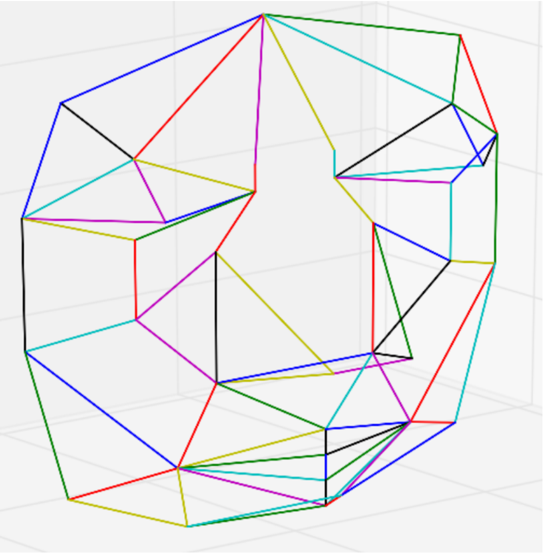
Visualisation d'objets de grande dimension

Olivier Roupin

Contexte et Objectifs

Visualisation d'objets

- D : dimension de départ < 100 dans notre étude
- d : dimension d'arrivée 2 ou 3



Démarche

→ Objectifs : Appréhender des objets de dimension élevée

1. Propriétés géométriques : distances, surfaces, volumes
2. Propriétés locales privilégiées (objet $\neq$ collection de points)

→ Moyens : Méthodes de réduction de dimension. Plusieurs approches et algorithmes existent dans d'autres domaines : visualisation de données, optimisation combinatoire

→ Réalisations et évaluation : détermination de critères spécifiques pour la visualisation d'objets, adaptation et hybridation de méthodes utilisées dans d'autres contextes, comparaison avec des approches simples (projections)

Détermination des critères d'évaluation

$\boxed{\text{Objet plongé dans } \Re^D \text{ à représenter dans } \Re^d}$

$\boxed{\text{Opérateur de réduction de dimension } x \in \Re^D \to \phi x \in \Re^d}$

Critères en Visualisation de données

Discussions avec J. Chaussard (LAGA, Univ. Paris Nord)

- Visualisation de graphes (graphe lisible) : Valeurs propres
  => pas de notion de structure géométrique !
- Classification/clustering : Plus proches voisins => fusion de points !
- Apprentissage (dimension de l'espace d'arrivée assez élevée) :
  Préservation des distances pour chaque couple (x, y)

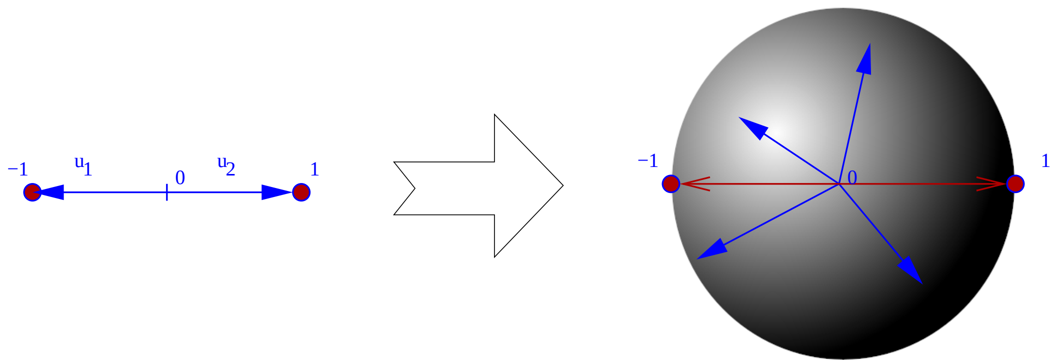
$$||x - y|| = ||\phi x - \phi y||$$

Détermination des critères d'évaluation

Optimisation combinatoire

Thèse de Vu Khac Ky (LIX, Polytechnique)

- Problèmes d'optimisation avec des variables décision donc de dimension 1
- Convexification/relaxation : Données dans un espace de dimension n (nombre de variables)

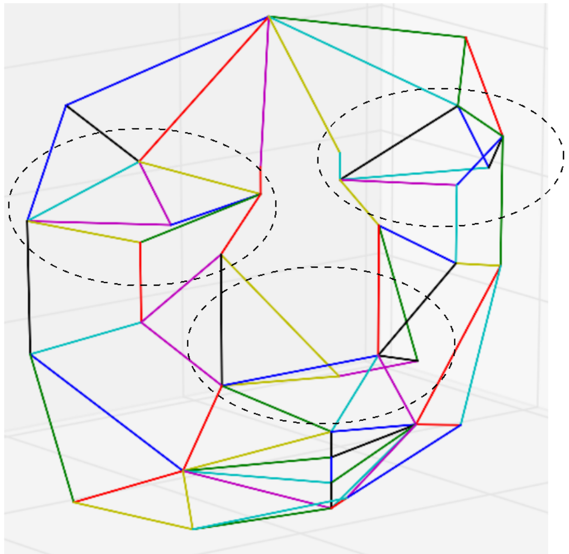


- Critère d'évaluation : fonction à optimiser
- Réduction de dimension : retour dans un espace de dimension 1 en dégradant le moins possible la valeur de la fonction

Détermination des critères d'évaluation

Dans notre contexte : visualisation d'objets

- Contribution pour préserver des structures locales : utilisation d'hyper-arêtes. Permet d'ajouter de l'information en plus des coordonnées des points



- Evaluer la préservation des hyper-arêtes en définissant une fonction adaptée (poids sur les couples de points)

La dimension de l'espace d'arrivée vaut 2 ou 3 : nous allons constater que l'utilisation exclusive de méthodes d'un seul domaine est insuffisante

Algorithme de Johnson-Lindenstrauss (visualisation de données)

Principe

- Objectif : Projection aléatoire en respectant au mieux les distances entre les points
- P : matrice $d \times D$ aléatoire, H : matrice $D \times D$, L : matrice $D \times D$ aléatoire
- P matrice de projection, HL matrice d'adoucissement
- $\phi = PHL$: opérateur linéaire !

Avantages

- Rapide : algèbre linéaire élémentaire.
- Bon respect des distances en dimension d'arrivée élevée :
  si d est de l'ordre du logarithme du nombre de points

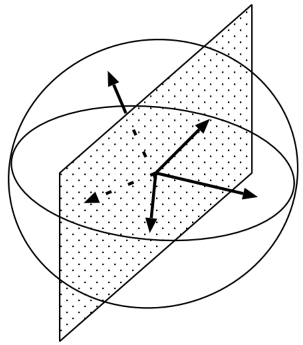
Inconvénient

- Distorsion potentiellement élevée en dimension faible

Algorithme de Goemans-Williamson (optimisation combinatoire)

Principe

- Un hyperplan (aléatoire) permet le passage direct de la dimension D à la dimension 1



- Contribution : réalisation d'une version déterministe pour contrôler la dimension d'arrivée
- Calcul de rotations successives dans le plan des 2 dernières coordonnées

$\rightarrow$ Avantage : existence d'une fonction à optimiser : les hyper-arêtes peuvent être prises en compte

$\rightarrow$ Inconvénient : Temps de calcul important en dimension élevée car la dimension est réduite d'une unité à la fois

Contribution personnelle

Adaptation et combinaison des algorithmes précédents

Idée

Passage du premier algorithme au second pour tirer profit de leurs qualités :

- Rapidité de Johnson-Lindenstrauss (JL)
- Prise en compte des hyper-arêtes dans Goemans-Williamson (GW)

Solution et mise en œuvre

Gestion dynamique du passage de (JL) à (GW) :

- Dichotomie en utilisant (JL) pour trouver la dimension de rupture : distorsion trop grande ou violation trop importante des hyper-arêtes
- Puis utilisation de (GW) jusqu'en dimension 2 ou 3

Choix des objets et des tests

Base d'objets (une centaine jusqu'à $D = 100$)

- hypercubes, k-simplexes de dimension élevée
- Objets particuliers (visage, théière,..)
- Rotations des objets précédents en dimension plus élevée
- Unions des objets précédents (objets avec des propriétés locales spécifiques)

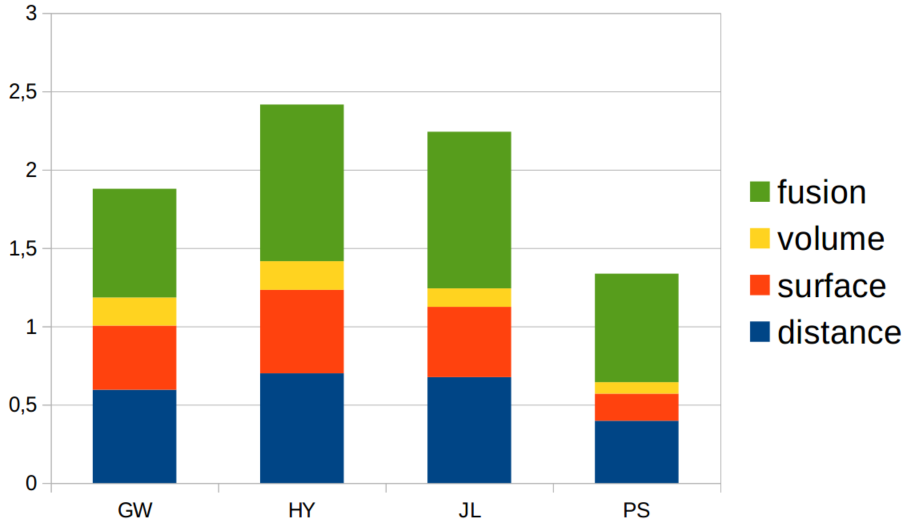
Protocole expérimental

- Tests avec un algorithme seul : (JL), (GW), projections simples
- Tests avec l'algorithme hybride (HY)
- Evaluation grâce à quatre indicateurs compris entre 0 et 1 calculés à partir :
  1. du nombre de fusions de points
  2. des distances (arête simple)
  3. des surfaces (hyper-arête de trois sommets)
  4. des volumes (hyper-arête de quatre sommets)

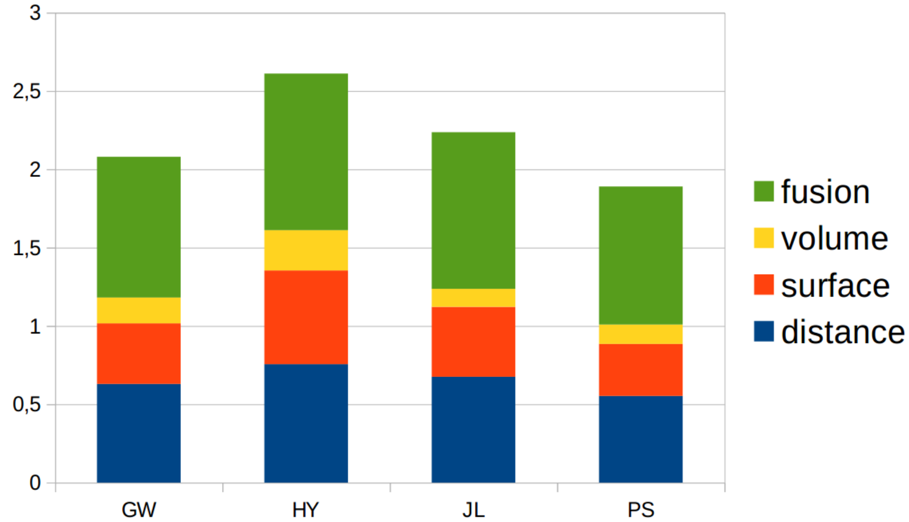
Synthèse des tests

Algorithmes/générateurs implémentés en C++/GSL et en python (2000 lignes)

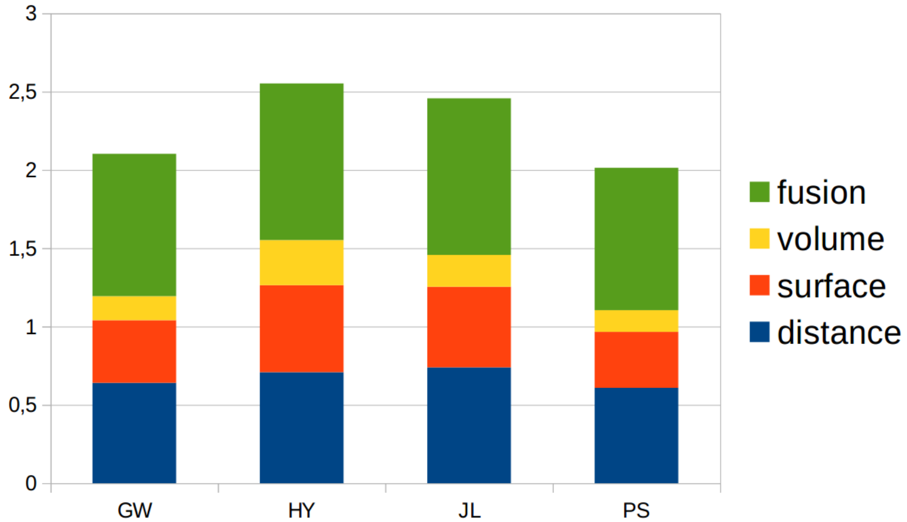
Moyennes des indicateurs pour chaque famille d'objets



hypercubes, k-simplexes

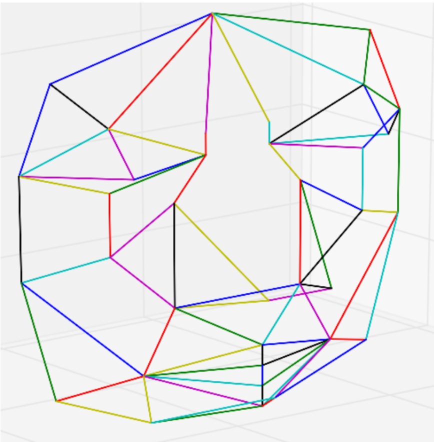


Objets particuliers (dont leurs rotations)

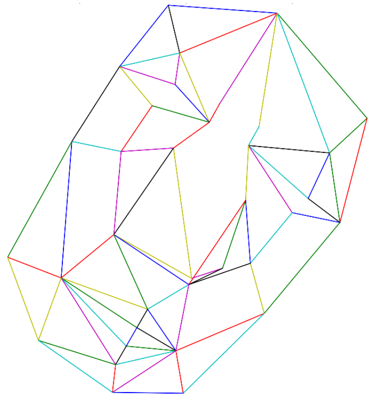


Unions d'objets (dont leurs rotations)

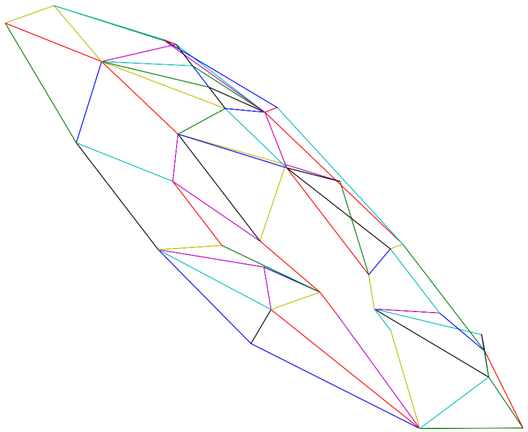
Un premier exemple : de $D = 4$ à $d = 2$



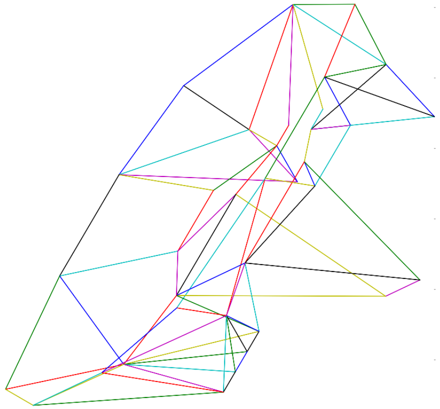
Objet initial en 3D (puis rotation en dimension 4)



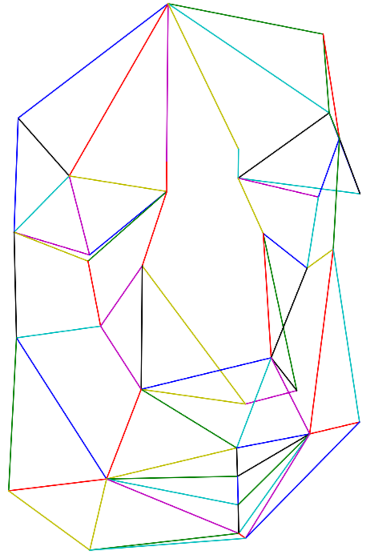
Goemans-Williamson (GW) $d = 2$



Projection simple $d = 2$

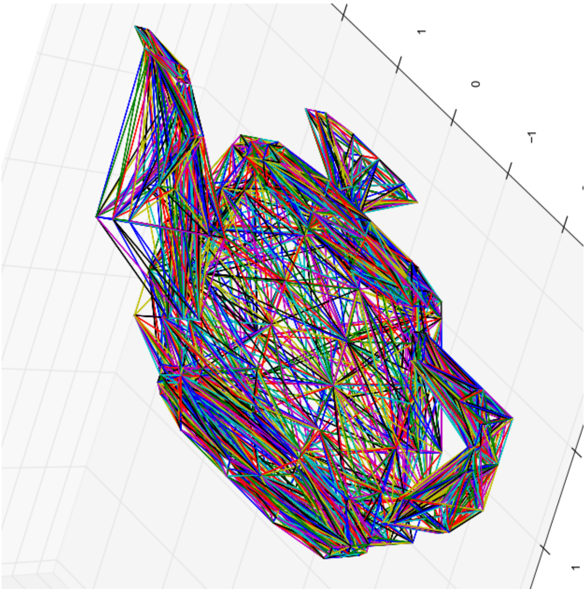


Johnson-Lindenstrauss (JL) $d = 2$

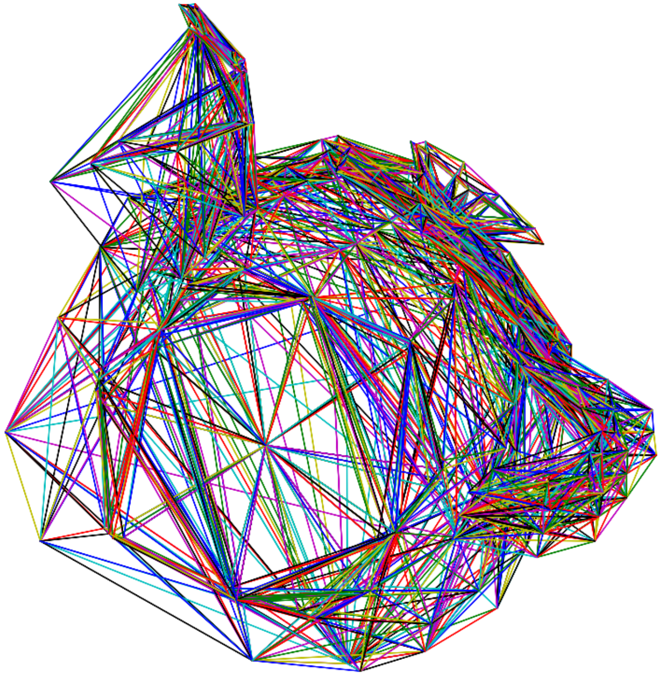


Algorithme hybride (HY) $d = 2$

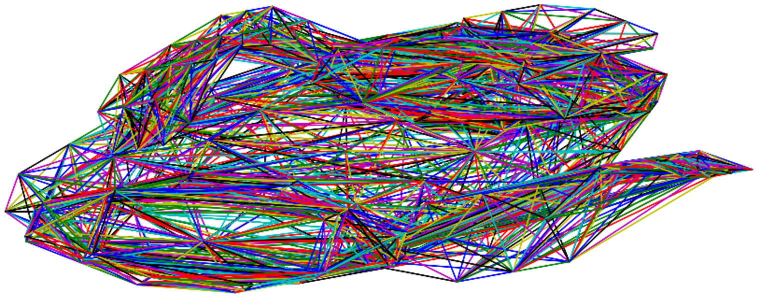
TeaPot (D=10 par rotation)



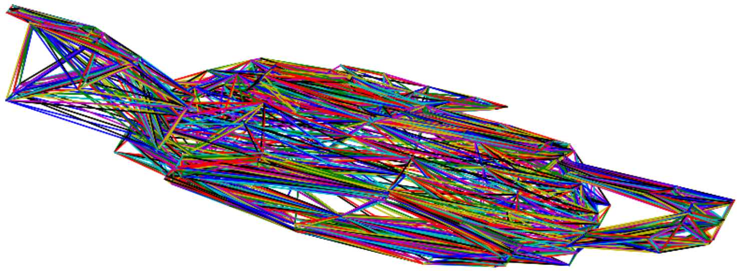
Original (avant rotation)



Goemans-Williamson

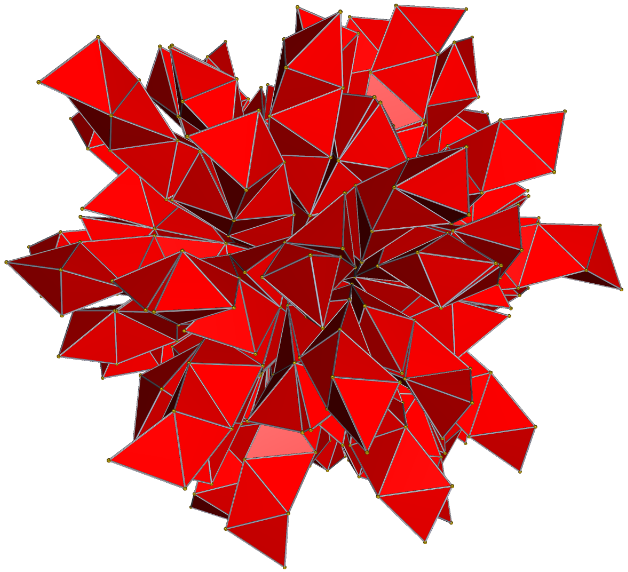


Johnson-Lindenstrauss

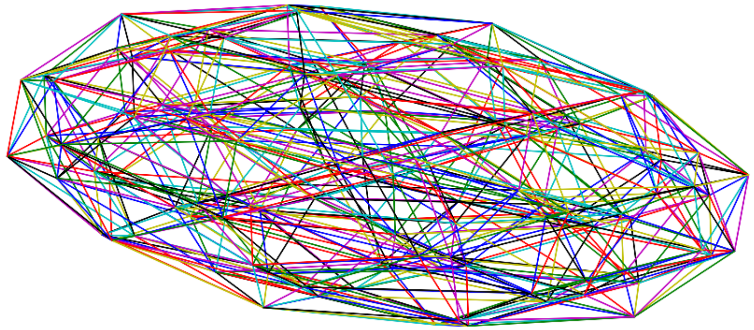


Hybride

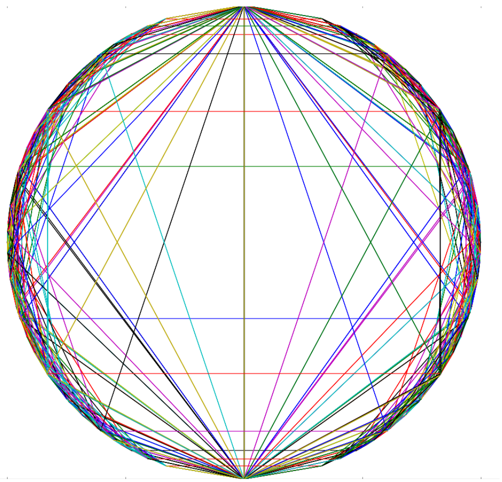
Tetraplex (D=4) : 600 cellules tétraédriques



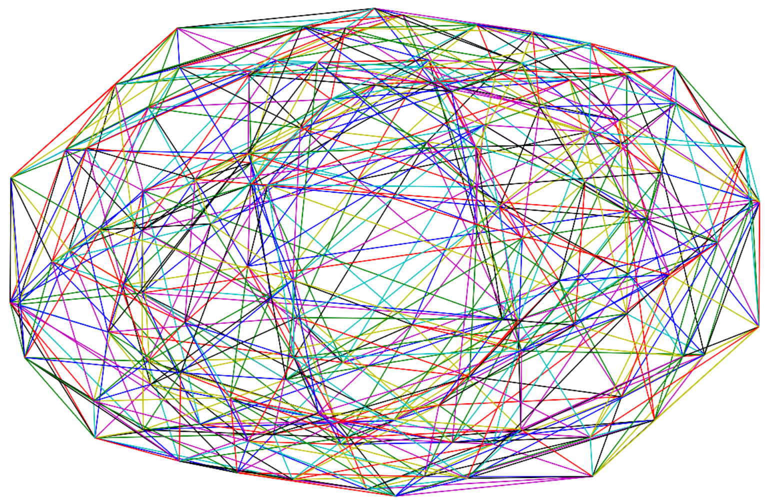
Patron du Tetraplex



Johnson-
Lindenstrauss



Goemans-
Williamson



Hybride

Conclusion et Perspectives

- Bilan
  1. La détermination des critères a permis d'effectuer les choix algorithmiques
  2. L'algorithme proposé combine les avantages des deux méthodes utilisées et est performant pour les critères retenus
  3. L'utilisateur peut définir des hyper-arêtes pour explorer une zone donnée
- Perspectives et développements possibles
  1. Utiliser l'algorithme "squelette" (Barequet, Huber) pour évaluer la préservation des propriétés topologiques des objets
  2. Déterminer des ordres particuliers dans l'algorithme de Goemans-Williamson pour réduire les dimensions en préservant au mieux les propriétés topologiques
  3. Déterminer le nombre et la nature des projections à effectuer pour proposer à l'utilisateur la meilleure compréhension possible de l'objet

Fichier d'entrée et algorithmes : exemples

5-simplexe

```
4 //dimension D
5 //nombre de points
//coordonnées des points
1 0 0 0
...
0.80901699 0.80901699 0.80901699
0.80901699
25 //nombre d'hyper-arêtes
//arêtes simples (distances)
2 0 1 ... 2 3 4
//hyper-arêtes de degré 3 (surfaces)
3 0 1 2 ... 3 2 3 4
//hyper-arêtes de degré 4 (volumes)
4 0 1 2 3 ... 4 1 2 3 4
```

./tipe cube10 GW 3

```
dim : 10 ; f = 2.59072e+06
dim : 9 ; f = 2.59072e+06
dim : 8 ; f = 2.59072e+06
dim : 7 ; f = 2.59072e+06
dim : 6 ; f = 2.59072e+06
dim : 5 ; f = 2.59072e+06
dim : 4 ; f = 2.59072e+06
dim : 3 ; f = 2.59072e+06
0.164317 0.02 0.00136931 3.1001e-05
```

./tipe cube10 JL 3

```
0.744249 0.645495 0.332948 1
```

./tipe cube10 HY 3

```
7 0.753511 0.56495 0.352156
5 0.757948 0.68137 0.418814
4 0.731155 0.702258 0.595657
dim : 4 ; f = 3.20923e+06
dim : 3 ; f = 3.33448e+06
0.713589 0.649956 0.46173 1
```