In [24]:

```python
import h5py
import numpy as np
import matplotlib.pyplot as plt
import CLB.CLBXMLWriter as CLBXML
import tempfile
import scipy.optimize as so
from display_xml import XML

from scipy.integrate import solve_ivp
```

First shot: spatially variable SIR model with uniform IC

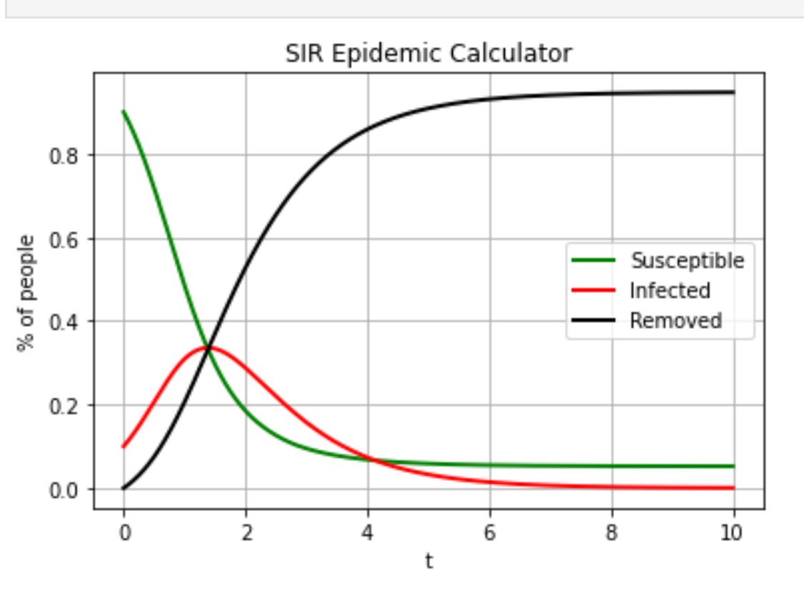
```
Zadanie E - Benchmark I: Poprawność symulacji równań typu SIR w domenie jednorodnej (0D) vs
analityczne/Runge-Kutta
 - output jak do raportu/artykułu (nie do szkolenia) .md
 - dodać skrypt do examples and papers
 - dodać ref do repo :)
```

To evaluate, if model is working, let's start with basic case: uniform initial distribution of SIR variables, with periodic BC. This should give solution for original SIR ODE, if β is sufficiently large.

Firstly, we prepare solution of ODE using SymPy's solve_ivp toolbox. For details see previous workshops.

In [102…

```python
#Test case settings

I_init = 0.1
S_init = 1. - I_init
R_init = 0
beta = 1e20
dt = 0.01
R0 = 3
T = 1000



##########################3
# FD SOLUTION #


def SIR_0D(t, z, beta, gamma, N):
    """
     # Susceptible → Infected → Removed
    :param t: time [days]
    :param z: Susceptible, Exposed, Infected, Removed
    :param beta: average number of contacts per day for each infected individual
    :param gamma: Between I and R, the transition rate is γ
    (simply the frequency of recoveries, that is, number of recovered or dead during one day
    divided by the total number of infected on that same day, supposing "day" is the time unit).
    If the duration of the infection is denoted D, then γ = 1/D.
    :return: derivatives [dS, dI, dR]
    """

    S, I, R = z
    dSdt = -beta*I*S/N
    dIdt = beta*I*S/N - I*gamma
    dRdt = I*gamma
    return [dSdt, dIdt, dRdt]



# INITIAL CONDItIONS
initial_susceptible = S_init # initial number of susceptible individuals in population.
initial_infections = I_init  # initial number of infected individuals in population.
initial_removed = R_init  # initial number of removed (recovered) individuals in population.
IC = np.array([initial_susceptible, initial_infections, initial_removed])

days_to_simulate = T*dt
sol = solve_ivp(SIR_0D,
                [0, days_to_simulate],
                IC,
                method='RK45',
                args=[R0, 1, 1],
                dense_output=True)

t = np.linspace(0, days_to_simulate, 1000)
z = sol.sol(t)

S_rk4, I_rk4, R_rk4 = z

params = {'legend.fontsize': 'xx-large',
          'figure.figsize': (14, 8),
         'axes.labelsize': 'xx-large',
         'axes.titlesize':'xx-large',
         'xtick.labelsize':'xx-large',
         'ytick.labelsize':'xx-large'}
axes = plt.gca()
plt.plot(t, S_rk4,
         color="green", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
         label='Susceptible')
plt.plot(t, I_rk4,
         color="red", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
         label='Infected')
plt.plot(t, R_rk4,
         color="black", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
         label='Removed')


plt.xlabel('t')
plt.ylabel('% of people')
plt.title('SIR Epidemic Calculator')
plt.legend()
plt.grid()
plt.show()
```



We will use **2q9_reaction_diffusion_system_SIR_ModifiedPeng** model (or the WSIR model), aleready referenced in this workshop

$$\frac{\partial}{\partial t}W = \beta_W\left[\frac{r^2}{8}W + (I - W)\right]$$

$$\frac{\partial}{\partial t}S = -\beta\frac{S}{N}W$$

$$\frac{\partial}{\partial t}I = \beta\frac{S}{N}W - \gamma I$$

$$\frac{\partial}{\partial t}R = \gamma I$$

where

$$S = ODE_1$$
$$I = ODE_2$$
$$R = ODE_3$$
$$C_1 = R_0$$
$$C_2 = \beta$$
$$C_3 = dt$$

$$S = ODE_1 \tag{1}$$
$$I = ODE_2 \tag{2}$$
$$R = ODE_3 \tag{3}$$
$$C_1 = R_0 \tag{4}$$
$$C_2 = \beta \tag{5}$$
$$C_3 = dt \tag{6}$$

$$S = ODE_1$$

In [103…

```python
CLBc = CLBXML.CLBConfigWriter( )

CLBc.addGeomParam('nx', 100)
CLBc.addGeomParam('ny', 2)


params = {
        "Diffusivity_DRE_1" : 1./6.,
        "C_1":R0,
        "C_2":beta,
        "C_3":dt,
        "Init_DRE_1":I_init, #This is W equation, initlally equall to I. large Beta reduces Relaxation-To-SIR
        "Init_ODE_1":S_init,
        "Init_ODE_2":I_init,
        "Init_ODE_3":R_init
}

CLBc.addModelParams(params)


CLBc.addHDF5()
solve = CLBc.addSolve(iterations=T)
CLBc.addHDF5(Iterations=int(T / 50), parent=solve)

CLBc.write('run.xml')
```

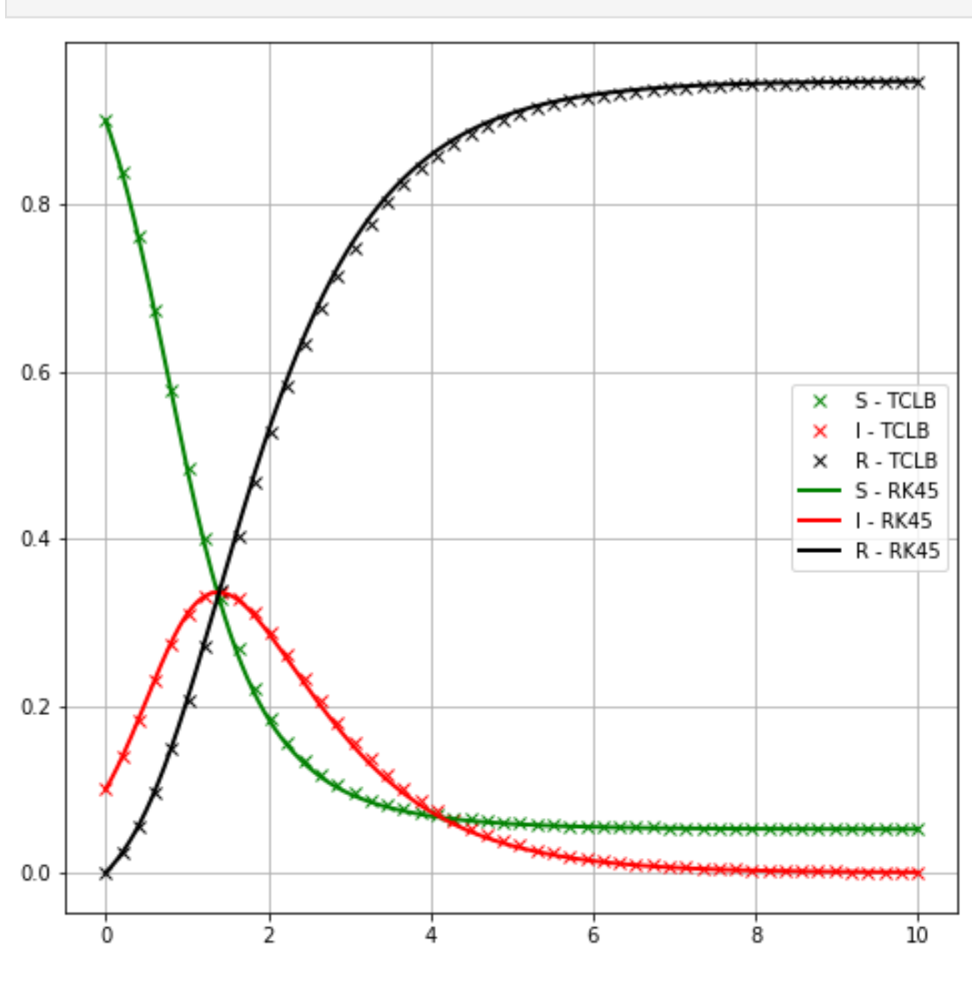
In [107…

```
! rm -rf output/* && tclb d2q9_reaction_diffusion_system_SIR_ModifiedPeng run.xml
```

```
MPMD: TCLB: local:0/1 work:0/1 ---  connected to:
[  ] -------------------------------------------------------------------------
[  ] -  CLB version:   v6.0-beta-1713-g0861a50                              -
[  ] -        Model: d2q9_reaction_diffusion_system_SIR_ModifiedPeng
[  ]                 -
[  ] -------------------------------------------------------------------------
[  ] Setting output path to: run
[  ] Discarding 1 comments
[ 0] Running on CPU
[ 0] WARNING: No "Units" element in config file
[  ] Mesh size in config file: 100x2x1
[  ] Global lattice size: 100x2x1
[  ] Max region size: 200. Mesh size 200. Overhead:  0%
[  ] Local lattice size: 100x2x1
Hello allocator!
[  ]   Threads  |      Action
[  ]     1x1    | Primal , NoGlobals , BaseIteration
[  ]     1x1    | Tangent , NoGlobals , BaseIteration
[  ]     1x1    | Optimize , NoGlobals , BaseIteration
[  ]     1x1    | SteadyAdjoint , NoGlobals , BaseIteration
[  ]     1x1    | Primal , IntegrateGlobals , BaseIteration
[  ]     1x1    | Tangent , IntegrateGlobals , BaseIteration
[  ]     1x1    | Optimize , IntegrateGlobals , BaseIteration
[  ]     1x1    | SteadyAdjoint , IntegrateGlobals , BaseIteration
[  ]     1x1    | Primal , OnlyObjective , BaseIteration
[  ]     1x1    | Tangent , OnlyObjective , BaseIteration
[  ]     1x1    | Optimize , OnlyObjective , BaseIteration
[  ]     1x1    | SteadyAdjoint , OnlyObjective , BaseIteration
[  ]     1x1    | Primal , NoGlobals , BaseInit
[  ]     1x1    | Tangent , NoGlobals , BaseInit
[  ]     1x1    | Optimize , NoGlobals , BaseInit
[  ]     1x1    | SteadyAdjoint , NoGlobals , BaseInit
[  ]     1x1    | Primal , IntegrateGlobals , BaseInit
[  ]     1x1    | Tangent , IntegrateGlobals , BaseInit
[  ]     1x1    | Optimize , IntegrateGlobals , BaseInit
[  ]     1x1    | SteadyAdjoint , IntegrateGlobals , BaseInit
[  ]     1x1    | Primal , OnlyObjective , BaseInit
[  ]     1x1    | Tangent , OnlyObjective , BaseInit
[  ]     1x1    | Optimize , OnlyObjective , BaseInit
[  ]     1x1    | SteadyAdjoint , OnlyObjective , BaseInit
[  ] [0] Cumulative allocation of 71424 b (71.4 kB)
[  ] Creating geom size:200
[  ] Setting output path to: run
[  ] Setting output path to: output/run
[  ] loading geometry ...
[  ] Setting number of zones to 1
[  ] Setting Diffusivity_DRE_1 to 0.1666666666666667 (0.166667)
[  ] [0] Settings [Diffusivity for  DRE_1] to 0.166667
[  ] Setting C_1 to 3.0000000000000000 (3.000000)
[  ] [0] Settings [Model parameter C_1] to 3.000000
[  ] Setting C_2 to 100000000000000000000.0000000000000000 (100000000000000000000.000
[  ] 000)
[  ] [0] Settings [Model parameter C_2] to 100000000000000000000.000000
[  ] Setting C_3 to 0.0100000000000000 (0.010000)
[  ] [0] Settings [Model parameter C_3] to 0.010000
[  ] Setting Init_DRE_1 in zone  (-1) to 0.1000000000000000 (0.100000)
[  ] Setting Init_ODE_1 in zone  (-1) to 0.9000000000000000 (0.900000)
[  ] Setting Init_ODE_2 in zone  (-1) to 0.1000000000000000 (0.100000)
[  ] Setting Init_ODE_3 in zone  (-1) to 0.0000000000000000 (0.000000)
[  ] Initializing Lattice ...
[  ] Callback HDF5 with no Iterations attribute==]  0s
[  ] Negotiated HDF5 chunks: 1x2x100[x3]
[  ]        0 it writing hdf5
[  ] Setting action Solve at 1000.000000 iterations
[  ] Setting callback HDF5 at 20.000000 iterations
[  ] Negotiated HDF5 chunks: 1x2x100[x3]
[  ] Adding HDF5 to the solver hands
[  ]      1.3 MLBUps      0.28 GB/s [====================]
[  ]       20 it writing hdf5
[  ]      0.5 MLBUps      0.10 GB/s [====================]
[  ]       40 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]       60 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]       80 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      100 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      120 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      140 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      160 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      180 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      200 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      220 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      240 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      260 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      280 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      300 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      320 it writing hdf5
[  ]      0.6 MLBUps      0.12 GB/s [====================]
[  ]      340 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      360 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      380 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      400 it writing hdf5
[  ]      0.6 MLBUps      0.13 GB/s [====================]
[  ]      420 it writing hdf5
[  ]      0.7 MLBUps      0.14 GB/s [====================]
[  ]      440 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      460 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      480 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      500 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      520 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      540 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      560 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      580 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      600 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      620 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      640 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      660 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      680 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      700 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      720 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      740 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      760 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      780 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      800 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      820 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      840 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      860 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      880 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      900 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      920 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      940 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      960 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]      980 it writing hdf5
[  ]      0.8 MLBUps      0.16 GB/s [====================]
[  ]     1000 it writing hdf5
[  ] Total duration: 0.338237 s = 0.005637 min = 0.000094 h
```

In [108…

```python
S = list()
I = list()
R = list()

for i in range(0,T,int(T / 50)):
    f = h5py.File('output/run_HDF5_%08d.h5'%i)
    #plt.plot(f['DRE_1'][0,25,:])
    S.append( f['ODE_1'][0,0,0] )
    I.append( f['ODE_2'][0,0,0] )
    R.append( f['ODE_3'][0,0,0] )

t_lb = np.linspace(0,T,len(S))*dt


plt.figure(figsize=(8,8))

plt.plot(t_lb,S, 'gx', label='S - TCLB')
plt.plot(t_lb,I, 'rx', label='I - TCLB')
plt.plot(t_lb,R, 'kx', label='R - TCLB')

plt.plot(t, S_rk4,
         color="green", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
         label='S - RK45')
plt.plot(t, I_rk4,
         color="red", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
         label='I - RK45')
plt.plot(t, R_rk4,
         color="black", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
         label='R - RK45')
plt.legend()

plt.grid(which='both')
```



We could use TCLB shortcuts in a loop, to see effects of β

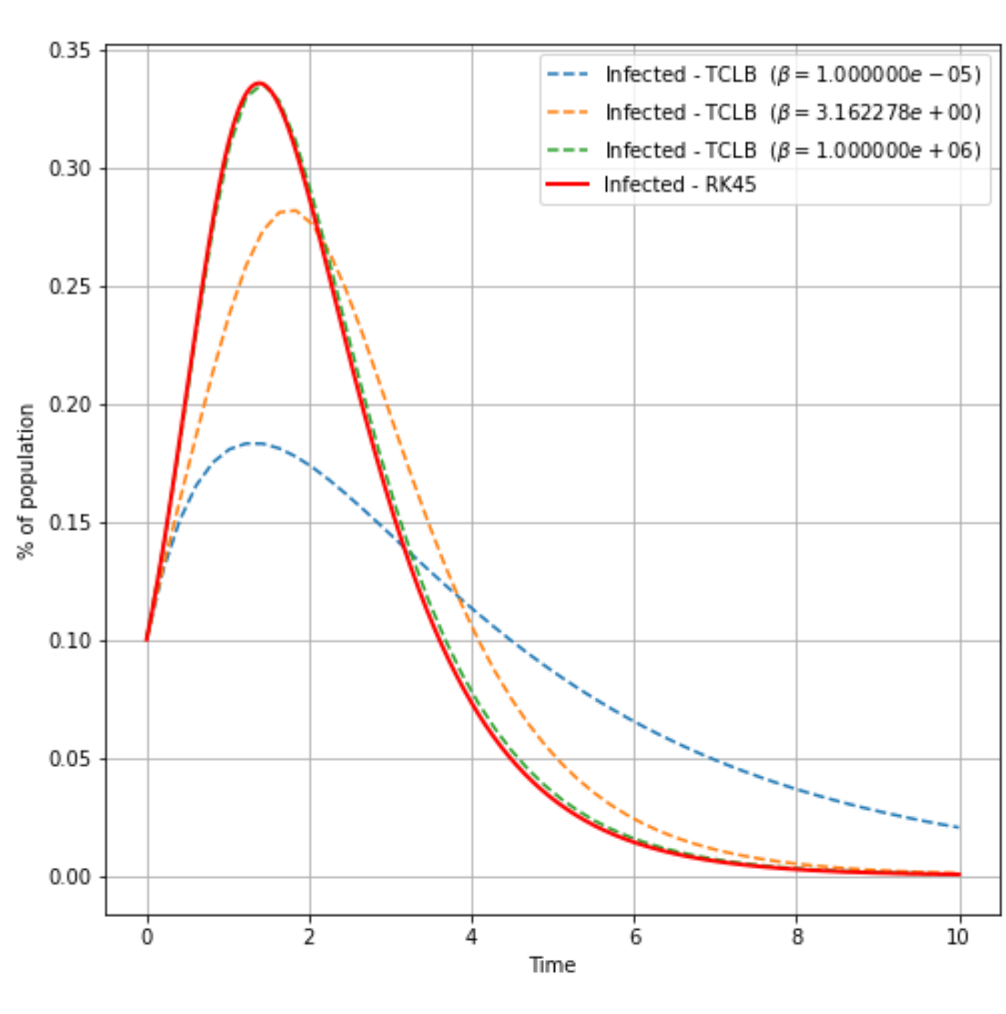
In [109…

```python
plt.figure(figsize=(8,8))

for _beta in np.logspace(-5,6,3):

    CLBc = CLBXML.CLBConfigWriter( )

    CLBc.addGeomParam('nx', 5)
    CLBc.addGeomParam('ny', 5)


    params = {
            "Diffusivity_DRE_1" : 1./6.,
            "C_1":R0,
            "C_2":_beta,
            "C_3":dt,
            "Init_DRE_1":I_init, #This is W equation, initlally equall to I. large Beta reduces Relaxation-To-S
            "Init_ODE_1":S_init,
            "Init_ODE_2":I_init,
            "Init_ODE_3":R_init
    }

    CLBc.addModelParams(params)


    CLBc.addHDF5()
    solve = CLBc.addSolve(iterations=T)
    CLBc.addHDF5(Iterations=int(T / 50), parent=solve)

    CLBc.write('run.xml')

    ! rm -rf output/* &&  tclb d2q9_reaction_diffusion_system_SIR_ModifiedPeng run.xml > /dev/null && echo "DON


    S = list()
    I = list()
    R = list()

    for i in range(0,T,int(T / 50)):
        f = h5py.File('output/run_HDF5_%08d.h5'%i)
        #plt.plot(f['DRE_1'][0,25,:])
        S.append( f['ODE_1'][0,0,0] )
        I.append( f['ODE_2'][0,0,0] )
        R.append( f['ODE_3'][0,0,0] )

    t_lb = np.linspace(0,T,len(S))*dt

#    plt.plot(t_lb,S, 'g-', label=r'S - TCLB ($\beta=$)')
    plt.plot(t_lb,I, '--', label=r'Infected - TCLB  ($\beta=%e$)'%_beta )

#    plt.plot(t_lb,R, 'k-', label='R - TCLB')

#plt.plot(t, S_rk4,
#         color="green", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
#         label='S - RK45')
plt.plot(t, I_rk4,
         color="red", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
         label='Infected - RK45')
#plt.plot(t, R_rk4,
#         color="black", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
#         label='R - RK45')
plt.legend()
plt.grid(which='both')
plt.xlabel('Time')
plt.ylabel('% of population')
```

```
Hello allocator!
DONE!
Hello allocator!
DONE!
Hello allocator!
DONE!
```

Out[109…

```
Text(0, 0.5, '% of population')
```



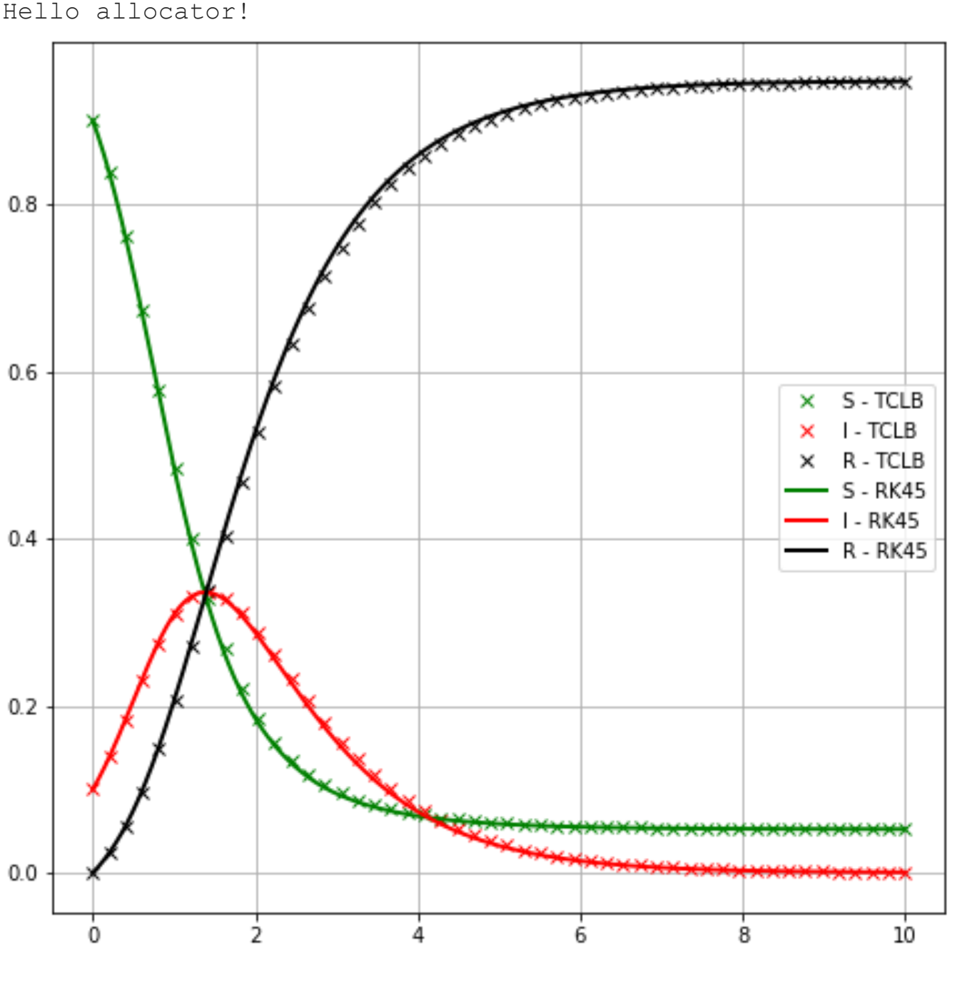
SIR - Simple Laplace

The same could be done with SIR_SimpleLaplace model - for large β results are identical

In [110…

```python
CLBc = CLBXML.CLBConfigWriter( )

CLBc.addGeomParam('nx', 5)
CLBc.addGeomParam('ny', 5)


params = {
        "Diffusivity_DRE_1" : 1./6.,
        "Diffusivity_DRE_2" : 1./6.,
        "Diffusivity_DRE_3" : 1./6.,
        "C_1":R0,
        "C_2":dt,
        "Init_DRE_1":S_init,
        "Init_DRE_2":I_init,
        "Init_DRE_3":R_init
}

CLBc.addModelParams(params)


CLBc.addHDF5()
solve = CLBc.addSolve(iterations=T)
CLBc.addHDF5(Iterations=int(T / 50), parent=solve)

CLBc.write('run.xml')

! rm -rf output/* &&  tclb d2q9_reaction_diffusion_system_SIR_SimpleLaplace run.xml > /dev/null


S = list()
I = list()
R = list()

for i in range(0,T,int(T / 50)):
    f = h5py.File('output/run_HDF5_%08d.h5'%i)
    #plt.plot(f['DRE_1'][0,25,:])
    S.append( f['DRE_1'][0,0,0] )
    I.append( f['DRE_2'][0,0,0] )
    R.append( f['DRE_3'][0,0,0] )

t_lb = np.linspace(0,T,len(S))*dt


plt.figure(figsize=(8,8))

plt.plot(t_lb,S, 'gx', label='S - TCLB')
plt.plot(t_lb,I, 'rx', label='I - TCLB')
plt.plot(t_lb,R, 'kx', label='R - TCLB')

plt.plot(t, S_rk4,
         color="green", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
         label='S - RK45')
plt.plot(t, I_rk4,
         color="red", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
         label='I - RK45')
plt.plot(t, R_rk4,
         color="black", marker="", markevery=1, markersize=15, linestyle="-", linewidth=2,
         label='R - RK45')
plt.legend()

plt.grid(which='both')
```

```
Hello allocator!
```



Simple spatially variable SIR model with non-uniform IC

In [ ]: