Ce TP est inspiré de la partie algorithmique de CCP-TSI-2002-2.

1 Présentation du problème

Le but est ici de calculer la longueur approchée d'une courbe plane entre deux points.
Le paramètre, appelé t, varie entre t_0 et t_1.
Dans tous les cas, on utilisera $n+1$ points de la courbe, appelés Ai de $i=0$ à n, qui correspondent à un découpage de $[t_0, t_1]$ en n segments égaux.
On programme 2 stratégies :

1) Une procédure `Lmin` qui calcule la somme des longueurs des segments reliant les points A_i indiqués.
Si la concavité de la courbe ne change pas, cette procédure calcule à l'évidence un *minorant* de la longueur de la courbe.

2) Une procédure `Lmaj` qui calcule aussi une somme de longueurs de segments.
Les points extrêmes sont A_0 et A_n, mais les points intermédiaires choisis, appelés B_i de $i=1$ à n, sont les points d'intersection des tangentes à la courbe en A_{i-1} et A_i.
Il y a donc ici $n+1$ segments ! Si la concavité de la courbe ne change pas, cette procédure calcule le plus souvent un *majorant* de la longueur de la courbe.

Faire une figure illustrant la situation avec $n=3$.

2 Procédure `Lmin`

2.1 `Lmin` pour une paramétrisation en cartésiennes

Ecrire une procédure `Lmin`.
Celle ci a 5 paramètres : $t \mapsto x(t)$, $t \mapsto y(t)$, t_0, t_1 et n ; par ailleurs, elle donne une valeur approchée de la longueur cherchée.

2.2 Petite idée de l'erreur commise

Ecrire une procédure `LminApp`.
Celle ci a 5 paramètres : $t \mapsto x(t)$, $t \mapsto y(t)$, t_0, t_1 et e.
C'est la première valeur de $\texttt{Lmin}\left(x, y, t_0, t_1, 2^{p+1}\right)$ telle que $\left|\texttt{Lmin}\,(x, y, t_0, t_1, 2^p) - \texttt{Lmin}\left(x, y, t_0, t_1, 2^{p+1}\right)\right| \leqslant e$.

2.3 En polaires

Ecrire des procédures `LminPol` et `LminPolApp` qui font la même chose pour une courbe en polaires ; il y a donc un paramètre de moins !

3 Procédure Lmaj

3.1 Les points B_i

Ecrire 2 procédure `Bx` et `By` qui calculent les coordonnées du point B_i.
Celles ci ont paramètres : $t \mapsto x(t)$, $t \mapsto y(t)$, t_0, t_1, n et i.

3.2 `Lmaj` pour une paramétrisation en cartésiennes

Ecrire une procédure `Lmaj`.
Celle ci a 5 paramètres : $t \mapsto x(t)$, $t \mapsto y(t)$, t_0, t_1 et n ; par ailleurs, elle donne une valeur approchée de la longueur cherchée.

3.3 Petite idée de l'erreur commise

Ecrire une procédure `LmajApp`.
Celle ci a 5 paramètres : $t \mapsto x(t)$, $t \mapsto y(t)$, t_0, t_1 et e.
C'est la première valeur de $\texttt{Lmaj}\left(x, y, t_0, t_1, 2^{p+1}\right)$ telle que $\left|\texttt{Lmaj}\,(x, y, t_0, t_1, 2^p) - \texttt{Lmaj}\left(x, y, t_0, t_1, 2^{p+1}\right)\right| \leqslant e.$.

4 Comparaison des deux procédés

Ecrire une procédure `L` de paramètres $t \mapsto x(t)$, $t \mapsto y(t)$, t_0, t_1 et e qui calcule `Lmin` et `Lmaj` telles que la différence des valeurs de $\texttt{Lmin}(x, y, t_0, t_1, 2^p)$ et $\texttt{Lmaj}(x, y, t_0, t_1, 2^p)$ soit en valeur absolue inférieure à e.

5 Application

En cartésienne, on essayera et appliquera ces procédures à un demi cercle de rayon 1 ; on comparera les résultats avec une valeur approchée de π.
En polaires, on le fera avec le cercle de rayon $\dfrac{1}{2}$, d'équation $\rho = \cos\theta$, c'est à dire pour θ variant entre 0 et π ; on comparera les résultats avec une valeur approchée de la longueur exacte, à savoir : π.
La première valeur de n utilisée sera 10.
La première valeur de e sera 0.01.