# Contents



# 1 préparatifs

- On importe les librairies
- On fixe les valeurs des constantes physiques
- On saisie les tableaux de valeurs de température

```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt

#Constants
M = 29.0e-3
R = 8.31
P0 = 1.0e5
g0 = 9.8
RT = 6.4e3
pi = np.pi

#Altitude en km
zexp = np.array([0.0, 5.0, 10.0, 12.0, 20.0, 25.0, 30.0, 35.0, 40.0,
                 45.0, 48.0, 52.0, 55.0, 60.0, 65.0, 70.0, 75.0, 80.0, 84.0, 92.0, 95.0,100.0])

Texp = np.array([15.0, -18.0, -49.0, -56.0, -56.0, -51.0, -46.0, -37.0,
                 -22.0, -8.0, -2.0, -2.0, -7.0, -17.0, -33.0, -54.0, -65.0, -79.0, -86.0,-86.0, -81.0, -72.0])
```

Voici quelques premiers éléments de correction.

Les lignes de code seront susceptibles d'être remaniées et/ou commentées en fonction de vos retours.

## 2 Q1 - formule de l'interpolation - Tableau

Ce point a été traité en séance au tableau.

## 3 Q2 - CODE

```
def T(z, unite):
    z_km = z / 1000 # Conversion en km pour comparaison dans la liste
    alpha = 1 # Valeur par défaut pour la conversion en K
    if unite == 'C':
        alpha = 0 # Pas de décalage pour la température en °C
    i = 0
    while z_km > zexp[i + 1]: # Recherche de l'indice i
        i = i + 1
    temperature = alpha*273 + Texp[i] + (z_km - zexp[i])/(zexp[i + 1] - zexp[i])*(Texp[i + 1] - Texp[i]) # Interpolation linéair
    return temperature
```

## 4 Q3 CODE

```
N = 1000 # Nombre de points
zmax = 100.0e3 # Altitude max (en m)
dz = zmax/(N - 1) # Pas spatial (en m)
zatm = np.array([k*dz for k in range(N)]) # Altitudes
Tatm = np.array([T(zatm[k], 'C') for k in range(N)]) # Températures
```

## 5 Q4 CODE

```
fig, ax = plt.subplots()
ax.plot( Tatm,zatm)
ax.plot( Tatm,zatm)
plt.savefig("graph_Q4.png")
```

## 6 Q5 TABLEAU

## 7 Q6 TABLEAU

## 8 Q7

```
def g(z): # Champ de pesanteur
    return g0 * RT**2 / (RT + z)**2
```

```
# Calcul du champ de pression par la méthode d'Euler
Patm = [P0] # Initialisation
for k in range(N - 1): # Il reste N - 1 termes à calculer
```

```
    Patm.append(Patm[k] - M*g(zatm[k])*Patm[k]*dz/(R*T(zatm[k], 'K')))
Patm = np.array(Patm) # Conversion en tableau
```

```
fig, ax = plt.subplots()
ax.plot( Patm,zatm)
plt.savefig("graph_Q7.png")
```

# 9 Q8 CODE

```
def masse_atm(z): # Calcul de la masse d'air jusqu'à l'altitude z
    masse = 0
    k = 0
    while zatm[k] < z: # On arrête le calcul à l'altitude z
        dm = 4*np.pi*(RT + z)**2*M*Patm[k]/(R*T(zatm[k], 'K'))*dz
        masse = masse + dm
        k = k + 1
    return masse
```

```
mtot = masse_atm(100e3) # Masse d'air dans l'atmosphère terrestre
print('Masse de l\'atmosphère :', mtot, 'kg')
```

```
mtropo = masse_atm(12e3) # Masse d'air dans la troposphère
print('Proportion d\'air dans la troposphère :', mtropo/mtot)
```

# 10 Q9 TABLEAU

```
Patm2 = [P0]
for k in range(N - 1):
    Patm2.append(Patm2[k] - M*g0*Patm2[k]*dz/(R*T(zatm[k], 'K')))
Patm2 = np.array(Patm2)
ecart1 = 100 * abs(Patm - Patm2) / Patm # Ecart relatif
```

```
fig, ax = plt.subplots()
ax.plot( ecart1,zatm)
plt.savefig("graph_Q9.png")
```

# 11 Q10 CODE

```
Piso = [P0]
for k in range(N - 1):
    Piso.append(Piso[k] - M*g0*Piso[k]*dz/(R*T(0, 'K')))
Piso = np.array(Piso)
ecart2 = 100 * abs(Piso - Patm) / Patm # Ecart relatif
```

```
fig, ax = plt.subplots()
ax.plot( ecart2,zatm)
plt.savefig("graph_Q10.png")
```

## 12 Q11 TABLEAU

## 13 Q12 CODE:TABLEAU

```
ztropo, Ttropo = [], [] # Initialisation des listes
k = 0
while zatm[k] < 10e3: # On sélectionne les données jusqu'à 10km
    ztropo.append(zatm[k])
    Ttropo.append(T(zatm[k], 'K'))
    k = k + 1 # NB^^C:: On a pris en fait 1 point sur 5 pour le graphe
# Régression linéaire T(z)=a*z+b
a, b = np.polyfit(ztropo, Ttropo, 1) # Calcul de la régression linéaire
Tlin = [a*z + b for z in ztropo] # Modèle linéaire de la température


print(a,b)
```

```
Pgradient = [P0]
for k in range(len(ztropo) - 1):
    Pgradient.append(Pgradient[k] - M*g0*Pgradient[k]*dz/(R*(a*zatm[k] + b)))
Pgradient = np.array(Pgradient)
ecart3 = 100 * abs(Pgradient - Patm) / Patm
```

```
fig, ax = plt.subplots()
ax.plot( ecart3,zatm)
plt.savefig("graph_Q12.png")
```