Data Article

Dataset for random uniform distributions of 2D circles and 3D spheres

Marek Wojciechowski

Łódź University of Technology; Faculty of Civil Engineering, Architecture and Environmental Engineering, Al. Politechniki 6, Łódź 90-924, Poland

ARTICLE INFO

Article history:
Received 16 May 2022
Revised 20 May 2022
Accepted 23 May 2022
Available online 28 May 2022

Keywords:
Random distribution of circles
Random distribution of spheres
Homogenization
Representative volume element
Statistical volume element

ABSTRACT

This dataset contains random uniform distributions for a large number of 2D and 3D balls, along with the description files. It provides the possibility for fast pick up of random, but repeatable, sets of smaller samples, with the guaranteed statistical properties such as random uniform distribution of balls, the predefined expected volume ratio of balls, and also the minimum distance between them. Samples are uniquely identified by the position coordinates in the provided large kernels. The sets of samples can be used in performing numerical predictions of different types for uniform ball distributions while keeping the numerical effort at a reasonable level. Specifically, this can be useful in computational homogenization of fiber and spherical particle reinforced composites, where the provided kernels can be viewed as representative volumes and the samples as the realizations of statistical volume elements. Some secondary results, like the numbers of samples of a given size assuring the required accuracy in expected ball volume ratio representation, are also provided. Data was created by means of the pseudo-random number generator using python scripting and can be loaded and used also in other programming environments.

© 2022 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/)

DOI of original article: 10.1016/j.compstruct.2022.115718
E-mail address: mwojc@p.lodz.pl
Social media: @mrkwjc

https://doi.org/10.1016/j.dib.2022.108318
2352-3409/© 2022 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/)
Specifications Table

| Subject          | Engineering                        |
|------------------|------------------------------------|
| Specific subject area | Generation of randomly distributed balls, homogenization of fiber reinforced and spherical particle reinforced random composites. |
| Type of data     | Output binary files (.npy format)   |
| How data were acquired | Data was acquired by numerical computations with the help of python scripting. |
| Data format      | Raw                                |
| Description of data collection | The dataset contains random uniform distributions for a large number of 2D and 3D balls. The following assumptions were made for generating these distributions: ball diameter is unitary, distribution is random uniform, the expected volume ratio of balls varies from 0.025 to 0.48 for 2D case and from 0.01 to 0.3 for 3D case with the step 0.005, and distance between balls is guaranteed to be greater than a value varying from 0 to 0.05 with the step 0.005. Ball positions for all geometrical configurations were generated with pseudo-random number generator available in NumPy and spatial subpackage present in SciPy. |
| Data source location | Łódź University of Technology, Łódź, Poland, EU |
| Data accessibility | Mendeley Data, https://doi.org/10.17632/nbtp99bd76.1 |
| Related research article | M. Wojciechowski, On generalized boundary conditions for mesoscopic volumes in computational homogenization, Composite Structures, 2022, https://doi.org/10.1016/j.compstruct.2022.115718 |

Value of the Data

- Dataset is useful because it provides kernels of precomputed positions of 2D and 3D ball centers with guaranteed random uniform distribution and with assumed volume ratio and distances between them. Dataset is important because it can be used for standardized, repeatable, and verifiable computations performed on sets of samples taken from the provided kernels.
- The data can benefit researchers and engineers dealing with random materials such as fiber reinforced composites and spherical particle reinforced composites. Another use is also possible wherever the random uniform distribution of points with guaranteed distance is required.
- Data is used for fast generation of sets of samples of randomly uniformly distributed 2D and 3D balls. These samples are picked up from the provided large collections of ball centers with known statistical properties (kernels). The benefit is that the sampling procedure is fast and the results are reliable.

1. Data Description

Main data is contained in two files “kernels2D.npy” and “kernels3D.npy”, which are the arrays of centers of balls with unitary radius $r$. Data is stored in NumPy binary format. The shape of these arrays is ($n_c, n_d, N, D$), where $n_c$ is the number of considered ball volume ratios, $n_d$ - number of considered distances, $N$ - maximum number of generated centers, and $D$ - space dimension. The “kernels2D.json” and “kernels3D.json” files are the description files stored in JavaScript Object Notation format. They can be previewed in Firefox web browser, edited in any text editor and loaded as dictionary-like data structure in Python, Matlab and other. The most important entries in these files are:

- $c$ - list of considered expected volume ratios $c$ of balls (of length $n_c$);
- $d$ - list of considered additional distances $d$ between balls (of length $n_d$); the final distance is guaranteed to be not less than $d$;
• Ncd - array of numbers of ball centers generated for all \((c, d)\) pairs; number of centers is always greater than 100,000;
• nsamples - a dictionary containing data for computing numbers of samples of given size required to obtain a mean value of the ball volume ratio at the required accuracy in relation to the expected volume ratio; entries of this dictionary are:
  – \(N\) - list of considered expected numbers of balls \(N\) in the sample;
  – \(R\) - array of the corresponding radii \(R\) of the samples computed for all considered \(c\) and \(N\) pairs;
  – \(\text{sigma}\) - estimations of the standard errors in ball volume ratio representation in the set of samples of given size; computed for all \(c, d, N\) triplets;
  – \(\text{nsamples}\) - required numbers of samples for all \(c, d, N\) triplets;
  – \(\text{rmargin}\) - required relative accuracy for which \(\text{nsamples}\) were generated;
  – \(\text{confidence}\) - level of confidence for which \(\text{nsamples}\) were generated;
this entry should be considered as secondary data.

The data repository contains also “example2D” and “example3D” directories, however, they are not considered as the part of this present data article.

In order to use this dataset the number of dimensions, ball volume ratio, and the required minimum distance between balls have to be chosen first. Next, the indices \(i, j\) in \(c\) and \(d\) lists for the chosen values have to be determined, respectively. The exact number of centers generated for chosen geometric configuration is then retrieved from \(\text{Ncd}\) array and finally, the kernel is acquired from appropriate.npy file. Consider this python code snippet as an example:

```python
import numpy as np
import json

description = json.load(open("data\kernels3D.json"))
kernels3D = np.load("data\kernels3D.npy")
c = 0.35; i = 65
d = 0.025; j = 5
N = description["Ncd"]\[i\][j]
kernel = kernels3D[i, j, N]
```

An analogous script will be written for the 2D case. Sampling from the loaded kernel is then performed as follows:

```python
from scipy.spatial.distance import cdist
P = [10., 10., 10.]
r = 6.
R = 1.
dist = cdist(r/kernel, [P]).T[0]
centers = kernel[dist < R*r]
```

The `centers` variable contains positions of balls of radius \(r = 1\) which intersect the sampling ball of the radius \(R = 6\) with the center located at the point \(P = (10, 10, 10)\) in the kernel coordinate system (see Fig. 1). Many randomly (or deterministically) placed samples of a given size can be retrieved from kernels giving the sets of samples of the required statistical properties. See Fig. 2 for the examples of ball volume ratio distributions for large sets of samples of different sizes.

2. Experimental Design, Materials and Methods

2.1. General setup

The following assumptions were made for generating kernels of ball centers:
Fig. 1. Exemplary samples of balls. On the left: sample taken from “kernels2D.npy” file for $c = 0.35$, $d = 0.025$, $r = 1$, $R = 6$ and $P = (10, 10)$. On the right: sample taken from “kernels3D.npy” file for $c = 0.065$, $d = 0.025$, $r = 1$, $R = 6$ and $P = (10, 10, 10)$. Volume ratio of the balls contained inside the samples is close to the expected values of $c$ (but not identical).

Fig. 2. Examples of distributions of ball volume ratio for random sets of samples of different sizes. On the left 2D case (‘kernels2D.npy’), on the right 3D case (‘kernels3D.npy’). Mean values oscillate close to the expected $c$ values and the deviations decrease with sample size.

- diameter $r$ of balls is unitary;
- distribution of balls is random uniform;
- expected volume ratio of balls $c$ varies from 0.025 to 0.48 for 2D case and from 0.01 to 0.3 for 3D case, with the step 0.005 for both cases;
- distance between balls is guaranteed to be greater than $d$, a value varying from 0 to 0.05 with the step 0.005.

Lower bounds for the expected volume ratio of balls $c$ are dictated by practical reasons (very sparse ball distributions are rarely needed), whereas the upper bounds are determined by the physical limits rather, as reported in Zubko and Zubko [1]. The minimum distance between balls is also considered for practical reasons, i.e. $d > 0$ is often required when creating high quality finite element meshes on the samples of balls.
Computations were performed using python scripting with the help of NumPy and SciPy packages. Ball centers were sampled using the default random number generator present in the NumPy package (version 1.20.1) with the default settings – see NumPy documentation and also [2] for further reference. The seed number for this generator was left unspecified, i.e. it has been taken automatically from the machine environment at the time of performing computations. Data was generated on the server equipped with two Intel Xeon Gold 5220R CPUs.

2.2. Kernel generation details

For ball radius \( r = 1 \) and for every considered expected volume ratio of the balls \( c \) and additional distance between them \( d \), the following procedure was performed:

1. Expected number of balls is assumed as \( N = 100,000 \).
2. Radius of the spherical domain occupied by the balls is calculated as:
\[
R = r \left( \frac{N}{c} \right)^{\frac{1}{D}}.
\]

3. List of ball centers \( P_{nk} \), where \( n = 1 \ldots \tilde{N} \) and \( k = 1 \ldots D \), is initialized with \( \tilde{N} = 0 \).
4. Direction vector \( \pi_k \) of the new ball center is drawn from Gaussian distribution \( (\mu = 0, \sigma = 1) \) and then normalized; this assures that all the generated direction points are distributed in a random uniform way on the surface of the domain with unitary radius [3].
5. Length scale factor \( \rho \) for the direction vector \( \pi_k \) is established using the inverse of a cumulative distribution function (CDF) that corresponds to the surface area of the spherical domain in \( D \) dimensions, i.e. \( \rho = x^D \) is taken, where \( x \) is a number drawn uniformly from the range \([0, 1]\).
6. Final position of the ball center is computed as:
\[
p_k = (R + r) \rho \pi_k.
\]
7. Distances between the newly generated center and other centers added previously to the array \( P_{nk} \) is computed as follows:
\[
d_n = \sqrt{\sum_k (P_{nk} - p_k)^2}.
\]
8. If \( d_n > (2 + d)r \) for all \( n \), then new point \( p_k \) is appended to the array \( P_{nk} \) (so that \( \tilde{N} \) increases by 1), otherwise new point is not appended.
9. Steps 4–8 are repeated up to achieving the required ball volume ratio \( c \). Note that the balls can be placed in such a way, that they cross the boundary of the domain and only partially contribute to the overall ball volume. This is true if \( ((R + r) \rho \in [R - r, R + r] \). This possibility is allowed for assuring a truly random uniform distribution of balls inside the domain. The side consequence is that the final number of generated centers is higher, then the expected number of balls, i.e.: \( \tilde{N} \geq N \).

The arrays \( P_{nk} \) generated with the above procedure for space dimensions \( D = 2 \) and \( D = 3 \) and for all considered \( c, d \) pairs are collected into two arrays and saved in “kernels2D.npy” and “kernels3D.npy” files, respectively, along with the JSON description files. Note, that for obtaining kernels of the identical properties, but for \( r \neq 1 \), it is sufficient to scale the existing coordinates by factor \( r \).

2.3. Determination of representative number of samples

Let us assume that the set of samples with radius \( R < R \) randomly picked up from the given geometrical configuration (i.e. for given \( D, c, d \)) must be such, that the mean volume ratio of
balls for this set does not differ from the expected c value by more than the predefined error margin δc with the confidence level A. Using the central limit theorem, it is assumed that the distribution of volume ratio means computed for many sets of samples of a given size tends to the normal distribution with mean value μc = c. The standard deviation σc depends on radius R (see Fig. 2). The number of necessary samples can be then determined by the following equation:

\[ N_{\text{samples}} = \left( \frac{Z(A) \cdot \sigma_c(R)}{\delta c} \right)^2, \]  \[ (4) \]

where \( Z(A) \) is half of the confidence interval for the normal distribution function. Standard deviations \( \sigma_c(R) \) have been estimated for some chosen R values. This is done simply by generating large number of random sets of samples. This number is taken somewhat arbitrarily as \( \max\left(250 \cdot \left(\frac{V}{R}\right)^D\right) \). Computed values are placed in sigma variable of nsamples entry in description JSON files, along with the corresponding R values. Using the computed \( \sigma_c(R) \) also \( N_{\text{samples}} \) have been computed for specific choices of confidence parameters, namely:

- \( \delta = 0.05, A = 0.99 \rightarrow Z(A) = 2.579 \) for 2D case;
- \( \delta = 0.05, A = 0.95 \rightarrow Z(A) = 1.96 \) for 3D case.

The pre-computed numbers are placed in nsamples variable of nsamples entry of the description JSON files.

**Declaration of Competing Interest**

The author declare that he has no known competing financial interests or personal relationships which have, or could be perceived to have, influenced the work reported in this article.

**Data Availability**

Dataset for Random Uniform Distributions of 2D Circles and 3D Spheres (Mendeley Data).

**CRediT Author Statement**

**Marek Wojciechowski:** Conceptualization, Methodology, Software, Data curation, Writing – original draft, Visualization, Investigation, Validation.

**Acknowledgments**

Author would like to acknowledge the European Regional Development Fund for supporting this study (project RPLD.01.01.00-10-0003/18 realized at Łódź University of Technology, Poland)

**References**

[1] E.I. Zubko, Y.E. Zubko, Maximum possible densities of random sphere packing within the composite, Powder Metall. Met. Ceram. 58 (3) (2019) 133–139, doi:10.1007/s11106-019-00056-1.

[2] M. E. O’Neill, PCG, A Family of Better Random Number Generators, 2014, (http://www.pcg-random.org/index.html).

[3] G. Marsaglia, Choosing a point from the surface of a sphere, Ann. Math. Stat. 43 (2) (1972) 645–646, doi:10.1214/aoms/1177692644.