Projet ISN

Dérivée formelle

Raphaël Letourneur

I – Présentation du projet et du contexte

Lors de notre année de Terminale, nous avons suivi des cours de spécialité en informatique et science du numérique. Durant cet apprentissage, nous avons donc dut, par groupe, réaliser un projet nous permettant de mettre en pratique ce que nous avions appris.

Pour ce projet, moi, ainsi que mon camarade avions décidé de créer un programme permettant de dériver une fonction mathématique de niveau Terminal Scientifique. Pour cela, nous avons utilisé le langage de programmation « Python ». Afin de réaliser notre objectif qui est de pouvoir dériver n'importe quelle fonction en quelques fraction de secondes, nous avons du passer par plusieurs version.

II – Les différentes versions

Comme mentionné au-dessus, nous avons utilisé plusieurs version afin d'atteindre notre but. En effet, nous nous sommes fixés des objectifs intermédiaires afin de ne pas se perdre dans la création de celui-ci ainsi que d'avoir des programmes intermédiaires fonctionnels. Ceci nous a également permis d'avoir un programme fonctionnel quasiment depuis la début de sa création nous permettant de relever quelques « bugs » pour le projet final auquel nous aurions pas pensé sans effectuer une multitude de tests.

1) Première version

La première version de notre programme permettait de dériver des fonctions polynômes. Nous avions commencé par faire un algorithme permettant de séparer la fonction à chaque opérateur tout en gardant les signes afin de pouvoir dériver chaque groupe puis les réassembler pour afficher le résultat à l'utilisateur. Ensuite nous avons fait un algorithme permettant de relever l'exposant via analyse syntaxique et le remplacer par sa valeur soustraite de un. Ensuite l'algorithme effectuait la multiplication entre l'exposant gardé en mémoire et le coefficient devant le « x » en question. Puis remplacer par la nouvelle valeur.

C'est l'application de la formule $x^n{}' = nx^{n-1}$

Exemple : $(x^2)' = 2x$; $(3x^4+12x^2-5x)' = 12x^3+24x-5$.

Remarque :

- Afin d'écrire l'exposant, l'utilisateur doit saisir « ^() » en ajoutant la valeur de l'exposant entre parenthèses.
- Nous avions fais en sorte que le programme n'affiche pas « ^(1) » ni « ^(0) » (écrit directement « x » au lieu de « x(^1) »)
- Remplace directement les « +- » par des « - »

2) Deuxième version

Après avoir élaboré le début de la structure du programme et obtenu un commencement fonctionnel, nous avons rajouté les fonctions nécessitant de dériver des parties de la fonction (Multiplication, division, ln(u), e(u), ^(u) ainsi que la racine carré qui se saisi en écrivant « sqrt(u) »)

Petit rappel des dérivés :

Dérivées et opérations Dans ce formulaire, u et v sont des fonctions		
Opérations sur les fonctions	Dérivées	Conditions
$f = u + v$	$f' = u' + v'$	u et v dérivables sur un intervalle I
$f = uv$	$f' = u'v + v'u$	u et v dérivables sur un intervalle I
$f = \frac{u}{v}$	$f' = \frac{u'v - v'u}{v^2}$	u et v dérivable sur un intervalle I et v ne s'annule pas sur cet intervalle I
$f = u^n$	$f' = \alpha u' u^{n-1}$	selon les valeurs de α
$f = \sqrt{u}$	$f' = \frac{u'}{2\sqrt{u}}$	u dérivable sur un intervalle I et $u > 0$
$f = e^u$	$f' = u' \times e^u$	u dérivable sur un intervalle I
$f = \ln u$	$f' = \frac{u'}{u}$	u dérivable sur un intervalle I et $u > 0$

Dans cette version, l'algorithme cherchais la présence de parenthèses et regardais ce qui se trouvait juste avant la parenthèse ouvrante afin de savoir si c'était un logarithme népérien, un exponentiel, une multiplication, une division, ou une racine carrée. Ensuite, il remplaçait le bloc partant du début de l'élément repéré jusqu'à la parenthèse fermente par le résultat de la formule de la dérivée approprié tout en faisant attention au coefficient avant les éléments clés.

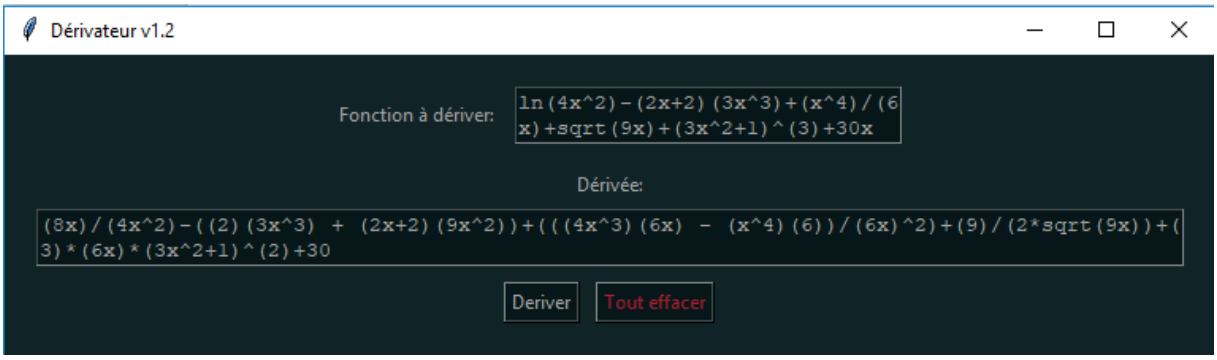
Cette version était un bon début mais comportait beaucoup de problème dans le cas où plusieurs de ces dérivées étaient présente dans la fonction.

3) *Troisième version (Final)*

Cette version est la dernière version, celle que nous allons proposer à notre oral et que vous avez en pièce-jointe. Il est vrai qu'elle est encore loin d'être parfaite mais est déjà très fonctionnel.

Dans cette version, nous avons rajouté un système de récurrence permettant de dériver des fonctions composées d'autres fonctions composées. Cela nous a permis de faire en sorte que notre algorithme puisse dériver des fonctions de taille infini avec des imbrications et des parenthèses un peu partout.

Ex : f(x) = ln(4x^2)-(2x+2)(3x^3)+(x^4)/(6x)+sqrt(9x)+(3x^2+1)^(3)+30x



Comme vous avez pu le voir, nous avons également rajouté une interface graphique afin de rendre l'utilisation de notre programme encore plus facile pour l'utilisateur.

4) *Amélioration possible*

Comme dit plus haut, notre programme bien que fonctionnellement achevé, peut encore être améliorer. En effet, on constate que, notre dérivateur ne simplifie pas le résultat.

De plus, un affichage sous forme de fraction pourrait permettre une meilleure lisibilité du résultat.

III – Les différentes fonctions

1) *Deriver_simple() ; fusion()*

La première fonction que nous avons fait est une fonction qui permet de dériver un polynôme, elle prend un paramètre qui est le polynôme à dériver. En effet,

de nombreuse fonction sont composé de polynôme. Cette fonction permet de chercher les « x » et d'appliquer la formule :

$\alpha x^n{}' = \alpha n x^{x-1}$ Avec $\alpha \epsilon R$

Cette fonction remplace aussi les « x^1 » par « x ».

Le début de la fonction permet de remplacer les « – » par des « +- ». Cela permet de séparer la fonction ensuite à chaque « + » et ainsi de couper à chaque « + » et chaque « - » tout en conservant le signe.

Enfin, cette fonction retourne une liste de la dérivée de tous les éléments qui seront ensuite fusionné par la fonction fusion().

La fonction fusion() prend donc en paramètre une liste à réassembler. La fonction met donc toutes les dérivées à la suite en faisant attention au signes avec une mise en affichage propre (pas de double signe tel que +-, -- ou ++)

2) *Decompose()*

La fonction décompose() prend elle aussi un seul paramètre qui ai la fonction saisi par l'utilisateur. Elle permet de chercher toute les opérations complexes (multiplication, division) ainsi que les fonctions spéciales tel que les logarithmes népérien (« ln() »), l'exponentiel (« e() ») etc…

Elle permet donc d'organiser une liste qui sera ensuite interprété par la fonction deriver_complexe() afin de savoir comment gérer chaque chaine de caractère et de savoir quelle formule il faut appliquer en decoupant correctement la fonction

Elle retourne donc une liste de chaque terme à dériver avec une information sur le type (signe des multiplication, division avant les termes ou découpage avec ln(), e() ou ^())

3) *Deriver_complexe()*

La fonction deriver_complexe() est la fonction principale, elle prend en paramètre la fonction que l'utilisateur doit dériver et appliquer ensuite les fonctions définis au-dessus. Ensuite, elle utilise la sortie de la fonction decompose() afin de savoir ce qu'elle doit faire.

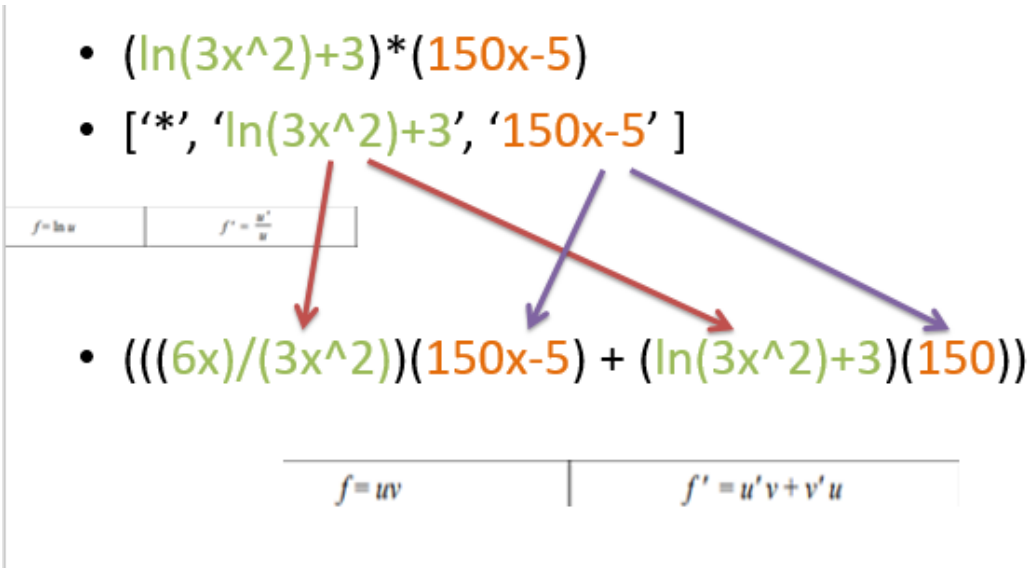
Pour les opérations (multiplication, division), la liste est composée en premier de l'opérateur puis l'algorithme applique la formule liés à cette opération

$f = uv$	$f' = u'v + v'u$
$f = \frac{u}{v}$	$f' = \frac{u'v - v'u}{v^2}$

Ensuite, l'algorithme regarde s'il y'en a, ce qu'il y'a avant les parenthèses ouvrantes (repère « ln », « e » etc…) et applique les formules revu dans le tableau plus haut.

S'il n'y a pas de parenthèses, c'est qu'il s'agit d'un polynôme donc la fonction deriver_simple() est appliquée.

Ex :



IV – Amélioration possible

Bien que nous avons réussi à finir notre programme à temps, nous sommes conscients qu'il est encore loin d'être parfait. En effet, nous avions encore de nombreuse idées d'ajout et d'amélioration.

- Simplification des résultats
- Ajout des fonctions trigonométrique
- Affichage sous forme de fraction

- Ajout de module d’étude de fonction tel que les tableaux de variation ou les études de signe.

V – Ce que j’ai appris

Ayant déjà des connaissances en programmation python, nos cours ainsi que la réalisation de ce projet m’a surtout appris à m’organiser pour le travail de groupe ainsi que pratiqué car je disposais de beaucoup de connaissance théorique mais n’avais jamais fini mes projets dans ce langage m’étant souvent découragé lors de la partie graphique.