Example 2 - Scipy special functions and PyFVTool

September 1, 2025

1 Example 2.

First, take an initial look at this notebook. Then, use “Kernel-> Restart & Run All” to re-evaluate the entire notebook. Running this notebook is a good test for your Python installation.

This example will use Scipy to evaluate the analytic solution for a heat transfer problem, and PyFVTool to solve the same problem by the finite-volume method.

No more details are given here. Look up more theory in the book by Crank (“Mathematics of Diffusion”).

[1]:
```python
import numpy as np
from numpy import exp
from scipy.special import jn_zeros, j0, j1
import matplotlib.pyplot as plt
import pyfvtool as pf
print('PyFVTool version', pf.__version__)
```

```
PyFVTool version 0.4.1
```

1.1 Analytic solution.

See the book by Crank (“Mathematics of Diffusion”), page 78, section 5.3.

Equation (5.22) reads

$$\frac{C-C_1}{C_0-C_1} = 1 - \frac{2}{a}\sum_{n=1}^{\infty}\frac{\exp(-D\alpha_n^2 t)J_0(r\alpha_n)}{\alpha_n J_1(a\alpha_n)}$$

Here we will evaluate and plot this equation.

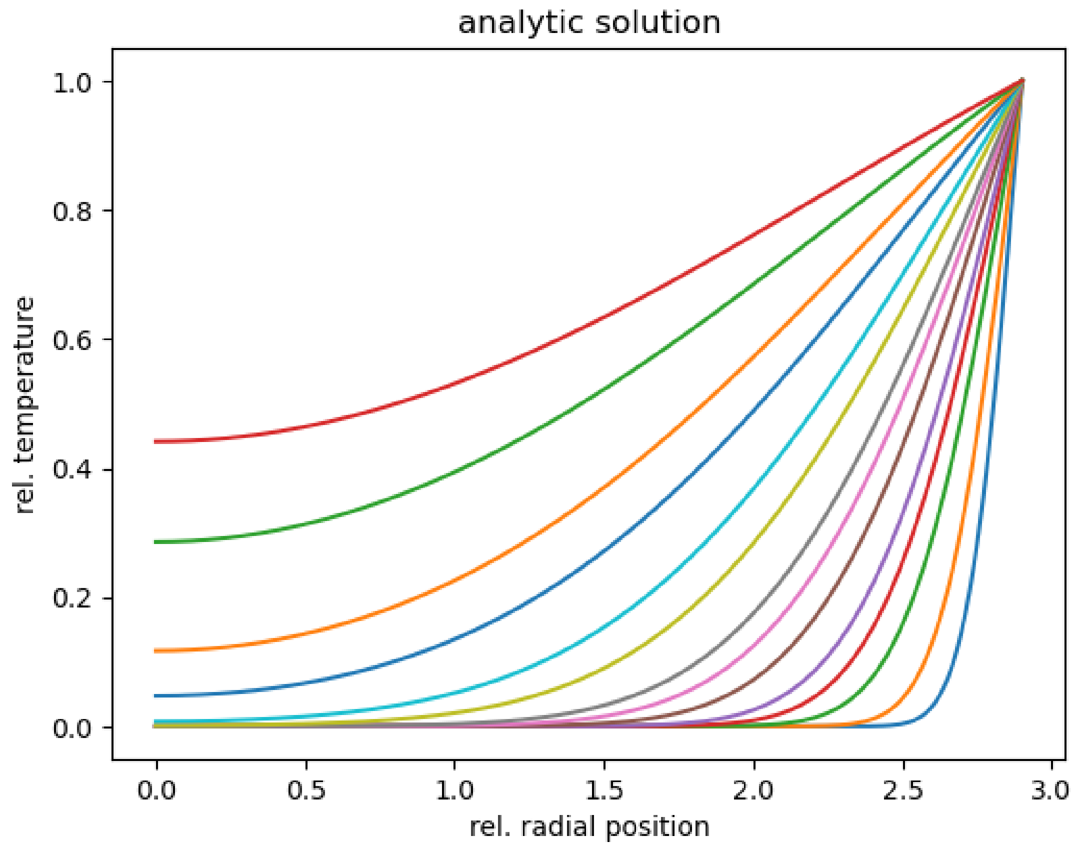
[2]:
```python
def crank522(r, t, a, D):
    '''evaluate eqn 5.22 for a given r,t
    a : radius of cylinder
    D : diffusion coefficient

    the following global variables need to be set

    Nterm_crank : number of terms to be evaluated
    '''
```

```
    global Nterm_crank

    aalp = jn_zeros(0, Nterm_crank)
    alpha = aalp/a

    XJ0 = exp(-D * alpha**2 * t) * j0(r*alpha)
    AJ1 = alpha * j1(aalp)

    S = np.sum(XJ0/AJ1)

    return 1.0 - (2.0/a) * S
```

[3]:
```
# set world parameters
a_val = 2.9
D_val = 1.9
Nterm_crank = 30
```

[4]:
```
# create radial axis for plotting analytic solution
r_an = np.linspace(0.,a_val,200)
```

[5]:
```
# evaluate crank522 at different points in time, and plot
for t in [0.005, 0.01, 0.02, 0.03, 0.04, 0.06, 0.08, 0.1, 0.15,
          0.2, 0.3, 0.4, 0.6, 0.8]:
    c = np.array([crank522(rr, t, a_val, D_val) for rr in r_an])
    plt.plot(r_an, c)
plt.xlabel('rel. radial position')
plt.ylabel('rel. temperature')
plt.title('analytic solution')
plt.show()
```



1.2 Finite-volume solution with PyFVTool.

Define 1D cylindrical grid with a variable called ‘c’, initialized to an initial value of 0.0 everywhere.

[6]:
```
Nr = 50
Lr = a_val
c_outer = 1.0 # (outer) boundary concentration
c0 = 0.0
deltat = 0.001
```

[7]:
```
mesh = pf.CylindricalGrid1D(Nr, Lr)
```

[8]:
```
c = pf.CellVariable(mesh, c0)
```

By default, the boundary conditions for a variable are of the ‘no flux’ (Neumann) type.

Here, we apply a different boundary condition: the outer (rightmost) boundary will be kept at 1.0 (Dirichlet boundary condition).

[9]:
```
# switch the right (=outer) boundary to Dirichlet: fixed concentration
c.BCs.right.a[:] = 0.0
```

```
c.BCs.right.b[:] = 1.0
c.BCs.right.c[:] = c_outer
```

When changing the BCs, it is necessary to update everything in the cell variable object by calling apply_BCs()

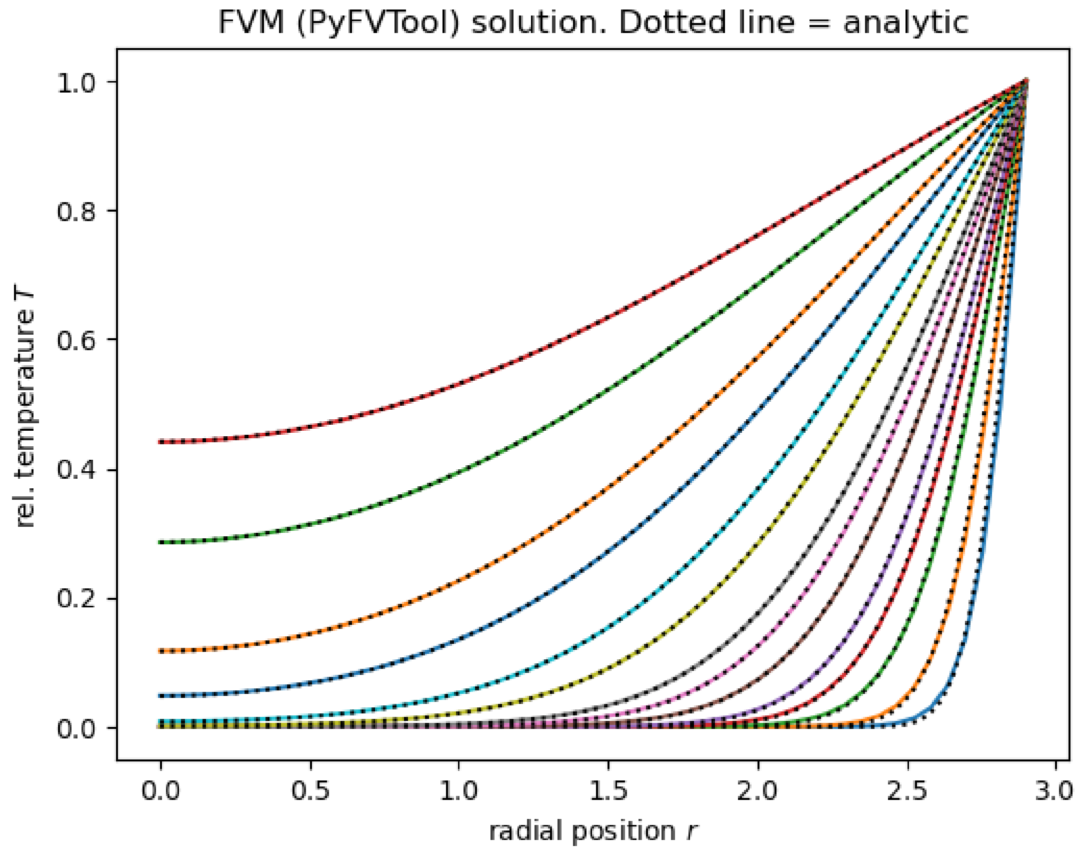
[10]:
```
c.apply_BCs()
```

[11]:
```
D = pf.CellVariable(mesh, D_val) # diffusion coefficient
alfa = pf.CellVariable(mesh, 1.0) # transientterm coefficient
```

[12]:
```
t = 0.0
```

Now, we solve the equation by taking time steps. We plot the solution at several specified time-points, together with the analytic solution.

[13]:
```
r_an = np.linspace(0., a_val,200) # axis for plotting analytic solution

sample_i = [5,10,20,30,40,60,80,100,150,200,300,400,600,800]

for i in range(0,1001):
    if i in sample_i:
        r, phi = c.plotprofile()
        plt.plot(r, phi)
        c_an = np.array([crank522(rr,t,a_val,D_val) for rr in r_an])
        plt.plot(r_an,c_an, 'k:')
    # calculate the value of D at the faces
    # as the harmonic mean of the values at the centers
    Dave = pf.harmonicMean(D)
    pf.solvePDE(c,
                [ pf.transientTerm(c, deltat, alfa),
                 -pf.diffusionTerm(Dave)])
    t += deltat
plt.xlabel('radial position $r$')
plt.ylabel('rel. temperature $T$')
plt.title('FVM (PyFVTool) solution. Dotted line = analytic');
```


FVM (PyFVTool) solution. Dotted line = analytic
rel. temperature T
radial position r
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0

1.3 End.

```
[ ]:
```