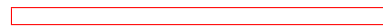


LinBox: Présentation générale et Solutions génériques pour l'algèbre linéaire



Projet LinBox

Pascal Giorgi

Laboratoire de l'informatique du parallélisme
ENS Lyon - CNRS - INRIA

Introduction

- Projet LinBox :

- 29 chercheurs (USA, Canada, France) .

- E.Kaltofen [NCSU], D.Saunders [Delaware], M.Giesbrecht [Waterloo], G.Villard [ENSL].

- Financé par NSF, NSERC, CNRS et ENSL.

- Site web: *<http://www.linalg.org>* et *<http://www.linalg.net>*

- Bibliothèque C++, avec licence GPL, (40000 lignes de code).

Principaux intérêts:

- Algèbre Linéaire.
- Matrices boîtes noires.
- Généricité sur les domaines de calcul.
- Solutions pour différents problèmes d'algèbre linéaire.

Plan de l'exposé

- I) Présentation de la bibliothèque LinBox.
- II) Généricité Algorithmes/Interfaces.
- III) Polynôme minimal: implantation générique.

I) Présentation de la bibliothèque LinBox

Structure de LinBox

- Trois niveaux d'implantation (permettant la réutilisation et la reconfiguration)



- Matrices boîtes noires & domaines de coefficients respectent leurs interfaces.
- Interface= classe C++ virtuelle, template *archetype*:
 - définit l'interface commune aux objets.
 - fournit une instance de code compilé.
 - contrôle l'explosion de code.
 - utilisation optionnelle.

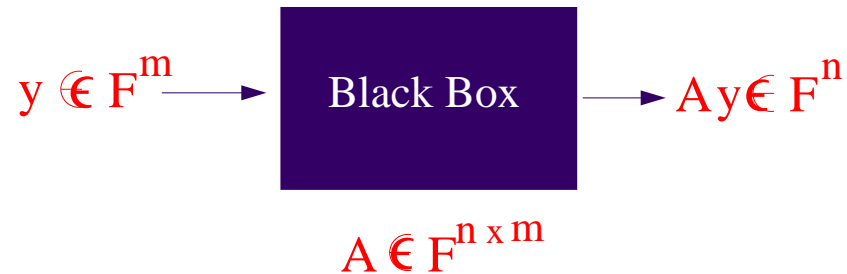
Structure des corps finis (domaine des coefficients)

- Encapsulation des types :Eléments, Générateur aléatoire d'éléments.
- Eléments: aucune information sur le corps d'appartenance.
- Corps: méthodes affectation, égalité, arithmétique, IO pour les éléments:

```
x = y      : F.assign(x,y)
x == y     : F.areEqual(x,y)
x = y + z  : F.add(x,y,z)
cout<< x   : F.write(cout,x)
```

- Intégration : LinBox directement, bibliothèques externes (au travers de wrappers).

Structure des matrices boîtes noires (Blackbox)



- Modèle des Blackbox paramétré par une classe de vecteurs (domaine d'application).
- Domaine de calcul passé comme paramètre ou spécifié comme attribut.
- Seule l'application à un vecteur est autorisée :

$$x = Ay \quad : \quad A.\text{apply}(x,y)$$

$$x = A^T y \quad : \quad A.\text{applyTranspose}(x,y)$$

- Récupération des dimensions de la matrice:

$A.\text{rowdim}()$

$A.\text{coldim}()$

Algorithmes dans LinBox

- **Basés sur des corps finis :**

- * Algorithmes probabilistes et heuristiques (Vérification, Las-Vegas, Monte-Carlo).
- * Polynôme minimal (Wiedemann/Lanczos, méthodes par blocs).
- * Préconditionnement (Rang , Déterminant).
- * Comparaison avec l'élimination de Gauss.

- **Méthodes disponibles pour les nombres rationnels et les entiers:**

- * Théorème des restes chinois avec les corps finis.
- * Reconstruction P-adique.
- * Forme normale de Smith (GAP package).
- * Systèmes linéaires diophantiens.

- **Implantations disponibles:**

- * Arithmétique spécialisée des corps finis (extensions, tables, polynomiales).
- * Solutions : $\left\{ \begin{array}{l} - \text{Système linéaire.} \\ - \text{Rang.} \\ - \text{Déterminant.} \\ - \text{(Polynôme caractéristique).} \\ - \text{Polynôme minimal.} \end{array} \right.$
- * Conditionneurs (Toeplitz, butterfly, creuse, diagonale).
- * Formes normales de matrices.

*

II) Généricité Algorithmes/Interfaces

Intérêts de la genericité ?

- Utilisation de bibliothèques externes efficaces (intégration facile)
- Evolution dynamique de la bibliothèque
- Modularité au niveau des solutions proposées

Performances préservées !

Quelle généricité ?

- **Corps Finis** : Opérations arithmétiques de base
- **Vecteurs** : Accès aux données (interface= STL)
- **BlackBox** : Produits matrices-vecteurs
- **Matrices** : Opérations de base + ...
- **Algorithmes** : Objets $\left\{ \begin{array}{l} \text{Matrices} \\ \text{BlackBox} \\ \text{Vecteurs} \\ \text{Corps Finis} \end{array} \right.$

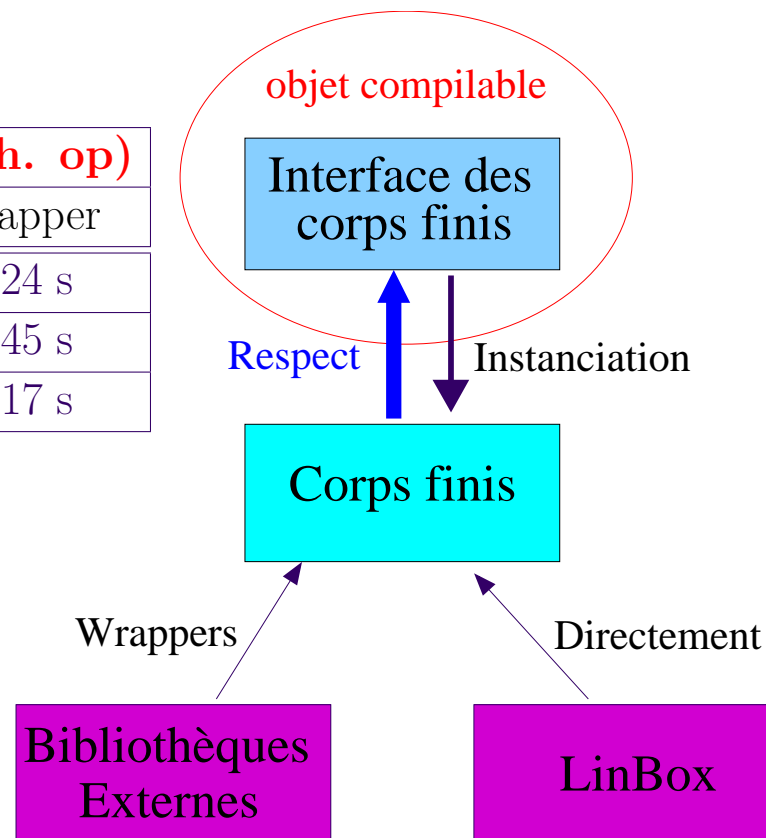
Interfaces des Corps Finis

Timings for different LinBox field levels (arith. op)

Library	Z/pZ	loop	Directly	Wrapper
NTL::ZZ_p	1978146594666	100.000	0.24 s	0.24 s
//	//	1.000.000	2.45 s	2.45 s
NTL::zz_p	553	300.000	0.16 s	0.17 s

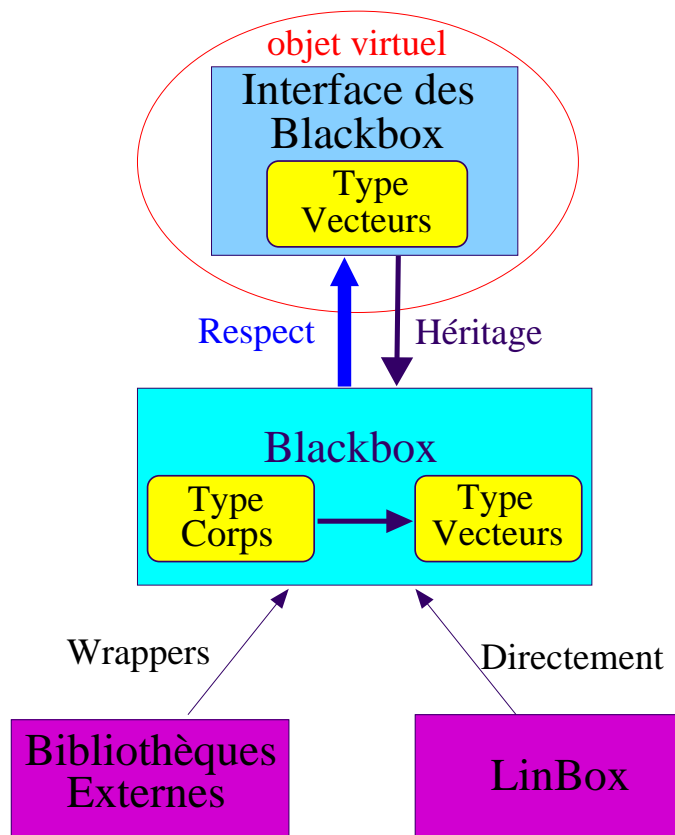
- Interface:

objet à part entière
 validations des corps finis
 évite la réécriture (template)

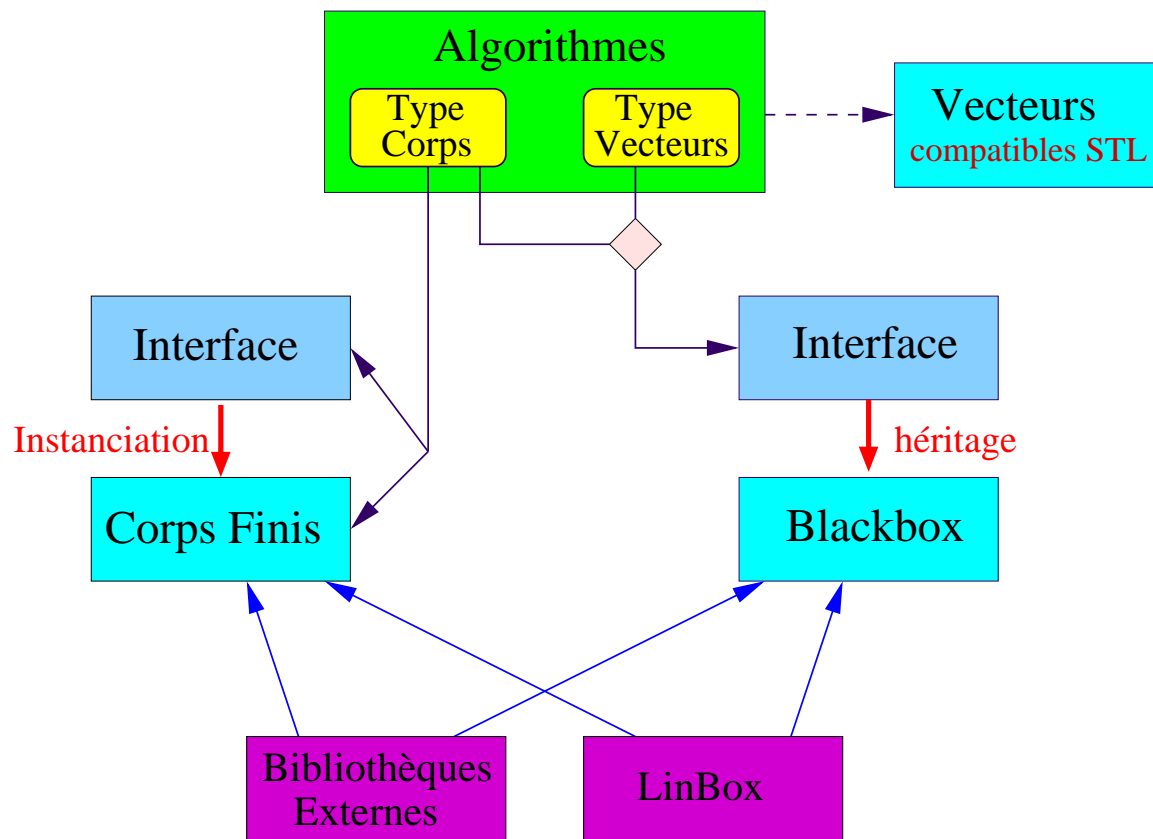


Interfaces des BlackBox

- Interface: purement virtuelle
- Paramétrée par un type de vecteur (compatible STL)
- Vecteurs: définis sur un type d'élément
- LinBox fournit 3 types de vecteurs :
 - Dense vector
 - Sparse sequence vector
 - Sparse map vector



LinBox: Implantation générique des Algorithmes



III) Polynôme minimal : implantation générique

Calcul du polynôme minimal.

Approche Wiedemann pour une matrice $\mathbf{A} \in \mathbf{K}^{n \times n}$:

- Etape I: Calcul de la suite $\mathbf{S} = \{\mathbf{U}\mathbf{A}^i\mathbf{V}\}_{0 \leq i \leq 2n}$, \mathbf{U} et \mathbf{V} vecteurs aleatoires.
- Etape II: Calcul de $\mathbf{G}(\mathbf{x}) =$ polynôme générateur de \mathbf{S} (Berlekamp/Massey).
- Polynôme minimal = polynôme miroir de $\mathbf{G}(\mathbf{x})$ (forte probabilité).

Généricité :

Etape II: Opérations de base sur les éléments de \mathbf{K} .

Etape I: Produits Matrice \times Vecteur (*Blackbox*).

Polynôme minimal par blocs

Même approche que pour le cas scalaire.

- Etape I: Calcul de la suite $\mathbf{S} = \{\mathbf{U}\mathbf{A}^i\mathbf{V}\}_{0 \leq i \leq N}$, $\mathbf{U} \in \mathbf{K}^{p \times n}$ et $\mathbf{V} \in \mathbf{K}^{n \times q}$ des blocs vectoriel aléatoires.
- Etape II: Calcul de $\mathbf{G}(\mathbf{x}) =$ polynôme matriciel générateur de \mathbf{S} .
- Polynôme minimal = polynôme miroir de $\mathbf{G}(\mathbf{x})$ (forte probabilité).



Généricité:

- Etape II: Opérations sur des matrices.

Problématique:

- Etape I: Produits Matrice \times Bloc vectoriel, Bloc vectoriel \times Bloc vectoriel (problème généricité/efficacité).

Solutions: Approche Blackbox (parallélisme)

Soient \mathbf{A}^i une Blackbox et \mathbf{S}_i une matrice.

- Version 1 (basique):

\mathbf{U}, \mathbf{V} blocs vectoriels (type vecteur de vecteurs).

$\mathbf{V} = \mathbf{A}^i \mathbf{V}$: produits Blackbox-vecteur.

$\mathbf{S}_i = \mathbf{UW}$: produits Vecteurs \times Vecteurs + construction matrice.

- Version 2 (rapide):

\mathbf{U}, \mathbf{V} matrices (extraction colonne) + synchronisation avec les vecteurs de \mathbf{A} .

$\mathbf{V} = \mathbf{A}^i \mathbf{V}$: produits Blackbox \times Colonne.

$\mathbf{S}_i = \mathbf{UW}$: produit de matrices.

Solutions: Approche Matbox (produit de matrices rapide)

- Matbox = extension des Blackbox aux matrices :
→ Seule l'application à une matrice est autorisée.



- Soient \mathbf{A}^i une Matbox et $\mathbf{U}, \mathbf{V}, \mathbf{S}_i$ des matrices.
 $\mathbf{V} = \mathbf{A}^i \mathbf{V}$: produit Matbox-matrice.
 $\mathbf{S}_i = \mathbf{U} \mathbf{W}$: produit de matrices.

Conclusion

- LinBox: réponse à une demande des utilisateurs.
- Traitement spécifique de problème concret (forme de Smith, Trefethen).
- La généricité dans LinBox est fortement définie par les interfaces.

Perspective

- Polynôme minimal par blocs: Comparaison avec d'autres implantations (ex: E.Thomé).
- Validaton du modèle matriciel.
- Interfaces génériques (projet Roxane).