Εισαγωγή στην python
(για Μαθηματική Μοντελοποίηση)

(Μεταβλητές, Συναρτήσεις, Γραφικές Παραστάσεις)

Διδάσκων: Σταύρος Κομηνέας

Χρήσιμοι σύνδεσμοι

- Μάθετε το linux command line.
- Εισαγωγικά μαθήματα python.

Εισαγωγή

Το πρόβλημα. Θα δούμε μία βασική διαφορική εξίσωση και έναν κώδικα για την λύση της στον οποίοι θα κοιτάξουμε τις απαραίτητες εντολές.

Ας θεωρήσουμε ραδιενεργούς πυρήνες και την διαδικασία με την οποία αυτοί διασπόνται. Αυτή είναι μια τυχαία διαδικασία και παρατηρούμε ότι υπάρχει μία πιθανότητα, έστω a, για την διάσπαση των πυρήνων. Δηλαδή, αν ο αριθμός των πυρήνων είναι N, τότε ο ρυθμός διάσπασης των πυρήνων είναι aN.
Ο αριθμός των πυρήνων ακολουθεί το μοντέλο

$$\frac{dN}{dt} = -aN$$

Θέλουμε να βρούμε τον αριθμό πυρήνων ως συνάρτηση του χρόνου N=N(t).
Ο κώδικας για την λύση του προβλήματος δίνεται στο τέλος της σελίδας.

Παρακάτω εξηγούμε κατά προτεραιότητα τις απλές αλλά σημαντικότερες εντολές ενώ κάποιες οι οποίες είναι λιγότερο σημαντικές εξηγούνται στο τέλος.

1ο κομμάτι του προγράμματος (Δεδομένα)

Εκχώρηση τιμής. Η βασικότερη λειτουργία είναι η εκχώρηση τιμής σε μεταβλητές. Π.χ.,
rate = 0.1
είναι εντολή η οποία ορίζει μία μεταβλητή με όνομα rate και της δίνει την τιμή 0.1.

Εισαγωγή τιμών.
Ας πάμε τώρα στο κύριο μέρος του προγράμματος
(το οποίο αρχίζει με το σχόλιο: - - - Main program - - -)
Στην εντολή
n0 = input('Give initial condition N0: ')
θα γίνει ανάγνωση από την οθόνη μίας τιμής η οποία θα δοθεί στην μεταβλητή n0 (θα χρησιμοποιηθεί ως η αρχική τιμή της μεταβλητής για την λύση της εξίσωσης).

Στις εντολές
t0 = input('Give initial time: ')
t1 = input('Give initial time: ')
ζητάει το πρόγραμμα τον αρχικό και τελικό χρόνο για την λύση της εξίσωσης

Οι εντολές
ntime = 20
time = linspace(t0,t1,ntime)
Κατασκευάζουν μία σειρά 20 αριθμών από το t0 έως το t1.

Άσκηση. Δοκιμάστε την εντολή linspace(0,5,6). Θα σας δώσει array([0., 1., 2., 3., 4., 5.])
Δηλαδή 6 τιμές, πρώτη τιμή 0, τελευταία το 5.

2ο κομμάτι του προγράμματος (λυση εξίσωσης)

nuclei = odeint(deriv,n0,time)

Η odeint αντιστοιχεί σε ένα μεγάλο κομμάτι κώδικα το οποίο είναι μία συνάρτηση και εμείς απλώς το χρησιμοποιούμε. Αυτό το κομμάτι κώδικα λύνει το πρόβλημα αρχικών τιμών για μία διαφορική εξίσωση (δεν θα δούμε εδώ πώς ακριβώς γίνεται αυτό).
Για να δουλέψει χρειάζεται τα εξής τρία ορίσματα

1. Την function deriv η οποία επιστρέφει την παράγωγο της συνάρτησης,
2. το n0 για την αρχική τιμή της συνάρτησης και
3. την σειρά τιμών για τον χρόνο time.

Το αποτέλεσμα το οποίο επιστρέφει η συνάρτηση odeint (στην μεταβλητή nuclei) είναι πάλι μία σειρά τιμών.

3ο κομμάτι του προγράμματος (γραφική παράσταση)

Όλες οι εντολές οι οποίες κατασκευάζουν ένα γράφημα περιέχονται σε μία βιβλιοθήκη της python το οποίο ονομάζεται matplotlib. Την έχουμε εισαγάγει με μία από την εντολές στην κορυφή του προγράμματος και της έχουμε δώσει το σύντομο όνομά plt.
Ας δούμε πώς κατασκευάζουμε ένα γράφημα. (Ένα μάθημα για γραφήματα μπορείτε να δείτε εδώ.)

```
plt.figure(figsize=(6,6))                # γίνεται εκκίνηση ενός νέου σχήματος
plt.xlabel(t,fontsize=20,labelpad=0)     # δίνεται τίτλος στον οριζόντιο άξονα
plt.ylabel(N,fontsize=20,labelpad=0)     # δίνεται τίτλος στον κάθετο άξονα

xmin = 0.0; xmax = 10.0
ymin = 0.0; ymax = 20.0
plt.axis([xmin,xmax,ymin,ymax])          # καθορίζονται τα όρια στους άξονες του γραφήματος

dx = 5.0; dy = 5.0
plt.xticks(plt.arange(xmin,xmax+dx,dx),fontsize=14)
plt.yticks(plt.arange(ymin,ymax+dy,dy),fontsize=14)
```

Θα τοποθετηθούν σημάδια στον οριζόντιο άξονα από την θέση xmin έως την xmax ανά διαστήματα ίσα με dx. Το μέγεθος της γραμματοσειράς θα είναι 14.

```
plt.plot(time,nuclei,'-ob')
plt.show()
```

Με την πρώτη εντολή γίνεται γραφική παράσταση των τιμών nuclei (οι οποίες υπολογίστηκαν από την συνάρτηση odeint) σαν συνάρτηση των τιμών time. Το γράφημα είναι με συνεχή (-) μπλε (b) γραμμή και με κυκλάκια (o). Το γράφημα εμφανίζεται στην οθόνη με την τελευταία εντολή.

Πακέτα (κορυφαίο κομμάτι του προγράμματος)

```
from scipy.integrate import odeint
from numpy import linspace
import matplotlib.pylab as plt        # for plotting commands
```

Οι εντολές αυτές οι οποίες μπαίνουν στην αρχή του προγράμματος θέτουν στην διάθεση του προγράμματος ορισμένα έτοιμα πακέτα. Π.χ., το πρώτο πακέτο θέτει στην διάθεσή μας την συνάρτηση odeint η οποία κάνει ολοκλήρωση του προβλήματος αρχικών τιμών. Το πακέρο matplotlib.pylab θέτει στην διάθεσή μας εργαλεία για σχεδιάση γραφικών

παραστάσεων.

Συνάρτηση

```
def deriv(y,t):              # return derivative of y
    return -rate*y
```

Είναι μία συνάρτηση η οποία δίνει το δεξιό μέλος της εξίσωσης την οποία θέλουμε να ολοκληρώσουμε (δηλαδή, την παράγωγο της συνάρτησης).
Βλέπουμε ότι το κομμάτι αυτό δεν ανήκει στο κύριο πρόγραμμα. Είναι ένα ξεχωριστό κομμάτι του προγράμματος και μπαίνει πριν το κύριο πρόγραμμα.

Σχόλια στον κώδικα: Όπου υπάρχει το σύμβολο # η συνέχεια της γραμμής αποτελεί σχόλιο, άρα την αγνοεί η python, (τέτοιες γραμμές είναι μόνο επεξηγήσεις προς τον άνθρωπο-αναγνώστη). Τα σχόλια είναι πολύ συχνά ουσιαστική βοήθεια. Για παράδειγμα, έφτιαξα αυτόν το κώδικα πριν καιρό αλλά έχω πλέον ξεχάσει τι ακριβώς είναι. Τα σχόλια που έγραψα τότε με βοηθούν τώρα να θυμηθώ. Επίσης, πρέπει να είναι βοηθητικά και για εσάς εφόσον βλέπετε τον κώδικα πρώτη φορά.

Κώδικας

```
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# It solves an initial value problem
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

from scipy.integrate import odeint
from numpy import linspace
import matplotlib.pylab as plt        # for plotting commands

rate = 0.1                        # parameter of population model

# right side of ODE
def deriv(y,t):                    # return derivative of y
   return -rate*y

# - - - Main program - - -

# input initial condition
n0 = input('Give initial condition N0: ')
```

```
n0 = float(n0)

# construct time interval to solve equation
t0 = input('Give initial time: ')
t0 = float(t0)
t1 = input('Give initial time: ')
t1 = float(t1)
ntime = 20
time = linspace(t0,t1,ntime)

# integrate ODE
nuclei = odeint(deriv,n0,time)

# - - - prepare graph
plt.figure(figsize=(6,6))
plt.xlabel('t',fontsize=20,labelpad=0)
plt.ylabel('N',fontsize=20,labelpad=0)

xmin = t0; xmax = 1.2*t1
ymin = 0.0; ymax = 1.2*max(nuclei)
plt.axis([xmin,xmax,ymin,ymax])

dx = 5.0; dy = 5.0
plt.xticks(plt.arange(xmin,xmax+dx,dx),fontsize=14)
plt.yticks(plt.arange(ymin,ymax+dy,dy),fontsize=14)

# plot a line
plt.plot(time,nuclei,'-ob')
plt.show()
```