Numerical Modelling of Chemical Diffusion in Petrology and Geochemistry

Evangelos Moulas

Metamorphic Processes Research Group

Institute of Geosciences & Mainz Institute of Multiscale Modeling (M^3ODEL), JGU Mainz

(evmoulas@uni-mainz.de)

doi: 10.5281/zenodo.10047582

version 1.2 – October 2023

Preface on version 1.0

These notes will be published online (at https://zenodo.org/) at the end of the workshop. Apart from the lecture material, I have provided code examples that the interested readers can copy and use for their own research. These notes do not cover the diffusion theory extensively, and the interested reader should refer to available textbooks (e.g. Balluffi et al., 2005, Zhang, 2008, see the reference list at the end). The codes and the rest of the material are provided with no warranty. They are mostly done for educational purposes and assume no programming experience on behalf of the user. Therefore, to more experienced programmers, these codes may look redundant.

Since this is the first version of these notes it is possible that these notes contain errors. Instead of waiting until I complete the 'perfect' notes I decided to proceed with the material that I already have. Any constructive feedback and comments are welcome and should be addressed to evmoulas@uni-mainz.de. I will try to update the notes every year depending on the requests that I get from the workshop participants or other interested readers. In case you find these notes useful, please cite them appropriately (see rules at https://zenodo.org/).

Writing these notes and the organization of the workshop would not have been possible without the support from the German Mineralogical Society (DMG) and the Mainz Institute of Multiscale Modeling (M^{3}ODEL). The institute of Geosciences in Mainz is acknowledged for hosting the workshop and Claudia Scheer is thanked for providing valuable assistance. I would also like to thank Lucie Tajčmanová, Roman Botcharnikov, Sumit Chakraborty and Boris Kaus for encouraging me to organize this workshop. Finally, I would like to thank Simon Boisserée and Annalena Stroh for going through this material and for providing feedback in advance.

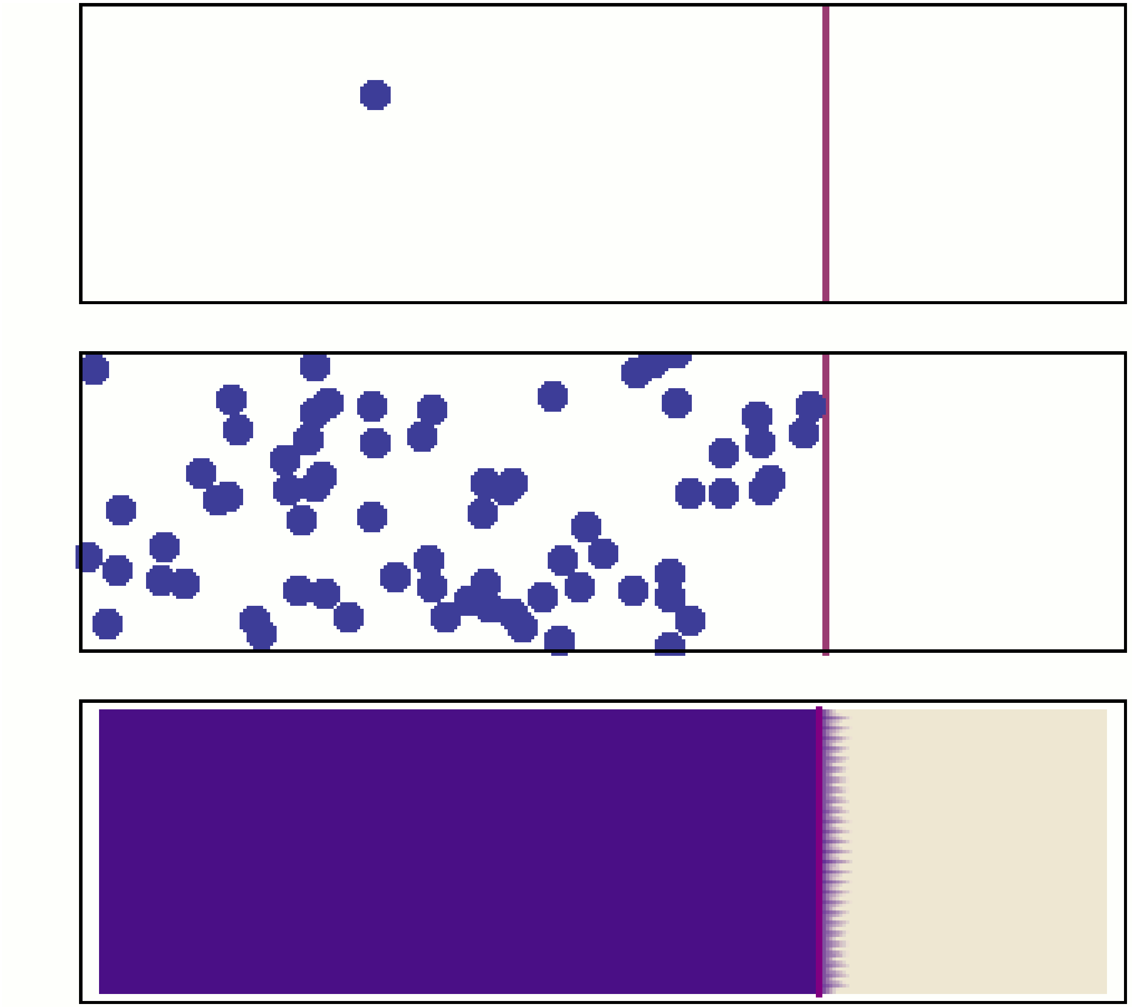
Evangelos Moulas
Mainz, July/2023

Workshop Contents

- Introduction to Diffusion
- Simple analytical solutions
- Introduction to Finite Differences
- Geospeedometry and time transformation techniques
- Introduction to the Finite Element Method
- Programming specific examples using the Finite Element Method
- Multicomponent Diffusion (optional)
- Nonlinear, concentration-dependent diffusion (optional)

Introduction to the Diffusion Equations

Introduction to Diffusion Equations



Diffusion processes are everywhere. From the diffusion of heat, diffusion of matter and diffusion of fluid pressure, diffusion is a fundamental process that we have to understand in order to model systems that evolve in time and in space. In geosciences in particular, many fundamental problems are related to diffusion processes and can be solved using techniques that were initially developed for different fields.

Image from Wikipedia

Introduction to Diffusion Equations

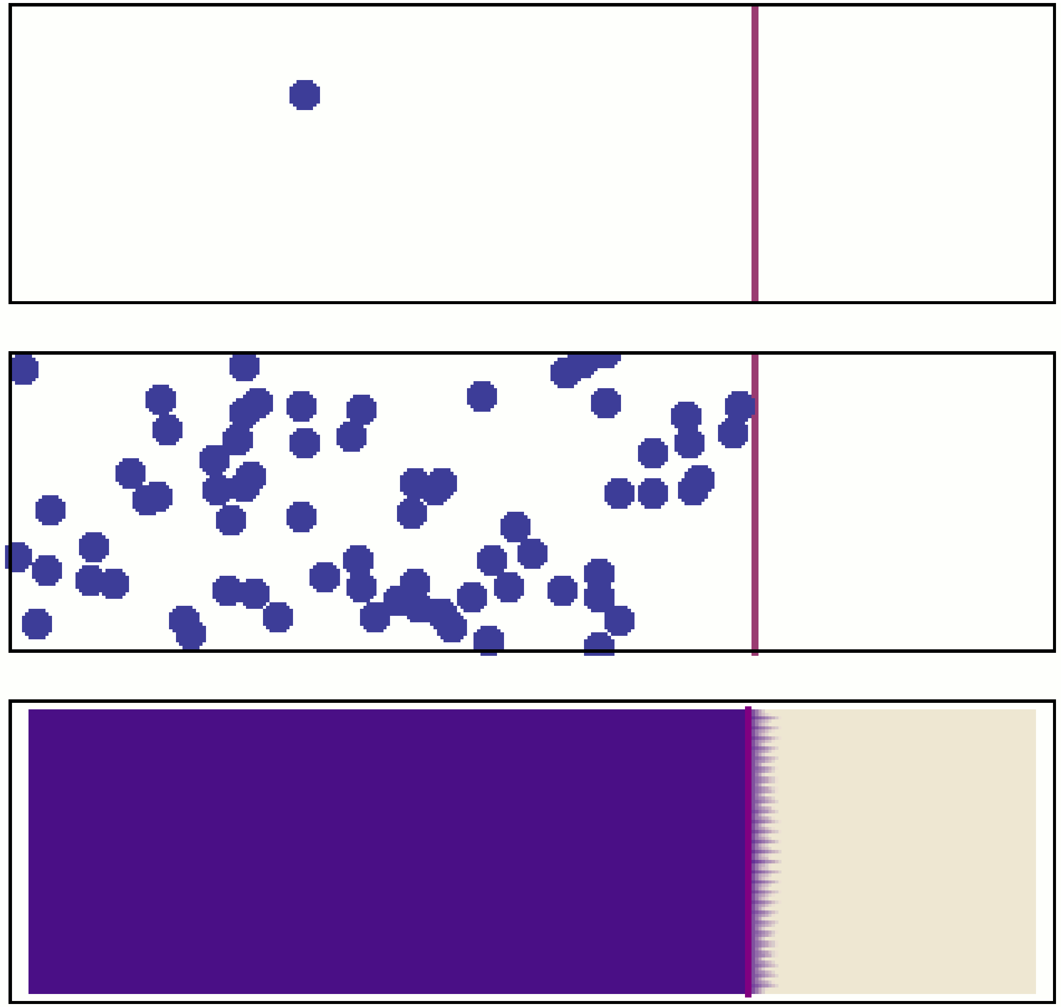
"If, in some cataclysm, all of scientific knowledge were to be destroyed, and only one sentence passed on to the next generation of creatures, what statement would contain the most information in the fewest words? I believe it is the atomic hypothesis that all things are made of atoms — little particles that move around **in perpetual motion**, attracting each other when they are a little distance apart, but repelling upon being squeezed into one another. In that one sentence, you will see, there is an enormous amount of information about the world, if just a little imagination and thinking are applied."

(from the Feynman lectures of Physics, vol.1 – online available at: https://www.feynmanlectures.caltech.edu/)

Richard P. Feynman
(Nobel Prize in Physics, 1965)
source: Wikipedia

I have highlighted "**perpetual motion**" since it is the main reason for diffusion. In other words, diffusion would not have been possible without the random motions of atoms.

Introduction to Diffusion Equations



Within any given fluid, the random movements of atoms, particles etc, can result in situations were mixing is enhanced.

The probability of the particles to mix is much larger than the probability where all the particles are “ordered” in one side of the crystal.

Image from Wikipedia

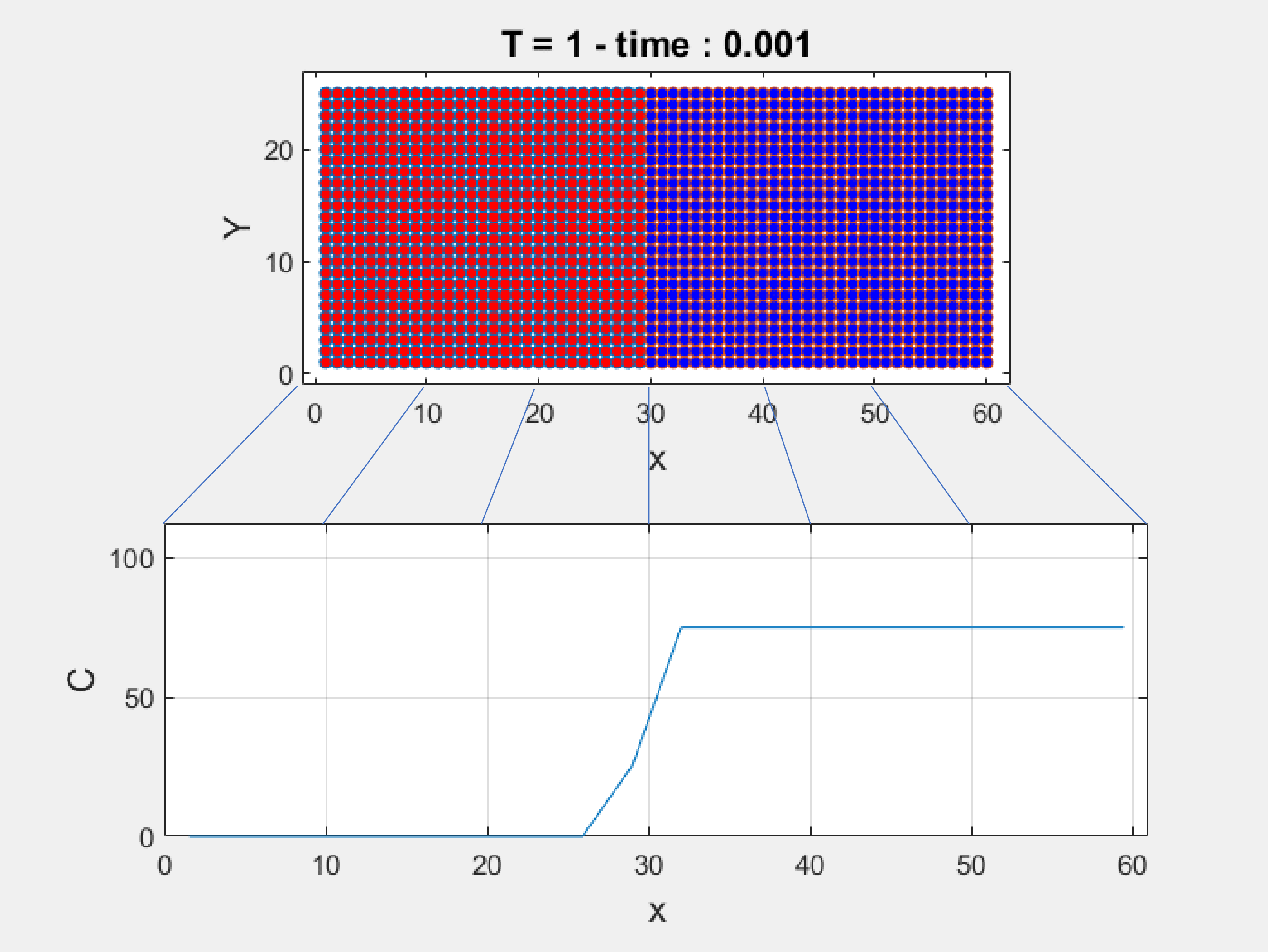
Introduction to Diffusion Equations

Molecular dynamics simulation for diffusion in a fluid.

In the animation to the right, note that the actual atoms go back and forth.

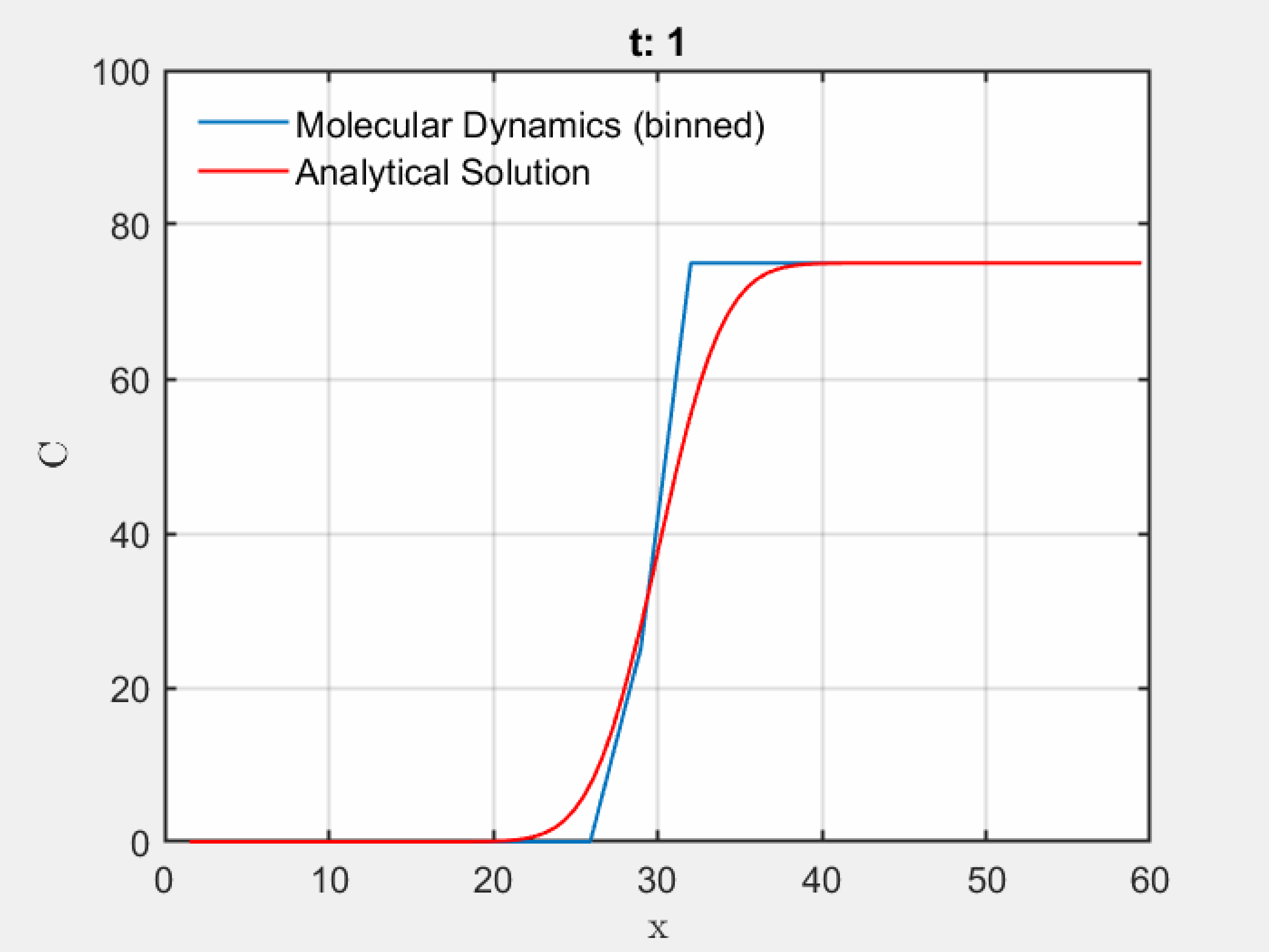
With time, there is a net movement of the material.

At the continuum level (below) we do not follow a specific particle, but instead, we describe how the concentration of a region changes.



Introduction to Diffusion Equations

Currently, we know the equations that govern the movement of atoms in the fluid (Newton's equations – classic MD). However, given the large numbers of particles involved, it is much easier to describe the macroscopic evolution of matter in a given element. For this reason we will use what we know from the physics of continuous matter.

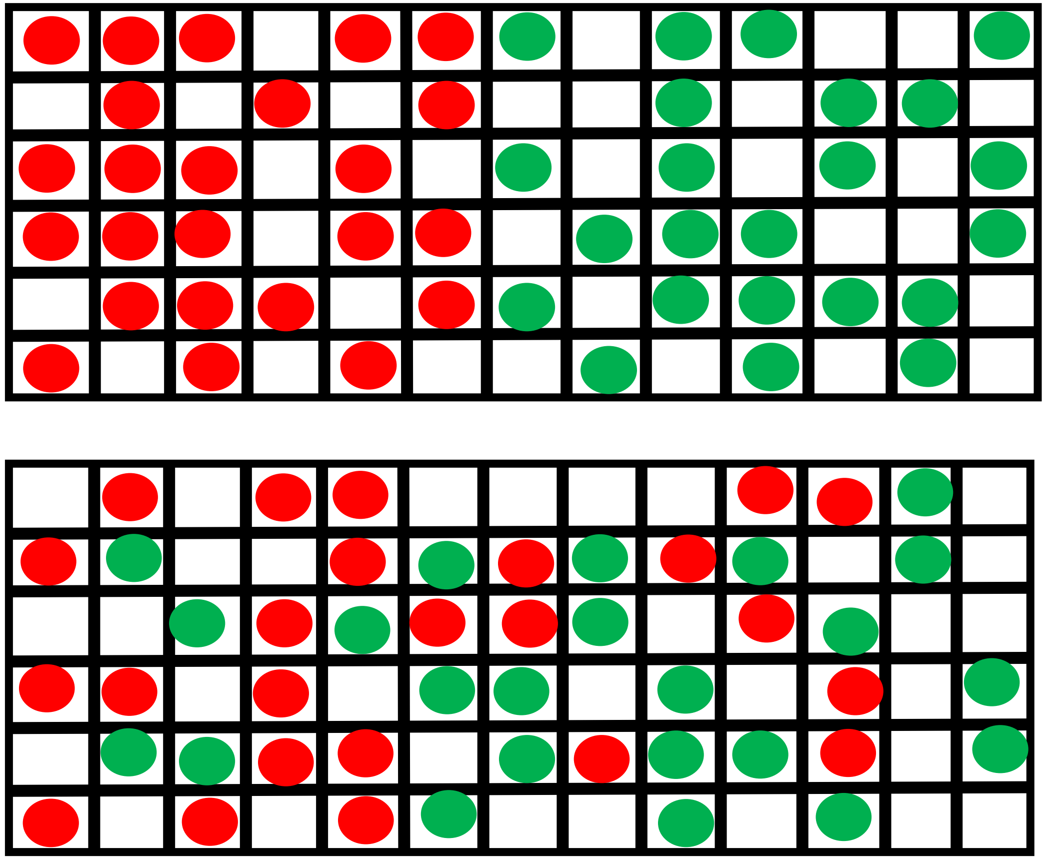


Introduction to Diffusion Equations

In solids, diffusion is assisted through vacancies and “similar” atoms can substitute each other in the lattice.

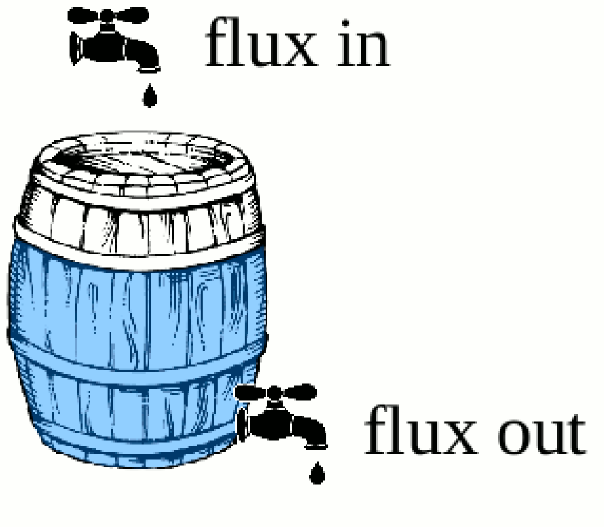
Again, the mixing of atoms is the result of random movements (or “jumps”) of the atoms. These jumps become more efficient at high temperatures.

The net result can also be described by a macroscopic flux.

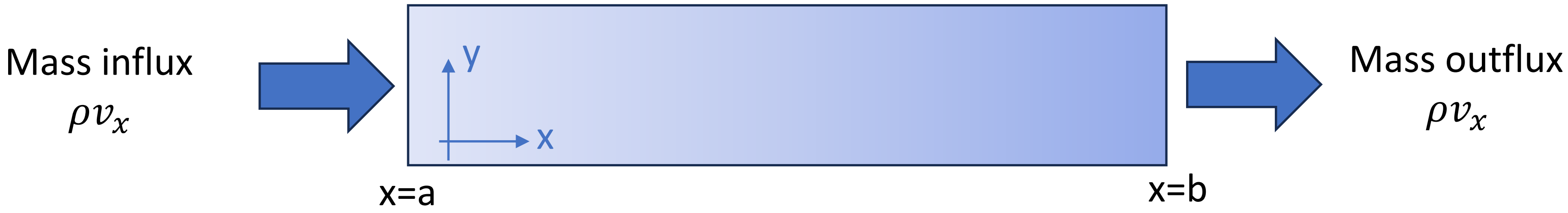


Diffusion Fundamentals

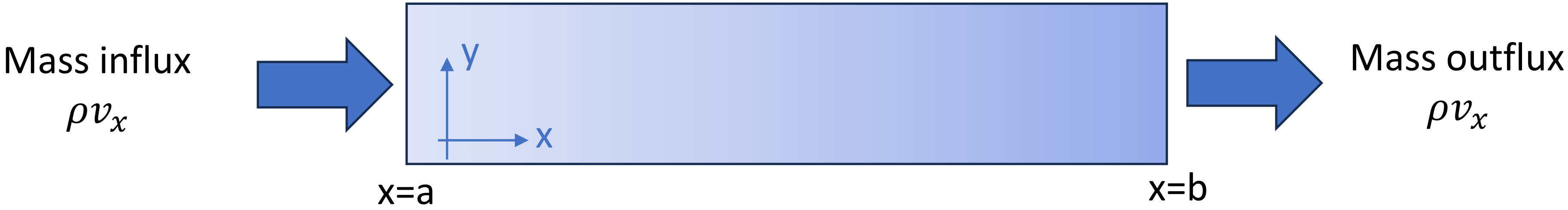
Conservation of mass states that **mass cannot be created or destroyed**. This is a more general statement than what is usually thought. In reality we can set up a problem based on mass balance without assuming that the mass of our system is constant. In other words, mass can travel around and does not have to remain constant at a given place.



To arrive to a more useful statement of mass conservation we will first consider the flow of mass in a 1-d slice of a continuum (e.g. from point a to point b).



Diffusion Fundamentals



Mass influx

ρv_x

Mass outflux

ρv_x

The conservation of mass is summarized as follows:

-The rate of mass change in an representative elementary volume is equal to the imbalance of the fluxes. Mathematically this is written as:

$$\frac{\partial}{\partial t}\int_a^b \rho\, dx = \rho(a,t)v_x(a,t) - \rho(b,t)v_x(b,t) \leftrightarrow \int_a^b \frac{\partial \rho}{\partial t} dx + [\rho v_x]_a^b \leftrightarrow$$

$$\int_a^b \frac{\partial \rho}{\partial t} dx + \int_a^b \frac{\partial(\rho v_x)}{\partial x} dx \leftrightarrow \boxed{\frac{\partial \rho}{\partial t} + \frac{\partial(\rho v_x)}{\partial x} = 0}$$

Diffusion Fundamentals

The conservation of mass can be generalized for every species, that is:

$$\frac{\partial \rho w^A}{\partial t} + \frac{\partial (\rho w^A v_x^A)}{\partial x} = 0$$

where w is the weight fraction of species A.
Note that the sum of weight fractions is 1, that is: $\sum_{i=1}^{c} w^i = 1$

For multiple species,
conservation of mass becomes:

$$\sum_{i=1}^{c} \frac{\partial \rho w^i}{\partial t} + \sum_{i=1}^{c} \frac{\partial \left(\rho w^i v_x^i \right)}{\partial x} = 0$$

Diffusion Fundamentals

Note that upon summation $\sum_{i=1}^{c} \rho w^i = \rho$ thus $\sum_{i=1}^{c} \frac{\partial \rho w^i}{\partial t} = \frac{\partial \rho}{\partial t}$

Similarly, the sum of velocities is: $\sum_{i=1}^{c} \rho w^i v_x^i = \rho v_x^{BC}$

where v_x^{BC} is the barycentric velocity. This is because for every point in space we can solve for velocity and observe that the result is a weighted average that uses weight fractions as weighting factors. That is (by definition):

$$v_x^{BC} = \sum_{i=1}^{c} w^i v_x^i$$

Diffusion Fundamentals

Using the previous definitions, we can recover the total mass balance after summation of the conservation of the individual species. For a given species A, the conservation law becomes:

$$\frac{\partial \rho w^A}{\partial t} + \frac{\partial(\rho w^A v_x^A)}{\partial x} = 0 \leftrightarrow \frac{\partial \rho w^A}{\partial t} + \frac{\partial\left(\rho w^A(v_x^A + v_x^{BC} - v_x^{BC})\right)}{\partial x} = 0$$

which can be written as:

$$\frac{\partial \rho w^A}{\partial t} + \frac{\partial(\rho w^A v_x^{BC})}{\partial x} + \frac{\partial\left(\rho w^A(v_x^A - v_x^{BC})\right)}{\partial x} = 0$$

Diffusion Fundamentals

In the previous equation, the 2nd term represents the bulk velocity (advection) and the last term represents the flow with respect to the bulk velocity (diffusion).

$$\frac{\partial \rho w^A}{\partial t} + \underbrace{\frac{\partial(\rho w^A v_x^{BC})}{\partial x}}_{\text{advection}} + \underbrace{\frac{\partial\left(\rho w^A (v_x^A - v_x^{BC})\right)}{\partial x}}_{\text{diffusion}} = 0$$

At this point we should see that the diffusive flux is: $J^A = \rho w^A (v_x^A - v_x^{BC})$

To a first approximation[1] the diffusive flux is proportional to the concentration gradient:

$$J^A = -\rho D \frac{\partial w^A}{\partial x}$$

[1] Strictly speaking it is proportional to chemical potential gradients, but we will not deal with that now

Diffusion Fundamentals

We thus have:

$$\frac{\partial \rho w^A}{\partial t} = D\frac{\partial^2 \rho w^A}{\partial x^2} - \frac{\partial(\rho w^A v_x^{BC})}{\partial x}$$

At the limit where the bulk velocity (advection) goes to zero and the density is constant, the previous can be written as:

$$\frac{\partial w^A}{\partial t} = D\frac{\partial^2 w^A}{\partial x^2}$$

The previous (not the general case) is also valid when w^A are molar fractions[2]. In this course, we will consider the general variable C that indicates mass fractions, mol fractions, or concentrations.

$$\boxed{\frac{\partial C}{\partial t} = D\frac{\partial^2 C}{\partial x^2}}$$

This is the main equation we are going to solve in this course.

[2] for details see Tajcmanova et al. (2021)

Diffusion Fundamentals

There are many analytical solutions for the diffusion equation. All these require knowledge for the initial and boundary conditions. The simplest case is the “error function” (half-space solution)

$$\frac{C(x,t) - C_{x=0}}{C_{inf} - C_{x=0}} = \mathrm{erf}(\frac{x}{2\sqrt{Dt}})$$

$$\mathrm{erf}(x) = \frac{1}{\sqrt{\pi}} \int_{-x}^{x} e^{-x'^2} dx'$$
$$= \frac{2}{\sqrt{\pi}} \int_{0}^{x} e^{-x'^2} dx'$$

where the value of “**erf**” is given as for any other function in MATLAB/OCTAVE etc. The derivation of the previous equation can be found in detail in Turcotte & Schubert (2014, p. 186-191) and in many other textbooks (Balluffi et al. 2005). We will just use it now.

Diffusion Fundamentals

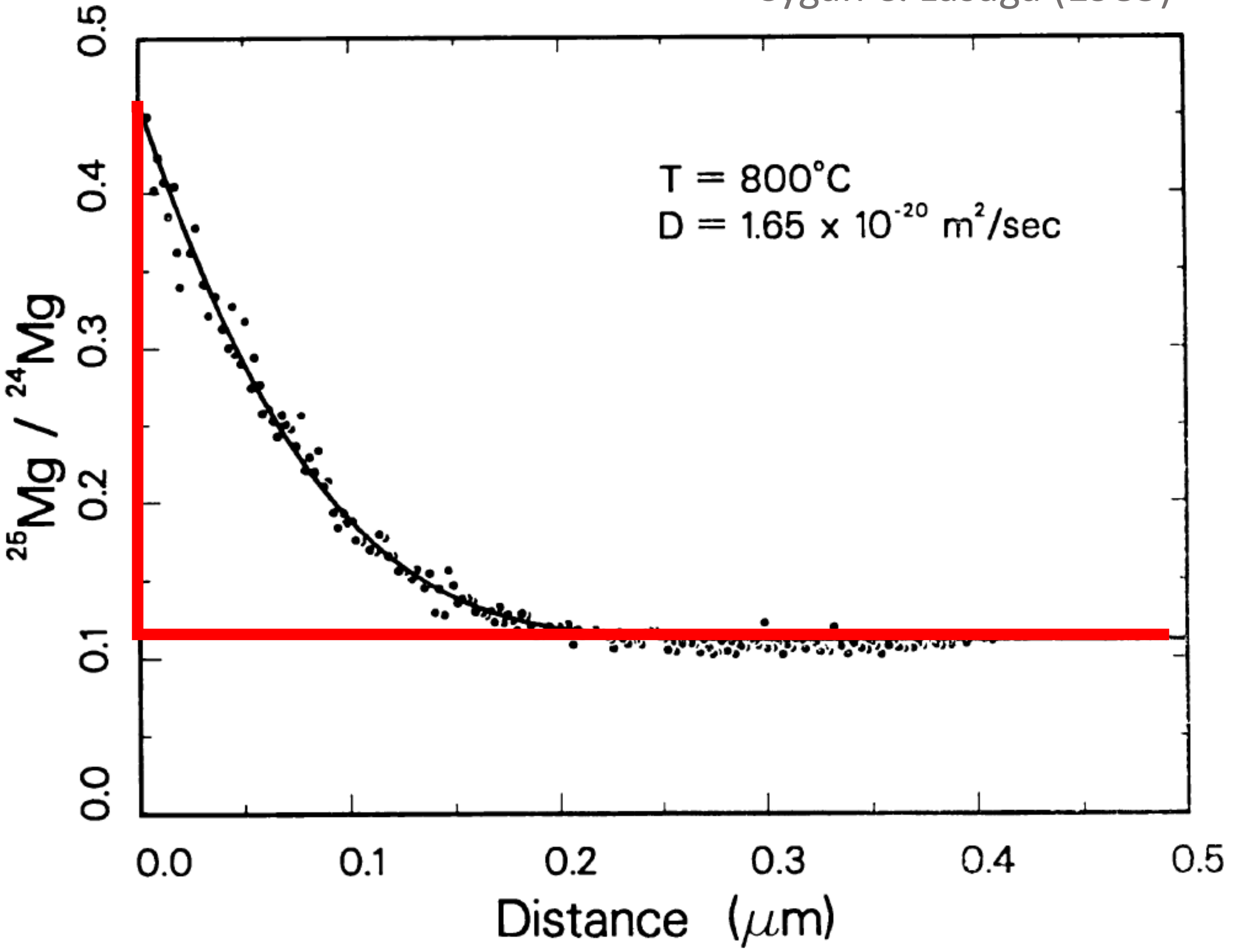
$$\frac{C(x,t) - C_{x=0}}{C_{inf} - C_{x=0}} = \mathrm{erf}(\frac{x}{2\sqrt{Dt}})$$

The red line indicates the initial condition (discontinuous at $x = 0$)

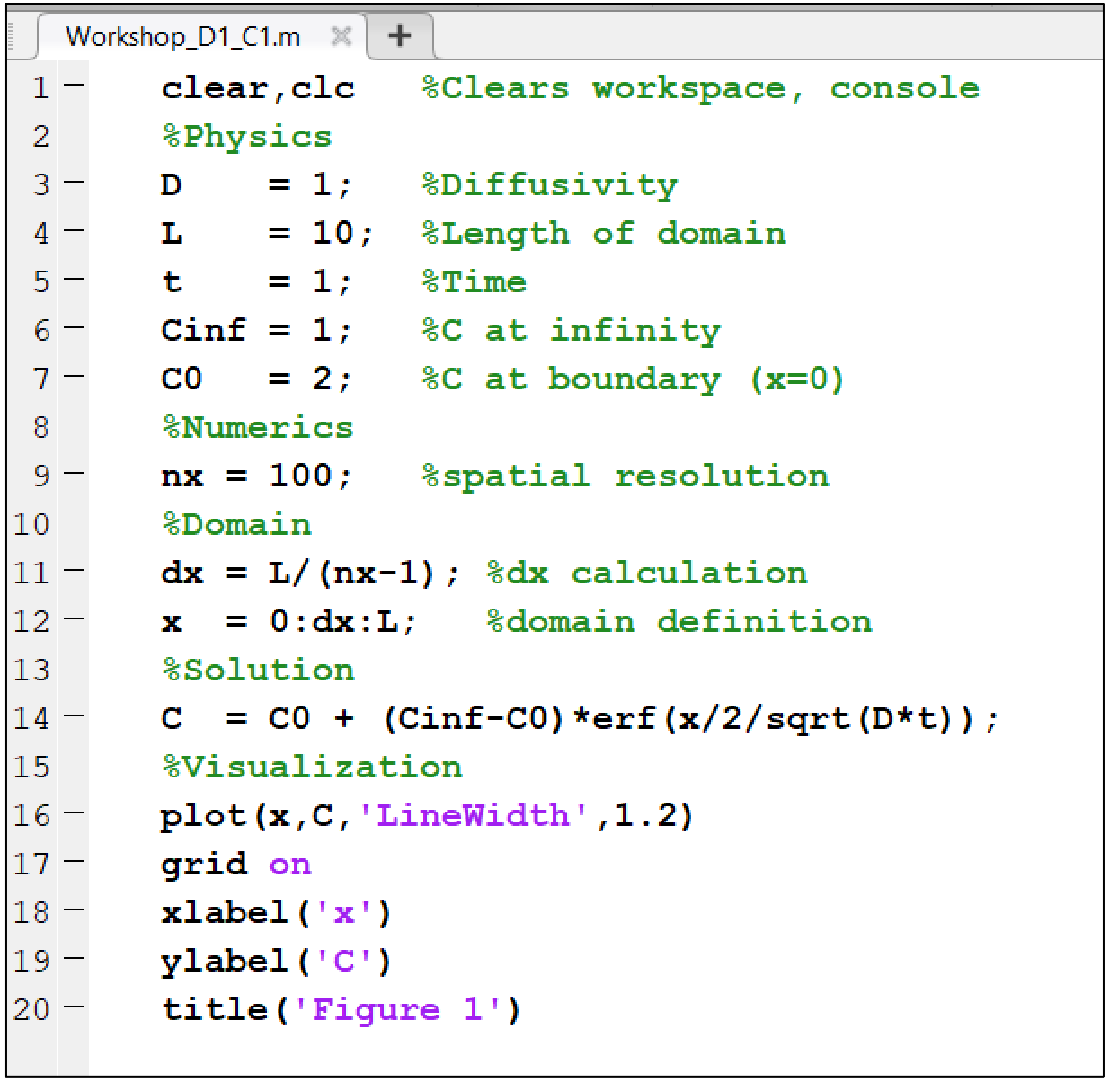
Exercise 1

Lets try to plot the solution!

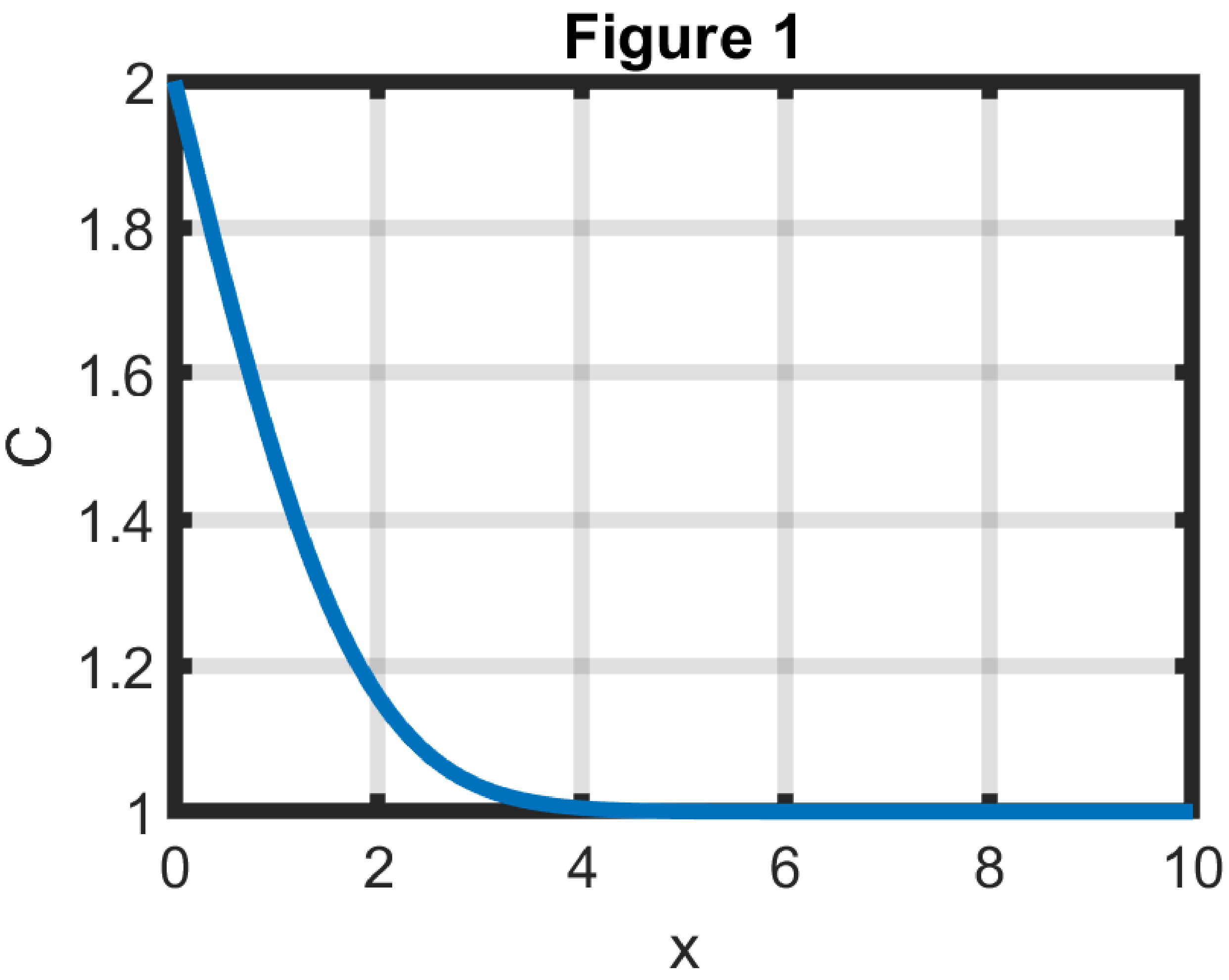
Cygan & Lasaga (1985)



Diffusion Fundamentals



```
clear,clc   %Clears workspace, console
%Physics
D    = 1;   %Diffusivity
L    = 10;  %Length of domain
t    = 1;   %Time
Cinf = 1;   %C at infinity
C0   = 2;   %C at boundary (x=0)
%Numerics
nx = 100;   %spatial resolution
%Domain
dx = L/(nx-1); %dx calculation
x  = 0:dx:L;   %domain definition
%Solution
C  = C0 + (Cinf-C0)*erf(x/2/sqrt(D*t));
%Visualization
plot(x,C,'LineWidth',1.2)
grid on
xlabel('x')
ylabel('C')
title('Figure 1')
```



Diffusion Fundamentals

An important feature of this analytical solution is that we can combine several “half-space solutions” (Note: This is because D is constant and the problem is linear)

$$C(x,t) = C_{x=0} + \left(C_{inf} - C_{x=0}\right)\mathrm{erf}\left(\frac{x}{2\sqrt{Dt}}\right)$$ classic

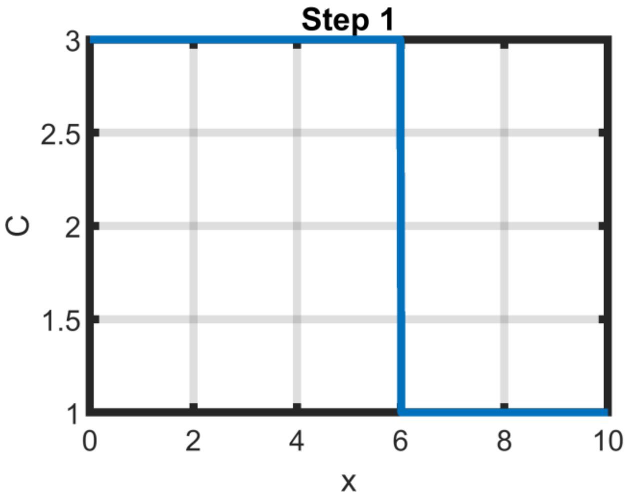
$$C(x,t) = C_{x=0} + \left(C_{inf} - C_{x=0}\right)\mathrm{erf}\left(\frac{x-H}{2\sqrt{Dt}}\right)$$ shifted by H to the right

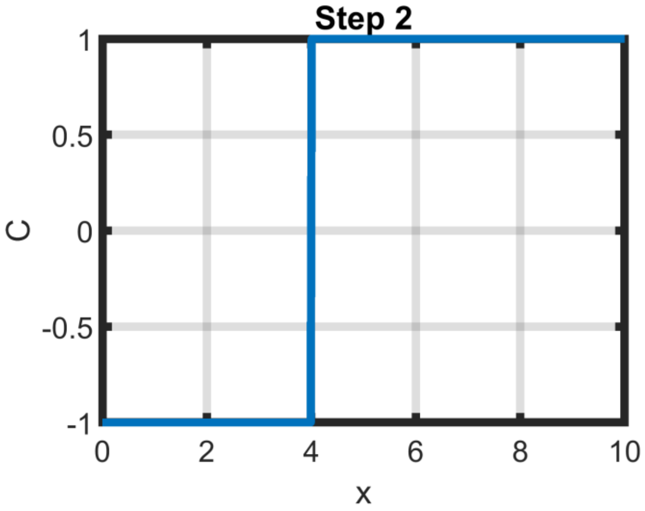
Diffusion Fundamentals

Let's consider now two specific solutions C_1 and C_2.

$$C_1 = C_{x=0} + (C_{inf} - C_{x=0})\mathrm{erf}(\frac{x-6H}{2\sqrt{Dt}})$$

$$C_2 = 0 + (C_{x=0} - C_{inf})\mathrm{erf}(\frac{x-4H}{2\sqrt{Dt}})$$



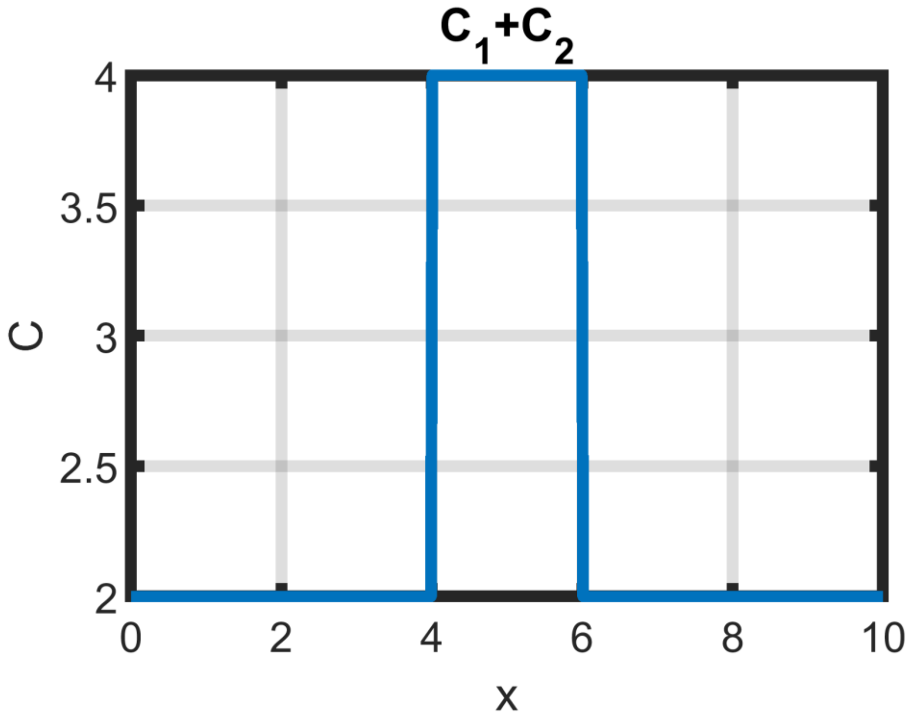


Diffusion Fundamentals

We can now add the two specific solutions. The result is still a solution (it works when the diffusion coefficient is constant).

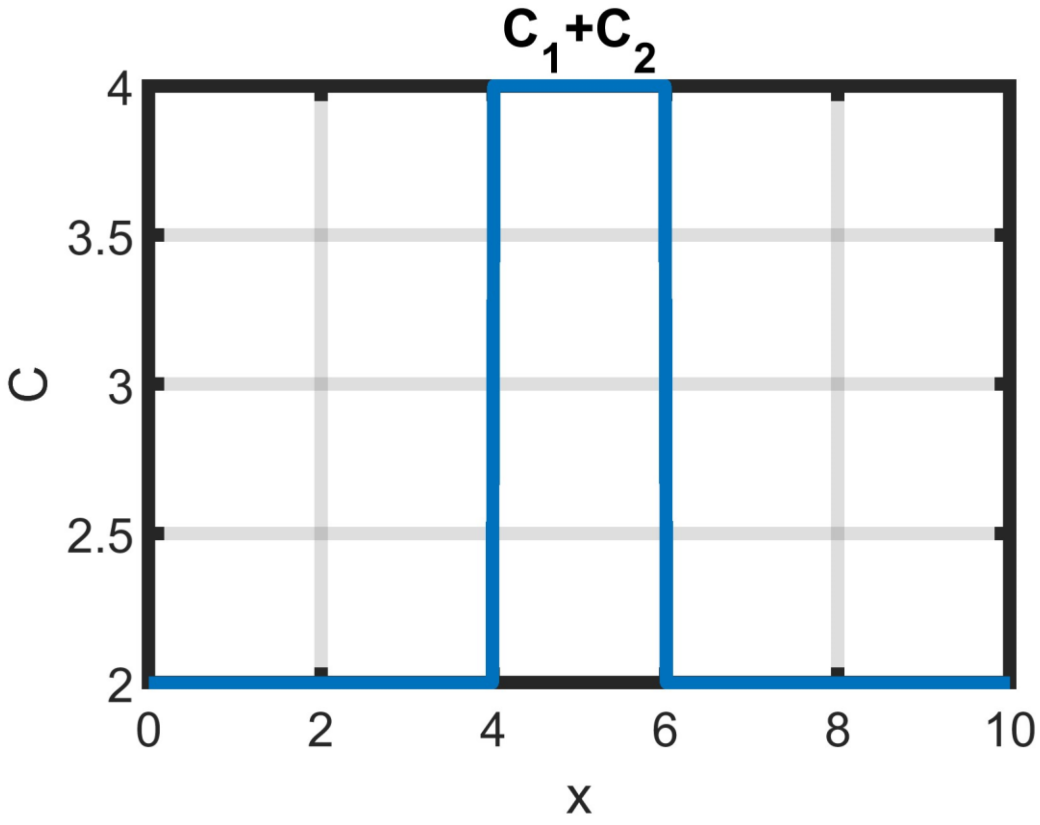
$$C = C_1 + C_2$$

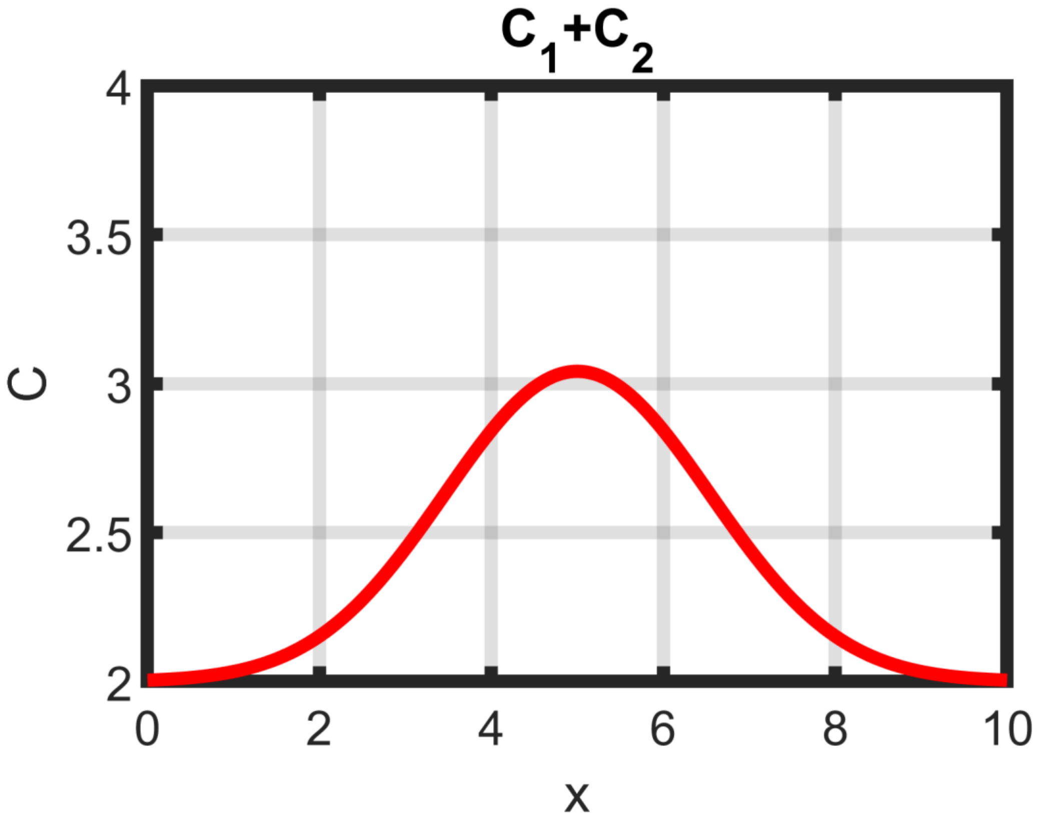
Note that this method assumes that the boundaries are at infinity (+/-). For practical purposes, we will set the step in the center of the domain and we will not consider results where the concentration changes (appreciably) at the boundaries



Diffusion Fundamentals

Another important feature is that when t approaches 0 the solution is not defined. You should then use a very small number to plot it (i.e. $t \sim 10^{-10}$)





Diffusion Fundamentals

Exercise 2

Plot the combined step as a function of time. C_1 and $C_2$are given below. You can chose arbitrary values for $x, D, t, H, C_{x=0}, C_{inf}$.

$$C_1 = C_{x=0} + \left(C_{inf} - C_{x=0}\right)\mathrm{erf}\left(\frac{x - 6H}{2\sqrt{Dt}}\right)$$

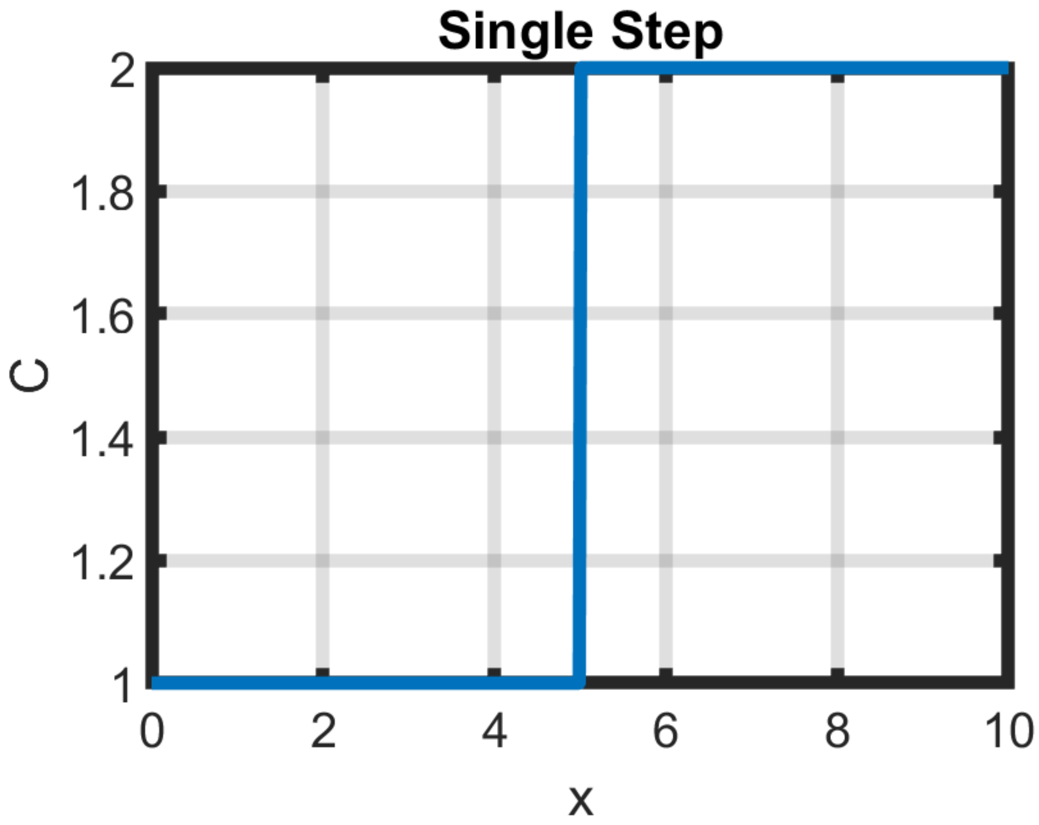
$$C_2 = 0 + \left(C_{x=0} - C_{inf}\right)\mathrm{erf}\left(\frac{x - 4H}{2\sqrt{Dt}}\right)$$

Diffusion Fundamentals

A more compact formula for a single step is:

$$C(x,t) = \frac{C_r + C_l}{2} + \frac{C_r - C_l}{2}\,\mathrm{erf}\left(\frac{x - x_s}{2\sqrt{Dt}}\right)$$

where C_r is the concentration value at the right side (2), C_l is the concentration value at the left side (1) and x_s is the location of the step (5).

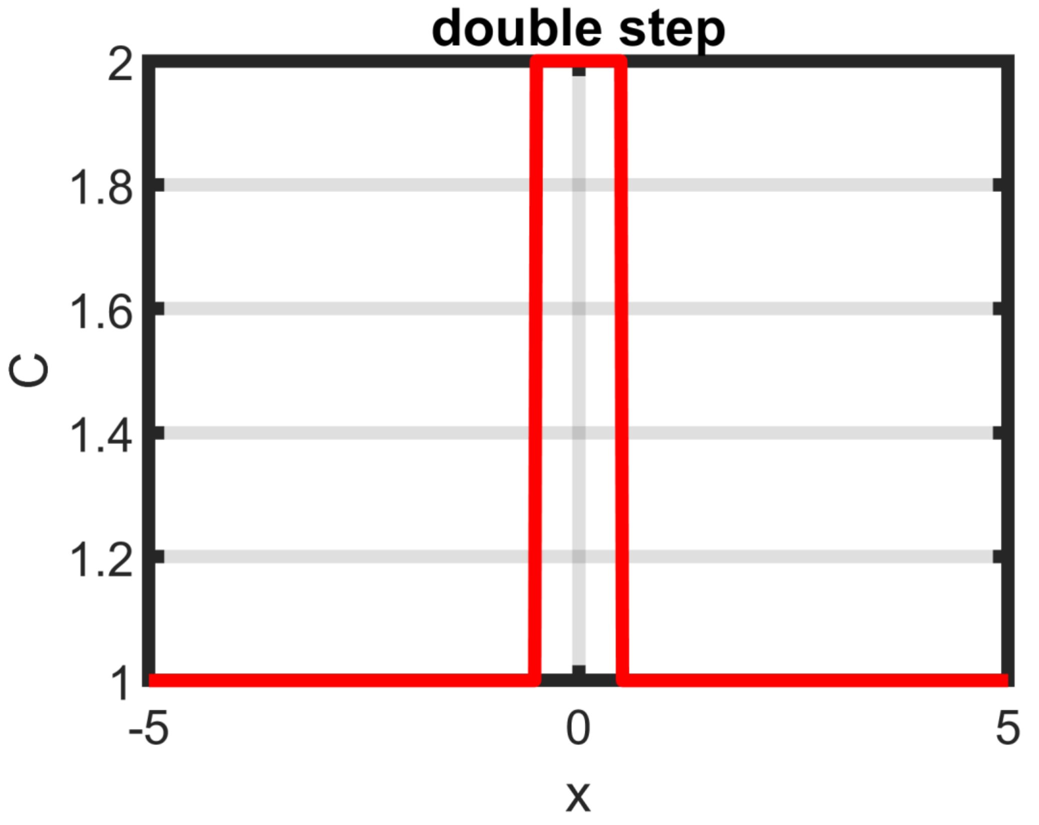


Diffusion Fundamentals

A more general form to describe a ‘square-wave’ (combination of 2 steps) is:

$$C(x,t) = C_{BG} + \frac{\Delta C}{2}\left[\mathrm{erf}\left(\frac{x+\frac{H}{2}}{2\sqrt{Dt}}\right) - \mathrm{erf}\left(\frac{x-\frac{H}{2}}{2\sqrt{Dt}}\right)\right]$$

where C_{BG} is a background value (e.g. 1), ΔC is the difference from this background (e.g. 1), and H is the thickness of the initial slab (e.g. 1). Note that the step is centered at $x = 0$.



Diffusion Fundamentals

The advantage of the analytical solution is that it is very fast to calculate. If we want to plot the solution while it is being calculated we have to use a ‘for’ or a ‘while’ loop.

We can start with a given timestep and update time as follows:

$$t^k = t^{k-1} + \Delta t$$

Then, for every time the solution can be evaluated.

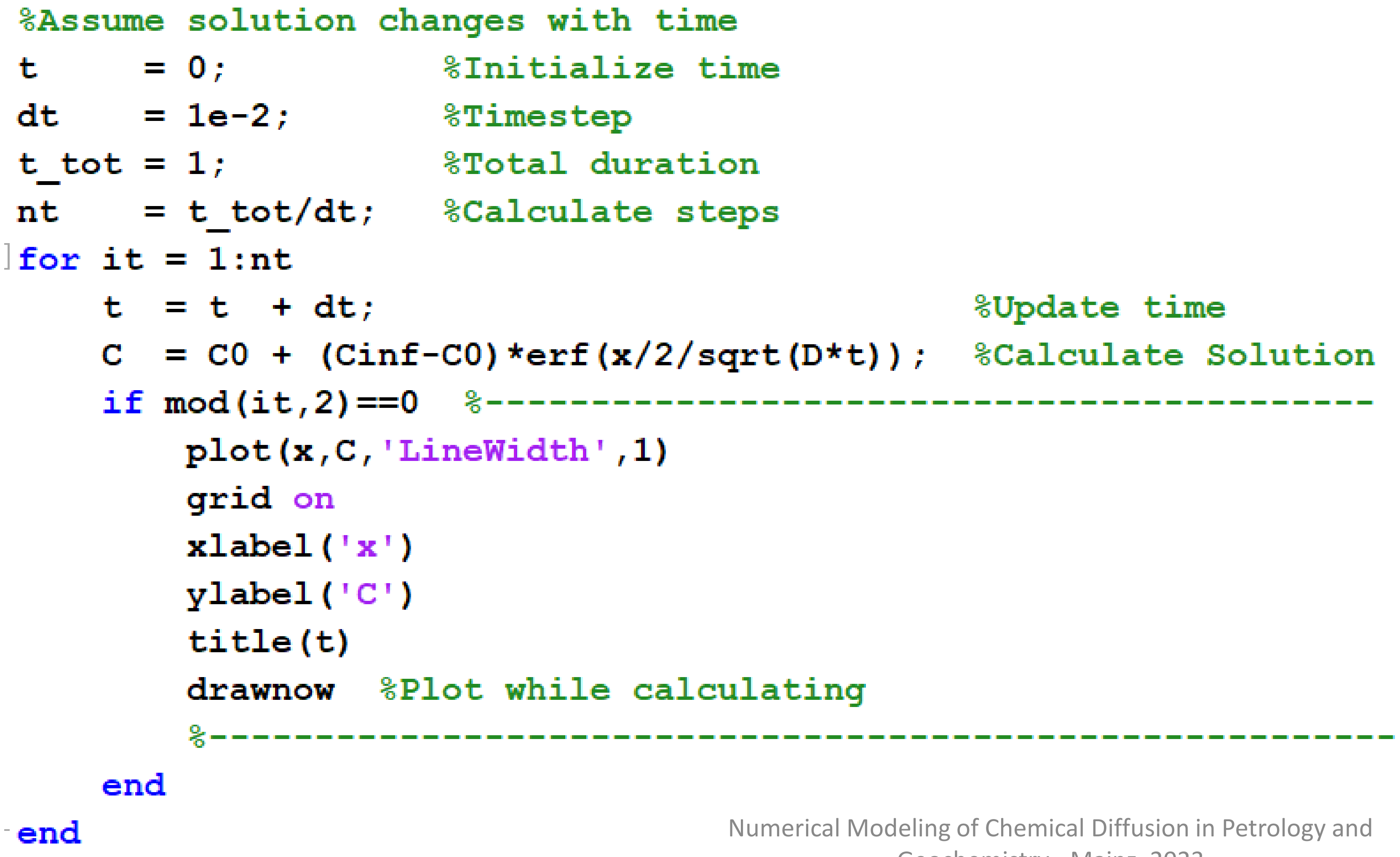
$$C(x,t) = C_{x=0} + \left(C_{inf} - C_{x=0}\right)\mathrm{erf}\left(\frac{x}{2\sqrt{D\boldsymbol{t^k}}}\right)$$

Update t in every iteration

Diffusion Fundamentals

Example 1

...

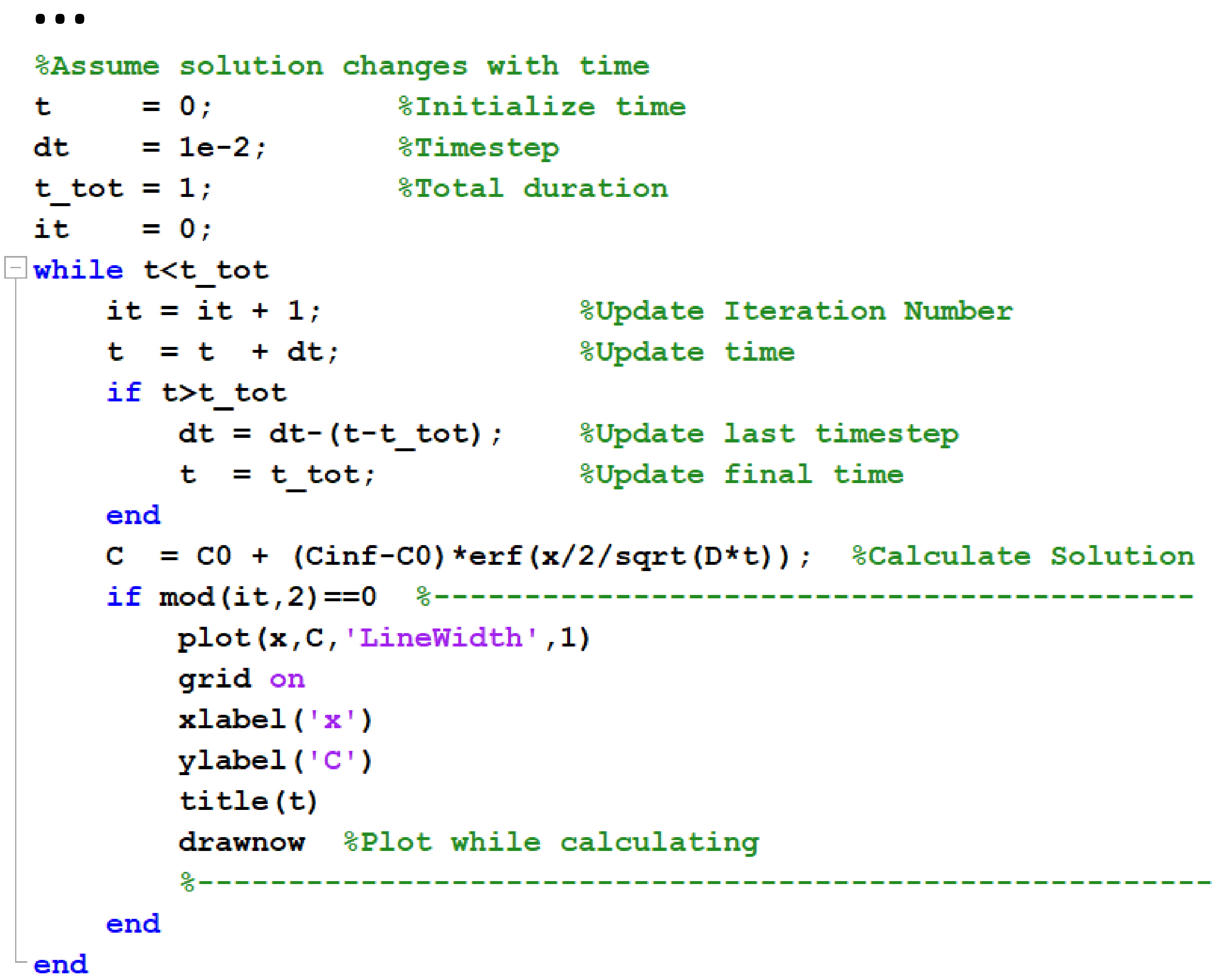


```
%Assume solution changes with time
t     = 0;          %Initialize time
dt    = 1e-2;       %Timestep
t_tot = 1;          %Total duration
nt    = t_tot/dt;   %Calculate steps
for it = 1:nt
    t  = t  + dt;                              %Update time
    C  = C0 + (Cinf-C0)*erf(x/2/sqrt(D*t));  %Calculate Solution
    if mod(it,2)==0  %------------------------------------------
        plot(x,C,'LineWidth',1)
        grid on
        xlabel('x')
        ylabel('C')
        title(t)
        drawnow  %Plot while calculating
        %--------------------------------------------------------
    end
end
```

Note that we do not need to plot during every step (we save some computation time)

Diffusion Fundamentals

Example 2



```
...
%Assume solution changes with time
t     = 0;          %Initialize time
dt    = 1e-2;       %Timestep
t_tot = 1;          %Total duration
it    = 0;
while t<t_tot
    it = it + 1;             %Update Iteration Number
    t  = t  + dt;            %Update time
    if t>t_tot
        dt = dt-(t-t_tot);   %Update last timestep
        t  = t_tot;          %Update final time
    end
    C  = C0 + (Cinf-C0)*erf(x/2/sqrt(D*t));  %Calculate Solution
    if mod(it,2)==0  %------------------------------------------
        plot(x,C,'LineWidth',1)
        grid on
        xlabel('x')
        ylabel('C')
        title(t)
        drawnow  %Plot while calculating
        %--------------------------------------------------------
    end
end
```

In case of a fixed dt, we need a correction in case the timestep overshoots

trology and

Introduction to Numerical Modelling

The finite difference approximation

In reality, we may need a solution for more general initial and boundary conditions. In those cases, we need to proceed with numerical techniques. The starting point is the Finite-Difference-Method (FDM) where the derivatives are approximated by finite differences. This is in fact the simplest method but it is very robust for simple diffusion problems.

As a starting point we will consider the following Ordinary Differential Equation (ODE):

$$\frac{dC}{dt} = -kC$$

where we have one variable that changes with time only. The previous can be discretized as:

$$\frac{\Delta C}{\Delta t} = -kC \leftrightarrow \frac{C^{new} - C^{old}}{\Delta t} = -kC^{old}$$

The finite difference approximation

The previous can be solved for the “new” value of C as follows (forward Euler Method):

$$C^{new} = C^{old} - \Delta t k C^{old}$$

Exercise 3

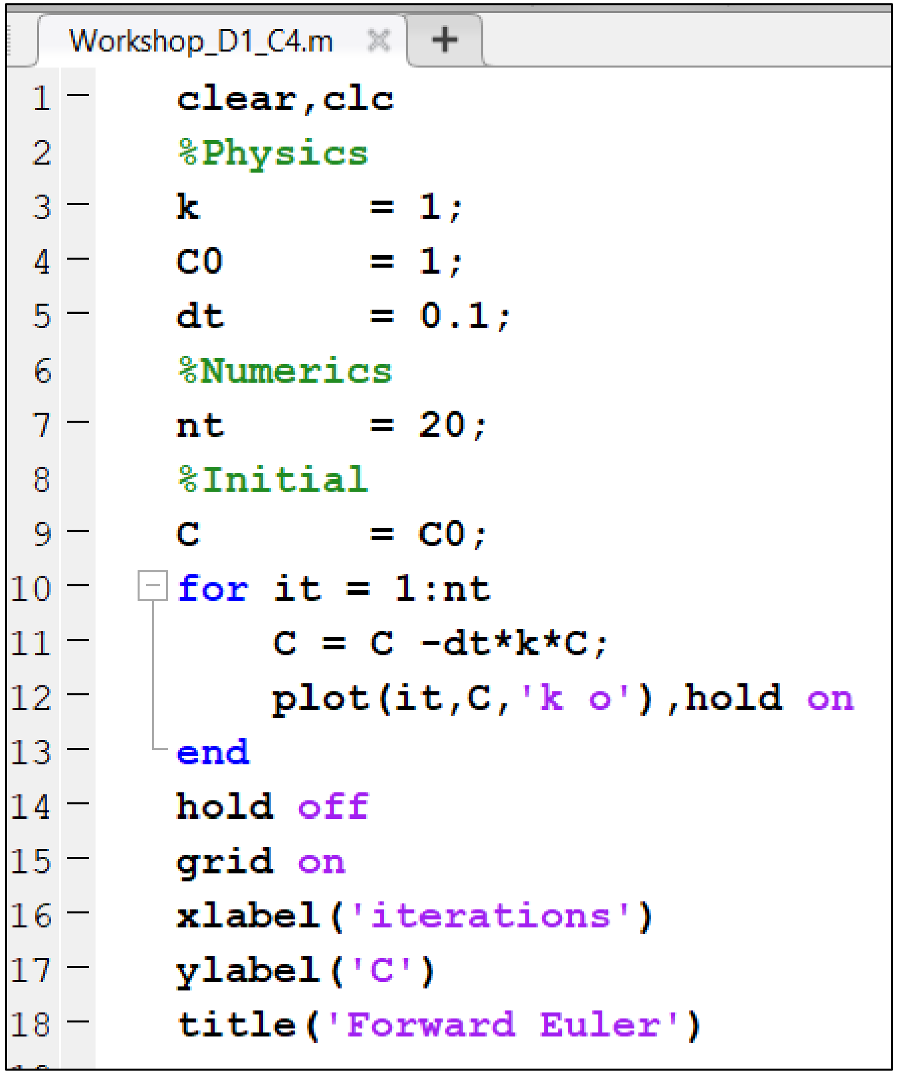
Lets try to calculate the solution for $k = 1, \ \Delta t = 0.1$ and $C = 1$ at $t = 0$

Exercise 4

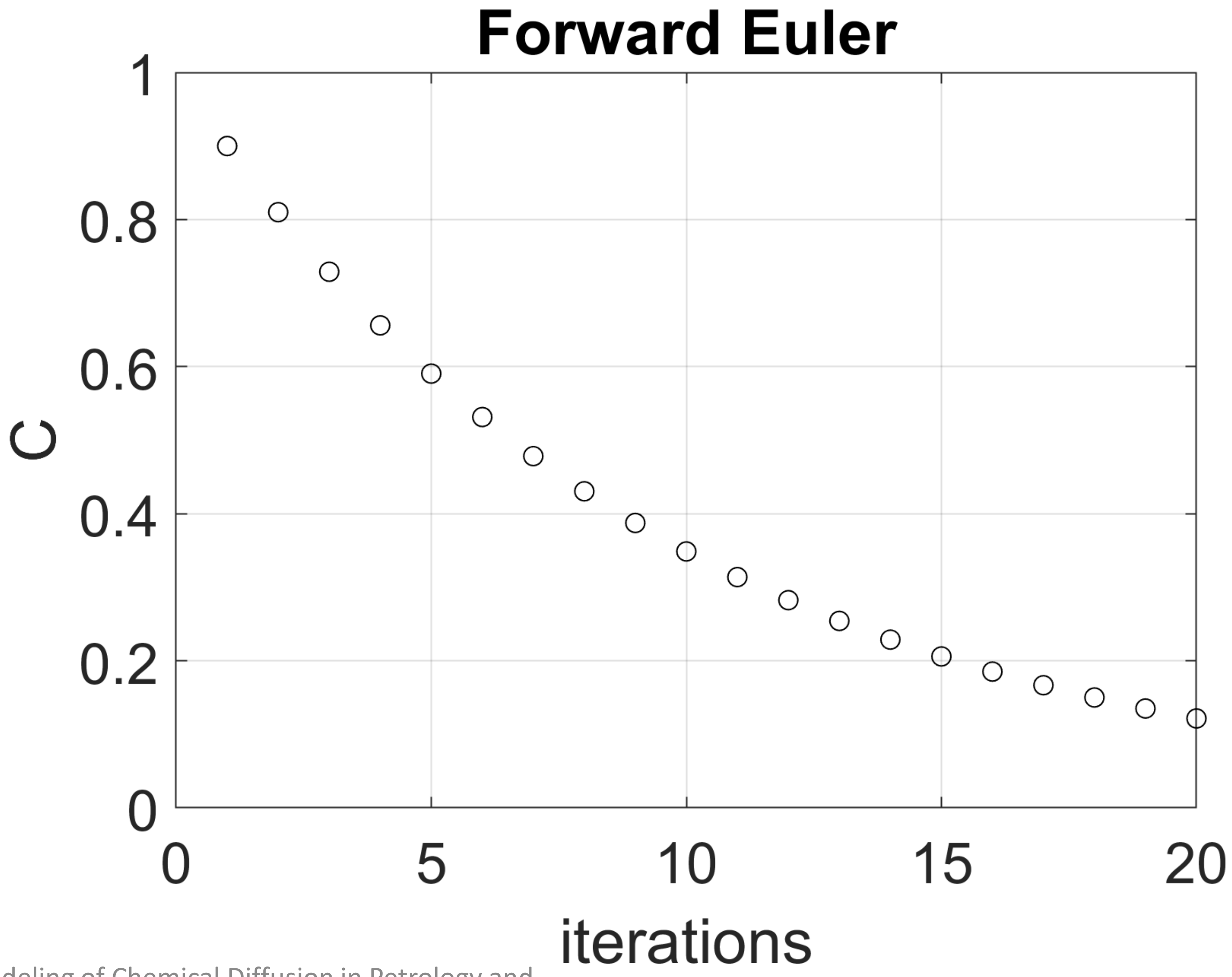
Try to change Δt in the previous example, what do you observe?

In the forward Euler method, the unknown is solved by using the previous value and by adding a correction. This method is also called “explicit”

The finite difference approximation



```
clear,clc
%Physics
k         = 1;
C0        = 1;
dt        = 0.1;
%Numerics
nt        = 20;
%Initial
C         = C0;
for it = 1:nt
    C = C -dt*k*C;
    plot(it,C,'k o'),hold on
end
hold off
grid on
xlabel('iterations')
ylabel('C')
title('Forward Euler')
```



The finite difference approximation

We can start again with the original discretized equation. However, we will now assume that the value of the Right-Hand-Side (RHS) is the (yet unknown) "new" value (backward Euler Method):

$$\frac{C^{new}-C^{old}}{\Delta t} = -kC^{new} \leftrightarrow C^{new} - C^{old} = -k\Delta t C^{new} \leftrightarrow$$

$$C^{new} + k\Delta t C^{new} = C^{old} \leftrightarrow$$

$$\underbrace{(1 + k\Delta t)}_{\mathrm{A}}\,\underbrace{C^{new}}_{\mathrm{x}} = \underbrace{C^{old}}_{\mathrm{b}} \leftrightarrow \mathrm{x} = \mathrm{A}^{-1}\mathrm{b}$$

we can solve for C^{new} if we have **as many independent equations as unknowns.**

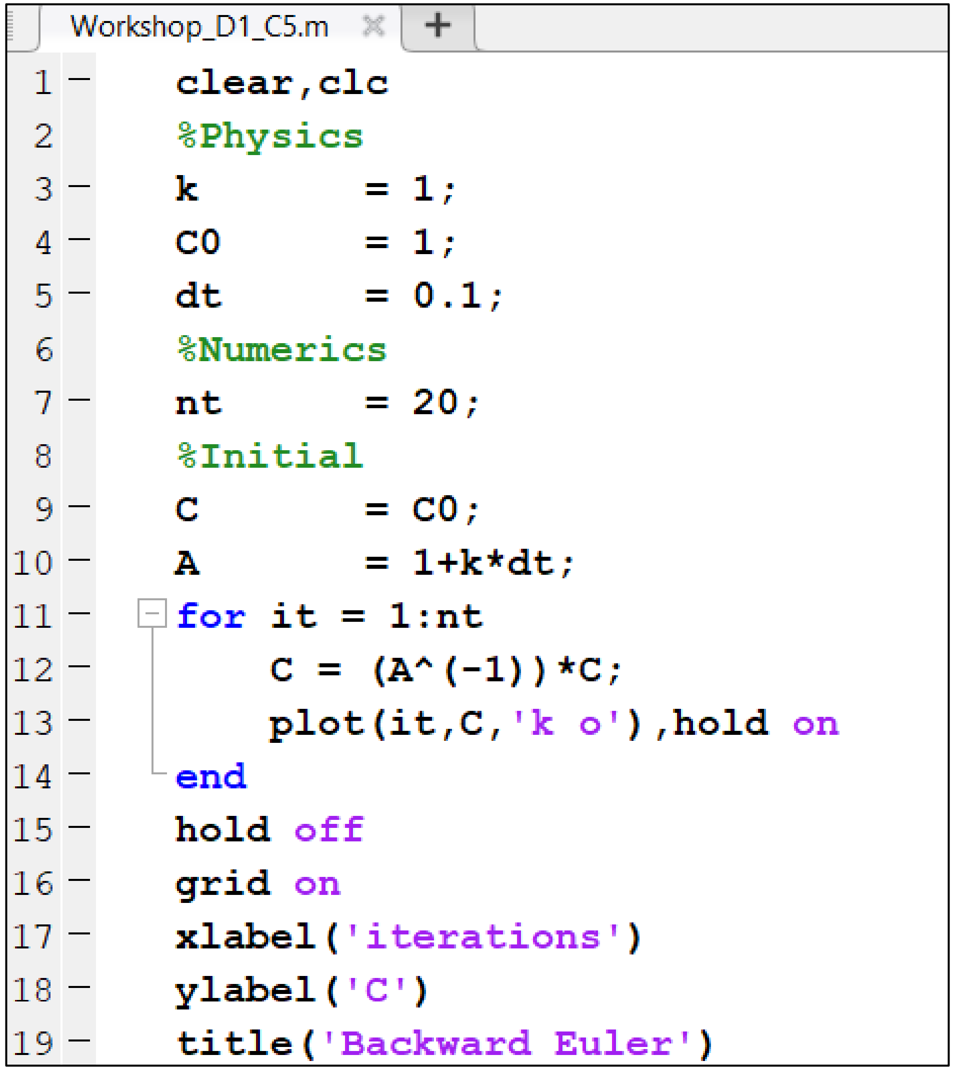
The finite difference approximation

Exercise 5

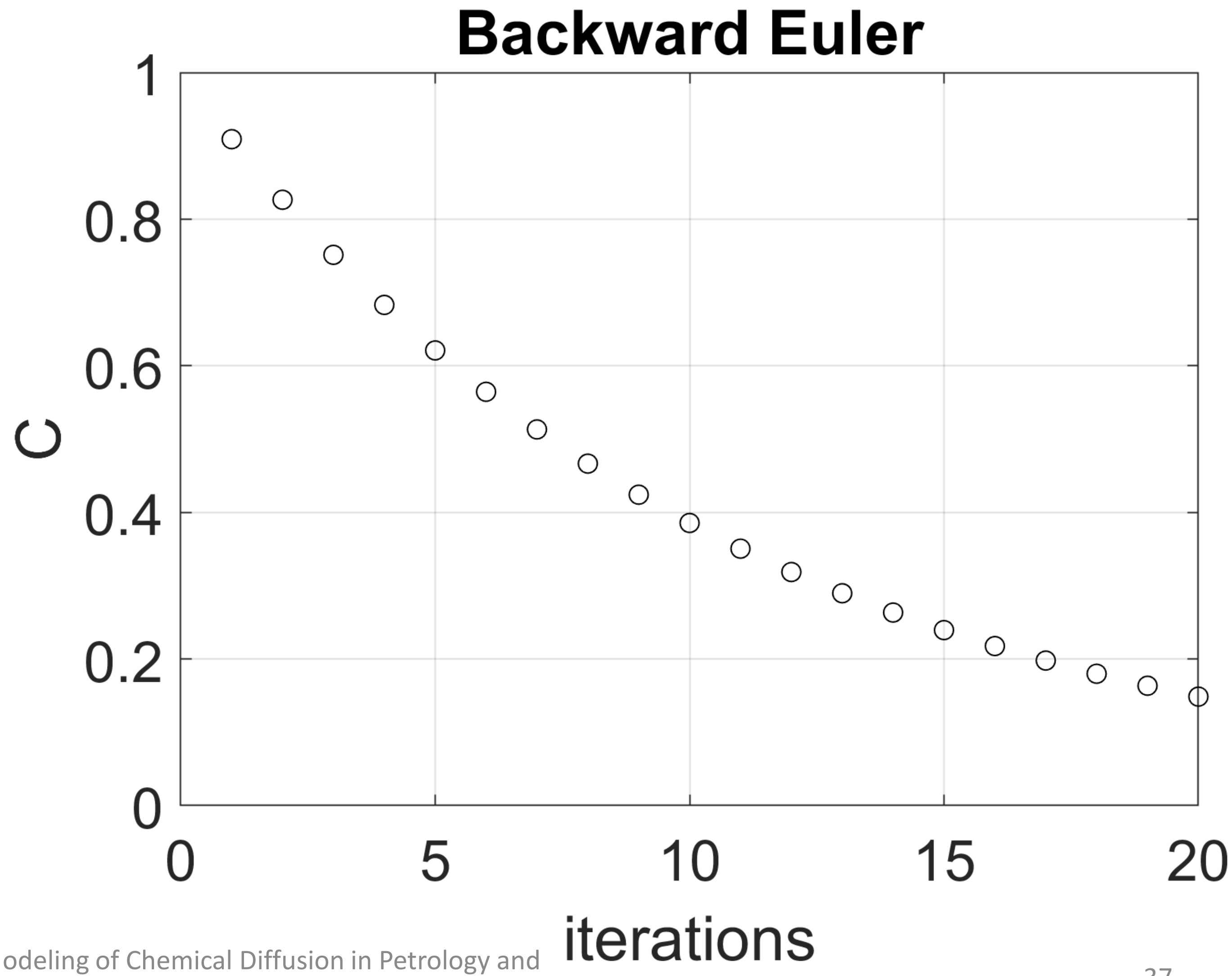
Program the backward Euler method (implicit method) and compare the result with the forward Euler method. The solution can be calculated by the following form:

$$\underbrace{C^{new}}_{\mathrm{x}} = \underbrace{(1 + k\Delta t)^{-1}}_{A^{-1}} \underbrace{C^{old}}_{\mathrm{b}}$$

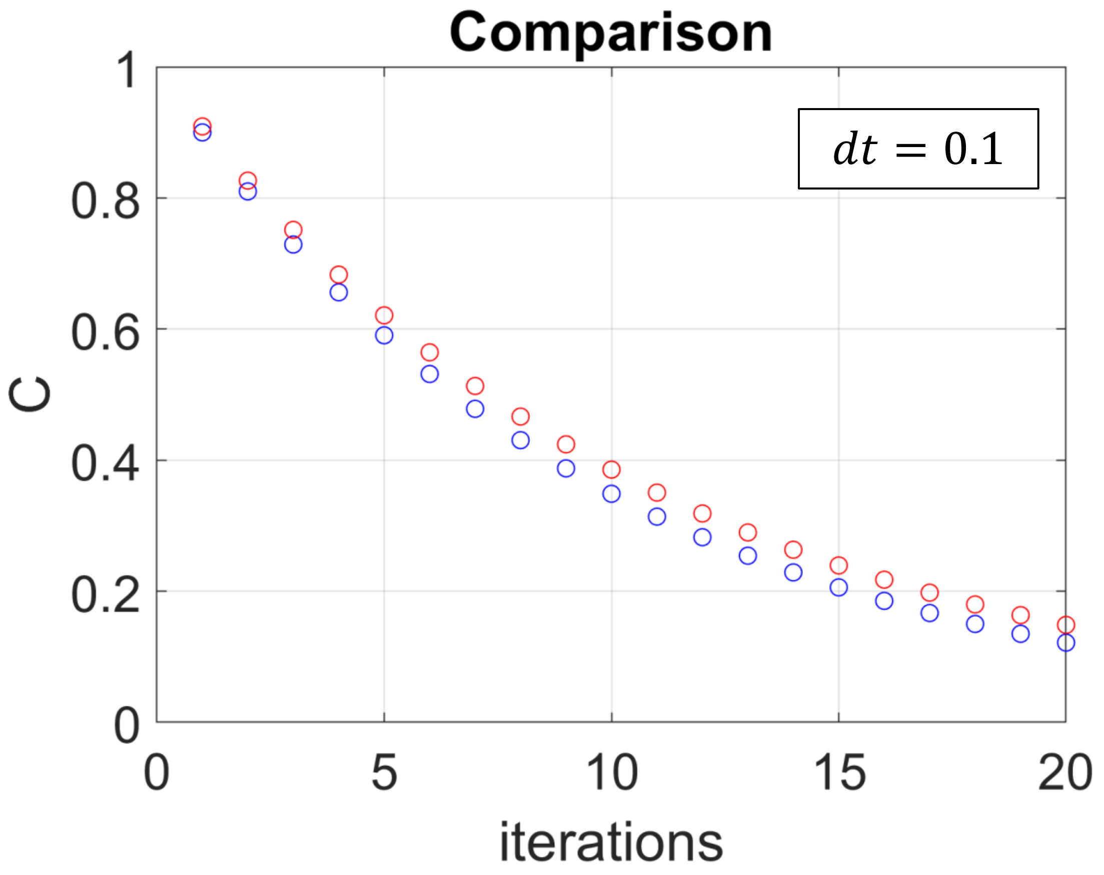
The finite difference approximation

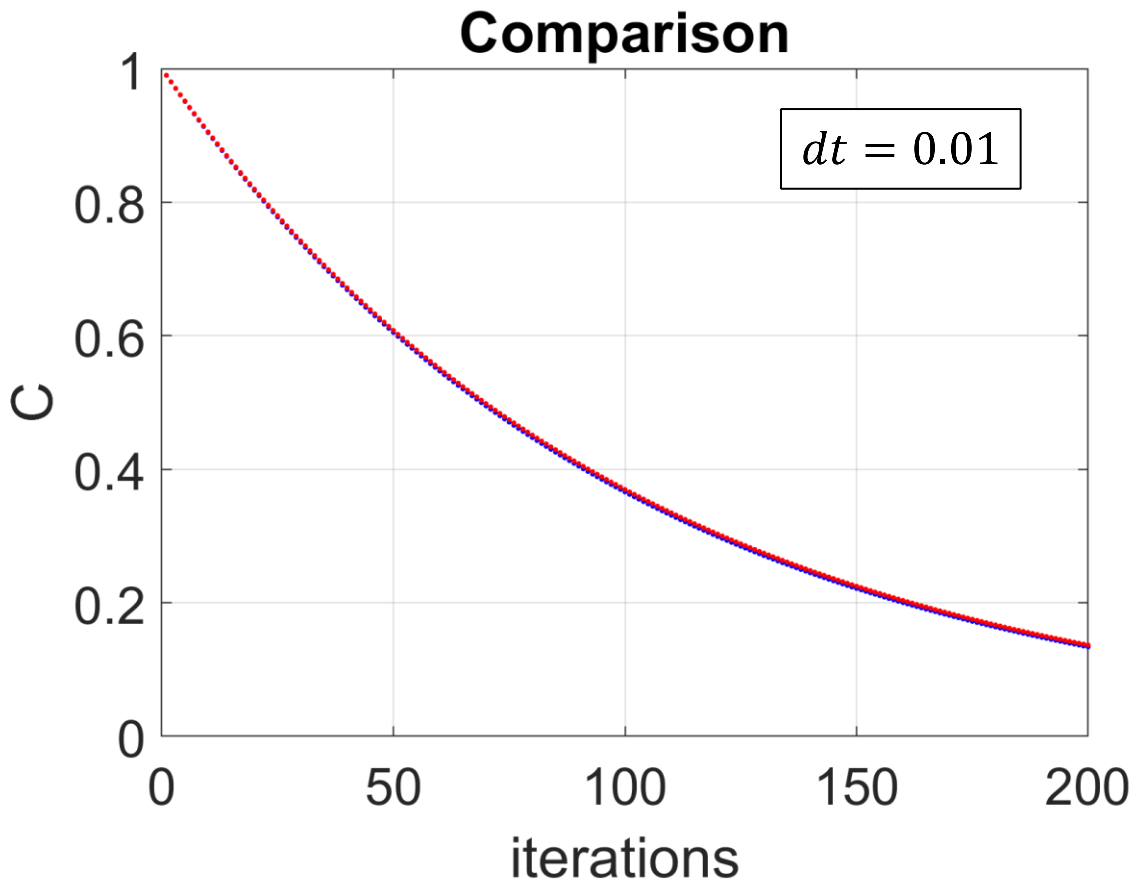


```
clear,clc
%Physics
k          = 1;
C0         = 1;
dt         = 0.1;
%Numerics
nt         = 20;
%Initial
C          = C0;
A          = 1+k*dt;
for it = 1:nt
    C = (A^(-1))*C;
    plot(it,C,'k o'),hold on
end
hold off
grid on
xlabel('iterations')
ylabel('C')
title('Backward Euler')
```



The finite difference approximation



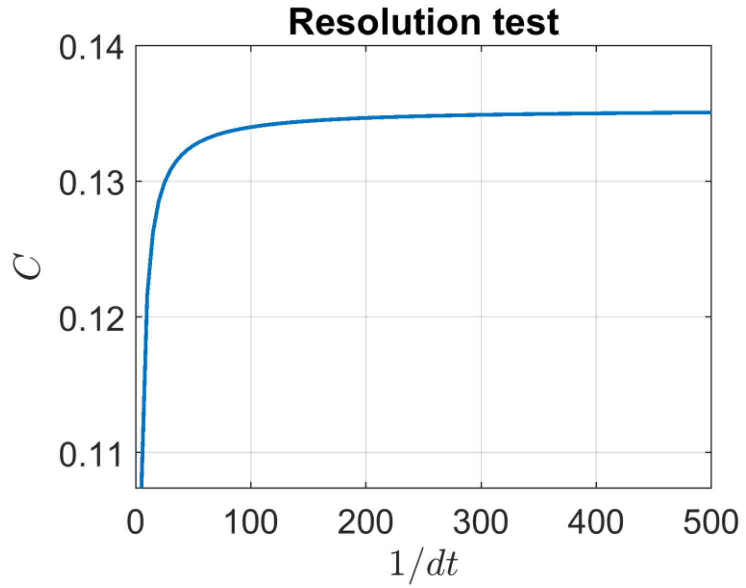


The finite difference approximation

Most important rule in numerical modelling!

A numerical solution must converge when the numerical resolution is increasing (this indicates that our solution converges to the true solution, provided the solution exists)

To test for convergence, we can plot a given calculated value (e.g. value of C after time $t = 2$) as a function of the numerical resolution. The figure on the right shows that the solution converges to a single value when dt becomes smaller (or $1/dt$ becomes larger)



Time derivatives and integration

In the previous examples there is a pattern with respect to time integration.

Explicit Method[1]

$$\mathrm{x^n = x^o + dt \cdot R}$$

[1] The explicit formulation can also be rewritten in the form of $\mathrm{x^n = A^{-1}b}$

Implicit Method

$$\mathrm{x^n = A^{-1}b}$$

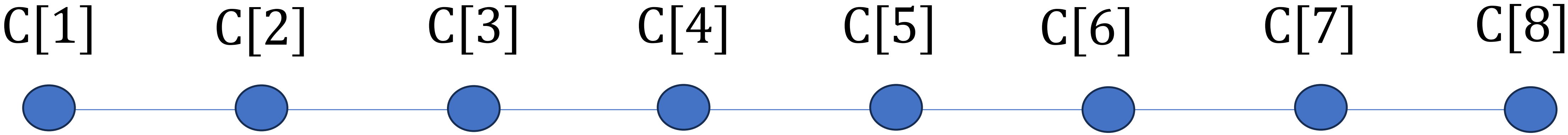
where

$$\mathrm{A} = 1 + k\Delta t$$
$$\mathrm{b = x^o}$$

Spatial derivatives

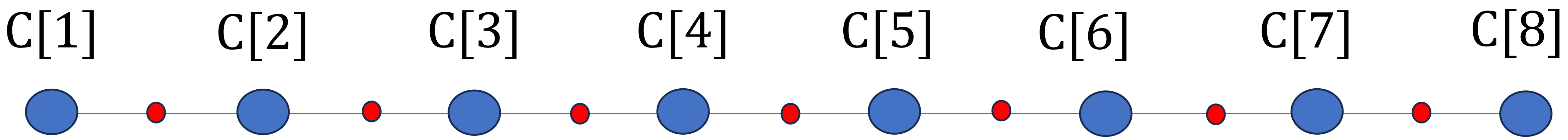
Lets go back to our main problem, the diffusion equation: $$\frac{\partial C}{\partial t} = D\,\frac{\partial^2 C}{\partial x^2}$$

Apart from the time derivatives, that we saw earlier, we need to discretize also the spatial derivatives. However, the concentration (C), is not a continuous function and it is known only in a few discrete places (here 8; see figure below):



Thus, the problem we have to solve first is:
How to calculate derivatives in space using the Finite-Difference Method?

Spatial derivatives



$$\frac{\partial C}{\partial x} \approx \frac{C_{i+1} - C_i}{\Delta x}$$

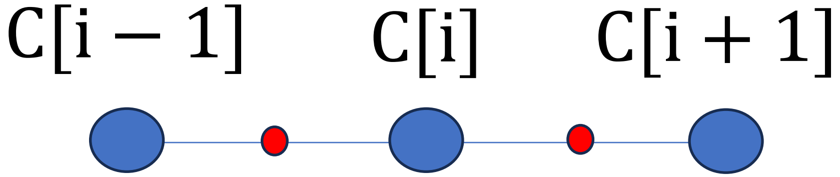
The derivatives in space are good approximations for the midpoint (red points) of the numerical grid.

Having a way to deal with derivatives in space and in time allows us to combine the two and approximate a Partial Differential Equation (PDE).

The finite difference approximation

Therefore, for the case of diffusion equation we have:

$$\frac{\partial C}{\partial t} = D\frac{\partial^2 C}{\partial x^2}$$



C[i − 1] C[i] C[i + 1]

The previous can be discretized using the explicit method as:

$$\frac{C_i^{new} - C_i^{old}}{\Delta t} = D\frac{1}{\Delta x}\left(\frac{\left(C_{i+1}^{old} - C_i^{old}\right)}{\Delta x} - \frac{\left(C_i^{old} - C_{i-1}^{old}\right)}{\Delta x}\right)$$

or if we assume uniform grid and solve for C_i^{new} :

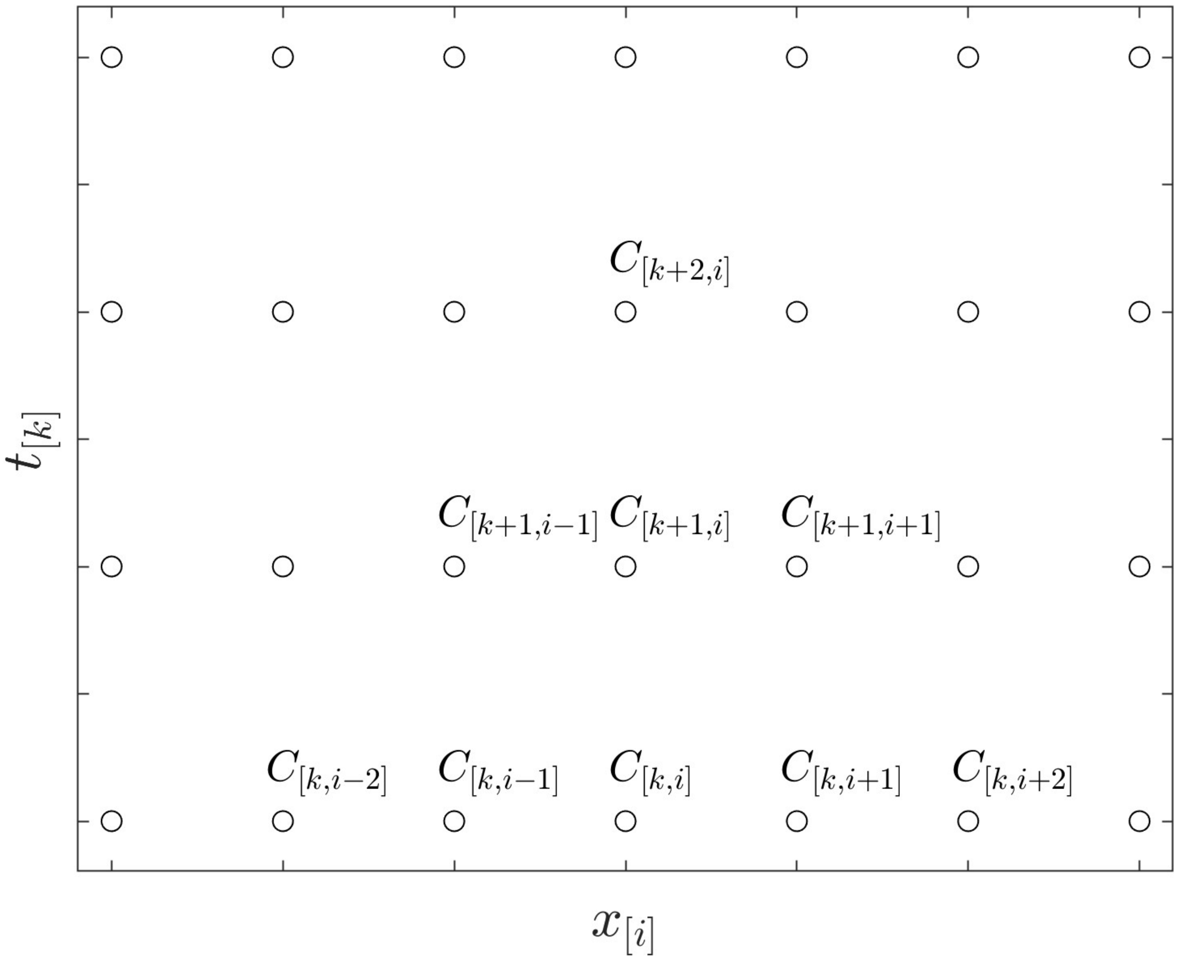
$$C_i^{new} = C_i^{old} + \frac{\Delta t D}{\Delta x^2}\left(C_{i+1}^{old} - 2C_i^{old} + C_{i-1}^{old}\right)$$

The finite difference approximation

We can sketch a grid that describes where our values are located in time and in space.

Note that for each value of C_k, we need three values of C_{k-1} (from the previous step). Thus, we need two more equations to complete our system.

The extra equations come from the boundary conditions.



The finite difference approximation

For the 1st point (Dirichlet BC)

$$C_1^k = \text{C1}$$

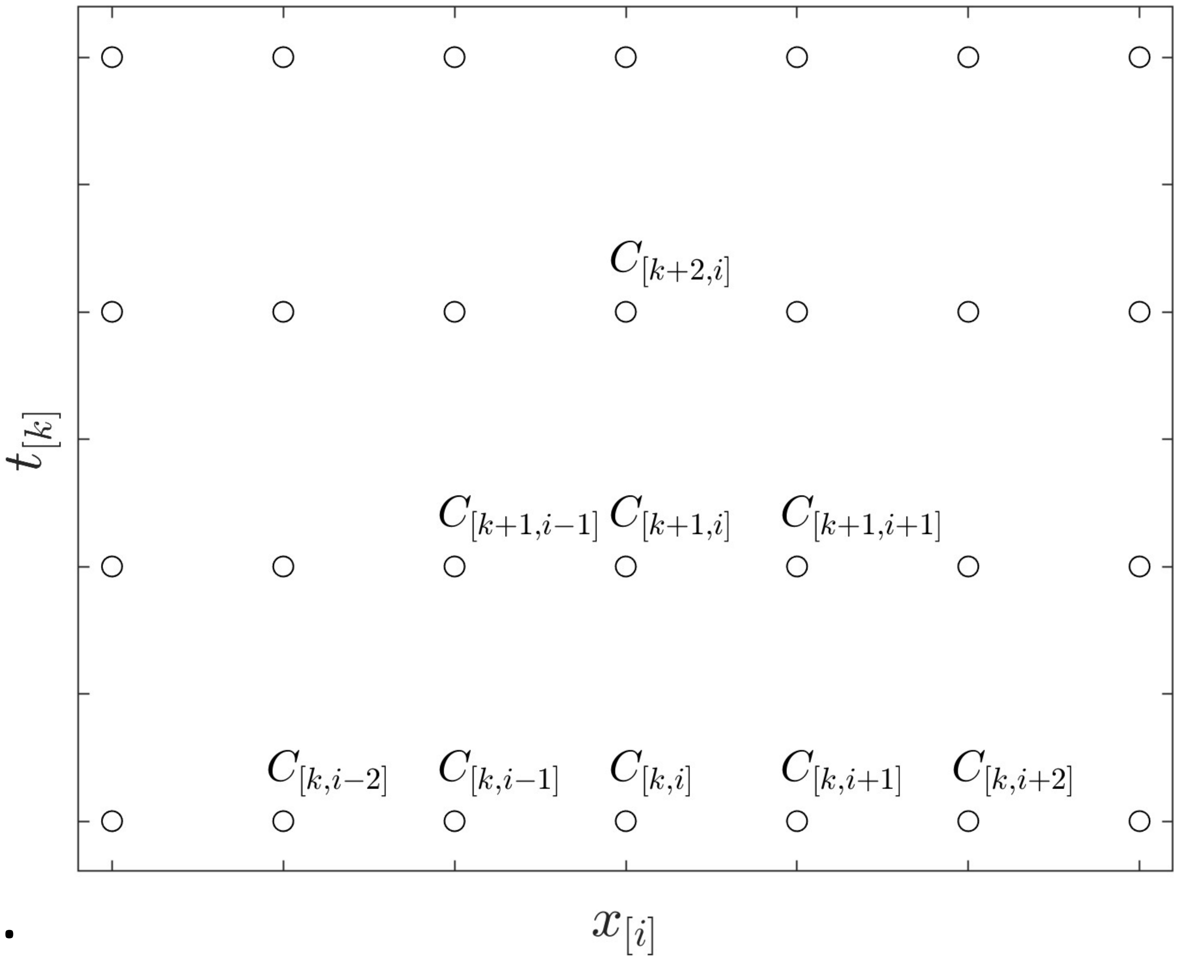
For the intermediate points

$$C_i^k = C_i^{k-1} + \frac{\Delta t D}{\Delta x^2}\left(C_{i+1}^{k-1} - 2C_i^{k-1} + C_{i-1}^{k-1}\right)$$

For the last point (Dirichlet BC)

$$C_{nx}^k = \text{C2}$$

where nx is the total number of points.



The finite difference approximation

For the 1st point (Dirichlet BC)

$$C_1^k = \text{C}1$$

For the intermediate points

$$C_i^k = C_i^{k-1} + \frac{\Delta t D}{\Delta x^2}\left(C_{i+1}^{k-1} - 2C_i^{k-1} + C_{i-1}^{k-1}\right)$$

For the last point (Dirichlet BC)

$$C_{nx}^k = \text{C}2$$

where nx is the total number of points.

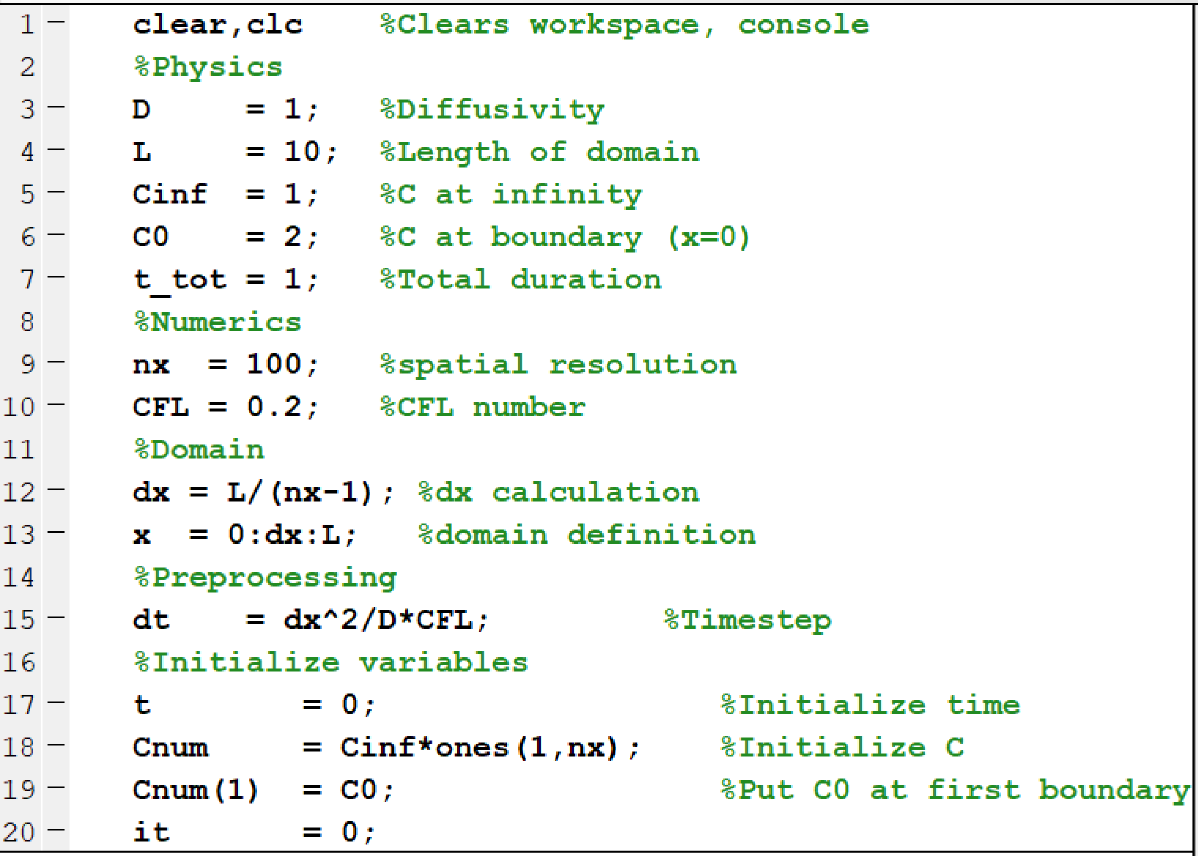
Exercise 6

Program the explicit diffusion method and try to reproduce the results from the half-space solution method

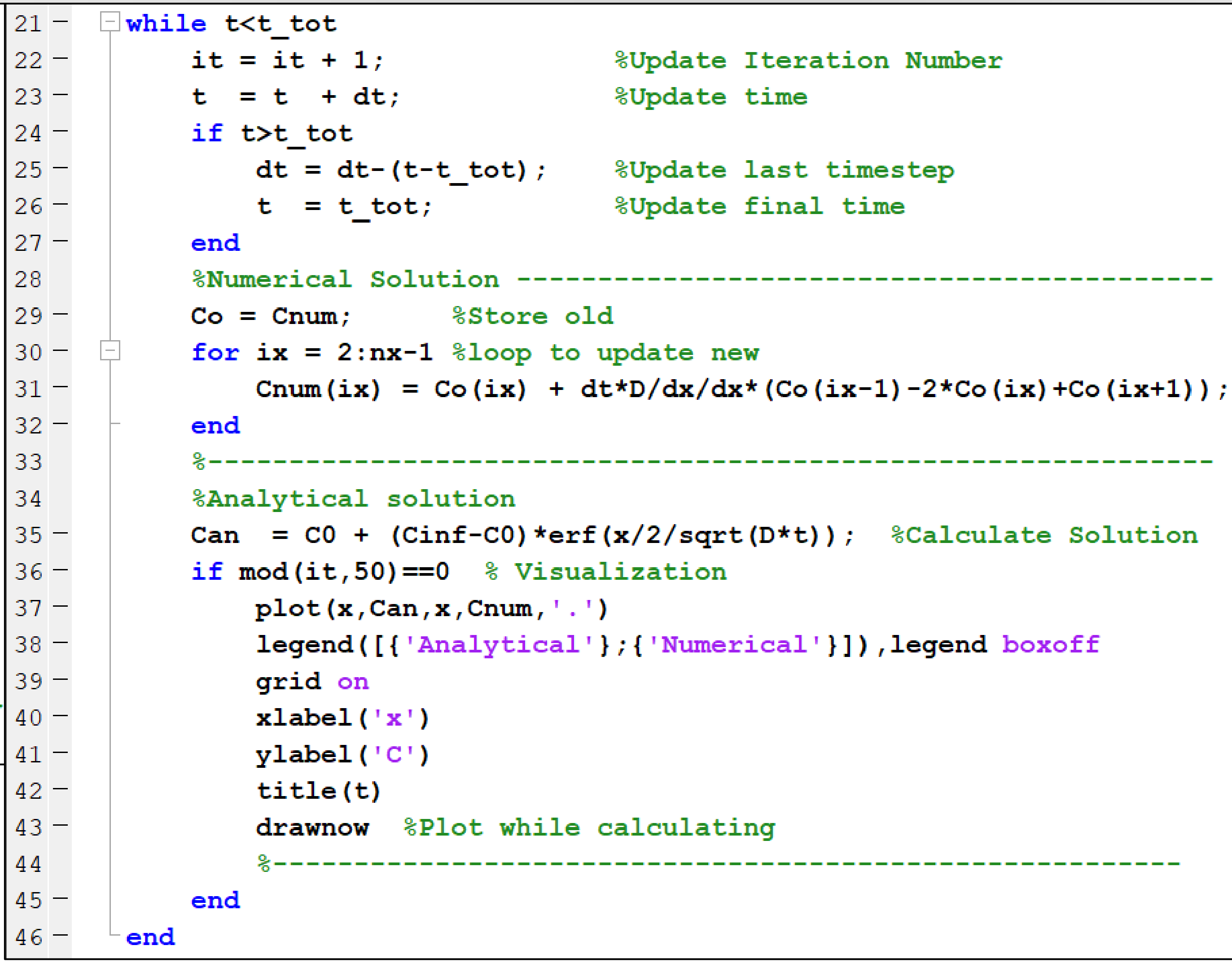
(use $dt = \frac{dx^2}{D} \cdot 0.2$)

(since the half-space solution assumes that the right boundary condition lies at infinity, we will move the boundary far away from the action area for now)

The finite difference approximation



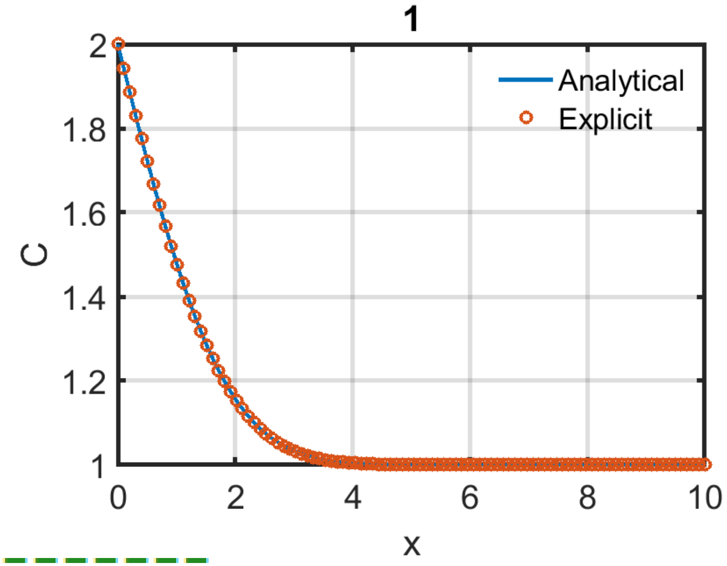
```matlab
clear,clc    %Clears workspace, console
%Physics
D     = 1;   %Diffusivity
L     = 10;  %Length of domain
Cinf  = 1;   %C at infinity
C0    = 2;   %C at boundary (x=0)
t_tot = 1;   %Total duration
%Numerics
nx  = 100;   %spatial resolution
CFL = 0.2;   %CFL number
%Domain
dx = L/(nx-1); %dx calculation
x  = 0:dx:L;   %domain definition
%Preprocessing
dt    = dx^2/D*CFL;         %Timestep
%Initialize variables
t        = 0;                   %Initialize time
Cnum     = Cinf*ones(1,nx);     %Initialize C
Cnum(1)  = C0;                  %Put C0 at first boundary
it       = 0;
```



```matlab
while t<t_tot
    it = it + 1;             %Update Iteration Number
    t  = t  + dt;            %Update time
    if t>t_tot
        dt = dt-(t-t_tot);   %Update last timestep
        t  = t_tot;          %Update final time
    end
    %Numerical Solution -----------------------------------------
    Co = Cnum;       %Store old
    for ix = 2:nx-1 %loop to update new
        Cnum(ix) = Co(ix) + dt*D/dx/dx*(Co(ix-1)-2*Co(ix)+Co(ix+1));
    end
    %-----------------------------------------------------------
    %Analytical solution
    Can   = C0 + (Cinf-C0)*erf(x/2/sqrt(D*t));  %Calculate Solution
    if mod(it,50)==0  % Visualization
        plot(x,Can,x,Cnum,'.')
        legend([{'Analytical'};{'Numerical'}]),legend boxoff
        grid on
        xlabel('x')
        ylabel('C')
        title(t)
        drawnow  %Plot while calculating
        %------------------------------------------------------
    end
end
```

The finite difference approximation

The most crucial part of the previous code is shown below. Note that we first store the old value before we calculate the derivatives. This is done to avoid mixing ‘new’ and ‘old’ values.



```
%Numerical Solution ------------------------------------------
Co = Cnum;       %Store old
for ix = 2:nx-1 %loop to update new
    Cnum(ix) = Co(ix) + dt*D/dx/dx*(Co(ix-1)-2*Co(ix)+Co(ix+1));
end
%------------------------------------------------------------
%Analytical solution
Can  = C0 + (Cinf-C0)*erf(x/2/sqrt(D*t));  %Calculate Solution
```

The implicit method

In the implicit method (allows larger timesteps), the PDE is discretized as follows

$$\frac{C_i^{new} - C_i^{old}}{\Delta t} = D\,\frac{1}{\Delta x}\left(\frac{(C_{i+1}^{new} - C_i^{new})}{\Delta x} - \frac{(C_i^{new} - C_{i-1}^{new})}{\Delta x}\right)$$

The previous can be re-arranged as:

$$C_i^{new} - D\,\frac{\Delta t}{\Delta x^2}(C_{i+1}^{new} - 2C_i^{new} + C_{i-1}^{new}) = C_i^{old}$$

with a bit of simplification

$$(1 + 2S)C_i^{new} - SC_{i+1}^{new} - SC_{i-1}^{new} = C_i^{old} \qquad S = D\Delta t\Delta x^{-2}$$

The implicit method

The previous can be written in a (**sparse**) matrix form as follows:

$$
\begin{bmatrix}
\cdots & & & & & \\
-S & 1+2S & -S & & & \\
 & -S & 1+2S & -S & & \\
 & & -S & 1+2S & -S & \\
 & & & -S & 1+2S & -S \\
 & & & & & \cdots
\end{bmatrix}
\begin{bmatrix} C_1^{new} \\ C_2^{new} \\ C_3^{new} \\ C_4^{new} \\ C_5^{new} \\ C_6^{new} \end{bmatrix}
=
\begin{bmatrix} C_1^{old} \\ C_2^{old} \\ C_3^{old} \\ C_4^{old} \\ C_5^{old} \\ C_6^{old} \end{bmatrix}
$$

$$(1+2S)C_i^{new} - SC_{i+1}^{new} - SC_{i-1}^{new} = C_i^{old} \qquad S = D\Delta t\Delta x^{-2}$$

The implicit method

$$\begin{array}{c} \text{BC} \\ \\ \\ \\ \\ \text{BC} \end{array} \underbrace{\begin{bmatrix} 1 & 0 & & & & \\ -S & 1+2S & -S & & & \\ & -S & 1+2S & -S & & \\ & & -S & 1+2S & -S & \\ & & & -S & 1+2S & -S \\ & & & & 0 & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} C_1^{new} \\ C_2^{new} \\ C_3^{new} \\ C_4^{new} \\ C_5^{new} \\ C_6^{new} \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} C_1^{old} \\ C_2^{old} \\ C_3^{old} \\ C_4^{old} \\ C_5^{old} \\ C_6^{old} \end{bmatrix}}_{b}$$

Do not forget that we need to fill the equations for the boundary conditions (BC) as well. For the case where the concentration does not change in time we need 1 in the diagonals and 0 everywhere else in the same line.

The implicit method

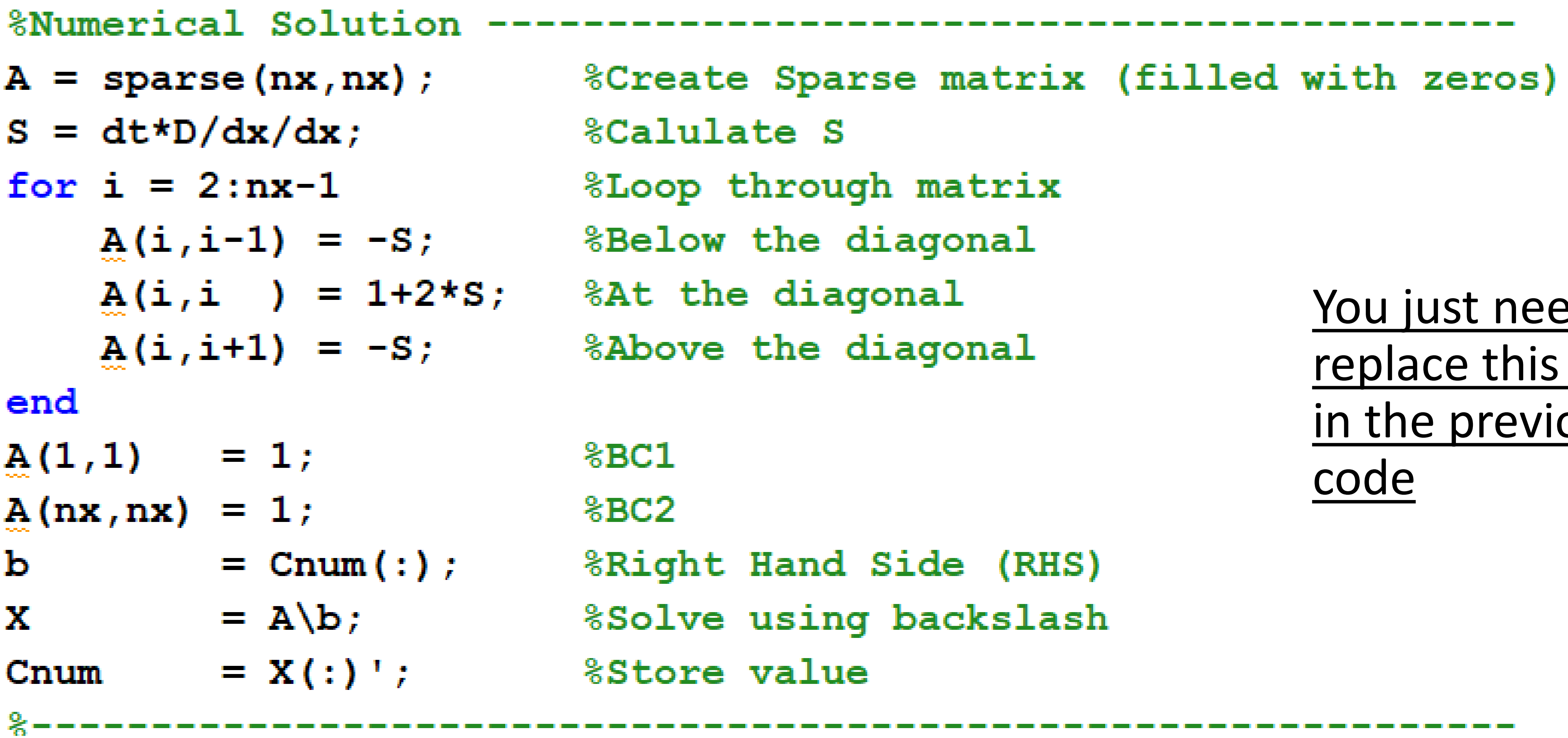
There are many ways to solve a system like $\mathbf{Ax} = \mathbf{b}$

1. $\mathrm{x} = \mathrm{A}^{-1}\mathrm{b}$ or in MATLAB language **x=inv(A)*b**
2. Actually, there is no need to compute the inverse first, i.e. **x=A\b**
3. … other iterative methods

Exercise 7

Program the implicit diffusion method to solve the half-space problem. You do not need to change the whole code, just the part where the numerical solution is calculated.

The finite difference approximation



```
%Numerical Solution -------------------------------------------
A = sparse(nx,nx);      %Create Sparse matrix (filled with zeros)
S = dt*D/dx/dx;         %Calulate S
for i = 2:nx-1          %Loop through matrix
    A(i,i-1) = -S;      %Below the diagonal
    A(i,i  ) = 1+2*S;   %At the diagonal
    A(i,i+1) = -S;      %Above the diagonal
end
A(1,1)   = 1;           %BC1
A(nx,nx) = 1;           %BC2
b        = Cnum(:);     %Right Hand Side (RHS)
X        = A\b;         %Solve using backslash
Cnum     = X(:)';       %Store value
%--------------------------------------------------------------
```

<u>You just need to replace this part in the previous code</u>

Geospeedometry and time-transformation techniques

Temperature dependence of diffusion

In the classic half-space solution we need to know the diffusivity of the material. If we do, we can solve the diffusion problem in order to estimate the diffusion timescale (also known as **Diffusion Chronometry**)

$$C(x,t) = C_{x=0} + (C_{inf} - C_{x=0})\mathrm{erf}(\frac{x}{2\sqrt{Dt}})$$

However, even in the simplest cases, D is a strong function of temperature.

$$D = D_0 \exp(-\frac{E}{RT})$$

where D_0 is a pre-exponential factor, E is the activation energy, R is the universal gas constant and T is the **absolute** temperature.

Temperature dependence of diffusion

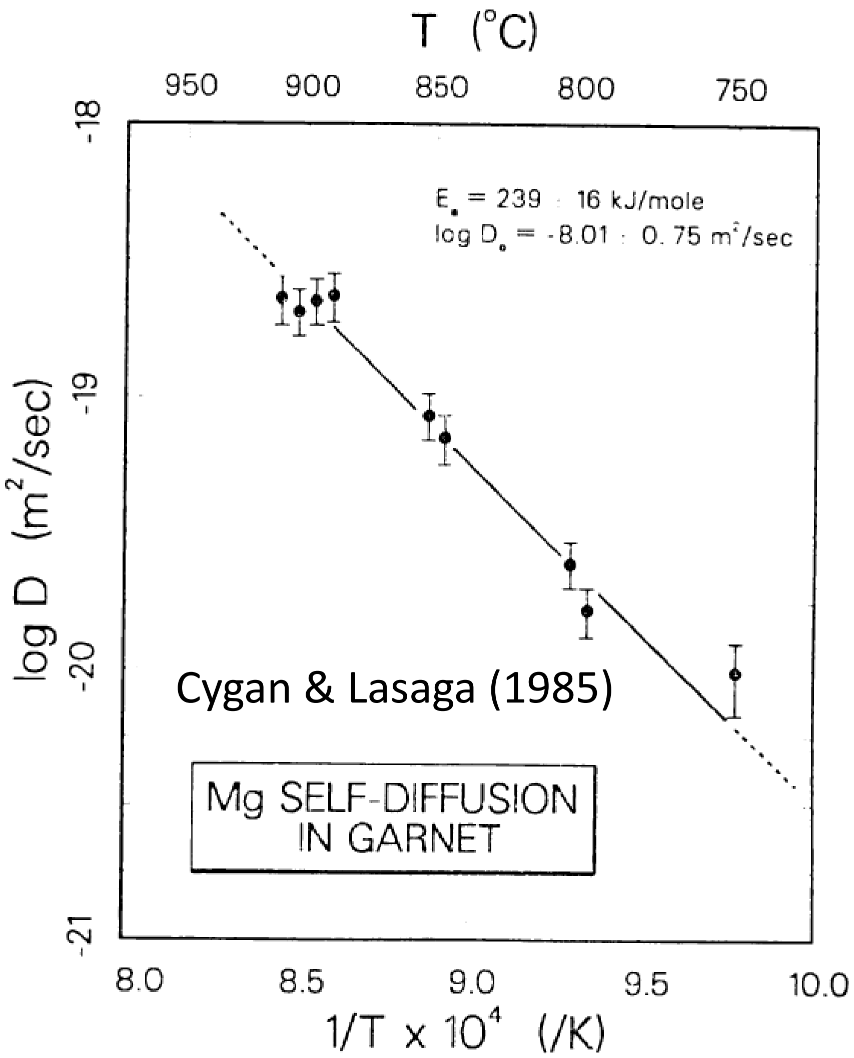
$$D = D_0 \exp(-\frac{E}{RT})$$

Arrhenius relationship

$$\ln(D) = \ln(D_0) - \frac{E}{RT}$$

$$\log(D) = \log(D_0) - \frac{E}{2.303RT}$$

Note: ln x = 2.303 log x



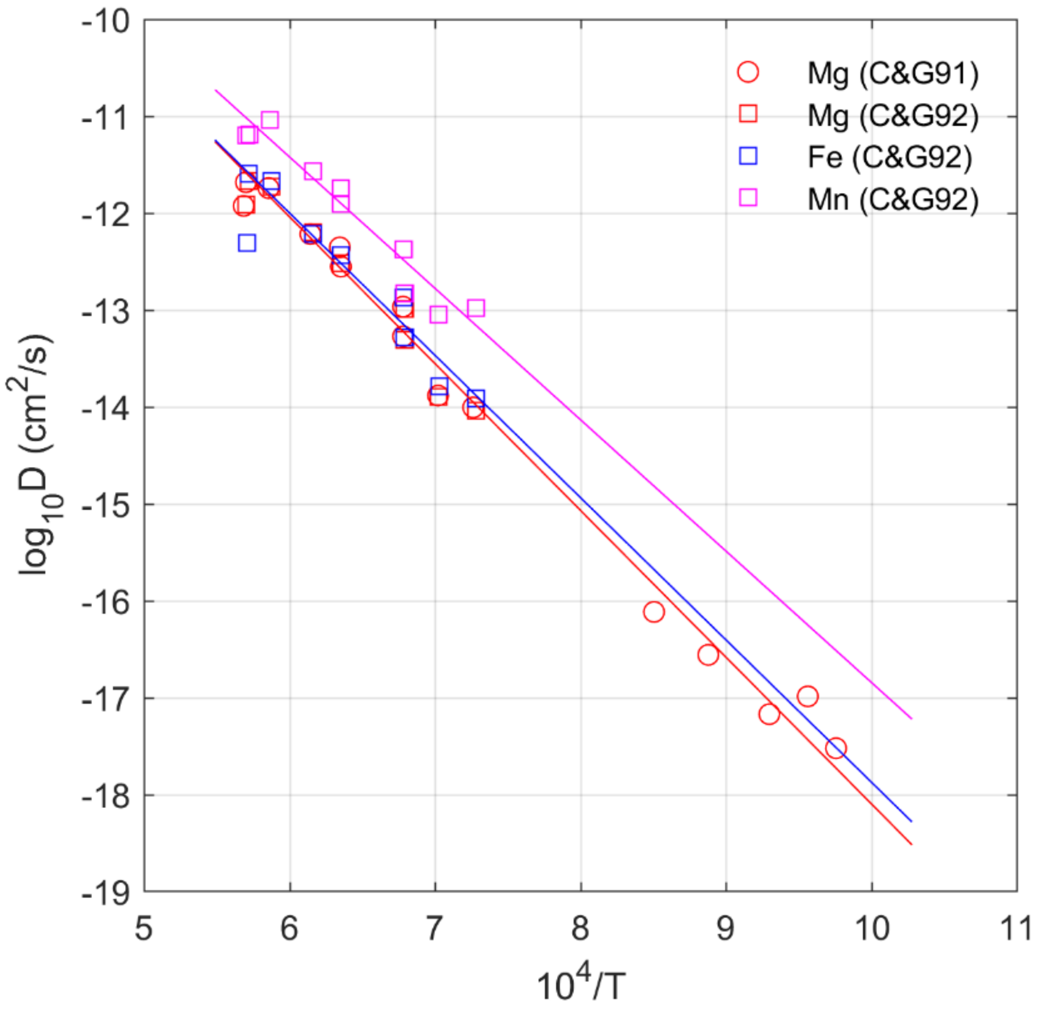
Cygan & Lasaga (1985)

Temperature dependence of diffusion

This strong temperature dependence can be seen for all minerals and elements (see also data compilation from Brady and Cherniak, 2010).

The figure on the right shows the temperature sensitivity of diffusion for major-elements in garnet (after Chakraborty and Ganguly, 1992).

The strong decrease of diffusivity at low temperatures is the reason why the profiles are preserved.



Paths with constant cooling rate

So, how could we use the solutions that we learned?

The first step is to do it numerically. I.e. we can assume a general function for cooling.

$$\frac{dT}{dt} = -s$$

where s is the cooling rate (in K or °C per sec). The previous can be solved explicitly as:

$$T = T_0 - st$$

Paths with constant cooling rate

Exercise 8

Program a numerical model that solves the half-space problem for the case of a cooling history. Use the implicit method (backward Euler) to update the temperature as a function of time.

At each step, time, temperature, diffusivity and timestep should be updated following the formulas given.

$$t = t + dt$$

$$T = T_0 - st$$

$$D = D_0 \exp(-\frac{E}{RT})$$

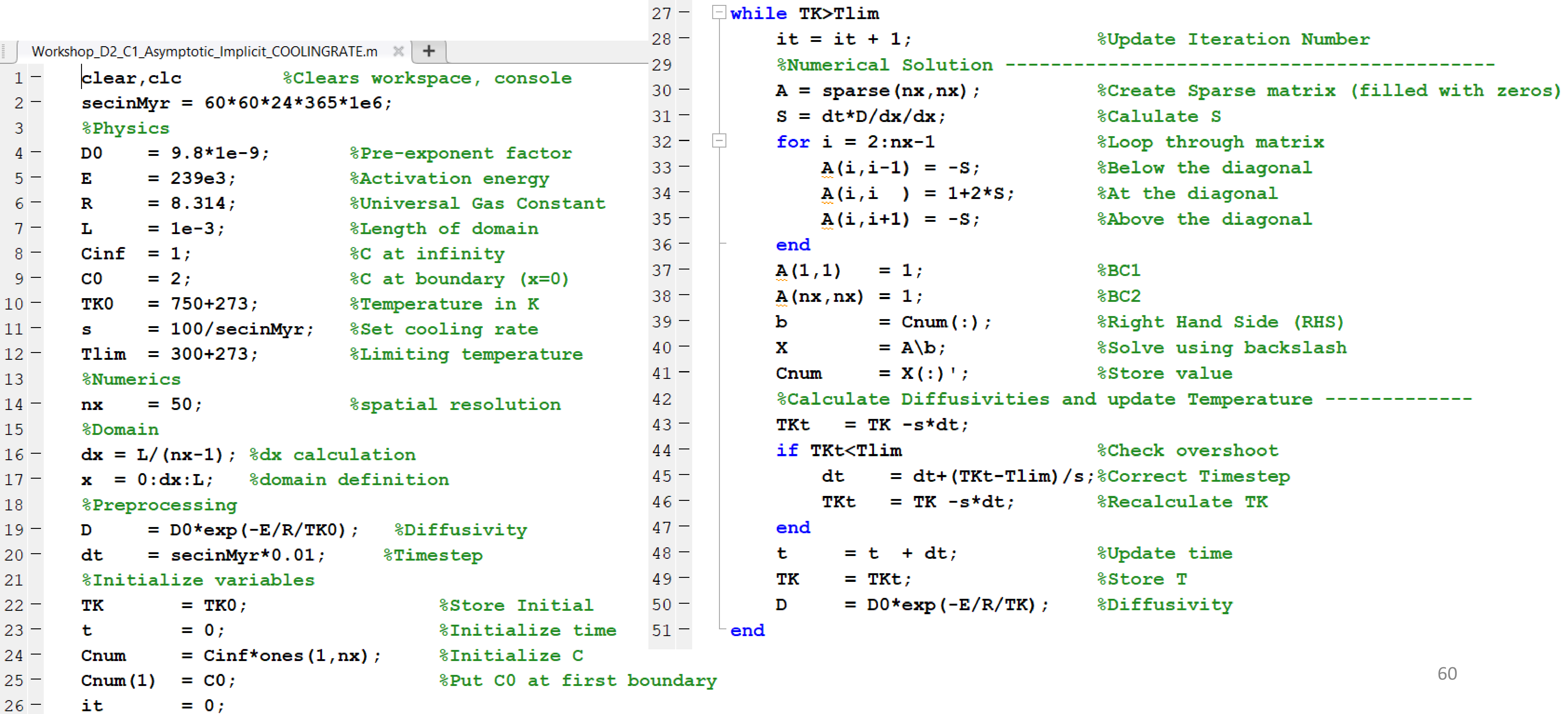
$$dt = 0.05\text{Myr}$$

use

$L = 1\text{mm}$, $D_0 = 9.8 \cdot 10^{-9}\text{m}^2/\text{s}$

$E = 239\text{kJ/mol}$, $s = 100\ °\text{C/Myr}$

Paths with constant cooling rate



```matlab
clear,clc          %Clears workspace, console
secinMyr = 60*60*24*365*1e6;
%Physics
D0     = 9.8*1e-9;        %Pre-exponent factor
E      = 239e3;           %Activation energy
R      = 8.314;           %Universal Gas Constant
L      = 1e-3;            %Length of domain
Cinf   = 1;               %C at infinity
C0     = 2;               %C at boundary (x=0)
TK0    = 750+273;         %Temperature in K
s      = 100/secinMyr;    %Set cooling rate
Tlim   = 300+273;         %Limiting temperature
%Numerics
nx     = 50;              %spatial resolution
%Domain
dx = L/(nx-1); %dx calculation
x  = 0:dx:L;   %domain definition
%Preprocessing
D      = D0*exp(-E/R/TK0);    %Diffusivity
dt     = secinMyr*0.01;      %Timestep
%Initialize variables
TK         = TK0;                %Store Initial
t          = 0;                  %Initialize time
Cnum       = Cinf*ones(1,nx);    %Initialize C
Cnum(1)    = C0;                 %Put C0 at first boundary
it         = 0;
while TK>Tlim
    it = it + 1;                        %Update Iteration Number
    %Numerical Solution ------------------------------------------
    A = sparse(nx,nx);                  %Create Sparse matrix (filled with zeros)
    S = dt*D/dx/dx;                     %Calulate S
    for i = 2:nx-1                      %Loop through matrix
        A(i,i-1) = -S;                  %Below the diagonal
        A(i,i  ) = 1+2*S;               %At the diagonal
        A(i,i+1) = -S;                  %Above the diagonal
    end
    A(1,1)    = 1;                      %BC1
    A(nx,nx)  = 1;                      %BC2
    b         = Cnum(:);                %Right Hand Side (RHS)
    X         = A\b;                    %Solve using backslash
    Cnum      = X(:)';                  %Store value
    %Calculate Diffusivities and update Temperature -------------
    TKt    = TK -s*dt;
    if TKt<Tlim                         %Check overshoot
        dt     = dt+(TKt-Tlim)/s;%Correct Timestep
        TKt    = TK -s*dt;              %Recalculate TK
    end
    t      = t  + dt;                   %Update time
    TK     = TKt;                       %Store T
    D      = D0*exp(-E/R/TK);           %Diffusivity
end
```

Paths with constant cooling rate

What we actually did, is that we solved the diffusion problem in small steps of isothermal diffusion. This is actually an integration.

Assume that the Temperature drops at a constant rate. Then, diffusivity ($\widetilde{D}$) drops very fast. If we normalize the diffusion coefficient using the maximum diffusion coefficient (D_M) we see the extent of diffusion as a function of t.

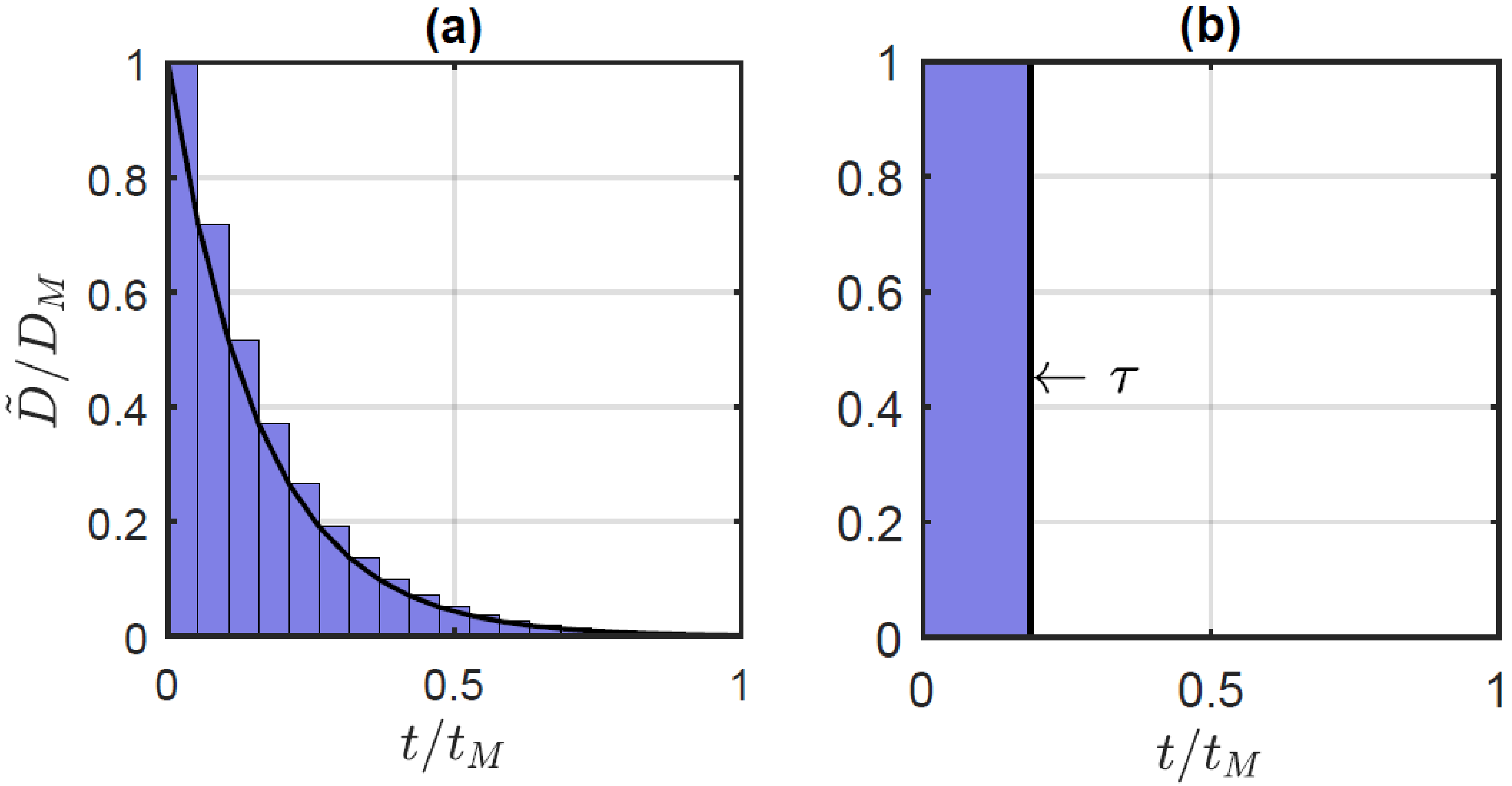


figure from Moulas et al. (in prep.)

Paths with constant cooling rate

Lets assume a cooling history with $T(t)$. Then the diffusion coefficient can be rewritten as:

$$\widetilde{D}(t) = D_0 \cdot exp\left(-\frac{E}{RT(t)}\right) = D_{\mathrm{MAX}} \cdot exp\left[-\frac{E}{R}\left(\frac{1}{T(t)} - \frac{1}{T_{\mathrm{MAX}}}\right)\right]$$

where D_{MAX} is the maximum diffusion coefficient at the starting temperature T_{MAX}

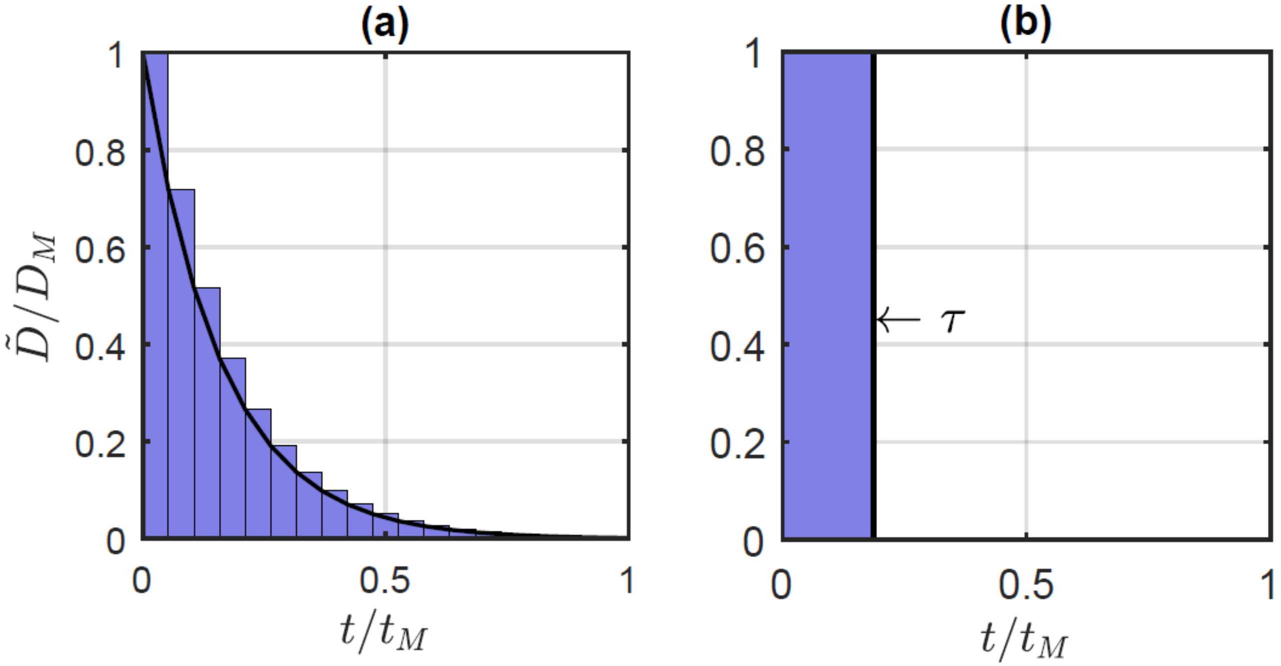


figure from Moulas et al. (in prep.)

Time-transformation techniques

In the case where diffusivity becomes a function of time, we can define the following 'time' variable τ (after Lasaga, 1983)

$$\tau(t) = \int_0^t \frac{\widetilde{D}(t')}{D_{MAX}} dt'$$

$$\widetilde{D} = \widetilde{D}\big(T(t)\big)$$

$$\tilde{C} = \tilde{C}(\tau(t), x)$$

This treatment converts the problem into an isothermal problem.

This results to

$$\frac{\partial \tilde{C}}{\partial t} = \widetilde{D}\frac{\partial^2 \tilde{C}}{\partial x^2} \leftrightarrow$$

$$\frac{\partial \tilde{C}}{\partial \tau}\frac{\partial \tau}{\partial t} = \frac{\partial \tilde{C}}{\partial \tau}\frac{\widetilde{D}}{D_{\mathrm{MAX}}} = \widetilde{D}\frac{\partial^2 \tilde{C}}{\partial x^2} \leftrightarrow$$

$$\frac{\partial \tilde{C}}{\partial \tau} = D_{\mathrm{MAX}}\frac{\partial^2 \tilde{C}}{\partial x^2}$$

Time-transformation techniques

This approach can be further simplified according to Crank (1956)

$$\frac{\partial \tilde{C}}{\partial t} = \widetilde{D}\frac{\partial^2 \tilde{C}}{\partial x^2} \leftrightarrow$$

$$\frac{\partial \tilde{C}}{\partial \zeta}\frac{\partial \zeta}{\partial t} = \frac{\partial \tilde{C}}{\partial \zeta}\widetilde{D} = \widetilde{D}\frac{\partial^2 \tilde{C}}{\partial x^2} \leftrightarrow$$

$$\frac{\partial \tilde{C}}{\partial \zeta} = \frac{\partial^2 \tilde{C}}{\partial x^2}$$

where

$$\zeta(t) = \int_0^t \widetilde{D}(t')dt'$$

$$\widetilde{D} = \widetilde{D}\big(T(t)\big)$$

$$\tilde{C} = \tilde{C}(\zeta(t), x)$$

This treatment converts the problem into a diffusion problem with an effective $D = 1$.

Time-transformation techniques

Then, we could use the following analytical form directly.

$$C(x,\zeta) = C_{x=0} + \left(C_{inf} - C_{x=0}\right)\mathrm{erf}\left(\frac{x}{2\sqrt{\zeta}}\right)$$

$$\zeta = \int_0^{t_{MAX}} \widetilde{D}(t)dt$$

The value of t_{MAX} does not matter, provided that $T(t_{MAX})$ is relatively low (and diffusion at $T(t_{MAX})$ is negligible). This is a very reasonable assumption if one considers the metamorphic/magmatic rocks that are analyzed.

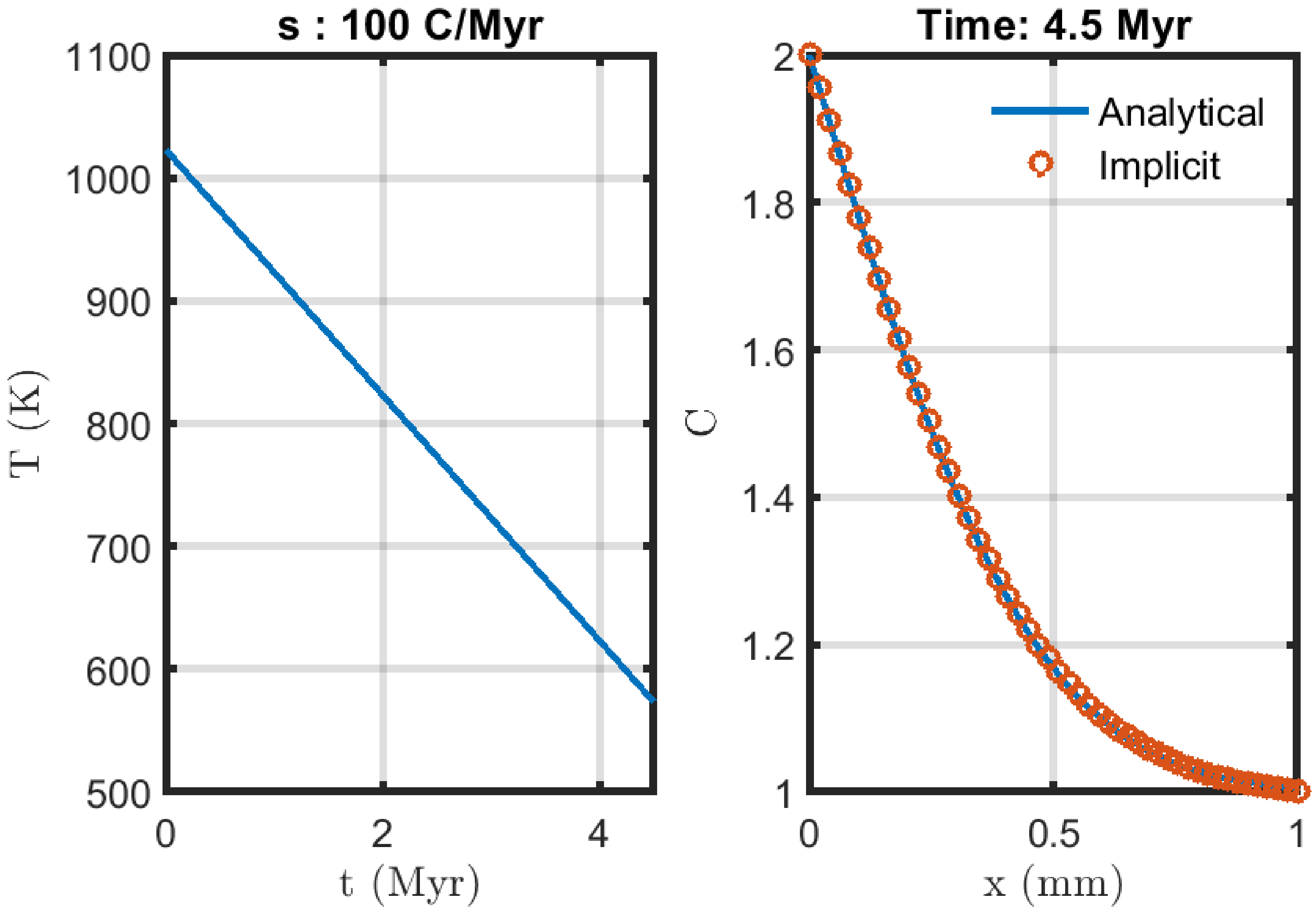
Time-transformation techniques

The results show the equivalence of the methods.

In this case, the integration of ζ was performed numerically.

In MATLAB, that is:

zeta = trapz(t,D)



Time-transformation techniques

Instead of doing the integration numerically, we can choose a temperature history in such a way that will help us with the calculations.

$$T(t) = \frac{T_{\mathrm{MAX}}}{1 + \frac{st}{T_{\mathrm{MAX}}}}$$

This is actually a quasi-linear path with an initial cooling rate equal to s

This path allows the analytical calculation of τ and ζ

$$\tau = \int_0^{t_{\mathrm{M}AX}} exp(-\gamma t)dt = \frac{1}{\gamma}[1 - exp(-\gamma t_{\mathrm{MAX}})] \qquad \gamma = \frac{sE}{RT_{MAX}^2}$$

$$\zeta = D_{MAX}\tau$$

Geospeedometry

Using the previous cooling path, the formula for τ can get a finite value (compressed time; after Lasaga, 1983)

$$\lim_{t_{MAX}\to\infty}(\tau) = \frac{1}{\gamma} = \frac{RT_{MAX}^2}{sE}$$

This means that **Diffusion Geochronometry** with $t = \tau$, can be used to obtain cooling rates (s) as in **Geospeedometry**[2]. THESE APPROACHES ARE EQUIVALENT.

That is:

$$s = \frac{RT_{MAX}^2}{E\tau} = \frac{RT_{MAX}^2 D_{MAX}}{E\zeta}$$

[2] The term " Geospeedometry " refers to the estimation of the "speed of cooling"

Geospeedometry

The equivalence of approaches shows that **the dilemma of Diffusion Chronometry versus Geospeedometry is false**. You can always have diffusion profiles that satisfy any random cooling path for a given value of ζ_{MAX}

The figure on the right shows temperature histories that would have identical diffusion solutions. Thus, the inversion to estimate time is non-unique.

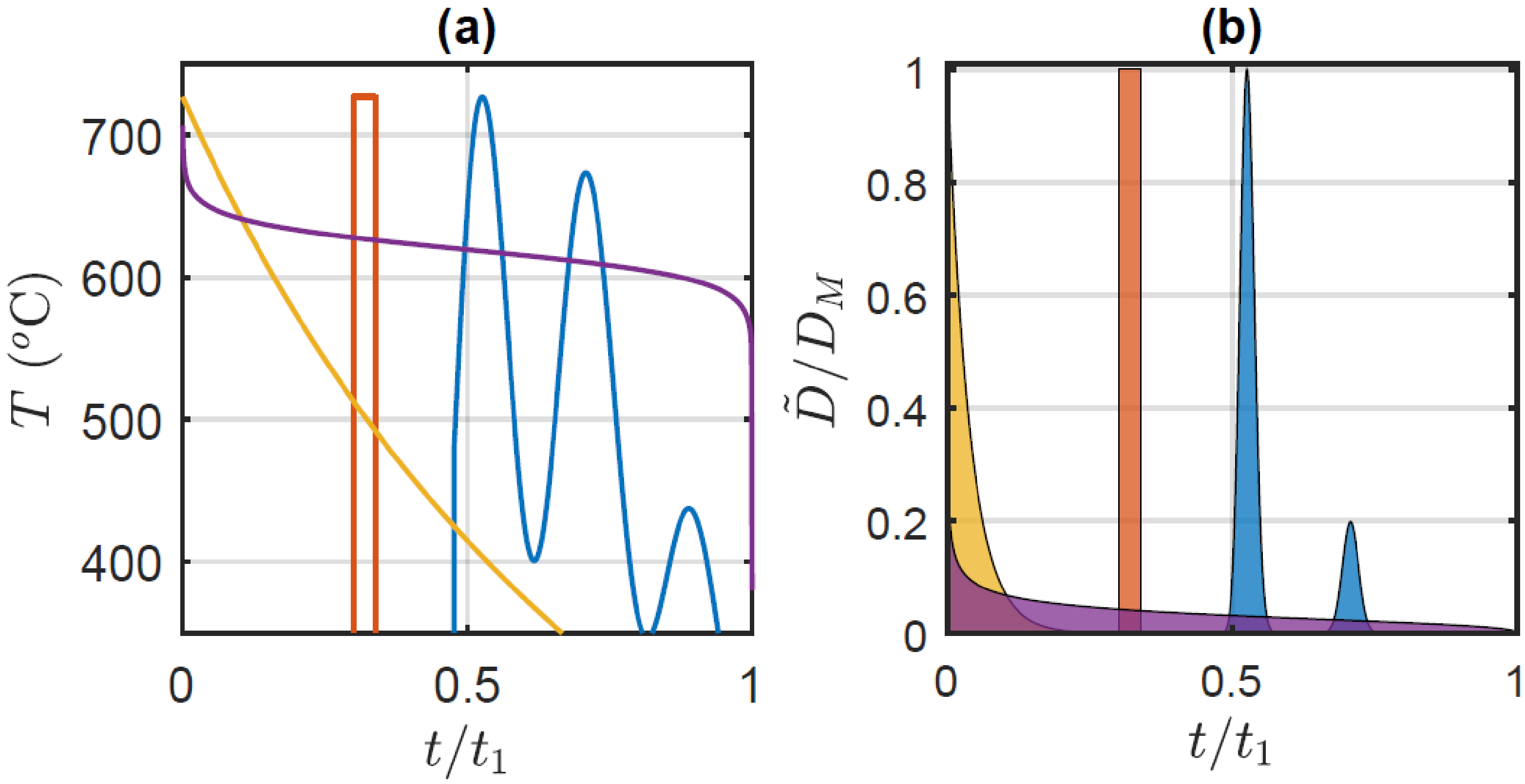
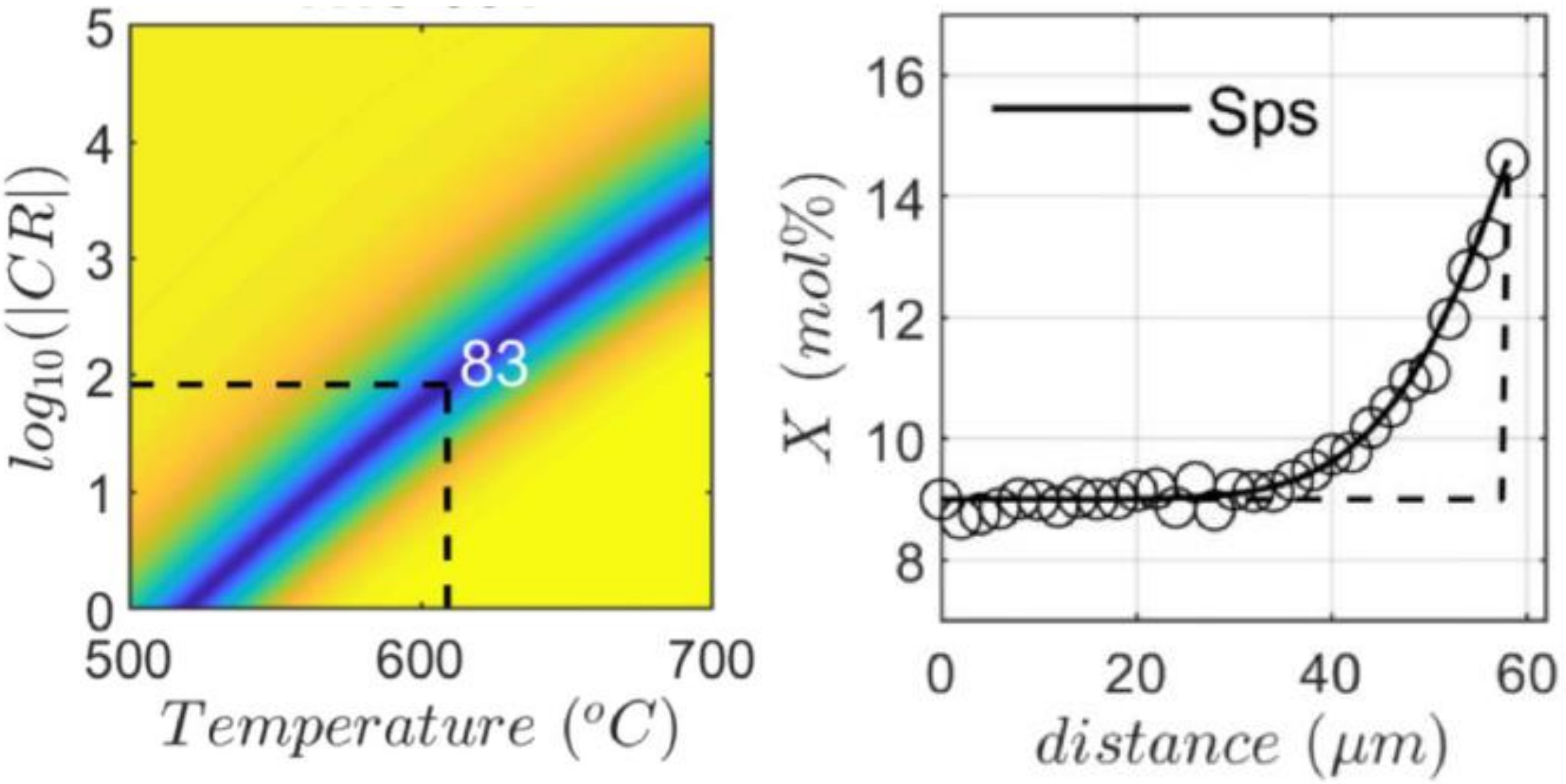


Figure from Moulas et al. (in prep.)

Geospeedometry

Note that even in simple cooling paths (e.g. with constant cooling rate) the results are non unique since the initial temperature may be different.



Cooling paths with different initial temperatures and cooling rates (left) result to the same diffusion profile (right).

figure after Burg & Moulas (2022)

Geospeedometry (Xtra)

The concept of Geospeedometry is not limited to diffusion coefficients that are constant. In fact, we can consider a diffusion coefficient which is also a function of concentration. That is:

$$D = F(T)G(C)$$

where $F(T)$ is a general function of T (as the normal diffusion coefficient) and $G(C)$ a function that depends on C. Then, the diffusion equation can be written as (assuming T is constant in space):

$$\frac{\partial C}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial C}{\partial x}\right) \leftrightarrow \frac{\partial C}{\partial t} = F(T)\frac{\partial}{\partial x}\left(G(C)\frac{\partial C}{\partial x}\right)$$

Geospeedometry (Xtra)

We can follow the same steps as before and assume that temperature changes with time $T(t)$. This leads to:

$$\frac{\partial \tilde{C}}{\partial t} = \tilde{F}\frac{\partial}{\partial x}\left(\tilde{G}(\tilde{C})\frac{\partial \tilde{C}}{\partial x}\right) \leftrightarrow$$

$$\frac{\partial \tilde{C}}{\partial \zeta}\frac{\partial \zeta}{\partial t} = \frac{\partial \tilde{C}}{\partial \zeta}\tilde{F} = \tilde{F}\frac{\partial}{\partial x}\left(\tilde{G}(\tilde{C})\frac{\partial \tilde{C}}{\partial x}\right) \leftrightarrow$$

$$\frac{\partial \tilde{C}}{\partial \zeta} = \frac{\partial}{\partial x}\left(\tilde{G}(\tilde{C})\frac{\partial \tilde{C}}{\partial x}\right)$$

where

$$\zeta(t) = \int_0^t \tilde{F}(t')dt'$$

$$\tilde{F} = \tilde{F}\big(T(t)\big)$$

$$\tilde{G} = \tilde{G}\big(\tilde{C}\big)$$

$$\tilde{C} = \tilde{C}(\zeta(t), x)$$

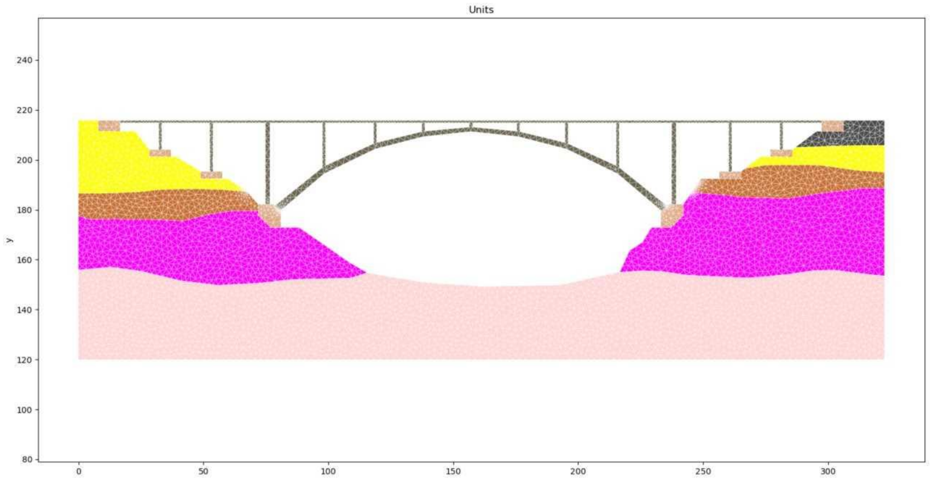
see also Schwinger et al., (2016)

Introduction to the Finite-Element-Method in 1 dimension

Finite Element Method

The finite element method (FEM) is another numerical method that can be used to solve the diffusion problem. It initially appears more complex compared to finite difference, however, it can be extended in higher dimensions in a straightforward manner.

The finite element method relies on the fact that we can ‘break’ and approximate our solution by many, low-degree, polynomials. The solution is thus ‘broken’ in many little elements (i.e. lines/areas/volumes in space).

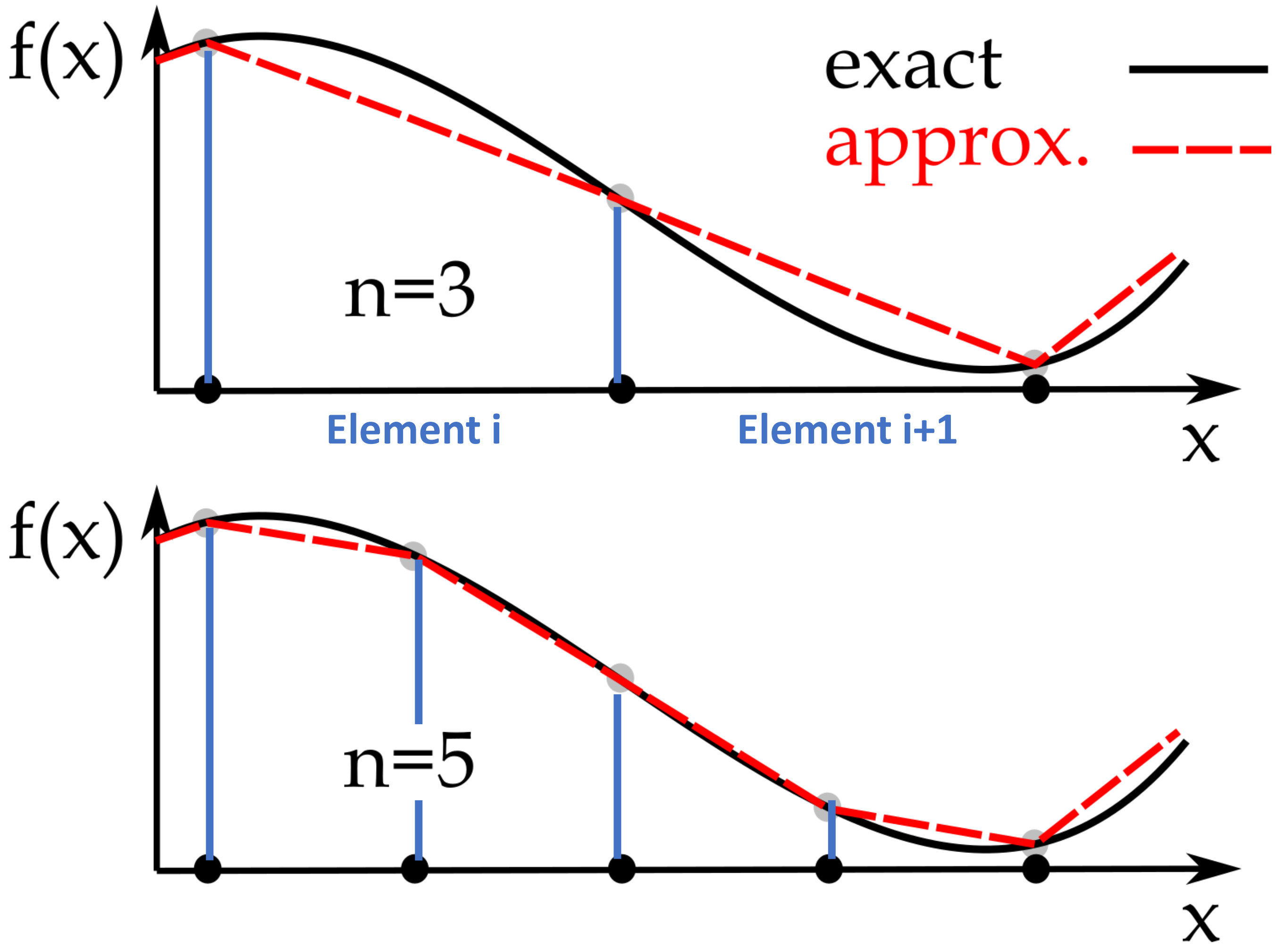


FE Model of a bridge
(image courtesy of A. Stroh)

Finite Element Method

With increasing number of elements, our numerical approximation becomes more and more accurate.

In the example shown on the right, we assume a linear approximation within each element. This means that each element will have two nodes (bold points).



Finite Element Method

We will start with a problem where its solution does not change with time. That is, an equation of the following form:

$$D\frac{d^2u}{dx^2} + F = 0$$

We can now assume that the solution in each element is a polynomial of the following form:

$$u \approx \bar{u} = \sum_{i=1}^{2} N_i(x)c_i = N_1(x)c_1 + N_2(x)c_2$$

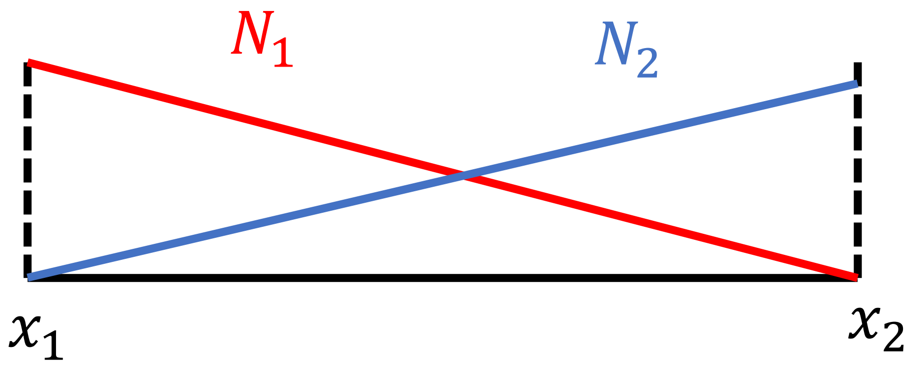
where $N_i(x)$ are polynomials of order 1 (also called shape functions) and c_i are the respective coefficients.

Finite Element Method

We will now consider the following polynomials

$$\begin{cases} N_1(x) = 1 - \dfrac{x - x_i}{x_{i+1} - x_i} \\ N_2(x) = \dfrac{x - x_i}{x_{i+1} - x_i} \end{cases}$$

where x is the spatial coordinate and $x_{i+1} - x_i$ is the length of the particular element. It becomes clear that N_1 becomes 1 and N_2 becomes 0 when $x = x_i$. In contrast, N_2 becomes 1 and N_1 becomes 0 when $x = x_{i+1}$. x_i and x_{i+1} are the edges of the element (also called nodes).



Finite Element Method

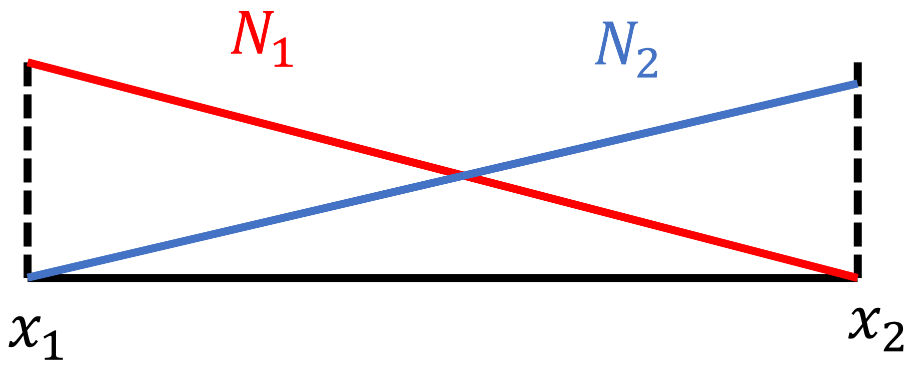
Note that $\sum_{i=1}^{2} N_i(x) = 1$

$$N_1(x) = 1 - \frac{x - x_i}{x_{i+1} - x_i}$$

$$N_2(x) = \frac{x - x_i}{x_{i+1} - x_i}$$

Also, note that if we take the points $x = x_1$ and $x = x_2$, the fact that the respective polynomials are 1 and zero, makes c_1 and c_2 equal to the solution of our function at these points. Therefore, it is common to write the numerical approximation as:

$$u \approx \bar{u} = \sum_{i=1}^{2} N_i(x) u_i = \begin{bmatrix} N_1 & N_2 \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix}$$



Finite Element Method

To finally solve our differential equation, we will first need to consider the problem in 1 element. Lets consider the problem at $x \in [x_1, x_2]$. The discretized equation has a residual R, that is given by:

$$R = D\frac{d^2\bar{u}}{dx^2} + F \qquad (1)$$

The residual must be as close to zero as possible when solving (1). To do that, we will use the concept of **Galerkin projection**. In brief, Boris Galerkin showed that to minimize the residual R, (1) must be 'orthogonal' to any 'test' function that lies in the same space as $\bar{u}$. Note that $\bar{u}$ lies in the space defined by N_1 and N_2 since $\bar{u} = \sum_{i=1}^{2} N_i(x)c_i$. In the Galerkin formalism, the test functions are the same as the functions used to discretize the solution (shape functions).

Finite Element Method

The concepts of ‘orthogonality’ and ‘space’ mentioned earlier are a bit more abstract than the equivalent terms in Euclidean geometry. Without going into details, we can already highlight some analogies from linear algebra and functional analysis that can be used. First, we have to define what is ‘linear’. In the more general sense, a function or an operator f is linear when the two following relations hold.

$$f(x_1 + x_2) = f(x_1) + f(x_2) \quad \text{and} \quad f(kx_1) = kf(x_1) \qquad (k\text{: scalar})$$

This is not to be confused with polynomials of degree 1 or equations of a line. For example, by setting $\bar{u} = \sum_{i=1}^{n} N_i(x)c_i$ allows the ‘decomposition’ of $\bar{u}$ by using simpler polynomials (not necessarily of order 1). This is the same principle as decomposing a vector into simpler basis vectors. Thus, by ‘space’ we refer to all the things that the combinations of polynomials can describe.

Finite Element Method

In a similar manner, the concept of ‘orthogonality’ is just a generalization from Euclidian geometry. In linear algebra, two vectors (of n dimensions) are ‘orthogonal’ (inner product = zero) if:

$$\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \sum_{i=1}^{n} a_i b_i = 0$$

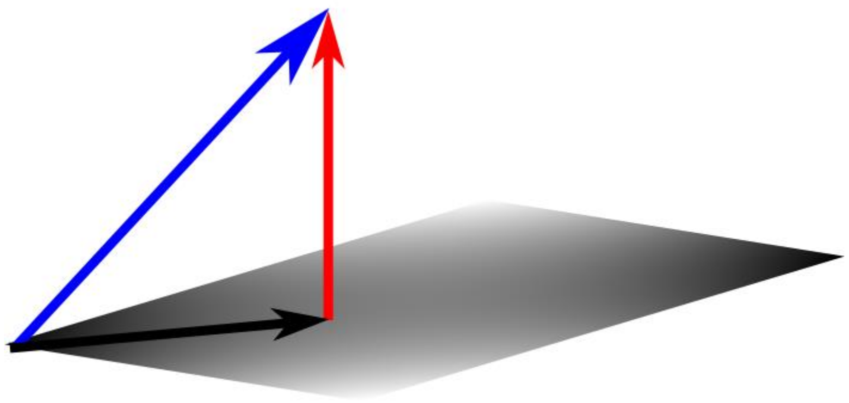
Following the same logic, two functions a, b are ‘orthogonal’ over a domain $x \in [x_1, x_2]$ if:

$$\langle a(x), b(x) \rangle = \int_{x_1}^{x_2} a(x) b(x) dx = 0$$

Finite Element Method

Coming back to our original problem, we realize that multiplying our discretized residuals by a shape function and integrating over a finite domain is like taking the inner product of a basis vector with our numerical error. The result should be zero because our approximate solution should be as close to the actual solution.

An analogy with vectors would be that our actual solution (blue vector) would be different from our numerical approximation (black vector). Our approximation lies in the space defined by our basis vectors (plane). To minimize the error (red vector) we must find a numerical approximation that is orthogonal to the error.



Finite Element Method

In practice, this means that if we multiply (1) by N_1 or N_2, and integrate the result, our residual should be zero. This is because our residual should be orthogonal to the space defined by N_1 and N_2. This means that:

$$R = D\frac{d^2\bar{u}}{dx^2} + F \tag{1}$$

and thus

$$\int_{x_1}^{x_2} N_1(x) D\frac{d^2\bar{u}}{dx^2} dx + \int_{x_1}^{x_2} N_1(x) F dx = 0 \tag{2}$$

Finite Element Method

We can now expand the spatial derivative in (2) as follows:

$$\frac{d^2\bar{u}}{dx^2} = \sum_{i=1}^{2} \frac{d^2 N_i(x)}{dx^2} u_i = \frac{d^2 N_1(x)}{dx^2} u_1 + \frac{d^2 N_2(x)}{dx^2} u_2 \qquad (3)$$

Thus, the first integral in (2), can be written as the sum of 2 integrals:

$$\int_{x_1}^{x_2} N_1(x) D \left[\frac{d^2 N_1(x)}{dx^2} u_1 + \frac{d^2 N_2(x)}{dx^2} u_2 \right] dx =$$

$$\int_{x_1}^{x_2} N_1(x) D \frac{d^2 N_1(x)}{dx^2} u_1 dx + \int_{x_1}^{x_2} N_1(x) D \frac{d^2 N_2(x)}{dx^2} u_2 dx$$

Finite Element Method

We can now use integration by parts in each term to obtain:

$$\int_{x_1}^{x_2} N_1(x) D \frac{d^2 N_1(x)}{dx^2} u_1 dx + \int_{x_1}^{x_2} N_1(x) D \frac{d^2 N_2(x)}{dx^2} u_2 dx =$$

$$-\int_{x_1}^{x_2} \frac{dN_1(x)}{dx} D \frac{dN_1(x)}{dx} u_1 dx - \int_{x_1}^{x_2} \frac{dN_1(x)}{dx} D \frac{dN_2(x)}{dx} u_2 dx + \left[N_1(x) D \frac{dN_1(x)}{dx} u_1 \right]_{x_1}^{x_2} + \left[N_1(x) D \frac{dN_2(x)}{dx} u_2 \right]_{x_1}^{x_2} =$$

$$-D \int_{x_1}^{x_2} \frac{dN_1(x)}{dx} \frac{dN_1(x)}{dx} u_1 dx - D \int_{x_1}^{x_2} \frac{dN_1(x)}{dx} \frac{dN_2(x)}{dx} u_2 dx + D \left[N_1(x) \left(\frac{dN_1(x)}{dx} u_1 + \frac{dN_2(x)}{dx} u_2 \right) \right]_{x_1}^{x_2} =$$

$$-D \int_{x_1}^{x_2} \frac{dN_1(x)}{dx} \frac{dN_1(x)}{dx} u_1 dx - D \int_{x_1}^{x_2} \frac{dN_1(x)}{dx} \frac{dN_2(x)}{dx} u_2 dx + D \left[N_1(x) \frac{d\bar{u}}{dx} \right]_{x_1}^{x_2}$$

Note that we have assumed that D is constant within the element. We repeat the same procedure for $N_2(x)$. I.e. multiply our residual equation by $N_2(x)$ and integrate…

Finite Element Method

By collecting both equations (the one that was multiplied by N_1 and the one that was multiplied by N_2) we obtain:

$$-D\int_{x_1}^{x_2}\begin{bmatrix}\frac{dN_1(x)}{dx}\frac{dN_1(x)}{dx} & \frac{dN_1(x)}{dx}\frac{dN_2(x)}{dx}\\ \frac{dN_2(x)}{dx}\frac{dN_1(x)}{dx} & \frac{dN_2(x)}{dx}\frac{dN_2(x)}{dx}\end{bmatrix}dx\begin{Bmatrix}u_1\\u_2\end{Bmatrix}$$

$$+\int_{x_1}^{x_2}\begin{Bmatrix}N_1(x)\\N_2(x)\end{Bmatrix}F\,dx+D\left[\begin{Bmatrix}N_1(x)\\N_2(x)\end{Bmatrix}\frac{d\bar{u}}{dx}\right]_{x_1}^{x_2}=\begin{Bmatrix}0\\0\end{Bmatrix}$$

These are two equations for 2 unknowns.

Finite Element Method

We can now repeat the same procedure for the second element. We then get

$$-D\int_{x_2}^{x_3}\begin{bmatrix}\frac{dN_1(x)}{dx}\frac{dN_1(x)}{dx} & \frac{dN_1(x)}{dx}\frac{dN_2(x)}{dx}\\ \frac{dN_2(x)}{dx}\frac{dN_1(x)}{dx} & \frac{dN_2(x)}{dx}\frac{dN_2(x)}{dx}\end{bmatrix}dx\begin{Bmatrix}u_2\\ u_3\end{Bmatrix}$$

$$+\int_{x_2}^{x_3}\begin{Bmatrix}N_1(x)\\ N_2(x)\end{Bmatrix}F\,dx+D\left[\begin{Bmatrix}N_1(x)\\ N_2(x)\end{Bmatrix}\quad\frac{d\bar{u}}{dx}\right]_{x_2}^{x_3}=\begin{Bmatrix}0\\ 0\end{Bmatrix}$$

These are also two equations for 2 unknowns. The same procedure is repeated for all the elements.

Finite Element Method

If we ignore the boundary terms (last term in previous equation), we can see that for every element we have a (local) system of equations of the form:

$$\mathbf{K}\{u_i\} = \mathbf{F}$$

where $\mathbf{K}$ and $\mathbf{F}$ are given by:

$$\mathbf{K} = -\mathrm{D}\int_{x_i}^{x_{i+1}} \begin{bmatrix} \frac{dN_1(x)}{dx}\frac{dN_1(x)}{dx} & \frac{dN_1(x)}{dx}\frac{dN_2(x)}{dx} \\ \frac{dN_2(x)}{dx}\frac{dN_1(x)}{dx} & \frac{dN_2(x)}{dx}\frac{dN_2(x)}{dx} \end{bmatrix} dx$$

$$\mathbf{F} = -\int_{x_i}^{x_{i+1}} \begin{Bmatrix} N_1(x) \\ N_2(x) \end{Bmatrix} F\,dx$$

Finite Element Method

To calculate the previous matrices, we need to calculate the derivatives and the integrals of the shape functions first. These are:

$$\int_{x_1}^{x_2} N_1(x)\,dx = \frac{\Delta x}{2}$$

$$\int_{x_1}^{x_2} N_2(x)\,dx = \frac{\Delta x}{2}$$

$$\frac{dN_1(x)}{dx} = -\frac{1}{\Delta x}$$

$$\frac{dN_2(x)}{dx} = \frac{1}{\Delta x}$$

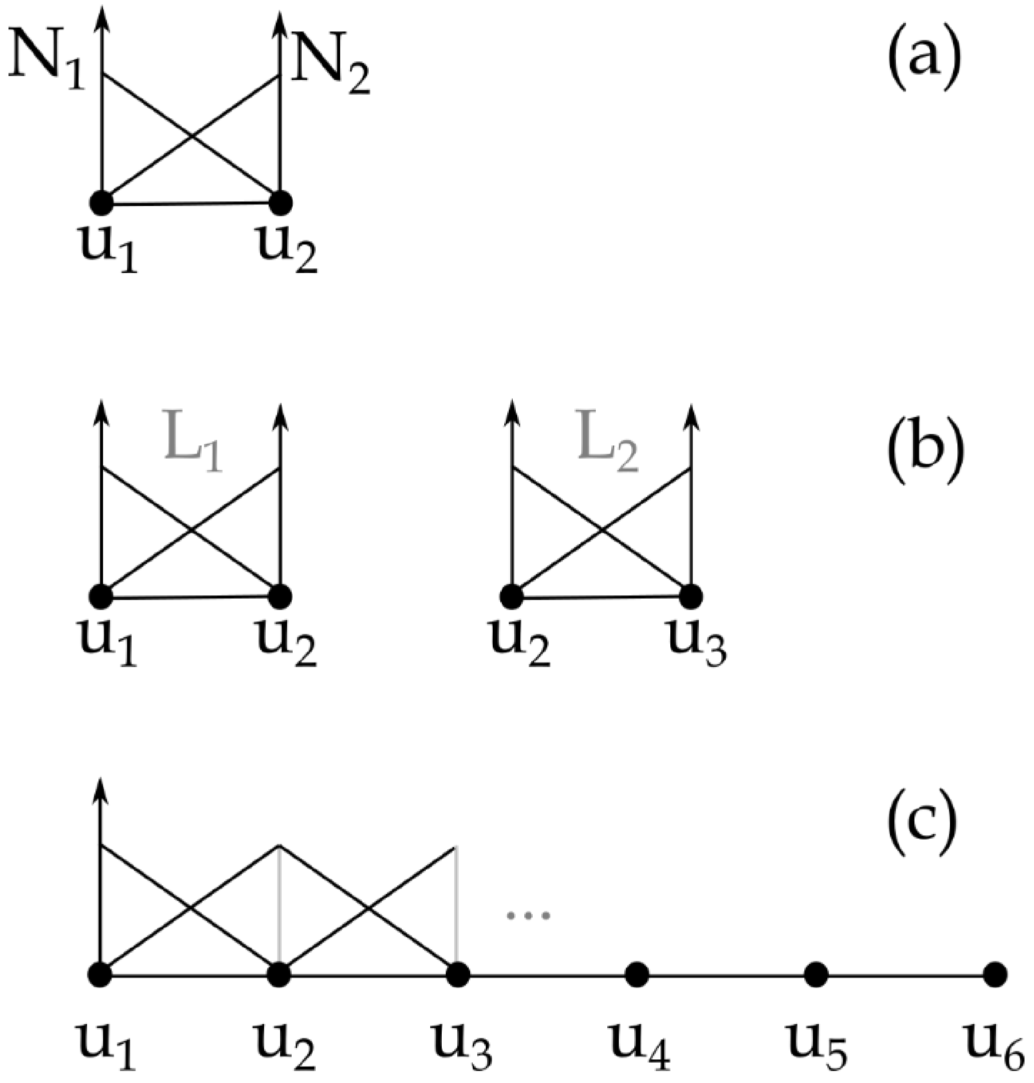
Finite Element Method

Using the previous solutions (assuming constant D and F within each element), the local element matrices become

$$\mathbf{K} = -D\begin{bmatrix} \frac{1}{\Delta x} & -\frac{1}{\Delta x} \\ -\frac{1}{\Delta x} & \frac{1}{\Delta x} \end{bmatrix} \qquad \mathbf{F} = -F\begin{bmatrix} \frac{\Delta x}{2} \\ \frac{\Delta x}{2} \end{bmatrix}$$

Finite Element Method

The principle in FEM is to combine all elements by putting all the local systems (L_1, L_2, etc) together. Thus, the local systems of equations can be added together to form a global system of equations. Due to the summation, the boundary terms vanish everywhere apart from the first and the last element. However, for cases with Dirichlet boundary conditions, the boundary terms can be neglected.



Finite Element Method

The combination of the local systems can be done by a simple summation of the terms in the global matrix. To see how this is done, we will consider two simple (2x2) systems that represent the local system of equations in two consecutive elements.

$$\begin{bmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix}$$

In this case, our unknowns are C_1 and C_2 (Equations 1, 2)

$$\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \begin{bmatrix} C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \end{bmatrix}$$

In this case, our unknowns are C_2 and C_3 (Equations 3, 4)

Note that C is continuous, i.e. it should have a single value (C_2) in both systems.

Finite Element Method

We can now add the second equation of the first system to the first equation of the second system, and obtain (after expansion):

$$\alpha_{21}\, C_1 + \alpha_{22} C_2 + b_{11} C_2 + b_{12} C_3 = g_2 + h_1$$

Grouping terms leads to:

$$\alpha_{21} C_1 + (\alpha_{22} + b_{11}) C_2 + b_{12} C_3 = g_2 + h_1$$

Thus, equations (1), (2+3) and (4) of the two previous systems can be written as:

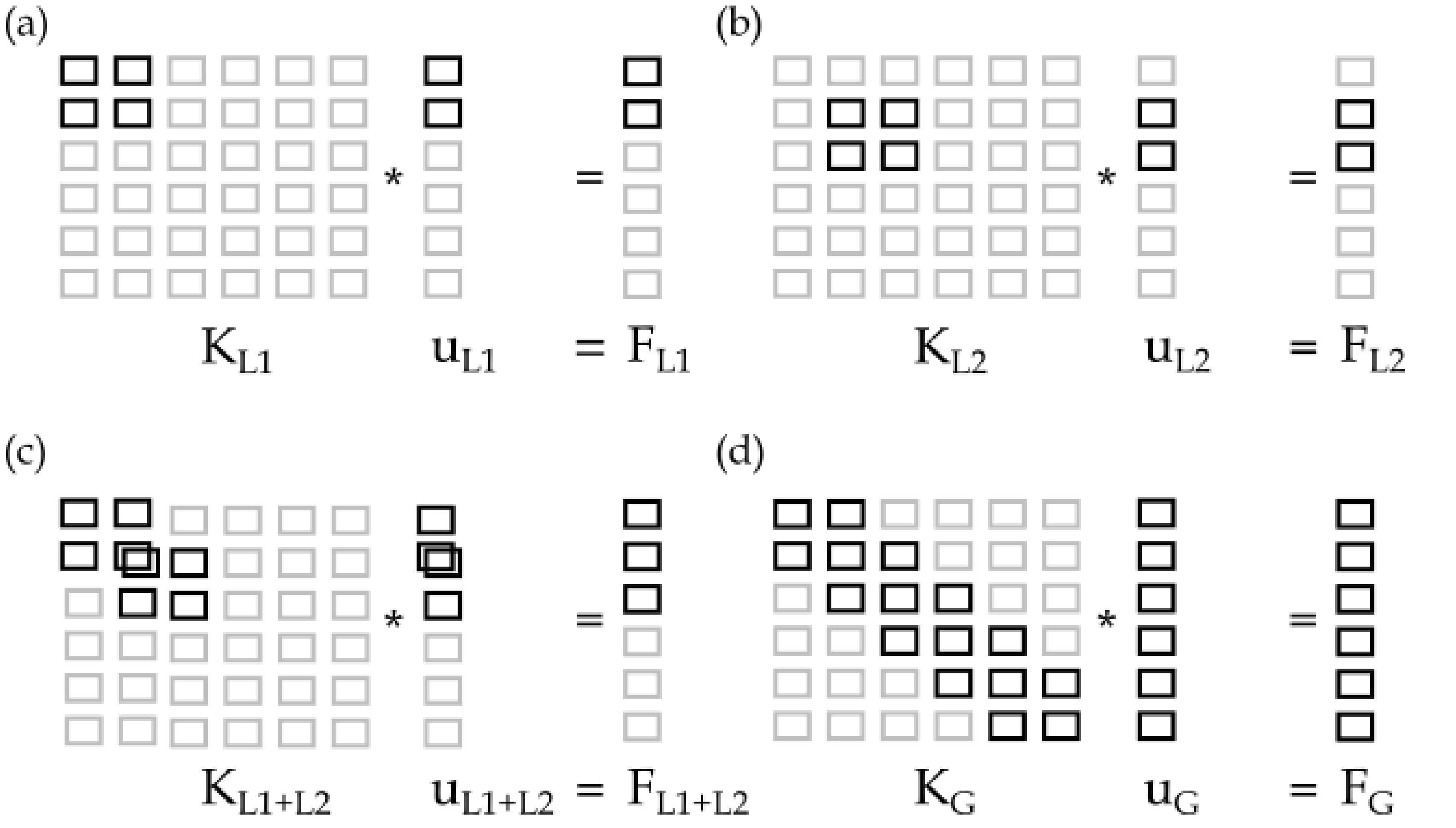
$$\begin{bmatrix} \alpha_{11} & \alpha_{12} & 0 \\ \alpha_{21} & \alpha_{22} + b_{11} & b_{12} \\ 0 & b_{21} & b_{22} \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 + h_1 \\ h_2 \end{bmatrix}$$

Finite Element Method

Following the previous approach, we can construct a global linear system of equations.

$$\mathbf{K_G}\{u_i\} = \mathbf{F_G}$$

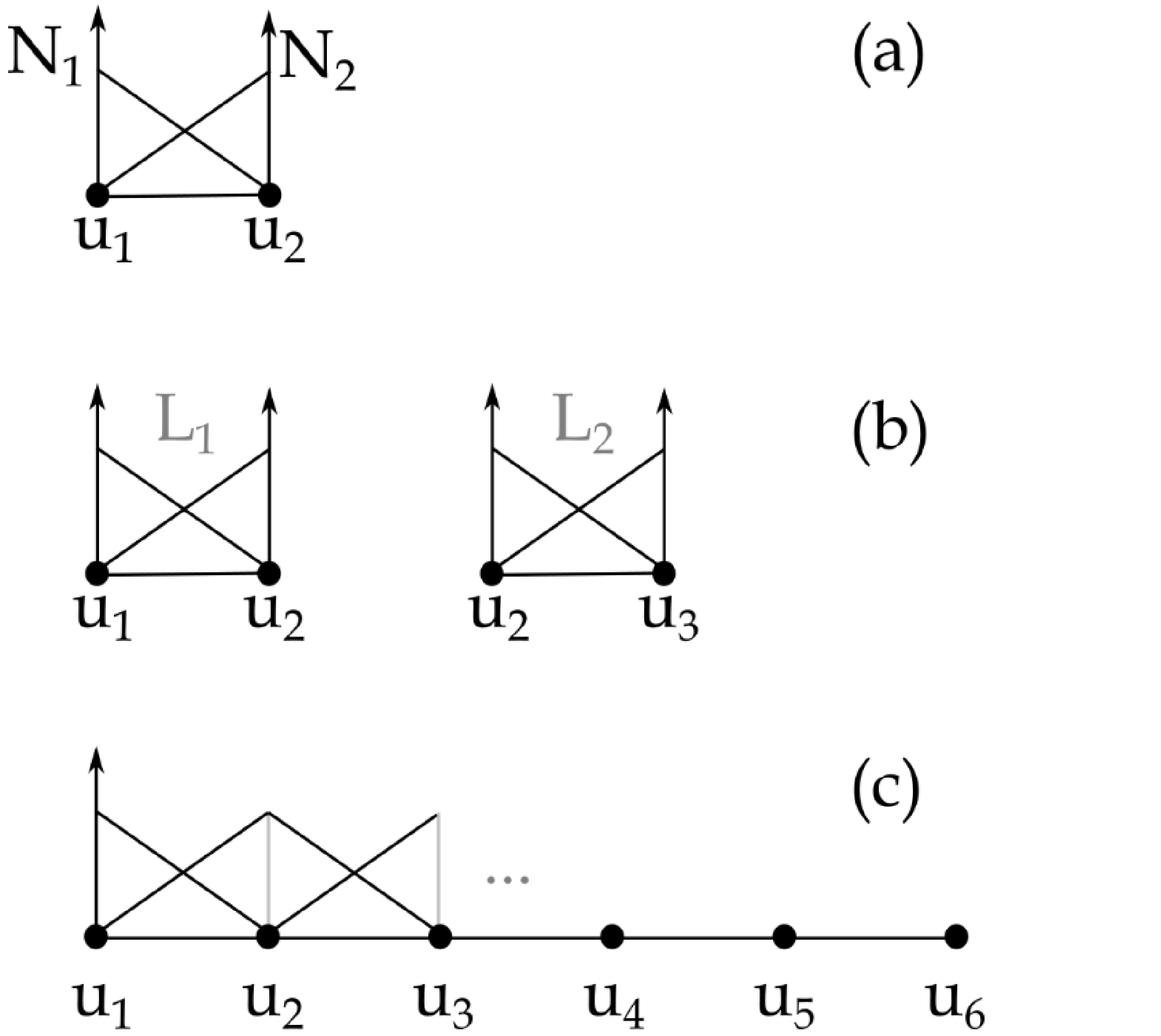
This system can be solved as shown previously for the FDM case (i.e. using backslash).


(a)
(b)
K_{L1} u_{L1} = F_{L1}
K_{L2} u_{L2} = F_{L2}
(c)
(d)
K_{L1+L2} u_{L1+L2} = F_{L1+L2}
K_G u_G = F_G

Finite Element Method

To proceed with the FEM code we need to create a relation (mapping) between the number of an element and its nodes. To do so, we will make a table El2N (Element to Node), where the columns represent the number of the elements and the lines contain the number of nodes. That is:

Element 1	Element 2	Element 3	…N
1	2	3	N
2	3	4	N+1



Finite Element Method

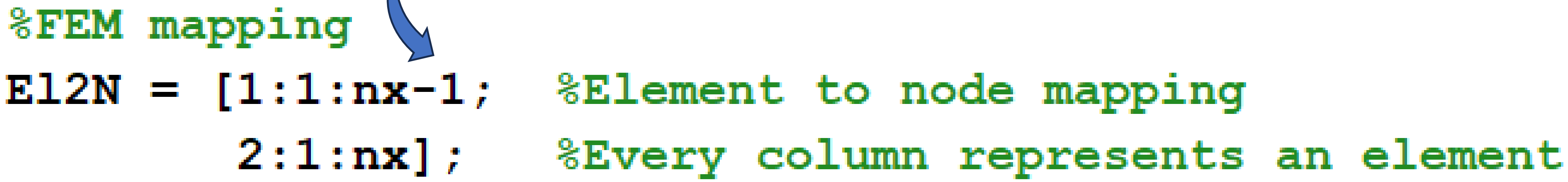
In table form:

Element 1	Element 2	Element 3	…N
1	2	3	N
2	3	4	N+1

The matrix El2N contains the mapping between Elements and Nodes.

In MATLAB:

$nel = nx - 1$



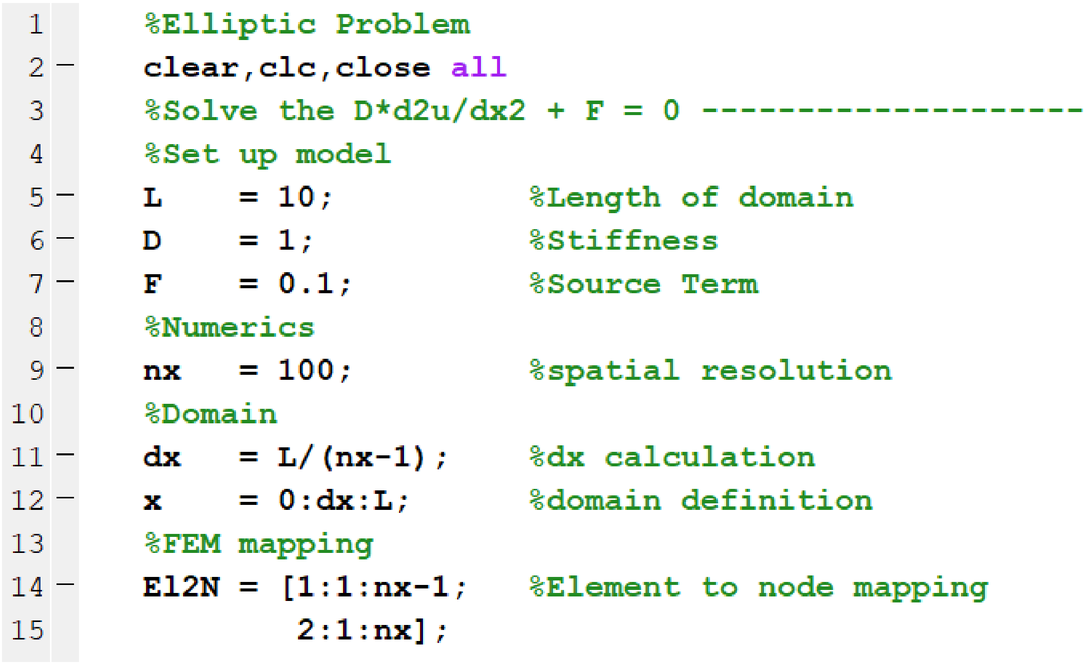
```
%FEM mapping
El2N = [1:1:nx-1;   %Element to node mapping
          2:1:nx];   %Every column represents an element
```

Finite Element Method

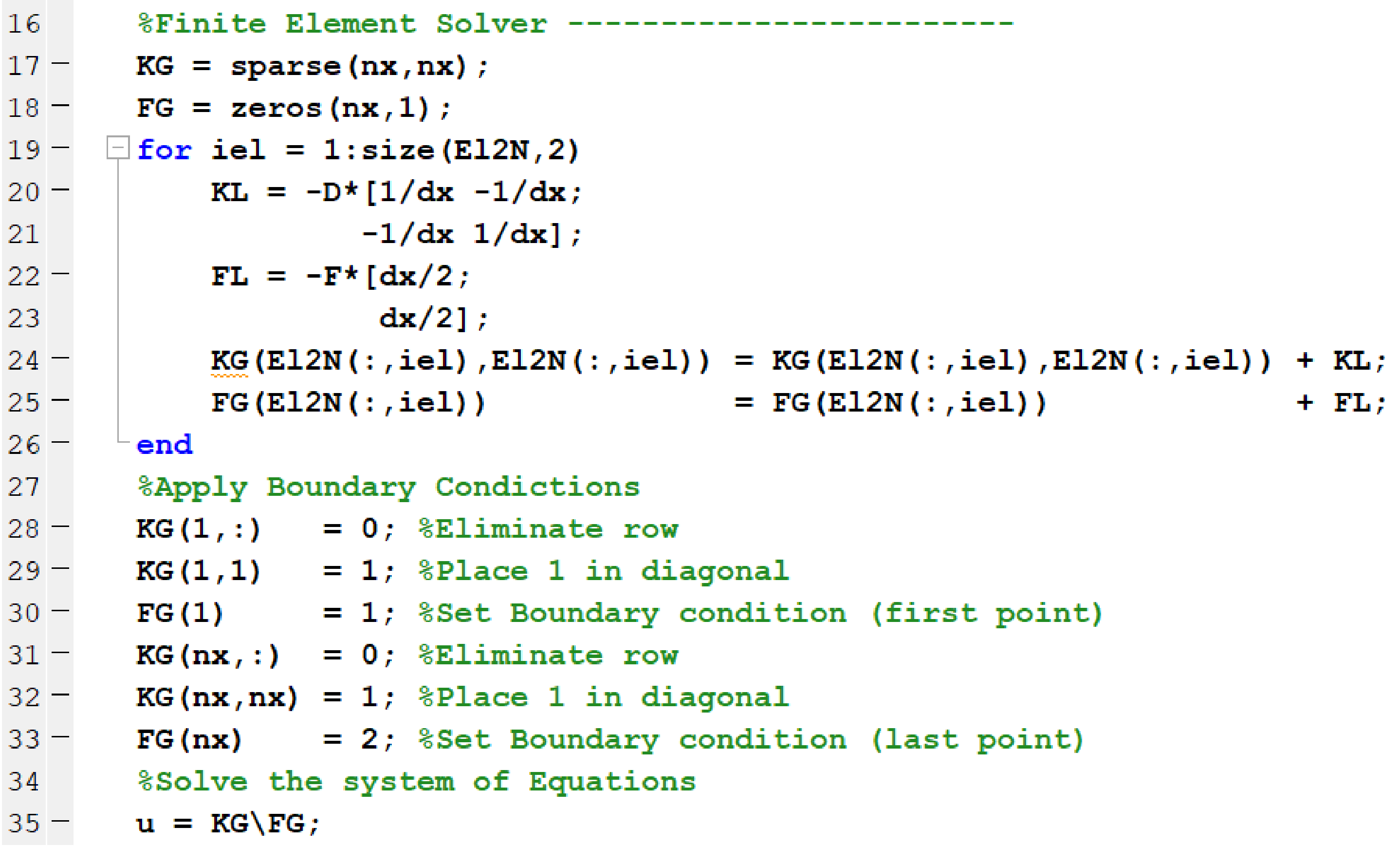
JG|U GEO SCIENCES MAINZ

Solving the steady-state problem

$$D\frac{d^2u}{dx^2}+F=0 \text{ with BC } \begin{cases}u(0)=1\\ u(L)=2\end{cases}$$



```
%Elliptic Problem
clear,clc,close all
%Solve the D*d2u/dx2 + F = 0 --------------------
%Set up model
L     = 10;          %Length of domain
D     = 1;           %Stiffness
F     = 0.1;         %Source Term
%Numerics
nx    = 100;         %spatial resolution
%Domain
dx    = L/(nx-1);    %dx calculation
x     = 0:dx:L;      %domain definition
%FEM mapping
El2N = [1:1:nx-1;    %Element to node mapping
        2:1:nx];
```



```
%Finite Element Solver ------------------------
KG = sparse(nx,nx);
FG = zeros(nx,1);
for iel = 1:size(El2N,2)
    KL = -D*[1/dx -1/dx;
            -1/dx 1/dx];
    FL = -F*[dx/2;
             dx/2];
    KG(El2N(:,iel),El2N(:,iel)) = KG(El2N(:,iel),El2N(:,iel)) + KL;
    FG(El2N(:,iel))              = FG(El2N(:,iel))              + FL;
end
%Apply Boundary Condictions
KG(1,:)    = 0; %Eliminate row
KG(1,1)    = 1; %Place 1 in diagonal
FG(1)      = 1; %Set Boundary condition (first point)
KG(nx,:)   = 0; %Eliminate row
KG(nx,nx)  = 1; %Place 1 in diagonal
FG(nx)     = 2; %Set Boundary condition (last point)
%Solve the system of Equations
u = KG\FG;
```

Finite Element Method

Solving the steady-state problem

$$D\frac{d^2u}{dx}+F=0 \text{ with BC } \begin{cases} u(0)=1 \\ u(L)=2 \end{cases}$$

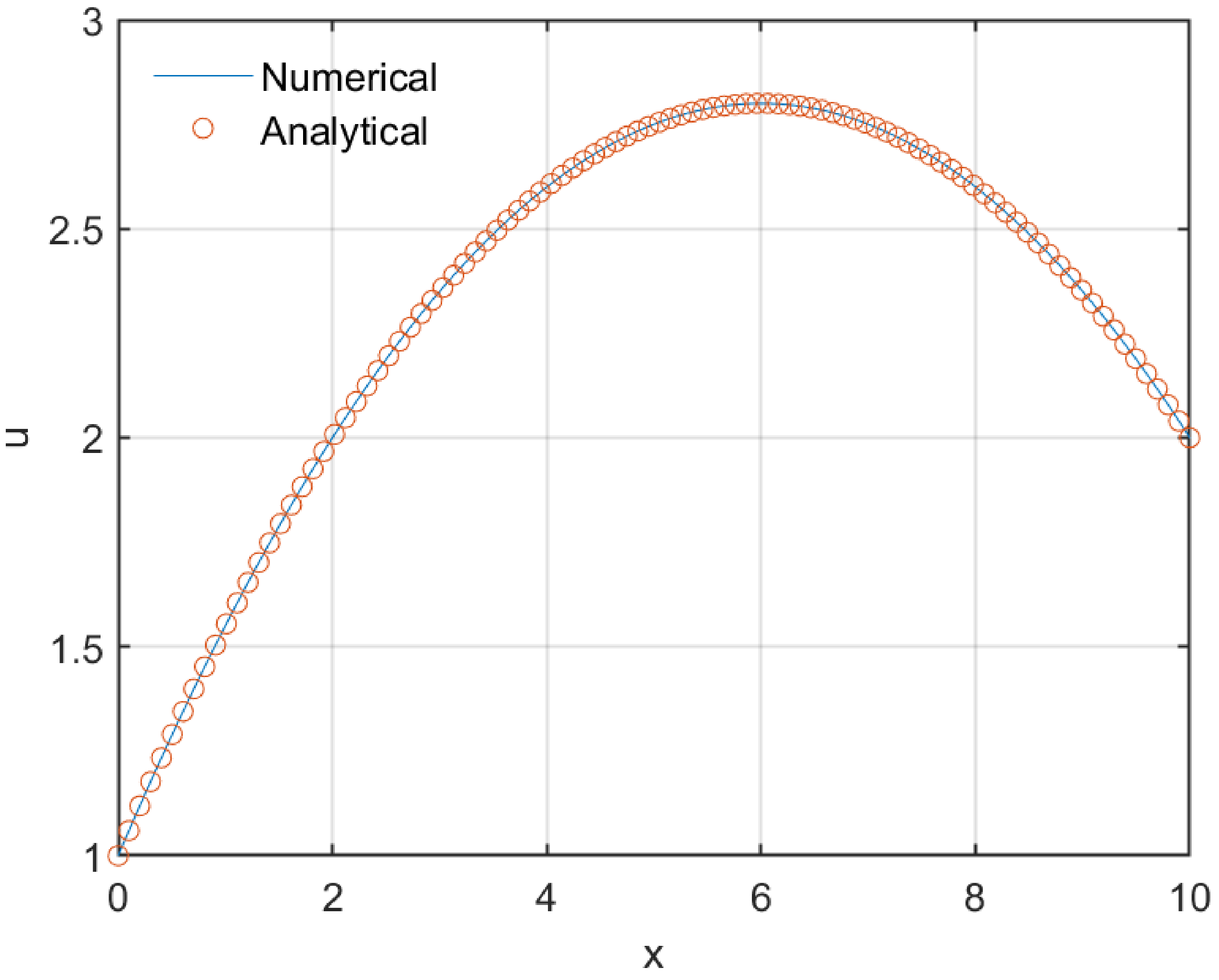
we can also test with the analytical solution. The analytical solution is given by:

$$u(x)=-\frac{F}{2D}x^2+c_1x+c_2$$

with

$$c_1=\frac{u(L)-u(0)}{L}+\frac{FL}{2D}$$

$$c_2=u(0)$$



Finite Element Method

Now we will consider a time-dependent problem like diffusion. We will start from the following form:

$$\frac{\partial C}{\partial t} = D\,\frac{\partial^2 C}{\partial x^2}$$

In a discretized form, this can be written as (implicit form):

$$\frac{\bar{C}^{new} - \bar{C}^{o}}{\Delta t} = D\,\frac{\partial^2 \bar{C}^{new}}{\partial x^2}$$

Finite Element Method

Lets consider a single element, bringing everything in the left side and multiplying with $N_1 : x \rightarrow N_1(x)$

$$\int_{x_1}^{x_2} N_1 \frac{\bar{C}^{new} - \bar{C}^{o}}{\Delta t} dx - \int_{x_1}^{x_2} N_1 D \frac{\partial^2 \bar{C}^{new}}{\partial x^2} dx = 0$$

We can now split the first term and rearrange to get:

$$\underbrace{\int_{x_1}^{x_2} N_1 \frac{\bar{C}^{new}}{\Delta t} dx}_{\text{new term}} - \underbrace{\int_{x_1}^{x_2} N_1 D \frac{\partial^2 \bar{C}^{new}}{\partial x^2} dx}_{\text{same as before}} = \underbrace{\int_{x_1}^{x_2} N_1 \frac{\bar{C}^{o}}{\Delta t} dx}_{\text{RHS based on } \bar{\text{C}}^{\text{o}}} \qquad (4)$$

Finite Element Method

By focusing on the diffusion (middle) term of Eq. (4), performing integration by parts and ignoring the boundary terms leads to:

$$-\int_{x_1}^{x_2} N_1 D \frac{\partial^2 \bar{C}^{new}}{\partial x^2} dx = \int_{x_1}^{x_2} \frac{\partial N_1}{\partial x} D \frac{\partial \bar{C}^{new}}{\partial x} dx$$

note the sign change!

Thus, Eq. (4) becomes

$$\underbrace{\int_{x_1}^{x_2} N_1 \frac{\bar{C}^{new}}{\Delta t} dx}_{\text{new term}} + \underbrace{\int_{x_1}^{x_2} \frac{\partial N_1}{\partial x} D \frac{\partial \bar{C}^{new}}{\partial x} dx}_{\text{same as before}} = \underbrace{\int_{x_1}^{x_2} N_1 \frac{\bar{C}^{o}}{\Delta t} dx}_{\text{RHS based on } \bar{\text{C}}^{\text{o}}}$$

Finite Element Method

Similarly we can expand $\bar{C} = \sum_{i=1}^{2} N_i \, \bar{C}_i$ then we have: $\bar{C} = \begin{bmatrix} N_1 & N_2 \end{bmatrix} \begin{Bmatrix} \bar{C}_1 \\ \bar{C}_2 \end{Bmatrix}$

Therefore, the time derivative can be approximated as:

$$\frac{\partial \bar{C}}{\partial t} \approx \frac{1}{\Delta t}\left(\begin{bmatrix} N_1 & N_2 \end{bmatrix} \begin{Bmatrix} \bar{C}_1^{new} \\ \bar{C}_2^{new} \end{Bmatrix} - \begin{bmatrix} N_1 & N_2 \end{bmatrix} \begin{Bmatrix} \bar{C}_1^{old} \\ \bar{C}_2^{old} \end{Bmatrix} \right)$$

which, after multiplying with the column of the shape functions we have:

$$\begin{Bmatrix} N_1 \\ N_2 \end{Bmatrix} \frac{\partial \bar{C}}{\partial t} \approx \frac{1}{\Delta t} \begin{Bmatrix} N_1 \\ N_2 \end{Bmatrix} \left(\begin{bmatrix} N_1 & N_2 \end{bmatrix} \begin{Bmatrix} \bar{C}_1^{new} \\ \bar{C}_2^{new} \end{Bmatrix} - \begin{bmatrix} N_1 & N_2 \end{bmatrix} \begin{Bmatrix} \bar{C}_1^{old} \\ \bar{C}_2^{old} \end{Bmatrix} \right)$$

Finite Element Method

Finally, by grouping all the terms, we obtain the following form:

$$[\mathbf{M}+\mathbf{K}]\begin{Bmatrix}\overline{C_1}^{new}\\ \overline{C_2}^{new}\end{Bmatrix}=\mathbf{M}\begin{Bmatrix}\overline{C_1}^{old}\\ \overline{C_2}^{old}\end{Bmatrix}$$

$$\mathbf{M}=\frac{1}{\Delta t}\int_{x_1}^{x_2}\begin{bmatrix}N_1N_1 & N_1N_2\\ N_2N_1 & N_2N_2\end{bmatrix}dx$$

$$\mathbf{K}=\mathrm{D}\int_{x_1}^{x_2}\begin{bmatrix}\frac{dN_1(x)}{dx}\frac{dN_1(x)}{dx} & \frac{dN_1(x)}{dx}\frac{dN_2(x)}{dx}\\ \frac{dN_2(x)}{dx}\frac{dN_1(x)}{dx} & \frac{dN_2(x)}{dx}\frac{dN_2(x)}{dx}\end{bmatrix}dx$$

Finite Element Method

Note that the matrix $\mathbf{K}$ is similar (only different sign) to what it was before (slide 90):

$$[\mathbf{M}+\mathbf{K}]\begin{Bmatrix}\overline{C_1}^{new}\\ \overline{C_2}^{new}\end{Bmatrix}=\mathbf{M}\begin{Bmatrix}\overline{C_1}^{old}\\ \overline{C_2}^{old}\end{Bmatrix}$$

$$\mathbf{K}=D\begin{bmatrix}\frac{1}{\Delta x} & -\frac{1}{\Delta x}\\ -\frac{1}{\Delta x} & \frac{1}{\Delta x}\end{bmatrix}$$

$$\mathbf{M}=\begin{bmatrix}\frac{\Delta x}{3\Delta t} & \frac{\Delta x}{6\Delta t}\\ \frac{\Delta x}{6\Delta t} & \frac{\Delta x}{3\Delta t}\end{bmatrix}$$

Finite Element Method

Alternatively, we can combine the matrices as:

$$\mathbf{L}\begin{Bmatrix}\overline{C_1}^{new}\\\overline{C_2}^{new}\end{Bmatrix}=\mathbf{R}\begin{Bmatrix}\overline{C_1}^{old}\\\overline{C_2}^{old}\end{Bmatrix}$$

$$\mathbf{L}=\begin{bmatrix}\frac{\Delta x}{3\Delta t}+\frac{D}{\Delta x} & \frac{\Delta x}{6\Delta t}-\frac{D}{\Delta x}\\\frac{\Delta x}{6\Delta t}-\frac{D}{\Delta x} & \frac{\Delta x}{3\Delta t}+\frac{D}{\Delta x}\end{bmatrix}\qquad\mathbf{R}=\begin{bmatrix}\frac{\Delta x}{3\Delta t} & \frac{\Delta x}{6\Delta t}\\\frac{\Delta x}{6\Delta t} & \frac{\Delta x}{3\Delta t}\end{bmatrix}$$

We can thus assemble the global matrix as before and solve for the concentration using $\boldsymbol{C}=\boldsymbol{L_G}\backslash\left(\mathbf{R_G}\{\boldsymbol{C^{old}}\}\right)$

Finite Element Method

In other words, in the previous problem (without time derivatives) we had

$$\mathbf{K_G}\{u_i\} = \mathbf{F_G}$$

and now (with time derivatives) we have:

$$\mathbf{L_G}\left\{\bar{C_i}^{new}\right\} = \mathbf{R_G}\left\{\bar{C_i}^{old}\right\}$$

Compared to the previous problem, there are two main differences:

1) The local L matrix is the sum of K and M matrices
2) The local R matrix needs to be updated in time using the old values of C.

Finite Element Method

Exercise 9

Make a finite element code to model chemical diffusion for a given time t. Use the parameters given below and compare the result with the analytical solution C_{an}. Note that $x \in [0, L]$

$$\Delta t = 10^{-2}$$

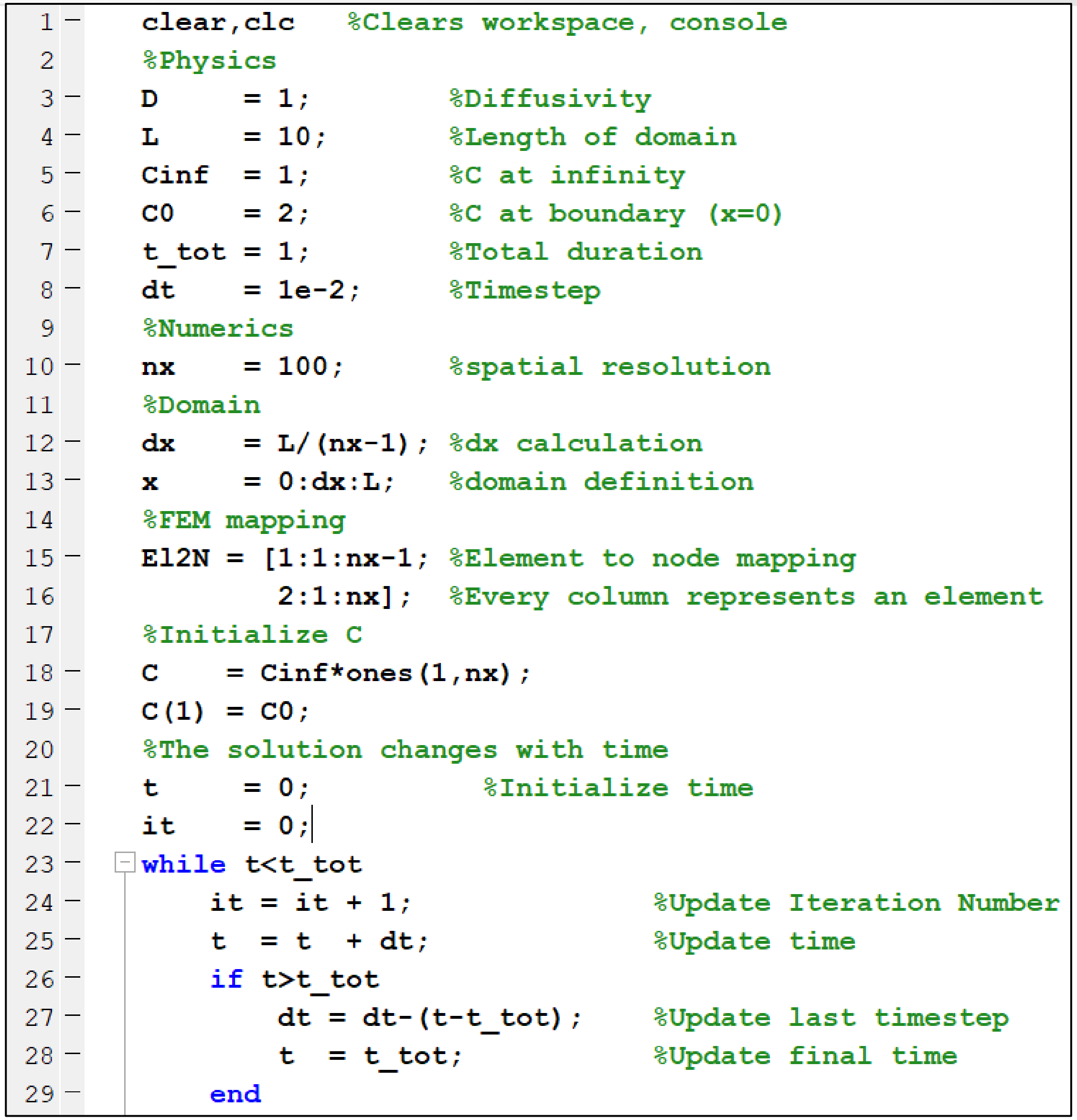
$$D = 1$$

$$L = 10$$

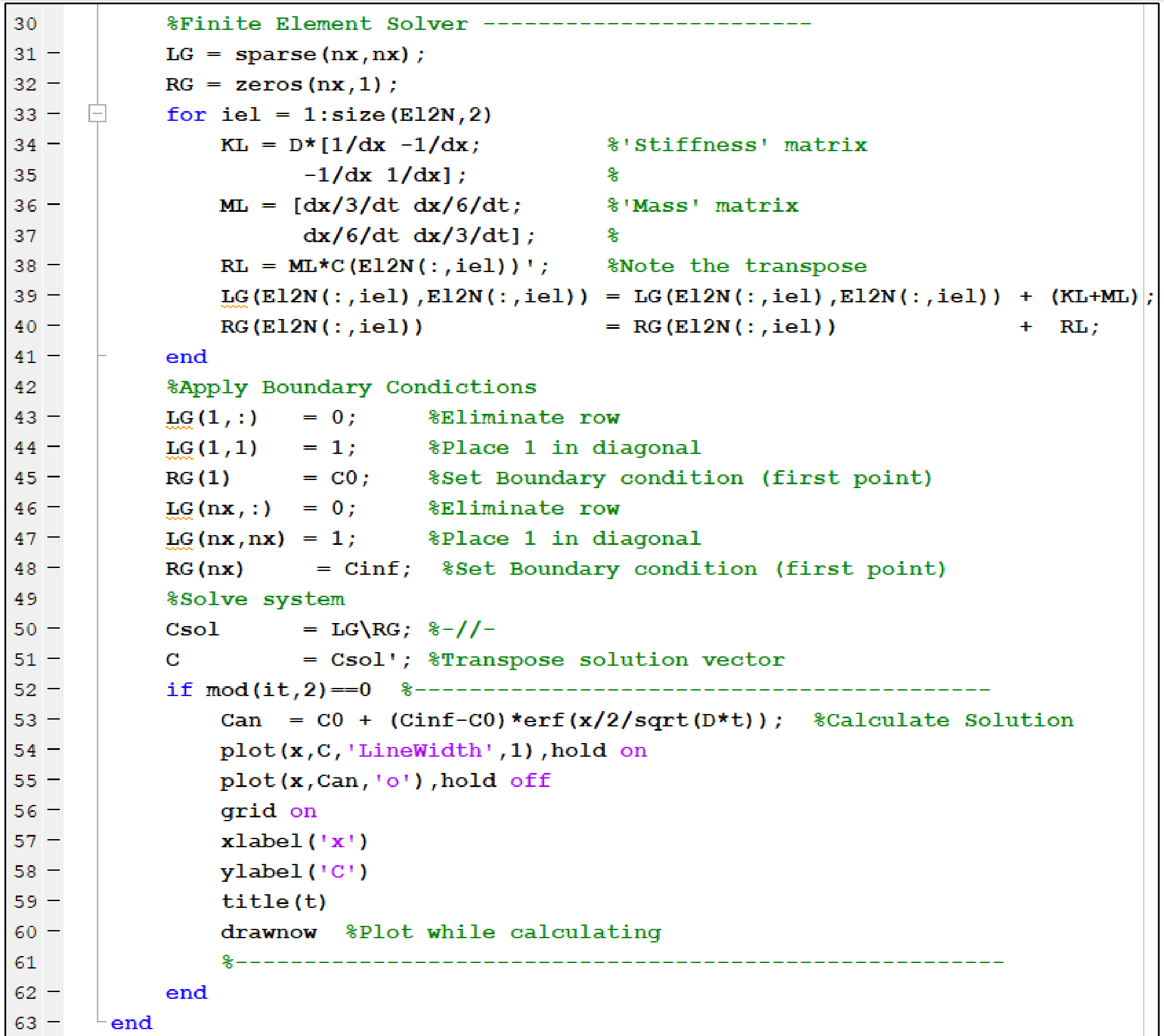
$$C(0,t) = \begin{cases} C_{inf} = 1\,, x > 0 \\ C_0 = 2, x = 0 \end{cases}$$

$$C_{an} = C_0 + (C_{inf} - C_0)\operatorname{erf}\left(\frac{x}{2\sqrt{Dt}}\right)$$

Finite Element Method



```matlab
clear,clc   %Clears workspace, console
%Physics
D     = 1;        %Diffusivity
L     = 10;       %Length of domain
Cinf  = 1;        %C at infinity
C0    = 2;        %C at boundary (x=0)
t_tot = 1;        %Total duration
dt    = 1e-2;     %Timestep
%Numerics
nx    = 100;      %spatial resolution
%Domain
dx    = L/(nx-1); %dx calculation
x     = 0:dx:L;   %domain definition
%FEM mapping
El2N = [1:1:nx-1; %Element to node mapping
        2:1:nx];  %Every column represents an element
%Initialize C
C    = Cinf*ones(1,nx);
C(1) = C0;
%The solution changes with time
t     = 0;          %Initialize time
it    = 0;
while t<t_tot
    it = it + 1;              %Update Iteration Number
    t  = t  + dt;             %Update time
    if t>t_tot
        dt = dt-(t-t_tot);    %Update last timestep
        t  = t_tot;           %Update final time
    end
```

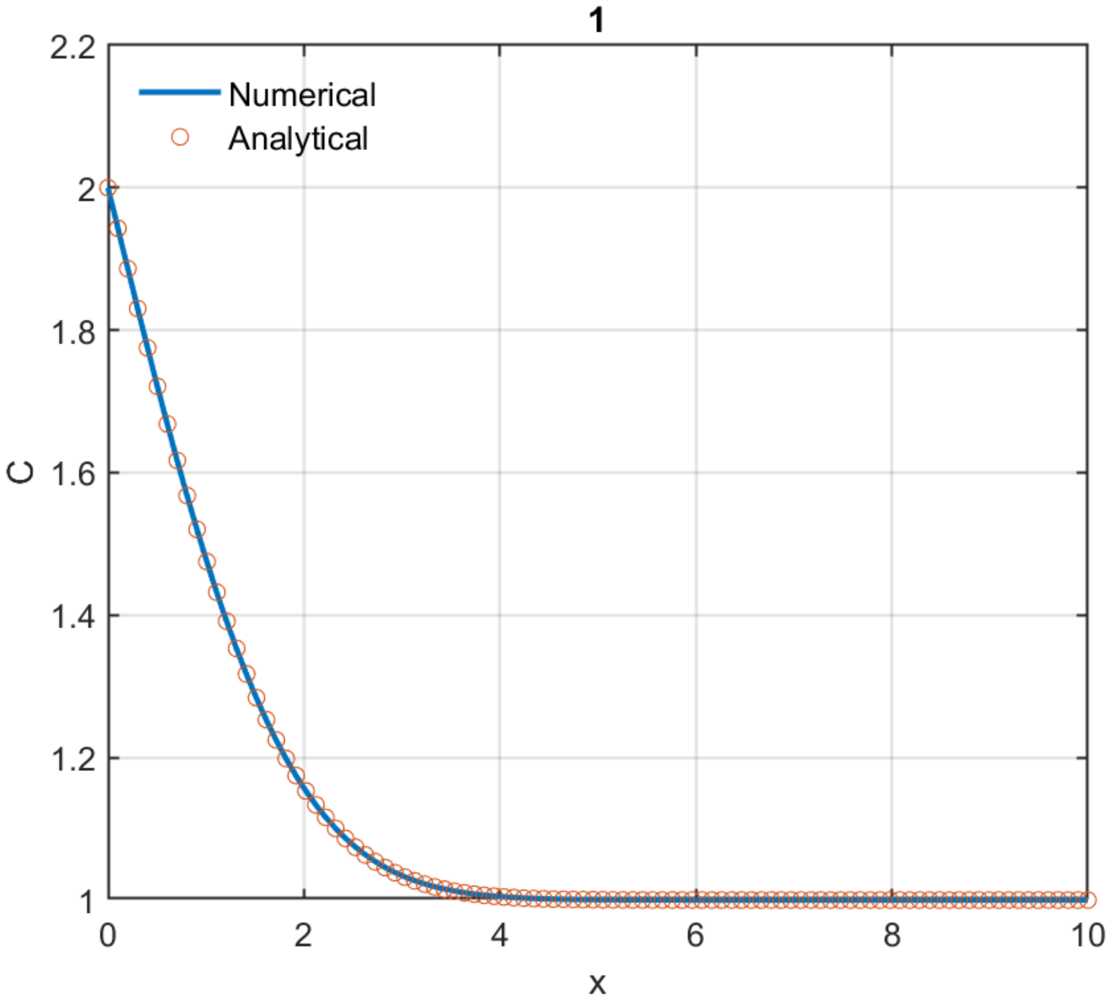


```matlab
    %Finite Element Solver ------------------------
    LG = sparse(nx,nx);
    RG = zeros(nx,1);
    for iel = 1:size(El2N,2)
        KL = D*[1/dx -1/dx;         %'Stiffness' matrix
              -1/dx 1/dx];          %
        ML = [dx/3/dt dx/6/dt;      %'Mass' matrix
              dx/6/dt dx/3/dt];     %
        RL = ML*C(El2N(:,iel))';    %Note the transpose
        LG(El2N(:,iel),El2N(:,iel)) = LG(El2N(:,iel),El2N(:,iel)) + (KL+ML);
        RG(El2N(:,iel))             = RG(El2N(:,iel))             +  RL;
    end
    %Apply Boundary Conditions
    LG(1,:)   = 0;     %Eliminate row
    LG(1,1)   = 1;     %Place 1 in diagonal
    RG(1)     = C0;    %Set Boundary condition (first point)
    LG(nx,:)  = 0;     %Eliminate row
    LG(nx,nx) = 1;     %Place 1 in diagonal
    RG(nx)     = Cinf;  %Set Boundary condition (first point)
    %Solve system
    Csol      = LG\RG; %-//-
    C         = Csol'; %Transpose solution vector
    if mod(it,2)==0  %-----------------------------------------
        Can  = C0 + (Cinf-C0)*erf(x/2/sqrt(D*t));  %Calculate Solution
        plot(x,C,'LineWidth',1),hold on
        plot(x,Can,'o'),hold off
        grid on
        xlabel('x')
        ylabel('C')
        title(t)
        drawnow  %Plot while calculating
        %-------------------------------------------------------
    end
end
```

Finite Element Method

Note that the analytical solution assumes far-field boundary conditions (at infinity).

This means that our numerical boundary conditions will not be a good approximation at large timescales (when diffusion affects the boundary)



Finite Element Method

Exercise 10

Make a finite element code for diffusion in minerals. You can assume an initial concentration profile with a 'step' (chose any value) and a grain of 1mm in length. If the boundaries are far away from the step we can ignore the boundary conditions. Use the parameters for Sr in apatite $E = 65{,}000$ cal/mol and $D_0 = 2.7 \cdot 10^{-3}$ cm²/s (Cherniak & Ryerson, 1993). Use $R = 1.987$ cal/mol/K. You can use different temperatures and check the difference in the results.

Finite Element Method

Exercise 11

Extend the previous code to solve the diffusion problem assuming an asymptotic cooling history of the form

$$T(t) = \frac{T_{\mathrm{MAX}}}{1 + \frac{st}{T_{\mathrm{MAX}}}}$$

where T_{MAX} is 1000K and s is 50K/Myr.

Exercise 12

Use the time-transformation technique to verify your results.

Multicomponent Diffusion

Multicomponent Diffusion (optional)

Multicomponent Diffusion

So far we have considered diffusion problems where for each point in space we had one concentration value. That is:

$$\frac{\partial C}{\partial t} = D\,\frac{\partial^2 C}{\partial x^2}$$

In the case of multicomponent diffusion, we can have many compositional variables in space (not all independent). For example, if we use 4 oxides to describe our composition, we would need 3 independent variables. That is because:

$$\sum_{i=1}^{\kappa} C_i = 1$$

which means that only $\kappa - 1$ components are independent. To obtain the concentrations one needs to multiply by density.

Multicomponent Diffusion

This allows us to write the multicomponent diffusion equation as:

$$\frac{\partial}{\partial t}\{\boldsymbol{C}\} = \frac{\partial}{\partial x}\left(\boldsymbol{D}\frac{\partial}{\partial x}\{\boldsymbol{C}\}\right) \qquad (5)$$

where now $\boldsymbol{C}$ represents a ‘vector’ of $\kappa - 1$ compositions for each point in space. In this case, $\boldsymbol{D}$ is an $(\kappa - 1) * (\kappa - 1)$ matrix. By assuming that $\boldsymbol{D}$ is independent of composition, Eq. (5) can be written as:

$$\frac{\partial}{\partial t}\{\boldsymbol{C}\} = \boldsymbol{D}\frac{\partial^2}{\partial x^2}\{\boldsymbol{C}\} \qquad (6)$$

Multicomponent Diffusion

For $\kappa = 4$ (3 independent), Eq. (6) can be written as:

$$\frac{\partial}{\partial t}\begin{Bmatrix} C_1 \\ C_2 \\ C_3 \end{Bmatrix} = \begin{bmatrix} D_{11} & D_{12} & D_{13} \\ D_{21} & D_{22} & D_{23} \\ D_{31} & D_{32} & D_{33} \end{bmatrix} \frac{\partial^2}{\partial x^2}\begin{Bmatrix} C_1 \\ C_2 \\ C_3 \end{Bmatrix}$$

which can be expanded as:

$$\begin{bmatrix} \frac{\partial C_1}{\partial t} \\ \frac{\partial C_2}{\partial t} \\ \frac{\partial C_3}{\partial t} \end{bmatrix} = \begin{bmatrix} D_{11}\frac{\partial^2 C_1}{\partial x^2} + D_{12}\frac{\partial^2 C_2}{\partial x^2} + D_{13}\frac{\partial^2 C_3}{\partial x^2} \\ D_{21}\frac{\partial^2 C_1}{\partial x^2} + D_{22}\frac{\partial^2 C_2}{\partial x^2} + D_{23}\frac{\partial^2 C_3}{\partial x^2} \\ D_{31}\frac{\partial^2 C_1}{\partial x^2} + D_{32}\frac{\partial^2 C_2}{\partial x^2} + D_{33}\frac{\partial^2 C_3}{\partial x^2} \end{bmatrix}$$

Multicomponent Diffusion

The previous form shows that each component may change as a function of the concentration gradients of the other components. This behavior is a consequence of the fact that diffusion fluxes are proportional to chemical potential gradients (see Balluffi et al., 2005, for theory, and, Chakraborty, 1994, for applications relevant to geosciences).

In this example we will focus on the case where the diffusion matrix is constant. The solution procedure closely follows the method of Toor (1964). Starting by Eq. (6) we can recognize that the derivative operators are 'linear' operators. Thus, Eq. (6) can be viewed as an equation of matrix-vector products.

$$L_t\{\boldsymbol{C}\} = \boldsymbol{D}L_{x2}\{\boldsymbol{C}\}$$

where L_t and L_{x2} are the linear operators that represent the differentials.

Multicomponent Diffusion

The advantage of the previous form is not yet obvious. However, we should mention that the linearity of the previous operators allows a great simplification in our problem. This is because the matrix $\boldsymbol{D}$ can be diagonalized. More specifically, **if an m by m (square) matrix D has m independent eigenvectors, then this matrix is diagonalizable**. That is:

$$\boldsymbol{D} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}$$

where $\boldsymbol{V}$ is the matrix that contains the eigenvectors of $\boldsymbol{D}$, $\boldsymbol{\Lambda}$ is a diagonal matrix that contains the eigenvalues of $\boldsymbol{D}$, and $\boldsymbol{V}^{-1}$ is the inverse of $\boldsymbol{V}$. For a 2 by 2 case the previous can be written in full as:

$$\begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} = \begin{bmatrix} V_1^1 & V_1^2 \\ V_2^1 & V_2^2 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} V_1^1 & V_1^2 \\ V_2^1 & V_2^2 \end{bmatrix}^{-1}$$

Multicomponent Diffusion

Thus, our multicomponent diffusion equations become:

$$L_t\{\boldsymbol{C}\} = \underbrace{\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}}_{\boldsymbol{D}} L_{x2}\{\boldsymbol{C}\}$$

At this point we can pre-multiply the previous equation by $\boldsymbol{V}^{-1}$. This results to:

$$\boldsymbol{V}^{-1}L_t\{\boldsymbol{C}\} = \boldsymbol{\Lambda}\boldsymbol{V}^{-1}L_{x2}\{\boldsymbol{C}\}$$

However, since the derivative operators are linear, we can place $\boldsymbol{V}^{-1}$ inside the derivatives to obtain:

$$L_t\{\boldsymbol{V}^{-1}\boldsymbol{C}\} = \boldsymbol{\Lambda}L_{x2}\{\boldsymbol{V}^{-1}\boldsymbol{C}\}$$

by setting $\boldsymbol{W} = \boldsymbol{V}^{-1}\boldsymbol{C}$ in the last equation we get:

$$L_t\{\boldsymbol{W}\} = \boldsymbol{\Lambda}L_{x2}\{\boldsymbol{W}\}$$

Multicomponent Diffusion

The equation $L_t(\boldsymbol{W}) = \boldsymbol{\Lambda} L_{x2}(\boldsymbol{W})$ represents a diagonal system of diffusion equations. That is:

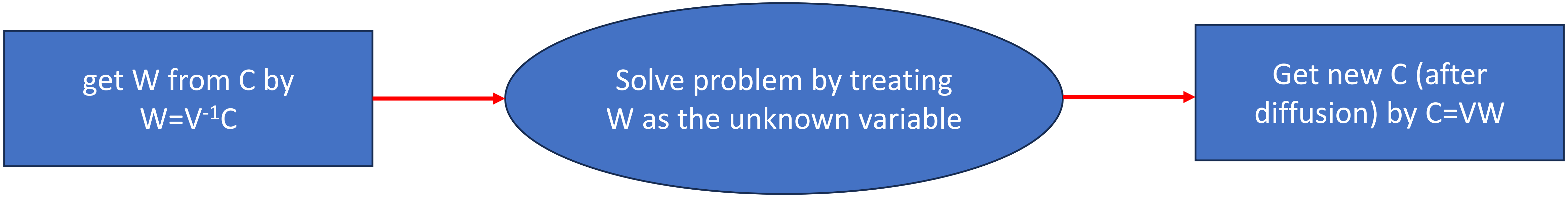
$$\frac{\partial}{\partial t}\{\boldsymbol{W}\} = \boldsymbol{\Lambda}\frac{\partial^2}{\partial x^2}\{\boldsymbol{W}\} \tag{7}$$

or, in the case of 3 independent components, in full form:

$$\begin{bmatrix} \frac{\partial W_1}{\partial t} \\ \frac{\partial W_2}{\partial t} \\ \frac{\partial W_3}{\partial t} \end{bmatrix} = \begin{bmatrix} \Lambda_1 & & \\ & \Lambda_2 & \\ & & \Lambda_3 \end{bmatrix} \begin{bmatrix} \frac{\partial^2 W_1}{\partial x^2} \\ \frac{\partial^2 W_2}{\partial x^2} \\ \frac{\partial^2 W_3}{\partial x^2} \end{bmatrix} = \begin{bmatrix} \Lambda_1 \frac{\partial^2 W_1}{\partial x^2} \\ \Lambda_2 \frac{\partial^2 W_2}{\partial x^2} \\ \Lambda_3 \frac{\partial^2 W_3}{\partial x^2} \end{bmatrix}$$

Multicomponent Diffusion

The advantage of the previous form is that all diffusion equations are independent from each other (decoupled). Thus, we could use all the tools we have already (numerical or analytical) to solve the equations. To do so, we first need to transform the compositional profiles (C) into the new independent variables (W). Then we can proceed by solving the diffusion problem, and then we can transform the problem back.


get W from C by $W=V^{-1}C$

Solve problem by treating W as the unknown variable

Get new C (after diffusion) by C=VW

Remember that this was only possible because the matrix of diffusion coefficients was independent of composition.

Multicomponent Diffusion

In MATLAB/OCTAVE, the matrix containing the eigenvectors and the diagonal matrix containing the eigenvalues are given by the function ‘eig’.

```
[V,LAM]=eig(D)
```

using the previous command, we will obtain the matrix of eigenvectors V and the matrix of eigenvalues LAM of the square matrix D. The inverse of matrix V can be obtained by:

```
invV=inv(V)
```

where invV is the variable that contains the inverse of matrix V.

Multicomponent Diffusion

Exercise 13 (Advanced)

Lets consider a domain $x \in [-1,1]$ and a system with three independent components that have the following initial values.

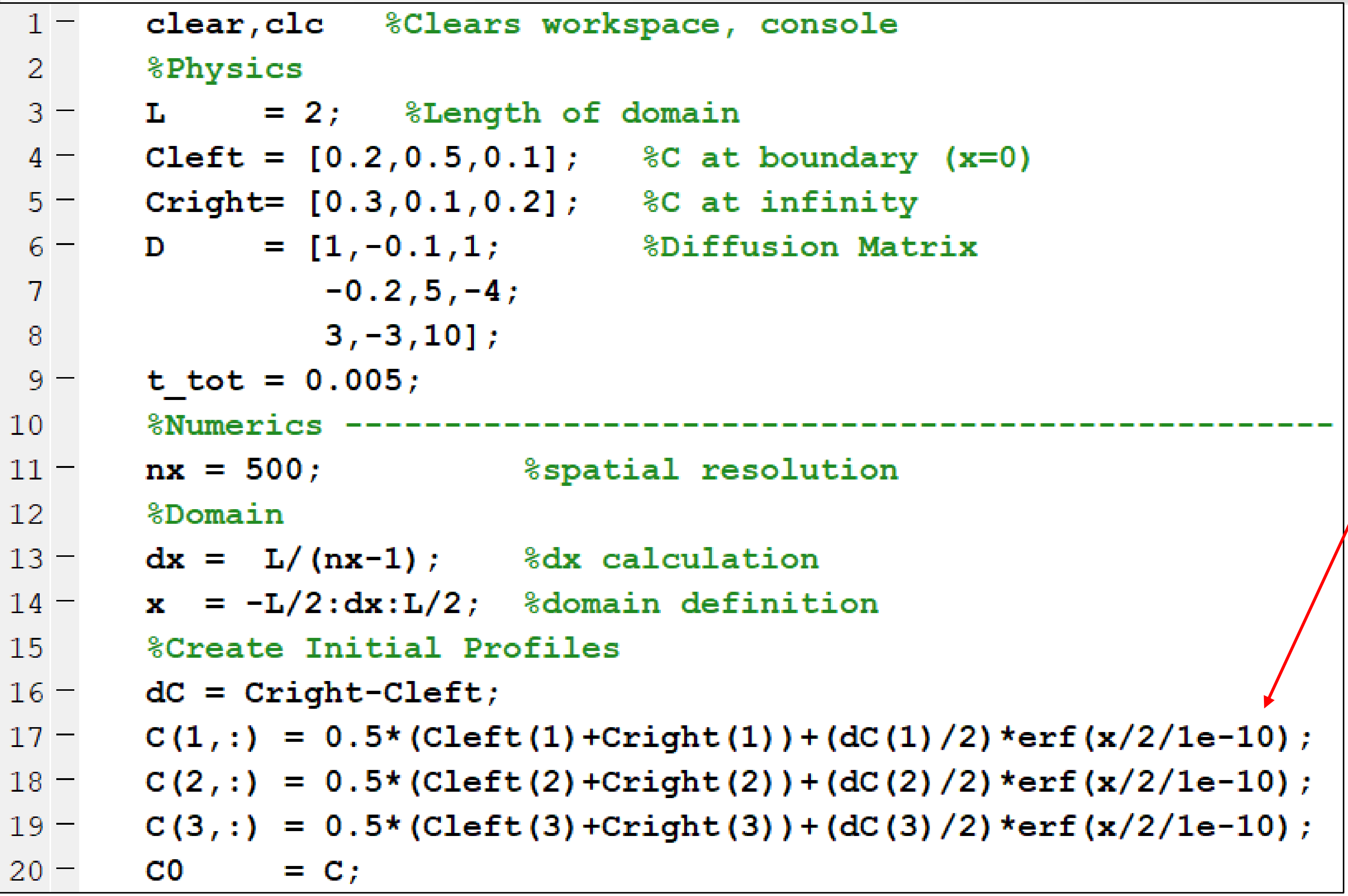
$$C_1 = \begin{cases} 0.2, x < 0 \\ 0.3, x \geq 0 \end{cases} \quad C_2 = \begin{cases} 0.5, x < 0 \\ 0.1, x \geq 0 \end{cases} \quad C_3 = \begin{cases} 0.1, x < 0 \\ 0.2, x \geq 0 \end{cases}$$

The diffusion coefficient matrix $\boldsymbol{D}$ is given by:

$$\boldsymbol{D} = \begin{bmatrix} 1 & -0.1 & 1 \\ -0.2 & 5 & -4 \\ 3 & -3 & 10 \end{bmatrix}$$

Calculate the compositions after the time $t = 0.05$.

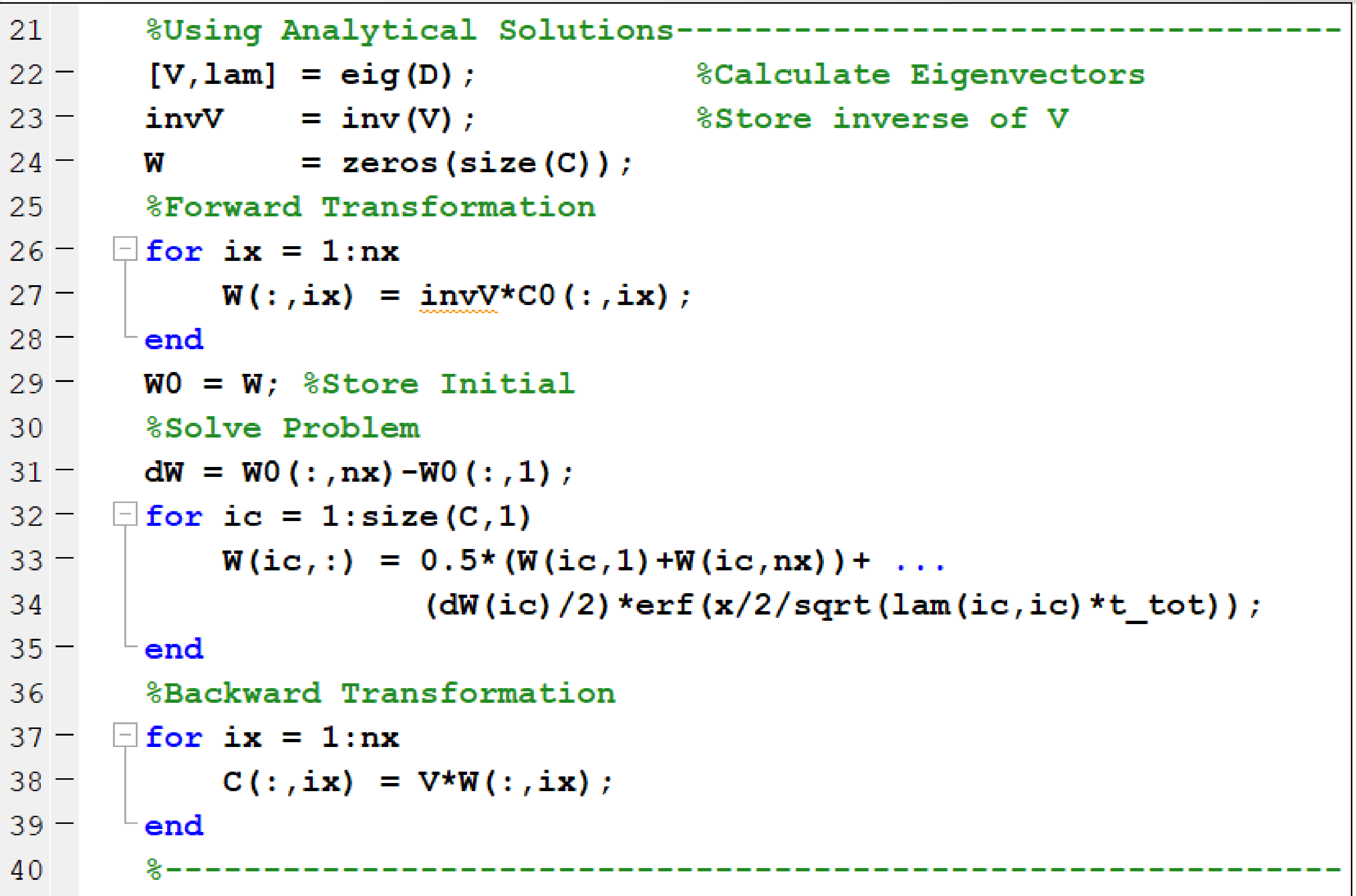
Multicomponent Diffusion



```
clear,clc   %Clears workspace, console
%Physics
L     = 2;   %Length of domain
Cleft = [0.2,0.5,0.1];   %C at boundary (x=0)
Cright= [0.3,0.1,0.2];   %C at infinity
D     = [1,-0.1,1;       %Diffusion Matrix
         -0.2,5,-4;
         3,-3,10];
t_tot = 0.005;
%Numerics --------------------------------------------------
nx = 500;          %spatial resolution
%Domain
dx =  L/(nx-1);     %dx calculation
x  = -L/2:dx:L/2;  %domain definition
%Create Initial Profiles
dC = Cright-Cleft;
C(1,:)  = 0.5*(Cleft(1)+Cright(1))+(dC(1)/2)*erf(x/2/1e-10);
C(2,:)  = 0.5*(Cleft(2)+Cright(2))+(dC(2)/2)*erf(x/2/1e-10);
C(3,:)  = 0.5*(Cleft(3)+Cright(3))+(dC(3)/2)*erf(x/2/1e-10);
C0      = C;
```

The number 10^{-10} is introduced to create a step-like initial profile

Multicomponent Diffusion

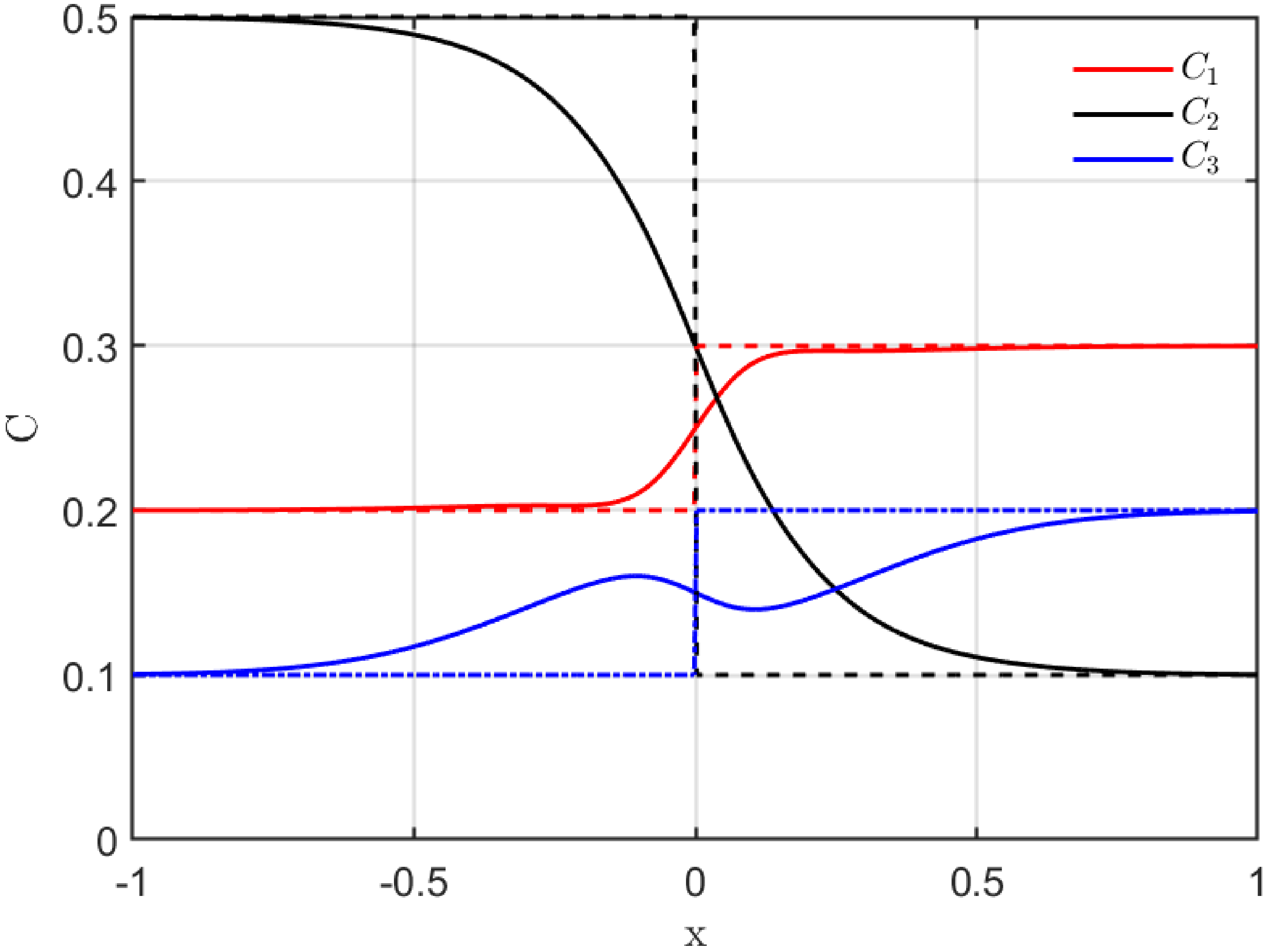


```
%Using Analytical Solutions----------------------------------
[V,lam] = eig(D);           %Calculate Eigenvectors
invV    = inv(V);           %Store inverse of V
W       = zeros(size(C));
%Forward Transformation
for ix = 1:nx
    W(:,ix) = invV*C0(:,ix);
end
W0 = W; %Store Initial
%Solve Problem
dW = W0(:,nx)-W0(:,1);
for ic = 1:size(C,1)
    W(ic,:) = 0.5*(W(ic,1)+W(ic,nx))+ ...
              (dW(ic)/2)*erf(x/2/sqrt(lam(ic,ic)*t_tot));
end
%Backward Transformation
for ix = 1:nx
    C(:,ix) = V*W(:,ix);
end
%------------------------------------------------------------
```

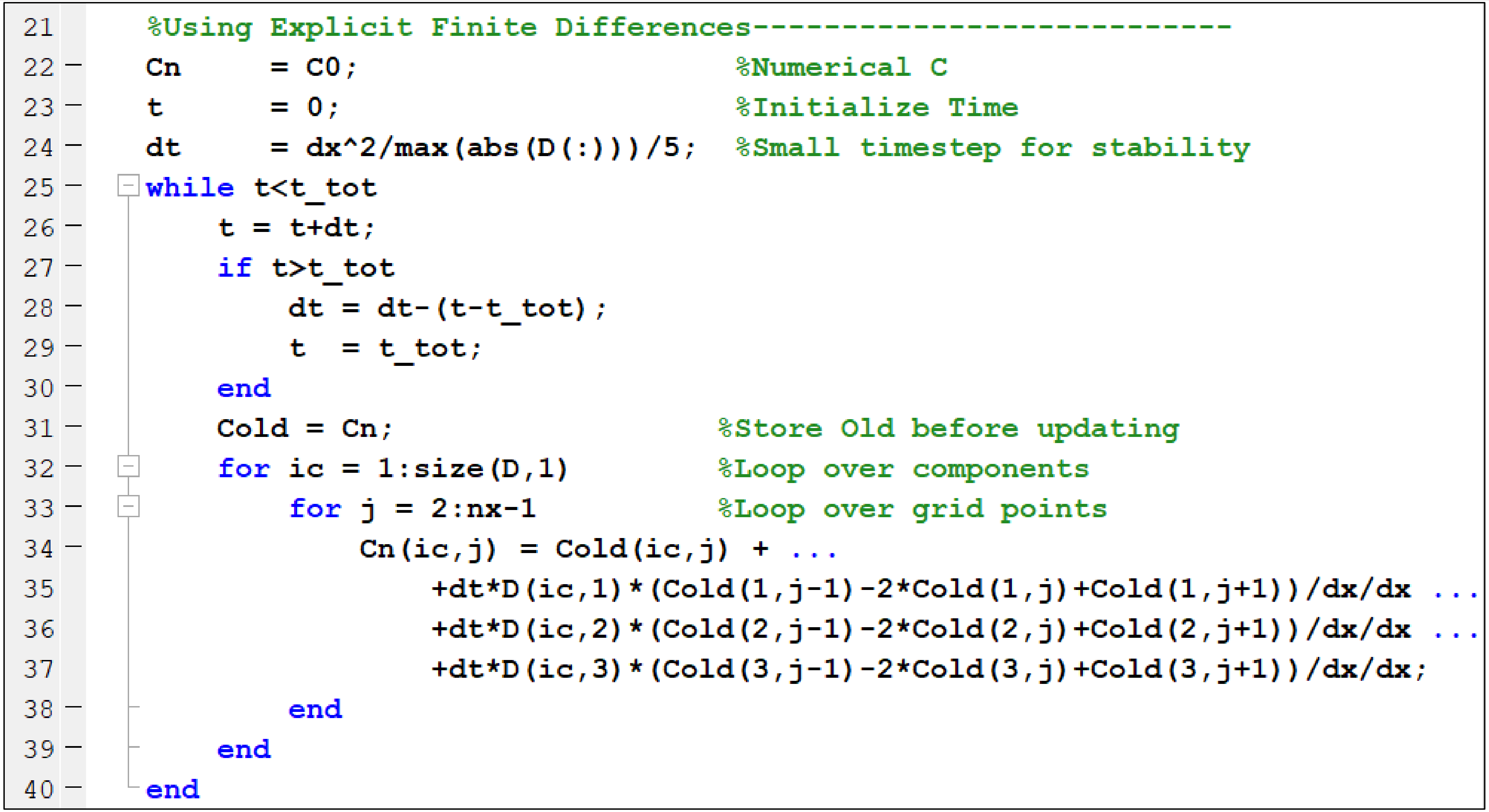
Multicomponent Diffusion

The solution should look like the plot on the right (dashed lines are initial profiles).

The existence of analytical solutions allows the testing of numerical codes.



Multicomponent Diffusion



```matlab
%Using Explicit Finite Differences---------------------------
Cn     = C0;                     %Numerical C
t      = 0;                      %Initialize Time
dt     = dx^2/max(abs(D(:)))/5;  %Small timestep for stability
while t<t_tot
    t = t+dt;
    if t>t_tot
        dt = dt-(t-t_tot);
        t  = t_tot;
    end
    Cold = Cn;                  %Store Old before updating
    for ic = 1:size(D,1)        %Loop over components
        for j = 2:nx-1          %Loop over grid points
            Cn(ic,j) = Cold(ic,j) + ...
                +dt*D(ic,1)*(Cold(1,j-1)-2*Cold(1,j)+Cold(1,j+1))/dx/dx ...
                +dt*D(ic,2)*(Cold(2,j-1)-2*Cold(2,j)+Cold(2,j+1))/dx/dx ...
                +dt*D(ic,3)*(Cold(3,j-1)-2*Cold(3,j)+Cold(3,j+1))/dx/dx;
        end
    end
end
```

Multicomponent Diffusion

Multicomponent Diffusion with the Finite-Element-Method

Multicomponent Diffusion

The main change in multicomponent-FEM is to consider multiple degrees of freedom in every point in space. For two independent components, the multicomponent diffusion equations read:

$$\frac{\partial}{\partial t}\begin{Bmatrix} C_1 \\ C_2 \end{Bmatrix} = \frac{\partial}{\partial x}\left(\begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix}\frac{\partial}{\partial x}\begin{Bmatrix} C_1 \\ C_2 \end{Bmatrix}\right)$$

this can be written as:

$$\frac{\partial}{\partial t}\begin{Bmatrix} C_1 \\ C_2 \end{Bmatrix} = -\begin{bmatrix} \frac{\partial}{\partial x}\left(\underbrace{-D_{11}\frac{\partial C_1}{\partial x}}_{J_{11}}\right) + \frac{\partial}{\partial x}\left(\underbrace{-D_{12}\frac{\partial C_2}{\partial x}}_{J_{12}}\right) \\ \frac{\partial}{\partial x}\left(\underbrace{-D_{21}\frac{\partial C_1}{\partial x}}_{J_{21}}\right) + \frac{\partial}{\partial x}\left(\underbrace{-D_{22}\frac{\partial C_2}{\partial x}}_{J_{22}}\right) \end{bmatrix}$$

Multicomponent Diffusion

$$\frac{\partial}{\partial t}\begin{Bmatrix}C_1\\ C_2\end{Bmatrix} = -\begin{bmatrix}\frac{\partial}{\partial x}\left(-\underbrace{D_{11}\frac{\partial C_1}{\partial x}}_{J_{11}}\right)+\frac{\partial}{\partial x}\left(-\underbrace{D_{12}\frac{\partial C_2}{\partial x}}_{J_{12}}\right)\\ \frac{\partial}{\partial x}\left(-\underbrace{D_{21}\frac{\partial C_1}{\partial x}}_{J_{21}}\right)+\frac{\partial}{\partial x}\left(-\underbrace{D_{22}\frac{\partial C_2}{\partial x}}_{J_{22}}\right)\end{bmatrix} = \left(\underbrace{\begin{bmatrix}\frac{\partial}{\partial x} & \frac{\partial}{\partial x} & & \\ & & \frac{\partial}{\partial x} & \frac{\partial}{\partial x}\end{bmatrix}}_{B_M^L}\begin{Bmatrix}J_{11}\\ J_{12}\\ J_{21}\\ J_{22}\end{Bmatrix}\right)$$

with the fluxes being:

$$\begin{Bmatrix}J_{11}\\ J_{12}\\ J_{21}\\ J_{22}\end{Bmatrix} = -\underbrace{\begin{bmatrix}D_{11} & & & \\ & D_{12} & & \\ & & D_{21} & \\ & & & D_{22}\end{bmatrix}}_{D_M}\underbrace{\begin{bmatrix}\frac{\partial}{\partial x} & \\ & \frac{\partial}{\partial x}\\ \frac{\partial}{\partial x} & \\ & \frac{\partial}{\partial x}\end{bmatrix}}_{B_M^R}\begin{Bmatrix}C_1\\ C_2\end{Bmatrix}$$

Multicomponent Diffusion

In operator notation, the previous form is written as:

$$\frac{\partial}{\partial t}\{\boldsymbol{C^n}\} - \boldsymbol{B_M^L D_M B_M^R}\{\boldsymbol{C^n}\} = \boldsymbol{0}$$

which after time discretization becomes:

$$\frac{\{\boldsymbol{C^n}\} - \{\boldsymbol{C^o}\}}{\Delta t} - \boldsymbol{B_M^L D_M B_M^R}\{\boldsymbol{C^n}\} = \boldsymbol{0}$$

Multicomponent Diffusion

Finally, the solution is given by
(for 1 shape function, please extend):

$$\left[\frac{1}{\Delta t}\boldsymbol{M_L} + \boldsymbol{K_L}\right]\begin{Bmatrix}\boldsymbol{C_{1_i}^n}\\ \boldsymbol{C_{2_i}^n}\end{Bmatrix} = \frac{1}{\Delta t}\boldsymbol{M_L}\begin{Bmatrix}\boldsymbol{C_{1_i}^o}\\ \boldsymbol{C_{2_i}^o}\end{Bmatrix}$$

where:

$$\boldsymbol{M_L} = \int_{x_1}^{x_2} \boldsymbol{M}^T \boldsymbol{M}\, dx = \int_{x_1}^{x_2} \begin{bmatrix} N_i & \\ & N_i \end{bmatrix} \begin{bmatrix} N_i & \\ & N_i \end{bmatrix} dx$$

$$\boldsymbol{K_L} = \int_{x_1}^{x_2} \underbrace{\begin{bmatrix} \frac{\partial N_i}{\partial x} & \frac{\partial N_i}{\partial x} & & \\ & & \frac{\partial N_i}{\partial x} & \frac{\partial N_i}{\partial x} \end{bmatrix}}_{\boldsymbol{B_M^L}} \underbrace{\begin{bmatrix} D_{11} & & & \\ & D_{12} & & \\ & & D_{21} & \\ & & & D_{22} \end{bmatrix}}_{\boldsymbol{D_M}} \underbrace{\begin{bmatrix} \frac{\partial N_i}{\partial x} & \\ & \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial x} & \\ & \frac{\partial N_i}{\partial x} \end{bmatrix}}_{\boldsymbol{B_M^R}} dx$$

Multicomponent Diffusion

To construct the previous matrices we had to multiply by the shape functions and integrate the result. After integration by parts on the diffusion term, using 2 linear shape functions and 2 degrees of freedom yields:

$$\frac{\boldsymbol{M_L}}{\Delta t} = \frac{1}{\Delta t}\begin{bmatrix} dx/3 & & dx/6 & \\ & dx/3 & & dx/6 \\ dx/6 & & dx/3 & \\ & dx/6 & & dx/3 \end{bmatrix}$$

$$\boldsymbol{K_L} = \frac{1}{\Delta x}\begin{bmatrix} \boldsymbol{D}_{ij} & -\boldsymbol{D}_{ij} \\ -\boldsymbol{D}_{ij} & \boldsymbol{D}_{ij} \end{bmatrix}$$

where $\boldsymbol{D}_{ij}$ represents the whole diffusivity matrix.

Multicomponent Diffusion

A more compact form is the following (without using fluxes)

$$\frac{\partial}{\partial t}\begin{Bmatrix} {C_1}_i \\ {C_2}_i \\ {C_1}_{i+1} \\ {C_2}_{i+1} \end{Bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial}{\partial x} & \\ & \frac{\partial}{\partial x} \\ \frac{\partial}{\partial x} & \\ & \frac{\partial}{\partial x} \end{bmatrix}}_{\boldsymbol{B}^{\boldsymbol{T}}} \underbrace{\begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix}}_{\boldsymbol{D}_{\boldsymbol{ij}}} \underbrace{\begin{bmatrix} \frac{\partial}{\partial x} & & \frac{\partial}{\partial x} & \\ & \frac{\partial}{\partial x} & & \frac{\partial}{\partial x} \end{bmatrix}}_{\boldsymbol{B}} \begin{Bmatrix} {C_1}_i \\ {C_2}_i \\ {C_1}_{i+1} \\ {C_2}_{i+1} \end{Bmatrix}$$

Multicomponent Diffusion

Note that the previous form gives: $$\left[\frac{1}{\Delta t}\boldsymbol{M_L} + \boldsymbol{K_L}\right]\begin{Bmatrix}\boldsymbol{C^n_{1_i}} \\ \boldsymbol{C^n_{2_i}}\end{Bmatrix} = \frac{1}{\Delta t}\boldsymbol{M_L}\begin{Bmatrix}\boldsymbol{C^o_{1_i}} \\ \boldsymbol{C^o_{2_i}}\end{Bmatrix}$$

$$\boldsymbol{M_L} = \int_{x_1}^{x_2} \boldsymbol{M}^T\boldsymbol{M}\,dx = \int_{x_1}^{x_2}\begin{bmatrix}N_i & \\ & N_i\end{bmatrix}\begin{bmatrix}N_i & \\ & N_i\end{bmatrix}dx$$

$$\boldsymbol{K_L} = \int_{x_1}^{x_2}\begin{bmatrix}\frac{\partial N_i}{\partial x} & \\ & \frac{\partial N_i}{\partial x}\end{bmatrix}\begin{bmatrix}D_{11} & D_{12} \\ D_{21} & D_{22}\end{bmatrix}\begin{bmatrix}\frac{\partial N_i}{\partial x} & \\ & \frac{\partial N_i}{\partial x}\end{bmatrix}dx$$

Following this form, you can extend this for two shape functions as in previous slide.

Multicomponent Diffusion

Exercise 14 (Advanced)

Solve the multicomponent problem by using the FEM method.

Nonlinear Diffusion

Nonlinear, concentration-dependent diffusion (optional)

Nonlinear Diffusion

So far we have seen cases where the diffusion coefficient (D) was independent of concentration. In many cases (e.g. Fe-Mg diffusion olivine), the diffusion coefficient depends on concentration (also in orientation). For example, Dohmen & Chakraborty (2007) provide the following form:

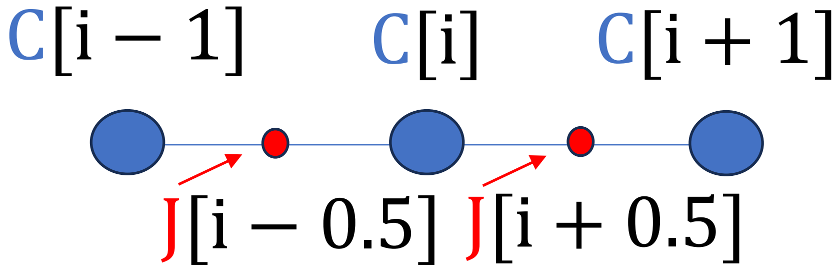
$$\log D\ (\mathrm{m}^2/\mathrm{s}) = -8.27 - \frac{226{,}000 + (P - 10^5) \cdot 7 \cdot 10^{-6}}{2.303 \cdot R \cdot T} + 3 \cdot (X_{Fe} - 0.14)$$

For constant $P - T$, diffusivity is a function of concentration (X_{Fe}). In this case, we cannot solve the classic diffusion equation, but we must solve the following:

$$\frac{\partial C}{\partial t} = \frac{\partial}{\partial x}\left(D(C)\frac{\partial C}{\partial x}\right)$$

Nonlinear Diffusion

The pressure and temperature dependence can be solved as in the case for Geospeedometry, and will not be repeated here. However, for the concentration-dependent part, we have to consider that concentration changes in space. Using the FDM, we will assume that the chemical flux is calculated at the midpoints of the grid (red points):



$\mathrm{C}[\mathrm{i}-1]$ $\mathrm{C}[\mathrm{i}]$ $\mathrm{C}[\mathrm{i}+1]$

$\mathrm{J}[\mathrm{i}-0.5]$ $\mathrm{J}[\mathrm{i}+0.5]$

Then, the diffusion equation can be discretized as (explicit method):

$$\frac{C_i^{new} - C_i^{old}}{\Delta t} = -\frac{1}{\Delta x}\left(J_{i+0.5} - J_{i-0.5}\right), \qquad \text{with} \quad J_{i+0.5} = -D\left(C_{i+0.5}^{old}\right)\frac{C_{i+1}^{old} - C_i^{old}}{\Delta x}$$

Nonlinear Diffusion

The procedure for extracting the coefficients is as before (e.g. slides 42 to 49). The main difference however is that in the implicit method **one needs to iterate to find the accurate solutions**. This is because the concentration is needed to calculate the diffusion coefficient.

Similarly, in the FEM method one needs to rederive the weak-form of the original problem by taking into account $D(C)$ in the integration procedure.

Concluding remarks

In this course we have seen some basic aspects of diffusion modelling with particular emphasis in petrologic/geochemical applications. I have tried to incorporate as many examples as possible while keeping the respective codes simple and short. Of course, many of such codes can be optimized further but numerical optimization was not the target in this course. For more advanced implementations using FEM and FDM examples, the interested reader is welcome to download and use KADMOS and GDIFF software packages that were created by the author.

- Moulas E., Brandon M.T. (2022) KADMOS: a Finite Element code for the calculation of apparent K-Ar ages in minerals. Zenodo. doi: 10.5281/zenodo.7358136
- Moulas, E., (2023) GDIFF: a Finite Difference code for the calculation of multicomponent diffusion in garnet. Zenodo. doi:10.5281/zenodo.7805989

References

- Balluffi, R.W., Allen, S.M. & Carter, W.C., (2005). *Kinetis of Materials*. Wiley-Interscience, 645 p.
- Burg, J.-P., & Moulas, E., (2022). Cooling-rate constraints from metapelites across two inverted metamorphic sequences of the Alpine-Himalayan belt; evidence for viscous heating. J. Struct. Geol., 156; https://doi.org/10.1016/j.jsg.2022.104536
- Brady, J. B., & Cherniak, D.J., (2010). Diffusion in Minerals: An Overview of Published Experimental Diffusion Data. Rev. Mineral. Geochem. 72, 899-920; https://doi.org/10.2138/rmg.2010.72.20
- Chakraborty, S., & Ganguly, J. (1992). Cation diffusion in aluminosilicate garnets: experimental determination in spessartine-almandine diffusion couples, evaluation of effective binary diffusion coefficients, and applications. Contrib. Mineral. Petrol. 111, 74–86; https://doi.org/10.1007/BF00296579
- Chakraborty, S., (1994). Relationships between Thermodynamic Mixing and Diffusive Transport in Multicomponent Solutions: Some Constraints and Potential Applications. J. Phys. Chem. 98, 4923 4926; https://doi.org/10.1021/j100069a026
- Cherniak, D.J., & Ryerson, F.J. (1993). A study of strontium diffusion in apatite using Rutherford backscattering spectroscopy and ion implantation. Geochim. Cosmochim. Acta. 125, 4653-4662; https://doi.org/10.1016/0016-7037(93)90190-8
- Crank, J., (1956), The mathematics of diffusion. Oxford University Press, 347 p.

References

- Cygan, R., & Lasaga, A. (1985). Self-diffusion of magnesium in garnet at 750 degrees to 900 degrees C. Am. J. Sci., 285 (4), 328-350; https://doi.org/10.2475/ajs.285.4.328
- Dohmen, R., & Chakraborty, S. (2007). Fe–Mg diffusion in olivine II: point defect chemistry, change of diffusion mechanisms and a model for calculation of diffusion coefficients in natural olivine. ERRATUM. Phys. Chem. Minerals, 34, 597-598; https://doi.org/10.1007/s00269-007-0185-3
- Lasaga, A. C., (1983), Geospeedometry: An extension to geothermometry, in Saxena, S. K. ed., Kinetics and Equilibrium in Mineral Reactions: Springer New York, p. 81–114.
- Moulas, E., Brandon, M., Bachmayr, M. *et al.* (in prep). The uncertainty principle of diffusion chronometry.
- Schwinger, S., Dohmen, R. & Schertl, H.-P. (2016). A combined diffusion and thermal modeling approach to determine peak temperatures of thermal metamorphism experienced by meteorites. Geochim. Cosmochim. Acta, 191, 255-276; http://dx.doi.org/10.1016/j.gca.2016.06.015
- Tajčmanová, L., Podladchikov, Y., Moulas, E. *et al.* (2021). The choice of a thermodynamic formulation dramatically affects modelled chemical zoning in minerals. *Sci. Rep.* 11, 18740; https://doi.org/10.1038/s41598-021-97568-x
- Toor, H.L. (1964). Solution of the Linearized Equations of Multicomponent Mass Transfer: 11. Matrix Methods. A.I.Ch.E. Journal, 10 (4), 463-465

References

- Turcotte, D., & Schubert, G. (2014). *Geodynamics* (3rd ed.). Cambridge University Press. doi:10.1017/CBO9780511843877
- Zhang, Y. (2008). *Geochemical Kinetics*. Princeton University Press. p.631