# pde.separation PDE solution by separation of variables

We are now ready to learn one of the most important techniques for solving PDEs: **separation of variables**. It applies only to **linear** PDEs since it will require the principle of superposition. Not all linear PDEs yield to this solution technique, but several that are important do.
The technique includes the following steps.

**assume a product solution** Assume the solution can be written as a **product solution** $u_p$: the product of functions of each independent variable.

**separate PDE** Substitute $u_p$ into the PDE and rearrange such that at least one side of the equation has functions of a single independent variabe. If this is possible, the PDE is called **separable**.

**set equal to a constant** Each side of the equation depends on different independent variables; therefore, they must each equal the same constant, often called $-\lambda$.

**repeat separation, as needed** If there are more than two independent variables, there will be an ODE in the separated variable and a PDE (with one fewer variables) in the other independent variables. Attempt to separate the PDE until only ODEs remain.

**solve each boundary value problem** Solve each boundary value problem ODE, ignoring the initial conditions for now.

**solve the time variable ODE** Solve for the general solution of the time variable ODE, sans initial conditions.

**construct the product solution** Multiply the solution in each variable to construct

the product solution $u_p$. If the boundary value problems were sturm-liouville, the product solution is a family of **eigenfunctions** from which any function can be constructed via a generalized fourier series.

**apply the initial condition** The product solutions individually usually do not meet the initial condition. However, a generalized fourier series of them nearly always does. **Superposition** tells us a linear combination of solutions to the PDE and boundary conditions is also a solution; the unique series that also satisfies the initial condition is the unique solution to the entire problem.

## Example pde.separation-1 re: 1D diffusion equation

Consider the one-dimensional diffusion equation PDE[a]

$$\partial_t u(t,x) = k\partial_{xx}^2 u(t,x) \tag{1}$$

with real constant $k$, with dirichlet boundary conditions on inverval $x \in [0, L]$

$$u(t,0) = 0 \tag{2a}$$
$$u(t,L) = 0, \tag{2b}$$

and with initial condition

$$u(0,x) = f(x), \tag{3}$$

where $f$ is some piecewise continuous function on $[0, L]$.

*a*. For more on the diffusion or heat equation, see Haberman (2018, § 2.3), Kreyszig (2011, § 12.5), and Strauss (2007, § 2.3).

*Assume a product solution*

First, we assume a product solution of the form $u_p(t,x) = T(t)X(x)$ where $T$ and $X$ are unknown functions on $t > 0$ and $x \in [0, L]$.

*Separate PDE*

Second, we substitute the product solution into Eq. 1 and separate variables:

$$T'X = kTX'' \Rightarrow \tag{4}$$

$$\frac{T'}{kT} = \frac{X''}{X}. \tag{5}$$

So it is separable! Note that we chose to group $k$ with $T$, which was arbitrary but conventional.

*Set equal to a constant*

Since these two sides depend on different independent variables ($t$ and $x$), they must equal the same constant we call $-\lambda$, so we have two ODEs:

$$\frac{T'}{kT} = -\lambda \quad \Rightarrow T' + \lambda kT = 0 \tag{6}$$

$$\frac{X''}{X} = -\lambda \quad \Rightarrow X'' + \lambda X = 0. \tag{7}$$

*Solve the boundary value problem*

The latter of these equations with the boundary conditions (2) is precisely the same sturm-liouville boundary value problem from Example pde.sturm-1, which had eigenfunctions

$$X_n(x) = \sin\left(\sqrt{\lambda_n}x\right) \tag{8a}$$

$$= \sin\left(\frac{n\pi}{L}x\right) \tag{8b}$$

with corresponding (positive) eigenvalues

$$\lambda_n = \left(\frac{n\pi}{L}\right)^2. \tag{9}$$

*Solve the time variable ODE*

The time variable ODE is homogeneous and has the familiar general solution

$$T(t) = ce^{-k\lambda t} \tag{10}$$

with real constant $c$. However, the boundary value problem restricted values of $\lambda$ to $\lambda_n$, so

$$T_n(t) = ce^{-k(n\pi/L)^2 t}. \tag{11}$$

*Construct the product solution*

The product solution is

$$\begin{aligned} u_p(t,x) &= T_n(t)X_n(x) \\ &= ce^{-k(n\pi/L)^2 t}\sin\left(\frac{n\pi}{L}x\right). \end{aligned} \tag{12}$$

This is a family of solutions that each satisfy only exotically specific initial conditions.

*Apply the initial condition*

The initial condition is $u(0,x) = f(x)$. The eigenfunctions of the boundary value problem form a fourier series that satisfies the initial condition on the interval $[0,L]$ if we extend $f$ to be periodic and odd over $x$ (Kreyszig, 2011, p. 550); we call the extension $f^*$. The odd series synthesis can be written

$$f^*(x) = \sum_{n=1}^{\infty} b_n \sin\left(\frac{n\pi}{L}x\right) \tag{13}$$

where the fourier analysis gives

$$b_n = \frac{2}{L}\int_0^L f^*(\chi)\sin\left(\frac{n\pi}{L}\chi\right). \tag{14}$$

So the complete solution is

$$u(t,x) = \sum_{n=1}^{\infty} b_n e^{-k(n\pi/L)^2 t}\sin\left(\frac{n\pi}{L}x\right). \tag{15}$$

Notice this satisfies the PDE, the boundary conditions, and the initial condition!

*Plotting solutions*

If we want to plot solutions, we need to specify an initial condition $u(0,x) = f^*(x)$ over $[0,L]$. We can choose anything piecewise continuous, but for simplicity let's let

$$f(x) = 1. \qquad (x \in [0,L])$$

The odd periodic extension is an odd square wave. The integral (14) gives

$$\begin{aligned} b_n &= \frac{4}{n\pi}(1-\cos(n\pi)) \\ &= \begin{cases} 0 & n \text{ even} \\ \frac{4}{n\pi} & n \text{ odd.} \end{cases} \end{aligned} \tag{16}$$

Now we can write the solution as

$$u(t,x) = \sum_{n=1,\, n \text{ odd}}^{\infty} \frac{4}{n\pi} e^{-k(n\pi/L)^2 t} \sin\left(\frac{n\pi}{L}x\right). \tag{17}$$

*Plotting in Python*

First, load some Python packages.

```
import numpy as np
import matplotlib.pyplot as plt
from IPython.display import display, Markdown, Latex
```

Set $k = L = 1$ and sum values for the first `N` terms of the solution.

```
L = 1
k = 1
N = 100
x = np.linspace(0,L,300)
t = np.linspace(0,2*(L/np.pi)**2,100)
u_n = np.zeros([len(t),len(x)])
for n in range(N):
  n = n+1 # because index starts at 0
  if n % 2 == 0: # even
    pass # already initialized to zeros
  else: # odd
    u_n += 4/(n*np.pi)*np.outer(
      np.exp(-k*(n*np.pi/L)**2*t),
      np.sin(n*np.pi/L*x)
    )
```

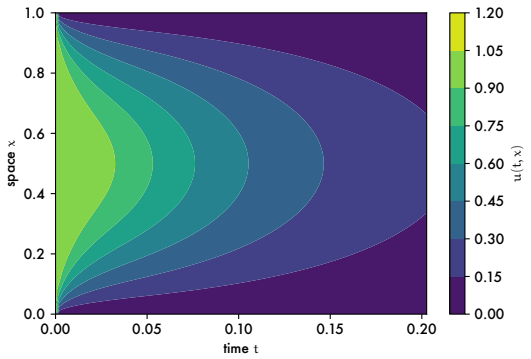
Let's first plot the initial condition.

```
p = plt.figure();
plt.plot(x,u_n[0,:]);
plt.xlabel('space $x$')
plt.ylabel('$u(0,x)$')
plt.show()
```

```
---------------------------------------------------------------------------

NameError                                 Traceback
↪  (most recent call last)

<ipython-input-1-53f11dec0549> in <module>
----> 1
↪  pxfer(draft_mode=draft_mode,fig_name=p,filename=the_notebook)


NameError: name 'pxfer' is not defined
```

Now we plot the entire response.

```
p = plt.figure();
plt.contourf(t,x,u_n.T)
c = plt.colorbar()
c.set_label('$u(t,x)')
plt.xlabel('time $t$')
plt.ylabel('space $x$')
plt.show()
```




We see the diffusive action proceeds as we expected.

. Python code in this section was generated from a Jupyter notebook named `pde_separation_example_01.ipynb` with a `python3` kernel.