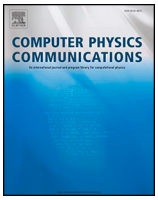
Computer Physics Communications 319 (2026) 109938

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc



Computational Physics

Calculating the demagnetisation factors and their volume distribution within (a) assemblies of discrete magnetic elements and (b) solid magnetic samples of any given shape: A material-independent and multi-scalar polar model approach

Steven M. McCann *, Tim Mercer

Jeremiah Horrocks Institute for Mathematics, Physics and Astronomy, University of Lancashire, Preston, PR1 2HE, United Kingdom

ARTICLE INFO

Edited by Prof. Andrew Hazel.

Keywords:
Demagnetisation
Magnetometric
Distribution
MATLAB
Cuboid
Cylinder
Ellipsoid

ABSTRACT

Measuring the magnetic characteristics of a magnetic sample, it is critical to evaluate the self-demagnetisation field, because it reduces the effective magnetic field experienced by the sample. The demagnetisation factor depends on the shape and nature of the sample, whether it is a solid, ordered assembly of magnetic elements, or randomly packed magnetic powder in a containing vessel. Literature provides limited information on the demagnetisation factor of packed powders, typically for a restricted number of container shapes. This paper introduces algorithms based on a polar model written in MATLAB 2022b, which calculates not only the average demagnetisation factor but also the entire distribution of demagnetisation factors for the constituent particles and, by extension, to any assembly of magnetic elements within a given volume. Furthermore, this study explains how to enhance the efficiency of these algorithms, reduce runtime, and apply them to any container shape.

The validity of the algorithms was assessed by calculating the data for three common container shapes described in literature over a range of aspect ratios: cuboids, ellipsoids, and cylinders. The calculated mean demagnetisation factors matched those found in the literature, typically within 0.05 %, 0.1 %, and 1 %, respectively, for these shapes, demonstrating that the algorithms could be extrapolated to calculate demagnetisation data for any container shape; by extension, the magnetometric demagnetisation factor (zero susceptibility) for any solid shape, a hitherto unattainable parameter.

As the method reduces to calculations based on geometry alone, it is material-independent and can be applied to any macro-, meso-, or microscale of interest.

1. Introduction

1.1. General concepts and motivation

If a solid sample of a magnetic material is magnetised by an applied field, the sample generates its own magnetic field that acts in the opposite direction to the magnetisation [1]. The demagnetisation field is influenced by the shape of the sample. For ellipsoids, the magnetisation is uniform, which leads to a uniform demagnetisation field. In other commonly encountered shapes, such as cylinders and rectangular prisms, the magnetisation is not uniform; in these cases, the demagnetisation field is averaged. The demagnetisation field H_d is proportional to the magnetisation M, which creates it:

$$H_d = -N_d M, \tag{1}$$

where N_d is the dimensionless demagnetisation factor. For ellipsoids, the demagnetisation factor depends on the ratios of the semi-major axes; for other shapes, both the geometry of the sample and the susceptibility of the material need to be considered. The sum of the demagnetisation factors of the sample taken in the three orthogonal axes N_x, N_y, *and* N_z is equal to one. The demagnetisation factors are either fluxmetric, with the averaged magnetisation and demagnetising field taken at the mid-point plane of the sample, or magnetometric, with the averaged magnetisation and demagnetising field taken over the entire sample. The use of demagnetisation fields and factors to correct magnetic measurements is important, as stated by Bahl [2] in a critical review of popular expressions and datasets for demagnetising factors for common shapes,

* Corresponding author.
E-mail address: SMMcCann@lancashire.ac.uk (S.M. McCann).

https://doi.org/10.1016/j.cpc.2025.109938
Received 5 March 2025; Received in revised form 9 October 2025; Accepted 2 November 2025
Available online 4 November 2025

comparing them to numerical techniques.

If the magnetisation is distributed within an assembly of magnetic elements, the overall structure can no longer be classified as solid, and Eq. (1) no longer applies. Instead, a demagnetisation factor must be applied to the overarching shape within which the elements are contained, such as a cuboid, ellipsoid, or cylinder, and the space between the elements considered.

There are numerous applications of this type, i.e. consisting of magnetic nanostructures and assemblies of nanoparticles. In medicine, nanoparticles are used both analytically, such as in biosensors [3] and contrast agents [4], and therapeutically, such as in drug delivery systems [5], artificial antibodies [6], and magnetic hyperthermia cancer treatments [7]. They are found in data storage devices, including upcoming new technologies utilising skyrmionics [8]; magnetic refrigeration [9], as a means of improving energy efficiency and reducing the environmental impact of conventional refrigeration systems; in catalysis as they provide a large surface area, they can be uniformly dispersed and easier to recover [10].

Although there are some examples in the literature that are unique to a specific solid shape of the sample, such as ellipsoids [11], cuboids [12], and cylinders [13], these are not applicable to real assemblies of discrete elements and cannot be applied to any general overarching shape of those assemblies.

There is a well-established methodology that is able to account for discrete elements [14,15]. By considering the volume packing fraction f, and the magnetometric demagnetisation factor D_z, (at a susceptibility of zero) of the sample if it consisted of a complete solid in the shape of the container, the overall demagnetisation factor in a given direction is now given by:

$$N_z = 1/3 + f(D_z - 1/3), \tag{2}$$

which reduces to $N_z = D_z$ for the $f = 1$ of a solid shape. However, this is limited to the elements being spherical particles and also that D_z must be known, limiting it to the common shapes analysed in the literature as described earlier.

Deviations from perfectly spherical particles have been studied by Bjork and Zhou [16], demonstrating that particle aspect ratio impacted particle orientations when packed, which in turn influences the overall demagnetisation factor. This, and other recent studies [17] have used commercially available finite element software, such as COMSOL Multiphysics® [18], to obtain demagnetisation information that is material-dependent, e.g. using its relative permeability. Other limitations using this approach includes a limit to the scale, e.g not applicable to nanoparticles.

The model presented in this study is not limited by the shape of the magnetic elements within the assembly, the material of those elements, or the need to find or calculate the respective D_z factor for a given overall shape. It has the further advantage of being applicable to any macro-, meso-, or microscale of interest and for these reasons should be of great interest to a wide range of both experimentalists and modellers alike.

1.2. Model concept and development

The key to the model is the consideration of the magnetic poles on the surface of fully magnetised particles arranged in a simple cubic lattice. The advantage of our approach is that the equations reduce to those of the geometry alone.

In this case, the particles were restricted to being spherical to test our results with those given by Eq. (2) using published D_z values from the literature. The model calculates the demagnetisation factors of each constituent particle so that the full spatial distributions of the particle demagnetisation factors can be obtained; much more information than can be extracted from Eq. (2) which only evaluates the mean demagnetisation factor. The model operates in two steps. The first is to perform a surface integral over a particle at a given position in the lattice to calculate its contribution to the demagnetisation field at the origin of the lattice; this is repeated for each lattice position. The second step involved performing a series of summations of contributions.

The first iteration of the model was used to support new experimental techniques for measuring the overall average demagnetisation factor of packed powder samples, which were glass cuboid containers packed with magnetite nanoparticles [19]. To ensure confidence in the model, the average demagnetisation factor of the sample was determined using both the model and Eq. (2) (with f measured during the packing of the sample, and D_z calculated using the equation derived by Aharoni for the demagnetisation factors of cuboids [12]), and there was perfect agreement between them.

In this paper, several algorithms are described that address a number of limitations of the first iteration, and these developments are all based on how the contributions to the demagnetisation field by each constituent particle are summated. The first shows how the run time of the model can be significantly reduced for cuboid samples, from several days for large samples in the first iteration to less than an hour. Cuboid-containing vessels are commonly used in our laboratory, and their shape is the easiest to model. Further motivation came from comments made regarding the publication of the first iteration of the model over the perceived high computational resources required to run such models (it was shown the model will run on a standard desktop PC). The second is a modification that allows packed samples in any container shape to be studied. To obtain confidence in this algorithm, testing was performed using ellipsoid container shapes to facilitate comparisons with Eq. (2), because the D_z values for ellipsoids are well documented [11]. The importance of this algorithm extends beyond that of powdered samples. By providing the demagnetisation factor of any assembly of discrete magnetic elements within any container shape, Eq. (2) can be used to determine the respective D_z value; therefore, it is possible to determine the demagnetisation factor of *any solid shape*, which is limited to magnetometric values at zero susceptibility. The third modification shows how any sample consisting of a uniform cross-section can be studied (e.g. prisms and cylinders). This can utilise the speed efficiencies of cuboids that are not available for any general shape. For confidence, comparisons can be made using Eq. (2), with model testing performed on cylinders, providing established values for D_Z [13].

The algorithms were written in MATLAB 2022b. The algorithms were tested using a desktop computer with Windows 11. The PC had an Intel i5-10400F CPU with a base speed of 2.90 GHz, six cores, and twelve logical processors. The timing performance of each script was determined using MATLAB profiler. The algorithm scripts and the performance test data generated by the profiler were recorded and stored in the GitHub repository:
https://github.com/physicssteve/Demagnetisation-Factor.git.

2. Method

2.1. The premise; an adaptation

This model uses an adaptation of the work of Bissell and Cookson et al. [20,21], who studied the demagnetisation factors of magnetic particulate recording media. The demagnetising factor of a single isolated particle, approximated as a prolate spheroid, is determined by examining how the magnetic poles on its surface produce a demagnetising magnetic field at its centre, using the coordinate system and magnetic state, as shown schematically in Fig. 1.

Our model simplifies this further by considering a spherical particle, that is, where $a = b$, modelled as a unit sphere centred at the origin, with points on its surface defined by the coordinates (o, p, q). The particle is fully magnetised in the z-axis direction so that the poles on its surface will cause the particle to have a demagnetisation factor in the z-direction of:

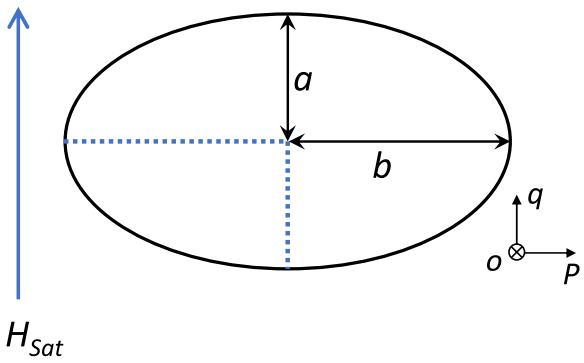


Fig. 1. Coordinate system and geometry of a single isolated particle based on the premise of Cookson [21]. The magnetisation is fully aligned along the z-axis direction as if subjected to a saturating field along the semi-major axis, a. In our case, the geometry is further simplified to that of a sphere by setting a = b.

$$N_Z = \int_{-a}^{a} \int_{-\mathrm{limp}(o)}^{\mathrm{limp}(o)} \left(\frac{-q(o,\ p)}{\left[o^2 + p^2 + (-q(o,\ p)^2\right]^{3/2}} - \frac{q(o,\ p)}{\left[o^2 + p^2 + (q(o,\ p)^2\right]^{3/2}} \right) dp\ do, \tag{3}$$

where $q\ (o,\ p)$ is given by:

$$q(o,p) = \sqrt{1 - p^2 - o^2}, \tag{4}$$

and the limit $p(o)$ is given by

$$\mathrm{limp}(0) = \sqrt{1 - o^2}. \tag{5}$$

It should be noted that the derivation of Eq. (3) in the original work involves division by a particle's or element's magnetisation that reduces the final outcome to that of geometry alone and is therefore material independent.

For a spherical particle surrounded by other identical particles magnetised in the z-direction, Eq.(3) can be adapted to evaluate the contribution to the demagnetising factor of the particle centred at the origin for any given particle in the assemblage. For ease of programming, the particles were arranged in a cubic lattice with a particle centred at every positive integer point $(x,\ y,\ z)$. To consider the volume packing fraction, a lattice scaling factor m is used, such that if the particles are in contact, m is equal to one.

$$N_Z(x,\ y,\ z) = \int_{-a}^{a} \int_{-\mathrm{limp}(o)}^{\mathrm{limp}(o)} \left(\frac{2mz - q(o,\ p)}{\left[(2mx + o)^2 + (2my + p)^2 + (2mz - q(o,\ p)^2\right]^{3/2}} - \frac{2mz + q(o,\ p)}{\left[(2mx + o)^2 + (2my + p)^2 + (2mz + q(o,\ p)^2\right]^{3/2}} \right) dp\ do. \tag{6}$$

Algorithm 1
MATLAB 2022b code was used to calculate the contributions of each particle in a simple cubic lattice to the demagnetization factor of the particle centred at the origin of the lattice.

```
%user defined variables.
  xlength = 101;
  ylength = 101;
  zlength = 101;
  vpf = 0.2;
  %set-up of the matrix Nz and lattice spacing m.
  Nz = zeros (xlength, ylength, zlength);
  m = (pi./(6.*vpf))^(1/3);
  %functions used in the integration
  qsphere = @ (x,y) sqrt(1-y.^2-x.^2);
  ymax = @ (x) sqrt (1-x.^2);
  ymin = @ (x) -1.*sqrt (1-x.^2);
  %calculation of Nz.
  for zg = 0:zlength-1
   Rindex = 0;
   for yg = 0:ylength-1
   for xg = Rindex:xlength-1
   negative = @ (x,y) ((2.*m.*zg)- qsphere(x,y))./(((2.*m.*xg)+x).^2 + ...
   ((2.*m.*yg) + y).^2 + ((2.*m.*zg)- qsphere(x,y)).^2).^1.5;
   positive = @ (x,y) ((2.*m.*zg)+ qsphere(x,y))./(((2.*m.*xg)+x).^2 + ...
   ((2.*m.*yg) + y).^2 + ((2.*m.*zg)+ qsphere(x,y)).^2).^1.5;
   fun = @ (x,y) negative(x,y) - positive(x,y);
   if xg == yg
   Nz (xg+1,yg+1,zg+1) = integral2 (fun,-1,1,ymin,ymax);
   else
   Nz (xg+1,yg+1,zg+1) = integral2 (fun,-1,1,ymin,ymax);
   Nz (yg+1,xg+1,zg+1) = Nz (xg+1,yg+1,zg+1);
   end
   end
   Rindex = Rindex+1;
   end
  end
```

The demagnetising factor of the particle at the origin is determined by summing the contributions from each particle and its own surface poles.

2.2. The contribution matrix

The first part of the model finds the contribution matrix *Nz*, with each element corresponding to the contribution to the demagnetising factor of the particle centred at the origin by a particle centred at (*xg, yg, zg*); Algorithm 1. The user defines the size of the lattice by stating the *x, y*, and *z* lengths (*xlength, ylength*, and *zlength*) and the volume packing fraction *vpf*. The limitation on the size of the lattice is simply imposed by the run time of the program; at the time of writing, a desktop computer typically takes four minutes per one million particles. Note that in the calculation of *Nz*, not every particle has to be assessed due to the symmetry of the lattice; hence the truncated number of calls in the "*xg*" for loop. The double integral in Eq. (6) is performed using the integral2 function: this function uses a tiled numerical integration method to carry out this evaluation [22].

Fig. 2 shows a representation of the matrix *Nz*. The element at index (1, 1, 1) in MATLAB corresponds to the contribution of the particle centred at the lattice position (0, 0, 0) on itself, while element (*xlength, ylength, zlength*) is the contribution of the particle farthest from the lattice origin. This matrix contains all the data required to determine the spatial distribution of the demagnetisation factors of the constituent particles in the z-direction, and therefore, the mean value for the sample.

The final format of the contribution matrix required for the summation sections of the model is formed by taking each element in the matrix N_Z *(x, y, z)* and duplicating the value to all combinations of *(*$\pm x$, $\pm y$, $\pm z$*)*, forming octants. Algorithm 2 shows this operation, with matrix *Nzfull* being the completed contribution or summation matrix and its central element being the element (1, 1, 1) of matrix *Nz*.

The left-hand side of Fig. 3 shows the matrix *Nzfull. The Nzfull* element (*xlength, ylength, zlength*) corresponds to the matrix *element, Nz* (1, 1, 1). The right-hand side shows the premise of the summing operation, which is used to determine the demagnetisation factor of each constituent particle. A subset of *Nzfull*, matching the dimensions of the lattice, starts at a position that fits exactly into one of the octants of matrix *Nzfull*, and summing the elements in this subset gives the demagnetisation factor of the constituent particle at one of the corners of the lattice. Shifting this subset to matrix *Nzfull* and then summing it yields the demagnetisation factor for each constituent particle in turn. This subset is represented in Fig. 3 as a semi-transparent green block. Note that *the Nzfull* element (*xlength, ylength, zlength*) is always contained within the subset, which is the self-demagnetisation field of the particle.

In principle, this summation process is simple and requires only a few lines of code. However, because we are summing the contribution of every particle in the lattice and then repeating this summation for every particle, it is an N^2 problem, and the computational time becomes a factor. Even for small lattice sizes, this simple summation can take several hours to complete. The upcoming sections deal with methods for reducing the computational time and how the summation matrix can be used to calculate the demagnetisation factors of other shapes moving away from the cuboids depicted here.

2.3. Fast calculation of the demagnetisation information for a cuboid sample

The subset is repeatedly moved through matrix *Nzfull,* and then summing the elements of the subset wastes resources simply as we repeat many of the same additions. To reduce this, the subset used to move through the matrix *Nzfull* can be considered as being constructed from columns in the *z-direction*, with the length of the column being equal to *zlength* (the size of the lattice in the *z-direction*). An example of this is shown in Fig. 4. The elements contained within each column can be summed, with the summation being recorded in another matrix, *Downstack* (right-hand side of Fig. 3). For each *x-y* coordinate, a column will need to be summed, of length *zlength*, for z coordinates going from (2*zlength* – 1) to *zlength*, so that all elements in *Nzfull* are covered. Algorithm 3 is the code for creating and populating the matrix *Downstack*, which holds all the column summations. The variables *i, j*, and *k* are related to the *x, y, and z* elements of the matrix *Nzfull*.

The final number of summations required per particle can be further reduced using the summed columns to form summation slices in the y-z plane (*xslices*) of *Nzfull*. An example is shown in Fig. 5, which is achieved by summing rows in the y-direction of the *Downstack* matrix of length *ylength* and creating a new matrix, *xslices*, to hold the slice summations. Algorithm 4 gives the code for this task, where *i, j, and k* are used as the *x, y, and z* coordinates of the matrix *Downstack*.

In the final stage, the total demagnetisation factor (saved to matrix *Demagmatrix*) of each constituent particle is obtained by summing the respective x slices associated with the particle, as shown in Fig. 6 and the code in Algorithm 5. The final part of the code converts the

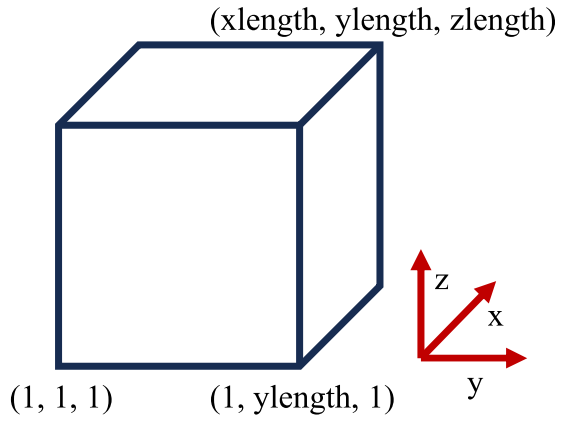


Fig. 2. Representation of matrix N_z. Each element contributes to the demagnetisation factor of the particle centred at the origin of the lattice for each particle in the lattice (xq, yq, zg). Element (1, 1, 1) is the self-contribution of the particle at the origin of the lattice (0, 0, 0). Note the shift in referencing the elements to the particles in the lattice and that N_z is the same size as the user-defined lattice.

Algorithm 2

MATLAB 2022b code was used to create matrix Nzfull by replicating the contents of matrix Nz to form eight octants centred on Nz's element (1, 1, 1).

```
%set-up "summation matrix"
  Nzfull = zeros ((2*xlength)-1,(2*ylength)-1,(2*zlength)-1);
  %populate "summation matrix"
  for zg = -zlength+1:zlength-1
   for yg = -ylength+1:ylength-1
   for xg = -xlength+1:xlength-1
   Nzfull(xg+xlength,yg+ylength,zg+zlength) = ...
   Nz(abs(xg)+1,abs(yg)+1,abs(zg)+1);
   end
   end
  end
```

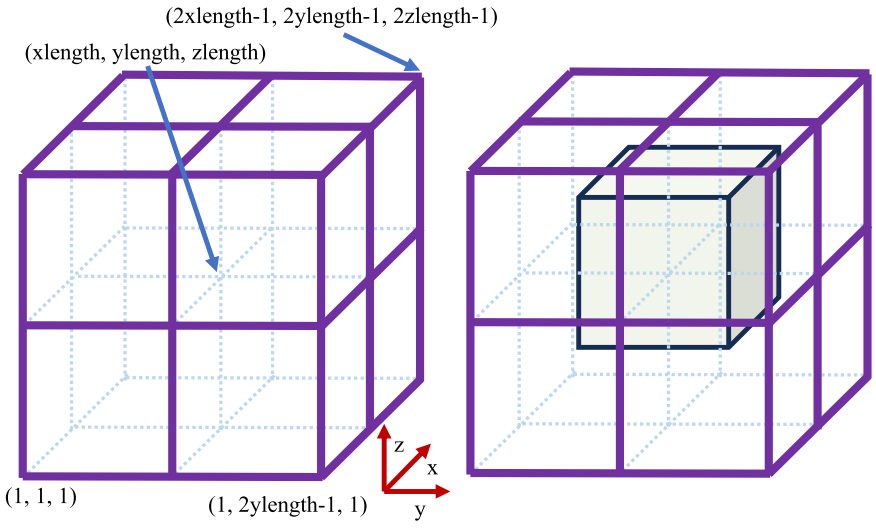


Fig. 3. The left-hand side shows the overall dimensions of the summation matrix Nzfull. The central element (xlength, ylength, and zlength) is equal to Nz (1, 1, 1). The right-hand side shows the basic premise of the summation technique. The overall demagnetisation factor of each constituent particle can be determined by moving a subset equal in size to the user-defined lattice through Nzfull and summing each time over its content. Note that the central element containing the particle's self-demagnetisation field contribution is always included in the subset.

demagnetisation factor to SI and calculates the mean demagnetisation factor of the sample. By removing the repetition of the summation process, this program takes seconds to run compared to the hours of labouriously moving and summing the contents of the subset.

2.4. *Fast calculation for a containing vessel of any shape*

For a container of any shape, reduction summation techniques used for simple cuboids are difficult to implement. An alternative is to use the

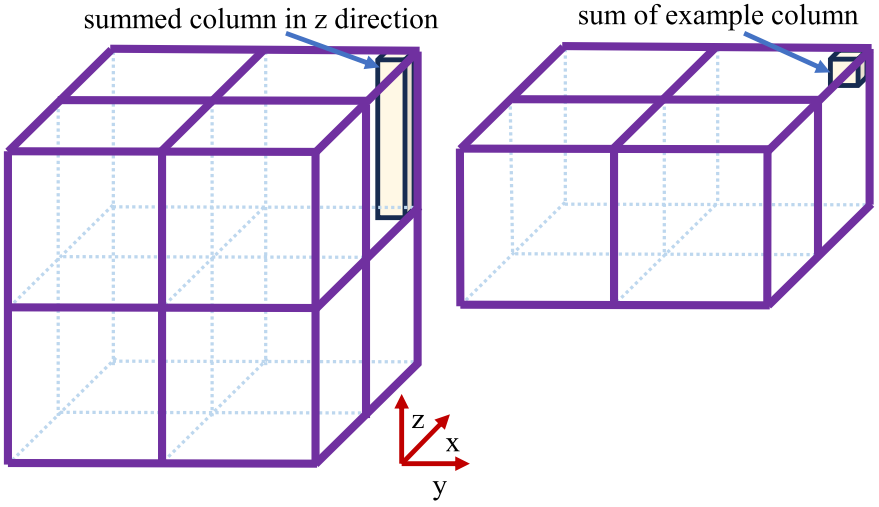


Fig. 4. The left-hand side shows the first-column summation, column length zlength, which is used to break down Nzfull into smaller chunks. A column summation is performed for every x y coordinate, starting in the z plane (2zlength -1), and then repeated for other z planes up to and including the z plane (zlength). Each column summation is written in the matrix Downstack. The right-hand side is a representation of Downstack and shows the location of the summation of the columns shown on the left.

Algorithm 3
MATLAB 2022b code was used to create and populate the matrix Downstack. Each element of Downstack is a column summation, length of zlength, taken from the matrix Nzfull.

```
%create Downstack matrix
  Downstack = zeros ((2*xlength)-1, (2*ylength)-1, zlength);
  % find column (z direction) sums and populate Downstack
  for k = (2*zlength)-1:-1:zlength
   for j = (2*ylength)-1:-1:1
   for i = (2*xlength)-1:-1:1
   Downstack(i, j, k-(zlength-1)) = sum(Nzfull(i, j, k-(zlength-1):k));
   end
   end
  end
```

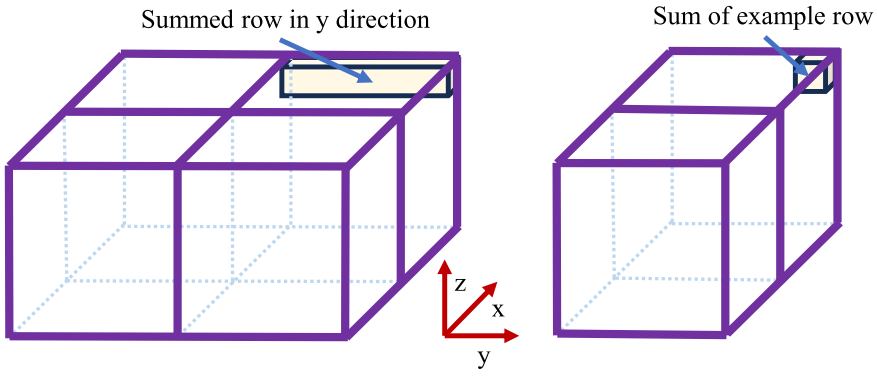


Fig. 5. The left-hand side shows the first-row summation of the matrix Downstack, length of ylength; this is equivalent to summing up a slice in the Ndfull matrix of dimensions (1, ylength, zlength). This summation would need to be repeated for every x z coordinate in each y plane, starting from the y plane (2ylength-1) and ending on (and including) the y plane (ylength). This "slice" summation is stored in the matrix xslice; a representation is shown in the right-hand side for this first-row summation.

Algorithm 4
MATLAB 2022b code was used to create and populate matrix xslices. Each element in this matrix is the summation of a row, length of ylength, taken from the matrix Downstack. This summation is equivalent to summing a slice in the matrix Ndfull in the x-plane of dimensions (1, ylength, zlength).

```
%create xslice
  xslice = zeros ((2*xlength)-1, ylength, zlength);
  %calculating all the x slice summations
  for k = zlength:-1:1
   for j = 0:ylength-1
   for i = (2*xlength)-1:-1:1
   xslice(i, ylength-j, k) = ...
   sum(Downstack(i, ylength-j:(2*ylength)-1-j, k));
   end
   end
  end
```

concept of regression towards the mean; if a mean is taken from an increasing number of samples taken from a population, the closer the mean for the sample is to the mean of the population. The results in this study show that it is possible to obtain the mean demagnetisation factor of a sample by examining only as little as 5 % (or even less) of the constituent particles, meaning that summing over the subset taken from the matrix *Nzfull* becomes a viable option.

For cuboids, the shape of the containing vessel is defined by the dimensions of the lattice. The cuboid lattice is still used in this variation of the summation code, but an additional matrix is used, a mask, which specifies if a particle is absent or present with its elements being either "0" or "1" respectively. In this manner, only the particles within the mask are considered in the final demagnetisation factor calculation, while maintaining the efficiency described earlier for the fast calculation of the cuboid matrix. This mask matrix, *Othello*, consists of a user-defined lattice, a cuboid, with the shape of interest etched within.

The mask matrix can be simply typed by the user. However, ellipsoid containers were used for testing because their demagnetisation factors are well known. An example program for generating ellipsoid masks is provided in Appendix A. Fig. 7 shows an example of a spherical global shape. The cuboid shape represents the matrix *Othello*; all the particles of *Othello* that reside in the sphere would be set at "1" while the others would be at "0".

Determination of the number of samples to be taken is based on the principle of regression to the mean, which is illustrated in the results section. Here, the first part of the program shows the implementation of the number of samples to be taken from the population of particles and then to randomly select particles up to a user-defined percentage, Algorithm 6. The fraction of samples to be taken *frac* in the example code is 5 %. The number of particles contained in the container *msize*, is found by counting the number of "1" contained within the mask matrix *Othello*. The total number of samples to be taken is the number of particles multiplied by the required sample fraction, *samples*.

The particles are then randomly selected from the population, and their coordinates are recorded in arrays *p, q,* and *r* (*x, y,* and *z* coordinates, respectively). A test is conducted to check whether the randomly selected coordinates are for a particle that is present in *Othello*; if not, another coordinate will be selected, and the test is reapplied.

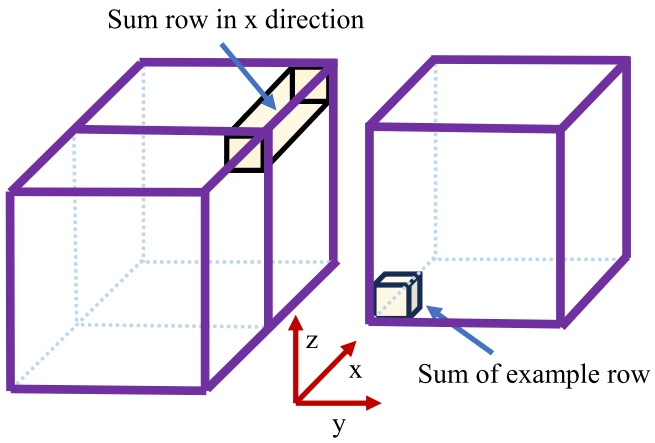


Fig. 6. The left-hand side shows the first-row summation, length of xlength, made in the matrix xslices, which provides the full demagnetisation factor of the particle at position (0, 0, 0) in the user-defined lattice. This subset was originally envisaged in the first iteration of the model. This summation is repeated for every y z coordinate for each x plane, starting from the x plane (2xlength-1) and ending at, and including the x plane (xlength). The resulting summations are written to the matrix Demagmatrix, as shown on the right-hand side. Each element in Demagmatrix is the full demagnetisation factor of a particle in the lattice. Note the shift in referencing as element (1, 1, 1) refers to the demagnetisation factor of the particle at the lattice position (0, 0, 0).

The second part of the program, Algorithm 7, creates a matrix *Nzsub*, which functions as the subset in the contribution matrix *Nzfull*, and a matrix *Demagmatrix*, to hold the summation of the contributions held in this subset. Based on the coordinates of each selected particle, the subset matrix is populated from *Nzfull* and multiplied by the mask matrix. This provides the actual contribution of the particles within the container shape. The demagnetisation factor of the particle is then determined by summing all the contributions within the subset. These were converted into SI values and the mean and standard error were calculated.

2.5. Fast calculation for container shapes with uniform cross-section

The technique presented in this section applies to any container vessel with a uniform cross-section in the *x-y* plane (perpendicular to the applied magnetic field in the *z-direction*). A representation of this is shown in Fig. 8. Similar to the cuboid program, the contribution matrix is summed up in columns in the z-direction, Algorithm 3. For a subset moving through *Nzfull*, representing the elements that must be summed per constituent particle, we can envisage that the subset consists of these columns. It is then possible within this subset to accept or reject columns to be included in the summation, depending on the required shape described by a 2-D mask matrix.

To define the shape, as for the generic container routine, a mask is required; in this case the mask takes the form of a 2-D matrix (*Othello*), again with "0" and "1" to represent the absence or presence of a particle. In this case, the mask defines the container shape in the *x-y* plane. Because the demagnetisation factors of cylinders are well defined in the literature, these were the shapes used for testing. A program to create this 2-D mask, a circle contained within the given lattice *x-y* space, is presented in Appendix B. Fig. 9 shows a top-down view of the matrix relationships. The purple squares represent the matrix *Downstack*. A subset 2-D matrix, *isolate*, represented by the black square, is moved

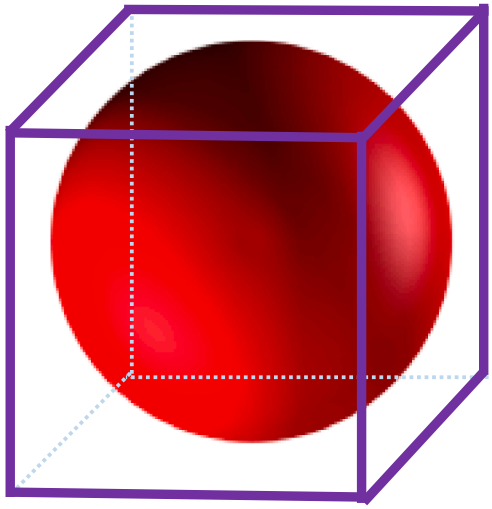


Fig. 7. Representation of mask matrix Othello for a spherical sample container shape. All the elements that sit fully within the sphere will be set at "1" while other elements will be at "0", representing the presence or absence of particles in the lattice.

Algorithm 5

MATLAB 2022b code was used to create and populate the matrix Demagmatrix. Each element in this matrix is the summation of a row, length of xlength, taken from the matrix xslices. This summation is equivalent to summing up the subset of matrix Ndfull, as originally intended, as shown in Fig. 3.

```
%create Demagmatrix
  Demagmatrix = zeros (xlength,ylength,zlength);
  %populate Demagmatrix
  for k = 0:zlength-1
   for j = 0:ylength-1
   for i = 0:xlength-1
   Demagmatrix(i+1, j+1, k+1) = ...
   sum(xslice(xlength-i:(2*xlength)-1-i, ylength-j, zlength-k));
   end
   end
  end
  Demagmatrix = abs(Demagmatrix)./(4*pi);
  valmean = mean(mean(mean(Demagmatrix)))
```

Algorithm 6
MATLAB 2022b code was used to calculate the number of particles required from the total population to be used to calculate the mean demagnetisation factor and their respective coordinates in the user-defined lattice.

```
%number of samples to take
 frac = 0.05;
 msize = sum(sum(sum(Othello)));
 samples = round(msize*frac);
 %select coordinates of particles to sample
 for i = 1:samples
  cond = false;
  while cond == false
  p(i) = randi(xlength);
  q(i) = randi(ylength);
  r(i) = randi(zlength);
  if Othello(p(i), q(i), r(i)) == 1
  cond = true;
  end
  end
 end
```

Algorithm 7
Matlab2022b code that calculates the demagnetisation factors of only the selected particles, calculating a mean value based on this limited selection.

```
%create matrices Nzsub (subset) and Demagmatrix (particle demagnetisation factors)
 Nzsub = zeros (xlength,ylength,zlength);
 Demagmatrix = zeros (xlength,ylength,zlength);
 %populate Demagmatrix
 for i = 1:samples
  Nzsub = Nzfull(xlength+1-p(i):(2*xlength)-p(i),...
  ylength+1-q(i):(2*ylength)-q(i),zlength+1-r(i):(2*zlength)-r(i));
  Nzsub = Nzsub.*Othello;
  Demagmatrix(p(i),q(i),r(i)) = sum(sum(sum(Nzsub,3)));
 end
 %calculation of the mean and standard error of the sample's demag factor.
 Demagmatrix = abs(Demagmatrix)./(4*pi);
 Demagmatrix = Demagmatrix.*Othello;
 Demagmatrix(Demagmatrix == 0) = NaN;
 valmean = mean(Demagmatrix,"all","omitnan")
 valstd = std(Demagmatrix,[],"all","omitnan");
```

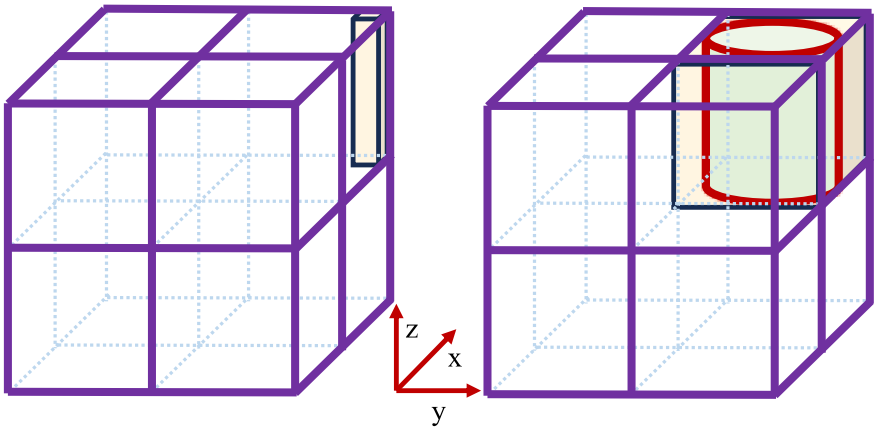


Fig. 8. The left-hand side is a representation of the matrix Nzfull, with the first column to be summed highlighted. This is the same technique as that used in the fast cuboid summation algorithm. However, the right-hand side shows the difference between the two algorithms: a mask can be used to define which columns to accept or reject as part of the overall summation, with acceptance being given to columns that fit in the desired shape, in this case, a cylinder.

through Downstack; for each element in *Downstack, isolate* is multiplied by the mask matrix *Othello* (same dimensions as *isolate*, with "1" values in the red circle) and the product, *tosum*, is then summed up to give the demagnetisation factor for a given lattice position, matrix *Demagmatrix*. By multiplying *Demagmatrix* by the mask matrix for each *z-layer, Demagmatrix* will only contain demagnetisation factors for particles present (rather than the contributions of magnetic fields generated by constituent particles at points outside the sample), as given in Algorithm 8.

3. Results and discussion

3.1. Fast calculation for cuboids

In previous work, using the original iteration of the model [19], a concern was addressed regarding sample size; testing to see if appreciable differences in the distributions of the demagnetisation factors of the constituent particles occurred if larger sample sizes were used, even if the overall aspect ratio of the containing shape remained the same. A

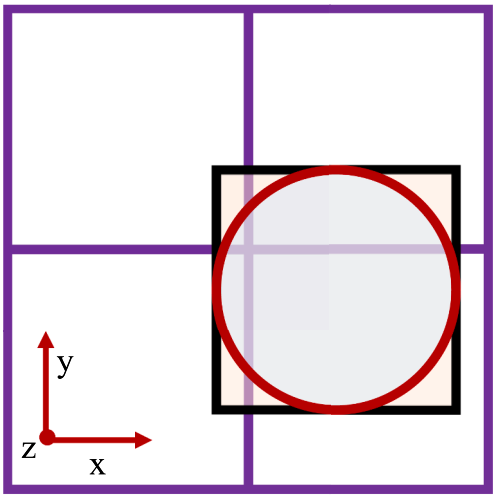


Fig. 9. A 2-D view of the matrix Downstack, looking down in the z-direction. The black square represents the matrix isolate, which is moved through Downstack, while the red circle within isolate, as defined by the matrix Othello, represents the columns that will be accepted for summation.

comparison was made between two cube samples, one with an edge length of 41 particles and the other 151 particles. The normalised distributions of the demagnetisation factors of the constituent particles were examined. A two-sample Kolmogorov-Smirnov test was performed with the null hypothesis that the two datasets were drawn from the same distribution. The test indicated that the null hypothesis was not rejected at the 5 % significance level with a p-value of 0.8938.

The difference between the original iteration of the model and the latest version is simply the efficiency, which significantly reduces runtime. The original version was capable of handling sample sizes that consisted of approximately one million particles; above this, run times quickly escalated to several days and were simply not feasible. With improvements in the efficiency of the model, larger samples could be evaluated. Fig. 10 shows the normalised distributions for cubic samples, one of the edge length 101 particles, and the other 501 particles; both samples have touching particles ($m = 1$, giving volume packing fractions of $\pi/6$). The two-sample Kolmogorov-Smirnov test again showed that the null hypothesis was not rejected at the 5 % significance level, but with an increased p-value of 0.9848. The larger sample's distribution appears smoother and does not form such a noticeable secondary high-end peak, but the similarity is striking. Because the demagnetisation factors in the model are all related to the geometry (the demagnetisation field within the constituent spherical particle is uniform and has no dependence on its susceptibility), it is not surprising that small sample

Algorithm 8

MATLAB 2022b code was used to calculate the demagnetisation factors for the constituent particles in a container shape with uniform cross-section, as defined in the 2-D mask matrix Othello.

```
%create Demagmatrix
  Demagmatrix = zeros (xlength,ylength,zlength);
  %isolating required columns per particle and populating Demagmatrix
  layer = 0;
  for k = zlength:-1:1
   layer = layer + 1;
   for j = 0:ylength-1
   for i = 0:xlength-1
   isolate = Downstack(xlength-i:(2*xlength)-1-i,...
   ylength-j:(2*ylength)-1-j, k);
   tosum = isolate.*Othello;
   Demagmatrix(i+1,j+1,layer) = sum(sum(tosum));
   end
   end
  end
  %"shaping" Demagmatrix; only for present particles
  for z = 1:zlength
   Demagmatrix(:,:,z) = Demagmatrix(:,:,z).*Othello;
  end
```

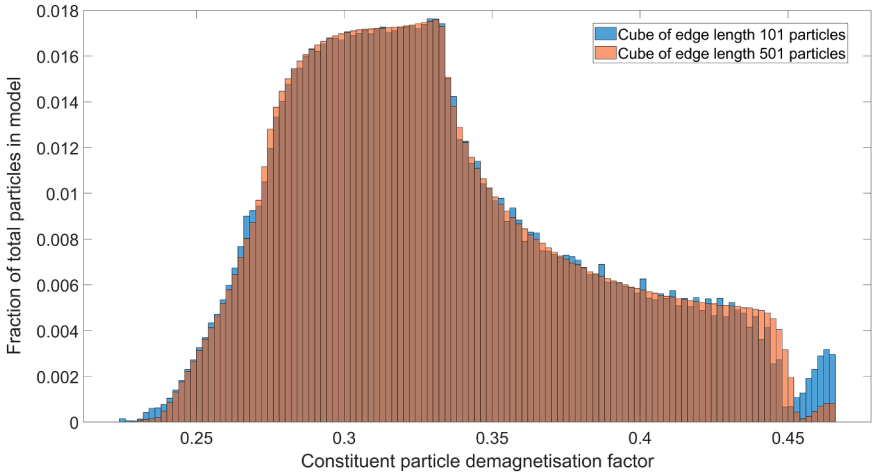


Fig. 10. Comparison of the distributions of the demagnetisation factors of the constituent particles for cubic samples with different edge lengths obtained from the model. A comparison is made between a sample of edge length 101 particles and another of length 501 particles.

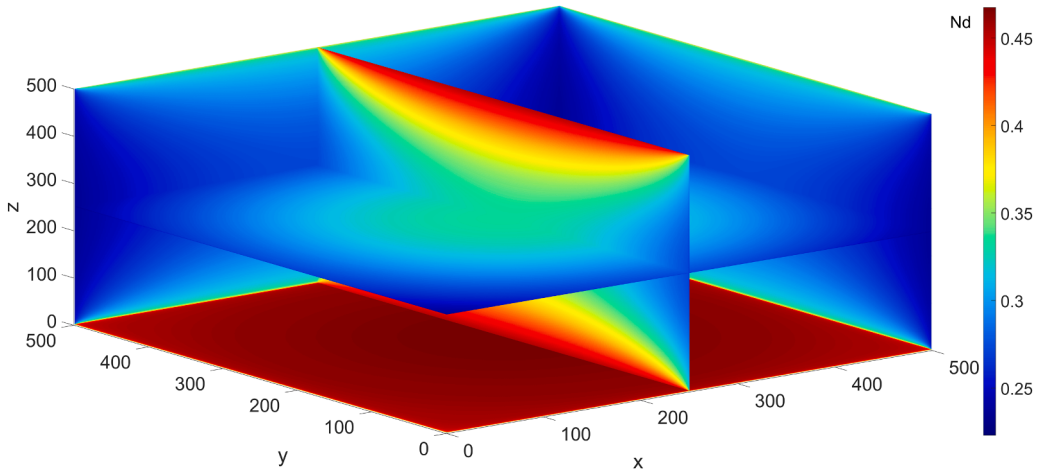


Fig. 11. Spatial distribution of demagnetisation factors of constituent particles for cubic sample with edge length of 501 particles.

Table 1
A comparison of the mean demagnetisation factors of the cuboid samples N_z obtained from the model and Eq. (2). Note that D_z is obtained from [12] and is for cuboids with aspect ratios varying from one to ten (comparing the z-direction size to that of x and y). This is a close but not exact match for the samples simulated in the model, apart from the z length of 101 particles).

Sample size (x, y, z)	D_z, demag. factor of overall shape (Aharoni)	N_z, mean demag. factor (Eq. (2))	N_z, mean demag. factor (model)	Ratio of N_z (model over Eq. (2) output)
(101, 101, 101)	0.33333	0.33333	0.33333	1.0000
(101, 101, 201)	0.19832	0.26264	0.26293	1.0011
(101, 101, 301)	0.14036	0.23229	0.23254	1.0011
(101, 101, 401)	0.10845	0.21558	0.21577	1.0009
(101, 101, 501)	0.088316	0.20504	0.20517	1.0006
(101, 101, 601)	0.074466	0.19779	0.19787	1.0004
(101, 101, 701)	0.064363	0.19250	0.19254	1.0002
(101, 101, 801)	0.056670	0.18847	0.18848	1.0001
(101, 101, 901)	0.050617	0.18530	0.18529	0.9999
(101, 101, 1001)	0.045731	0.18274	0.18271	0.9998

sizes would give a good approximation to that of larger samples, as the containing shape itself is important. Larger sample sizes yielded better approximations, but there was an element of diminishing returns between the computational time and the quality of data obtained.

The model was designed such that the 3D spatial distribution of the demagnetisation factors of the constituent particles can be found, as shown in Fig. 11, for a cube sample with an edge length of 501 particles that are in contact. The figure shows slices taken from the output of the model, illustrating the spatial distribution of the demagnetisation factors of the constituent particles. With the applied field in the z-direction, as shown, the largest N_z values were observed on the top and bottom surfaces, while the lowest values were observed on the side surfaces, as expected. For clarity, only two slices cutting through the cuboid are shown and illustrate the tendency to converge on N_z values of $^1/_3$ towards the centre particle. This was consistent with the dominance of the cuboid shape at the surfaces (and their associated surface poles) reducing to that of a single isolated particle.

To demonstrate the validity of the model, the mean demagnetisation factor N_Z was calculated from the spatial distribution and compared with the output of Eq. (2). This can be performed for a range of cuboid sizes at a set packing fraction volume. The model was used with a set number of particles in the *x*- and *y-directions* of 101 particles and at a volume packing fraction of $\pi/6$ ($m = 1$). The sample size in the *z-direction* was varied between each test, ranging from 101 to 1001 particles with a step size of 100. The mean demagnetisation factors for these ten tests are shown in Table 1. The table also includes the mean demagnetisation factor calculated using Eq. (2): The value of the demagnetisation factor of the cuboid D_z taken from the calculations of Aharoni [12] (this is the magnetometric demagnetisation factor with susceptibility being zero). A comparison of these two values of N_Z illustrates their closeness, giving confidence in the model. The full model data are summarised as cumulative distribution functions, as plotted in Fig. 12.

As the cuboid became increasingly elongated in the *z-direction*, the distribution of the constituent particle demagnetisation factors became sharper and skewed, and the mean demagnetisation factor decreased. This behaviour follows the pattern observed in solid samples [1,12], whereby increasingly elongated samples would tend to have demagnetisation factors approaching zero in the direction of elongation. Fig. 13 and Fig. 14 show the normalised and spatial distributions of the demagnetisation factors of the constituent particles held within a cuboid sample with a ratio of 10:1. When compared to the same matching data for a cubic sample (ratio 1:1) in Fig. 10 and Fig. 11, the impact of the elongation is evident.

The impact of removing duplication in the summing process carried out in this new algorithm on performance was evident. A linear relationship between execution time and simulation size was found, running at 10.2 seconds per million particles. This compares to the squared relationship for the original simple summation process with an execution time of approximately 4,000 seconds for a simulation of a million particles and 100,000 seconds for five million particles. Further details are given in Appendix C and are illustrated in Fig. C 1.

3.2. Fast calculation for any shape by using regression to the mean

At the heart of this process is the use of regression towards the mean to enable a feasible runtime. To evaluate the amount of sampling required to obtain meaningful data, a dataset was used that had a known average demagnetisation factor. In this case, the spatial distribution of

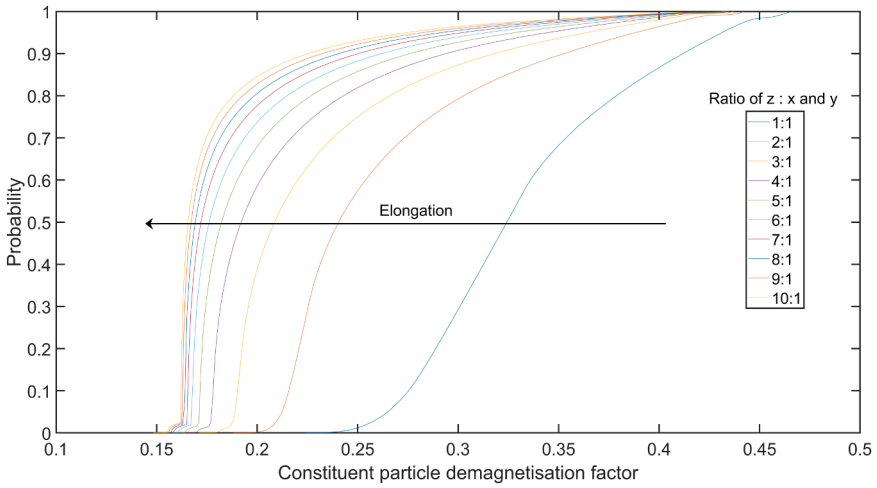


Fig. 12. Cumulative frequency distribution of the cuboid samples. As the cuboid became elongated, the distributions became sharper and more asymmetrical. The ratios given here (z-direction size compared to the x-y size) have been rounded for simplicity.

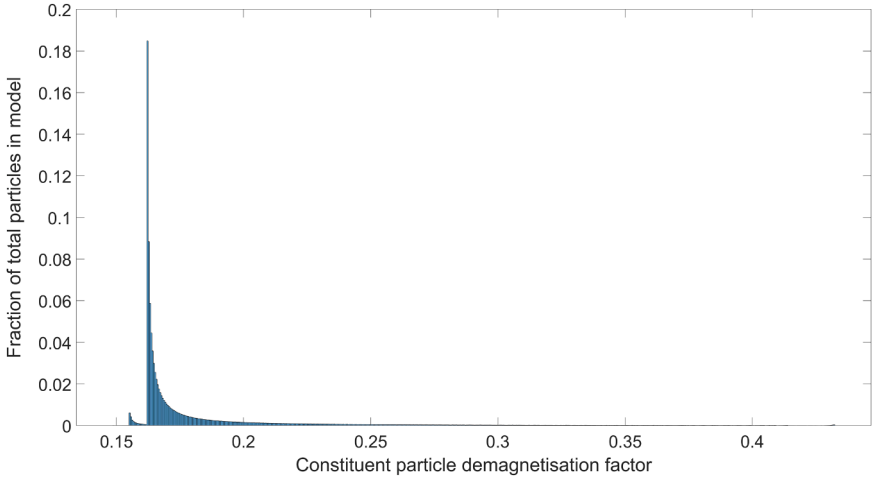


Fig. 13. Demagnetisation factors of constituent particles for a cubic sample with dimensions of 101 × 101 × 1001. This clearly shows a shift to lower values of demagnetisation factors as the cuboids are elongated, with sharper peaks and asymmetric distributions compared to Fig. 10.

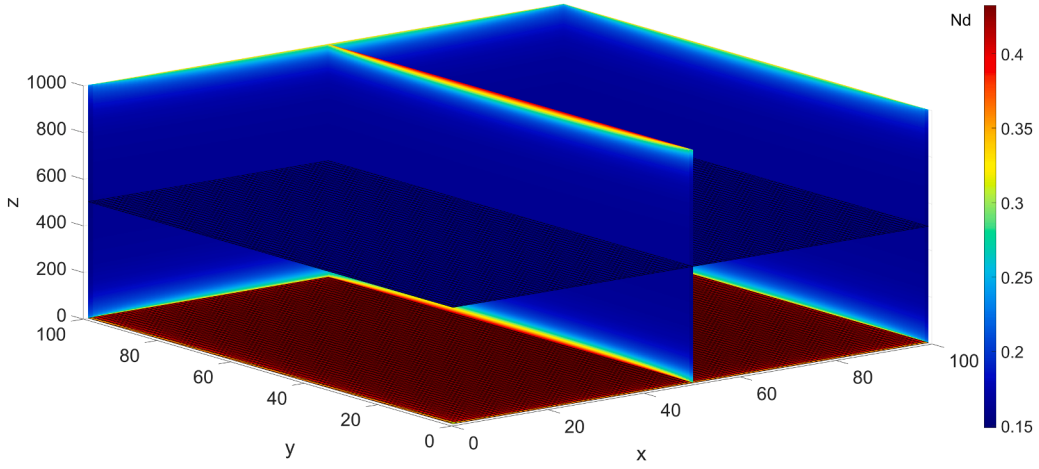


Fig. 14. Spatial distribution of demagnetisation factors of constituent particles for a cuboid sample with dimensions of 101 × 101 × 1001. Compared with a cubic sample, Fig. 11, the constituent particle demagnetisation factors are typically lower. The particles within the bulk have relatively consistent values with those near the top and bottom surfaces, where the magnetic field enters and leaves the sample, with a large variation from the mean.

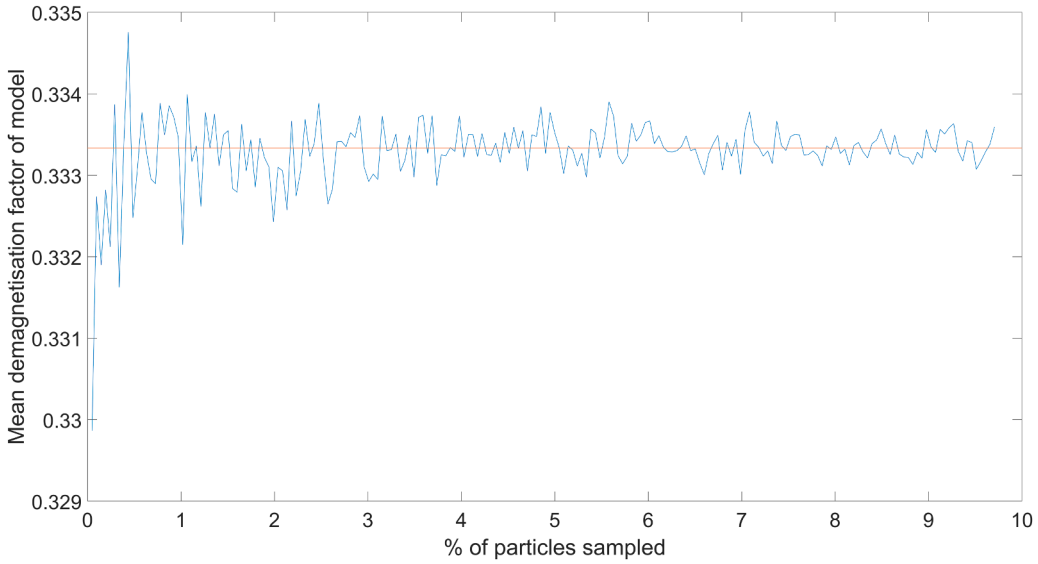


Fig. 15. The mean demagnetisation factor of a cubic sample with an edge length of 101 particles was calculated from the increasing percentage of the sampled particles. The mean value of such a sample is $^1/_3$ (orange line), and the graph shows that as more particles are sampled, the calculated mean value regresses towards this.

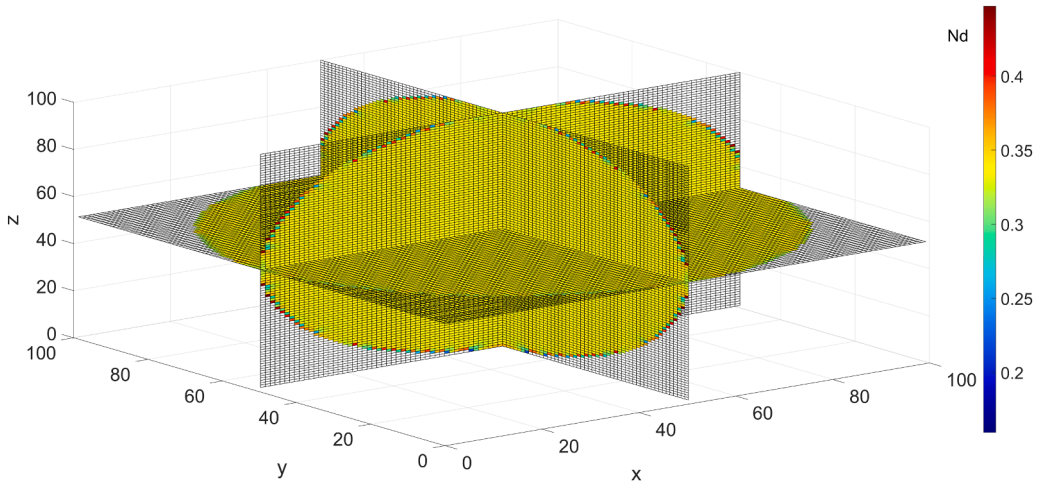


Fig. 16. Spatial distribution of the demagnetisation factors of the constituent particles for a spherical sample with a diameter of 101 particles. Most of the particles have values of approximately $^1/_3$, which is expected because this is the mean demagnetisation factor of a sphere and the demagnetisation field of an ellipsoid is uniform. The particles closer to the surface of the sphere deviate from this value.

the demagnetisation factors for the constituent particles of a cube sample is edge length 101 particles (m = 1) with a mean demagnetisation factor of $^1/_3$. By randomly selecting particles and maintaining a running average, a comparison can be made between this value and the known average for the entire dataset. This can be performed several times to obtain confidence in the results, as shown in Fig. 15. Typically, after sampling 5 % of the particles, the running average was within 0.2 % of the mean of the entire sample, and this Fig. was used throughout most of the testing for this algorithm.

To gain confidence in this element of the model, it was decided to evaluate it while examining ellipsoid container shapes, ellipsoids with well-defined D_z values [11], and the interesting characteristic that for solid volumes, the demagnetisation field is uniform for ellipsoids throughout. Additionally, the random sampling element was removed (100 % of particles examined) for particles within a sphere with a diameter of 101 particles. This enabled us to obtain a spatial distribution at the cost of a significantly longer running time for this container shape. The spatial distribution of the demagnetisation factors of the constituent particles is shown in Fig. 16. Of interest is the uniformity of the demagnetisation factors, clustered about the expected value of $^1/_3$ with deviations occurring towards the surface. This is consistent with the expectation of a solid sphere having a uniform demagnetisation field [1], with the difference caused by the packed powder nature of the sample. Tight clustering is illustrated by the distribution shown in Fig. 17. Fig. 18 shows a look down view onto the spherical sample such that the demagnetisation factors of the particles on the surface of the upper hemisphere are presented in a 2-D format. Here, it can be clearly observed that there is a high variation in the demagnetisation factors at the surface compared with the bulk.

The model was run for samples that consisted of ellipsoid container shapes, packed with particles with a volume packing fraction of $\pi/6$ (m = 1), with a constant diameter in the x and y directions of length 101 particles, sampling 5 % of the constituent particles. The first run had a z-direction diameter of 101 particles, with subsequent runs increasing this diameter by 100 each time, up to a final value of 1001. The mean demagnetisation factors of the samples are listed in Table 2. For

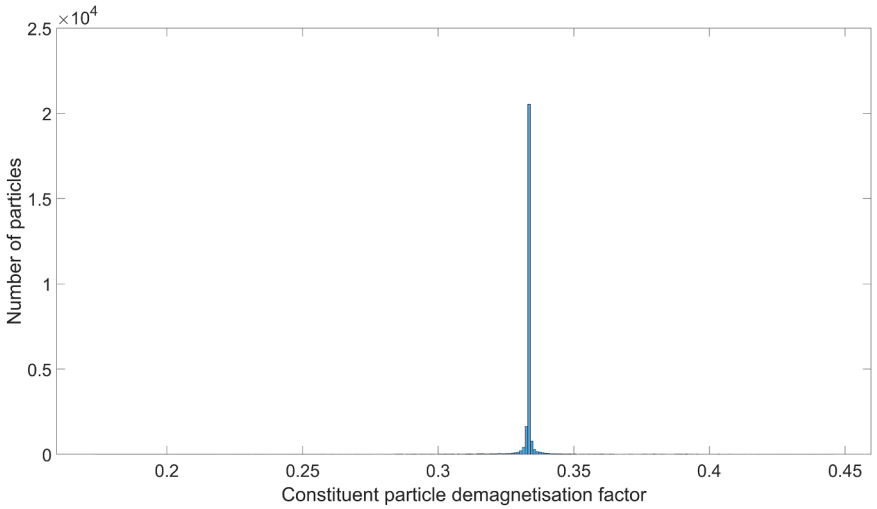


Fig. 17. Distribution of the constituent particle's factor for a sphere of diameter 101 particles. The distribution is sharp and centred at a mean value of $^1/_3$. The x-axis scale includes the minimum constituent particle demagnetisation factor of 0.1595 and maximum of 0.4470, thus emphasising the narrowness of the distribution.

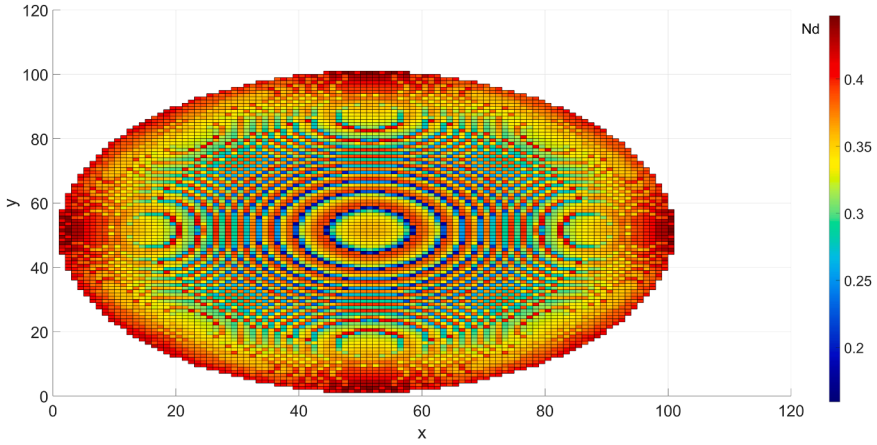


Fig. 18. Spatial distribution of the demagnetisation factors of the particles at the surface of the upper hemisphere of the spherical sample: An effective lookdown view of the data, as shown in Fig. 16, following the axis of the magnetic field. This illustrates the large variation in the demagnetisation factors of the particles compared with those within the bulk of the sphere.

comparison, Table 2 lists the mean demagnetisation factors calculated using Eq. (2). For the ellipsoid samples, the N_z values from both the model and Eq. (2) are close, thus providing confidence in the model. The full model data are summarised in the cumulative distribution functions plotted in Fig. 19. As the ellipsoid is elongated in the z-direction, the distribution of the demagnetisation factors of the individual particles shifts to lower values; however, the sharpness of the distribution remains, which is consistent with the demagnetisation field within the sample being uniform within the bulk.

Apart from being able to obtain more information about the demagnetisation of a sample than the mean value that can be extracted from Eq. (2), it is possible to obtain the magnetometric demagnetisation factor (for zero susceptibility) for any solid shape. The shape would be defined in the same way as used here for the ellipsoids using the "0" and "1" as absent or present particles in the *Othello* matrix. The mean value of the demagnetisation factor output from the model, along with the volume packing fraction (simple calculation), can then be inputted into Eq. (2), which can then be used to obtain the demagnetisation factor of the shape D_z.

Since the summation process used in this algorithm is a simple summation over a selected subsection of the matrix *Nzfull*, the actual performance speed is slow, on par with the original model's cuboid summation process (and hence the need to use sampling and regression to the mean). Typically, in testing, the improvements using the sampling and regression approach mean this algorithm was working at 3.95 seconds per billion summations performed.

3.3. Fast calculation for container shapes with uniform cross-section

To allow a clear comparison with the accepted values in the literature, this algorithm was used to examine the demagnetisation information for cylindrical samples packed with particles to obtain a volume packing fraction of 0.513 (m = 1). The cylinders had a consistent diameter of 101 particles, with the length of the cylinder (z-direction) altered. The samples were investigated as cylinders with lengths ranging from 101 to 1001 particles in steps of 100. In addition, the lengths of the 51 and 2001 particles were examined. An example of the spatial distribution of a cylindrical sample is shown in Fig. 20. The mean demagnetisation factors of the samples are listed in Table 3. For comparison, the table also includes the mean demagnetisation factor calculated using Eq. (2). Similar to cuboid samples, if a sample consists of a solid cylinder, its demagnetisation field is not uniform, and therefore its susceptibility must be considered. Hence, the D_z values used in Eq. (2) corresponds to zero susceptibility [13]. A close relationship exists

Table 2
A comparison of the mean demagnetisation factors of ellipsoid samples N_z obtained from the model and Eq. (2), with a fixed packing fraction of $\pi / 6$ ($m = 1$). Note that D_z is obtained from [11] and is for ellipsoids with aspect ratios in the range of one to ten (comparing the z-direction size to that of x y). There is a close but not exact match for the samples simulated in the model, apart from the z-length of 101 particles).

Sample size: diameters (x, y, z)	D_z, demag. factor of overall shape (Osborn)	N_z, mean demag. factor (Eq. (2))	N_z, mean demag. factor (model)	Ratio of N_z (model over Eq. (2) output)
(101, 101, 101)	0.33333	0.33333	0.33332	1.0000
(101, 101, 201)	0.17356	0.24965	0.25027	1.0025
(101, 101, 301)	0.10871	0.21571	0.21630	1.0027
(101, 101, 401)	0.075407	0.19825	0.19875	1.0025
(101, 101, 501)	0.055821	0.18803	0.18837	1.0018
(101, 101, 601)	0.043230	0.18142	0.18167	1.0013
(101, 101, 701)	0.034609	0.17691	0.17710	1.0011
(101, 101, 801)	0.028421	0.17366	0.17382	1.0009
(101, 101, 901)	0.023816	0.17125	0.17132	1.0004
(101, 101, 1001)	0.020286	0.16940	0.16945	1.0003

between the N_z values obtained from the model and those obtained from Eq. (2); however, there was slightly more variation than that for the previously discussed shapes. The full model data are summarised in the cumulative distribution functions plotted in Fig. 21. Similar to that of the cuboid samples, when the length of the sample in the z-direction was low, the distribution of the constituent particle's demagnetisation factors was broad; however, as the length increased, the distribution became increasingly sharper and asymmetrical in the same manner as that for the elongation of a cuboid.

Since this algorithm used some of the techniques utilised in the cuboid summation process, its execution time was relatively quick. On the cylinder simulations, this script was typically taking 27.7 seconds per million lattice spaces.

3.4. *The contribution matrix: timing performance*

All the summation techniques discussed require the contribution matrix *Nz* as an input. With the improvements and new techniques used, it is the generation of this matrix that is now, in most cases, the most time-consuming part of the overall simulation. The relationship between execution time and simulation size is linear, typically taking 374 seconds per million particle contributions.

4. Conclusion

A polar-based model was developed to determine the overall (mean) magnetometric demagnetisation factor of an assembly of spherical particles contained within a given shape, in conjunction with their distribution throughout the volume.

Initially, based on a model developed for cuboid containers [19], the efficiency was significantly improved by eliminating unnecessary repetitions of particle summations when moving from element to element, thereby reducing the total calculation time, typically from hours to seconds. The method is equally applicable to any volume with a uniform cross-sectional shape and was tested using cylinders.

Extending the model further using the concept of a mask applied to the original cuboid matrix allowed demagnetisation factors for a series of different shapes to be calculated. The ellipsoids and cylinders were chosen to enable comparison with the accepted values for solid shapes, and in each case, compared well to the model by applying the known solid factors to the simple analytical expression, Eq. (2), which considers the effects of the volumetric packing fraction.

Further efficiencies were obtained by utilising a regression to the mean methodology, which showed sufficient convergence on ellipsoids after randomly sampling as little as 5 % of the constituent particles. Ellipsoids were deliberately chosen to provide confidence more widely in the model when applied to other shapes because of (i) variation in the cross-section and (ii) characteristically known to have a uniform

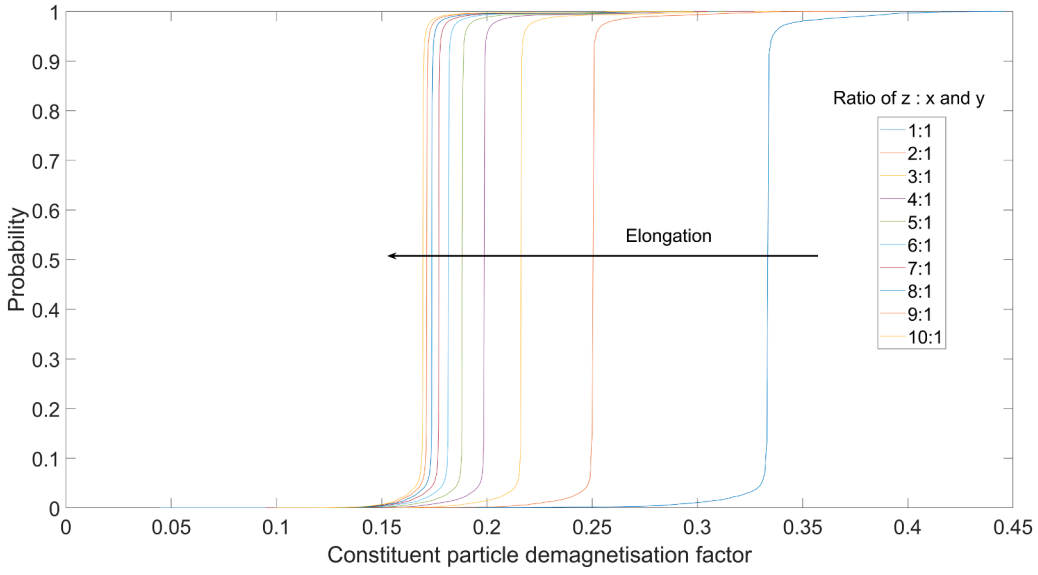


Fig. 19. Cumulative frequency distribution of ellipsoid samples. As the ellipsoid became elongated, the demagnetisation factor of the constituent particles decreased, but the distributions remained sharp and tightly clustered around the mean value. The ratios given here (z-direction size compared to the x-y size) have been rounded for simplicity.

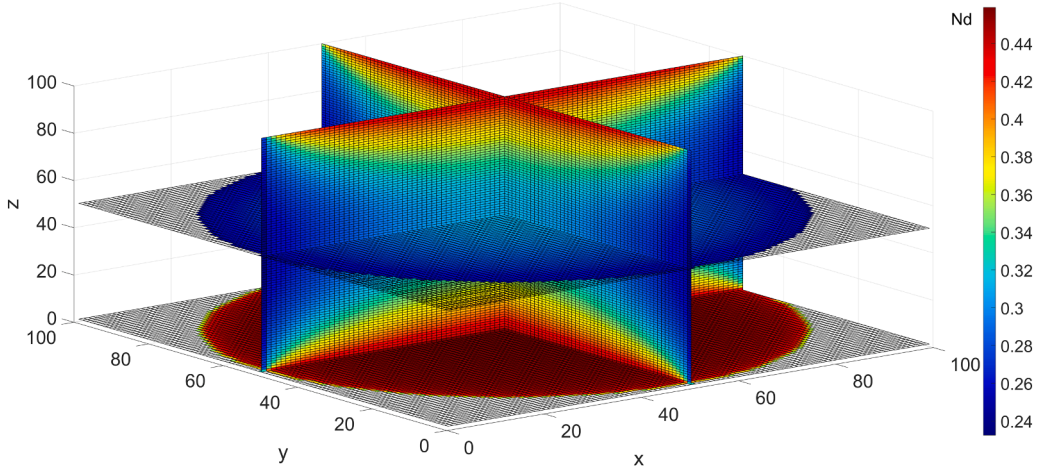


Fig. 20. The spatial distributions of the demagnetisation factors of the constituent particles for a cylindrical sample of diameter and length were both equal to 101 particles.

Table 3
A comparison of the mean demagnetisation factors of cylindrical samples N_z obtained from the model and Eq. (2). Note that the D_z values are obtained from [13], and are for cylinders with aspect ratios that closely match those used in the model (comparing the z-direction size to that of the x y).

Sample size: length of cylinder (z direction)	D_z, demag. factor of overall shape (Chen et al)	N_z, mean demag. factor (Eq. (2))	N_z, mean demag. factor (model)	Ratio of N_z (model over Eq. (2) output)
51	0.47449	0.40570	0.40481	0.9978
101	0.31157	0.32218	0.32065	0.9953
201	0.18186	0.25567	0.25336	0.9910
301	0.12777	0.22794	0.22521	0.9880
401			0.20987	
501	0.079908	0.20340	0.20025	0.9845
601			0.19365	
701			0.18885	
801			0.18520	
901			0.18233	
1001	0.041196	0.18356	0.18002	0.9807
2001	0.020909	0.17315	0.16941	0.9784

demagnetisation field at any point within the volume. This method yields a final single demagnetisation factor that is comparable to the calculation of all particles. The cost is the loss of the full 3D spatial distribution, and thus is a user-dependent choice.

Confidence in the model gained by using these well-known shapes should now allow demagnetisation factors for any shape to be determined, and by using Eq. (2) can be extended to calculate unknown factors for the solid forms of these shapes: a useful result for both experimental and modelling observations.

A major advantage of using a polar model approach is that the equations used, see (3) to (6), reduce to that of geometry alone, provided that the constituent particles have the same ellipsoid shape. By approximating real particles to ellipsoids, the model is material independent and applicable to any scale and shape of the containing vessel.

In addition to the single demagnetisation figure obtained here (the average of the assembly), the model also allows the distribution of the individual particle factors to be mapped spatially over the volume and shows interesting and highly skewed values and patterns towards the surfaces. There is some evidence from our previous experimental work [19] that when highly skewed distributions are expected owing to the containing shape, the easily obtained modal value is closer to the real

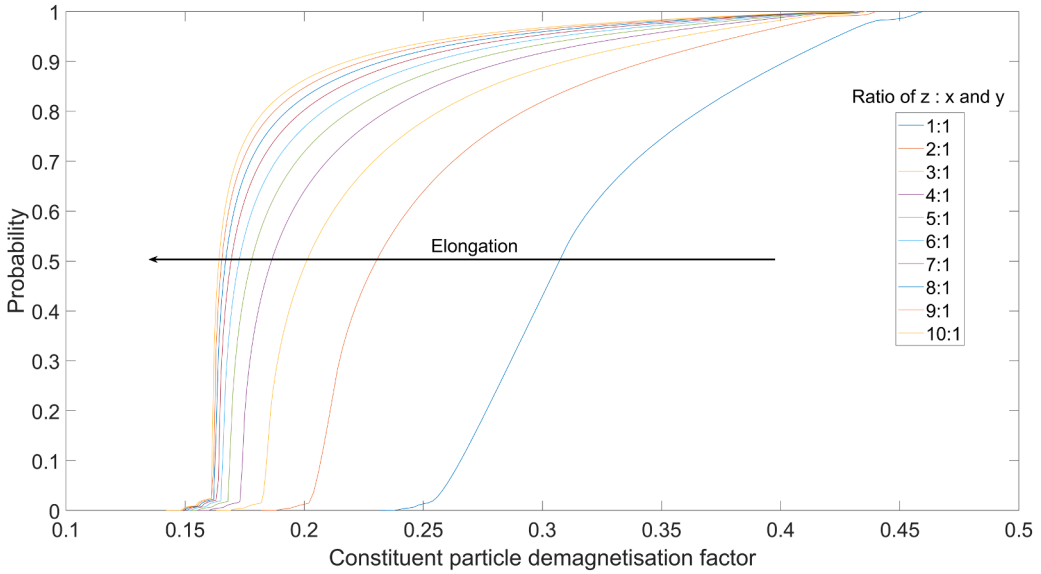


Fig. 21. Cumulative frequency distribution for the cylindrical samples. As the cylinder was elongated, the distributions became sharper and more asymmetrical, as observed for cuboid samples. The ratios given here (z-direction size compared to the x-y size) have been rounded for simplicity.

value of the system. Furthermore, some preliminary work to examine the effects of particle agglomeration in samples of the same container size, shape, and packing fraction resulted in significant changes in the spatial distributions, and as this may affect the experimental observations, it is the subject of further work.

During the development and writing of the algorithms used in this paper, it was our intention to make sure that the coding was as easy to follow as possible and that could be run on an inexpensive non-specialist computer. Using nested for statements within the code was a decision that we made since it seemed more intuitive than using linear indexing [23]: the speed difference being too marginally a gain over readability. We also wanted to keep the coding as such that it was not dependent on hardware setups. In future developments of the algorithms we will be looking at using parallel computing as means of further improving run speed; for example, using "*parfor* loops" (parallel for loops). Since *parfor* loops cannot be nested, this future coding will have to use linear indexing to maximise the full potential of parallelisation [24].

Data availability

The code, performance data,and example mask matrices, are available at: https://github.com/physicssteve/Demagnetisation-Factor.git

CRediT authorship contribution statement

Steven M. McCann: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Conceptualization. **Tim Mercer:** Writing – review & editing, Writing – original draft, Validation, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. An example program for generating ellipsoid masks

To provide evidence that the algorithms that calculate the demagnetisation information for any shape are valid, ellipsoids were selected as the demonstration shape for comparison with data in the literature. The shape of interest is "etched" into a cuboid lattice using "1" and "0" to signify the presence or absence respectively of a particle in the lattice: this is the *Othello* matrix, or mask, used in the "any shape" algorithms. Algorithm 9 is an example program used to create a mask, in this case, an ellipsoid of equal size in the *x*- and *y-directions*, of 101 particles, and with a *z-length* of 901 particles. In this example, the initial lattice consists of touching spherical particles of unity diameter in a simple cubic arrangement. The first part of the program accepts or rejects particles depending on whether their centres lie within the desired ellipsoid shape. The second part examines the particles at the surface of this "rough and ready" ellipsoid mapped out in the first section, and tests to determine whether the particle touches or intersects with the required ellipsoid shape. This is achieved by solving the equation for a given particle with the equation governing the ellipsoid shape and testing to determine whether the solution is purely imaginary (lies fully within the ellipsoid) or has real elements (intersects or touches the ellipsoid).

Algorithm 9
MATLAB 2022b code that generates the mask matrix Othello for an ellipsoid of approximate ratio of 1:1:9 (x, y, z). The first stage of the code is to assess whether a particle's centre lies within the overall ellipsoid container shape. In the second stage, the particles at the surface are tested to confirm that they fully lie within the container shape.

```
Othello = zeros(101,101,901);
  syms x y z
  %rough check: test to see if the centre of the particles lies within the
  %ellipsoid.
  for k = 0.5:900.5
  for j = 0.5:100.5
  for i = 0.5:100.5
  ellip=((i-50.5)/50.5)^2 + ((j-50.5)/50.5)^2 + ((k-450.5)/450.5)^2;
  if ellip < 1
  Othello(i+0.5, j+0.5, k+0.5) = 1;
  end
  end
  end
  end
  %detailed check to test if the full particle resides within the ellipsoid.
  for j = 1:101
  for i = 1:101
  k = 1;
  if sum(Othello(i, j, 1:901)) == 0
  cond = false;
  else
  cond = true;
  end
  while cond == true && k <=451
  if Othello(i, j, k) == 1
  eqns = [((x-50.5)/50.5)^2 + ((y-50.5)/50.5)^2 + ((z-450.5)/450.5)^2 == 1, (x-i-0.5)^2 + (y-j-0.5)^2 + (z-k-0.5)^2 ==
  0.5^2];
  vars = [x y z];
  [X Y Z] = solve(eqns, vars);
  X = double(X);
  testX = isreal(X);
  Y = double(Y);
  testY = isreal(Y);
```

(*continued on next page*)

Algorithm 9 (*continued*)

```
Z = double(Z);
testZ = isreal(Z);
if testX == true && testY == true && testZ == true
Othello(i, j, k) = 0;
Othello(i, j, 901-k+1) = 0;
else
cond = false;
end
end
k = k+1;
end
end
end
clearvars -except Othello
```

Appendix B. An example program to create a 2-D mask for container shapes with uniform cross-section

The masks required for container shapes with a consistent cross-sectional area in the *xy* plane are much simpler to create because only the absence or presence of particles in this plane is required (because it can be duplicated along the *z-axis*. As cylinders are well documented in the literature, this shape was selected for testing, which requires the generation of circular masks. Algorithm 10 is an example of a piece of code used to create such a mask, which is a circular mask with a diameter of 101 particles.

Algorithm 10
MATLAB 2022b code that generates the mask matrix Othello for a circular cross section.

```
Othello = zeros(101);
  for j = 0.5:1:100.5
  for i = 0.5:1:100.5
  d = sqrt((50.5-i)^2 + (50.5-j)^2);

  if d <= 50.5 - 0.5
  Othello(i+0.5, j+0.5) = 1;
  else
  Othello(i+0.5, j+0.5) = 0;
  end
  end
  end
  clearvars -except Othello
```

Appendix C. Comparison of new fast cuboid algorithm with the original

Performance testing was carried out on five cuboid simulations of touching particles. Each simulation had a consistent x- and y-axes length of 101 particles each and a z- axis initially set at 101 particles, and then increased in steps of 100 to a final value of 501. A direct comparison was made between the new fast cuboid algorithm and the original model's simple summation process, Fig. C 1. The new algorithm reduced execution time drastically, shifting the relationship between execution time and simulation size from a quadratic to linear. The new process now takes approximately 10 seconds per million particles in the simulation.

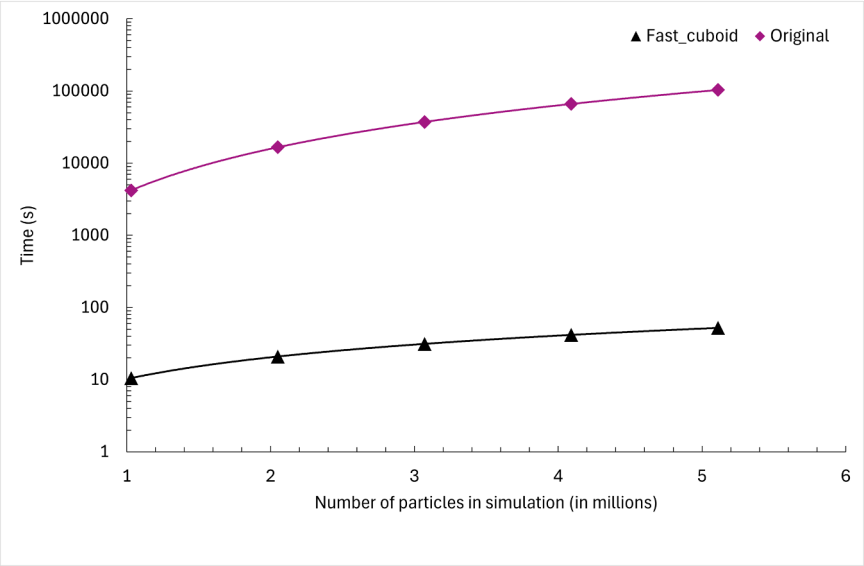


Fig. C. 1: comparison of the new fast cuboid summation script compared to the original model's summation process of a simple sum taken over a subset of the matrix Nzfull for each particle. Note the log scale used on the y- axis.

References

[1] B.D. Cullity, C.D. Graham, Introduction to magnetic materials,, 2nd ed, IEEE/Wiley, Hoboken, N.J, 2009.

[2] C.R.H. Bahl, Estimating the demagnetization factors for regular permanent magnet pieces, AIP Adv. 11 (2021) 075028, https://doi.org/10.1063/5.0060897.

[3] Z. He, C. Liu, Z. Li, Z. Chu, X. Chen, X. Chen, Y. Guo, Advances in the use of nanomaterials for nucleic acid detection in point-of-care testing devices: A review, Front. Bioeng. Biotechnol. 10 (2022) 1020444, https://doi.org/10.3389/fbioe.2022.1020444.

[4] Q. Shen, C. Yu, Advances in superparamagnetic iron oxide nanoparticles modified with branched polyethyleneimine for multimodal imaging, Front. Bioeng. Biotechnol. 11 (2024) 1323316, https://doi.org/10.3389/fbioe.2023.1323316.

[5] R.H. Fang, W. Gao, L. Zhang, Targeting drugs to tumours using cell membrane-coated nanoparticles, Nat. Rev. Clin. Oncol. 20 (2023) 33–48, https://doi.org/10.1038/s41571-022-00699-x.

[6] A.N. Stephen, T. Mercer, W. Stockburn, S.R. Dennison, J.E. Readman, S.M. Reddy, Simple size tuning of magnetic nanoparticles using a microwave solvothermal method and their application in facilitating the solid-phase synthesis of molecularly imprinted polymers, Mater. Adv. 6 (2025) 2016–2028, https://doi.org/10.1039/D4MA01115E.

[7] A. Shakeri-Zadeh, J.W.M. Bulte, Imaging-guided precision hyperthermia with magnetic nanoparticles, Nat. Rev. Bioeng. (2024), https://doi.org/10.1038/s44222-024-00257-3.

[8] M.M. Rajib, N. Bindal, R.K. Raj, B.K. Kaushik, J. Atulasimha, Skyrmion-mediated nonvolatile ternary memory, Sci. Rep. 14 (2024) 17199, https://doi.org/10.1038/s41598-024-66853-w.

[9] J.J.B. Levinsky, B. Beckmann, T. Gottschall, D. Koch, M. Ahmadi, O. Gutfleisch, G. R. Blake, Giant magnetocaloric effect in a rare-earth-free layered coordination polymer at liquid hydrogen temperatures, Nat. Commun. 15 (2024) 8559, https://doi.org/10.1038/s41467-024-52837-x.

[10] S. Praharaj, D. Rout, Magnetic nanoparticles in catalysis industry, Fundam. Ind. Appl. Magn. Nanoparticles, Elsevier (2022) 477–495, https://doi.org/10.1016/B978-0-12-822819-7.00022-3.

[11] J.A. Osborn, Demagnetizing factors of the general ellipsoid, Phys. Rev. 67 (1945) 351–357, https://doi.org/10.1103/PhysRev.67.351.

[12] A. Aharoni, Demagnetizing factors for rectangular ferromagnetic prisms, J. Appl. Phys. 83 (1998) 3432–3434, https://doi.org/10.1063/1.367113.

[13] D.-X. Chen, E. Pardo, A. Sanchez, Fluxmetric and magnetometric demagnetizing factors for cylinders, J. Magn. Magn. Mater. 306 (2006) 135–146, https://doi.org/10.1016/j.jmmm.2006.02.235.

[14] G. Breit, Calculations of the effective permeability and dielectric constant of a powder, Commun. Phys. Lab. Univ. Leiden 46 (1922) 293–308.

[15] B. Bleaney, R.A. Hull, F.A. Lindemann, The effective susceptibility of a paramagnetic powder, Proc. R. Soc. Lond. Ser. Math. Phys. Sci. 178 (1941) 86–92, https://doi.org/10.1098/rspa.1941.0045.

[16] R. Bjørk, Z. Zhou, The demagnetization factor for randomly packed spheroidal particles, J. Magn. Magn. Mater. 476 (2019) 417–422, https://doi.org/10.1016/j.jmmm.2019.01.005.

[17] N. Mosleh, A.R. Insinga, C.R.H. Bahl, R. Bjørk, The magnetic properties of packings of cylinders, J. Magn. Magn. Mater. 607 (2024) 172391, https://doi.org/10.1016/j.jmmm.2024.172391.

[18] COMSOL Multiphysics. www.comsol.com, 2024.

[19] S.M. McCann, J. Leach, S.M. Reddy, T. Mercer, Methods of investigating the demagnetization factors within assemblies of superparamagnetic nanoparticles, AIP Adv. 12 (2022) 075212, https://doi.org/10.1063/5.0095899.

[20] P.R. Bissell, D.A. Parker, G.E. Kay, R.D. Cookson, Validity of the sheet demagnetising factor in characterisation of advanced metal particle tapes, J. Magn. Magn. Mater. 242 (1) (2001) 359–361, https://doi.org/10.1016/S0304-8853(01)01159-3.

[21] R.D. Cookson, Transverse susceptibility studies of recording media, PhD thesis, University of Central Lancashire, 2002.

[22] R.L. Lipsman, J.M. Rosenberg, Multivariable Calculus with MATLAB®, Springer International Publishing, Cham, 2017 https://doi.org/10.1007/978-3-319-65070-8.

[23] S. Attaway, MATLAB: a practical introduction to programming and problem solving, Elsevier, Butterworth-Heinemann, Kidlington, Oxford Cambridge, MA, 2023. Sixth edition.

[24] Y.M. Altman, Accelerating MATLAB® performance: 1001 tips to speed up MATLAB programs, CRC press, Boca Raton, 2015.