

## SHORT GUIDE TO THE USE OF THE FINITE DIFFERENCE METHOD FOR NUMERICAL CALCULATIONS WITH THE DIFFUSION EQUATION

The first step is to discretize the space and time coordinates into finite but small increments to make a grid. This is shown in **Figure A1**. We have a step size  $\Delta x$  on the space coordinate, and  $\Delta t$  on the time coordinate. In the simplest implementations these are taken to be constants, but are not required to be so. These define the smallest variations that can be “recognized” by the computer at that particular point of space and time. For example, if we let  $\Delta x = 5 \mu\text{m}$  and  $\Delta t = 3600 \text{ s}$ , then distances smaller than  $5 \mu\text{m}$  and time scales smaller than  $3600 \text{ s}$  cannot be resolved by the numerical calculation. Any numerical calculation is a trade off between speed (larger the individual steps, fewer of these are required to get to any particular value) and accuracy (smaller the step, closer it is to being truly infinitesimal, as required by rigorous mathematics). We are concerned with describing concentrations as a function of space and time—the variable  $C(x,t)$  used in the text. In the numerical implementation, we can associate each value of concentration with two indices written as subscripts—one for space ( $i$ ) and the other for time ( $j$ ), e.g.,  $C_{i,j}$ . If we write, for example,  $C_{2,2}$  we are referring to the concentration at grid point number 2 on the spatial scale at time step number 2. For our chosen value of  $\Delta x = 5 \mu\text{m}$  and  $\Delta t = 3600 \text{ s}$ ,  $C_{2,2}$  is the concentration at a distance of  $10 \mu\text{m}$  from our arbitrarily chosen zero position after 2 hours (Fig. A1).

The next step is to convert the differential expressions into discrete forms using the Taylor Series approximation (see Crank 1975, for a detailed exposition geared to the solution of diffusion problems). There are several levels of approximation that may be used (first order, second order etc.) and different options about the choice of neighborhood in which to carry out the Taylor expansion (forward difference, backward difference, central difference etc.). The simplest of these, in a forward difference scheme, can be written as

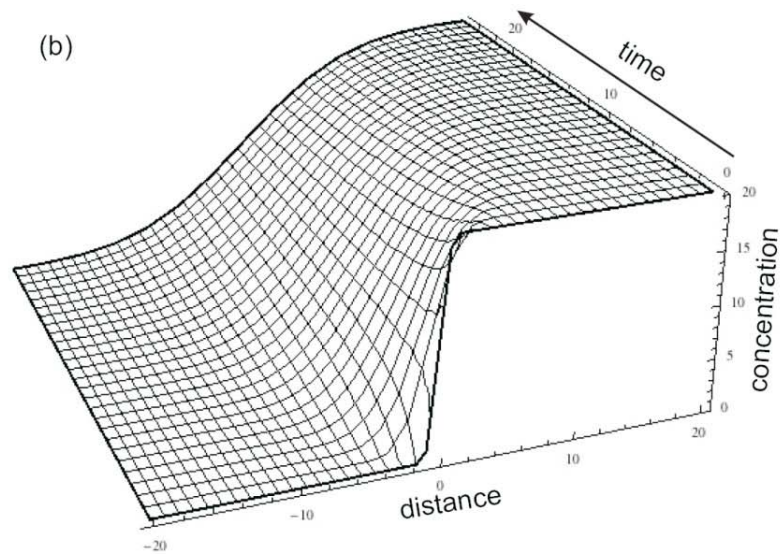
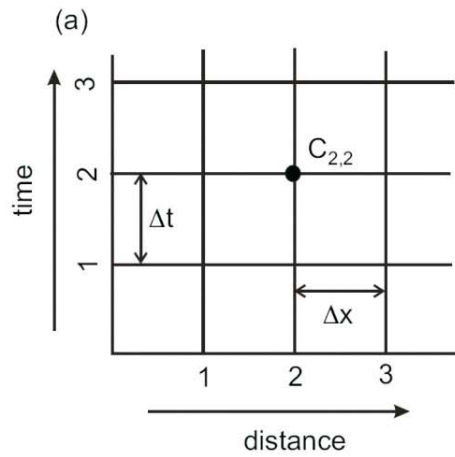
$$\frac{\partial C}{\partial x} = \frac{C_{i+1,j} - C_{i,j}}{\Delta x}$$

The most important point to note here is that this expression converts a differential expression into a simple algebraic expression that can be manipulated by straightforward rules of addition and multiplication. Higher derivatives may be written by repeating the same approach. For example,

$$\frac{\partial^2 C}{\partial x^2} = \frac{\partial}{\partial x} \left( \frac{\partial C}{\partial x} \right) = \frac{1}{\Delta x} \left( \left( \frac{C_{i+1,j} - C_{i,j}}{\Delta x} \right) - \left( \frac{C_{i,j} - C_{i-1,j}}{\Delta x} \right) \right) = \frac{C_{i+1,j} - 2C_{i,j} + C_{i-1,j}}{(\Delta x)^2}$$

Similarly, for the time derivative we have

$$\frac{\partial C}{\partial t} = \frac{C_{i,j+1} - C_{i,j}}{\Delta t}$$



**Figure A1.** Visualization of the discretization of time and space used in finite difference algorithms. (a) There is a value of concentration associated with each point in space and time. For example,  $C_{2,2}$  stands for the value of concentration at a distance of  $2 \cdot \Delta x$  at the point of time  $2 \cdot \Delta t$ . (b) Three-dimensional visualization of how the concentration changes due to diffusion using the discretization of space and time. See also text for more explanation.

If we put it all together and move terms around using algebra, the continuity Equation (11a) in the text translates to

$$C_{i,j+1} = C_{i,j} + D\Delta t \left( \frac{C_{i+1,j} - 2C_{i,j} + C_{i-1,j}}{\Delta x^2} \right) \quad (\text{A1})$$

for calculating the concentration at a grid point  $i$  at the next time step,  $j+1$ . This formula can be iterated over all grid points, shifting the index  $i$  by 1 each time, to obtain a complete concentration profile at the time step  $j+1$ . The entire procedure may be then repeated to calculate the concentration profile at the next time step,  $j+2$ , using the values determined at step  $j+1$  as the “old” values. If this is done until the index  $j$  reaches the value 2 (with the time step,  $\Delta t = 3600$  s, as above), we would have calculated a concentration profile that would be measured after 2 hours of diffusion. This numerical method is called the explicit finite difference method.

These simple formulae and iterations can be implemented on a standard spreadsheet program such as Excel®. On the MSA WWW site (<http://www.minsocam.org/MSA/RIM>), we provide a spreadsheet (olivine\_diffu\_excel.xls) and a Mathematica® notebook (olivine\_diffu\_mathema.nb) with an example for Fe-Mg diffusion in olivine, but it is an instructive exercise to consider the various constitutive equations described in the text, derive the corresponding continuity equations, and then digitize them to the discrete numerical form as above. For example, if the diffusion coefficient depends on concentration we need to solve another form of the diffusion equation (Eqn. 11b), which in finite difference form is:

$$C_{i,j+1} = C_{i,j} + \Delta t \left( \frac{D_{i+1,j} - D_{i,j}}{\Delta x} \right) \left( \frac{C_{i+1,j} - C_{i,j}}{\Delta x} \right) + D_{i,j} \Delta t \left( \frac{C_{i+1,j} - 2C_{i,j} + C_{i-1,j}}{\Delta x^2} \right) \quad (\text{A2})$$

For actual numerical implementation it is necessary to pay attention to a few other details. For any explicit numerical scheme to be stable, it is necessary that the so-called Courant condition be fulfilled. For Equation (A1),  $D$ ,  $\Delta x$  and  $\Delta t$  can be combined together into one parameter,  $r = D\Delta t / (\Delta x)^2$ . The Courant condition is that  $r < 0.5$  (e.g., see Crank 1975). If the scheme has been implemented on a spreadsheet, it is easy to see what happens when  $r$  exceeds a value of about 0.5. If  $D$ ,  $\Delta x$  and  $\Delta t$  are not constant during the course of a calculation (e.g., Eqn. A2 where  $D$  change with position or concentration, or for a non-isothermal case, where  $D$  changes with temperature, and hence, time), then it is necessary to ensure that the Courant condition is fulfilled at each grid point. With these equations and the boundary and initial conditions for the case at hand we can solve the diffusion equation and fit compositional profiles. For example, for the case of no flux at the boundary, we write  $C_{1,j(\text{all})} = C_{2,j(\text{all})}$ , and likewise  $C_{n,j(\text{all})} = C_{n-1,j(\text{all})}$  (where  $n$  is the total number of space grid points and  $j(\text{all})$  means for all times; this is the condition used in Fig. 6a). For the case that there is exchange, but a constant composition at the boundary ( $C_b$ ) we have  $C_{1,j(\text{all})} = C_{n,j(\text{all})} = C_b$  (this is the condition used in Fig. 6b, with  $C_b = 40$ ).

Inspection of the finite difference formula provides several insights into the advantages and limitations of the numerical method. Some of these are:

- (1) If an explicit method is to be used, we cannot choose  $D$ ,  $\Delta x$  and  $\Delta t$  arbitrarily. As  $D$  and  $\Delta x$  are usually specified by the problem at hand, the choice of  $\Delta t$  is limited by the Courant condition. If it turns out that  $\Delta t$  needs to be 3600 s or less, for example, it means that to calculate a concentration profile after 100 years of diffusion one would need to carry out the steps noted above 876000 times at each spatial grid

point. For calculations to millions of years (e.g., cooling of a pluton), the difficulty escalates. This is why it is necessary to use a programming tool or language to carry out realistic calculations and “iteration by hand” on a spreadsheet can become a little too cumbersome. Packages such as Mathematica®, Mathcad®, Maple®, or Matlab® allow such implementation of automated iteration without full-fledged programming using a high level language (e.g., C, Fortran). The latter are in general faster.

- (2) Inspection of the formula above shows why it is necessary to have the two boundary conditions and one initial condition. To calculate the concentration at a grid point  $i$ , it is necessary to know the concentrations at grid points  $i-1$  and  $i+1$ . These formulae cannot be used to calculate concentrations at the first (no  $i-1$  grid point available) and the last (no  $i+1$  grid point available) positions. The two boundary conditions provide the additional information to calculate concentrations at these points. To obtain the concentrations at time step  $j+1$ , we need information about concentrations at time step  $j$ . But these formulae cannot be used for the first time step and therefore we need the initial conditions to start the iterations. It is an instructive exercise to try to implement various forms of boundary conditions (concentration held constant, no flux etc.) in the discrete form.
- (3) As these calculations proceed by explicitly determining the concentrations at each position at each time step, it is easy to add variations in the model. For example, to start off a calculation we need to define the concentrations at each grid point (the initial condition). Because each concentration is specified individually, it is irrelevant whether the concentration profile is homogeneous, has a regular geometric form, or is of an arbitrary shape. Similarly, if we wish to change the boundary conditions subject to which diffusion occurs after, say, the 20<sup>th</sup> time step, we can directly do so. This is where the flexibility of the numerical method noted in the text comes from. This is also how we can take outputs from thermodynamic programs such as MELTS and track diffusion along complex  $P$ - $T$ - $f_{\text{O}_2}$ - $f_{\text{H}_2\text{O}}$  paths, with boundary conditions and diffusion coefficients that vary as the coexisting phase assemblage and equilibrium conditions change. More about discretization and numerical solution can be found in for example in Press et al. (2007).