# A reduced concurrent memory access method to accelerate the computation of the lineal path function on large microstructures

Edward J. Bissaker[1] Bishnu P. Lamichhane[2]
David R. Jenkins[3]

(Received 31 January 2023; revised 15 April 2024)

**Abstract**

The Concurrent Reduced Memory Access method (CRMA) is a scalable memory-efficient Monte Carlo method for computing the lineal path function. It addresses an inherent memory bottleneck of lineal path function algorithms by utilising known properties of the two-point correlation function to reduce the number of voxels where the phase value must be evaluated. The CRMA method reduces the computation time and improves the scalability characteristics of the traditional lineal path function Monte Carlo methods. CRMA also provides additional information useful for analysing microstructures since the two-point correlation function is computed as part of the method. The CRMA method

DOI:10.21914/anziamj.v64.17973,  Published 2024-05-04, as part of the Proceedings of the 20th Biennial Computational Techniques and Applications Conference. ISSN 1445-8810. (Print two pages per sheet of paper.) 

offers an efficient, scalable and extendable solution for computing the lineal path function.

# Contents



# 1 Introduction

Quantitative analysis of microstructures is widely used in various disciplines to understand the relationship between material structure and physical properties [18]. In recent years, micro-computed tomography (CT) has become increasingly popular for studying material microstructures, as it allows for non-destructive imaging of the internal structure of a material and provides a dataset of images for quantitative analysis [2, 8]. Micro-CT captures high-resolution images of materials, and it is not uncommon for three-dimensional images to contain billions of voxels. Developing appropriate methodologies, tools, and algorithms to utilise the information collected in these large images is an active area of research [6, 7, 12].

Two microstructure descriptors commonly used to investigate microstructure formations are the lineal path function and the two-point correlation function [20]. The lineal path function gives the probability that a line segment of a pre-specified length will be contained inside a phase of interest when inserted randomly into a sample. Similarly, the two-point correlation function gives the probability that two locations at a set distance apart are in the same phase. These descriptors provide information about microstructure

features at different length scales and are often combined to give a more complete representation of a material [12]. Traditional methods for approximating the lineal path function involve Monte Carlo sampling schemes [14], whereas recent work focuses on computing the lineal path function directly from the chord length distribution function, another related microstructure descriptor [17, 16]. Evaluating the lineal path function is computationally challenging because any lineal path function algorithm needs to check the material phase value at many voxels [17, 16, 21, 7, 20].

When a slight loss in precision is acceptable, Monte Carlo formulations reduce the number of voxel phase evaluations required and provide an accelerated method to estimate the lineal path function [21]. In time-sensitive applications, such as iterative algorithms, Monte Carlo formulations are often favoured over a complete computation. The wide range of applications and the utility of the lineal path function in materials classification has made optimising its computation an ongoing area of research [7].

Images collected with micro-CT contain large amounts of data, almost always exceeding the limited sizes of current CPU caches. Therefore, data must be continuously moved from RAM to CPU caches, resulting in a memory-bound algorithm [11]. Additionally, since voxel locations often do not correspond to adjacent or proximate memory locations, many voxels require a new cache block to be read when evaluating a single line segment [9].

Measuring the lineal path function for arbitrary direction line segments results in challenging memory access patterns since values corresponding to a segment often do not lie in contiguous memory blocks. However, evaluations for arbitrary direction line segments are essential to determine angular variation in heterogenous and anisotropic materials [19]. Efforts have been made to enhance the efficiency of lineal path function computation by creating better contiguous memory patterns for chord-length-based algorithms [7]. Improving general memory access patterns is still challenging for Monte Carlo methods that consider line segments with arbitrary orientation since the values that must be accessed are unknown a priori. Furthermore, typical CPU optimisation

techniques, such as prefetching, are also difficult to utilise since the values that will be accessed as part of the Monte Carlo procedure are random. Parallelisation is also less effective when the algorithm is memory-bound since simultaneous computation is limited by the memory bus bandwidth [9]. Thus, methods to reduce the memory-bound on the lineal path function computation via Monte Carlo are beneficial and necessary for scalability and high performance of the algorithm.

In this article, we propose the Concurrent Reduced Memory Access (CRMA) method to accelerate the computation of the lineal path function by utilising properties of the two-point correlation function. This is achieved by reducing the number of voxels where the phase must be evaluated, minimising data transfer from RAM to cache. Section 2 provides details on the computation of the lineal path and two-point correlation functions on a digital microstructure and introduces the CRMA method. Section 3 compares CRMA with an original Monte Carlo method to compute the lineal path function. We demonstrate the reduction in memory reads, computation time, and scalability achieved by CRMA compared with an original Monte Carlo method. Our results show that with this reduced memory access formulation, the computation time for the lineal path function is reduced by 40%, the memory accesses are reduced by 35%, and scalability is improved when multithreaded on an 12-core CPU. Furthermore, since CRMA also concurrently computes the two-point correlation function, this combination of statistical descriptors is useful in material classification and synthetic microstructure generation, where a combination of descriptors is preferred. [12]. Finally, Section 4 provides conclusions and discusses future high-performance computing applications of CRMA.

## 2 Concurrent reduced memory access method

Micro-CT images produce a three-dimensional digital representation of the internal microstructure of a material. Micro-CT images are greyscale and stored in a 3D array $\tilde{\mathcal{A}}$ consisting of a collection of $\mathsf{N}_1$ two-dimensional

images, each with $N_2$ rows and $N_3$ columns. Typical ranges for $\{N_i\}_{i=1}^3$ are between 80 and 1000 [13]. For the remainder of this article, we consider equal dimensions $N$, hence $N = N_1 = N_2 = N_3$, and the values in $\tilde{\mathcal{A}}$ to be 8-bit integers ranging from 0 to 255. To focus our analysis on the solid material formations within the microstructure, we create a second array $\mathcal{A}$ containing two distinct phases: $\Omega_1$ (solid phase) and $\Omega_0$ (pore phase). Each value in $\mathcal{A}$ corresponds to a value in $\tilde{\mathcal{A}}$, and the phase value is determined using the Otsu thresholding method applied to the integer entries of $\tilde{\mathcal{A}}$ [15]. The appropriateness of the threshold is confirmed by visually inspecting and confirming that solid regions in $\mathcal{A}$ correspond to the non-pore regions in the original CT-image $\tilde{\mathcal{A}}$ [1]. In $\mathcal{A}$, all voxels in $\Omega_1$ are assigned the integer value 1, while voxels in $\Omega_0$ are assigned the value 0. Voxel positions in $\mathcal{A}$ are indexed by $j$, $k$ and $l$ for each axis.

Let $\mathcal{A}$ be a digital microstructure defined above and $[r_{min}, r_{max}]$ be a chosen range of line segment lengths. We introduce a vector $\mathbf{r} := \boldsymbol{e}_1 v_1 + \boldsymbol{e}_2 v_2 + \boldsymbol{e}_3 v_3$, where $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3\} \subset \mathbb{R}^3$ is a set of unit basis vectors and $\{v_1, v_2, v_3\} \subset \mathbb{R}$ are scalars such that

$$r_{min} \leqslant \|\mathbf{r}\| = \sqrt{v_1^2 + v_2^2 + v_3^2} \leqslant r_{max}\,.$$

The vector $\mathbf{r}$ can be translated such that it *originates* at a voxel $\boldsymbol{v}_O := (j_O, k_O, l_O)$ in $\mathcal{A}$ and *terminates* at voxel

$$\boldsymbol{v}_T(\boldsymbol{v}_O, \mathbf{r}) := \boldsymbol{v}_O + \mathbf{r} = ([j_O + v_1], [k_O + v_2], [l_O + v_3])\,,$$

where $[\cdot]$ denotes the standard integer rounding function. Arbitrary direction and segment lengths are introduced by varying $v_1$, $v_2$ and $v_3$. Methods to analyse the microstructure of a material must be generalisable to non-periodic microstructures, and it has been demonstrated that the assumption of periodic boundaries does not introduce a systematic bias to the statistical descriptors [5]. Suppose $\boldsymbol{v}_T = (j_T, k_T, l_T)$ is located outside the boundary, that is, $j_T$, $k_T$ or $l_T$ is greater than $N$. In that case, periodic boundary conditions are applied by taking each voxel index exceeding the boundary modulo the

corresponding dimension of $\mathcal{A}$, for example

$$\boldsymbol{\nu}_T = (j_T(\mathrm{mod}N), k_T, l_T(\mathrm{mod}N)) \quad \text{if } j_T > N \text{ and } l_T > N\,.$$

The lineal path function $L(r)$ on phase $\Omega_1$ is defined as the probability a line segment with length $r$ is wholly contained in the solid phase $\Omega_1$ when randomly inserted into a sample, and it contains coarse-level connectedness information about the microstructure [14]. The lineal path function has the properties

$$0 \leqslant L(r) \leqslant \Phi\,, \quad L(r) \to 0 \text{ as } r \to \infty\,. \tag{1}$$

where $\Phi$ is the phase volume fraction of the solid phase which decays to zero under the assumption that the material lacks long-range order [7].

Bresenham's line algorithm [3] is used to compute the voxel locations corresponding to the line segments inserted in $\mathcal{A}$. It is an efficient line drawing algorithm used in computer graphics that computes the voxel locations of the intermediate voxels of a line specified by its originating and terminating voxels, $\boldsymbol{\nu}_O$ and $\boldsymbol{\nu}_T$. Bresenham's line algorithm uses only integer arithmetic when computing the voxel positions, and the discrete line generated by the algorithm may contain a number of additional voxels not equal to the segment length $\|\mathbf{r}\|$. The *Bresenham's line evaluator* function takes an originating voxel $\boldsymbol{\nu}_O$ and terminating voxel $\boldsymbol{\nu}_T(\boldsymbol{\nu}_O, \mathbf{r})$ and determines if all the voxels corresponding to the line segment given by $\mathbf{r}$ are in the solid phase:

$$\mathcal{BLE}(\boldsymbol{\nu}_O, \boldsymbol{\nu}_T) = \begin{cases} 1 & \text{Voxels between } \boldsymbol{\nu}_O \text{ and } \boldsymbol{\nu}_T(\boldsymbol{\nu}_O, \mathbf{r}) \text{ are in } \Omega_1, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Let $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_M$ be $M$ vectors in $\mathcal{A}$ with the same length $r = \|\mathbf{r}\|$ and a range of orientations. The lineal path function $L(r)$ is computed by counting the number of segments in the solid phase. Thus, for a sufficiently large random sample of $S$ initial positions uniformly distributed throughout $\mathcal{A}$ we approximate

$$L(r) \approx \frac{1}{S} \sum_{s=1}^{S} \left( \frac{1}{M} \sum_{m=1}^{M} \mathcal{BLE}(\boldsymbol{\nu}_O^s, \boldsymbol{\nu}_T(\boldsymbol{\nu}_O^s, \mathbf{r}_m)) \right). \tag{3}$$

Since the entries of $\mathcal{A}$ are in $\{0, 1\}$, evaluating $\mathcal{BLE}(\boldsymbol{\nu}_O^s, \boldsymbol{\nu}_T(\boldsymbol{\nu}_O^s, \mathbf{r}_m))$ is completed by taking the product of the line segment voxel values. Thus, (3) gives the portion of segments in the solid phase. The above process is repeated for various lengths $r \in [r_{min}, r_{max}]$, which provides a lineal path function distribution.

The previous sampling technique corresponds to the original formulation for the Monte Carlo lineal path function [20], referred to as the *Original MC* method for the remainder of this article, and is used for the comparisons in Section 3.

The two-point correlation function $s_2(r)$ is a microstructure descriptor that gives the probability that two voxels $\boldsymbol{\nu}_O$ and $\boldsymbol{\nu}_T$ at length $r$ apart are contained in the solid phase. The two-point correlation function has the properties

$$\Phi^2 \leqslant s_2(r) \leqslant \Phi\,, \quad s_2(r) \to \Phi^2 \text{ as } r \to \infty\,, \tag{4}$$

for $\Phi$ the phase volume fraction of the solid phase and where the infinite $r$ limit assumes the material lacks long-range order. The computation of the two-point correlation function is closely related to the lineal path function (3), except that the two-point correlation function requires only the phase values at the originating and terminating voxel to be evaluated, and not any intermediate voxel locations. Therefore, with the required adjustment to (3), the two-point correlation function is approximated as

$$s_2(r) \approx \frac{1}{S}\sum_{s=1}^{S}\left(\frac{1}{M}\sum_{m=1}^{M}\mathcal{A}(\boldsymbol{\nu}_O^s)\mathcal{A}(\boldsymbol{\nu}_T(\boldsymbol{\nu}_O^s, \mathbf{r}_m))\right), \tag{5}$$

for $\mathcal{A} \in \{0, 1\}$, and where these two phase values are evaluated at the voxels $\boldsymbol{\nu}_O^s$ and $\boldsymbol{\nu}_T$ which are separated by vector $\mathbf{r}_m$ with length $r$.

Since (5) and (3) both require a set of vectors and originating voxels, the CRMA method is constructed by combining (3) and (5), whereby the local correlation for a given line segment is computed first, and only if $\mathcal{A}(\boldsymbol{\nu}_O)\mathcal{A}(\boldsymbol{\nu}_T) \neq 0$, then the corresponding lineal path evaluation $\mathcal{BLE}(\boldsymbol{\nu}_O, \boldsymbol{\nu}_T)$ is commenced.

Since $\Phi^2 \leqslant s_2(r) \leqslant \Phi$, the CRMA method has a probability between $(1-\Phi)$ and $(1-\Phi^2)$, increasing with segment length $r$, of only needing to test the end voxels of a line segment, and not evaluate the intermediate voxels. This probability is upper bounded by the long-range values of $s_2(r)$ and is always defined and non-zero for porous materials. Therefore, the probability of requiring only two memory accesses is $(1-\Phi^2)$, even when $r$ grows large, making the CRMA method increasingly efficient for computing longer-length line segments. For some porous materials, $\Phi$ can exceed 60%. Hence, the probability $(1-\Phi^2)$ of requiring only two memory accesses is not necessarily small for large $r$ [10].

Memory reads from main memory may require an order of magnitude more CPU clock cycles when compared to completing a single floating point operation. Therefore, reducing the number of memory accesses is an effective way to accelerate the computation of the lineal path function, as voxels that need to be tested for most vector orientations are not located in adjacent or proximate memory locations. Therefore, a new cache block must be read for many values, resulting in an inefficient, congested data transfer from RAM to the CPU cache [9].

Very efficient methods to compute the two-point correlation function using Fast Fourier transforms (FFT) [4] are often used in combination with lineal path function algorithms [7]. However, FFT methods do not provide the local correlation information for a given line segment that allows the quicker computation of the lineal path function.

To initialise the CRMA method, two integer storage arrays, TPC and LPF, with $M$ entries each, are generated to store the two-point correlation and lineal path function for each length. Vectors $\mathbf{r}_1, \ldots, \mathbf{r}_M$ are each comprised of 14 directions corresponding to the vertices and faces of a cube, and are initialised and used throughout the computation for the various samples. During each iteration, a new random originating voxel $\boldsymbol{\nu}_O$ is selected for each of the $M$ vectors. Subsequently, if the local two-point correlation condition $\mathcal{A}(\boldsymbol{\nu}_O)\mathcal{A}(\boldsymbol{\nu}_T) \neq 0$ is satisfied, the $m$th value of the corresponding TPC array

**Algorithm 1** CRMA

**Require:** $\mathcal{A}$, $\mathbf{r}_1, \ldots, \mathbf{r}_M$, $S$.
**Initialise** LPF, TPC
**while** $s < S$ **do**
  **for each** $\mathbf{r}_m, m = 1, \ldots, M$ **do**
    Randomly select voxel location $\boldsymbol{\nu}_O^s$ in $\mathcal{A}$.
    Compute $\boldsymbol{\nu}_T^s(\boldsymbol{\nu}_O^s, \mathbf{r}_m)$.
    Apply boundary conditions: if $\boldsymbol{\nu}_T^s \notin \mathcal{A}$, then
      $\boldsymbol{\nu}_T^s(j, k, l) \rightarrow (j_T(\mathrm{mod} N), k_T(\mathrm{mod} N), l_T(\mathrm{mod} N))$.
    **if** $\boldsymbol{\nu}_O^s$ and $\boldsymbol{\nu}_T^s$ in solid phase **then**
      TPC$[m]$ += 1
      **if** $\mathcal{BLE}(\boldsymbol{\nu}_O^s, \boldsymbol{\nu}_T^s) > 0$ **then**
        LPF$[m]$ += 1
      **end if**
    **end if**
  **end for**
  $s$ += 1
**end while**
**for each** $m$ in $\{1, \ldots, M\}$ **do**
  TPC$[m]/S$ and LPF$[m]/S$
**end for**
Average TPC and LPF over $M$.
**return** LPF and TPC

is incremented. Similarly, if $\mathcal{A}(\boldsymbol{\nu}_O)\mathcal{A}(\boldsymbol{\nu}_T) \neq 0$ and $\mathcal{BLE}(\boldsymbol{\nu}_O, \boldsymbol{\nu}_T) > 0$, the $m$th value of the LPF array is incremented. Upon completion of the iterations, the $M$ values in TPC and LPF are divided by $S$. Finally, the various orientations are averaged over $M$ to compute $L(r)$. The complete CRMA algorithm is outlined in Algorithm 1. To compute the complete lineal path function distribution $L(r)_{[r_{min}, r_{max}]}$, a range of line segment lengths $r \in [r_{min}, r_{max}]$ are taken to be monotonically increasing integer values where $r_{min} \geqslant 0$ and $r_{max} < N$, for $N$ the dimension of $\mathcal{A}$, and Algorithm 1 is repeated for each length.

**Algorithm 2** Original MC

**Require:** $\mathcal{A}$, $\mathbf{r}_1, \ldots, \mathbf{r}_M$, $S$

**Initialise** LPF
**while** $s < S$ **do**
  **for each** $\mathbf{r}_m, m = 1, \ldots, M$ **do**
    randomly select voxel location $\boldsymbol{\nu}_O^s$ in $\mathcal{A}$
    compute $\boldsymbol{\nu}_T^s(\boldsymbol{\nu}_O^s, \mathbf{r}_m)$
    apply boundary conditions: If $\boldsymbol{\nu}_T^s \notin \mathcal{A}$, then
      $\boldsymbol{\nu}_T^s(j, k, l) \rightarrow (j_T(\mathrm{mod} N), k_T(\mathrm{mod} N), l_T(\mathrm{mod} N))$
    **if** $\mathcal{BLE}(\boldsymbol{\nu}_O^s, \boldsymbol{\nu}_T^s) > 0$ **then**
      LPF$[m]$ += 1.
    **end if**
  **end for**
  $s$ += 1
**end while**
**for** $m$ in $\{1, \ldots, M\}$ **do**
  LPF$[m]/S$
**end for**
Average LPF over $M$.
**return** LPF

The Original MC method (outlined in Algorithm 2) is used for comparison with CRMA and uses the same orientations $M$ and number of samples $S$. The critical difference is that segments are evaluated by checking $\mathcal{BLE}(\boldsymbol{\nu}_O, \boldsymbol{\nu}_T) > 0$, without the local correlation computation. When $\mathcal{BLE}(\boldsymbol{\nu}_O, \boldsymbol{\nu}_T)$ is evaluated, if at any voxel the phase is not solid, then the evaluation is concluded. To compute the complete lineal path function distribution, Algorithm 2 is repeated for each length. Every effort has been made to optimise the efficiency of the Original MC method.

# 3 Results

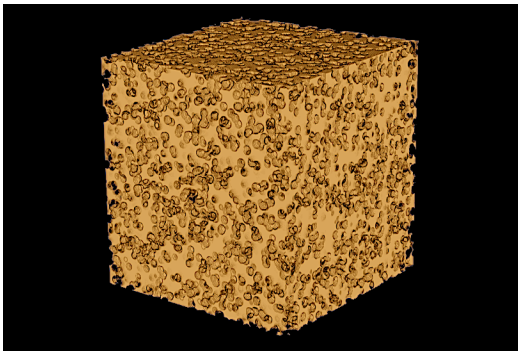
Validation of the solution for both the CRMA method and the Original MC method is performed using a test microstructure consisting of overlapping 3-dimensional spherical porous inclusions with a fixed radius $R$ (Figure 1(a)) [19]. The lineal path function for this test microstructure (Figure 1(b,c)) has a known analytic formula for phase $\Omega_1$ [14] given by

$$L^{(1)}(r) = \Phi^{1+\frac{3}{4}\frac{r}{R}}, \tag{6}$$

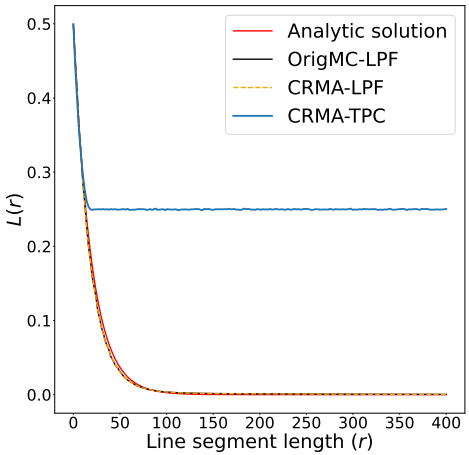
where $r$ is the line segment length, $\Phi$ the phase volume fraction, and $R$ the radius of the inclusions [14]. For the test microstructure, we consider line segments with even integer lengths ranging from $0, 2, \ldots, 400$, $\Phi = 0.5$, and $R = 10$.

There is good agreement between the lineal path function computed using CRMA and Original MC (Figure 1(b,c)), as well as with the theoretical value for the test microstructure given by (6). At line segment length $r > 100$, an increased error exists between the two Monte Carlo methods and the theoretical value. However, we see from Figure 1(b) that values of $L(r)$ for $r > 100$ are very close to zero, and the exponentially small values given by (6) are not captured in this region with $10^7$ samples. The Monte Carlo simulations were completed using $10^7$ line segments of each length $r$, and while the smoothness of the CRMA and Original MC lineal path functions indicate some initial method convergence, no convergence criteria were used in this assessment.

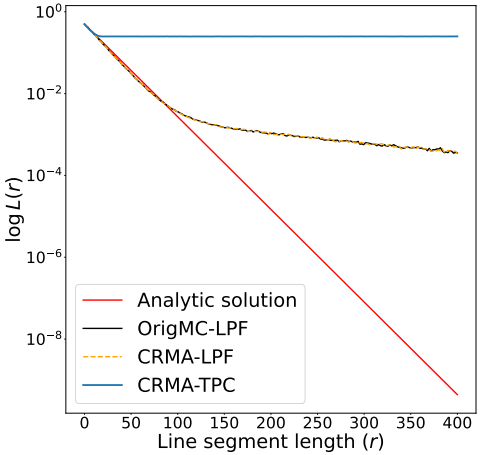
Table 1 provides the raw computation time and mean squared error (MSE) versus the analytic solution for both the CRMA and Original MC simulations shown in Figure 1(b,c). The overall reduction in computation time from the CRMA method is approximately 40%. Due to the inherent randomness associated with both procedures, a degree of error between the two methods is expected, and the magnitude of this disparity underscores a fundamental agreement between the two methods. In order to assess the impact of memory access patterns on the computational efficiency of the CRMA method,



(a)




(b)




(c)

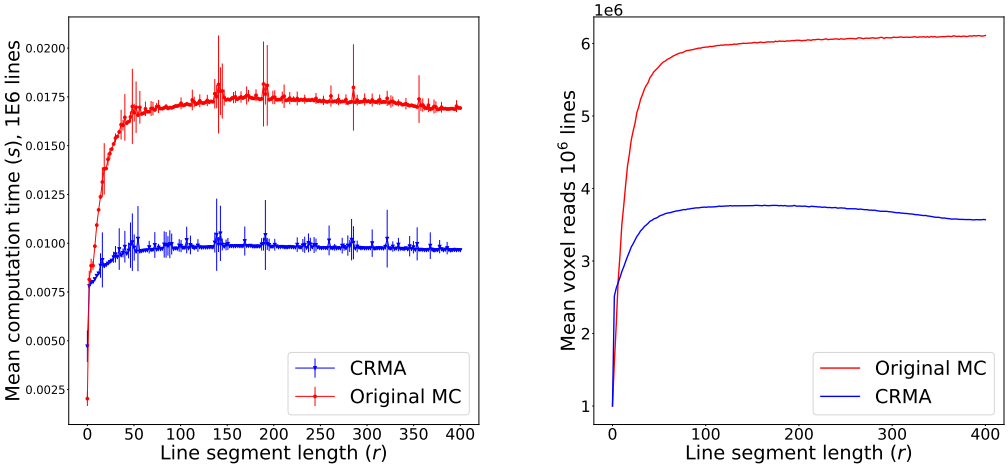
Figure 1: (a) $500^3$ voxel test microstructure. (b) Comparison of the CRMA method, the Original MC method and the analytic lineal path function result (6). (c) Logarithmic scale comparison of the two methods demonstrates sound agreement for all $r$ plotted. Variation from the analytic solution occurs for segments with length $r > 100$. The two-point correlation function produced during CRMA computation is included as CRMA-TPC in (b) and (c).

Table 1: Comparison of overall computation time and mean squared error (MSE) for $10^7$ line segments at length $r = [0, 2, \ldots, 400]$ of the CRMA and Original MC methods when compared to the known analytic solution (6) for the test microstructure in Figure 1(a).

| method | compute time (s) | MSE vs analytic (6) |
|---|---|---|
| CRMA | 21 | $1.42 \times 10^{-5}$ |
| Orig MC | 35 | $1.40 \times 10^{-5}$ |

the average time required to evaluate $10^6$ iterations was computed across varying line segment lengths $r = 0, 2, \ldots, 400$, based on ten independent trials. Figure 2(a) illustrates that for line segments with $r < 12$, the CRMA method exhibits a slight increase in mean computation time relative to the Original MC, due to additional computation and memory writing demands. Conversely, for all $r \geqslant 12$, a discernible reduction in mean computation time is evident for the CRMA method. To link the memory reads to the mean computation time, minor modifications were made to the code to count the total number of voxels accessed in $\mathcal{A}$ over ten runs of $10^6$ iterations (Figure 2(b)). The CRMA method achieves an overarching reduction of approximately 35% in mean excess memory accesses (exceeding one per segment) across all line segment lengths $r$. This reduction in mean excess memory accesses (Figure 2(b)) corresponds with the observed reduction in computation time (Figure 2(a)), thereby attributing the enhancement in computation time to mitigated memory constraints on the algorithm.

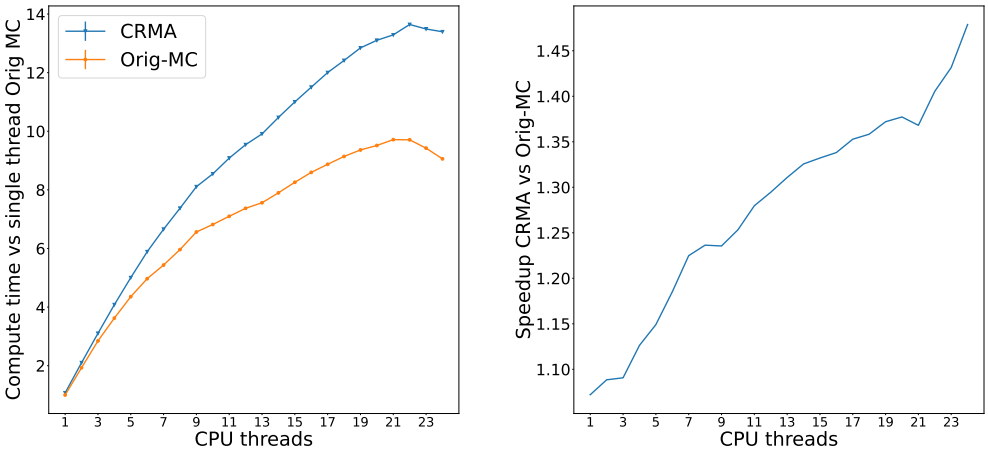
Both the CRMA and Original MC methods are parallelised identically by assigning each $\mathbf{r}_1, \ldots, \mathbf{r}_M$ to a single thread, where each thread manages multiple segments. Load balancing is implemented so each thread processes line segments of various lengths. This formulation is advantageous as it reduces potential thread write conflicts and facilitates scalability across larger systems. Figure 3 demonstrates the scalability of the two methods across 24 threads on a single 12-core CPU. To test the scalability of the methods $10^6$ line segments at random lengths and orientations are tested throughout the sample, and the




(a) (b)

Figure 2: (a) Mean computation time for $10^6$ lines with various lengths r (CPU: Ryzen 5900× 3.7 GHz, 12 core, 24 threads) averaged over ten independent runs on the test microstructure. Error bars represent the standard deviation in timing across different runs. (b) Mean memory accesses required over ten runs of $10^6$ segments at distances ranging from 0 to 400 voxels.

mean compute time utilising a fixed number of threads is averaged across ten independent trials. The Original MC method exhibits a greater than 8-fold reduction in computation time when executed in parallel on 24 threads on a single 12-core CPU (Figure 3(a)), and the CRMA method achieves more than 12-fold improvement over a single-threaded Original MC method. Since the CRMA method reduces memory reads and transfer requirements, it achieves additional speedup compared to the Original MC method when scaled on the same hardware (Figure 3(b)), with this speedup increasing with the number of threads. The total improved speedup attained through parallelisation of the CRMA method is increased by 40% compared to the Original MC method. There is a reduction in scaling performance for both methods at high $(23+)$




(a) (b)

Figure 3: (a) Scalability of CRMA and Original MC on Processor: Ryzen 5900× 3.7 GHz, 12 core, 24 threads (b) Relative improvement per thread of speedup of CRMA compared to the Original MC method. The test involves the computation of $10^6$ line segments across various numbers of CPU threads.

thread counts, likely the result of limitations on the small size of the CPU cache used for testing (64 MB L3, 6 MB L2).

# 4 Discussion and conclusions

The CRMA method achieves a reduced computation time compared to the Original MC formulation by increasing the likelihood of line segment rejection with only two memory reads and by reducing the total memory reads necessary to compute the lineal path function. Additionally, computing the two-point correlation function without additional computational overhead can be advantageous in specific applications [7]. The algorithm exhibits

linear scalability on a single hyper-threaded CPU and demonstrates improved scalability compared to the Original MC formulation. Furthermore, the CRMA algorithm can be extended to run on a multi-node high-performance computing system, achieving a second level of linear scaling across Monte Carlo sample iterations, thereby providing sufficient computational throughput for very large microstructures. CRMA offers robust algorithm acceleration by leveraging known properties of the two-point correlation function to reduce the required memory accesses when evaluating the lineal path function.

**Acknowledgements** The authors acknowledge support for this research through a BHP PhD Research Scholarship.

# References

[1] A. A. Agra, A. Nicolodi, B. D. Flores, I. V. Flores, G. L. R. da Silva, A. C. F. Vilela, and E. Osório. "Automated procedure for coke microstructural characterization in imagej software aiming industrial application". In: *Fuel* 304, 121374 (2021). DOI: 10.1016/j.fuel.2021.121374 (cit. on p. C182).

[2] J. Baruchel, P. Bleuet, A. Bravin, P. Coan, E. Lima, A. Madsen, W. Ludwig, P. Pernot, and J. Susini. "Advances in synchrotron hard X-ray based imaging". In: *Comptes Rendus Physique* 9.5-6 (2008), pp. 624–641. DOI: 10.1016/j.crhy.2007.08.003 (cit. on p. C179).

[3] J. E. Bresenham. "Algorithm for computer control of a digital plotter". In: *IBM Sys. J.* 4.1 (1965), pp. 25–30. DOI: 10.1147/sj.41.0025 (cit. on p. C183).

[4] D. T. Fullwood, S. R. Kalidindi, S. R. Niezgoda, A. Fast, and N. Hampson. "Gradient-based microstructure reconstructions from distributions using fast Fourier transforms". In: *Mat. Sci. Eng.: A* 494.1-2 (2008), pp. 68–72. DOI: 10.1016/j.msea.2007.10.087 (cit. on p. C185).

[5] J. Gajdošík, J. Zeman, and M. Šejnoha. "Qualitative analysis of fiber composite microstructure: Influence of boundary conditions". In: *Prob. Eng. Mech.* 21.4 (2006), pp. 317–329. DOI: 10.1016/j.probengmech.2005.11.006 (cit. on p. C182).

[6] E. Y. Guo, N. Chawla, T. Jing, S. Torquato, and Y. Jiao. "Accurate modeling and reconstruction of three-dimensional percolating filamentary microstructures from two-dimensional micrographs via dilation-erosion method". In: *Mat. Character.* 89 (2014), pp. 33–42. DOI: 10.1016/j.matchar.2013.12.011 (cit. on p. C179).

[7] J. Havelka, A. Kučerová, and J. Sỳkora. "Compression and reconstruction of random microstructures using accelerated lineal path function". In: *Comput. Mat. Sci.* 122 (2016), pp. 102–117. DOI: 10.1016/j.commatsci.2016.04.044 (cit. on pp. C179, C180, C183, C185, C192).

[8] J. H. Kinney and M. C. Nichols. "X-ray tomographic microscopy (XTM) using synchrotron radiation". In: *Ann. Rev. Mat. Sci.* 22.1 (1992), pp. 121–152. DOI: 10.1146/annurev.ms.22.080192.001005 (cit. on p. C179).

[9] D. Kirk and W.-m. W. Hwu. *Programming massively parallel processors: A hands-on approach*. Morgan Kaufmann, 2016. URL: https://shop.elsevier.com/books/programming-massively-parallel-processors/kirk/978-0-12-811986-0 (cit. on pp. C180, C181, C185).

[10] J. Kováčik. "Correlation between Young's modulus and porosity in porous materials". In: *J. Matt. Sci. Lett.* 18.13 (1999), pp. 1007–1010. DOI: 10.1023/A:1006669914946 (cit. on p. C185).

[11] J. Kukunas. *Power and performance: Software analysis and optimization*. Morgan Kaufmann, 2015. URL: https://www.sciencedirect.com/book/9780128007266/power-and-performance (cit. on p. C180).

[12] D. S. Li, M. A. Tschopp, M. Khaleel, and X. Sun. "Comparison of reconstructed spatial microstructure images using different statistical descriptors". In: *Comput. Mat. Sci.* 51.1 (2012), pp. 437–444. DOI: 10.1016/j.commatsci.2011.07.056 (cit. on pp. C179, C180, C181).

[13] H. Lomas, D. R. Jenkins, M. R. Mahoney, R. Pearce, R. Roest, K. Steel, and S. Mayo. "Examining mechanisms of metallurgical coke fracture using micro-CT imaging and analysis". In: *Fuel Process. Tech.* 155 (2017), pp. 183–190. DOI: 10.1016/j.fuproc.2016.05.039 (cit. on p. C182).

[14] B. Lu and S. Torquato. "Lineal-path function for random heterogeneous materials". In: *Phys. Rev. A* 45.2 (1992), pp. 922–929. DOI: 10.1103/PhysRevA.45.922 (cit. on pp. C180, C183, C188).

[15] N. Otsu. "A threshold selection method from gray-level histograms". In: *IEEE Trans. Sys., Man. Cyber.* 9.1 (1979), pp. 62–66. DOI: 10.1109/TSMC.1979.4310076 (cit. on p. C182).

[16] H. Singh, A. M. Gokhale, S. I. Lieberman, and S. Tamirisakandala. "Image based computations of lineal path probability distributions for microstructure representation". In: *Mat. Sci. Eng.: A* 474.1-2 (2008), pp. 104–111. DOI: 10.1016/j.msea.2007.03.099 (cit. on p. C180).

[17] M. S. Talukdar, O. Torsaeter, and M. A. Ioannidis. "Stochastic reconstruction of particulate media from two-dimensional images". In: *J. Colloid Interface Sci.* 248.2 (2002), pp. 419–428. DOI: 10.1006/jcis.2001.8064 (cit. on p. C180).

[18] S. Torquato. "Microstructure characterization and bulk properties of disordered two-phase media". In: *J. Stat. Phys.* 45.5 (1986), pp. 843–873. DOI: 10.1007/BF01020577 (cit. on p. C179).

[19] D. M. Turner, S. R. Niezgoda, and S. R. Kalidindi. "Efficient computation of the angularly resolved chord length distributions and lineal path functions in large microstructure datasets". In: *Mod. Sim. Mat. Sci. Eng.* 24.7, 075002 (2016). DOI: 10.1088/0965-0393/24/7/075002 (cit. on pp. C180, C188).

[20] C. L. Y. Yeong and S. Torquato. "Reconstructing random media". In: *Phys. Rev. E* 57.1 (1998), pp. 495–506. DOI: 10.1103/PhysRevE.57.495 (cit. on pp. C179, C180, C184).

[21] J. Zeman. *Analysis of composite materials with random microstructure*. Czech Technical University, Faculty of Civil Engineering, 2003. URL: https://katalog.cbvk.cz/arl-cbvk/en/detail-cbvk_us_cat-0288377-Analysis-of-composite-materials-with-random-microstructure/ (cit. on p. C180).

## Author addresses

1. **Edward J. Bissaker**, School of Information and Physical Sciences, University of Newcastle, New South Wales, AUSTRALIA.
mailto:edward.bissaker@newcastle.edu.au
orcid:0000-0002-1608-286X

2. **Bishnu P. Lamichhane**, School of Information and Physical Sciences, University of Newcastle, New South Wales, AUSTRALIA.
mailto:bishnu.lamichhane@newcastle.edu.au
orcid:0000-0002-9184-8941

3. **David R. Jenkins**, School of Engineering, University of Newcastle, New South Wales, AUSTRALIA.
mailto:david.jenkins@newcastle.edu.au
orcid:0000-0001-9660-0585