The Cost of Accurate Numerical Linear Algebra

or

Can we evaluate polynomials accurately?

James Demmel
Mathematics and Computer Science
UC Berkeley

Joint work with
Ioana Dumitriu, Olga Holtz

Plamen Koev, Yozo Hida, Ben Diament
W. Kahan, Ming Gu, Stan Eisenstat, Ivan Slapničar,
Krešimir Veselić, Zlatko Drmač

Supported by NSF and DOE

Outline

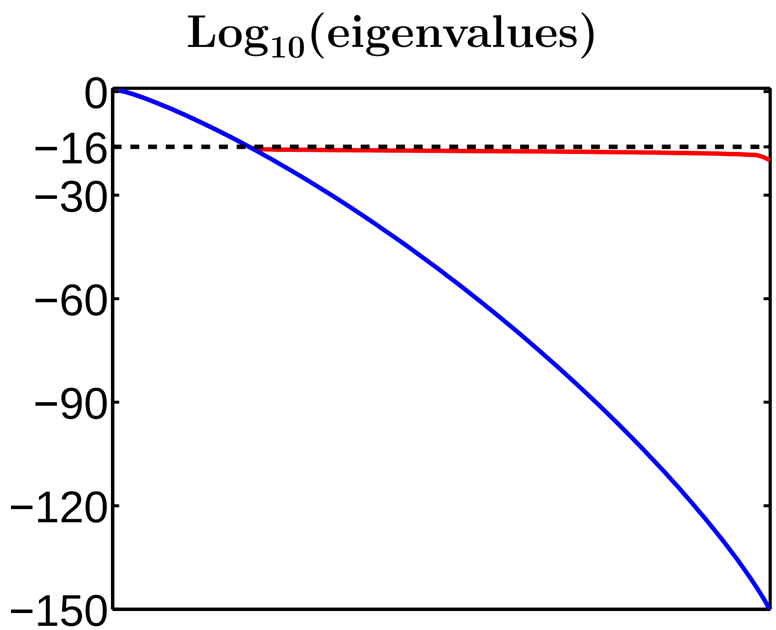
1. Motivation and Goals
2. What we can do in Traditional Model (TM) of arithmetic
3. What these example have in common:
   a condition for accurate evaluation in TM

Goal

- Compute $y = f(x)$ with floating point data x accurately and efficiently
- $f(x)$ may be
  - Rational function
  - Solution of linear system $Ay = b$
  - Solution of eigenvalue problem $Ay = \lambda y$ ...
- Accurately means with guaranteed relative error $e < 1$
  - $|y_{\text{computed}} - y| \le e \cdot |y|$
  - $e = 10^{-2}$ means 2 leading digits of y_{computed} correct
  - $y_{\text{computed}} = 0 = y$ must be exact
- Efficiently means in "polynomial time"
- Abbreviation: CAE means "Compute Accurately and Efficiently"

Example: 100 by 100 Hilbert Matrix $H(i,j) = 1/(i+j-1)$

- Eigenvalues range from 1 down to 10^{-150}
- Old algorithm, New Algorithm, both in 16 digit arithmetic



- Cost of Old algorithm in high enough precision $= O(n^3 D^2)$ where $D = \#\text{ digits} = \log(\lambda_{\max}/\lambda_{\min}) = \log \text{cond}(A) = 150$ decimal digits
- Cost of New algorithm $= O(n^3 \log D)$
- When D large, new algorithm exponentially faster
- New algorithm exploits structure of Cauchy matrices

Example: Adding Numbers in Traditional Model of Arithmetic

- $fl(a \otimes b) = (a \otimes b)(1 + \delta)$ where roundoff error $|\delta| \leq \epsilon \ll 1$
- How can we lose accuracy?
  - OK to multiply, divide, add positive numbers
  - OK to subtract exact numbers (initial data)
  - Accuracy may only be lost when subtracting approximate results:

```
   .12345xxx
 - .12345yyy
 -----------
   .00000zzz
```

- Thm: In Traditional Model it is impossible to add $x + y + z$ accurately
  - Proof sketch later
- Adding numbers represented as bits easier ...
  - Later

Structure of Results (1)

- Classes of rational expressions (matrices whose entries are expressions) that we can CAE depends strongly on Model of FP Arithmetic

  1. Traditional Model (TM for short):
     $fl(a \otimes b) = (a \otimes b)(1 + \delta)$ where $|\delta| \leq \epsilon \ll 1$
     no over/underflow
  2. Bit model: inputs are $m \cdot 2^e$, with "long exponents" e (LEM for short)
  3. Bit model: inputs are $m \cdot 2^e$, with "short exponents" e (SEM for short)

  4. Other models have been proposed (not today)
     (a) Blum/Shub/Smale
     (b) Cucker/Smale
     (c) Pour-El/Richards

Structure of Results (2)

- Classes of expressions (matrices) that we can CAE are described by factorizability properties of expressions (minors of matrices)

$$\mathbf{TM} \underset{\neq}{\subset} \mathbf{LEM} \underset{\neq ?}{\subset} \mathbf{SEM}$$

- New algorithms can be exponentially faster than conventional algorithms that just use high enough precision
- Cost(CAE in LEM) related to Cost(using symbolic computing)
- Cost(CAE in SEM) related to Cost(using integers)

Structure of Results (2)

- Classes of expressions (matrices) that we can CAE are described by factorizability properties of expressions (minors of matrices)

$$\mathbf{TM} \underset{\neq}{\subseteq} \mathbf{LEM} \underset{\neq ?}{\subseteq} \mathbf{SEM}$$

- New algorithms can be exponentially faster than conventional algorithms that just use high enough precision
- Cost(CAE in LEM) related to Cost(using symbolic computing)
- Cost(CAE in SEM) related to Cost(using integers)

- New results:
  - Necessary condition on polynomials for existence of algorithm for accurate evaluation in TM model
  - (Conjecture from ICM 2002 wrong)

Central Role of Minors

- Being able to CAE det(A) is necessary for CAE
  - $A = LDU$ with pivoting
  - $A = QR$
  - Eigenvalues λ_i of A ...
    - * Proof: $\det(A) = \pm \prod_i D_{ii} = \pm \prod_i R_{ii} = \prod_i \lambda_i = \cdots$
- Being able to CAE all minors of A is sufficient for CAE
  - A^{-1}
    - * Proof: Cramer's rule, only need $n^2 + 1$ minors
  - $A = LDU$ with pivoting
    - * Proof: Each entry of L, D, U a quotient of minors; $O(n^3)$ needed
  - Singular values of A (Square roots of eigenvalues of $A^T A$)
    - * Proof: $A = LDU$ with complete pivoting, then SVD of LDU
  - Eigenvalues of Totally Positive matrices (Koev)
- Similar result for pseudoinverse via minors of $\begin{bmatrix} I & A \\ A^T & 0 \end{bmatrix}$, etc.
- Examine which expressions (minors) we can CAE

Outline

1. **Motivation and Goals**
2. **What we can do in Traditional Model (TM) of arithmetic**
3. **What these example have in common:**
   a condition for accurate evaluation in TM

Cost of Accuracy in TM (1)

Matrix Type	det(A)	A^{-1}	Minor	GENP	GEPP	GECP	SVD	NENP	EVD
Cauchy									
TP Cauchy									
Vandermonde									
TP Vandermonde									
Confluent Vandermonde									
TP Confluent Vandermonde									
Vandermonde 3 Term Orth. Poly.									
Generalized Vandermonde									
TP Generalized Vandermonde									
Any TP									

GENP/PP/CP = Gaussian Elimination with No/Partial/Complete Pivoting
SVD = Singular Value Decomposition
NENP = Neville Elimination (bidiagonal factorization) with No Pivoting
EVD = Eigenvalue Decomposition

Cost of Accuracy in TM (2)

TP = Totally Positive (all minors nonnegative)

Matrix Type	
Cauchy	$C_{ij} = 1/(x_i + y_j)$
TP Cauchy	$x_i \nearrow$, $y_j \nearrow$, $x_1 + y_1 > 0$
Vandermonde	$V_{ij} = x_i^{j-1}$, x_i distinct
TP Vandermonde	$0 < x_i \nearrow$
Confluent Vandermonde	if some x_i coincide, differentiate rows of V
TP Confluent Vandermonde	$0 < x_i \nearrow$
Vandermonde 3 Term Orth. Poly.	$V_{ij} = P_j(x_i)$, P_j orthogonal polynomial from 3-term recurrence
Generalized Vandermonde	$G_{ij} = x_i^{\lambda_j + j - 1}$, λ_j nonnegative increasing integer sequence
TP Generalized Vandermonde	$0 < x_i \nearrow$
Any TP	Given by its Neville Factorization

Cost of Accuracy in TM
Known results + New Results

Matrix Type	det(A)	A^{-1}	Minor	GENP	GEPP	GECP	SVD	NENP	EVD
Cauchy	n^2	n^2	n^2	n^2	n^2	n^3	n^3	n^2	
TP Cauchy	n^2	n^2	n^2	n^2	n^2	n^3	n^3	n^2	n^3
Vandermonde	n^2	No	No	No	No	No	n^3	n^2	
TP Vandermonde	n^2	n^3	exp	n^2	n^2	exp	n^3	n^2	n^3
Confluent Vandermonde	n^2	No	No	No	No	No		n^2	
TP Confluent Vandermonde	n^2	n^3		n^3			n^3	n^2	n^3
Vandermonde 3 Term Orth. Poly.	n^2						n^3		
Generalized Vandermonde	No	No	No	No	No	No		No	
TP Generalized Vandermonde	Λn^2	Λn^3	exp	Λn^2	Λn^2	exp	Λn^3	Λn^2	Λn^3
Any TP	n	n^3	exp	n^3	exp	exp	n^3	0	n^3

Other examples in Traditional Model

- Diagonal * Totally Unimodular (TU) * Diagonal
  - TU $\Leftrightarrow$ each minor $\in \{0, \pm 1\}$
  - Poincaré: Signed incidence matrix on graph $\Rightarrow$ TU
  - Includes 2nd centered difference approximations to Sturm-Liouville equations and elliptic PDEs on uniform meshes
  - One-line change to GECP makes it accurate, then SVD, EVD
- Sparse matrices with
  - Acyclic sparsity patterns, GECP cost $= O(n^3)$
  - Particular sparsity and sign patterns ("Total Sign Compound") GECP Cost $= O(n^4)$
- Weakly Diagonally Dominant (WDD) M-Matrices
  - M-matrix: off-diagonal $A_{ij} < 0$, all $(A^{-1})_{ij} > 0$
  - WDD: nonnegative row sums $s_i = \sum_j A_{ij} \geq 0$
  - Modify GECP to update s_i, off-diagonal A_{ij}, cost $= O(n^3)$
- What do these examples have in common?

Outline

1. Motivation and Goals
2. What we can do in Traditional Model (TM) of arithmetic
3. What these example have in common:
   a condition for accurate evaluation in TM

What do all these examples have in common?

- Recall models of computation:
  - Traditional Model (TM):
    $fl(a \otimes b) = (a \otimes b)(1+\delta)$ where $|\delta| \le \epsilon < 1$, δ real
  - Long Exponent Model (LEM): inputs are $m \cdot 2^e$, with "long" e
  - Short Exponent Model (SEM): inputs are $m \cdot 2^e$, with "short" e
- Goals: Given choice of model
  - Decide if $\exists$ algorithm $alg(x, \delta)$ to evaluate multivariate polynomial $p(x)$ with small relative error on domain $\mathcal{D}$:

    $\forall\ 0 < \eta < 1$ ... η = desired relative error

    $\exists\ 0 < \epsilon < 1$ ... ϵ = maximum rounding error

    $\forall\ x \in \mathcal{D}$ ... for all x in the domain

    $\forall\ |\delta_i| \le \epsilon$ ... for all rounding errors bounded by ϵ

    $|alg(x,\delta) - p(x)| \le \eta \cdot |p(x)|$ ... relative error is at most η
  - If so, is there a polynomial-time algorithm?
  - Given $p(x)$ and $\mathcal{D}$, seek effective procedure to exhibit algorithm, or show one does not exist

What is known about existence of accurate algorithms?

- Depends on
  - Choice of model (TM, LEM, SEM)
  - TM needs more details to be formal
  - How $p(x)$ presented (explicit, determinant, ...)
- Existence of accurate algorithm
  - Bit Models (LEM and SEM): An accurate algorithm always exists
  - TM: may or may not exist
    We show current progress towards a decision procedure
- Existence of polynomial-time accurate algorithm
  - TM $\subsetneq$ LEM $\underset{\neq ?}{\subseteq}$ SEM

Formalizing an Algorithm under Traditional Model

- **Numerical operations included**
  - Could include $\pm$, $\times$, $\div$, unary $-$, ...
  - We omit $\div$ (restrictive?)
  - We say unary $-$ is exact (true in practice)
- **Comparison and Branching**
  - Assume branching on exact comparisons $a > b$, $c \leq d$, ...
  - Will sketch proof in nonbranching case
- **Determinism**
  - Is $3 + 7$ same no matter where computed?
  - Will assume nondeterministic for now (try to include later...)
- **Available constants**
  - With $\sqrt{2}$, could compute $x^2 - 2 = (x - \sqrt{2}) \times (x + \sqrt{2})$ accurately, else not
  - Will sketch proof when no constants
  - Limits us to integer coefficients, zero constant term in $p(x)$
    - Replace $2 \times x$ by $x + x$, etc.
    - No loss of generality for homogeneous polynomials, integer coeffs

Recognizing Accuracy

- Ex: Compute $p(x) = x_1 + x_2 + x_3$
  - Try $alg(x, \delta) = ((x_1 + x_2)(1 + \delta_1) + x_3)(1 + \delta_2)$
  - $rel\_err(x, \delta) = \frac{alg(x,\delta) - p(x)}{p(x)} = \frac{x_1 + x_2}{x_1 + x_2 + x_3}(\delta_1 + \delta_2 + \delta_1 \cdot \delta_2) + \frac{x_3}{x_1 + x_2 + x_3}(\delta_2)$
  - $\forall \epsilon > 0$, $rel\_err(x, \delta)$ unbounded on an open subset of (x, δ) with $|\delta_i| < \epsilon$
- Generally: $rel\_err(x, \delta) = \sum_r \frac{p_r(x)}{p(x)} \cdot q_r(\delta)$
  - Each $\frac{p_r(x)}{p(x)}$ must be bounded near $p(x) = 0$
- Ex: $p(x)$ positive definite and homogeneous, degree d
  - If $p_r(x)$ also homogeneous, degree d, then $\frac{p_r(x)}{p(x)}$ bounded
  - Holds if all intermediate results are homogeneous

Examples

- $M_2(x, y, z) = z^6 + x^2 \cdot y^2 \cdot (x^2 + y^2 - 2 \cdot z^2)$
  - Positive definite and homogenous, easy to evaluate accurately
- $M_3(x, y, z) = z^6 + x^2 \cdot y^2 \cdot (x^2 + y^2 - 3 \cdot z^2)$
  - Motzkin polynomial, nonnegative, zero at $|x| = |y| = |z|$

$$
\begin{array}{ll}
\text{if} & |x-z| \le |x+z| \wedge |y-z| \le |y+z| \\
p = & z^4 \cdot [4((x-z)^2 + (y-z)^2 + (x-z)(y-z))] + \\
& +z^3 \cdot [2(2(x-z)^3 + 5(y-z)(x-z)^2 + 5(y-z)^2(x-z) + \\
& \qquad 2(y-z)^3)] + \\
& +z^2 \cdot [(x-z)^4 + 8(y-z)(x-z)^3 + 9(y-z)^2(x-z)^2 + \\
& \qquad 8(y-z)^3(x-z) + (y-z)^4] + \\
& +z \cdot [2(y-z)(x-z)((x-z)^3 + 2(y-z)(x-z)^2 + \\
& \qquad 2(y-z)^2(x-z) + (y-z)^3] + \\
& +(y-z)^2(x-z)^2((x-z)^2 + (y-z)^2) \\
\text{else} & \ldots\ 2^{\#\text{vars}-1} \text{ more analogous cases}
\end{array}
$$

- $M_4(x, y, z) = z^6 + x^2 \cdot y^2 \cdot (x^2 + y^2 - 4 \cdot z^2)$
  - Impossible to evaluate accurately

Allowable Sets

- Define basic *allowable sets*
  - $Z_i = \{x : \ x_i = 0\}$
  - $S_{ij} = \{x : \ x_i + x_j = 0\}$
  - $D_{ij} = \{x : \ x_i - x_j = 0\}$
- Def: A set is *allowable* if it can be written as an arbitrary union and intersection of basic allowable sets (plus null set, R^n)
- Def: Allow(x) is the smallest allowable set containing x

$$\mathrm{Allow}(x) = \mathrm{R}^n \cap (\cap_{i:\ x_i=0} Z_i) \cap (\cap_{i,j:\ x_i+x_j=0} S_{ij}) \cap (\cap_{i,j:\ x_i-x_j=0} D_{ij})$$

- Ex: $\mathrm{Allow}((0, 1, -1, 2)) = Z_1 \cap S_{23}$
- We say $p(x)$ allowable if its variety $V(p)$ is allowable
- If $p(x)$ not allowable, then

$$G(p) \equiv V(p) - \cup A$$

  is nonempty, where the union is over all allowable sets A contained in $V(p)$
- Def: $G(p)$ called the set of points in "general position" in $V(p)$

Existence of an Accurate Algorithm in TM

- Consider algorithms that
  - Include $\pm$, $\times$, branching
  - $\pm$ and $\times$ incur $1+\delta$ errors
  - Comparisons and unary negation are exact
  - No branching on η, ϵ
  - No explicit constants (limits results to integer coefficients, no constant term)
  - Nondeterministic rounding errors
  - Domain $\mathcal{D} = \mathrm{R}^n$
- Theorem: A necessary condition for the existence of an accurate algorithm to evaluate $p(x)$ on R^n is that $V(p)$ be allowable.

Examples

- $p(x, y, z) = x + y + z$ not allowable (D., Koev)
- $M_2(x, y, z) = z^6 + x^2 \cdot y^2 \cdot (x^2 + y^2 - 2 \cdot z^2)$ is allowable: $V(M_2) = \{0\}$
- $M_3(x, y, z) = z^6 + x^2 \cdot y^2 \cdot (x^2 + y^2 - 3 \cdot z^2)$ is allowable: $V(M_3) = \{|x| = |y| = |z|\}$.
- $M_4(x, y, z) = z^6 + x^2 \cdot y^2 \cdot (x^2 + y^2 - 4 \cdot z^2)$ is unallowable
- Allowable $V(p)$ *not* a sufficient condition for an accurate algorithm: $p(x, y, z, w) = w^4 + w^2 \cdot (x + y + z)^2$ has allowable $V(p) = \{w = 0\}$, but (apparently) can't be evaluated accurately

Proof Sketch (1):

- Assume no branching for simplicity
- Let $alg(x, \delta)$ denote result of any computation.
- Main Lemma: Choose any x. One of following two cases must hold:
  1. $alg(x, \delta)$ is nonzero at x for all δ in a Zariski-open set
  2. $alg(y, \delta) = 0$ for all $y \in \text{Allow}(x)$ and all δ
- Suppose $V(p)$ not allowable. Choose any $x \in G(p) \subset V(p)$. Then either
  1. $alg(x, \delta)$ is nonzero at x for all δ in a Zariski-open set
     but $p(x) = 0$, so the relative error is ∞
  2. $alg(y, \delta) = 0$ for all $y \in \text{Allow}(x)$ and all δ
     but $p(y) \neq 0$ a.e., so the relative error is 1

Proof Sketch (2): Main Lemma

- Main Lemma: Choose any x. One of following two cases must hold:
  1. $alg(x, \delta)$ is nonzero at x for all δ in a Zariski-open set
  2. $alg(y, \delta) = 0$ for all $y \in \text{Allow}(x)$ and all δ
- For simplicity, suppose no branching, no data reuse, nondeterminism
  - Implies that $alg(x, \delta)$ can be represented as a graph:
    * Source nodes representing data x_i, output edges connected to ...
    * Computational nodes, arranged in a tree, of following kinds:
      · 2-inputs, producing $fl(a \otimes b) = (a \otimes b)(1 + \delta_{\text{node}})$ $(\otimes \in \{+, -, \times\})$ with independent $|\delta_{\text{node}}| \leq \epsilon$ for each node
      · 1-input, producing $fl(x \otimes x) = (x \otimes x)(1 + \delta_{\text{node}})$ (note: $fl(x - x) = 0$ exactly)
      · 1-input, producing $-x$ exactly
    * Destination node, one input, no output

Proof Sketch (3): Main Lemma (cont)

- Main Lemma: Choose any x. One of following two cases must hold:

  1. $alg(x, \delta)$ is nonzero at x for all δ in a Zariski-open set
  2. $alg(y, \delta) = 0$ for all $y \in \text{Allow}(x)$ and all δ

- Def: Choose x. Call computational node "nontrivial" if it

  – Computes $fl(a \pm b)$, both a and b nonzero as polynomials in δ

  – At least one of a and b not an input x_i

- Lemma: Output of all nontrivial nodes nonzero on Zariski-open set of δ
- If ultimate output is from nontrivial node, done (Case 1)
- Otherwise, "trace back" zero output through tree as far as possible
- Can show (case analysis) that zero must result from one of

  – $x_i = 0$ (allowable)

  – $x_i \pm x_j = 0$ (allowable)

  – $x - x$ or $x + (-x)$ (in which case $alg(x, \delta) \equiv 0$)

- In any case, $alg(y, \delta)$ must be zero on Allow(x) (Case 2)

Other results and Future Work

- Large relative error, if it occurs, occurs on open set of (x, δ)
  - So hard problems not of measure zero
- Want to incorporate
  - Determinism (simulate deterministic machine by nondeterministic one)
  - Constants (add $\{x : x_i \pm \alpha = 0\}$ to basic allowable sets for constant α)
  - Domain $\mathcal{D}$ limited to (allowable?) semialgebraic sets
  - Division and rational functions
- Complete decision procedure, just not necessary or sufficient conditions
  - Since $p(x) = x_1^{2n} + x_1^2 \cdot (q(x_2, .., x_n))^2$ has $V(p) = \{x : x_1 = 0\}$, behavior of $q()$ "hidden"
  - Need to inductively "unfold" $V(p)$
- Extend to complex arithmetic, interval arithmetic
- Perturbation theory
  - Conj: Accurate evaluation possible iff condition number can have certain singularities

In Contrast: Adding Numbers in Bit Model of Arithmetic

- $x = m \cdot 2^e$ where m and e are integers, m at most b bits
- $fl(x+y)$ is correctly rounded result
- Cancellation is obstable to accuracy:
  - $(2^e + 1) - 2^e$ requires e bits of intermediate precision
  - Not polynomial time!
- "Sort and Sum" Algorithm for $S = \sum_{i=1}^{n} x_i$

  Sort so $|e_1| \geq |e_2| \geq \cdots \geq |e_n|$ ... $|x_1| \geq \cdots \geq |x_n|$ more than enough
  $S = 0$ ... $B > b$ bits
  for $i = 1$ to n
  $\quad S = S + x_i$

- Thm: Let $N = 1 + 2^{B-b} + 2^{B-2b} + \cdots 2^{B \bmod b} = 1 + \lceil \frac{2^{B-b}}{1-2^{-b}} \rceil$. Then
  - If $n \leq N$, then S accurate to nearly b bits, despite any cancellation
  - If $n \geq N + 2$, then S may be completely wrong (wrong sign)
  - If $n = N + 1$, more cases ...
- Ex: x_i double ($b = 53$), S extended ($B = 64$) $\Rightarrow N = 2049$

Conclusions

- **We have identified many classes of floating point expressions and matrix computations that permit**
  - **Accurate solutions: relative error < 1**
  - **Efficient solutions: time = poly(input size)**
- **Explored 3 natural models of arithmetic**
  - **Traditional Model (TM)**
  - **Long Exponent Model (LEM)**
  - **Short Exponent Model (SEM)**
- **New efficient algorithms for each: TM $\subsetneq$ LEM $\subsetneq ?$ SEM**
- **New necessary condition for existence of accurate algorithm to evaluate $p(x)$ in TM – working on effective decision procedure**
- **Lots of open problems**
- **For more information see**
  - `www.cs.berkeley.edu/~demmel`
  - `math.mit.edu/~plamen`

Extra Slides

Traditional Model - What we can do

- **What do all these examples have in common?**
- **Goal: evaluate homogeneous polynomial $f(x)$ accurately on domain $\mathcal{D}$**
- **Property A: $f = \Pi_m f_m$ where each factor f_m satisfies one of**
  1. **f_m of the form x_i, $x_i - x_j$ or $x_i + x_j$, or**
  2. **$|f_m|$ bounded away from 0 on $\mathcal{D}$**
- **Conjecture 1: f satisfies Prop. A iff $f(x)$ can be evaluated accurately**
- **Conjecture 2: f satisfies Prop. A iff $f(x)$ has a *relative perturbation theory*:**
  - **relative error in output = O($\kappa_{rel}\cdot$ relative error in input)**
  - **$\kappa_{rel} = O(1/\min \frac{|x_i \pm x_j|}{|x_i|+|x_j|}) = O(1/$ smallest relative gap among inputs)**
  - **Tiny outputs often well conditioned**
  - **Relative perturbation theory justifies computing them!**

Traditional Model - What we can do

- **What do all these examples have in common?**
- **Goal: evaluate homogeneous polynomial $f(x)$ accurately on domain $\mathcal{D}$**
- **Property A: $f = \Pi_m f_m$ where each factor f_m satisfies one of**
  1. **f_m of the form x_i, $x_i - x_j$ or $x_i + x_j$, or**
  2. **$|f_m|$ bounded away from 0 on $\mathcal{D}$**
- **Conjecture 1: f satisfies Prop. A iff $f(x)$ can be evaluated accurately**
- **Conjecture 2: f satisfies Prop. A iff $f(x)$ has a *relative perturbation theory*:**
  - **relative error in output = O($\kappa_{rel}\cdot$ relative error in input)**
  - **$\kappa_{rel} = O(1/\min \frac{|x_i \pm x_j|}{|x_i|+|x_j|}) = O(1/$ smallest relative gap among inputs)**
  - **Tiny outputs often well conditioned**
  - **Relative perturbation theory justifies computing them!**
- **WRONG**
  - **Conjecture only true in "if" direction**
  - **$w^4 + w^2 \cdot (x+y)^2$ ok**
  - **$w^4 + w^2 \cdot (x+y+z)^2$ not ok**
  - **Both irreducible with same real variety $\{w = 0\}$**

Bit Models of Arithmetic

- Inputs of form $x = m \cdot 2^e$, e and m integers
- size(x) = # bits used to represent x = #bits(m) + #bits(e)
- Can evaluate any rational expression accurately
  - Convert to poly/poly, using high enough precision
  - Question is cost
- Cost depends strongly on #bits(e)
  - Short Exponent Model (SEM)
    * #bits(e) = O(log(#bits(m)))
    * Equivalent to integer arithmetic
    * Can CAE many problems
  - Long Exponent Model (LEM)
    * #bits(e) and #bits(m) independent
    * Natural model for algorithm design
    * Like symbolic algebra, which is much harder

Differences between Short and Long Exponent Models

- **SEM and integer arithmetic "equivalent"**
  - **Represent $m \cdot 2^e$ as integer with**
    $\#\text{bits} = \#\text{bits}(m) + e \approx \#\text{bits}(m) + 2^{\#\text{bits}(e)} = \text{poly}(\#\text{bits}(m))$
  - **Any minor of any SEM matrix A computable accurately in poly time**
    - **Use Clarkson's Algorithm**
  - **Can do accurate linear algebra in polynomial time**
- **LEM and integer arithmetic not equivalent**
  - **$\Pi_{i=1}^{n}(1 + x_i)$ can have exponentially more bits if x_i LEM than SEM**
  - **Getting arbitrary bit of $\Pi_{i=1}^{n}(1 + x_i)$ as hard as permanent**
  - **Testing if an LEM matrix is singular may not be in NP**
  - **For efficiency, matrices need structure**
- **Cond(A) in LEM can be exponentially larger than in SEM**
  - **SEM: $\log \text{cond}(A)$ is poly(size(A))**
  - **LEM: $\log \text{cond}(A)$ can be exponential in size(A)**

Which FP Expressions can we CAE in the Long Exponent Model (LEM)?

- Def: $r(x)$ is in factored form if it is written as explicit product of sparse polynomials
  - E.g.: *not* as determinant of general matrix
- Def: size(r) = #bits to write down r
- Theorem: We can CAE r in time poly(size(r))
  - Compute monomials in each sparse polynomial exactly
  - Add them in decreasing order by magnitude, with rounding (see Hida's talk)
- Def: A family $A_n(x)$ of n-by-n rational matrices is polyfactorable if each minor $r(x)$ is in factored form of size size(r) = O(poly(n))
- Theorem: Suppose $A_n(x)$ is polyfactorable. Then in the LEM we can CAE LU with pivoting, A^{-1}, singular values.

Summary of differences between Arithmetic Models

- **What can we CAE in LEM that we could not in TM?**
  - **Rational Expressions**
    - **LEM: anything in factored form**
    - **TM: not $x + y + z$ or any expression with nontrivial real variety**
  - **Matrix computations: polyfactorable matrices**
    - **Take any $A(x)$ that we can CAE in TM, substitute $x_i = \text{poly}_i(y_1, ..., y_n)$**
    - **Green's matrices (inverses of tridiagonals, represented as $A_{ij} = x_i y_j$)**
- **What can we CAE in SEM that we could not in LEM?**
  - **$\det(A)$ where each A_{ij} is a general floating point number**

Open Questions

- Are there FP expressions that we provably cannot CAE in LEM?
  - $\Pi_{i=1}^{n}(1 + x_i) - \Pi_{j=1}^{n}(1 + y_j)$
  - Determinant of general (or just tridiagonal) matrix
- What changes if we have sign information?
  - We have accurate algorithms for all TP matrices, but not efficient
  - How big a class of TP matrices can we do efficiently?
    (see Koev's talk)
- Differential equations
  - Only simplest ones understood (eg M-matrices)
  - What about other discretizations?
  - Conjecture: Accuracy depends only on geometry, not material properties
- Accuracy of singular vectors, eigenvectors
- What about nonsymmetric eigenproblem?

Conclusions

- **We have identified many classes of floating point expressions and matrix computations that permit**
  - **Accurate solutions: relative error < 1**
  - **Efficient solutions: time = poly(input size)**
- **Explored 3 natural models of arithmetic**
  - **Traditional Model (TM)**
  - **Long Exponent Model (LEM)**
  - **Short Exponent Model (SEM)**
- **New efficient algorithms for each**
- **TM $\subsetneq$ LEM $\overset{\subseteq}{\neq ?}$ SEM**
- **Lots of open problems**
- **For more information see**
  - `www.cs.berkeley.edu/~demmel`
  - `www-math.mit.edu/~koev`