PYTHON : MÉTHODES D'APPROXIMATION NUMÉRIQUES

1 RÉSOLUTION APPROCHÉE D'UNE ÉQUATION DIFFÉRENTIELLE PAR LA MÉTHODE D'EULER

Soit une équation différentielle (pas nécessairement linéaire) sur $[a,b]$. $y'(t)=\varphi(y(t),t)$.

On divise l'intervalle $[a,b]$ en n intervalles de même longueur, et on pose $\forall k\in[\![0,n]\!],\ t_k=a+k\frac{b-a}{n}$

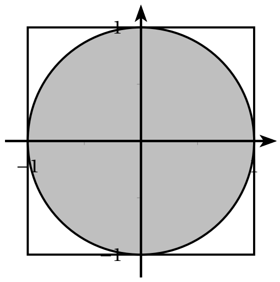
On approxime $y'(t)$ par $y'(t)\approx\frac{y(t_{k+1})-y(t_k)}{t_{k+1}-t_k}=n\frac{y(t_{k+1})-y(t_k)}{b-a}$ et on pose $y_k=y(t_k)$.

1) Donner la valeur de y_{k+1} en fonction de y_k et de k.

2) En déduire un algorithme qui donne y_k en fonction de y_0.
Programmer la fonction **euler**(`phi, a,b,y0,n`) qui trace la courbe de la solution approchée. Tester avec une équation différentielle linéaire simple du premier ordre et comparer avec la solution exacte.

2 APPROXIMATION DE π PAR LA MÉTHODE DE MONTE CARLO

On considère le cercle de centre 0 et de rayon 1. On se place dans le carré $[-1,1]\times[-1,1]$. Si on choisit au hasard un grand nombre de points dans ce carré, alors la proportion de points se trouvant dans le disque par rapport à ceux se trouvant dans le carré donne une valeur approchée de l'aire du disque sur l'aire du carrée : $\frac{\pi}{4}$



1) Écrire une fonction Python **disque(x,y)** qui teste si un point de coordonnées (x,y) se trouve dans le disque (renvoie `True` ou `False`).

2) Écrire une fonction Python **monteCarlo(n)** qui choisit successivement n points aléatoires dans le carré et renvoie la proportion de ceux qui se trouvent dans le disque.

3) Calculer une valeur approchée de π.

3 APPROXIMATION DE π PAR LES SOMMES DE RIEMANN

1) Positionnement du problème

(a) Soit $f:x\mapsto\sqrt{1-x^2}$, donner le domaine de définition de f et tracer sa courbe représentative en justifiant.

(b) À l'aide de la définition, calculer $I=\displaystyle\int_0^1 f(x)\,\mathrm{d}x$.

(c) Donner la somme de Riemann qui converge vers I et un majorant de l'erreur $\left|R_n(f)-I\right|$ en fonction de n.

2) Programmation de la méthode des rectangles

- **Pour ceux qui ont des difficultés :**

  (a) Définir une fonction **rectangles(n)** qui renvoie la valeur de $\frac{1}{n}\sum_{k=0}^{n-1}\sqrt{1-\frac{k^2}{n^2}}$.

  (b) Utiliser cette fonction pour touver une valeur approchée de π.

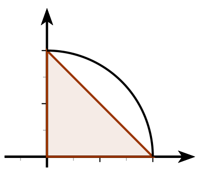
- **Pour ceux qui sont plus à l'aise :**

  (a) Définir une fonction **rectangles**(`phi,a,b,n`) qui prend en argument une fonction `phi`, deux réels `a` et `b`, et un entier non nul `n` et qui renvoie $R_n(\varphi)$.

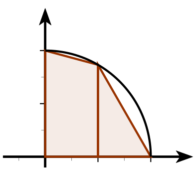
  (b) Définir une fonction **approx(phi, a,b,eps)** qui prend en argument une fonction `phi`, deux réels `a` et `b`, et un troisième réel `eps`, et qui renvoie une approximation de $\int_a^b$ `phi` à `eps` près.

  (c) Appliquer cette fonction à f et en déduire une valeur approchée de π à 10^{-4} près.

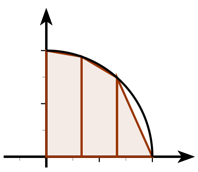
3) Amélioration : méthode des trapèzes
Au lieu d'approximer l'aire par des rectangles, on l'approxime par des trapèzes (voir la figure).
Programmer la fonction **trapezes**(`phi,a,b,n`) qui prend en argument une fonction `phi`, deux réels `a` et `b`, et un entier naturel `n` non nul et qui renvoie l'aire correspondant à la n-ième étape.
Utiliser cette fonction pour donner une approximation de π.



Étape 1



Étape 2



Étape 3