The use of morphological algorithms and finite difference methods on 3-dimensional paper coating structures for identifying pores, gathering statistical data and physical constants

T. Byholm[a], **P. Alam**[b], **J. Westerholm**[c], **and M. Toivakka**[d]

[a,b,d]Laboratory of Paper Coating and Converting
Åbo Akademi University, Porthaninkatu 3, FI-20500 Turku

[c]Department of Information Technologies
Åbo Akademi University, Lemminkäisenkatu 14 – 18, FI-20520 Turku

Presented at the 13th International Coating Science and Technology Symposium, September 10-13, 2006, Denver, Colorado[1]

1 Introduction

The optimal set of end-use properties of coated paper is a function of its intended application. The coating structure of the paper controls a wide variety of physical properties such as light scattering, fluid absorption, surface strength and runnability since these are all affected by the microscopic structure of the coating. In this paper we suggest using computer algorithms to analyse these coating structures by dividing the porous media into separate entities called pores. This enables the extraction of statistical data of the void structure including the pore size, pore connectivity and the throat area between pores. Various algorithms for accomplishing this task, have been proposed, mainly different types of thinning algorithms [1,2,3,4] or closely related skeletonisation algorithms, which rely on erosion methodologies. These algorithms often show problems concerning robustness due to digitalisation problems, resulting in unpredictable results as pointed out by the authors of the Maximal Balls algorithm (MBA) [4]. One of the main challenges facing erosion-type algorithms is to ensure that the erosion process is proceeding with equal speed in all directions. Voxels diagonally offset from a central voxel represent a larger distance than voxels that are horizontally or vertically offset from the centre point. The MBA and skeletonisation algorithms use the same principles based on spherical volumes and show promising results regarding these problems [4]. It is for this reason that the MBA was chosen as a starting point. We suggest improvements to the original MBA [4] in order to increase efficiency on large scale problems and to customise it for paper coating requirements. We take advantage of some fundamental geometrical properties for avoiding calculations and to enable pre-calculation of data. We suggest new data representations for introducing significant memory optimisations by flattening hierarchical structures essential to the algorithm. Moreover we introduce a straightforward solution for the removal of false pore maxima. Computational methods for calculating important physical properties of coatings are also suggested in this paper. More specifically, the tortuosity and permeability are calculated using packings generated using random particle packing algorithms.

2 The Maximal Balls algorithm

The basic methodology of the Maximal Balls algorithm (MBA) will be presented in this section. More detail can be found in [4]. Analysing porous structures statistically requires a division of the porous structure into separate entities. Essentially it is important to distinguish between two separate types of entities, the pore bodies and the throats or necks between the pore bodies. The original definition of the MBA describes pores as being the larger bodies in the porous media with the main function of storing liquid, whereas the throats are referred to as the smaller 3-dimensional bodies forming connections between pores. In this paper we keep the definition of a pore but throats are redefined as an area describing the minimum of the hydraulic radius between two pore entities.

Basically, the MBA creates a hierarchical structure of so called maximal balls. A maximal ball is calculated for each point (x,y,z) inside a pore of a porous media finding the maximum size of a sphere centred at the point, which is completely inside the porous volume. This is analogous to inflating rubber balloons in hollow space until they touch the surrounding wall, without allowing the balloon to deform reversibly. Furthermore, a maximal ball must not be inscribed in another ball. These balls are removed, since they do not describe anything outside their master. The remaining maximal balls, which completely fill the porous media, are ordered hierarchically according to their radius as compared with neighbouring balls. A ball is considered hierarchically lower if its centre point is contained in a ball with a larger radius. When the algorithm has refined the hierarchical structure, three

types of maximal balls remain. Balls belonging to the first group have no masters and define a local maxima . The second group comprises both masters and slaves that define how a pore shrinks towards a corner or a throat. Balls belonging to the last group have two or more masters, indicating that they reside in a throat. This information is used to identify how the pore space is divided into separate entities.

2.1 Dividing the pore space

In this section most effort will go into describing how our implementation differs from the original MBA. Simple cubic packings and hexagonal close-packed sets were used for validation. The space that we are analysing is represented as a voxel matrix. A voxel is a cubic volume that represents one sample point of our digitalised material. A voxel can either be a solid phase voxel, representing a volume belonging to the solid material or a void phase voxel, which represents the empty space between solid particles.

2.2 The creation of the maximal balls

The first part of the algorithm generates maximal balls for every voxel belonging to the pore space. The most straightforward method for accomplishing this task is a stepwise increase of the ball size, ensuring that there does not exist any position that overlap any of the solid phase before continuing to the next step. To increase the efficiency of this procedure it is recommended that a set of different sized balls be generated beforehand, e.g. from radius 1 to 10. These balls must be generated on demand and stored for later use. In order to optimise this process, coordinates belonging to a ball are stored in a simple array and only one quadrant of the ball is generated at first. The other seven quadrants are created by mirroring the original quadrant with respect to the centre point. The coordinates from different quadrants are interleaved for increasing the possibility of an early conflict. Nevertheless, interleaving will not help us to any greater extent when using an algorithm that increments the radius by one and rechecks for collision. We would only benefit from it if the size is checked where the first collision occurs. It is preferable to use a smarter algorithm that makes use of the well known binary search algorithm [5]. Radii are not checked in order, but according to the binary search pattern, thus we will find the correct radius in O(log n) time where n is an estimation of the largest radius present in the medium.

The shape of the balls used will affect how the hierarchy of maximal balls are connected. The easiest way to shape the sphere is to define voxels within the radius of the sphere to belong to its domain, but this results in an uneven shape, especially in the outer corners of the quadrants. Instead we calculate the volume fraction of all voxels that are inside the sphere, using sub-pixel division [6]. Every voxel of the border area is divided into a pre-defined set of sub voxels, and the number of voxels included is used for defining the estimated volume included. A threshold is then applied for deciding how big a part has to be included before the voxel is added to the sphere. The end result is a smoother, more sphere-like shape.

2.3 Elimination of inscribed maximal balls

After all maximal balls have been calculated, the next step is to remove balls that are completely inscribed in larger balls, since they repeat already known information. To achieve this, the volume of all maximal balls in the set is sought through. For every voxel, its corresponding maximal ball will be removed if all voxels in its domain are inside the larger ball. We use a simple approach, with only one representation for the maximal ball radii, a 3-dimensional matrix, where voxels belonging to particles or void space are defined with separate id's and maximal balls are identifed with a number representing their corresponding radius.

When eliminating an inscribed maximal ball, the most straightforward method would be to ensure that all voxels belonging to the inscribed ball are inside the domain of the larger ball. It is easy to add some effective optimisations at this point. Firstly, only the quadrant where the inscribed ball was found needs to be checked. If this quadrant is included then all other quadrants will also be included. Furthermore, only the outermost shell of the inscribed ball needs to be checked, the inner layers being closer to the centre.

Nevertheless, pre-calculation is the most useful tool. Our problem is limited to the size of the larger ball, the size of the inscribed ball and their positions relative to each other. The basic idea is to create a large array where for every ball size up to a certain limit, all possible positions and sizes of inscribed balls are calculated. Every position in the array defines whether it is a valid position or not, in other words it tells us if a ball of size y would be completely inscribed if it were positioned at (a,b,c). In the end, this gives us a radical improvement in speed, since we only need to do an array lookup with complexity O(1) to check if a ball is included, instead of using an $O(n^2)$ algorithm for testing it manually. Manual testing requires testing all voxels *a* in the master sphere for collision with all voxels *b* in the slave sphere, which gives us $O(a \times b)$, that is , essentially $O(n^2)$.

2.4 Elimination of problems in the hierarchy

The original implementation [4] does not mention any specific problems in the hierarchy, which may be due to the reasonably low porosities that were analysed, typically below 20%. In our structures the porosities calculated are often around 30-50% and we found related problems that need to be dealt with. These high porosities introduced the possibility for the so called false local maxima. They

are often formed in narrow throats which are reasonably long, creating a row of maximal balls of equal size. This results in the following problem; the algorithm never adds balls of the same size to the hierarchy, but rather, it makes these equally sized balls form their own local maxima. The easiest solution to this challenge seems to be to use the information gathered in the hierarchical structure to remove the false maxima. All the maximal balls are scanned through and those masters without any masters higher up in the hierarchy are identified. If a maximal ball is found in its domain with an equal radius and subordination under another ball, it means that the ball should be classified as a slave under it. When scanning through once, the false maxima closest to the normal structure will be included, and the same procedure will be repeated until there are no more false maxima to be added.

2.5 Reduction of memory consumption

The application of hierarchy is straightforward, but we also have to consider how to minimise the memory usage. It is difficult to predict the memory consumption because it varies significantly depending on the porous structure. Typically a $(200)^3$ matrix with a particle concentration of around 60% might easily use up to 1GB of memory. There are basically two important observations that can be made. Firstly, the authors of the maximal balls algorithm suggest the use of objects for representing the hierarchical structure. Code readability is enhanced while increasing the memory usage but simple arrays would be preferable. The initialisation of the hierarchy is done by assigning balls inside their domain with a smaller radius as their slaves. Later, the hierarchy is refined, making balls highest up in the hierarchy the only masters of their subordinate balls. This removes significant amounts of master-slave relations, eliminating multiple references to the same ball. Moreover, if we succeed in flattening the hierarchy from the beginning, it will be much easier to represent the hierarchical structure. An array map the same size as the one representing the size of the maximal balls could be used. This map contains the identification of the maximal ball that it belongs to, handling the few balls with multiple masters separately.

To apply the memory optimisation, we need to identify all the pores that are local maxima, in other words, those that reside in the centre of a pore. This process is easy to undertake when removing inscribed balls. When removing all smaller balls inscribed in a ball, we assign negative radii to those balls that are not removed but have a smaller radius. Thus, only balls with a locally maximal radius (no neighbouring sphere with a larger radius exists), will have positive radii. They can be used as a starting point for creating the hierarchy. We select one local maximum at a time and then recursively add their slaves to their slave list.

3 Computational methods for calculating physical properties

Fluid flow simulations are undertaken in computational fluid dynamics software FLOW3D. This software is built upon finite difference methods for solving partial differential equations related to fluid flow problems. The Navier-Stokes equations for incompressible Newtonian fluids were used to define the characteristics of flow, Equations 1 and 2. In these equations, ρ is the density, η is the viscosity, p is the pressure, $\mathbf{F}$ is the volume force field, u is a directional velocity and t is the time.

$$\rho\frac{\partial u}{\partial t}-\eta\nabla^2\mathbf{u}+\rho\left(\mathbf{u}\cdot\nabla\right)\mathbf{u}+\nabla p=\mathbf{F} \quad (1)$$

$$\nabla\cdot\mathbf{u}=0 \quad (2)$$

Dimensionless tracers are used to follow the tortuous flow of the fluid material. Distances are output in orthogonal Cartesian axes (x, y, z) as a function of time. The z axis in this case represents the thickness direction of the coating material. The distances output were converted to discrete linear Euclidean spatial distances and then to the total distance travelled through the coating, L_e, as shown in Equation 3. The tortuosity, τ, is then calculated by dividing the total distance travelled through the coating by the linear distance travelled through the coating thickness, L, in parallel with the z Cartesian axis, Equation 4.

$$L_e\left(\mathrm{t}\right)=\sum_i\sqrt{\delta x_i^2\left(\mathrm{t}_i\right)+\delta y_i^2\left(\mathrm{t}_i\right)+\delta z_i^2\left(\mathrm{t}_i\right)} \quad (3)$$

$$\tau\left(\mathrm{t}\right)=\frac{L_e\left(\mathrm{t}\right)}{L\left(\mathrm{t}\right)} \quad (4)$$

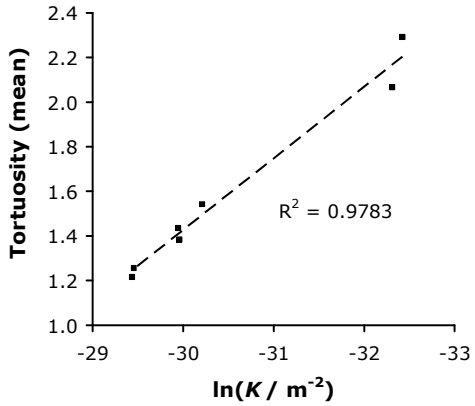

Fig.1 Tortuosity plotted against ln(K)

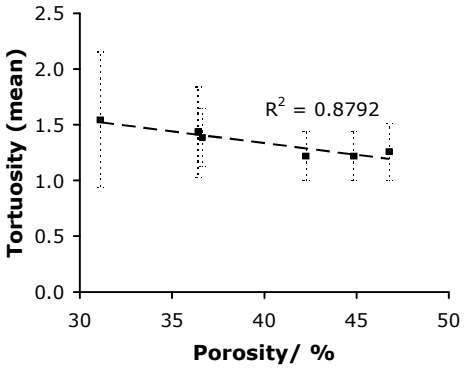

Fig. 2 Tortuosity as a function of porosity

Permeability, K, is calculated using D'Arcy's law, Equation 5, where where μ is the fluid viscosity, A is the packing area cross section, l is the distance through the packing, q is the flux and ΔP is the pressure difference between the top and the bottom of the packing.

$$K = \frac{q \cdot \mu \cdot l}{A \cdot \Delta P} \quad (5)$$

Values of tortuosity and ln(K) are subsequently plotted in Figure 1 for packings comprised of different plate-like geometries. A high coefficient of determination R^2 exists between these variables relative to a linear regression fit. In Figures 2 and 3 respectively, the packing tortuosity and ln(K) are plotted as a function of porosity values acquired from the MBA. It is expected that both tortuosity and permeability decrease as a function of increasing pore space, both of which are noticeable in Figures 2 and 3 respectively.

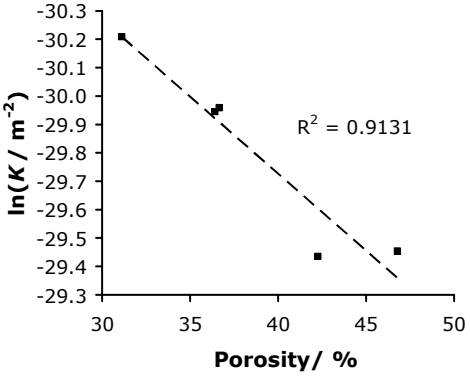

Fig. 3 ln(K) as a function of porosity

4 Conclusions

As was shown in [4], the Maximal Balls algorithm is a robust tool for extracting vital information from porous structures. Its capability to store the complete porous structure in the hierarchy of maximal balls helps us to enhance the stability of the algorithm. We have added straightforward extensions to the algorithm that enable the removal of false maxima and the identification of the throat as an area. This will increase further the robustness of the algorithm. Furthermore, improvements in the speed of the algorithm were found using different optimisation techniques, such as pre-calculation of data and reductions of calculations. Most improvements show a radical reduction in algorithmic complexity, from exponential to constant time in the most extreme case. A suggestion was made on how memory consumption could be considerably reduced through a flattening of the MB hierarchy. Pore structures in porous media are related to physical properties such as tortuosity and permeability. We have shown that these properties can be calculated computationally and related directly to porosity values acquired from the MBA. Our understanding of porous media will be further advanced by correlating physical properties to statistical data generated by the MBA in relation to the pore and neck space characteristics.

The greatest challenges regarding the algorithm is the management of resources, both memory and computational power. The extreme amounts of data that would be needed to realistically capture the properties of a poly-disperse particle packing with large particle size distributions give rise to the need for processing considerable amounts of data. It is interesting to see that these kinds of algorithms can be used in very diverse fields of science, including petroleum science, medicine and paper science.

References:

[1] M. Toivakka and K. Nyfors. Pore space characterization of coating layers. *TAPPI Journal 2001 Vol. 84(3)*

[2] Al-Raoush and R. Ibrahim. Extraction of Physically-Realistic Pore Network Properties from Three-Dimensional Synchrotron Microtomography Images of Unconsolidated Porus Media. *Louisina State Univerity 2002*

[3] L. Lam, S-W Lee and C.Y. Suen, Thinning Methodologies – A Comprehensive Survey, IEEE *Transactions on Pattern Analysis and Machine Intelligence, Vol 14, No 9, Sept 1992*

[4] D.Silin and G. Jin, T.Patzek, Robust Determination of the Pore Space Morphology in Sedimentary Rocks. *SPE 84296, ATCE, Denver 2003*

[5] Robert Sedgewick, Algorithms in C++ Third Edition*, Addison-Wesley 1998*

[6] J.D. Foley, A. van Dam, S.K. Feiner and J.F. Hughes, Computer Graphics: Principles and Practice in C 2nd Edition, *Addison-Wesley 1996*