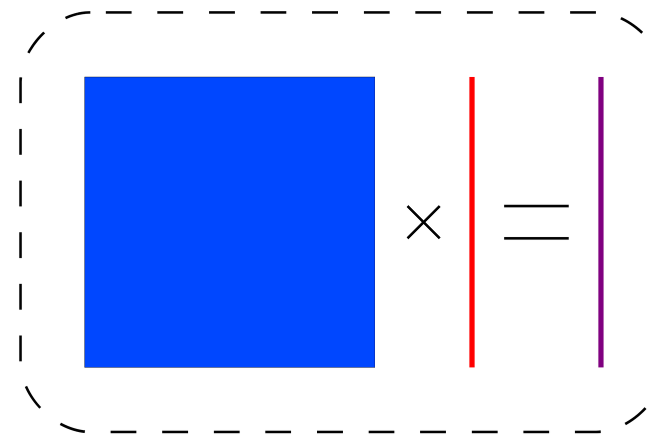
Symbolic/numeric exact rational linear system solver

Bryan Youse, B. David Saunders
University of Delaware, Newark, Delaware

Thanks to NSF Grant CCF-0830130

Solving an integer linear system: $Ax = b$



Symbolic

Software: LinBox, Maple, ...

Modular Methods

- Dixon's method: p-adic lifting
- Chinese Remaindering: computation modulo many prime numbers

- **Deliver the exact answer without error**
- **Not subject to machine limitations**
- Computationally expensive

Numeric

Software: LAPACK, MATLAB, many more

Direct Methods

- Gaussian Elimination
- QR Factorization

Iterative Methods

- Jacobi's method
- Lanczos' method
- GMRES method

- **Well studied; highly tuned**
- Solution accuracy limited by machine word size to floating point precision

Symbolic/Numeric Hybrid

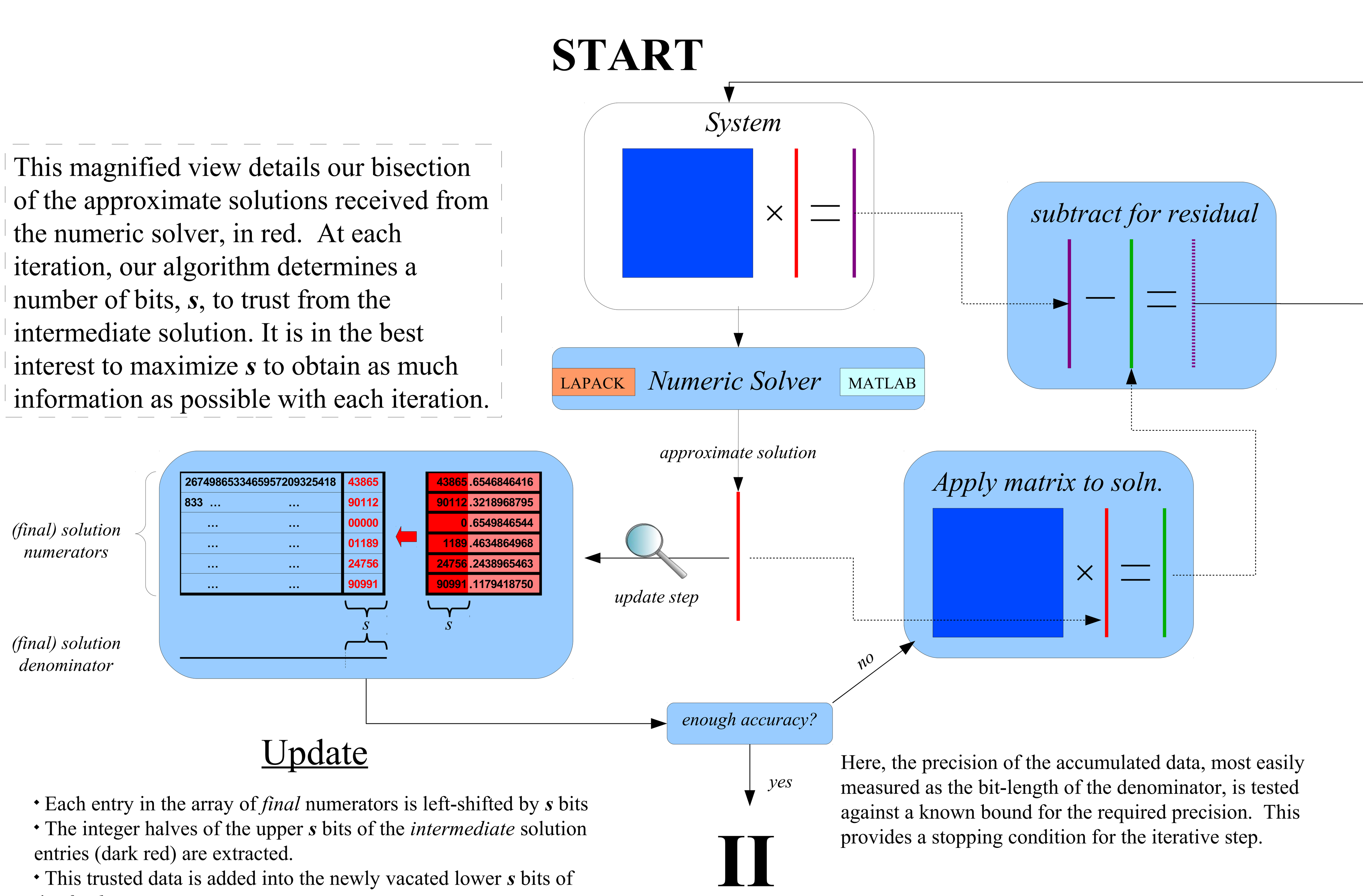
The best of both worlds! Our approach, an improvement on earlier work by Wan [1], can solve exactly well-conditioned integer linear systems using numeric solving methods. This is made possible by the following **fact:**

If two rational numbers $r_1 = \frac{a}{b}$, $r_2 = \frac{c}{d}$ are given with gcd(a, b) = 1, gcd(c, d) = 1, and $r_1 \neq r_2$, then $|r_1 - r_2| \geq \frac{1}{bd}$

i.e. rational numbers with bound denominators are **discrete**. If we can obtain a solution with very high accuracy, we can reconstruct the rational solution

I. *Iterative Refinement*

This magnified view details our bisection of the approximate solutions received from the numeric solver, in red. At each iteration, our algorithm determines a number of bits, ***s***, to trust from the intermediate solution. It is in the best interest to maximize ***s*** to obtain as much information as possible with each iteration.



Update

- Each entry in the array of *final* numerators is left-shifted by ***s*** bits
- The integer halves of the upper ***s*** bits of the *intermediate* solution entries (dark red) are extracted.
- This trusted data is added into the newly vacated lower ***s*** bits of the *final* numerators
- The *final* solution's (common) denominator is left-shifted by ***s*** bits

Here, the precision of the accumulated data, most easily measured as the bit-length of the denominator, is tested against a known bound for the required precision. This provides a stopping condition for the iterative step.

II. *Rational Reconstruction*

The Iterative Refinement step produces a vector of numerators and their common denominator. These can be viewed as real numbers. The size of the denominator of the exact rational solution is less than the product of the Hadamard bound of the input matrix and the norm of the input right-hand side vector. Because we know this bound, we can reconstruct the solution.

The provided real numbers can be viewed as a continued fraction (right). The rational solution we seek is some truncation of this fraction, which is unique, owing to the central fact presented above. We use a modified form of Euclid's gcd algorithm to determine the location for this truncation

$$x = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\cdots}}}}$$

$$a_0, a_1, \ldots \in \mathbb{Z}$$

Outlook

A table comparing solve time (s) for systems of various orders with an asymptotically equivalent algorithm, Dixon's p-adic lifting method [2].

	500	1000	1500	2000	2500	3000	3500	4000	4500
Dixon	0.765	5.163	16.99	43.25	79.29	117.05	181.96	271.5	374.3
Hybrid	0.403	2.867	10.07	27.69	47.97	72	116.14	179	252.1

Clearly, the efficiency that employing a numeric solver provides is worthwhile. Follow up work: a) using this method with specialized sparse solvers in place of dense numeric solvers. b) determining early termination conditions for both steps **I** and **II** to realize greater speedup**.**

[1] Zhendong Wan. An algorithm to solve integer linear systems exactly using numerical methods. Journal of Symbolic Computation, 2006.
[2] G.I. Malaschonok. Solution of Systems of Linear Equations by the p-adic Method. Programming and Computer Software, Vol. 29, 2003.