**1 Introduction**

Linear algebra computation over finite fields is important because many symbolic computation problems directly or indirectly rely on linear algebra using efficient finite field representations.

The use of block algorithms for linear algebra organizes algorithms to use mataix multiplications, with two following consequences:

1. **The construction of X implies that its first i rows, it is upper triangular with unit diagonal.**

2. **Krylov-LU method**

3. **Krylov-LU method**

The user must choose between them, depending on the number of invariant factors of the entry matrix determines which one should be used: Krylov-LU is the faster one. The reason for this is that Krylov-LU performs as many LSP factorization as the number of blocks, whereas Keller-Gehrig needs at most $sN^2$ of them. More precisely, the number of LSP factorizations for the Krylov-LU algorithm is $N$, whereas is the order of the biggest invariant factor. This explains why the computation time is decreasing with the number of blocks, for a fixed matrix order.

From these computations, it appears that neither algorithm is always better. The user must choose between them, depending on the number of invariant factors of the Frobenius form. If this information is not available, these algorithms could be combined into a hybrid: Krylov-LU is used for the computation of the first block, followed by the Keller-Gehrig algorithm for the rest of the computation.

**6 Experiments:**

**Krylov-LU vs. Keller-Gehrig and Ibarra**

For our experiments, we implemented these algorithms in C++, using a symbolic representation, and the standard BLAS interface. We used the modular finite field representation of CADO and the portable BLAS ATLAS.

The memory complexity of our implementation is $\mu N$ bytes for Keller-Gehrig algorithm, and $\mu N^2$ bytes for Krylov-LU (a field element is represented on $\mu$ bytes and the BLAS routines are working on double floating point $N$ bytes). This sets the limits respectively of $\mu < 0.37$ and $\mu < 1.17$ for Keller-Gehrig and Krylov-LU, when working with 1GB of RAM.

**7 Conclusion**

We have presented two algorithms for the computation of the characteristic polynomial over a finite field. They are both based on existing ideas, respectively developed by Krylov and Danilevski, and by Keller-Gehrig.

Our contribution was to show that the implementation of the LSP factorization of Ibarra & al. as a basic tool for these algorithms was not only possible, but also led to interesting improvements. Time complexity remains the same but the use of block algorithms provides high practical efficiency and algorithms become simpler from the program's point of view. We have also presented a new algorithm for the computation of dense matrices over non-field representation.

The two algorithms are complementary and the number of invariant factors of the entry matrix determines which one should be used: Krylov-LU for few blocks, and Keller-Gehrig for many blocks. Therefore a hybrid algorithm can be deduced for the case of a matrix whose structure is unknown.

These algorithms have been implemented in a specific package designed for efficient dense linear algebra over finite fields: **Pygrace)**. They are also available within the Linbox package designed for linear algebra.

It is not difficult to deal with the case of sparse matrices, for which our algorithms are also very efficient in memory. This will enable anyone interested in the computation of the characteristic polynomial of dense or sparse matrices over finite fields to benefit from high performance.

**References**


**Department of Computer and Information Science, University of Delaware, Newark, DE 19716 USA.**

11th June 2003