Reliability of numerical algorithms : structured pseudosolutions and accuracy

Stef Graillat

Université de Perpignan Via Domitia
Laboratoire LP2A
Équipe de recherche en Informatique DALI

November, 30th 2005

Advisor: Philippe Langlois

Which problems to solve with numerical algorithms?

Main problems in numerical polynomial and linear algebra

- Polynomial evaluation
  - Newton's method, interpolation, ...
- Computation of zeros of polynomial, polynomial systems
  - computer aided design, robotics, ...
- Solving linear systems
  - finite element method for PDE, ...
- Computation of eigenvalues, eigenvectors of matrices
  - stability in control theory, PageRank (Google), ...

Real problems and implemented algorithms are uncertain

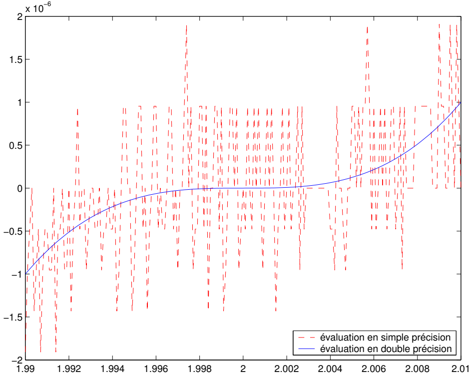
Solving the previous problems suffers from two difficulties:

- Uncertainties in the data
  - influence on the zeros: pseudozeros
  - influence real/complex perturbations
  - influence of the structure in some matrix problems
- Uncertainties in the computation: finite precision
  - for the polynomial evaluation

How to deal with such uncertainties?

Loss of accuracy in the polynomial evaluation

Evaluation of the polynomial $p(x) = (x-2)^3 = x^3 - 6x^2 + 12x - 8$ for about 200 points near $x = 2$ in single and double precision



Problems in finite precision computation

Aims : Solving the previous problems being accurate and reliable

- Understanding the influence of the finite precision on the numerical quality of numerical software
  - inaccurate results;
  - numerical instabilities.
- controlling and limiting harmful effect

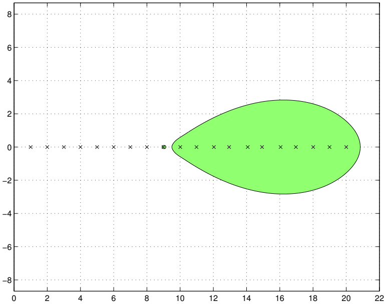
How to be more accurate without large overheads?

Data known with uncertainties

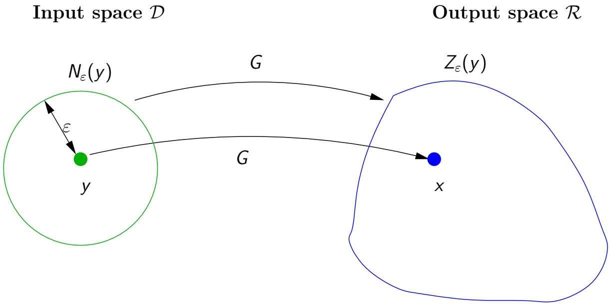
Computing the zeros of the Wilkinson polynomial of degree 20

$$\begin{aligned} W(x) &= (x-1)(x-2)\cdots(x-20) \\ &= x^{20} - 210x^{19} + \cdots + 20! \end{aligned}$$

Uncertainty of 2^{-23} on the coefficient of x^{19}



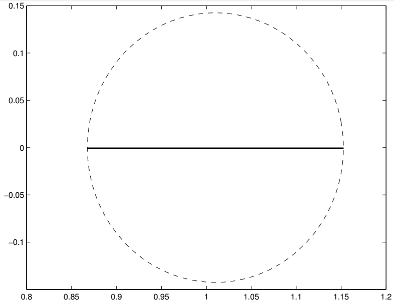
How to deal with uncertainties on the data?



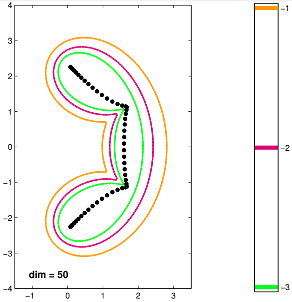
- computation of polynomial zeros $\longrightarrow$ pseudozeros
- computation of eigenvalues $\longrightarrow$ pseudospectra

Does the notion of pseudosolutions enable us to solve some problems?

Influence of the structure of perturbations



real and complex pseudozeros of $p(z) = z - 1$ for $\varepsilon = 0.1$



Pseudospectra of a Toeplitz matrix

$$\begin{pmatrix} t_0 & t_{-1} & \cdots & t_{1-n} \\ t_1 & t_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_{-1} \\ t_{n-1} & \cdots & t_1 & t_0 \end{pmatrix}$$

Different types and sizes of perturbations

Influence of the size and the structure of perturbations

- Structured perturbations:
  - polynomials: real coefficients
  - matrices: symmetric, Toeplitz, Hankel, circulant, ...

  Does the taking into account of the structure enable us to improve the accuracy and stability of algorithms?
- Size of perturbations:
  - infinitely small $\longrightarrow$ condition number
  - finite $\longrightarrow$ backward error, pseudosolutions

Notion of structured condition number, real pseudozeros and structured pseudospectra

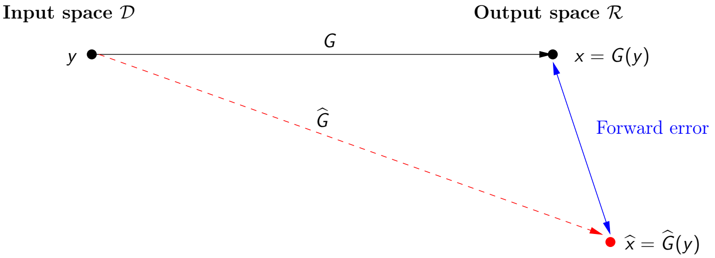
Problems in computing with uncertainties

Understanding the difficulties to deal with uncertainties:

- Controlling the effects of uncertainties:
  - How to measure the difficulty of solving the problem?
  - How to appreciate the reliability of the algorithm?
  - How to estimate the accuracy of the computed solution?
- Limiting the effect of finite precision
  - How to improve the accuracy of the solution?

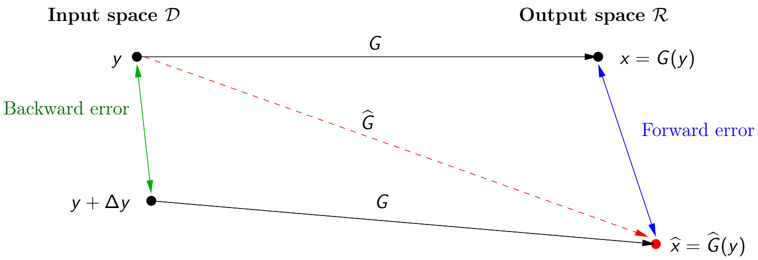
Which notions to answer these questions?

Error analysis



- Forward error analysis
- Backward error analysis
  Identify $\widehat{x}$ as the solution of a perturbed problem:
  $\widehat{x} = G(y + \Delta y)$.

Error analysis



- Forward error analysis
- Backward error analysis
  Identify $\widehat{x}$ as the solution of a perturbed problem:
  $\widehat{x} = G(y + \Delta y)$.

Advantages of backward error analysis

- **How to estimate the accuracy of the computed solution?**
  At the first order, we have the rule of thumb:

  $$\text{forward error} \lesssim \text{condition number} \times \text{backward error.}$$

- **How to measure the difficulty of solving the problem ?**
  Condition number measures the sensitivity of the solution to perturbation in the data

  $$\text{Condition number} : K(P, y) := \lim_{\varepsilon \to 0} \sup_{\Delta y \in \mathcal{P}(\varepsilon)} \left\{ \frac{\|\Delta x\|_{\mathcal{R}}}{\|\Delta y\|_{\mathcal{D}}} \right\}$$

- **How to appreciate the reliability of the algorithm?**
  Backward error measures the distance between the problem we solved and the initial problem.

  $$\text{Backward error} : \eta(\widehat{x}) = \min_{\Delta y \in \mathcal{D}} \{\|\Delta y\|_{\mathcal{D}} : \widehat{x} = G(y + \Delta y)\}$$

Advantages of backward error analysis

- **How to estimate the accuracy of the computed solution?**
  At the first order, we have the rule of thumb:

  $$\text{forward error} \lesssim \text{condition number} \times \text{backward error.}$$

- **How to measure the difficulty of solving the problem ?**
  Condition number measures the sensitivity of the solution to perturbation in the data

  $$\text{Condition number}: K(P,y) := \lim_{\varepsilon \to 0} \sup_{\Delta y \in \mathcal{P}(\varepsilon)} \left\{ \frac{\|\Delta x\|_{\mathcal{R}}}{\|\Delta y\|_{\mathcal{D}}} \right\}$$

- **How to appreciate the reliability of the algorithm?**
  Backward error measures the distance between the problem we solved and the initial problem.

  $$\text{Backward error}: \eta(\widehat{x}) = \min_{\Delta y \in \mathcal{D}} \{\|\Delta y\|_{\mathcal{D}} : \widehat{x} = G(y + \Delta y)\}$$

Advantages of backward error analysis

- **How to estimate the accuracy of the computed solution?**
  At the first order, we have the rule of thumb:

  $$\text{forward error} \lesssim \text{condition number} \times \text{backward error.}$$

- **How to measure the difficulty of solving the problem ?**
  Condition number measures the sensitivity of the solution to perturbation in the data

  Condition number : $K(P, y) := \lim_{\varepsilon \to 0} \sup_{\Delta y \in \mathcal{P}(\varepsilon)} \left\{ \frac{\|\Delta x\|_{\mathcal{R}}}{\|\Delta y\|_{\mathcal{D}}} \right\}$

- **How to appreciate the reliability of the algorithm?**
  Backward error measures the distance between the problem we solved and the initial problem.

  Backward error : $\eta(\widehat{x}) = \min_{\Delta y \in \mathcal{D}} \{ \|\Delta y\|_{\mathcal{D}} : \widehat{x} = G(y + \Delta y) \}$

Outline

Pseudozeros: definition (1/2)

$\mathcal{P}_n$: polynomials of $\mathbb{C}[z]$ of degree at most n
$\mathcal{M}_n$: monic polynomials of $\mathcal{P}_n$ of degree n

$$p(z) = \sum_{i=0}^{n} p_i z^i, \quad \|p\| = (\sum_{i=0}^{n} |p_i|^2)^{1/2}$$

Definition 1 (Perturbation)

Neighborhood of polynomial $p \in \mathcal{M}_n$

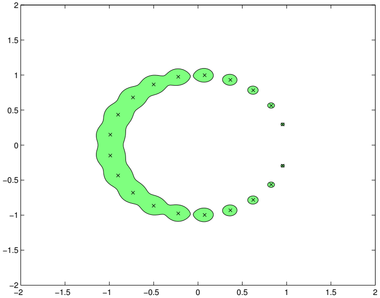
$$N_\varepsilon(p) = \{\widehat{p} \in \mathcal{M}_n : \|p - \widehat{p}\| \leq \varepsilon\}$$

Pseudozeros: definition (2/2)

Definition 2 (ε-pseudozero set)

$$Z_\varepsilon(p) = \{z \in \mathbb{C} : \widehat{p}(z) = 0 \text{ for } \widehat{p} \in N_\varepsilon(p)\}$$

$$p(z) = 1 + z + z^2 + \cdots + z^{20}$$
with $\varepsilon = 0.3$



Pseudozeros are computable

Theorem 1 (Trefethen and Toh, 1994)

The ε-pseudozero set satisfies

$$Z_\varepsilon(p) = \left\{ z \in \mathbb{C} : g(z) := \frac{|p(z)|}{\|\underline{z}\|} \le \varepsilon \right\},$$

where $\underline{z} = (1, z, \dots, z^{n-1})$.

Algorithm 1 (Drawing of ε-pseudozero set)

1. We mesh a square containing all the pseudozeros of p (MATLAB command: `meshgrid`).
2. We compute $g(z) := \frac{|p(z)|}{\|\underline{z}\|}$ for all the nodes z of the grid.
3. We plot the contour level $|g(z)| = \varepsilon$ (MATLAB command: `contour`).

History of pseudozero set

- Mosier (1986) : definition and study for the ∞-norm.
- Trefethen and Toh (1994) : study for the 2-norm.
  pseudozeros $\approx$ pseudospectra of the companion matrix.
- Zhang (2001) : use pseudozero as a tool to study condition number for the polynomial evaluation.
- Stetter (2004) : *Numerical Polynomial Algebra* (SIAM). General framework for working with polynomials only known with uncertainties

Can we use pseudozero sets to solve some problems?

Stability of polynomials

Definition 3

A polynomial is stable if all its zeros have negative real part.

The function *abscissa* $a : \mathcal{P} \to \mathbb{R}$ is defined by

$$a(p) = \max\{\mathrm{Re}(z) : p(z) = 0\}.$$

A polynomial p is stable $\iff a(p) < 0$

Motivations

In control theory, transfer function are often written as $H(p) = \frac{N(p)}{D(p)}$ where N and D are polynomials.

The system is stable if D is a stable polynomial

Question : If D is stable, is it still stable when perturbed?

(we assume that D is monic)

Pseudozero abscissa mapping

Definition 4

ε-pseudozero abscissa mapping $a_\varepsilon : \mathcal{P}_n \to \mathbb{R}$:

$$a_\varepsilon(p) = \max\{\mathrm{Re}(z) : z \in Z_\varepsilon(p)\}.$$

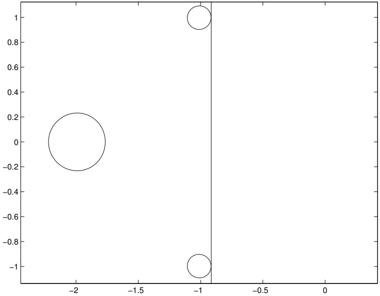
A polynomial p is ε-robustly stable $\iff a_\varepsilon(p) < 0$

Statement of the problem:

Given a polynomial $p \in \mathcal{M}_n$ and $\varepsilon > 0$, let us compute $a_\varepsilon(p)$.

A plotting algorithm

- Draw the ε-pseudozero set
- Draw the vertical line that intersects the right-most point within the ε-pseudozero set



ε-pseudozero set of $p(z) = z^3 + 4z^2 + 6z + 4$ for $\varepsilon = 0.1$

$$a_\varepsilon(p) \approx -0.9$$

Our solution

The results

- an algorithm computing $a_\varepsilon(p)$ with a tolerance τ
- a drawing of the ε-pseudozero set
  $\longrightarrow$ qualitative analysis of the result
  $\longrightarrow$ visualization of the result

Tools

- an explicit formula that defines the pseudozero set
- the continuous dependency of the zeros w.r.t the polynomial coefficients
- the Sturm sequences to count the number of real zeros

Another characterization of the pseudozero set

Let us denote $h_{p,\varepsilon} : \mathbb{R}^2 \to \mathbb{R}$ the function

$$h_{p,\varepsilon}(x,y) = |p(x+iy)|^2 - \varepsilon^2 \sum_{j=0}^{n-1} (x^2+y^2)^j.$$

Then

$$Z_\varepsilon(p) = \{(x,y) \in \mathbb{R}^2 : h_{p,\varepsilon}(x,y) \le 0\}$$

$\Longrightarrow$ $h_{p,\varepsilon}(\cdot, y)$ and $h_{p,\varepsilon}(x, \cdot)$ are polynomials of degree $2n$.

Theorem 2

For any real $x \ge a(p)$, $x \le a_\varepsilon(p)$ if and only if the equation $h_{p,\varepsilon}(x,y) = 0$ has a real solution y.

A symbolic-numerical bisection algorithm

Algorithm 2

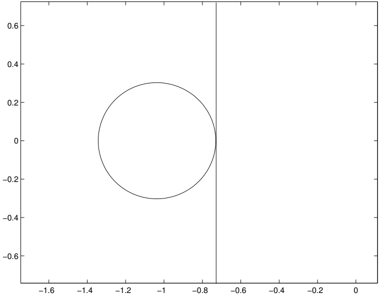
Require: a stable polynomial p, the parameter ε, the tolerance τ on the accuracy of $a_\varepsilon(p)$

Ensure: a number α such that $|\alpha - a_\varepsilon(p)| \leq \tau$

```
γ := a(p),    δ := ‖p‖ + ε
while |γ − δ| > τ do
    x := (γ+δ)/2
    if the equation h_{p,ε}(x, y) = 0 has a solution y real then
        δ := x
    else
        γ := x
    end if
end while
return α = (γ+δ)/2
```

Numerical simulation

For $p(z) = z^5 + z^4 + 10z^3 + 10z^2 + 5z + 1$, $\varepsilon = 0.001$ and $\tau = 0.00001$ the algorithm gives $a_\varepsilon(p) \approx -0.719669$



ε-pseudozero set of $p(z) = z^5 + z^4 + 10z^3 + 10z^2 + 5z + 1$ for $\varepsilon = 0.001$

Outline

1. Motivations
2. Pseudozeros and application in control theory
3. Accurate polynomial evaluation
4. Other results
   - Real perturbations
   - Influence of the structure
5. Summary and future work

Floating point number

Floating point system $\mathbb{F} \subset \mathbb{R}$:

$$x = \pm \underbrace{x_0.x_1 \ldots x_{p-1}}_{mantissa} \times b^e, \quad 0 \leq x_i \leq b-1, \quad x_0 \neq 0$$

b : basis, p : precision, e : exponent range s.t. $e_{\min} \leq e \leq e_{\max}$

Machine epsilon $\epsilon = b^{1-p}$, $|1^+ - 1| = \epsilon$

Approximation of $\mathbb{R}$ by $\mathbb{F}$, rounding $\mathrm{fl} : \mathbb{R} \to \mathbb{F}$
Let $x \in \mathbb{R}$ then

$$\mathrm{fl}(x) = x(1+\delta), \quad |\delta| \leq \mathbf{u}.$$

Unit roundoff $\mathbf{u} = \epsilon/2$ for round-to-nearest

Standard model of floating point arithmetic

Let $x, y \in \mathbb{F}$,

$$\text{fl}(x \circ y) = (x \circ y)(1 + \delta), \quad |\delta| \leq \mathbf{u}, \quad \circ \in \{+, -, \cdot, /\}$$

IEEE 754 standard (1985)

Type	Size	Mantissa	Exponent	Unit roundoff	Range
Double	64 bits	52+1 bits	11 bits	$\mathbf{u} = 2^{-53} \approx 1,11 \times 10^{-16}$	$\approx 10^{\pm 308}$

For a more precise evaluation scheme

- Accurate evaluation of $p(x)$: the compensated Horner scheme and the compensated rule of thumb [1]
- An improved and validated error bound
- Theoretical and experimental results exhibit the
  - actual accuracy: twice the current working precision behavior,
  - actual speed: twice faster than the corresponding double-double implementation

[1] SG, N. Louvet, PhL. Compensated Horner Scheme. Submitted to SISC

More accuracy, how ?

More internal precision:

- hardware
  - extended precision in x86 architecture
- software
  - fixed length expansions libraries: double-double (Briggs, Bailey, Hida, Li), quad-double (Bailey, Hida, Li)
  - arbitrary length expansions libraries: Priest, Shewchuk
  - arbitrary multiprecision libraries: MP, MPFUN/ARPREC, MPFR

Correcting rounding errors:

- compensated summation (Kahan,1965) and doubly compensated summation (Priest,1991), etc.
- accurate sum and dot product: Ogita, Rump and Oishi (2005) $\longrightarrow$ twice the current working precision behavior and fast compared to double-double library

More accuracy, how ?

More internal precision:

- hardware
  - extended precision in x86 architecture
- software
  - fixed length expansions libraries: double-double (Briggs, Bailey, Hida, Li), quad-double (Bailey, Hida, Li)
  - arbitrary length expansions libraries: Priest, Shewchuk
  - arbitrary multiprecision libraries: MP, MPFUN/ARPREC, MPFR

Correcting rounding errors:

- compensated summation (Kahan,1965) and doubly compensated summation (Priest,1991), etc.
- accurate sum and dot product: Ogita, Rump and Oishi (2005) $\longrightarrow$ twice the current working precision behavior and fast compared to double-double library

At current working precision ...

Rule of thumb for backward stable algorithms :

solution accuracy ≈ condition number × computing precision

1. IEEE-754 precision: double ($\mathbf{u} = 2^{-53} \approx 10^{-16}$)
2. Condition number for the evaluation of $p(x) = \sum_{i=0}^{n} a_i x^i$:

$$\mathrm{cond}(p,x) = \frac{\sum_{i=0}^{n} |a_i||x|^i}{|\sum_{i=0}^{n} a_i x^i|} = \frac{\widetilde{p}(|x|)}{|p(x)|}, \text{ always } \geq 1.$$

3. Accuracy of the solution $\widehat{p}(x)$:

$$\boxed{\frac{|p(x) - \widehat{p}(x)|}{|p(x)|} \leq \alpha(n) \times \mathrm{cond}(p,x) \times \mathbf{u}}$$

with $\alpha(n) \approx 2n$

What means "twice the working precision behavior"?

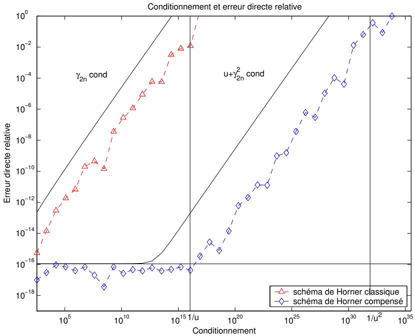
Compensated rule of thumb:

solution accuracy $\lesssim$ precision + condition number $\times$ precision^2

Three regimes in precision for the evaluation of $\widehat{p}(x)$:

1) condition number $\leq 1/\mathbf{u}$: the accuracy of $\widehat{p}(x)$ is optimal

$$\frac{|\widehat{p}(x) - p(x)|}{|p(x)|} \approx \mathbf{u}$$



What means "twice the working precision behavior"?

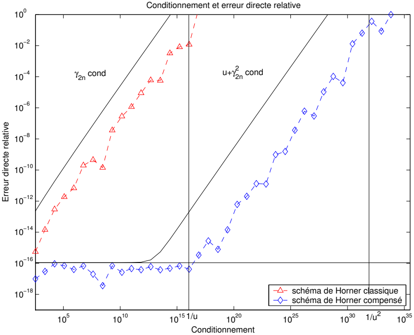
Compensated rule of thumb:

solution accuracy $\lesssim$ precision + condition number $\times$ precision^2

Three regimes in precision for the evaluation of $\widehat{p}(x)$:

2) $1/\mathbf{u} \leq$ condition number $\leq 1/\mathbf{u}^2$: the result $\widehat{p}(x)$ verifies

$$\frac{|\widehat{p}(x) - p(x)|}{|p(x)|} \approx \text{cond} \times \mathbf{u}^2$$



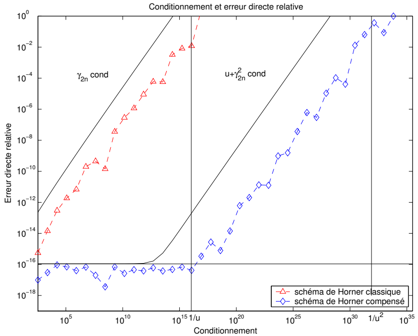
What means "twice the working precision behavior"?

Compensated rule of thumb:

solution accuracy $\lesssim$ precision + condition number $\times$ precision^2

Three regimes in precision for the evaluation of $\widehat{p}(x)$:

3) no more accuracy when condition number $> 1/\mathbf{u}^2$.



The Horner scheme

Algorithm 3 (Horner scheme)

function $\texttt{res} = \texttt{Horner}(p, x)$
  $s_n = a_n$
  for $i = n-1 : -1 : 0$
    $p_i = \mathrm{fl}(s_{i+1} \cdot x)$ % rounding error π_i
    $s_i = \mathrm{fl}(p_i + a_i)$ % rounding error σ_i
  end
  $\texttt{res} = s_0$

$\gamma_n = n\mathbf{u}/(1 - n\mathbf{u}) \approx n\mathbf{u}$

$$\frac{|p(x) - \texttt{Horner}(p,x)|}{|p(x)|} \leq \underbrace{\gamma_{2n}}_{\approx 2n\mathbf{u}} \mathrm{cond}(p, x)$$

Error-free transformations for sum and product

$$
\begin{array}{llll}
x = \mathrm{fl}(a \pm b) & \Rightarrow & a \pm b = x + y & \text{with } y \in \mathbb{F}, \\
x = \mathrm{fl}(a \cdot b) & \Rightarrow & a \cdot b = x + y & \text{with } y \in \mathbb{F},
\end{array}
$$

For the sum, algorithms by Dekker (1971) and Knuth (1974)

Algorithm 4 (Error-free transformation of the sum of 2 floating point numbers)

function $[x, y] = \mathtt{TwoSum}(a, b)$
  $x = \mathrm{fl}(a + b)$
  $z = \mathrm{fl}(x - a)$
  $y = \mathrm{fl}((a - (x - z)) + (b - z))$

Product : algorithm TwoProduct by Veltkamp and Dekker (1971)

Error-free transformation for the Horner scheme

$$p(x) = \texttt{Horner}(p, x) + (p_\pi + p_\sigma)(x)$$

Algorithm 5 (Error-free transformation for the Horner scheme)

function $[\texttt{Horner}(p, x), p_\pi, p_\sigma] = \texttt{EFTHorner}(p, x)$
$s_n = a_n$
for $i = n-1 : -1 : 0$
  $[p_i, \pi_i] = \texttt{TwoProduct}(s_{i+1}, x)$
  $[s_i, \sigma_i] = \texttt{TwoSum}(p_i, a_i)$
  Let π_i be the coefficient of degree i of p_π
  Let σ_i be the coefficient of degree i of p_σ
end
$\texttt{Horner}(p, x) = s_0$

Compensated Horner scheme

Algorithm 6 (Compensated Horner scheme)

function $\texttt{res} = \texttt{CompHorner}(p, x)$
$[h, p_\pi, p_\sigma] = \texttt{EFTHorner}(p, x)$
$c = \texttt{Horner}(p_\pi + p_\sigma, x)$
$\texttt{res} = \mathrm{fl}(h + c)$

Accuracy of the compensated Horner scheme

Theorem 3

Let p be a polynomial of degree n with floating point coefficients, and x be a floating point value. Then if no underflow occurs,

$$\frac{|\mathtt{CompHorner}(p,x) - p(x)|}{|p(x)|} \leq \mathbf{u} + \underbrace{\gamma_{2n}^2}_{\approx 4n^2\mathbf{u}^2} \operatorname{cond}(p,x).$$

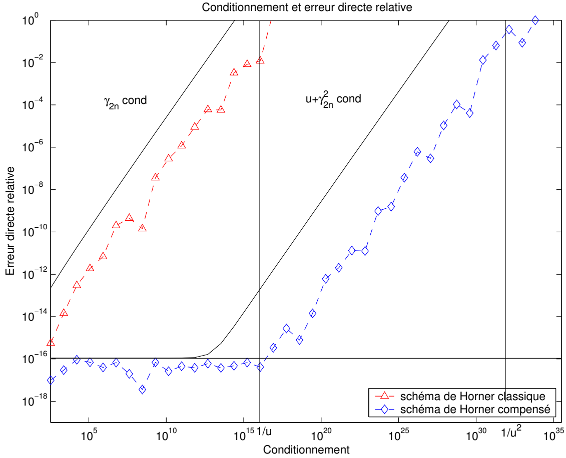
- Key point in the proof:

$$(\widetilde{p_\pi} + \widetilde{p_\sigma})(|x|) \leq \gamma_{2n}\widetilde{p}(|x|)$$

- a similar bound is proved in presence of underflow

Numerical experiments: testing the accuracy

Evaluation of $p_n(x) = (x-1)^n$ for $x = \mathrm{fl}(1.333)$ and $n = 3, \ldots, 42$



Numerical experiments: testing the speed efficiency

We compare

- Horner: IEEE 754 double precision Horner scheme
- CompHorner: our Compensated Horner scheme
- DDHorner: Horner scheme with internal double-double computation

All computations are performed in C language and IEEE 754 double precision

Pentium 4: 3.0GHz, 1024kB cache L2 - GCC 3.4.1				
ratio	minimum	mean	maximum	theoretical
CompHorner/Horner	1.5	2.9	3.2	13
DDHorner/Horner	2.3	8.4	9.4	17

$\longrightarrow$ compensated Horner scheme = Horner scheme with double-double without renormalization

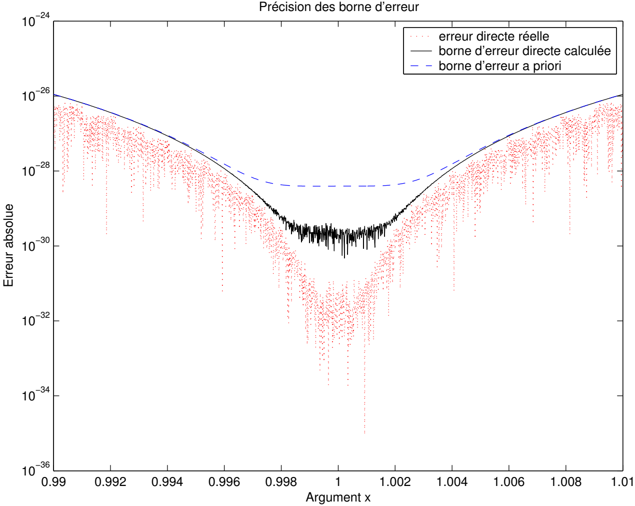
A dynamic error bound

Theorem 4

Given a polynomial p of degree n with floating point coefficients, and a floating point value x, we consider $\texttt{res} = \texttt{CompHorner}(p, x)$. The absolute forward error affecting the evaluation is bounded according to

$$|\texttt{CompHorner}(p, x) - p(x)| \leq \mathrm{fl}((\mathbf{u}|\texttt{res}| + (\gamma_{4n+2}\texttt{Horner}(\widetilde{p_\pi} + \widetilde{p_\sigma}, |x|) + 2\mathbf{u}^2|\texttt{res}|))).$$

Accuracy of the bound for $p_5(x) = (x-1)^5$



Outline

Real perturbations (1/2)

Motivations :

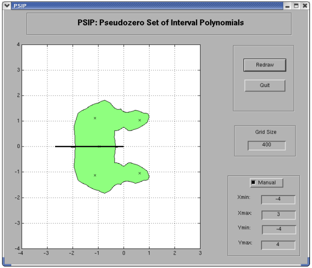
- rounding errors are always real
- uncertain data in engineering are often real

Results :

- Real condition number and backward error for polynomial evaluation and zeros
  $\longrightarrow$ explicit formulas for those condition numbers and backward errors
  $\longrightarrow$ the ratio between the real condition number and the classical condition number lies in the interval $[1, \sqrt{2}]$
  $\longrightarrow$ The real backward error can be larger that the classical backward error

Real perturbations (2/2)

- Zeros of interval polynomials[2]
  $\longrightarrow$ Matlab tool for drawing zeros of interval polynomials



- Real pseudozero set for multivariate polynomials[3]
  $\longrightarrow$ an explicit formula for computing this set

[2] SG & PhL. Pseudozero set of interval polynomials. ACM SAC 2006
[3] SG. Pseudozero set of multivariate polynomials. Poster CASC 2005

Pseudospectra and structured condition numbers (1/2)

Motivations :

- structured error analysis
- classical structures Toeplitz, Hankel, circulant, symmetric, ...
- structures deriving from Lie and Jordan algebras

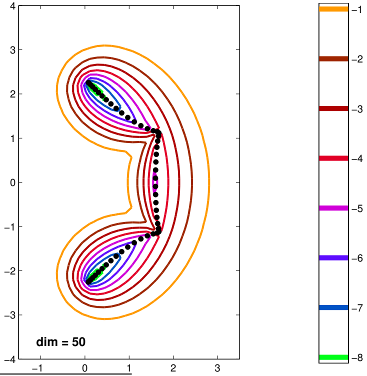
Results :

- Structured condition numbers for matrix problems[4]
  → structured error analysis with Lie and Jordan algebras
  → little or no differences between structured and unstructured condition numbers for these structures, similar results for the backward error

[4] F. Tissseur & SG. Structured Condition Numbers and Backward Errors in Scalar Product Spaces, Research Report

Pseudospectra and structured condition numbers (2/2)

- Pseudospectra and structures[5]
  $\longrightarrow$ for Toeplitz, Hankel, circulant structures, the pseudospectra equals the structured pseudospectra



[5] SG. A note on structured pseudospectra. J. Comput. Appl. Math., 2006.

Outline

Summary and future work on improving the accuracy

Summary

- A compensated Horner scheme: accurate polynomial evaluation
- Fast and accurate computation of geometric predicates

Future work

- double-double and XBLAS without renormalization
- Increasing the accuracy of algorithms with Newton's methods and iterative refinement

Summary and future work on the use of pseudozero set

Summary

- Applications of pseudozero set to test the approximate coprimeness of polynomials[6]
- Applications of pseudozero set to compute stability radius and pseudozero abscissa[7]

Future work

- Certify the drawing of pseudozero set using interval arithmetic (for example the Sivia algorithm by Jaulin and Walter)

---

[6] SG & PhL. Testing polynomial primality with pseudozeros. RNC'5

[7] SG. Computation of pseudozero abscissa. SYNASC 2004

Summary and future work on real perturbations

Summary

- Real condition number and real backward error for polynomial evaluation and zeros
- MATLAB tool for drawing pseudozeros of interval polynomials
- Generalization of real pseudozero set to multivariate polynomials

Future work

- Generalization to real pseudospectra
- Real condition number for generalized eigenvalue problems

Summary and future work on structured linear algebra

Summary

- Structured pseudospectra for Toeplitz, Hankel, circulant, symmetric, skew-symmetric structures
- Structured error analysis for structures deriving from Lie and Jordan algebra for linear systems, distance to singularity and inversion

Future work

- Structured error analysis for least square problems and Penrose-Moore inversion
- Same thing with Drazin inversion (singular linear systems)

Thank you for your attention