“To what extent can we rely on the results of scientific computations ?"

ACAT ’07, Amsterdam, The Netherlands

Les Hatton

Professor of Forensic Software Engineering
Kingston University, UK
L.Hatton@kingston.ac.uk, lesh@leshatton.org

Version 1.1: 20/Apr/2007

Overview

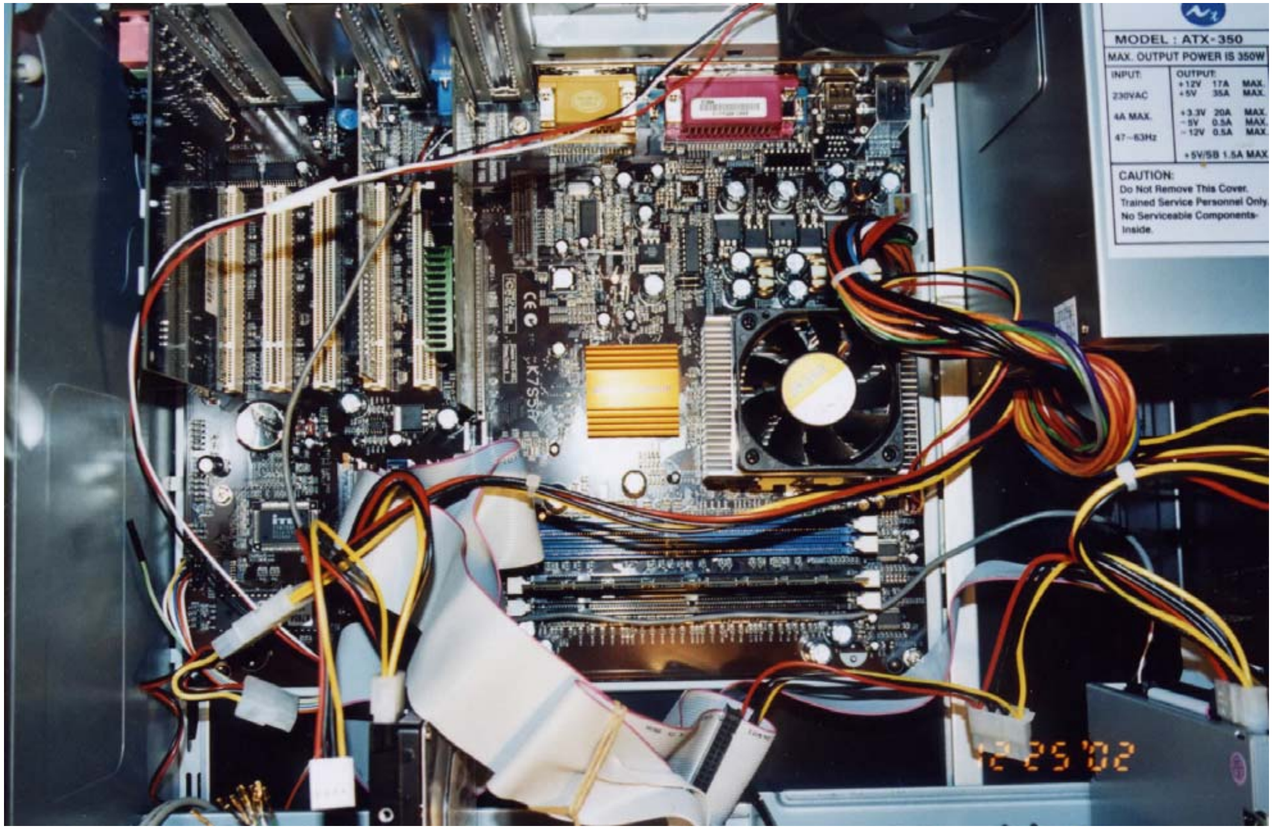
We are doing pretty well with hardware ...

1984

Mine 8-)

Its big brother :-(

2005



Build your own .. http://www.leshatton.org/lxf37_pc.html

Old and new

1984

- Cray X MP, 0.5 sec.

2005

- Hand-built 200 quid PC with bits from a computer fair, 0.044 sec.

If you have a Fortran compiler, feel free to download the benchmark from:-

http://www.leshatton.org/FB_885.html

Overview

So how are we doing with software ...

In at the deep end ... 1974

$$\frac{\partial \vec{u}}{\partial t} + \underset{\uparrow}{\left(\vec{u}.\nabla\right)}\vec{u} = -\frac{1}{\rho}\nabla p + \nu\nabla^2\vec{u}$$

UK Meteorological Office standard 10-level numerical weather prediction model. This term was dropped every other time step due to a software defect.

Reinstating it led to almost no difference in a 72 hour forecast !

Overview

- *Sources of error in numerical modelling*
- The unpleasant nature of software defect
- What can we do about it ?

Sources of error in numerical modelling

- Precision problems
- Algorithmic problems
- Data problems
- Software problems

Scientists are used to dealing with the first three but not the last ...

Overview

- Sources of error in numerical modelling
- *The unpleasant nature of software defect*
- What can we do about it ?

The unpleasant nature of software defect ...

- Its inevitable – failure is a natural property of an engineering system
- Its unquantifiable so its easy to get misleading results without realising
- Defects can take a very long time to appear for the first time
- All numerical results derived using software are contaminated at some level.

Its inevitable - “Zero defect”

- Some comments on the chimera of zero defect
  - The chance of achieving it is vanishingly small
  - If you ever succeed you won’t know it
  - If you ever succeed you won’t be able to prove it
  - If you ever succeed you won’t be able to repeat it

So how many defects do we have ?

- If you count all faults that failed and you have < 1 per 1000 executable lines of code (KXLOC) in the entire life-cycle of the system you are doing almost as well as anybody ever has.
- By this measure, the best ever is around 0.1, (NASA Shuttle software and the Linux kernel)
- Typical reasonable systems lie in the range 2-10 per KXLOC.

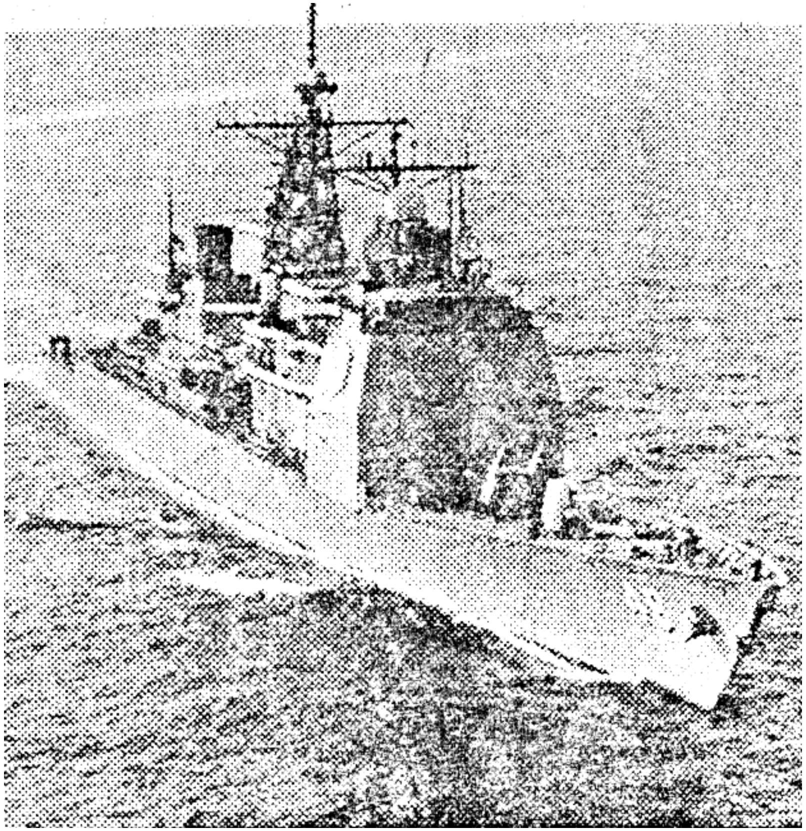
So how much code do we have ?

The amount of software in consumer electronic products is currently doubling about every 18 months.

- Line-scan TVs have ~500,000 lines of C.
- There are around 1-3 million lines of C in a car.
- The F/A-22 (Raptor) fighter has around 2 million lines of code.
- The Airbus A340 and Boeing 777 have 3-4 million lines of code, (more later...).
- *A reasonably small scientific computation might have 100,000 lines of code.*

Not a good decade:

The USS Yorktown in “Please wait …” mode



Not a good decade:

An Airbus having a bad day

A Tarom airlines Airbus which performed an uncontrolled dive, climb, roll and spin near Orly in 1995 due to ‘a fault in the automatic pilot’. The plane landed safely, a tribute to the pilots’ skill.

Not a good decade:

Ariane 5: What goes up …

Smoke from the explosion
June 4, 1996 (AP Photo)

Whoops …

More avionics …

- 28/Jul/2003. “As recently as February, test pilots of the new F/A-22 (Raptor) fighter were spending an average of 14 minutes per flight rebooting critical systems. This is now down to only 36 seconds per flight.

  Washington Post.

Whoops …

Safety is not the same as Reliability

- **Thomas the friendly torpedo …**

Whoops …

Cars …:

- 14/Apr/2004. Ford is recalling 363,440 of its 2001-2003 Ford Escape vehicles due to software problems in power-train causing engine stalling.

  Detroit News

- 17/Mar/04. 2003 US vehicle recalls hit 19.5 million in spite of 'engineering never being better'. Experts cite problem-prone computers as significant factor.
- 09/Mar/04, Toyota faces US safety investigation and potential recall of 1 million of its best-selling Camry and Lexus ES300 sedans because of reports of unexpected acceleration causing 30 crashes.

  Detroit Free Press

OS Reliability

- 24.5 million XP crashes per day

http://www.pcmag.com/article2/0,4149,1210067,00.asp

- 5% of Windows Computers crash more than twice a day

http://www.nytimes.com/2003/07/25/technology/25SOFT.html

Its unquantifiable ...

10/April/2006: Malaysian man gets $218 trillion phone bill. (Associated Press).

- Telekom Malaysia gave 10 days to pay. They later decided it was “a little excessive”.

What is excessive and how would we know ?

Subtle errors,
(the T2 experiment, 1990-1994)

Lets ask a simple question:-

"If different sets of programmers program the same algorithms in the same programming language and feed them the same data with the same disposable parameters …

How well do the results agree ?"

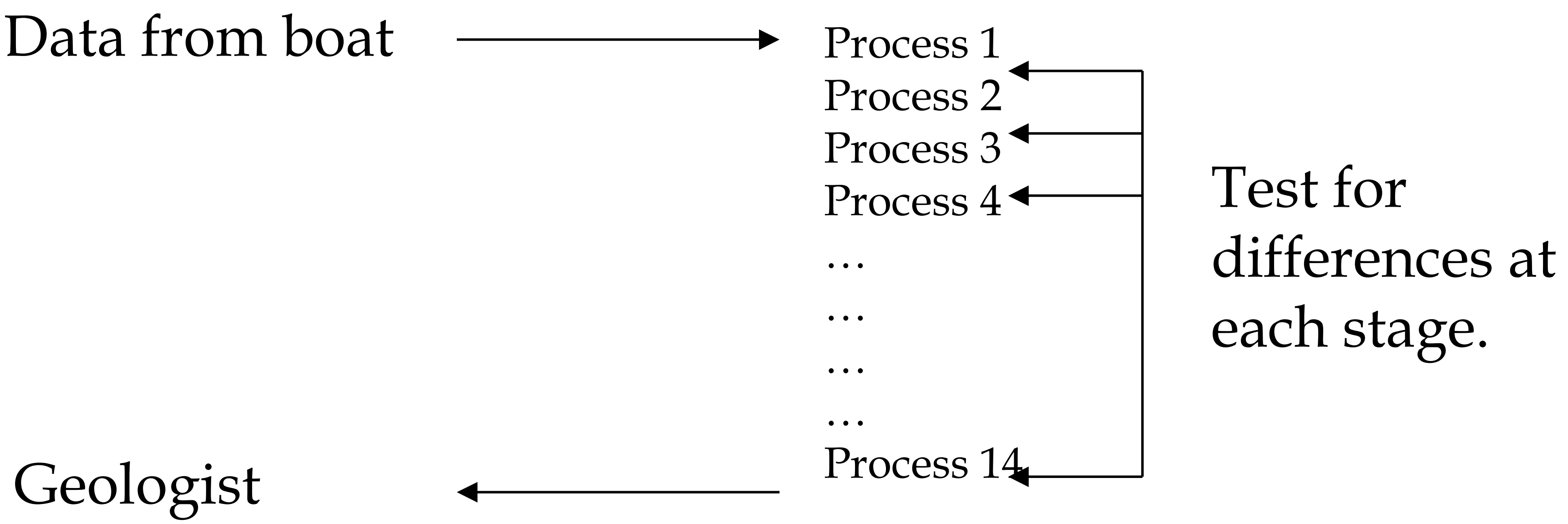
As a counterpoint, lets imagine that people drill $30 million oil wells on the results :-)

How seismic data are acquired

Boats like these acquire between 1 and 5 Mb per second for several weeks.

A typical survey might contain 5 Terabytes.

How seismic data are processed and how the experiment was done



Data from boat

Process 1

Process 2

Process 3

Process 4

…

…

…

…

Process 14

Geologist

Test for differences at each stage.

Algorithms used

- As well as some bespoke algorithms, the following are used regularly ...
  - Multi-dimensional Fourier Transforms
  - Multi-dimensions Wiener filtering
  - Inversion of very large sparse matrices
  - Inversion of scalar wave equation
  - Various kinds of statistical correlation algorithms
  - ... lots more

At the time of this experiment, the principle language used was Fortran.

Similarity v. coordinate: No feedback

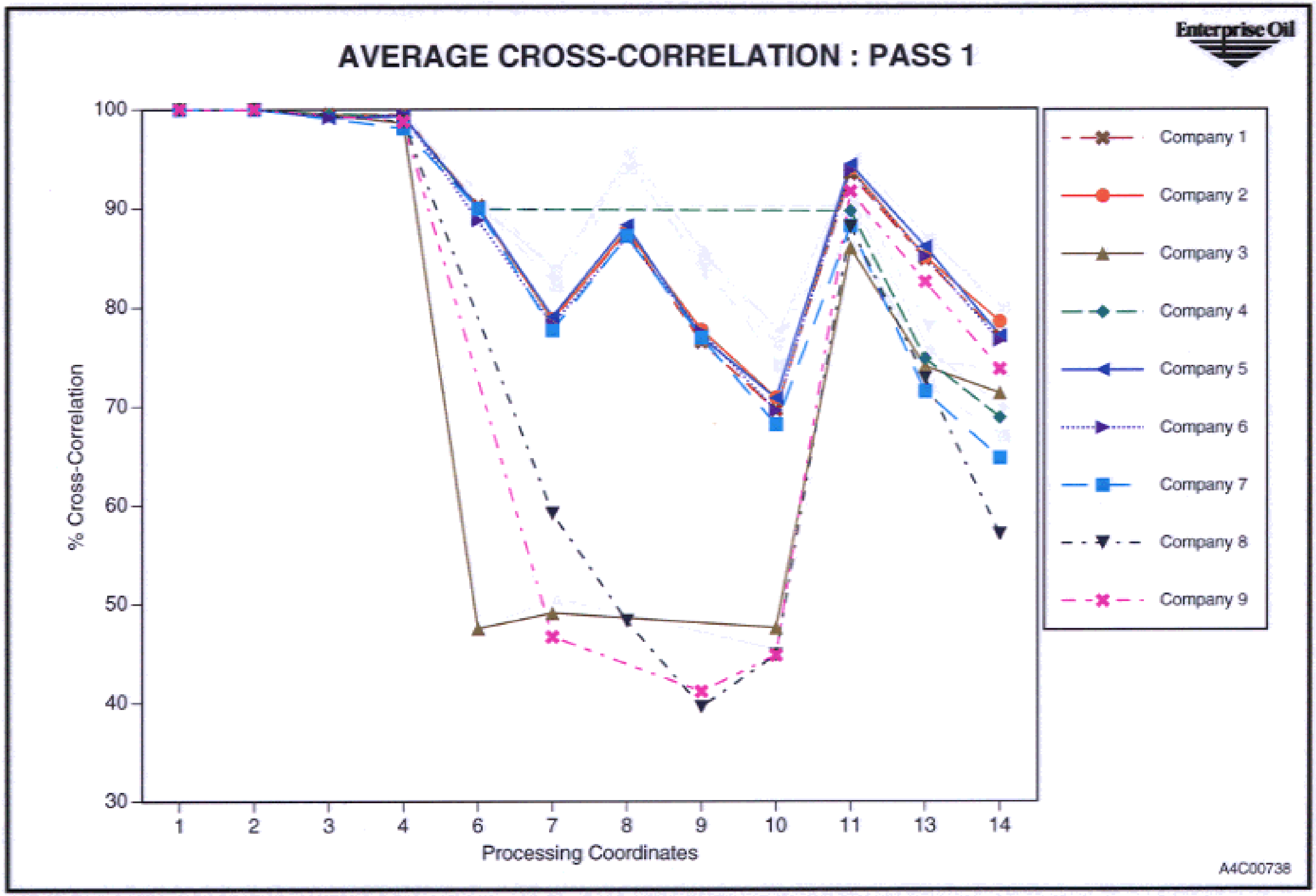
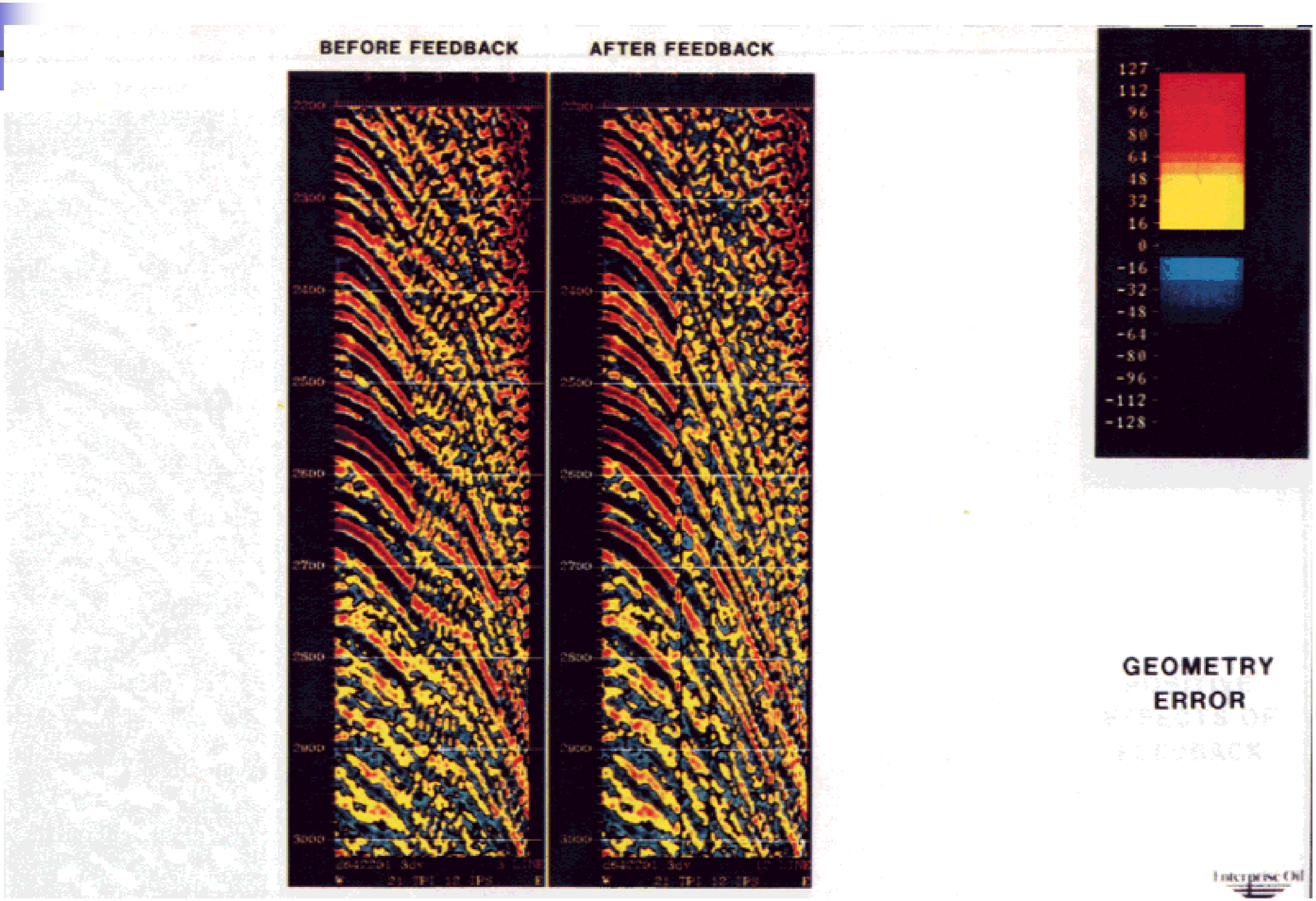
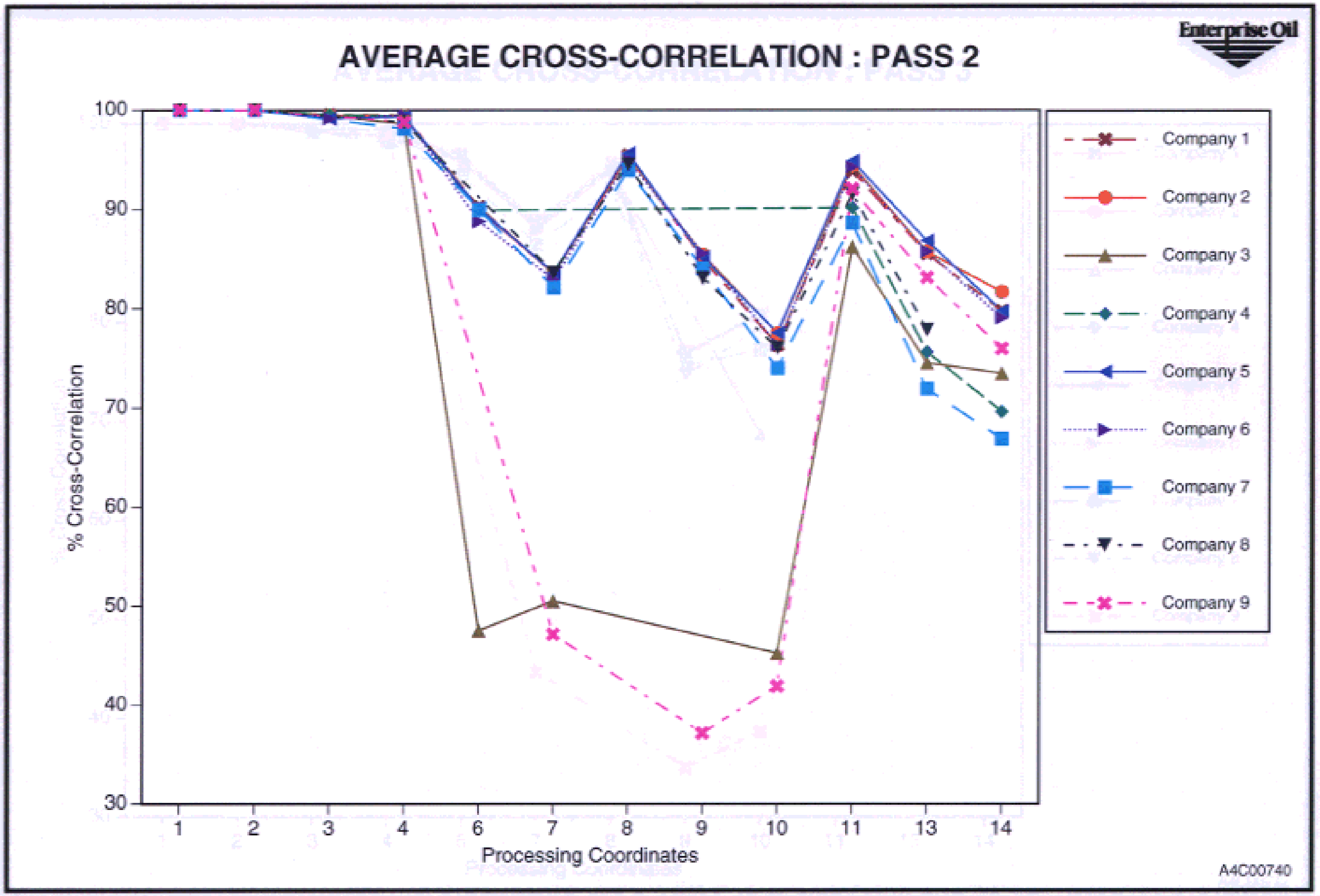


Figure 2.1

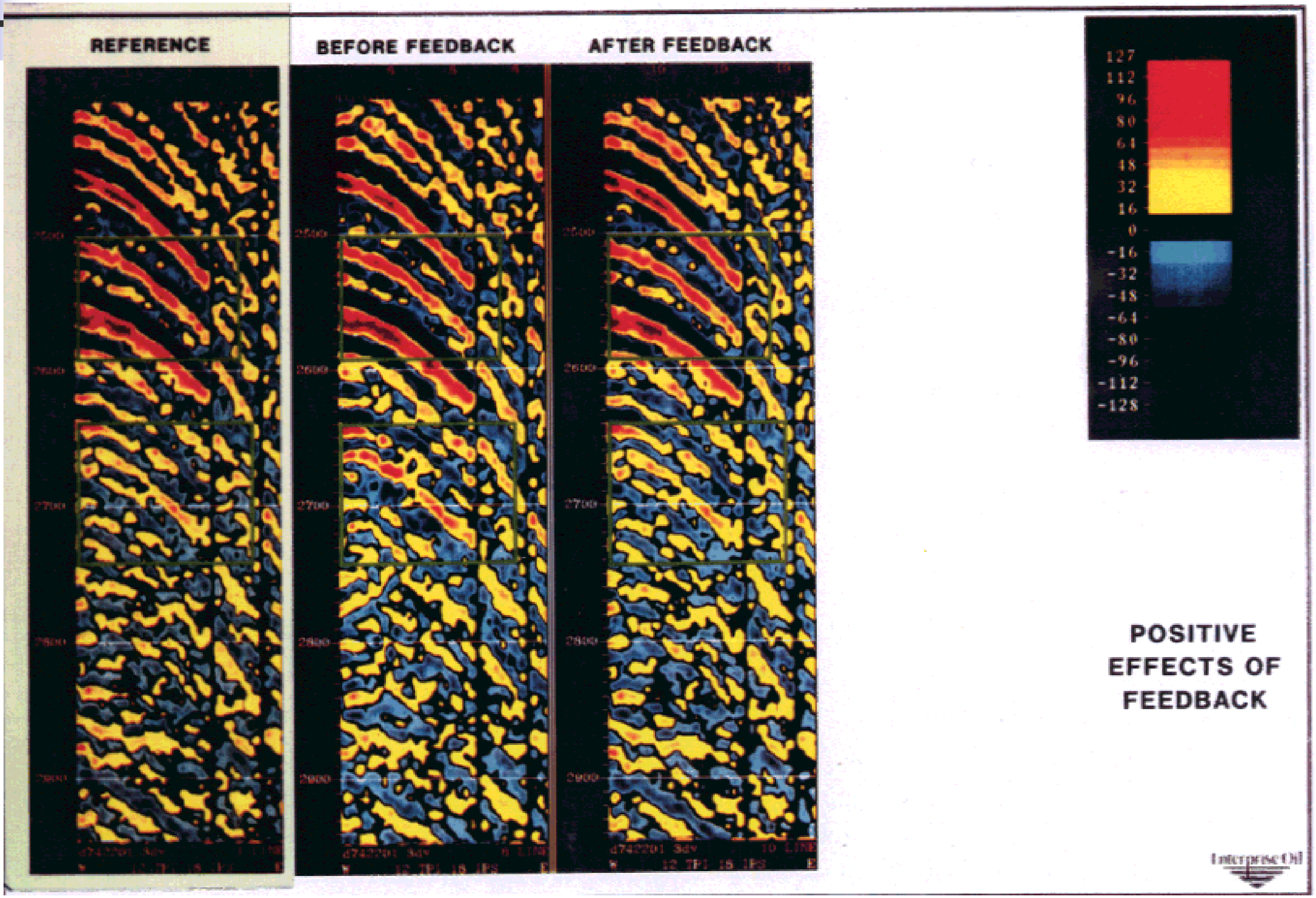
Defect example 1: feedback detail



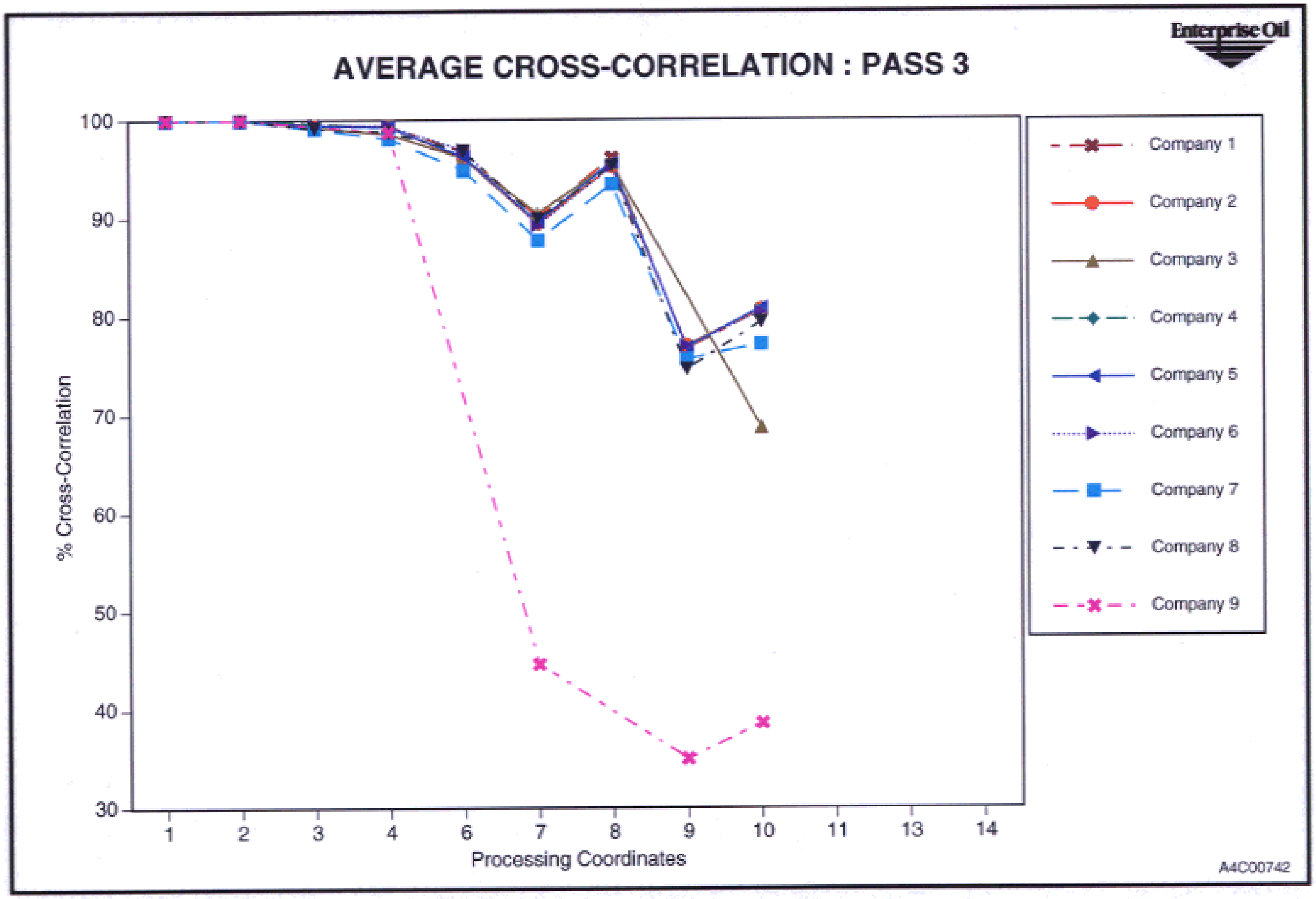
Similarity v. coordinate: Feedback to company 8



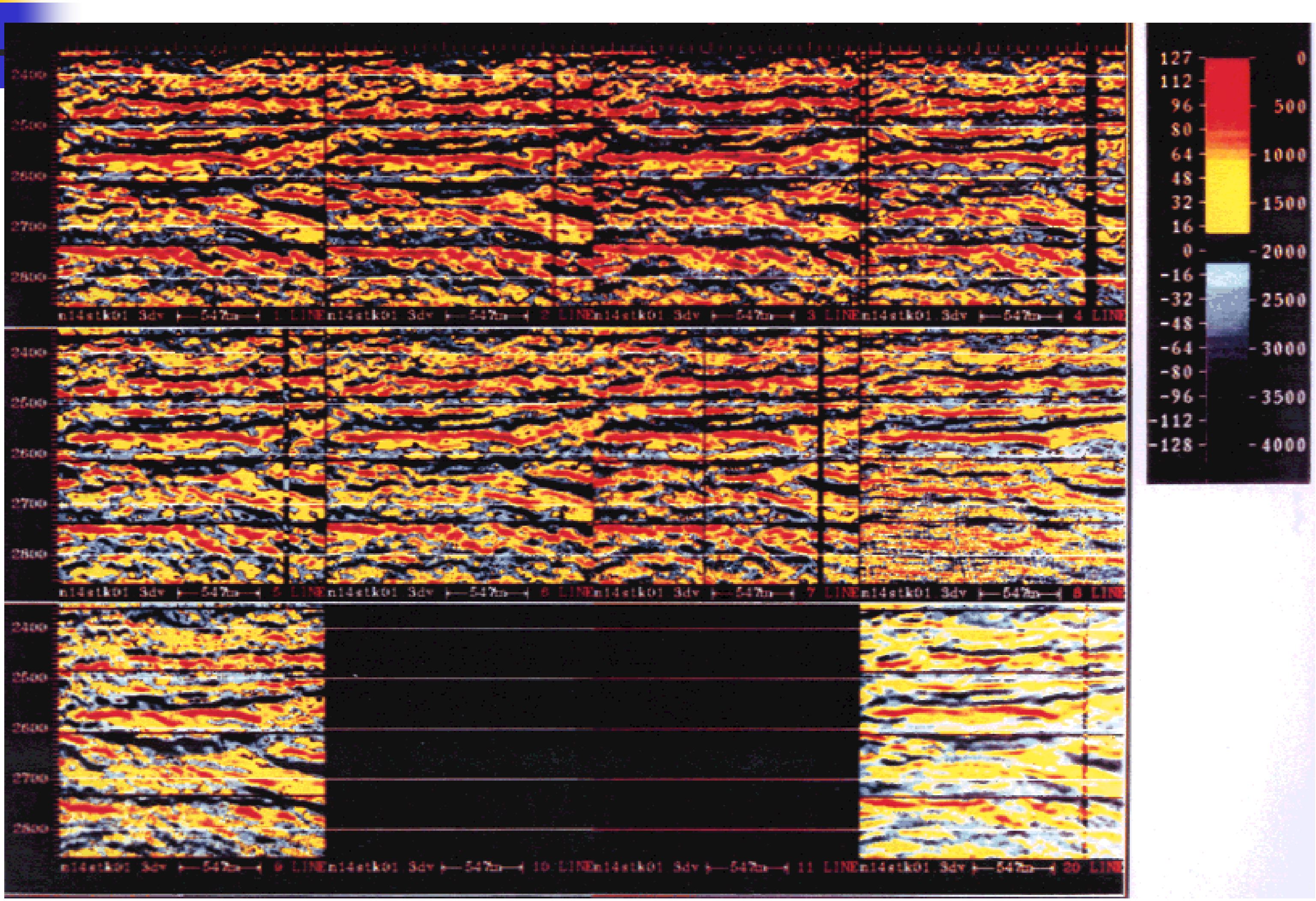
Defect example 2: feedback detail



Similarity v. coordinate: Feedback to company 3



The end product: 9 subtly different views of the geology



T2 Results

- The accompanying slides illustrate:
  - Only 1-2 significant figures agreement after processing.
  - Disagreement is non-random and alternate views seem equally plausible
  - Feedback of anomalies along with other evidence confirms source of disagreement as software failure.

A summary of 10 years of failure experiments

Seismic processing software environment	Number of significant figures agreement
32 bit floating point arithmetic.	6
Same software on different platforms, same data.	4
Same software on same platform, 5-1 lossy compression.	3-4
Same software subjected to continual 'enhancement'	1-2
T2: different software, same specs, same data, same language, same parameters.	1

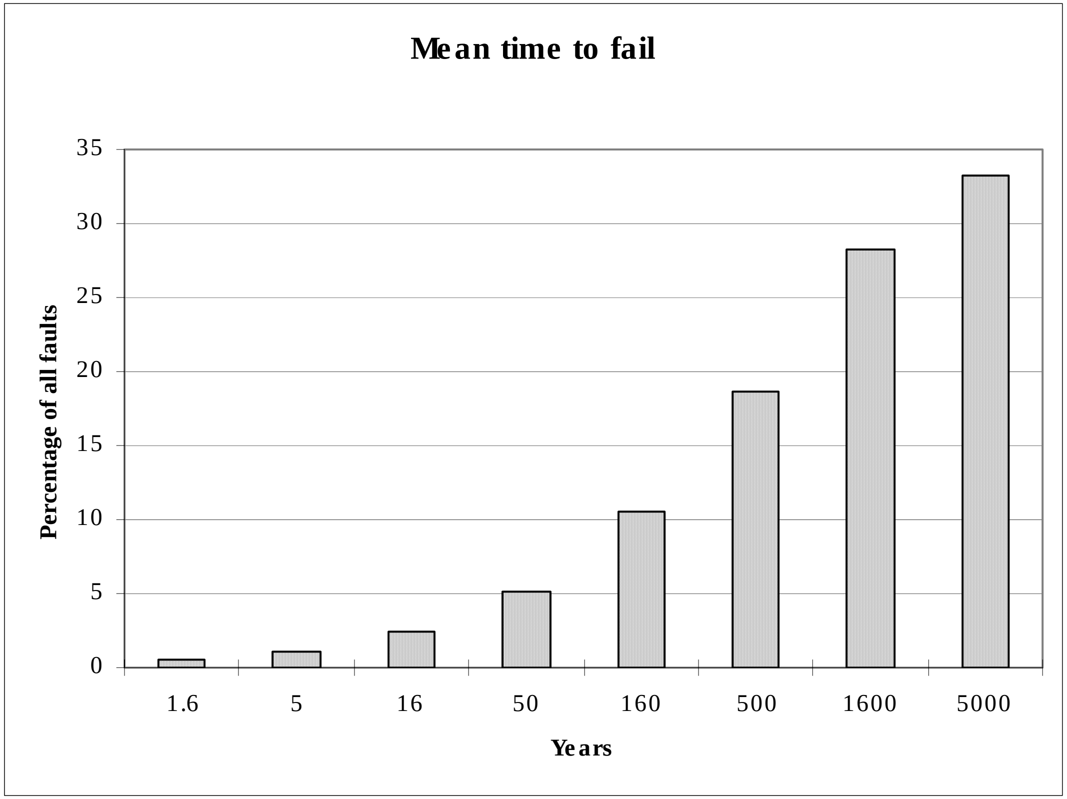
Portability degradation

Compression degradation

Maintenance degradation

Diversity degradation

Defects can take a very long time to appear for the first time, (Adams 1984)



All numerical results derived from software are contaminated ...

Define ...

- *Static code fault*: property of computer program likely to fail under some circumstances
- *Dynamic failure:* any difference between the actual and expected behaviour at run-time

If we can find a statistically significant connection between these, we could predict the likelihood of the presence of failure from the source code alone

All numerical results derived from software are contaminated …

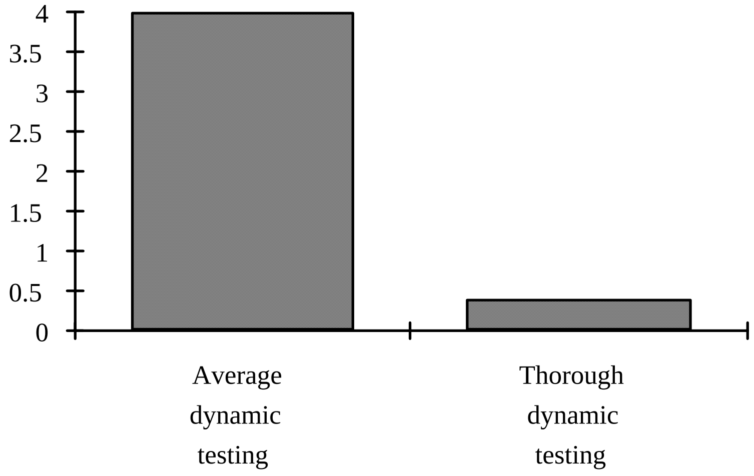
- Find a suitable static code measure which is highly correlated to failures in known cases
- Use this to predict likely presence of failure in unknown cases from the code only.

Do statically detectable faults fail ?

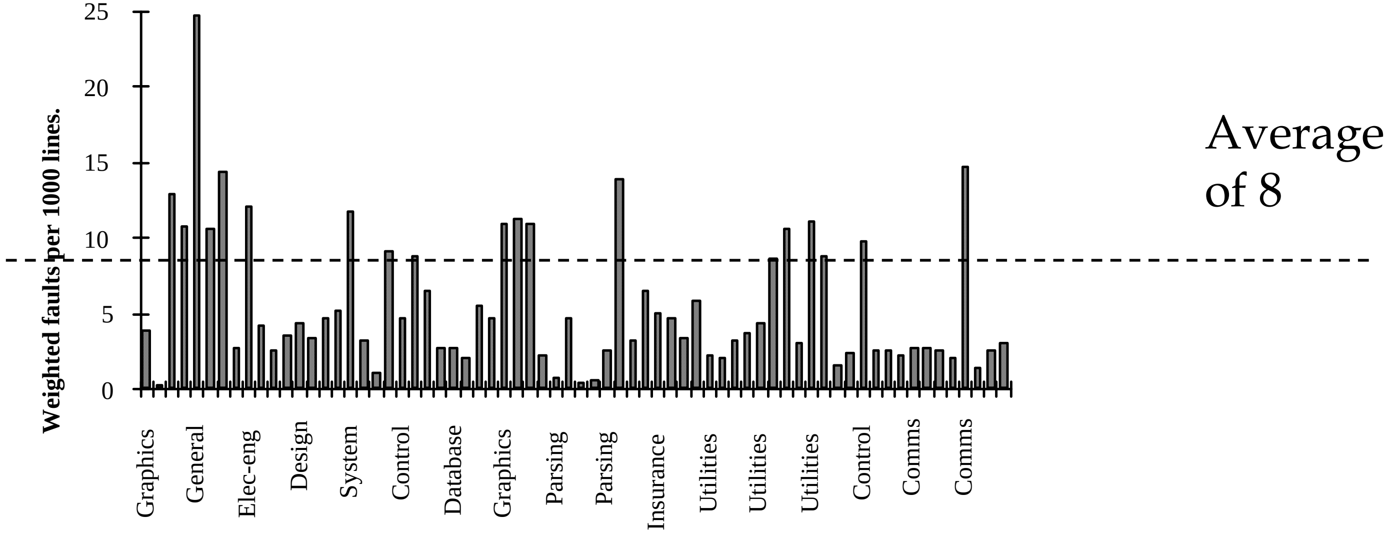
The selected fault type is the occurrence rate of mistakes with the programming language.

The faults are highly correlated with dynamic failures

Data derived from CAA CDIS



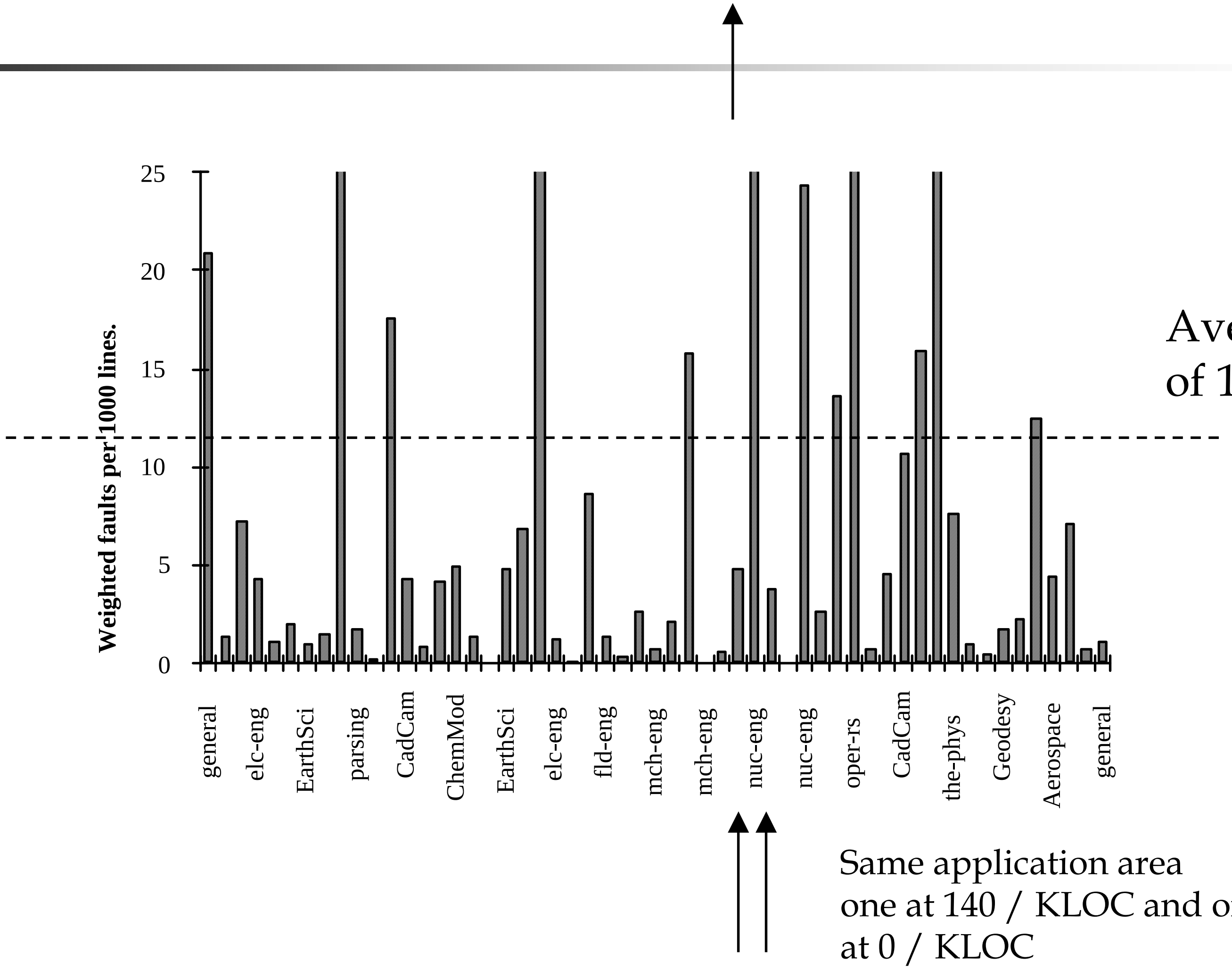
Fault frequencies in C applications



Average of 8

Survey: 1993-1998

Fault frequencies in Fortran 77 applications



Average of 12

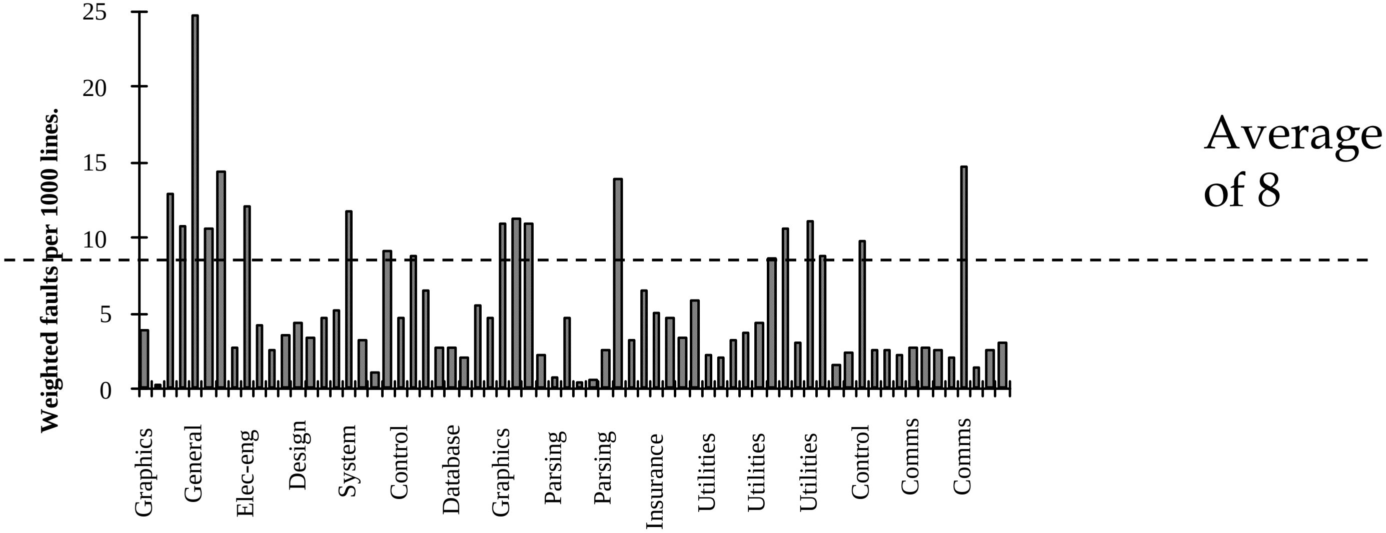
Same application area one at 140 / KLOC and one at 0 / KLOC

Fault frequencies in C applications - revisited

Recent examples:
Netscape Javascript Interpreter, 2003
14.78 per KSLOC

F1 racing car software 2003
13.47 per KSLOC

Government agency, 2005
0 per KSLOC



Average of 8

Survey: 1993-1998

Note:

- Software fails frequently. When it does it is sometimes impossible to fix
- Software failure is highly unpredictable
- It doesn’t really matter which programming language you use
- Software development is immature and little progress has been made in reliability in the last 25-30 years
- Many software failures can take an astonishingly long time to appear for the first time
- New bespoke projects have a very low success rate
- We have no technology to guarantee the absence of defect
- The cost of failure is limited only by the imagination
- We have an educational problem not a technology problem.

Overview

- Sources of error in numerical modelling
- The unpleasant nature of software defect
- *What can we do about it ?*

What can we do about it ?

- Do not use other people’s code
  - Use every opportunity for independent verification
  - Try different languages and compilers

What can we do about it ?

- The role of open source
  - Open source appears to get incrementally more reliable amongst other things.

What can we do about it ?

- The computable paper (Claerbout and collaborators)
  - Any scientific paper involving computation should publish:-
    - The science for peer review
    - The code for peer review
    - The environment in sufficient detail for repeatability
  - There is an example at:-
    - http://www.leshatton.org/NS_03.html

Other information

For more information and downloadable papers see:-

http://www.leshatton.org/