The LinBox Project for Linear Algebra Computation

A Practical Tutorial

Daniel S. Roche

ORCCA
Ontario Research Centre for Computer Algebra

Symbolic Computation Group
School of Computer Science
University of Waterloo

UNIVERSITY OF Waterloo

MOCAA 2008
University of Western Ontario
8 May 2008

Goals for Today

I want to convince you that LinBox is. . .

- **the** tool for exact linear algebra
- easy to install and use (if you want it to be)
- worth getting involved with

Significant Publications

1986 Wiedemann: *Solving sparse linear equations over finite fields*

1991 Kaltofen & Saunders: On Wiedemann's Algorithm

1993 Coppersmith: Block Lanczos, Block Wiedemann

1995 Montgomery: Implementation of Block Lanczos

1997 Villard: Analysis of Block Wiedemann

2001 Giesbrecht; Dumas, Saunders, & Villard: Smith forms

2001 Chen, Eberly, Kaltofen, Saunders, Turner, Villard:
Efficient Matrix Preconditioners for Black Box Linear Algebra

2002 Dumas, Gautier, Giesbrecht, Giorgi, Hovinen, Kaltofen, Saunders, Turner, Villard:
Linbox: A Generic Library For Exact Linear Algebra

2002 Dumas, Gautier, Pernet:
FFLAS: Finite Field Linear Algebra Subroutines

LinBox Milestones

2000 Initial design meetings

2002 World Scientific paper

2004 BLAS is thoroughly integrated

2005 LinBox 1.0 Released
JGD gives tutorial at ISSAC in Beijing
LinBox included in open-source computer algebra CD

2006 Maple interface and web computation server

2007 SAGE integration

2008 Linbox 1.5

What is LinBox?

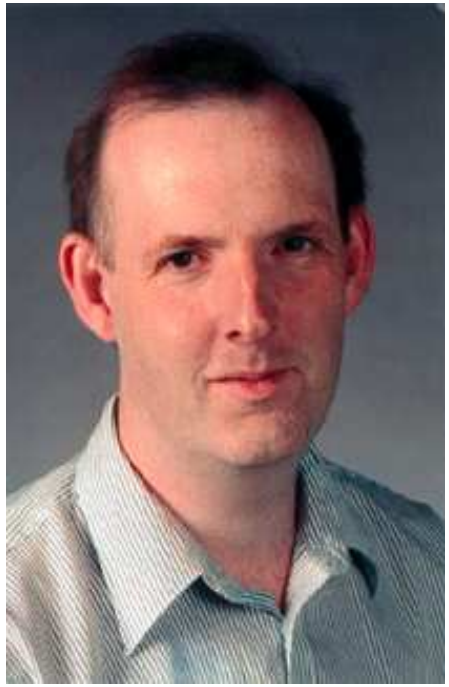
- C++ Library for Exact Computational Linear Algebra
- Open-source (LGPL), international research project
- Generic, using C++ templates
- Middleware — uses fast, low-level libraries,
  used by higher-level CA systems
- Originally for sparse & structured blackbox computations
- Now incorporates dense as well

Fathers of LinBox

Dave Saunders

Erich Kaltofen

Gilles Villard



Mark Giesbrecht

Developers

- Jean-Guillaume Dumas
- Bradford Hovinen
- Will Turner
- David Pritchard
- Clement Pernet
- Pascal Giorgi
- Zhendong Wang
- William Stein
- Mike Abshoff

Wiedemann's Algorithm

The Problem

Given nonsingular $A \in \mathsf{F}^{n \times n}$ and $\mathbf{v} \in \mathsf{F}^n$ find $\mathbf{x} \in \mathsf{F}^n$ such that $A\mathbf{x} = \mathbf{v}$.

Wiedemann's Algorithm

The Problem

Given nonsingular $A \in \mathsf{F}^{n \times n}$ and $\mathbf{v} \in \mathsf{F}^n$ find $\mathbf{x} \in \mathsf{F}^n$ such that $A\mathbf{x} = \mathbf{v}$.

The Algorithm

First, project on the right:

The sequence

$$\mathbf{v}, \quad A\mathbf{v}, \quad A^2\mathbf{v}, \quad A^3\mathbf{v}, \ \ldots$$

is linearly recurrent of degree n.

Note: $A^i\mathbf{v} = A \cdot (A^{i-1}\mathbf{v}) = A \cdot (A \cdot (A \cdots A\mathbf{v}))$

Wiedemann's Algorithm

The Problem

Given nonsingular $A \in \mathsf{F}^{n \times n}$ and $\mathbf{v} \in \mathsf{F}^n$ find $\mathbf{x} \in \mathsf{F}^n$ such that $A\mathbf{x} = \mathbf{v}$.

The Algorithm

Then, project on the left with random $\mathbf{u} \in \mathsf{F}^n$:

The sequence

$$\mathbf{u}^T\mathbf{v},\ \mathbf{u}^T A\mathbf{v},\ \mathbf{u}^T A^2\mathbf{v},\ \mathbf{u}^T A^3\mathbf{v},\ \ldots$$

has the same linear recurrence with high probability.

Wiedemann's Algorithm

The Problem

Given nonsingular $A \in \mathsf{F}^{n \times n}$ and $\mathbf{v} \in \mathsf{F}^n$ find $\mathbf{x} \in \mathsf{F}^n$ such that $A\mathbf{x} = \mathbf{v}$.

The Algorithm

Compute $f \in \mathsf{F}[x]$, the minimum polynomial of $(\mathbf{u}^T A^i \mathbf{v})_{i \geq 0}$.

$$\begin{aligned} f(x) &= -1 + f_1 x + f_2 x^2 + \cdots + f_n x^n \\ 0 &= -\mathbf{u}^T\mathbf{v} + f_1 \mathbf{u}^T A\mathbf{v} + f_2 \mathbf{u}^T A^2\mathbf{v} + \cdots + f_n \mathbf{u}^T A^n \mathbf{v} \end{aligned}$$

Wiedemann's Algorithm

The Problem

Given nonsingular $A \in \mathsf{F}^{n \times n}$ and $\mathbf{v} \in \mathsf{F}^n$ find $\mathbf{x} \in \mathsf{F}^n$ such that $A\mathbf{x} = \mathbf{v}$.

The Algorithm

W.h.p. f is the minimum polynomial of $(A^i\mathbf{v})_{i \geq 0}$.

$$\begin{aligned} f(x) &= -1 + f_1 x + f_2 x^2 + \cdots + f_n x^n \\ 0 &= -\mathbf{u}^T\mathbf{v} + f_1 \mathbf{u}^T A\mathbf{v} + f_2 \mathbf{u}^T A^2\mathbf{v} + \cdots + f_n \mathbf{u}^T A^n\mathbf{v} \\ \mathbf{0} &= -\mathbf{v} + f_1 A\mathbf{v} + f_2 A^2\mathbf{v} + \cdots + f_n A^n\mathbf{v} \end{aligned}$$

Wiedemann's Algorithm

The Problem

Given nonsingular $A \in \mathsf{F}^{n \times n}$ and $\mathbf{v} \in \mathsf{F}^n$ find $\mathbf{x} \in \mathsf{F}^n$ such that $A\mathbf{x} = \mathbf{v}$.

The Algorithm

Rearrange and multiply by A^{-1}:

$$
\begin{aligned}
f(x) &= -1 + f_1 x + f_2 x^2 + \cdots + f_n x^n \\
0 &= -\mathbf{u}^T\mathbf{v} + f_1 \mathbf{u}^T A\mathbf{v} + f_2 \mathbf{u}^T A^2\mathbf{v} + \cdots + f_n \mathbf{u}^T A^n \mathbf{v} \\
\mathbf{0} &= -\mathbf{v} + f_1 A\mathbf{v} + f_2 A^2\mathbf{v} + \cdots + f_n A^n \mathbf{v} \\
\mathbf{x} = A^{-1}\mathbf{v} &= f_1 \mathbf{v} + f_2 A\mathbf{v} + \cdots + f_n A^{n-1}\mathbf{v}
\end{aligned}
$$

Black Box Approach

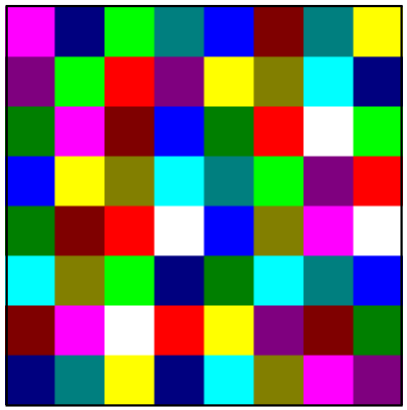
"Usual" Goals

- $O(n)$ black box calls
- $O(n^2)$ other work
- $O(n)$ space

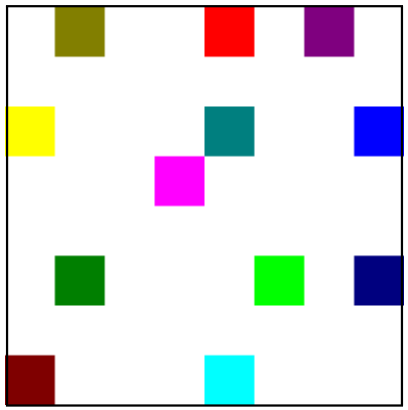
Good Black Box Algorithms for:
minimum polynomial, rank, determinant, linear system solving, Smith normal form, . . .

Black Box Classes

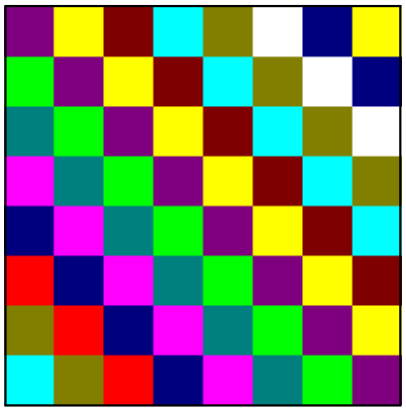
Common costs for black box vector apply:



Dense
$O(n^2)$



Sparse
s entries
$O(s)$



Toeplitz
$O(\mathsf{M}(n))$

Other blackboxes: Hankel, Sylvester, Diagonal, Hilbert, . . .

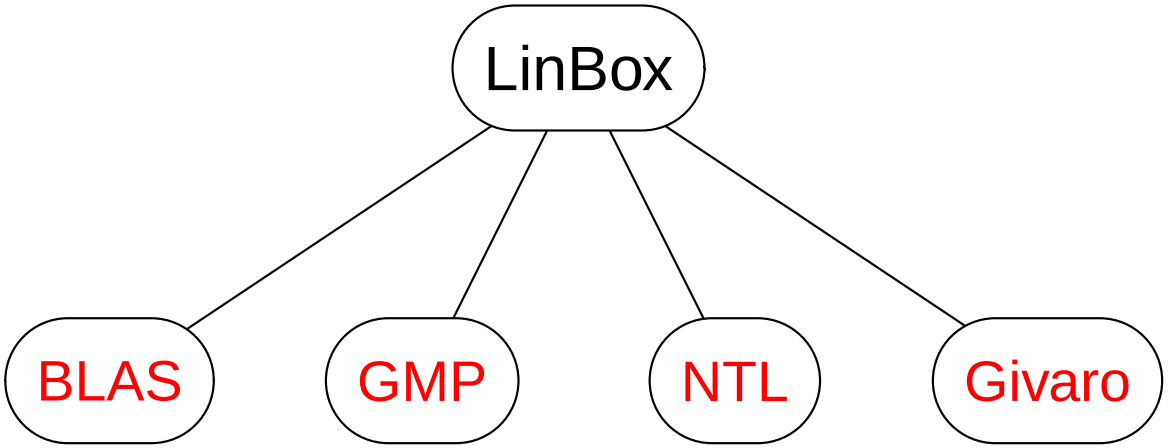
Block Methods

Similar to before, vectors now replaced by rectangular matrices.

- Many mathematical complications
- Allows use of fast dense methods (i.e. BLAS)
- Usually a tradeoff

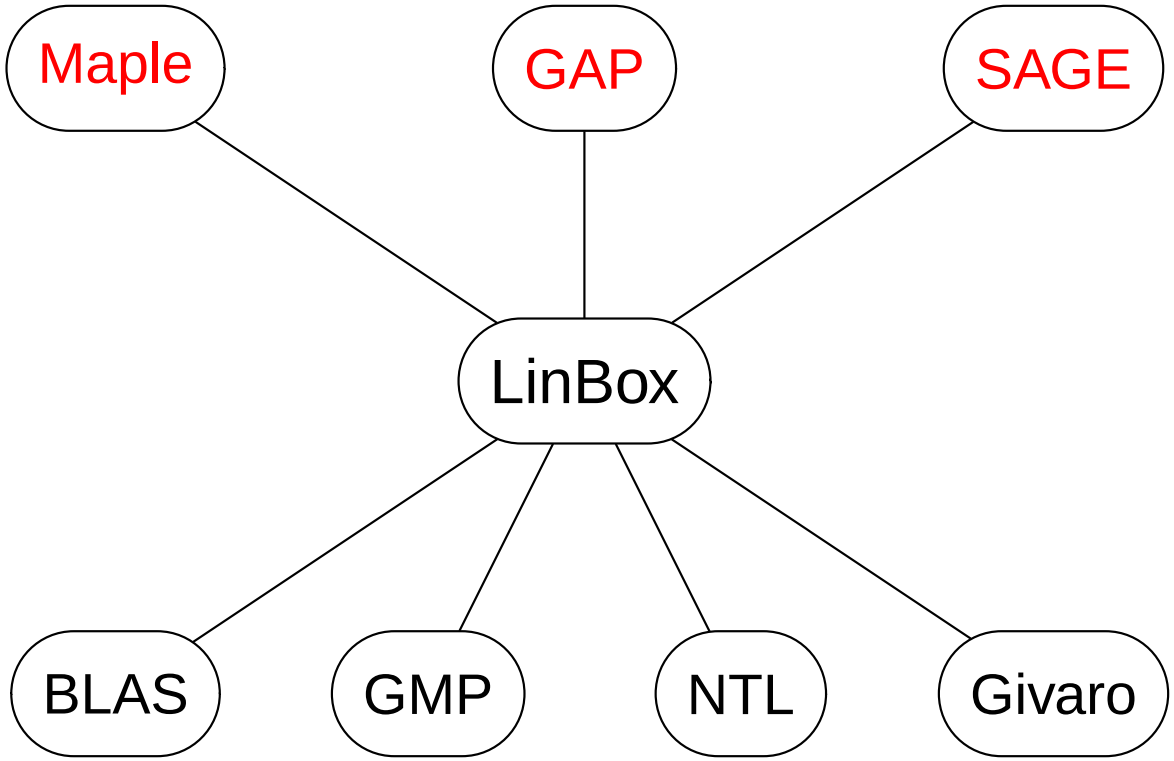
LinBox as Middleware

LinBox as Middleware



Packages used by LinBox

LinBox as Middleware



Higher-level Systems using LinBox

Basic Linear Algebra Subprograms

Extremely efficient low-level routines for floating-point linear algebra:

BLAS Levels

1. Scalar product, vector-scalar product, vector-vector sum
2. Dot product, matrix-vector product
3. Matrix-matrix product

Use either:

- Proprietary BLAS — for most architectures (e.g. Intel's MKL)
- Free BLAS (e.g. ATLAS, GotoBLAS)

GNU Multiplie Precision

Provides routines for exact, multiplie-precision integer arithmetic.

Classical, Karatsuba, Toom-Cook, and FFT multiplication (with crossovers).

Advantages:

- Many inner loops written in assembly
- Large user community

NTL, Givaro, . . .

- Libraries for number theory (field, ring, polynomial arithmetic).
- Used to work with matrices over finite fields, polynomials, etc.
- Also used for structured matrix computations (e.g. Toeplitz)
- Free, open-source, efficient

Three Choices to Make

1. **Underlying Domain** (“field”)
   - Integers, Rationals
   - Prime and Prime Power Fields
   - Polynomial Rings
2. **Matrix Representation**
   - Sparse Black Box
   - Dense Explicit
   - Structured Black Box
3. **Algorithm**

Vertical Structure Approach

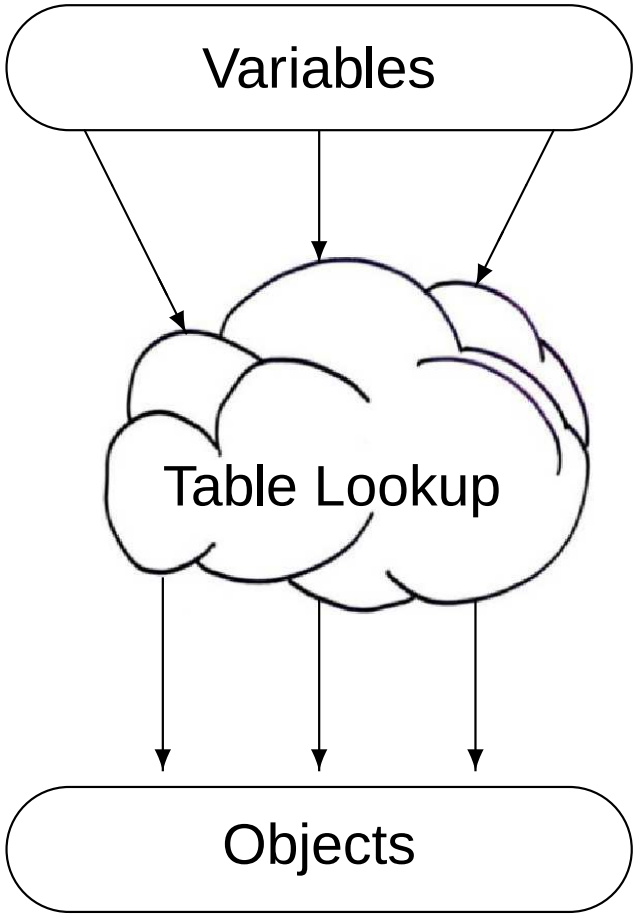
Used e.g. by NTL:

zz_p	GF2	ZZ_pE
zz_pX	GF2X	ZZ_pEX
vec_zz_p	vec_GF2	vec_ZZ_pE
mat_zz_p	mat_GF2	mat_ZZ_pE
⋮	⋮	⋮

- Only one choice given underlying domain!

Dynamic Type-Checking

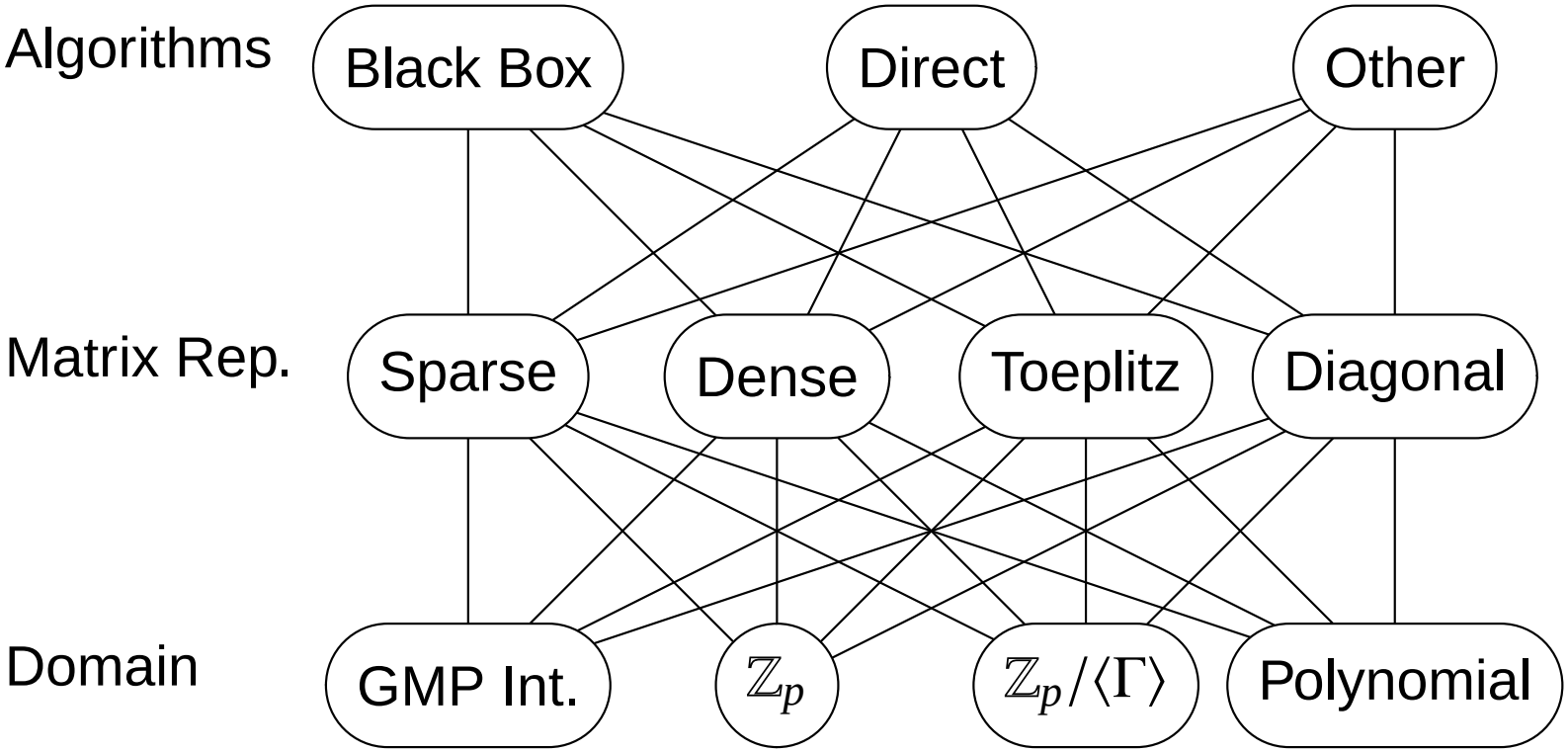
- Traditional polymorphism, required in Java, possible in C++



Have to do a table lookup
for every variable reference!

Our Approach: C++ Templates

- All type decisions made at compile-time
- Every combination is possible
- Huge efficiency gains over polymorphism (pipelining, inlining)



Drawbacks to C++ Templates

- Programmers can make bad decisions
  (e.g. using elimination algorithm with sparse blackbox matrix)

- Binary code bloat
  Every template combination which is used must be compiled.
  Creating a complete, precompiled library is impractical.

Levels of Use

1. Online Computation Server
   - Does not require any installation
2. Maple, SAGE, `examples` directory
   - User sees no C++ code
3. `solutions` directory
   - User chooses field, blackbox, writes in C++
4. Expert user
   - Direct programming — complete control

Installation

Installing LinBox is not difficult!

- Installing ATLAS can be tricky
- *All* LinBox dependencies can be found in Debian repo.
- Getting maximal efficiency is inherently difficult
- Middleware is inherently troublesome

Folder Structure

`examples`	Ready-to-go programs for typical problems
`interfaces`	Connections to Maple, SAGE, etc.
`linbox`	Library code
`/field`	Underlying domains
`/blackbox`	Matrix representations
`/algorithms`	Heart of the library
`/solutions`	Code for typical problems
`/randiter`	Random element generation
`/util`	General utilities (e.g. I/O)
`doc`	Documentation generation
`tests`	Correctness checks and benchmarks

Input/Output

If using LinBox at “Level” 1 or 2, I/O is handled automatically.

Otherwise, use the `MatrixStream` class.

- Automatic recognition of many formats
- Outputs (row,column,value) “triples” or a single dense array
- Connects to most matrix representations
- Allows any rep. to be read from any file

Similar ideas for output have been proposed; not yet implemented.

What LinBox is Great At

LinBox is great at computing

{rank, determinant, linear solution, characteristic polynomial, Smith form}

for matrices which are

{dense, sparse, structured}

over

{integers, rationals, prime fields, extension fields, polynomials}.

What LinBox is Great At

LinBox is great at computing

{rank, determinant, linear solution, characteristic polynomial, Smith form}

for matrices which are

{dense, sparse, structured}

over

{integers, rationals, prime fields, extension fields, polynomials}.

What LinBox is Great At

LinBox is great at computing

{rank, determinant, linear solution, characteristic polynomial, Smith form}

for matrices which are

{dense, sparse, structured}

over

{integers, rationals, prime fields, extension fields, polynomials}.

What LinBox is *not* Great At

- Inverse computation
- Nullspace computation
- Support for $\mathrm{GF}(2)$
- Parallelism
- Block methods

Sofware Engineering

LinBox has a rapidly growing user base (!)

We need some software engineering!

- Configure/Install (autohell)
- Removing legacy code
- Restructuring matrix/blackbox distinction
- Interfaces

Sofware Engineering

LinBox has a rapidly growing user base (!)

We need some software engineering!

- Configure/Install (autohell)
- Removing legacy code
- Restructuring matrix/blackbox distinction
- Interfaces

ANY VOLUNTEERS?