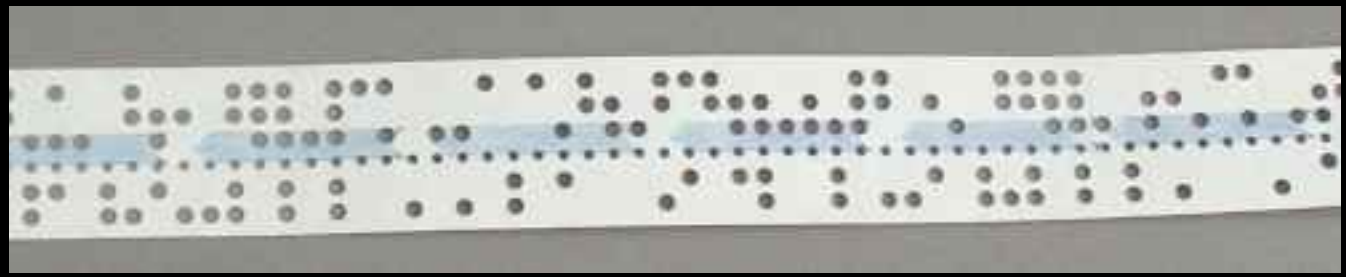
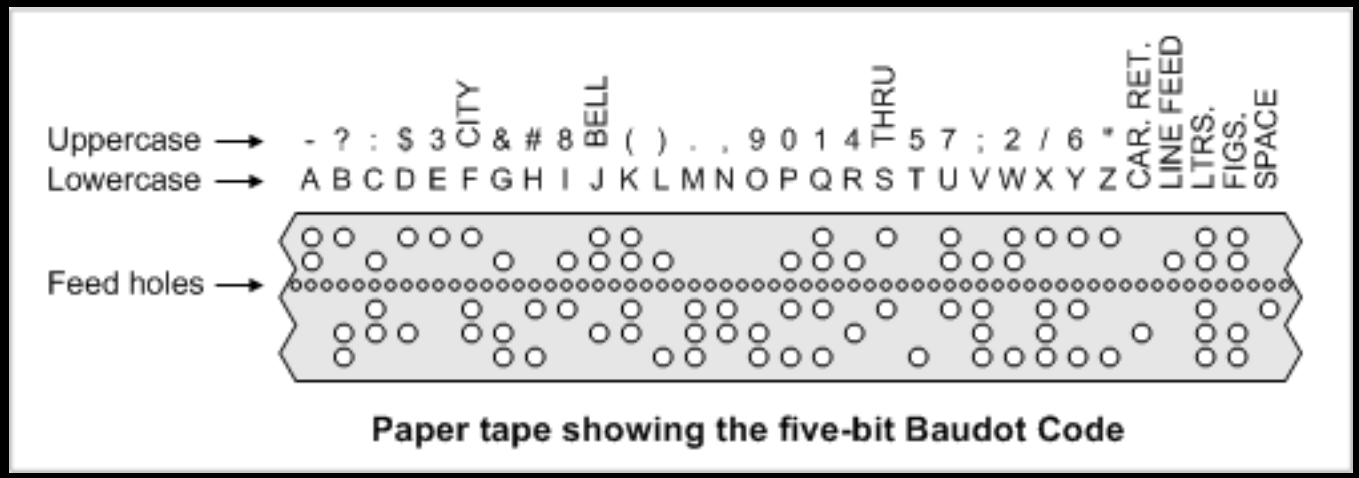
An Introduction to the Quality of Computed Solutions

Sven Hammarling
NAG Ltd, Oxford
sven@nag.co.uk

Eliot 803

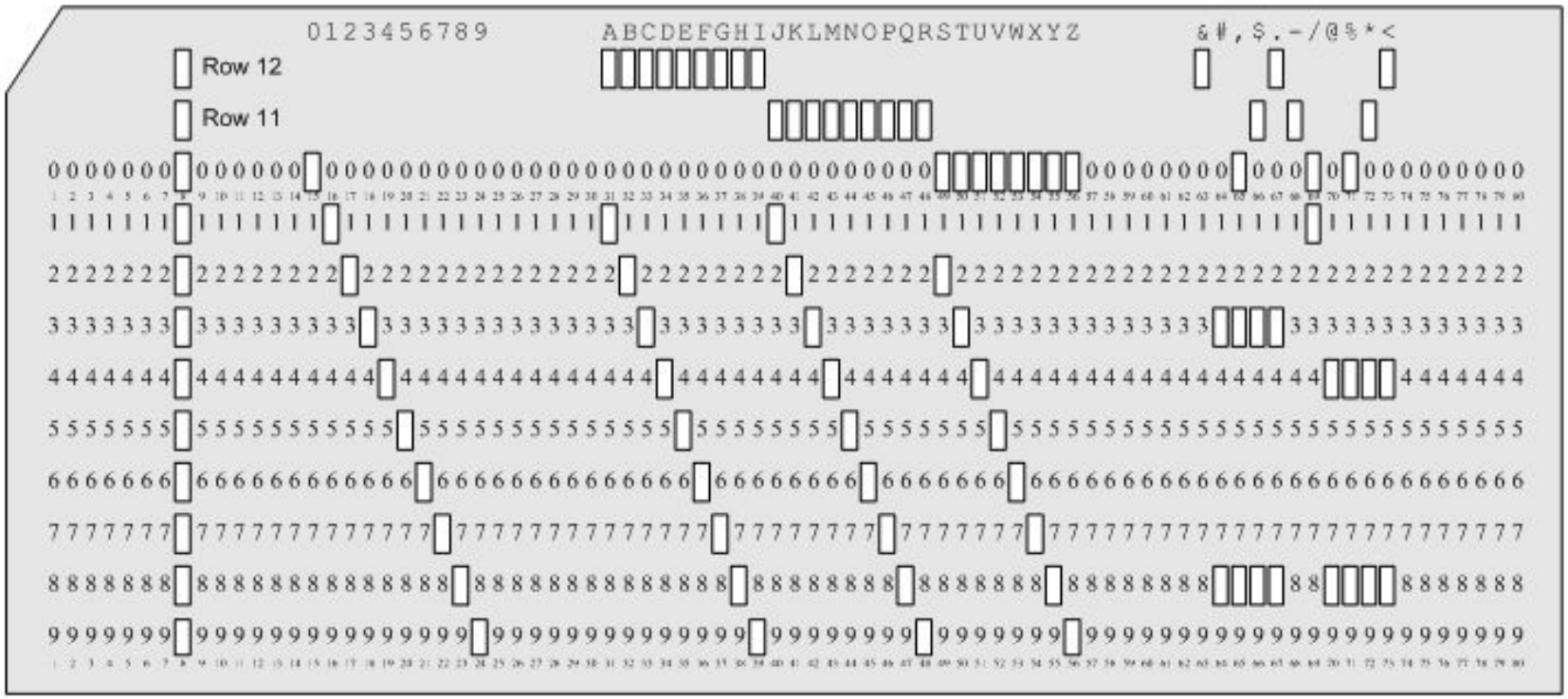
Paper Tape





Paper tape showing the five-bit Baudot Code

Punched Card



IBM 80-Column punched card format

Eliot 803 Backing Tape

IFIP WG5 Book

This lecture is based upon Chapter 4 of:

"Accuracy and Reliability in Scientific Computing" edited by Bo Einarsson under the auspices of IFIP WG 2.5, Project 68, published by SIAM, 2005 (eprints.ma.man.ac.uk/101/)

N. J. Higham. *Accuracy and Stability of Numerical Algorithms*, 2nd edition, SIAM, 2002

Motivation

- What is the quality of a computed solution and why we should worry about the quality?
- Solutions returned by algorithms implemented on computers are not necessarily reliable
- Ideas such as condition, stability and error analysis, help us understand how to measure the quality of a computed solution
- We need to provide sensible models and data
- Important for both users and developers

Example - Sample Variance

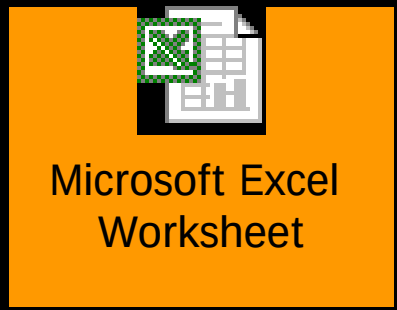
$$s_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2 \qquad (1)$$

$$s_n^2 = \frac{1}{n-1}\left(\sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2\right) \qquad (2)$$

For $x^T = (10000, 10001, 10002)$ using 8 figure arithmetic

(1) gives: 1.0, (2) gives: 0.0

Excel Example: Standard Deviation



German High Speed Trains (ICE)

Quality of Computed Solutions

The quality of computed solutions is concerned with assessing how good a computed solution is in some appropriate measure

Quality software should implement reliable algorithms and should, where appropriate, provide measures of solution quality

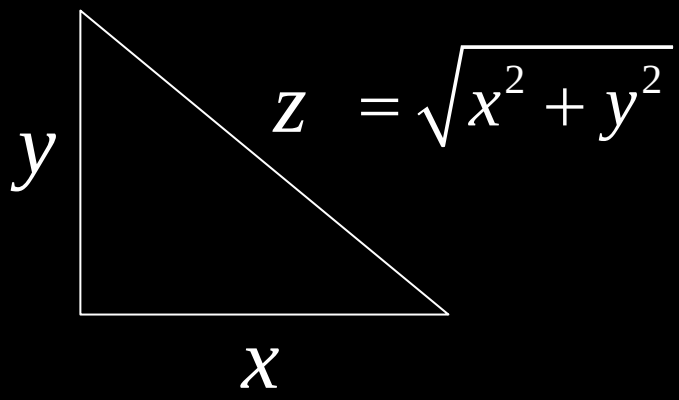
Example - Means

Mean of a set of numbers can be outside their range. Using 3 figure arithmetic:

$$(5.01 + 5.03)/2 = 10.0/2 = 5.00$$

Does this matter? If so, can we guarantee an answer within range?

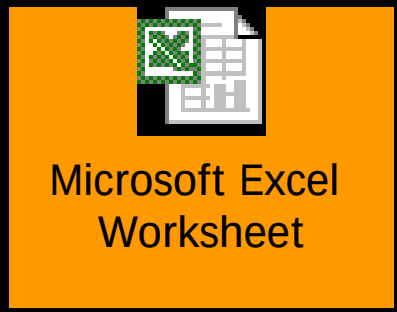
Overflow/Underflow Example:
Hypotenuse of a right angled triangle



$$a = \max(x, y), b = \min(x, y)$$

$$z = \begin{cases} a\sqrt{1+\left(\frac{b}{a}\right)^2}, & a > 0 \\ 0, & a = 0 \end{cases}$$

Excel Example:
Standard Deviation - Overflow



Condition

The *condition* of a problem is concerned with the sensitivity of the problem to perturbations in the data

A problem is ill-conditioned if small changes in the data cause relatively large changes in the solution. Otherwise a problem is well-conditioned

Condition Number for Function of One Variable

Let f be twice differentiable, $y = f(x)$ and $\hat{y} = f(x+\varepsilon)$. Then

$$\hat{y} - y = f(x+\varepsilon) - f(x)$$

$$= f'(x)\varepsilon + \frac{f''(x+\theta\varepsilon)}{2!}\varepsilon^2, \theta \in (0,1)$$

giving

$$\frac{\hat{y}-y}{y} = \left(\frac{xf'(x)}{f(x)}\right)\frac{\varepsilon}{x} + O\left(\varepsilon^2\right)$$

Condition Number for Function of One Variable (cont'd)

The quantity

$$\kappa(x) = \left| \frac{x f'(x)}{f(x)} \right|$$

is called the *condition number* of f since

$$\left| \frac{\hat{y} - y}{y} \right| \approx \kappa(x) \left| \frac{\varepsilon}{x} \right|$$

Example

$$y = f(x) = \cos x$$

Then

$$\kappa(x) = |x \tan x|$$

Thus $\cos x$ is most sensitive close to asymptotes of $\tan x$. For example with $x = 1.57$ and $\varepsilon = -0.01$

$$\kappa(x)\left|\frac{\varepsilon}{x}\right| \approx 12.5577.$$

Actually, $|(\hat{y} - y)/y| = 12.55739\ldots$

Condition Number for Linear Equations

$$Ax = b, \quad (A+E)\tilde{x} = b$$

$$A(x-\tilde{x}) = E\tilde{x}, \quad \text{so that} \quad x-\tilde{x} = A^{-1}E\tilde{x}$$

giving

$$\frac{\|x-\tilde{x}\|}{\|\tilde{x}\|} \le \|A^{-1}\| \cdot \|E\| = \kappa(A)\frac{\|E\|}{\|A\|},$$

where $\kappa(A) = \|A\| \cdot \|A^{-1}\|$ is the condition number of A with respect to solving linear equations

Condition Examples (cont'd)

$$99x_1 + 98x_2 = 197$$

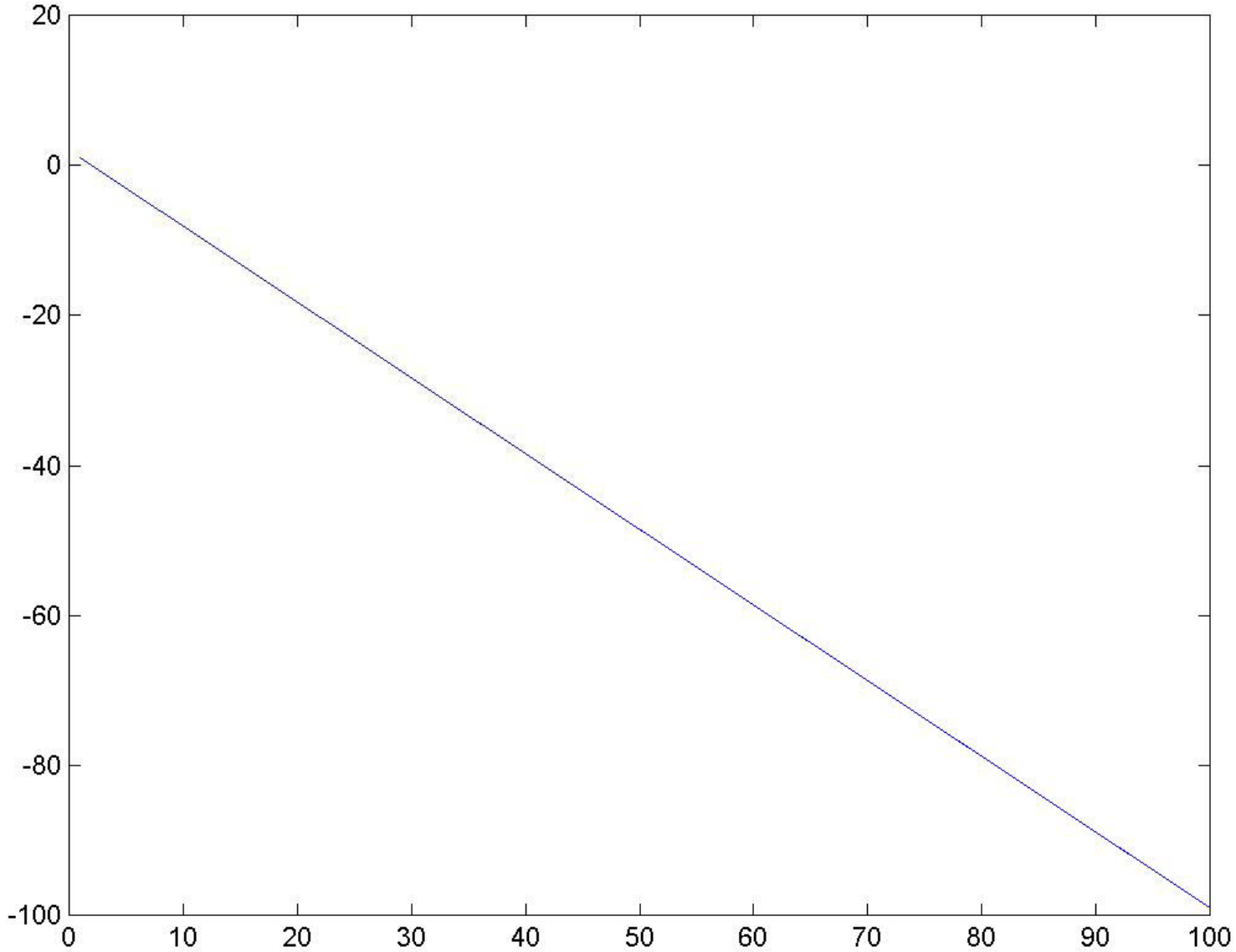
$$100x_1 + 99x_2 = 199$$

$$x_1 = x_2 = 1$$

$$98.99x_1 + 98x_2 = 197$$

$$100x_1 + 99x_2 = 199$$

$$x_1 = 100, \quad x_2 = -99$$


20
0
-20
-40
-60
-80
-100
0
10
20
30
40
50
60
70
80
90
100

Condition Number Example

$$A = \begin{pmatrix} 99 & 98 \\ 100 & 99 \end{pmatrix}, \quad \|A\|_1 = 199$$

$$A^{-1} = \begin{pmatrix} 99 & -98 \\ -100 & 99 \end{pmatrix}, \quad \|A^{-1}\|_1 = 199$$

$$\kappa_1(A) = 199^2 \approx 4 \times 10^4$$

Stability

The *stability* of a method for solving a problem is concerned with the sensitivity of the method to (rounding) errors in the solution process

A method that guarantees as accurate a solution as the data warrants is said to be stable, otherwise the method is unstable

Quadratic Equation

$$q(x) = 1.6x^2 - 100.1x + 1.251 = 0$$

Four significant figure arithmetic on the standard formula

$$x = (-b \pm \sqrt{b^2 - 4ac}) \big/ (2a)$$

gives

$$x_1 = 62.53, \quad x_2 = 0.03125,$$

but using

$$x_2 = c/(ax_1)$$

gives

$$x_2 = 0.01251$$

Exact solution is

$$x_1 = 62.55, \quad x_2 = 0.0125$$

Recurrence Relation

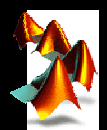
$$y_n = (1/e)\int_0^1 x^n e^x dx$$

Easy to show that

$$0 \le y_n \le 1/(n+1)$$

Integrating by parts gives

$$y_n = 1 - n y_{n-1},\ y_0 = 1 - 1/e \approx 0.632121$$



recursive.m

Recurrence Relation (Cont'd)

$$\tilde{y}_0 = y_0 + \varepsilon$$

$$\tilde{y}_1 = 1 - \tilde{y}_0 = y_1 - \varepsilon$$

$$\tilde{y}_2 = 1 - 2\tilde{y}_1 = y_2 + 2\varepsilon$$

$$\tilde{y}_3 = 1 - 3\tilde{y}_2 = y_3 - 6\varepsilon$$

In general

$$\tilde{y}_n = y_n + (-1)^n n!\varepsilon$$

Error Analysis

Error analysis is concerned with analysing the cumulative effects of errors

Usually these errors will be rounding or truncation errors

Definitions

Error analysis is concerned with establishing whether or not an algorithm is stable for the problem in hand

A *forward error analysis* is concerned with how close the computed solution is to the exact solution

A *backward error analysis* is concerned with how well the computed solution satisfies the problem to be solved

Example

$$Ax = b, \quad r = b - A\tilde{x}$$

$$A = \begin{pmatrix} 99 & 98 \\ 100 & 99 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, x = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Consider $\tilde{x} = \begin{pmatrix} 2.97 \\ -2.99 \end{pmatrix}$

Example

$$Ax = b, \quad r = b - A\tilde{x}$$

$$A = \begin{pmatrix} 99 & 98 \\ 100 & 99 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, x = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Consider $\tilde{x} = \begin{pmatrix} 2.97 \\ -2.99 \end{pmatrix}$

$$\tilde{x} - x = \begin{pmatrix} 1.97 \\ -1.99 \end{pmatrix}$$

Example

$$Ax = b, \quad r = b - A\tilde{x}$$

$$A = \begin{pmatrix} 99 & 98 \\ 100 & 99 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, x = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Consider $\tilde{x} = \begin{pmatrix} 2.97 \\ -2.99 \end{pmatrix}$

$$\tilde{x} - x = \begin{pmatrix} 1.97 \\ -1.99 \end{pmatrix}, \quad \text{but } r = \begin{pmatrix} -0.01 \\ 0.01 \end{pmatrix}$$

Example

$$Ax = b, \quad r = b - A\tilde{x}$$

$$A = \begin{pmatrix} 99 & 98 \\ 100 & 99 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, x = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Consider $\tilde{x} = \begin{pmatrix} 2.97 \\ -2.99 \end{pmatrix}$

$$\tilde{x} - x = \begin{pmatrix} 1.97 \\ -1.99 \end{pmatrix}, \quad \text{but } r = \begin{pmatrix} -0.01 \\ 0.01 \end{pmatrix}$$

Now consider $\tilde{x} = \begin{pmatrix} 1.01 \\ -0.99 \end{pmatrix}$

Example

$$Ax = b, \quad r = b - A\tilde{x}$$

$$A = \begin{pmatrix} 99 & 98 \\ 100 & 99 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, x = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Consider $\tilde{x} = \begin{pmatrix} 2.97 \\ -2.99 \end{pmatrix}$

$$\tilde{x} - x = \begin{pmatrix} 1.97 \\ -1.99 \end{pmatrix}, \quad \text{but } r = \begin{pmatrix} -0.01 \\ 0.01 \end{pmatrix}$$

Now consider $\tilde{x} = \begin{pmatrix} 1.01 \\ -0.99 \end{pmatrix}$,

$$\tilde{x} - x = \begin{pmatrix} 0.01 \\ 0.01 \end{pmatrix}$$

Example

$$Ax = b, \quad r = b - A\tilde{x}$$

$$A = \begin{pmatrix} 99 & 98 \\ 100 & 99 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, x = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Consider $\tilde{x} = \begin{pmatrix} 2.97 \\ -2.99 \end{pmatrix}$

$$\tilde{x} - x = \begin{pmatrix} 1.97 \\ -1.99 \end{pmatrix}, \quad \text{but } r = \begin{pmatrix} -0.01 \\ 0.01 \end{pmatrix}$$

Now consider $\tilde{x} = \begin{pmatrix} 1.01 \\ -0.99 \end{pmatrix}$,

$$\tilde{x} - x = \begin{pmatrix} 0.01 \\ 0.01 \end{pmatrix}, \quad \text{but } r = \begin{pmatrix} -1.97 \\ -1.97 \end{pmatrix}$$

Quadratic Equation

$$q(x) = 1.6x^2 - 100.1x + 1.251$$

Rounding to four significant figures,

$$1.6\,(x - 62.53)(x - 0.03125) = 1.6x^2 - 100.1x + 3.127,$$

so neither forward, nor backward stable. But

$$1.6\,(x - 62.53)(x - 0.01251) = 1.6x^2 - 100.1x + 1.252,$$

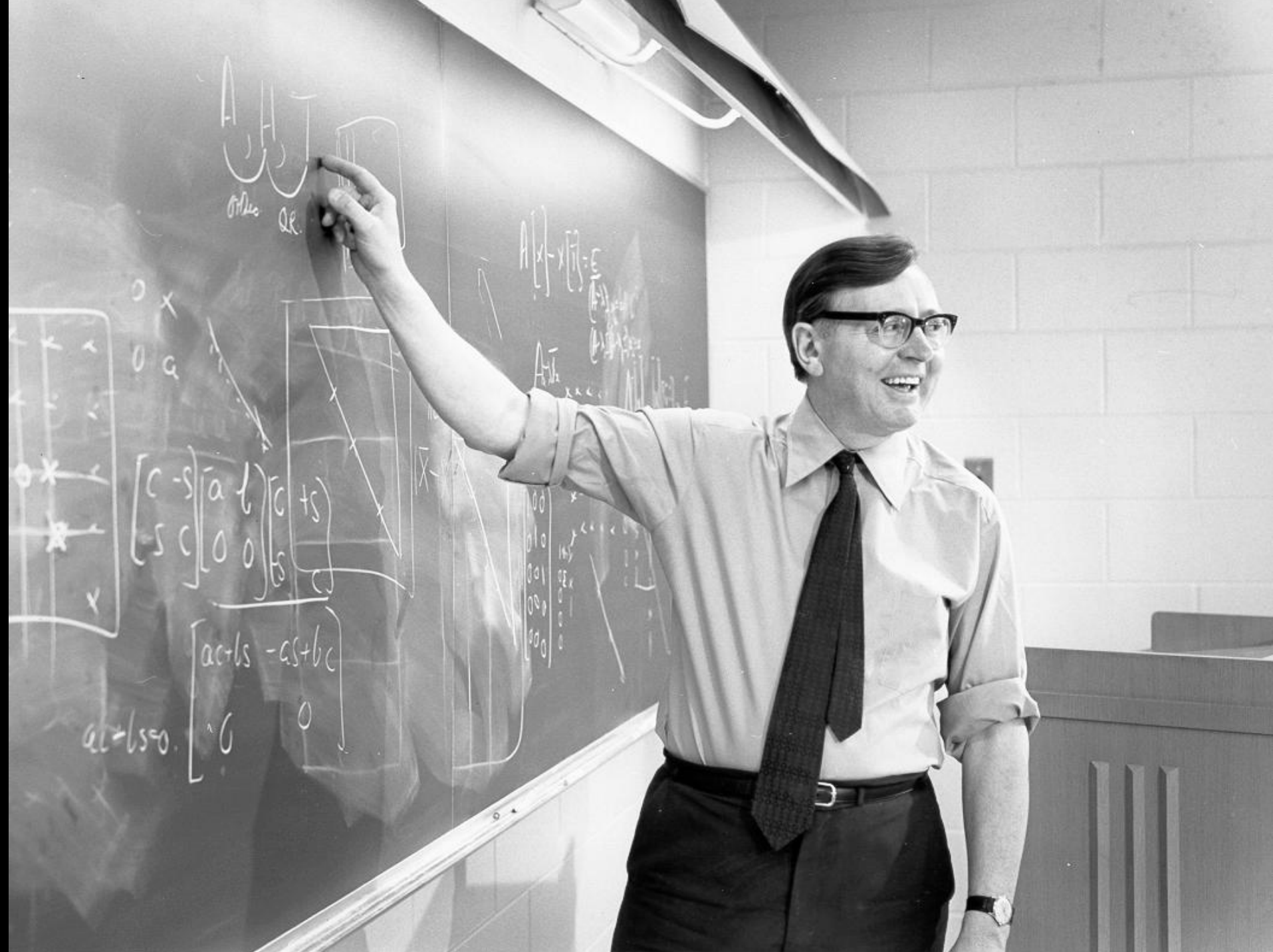
so both forward and backward stable

The Purpose of Error Analysis

"The clear identification of the factors determining the stability of an algorithm soon led to the development of better algorithms. …

"For me, then, the primary purpose of the rounding error analysis was insight."

Wilkinson, 1986
Bulletin of the IMA, Vol 22, p197



A Posteriori Bound

$$r = A\tilde{x} - b, \quad E = \left(r\tilde{x}^T\right) \Big/ \left(\tilde{x}^T\tilde{x}\right)$$

Then

$$\left(A + E\right)\tilde{x} = b.$$

and this is an E that minimizes $\|E\|_F$ (and $\|E\|_2$)

If $\varepsilon = \dfrac{\|r\|_F}{\|A\|_F \, \|\tilde{x}\|_F}$ then $\|E\|_F = \varepsilon \|A\|_F$

Backward Error and Perturbation Analysis

$$Ax = b, \quad (A+E)\tilde{x} = b$$

If we know how perturbations in A affect the solution x, then we can estimate the accuracy of the computed solution $\tilde{x}$. That is, an estimate of the backward error allows us to estimate the forward error.

$$\frac{\|x-\tilde{x}\|}{\|\tilde{x}\|} \leq \kappa(A)\frac{\|E\|}{\|A\|}$$

Condition and Error Analysis

Approximately:

forward error $\leq$

condition number $\times$ backward error

$$\left(\left| \frac{\hat{y} - y}{y} \right| \approx \kappa(x) \left| \frac{\Delta x}{x} \right|, \quad \frac{\|x - \tilde{x}\|}{\|\tilde{x}\|} \leq \kappa(A) \frac{\|E\|}{\|A\|} \right)$$

Floating Point Numbers - Representation

$$\boxed{x = \pm m \times b^{e-t}, \quad 0 \le m \le b^t - 1}$$

b – base (or radix)

t – precision

e – exponent, with exponent range $[e_{\min}, e_{\max}]$

m – mantissa (or significand)

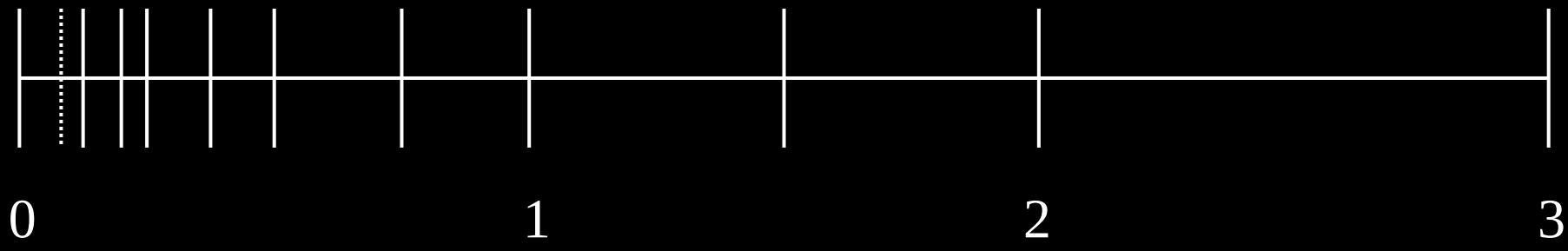
If $x \neq 0$ and $m \geq b^{t-1}$ then x is *normalized*, otherwise $x \neq 0$ is *denormalized*. Denormalized numbers between 0 and the smallest normalized number are called *subnormal*.

$$u = \tfrac{1}{2} \times b^{1-t}$$

is called the *relative machine precision.*

Floating Point Numbers - Example (cont'd)

$$b = 2, \quad t = 2, \quad e_{\min} = -2, \quad e_{\max} = 2$$



The *relative machine precision* is

$$u = \tfrac{1}{2} \times b^{1-t} \quad \left(= \tfrac{1}{4}\right)$$

See LAPACK routine `_LAMCH`, or the Matlab built in `eps` ($2u$)

IEEE Arithmetic Formats

Format	Pr ecision	Exponent	Approx Range	Approx precision
Single	24 bits	8 bits	$10^{\pm 38}$	10^{-8}
Double	53 bits	11 bits	$10^{\pm 308}$	10^{-16}
Extended	≥ 64	≥ 15	$10^{\pm 4932}$	10^{-20}

Floating Point Error Analysis

Floating point error analysis is concerned with the analysis of errors in the presence of floating point arithmetic

It is concerned with relative errors

Floating Point Error Analysis

Let x be a real number. Then we use the notation:

$$fl(x) = \text{ floating point value of } x$$

Fundamental assumption:

$$\boxed{fl(x) = x(1+\varepsilon), |\varepsilon| \le u}$$

where u is the unit rounding error, or relative machine precision. Of course

$$\frac{fl(x) - x}{x} = \varepsilon$$

Floating Point Error Analysis

If x and y are floating point numbers then

$$\boxed{fl(x \otimes y) = (x \otimes y)(1+\varepsilon), \quad |\varepsilon| \leq u}$$

where

$$\otimes \equiv +, -, \times, \div$$

Cancellation (Summation) Example

$$x = 1.000 + 1.000 \times 10^4 - 1.000 \times 10^4 = 1$$

In four figure arithmetic

$$\tilde{x} = fl\left(fl\left(1.000 + 1.000 \times 10^4\right) - 1.000 \times 10^4 \right)$$

$$= fl\left(1.000 \times 10^4 - 1.000 \times 10^4\right)$$

$$= 0$$

Note that

$$\tilde{x} = 1.000 + 1.000 \times 10^4 - 1.0001 \times 10^4$$

Difference of Two Squares

$$z = x^2 - y^2 \equiv (x+y)(x-y)$$

$$\tilde{z} = fl\left(x^2 - y^2\right) = x^2(1+\varepsilon_1) - y^2(1+\varepsilon_2)$$

$$= z + \left(x^2\varepsilon_1 - y^2\varepsilon_2\right),\ \varepsilon_1, \varepsilon_2 \le 2u$$

$$\hat{z} = fl\left((x+y)(x-y)\right) = (x+y)(x-y)(1+\varepsilon)$$

$$= z(1+\varepsilon),\ \varepsilon \le 3u$$

$x = 543.2,\ y = 543.1,$ so that $z = 108.63$

In 4 figure arithmetic

$\tilde{z} = 100,$ but $\hat{z} = 108.6$

Other Problems

Similar results to $Ax = b$ can be obtained for many other problems of linear algebra.

For example

$$Ax = \lambda x, \quad (A + E)\tilde{x} = \tilde{\lambda}\tilde{x},$$

with $\|E\|$ small relative to $\|A\|$ for stable methods for solving the eigenvalue problem

LAPACK

Users'

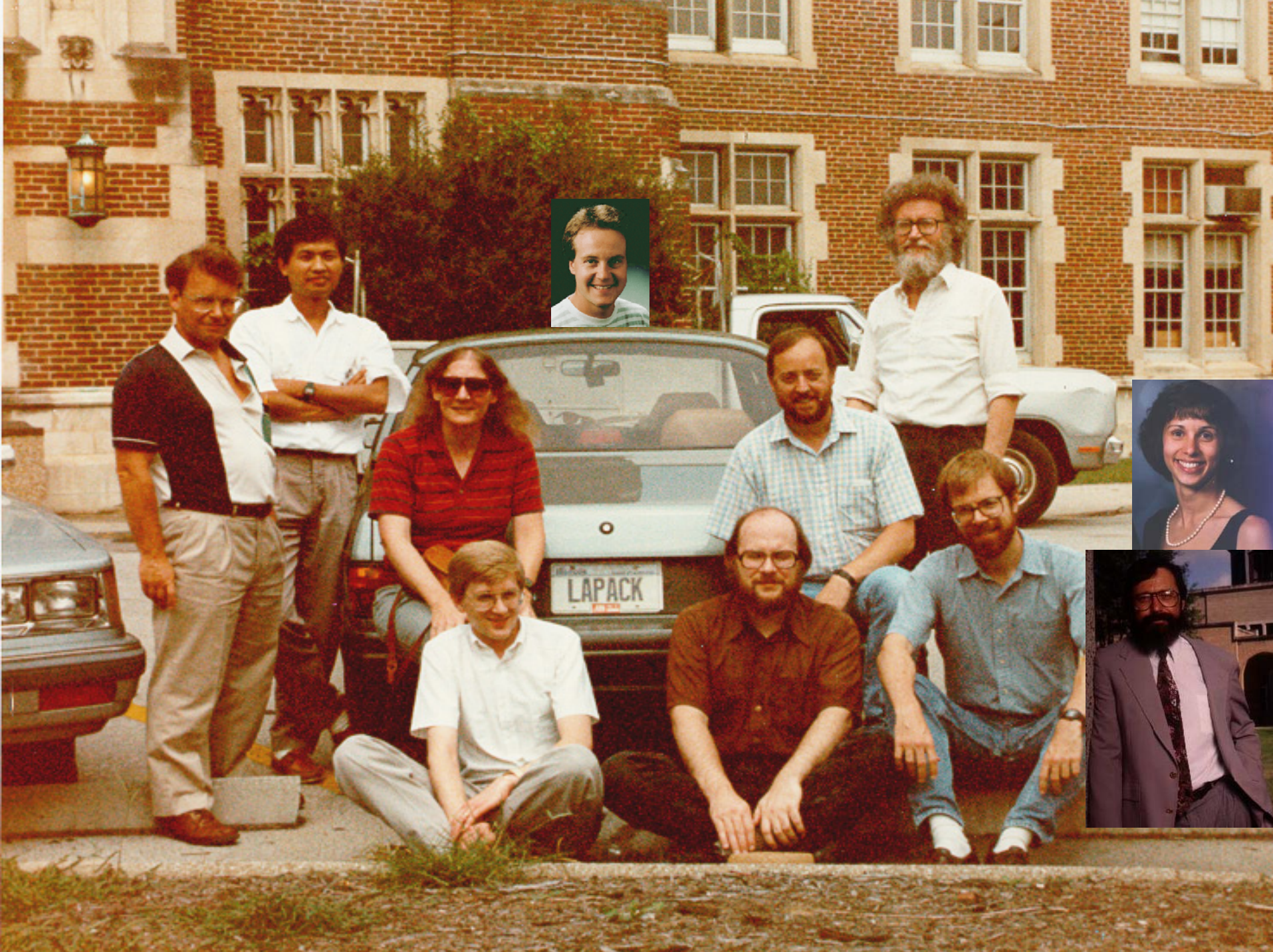
Guide

E. Anderson, Z. Bai, C. Bischof
S. Blackford, J. Demmel, J. Dongarra
J. Du Croz, A. Greenbaum, S. Hammarling
A. McKenney, D. Sorensen

SIAM, 1999 (3rd Edition)

Proceeds to SIAM student travel fund

http://www.netlib.org/lapack/lug/lapack_lug.html


LAPACK

LAPACK and Error Bounds

"In Addition to providing faster routines than previously available, LAPACK provides more comprehensive and better error bounds. Our goal is to provide error bounds for most quantities computed by LAPACK."

LAPACK Users' Guide, Chapter 4 – Accuracy and Stability

Error Bounds in LAPACK

The Users' Guide gives details of error bounds and code fragments to compute those bounds. In many cases the routines return the bounds directly

Error Bounds in LAPACK: Example

DGESVX is an 'expert' driver for solving $AX = B$

`DGESVX(...,RCOND,FERR,BERR,WORK,...,INFO)`

`RCOND` : Estimate of $1/\kappa(A)$

`FERR(j)`: Estimated forward error for X_j

`BERR(j)`: Componentwise relative backward error for X_j

(smallest relative change in any element of A and B

that makes X_j an exact solution)

`WORK(1)`: Reciprocal of pivot growth factor $(1/g)$

`INFO` : > 0 if A is singular or nearly singular

ODE Software Example

NAG routines D02PCF and D02PDF integrate

$$y' = f(t, y) \text{ given } y(t_0) = y_0,$$

where y is the n element solution vector and t is the independent variable, using a Runge-Kutta method. (These routines are based upon RKSUITE)

ODE Software Example (contd)

NAG routine D02PZF returns global error estimates for the Runge-Kutta solvers D02PCF and D02PDF:

`D02PZF( RMSERR, ERRMAX, TERRMX,…)`

`RMSERR:` Approximate Root mean square error

`ERRMAX:` Max approximate true error

`TERRMX:` Point at which max approximate true error occurred

LAPACK and Scaling

- Many routines perform scaling to move data away from overflow and underflow thresholds
  - E.g. DGEEVX, DGELSS
- A number of routines offer options for equilibration and balancing
  - E.g. DGESVX, DGEEVX

Science → Mathematical Model

- Mathematical model should be as realistic as possible
  - Millennium bridge
  - North Sea oil rig?
- If possible, model as a well-conditioned, well-posed problem

$$y_n = (1/e)\int_0^1 x^n e^x dx \neq$$

$$y_n = 1 - n y_{n-1},\ y_0 = 1 - 1/e$$

Modelling or Software? London Millennium Bridge, 2000

Tacoma Bridge

What to do if Software does not Provide Error Estimates?

- Run the problem with perturbed data
- Better still, use a software tool such as PRECISE which allows you to perform a stochastic analysis http://www.cerfacs.fr/algor/Softs/PRECISE/index.html
- Interval analysis
  - Interval analysis can provide automatic error bounds for a number of problems
  - There is an interval arithmetic toolbox for MATLAB, INTLAB, and the Sun compiler supports interval arithmetic
- Put pressure on developers to provide estimates

Quote

“You have been solving these damn problems better than I can pose them.”

Sir Edward Bullard, Director NPL, in a remark to Wilkinson. (mid 1950s.)

(Provide solutions that are at least as good as the data deserves)

Web Sites

- Disasters attributable to bad numerical computing: http://www.math.psu.edu/dna/disasters/
- Numerical problems: RISKS-LIST: http://catless.ncl.ac.uk/Risks/
- London millennium bridge: http://www.arup.com/MillenniumBridge/
- Stories for Computation: Why Care Is Needed: http://www.cs.clemson.edu/~steve/stories/stories.html
- Software Bugs - Software Glitches: http://wwwzenger.informatik.tu-muenchen.de/persons/huckle/bugse.html