Enregistrements

Option informatique - TP nº 2.8 - Olivier Reynet

A Nombres complexes

On souhaite calculer des fonctions de nombres complexes. On dispose du type **enregistrement** OCaml suivant :

```
type complexe = {mutable re : float; mutable im : float}
```

---

ce qui signifie qu'un nombre complexe est une paire de flottants.

Le mot-clef `mutable` signifie qu'on peut modifier l'enregistrement.

A1. Écrire les fonctions de signature `ez : complexe -> float` et `im : complexe -> float` qui renvoie respectivement la partie réelle et la partie imaginaire d'un complexe.

A2. Écrire une fonction de signature `conjugue : complexe -> unit` qui modifie en place un complexe pour qu'il devienne son conjugué.

A3. Écrire une fonction de signature `mise_a_zero : complexe -> unit` qui met à zéro en place les parties réelles et imaginaires du nombre complexe.

A4. Écrire une fonction de signature `norme : complexe -> float` qui renvoie norme d'un nombre complexe.

A5. Écrire une fonction de signature `add : complexe -> complexe -> complexe` qui renvoie un nouveau nombre complexe valant la somme des deux paramètres.

On peut naturellement décliner toutes les opérations.

B Autodifférentiation

L'autodifférentiation est un concept qui permet d'évaluer simultanément une fonction et sa dérivée en un point. L'idée est simple et repose sur les définitions de la dérivée :

- La dérivée d'une constante est nulle.
- La dérivée d'une varaible vaut 1.
- La dérviée d'une somme (d'une différence) est la somme (différence) des dérivées.
- La dérivée d'un produit (d'une division) est le résultat d'une somme et d'une multiplication.

■ **Exemple 1 — Autodifférentiation de** $f(x) = 3x^3 + 2x - 5$**.** Soit le tuple $(x, 1)$ représentant la varaible x et sa dérivée. Pour calculer f en 2 on procède comme d'habitude. Par contre, on évalue sa dérivée en 2 suivant les règles de calcul des dérivées.

- -5 résulte en $(-5, 0)$ puisque la dérivée d'une constante est nulle.
- $2x$ résulte en $(4, 2)$, car $(ab)' = a'b + ab' = 2$ avec $a = 2$, $a' = 0, b = x$ et $b' = 1$.
- $3x^3$ résulte en $(24, 36)$. On procède par multiplications successives de $(2, 1)$ puis multiplication par la constante $(3, 0)$. Cela donne : $(2, 1) \to (4, 2) \to (8, 24)$ puis $(24, 36)$.

> Au final, on trouve $(24,36)+(4,2)+(-5,0)=(23,38)$, qui vaut bien $f'(x)=9x^2+2$, c'est-à-dire $36+2=38$.

En OCaml, on crée un type pour représenter la valeur d'une fonction en un point et sa dérivée comme suit :

```
type duo = { valeur : float; deriv : float}
```

B6. Écrire une fonction de signature `var : float -> duo` qui crée un type duo de dérivée 1 et de valeur donnée par le flottant.

B7. Écrire une fonction de signature `const : float -> duo` qui crée une constante de type duo `c` de dérivée 0.

B8. Écrire les fonctions de signature :

- `add : duo -> duo -> duo`
- `sub : duo -> duo -> duo`
- `mul : duo -> duo -> duo`
- `mul : duo -> duo -> duo`

qui implémentent les opérations arithmétiques sur des types `duo`, normalement sur `valeur` et selon les règles de la dérivation pour `derivee`.

B9. En utilisant l'algorithme d'exponentiation rapide, écrire une fonction de signature `pow : duo -> int -> duo` qui calcule la puissande d'un type `duo`.

Pour faciler l'écriture des expressions, on se dote des fonctions suivantes :

```
(* Opérateurs infixes pour faciliter l'écriture *)
let ( +@ ) = add
let ( -@ ) = sub
let ( *@ ) = mul
let ( /@ ) = div
let ( **@ ) a n = pow a n
```

Ainsi on peut écrire rapirement $f(x)=3x^3+2x-5$:

```
let polynome x_val =
    let x = var x_val in
    let term1 = const 3.0 *@ (x **@ 3) in
    let term2 = const 2.0 *@ x in
    let term3 = const 5.0 in
  term1 +@ term2 -@ term3;;
```

et calculer le polynôme en 2 et sa dérivée :

```
polynome 2.
```