

Step 0.

```

initialize wind simulation parameters
for  $r = 1 : 2N$ 
    for  $k = 1 : N_\omega$ 
        calculate  $\omega_r(k)$  as a function of  $r$  and  $k$ 
        for  $i = 1 : 2N$ 
            for  $j = i : 2N$ 
                 $G(i, j) = G_{\underline{u}\underline{u}_{ij}}(\omega_r(k))$ 
            end
        end
    end
     $\text{matrG} = G + \text{upper\_triangular\_part}(G)^T$ 
     $T = \text{Cholesky\_decomposition}(\text{matrG})^T$ 
     $\text{matrT}(:, k, r) = T(:, r)$ 
end
end
save wind simulation parameters
save  $\text{matrT}$ .

```

Step 1.

```

load wind simulation parameters
load  $\text{matrT}$ 
 $\text{field\_u} = \text{zero\_matrix}(2N, N_t)$ 
 $\text{time} = [\Delta t, 2\Delta t, \dots, N_t\Delta t]$ 
 $\text{phi} = 2\pi \cdot \text{uniformly\_distributed\_random\_numbers}$ 
     $(2N, N_\omega)$ 
for  $i = 1 : 2N$ 
    for  $r = 1 : i$ 
        for  $k = 1 : N_\omega$ 
             $B_{rk}(k) = \text{matrT}(i, k, r) \cdot \exp(i \cdot \text{phi}(r, k))$ 
        end
         $\text{add\_field\_u} = N_\omega \cdot \text{Inverse\_FFT}(B_{rk})$ 
        for  $j = 1 : N_t/N_\omega$ 
             $\text{field\_u}(i, (j-1) \cdot N_\omega + 1 : j \cdot N_\omega)$ 
             $= \text{field\_u}(i, (j-1) \cdot N_\omega + 1 : j \cdot N_\omega)$ 
             $+ \sqrt{\Delta\omega} \cdot \text{real}(\text{add\_field\_u} \cdot \exp(i \cdot r/2N$ 
             $\cdot \text{time}((j-1) \cdot N_\omega + 1 : j \cdot N_\omega) \cdot \Delta\omega))$ 
        end
    end
end
end
save  $\text{field\_u}$ .

```

B.2. POD-Based Algorithm

Step 0.

```

initialize wind simulation parameters
for  $k = 1 : \hat{N}_\omega$ 
    for  $i = 1 : 2N$ 
        for  $j = i : 2N$ 
             $G(i, j) = G_{\underline{u}\underline{u}_{ij}}(\hat{\omega}_k)$ 
        end
    end
     $\text{matrG\_hat} = G + \text{upper\_triangular\_part}(G)^T$ 
     $\text{matrPSI\_hat}(:, :, k) = \text{matrix\_eigenvectors}(\text{matrG\_hat})$ 
     $\text{matrGAMMA\_hat}(:, :, k) = \text{eigenvalues}(\text{matrG\_hat})$ 
end
save wind simulation parameters
save  $\text{matrPSI\_hat}$  and  $\text{matrGAMMA\_hat}$ 

```

Step 1.

```

load wind simulation parameters
load  $\text{matrPSI\_hat}$  and  $\text{matrGAMMA\_hat}$ 
 $\text{field\_u} = \text{zero\_matrix}(2N, N_t)$ 
 $R = 1/\sqrt{2} \cdot \text{normally\_distributed\_random\_numbers}$ 
     $(N_\omega, 2N)$ 
 $I = 1/\sqrt{2} \cdot \text{normally\_distributed\_random\_numbers}$ 
     $(N_\omega, 2N)$ 
 $\text{time} = [\Delta t, 2\Delta t, \dots, N_t\Delta t]$ 
for  $k = 1 : N_\omega$ 
    if  $k \leq N_t/2 + 1$ 
         $sk = \log(k-1)/\log(N_t/2) \cdot (\hat{N}_\omega - 1) + 1$ 
         $\text{PSI\_k} = \text{matrPSI\_hat}(:, :, \text{round}(sk))$ 
         $\text{GAMMA\_k} = (sk - \text{floor}(sk)) \cdot \text{matrGAMMA\_hat}$ 
             $(:, :, \text{ceil}(sk))$ 
             $+ (1 - sk + \text{floor}(sk)) \cdot \text{matrGAMMA\_hat}$ 
             $(:, :, \text{floor}(sk))$ 
    end
    if  $k > N_t/2 + 1$ 
         $sk2 = \log(N_\omega + 1 - k)/\log(N_t/2) \cdot (\hat{N}_\omega - 1) + 1$ 
         $\text{PSI\_k} = \text{matrPSI\_hat}(:, :, \text{round}(sk2))$ 
         $\text{GAMMA\_k} = (sk2 - \text{floor}(sk2)) \cdot \text{matrGAMMA\_hat}$ 
             $\cdot \text{matrGAMMA\_hat}(:, :, \text{ceil}(sk2))$ 
             $+ (1 - sk2 + \text{floor}(sk2)) \cdot \text{matrGAMMA\_hat}$ 
             $(:, :, \text{floor}(sk2))$ 
    end
end

```

```

add_field_u = zero_matrix(2N, 1)
for r = 1 : 2N
    add_field_u = add_field_u
    + sqrt(Delta_w * GAMMA_k(r, 1))
    * PSI_k(:, r) * (R(k, r) + sqrt(-1) * I(k, r))
end
field_u(:, k) = field_u(:, k) + add_field_u
end
for r = 1 : 2N
    field_u(r, :) = N_w * Inverse_FFT(field_u(r, :))
end
field_u = real(field_u)
save field_u

```

B.3. AR Algorithm

Step 0.

```

for i = 1 : 2N
    for j = i : 2N
        for k = N_w/2 + 1 : N_w
            matrS(i, j, k - N_w/2) = S_uu_jj(w_k)
        end
        matrR(i, j, :) = Inverse_FFT(matrS(i, j, :))
    end
end
for j = 1 : p_1 + 1
    matrR(:, :, j) = matrR(:, :, j)
    + upper_triangular_part(matrR(:, :, j))^T
end
arrange matrK = K and matrB = b from matrR
solve hyper-system (A.7) matrphi = matrK \ matrB
B0 = matrR(:, :, 1)
for i = 1 : p_1
    phi(:, :, i) = matrphi(2N(i - 1) + 1 : 2Ni, :)^T
    B0 = B0 - phi(:, :, i) * matrR(:, :, i + 1)
end
B0 = Cholesky_decomposition(B0)^T
save phi and B0

```

Step 1.

```

load wind simulation parameters
load phi and B0
a = normally_distributed_random_numbers
(2N, N_t + p_1)
field_u = zero_matrix(2N, N_t)

```

```

for j = p_1 + 1 : N_t + p_1
    for i = 1 : p_1
        field_u(:, j) = field_u(:, j)
        + phi(:, :, i) * field_u(:, j - i)
    end
    field_u(:, j) = field_u(:, j) + B0 * a(:, j)
end
field_u = field_u(:, p_1 + 1 : N_t + p_1)
save field_u.

```

References

- [1] G. Deodatis, "Simulation of ergodic multivariate stochastic processes," *Journal of Engineering Mechanics*, vol. 122, no. 8, pp. 778–787, 1996.
- [2] M. Shinozuka and G. Deodatis, "Simulation of stochastic processes and fields," *Probabilistic Engineering Mechanics*, vol. 12, no. 4, pp. 203–207, 1997.
- [3] Y. Li and A. Kareem, "Simulation of multivariate random processes: hybrid DFT and digital filtering approach," *Journal of Engineering Mechanics*, vol. 119, no. 5, pp. 1078–1098, 1993.
- [4] E. Simiu and R. Scanlan, *Wind Effects on Structures: An Introduction to Wind Engineering*, Wiley, New York, NY, USA, 1996.
- [5] A. Kareem and T. Kijewski, "Time-frequency analysis of wind effects on structures," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 90, no. 12–15, pp. 1435–1452, 2002.
- [6] M. Shinozuka and C.-M. Jan, "Digital simulation of random processes and its applications," *Journal of Sound and Vibration*, vol. 25, no. 1, pp. 111–128, 1972.
- [7] R. Rossi, M. Lazzari, and R. Vitaliani, "Wind field simulation for structural engineering purposes," *International Journal for Numerical Methods in Engineering*, vol. 61, no. 5, pp. 738–763, 2004.
- [8] Y. Li and A. Kareem, "ARMA systems in wind engineering," *Probabilistic Engineering Mechanics*, vol. 5, no. 2, pp. 50–59, 1990.
- [9] M. Di Paola, "Digital simulation of wind field velocity," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 74–76, pp. 91–109, 1998.
- [10] G. Solari and L. Carassale, "Modal transformation tools in structural dynamics and wind engineering," *Wind and Structures, An International Journal*, vol. 3, no. 4, pp. 221–241, 2000.
- [11] L. Carassale and G. Solari, "Monte Carlo simulation of wind velocity fields on complex structures," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 94, no. 5, pp. 323–339, 2006.
- [12] E. Samaras, M. Shinozuka, and A. Tsurui, "ARMA representation of random processes," *Journal of Engineering Mechanics*, vol. 111, no. 3, pp. 449–461, 1985.
- [13] M. P. Mignolet and P. D. Spanos, "MA to ARMA modeling of wind," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 36, no. 1–3, pp. 429–438, 1990.
- [14] M. Di Paola and I. Gullo, "Digital generation of multivariate wind field processes," *Probabilistic Engineering Mechanics*, vol. 16, no. 1, pp. 1–10, 2001.

- [15] M. Grigoriu, "Crossing of non-Gaussian translation processes," *Journal of Engineering Mechanics*, vol. 110, no. 4, pp. 610–620, 1984.
- [16] M. Grigoriu, "Simulation of stationary non-Gaussian translation processes," *Journal of Engineering Mechanics*, vol. 124, no. 2, pp. 121–126, 1998.
- [17] R. Popescu, G. Deodatis, and J. H. Prevost, "Simulation of homogeneous nonGaussian stochastic vector fields," *Probabilistic Engineering Mechanics*, vol. 13, no. 1, pp. 1–13, 1998.
- [18] M. Gioffrè, V. Gusella, and M. Grigoriu, "Simulation of non-Gaussian field applied to wind pressure fluctuations," *Probabilistic Engineering Mechanics*, vol. 15, no. 4, pp. 339–345, 2000.
- [19] C. Borri and L. Facchini, "Artificial generation on non-Gaussian 3D wind pressure fields on structures or bluff body surfaces," in *Proceedings of the 8th ASCE Specialty Conference on Probabilistic Mechanics and Structural Reliability*, Notre Dame, Ind, USA, 2000.
- [20] V. Gusella and A. L. Materazzi, "Non-Gaussian response of MDOF wind-exposed structures: analysis by bicorrelation function and bispectrum," *Meccanica*, vol. 33, no. 3, pp. 299–307, 1998.
- [21] V. Gusella and A. L. Materazzi, "Non-Gaussian along-wind response analysis in time and frequency domains," *Engineering Structures*, vol. 22, no. 1, pp. 49–57, 2000.
- [22] A. Kareem, "Numerical simulation of wind effects: a probabilistic perspective," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 96, no. 10–11, pp. 1472–1497, 2008.
- [23] G. Solari and G. Piccardo, "Probabilistic 3-D turbulence modeling for gust buffeting of structures," *Probabilistic Engineering Mechanics*, vol. 16, no. 1, pp. 73–86, 2001.
- [24] The Mathworks Inc, "MatLab and Symulink," Natick, Mass, USA, 2002.
- [25] G. Augusti, C. Borri, and V. Gusella, "Simulation of wind loading and response of geometrically non-linear structures with particular reference to large antennas," *Structural Safety*, vol. 8, no. 1–4, pp. 161–179, 1990.
- [26] F. Petrini, F. Giuliano, and F. Bontempi, "Comparison of time domain techniques for the evaluation of the response and the stability in long span suspension bridges," *Computers and Structures*, vol. 85, no. 11–14, pp. 1032–1048, 2007.
- [27] F. Cluni, V. Gusella, and F. Ubertini, "A parametric investigation of wind-induced cable fatigue," *Engineering Structures*, vol. 29, no. 11, pp. 3094–3105, 2007.
- [28] F. Ubertini, *Wind effects on bridges: response, stability and control*, Ph.D. thesis, University of Pavia, 2008.
- [29] F. Casciati and F. Ubertini, "Nonlinear vibration of shallow cables with semiactive tuned mass damper," *Nonlinear Dynamics*, vol. 53, no. 1–2, pp. 89–106, 2008.
- [30] L. Faravelli and F. Ubertini, "Nonlinear state observation for cable dynamics," *Journal of Vibration and Control*, vol. 15, no. 7, pp. 1049–1077, 2009.