Lucie Delaire - Marine Dos

Algèbre de Grassmann- Cayley
Projet de mathématiques IMAC2
Développement d'une librairie C++

6 janvier 2014

1 Introduction

«Ce projet consiste à implémenter une bibliothèque en C++ permettant d'utiliser l'algèbre de Grassmann-Cayley. Il s'agit d'un algèbre permettant de manipuler facilement les points, les droites et les plans dans un espace projectif.»

Nous avons donc créé les structures relatives à l'algèbre de Grassmann-Cayley ainsi que les méthodes nécessaires au bon fonctionnement de l'algèbre.

2 Présentation de la bibliothèque

2.1 Éléments demandés, codés et qui fonctionnent

Bibliothèque
- Programme d'exemple d'utilisation de la librairie
- Réalisée en C++
- Utilisation d'un makefile générée par cmake

Ensemble des structures
- Scalaire et anti quadvecteur
- Vecteur et anti trivecteur
- Bivecteur et anti bivecteur
- Trivecteur et anti vecteur
- Quadvecteur et anti scalaire

Fonctionnalités pour chaque structure
- Constructeur par défaut
- Constructeur par recopie
- Constructeur avec paramètre(s)
- Opérateur d'affectation =
- Opérateur wedge ^
- Convertisseur en base duale ~
- Opérateur « pour l'affichage
- Opérateur « pour l'initialisation (hérité de Eigen)

Programme d'exemple présentant les fonctionnalités

2.2 Éléments demandés, codés et mais qui ne fonctionnent pas

Aucun

2.3 Éléments demandés mais non codés

Aucun

2.4 Éléments non demandés, codés et qui fonctionnent

Fonctionnalité pour chaque structure
- Opérateur de comparaison ==
- Opérateur de comparaison =!
- Opérateur anti wedge |
- Accès à une valeur grâce au nom de la composante
- Création des vecteurs u et v pour les droites de Plücker
- Création d'une droite de Plücker grâce à 2 points
- Création d'un plan grâce à 3 points

2.5 Éléments non demandés et non codés ou qui ne fonctionnent pas

Visualisation des éléments (non codé) Calcul du point d'une droite de Plücker le plus proche de l'origine (non codé) Caclcul de la distance entre une droite de Plücker et un point

3 Structures de données

- Scalaire
  - Comporte une donnée propre
  - Composante dans l'ordre : 1
- Vecteur
  - Hérite de Eigen::Vector4d
  - Composante dans l'ordre : e_1, e_2, e_3, e_4
- Bivecteur
  - Hérite de Eigen::VectorXd
  - Composante dans l'ordre : e_{12}, e_{13}, e_{14}, e_{23}, e_{24}, e_{34}
- Trivecteur
  - Hérite de Eigen::Vector4d
  - Composante dans l'ordre : e_{123}, e_{124}, e_{134}, e_{234}
- Quadvecteur
  - Comporte une donnée propre
  - Composante dans l'ordre : e_{1234}
- Anti quadvecteur
  - Comporte une donnée propre
  - Composante dans l'ordre : $\bar{e}_{1234}$
- Anti trivecteur
  - Hérite de Eigen::Vector4d
  - Composante dans l'ordre : $\bar{e}_{123}$, $\bar{e}_{124}$, $\bar{e}_{134}$, $\bar{e}_{234}$
- Anti bivecteur
  - Hérite de Eigen::VectorXd
  - Composante dans l'ordre : $\bar{e}_{12}$, $\bar{e}_{13}$, $\bar{e}_{14}$, $\bar{e}_{23}$, $\bar{e}_{24}$, $\bar{e}_{34}$
- Anti vecteur
  - Hérite de Eigen::Vector4d

  - Composante dans l'ordre : $\bar{e}_1$, $\bar{e}_2$, $\bar{e}_3$, $\bar{e}_4$
- Anti scalaire
  - Comporte une donnée propre
  - Composante dans l'ordre : $\bar{1}$

4 Fonctionnalités

4.1 Opérateur wedge ˆ

L'opérateur wedge se défini en fonction des grades des éléments. Dans $\mathbb{P}^3$, le grade maximal est 4. Il est donc possible de faire un wedge entre 2 élèments seulement si la somme des grades est inférieur ou égale à 4.

Pour trouver comment calculer un wedge, nous avons choisi de développer sur papier le résultat pour chaque éléments de l'algèbre de Grassmann-Cayley puis d'appliquer en code la formule trouvée.

Effectuer l'opération wedge dans un sens ou dans l'autre ne reviens pas à la même chose puisque $a\hat{}b = -b\hat{}a$. Afin de pouvoir disposer des opérateurs wedge dans les deux sens, nous n'en n'avons défini qu'un. Si on veux faire le wedge dans l'autre sens, la méthode appelle le wedge dans le premier sens puis change le signe de chaque composante.

4.2 Opérateur de conversion en base duale ˜

L'opérateur ˜permet la transformation en base duale des élèments de l'algèbre de Grassmann-Cayley. Les transformations se font dans les deux sens.

Pour trouver la formule de conversion en base duale, nous avons choisi de développer sur papier le résultat pour chaque éléments de l'algèbre de Grassmann-Cayley puis d'appliquer en code la formule trouvée. La difficulté pour développer ce calcul

4.3 Opérateur anti-wedge |

L'opérateur | permet de calculer des anti-wedge. Il est possible de calculer un anti-wedge si la différence de grade entre le plus grand des k-blade et le plus petit est positive.

Calculer un antiwedge reviens à faire un wedge dans la base duale de l'élément. Dans la logique du code, le calcul d'un anti wedge converti chaque élèment dans sa base duale, fais le wedge entre le deux et de renvoie le résultat. Pour cela, on réutilise donc les méthodes décrites ci-dessus.

4.4 Opérateur « pour l'affichage

Lors de l'affichage d'une structure de l'algèbre de Grassmann-Cayley, on affiche pour chaque valeur la composante associée. On n'affiche pas la composante du scalaire car elle est simple.

Les scalaires, les quad vecteurs, les anti scalaires et les anti quadvecteurs sont affichés sans crochets alors que les autres structures comportent des crochets pour mieux signifier la présence d'un ensemble de valeurs.

4.5 Opérateurs de comparaison == et !=

En générale, les méthodes de comparaison appellent la méthode de comparaison de Eigen. Pour les scalaires, les quad vecteurs, les anti scalaires et les anti quadvecteurs, les opérateurs comparent les données membres de chacunes des structures.

Cependant, au sens de l'algèbre de Grassmann-Cayley, les vecteurs x ($x \in \mathbb{P}^3$) et αx ($\alpha \in \mathbb{P}^*$) représentent la même entité. La méthode de comparaison des vecteurs est donc différente. On calcule un facteur grâce à la valeur de la première composante. Puis, on compare les valeurs des 3 autres composantes à un facteur près.

4.6 Accès aux valeurs par le nom des composantes

Pour faciliter l'accès des structures, on peut récupérer les valeurs en connaisant le nom de la composante. La méthode permet de retourner la valeur en fonction de la position de la composante.

4.7 Création des vecteurs u et v pour les droites de Plücker

Les droites de Plücker peuvent être définies par deux vecteurs :
- u : vecteur support de la droite
- v : moment de la droite

Les vecteurs sont créés grâce aux composantes de la droites de Plücker tel que :

$$u = (l_3, l_5, l_6)$$

$$v = (-l_4, l_2, -l_1)$$

4.8 Création d'une droite de Plücker grâce à 2 points

Une droite de Plücker correspond à la droite passant par 2 point. Celle-ci peut être calculé grâce à un wedge des deux points. On propose à l'utilisateur de créer une droite en passant en paramètre les deux points et en lui retournant la droite passant par ces 2 ponts.

4.9 Création d'un plan grâce à 3 points

Une plan peut être calculé grâce à un wedge entre trois points. On propose à l'utilisateur de créer un plan en passant en paramètre les trois points et en lui retournant le plan créé par ces points.

5 Problèmes rencontrés et solutions

5.1 Compréhension des wedges

Pour comprendre le calcul effectué par le wedge, nous avons développé le calcul à la main sur papier. Pour les vecteurs, nous avons réussi à trouver un algorithme de calcul du wedge. Pour les autres structures, nous avons simplement calculé la valeur des composantes que nous stockons dans la nouvelle structure.

Data: un vecteur
Result: un bivecteur
`// Déclaration du bivecteur`
for *i allant de 0 à taille d'un vecteur* **do**
  for *j allant de i+1 à taille d'un vecteur* **do**
    composante ième du bivecteur = (composante ième de actuel * composante jème de l'autre) - (composante jème de actuel * composante ième de l'autre);
  end
end
Retour du bivecteur ;

Algorithm 1: Wedge de deux vecteurs

Data: un vecteur
Result: un trivecteur
`// Déclaration du trivecteur`
`// Calcul des 4 composantes`
trivector[0] = bivecteur[0][0]*vecteur[2] - bivecteur[0][1]*vecteur[1] + bivecteur[0][3]*vecteur[0];
...;
Retour du bivecteur ;

Algorithm 2: Wedge d'un bivecteur et d'un vecteur

5.2 Vérification des résultats

Les résulats des différents algorithmes n'ont pas pu être tous vérifiés. Une représentation graphique aurait été appréciée pour vérifier les résulats. Par exemple, une droite de Plücker créer à partir de deux points passe-t-elle bien par les deux points en question ?