DRAFT! Advection, Diffusion & Reaction Modeling

Marc Los Huertos

February 19, 2024 (ver. 0.69 working toward v .8)

The movement of compounds in the environment is driven by two processes, advection and diffusion. The compounds are also subject to transformations or reactions. Thus, to monitor the fate and transport of compounds in the environment, we can capitalize on mathematical models that have been used to describe advection, diffusion, and reaction.

Introduction

Fate and Transport Processes

The fate and transport of compounds in the environment is subject to an array of processes. For example, pollutants might carried by the wind (advection) and spread out (dispersion or diffusion). In addition, the pollutant might be transformed into different compounds (reaction). Together, these three processes, advection, diffusion, and reaction (Figure 1).

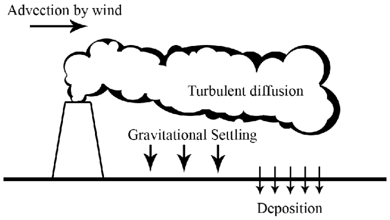


Figure 1: A simple diagram of advection and diffusion that inclues how "solutes" might be deposited downwind of a stationary source if air pollution. At the scale analyzed there, the term turbulant diffusion is different than molecular diffusion, but might be modelled in a similar way.

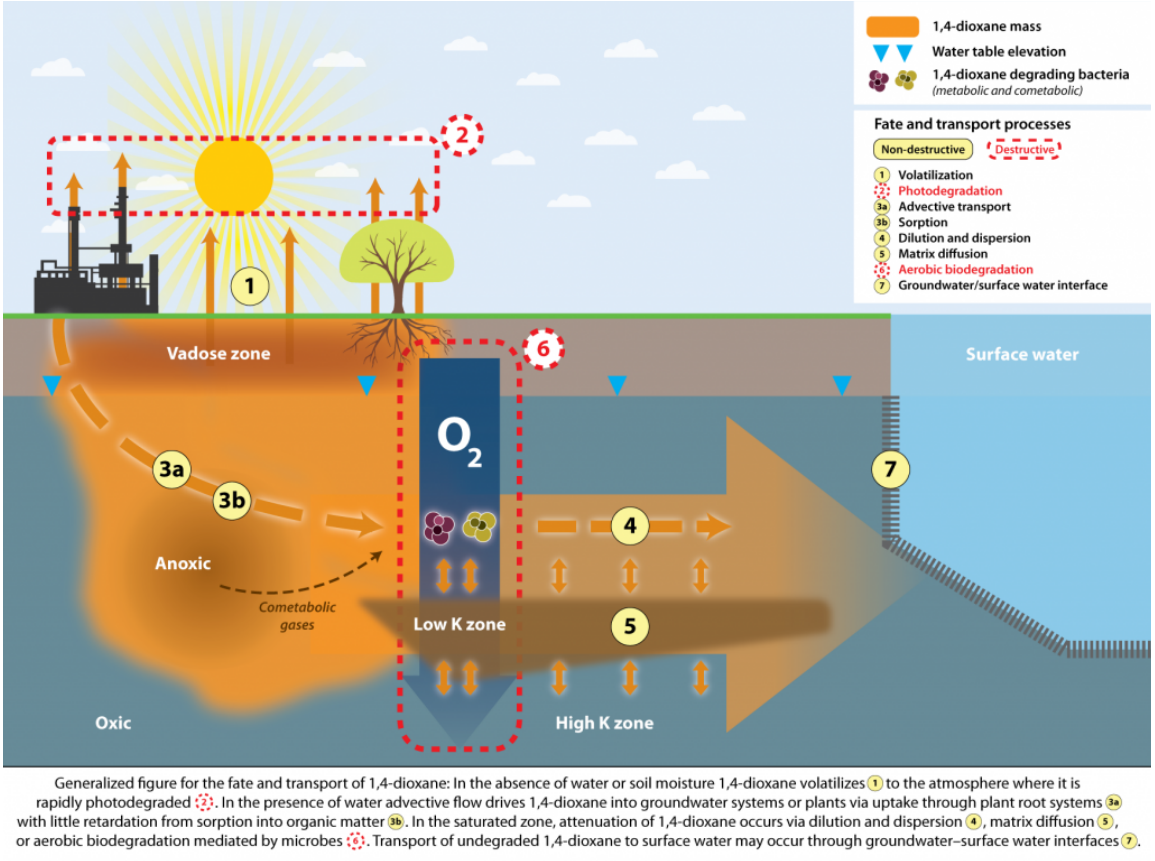
These three processes have profound implications – they provide the a framework to understand and quantify the fate and transport of pollutants in the environment (Figure 2), the movement of nutrients in the soil, and the movement of solutes in the human body. By understanding of these processes, we have tools to characterize environmental quality and its implications on human and non-human populations. Moreover, by modeling these processes, we can develop more effective policy, regulation, and mitigation strategies.

Because these processes are complex and are often difficult to measure, we rely on models to help us understand the movement of solutes in the environment. These models are based on the fundamental mathematical equations to describe advection, diffusion, and reaction.

The Processes

Advection and diffusion govern the transport of solutes in the environment. Advection is the transport of a solute by the bulk motion of the fluid. Advection depends on velocity or v in this session. Diffusion is the transport of a solute by random molecular motion. Diffusion depends on the diffusion coefficient and concentration gradient or $\frac{\partial^2 C}{\partial x^2}$ in this session. Notice that this is the second derivative.

Reaction is the transformation of a solute into a different compound. Reaction depends on the reaction rate or k in this session.



Generalized figure for the fate and transport of 1,4-dioxane: In the absence of water or soil moisture 1,4-dioxane volatilizes (1) to the atmosphere where it is rapidly photodegraded (2). In the presence of water advective flow drives 1,4-dioxane into groundwater systems or plants via uptake through plant root systems (3a) with little retardation from sorption into organic matter (3b). In the saturated zone, attenuation of 1,4-dioxane occurs via dilution and dispersion (4), matrix diffusion (5), or aerobic biodegradation mediated by microbes (6). Transport of undegraded 1,4-dioxane to surface water may occur through groundwater–surface water interfaces (7).

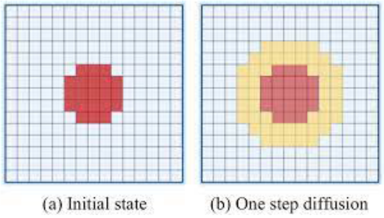
Figure 2: A diagram of the major processes that influence the fate and transport of a dioxane plume from a ce in the environment (Source: `https://14d-1.itrcweb.org/environmental-fate-transport-and-investigative-str`

1,4-Dioxane is often referred to as a "forever" compound.

In a simple example, if we have a dose of a solute, like a dye in the center of some media, we'll see it spread out by diffusion (Figure 3). If the media is moving, we'll see the dye move with the bulk motion of the media. If the dye reacts with the media, we'll see the dye disappear.

Besides examples we often see with respect to air pollution sources (Figure 1), we might think about solutes as pollutants or nutrients. For example, the movement of a nutrient in a river is driven by the bulk motion of the water (advection) and the random motion of the molecules (diffusion) and the reactions that might occur in the water column and sediments (Figure4).

Figure 3: A simple diagram of 2D diffusion. To solve these equations, we'll use a numerical approach and descretize the media into a grid. We'll then solve the equations for each grid cell.



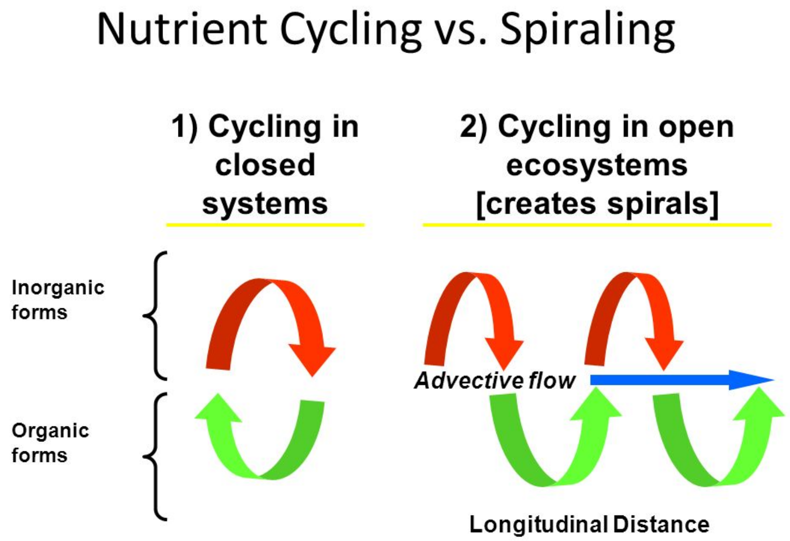

Nutrient Cycling vs. Spiraling
1) Cycling in closed systems
2) Cycling in open ecosystems [creates spirals]
Inorganic forms
Organic forms
Advective flow
Longitudinal Distance

Figure 4: A simple diagram of advection and nutrient reactions (organic substances and inorganic substances) in a river. Although not shown, you might also think about how diffusion might influence the movement of nutrients in the river and in the sediments. The zone where water moves into the sediment bed is called the hyperreic zone. The porosity of the sediments will allow more advective flow that might influence the reaction capacity of solutes in the sediments.

Session Goals

We will not become experts in advection-diffusion-reaction modeling, but we will become familiar with the processes and the equations that describe them. Moreover, we'll see a bit more about how R can be used to model these processes. After this session, I hope you can do the following:

1. Describe the physical processes of advection and diffusion and solute reaction
2. Describe the equations used to model A-D-R.
3. Analyze 1-dimensional movement using advection equations in R.
4. Describe diffusion mathematically
5. Analyze 1-dimensional adveecton-diffusion using R.

6. Appreciate how two-dimensional analysis of advection-diffusion can be modeled in R.

In this session, we'll want to think about the movement of solutes in the porous media, i.e. a soil with air space, sediments with water between the particles. We will refer to the porousity as ξ, which is a proportion between 0 and 1.

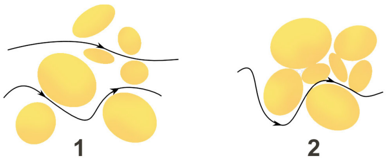


Figure 5: Notice how the porosity of the media can influence the path of the fluid. In ground water, this is measured as permeability and can be used to evaluate the flow chacterstitics in aquifers and oil fields. The permeability is a function of the porosity and the connectivity of the pores.

The Processes and the Equations to Describe Them

Advection and Convection: Material and Heat

Advection is the transport of a substance by bulk motion. Convection is the transfer of heat by the actual movement of the warmed matter. The equation that are used to describe advection and convection are similar, but the physical processes are different.

An Equation that Often Creates Anxiety

The advection equation is a partial differential equation that describes the movement of a substance in a fluid. The equation is derived from the conservation of mass. The equation is:

$$-\frac{1}{A_x\xi_x}\cdot\left(\frac{\partial}{\partial x}A_x\cdot\left(-D\cdot\frac{\partial\xi_x C}{\partial x}\right)-\frac{\partial}{\partial x}\left(A_x\cdot v\cdot\xi_x C\right)\right) \tag{1}$$

$$\frac{\partial C}{\partial t}=D_x\frac{partial^2 C}{partial x^2}+D_y\frac{partial^2 C}{partial y^2}+D_z\frac{partial^2 C}{partial z^2}-v_x\frac{\partial C}{\partial x}-\lambda RC \tag{2}$$

Here D is the "diffusion coefficient", ν is the "advection rate" (or velocity), and A_x and ξ are the surface area and volume fraction, respectively.[1]

Assuming that A, ξ, D and v are constant along x, we can rewrite this in a more general form:

$$\frac{\partial C/\partial t}{=}D\frac{\partial^2 C}{\partial x^2}-\nu\frac{\partial C}{\partial x}-R \tag{3}$$

where $\nu = v/A_x\xi_x$ is the "velocity" of the fluid.

The movement of compounds in the environment is driven by two processes, advection and diffusion. Of course, these processes occur in three dimensions, but for this class we'll begin with one dimensional processes before getting to more complicated examples.

Nevertheless, let's look at the 3-D advection-diffusion-reacton equation in three dimensions:

$$\frac{\partial C}{\partial t}=\nabla\cdot(D\nabla C-\nu C)+R \tag{4}$$

[1] What you look at this equation, think about how it makes you feel. It's a bit intimidating, to say the least. But that's only part of the story. I believe these equations, when put in front of is generates anxiety – and this anxiety can be a barrier to learning. While it's easy to claim we can just put this anxiety away seems to be be a disservice and acknoweldgement of our emotional responses. Before moving forward, let's take a moment to acknowledge the anxiety and see where in your body you are feeling that. Take a deep breath and let it out several times before moving forward.

where C is the concentration of the solute, t is time, v is the velocity of the fluid, D is the diffusion coefficient, and R is the reaction term.

Ok, what what is ∇? It's the gradient operator, which is a vector operator that operates on a scalar function to produce a vector whose magnitude is the maximum rate of change of the function at the point of the gradient and that points in the direction of that maximum rate of change. $\nabla\cdot$ represents divergence. In this equation, ∇C represents concentration gradient.

Advection

Advection is the process of transport of a solute by the bulk motion of the fluid. The rate of advection is proportional to the velocity of the fluid and the concentration of the solute. The rate of advection is given by the equation:

$$\frac{\partial C}{\partial t} + \nabla \cdot (vC) = 0 \tag{5}$$

where C is the concentration of the solute, t is time, and v is the velocity of the fluid.

For one dimensional systems, the equation can be written as:

$$\frac{\partial C}{\partial t} + v\frac{\partial C}{\partial x} = 0 \tag{6}$$

or

$$\frac{\partial C}{\partial t} + v_x\frac{\partial C}{\partial x} \tag{7}$$

where C is the concentration of the solute, t is time, v is the velocity of the fluid, and x is the spatial coordinate.[2]

The advection equation is not simple to solve numerically: the system is a hyperbolic partial differential equation, and interest typically centers on discontinuous "shock" solutions (which are notoriously difficult for numerical schemes to handle).

T

Diffusion

Diffusion is the process of transport of a solute by random molecular motion. The rate of diffusion is proportional to the concentration gradient of the solute. The rate of diffusion is given by the equation:

$$\frac{\partial C}{\partial t} = D\nabla^2 C \tag{8}$$

where D is the diffusion coefficient.

[2] This can also be solved as a flux density – the amount of solute that crosses a unit area per unit time, $\mathbf{j}_{\text{adv}} = \mathbf{v}c$.

Advection-Diffusion Equation

The advection-diffusion equation is a combination of the advection and diffusion equations. The advection-diffusion equation is given by the equation:

$$\frac{\partial C}{\partial t} + \nabla \cdot (\nu C) = D \nabla^2 C \tag{9}$$

or

$$\frac{\partial C}{\partial t} + u \frac{\partial C}{\partial x} = D \frac{\partial^2 C}{\partial x^2} \tag{10}$$

where C is the concentration of the solute, t is time, ν is the velocity of the fluid, and D is the diffusion coefficient.

Advection-Difffusion-Reaction Equation

The advection-diffusion-reaction equation is a combination the advection, diffusion, and reaction equations. The advection-diffusion-reaction equation is given by the equation:

$$\frac{\partial C}{\partial t} + \nabla \cdot (\nu C) = D \nabla^2 C + R \tag{11}$$

where C is the concentration of the solute, t is time, ν is the velocity of the fluid, D is the diffusion coefficient, and R is the reaction term.

or in one-dimension:

$$\frac{\partial C}{\partial t} = \nu \frac{\partial C}{\partial x} = D \frac{\partial^2 C}{\partial x^2} + R \tag{12}$$

where C is the concentration of the solute, t is time, ν is the velocity of the fluid, D is the diffusion coefficient, and R is the reaction term.

Advenction-Difffusion-Reaction in multi-phase systems and for shapes with variable geometry

The advection-diffusion-reaction equation can be extended to multi-phase systems and to shapes with variable geometry. The advection-diffusion-reaction equation for multi-phase systems and for shapes with variable geometry is given by the equation:

$$\frac{\partial C}{\partial t} + \nabla \cdot (\nu C) = D \nabla^2 C + R \tag{13}$$

where C is the concentration of the solute, t is time, ν is the velocity of the fluid, D is the diffusion coefficient, and R is the reaction term.

Applications using R

ReacTran Package

Soetaert et al. (2017) have developed a nice library in R that solves these equations using finite-difference solutions.

The ReacTran package is a collection of functions for modeling solute transport in 1D, 2D, and 3D. The package includes functions for solving the advection-diffusion equation, the advection-diffusion-reaction equation, and the advection-diffusion-reaction equation for multi-phase systems.

The package also includes functions for solving:

- Advection-diffusion equations in 1D, 2D, and 3D
- Advection-diffusion-reaction equations in 1D, 2D, and 3D
- Advection-diffusion-reaction equations for multi-phase systems

Using R as a Modelling Environment to Solve PDEs and ODEs

We can use R to solve the advection-diffusion equation. The 'deSolve' package to solve the ReacTran library functions decomposing the partial differential equatation (PDE) into a descretized by space and solving be ordinary differential equations (ODE).

```
library(ReacTran) # Load the ReacTran package

## Loading required package:  rootSolve
## Loading required package:  deSolve
## Loading required package:  shape

library(deSolve) # Load the deSolve package
library(xtable) # Load the xtable package to help format tables outputs
```

1D Transportion Model

The 'ReacTran' package provides a function to solve the advection-diffusion-reaction equation for a simple one-dimensional case.

```
tran.1D(C = 1, D = 0, flux.up = 1, v = 5, A= 1, dx = 1, full.output = TRUE)

## $dC
## [1] -4
##
## $C.up
## [1] 0.2
##
```

```
## $C.down
## [1] 1
##
## $dif.flux
## [1] 0 0
##
## $adv.flux
## [1] 1 5
##
## $flux
## [1] 1 5
##
## $flux.up
## [1] 1
##
## $flux.down
## [1] 5
```

Solving a 1-D reaction tranport model

```
library(ReacTran)
out <- steady.1D(func = advModel, y = runif(25), params = parms, nspace= 1, positive = TRUE)
```

We can use R to solve the advection-diffusion-reaction equation. The following code uses the ‘deSolve‘ package to solve the advection-diffusion-reaction equation for a simple one-dimensional case.

```
# Load the deSolve package
library(deSolve)

# Define the advection-diffusion-reaction equation
advection_diffusion_reaction <- function(t, C, parms) {
  with(as.list(parms), {
    dC <- D * (diff(C, lag = 2) - 2 * diff(C, lag = 1) + diff(C, lag = 0)) / dx^2 - k * C
    dC[1] <- 0
    dC[n] <- 0
    list(dC)
  })
}

# Set the parameters
parms <- list(
  D = 0.1,  # Diffusion coefficient
  dx = 0.1,  # Spatial step
  k = 0.01  # Reaction rate
)

# Set the initial conditions
C0 <- c(0, rep(0, 98), 1, rep(0, 98), 0)

# Set the times at which to evaluate the solution
times <- seq(0, 100, by = 1)

# Solve the advection-diffusion-reaction equation
out <- ode(y = C0, times = times, func = advection_diffusion_reaction, parms = parms)

# Plot the solution
plot(out, xlab = "Distance", ylab = "Concentration", type = "l")
```

1-D Reaction-Transport Model

For this example, we will solve the advection-diffusion-reaction equation for a simple one-dimensional case, where the reaction term is given by $R = kC$, and the initial concentration is given by $C(x, 0) = 1$ for $x <?$ and $C(x, 0) = 1$ for $x \geq 0.5$.

	Value
F0	1.00
v	1.00
k	0.10
D	0.00
dx	1.00

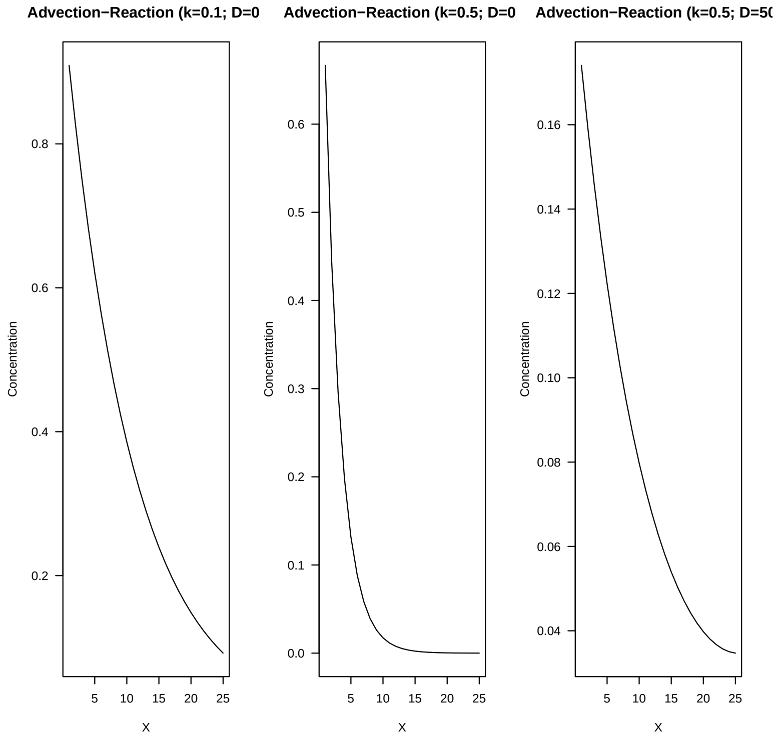
```
advModel <- function(t, C, parms) {
  with(as.list(parms), {
    Tran <- tran.1D(C = C, D = D, flux.up = F0, v = v, dx = dx)
    Consumption = k * C
    dC <- Tran$dC - Consumption

    return(list(dC = dC, Consumption = Consumption, flux.up = Tran$flux.up, flux.down = Tran$flux.down))
  })
}

out <- steady.1D(func = advModel, y = runif(25), parms = parms, nspec= 1, positive = TRUE)

parms <- c(F0 = 1, v=1, k = 0.5, D=0, dx = 1)
```

```
out2 <- steady.1D(func = advModel, y = runif(25), parms = parms, nspec= 1, positive = TRUE)

parms <- c(F0 = 1, v=1, k = 0.5, D=50, dx = 1)
out3 <- steady.1D(func = advModel, y = runif(25), parms = parms, nspec= 1, positive = TRUE)
```

We can look at the output, using a simple call of the object, but without more information, it's not clear what we are looking at.

```
out

## $y
##  [1] 0.9090909 0.8264463 0.7513148 0.6830134 0.6209213 0.5644739 0.5131581
##  [8] 0.4665074 0.4240976 0.3855433 0.3504939 0.3186308 0.2896644 0.2633313
## [15] 0.2393920 0.2176291 0.1978447 0.1798588 0.1635080 0.1486436 0.1351306
## [22] 0.1228460 0.1116782 0.1015256 0.0922960
##
## $Consumption
##  [1] 0.09090909 0.08264463 0.07513148 0.06830134 0.06209213 0.05644739
##  [7] 0.05131581 0.04665074 0.04240976 0.03855433 0.03504939 0.03186308
## [13] 0.02896644 0.02633313 0.02393920 0.02176291 0.01978447 0.01798588
## [19] 0.01635080 0.01486436 0.01351306 0.01228460 0.01116782 0.01015256
## [25] 0.00922960
##
## $flux.up
## [1] 1
##
## $flux.down
## [1] 0.092296
##
## attr(,"precis")
## [1] 2.642744e-01 2.028815e-09
## attr(,"steady")
## [1] TRUE
## attr(,"class")
## [1] "steady1D"  "rootSolve" "list"
## attr(,"dimens")
## [1] 25
## attr(,"nspec")
## [1] 1
```

Thus, we might be better off plotting the output. I am not sure why the plot functions are ignoring my par() call, perhaps this will be fixed by version 0.9!



asdf

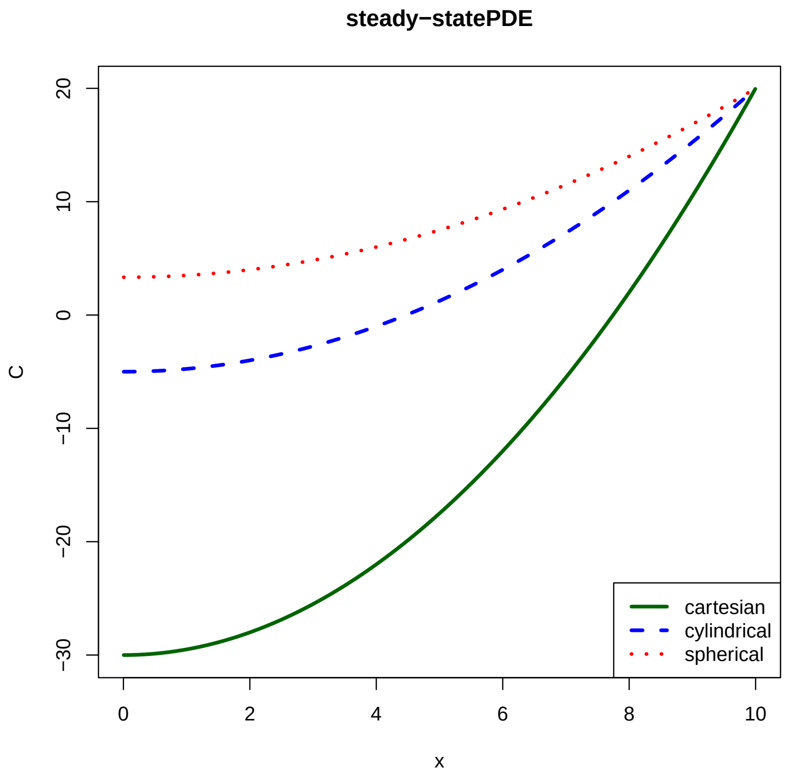
```
Grid<-setup.grid.1D(N=1000,L=10)
r <-setup.prop.1D(grid=Grid,func=function(r)r)
r2<-setup.prop.1D(grid=Grid,func=function(r)r^2)


pde1D<-function(t,C,parms,A=1){
  tran<-tran.1D(C=C,A=A,D=D,C.down=Cext, dx=Grid)$dC
  list(tran-Q)
}

D<-1
Q<-1
Cext<-20




Cartesian <-steady.1D(y=runif(Grid$N), func=pde1D,parms=NULL,nspec=1,A=1)

Cylindrical<-steady.1D(y=runif(Grid$N), func=pde1D,parms=NULL,nspec=1,A=r)
```

```
print(system.time( Spherical <-steady.1D(y=runif(Grid$N), func=pde1D,parms=NULL,nspec=1,A=r2) ))

##    user  system elapsed
##   0.001   0.000   0.002
```

user systemelapsed 0.02 0.00 0.00 Thevaluesofthestate-variables(y)areplottedagainsttheradialdistance, inthemiddleof thegridcells

(Grid$x.mid$).

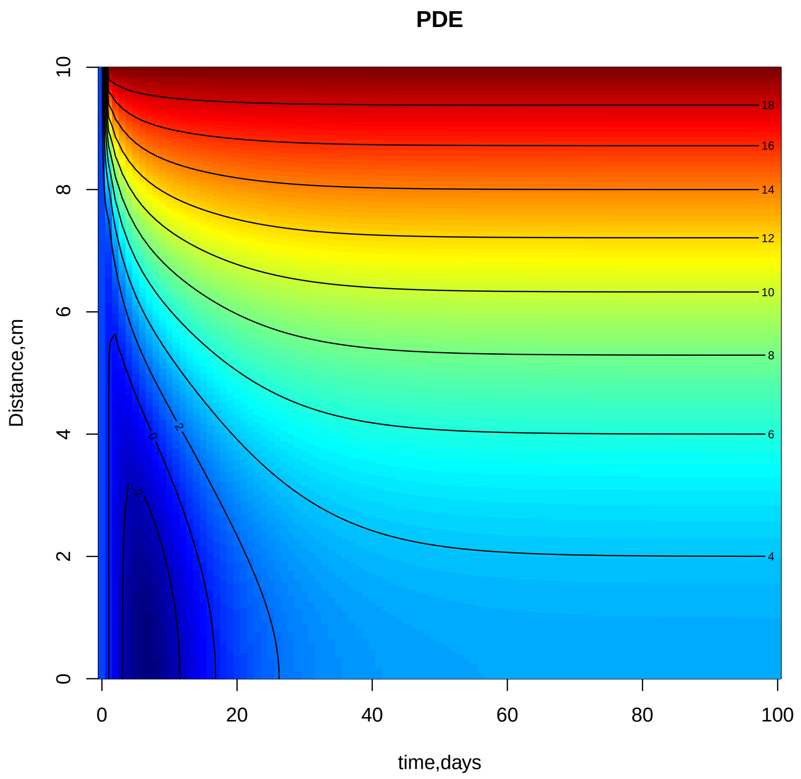
```
par(mfrow=c(1,1))
plot(Grid$x.mid,Cartesian$y,type="l",main="steady-statePDE", lwd=3,xlab="x",ylab="C",col="darkgreen",lty=1
lines(Grid$x.mid,Cylindrical$y,lwd=3,col="blue",lty=2)
lines(Grid$x.mid,Spherical$y,lwd=3,col="red",lty=3)
legend("bottomright",c("cartesian","cylindrical","spherical"), col=c("darkgreen","blue","red"),lwd=3,lty=1
```



asdfasdf

```
times<-seq(0,100,by=1)
```

```
system.time(
  out<-ode.1D(y=rep(1,Grid$N),times=times,func=pde1D, parms=NULL,nspec=1,A=r2)
  )

##    user  system elapsed 
##   0.153   0.012   0.165

tail(out[,1:4],n=3)

##         time        1        2        3
##  [99,]    98 3.332278 3.332303 3.332366
## [100,]    99 3.332383 3.332408 3.332471
## [101,]   100 3.332478 3.332503 3.332566

image(out,grid=Grid$x.mid,xlab="time,days", ylab="Distance,cm",main="PDE",add.contour=TRUE)
```



Oxygen Consumption Porous Spherical Particle

We will be modeling the consumption of oxygen in a "sand-sized" porous spherical particle. The model is based on the following equation:

$$\frac{\partial C}{\partial t} = -v\frac{\partial C}{\partial x} - k(C)$$

where C is the concentration of oxygen, D is the diffusion coefficient, v is the velocity of the fluid, and $k(C)$ is the rate of oxygen consumption.

At this scale the velocity will be zero. Thus, we will rely on diffusion to for the oxygen movement to where it is consumed.

We start with defining the size and porosity of the particle and use R to create a grid to solve the advection-diffusion-reaction equation.

Table 1: Chararacteristics of the Particle

Parameter	Description	Typical Range	Modelled Value
R	Radius of the particle	$0.005 - 0.2$ cm	0.025 cm
Porosity	Proportion of void space	$0.005 - 0.7$	0.7

We will create a grid to model the particle with Radius R and N (= 100) grid points. We will also define the properties of the particle such as porosity, diffusion coefficient (D = 400), and the rate of oxygen consumption, R_{02} = 10^6.

Although we are modeling a one-dimensional system, we will need to create a grid surface as a circle to effectively model the particle surface area changes as O2 diffuses into the particle and is consumed by the reactions in the particle.

```
grid <- setup.grid.1D(x.up=0, L = R, N = N) # Grid definition
por.grid <- setup.prop.1D(value=por, grid=grid) # Porosity
D.grid <- setup.prop.1D(value=D, grid=grid) # Diffusion coefficient

sphere.surf <- function(x) 4*pi*x^2 # Surface area of a sphere
A.grid <- setup.prop.1D(func=sphere.surf,  grid=grid) # Surface area
```

Finally, we need to define the O2 concentration at the surface of the particle and the O2 consumption rate of the particle.

Table 2: Oxygen Consumption in the Particle

Parameter	Description	Typical Range	Modelled Value
C_{ow}	Concentration of O2 in Water	$0.1 - 0.3$ μmols/cm^{-3}	0.25 μmols/cm^{-3}
R_{02}	Rate of oxygen consumption	$10^5 - 10^6$ μmols/cm^{-3} /year	10^6 μmols/cm^{-3}/year
K_s	O2 saturation	$0.001 - 0.01$ μmols/cm^{-3}	0.005 μmols/cm^{-3}

Note, we often measure oxygen using ppm (parts per million), but the model uses μmols/cm^{-3}. The conversion is 1 ppm = 0.0224 μmols/cm^{-3}, thus, we are modeling Ks within a range of 2.24 - 6.72 ppm, using Ks = 0.22 μmols/cm^{-3}.

Next we create a function to model the oxygen consumption in the, particle that relies on We will use the `tran.1D` function to solve the advection-diffusion equation and the `steady.1D` function to solve the steady state solution of the advection-diffusion-reaction equation.

```
Aggregate.Model <- function(time, O2, pars) {
 tran <- tran.1D(C = O2, C.down = C.ow.O2, D = D.grid,
     A=A.grid, VF = por.grid, dx = grid)
  reac <- - R.O2 * (O2 /(Ks + O2))
  return(list(dCdt= tran$dC + reac, reac = reac,
        flux.up=tran$flux.up, flux.down=tran$flux.down))
}


O2.agg <- steady.1D(y = runif(N), func=Aggregate.Model,
          nspec=1, positive=TRUE, atol = 1e-10)
```

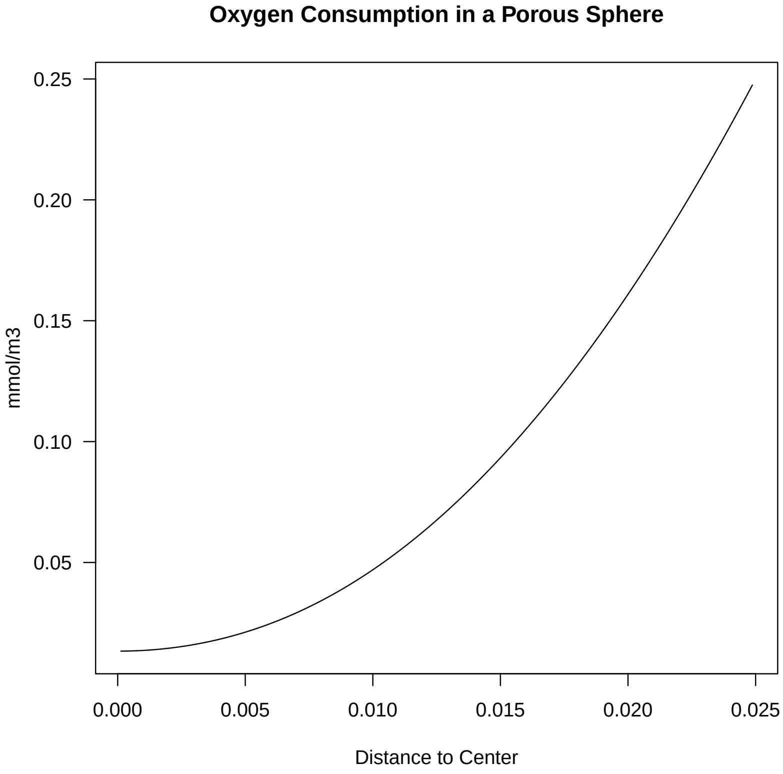
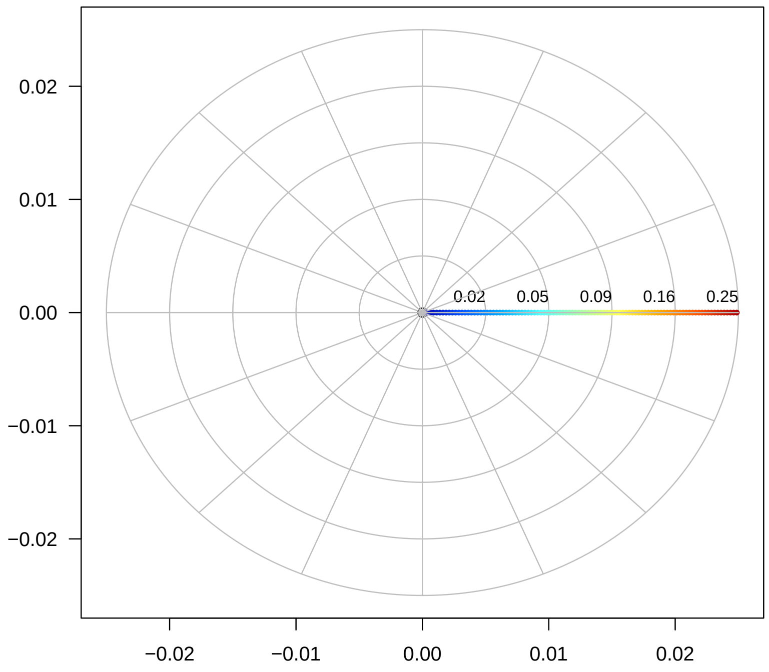


Figure 6: Oxygen Consumption in a Porous Sphere

The plot shows the oxygen concentration in the particle. The concentration is highest at the surface and decreases as it moves into the particle. The concentration is zero at the center of the particle.

```
diffusion2D <- function(t,conc,par){
Conc <- matrix(nr=n,nc=n,data=conc) # vector to 2-D matrix
dConc <- -r*Conc*Conc # consumption
```



```
BND <- rep(1,n) # boundary concentration

# constant production in certain cells
dConc[ii]<- dConc[ii]+ p

#diffusion in X-direction; boundaries=imposed concentration

Flux <- -Dx * rbind(rep(0,n),(Conc[2:n,]-Conc[1:(n-1),]),rep(0,n))/dx
dConc <- dConc - (Flux[2:(n+1),]-Flux[1:n,])/dx

#diffusion in Y-direction
Flux <- -Dy * cbind(rep(0,n),(Conc[,2:n]-Conc[,1:(n-1)]),rep(0,n))/dy
dConc <- dConc - (Flux[,2:(n+1)]-Flux[,1:n])/dy

return(list(as.vector(dConc)))
}
```

After specifying the values of the parameters, 10 cells on the 2-D grid where there will be substance produced are randomly selected (ii).

14 Package rootSolve : roots, gradients and steady-states in R 0.0 0.2 0.4 0.6 0.8 1.0 0.0 0.2 0.4 0.6 0.8 1.0 2-D diffusion+production x y Figure 5: Steady-state solution of the nonlinear 2-Dimensional model

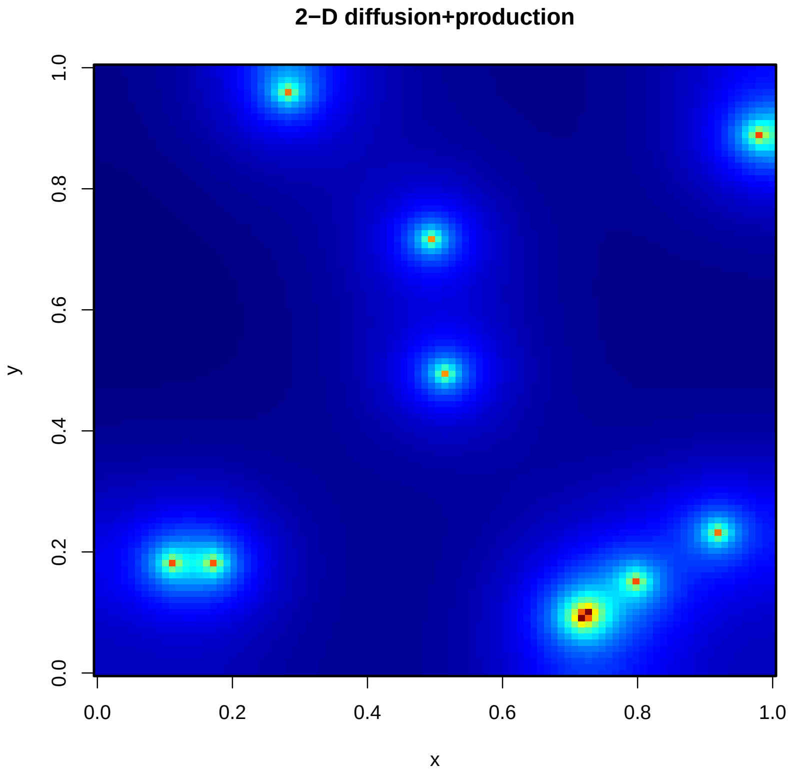
```r
# parameters
dy <- dx <- 1 # grid size
Dy <- Dx <- 1.5 # diffusion coeff, X- and Y-direction
r <- 0.01 # 2-nd-order consumption rate (/time)
p <- 20 # 0-th order production rate (CONC/t)
n <- 100
# 10 random cells where substance is produced at rate p
ii <- trunc(cbind(runif(10)*n+1,runif(10)*n+1))
```

The steady-state is found using function steady.2D. It takes as arguments a.o. the dimensionality of the problem (dimens) and lrw=1000000, the length of the work array needed by the solver. If this value is set too small, the solver will return with the size needed. It takes about 0.5 second to solve this 10000 state variable model.

```r
Conc0 <- matrix(nr=n,nc=n,10.)
print(system.time(
ST3 <- steady.2D(Conc0,func=diffusion2D,parms=NULL,pos=TRUE,dimens=c(n,n),
lrw=1000000,atol=1e-10,rtol=1e-10,ctol=1e-10)
))

##    user  system elapsed
##   0.217   0.010   0.227
```

user system elapsed 1.044 0.032 1.076 The S3 image method is used to generate the steady-state plot.

```r
image(ST3,main="2-D diffusion+production", xlab="x", ylab="y")
```



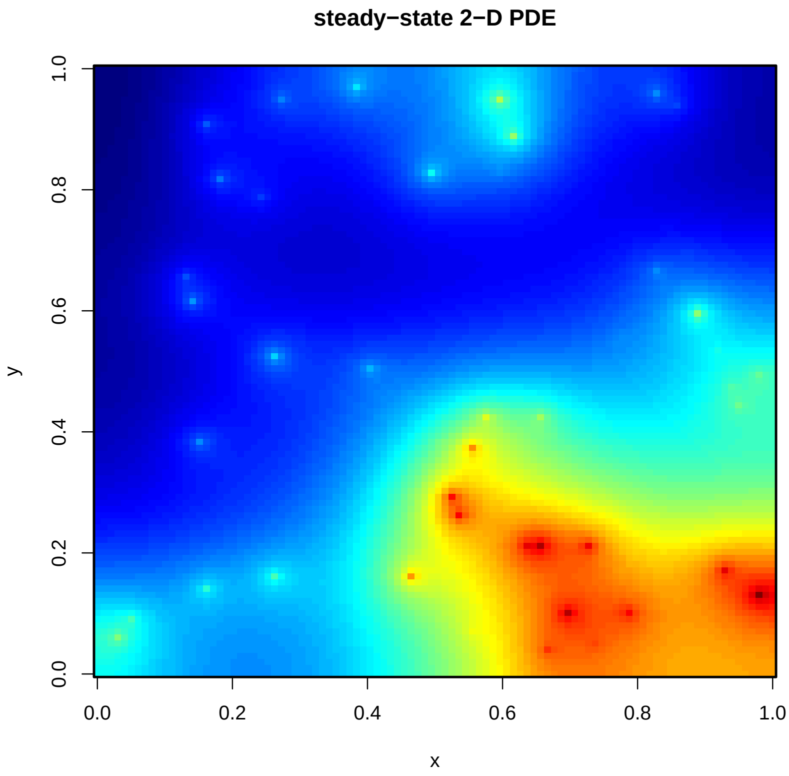
```
pde2D <- function (t, y, parms) {
CONC <- matrix(nr = n, nc = n, y)
Tran <- tran.2D(CONC, D.x = Dx, D.y = Dy, dx = dx, dy = dy)
dCONC <- Tran$dC - r * CONC
dCONC[ii]<- dCONC[ii] + p
return(list(as.vector(dCONC)))
}
```

Before running the model, the grid sizes (dx, dx), diffusion coefficients (Dx, Dy), 1st order consumption rate (r) are defined. There are 100 boxes in x- and y direction (n). Furthermore, we assume that the substance is produced in 50 randomly chosen cells (ii) at a constant rate (p):

```
n <- 100
dy <- dx <- 1
Dy <- Dx <- 2
r <- 0.001
p <- runif(50)
ii <- trunc(cbind(runif(50) * n, runif(50) * n) + 1)
```

```
Conc0 <- matrix(nr = n, nc = n, 10)
print(system.time(ST <- steady.2D(y = Conc0, func = pde2D,
parms = NULL, dimens = c(n, n), lrw = 6e+05)))

##    user  system elapsed
##   0.068   0.000   0.068
```

```
image(ST, main = "steady-state 2-D PDE")
```



Considering 2D Models

2.4. Steady-state solution of 2-D PDEs Function steady.2D efficiently finds the steady-state of 2-dimensional problems. Karline Soetaert 13 In the following model @C @t = Dx @2C @x2 + Dy @2C @y2 .. r C2 + pxy a substance C is consumed at a quadratic rate (r C2), while dispersing in X- and Y-direction. At certain positions (x,y) the substance is produced (rate p). The model is solved on a square (100*100) grid. There are zero-flux boundary conditions at the 4 boundaries. The term Dx @2C @x2 is in fact shorthand for: .. @Flux @x where Flux

= ..Dx @C @x i.e. it is the negative of the flux gradient, where the flux is due to diffusion. In the numerical approximation fo the flux, the concentration gradient is approximated as the subtraction of two matrices, with the columns or rows shifted (e.g. Conc[2:n,]-Conc[1:(n-1),]). The flux gradient is then also approximated by subtracting entire matrices (e.g. Flux[2:(n+1),]-Flux[1:(n),]). This is very fast. The zero-flux at the boundaries is imposed by binding a column or row with 0-s.

The Theory

Solving the advection-diffusion equation

The advection-diffusion equation is a partial differential equation that describes the transport of a substance by both diffusion and advection.

Goals of these notes

- Introduce the advection-diffusion equation
- Find solution for steady-state XX transfer with a constant XX flux
- Find solution for steady-state solute transfer with a constant basal??

Advection-diffusion equation in 1D

To show how the advection equation can be solved, we're actually going to look at a combination of the advection and diffusion equations applied to compound advection and diffusion.

transfer equations in 1D, with math that is straightforward to follow.

Disperson is a diffusion process caused by interactions of atoms or molecules, which can be simulated using the diffusion equation we saw in last week's notes.

Mathematically, we'll start with our two equations: (1) The diffusion equation without heat production and (2) the advection equation, then combine them.

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial z^2} \text{ Diffusion}$$

(14)

$$\frac{\partial T}{\partial t} = v_z \frac{\partial T}{\partial z} \text{ Advection}$$

(15)

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial z^2} + v_z \frac{\partial T}{\partial z} \quad \text{Diffusion + Advection} \qquad (16)$$

In steady state, we can ignore the transient term $\partial T/\partial t$, so

$$\cancel{\frac{\partial T}{\partial t}}\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial z^2} + v_z \frac{\partial T}{\partial z} \quad \text{Steady-state advection-diffusion equation}$$

(17)

$$\partial^2 T_{\overline{\partial z^2 = -\frac{v_z}{\kappa}\frac{\partial T}{\partial z} \quad \text{Rearranged}}}$$

(18)

Another way to write the previous equation is

$$\frac{\partial}{\partial z}\left(\frac{\partial T}{\partial z}\right) = -\frac{v_z}{\kappa}\frac{\partial T}{\partial z}$$

(19)

In this case, we can make some substitutions and find something quite useful. Assume $f = \partial T/\partial z$ and $c = v_z/\kappa$. With this, we can say $f'(z) = -cf(z)$.

This is a common form of differential equation with a solution $f(z) = f(0)\mathrm{e}^{cz}$. Thus, in terms of our equation we can say

$$\partial T_{\overline{\partial z = -\frac{\partial T}{\partial z}\big|_{(z=0)} \mathrm{e}^{-(v_z z/\kappa)}}}$$

(20)

Solutions to the steady-state advection-diffusion equation

The simplest solution to the previous equation is to assume a constant temperature gradient

$$\frac{\partial T}{\partial z} = -\left.\frac{\partial T}{\partial z}\right|_{(z=0)} e^{-(v_z z/\kappa)} = g e^{-(v_z z/\kappa)}$$

(21)

$\int \frac{\partial T}{\partial z} = g \int e^{-(v_z z/\kappa)}$Integrate
T(z) = -g $\kappa \frac{}{v_z e^{-(v_z z/\kappa)} + c_1}$(22)
Assume $T(0) = 0.$

T(z) = -g $\kappa \frac{}{v_z e^{-(v_z z/\kappa)} + c_1}$
(23)

$$0 = -\frac{g\kappa}{v_z} \cancelto{1}{e^{-(v_z 0/\kappa)}} + c_1$$

c$_1 = \frac{g\kappa}{v_z}$
(24)

Thus, we find

T(z) = -g $\kappa \frac{}{v_z e^{-(v_z z/\kappa)} + \frac{g\kappa}{v_z}}$
(25)

$$T(z) = \frac{g\kappa}{v_z}\left(1 - e^{-(v_z z/\kappa)}\right) \quad \text{Rearranged}$$

What should our temperature profile look like?
At constant z , what happens to T if v_z gets large?
Constant temperature T_L at $z = L$

A more useful second boundary condition is to assume $T(L) = T_L$. In this case

$$T(z) = T_L \left(\frac{1-e^{-(v_z z/\kappa)}}{1-e^{-(v_z L/\kappa)}} \right) (26)$$

Advection-diffusion equation take-home messages

- Math gets a bit more complex, even for the 'simplest' cases; Often need numerical methods for more complex geometries
- Behavior of the equation is strongly controlled by the boundary conditions
- Even these simple equations can be quite useful. Advection can be a significant influence on the thermal field and these simple calculations allow you to estimate when it is a factor and its magnitude of influence.

Some Caveats:

- Steady-state
- 1-D
- Constants assumed to be constant :)
- No heat production

Conclusion

https://en.wikipedia.org/wiki/Convection%E2%80%93diffusion_equation

References

Soetaert, K., Meysman, F., and Soetaert, M. K. (2017). Package âĂŸreactranâĂŹ.