helpers.py

```python
import csv
import numpy as np
import matplotlib.pyplot as plt
import casadi

P = np.array([1058304,915796,983789,384803,203035,99516])
n_a = len(P)

def definitions():
  I_0 = 0.1/100*P
  S_0 = P - I_0
  R_0 = np.zeros(n_a)
  D_0 = np.zeros(n_a)

  l   = np.repeat(0.05, n_a)
  g_R = np.array([0.7657411   ,0.7842402  ,0.8012127 ,0.9018488 , 0.2802379 ,0.5864928 ])
  g_D = np.array([0.0015683025,0.004833996,0.09288585,0.09685946, 0.17079121,0.56594825])

  with open("./contact.csv", 'r') as f:
    reader = csv.reader(f)
    data = list(reader)
  C = np.array(data, dtype=float)

  u_max = 55191

  return P, S_0, I_0, R_0, D_0, l, C, g_R, g_D, u_max

def wrap(S, I, R, D):
  if type(S) is np.ndarray:
    return np.concatenate([S, I, R, D])
  if type(S) is casadi.MX:
    return casadi.vertcat(S, I, R, D)

def unwrap(y):
  S = y[0*n_a:1*n_a]
  I = y[1*n_a:2*n_a]
  R = y[2*n_a:3*n_a]
  D = y[3*n_a:4*n_a]
  return S, I, R, D

def solve_ivp_discrete(system, t_span, y_0, args):
  k = t_span[-1] - t_span[0] + 1
  t = np.linspace(t_span[0], t_span[-1], k)

  y = np.empty((len(y_0),k))
  for i, t_ in enumerate(t):
    if i == 0:
      y[:,i] = y_0
    else:
      y[:,i] = system(t_, y[:,i-1], *args)
  return t, y

def recover_control(t, y, u, u_max):
  u_ = np.zeros((n_a, len(t)))
  for i, t_ in enumerate(t):
    y_ = y[:,i]
    u_[:,i] = u(t_, y_, u_max)
  return u_

def plot(t, y, u, discrete=False):
  groups = ['[0,25)', '[25,45)', '[45,65)', '[65,75)', '[75,85)', '85+']
  assert(len(groups) )

  S, I, R, D = unwrap(y)

  fig, ax = plt.subplots(5, 2, facecolor=(1,1,1,1), figsize=(20,15), sharex=True)

  if discrete:
    ds = 'steps-post'
    step = 'post'
  else:
    ds = 'default'
    step = None

  for i in range(n_a):
    ax[0,0].plot(t, S[i], drawstyle=ds)
    ax[1,0].plot(t, I[i], drawstyle=ds)
    ax[2,0].plot(t, R[i], drawstyle=ds)
    ax[3,0].plot(t, D[i], drawstyle=ds)
    ax[4,0].plot(t, u[i], drawstyle=ds)

  ax[0,1].stackplot(t, S, step=step)
  ax[1,1].stackplot(t, I, step=step)
  ax[2,1].stackplot(t, R, step=step)
  ax[3,1].stackplot(t, D, step=step)
  ax[4,1].stackplot(t, u, step=step)

  for i in range(2):
    ax[0,i].set_ylabel('Susceptibles')
    ax[1,i].set_ylabel('Infected')
    ax[2,i].set_ylabel('Recovered')
    ax[3,i].set_ylabel('Deceased')
    ax[4,i].set_ylabel('Vaccination Rate')

  for ax_ in ax.flatten():
    ax_.set_xlim((t[0],t[-1]))
    ax_.grid()

  for ax_ in ax[-1,:]:
    ax_.set_xlabel('time')

  fig.tight_layout()
  fig.legend(groups, title='Age Group',ncols=n_a, loc="upper center", bbox_to_anchor=(0.5, 0))
```

```
    plt.show()

```