

ISSN 1360-1725



**Simulation of grain boundary diffusion creep: analysis of some
new numerical techniques**

Judith M. Ford, Neville J. Ford and John Wheeler

Numerical Analysis Report No. 435

Manchester Centre for Computational Mathematics
Numerical Analysis Reports

DEPARTMENTS OF MATHEMATICS

Reports available from: And over the World-Wide Web from URLs
Department of Mathematics <http://www.ma.man.ac.uk/MCCM/MCCM.html>
University of Manchester <ftp://ftp.ma.man.ac.uk/pub/narep>
Manchester M13 9PL
England

Simulation of grain boundary diffusion creep: analysis of some new numerical techniques*

Judith M. Ford[†] Neville J. Ford[‡] and John Wheeler[§]

October 21, 2003

Abstract

We consider the simulation of deformation of polycrystalline materials by grain boundary diffusion creep. For a given network of grain boundaries intersecting at nodes, with appropriate boundary conditions, we can calculate the rate at which material will be dissolved or deposited along each grain boundary and hence predict the rate at which each grain will move to accommodate this dissolution/deposition. We discuss two numerical methods for simulating the network changes over a finite time interval, based on using the movement of adjacent grain boundaries over a small time interval to estimate the velocities of the nodes. (The second of these methods has enabled us to speed up solution by 100 times in typical experiments compared with a naïve forward-Euler approach.) We show that the accuracy with which the node velocities can be estimated is dependent only on the precision of the machine with which they are computed and deduce that, for all practical purposes, the lack of precise node velocity values does not detract from the quality of our solution. Finally, we consider the underlying stability of the problem under various different boundary conditions and conclude that our methods have the potential for providing useful insight into the effect of grain size and shape on deformation in polycrystalline materials.

Keywords: diffusion creep, microstructure, grain boundaries, stability, predictor-corrector, Euler’s method, Runge-Kutta methods

1 Introduction

Crystalline materials have a microstructure of “grains”, separated from one another by “grain boundaries”. The precise nature of these differs from one material to another and also varies according to the environmental conditions. In this article we use a 2-dimensional model in which each grain is considered to be a rigid

*This work was supported by NERC Research Grant NER/B/S/2000/000667

[†]j.ford@umist.ac.uk Department of Mathematics, UMIST.

[‡]njford@chester.ac.uk Department of Mathematics, University College Chester.

[§]johnwh@liv.ac.uk Department of Earth and Ocean Sciences, University of Liverpool.

body separated from the adjoining grains by infinitesimally narrow straight grain boundaries.

Coble creep (see [3]) occurs when external stresses cause matter to be transported by grain-boundary diffusion from boundaries under compression to boundaries under tension. As a result, individual grains tend to elongate producing macroscopic strains. Pressure solution creep, where material diffuses along fluid-filled grain-boundaries is very similar and can be modelled using the same equations (see [5, 13]). Both Coble creep and pressure solution creep are important mechanisms in rock deformation (see [4, 5, 11, 16]) and Coble creep is also a factor determining the strength of metals, ceramics and other materials (see [2]). Simulation of grain-boundary diffusion creep can provide useful insights into how the microstructure of a material affects its behaviour under stress. Hazzledine & Schneibel [9] derived a system of equations relating the stress distribution and *initial* grain velocities to the network configuration and boundary conditions. More recently, Kim & Hiraga [10] used a very much simplified model, in which the stress is assumed constant along each grain boundary, to simulate deformation over time. Ford *et al.* [6] extended the Hazzledine & Schneibel model to include a greater variety of boundary conditions and introduced a time-stepping algorithm to enable deformation to be simulated over time more realistically than using the Kim & Hiraga model. Here we present two numerical methods for implementing the time-stepping procedure and analyse them mathematically.

The paper is structured as follows. We describe the mathematical model in §2 and show how it can be used to define a differential equation describing the node movements over time in §3. In §4 we present numerical schemes for performing the simulation and analyse their convergence properties. The performance of these methods is illustrated by example results presented in §§5 and 6. In §7 we discuss the stability of the underlying problem under various types of boundary conditions. We conclude with a discussion of some possible future developments.

2 Defining the problem

The flux in atoms/m²/s along a grain boundary is given (see, for example, [14]) by

$$J(x) = -\frac{D}{kT} \frac{\partial \sigma_n(x)}{\partial x}, \quad (2.1)$$

where D is the grain-boundary self-diffusivity, k is Boltzmann's constant, T is the absolute temperature, and $\sigma_n(x)$ is normal stress along the boundary, x being the distance along the boundary from a given point. (Compressive stresses are taken to be positive and tensile stresses negative.)

Consider a polycrystal (such as that illustrated in figure 2.1) made up of a pre-defined number of grains with straight grain boundaries meeting at triple-points (3-nodes) and double-points (2-nodes). The configuration of the network at a given

time can be defined by the positions of the nodes and topological information about which nodes are joined to which others.

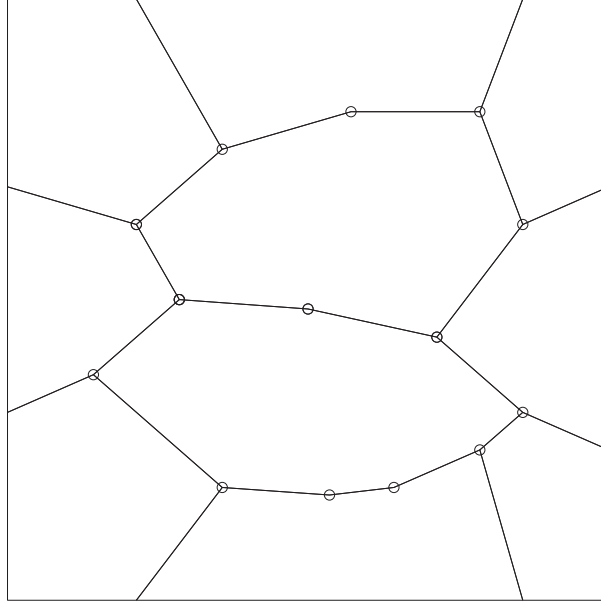


Figure 2.1: 10-grain polycrystal

We define the following constants relating to the network of grain boundaries:

- n_g number of grains
- n_{gb} number of grain boundaries
- n_t number of triple-points (3-nodes)
- n_d number of double-points (2-nodes)
- n_s number of grain-boundary/surface intersections (1-nodes)

With this notation, we have the useful relationship (see e.g. [1])

$$3n_t + 2n_d + n_s = 2n_{gb}. \quad (2.2)$$

We make the following assumptions (see [9, 6]):

1. The grains are rigid.
2. The grain boundaries do not support shear stress.
3. No cracks can open up between grains and grains cannot overlap each other.
4. The insertion or removal of matter at nodes is a second-order effect which can be ignored.
5. Matter is deposited/dissolved equally on both grains adjacent to a grain-boundary.

From equation (2.1) we can deduce that the rate, measured as a velocity, at which material is plated out at the grain-boundary is

$$r(x) = -\frac{\partial J}{\partial x} = \frac{Dw\Omega}{kT} \frac{\partial^2 \sigma_n(x)}{\partial x^2}, \quad (2.3)$$

where w is the grain-boundary width and Ω is the atomic volume. (Dissolution can be thought of as negative plating.) Under the assumptions that the grains are rigid and that no voids or overlaps are formed, the plating-rate must be a linear function of x (see [9]). Hence the stress along grain boundary k is a cubic function of x and can be described in terms of four coefficients:

$$\sigma_n^{(k)} = \alpha_0^{(k)} + \alpha_1^{(k)}x + \alpha_2^{(k)}x^2 + \alpha_3^{(k)}x^3. \quad (2.4)$$

As material is plated out or dissolved at the grain boundaries, the grains will be forced to move in order to accommodate this. We can define the movement of grain j in terms of its horizontal and vertical velocities $w_1^{(j)}$, $w_2^{(j)}$ and its angular velocity $\Omega^{(j)}$, measured about a fixed origin. If the grain velocities and plating rates are known, the velocities of the grain boundaries can be deduced. Hence the motion of the grain boundaries is completely determined by the $4n_{gb}$ stress coefficients and the $3n_g$ grain velocities. These can be computed for any given grain boundary network and set of boundary conditions by solving a system of linear equations. The detailed derivation of this system can be found in [6]. A similar system for a more limited choice of boundary conditions is derived in [9]. For completeness, we include a summary here.

2.1 Conditions at grain-boundary intersections

At the point where two or more grain boundaries meet, the stress must be continuous, since otherwise there would be infinite fluxes at this point. So at the intersection of boundaries k , l , m we have

$$\sigma_n^{(k)} = \sigma_n^{(l)} = \sigma_n^{(m)}. \quad (2.5)$$

The net flux flowing out of each intersection must be zero, to prevent void formation:

$$J^{(k)} + J^{(l)} + J^{(m)} = 0. \quad (2.6)$$

At nodes where a grain-boundary meets the surface of the structure we impose boundary conditions specifying *either* the flux *or* the stress. For example, if the system is considered to be closed, we would impose a zero flux condition at boundary nodes. In total, from these conditions we have 3 equations to be satisfied for each triple-junction, 2 for each 2-node and one for each surface intersection, giving a total of $3n_t + 2n_d + n_s = 2n_{gb}$.

2.2 Mechanical equilibrium of the grains

We require each of the n_g grains to be in mechanical equilibrium, giving three conditions on the stress coefficients of the grain boundaries surrounding each grain, a total of $3n_g$ equations. Grains that are not entirely enclosed within the polycrystal will also have forces acting on their external faces. These forces are specified as boundary conditions.

2.3 Plating-rate compatibility

We assume that each grain is a rigid body so that its motion can be described in terms of a translation velocity and an angular velocity. Figure 2.2 shows two grains that have moved apart because material has been plated out on to their surfaces at the boundary between them. Sliding along the grain boundary has also occurred to accommodate deposition/dissolution along other grain boundaries. The assumption that there are no voids or overlaps between the grains means that the line defining the new grain boundary (the dotted line in the figure) must be the same, whether this is based on the movement of grain *A* and the plating-rate on to the boundary of *A* or on the movement and plating of grain *B*. This gives two equations relating the stress coefficients for each grain boundary to the velocity components of the two adjacent grains, a total of $2n_{gb}$ equations.

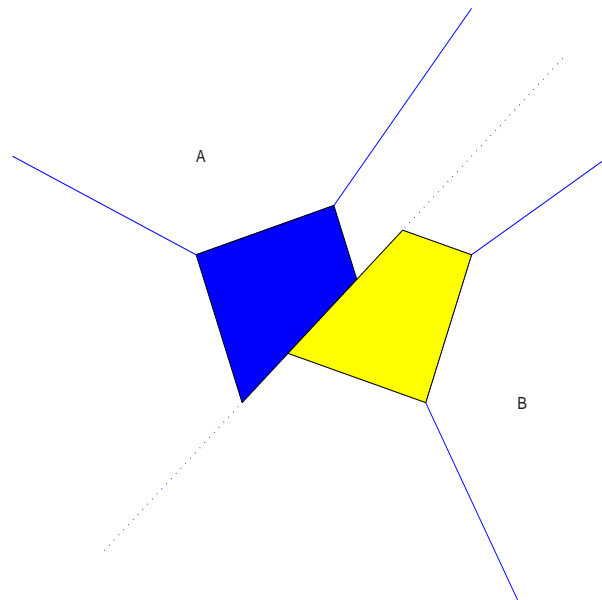


Figure 2.2: Motion of the boundary between two grains. Shading indicates where plating has filled gaps between the grains.

3 Formulation of the problem as a differential equation

Given any network configuration we have a system of $4n_{gb} + 3n_g$ equations (see §2(2.1), (2.2) & (2.3)) in the same number of unknowns, which can be solved to give the stress coefficients for each grain boundary and the velocity components for each grain. Once the stress distribution is known, the precipitation rate of material on either side of each grain boundary (see figure 2.2) can be evaluated using (2.3) and hence the velocity (and angular velocity) of each grain boundary can be determined. As the grain boundaries move, the stress distribution changes to take account of the new geometry. We wish to simulate deformation over a period of time by repeatedly determining the stress distribution of the current configuration and then updating the positions of the grain boundaries. To do this we must determine how the nodes move.

3.1 Specifying the node movements

We can think of our task as being the solution of a (rather complicated) first order ordinary differential equation of the form

$$y'(t) = f(y), \quad y(0) = y_0, \quad (3.1)$$

where y is a vector containing the co-ordinates of the nodes and f is some function, which cannot be written down explicitly but depends on the relationship between the node positions, the stress distribution and the resultant grain boundary movements.

To obtain the node velocities we need to consider how the motion of the grain boundaries will affect the nodes. Figure 3.1 illustrates the effect of deposition of material along grain boundaries. Here three grains move apart as material is deposited along the boundaries between them. The new grain boundary positions are marked by dotted lines. They no longer meet at a single point, but their intersections form a small triangle. An appropriate choice for our new node position is a point that is equidistant from the three boundaries. This criterion is satisfied by the centre of the incircle of the triangle defined by the grain boundary intersections. If two of the grain boundaries are parallel the position midway between the two parallel boundaries and lying on the third boundary is chosen. In the case of 2-nodes the node is placed at the point of intersection of the new grain boundaries (or, if they are parallel, at the point midway between the points on each new boundary corresponding to the original node).

This effectively provides us with a definition of our function f :

$$f(y(t)) = \lim_{\epsilon \rightarrow 0} \frac{\hat{y}_\epsilon(t) - y(t)}{\epsilon}, \quad (\text{if the limit exists}) \quad (3.2)$$

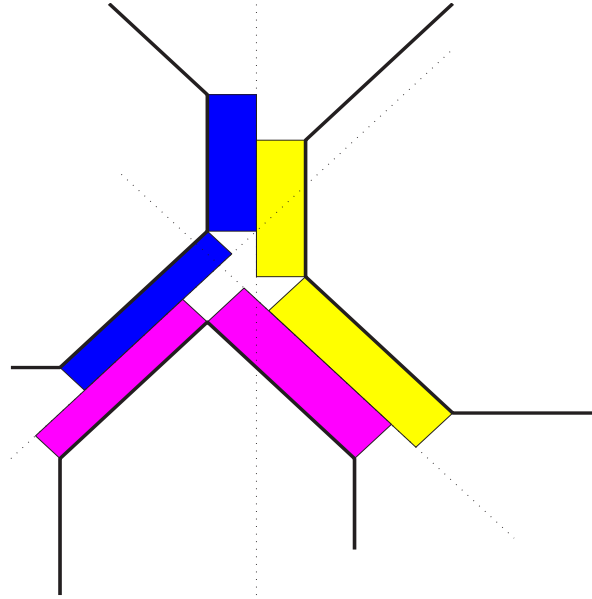


Figure 3.1: 3-node after grain boundary movements.

where, provided that $y(t)$ is known, the approximation $\hat{y}_\epsilon(t)$ to $y(t + \epsilon)$ can be computed by the following process:

Algorithm 3.1 *Estimation of node positions after time increment ϵ .*

1. compute the current stress distribution;
2. use this to find the current grain velocities and deposition/dissolution rates;
3. find the grain boundary positions after time ϵ , based on the computed velocities;
4. choose the new node positions by finding the centre of the incircle of the triangle of grain boundary intersections for each node, as described above.

4 Two numerical solution methods

There are many numerical methods available for solving equations of the form (3.1), but we are restricted in our choice by the fact that we do not have an explicit way of computing $f(y)$. Instead we work with the approximation

$$f_\epsilon(y(t)) = \frac{\hat{y}_\epsilon(t) - y(t)}{\epsilon}. \quad (4.1)$$

The accuracy of our final solution will depend on the accuracy with which $f_\epsilon(y(t))$ approximates $f(y(t))$. We now present some results that show that, by choosing ϵ to be sufficiently small, we can approximate $f(y(t))$ as accurately as we require.

Lemma 4.1 *For any network configuration, $y(t)$, and any $n \in \mathbb{N}$, there exists $\epsilon_{n,y} > 0$ such that $g(s) = \hat{y}_s(t) - y(t)$ is n times differentiable within the domain $D_{n,y} = \{s : |s| < \epsilon_{n,y}\}$.*

Proof In the interests of brevity, we present only an outline proof. We first show that, for each individual node n_i , there exists $\epsilon_{n,y,i} > 0$ such that the mapping g_i , from n_i to the new node position $n_i^{(s)}$ after a time step s , is differentiable n times within the domain $D_{n,y,i} = \{s : |s| < \epsilon_{n,y,i}\}$. It then follows that, within the domain $D_{n,y} = \{s : |s| < \min_i(\epsilon_{n,y,i})\}$, $g(s) = \hat{y}_s(t) - y(t)$ is n times differentiable.

Consider the i th node, n_i of the network. The three grain boundaries L_1, L_2, L_3 meeting at n_i are completely determined by the position vector u of the node and the vectors v_1, v_2, v_3 defining the directions of the grain boundaries. After any time step s , these boundaries are each translated and rotated by multiples of s . Hence the mapping $L_j \rightarrow L_j^{(s)}$ is clearly differentiable (with respect to s) n times for any $n \in \mathbb{N}$. The new node position $n_i^{(s)}$ is a function of the grain boundaries $L_1^{(s)}, L_2^{(s)}, L_3^{(s)}$. This function \mathcal{G}_i is smooth, with respect to the components of the $L_j^{(s)}$, *except* at any point where two of the grain boundaries rotate to become parallel without coinciding. (At such a point the centre of the incircle “jumps” across from one side of the third boundary to the other.) Such a point can only occur for values of s at a finite distance away from zero. (At $s = 0$ parallel boundaries must be coincident.) Hence, by choosing $s_{n,y,i}$ to be sufficiently small, we can ensure that \mathcal{G}_i is n times differentiable, with respect to the components of the $L_j^{(s)}$, within $D_{n,y,i} = \{s : |s| < \epsilon_{n,y,i}\}$. Hence, by the chain rule, g_i is differentiable n times with respect to s for $s \in D_{n,y,i}$. It follows that $g(s)$ is n times differentiable within $D_{n,y}$. \square

Proposition 4.1 $f_\epsilon(y(t)) = f(y(t)) + \mathcal{O}(\epsilon)$.

proof Let $g(\epsilon) = \hat{y}_\epsilon(t) - y(t)$. Provided that g is at least three times differentiable in a domain about 0,

$$\hat{y}_\epsilon(t) = y(t) + g(\epsilon) = y(t) + g(0) + \epsilon g'(0) + \epsilon^2 g''(0) + \mathcal{O}(\epsilon^3),$$

which (since $g(0) = 0$) implies that

$$f(y(t)) = \lim_{\epsilon \rightarrow 0} \frac{\hat{y}_\epsilon(t) - y(t)}{\epsilon} = g'(0).$$

By definition,

$$f_\epsilon(y(t)) = \frac{g(\epsilon)}{\epsilon} = \frac{g(0) + \epsilon g'(0) + \epsilon^2 g''(0) + \mathcal{O}(\epsilon^3)}{\epsilon}.$$

Hence

$$f_\epsilon(y(t)) - f(y(t)) = \mathcal{O}(\epsilon).$$

From lemma 4.1 there exists a domain $D_{3,y}$ about $\epsilon = 0$ within which $g(\epsilon)$ is three times differentiable. Hence $f_\epsilon(y(t)) = f(y(t)) + \mathcal{O}(\epsilon)$. \square

Remark Our definition of $\hat{y}_\epsilon(t)$ is not the only one that satisfies the conditions of proposition 4.1. However, it is noteworthy that the seemingly obvious choice of the centroid of the triangle of grain boundary intersections does *not* satisfy lemma 4.1 because the mapping $\mathcal{G}_i(L_1^{(s)}, L_2^{(s)}, L_3^{(s)})$ is discontinuous whenever two of the grain boundaries rotate to become parallel (even if they then coincide). If used, this method produces large inaccuracies in the positions of certain nodes over even short time intervals.

We now look in more detail at two numerical methods which we have implemented to simulate grain boundary creep.

4.1 The forward Euler method

One of the simplest numerical methods for solving (3.1) is the forward Euler method and its ease of implementation makes it attractive for simulations of the type with which we are concerned here. To find $y_N = y(Nh)$ we compute $y_n \approx y(nh)$, $n = 1, 2, \dots, N$ using the recurrence relation

$$y_{n+1} = y_n + hf(y_n). \quad (4.2)$$

This method yields a solution that converges to the true solution as $h \rightarrow 0$, but the relatively small *region of absolute stability* (see, for example, [12]) of the Euler method means that a very small h value may be required in order to obtain a solution that accurately reflects the true solution behaviour. Moreover, using finite-precision arithmetic, the round-off errors produced at each step accumulate in such a way that there is a threshold below which reducing the value of h actually *increases* the error in the solution (see [12, §5.4-1]).

Since we can only *estimate* $f(y_n)$, our solution using forward Euler will be given by

$$y_{n+1} = y_n + hf_\epsilon(y_n), \quad (4.3)$$

where $f_\epsilon(y_n) = \frac{\hat{y}_n - y_n}{\epsilon}$ and \hat{y}_n is computed using algorithm 3.1. As we will show in §(4.3), using this approximation, the forward Euler method gives $\mathcal{O}(h)$ convergence provided that $f_\epsilon(y_n) - f(y_n) = \mathcal{O}(h)$, which, in view of proposition 4.1, is equivalent to a requirement that $\epsilon = \mathcal{O}(h)$.

Remark. If we choose $\epsilon = h$, we do not need to compute $f_\epsilon(y_n)$ explicitly since the new node positions computed in order to find $f_\epsilon(y_n)$ are the same as those given by (4.3).

4.2 A predictor-corrector method

We wish to be able to perform our simulation over a long time period, so we seek a method which will give reasonable accuracy over a longer time interval than is possible using forward Euler. We are restricted in our choice of method by the lack of an explicit definition for $f(y)$, which makes the use of implicit solution methods

difficult. For this reason we adopt a PECE (**P**redict **E**valuate **C**orrect **E**valuate) method based on a forward Euler predictor and a trapezoidal rule corrector (in other words, a one-step Adams-Bashforth-Moulton scheme, cf. [12, §5.5-2]). The trapezoidal corrector is particularly attractive because of its good stability properties (see, for example, [7, 8]). Our chosen PECE method computes y_{n+1} by the following steps:

1. Use forward Euler to “predict” $y_{n+1}^{(0)} = y_n + hf_\epsilon(y_n)$.
2. Compute $f_\epsilon(y_{n+1}^{(0)})$.
3. Use trapezium rule to “correct”: $y_{n+1}^{(i)} = y_n + \frac{h}{2} (f_\epsilon(y_n) + f_\epsilon(y_{n+1}^{(i-1)}))$, repeatedly until $|y_{n+1}^{(i)} - y_{n+1}^{(i-1)}|$ is sufficiently small.
4. Set $y_{n+1} = y_{n+1}^{(i)}$.
5. Compute $f_\epsilon(y_{n+1})$.

It can be shown (see §4.3) that this method has $\mathcal{O}(h^2)$ convergence, provided that $f_\epsilon(y_n) - f(y_n) = \mathcal{O}(h^2)$. In view of proposition 4.1, this is equivalent to a requirement that $\epsilon = \mathcal{O}(h^2)$. This means that this method has the potential to give improved accuracy without the need to increase the number of time steps used for simulation over a given interval, because for a given value of $h \ll 1$ the $\mathcal{O}(h^2)$ error in the solution using the predictor-corrector method will be much smaller than the $\mathcal{O}(h)$ error of the Euler scheme. In numerical experiments we found that, when this new method was used with only one corrector step, the time step h could be increased by a factor of 100 compared with the forward Euler method without affecting the node positions over a given time period, enabling us to simulate over much longer time intervals at reasonable cost in terms of computer time.

Remark. The accuracy of f_ϵ can be increased by reducing the magnitude of ϵ . Using infinite precision arithmetic, it would be possible to compute f_ϵ to any required accuracy at no additional cost by using a sufficiently small ϵ . However, on a finite precision machine there will be a minimum value of ϵ below which rounding errors cause a *decrease* in accuracy. In our simulations a suitable value of ϵ was found empirically by simulating the deformation of a small test network using a fixed (small) value of h and fixed time interval. Starting from $\epsilon = h$ we repeatedly reduced the value of ϵ and observed the behaviour of the solution. At first the node co-ordinates appeared to be converging towards a fixed point as ϵ decreased, but once ϵ fell below a critical value the solution began to behave erratically, indicating that rounding errors were beginning to have an effect. ϵ was chosen to be a value well above this observed critical point, but much smaller than h^2 for any feasible choice of h .

4.3 Convergence analysis

The two numerical schemes outlined above can be thought of as special cases of a general scheme known as a Runge-Kutta method, whose definition can be found in [7] and is reproduced below.

Algorithm 4.1 *s-stage explicit Runge-Kutta method*

Let s be an integer and $a_{21}, a_{31}, a_{32}, \dots, a_{s1}, a_{s2}, \dots, a_{s,s-1}, b_1, \dots, b_s$ be real coefficients. Then the approximate solution to (3.1) defined by these coefficients is given by:

$$\begin{aligned} k_1 &= f(y_0) \\ k_2 &= f(y_0 + ha_{21}k_1) \\ k_3 &= f(y_0 + h(a_{31}k_1 + a_{32}k_2)) \\ &\dots \\ k_s &= f(y_0 + h(a_{s1}k_1 + \dots + a_{s,s-1}k_{s-1})) \\ y_1 &= y_0 + h(b_1k_1 + \dots + b_s k_s). \end{aligned}$$

The function $\Phi(y_0, h) = b_1k_1 + \dots + b_s k_s$ is known as the *increment function* of the method.

It is easy to verify that the Euler method is a one-stage explicit Runge-Kutta method with $\Phi(y_0, h) = f(y_0)$. Similarly, the above Predictor-corrector method, using r corrector steps, is an $(r + 1)$ -stage explicit Runge-Kutta method with $k_i = f(y_0 + \frac{h}{2}(k_1 + k_{i-1}))$ for $i = 2, \dots, r$, and $\Phi(y_0, h) = \frac{1}{2}(k_1 + k_r)$. In particular, when a single corrector step is used, we have

$$\begin{aligned} k_1 &= f(y_0) \\ k_2 &= f(y_0 + hk_1) \\ y_1 &= \frac{h}{2}(k_1 + k_2). \end{aligned}$$

Definition 4.1 The *local error* (i.e. the error in the computed solution over a single time-step) of the Runge-Kutta algorithm 4.1 is defined to be

$$|y(t+h) - y(t) - h\Phi(y(t), h)|. \quad (4.4)$$

The *global error* $E = y(t_N) - y_N$ in the final solution after N steps can be estimated using the following theorem (cf. theorem 3.6 in [7]):

Theorem 4.2 *Global error estimate*

Suppose that the local error of a numerical scheme satisfies, for given initial values,

$$|y(t+h) - y(t) - h\Phi(y(t), h)| \leq Ch^{p+1}, \quad (4.5)$$

and that, in a neighbourhood of the solution, the increment function Φ satisfies

$$|\Phi(z, h) - \Phi(y, h)| \leq \Lambda|z - y|. \quad (4.6)$$

Then the *global error* satisfies

$$|E| \leq h^p \frac{C}{\Lambda} \left(e^{\Lambda(t_N - t_0)} - 1 \right), \quad (4.7)$$

where $h = \max(h_i)$, h_i being the step length at the i -th step.

At this stage we need to make clearer those assumptions on the behaviour of the function f and the solution y that we will use in our subsequent analysis.

H1 f satisfies a Lipschitz condition $|f(y) - f(z)| \leq L|y - z|$ for all $y, z \in \mathbb{R}^n$ for some fixed $L > 0$,

H2 $y(t)$ is differentiable an appropriate number of times (to enable the analysis of the numerical scheme under consideration),

H3 $|f(y) - f_\epsilon(y)| = \epsilon_y < k_y \epsilon$ for all $y \in \mathbb{R}^n$ for some $k_y < \kappa < \infty$.

H1 and H2 are reasonable in the case of the grain boundary diffusion problem described in §2 since they correspond to the usual modelling assumptions that the grains move smoothly without cracks forming between them. (We shall see, in §7, that in fact additional constraints on the grain movements are necessary in order to *guarantee* that H1 is satisfied.) H3 is true provided that an appropriate method is used to choose a node position, given the equations of the adjacent grain boundaries. For example, as we remarked earlier, if the centroid (rather than the centre of the incircle) of the triangle of grain boundary intersections is used to define the new node position then H3 is no longer true because when two of the three grain boundaries are nearly parallel a small change in the grain boundary positions may result in a very large change in the position of the centroid so that the limit $\lim_{\delta t \rightarrow 0^+} \frac{\hat{y}(t+\delta t) - y(t)}{\delta t}$ does not exist.

We are now equipped to establish the order of convergence of our two numerical schemes, which are approximate Runge-Kutta methods in which we replace the increment function $\Phi(y_0, h)$ by an approximation $\Phi_\epsilon(y_0, h)$.

Lemma 4.3 *When $f(y)$ is approximated by $f_\epsilon(y)$ in algorithm 4.1, where $f(y)$ satisfies H1 and $f_\epsilon(y)$ satisfies H3, the new increment function $\Phi_\epsilon(y_0, h)$ satisfies $\Phi_\epsilon(y_0, h) = \Phi(y_0, h) + \mathcal{O}(\epsilon)$.*

proof Let $k_{i\epsilon}$ denote the perturbed value of k_i when f is replaced by f_ϵ .

$$\begin{aligned} k_{1\epsilon} &= f_\epsilon(y_0) \\ &= f(y_0) + \mathcal{O}(\epsilon) \quad (H3) \\ &= k_1 + \mathcal{O}(\epsilon); \end{aligned}$$

$$\begin{aligned}
k_{2\epsilon} &= f_\epsilon(y_0 + ha_{21}k_{1\epsilon}) \\
&= f(y_0 + ha_{21}k_{1\epsilon}) + \mathcal{O}(\epsilon) \quad (H3) \\
&= f(y_0 + ha_{21}(k_1 + \mathcal{O}(\epsilon))) + \mathcal{O}(\epsilon) \quad (\text{from above}) \\
&= f(y_0 + ha_{21}k_1) + \mathcal{O}(\epsilon) \quad (H1) \\
&= k_2 + \mathcal{O}(\epsilon);
\end{aligned}$$

...

$$\begin{aligned}
k_{s\epsilon} &= f_\epsilon(y_0 + h(a_{s1}k_{1\epsilon} + \dots + a_{s,s-1}k_{s-1\epsilon})) \\
&= f(y_0 + h(a_{s1}k_{1\epsilon} + \dots + a_{s,s-1}k_{s-1\epsilon})) + \mathcal{O}(\epsilon) \\
&= k_s + \mathcal{O}(\epsilon).
\end{aligned}$$

Hence $\Phi_\epsilon(y_0, h) = b_1k_{1\epsilon} + \dots + b_s k_{s\epsilon} = b_1(k_1 + \mathcal{O}(\epsilon)) + \dots + b_s(k_s + \mathcal{O}(\epsilon)) = \Phi(y_0, h) + \mathcal{O}(\epsilon)$. \square

Theorem 4.4 *When $f(y)$ is approximated by $f_\epsilon(y) = f(y) + \mathcal{O}(\epsilon)$ in algorithm 4.1 then, assuming that hypotheses H1-H3 are satisfied, the global error in the solution to (3.1) is $\mathcal{O}(h^p)$ whenever the local error of the exact method is $\mathcal{O}(h^{p+1})$, provided that $\epsilon = \mathcal{O}(h^p)$.*

proof From lemma 4.3, the local error is given by

$$|y(t+h) - y(t) - h\Phi_\epsilon(y(t), h)| = |y(t+h) - y(t) - h\Phi(y(t), h)| + \mathcal{O}(h\epsilon) = \mathcal{O}(h^{p+1}). \quad (4.8)$$

Hence, using theorem 4.2, the global error is $\mathcal{O}(h^p)$. \square

It is well-known (and straightforward to prove) that the Euler method has $\mathcal{O}(h^2)$ local error provided that the solution $y(t)$ is twice differentiable, so we can deduce that our approximate Euler method will have a global error of order h , provided that $\epsilon = \mathcal{O}(h)$. Similarly, the predictor-corrector method has $\mathcal{O}(h^3)$ local error provided that $y(t)$ is three times differentiable. Hence our approximate version will have a global error of order h^2 , provided that $\epsilon = \mathcal{O}(h^2)$. Our numerical experiments (see §5) confirm the orders of convergence for these two methods when used to simulate deformation of a small network of grain boundaries.

5 Demonstration of efficiency improvements

We have implemented our simulation methods in a program written in C together with a suite of Matlab m-files which can be used to produce graphical displays of the grain boundary network as it deforms over time. We have used it to simulate deformation of small polycrystals under a variety of different boundary conditions. As an initial test, we simulated a network of regular hexagonal grains and found that, when a uniform uniaxial stress is applied, the initial normal stress distribution

given by our program agrees with the analytic results of Spingarn & Nix [15]. We discuss further the behaviour of this network in section 7.

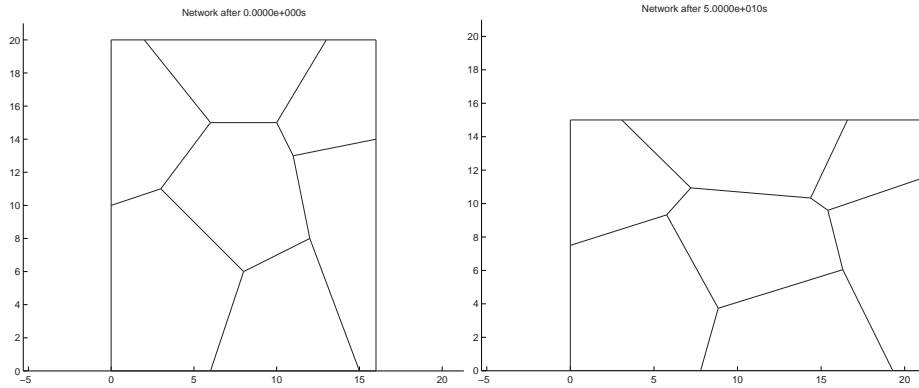


Figure 5.1: Deformation of a small polycrystal with irregular grain structure under compressive stress with velocity boundary conditions.

To investigate the order of convergence of the two numerical methods, we used the small grain boundary network shown in figure 5.1. When compressive stress is applied in the vertical direction and the sides of the polycrystal are constrained to move without tilting of the boundary grains, the structure deforms as shown, with the grains shortening in the direction of the applied stress as material moves from grain boundaries under compressive stress to those under (relative) tensile stress. This grain network has 16 nodes, so that the network at any given time t can be defined by the 32 node co-ordinates, which we denote by the vector $z(t)$. Let $z_h(t)$ be the value of $z(t)$ obtained when the problem is solved numerically using a (fixed) time step h .

We used each of our two methods to simulate deformation over a fixed time interval $T = 5 \times 10^{10}$ s using different values of h . Table 5.1 shows that, as the number of time steps is increased from 5 to 500 000 (so that h is reduced in magnitude from 10^{10} s to 10^5 s) $z_h(T)$ tends to a limit with convergence of order h in the Euler case and order h^2 using the predictor-corrector method with a fixed ϵ . Notice that, if the predictor-corrector method is used with $\epsilon = h$, convergence is reduced to $\mathcal{O}(h)$ because the local error is now $\mathcal{O}(\epsilon h) = \mathcal{O}(h^2)$. Using the predictor-corrector method with fixed small ϵ the results for $h = 10^5, 10^6$ and 10^7 are identical, to machine precision, so there is no advantage to be gained by using a value of h smaller than 10^7 . In all our experiments a *single* corrector step was used.

Table 5.2 illustrates how use of the predictor-corrector method, with suitably chosen fixed ϵ , enables us to simulate deformation accurately using many fewer time-steps than using the Euler method. Different numbers N of time-steps were used to compute the solution at $T = 5 \times 10^{10}$ s and these solutions were compared with the “exact” solution $\bar{z}(T)$ obtained using the predictor-corrector method with

Table 5.1: Comparison of convergence of Euler and predictor-corrector methods. For values of h from 1 to 10000, the value of $|z_{10h}(T) - z_h(T)|/|z_1(T)|$ is shown for the Euler and predictor-corrector methods using both fixed ϵ and $\epsilon = h$.

h	Euler, $\epsilon = h$	Euler, $\epsilon = 10^3$	P-C, $\epsilon = h$	P-C, $\epsilon = 10^3$
10^5	1.4912×10^{-6}	1.2807×10^{-6}	8.7053×10^{-7}	0
10^6	1.4992×10^{-5}	1.2737×10^{-5}	8.7045×10^{-6}	0
10^7	1.4972×10^{-4}	1.2735×10^{-4}	8.6892×10^{-5}	9.1601×10^{-8}
10^8	1.4781×10^{-3}	1.2636×10^{-3}	8.6029×10^{-4}	7.7337×10^{-6}
10^9	1.3326×10^{-2}	1.1751×10^{-2}	7.7078×10^{-3}	7.2644×10^{-4}

Table 5.2: Comparison of efficiency of Euler and predictor-corrector methods. Using different numbers, N , of time steps, $|z_h(T) - \bar{z}(T)|/|\bar{z}(T)|$ is shown for the Euler and predictor-corrector methods using both fixed ϵ and $\epsilon = h$.

N	Euler, $\epsilon = h$	Euler, $\epsilon = 10^3$	P-C, $\epsilon = h$	P-C, $\epsilon = 10^3$
500 000	1.6746×10^{-7}	1.4407×10^{-7}	8.7911×10^{-8}	0
50 000	1.6565×10^{-6}	1.4217×10^{-6}	9.5553×10^{-7}	0
5000	1.6648×10^{-5}	1.4158×10^{-5}	9.6596×10^{-6}	0
500	1.6637×10^{-4}	1.4150×10^{-4}	9.6552×10^{-5}	9.3981×10^{-8}
50	1.6445×10^{-3}	1.4051×10^{-3}	9.5683×10^{-4}	7.8159×10^{-6}
5	1.4967×10^{-2}	1.3156×10^{-2}	8.6628×10^{-3}	7.3425×10^{-4}

the smallest value of h (i.e. $h = 10^5$ s). The accuracies of the two Euler methods and the predictor-corrector method with $\epsilon = h$ are very similar, but using the predictor-corrector method with fixed $\epsilon < h^2$ allows the number of time steps to be reduced by a factor of 100 or more without affecting the accuracy of the solution. The cost of each predictor-corrector step, using one corrector iteration, is a little over twice that of each Euler step (since it is necessary to compute two Euler steps and then to apply the trapezoidal correction). This means that using the predictor-corrector method reduces the amount of processor time required to obtain a given accuracy by a factor of about 50 (more if higher accuracy is required). For example, to obtain the same accuracy as that given by the predictor corrector method using 50 steps requires 10 000 steps of the Euler method and the corresponding solution times on a Pentium 3 PC were 0.66s and 70.91s respectively, making the predictor-corrector method the more cost-effective by a factor of over 100 times.

6 Further Examples

We now present some other examples to illustrate how the program can be used to predict the behaviour of granular materials under stress.

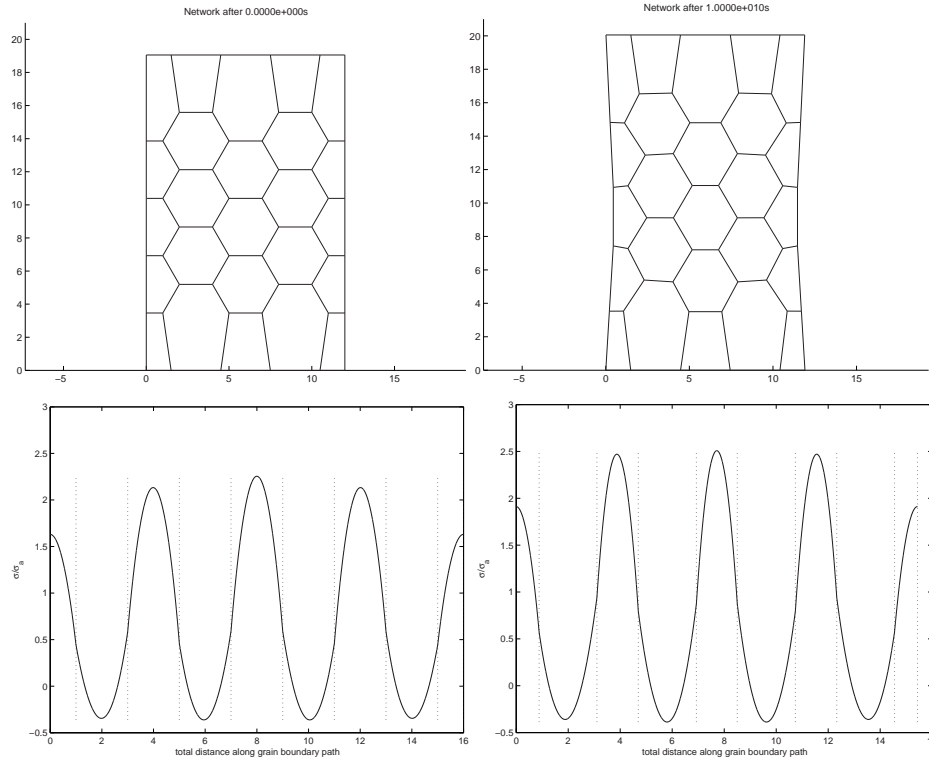


Figure 6.1: Deformation of a polycrystal under tensile stress and corresponding stress distributions.

Figure 6.1 shows the deformation of a polycrystal consisting mainly of regular hexagonal grains when a tensile force is exerted in the y direction. The total force is kept constant over time and the top and bottom surfaces of the polycrystal are forced to remain horizontal. Notice that necking occurs, although *no* horizontal force is applied to the top and bottom grains. This suggests that necking is not caused solely by frictional forces preventing horizontal movement of grains at the top and bottom boundaries, as has sometimes been suggested. The graphs show the normal stress distribution, as a fraction of the average applied stress, in the undeformed and deformed polycrystal along a path through the grain boundary network from left to right. The stress is considerably higher in the deformed polycrystal because the chosen path is near the centre of the polycrystal where maximum necking has occurred.

Figure 6.2 shows an irregular grain structure with a large grain near the centre. When a tensile force is applied so that the top of the polycrystal is constrained to move upwards at constant speed the large grain inhibits necking, which now occurs above and below the centre.

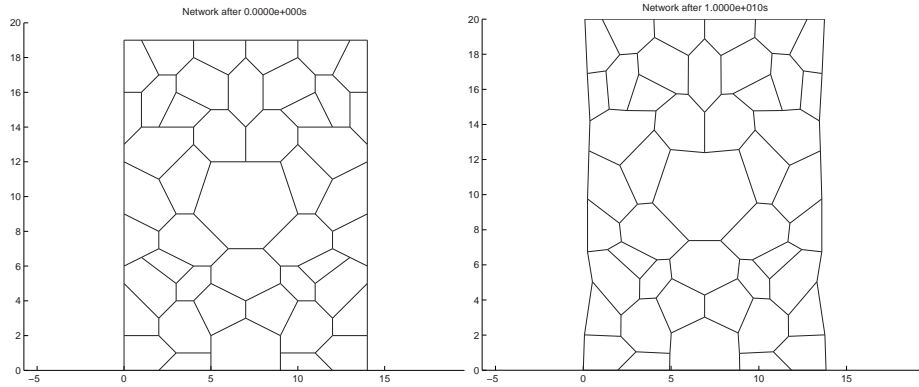


Figure 6.2: Deformation of a polycrystal with irregular grain structure under tensile stress.

7 Stability of the model

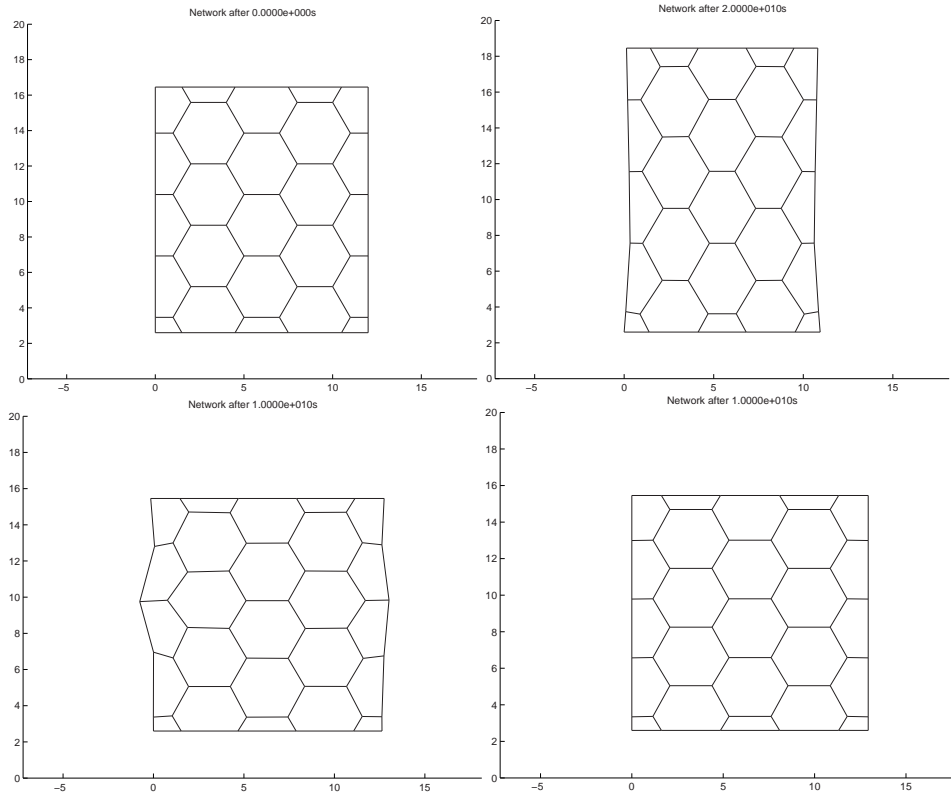


Figure 7.1: Network of regular hexagonal grains: undeformed (top left), deformed by tensile stress (top right), by compressive stress (bottom left) and by compressive stress with sides constrained .

Figure 7.1 shows a network made up of regular hexagonal grains. We used this network to test our simulation algorithm because

- the initial stress distribution can be computed analytically and compared with the numerical result, and
- the symmetry of the structure allows us to predict, to a certain extent, how the deformation will proceed.

In particular, it is clear that if a symmetric external stress distribution is applied the deformation must progress symmetrically. We found that this symmetry is accurately reflected in the simulated behaviour when *either* a tensile force was applied to the top and bottom of the structure *or* a compressive force was applied to the top and bottom with the side grains constrained to move with the same x -velocity and without tilting. But when a compressive force was applied with the sides unconstrained the structure rapidly became non-symmetric and started to buckle. At the onset of buckling the force required to maintain a constant downward velocity of the top grains decreased rapidly.

These results are precisely what one might expect to happen when a column of deformable material is subjected to the forces described. In a practical experiment we would not expect an unconstrained structure to deform symmetrically under a compressive force because small imperfections in the material or the method of applying the force would cause a small degree of asymmetry to develop, which we would expect to become greater over time. By contrast, if a slightly asymmetric column of material is subjected to a symmetric tensile force we would expect the material to deform in such a way as to become more symmetric. When velocity boundary conditions are applied at the sides, it is not possible for the overall shape to become asymmetric, so we do not expect a small change in the initial conditions to cause a large change in the final outcome.

These considerations strongly suggest that problems in which the side grains are unconstrained may have an inherent instability. By this we mean that a small change in the initial conditions can result in a large change in the behaviour over time. Solution of such a problem by a numerical method is difficult and may in some cases be impossible. (See [12, §1.7].)

The instability is illustrated in figure 7.2, which shows two very similar grain boundary networks. The right-hand network differs from the left-hand one only in the angle of one of the grain boundaries. In the left-hand network the two oblique boundaries form a single straight line, while in the other network they are at slightly different angles. If a compressive force is applied in the y direction to the left-hand network the two upper grains will slide downward, relative to the lower grain, with increasing speed without any dissolution/deposition being necessary. The lack of any shear forces along the grain boundaries means that it is not possible for the grains to be in mechanical equilibrium. The right-hand network, on the other hand, cannot deform by sliding unless some deposition/dissolution occurs along the grain boundaries. This is true *however small* the angle between the two oblique grain boundaries may be, provided only that it is non-zero. As this angle decreases, the

stress distributions along the grain boundaries will vary smoothly until the point at which the grain boundaries become parallel, when they will become indeterminate. If the sides of the network are constrained, by imposing velocity boundary conditions, this discontinuity does not occur; indeed no deformation is possible for either network since the single grain across the bottom of the network cannot change its width.

More generally, when the grain velocities are constrained on all exterior boundaries, grain boundary sliding can only occur in combination with diffusional transport of matter which fills voids created by the grain movement. Small changes in the network configuration will result in small changes in the stress distribution and hence in the dissolution rate along the boundaries, which will cause only small differences in the grain movements. If the grains are *not* so constrained, a small change in the initial conditions may enable one or more grains to slide in a way that is not controlled by the dissolution rates along the grain boundaries, creating a large difference in the subsequent motion of the grains. This violates hypothesis H1 of §4(4.3). Hence, we expect our simulation to give an accurate prediction of behaviour over time in the case of constrained networks, but cannot be so confident in the unconstrained case.

However, the aim of our simulation is to model the behaviour of materials under stress, rather than to solve (3.1). The (unintentional) errors introduced by our numerical scheme have produced results that reflect reality in ways that the “true” solution of (3.1) would not. This suggests that one way of making our *model* more realistic might be the introduction of a stochastic element, which would take into account slight random differences within the grain structure.

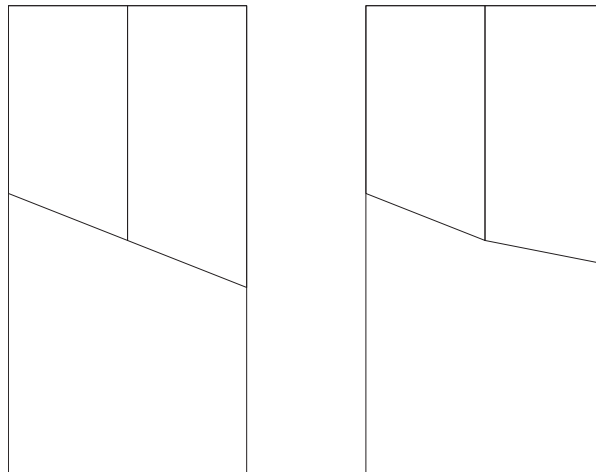


Figure 7.2: Two networks which may deform very differently

8 Conclusion and future work

We have developed numerical methods for simulating grain boundary diffusion creep that allow deformation to be modelled over relatively long time periods. We have shown that these methods will give reliable results under certain hypotheses on the nature of the node movements. This leads us to conclude that, for networks in which the grains are constrained, to prevent sliding except when accommodated by diffusional transport of matter, our methods can be expected to provide an accurate prediction of behaviour over time. For unconstrained networks, the predictions should be seen as qualitative rather than quantitative because the behaviour of the network is then inherently unstable. Some experimental evidence suggests that, for vertical compression of networks *without* velocity boundary conditions on the side grains, the predictor-corrector method may approach a “solution” with order h (rather than order h^2) convergence as $h \rightarrow 0$, but further work is needed to test this hypothesis and to establish a theoretical basis for it.

Simulation of this kind has applications in the fields of materials science (where it can provide information about the effect of grain microstructures on the strength of materials under stress) and in earth sciences (where grain boundary diffusion is an important mechanism in both rock deformation and metamorphism). We have implemented these methods in a computer program which can be used to investigate the effect of irregularities in grain shapes and sizes on the strength of single-phase materials. We have also used it to model certain multi-phase materials and have confirmed some predictions about the behaviour of multi-phase rocks undergoing diffusion creep made by Wheeler [16]. The simulation has also highlighted some new and surprising phenomena regarding solution creep in materials where phases have shared components, which we are currently studying in more detail.

The two numerical methods that we have implemented can be viewed as approximate Runge-Kutta methods of first and second order respectively. It would, in principle, be straightforward to implement higher order explicit Runge-Kutta methods, but care would be needed since the use of an order h^p method requires $y_\epsilon(t)$ to be computed to order h^p accuracy. This means that, as h decreases in magnitude, ϵ must decrease at a faster rate so that the critical value below which rounding errors become significant will be reached for a larger value of h .

When the simulation is run over long time periods, deformation can result in nodes crossing grain boundaries, which is clearly a physical impossibility. The detection of such occurrences and determination of appropriate action to be taken are issues that we plan to address in the near future. The possibility of introducing a stochastic element into the model to take into account small random effects is being considered.

References

- [1] B. Andrásfai. *Introductory Graph Theory*. Adam Hilger, Bristol, 1977.
- [2] B. Burton. *Diffusional creep of polycrystalline materials*. Trans. Tech. Pub., Aedermansdorff, 1977.
- [3] R. L. Coble. A model for boundary diffusion controlled creep in polycrystalline materials. *Journal of Applied Physics*, 34:1679–1682, 1963.
- [4] T. Dewers & P. Ortoleva. A coupled reaction/transport/mechanical model for intergranular pressure solution stylolites, and differential compaction and cementation in clean sandstones. *Geochim. Cosmochim. Acta*, 54:1609–1625, 1990.
- [5] D. Elliott. Diffusion flow laws in metamorphic rocks. *Geol. Soc. Am. Bull.*, 84:2645–2664, 1973.
- [6] J. M. Ford, J. Wheeler & A. B. Movchan. Computer simulation of grain boundary creep. *Acta Mater.*, 50(15):3941–3955, 2002.
- [7] E. Hairer, S. Nørsett & G. Wanner. *Solving Ordinary Differential Equations I* 2nd edn. Springer-Verlag, Berlin, 1993.
- [8] E. Hairer & G. Wanner. *Solving Ordinary Differential Equations II*. Springer-Verlag, Berlin, 1991.
- [9] P. M. Hazzledine & J. H. Schneibel. Theory of Coble creep for irregular grain structures. *Acta Metallurgica Materialia*, 41(4):1253–1262, 1993.
- [10] B.-N. Kim & K. Hiraga. Simulation of diffusional creep accompanied by grain growth in two-dimensional polycrystalline solids. *Acta Materialia*, 48:4151–4159, 2000.
- [11] J.-P. Poirier. *Creep of Crystals*. CUP, Cambridge, 1985.
- [12] A. Ralston & P. Rabinowitz. *A First Course in Numerical Analysis*. McGraw-Hill, London, 1978.
- [13] E. H. Rutter. The kinetics of rock deformation by pressure solution. *Trans. R. Soc. Lond.*, A283:203–219, 1976.
- [14] P. G. Shewmon. *Transformations in Metals*. McGraw-Hill, New York, 1969.
- [15] J. R. Spingarn & W. D. Nix. Diffusional creep and diffusionally accommodated grain rearrangement. *Acta Metallurgica*, 26:1389–1398, 1978.
- [16] J. Wheeler. Importance of pressure solution and Coble creep in the deformation of polymineralic rocks. *Journal of Geophysical Research*, 97(B4):4579–4586, 1992.