Polynômes : la méthode de Horner

Avez vous remarqué que $7x^4 + 11x^3 + 3x^2 + 5x +2 = (((7 x + 11) x + 3) x + 5) x + 2$?

Cela revient à effectuer les opérations successives suivantes :

- prendre le coefficient de x^4
- multiplier par x
- ajouter le coefficient de x^3
- multiplier par x
- ajouter le coefficient de x^2
- multiplier par x
- ajouter le coefficient de x^1
- multiplier par x
- ajouter le coefficient de x^0

Nous avons donc 4 multiplications à effectuer, et pas 4 + 3 + 2 + 1 multiplications en absence de réarrangement. De plus, on répète systématiquement l'alternance des opérations “multiplier par x” et “ajouter un coefficient”.

Cette façon d'évaluer le polynôme s'appelle la méthode de Horner et est particulièrement efficace lorsque n est grand. La méthode débouche sur un algorithme facile à écrire sous forme d'une instruction de répétition.

<sxh python; title : poly06-horner.py> #!/usr/bin/python # -*- coding: UTF-8 -*- “”“ écriture d'un programme pour évaluer des polynomes ”“” from math import *

def polyeval(x,a):

```
    """application de l'agorithme de Horner
    cf. http://fr.wikipedia.org/wiki/M%C3%A9thode_de_Ruffini-Horner
    """
    n=len(a)-1
    p=a[n]
    for i in range(n-1,-1,-1):
        p=p*x+a[i]
    return p
```

x=2. # x particulier a=[1,1,1,1,1,1,1,1,1,1,1] # coefficients particuliers print polyeval(x,a) # on doit obtenir un exposant de deux moins un

varx=0.5 varcoef=[1.,2.,3.,4.,5.,6.,7.,8.,9.,10.] print polyeval(varx,varcoef)

for j in range(0,11,1):

```
    vax=float(j)*0.1
    rep=sin(polyeval(vax,varcoef))
```

```
  print rep
```

</sxh>

Écrivons à présent d'autres fonctions qui seront très utiles. Pour commencer :

- la fonction de multiplication d'un polynome pas un scalaire
- la fonction d'addition de deux polynomes

Proposition à la page suivante !

From:
https://dvillers.umons.ac.be/wiki/ - **Didier Villers, UMONS - wiki**

Permanent link:
https://dvillers.umons.ac.be/wiki/teaching:progappchim:polynomes-6?rev=1352393015

Last update: **2012/11/08 17:43**