# Numerical Accuracy and Floating-Point Maths

David Grellscheid
based on slides by Axel Kohlmeyer, Temple U.

## Before computations:

Modelling: neglecting certain properties
Empirical data: not every input is known perfectly

Previous computations: data may be taken from other (error-prone) numerical methods

Sloppy programming (e.g. inconsistent conversions)

## During computations:

Truncation: a numerical method approximates a continuous solution

Rounding: computers offer only finite precision in representing real numbers

Computing the surface of the earth using

$$A = 4\pi r^2$$

This involves several approximations:

Modelling: the earth is not exactly a sphere

Measurement: earth's radius is an empirical number

Truncation: the value of $\pi$ is truncated

Rounding: all numbers used are rounded due to arithmetic operations in the computer

Total error is the sum of all errors; usually one will dominate

The Abdus Salam International Centre for Theoretical Physics

Computing the surface of the earth using

$$A = 4\pi r^2$$

This involves several approximations:

Modelling: the earth is not exactly a sphere

Measurement: earth's radius is an empirical number

Truncation: the value of $\pi$ is truncated

Rounding: all numbers used are rounded due to arithmetic operations in the computer

Total error is the sum of all errors; usually one will dominate

Real numbers have unlimited accuracy
On a computer, need to represent them in finite width

One option: fixed point numbers

16-bit fixed: range ±32768, step size 1

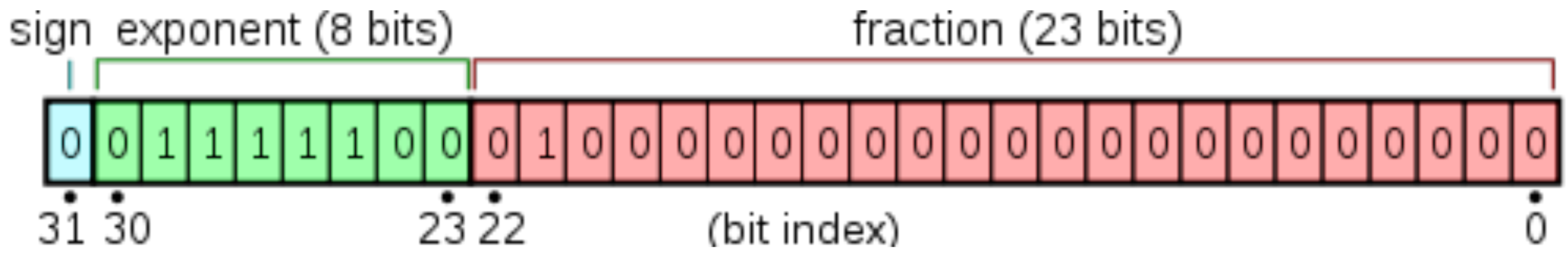
12/4-bit fixed: range ±2048, step size 0.0625

ICTP The Abdus Salam International Centre for Theoretical Physics

Need wider range in the same number of bits,
keeping reasonable precision.
Relative precision often sufficient

$[\pm][1.\text{fraction}]\ 2^{[\text{exponent}]}$

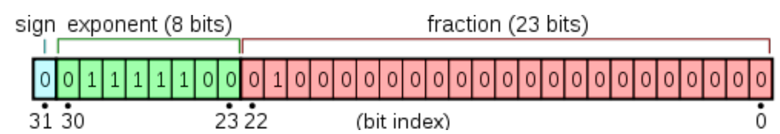
Single-precision floating point (float, 4 bytes)



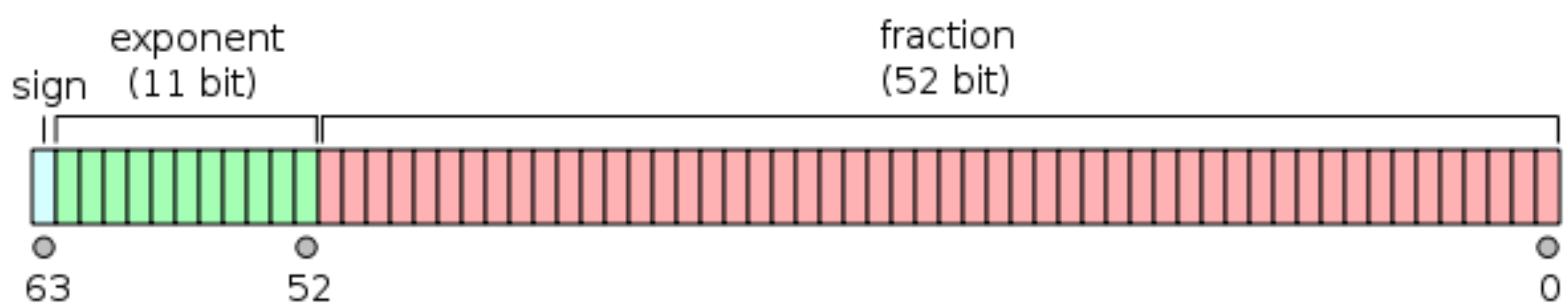

$2^{23}$ = 8M available numbers between 1 and 2,
**but** only 4k numbers between 2048 and 2049

# IEEE 754

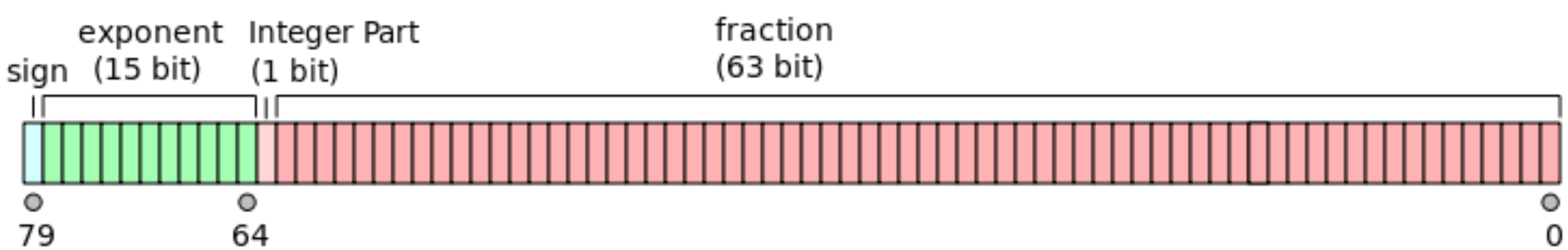
## Single-precision floating point (float, 4 bytes)




## Double-precision floating point (double, 8 bytes)




## Extended-precision floating point (long double, 10 bytes)



# Maths pitfalls

## Gaps in representation

$2^{23}$ = 8M available numbers between 1 and 2,
**but** only 4k numbers between 2048 and 2049

**Almost all** reals **cannot** be represented exactly

demo

# Maths pitfalls

## Gaps in representation

$2^{23}$ = 8M available numbers between 1 and 2,
**but** only 4k numbers between 2048 and 2049

## **Almost all** reals **cannot** be represented exactly

```
>>> 0.1+0.2
0.30000000000000004

>>> print '{:10.60f}'.format(0.1)
0.100000000000000005551115123125782702118158340454101562500000

>>> print '{:10.60f}'.format(0.2)
0.200000000000000011102230246251565404236316680908203125000000

>>> print '{:10.60f}'.format(0.3)
0.299999999999999988897769753748434595763683319091796875000000


>>> print '{:10.60f}'.format(0.1+0.2)
0.300000000000000044408920985006261616945266723632812500000000
```

# Maths pitfalls

## Gaps in representation

$2^{23}$ = 8M available numbers between 1 and 2,
**but** only 4k numbers between 2048 and 2049

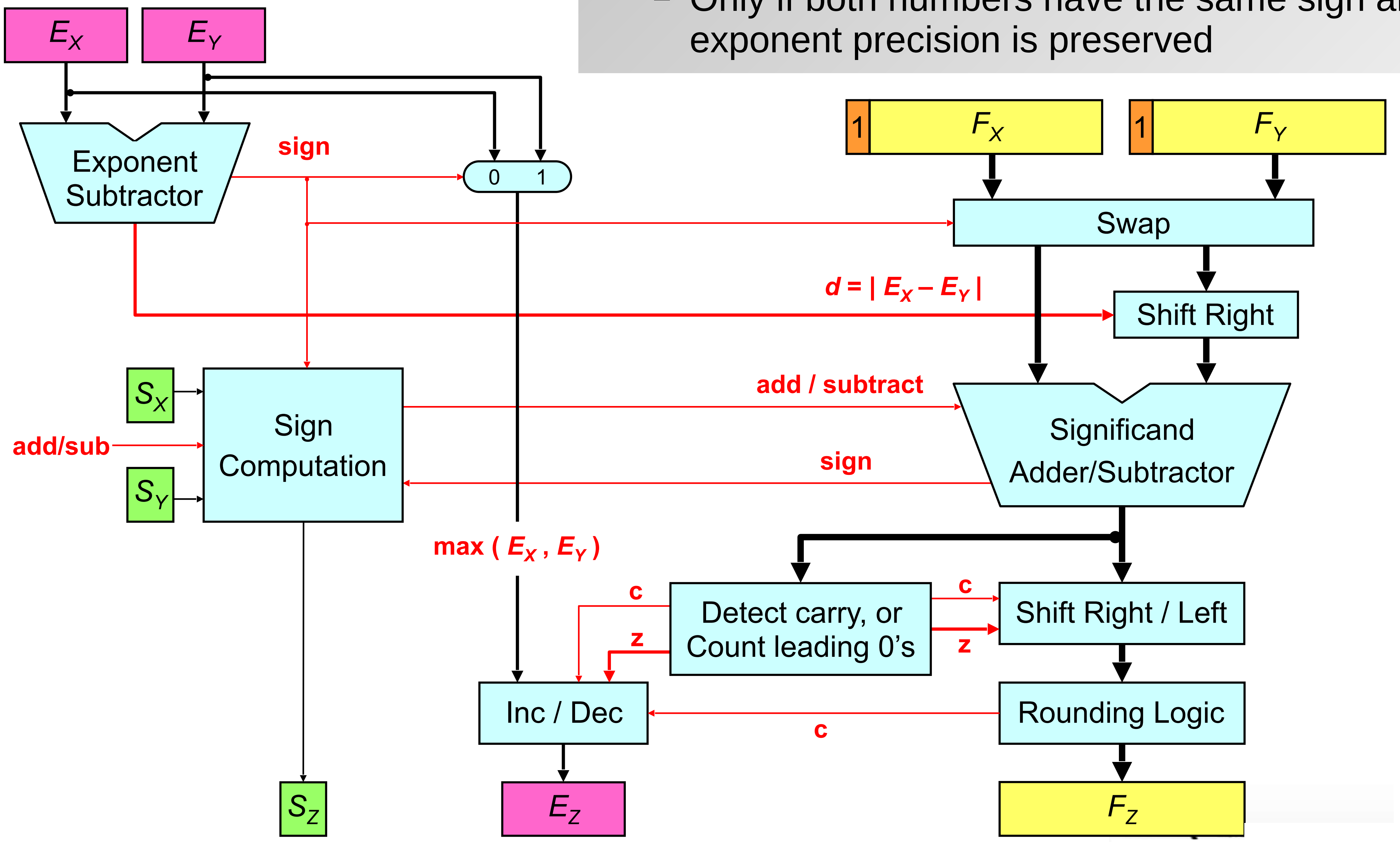
## **Almost all** reals **cannot** be represented exactly

```
>>> 0.1+0.2
0.30000000000000004

>>> print '{:10.60f}'.format(0.1)
0.1000000000000000055511151231257827021

>>> print '{:10.60f}'.format(0.2)
0.200000000000000011102230246251565404236316680908203125000000

>>> print '{:10.60f}'.format(0.3)
0.299999999999999988897769753748434595763683319091796875000000


>>> print '{:10.60f}'.format(0.1+0.2)
0.300000000000000044408920985006261616945266723632812500000000
```

```
>>> 0.1 + 0.2 == 0.3
False
```

# Addition:

- Right bitshift mantissa and increment exponent of smaller number until both exponents are the same
- Add mantissa of both numbers and bitshift until mantissa is between 1.0 and 2.0 again
- Only if both numbers have the same sign and the same exponent precision is preserved




$[\pm][1.\text{fraction}]\ 2^{[\text{exponer}}$

# Example:
## 8-bit number with 4-bit mantissa

| | | |
|---|---|---|
| 0.1 | $1.1001 \times 2^{-4} = 25/256$ | 0.09765625 |
| 0.2 | $1.1001 \times 2^{-3} = 25/128$ | 0.1953125 |

# Example:
## 8-bit number with 4-bit mantissa

| | | |
|---|---|---|
| 0.1 | $1.1001 \times 2^{-4} = 25/256$ | 0.09765625 |
| 0.2 | $1.1001 \times 2^{-3} = 25/128$ | 0.1953125 |
| 0.3 | $1.0011 \times 2^{-2} = 19/64$ $= 76/256$ | 0.296875 |

# Example:
## 8-bit number with 4-bit mantissa

| | | |
|---|---|---|
| 0.1 | $1.1001 \times 2^{-4} = 25/256$ | 0.09765625 |
| 0.2 | $1.1001 \times 2^{-3} = 25/128$ | 0.1953125 |
| 0.3 | $1.0011 \times 2^{-2} = 19/64$ $= 76/256$ | 0.296875 |

---

0.1+0.2

$$\begin{array}{r} 1.1001 \times 2^{-4} \\ +1.1001 \times 2^{-3} \\ \hline \end{array}$$

# Example:
## 8-bit number with 4-bit mantissa

| | | |
|---|---|---|
| 0.1 | $1.1001 \times 2^{-4} = 25/256$ | 0.09765625 |
| 0.2 | $1.1001 \times 2^{-3} = 25/128$ | 0.1953125 |
| 0.3 | $1.0011 \times 2^{-2} = 19/64$ $= 76/256$ | 0.296875 |

---

0.1+0.2

$$
\begin{array}{r}
0.1100 \times 2^{-3} \\
+1.1001 \times 2^{-3} \\
\hline
\end{array}
$$

# Example:
## 8-bit number with 4-bit mantissa

| | | | |
|---|---|---|---|
| 0.1 | $1.1001 \times 2^{-4} = 25/256$ | | 0.09765625 |
| 0.2 | $1.1001 \times 2^{-3} = 25/128$ | | 0.1953125 |
| 0.3 | $1.0011 \times 2^{-2} = 19/64$ | $= 76/256$ | 0.296875 |

---

0.1+0.2

$$
\begin{array}{r}
0.1100 \times 2^{-3} \\
+1.1001 \times 2^{-3} \\
\hline
10.0101 \times 2^{-3}
\end{array}
$$

# Example:
## 8-bit number with 4-bit mantissa

| | | | |
|---|---|---|---|
| 0.1 | $1.1001 \times 2^{-4} = 25/256$ | | 0.09765625 |
| 0.2 | $1.1001 \times 2^{-3} = 25/128$ | | 0.1953125 |
| 0.3 | $1.0011 \times 2^{-2} = 19/64$ | $= 76/256$ | 0.296875 |

---

0.1+0.2

$$
\begin{array}{r}
0.1100 \times 2^{-3} \\
+1.1001 \times 2^{-3} \\
\hline
1.0010 \times 2^{-2}
\end{array}
$$

# Example:

## 8-bit number with 4-bit mantissa

| | | |
|---|---|---|
| 0.1 | $1.1001 \times 2^{-4} = 25/256$ | 0.09765625 |
| 0.2 | $1.1001 \times 2^{-3} = 25/128$ | 0.1953125 |
| 0.3 | $1.0011 \times 2^{-2} = 19/64$ $= 76/256$ | 0.296875 |

---

0.1+0.2

$$\begin{array}{r} 0.1100 \times 2^{-3} \\ +1.1001 \times 2^{-3} \\ \hline 1.0010 \times 2^{-2} \end{array} = 18/64$$

0.28125

# Example:
## 8-bit number with 4-bit mantissa

| | | |
|---|---|---|
| 0.1 | $1.1001 \times 2^{-4} = 25/256$ | 0.09765625 |
| 0.2 | $1.1001 \times 2^{-3} = 25/128$ | 0.1953125 |
| 0.3 | $1.0011 \times 2^{-2} = 19/64$ <br> $= 76/256$ | 0.296875 |

---

0.1+0.2

$$
\begin{array}{r}
0.1100 \times 2^{-3} \\
+1.1001 \times 2^{-3} \\
\hline
1.0010 \times 2^{-2}
\end{array}
= 18/64 \qquad 0.28125
$$

Online IEEE calculator, e.g.: `http://weitz.de/ieee/`

# Maths pitfalls

## FP maths is commutative, **but not associative**

| Value1 | Value2 | Value3 | Value4 | Sum |
|---|---|---|---|---|
| 1.0E+30 | -1.0E+30 | 9.5 | -2.3 | 7.2 |
| 1.0E+30 | 9.5 | -1.0E+30 | -2.3 | -2.3 |
| 1.0E+30 | 9.5 | -2.3 | -1.0E+30 | 0 |

the result of a summation depends on the order of how the numbers are summed up

results may change significantly, if a compiler changes the order of operations for optimisation

prefer adding numbers of same magnitude

avoid subtracting very similar numbers

# Ill-conditioned matrices

Small changes lead to big steps in the solution.
Compare:

$$1.0000\,x + 1.0000\,y = 2.0000$$
$$1.0000\,x + 1.0001\,y = 2.0000$$

with:

$$1.0000\,x + 1.0000\,y = 2.0000$$
$$1.0000\,x + 1.0001\,y = 2.0001$$

# Ill-conditioned matrices

Small changes lead to big steps in the solution.
Compare:

$$1.0000\,x + 1.0000\,y = 2.0000$$
$$1.0000\,x + 1.0001\,y = 2.0000$$

$x$=2, $y$=0

with:

$$1.0000\,x + 1.0000\,y = 2.0000$$
$$1.0000\,x + 1.0001\,y = 2.0001$$

# Ill-conditioned matrices

Small changes lead to big steps in the solution.
Compare:

$$1.0000\,x + 1.0000\,y = 2.0000$$
$$1.0000\,x + 1.0001\,y = 2.0000$$

$x$=2, $y$=0

with:

$$1.0000\,x + 1.0000\,y = 2.0000$$
$$1.0000\,x + 1.0001\,y = 2.0001$$

$x$=1, $y$=1

# Ill-conditioned matrices

Small changes lead to big steps in the solution.
Compare:

$$\begin{aligned} 1.0000\,x + 1.0000\,y &= 2.0000 \\ 1.0000\,x + 1.0001\,y &= 2.0000 \end{aligned} \qquad x{=}2,\ y{=}0$$

with:

$$\begin{aligned} 1.0000\,x + 1.0000\,y &= 2.0000 \\ 1.0000\,x + 1.0001\,y &= 2.0001 \end{aligned} \qquad x{=}1,\ y{=}1$$

Danger if 2.0000 and 2.0001 look the same as Float!

$$\frac{x}{1000} + y = 1$$

$$x + y = 2$$

$$\frac{x}{1000} + y = 1$$

$$x + y = 2$$

$$x = \frac{1000}{999}$$

$$y = \frac{998}{999}$$

$$\frac{x}{1000} + y = 1$$

$$x + y = 2$$

$$x = \frac{1000}{999}$$

$$y = \frac{998}{999}$$

$$\frac{x}{1000} + y = 1$$

$$-999y = -998$$

$$\frac{x}{1000} + y = 1$$

$$x + y = 2$$

$$x = \frac{1000}{999}$$

$$y = \frac{998}{999}$$

$$\frac{x}{1000} + y = 1$$

$$-999y = -998$$

$$\frac{x}{1000} + y = 1$$

$$y = 1.00$$

$$\frac{x}{1000} + y = 1 \qquad x = \frac{1000}{999}$$

$$x + y = 2 \qquad y = \frac{998}{999}$$

$$\frac{x}{1000} + y = 1$$

$$-999y = -998$$

$$\frac{x}{1000} + y = 1$$

$$y = 1.00$$

$$\frac{999}{1000}y = \frac{998}{1000}$$

$$x + y = 2$$

The Abdus Salam International Centre for Theoretical Physics

$$\frac{x}{1000} + y = 1$$
$$x + y = 2$$

$$x = \frac{1000}{999}$$
$$y = \frac{998}{999}$$

$$\frac{x}{1000} + y = 1$$
$$-999y = -998$$

$$\frac{x}{1000} + y = 1$$
$$y = 1.00$$

$$\frac{999}{1000}y = \frac{998}{1000}$$
$$x + y = 2$$

$$1.00\,y = 1.00$$
$$x + y = 2$$

$$\frac{x}{1000} + y = 1 \qquad x = \frac{1000}{999}$$

$$x + y = 2 \qquad y = \frac{998}{999}$$

$$\frac{x}{1000} + y = 1$$

$$-999y = -998$$

$$\frac{x}{1000} + y = 1$$

$$y = 1.00$$

$$\boxed{x = 0}$$

$$\frac{999}{1000}y = \frac{998}{1000}$$

$$x + y = 2$$

$$1.00\,y = 1.00$$

$$x + y = 2$$

$$\frac{x}{1000} + y = 1 \qquad x = \frac{1000}{999}$$

$$x + y = 2 \qquad y = \frac{998}{999}$$

$$\frac{x}{1000} + y = 1$$

$$-999y = -998$$

$$\frac{x}{1000} + y = 1$$

$$y = 1.00$$

$$x = 0$$

$$\frac{999}{1000}y = \frac{998}{1000}$$

$$x + y = 2$$

$$1.00\,y = 1.00$$

$$x + y = 2$$

$$x = 1$$

$$\frac{x}{1000} + y = 1 \qquad x = \frac{1000}{999}$$

$$x + y = 2 \qquad y = \frac{998}{999}$$

$$\frac{x}{1000} + y = 1$$

$$-999y = -998$$

$$\frac{999}{1000}y = \frac{998}{1000}$$

$$x + y = 2$$

**Choice of algorithm matters!**

$x = 0$

$x = 1$

*Japan J. Indust. Appl. Math.*, 26 (2009), 249–277

**Inversion of Extremely Ill-Conditioned Matrices in Floating-Point**

Siegfried M. Rump

$$A_4 = \begin{pmatrix} -5046135670319638 & -3871391041510136 & -5206336348183639 & -6745986988231149 \\ -640032173419322 & 8694411469684959 & -564323984386760 & -2807912511823001 \\ -16935782447203334 & -18752427538303772 & -8188807358110413 & -14820968618548534 \\ -1069537498856711 & -14079150289610606 & 7074216604373039 & 7257960283978710 \end{pmatrix}$$

*Japan J. Indust. Appl. Math., 26 (2009), 249–277*

The Abdus Salam International Centre for Theoretical Physics

**Inversion of Extremely Ill-Conditioned Matrices in Floating-Point**

Siegfried M. RUMP

$$A_4 = \begin{pmatrix} -5046135670319638 & -3871391041510136 & -5206336348183639 & -6745986988231149 \\ -640032173419322 & 8694411469684959 & -564323984386760 & -2807912511823001 \\ -16935782447203334 & -18752427538303772 & -8188807358110413 & -14820968618548534 \\ -1069537498856711 & -14079150289610606 & 7074216604373039 & 7257960283978710 \end{pmatrix}$$

$$\mathtt{inv}_{\mathrm{fl}}(A_4) = \begin{pmatrix} -3.11 & -1.03 & 1.04 & -1.17 \\ 0.88 & 0.29 & -0.29 & 0.33 \\ -2.82 & -0.94 & 0.94 & -1.06 \\ 4.00 & 1.33 & -1.34 & 1.50 \end{pmatrix}$$

*Japan J. Indust. Appl. Math., 26 (2009), 249–277*

**Inversion of Extremely Ill-Conditioned Matrices in Floating-Point**

Siegfried M. Rump

$$A_4 = \begin{pmatrix} -5046135670319638 & -3871391041510136 & -5206336348183639 & -6745986988231149 \\ -640032173419322 & 8694411469684959 & -564323984386760 & -2807912511823001 \\ -16935782447203334 & -18752427538303772 & -8188807358110413 & -14820968618548534 \\ -1069537498856711 & -14079150289610606 & 7074216604373039 & 7257960283978710 \end{pmatrix}$$

$$\mathtt{inv}_{\mathrm{fl}}(A_4) = \begin{pmatrix} -3.11 & -1.03 & 1.04 & -1.17 \\ 0.88 & 0.29 & -0.29 & 0.33 \\ -2.82 & -0.94 & 0.94 & -1.06 \\ 4.00 & 1.33 & -1.34 & 1.50 \end{pmatrix}$$

$$\mathrm{fl}(A_4^{-1}) = \begin{pmatrix} 8.97 \cdot 10^{47} & 2.98 \cdot 10^{47} & -3.00 \cdot 10^{47} & 3.37 \cdot 10^{47} \\ -2.54 \cdot 10^{47} & -8.43 \cdot 10^{46} & 8.48 \cdot 10^{46} & -9.53 \cdot 10^{46} \\ 8.14 \cdot 10^{47} & 2.71 \cdot 10^{47} & -2.72 \cdot 10^{47} & 3.06 \cdot 10^{47} \\ -1.15 \cdot 10^{48} & -3.84 \cdot 10^{47} & 3.85 \cdot 10^{47} & -4.33 \cdot 10^{47} \end{pmatrix}$$