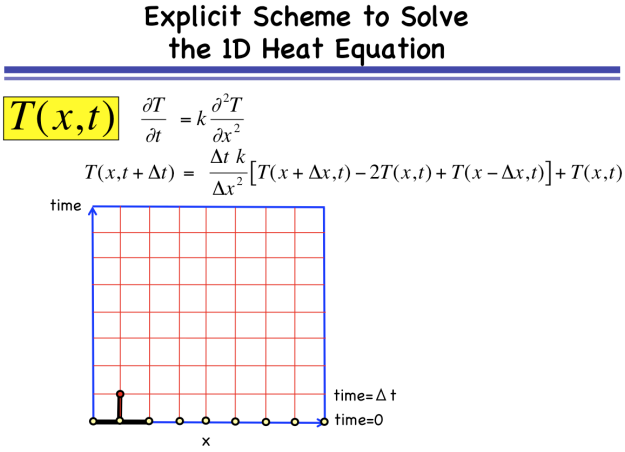
Computer Lab Assignment 8
Solving the 1D and 2D Heat Equations in Real Time

No Python code will be provided for this computer lab. Please recycle your notebooks from last week or create it one from scratch. The goal of this lab is to construct real-time solutions for 1D and 2D cooling problems. First please review the analytical and the discretized form of the 1D heat equation given in following lecture slide:



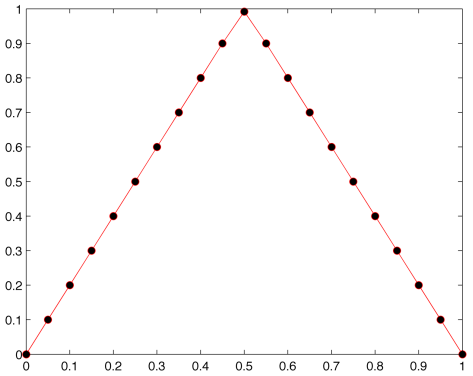
(1) The discretized form for $T(x,t+\Delta t)$ needs to be transformed yet again to index notation that Python can handle. We will use a 1D array `T[i]` to present $T(x,t)$ and `Tnew[i]` to present $T(x,t+\Delta t)$. (Alternatively one could also introduce an index j and use `T[i,j]` to represent space and time but we avoid introducing an index j to save memory.) Now fill in the blanks:

$T(x,t)$	$\rightarrow$	`T[i]`
$T(x,t+\Delta t)$	$\rightarrow$	`Tnew[i]`
$T(x+\Delta x,t)$	$\rightarrow$	*?*
$T(x-\Delta x,t)$	$\rightarrow$	*?*

and write down the resulting equation for `Tnew[i]`:

`Tnew[i]` = _____

(2) As next step, we need to prepare the initial conditions, T[i]. Discretize the interval x=[0,L] in N sections requiring N+1 points. Set L=1.0 and N=20 and reproduce the following, somewhat artificial, initial temperature distribution:



Issue a plot command that reproduces the picture above. Please make sure that *T(x=0)* and *T(x=1)* are indeed set to 0, and that *T(x=0.5)* equals 1.

(3) Set the heat conduction coefficient *k=1* and assume *T(x=0)=T(x=1)=0* as boundary conditions. Now write a Python loop that computes `Tnew[i]` from `T[i]` using the equation from step (1). Assume a time step of 10^{-4}. Then copy `Tnew` to back T and plot the temperature distribution for this first step.

(4) Now introduce an outer Python loop that repeats step (3) many times. You may not want to plot the temperature distribution at every step but when you do then please add the commands `plt.ylim(0,1)` to keep the Y axis from changing. Run your code for a large number of iterations until less than 1% of the initial thermal energy is left.

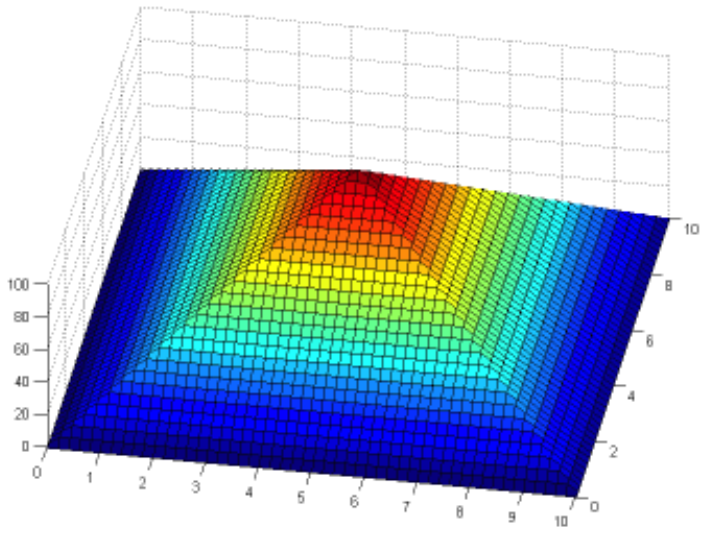
- How much time did this approximately take?
- Where did the heat go?
- Why does the heat distribution get smoother and smoother?

Well done if you see the heat disappear! You completed the most time consuming part.

(5) Now we pretend to be a bit impatient and run with a larger time steps. Please increase Δt step by step until you see a drastic change in the behavior of the resulting solution. What happens? Then carefully determine the *critical* time step, Δt_C, where this change occurs.

Solving the 2D case

(1) First we need to step up an initial temperature profile. We suggest you employ this pyramidal profile with temperature values between 0 and 100:



There are different ways to set up a pyramidal shape. Here are two options: (a) You can use `np.min( [ f1(x,y), f2(x,y), f3(x,y), f4(x,y) ] )`, where the functions f_1-f_4 represent the four faces of the pyramid. (b) You assume the pyramid is collection of nesting squares at different elevations. Both approach require a similar amount of coding.

(2) Replace the formula for `Tnew[i,j]` with the discretized 2D heat equation in real time. The following slide from lecture 13 shows the right equation in math (not Python) language. Choose whatever you consider appropriate values of k, dt, dx and dy. We recommend setting dx = dy and choosing k and dt such that $\eta = \frac{k\Delta t}{\Delta x^2} < \frac{1}{4}$.

$\frac{\partial T}{\partial t} = k\frac{\partial^2 T}{\partial x^2}$ **Compare 1D and 2D** $\frac{\partial T}{\partial t} = k\left[\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right]$

$T(x,t)$ $\quad T(x,t+\Delta t) = \frac{\Delta t\, k}{\Delta x^2}\left[T(x+\Delta x,t) - 2T(x,t) + T(x-\Delta x,t)\right] + T(x,t)$

T_i^n $\quad T_i^{n+1} = \eta\left[T_{i+1}^n - 2T_i^n + T_{i-1}^n\right] + T_i^n$ $\quad \eta = \frac{\Delta t\, k}{\Delta x^2} < \frac{1}{2}$

$T(x,y,t)$ $\quad T(x,y,t+\Delta t) = \frac{\Delta t\, k}{\Delta x^2}\left[T(x+\Delta x,y,t) - 2T(x,y,t) + T(x-\Delta x,y,t)\right] + \frac{\Delta t\, k}{\Delta y^2}\left[T(x,y+\Delta y,t) - 2T(x,y,t) + T(x,y-\Delta y,t)\right] + T(x,y,t)$

$T_{i,j}^n$ $\quad T_{i,j}^{n+1} = \eta\left[T_{i+1,j}^n + T_{i-1,j}^n + T_{i,j+1}^n + T_{i,j-1}^n - 4T_{i,j}^n\right] + T_{i,j}^n$ $\quad \eta = \frac{\Delta t\, k}{\Delta x^2} < \frac{1}{4}$

(3) Run your code and see if you see the pyramid melt away.