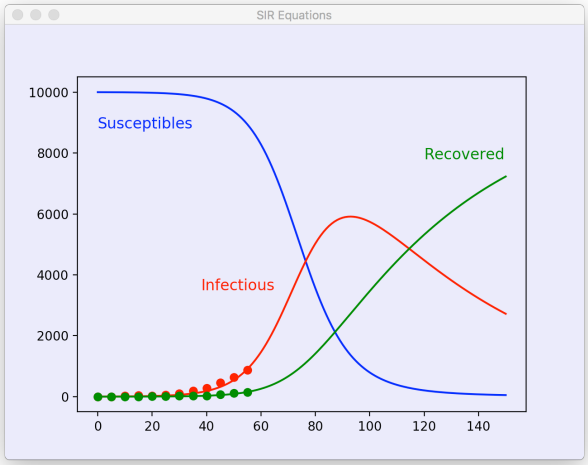
Python for dummies : problème 6
SIR : les équations de Kermack et McKendrick :
Un modèle d'épidémie de 1927.

Il s'agit de résoudre le système d'équations différentielles ordinaires :

$$\begin{cases} S'(t) &= -\beta S(t)I(t) \\ I'(t) &= \beta S(t)I(t) - \gamma I(t) \\ R'(t) &= \gamma I(t) \end{cases}$$

où $S(t)$ est le nombre d'individus sains et susceptibles d'être infectés. Ensuite, $I(t)$ est le nombre d'individus infectés et infectieux. Finalement, $R(t)$ est le nombre d'individus immunisés, morts, isolés ou guéris, qui ne peuvent pas ou plus transmettre la maladie. Pour être optimiste, on va dire que ce sont les personnes guéries.



Les modèles compartimentaux permettent de prédire les propagations des maladies infectieuses. Toute la population est divisée en compartiments, en supposant que chaque individu dans le même compartiment a les mêmes caractéristiques. Ces modèles compartimentaux peuvent être utilisés pour prédire la propagation d'une maladie et la durée d'une épidémie. En outre, cela permet aussi de comprendre comment différentes situations peuvent affecter les résultats de l'épidémie, par exemple, quelle est la technique la plus efficace pour délivrer un nombre limité de vaccins dans une population donnée.... Historiquemement, une étape décisive dans l'étude mathématique de la dynamique des épidémies a été franchie en 1927 avec le modèle SIR de Kermack et McKendrick. Le modèle SIR est l'un des modèles compartimentaux les plus simples. Le modèle comprend donc trois compartiments S, I et R. La taux de transfert entre les compartiment S et I (la contamination :-) est donnée βSI, où β caractérise le nombre moyen de contacts par personne et par jour, multiplié par la probabilité de transmission de la maladie lors d'un contact entre un sujet sensible et un sujet infectieux. Le taux de transfert entre le I et R est simplement défini par γI où $1/\gamma$ caractérise le temps moyen requis pour être guéri et ne plus être infectieux.[1]

En définissant $\mathbf{u}(t) = \begin{bmatrix} S(t) \\ I(t) \\ R(t) \end{bmatrix}$ et $\mathbf{f}(\mathbf{u}(t)) = \begin{bmatrix} -\beta S(t)I(t) \\ \beta S(t)I(t) - \gamma I(t) \\ \gamma I(t) \end{bmatrix}$,

on peut écrire les trois équations du modèle SIR sous la forme compacte :

$$\mathbf{u}'(t) = \mathbf{f}(\mathbf{u})$$

Pour résoudre ce problème, nous allons considérer les méthodes d'Euler explicite, de Taylor d'ordre quatre et de Runge-Kutta d'ordre quatre. En prenant comme conditions initiales $S(0) = 10000$, $I(0) = 2$ et $R(0) = 0$, ce devoir consistera à obtrenir les trois composantes de $\mathbf{u}$ pour $0 \leq t \leq 150$ jours pour des paramètres β et γ donnés avec ces trois méthodes numériques.

[1] https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology

Plus précisément, on vous demande de :

1. Tout d'abord, implémenter la fonction suivante
   `X,U = epidemicEuler(Xstart,Xend,Ustart,n,beta,gamma)`.

   - Les arguments `Xstart` et `Xend` définissent l'intervalle d'intégration temporelle.
   - L'argument `Ustart` définit les 3 valeurs initiales pour le vecteur (S, I, R) dans une liste ou un tableau de longueur 3.
   - L'argument `n` contient le nombre des pas égaux à effectuer avec la méthode d'Euler.
   - Les arguments `beta` et `gamma` sont les deux paramètres du modèle SIR.
   - Votre fonction fournira tout d'abord les $n+1$ temps X_i et les valeurs discrète (S_i, I_i, R_i). Le tableau `X` sera un tableau unidimensionnel `numpy.ndarray` de longueur $n+1$, tandis que le tableau `U` sera un tableau bidimensionnel `numpy.array` de taille $(n+1) \times 3$.

2. Ensuite, implémenter la méthode de Taylor d'ordre quatre dans la fonction `X,U = epidemicTaylor(Xstart,Xend,Ustart,n,beta,gamma)`.

3. Finalement, implémenter la méthode dde RungeKutta d'ordre quatre dans la fonction `X,U = epidemicRungeKutta(Xstart,Xend,Ustart,n,beta,gamma)`. Les méthodes de Taylor et de Runge-Kutta ne sont pas identiques pour un problème non-linéaire !

4. Un programme `epidemicTest.py` vous est fourni pour tester l'ordre de convergence de vos méthodes.

```
from numpy import *
from epidemic import epidemicEuler,epidemicTaylor,epidemicRungeKutta

class EpidemicExplMethods(object):
  def __init__(self,name,order,integrator):
    self.name = name
    self.order = order
    self.f = integrator

integrators = [EpidemicExplMethods("Explicit Euler",1,epidemicEuler),
               EpidemicExplMethods("Taylor",4,epidemicTaylor),
               EpidemicExplMethods("Runge-Kutta",4,epidemicRungeKutta)]

Ustart = [10000,2,0]; beta  = 0.00001315; gamma = 0.01798476
Xstart = 0; Xend = 50; Uref = 541.867329232525; erros = zeros(4)

for integrator in integrators:
  for j in range(4):
    n = 50*pow(2,j); h = (Xend - Xstart)/n
    X,U = integrator.f(Xstart,Xend,Ustart,n,beta,gamma)
    error[j] = abs(U[-1,1]-Uref)
    print(" ==== %4s (order=%d) n=%6d h=%6.3f :    U(%d) = %14.7e : eh(Xend) = %8.2e "
            % (integrator.name.rjust(15),integrator.order,n,h,Xend,U[-1,1],error[j]))
  order = mean(log(error[:-1]/error[1:])/log(2))
  print(" ================ Estimated order of %s : %.4f " % (integrator.name,order))
```

Bien regarder comment on évalue la manière d'estimer l'erreur de convergence d'une méthode de manière expérimentale ! Evidemment, sur le serveur, nous utiliserons d'autres arguments que ceux de ce programme : il est donc nécessaire d'écrire le code de manière tout-à-fait générale puisque on pourra par exemple changer la valeur de β :-)

5. Votre fonction (avec les éventuelles sous-fonctions que vous auriez créées) sera incluse dans un unique fichier `epidemic.py`, sans y adjoindre le programme de test fourni ! Cette fonction devra être soumise via le web avant le dimanche 29 mars à 23h59 : ce travail est individuel et sera évalué.

Le modèle SIR : est-ce pertinent pour l'épidémie actuelle du COVID19 ?

Il est possible d'identifier les paramètres β et γ de ce modèle pour le COVID-19 aux données actuelles du COVID19[2]. A titre d'exemple, considérons le cas du Japon et sélectionnons les deux paramètres pour avoir une bonne correspondance entre les observations et le modèle. Par contre, j'ai choisi de manière assez brutale et un peu arbitraire les conditions initiales... En fait, comment décider des valeurs initiales des compartiments ? Le modèle SIR s'attend à ce que le compartiment des personnes susceptibles d'être contaminés soit homogène, bien mélangé et accessible les uns aux autres. Ce n'est pas le cas de toute la population du Japon et il n'est donc ni réaliste, ni raisonnable de considérer l'ensemble de tous les japonais. Une option est de remplacer la totalité de la population du Japon qui est réparti dans des zones géographiques distinctes et pas parfaitement homogènes et mélangées, par une population parfaitement mélangée mais de taille plus réduite : j'ai choisi une population de 10000 personnes. Oui, c'est assez arbitraire et donc ma prédiction est assez peu fiable. Mais sur cette base, voilà une prédiction de l'évolution de l'épidémie du COVID-19 pour le Japon : eh bien, faudra être patients, les gamins !

```
Xstart = 0; Xend = 150;
m = 10; n = Xend * m

#
#  Pour comprendre pourquoi se confiner est utile :-)
#  Se confiner permet de réduire la valeur de beta
#  Tester les deux lignes qui précèdent !
#  Ensuite, on peut aussi imaginer de faire varier beta dans le temps pour être plus réaliste !
#
#     Xend = 300; beta = beta/2
#     Xend = 1200; beta = beta/4
#

X,U = epidemicRungeKutta(Xstart,Xend,Ustart,n,beta,gamma)

dataConfirmed = array([2,1,2,2,4,4,7,7,11,15,20,20,20,22,22,45,25,25,26,26,26,28,28,29,43,59,66,
                       74,84,94,105,122,147,159,170,189,214,228,241,256,274,293,331,360,420,461,
                       502,511,581,639,639,701,773,839,825,878,889,924,963])
dataRecovered = array([0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,4,9,9,9,9,12,12,12,13,18,18,22,22,
                       22,22,22,22,22,32,32,32,43,43,43,46,76,76,76,101,118,118,118,118,118,
                       144,144,144,150,191])
days = arange(dataConfirmed.shape[0])

from matplotlib import pyplot as plt
plt.figure("SIR Equations")
X=X[::m]; U=U[::m,:];
plt.plot(X,U[:,0],'-b',X,U[:,1],'-r',X,U[:,2],'-g')
plt.plot(days[::5],dataConfirmed[::5],'or')
plt.plot(days[::5],dataRecovered[::5],'og')
plt.text(0,8800,"Susceptibles",color='blue',fontsize=12)
plt.text(38,3500,"Infectious",color='red',fontsize=12)
plt.text(120,7800,"Recovered",color='green',fontsize=12)
plt.show()
```

[2] https://data.humdata.org/dataset/novel-coronavirus-2019-ncov-cases

Attention : ce petit exercice a juste pour objet d'appliquer des méthodes numériques et de vous faire découvrir de manière un peu intutitive, la dynamique d'une propagation d'épidémies. C'est le modèle le plus simple possible : vous n'êtes donc pas encore des épidémiologistes, ni amateurs, ni même dilettantes. D'ailleurs, le professeur n'y connaît rien non plus : donc, ma prédiction n'a strictement aucune valeur scientifique : c'est juste, un petit essai pour comprendre les méthodes numériques. C'est seulement l'occasion d'essayer de vous convaincre que des personnes vraiment expertes en prédiction de la propagation de maladies infectieuses utilisent aujourd'hui avec brio les méthodes numériques pour essayer de mieux comprendre la propagation de l'épidémie d'aujourd'hui. Comme me le signale un collègue et ami, le modèle SIR est conceptuellement très intéressant, mais fréquemment inutile du point de vue quantitatif, sauf si l'on se contente de faire curve fitting qui, en définitive, ne signifie sans doute pas grand chose. Pour développer un modèle capable de prédire quelque chose, il faudrait coupler SIR, des réseaux et sans doute bien d'autres choses[3].

Méthodes explicites d'Euler, de Taylor d'ordre 4 et de Runge-Kutta d'ordre 4

- La méthode d'Euler est définie par : $\boxed{\mathbf{U}_{i+1} = \mathbf{U}_i + h\,\mathbf{F}_i}$

  où $\mathbf{F}_i$ représente la fonction $\mathbf{f}(\mathbf{u})$ évaluée en $\mathbf{u} = \mathbf{U}_i = \begin{bmatrix} S_i \\ I_i \\ R_i \end{bmatrix}$.

- La méthode de Taylor est définie par :

$$\boxed{\mathbf{U}_{i+1} = \mathbf{U}_i + h\,\mathbf{F}_i + \frac{h^2}{2}\mathbf{F}_i^{'} + \frac{h^3}{6}\mathbf{F}_i^{''} + \frac{h^4}{24}\mathbf{F}_i^{'''}}$$

  où $\mathbf{F}_i^{'}$, $\mathbf{F}_i^{''}$ et $\mathbf{F}_i^{'''}$ sont les dérivées de $\mathbf{f}(\mathbf{u}(t))$ par rapport à t et évaluées en $\mathbf{u} = \mathbf{U}_i$.

- La méthode de Runge-Kutta est définie par :

$$\boxed{\begin{aligned} \mathbf{U}_{i+1} &= \mathbf{U}_i + \tfrac{h}{6}(\mathbf{K}_1 + 2\mathbf{K}_2 + 2\mathbf{K}_3 + \mathbf{K_4}) \\ \mathbf{K}_1 &= \mathbf{f}(\mathbf{U}_i) \\ \mathbf{K}_2 &= \mathbf{f}(\mathbf{U}_i + \tfrac{h}{2}\mathbf{K}_1) \\ \mathbf{K}_3 &= \mathbf{f}(\mathbf{U}_i + \tfrac{h}{2}\mathbf{K}_2) \\ \mathbf{K}_4 &= \mathbf{f}(\mathbf{U}_i + h\mathbf{K}_3) \end{aligned}}$$

[3] https://www.imperial.ac.uk/school-public-health/infectious-disease-epidemiology/